Altair Knowledge Studio® 2022.3

User Guide

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Table of Contents

Introduction 11
Summary of New Features 14
Version Compatibility 23
Technical Information and Client Support 24
  Hardware and Operating System Requirements 24
  Technical Support 27
  Copyright Information 27
License Administration 33
  License Types 33
  Altair Units Licensing 34
    Configuring Altair Units Licensing 34
    Authorize Machine for Managed Altair Units License 42
  Registration-Key Licensing 46
    Altair License Fulfillment Server (ALFS) 46
    Registration Keys 46
    Prerequisites for Installing Registration-Key Licenses 47
    License Manager for Registration-Key Licenses 48
Connecting to the Data Mining Engine 60
  Connecting to the Data Mining Engine 60
  Setting the Working Directory 62
Projects and Workflows 65
  Projects 65
  Creating and Deleting Projects 68
  Workflows 68
  Project Documents in Client-Server Mode 85
  Sample Projects 86
    Prepare Sample Data 86
    Data Definition for the Credit Example Dataset 88
  Task Scheduler 91
Data Source Connections, Data Import and Export 98
  Accessing Data Sources 98
  Dataset Link 104
  File Import 110
    Select Source 110
    Text File Preview 118
    Fixed Width Editor 119
    Text Delimiter 119
    Field Selector 120
    Properties and Preview 124
  Folder Import 135

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Import from ODBC Data Sources 147
Import from ODBC data sources 147
ODBC Connection Setup 148
Table Selection 152
Dataset Name 154
Field Selector 157
Properties and Preview 160
In-Database Analytics 171
Data Export 175
Overview 175
File Export 177
Overview 177
Text File Format Options 179
Export to Databases (ODBC) 180
ODBC Export Node 180
ODBC Connection Setup 181
Field Selection 185
CREATE TABLE Statement 187
Export Dataset to Monarch DPS 188
Desktop Document and Report Links 191
Data Manipulation 196
Overview 196
Dataset Operations 197
Aggregating Data 197
Appending Datasets 201
Removing Duplicate Records 208
Filter 211
Filtering Join 222
Joining Datasets 225
Make Time Series 234
Partitioning 241
Overview 241
Defining Partitions 242
Partition: Summary Report 250
Pivot Table 254
Sort, Rank, Lag 258
Auto Feature Engineering 265
Variable Transformations 274
Variable Transformations Node 274
Expression Editor 282
Expression Editor: Overview 282
Functions and Operators 286

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Division and Infinity 288
Specifying Format 291
Function Syntax 294
Expression Helpers 340
Helper: Binning 342
Helper: Date and Time Join 345
Helper: Date and Time Split 350
Helper: Substitute Missing Values 355
Helper: Generation of Dummy Variables 355
Helper: Standardization 357
Helper: Outliers Clipper 362
Helper: Weight 363
Helper: Log Transform 367
Helper: Power Transform 367
Helper: Text Join 368
Helper: Text Split 373
Optimal Binning 379
Helper: Optimal Binning 379
Expression Generation Options 382
Select Dependent Variable Value 383
View or Edit Bins 384
View Resulting Expressions 387
Helper: Interaction Terms 388
Transforming Multiple Variables 390
Defining Special Values 391
Import Expressions 397
Export Expressions 399
Substitute Missing Values 400
Handle Class Imbalance 404
WOE Transformation 410
WOE Template 421
Data Profiling 426
Dataset Views: Summary 426
Weight 427
Overview Report 429
Customizing the Overview Report 435
Dataset Chart 436
Edit Ranges 437
Segment Viewer 440
Segment Viewer Options 441
Segment Viewer Charts 442
Segment Viewer: Advanced Options 443

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Copying the Charts 444
Cross Tabs 445
Advanced Options 448
Cross Tabs: Chart 451
Cross Tabs: Report 453
Characteristic Analysis 454
Correlations 458
Correlation Options 459
Correlations View 461
Saved Charts 463
Single Chart View 467
Interactive Filtering on Datasets 472
Outlier Detector 479
Measure of Predictive Power 493
Variable Importance 498
Multicollinearity Analysis 502
Variable Selection 507
Simpson's Paradox Detection 510
Schema 514
Autocorrelation (Dataset Tab) 525
Modeling 531
Model Instances 531
Decision Trees 534
What is a Decision Tree 534
What is a Split 535
Inserting a Decision Tree 537
Select Dependent Variable 538
Training Parameters 540
Measure 548
Split Report 552
Auto Grow Parameters 553
Creating and Modifying Splits 555
Find Split 555
Next Split 559
Previous Split 559
Go To Split 560
Automatic Grow 561
Force Split 563
Choosing the split variable 565
Range Editor for Discrete Variables 567
Range Editor for Continuous Variables 571
Ignore Variable 573

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Erase Child Nodes 574
Editing Splits 575
Editing Split for Continuous Variable 581
Editing Split for Discrete Variable 583
Force/Edit Split with Requested DV Percentage 586
Saving Changes 590
Viewing a Decision Tree 591
Node Chart 614
Single Chart View 625
Copying Splits and Trees 630
Modifying Tree Attributes 639
Using the Attribute Editor 639
Weight in Decision Trees 645
Variable Usage Types 647
Default Number of Bins 648
Accuracy Estimate: Resubstitution 654
Creating Calculation from Split 655
Ensemble Trees 660
Bagging (Bootstrap Aggregating) 662
Random Forest 668
Boosting 674
Predictive Models 680
Weight in Predictive Models 684
Linear Regression Model 687
Generalized Linear Model 700
Logistic Regression Model 708
Constrained Regression 722
Regularization 725
PLS Regression Model 729
Deep Learning (Neural Network) 732
Deep Learning 732
Inserting a Deep Learning Model 737
Keras Deep Learning 747
Keras Deep Learning Scoring 755
Keras Deep Learning Validation 762
Reject Inference 768
Overview 768
Reject Inference Wizard 771
Reject Inference Results 786
Sklearn XGBoost Model 790
Model Attribute Editor 801
Scorecards 807

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

About Scorecards 807
Inserting a Scorecard Model 808
Scorecard Editor 813
Model Selector 819
Model Stacking 823
Novelty & Outlier Detector 836
AutoML 847
Insert Model from XML/PMML 867
Strategy Trees 869
Strategy Trees: Overview 869
Inserting a Strategy Tree 871
Viewing a Strategy Tree 873
Single Chart View 882
Adding Node Calculations 887
Expression Editor 889
Expression Helpers 894
Treatments 911
Managing Treatments 912
Generating Treatments From Calculations 915
Treatment Report 917
Creating and Modifying Splits 917
Find Split 917
Force Split 919
Next / Previous / Go To Split / Copy & Paste Splits 921
Edit Split 923
Erase Child Nodes / Ignore Variable 928
Modifying Tree Attributes 929
Using the Attribute Editor 929
Validating and Scoring a Strategy 940
Strategy Validation 940
Strategy Scoring 950
Optimization in Strategy Trees 959
Example Workflow 959
Cluster Analysis 969
Inserting Cluster Analysis 970
Model Attribute Editor 976
Viewing a Cluster Model 983
Market Basket Analysis 988
Overview 988
Inserting Market Basket Analysis 990
Market Basket Analysis Model Views 993
Principal Component Analysis 1005

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Inserting Principal Component Analysis and Viewing Results 1006
Exporting Calculations From PCA and Factor Analysis 1011
Scoring a Principal Component Analysis Model 1013
Factor Analysis 1019
Inserting Factor Analysis 1019
Exporting Calculations From PCA and Factor Analysis 1024
Scoring a Factor Analysis Model 1026
Time Series 1032
Time Series Decomposition 1032
Time Series Decomposition Plots 1038
Keras Forecasting 1042
ARIMA Forecasting 1052
Model Evaluation 1064
Model Validation 1064
Keras Deep Learning Validation 1078
Sklearn GLM Validation 1085
Sklearn XGB Validation 1092
Cross Validation 1102
Stability & Characteristic Reports 1107
Variable Importance 1112
Explainable AI 1116
Model Analyzer 1133
Models Analyzer Views 1136
Cumulative Lift 1136
Lift 1141
K-S Chart 1144
ROC Chart 1146
Goodness of Fit (GOF) 1149
Profit Curve 1153
Model Selector 1154
Model Deployment 1159
Scoring 1159
Scoring Wizard for Python-based Models 1173
Novelty & Outlier Scoring 1173
Keras Deep Learning Scoring 1179
Keras Forecasting Scoring 1186
Scoring a GLM Model 1192
Scoring an XGBoost Model 1199
ARIMA Scoring 1205
Deploying Association Rules 1211
Code Generation 1222
Upload Model to SmartWorks Analytics 1254

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Upload Model to Real-Time Scoring Service....1256
Optimization....1258
Simple Budget Optimization....1258
Simple Record-Level Optimization....1262
Record-Level Optimization....1267
Dataset-Level Optimization....1273
Multiple Scenario Optimization....1277
Copy, Export, and Import Problem Formulations and Variables....1284
Task Scheduler....1288
Integration using SAS Language....1295
Integration with R and Python....1318
R Code Node....1318
Survival Analysis....1330
Python Code Node....1336
Custom Library Palette....1354
Keras Forecasting....1367
Keras Deep Learning Scoring....1378
Keras Deep Learning Validation....1384
Auto Feature Engineering....1391
Generalized Linear Model....1400
ARIMA Forecasting....1408
ARIMA Scoring....1419
Generalized Linear Model....1423
Sklearn GLM Validation....1431
Scoring a GLM Model....1438
Novelty & Outlier Detector....1445
Novelty & Outlier Scoring....1455
Sklearn XGBoost Model....1461
Scoring an XGBoost Model....1472
Sklearn XGB Validation....1478
Integration with Tableau and Qlik....1486
Integration with Tableau....1486
Creating a Tableau Workbook....1487
Adding Tableau Workbooks and Templates to Workflow....1493
Publishing a Tableau Workbook....1494
Downloading Tableau Templates....1497
Integration with Qlik....1501
Creating a Qlik File....1502
Selecting and Reusing Variables with Variable Lists....1508
Variable List....1508
Field Finder....1517
Options and Preferences....1523

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Auto-Documentation, Logging, and Command Line Tools 1556
Desktop Document and Report Links 1556
Self Documentation 1559
Model Documentation 1564
Activity Log and Code Logs 1569
KSQuery Command Line Interface 1573
DMX Code Generation 1576
Find and Replace Text 1587
Printing 1588
Mathematical Formulations 1593
Decision Tree 1593
Predictive Models: Regression 1605
Deep Learning (Multi-layer Neural Network) 1624
Cluster Analysis Models 1629
Principal Component Analysis 1635
References 1639
Index 1642

All rights reserved. No part of this document may be reproduced or transmitted in any form or by any means without prior written permission of Altair Engineering, Inc. (Altair). KnowledgeSEEKER, Knowledge Seeker, KnowledgeSTUDIO, Knowledge Studio, KnowledgeREADER, KnowledgeMANAGER, and InsightOPTIMIZER, are service marks, trademarks, or registered trademarks of Altair. All other products, logos, or brand names are either service marks, trademarks, or registered trademarks of their respective owners. Altair data mining technology is protected under Canadian, US and international copyright laws and applications have been filed under Canadian, US and international patent laws.

Introduction

Welcome to version 2022.3 of ***Knowledge Seeker*** and ***Knowledge Studio*** of the **Altair Data Analytics** product suite!

- **Knowledge Seeker®** is a data mining and predictive analytics software tool used by business analysts and advanced data scientists for data exploration, decision tree analysis, and strategy development. Knowledge Seeker capabilities include data preparation and profiling with advanced visualization, decision trees, ensemble trees, strategy design, model performance evaluation, and deployment of decision tree models and strategies. The product provides the capability to create repeatable workflows and execute them with automatic data refresh. All models and some data profiling and manipulation operations can be automatically translated to SAS Language using automated code generation.

- **Knowledge Studio®** is a data mining and predictive analytics software tool which enables advanced analysts to build and deploy predictive models using repeatable workflows. Knowledge Studio features cover all stages of the predictive analytics life cycle, from data preparation and profiling, variable selection, decision tree analysis, building predictive models, cluster analysis, principal component analysis, market basket analysis, building scorecards, applying reject inference techniques, model performance evaluation, building and deploying strategies. All native models and some data profiling and manipulation operations can be automatically translated to SAS Language using automated code generation. Optimization algorithms available in Knowledge Studio maximize or minimize any user-defined metric representing your business objective while maintaining relevant business constraints. Linear optimization on the dataset level handles user-defined linear objective functions with linear constraints, covering problems on the level of aggregating the values over a given dataset. Nonlinear optimization can be performed on both dataset level and record level. For details, see the Help topics in the Optimization section.

- **Knowledge Studio® for Apache Spark™** is a comprehensive data science platform integrated with Apache Spark technology to provide advanced data mining and predictive analytics on large-scale distributed data structures such as Hadoop HDFS, Amazon S3, and other storage types supported by Spark. Data can be loaded directly from Hadoop HDFS and ViewFs, Hadoop Archive, Amazon S3, FTP and Network Shares. Hive tables, Text (CSV) and Parquet formats are supported. Data manipulation, profiling, modeling (including interactive decision trees and strategy trees), model evaluation and deployment are supported on Spark. Python code is automatically generated for all operations. Custom programs for data science tasks using any available algorithm libraries in Python and PySpark can be written within Knowledge Studio workflows. Jupyter Notebook can be launched from within a Knowledge Studio workflow for more advanced coding environment. All Knowledge Studio workflow tasks on Spark can be scheduled for automatic execution. See the Help section "*Integration with Apache Spark*" for more information.

Integration features:

- **Integration using SAS Language** allows users to write and execute SAS Language programs in Knowledge Studio workflows. SAS System files and WPD files can be used as dataset nodes and converted to Altair KDD datasets. This feature is powered by Altair Analytics Workbench, formerly known as WPS Workbench.

- **R and Python Language Integration** is a standard feature in Knowledge Seeker and Knowledge Studio. Users can write and execute R or Python code within the R

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Code node or Python Code node and use its outputs in further operations within the same workflows. All the versatile data manipulation functions and data science algorithms of R and Python can therefore be used within Knowledge Seeker and Knowledge Studio. For details, see the Help topics for R and Python integration features.

- **Tableau Integration** and **Qlik Integration** are standard features in Knowledge Seeker and Knowledge Studio.

Add-ons:

Some features are available as add-ons for the products that do not include them as standard features. The available add-ons include In-Database Analytics, and Text Analysis.

- **In-Database Analytics** is available as an add-on for Knowledge Seeker and Knowledge Studio. It provides an in-database connector for Microsoft SQL Server, Oracle, and Teradata for data preparation and profiling, decision trees, strategy trees, model performance evaluation, and deployment of decision and strategy trees.

Main Differentiators of Knowledge Seeker and Knowledge Studio

- intuitive graphical user interface and superior visualization of results
- superior usability and short learning curve
- visual workflows with automatic data refresh
- best-in-class decision trees
- strategy trees for developing and deploying strategies
- integration with other analytics and database environments via data import and export drivers, automated code generation for model deployment, creating and running custom and pre-built programs in the languages of R and Python
- SAS Language code generation for entire workflow or any part of it

The Welcome Page

The Welcome page displayed upon starting the application provides links to the following features:

- **Recent Working Directories**

  Connect to any of the recently used Working Directories. The Working Directory locations were saved from your previous sessions.

- **Create a New Project**

  A **project** consists of one or more workflows and the results of their execution: a collection of datasets and analysis results such as decision trees and other models, strategy trees, model analyzers, generated code, and other objects. Creating a new project is the first step in your work with the application. The **Create New Project** link in the Welcome screen is a shortcut to the **File | Create Project** menu command.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- **Getting Started**
  - ***Help:*** Click the Help link to access the built-in user guide, also available from the Help menu.
  - ***Tutorial***: The tutorial is a quick and easy way to learn how to use the application. It uses sample projects included with the product to demonstrate the most commonly used functions. While completing the tutorial lessons, you will learn how to create a project, import a dataset, profile your data, build models, validate them and analyze their performance. This link in the Welcome Screen is a shortcut to the **Help | Tutorial** menu command.
  - ***User Community***: Join our online community of experts and users to share insights, collaborate with colleagues and find more ways to take full advantage of your product. This link takes you directly to the Knowledge Studio and Knowledge Seeker user forum, where you can participate in online discussions, network with users like yourself, and get answers to your questions with 24/7 support from the community members. The User Community link is also available under the Help menu: **Help | User Community.**

Learn About the **Altair Data Analytics**™ Products and Altair Services

To learn more about the Altair Data Analytics products, as well as services and training available to Altair customers, please visit our website https://www.altair.com/knowledge-studio/ or select "*About Altair Data Analytics*" from the Help menu. For customer support information, see https://community.altair.com/community and the Technical Support Help topic.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Summary of New Features

This topic contains a summary of new features in version 2022.3 as well as the history of features introduced in the previous product versions starting from 2020.0.

A history of all features introduced in our software releases since version 4.0 is available upon request.

See the topic Version Compatibility for information on project compatibility when upgrading from an older version.

New Features in Version 2022.3 Compared to 2022.2

Data Import and Data Preparation

- Monarch DPS Import and Export nodes are enabled or disabled depending on whether Monarch Data Prep Studio is installed.

Predictive Models and Time Series Models

- Enhanced Keras Deep Learning model wizard for more granular model control, pre-processing and post-processing, including imputation, scaling, optimizer, and early stopping options.
- Enhanced and optimized XGBoost model wizard. Task type selection (Classification / Regression); No output dataset created after training.
- Refactored Scoring code for Keras Deep Learning model to support for Regression and Classification models with single and multiple dependent variables.
- Milliseconds are supported for the Flexible Split partitioning in the Time Series Decomposition wizard.

Model Validation and Deployment

- Model Validation node is enabled for Keras Deep Learning models.
- Scoring can now be applied to Scorecard models imported from PMML.

New Features in Version 2022.2 Compared to 2022.1

Data Import, Data Profiling and Preparation

- **WPD files** and **SAS System files** can be used as dataset nodes on the Knowledge Studio workflows.
- **SAS Language Data Converter** node in the Connect palette converts data between three formats: Altair KDD datasets, SAS System files (*.sas7bdat), and WPD files (the WPS Workbench data format).
- The new node **Import from Monarch DPS** in the Connect palette imports data from Altair Monarch Data Prep Studio.
- Time and Timestamp values with **milliseconds** are now supported.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Milliseconds are displayed in the data views, including data preview in the Import wizards, the Data tab of datasets, Node Data tab of Decision Trees, and the Data Preview in Variable Transformations.
- Milliseconds are supported in the Make Time Series node.

Predictive Models and Time Series Models

- **Imputation strategy** options (median, mean, and mode) are now provided for preprocessing in the Novelty & Outlier Detector model.
- **Scaling strategy** and **imputation strategy** options are now provided for preprocessing in the Generalized Linear Model (GLM).
- Aggregation at the millisecond level is now supported in the Time Series Decomposition node.

Model Deployment

- Market Basket Analysis scoring code in the SAS Language can now be generated from the context menu of the Scoring node.

Integration using SAS Language

- **SAS Language Code** node in the Action palette allows users to write and execute programs in the SAS Language and embed them in the Knowledge Studio workflows. This feature is powered by the integration with Altair Analytics Workbench, formerly known as WPS Workbench.

New Features in Version 2022.1 Compared to 2022.0

Data Import

- File Import wizard for Text files allows specifying the decimal separator and group separator symbols.
- File Export wizard for Text files allows specifying the decimal separator symbol.
- File Import for Excel automatically trims leading and trailing spaces from column names.

Predictive Models

- An **AutoML** node can be copied from one workflow tab to another. The resulting node inherits all settings of the original AutoML node.

Model Evaluation

- The **Model Evaluation** node is now applicable to the **XGBoost** and **GLM** model nodes.

Model Deployment

- The **Scoring** wizard for the **XGBoost Model** includes a new page for specifying the names for the scoring fields and the cut-off values for the probability of the dependent variables.
- The **Scoring** node is now applicable to the **Keras Forecasting** model to produce forecasts on new datasets.

Integration with R and Python

- Path to R executable can now be specified in the **Survival Analysis** node wizard.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

New Features in Version 2022.0 Compared to 2021.3

Data Import

- File Import wizard allows importing data from any worksheet of multi-sheet Excel files.

Data Preparation and Profiling

- The **Binning** option is provided in Correlations to improve the computation performance for the Spearman, Kendall, and Hoeffding coefficients for large datasets with high-cardinality variables.
- Three more aggregation functions are supported in the Expression Editor for datasets, Strategy Trees, and Optimization nodes: **DEVSQR** (the sum of the squared deviations from the mean), **AVGPROD** (the average of the products of the argument variables), and **SUMPROD** (the sum of the products of the argument variables).

Decision Trees and Ensemble Trees

- **Variable Importance** is now calculated in Ensemble Tree models (Random Forest, Bagging, and Boosting).
- **Boosting** node wizard allows continuous dependent variables and no longer requires setting the *Target category*.

Model Evaluation

- **Decision Tree** tab added to the **Explainable AI** node outputs.
- Improvements in the **SHAP** tab of the **Explainable AI** node.
- Unified **Model Validation** node for all types of models.

Model Deployment

- Unified **Scoring** node for all types of models.

Optimization

- The Dataset-level Optimization node wizard now allows specifying constraints using new syntax - a free form of boolean expressions using any variable or a combination of variables, applicable to aggregate functions as well.

Integration

- Python and R execution environments can be specified for the **Python Code** node and **R Code** node.
- Models can be uploaded to Altair SmartWorks Analytics.

Knowledge Studio for Apache Spark

- **Desktop Document** and **HTML Report** nodes are now enabled in the Connect palette of Knowledge Studio for Apache Spark.

New Features in Version 2021.3 Compared to 2021.2

Decision Trees

- The Decision Tree node wizard now has an option to modify the tree settings when a template tree is used.

Modeling

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- The new **Sklearn Novelty & Outlier Detector** node in the **Model** palette of Knowledge Studio helps determine whether a set of observations belong to the same distribution as the other observations in a dataset or should be classified as outliers. The node uses functions of **`scikit-learn`** - a popular Python library for machine learning. This feature requires Docker.
- The new **Sklearn XGB** node in the **Model** palette of Knowledge Studio allows you to build models using the Python implementation of XGBoost (Extreme Gradient Boost) algorithm. It uses gradient boosting functions for classification and regression in **`scikit-learn`** - a popular Python library for machine learning. This feature requires Docker.

Model Evaluation

- **SHAP** (SHapley Additive exPlanations) plots are added in the Explainable AI node output.

Model Deployment

- The **Sklearn Novelty & Outlier Scoring** node in the **Action** palette performs scoring for the Sklearn Novelty & Outlier Detector node. The node uses the **`scikit-learn`** library of Python and requires Docker.
- The **Sklearn XGB Scoring** node in the **Action** palette performs scoring for the Sklearn XGB node. The node uses the **`scikit-learn`** library of Python and requires Docker.
- Automated **C Code generation** is now enabled in the Generate Code node for all native models except Market Basket Analysis.

Knowledge Studio for Apache Spark

- Apache Spark™ version 3.1.x is now supported.

New Features in Version 2021.2 Compared to 2021.1

Installation and Configuration

- Application preferences can now be set at the installation time using an option in the silent install. A previously saved or exported preferences file can be specified in the command line during the silent install.
- Application preferences can be imported to or exported from the **Preferences** dialog under the **Tools** menu.

Data Preparation and Profiling

- The new **Substitute Missing Values** node in the Manipulate palette performs the substitution of missing values for multiple variables at a time and also provides an option for handling infinity values +/-{INF}. Unlike the analogous features in the Variable Transformation node, this node substitutes the values in the original variables rather than creating new variables.
- The new **Handle Class Imbalance** node in the Manipulate palette balances out the classes of a dependent variable with skewed distribution via oversampling the minority class by creating artificial data points (synthesizing data). This node uses

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

`Imbalanced-Learn` (`imblearn`) - a popular Python library dealing with classification with imbalanced classes when preparing data for modeling. This feature requires Docker.

- The new **Simpson's Paradox Detection** node in the Profile palette performs automatic detection of the Simpson's Paradox in data with respect to the selected variable. Detecting this phenomenon helps avoid the misinterpretation of trends and drawing wrong conclusions about biases or causal relationships.

Modeling, Time Series Analysis, and Scorecards

- **Generalized Linear Model**: The new **Sklearn GLM** node in the **Model** palette of Knowledge Studio allows you to build regression models with a dependent variable that does not necessarily follow the normal distribution. The node uses linear model methods of **`scikit-learn`** - a popular Python library for machine learning. This feature requires Docker.

Model Evaluation and Deployment

- The **Sklearn GLM Validation** node in the Evaluate palette performs validation for the Sklearn GLM node (Generalized Linear Model). The node uses linear model methods of the Python library **`scikit-learn`** and requires Docker.
- The **Sklearn GLM Scoring** node in the Action palette performs scoring for the Sklearn GLM node (Generalized Linear Model). The node uses linear model methods of the Python library **`scikit-learn`** and requires Docker.

R and Python Language Integration

- Automated **R Code generation** is now enabled in the Generate Code node for regression models, Regularization, Cluster Analysis, Deep Learning, Factor Analysis, PCA, Reject Inference, Scorecards, and Market Basket Analysis models.

New Features in Version 2021.1 Compared to 2021.0

Data Preparation

- The new **Auto Feature Engineering** node in the Manipulate palette automatically creates new features using transformations and aggregations of different types from two or more input datasets. This feature requires Docker.

Decision Trees and Strategy Trees

- The number of missing values is now displayed for continuous dependent variables in Strategy tree nodes.

Modeling, Time Series Analysis, and Scorecards

- The new **ARIMA Forecasting** node in the Model palette performs ARIMA forecasting with **`pmdarima`** - a Python statistical library for time series forecasting. This feature requires Docker.
- The new **Points Range Scaling** option in the Scorecard node allows setting a maximum and minimum number of points instead of the Odds Scaling based on the base points, base odds, and points to double the odds.
- The **Cross-Validation** option is now available in the Modelling settings of AutoML.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Selecting a single model type with Grid Search is allowed at the Modelling stage in AutoML..

R and Python Language Integration

- Code in the Python Code Node can now be easily updated from Jupyter Notebook without closing the notebook.

Optimization

- Global decision variables can now be used in the Dataset-Level Optimization and Multiple Scenario Optimization nodes.

Access to Support Forums and Online Community

- The link to the Knowledge Studio user community is now available in the Welcome screen and on the Help menu.

New Features in Version 2021.0 Compared to 2020.3

Data Preparation

- The WOE node Wizard has two more output format options: Microsoft Excel and Microsoft Word.
- The command Copy Data to Excel is now available in the WOE Report.

Modeling

- Feature Engineering (Variable Transformations) added in the AutoML node wizard.

R and Python Language Integration

- Jupyter Notebook integration in the Python Code Node.
- Keras Forecasting node.
- Ability to configure GPU use for TensorFlow code in Keras features.
- In the R Code editor, the code can be saved as a new node in the new **Custom Library** palette.

New Platform Support

- Red Hat Enterprise Linux 8 and CentOS 8 are now supported.

New Features in Version 2020.3 Compared to 2020.2

Data Preparation

- New functions FINITE() and PFINITE() are available in the Expression Editor of Variable Transformations. The functions replace infinite values with user-defined values.

Decision Trees and Strategy Trees

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- A new option in Attribute Editor to isolate missing values in a separate bin (applies to Find Split and Automatic Grow).
- Attribute Editor performance has been substantially improved for wide datasets.
- The dependent variable distribution is preserved by default during split operations in Strategy Trees where applicable.

Modeling

- **Grid Search** capability for hyperparameter tuning in the AutoML node wizard. The hyperparameter settings can be saved, exported and re-imported.
- Cluster Analysis and Deep Learning model wizards have a new option to perform missing and infinite value substitution before normalization.
- Numeric format and precision options and better scaling of continuous variable charts are provided in the Cluster Analysis Segment Viewer.

Model Evaluation

- Local Interpretable Model-Agnostic Explanations (LIME plots) added in the Explainable AI node output.

R and Python Language Integration

- User interface enhancements in the Python Code node and R Code node:
  - Syntax highlighting
  - Auto-complete feature for commands and keywords
  - Line numbers; collapsing and expanding blocks of code
  - Undo, Redo, Find, Replace, Go to Line, and Save As commands available in the context menu and as keyboard shortcuts.
- In the Python Code editor, the code can be saved as a new node in the new **Custom Library** palette.
- In the R Code editor, R plots can be viewed and managed in a separate frame, and plots can be saved as image files.

New Features in Version 2020.2 Compared to 2020.1

Data Import and Export

- Datasets can be exported to **Altair Monarch** version 15.x or 2020 as delimited text files. The new node **Export to Monarch** is provided in the **Connect** palette.

Data Preparation

- **Renaming variables**: Any variable can now be renamed in the Dataset Editor interface of the **Variable Transformations** node.

Modeling

- **AutoML (Automated Machine Learning)**: The new **AutoML** node in the **Model** palette of Knowledge Studio creates a complete modeling workflow starting from data preparation (variable selection and data transformations), to modeling, validation, and selection of the best model for deployment. This feature is included in Knowledge Studio.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Model Evaluation

- The new **Explainable AI** node in the **Evaluate** palette of Knowledge Studio helps explain and evaluate the results of "black-box" models such as Random Forest and Deep Learning (Neural Networks).

R Language Integration Update

- **R versions 4.0.0** and higher are now supported by the **R Code** node.

New Features in Version 2020.1 Compared to 2020.0

Data Import and Export

- Data import from R is now supported for RData files created in R version 3.6.x and higher.

Data Preparation and Profiling

- The new **Pivot Table** node in the Manipulate palette for rearranging and summarizing data.
- **Data Preview** feature in the Variable Transformations node. The transformed data can now be previewed before running the transformations.
    - Data Preview in the Expression Editor to preview data while adding a new calculated field.
    - Data Preview in the Dataset Editor to preview top N records of the transformed dataset.
- Field reordering is now supported in the Dataset Editor of the Variable Transformations node.

New model types

- The new **Model Stacking** node in the Model palette is an ensemble model that combines predictive scores produced by multiple models of different types and builds the final model.
- The new **Keras Deep Learning** node in the Model palette uses the Keras neural network library via integration with Python. The **Keras Scoring** node is provided for deploying Keras Deep Learnings.
    - *Note: Keras Deep Learning features require Docker software for container support as a prerequisite. Keras Deep Learning features are not supported on Red Hat Enterprise Linux 6 & 7 and CentOS 6 & 7.*

Integration with Tableau®

- Integration with Tableau® software is now supported for Tableau versions 2019.1 and higher.
    - *Note: Tableau integration is not supported in the client/server configurations for Red Hat Linux 6 and CentOS 6*.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

New Features in Version 2020.0 Compared to 10.9

Data Import and Export

- Unified file import: A single File Import node for all supported source formats.
- Unified file export: A single File Export node for all supported target formats.
- Enhanced data preview on import.
- The new Folder Import node imports data from all files in a folder in one take, with options to import data into separate datasets or append to a single dataset.

Data Preparation and Profiling

- The new **Sort, Rank, Lag** node in the Manipulate palette for sorting data, creating ranking fields and lagged variables.
- The new **Multicollinearity Analysis** node in the Profile palette for automatic removal of highly correlated variables.
- The **Append** node now admits more than two input datasets to append multiple datasets at a time.

New model types

- The **Time Series Decomposition** node in the Model palette performs transformation of time series data into component: Series, Trend, Seasonal, and Residuals.

Code Generation

- Unified code generation node: A single code generation node for all supported target code types.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Version Compatibility

- Version 2022.3 can be installed and run side by side with any previous version.
- If you choose to install version 2022.3 side by side with your current version, use separate working directories for each version, since older projects modified in 2022.3 may no longer be compatible with their original version. See the topic Setting the Working Directory.

Project Compatibility

- Projects created in versions 9.3 -- 2022.2 can be opened and modified in version 2022.3. They are automatically converted to the latest version.
- Projects created in versions 7.0 – 8.7 can be opened in version 2022.3, but it will not automatically reconstruct a workflow from the original project, so the workflow canvas will initially be empty. Nevertheless, all original objects (datasets, decision trees, models, etc.) can be opened from the Project Explorer pane and modified. Double-click any object in Project Explorer to open it. Datasets and models can be manually placed on the workflow canvas from Project Explorer using the right-click context menu command **Put on Workflow Canvas** or the nodes **Dataset Link** and **Model Link** in the **Connect** palette.
- Forward compatibility is not supported: models and other objects in projects created or modified in version 2022.3 may not be loaded or displayed correctly when opened in an older version.

For more information on the Knowledge Studio and Knowledge Seeker project format, see the topics Projects and Workflows.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Technical Information and Client Support

Hardware and Operating System Requirements

Overview: Supported Operating Systems

Knowledge Seeker and Knowledge Studio 2022.3 are available in the standalone and client/server configurations for the following operating systems:

KS Workstation (Standalone or Client)	KS Server
Windows on x86-64 platform: Windows 10 Windows 11 Windows Server 2016 Windows Server 2019 Windows Server 2022	Windows Server on x86-64 platform: Windows Server 2012 R2 Windows Server 2016 Windows Server 2019 Windows Server 2022
	Linux on x86-64 platform: Red Hat Enterprise Linux 7 and 8 CentOS 7 and 8

- All products are supported only on 64-bit systems.
- Knowledge Studio for Apache Spark requires Apache Spark server. System requirements for this product are described at the end of Section 3.2 below.

System Requirements

This section describes the minimum and recommended system requirements for both desktop (standalone) and client/server configurations.

Microsoft .NET Framework 4.6.1 (or higher) **and .NET Framework 3.5 Service Pack 1** are prerequisites for KS Workstation. The KS Workstation installation package includes web installers for these components and installs them automatically if they are not found on the system.

Microsoft Visual C++ 2015-2022 Redistributable package is required for KS Workstation and KS Server. Both KS Workstation and KS Server installers include the web installer for Microsoft Visual C++ 2015-2022 Redistributable and installs it unless the same or a higher version is already present.

KS Workstation includes KS PDF Printer, which enables the Copy to PDF functionality for exporting decision and strategy trees, grids, charts and reports to PDF. KS PDF Printer uses Amyuni PDF Converter by Amyuni Technologies, Inc. Note: In case KS PDF Printer is removed from the system,

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

it can be restored by running AngossPDFInstall.cmd from the folder <Drive>:\Program Files\Altair Data Analytics\KS Workstation 2022.3.0\bin.

R integration features require R Statistical Software v3.5.0 or higher and KS Library 2022.3 for R. KS Library for R is provided as separate installation package available on the Altair Data Analytics software download site.

Python integration requires Python version 3.7 or higher. See the deployment guide for details.

The following are the minimum and recommended system requirements for all configurations:

1. Standalone Configuration

KS Workstation 2022.3 – Standalone	
Minimum	**Recommended**

• Windows 10 or 11 (64-bit)
• x86-64-based CPU, 1.5 GHz
• 4 GB RAM
• Disk space:
  • 4 GB for application program files**
  • 15+ GB for user's projects
• Monitor resolution: 1152x864
	• Windows 10 (64-bit)
	• Quad-core x86-64-based CPU, 2 GHz, or better*
	• 8 GB RAM or more*
	• Disk space:
	  • 4 GB for application program files**
	  • 20+ GB for user's projects*
	• Monitor resolution: 1680x1050 or better

*** Note**: Recommended CPU, RAM, and disk space depend on the typical size of analytic datasets and the type of analysis to be performed. If you need our help in determining the optimal hardware parameters for your desktop for the type of analysis you need to perform, please contact Customer Support.

** If Python Code node is used, Python and the required packages may take at least 3 GB of disk space, depending on the Python distribution you intend to use.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

2. Client/Server Configuration

KS Workstation 2022.3 – Client	
Minimum	**Recommended**
• Windows 10 or 11 (64-bit) • x86-64-based CPU, 1.5 GHz • 4 GB RAM • Disk space: 4 GB for application program files • Monitor resolution: 1152x864	• Windows 10 (64-bit) • x86-64-based CPU, 2 GHz, or better • 8 GB RAM or more • Disk space: 4 GB for application program files • Monitor resolution: 1680x1050 or better

KS Server 2022.3	
Minimum	**Recommended**
• Windows Server 2016, 2019, 2022; Red Hat Enterprise Linux 7 or 8; CentOS 7 or 8 • Quad-core x86-64-based CPU, 1.5 GHz • 8 GB RAM • Disk space: 4 GB for application program files*** • 10+ GB per user for user projects	• Windows Server 2016, 2019, or 2022; Red Hat Enterprise Linux 7 or 8; CentOS 7 or 8. • Multi-core x86-64-based CPU(s) with at least 8 cores in total, 2 GHz, or better ** • 16 GB RAM or more** • Disk space: 4 GB for application program files*** • 15+ GB per user for user projects**

**** Note**: System requirements for the server depend on the maximum number of users expected to be using KS Server concurrently. The KS Server performance also depends on the typical size of analytic datasets, on whether any other applications run on the same server, and other factors. To ensure that the software will be compatible with your environment and will deliver acceptable performance, please fill in the Client Installation Questionnaire. If you need our help in determining the server size sufficient for the desired number of concurrent users of KS Server, please contact Customer Support.

*** If Python integration features are used, Python and the required packages may take at least 3 GB of disk space on the server, depending on the Python distribution you intend to use.

Please refer to the Deployment Guide for installation guidelines.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Technical Support

Licensed users seeking technical support for the application may access the Altair Support Portal at https://community.altair.com/community, email us at dasupport@altair.com, or call 1-888-715-7391 (toll-free in North America) or +1-416-593-2417.

We welcome your suggestions on how to make our application easier to understand and use. Please forward your comments and suggestions to dasupport@altair.com.

New Releases and Updates

We plan future updates, upgrades, and new releases which will be available to licensed users. Visit our website https://www.altair.com/knowledge-studio/ for the latest news about our products. Contact Customer Support if you have questions about upgrading your software.

Documentation

The following additional resources are available for customers on the software download site or upon request:

- **Release Notes** (PDF)
- **User Guides** – PDF versions of the Help system content.
- **Product Guides** – A brief summary of the product features for each product (PDF).
- **Deployment Guide** – The installation and configuration guide for all configurations of Knowledge Seeker and Knowledge Studio (PDF).
- **Creating Scorecards with Knowledge Studio** – A white paper on building scorecards in Knowledge Studio (PDF).
- **KS Server Administration Utilities**. – A user guide to simple utilities for usage monitoring and user process administration for the client/server configuration (PDF).

Copyright Information

Copyright © 1994-2022 Altair Engineering Canada, Ltd. All rights reserved.

Copyright © 2018-2022 Altair Engineering Inc. All rights reserved.

This computer program and related documentation are protected under national and international copyright and intellectual property laws. Use hereof is subject to the end user license agreement between the licensed user and Altair Engineering, Inc. Unauthorized reproduction, distribution or other use of this computer program, including use hereof by any person other then the authorized licensee hereof, is expressly prohibited under such laws, may result in serious civil and criminal penalties, and in the event of violations will be prosecuted to the fullest extent of the law.

Also, portions of the software utilize the following copyrighted material, the use of which is hereby acknowledged:

Copyright © 2008-2022 Software FX, Inc. All Rights Reserved. **Chart FX** is a registered trademark of Software FX, Inc.

Copyright © 1998-2011 Crownwood Software, Inc. All Rights Reserved. (Registered number 03592253 / VAT No. 711126780). **DotNet Magic** is a registered trademark of Crownwood Software, Ltd.

Copyright ©1992-2022 Infragistics, Inc., 2 Commerce Drive, Cranbury, NJ 08512. All rights reserved. Infragistics, **NetAdvantage**, and the Infragistics logo are registered trademarks of Infragistics, Inc.

Copyright ©2014-2022 Northwoods Software Corp. (**GoDiagram for WinForms**) All rights reserved. Northwoods Software® is a Registered Trademark of Northwoods Software Corporation.

Copyright ©2001-2022 Amyuni Technologies Inc. (**Amyuni PDF Converter**) All Rights Reserved. Amyuni and the Amyuni logo are trademarks of Amyuni Technologies Inc.

Copyright ©1986-2022 Circle Systems, Inc. (**Stat/Transfer** drivers for data transfer) All rights reserved.

Copyright © 2003-2022 Tableau Software. All Rights Reserved. **Tableau**® and Tableau Software® are registered trademarks of Tableau Software Inc. License agreements and terms: http://www.tableau.com/legal

Copyright © 2022 ActiveState Software. All rights reserved. **ActivePython** 3.8 is used under an enterprise license with ActiveState. The ActivePython distribution includes the following Python packages subject to their respective copyrights and end-user license agreements:

absl_py 0.14.1, alembic 1.4.1, astunparse 1.6.3, autonormalize 1.0.1, azure_core 1.23.0, azure_storage_blob 12.11.0, boto3 1.17.22, botocore 1.20.112, cachetools 4.2.4, certifi 2021.10.8, cffi 1.15.0, charset_normalizer 2.0.7, click 8.0.3, cloudpickle 2.0.0, colorama 0.4.4, cryptography 36.0.2, Cython 0.29.17, dask 2021.9.1, databricks_cli 0.16.4, distributed 2021.9.1, docker 5.0.3, entrypoints 0.4, featuretools 0.23.0, Flask 2.1.0, fsspec 2021.10.0, gast 0.3.3, gitdb 4.0.9, GitPython 3.1.27, google_auth_oauthlib 0.4.6, google_auth 2.3.0, google_pasta 0.2.0, gorilla 0.4.0, grpcio 1.41.0, h5py 2.10.0, HeapDict 1.0.1, idna 3.3, imbalanced_learn 0.8.0, importlib_metadata 4.11.3, isodate 0.6.1, itsdangerous 2.1.2, Jinja2 3.0.2, jmespath 0.10.0, joblib 1.1.0, Keras_Preprocessing 1.1.2, Keras 2.4.3, locket 0.2.1, Mako 1.2.0, Markdown 3.3.4, MarkupSafe 2.0.1, mlflow 1.11.0, msgpack 1.0.2, msrest 0.6.21, numpy 1.18.5, oauthlib 3.1.1, opt_einsum 3.3.0, packaging 21.0, pandas 1.2.3, partd 1.2.0, patsy 0.5.2, pip 21.3, pip 22.0.4, pmdarima 1.7.1, prometheus_client 0.13.1, prometheus_flask_exporter 0.19.0, protobuf 3.18.1, psutil 5.8.0, pyasn1_modules 0.2.8, pyasn1 0.4.8, pycparser 2.21, pyparsing 2.4.7, python_dateutil 2.8.2, python_editor 1.0.4, pytz 2021.3, pywin32 227, PyYAML 6.0, querystring_parser 1.2.4, requests_oauthlib 1.3.0, requests

2.26.0, rsa 4.7.2, s3transfer 0.3.7, scikit_learn 0.24.2, scipy 1.7.1, setuptools 49.6.0, six 1.16.0, smmap 5.0.0, sortedcontainers 2.4.0, SQLAlchemy 1.3.13, sqlparse 0.4.2, statsmodels 0.11.1, tabulate 0.8.9, tblib 1.7.0, tensorboard_data_server 0.6.1, tensorboard_plugin_wit 1.8.0, tensorboard 2.7.0, tensorflow_gpu_estimator 2.3.0, tensorflow_gpu 2.3.1, termcolor 1.1.0, threadpoolctl 3.0.0, toolz 0.11.1, tornado 6.1, tqdm 4.62.3, typing_extensions 4.1.1, urllib3 1.26.7, waitress 2.1.1, websocket_client 1.3.2, Werkzeug 2.0.2, wheel 0.37.0, wrapt 1.13.2, xgboost 1.4.2, zict 2.0.0, zipp 3.7.0.

Copyright © 2018-2022 Intel Corporation. **Intel® Math Kernel Library** (**Intel® MKL**) is used under the terms of the Intel Simplified Software License: https://software.intel.com/en-us/license/intel-simplified-software-license

Bonmin (Basic Open-source Nonlinear Mixed INteger programming) is a numerical analysis library for solving general MINLP (Mixed Integer NonLinear Programming) problems. Bonmin version 1.5 is used under the terms of the Eclipse Public License 1.0 (EPL-1.0) https://www.eclipse.org/legal/epl-v10.html. Bonmin project description: https://projects.coin-or.org/Bonmin. Source code: https://projects.coin-or.org/Bonmin/browser and https://github.com/coin-or/Bonmin.

Copyright © Microsoft Corporation. **Microsoft Automatic Graph Layout, MSAGL**. All rights reserved. MSAGL version 2.1 is used under the terms and conditions of the MIT License: https://github.com/Microsoft/automatic-graph-layout/blob/master/LICENSE. Source code: https://github.com/Microsoft/automatic-graph-layout.

Copyright © 2002-2012 Davide Icardi, Darius Damalakas. **SourceGrid** version 2.0.9.0 is used under the terms of the MIT License: https://github.com/siemens/sourcegrid/blob/master/LICENSE. Source code: https://github.com/siemens/sourcegrid. See also https://www.codeproject.com/Articles/3531/SourceGrid-Open-Source-C-Grid-Control.

Copyright © 1998-2022 The OpenSSL Project. All rights reserved. Software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/) is used under the terms and conditions of the OpenSSL license http://www.openssl.org/source/license.html. Source code: https://github.com/openssl/openssl. OpenSSL is a registered trademark owned by OpenSSL Software Foundation.

Copyright © 2014-2022 The OpenBSD Project. All rights reserved. **LibreSSL** version 2.5.5 is used under the terms and conditions of the Apache License version 1.0 https://www.apache.org/licenses/LICENSE-1.0, 4-clause BSD License https://directory.fsf.org/wiki/License:BSD-4-Clause, and ISC license https://www.isc.org/downloads/software-support-policy/isc-license/. Source code: https://github.com/libressl-portable/.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Copyright © 1998 John E. Bossom. Copyright © 1999, 2006 Pthreads-win32 contributors (http://sources.redhat.com/pthreads-win32/contributors.html). **Pthreads-win32** version 2.9.0.0 - POSIX Threads Library for Win32 - is used under the terms of the GNU Lesser General Public License (LGPL) version 2.1: https://www.sourceware.org/pthreads-win32/copying.lib.html. Source code: ftp://sourceware.org/pub/pthreads-win32.

Copyright © 2022, Applied Informatics Software Engineering GmbH and Contributors. **POCO C++ Libraries** version 1.11.0 are used under the terms and conditions of the Boost Software License 1.0 https://pocoproject.org/about.html#license. Portions of the POCO C++ Libraries utilize copyrighted material acknowledged at https://pocoproject.org/docs/90100-Acknowledgements.html. POCO Project description and source: https://pocoproject.org/.

Copyright © 2013 John Dickinson. The **C# wrapper for SHBrowseForFolder** is used under the terms and conditions of the MIT License http://opensource.org/licenses/mit-license.php. Source code and description: http://unafaltadecomprension.blogspot.ca/2013/04/browsing-for-files-and-folders-c.html.

The **libiconv** library version 1.14 for text encoding conversion is used under the terms of the GNU Lesser General Public License (LGPL): http://www.gnu.org/licenses/lgpl.html. Description: https://www.gnu.org/software/libiconv/. Source code: https://savannah.gnu.org/projects/libiconv/

The .NET Gold Parser Engine **bsn-goldparser** version 5.2.0 - is used under the terms of the GNU Lesser General Public License (LGPL) http://www.gnu.org/licenses/lgpl.html. Description: http://www.goldparser.org/doc/index.htm. Source code: https://github.com/siriusch/bsn.GoldParser.

Copyright © 2011, 2013 Vincent Tan Wai Lip. The **SpreadsheetLight** library version 3.4 (27 July 2013) is used under the terms of the MIT License http://spreadsheetlight.com/license. Source code: http://spreadsheetlight.com/download.

The linear programming solver **CLP (COIN-OR LP)** version 1.14.0 is used under the terms of the Eclipse Public License 1.0 (EPL-1.0): https://github.com/coin-or/Clp/blob/master/LICENSE. Source code: https://github.com/coin-or/Clp.

Copyright © 2018-2022 Newtonsoft. Copyright © 2007 James Newton-King. A JSON framework for .NET, **JamesNK/Newtonsoft.Json** version 6.0 is used under the terms of the MIT License https://github.com/JamesNK/Newtonsoft.Json/blob/master/LICENSE.md. Description: http://www.newtonsoft.com/json. Source code: https://github.com/JamesNK/Newtonsoft.Json.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

DocX version 1.0 is a .NET library for manipulating Microsoft Word files used under the terms of the Microsoft Public License (Ms-PL): https://opensource.org/licenses/MS-PL. Description and source code: https://github.com/xceedsoftware/DocX.

Copyright © 1998-2022 Neil Hodgson (neilh@scintilla.org). All rights reserved. **Scintilla** is a free source code editing component used under the terms of the license for Lexilla, Scintilla, and SciTE https://www.scintilla.org/License.txt. Source code: https://sourceforge.net/projects/scintilla.

Copyright © 2006, 2007 Martin R. Gagné (martingagne@gmail.com). All rights reserved. The loading circle animation component v1.0 for .NET is used under the terms of the Code Project Open License (CPOL): https://www.codeproject.com/info/cpol10.aspx. Description and source code: https://www.codeproject.com/Articles/14841/How-to-write-a-loading-circle-animation-in-NET.

Copyright © 2015 Elad Zelingher, CodeSharp. All rights reserved. A C# implementation of **WAMP** (The Web Application Messaging Protocol) version 18.11.1 is used under the terms of The FreeBSD License: https://github.com/Code-Sharp/WampSharp/blob/master/LICENSE. Description and source code: https://github.com/Code-Sharp/WampSharp.

A Windows SSPI SSL implementation for CUPS version 2.2.9 is used under the terms and conditions of the Apache License Version 2.0 https://www.cups.org/doc/license.html. Source code: https://github.com/apple/cups.

Icons made by Freepik (http://www.freepik.com) from Flaticon (http://www.flaticon.com) are used under the terms of the Creative Commons BY 3.0 license: http://creativecommons.org/licenses/by/3.0/

Icons of The Noun Project (http://www.thenounproject.com) are used under the Terms of Use: https://thenounproject.com/legal/, Creative Commons Attribution (CC BY) and Public Domain (PD) licenses https://creativecommons.org/licenses/by/3.0/, https://creativecommons.org/publicdomain/mark/1.0/.

Apache, Apache Spark, Spark and the Spark logo are trademarks of the Apache Software Foundation. Knowledge Studio for Apache Spark provides predictive analytics on distributed storage types supported by Apache Spark.

Microsoft, MS, Visual C++, C#, Excel, Access, Windows, and Windows NT are registered trademarks, and ActiveX, Developer Studio, DirectX, and Visual J++ are trademarks of Microsoft Corporation.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

All other products, logos, or brand names are either service marks, trademarks, or registered trademarks of their respective owners.

Altair would like to thank the UCI Repository of Machine Learning Databases for the "Census" database distributed by us and used throughout the tutorial. Reference: Merz, C.J., & Murphy, P.M. (1998). UCI Repository of machine learning databases [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science.

The above third-party components that are licensed under, and subject to the permission of the MIT License or the Lesser General Public License (LGPL) are provided "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THESE COMPONENTS OR THE USE OF OTHER DEALINGS IN THESE COMPONENTS.

See also the Altair Data Analytics Software License Agreement.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

License Administration

License Types

Two licensing systems are supported for Altair Data Analytics applications:

- Altair Units licensing
- Registration-key licensing

The licensing system that your product is using depends on the terms of your contract.

Altair Units Licensing

Altair Units is a value-based license management system enabling metered usage of an entire suite of products. All Altair Data Analytics products have a capability to use a single pool of recyclable Altair Units.

If you purchased a Altair Data Analytics product with an Altair Units license, you will receive a license file and Altair Licensing Quick Installation Guide, which describes the installation and configuration of Altair License Manager with the License Server. The process of configuring the end users' machines with Knowledge Studio or Knowledge Seeker to connect to Altair License Server is described in the topic Configuring Altair Units Licensing.

Registration-Key Licensing

Registration keys issued by Altair License Administrator are used by the end users to acquire licenses online from the Altair License Fulfillment Server (ALFS) – a secure self-service license generation website.

The licenses are acquired from ALFS using the License Manager command in the Help menu of Altair Data Analytics applications. Registration-key licenses can also be acquired by email when network access to ALFS is not available.

Registration-key licenses can be subscription-based and evaluation licenses.

Named User Licenses (Legacy)

Named-User licenses are user-based node-locked licenses available for both the standalone and client/server configurations. Once installed, they are locked to the host machine name and the user ID of the user who installed them.

Concurrent User Licenses (Legacy)

Concurrent user licenses are available only for the client/server configuration of Altair software or the standalone configuration on a server accessed via Remote Desktop Connection or Citrix. They are node-locked licenses that will only operate on a specific

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

server for no more than a preset number of concurrent user connections. The total number of potential users is not limited. Concurrent licenses are suitable for customers who will have a high turnover of end users of the application.

A concurrent license has a restriction on the number of users that can simultaneously work with the software in client/server mode, but has no restriction on the identity of the user. Anyone having the client component (KS Workstation software) installed on their machine can connect to the server component and work with it, as long as the number of users connected at the same time does not exceed the limit imposed by the license.

See also:

Configuring Altair Units Licensing

Altair License Fulfillment Server (ALFS)

License Manager for Registration-Key Licenses

Altair Units Licensing

Configuring Altair Units Licensing

If you purchased a Altair Data Analytics product with an Altair Units license, the application must be configured to connect to an Altair License Server (unless you received a Standalone license). This section describes the process of configuring end users' machines with Knowledge Studio or Knowledge Seeker applications to connect to Altair License Server.

- If you are using Altair License Server on premises, the installation of the Altair License Manager software on the server is described in the Altair License Quick Installation Guide that your organization received from Altair. In what follows, it is assumed that Altair License Manager is already installed and configured.
- If you are using a Managed Altair Units license, where the License Server is hosted by Altair, follow the steps in the topic Authorize Machine for Managed Altair Units License before performing the configuration steps below.

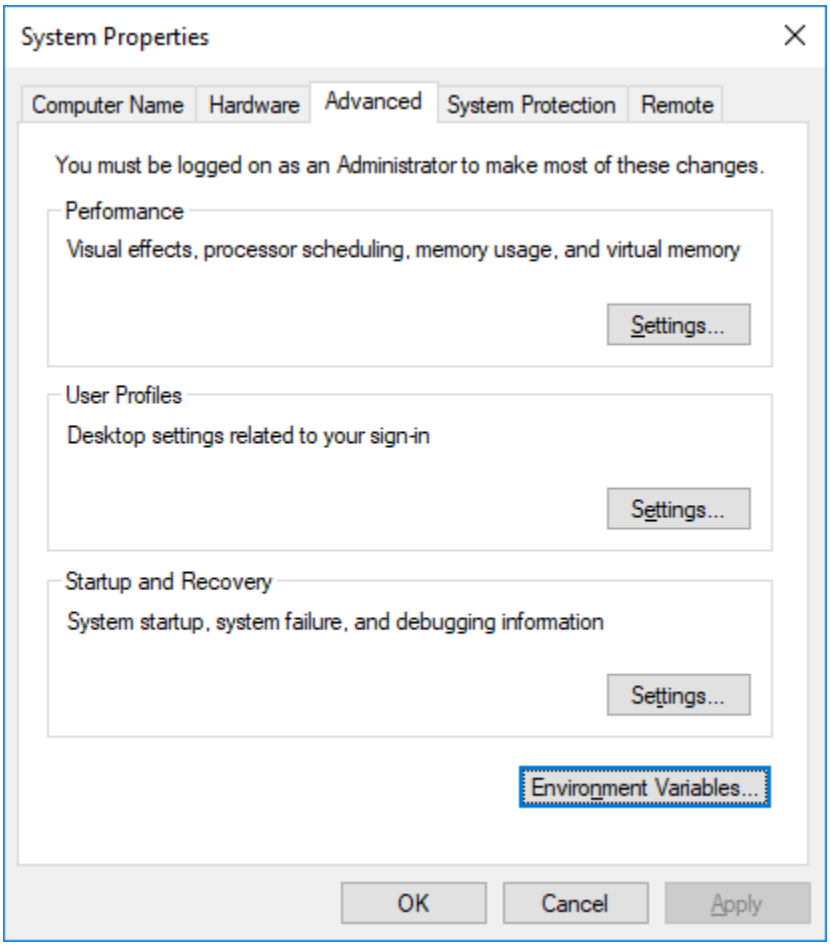
There are three configurations of the application:

- Standalone
- Client/Server for Windows Server
- Client/Server for Linux

Standalone

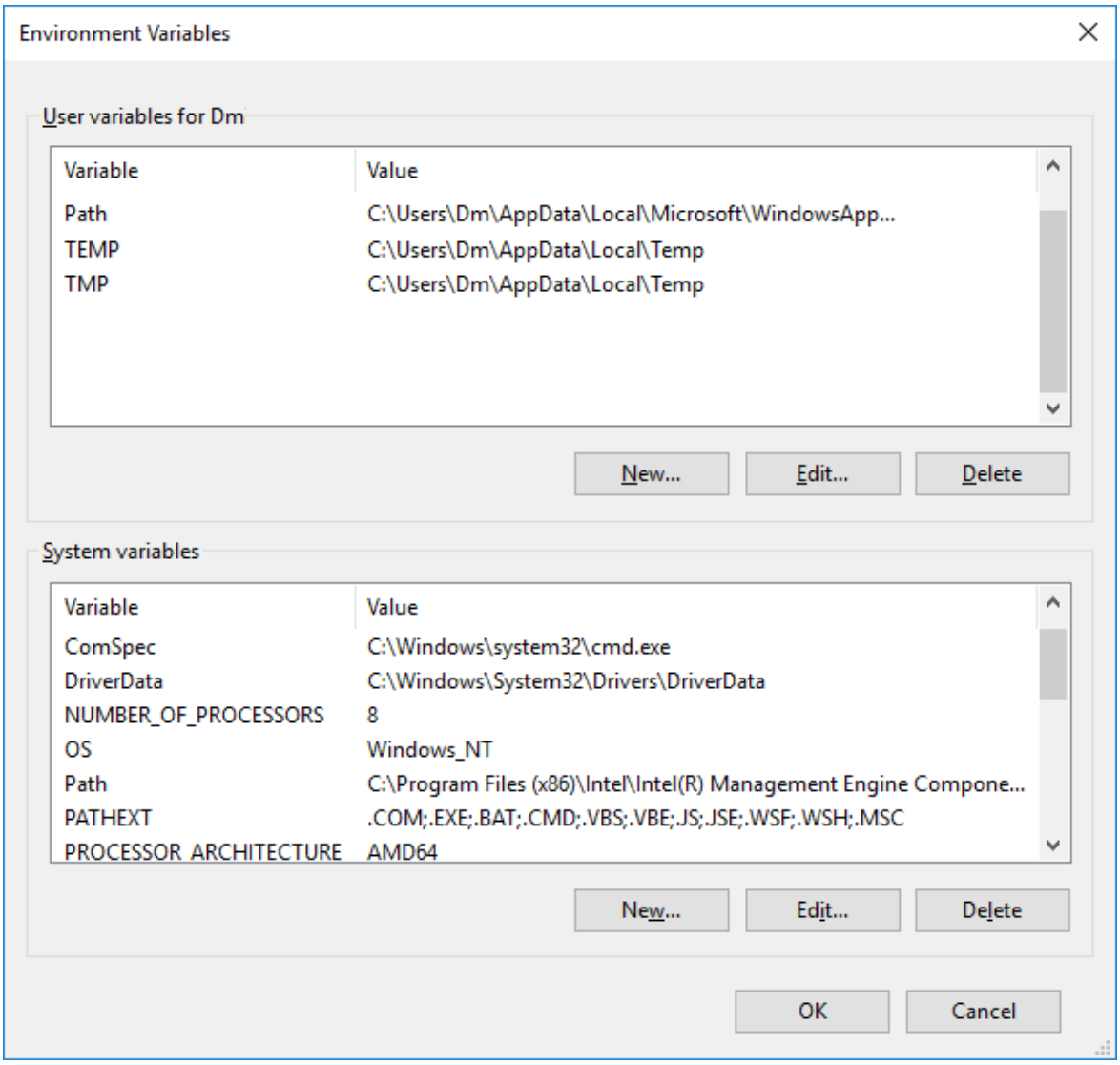
From the Windows Control Panel, select **System** and click **Advanced System Properties**.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

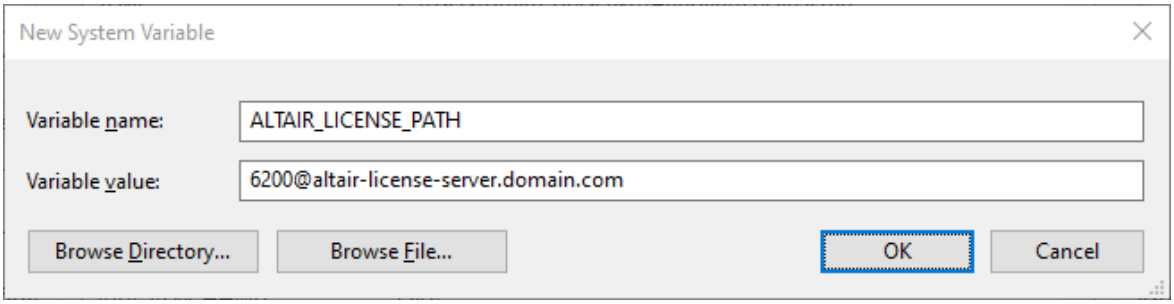


In the Advanced tab of the System Properties dialog, click the **Environment Variables** button.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click the **New** button in the *System variables* section or in the *User variables* section, depending on whether the Altair license server connection is required for all users on this machine or only the user ID currently logged in.



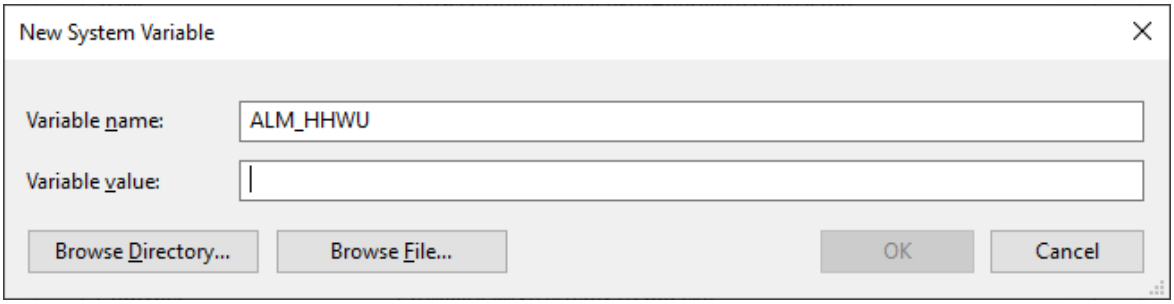
- Set *Variable name* to **ALTAIR_LICENSE_PATH**.
- For the *Variable value* field:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- If you are using Altair License Server, enter its address and port number. The format is: **NNNN@<server-name>**, where NNNN is the port number and <server-name> is the Altair License Server host name. Example: *6200@altair-license-server.domain.com*.
- If you are using Managed Altair Units, set the value to false

Click **OK** to create the variable.

Click the **New** button in the same section as ALTAIR_LICENSE_PATH.

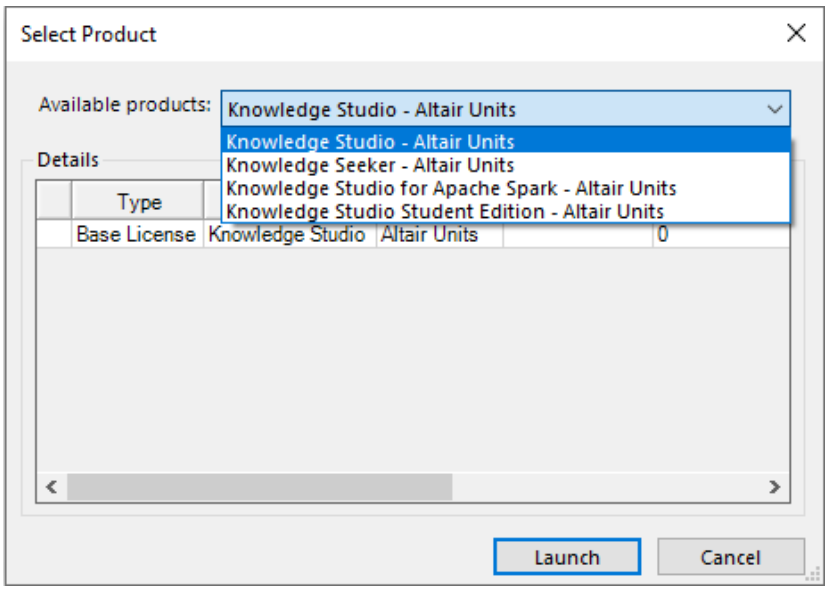


- Set *Variable name* to **ALM_HHWU**
- To use Managed Altair Units, set the value of ALM_HHWU to **true**. Otherwise, set it to **false**, assuming ALTAIR_LICENSE_PATH is defined and set to a value other than false.

Click **OK** in all dialogs to save the changes and exit the Advanced System Properties dialog.

Start the KS Workstation application.

The Product Selection window will prompt you to select one of the three products: Knowledge Seeker, Knowledge Studio, or Knowledge Studio for Apache Spark.



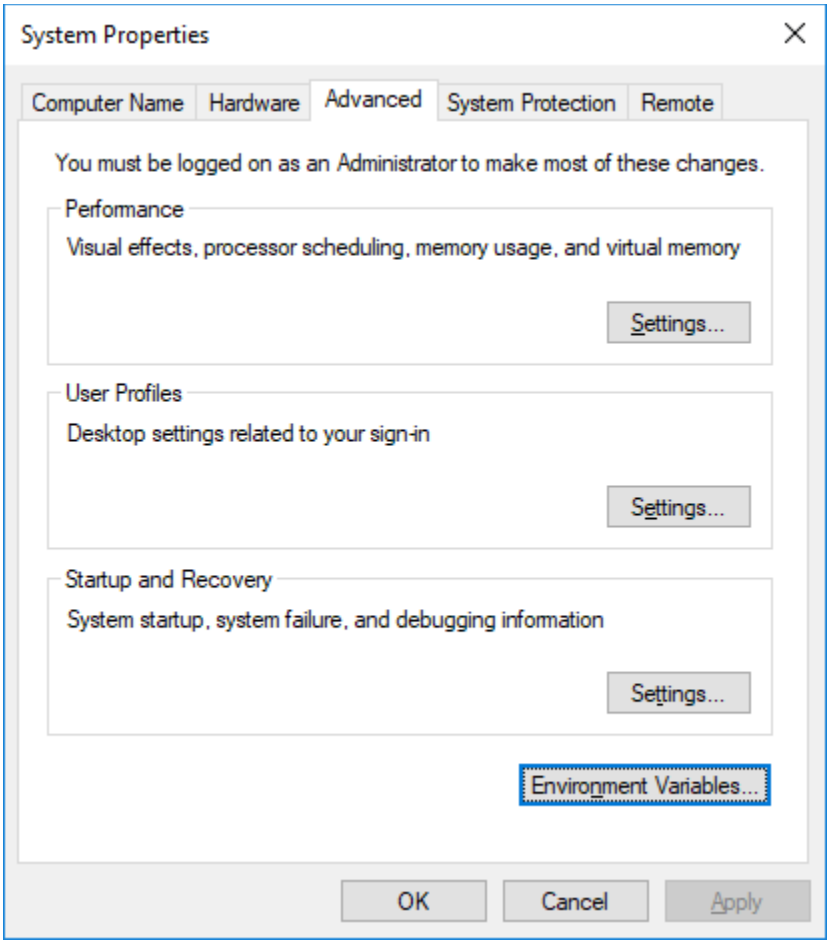
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Select the desired product and click **Launch**.

To switch to a different product, select the Connect command under the File menu and click OK. The Product Selection dialog will open. Select the desired product and click **Launch**.

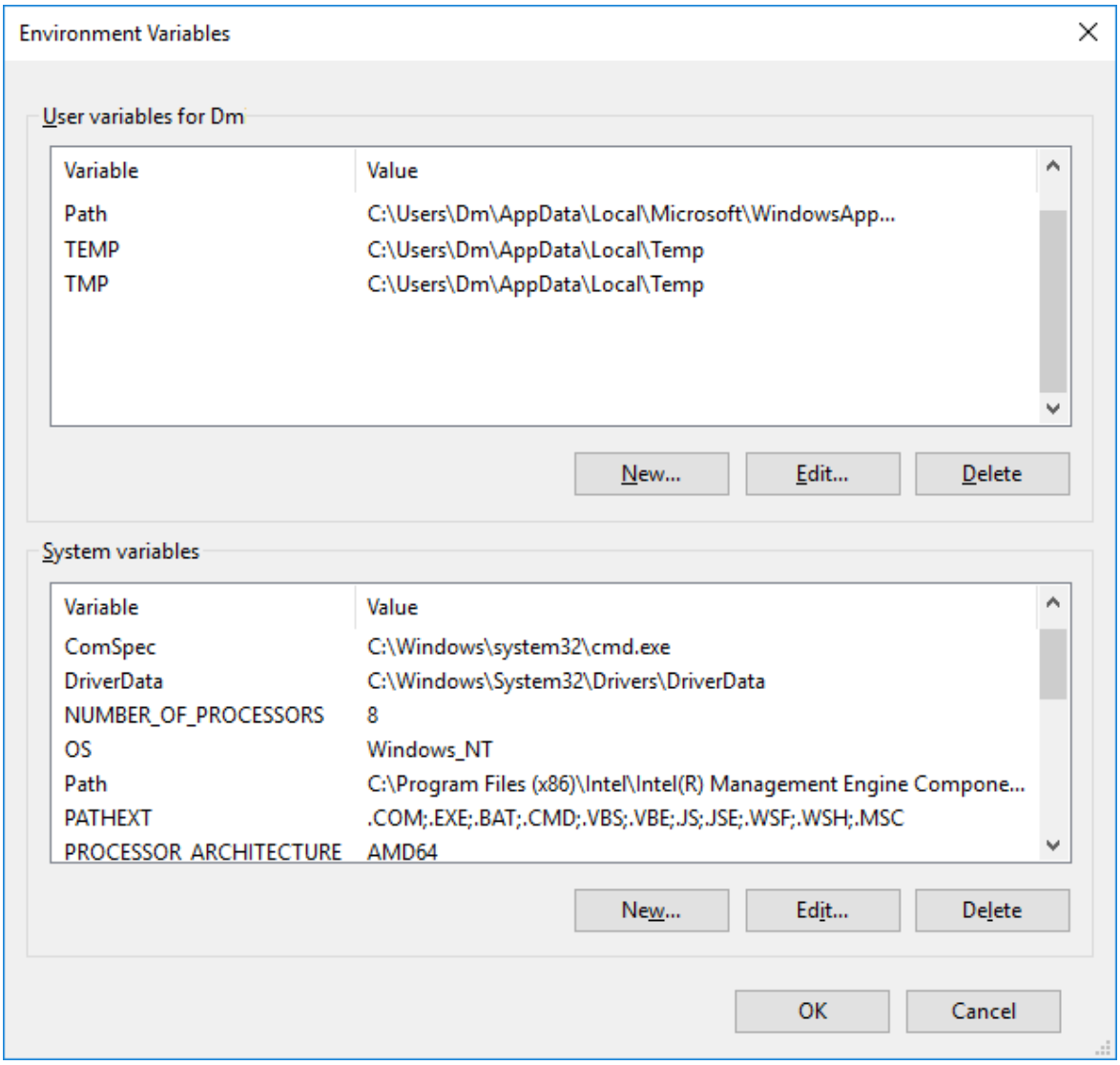
Client/Server for Windows

From the Windows Server Control Panel, select **System** and click **Advanced System Properties**.

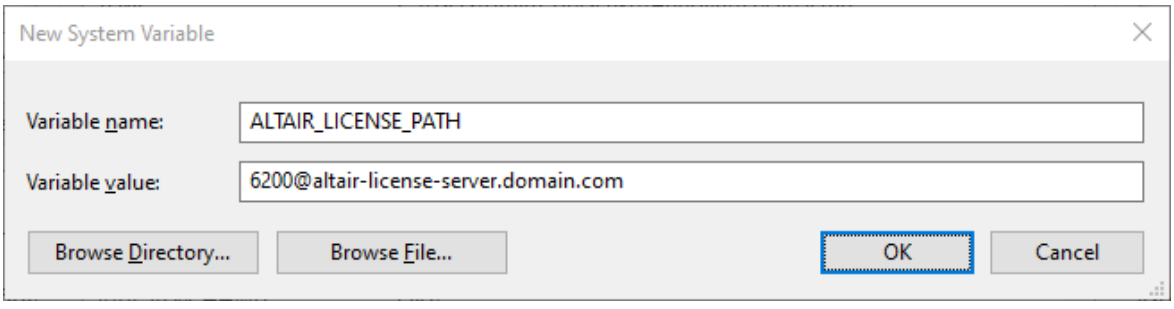


In the Advanced tab of the System Properties dialog, click the **Environment Variables** button.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click the **New** button in the *System variables* section.



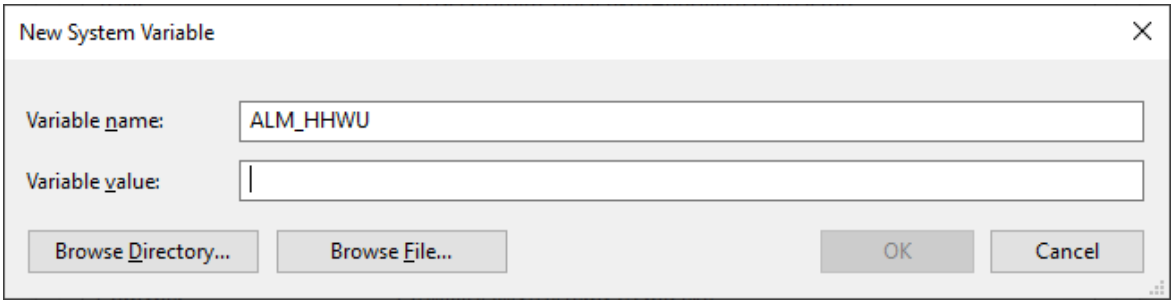
- Set *Variable name* to **ALTAIR_LICENSE_PATH**.
- For the *Variable value* field:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- If you are using Altair License Server, enter its address and port number. The format is: **NNNN@<server-name>**, where NNNN is the port number and <server-name> is the Altair License Server host name. Example: *6200@altair-license-server.domain.com*.
- If you are using Managed Altair Units, set the value to false

- Click **OK** to create the variable.

Click the **New** button in the same section as ALTAIR_LICENSE_PATH.



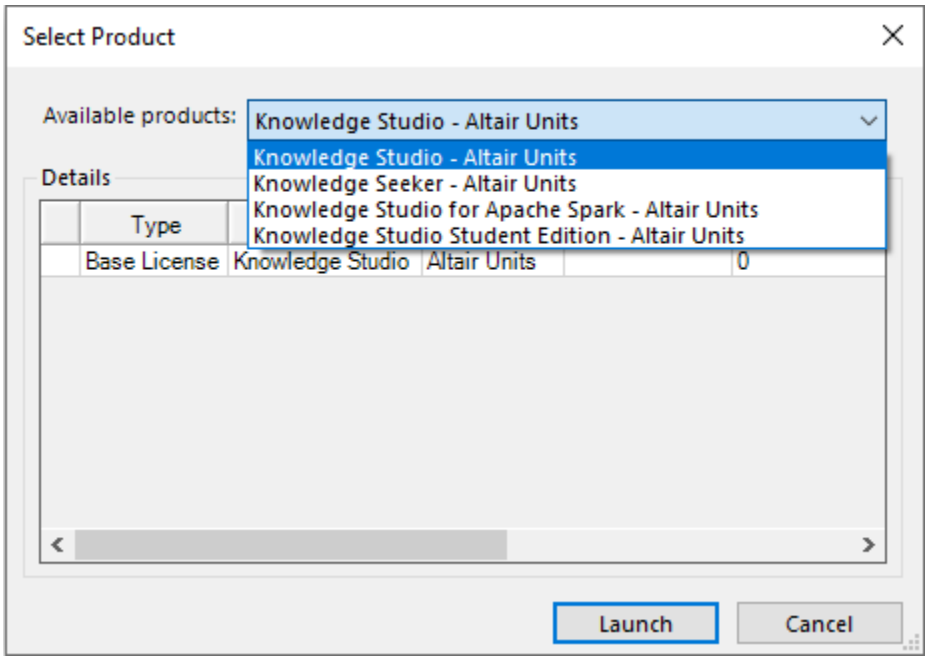
- Set *Variable name* to **ALM_HHWU**
- To use Managed Altair Units, set the value of ALM_HHWU to **true**. Otherwise, set it to **false**, assuming ALTAIR_LICENSE_PATH is defined and set to a value other than false.

Click **OK** in all dialogs to save the changes and exit the Advanced System Properties dialog.

Start the KS Workstation application on any client machine.

Connect to KS Server. The Product Selection window will prompt you to select one of the three products: Knowledge Seeker, Knowledge Studio, or Knowledge Studio for Apache Spark.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Select the desired product and click **Launch**.

To switch to a different product, select the **Connect** command under the **File** menu, enter your Windows password and click OK. The Product Selection dialog will open. Select the desired product and click **Launch**.

Client/Server for Linux

Define the environment variable ALTAIR_LICENSE_PATH in the KS Server startup script **ks_start** as described below. For information about ks_start, see the section “Configuring KS Server on Linux” in the Deployment Guide.

Set the value ALTAIR_LICENSE_PATH to the address of the Altair License Server. The format is: **NNNN@<server-name>**, where NNNN is the port number and <server-name> is the Altair License Server host name or IP address. Example:

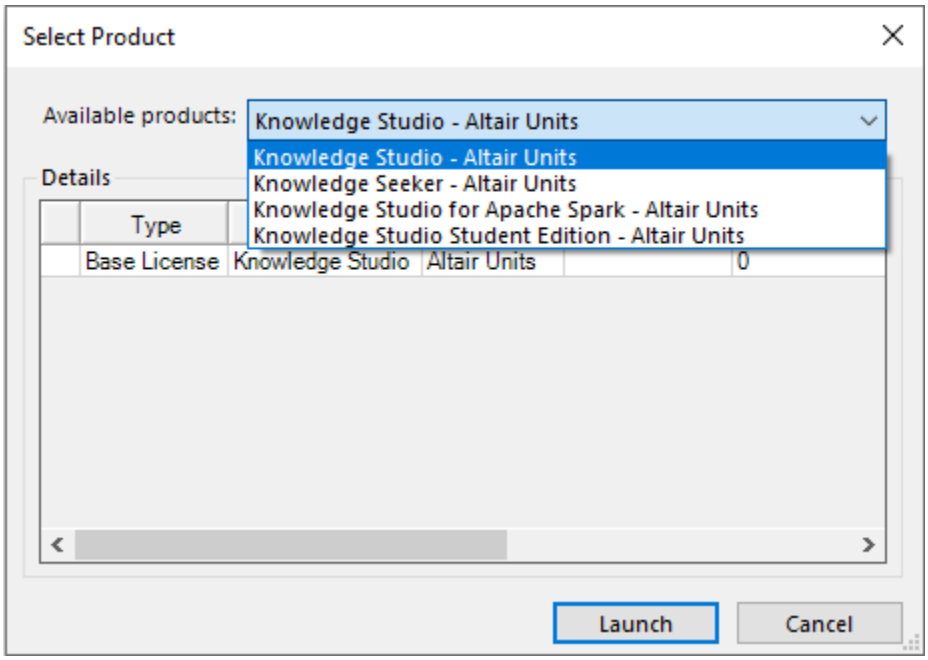
```
export ALTAIR_LICENSE_PATH=6200@als-server
```

Save the changes in ks_start and restart KS Server as described in the section “Starting and Stopping the KS Service” in the Deployment Guide.

Define ALTAIR_LICENSE_PATH the same way in the script ksenv.sh used for troubleshooting purposes. This script is in the same directory as ks_start.

Start KS Workstation on any client machine and connect to KS Server. The Product Selection window will prompt you to select one of the three products: Knowledge Seeker, Knowledge Studio, or Knowledge Studio for Apache Spark.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Select the desired product and click **Launch**.

To switch to a different product, select the **Connect** command under the **File** menu, enter your Linux password and click OK. The Product Selection dialog will open. Select the desired product and click **Launch**.

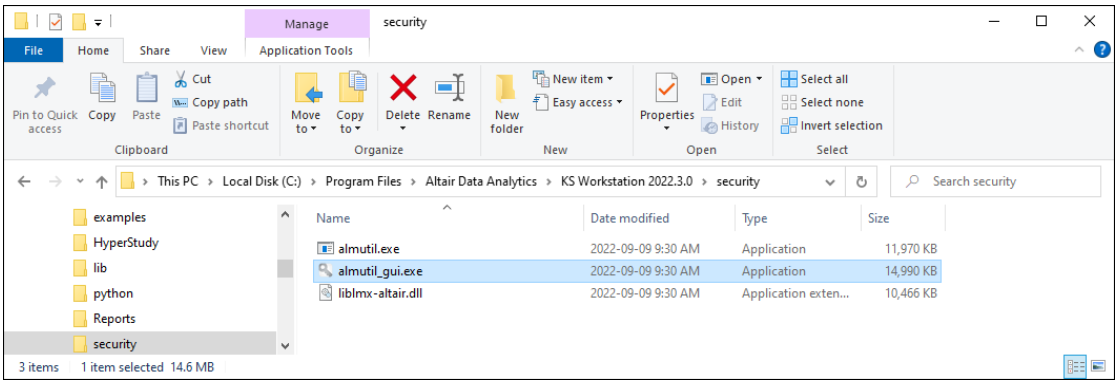
Authorize Machine for Managed Altair Units License

This topic describes how to authorize your machine for a Managed Altair Units license, where the License Server is hosted by Altair. This step is not required if you are using Altair License Server on premises.

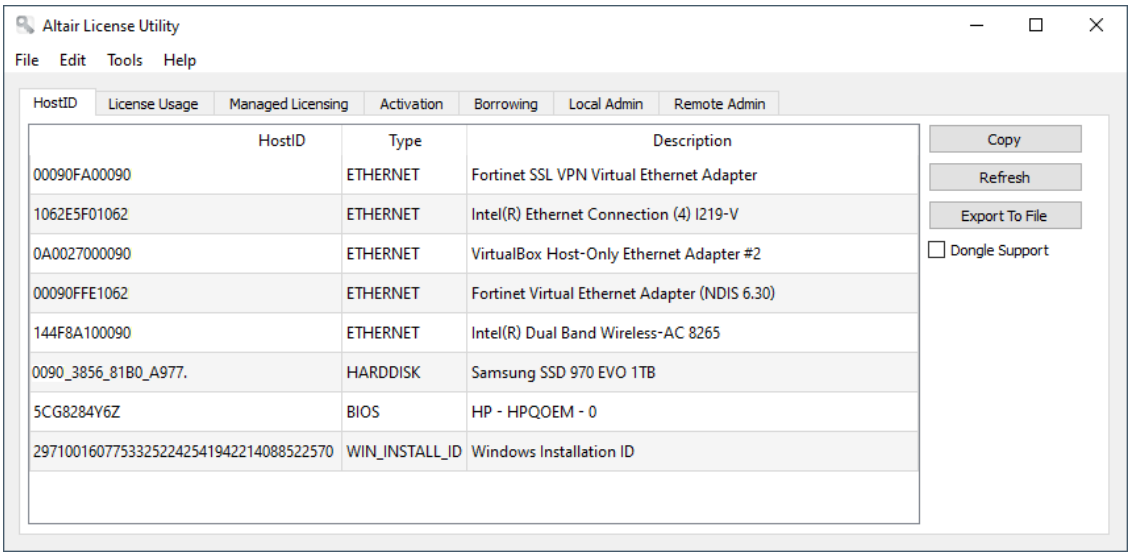
Standalone configuration for a single user:

1. If you have not registered on the AltairOne portal yet, ask a user with Altair license admin role at your company to add you as a user at https://AltairOne.com. You will receive an email from Altair with a link to activate your AltairOne account. Click the activation link in your email and register on the AltairOne portal.
2. Open the file browser and navigate to the folder `C:\Program Files\Altair Data Analytics\KS Workstation 2022.3.0\security`

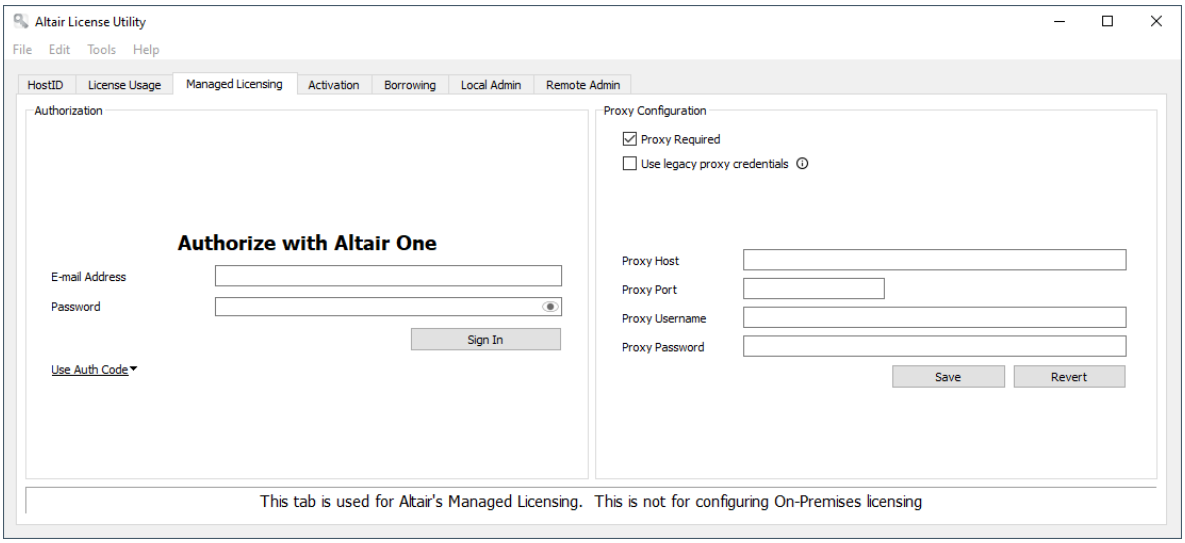
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



3. Double-click on the **almutil_gui**. The Altair License Utility window opens:



4. Go to the **Managed Licensing** tab of Altair License Utility window:



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

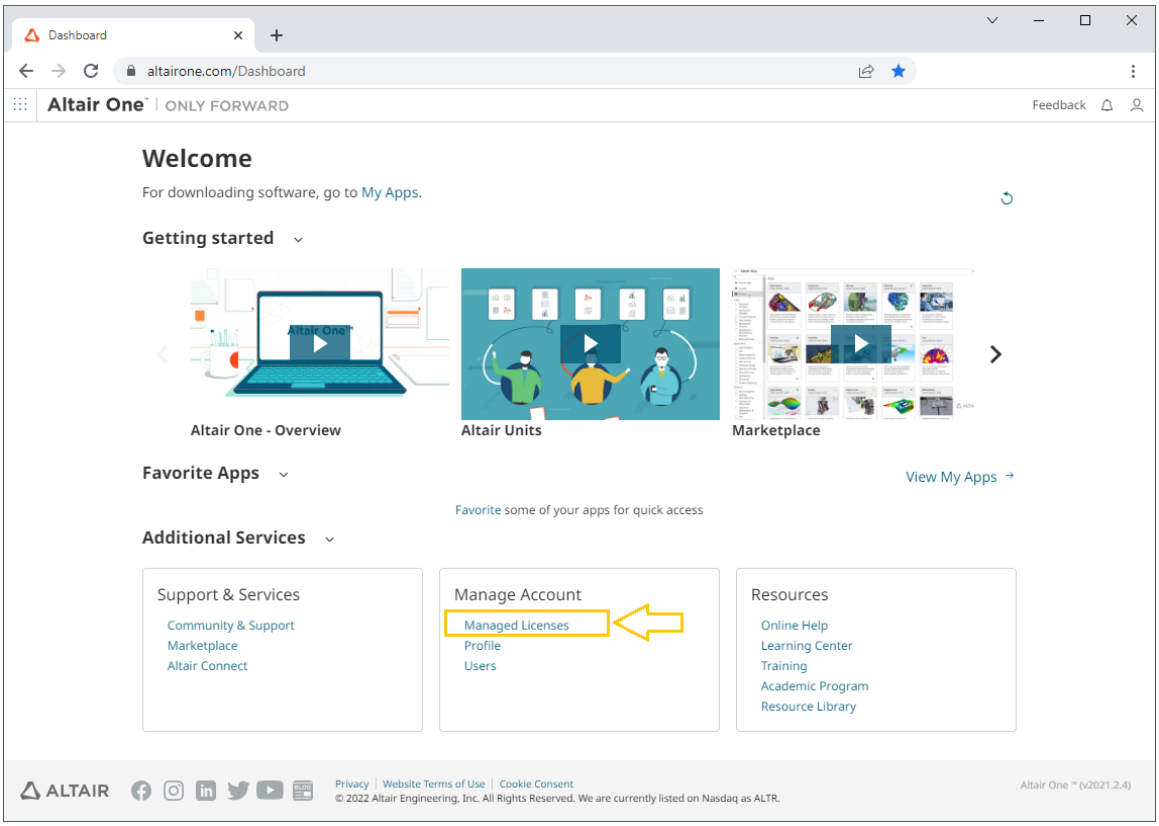
5. Specify your AltairOne credentials: your email address and password that you used to register at the AltairOne portal. Click **Sign In**.

   Alternatively, if you are not registered at the AltairOne portal and received an authorization code an Altair license admin user at your company, click Use Auth Code, specify the authorization code received from this user and click **Authorize**.

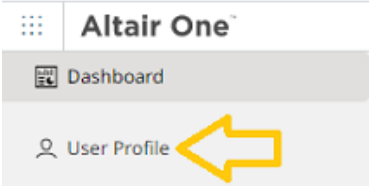
   You will get the message: “Successfully authorized this machine with the Hosted HWU system”. Click **OK**.

Client/Server configuration or Standalone configuration for multiuser access:

1. If you have not registered yet on the AltairOne portal, ask a user with Altair license admin role at your company to add you as a user at https://AltairOne.com. You will receive an email from Altair with a link to activate your Altair One account. Click the activation link in your email and register on the Altair One portal.
2. Log on to https://altairone.com/Dashboard. In the Manage Account section of the Altair One Dashboard page, click **Managed Licenses**, as shown below.

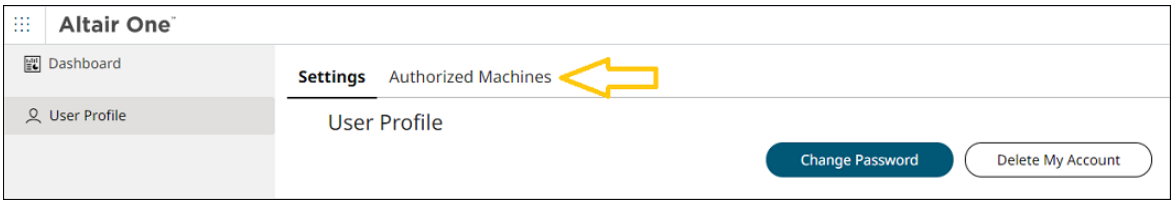


3. In the left pane, click **User Profile**:

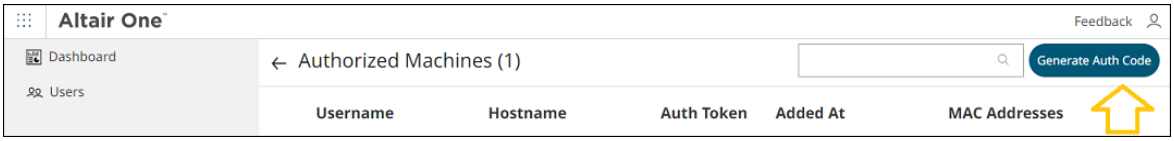


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

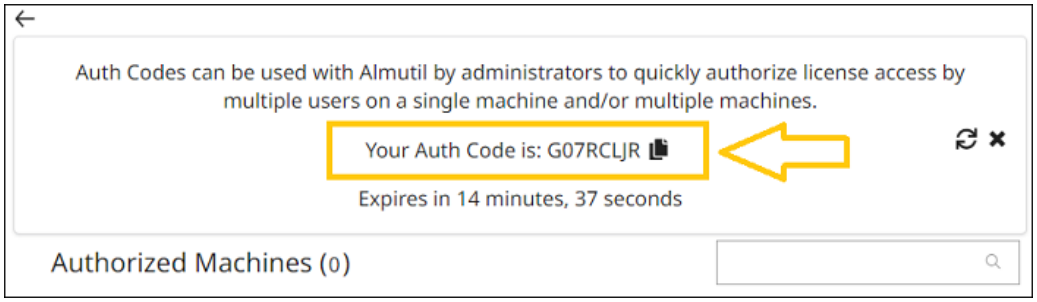
4. Click **Authorized Machines** in the User Profile page:



5. In the Authorized Machines page, click **Generate Auth Code** at the top right corner:



6. This will generate an authorization code to authorize a machine for the Altair license. Copy the code to the clipboard. Note: The code is valid for 15 minutes, so you must perform the next 7 steps within 15 minutes of generating the code.



7. After copying the code:

- For the Standalone configuration on Windows: run Command Prompt as administrator and change to the folder `C:\Program Files\Altair Data Analytics\KS Workstation 2022.3.0\security`
- For the Client/Server configuration on Windows: run Command Prompt as administrator and change to the folder `C:\Program Files\Altair Data Analytics\KS Server 2022.3.0\security`
- For the client/server configuration on Linux: Change to `/usr/local/altair/licensing15.1/bin/` or to a directory that contains **almutil**. If you cannot find almutil on your system, please contact Altair Customer Support.

8. Run the following command as administrator, where *NNNN* is the authorization code generated at the previous step:

```
almutil -a1auth -system -code NNNNN
```

The almutil tool can also be used to test the connectivity with AltairOne servers and configure proxy settings for use with AltairOne. Run `almutil` to view the detailed help information on the usage of the utility.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

9. Once you get the message that the machine was successfully authorized, restart the KS Server service (in the client/server configuration) or KS Workstation (in the Standalone configuration).

Make sure you set the environment variables for Managed Units as described in the section Configuring Altair Units Licensing.

Registration-Key Licensing

Altair License Fulfillment Server (ALFS)

ALFS is a secure self-service website used to acquire licenses and view license usage information. A Registration Key is required to log in to ALFS. Altair Data Analytics applications access ALFS through the License Manager feature available from the Help menu. ALFS can also be accessed directly from a Web browser at https://license.angoss.com/.

ALFS keeps a copy of all acquired licenses. In the event of a system failure or accidental deletion of the license on your computer, the license can be retrieved from ALFS and reinstalled.

See also:

Registration Keys

License Manager for Registration-Key Licenses

License Types

Registration Keys

After you purchase, renew, or request an evaluation of a Altair Data Analytics product with a registration-key license, Altair sends you a registration key that provides access to your license(s). A registration key is a string about 30 characters long.

Types of Registration-Key Licenses

Named User Licenses

Named-User licenses are user-based node-locked licenses available for both the standalone and client/server configurations. They will only operate on a specific machine for a specific user ID.

CPU & RAM-based Licenses

CPU & RAM-based licenses are legacy licenses available only for Windows Server and Linux platforms. These node-locked licenses tied to the machine ID of the host server activate the software only if the number of logical CPUs and the amount of RAM do not exceed the limits specified in the license. The total number of users may or may not be limited, depending on the contract terms.

See also:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Altair License Fulfillment Server (ALFS)
License Manager for Registration-Key Licenses
License Types

Prerequisites for Installing Registration-Key Licenses

Administrative rights are not required to install licenses. The end users must have Read & Write permissions for the license directory.

If an IT administrator assists the intended user in activating the license, the administrator should log in with the user ID of the intended user.

The path to the license directory is displayed in the License Manager dialog, which can be accessed from the Help menu.

Client/Server Configuration

In the client/server configuration, the license resides on the server. Before installing the license, make sure that KS Workstation is configured to connect to the server.

1. Start KS Workstation using the desktop icon or the Altair Data Analytics program folder in the Windows Start menu.

2. You will be prompted for your password to log in to KS Server. Enter your password (your Windows or Linux user password, depending on the server OS).

If you are not prompted for your password:

- Select the **Connect** command from the **File** menu.
- In the Establish Connection dialog:
  - Set the Mode to Client/Server Login.
  - Enter the server name or IP address and the port number. The default port is 5470.
  - Specify the working directory if known; otherwise, leave the field blank. When the connection is established, the file browser will let you browse the server and select a directory if it was not specified correctly.
  - Enter the user ID and password (your Windows or Linux user credentials, depending on the server OS).

3. Click OK to connect. After the connection has been established, the License Manager window will open. It can also be accessed from the Help menu.

Proceed to the topic License Manager for Registration-Key Licenses.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

License Manager for Registration-Key Licenses

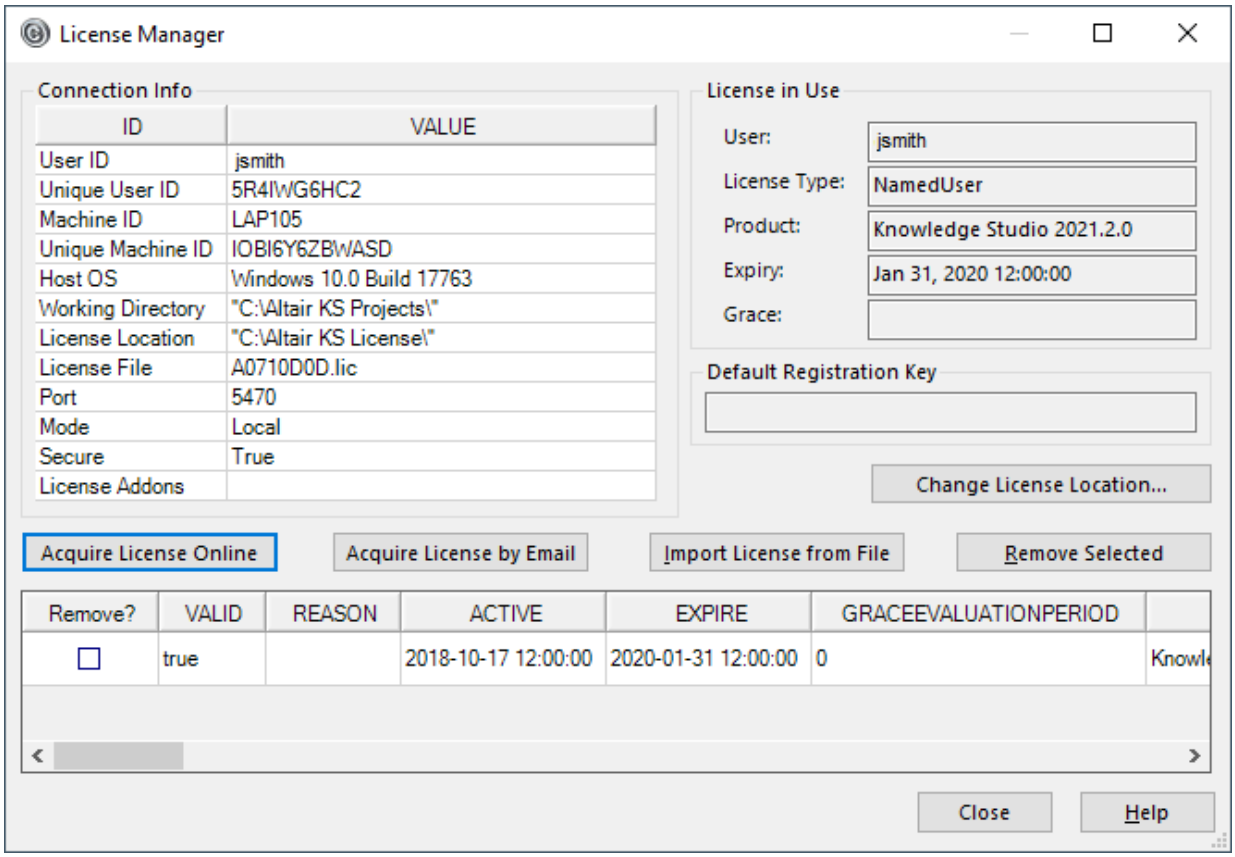
The **License Manager for registration-key licenses** can be accessed from the **Help** menu of the application.

> *Note: This License Manager dialog is not to be confused with the Altair License Manager software, which is a part of the Altair Units Licensing system.*

To prepare for license installation via the License Manager interface, see the topic Prerequisites for Installing Registration-Key Licenses.

The License Manager for registration-key licenses can be used to perform the following tasks:

- Install a license using one of the three available options: acquire the license online, by email, or import from file.
- Remove license(s)
- View currently installed licenses
- Obtain the information required for license troubleshooting or requesting licenses.
- Access the Altair License Fulfillment server (ALFS) to view the license usage and history for a given registration key.
- Change the license location (in the Standalone mode only)



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Installing Licenses Using the License Manager

NOTE: ***For the Client/Server configuration only****:* Connect from your KS Workstation (client) to KS Server BEFORE generating or installing the license(s). If you are an administrator assisting the intended user, connect to the server using the credentials of the intended user. If you are not prompted for the password upon launching KS Workstation, close all dialogs and select **Connect** command from the **File** menu.

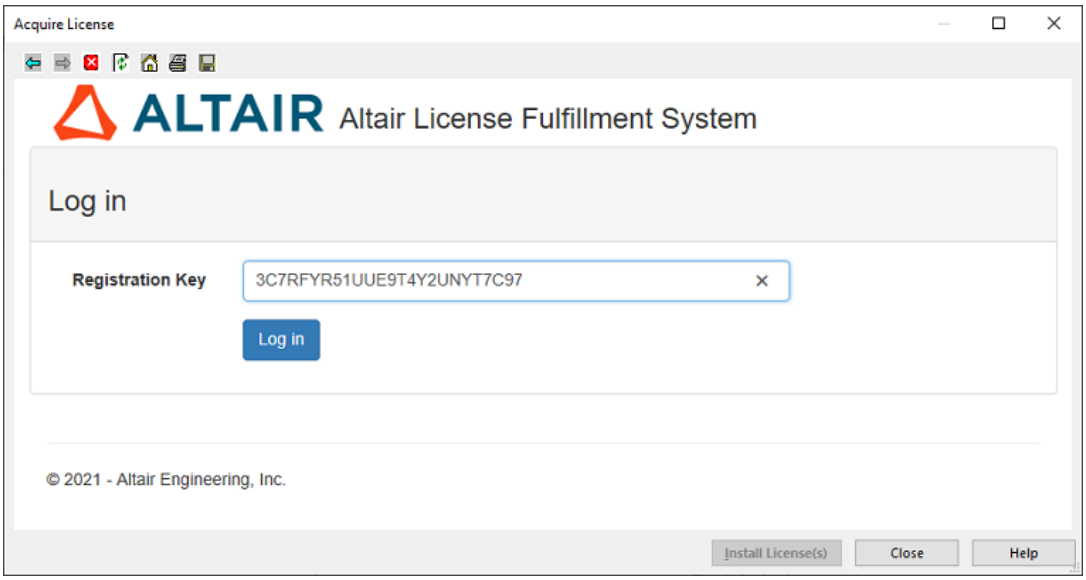
Acquiring Licenses Online

Conditions:

- Internet access is required on the end user's machine.
- The ALFS website must be accessible.
- Email service is not required on the end user's machine.

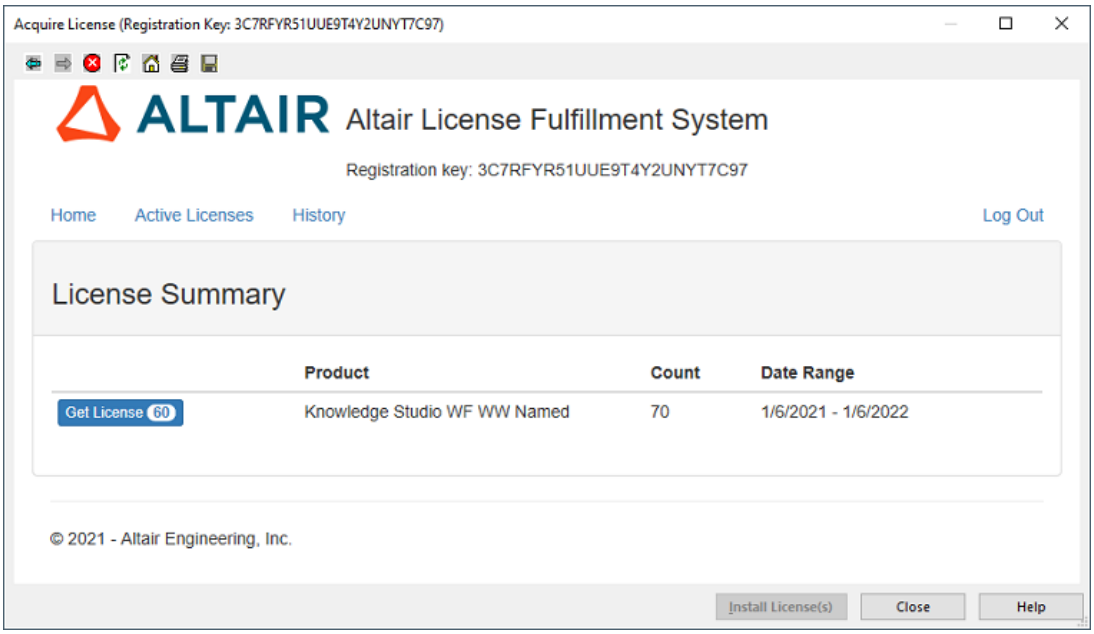
Procedure:

- Clicking the **Acquire License Online** button in License Manager navigates the user to the secure Altair License Fulfillment Server (ALFS), where purchased licenses can be generated, installed, or retrieved.



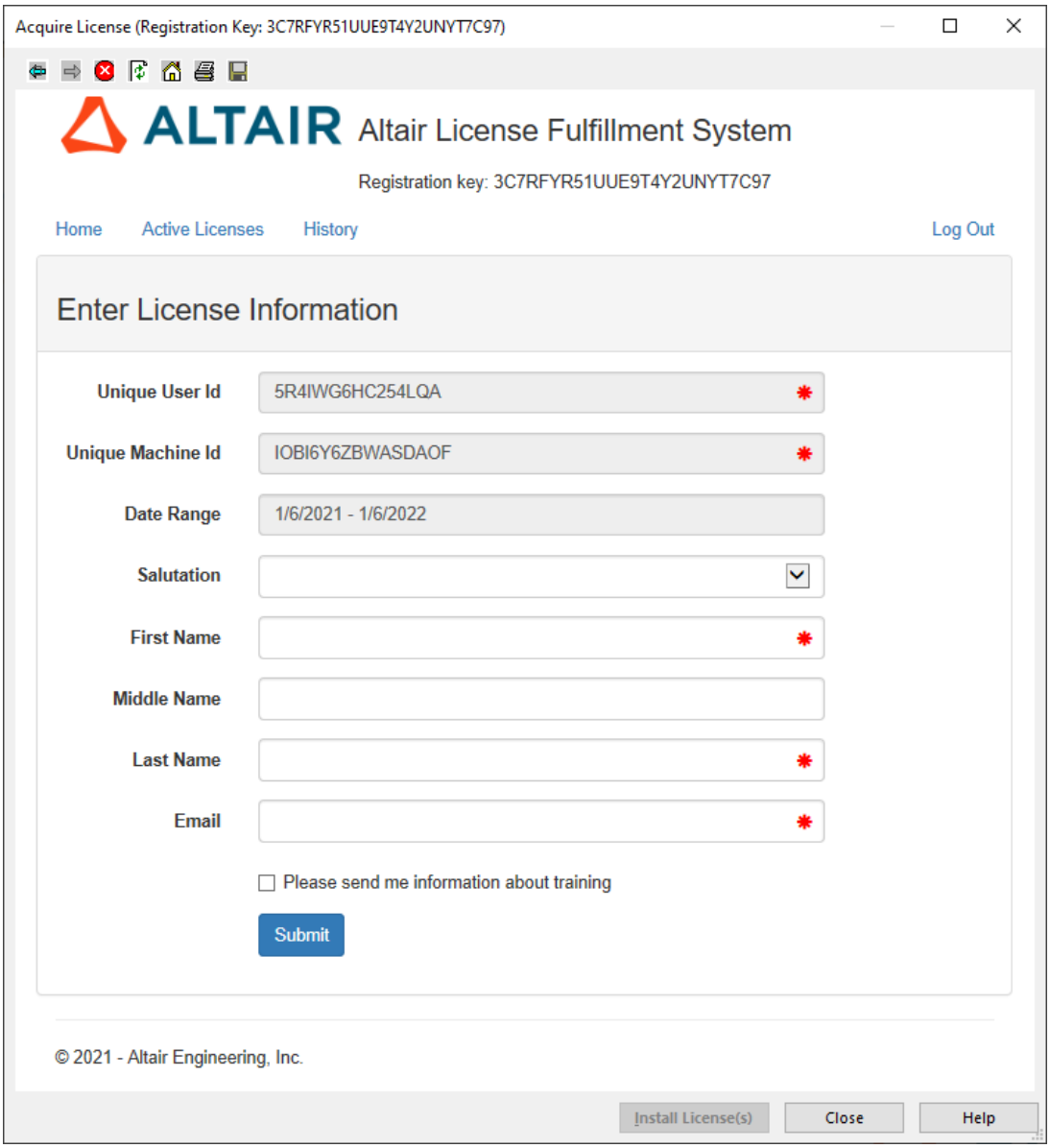
- The user logs on to ALFS using the Registration Key provided by Altair. The Registration Key should be auto-populated for end users when they navigate to ALFS via the License Manager, if a registration key was specified during the installation of KS Workstation.
- NOTE: In the case of named-user licenses: the licenses generated on the ALF site accessed via the License Manager are tied to the user ID of the user currently logged in to the KS Workstation host machine (in the Standalone configuration) or KS Server (in the client/server configuration). Before acquiring a license, log in as the intended user of the application.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- After logging in to ALFS, a screen appears that lists all purchased licenses associated with the Registration Key and indicates the number of licenses available. To generate a license for a product with an available license, click on the Get License button next to the product name.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

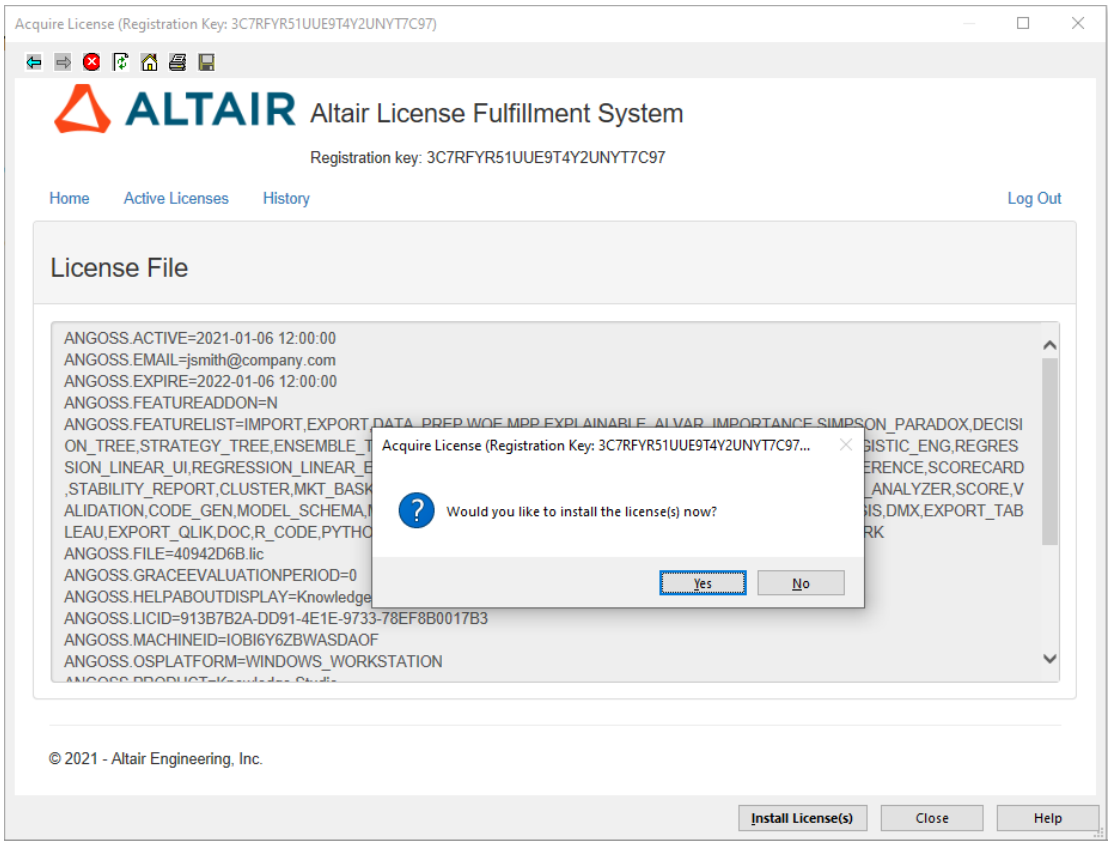


- A form will appear where information needed to generate the license can be entered.
- The Unique Machine ID and Unique User ID fields will be auto-populated by the application. If they are not populated, then the Unique Machine ID and Unique User ID can be copied and pasted from the Connection Info box of the License Manager.
- Note: This Unique Machine ID and Unique User ID are not the actual machine ID and user ID and but encrypted versions of them.
- The ‘Product’, ‘Date Range’, and the ‘Number of Concurrent Users’ (only visible in the case of a concurrent-user license) fields are also auto-populated and are not editable.
- First Name, Last Name, and Email Address should be entered, as they are required in order to generate the license.
- For concurrent-user licenses only:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Specify the folder for the concurrency lock file in the *Concurrency Lock File Location* field. In most cases we recommend specifying the license folder. On Windows server, it is usually C:\Program Files\Altair Data Analytics\License, while on Linux it is the **license** subdirectory of the installation folder (for example, /opt/Altair-Data-Analytics/license). The concurrency lock file is required for the controlling concurrent connections. This location needs to have read and write permissions for all users of the application.

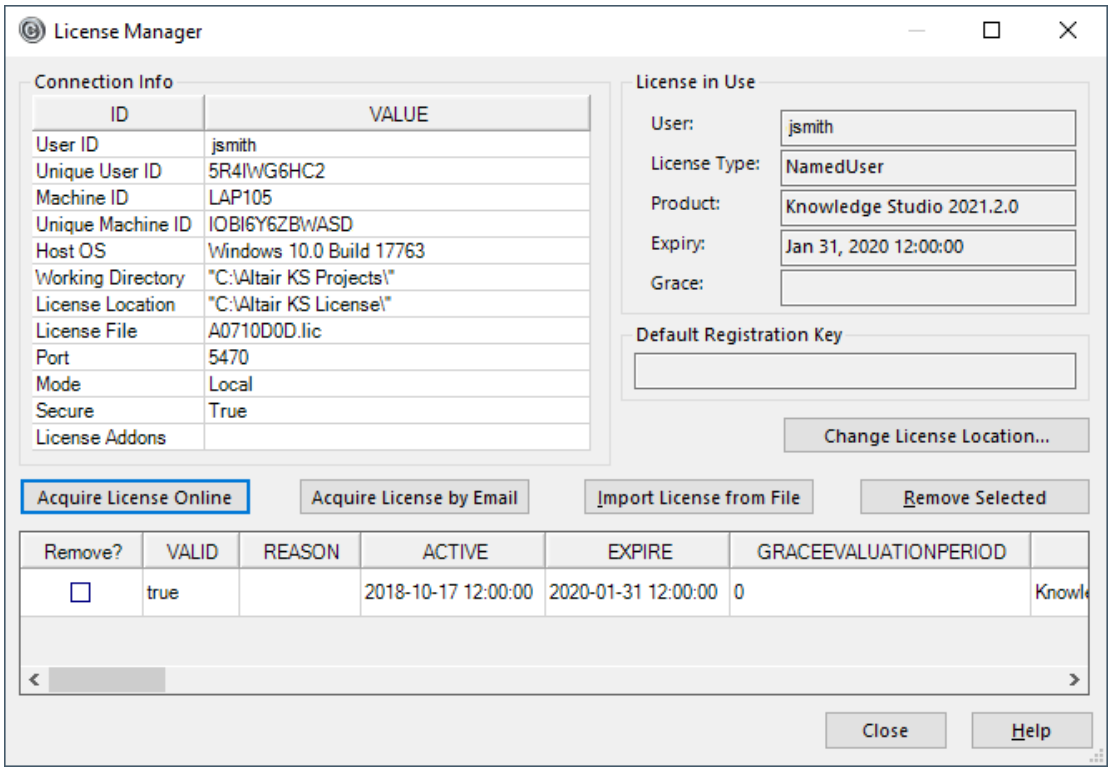
- Click the Submit button after the required fields (indicated by an asterisk) have been entered.
- Review the information that is being entered. Click Submit after verifying the correct details have been entered.
- The user is taken to the screen that displays the license content:



Clicking the Install License(s) button in the lower right-hand side of the window will install the license.

- The user will be returned to the License Manager window with the newly installed license(s) displayed in the lower frame of the dialog as a white row with the columns Remove?, VALID, REASON, etc.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- Close the License Manager dialog.

The menus File, Edit, View, Insert, Tools, Window, and Help should now be enabled.

Acquiring Licenses by Email

Conditions:

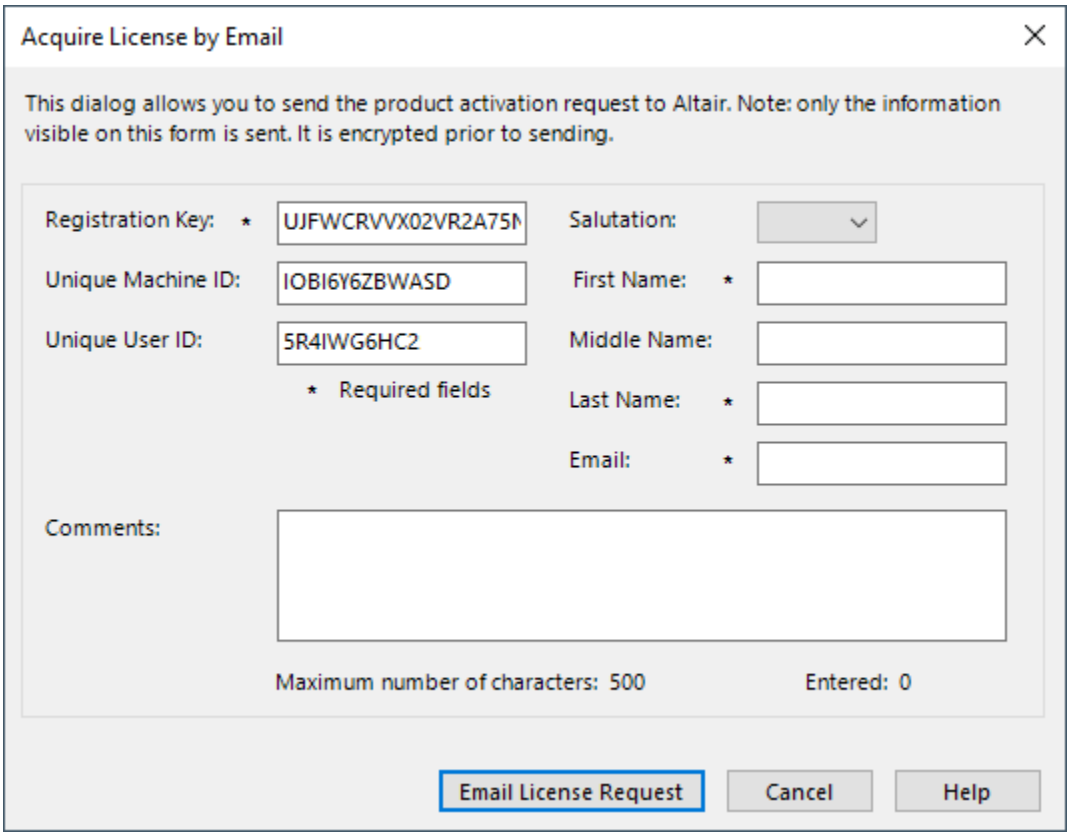
- Email service is required on the end user's machine.
- Internet access is not required on the end user's machine.

Clicking **Acquire License by Email** will generate an email message to be sent to Altair Customer Support with the information required to create a license.

Note: *By default, User based licenses will become locked to the user profile that was logged on at the time of license generation. This does not concern Concurrent licenses.*

Warning: If the licenses to be installed are for the client/server configuration, please ensure to connect from your instance of KS Workstation to the server component BEFORE generating or installing any license. To connect to the server, select **File | Connect** from the application menu.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The Registration Key, Unique Machine ID, and Unique User ID fields should already be populated. These fields should not be changed.

The Registration password is blank by default, so this field should be left blank.

1. Enter your first name, last name, and email address.

2. Enter any comments if necessary.

3. Click the **Email License Request** button. The license request will be sent to Altair Customer Support at productactivation@datawatch.com.

4. Once you receive the email from productactivation@datawatch.com in response to your request, open the attached license file in Notepad or any other text editor and copy its contents to the clipboard.

9. Run KS Workstation. Click the button "Import License from File" in the License Manager window. Paste the license content into the pop-up window and click OK.

Note: *When copying and pasting the license text, make sure the text is copied as is.*

The license should now be installed and shown in the lower frame of the License Manager. Close the License Manager.

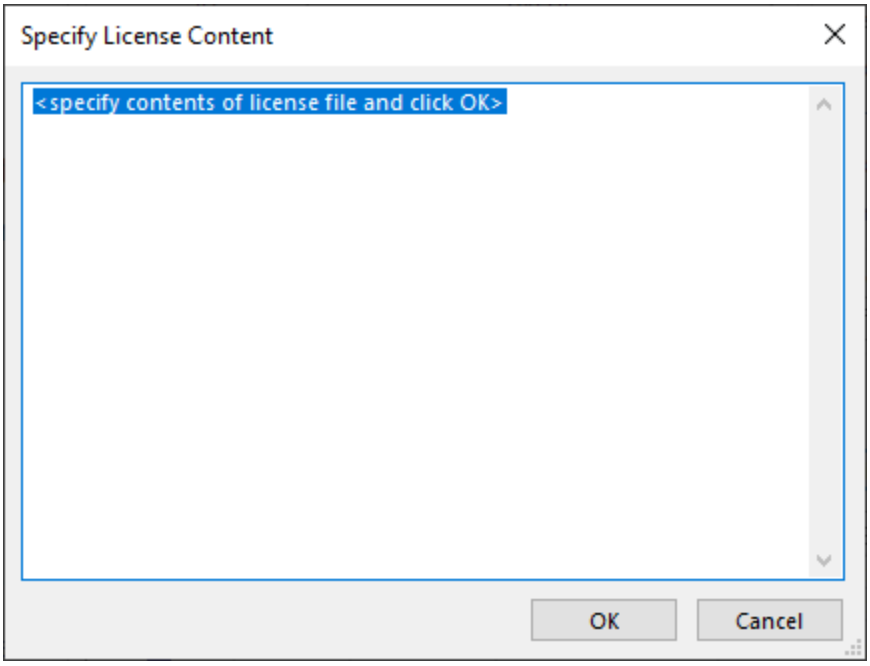
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Import License from File

Conditions:

- Internet access is not required on the end user's machine.
- Email service is not required on the end user's machine.

Clicking the **Import License from File** button in License Manager opens the Specify License Content dialog.



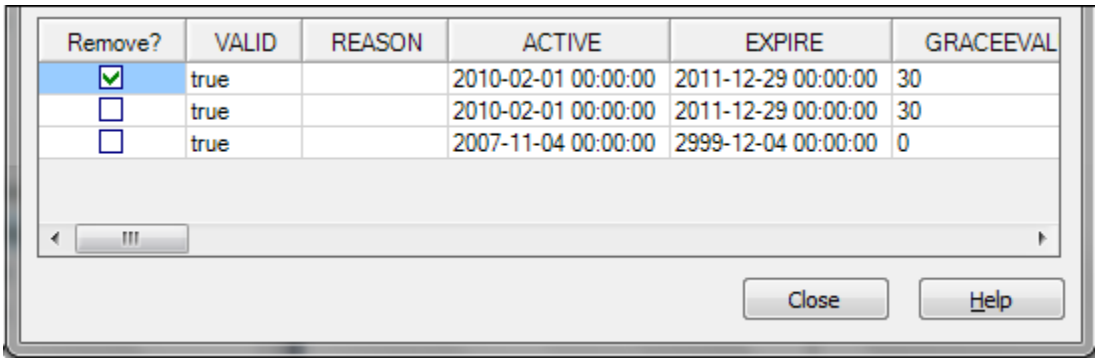
The license text copied from an email or from ALFS should be pasted into this dialog. Clicking OK will install the license and return the user to the License Manager with the newly installed license visible.

Removing Licenses

Licenses can be uninstalled and removed from the application by using the **Remove Selected** button of the License Manager. A confirmation email will be sent to Customer Support at productactivation@datawatch.com, indicating the license to be removed.

- Click on the checkbox in the 'Remove?' column of the license to be removed.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- Multiple licenses can be removed at the same time if needed by selecting and checking their respective check boxes.
- After all licenses to be removed have been selected, click the 'Remove Selected' button.
- The Confirm License Deletion dialog will appear giving the user a chance to review the license(s) that will be removed.
- Users have the option to send a confirmation email to Customer Support. By default, the checkbox at the bottom of the Confirm License Deletion dialog is checked and an email will be sent to Customer Support.
- Clicking OK in the Confirm License Deletion dialog will remove the selected license(s) and send a confirmation email to Customer Support if the checkbox had not been unchecked.
- The user will be returned to the License Manager with the list of licenses installed, no longer containing the license(s) that were removed.

Acquiring Licenses by Email

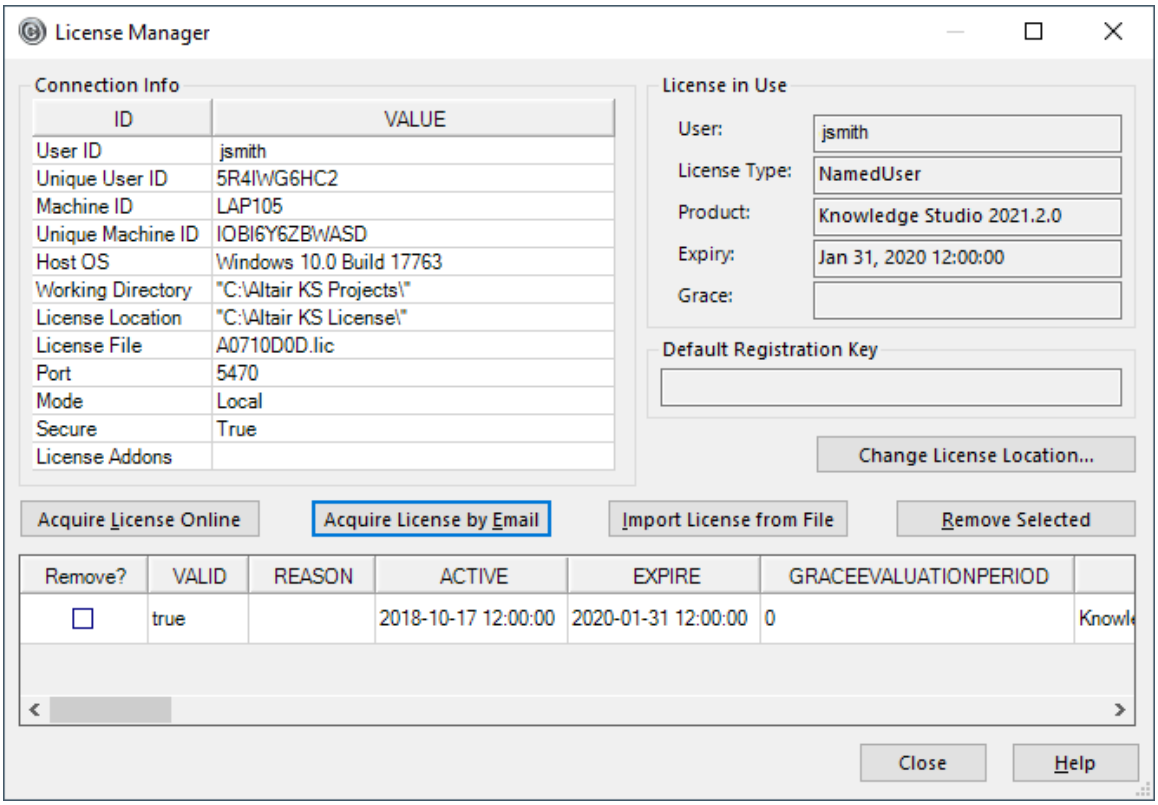
Use the function **Acquire License by Email** in the License Manager if there is no Internet access on your machine. This will generate an email message to be sent to Altair Customer Support with the information required to generate a license.

> **Note:** *If you are an administrator acquiring a named-user license for another user, make sure that you log in to Windows with the intended user's ID.*

If you are using the application in the client/server configuration, connect to KS Server first: Start KS Workstation, select the **Connect** command from the **File** menu, choose the **Server Login** mode, and specify the connection properties.

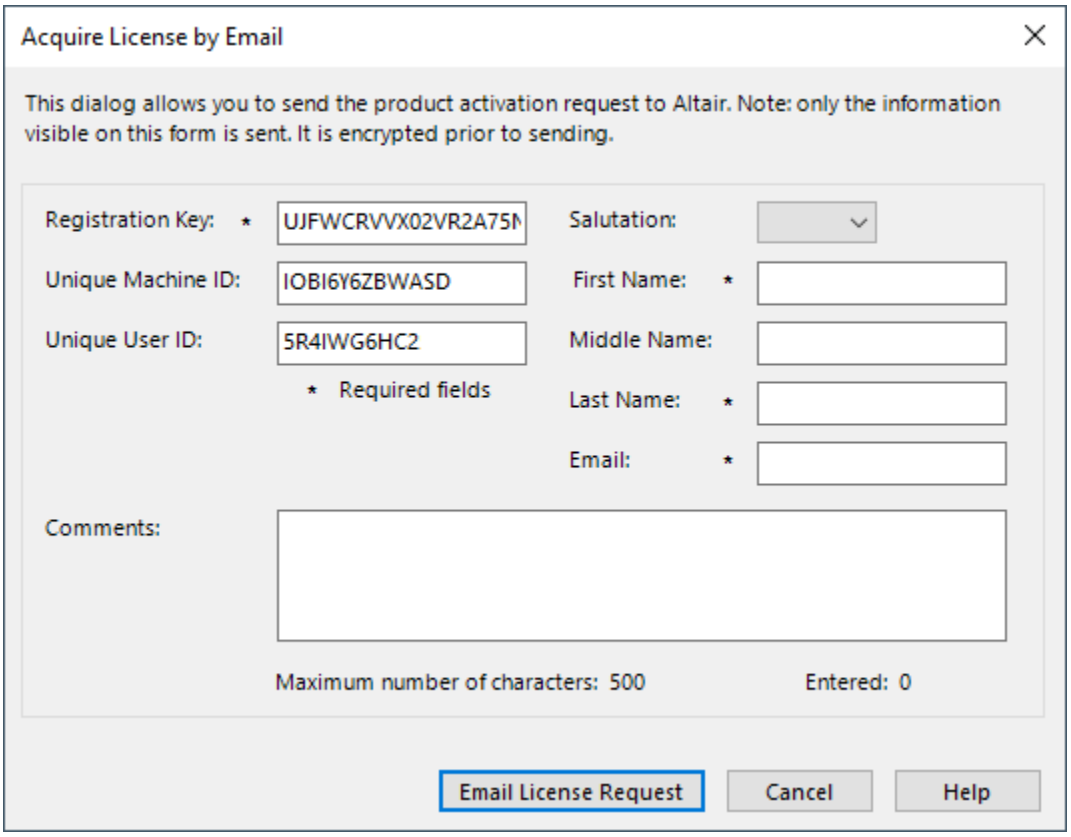
The License Manager window opens when you start the application. It is also available under the **Help** menu. Click the button **Acquire License by Email** in the License Manager.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The Acquire License by Email dialog opens. The Unique Machine ID and Unique User ID fields are populated automatically, and these values should not be changed.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



1. Enter your first name, last name, and email address.

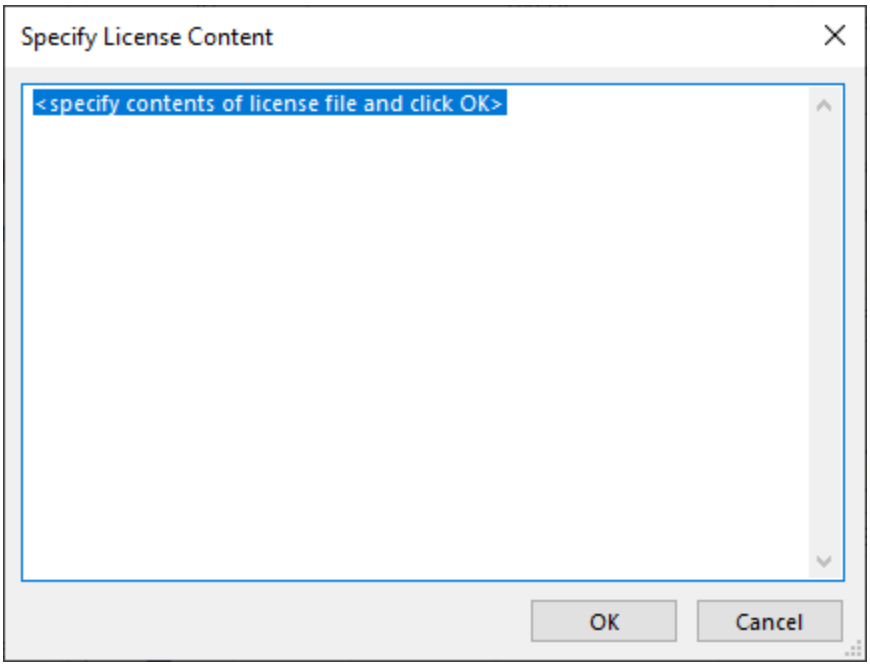
2. Enter any comments if necessary.

3. Click the **Email License Request** button. The license request will be sent to Altair Customer Support at productactivation@datawatch.com.

4. Once you receive an email from Altair Customer Support in response to your request, open the attached license file in Notepad or any other text editor and copy its contents to the clipboard.

9. Run KS Workstation and open License Manager. Click the button **Import License from File** in the License Manager window. Paste the license content into the Specify License Content dialog. Do not change the license text. Cllck OK.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The installed license will be shown in the lower frame of the License Manager. Close the License Manager.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Connecting to the Data Mining Engine

Connecting to the Data Mining Engine

The Data Mining Engine is the core component of Knowledge Seeker and Knowledge Studio that performs data mining and predictive analysis operations such as calculating summary statistics, creating and scoring a predictive model, etc.

In the Standalone configuration, both the Data Mining Engine and the user interface component reside on the same computer. In the Client/Server configuration, the Data Mining Engine resides on a server. In all configurations, when you launch KS Workstation, it automatically connects to the Data Mining Engine, checks the license, and opens the Working Directory with your projects in the Project Explorer (the left panel). The name of the Working Directory is displayed at the top of the Project Explorer panel.

In the standalone mode, you normally would not need to connect to the Data Mining Engine manually unless the Working Directory specified in the current settings is not accessible or does not exist.

Establishing a connection using the File | Connect menu command is necessary when:

- the Project Explorer panel is blank, and the commands **Create Project** and **Set Working Directory** under the File menu are disabled, which may happen in the following cases:
  - the default Working Directory is invalid (does not exist or is not accessible)
  - the default client/server connection information was not specified at the installation stage
  - after a previous unsuccessful attempt to connect to the server (for example, if incorrect user credentials or host information was used)
  - after the connection was terminated abnormally or reset as a result of server maintenance
- connecting to KS Server on a different server
- changing from the standalone to the client/server mode or vice versa

To connect to the Data Mining Engine, select the menu command

File | Connect

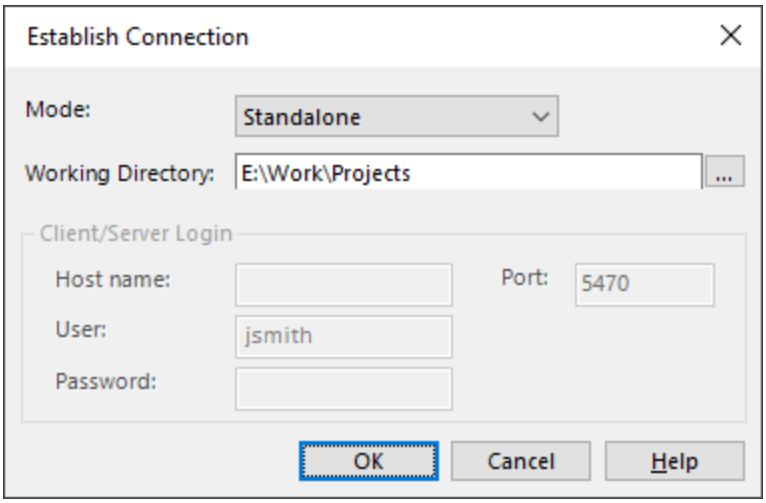
or click the leftmost icon on the application toolbar. This brings up the Establish Connection dialog box.

Standalone Mode

When working in standalone mode, all processing for the application occurs on the user's workstation and the data files are stored either on a local hard drive or on a network file system. The default working directory is the Altair folder under the My Documents folder, but it can be set to any other location at the installation stage. If the working directory is accessible, the

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

connection is established automatically upon launching the application. Otherwise, select **File | Connect** from the menu and specify the correct name for the Working Directory:



Mode	Standalone
Working Directory	The Working Directory is the location where your projects and the related documents are stored. It is sometimes referred to as the Mining Location.

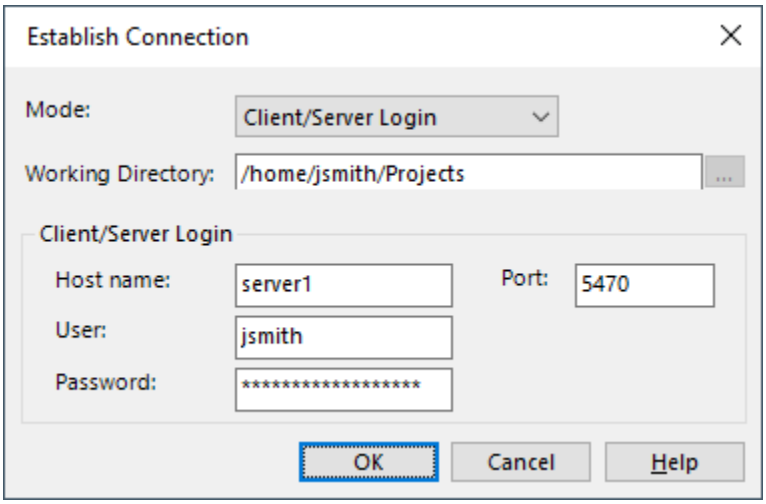
If already connected, you can change the current Working Directory at any time using the menu command **File | Set Working Directory** or by clicking the second icon from the left on the application toolbar. See the Setting the Working Directory topic for details.

Client/Server Login Mode

In the client/server mode, all processing for the application occurs on the server machine that hosts the KS Server with the Data Mining Engine. The user's workstation only contains the client component with graphical user interface to connect to the server and visualize the results. The connection properties can be set during the installation process. If all the properties are set during the installation, the connection is established automatically as you launch the client component and specify your password for the server.

If you need to establish the connection in one of the cases indicated above, you must select the command **File | Connect** and enter the connection information for the server that hosts the KS Server. For example:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Mode	Client/Server Login
Working Directory	The Working Directory is the location where project related documents are stored. It is sometimes referred to as the mining location.
Host name	The name or IP address of the machine hosting the KS Server.
Port	The number of the port to access the KS Server.
User	User ID recognized by the KS Server host machine.
Password	User password.

If already connected, you can change the current Working Directory at any time using the menu command **File | Set Working Directory**, by clicking the second icon from the left on the application toolbar, or pressing CTRL W on your keyboard.

See also:

- Setting the Working Directory
- About Projects
- Creating a New Project

Setting the Working Directory

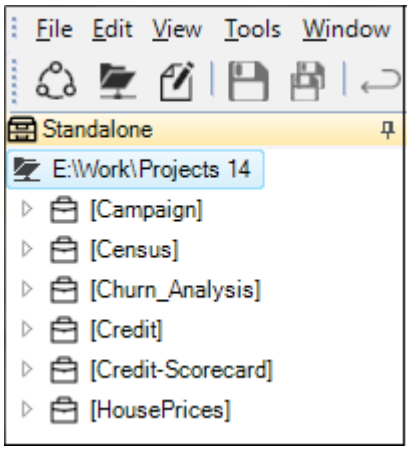
The Working Directory is the location where your projects are stored. You can have more than one Working Directory if you have more than one set of projects. You can change your Working Directory at any time.

The Working Directory is usually the Altair folder under My Documents, which is the installation default for the Standalone mode.

In the Standalone configuration, the Working Directory is usually a local folder on the user's workstation, but it can also be on a network drive. In the Client/Server configuration, it is usually on the server machine that hosts KS Server.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

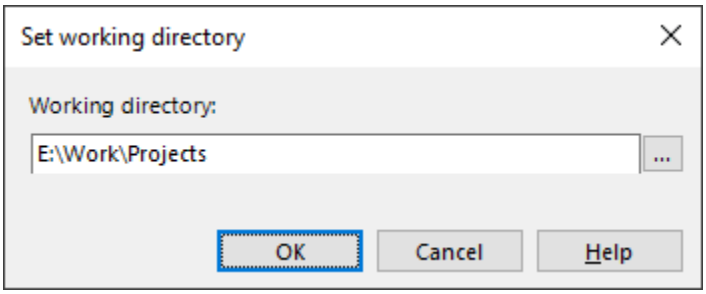
The Working Directory name is displayed at the top level of the Project Explorer panel. See the topic About Projects for details.



To change the Working Directory:

From the menu:

File | Set Working Directory

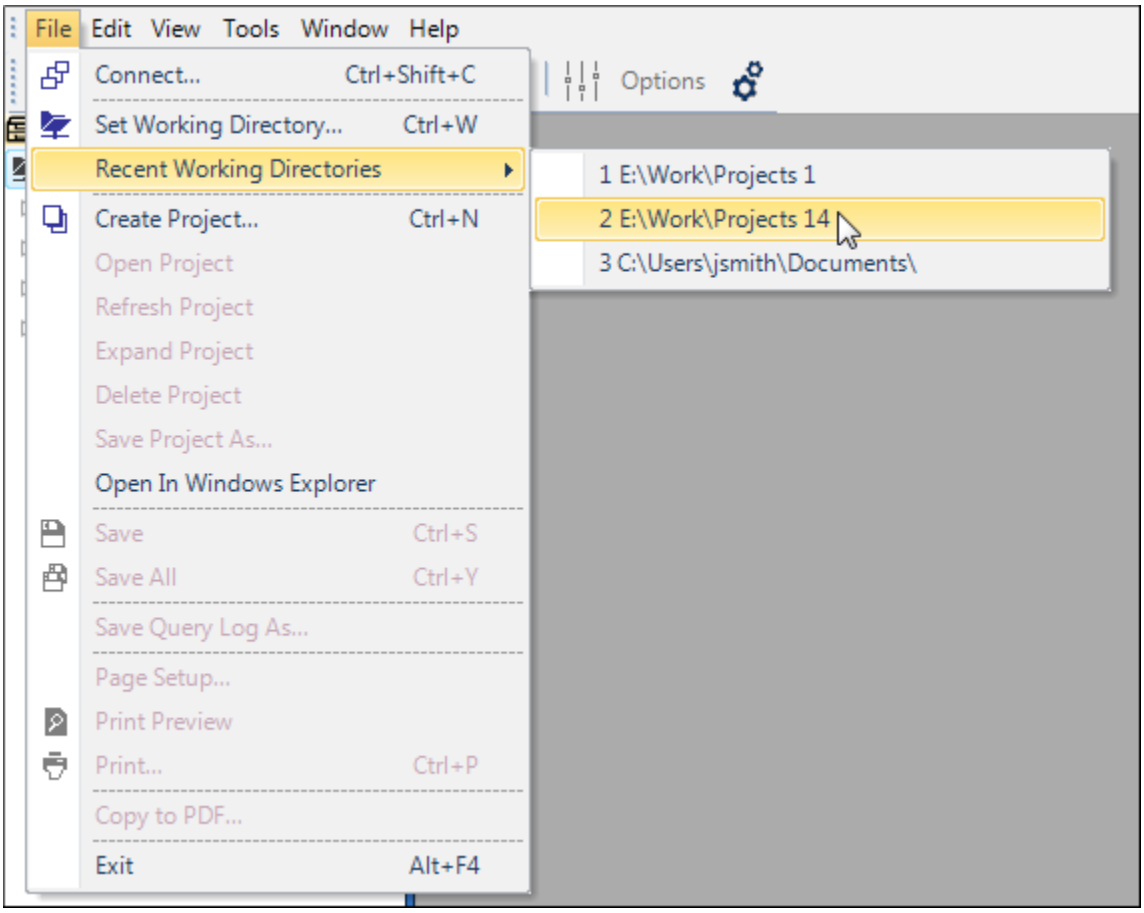


Type the name of the new Working Directory or use the browse button to find the desired directory.

You can open the Working Directory in Windows Explorer by right-clicking on its name in Project Outline and selecting *Open in Explorer* from the context menu. This may be useful if you wish to view error logs written to the working directory, back up your projects, share them with other users, and so on.

A list of recently used Working Directories is available from the File menu for easy switching between working directories.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also:

[About Projects]

[Creating a New Project]

[Connecting to the Data Mining Engine]

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Projects and Workflows

Projects

A project contains workflows and the objects produced by executing operations in these workflows, i.e., datasets, models, reports, etc. Each project is represented by a folder in the file system.

The Project Explorer (the left panel) shows all projects in the current Working Directory, which is set using the **File | Set Working Directory** command. You can view and work with multiple projects in the Project Explorer at the same time.

Each new project you create by selecting **File | New Project** from the menu is represented by a physical folder in your current Working Directory. Individual project components (models, model analyzers, etc.) are stored as files in the project folder.

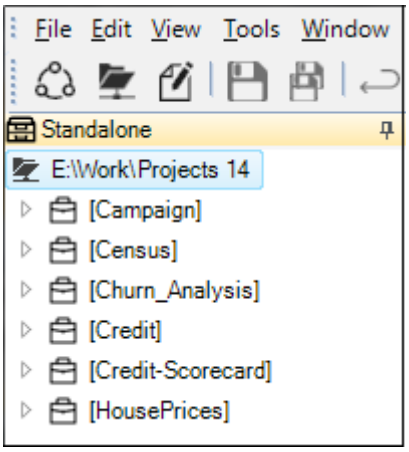
Conversely, every folder inside the current Working Directory is considered a project. In fact, you can copy files to a project folder and they will appear in the Project Explorer after the project is refreshed.

Any project can be read by the KS Data Mining Engine of the same version on any supported operating system.

Project Explorer

The Project Explorer panel on the left displays the list of all projects in your Working Directory. Expand a project by clicking on the triangle on the left-hand side to view its contents.

The Working Directory name is displayed at the top level of the Project Explorer panel. All projects contained in the Working Directory are displayed beneath its name. In this example, there are seven projects in the Working Directory.



The hierarchy of objects within a project is displayed in a typical Windows Explorer-like fashion. At the top of the hierarchy is the project. Attached to the project are dataset objects, models, reports, generated code, model analyzers, and possibly some desktop documents. Decision trees and

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

predictive models are normally child objects of datasets. Some types of objects are always automatically placed at the highest level (as a child object of the project itself), for example, Model Analyzer.

To open any object, such as a dataset, double-click on it in the Project Explorer or right-click on it and select **Open** from the context menu. Alternatively, select the item and press Enter. This will display the default view of the open object (for example, the Overview Report of a dataset).

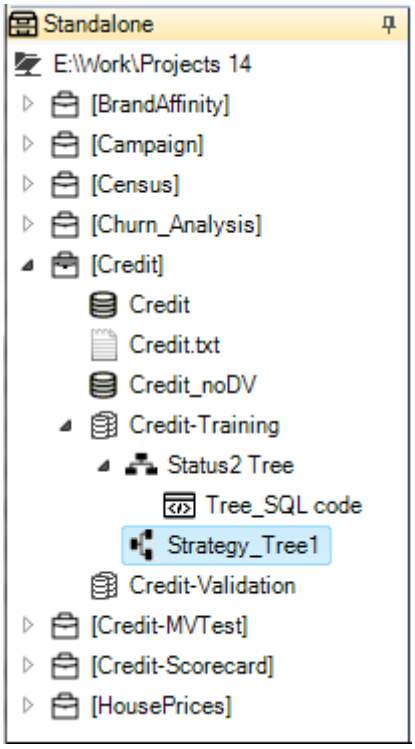
For easy navigation between the currently open objects in your project as well as between currently open projects, make sure you use the Workbook Tabs mode (menu: **Tools | Preferences**, go to the **General** tab and select Workbook Tabs). In this mode, the set of tabs at the top of the workspace window represents objects and projects that are currently open.

Use the view tabs along the bottom of the object window to navigate through its views.

Normally, Project Explorer appears on the left side in a *docked* position. You can choose to auto-hide the Project Explorer window or make it floating and move it anywhere within the application window. To this end, just right-click its title bar and select **Floating** or **Auto-Hide**, or click its title bar and drag it to the desired location.

In the example below, the Census project has been expanded to show all the elements that belong to it. The Census project contains a ***dataset*** (Census) with two *partitions* (Census-Training and Census-Validation), a ***Decision Tree*** (Income), a *Strategy Tree* (Strategy1), *SQL code* generated for the Income Tree (SQL rules), and *Model Analyzer* for analyzing the decision tree performance.

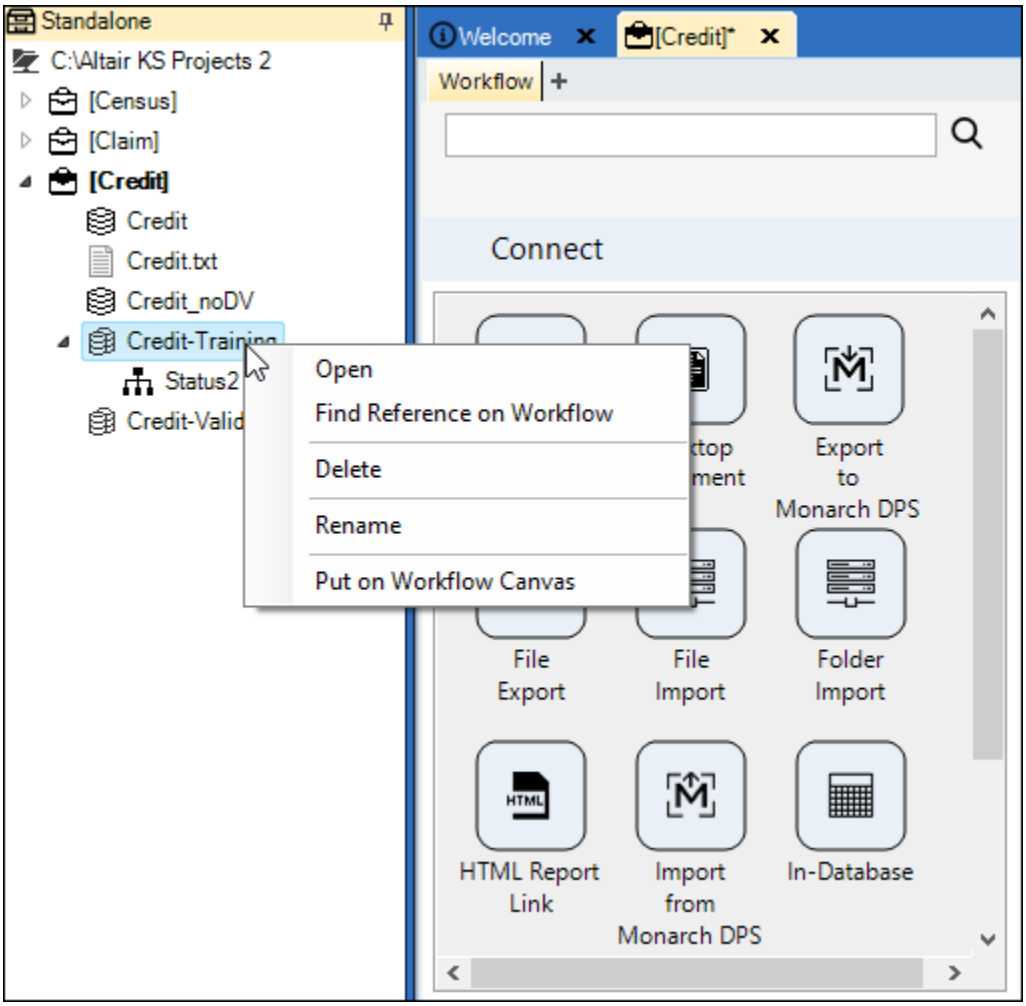
The title bar of the Project Explorer indicates whether you are working with data mining engine installed on the local machine (Standalone) or a remote server (Client/Server).



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

You can rename any project or object within the project tree. The rename operation will be tracked in the activity log so you can see the history by selecting **View | Activity Log** from the menu. To rename an object select it in the Project Outline and choose the **Rename** command from the **Edit** menu. You can also right-click on the object and select **Rename** from the context menu, or press the F2 key.

All objects within the project explorer have the right-click context menu. Common commands such as **Rename**, **Delete** and **Open in Explorer** are shown in the context menu. You can use the context menu for dataset objects to insert Measure of Predictive Power or a desktop document.



It is possible to select more than one object at a time. This is done by holding down the Control (Ctrl) or Shift keys while clicking the mouse. In this case, the list of commands in the right-click menu shows only operations applicable to multiple datasets. Operations such as **Delete** can be performed on multiple objects at a time.

See also:

- Setting the Working Directory
- Creating and Deleting Projects
- Workflows

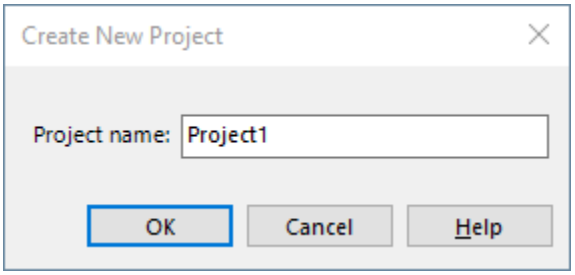
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Creating and Deleting Projects

Creating Projects

A project contains process workflows and the results of their execution, i.e., analysis objects such as datasets, models, reports, etc. The Project Explorer panel shows all projects in the current Working Directory, which is set using the **File | Set Working Directory** command. See Projects for details.

To create a new project in the current Working Directory, select the command **File | Create Project** from the main menu.



Enter the project name and click OK. A new project with this name will appear in the Project Explorer.

The project name:

- **cannot** contain the following special characters:

  [ ] " * + # & ! % ( ) { } ` ' @ $ ^ , . / : ; < = > ? \ |

- **can** contain spaces, for example, "**Credit Risk Project**".

Deleting Projects

To delete a project, right-click on the project name in the Project Explorer and select **Delete** from the context menu. Alternatively, click on it and choose **File | Delete Project** from the main menu. The project with all its contents will be deleted from the Project Explorer and from the disk.

See also:

Setting the Working Directory

Projects

Workflows

Workflows

Introduction

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

A ***workflow*** defines a data mining process in terms of operational nodes and data flows. A typical process starts with extraction and preparation of the data, goes through modeling and ends with model deployment or outputting deployment code, and optionally exporting data.

Similar to a program, a workflow is a description of interconnected operations together with their inputs and outputs which can be re-run multiple times with different inputs.

The Altair workflow automation feature allows you to easily construct such workflows in a visual, interactive way: by dragging, dropping and connecting process nodes on the visual canvas and specifying their inputs, outputs, and parameters.

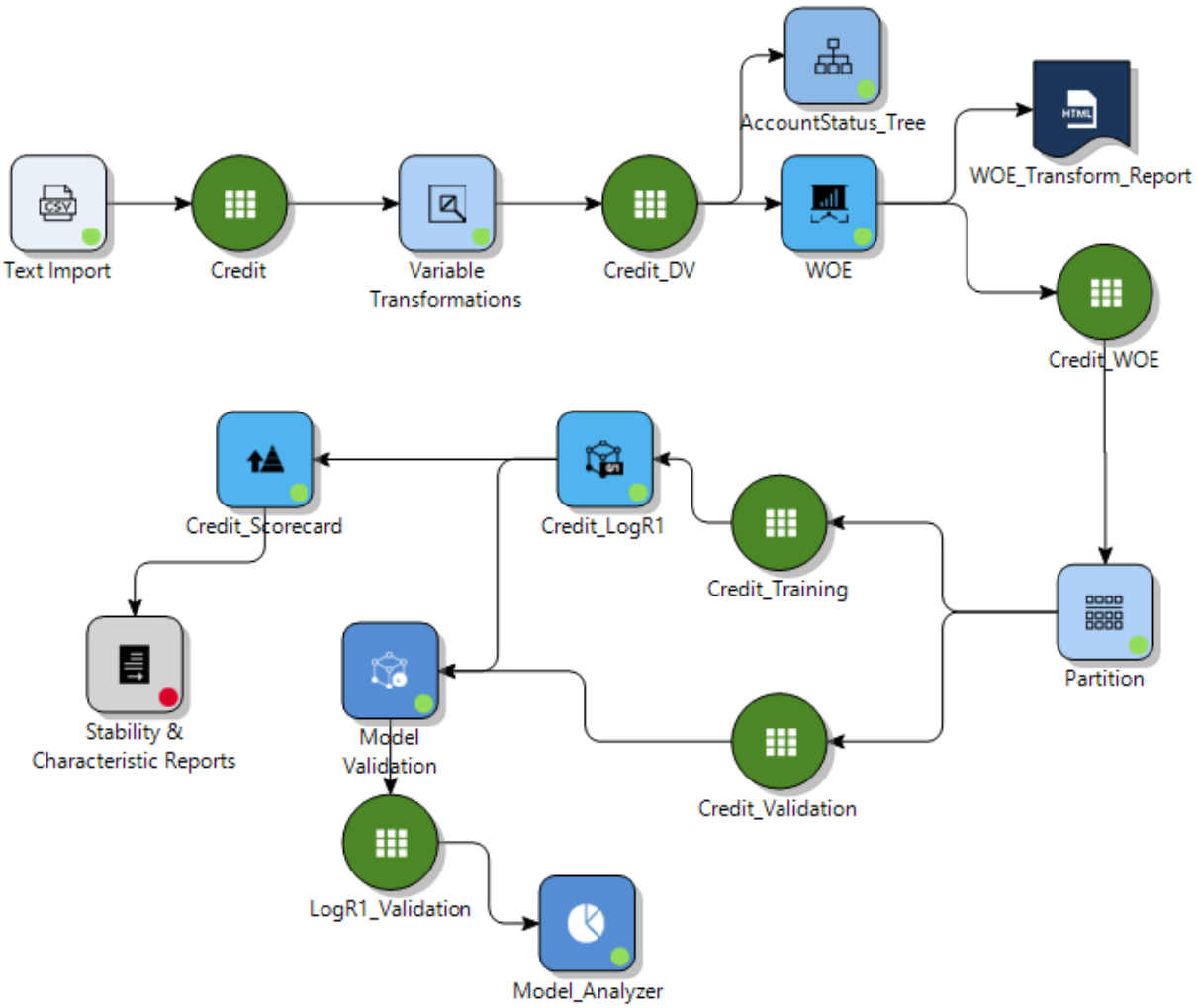
The process defined by a workflow constructed this way can easily be re-run on updated or refreshed data inputs. Particular data transformations, modeling, and other operations you defined in a workflow can be repeated as long as the structure of the new input data is the same.

Altair workflow execution features allow you to re-run individual nodes and segments of the workflow as well as the entire process.

One of the most useful deployment features is automatic SAS Language code generation for all stages of the Scorecard building workflow. Code can be generated for the whole workflow, sub-processes, or individual nodes. Then it can be copied directly into a SAS System and run there. Other types of code such as SQL Function can also be generated.

Here is an example of a workflow that builds a decision tree, a logistic regression, a scorecard, scores it and analyzes its performance:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



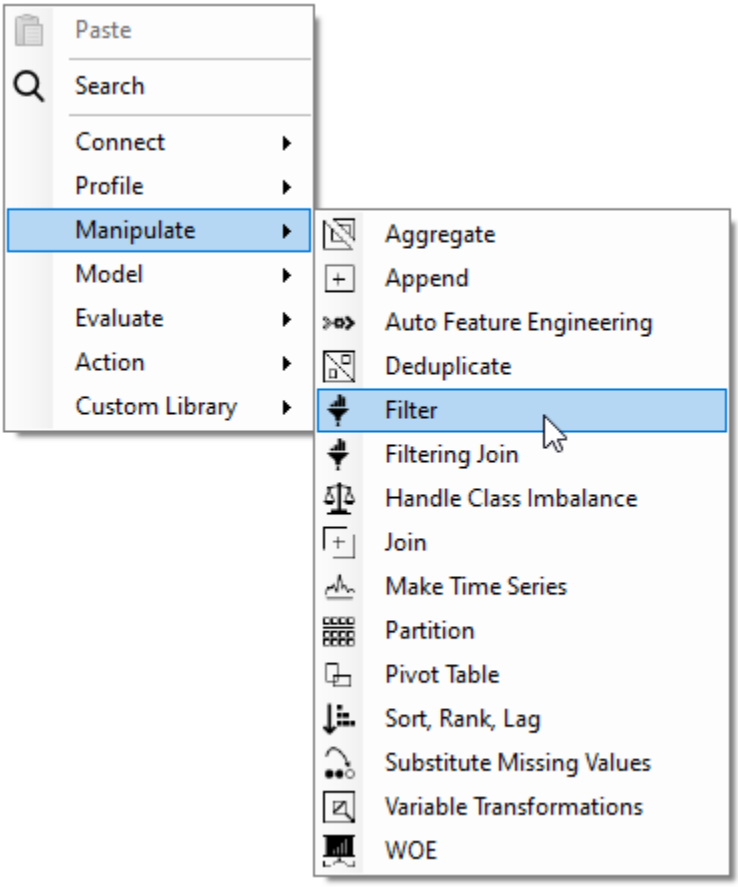
Adding Nodes to Workflow

There are several ways to add nodes to the workflow canvas:

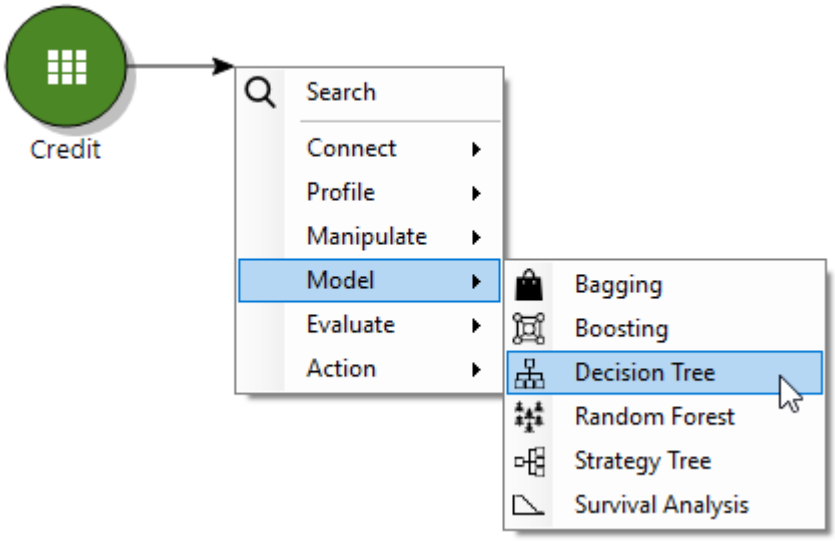
1. Drag-and-drop: Drag the desired node from a functional palette (Connect, Profile, Manipulate, Model, Evaluate, or Action) to the workflow canvas.

2. Right-click on the canvas and select the desired node name from the hierarchical context menu.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



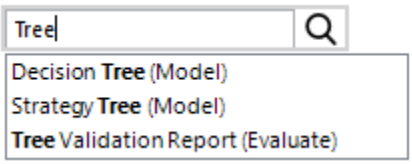
3. Drag a connection arrow from the input node to an area on the workflow canvas. Release the mouse and select the desired node category and then the node name from the context menu.



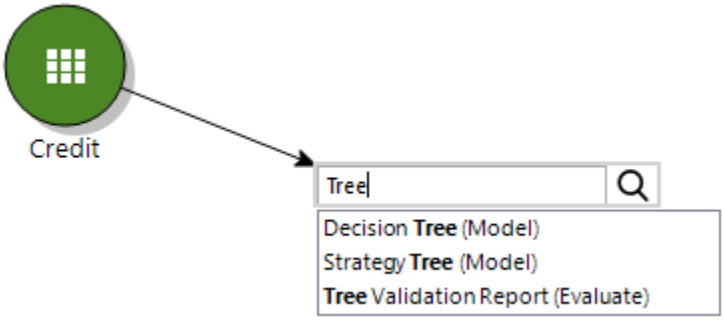
4. Using a keyboard shortcut: Click on the workflow canvas background and press **CTRL + I** on the keyboard. ("I" for "Insert"). The Search box will appear on the canvas near the mouse pointer. Start

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

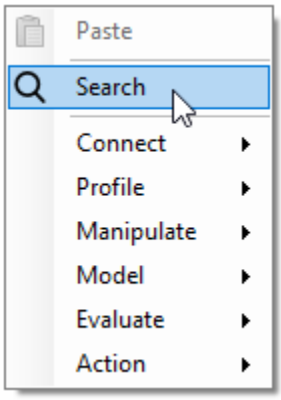
typing the name of the node you need to insert on the workflow. The auto-complete feature suggests the variants as you type. Select the desired node name from the drop-down list of suggested nodes.

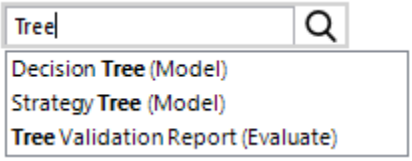


If you select an existing node on a canvas (by clicking on it) right before pressing **CTRL + I**, the search box appears with the connection arrow from the assumed input node. Once you are finished specifying the name of the desired node in the search box, the new node will be added, already connected to the input node.



5. Right-click on the workflow canvas background and select the **Search** command from the context menu. The Search box will appear. Start typing the name of the node you wish to insert on the workflow. The auto-complete feature suggests the variants as you type. Select the desired node name from the drop-down list of suggested nodes.



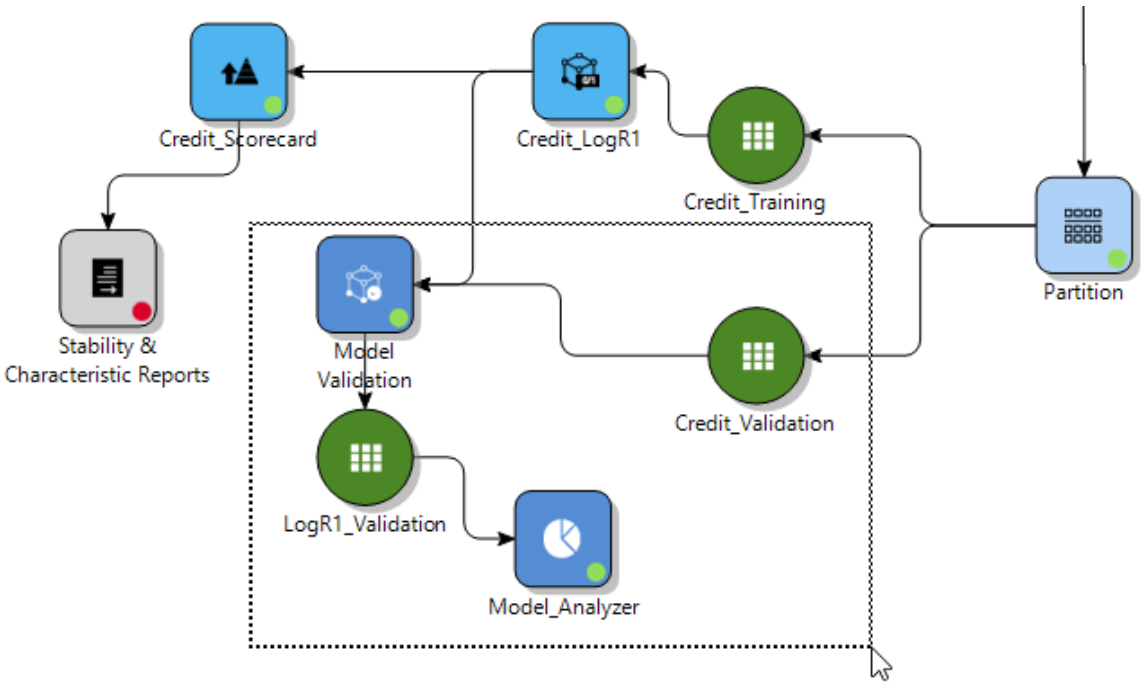


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Manipulating Nodes and Groups of Nodes

Nodes can be moved around the workflow canvas simply by dragging and dropping.

- To move a single node, click inside the node (closer to the centre rather than the edge) and drag it to the desired location.
- To move a group of nodes, select the desired group first by dragging a rectangle over the area enclosing these nodes as shown in the example below.

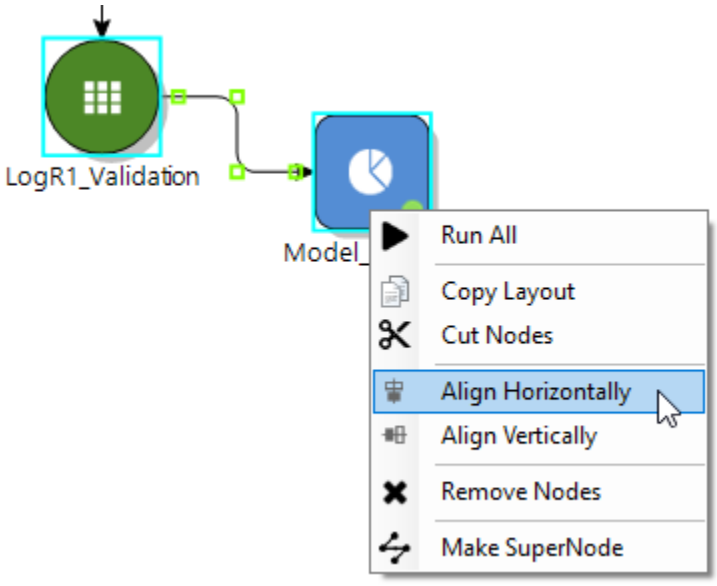


.

The nodes in the selected group will become highlighted with a light-gray frame around each node. Then click inside any node in the group (closer to the centre rather than the edge) and drag it to the desired location. This will move the whole group, provided that it is still highlighted.

- To align a group of nodes, select the desired group by dragging a rectangle over the area enclosing these nodes as shown above. The group becomes highlighted. Right-click on any node in the group and select "Align Horizontally" or "Align Vertically" from the context menu, as shown below.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Supernodes

Supernodes represent a convenient way to organize workflows.

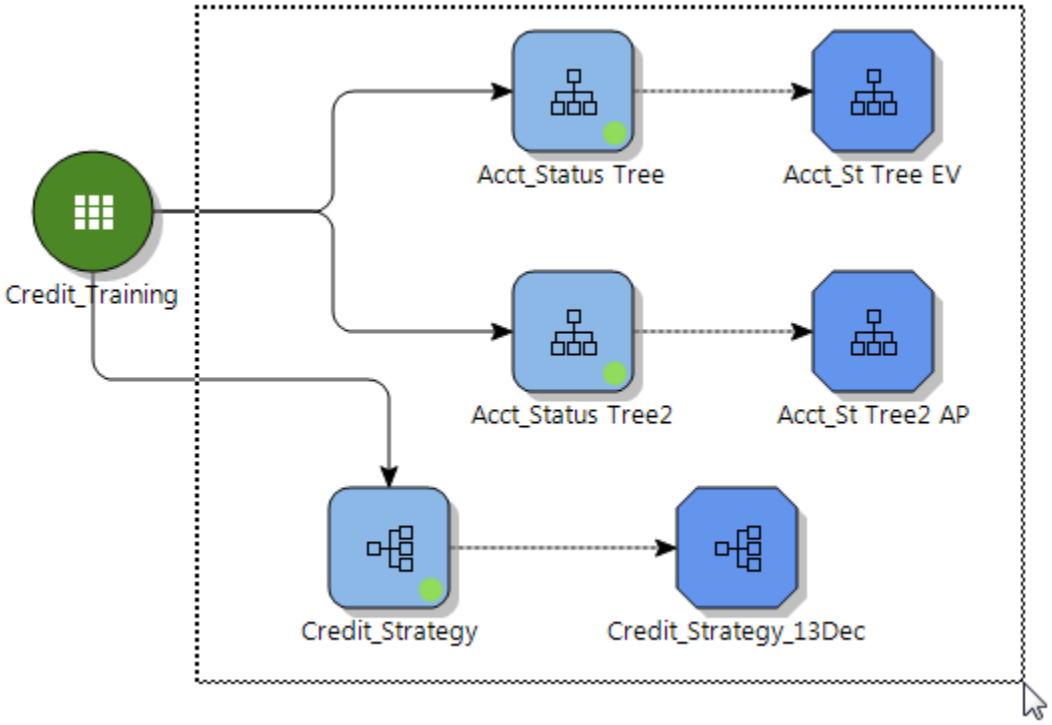
When building or viewing large and complex workflows with a lot of nodes and connections, you can merge any group of nodes into a single "**supernode**". A supernode is a cluster of nodes merged into a single node. This way the whole sections of a workflow can be contained inside a single node for operational and viewing convenience. The functional schema of the workflow is preserved as the supernode's input and output connections are the same as those of its members. A supernode can be ungrouped at any time to restore the original representation of the corresponding group of nodes on the canvas.

To create a supernode, select the desired group of nodes in any of the following ways:

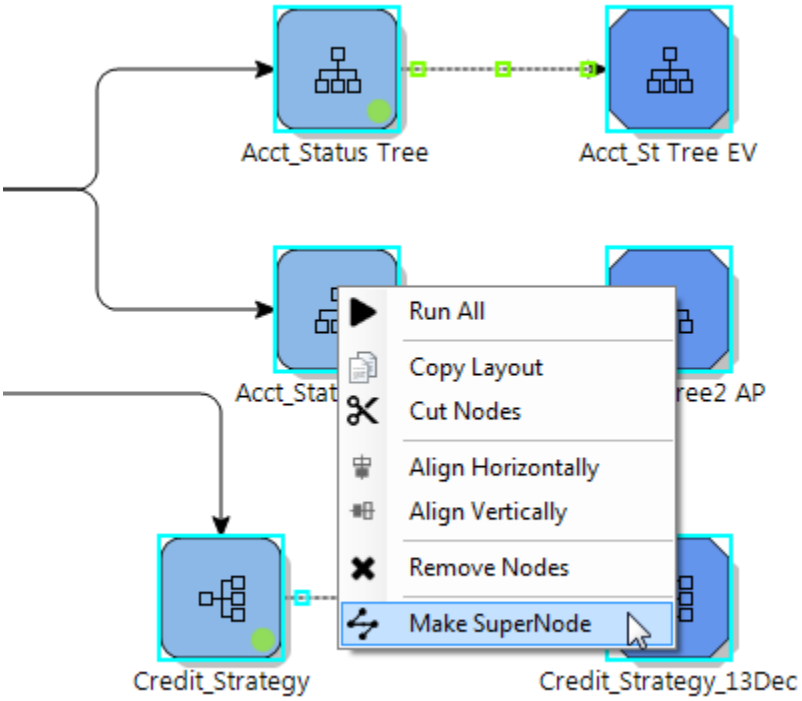
- by dragging a rectangle over the area enclosing these nodes as shown in the example fragment below; or
- by clicking on the desired nodes while holding down the CTRL or SHIFT key.

The nodes in the selected group will become highlighted with a light-gray frame around each node. You can add / exclude nodes to / from the current selection by clicking on the (un)desired nodes while holding down the CTRL key.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

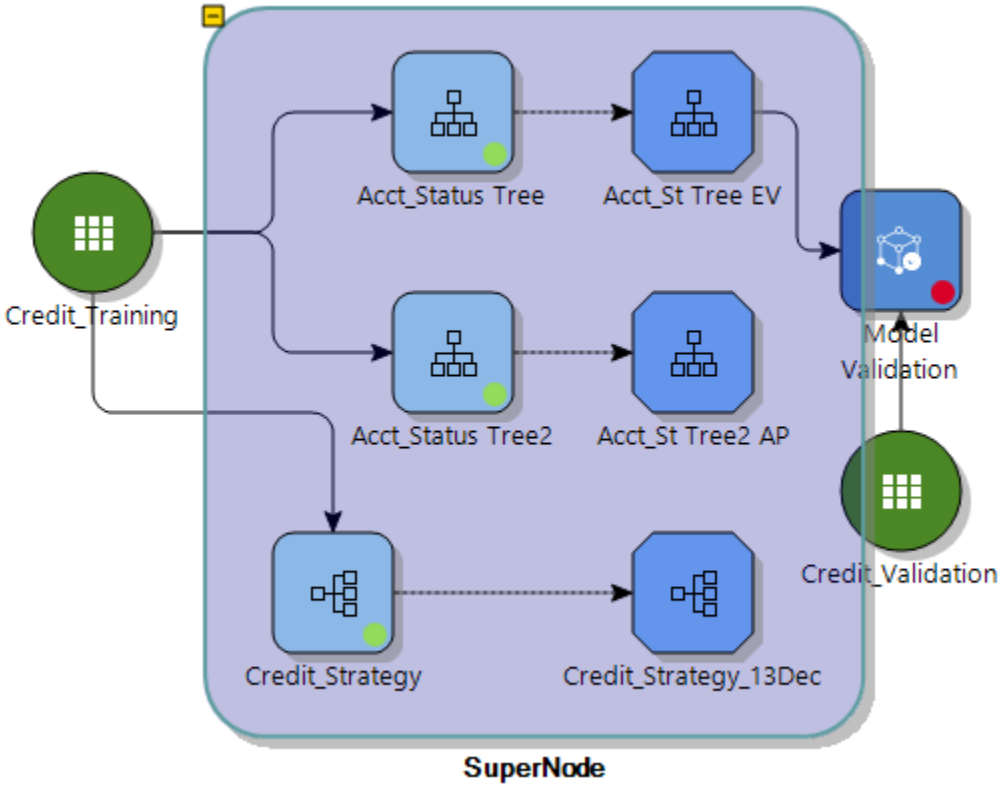


Right-click on any node in the highlighted group and select **Make SuperNode** from the context menu, as shown below:



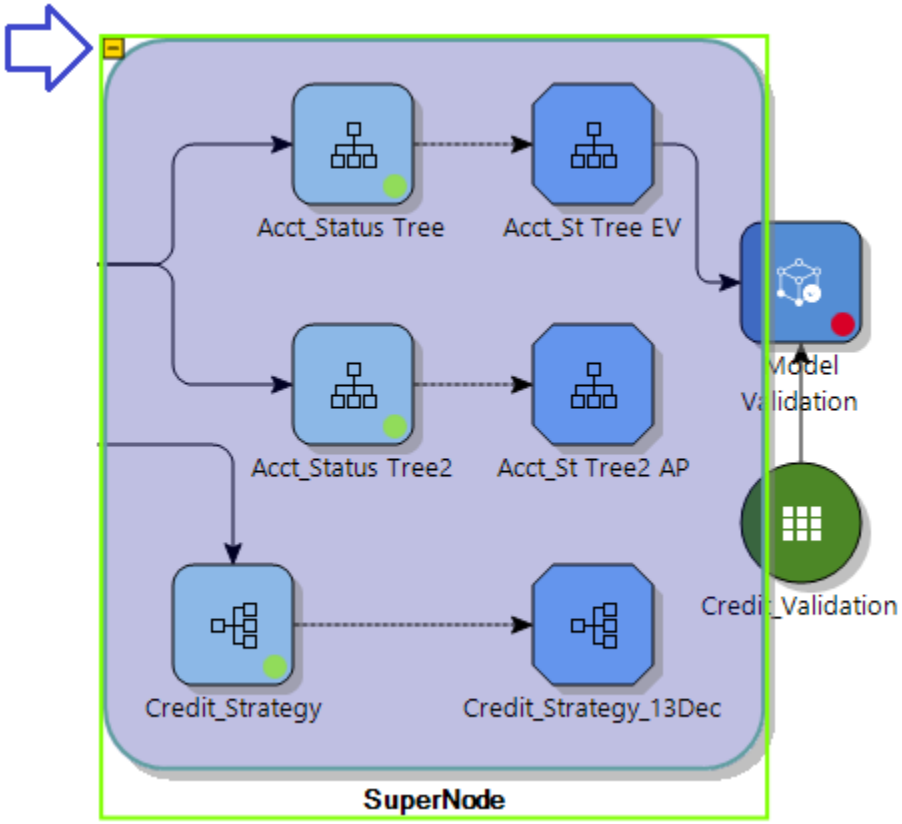
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

A light-blue frame enclosing the selected group of nodes indicates the supernode. Any nodes that were not in the original selection are shown as partially shaded. In the example below, only the 6 tree nodes comprise the supernode.

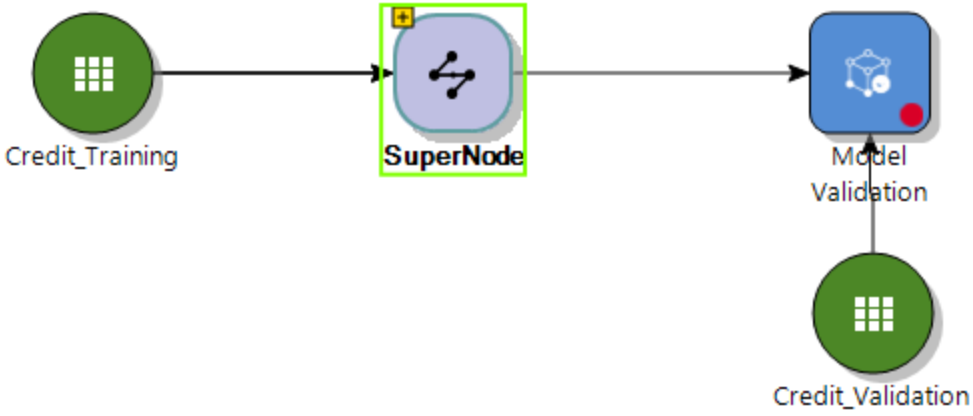


To merge (collapse) the group into a single node, click on the "minimize" control at the upper-left corner of the supernode frame.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



As a result, the highlighted group is merged into a single node labelled "SuperNode".

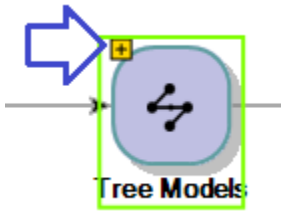


To rename a supernode, click on its label and edit the text as necessary.

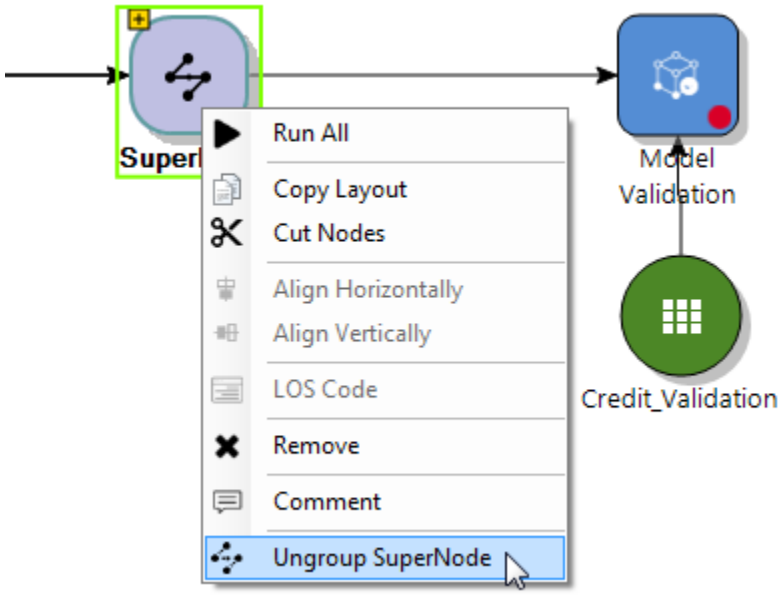


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To expand a supernode, click on the plus sign (the "expand" control) at the upper-left corner of the supernode. The supernode will be expanded (but not ungrouped), showing all nodes within it.



To ungroup a supernode, right-click on it and select **Ungroup SuperNode** from the context menu. This can be done in any state of the supernode (compact or expanded).



Other operations with Supernodes:

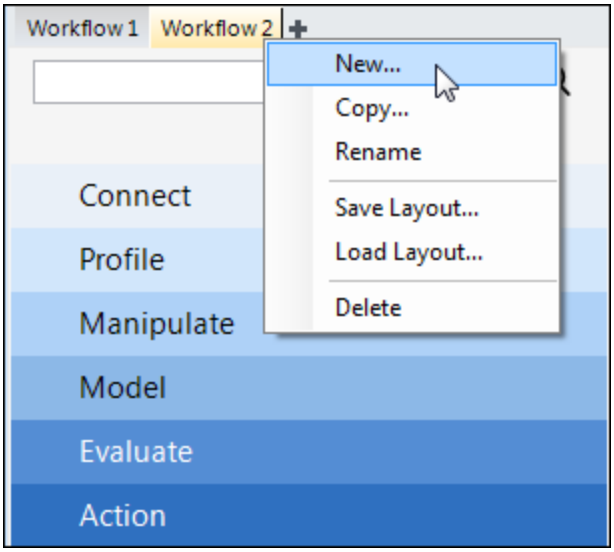
- To add any node on the canvas to a Supernode click on it to select, then press the SHIFT key and drag the node to the Supernode while holding down the SHIFT key. You can also drag a node from a functional palette directly to a Supernode in the same way.
- To remove a node from a Supernode, expand the Supernode if it is in the collapsed form, click on the node you wish to remove, then press the SHIFT key and drag the node out of the Supernode. The same can be done with multiple nodes at the same time.
- To copy or move a Supernode to another workflow, right-click on it and select "**Copy Layout**" or "**Cut Nodes**" from the context menu. Then switch to another Workflow tab and paste the supernode.
- Moving (dragging) nodes within a Supernode towards its borders will resize the Supernode.
- You can add a comment to a Supernode the same way as to any normal node (right-click and select the **Comment** command from the context menu).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- To run a Supernode right-click on it and select **Run All**. This will run all nodes within the Supernode.
- It is possible to create nested Supernodes (put one supernode inside another).

Managing Workflows

You can create more than one process in a single workflow canvas, and also have multiple workflow canvasses in a single project. If one canvas gets too crowded, create a new one by either clicking the **plus** sign or by right-clicking on the Workflow tab at the top of the Functional Palette panel and selecting **New** from the context menu. Switch between multiple workflows by clicking on their tabs. To change the order of the Workflow tabs, drag one of the tabs to the desired position relative to other tabs. You can also use the right-click context menu of the Workflow tabs to rename or delete workflows. The same commands are available under the Workflow menu.



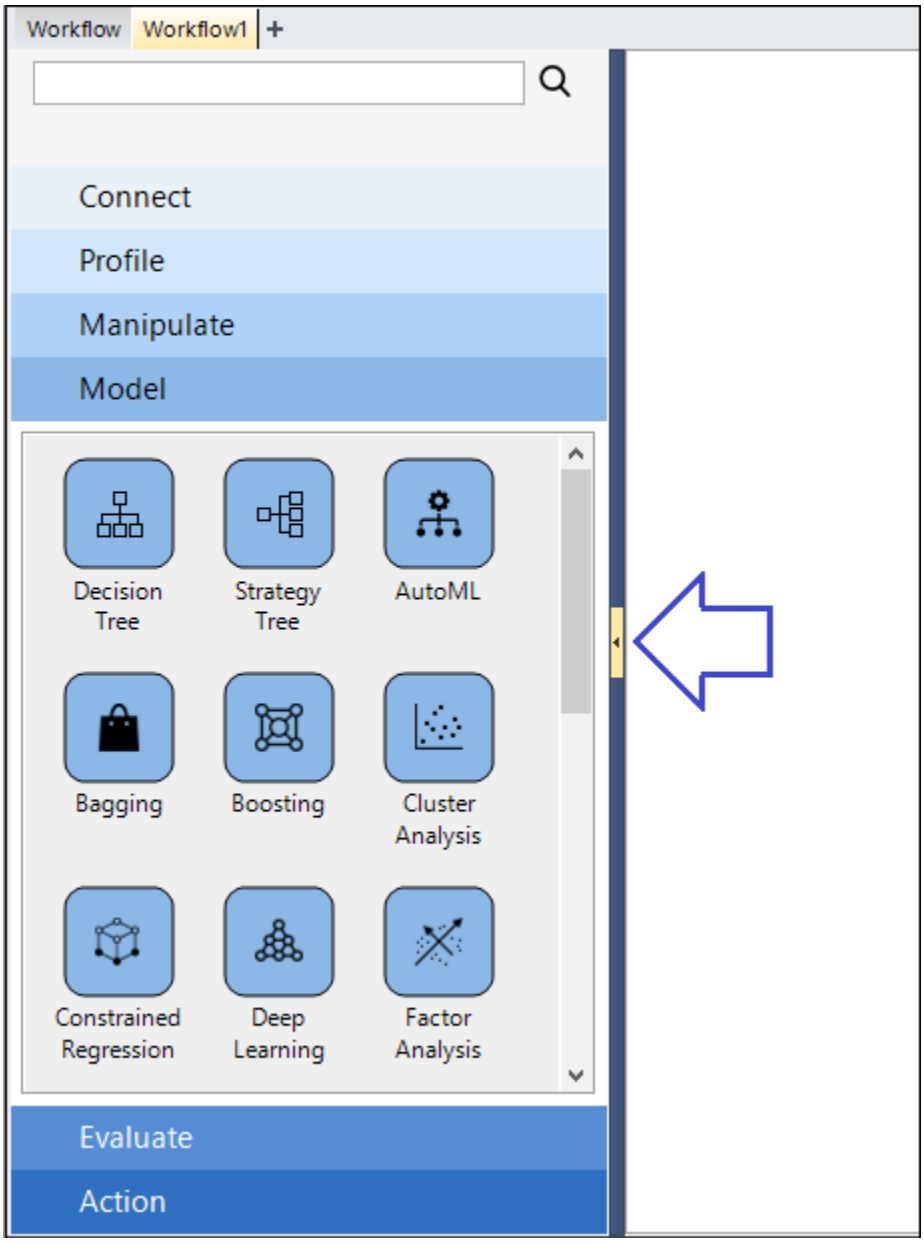
While creating and running workflows you can also easily keep track of the objects produced by running them. The project folders and objects created inside them are displayed in the **Project Explorer** panel on the left-hand side. Aside from showing all projects in your Working Directory, Project Explorer allows you to view outputs and delete the outputs that are no longer necessary.

View Options

Docking Panels

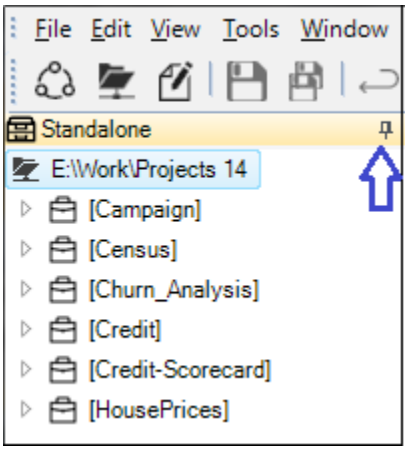
If you need to build or view large and complex processes in your Workflow canvas, you can save some space by docking the Functional Palette panel to the left. Just click on the arrow in the middle of the separator between the Functional Palette panel and the canvas:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



You can also auto-hide the Project Explorer panel. Just click on the pin at the top right corner of the Project Explorer panel. To switch back from the auto-hide mode, open it and click on the pin again.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Switching to Workflow from Object Views

When a specific object in a workflow is open (a tree, a dataset, etc.), you can easily switch from any object view back to the workflow in any of the following ways:

- using the toolbar button  ("*Switch to Workflow*")
- clicking on the Workflow tab representing the current project at the top of the object view
- double-clicking on the project name in the Project Explorer panel
- using the keyboard shortcut CTRL TAB (if the workflow was open immediately before the current object view)

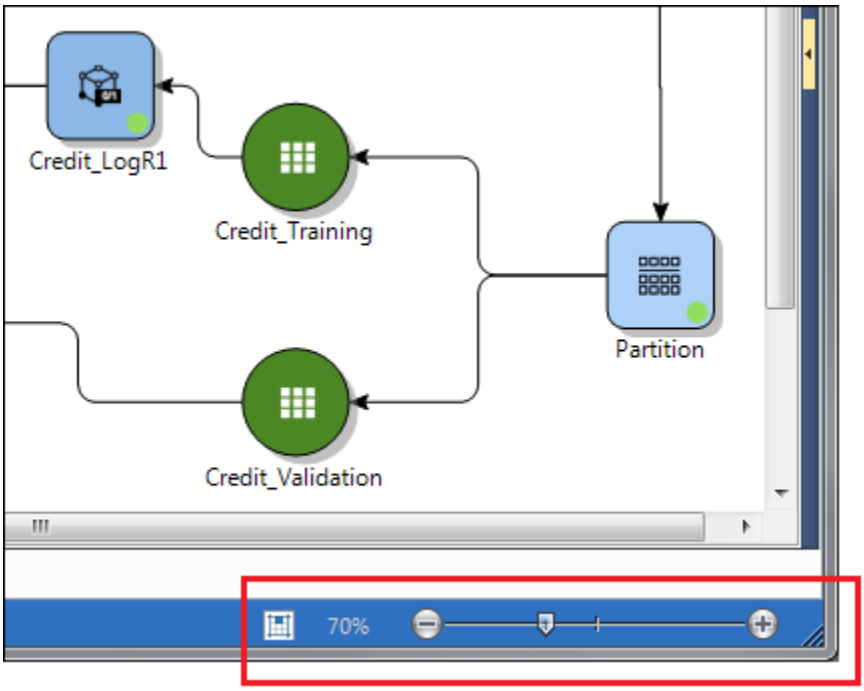
Pan and Zoom

Zoom:

You can easily navigate across a large workflow using pan and zoom features when the workflow is larger than the application window. Zoom in or out using the standard zoom control with a slider provided in the Status bar at the bottom right corner, as shown in the screenshot below. Zooming can be performed in several ways:

- moving the slider or clicking the +/- icons on the zoom control;
- scrolling the mouse wheel while pressing down the CTRL key;
- using the keyboard shortcuts CTRL - (zoom out), CTRL + (zoom in), and CTRL 0 (restore zoom to 100%).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Pan:

The pan control allows you to drag the workflow canvas around with your mouse instead of using the scroll bars. While holding down the CTRL key on your keyboard, click anywhere in the workflow canvas, hold the left mouse button down and start dragging to move the canvas.

Mini-map

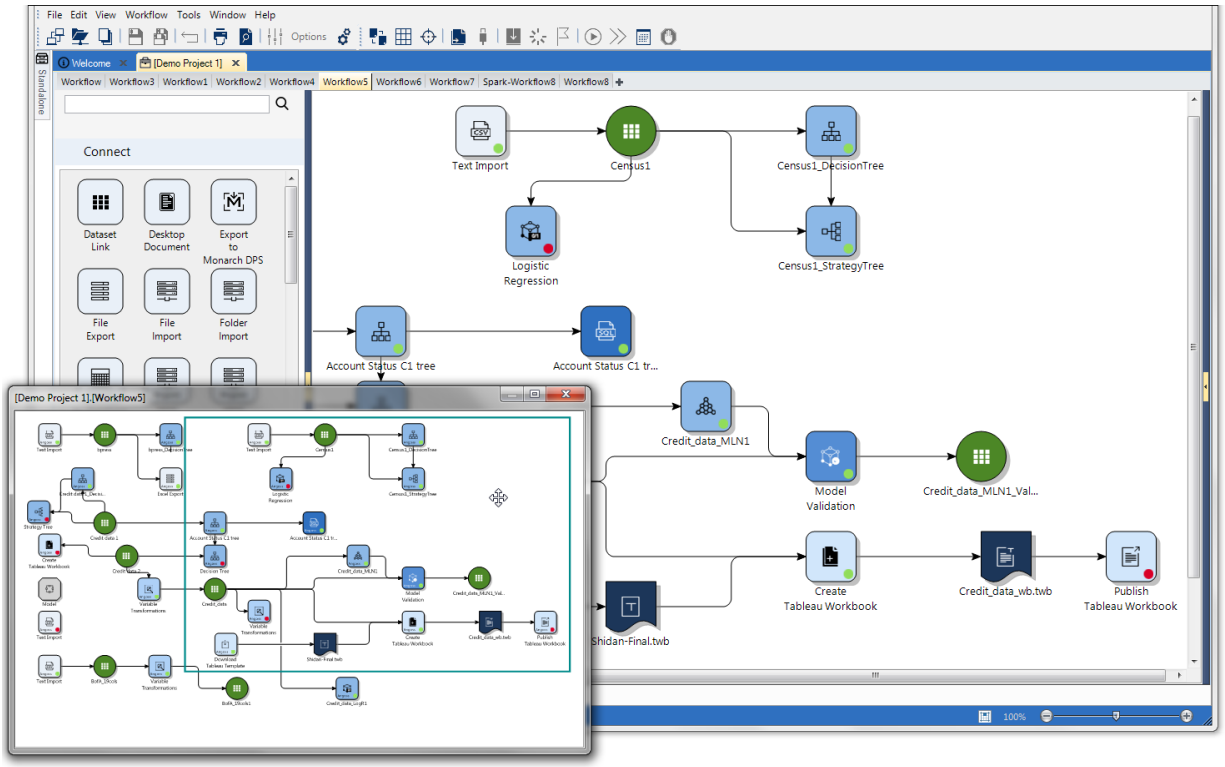
An interactive mini-map of the workflow can be opened by clicking the mini-map button in the toolbar  . It shows the current view of the workflow relative to the whole workflow and allows to easily navigate across large workflows that don't normally fit in a single view, as well as zoom in and out.

A transparent rectangular frame inside the mini-map serves as a movable lens or magnifying glass and indicates the part of the workflow shown in the current view. The fastest way to navigate to the desired part of the workflow is to click on that part in the mini-map. The frame will move and the Workflow view will be updated immediately to show the part you clicked on. Alternatively, click anywhere within the frame and drag it around the mini-map frame, dropping at the desired position. The view of the workflow will be updated dynamically depending on the position of the frame.

You can resize the mini-map, and you can also resize the "magnifying" frame by dragging on any of its corners. Changing the frame size will cause the workflow to zoom in or out.

Each project can have only one mini-map. If you switch to another workflow within the same project, the mini-map will change accordingly to show the current workflow.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

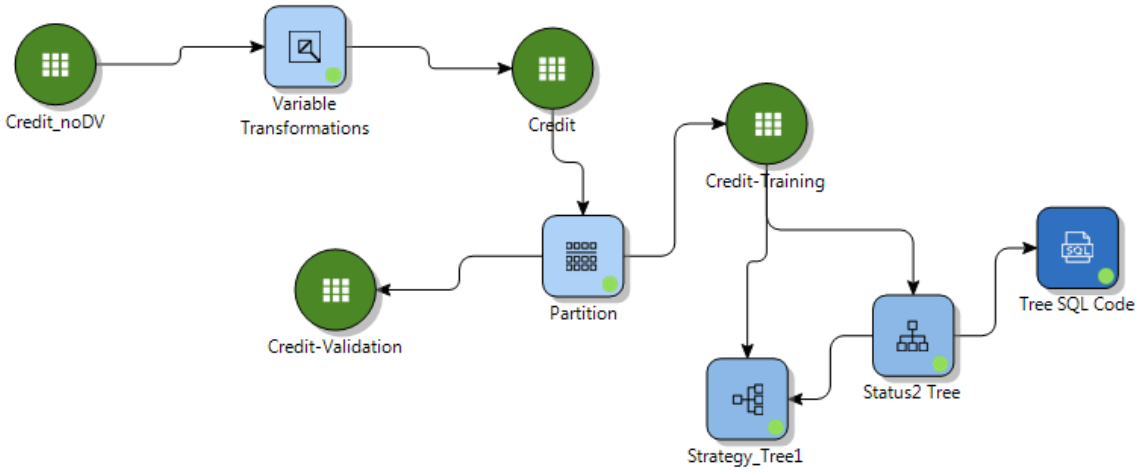


Auto-arrange All Nodes in a Workflow

You can auto-arrange nodes in your workflow to align all connections in the horizontal direction by clicking the Auto-Arrange button  in the toolbar. To undo the change, click the button again.

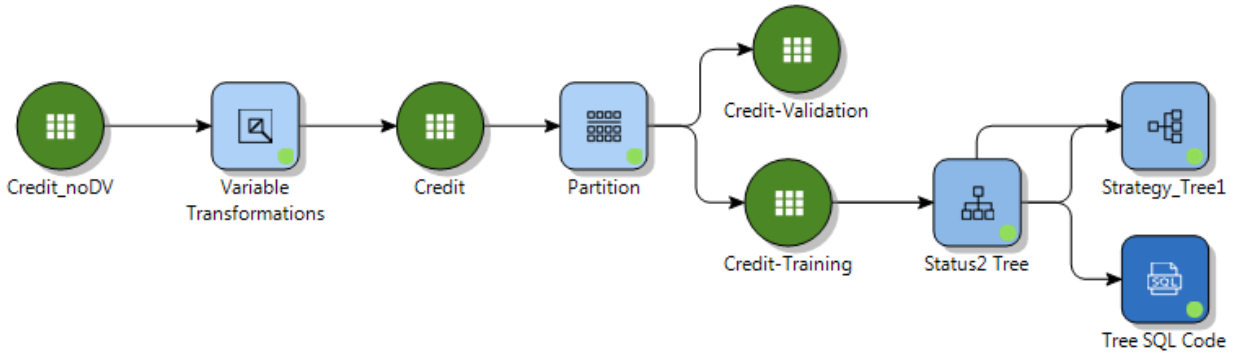
Here is an example of a workflow before and after auto-arranging:

1. Before Auto-Arrange:



2. After Auto-Arrange:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

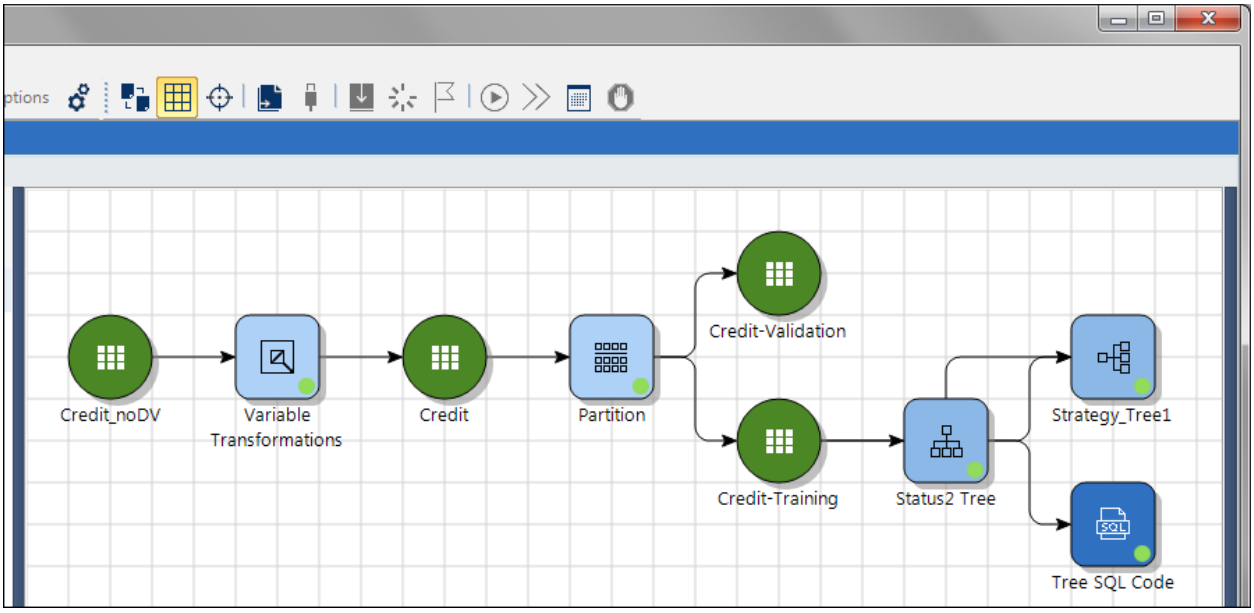


Undo

To undo an Auto-arrange action or any other node manipulation operation in the workflow, click the Undo button  in the toolbar. Click it several times if you need to undo a series of operations.

Show Grid

To show the grid in the background of the Workflow canvas, click the Show Grid button  in the toolbar. To hide the grid, click the same button again. When the grid is shown, nodes snap to the grid when being moved.



See also:

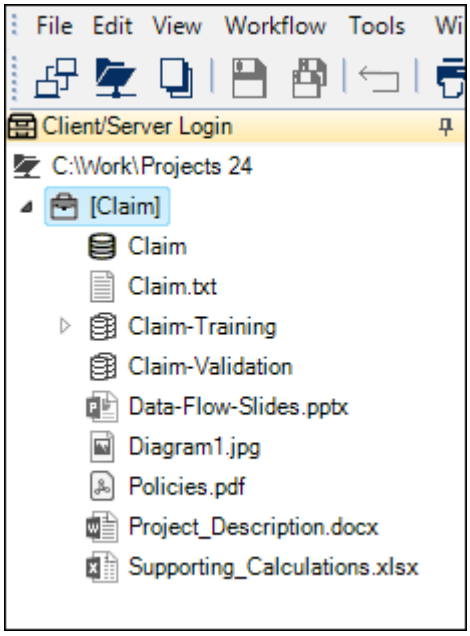
- Setting the Working Directory
- Projects

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Project Documents in Client-Server Mode

In general, by **project documents** we mean any files within the project folder that are associated with other programs, such as PDF files, Microsoft Office documents, text files, etc. PDFs or Office documents can be placed inside project folders for the purpose of project documentation to provide descriptions or comments. Charts or reports you export from the application views to Excel or PDF are often saved in the project folder by default. See also the topics Desktop Document Link and HTML Report Link.

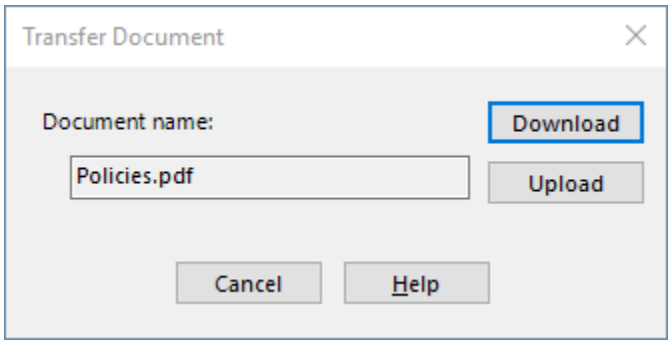
In the example screenshot below, several documents and an image file are present in the project folder.



The project document icons in the Project Explorer panel indicate their file associations. You can double-click on a project document in the Project Explorer or right-click on it and select **Open** from the context menu. Alternatively, select the item and press Enter. If you are working in the Standalone mode, the associated program on your machine will open the document.

If you are working in the **Client-Server** mode, the document will need to be transferred from the server to the client machine first:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Download** to transfer the file from the server to the client machine. The Windows Explorer browser dialog will open. Browse the client's file system and select the desired folder to save the document. The document will be opened automatically by the associated program on the client. For example, if you transfer an Excel document, it will will opened in Microsoft Excel, assuming Microsoft Office is installed on the client.

If, on the other hand, you need to synchronize a project document on the server with the local copy on the client, click the **Upload** button and browse for the local file. Upon selecting the file and clicking "Open", it will be transferred to the server and saved in the project folder, overwriting the target file there.

Note: It is not recommended to transfer large files between the server and the client using this interface. The transfer may a take a long time if the file size is dozens of Megabytes or more than a Gigabyte. In this case, use file transfer tools such as an SFTP client (WinSCP, FileZilla, etc.). If the server is in your Windows domain, you may also have access to your Working Directory via a mapped network folder or a UNC path.

Sample Projects

Prepare Sample Data

The **Prepare Sample Data** command under the **Help** menu allows you to install sample projects to the current working directory.

Each sample project contains a dataset and one or several models related to a particular industry such as Marketing Analytics or Credit Risk Management. See a brief description below.

Project	Industry	Description
Census	Marketing	This project contains a dataset census data and a decision tree model based on it. A marketing analyst may want to profile the data and identify the most important characteristics of high income customers in order to target them effectively.
Claim	Insurance	This project contains a dataset with auto insurance data and a decision tree. The model in this project was created to help solve the business problem of identifying potentially fraudulent claims.

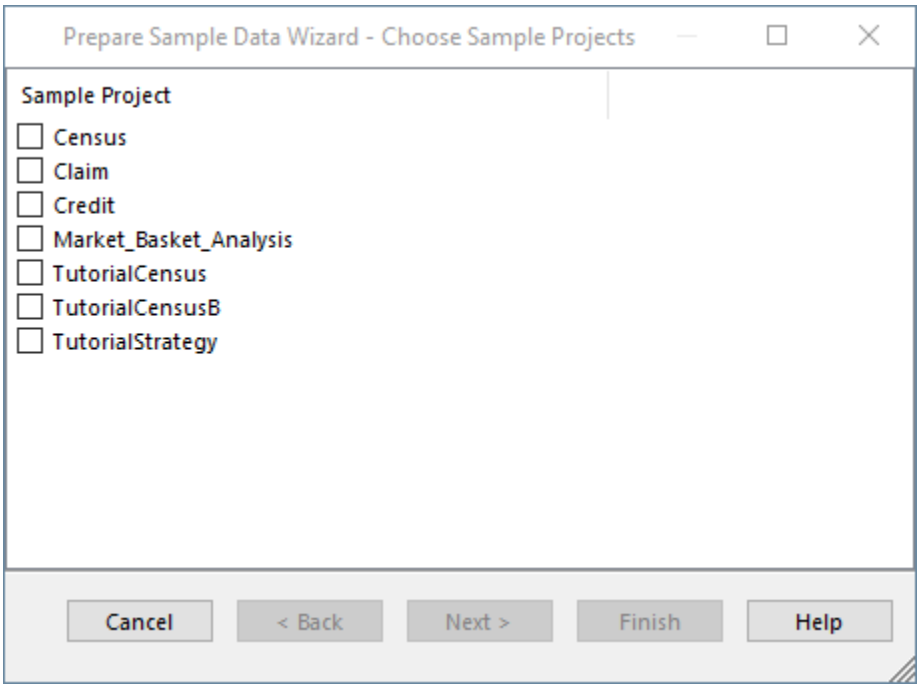
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Credit	Credit Risk	This project contains a dataset with credit accounts information and a decision tree with the account status as the dependent variable. A Credit Risk Analyst may use the tree to explore the most significant predictors of default (Bad Debt) and explore the segments with the worst debt percentage. A detailed Data Definition for the Credit dataset is available in the topic Credit Risk: Data Definition.
TutorialCensus	Marketing	A sample project with Census dataset for the Tutorial. See **Help | Tutorial**.
TutorialCensusB	Marketing Price Analytics	A sample project with the *Census* and *House prices* datasets for the advanced tutorial for Knowledge Studio users. See **Help | Tutorial.**
TutorialStrategy	Marketing	A sample project with Census dataset for the Strategy Tree tutorial. See **Help | Tutorial.**

To start deploying a sample project:

From the Menu

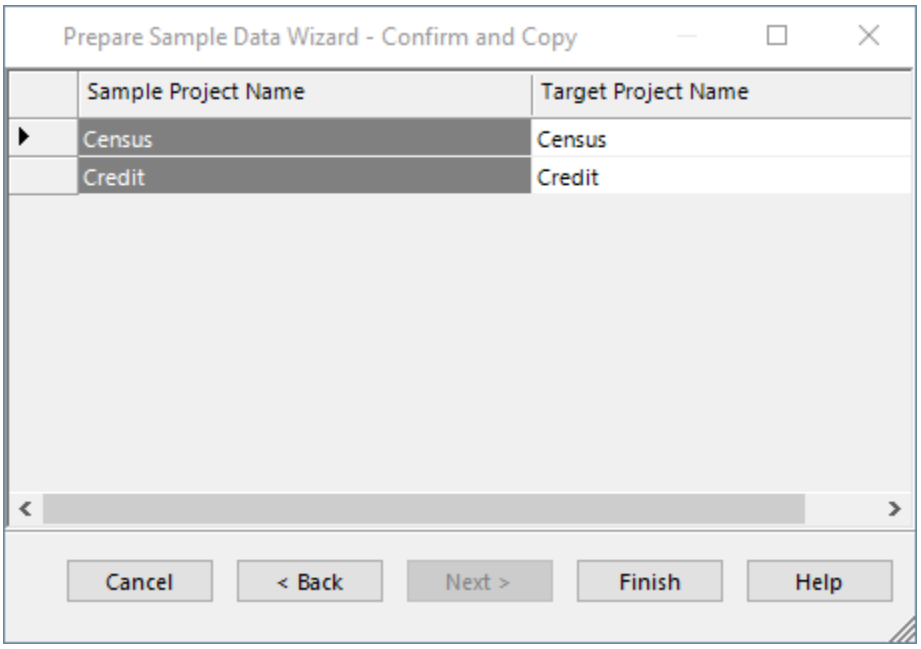
Help | Prepare Sample Data



Select the project you wish to deploy and click Next.

At the next step you can rename the projects if necessary.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Finish** to deploy the selected projects. The projects appear in the Project Explorer pane. Double-click the resulting projects to open them.

See also: Credit Risk: Data Definition

Data Definition for the Credit Example Dataset

In cooperation with the IT, Credit Approval and the Collection and Credit Management departments, the following data elements needed to address the stated business and data mining objectives have been collected for a total of 29167 customer accounts out of which about 1400 were classified as “Bad Debt” Status. The following provides a description of the fields in the dataset:

Age: Age of the customer in years rounded to the nearest integer.

Region: State of residence of the customer. To reduce cardinality, only the 12 most common states are specifically identified. The remaining states are grouped into a thirteenth category.

City: City of residence. In this case, it was found that there were too many cities and there was no reasonable way to aggregate them into fewer categories. Therefore, it was decided to drop this field.

Gender: Gender of credit applicant. This field contained a large number of missing values (more than 90%). Therefore, it was dropped from further modeling.

Agent ID: ID of the Agent who processed the credit application. This field was coded as A0 for the first agent, and A1 for second agent and so on. There are 100 agents. This field was originally an integer. However, to properly represent it as a categorical, unordered variable, it was necessary to add the character A to the codes to force it to be a string. In view of the fact that the purpose of the

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

study was to investigate the bad debt behavior, and the problem of grouping the agent ID's without introducing bias to the data, this field was not used in modeling.

Year at Address: Number of years spent at address as declared on the application rounded to the nearest integer. A missing value represents that the customer did not complete the field; while a zero means that, the customer spent less than a year at the current address.

Bank Reference: Number of bank accounts listed in application. This represents the total number of bank accounts, either with the same bank or different banks, even if the details are not given. Missing value or zero means no reference is made in the application.

Fin Co Ref: Number of reference to a financial institution used as the client's financial reference other than a regular bank. Missing value or zero means no reference is made in the application.

Dept Store Ref: Number of department store references on application. Missing value or zero means no reference is made in the application.

Bank Dda Ref: Number of bank checking account references declared on the application. Missing value or zero means no reference is made in the application.

Bank Save Ref: Number of Bank savings account reference on the application. Missing value or zero means no reference is made in the application.

Credit Card Ref: Number of Credit Card references on the application. Missing value or zero means no reference is made in the application.

Residential Status: Residential status of customer. This field originally contained many different categories. The three top recurring categories have been selected (own home, rent, live with parents) and the rest have been grouped as "other". A missing value indicates that the customer did not fill that field on the application.

Annual Income: Annual income of applicant as declared on application in thousands of dollars. However, given that the number of missing values in this field exceeded 90%, it was decided that this field could not reliably be used in modeling.

The following set of fields contain credit rating information with credit report summary. This particular credit agency provided the company with a credit rating on the customers in the form of five indicators representing a lower and an upper limit on a credit rating scale, the number of credit checks last year, the worst rating ever given to that customer and a recommended line of credit.

Recommended Line: Recommended value of credit to be given to the customer at time of the application.

Worst Rating Des: Worst rating designation given by the Credit Rating Agency to that customer at any previous time on record. This field shows any potential previous problems the customer had with his/her credit rating.

Year Last Check: The number of credit checks performed on this customer last year by the Credit Rating Agency. This is not necessarily a negative indicator for the credit rating of the customer. Every time a vendor offers any form of credit, that vendor may perform a check with the credit agency.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Major Derog Num: Major derogatory rating given by the credit rating agency. This number represents a lower limit on the credit rating scale used by the agency.

Satisfactory Num: The Satisfactory number given by credit rating agency. This is the upper limit on the credit scale.

The following set of fields contain the summary of the transaction history of the customers in the form of averages on the monthly balances, the amount purchased, the total service charge revenue from each account, and the total profit made.

Avg Monthly Balance: Average monthly balance incurred by the customer. The period over which the average value is calculated was taken as the last two years. For customers who had been with the company for less than two years, the average was calculated on the basis of their actual tenure in months. For customers identified as bad debt, the average was calculated as the time period from the month when they started using their credit card to the month they stopping paying the bill.

Total Amount Purchased: Total amount purchased to-date using this credit card.

Total Service Charge Revenue: Total service charges charged on this credit card.

Profit: Profit achieved by the company on this credit card account. In the cases of customers ending up with bad debt and a loss for the company, the profit was entered as a negative value. The profit number also includes the estimated cost of managing the credit account with the customer, with the exception of the additional costs incurred in attempting to collect the bad debts.

The following set of fields characterize the information about the company's process of managing the customers' credit accounts. They contain the aggregation of the data representing the different steps used by the company to prevent uncollected debts from the point of application processing, by giving an initial score, to the different levels of attempting to collect the bills.

Initial Score: The score given to this account at initial stage of evaluating application by the internal credit-scoring group in the company. This score is independent of the scores obtained by the credit agency.

Number Inquiries: Number of inquiries made about the customer during initial phase of application processing. These inquiries included the checks made on the different information the customer provided on the application such as bank accounts and credit card accounts.

Delinquent Lev 1: This field records the number of times the customer has been sent a friendly letter reminding him/her of late payment. ABC handled these letters.

Delinquent Lev 2: The number of times the customer has been sent a "strong language" letter for late payment. ABC also handled these letters.

Delinquent Lev 3: The number of letters sent by the collection agency to that customer. A zero value indicates that the customer's balance has not been sent to the collection agency.

Amount Lost: Amount of money lost due to unpaid balance. This amount does not include the expenses of managing the account nor the fees paid to the collection agency in the cases of bad debt.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Credit Line: The Actual credit limit the customer given to the customer.

Status: Status of account. The categories of the account status were “Bad Debt”, “OK” or “Closed”. The category “Closed” was for accounts that were “OK” but cancelled by the customer. The “Closed” accounts were treated as “OK” accounts from the point of view no bad debt.

Task Scheduler

The **Task Scheduler** allows you to schedule the execution of one or more functional nodes in Altair workflows. Scheduled tasks can run even in unattended mode when the application is closed and the user is logged out. This can be especially beneficial for complex tasks or very large data volumes that take a lot of time to process.

The task scheduler is available only for the **Multiple Scenario Optimization** node in Knowledge Studio.

The scheduler can be invoked in three ways:

- from the node wizard, using the **Schedule** button in one of the wizard pages
- from the context menu when right-clicking on the desired node
- from the main menu, using the **Scheduled Tasks** icon  (in this context it only shows the list of existing scheduled tasks and does not allow to schedule new tasks).

If scheduled tasks on your computer can only be created by users with administrative privileges, you may need to start Knowledge Studio **as administrator** before using the scheduling feature.

On Windows 10, 11, and Windows Server 2016, 2019, 2022, you will also need to edit the properties of the newly created task in Windows Task Scheduler. See the section *Modifying Scheduled Task Properties* below.

Scheduled Tasks and Licensing

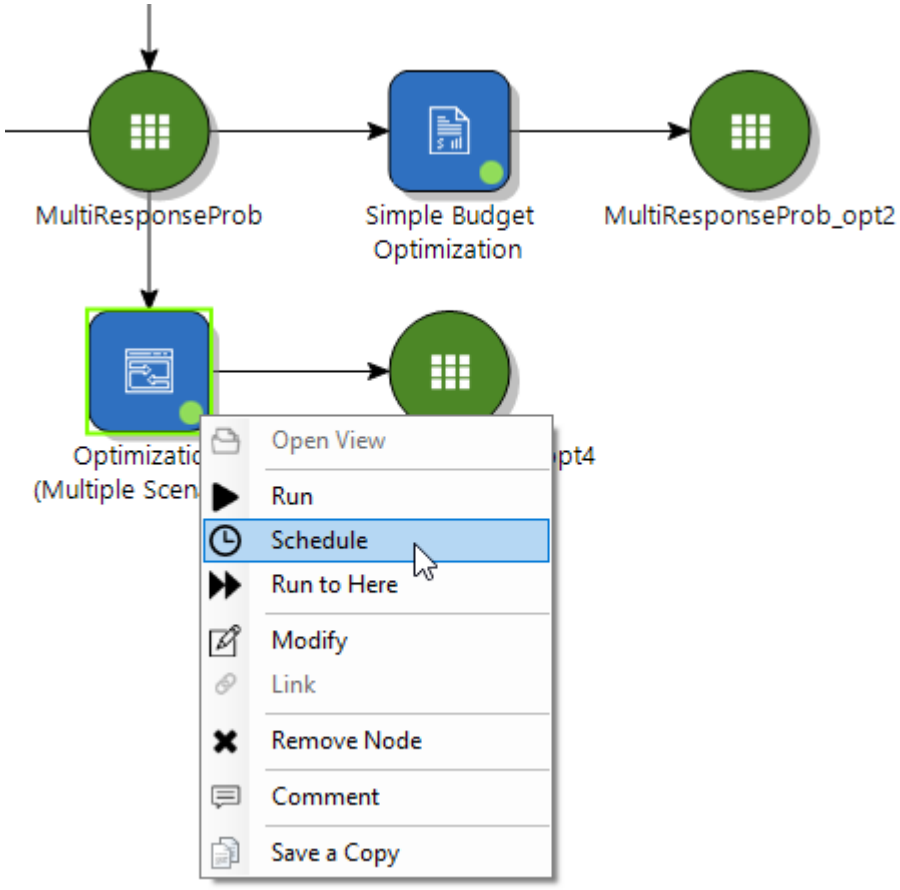
A scheduled task launches a separate instance of the application executable in the background, and therefore it will perform a separate license check when it starts.

- In the case of a named-user license, if you modify the task properties in Task Scheduler or cron, make sure the task is configured to run under the account of a licensed user.
- In the case of a concurrent license, the instance launched by the scheduled task will use a new connection, so it will run only if connections are available at the scheduled start time. Since tasks are usually scheduled to run in unattended mode when the application is closed, one user ID would not normally take more than one connection. However, the concurrent connections limit should be taken into account when scheduling multiple tasks.

Multiple Scenario Optimization node

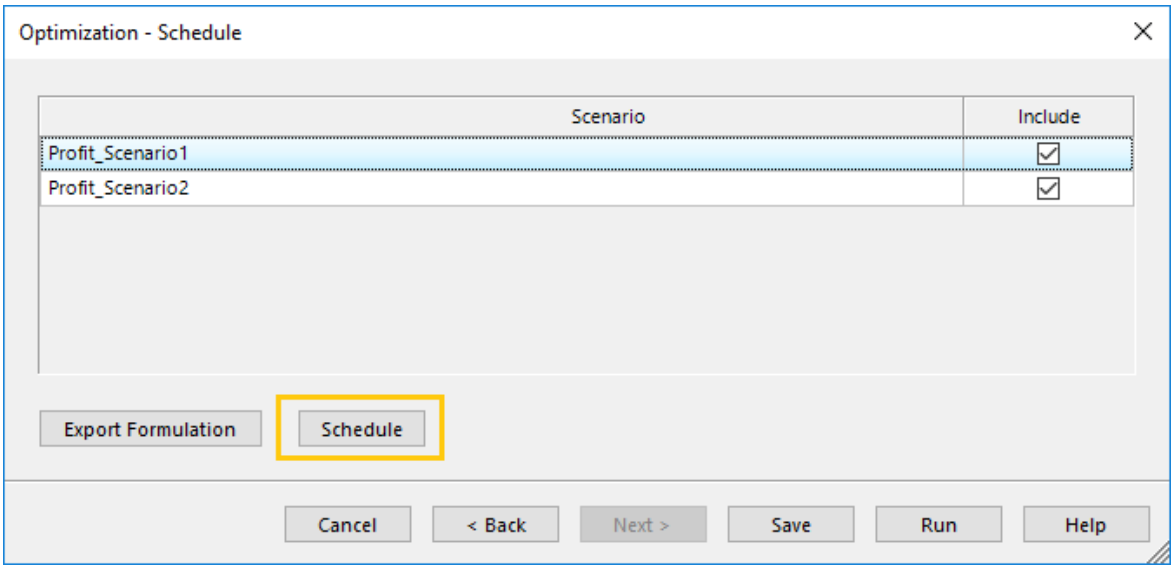
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

In Knowledge Studio, you can schedule the execution of optimization tasks for Multiple Scenario Optimization. If all settings in the node wizard have been defined, right-click on the node and select the **Schedule** command from the context menu:

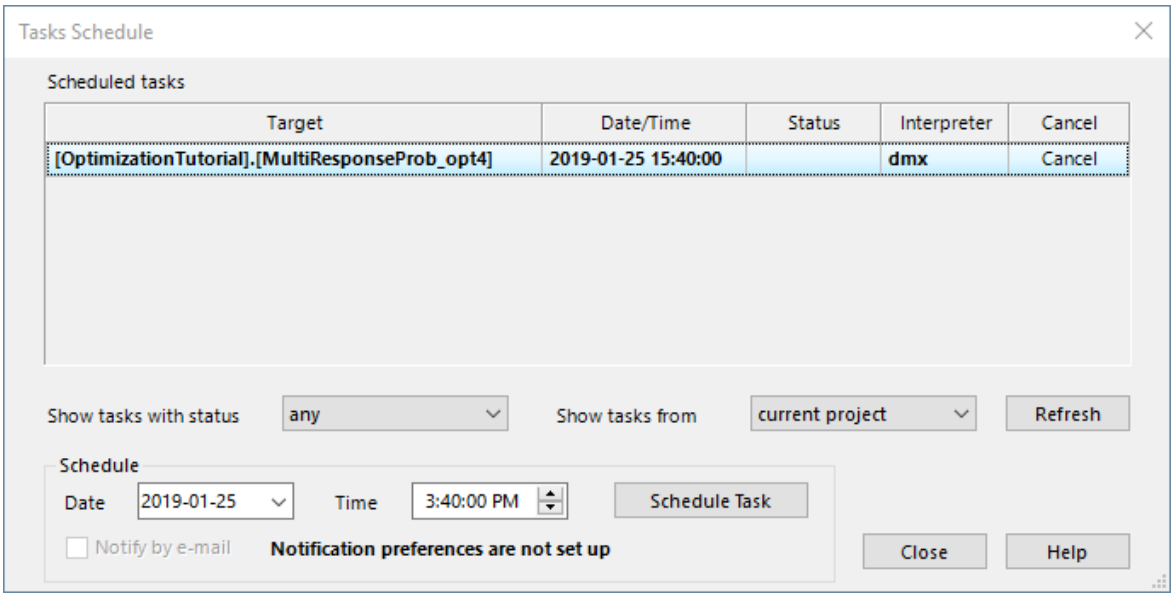


Alternatively, double-click the Multiple Scenario Optimization node to open the wizard and proceed to the last page. Select the required scenarios in the *Include* column and click the **Schedule** button.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



This opens the Tasks Schedule dialog, where you can view the previously defined tasks (if any) along with their status and properties, and schedule a new task to run the selected scenarios.



- To schedule a new task, specify the desired date and time to run it, then click the **Schedule Task** button. A new line representing this task will appear in the list above.
- To cancel an existing task, click **Cancel** for this task in the Cancel column on the right.

Once the task has been scheduled, close the Tasks Schedule dialog. The Multiple Scenario Optimization node will have a clock icon indicating that the task defined by the node is scheduled to run.

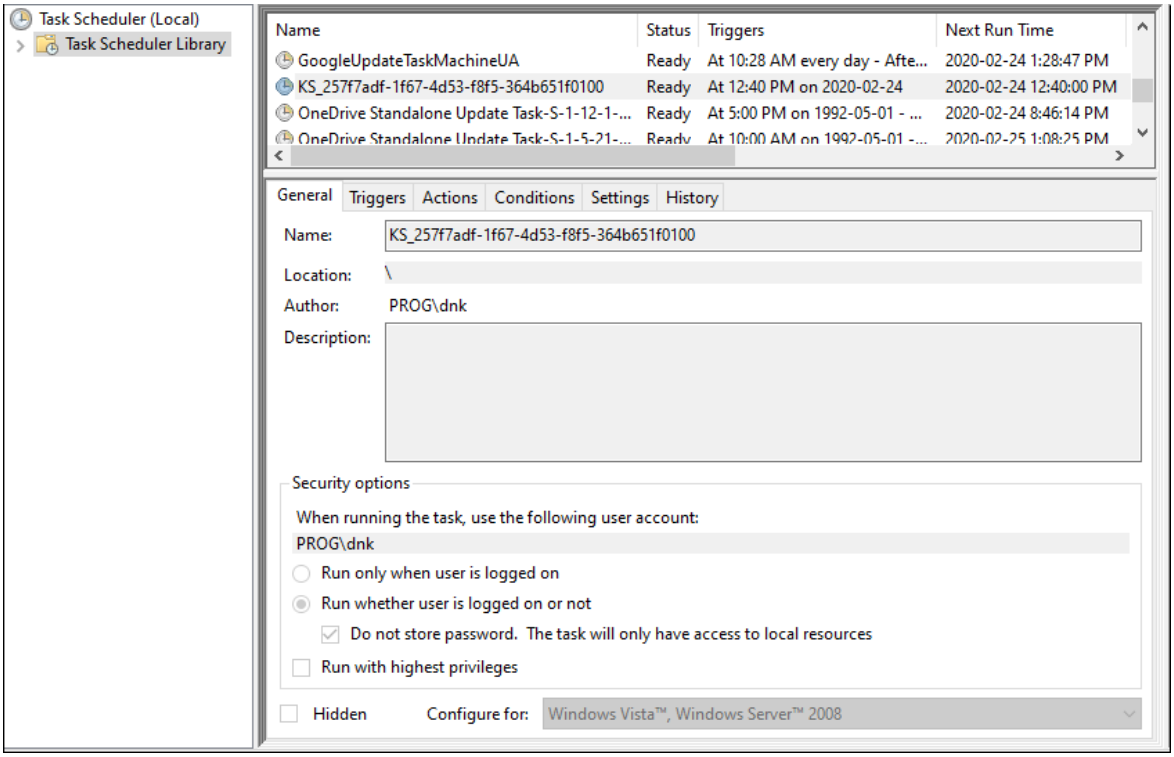
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Modifying Scheduled Task Properties

The scheduled task properties can be modified if necessary using Windows Task Scheduler (on Windows) or *cron* (on Linux).

For example, to modify the task properties on Windows, open Task Scheduler from Windows Administrative Tools and select *Task Scheduler Library* in the leftmost panel. The name of the task created by Knowledge Studio starts with "**KS**_". Double click on it and modify the settings in the *General*, *Triggers*, *Conditions*, and other tabs if necessary.

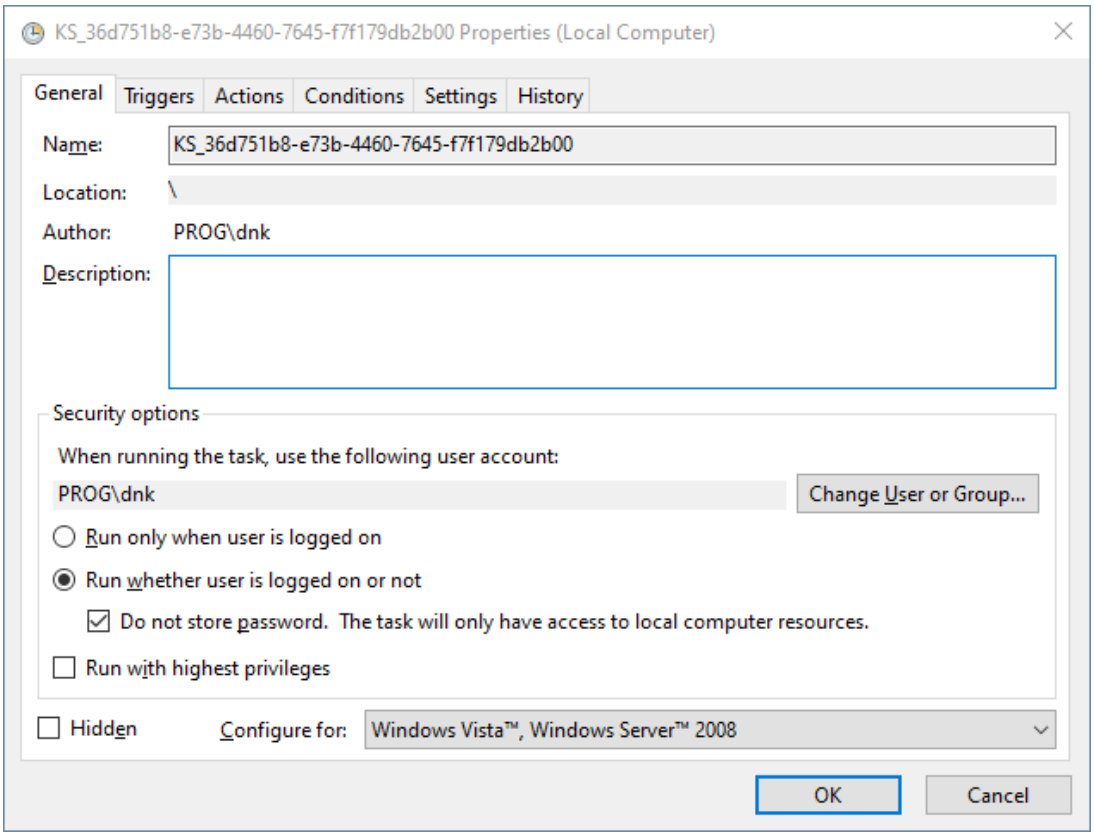


On Windows 10 and 11, Windows Server 2016, 2019, 2022, the following modifications may be necessary:

- In the ***General*** tab:
  - You may need to deselect the option "*Do not store password. The task will only have access to local resources*". When applying this change, you will need to specify the password for the account you chose for running the task.

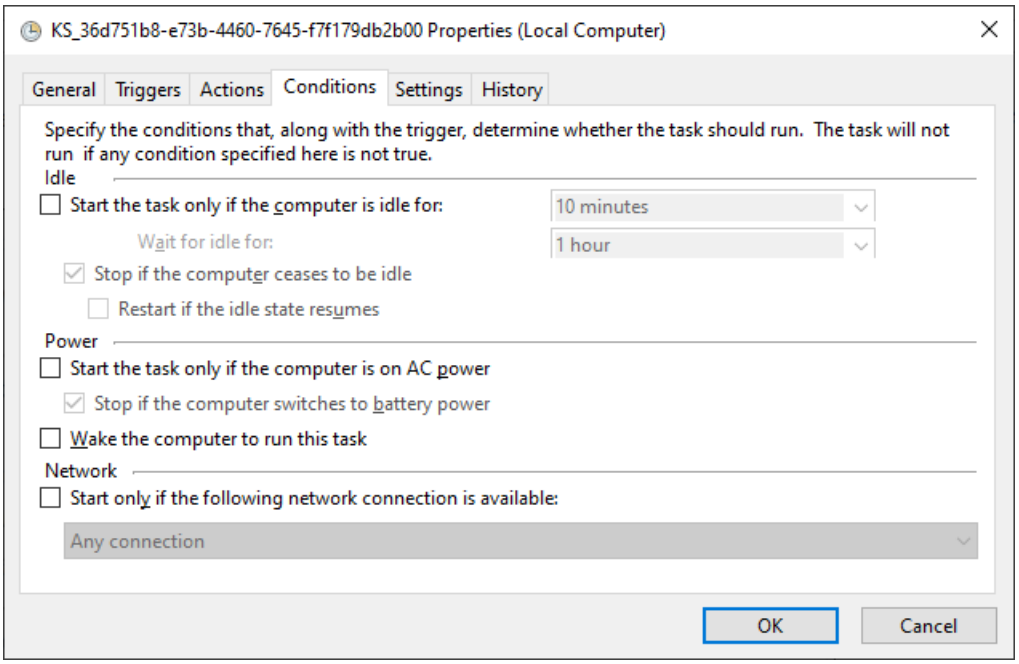
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- If necessary, click the button "*Change User or Group*" and specify a different user account for running the task. This must be an account of a licensed user. In the case of a concurrent license or Altair Units license, you can choose any user account as long as it has sufficient privileges to start the program as defined in the *Actions* tab. See the note on Scheduled Task and Licensing at the beginning of this topic.



- In the ***Conditions*** tab, specify the desired power and network availability options. Note: On laptops, the option "*Start the task only of the computer is on AC power*" is selected by default, so you may want to deselect it.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

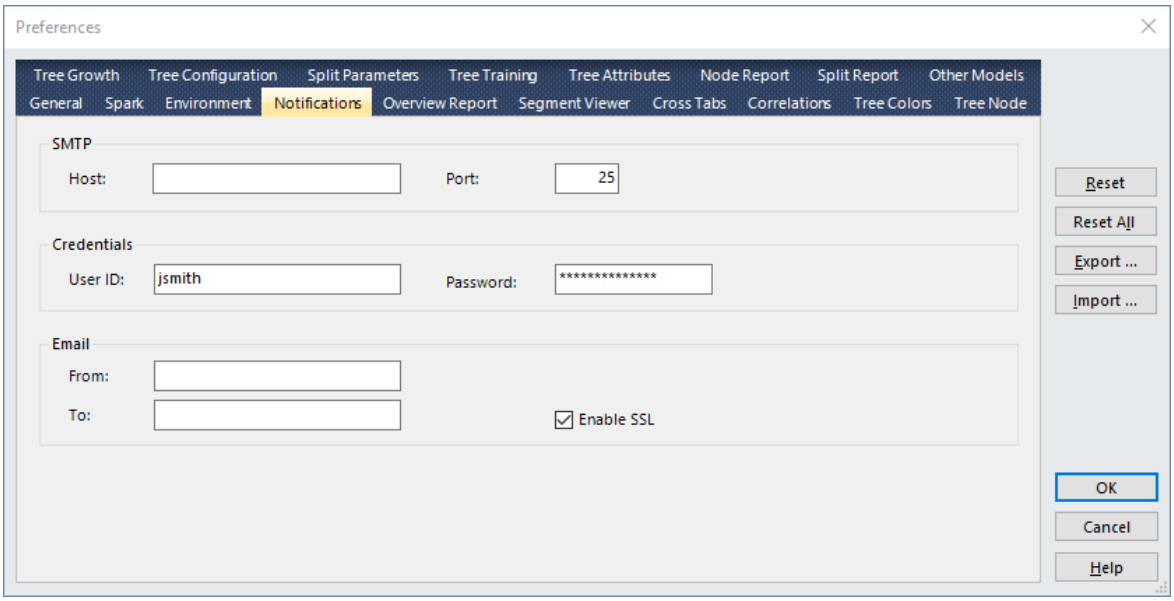


Email Notification Options

If you need to set up Email notifications in the scheduler to inform any user(s) of the task status changes, you need to set up Notification preferences in the application preferences under the **Tools** menu. After that, the checkbox **Notify by email** will be enabled at the bottom of the Tasks Schedule dialog, and you can select it if necessary. When setting the Email Notification options using **Tools | Preferences**, specify the SMTP host name or IP address, SMTP port, User ID and password (optional), and the *From/To* Email addresses. The option **Enable SSL** is selected by default, but you may need to disable it (depending on the security settings in your Email client). If you are not getting notifications after configuring these settings, deselect the Enable SSL option and try again.

The settings shown in the image below are given as an example template. Ask your IT team for assistance if you are not sure what parameters to specify.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Data Source Connections, Data Import and Export

Accessing Data Sources

To analyze your data, you must first make it accessible within your project by **importing** it from a source file or database into a *dataset* in the Altair Data Analytics native format (KDD). The KDD file format is a compressed binary file format optimized for the best performance of Altair models. The maximum data size that can be stored in a KDD file is 65534 columns (the 16-bit encoding limit: 2^16 - 2) and approximately 4.29 billion rows (the 32 bit-encoding limit: 2^32 - 2 = 4294967294).

With the In-Database Analytics connector you can perform in-place analysis of database tables in SQL Server, Teradata, or Oracle without importing data. Please see the In-Database Analytics topic for details.

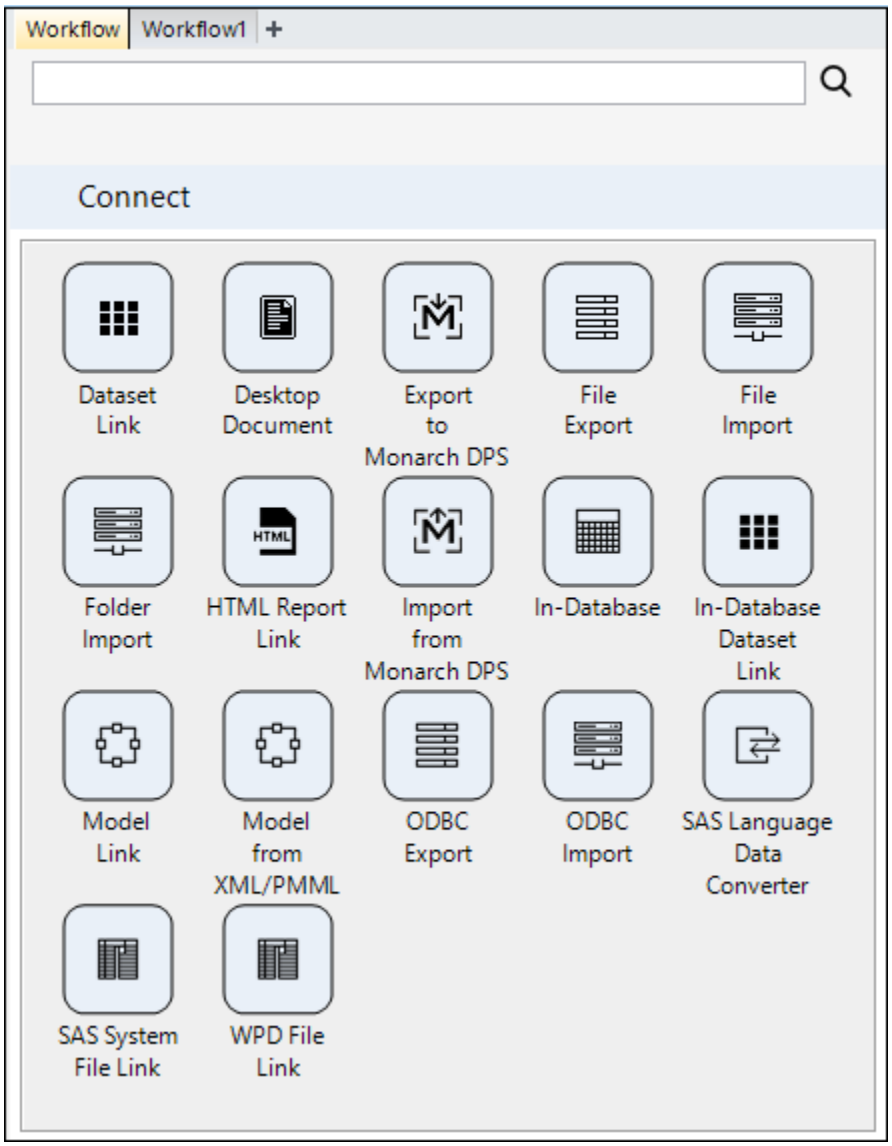
Once the data is accessible within the current project, it is represented by a *dataset object* in the Project Explorer panel and by a *dataset node* on the Workflow canvas. ***Dataset nodes*** on the Workflow canvas are ***references*** to physical datasets in the project. When you delete a dataset node from the canvas, you only delete a reference to the corresponding dataset. The referenced dataset object is **not** deleted. It remains in the *Project Explorer* panel. The physical dataset is deleted from the project folder only when you delete it from the Project Explorer panel, for example, using the right-click context menu. An in-database table can never be deleted from the source database via the Altair Data Analytics application interface. See the topics Projects and Workflows for details.

Finally, you can copy KDD datasets from other projects or repositories to the current project. See the **Dataset Link** subsection below.

Most common encodings in the source data are detected automatically. If your data contains Asian, Cyrillic, or other non-Latin-based character sets that are not in Unicode or UTF-8 encoding, you may need to explicitly specify the encoding of the data to be imported. For the list of supported encodings, see the section "*Note on Languages and Encodings"* at the end of this topic.

Data import, export, and other data access operations are listed in the **Connect** palette. Use the appropriate node to access your data.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



File Import

The File Import node allows you to import data from several popular file formats. Source files can be on the local file system or on network file systems as long as they are accessible from your computer or the application server (if you are using the client/server configuration).

The following source file formats are supported by the **File Import** node:

- **Text** (delimited and fixed-width)
- **Microsoft Excel**
  - Note: If your Excel file has more than one sheet and you want to import data from multiple sheets at once, use the ODBC Import node instead.
- **R**

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Data can be imported from R workspace files. The standard R file extension assumed by the wizard is ***.rdata**. The data file should contain at least one of the following kinds of objects: two dimensional matrices, vectors, factors, or dataframes. See https://www.r-project.org/ for reference.

- **SAS System Files** (*.sas7bdat, versions 8.x and 9.x)
- **SAS System transport** (*.ssd and *.tpt files generated by the XPORT engine)
- **SAS System CPORT** (*.stc files generated by the CPORT procedure; supported on Windows only)
- **SPSS Native** (*.sav)
- **SPSS Portable** (*.por)
- **Native Altair datasets** (KDD file format)

See the **File Import** topic for the instructions on using the File Import node.

Folder Import

The **Folder Import** node is an extension of the File Import function that allows you to import data from all files in a selected folder at once.

Database Connectivity: Data Import from ODBC Sources and In-Database Analytics

Microsoft® Windows platforms

Knowledge Seeker and Knowledge Studio for Windows support data import for all database systems accessible via ODBC, such as Microsoft SQL Server, Oracle, DB2, Teradata, Apache Hive, etc.

Note: For an ODBC data source to be accessible, a 64-bit ODBC driver for the source database must be present on the machine where the Altair Data Analytics software is installed. All ODBC drivers available on a Windows system are listed in the ODBC Data Source Administrator. See the topic Import from ODBC data sources.

In-Database Analytics is only supported for Microsoft SQL Server (including Parallel Data Warehouse), Oracle, and Teradata.

Linux

Altair Data Analytics software for Linux uses the **unixODBC** driver manager for database drivers. The ODBC drivers are not included with KS Server. We recommend downloading and installing native ODBC drivers for the required databases. In most cases they are available for free on the database vendors' websites. The Altair Deployment Guide describes how to configure database connections. Please contact Customer Support for assistance if necessary.

See the topic Import from ODBC data sources for details.

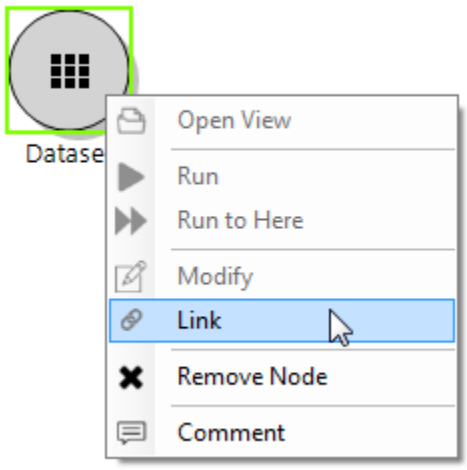
Dataset Link (Dataset)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The **Dataset Link** node in the **Connect** palette provides a quick way to put existing datasets (Altair Data Analytics data files) on the workflow canvas. If the data file you are linking to is not in the current project folder, the Dataset Link function first copies it to the current project folder, then creates a dataset node. The source dataset may or may not already be represented by a node on the canvas. Since a dataset node on the workflow canvas is a reference to the physical dataset (data file) in the Project Explorer panel, you can have as many nodes linked to the same physical dataset as desired.

Note: if you no longer need the import operation after importing a dataset (i.e., you only need the output dataset from now on, never going to rerun the import), you can delete the import process node from the canvas. This will not affect the dataset node: the reference to the output dataset created by this import operation will remain.

In some situations, some datasets in the project do not appear on the workflow canvas. For example, if you open a project created in an older version that did not support workflows, or if you manually copied the dataset files to the current project folder. To put them on the workflow, drag the **Dataset Link** node to the canvas and link it to the desired dataset using the **Link** command in the right-click context menu. This brings up the Link Dataset dialog. Browse for the desired dataset and click OK.



For details, see the Dataset Link topic.

A Note on Languages and Encodings

With Altair Data Analytics software, you can analyze data in any language. However, depending on the language and the source data format, you may have to explicitly specify a particular encoding in the Insert Dataset Wizard.

You may also need to check your Regional and Language Settings on your Windows system to make sure that it supports the languages you need. Some languages may not be installed by default and you will have to add them.

If you are using the ODBC Import Driver or the In-Database Analytics connector, in order to ensure data in specific languages is imported properly, you may need to include the following statements to the ODBC Connection String: "CHARSET=UTF-8" or "FWC=T" (Force SQL_WCHAR Support

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Enabled). Refer to the ODBC Connection Setup Help topic or ask your System Administrator if necessary.

The list of supported encodings:

UTF-8	8-bit transformation format of ISO 10646 (Unicode)
UTF-16	16-bit transformation format for ISO 10646 (Unicode)
UCS-2	UCS-2, an encoding of ISO 10646 (Unicode)
ASCII	American Standard Code for Information Interchange
SHIFT-JIS	Japanese Shift-JIS (same as CP932)
UTF-7	UTF-7, A Mail-Safe Transformation Format of Unicode
ARMSCII-8	Armenian (Armenian Standard Code for Information Interchange)
BIG5	Big5, Traditional Chinese
BIG5-HKSCS	Big5 with Hong Kong extensions, Traditional Chinese
CP1133	Lao ISO-8
CP1250	Windows ANSI code page 1250 - Central Europe
CP1251	Windows ANSI code page 1251 - Cyrillic
CP1252	Windows ANSI code page 1252 - Latin 1
CP1253	Windows ANSI code page 1253 - Greek
CP1254	Windows ANSI code page 1254 - Turkish
CP1255	Windows ANSI code page 1255 - Hebrew
CP1256	Windows ANSI code page 1256 - Arabic
CP1257	Windows ANSI code page 1257 - Baltic
CP1258	Windows ANSI/OEM code page 1258 - Vietnamese
CP850	Windows OEM code page 850 - Multilingual Latin 1
CP862	Windows OEM code page 862 - Hebrew
CP866	Windows OEM code page 866 - Russian
CP874	Windows ANSI/OEM code page 874 - Thai
CP932	Windows ANSI/OEM code page 932 - Japanese Shift-JIS
CP949	Windows ANSI/OEM code page 949 - Korean
CP950	Windows ANSI/OEM code page 950 - Traditional Chinese Big5
EUC-CN	GB2312, EUC encoding, Simplified Chinese
EUC-JP	JIS X 0201, 0208, 0212, EUC encoding, Japanese
EUC-KR	KS C 5601, EUC encoding, Korean
EUC-TW	CNS11643 (Plane 1-3), EUC encoding, Traditional Chinese
GB18030	Chinese National Standard GB 18030-2000
GBK	GBK, Simplified Chinese (extension to the Chinese National Standard GB 2312-80 character set)
Georgian-Academy	Georgian (the University of Georgia standard)
Georgian-PS	Georgian (the Parliament - Soros Foundation standard)
HP-ROMAN8	HP-Roman8 - Latin (Western European) 8-bit ASCII-compatible character set used on HP-UX

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

HZ	HZ-GB2312 Simplified Chinese
ISO-2022-CN	Simplified and Traditional Chinese (supported character sets: ASCII, GB2312, CNS 11643 Plane1, CNS 11643 Plane2).
ISO-2022-CN-EXT	Simplified and Traditional Chinese (Extension of ISO-2022-CN supporting GB/T 12345, planes 3 to 7 of CNS 11653, and GB 7590).
ISO-2022-JP	Japanese (supported character sets: JIS X 0201, JIS C 6226, JIS X 0208).
ISO-2022-JP-1	Japanese
ISO-2022-JP-2	Japanese (Extension of ISO-2022-JP supporting JIS X 0212, GB2312, and KS C 5601).
ISO-2022-KR	Korean (supported character sets: ASCII and KS C 5601)
ISO-8859-1	ISO-8859-1 Latin 1 (Western European)
ISO-8859-2	ISO-8859-2 Latin 2 (Central and Eastern Europe - non-Cyrillic)
ISO-8859-3	ISO-8859-3 Latin 3 (Esperanto, Maltese, and Turkish)
ISO-8859-4	ISO-8859-4 Latin 4 (Baltic)
ISO-8859-5	ISO-8859-5 (Cyrillic)
ISO-8859-6	ISO-8859-6 (Arabic)
ISO-8859-7	ISO-8859-7 (Greek)
ISO-8859-8	ISO-8859-8 (Hebrew)
ISO-8859-9	ISO-8859-9 Latin 5
ISO-8859-10	ISO-8859-10 Latin 10 (Scandinavian; includes Lapp and Icelandic)
ISO-8859-13	ISO-8859-13 (Baltic; including Latvian)
ISO-8859-14	ISO-8859-14 Latin 8 (Celtic)
ISO-8859-15	ISO-8859-15 Latin 9 (specialization of ISO-8859-1; includes Euro sign)
JOHAB	Korean - Johab
KOI8-R	Russian - KOI8-R
KOI8-U	Ukranian - KOI8-U
MacArabic	Macintosh Arabic
MacCentralEurope	Macintosh Latin-2
MacCroatian	Macintosh Croatian
MacCyrillic	Macintosh Cyrillic
MacGreek	Macintosh Greek
MacHebrew	Macintosh Hebrew
MacIceland	Macintosh Icelandic
MacRoman	Macintosh Roman
MacRomania	Macintosh Romania
MacThai	Macintosh Thai
MacTurkish	Macintosh Turkish
MacUkraine	Macintosh Ukraine
MuleLao-1	Laotian (An extension of ASCII for Laotian)
NEXTSTEP	NeXTSTEP Encoding Vector
TCVN	Vietnamese (Vietnamese national standard)
TIS-620	Thai (Thai Industrial Standard 620-2533)
VISCII	Vietnamese (Vietnamese Standard Code for Information Interchange)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

See also:

Dataset Link

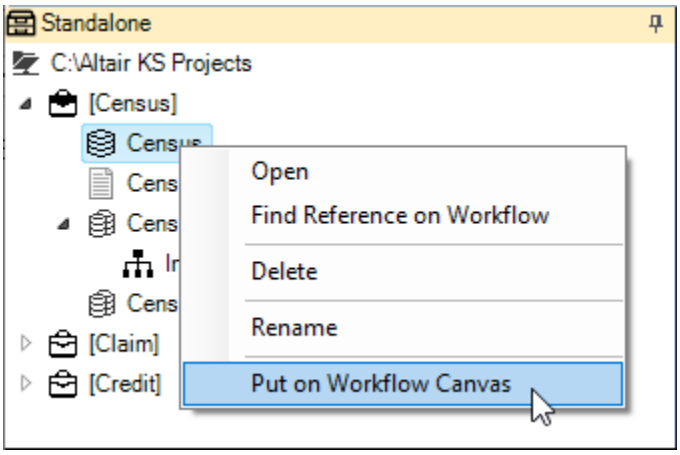
In-Database Analytics

Dataset Link

Altair Data Analytics (KDD) data files can be used as data nodes in the workflow directly, with no need for the data import step.

To link an existing KDD data file to a dataset node on the workflow canvas:

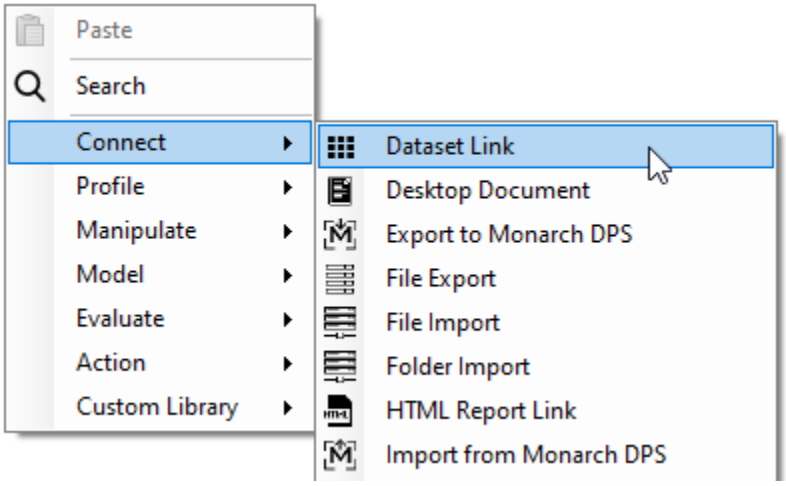
- If the dataset is in the current project folder and is shown in the Project Explorer panel, right-click on it in the Project Explorer and select the command **Put on Workflow Canvas** from the context menu.



Otherwise, if the KS data file you wish to put on the Workflow canvas is not in the current project folder:

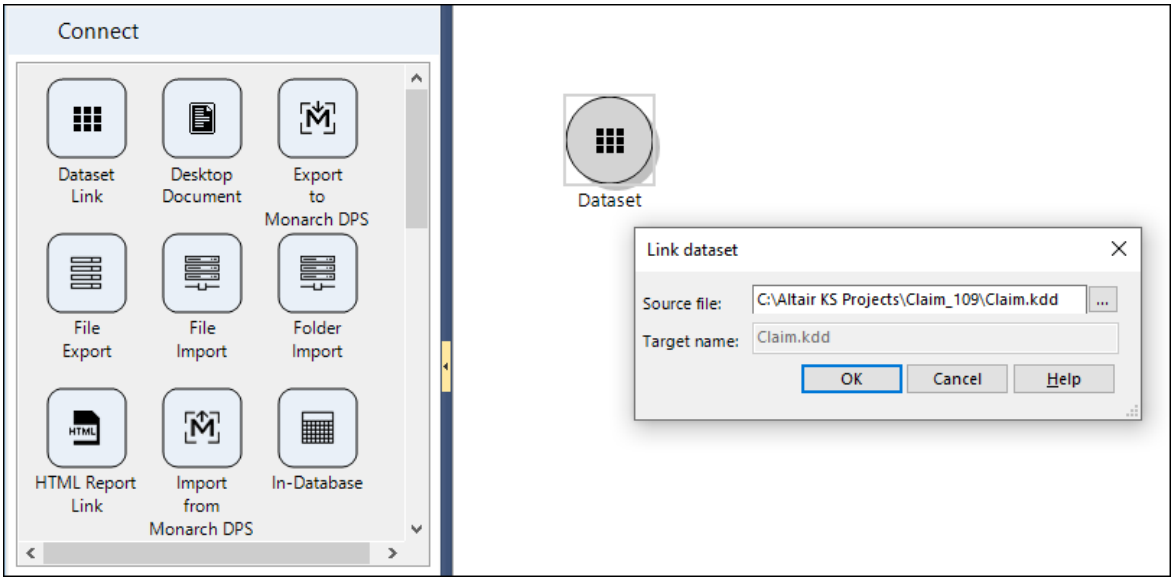
- Right-click anywhere on the Workflow canvas background. From the context menu, select **Connect --> Dataset Link** as shown in the example below. A new **Dataset** link node will be created.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



OR:

- Drag the **Dataset Link** node from the **Connect** palette onto the canvas. Then double-click the link node or right-click on it and select **Link** from the context menu.

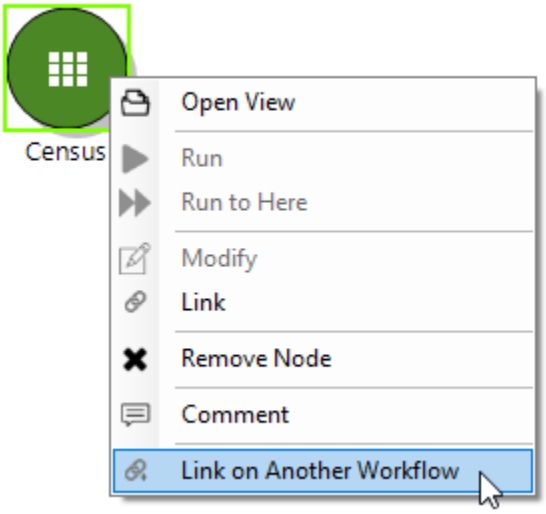


- Browse for the source file and specify its name in the Source File field
- Click OK. The source data file will be copied to the current project folder, shown in the Project Explorer, and linked as a new dataset node on the Workflow canvas.

Double-click the dataset to view its contents if necessary.

For an existing dataset node on a workflow canvas, you can create a copy of it in a new workflow. The **Link on Another Workflow** command on the right-click context menu will automatically create a new workflow and put the dataset node there.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

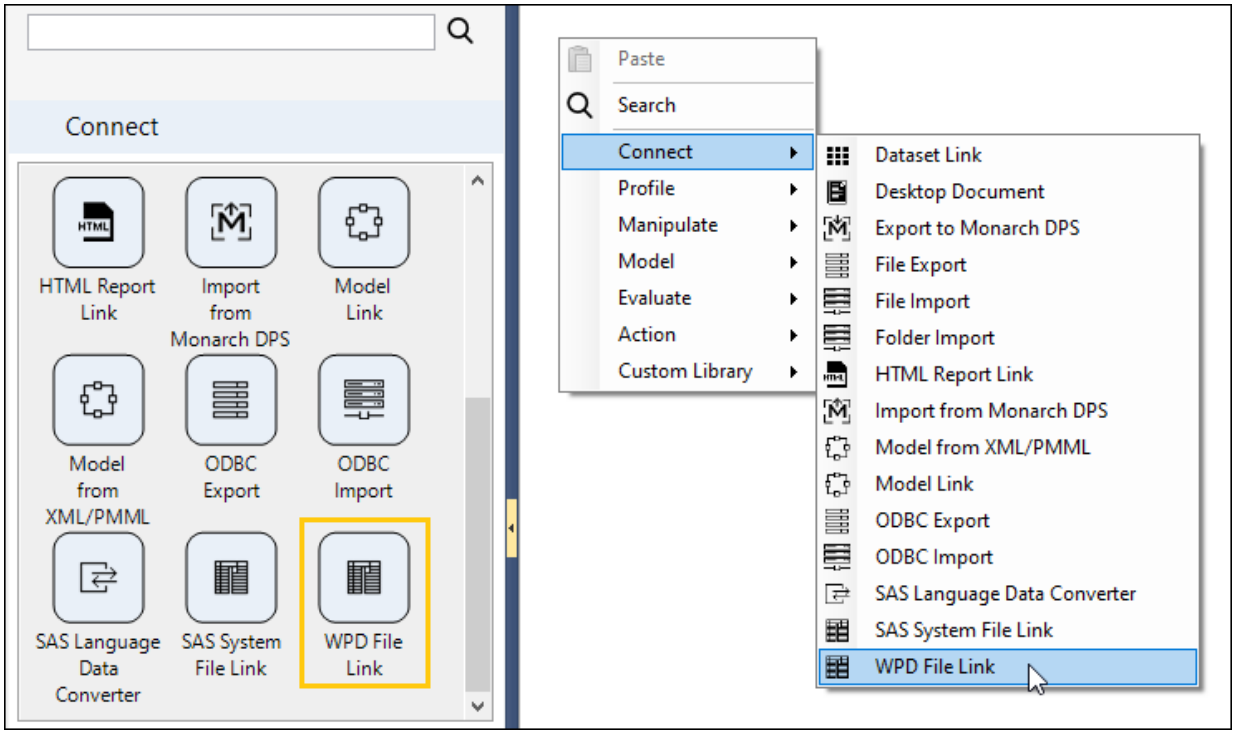


See also: Importing Data

WPD files (*.wpd) are data files in the format of Altair Analytics Workbench (WPS Workbench). They can be converted to Altair KDD and SAS System file formats using the **SAS Language Data Converter** node.

To create a dataset node linked to a WPD file:

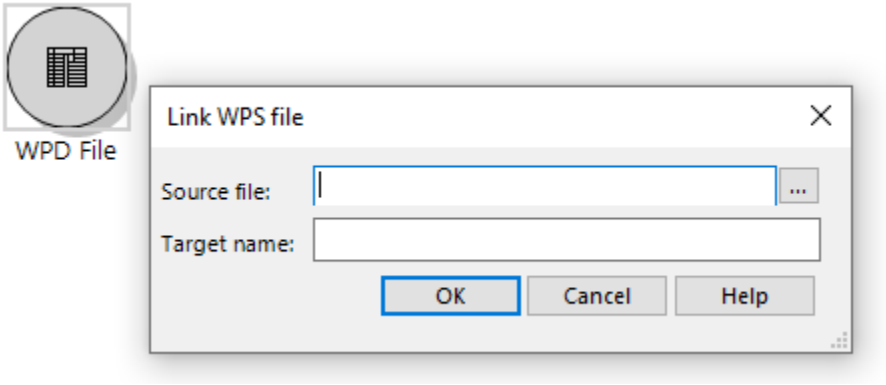
- Right-click anywhere on the Workflow canvas and select the context menu command **Connect --> WPD File Link** as shown below. A new **WPD File** will be created, and the file link dialog will open.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

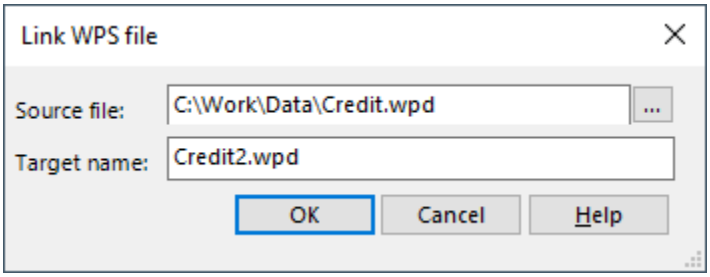
OR:

- Drag the **WPD File Link** node from the **Connect** palette onto the canvas. Then double-click the node or right-click on it and select **Link** from the context menu.



In the Link WPD File dialog:

- Browse for the source file (*.sas7bdat) and specify its name in the *Source file* field
- In the *Target name* field, specify the name for the new dataset node that the file will be linked to. If you specify the target name without extension, the *.sas7bdat extension will be appended automatically.

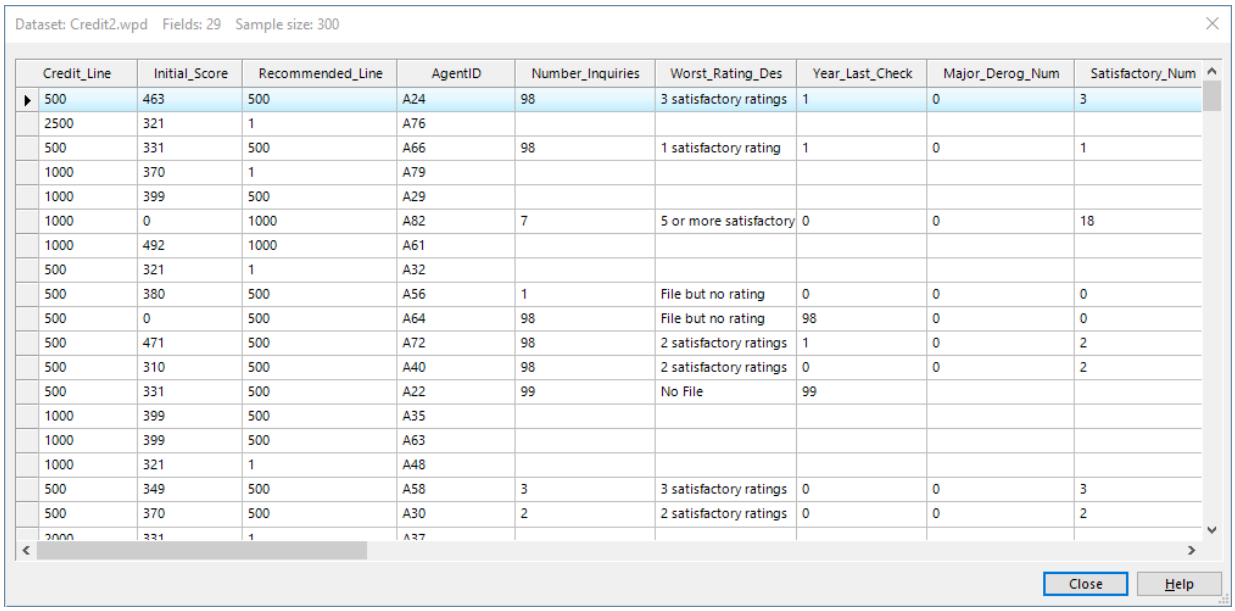


- Click **OK**. The source file will be copied to the current project folder, shown in the Project Explorer pane, and linked as a new dataset node on the Workflow canvas.



Double-click the dataset node to view its contents if necessary. This will open the data preview window, which shows the top 300 rows of the dataset. The data cells in the grid are not editable.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

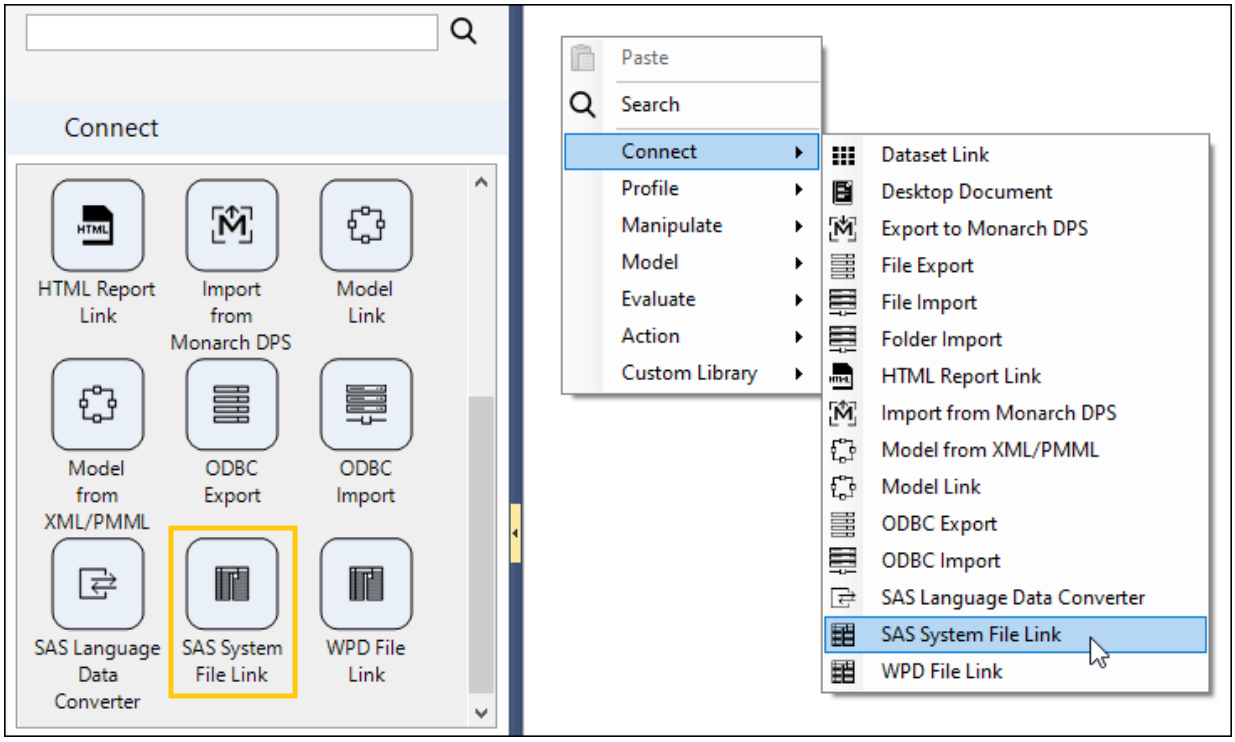


Credit_Line	Initial_Score	Recommended_Line	AgentID	Number_Inquiries	Worst_Rating_Des	Year_Last_Check	Major_Derog_Num	Satisfactory_Num
500	463	500	A24	98	3 satisfactory ratings	1	0	3
2500	321	1	A76					
500	331	500	A66	98	1 satisfactory rating	1	0	1
1000	370	1	A79					
1000	399	500	A29					
1000	0	1000	A82	7	5 or more satisfactory	0	0	18
1000	492	1000	A61					
500	321	1	A32					
500	380	500	A56	1	File but no rating	0	0	0
500	0	500	A64	98	File but no rating	98	0	0
500	471	500	A72	98	2 satisfactory ratings	1	0	2
500	310	500	A40	98	2 satisfactory ratings	0	0	2
500	331	500	A22	99	No File	99		
1000	399	500	A35					
1000	399	500	A63					
1000	321	1	A48					
500	349	500	A58	3	3 satisfactory ratings	0	0	3
500	370	500	A30	2	2 satisfactory ratings	0	0	2

SAS System files (*.sas7bdat) can be used as data nodes in the Workflow. They can be converted to Altair KDD and WPD formats using the SAS Language Data Converter node.

To create a dataset node linked to a SAS System file:

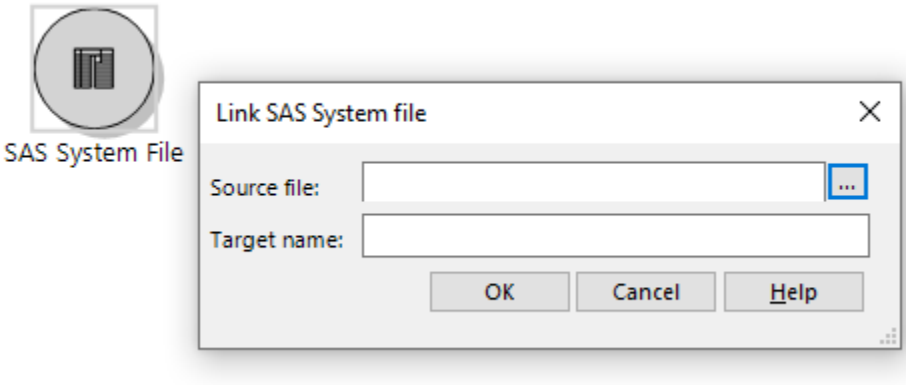
- Right-click anywhere on the Workflow canvas and select the context menu command **Connect --> SAS System File Link** as shown below. A new **SAS System File** will be created, and the file link dialog will open.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

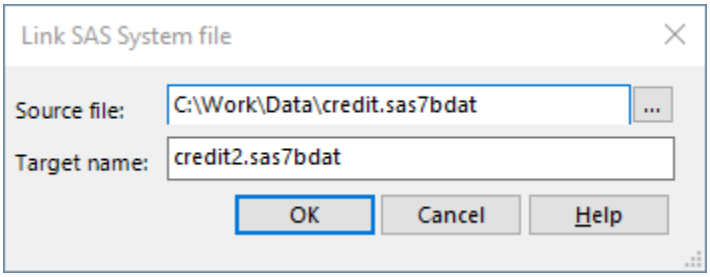
OR:

- Drag the **SAS System File Link** node from the **Connect** palette onto the canvas. Then double-click the node or right-click on it and select **Link** from the context menu.



In the Link SAS System File dialog:

- Browse for the source file (*.sas7bdat) and specify its name in the *Source file* field
- In the *Target name* field, specify the name for the new dataset node that the file will be linked to. If you specify the target name without extension, the *.sas7bdat extension will be appended automatically.

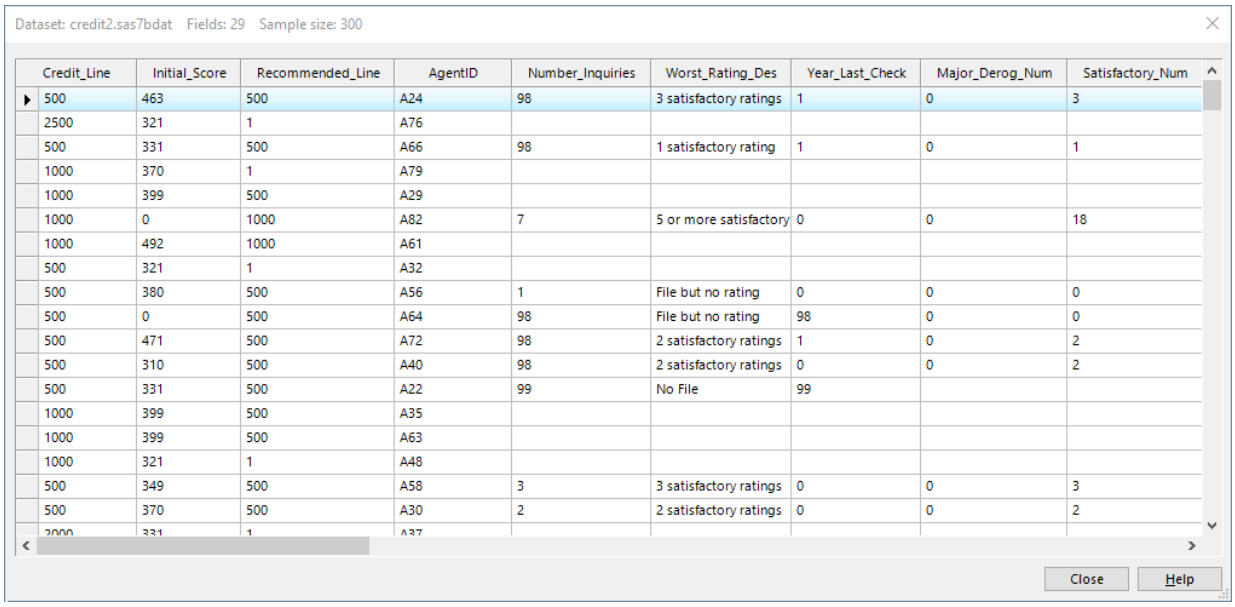


- Click **OK**. The source file will be copied to the current project folder, shown in the Project Explorer pane, and linked as a new dataset node on the Workflow canvas.



Double-click the dataset node to view its contents if necessary. This will open the data preview window, which shows the top 300 rows of the dataset. The data cells in the grid are not editable.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Credit_Line	Initial_Score	Recommended_Line	AgentID	Number_Inquiries	Worst_Rating_Des	Year_Last_Check	Major_Derog_Num	Satisfactory_Num
500	463	500	A24	98	3 satisfactory ratings	1	0	3
2500	321	1	A76					
500	331	500	A66	98	1 satisfactory rating	1	0	1
1000	370	1	A79					
1000	399	500	A29					
1000	0	1000	A82	7	5 or more satisfactory	0	0	18
1000	492	1000	A61					
500	321	1	A32					
500	380	500	A56	1	File but no rating	0	0	0
500	0	500	A64	98	File but no rating	98	0	0
500	471	500	A72	98	2 satisfactory ratings	1	0	2
500	310	500	A40	98	2 satisfactory ratings	0	0	2
500	331	500	A22	99	No File	99		
1000	399	500	A35					
1000	399	500	A63					
1000	321	1	A48					
500	349	500	A58	3	3 satisfactory ratings	0	0	3
500	370	500	A30	2	2 satisfactory ratings	0	0	2
2000	331	1	A37					

File Import

Select Source

For general information on accessing data for analysis, see the topic Accessing Data Sources.

The **File Import** node allows you to import data from several popular file formats. Source files can be on the local file system or network file systems as long as they are accessible from your computer or the application server (if you are using the client/server configuration).

Data is imported into a *dataset* in the Altair Data Analytics native format (KDD), which is a compressed binary file format. The maximum data size that can be imported is 65534 columns (the 16-bit encoding limit: 2^16 - 2) and approximately 4.29 billion rows (the 32 bit-encoding limit: 2^32 - 2 = 4294967294).

The following source file formats are supported:

- **Text** (delimited and fixed-width)
- **Microsoft Excel** (*.xlsx, *.xls, *.xlsm, *.xlsb)
  - Note: To import data from ***.csv** files, use the **Text** source file type
- **R**
  - R workspace files. The standard R file extension assumed by the wizard is ***.rdata**. The data file should contain at least one of the following kinds of objects: two dimensional matrices, vectors, factors, or dataframes. See https://www.r-project.org/ for reference.
- **SAS System Files** (*.sas7bdat)
- **SAS System transport Files** (*.ssd and *.tpt files generated by the XPORT engine)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- **SAS System CPORT Files** (*.stc files generated by the CPORT procedure; supported on Windows only)
- **SPSS Native** (*.sav)
- **SPSS Portable** (*.por)
- **Native Altair datasets** (KDD file format)
    - These are data files in a compressed binary format optimized for the best performance of predictive models and other functions in Knowledge Studio and Knowledge Seeker. Any native Altair dataset (**KDD** file format) is represented by two or three files:
        - <dataset_name>.kdd
        - <dataset_name>.ksd-data
        - <dataset_name>.ksd (optional; the *.ksd component may not be present for partitions and some other contexts)
    - The **KDD Import** node can be used as an equivalent of the Copy/Paste or "Save As" commands for datasets in the current project, or to import datasets from other projects, repositories, or network folders.
    - **Note**: if the source dataset has any expressions for calculated columns added using Variable Transformations, Scoring, or other operations, the expressions will be removed in the resulting dataset. More precisely, the calculations will be instantiated (similar to the Paste Values option in Microsoft Excel when copying and pasting data from cells with formulas), which means that the data will remain, but the columns will no longer have expressions.

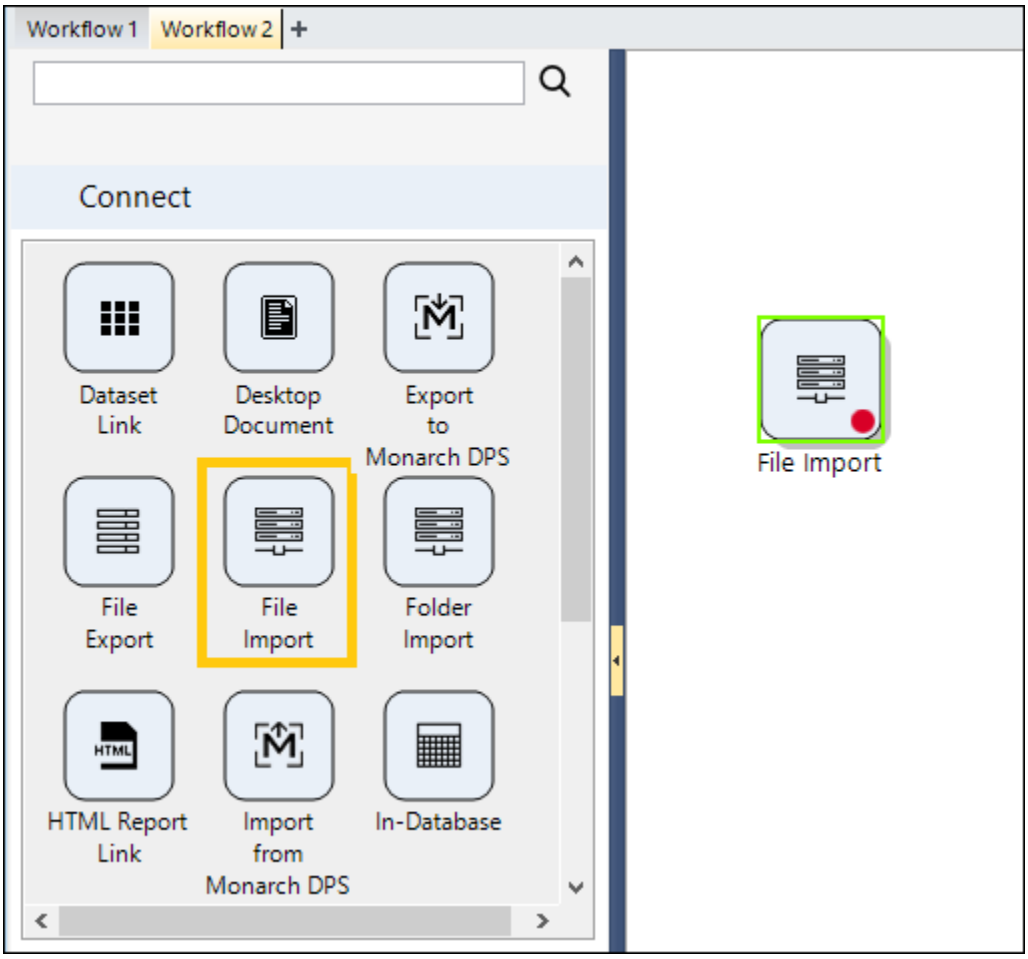
To import data from a file, use any of the following ways:

- Drag the **File Import** node from the **Connect** palette onto the workflow canvas. Then double-click the node to open the wizard.

OR:

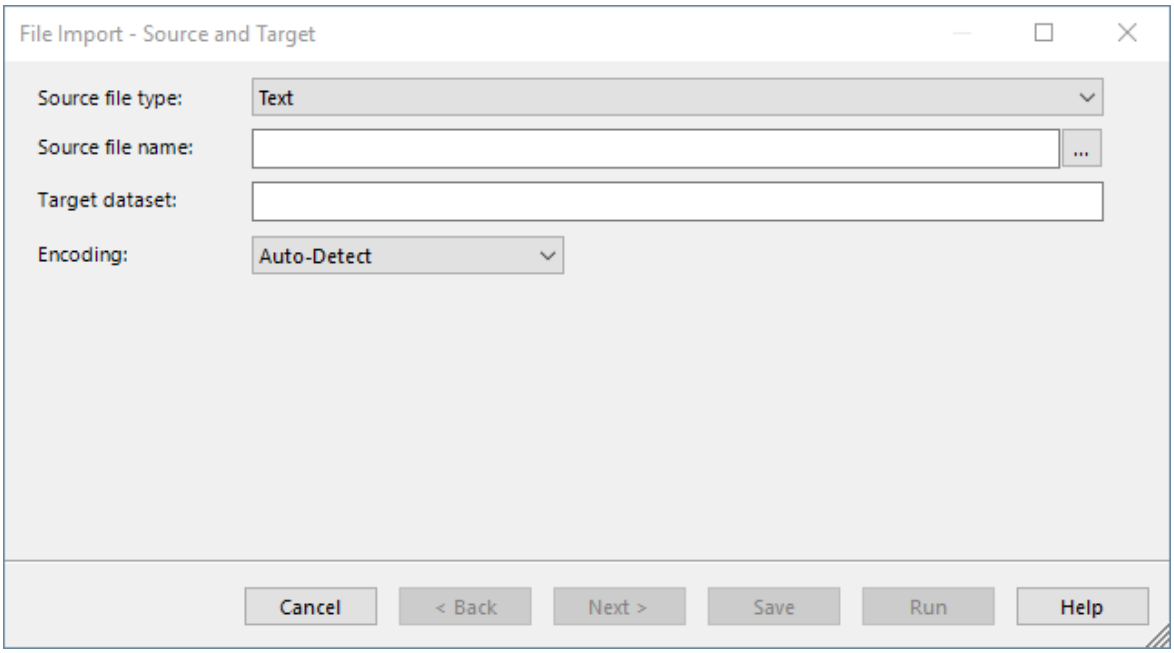
- Right-click anywhere on the Workflow canvas background. From the context menu, select **Connect --> File Import**. A new **File Import** node will be created and the wizard will open.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

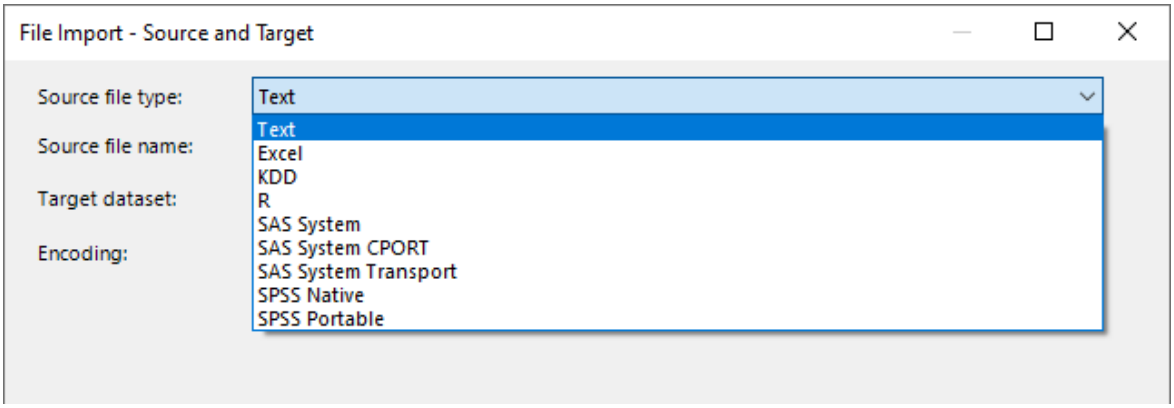


If the File Import node is already on the canvas, double-click it or right-click on it and select **Modify** from the context menu to open the wizard.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Select the desired source format from the *Source file type* drop-down list:

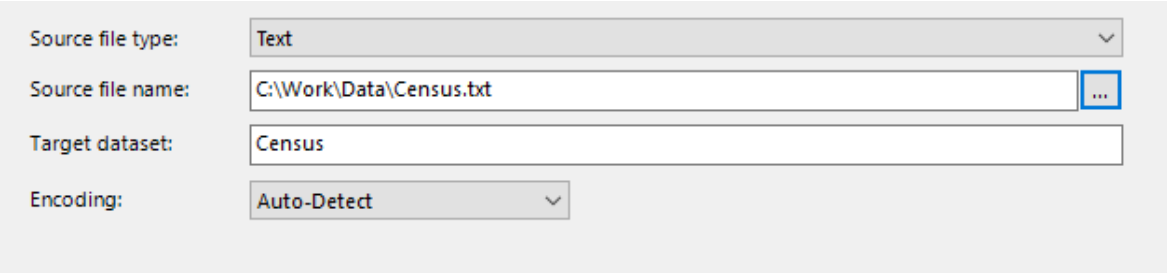


In the *Target dataset* field, enter the name of the new dataset to be created as a result. By default, it is the same as the name of the source file or table. The dataset name can contain spaces, but **cannot** contain the following special characters:

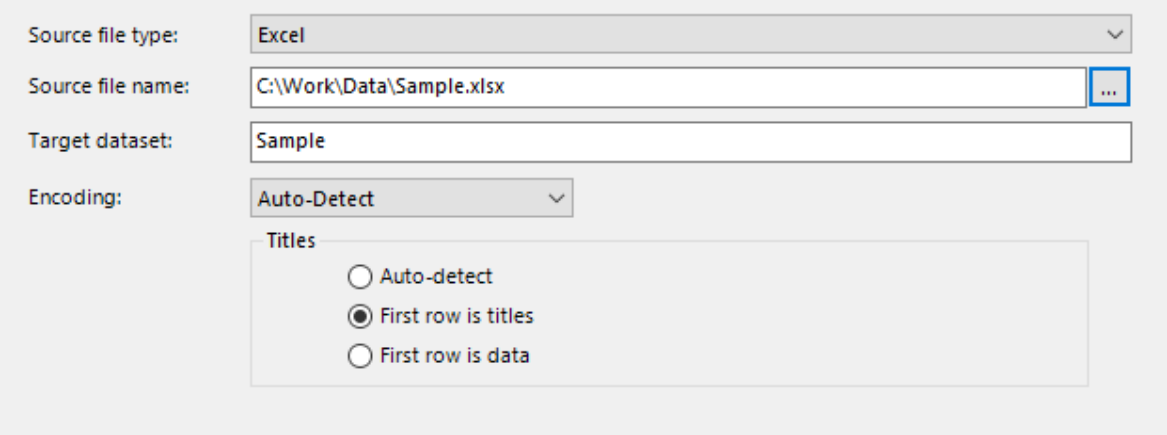
```
[ ] " * + # & ! % ( ) { } ` ' @ $ ^ , . / : ; < = > ? \ |
```

Specify the source file to import and the name for the new dataset. Enter the path to the source file or use the browse button to browse for the desired file. For example:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If you are importing an **Excel** spreadsheet, this wizard page includes additional options that determine how to treat the first row in the source spreadsheet. The option *First row is titles* is the default. Choose the option *First row is data* if you want to force the wizard to treat all rows as data. In this case the fields in the resulting dataset will be named automatically as C1, C2, C3, ... and so on. If your spreadsheet contains multiple worksheets, you will be able to select the desired worksheet at the next step.



When browsing for the source file, the following default file extensions are assumed:

Source File Type	**Default File Extension**
Microsoft Excel	*.xls,*.xlsx
Text	*.txt; *.csv
R	*.rdata
SAS System	*.sas7bdat
SAS System transport	*.ssd; *.tpt
SAS System CPORT	*.stc
SPSS Native	*.sav
SPSS Portable	*.por
KDD	*.kdd (native Altair Data Analytics data format)

If the extension of your source file is not typical for the selected format, choose All (*.*) as the file type in the file browser.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Click **Next** to continue specifying the import options (for example, exclude some fields or change data types, etc.). For the Text file type, the next step is Text File Preview. For other file types, the next step is Field Selector. You can also click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.

Encodings

With Altair Data Analytics software, you can analyze data in any language. The ***Auto-detect*** option for Encoding in the File Import wizard will automatically detect most common encodings.However, depending on the language and the source data format, you may have to explicitly specify a particular encoding.

You may also need to check your Regional and Language Settings on your Windows system to make sure that it supports the languages you need. Some languages may not be installed by default, and you will have to add them first.

The list of supported encodings:

UTF-8	8-bit transformation format of ISO 10646 (Unicode)
UTF-16	16-bit transformation format for ISO 10646 (Unicode)
UCS-2	UCS-2, an encoding of ISO 10646 (Unicode)
ASCII	American Standard Code for Information Interchange
SHIFT-JIS	Japanese Shift-JIS (same as CP932)
UTF-7	UTF-7, A Mail-Safe Transformation Format of Unicode
ARMSCII-8	Armenian (Armenian Standard Code for Information Interchange)
BIG5	Big5, Traditional Chinese
BIG5-HKSCS	Big5 with Hong Kong extensions, Traditional Chinese
CP1133	Lao ISO-8
CP1250	Windows ANSI code page 1250 - Central Europe
CP1251	Windows ANSI code page 1251 - Cyrillic
CP1252	Windows ANSI code page 1252 - Latin 1
CP1253	Windows ANSI code page 1253 - Greek
CP1254	Windows ANSI code page 1254 - Turkish
CP1255	Windows ANSI code page 1255 - Hebrew
CP1256	Windows ANSI code page 1256 - Arabic
CP1257	Windows ANSI code page 1257 - Baltic
CP1258	Windows ANSI/OEM code page 1258 - Vietnamese
CP850	Windows OEM code page 850 - Multilingual Latin 1
CP862	Windows OEM code page 862 - Hebrew
CP866	Windows OEM code page 866 - Russian
CP874	Windows ANSI/OEM code page 874 - Thai
CP932	Windows ANSI/OEM code page 932 - Japanese Shift-JIS
CP949	Windows ANSI/OEM code page 949 - Korean
CP950	Windows ANSI/OEM code page 950 - Traditional Chinese Big5

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

EUC-CN	GB2312, EUC encoding, Simplified Chinese
EUC-JP	JIS X 0201, 0208, 0212, EUC encoding, Japanese
EUC-KR	KS C 5601, EUC encoding, Korean
EUC-TW	CNS11643 (Plane 1-3), EUC encoding, Traditional Chinese
GB18030	Chinese National Standard GB 18030-2000
GBK	GBK, Simplified Chinese (extension to the Chinese National Standard GB 2312-80 character set)
Georgian-Academy	Georgian (the University of Georgia standard)
Georgian-PS	Georgian (the Parliament - Soros Foundation standard)
HP-ROMAN8	HP-Roman8 - Latin (Western European) 8-bit ASCII-compatible character set used on HP-UX
HZ	HZ-GB2312 Simplified Chinese
ISO-2022-CN	Simplified and Traditional Chinese (supported character sets: ASCII, GB2312, CNS 11643 Plane1, CNS 11643 Plane2).
ISO-2022-CN-EXT	Simplified and Traditional Chinese (Extension of ISO-2022-CN supporting GB/T 12345, planes 3 to 7 of CNS 11653, and GB 7590).
ISO-2022-JP	Japanese (supported character sets: JIS X 0201, JIS C 6226, JIS X 0208).
ISO-2022-JP-1	Japanese
ISO-2022-JP-2	Japanese (Extension of ISO-2022-JP supporting JIS X 0212, GB2312, and KS C 5601).
ISO-2022-KR	Korean (supported character sets: ASCII and KS C 5601)
ISO-8859-1	ISO-8859-1 Latin 1 (Western European)
ISO-8859-2	ISO-8859-2 Latin 2 (Central and Eastern Europe - non-Cyrillic)
ISO-8859-3	ISO-8859-3 Latin 3 (Esperanto, Maltese, and Turkish)
ISO-8859-4	ISO-8859-4 Latin 4 (Baltic)
ISO-8859-5	ISO-8859-5 (Cyrillic)
ISO-8859-6	ISO-8859-6 (Arabic)
ISO-8859-7	ISO-8859-7 (Greek)
ISO-8859-8	ISO-8859-8 (Hebrew)
ISO-8859-9	ISO-8859-9 Latin 5
ISO-8859-10	ISO-8859-10 Latin 10 (Scandinavian; includes Lapp and Icelandic)
ISO-8859-13	ISO-8859-13 (Baltic; including Latvian)
ISO-8859-14	ISO-8859-14 Latin 8 (Celtic)
ISO-8859-15	ISO-8859-15 Latin 9 (specialization of ISO-8859-1; includes Euro sign)
JOHAB	Korean - Johab
KOI8-R	Russian - KOI8-R
KOI8-U	Ukranian - KOI8-U
MacArabic	Macintosh Arabic
MacCentralEurope	Macintosh Latin-2
MacCroatian	Macintosh Croatian
MacCyrillic	Macintosh Cyrillic
MacGreek	Macintosh Greek

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

MacHebrew	Macintosh Hebrew
MacIceland	Macintosh Icelandic
MacRoman	Macintosh Roman
MacRomania	Macintosh Romania
MacThai	Macintosh Thai
MacTurkish	Macintosh Turkish
MacUkraine	Macintosh Ukraine
MuleLao-1	Laotian (An extension of ASCII for Laotian)
NEXTSTEP	NeXTSTEP Encoding Vector
TCVN	Vietnamese (Vietnamese national standard)
TIS-620	Thai (Thai Industrial Standard 620-2533)
VISCII	Vietnamese (Vietnamese Standard Code for Information Interchange)

Known Issues

RData file import

The **File Import** node may fail to import a valid R data file with the following exception reported "*There was an error importing the dataset. Make sure that the correct driver for the type of file was used. The following exception occurred*: *E00007E: EXTERNAL: IMPORT: The input RData file contains 0 members available for transfer".* This may be caused by some extra classes in the source RData file.

Workaround: Load the RData file into R or RStudio and remove these classes as follows:

```
table <- load('filename.RData')
str(table)
  chr "df"
str(df)
  Classes 'tbl_df', 'tbl' and 'data.frame': <# observations/variables reported>
table1 = as.data.frame(df)
save(table1, file='filename_clean.RData')
```

Upon removing the classes (in this case, 'tbl_df' and 'tbl') with the `as.data.frame()` call, the revised table saved back into an RData file `filename_clean.RData` can be successfully imported by the **File Import** node.

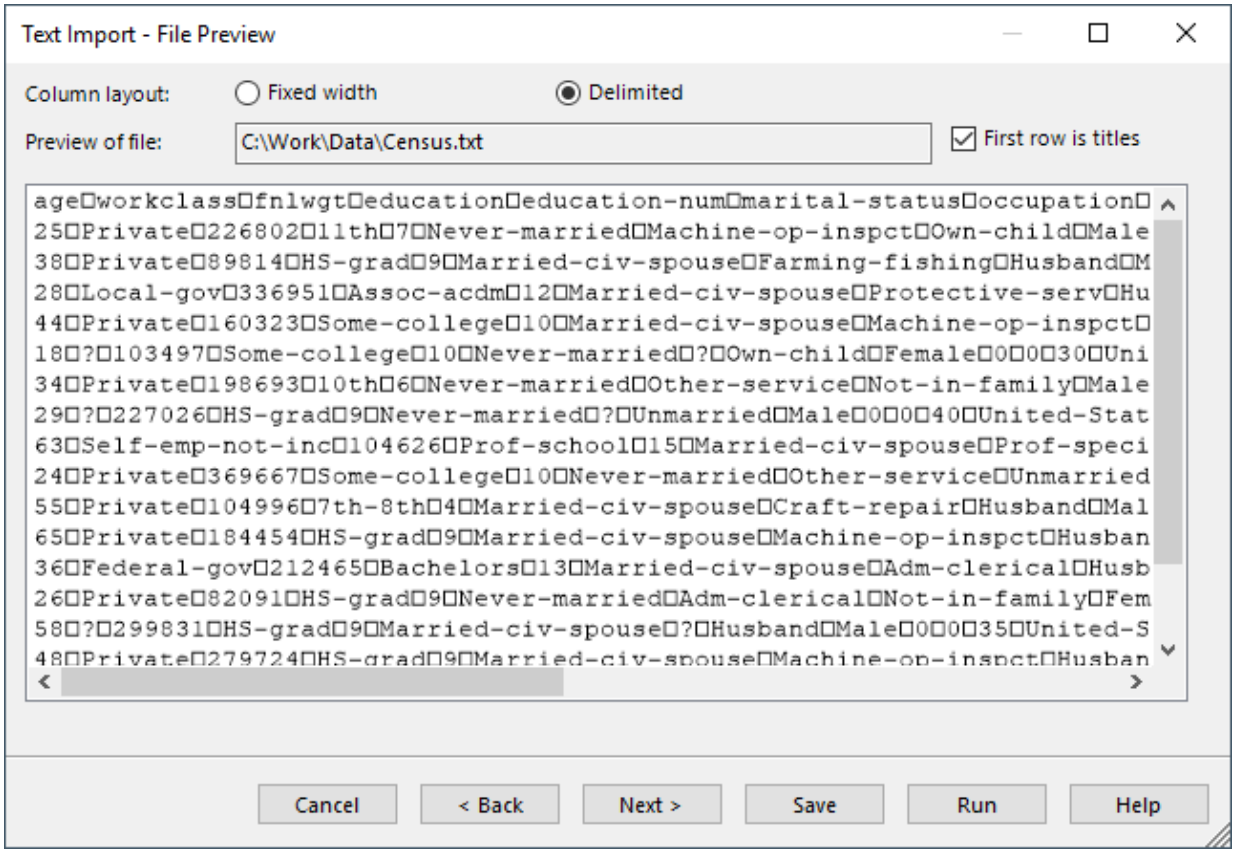
Proceed to the next step to continue specifying the import options. For the Text file type, the next step is Text File Preview. For other file types, the next step is Field Selector.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Text File Preview

This wizard step allows you to preview the text file you are importing and choose the column layout.

Note: If the chose the wrong encoding at the first step of the wizard, or if the encoding could not be auto-detected, the data in the preview window will not be displayed correctly. In this case, use the **Back** button to return to the first page and choose the correct encoding. For the list of supported encodings, see "A Note on Languages and Encodings" in the topic File Import: Select Source.



Column Layout

Specify the column layout of your source file: *Fixed Width* or *Delimited*. By default, it is assumed that the source file is delimited.

First Row is Titles

Check this box if the first row of your file contains the column names. The import takes the first row of data as the column names and starts importing data starting from the second row.

Both **Next** and **Run** buttons are enabled. At this point you can press **Run** to start the import with default settings, or click **Next** to continue specifying the import options.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If you choose the *Delimited* file option, the next step allows you to specify the text delimiter.

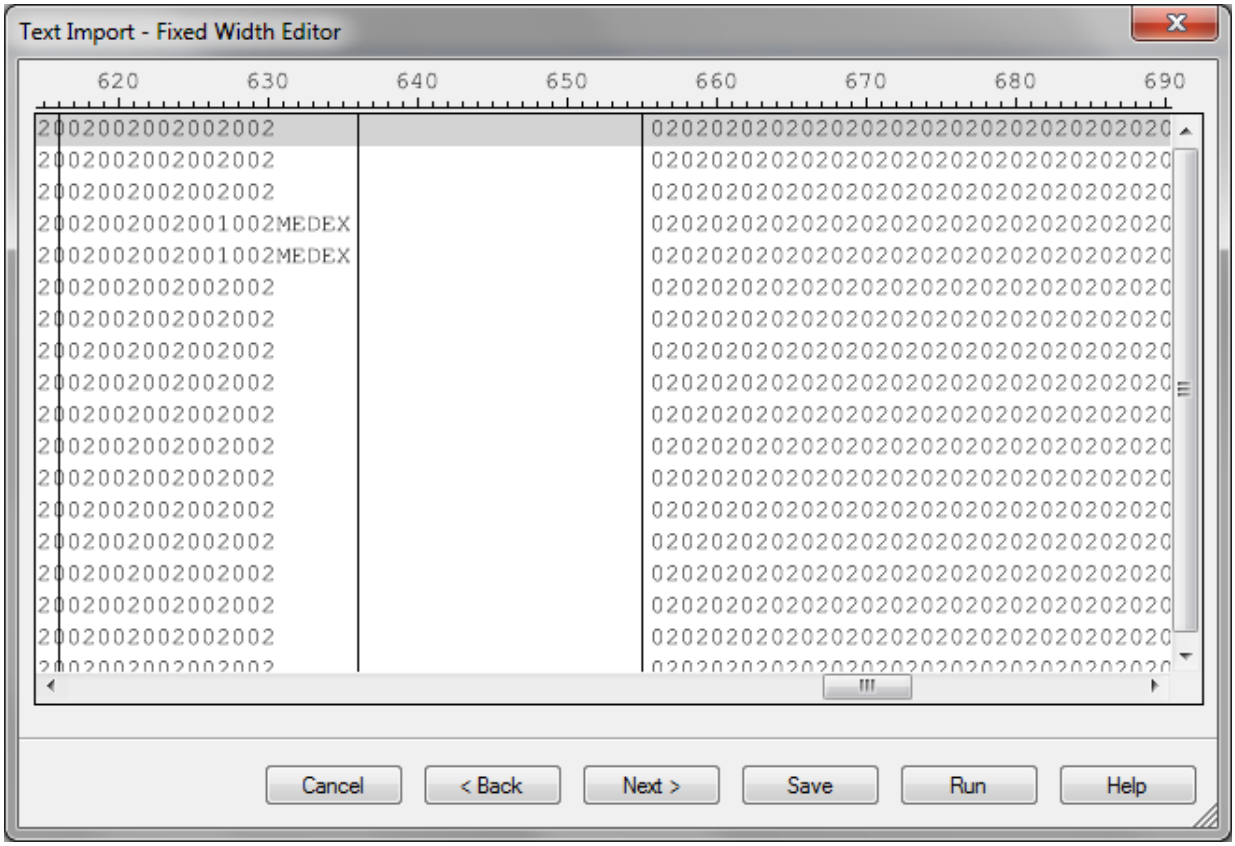
If you choose the *Fixed width* option, the next step provides the way to specify the column boundaries in the Fixed Width Editor.

Fixed Width Editor

If you are importing a fixed-width text file, the Fixed Width Editor page of the File Import wizard helps you specify column boundaries for the column layout.

To insert a vertical line representing the start point (the left boundary) of a column, click slightly to the left of the first character in the column. A vertical separator line will appear as a result. The ruler on top of the preview window helps to place the lines at the necessary intervals.

To remove a column boundary, click slightly to the left of the line you want to remove.



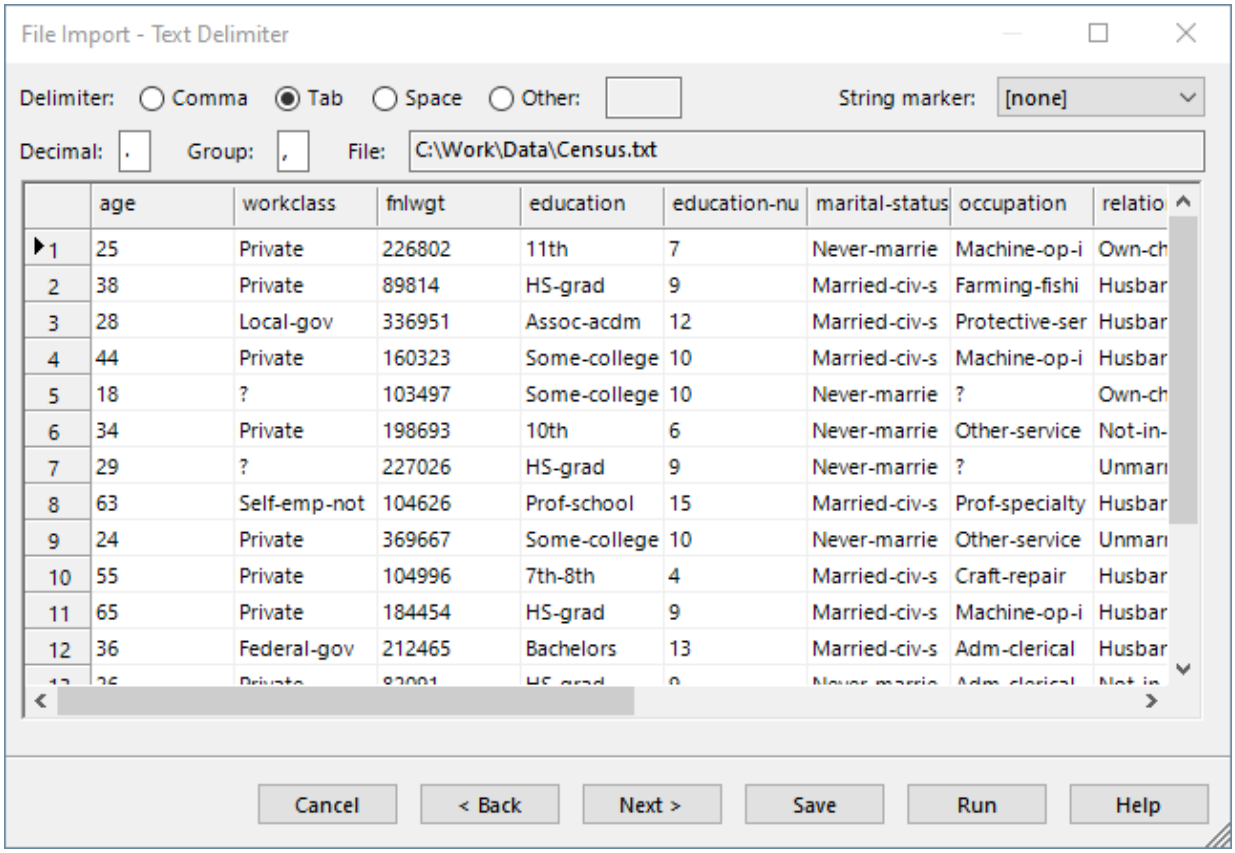
When finished specifying the column boundaries, click **Next** to proceed to the Field Selector step.

Text Delimiter

At this step, the File Import wizard tries to automatically detect the column delimiter in your source text file.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- If the delimiter chosen by the wizard is correct, you will see your data properly separated into columns. Otherwise, choose the correct delimiter or select the ***Other*** option and enter the character representing the delimiter.
- Specify the string marker if necessary.
- Specify the decimal separator and the digit grouping symbol. These options are useful when importing numeric data from a country whose numeric format differs from that in the current regioinal settings.



	age	workclass	fnlwgt	education	education-nu	marital-status	occupation	relatio
1	25	Private	226802	11th	7	Never-marrie	Machine-op-i	Own-ch
2	38	Private	89814	HS-grad	9	Married-civ-s	Farming-fishi	Husbar
3	28	Local-gov	336951	Assoc-acdm	12	Married-civ-s	Protective-ser	Husbar
4	44	Private	160323	Some-college	10	Married-civ-s	Machine-op-i	Husbar
5	18	?	103497	Some-college	10	Never-marrie	?	Own-ch
6	34	Private	198693	10th	6	Never-marrie	Other-service	Not-in-
7	29	?	227026	HS-grad	9	Never-marrie	?	Unmari
8	63	Self-emp-not	104626	Prof-school	15	Married-civ-s	Prof-specialty	Husbar
9	24	Private	369667	Some-college	10	Never-marrie	Other-service	Unmari
10	55	Private	104996	7th-8th	4	Married-civ-s	Craft-repair	Husbar
11	65	Private	184454	HS-grad	9	Married-civ-s	Machine-op-i	Husbar
12	36	Federal-gov	212465	Bachelors	13	Married-civ-s	Adm-clerical	Husbar

Both **Next** and **Run** buttons are enabled. At this point you can press **Run** to start the import with default settings, or click **Next** to continue specifying the import options.

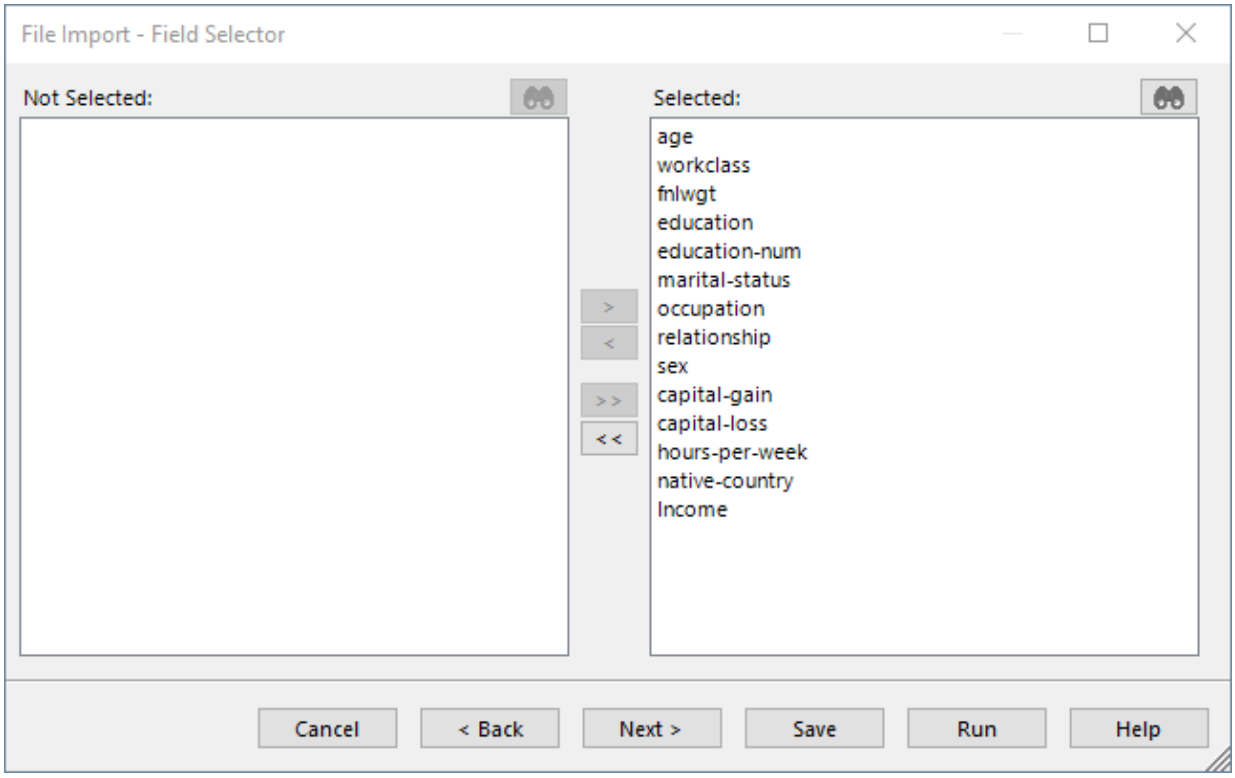
Proceed to the next step to select the fields to import (see Field Selector).

Field Selector

In the Field Selector page of the wizard, choose the fields to import from your data source. By default, all fields are selected. The field names listed on the right will be imported. Field names listed on the left will be excluded from the import.

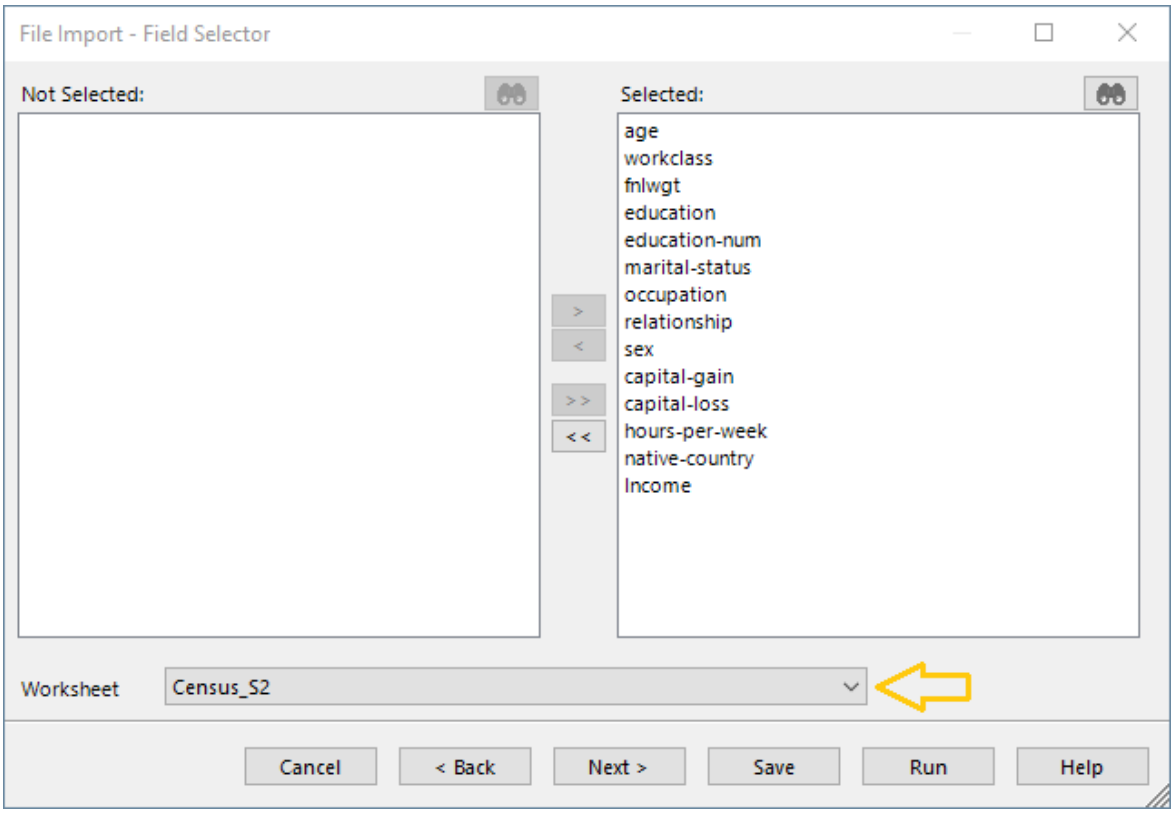
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To move a field from the include list to the exclude list, highlight the field name(s) you want to exclude and click the left arrow button **<**. To include a field, click the left arrow button **>**. You can also move field names from one list to the other by double-clicking on them.



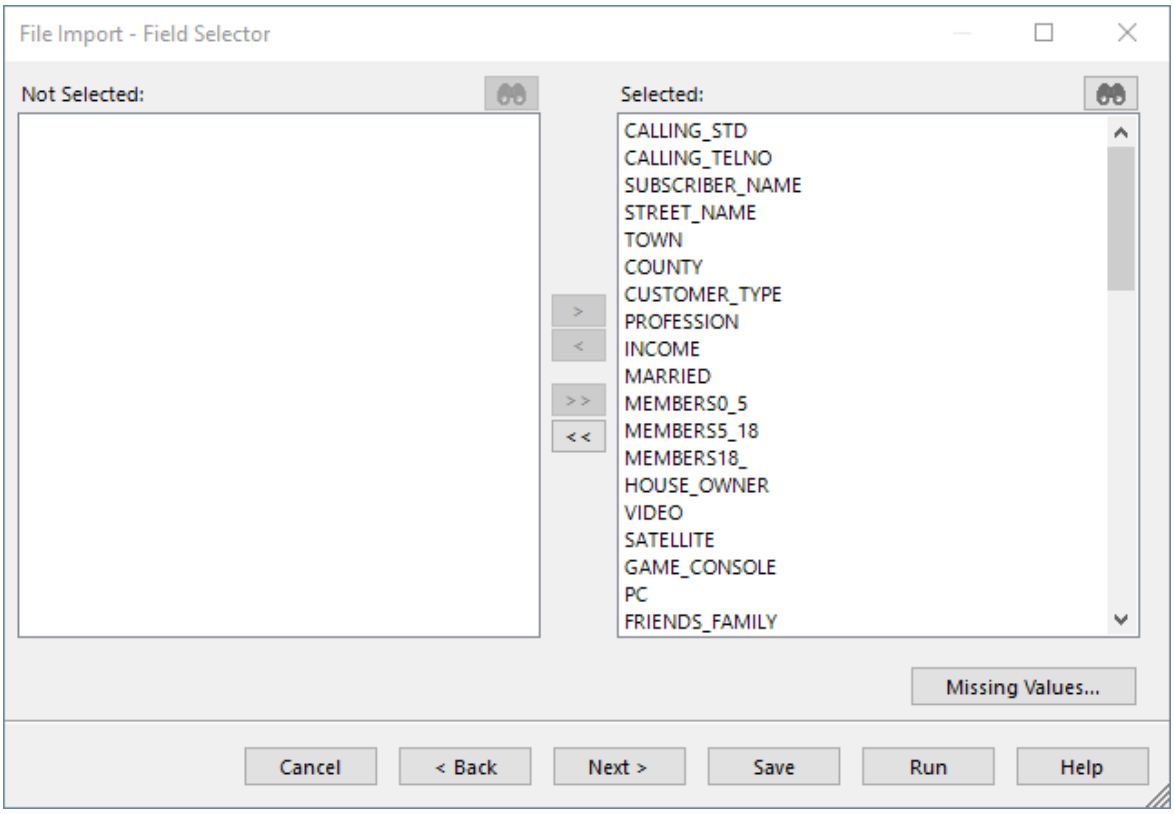
If you are importing data from an Excel spreadsheet with multiple sheets, select the desired worksheet from the drop-down list at the bottom of the wizard page.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



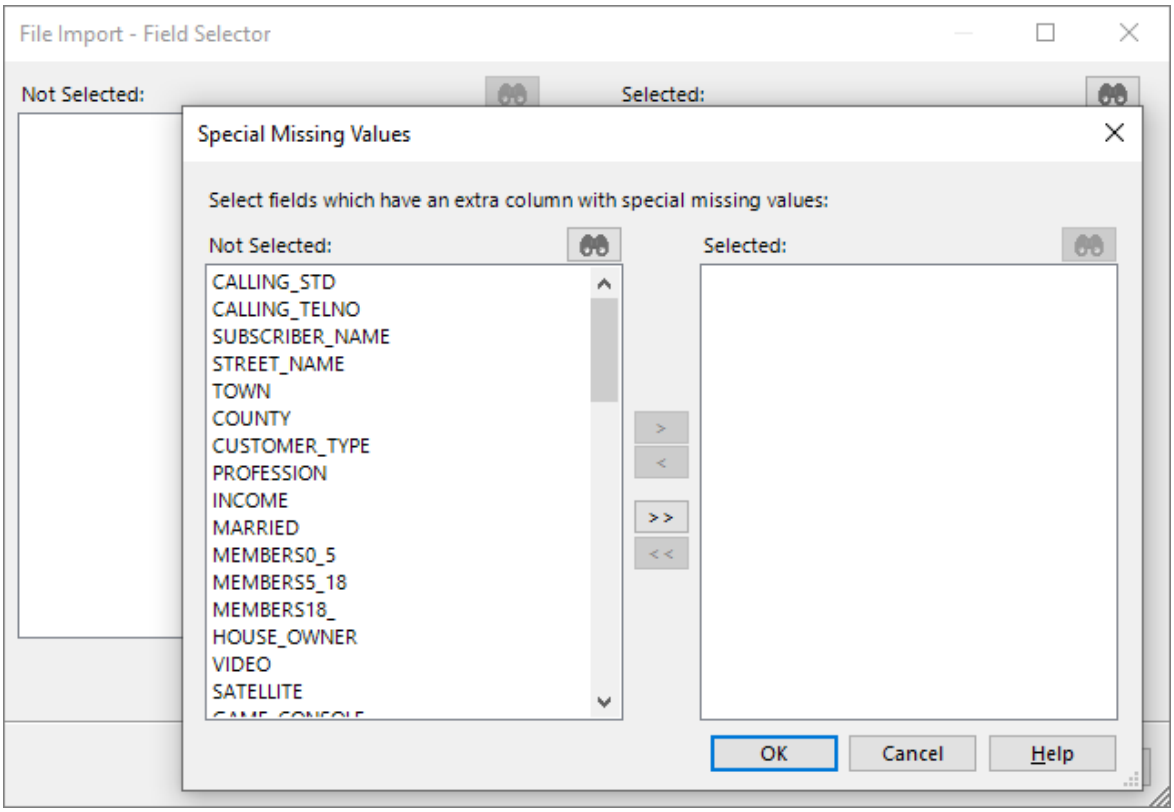
If you are importing data from a SAS System file, you have an option of defining special missing values by clicking the **Missing Values** button at the bottom right corner.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Select the fields that have an extra column with special missing values:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Next, Save, and **Run** buttons are enabled. At this point you can press **Save** or **Run** to start the import with default settings, or click **Next** to continue specifying the import options.

Proceed to the next step to change the field types or define a sample to import (see Field Transformer).

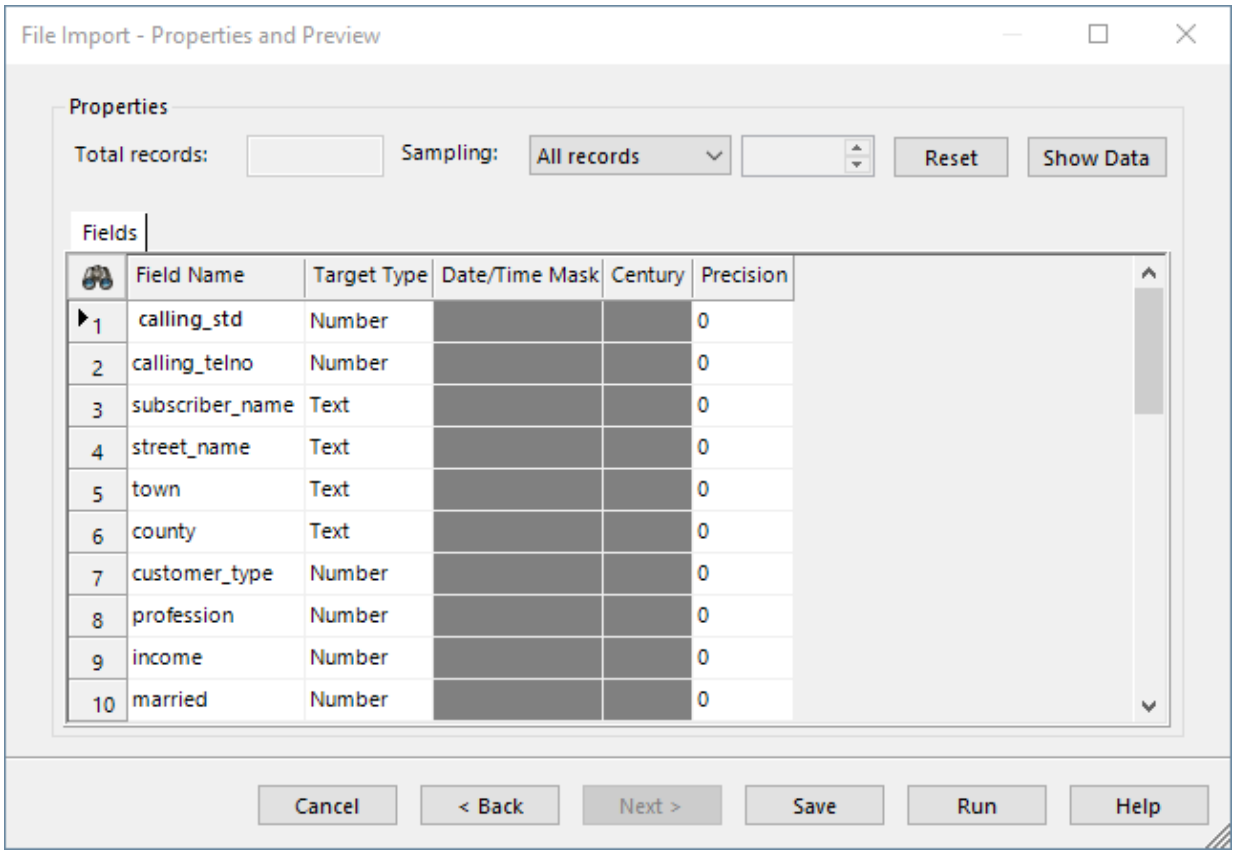
Properties and Preview

The **Properties and Preview** page of the import wizards allows you to:

- modify the field type and specify precision for numeric fields
- specify the sample size - the number or percentage of records to import
- specify a date mask for date, time, and timestamp fields
- preview the data as it will appear in the imported dataset

The Properties and Preview page is the same for all source data formats.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

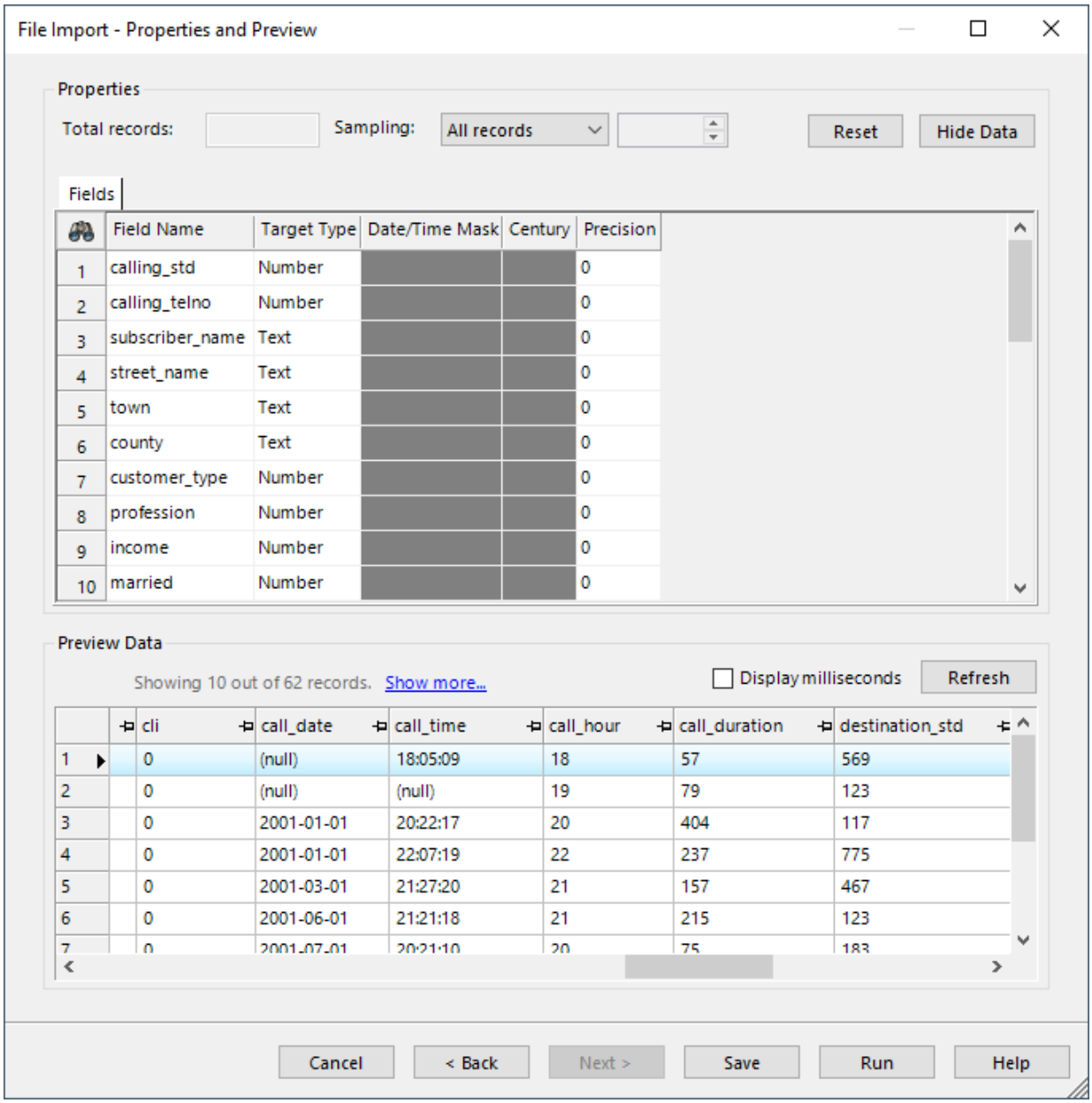


The **Total records** field shows the total number of records in the source file for all source types except Text files and database tables (in ODBC Import).

The **Reset** button resets all the settings in this page to the default values.

The **Show Data** button allows you to preview the data as it will appear in the imported dataset. This function is especially useful when importing data from text files to make sure the date/time fields are correctly imported as Date/Time/Timestamp, with correct date/time masks. See the subsection **Specifying Date/Time Masks** further in this topic.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Sampling

The default is set to import all records from the source file or table. Choose a sampling option is you only need to import a sample of the data.

All Records	Imports all records in the source file or table.
Number records	Imports the specified number of records as a **random sample** of the source data.
Percentage	Imports the specified percentage of records of the source data as a **random sample**.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Top N record	Imports only the first N records of the source data, where N is the number specified by the user.

Field Editor

Field Name	The name of the field as it will appear in the imported dataset. This column is editable. You can rename any field if desired.
Target Type	The data type of the field. Admissible types are *Text, Integer, Number, Boolean, Date, Time,* and *Timestamp*. To change the data type for a field, click in the Target Type column and select the desired type from the drop-down list. To set the type of multiple fields at a time, drag the mouse in the first (index) column of the grid or click in the first column while holding down the SHIFT or CTRL key. Then click in the Target Type column for any of the highlighted fields and select the desired type from the list. This will set the same type for all of the highlighted fields. The **Preview** button allows you to preview the data as it will appear in the imported dataset. Note that if you change the type for a field, and the new type is incompatible with the raw data in this field (for instance, if you set the type to "Number" for a text field), the Preview window will show ***null*** for the values in this field, and this is how it will be imported if you proceed with the import. If you change the types in this page, especially for multiple fields at a time, make sure you specify the types compatible with the source data.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Precision

Depending on the data type, *precision* affects the imported field in different ways.

Numeric (Integer, Number):

- When the ***precision*** is a **positive** number, the values in the selected field will be rounded to the specified number of decimal places. For example, if you want only 2 decimal places for the values in the selected field, enter 2 in the ***precision*** column. All the values in this field will be rounded to 2 decimal places when imported. For example, 34.42852 will be imported as 34.43. To round the values of a Number field to the nearest Integer, just change its **Target Type** from Number to Integer. If the field type is Integer, positive precision has no effect.
- When the ***precision*** is a **negative** number, the values in the selected field will be rounded to the left of the decimal point. For example, to round all values in a numeric field to the nearest hundred, specify the precision value **-2**. The values in the selected field will be rounded to the nearest hundred when imported. For example, the value 7321 will be imported as 7300, and the value 7368 will be imported as 7400.
- Precision **0** (default) means that every value in the field is imported as is.

Text:

- When the ***precision*** is a **positive** number, only the specified number of characters, starting from the beginning, will be imported for each field. For example, if you want only the first 3 characters of a name to be imported for this field, enter 3 as the precision. For example, the string **'Jack Smith'**, will be imported as **'Jac'**.
- When the ***precision*** is a **negative** number, only the specified number of characters, starting from the end, will be imported for each field. For example, you only want the last 3 characters of a name to be imported for this field. Enter -3 as the ***precision***. All the strings in this field will only be the last 3 letters. A field with the string **'Jack Smith'**, will be imported as **'ith'**.
- Precision **0** (default) means that that every value in the field is imported as is.

Date/Time Mask

Applicable only when Target Type is *Date, Time,* or *Timestamp*. In this case, the date/time mask must be defined. The Date/Time mask definition dialog opens if you change the Target Type of a field to Date, Time, or Timestamp. If the field already has one of these types, the mask can be modified by double-clicking in the Date/Time Mask column.

Date mask must be specified with capital letters D(day), M(month), Y(year), for example, `YYYY-MM-DD`, `MM/DD/YYYY`, etc.

Time mask must be specified with lowercase letters `h` (hours), `m` (minutes), `s` (seconds), `f` (milliseconds), and `tt` (AM or PM). Examples of time masks:

```
hh:mm:ss
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
hh:mm:ss:fff
hh:mm tt
```

The Date/Time mask definition dialog provides several standard masks and allows specifying a custom mask.

To make sure that you specify the correct date/time mask, check the source values in the Text File Preview or Text Delimiter pages of the wizard. Click the **Show Data** button in the Properties and Preview page. For details, see the subsection **Specifying Date/Time Masks** below.

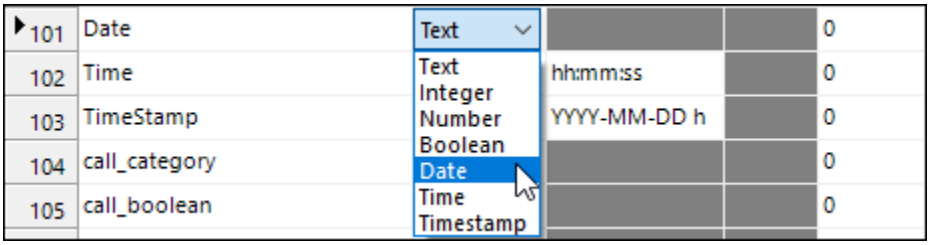
Century Applicable only when Target Type is Date or Timestamp. This is entered through the Date/Time Mask definition dialog. If the date/time mask is already defined, double-click in the Century column to open the Date/Time Mask dialog.

Specifying Date/Time Masks

Metadata in SAS System, Excel, SPSS, and R files and ODBC data sources contains information about field types. The import function for these data types reads the metadata from the source and automatically detects data types, then assigns the target field types accordingly. For these sources, you do not have to verify and explicitly set the target type in the import wizard unless you intend to change the types for specific fields.

Text files (*.txt, *.csv) normally do not contain any metadata, so the import wizard guesses the field types based on the first 20 records of the source text file. Date, Time, and Timestamp fields in text files are read as **Text** data type by default. When importing text files, check the target data types in the last page of the File Import wizard. Set the desired types for the fields whose types were not detected automatically, such as Date, Time, and Timestamp fields.

The Date/Time mask definition dialog opens when you change the Target Type of a field to Date, Time, or Timestamp.



For fields already defined as Date/Time/Timestamp, the mask can be modified by double-clicking in the Date/Time Mask column.

The **Mask** field provides standard mask options to be selected from a drop-down list, while also allowing to specify a custom mask. In the same dialog, the **Raw Data Preview** grid shows the original values in the selected field(s) being formatted.

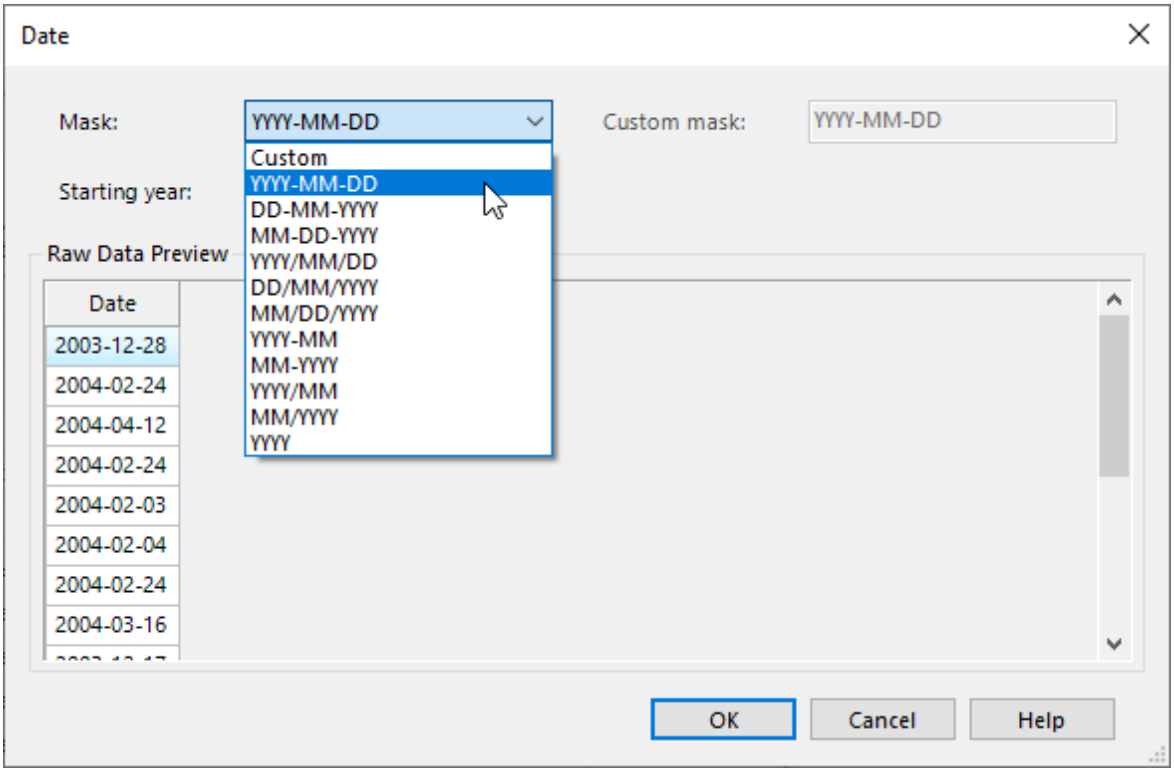
Select the appropriate mask that fits the data in the selected field(s) displayed in the Raw Data Preview frame. If the dropdown does not contain the correct mask, select "*Custom*" and enter the correct format in the **Custom mask** field.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Specifying the correct mask ensures that the data is **read** correctly. Once the values are read, the way they are **displayed** may also depend on the regional settings (number, time, and date formats) on the host operating system.

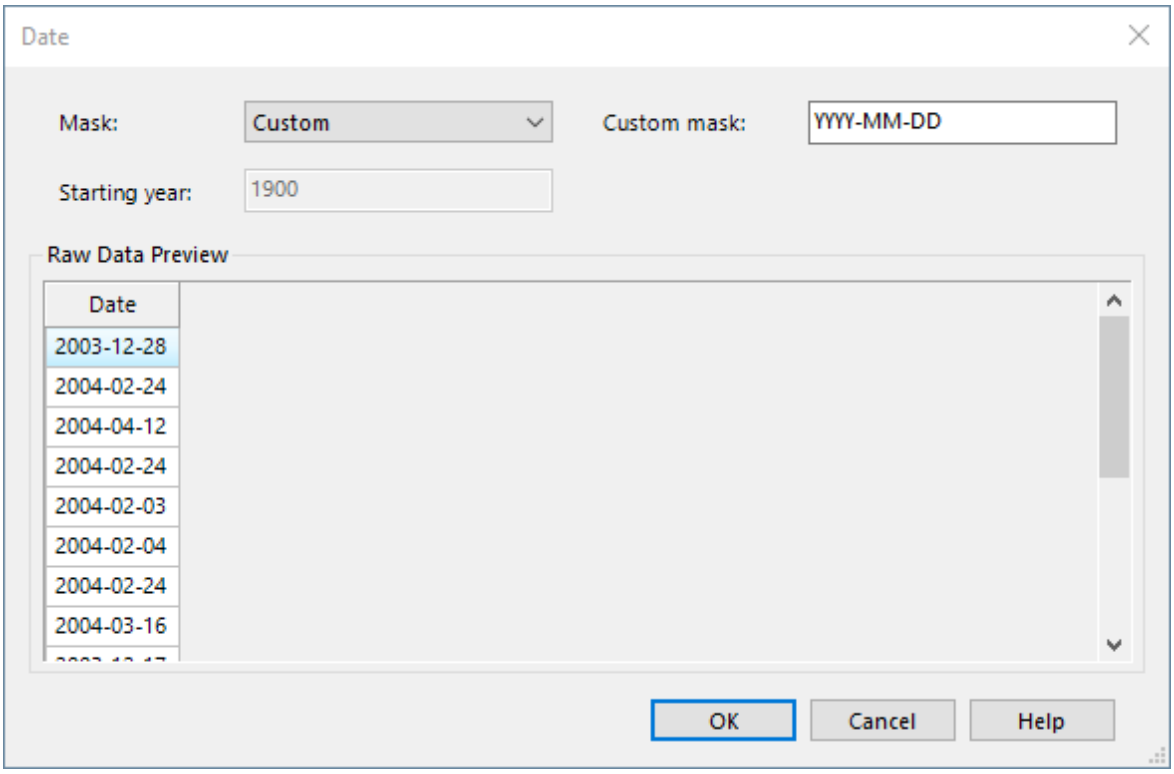
Specifying the wrong mask results in the values being imported as *null* (missing values).

Example 1: Specifying a mask for a ***Date*** field



If the **Custom** option is selected in the Mask drop-down list, the **Custom mask** field becomes enabled for text input:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



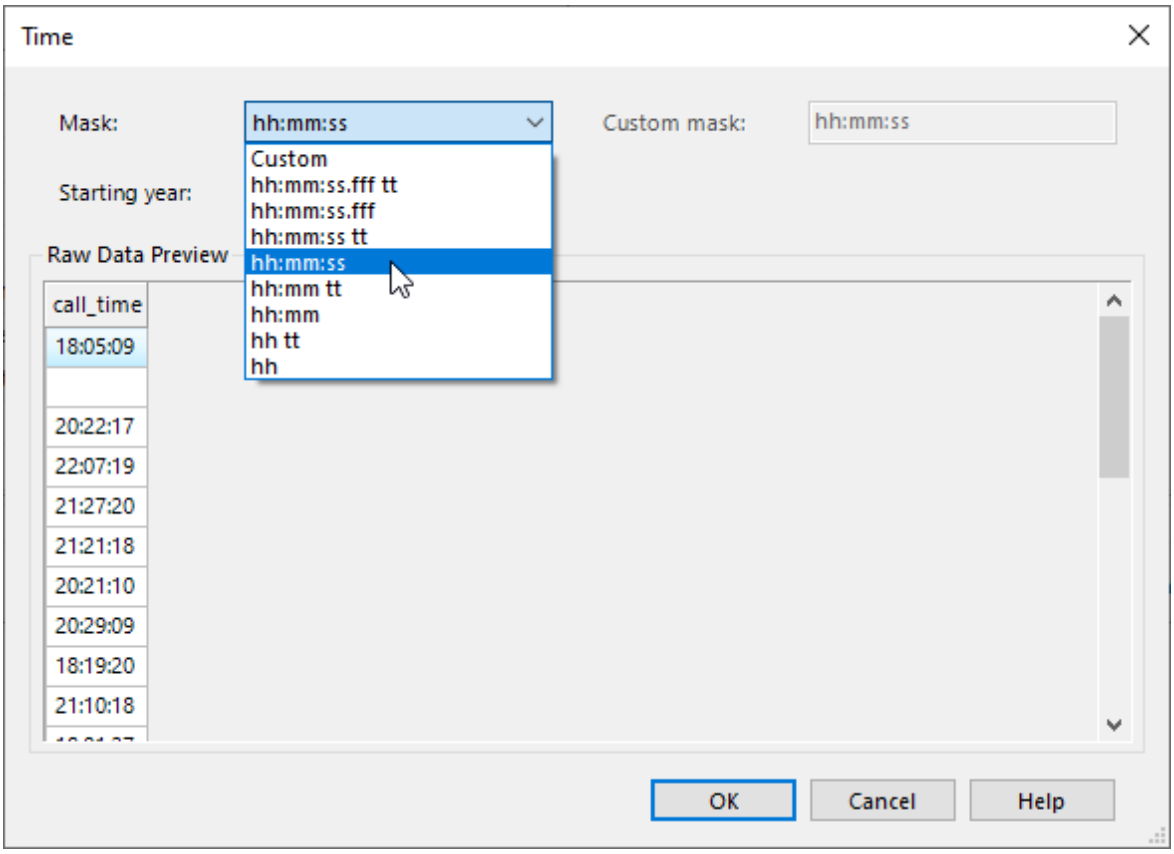
Example 2: Specifying a mask for a ***Time*** field.

The standard time masks available for selection include those with milliseconds and AM/PM designations.

Examples of time data and the corresponding masks:

Time data example	*Mask*
05:15 PM	`hh:mm tt`
13:47:33	`hh:mm:ss`
17:15:48.641	`hh:mm:ss:fff`
05:15:48.641 PM	`hh:mm:ss:fff tt`

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Example 3: Specifying a mask for a ***Timestamp*** field.

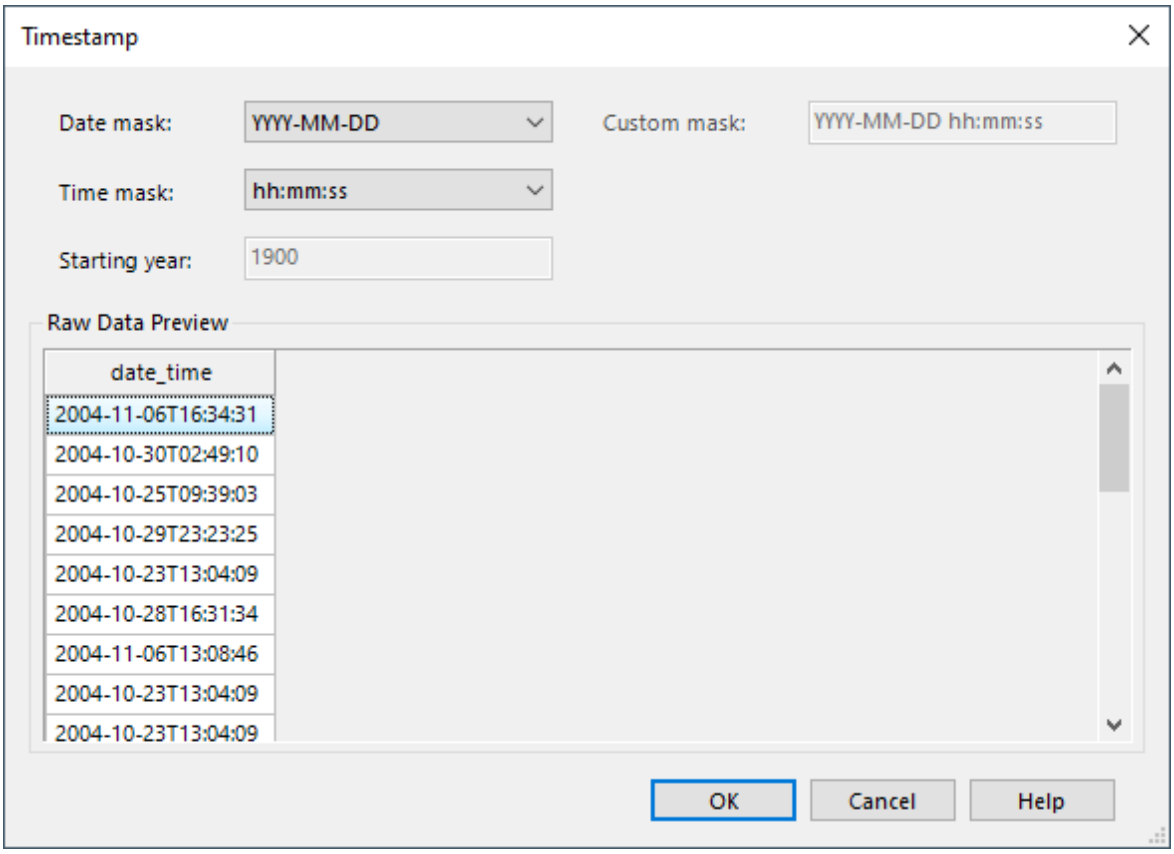
For Timestamp fields there are two parts of the mask to select – one for the date part and one for the time part. The date and time parts in the source data are assumed to be separated by a single space or the letter *T*.

The Time masks available for selection include those with milliseconds and AM/PM designations.

Examples of timestamp data and the corresponding masks:

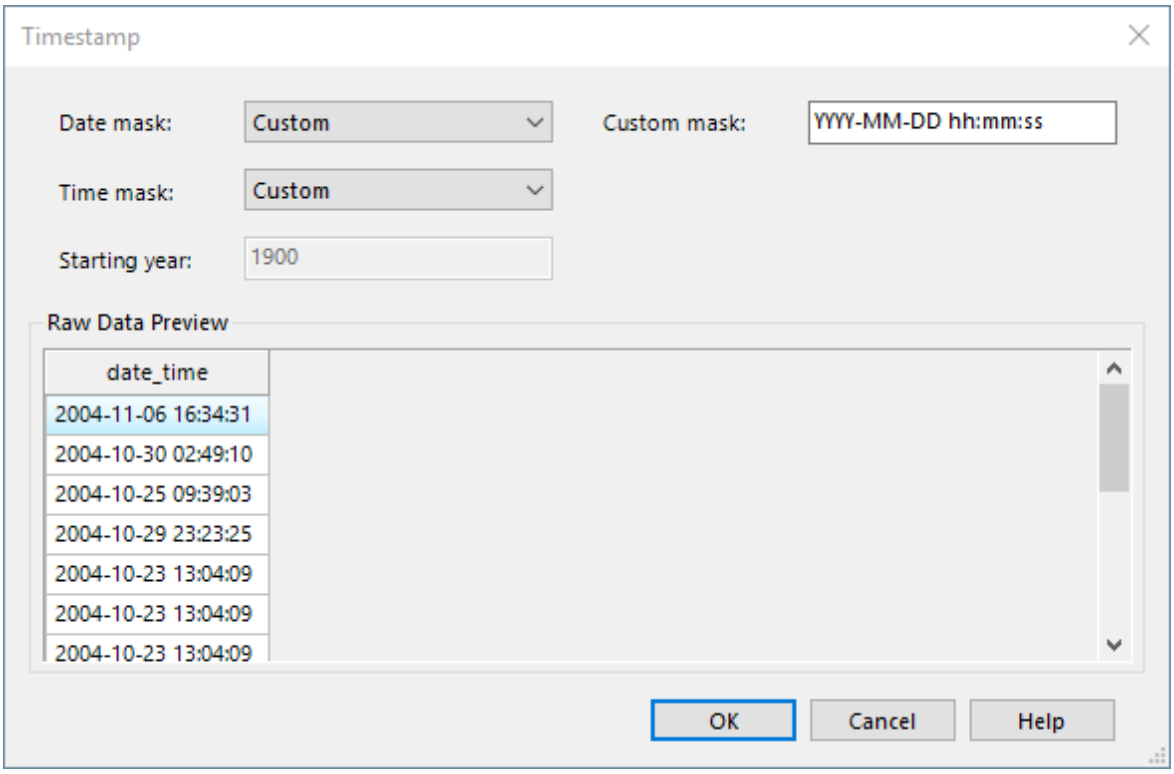
Timestamp data example	*Date Mask*	*Time Mask*
2012-10-01 05:15 PM	`YYYY-MM-DD`	`hh:mm tt`
2012-10-01T05:15 PM	`YYYY-MM-DD`	`hh:mm tt`
03/24/2019 13:47:33	`MM/DD/YYYY`	`hh:mm:ss`
1984/03/14 17:49:11.641	`YYYY/MM/DD`	`hh:mm:ss:fff`
27/10/2013 05:15:48.641 PM	`DD/MM/YYYY`	`hh:mm:ss:fff tt`

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If the **Custom** option is selected in the Mask drop-down list, the **Custom mask** field becomes enabled for text input. By default, the Custom mask box concatenates the two previously entered masks, separating them with a single space.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The mask definition will be applied to the selected field(s). If multiple fields were selected in the Properties and Preview page, then the Raw Data Preview grid will contain all of the selected fields, and all of them will be assigned the same mask once you click **OK**.

Click **Show Data** in the Properties and Preview page to verify that the values are displayed correctly in the fields whose Date, Time, or Timestamp masks you specified.

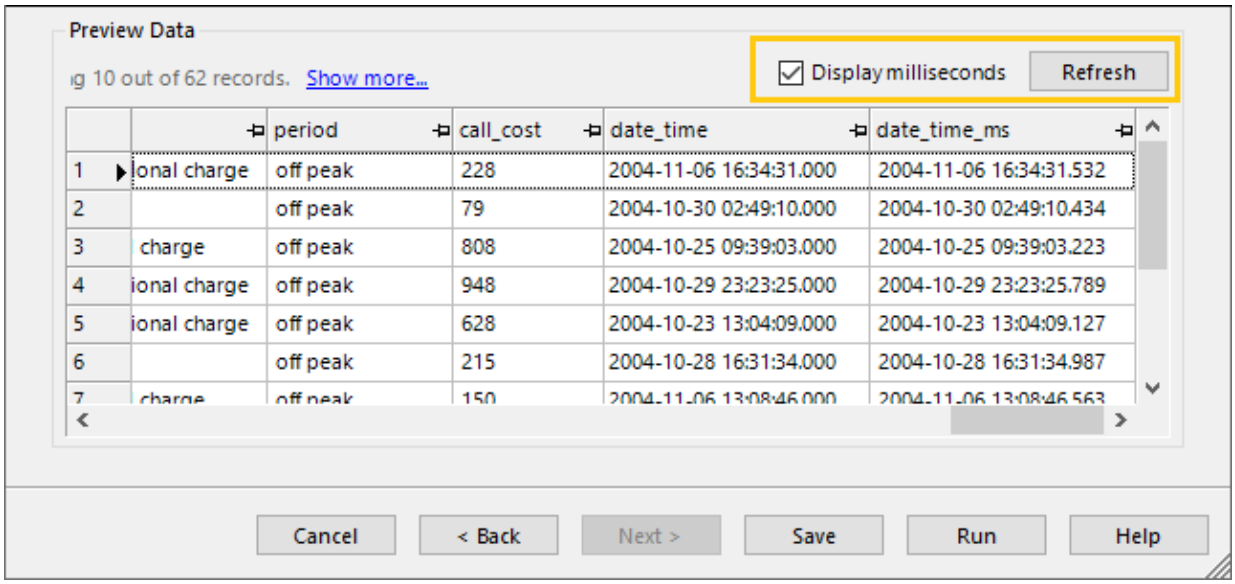
Preview Data

Click **Show Data** to preview the data as it will appear in the imported dataset. This opens the **Preview Data** frame at the bottom of the window. Click **Refresh** to show or update the data in the preview frame. By default, the top 10 records of the transformed dataset are displayed. Click "*Show more...*" or just scroll down in the Preview Data frame to view more records. Click **Hide Data** to hide the Preview Data frame.

The way the values are displayed may depend on the regional settings (number, time, and date formats) on the host operating system.

If your source data contains Time or Timestamp values with milliseconds, select the option ***Display milliseconds*** and click **Refresh**. The milliseconds part will be displayed for all Time and Timestamp fields. An example is shown below.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The option ***Display milliseconds*** explicitly applies the custom format strings defined by the Date and Time masks you specified. If the option ***Display milliseconds*** is **not** selected, Time and Timestamp values are displayed in the Windows' standard format for time with milliseconds, exactly as the values are stored (for example, 2008-01-01T00:30:45.1250000 - note the trailing zeroes).

Specifying correct masks for Date, Time, and Timestamp fields is necessary for the data to be read correctly and imported successfully. Note that the way the values are **displayed** in the preview and in the resulting dataset may be different from the masks you specified depending on the regional settings (number, time, and date formats) on the host operating system. For example, if dates in your source file are in the format YYYY-MM-DD, you must specify the corresponding mask "YYYY-MM-DD" for the data to be read correctly. However, if the Date format in your Regional Settings is DD/MM/YYYY, then the values will be displayed in the format DD/MM/YYYY in the preview frame and in the resulting dataset.

In the imported data, Time and Timestamp values with milliseconds can be displayed only in the views that show raw data - the Data tab of datasets, the Node Data tab of Decision Trees and Strategy Trees, and Preview Data frames in the nodes such as Variable Transformations.

Folder Import

For general information on accessing data for analysis, see the topic Accessing Data Sources.

The **Folder Import** node allows you to import multiple files of the same format and structure in one take - either into a single dataset (appending them to each other), or into separate datasets. The Folder Import node supports the same source file formats as the File Import node, and its wizard is similar, except that it allows you to manage multiple files in parallel and optionally append them at the end.

The folder with your source files can be on the local file system or on a network file system, as long as it is accessible from your computer or the KS Server host (if you are using the client/server configuration).

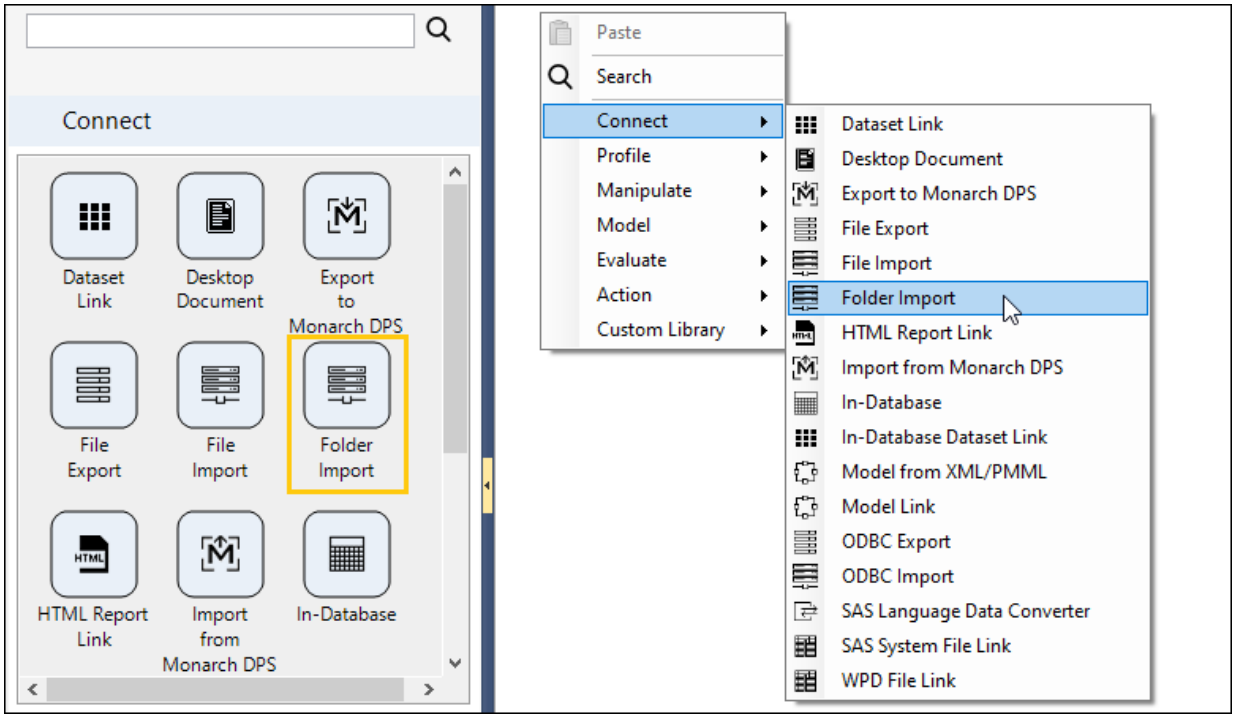
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To import data from multiple files contained in the same folder, use any of the following ways:

- Drag the **Folder Import** node from the **Connect** palette onto the workflow canvas. Then double-click the node to open the wizard.

OR:

- Right-click anywhere on the Workflow canvas background. From the context menu, select **Connect --> Folder Import**. A new **Folder Import** node will be created, and the wizard will open.

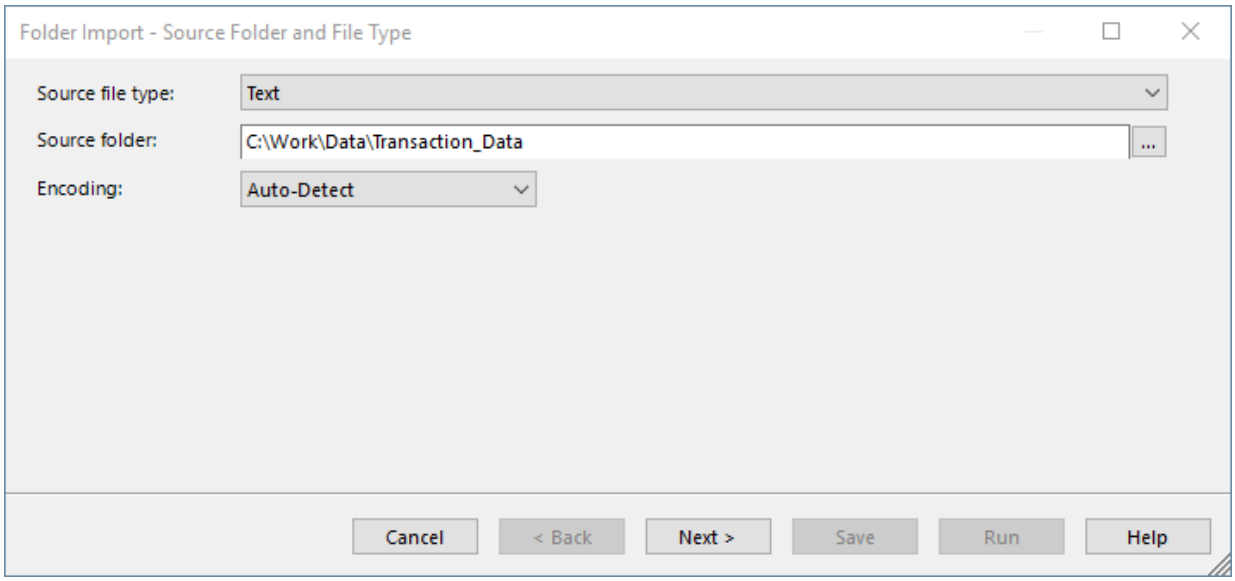


If the Folder Import node is already on the canvas, double-click it or right-click on it and select **Modify** from the context menu to open the wizard.

In the first page of the wizard:

- Select the desired source format from the *Source file type* drop-down list. The wizard will only handle source files in the selected format.
- Specify the path to the folder that contains your source files to import. Enter the path or use the browse button on the right to browse for the folder.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

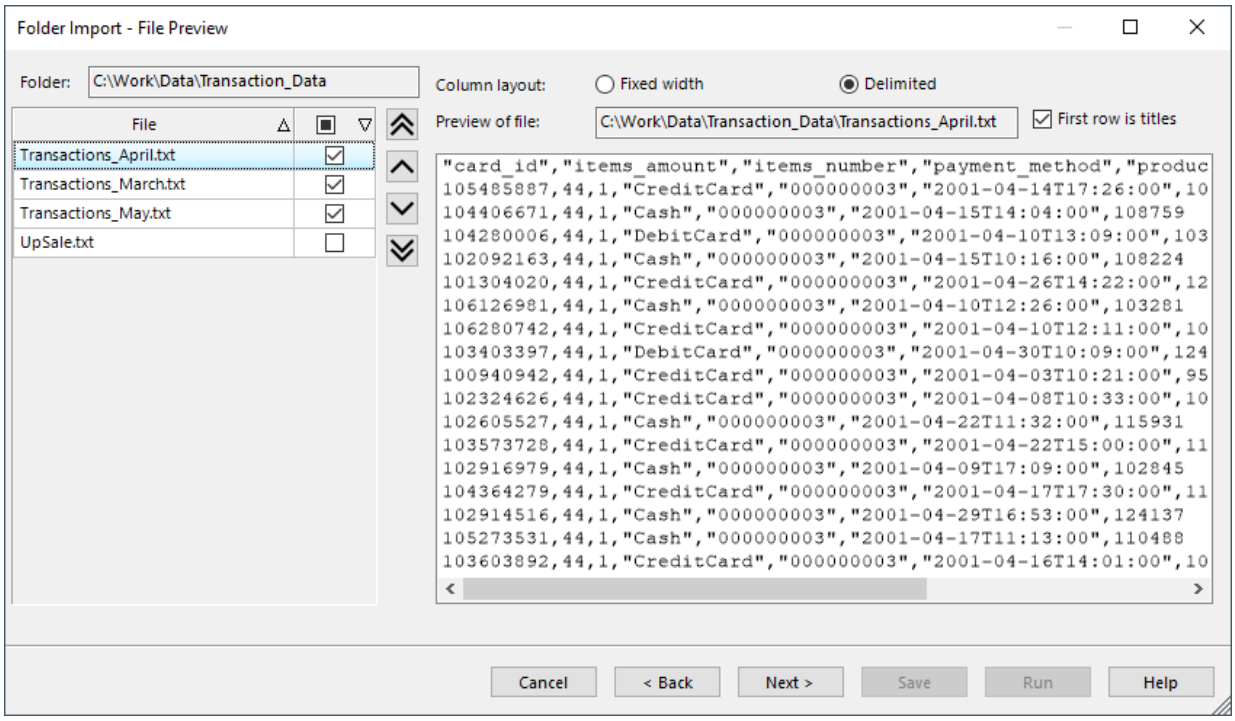


If you select the **Excel** file format, this wizard page will show additional options that determine how to treat the first row in all source spreadsheets to be imported. The option *First row is titles* is the default. Choose the option *First row is data* if you want to force the wizard to treat all rows as data. In this case the fields in the resulting dataset will be named automatically as C1, C2, C3, ... and so on, but you will be able to rename them at the last step.

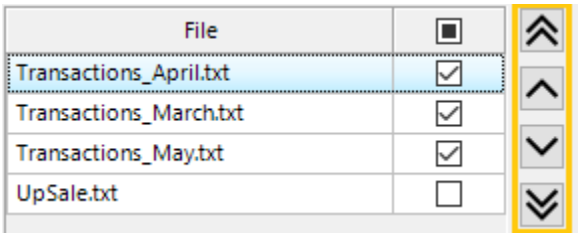
For information on encodings, see the File Import topic.

If you selected the **Text** file format, the next page is File Preview. It is the same as the Text File Preview page in the File Import wizard, but it allows you to preview all text files in the selected folder. To switch between different files, click on the file names in the list of available files on the left. You can deselect any file if you want to exclude it from the import.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

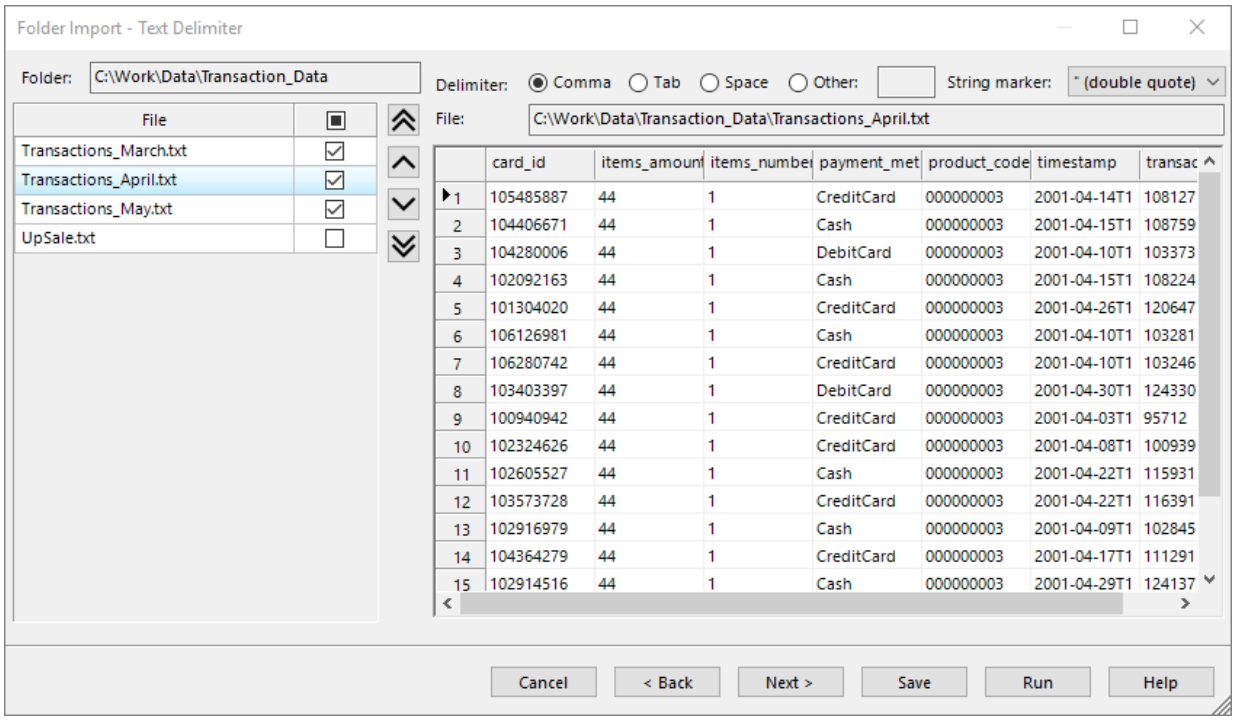


Use the arrow buttons on the right-hand side of the grid to change the order of viewing the files. This will also determine the order of appending the files if this option is selected at the end.



If you selected the **Text** file format, the next page allows you to specify the delimiter. It is the same as the Text Delimiter page in the File Import wizard, but it allows you manage all the selected files in parallel. To switch between different files, click on their names in the grid on the left. You can deselect any file if you want to exclude it from the import and also change their order using the arrow buttons. See the Text Delimiter topic for details.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



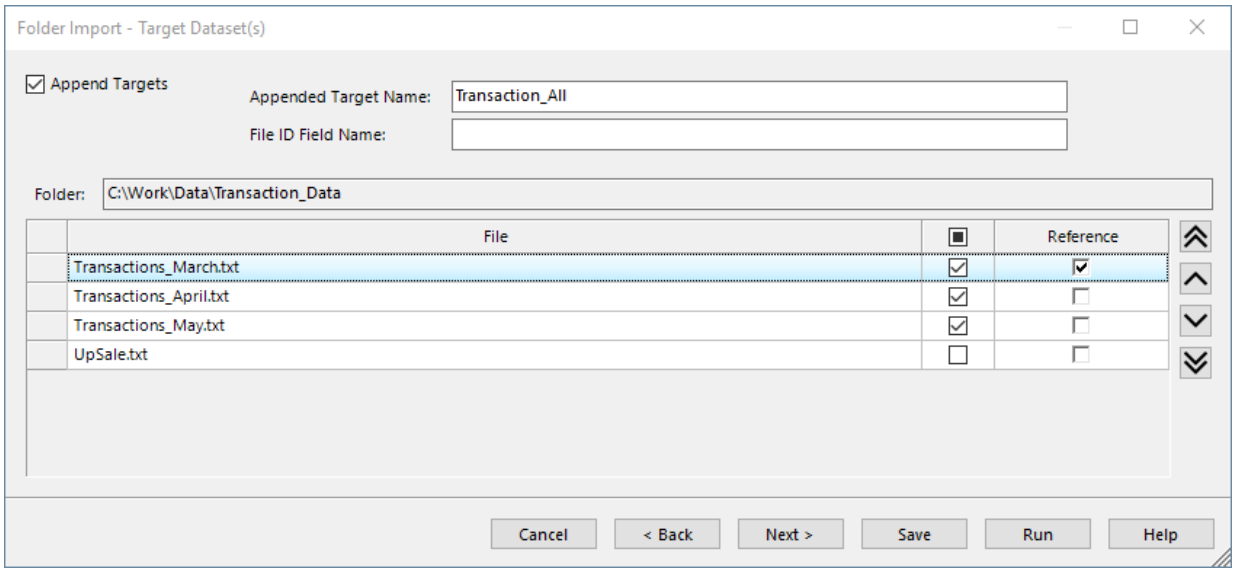
Click **Next**.

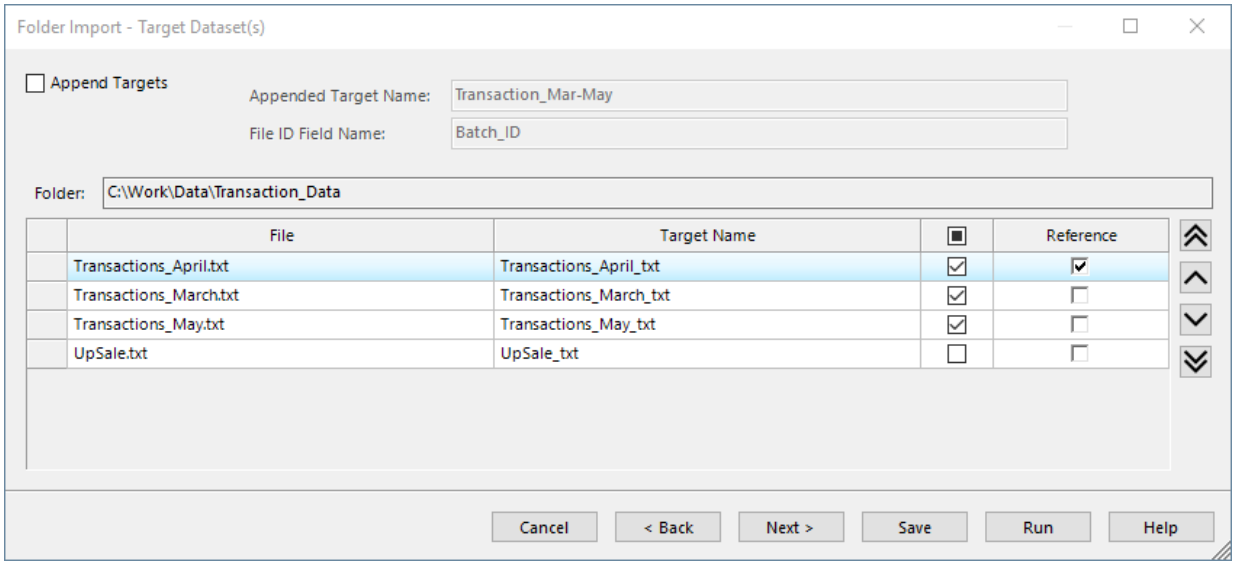
In the Target Dataset(s) page:

Choose whether to append the data from all source files to create a single dataset. In this case specify the name for the target dataset in the *Appended Target Name* field. Optionally, specifying the name for the File ID field to be created.

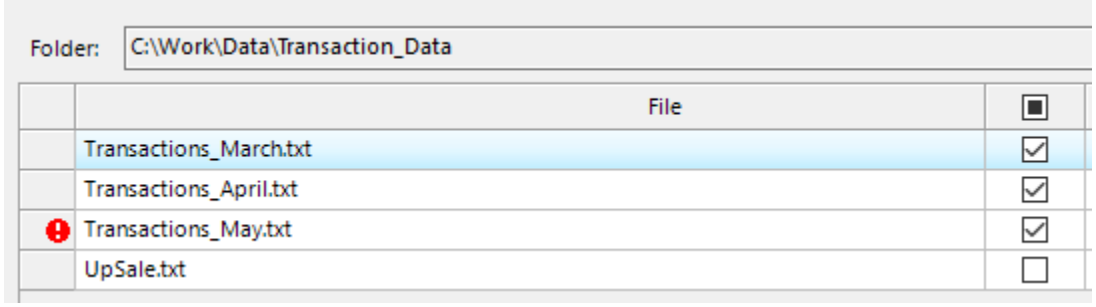
- Select or deselect the source files.
- Choose one file as a ***reference***. The structure of the other selected files must be the same as that of the reference file.
- Specify the order of the files using the arrow buttons on the right-hand side of the grid.
- If you deselect the Append option, the Target Name fields displays the default names for the datasets to be created. Modify the default names if necessary.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

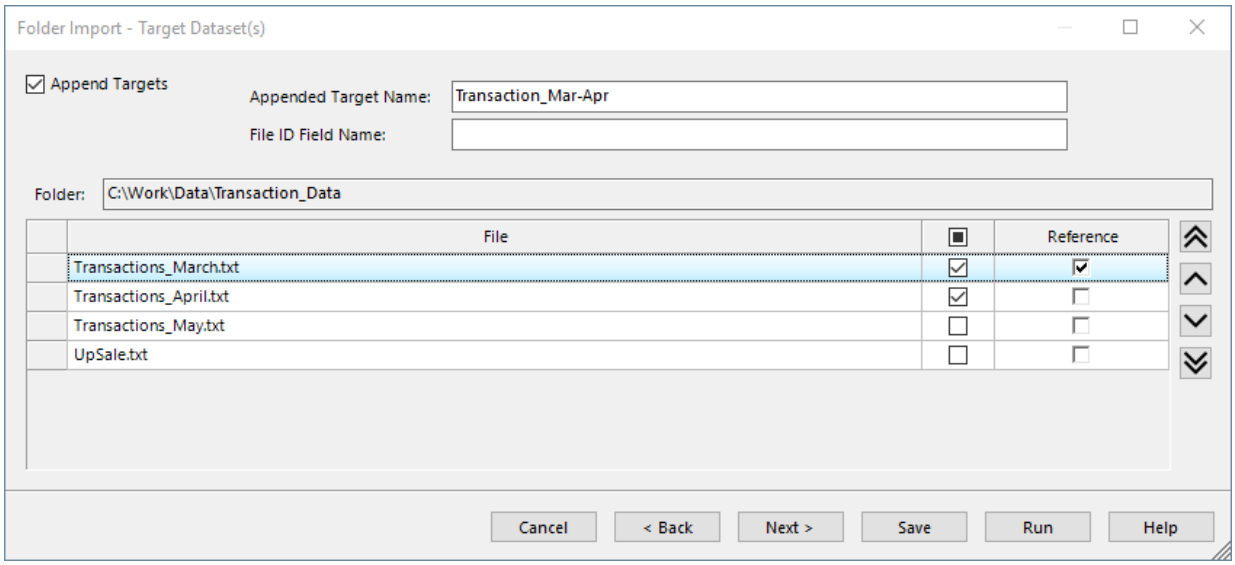




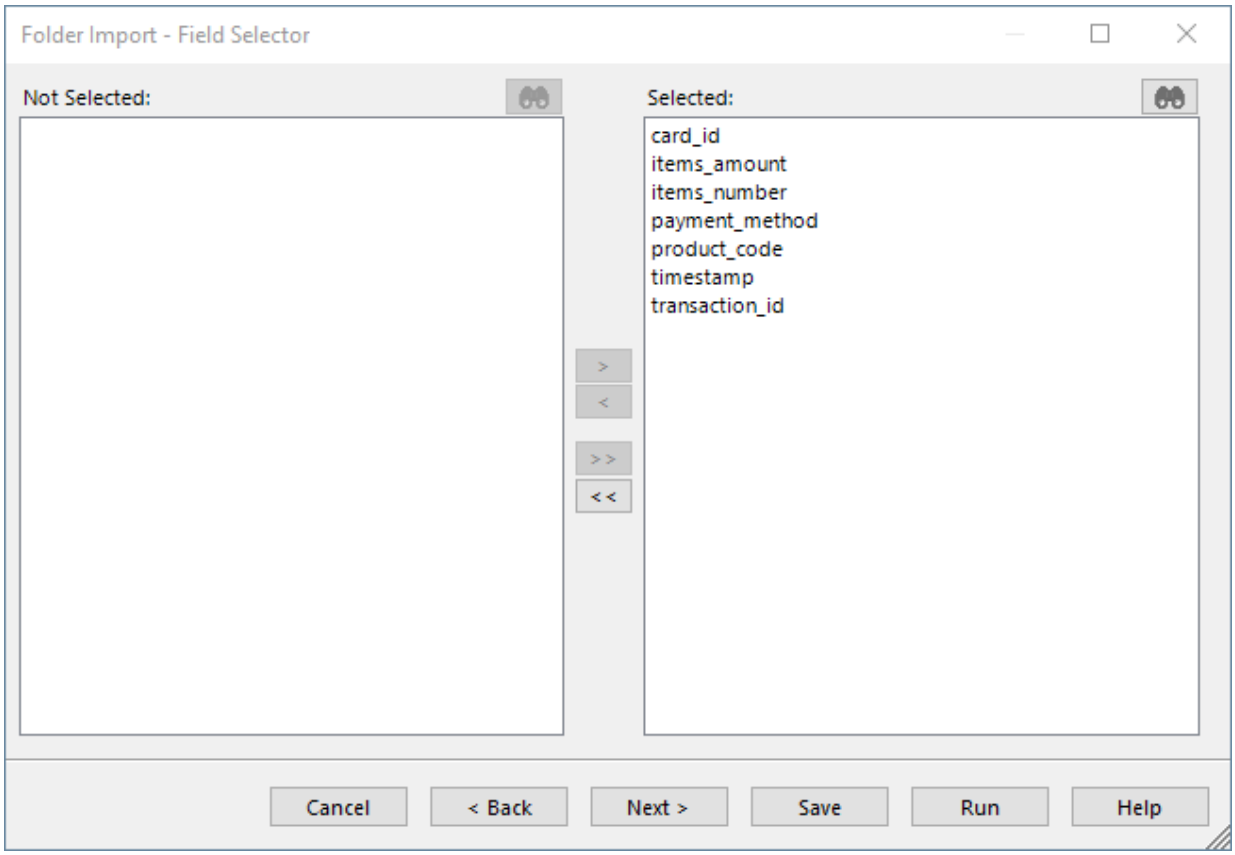
Click **Next**. If the structure of any of the selected files does not match the structure of the reference file (has different field names or field types), then you will get a warning "*Validation failed for some targets*.". Fix the structure of the highlighted source file or exclude it from the import here.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.



After deselecting the files that need not be imported, click **Next** to select the fields to be included in the resulting dataset(s). This wizard page is the same as Field Selector.



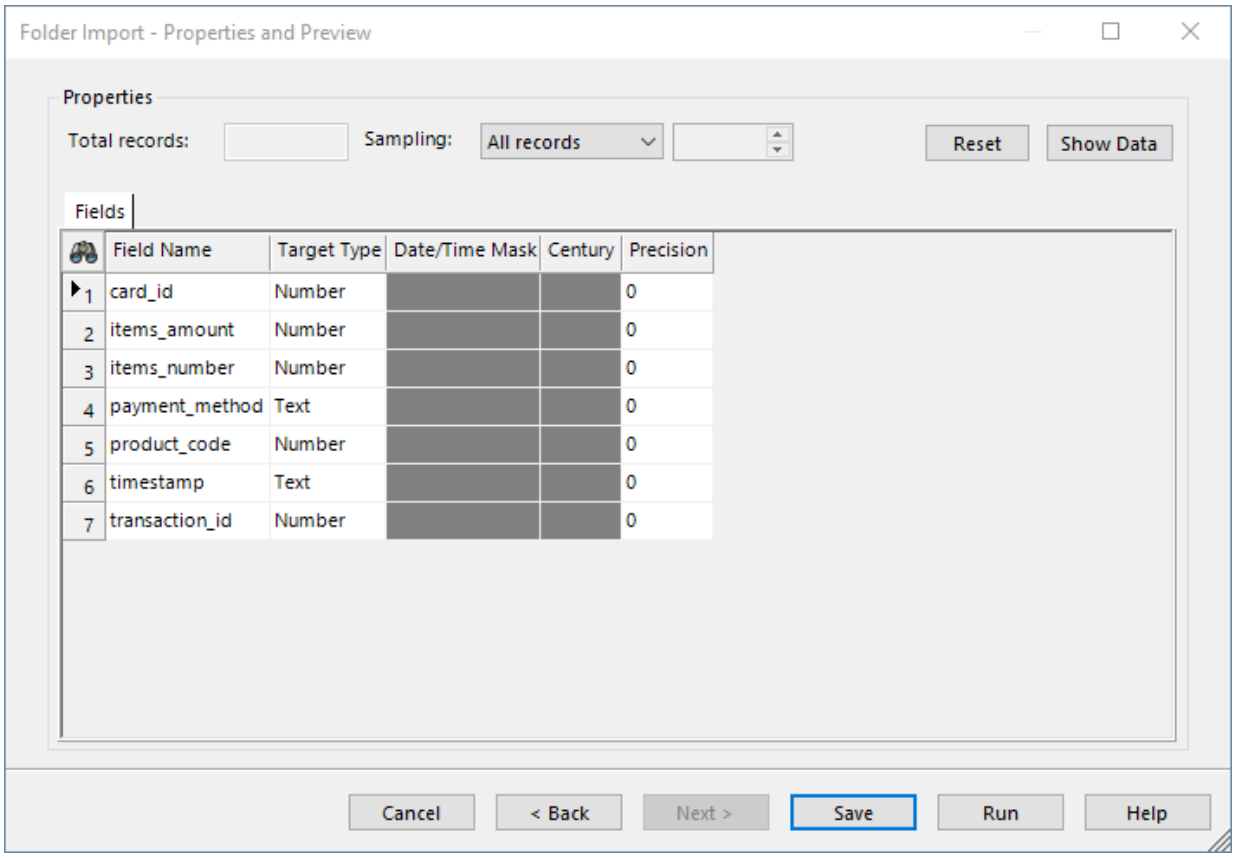
Click **Next**.

This page is the same as the Properties and Preview page of the File Import wizard except for some minor differences noted below. In the Properties and Preview page:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Modify the field type and specify precision if necessary.
- Specify the sample size - the number or percentage of records to import from each source file.
    - Note: If you selected the *Append* option at the previous step, sampling will be performed before appending. For example, if you specify 1000 records for Sampling while importing three files with the *append* option, the resulting dataset will contain 3000 records.
- Specify a date mask for date, time, and timestamp fields (if any)
- Preview the data as it will appear in the imported dataset.
    - Note: The preview frame shows only records from the *reference dataset*.

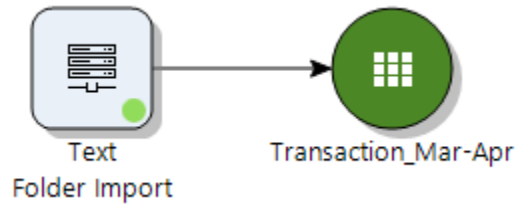
For the detailed description of options in this page, see the Properties and Preview topic.

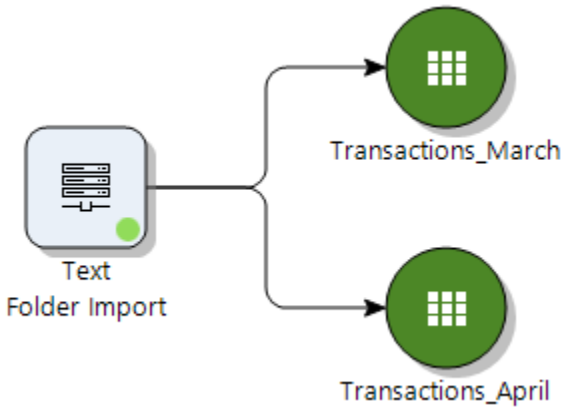


Click **Save** to save the settings or click **Run** to save the settings and run the node.

Upon running the node, double-click the output dataset node(s) to view the results. There will be a single output dataset if you chose to append data, and multiple output datasets otherwise.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.





See also:

Accessing Data Sources.

File Import

Import from ODBC data sources

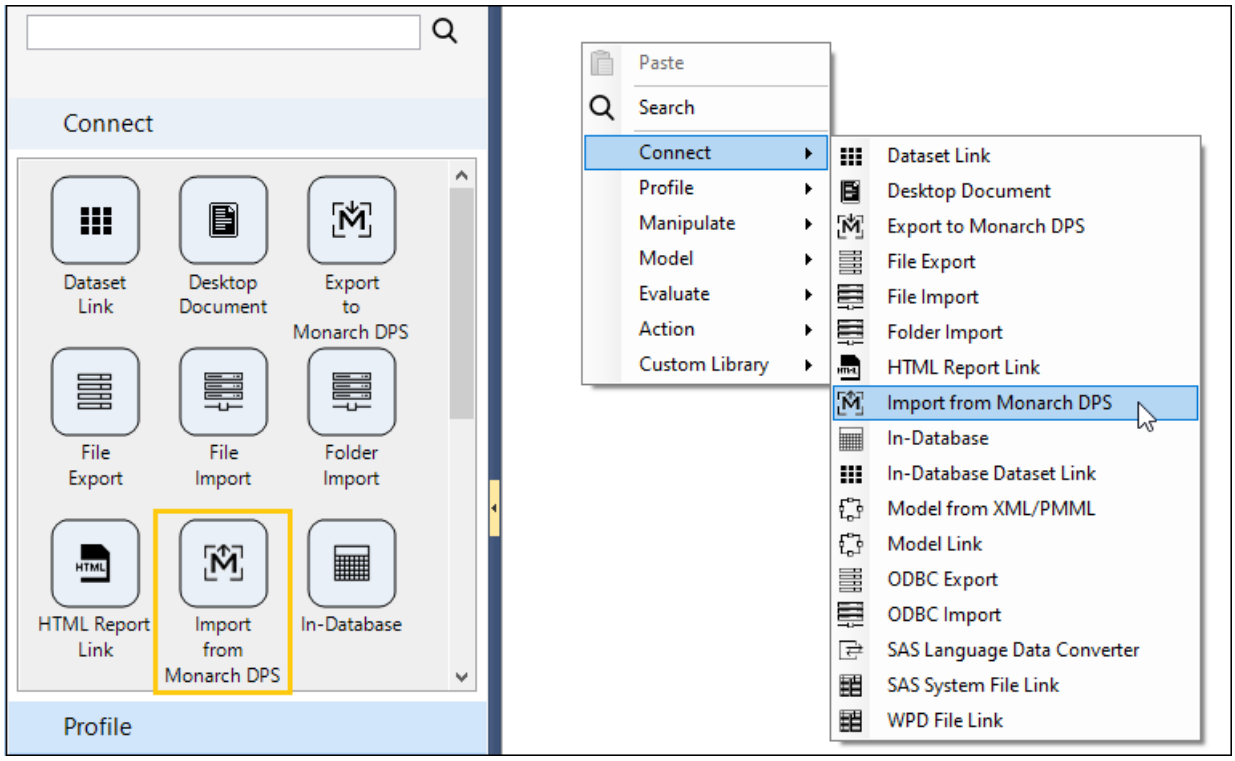
To import data from Monarch Data Prep Studio, use any of the following ways:

- Drag the node **Import from Monarch DPS** from the **Connect** palette onto the workflow canvas. Then double-click the new node to open the wizard.

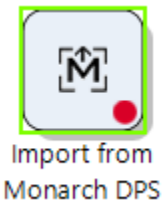
OR:

- Right-click anywhere on the Workflow canvas background. From the context menu, select **Connect --> Import from Monarch DPS**. A new **Import from Monarch DPS** node will be created and the wizard will open.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



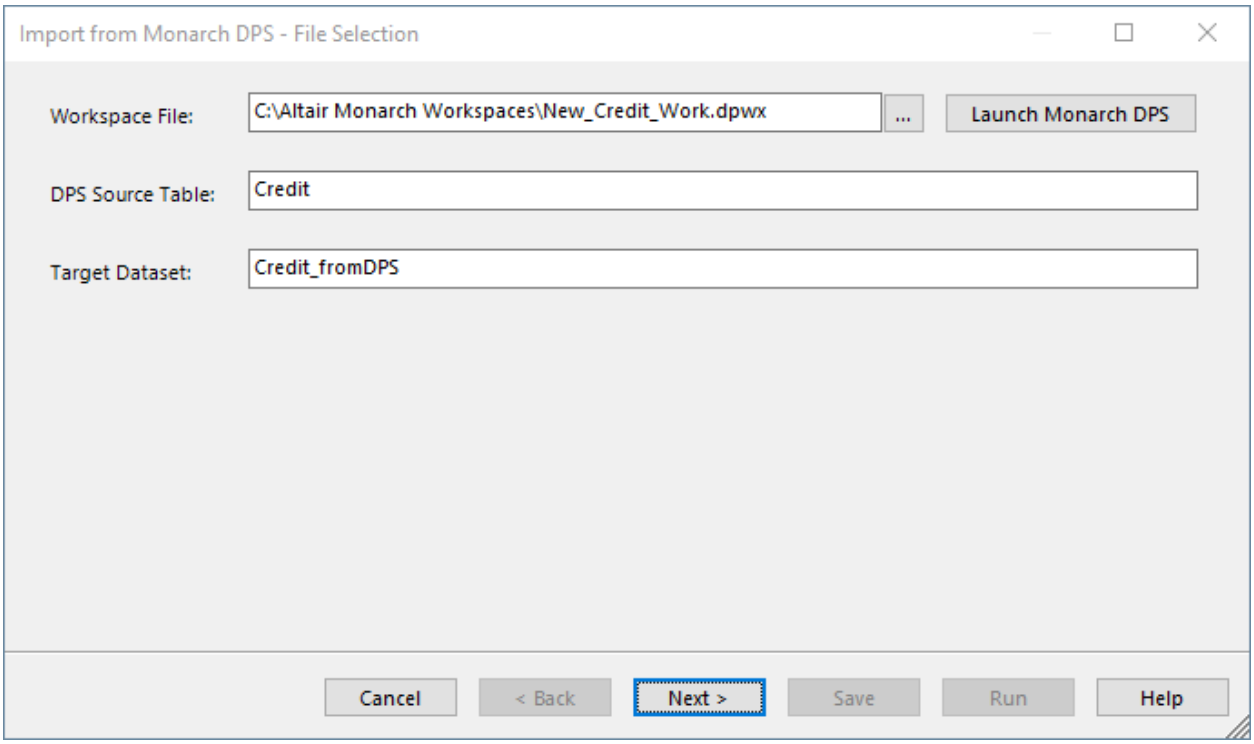
If the **Import from Monarch DPS** node is already on the canvas, double-click it or right-click on it and select **Modify** from the context menu to open the wizard.



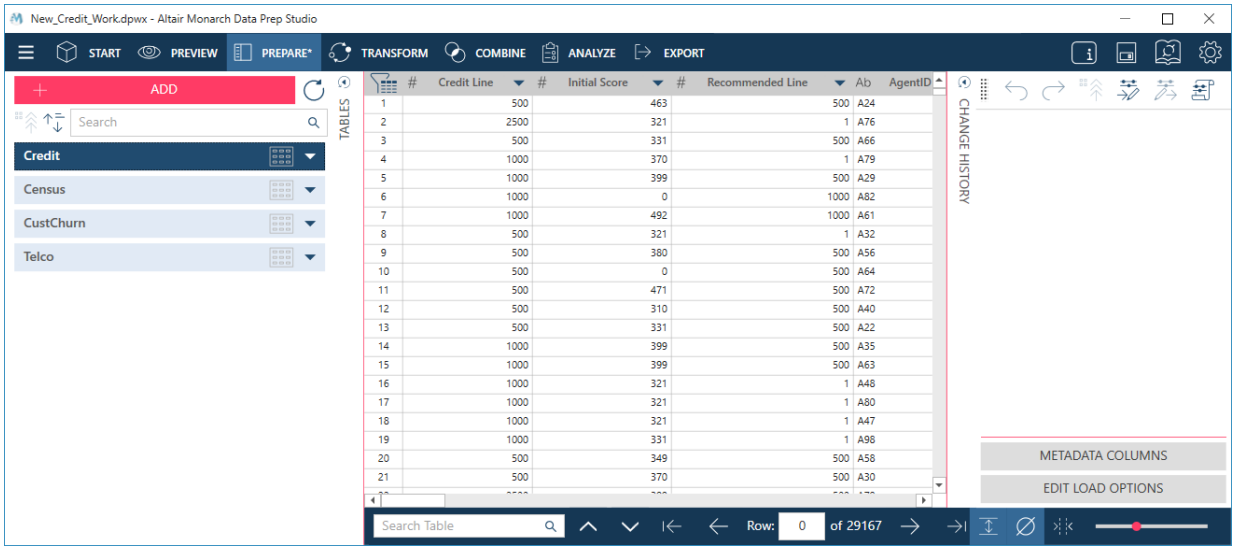
In the first page of the wizard:

- Specify the Monarch workspace file that contains the table to import the data from. Click the browse button to search for the workspace file. If necessary, click **Launch Monarch DPS** to search for the desired source data in your DPS workspace.
- Specify the name of the source table to import from the Monarch DPS Workspace file. In the case of hierarchical tables in the workspace, the names of the tables in the hierarchy should be separated by the @ sign. See an example below.
- In the *Target Dataset* field, enter the name of the new dataset to be created as a result of the import.
    - *Note*: The dataset name can contain spaces, but cannot contain the following special characters: `[]"*+#&!%(){}`'@$^,./:;<=>?\|`

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



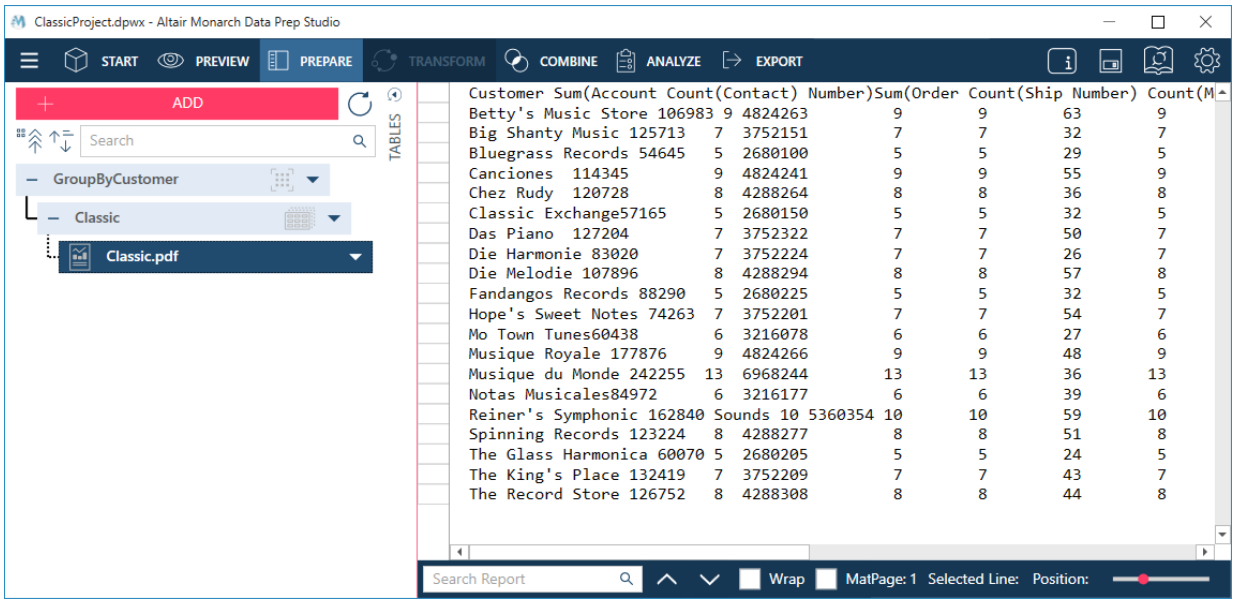
Click **Launch Monarch DPS** to look for the desired table in the selected workspace file if you do not know the exact table name. Monarch Data Prep Studio opens the workspace file in a separate window.



Once you specify all imputs, click **Next.**

Note: In the case of hierarchical tables in the workspace, the names of the tables in the hierarchy should be separated by the @ sign. For example, when specifying the DPS Source Table name for the *Classic* table in the project shown below, specify the name *GroupByCustomer@Classic*

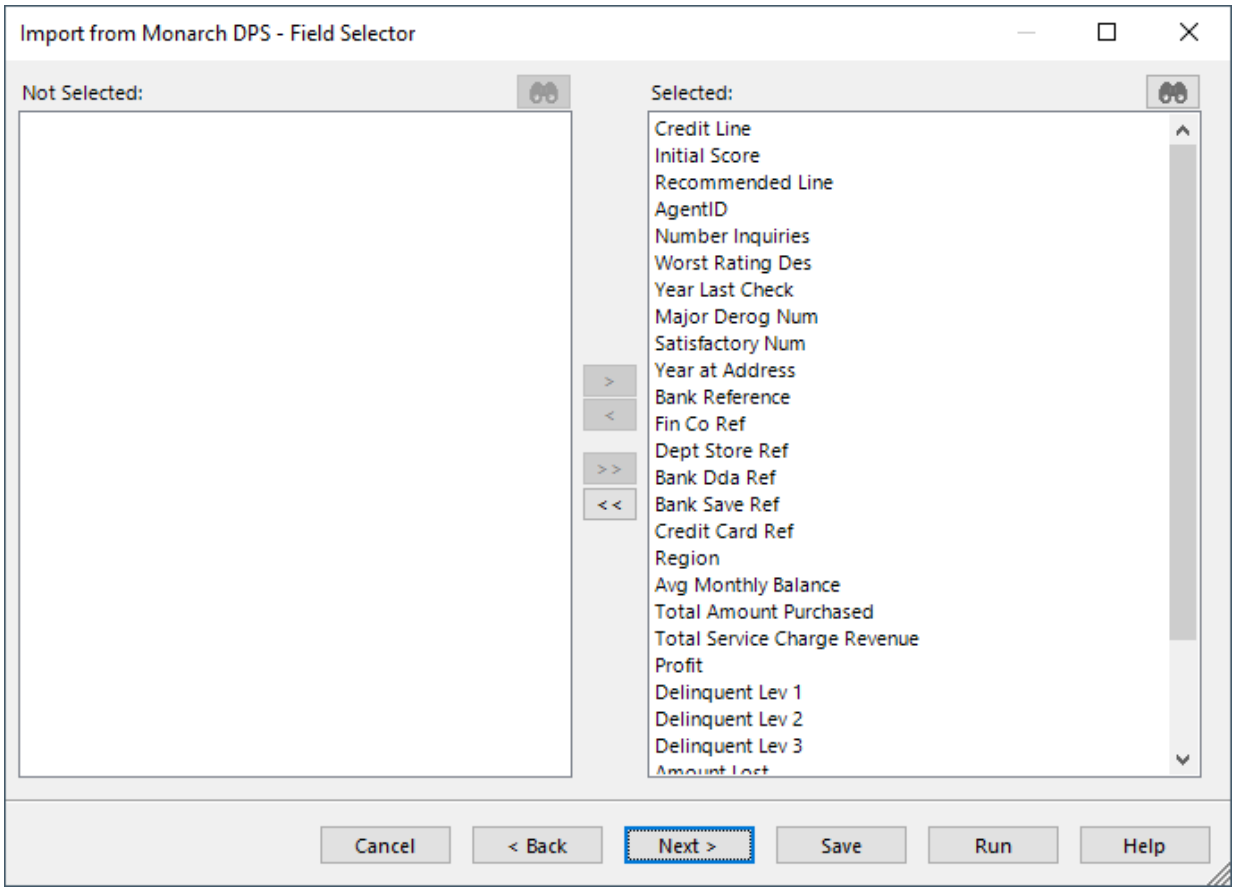
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



In the **Field Selector** page of the wizard, choose the fields to import from your data source. By default, all fields are selected. The field names listed on the right will be imported. Field names listed on the left will be excluded from the import.

To move a field from the include list to the exclude list, highlight the field name(s) you want to exclude and click the left arrow button **<**. To include a field, click the left arrow button **>**. You can also move field names from one list to the other by double-clicking on them.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Next, Save, and **Run** buttons are enabled. At this point you can press **Save** or **Run** to start the import with default settings, or click **Next** to continue specifying the import options.

Proceed to the next step to change field types, define a sample to import, or preview the data before it is imported. See Properties and Preview.

Import from ODBC Data Sources

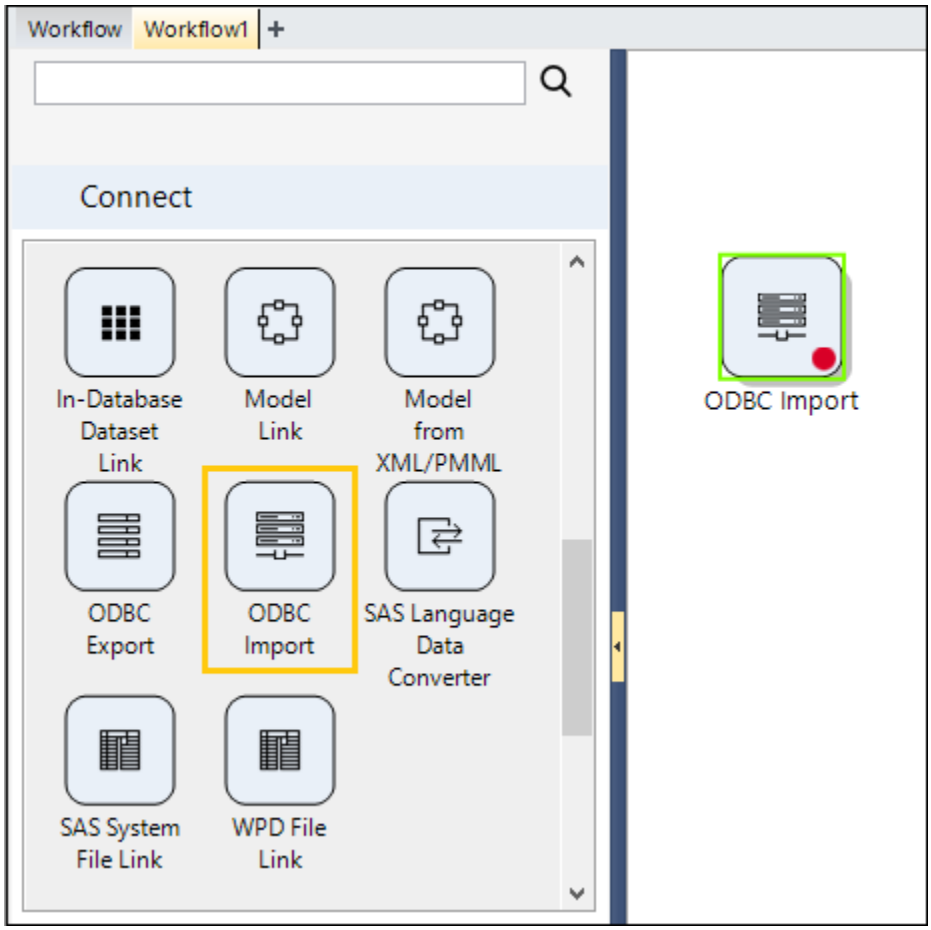
Import from ODBC data sources

ODBC (Open Database Connectivity) is a standard database access method. Using the ODBC Import Wizard, you can import data from a variety of databases and file formats, provided that the appropriate ODBC driver is available. See the topic Accessing Data Sources for details.

The maximum data size that the **ODBC Import** node can import is 65534 columns (the 16-bit encoding limit: 2^16 - 2) and approximately 4.29 billion rows (the 32 bit-encoding limit: 2^32 - 2 = 4294967294).

To import data from a database table or view, drag the **ODBC Import** node from the **Connect** palette to the Workflow canvas. (Alternatively, right-click anywhere on the canvas background and select **Connect --> ODBC Import** from the context menu). Then double-click on the new node or right-click on it and select **Modify**.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The ODBC node wizard will open. Proceed to the next step to specify the ODBC connection properties. See the topic ODBC connection setup.

ODBC Connection Setup

The ODBC Connection dialog provides two ways to set up the ODBC connection:

- Specifying an ODBC connection string
- Selecting a DSN (Data Source Name)

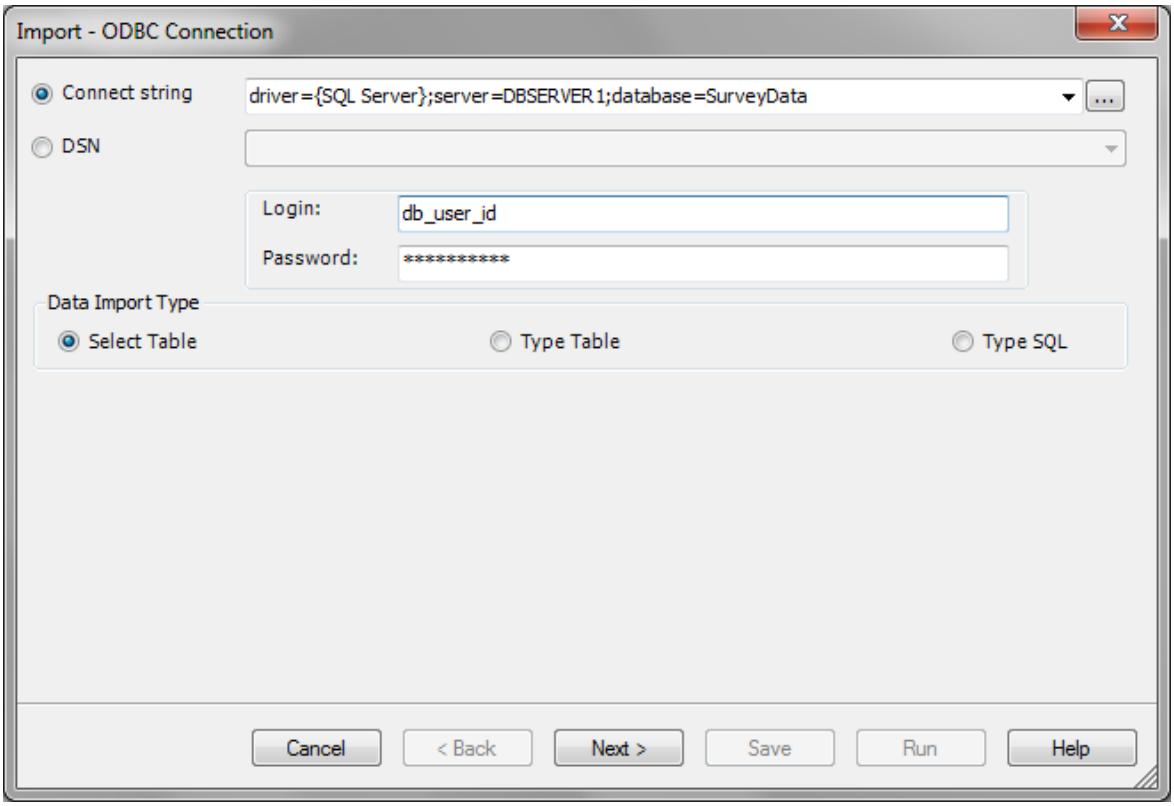
It also allows you to choose whether you would like to type a table name (or a view name), or type a full SQL statement to be passed through to the database during import. This is helpful when you have a large number or tables or when you want something more complex than a simple table or view selection.

The Connect String field keeps the list of most recently used ODBC connect strings for your convenience. If you want to delete one of the saved connect strings from the list, just highlight it while the list is open and press the Delete key.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To make sure that data in specific languages is imported properly, you may need to include the following statements to the ODBC Connection String: "CHARSET=UTF-8" or "FWC=T" (Force SQL_WCHAR Support Enabled). Ask your Database Administrator for assistance if necessary.

Note: When using ODBC import in the client/server configuration for Linux, text field values exceeding 255 characters in length are not imported unless the default length of the Varchar fields in the source database is increased from the 255 default to a value equal to or greater than the maximum field length.



Note: If your ODBC data source is an Excel file, make sure the file is closed before starting the import. No records will be imported if the file is in use.

1. The Connect String Option

The Connect String option allows you to

- enter the ODBC connection string manually, or
- construct the ODBC connection string using the ODBC Connection Information dialog, which is opened by clicking on the browse button to the right of the Connect String field. It prompts the user for the data source type (Oracle, MS SQL Server, etc.) and connection parameters depending on the selected data source type. (*Note: this dialog is not available in client/server mode with server component on Unix.*)

Example connection strings for Windows:

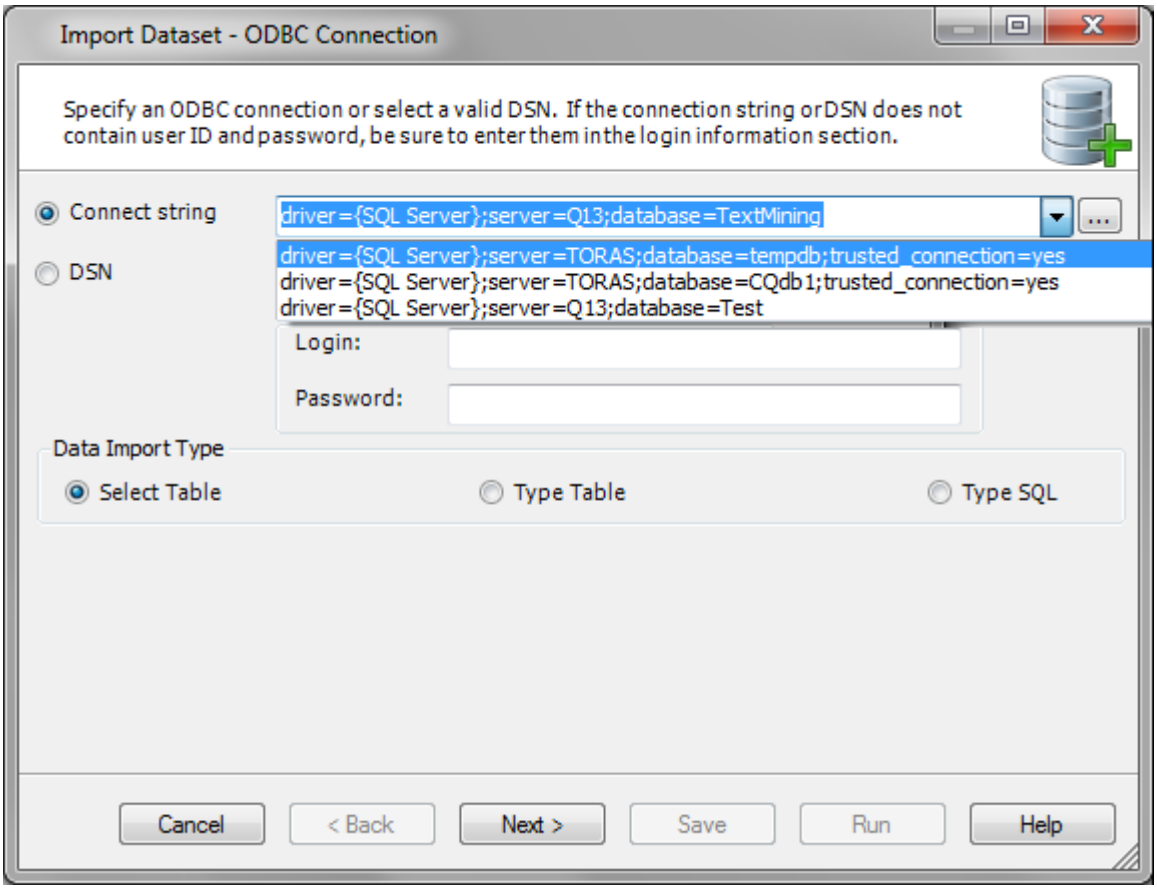
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

SQL Server	Driver={SQL Server};Server=*<server name>*;Database=<*database_name*>;UID=*<login>*;Pwd=*<password>;*
Oracle	Driver={Oracle in OraHome92};DBQ=*<TNS Service Name>*;UID=*<login>*;Pwd=*<password>;*

If you include the login ID and password in the Connection String, you do not need to specify them in the Login and Password fields in the lower part of the ODBC Connection dialog. Otherwise, the Login and Password fields must be filled in.

The ODBC connection string may require multiple options to be specified. For example, for an Oracle database whose character set supports languages such as Japanese, Chinese, etc., SHIFT-JIS, Unicode, UTF8), the option FWC=T (Force SQL_WCHAR Support Enabled) must be included in the connect string in order for all the characters to be displayed correctly. If you are unsure of how to proceed, ask your System Administrator for the required information.

A list of recently used connect strings in the Connect String drop-down list provides an easy way to connect to most frequently used databases.



ODBC Connection Information dialog.

The ODBC Connection Information dialog helps you to build the ODBC connection string.

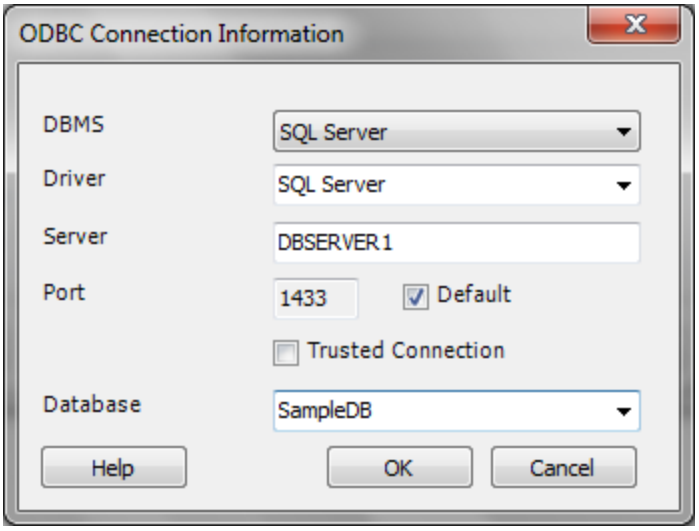
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Select your data source type from the DBMS drop-down list.

Note: This list contains only the most commonly used databases. If the required database type is not listed, use the DSN option rather than the Connect string option (see below).

- Choose the database driver. If there is more than one driver available, make sure you select the driver of the right version, especially in the case of Microsoft Access or Excel.
- The remaining parameters to be specified in this dialog are specific to each data source type. Usually they include the database host name and the database name. Other parameters may include the database port number, authentication type, etc. If you are unsure of how to proceed, ask your System Administrator for the required information.

Example: The case of a SQL Server database:



2. The DSN (Data Source Name) Option

The DSN option can be used to connect to a data source if you already have a DSN on your system that is configured to point to this data source.

When the DSN option is selected, all Data Source Names defined on your system become available in the DSN drop-down list. Open the drop-down list and select the desired DSN.

On Windows, Data Source Names are defined in the standard ODBC Data Source Administrator (in Administrative Tools). On Unix, Data Source Names are defined in the `odbc.ini` file. If you need to create or configure a DSN and are unsure of how to proceed, ask your System Administrator or refer to your data source application's documentation.

After specifying the ODBC connection properties, click Next.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

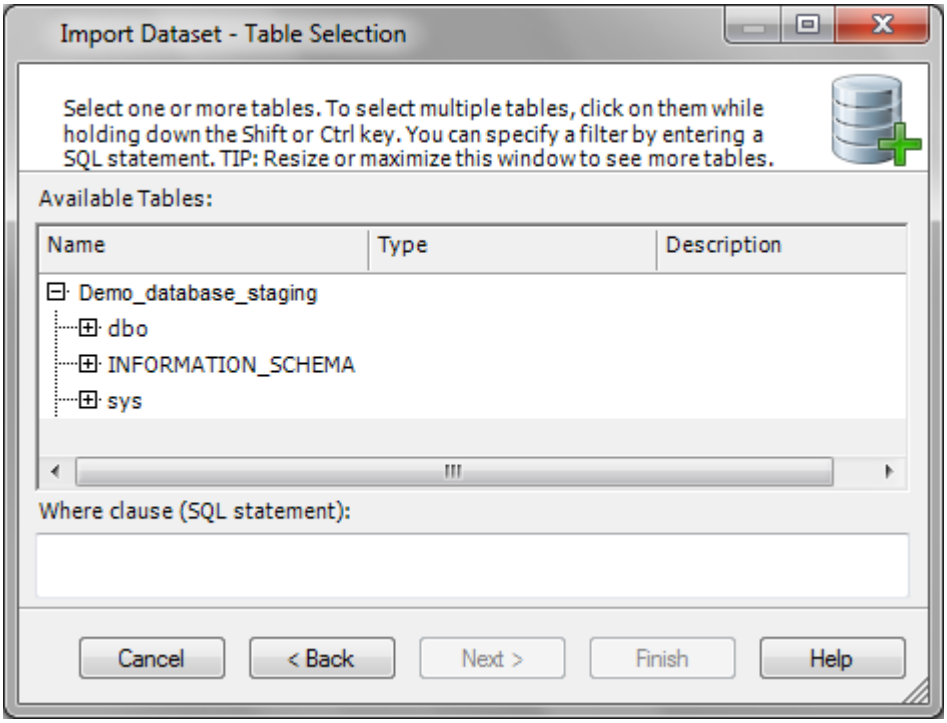
- If you are **importing** data or using the In-Database connector, the next step depends on the Data Import Type you specified.
  - If you chose either the Select Table or Type Table option, proceed to the Table Selection step.
  - If you chose the Type SQL option, the next step is to specify the dataset name and a SQL select statement for retrieving data. Proceed to the Dataset Name step.
- If you are **exporting** data, proceed to the Table Selection step.

See Also:

Import from ODBC data sources

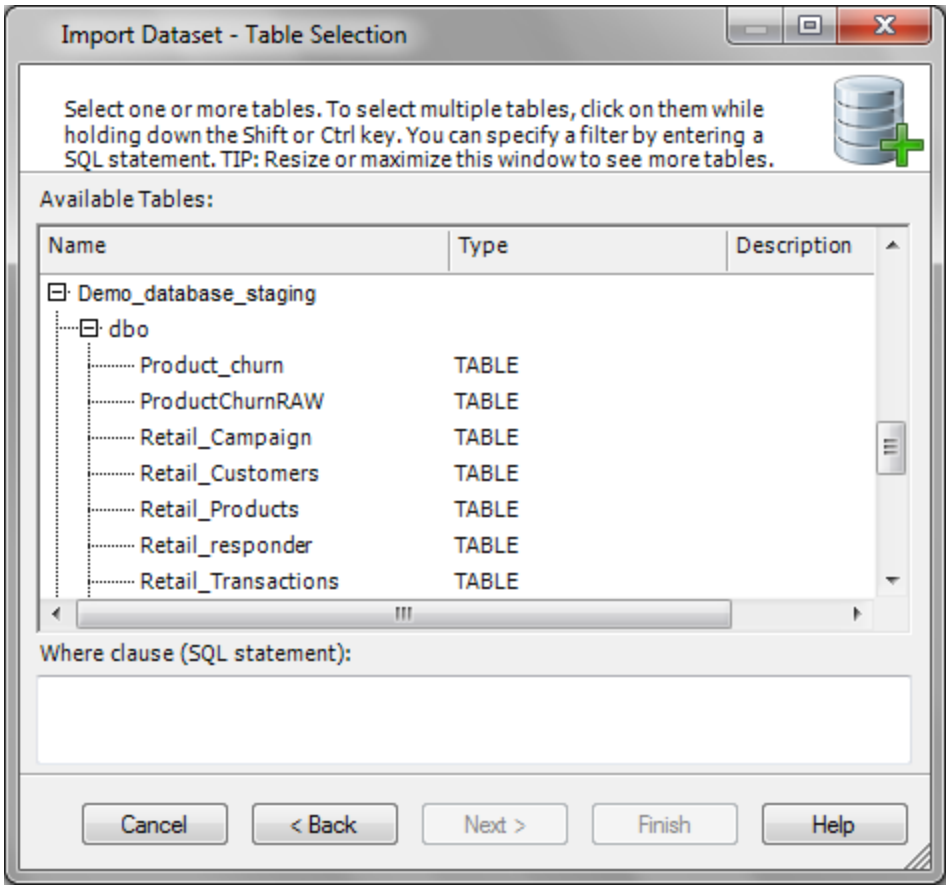
Table Selection

Once the connection to the data source is established, a list of available database schemas is displayed in the Table Selection page. At this step you can select the desired tables or views and specify custom conditions.



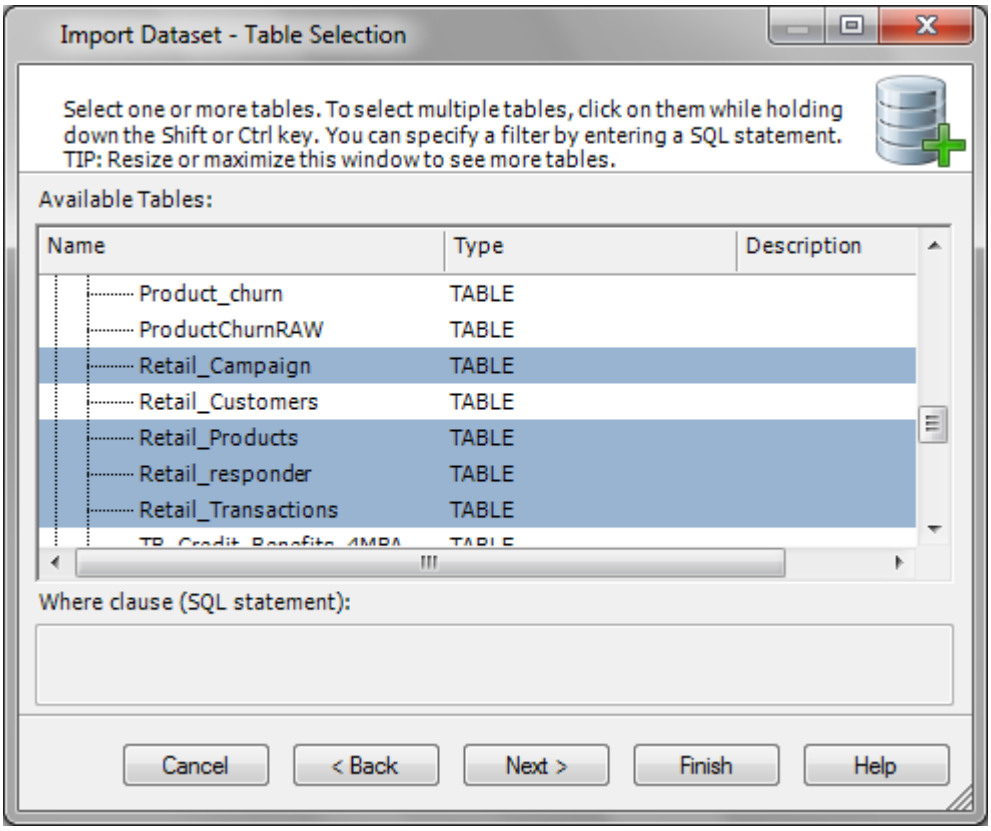
Expand the schema that contains the table(s) or view(s) you wish to import or connect to. Note that you can resize or maximize this window to see more schemas. When a schema is expanded, its tables and views are displayed beneath the schema name:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



You can select one or more tables or views to import or connect to. To select several tables or views, click on them while holding down the Shift or Control (Ctrl) key. In the example below, four tables are selected.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If desired, in the bottom frame titled "***Where Clause* (*SQL statement*)**" you can enter a condition to be imposed on the data to be imported. This condition should be in the form of a standard WHERE clause in SQL. For example, "*Age > 30 and Income >= 50000".* The WHERE keyword need not be entered.

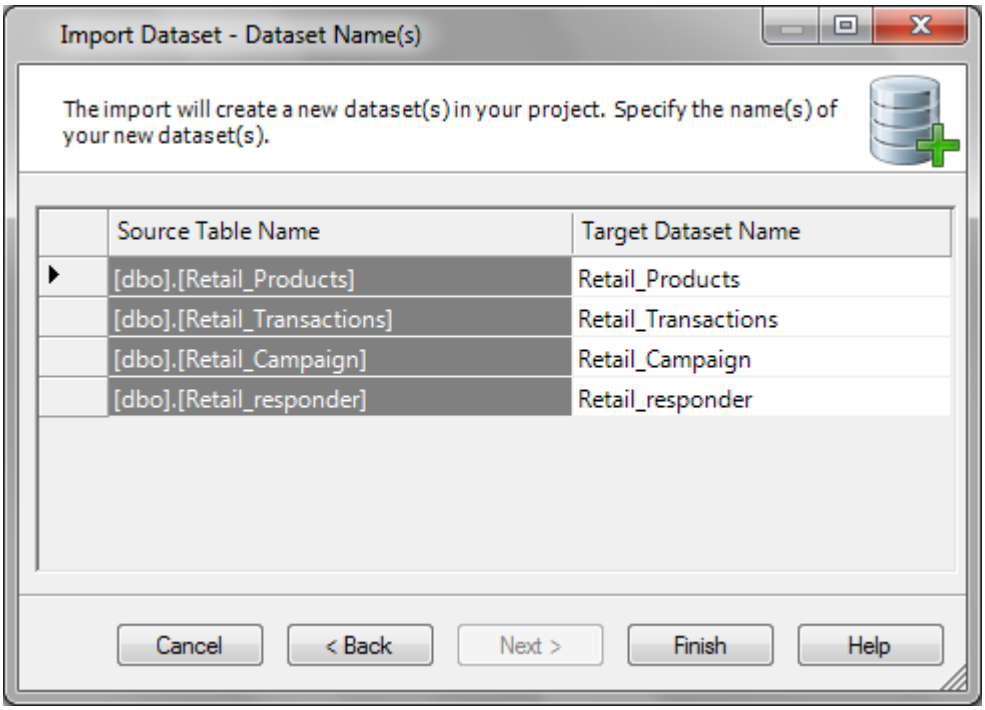
Both Next and Finish buttons are enabled at this step. If you click Finish now, the resulting datasets in the Project Outline will have the same names as the selected source tables or views. Click Next if you want to modify the names of the target datasets as they will appear in the Project Outline. Proceed to the topic Dataset Name(s) for more information about the next step.

Dataset Name

In this page you can specify the names of the new dataset(s) as they will appear in the Project Outline.

If you chose the **Select Table** option at the ODBC Connection Setup step, you will see the following dialog:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The default names for the new datasets are the same as the names of the selected source tables or views. At this step you can modify them as necessary.

Dataset naming conventions:

- dataset names **cannot** contain the following special characters:

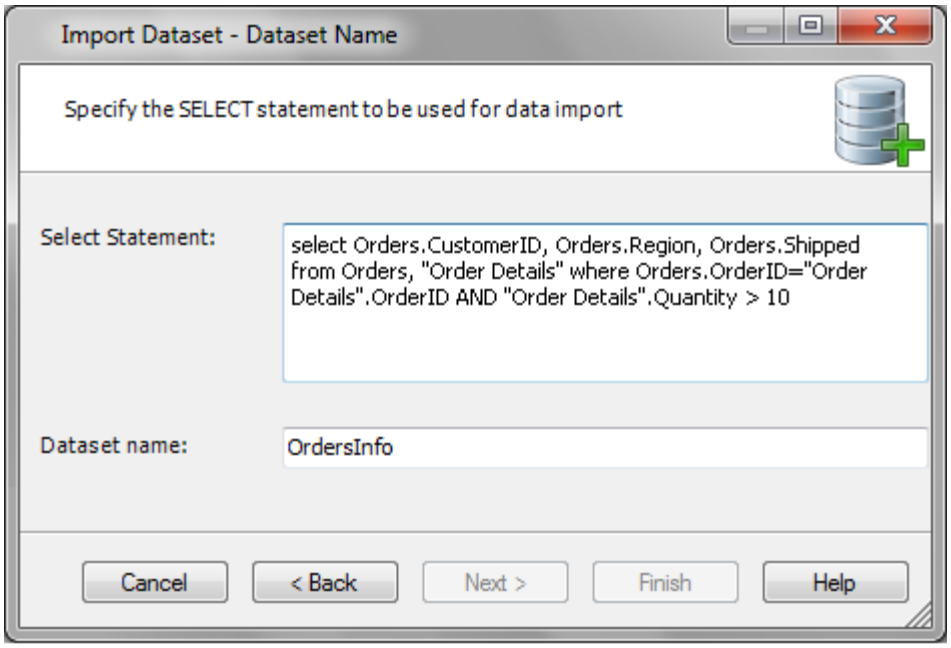
  [ ] " * + # & ! % ( ) { } ` ' @ $ ^ , . / : ; < = > ? \ |

- dataset names **can** contain spaces, for example, "Credit Data".

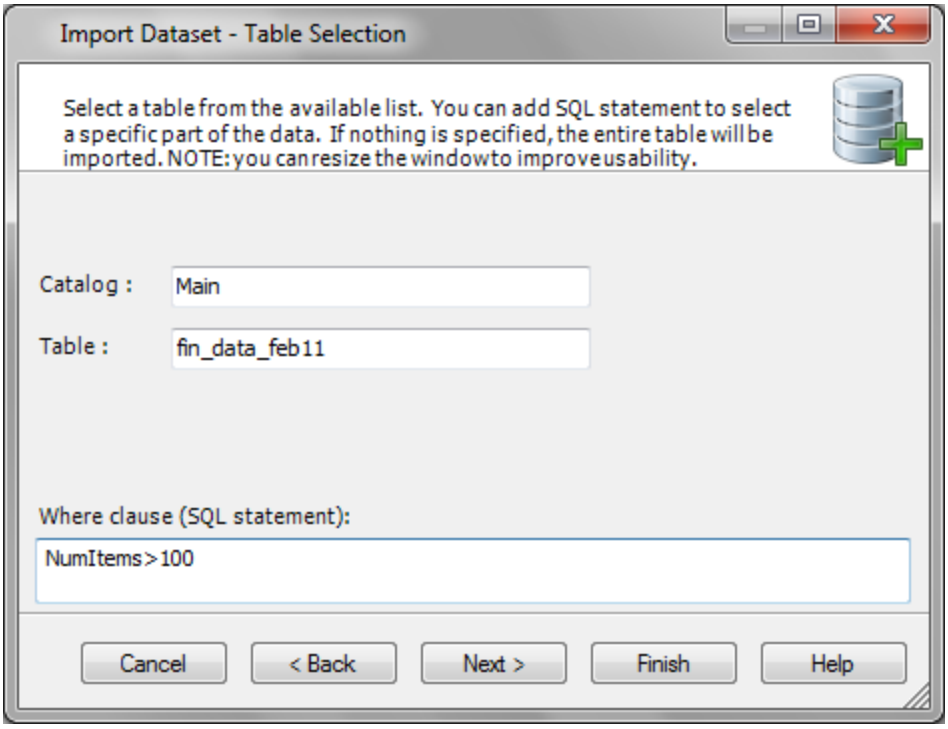
Upon clicking Finish, all of the selected tables will be imported at the same time. In the case of In-Database Analytics, connection with all of the selected tables will be established, and new dataset objects representing these tables will appear in the Project Outline.

If you chose the **Type SQL** option at the ODBC Connection Setup step, you also need to enter the SQL statement for retrieving data from the database. The statement can involve several tables or views. In this case you will see the following dialog (a SELECT statement is entered here as an example):

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If you chose the **Type Table** option at the at the ODBC Connection Setup step, you need to specify the catalog or schema name and the source table name, with an optional WHERE clause to narrow your record selection:



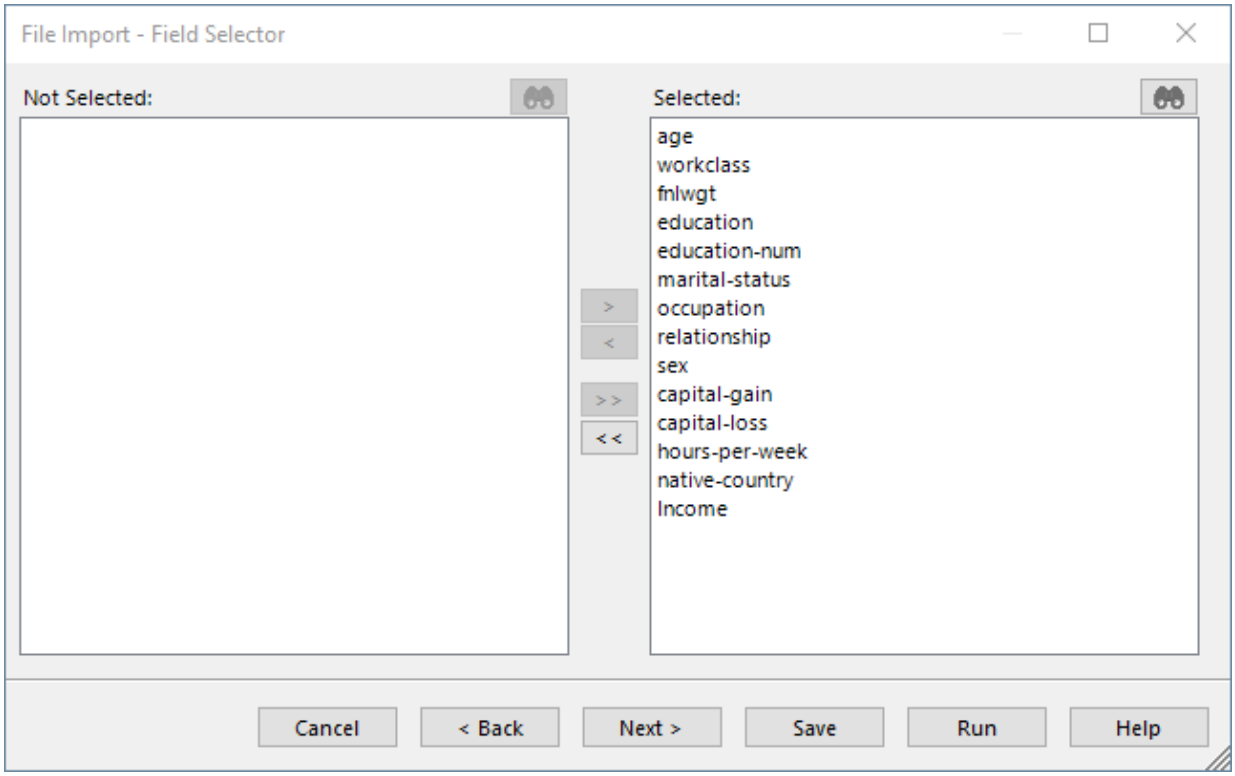
At this point you can press **Finish** to start the import with default settings or **Next** (if applicable) to specify further import options.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Field Selector

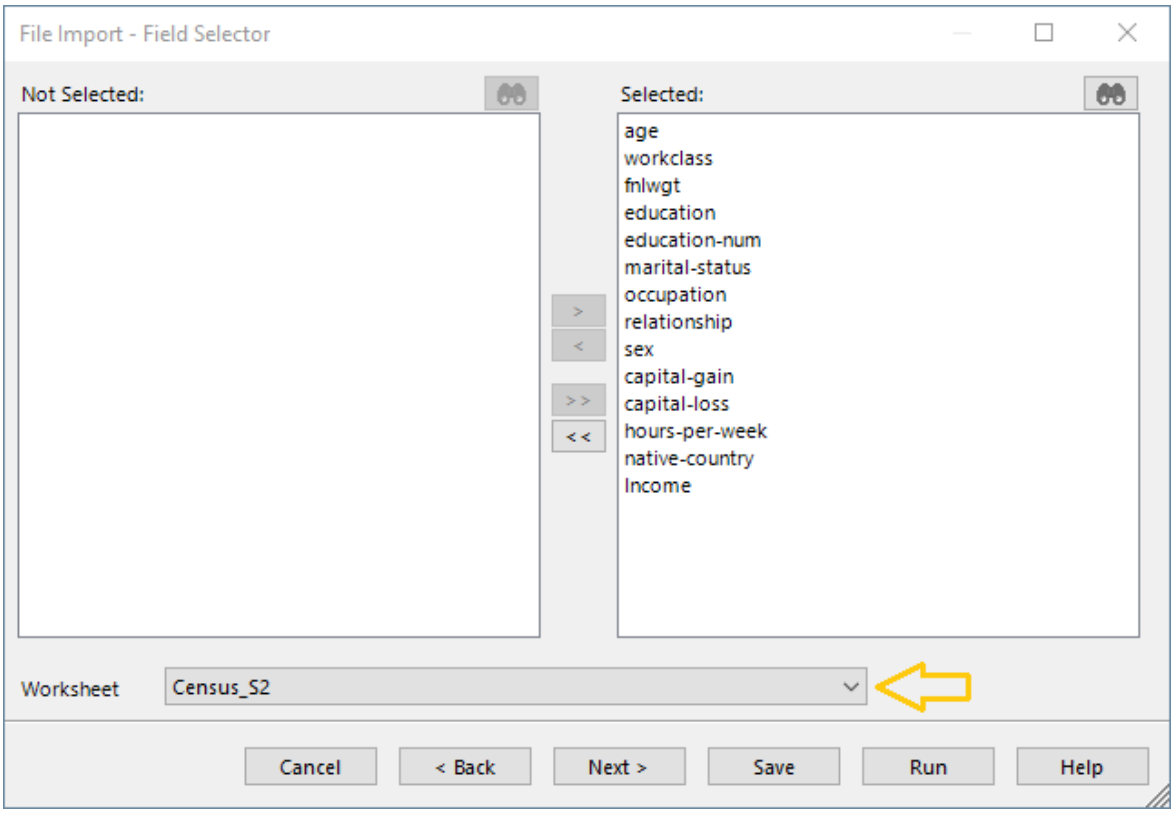
In the Field Selector page of the wizard, choose the fields to import from your data source. By default, all fields are selected. The field names listed on the right will be imported. Field names listed on the left will be excluded from the import.

To move a field from the include list to the exclude list, highlight the field name(s) you want to exclude and click the left arrow button **<**. To include a field, click the left arrow button **>**. You can also move field names from one list to the other by double-clicking on them.



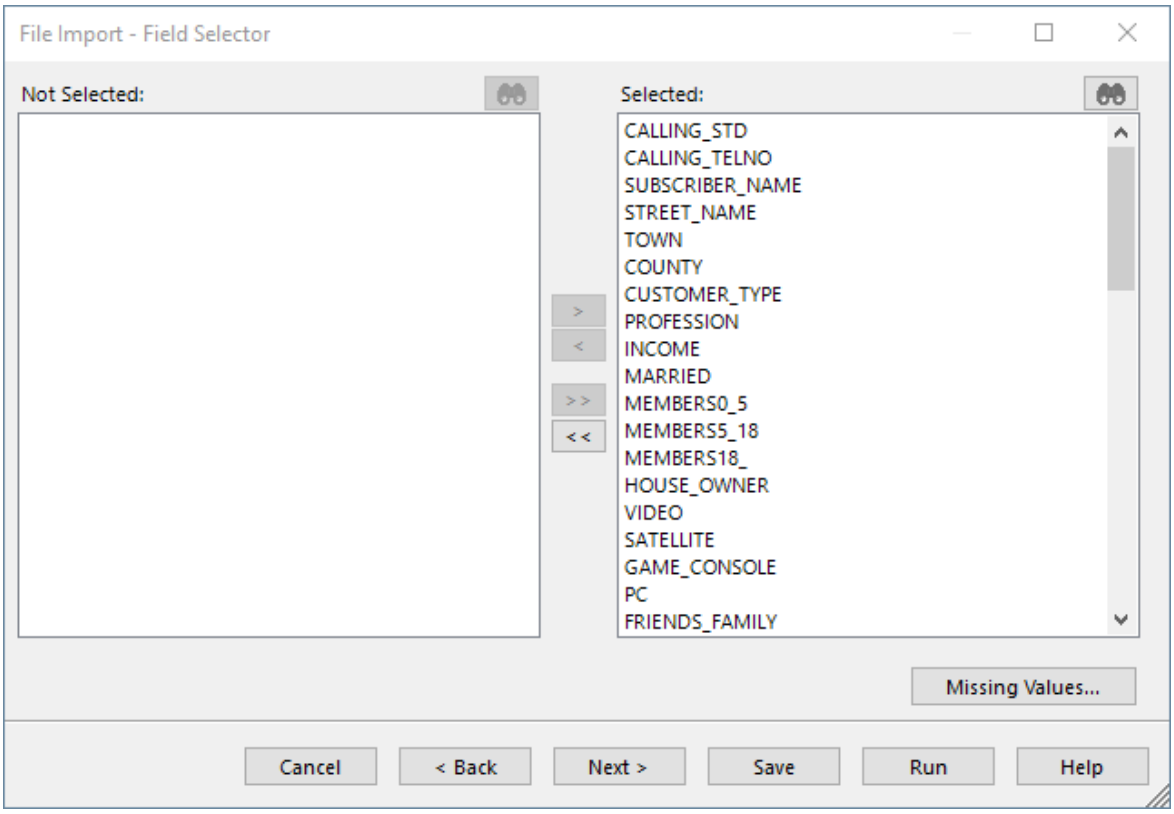
If you are importing data from an Excel spreadsheet with multiple sheets, select the desired worksheet from the drop-down list at the bottom of the wizard page.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



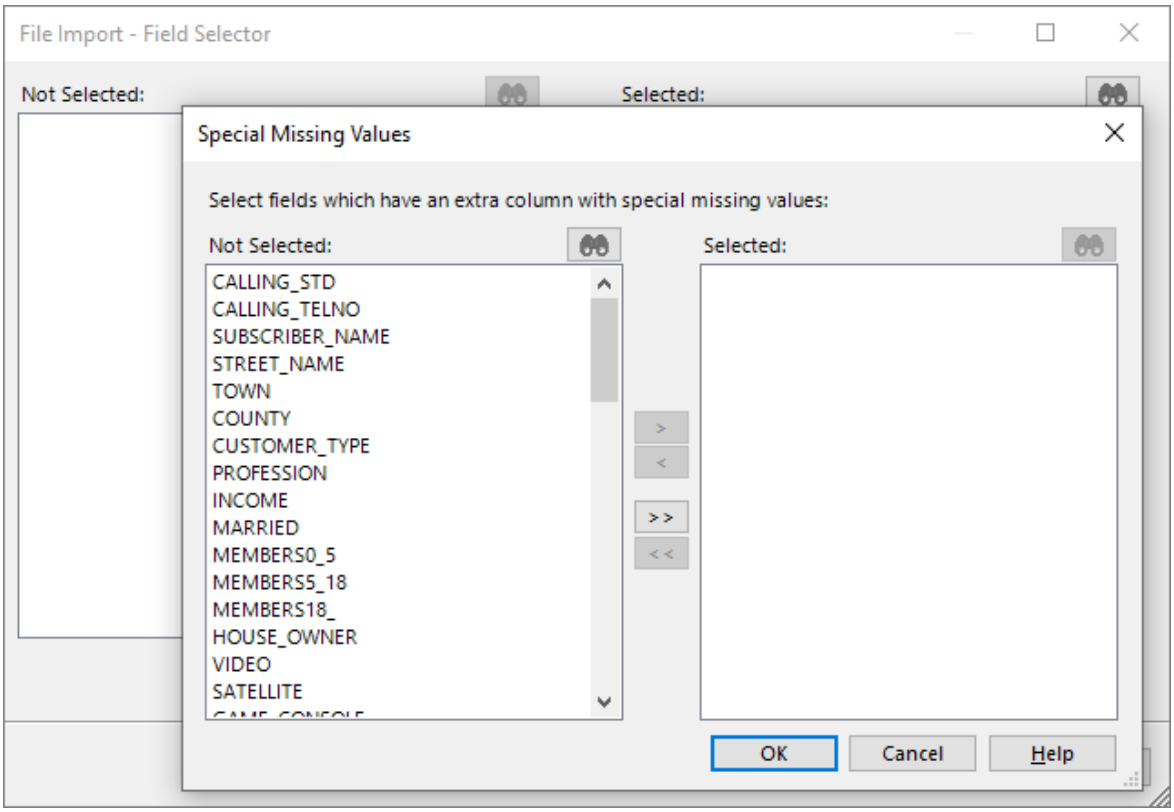
If you are importing data from a SAS System file, you have an option of defining special missing values by clicking the **Missing Values** button at the bottom right corner.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Select the fields that have an extra column with special missing values:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Next, Save, and **Run** buttons are enabled. At this point you can press **Save** or **Run** to start the import with default settings, or click **Next** to continue specifying the import options.

Proceed to the next step to change the field types or define a sample to import (see Field Transformer).

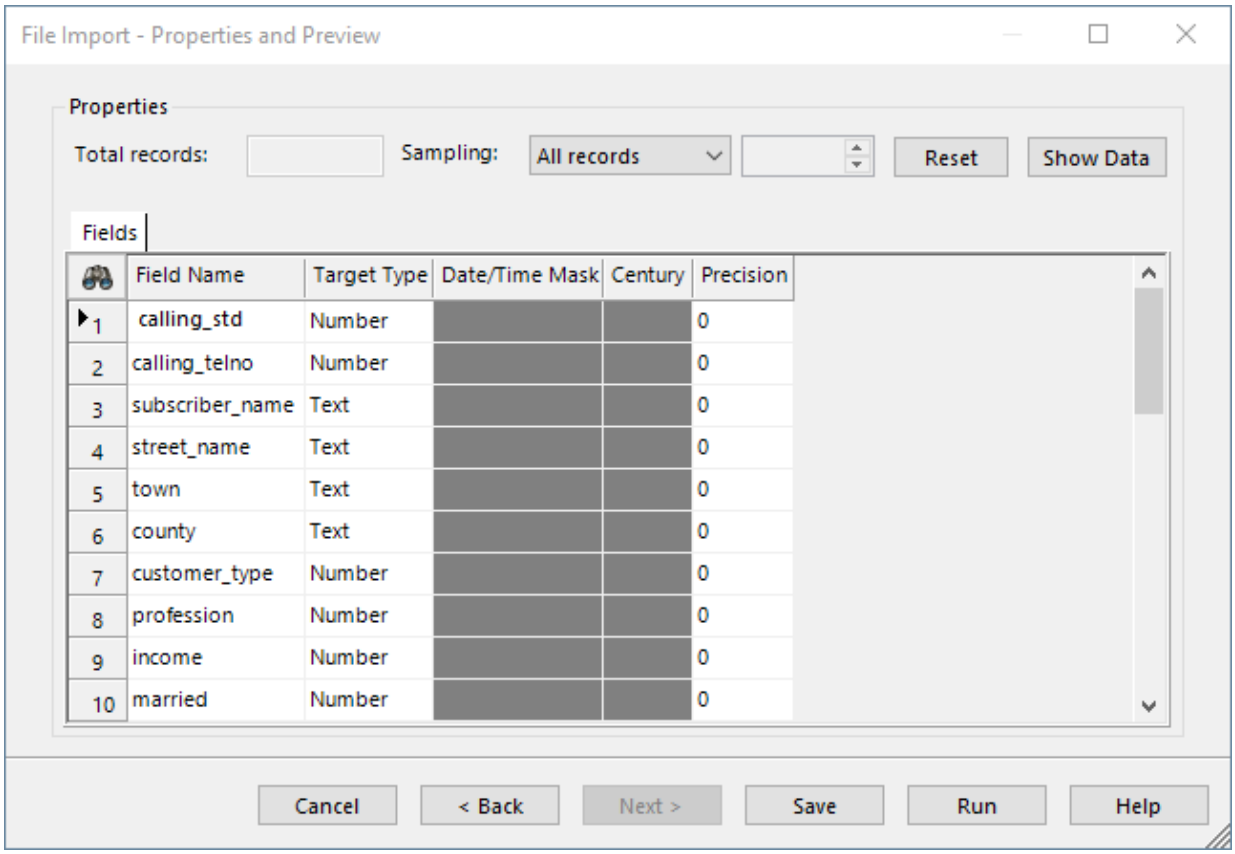
Properties and Preview

The **Properties and Preview** page of the import wizards allows you to:

- modify the field type and specify precision for numeric fields
- specify the sample size - the number or percentage of records to import
- specify a date mask for date, time, and timestamp fields
- preview the data as it will appear in the imported dataset

The Properties and Preview page is the same for all source data formats.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

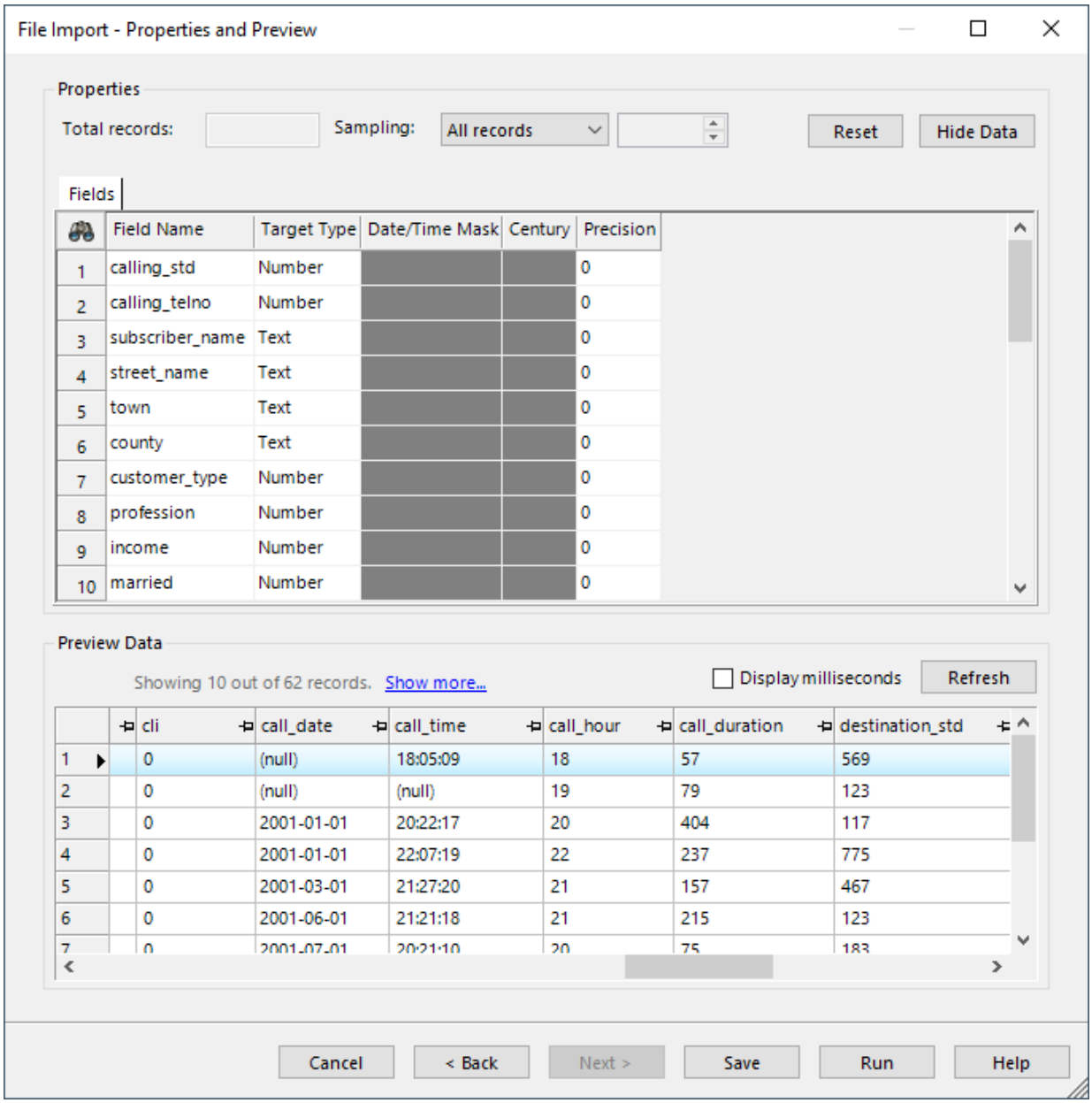


The **Total records** field shows the total number of records in the source file for all source types except Text files and database tables (in ODBC Import).

The **Reset** button resets all the settings in this page to the default values.

The **Show Data** button allows you to preview the data as it will appear in the imported dataset. This function is especially useful when importing data from text files to make sure the date/time fields are correctly imported as Date/Time/Timestamp, with correct date/time masks. See the subsection **Specifying Date/Time Masks** further in this topic.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Sampling

The default is set to import all records from the source file or table. Choose a sampling option is you only need to import a sample of the data.

All Records	Imports all records in the source file or table.
Number records	Imports the specified number of records as a **random sample** of the source data.
Percentage	Imports the specified percentage of records of the source data as a **random sample**.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Top N record	Imports only the first N records of the source data, where N is the number specified by the user.

Field Editor

Field Name	The name of the field as it will appear in the imported dataset. This column is editable. You can rename any field if desired.
Target Type	The data type of the field. Admissible types are *Text, Integer, Number, Boolean, Date, Time,* and *Timestamp*. To change the data type for a field, click in the Target Type column and select the desired type from the drop-down list. To set the type of multiple fields at a time, drag the mouse in the first (index) column of the grid or click in the first column while holding down the SHIFT or CTRL key. Then click in the Target Type column for any of the highlighted fields and select the desired type from the list. This will set the same type for all of the highlighted fields. The **Preview** button allows you to preview the data as it will appear in the imported dataset. Note that if you change the type for a field, and the new type is incompatible with the raw data in this field (for instance, if you set the type to "Number" for a text field), the Preview window will show ***null*** for the values in this field, and this is how it will be imported if you proceed with the import. If you change the types in this page, especially for multiple fields at a time, make sure you specify the types compatible with the source data.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Precision

Depending on the data type, *precision* affects the imported field in different ways.

Numeric (Integer, Number):

- When the ***precision*** is a **positive** number, the values in the selected field will be rounded to the specified number of decimal places. For example, if you want only 2 decimal places for the values in the selected field, enter 2 in the ***precision*** column. All the values in this field will be rounded to 2 decimal places when imported. For example, 34.42852 will be imported as 34.43. To round the values of a Number field to the nearest Integer, just change its **Target Type** from Number to Integer. If the field type is Integer, positive precision has no effect.
- When the ***precision*** is a **negative** number, the values in the selected field will be rounded to the left of the decimal point. For example, to round all values in a numeric field to the nearest hundred, specify the precision value **-2**. The values in the selected field will be rounded to the nearest hundred when imported. For example, the value 7321 will be imported as 7300, and the value 7368 will be imported as 7400.
- Precision **0** (default) means that every value in the field is imported as is.

Text:

- When the ***precision*** is a **positive** number, only the specified number of characters, starting from the beginning, will be imported for each field. For example, if you want only the first 3 characters of a name to be imported for this field, enter 3 as the precision. For example, the string **'Jack Smith'**, will be imported as **'Jac'**.
- When the ***precision*** is a **negative** number, only the specified number of characters, starting from the end, will be imported for each field. For example, you only want the last 3 characters of a name to be imported for this field. Enter -3 as the ***precision***. All the strings in this field will only be the last 3 letters. A field with the string **'Jack Smith'**, will be imported as **'ith'**.
- Precision **0** (default) means that that every value in the field is imported as is.

Date/Time Mask

Applicable only when Target Type is *Date, Time,* or *Timestamp*. In this case, the date/time mask must be defined. The Date/Time mask definition dialog opens if you change the Target Type of a field to Date, Time, or Timestamp. If the field already has one of these types, the mask can be modified by double-clicking in the Date/Time Mask column.

Date mask must be specified with capital letters D(day), M(month), Y(year), for example, `YYYY-MM-DD`, `MM/DD/YYYY`, etc.

Time mask must be specified with lowercase letters `h` (hours), `m` (minutes), `s` (seconds), `f` (milliseconds), and `tt` (AM or PM). Examples of time masks:

```
hh:mm:ss
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
hh:mm:ss:fff
hh:mm tt
```

The Date/Time mask definition dialog provides several standard masks and allows specifying a custom mask.

To make sure that you specify the correct date/time mask, check the source values in the Text File Preview or Text Delimiter pages of the wizard. Click the **Show Data** button in the Properties and Preview page. For details, see the subsection **Specifying Date/Time Masks** below.

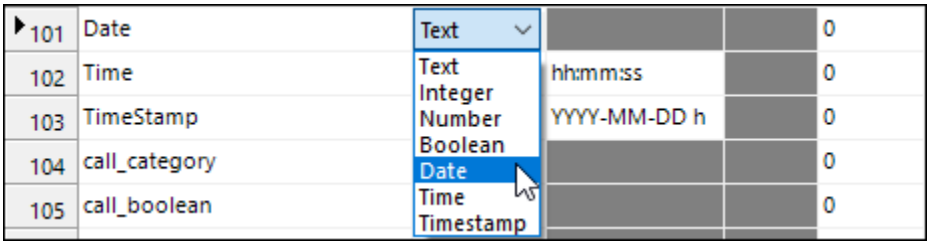
Century Applicable only when Target Type is Date or Timestamp. This is entered through the Date/Time Mask definition dialog. If the date/time mask is already defined, double-click in the Century column to open the Date/Time Mask dialog.

Specifying Date/Time Masks

Metadata in SAS System, Excel, SPSS, and R files and ODBC data sources contains information about field types. The import function for these data types reads the metadata from the source and automatically detects data types, then assigns the target field types accordingly. For these sources, you do not have to verify and explicitly set the target type in the import wizard unless you intend to change the types for specific fields.

Text files (*.txt, *.csv) normally do not contain any metadata, so the import wizard guesses the field types based on the first 20 records of the source text file. Date, Time, and Timestamp fields in text files are read as **Text** data type by default. When importing text files, check the target data types in the last page of the File Import wizard. Set the desired types for the fields whose types were not detected automatically, such as Date, Time, and Timestamp fields.

The Date/Time mask definition dialog opens when you change the Target Type of a field to Date, Time, or Timestamp.



For fields already defined as Date/Time/Timestamp, the mask can be modified by double-clicking in the Date/Time Mask column.

The **Mask** field provides standard mask options to be selected from a drop-down list, while also allowing to specify a custom mask. In the same dialog, the **Raw Data Preview** grid shows the original values in the selected field(s) being formatted.

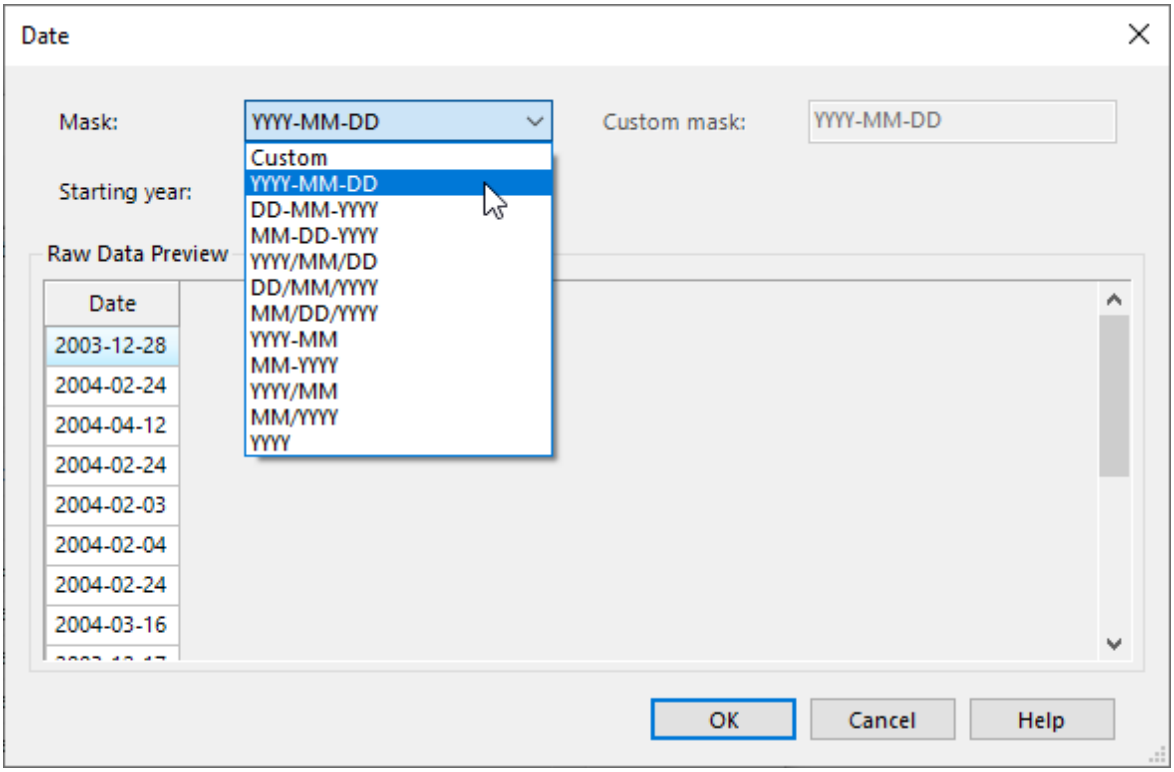
Select the appropriate mask that fits the data in the selected field(s) displayed in the Raw Data Preview frame. If the dropdown does not contain the correct mask, select "*Custom*" and enter the correct format in the **Custom mask** field.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Specifying the correct mask ensures that the data is **read** correctly. Once the values are read, the way they are **displayed** may also depend on the regional settings (number, time, and date formats) on the host operating system.

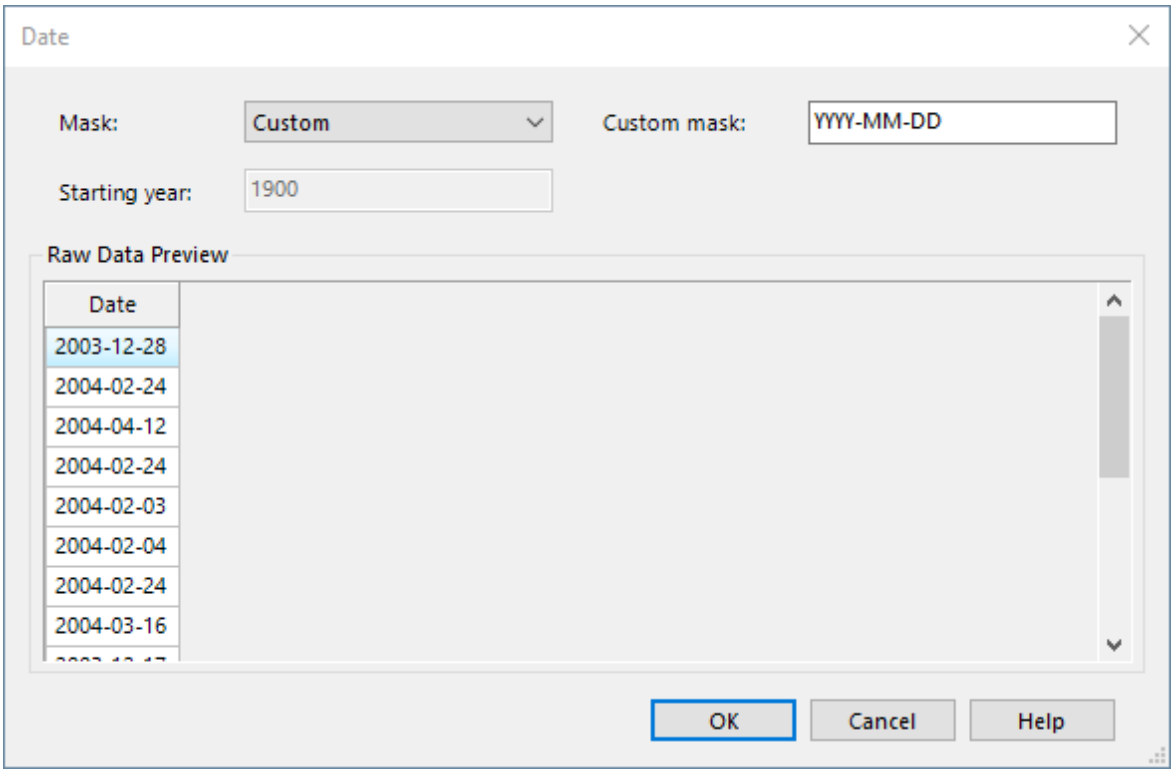
Specifying the wrong mask results in the values being imported as *null* (missing values).

Example 1: Specifying a mask for a ***Date*** field



If the **Custom** option is selected in the Mask drop-down list, the **Custom mask** field becomes enabled for text input:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



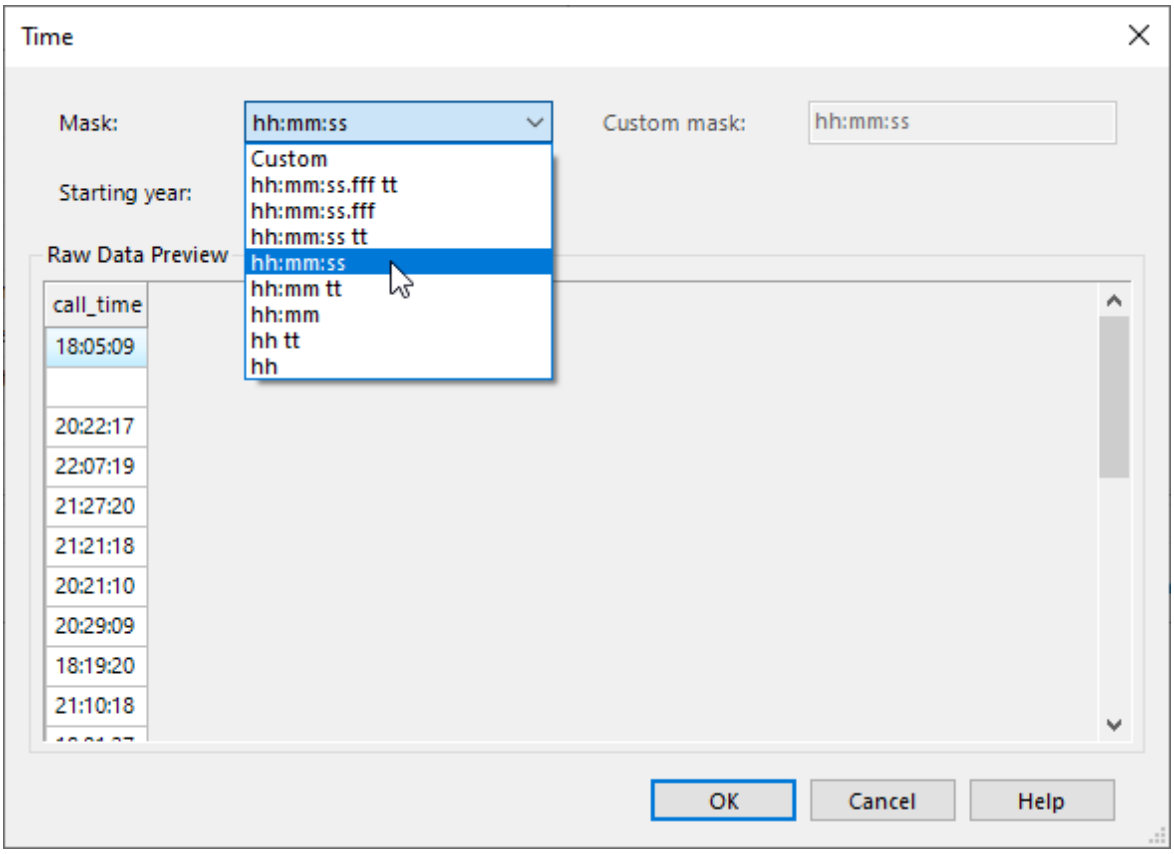
Example 2: Specifying a mask for a ***Time*** field.

The standard time masks available for selection include those with milliseconds and AM/PM designations.

Examples of time data and the corresponding masks:

Time data example	*Mask*
05:15 PM	`hh:mm tt`
13:47:33	`hh:mm:ss`
17:15:48.641	`hh:mm:ss:fff`
05:15:48.641 PM	`hh:mm:ss:fff tt`

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Example 3: Specifying a mask for a ***Timestamp*** field.

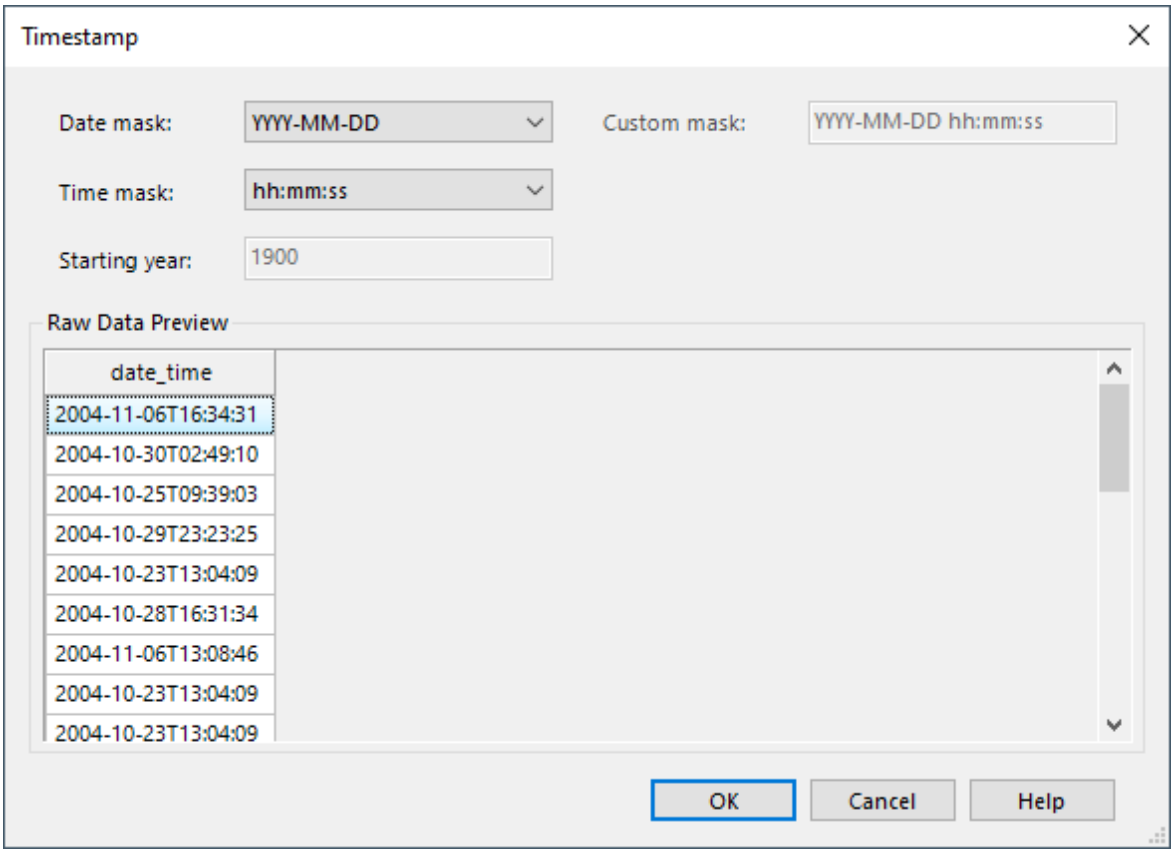
For Timestamp fields there are two parts of the mask to select – one for the date part and one for the time part. The date and time parts in the source data are assumed to be separated by a single space or the letter *T*.

The Time masks available for selection include those with milliseconds and AM/PM designations.

Examples of timestamp data and the corresponding masks:

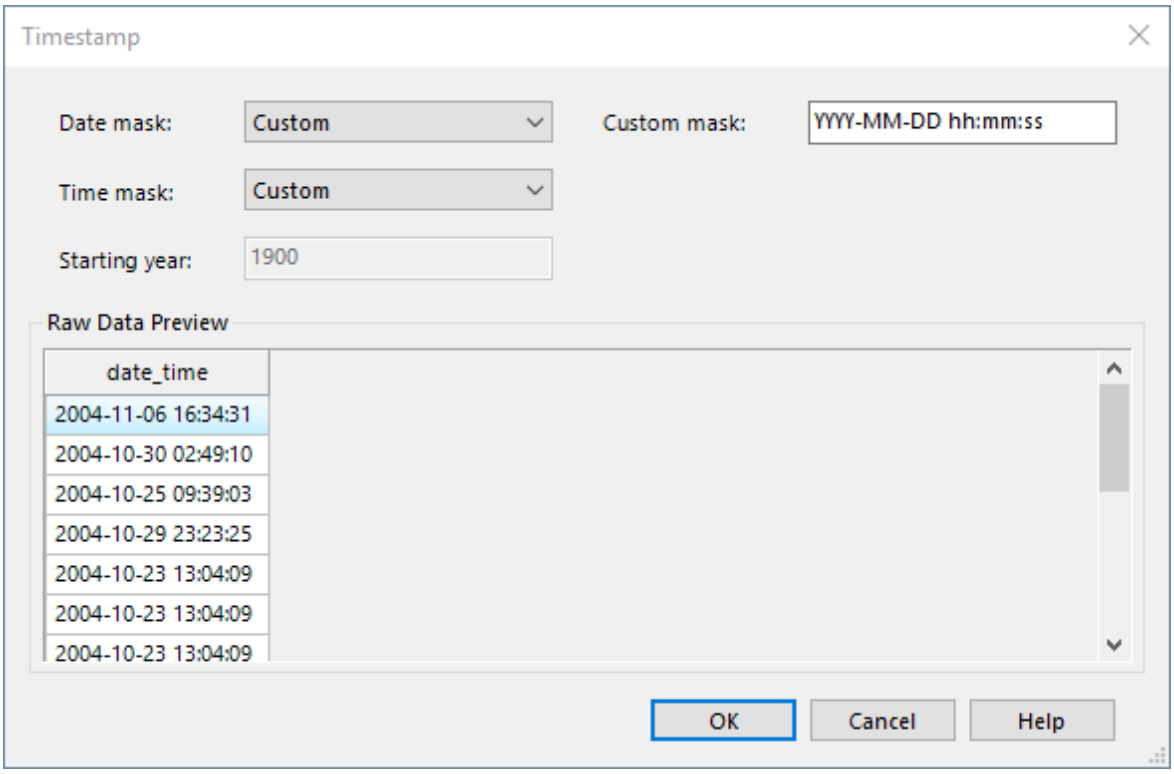
Timestamp data example	*Date Mask*	*Time Mask*
2012-10-01 05:15 PM	`YYYY-MM-DD`	`hh:mm tt`
2012-10-01T05:15 PM	`YYYY-MM-DD`	`hh:mm tt`
03/24/2019 13:47:33	`MM/DD/YYYY`	`hh:mm:ss`
1984/03/14 17:49:11.641	`YYYY/MM/DD`	`hh:mm:ss:fff`
27/10/2013 05:15:48.641 PM	`DD/MM/YYYY`	`hh:mm:ss:fff tt`

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If the **Custom** option is selected in the Mask drop-down list, the **Custom mask** field becomes enabled for text input. By default, the Custom mask box concatenates the two previously entered masks, separating them with a single space.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The mask definition will be applied to the selected field(s). If multiple fields were selected in the Properties and Preview page, then the Raw Data Preview grid will contain all of the selected fields, and all of them will be assigned the same mask once you click **OK**.

Click **Show Data** in the Properties and Preview page to verify that the values are displayed correctly in the fields whose Date, Time, or Timestamp masks you specified.

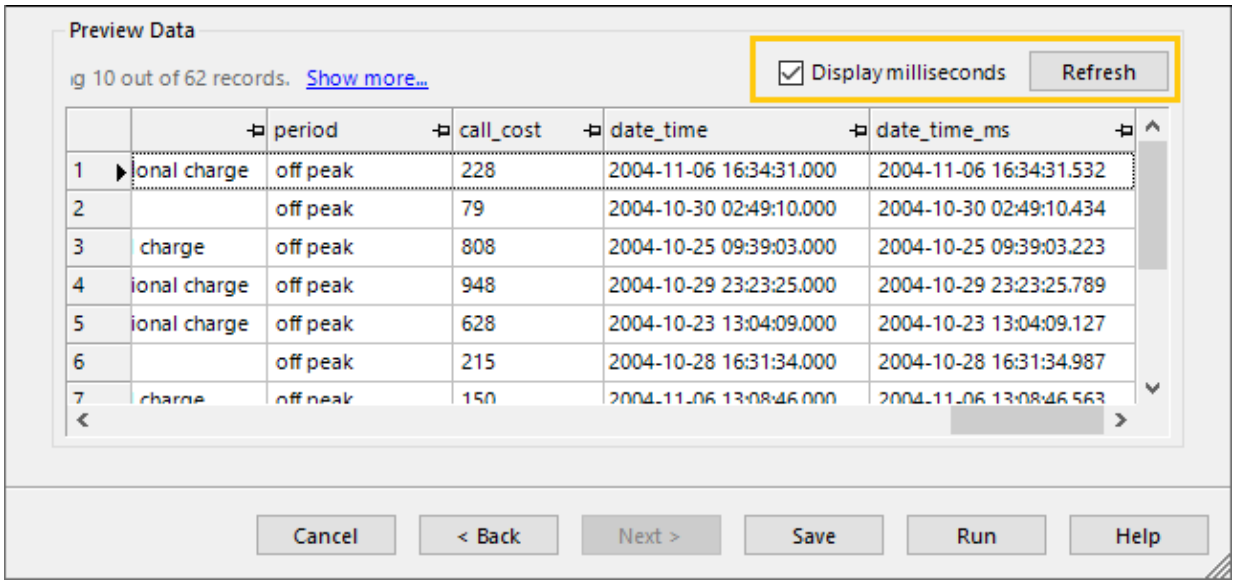
Preview Data

Click **Show Data** to preview the data as it will appear in the imported dataset. This opens the **Preview Data** frame at the bottom of the window. Click **Refresh** to show or update the data in the preview frame. By default, the top 10 records of the transformed dataset are displayed. Click "*Show more...*" or just scroll down in the Preview Data frame to view more records. Click **Hide Data** to hide the Preview Data frame.

The way the values are displayed may depend on the regional settings (number, time, and date formats) on the host operating system.

If your source data contains Time or Timestamp values with milliseconds, select the option ***Display milliseconds*** and click **Refresh**. The milliseconds part will be displayed for all Time and Timestamp fields. An example is shown below.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The option ***Display milliseconds*** explicitly applies the custom format strings defined by the Date and Time masks you specified. If the option ***Display milliseconds*** is **not** selected, Time and Timestamp values are displayed in the Windows' standard format for time with milliseconds, exactly as the values are stored (for example, 2008-01-01T00:30:45.1250000 - note the trailing zeroes).

Specifying correct masks for Date, Time, and Timestamp fields is necessary for the data to be read correctly and imported successfully. Note that the way the values are **displayed** in the preview and in the resulting dataset may be different from the masks you specified depending on the regional settings (number, time, and date formats) on the host operating system. For example, if dates in your source file are in the format YYYY-MM-DD, you must specify the corresponding mask "YYYY-MM-DD" for the data to be read correctly. However, if the Date format in your Regional Settings is DD/MM/YYYY, then the values will be displayed in the format DD/MM/YYYY in the preview frame and in the resulting dataset.

In the imported data, Time and Timestamp values with milliseconds can be displayed only in the views that show raw data - the Data tab of datasets, the Node Data tab of Decision Trees and Strategy Trees, and Preview Data frames in the nodes such as Variable Transformations.

In-Database Analytics

In-Database Analytics is available as an add-on for Knowledge Seeker and Knowledge Studio. It allows users to perform data mining tasks directly in the source database without importing tables.

Overview

In-Database Analytics helps reduce the resources required to analyze data by performing data mining tasks directly in the source database without importing tables into Altair data files.

Altair In-Database Analytics is compatible with Teradata, SQL Server, Oracle, and Spark SQL. Since In-Database Analytics uses ODBC interface for database access, ODBC drivers are required for each respective database.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

ODBC drivers on Windows:

Microsoft SQL Server ODBC driver is standard on all Windows systems. ODBC drivers for Teradata and Oracle usually come standard with the respective database client software and are available on the vendor's web sites.

ODBC drivers on and Linux:

KS Server for Linux requires unixODBC driver manager for database drivers. We recommend downloading and installing native ODBC drivers for the required databases.

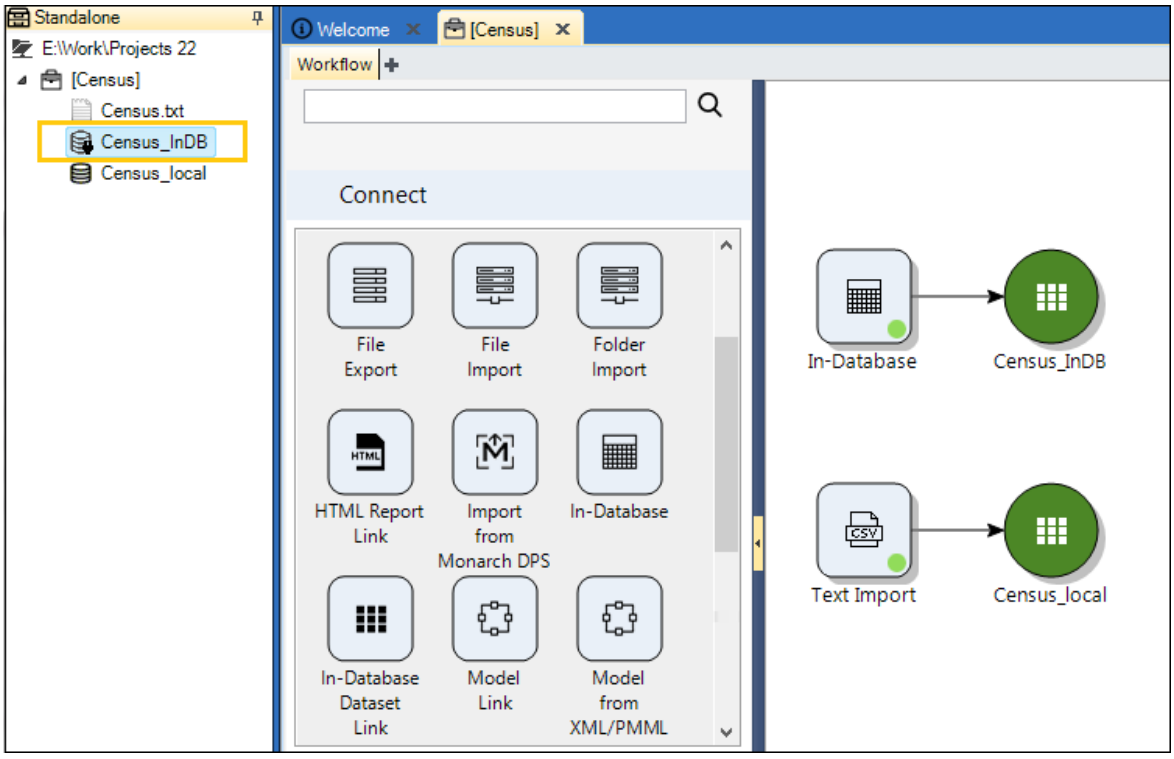
All Data Preparation, Decision Tree and Strategy Tree functionality is supported in the in-database mode, including scoring, validation, and Model Analyzer for Decision Trees. Thus, both in-database model creation and model execution are supported, enabling complete workflow from development to deployment within the database.

In-database datasets can have calculated columns added to or removed from the table allowing for greater flexibility when mining or building models on an in-database dataset. All changes to in-database tables made from within the application are reflected in the database table. See the Dataset Editor topic for more information about calculated columns in the In-Database Analytics Mode.

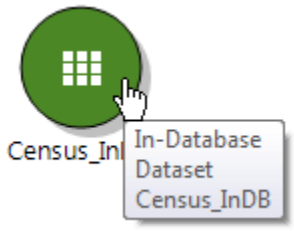
Predictive Models, Cluster Analysis Models, and Principal Component Analysis are not available in the In-Database Analytics mode, since these models usually require a major portion of the data to be imported from the source to the Altair native format. The corresponding menu commands under the Insert menu are disabled for In-Database Analytics datasets. Data Partitioning is also not supported for the In-Database Analytics mode.

To start In-Database Analytics on a table in a database, you must attach to the table using the Insert Dataset Wizard in a way similar to establishing ODBC connection. The table you attached to will be displayed in the Project Explorer as a special dataset represented by an icon with a plug at the bottom-right corner. See the example screenshot below:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



On the canvas, the nodes for all types of datasets look the same, but if you hover your mouse over the dataset node directly linked to a database table, the tooltip text will indicate that it is an "In-Database Dataset", as shown below:



You can perform all the usual data mining tasks with this dataset the same way as you would with the usual dataset. The only difference is that there is no file representing this dataset in your project directory, since no data was imported from the source. The analysis is performed directly on the table in the database.

Note that if you need to open an In-Database Analytics dataset that was inserted in a previous session, you will need to log in to the corresponding database again using your database user credentials.

Note: Because of the nature of In-Database Analytics, this mode has some limitations compared to working with an Altair dataset:

SQL Server columns of type Text are not supported. Use the nvarchar(max) type instead. If the SQL Server table you want to connect to has a Text type column, create a copy of this column of type nvarchar(max).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

In the In-Database Analytics mode for Teradata, Microsoft SQL Server, SQL Server PDW, and Spark, the following statistics in the Overview Report and Expression Editor are not available since there are no standard SQL operations for them: Unique Count, Variance, DevSqr, Mode (Mode1 and Mode2), Lower Quartile, Upper Quartile, Median, Skewness, and Kurtosis.

In the In-Database Analytics mode for the Oracle database, Variance, DevSqr, Mode and Median are supported for numeric and date/time columns only. Unique Count, Lower Quartile, Upper Quartile, Skewness, and Kurtosis are not supported.

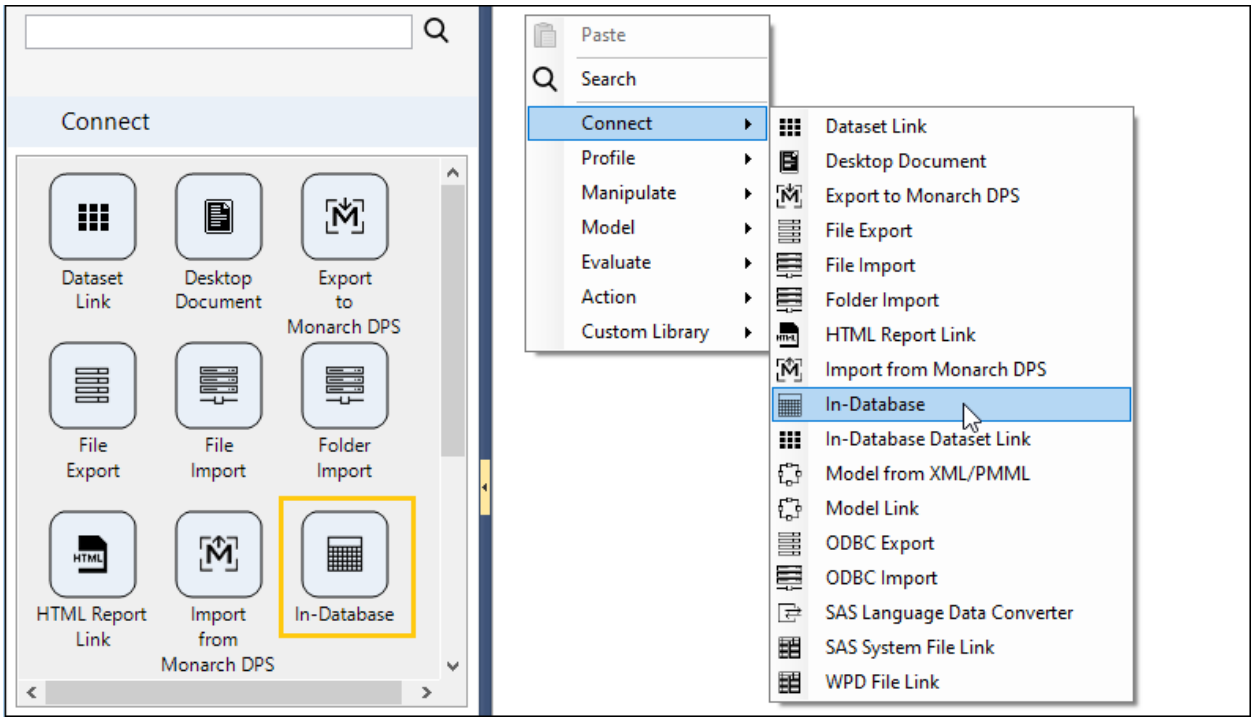
Connecting to a database table for In-Database Analytics

To connect to a database table using the In-Database Analytics connector:

- right-click anywhere on the canvas background and select **Connect --> In-Database** from the context menu

  OR

- drag the **In-Database** node from the **Connect** palette to the Workflow canvas. Then double-click on the new node or right-click and select **Modify**.



The ODBC connection wizard will open. The next steps are exactly the same as in the ODBC Import Wizard. Proceed to the section describing ODBC Connection Setup and follow the subsequent steps.

More information about In-Database scoring and validation can be found in the topics About Scoring and About Validation.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Data Export

Overview

Data from datasets and nodes of decision trees and strategy trees can be exported to files or databases. The supported output file formats are Text, SAS System (*.sas7bdat), Excel, SPSS, R, and KDD (native Altair file format). The destination database can be any database accessible via ODBC.

Data can also be exported to Altair Monarch Data Prep Studio. See Export Dataset to Monarch for details.

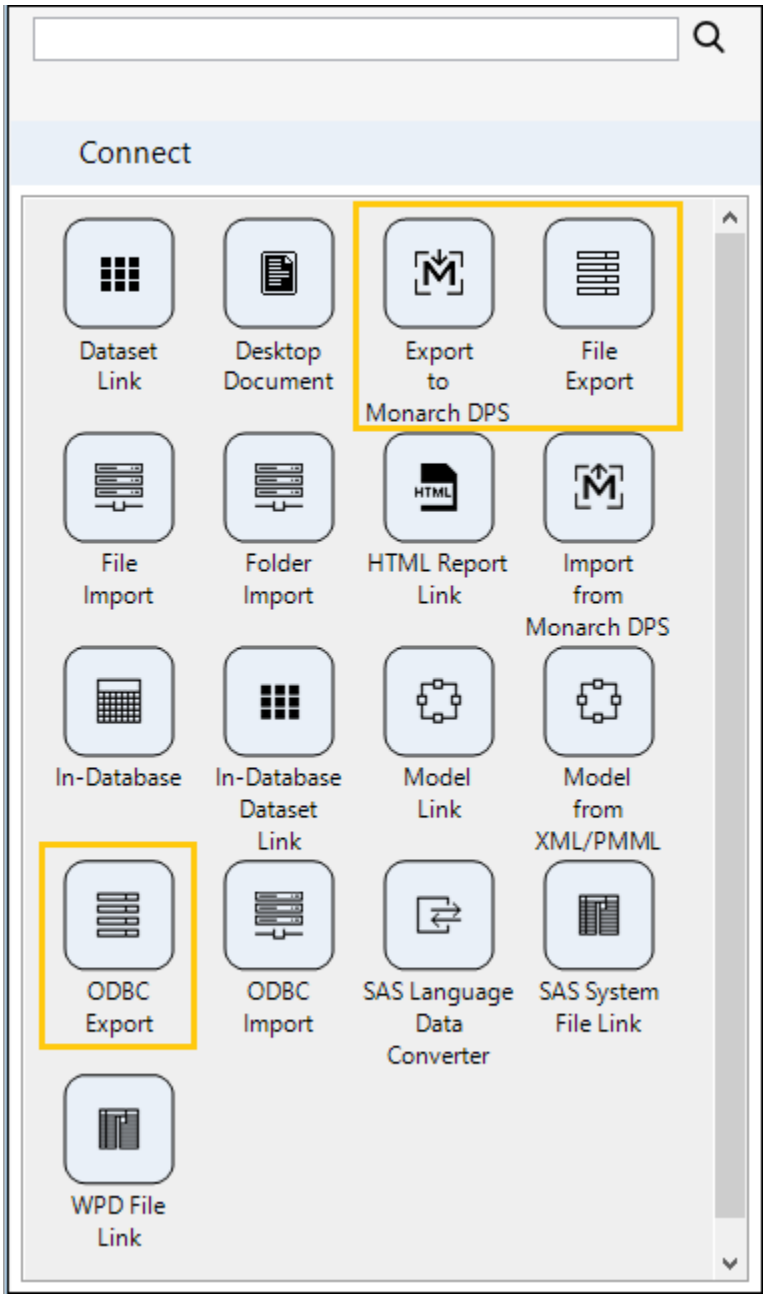
To export data from a dataset to a file or a database:

- Drag a connection arrow from the dataset node to any area on the Workflow canvas. Releasing the mouse button will bring up a context menu. From the context menu, select **Connect --> File Export** (if exporting to a file) or **Connect --> ODBC Export** (if exporting to a database). A new Export process node will be created, and the wizard will open.

OR:

- Depending on the desired target (file or database), drag the **File Export** or the **ODBC Export** node from the **Connect** palette to the workflow canvas and connect the input dataset to the export node. Then double-click on the new node or right-click and select **Modify**.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The following output formats are supported:

- Text
- Microsoft Excel
- R (in ASCII format)
- SAS System
- SPSS Native
- KDD (native Altair file format)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- any database accessible via ODBC

More details on supported database systems are available in the Database Connectivity section of the topic Importing data.

Exporting to Excel, R, SAS System, SPSS, and native Altair file format (KDD) is straightforward. You only need to specify the output file name and the fields to be included.

Note that a dataset exported to the native Altair file format (KDD) is represented by three files (*.ksd, *kdd, and *.ksd-data). For example, exporting a dataset called CreditData produces three files CreditData.ksd, CreditData.kdd, and CreditData.ksd-data. The ksd-data file contains the data, while the kdd and ksd files are for indexing purposes. You can copy these files to a new or existing project folder in your Working Directory and refresh it. The dataset will appear in the project outline when you open the project.

When exporting data as a text file, you can specify format options such as the column delimiter. For details, see the topic Text File Format Options.

Exporting to a database table is covered in the topics ODBC Export Node, ODBC connection setup, and Create Table Statement.

Limitations and Field Naming Conventions

- When exporting data to external files or databases, be aware that the target file format or database system may have field naming conventions different from those in Altair datasets. Please check your field names for consistency with the target format before exporting your data.
- When exporting data as a SAS System, SPSS, or Excel file, the length of the exported field names is limited to 255 characters. Field names longer than 255 characters are truncated for these output formats.

File Export

Overview

Data from datasets, decision tree nodes, and strategy tree nodes can be exported to files in the following file formats:

- Text
- Microsoft Excel
- R (in ASCII format)
- SAS System (*.sas7bdat)
- SPSS Native (*.sav)
- KDD (native Altair file format)

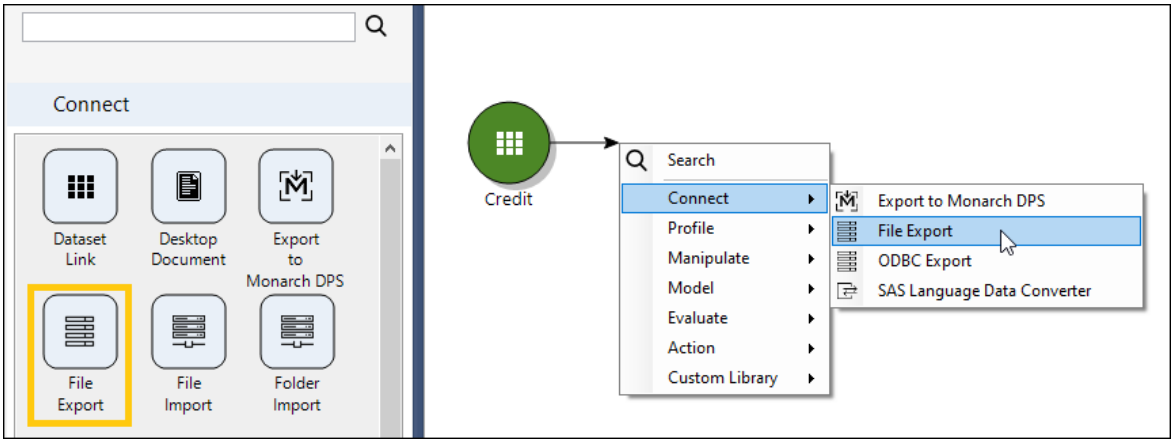
To export data from a dataset to a file:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Drag a connection arrow from the dataset node to any area on the Workflow canvas. Releasing the mouse button will bring up a context menu. From the context menu, select **Connect --> File Export**. A new export node will be created, and the wizard will open.

OR:

- Drag the **File Export** node from the **Connect** palette to the workflow canvas and connect the input dataset to the export node. Then double-click on the File Export node or right-click it and select **Modify**.



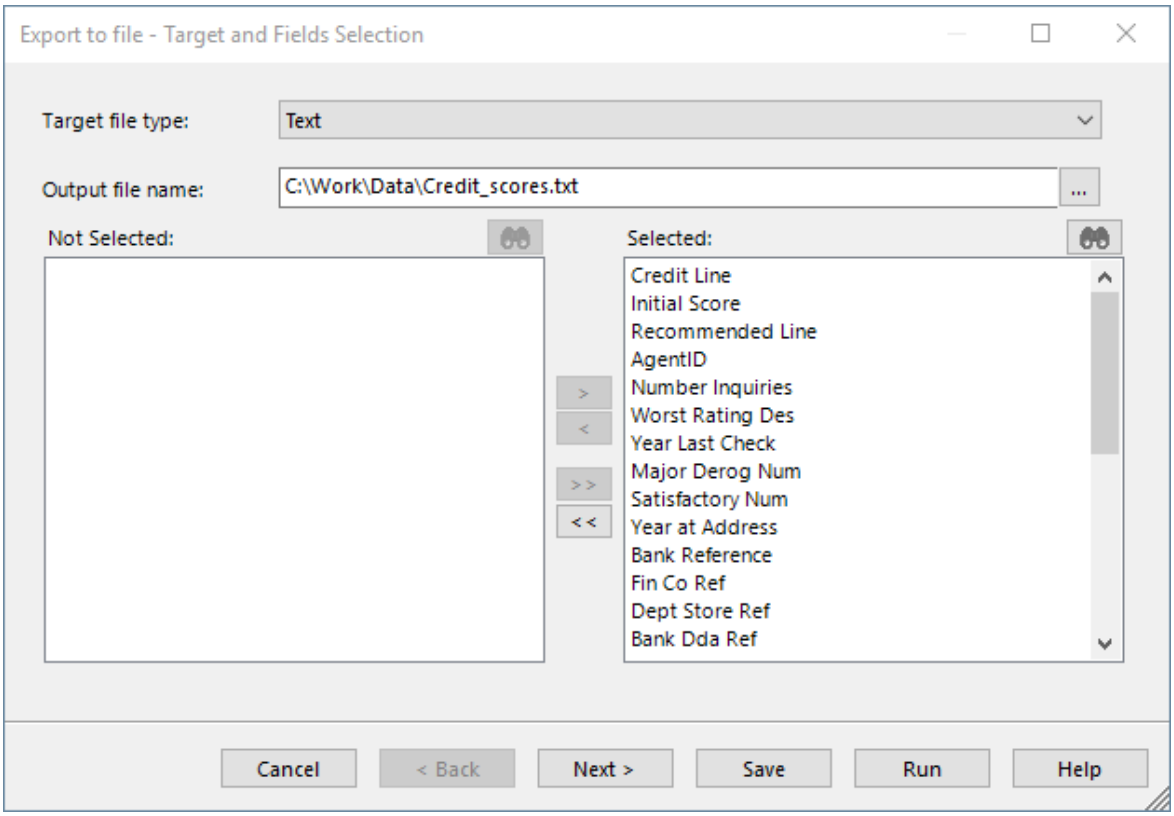
If the File Export node is already on the workflow canvas, double-click it or right-click on it and select **Modify** from the context menu.



In the first page of the wizard, specify the name for the output file. Click the browse button if necessary to specify the desired destination for the output file. If you are working in the client/server mode, note that the file will be created on the server file system.

By default, all fields are selected for export. If necessary, deselect the fields that are not required. To move the fields from one list to the other, highlight the desired fields and use the arrow buttons or double-click on the field names.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If you are exporting data to a text file, click **Next** to specify the delimiter and other format options for the output file. See the topic Text File Format Options for details.

Otherwise, click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.

Limitations and Field Naming Conventions

- The selected file format may have field naming conventions that are different from those in native Altair KDD datasets. Please check the field names for consistency with the target format before exporting your data.
- When exporting data as a SAS System, SPSS, and Excel file, the length of the exported field names is limited to 255 characters. Field names longer than 255 characters are truncated for these output formats.

See also:

Text File Format Options

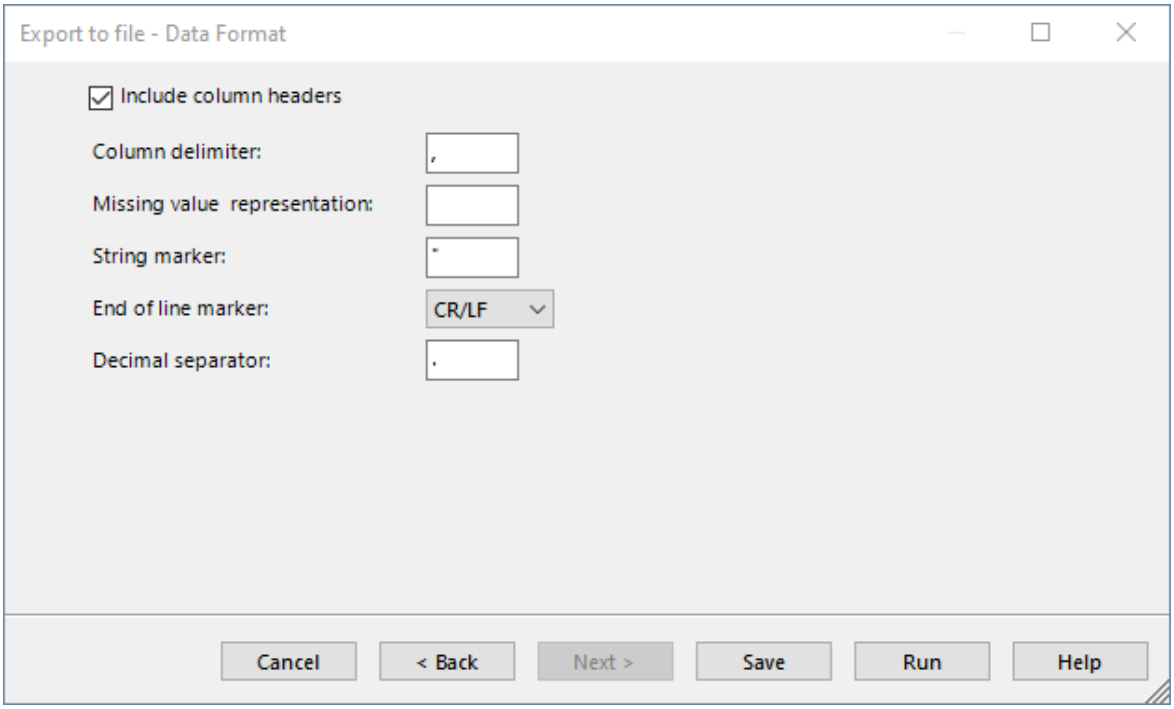
Data Export (Overview)

Text File Format Options

When exporting data to a text file, the last page of the File Export wizard allows you to specify format options for the output file:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- include or exclude column headers (included by default)
- column delimiter (the default is comma; to specify tab as delimiter, enter \t, \T, tab, or TAB in this field)
- missing value representation (the default is a blank value)
- string marker (the default is double quotes)
- end of line marker (the default is CR/LF, which is a standard end of line marker on Windows; If the output text file is to be written to a Linux or Unix system, select the LF option)
- decimal separator (the default is based on the numeric format in the current regioinal settings)



Data is exported to text in the UTF-8 encoding.

See also:

Data Export (Overview)
Target File and Field Selection

Export to Databases (ODBC)

ODBC Export Node

Data from datasets, decision tree nodes, and strategy tree nodes can be exported to databases. The destination database can be any database accessible via ODBC.

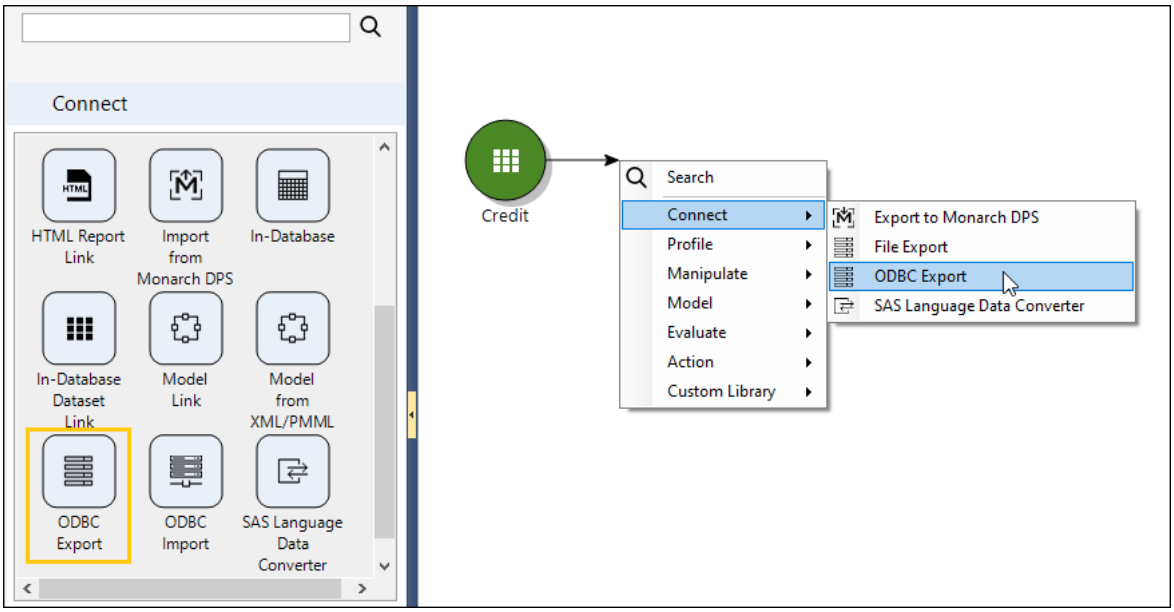
To export data from a dataset to a database table:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

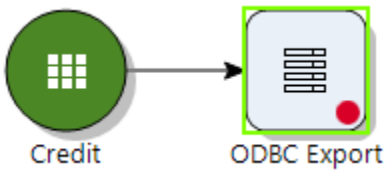
- Drag a connection arrow from the dataset node to any area on the Workflow canvas. Releasing the mouse button will bring up a context menu. From the context menu, select **Connect --> ODBC Export**. A new ODBC Export node will be created, and the wizard will open.

OR:

- Drag the **ODBC Export** node from the **Connect** palette to the workflow canvas and connect the input dataset to the export node. Then double-click on the ODBC Export node or right-click it and select **Modify**.



If the ODBC Export node is already on the workflow canvas, double-click it or right-click on it and select **Modify** from the context menu to open the wizard.



The first page of the ODBC Export node wizard is the same as the ODBC connection setup for ODBC Import: It allows you to specify the connection parameters and connect to the target database. See the ODBC connection setup topic for details.

ODBC Connection Setup

The ODBC Connection dialog provides two ways to set up the ODBC connection:

- Specifying an ODBC connection string
- Selecting a DSN (Data Source Name)

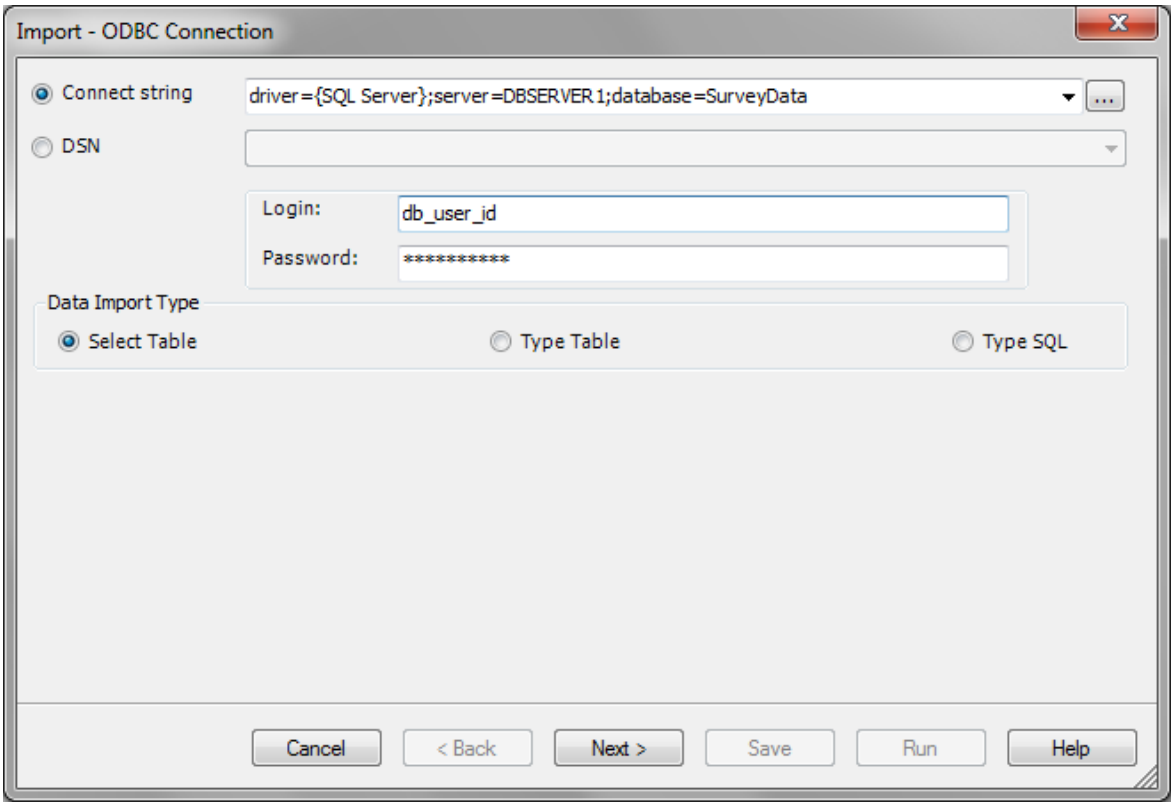
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

It also allows you to choose whether you would like to type a table name (or a view name), or type a full SQL statement to be passed through to the database during import. This is helpful when you have a large number or tables or when you want something more complex than a simple table or view selection.

The Connect String field keeps the list of most recently used ODBC connect strings for your convenience. If you want to delete one of the saved connect strings from the list, just highlight it while the list is open and press the Delete key.

To make sure that data in specific languages is imported properly, you may need to include the following statements to the ODBC Connection String: "CHARSET=UTF-8" or "FWC=T" (Force SQL_WCHAR Support Enabled). Ask your Database Administrator for assistance if necessary.

Note: When using ODBC import in the client/server configuration for Linux, text field values exceeding 255 characters in length are not imported unless the default length of the Varchar fields in the source database is increased from the 255 default to a value equal to or greater than the maximum field length.



Note: If your ODBC data source is an Excel file, make sure the file is closed before starting the import. No records will be imported if the file is in use.

1. The Connect String Option

The Connect String option allows you to

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- enter the ODBC connection string manually, or
- construct the ODBC connection string using the ODBC Connection Information dialog, which is opened by clicking on the browse button to the right of the Connect String field. It prompts the user for the data source type (Oracle, MS SQL Server, etc.) and connection parameters depending on the selected data source type. (*Note: this dialog is not available in client/server mode with server component on Unix.*)

Example connection strings for Windows:

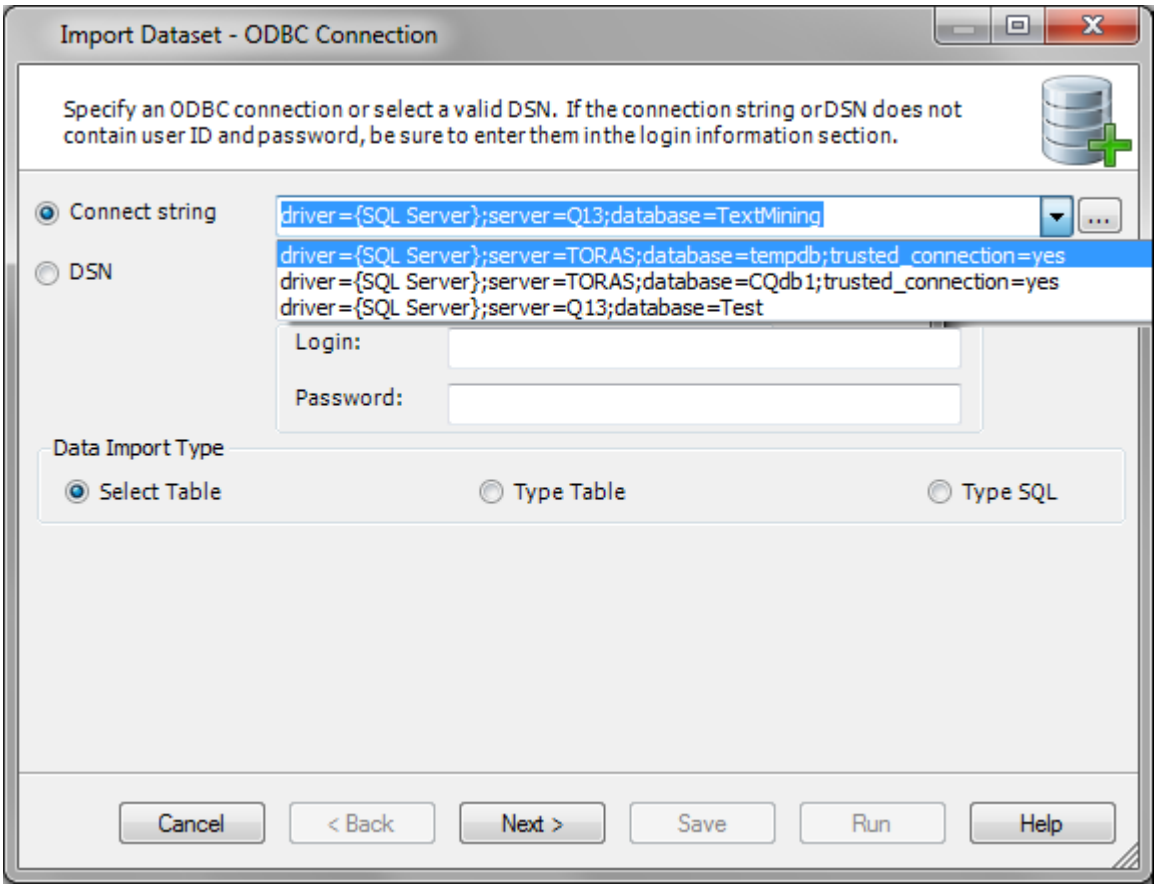
SQL Server	Driver={SQL Server};Server=*<server name>*;Database=<*database_name>*;UID=*<login>*;Pwd=*<password>;*
Oracle	Driver={Oracle in OraHome92};DBQ=*<TNS Service Name>*;UID=*<login>*;Pwd=*<password>;*

If you include the login ID and password in the Connection String, you do not need to specify them in the Login and Password fields in the lower part of the ODBC Connection dialog. Otherwise, the Login and Password fields must be filled in.

The ODBC connection string may require multiple options to be specified. For example, for an Oracle database whose character set supports languages such as Japanese, Chinese, etc., SHIFT-JIS, Unicode, UTF8), the option FWC=T (Force SQL_WCHAR Support Enabled) must be included in the connect string in order for all the characters to be displayed correctly. If you are unsure of how to proceed, ask your System Administrator for the required information.

A list of recently used connect strings in the Connect String drop-down list provides an easy way to connect to most frequently used databases.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



ODBC Connection Information dialog.

The ODBC Connection Information dialog helps you to build the ODBC connection string.

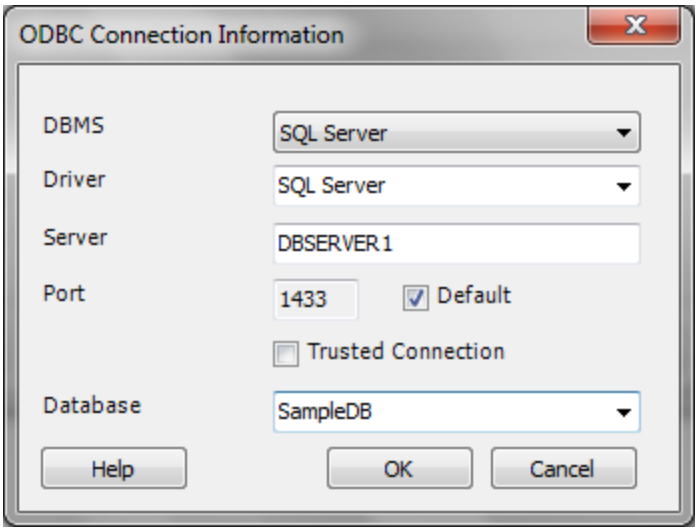
- Select your data source type from the DBMS drop-down list.

Note: This list contains only the most commonly used databases. If the required database type is not listed, use the DSN option rather than the Connect string option (see below).

- Choose the database driver. If there is more than one driver available, make sure you select the driver of the right version, especially in the case of Microsoft Access or Excel.
- The remaining parameters to be specified in this dialog are specific to each data source type. Usually, they include the database host name and the database name. Other parameters may include the database port number, authentication type, etc. If you are unsure of how to proceed, ask your System Administrator for the required information.

Example: The case of a SQL Server database:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



2. The DSN (Data Source Name) Option

The DSN option can be used to connect to a data source if you already have a DSN on your system that is configured to point to this data source.

When the DSN option is selected, all Data Source Names defined on your system become available in the DSN drop-down list. Open the drop-down list and select the desired DSN.

On Windows, Data Source Names are defined in the standard ODBC Data Source Administrator (in Administrative Tools). On Unix, Data Source Names are defined in the `odbc.ini` file. If you need to create or configure a DSN and are unsure of how to proceed, ask your System Administrator or refer to your data source application's documentation.

After specifying the ODBC connection properties, click Next.

- If you are **importing** data or using the In-Database connector, the next step depends on the Data Import Type you specified.
  - If you chose either the Select Table or Type Table option, proceed to the Table Selection step.
  - If you chose the Type SQL option, the next step is to specify the dataset name and a SQL select statement for retrieving data. Proceed to the Dataset Name step.
- If you are **exporting** data, proceed to the Table Selection step.

See Also:

Import from ODBC data sources

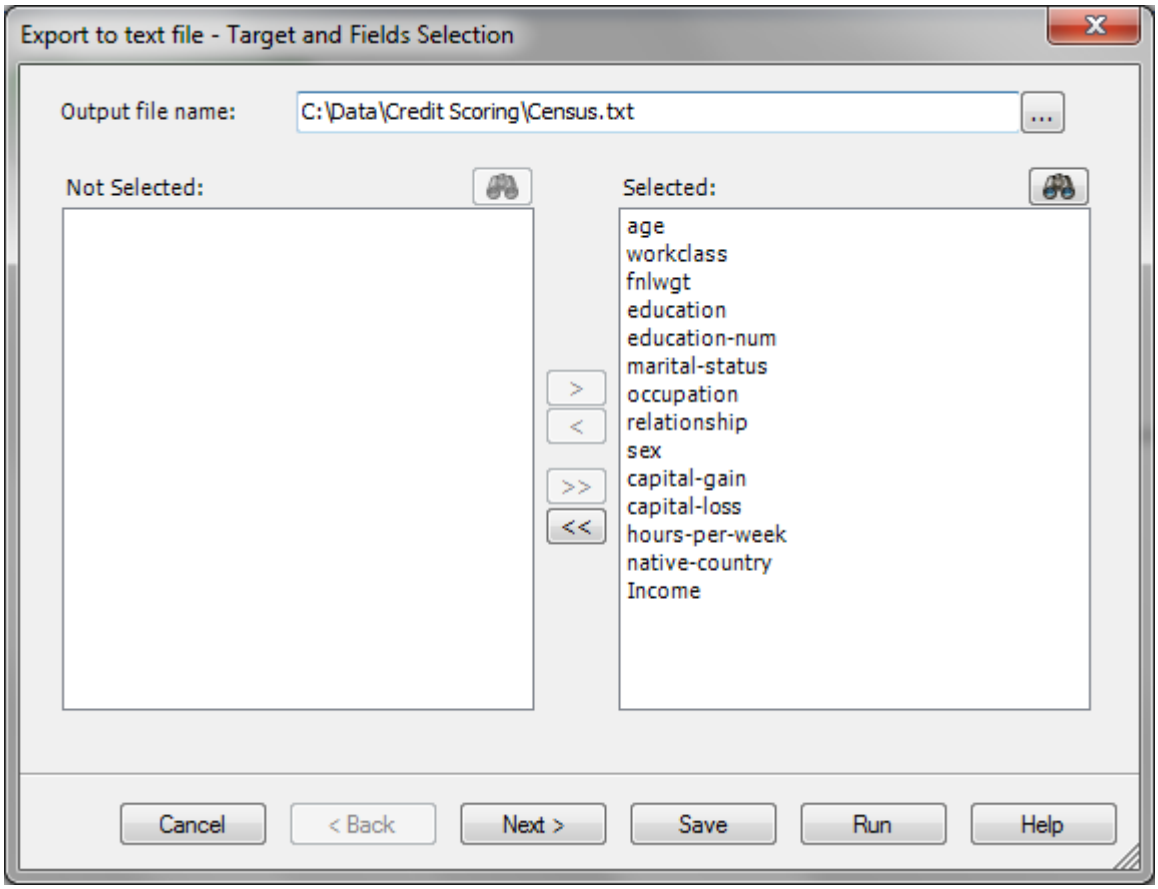
Field Selection

Select the fields to be included in the output.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If you are exporting a dataset, all fields displayed in the Data tab are selected by default. If you are exporting data from a tree node, all fields displayed in the Node Data tab are selected by default.

To move the fields from one list to the other, highlight the desired fields and use the arrow buttons or double-click on the field names.



If you are exporting data to a text file, click Next to specify the delimiter and other options. See the topic Text File Format Options for details.

If you are exporting data to a database table via ODBC, click Next to preview and edit the CREATE TABLE statement. See the topic Create Table Statement for details.

Limitations and Field Naming Conventions

When exporting data from to external files or databases, be aware that the target file format or database system may have field naming conventions different from those in Altair KDD datasets. Please check your field names for consistency with the target format before exporting your data.

Also, please note that the export drivers for SAS System, SPSS, and Excel file formats limit the length the exported field names to 255 characters. Field names longer than 255 characters are truncated by these drivers.

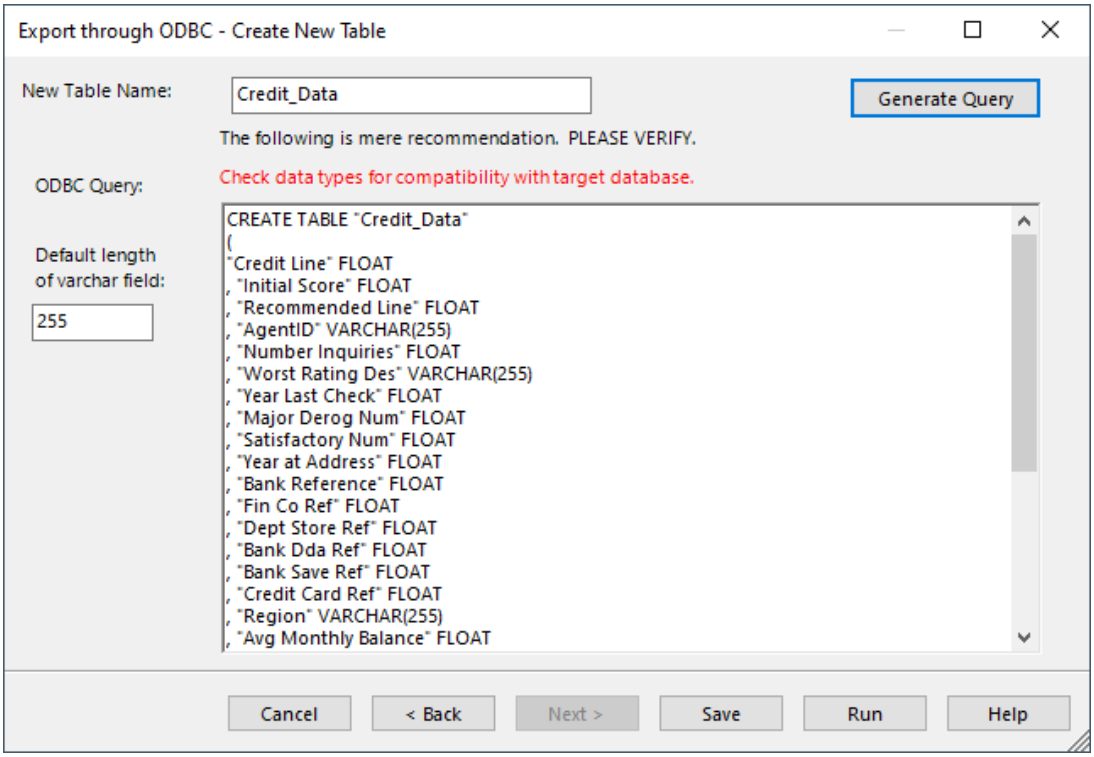
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

See also: Data Export (Overview)

CREATE TABLE Statement

This step allows you to preview and edit the automatically generated CREATE TABLE statement to be passed to the ODBC driver, which will perform the data export.

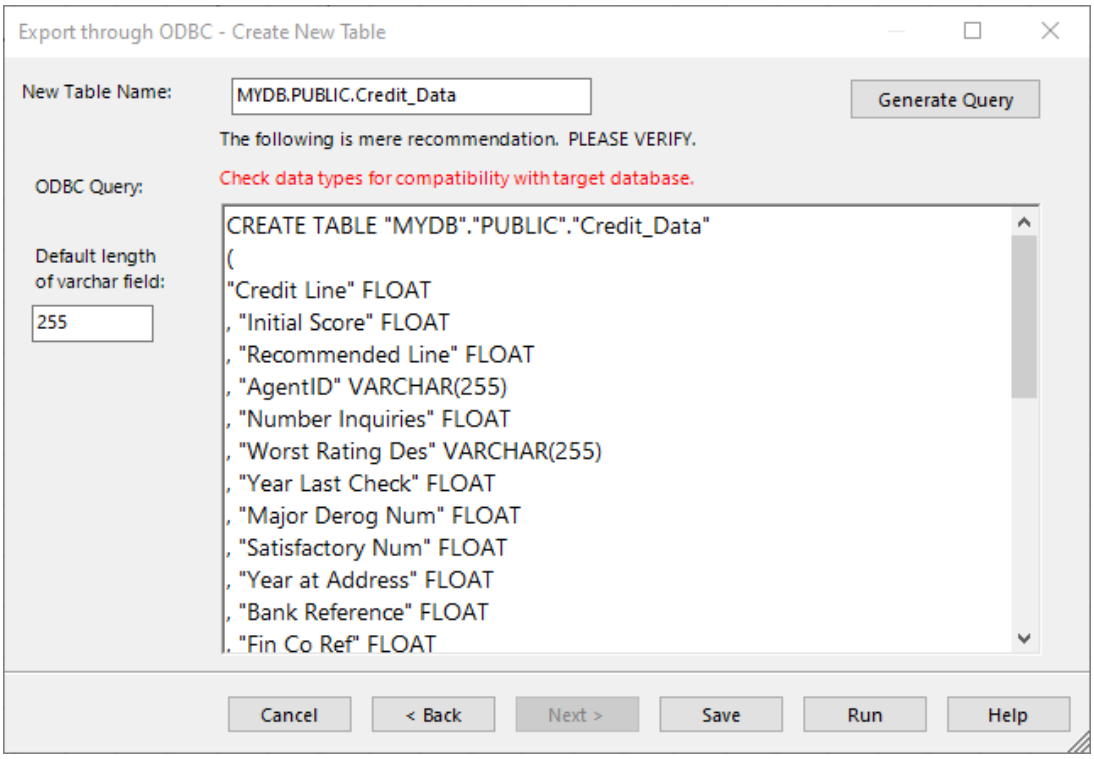
- Specify the name for the new table. After modifying the table name supplied by default, click the **Preview Query** button to update the CREATE TABLE statement in the *ODBC Query* frame.



  - Depending on the target database system, you may need to specify a *fully qualified table name*, which includes a database name and a schema name. For example, for the Snowflake database, if you intend to export data to a database MYDB under the PUBLIC schema, specify the fully qualified table name like this: `MYDB.PUBLIC.my_new_table`.

- Review the automatically generated CREATE TABLE statement in the ODBC Query frame. The syntax of the statement follows the SQL Server standards (Transact-SQL). When exporting data to Oracle, DB2, or any database system other than SQL Server, you may need to edit the statement to make it compatible with the SQL syntax of the target database system. For example, numeric, date, and other data types may have different names in different database systems.
  - For example, if the target database platform is Snowflake, the table name in the CREATE TABLE statement must be a *fully qualified table name*, where all three parts must be enclosed in double quotes. For example, if the data is to be exported to a database MYDB under the PUBLIC schema, the statement must start as follows: `CREATE TABLE "MYDB"."PUBLIC"."my_new_table"`. See the example screenshot below.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- The default length of String (varchar) fields is 255. It is applied to all String fields in the auto-generated statement. Modify it if you know the maximum length of String fields in your dataset, then update the statement by clicking **Generate Query**. You can also edit the statement to specify the length for each string field.

After making sure the syntax of the statement is compatible with the target database system, click **Save** to save the settings or click **Run** to save the settings and run the process immediately.

See also:

Data Export (Overview)

Import from ODBC data sources

Export Dataset to Monarch DPS

Datasets can be exported to **Altair Monarch Data Prep Studio** as delimited text files.

Prerequisite: Altair Monarch version 2020 or higher must be installed.

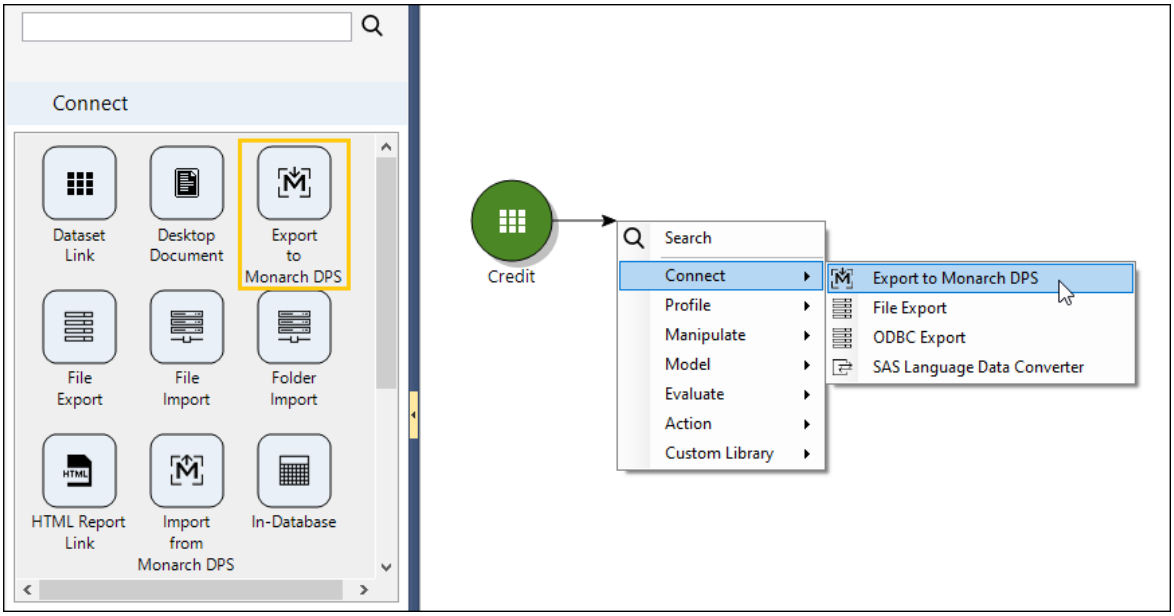
To export data from a dataset to Altair Monarch Data Prep Studio:

- Drag a connection arrow from the dataset node to any area on the Workflow canvas. Releasing the mouse button will bring up the context menu. From the context menu, select **Connect --> Export to Monarch DPS.** See the example screenshot below. A new node **Export to Monarch DPS** will be created, and the wizard will open.

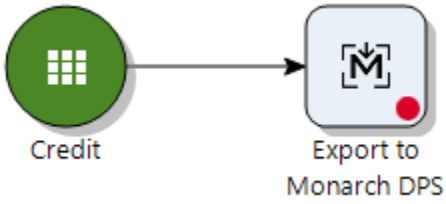
OR:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Drag the node **Export to Monarch DPS** from the **Connect** palette to the canvas and connect the input dataset to the Export node. Then double-click on the new node or right-click on it and select **Modify**.

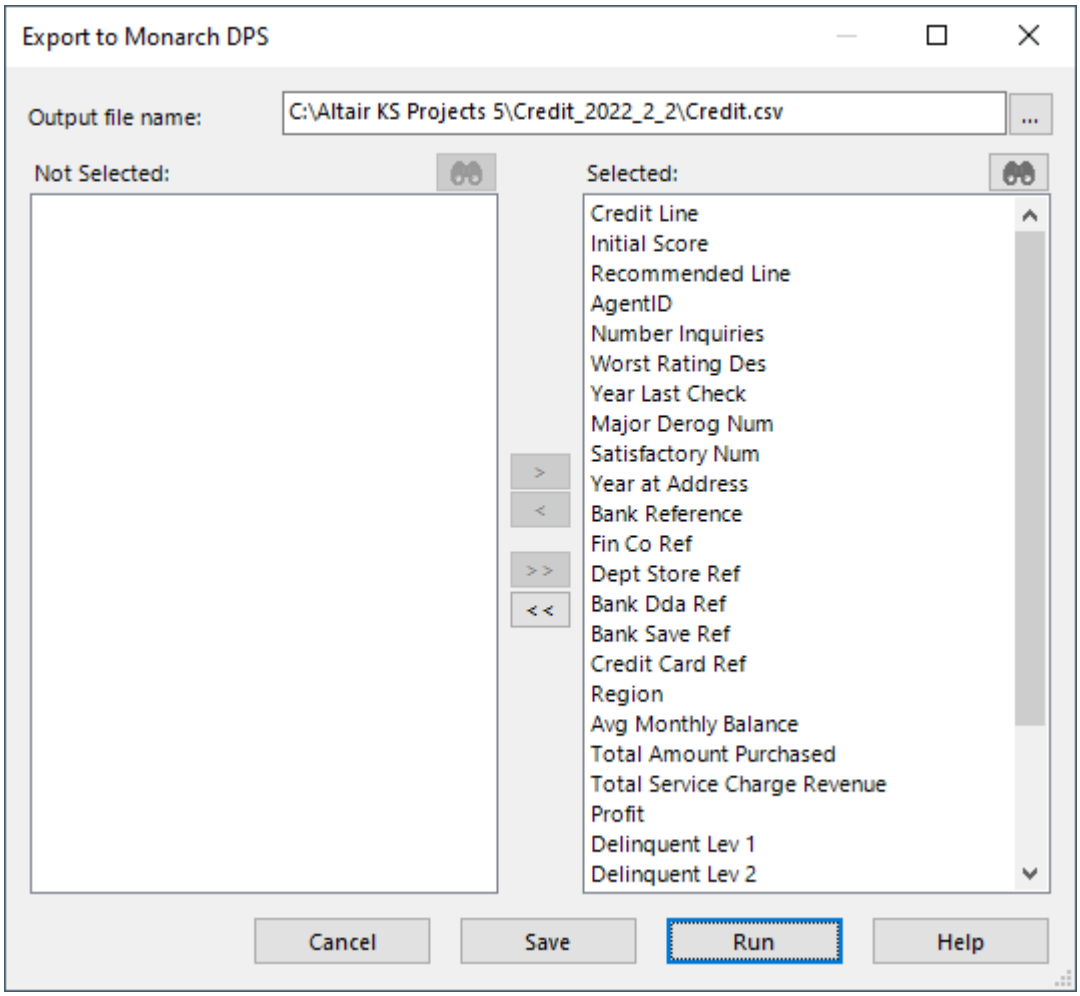


If the **Export to Monarch DPS** node is already on the canvas, double-click it or right-click on it and select **Modify** from the context menu to open the wizard.



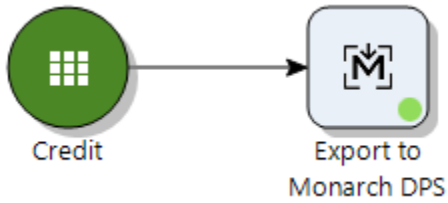
Select the fields to export. By default, all fields are selected. The fields can be included or excluded using the arrow buttons or by double-clicking on individual field names. You can also use the Field Finder (binocular buttons) to find a particular field among hundreds of fields.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



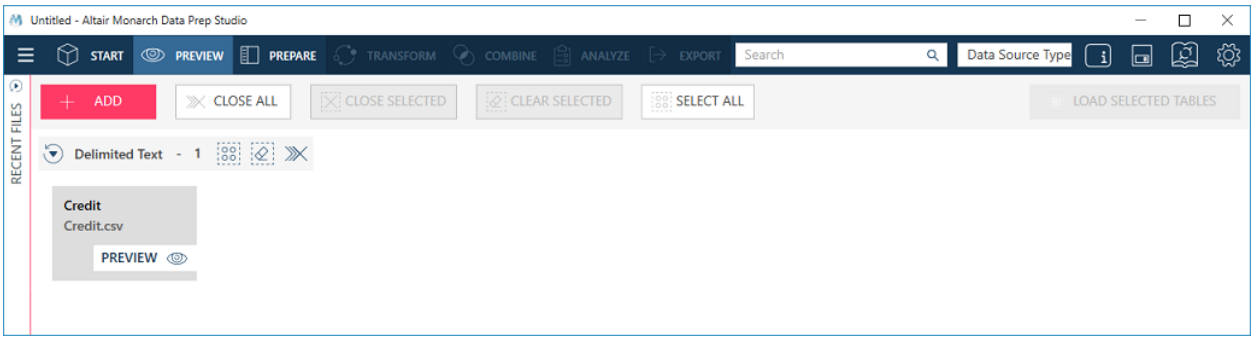
When exporting data to Monarch, check your field names for consistency with the field naming conventions in Monarch.

Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.

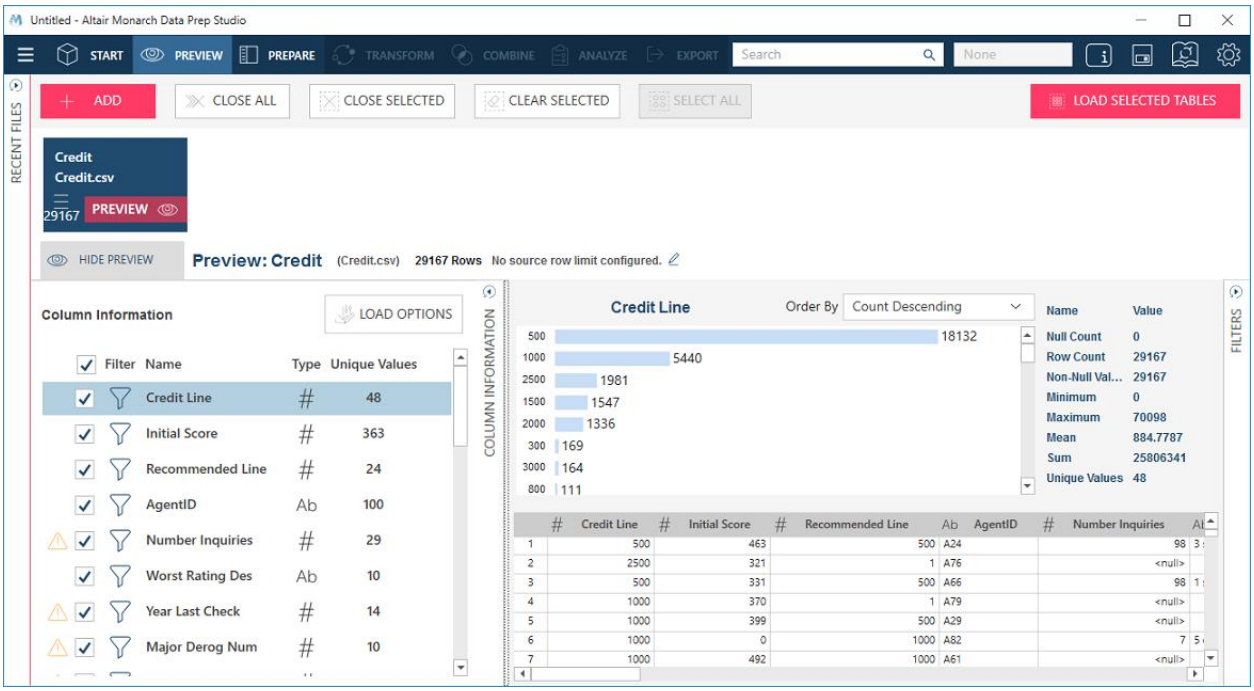


Once the process is finished, double-click the **Export to Monarch DPS** node. This will launch Monarch Data Prep Studio if it installed and licensed. The exported dataset in the csv format is shown in the **Preview** tab:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Preview** on the exported CSV dataset icon in the Monarch DPS window.



Note: If any numeric variable in the source dataset contains infinities or NaN values, they are converted to NULLs (missing values) in the exported data. (For information about infinities and NaN values, see Division and Infinity)

Desktop Document and Report Links

If you would like to add project documentation in an arbitrary format (PDF, Microsoft Word, Excel, plain text, etc.) and embed your document within the project workflow, you can do that using the **Desktop Document** node.

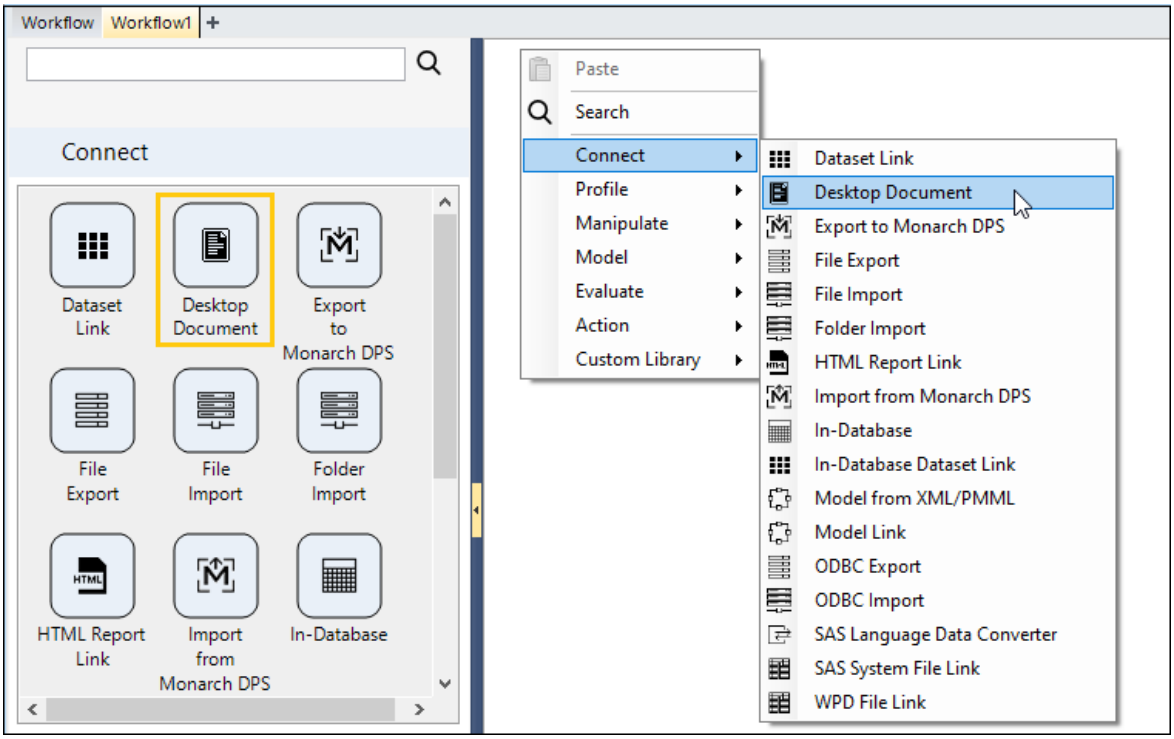
To add any document to the workflow of the current project:

- Right-click anywhere on the Workflow canvas background. From the context menu, select **Connect --> Desktop Document** as shown in the example below. A new **Desktop Document** node will be created and the Link Document dialog will open.

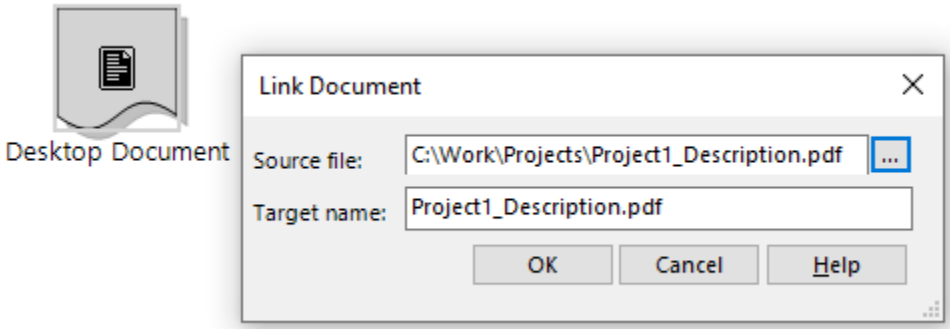
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

OR:

- Drag the **Desktop Document** node from the **Connect** palette onto the canvas. Then double-click the node or right-click on it and select **Link** from the context menu.



- Browse for the desired file and specify the file path in the *Source file* field.
- Specify the name for the document node on the workflow canvas.

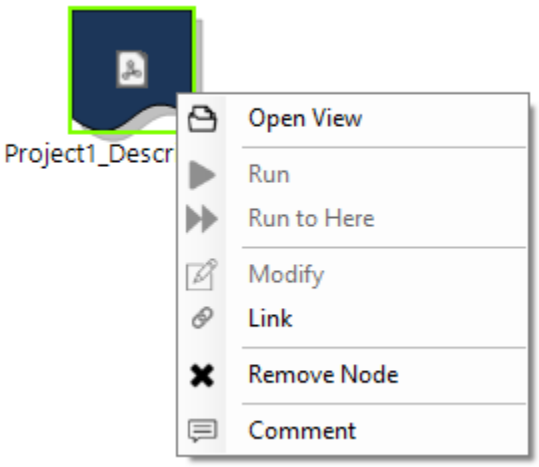


- Click **OK**. The document file will be copied to the current project folder, shown in the Project Explorer, and linked as a new Desktop Document node on the Workflow canvas.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Double-click the document node to open it or right-click on it and select **Open View**. The document will be opened by the program currently associated with its file extension. If Windows cannot find a program associated with this file extension, it will display a standard dialog prompting you to select an application for opening this file type.



You can also link a different document file to this node at any time by selecting the **Link** command from the context menu of the node. In this case the previously linked document file will still remain in the project folder, but the node will now be linked to the new file, which will also be copied to the project folder.

If you would like to add project documentation in the HTML format and embed an HTML file within the project workflow, you can do that using the **HTML Report Link** node.

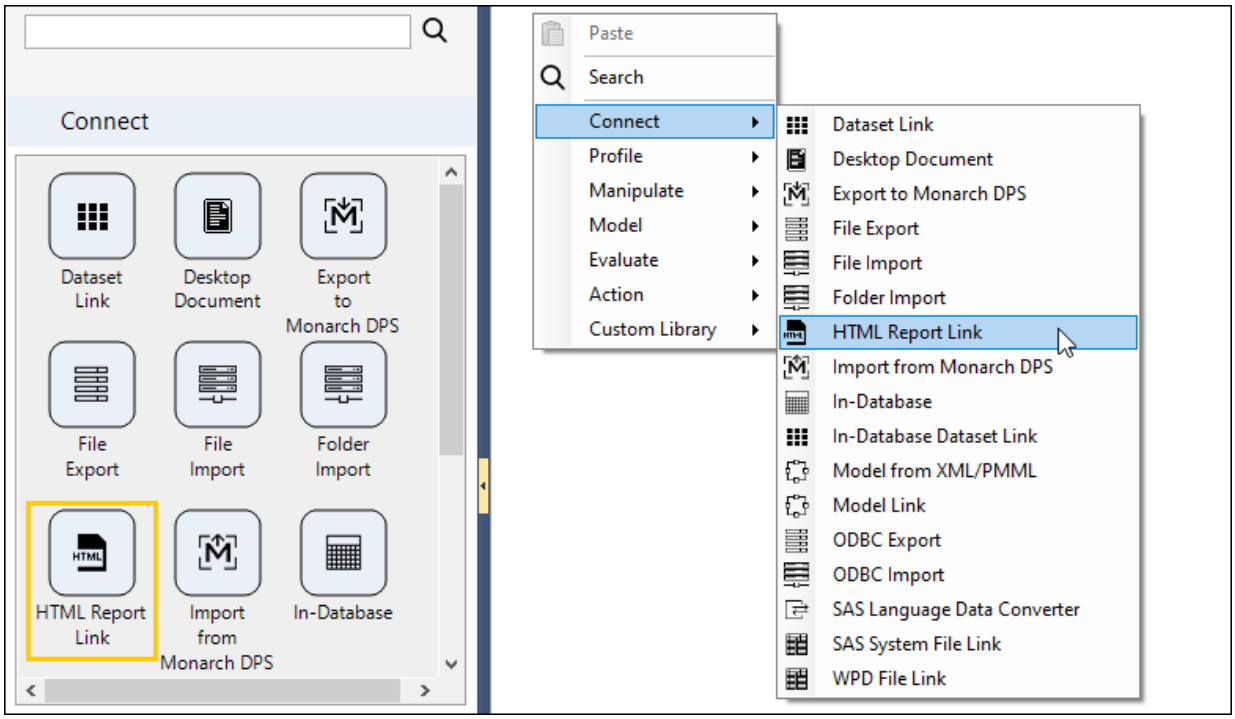
To add an HTML report to the workflow of the current project:

- Right-click anywhere on the Workflow canvas background. From the context menu, select **Connect --> HTML Report Link** as shown in the example below. A new **HTML Report Link** node will be created and the Link HTML File dialog will open.

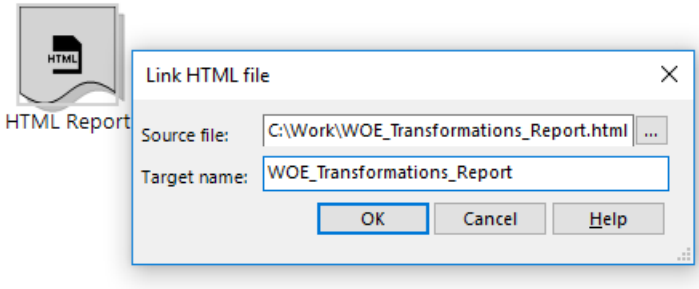
OR:

- Drag the **HTML Report Link** node from the **Connect** palette onto the canvas. Then double-click the node or right-click on it and select **Link** from the context menu.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- Browse for the source HTML file and specify the file path in the *Source file* field.
- Specify the name for the report node on the workflow canvas.

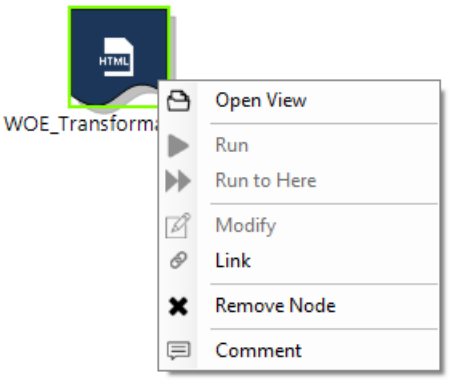


- Click **OK**. The source HTML file will be copied to the current project folder, shown in the Project Explorer, and linked as a new HTML Report node on the Workflow canvas.



Double-click the report node to open the report or right-click on it and select **Open View**. You can also link a different HTML file to this node by selecting the **Link** command. In this case the previously linked HTML file will still remain in the project folder, but the node on the workflow will now be linked to the new HTML file, which will also be copied to the project folder.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



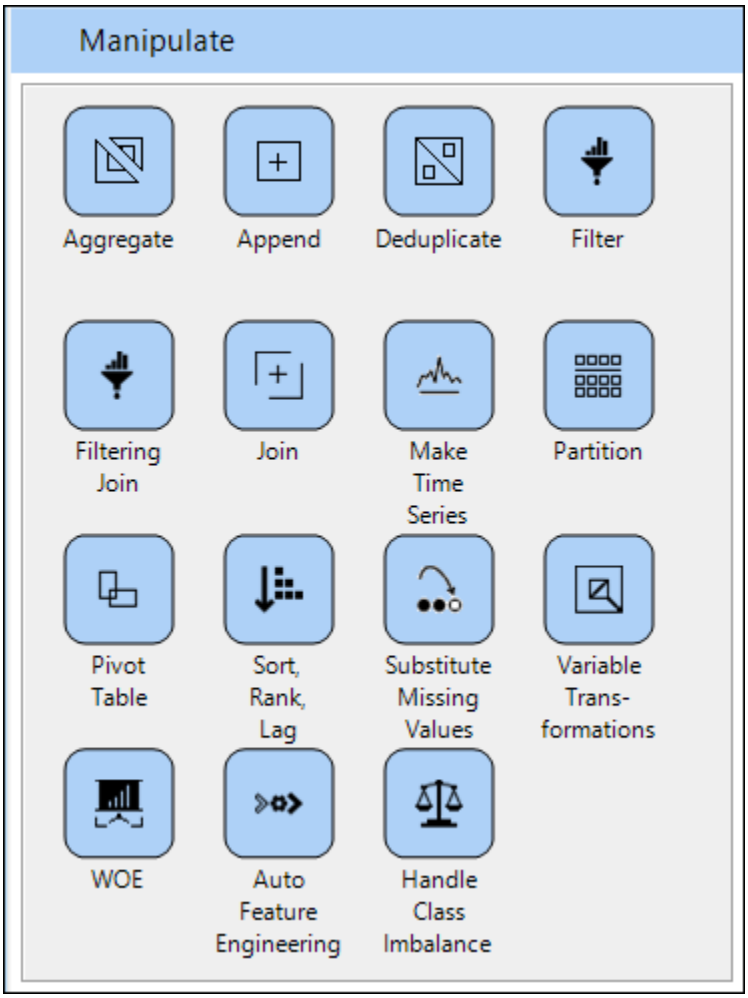
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Data Manipulation

Overview

The functions available in the **Manipulate** palette help you prepare the data for analysis if you have not already done so in the data source before importing it.

The **Manipulate** palette includes the following nodes:



Note: The **WOE** node is available in Knowledge Studio, but not in Knowledge Seeker.

Dataset-level manipulation operations:

- Aggregating data - Group records on one or more attributes of a dataset and replace the other attributes with summary statistics based on the resulting groups of records.
- Appending datasets - Merge two datasets by adding all records from one dataset to the end of the other dataset.
- Removing duplicate records

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Filter - Select or delete the records satisfying user-defined conditions, producing a new, filtered dataset.
- Filtering Join - Joins two datasets to produce a new one containing the records from the first input dataset that either match (*semi-join*) or do not match (*anti-join*) in the second input dataset.
- Joining datasets - Creating a new dataset with fields from two source datasets in your project.
- Make Time Series - Data preparation for Time Series Analysis: Produces time series datasets containing metadata that identifies one column as the time index and another column as the time label, aggregated over uder-defined time units.
- Partitioning
- Pivot Table - Rearranging and summarizing the data in Altair datasets to prepare it for modelling. The Pivot Table node wizard creates a new dataset with the user-defined grouping and summary statistics.
- Sort, Rank, Lag - Sorting data, creating ranking fields and lagged variables.
- Auto Feature Engineering - Automatically creates new features (variables) using transformations and aggregations of different types from two or more input datasets and makes them into a new dataset.

Data manipulation at the variable level:

- Variable Transformations
- Handle Class Imbalance - Balances out the classes of a dependent variable with skewed distribution via oversampling the minority class by creating artificial data points (synthesizing data).
- Substitute Missing Values - Performs substitution of missing values for multiple variables at a time.
- Weight of Evidence (WOE) transformation - Coarse-classing variables using Weight of Evidence. Binning transformations commonly used as one of the data preparation stages in creating scorecards.

All data preparation operations create new datasets without modifying the input datasets. For example, if you apply the **Deduplicate** node to a dataset, its records will not be affected, but a copy of the input dataset without duplicate records will be created.

Dataset Operations

Aggregating Data

Data aggregation is used to group records on one or more attributes and replace the other attributes with summary statistics based on the resulting groups of records.

A common example is preparing data for building a model of customer behavior when all you have available is transactional data from a point-of-sale system. The POS data typically has one record for each transaction. In order to model at the customer level you will need to aggregate this data to have only a single record per customer, with attributes containing aggregate information about their transactions. This aggregate data could contain things like the total number of transactions, total amount purchased, or average purchase size per customer.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

In the example below, records with the same customer ID (loyalty card ID) need to be aggregated. For each customer, the following summary statistics needs to be calculated: the total number of purchased items, the total number of transactions, the preferred payment method (i.e., most frequent payment method).

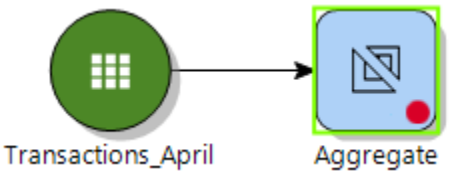
	transaction_id	card_id	payment_method	timestamp	product_code	items_number	items_amount
1	145419	100000111	Cash	19/05/2001	000000078	1	89
2	121302	100000111	Cash	27/04/2001	000000284	1	980
3	51466	100000111	Cash	20/02/2001	000000488	1	69
4	143150	100000111	Cash	17/05/2001	000000595	1	299
5	51036	100000111	Cash	20/02/2001	000000622	1	69
6	205287	100000111	CreditCard	19/07/2001	000000808	1	3599
7	54688	101213614	CreditCard	23/02/2001	000000049	1	69
8	327653	101213614	CreditCard	07/12/2001	000000061	1	69
9	95799	101213614	CreditCard	03/04/2001	000000064	1	79
10	91569	101213614	CreditCard	30/03/2001	000000066	1	69
11	127498	101213614	CreditCard	02/05/2001	000000079	1	89
12	161202	101213614	CreditCard	03/06/2001	000000084	1	89
13	54689	101213614	CreditCard	23/02/2001	000000084	2	178

The resulting dataset will have one record per customer ID, and three fields for the required summary statistics:

	card_id	Total items purchased	Total transactions	Preferred payment method
1	100000111	5105	6	Cash
2	100000199	3137	3	CreditCard
3	100000343	726	4	CreditCard
4	100000375	366	4	CreditCard
5	100000392	8669	31	DebitCard
6	100000444	3537	22	DebitCard
7	100000462	1054	3	DebitCard
8	100000480	1369	4	CreditCard

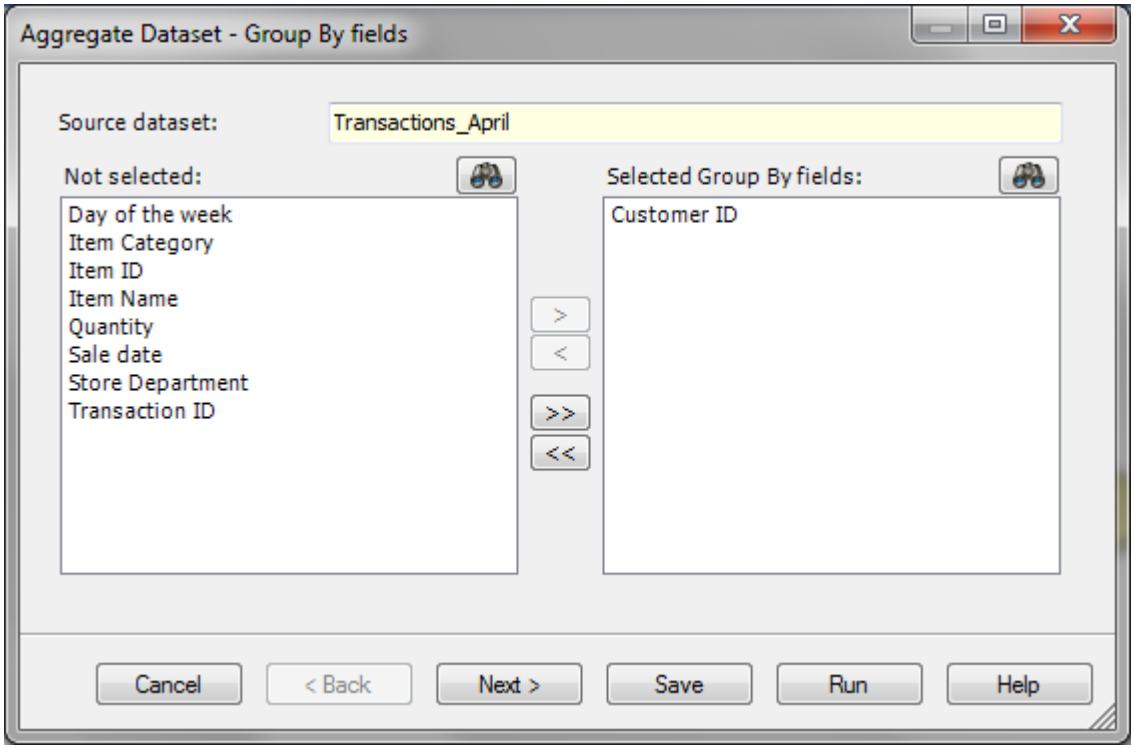
The Aggregation Operation

To introduce a data aggregation operation into your workflow, drag the **Aggregate** node from the **Manipulate** palette onto the canvas and connect the input dataset to the node.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To specify the conditions, double-click the Aggregate node or right-click on it and select **Modify**. In the first page of the wizard, select the fields you need to group by. These are the fields representing the attributes you need to generate summary statistics for. Seleting group-by fields is optional.



Click Next. In the *Aggregate fields* page, specify the name for the output (aggregated) dataset. The default name is formed by adding the suffix *_aggregated* to the name of the source dataset.

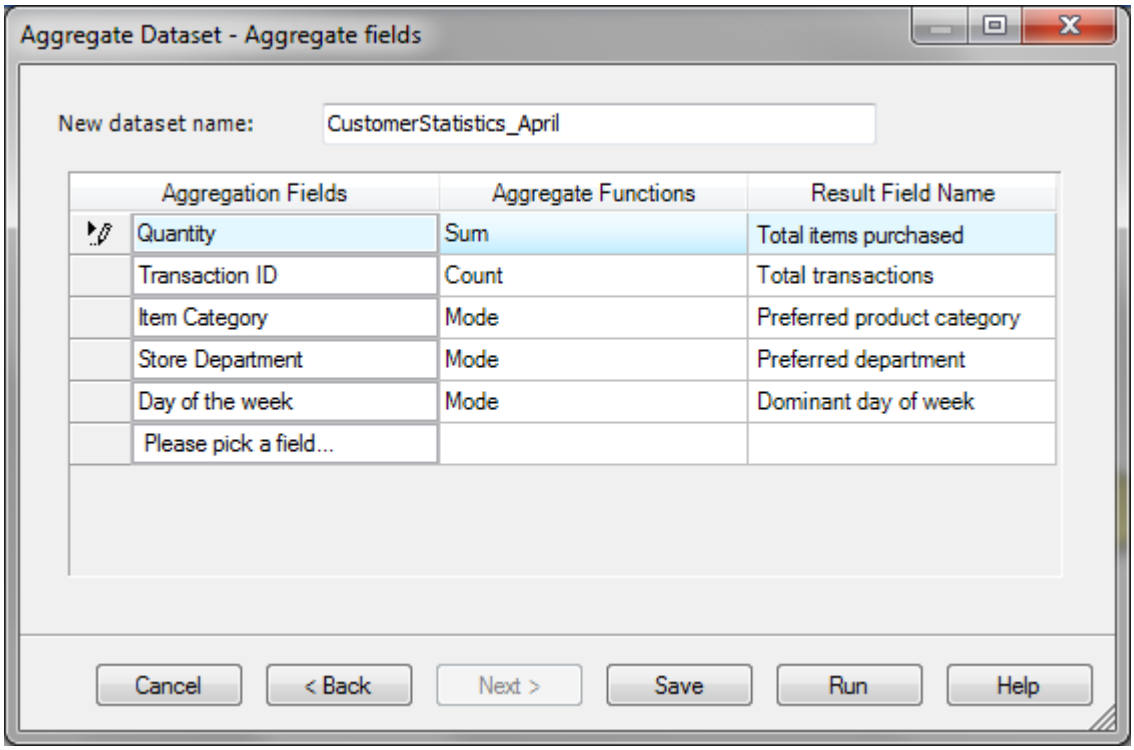
Then select the aggregation fields, aggregate functions to be applied to each aggregation field, and specify the name for the resulting summary statistics fields.

The aggregate functions available for numeric fields are: *Sum*, *Average*, *Count*, *Maximum, Minimum, Mode, Median, Variance, and Standard Deviation*. For other data types, available aggregate functions are: *Count, Maximum, Minimum, Mode, and Median*.

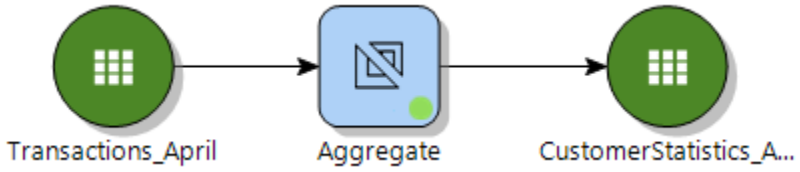
Note that *Count, Variance, Mode* and *Median* are not available in the In-Database Analytics mode, since the supported database systems have no standard SQL operations for them.

In the example below, the resulting field *Total items purchased* will represent the total quantity of items purchased - SUM([Quantity]) - for each individual Customer ID (the Group By field selected at the previous step).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

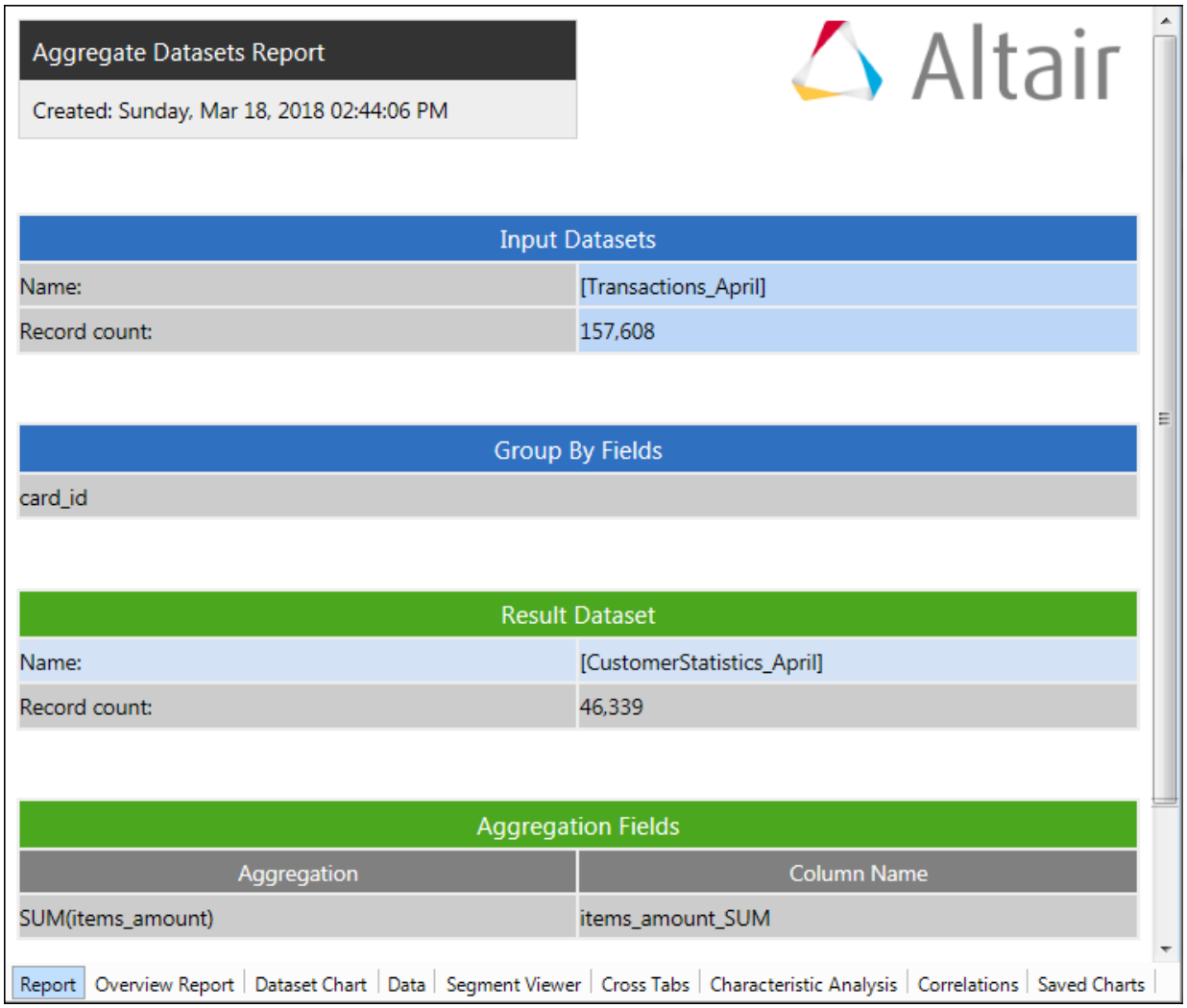


Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the operation, the node representing the aggregated dataset appears on the Workflow canvas, and the corresponding dataset appears in the Project Explorer panel.



The *Report* tab of the resulting dataset provides the information on the input and output datasets, the total number of records in each, the Group By field, and the aggregation fields that were specified in the Aggregate Dataset Wizard.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The other tabs, such as *Overview Report*, provide the same functionality as in any other dataset, and all the usual dataset operations are applicable.

See also:

[About Data Preparation]

[Appending Datasets]

[Removing Duplicate Records]

[Joining Datasets]

Appending Datasets

The **Append** node in the **Manipulate** palette merges two or more datasets into a single dataset by consecutively appending them to each other in the user-defined order.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

An example use case is the task of merging customer transaction data that arrives in monthly batches. All datasets in this case have the same structure, and the records of each batch can be easily appended at the end of the previous one.

The append operation normally requires that the structure of the datasets be the same, i.e., that their corresponding columns have matching or compatible types and go in the same order. The Append node wizard provides additional flexibility by allowing the user to map the columns, so their names and order in different datasets need not be the same. However, the data types of the mapped columns must match or be compatible. Intelligent defaults for the mapping are provided to automatically map any columns that have the same name and data type in all input datasets.

The first dataset determines the output columns in the resulting dataset. For example, if Dataset 1 has fewer columns than all other datasets, the columns of Dataset 1 not matched with any column in the other datasets are appended with missing values (null) in the resulting dataset.

This simple example for the case of 2 datasets shows retail transaction datasets for January and February that need to be merged. The timestamp field indicates the transaction date:

	transaction_id	card_id	payment_method	timestamp	product_code	items_number
1	3260	104711180	Cash	04.01.2001	000000003	1
2	30405	102936649	CreditCard	31.01.2001	000000003	1
3	28080	103696712	CreditCard	28.01.2001	000000003	1
4	10637	100826685	DebitCard	11.01.2001	000000003	1
5	18257	102633333	DebitCard	19.01.2001	000000003	1
6	8202	100016812	DebitCard	09.01.2001	000000003	1
7	6277	103940003	Cash	07.01.2001	000000003	2

+

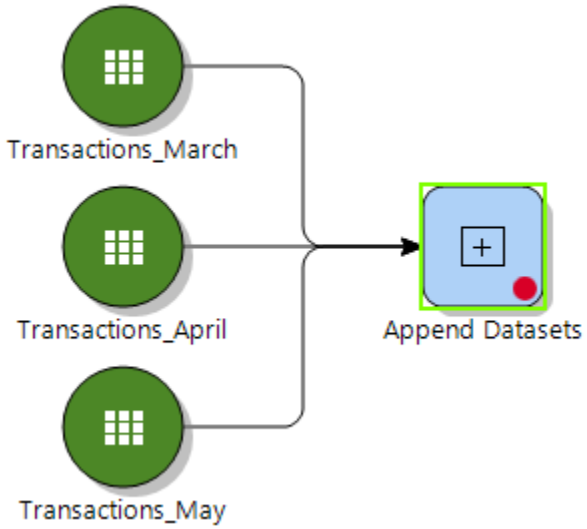
	transaction_id	card_id	payment_method	timestamp	product_code	items_number
1	57280	100263804	CreditCard	26.02.2001	000000003	1
2	55295	105112071	DebitCard	24.02.2001	000000003	1
3	39738	105114408	DebitCard	09.02.2001	000000003	1
4	49118	105201505	CreditCard	18.02.2001	000000003	1
5	49749	106033341	CreditCard	18.02.2001	000000003	1
6	51705	100770252	CreditCard	20.02.2001	000000003	1

The resulting dataset will have the same columns, with the February data rows appended at the end of the January data.

The Append Operation

To introduce an Append operation into your workflow, drag the **Append** node from the **Manipulate** palette onto the canvas and connect the input datasets to the Append node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



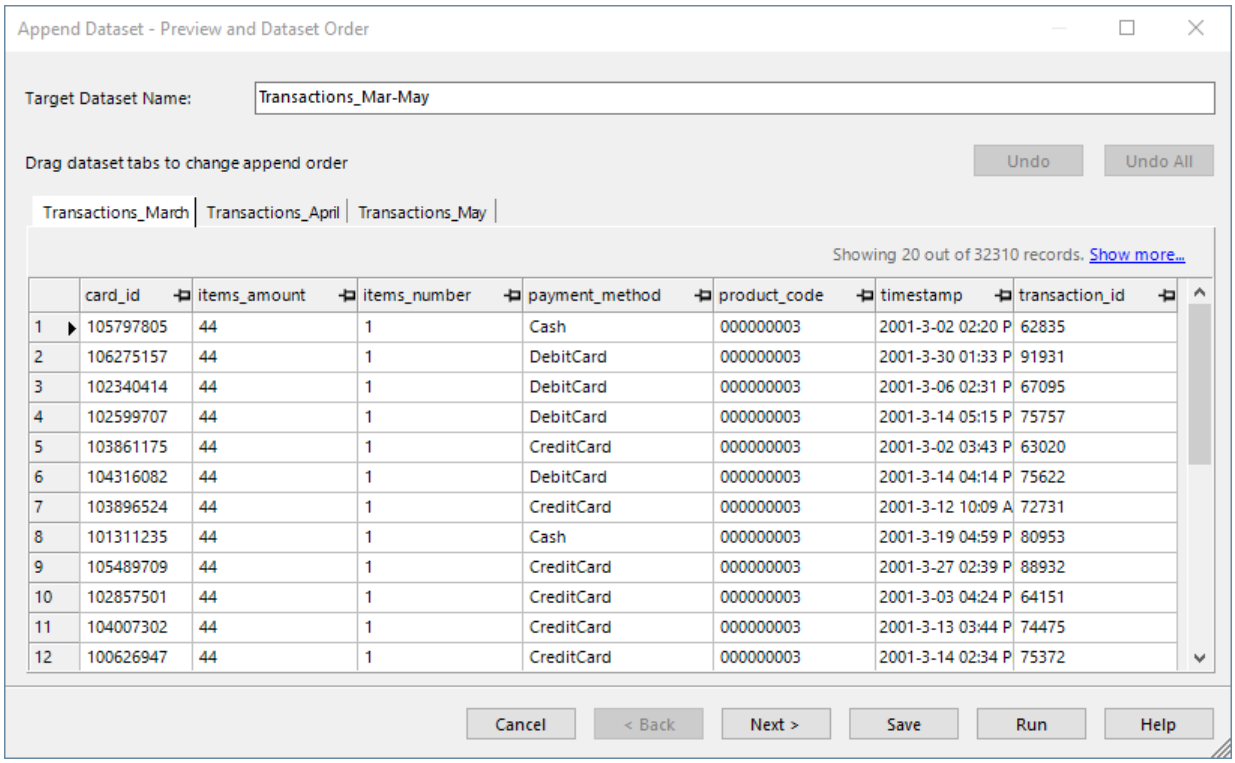
To specify the append conditions, double-click the Append Datasets node or right-click on it and select **Modify**.

In the first page of the wizard, each input dataset is represented by a separate tab. Each tab provides data preview.

- Specify the appending order by dragging any of the tabs to the desired position.
- Specify the name for the output dataset.

In the example below, the input datasets will be appended in the chronological order, which coincides with the tabs order, and the result will be named *Transactions_Mar-May.*

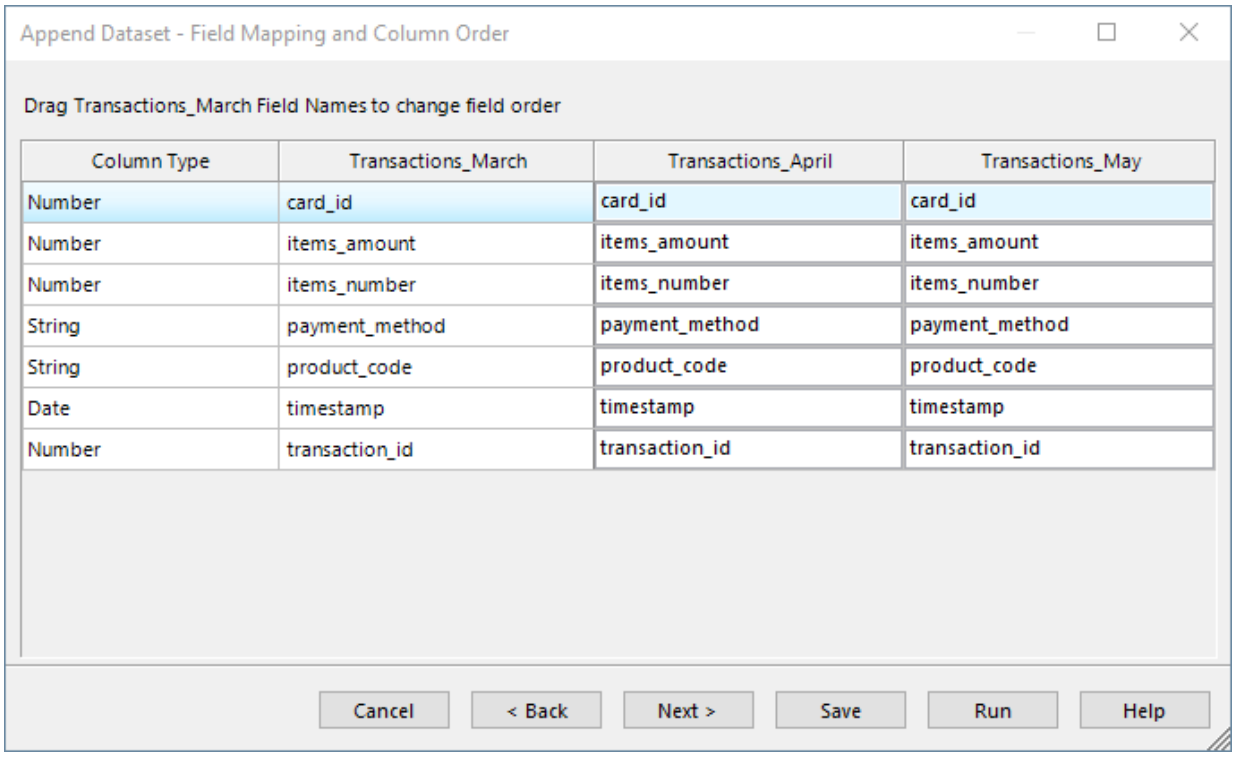
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



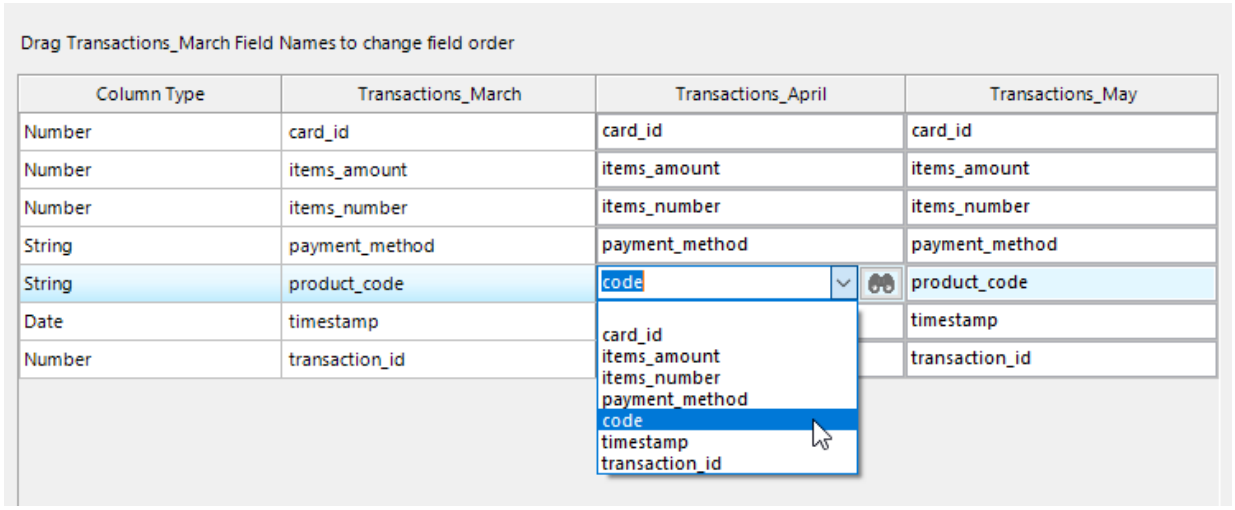
	card_id	items_amount	items_number	payment_method	product_code	timestamp	transaction_id
1	105797805	44	1	Cash	000000003	2001-3-02 02:20 P	62835
2	106275157	44	1	DebitCard	000000003	2001-3-30 01:33 P	91931
3	102340414	44	1	DebitCard	000000003	2001-3-06 02:31 P	67095
4	102599707	44	1	DebitCard	000000003	2001-3-14 05:15 P	75757
5	103861175	44	1	CreditCard	000000003	2001-3-02 03:43 P	63020
6	104316082	44	1	DebitCard	000000003	2001-3-14 04:14 P	75622
7	103896524	44	1	CreditCard	000000003	2001-3-12 10:09 A	72731
8	101311235	44	1	Cash	000000003	2001-3-19 04:59 P	80953
9	105489709	44	1	CreditCard	000000003	2001-3-27 02:39 P	88932
10	102857501	44	1	CreditCard	000000003	2001-3-03 04:24 P	64151
11	104007302	44	1	CreditCard	000000003	2001-3-13 03:44 P	74475
12	100626947	44	1	CreditCard	000000003	2001-3-14 02:34 P	75372

In the second page of the wizard, verify the mapping of the fields across all input datasets and change it if necessary. You will only be able to map the fields of the same or compatible data types (for example, Number to Number, String to String, etc.) To change the mapping for any field, click on the corresponding field in the same row of any other column and select the desired field from the drop-down list. You can also change the field order by dragging the field names of the first dataset up or down.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

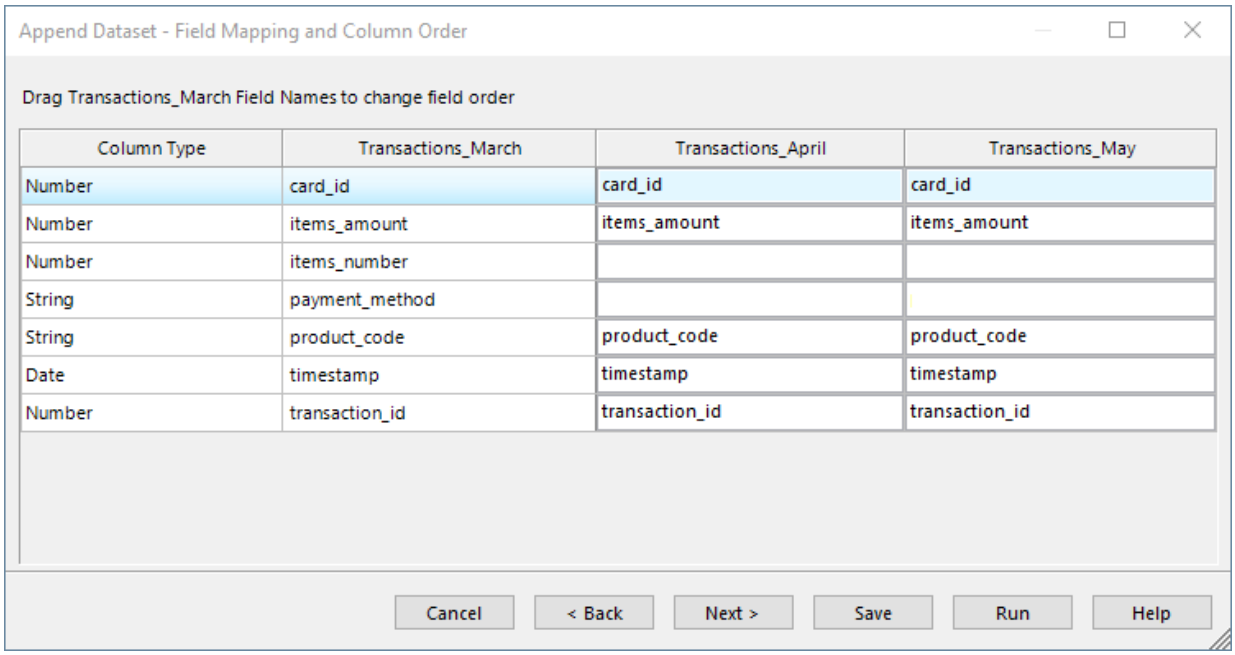


In the example below, we map the field ***product_code*** in dataset Transactions_March to the field ***code*** in dataset Transactions_April:

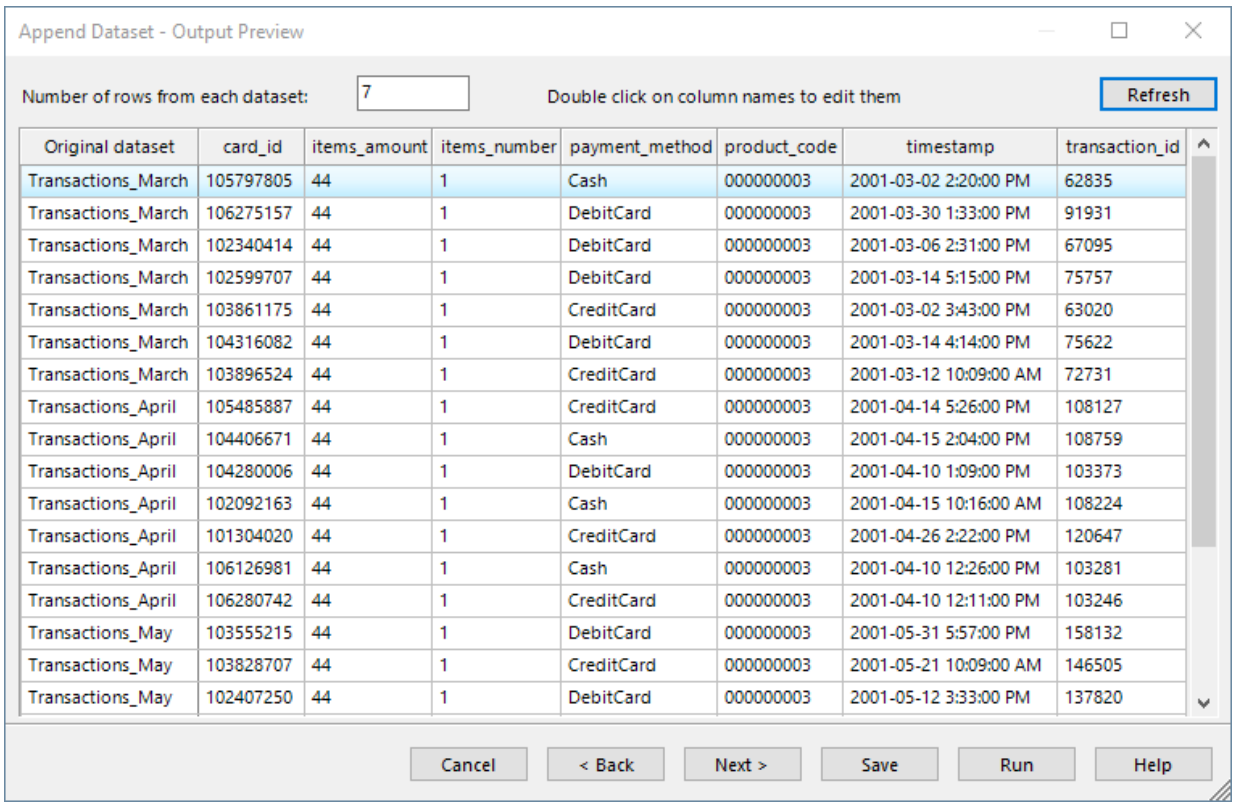


In another example below, datasets Transactions_April and Transactions_May do not have columns ***items_number*** and ***payment_method***, so the rows in these columns of Transactions_March will be appended with missing values in the output dataset.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

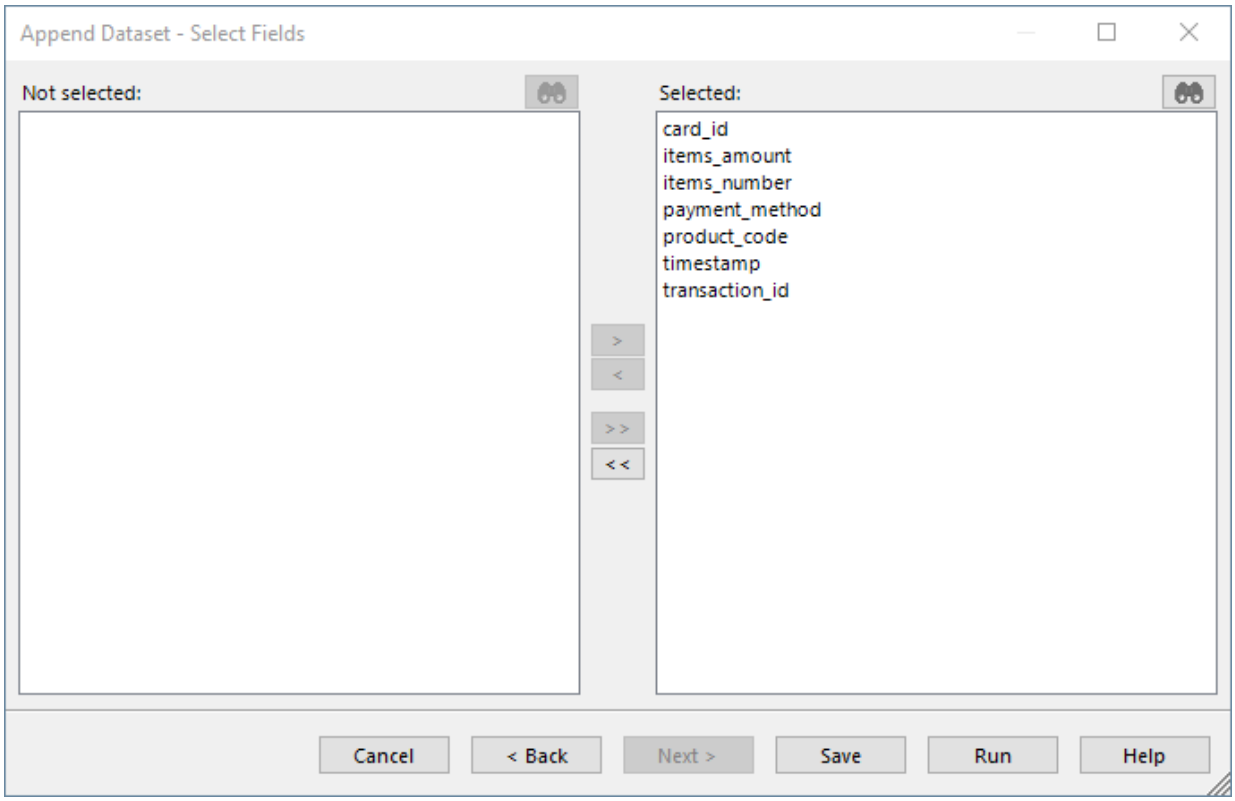


Click **Next** to preview the output dataset. If necessary, change the number of rows displayed for each dataset and click **Refresh**. You can change the column names in this grid by double-clicking on the column headers.

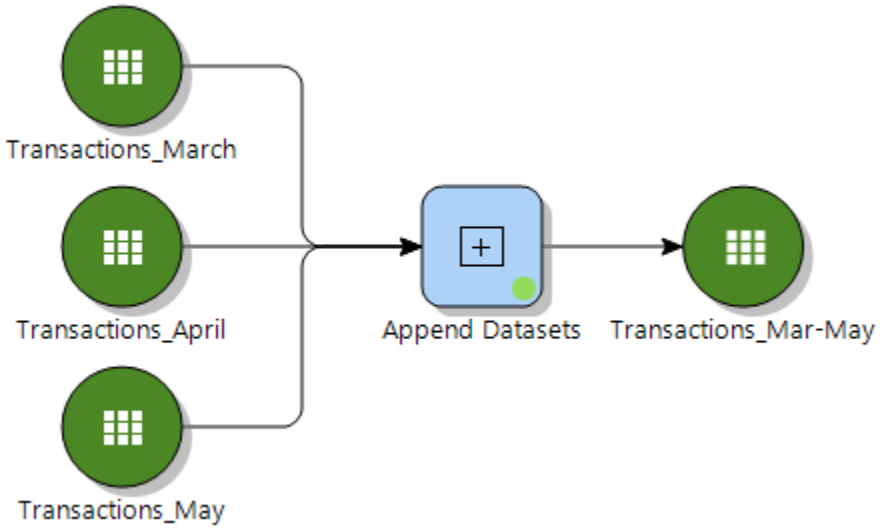


Click **Next**. In the next page, select or deselect the fields for the output dataset. By default, all fields are selected.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

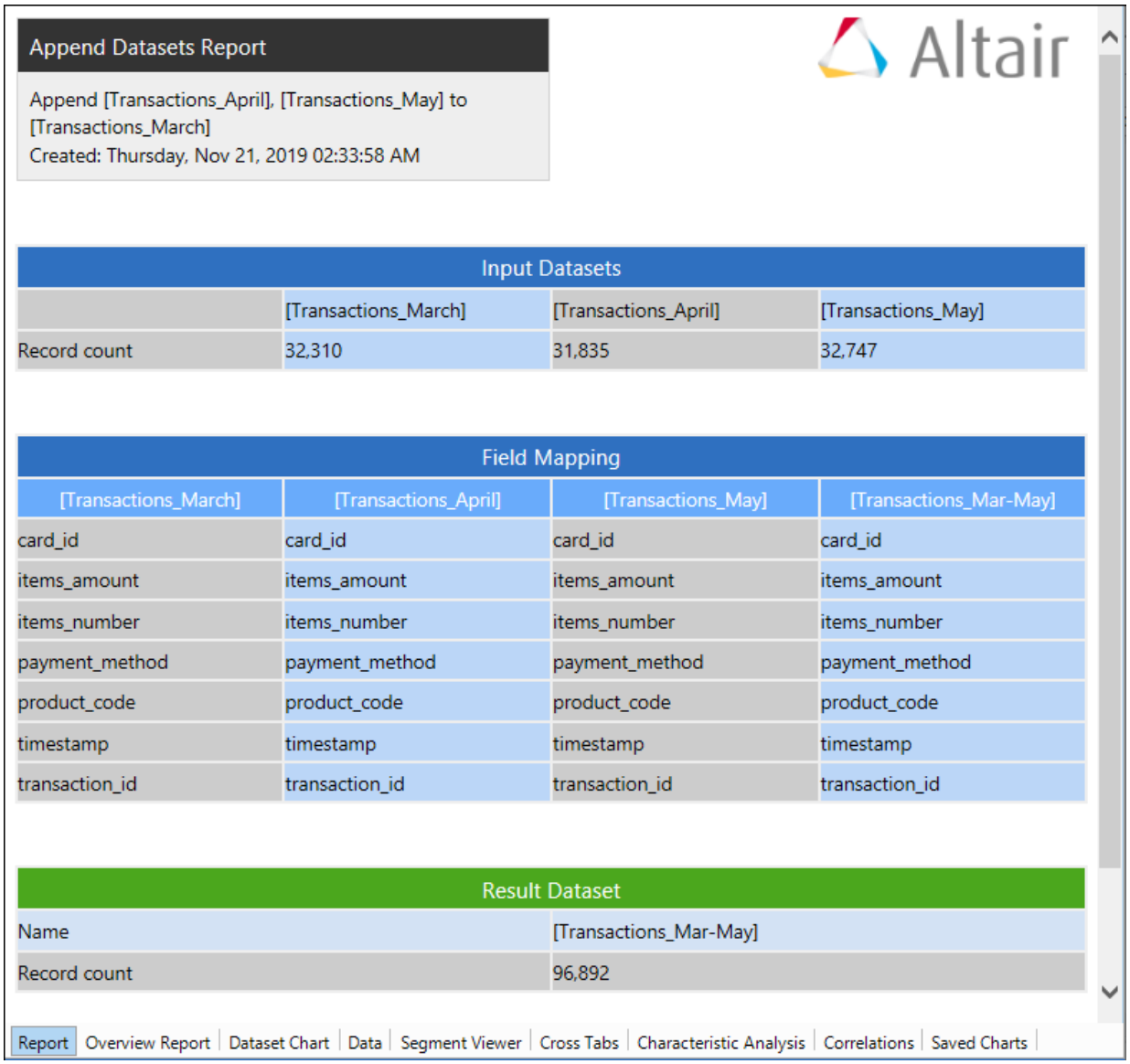


As a result, the node representing the output dataset appears on the workflow canvas, and the output dataset itself is created as concatenation of Dataset 1 and Dataset 2, and is shown at the same level with the source datasets in the Project Explorer:



The *Report* tab of the resulting dataset provides the information on the input and output datasets, the total number of records in each, and the field mapping that was specified in the Append node wizard.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The other tabs, such as *Overview Report*, provide the same functionality as in any other dataset, and all the usual dataset operations are applicable.

See also:

[About Data Manipulation]

[Joining Datasets]

[Removing Duplicate Records]

[Aggregating Data]

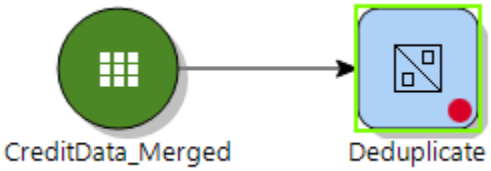
Removing Duplicate Records

Eliminating redundant data is an important stage in the data cleanup process to prepare for data mining and building predictive models.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

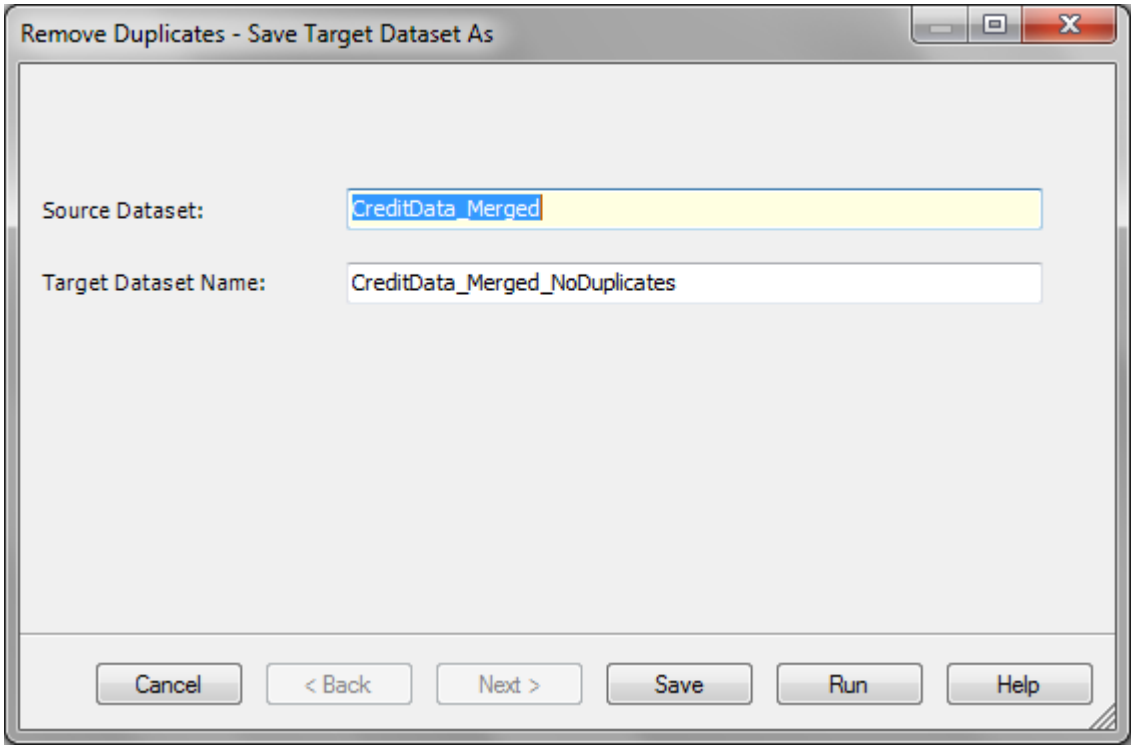
For example, unwanted duplicate records may occur in the dataset resulting from merging data from many sources using the Append operation.

To remove records that contain duplicate entries, drag the **Deduplicate** node from the **Manipulate** palette onto the canvas and connect the input dataset to the node.



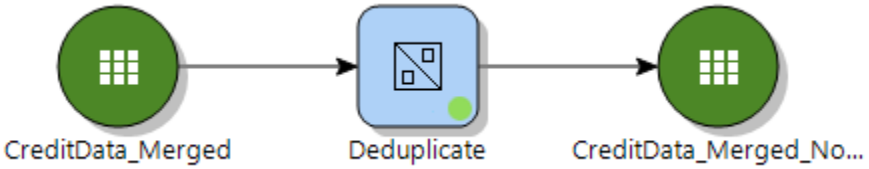
To specify the conditions, double-click the Deduplicate node or right-click on it and select **Modify**.

In the Remove Duplicates dialog, specify the name of the output dataset in the Target Dataset Name field. The default name is formed by adding the *_NoDuplicates* suffix to the name of the source dataset.

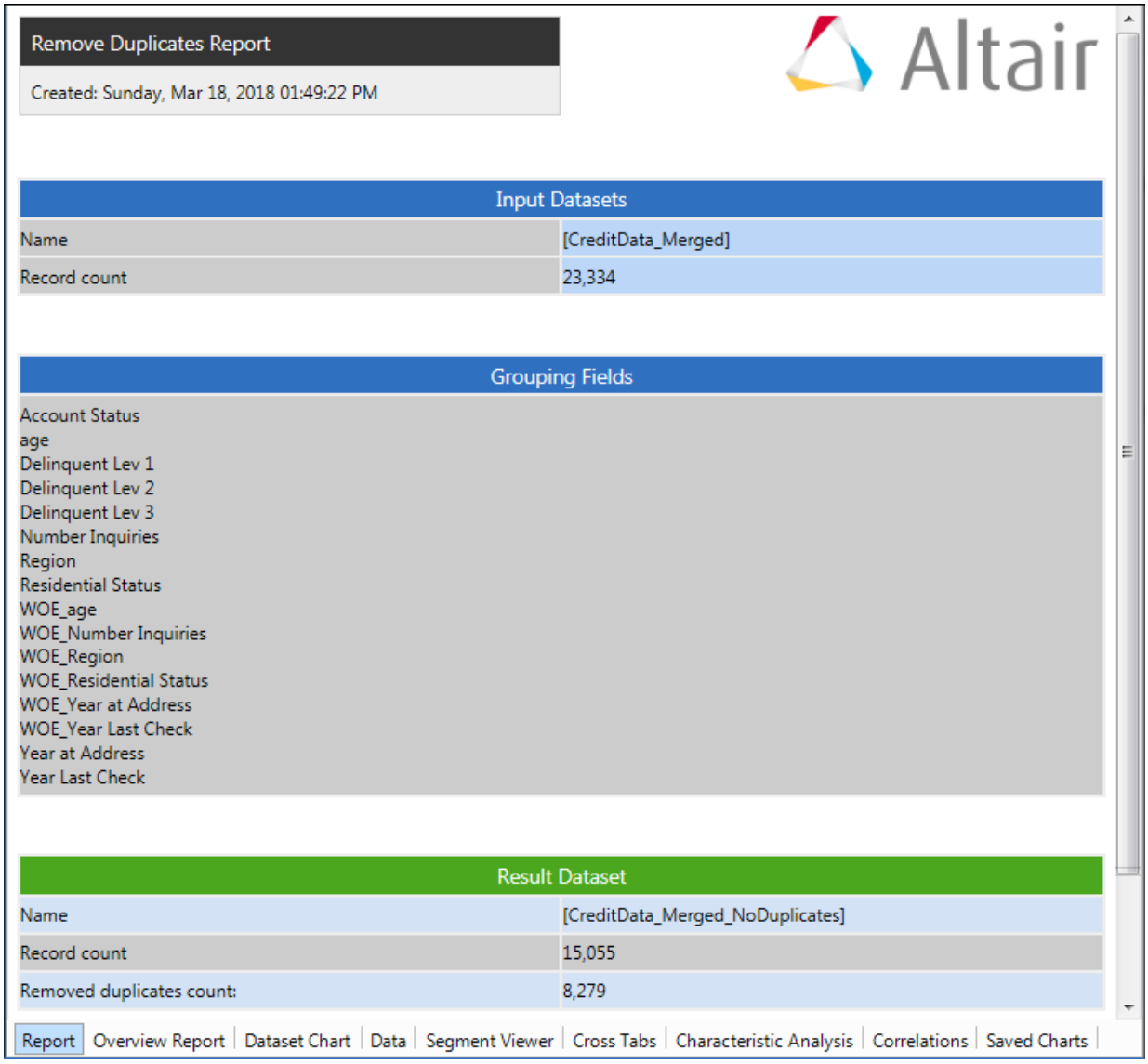


Click **Save** to save the settings or click **Run** to save the settings and run the process immediately. Upon running the node double-click the output dataset node to view the results. The output dataset contains only unique records from the source dataset.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The *Report* tab of the resulting dataset provides the information on the input and output datasets, the total number of records, and the number of duplicate records removed.



The other tabs, such as *Overview Report*, provide the same functionality as in any other dataset, and all the usual dataset operations are applicable.

See also:

[About Data Preparation]

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Appending Datasets

Aggregating Data

Joining Datasets

Filter

The **Filter** node in the **Manipulate** palette is used to select or delete the records satisfying user-defined conditions. It produces a new, filtered dataset on the workflow.

Note: Interactive filtering can be performed within the dataset views without creating a new dataset. A dockable filter panel enabled in all dataset tabs serves this purpose. Note, however, that the Filter node functionality is more advanced than that of the interactive filter panel in datasets. For example, the Filter node allows you to specify complex conditions using SQL expressions. If desired, you can start defining filter conditions while exploring dataset views and then export them to a new Filter node on the workflow to work further. See the topic Interactive Filtering on Datasets for details.

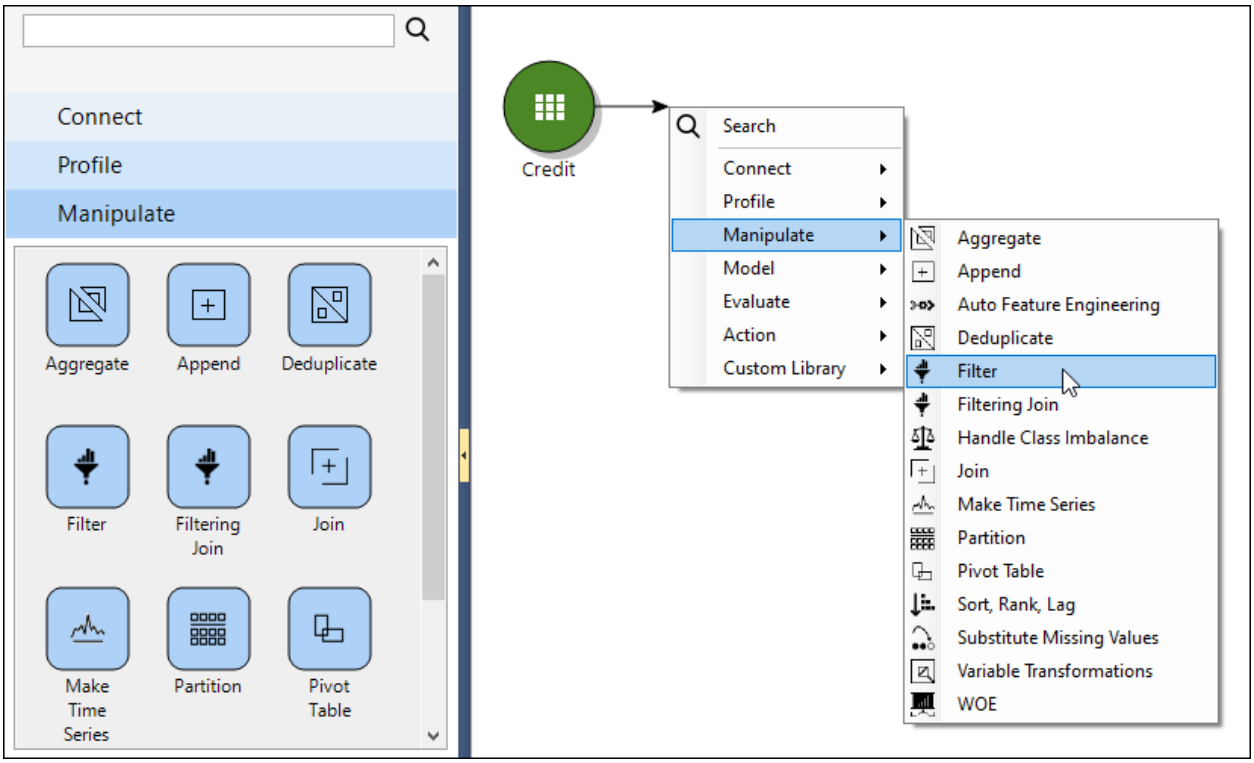
To add a Filter node to the workflow:

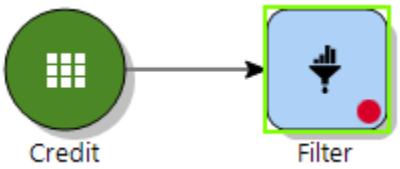
- Drag a connection arrow from the input dataset node to any area on the Workflow canvas background. Releasing the mouse button brings up a context menu. From the context menu, select **Manipulate --> Filter**, as in the example screenshot below. A new **Filter** node will be created, and the wizard will open.

  OR:

- Drag the **Filter** node from the **Manipulate** palette onto the canvas and connect the input dataset.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

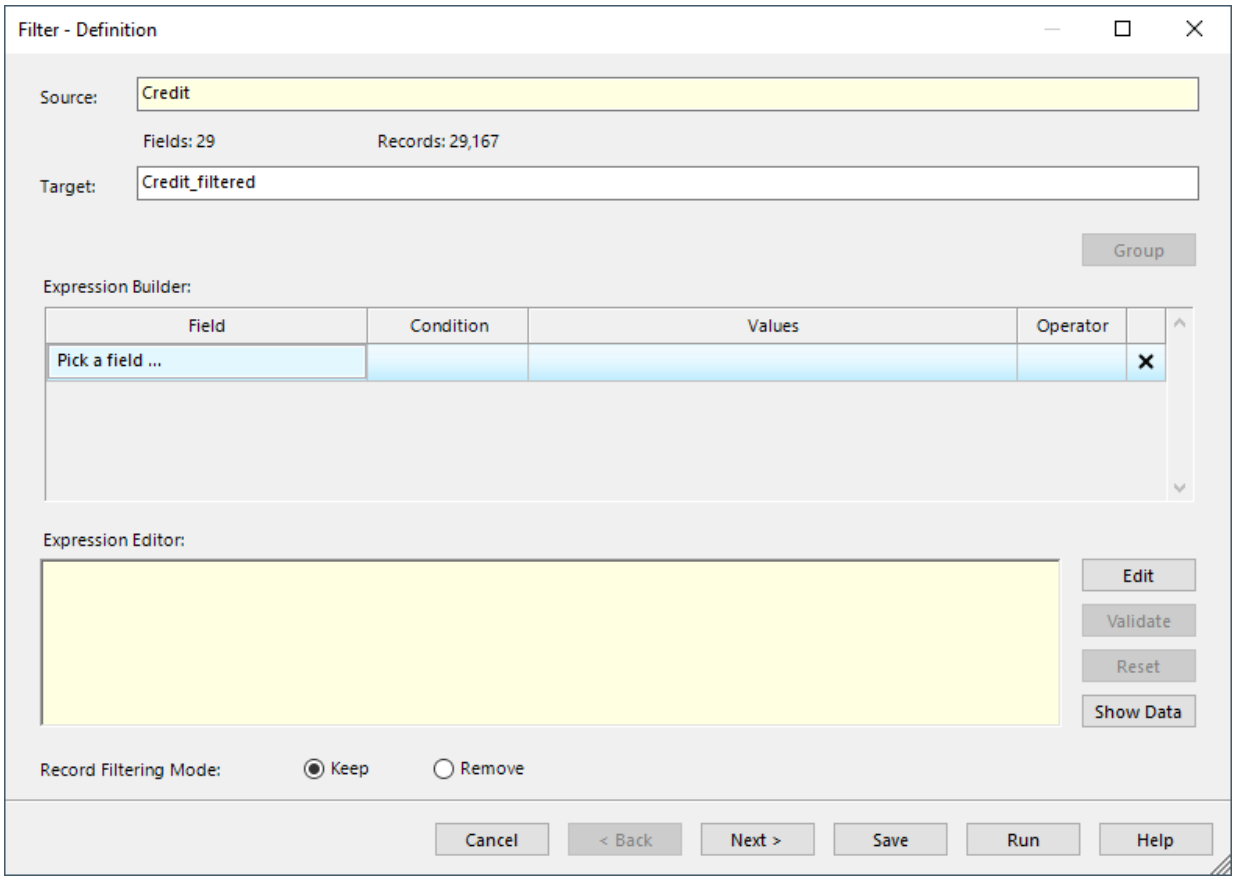




If the Filter node is already on the workflow canvas, double-click it to open the wizard. Alternatively, right-click on the node and select **Modify**.

The wizard helps you build an expression for the filter. Each row in the *Expression Builder* grid represents a condition on a certain field. This is similar to common query-building interfaces. The conditions are combined using the **AND** / **OR** operators selected in the last column. You can also manually type the desired SQL expression in the *Expression Editor* frame.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

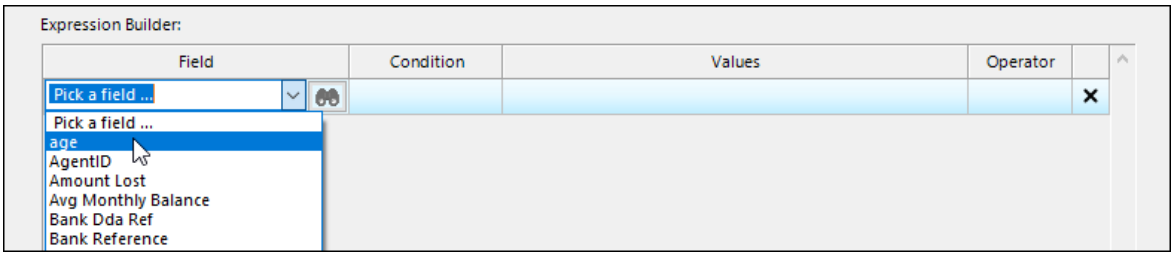


In the first page of the wizard:

- The **Source** field displays the name of the input dataset of the Filter node.
- Specify the name for the output (filtered) dataset in the **Target** field.
- Add conditions using the Expression Builder and optionally edit it using the Expression Editor described below.

Expression Builder

Define the filtering conditions by adding them in the Expression Builder grid, one row per field. First, click in the **Field** column to pick a field. Select the desired field from the drop-down list. If there are hundreds of fields, you can use the binoculars button to open the Field Finder.

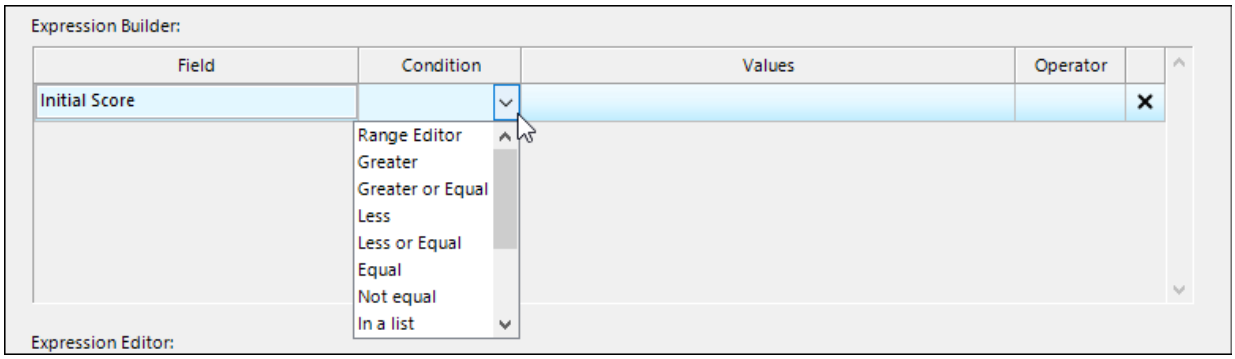


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Click in the **Condition** column and select the desired condition (relation) from the drop-down list. The available conditions are:

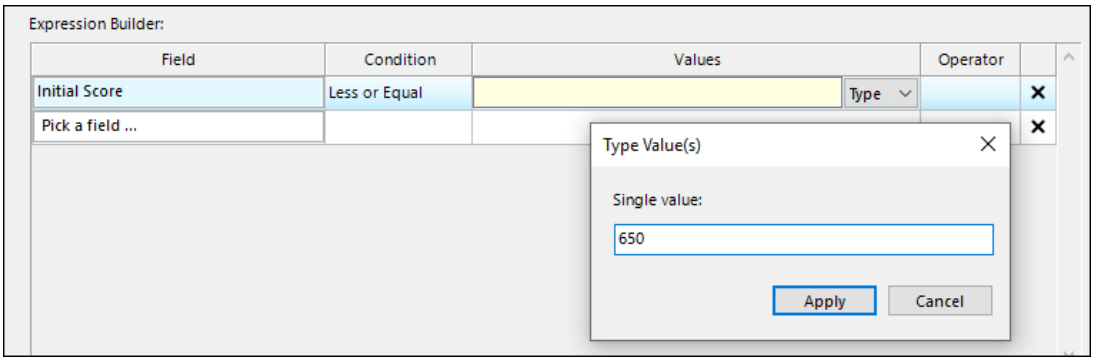
- for numeric and Date/Time fields: *Range Editor / Greater / Greater or equal / Less / Less or equal / Equal / Not equal / In a list / Not in a list / Between / Not between / Is null / Is not null*
- for String and Boolean fields: *Range Editor / Equal / Not equal / In a list / Not in a list / Is null / Is not null*

The **Range Editor** option is available for all types of fields. Its function is described further in this topic.



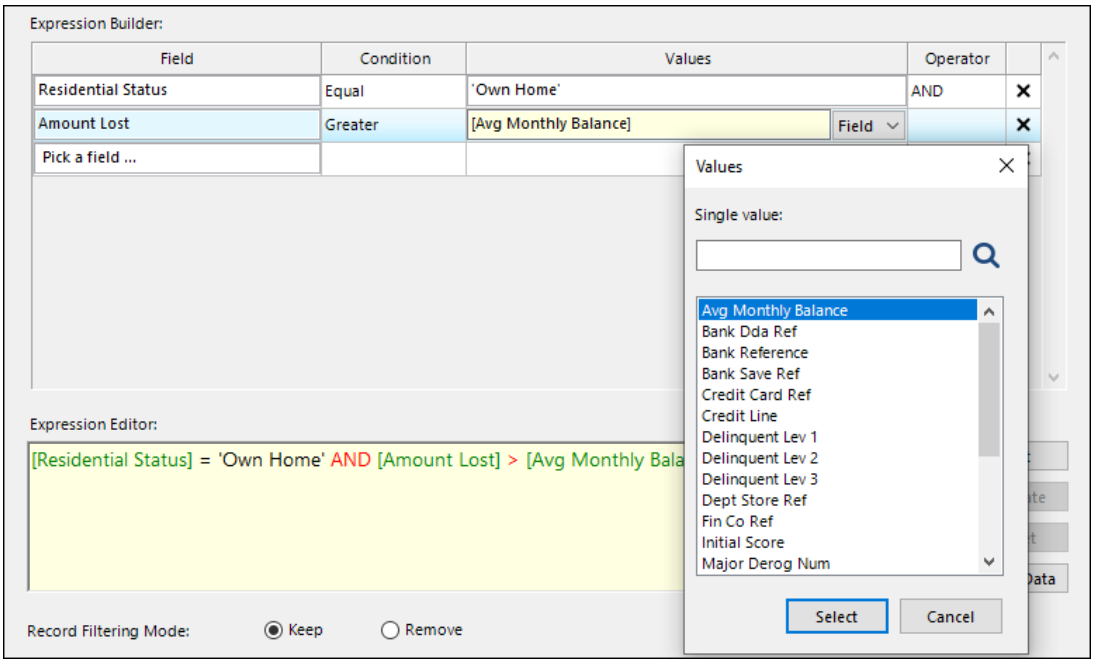
Click in the **Values** column to specify the values for the condition. A drop-down list provides three options: "*Select*", "*Type*", and "*Field*".

- Choose "*Select*" to choose from the list of all values of the variable.
- Choose "*Type*" to enter a custom value manually:



- Choose "*Field*" to select another field that the selected one should be compared to. This should be a field of a compatible data type. From the drop-down list of available fields, select the desired field to compare. You can type in the Search field for quick search.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

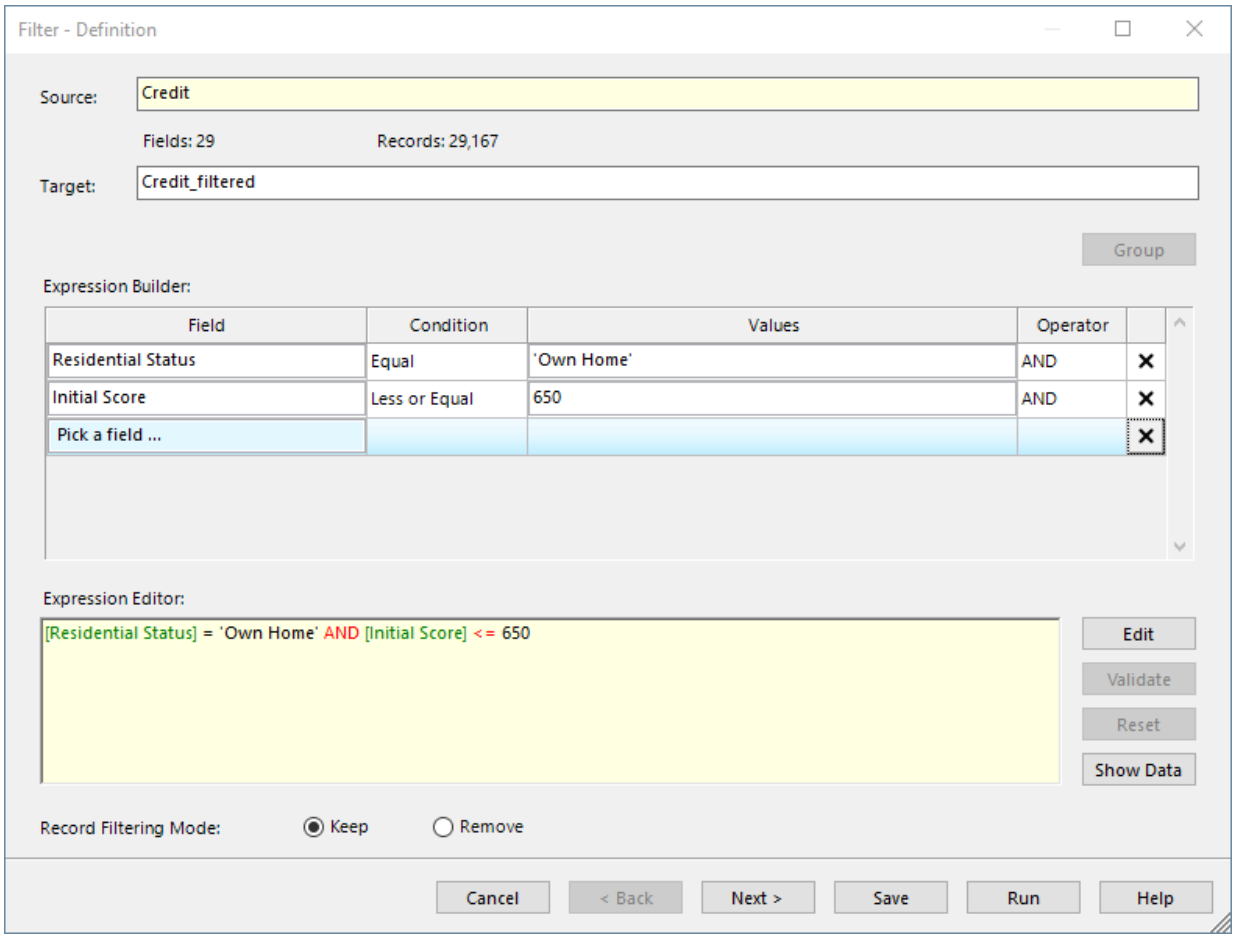


Once the condition is specified, its expression is shown in the Expression Editor frame at the bottom. If required, click in the **Operator** column to select the operator (**AND** / **OR**) supposed to combine the current and the next condition. Once the Operator is selected, a new empty row will be added automatically to get the input for the next condition. Add more conditions if required.

> Note: Conditions like (`(x < 10) OR (x > 100)`) for numeric and Date/Time variables can be entered in a single row using the options "***Not between***" or "***Range Editor***".

As you add conditions, upon adding each row, the expression for the added condition is appended to the resulting expression in the **Expression Editor** frame.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



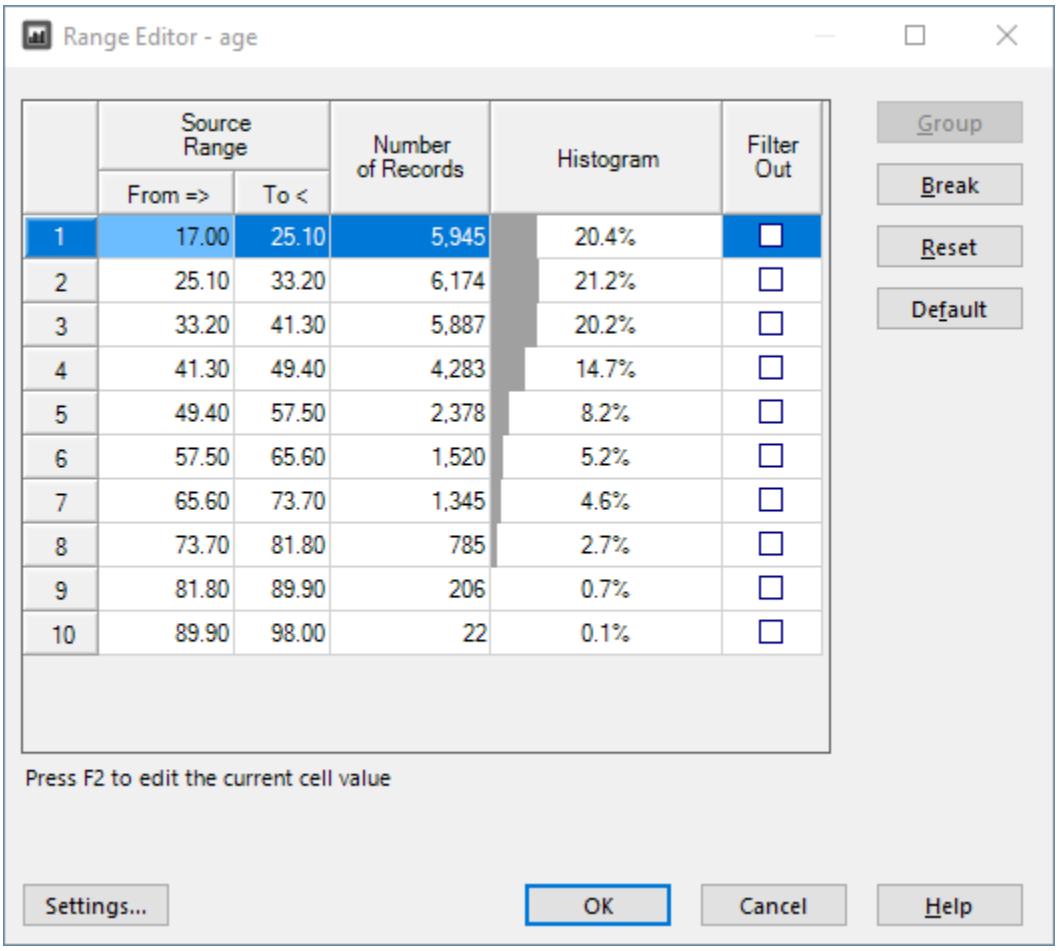
Range Editor for Conditions

If Condition is set to **Range Editor**, the Edit Ranges button appears in the **Values** column. Click it to specify the range(s) of values for the condition in the current row.

Field	Condition	Values	Operator	
Initial Score	Less or Equal	650	AND	✕
Residential Status	Equal	'Own Home'	AND	✕
age	Range Editor	Edit Ranges		✕

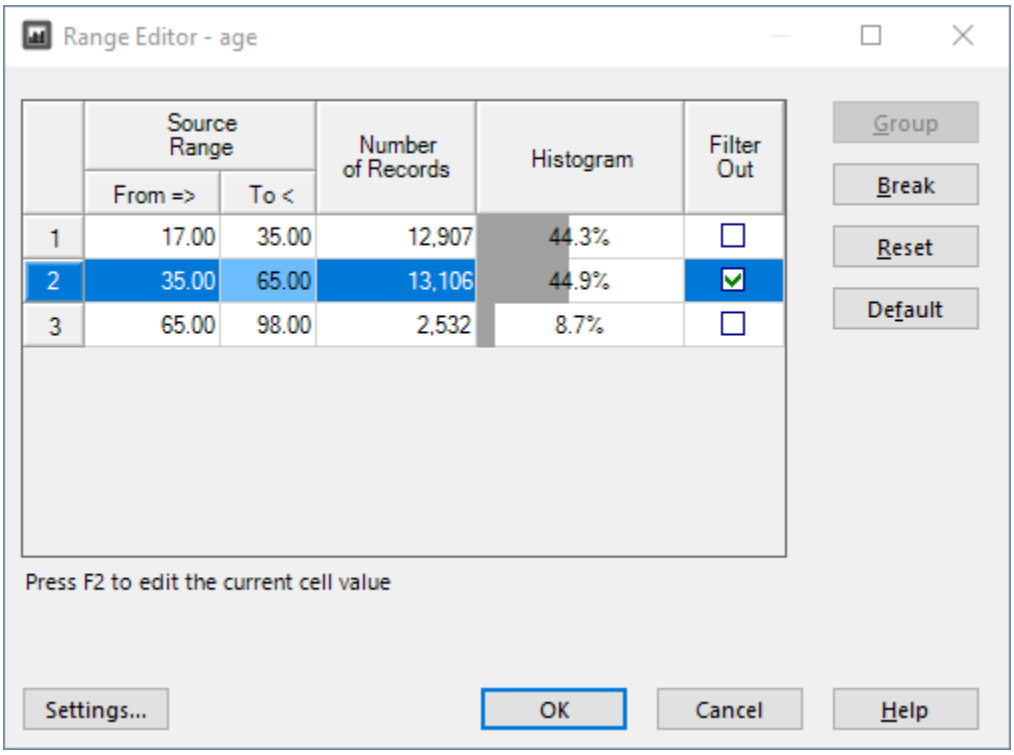
The Range Editor window opens. Its interface is similar to the Range Editor in the Variable Transformation node. For numeric and Date/Time variables, it is similar to the Continuous Range Editor, while for String variables it is similar to the Discrete Range Editor. The main difference is the presence of the **Filter Out** column, where you can exclude specific ranges from the condition.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



As usual, you can group, break, and reset ranges, edit their left and right boundaries, and change the number of bins by clicking the Settings button. While editing the ranges, you can see the changes in the interactive histogram. In the following example, the middle range [35, 60) for the age field is excluded from the condition:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The corresponding expression to be appended in the Expression Editor frame will be

```
          AND (([age] >= 17.0 AND [age] < 35.0) OR ([age] >= 65.0 AND
[age] <= 98.0))
```

Once you finish editing ranges and click OK, the expression for the current condition is appended to the Expression Editor frame.

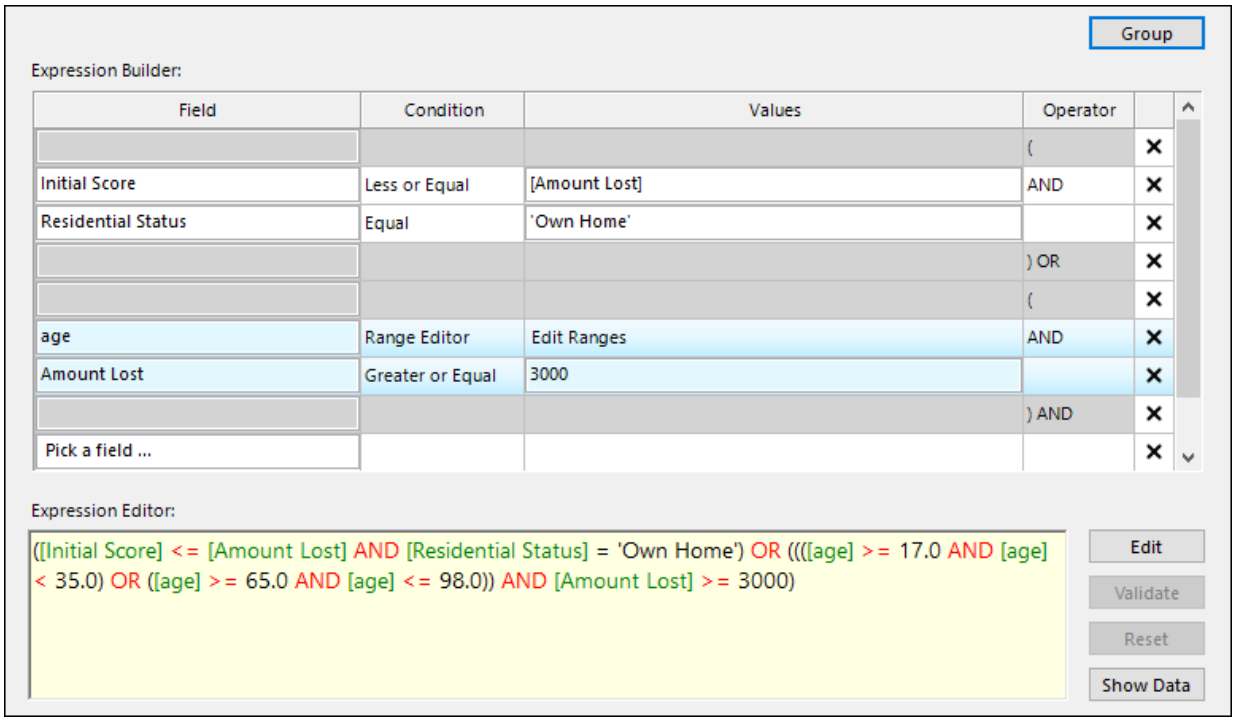
Grouping and Deleting Conditions

To **delete** a condition from the filter, click on the close button in the rightmost column  . Alternatively, click on the field name in the Field column and press the Delete key. The row representing the condition will be removed from the grid.

To group conditions, select multiple rows by dragging the mouse down in the Operator column or by clicking in the Operator column while holding down the SHIFT or CTRL key on your keyboard. Once multiple rows are selected (highlighted), the **Group** button becomes enabled. Click the **Group** button. As a result, brackets get added around the selected conditions. The content of the Expression Editor frame gets updated accordingly.

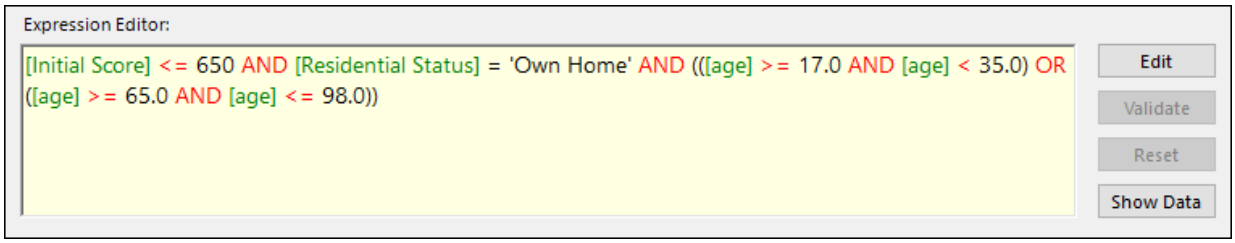
In the example below, each of two pairs of conditions joined by the AND operator is grouped, while the two pairs are combined with the OR operator.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Groups of conditions can be reordered by moving / dragging them up and down in the Expression Builder grid. The content of the Expression Editor frame gets updated accordingly.

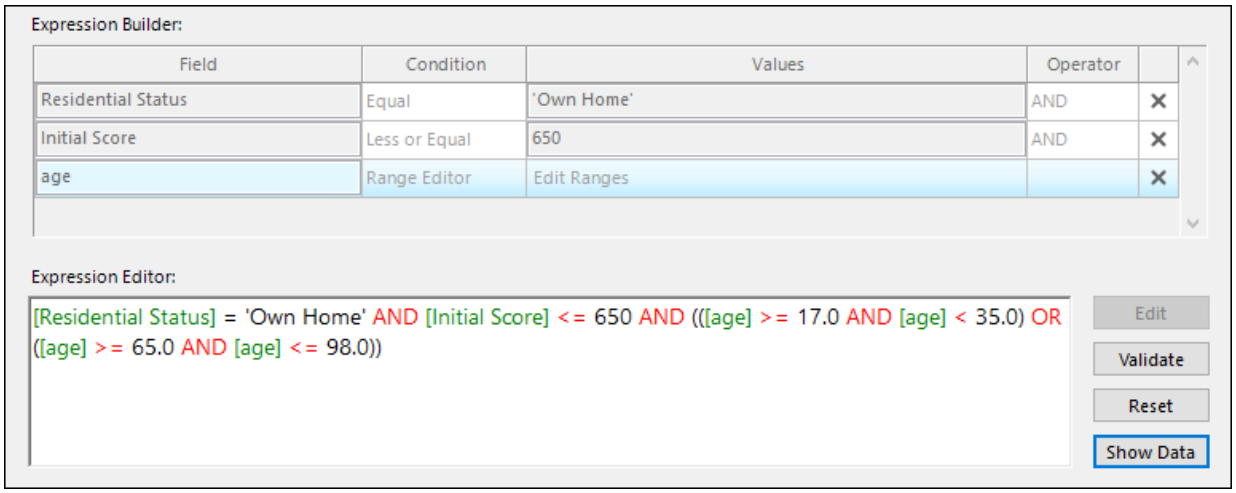
Expression Editor



Once you finish specifying conditions in the Expression Builder grid, you can modify the expression in the Expression Editor by clicking the **Edit** button. You can certainly click Edit from the very beginning and enter the desired SQL code without using the Expression Builder at all.

Once you click **Edit**, you enter the Edit mode to enter or modify the code. In this mode the Expression Builder grid is not accessible and is grayed out.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



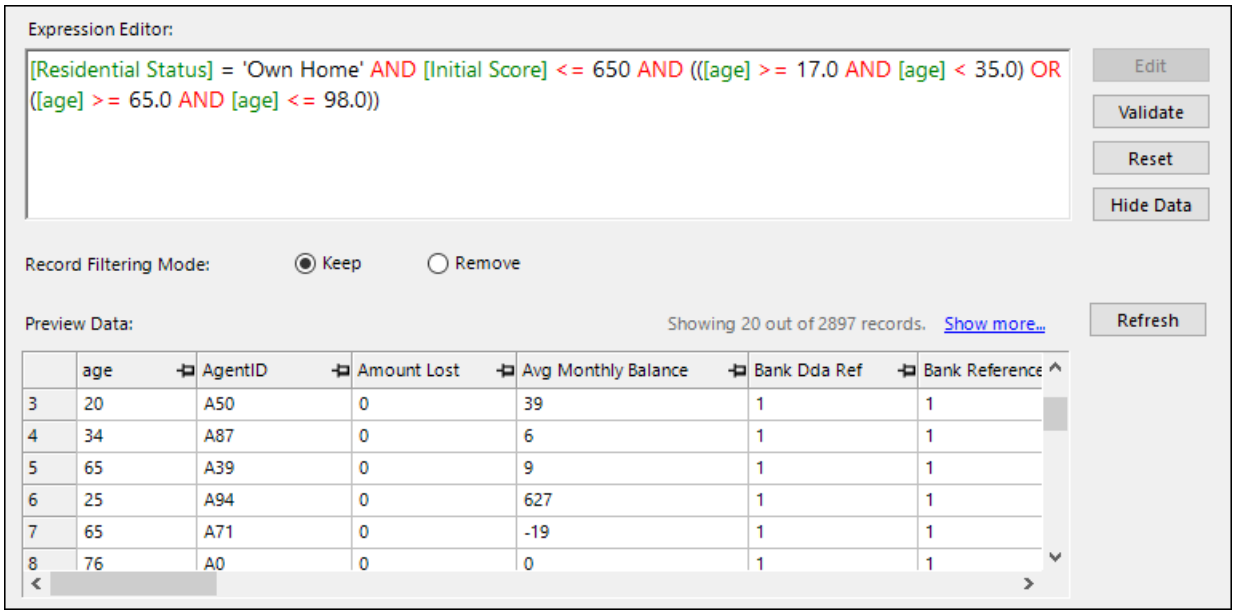
To exit the Edit mode click **Reset**. The **Reset** button cancels the changes in the Expression Editor and re-enables the Expression Builder grid. The expression is reset to match the Expression Builder content.

Click **Validate** to check the code syntax.

Click the **Show Data** button to preview the filtered data and see the resulting row count. This opens the **Preview Data** grid beneath the Expression Editor frame. The preview shows the top 10 rows of the filtered data. You can view more rows by scrolling down or clicking "*Show more*" at the top-right of the grid.

You may need to click the **Refresh** button if you recently edited the expression or modified the conditions in the Expression Builder. The Refresh button also automatically checks the code syntax.

Click **Hide Data** to hide the Data Preview grid.



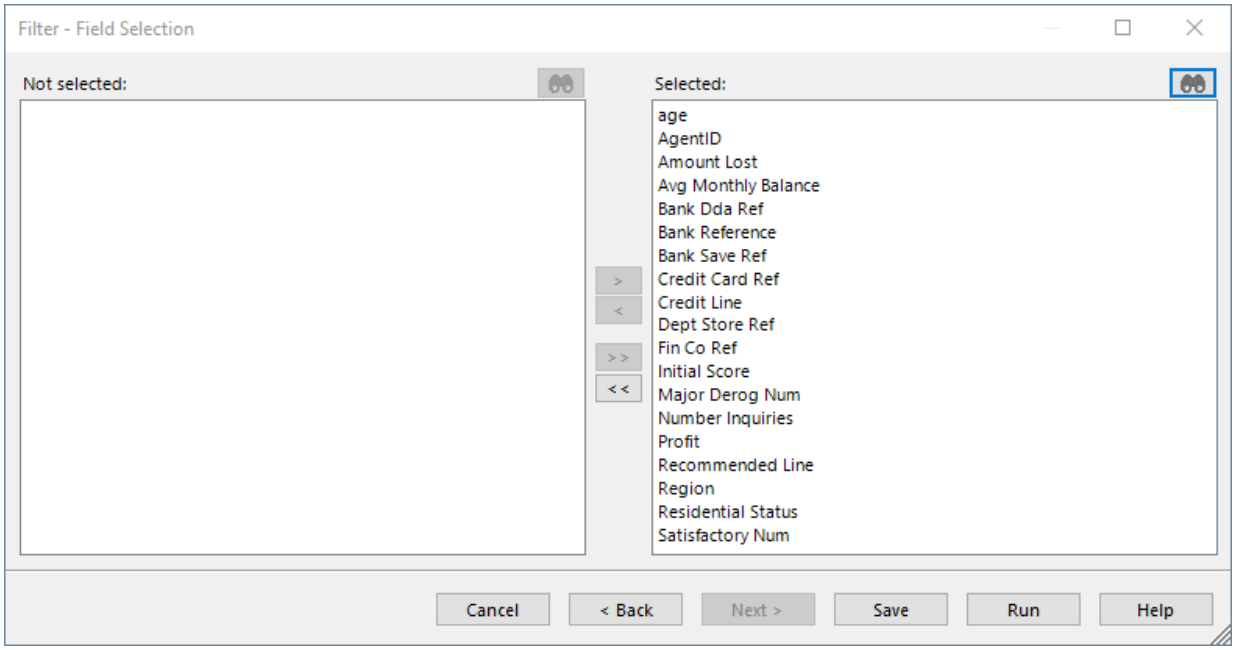
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Filtering Mode

For **Record Filtering Mode** at the bottom of the wizard page, choose whether to filter records by keeping or removing them. Selecting the **Keep** radio button only keeps the records that meet the specified conditions, while selecting **Remove** removes the rows that meet the specified conditions. The default is **Keep**.

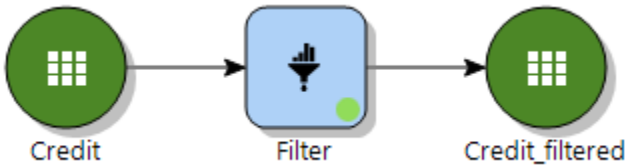
In this page you can click **Save** to save the settings or click **Run** to save the settings and run the node. By default, all fields get included in the output.

If required, click **Next** and select the fields from the input dataset to include or exclude for the filtered dataset on the output.



Click **Save** to save the settings or click **Run** to save the settings and run the node.

Upon running the node double-click the output dataset node to view the results.



See also: Interactive Filtering on Datasets

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Filtering Join

The **Filtering Join** node in the **Manipulate** palette joins two datasets to produce a new one containing the records from the first input dataset that either match (*semi-join*) or do not match (*anti-join*) in the second input dataset.

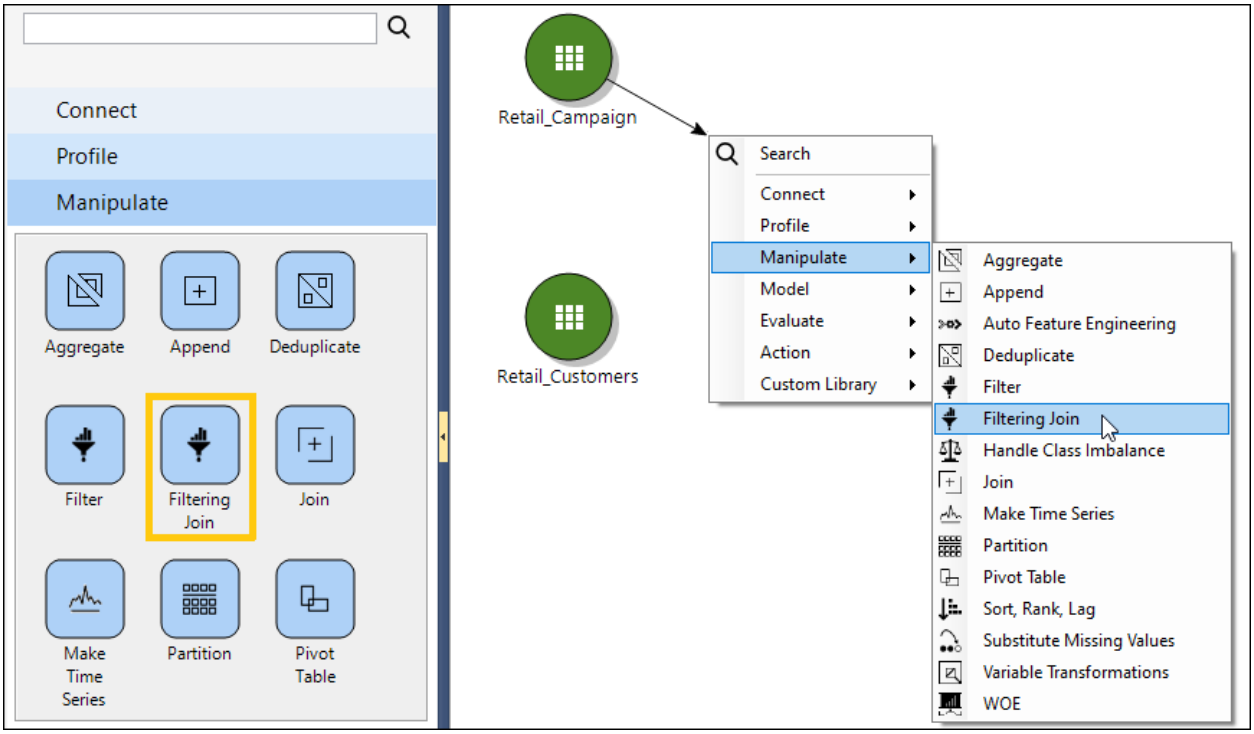
To join more than two datasets, you need to consecutively join pairs of datasets.

To perform a filtering join of two datasets:

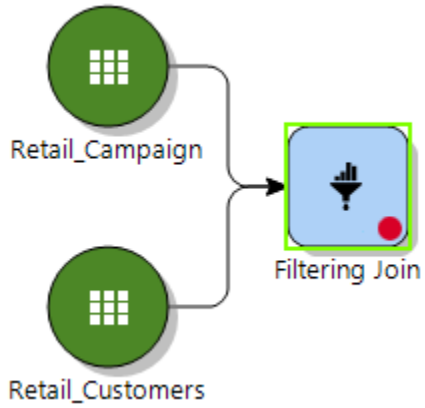
- Drag a connection arrow from one of the dataset nodes to any area on the Workflow canvas background. Releasing the mouse button brings up a context menu. From the context menu, select **Manipulate --> Filtering Join**, as in the example screenshot below. A new **Filtering Join** node will be created. Then connect the other input dataset to this node.

OR:

- Drag the **Filtering Join** node from the **Manipulate** palette onto the canvas and connect two input datasets to it.



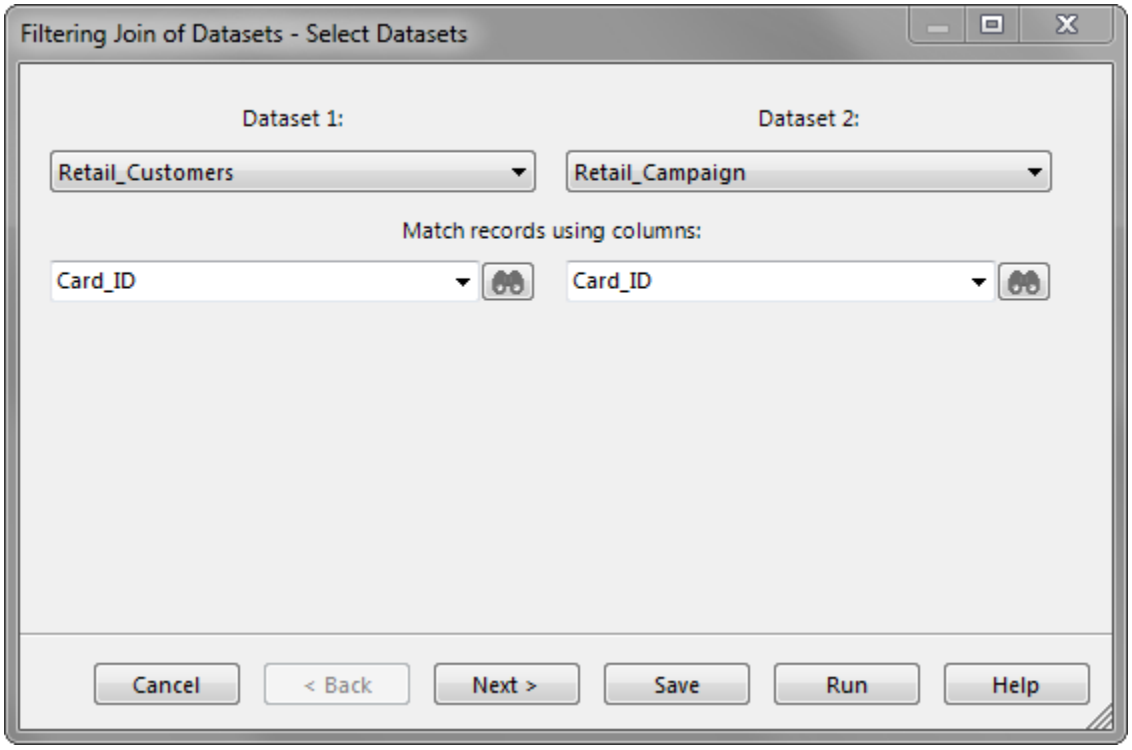
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Double-click the **Filtering Join** node to open the wizard. Alternatively, right-click on it and select **Modify**.

In the first page of the wizard:

- select the order for the datasets you wish to join (***Dataset 1*** and ***Dataset 2***). Dataset 1 is the one whose records will be in the output, filtered by the match condition selected at the next step;
- select the join column in each dataset. The records in the input datasets will be matched by the selected column. The corresponding columns should have compatible data types in both datasets.



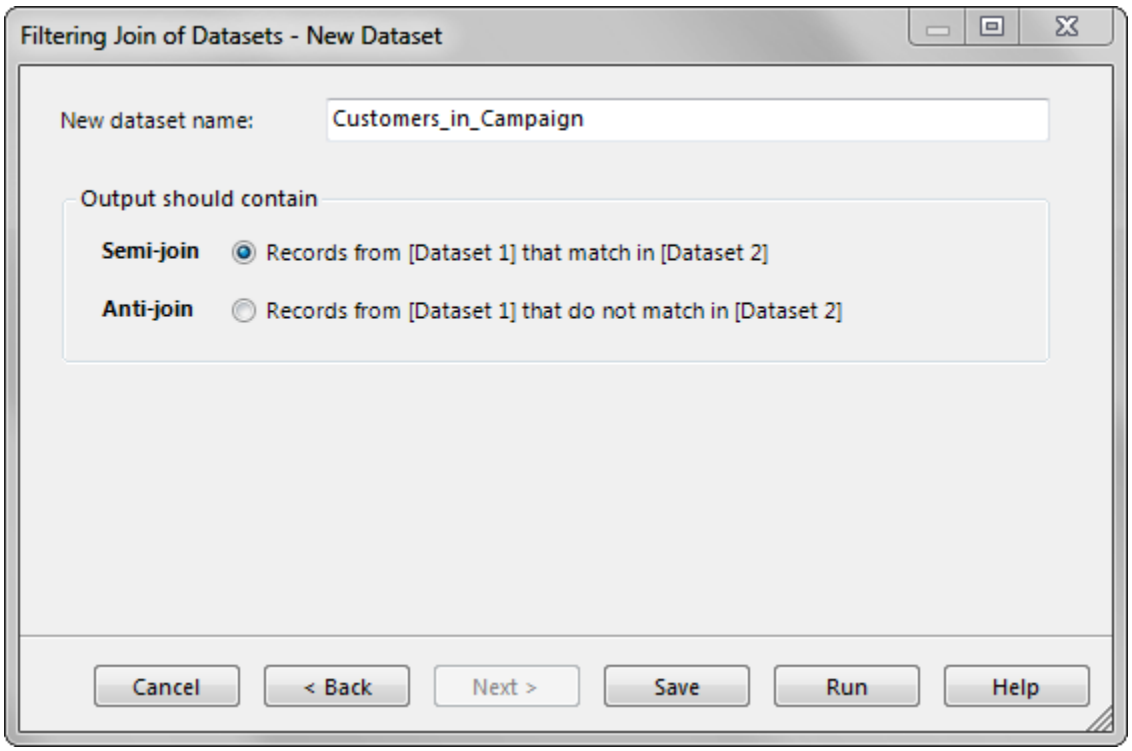
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

In the above example, a dataset with customer demographics data is being joined with a dataset containing campaign response data. Both datasets have a Card ID field (loyalty card number).

Click **Next**. In the next page, specify the name for the output dataset and select the join type depending on what records you want to include in the result:

- ***Semi-join***: Include the records from Dataset 1 that match in Dataset 2
- ***Anti-join***: include the records from Dataset 1 that do **not** match in Dataset 2

In both cases, only columns from Dataset 1 are included in the output.

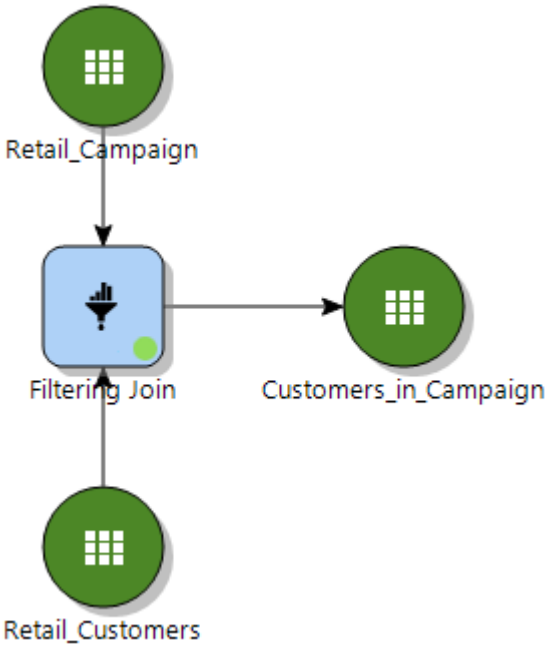


In the above example, only columns from the Retail_Customers dataset will be present in the output dataset because it was chosen as Dataset 1. If *Semi-join* is selected, the output will contain the records from Retail_Customers that match in Retail_Campaign (customers who responded to the marketing campaign). If *Anti-join* is selected, the output will contain the records from Retail_Customers that do not match in Retail_Campaign (customers who were not included or did not respond to the marketing campaign).

Click **Save** to save the settings or click **Run** to save the settings and run the process immediately.

Upon running the node double-click the output dataset node to view the results.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Joining Datasets

Datasets containing different information about related entities often need to be joined in a single dataset for analysis. The joining is possible if the datasets have at least one common attribute that serves as a link.

For example, retail transaction data and customer demographics data often come from two different sources, but each contains a customer ID field that can be used to link them together.

In the simple example below, we have a retail transaction dataset with fields like transaction ID, customer loyalty card ID, payment method, time of purchase, product code, etc., and a customer information dataset with fields like loyalty card ID, gender, birth date, etc. In both datasets, customers are identified by their loyalty card IDs, so we can join these datasets using the **Card_ID** field as a link:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

	transaction_id	card_id	payment_method	timestamp	product_code	items_number
1	254822	103436394	DebitCard	17/09/2001	000000003	1
2	3260	104711180	Cash	04/01/2001	000000003	1
3	108127	105485887	CreditCard	14/04/2001	000000003	1
4	246392	105267907	CreditCard	07/09/2001	000000003	1
5	158132	103555215	DebitCard	31/05/2001	000000003	1



	Card_ID	Registration_Date	Gender	Birth_Date
1	102493233	28/02/2000	F	01/09/1970
2	103310901	27/11/2001	F	01/04/1939
3	102545868	20/12/1999	F	06/11/1947
4	102530803	28/12/1998	F	28/05/1971

As a result, we get a dataset with the columns from both source datasets combined. The records are matched using the common **card_id** field:

	transaction_id	payment_method	timestamp	product_code	items_number	card_id	Registration_Date	Gender	Birth_Date
1	145419	Cash	19/05/2001 5:49 PM	000000078	1	100000111	24/07/2000	F	08/06/1964
2	121302	Cash	27/04/2001 11:23 AM	000000284	1	100000111	24/07/2000	F	08/06/1964
3	51466	Cash	20/02/2001 1:41 PM	000000488	1	100000111	24/07/2000	F	08/06/1964
4	143150	Cash	17/05/2001 4:15 PM	000000595	1	100000111	24/07/2000	F	08/06/1964
5	51036	Cash	20/02/2001 10:20 AM	000000622	1	100000111	24/07/2000	F	08/06/1964
6	205287	CreditCard	19/07/2001 4:50 PM	000000808	1	100000111	24/07/2000	F	08/06/1964
7	188736	DebitCard	28/06/2001 5:34 PM	000000537	1	100000199	06/11/1999	M	19/10/1962
8	231408	CreditCard	20/08/2001 4:46 PM	000000783	1	100000199	06/11/1999	M	19/10/1962

There is a question of how to deal with the records in each dataset that don't have matching values in the link column of the other dataset. For example, this happens if one dataset has more records than the other one. There are three options you can use for the join, depending on which records you wish to include in the result. These options are described below based on simple examples.

Examples of Join Types

Consider the example tables *Customers* and *Orders*:

Customers

Customer ID	Last Name
1001	Peterson
1002	Fischer
1003	Lee
1004	Hanson
1005	Brummett

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Orders

Order No.	Customer ID	Amount
77895	1003	123.33
44678	1003	356.33
24562	1001	122.76
34764	1015	198.31

The purpose of the join is to combine information from both tables. The Customer ID column is present in both tables. We will use it as the link column to join the tables.

Note that the customer with ID 1015 is no longer registered in the *Customers* table, and several customers in the *Customers* table have not placed any orders.

1. **Inner join** (include only records that exist in **both** datasets):

   All pairs of records which have the same Customer ID value are found in the *Customers* and *Orders* tables and every such pair is combined into a result record. The resulting table consists of all records combined this way. For simplicity, the resulting table contains only one copy of the join column (In this case, one column *Customer ID* instead of two columns *Customers_Customer ID* and *Orders_Customer ID*).

Customer ID	Customers_Last Name	Orders_Amount	Orders_Order No.
1001	Peterson	122.76	24562
1003	Lee	123.33	77895
1003	Lee	356.33	44678

2. Left outer join (Include **all** records from the first dataset)

   If we consider *Customers* to be the first (or "left") dataset, the resulting table is as follows. The result of a left outer join for tables *Customers* and *Orders* contains **all** records of the *Customers* table even if no matching record is found for any of them in the *Orders* table with the same Customer ID value. In that case the combined record in the resulting table will contain ***null*** in each column from Orders.

Customer ID	Customers_Last Name	Orders_Amount	Orders_Order No.
1001	Peterson	122.76	24562
1002	Fischer	(null)	(null)
1003	Lee	123.33	77895
1003	Lee	356.33	44678
1004	Hanson	(null)	(null)
1005	Brummett	(null)	(null)

3. Right outer join (include **all** records from the second dataset)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The concept of the right outer join is symmetric to that of the left outer join. Just substitute the first table for the second and vice versa. If we consider *Customers* to be the first dataset and Orders to be the second, the resulting table is as follows.

The result of a right outer join for tables *Customers* and *Orders* contains **all** records of the *Orders* table even if no matching record is found for any of them in the *Customers* table with the same Customer ID value. In that case the combined record in the resulting table will contain ***null*** in each column from *Customers*.

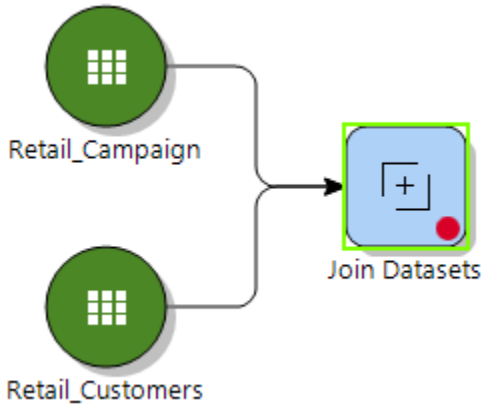
Customer ID	Customers_Last Name	Orders_Amount	Orders_Order No.
1001	Peterson	122.76	24562
1003	Lee	123.33	77895
1003	Lee	356.33	44678
1015	(null)	198.31	34764

The **Join** node in the **Manipulate** palette allows you to define the creation of a new dataset with fields from two source datasets in your project. By consecutively joining pairs of datasets, you can create a single dataset for analysis from many different input sources.

The **Join** wizard lets you choose the source datasets, specify the link - a single field that contains an identifier present in both datasets - and choose the attributes from each source dataset to be included in the new dataset. You can also specify which records to include in the output – only records that appear in both datasets, or all records from one of the source datasets.

Specifying the Join Operation

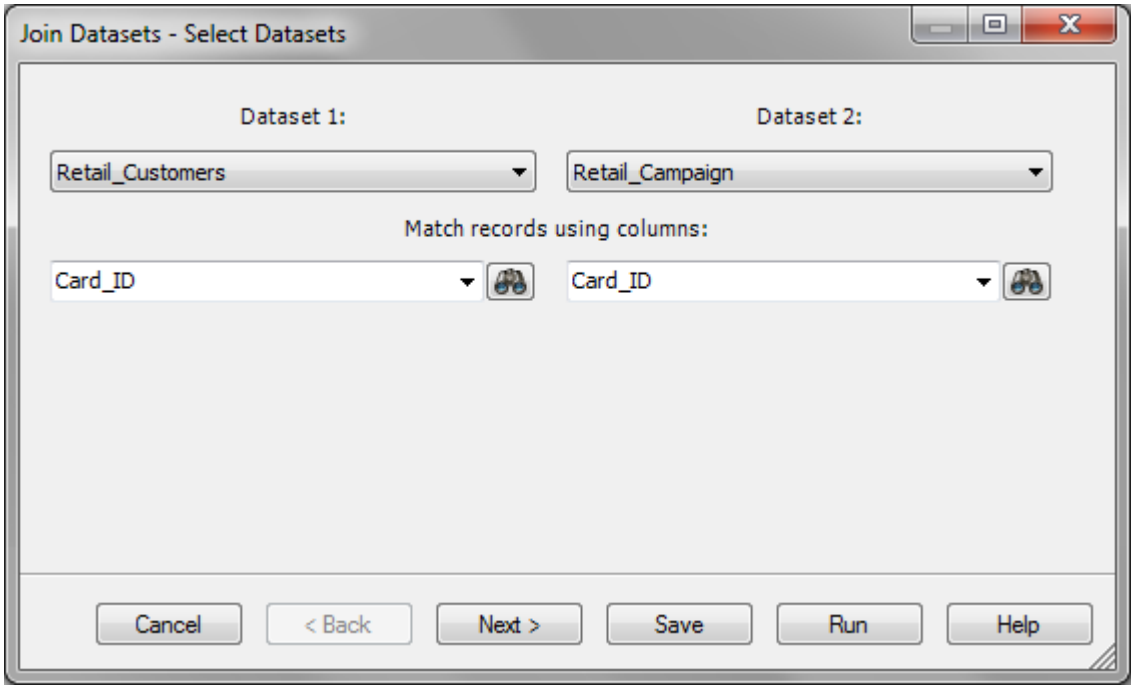
To introduce a join operation into your workflow, drag the **Join** node from the **Manipulate** palette onto the canvas and connect the two input datasets to the node.



To specify the join conditions, double-click the Join Datasets node or right-click on it and select **Modify**.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

In the first page of the wizard, select the datasets you wish to join. Select the join column in each dataset. The first pair of matching column names is selected by default. Make sure the join columns are of the same data type.

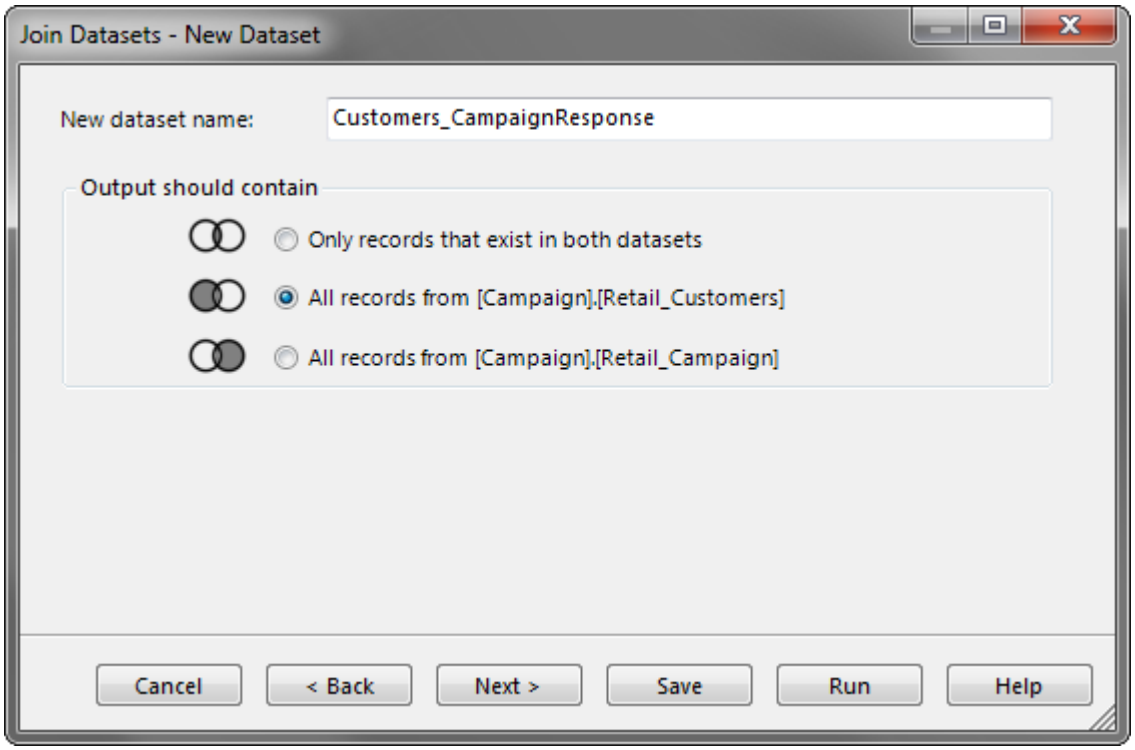


In the example screenshot above, we chose to join a dataset with campaign responses and a dataset with some demographics data for customers in a retail company's loyalty program. Both datasets have a loyalty card ID field. This will be useful to build a response model with demographics attributes as independent variables.

Click Next. In the *New Dataset* page, specify the name for the new dataset and select the type of join depending on which records you wish to include in the result. The available options are:

- include only records that exist in both datasets (*inner join*) - this is the default option
- include all records from Dataset 1 (*left outer join*)
- include all records from Dataset 2 (*right outer join*)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

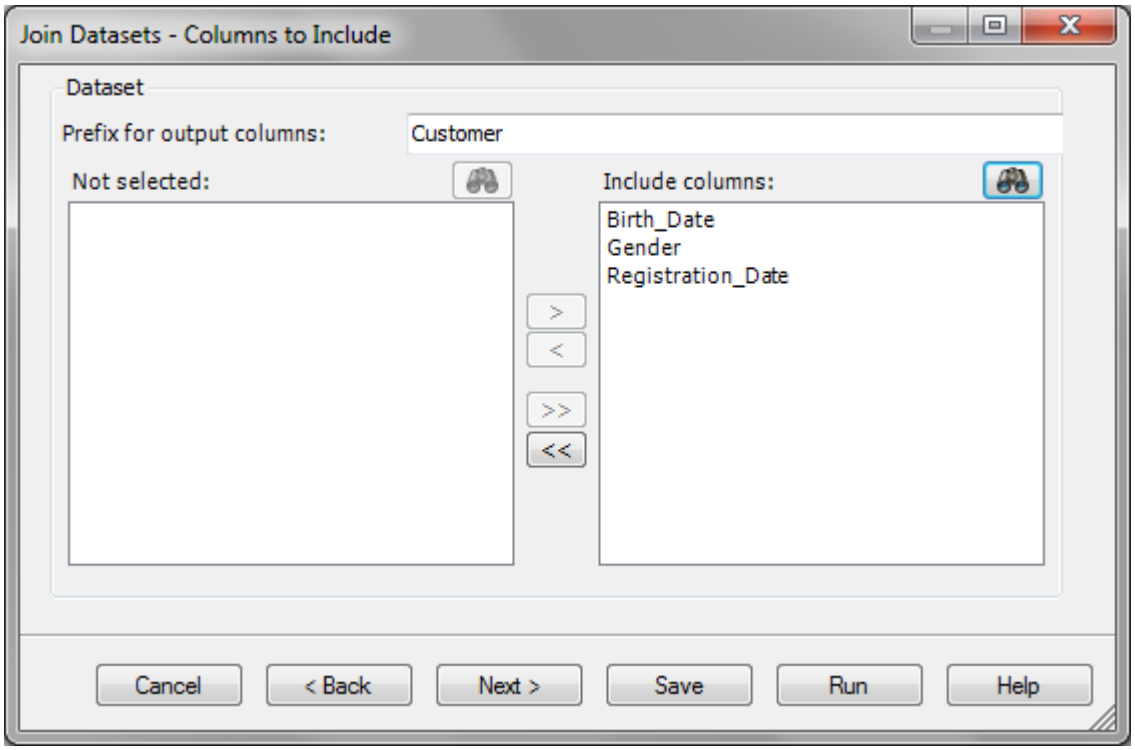


In the case where all records from one of the datasets are included (left or right outer join), any records that do not match will have null values inserted for attributes from the unmatched side of the join.

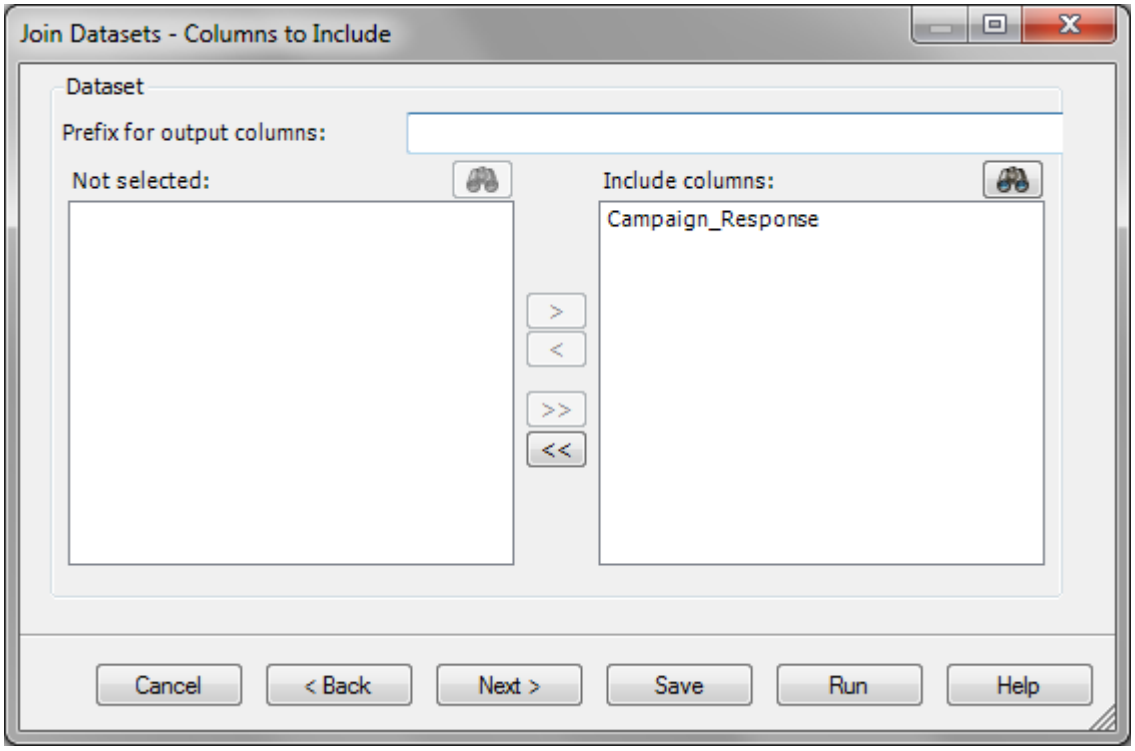
The default output dataset name is the concatenation of the two source dataset names with an underscore in between. Modify it if necessary.

In the next page, select the fields to include in the output from the first dataset and specify an optional prefix for these fields in the new dataset. The prefix is followed by an underscore. You will be able to modify the field names at the last wizard step if necessary.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

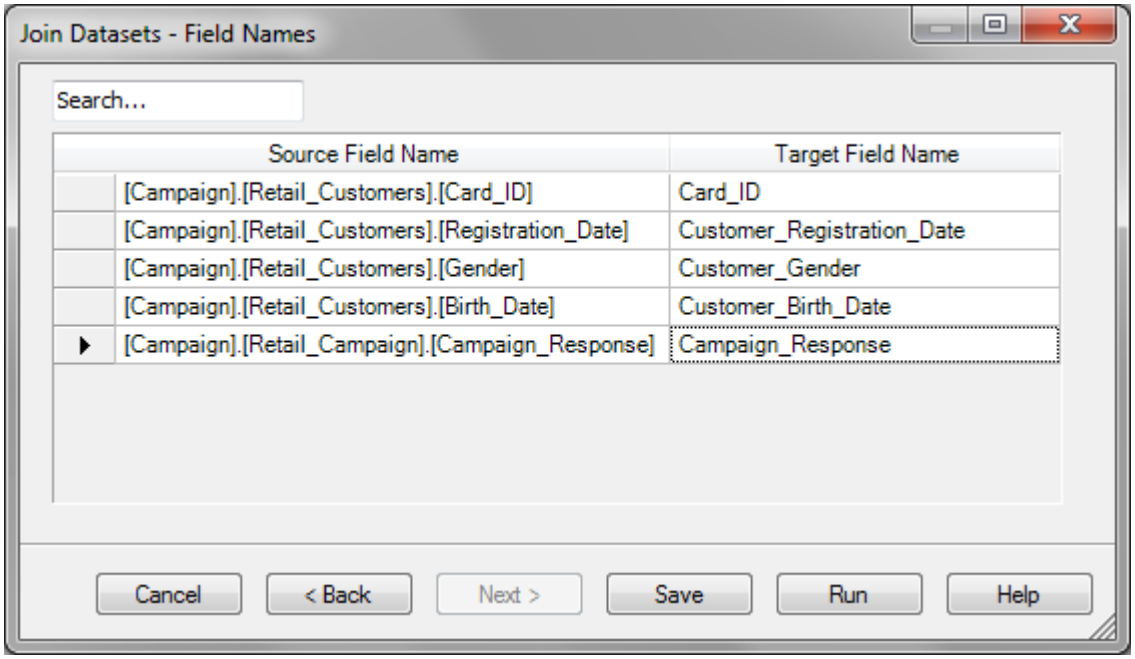


In the next page, select the fields to include in the output from the second dataset and specify an optional prefix for these fields in the new dataset. You will be able to modify the field names at the last wizard step if necessary.

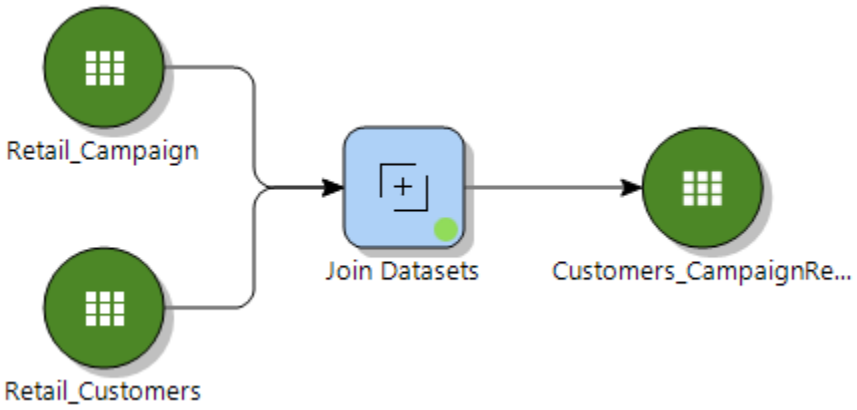


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

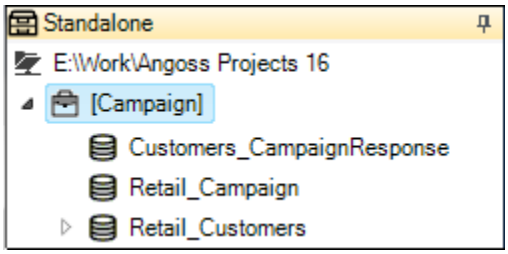
In the next wizard page you have an option of specifying field names for the new dataset if you want them to be different from the names generated automatically. If the resulting dataset represents a table in the In-Database Analytics mode, make sure the new field names comply with the naming standards of your target database. For example, field names in different databases may have restrictions regarding their length, special characters, etc.



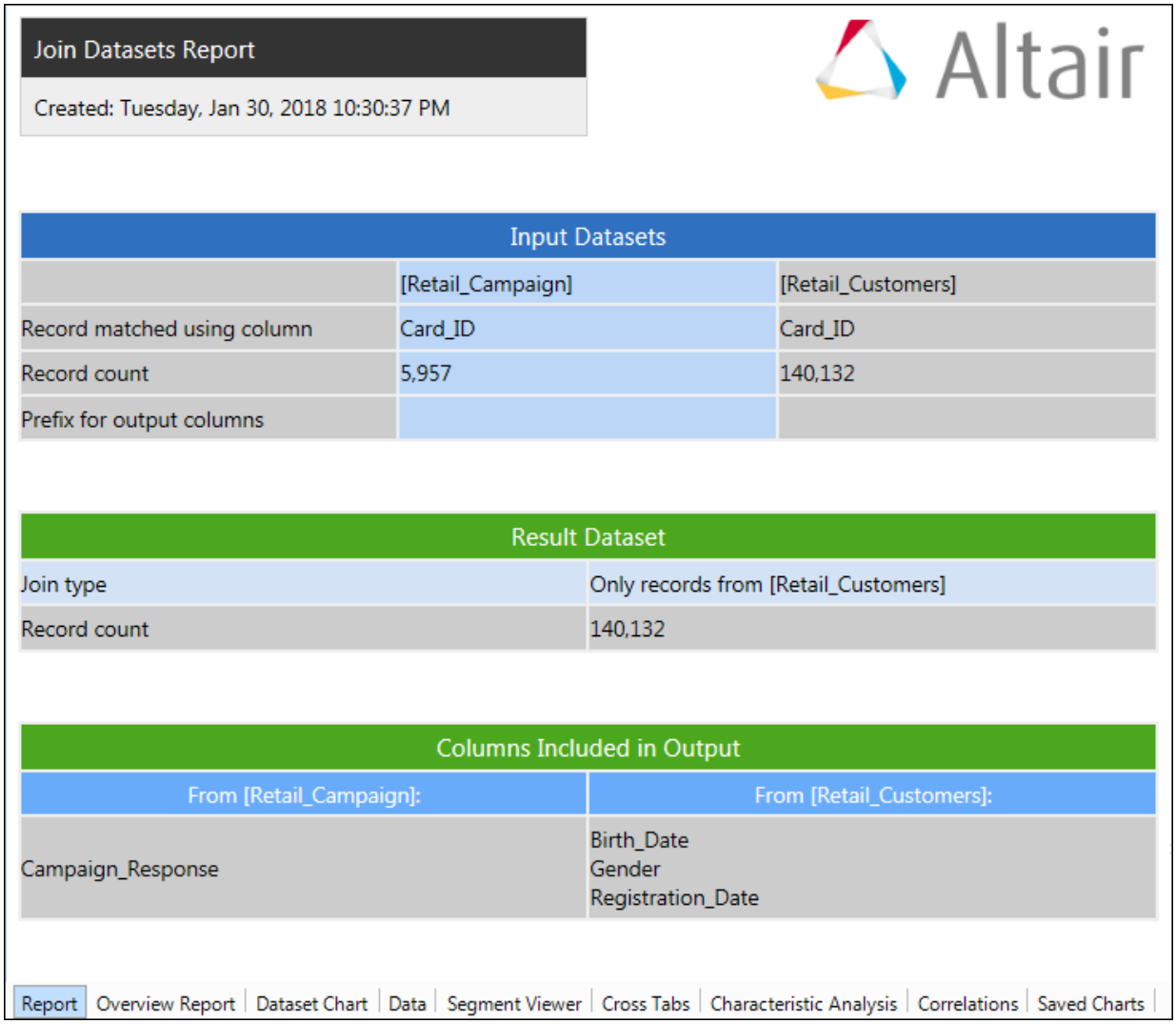
Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. In the example below, the *Customers_CampaignResponse* dataset node representing the left outer join of *Retail_Customers* and *Retail Campaign* appears on the right.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The *Report* tab of the resulting dataset provides the information on all the settings that were specified in the Wizard: the input and output datasets, the total number of records in each dataset, the join type, the join column, the specified column prefixes (if any), and the columns included in the output dataset.



The other tabs, such as *Overview Report*, provide the same functionality as in any other dataset, and all the usual dataset operations are applicable.

See also: Filtering Join

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Make Time Series

The **Make Time Series** node in the **Manipulate** palette produces *time series datasets* containing metadata that identifies one column as the time index and another column as the time label. For example, when making time series from a dataset containing weekly sales data, the user indicates the time variable in the input, and the wizard creates time index variables (series ID variables) for the user-defined periods such as weeks, months, or years, and performs the necessary data aggregation over these periods.

The series data in these time series datasets can then be visualized using special dataset views (integrated tabs) where you can plot time series charts and autocorrelation charts. These special views are preserved under variable transformations and other data preparation operations you may subsequently apply to the time series datasets in the workflow.

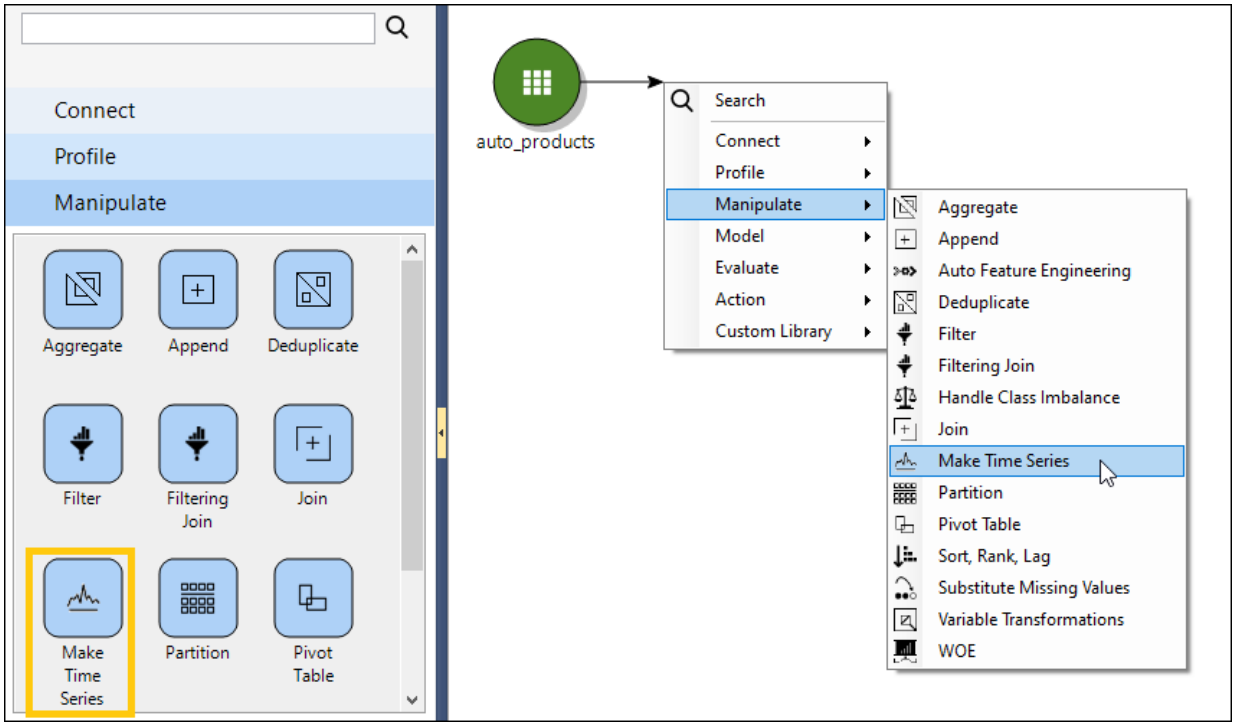
The input dataset should contain at least one Numeric, Integer, Date, or Timestamp field with no missing values.

To add a Make Time Series node to the workflow:

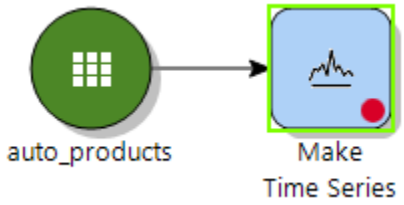
- Drag a connection arrow from the input dataset to any area on the workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Manipulate --> Make Time Series**, as in the example screenshot below. A new **Make Time Series** node will be created, and the wizard will open.

  OR:

- Drag the **Make Time Series** node from the **Manipulate** palette onto the canvas and connect the input dataset to it.



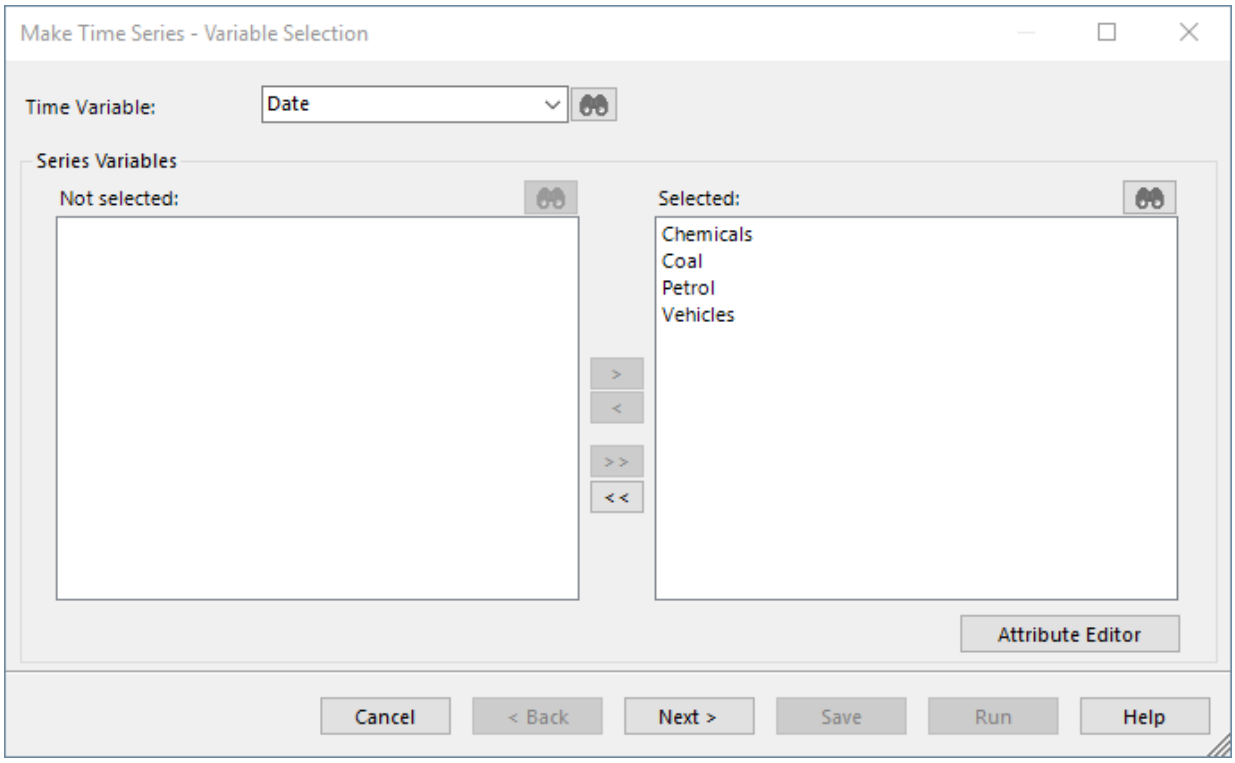
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If the Make Time Series node is already on the canvas, double-click it or right-click on it and select **Modify**.

In the first page of the wizard:

- Select the **Time** variable (any variable of type Date, Timestamp, Integer, or Number is admissible).
- Select the **Series** variables using the arrow buttons. You can select or deselect an individual variable by double-clicking on it.
- Click the **Attribute Editor** button if you need to change the role, usage, or aggregation type of any variable.



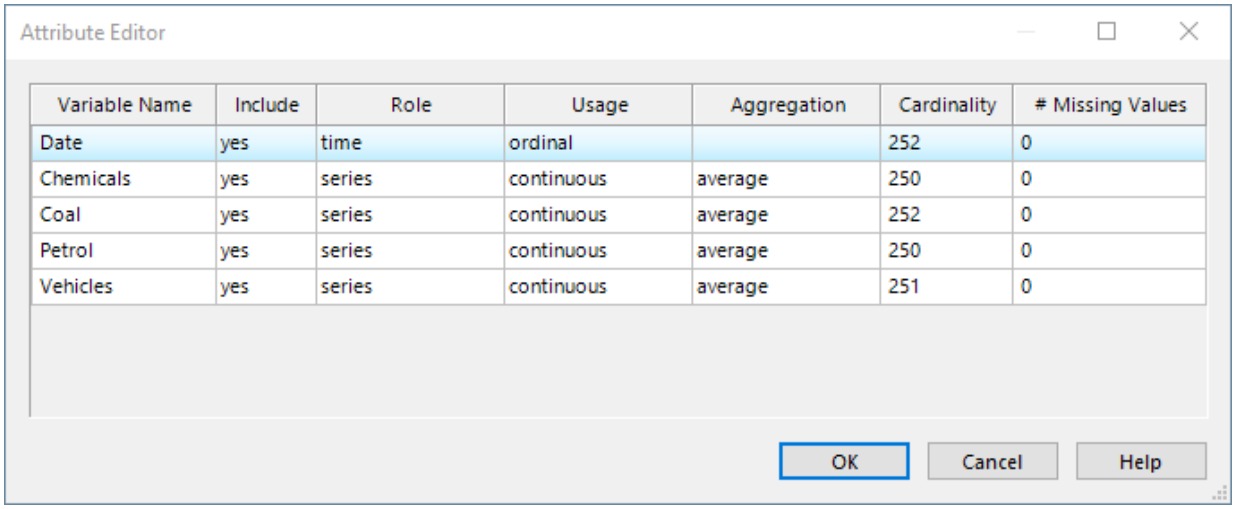
The **Attribute Editor** dialog shows the cardinality and the number of missing values for the selected variables, and allows to choose the following options for the variable properties:

- **Roles**: *time* and *series.* Exactly one variable with the *time* role must be included.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- **Usage options**: *continuous*, *discrete*, and *ordinal.* (The meaning of these options is essentially the same as in the Attribute Editor for Decision Trees. For the Time variable, the usage can only be *ordinal*)
- **Aggregate functions** for the series variables: *average, count, maximum, minimum, median, mode, standard deviation, sum, unique count, and variance.* The defaults depend on variable types and are the same as in the Aggregation node.

You can sort any field by clicking on its header.

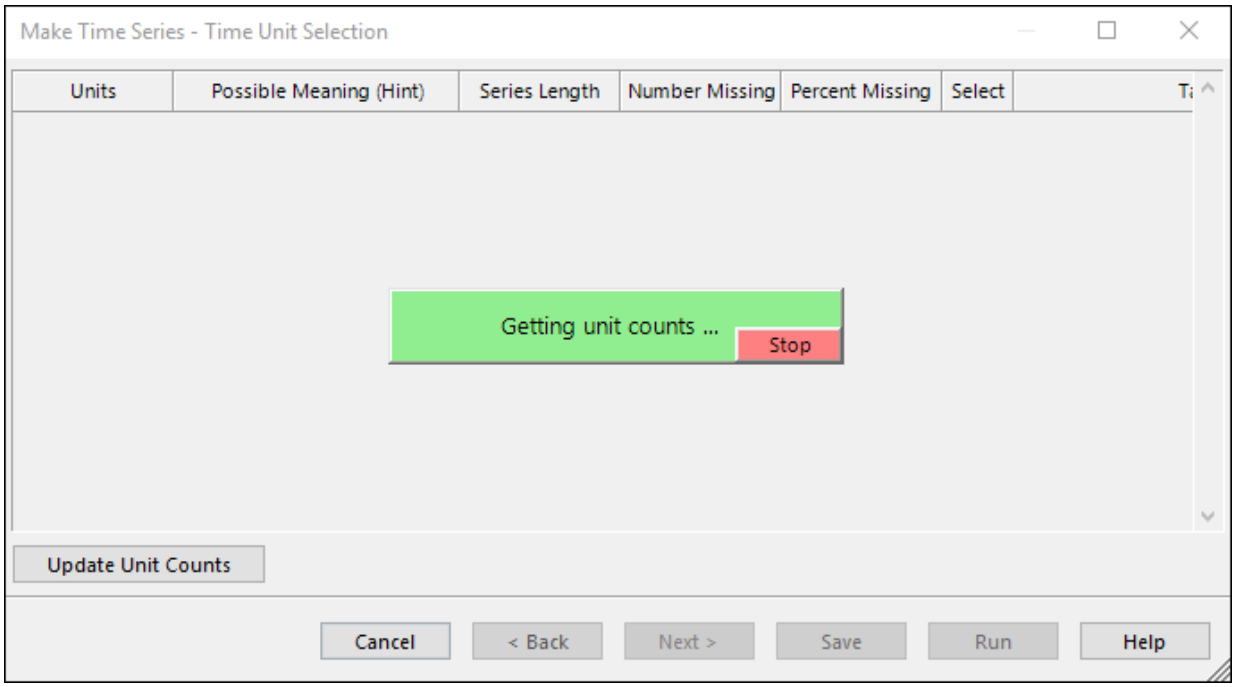


Click **OK** to apply the changes in Attribute Editor, then click **Next**.

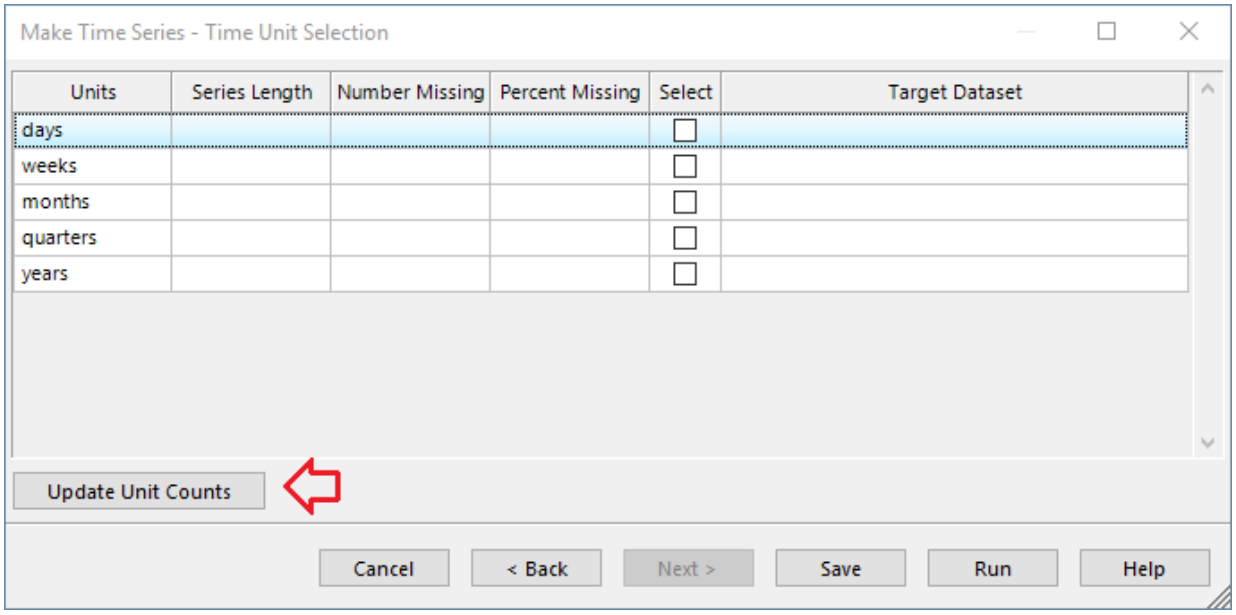
The next page allows you to select the desired time units and specify the output dataset names.

For large datasets, calculating the time units and related series statistics for the Time Unit Selection page may take some time. In this case a task status bar provides the **Stop** button in case you need to cancel the computation:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If you interrupt the computation by clicking the **Stop** button, the grid cells that have not been calculated yet will be blank. You can click the **Update Unit Counts** button to complete the calculation at your convenience:



Once the grid in the Time Series Selection page has been populated:

- Select the desired **time units** (temporal length of one time-step in the resulting time series, which determines the periods for data aggregation). The unit options depend on the data type of the selected *Time* variable. The smallest available unit is milliseconds. See below for details.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- In the **Target Dataset** field, edit the names of the target datasets if necessary. A separate output dataset will be created for each time unit selected.

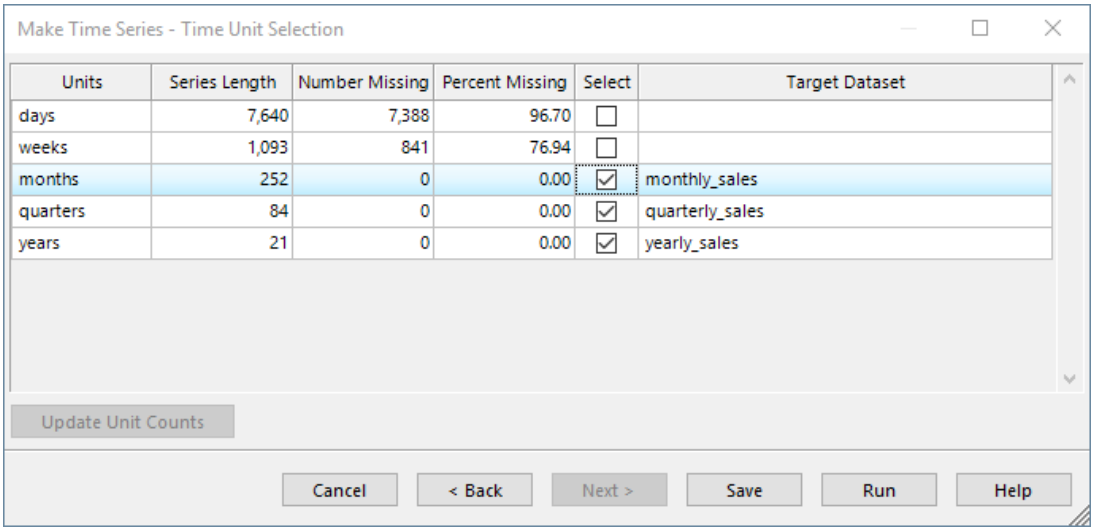
For reference, this page also shows the following information:

- Series Length: Number of time-steps in the resulting time series.
- Number Missing: Number of time-steps without any corresponding records within the input dataset.
- Percent Missing: 100 * (*Number Missing* divided by *Series Length*).

Unit options:

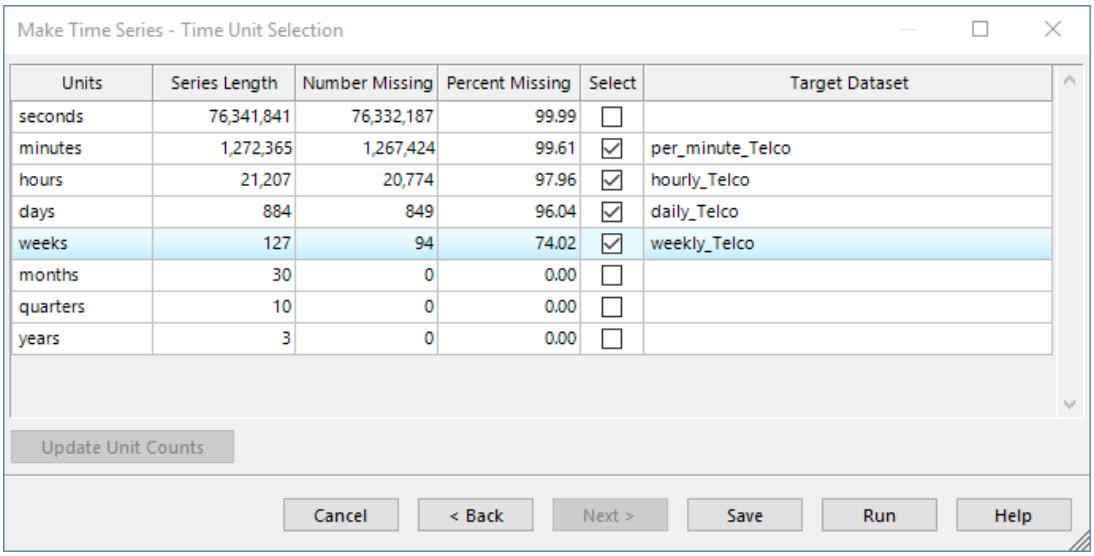
- If the selected Time variable has type *Date*, the available units are *days, weeks, months, quarters,* and *years*.
- If the selected Time variable is *Timestamp*, then *milliseconds*, *seconds, minutes, and hours* are available in addition to the Date unit types.
- If the selected Time variable is *Integer* or *Number*, the units depend on the cardinality and the distribution of the variable, and their meaning is suggested in the *Possible Meaning* column. See the example screenshot below.

Example 1: Time variable of type *Date:*

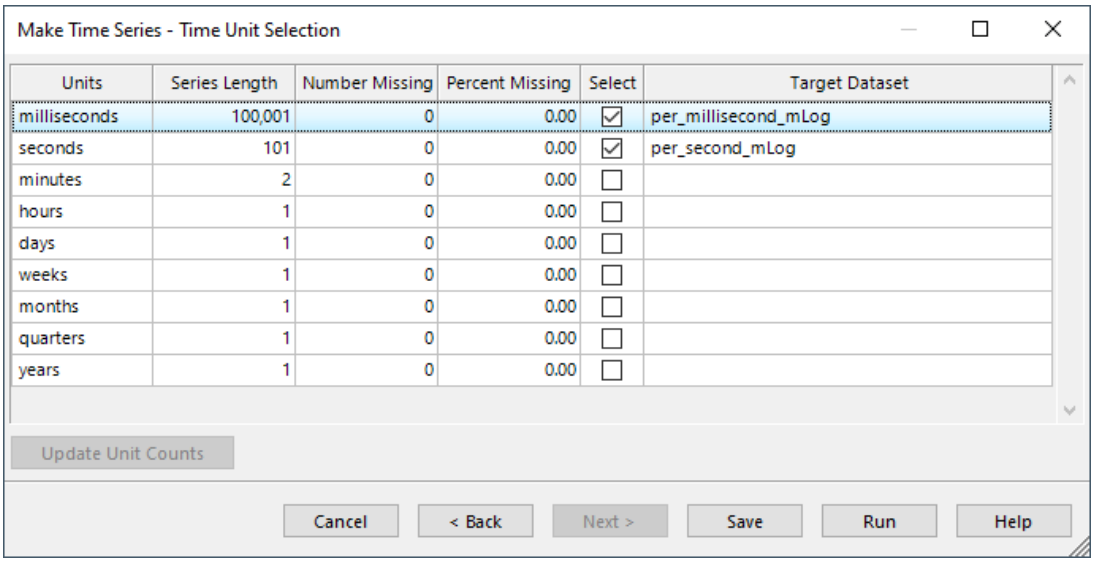


Example 2A: Time variable of type *Timestamp,* where the data has seconds as the smallest time unit (for example, YYYY-MM-DD hh:mm:ss):

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



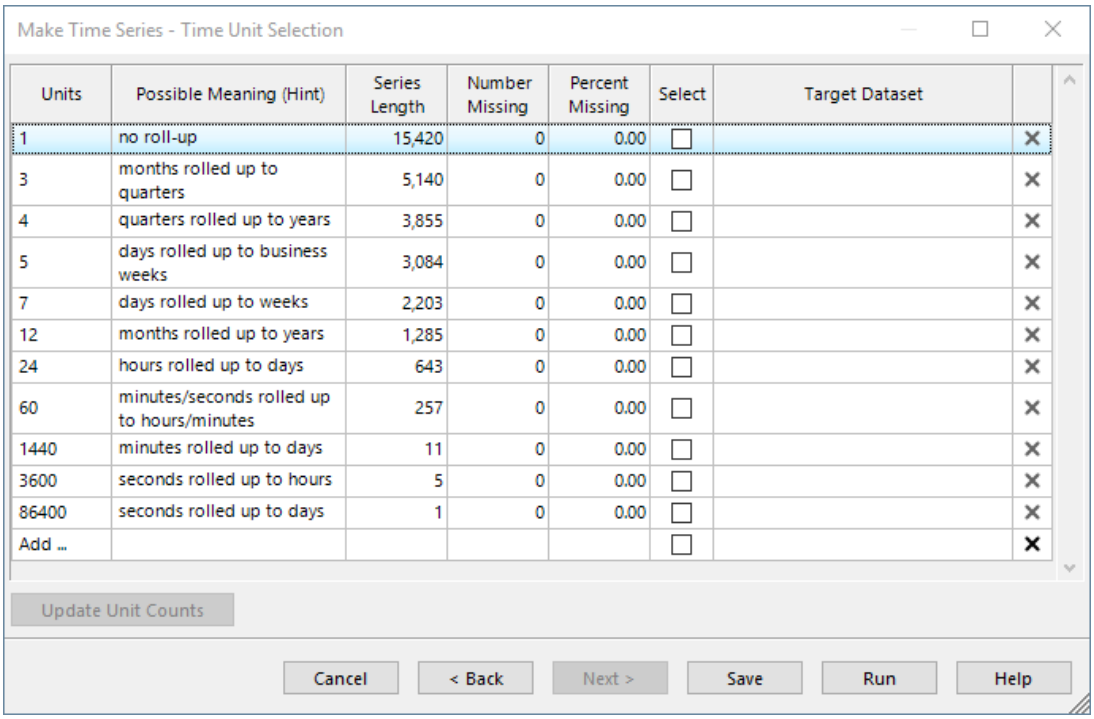
Example 2B: Time variable of type *Timestamp*, where the data has milliseconds as the smallest time unit (for example, YYYY-MM-DD hh:mm:ss.fff):



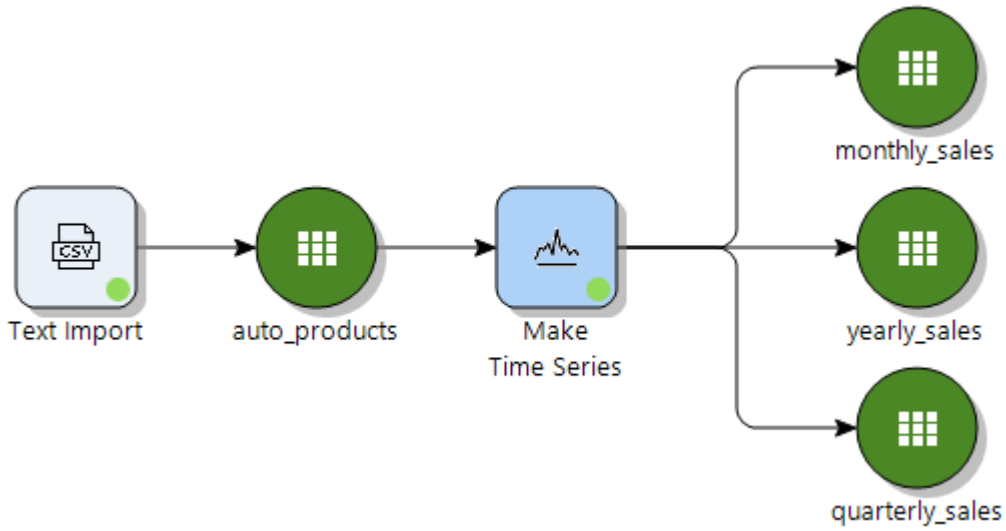
Example 3: Time variable of type *Integer* or *Number:*

Note: To add a new row, click on "**Add...**" in the last row of the Units column.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. In the example below, the Make Time Series node generates time series datasets monthly_sales, yearly_sales, and quarterly_sales.



The output datasets of a Make Time Series node have two additional tabs in its viewer – Time Series tab and Autocorrelation tab.

Double-click on the desired output dataset to view the created series and build the time series plots and autocorrelation plots if necessary.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

See also:

Time Series (Dataset Tab)

Autocorrelation (Dataset Tab)

Partitioning

Overview

You may need to partition your datasets for the following purposes:

1. Dataset partitioning is an important preparation stage in the ***modelling process***. A model is usually built on a ***training*** sample and then validated on a ***test*** (or ***validation***) sample in order to ensure the adequacy of fitting the model and to test its predictive power.

2. Dataset partitioning can be used for ***stratification*** (grouping records into homogeneous subgroups according to a condition imposed on the distribution of one of the variables). For example, selecting a sample where the distribution of females and males in census data is 40% / 60%. Stratification is a typical task, for example, in survey sampling for planning statistical research.

3. Partitioning can simply be a way to select smaller samples for analysis if your dataset is too large.

These tasks can be performed using the **Partition** node, which allows you to create *random* or *stratified* partitions of your datasets.

The partitioning features include:

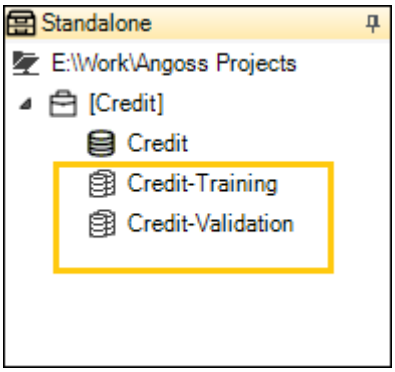
- Defining multiple partitions in a single dialog
- Editing and deleting partitions with full view of available records allocation and other partitions being defined
- Visualization and preview of statistics for the partitions to be inserted
- For stratified partitions, pie charts show the distribution of the stratification variable in each partition
- Support for data weighting: Taking into account the weight defined for the parent dataset

NOTE: Multiple partitions defined in one wizard session are ***disjoint***.

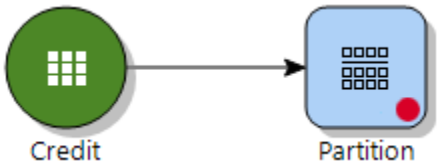
Each partition is:

- a separate dataset with standard properties and views such as Overview Report and Segment Viewer.
- displayed in the Project Explorer at the same level as the parent dataset, but distinguished from the main dataset using specific icons as shown below. In this example, the Credit-Training and Credit-Validation datasets were created as partitions of the Credit dataset:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



To introduce a partitioning operation into your workflow, drag the Partition node onto the canvas and connect the dataset to the node.



To define the partitions, double-click the Partition node or right-click on it and select **Modify**. This brings up the partition dialog. Proceed to Defining Partitions for the complete description of its functions.

Defining Partitions

This topic describes the Partition node functions in detail. For general information on partitioning, see About Partitioning.

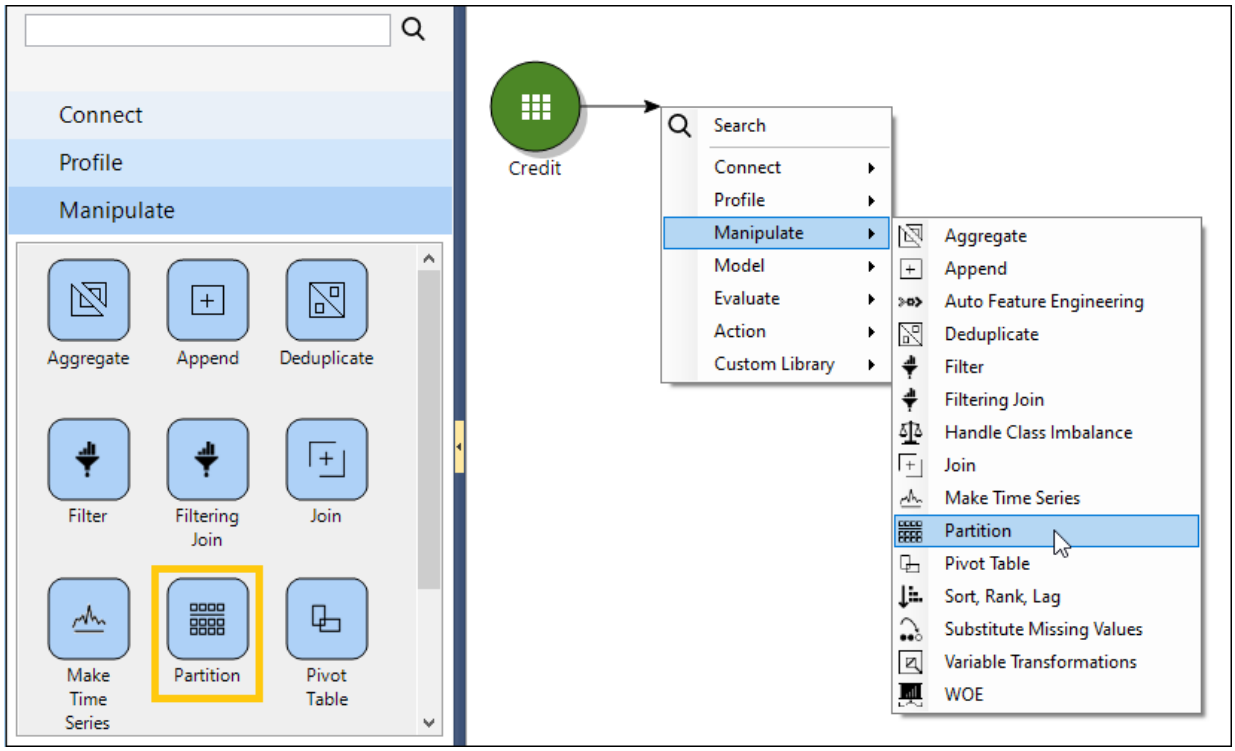
To introduce a partitioning operation into your workflow:

- Drag a connection arrow from the dataset node to any area on the Workflow canvas background. This brings up a context menu that allows you to select the desired operation. From the context menu, select **Manipulate --> Partition** as shown in the example below. A new **Partition** node will be created and the wizard will open.

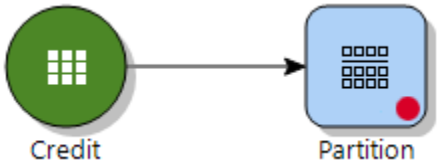
OR:

- Drag the **Partition** node from the **Manipulate** palette onto the canvas and connect the input dataset to the node

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



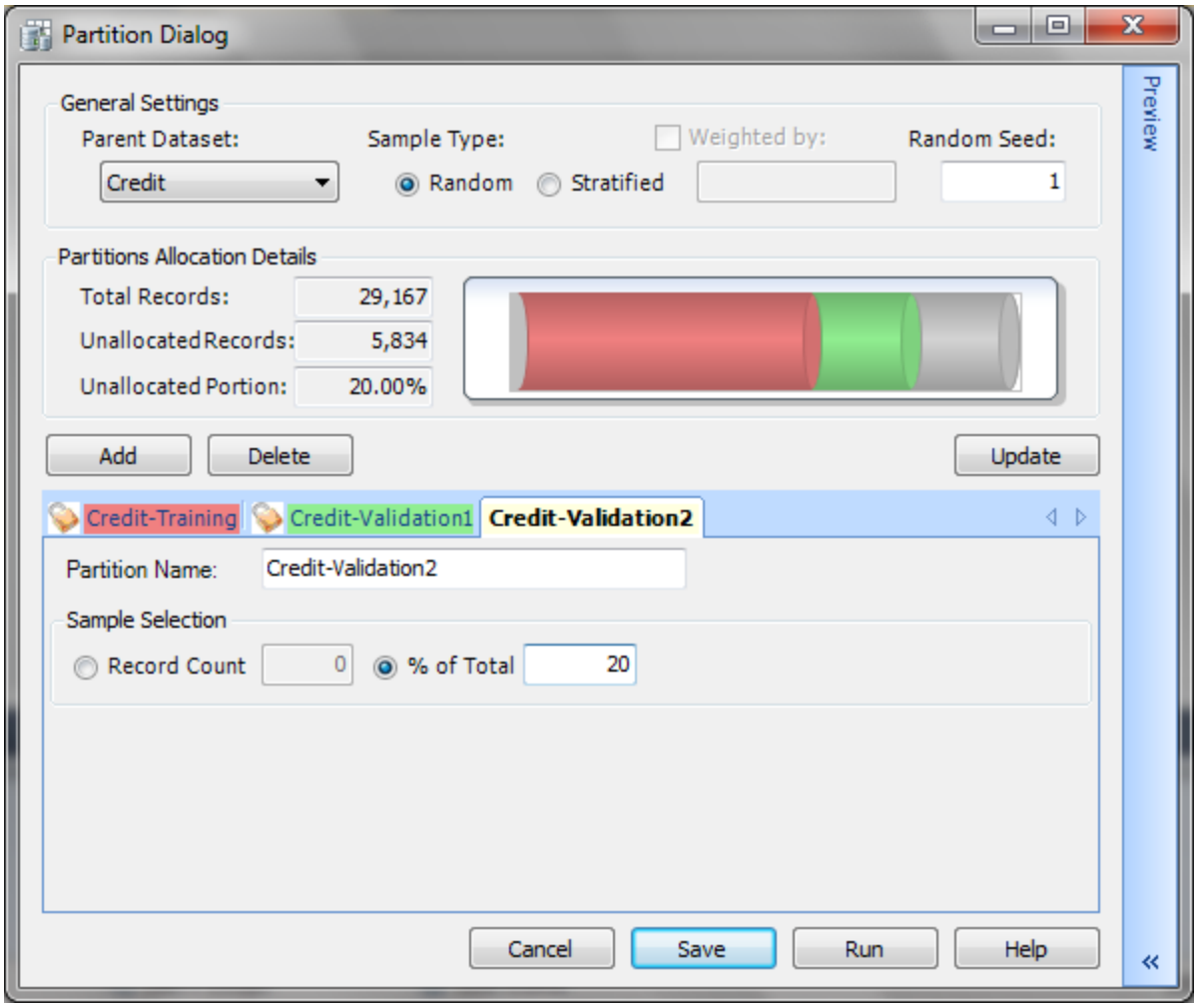
If the wizard does not open automatically, double-click the Partition node or right-click on it and select **Modify**.



In the partition dialog, you can define multiple partitions of random or stratified type, edit and delete partitions, view the partitioning progress with currently allocated and unallocated portions, and preview the summary statistics for the defined partitions before they are created.

Select the dataset to partition (Parent Dataset), the Sample Type (**Random** or **Stratified**) and click **Add** to create a new partition. A new tab will appear in the lower part of the dialog for specifying the partition parameters. Each partition is defined in a separate tab and you can easily switch between the partitions by clicking the corresponding tabs.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Parent Dataset	The dataset to be partitioned. All datasets in the current project are available for selection except the datasets in the In-Database Analytics mode. The dataset selected by default is the one that is currently open.
Sample Type	**Random**: Dataset records are selected at random for each partition. **Stratified**: A sample with a specific user-defined distribution of a selected field (called the *stratification variable*). More details on the stratified sample type are provided below. NOTE: The default Sample Type is *Random*. If you want to insert a stratified sample, select the *Stratified* option before adding partitions. When you add more than one partition at the same time, all of them are of the same sample type (either random or all stratified).
Random Seed	In general, the purpose of the random seed is to provide repeatable results of procedures that involve randomness. In the case of partitioning, when you insert random partitions of the same size, the resulting partitions may be different because of random record selection. However, if you specify the same value of the random seed each time, then the resulting partitions will be the same

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

because the first random selection of record indices will be remembered and applied in each case with the same seed value.

Weighted by If a weight is defined in the parent dataset, then the name of the weight field is displayed. The partitions being defined can be based on either weighted or non-weighted record count, the default being weighted. The weighted option is available only for the random sample type.

If this box is checked, records are selected for weighted samples based on the weight value: the higher the weight value, the higher the probability of the record being selected into the sample. Records with zero weight value are not included in the sample. Weights are not re-scaled in the resulting partition. (Note the difference from versions 7.0.x, which used a "stochastic sample" mode in the presence of non-integer weights, and the weight values in the resulting partitions could be re-scaled to provide the exact requested row counts.)

If the weighted option is selected, the resulting partitions inherit the weight field from their parent dataset. You can change the weight independently in each partition the same way as in any dataset after they are inserted and appear in the project outline.

Note that the total and unallocated number of records displayed in the Partition Allocation Details may not necessarily be integer (as the sum of the weight field is not necessarily integer). See the Weight topic for details.

Partition Name The name of the partition being defined.

Note: If you change the partition name, the name displayed in the partition tab header is not updated until you click Update or switch to a different partition or preview the partition summary.

Sample Selection The requested partition size can be specified either as a number of records or percentage of total.

Record Count: Enter the number of records to allocate for the partition.

% of Total: Enter the percentage of the total number of records in the dataset requested for the partition. Note that the percentage you entered may not necessarily represent an integer number of records (for example, if you request 50% while the total number of records is odd). For this reason, when you click Update or switch to a different partition or preview the partition summary, you may notice that sometimes the percentage you entered will be automatically adjusted to ensure that the number of records it represents is integer.

Add / Delete Click the **Add** button to add a new partition. This will create a new tab for specifying parameters for the new partition.

To delete the partition you are editing, click the **Delete** button. The active tab will be deleted. To delete another partition, click the corresponding tab and then click **Delete**.

Update Use the **Update** button when you want to save your changes, update the numbers of allocated records and the unallocated portion, and the

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

progress bar for the current record allocation. The progress bar highlights with different colors the portions of the dataset representing the partitions defined so far. (For example, in the screenshot above, the pink cylinder represents the *Credit_Training* partiton, and the green one represents the *Credit_Validation-1* partition. Note that *Credit_Validation-2* is still being edited, so it is not shown on the progress bar until an update is performed.)

When clicking **Update**, you also automatically exit the edit mode. If you need to make more changes, you can return to the edit mode by clicking the tab for the partition you want to edit.

Partitions Allocation Details

Total Records: Displays the total number of records in the parent dataset.

Unallocated Records: The number of available records in the parent dataset that have not been assigned to any partition.

Unallocated Portion (%): The percentage of available records in the parent dataset that have not been assigned to any partition.

NOTE: The Unallocated Records and Unallocated Portion (%) fields are not updated until you click **Update** or switch to a different partition.

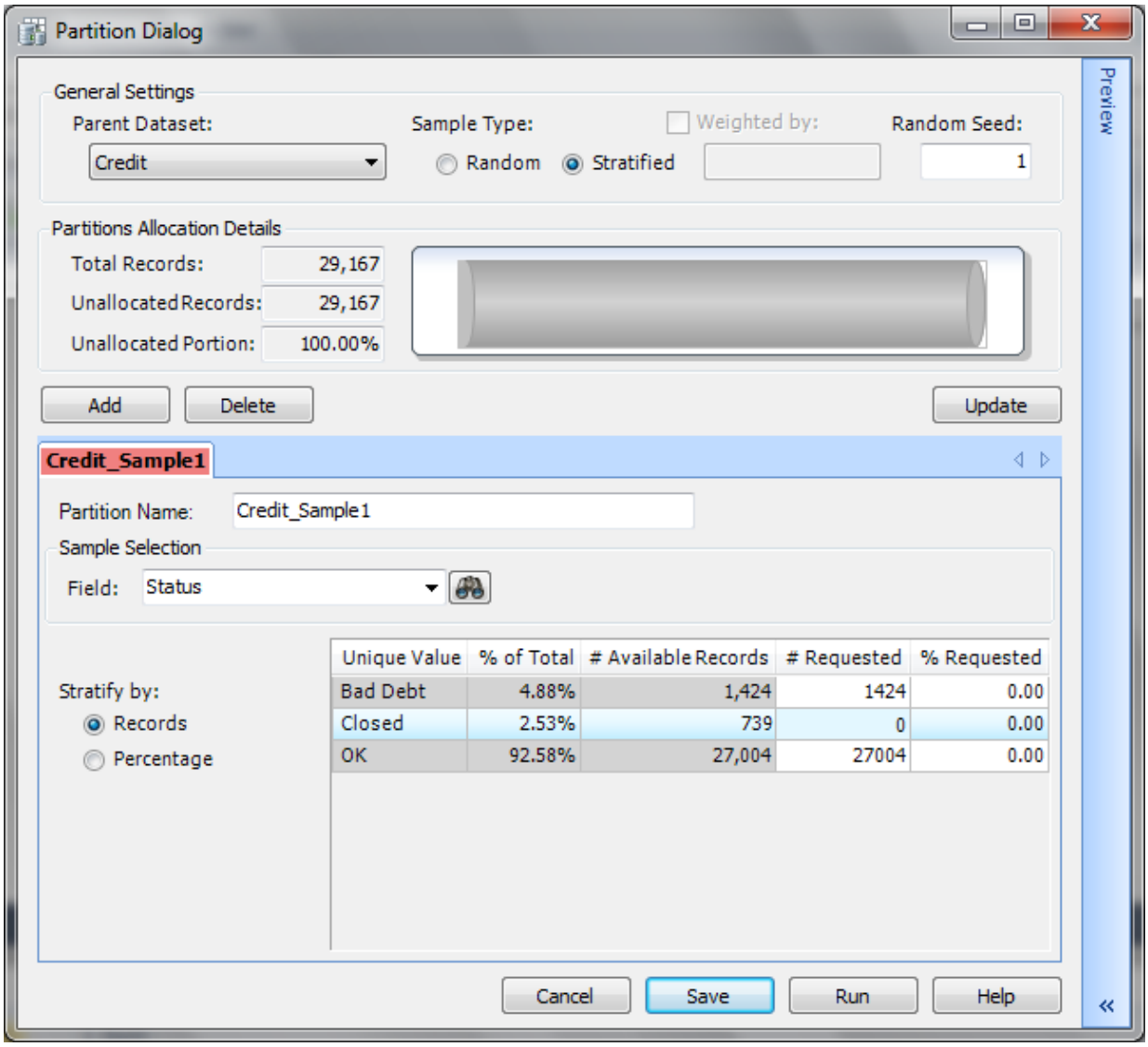
Sample Type: Stratified

To create a stratified sample, select the **Stratified** option for Sample Type in the Partition Dialog.

Click **Add** to create a new partition and specify the partition parameters in the new tab. When defining a stratified partition, you must select the stratification field and specify the requested distribution for this field in your sample. In this example (see Fig. 1), from a dataset with credit accounts information we select a sample that does not contain records representing closed accounts.

Fig. 1

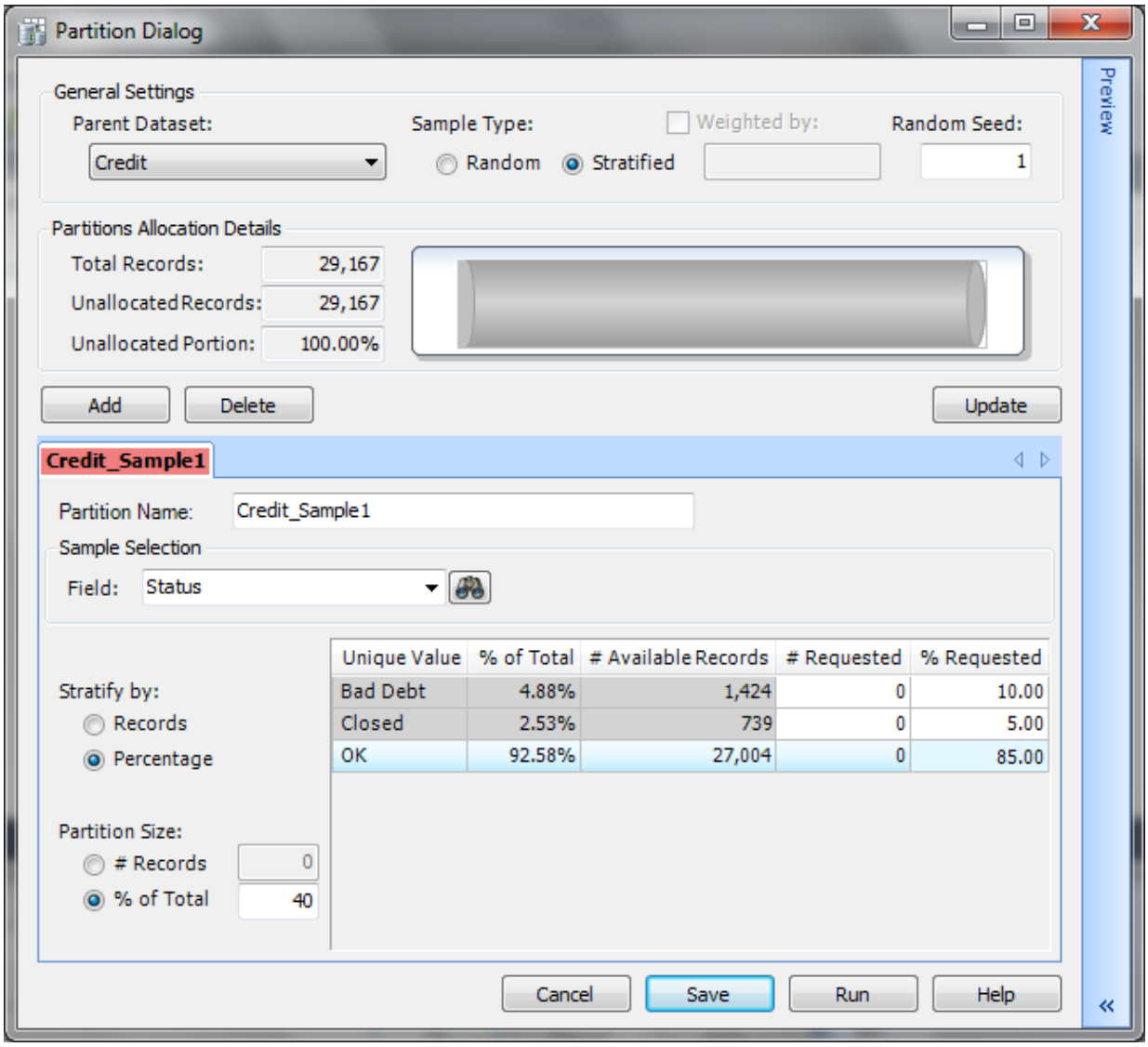
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If you want to specify the distribution in percentages rather than the number of records for each value of the stratification field, select the option **Stratify by Percentage**. In this case you will also need to specify the partition size. In this example (see Fig. 2), we select a sample with no closed accounts and 95% to 5% ratio of accounts with Status = Bad Debt and those with Status = OK, and define the partition size to be 80% of the parent dataset.

Fig. 2

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Field Stratification field. When you select the stratification field, the table showing the available and requested distribution of its values appears on the right. See below for the explanation of the table columns.

Note: Only fields of cardinality <=100 are available for selection as a stratification field.

Stratify by Records / Percentage **Records**: If you choose the Records option, you will need to specify the exact number of records requested for each of the values of the stratification field (shown in the Unique Value column on the right). Enter the numbers in the **# Requested** column.

Percentage: If you choose the Percentage option, you will need to specify the desired percentages of the values of the stratification field in your sample. The sum of the percentages entered for all values must add up to 100%.

Partition Size You will need to define the partition size only if you select the **Stratify by Percentage** option (see above).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

 # Records: Enter the number of records requested for the partition.

% of Total: Specify the partition size as a percentage of the parent dataset size.

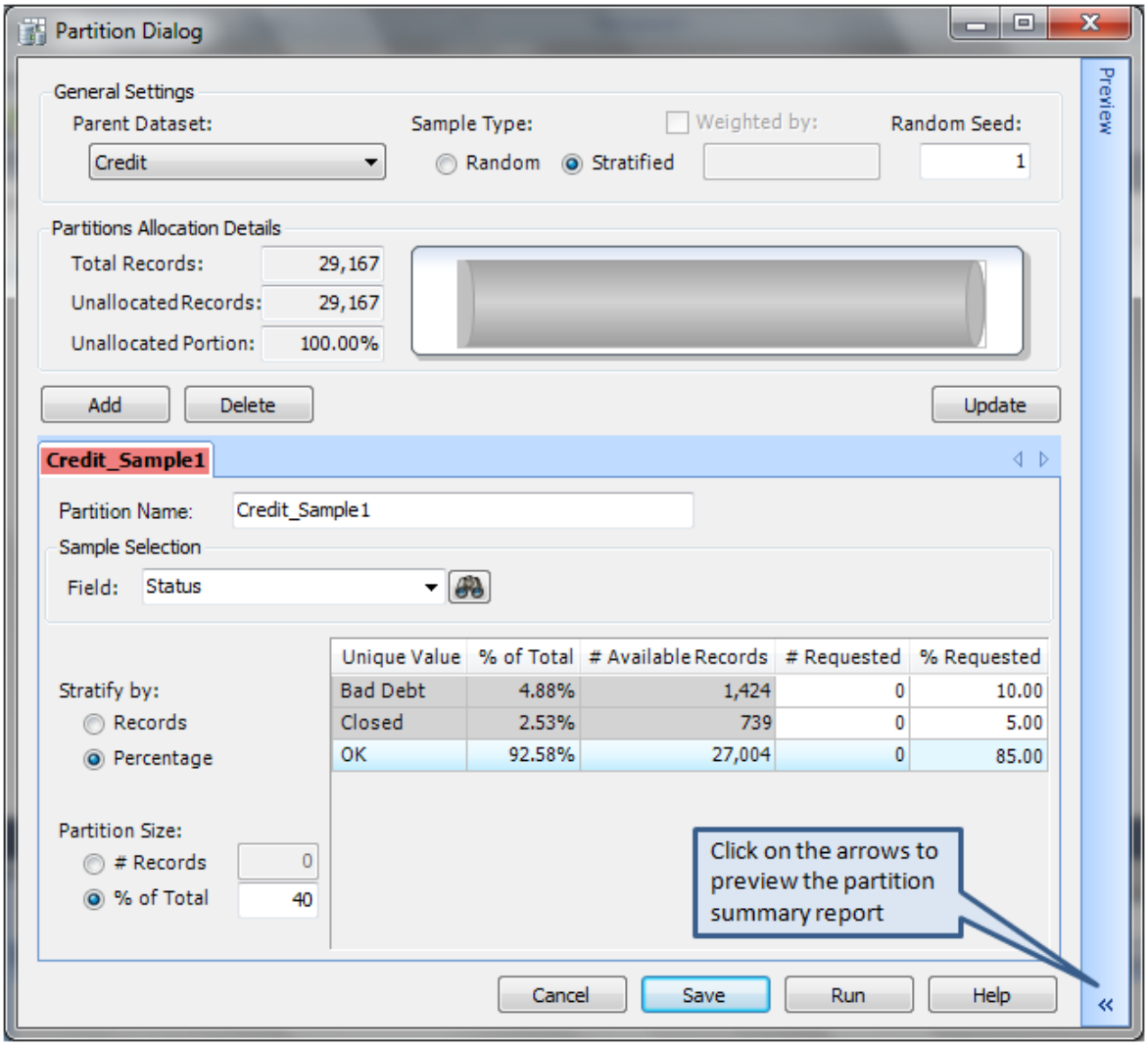
Note that the percentage you entered may not necessarily represent an integer number of records (for example, if you request 50% while the total number of records is odd). For this reason, when you click Update or switch to a different partition, you may notice that sometimes the percentage you entered will be automatically adjusted to ensure that the number of records it represents is integer.

Table: available and requested distribution of the stratification field

Unique Value	The values of the stratification field. Tip: If the stratification field is of String type and its values are too long for the column width, you can resize the partition dialog by dragging on its right side to make the columns wider.
% of Total	The percentage of the corresponding value in the stratification field in the parent dataset.
# Available Records	The number of available (unallocated) records with the corresponding value in the stratification field in the parent dataset.
# Requested	If you choose the **Stratify by Records** option, specify the exact number of records requested for each of the values of the stratification field in this column. In the above example (see Fig. 1), we requested all available records with Status = Bad Debt and Status = OK (1424 and 27004, respectively), and no records with Status = Closed for our sample.
% Requested	If you choose the **Stratify by Percentage** option, use this column to specify the desired percentages of the values of the stratification field in your sample. The sum of the percentages entered for all values must add up to 100%. In the above example (see Fig. 2), we requested the ratio of 5% to 95% for the values *Bad Debt* and *OK* in the Status field of our sample. Note that in this case you also need to specify the partition size.

Partition Summary Report

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



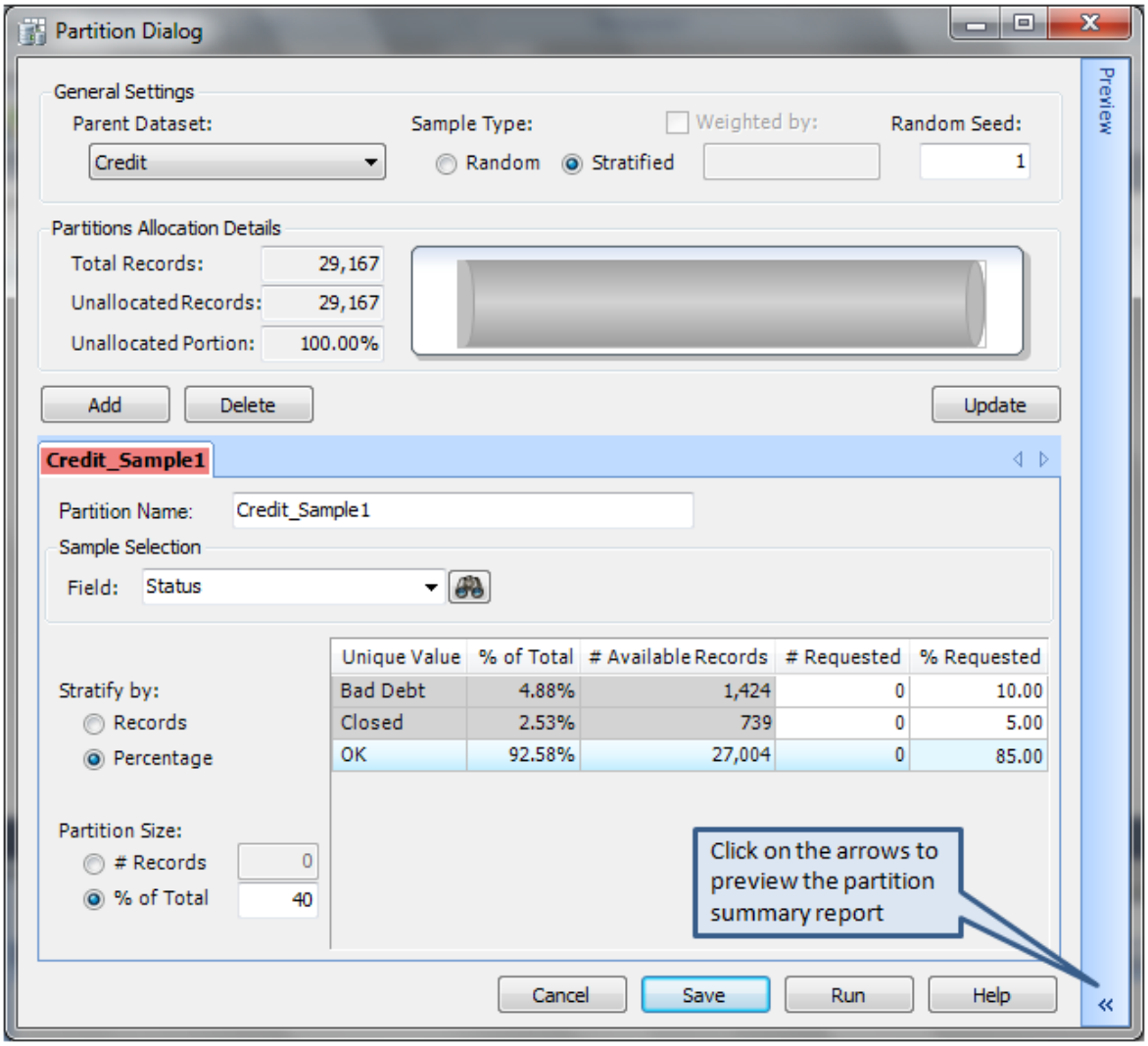
At any point while specifying the partition parameters, you can preview the partition summary report by clicking the arrow at the bottom of the vertical blue bar on the right side of the dialog. In the case of Stratified partitions, the summary includes the pie charts of the requested distribution of the stratification variable for each partition defined. See Partition Summary Report for details.

Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.

Partition: Summary Report

When defining partitions, you can preview the partition summary report by clicking the arrow at the bottom of the vertical blue bar on the right side of the Partition Dialog.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



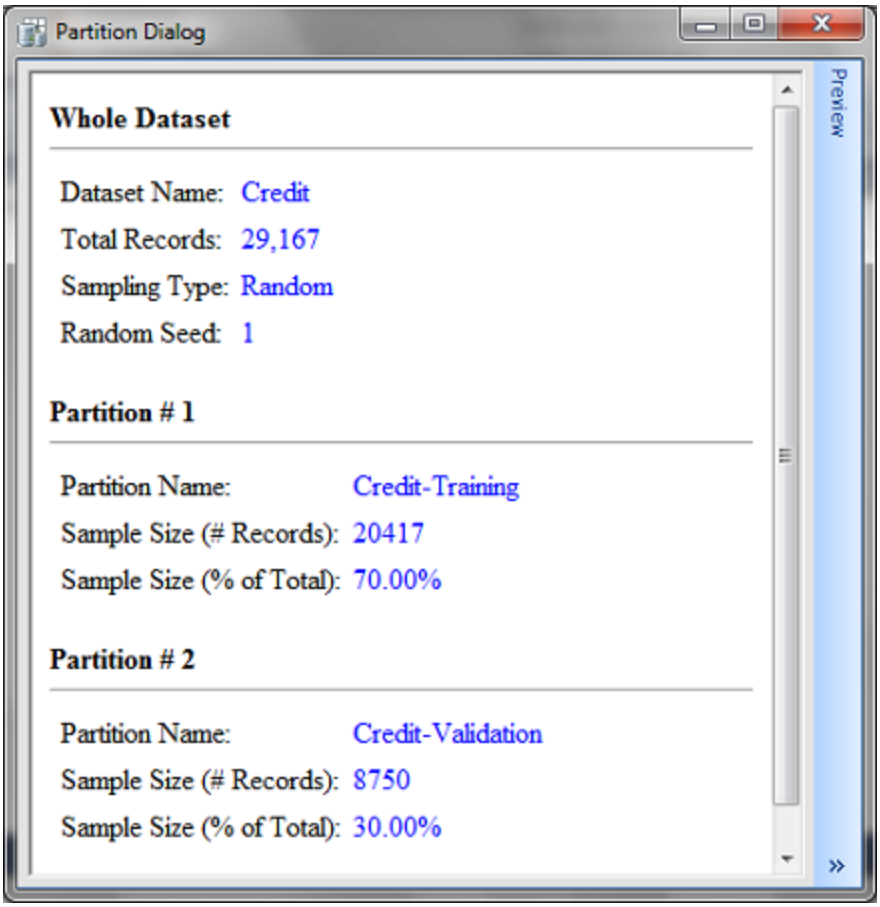
The partition summary report shows the statistics for all partitions you defined.

In the case of stratified partitions, the report includes the pie charts for the requested distribution of the stratification variable in all partitions you defined.

If there is a partition allocation error (for example, a zero size partition), an error message will be shown in the report. Click the arrow again to close the preview and correct the errors in the partition parameters.

Example 1: Summary report for the random sample type:

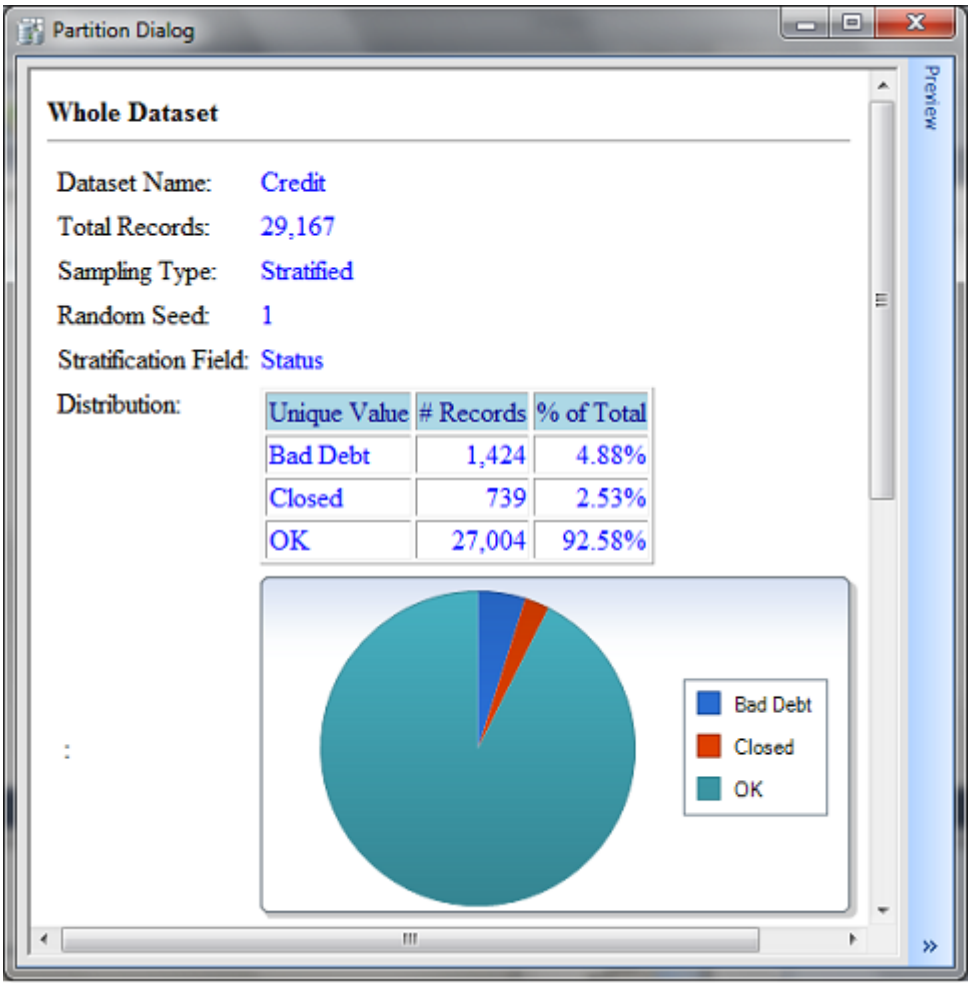
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Example 2: Summary report for the stratified sample type:

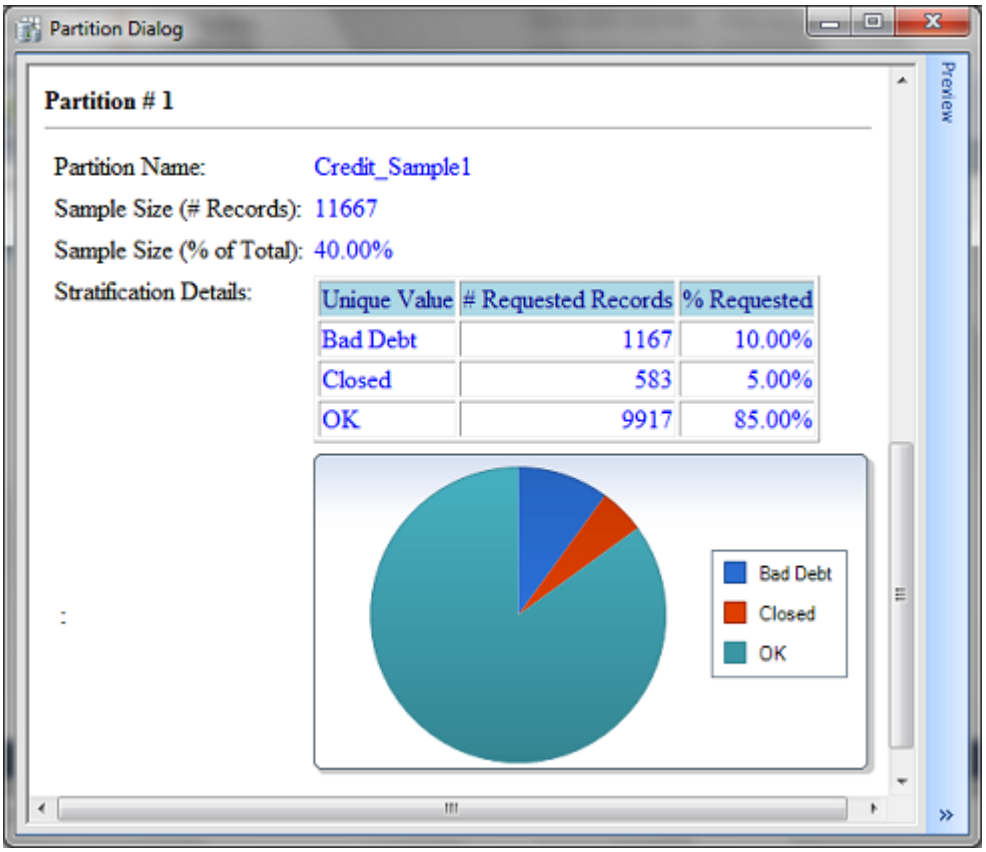
The first part of the report shows the original distribution of the stratification variable in the whole (parent) dataset in both tabular and graphical form (pie chart):

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Scroll down in the report window to view the requested distribution of the stratification variable in the partition(s) you defined:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click the arrow again to close the preview and return to editing the partition parameters.

See also:

[About Partitioning]

[Defining Partitions]

Pivot Table

The **Pivot Table** node in the **Manipulate** palette allows you to rearrange and summarize the data in Altair datasets to prepare it for modelling. The node wizard creates a new dataset with the user-defined grouping and summary statistics.

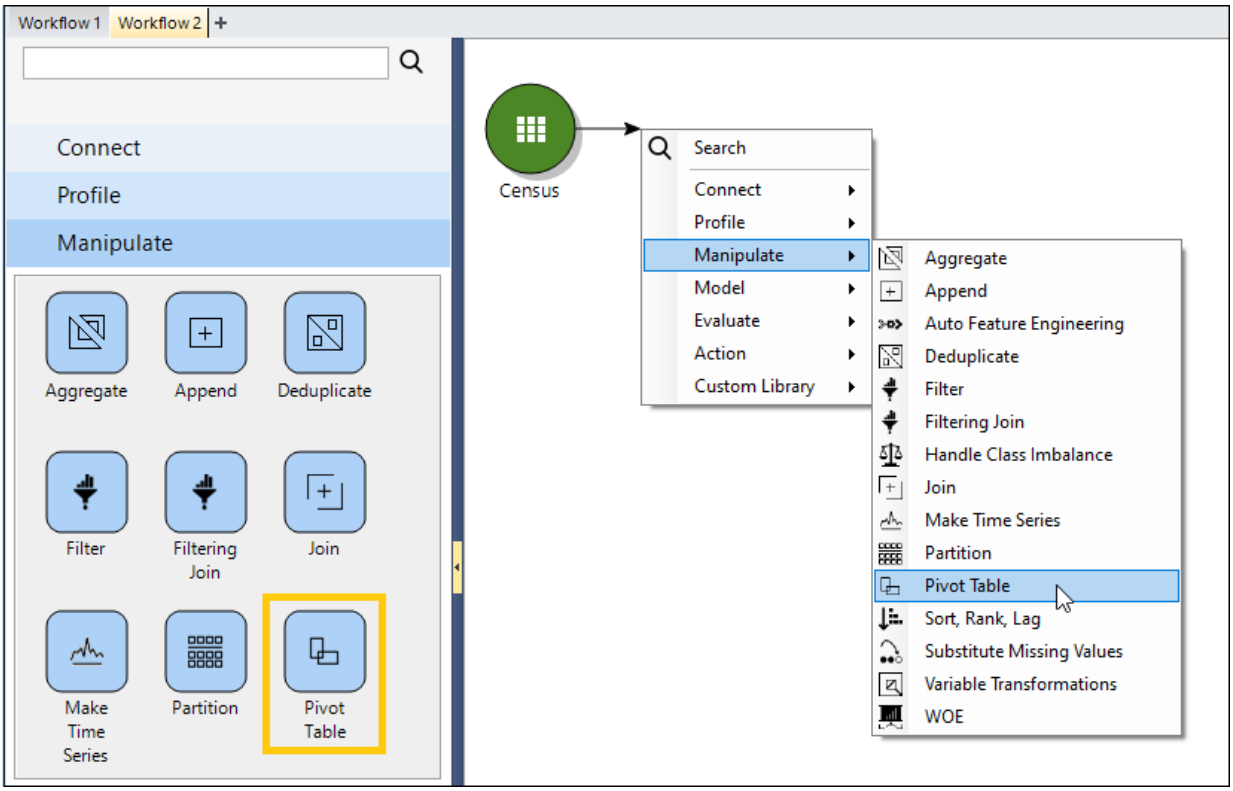
To add a Pivot Table node to the workflow:

- Drag a connection arrow from the input dataset to any area on the workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Manipulate --> Pivot Table**, as in the example screenshot below. A new **Pivot Table** node will be created, and the wizard will open.

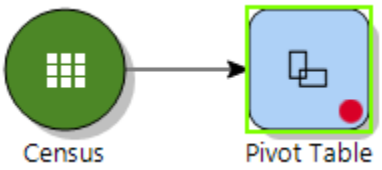
  OR:

- Drag the **Pivot Table** node from the **Manipulate** palette onto the canvas and connect the input dataset to it.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



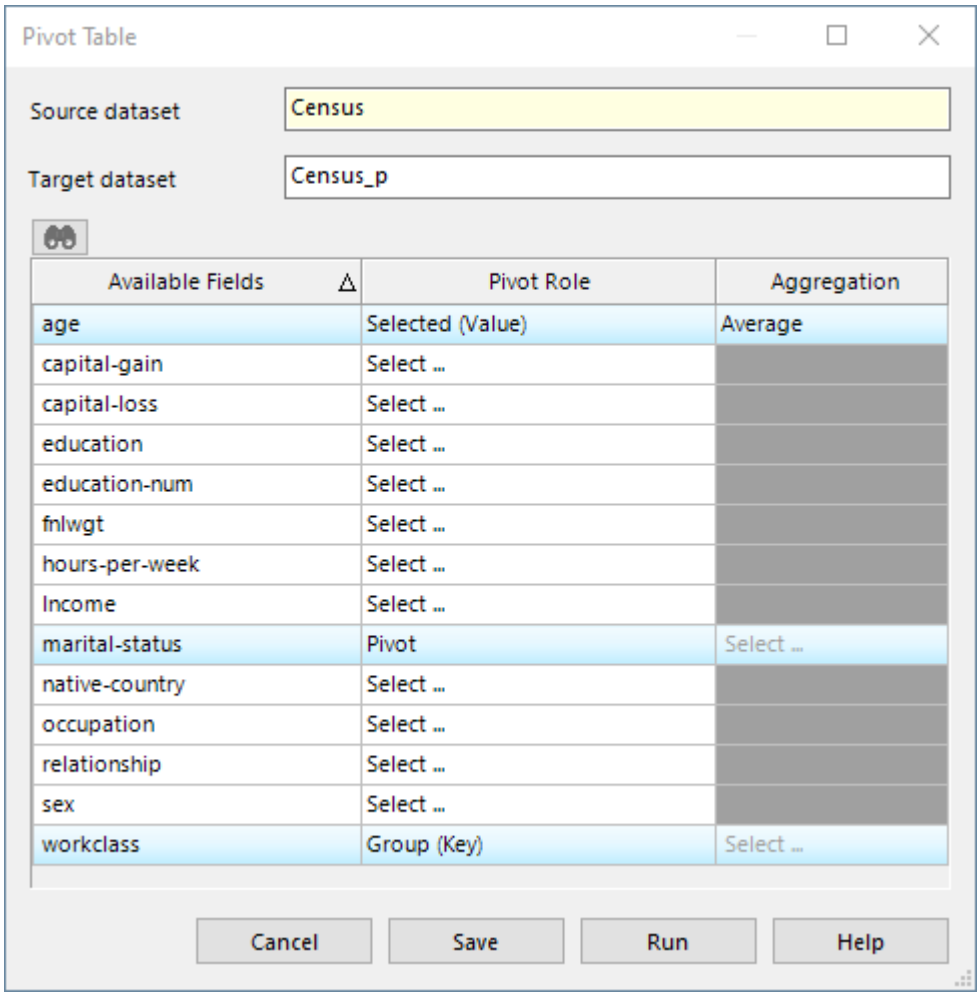
If the **Pivot Table** node is already on the canvas, double-click it or right-click on it and select **Modify** from the context menu to open the wizard.



In the Pivot Table wizard page:

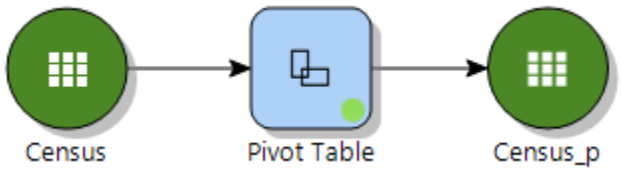
- Specify the name for the output dataset in the *Target dataset* field.
- Choose one or more fields as *Pivot* field(s). These are the fields to be pivoted. Their values will be the names of the new variables to be created.
- Choose one or more fields as *Group (Key)* fields. These are the field(s) to be used as keys, which will be used to group by when aggregating the records in the dataset.
- Choose one or more fields as the *Selected (Value)* field(s). These are the fields whose summary statistics will be calculated to become the values of the new variables. For each of these fields, choose the desired aggregation type in the *Aggregation* column. The available options for aggregation type depend on the data type of the fields:
    - For Number, Integer, Date, Time, and Timestamp fields: *Sum, Average, Count, Maximum, Minimum, Mode, Median,* and *Unique Count*
    - For String and Boolean fields: *Count, Maximum, Minimum, Mode, Median,* and *Unique Count*

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Note: If the number of new variables created from all fields with Pivot role exceeds 50, the wizard displays a warning that this will result in increased calculation time.

Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. Double-click the output dataset node to view the results.

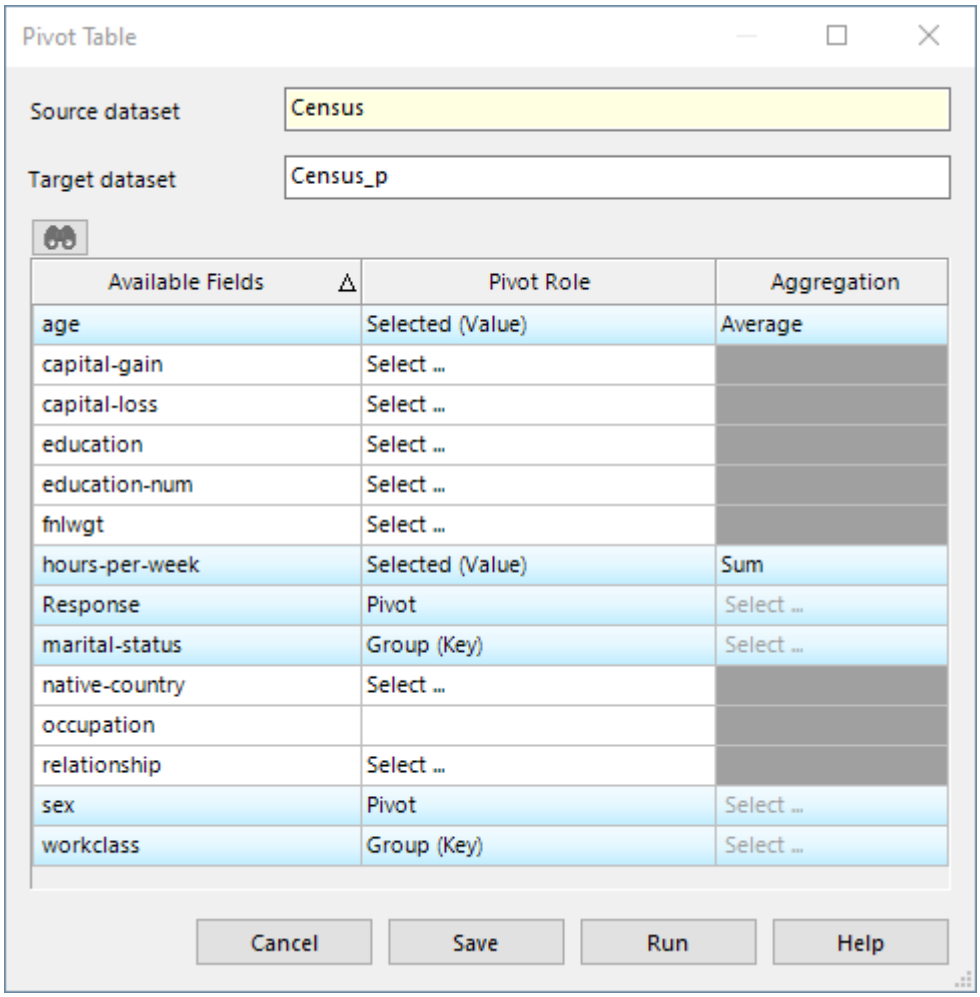


In the above example, the output dataset will show average age calculated for each workclass in each marital-status category. The marital-status categories will become the new variables in the output dataset. See below.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

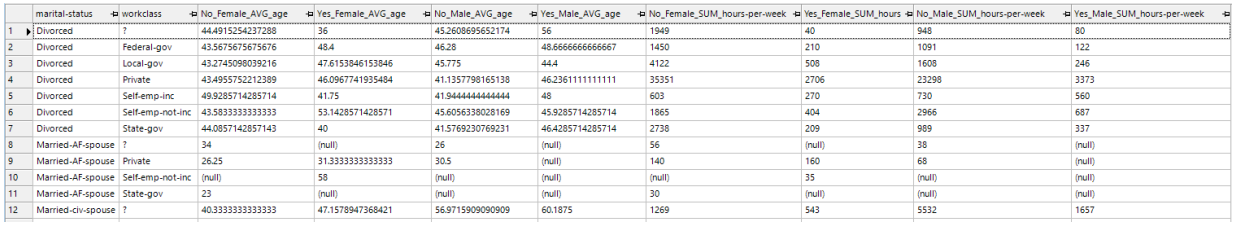
	workclass	Divorced_AVG_age	Married-AF-spouse_AVG_age	Married-civ-spouse_AVG_age	Married-spouse-absent_AVG_age	Never-married_AVG_age	Separated_AVG_age	Widowed_AVG_age
1	?	44.8705882352941	31.3333333333333	54.1168384879725	44.1333333333333	23.4264392324094	45.3548387096774	64.3623188405797
2	Federal-gov	45.1	(null)	45.188	49.5	33.9593495934959	43.9230769230769	57
3	Local-gov	44.2875	(null)	44.0194931773879	37.9090909090909	31.1940298507463	42.8918918918919	60.7222222222222
4	Never-worked	(null)	(null)	(null)	20	17.5	(null)	(null)
5	Private	42.9096651926721	28.8888888888889	41.5682345496731	39.5827814569536	27.7581957111166	39.3972602739726	58.7072368421053
6	Self-emp-inc	45.6739130434783	(null)	47.4192037470726	43.3333333333333	31.7093023255814	53.8	66.6666666666667
7	Self-emp-not-inc	45.3214285714286	58	48.1327231121281	45	33.6127450980392	43.625	62.5849056603774
8	State-gov	43.5094339622642	23	43.8443708609271	46.75	30.4887892376682	37.8636363636364	57.3333333333333
9	Without-pay	(null)	(null)	55.6	(null)	23	(null)	(null)

If multiple fields are chosen for each role, for example, two for *Group (Key)*, two for *Pivot*, and two for *Selected (Value)*, as shown below:



then the result will be as follows:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



	marital-status	workclass	No_Female_AVG_age	Yes_Female_AVG_age	No_Male_AVG_age	Yes_Male_AVG_age	No_Female_SUM_hours-per-week	Yes_Female_SUM_hours	No_Male_SUM_hours-per-week	Yes_Male_SUM_hours-per-week
1	Divorced	?	44.4915254237288	36	45.2608695652174	56	1949	40	948	80
2	Divorced	Federal-gov	43.5675675675676	48.4	46.28	48.6666666666667	1450	210	1091	122
3	Divorced	Local-gov	43.2745098039216	47.6153846153846	45.775	44.4	4122	508	1608	246
4	Divorced	Private	43.4955752212389	46.0967741935484	41.1357798165138	46.2361111111111	35351	2706	23298	3373
5	Divorced	Self-emp-inc	49.9285714285714	41.75	41.9444444444444	48	603	270	730	560
6	Divorced	Self-emp-not-inc	43.5833333333333	53.1428571428571	45.6056338028169	45.9285714285714	1865	404	2966	687
7	Divorced	State-gov	44.0857142857143	40	41.5769230769231	46.4285714285714	2738	209	989	337
8	Married-AF-spouse	?	34	(null)	26	(null)	56	(null)	38	(null)
9	Married-AF-spouse	Private	26.25	31.3333333333333	30.5	(null)	140	160	68	(null)
10	Married-AF-spouse	Self-emp-not-inc	(null)	58	(null)	(null)	(null)	35	(null)	(null)
11	Married-AF-spouse	State-gov	23	(null)	(null)	(null)	30	(null)	(null)	(null)
12	Married-civ-spouse	?	40.3333333333333	47.1578947368421	56.9715909090909	60.1875	1269	543	5532	1657

See also:

[About Data Manipulation]

[Aggregating Data]

Sort, Rank, Lag

The **Sort, Rank, Lag** node in the **Manipulate** palette allows you to sort data, create ranking fields, and lagged variables. The wizard creates a new dataset with sorted and ranked data and performs the necessary grouping. The group-by and lagged variable creation steps are optional and can be skipped.

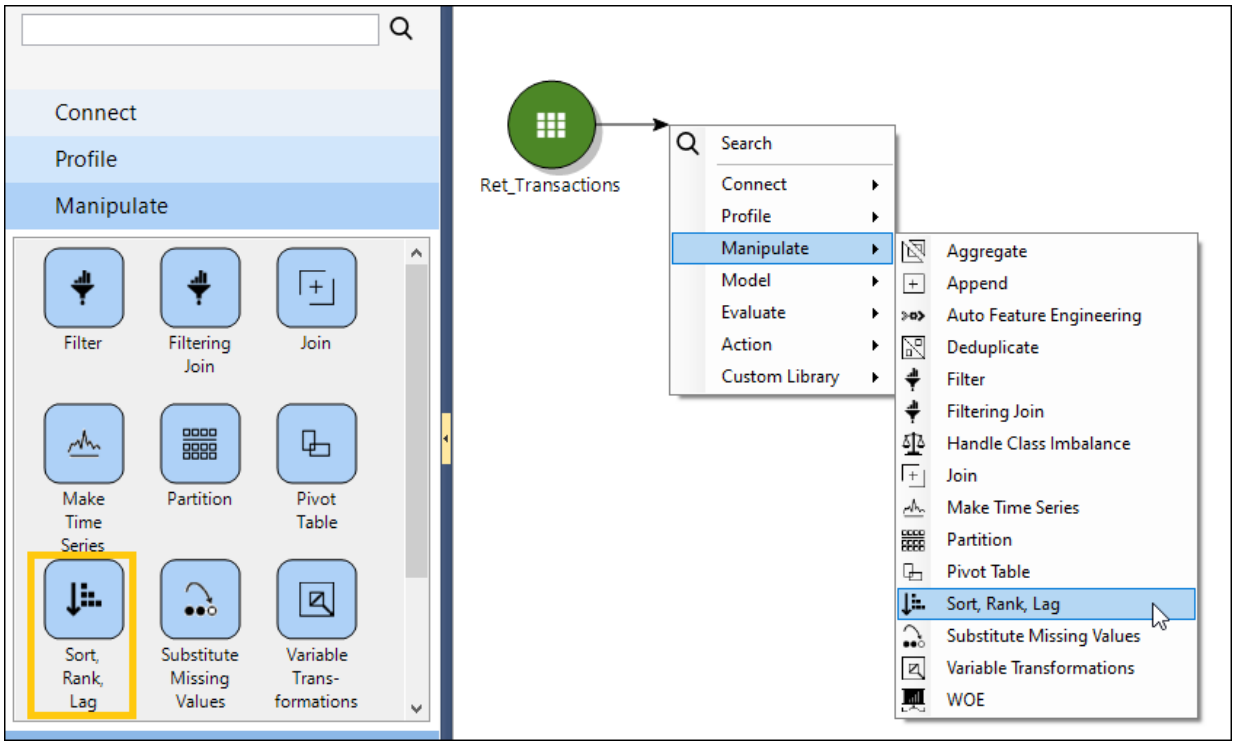
To add a Sort, Rank, Lag node to the workflow:

- Drag a connection arrow from the input dataset to any area on the workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Manipulate --> Sort, Rank, Lag**, as in the example screenshot below. A new **Sort, Rank, Lag** node will be created, and the wizard will open.

  OR:

- Drag the **Sort, Rank, Lag** node from the **Manipulate** palette onto the canvas and connect the input dataset to it.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

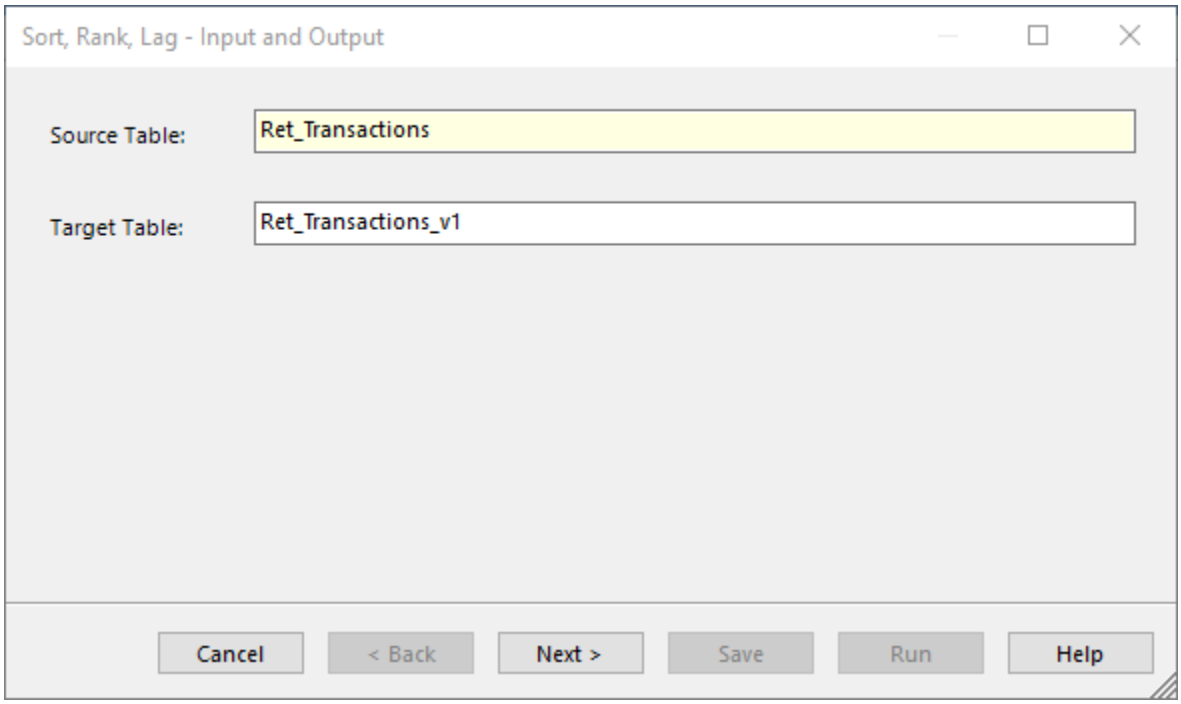


If the **Sort, Rank, Lag** node is already on the canvas, double-click it or right-click on it and select **Modify** from the context menu to open the wizard.



Specify the name for the output dataset in the first page of the wizard.

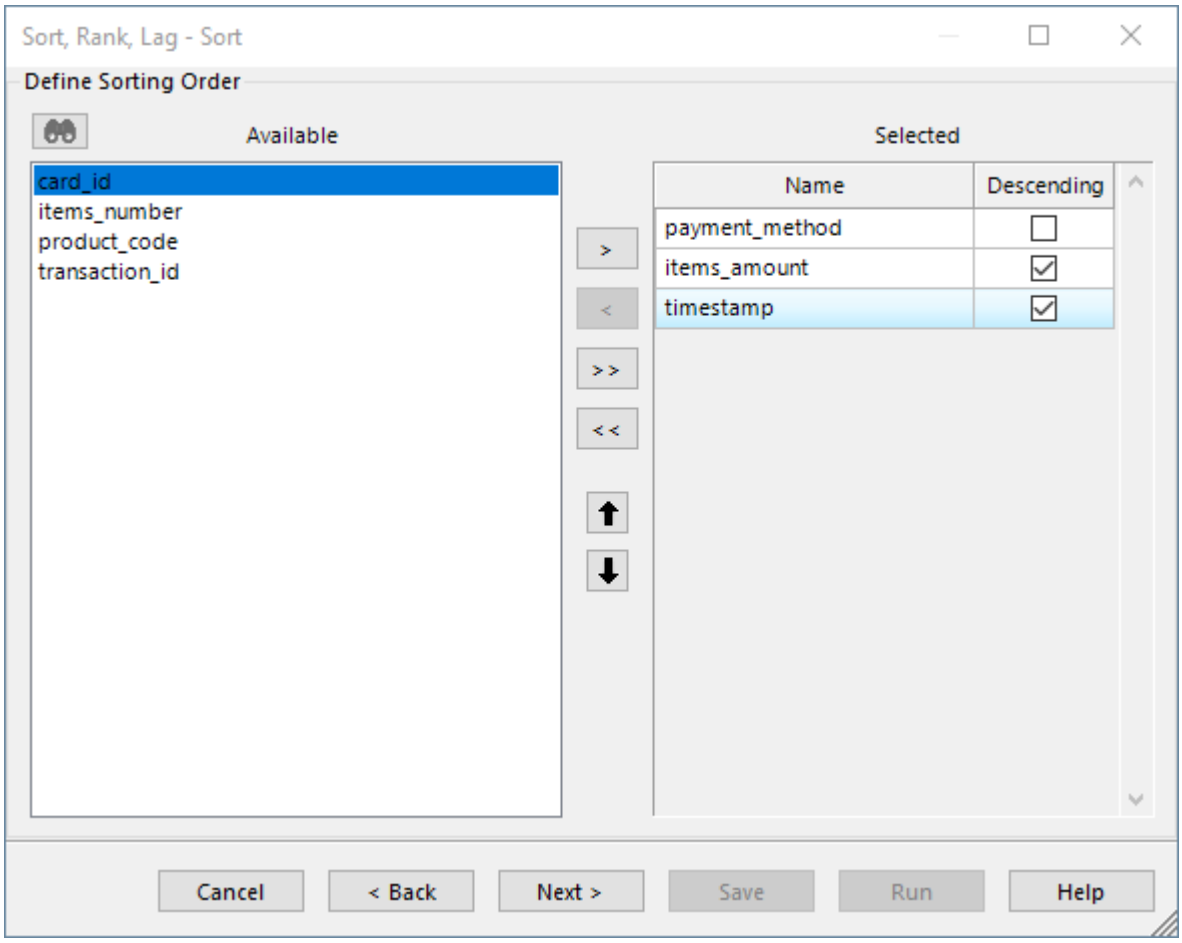
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The second page of the wizard provides the sorting options.

- Select the variables to be sorted. At least one variable must be selected.
- By default, variables are sorted in the ascending order. Choose the descending order by selecting the checkbox on the right, if required.
- The data will be sorted in the order of variables in the **Selected** list. Use the up and down arrow buttons to change the order if necessary.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**. The next page provides grouping options.

Select the variable(s) to group by or click **Next** to skip this step.

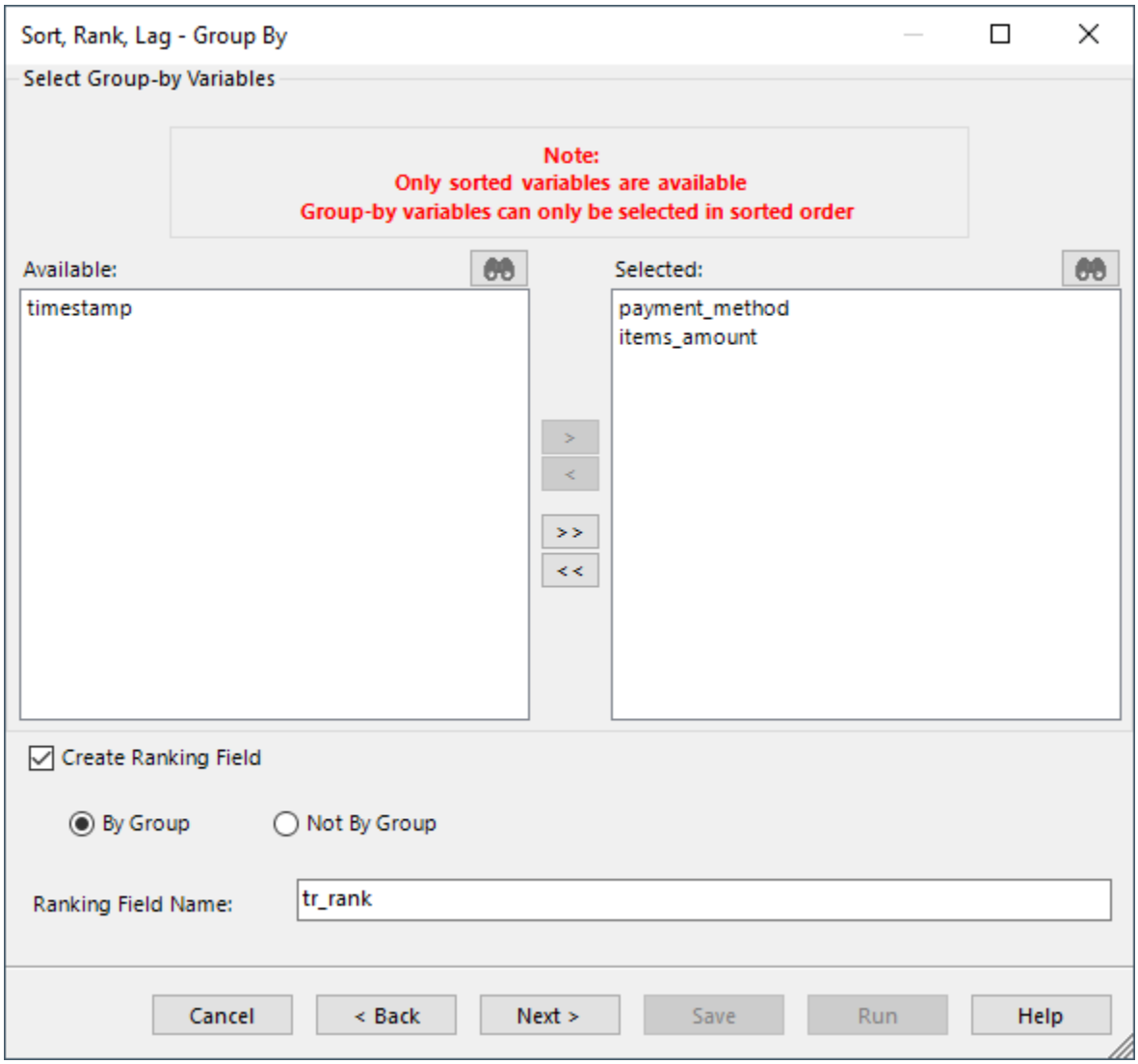
Group-by variables can only be selected in the sorted order. In the example above, only the following alternatives are possible:

- *payment_method*
- *payment_method, items_amount*
- *payment_method, items_amount, timestamp*

If you select a variable from the middle of the sorted list, all variables above it will be automatically selected too. For example, *items_amount*, then both *payment_method* and *items_amount* will be selected. Conversely, when deselecting variables, if you try to deselect a variable from the middle of the list, all variables below it will be automatically deselected too. In the example below, if you try to deselect *payment_method*, then both *payment_method* and *items_amount* will be deselected.

To create a ranking field based on the sorted data, select the checkbox "***Create Ranking Field***". This option is not selected by default. Specify the name for the ranking field and choose whether to create ranking by group or not.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



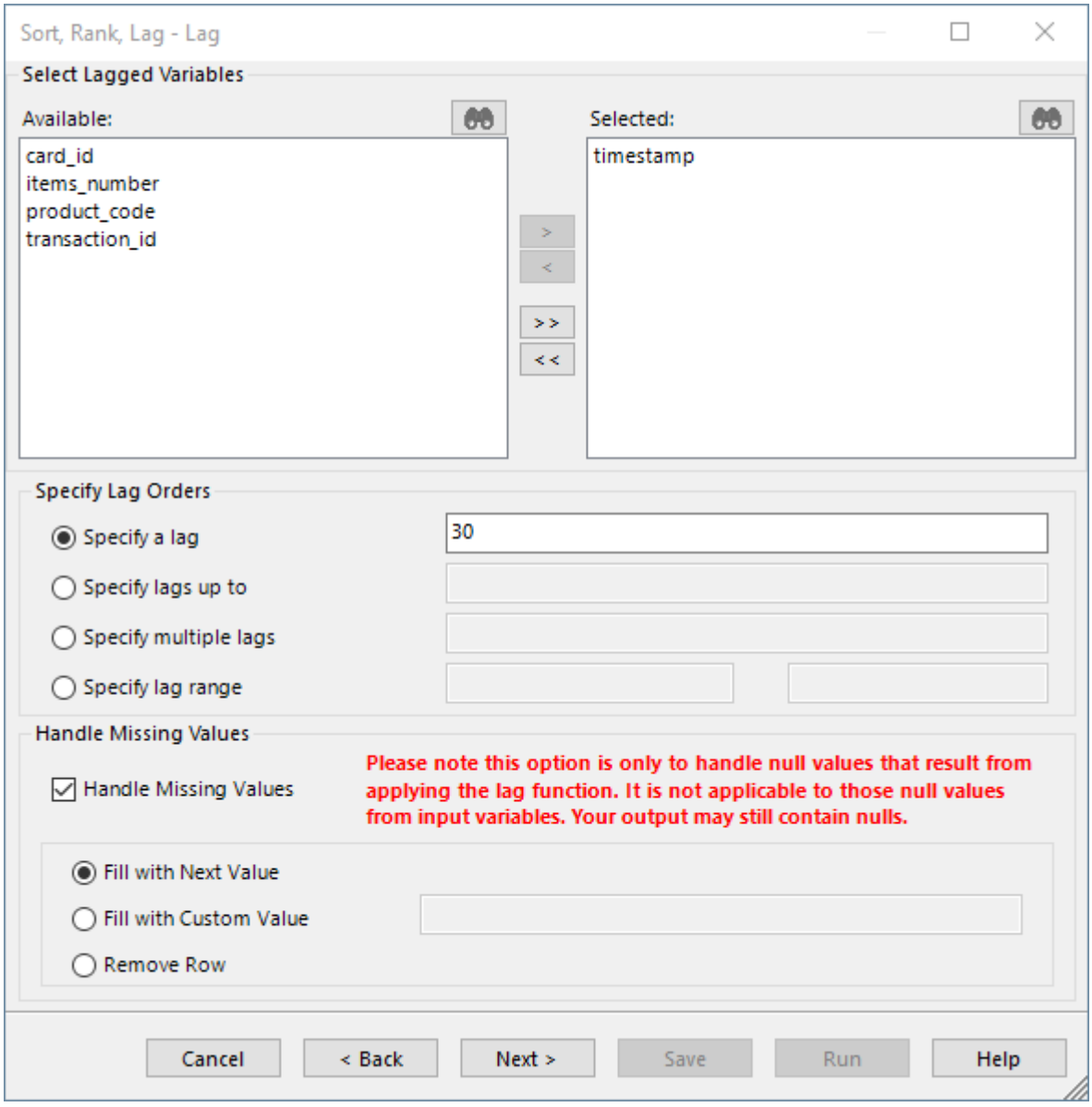
Click **Next** to proceed to the Lag variables page.

- Select the variable(s) to which a lag function is to be applied to create lagged variable(s). All variables except the group-by variables are available for selection. If lag variables are not required, click **Next** to skip this step.
- Specify lags by selecting the desired option and entering the appropriate values.
  - "***Specify a lag***" is the default selection. A single lagged variable will be created. Enter a positive integer number for the lag value.
  - Selecting the option "***Specify lags up to***" and specifying a positive integer value N will create N lagged variables with lags *1*, *2*, *3*, ... *N*.
  - If you select the option "***Specify multiple lags***", enter a comma-separated sequence of positive integers representing the lag values. For example, entering ***1, 3, 12*** will create three lagged variables with lags 1, 3, and 12, respectively. Spaces between the consecutive values are not mandatory.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- If you select the option "***Specify lag range***", specify the range boundaries for the lags. For example, entering ***6*** and ***12*** will create six lagged variables with lags *6*, *7*, *8*, *9, 10, 11, 12*.

- The newly generated lagged variables will contain missing values. If you want them to be handled in a specific way, select the checkbox "***Handle Missing Values***" and choose the preferred method:
  - "***Fill with Next Value***" (selected by default)
  - "***Fill with Custom Value***": In this case, enter the desired value to substitute for the missing values.
  - "***Remove Row***": Choose this option to drop the records with missing values in the lagged variables.

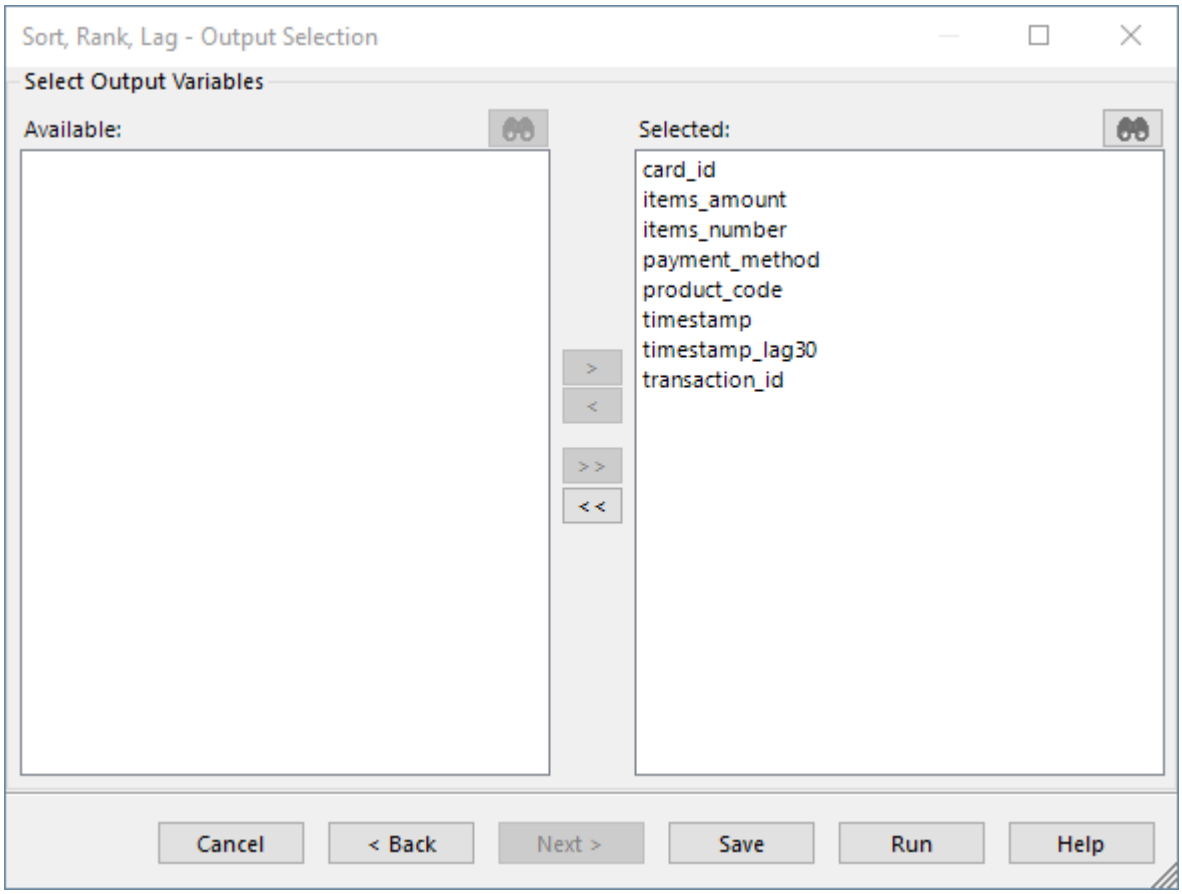


Click **Next**.

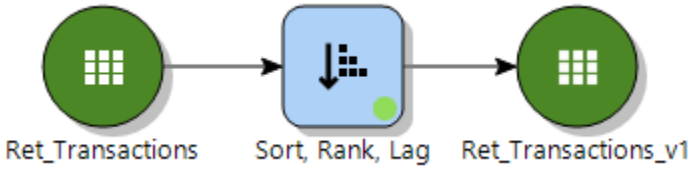
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

In the last page, select the variables to be included in the output dataset.

By default, all variables are selected for the output, including all sorted variables, group-by variables, lagged variables (if any), and the remaining variables.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. Double-click the output dataset node to view the results.



See also:

- About Data Manipulation
- Aggregating Data
- Filter

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Auto Feature Engineering

Overview

The **Auto Feature Engineering** node in the **Manipulate** palette automatically creates new features (variables) using transformations and aggregations of different types from two or more input datasets and makes them into a new dataset.

This node is integrated with Python and uses **`Featuretools -`** a popular Python library for feature engineering which creates features from temporal and relational datasets to prepare data for modeling. For details, see the **`featuretools`** documentation at https://www.featuretools.com. The node wizard provides a convenient graphical interface to specify the options and parameters for `featuretools`. The wizard automatically translates the inputs into Python code, which can be previewed before running the node. The input datasets are converted from the Altair KDD format to pandas dataframes in the process. The resulting dataframe is converted back to produce the output Altair dataset.

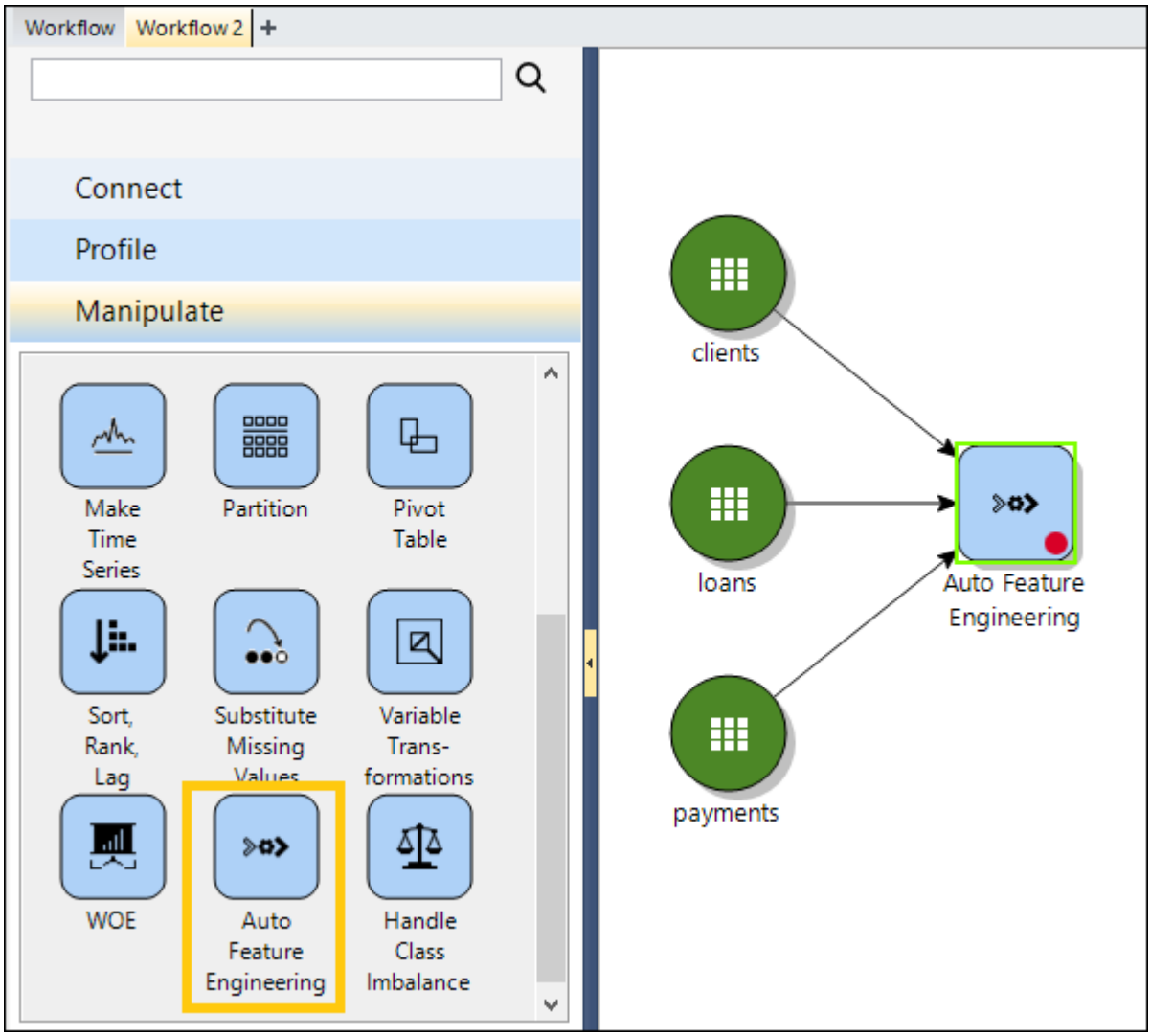
Data Manipulation with Auto Feature Engineering

Prerequisites: The Auto Feature Engineering requires at least two datasets as inputs. All input datasets must have ID fields and time index fields and represent tables linked in parent-child relationships.

To add an Auto Feature Engineering node to your workflow:

Drag the **Auto Feature Engineering** node from the **Manipulate** palette onto the workflow canvas. Alternatively, right-click anywhere on the canvas background and select **Manipulate --> Auto Feature Engineering** from the context menu. Connect the input datasets to the Auto Feature Engineering node. An example with three input datasets is shown below.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

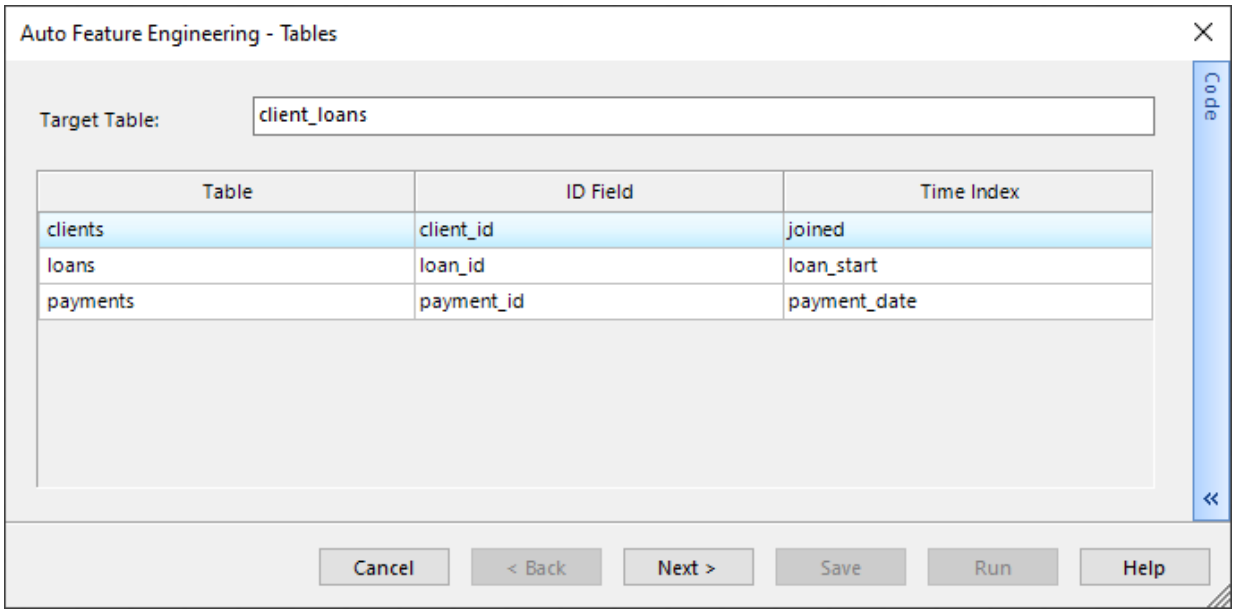


Double-click the Auto Feature Engineering node or right-click on it and select **Modify** to open the wizard.

In the first page of the wizard:

- Specify the name for the output dataset in the *Target Table* field.
- Select the ID field for each input dataset in the *ID Field* column.
- Select the *Time Index* field for each input dataset. This must be a Timestamp, Number or Integer field with no missing values.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

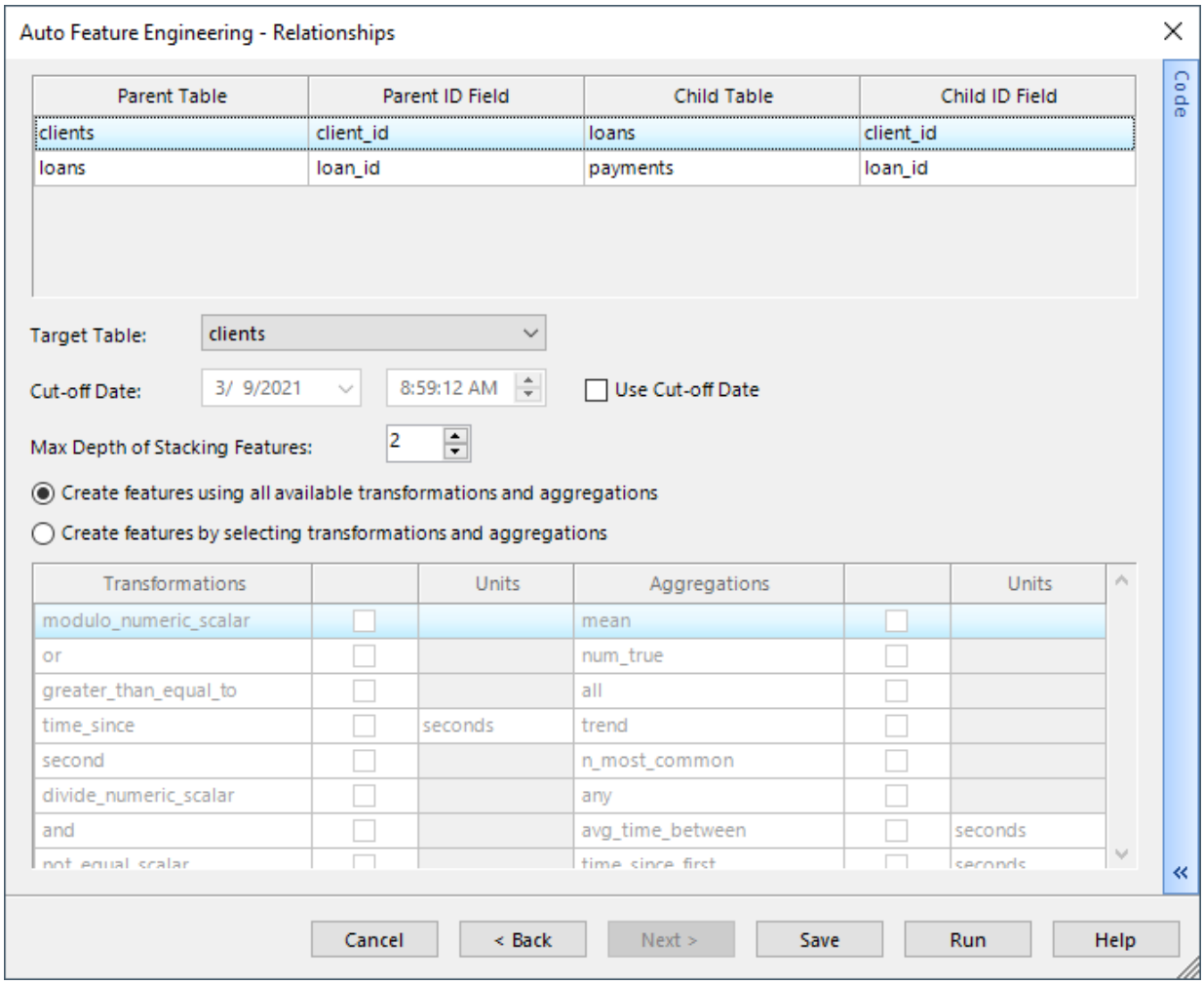


Click **Next**.

In the Relationships page:

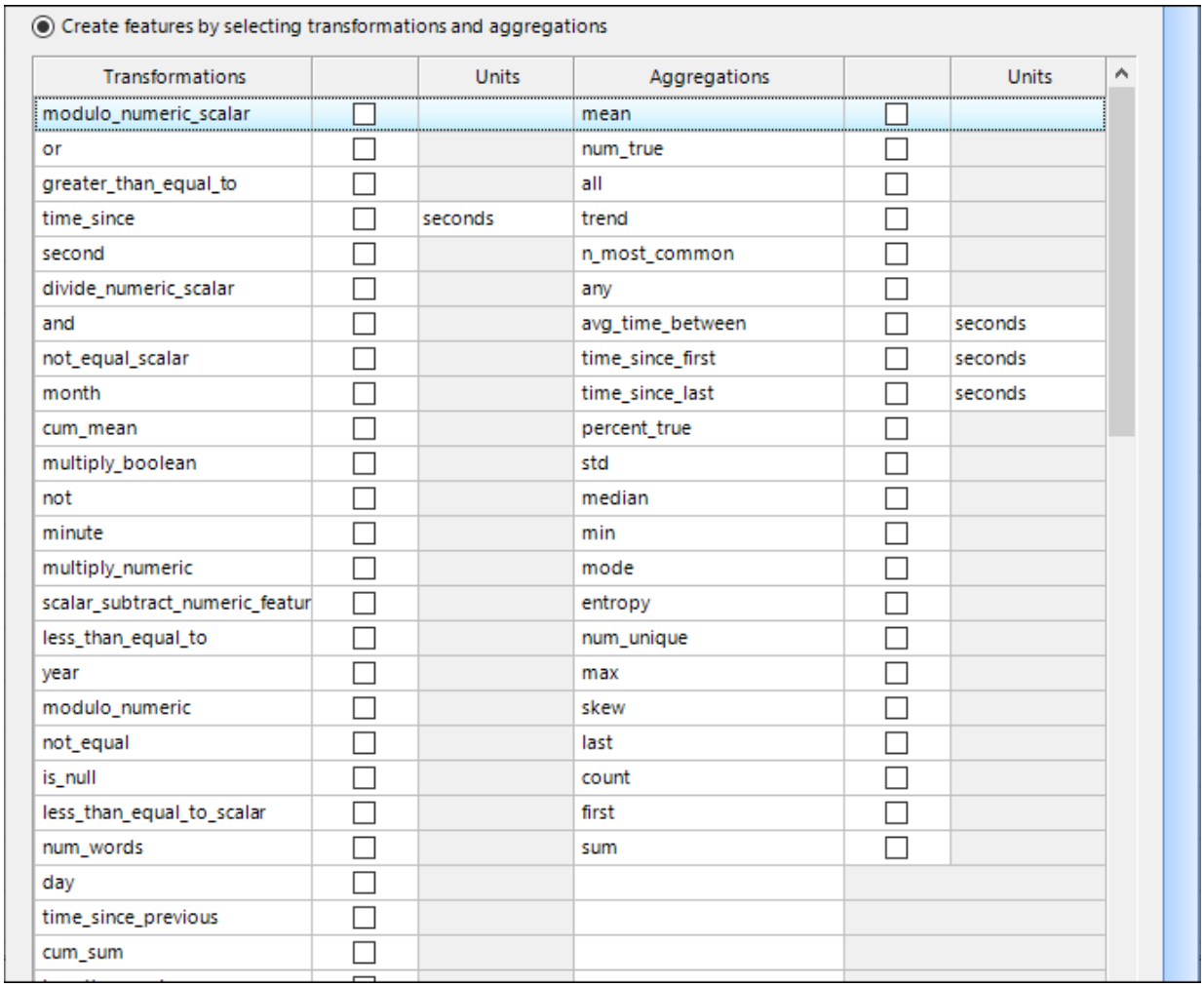
- Define the parent-child relationships for the tables representing the input datasets: For each Parent Table, select its Child Table. For the *Child ID Field,* specify the ID in the child table that links it to the parent table. In the example below, the clients table is a parent of the loans table, which is a parent of the payments table. The client_id in the loans table links it to the parent clients table, while the loan_id in the payments table links it to the parent loans table. In the general case, the link field names may be different in the parent and child tables.
- For *Target Table*, select the dataset that the output will be based on (usually the one that represents the base table in the hierarchy).
- Specify the cut-off date if necessary. Select the "*Use Cut-off Date*" box first, then enter the date and time.
- Specify the maximum depth of stacking features. The default is 2. Up to 3 levels are allowed.
- Available transformations and aggregations are listed in the table at the bottom of the wizard page. Choose whether to automatically create all of them or select the desired transformations and aggregations manually. The full list with descriptions is provided in the Appendix at the end of this Help topic.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



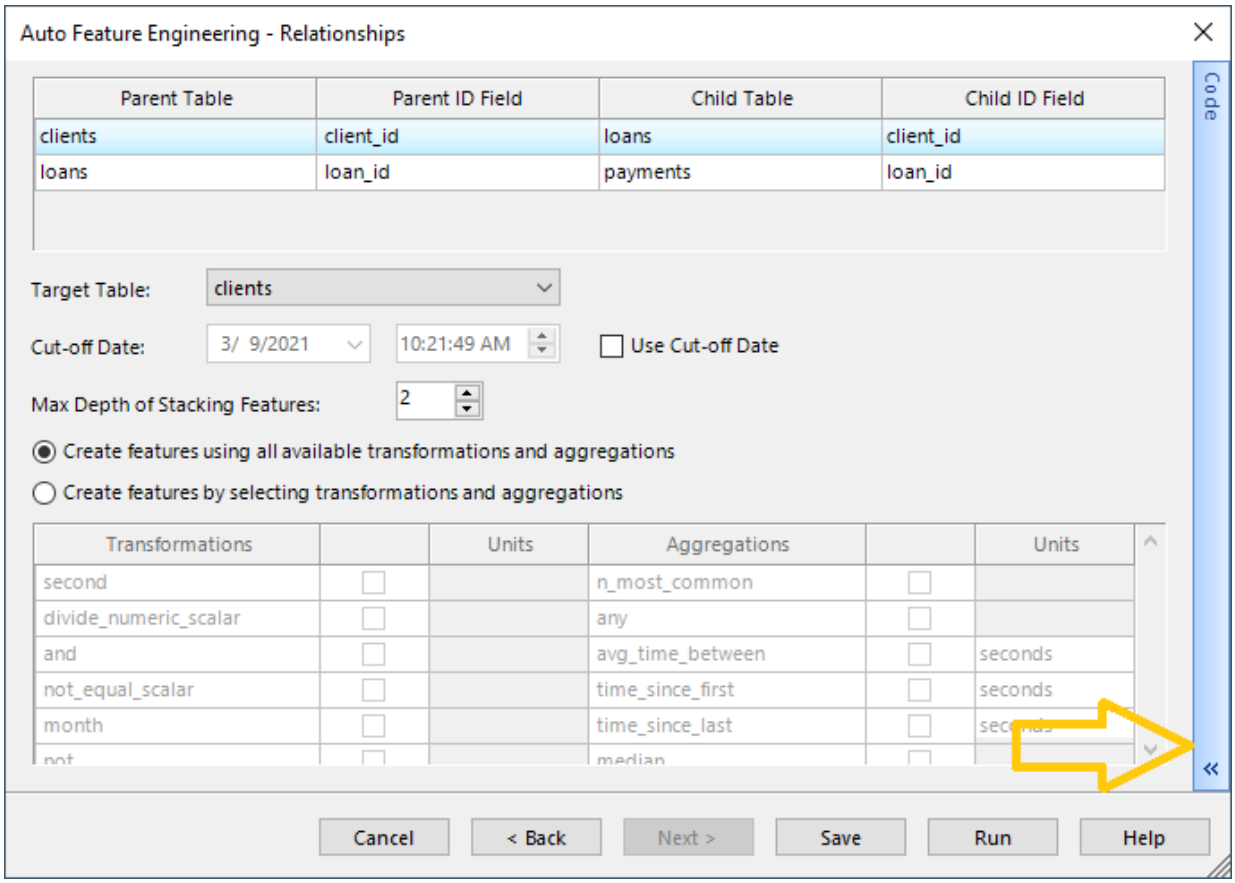
When selecting the desired transformations and aggregations manually, define the units for time and date function values.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



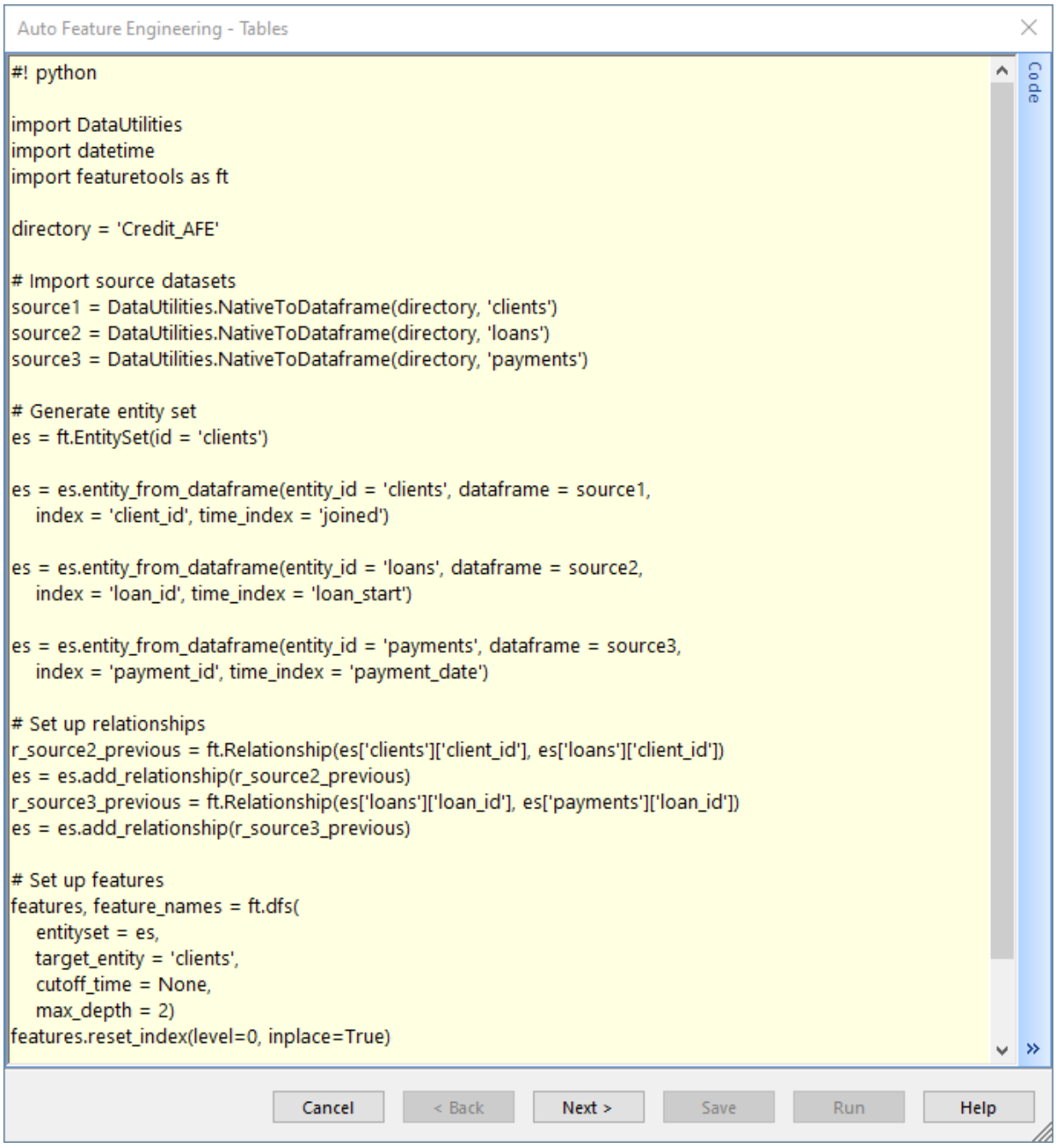
To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



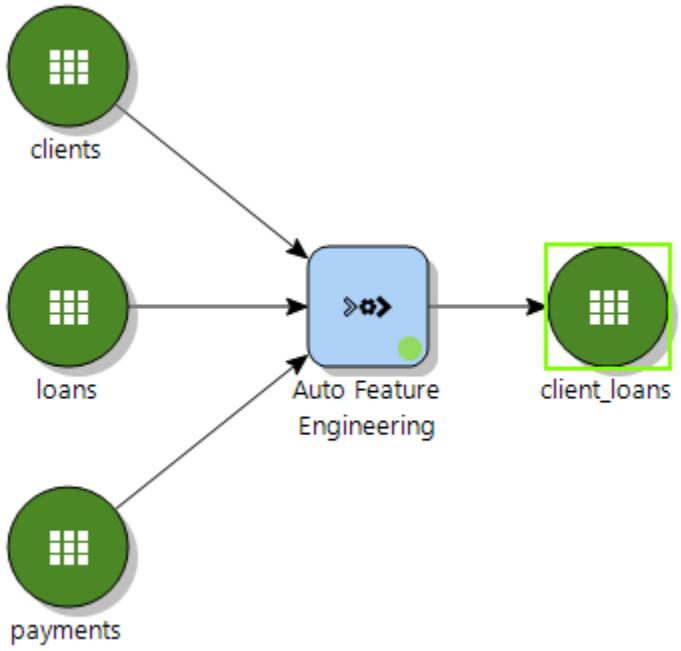
This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

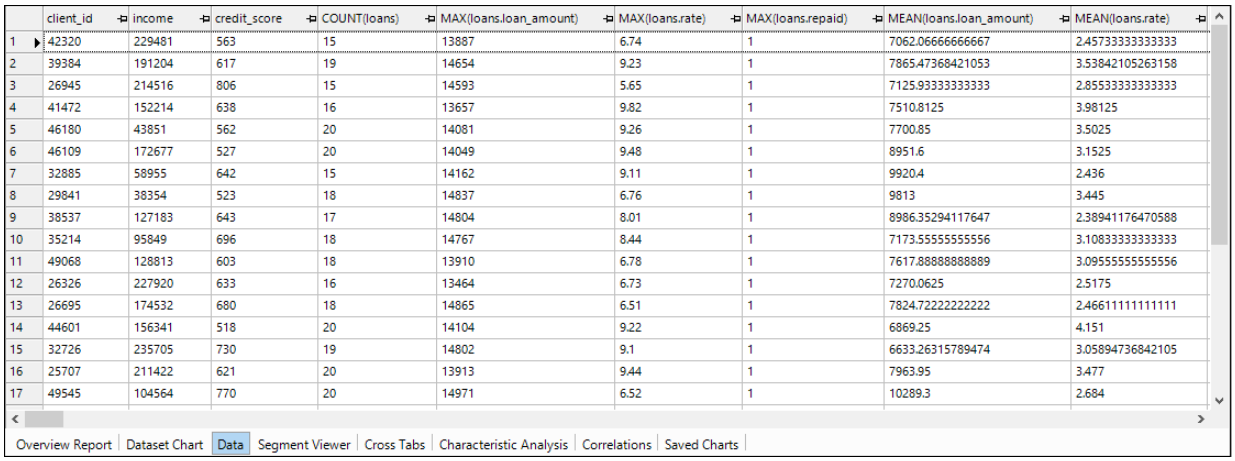


Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. The code generated from your inputs runs in Python. The input datasets are converted to Pandas dataframes, and the functions of the `featuretools` library are applied. The output data is converted from Pandas to the Altair KDD format and is shown as the output dataset node at the end.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Double-click the output dataset node to view the results and explore the generated features.



	client_id	income	credit_score	COUNT(loans)	MAX(loans.loan_amount)	MAX(loans.rate)	MAX(loans.repaid)	MEAN(loans.loan_amount)	MEAN(loans.rate)
1	42320	229481	563	15	13887	6.74	1	7062.06666666667	2.45733333333333
2	39384	191204	617	19	14654	9.23	1	7865.47368421053	3.53842105263158
3	26945	214516	806	15	14593	5.65	1	7125.93333333333	2.85533333333333
4	41472	152214	638	16	13657	9.82	1	7510.8125	3.98125
5	46180	43851	562	20	14081	9.26	1	7700.85	3.5025
6	46109	172677	527	20	14049	9.48	1	8951.6	3.1525
7	32885	58955	642	15	14162	9.11	1	9920.4	2.436
8	29841	38354	523	18	14837	6.76	1	9813	3.445
9	38537	127183	643	17	14804	8.01	1	8986.35294117647	2.38941176470588
10	35214	95849	696	18	14767	8.44	1	7173.55555555556	3.10833333333333
11	49068	128813	603	18	13910	6.78	1	7617.88888888889	3.09555555555556
12	26326	227920	633	16	13464	6.73	1	7270.0625	2.5175
13	26695	174532	680	18	14865	6.51	1	7824.72222222222	2.46611111111111
14	44601	156341	518	20	14104	9.22	1	6869.25	4.151
15	32726	235705	730	19	14802	9.1	1	6633.26315789474	3.05894736842105
16	25707	211422	621	20	13913	9.44	1	7963.95	3.477
17	49545	104564	770	20	14971	6.52	1	10289.3	2.684

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

Supported Transformations and Aggregations

#	Name	Type	Description
0	mean	aggregation	Computes the average for a list of values.
1	num_true	aggregation	Counts the number of `True` values.
2	all	aggregation	Calculates if all values are 'True' in a list.
3	trend	aggregation	Calculates the trend of a variable over time.
4	n_most_common	aggregation	Determines the `n` most common elements.
5	any	aggregation	Determines if any value is 'True' in a list.
6	time_since_first	aggregation	Calculates the time elapsed since the first datetime (in seconds).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

#	Name	Type	Description
7	percent_true	aggregation	Determines the percent of `True` values.
8	std	aggregation	Computes the dispersion relative to the mean value, ignoring `NaN`.
9	time_since_last	aggregation	Calculates the time elapsed since the last datetime (default in seconds).
10	median	aggregation	Determines the middlemost number in a list of values.
11	min	aggregation	Calculates the smallest value, ignoring `NaN` values.
12	mode	aggregation	Determines the most commonly repeated value.
13	entropy	aggregation	Calculates the entropy for a categorical variable.
14	num_unique	aggregation	Determines the number of distinct values, ignoring `NaN` values.
15	max	aggregation	Calculates the highest value, ignoring `NaN` values.
16	avg_time_between	aggregation	Computes the average number of seconds between consecutive events.
17	skew	aggregation	Computes the extent to which a distribution differs from a normal distribution.
18	last	aggregation	Determines the last value in a list.
19	count	aggregation	Determines the total number of values, excluding `NaN`.
20	first	aggregation	Determines the first value in a list.
21	sum	aggregation	Calculates the total addition, ignoring `NaN`.
22	modulo_numeric_scalar	transform	Return the modulo of each element in the list by a scalar.
23	or	transform	Element-wise logical OR of two lists.
24	greater_than_equal_to	transform	Determines if values in one list are greater than or equal to another list.
25	time_since	transform	Calculates time from a value to a specified cutoff datetime.
26	second	transform	Determines the seconds value of a datetime.
27	divide_numeric_scalar	transform	Divide each element in the list by a scalar.
28	and	transform	Element-wise logical AND of two lists.
29	not_equal_scalar	transform	Determines if values in a list are not equal to a given scalar.
30	month	transform	Determines the month value of a datetime.
31	cum_mean	transform	Calculates the cumulative mean.
32	multiply_Boolean	transform	Element-wise multiplication of two lists of Boolean values.
33	not	transform	Negates a Boolean value.
34	minute	transform	Determines the minutes value of a datetime.
35	multiply_numeric	transform	Element-wise multiplication of two lists.
36	scalar_subtract_numeric_feature	transform	Subtract each value in the list from a given scalar.
37	less_than_equal_to	transform	Determines if values in one list are less than or equal to another list.
38	year	transform	Determines the year value of a datetime.
39	modulo_numeric	transform	Element-wise modulo of two lists.
40	not_equal	transform	Determines if values in one list are not equal to another list.
41	is_null	transform	Determines if a value is null.
42	less_than_equal_to_scalar	transform	Determines if values are less than or equal to a given scalar.
43	num_words	transform	Determines the number of words in a string by counting the spaces.
44	day	transform	Determines the day of the month from a datetime.
45	time_since_previous	transform	Compute the time since the previous entry in a list.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

#	Name	Type	Description
46	cum_sum	transform	Calculates the cumulative sum.
47	less_than_scalar	transform	Determines if values are less than a given scalar.
48	diff	transform	Compute the difference between the value in a list and the previous value in that list.
49	is_weekend	transform	Determines if a date falls on a weekend.
50	subtract_numeric_scalar	transform	Subtract a scalar from each element in the list.
51	subtract_numeric	transform	Element-wise subtraction of two lists.
52	greater_than_scalar	transform	Determines if values are greater than a given scalar.
53	longitude	transform	Returns the second tuple value in a list of LatLong tuples.
54	negate	transform	Negates a numeric value.
55	week	transform	Determines the week of the year from a datetime.
56	cum_max	transform	Calculates the cumulative maximum.
57	divide_numeric	transform	Element-wise division of two lists.
58	greater_than_equal_to_scalar	transform	Determines if values are greater than or equal to a given scalar.
59	hour	transform	Determines the hour value of a datetime.
60	cum_count	transform	Calculates the cumulative count.
61	multiply_numeric_scalar	transform	Multiply each element in the list by a scalar.
62	weekday	transform	Determines the day of the week from a datetime.
63	modulo_by_feature	transform	Return the modulo of a scalar by each element in the list.
64	add_numeric	transform	Element-wise addition of two lists.
65	greater_than	transform	Determines if values in one list are greater than another list.
66	haversine	transform	Calculates the approximate haversine distance between two LatLong variable types.
67	less_than	transform	Determines if values in one list are less than another list.
68	add_numeric_scalar	transform	Add a scalar to each value in the list.
69	absolute	transform	Computes the absolute value of a number.
70	equal_scalar	transform	Determines if values in a list are equal to a given scalar.
71	isin	transform	Determines whether a value is present in a provided list.
72	percentile	transform	Determines the percentile rank for each value in a list.
73	divide_by_feature	transform	Divide a scalar by each value in the list.
74	cum_min	transform	Calculates the cumulative minimum.
75	equal	transform	Determines if values in one list are equal to another list.
76	age	transform	Calculates the age in years as a floating-point number given a date of birth.
77	latitude	transform	Returns the first tuple value in a list of LatLong tuples.
78	num_characters	transform	Calculates the number of characters in a string.

Variable Transformations

Variable Transformations Node

The **Variable Transformations** node provides the following data manipulation capabilities:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- add new calculated fields (one or multiple at a time)
- delete fields
- edit expressions for calculated fields
- rename fields
- change the data types of fields
- specify the display format (such as Currency, Percent, etc.; specify numeric precision)
- define special values
- reorder fields
- export expressions to a file
- import expressions from other datasets or expression definition files

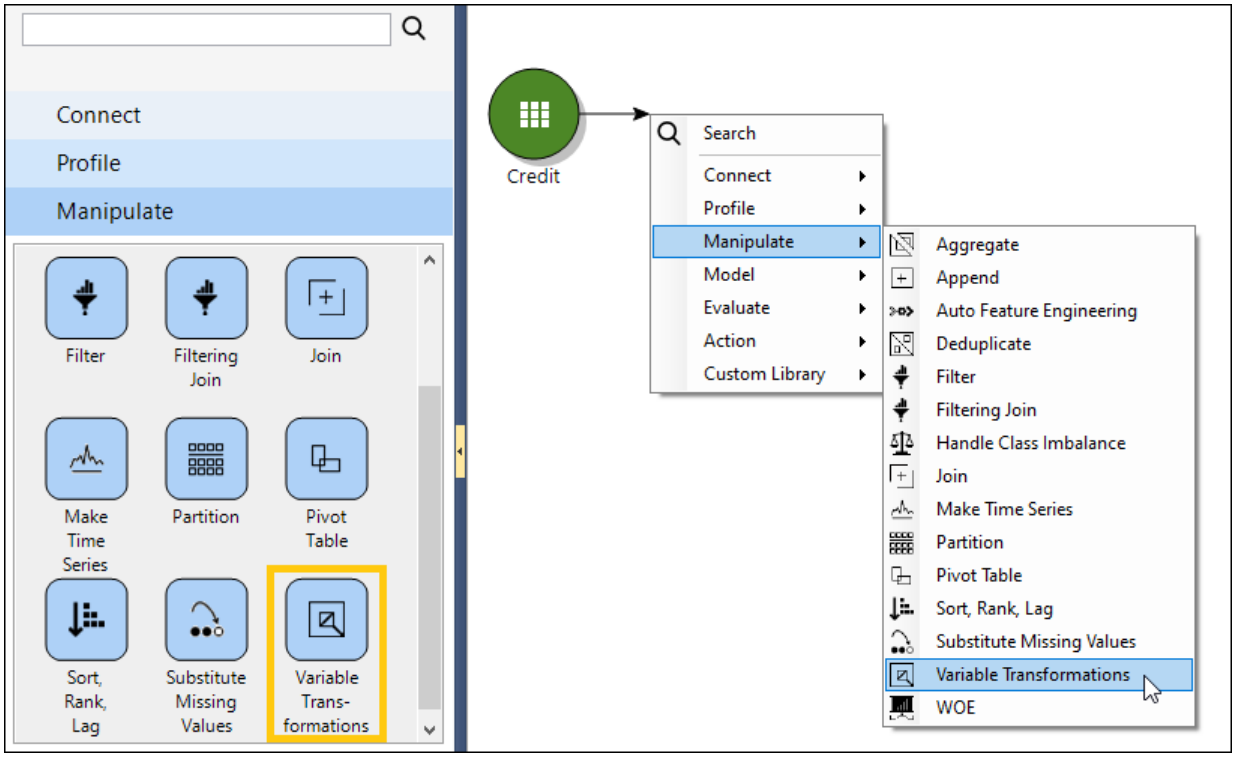
The transformed data can be previewed before running the transformations.

To perform variable transformations in your workflow:

- Drag a connection arrow from the dataset node to any area on the Workflow canvas background. This brings up a context menu that allows you to select the desired operation. From the context menu, select **Manipulate --> Variable Transformations** as shown in the example below. A new **Variable Transformations** node will be created and the wizard will open.

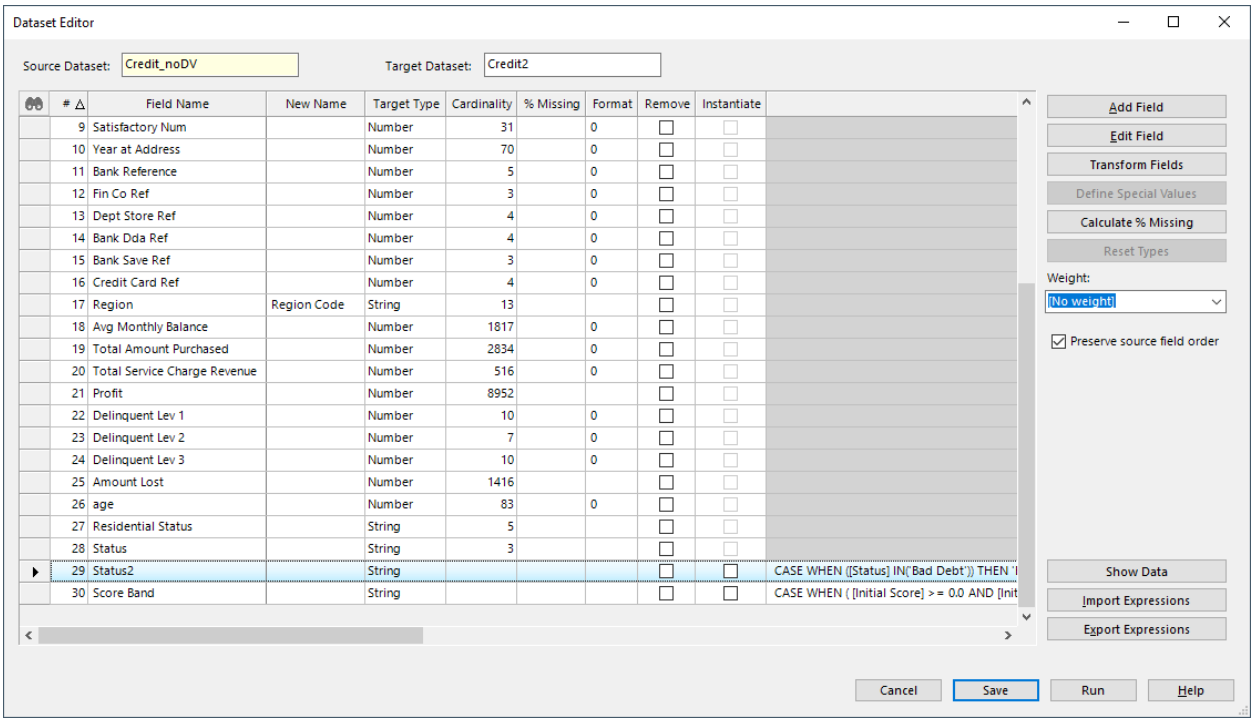
OR:

- Drag the **Variable Transformations** node from the **Manipulate** palette onto the canvas and connect the input dataset to the node



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Double-click on the Variable Transformations node or right-click on it and select **Modify** from the context menu. This will bring up the Dataset Editor window:



All columns in this grid are sortable. For example, if you want to remove all String fields, click on the *Target Type* column header to group fields by type. It will then be convenient to highlight all String fields at once (click and drag the mouse down) and mark them all at once for deletion by placing a checkmark in the Remove column for any of the highlighted fields.

Advanced field finding and multi-selection capabilities are also available via the Field Finder dialog that can be opened by clicking the binoculars button at the top-left corner of the grid. In the Field Finder you can use previously saved Variable Lists.

Field Name	The name of the field. This column is read-only. If you wish to change the name of a raw (non-calculated) field, you need to re-import your dataset and change the field names in the last page of the import wizard (the Field Transformer page). Alternatively, you can add a new field as a copy of an existing raw field and specify the desired name.
New Name	To rename a field, specify the new name in this column.
Target Type	The data type of the field. Data types include *String, Integer, Number, Boolean, Date, Time*, and *Timestamp*. The data type can be changed for any field by clicking in this column and selecting the desired type from the drop-down list.
Cardinality	The number of distinct values in the field.
% Missing	This column shows the percentage of missing values in each field if the user clicks the button **Calculate % Missing**.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

	(Not applicable to the new variables being added in the current Dataset Editor session.)
Format	The display format for the values in a given field. The format can include precision specifier for numeric fields. To specify or modify the format for a field, select the field and click the **Edit Field** button. For the list of available formats and the ways to specify custom precision, see the topic Specifying Format.
Remove	This column is used to select the variables to be removed from the input dataset. The output dataset will not contain the variables selected for removal.
Instantiate	This column is used to indicate whether a calculated variable must be ***instantiated***. This means that the variable's expression (shown in the Expressions column) will be removed in the output dataset, thus making it a raw variable rather than calculated. This option is equivalent to the *Paste Values* option of the Paste command in Microsoft Excel, which only pastes values without the functions that define them.
Expression	Only applies to calculated fields (fields added in the Dataset Editor using the Expression Editor). Displays the expression specified for the calculated field. The expression can be modified by clicking the Edit Field button and editing the expression in the Expression Editor. See the topic About the Expression Editor.

Renaming Fields

To rename a field, select it and enter the desired name in the **New Name** column.

Note: It is recommended to rename fields in a separate Variable Transformation node, since combining the renaming operations with creating new variables in the same transformation session may lead to slower performance.

Reordering Fields

If you want the order of fields in the output dataset to be the same as in the Dataset Editor, deselect the option "**Preserve source field order**".

Fields can be reordered using the following methods:

- Drag and drop:
  - Click on the field name to select it. When the row representing this field is highlighted, click on the field name again and hold the left mouse button down. Drag the field name up or down to the desired position.
- Sorting on any column in the grid, for example, *Target Type*, *Cardinality*, *% Missing*, and others.

If the "**Preserve source field order**" option is selected, the order of fields in the output dataset will be the same as in the input dataset (determined by the index column - the leftmost column in the Dataset Editor grid). This is the default setting.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Adding and Removing Fields

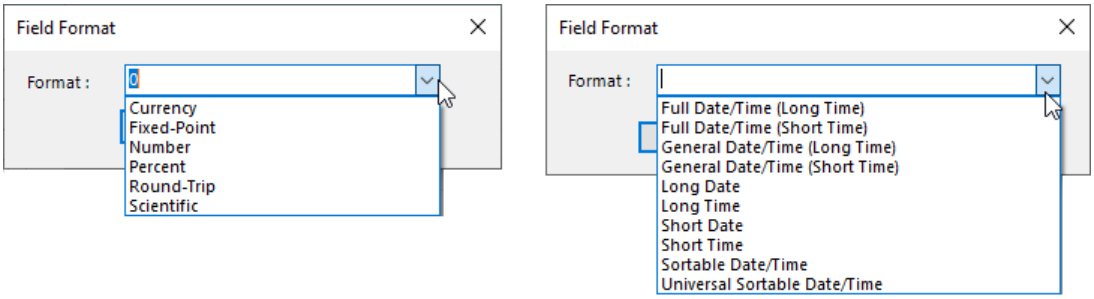
To add a new calculated field, click the Add Field button. This will open the Expression Editor window, which allows you to enter a SQL expression or use one of the Helper wizards for most common transformations. See the topic About the Expression Editor.

To delete a field, select it by clicking in the ***Remove*** column for the field you want to remove. Upon running the Variable Transformations node, the variables selected for removal will be not present in the output dataset.

Editing Fields

Allows you to edit the expressions for *calculated fields* using the Expression Editor.

For the original (raw) fields, which have no expressions, the **Edit Field** button opens the Format String dialog used to specify the format. The available formats depend on the data type of the selected field. The first example below shows the available options for a numeric field. The second one shows the options for Date, Time, and Timestamp fields. For detailed description of format strings, see Specifying Format.



Note: If you choose one of the Date/Time formats for a Time field, the default date part will be added to the Time part (the same year/month/day for all time values). For Time fields, *Long Time* and *Short Time* are the only appropriate formats to display time with or without seconds.

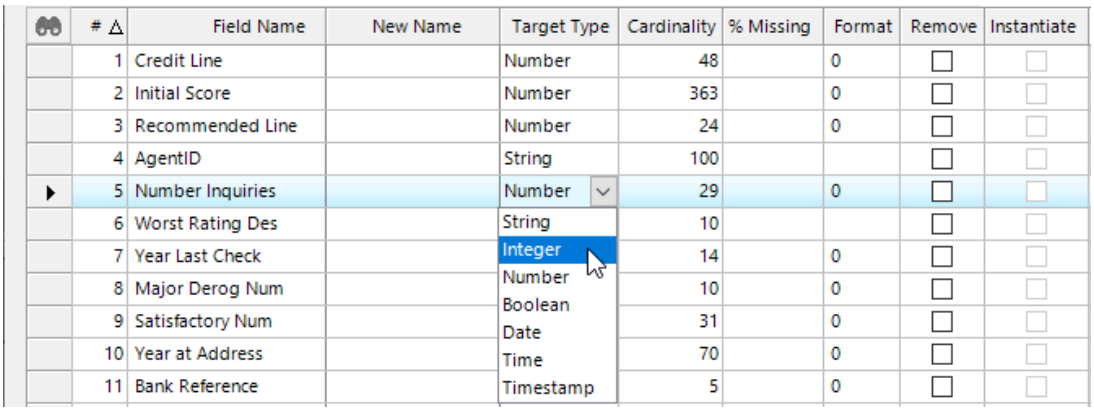
Changing Data Types

To change the data type of a field, click in the **Target Type** column for the desired field and select the desired type from the drop-down list.

> Note: Data types cannot be changed in the Target Type column for the *new* calculated variables being added in the current Dataset Editor session. For the newly added variables, the data type is determined by its expression entered in Expression Editor. However, you can use explicit type conversion in Expression Editor using the command **`CAST(<expression> AS <type>)`**, where <type> can be `STRING`, `DOUBLE`, `INTEGER`, `BOOLEAN`, `DATE`, `TIME`, or `TIMESTAMP`.

Example: Changing the data type of a field from *Number* (real) to *Integer*:

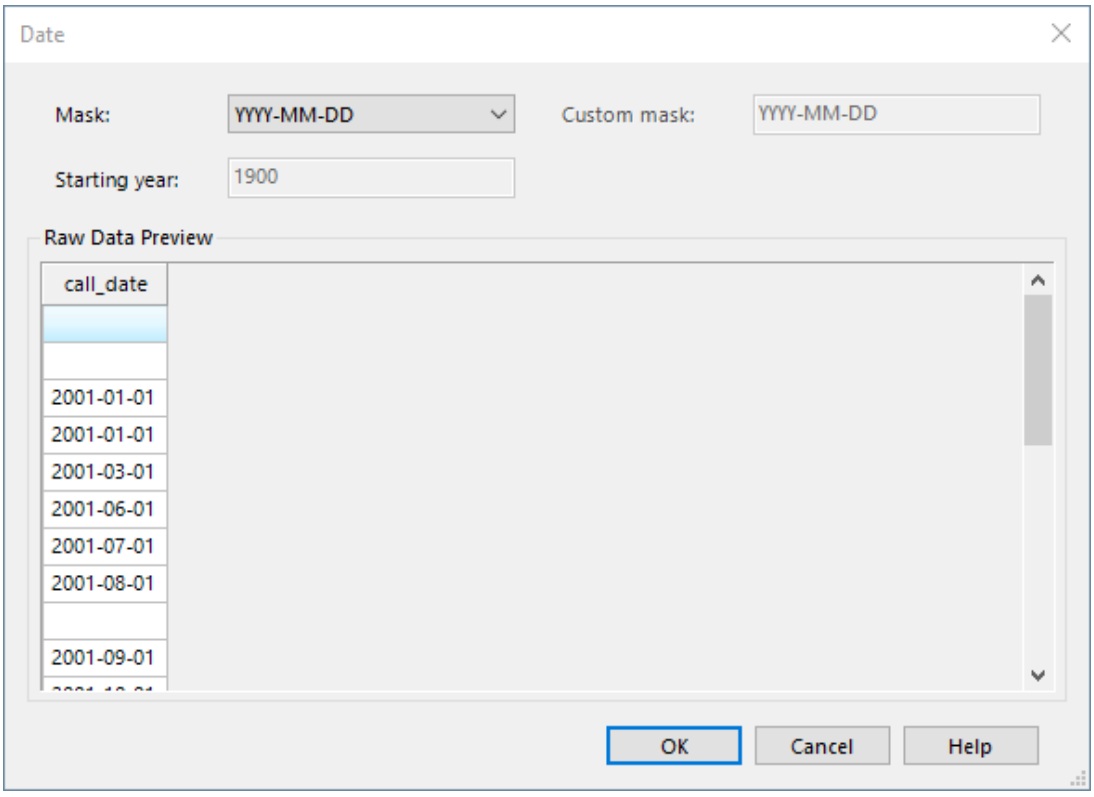
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



You can select multiple variables at a time to change their type at once. Click in any column (except Target Type) and drag the mouse to select multiple rows. Alternatively, use the combination of the mouse click and the SHIFT or CTRL key to select the desired rows. For example, you can sort variables by type, then select the top 100 String variables and convert them to Number by clicking in the Target Type column for the first of the selected rows and selecting the Number type.

To reset the data types of all fields to the original types assigned by default, click the **Reset Types** button.

If you change the type of a field to *Date*, *Time*, or *Timestamp*, you will be able to define the date/time mask for the field.

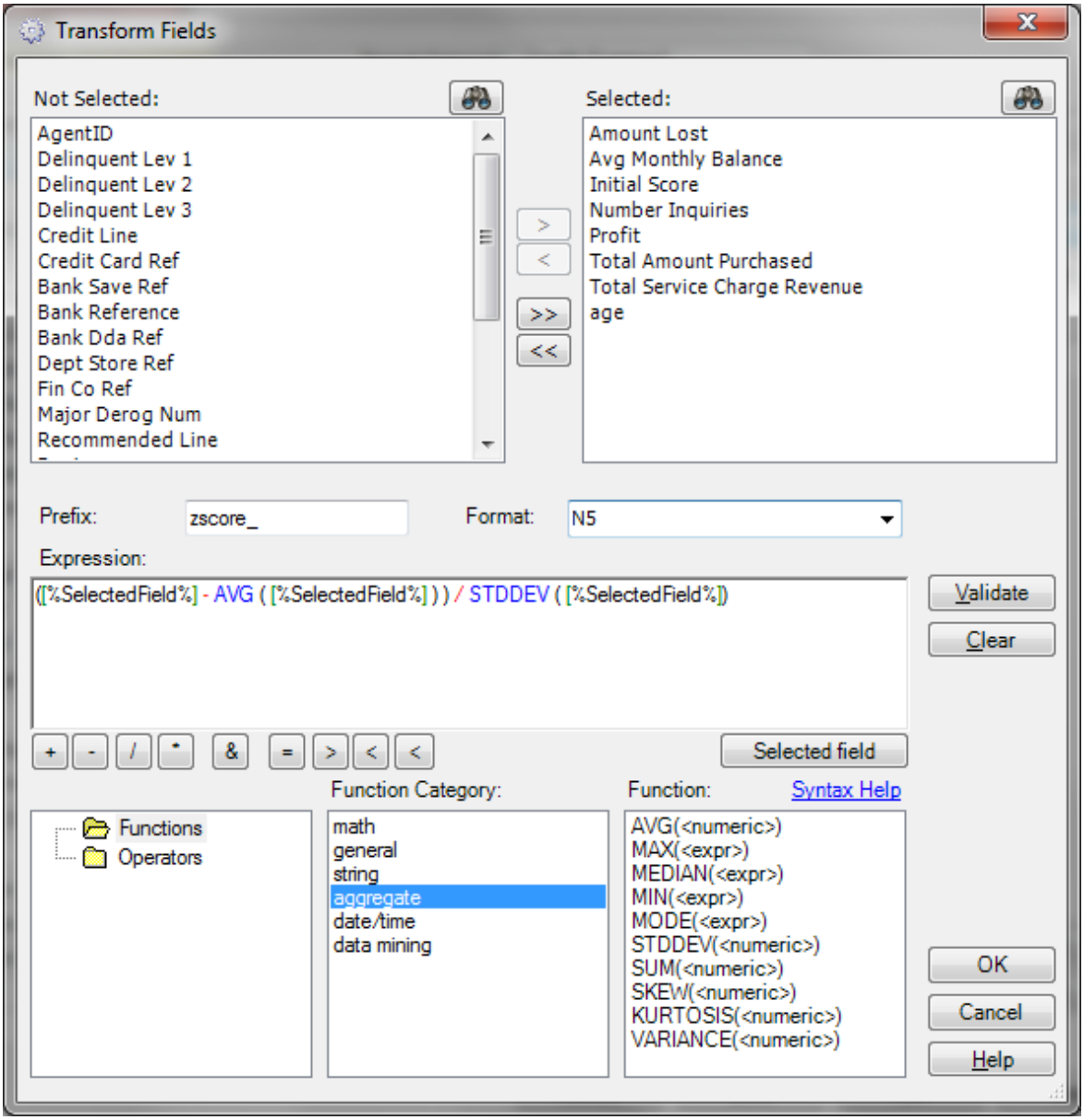


Transforming Multiple Fields

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The **Transform Fields** button opens an advanced Expression Editor that provides the capability to transform multiple variables at a time: Define an SQL expression and apply it to as many variables as desired. This may be useful in many cases, for example, for standard normalization of multiple numeric variables. The expression specified in the Expression field is applied to all variables in the Selected column at the top right. The variable **[%SelectedField%]** stands for all selected variables. You can enter it quickly at any time by clicking the ***Selected field*** button. See also the topic About the Expression Editor.

In the example below, for each selected variable **x**, the transformation `(x - Mean) / (Standard Deviation)` will be applied.



For detailed description of format strings, see Specifying Format.

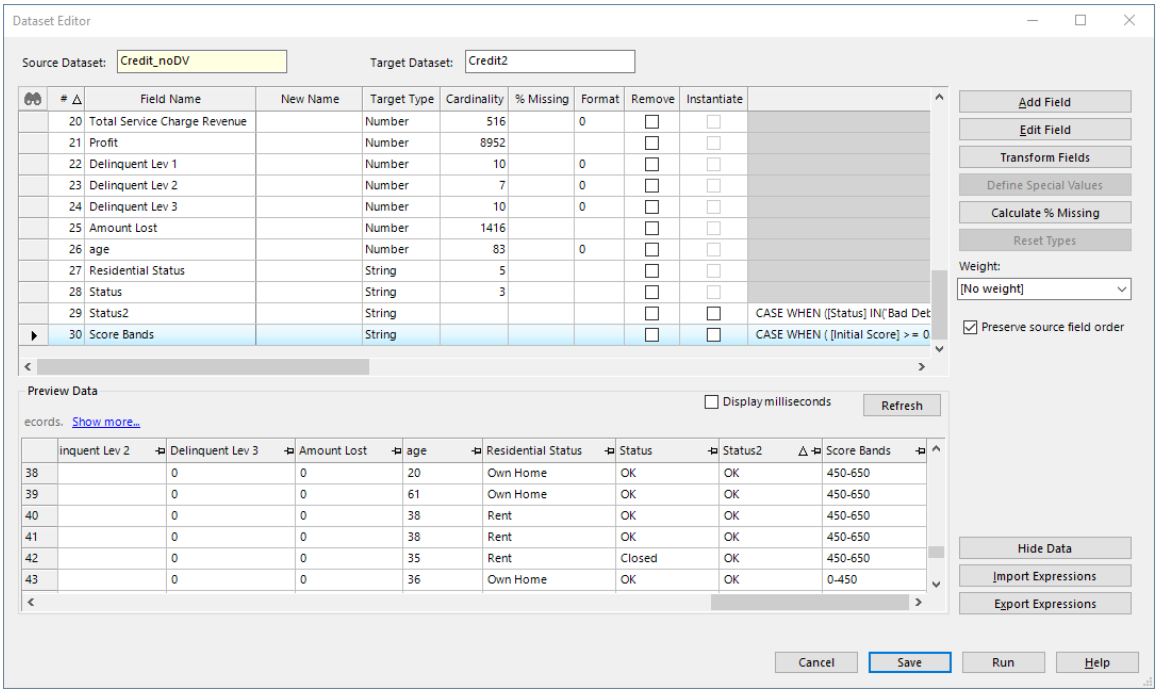
Define Special Values

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Click this button if you need to define and manage ***special values*** (e.g., if a value like "-999" in an otherwise positive numeric field has a special meaning). To define and, if necessary, substitute special values, in the general case, one new field will be created for value substitution and another to keep the information about the values. See the topic Defining Special Values for details.

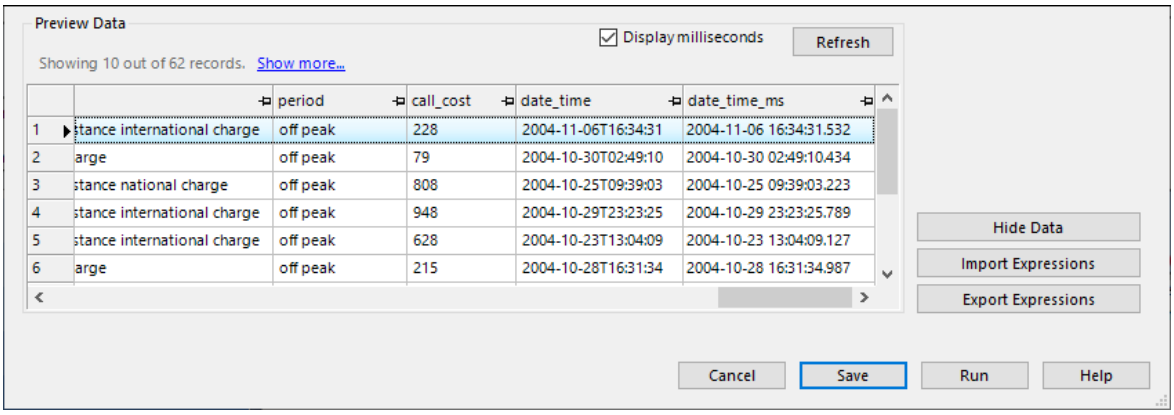
Show Data / Hide Data

Click the **Show Data** button to preview the transformed data before running the specified transformations. This opens the **Preview Data** frame at the bottom of the Dataset Editor window. Click **Refresh** to show or update the data displayed in the preview frame. By default, the top 10 records of the transformed dataset are displayed. Click "*Show more...*" or just scroll down in the Preview Data frame if you would like to view more records. Click **Hide Data** to hide the Preview Data panel. In the example below, the preview frame shows the records with values of the new calculated fields *Status2* and *Score Bands* whose expressions are defined in the upper frame.



If your source data contains Time or Timestamp values with milliseconds, select the option ***Display milliseconds*** and click **Refresh**. The milliseconds part will be displayed for all Time and Timestamp fields. An example is shown below.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Import Expressions

Expressions for calculated fields in other datasets can be imported to be applied to any field in the current dataset, provided that all the dependent fields occurring in the expression are present in the current dataset.

Expressions can also be imported from expression files (in XML format) created using the **Export Expressions** command.

Export Expressions

Expressions for calculated fields in the current dataset can be exported to an external XML file that can be used when adding or editing calculated fields in other datasets (using the **Import Expressions** command).

Click **Save** to save your definitions and changes. Click **Run** if you want to save and execute the specified transformations immediately.

See also: Expression Editor: Overview.

Expression Editor

Expression Editor: Overview

Expression Editor allows you to build expressions in order to create new fields in your dataset. You can manually enter a SQL expression or use one of the Helper wizards for most common transformations. These new fields are often referred to as *calculated columns*.

Note that Strategy Trees have their own Expression Editor for node calculations. It is described in the topic Expression Editor for Node Calculations of the Strategy Tree Help section.

Some of the possible objectives that can be achieved by creating a new field are:

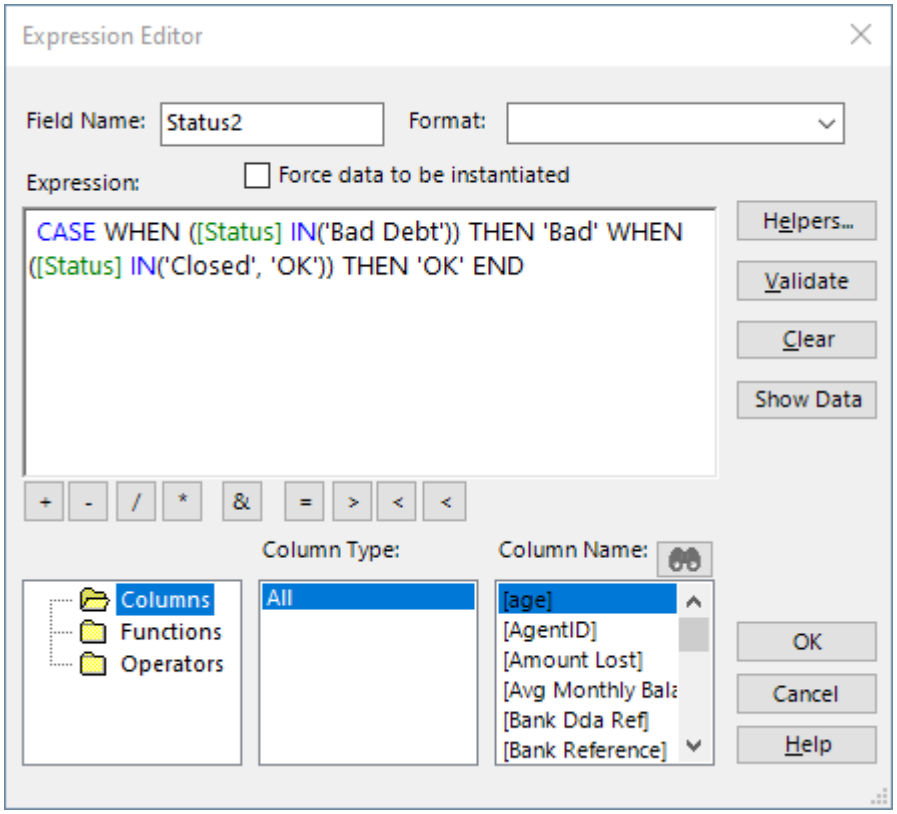
- mapping the categories of a field to new categories

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- merging several categories into one
- creating a discrete representation (binning) of a continuous variable
- performing calculations on existing numeric fields to express the results in a new field
- replacing missing values with a custom value (user-specified or calculated)

Expression Editor is opened from the Dataset Editor dialog in the Variable Transformations Node wizard when you click the **Add Field** or **Edit Field** button.

Expression Editor uses SQL expressions to create new columns in the dataset.



Field Name	The name for the new field (variable).
Expression	The expression that defines the new variable. You can manually type your an expression in the Transact SQL syntax and the Columns, Functions, and Operators reference in the bottom part of the dialog. Alternatively, use the **Helpers** button that launched a wizard for constructing most common expressions. Note that variable names should be enclosed in square brackets. For reference on the function syntax, click on the Functions folder and then click the Syntax Help link.
Format	The display format of the resulting value. If left blank, the application default is assumed. You can set formats such as Currency, Percent, Scientific, and many more. See the topic Specifying Format for details.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Force data to be instantiated	If this option is selected, the expression that defines the values of the new column will be removed in the new dataset (the new column will be a "raw" column with no expression). This option is similar to the Paste Values option of the Paste function in Microsoft Excel, which only pastes values (without the function that defines them).

Helpers

The purpose of the helpers is to help you create most commonly used expressions for your calculated columns using a simple wizard-like process. After you construct an expression using a helper and click Finish in the helper wizard, the expression appears in Expression Editor, so you can edit it further, if necessary, before the calculated column is added. See Expression Helpers for details.

Validate

The **Validate** button validates the syntax of the expression you entered. The syntax of expressions is based on the standard SQL syntax. There are some additional syntax rules for referring to objects and projects. See also the Function Syntax topic.

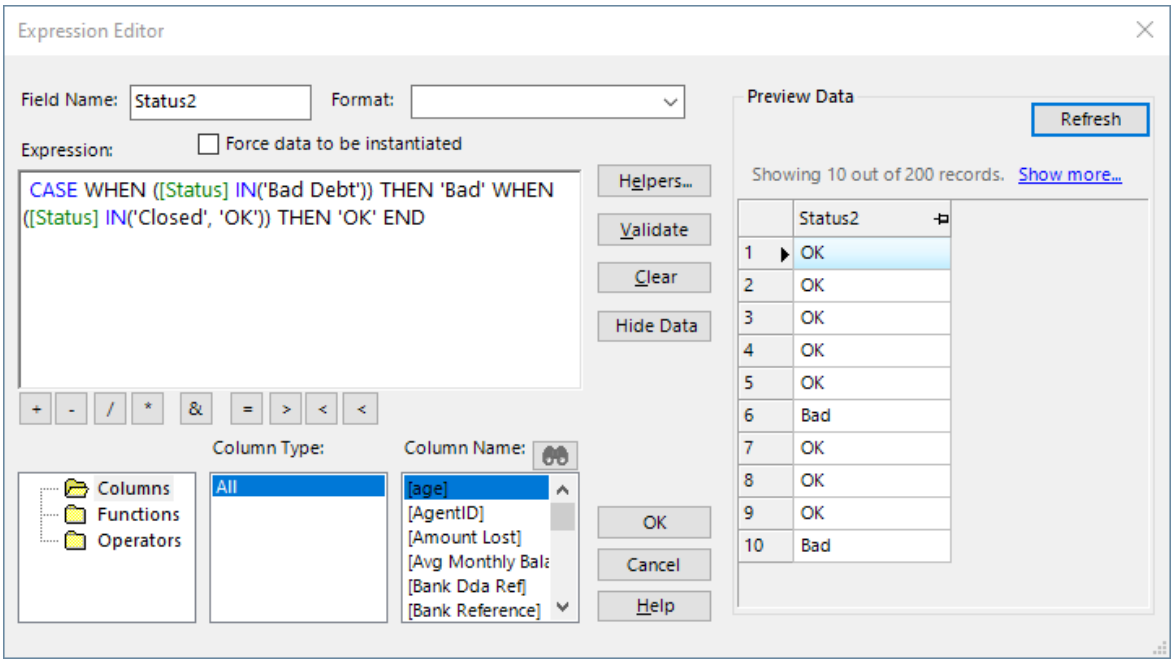
Clear

The **Clear** button clears the Expression frame. This action cannot be undone.

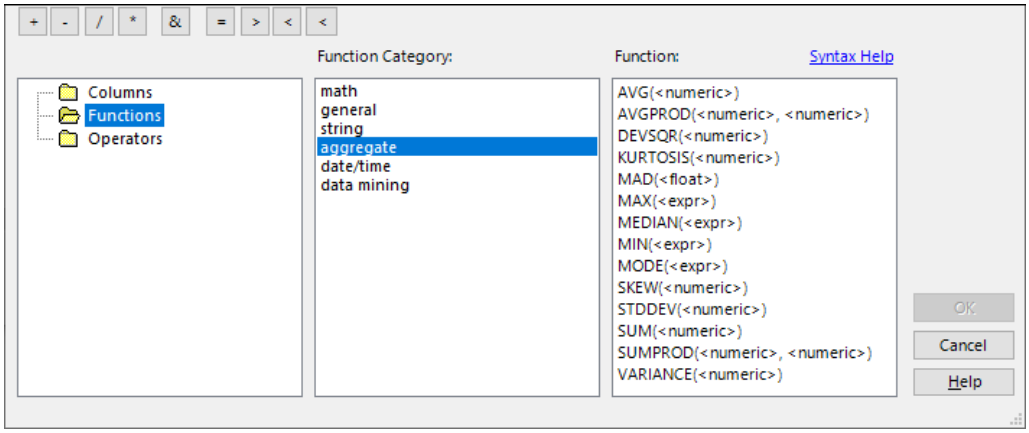
Show Data / Hide Data

Click **Show Data** if you would like to preview the values of the new variable defined by the specified expression. This opens the **Preview Data** panel on the right-hand side of the Expression Editor window. Click **Refresh** to show or update the data displayed in the panel. By default, the top 10 records of the new variable are displayed. Click "*Show more...*" if you would like to preview more records. Click **Hide Data** to hide the Preview Data panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Columns, Functions and Operators



The folders *Columns*, *Functions*, and *Operators* provide help in building your expressions. The Columns list displays the names of all the columns in the dataset. The Functions list contains standard functions for SQL expressions, various mathematical, string, and date/time functions. The Operators list contains arithmetic, logical, and bitwise operators.

See the topic Expression Editor: Functions and Operators for details.

See also: Function Syntax, Specifying Format.

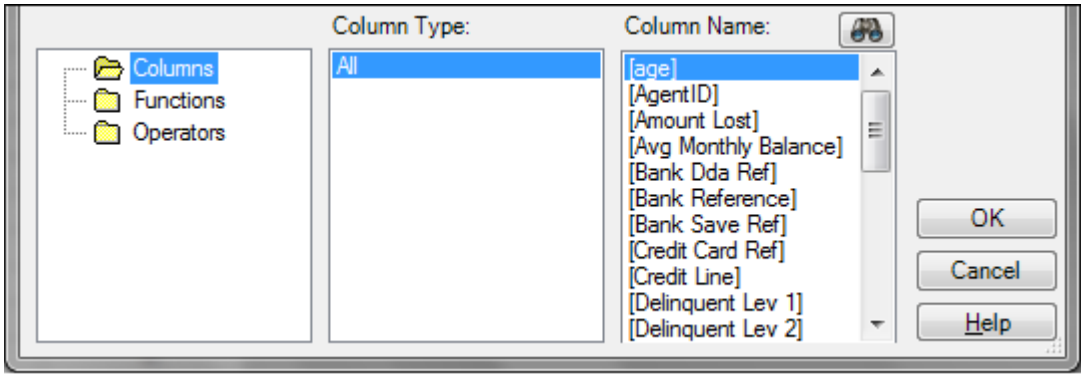
For special notes about calculated columns in the In-Database Analytics Mode, see the Variable Transformations Node section.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

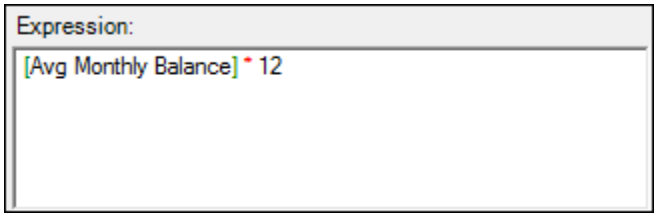
Functions and Operators

Columns

Click on the **Columns** folder to open the list of all columns in the dataset.



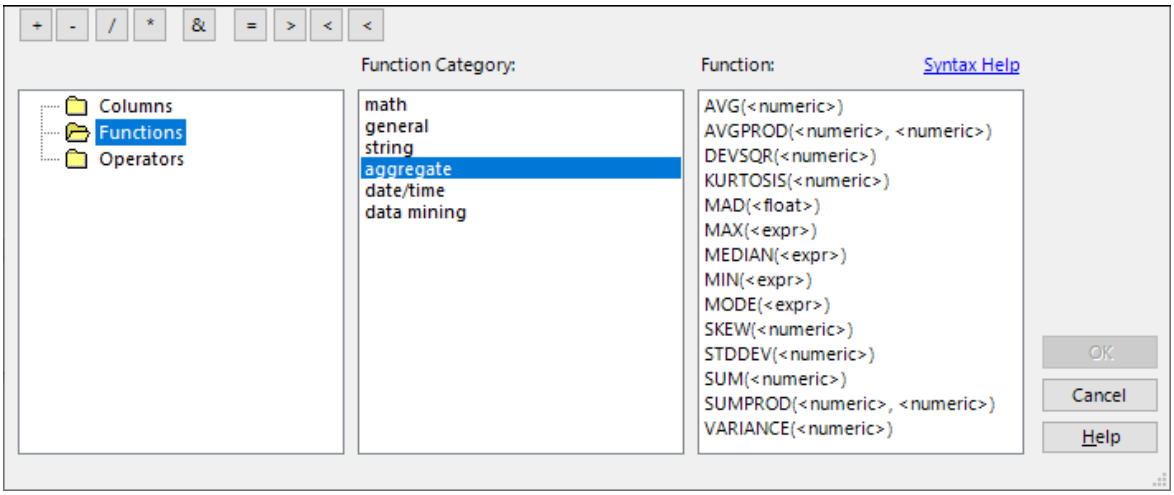
If you want to use a certain column in the calculation being defined, double-click on the column name and it will appear in the expression box. Expressions can be typed manually into the expression box. Note that column names must be enclosed in the square brackets [ ].



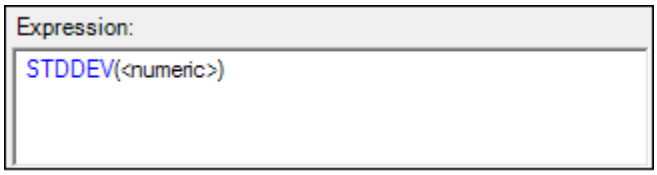
Functions

The **Functions** folder contains most commonly used mathematical, string, date/time, data mining, and other SQL functions. The **Function** box lists the functions of each category. The type of their arguments is indicated in parentheses. You can click the **Syntax Help** link above the Function box to see the syntax and examples for any function. For information and examples on these functions, see the Function Syntax topic.

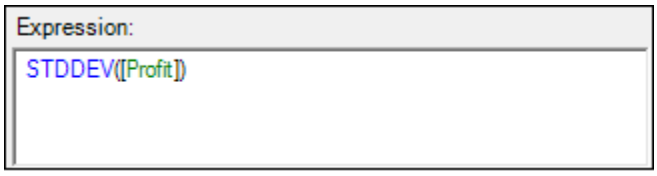
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Double-click on any function to make it appear in the expression box.



Replace the data type description in angular brackets with the desired column name, numeric value, or expression. A column name can be entered in the expression box by selecting the Columns folder and double-clicking the desired column name. The data type of the entered value or column must be compatible with the type of the function argument. For example the standard deviation of the numeric variable Profit:



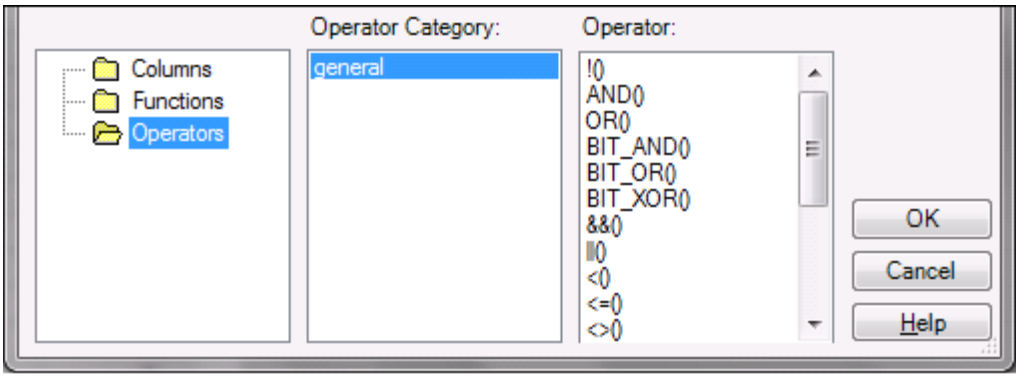
As with all expressions, these functions can be typed manually into the expression box.

Aggregate expressions can be used in the Expression Editor at the row level. For example, you can use an expression like **`[age]/avg([age])`**. Note that aggregate expressions are converted to constants when the formula is copied into a model. For example, [age]/avg([age]) becomes [age]/35.56 when the expression is embedded into a model.

Operators

The operators list contains most common operators used in SQL Queries. They include addition, subtraction, multiplication and division operators, comparison operators (<,>,=,>=,<=, etc.), logical operators AND, OR, NOT, etc., and bitwise operators.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also:

Function Syntax

Specifying Format

Division and Infinity

Two types of division are supported: the ordinary division operator (referred to as ***slash division***, or ***/***) and the ***safe division*** function `safe_division()`. They differ in the ways they treat integer arguments and extreme cases such as division by zero and the infinity value.

Slash Division (/)

1. Slash division of integers always returns an integer. Any fractional part of the result is truncated. For example, in slash division `1/2` equals 0. This behaviour is the same as in most implementations of SQL, such as in *Transact SQL* used in Microsoft SQL Server.

2. Slash division by zero normally results in the ***null*** value. No error message is provided, and the resulting *nulls* are processed by subsequent operations (if any) the usual way missing values are processed. For example, a Variable Transformation node adding a new column whose expression incudes slash division by a variable containing zeroes will run without interruption. Any divisions by zero will produce nulls in the new variable. There are contexts, however, where slash division by zero could produce an error message.

3. Slash division with both dividend and divisor being zero (`0/0`) always produces either *null*, or an error message, or NaN (not a number, i.e., "undefined" value), even when a specific user-defined value may be preferable in this case.

Safe Division

The safe division function `SAFE_DIVISION()` is available in the **math** function category. It is provided as a useful alternative to slash division to handle cases in data science computations where the above three features may not be adequate. For example, despite the

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

integer data type of operands, their division may reasonably be expected to produce a real number (e.g., 1 divided by 2 to produce 0.5), while division by zero may be expected to produce positive or negative infinity.

1. Safe division of integers always returns a real number (a value of type Number, not rounded or truncated). This behaviour is the same as in some statistical computing languages such as R or Python..

2. Safe division of a positive or negative number by zero produces positive or negative infinity, respectively. Infinity is designated as ∞ or the word "infinity", depending on the context in the application, and on the OS platform. For example, on Windows 7 it is always represented by the word "infinity".

3. Safe division with both dividend and divisor being zero can be configured to produce a specific user-defined value. This is achieved by specifying the desired value as the third argument of the SAFE_DIVISION() function.

> The capability to produce a user-defined value for the case of zero divided by zero is useful in many contexts. For example, in outlier detection, candidate outliers are often identified as values with large absolute z-score. This method presents a technical ambiguity for outlier detection in constant subsets of fields, because taking the z-score of a constant amounts to dividing zero by zero. To see this, note that a z-score equals the deviation from the mean, divided by the standard deviation. The mean of a constant is the constant itself, and the standard deviation of a constant is zero – therefore the z-score of a constant equals `0/0`. Although `0/0` is technically undefined, in this application of outlier detection, most users would expect that the z-score is zero – there are no outliers in a constant field.

Safe division is represented by a three-argument function with the following syntax:

```
SAFE_DIVISION(numerator_expression,
denominator_expression, [limit_expression])
```

The first two arguments are the same as the arguments for slash division. The third argument, limit_expression, is optional. It provides the expression to return in cases that are numerically equivalent to zero divided by zero. If only two arguments are provided, the default limit_expression is NULL.

As an example of safe division, the following code computes a *z*-score for the variable capital-gain:

```
SAFE_DIVISION([capital-gain]     -     AVG([capital-gain]),
STDDEV([capital-gain]), 0.0)
```

Infinity

The `SAFE_DIVISION()` function can return positive or negative infinity. To refer to infinity in the SQL code that you write in the Expression Editor (in the Variable Transformations node or in Strategy Tree calculations), use the **INF** token in curly brackets: **{INF}** for positive infinity

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

and **-{INF}** as negative infinity. The following example expression assigns negative infinity if a certian condition is satisfied:

```
CASE WHEN[loss_yr1] < SUM([loss_yr0]) THEN -{INF} ELSE [loss_yr1] END
```


Algebraic operations involving infinity are defined according to the usual mathematical conventions for the extended real number line.

Aggregate functions based on averages or sums:

For the aggregate functions `AVG(), STDDEV(), SKEW(), KURTOSIS(), SUM(),` etc, applied to a set of numbers that includes infinity, the results are defined by the following rules:.

Aggregate function	Set includes either positive or negative infinty (but not both)	Set includes positive and negative infinty
AVG, SUM, SKEW	+{INF} or -{INF}	NULL
STDDEV, VARIANCE, KURTOSIS	+{INF}	+{INF}

Aggregate functions based on order statistics and frequencies:

- MAX
- MIN
- MEDIAN
- MODE
- MAD (Median Absolute Deviation)

These functions operate by sorting the unique values based on value and/or frequency, and then choosing one of the values of the set (except for MAD, where a transformed version of one of the values is returned). Infinities pose no special issues for these aggregate functions based on order statistics and frequencies because infinities can be both ordered and counted.

See also:

Function Syntax

Expression Editor: Functions and Operators

Specifying Format

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Specifying Format

You can specify various formats, such as *Currency*, *Percent*, etc., and precision for displaying your data. The format you specify in Dataset Editor and its Expression Editor affects all views of the dataset except the Data tab and the Correlations tab (since they show the unformatted data) and the items depending on the dataset, such as decision trees.

- To specify a format for a existing field, apply the **Variable Transformations** node to your dataset (or open an existing one), select the field and click the **Edit Field** button. The Format String dialog opens.
- If you are adding a calculated field, you can specify the format string for it in the Expression Editor dialog.

  Note: The **data type** of a *calculated field* is determined by its expression, but you can also use explicit type conversion using the command `CAST (<expression> AS <type>)`, where type can be `STRING, DOUBLE, INTEGER, BOOLEAN, DATE, TIME, TIMESTAMP`. The data type of any field can also be changed in the Dataset Editor interface of the Variable Transformations node wizard.

You can either choose one of the standard formats from the Format String drop-down list or enter a custom format in the Format String field.

a) Standard numeric format strings

Format String	Description
Currency (C)	Represents a currency amount. The currency symbol and the default precision are defined by the settings in the Regional Options of your Windows system (see **Control Panel | Regional and Language Options**). To specify the Currency format with custom precision, you can type C`nn` in the Format String field, where `nn` is an optional integer called the precision specifier, which ranges from 0 to 99 and controls the number of significant digits or zeros to the right of a decimal. The format string cannot contain white spaces.
Percent (P)	Represents a percent. If the number is negative, the string starts with a minus sign. The converted number is multiplied by 100 in order to be presented as a percentage. You can specify the Percent format with custom precision by typing P`nn` in the Format String field, where the precision specifier `nn` is an integer that indicates the desired number of decimal places.
Scientific (S)	Represents a string of the form "**-d.ddd…E+ddd**" or "**-d.ddd…e+ddd**", where each 'd' indicates a digit (0-9). A minus sign is added if the number is negative. One digit always precedes the decimal point. The precision specifier indicates the desired number of digits after the decimal point. If the precision specifier is omitted, a default of six digits after the decimal point is used. The case of the format specifier indicates whether to prefix the exponent with an 'E' or an 'e'. The exponent always consists of a plus or minus sign and a minimum of three digits. The exponent is padded with zeros to meet this minimum, if required.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Number (N)	Represents a string of the form "**-d,ddd,ddd.ddd…**", where each '**d**' indicates a digit (0-9). A minus sign is added if the number is negative. Thousand separators are inserted between each group of three digits to the left of the decimal point. The default precision is defined by the settings in the Regional Options of your Windows system (see **Control Panel | Regional and Language Options**). You can specify the Number format with custom precision by typing N`nn` in the Format String field, where the precision specifier `nn` is an integer that indicates the desired number of decimal places.
General (G, g)	Represents the most compact of either fixed-point or scientific notation, depending on the type of the number and whether a precision specifier is present. You can specify the custom General format by typing G`nn` or g`nn` in the Format String field, where xx is the precision specifier. Fixed-point notation is used if the exponent that would result from expressing the number in scientific notation is greater than -5 and less than the precision specifier; otherwise, scientific notation is used. The result contains a decimal point if required and trailing zeroes are omitted. If the precision specifier is present and the number of significant digits in the result exceeds the specified precision, then the excess trailing digits are removed by rounding. If scientific notation is used, the exponent in the result is prefixed with 'E' if the format specifier is 'G', or 'e' if the format specifier is 'g'. The exception to the preceding rule is if the number is a Decimal and the precision specifier is omitted. In that case, fixed-point notation is always used and trailing zeroes are preserved.
Decimal (D)	Applicable for the types Integer and String. Represents a string of decimal digits (0-9), prefixed by a minus sign if the number is negative. You can specify the custom Decimal format by typing D`nn` in the Format String field, where the precision specifier `nn` is an integer that indicates the minimum number of digits desired in the resulting string. If required, the number is padded with zeros to its left to produce the number of digits given by the precision specifier. For example, with Format String specified as D10, the number 156 will be represented as 0000000156.
Fixed-Point (F)	Represents a string of the form "**-ddd.ddd…**" where each '**d**' indicates a digit (0-9). The string starts with a minus sign if the number is negative. The default precision is defined by the settings in the Regional Options of your Windows system (see **Control Panel | Regional and Language Options**). You can specify the Fixed-point format with custom precision by typing F`nn` in the Format String field, where the precision specifier `nn` is an integer that indicates the desired number of decimal places. For example, with the Format String specified as F05, the number 5.26 will be represented as 5.26000.
Round-trip	Applicable to fields of the Number type only. The round-trip specifier guarantees that a numeric value converted to a string will be parsed back into the same numeric value. When a numeric value is formatted using this specifier, it is first tested using the general format, with 15 spaces of precision. If the value is successfully parsed back to the same numeric value, it is formatted using the general format specifier. However, if the

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

	value is not successfully parsed back to the same numeric value, then the value is formatted using 17 digits of precision.
Hexadecimal (X, x)	Applicable to fields of the Integer type only. Represents a string of hexadecimal digits. If, instead of choosing 'Hexadecimal' from the drop-down list, you enter the format specifier 'X' or 'x', note that the case of the letter indicates whether to use uppercase or lowercase characters for the hexadecimal digits greater than 9. For example, use 'X' to produce "ABCDEF", and 'x' to produce "abcdef". You can specify the custom Hexadecimal format by typing X`nn` in the Format String field, where the precision specifier `nn` is an integer that indicates the minimum number of digits desired in the resulting string. If required, the number is padded with zeros to its left to produce the number of digits given by the precision specifier.

A more detailed description of standard numeric format strings is available at the Microsoft documentation site https://docs.microsoft.com/en-us/dotnet/standard/base-types/standard-numeric-format-strings.

b) Custom numeric format strings

A simple way to specify custom precision is to enter a custom numeric format string of the form 00.0000... or ##.#####..... For example, with the format string 000.0000, the number 56.73 will be displayed as 056.7300. The format string ###.#### also increases the precision to four decimal places, but does not pad the number with zeroes, so the number 56.73 is displayed in the same form (56.73). Both format strings 0.0 and #.# round 56.73 to 56.7.

Format String	Description
0 (zero placeholder)	If the value being formatted has a digit in the position where the '0' appears in the format string, then that digit is copied to the result string. The position of the leftmost '0' before the decimal point and the rightmost '0' after the decimal point determines the range of digits that are always present in the result string. The "00" specifier causes the value to be rounded to the nearest digit preceding the decimal, where rounding away from zero is always used.
# (digit placeholder)	If the value being formatted has a digit in the position where the '#' appears in the format string, then that digit is copied to the result string. Otherwise, nothing is stored in that position in the result string. Note that this specifier never displays the '0' character if it is not a significant digit, even if '0' is the only digit in the string. It will display the '0' character if it is a significant digit in the number being displayed. The "##" format string causes the value to be rounded to the nearest digit preceding the decimal, where rounding away from zero is always used.
. (decimal point)	The first '.' character in the format string determines the location of the decimal separator in the formatted value; any additional '.' characters are ignored. The actual character used as the decimal separator is determined by the settings in the Regional Options of your Windows system (see **Control Panel | Regional and Language Options**).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

A more detailed description of custom numeric format strings is given in https://docs.microsoft.com/en-us/dotnet/standard/base-types/custom-numeric-format-strings

c) Standard DateTime format strings

- Short Date
- Long Date
- Short Time
- Long Time
- Sortable Date/Time
- Universal Date/Time

Note: If you choose one of the Date/Time formats for a Time field, the default date part will be added to the Time part (the same year/month/day for all time values). For Time fields, *Long Time* and *Short Time* are the only appropriate formats to display time with or without seconds.

A detailed description of standard Date and Time format strings can be found at https://docs.microsoft.com/en-us/dotnet/standard/base-types/standard-date-and-time-format-strings.

A detailed description of custom Date and Time format strings is given at https://docs.microsoft.com/en-us/dotnet/standard/base-types/custom-date-and-time-format-strings

Function Syntax

Introduction

This topic describes the functions that can be used in the Expression Editor for adding calculated columns in datasets and node calculations in Strategy Trees. The list of all supported functions with their syntax and related examples is given below.

Functions take inputs (*arguments*) and produce the result of a certain type. For example, the mathematical function SQRT produces the square root of its numeric argument, e.g., SQRT(25)=5.0. Some functions have no arguments, such as RAND, which generates a random number, or CDATE, which produces the current date.

When used in an expression, a function name is followed by its arguments in round brackets. Functions that require no arguments must still be followed by round brackets, for example, RAND() and CDATE(). The argument can be a value, a column name, or an expression of the *data type* that is valid for this function. Otherwise the result will be *null*. For example, the argument of the function SQRT must be a nonnegative number. Column names in expressions must be enclosed in square brackets.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

All String functions and operators are case-sensitive with respect to their arguments.

See the topic Expression Editor: Functions and Operators for more general information about functions and operators.

Arguments

TYPE	DESCRIPTION
char	Character in single quotes. Examples: 's', '@', '2'.
expression	Expression constructed from arguments of other types. For example, the addition of two columns. Note: This isn't a specific data type.
float	Floating point number. Examples: 1.4, 32.54, .002.
integer	Integer number. Examples: 1, 38, -189, 0.
numeric	Any type of number.
string	String: one or more characters in single quotes. Some examples: 's', '2', and 'word'.
date	Date formatted according to the current system settings, for example, DD/MM/YYYY. Date constants used as function arguments must be enclosed in curly brackets, for example: DAYOFYEAR({2012-03-31})
timestamp	Date and time formatted according to the current system settings, for example, DD/MM/YYYY hh:mm:ss. Timestamp constants used as function arguments must be enclosed in curly brackets, for example: DAYOFYEAR({2012-03-31 17:45:51})
...	Variable number of arguments. Some functions operate on a list of items where the number of items is not predetermined.

Notation: Arguments in angle brackets <> must be included. Arguments in square brackets [ ] are optional. These brackets are only used to show whether the argument is required and should **not** be included as part of the argument. Square brackets [ ] are used to indicate a column name.

When more than one argument is required, the arguments are separated by a comma. If a function has no arguments, empty brackets must be used after its name (for example, CDATE()). This does not apply to statement keywords such as CASE, WHEN, THEN, ELSE, and END.

A Note on Aggregate Functions

Aggregate functions can be used in the Expression Editor at the row level. For example, [age]/AVG([age]). Note that aggregate expressions are converted to constants when the formula is copied into a model. For example, if AVG([age]) = 35.56, then the expression [age]/AVG([age]) becomes [age]/35.56 when embedded into a model.

The Functions

ABS

Math Function

Syntax: ABS (<float>)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

ABS returns the absolute value (or modulus) of the ***argument***. This has the effect of converting negative numbers to positive values. Zero, and numbers greater than zero, are not affected. Thus, both ABS(32) and ABS(-32) return 32.

For example, if values of Profit in your dataset can be positive or negative (loss), and we want to flag all customers associated with high profit or loss, we could create a True/False field with an expression similar to the following:

```
CASE WHEN ABS([Profit]) >= 5000 THEN true ELSE false END
```

ACOS

Math Function

Syntax: ACOS (<float>)

This trigonometric function is the inverse of COS. It returns the arc cosine of the ***argument***. It returns *null* if the argument is outside of the interval [-1, 1].

Some examples:

```
ACOS(1) = 0

ACOS(1.0001) = null

ACOS(0) = 1.5707963267949
```

ACOSH

Math Function

Syntax: ACOSH (<float>)

The ACOSH function is the inverse of COSH. It returns the inverse hyperbolic cosine of the ***argument***. It returns *null* if the argument is less than 1.

Some examples:

```
ACOSH(35) = 3.91162276521459

ACOSH(0.2) = null

ACOSH(6) = 2.47788873028848
```

ARGMAX

Math Function

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Syntax: ARGMAX (<numeric>, [<numeric>,....,<numeric>])

ARGMAX takes one or more numeric arguments and returns the index of the largest argument (i.e., its position in the list of arguments, counting from the left).

Some examples:

ARGMAX(0.25, 0.5, 0.33) = 2

ARGMAX(2, -106) = 1

ARGMIN

Math Function

Syntax: ARGMIN (<numeric>, [<numeric>,....,<numeric>])

ARGMIN takes one or more numeric arguments and returns the index of the smallest argument (i.e., its position in the list of arguments, counting from the left).

Some examples:

ARGMIN(0.25, 0.5, 0.33) = 1

ARGMIN(2, -106) = 2

ASCII

Math Function

Syntax: ASCII (<char>)

This returns the ASCII code value of the leftmost character of the ***argument***. The character set of the argument string is assumed to be ASCII. The function returns 0 if the argument is an empty string, and null if the argument is null.

Some examples:

ASCII('a') = 97

ASCII('ab') = 97

ASCII('') = 0

The inverse of ASCII is CHR. The description of the CHR function in this Help topic contains a table listing the ASCII code values.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

ASIN

Math Function

Syntax: ASIN (<float>)

ASIN is the inverse of SIN. It returns the arc sine of the ***argument***. It returns *null* if the argument is outside of the interval [-1, 1].

Some examples:

ASIN(0.2) = 0.201357920790331

ASIN(1.2) = null

ASINH

Math Function

Syntax: ASINH (<float>)

The ASINH function is the inverse of SINH. It returns the inverse hyperbolic sine of the ***argument***.

Some examples:

ASINH(35.0) = 4.24869926123636

ASINH(0.2) = 0.198690110349241

ATAN

Math Function

Syntax: ATAN (<float>)

ATAN is the inverse of TAN. It returns the arc tangent of the ***argument***.

Some examples:

ATAN(2.0) = 1.10714871779409

ATAN(-2,2) = -0.785398163397448

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Note: This function can take an optional second argument: ATAN(<float> [,float]), in which case it is equivalent to ATAN2.

ATAN2

Math Function

Syntax: ATAN2 (<float>, <float>)

ATAN2 returns the arc tangent of the two ***arguments***. It is similar to calculating the arc tangent of y/x, except that the signs of both arguments are used to determine the quadrant of the result.

Some examples:

ATAN2(-2,2) = -0.785398163397448

ATAN2(3.14159,0) = 1.57079632679489

ATANH

Math Function

Syntax: ATANH (<float>)

ATANH is the inverse of TANH. It returns the inverse hyperbolic tangent of its numeric argument. The argument must be within the interval (-1, 1).

ATANH(-0.1) = -0.10034

AVG

Aggregate Function

Syntax: AVG (<expression>)

AVG is an aggregate function that returns the average value of its argument. The argument must be a numeric column or an expression that involves only numeric columns. For example, if the argument is a numeric column, this is the average value of this column across all records of the dataset (if used for a calculated column in a dataset) or the current node (if used for a node calculation in a Strategy Tree).

For example, to get the average age for all customers in your dataset, you might use:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
AVG( [Age] )
```

The following example calculates the relative profit (profit normalized by the average profit value).

```
[Profit] / AVG( [Profit] )
```


AVGPROD

Aggregate Function

Syntax: AVGPROD (<numeric>,<numeric>)

The AVGPROD function is an aggregate function that returns the average of the products of the argument variables.

Example for two numeric variables:

```
AVG([variable_1], [variable_2])
```

CASE

General Function

Syntax: CASE WHEN <expression> THEN <expression> [WHEN <expression> THEN <expression>] [ELSE <expression>] END

CASE is a conditional statement providing if-then-else logic (flow control). Used with the keywords WHEN, THEN, ELSE, and END, the CASE statement opens a structure that allows you to branch off to different operations based on specific conditions.

The following example returns the string 'Non-Voter' when the record represents either a person under the age of 18 or a person who is not a citizen. In all other cases, 'Voter' is returned.

```
CASE
  WHEN [Age] < 18 THEN
    'Non-Voter'
  WHEN [LegalStatus] <> 'Citizen' THEN
    'Non-Voter'
  ELSE
    'Voter'
END
```

Indenting the statements is not required but it helps keep the logic visual especially when there are more than one WHEN statement.

The keywords CASE, WHEN, THEN, ELSE, and END do not require parentheses.

As usual, for any String comparisons, the operators are case-sensitive with respect to their String arguments.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

See also IN and LIKE.

CAST

General Function

Syntax: CAST (<expression> AS <type>)

CAST converts a value from one *type* to another *type* (e.g., the number 123 to the string '123', or vice versa). The argument for CAST is a combination of a value and the desired type with the keyword AS between them.

For example, to convert the value in the [ProductNumber] field from a number to a string:

```
CAST(  [ProductNumber] AS STRING )
```

To add the characters EU- to the front of the product number:

```
CONCAT( 'EU-', CAST(  [ProductNumber] AS STRING ) )
```

Valid types include: *char*, *float*, *integer*, *numeric*, *string*, and *timestamp*.

CDATE

Date/Time Function

Syntax: CDATE()

The CDATE() function returns the current date and time as a value in *MM/DD/YYYY hh:mm:ss* format.

For example, CDATE() might return:

```
CDATE() = 01/27/2013 7:42:49 AM
```

See also: CURRENT_DATE, CTIME

CDATETIME

Date/Time Function

Syntax: CDATETIME()

This is an alternative to CDATE.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Example:

CDATETIME() = 01/27/2013 7:42:49 AM

CEIL

Math Function

Syntax: CEIL(<float>)

CEIL returns the smallest integer value not less than the ***argument***. That is, the value is rounded up to the nearest integer that is greater or equal to this value.

Some examples:

CEIL(1.23) = 2

CEIL(-1.23) = -1

This is the opposite of FLOOR.

See also: CEILING

CEILING

Math Function

Syntax: CEILING(<float>)

This is an alternative to CEIL.

Example:

CEILING(4.16) = 5

CHARINDEX

String Function

Syntax: CHARINDEX(<string>,<string>)

The CHARINDEX function returns the position of the first string argument within the second string argument. If there is more than one occurrence, CHARINDEX returns the position of the first occurrence. If the first string argument is not found within the second one, CHARINDEX returns 0.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Examples:

CHARINDEX('l', 'toolbox') = 4

CHARINDEX('o', 'toolbox') = 2

CHARINDEX('box', 'toolbox') = 5

CHARINDEX('toolbox', 'box') = 0

See also: POSSTR

CHARACTER LENGTH

String Function

Syntax: CHARACTER LENGTH(<string>)

This function returns the number of characters in the string ***argument*** - spaces are included. If the argument is an empty string, 0 is returned.

Some examples:

CHARACTER LENGTH('To') = 2

CHARACTER LENGTH('To be') = 5

The following example chops the name in the [LastName] field if it is greater than 25 characters long. First it checks the length of the [LastName] field: if it is greater than 25, it truncates the name using the SUBSTRING function; if it is not, it simply returns the name.

```
CASE
   WHEN CHARACTER LENGTH( [LastName] ) > 25 THEN
      SUBSTRING( [LastName] , 1, 25)
   ELSE
      [LastName]
END
```

Note that this does not change the value in the [LastName] field but rather places the value in the field created with the Expression Editor.

See also: LEN

CHR

General Function

Syntax: CHR(<integer>, ...,<integer>)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The CHR function returns a string consisting of the characters given by the ASCII code values of the integers in the ***arguments***. Note that you can use a variable number of arguments but they must all be integers. There are 95 printable ASCII characters, numbered 32 to 126. The first character is a space.

	32	33	34	35	36	37	38	39	40
		!	"	#	$	%	&	'	(
41	42	43	44	45	46	47	48	49	50
)	*	+	,	-	.	/	0	1	2
51	52	53	54	55	56	57	58	59	60
3	4	5	6	7	8	9	:	;	<
61	62	63	64	65	66	67	68	69	70
=	>	?	@	A	B	C	D	E	F
71	72	73	74	75	76	77	78	79	80
G	H	I	J	K	L	M	N	O	P
81	82	83	84	85	86	87	88	89	90
Q	R	S	T	U	V	W	X	Y	Z
91	92	93	94	95	96	97	98	99	100
[	\	]	^	_	`	a	b	c	d
101	102	103	104	105	106	107	108	109	110
e	f	g	h	i	j	k	l	m	n
111	112	113	114	115	116	117	118	119	120
o	p	q	r	s	t	u	v	w	x
121	122	123	124	125	126				
y	z	{	|	}	~				

Some examples:

CHR(80,77,78) = PMN

CHR(64) = @

CHR(73,32,97,109) = I am

The inverse of CHR is ASCII.

CONCAT

String Function

Syntax: CONCAT(<string>,<string>)

This returns the string that results from concatenating the ***arguments***. You must pass two string arguments.

For example:

CONCAT('E', 'mail') = Email

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

In the following example a space is first added to the end of the value in the [FirstName] field, then [LastName] is added to that:

```
CONCAT( CONCAT([FirstName],' '), [LastName] )
```


COS

Math Function

Syntax: COS(<float>)

The COS function returns the cosine of the ***argument***, where the argument is given in radians.

Some examples:

```
COS(3.14159) = -1

COS(45) = 0.52532198881773
```

COSH

Math Function

Syntax: COSH(<float>)

The COSH function returns the hyperbolic cosine of the ***argument***, where the argument is given in radians.

Some examples:

```
COSH(3.14159) = 11.5919226299454

COSH(0) = 1
```

COUNT

General Function

Syntax: COUNT(<expression>)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

COUNT (*)

The COUNT function returns the number of records in the dataset (if used in an expression for a calculated column in the dataset) or the number of records in the node (if used in a node calculation in a Strategy Tree).

The argument expression type must be integer. You can also use the wildcard symbol * for the function to return the total number of input records, including duplicates and those with null values.

The following example calculates the percentage of males in the dataset. Note that it uses the CAST function to convert the COUNT result from integer to float in order for the entire result to be a floating point number rather than an integer:

```
CAST(COUNT(CASE WHEN [Gender] = 'Male' THEN 1 ELSE null END) AS double) / CAST(COUNT(*) AS double)
```

COVARIANCE

General Function

Syntax: COVARIANCE (<expression>, <expression>)

The argument expression must involve only numeric columns.

Covariance is the measure of how much two variables vary together (as distinct from variance, which measures how much a single variable varies). If two variables tend to vary together (that is, when one of them is above its expected value, then the other variable tends to be above its expected value too), then the covariance between the two variables will be positive.

On the other hand, if when one of them is above its expected value, the other variable tends to be below its expected value, then the covariance between the two variables will be negative.

If X and Y are independent, then their covariance is zero.

The converse, however, is not true: if X and Y have covariance zero, they need not be independent.

The units of measurement of the covariance(X, Y) are those of X times those of Y. By contrast, correlation, which depends on the covariance, is a dimensionless measure of linear dependence.

The following example calculates the covariance between the variables *hours-per-week* and *capital-gain*:

```
COVARIANCE([hours-per-week],[capital-gain])
```

CTIME

Date/Time Function

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Syntax: CTIME()

CTIME returns the current time as a value in seconds. In a time field, it will appear in an *hh:mm:ss* format.

These examples are all from 5:07:42 PM:

CTIME() = 61662

The following are values that would appear within a time field:

CTIME() = 17:07:42

CTIME() + 3600 = 18:07:42

See also: CURRENT_TIME

CTIMESTAMP

Date/Time Function

Syntax: CTIMESTAMP()

This is an alternative to CDATETIME

Example:

CTIMESTAMP() = 01/27/2013 07:42:49

See also: CDATE, CTIME

CTZNAME

Date/Time Function

Syntax: CTZNAME()

CTZNAME returns the system time zone name.

For example:

CTZNAME() = Eastern Standard Time

CURRENT_DATE

Date/Time Function

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Syntax: CURRENT_DATE()

This is an alternative to CDATE.

Example:

```
CURRENT_DATE() = 01/27/2013 7:42:49 AM
```

CURRENT_TIME

Date/Time Function

Syntax: CURRENT_TIME()

This is an alternative to CTIME.

Examples:

```
CURRENT_TIME() = 06:55:42
```

```
CURRENT_TIME() + 3600 = 07:55:42
```

CURRENT_TIMESTAMP

Date/Time Function

Syntax: CURRENT_TIMESTAMP()

This is an alternative to CTIMESTAMP and CDATETIME.

Example:

```
CURRENT_TIMESTAMP() = 01/27/2013 06:55:42
```

CURRENT_USER

General Function

Syntax: CURRENT_USER()

The current user is only applicable in multiple user situations. Otherwise, it returns a (null) string since there is no user name.

```
CURRENT_USER()
```

DATEADD

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Syntax: DATEADD(<string>,<integer>,<timestamp>)

The DATEADD function adds time to a timestamp value.

- The first ***argument*** specifies the type of time value to add - valid types are the strings: 'SECOND', 'MINUTE', 'HOUR', 'DAY', 'MONTH', and 'YEAR'.
- The second ***argument*** specifies the number of units to add. Negative numbers subtract the time.
- The third ***argument*** is a timestamp that specifies the starting time.

DATEADD returns a timestamp as a value in *MM/DD/YYYY hh:mm:ss* format.

In the following examples, the 0 starting time argument represents *1/1/2000 12:00:00 AM*:

```
DATEADD( 'SECOND', 45, 0 )  = 1/1/2000 12:00:45 AM

DATEADD( 'MINUTE', 22, 0 )  = 1/1/2000 12:22:00 AM

DATEADD( 'HOUR', 48, 0 )  = 1/3/2000 12:00:00 AM

DATEADD( 'DAY', 2, 0 )  = 1/3/2000 12:00:00 AM

DATEADD( 'MONTH', 4, 0 )  = 5/1/2000 12:00:00 AM

DATEADD( 'YEAR', 7, 0 )  = 1/1/2007 12:00:00 AM

DATEADD( 'YEAR', -1, 0 )  = 1/1/1999 12:00:00 AM
```

Other starting times might be specified as:

```
DATEADD( 'HOUR', -5, CDATE() )

DATEADD( 'DAY', 14, [DateToShip] )
```

DATEDIFF

Date/Time Function

Syntax: DATEDIFF(<string>,<timestamp>,<timestamp>)

The DATEDIFF function returns the difference between two timestamp values.

- The first ***argument*** specifies the type of time value to compare - valid types are the strings: 'SECOND', 'MINUTE', 'HOUR', 'DAY', 'MONTH', and 'YEAR'.
- The second ***argument*** is a timestamp that specifies the starting time.
- The third ***argument*** is a timestamp that specifies the ending time.

DATEDIFF returns the number based on the first argument type.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

In the following examples, the 0 starting time argument represents *1/1/2000 12:00:00 AM*. The ending time is in seconds:

```
DATEDIFF( 'SECOND', 0, 10 )  = 10

DATEDIFF( 'MINUTE', 0, 120 )  = 2
```


Other arguments might be specified as:

```
DATEDIFF( 'DAY', [OrderDate], CDATE() )

DATEDIFF( 'MONTH', [OrderDate], [ShippedDate] )
```

You may wish to use the ROUND function:

```
ROUND(  DATEDIFF( 'YEAR', [AccountOpenDate], [AccountCloseDate] ) )
```

DATENAME

Date/Time Function

Syntax: DATENAME(<string>,<timestamp>)

The DATENAME function returns a string representing a part of a timestamp, that is, it returns a value such as 'Friday', 'December', or '2007'.

- The first ***argument*** specifies the type of time value to return - valid types are the strings: 'SECOND', 'MINUTE', 'HOUR', 'DAY', 'MONTH', and 'YEAR'.
- The second ***argument*** is a timestamp that specifies the time or date.

In the following examples, the current date returned by CDATE is *12/28/2006 04:30:00 PM*:

```
DATENAME( 'HOUR', CDATE() )  = '16'

DATENAME( 'DAY', CDATE() )  = 'Thursday'

DATENAME( 'MONTH', CDATE() )  = 'December'

DATENAME( 'YEAR', CDATE() )  = '2006'
```

To return a numeric value rather than a string, see DATEPART.

DATEPART

Date/Time Function

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Syntax: DATEPART(<string>,<timestamp>)

The DATEPART function returns a number representing a part of a timestamp.

- The first ***argument*** specifies the type of time value to return - valid types are the strings: 'SECOND', 'MINUTE', 'HOUR', 'DAY', 'MONTH', and 'YEAR'.
- The second ***argument*** is a timestamp that specifies the time or date.

In the following examples, the current date returned by CDATETIME is *12/28/2006 17:30:15*

```
DATEPART( 'HOUR', CDATETIME() )  = 17

DATEPART( 'DAY', CDATETIME() )  = 28

DATEPART( 'MONTH', CDATETIME() )  = 12

DATEPART( 'YEAR', CDATETIME() )  = 2006
```

To return a string value rather than a number, see DATENAME.

See also: SECOND, MINUTE, HOUR, DAY, WEEK, MONTH, YEAR

DAY

Date/Time Function

Syntax: DAY(<date>) or DAY (<timestamp>)

The DAY function returns a number representing the day part of the date or timestamp argument. This is the same as DATEPART('DAY', <timestamp>)

Example:

If the current date returned by CDATETIME is *12/28/2006 17:30:15,* then

```
DAY(CDATETIME() )  = 28
```

To return a string value rather than a number, see DATENAME.

See also: DATEPART, SECOND, MINUTE, HOUR, WEEK, MONTH, YEAR

DAYOFYEAR

Date/Time Function

Syntax: DAYOFYEAR(<date>)
DAYOFYEAR (<timestamp>)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The DAYOFYEAR function returns the day of the year expressed as a value between 1 and 366 representing the specified date or timestamp argument.

Examples:

```
DAYOFYEAR({2013-02-05}) = 36

DAYOFYEAR({2011-12-31}) = 365

DAYOFYEAR({2012-12-31}) = 366
```

If the current date returned by CDATE is 2013-03-18, then

```
DAYOFYEAR(CDATE()) = 77
```

See also: DATEPART, DAY

DAYOFWEEK

Date/Time Function

Syntax: DAYOFWEEK(<date>)

DAYOFWEEK (<timestamp>)

The DAYOFWEEK function returns a number from 1 (Sunday) to 7 (Saturday) representing the day of the week for the specified date or timestamp argument.

Examples:

```
DAYOFWEEK({10/03/2013}) = 1
```

If the current date returned by CDATE is *16/03/2013,* then

```
DAYOFWEEK(CDATE()) = 7
```

See also: WEEKDAY, DATEPART, DAY

DEGREES

Math Function

Syntax: DEGREES(<float>)

The DEGREES function returns the ***argument*** converted from radians to degrees.

Some examples:

```
DEGREES(3.14159) = 180
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

DEGREES(1.0995575) = 63

This function is the opposite of RADIANS.

DEVSQR

Aggregate Function

Syntax: DEVSQR(<numeric>)

DEVSQR is an aggregate function that returns the sum of the squared deviations from the mean of the argument variable or array.

Syntax examples:

DEVSQR([variable_1])

DEVSQR([variable_1] * [variable_2])

EXP

Math Function

Syntax: EXP(<float>)

The EXP function returns the value of *e* (the base of natural logarithms) raised to the power of the ***argument***.

Some examples:

EXP(2) = 7.38905609893065

EXP(0) = 1

EXP(-2) = 0.135335283236613

FINITE

General Function

Syntax: FINITE(<numeric> [,<numeric>])

The FINITE() function replaces an infinite value ∞ in the first argument (a numeric field) with a finite value defined by the second argument or with *null* if the second argument is not specified. See the topic Division and Infinity for detailed information on infinity.

Examples:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

FINITE({INF}) = FINITE(-{INF}) = null

FINITE({INF},99) = FINITE(-{INF},99) = 99

For a numeric variable num_var1 that contains infinite values ∞ ({INF} or -{INF}):

FINITE([num_var1]) is equal to the value of num_var1 whenever it is finite and *null* otherwise

FINITE([num_var1], 999) is equal to the value of num_var1 whenever it is finite and 999 otherwise

FINITE([num_var1], [num_var2]) is equal to the value of num_var1 whenever it is finite and the value of the variable num_var2 otherwise

FLOOR

Math Function

Syntax: FLOOR(<float>)

FLOOR returns the largest integer value not greater than the ***argument***. That is, the value is rounded down to the nearest integer. This is the opposite of CEIL.

Some examples:

FLOOR(1.23) = 1

FLOOR(-1.23) = -2

GETDATE

Date/Time Function

Syntax: GETDATE()

Returns the current system date in the standard internal format for datetime values.

Example:

GETDATE() = 24/04/2013

See also: CDATE

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

GETDATETIME

Date/Time Function

Syntax: GETDATETIME()

Returns the current system date and time in the standard internal format for datetime values.

See also: CDATETIME

GETUTCDATE

Date/Time Function

Syntax: GETUTCDATE()

Returns the current system date and time in the Coordinated Universal Time (UTC) standard.

HOUR

Date/Time Function

Syntax: HOUR (<timestamp>)

The HOUR function returns a number representing the hour part of the date or timestamp argument. This is the same as DATEPART('HOUR', <timestamp>)

Example:

If the current date returned by CDATETIME is *12/28/2006 17:30:15,* then

```
HOUR( CDATETIME() )  = 17
```

See also: DATEPART, SECOND, MINUTE, DAY, WEEK, MONTH, YEAR

IN

General Function

Syntax: IN(<expression> {,<expression>,....})

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

This function determines if a given value matches any value in a list. Items in the list are separated by commas. The result is of boolean type (True or False).

The following example flags records in which the [native-country] field contains either Mexico, Canada, or United-States:

```
CASE
   WHEN  [native-country] IN('Mexico', 'Canada', 'United-States') then
      true
   ELSE
      false
END
```

See also CASE.

ISDATE

Date/Time Function

Syntax: ISDATE(<string>)

ISDATE returns a TRUE/FALSE value based on whether the ***argument*** is a valid date.

Some examples:

```
ISDATE( 'word' ) = FALSE

ISDATE( '1/1/2000 12:00:45 AM' ) = TRUE

ISDATE( 'sunday' ) = TRUE

ISDATE( '12:00:00 AM' ) = TRUE

ISDATE( 'May 10, 2007' ) = TRUE
```

LCASE

String Function

Syntax: LCASE(<string>)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

This function returns the string ***argument*** with all characters changed to lowercase.

For example:

```
LCASE('MODEL') = model
```

In the following example, we can flag all records with the name *Smith* regardless of whether the record contains *Smith*, *SMITH*, or even *sMith*. Here, we populate a True/False field with the expression:

```
CASE WHEN LCASE([LastName]) = 'smith' THEN true ELSE false END
```

This is the opposite of UCASE.

LEN

String Function

Syntax: LEN(<string>)

This is an alternative to CHARACTER LENGTH.

Examples:

```
LEN('UK') = 2
```

```
LEN('United Kingdom') = 14
```

LIKE

General Function

Syntax: LIKE <string>

This determines whether or not a given character string matches a specified pattern.

A pattern can include regular characters and wildcard characters. During pattern matching, regular characters must exactly match the characters specified in the character string. Wildcard characters, however, can be matched with any fragments of the character string. Using wildcard characters makes the LIKE operator more flexible than using the = and != string comparison operators.

WILDCARD	EXPLANATION
%	Any string of zero or more characters.
_ (underscore)	Any single character.
[ ]	Any single character within the specified range ([a-f]) or set ([abcdef]).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

[^]	Any single character not within the specified range ([^a-f]) or set ([^abcdef]).

In the following example, a match is found when the ProductCode begins with any character and is followed by MMV- plus any number of other characters (e.g., uMMV-2983 or 1MMV-kratt43).


```
CASE
   WHEN  [ProductCode] LIKE '_MMV-%' then
      true
   ELSE
      false
END
```

See also CASE.

LOG

Math Function

Syntax: LOG(<float>)

The LOG function returns the natural logarithm of the ***argument***. The argument must be larger than zero.

Some examples:

```
LOG(2) = 0.693147180559945
```

If you want the log of a number *X* to some arbitary base *B*, use the formula LOG(X)/LOG(B).

LOG10

Math Function

Syntax: LOG10(<float>)

The LOG10 function returns the base-10 logarithm of the ***argument***. The argument must be larger than zero.

Some examples:

```
LOG10(2) = 0.301029995663981
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

LOG10(100) = 2

LOG2

Math Function

Syntax: LOG2(<float>)

The LOG2 function returns the base-2 logarithm of the ***argument***. The argument must be larger than zero.

Some examples:

LOG2(2) = 1

LOG2(3) = 1.5849625007211561

LOWER

String Function

Syntax: LOWER(<string>)

This is an alternative to LCASE.

LTRIM

String Function

Syntax: LTRIM(<string>)

The LTRIM function returns the string ***argument*** with leading space characters removed.

For example:

LTRIM(' word') = word

This function is related to RTRIM and TRIM.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

MAD

Aggregate Function

Syntax: MAD(<expression>)

The aggregate function MAD() is the median absolute deviation of a numeric variable defined by <expression> . It is equal to the median of the absolute deviations from the median of the argument expression. See MEDIAN.

For example, the median absolute deviation of the Age column in a record set is given by

```
MAD([Age])
```

This function is related to MEDIAN.

MAX

Aggregate Function

Syntax: MAX(<expr>,<expr>,[...])

MAX returns the largest value in the list of ***arguments***. You must have at least two arguments which should all be of the same type: numeric or string.

The arguments are compared and the value type returned using these rules:

- If all arguments are integers, they are compared as integers.
- If all arguments are numbers, but not all are integers, they are compared as real numbers.
- If all arguments are strings, they are compared as case-sensitive strings.

Some examples:

```
MAX(2,0) = 2

MAX(34.0,3,5.0,767.0) = 767

MAX('B','A','C') = C

MAX('b','A','C') = b
```

MEDIAN

Aggregate Function

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Syntax: MEDIAN(<expression>)

MEDIAN returns the middle value in the numeric column specified by the argument. This is the middle value across all records. When there is an odd number of records, the median is simply the middle number. When there is an even number of records, the median is the mean of the two middle numbers.

For example, to get the median age in a record set, you might use:

```
MEDIAN( [Age] )
```

The following example flags records with high incomes if the median age in the dataset is less than 30. However, it returns False when an income is low or in all cases if the median age is greater than 29.

```
CASE
   WHEN MEDIAN(  [age] ) < 30 THEN
      CASE
         WHEN [Income] = ( '>50K' ) THEN
            True
         ELSE
             False
      END
   ELSE
      False
END
```

MIN

Aggregate Function

Syntax: MIN(<expr>,<expr>,[...])

MIN returns the smallest value in the list of ***arguments***. You must have at least two arguments which should all be of the same type: numeric or string.

The arguments are compared and the value type returned using these rules:

- If all arguments are integers, they are compared as integers.
- If all arguments are numbers, but not all are integers, they are compared as real numbers.
- If all arguments are strings, they are compared as case-sensitive strings.

Some examples:

```
MIN(2,0) = 0
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

MIN(34.0,3,5.0,767.0) = 3

MIN('B','A','C') = A

MIN('a','A','C') = A

MINUTE

Date/Time Function

Syntax: MINUTE (<timestamp>)

The MINUTE function returns a number representing the minute part of the date or timestamp argument. This is the same as DATEPART('MINUTE', <timestamp>)

Example:

If the current date returned by CDATETIME is *12/28/2006 17:30:15,* then

MINUTE(CDATETIME()) = 30

See also: DATEPART, SECOND, HOUR, DAY, WEEK, MONTH, YEAR

MKDATE

Date/Time Function

Syntax: MKDATE(<integer>,<integer>,<integer>)

The MKDATE function returns a timestamp value based on the integer arguments.

- The first ***argument*** specifies the year.
- The second ***argument*** specifies the month.
- The third ***argument*** specifies the day.

If the day argument is larger than possible, the month value is increased. Likewise, if the month argument is larger than 12, the year value is increased.

Some examples:

MKDATE(2007, 10, 29) = 10/29/2007 12:00:00 AM

MKDATE(2007, 10, 32) = 11/1/2007 12:00:00 AM

MKDATE(2007, 13, 25) = 1/25/2008 12:00:00 AM

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

MKTIME

Date/Time Function

Syntax: MKTIME(<integer>,<integer>,<integer>)

The MKTIME function returns a time value based on the integer arguments.

- The first ***argument*** specifies the hour.
- The second ***argument*** specifies the minute.
- The third ***argument*** specifies the second (time).

If the second (time) argument is larger than 59, the minute value is increased. Likewise, if the minute argument is larger than 59, the hour value is increased. The hour value is simply rolled over after 24.

Some examples:

```
MKTIME( 10, 30, 0 ) = 10:30:00

MKTIME( 10, 30, 72 ) = 10:31:12

MKTIME( 10, 61, 0 ) = 11:01:00

MKTIME( 25, 30, 0 ) = 01:30:00
```

MKTIMESTAMP

Date/Time Function

Syntax: MKTIMESTAMP(<integer>,<integer>,<integer>,<integer>,<integer>,<integer>)

The MKTIMESTAMP function returns a timestamp value based on the six integer arguments. This is a combination of the MKDATE and MKTIME functions.

- The first ***argument*** specifies the year.
- The second ***argument*** specifies the month.
- The third ***argument*** specifies the day.
- The fourth ***argument*** specifies the hour.
- The fifth ***argument*** specifies the minute.
- The sixth ***argument*** specifies the second (time).

The same rules apply here as for MKDATE and MKTIME.

For example:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
MKTIMESTAMP( 2006, 10, 23, 8, 45, 30 ) = 10/23/2006 8:45:30
```

MODE

Aggregate Function

Syntax: MODE(<expression>)

Mode returns the most common value in the column specified by the argument. This is the most common value across all records.

For example, to get the most common country name in a record set, you might use:

```
MODE( [Country] )
```

The following example flags records with high incomes if the most common country name in the dataset is United States. However, it returns False when an income is low or in all cases if the most common country name is something else.

```
CASE
   WHEN MODE( [Country] ) = 'United States' THEN
      CASE
         WHEN [Income] = ( '>50K' ) THEN
            True
         ELSE
             False
      END
   ELSE
      False
END
```

MONTH

Date/Time Function

Syntax: MONTH (<date>)

MONTH (<timestamp>)

The MONTH function returns a number representing the month part of the date or timestamp argument. This is the same as DATEPART('MONTH', <timestamp>)

Example:

If the current date returned by CDATETIME is *12/28/2006 17:30:15,* then

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
MONTH( CDATETIME() )  = 12
```

See also: DATEPART, SECOND, MINUTE, HOUR, DAY, WEEK, YEAR

MONTHNAME

Date/Time Function

Syntax: MONTHNAME(<integer>)

This returns the name of the month based on the integer argument.

For example:

```
MONTHNAME( 1 ) = 'January'
```

NULLIF

General Function

Syntax: NULLIF(<expr>,<expr>)

NULLIF returns a null value if its two arguments equivalent. It returns the value of the first expression if they are not equivalent. Note that NULLIF is case sensitive when string types are compared.

Some examples:

```
NULLIF( 1, 1 ) = null

NULLIF( 3, 2 ) = 3

NULLIF( 'one', 'one' ) = null

NULLIF( 'one', 'One' ) = 'one'
```

PARSEPARTS

String Function

Syntax: PARSEPARTS(<string>,<string>,<integer>,<integer>)

The PARSEPARTS function works as follows:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Step 1. The first string argument is split into a list of substrings using the first character of the second argument (the delimiter argument).

Step 2. Each substring is numbered from 0. For example, in the case of PARSEPARTS('a,b,c,d,e', ',$$!', 2, 4) the list would be

Number	Sub-string
0	a
1	b
2	c
3	d
4	e

Step 3. The return value is built up from concatenating the substrings in the range defined by the third and fourth arguments, inserting the delimiter argument between each substring. In the case of PARSEPARTS('a,b,c,d,e', ',$$!', 2, 4), the range is from the 2nd to 4th substring. This produces the result “c,$$!d,$$!e”

Examples:

PARSEPARTS('a,b,c,d,e', ',$$!', 2, 4) = c,$$!d,$$!e

PARSEPARTS('a,b,c,d,e', ',$$!', 2, 3) = c,$$!d

PASTE

String Function

Syntax: PASTE(<string>,<string>,<string>,[<string>.....])

The PASTE function concatenates its string arguments, starting from the second argument. The first argument is used as a separator between the terms in the resulting string. This function is similar to the PASTE function in R, except that the separator is specified as the first argument instead of the *sep* option in R.

The string arguments must be enclosed in single quotes. At least 3 arguments must be specified for a meaningful output.

Examples:

PASTE('-','A','B') = 'A-B'

PASTE('-','A','B','C') = 'A-B-C'

PASTE('-','A') = 'A'

PASTE('A') = '' (empty string)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

PFINITE

General Function

Syntax: PFINITE(<numeric> [,<numeric>])

This function is a variant of FINITE: It provides positive finite conversion by replacing nonpositive and infinite values of the first argument with the value of the second argument.

The PFINITE() function replaces an infinite value ∞, zero, or any negative value in the first argument (a numeric field) with a finite value defined by the second argument or with *null* if the second argument is not specified. See the topic Division and Infinity for detailed information on infinity.

Examples:

```
PFINITE({INF}) = PFINITE(-{INF}) = null

PFINITE(-4) = null

PFINITE({INF},99) = PFINITE(-{INF},99) = 99

PFINITE(-4, 99) = 99
```

For a numeric variable num_var1 that contains infinite values ∞ ({INF} or -{INF}) or zeroes or negative values:

PFINITE([num_var1]) is equal to the value of num_var1 whenever it is a positive finite number and is *null* otherwise

PFINITE([num_var1], 999) is equal to the value of num_var1 whenever it is a positive finite number and 999 otherwise

PFINITE([num_var1], [num_var2]) is equal to the value of num_var1 whenever it is a positive finite number and the value of the variable num_var2 otherwise

POSSTR

String Function

Syntax: POSSTR(<string>,<string>)

The POSSTR function returns the position of the second string argument within the first string argument. If there is more than one occurrence, POSSTR returns the position of the first occurrence. If the second string argument is not found within the first one, POSSTR returns 0.

```
POSSTR('toolbox', 'o') = 2

POSSTR('toolbox', 'box') = 5
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

POSSTR('box', 'toolbox') = 0

See also: CHARINDEX

POW

Math Function

Syntax: POW(<float>,<float>)

The POW function returns the value of the first ***argument*** raised to the power of the second ***argument***. This is the same as the POWER function.

Some examples:

POW(2,2) = 4

POW(2,-2) = 0.25

POW(2,3) = 8

POW(3,2) = 9

POWER

Math Function

Syntax: POWER(<float>,<float>)

This is an alternative to POW.

QUARTER

Date/Time Function

Syntax: QUARTER(<date>)

QUARTER (<timestamp>)

The QUARTER function returns the quarter of the year in which the date or timestamp argument occurs. It is expressed as a value between 1 and 4.

Some examples:

QUARTER({01/01/2013}) = 1

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

QUARTER({24/12/2012}) = 4

See also: DATEPART, MONTH, MONTHNAME

RADIANS

Math Function

Syntax: RADIANS(<float>)

The RADIANS function returns the ***argument*** converted from degrees to radians.

Some examples:

RADIANS(180) = 3.14159265358979

RADIANS(63) = 1.09955742875643

RADIANS(90) = 1.5707963267949

This function is the opposite of DEGREES.

RAND

General Function

Syntax: RAND()

The RAND function returns a random floating-point value in the range 0 to 1.0.

The following values may be returned:

RAND() = 0.44825727492570877

RAND() = 0.7087954482994973

RAND() = 0.98769719549454749

RANDOM

General Function

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Syntax: RANDOM()

This is an alternative to RAND.

REPEAT

String Function

Syntax: REPEAT(<string>,<integer>)

REPEAT returns a string consisting of the first ***argument*** repeated by the number of times specified by the second argument. If the second argument is less than 1, REPEAT returns a null string.

Some examples:

REPEAT('Loo', 2) = 'LooLoo'

REPEAT('-', 8) = '--------'

REPEAT('Can', 0) = '(null)'

REPLACE

String Function

Syntax: REPLACE(<string>,<string>,<string>)

This function replaces all or some of the string in the first ***argument***. The second ***argument*** determines what is replaced while the third ***argument*** is the replacement text. If the value of the second argument is not found, the original value in the first argument is returned.

Some examples:

REPLACE('overview', 'view', 'take') = 'overtake'

REPLACE('overview', 'cast', 'take') = 'overview'

REPLACE([ProductNumber], '-US', '-EU')

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

ROUND

Math Function

Syntax: ROUND(<float>,<integer>)

ROUND returns the first ***argument*** , rounded to the number of decimals specified by the second ***argument***. If the second argument is 0, the result will have no decimal point or fractional part. To round values to the nearest ten, use -1, to the nearest hundred, use -2, and so on.

Some examples:

```
ROUND(1.398, 1) = 1.4

ROUND(2.298, 0) = 2

ROUND(26, -1) = 30

ROUND(75, -2) = 100
```

ROW_NUMBER

General Function

Syntax: ROW_NUMBER()

The ROW_NUMBER function returns the sequential number of a record (row) in the current dataset, starting at 1 for the first record. You can use this function, for example, to add an index column.

The following example expression for a new calculated column populates its values with record numbers for all records up to #1000 and the rest with nulls:

```
CASE WHEN ROW_NUMBER() <= 1000 THEN ROW_NUMBER() ELSE null END
```

RTRIM

String Function

Syntax: RTRIM(<string>)

The RTRIM function returns the string ***argument*** with trailing space characters removed.

For example:

```
RTRIM('word   ') = 'word'
```

This function is related to LTRIM and TRIM.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

SAFE_DIVISION

Math Function

Syntax: SAFE_DIVISION(<numerator_expression>, <denominator_expression>, [limit_expression])

All arguments must be numeric.

The first two arguments are the same as the arguments for regular slash division (**/**). The third argument, *limit_expression*, is optional. It provides the expression to return in cases that are numerically equivalent to zero divided by zero. If only two arguments are provided, the default limit_expression is **{NaN}**, which is a token that designates an undefined value ("not a number"). NaN values are treated identically to *null*. The purpose of having {NaN} in addition to NULL is to make a distinction between defined singular or missing values on one hand, and *undefined* values on the other (e.g., when a function is applied to arguments that are outside of its domain of definition).

SAFE_DIVISION() is similar to the ordinary slash division (**/**), with the following exceptions:

1. Safe division of integers always returns a real number (a value of type Number, not rounded or truncated). This behaviour is the same as in some statistical computing languages such as R or Python..

2. Safe division of a positive or negative number by zero produces positive or negative infinity, respectively. Infinity is designated as ∞ or the word "infinity", depending on the context in the application, and on the OS platform. For example, on Windows 7 it is always represented by the word "infinity". See the topic Division and Infinity for detailed information on infinity.

3. Safe division with both dividend and divisor being zero can be configured to produce a specific user-defined value. This is achieved by specifying the desired value as the third argument of the safe_division() function.

> The capability to produce a user-defined value for the case of zero divided by zero is useful in many contexts. For example, in outlier detection, candidate outliers are often identified as values with large absolute z-score. This method presents a technical ambiguity for outlier detection in constant subsets of fields, because taking the z-score of a constant amounts to dividing zero by zero. To see this, note that a z-score equals the deviation from the mean, divided by the standard deviation. The mean of a constant is the constant itself, and the standard deviation of a constant is zero – therefore the z-score of a constant equals `0/0`. Although `0/0` is technically undefined, in this application of outlier detection, most users would expect that the z-score is zero – there are no outliers in a constant field.

Example: The following code computes a z-score for the variable capital-gain:

```
SAFE_DIVISION([capital-gain] - AVG([capital-gain]), STDDEV([capital-gain]), 0.0)
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

SECOND

Date/Time Function

Syntax: SECOND (<timestamp>)

The SECOND function returns a number representing the seconds part of the date or timestamp argument. This is the same as DATEPART('SECOND', <timestamp>)

Example:

If the current date returned by CDATETIME is *12/28/2006 17:30:15,* then

```
SECOND( CDATETIME() )  = 15
```

See also: DATEPART, MINUTE, HOUR, DAY, WEEK, MONTH, YEAR

SIN

Math Function

Syntax: SIN(<float>)

Returns the sine of the ***argument***, where the argument is given in radians.

For example:

```
SIN(3.14159) = 0
```

SINH

Math Function

Syntax: SINH(<float>)

Returns the hyperbolic sine of the ***argument***, where the argument is given in radians.

For example:

```
SINH(1.57079632679) = 2.30129890231
```

SOUNDEX

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Syntax: SOUNDEX(<string>)

This returns a *soundex* string based on the value in the ***argument***. Two strings that sound almost the same should have identical soundex strings. All international alpha characters outside the A-Z range are treated as vowels.

Some examples:

SOUNDEX('Hello') = 'H400'

SOUNDEX('Halo') = 'H400'

SOUNDEX('Quadratically') = 'Q363'

SQRT

Math Function

Syntax: SQRT(<float>)

The SQRT function returns the square root of the ***argument***. The argument must be a nonnegative number.

For example:

SQRT(4.0) = 2

SQRT(20.0) = 4.47213595499958

If the argument is negative, the (null) value is returned.

SQUARE

Math Function

Syntax: SQUARE(<float>)

This returns the square of the ***argument***.

For example:

SQUARE(5) = 25

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

STDDEV

Aggregate Function

Syntax: STDDEV(<numeric>)

STDDEV returns the standard deviation in the numeric column specified by the argument. This is the standard deviation across all records.

For example, to get the standard deviation of age in a record set, you might use:

```
STDDEV( [Age] )
```

SUBSTR

String Function

Syntax: SUBSTR(<string>,<integer>,<integer>)

SUBSTR returns a string found within the first ***argument***. The second ***argument*** determines the starting position where 1 is the first character. The third argument determines the length of the returned string. If there is no string found, a (null) string is returned.

Some examples:

```
SUBSTR( '25-XMT', 4, 3 ) = 'XMT'

SUBSTR( '25-XMT', 1, 2 ) = '25'

SUBSTR( '25-XMT', 0, 1 ) = null
```

In the following example returns 'United States' if the two letters in the 10th and 11th position are 'US'. Otherwise, the return string is 'International'.

```
CASE
   WHEN  SUBSTR( [ProductCode], 10, 2) = 'US' THEN
      'United States'
   ELSE
      'International'
END
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

SUBSTRING

String Function

Syntax: SUBSTRING(<string>,<integer>,<integer>)

This is an alternative to SUBSTR.

SUM

Aggregate Function

Syntax: SUM(<expression>)

The aggregate function SUM returns the sum of its argument - a numeric column or an expression involving numeric columns. This is the sum over all records in the dataset if it is used for a calculated column or over all records in the node if used in a node calculation.

For example, to get the sum of Amount Lost in a record set, you might use:

```
SUM( [Amount Lost] )
```

SUMPROD

Aggregate Function

Syntax: SUMPROD(<numeric>,<numeric>)

SUMPROD is an aggregate function that returns the sum of the products of the argument variables.

Example for two numeric variables:

```
SUMPROD([variable_1],[variable_2])
```

TAN

Math Function

Syntax: TAN(<float>)

The TAN function returns the tangent of the ***argument***, where the argument is given in radians.

For example:

```
TAN(3.14159 + 1) = 1.5574077246549018
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

TANH

Math Function

Syntax: TANH(<float>)

The TANH function returns the hyperbolic tangent of the ***argument***, where the argument is given in radians.

For example:

```
TANH(1.57079632679) = 0.91715233566649657
```

TRIM

String Function

Syntax: TRIM(<string>)

The TRIM function returns the string ***argument*** with all leading and trailing spaces removed.

For example:

```
TRIM('  word   ') = 'word'
```

See also: LTRIM and RTRIM.

TRUNC

Math Function

Syntax: TRUNC(<float>,<integer>)

The TRUNC function returns the first ***argument*** truncated to the number of decimals specified by the second ***argument***. The value is not rounded up or down but simply cut off. If the second argument is 0, the result will have no decimal point or fractional part.

Some examples:

```
TRUNC(1.223,1) = 1.2

TRUNC(1.999,2) = 1.99

TRUNC(1.999,0) = 1
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

TRUNCATE

Math Function

Syntax: TRUNCATE(<float>,<integer>)

This is an alternative to TRUNC.

UCASE

String Function

Syntax: UCASE(<string>)

This function returns the string ***argument*** with all characters changed to uppercase according to the current character set mapping (the default is ISO-8859-1 Latin1).

For example:

```
UCASE('Part') = 'PART'
```

In the following example, we can flag all records with the name *Smith* regardless of whether the record contains *Smith*, *SMITH*, or even *sMith*. Here, we populate a True/False field with the expression:

```
CASE WHEN UCASE([LastName]) = 'SMITH' THEN true ELSE false END
```

See also: LCASE.

UPPER(<string>)

String Function

Syntax: UPPER(<string>)

This is an alternative to UCASE.

WEEK

Date/Time Function

Syntax: WEEK (<date>)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

WEEK (<timestamp>)

The WEEK function returns a number representing the week part of the date or timestamp argument. This is the same as DATEPART('WEEK', <timestamp>)

Example:

If the current date returned by CDATE is *12/28/2006,* then

```
WEEK( CDATE() ) = 52
```

See also: DATEPART, SECOND, MINUTE, HOUR, DAY, MONTH, YEAR

WEEKDAY

Date/Time Function

Syntax: WEEKDAY (<date>)

WEEKDAY (<timestamp>)

This is an alternative to DAYOFWEEK.

Examples:

```
WEEKDAY({10/03/2013}) = 1
```

If the current date returned by CDATE is *16/03/2013,* then

```
WEEKDAY(CDATE()) = 7
```

See also: DATEPART, DAY

YEAR

Date/Time Function

Syntax: YEAR (<date>)

YEAR (<timestamp>)

The YEAR function returns a number representing the year part of the date or timestamp argument. This is the same as DATEPART('YEAR', <timestamp>)

Example:

If the current date returned by CDATE is *12/28/2006,* then

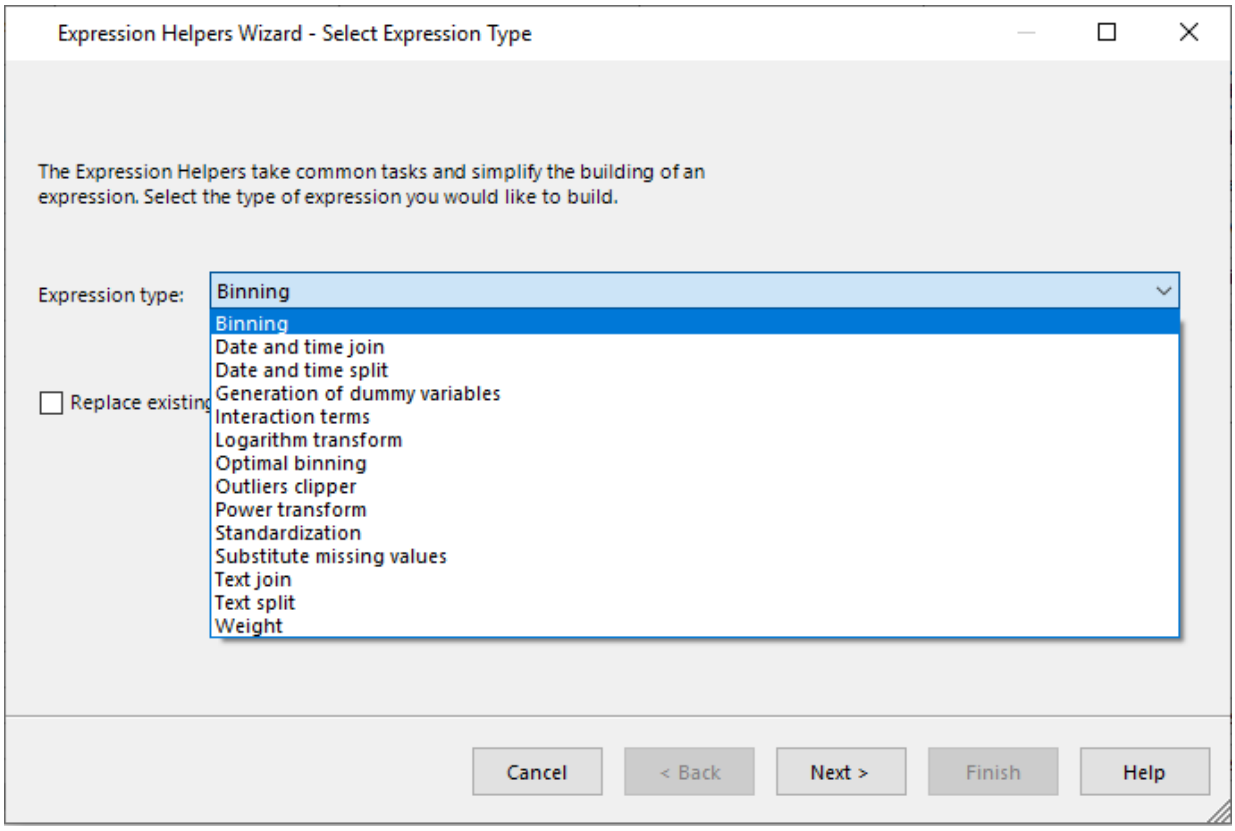
```
YEAR( CDATE() ) = 2006
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

See also: DATEPART, SECOND, MINUTE, HOUR, DAY, WEEK, MONTH

Expression Helpers

The Expression Helpers take common tasks and simplify the building of an expression. Some of the common tasks are: Grouping and mapping categories of discrete variables; creating and modifying ranges of continuous and numeric variables; missing value substitution; creating dummy variables; etc.



Binning

Allows you to create a new column from existing columns by establishing new ranges for continuous variables or new category groupings for discrete variables. See Helper: Binning for details.

Date and time join

Allows you to easily join date and time information that is spread across multiple fields into a single Date, Time, or Timestamp field. See Helper: Date and Time Join for details.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Date and time split

Allows you to split a Date, Time, or Timestamp field into several fields representing the date/time components. See Helper: Date and Time Split for details.

Generation of Dummy Variables

Allows you to create a set of dummy, or indicator variables from a discrete variable. A single column with several string values is transformed into a series of binary columns with values 0 and 1. Note that, as a part of normalization process in predictive and cluster model training, all String variables are automatically transformed into a set of indicator variables. The helper allows you to do this explicitly, outside of the model building process.

See the topic Helper: Generation of Dummy Variables for details.

Interaction Terms

Allows you to create many variables based on the product of the selected variables and/or their squares. See Helper: Interaction Terms for details.

Logarithm Transform

Allows you to create a new column by taking the Logarithm of an existing column. See Helper: Log Transform for details.

Optimal Binning

Allows you to create transformations that group each field (or variable) into new categories based on bins created in a decision tree interface. In particular, you can define the bins that will provide the highest predictive power with respect to a given dependent variable. This helper allows you to transform many fields at the same time rather than doing it on a field by field basis. See Helper: Optimal Binning for details. Note that the optimal binning with Weight of Evidence is now represented by a separate node in the Manipulate palette.

Outliers Clipper

Allows you to create a new column by changing the minimum and maximum values in an existing column of numeric data. See Helper: Outliers Clipper for details.

Power Transform

Allows you to create a new column by taking a base to the power of an existing column. See Helper: Power Transform for details.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Standardization

Allows you to standardize (rescale) numeric variables. See Standardization for details.

Substitute Missing Values

Allows you to create a new column by substituting missing values in an existing column with some value, such as the mean. See Helper: Substitute Missing Values for details.

Text join

Allows you to easily concatenate text field values and constants in one or several fields using graphical user interface instead of writing SQL code with CONCAT() or PASTE() functions in Expression Editor. See Helper: Text Join for details.

Text split

Allows you to break a field of String type into several fields. Based on your inputs in its graphical user interface, the helper wizard automatically builds the required expression using the PARSEPARTS() function. See Helper: Text Split for details.

Weight

Allows you to create weight variables. See Helper: Weight for details.

See also:

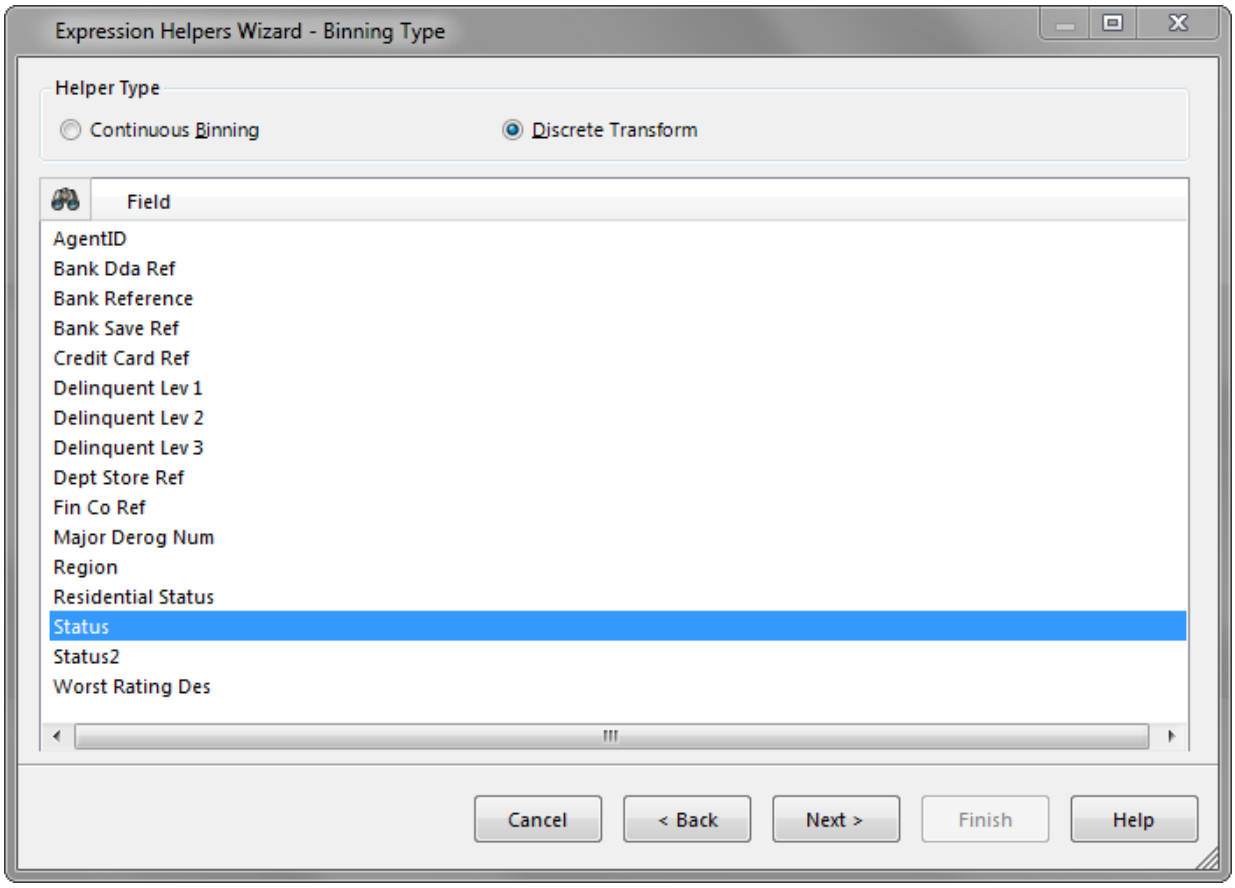
Expression Editor: Overview

Dataset Editor

Helper: Binning

The Binning helper allows you to create a new column that changes the bins of the original variable.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Continuous Binning

Creates a calculated column which is the result of changing the bins of a continuous variable. The resulting new column will be discrete.

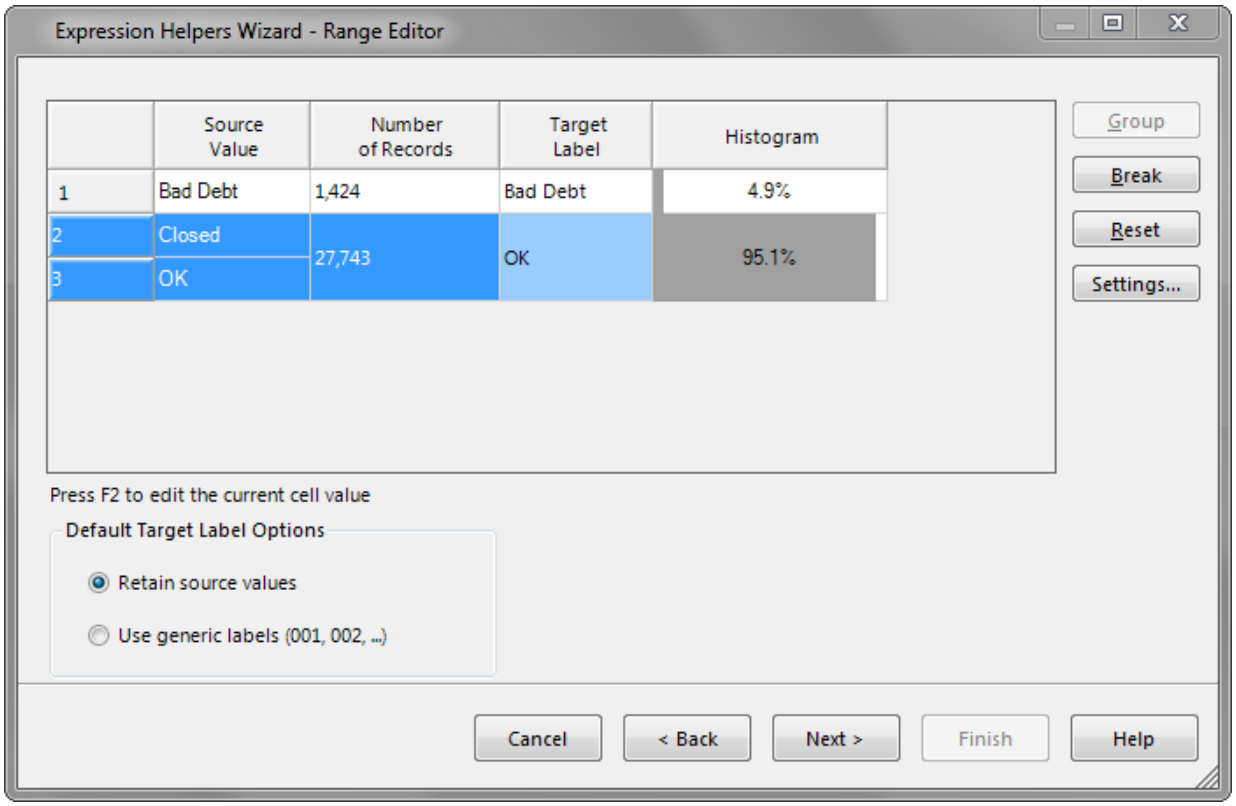
Discrete Transform

Creates a calculated column which is the result of changing the bins of a discrete variable.

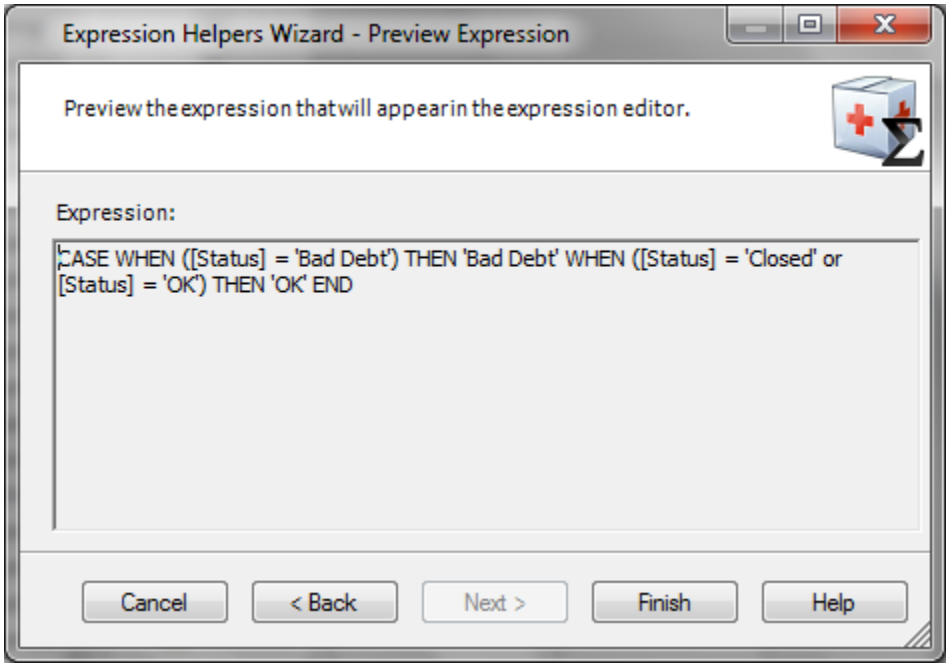
Click Next to edit the bins of the selected variable in the **Range Editor** dialog.

In the screenshot below, the discrete variable *Status* with values 'Bad Debt', 'Closed' and 'OK' is transformed to a new variable with values 'Bad Debt' and 'OK', so that 'Closed' is mapped to 'OK', while the other values remain unchanged. Thus, we grouped two categories into one, assuming the values 'Closed' and 'OK' are equivalent for the purposes of our analysis.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click Next to generate the expression and preview it.



Click Finish to add the expression and return to Expression Editor.

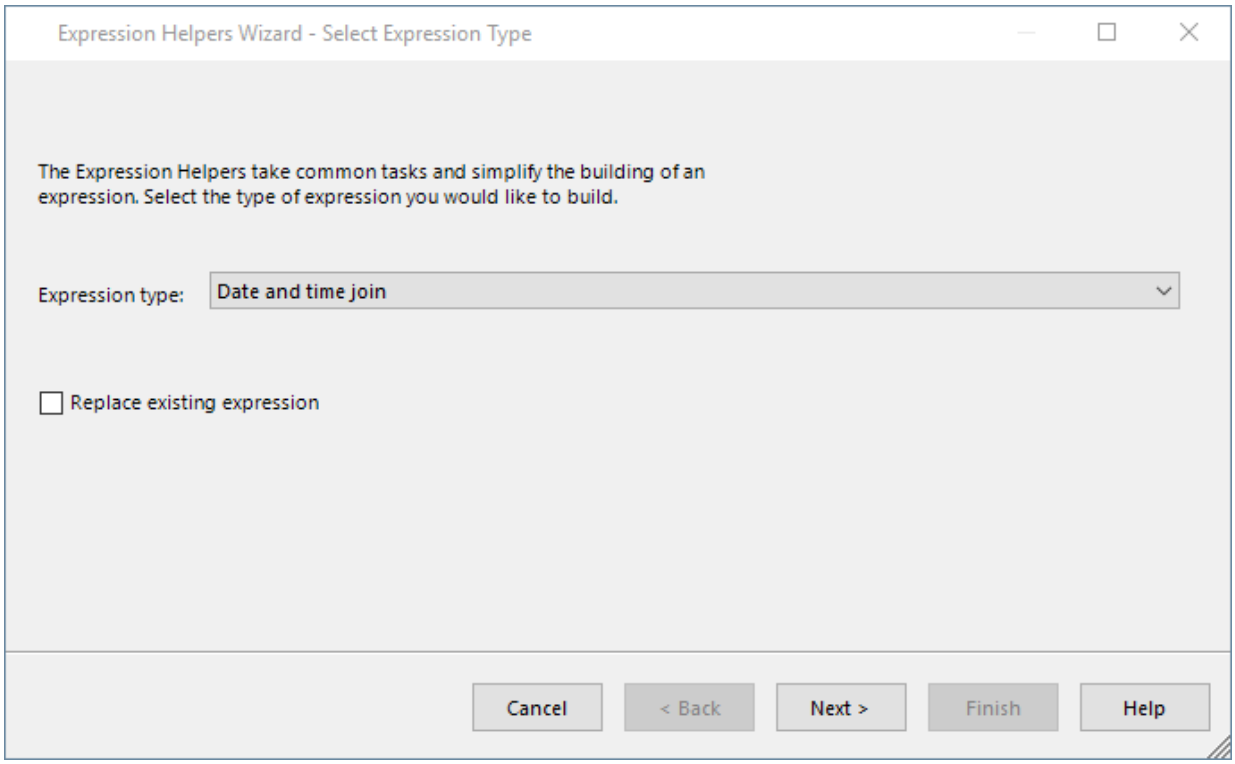
See also:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Helper: Date and Time Join

The **Date and time join** helper allows you to easily join date and time information that is spread across multiple fields into a single Date, Time, or Timestamp field. Based on your inputs in its graphical user interface, the helper wizard automatically builds the required expression using the MKDATE(), MKTIME(), or MKTIMESTAMP() functions in Expression Editor and creates the new field.

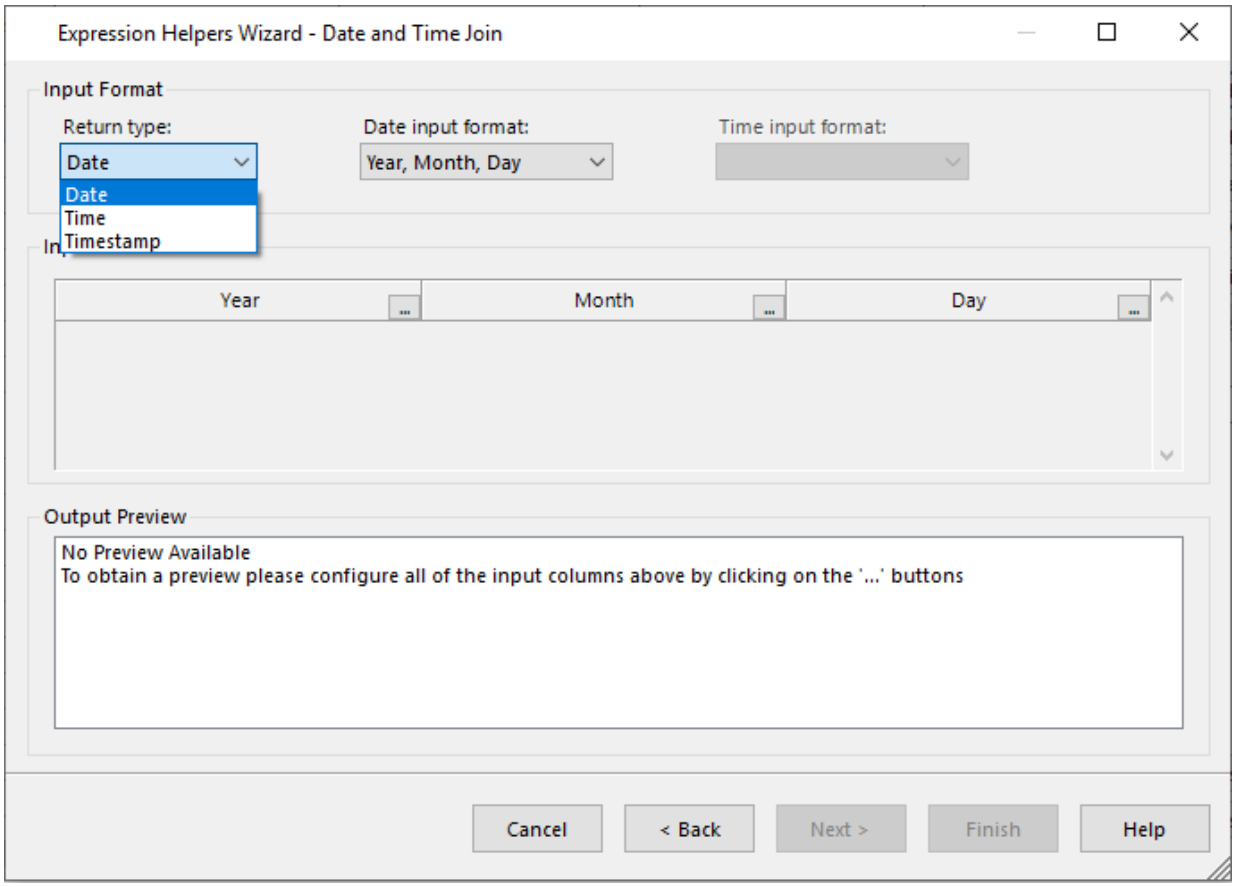
After clicking the Helpers button in the Expression Editor, select **Date and time join** for the Expression type:



Click **Next**.

The next page provides the interface to interactively build an expression for concatenating string, date, and time values while previewing the result in real time.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Specifying Input Format

In the **Input Format** frame, specify how the date and time information is spread across multiple fields in the input data.

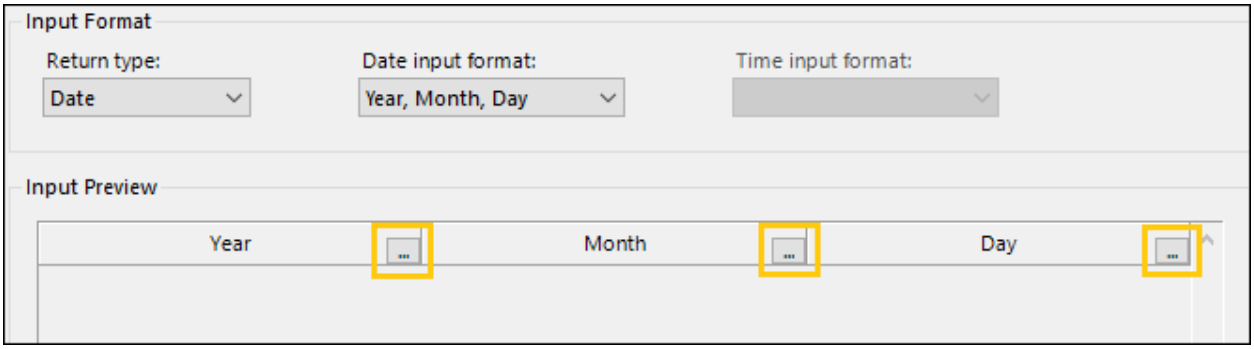
- Select the **Return type** - Date, Time, or Timestamp. This is the data type of the output field.
- The details for the Date and Time input format depend on the selected return type.
    - For the Timestamp return type, specify both Date and Time input formats.
    - If the selected return type is Date (Time) then only the Date (Time) input format must be specified.
    - *Complete Date* or *Complete Time* means that the existing Date or Time information, respectively, is already together in a single field or constant value.
    - A list of Date/Time components (such as *Year, Month, Day*) indicates that the information must be joined from multiple fields and/or constant values.

Input Preview

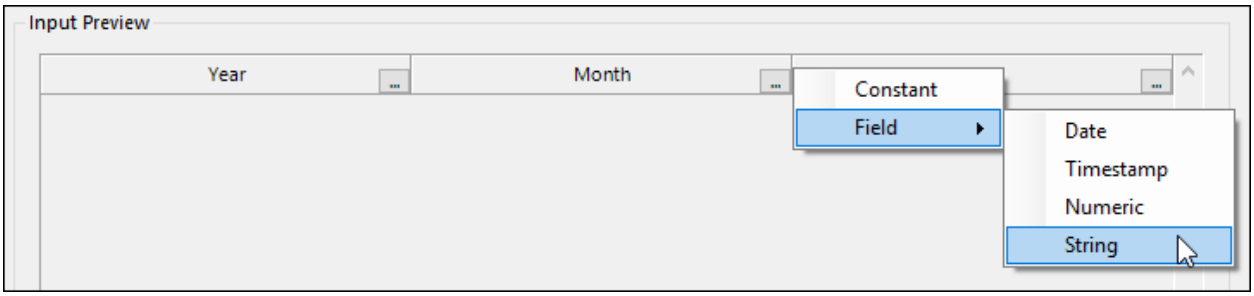
In the **Input Preview** frame, the columns to be joined, as well as the displayed results, depend on the specified input format. For example, if you select the Date return type, and the Date input format "Year, Month, Day", then you need to select either existing fields or specify constants for the Year,

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Month, and Day components. For this purpose, click the small buttons on the right side of each header in the Input Preview grid columns:



The following example show the available options for the Month component (field):

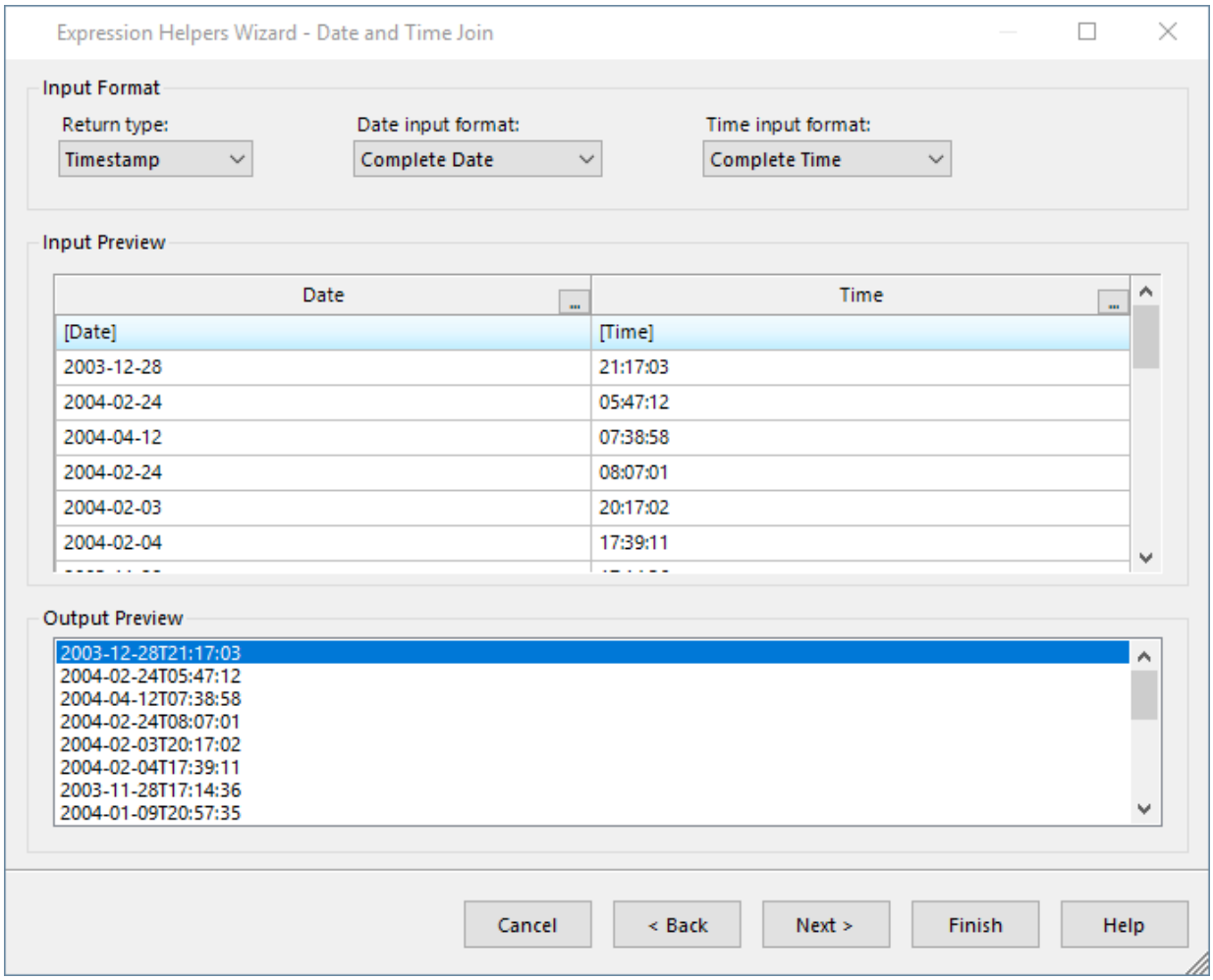


Output Preview

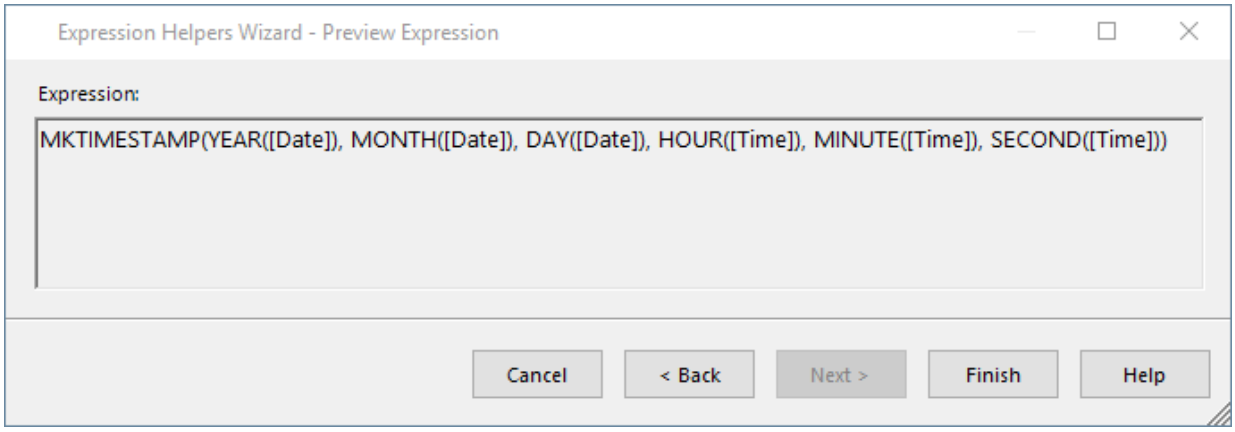
As you build the expression by specifying the input format and the components to join, the **Output Preview** frame shows the value of the joined field to be produced as output. It displays the first 20 records of the output field.

In the example below, we are joining a Date and Time columns (called "Date" and "Time") into a single Timestamp column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

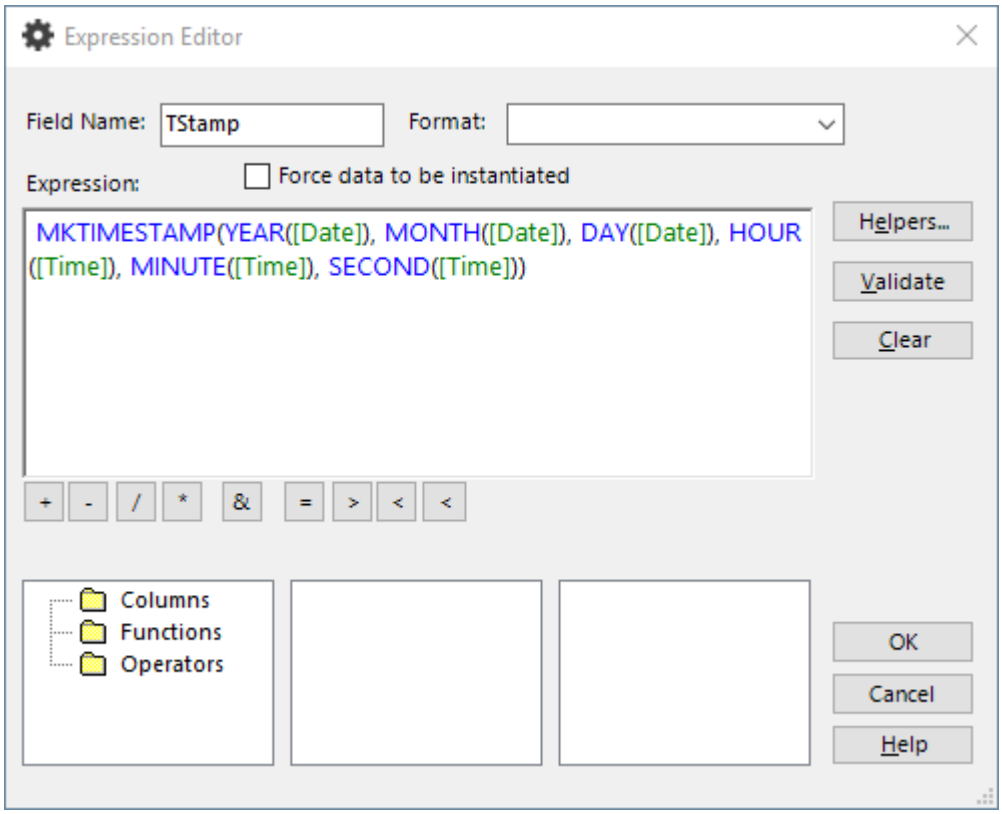


Click **Next** to preview the SQL expression generated by the wizard from your inputs.

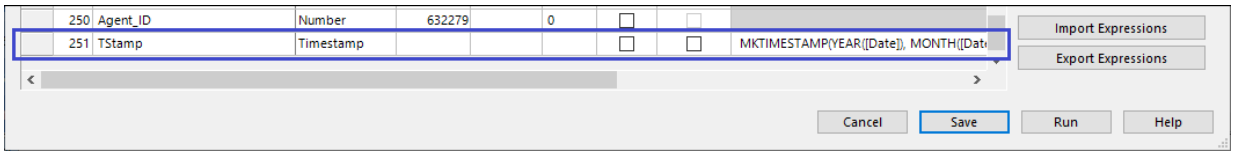


Click **Finish** to generate the expression and view it in the **Expression Editor** dialog, where you can specify the name for the new field and manually edit the resulting expression if necessary.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



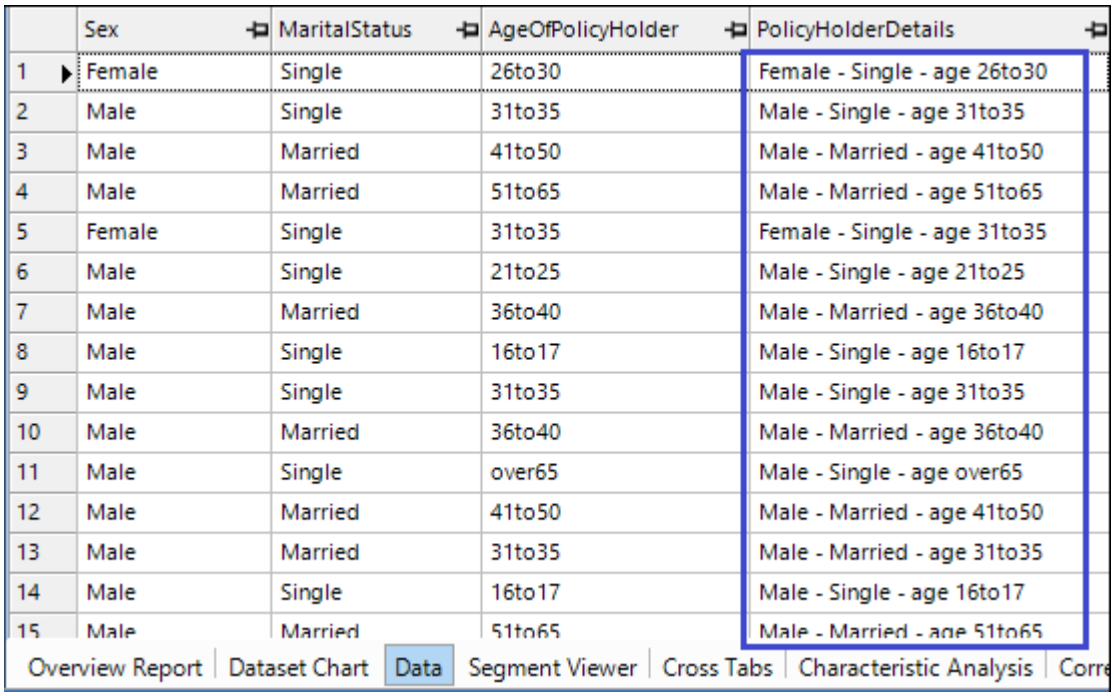
Click OK in the Expression Editor. The generated expression for the new field is shown in the **Dataset Editor** dialog of the Variable Transformations wizard. If you want to edit the expression again, select the new variable and click **Edit Field**. Note that at this stage you will only be able to edit it in the Expression Editor, without invoking the Date and time join helper wizard again.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the node, the resulting dataset node appears on the workflow canvas. Double-click on the output dataset to view the data in the new field.

Example: In the following fragment of the Data tab of the output dataset, the new field is shown on the right, next to the original fields that were used to concatenate their values using the *Date and time join* helper.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



	Sex	MaritalStatus	AgeOfPolicyHolder	PolicyHolderDetails
1	Female	Single	26to30	Female - Single - age 26to30
2	Male	Single	31to35	Male - Single - age 31to35
3	Male	Married	41to50	Male - Married - age 41to50
4	Male	Married	51to65	Male - Married - age 51to65
5	Female	Single	31to35	Female - Single - age 31to35
6	Male	Single	21to25	Male - Single - age 21to25
7	Male	Married	36to40	Male - Married - age 36to40
8	Male	Single	16to17	Male - Single - age 16to17
9	Male	Single	31to35	Male - Single - age 31to35
10	Male	Married	36to40	Male - Married - age 36to40
11	Male	Single	over65	Male - Single - age over65
12	Male	Married	41to50	Male - Married - age 41to50
13	Male	Married	31to35	Male - Married - age 31to35
14	Male	Single	16to17	Male - Single - age 16to17
15	Male	Married	51to65	Male - Married - age 51to65

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Corre

See also:

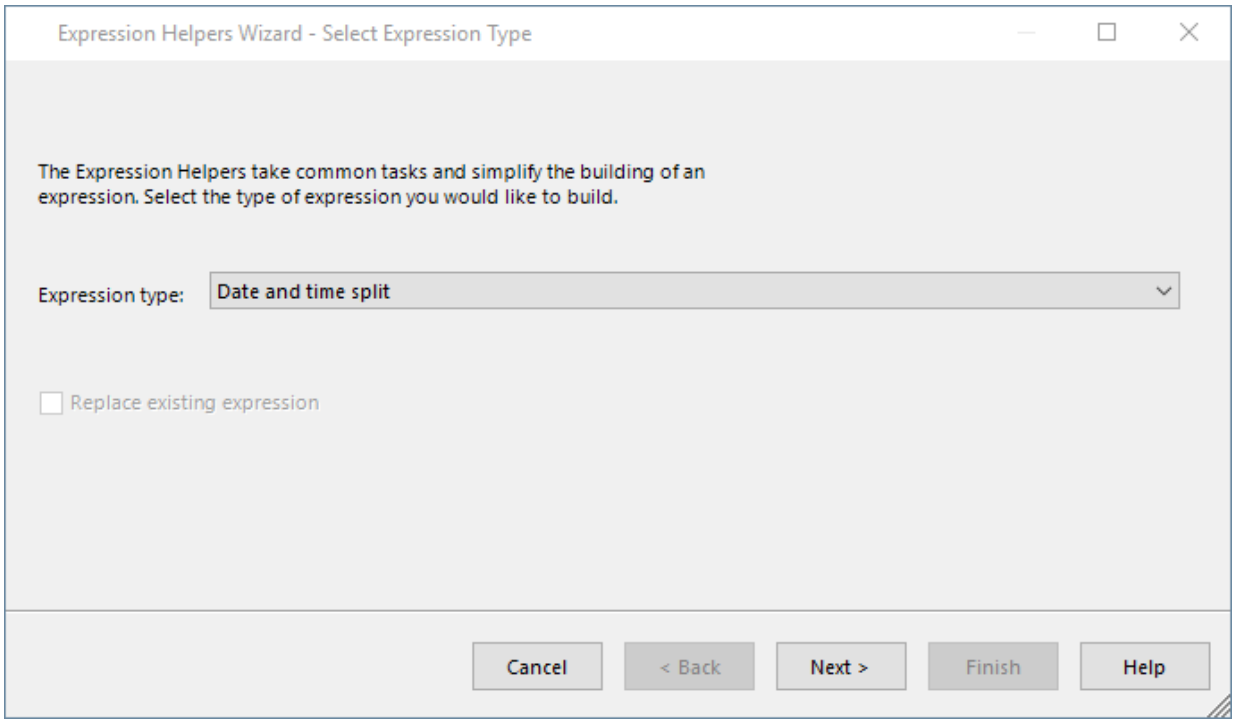
- Helper: Date and Time Split
- Helper: Text Join

Helper: Date and Time Split

The **Date and time split** helper allows you to split a Date, Time, or Timestamp field into several fields representing the date/time components. Based on your inputs in its graphical user interface, the helper wizard automatically builds the required expressions using the functions SECOND(), MINUTE(), HOUR(), WEEKDAY(), YEAR(), and the like, and creates the new fields.

After clicking the Helpers button in the Expression Editor, select **Date and time split** for the Expression type:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

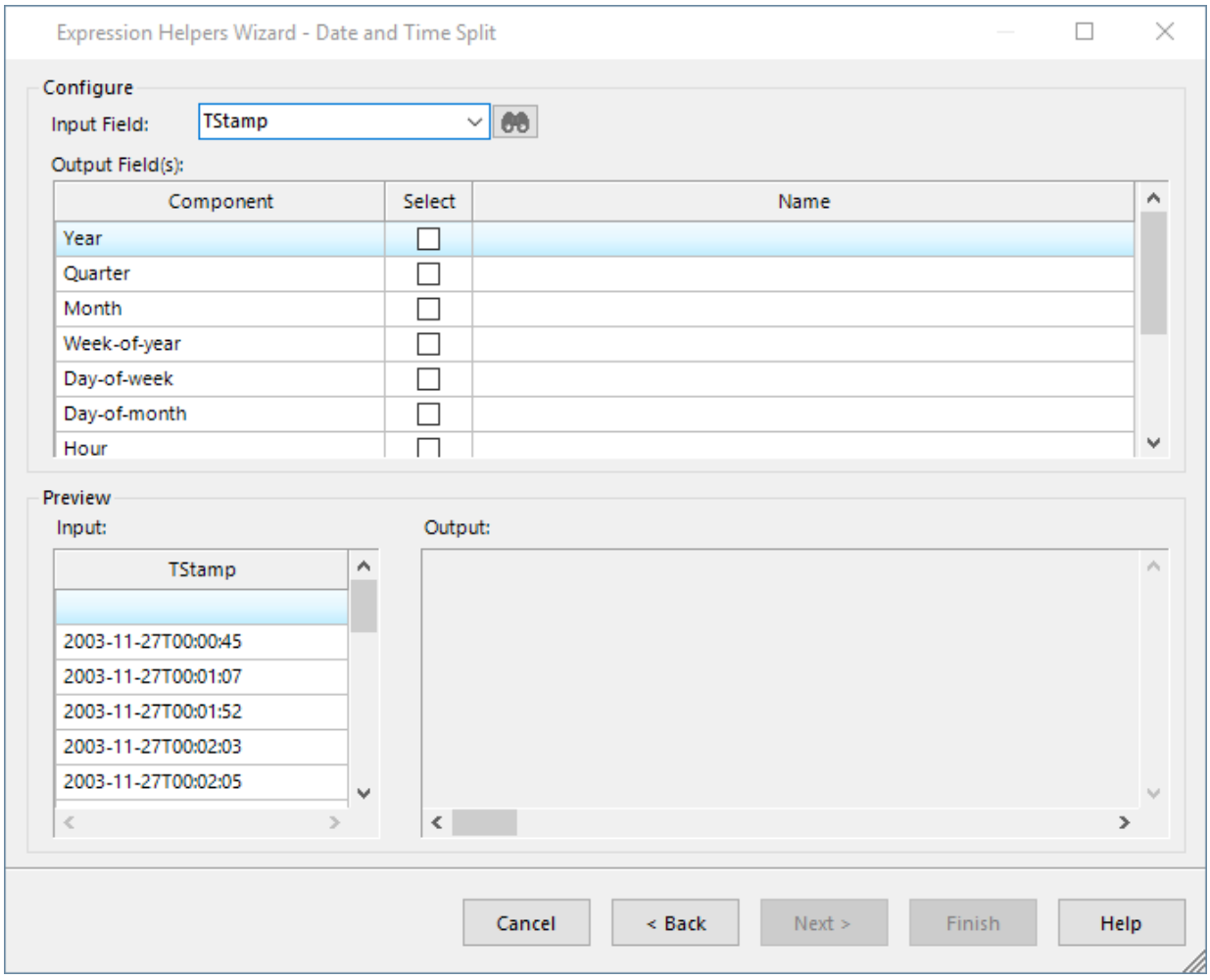


Click **Next**.

The next page of the wizard provides the interface to interactively build an expression for splitting the selected field while previewing the result in real time.

Select the **Input Field** from the drop-down list. The ***Input*** preview frame at the bottom-left is populated with the top 20 values of the selected input field. The Input field should be of type Date, Time, or Timestamp.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Defining the Output Fields

In the **Output Fields** frame:

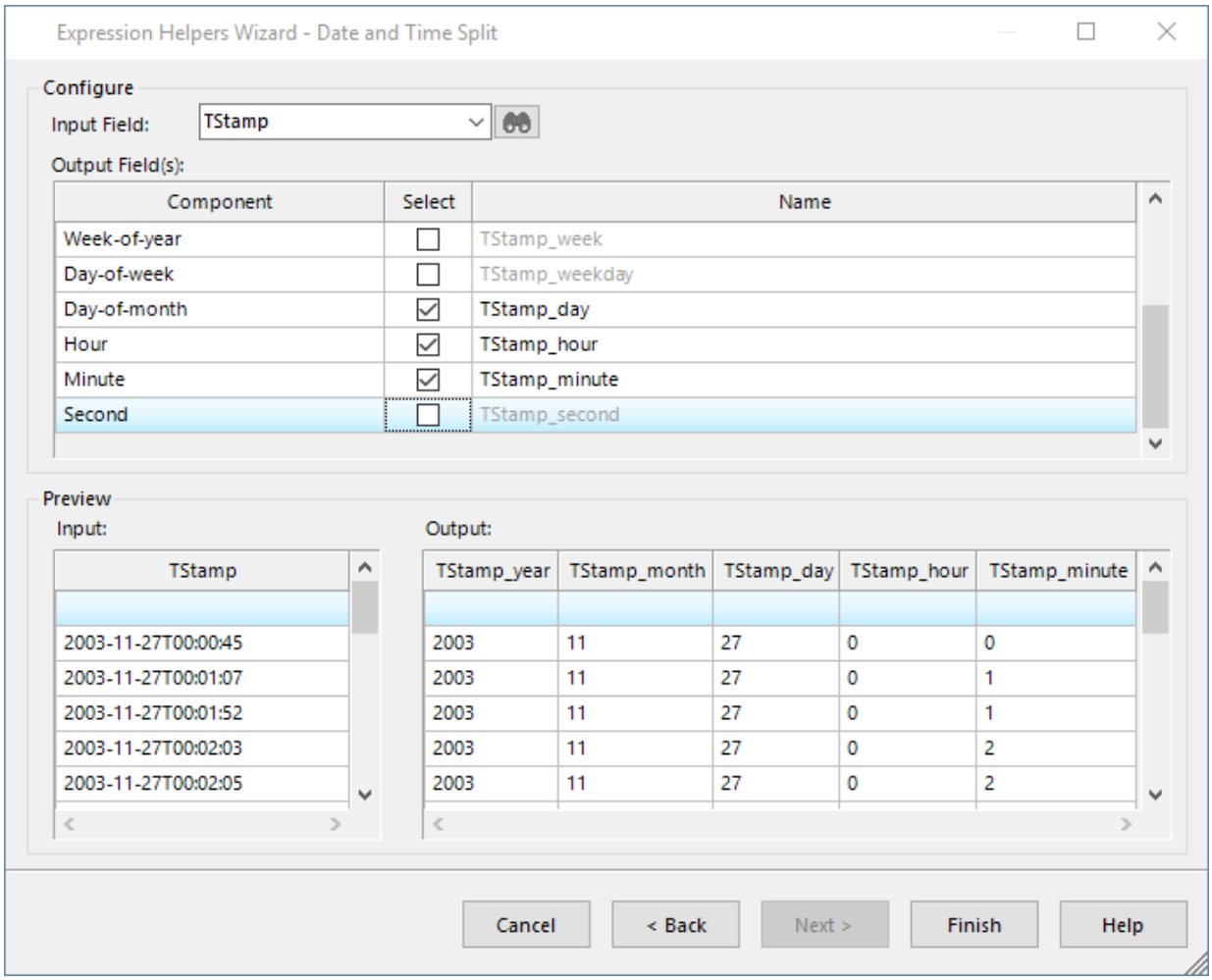
- Specify the required components of the input field that should be extracted and made into separate fields. The available components depend on the type of the input field.
    - For the *Date* input field – Year, Quarter, Month, Week-of-year, Day-of-week, Day-of-month
    - For the *Time* input field – Hour, Minute, Second
    - For the *Timestamp* input field – all of the above
- Specify the names of the new fields which the input field should be split into.

Output Preview

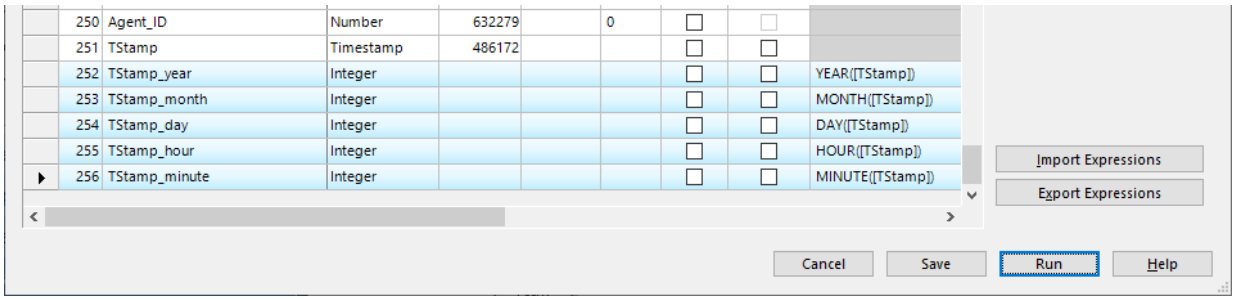
As you build the expression by specifying the delimiters and the output fields, the **Output** preview frame shows the values of the fields to be produced after splitting the input field. It displays the first 20 records of the output fields.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

In the example below, we split the Timestamp input field called "TStamp" into 5 fields: *TStamp_year, TStamp_month, TStamp_day, TStamp_hour,* and *TStamp_minute:*

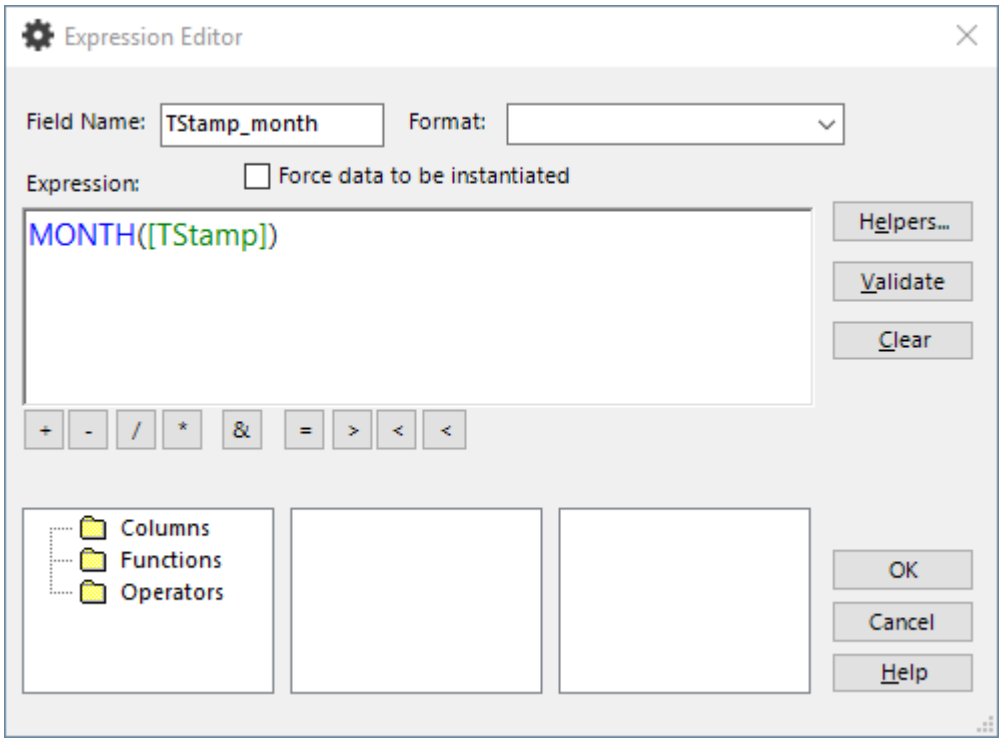


Click **Finish** to generate the SQL expressions for the output fields. The generated expressions for the new fields are shown in the **Dataset Editor** dialog of the Variable Transformations wizard.



If you want to view or edit the expressions, select the desired field and click **Edit Field** to open the **Expression Editor**. Note that at this stage you will only be able to edit the expressions manually in the Expression Editor, not in the *Date and time split* helper wizard. Here is the expression for the *TStamp_month* field from the above example:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** in the Dataset Editor window to save the settings, or click **Run** to save the settings and run the process immediately. After running the node, the resulting dataset node appears on the workflow canvas. Double-click on the output dataset to view the data in the new field.

Example: In the following fragment of the Data tab of the output dataset, the new fields are shown on the right, next to the original field that was split using the *Date and time split* helper.

	TStamp	TStamp_year	TStamp_month	TStamp_day	TStamp_hour	TStamp_minute
1	2003-12-28 09:17 PM	2003	12	28	21	17
2	2004-2-24 05:47 AM	2004	2	24	5	47
3	2004-4-12 07:38 AM	2004	4	12	7	38
4	2004-2-24 08:07 AM	2004	2	24	8	7
5	2004-2-03 08:17 PM	2004	2	3	20	17
6	2004-2-04 05:39 PM	2004	2	4	17	39
7	2003-11-28 05:14 PM	2003	11	28	17	14
8	2004-1-09 08:57 PM	2004	1	9	20	57
9	2004-2-24 11:34 AM	2004	2	24	11	34
10	2004-3-16 04:37 PM	2004	3	16	16	37
11	2003-12-17 06:57 PM	2003	12	17	18	57
12	2004-3-29 11:52 PM	2004	3	29	23	52
13	2004-3-17 02:01 AM	2004	3	17	2	1
14	2004-4-15 07:04 PM	2004	4	15	19	4
15	2003-12-08 08:29 AM	2003	12	8	8	29

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

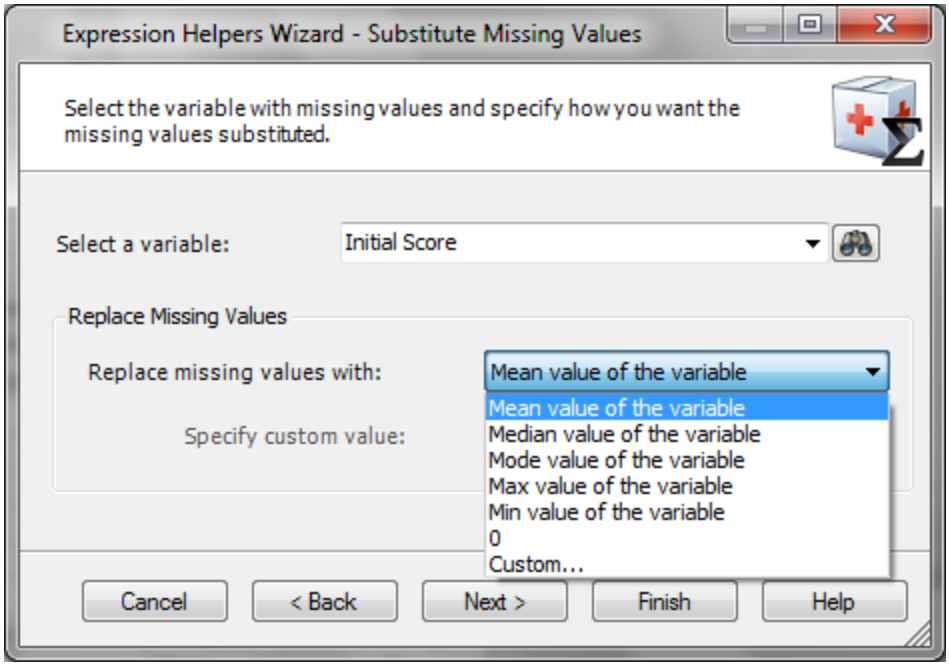
See also:

[Helper: Date and Time Join]

[Helper: Text Join]

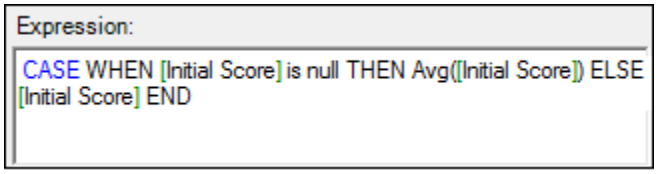
Helper: Substitute Missing Values

This helper allows you to create a new column by substituting a specific value for all missing values in a column. For example, you can substitute the mean or maximum value of the same column, or use a custom value. The available options are shown below:



Click **Next** to preview the resulting expression. Then click **Finish** to add it to the Expression Editor window.

An example of the resulting expression:



See also: [Substitute Missing Values]

Helper: Generation of Dummy Variables

This helper allows you to create a set of dummy, or indicator variables from a discrete variable. Based on a single discrete column, it creates a series of binary variables with values 0 and 1.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

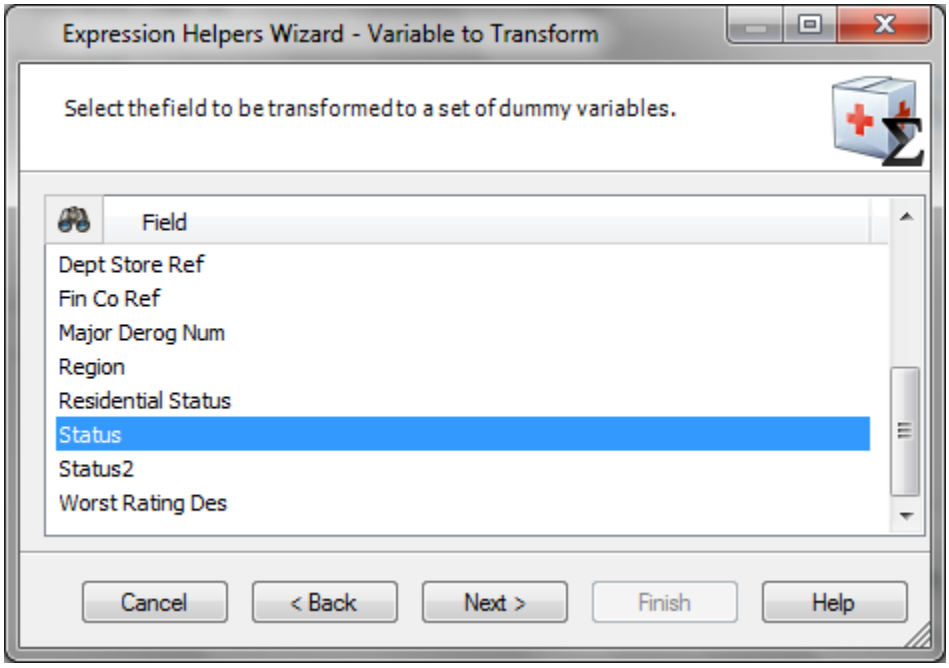
For example, if a variable called *Status* in a credit account dataset has values "OK", "Bad Debt", and "Closed", the helper transforms it into the set of three dummy variables with values 0 and 1 indicating the value taken by the original *Status* variable.

Status	***Status_OK***	***Status_Bad Debt***	***Status_Closed***
OK	1	0	0
Bad Debt	0	1	0
Closed	0	0	1

To generate dummy variables, click the Helper button in the Expression Editor and select *Generation of Dummy Variables* as Expression type. Click Next.

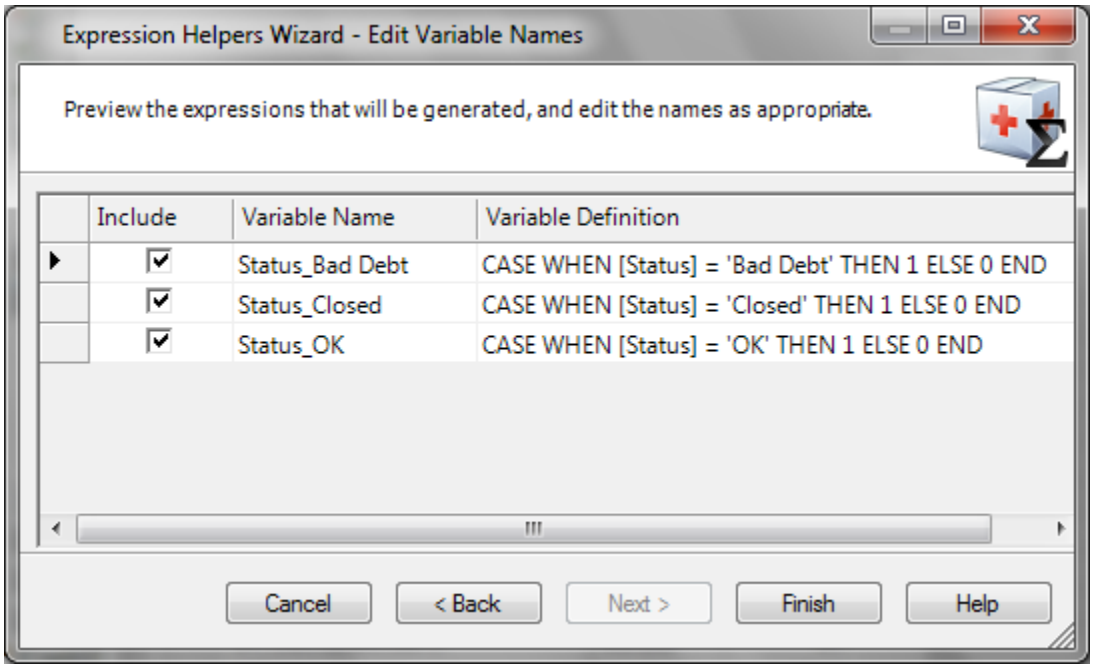
In the following page, select the field you wish to transform into a set of dummy variables. Available fields include:

- all String fields
- non-String fields whose cardinality does not exceed 10



Click Next. The following step allows you to rename the dummy variables and exclude some of them if necessary. The default names are derived from the original variable name by appending an underscore and the respective values.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click Finish. The new dummy variables will be added to the Dataset Editor. Click OK in the Dataset Editor dialog to add the variables to the dataset.

If necessary, you can edit the expressions for the new variables by clicking the Edit Field button in Dataset Editor.

Note: As a part of the automatic normalization of variables in predictive model and cluster model training, all String variables are implicitly transformed into a set of dummy variables. The helper allows you to do this explicitly for all types of discrete variables, outside of the model creation process.

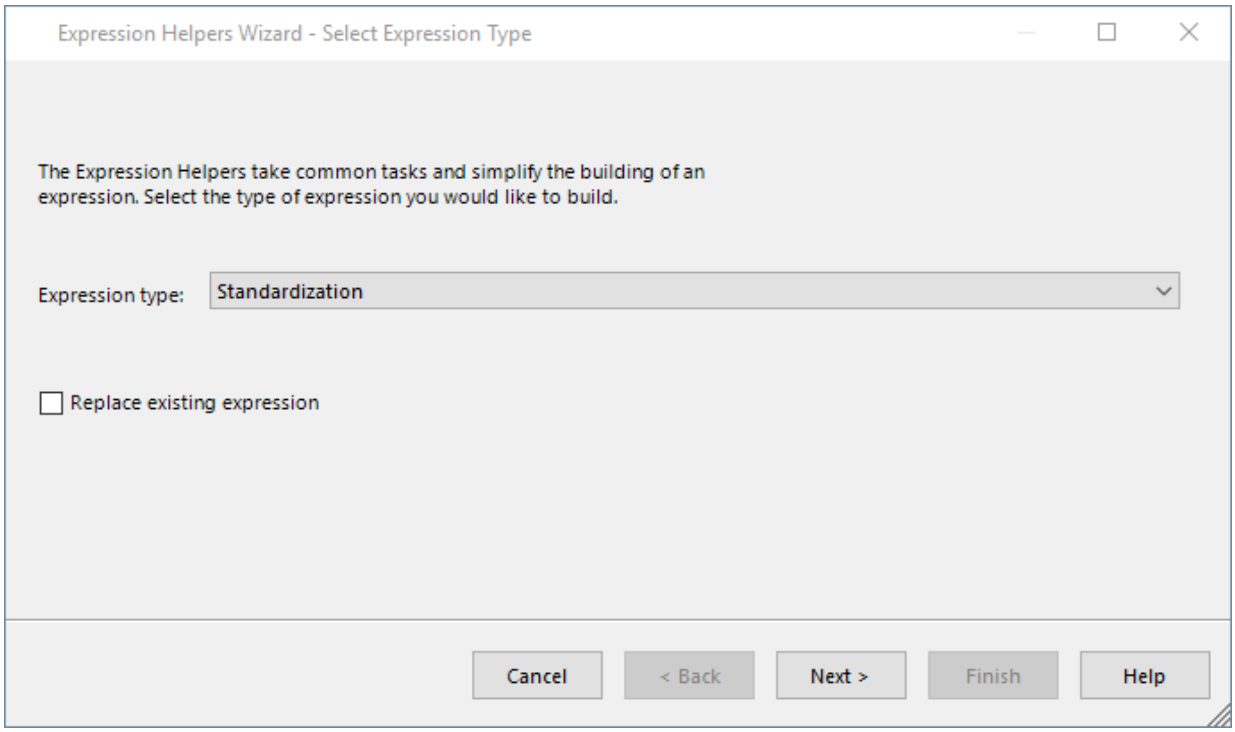
See also: Expression Helpers, Function Syntax, Binning Helper.

Helper: Standardization

The **Standardization** helper allows you to standardize (rescale) numeric variables. One of the most common standardization types is the transformation that produces variables with a mean value of 0 and a standard deviation of 1. The helper allows you to standardize multiple variables at a time.

After clicking the Helpers button in the Expression Editor, select **Standardization** for the Expression type:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



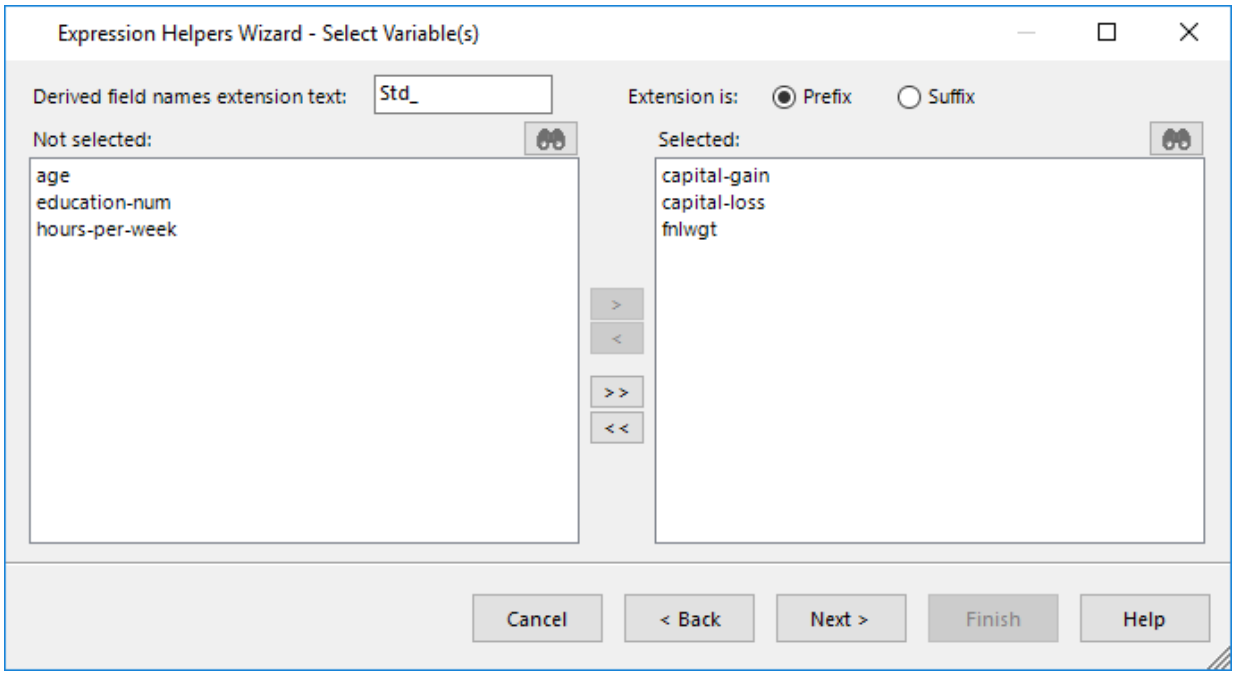
Click Next.

In the next page:

- select the variables to standardize (only numeric variables are available for selection);
- select the type of the extension text (prefix or suffix) to be added to the original field names;
- specify the extension text to be added to the original field names; the default prefix is std_ and suffix is _std.

If the dataset has hundreds or thousands of variables, use the binoculars button to open the Field Finder where you can easily find and select the desired variables.

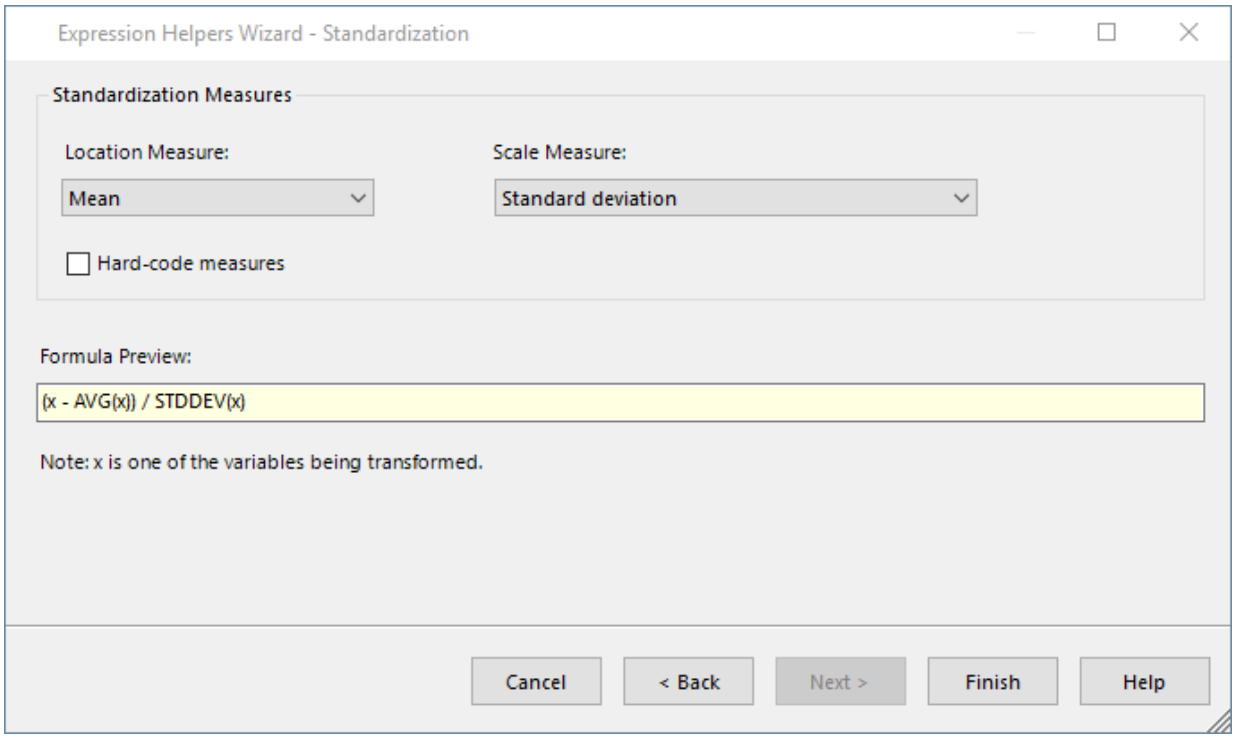
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click Next.

In the next page, select the desired **Location Measure** and **Scale Measure** to generate the standardization formula. As soon as the measures are selected, the Formula Preview field shows the coresponding expression to be applied to all variables selected at the previous step. The formula is (x - selected location measure) / (selected scale measure), where x stands for any of the variables selected at the previous step. For example, if you selected Mean and Standard deviation, then the formula is (x - AVG(x))/STDDEV(x).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The following Location Measures and Scaling Measures are available:

Location Measures:

- Mean
- Median: The middle value of the ordered set of values in the field across all records, i.e., the value such that an equal number of values are less than and greater than this value (if the cardinality is odd), or the lesser of the two central values (if the cardinality is even).
- Mode: The most common value in the field
- Min
- Max
- Zero

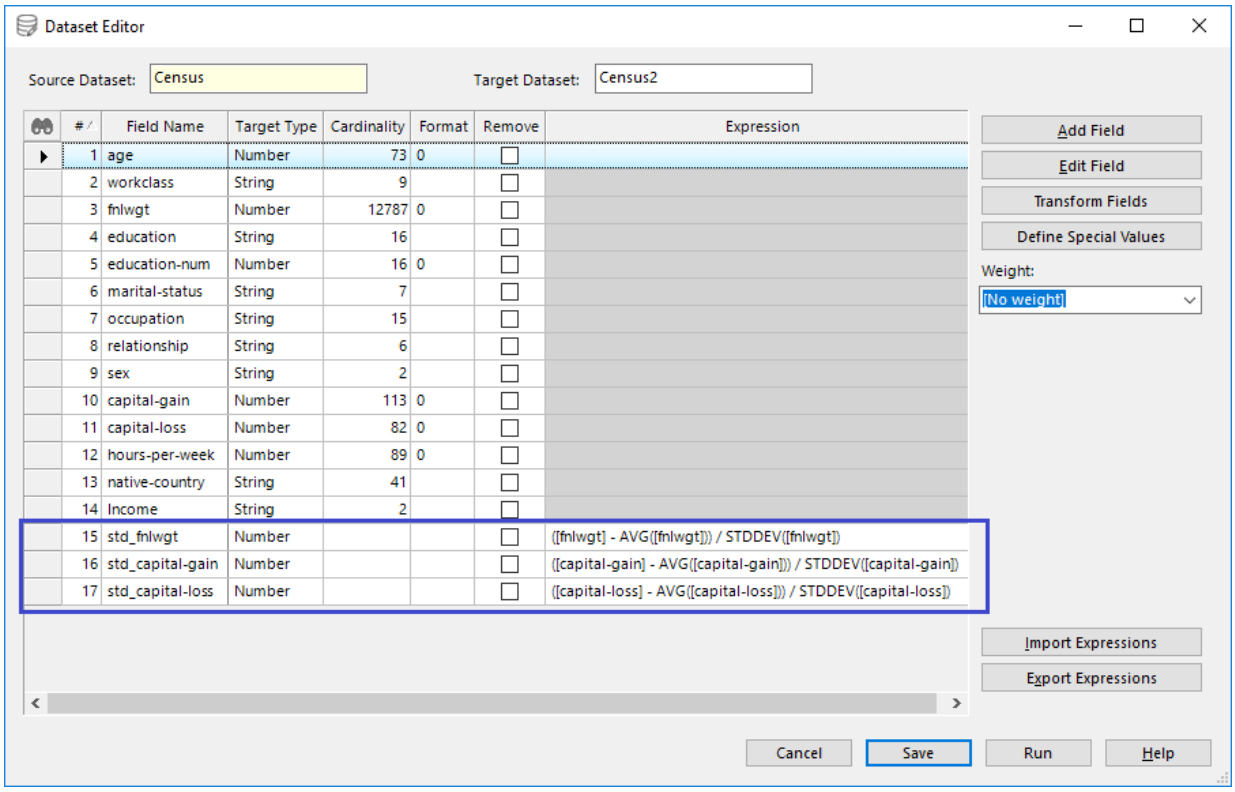
Scale Measures:

- Standard deviation
- Two times the standard deviation: 2 * StdDev
- Data range: The difference between the maximum and the minimum value in the field
- Interquartile range: The difference between the upper quartile and the lower quartile (the 75th and 25th percentiles)
- Median absolute deviation: The median of the absolute deviations from the median of the field.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Select the option **Hard-code measure** if you want the values of the location measure and scale measure to be calculated immediately and pasted in the formula as values. For example, if you chose "Mean" for location measure and "Standard deviation" for scale measure, selecting the *Hard-code measure* option will cause the AVG and STDDEV in the (x - AVG(x)) / STDDEV(x) formula to be calculated for each selected variable x, and the resulting numeric values will be substituted in the expression to be added to the Expression Editor. Example: (x - 38.77) / 13.85. This is useful if you need to export expressions and use them later for a different dataset, but you prefer to use the values calculated on the original dataset rather than calculate the measures on the new data.

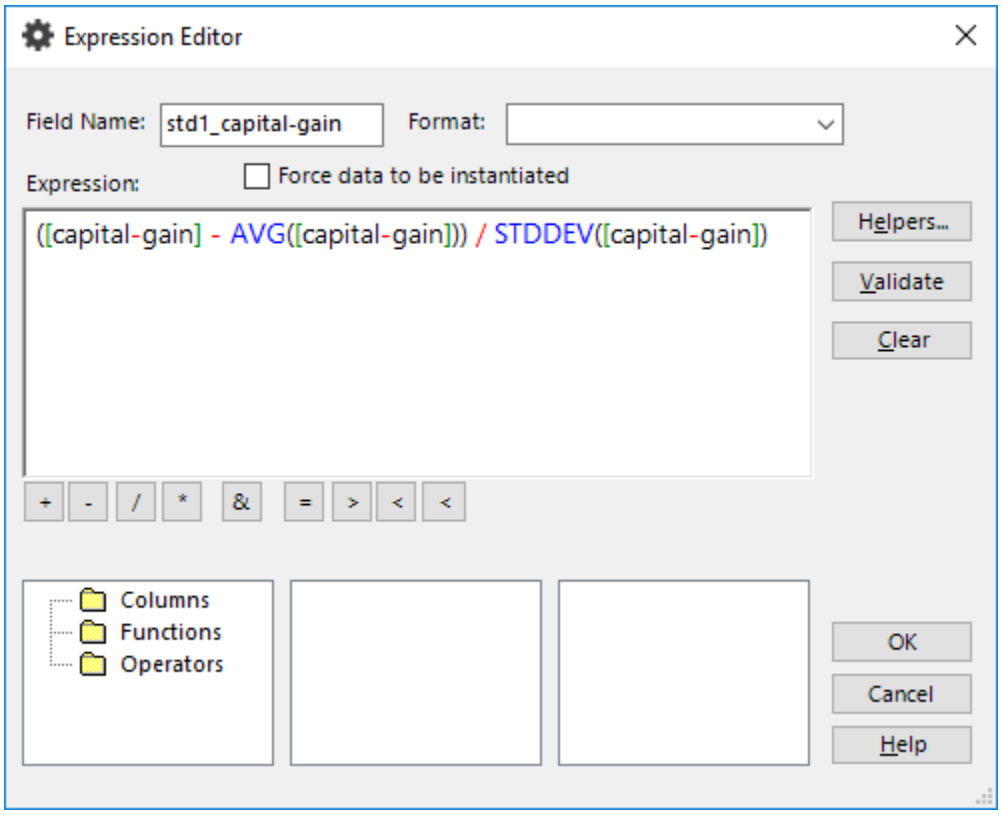
Click Finish to generate the standardization expressions for the selected variables. If you selected more than one variable, the generated expressions for the new fields to be created are shown in the Dataset Editor dialog. If you want to edit any of these expressions, select the desired variable and click **Edit Field**.



#	Field Name	Target Type	Cardinality	Format	Remove	Expression
1	age	Number	73	0	☐	
2	workclass	String	9		☐	
3	fnlwgt	Number	12787	0	☐	
4	education	String	16		☐	
5	education-num	Number	16	0	☐	
6	marital-status	String	7		☐	
7	occupation	String	15		☐	
8	relationship	String	6		☐	
9	sex	String	2		☐	
10	capital-gain	Number	113	0	☐	
11	capital-loss	Number	82	0	☐	
12	hours-per-week	Number	89	0	☐	
13	native-country	String	41		☐	
14	Income	String	2		☐	
15	std_fnlwgt	Number			☐	([fnlwgt] - AVG([fnlwgt])) / STDDEV([fnlwgt])
16	std_capital-gain	Number			☐	([capital-gain] - AVG([capital-gain])) / STDDEV([capital-gain])
17	std_capital-loss	Number			☐	([capital-loss] - AVG([capital-loss])) / STDDEV([capital-loss])

If you selected a single variable, then the generated expression is shown in the Expression Preview and then the Expression Editor dialog.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



In this case, edit the expression if required and then click OK to add the expression to the Dataset Editor.

Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the operation, the resulting dataset node appears on the workflow canvas.

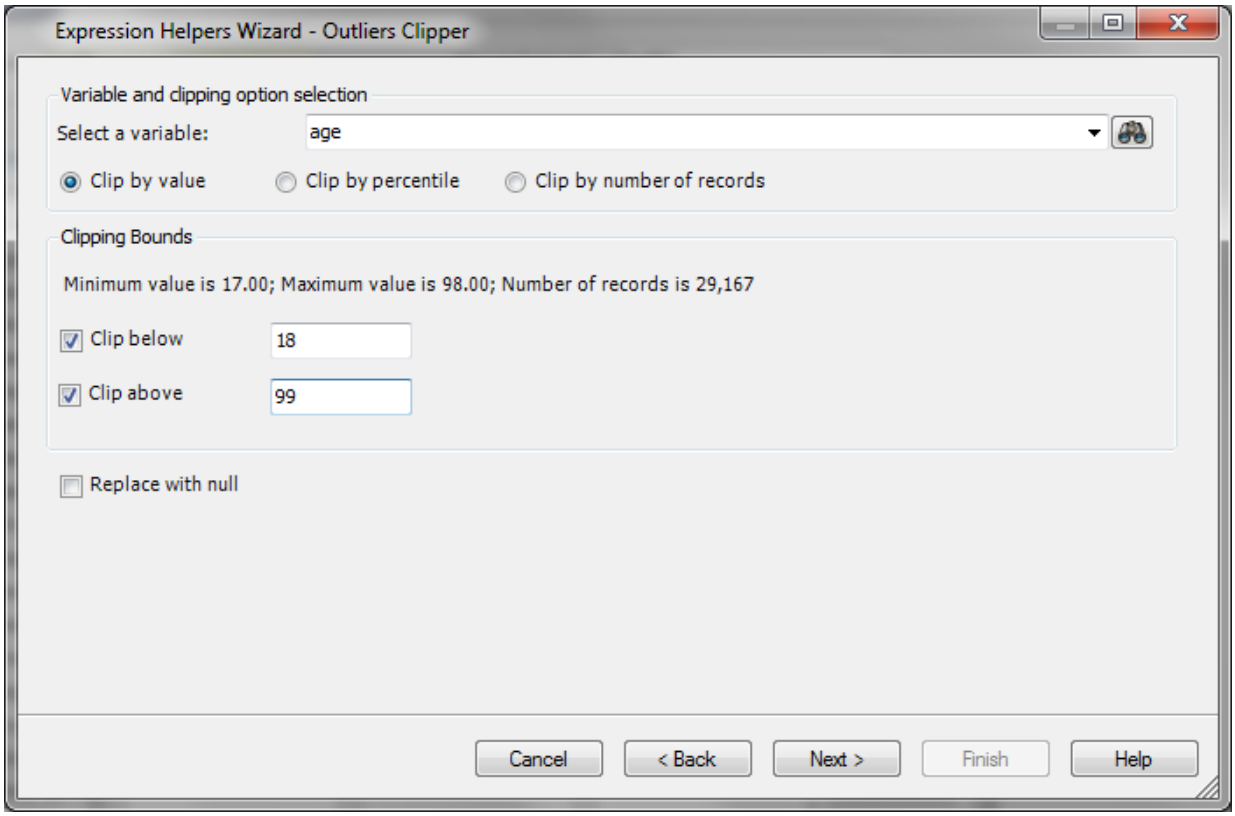
Helper: Outliers Clipper

The Outliers Clipper Helper allows you to handle extreme values in your data. It helps you create a new column by changing the minimum and maximum values in an existing numeric column.

Clipping type options:

- Clip by value (default): Replaces the lower and upper extreme values with the specified lower and upper boundaries, respectively.
- Clip by percentile: If you enter numbers N and M for above and below, respectively, then the top N and the bottom M percent of the records will be clipped. The outliers will be replaced by the next highest/lowest unclipped value unless the option "*Replace with null*" is chosen.
- Clip by number of records: If you enter numbers N and M for above and below, respectively, then the top N and bottom M records will be dropped. The outlier values will be replaced by the value of the next smallest/biggest (unclipped) record.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next** to preview the expression constructed by the wizard, or click **Finish** to add the expression to the Expression Editor.

In the example for the *age* variable shown above, with the 'Clip by value' option, the resulting expression is

```
CASE  WHEN [age] < 18 THEN 18 WHEN [age] > 99 THEN 65 ELSE [age] END
```

This transformation is also known as *winsorizing*. If you would rather do *trimming* (i.e., discard the extreme values by replacing them with a null value), just edit the resulting expression by specifying ***null*** after THEN:

```
CASE  WHEN [age] < 18 THEN null WHEN [age] > 99 THEN null ELSE [age] END
```

Select the option "*Replace with null*" if you need to substitute null for the outlier values instead of replacing the outliers with the cutoff values.

See Also:

Outlier Detector

Helper: Weight

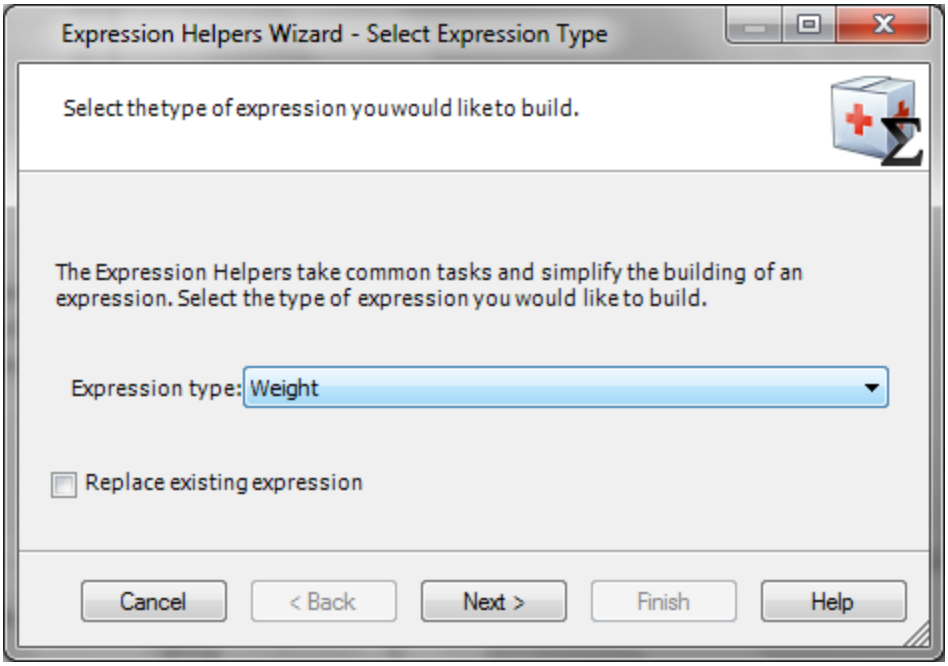
The purpose of the Weight Helper in the Dataset Editor is to provide a simple and convenient way to add a weight as a calculated column.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The Weight Helper is especially convenient for adding probability (sample) weights, which represent the portion of specific observations selected into the sample from a population. For example, if the population of the customers who responded positively to your marketing campaign is 50000 and you have selected 20000 of them into your sample when you imported a part of the data, your sampling fraction is 2/5. Then the value of the weight for such customers is the inverse of the sampling fraction = 5/2 = 2.5

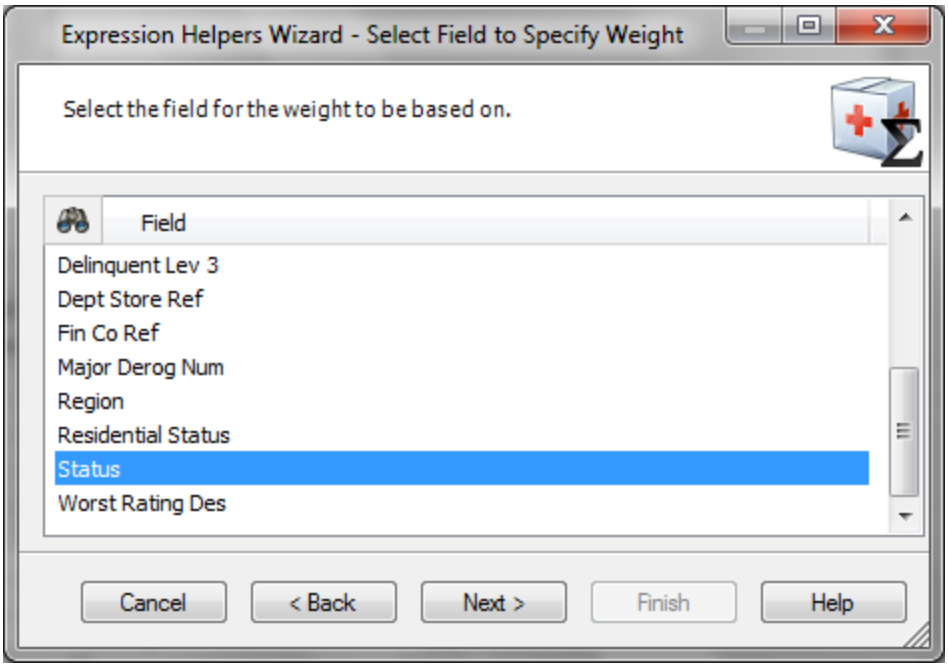
For more information about weights in datasets, see the Weight topic.

Adding a weight based on the values of a target field (similar to inserting a stratified partition) is done in the Variable Transformations Node wizard by clicking the **Add** button in the Dataset Editor dialog and selecting “Weight” for the Expression Type:

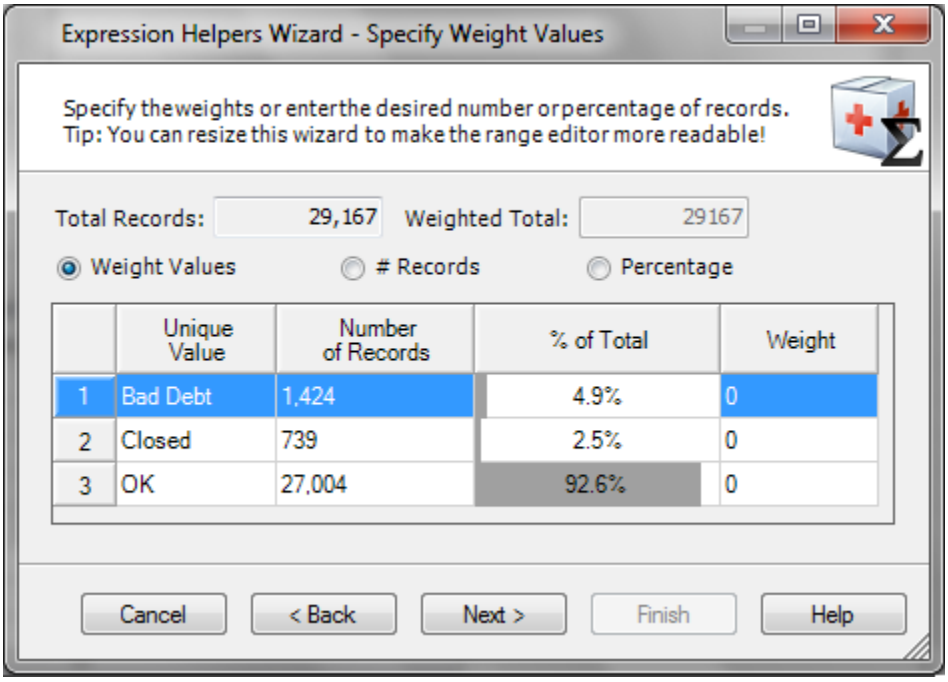


In the next page, select the target field that the weight should be based on.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

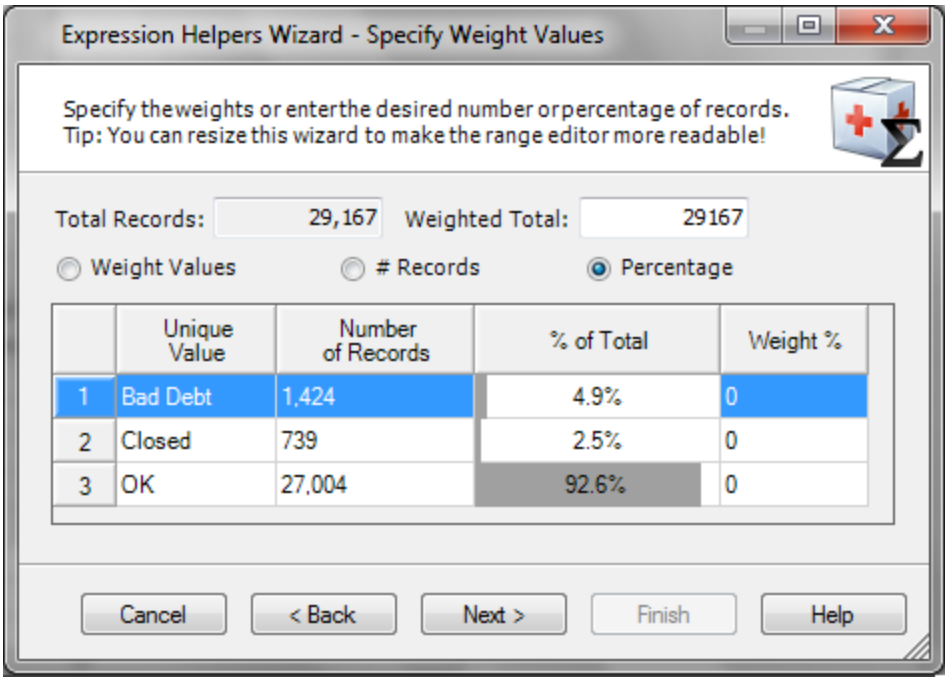


In the next page, you can define the weight in either of the following ways:



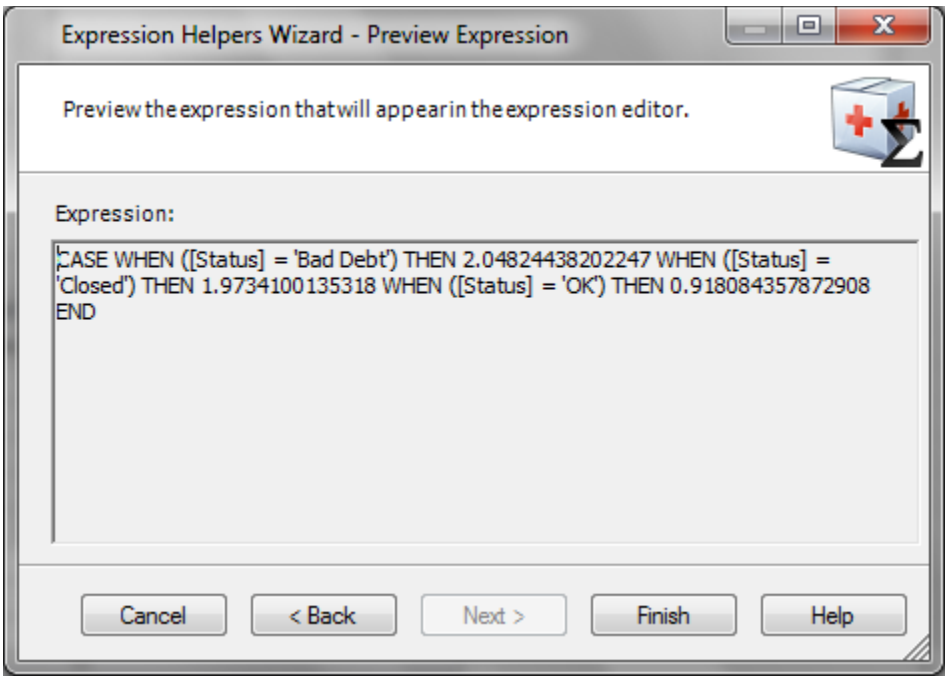
- Enter the weights for each value of the target variable in the Weight field.
- Enter the desired frequencies (the number of records requested for each value). If you choose this option, the last column header changes to “# Records”.
- Enter the desired distribution (percentages). If you choose this option, the last column header changes to “Weighted %” and the Weighted Total field becomes editable:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



NOTE: The sum of the values entered in the Weighted % field are interpreted as portions of the number in the Weighted Total field. They may not necessarily add up to 100%. If you click Next while the sum is not equal to 100%, then the warning message will appear: “The sum of the percentages entered does not equal 100%. The actual weighted total will differ from the number specified in the Weighted Total field." You can choose to proceed anyway, or change the percentages entered.

The next step allows you to preview the expression automatically generated based on your inputs. The expression defines the values of the new weight column.



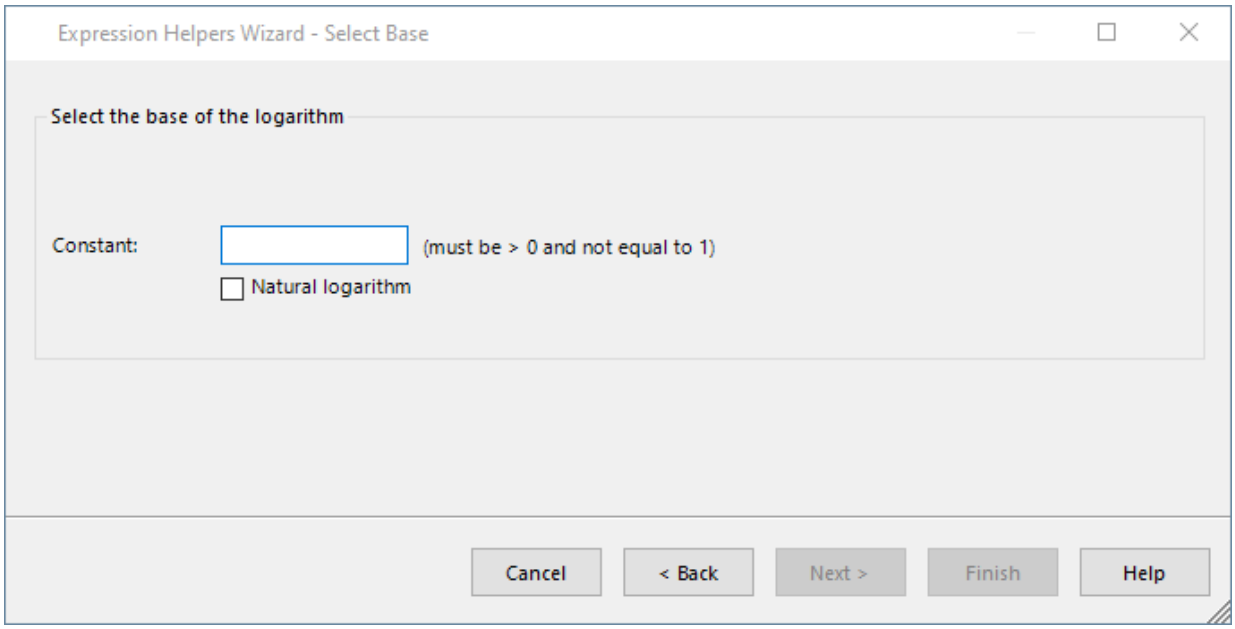
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Click Finish to accept and return to the Dataset Editor window and then click OK to add the column.

Note that a calculated column added using the Weight Helper does not automatically become a weight (i.e., is not automatically set as weight in Overview Report). You need to explicitly set it as weight in Overview Report if necessary. See the Weight topic for details on setting weight.

Helper: Log Transform

The Logarithm transform helper allows you to add a new column that is the logarithm of a numeric column, where the base is user-specified.



The first page of the helper prompts for the base of the logarithm. The base must be a positive number that is not equal to 1.

Natural logarithm is based on the constant ***e*** (= approx. 2.71828182845904).

Click *Next* to select the numeric column for which you want the logarithm. For the logarithm to be defined, the values in the selected column must be positive real numbers. For any nonpositive value, the resulting value in the calculated column will be *null*.

Example:

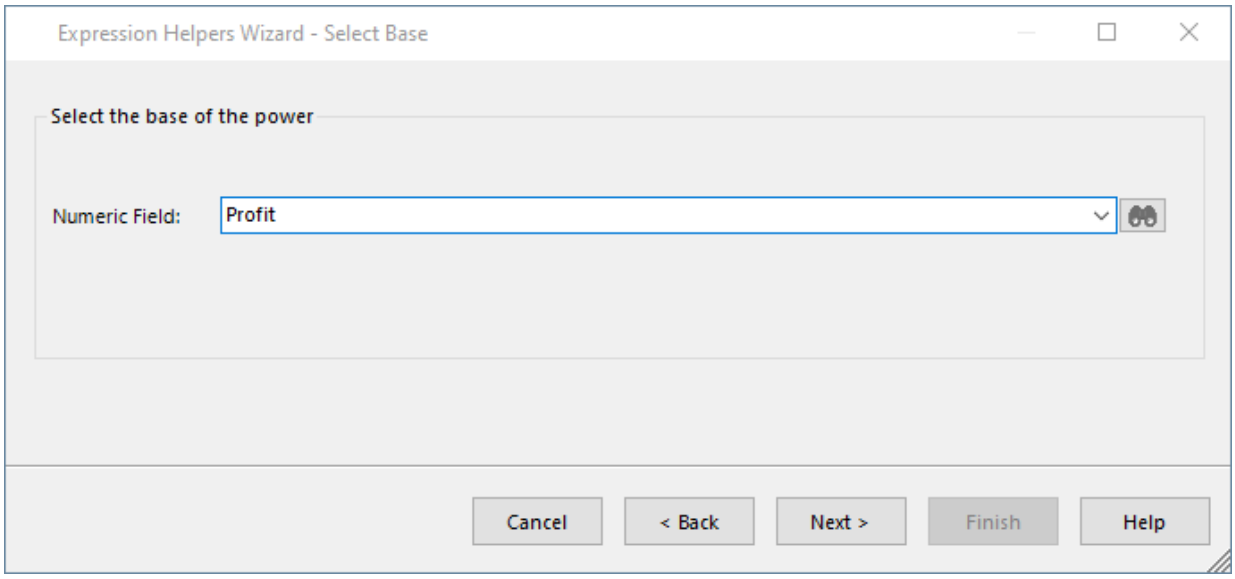
> If the base is 10 and the selected column is Profit, the resulting expression will be *log*([Profit]) / *log*(10), where *log* with no base specified stands for the natural logarithm.

Helper: Power Transform

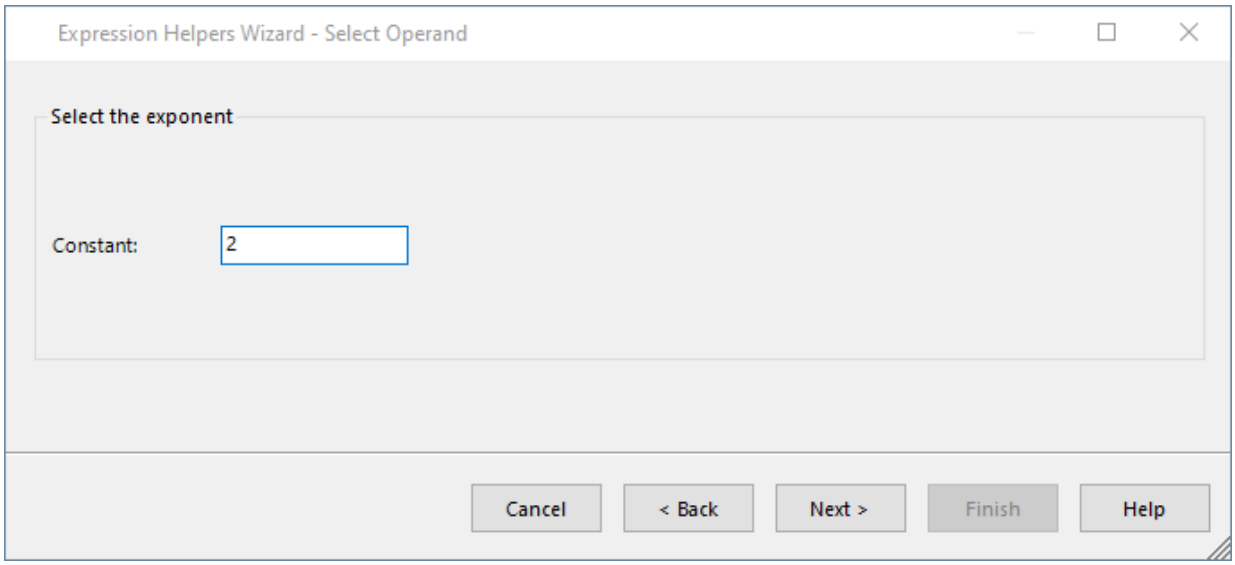
The Power transform helper allows you to add a column whose values are equal to the values of a selected numeric column raised to the power you specify.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The first page of the helper prompts for the numeric column to be used as the base of the power. The values in this column can be any real numbers.



The second page prompts for the exponent to which the base is raised. It can be any real number.



Example:

If the column selected as the base is Profit and the exponent is 2, the resulting expression will be `pow([Profit], 2)`.

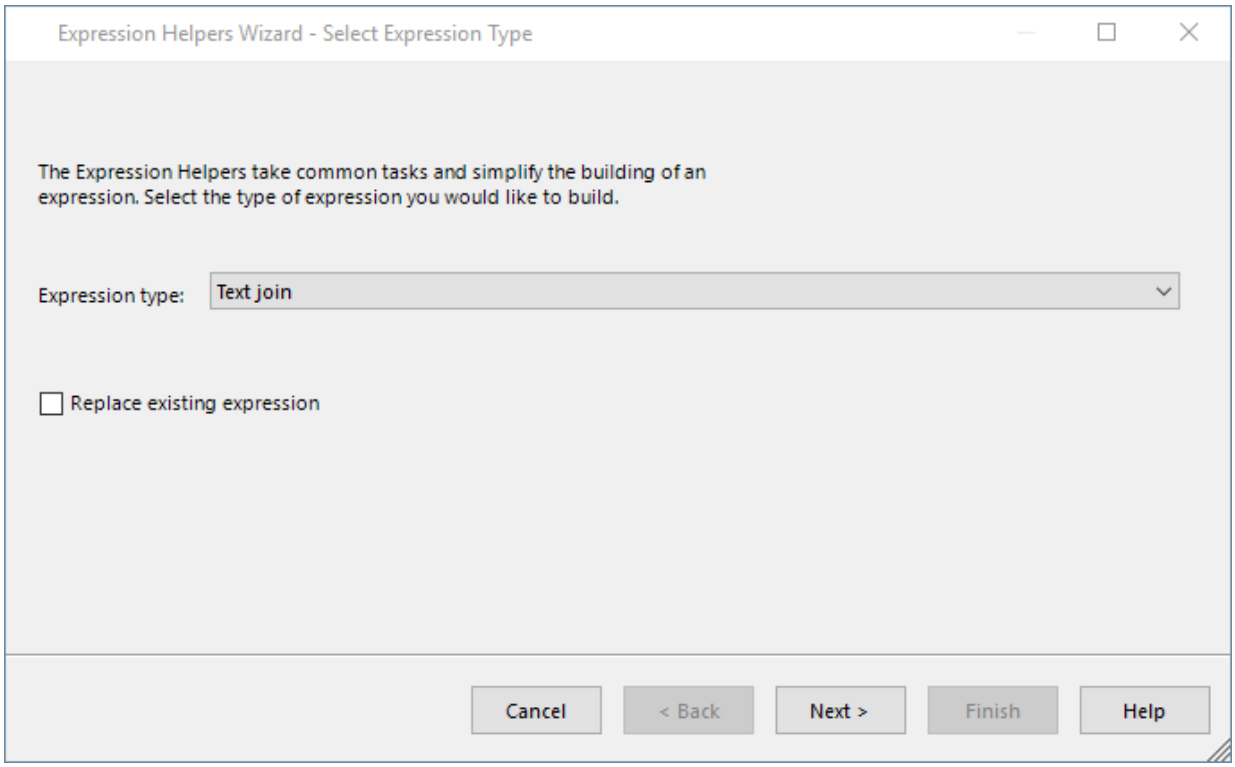
Helper: Text Join

The **Text join** helper allows you to easily concatenate text field values and constants in one or several fields using graphical user interface instead of writing SQL code with CONCAT() or

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

PASTE() functions in Expression Editor. The wizard automatically builds the required expression for you.

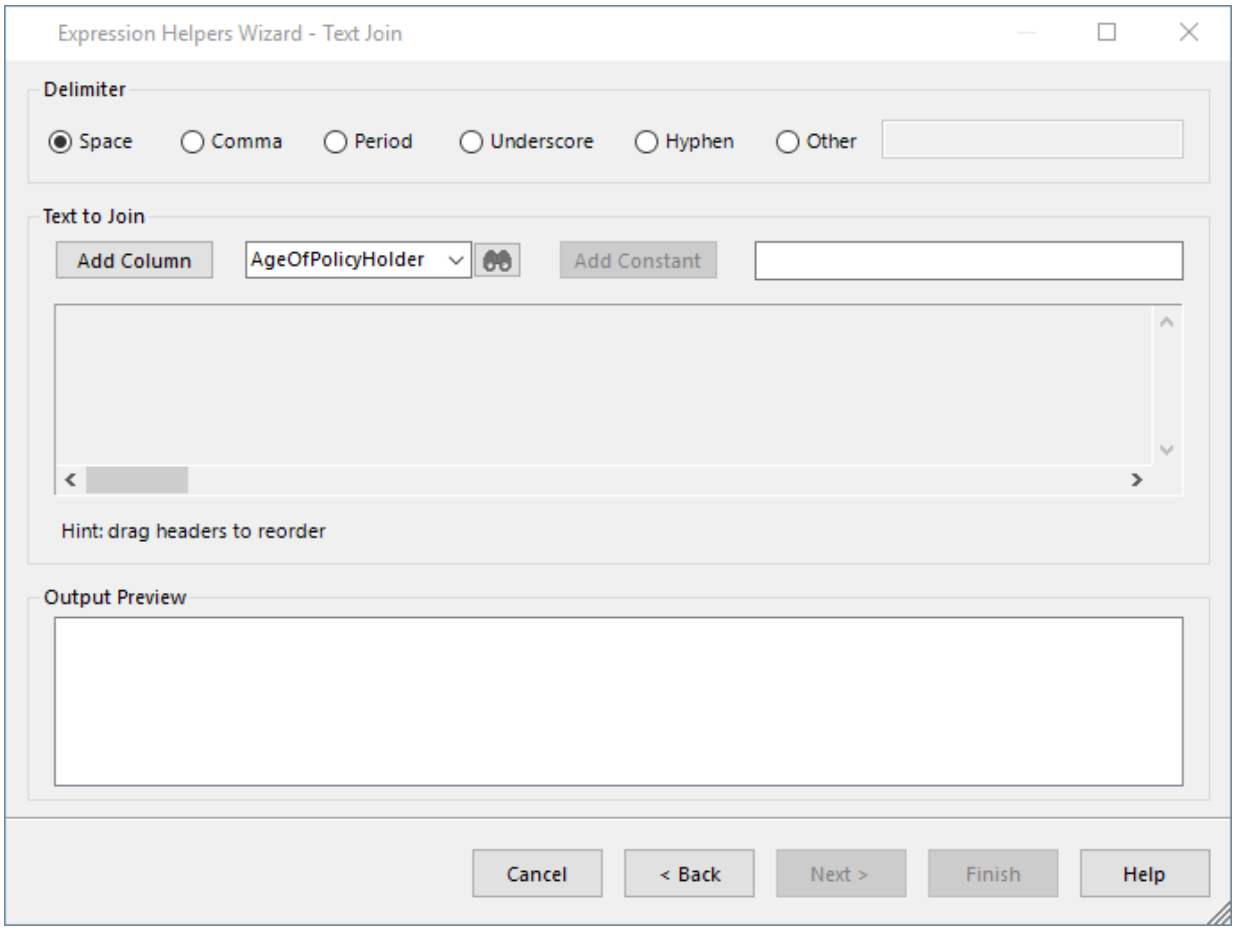
After clicking the Helpers button in the Expression Editor, select **Text join** for the Expression type:



Click **Next**.

The next page provides the interface to interactively build an expression for concatenating string values while previewing the result in real time.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Select a **delimiter** - a string to separate the field values and constants to join. The options are *Space, Comma, Period, Hyphen, Underscore,* and *Other (custom)*.

> If you select ***Other***, specify the custom value. The blank value (empty string) is also admissible. If you want no delimiter inserted between the values being concatenated, select the *Other* option and don't specify anything for its value.

Adding Columns and Constants for Expression

In the Text to Join frame, select and add the fields whose values should be joined (concatenated) and specify the contant strings. Add them in the order in which they are to be joined. Note that you can change the order later by dragging the columns to the desired positions.

- Click the **Add Column** buttton to add the selected fields to the grid, similar to the Data tab in datasets, which shows the structure of the expressions being constructed. The order of the columns in the grid determines the concatenation order. If necessary, use the binoculars button to open the Field Finder window, where you can add multiple fields at a time. After adding any column, you can select a different one and click Add Column, and so on. Each column may be added more than once.
- The **Add Constant** button adds a constant column to the grid using the value in the associated text input box on the right. The constant columns are numbered in order of creation (the headers are: 'Constant 1', 'Constant 2', ...).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

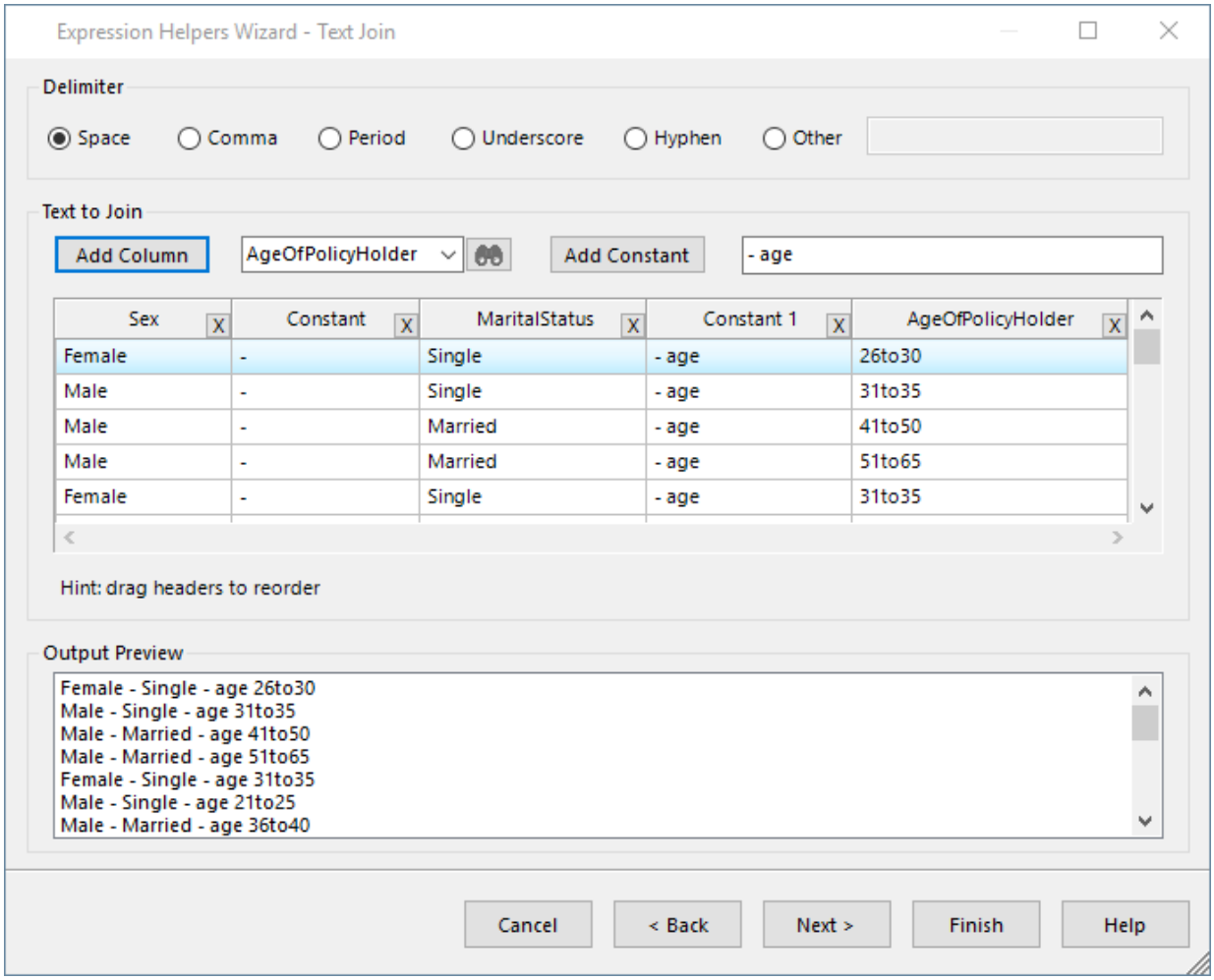
Modifying the Expression

- Columns can be deleted by clicking the [x] icon in their header.
- The order of the columns in the grid can be reordered by dragging them to the desired positions.

Output Preview

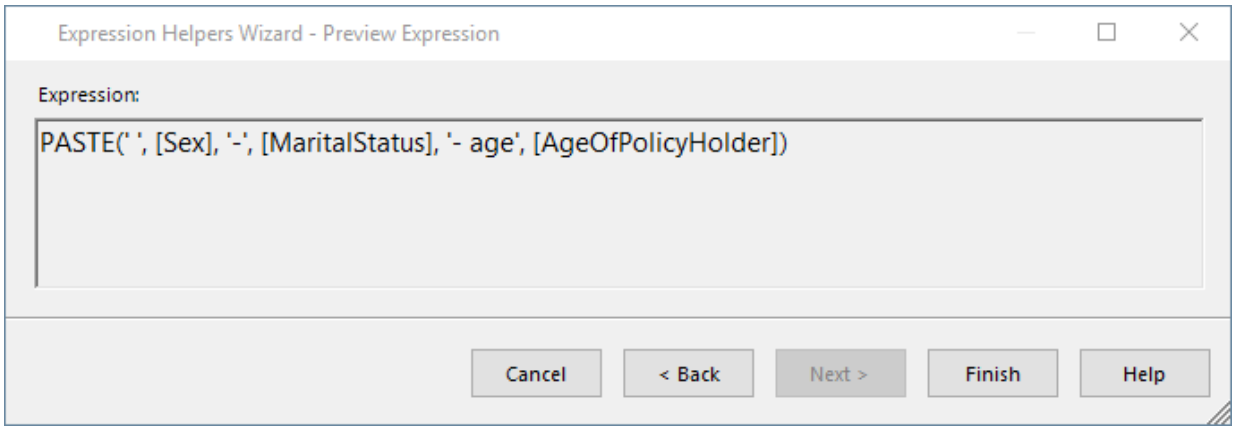
As you build the expression, the **Output Preview** frame shows the value of the joined field to be produced as output. It displays the first 20 records of the output field.

In the example below, we concatenated the values in three columns: Sex, Marital Status, and AgeOfPolicyHolder, using Space as delimiter. We used the **Add Constant** function to add a hyphen (-) between the first two field values, and the string constant "**-age**" as a prefix for the last field value. Thus, from the first three field values "*Female*" "*Single*" "*26to30*" the wizard constructs the string value "*Female - Single - age 26to30*", and so on.

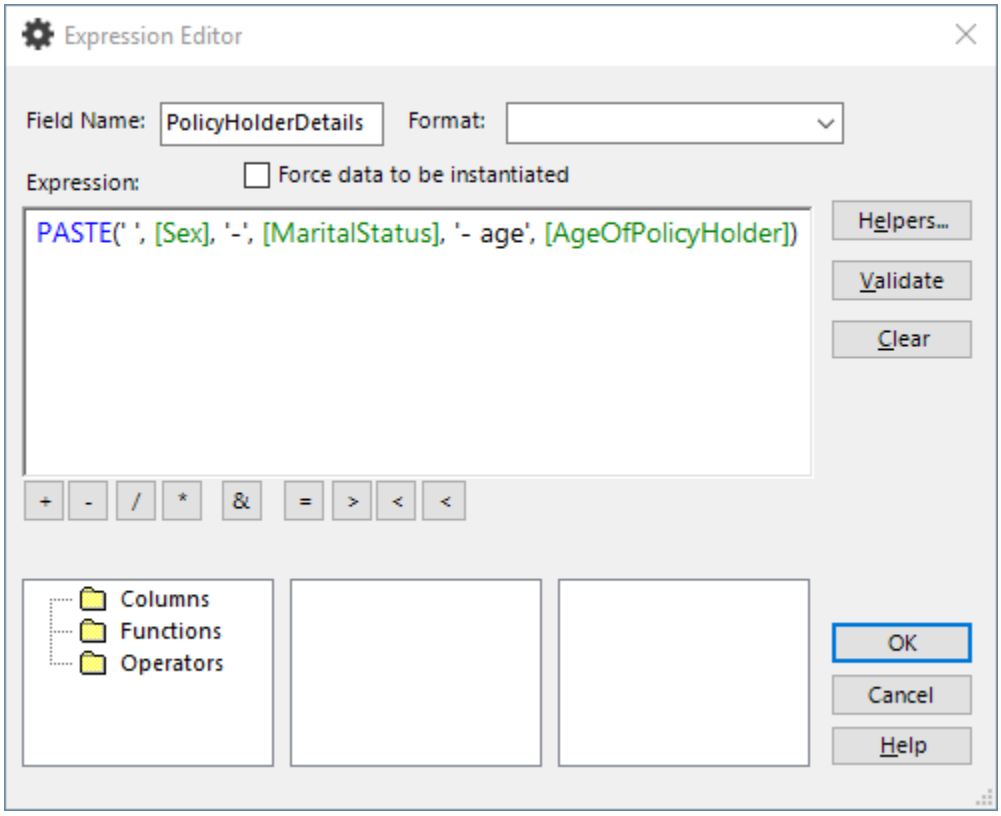


Click **Next** to preview the SQL expression generated by the wizard from your inputs.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

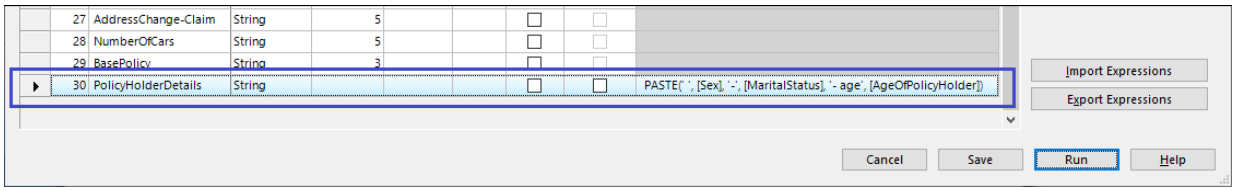


Click **Finish** to generate the expression and view it in the **Expression Editor** dialog, where you can specify the name for the new text join field and manually edit the resulting expression if necessary.



Click OK in the Expression Editor. The generated expression for the new field is shown in the **Dataset Editor** dialog of the Variable Transformations wizard. If you want to edit the expression again, select the new variable and click **Edit Field**. Note that at this stage you will only be able to edit it in the Expression Editor, without invoking the Text Join helper wizard again.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the node, the resulting dataset node appears on the workflow canvas. Double-click on the output dataset to view the data in the new field.

Example: In the following fragment of the Data tab of the output dataset, the new field is shown on the right, next to the original fields that were used to concatenate their values using the Text join helper.

	Sex	MaritalStatus	AgeOfPolicyHolder	PolicyHolderDetails
1	Female	Single	26to30	Female - Single - age 26to30
2	Male	Single	31to35	Male - Single - age 31to35
3	Male	Married	41to50	Male - Married - age 41to50
4	Male	Married	51to65	Male - Married - age 51to65
5	Female	Single	31to35	Female - Single - age 31to35
6	Male	Single	21to25	Male - Single - age 21to25
7	Male	Married	36to40	Male - Married - age 36to40
8	Male	Single	16to17	Male - Single - age 16to17
9	Male	Single	31to35	Male - Single - age 31to35
10	Male	Married	36to40	Male - Married - age 36to40
11	Male	Single	over65	Male - Single - age over65
12	Male	Married	41to50	Male - Married - age 41to50
13	Male	Married	31to35	Male - Married - age 31to35
14	Male	Single	16to17	Male - Single - age 16to17
15	Male	Married	51to65	Male - Married - age 51to65

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Corre

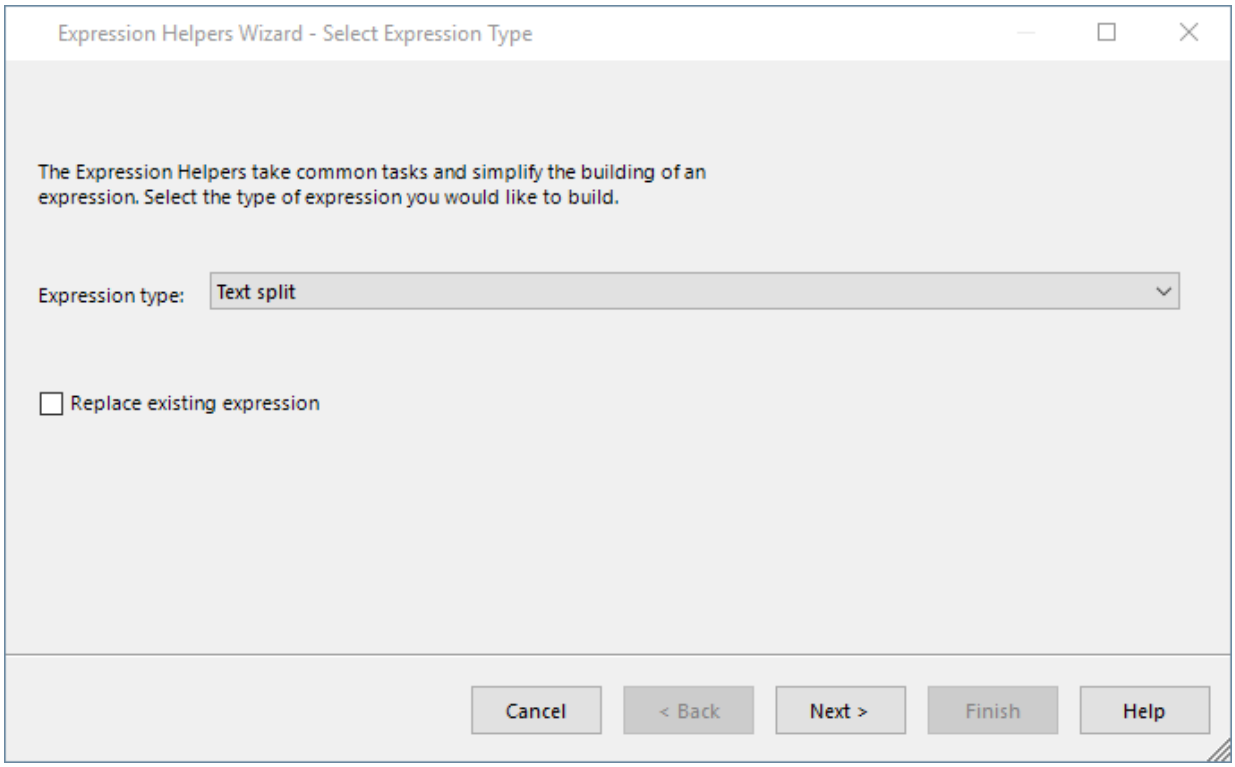
See also: Helper: Text Split

Helper: Text Split

The **Text split** helper allows you to break a field of String type into several fields. Based on your inputs in its graphical user interface, the helper wizard automatically builds the required expression using the PARSEPARTS() function.

After clicking the Helpers button in the Expression Editor, select **Text split** for the Expression type:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



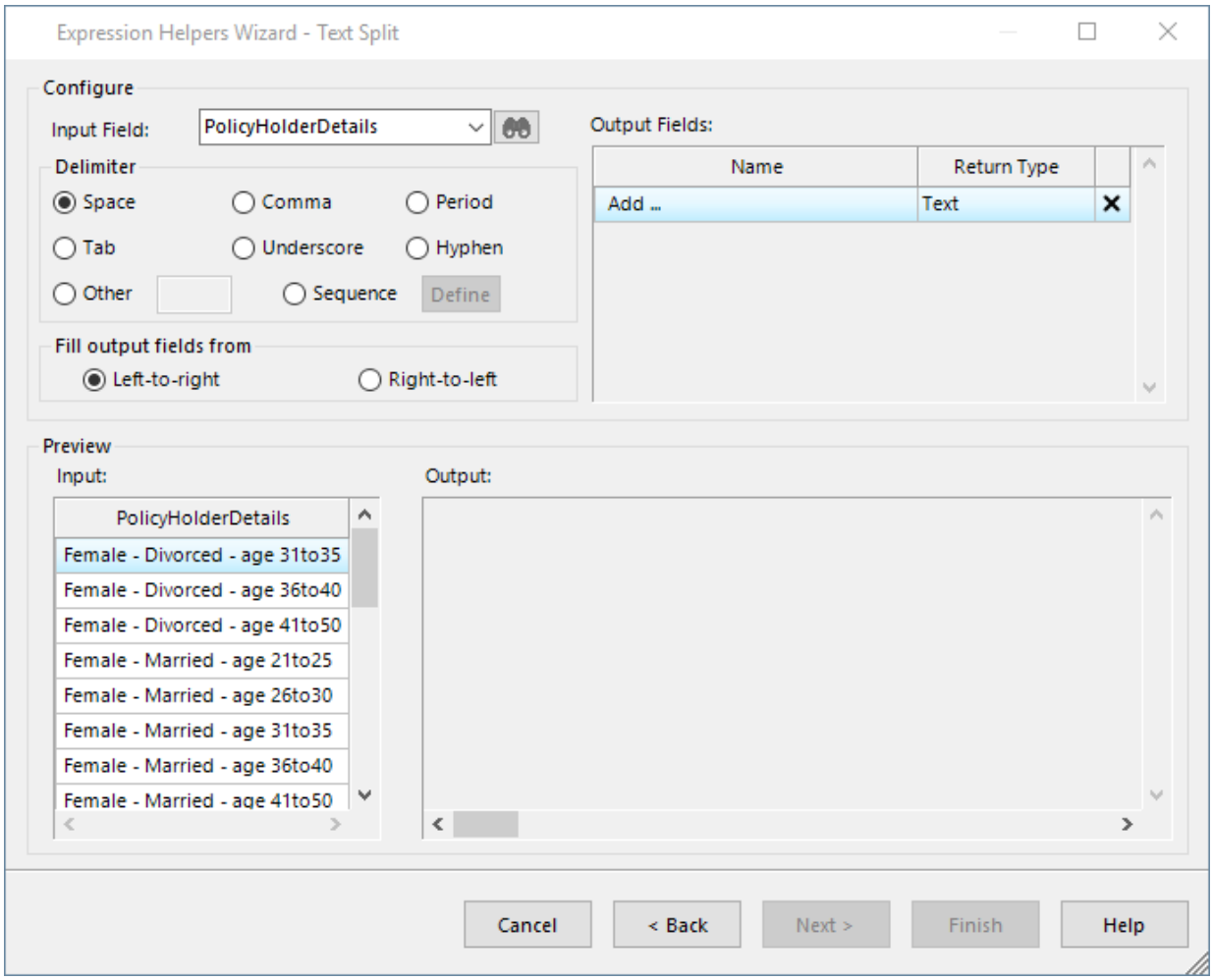
Click **Next**.

For clarity and convenience, in the examples used throughout this topic, we are splitting the field constructed in the Text join helper examples, going the opposite way.

The next page of the wizard provides the interface to interactively build an expression for splitting the field while previewing the result in real time.

Select the **Input Field** from the drop-down list. The Input frame at the bottom-left is populated with the top 20 values of the selected input field. Note that non-string fields are available as well. They are automatically cast to String type in the generated expression.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

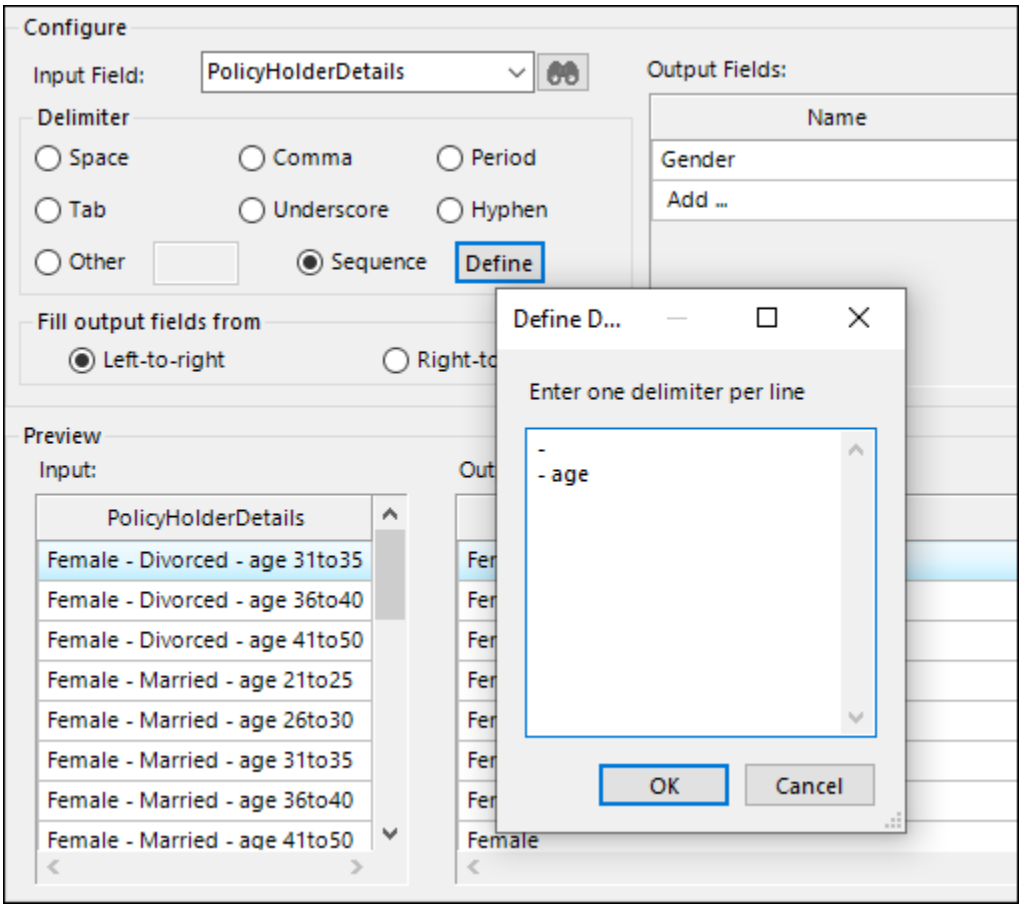


Delimiter Options

Select the **delimiter** - the character(s) that separate the substrings into which you need to split the values of the input field. The options are *Space, Comma, Period, Tab, Hyphen, Underscore, Other (custom)*, or *Sequence.*

- If you select ***Custom***, specify a custom value, which can consist of several characters.
- The ***Sequence*** option is actually a sequence of delimiters. If you select it, click **Define** and enter the sequence of delimiters separating the successive substrings into which you are splitting the input field values. In the example below, hyphen "-" is the first delimiter, and the string "-age" is the second one.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Defining the Output Fields

In the **Output Fields** frame, specify the names of the fields which the input field should be split into.

Note: if you specify too many fields, you may get fields at least partially filled with nulls (missing values). In that case you may want to fill the output fields from right to left instead of from left to right, so select the appropriate radio button for this purpose.

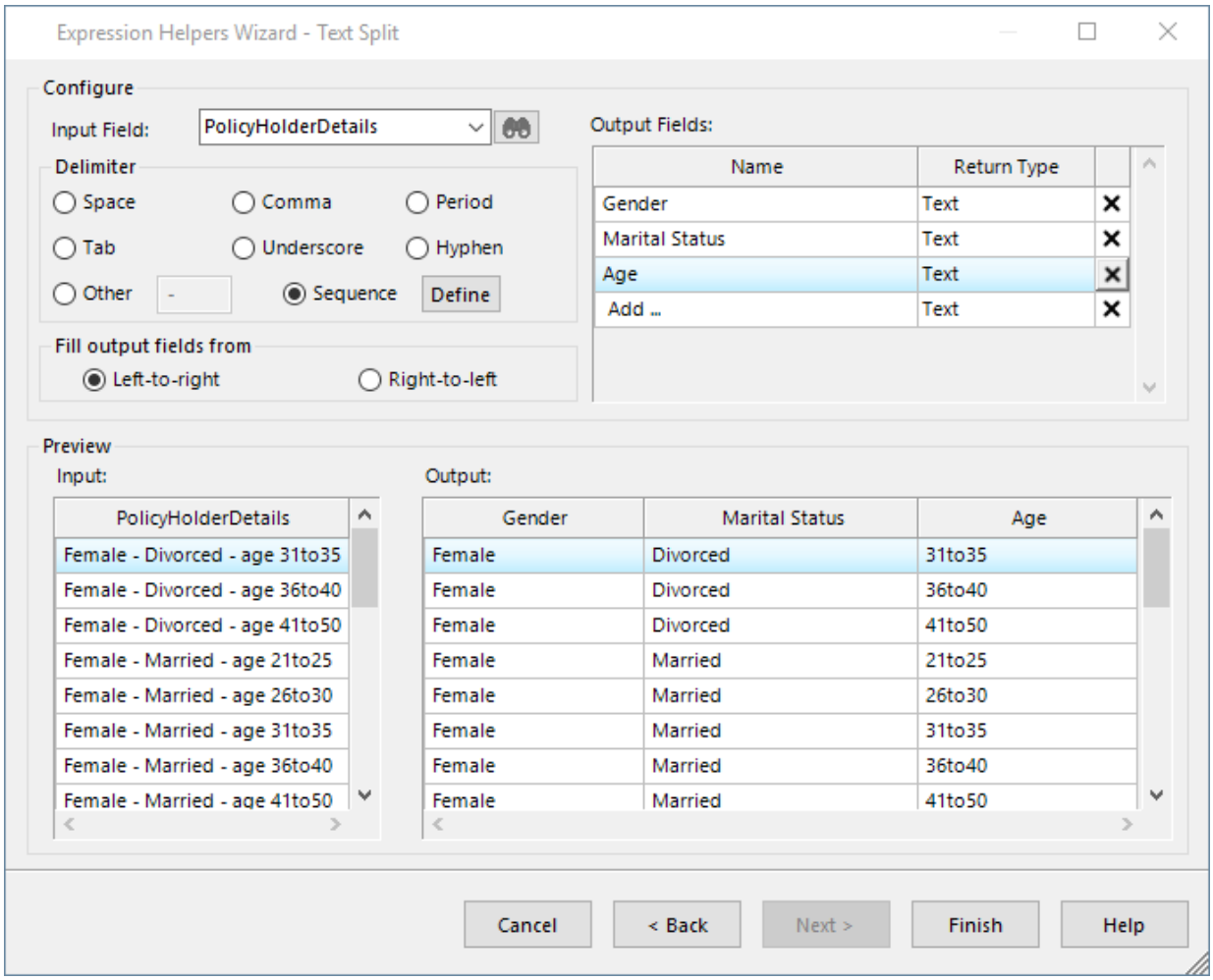
- Upon specifying the field name and type in each row, press Enter to start specifying the next field name.
- Click on the [x] button in the rightmost column if you need to remove a field.

Output Preview

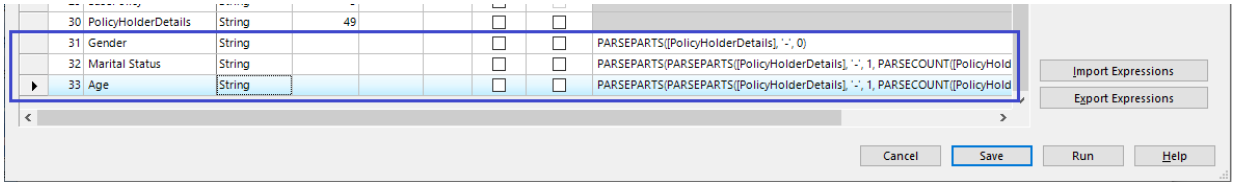
As you build the expression by specifying the delimiters and the output fields, the **Output Preview** frame shows the values of the fields to be produced after splitting the input field. It displays the first 20 records of the output fields.

In the example below, we split the values of the *PolicyHolderDetails* field into three fields: *Gender, Marital Status*, and *Age*.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

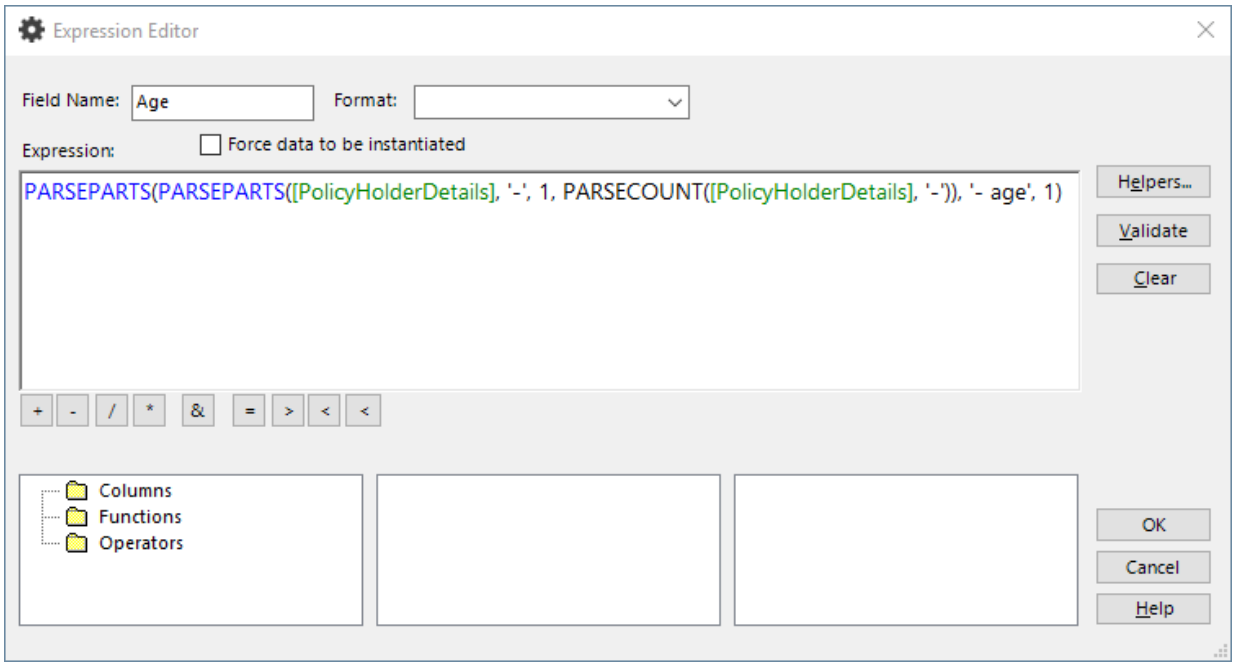


Click **Finish** to generate the SQL expressions for the output fields. The generated expressions for the new fields are shown in the **Dataset Editor** dialog of the Variable Transformations wizard.



If you want to view or edit the expressions, select the desired field and click **Edit Field** to open the **Expression Editor**. Note that at this stage you will only be able to edit the expressions manually in the Expression Editor, not in the Text split helper wizard. Here is the expression for the *Age* field from the above example:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** in the Dataset Editor window to save the settings, or click **Run** to save the settings and run the process immediately. After running the node, the resulting dataset node appears on the workflow canvas. Double-click on the output dataset to view the data in the new field.

Example: In the following fragment of the Data tab of the output dataset, the new fields are shown on the right, next to the original field that was split using the Text split helper.

	PolicyHolderDetails	Gender	Marital Status	Age
1	Female - Single - age 26to30	Female	Single	26to30
2	Male - Single - age 31to35	Male	Single	31to35
3	Male - Married - age 41to50	Male	Married	41to50
4	Male - Married - age 51to65	Male	Married	51to65
5	Female - Single - age 31to35	Female	Single	31to35
6	Male - Single - age 21to25	Male	Single	21to25
7	Male - Married - age 36to40	Male	Married	36to40
8	Male - Single - age 16to17	Male	Single	16to17
9	Male - Single - age 31to35	Male	Single	31to35
10	Male - Married - age 36to40	Male	Married	36to40
11	Male - Single - age over65	Male	Single	over65
12	Male - Married - age 41to50	Male	Married	41to50
13	Male - Married - age 31to35	Male	Married	31to35
14	Male - Single - age 16to17	Male	Single	16to17
15	Male - Married - age 51to65	Male	Married	51to65
16	Male - Single - age 36to40	Male	Single	36to40

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analys

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

See also: Helper: Text Join

Optimal Binning

Helper: Optimal Binning

The Optimal Binning helper allows you to create transformations that group each field (or variable) into new categories that have the highest predictive power with respect to a specified dependent variable. This helper allows you to transform many fields at the same time rather than doing it on a field by field basis.

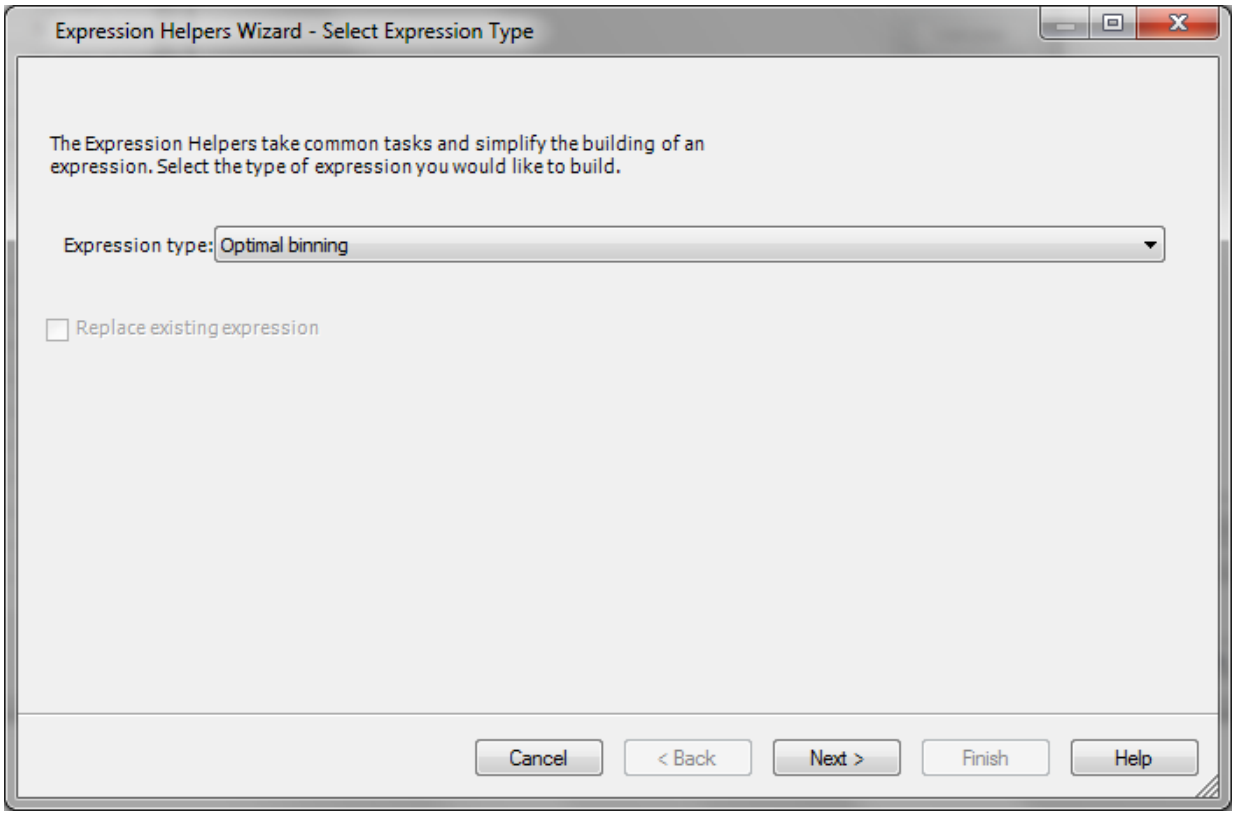
For Optimal Binning with Weight of Evidence as a coarse classing stage of a scorecard development process, see the topic WOE Transformation.

Optimal binning is used for:

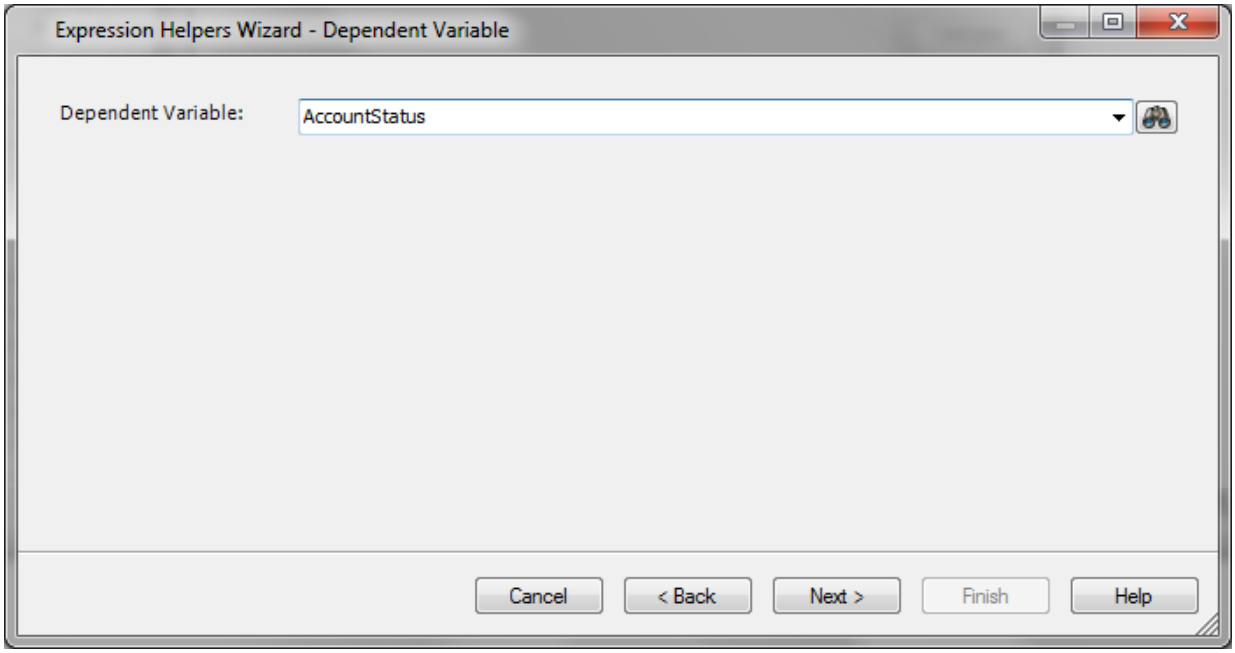
- Discretizing continuous independent variables in a meaningful way
- Reducing the cardinality of discrete independent variables in order to improve their predictive power
- Conditioning the data in a way that improves the performance of predictive models

After clicking the Helpers button in the Expression Editor, select **Optimal Binning** for the Expression type:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

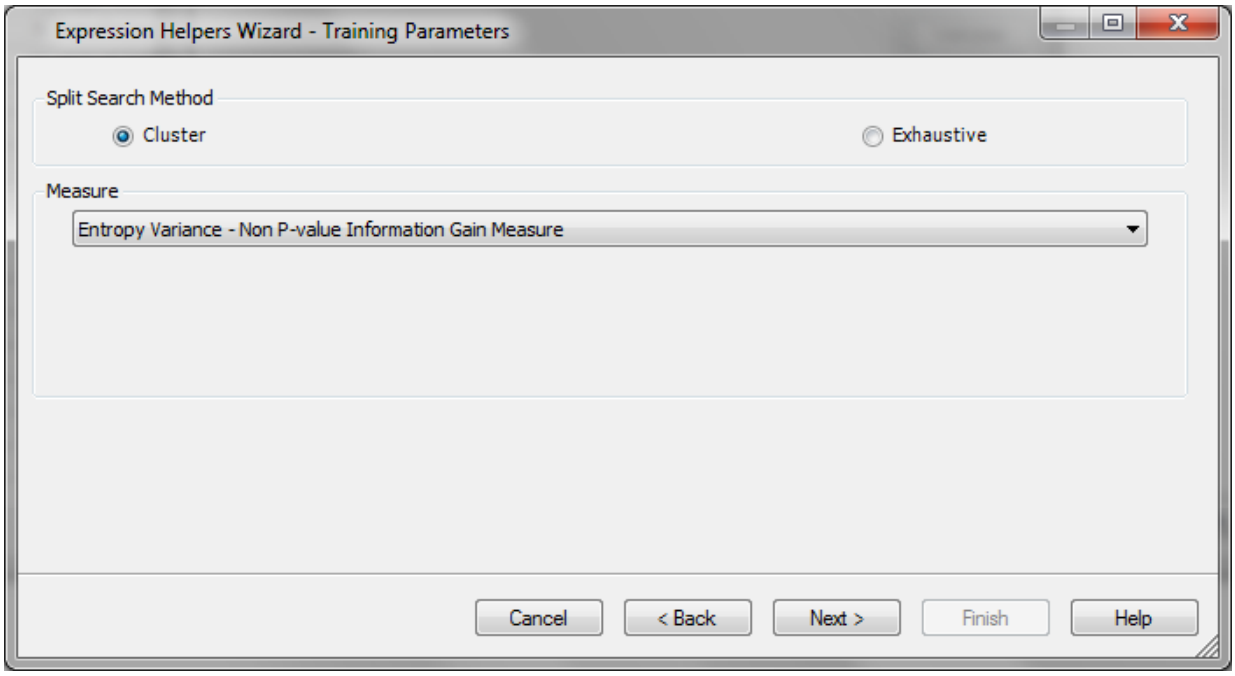


Click Next. At the first step of the Optimal Binning wizard, choose a dependent variable. If the dataset has hundreds or thousands of variables, use the binoculars button to opens the Field Finder where you can easily find and select the variable you are looking for.

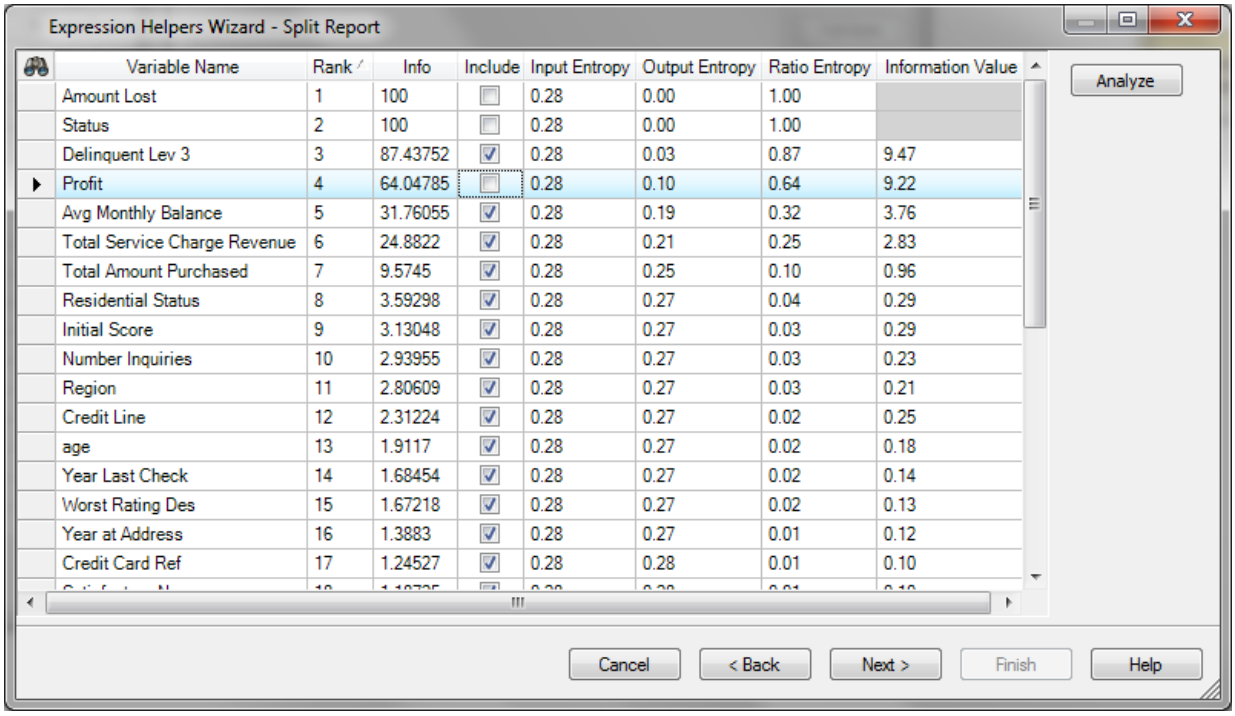


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The next step allows you to define the tree algorithm that controls the binning. This step is the same as in the decision tree wizard. See the topic Inserting a Decision Tree for details.



At the next step, select the variables to perform the binning for. Note that variables can be excluded at a later stage if necessary. This step is identical to the variable selection step in the Decision Tree Wizard. Click **Analyze** if you need to calculate the significance of the input variables with respect to the selected DV at this stage.



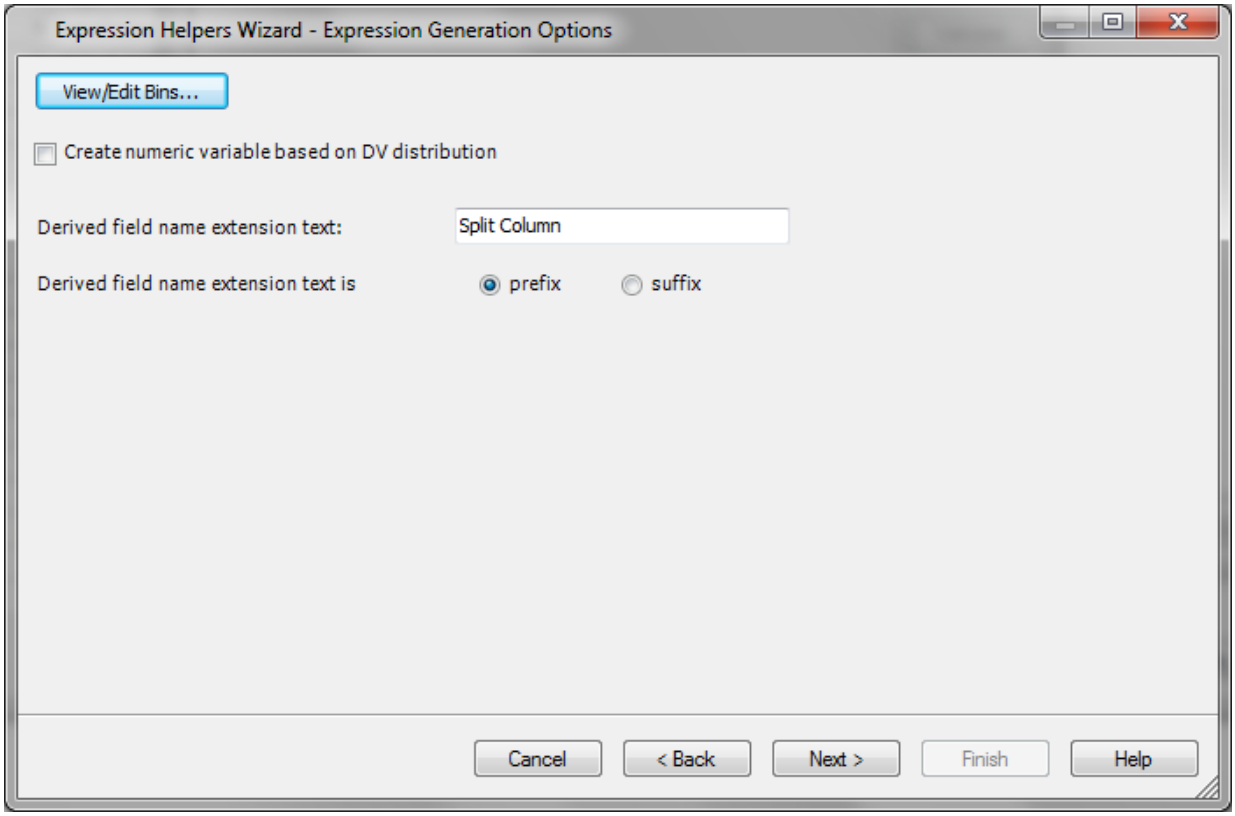
Variable Name	Rank	Info	Include	Input Entropy	Output Entropy	Ratio Entropy	Information Value
Amount Lost	1	100	☐	0.28	0.00	1.00	
Status	2	100	☐	0.28	0.00	1.00	
Delinquent Lev 3	3	87.43752	☑	0.28	0.03	0.87	9.47
Profit	4	64.04785	☐	0.28	0.10	0.64	9.22
Avg Monthly Balance	5	31.76055	☑	0.28	0.19	0.32	3.76
Total Service Charge Revenue	6	24.8822	☑	0.28	0.21	0.25	2.83
Total Amount Purchased	7	9.5745	☑	0.28	0.25	0.10	0.96
Residential Status	8	3.59298	☑	0.28	0.27	0.04	0.29
Initial Score	9	3.13048	☑	0.28	0.27	0.03	0.29
Number Inquiries	10	2.93955	☑	0.28	0.27	0.03	0.23
Region	11	2.80609	☑	0.28	0.27	0.03	0.21
Credit Line	12	2.31224	☑	0.28	0.27	0.02	0.25
age	13	1.9117	☑	0.28	0.27	0.02	0.18
Year Last Check	14	1.68454	☑	0.28	0.27	0.02	0.14
Worst Rating Des	15	1.67218	☑	0.28	0.27	0.02	0.13
Year at Address	16	1.3883	☑	0.28	0.27	0.01	0.12
Credit Card Ref	17	1.24527	☑	0.28	0.28	0.01	0.10

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Click **Next** to proceed to the Expression Generation Options step, which allows you to control general and detailed aspects of the binning.

Expression Generation Options

At this step of the Optimal Binning helper, define the options for the names of the new variables and launch the powerful Optimal Binning Editor to perform the automated and/or manual binning using a Decision Tree interface.



View/Edit Bins	This button opens the Optimal Binning Editor to view and modify the binning for all of the selected variables. See the description of the Optimal Binning Editor for details.
Create numeric variables based on DV distribution	If this is selected, then the CASE statements that represent the bins will express the probability of a specific DV outcome, which must be chosen at the next step. This may be helpful when conditioning the data for a regression model as it forces linearity. See the topic Select Dependent Variable Value. If this option is not selected, then the CASE statements produce a text variable whose values represent the bin labels. If you train a regression model using these variables, the training process will create indicator variables from these discrete inputs.
Derived field name extension text	The text entered here will be added to the original variable names as a prefix or suffix to form the names of the derived variables produced by this helper. The default prefix is "*Split Column*".

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

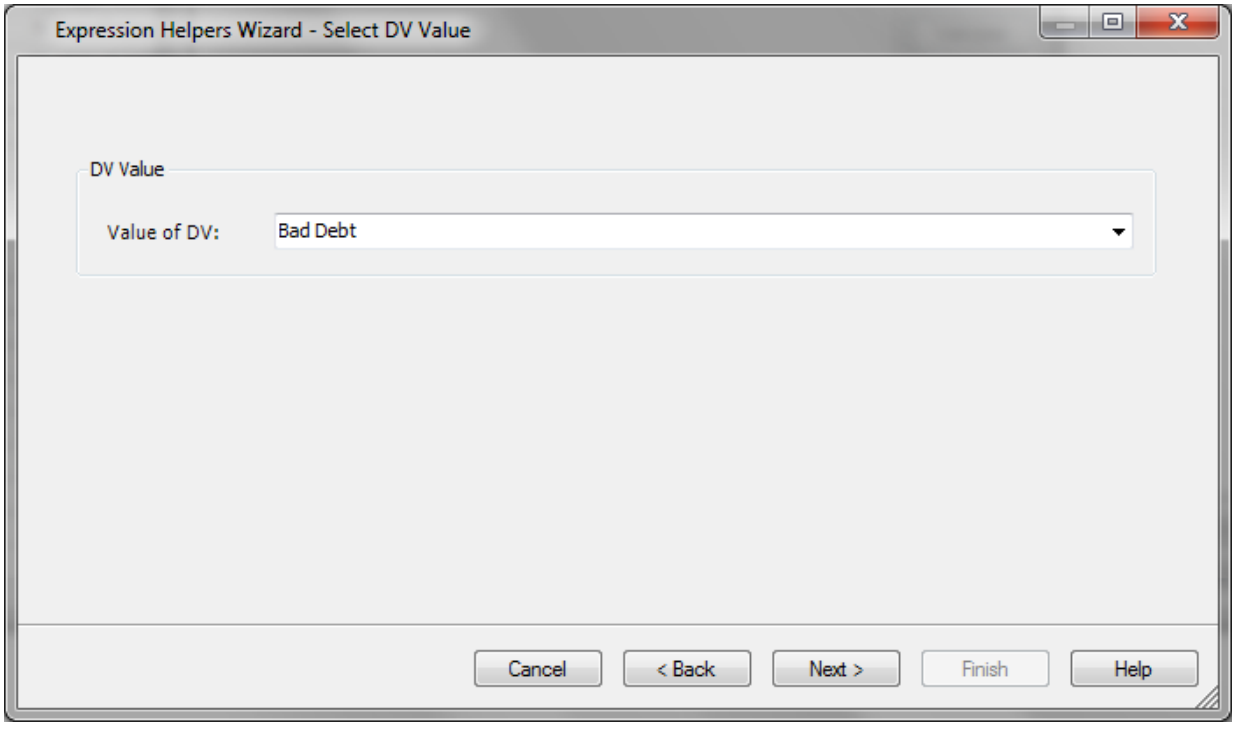
Prefix / Suffix If the *Prefix* option is selected, then the derived variable names are formed by adding the specified extension text at the begging of the original field names. With the *Suffix* option selected, the extension text will be appended at the end of the original field name. The default option is *Prefix*.

Using *prefix* as opposed to *suffix* can be more useful in many cases, as the derived fields can be conveniently identified and grouped by sorting by field name.

In most cases, you need to view or edit the binning of the selected variables before generating the binning expressions. Click the **View/Edit Bins** button to open the Define Tree dialog, which represents the bins as splits of a decision tree.

Select Dependent Variable Value

If you selected the option "*Create numeric variables based on DV distribution*" at the previous step, then the Helper will construct the CASE statements expressing the probabilities of one of the DV values in the bins defined in the Optimal Binning Editor (launched by the View/Edit Bins button). Specify the desired value of the dependent variable at this step:



Here is an example fragment of the resulting expression for one of the input variables in this case:

```
CASE WHEN ([Initial Score]>=0 AND [Initial Score]<255) THEN 0.0505 WHEN ([Initial Score]>=255 AND [Initial Score]<305) THEN 0.0919 ..... ELSE NULL END
```

where 0.0505 and 0.0919 are the probabilities of Account Status = 'Bad Debt' in the two bins produced by the split on the independent variable *Initial Score*.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

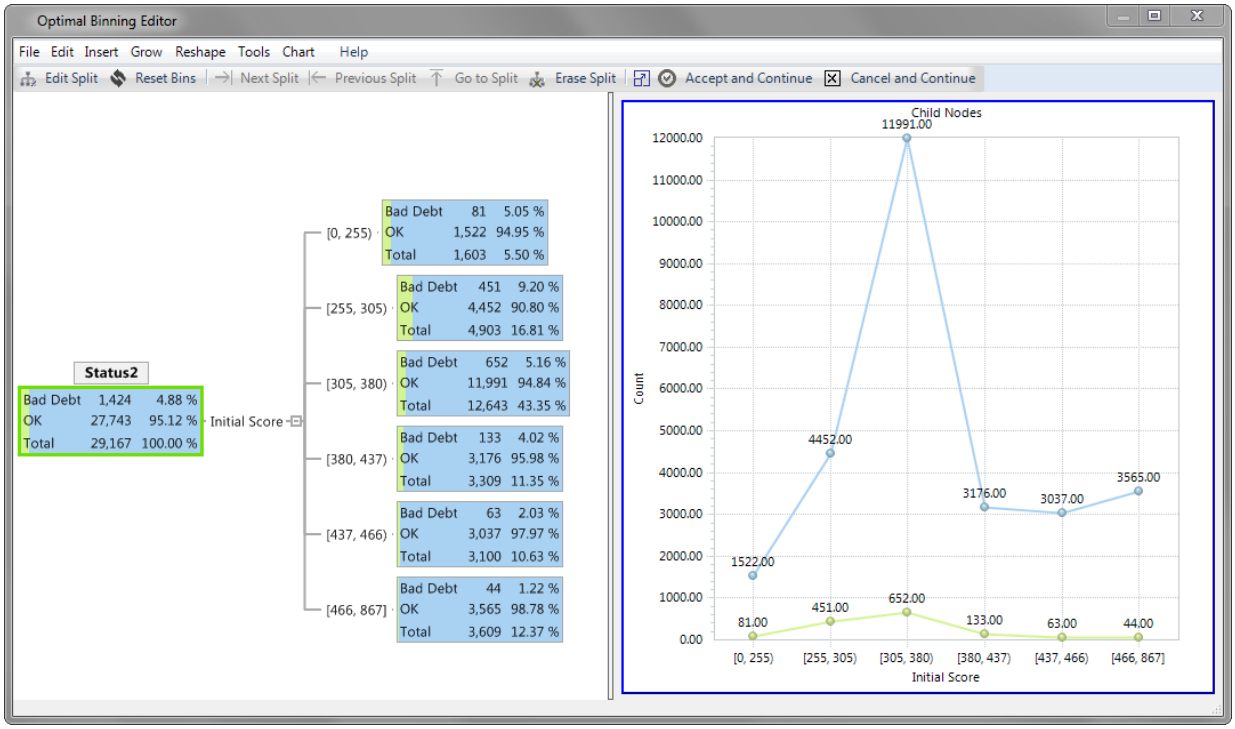
This is sometimes helpful when conditioning the data for a regression model as it forces linearity.

See also: Helper: Optimal Binning and WOE, Expression Generation Options, View or Edit Binning.

View or Edit Bins

The Optimal Binning Editor is launched by the **View/Edit Bins** button in the Expression Generation Options page of the Optimal Binning Helper.

It helps you to view or edit the binning for all variables selected at the previous step. This dialog represents the bins in a decision tree form and allows you to browse through the list of variables, remove unwanted variables and modify the binning.



The diagram on the right-hand side shows the frequencies of the DV values for each bin (i.e., for each child node of the root node, from top to bottom).

Edit Split	Allows you to manually modify the bins of the currently shown variable using Range Editor, which has the same form as in the usual Decision Tree. See below for details.
Reset Bins	Recomputes the bins, effectively removing any manual edits and deletions you have made to **all** of the variables. This is the same as the Find Split command of the decision tree.
Next Split ***Previous Split*** ***Go to Split***	These commands allow you to navigate between the splits in order to view the binning for different variables. See the corresponding Decision Tree topics.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Erase Split	Erases the current split. If you wish to include the variable again, you can do this using the Tools | Attribute Editor command.
Accept and Continue	This command closes the dialog and maintains all of the changes that you have made.
Cancel and Continue	This command closes the dialog and discards all of the changes you have made.
Insert | Variable List	If you want to save the report showing the segmentation used, this command will create a Variable List from the current tree structure.
Tools | Attribute Editor	This command allows you to control the detailed behavior of the binning for each of the variables. See the topic Using the Attribute Editor for details.
Tools | Options	This command allows you to control tree level attributes that control the tree display and binning. See the list of Tree options in the Preferences topic for more details.

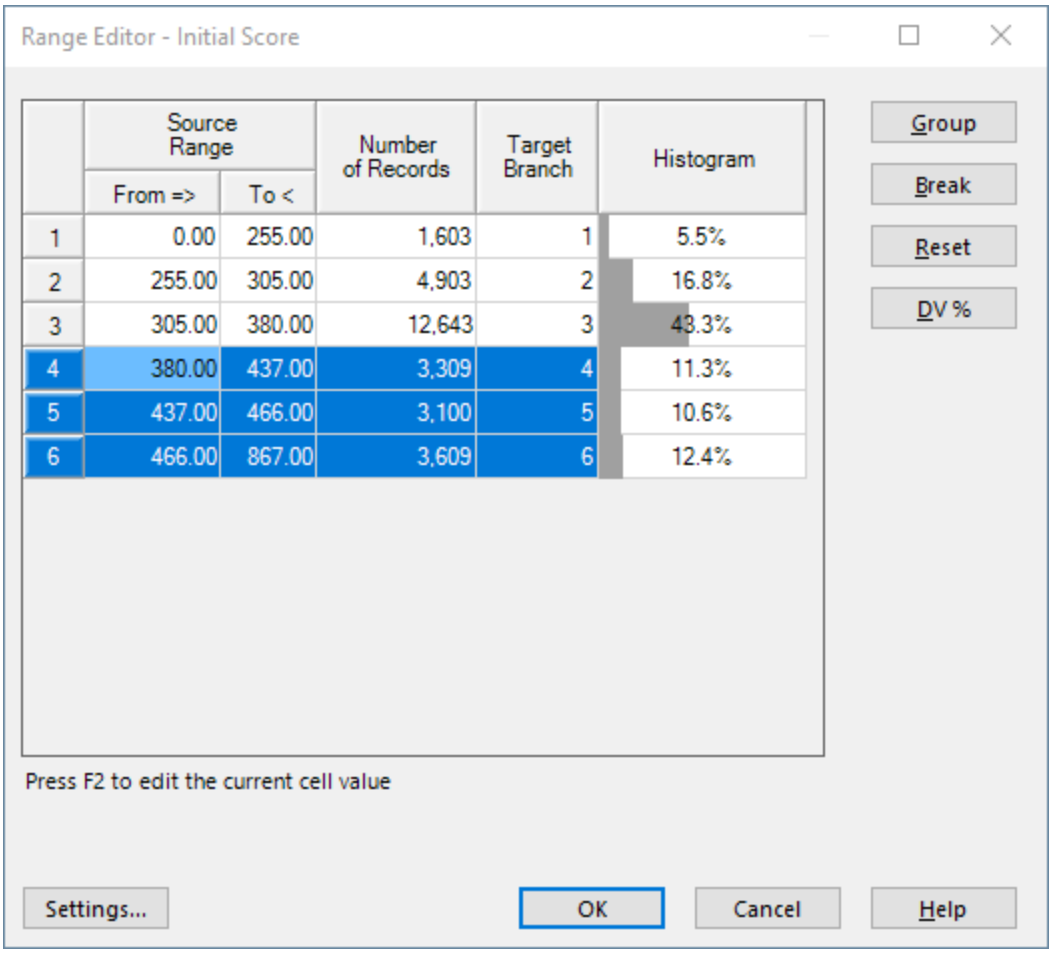
If necessary, you can set a weight variable in the Attribute Editor, which can be opened from the Tools menu. The frequencies and percentages will be adjusted accordingly.

Edit Split

Editing splits in this dialog is like editing splits in a normal decision tree. See the topic Edit Split for more details on the Edit Split command. In the context of building a scorecard, you may want to ensure that the splits have a logical trend.

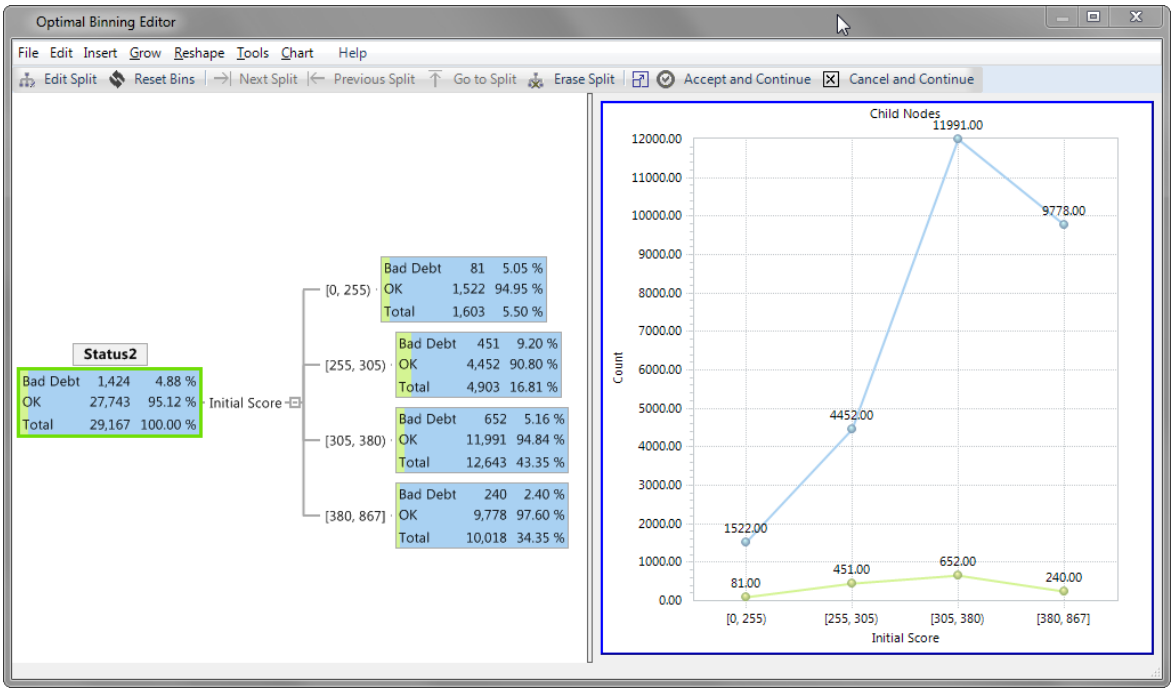
For example, consider the split on the Initial Score variable shown above. Using the Edit Split command, you can adjust the ranges so that the trend of the frequency of Account Status = 'OK' or 'Bad Debt' is according to your business rules.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



After grouping the last three ranges as shown in the screenshot above, we have:.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



When finished editing the bins, click the **Accept and Continue** button in the toolbar (or select **File | Accept and Continue** from the menu). This will close the Optimal Binning Editor, and you will return to the Expression Generation Options page, where you can click **Next** to generate the binning expressions. This brings you to the next wizard page that displays the resulting expressions.

If you want to close the dialog and discard your changes, click **Cancel and Continue**.

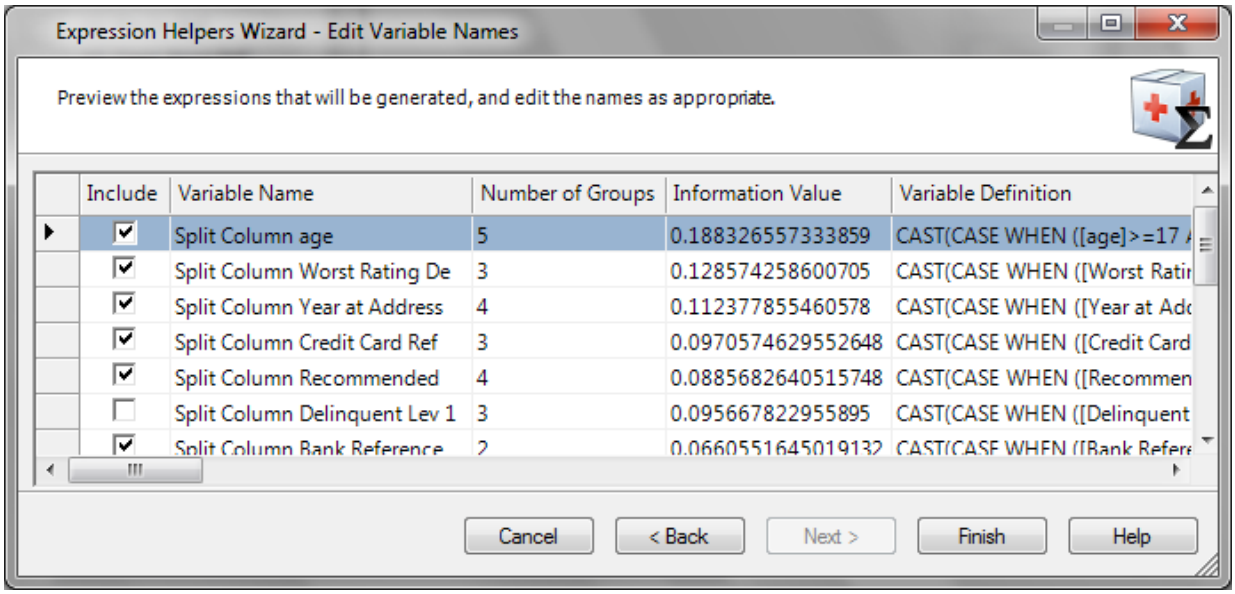
Note that after the new calculated columns have been generated by this wizard, you can modify the binning using the same interface based on the decision tree framework. Just double-click the Variable Transformations node, select any of the variables created with this wizard, and click the Edit button.

Proceed to Viewing the Resulting Expressions.

View Resulting Expressions

The resulting expressions are displayed in a grid for review. You may exclude some variables, edit the variable names, and even edit expressions here. The Information Value is displayed if applicable (i.e., if the selected dependent variable is binary). The formula for Information Value can be found in the topic Mathematical Formulations: Decision Tree.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



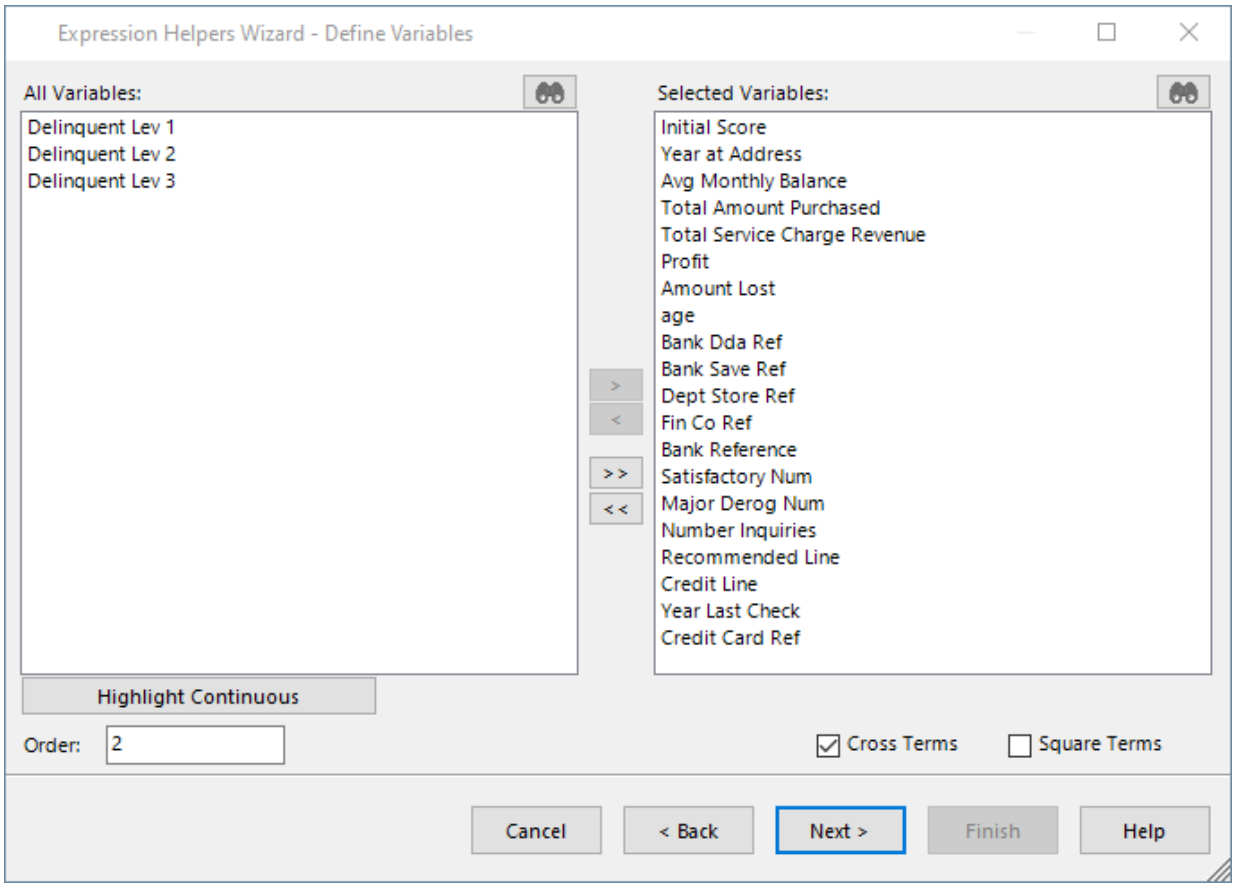
Clicking **Finish** will bring you to the Dataset Editor, where all the new columns are listed. Click **OK** in the Dataset Editor to create the columns.

Helper: Interaction Terms

The Interaction Terms helper allows you to create many variables based on the product of the selected variables and/or their squares. The use of Interaction Terms is a common precursor to training a linear or logistic regression model because it allows the introduction of nonlinear effects in the modeling data in a simple and effective way.

Note that since this helper creates multiple calculated columns, the new field name in the Expression Editor is irrelevant. If necessary, you can edit the names of the new fields at the last step of the Interaction Terms Helper wizard.

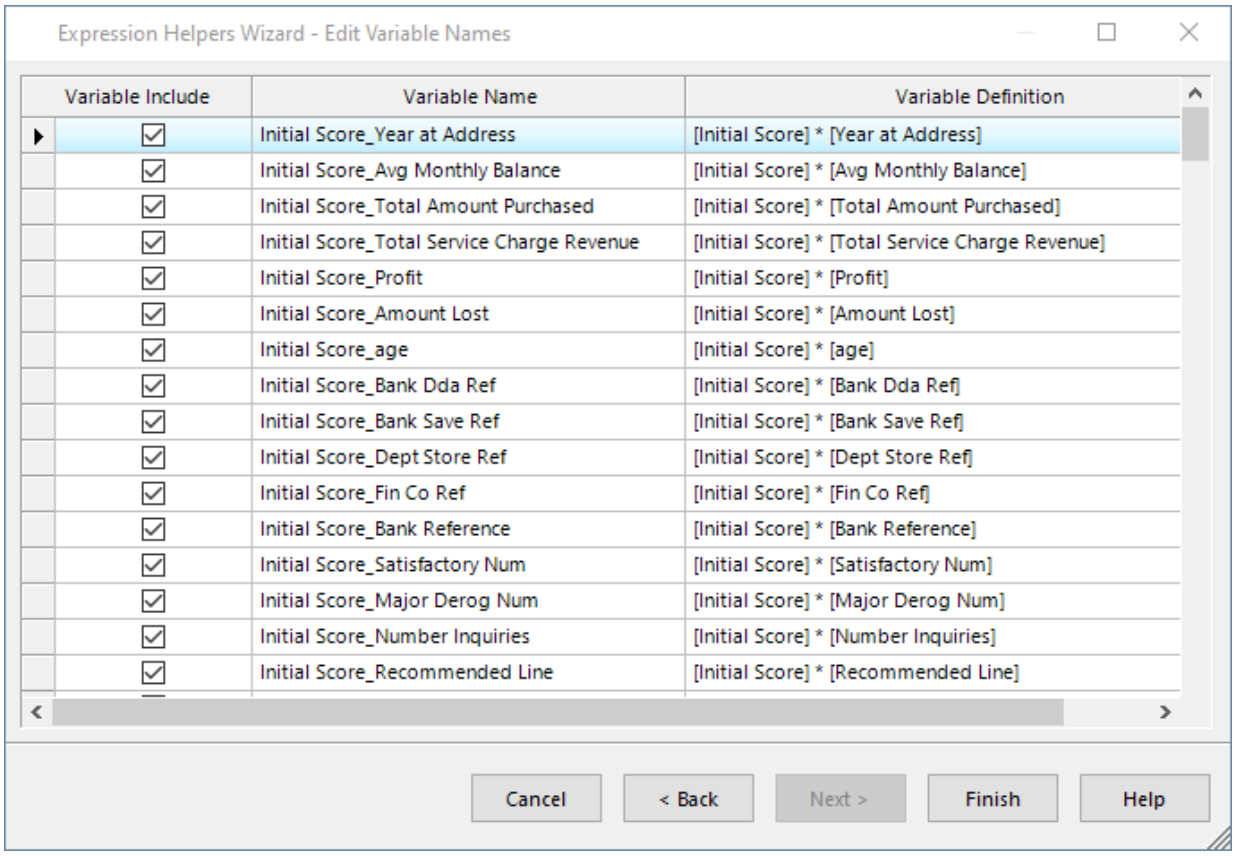
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Selected Variables	Move variables between the All Variables list to the Selected Variables list by highlighting them and pressing the single arrow or by double-clicking the field names. Variables in the Selected Variables list will be included in the combinations.
	The binoculars button opens a Field Finder where you can easily find and select the variables you are looking for.
Highlight Continuous	Highlight the numeric variables with cardinality of 30 or greater
Order	Specify whether to produce 2nd, or 3rd-order interactions. 2 means creating a combination of each pair of variables. 3 means creating a combination for each unique three-way grouping of variables.
Cross Terms	Generate interaction terms
Square Terms	Generate the square of the selected variables (or cubes if the Order is 2)

Click **Next** to generate the expressions for the interaction terms and view the resulting expressions displayed in a grid for review. You may exclude some variables, edit the variable names, and even edit expressions here.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Clicking **Finish** will bring you to the Dataset Editor, where all the new columns are listed. Click **OK** in the Dataset Editor to create the columns.

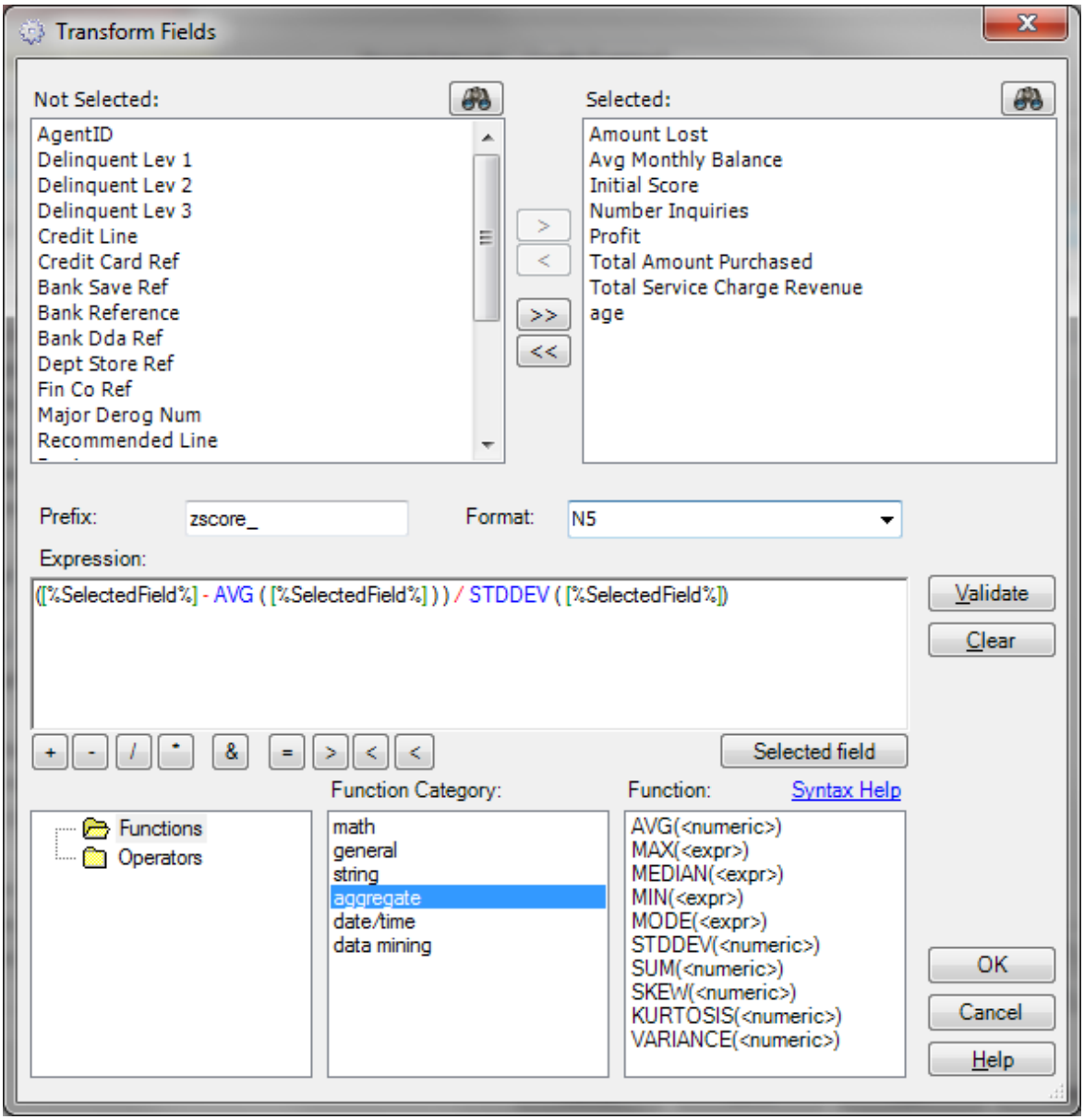
Transforming Multiple Variables

Transform Fields

The Transform Fields button in the Dataset Editor opens an advanced Expression Editor that provides the capability to transform multiple variables at a time: Define an SQL expression and apply it to as many variables as desired. This may be useful in many cases, for example, for standard normalization of multiple numeric variables. The expression specified in the Expression field is applied to all variables in the Selected column at the top right. The variable **[%SelectedField%]** stands for all selected variables. You can enter it quickly at any time by clicking the ***Selected field*** button. See also the topic About the Expression Editor.

In the example below, for each selected variable **x**, the transformation `(x - Mean) / (Standard Deviation)` will be applied.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



For detailed description of format strings, see Specifying Format.

See also:

- Dataset Editor
- Expression Editor: Overview
- Substitute Missing Values

Defining Special Values

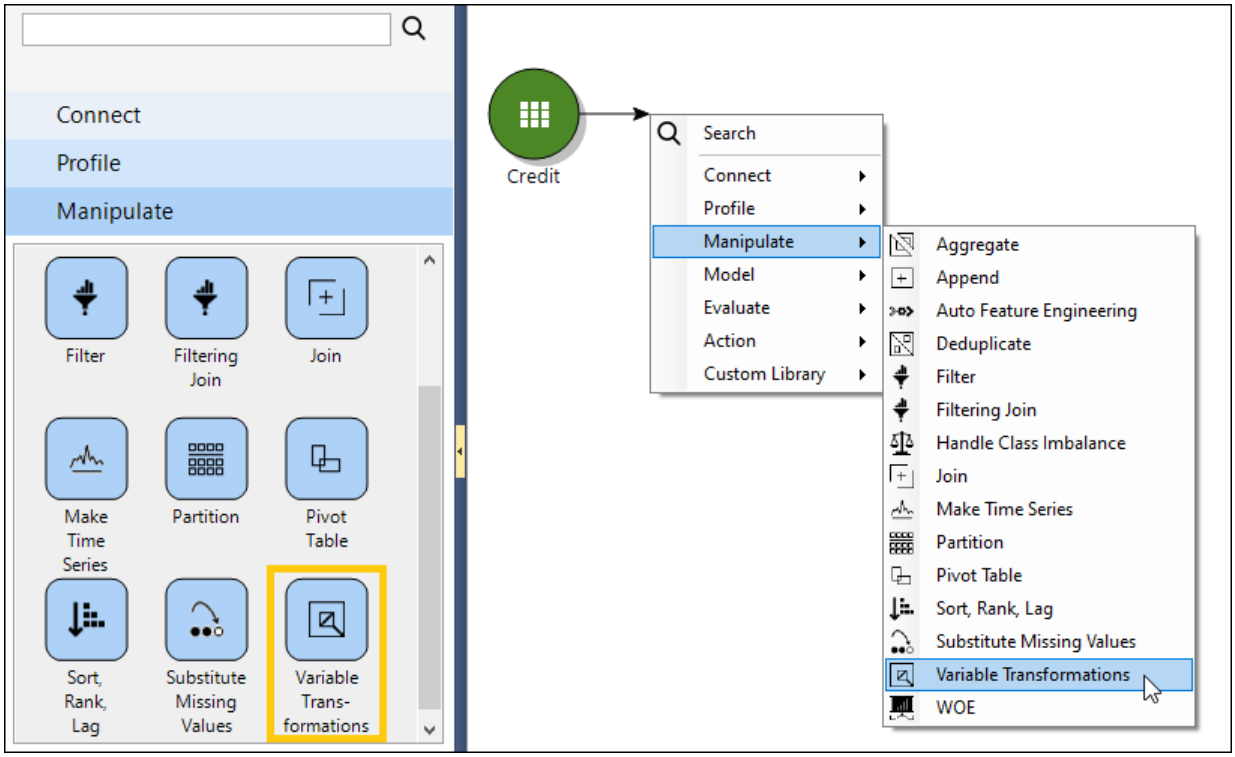
Certain values in some fields of a dataset may have special meaning and may need to be identified accordingly. The Dataset Editor in the Variable Transformations node wizard provides the capability

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

to define and manage ***special values*** (e.g., if a value like "-999" in an otherwise positive numeric field has a special meaning). It also provides the means to define certain values as null values (e.g., if the character "?" means missing data in a given field).

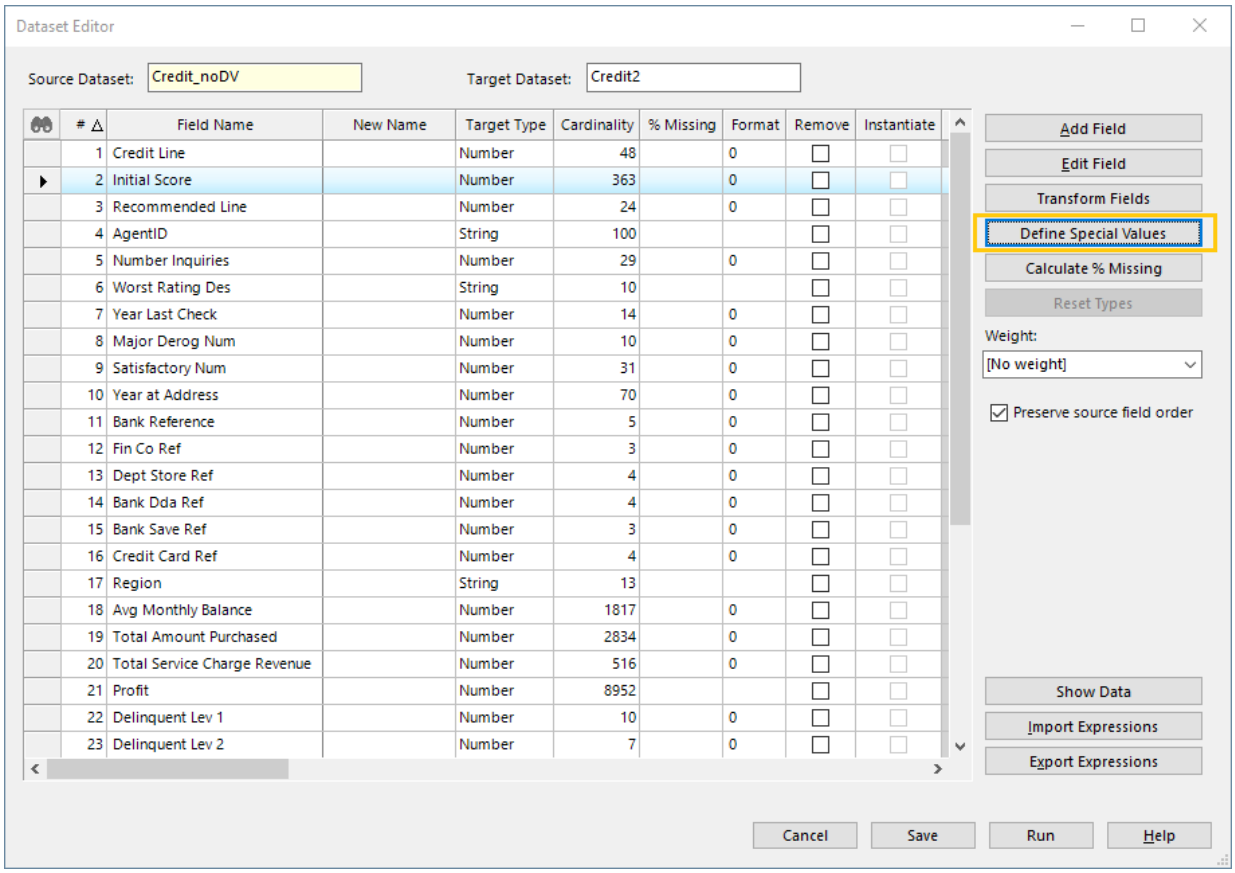
To define and, if necessary, substitute special values, in the general case, one new field will be created for value substitution and another to keep the information about the values.

Add a new **Variable Transformations** node or use an existing one. See the topic Variable Transformations Node for information about the **Variable Transformations** node and Dataset Editor.



In the Dataset Editor, click on the field that contains values to be defined as special. The field becomes highlighted. Click on the button **Define Special Values** on the right-hand side.

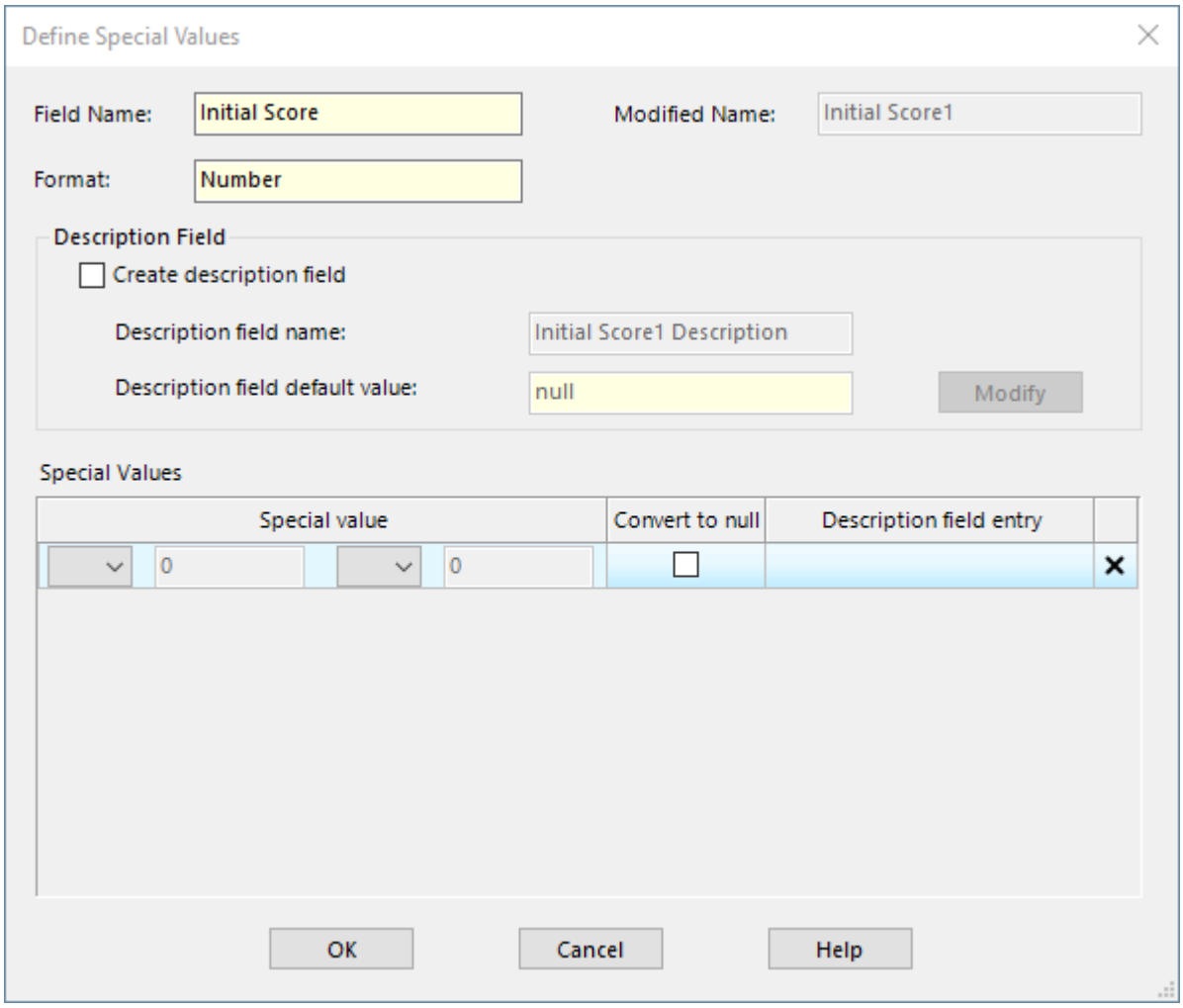
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The Define Special Values editor window opens. The name of the currently selected field is displayed at the top. The wizard allows you to do the following:

- Convert certain values to ***null*** or leave them as they are. A new duplicate field will be created, to preserve the original information.
- Create a description field where you can assign designations based on the special flagged values.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



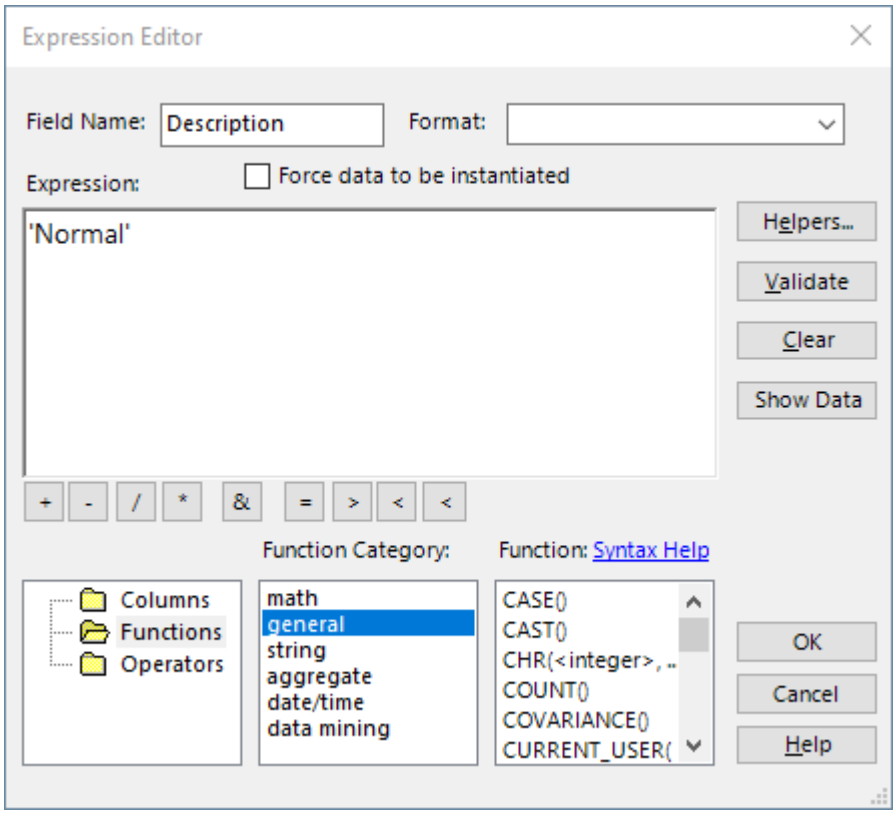
Example: Suppose the field *Initial Score* with "normal" values between 100 and 700 also contains values like 999 which means that the record does not have an initial score, and values like 0 which means the score or the whole record could not be identified.

If both 999 and 0 need to be converted to ***null*** for the purposes of running calculations, a new field can be created where records with 999 are labeled 'No score', records with 000 get a value of "Not found", and all other records are labeled 'Normal'. The information in the original column need to be preserved.

Value conversion options: Convert to null, a custom string, a number, or an expression. Note that all string values must be specified in single quotes.

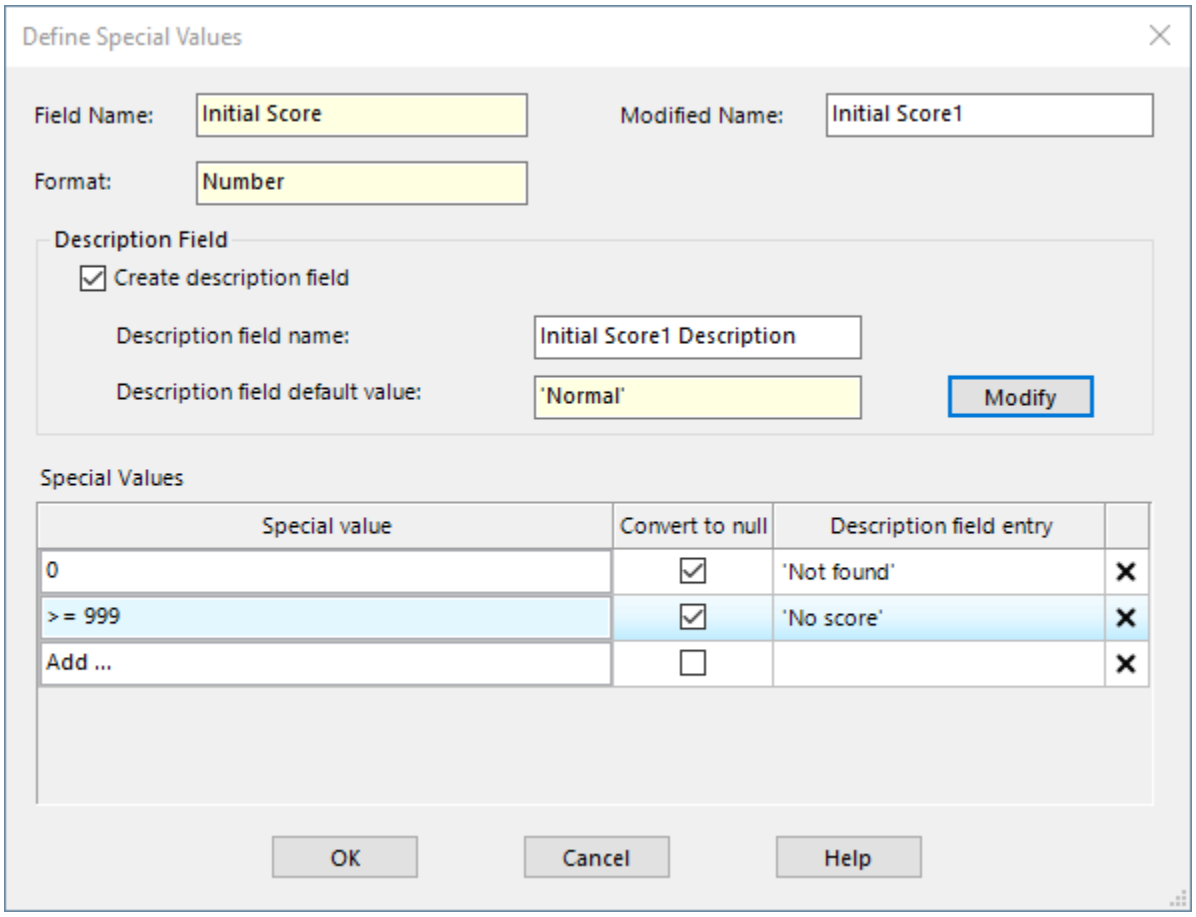
If you wish to define a description field, select the option "*Create description field*". Click the **Modify** button to specify the default value that will flag all "normal" (non-special) values. Note that the wizard (Expression Editor) allows you not only to define constant values, but arbitrary expressions using other fields.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

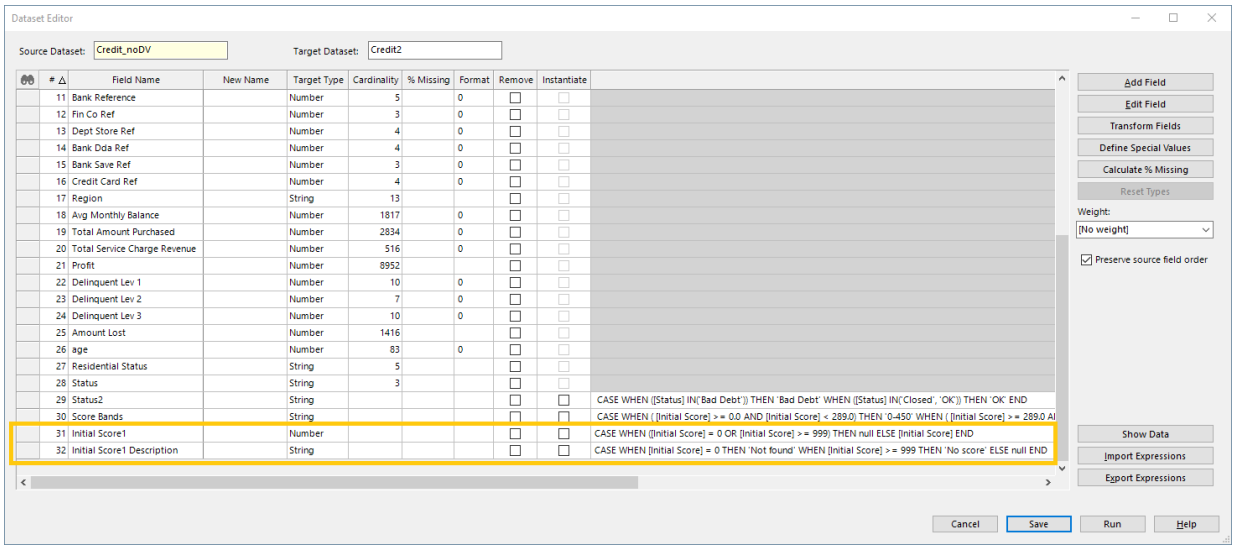


In the **Special Values** table, add the special values you need to flag. Specify whether to convert them to null in the new field or leave as they are. Note that leaving the values unchanged for numeric variables may substantially affect calculations with this field. In the example below, values 0 and all values exceeding 999 in the Initial Score field will be converted to null and in the new field they will be designated 'Not found' and 'No score', respectively. The normal ones will be labeled 'Normal'.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click OK. This creates two new fields (if the Description field option was selected). In the first one, the special values are replaced by null, while the second contains the flags you specified for the special values.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the operation, double-click the output dataset node to view the results in its Data tab.

See Also:

- Variable Transformations Node
- Expression Editor: Overview

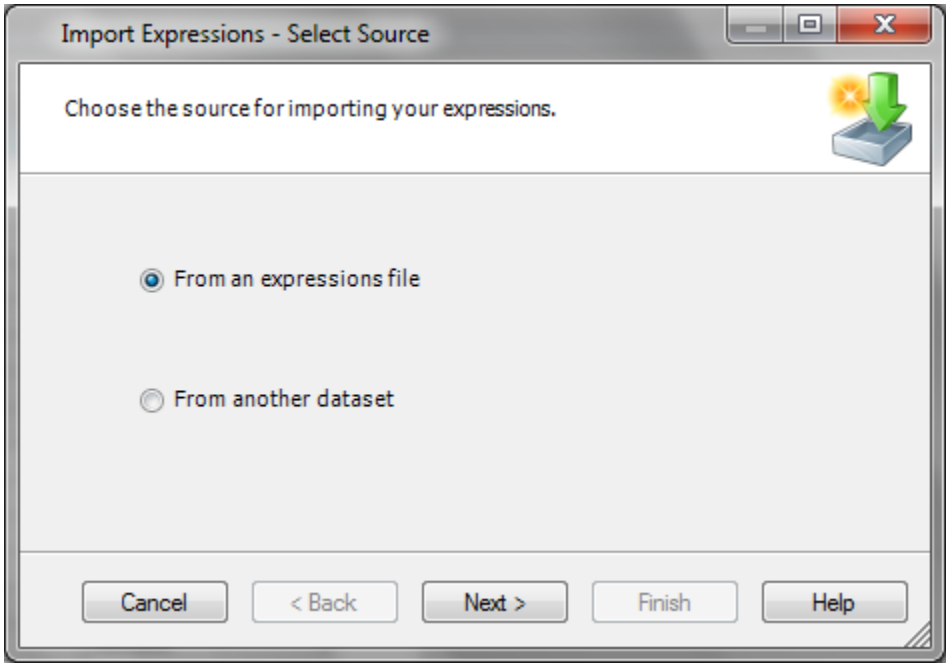
Import Expressions

Dataset Editor allows you to create new calculated columns in your dataset by importing expressions from calculated columns of another dataset or from an expressions file (XML).

For the source dataset you can choose any dataset in the current project or any other project from your Working Directory.

An *expressions XML file* is a file created previously using the Export Expressions function in Dataset Editor.

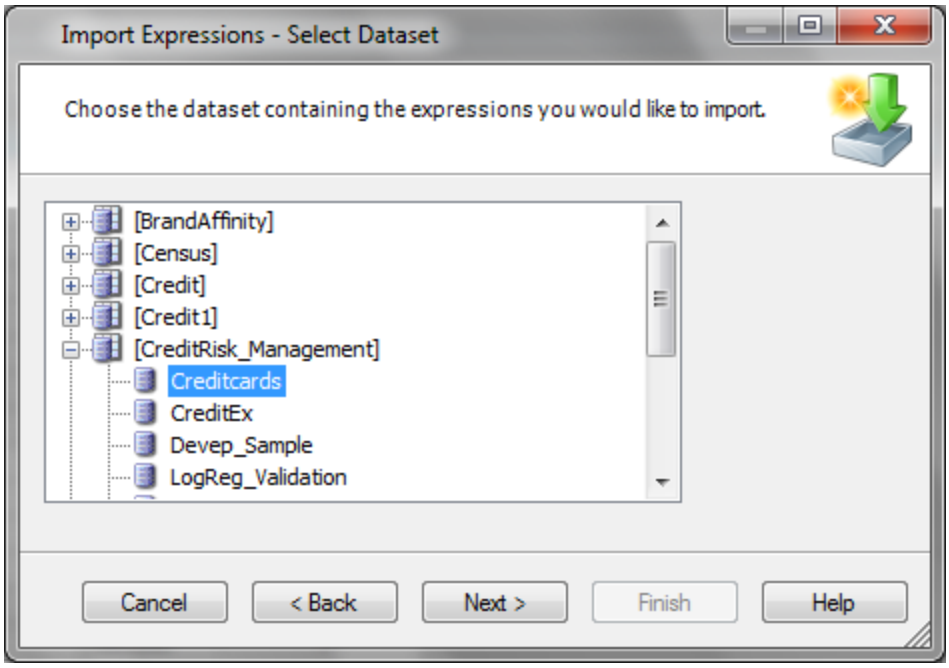
Click the **Import Expressions** button and select the source of expressions to import:



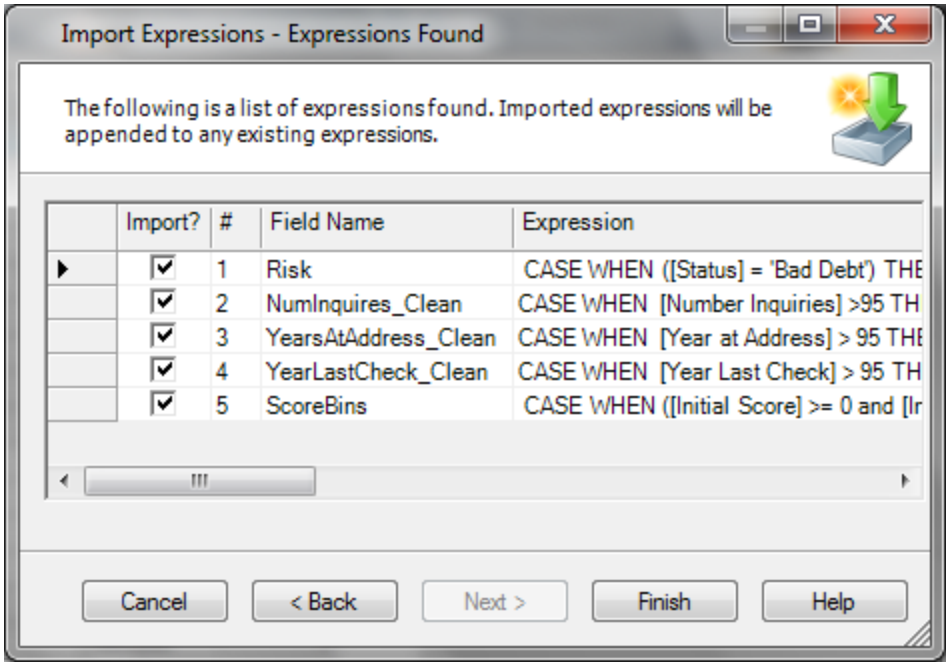
If you chose the expressions file option, the next page prompts you to specify the name of the source file (a valid XML file with expressions defining columns).

If you chose the dataset option, the next page allows you to browse for the source dataset. Any dataset from any project in the Working Directory can be selected.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Select the source dataset and click **Next**. The next page displays the expressions from the selected dataset or expressions file (if there are any).



Here you can select the desired expressions (calculated fields) to import, edit the target field name, or even edit the expression.

A warning is provided if the current dataset already has a field with the same name. In this case, modify the target field name in this dialog.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

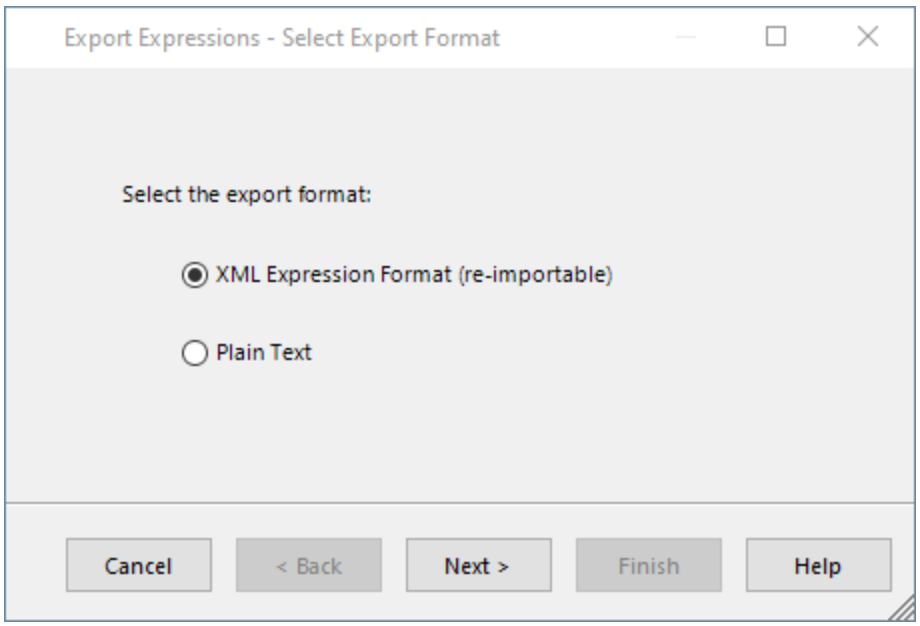
See also: Export Expressions

Export Expressions

Dataset Editor allows you to export expressions from calculated columns to external files (XML or plain text). This may be necessary for example in the case where you created complex expressions for some columns and want to save them for future use in other datasets, or even share them with other users.

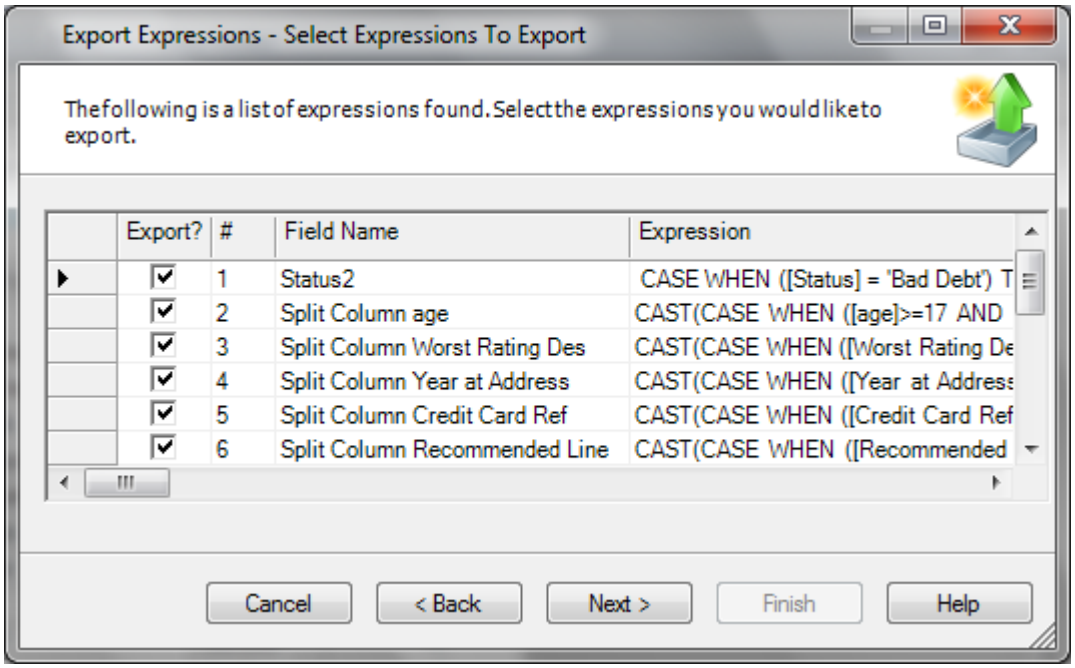
Click the **Export Expressions** button and select the type for the target file.

- If you choose **XML Expression Format (re-importable)**, the expressions will be written to an XML file that can be imported in the future using the Import Expressions command.
- If you choose **Plain Text**, the expressions will be exported to a plain text file exactly as they are displayed in Dataset Editor or Expression Editor. Note that a plain text expressions file cannot be imported.

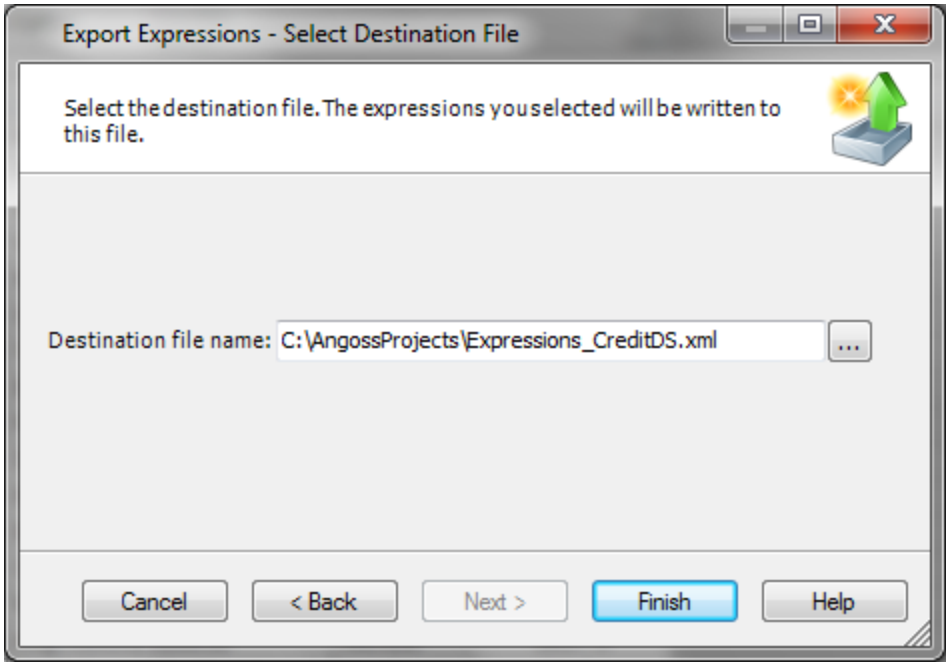


Click **Next** to select the expressions to export. The next page displays expressions for all calculated columns in the current dataset. You can unselect the expressions you don't need to export.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



At the next step, specify the location and name of the destination file:



See also: Import Expressions

Substitute Missing Values

The **Substitute Missing Values** node in the **Manipulate** palette performs substitution of missing values for multiple variables at a time. Unlike the Substitute Missing Values Helper or the Transform Fields function in the **Variable Transformation** node, this node does not create new variables, but

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

substitutes the values in the original variables. It also has an option for handling infinity values +/-{INF}. Applying this node may be especially useful to prepare data for imbalanced class handling and the Auto Feature Engineering node transformations.

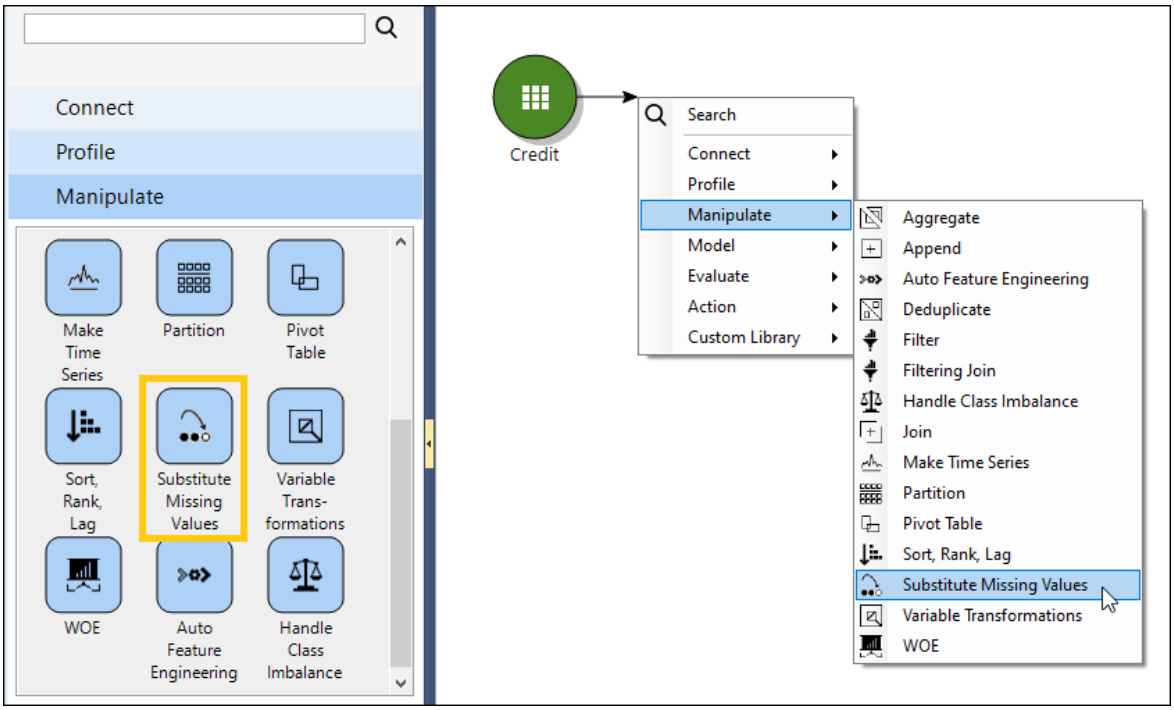
The node wizard allows you to select the variables to apply the substitution and the choose the substitution method for each variable type.

To add a Substitute Missing Values node to the workflow:

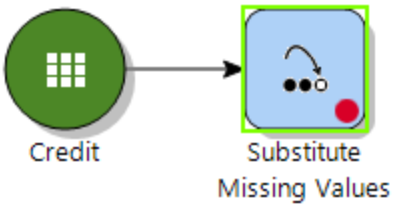
- Drag a connection arrow from the input dataset to any area on the workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Manipulate –> Substitute Missing Values**, as in the example screenshot below. A new **Substitute Missing Values** node will be created, and the wizard will open.

OR:

- Drag the **Substitute Missing Values** node from the **Manipulate** palette onto the canvas and connect the input dataset to it.



If the **Substitute Missing Values** node is already on the canvas, double-click it or right-click on it and select **Modify** from the context menu to open the wizard.

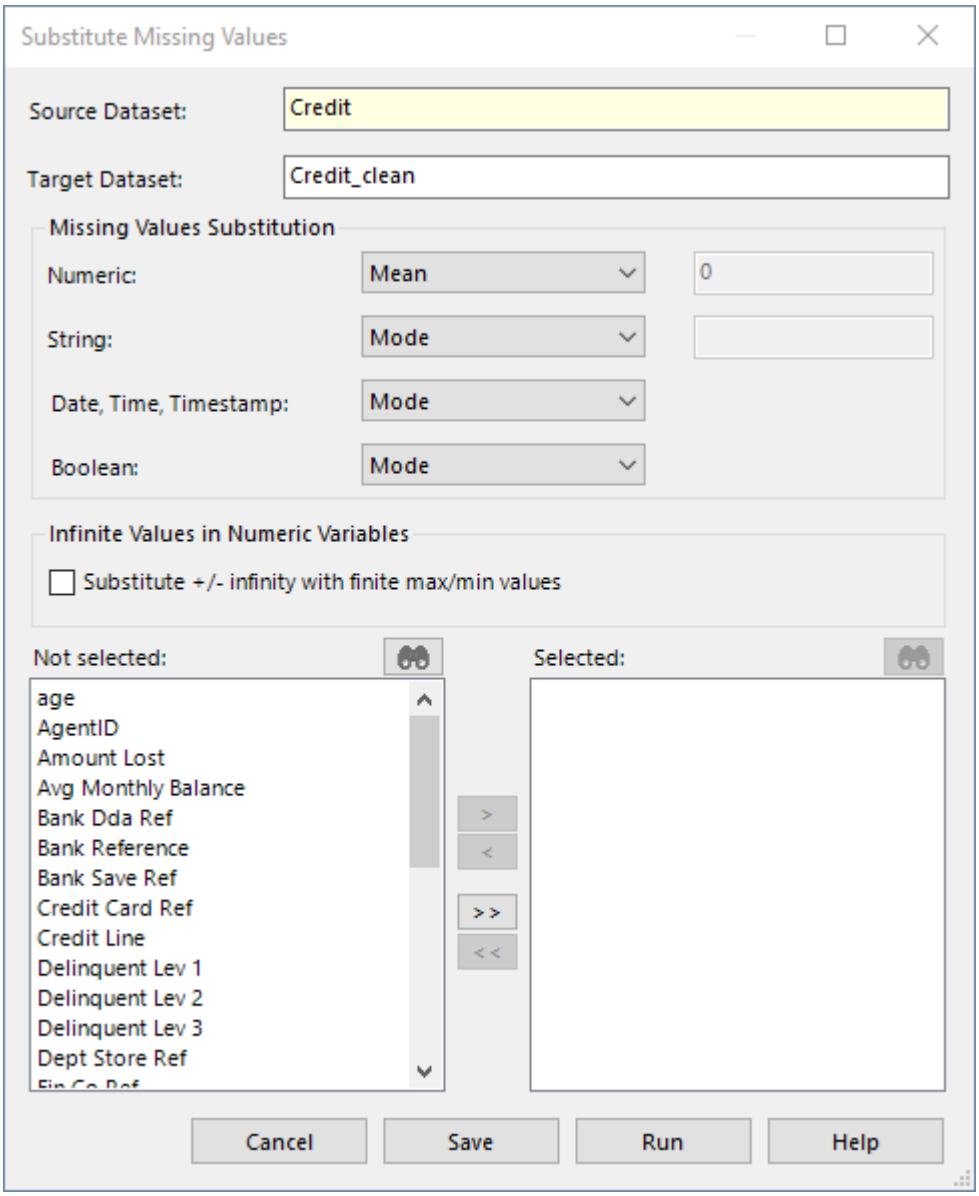


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

In the wizard page:

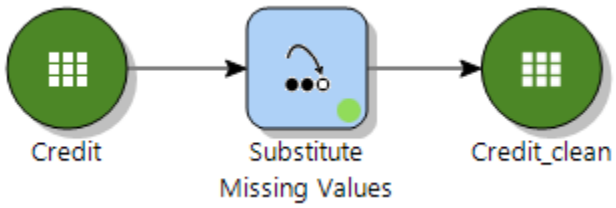
- Enter the name for the output dataset in the *Target Dataset* field.
- Choose the desired option for substituting missing values for each variable type. The following substitution options are available:
    - Numeric variables: Mode. Max, Min, Mean, Median, Zero, or Custom Value. The default is Mean.
        - If you choose *Custom Value*, the field on the right becomes editable. Enter the desired numeric value*. Note: When substituting the custom value in Integer variables. a non-integer value will be rounded to the nearest integer.*
    - String variables: Mode. Max, Min, or Custom String Value. The default is Mode.
        - If you choose *Custom String Value*, the field on the right becomes editable. Enter the desired value*.*
    - Date, Time, and Timestamp variables: Mode. Max, Min, Mean, Median. The default is Mode.
    - Boolean variables: Mode, TRUE, FALSE. The default is Mode.
- Select the option "*Substitute +/-infinity with finite max/min values*" to substitute the infinite values +{INF} and -{INF} in each of the selected variables with its finite maximum or minimum value, respectively.
- Select the variables to which you want to apply the specified missing value substitutions and infinite value substitution.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



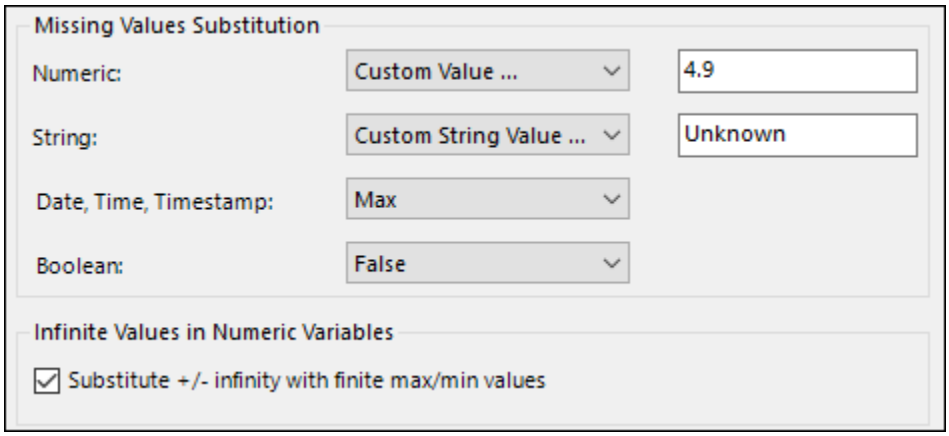
Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. The specified substitutions will be applied to the selected variables depending on their data type.

Double-click the output dataset node to view the results.



Example:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



When the options are specified as above:

- Missing values in all selected variables of the *Number* type will be replaced with 4.9
- Missing values in all selected variables of the *Integer* type will be replaced with 5
- Missing values in all selected String variables will be replaced with the string '*Unknown*'
- Missing values in all selected Date, Time, and Timestamp variables will be replaced with the maximum value of these variables (the most recent Date, Time, or Timestamp value, respectively)
- Missing values in all selected Boolean variables will be replaced with the value FALSE

See also:

Substitute Missing Values Helper

Transform Fields

Handle Class Imbalance

Overview

The **Handle Class Imbalance** node in the **Manipulate** palette balances out the classes of a dependent variable with skewed distribution via oversampling the minority class by creating artificial data points (synthesizing data). This data preparation technique improves the performance of predictive models by helping avoid the situation where model predictions disproportionately favor the majority class (the category with higher frequency).

This node is integrated with Python and uses **`Imbalanced-Learn (imblearn)`** - a popular Python library that provides tools to deal with classification with imbalanced classes when preparing data for modeling. For details, see the **`Imbalanced-Learn`** documentation at https://imbalanced-learn.org/. The node wizard provides a convenient graphical interface to specify the options and parameters for `imblearn`. The wizard automatically translates the inputs into Python code, which can be previewed before running the node. The input dataset is converted from the Altair KDD format to pandas dataframes in the process. The resulting dataframe is converted back to produce the output Altair dataset.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Handling Class Imbalance

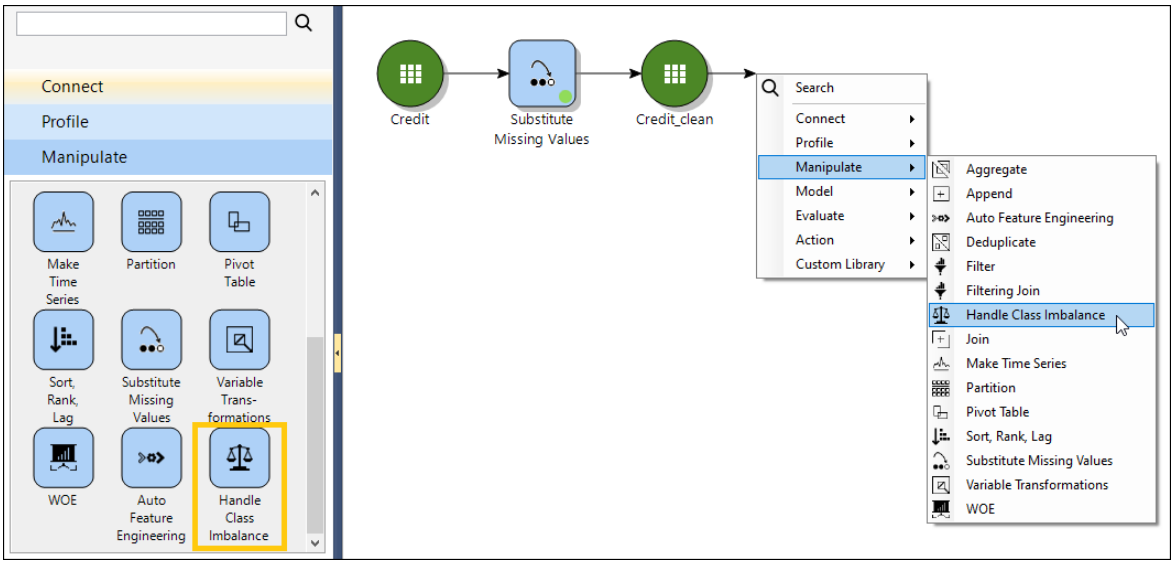
The Handle Class Imbalance node requires a single dataset node as input and produces a single output dataset. It is a good practice to perform missing value substitution before handling class imbalances.

To add a Handle Class Imbalance node to the workflow:

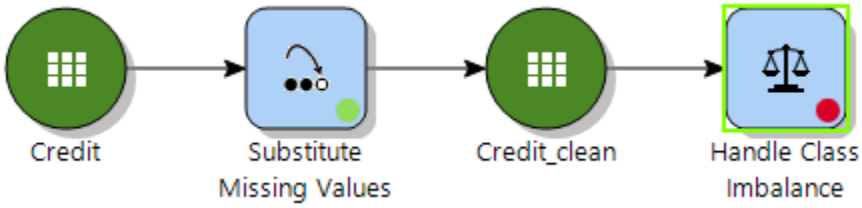
- Drag a connection arrow from the input dataset to any area on the workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Manipulate --> Handle Class Imbalance**, as shown below. A new **Handle Class Imbalance** node will be created, and the wizard will open.

OR:

- Drag the **Handle Class Imbalance** node from the **Manipulate** palette onto the canvas and connect the input dataset to it.



If the **Handle Class Imbalance** node is already on the canvas, double-click it or right-click on it and select **Modify** from the context menu to open the wizard.



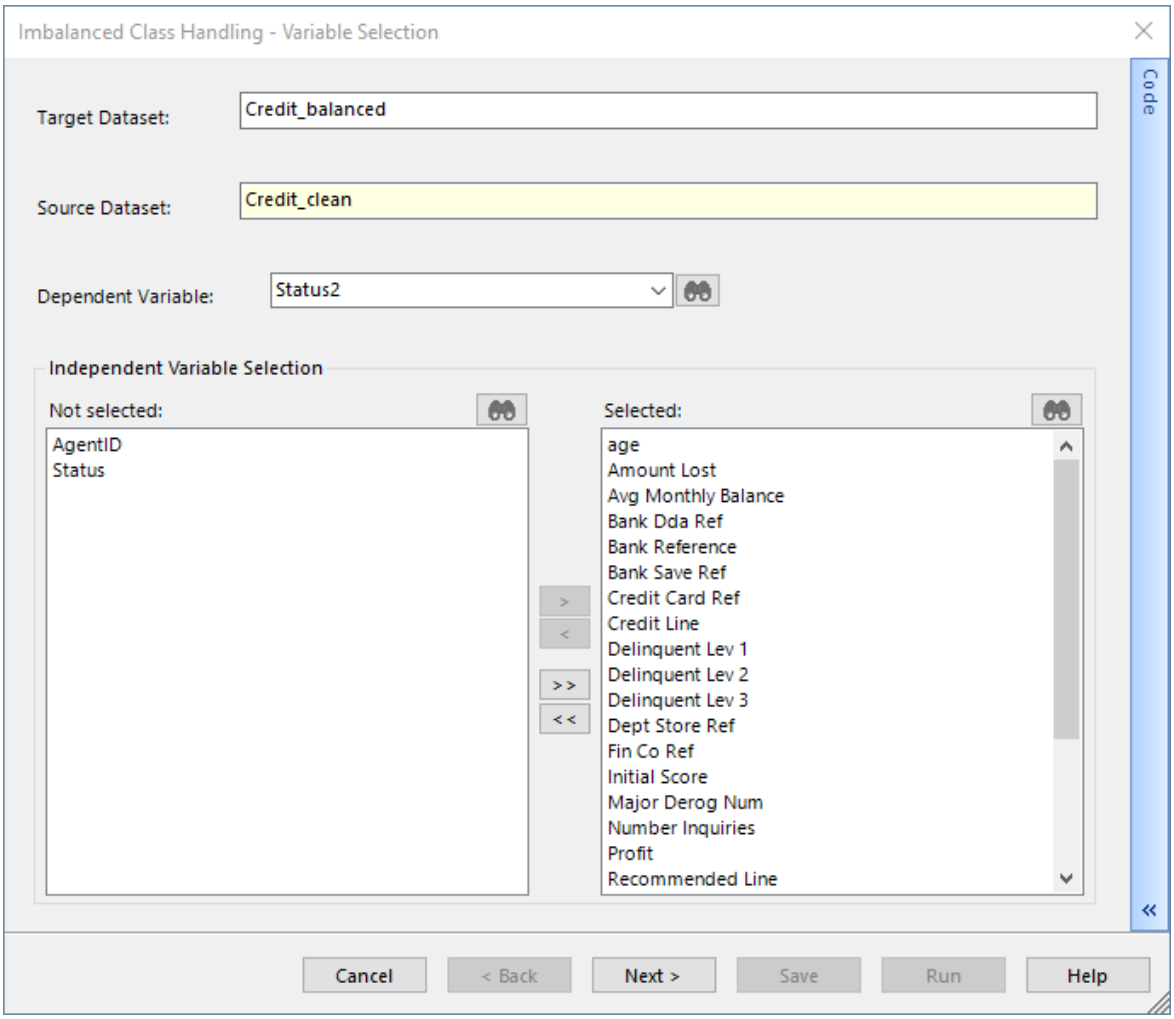
In the first page of the wizard:

- Specify the name for the output dataset in the *Target Dataset* field
- Select the dependent variable whose classes need to be balanced

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Select the desired independent variables

Tip: Advanced field finding and multi-selection capabilities are also available via the Field Finder dialog, which can be opened by clicking the binoculars button.



Click **Next**.

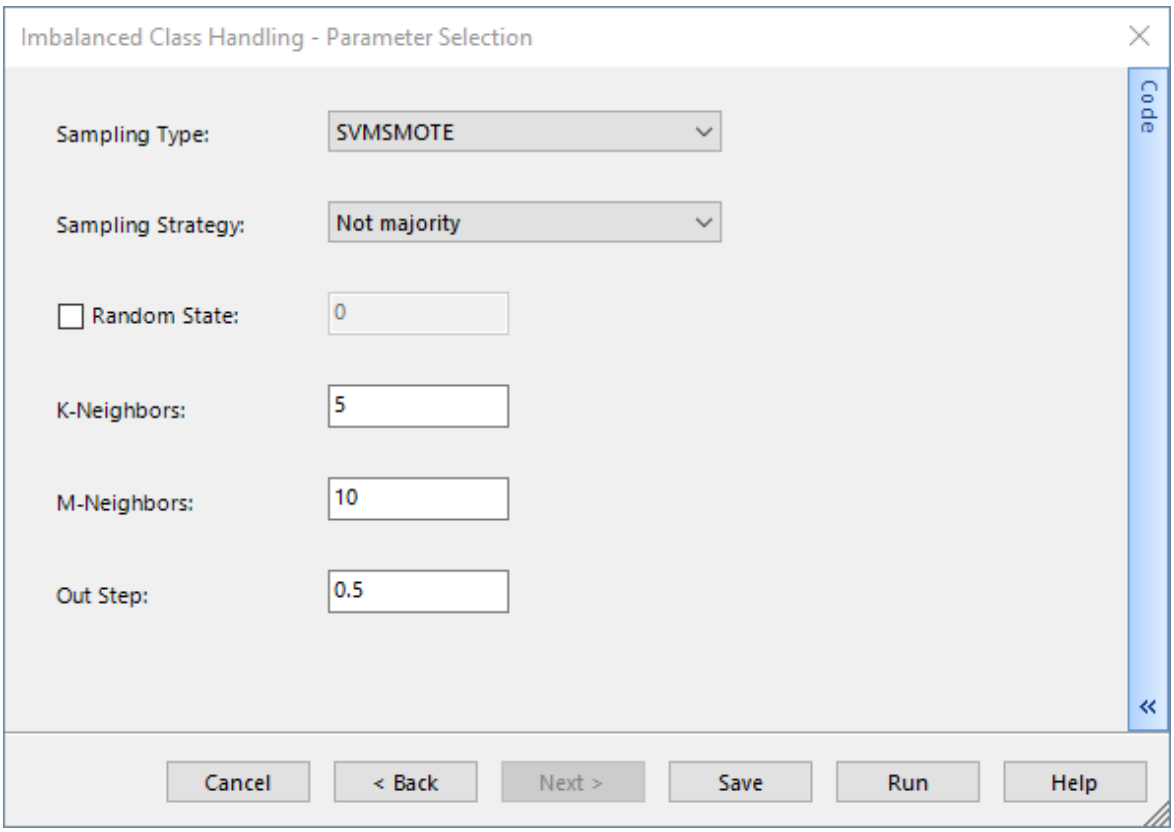
In the Parameter Selection page:

- Select the **Sampling Type**. The available sampling types depend on the selected independent variables.
    - If all of the selected independent variables are continuous, then the applicable Sampling Types are SMOTE, ADASYN, BorderlineSMOTE, KMeansSMOTE, and SVMSMOTE.
    - If the selected independent variables are of mixed types (nominal and continuous), then the only available sampling type is SMOTENC.
- Select the **Sampling Strategy**. The available sampling strategies depend on the selected dependent variable and independent variables.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

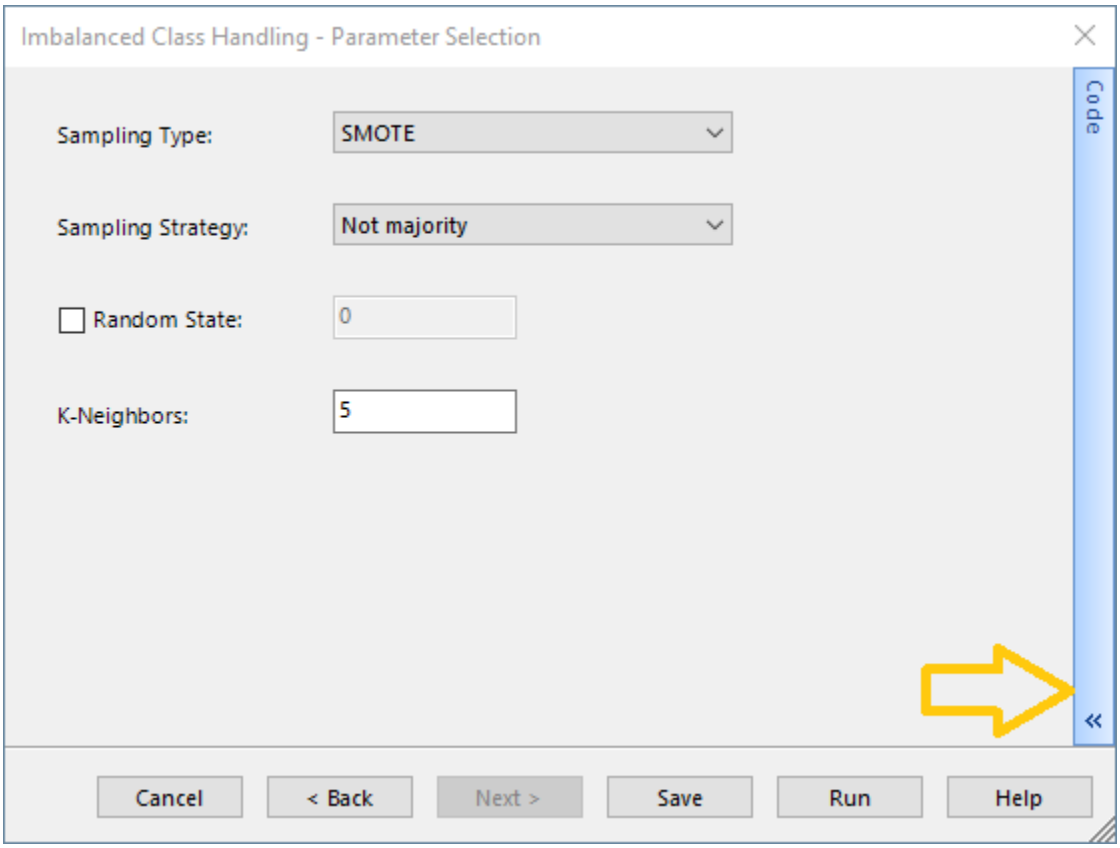
    - If the dependent variable is binary and all of the selected independent variables are continuous, then the only available Sampling Strategy is "Not majority".
    - Otherwise, three Sampling Strategies are available: "Not majority", "Not minority", and "Minority".
- Specify the **Random State** if necessary.
- Specify the **K-Neighbors** value. The default is 5.
- If the selected Sampling Type is BorderlineSMOTE or SVMSMOTE, specify the number of **M-Neighbors**. The default is 10.
- If the selected Sampling Type is SVMSMOTE, specify the **Out Step** value. The default is 0.5.

See the **`Imbalanced-Learn`** documentation at https://imbalanced-learn.org/ for the description of the sampling types (SMOTE, SMOTENC, and others) sampling strategies, and their parameters.



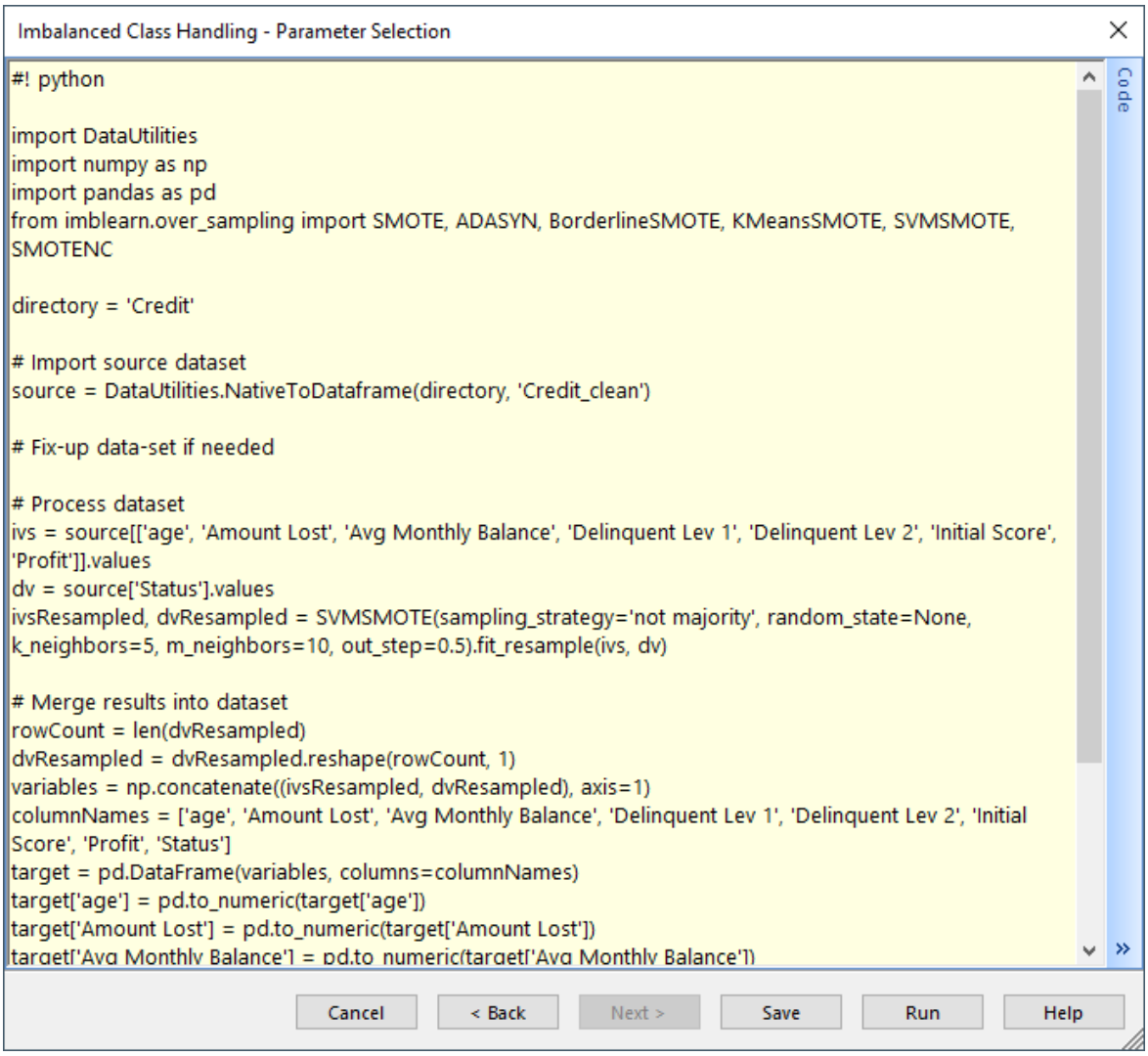
To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



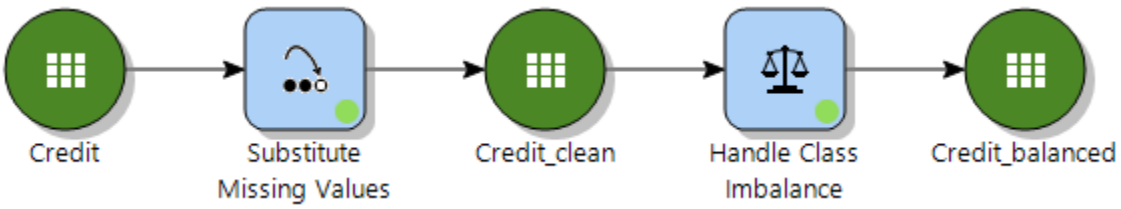
This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. The code generated from your inputs runs in a Python session. The input datasets are converted to Pandas dataframes, and the functions of the `imblearn` library are applied. The output data is converted from Pandas to the Altair KDD format and is shown as the output dataset node at the end.

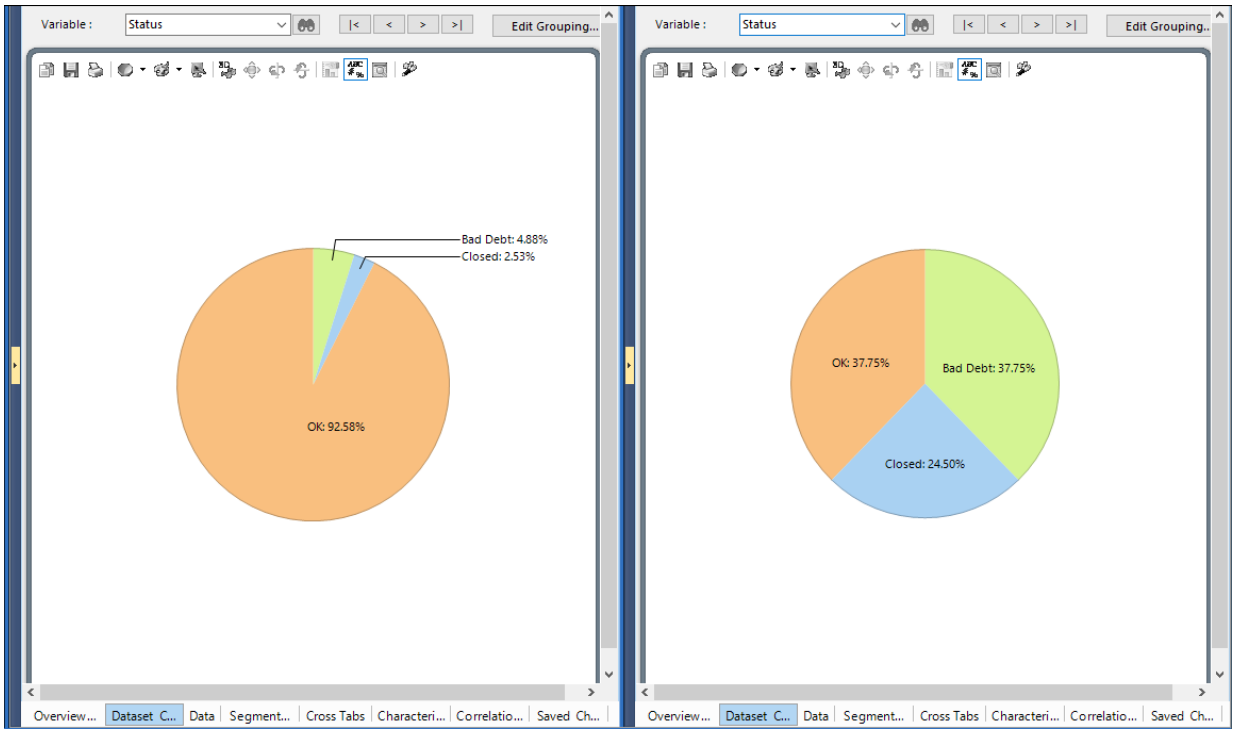
Double-click the output dataset node to view the results.



Select the dependent variable in the Overview Report and click the Dataset Chart tab to view the new distribution.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Here is an example comparing the dependent variable distribution before and after oversampling using the **Handle Class Imbalance** node:



See also:

- Substitute Missing Values
- Outlier Detector
- Auto Feature Engineering

WOE Transformation

Optimal Binning with Weight of Evidence transformations allows you to create transformations that group each variable into new categories that have the highest predictive power with respect to a specified dependent variable. This operation allows you to transform many fields at the same time rather than doing it on a field by field basis.

Optimal binning is used for:

- Reducing the cardinality of discrete independent variables in order to improve their predictive power
- Coarse classing of independent variables as part of a scorecard development process. For more information on creating scorecards, see the section "Building Scorecards" in the Tutorial, which is available under the Help menu.
- Conditioning the data in a way that improves the performance of linear and logistic regression models

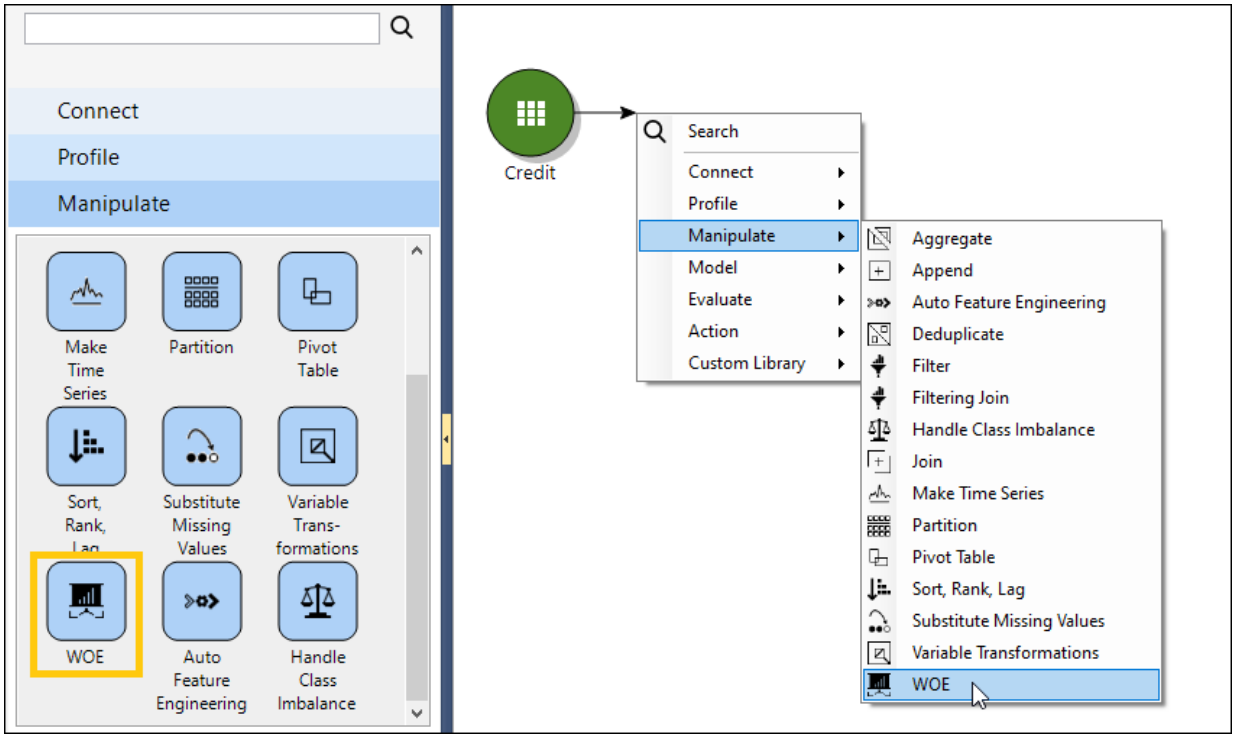
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To add a Weight of Evidence Transformation:

- Drag a connection arrow from the dataset node to any area on the Workflow canvas background. Releasing the mouse button brings up a context menu allowing you to select the desired operation. From the context menu, select **Manipulate --> WOE** as in the example screenshot below. A new **WOE** process node will be created.

OR:

- Drag the **WOE** node from the **Manipulate** palette onto the canvas and connect the input dataset to the node.

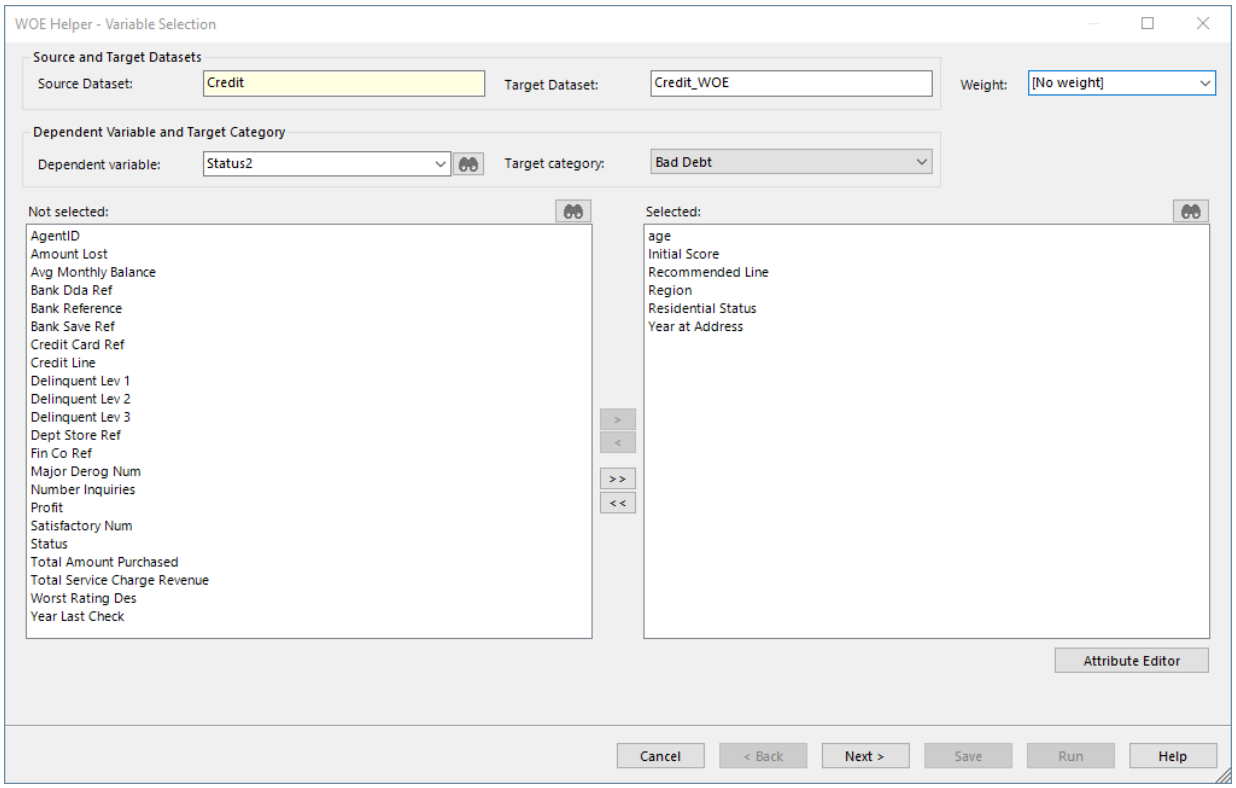




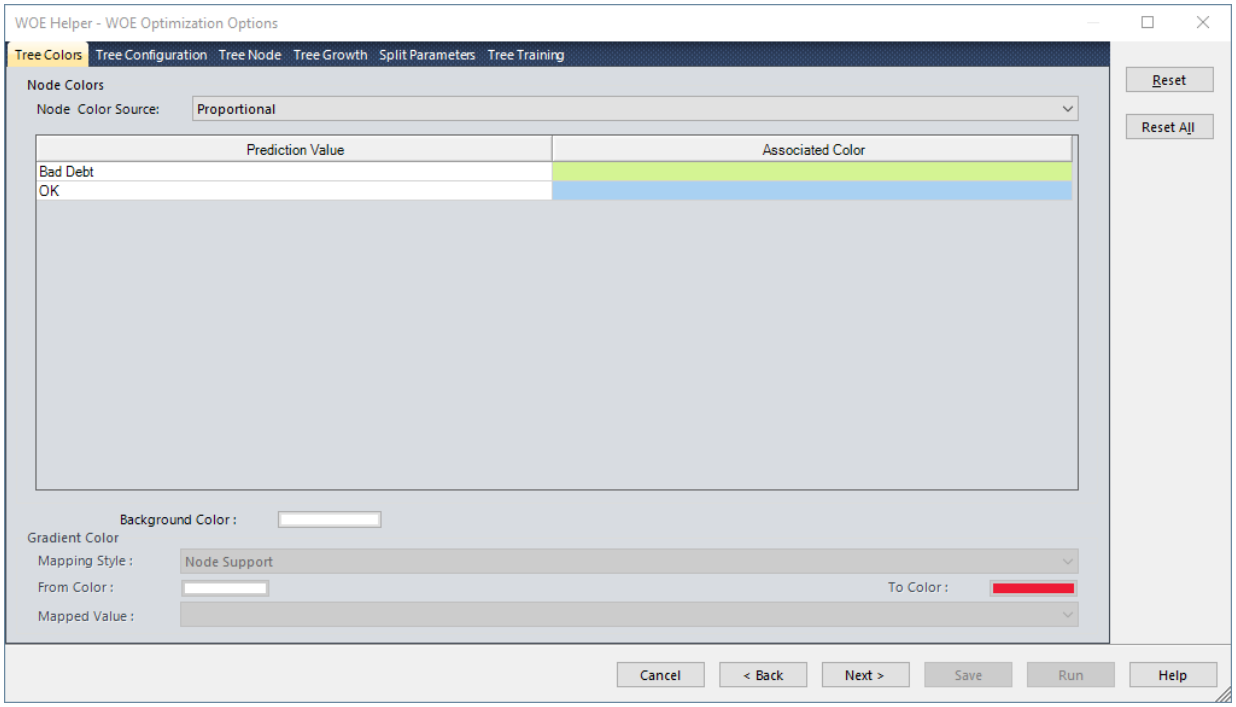
Double-click the **WOE** node or right-click on it and select **Modify** to open the wizard.

The first step of the helper is to select the dependent variable and the target category, and select the independent variables. Only binary variables are available for selection as *Dependent variable*.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

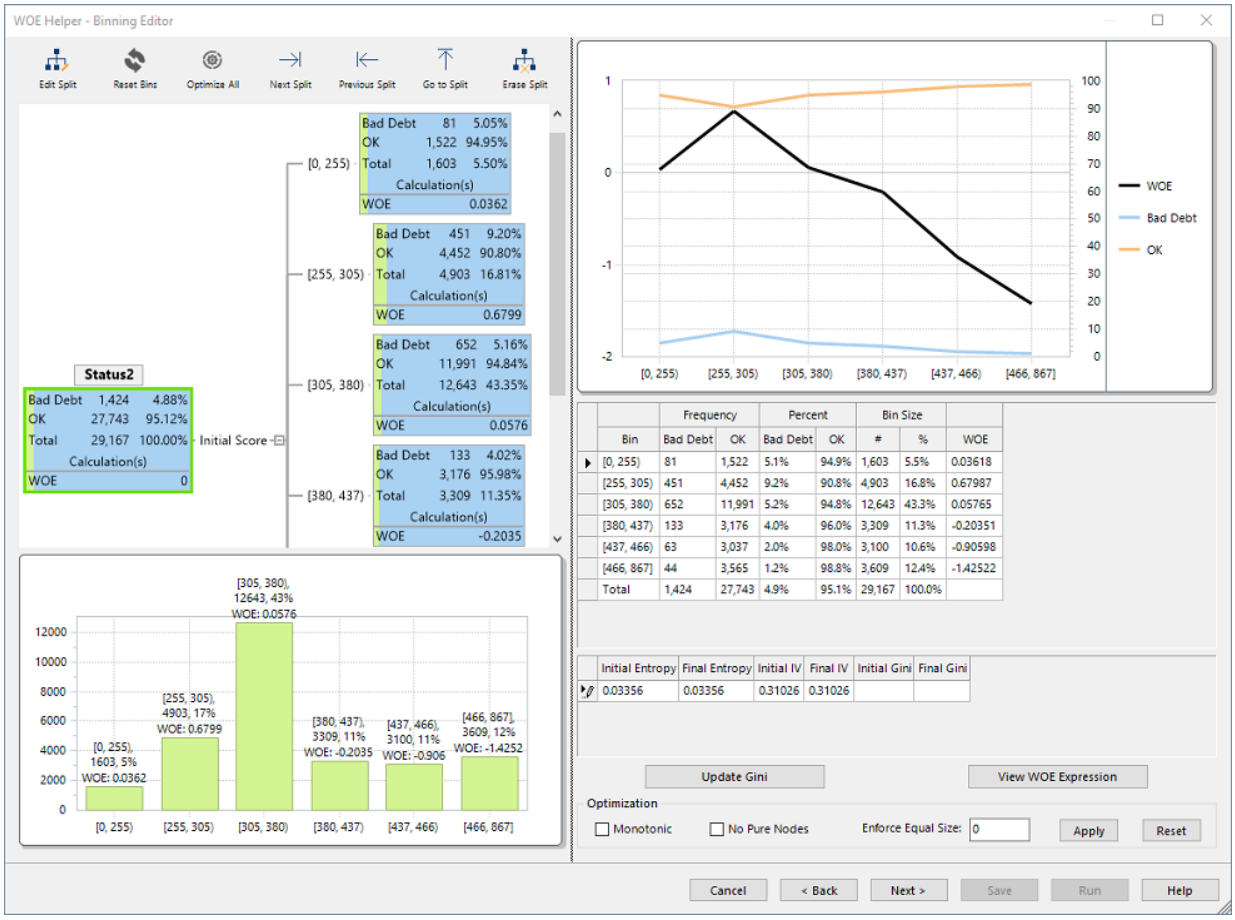


At the next step, specify the decision tree options. These are the same options you specify when inserting a decision tree. See the topic Inserting a Decision Tree for details.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The Binning Editor helps you to view or edit the binning on a variable by variable basis. This dialog represents the bins in a tree form and allows you to browse through the list of variables, remove unwanted variables and modify the binning.



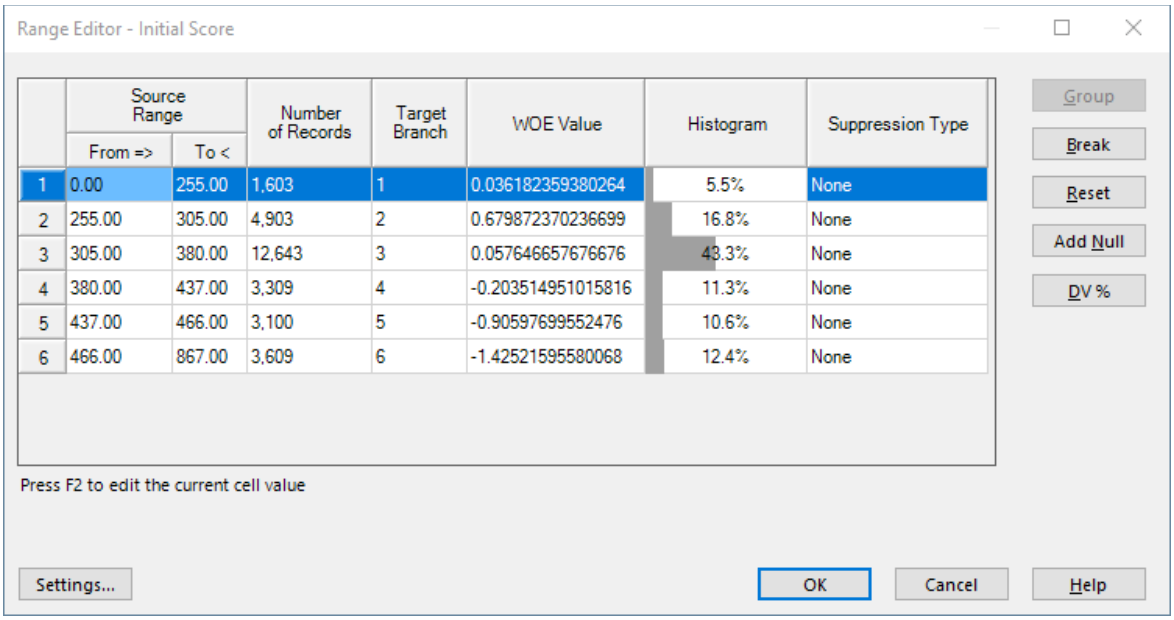
Edit Split	Allows you to manually modify the bins of the currently displayed split variable using the **Range Editor.** This is an enhanced version of the Edit Split command in a Decision Tree. See below for details.
Reset Bins	Recalculates the bins for all variables, effectively removing any manual edits you may have made previously in the current WOE Editor session. This is equivalent to the **Find Split** command in a decision tree.
Optimize All	Performs advanced binning optimizations for Weight of Evidence. See the details below.
Next Split ***Previous Split*** ***Go to Split***	These commands allow you to navigate between the splits in order to view the binning for different variables. See the corresponding Decision Tree topics.
Erase Split	Erases the current split. If you wish to include the variable again, you can do this by right-clicking on the tree background and selecting the **Attribute Editor** command.
Right-click | Attribute Editor	This command allows you to control the detailed behavior of the binning for each of the variables. See the topic Using the Attribute Editor for details.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

View WOE Expression	This command shows the SQL expression used to compute the Weight of Evidence in each node.

Edit Split

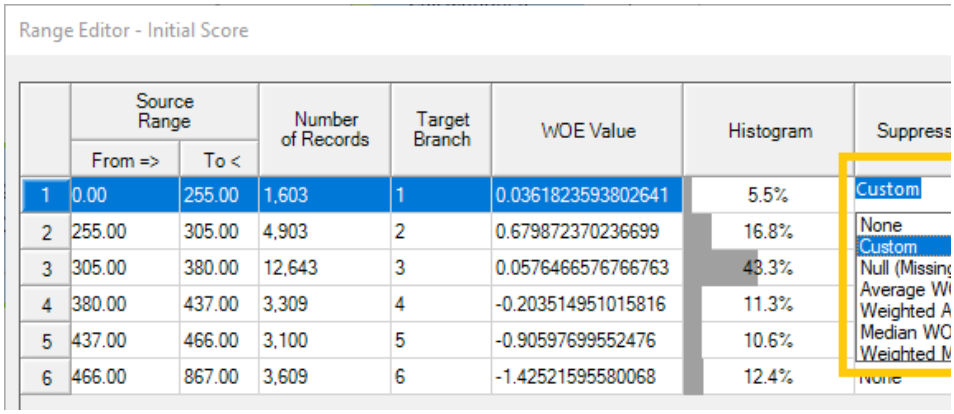
Editing splits in the WOE Binning Editor is like editing splits in a normal decision tree, with a few additional features specific for Weight of Evidence. See the Edit Split topic for the description of common Edit Split features.



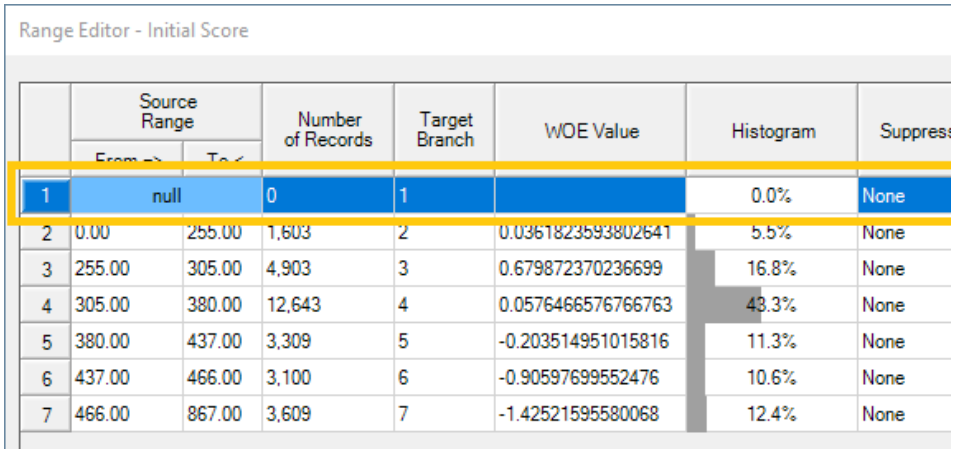
The following Range Editor features are specific to the WOE Binning Editor:

WOE Value	The WOE values displayed in the Range Editor for each bin / interval are calculated by the formula that can be viewed by clicking the button **View WOE Expression** in the Binning Editor window. You can suppress the WOE calculation for any bin and specify the desired value if necessary. By default, WOE Values are not editable (*Suppression Type* is *None*). For different Suppression Type options, see below.
Suppression Type	This option allows you to suppress the WOE calculation for any bin and either specify a custom value, substitute null (missing value), or automatically calculate and substitute the average or median WOE value. For example, to specify a custom WOE value, click in the *Suppression Type* column for the selected bin and select '*Custom*' from the drop-down list. The WOE Value field for this bin becomes editable. Click in the WOE Value field and enter the desired value.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

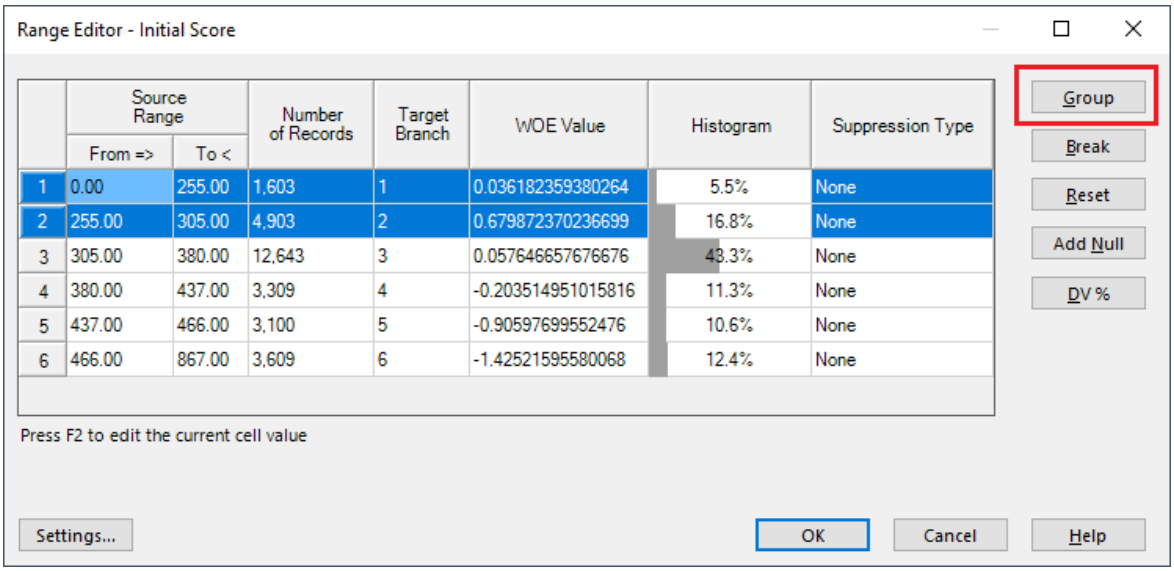


Add Null If the variable being transformed has no missing values, you may want to specify a custom WOE value for the case of missing values in the input data for models that will use this WOE variable. The **Add Null** button inserts a new row with the null value in the Source Range column. The Number of Records for the added null value will naturally be 0. After that you can select a Suppression Type (see above) and specify the desired WOE value.

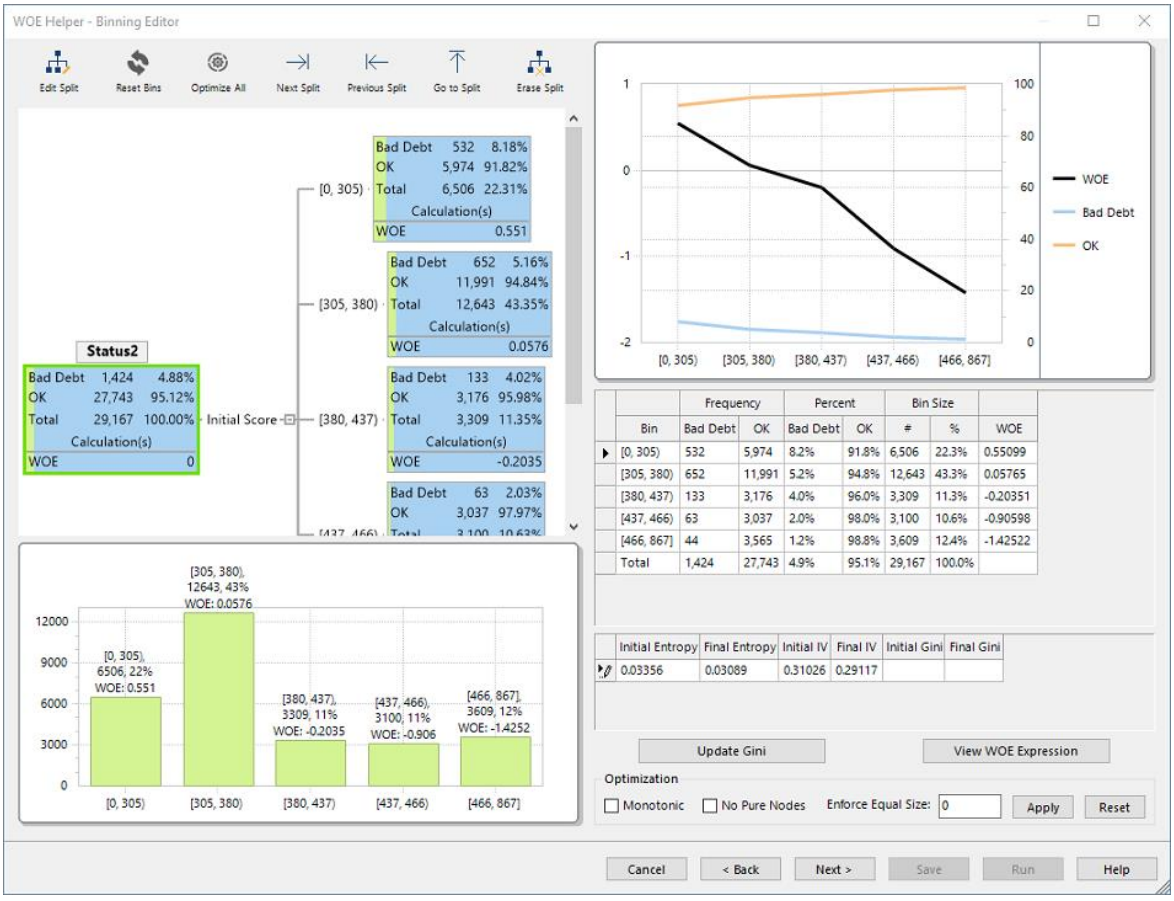


In the context of building a scorecard, you may want to ensure that the Weight of Evidence have a logical trend. For example, consider the split in the screenshot above. Here, the lowest bureau scores in the interval [0, 255] are not the worst group (bin) in terms of the Bad Debt percentage and WOE values. The expectation is that the higher the bureau score, the lower the risk. Using the **Edit Split** command, you can adjust the ranges so that the dependence is monotonic.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



After grouping the ranges [0,255) and [255,305) as shown in the screenshot above, we obtain a monotonic trend.



Ensuring that the trends are logical has a variety of benefits for building scorecard models and other purposes:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- The model is more generalized and reduces the “overfit” phenomena. Rather than the model representing quirkiness found only in the training dataset, the scorecard should be robust and have a longer shelf.
- The model will map to general business experience so there will be less resistance to adapting the scorecard in business processes.

Advanced Binning Optimizations for Weight of Evidence

The advanced optimizations feature of the Optimal Binning and WOE Editor allows you to further optimize the binning:

- Ensure the monotonicity of Weight of Evidence, and therefore the independent variable distribution trend across the bins.
- Prevent the creation of bins with null information value - "pure" bins with zero percentage of one of the target values. For details on the information value, see Mathematical Formulations: Decision Tree.
- Produce bins of equal or approximately equal size whenever possible.

To perform these optimizations on a single variable (the one currently displayed in the WOE Editor), choose the desired options: *Monotonic*, *No Pure Nodes*, and specify the degree of enforcing equal size of bins (0 - no enforcing; 1- maximum possible enforcement). Click **Apply**.

To perform these optimizations on all variables, click the **Optimize All** button in the toolbar.

To cancel advanced optimizations for the currently displayed variable, click **Reset**.

To cancel advanced optimizations for all variables, click the **Reset Bins** button in the toolbar.

The initial and final values of *Entropy Ratio* and *Information Value* (before and after optimization) are displayed in the table below the WOE chart in order to assess how the WOE transformations affected your variables. Here IV means "Information Value".

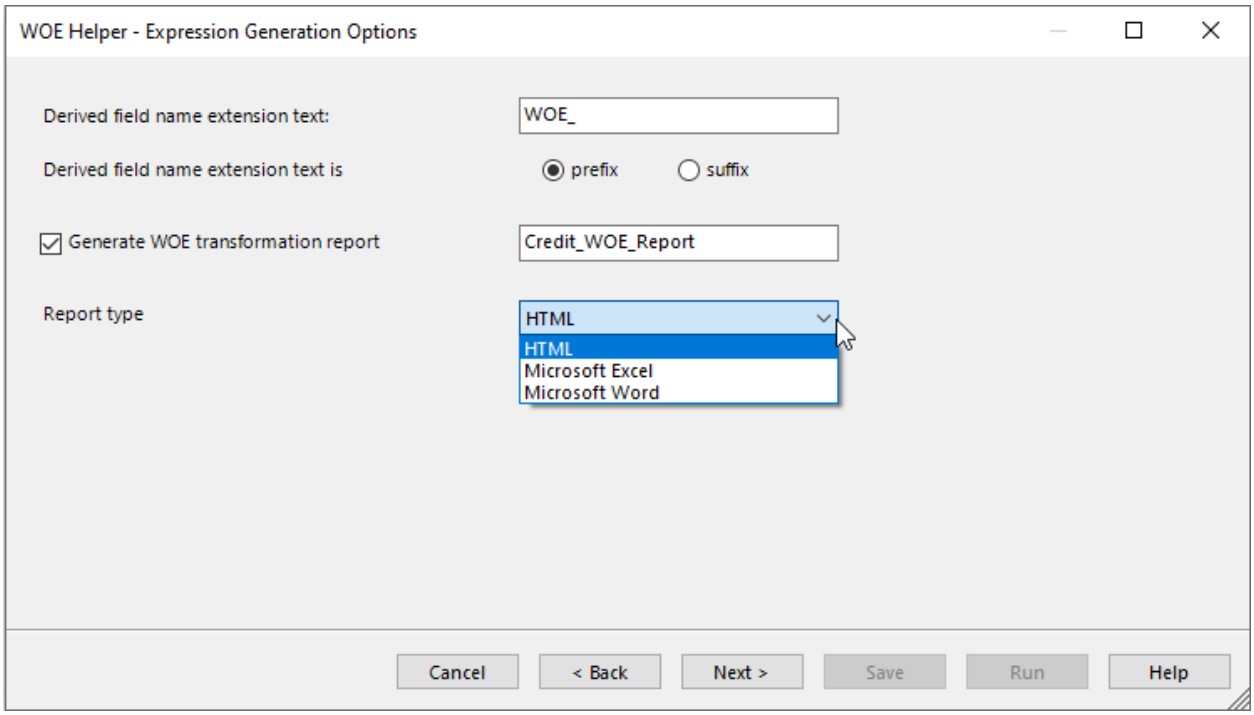
	Initial Entropy	Final Entropy	Initial IV	Final IV	Initial Gini	Final Gini
▶	0.02038	0.0187	0.18833	0.16808	0.98	0.98

The formula for Information Value can be found in the topic Mathematical Formulations: Decision Tree.

Expression Generation Options

When you are done editing the bins, click **Next**. This will bring you to the Expression Generation Options page.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



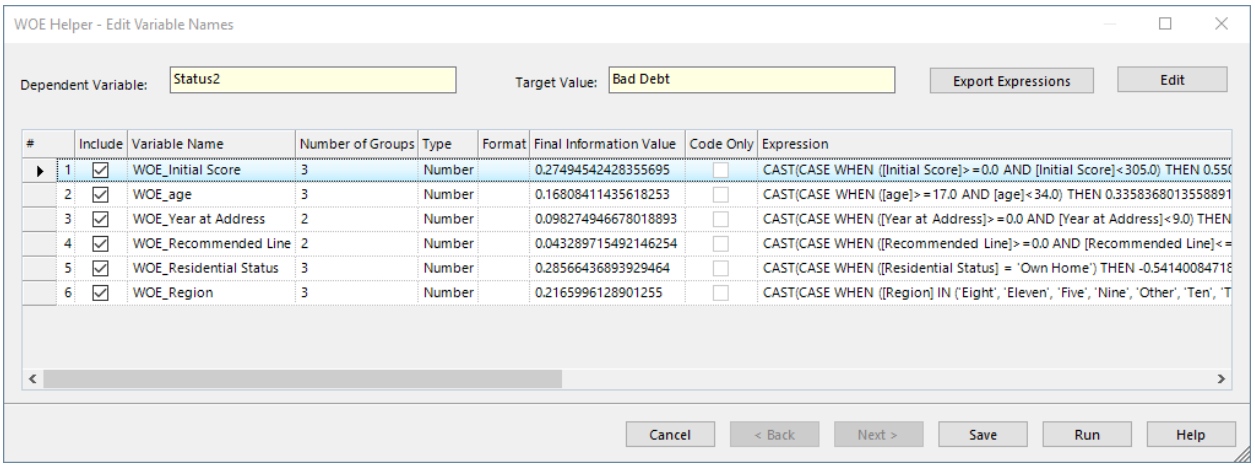
Derived field name extension text	The text entered here will be added to the original variable names as a prefix or suffix to form the names of the derived variables produced by this helper. The default prefix is "*WOE_*".
Prefix / Suffix	If the *Prefix* option is selected, then the derived variable names are formed by adding the specified extension text at the begging of the original field names. If the *Suffix* option is selected, the extension text will be appended at the end of the original field name. The default option is Prefix. Using prefix as opposed to suffix can be more useful in many cases, as the derived fields can be conveniently identified and grouped by sorting by field name.
Generate WOE transformation report	Select this option if you want to generate a WOE transformation report. Specify the name for the report and select the report type: ***HTML***, ***Microsoft Excel***, or ***Microsoft Word***. The default is ***HTML***. If this option is selected, in addition to the output dataset, a new report node will be created on the workflow. The report contains WOE binning information for every variable in the form of tables and WOE charts. An example report fragment is shown below.

Click **Next** to generate the WOE binning expressions.

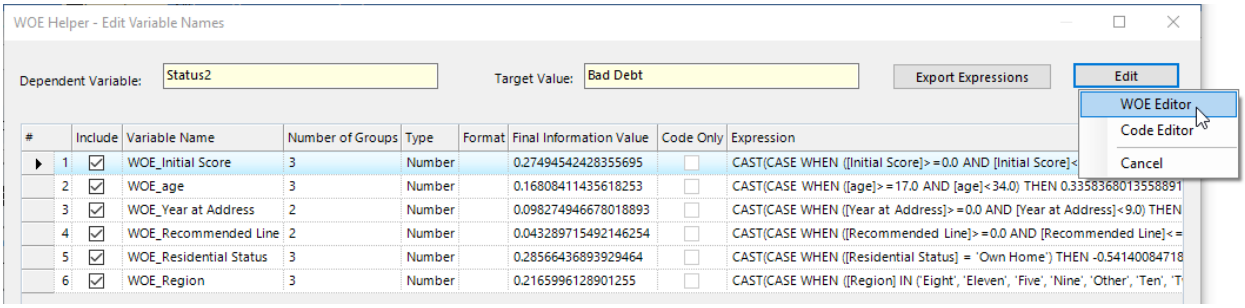
Review and Edit WOE Expressions

The resulting expressions are displayed in a grid for review. In this page you may exclude some variables, edit the variable names, and even edit expressions. The Information Value ("*Final Information Value*") is displayed for each WOE variable, if defined.

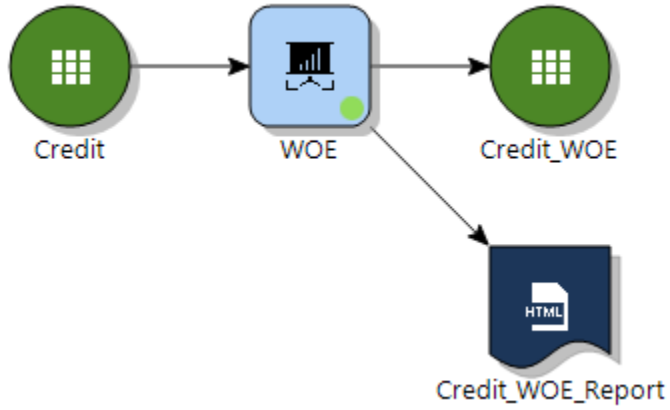
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



After the expressions have been generated, and even after running the node, you can modify the binning using the same visual interface based on the decision tree framework. Just double-click the WOE node, click **Edit**, and select **WOE Editor**.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. Double-click the output dataset node to view the results.

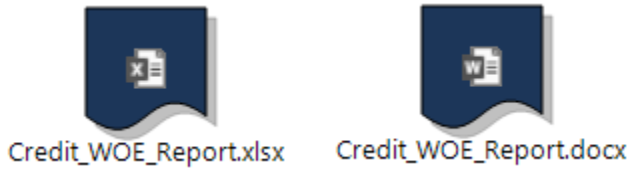


- If you need to edit the WOE expressions at any time after running the node, double-click on it or right-click on it and select **Open View**.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- To add or remove variables and rebuild the WOE transformation, right-click on the WOE node and select **Modify**.

If you chose to generate the WOE Transformation Report, double-click on the report node to view the report or right-click on it and select **Open View**. If you chose the *Microsoft Excel* or *Microsoft Word* type, the report node will have the corresponding application icon.



The report will be opened in the associated MS Office application (*Microsoft Excel* or *Microsoft Word*) if it is installed on your computer.

Example WOE Transformation Report

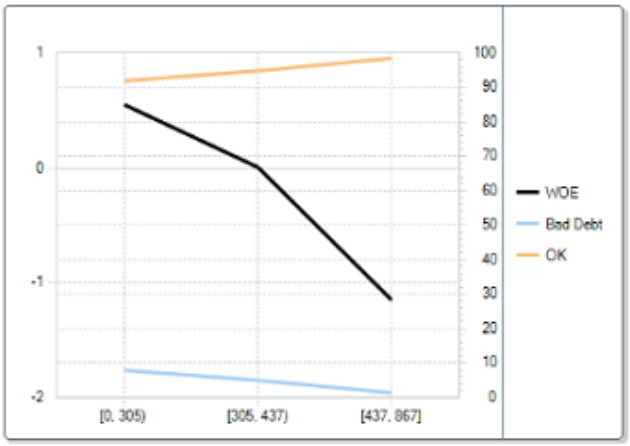
A fragment of a WOE Transformation Report (WOE binning information for the *Initial Score* variable is shown)

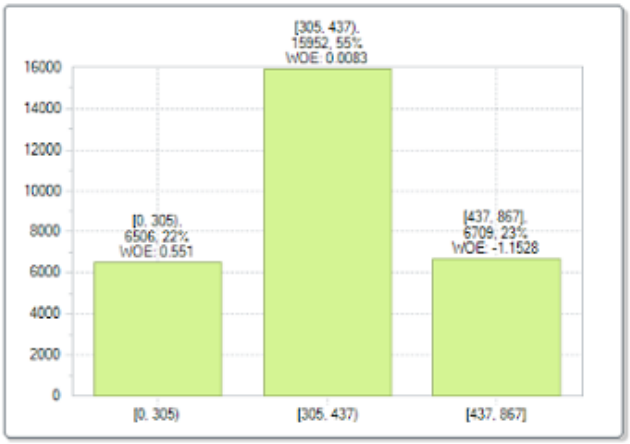
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

WOE_Initial Score (Derived from Initial Score)

	Frequency		Percent		Bin Size		
Bin	Bad Debt	OK	Bad Debt	OK	#	%	WOE
[0, 305)	532	5,974	8.2%	91.8%	6,506	22.3%	0.55099
[305, 437)	785	15,167	4.9%	95.1%	15,952	54.7%	0.00832
[437, 867]	107	6,602	1.6%	98.4%	6,709	23.0%	-1.15279
Total	1,424	27,743	4.9%	95.1%	29,167	100%	

Initial Entropy	Final Entropy	Initial IV	Final IV	Initial Gini	Final Gini
0.03356	0.0296	0.31026	0.27495	0.98	0.98





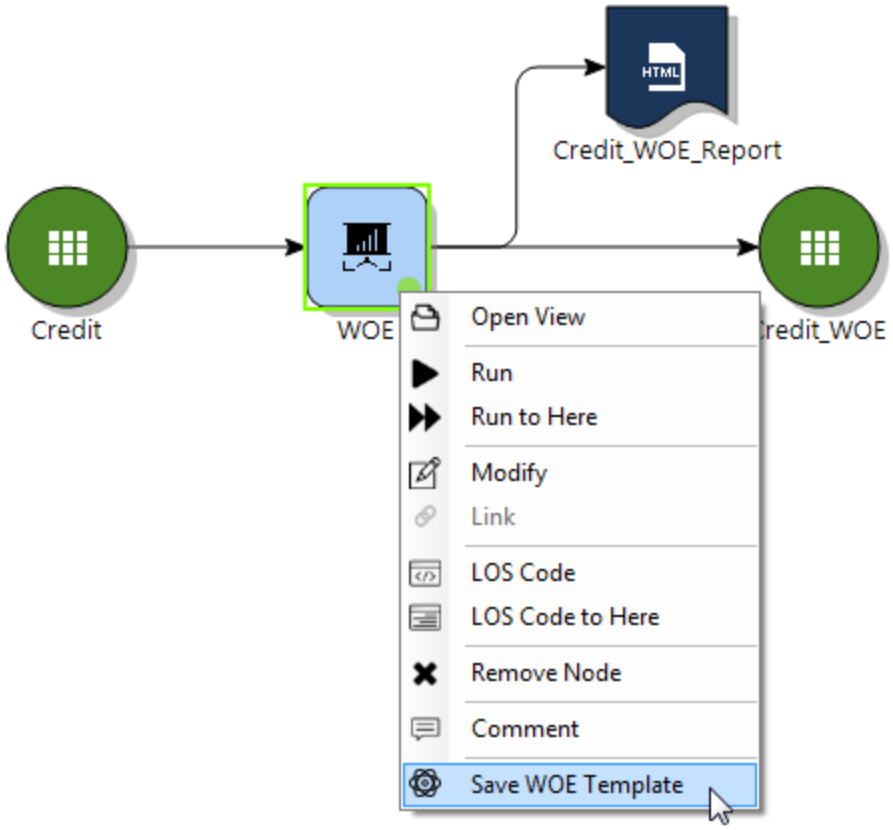
See also: Helper: Optimal Binning

WOE Template

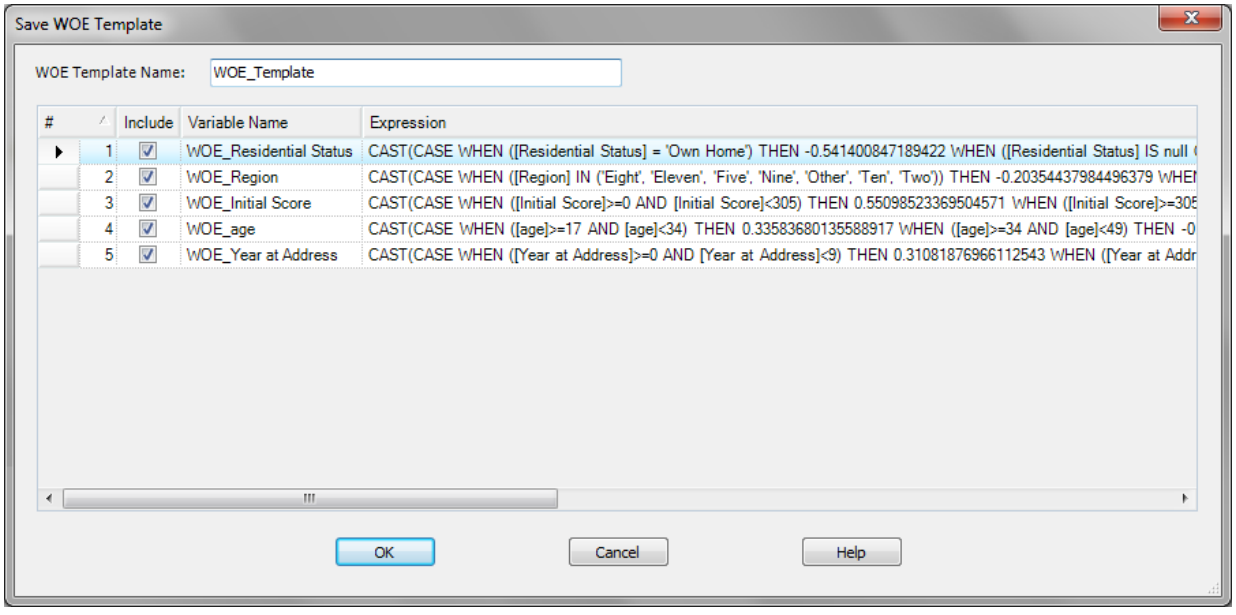
The **WOE Template** node allows you to save Weight of Evidence transformations created in the WOE node and apply these transformations to new WOE nodes.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The WOE Template is created by right-clicking a **WOE** node and selecting the command "**Save WOE Template**" from the context menu as shown below:

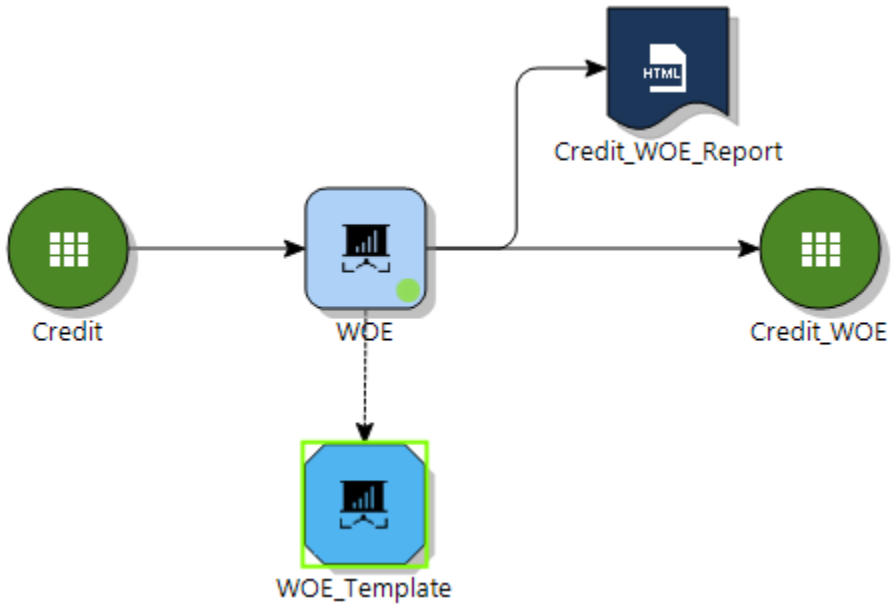


In the Save WOE Template dialog, specify the template name and select the WOE expressions to include in the template.



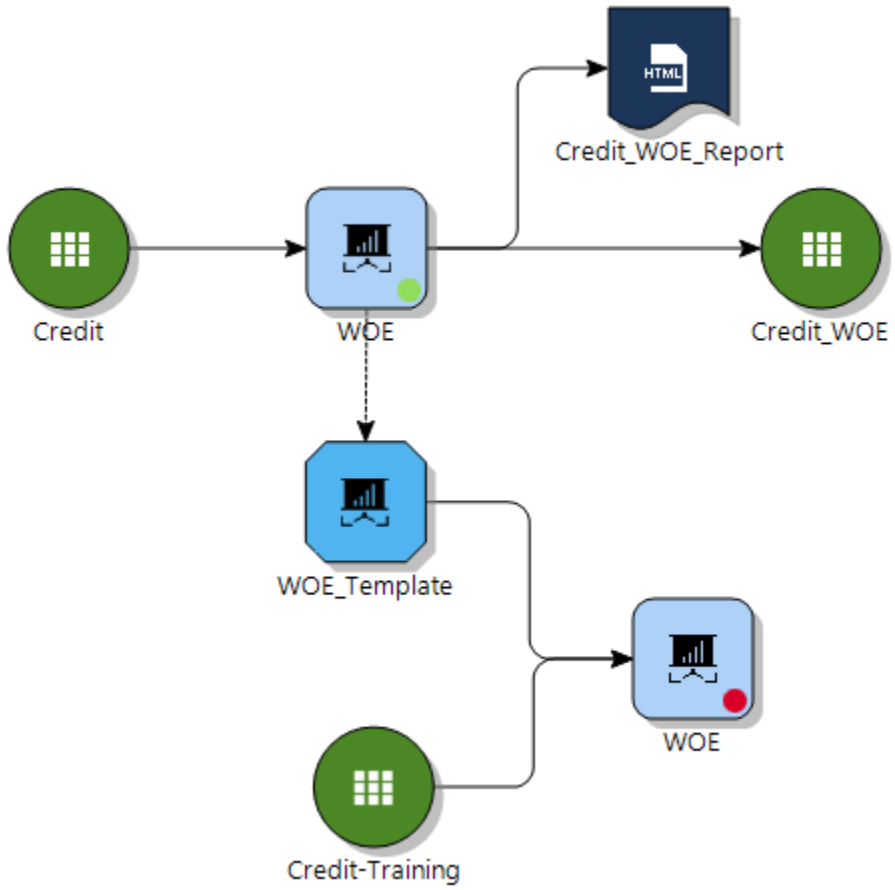
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The newly created WOE Template node is connected to the WOE node with a dashed line - the same type of connection line as Model Instances. This means that the template node is never overwritten when re-running the workflow or the parent WOE node. Also, you can disconnect the template node from the parent WOE node at any time.



To apply the WOE transformations defined in the WOE Template to a new WOE node, connect it to the WOE node and connect an input dataset to the same WOE node..Then double-click the WOE node to view or modify the expressions.

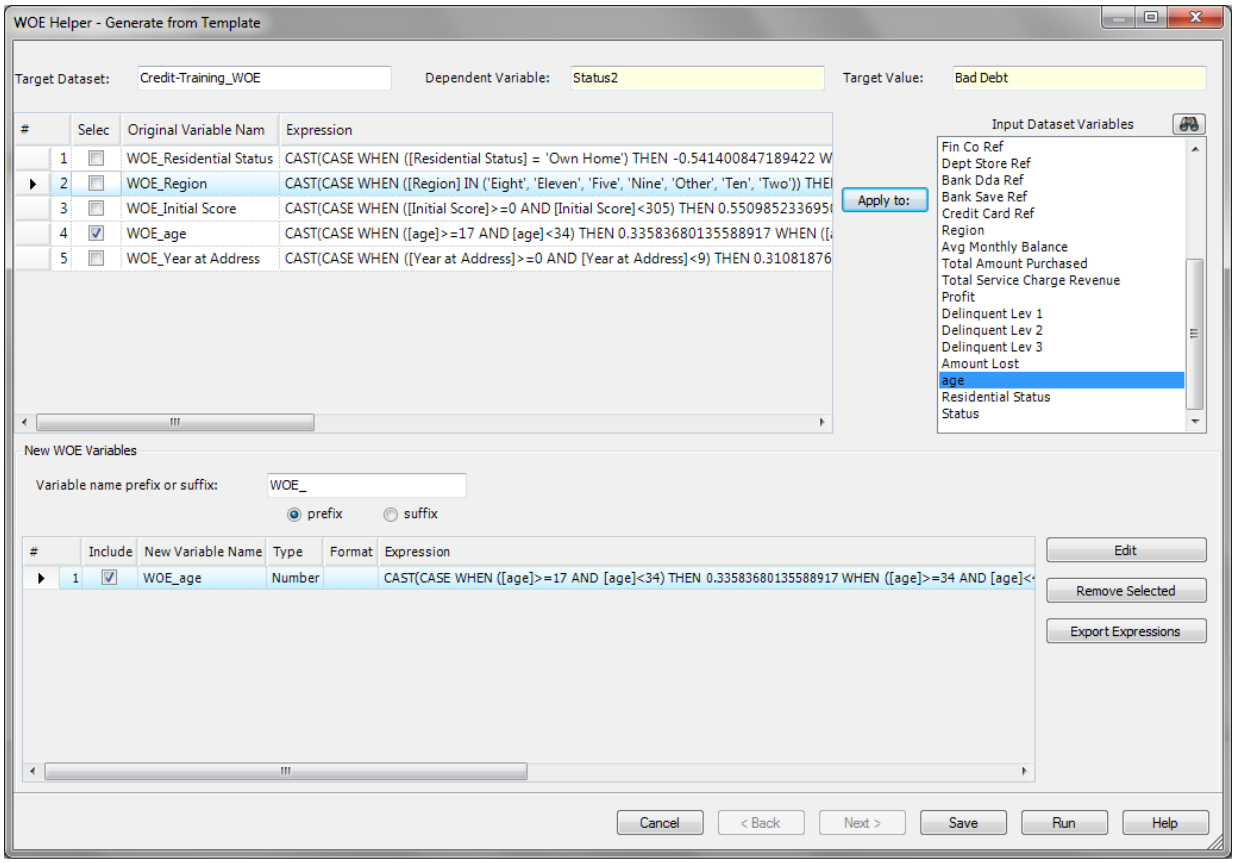
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



When you open a WOE node to which a WOE Template node is connected, it has the list of all WOE expressions in the input template and, on the right, it has the list of all variables from the input dataset.

The "**Apply to**:" button applies the selected WOE expressions on the left to the selected dataset variables on the right just as any function is applied to the arguments. Each selected WOE expression on the left is applied to each of the selected variables on the right.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also: [WOE Transformation]

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Data Profiling

Dataset Views: Summary

To profile your data, open the desired dataset and analyze statistics and charts provided in the dataset views. To navigate to the desired view, click on the corresponding tab at the bottom of the window. The following dataset views are available:

- Overview Report
- The Overview Report summarizes your data with univariate statistics for each variable. You can also set a weight field in Overview Report. The available statistics are grouped into categories: Basic Measures, Measures of Central Tendency, Measures of Dispersion, Quantiles, Extreme Observations, Tests for Normality, Tests for Location, and Confidence Interval for Population Mean.

- Dataset Chart
- The Dataset Chart displays univariate charts that show the distribution of data values for each field.

- Data
- The Data view displays the data in the selected fields of the dataset in the tabular form.

- Segment Viewer
- The Segment Viewer visualizes the distributions of the independent variables under each category of the selected segmentation variable.

- Cross Tabs
- The Cross Tabs view visualizes your data when plotting the categories of one field in relation to the categories of another field, as in pivot tables.

- Characteristic Analysis
- The Characteristic Analysis view allows you to analyze trends of the categories of a dependent variable in both chart and tabular format.

- Correlations
- The Correlations allows you to explore the strength of association between the variables in your dataset. It displays four types of correlation coefficients for the selected fields: Pearson, Spearman’s rho, Kendall's tau-b, and Hoeffding coefficients.

- Saved Charts
- The Saved Charts tab represents the library of charts you saved in other views of the current dataset.

- Interactive Filter
- A dockable filter panel available for all Dataset tabs that provides interactive filtering capabilities on the current dataset. The filter condition is synchronously applied to the data in all dataset tabs, and the filtering can be toggled on or off at any time.

All charts and tables can be copied and pasted into any Microsoft Office application or image editor using the Edit menu commands, or exported to PDF.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Weight

Your data can contain a special numeric field whose values weight the values in the other fields that represent your analysis variables. If the weight field does not exist in the data source to be imported, it can be created using the Variable Transformation node after importing the data.

Weight can be used at the dataset level, as well as in decision trees, strategy trees, predictive and cluster models. All charts and reports take weight into account and display weighted statistics when a weight is defined.

A weight can be interpreted in different ways. The most common weight types are frequency and probability (sample) weights.

- Frequency weights are usually integer numbers that indicate how many observations each record represents. They can be used to compress the data: for example, instead of having N identical records in your data, you can include just one of them and put N in the same row in an additional field called Weight.
- Probability (or sample) weights represent the portion of specific observations selected into the sample from a population. For example, if the population of the customers who responded positively to your marketing campaign is 50000 and you select 20000 of them into your sample, your sampling fraction is 2/5. Then the value of the weight for such customers is the inverse of the sampling fraction = 5/2 = 2.5.

Any numeric field with the following properties can be used as weight in Altair Data Analytics applications:

- Data type is Integer or Number (note: the Number type represents all real values)
- All values are nonnegative
- No missing values

In Altair Data Analytics applications, weight is treated as a generalized frequency weight (since the weight values do not have to be integer), so the total number of records in the weighted dataset is always equal to the sum of the weight field and is not necessarily integer.

Since the weight values don't have to be integer, a weight can be interpreted as a frequency weight or a probability (sample) weight. Note that if you want the displayed total number of records in the sample remain the same after setting a weight, you can always normalize the weight field by multiplying it by *N* / SUM(<weight field>) in the Dataset Editor, where *N* is the actual (unweighted) sample size.

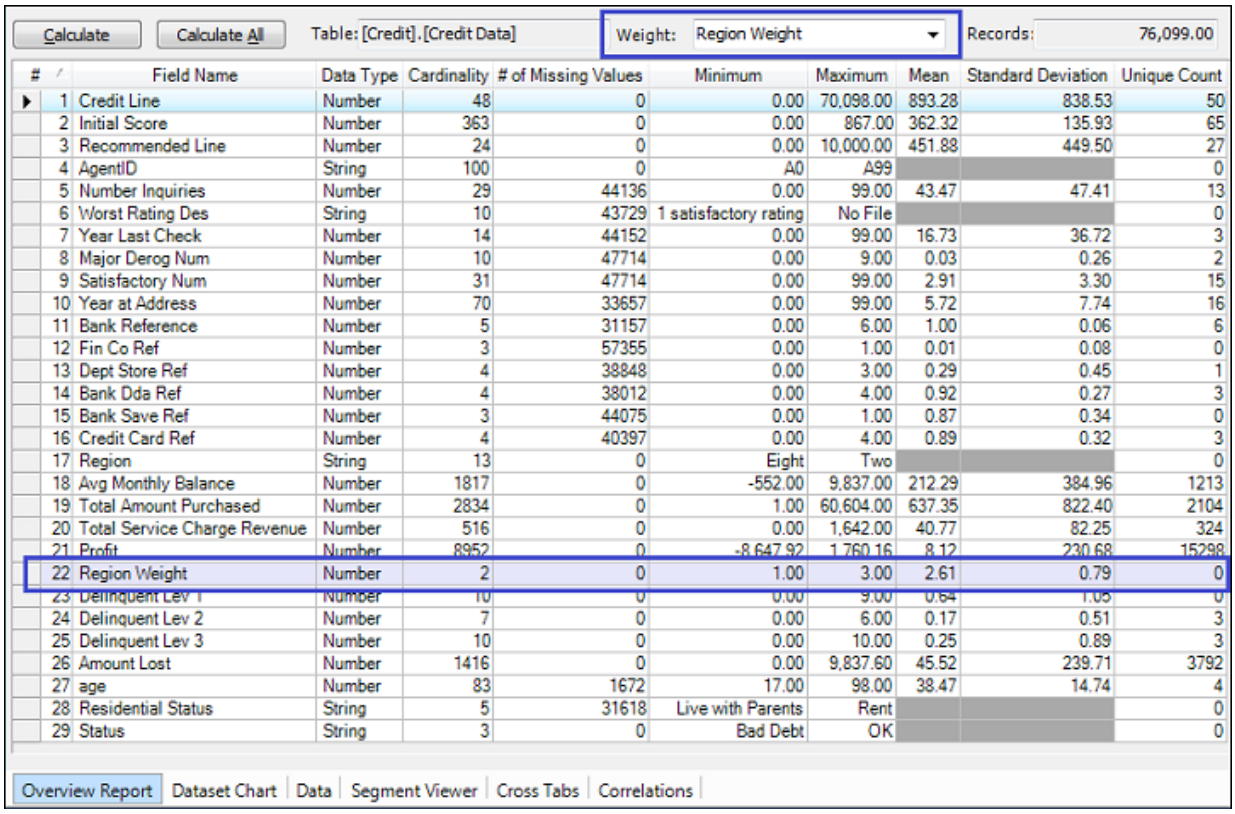
Specifying Weight

If the field you would like to use as weight already exists in your dataset, the weight can be set in Overview Report by selecting the field from the Weight drop-down list (see the screenshot below). Only nonnegative numeric fields with no missing values are available to be set as weight.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If you want to create a weight, you can add it as a calculated column in Dataset Editor. A convenient wizard (Weight Helper) is provided for this purpose. See the Weight Helper topic for details.

When a weight is set, the row in Overview Report representing the weight field is highlighted to distinguish it from the analysis variables.



Calculate | Calculate All | Table: [Credit].[Credit Data] | Weight: Region Weight | Records: 76,099.00

#	Field Name	Data Type	Cardinality	# of Missing Values	Minimum	Maximum	Mean	Standard Deviation	Unique Count
1	Credit Line	Number	48	0	0.00	70,098.00	893.28	838.53	50
2	Initial Score	Number	363	0	0.00	867.00	362.32	135.93	65
3	Recommended Line	Number	24	0	0.00	10,000.00	451.88	449.50	27
4	AgentID	String	100	0	A0	A99			0
5	Number Inquiries	Number	29	44136	0.00	99.00	43.47	47.41	13
6	Worst Rating Des	String	10	43729	1 satisfactory rating	No File			0
7	Year Last Check	Number	14	44152	0.00	99.00	16.73	36.72	3
8	Major Derog Num	Number	10	47714	0.00	9.00	0.03	0.26	2
9	Satisfactory Num	Number	31	47714	0.00	99.00	2.91	3.30	15
10	Year at Address	Number	70	33657	0.00	99.00	5.72	7.74	16
11	Bank Reference	Number	5	31157	0.00	6.00	1.00	0.06	6
12	Fin Co Ref	Number	3	57355	0.00	1.00	0.01	0.08	0
13	Dept Store Ref	Number	4	38848	0.00	3.00	0.29	0.45	1
14	Bank Dda Ref	Number	4	38012	0.00	4.00	0.92	0.27	3
15	Bank Save Ref	Number	3	44075	0.00	1.00	0.87	0.34	0
16	Credit Card Ref	Number	4	40397	0.00	4.00	0.89	0.32	3
17	Region	String	13	0	Eight	Two			0
18	Avg Monthly Balance	Number	1817	0	-552.00	9,837.00	212.29	384.96	1213
19	Total Amount Purchased	Number	2834	0	1.00	60,604.00	637.35	822.40	2104
20	Total Service Charge Revenue	Number	516	0	0.00	1,642.00	40.77	82.25	324
21	Profit	Number	8952	0	-8,647.92	1,760.16	8.12	230.68	15298
22	Region Weight	Number	2	0	1.00	3.00	2.61	0.79	0
23	Delinquent Lev 1	Number	10	0	0.00	9.00	0.64	1.05	0
24	Delinquent Lev 2	Number	7	0	0.00	6.00	0.17	0.51	3
25	Delinquent Lev 3	Number	10	0	0.00	10.00	0.25	0.89	3
26	Amount Lost	Number	1416	0	0.00	9,837.60	45.52	239.71	3792
27	age	Number	83	1672	17.00	98.00	38.47	14.74	4
28	Residential Status	String	5	31618	Live with Parents	Rent			0
29	Status	String	3	0	Bad Debt	OK			0

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Correlations

Distributions in all charts (Dataset Chart tab, Segment Viewer tab, Cross Tabs, Correlations) and tables are automatically recalculated based on the defined weight.

The weighted distributions for variables other than the weight are as follows:

If W is the weight variable, F_i is the i th value of a variable F, and SUM(W_i) is the sum of the weight field over all records where $F=F_i$, then

- The weighted total number of records = SUM(W)
- The weighted frequency for the value F_i = SUM(W_i)
- The weighted percentage of the value F_i = 100 * SUM(W_i) / SUM(W)

Removing Weight. Switching between Weighted and Unweighted Representations

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Once you set the weight, you can easily toggle back and forth between weighted and unweighted representations in all dataset views using the menu command **Edit | Toggle Weight On/Off** or the toolbar button  .

If a weight is set, the weight toggle button becomes highlighted: 

There are three ways to turn the weighting off:

- select “[No Weight]” in the Weight drop-down list
- select the menu command **Edit | Toggle Weight On/Off**
- click the weight toggle button  in the toolbar

The statistics and charts in all dataset views are automatically recalculated and updated whenever you toggle the weight off or on. This makes it very easy to compare various statistics in weighed and unweighted samples.

If you remove the weight, the univariate statistics in the Overview Report are automatically reset to uncalculated. Click **Calculate All** if necessary to recalculate.

NOTE: The univariate statistics shown for the weight field in Overview Report are the same as they were before setting this field as weight (i.e., the weight field is not weighted by itself). The same is true for all other dataset views: they show the *original* distribution of the weight variable (not weighted by itself).

Relationship between Weights in Datasets and Dependent Models

The weights in a dataset and its dependent models can be changed or reset independently. When you toggle weight on or off in the dataset Overview Report, this does not affect existing trees and other models depending on the dataset.

By default, any new model created on the basis of a weighted dataset inherits the weight from it. More precisely, when inserting a model for a weighted dataset, the weight field defined in the dataset is automatically listed in the model's Attribute Editor as "weight" (in the Usage column). You can override this in the Insert Tree (or Predictive / Cluster Model) Wizard and redefine the field as an analysis variable by changing the usage from "weight" to "independent" if desired. Once the model is inserted, it is independent of the parent dataset in the sense that any changes of weights in the model will not affect the parent dataset and vice versa.

Overview Report

The **Overview Report** is a dataset view that displays the dataset structure and univariate statistics such as the mean, maximum, and minimum values, the number of missing values, etc.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

You can customize the Overview Report by choosing which statistics to display or hide using the command **Tools | Options** from the main menu. To set global preferences for new datasets, use **Tools | Preferences.** See the topic Customizing the Overview Report for details. Note that Preferences are not applied to objects already created.

The statistics available in the Overview Report depend on the dataset type. For native Altair KDD datasets, there are 8 groups of statistics with the total of 40 different statistics available. For Spark dataframes in Knowledge Studio for Apache Spark, there are 9 univariate statistics available in Overview Report. See also Options and Preferences: Overview Report.

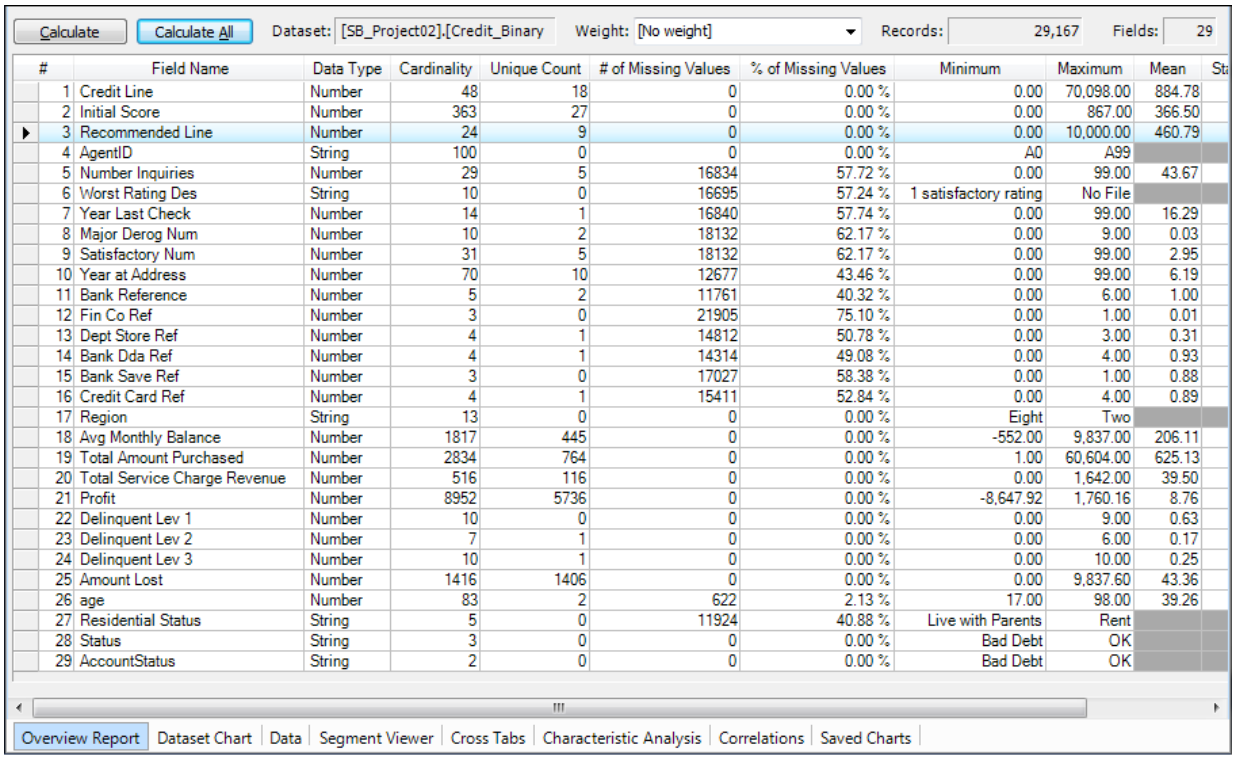
Calculate and **Calculate All** commands

Upon importing a new dataset, only the columns selected in Tools | Preferences -> Overview Report are displayed. You can add more statistics for the current dataset using the **Tools | Options** command.

Click the **Calculate All** button to calculate the statistics for all fields. If you want to see the statistics for one field only, select the desired field and click the **Calculate** button.

A **weight** can be set in Overview Report by selecting the desired field from the *Weight* drop-down list. Note that the list contains only numeric fields with nonnegative values. For details about weights, see the Weight topic.

All columns in the Overview Report are sortable.



#	Field Name	Data Type	Cardinality	Unique Count	# of Missing Values	% of Missing Values	Minimum	Maximum	Mean
1	Credit Line	Number	48	18	0	0.00 %	0.00	70,098.00	884.78
2	Initial Score	Number	363	27	0	0.00 %	0.00	867.00	366.50
3	Recommended Line	Number	24	9	0	0.00 %	0.00	10,000.00	460.79
4	AgentID	String	100	0	0	0.00 %	A0	A99	
5	Number Inquiries	Number	29	5	16834	57.72 %	0.00	99.00	43.67
6	Worst Rating Des	String	10	0	16695	57.24 %	1 satisfactory rating	No File	
7	Year Last Check	Number	14	1	16840	57.74 %	0.00	99.00	16.29
8	Major Derog Num	Number	10	2	18132	62.17 %	0.00	9.00	0.03
9	Satisfactory Num	Number	31	5	18132	62.17 %	0.00	99.00	2.95
10	Year at Address	Number	70	10	12677	43.46 %	0.00	99.00	6.19
11	Bank Reference	Number	5	2	11761	40.32 %	0.00	6.00	1.00
12	Fin Co Ref	Number	3	0	21905	75.10 %	0.00	1.00	0.01
13	Dept Store Ref	Number	4	1	14812	50.78 %	0.00	3.00	0.31
14	Bank Dda Ref	Number	4	1	14314	49.08 %	0.00	4.00	0.93
15	Bank Save Ref	Number	3	0	17027	58.38 %	0.00	1.00	0.88
16	Credit Card Ref	Number	4	1	15411	52.84 %	0.00	4.00	0.89
17	Region	String	13	0	0	0.00 %	Eight	Two	
18	Avg Monthly Balance	Number	1817	445	0	0.00 %	-552.00	9,837.00	206.11
19	Total Amount Purchased	Number	2834	764	0	0.00 %	1.00	60,604.00	625.13
20	Total Service Charge Revenue	Number	516	116	0	0.00 %	0.00	1,642.00	39.50
21	Profit	Number	8952	5736	0	0.00 %	-8,647.92	1,760.16	8.76
22	Delinquent Lev 1	Number	10	0	0	0.00 %	0.00	9.00	0.63
23	Delinquent Lev 2	Number	7	1	0	0.00 %	0.00	6.00	0.17
24	Delinquent Lev 3	Number	10	1	0	0.00 %	0.00	10.00	0.25
25	Amount Lost	Number	1416	1406	0	0.00 %	0.00	9,837.60	43.36
26	age	Number	83	2	622	2.13 %	17.00	98.00	39.26
27	Residential Status	String	5	0	11924	40.88 %	Live with Parents	Rent	
28	Status	String	3	0	0	0.00 %	Bad Debt	OK	
29	AccountStatus	String	2	0	0	0.00 %	Bad Debt	OK	

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The contents of the Overview Report can be copied and pasted into a Microsoft Office application such as Word or Excel using the menu commands **Edit | Select All** and **Edit | Copy** or the keyboard shortcuts CTRL A and CTRL C.

General attributes of the dataset are displayed at the top of the Overview Report:

Dataset	Displays the dataset name with the parent project name in the format **[project name].[dataset name]**
Weight	The weight field (if specified)
Records	The total number of records in the dataset
Fields	The total number of fields in the dataset

Attributes and univariate statistics displayed in Overview Report:

The following attributes and statistics are available in the Overview Report. Not all of them are displayed by default. See the topic Customizing the Overview Report for details.

Name and Type	
#	Field number. Fields are numbered in the same order as in the original data source.
Field Name	Field name. Fields cannot be renamed unless they are calculated fields added with the Data Transformation operation. If you need to rename a raw field, re-import the dataset and rename the field in the last page of the Insert Dataset Wizard (the Field Transformer page).
Field Label	The label of the field. This is an editable field for specifying an optional field label. In some application views you can choose whether to display the original field names or labels.
Data Type	The data type of the field. Data types include Text, Integer, Number, Boolean, Date, Time, and Timestamp. Data types cannot be changed once the dataset has been imported. If you need to change the type of a field, either re-import your dataset and change the types in the last page of the import wizard (the Field Transformer page), or create a copy of the field converting it to the desired type using Data Transformations. Example type conversion command: CAST ([field_name] AS Integer)
Basic Measures	
Cardinality	The number of distinct values in the field.
Unique Count	The number of values that occur exactly once.
# of Missing Values	The number of missing values in the field.
% of Missing Values	The percentage of missing values in the field.
# of Non-Missing Values	The number of non-missing values in the field.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

% of Non-Missing Values	The percent of non-missing values in the field.
Minimum	The minimum value in the field. Also applicable to String fields (with respect to the lexicographic order).
Maximum	The maximum value in the field. Also applicable to String fields (with respect to the lexicographic order).
Sum	The sum of all values in the field.
Measures of Central Tendency	
Mean	The mean value of the field.
Median	The middle value of the ordered set of values in the field across all records, i.e., the value such that an equal number of values are less than and greater than this value (if the cardinality is odd), or the lesser of the two central values (if the cardinality is even). Also, see Quantiles below.
Mode 1	The most common value in the field.
Mode 2	The second most common value in the field.
Measures of Dispersion	
Standard Deviation	The standard deviation of the field values (the square root of variance).
Range	The difference between the maximum and the minimum values in the field.
Interquartile Range	The difference between the upper and lower quartiles (the 75th and 25th percentiles). See the Quantiles section below.
Coefficient of Variation	The ratio of the standard deviation to the mean value of the field.
Std Error Mean	The standard error of the mean: The ratio of the standard deviation to the square root of the total number of records.
Variance	The variance of the field values (the second central moment). It is equal to the mean of the squared values minus the square of the mean value of the field.
Skewness	The skewness of the value distribution in the field. Characterizes the asymmetry of the distribution of numeric values in the field. A zero value indicates relatively even distribution of values on both sides of the mean.
Kurtosis	The kurtosis of the value distribution in the field. Characterizes the shape of the distribution of numeric values in the field in terms of the width of peak, tail weight, and lack of shoulders.
Median Absolute Deviation	The median absolute deviation (MAD) of the values in the field. It is equal to the median of the absolute deviations from the *median* of the field (see above).
Quantiles	

Quantiles are data points separating equal-sized (or approximately equal-sized) subsets of the ordered set of all values in a numeric field. The method used here to compute quantiles is as follows. Let x[1], ... , x[n] be the ordered set of all distinct non-missing values

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

in the field x in the ascending order. x[i] is called the *ith order statistic.* Let 0 < p < 1 and let j be the integer part of np - 1/2. Then:

Q(p) = x[j] if j = np - 1/2 and j is even

Q(p) = x[j+1] otherwise

In the list below, p is expressed as percentage (1%, 5%, etc.).

Customize	Custom quantile. If the desired quantile is not among the ones listed below, specify the desired percentage here.
1%	The lower percentile (1st percentile).
5%	The 5th percentile.
10%	The lower decile (10th percentile).
25%	The lower quartile (the 25th percentile).
75%	The upper quartile (the 75th percentile).
90%	The upper decile (90th percentile).
95%	The 95th percentile.
99%	The upper percentile (99th percentile).
Extreme Observations	
Lowest Observations	Show N lowest values, where N is specified in the Parameters column. The default is 5 values.
Highest Observations	Show N highest values, where N is specified in the Parameters column The default is 5 values.
Tests for Normality	
Shapiro-Wilk Test	The Shapiro-Wilk test for the null hypothesis that the data in the field is normally distributed. It is the ratio of the square of a linear combination of the order statistics x[1], ... , x[n] and a sum-of-squares estimator of the variance. The range of values for the statistic is (0, 1]. The null hypothesis is rejected if the value is too small. See Shapiro and Wilk (1965) in References.
Cramér-Von Mises Criterion	A goodness-of-fit test based on the empirical distribution function (EDF). When used as a test for normality, the Cramér-Von Mises Criterion evaluates the squared difference between the EDF of the data in the field and the standard normal distribution function F. Let x[1], ... , x[n] be the ordered set of all distinct non-missing values in the field x in the ascending order. Then the statistic is computed as $\frac{1}{12n} + \sum_{i=1}^{n} \left(F(x_i) - \frac{2i-1}{2n} \right)^2$ The null hypothesis that the data has standard normal distribution is rejected If the value is larger than the chosen threshold.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Kolmogorov -Smirnov Test

A goodness-of-fit test based on the empirical distribution function (EDF). When used as a test for normality, the Kolmogorov-Smirnov statistic measures the largest vertical difference between the EDF of the data in the field and the standard normal distribution function F with the same mean and variance. Let x[1], ... , x[n] be the ordered set of all distinct non-missing values in the field x in the ascending order. Then the statistic is the larger of the two values

$$\max_i \left(\frac{i}{n} - F(x_i)\right) \text{ and } \max_i \left(F(x_i) - \frac{i-1}{n}\right)$$

Anderson-Darling Test

A goodness-of-fit test based on the empirical distribution function (EDF). When used as a test for normality, the Anderson-Darling Test evaluates the weighted squared difference between the EDF of the data in the field and the standard normal distribution function F. Let x[1], ... , x[n] be the ordered set of all distinct non-missing values in the field x in the ascending order, and y[i] be their standardized values. Then the statistic is computed as

$$-n - \frac{1}{n}\sum_{i=1}^{n}((2i-1)\ln F(x_i) + (2(n-i)+1)\ln(1-F(x_i)))$$

The null hypothesis that the data has standard normal distribution F is rejected If the value is larger than the chosen threshold.

Tests for Location

Student's t-test

A test statistic for the null hypothesis that the mean value of the field is equal to the specified value of μ_0. It is set to zero by default. To change the value of μ_0, click the **Set μ_0 values** button. The statistic is calculated as

$$\frac{\mu - \mu_0}{\sigma/\sqrt{n}}$$

where μ is the mean value of the field, and σ is the standard deviation, and n is the number of all distinct nonmissing values in the field.

Sign Test

A test statistic for the null hypothesis that the median value of the field is equal to the specified value of μ_0. It is set to zero by default. To change the value of μ_0, click the **Set μ_0 values** button. The statistic is calculated as (n1 - n2)/2, where n1 is the number of values greater than μ_0, and n2 is the number of values less than μ_0.

Wilcoxon signed-rank test

A test statistic for the null hypothesis that the median value of the field is equal to the specified value of μ_0 under the assumption of a symmetric distribution. It is set to zero by default. Let

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

	$R(\lvert x_i - \mu_0 \rvert)$ be the rank of nonzero $\lvert x_i - \mu_0 \rvert$ and n_t be the number of values x[i] not equal to μ_0. The test statistic is $\sum_{x_i \neq \mu_0} R(\lvert x_i - \mu_0 \rvert) - \frac{n_t(n_t+1)}{4}$
Confidence Interval for Population Mean	A confidence interval for the mean is an estimated range of values which includes the mean of the variable at the probability level specified in the Parameters column. The default probability level is 95%. If this statistic is selected, the lower and upper bounds for the interval are displayed.

See also:

Customizing the Overview Report

Options and Preferences

Customizing the Overview Report

You can customize the Overview Report by selecting the attributes and statistics to be displayed and changing the font.

To customize the report for the current dataset, select the **Tools | Options...** command from the menu or click the **Options** button in the toolbar.

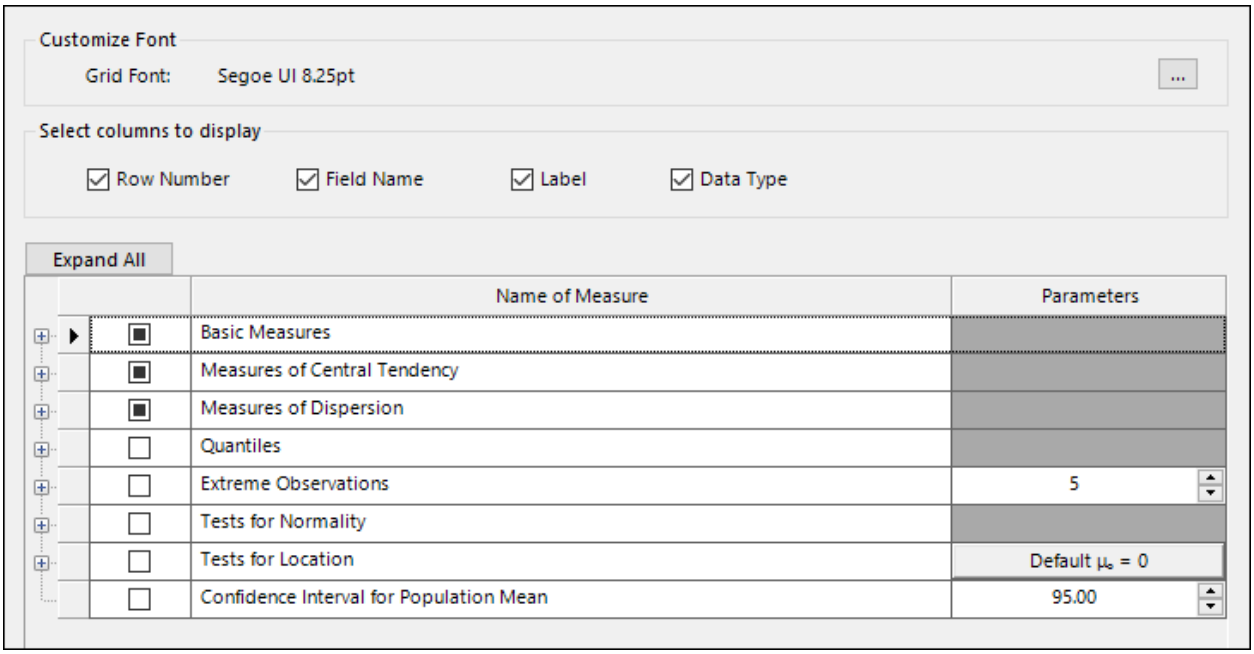
To set the columns to be displayed by default in the Overview Report for all *new* datasets, select **Tools | Preferences** instead of **Options** and choose the Overview Report tab. Changes in Preferences do not affect the datasets already created.

The statistics available in the Overview Report depend on the dataset type. For native Altair KDD datasets, there are 8 groups of statistics with the total of 40 different statistics available. For Spark dataframes in Knowledge Studio for Apache Spark, there are 9 univariate statistics available in Overview Report. See also Options and Preferences: Overview Report.

You can expand any group by clicking on the plus sign on the left, or expand all groups by clicking Expand All. The application default includes cardinality, unique count, the number and percentage of missing values, minimum, maximum, mean, and standard deviation. For the description of all available statistics, see the Overview Report topic.

Select the desired attributes and statistics to be displayed and change the corresponding parameters if necessary.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click OK. Then click **Calculate** or **Calculate All** in the Overview Report to calculate the newly added statistics.

Clicking **Reset** in the Options dialog will reset everything to the current Preferences. Clicking **Reset** in the Preferences dialog will reset everything to the application defaults described above.

Note: If you include all available statistics, they may take a while to calculate for very large datasets.

Note that the following statistics are not available in the In-Database Analytics mode, since there are no standard SQL operations for them: Unique Count, Variance, DevSqr, Mode (Mode1 and Mode2), Lower Quartile, Upper Quartile, Median, Skewness, and Kurtosis.

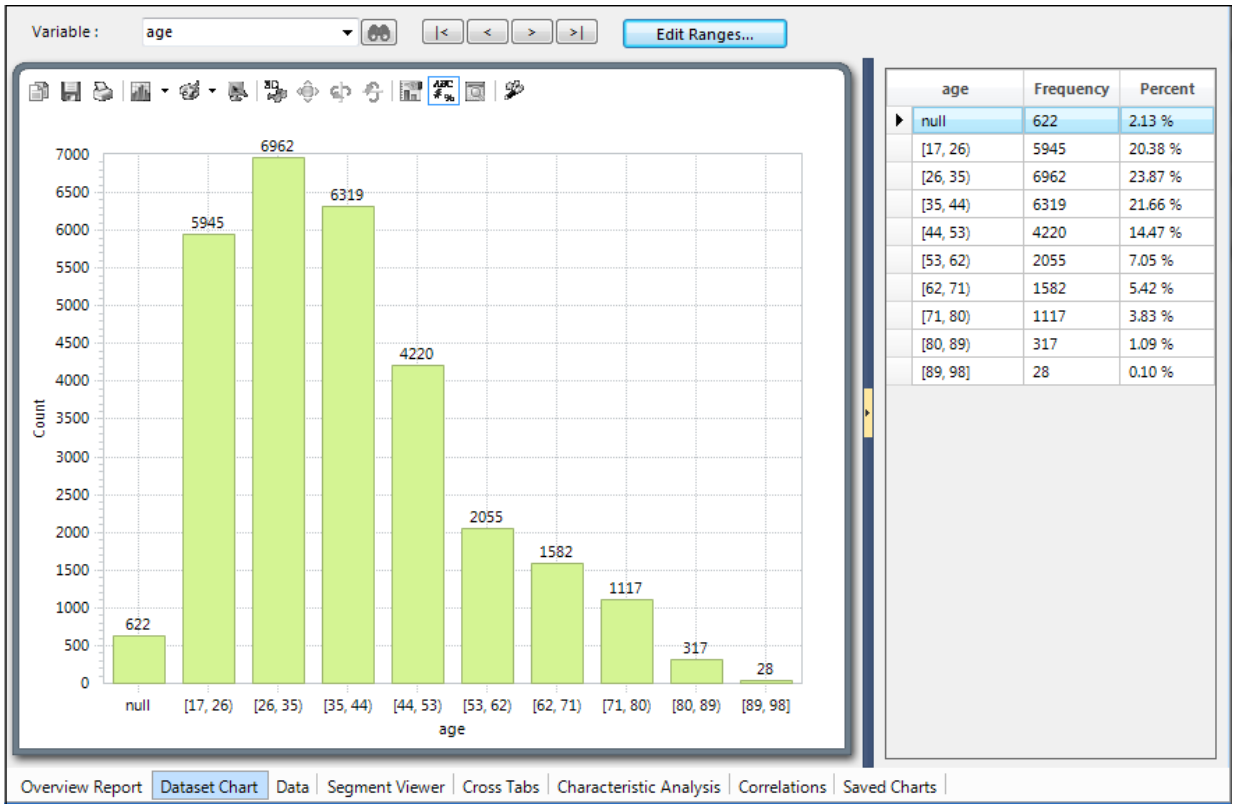
Dataset Chart

The **Dataset Chart** is a view of the dataset that displays the distribution of data values for a given field in a chart.

If the field is continuous, the distribution is displayed as a histogram. If the field is discrete, the distribution is displayed as a pie chart. Various chart display settings including the chart type, labels, colors, and fonts can be changed using the **Chart** menu or the context menu that appears when right-clicking the chart. If you want to see the chart data, select the **Data Editor** item from the context menu.

You can change the field being displayed by choosing from the drop-down list labeled **Field** or using the navigation buttons to the right of the drop-down list.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



By default, the chart for discrete variables shows a maximum of 10 categories. If the cardinality of a discrete variable is larger than 10, the remaining categories are mapped into a category designated by the symbol '###'. If you need to increase the maximum number of categories displayed, click the **Edit Grouping** button to change the threshold value.

Note: In Dataset Chart, a numeric, date, time, or timestamp variable is treated as discrete if its cardinality is less or equal to 10. In this case, its distribution is represented in a Pie Chart by default. Otherwise, it is considered to be continuous and its distribution is represented in a Histogram by default.

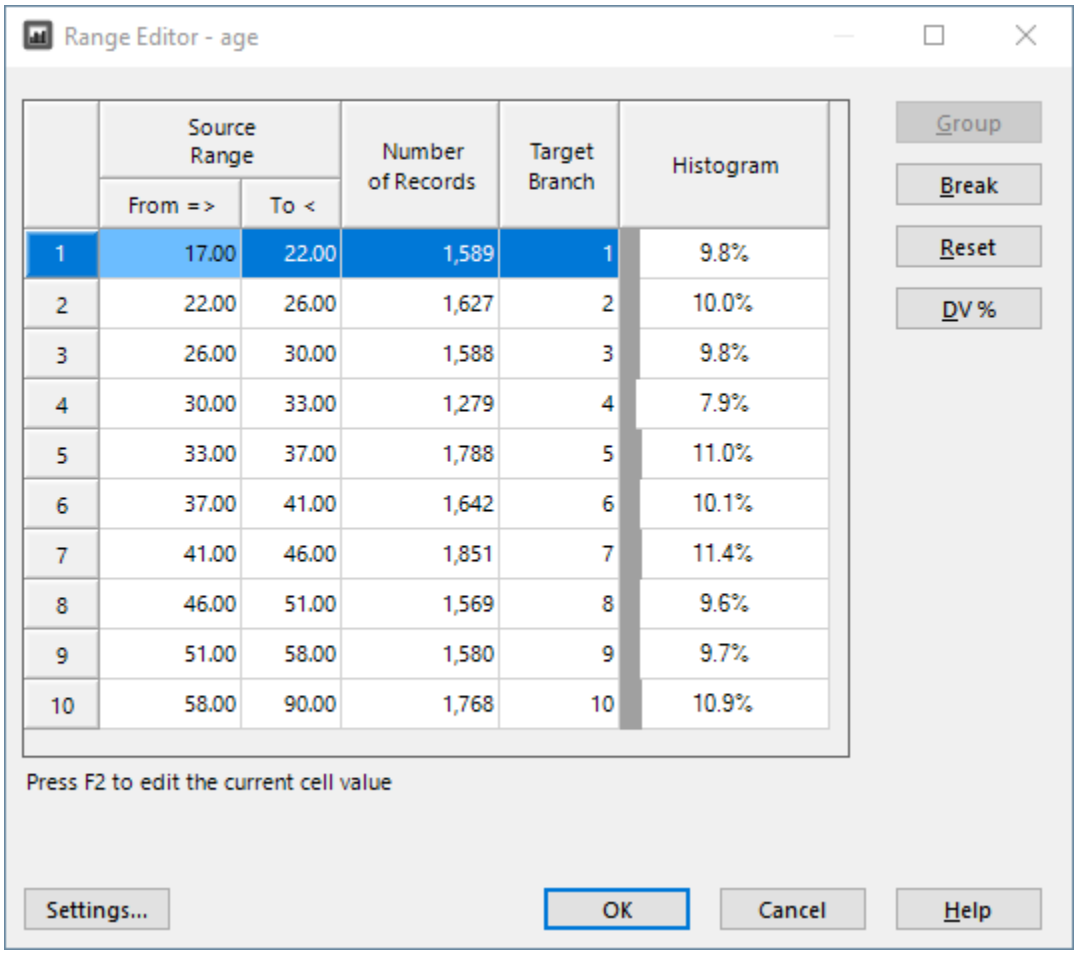
The Edit Ranges button is only available if the field to display is continuous. This feature allows you to you to change the displayed ranges of a continuous variable.

The **dataset chart** and the associated data can be copied and pasted into an Microsoft Office document. This allows the rapid and easy preparation of reports, presentations and summaries.

Edit Ranges

Range Editor allows you to define the boundaries of the bins for a continuous variable and the way its data should be grouped and distributed across the bins.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Range Display

Source Range From=> To<	Indicates the boundaries of each interval (bin) for the selected variable distribution.
Number of Records	Number of records contained in the bin.
Target Branch	Represents the index of the interval.
Histogram	Percentage of the total number of records in the dataset contained in the bin.

Range Editing

Group	Allows you to combine ranges. Only contiguous ranges can be grouped. Example of Group
Break	Allows you to add a new interval to the distribution by specifying break points within an existing interval. Example of Break

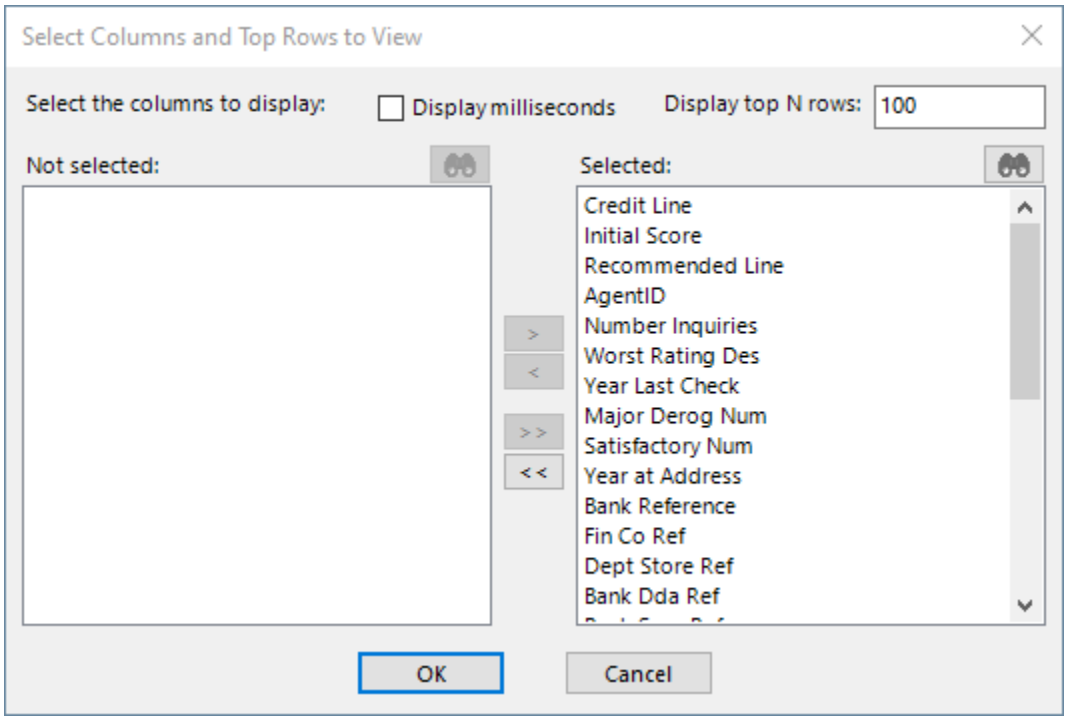
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Reset	The reset button allows you to return to the original ranges displayed when the range editor was first opened.
Settings	The settings button allows you to change the default range settings. See Settings.

The **Data** view is a view of the dataset and displays the raw data in the dataset. Missing values in the dataset are displayed as `null`.

The first time you click on the Data tab, you are prompted to select the fields you would like to be displayed. All fields are selected by default.

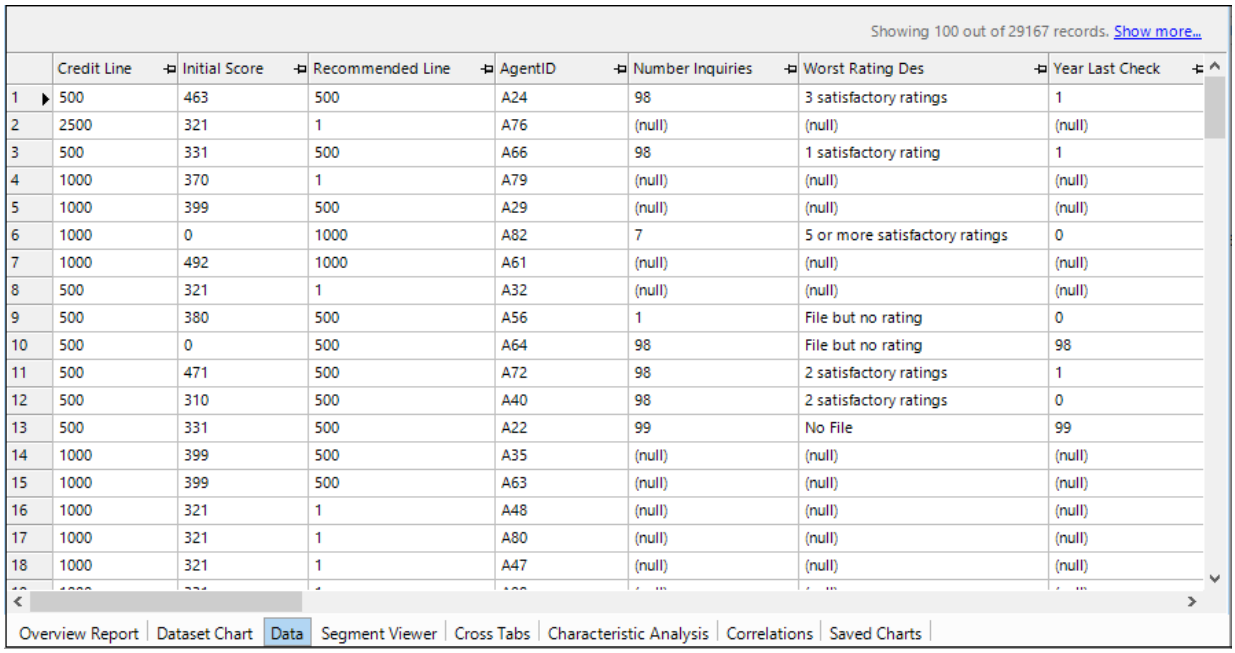
- Select or deselect the fields by clicking on the arrow buttons or double-clicking on the field names.
- Specify the number of top rows in the dataset to display by default. Top 100 rows is the default. You will be able to load more rows if necessary.
- If your source data contains Time or Timestamp values with milliseconds, select the option ***Display milliseconds***.



Note that new calculated columns added in the Dataset Editor are not displayed by default in the Data tab if it had been opened before you added the new columns. Click the **Options** button in the toolbar to include the newly added columns in the Data view.

Click **Show more..** at the top right to load more rows if necessary.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Showing 100 out of 29167 records. Show more...

	Credit Line	Initial Score	Recommended Line	AgentID	Number Inquiries	Worst Rating Des	Year Last Check
1	500	463	500	A24	98	3 satisfactory ratings	1
2	2500	321	1	A76	(null)	(null)	(null)
3	500	331	500	A66	98	1 satisfactory rating	1
4	1000	370	1	A79	(null)	(null)	(null)
5	1000	399	500	A29	(null)	(null)	(null)
6	1000	0	1000	A82	7	5 or more satisfactory ratings	0
7	1000	492	1000	A61	(null)	(null)	(null)
8	500	321	1	A32	(null)	(null)	(null)
9	500	380	500	A56	1	File but no rating	0
10	500	0	500	A64	98	File but no rating	98
11	500	471	500	A72	98	2 satisfactory ratings	1
12	500	310	500	A40	98	2 satisfactory ratings	0
13	500	331	500	A22	99	No File	99
14	1000	399	500	A35	(null)	(null)	(null)
15	1000	399	500	A63	(null)	(null)	(null)
16	1000	321	1	A48	(null)	(null)	(null)
17	1000	321	1	A80	(null)	(null)	(null)
18	1000	321	1	A47	(null)	(null)	(null)

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

The data can be copied and pasted into Microsoft Office applications. Drag the mouse down the grid or use the CTRL + mouseclick to select the desired rows, then use the right-click context menu on the grid to copy the rows. You can also use the **Edit** menu commands **Select All** and **Copy.**

To paste the data into a Microsoft Office application such as Excel, open the application and paste the data from the clipboard.

Segment Viewer

The **segment viewer** helps you to visualize the distributions of the independent variables under each category of the segmentation variable. Specifically, you can determine if the distributions of the independent variables vary across the segmented variable. If you can see this variation, then this may be a variable of interest for you in your analysis.

The first time the **segment viewer** is viewed for a dataset, the Options dialog appears. You can call up Options manually:

From the menu:

Tools | Options...

From the toolbar:

Click on the Options icon

The **segment viewer charts** and the associated data can be copied and pasted into is Microsoft Office applications. This allows the rapid and easy preparation of reports, presentations and summaries.

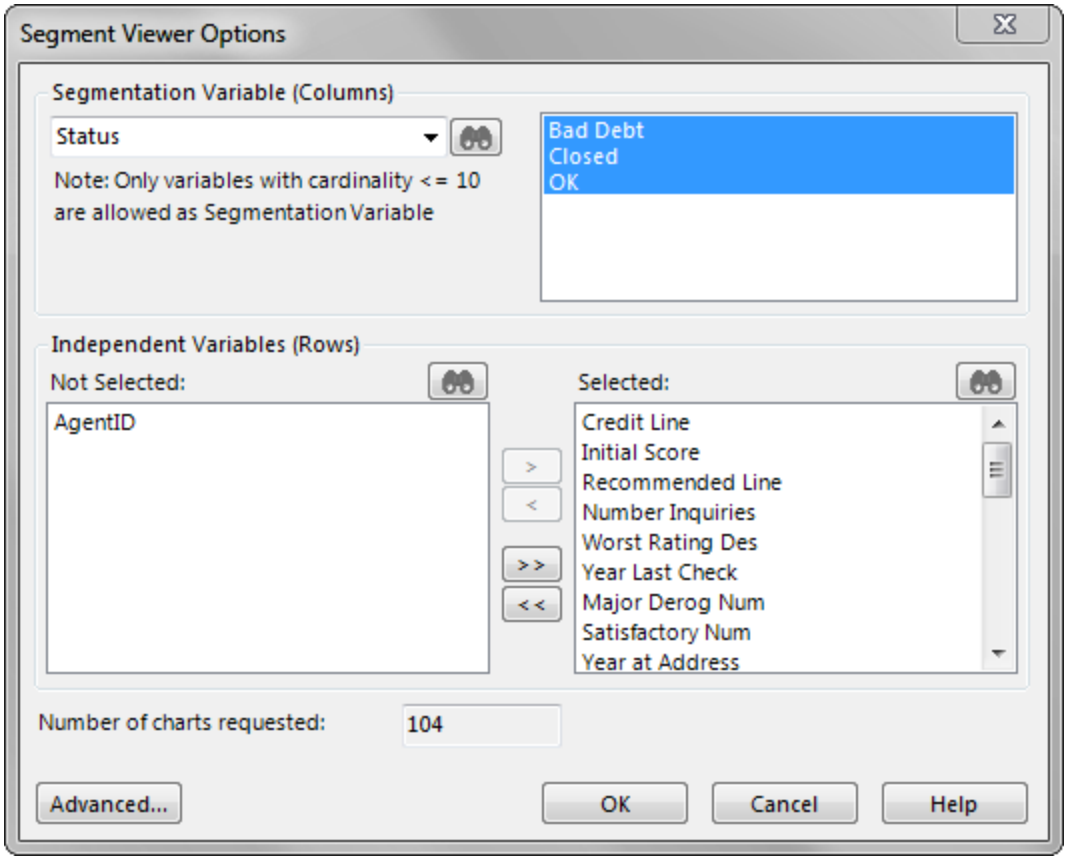
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

See also:

Segment Viewer: Options
Segment Viewer Charts
Segment Viewer: Advanced Options
Copying the Charts

Segment Viewer Options

The segment viewer options allow you to pick a dependent variable to compare to multiple independent variables.



Segmentation Variable	Variable to segment. The categories of the chosen variable are displayed to the right. Charts will be created for the highlighted categories. All the categories are highlighted by default. The segmentation variable should have no more than 10 distinct values.
Independent Variable	Variables to compare to the dependent, or segment, variable. One or more independent variables can be selected.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Number of Charts Requested	Number of charts that will be displayed in the segment viewer based on the number of categories in the dependent variable multiplied by the number of independent variables chosen. If the number of charts is greater than 50, a warning message appears. Reduce the number of categories to display for the segmentation variable, if necessary.

Click the Advanced button if you would like to change the binning for the selected variables. See the topic Segment Viewer: Advanced Options for details.

Segment Viewer Charts

The segment viewer charts are created based on the parameters specified in the Segment Viewer Options.

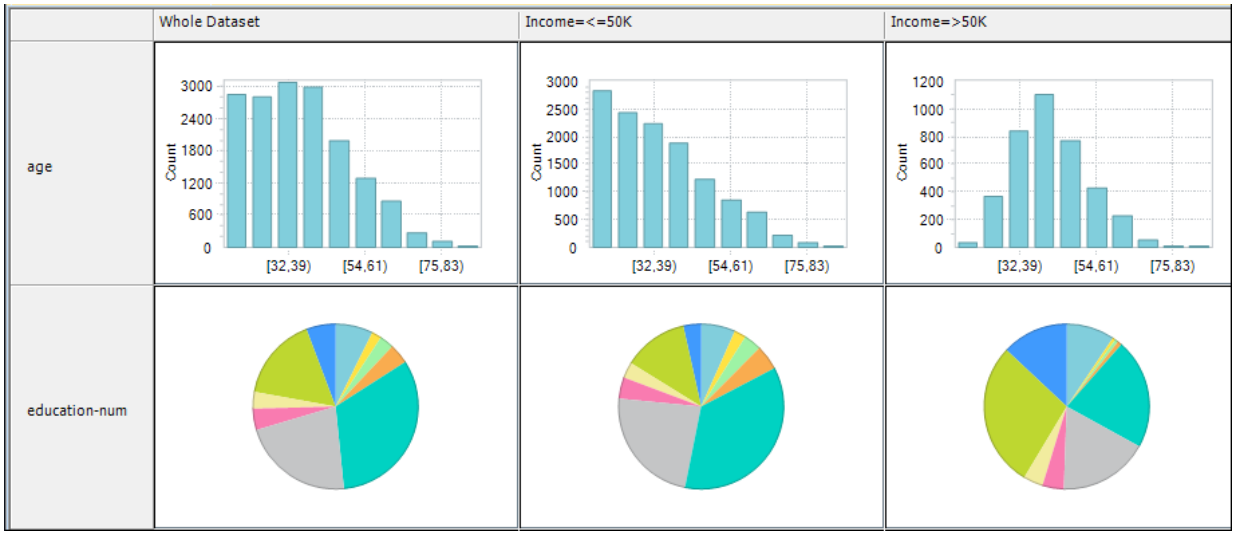
The first column, ***Whole Dataset***, always displays the distribution of the independent variable for all categories of the segmentation variable. The rest of the columns are the distribution of the independent variable across a each category of the segmentation variable.

All non-integer numeric variables regardless of their cardinality are always treated as continuous and displayed in the form of bar charts by default.

Integer and date/time variables are displayed in the form of bar charts if their cardinality is higher than the threshold defined in the Advanced Segment Viewer Options. Otherwise, they are considered discrete and are displayed in the form of pie charts.

String and Boolean variables are always displayed as pie charts by default. To change the pie chart settings such as the maximum number of bins displayed, use the Advanced Segment Viewer Options.

Figure 1



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

In Figure 1, we have the segmentation variable ***Income*** that contains the categories ***<=50K*** and ***>50K*** and the independent variable ***Age***.

The first column, ***Whole Dataset***, displays the distribution of ***Age*** for both the ***<=50K*** and ***>50K*** categories. You can see this same distribution of ***Age*** in the **Dataset Chart** (given the same default settings).

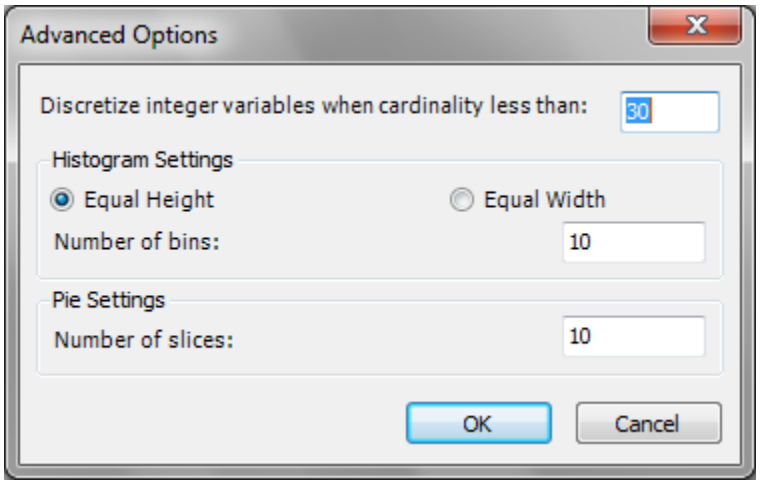
Trends can be observed with these charts. For example:

- Compare the distribution of ***<=50K*** with the whole dataset. Visually the distributions do not look too different.
- Compare the distribution of ***>50K*** with the whole dataset. Visually you can see a distinct variation with the distributions. Specifically, people who make >50K tend to be older. Although this observation may be intuitive, it illustrates how the segment viewer can show you, visually, the differences in the distributions.

There may be more charts displayed then can fit in the view. Use the **Fit to Page** button in the toolbar to fit all charts to the width of the Segment Viewer tab.

Segment Viewer: Advanced Options

The segment viewer settings allow you to change how the charts are displayed.



Discretize integer variables when cardinality is less than:	The distribution of integer variables whose cardinality is less than or equal to this number will be displayed in the form of bar charts (histograms) by default. Otherwise, it will be displayed as pie charts. The default is 30.

Histogram Settings

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Equal Height	Determines how the data is distributed in the chart. When this is checked, the data is distributed so that each bin contains approximately the same number of observations.
Equal Width	Determines how the data is distributed in the chart. When this is checked, the width of the intervals on the x-axis will be the same for all bins.
Number of bins	Number of bins in which to distribute the data. The default is 10.

Pie Settings

Number of slices	Sets the number of categories to display on the pie chart. If the number of categories is greater than this number, then the remaining categories are combined into a category called ###. The default is 10.

Copying the Charts

Copying all Charts

To copy all the charts that are visible:

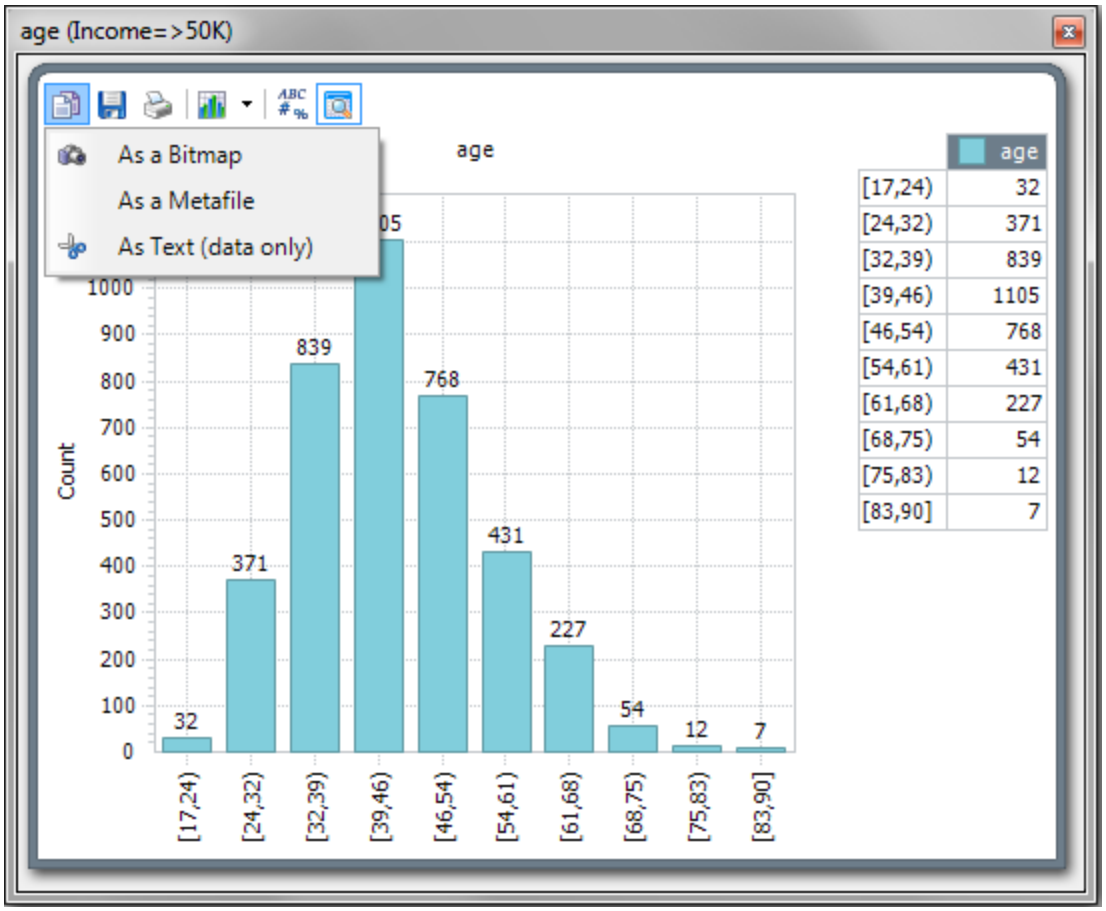
1. Right-click on the segment viewer workspace
2. From the popup, choose **Copy**
3. Paste into any Office application

Alternatively, you can use the **Edit | Copy** menu command.

Copying Individual Charts

1. Double-click on a chart in the segment viewer workspace. This will bring up the chart in a separate window.
2. Click the top left icon ("Copy to Clipboard")

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



1. Choose the type of copy

As a Bitmap	The chart is copied into the clipboard as a bitmap image.
As a Metafile	The chart is copied into the clipboard as a metafile. A metafile is a mechanism for retaining and transporting graphical data.
As Text (data only)	Only the chart data is copied (in the tabular form).

Cross Tabs

The **Cross Tabs** view helps you to visualize your data in cross tables between two or more variables. The cross tables can be viewed in the chart form or a tabular form (report mode).

You can switch between the chart mode and the report mode by clicking the Show As Report icon  in the toolbar or selecting **View | Show As Report** from the menu.

You can choose more than one variable to be shown on each axis of the chart.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The cross tabs chart and the associated data can be copied and pasted into Microsoft Office applications for quick and easy preparation of reports, presentations, and summaries.

The first time you navigate to the **Cross Tabs** view of a dataset, the Cross Tabs Options dialog appears automatically. It guides you through the process of creating a cross table chart and report.

This section deals with specifying the necessary parameters to create the Cross Tabs chart and report. The resulting chart and report are described in the topics Cross Tabs: Chart and Cross Tabs: Report.

If the options have already been defined and the chart is displayed, you can modify the Cross Tabs settings as follows:

From the menu, select

Tools | Options...

From the toolbar

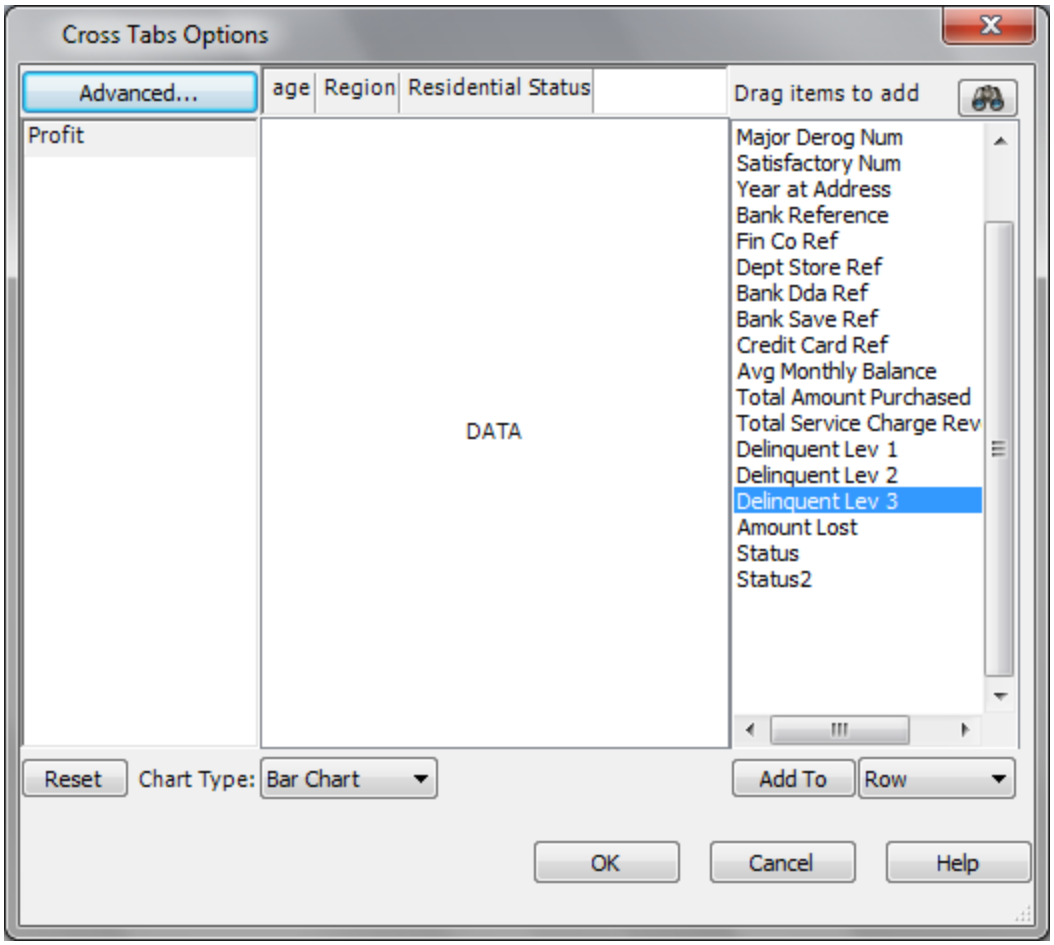
Click on the icon 

The Cross Tabs Options dialog prompts you to select one or more variables for each axis (**ROW** and **COLUMN** sections of the grid represent the axes), specify the measure to be displayed for each Cross Tabs cell (in the **DATA** section), specify the chart type, and optionally specify advanced binning options for each variable.

The right panel of the dialog contains the list of available variables. Advanced variable selection, which is especially useful if you have a large number of variables, can be performed using the Field Finder (launched by clicking the binoculars button ).

The dialog supports a drag and drop interface similar to Microsoft Excel Pivot Table layout dialog.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- To add a variable to any of the dimensions (axes), drag it from the list of variables to the desired dimension (ROW or COLUMN sections). To remove a variable from a section, drag it back to the list of variables or any other section of the grid.
- You can drag any ***numeric*** variable onto the DATA portion of the grid. If no variable is selected for the DATA portion of grid, the assumed measure is ***record count***, which is displayed for each Cross Tabs cell. If you add a variable to the DATA section, the default measure is assumed to be the average of that variable in each Cross Tabs cell. To change the measure by specifying a different aggregate function, see Advanced Options.

- You can add more than one numeric variable into the DATA section, thus specifying several measures. In this case, when viewing the results, you will be able to choose which measure to display (one at a time).
- To add several variables at once (to any section of the grid), first highlight them by pressing and holding the SHIFT or CTRL key and selecting the desired fields with the mouse. From the list box beneath the list of variables, choose where you want to add the variables to (Row, Column, or Data) and click the **Add To** button.
- To remove variables from all sections back to the variable list, click the **Reset** button in the bottom left corner of the dialog.

If you leave the ROW or COLUMN axis blank, a simple one-dimensional result is produced which is the same as charts in the Dataset Chart view.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Double-clicking on a variable added to the ROW or COLUMN section brings up the Advanced Options dialog, which allows you to specify advanced binning options for the variable. Alternatively, you can open the Advanced Options dialog by clicking the **Advanced** button at the top left corner of the dialog. See Advanced Options for details.

Chart Type

Bar Chart is the default chart type. The other options are **Surface Chart** and two-dimensional **Scatter Plot**.

You can specify a measure other than record count (by adding a variable to the DATA section) only if either Bar Chart or Surface Chart is selected.

The Scatter Plot option requires that the following conditions be satisfied:

1. Only one variable per axis
2. Both axis variables must be numeric
3. No Data field specified

See also:

Cross Tabs: Advanced Options

Cross Tabs: Chart

Cross Tabs: Report

Advanced Options

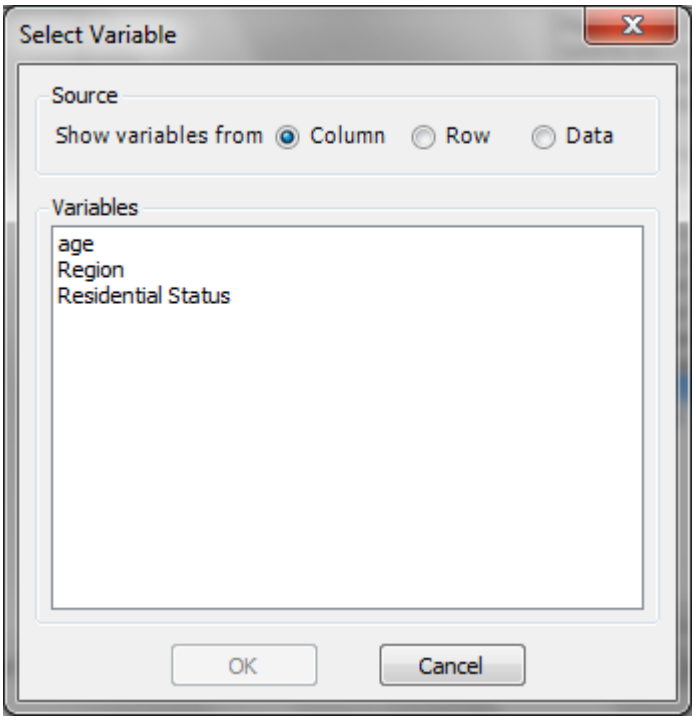
Using the Advanced Options of the Cross Tabs Options dialog, you can

1. specify custom binning for continuous variables
2. specify custom grouping for the categories of discrete variables
3. specify an aggregate function of your choice for the selected measure (determined by the variable(s) in the DATA section, if any)

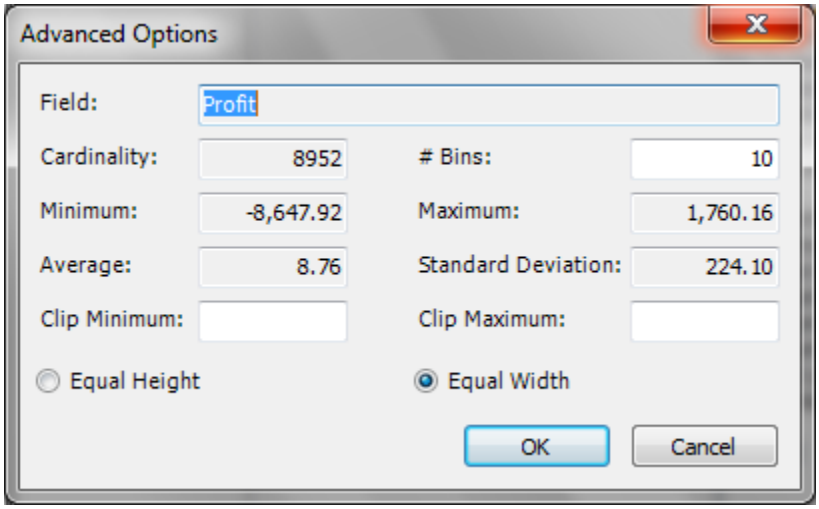
In any of the above cases, double-click the desired variable in the ROW, COLUMN or DATA sections of the grid.

Alternatively, click the **Advanced** button at the top left corner of the dialog, choose a source (Row, Column, or Data) and then select the desired variable from the selected source:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



1. If you are changing the binning for a **continuous variable**, a dialog of the following form appears:

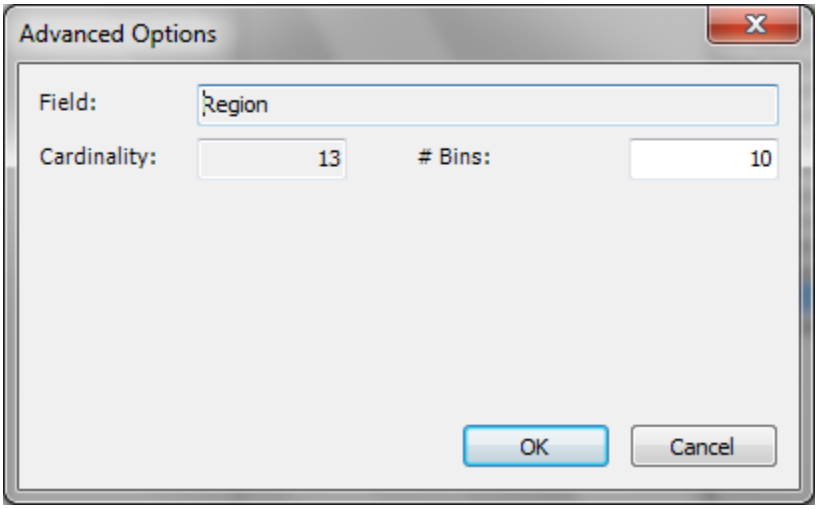


Equal Width	If selected, all intervals (bins) for the selected variable will have the same width.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Equal Height	If selected, the data in the intervals (bins) for the selected variable is distributed so that each bin contains approximately the same number of observations.
Cardinality	The cardinality (the number of unique values) of the selected variable.
# of bins	The number of bins (intervals) in the displayed range of the selected variable.
Minimum	The minimum value of the selected variable in the whole dataset.
Maximum	The maximum value of the selected variable in the whole dataset.
Average	The average of the selected variable in the whole dataset.
Standard Deviation	The standard deviation of the selected variable in the whole dataset.
Clip Minimum	Sets the left boundary for the displayed range of the selected variable. All values smaller than this value (if any) will be ignored.
Clip Maximum	Sets the right boundary for the displayed range of the selected variable. All values larger than this value (if any) will be ignored.

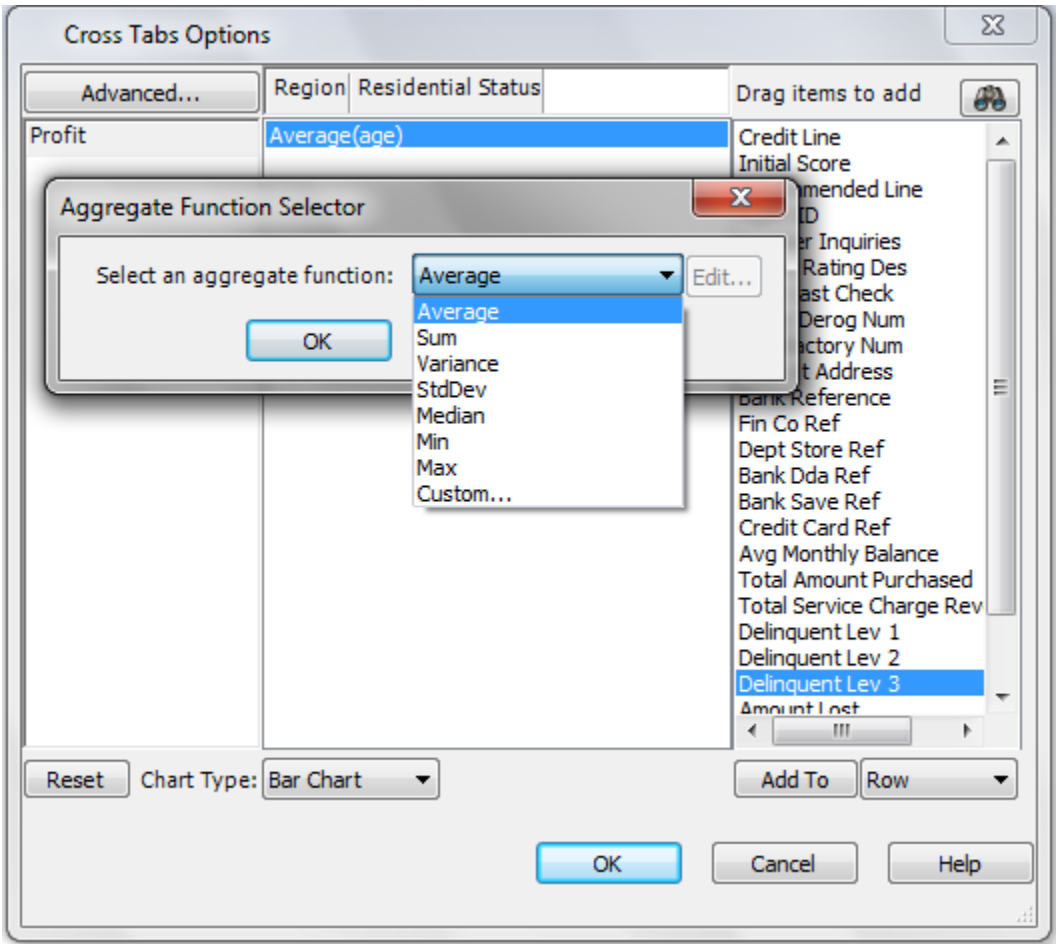
2. If you are changing the grouping for the categories of a **discrete variable**, a dialog of the following form appears:



You can reduce the number of bins if necessary. Categories with the least frequencies will be grouped first.

3. If you are **changing the measure** by changing the aggregate function for the variable in the DATA section, the following dialog appears:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Here you can choose between Average, Sum, Variance, Standard Deviation, Median, Minimum, Maximum, or Custom aggregate functions.

If Custom is selected, you can enter a custom aggregate expression of any complexity using the Expression Editor, which opens automatically. After you finish specifying your custom aggregate expression and close Expression Editor, you can return to the expression by clicking the Edit button.

See also:

Cross Tabs Overview and Options

Cross Tabs: Chart

Cross Tabs: Report

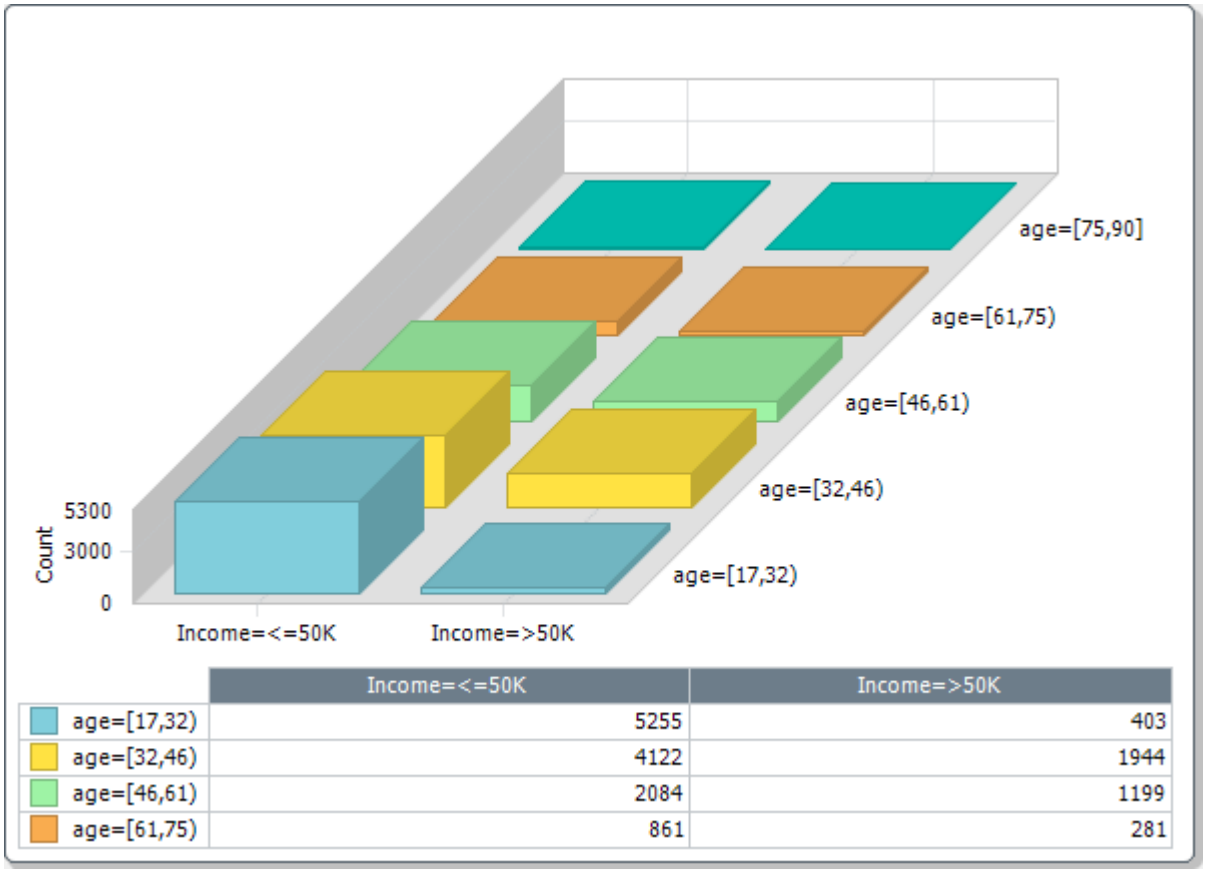
Cross Tabs: Chart

The Chart mode of the **Cross Tabs** view displays the cross tables between two or more variables in the form of a Bar Chart, Surface Chart, or Scatter Plot based on the parameters specified in the Cross Tabs Options dialog.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Note: You can switch between the chart mode and the report mode by clicking the Show As Report icon  in the toolbar or selecting **View | Show As Report** from the menu.

In this example, the chart displays the number of people in each ***Income*** category for each ***Age*** interval. From this chart we can see that the majority of people whose income is above $50000 are in the age group between approximately 26 and 54.

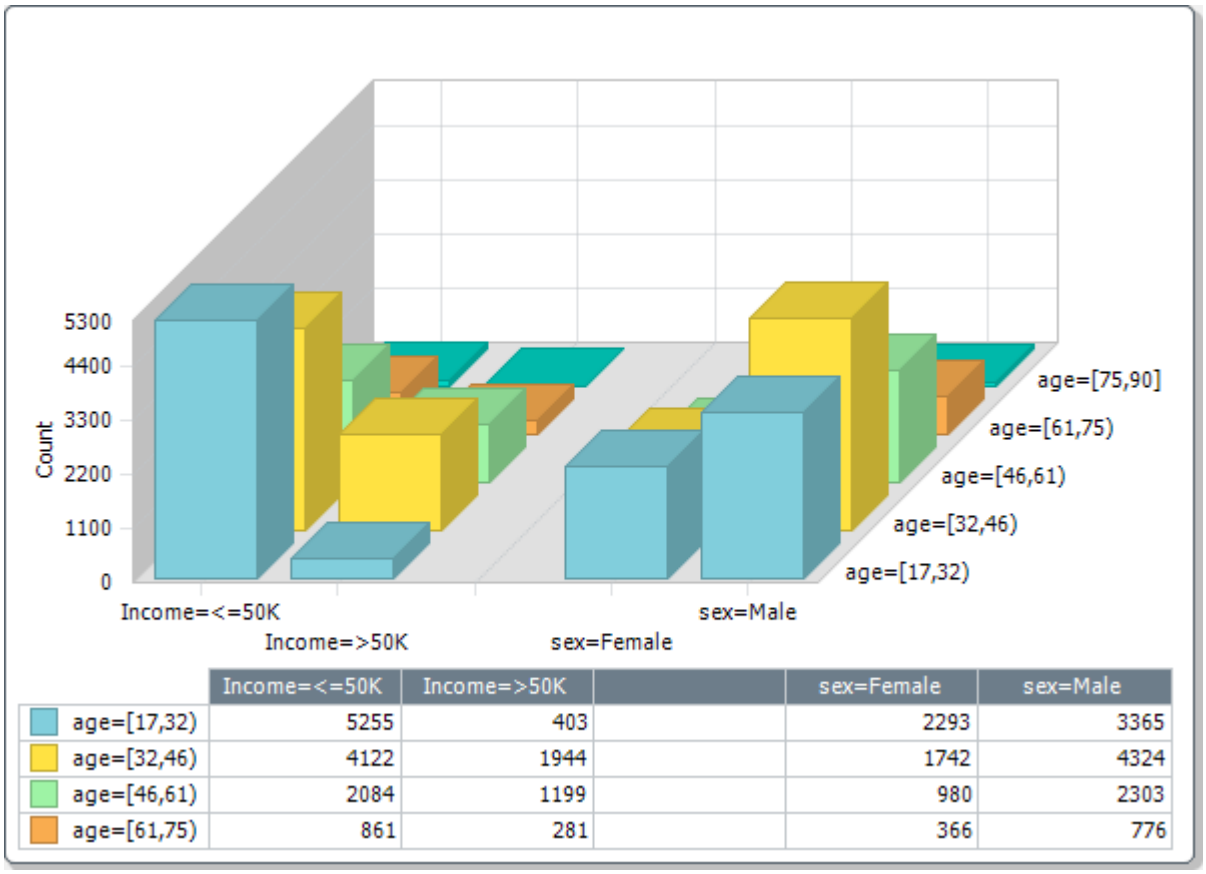


	Income=<=50K	Income=>50K
age=[17,32)	5255	403
age=[32,46)	4122	1944
age=[46,61)	2084	1199
age=[61,75)	861	281

If you choose more than one variable for one of the dimensions (axes), they will be separated by an empty row in the chart for clarity, as shown in the following example.

Here ***Income*** and ***sex*** variables were added to the ROW section in the Cross Tabs Options dialog:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



	Income=<=50K	Income=>50K		sex=Female	sex=Male
age=[17,32)	5255	403		2293	3365
age=[32,46)	4122	1944		1742	4324
age=[46,61)	2084	1199		980	2303
age=[61,75)	861	281		366	776

If you would like to change the binning for the continuous variables, group some categories of the discrete variables, or specify a different measure, open the Cross Tabs Options and double-click on the desired variable. See Advanced Options for details.

See also:

Cross Tabs Overview and Options

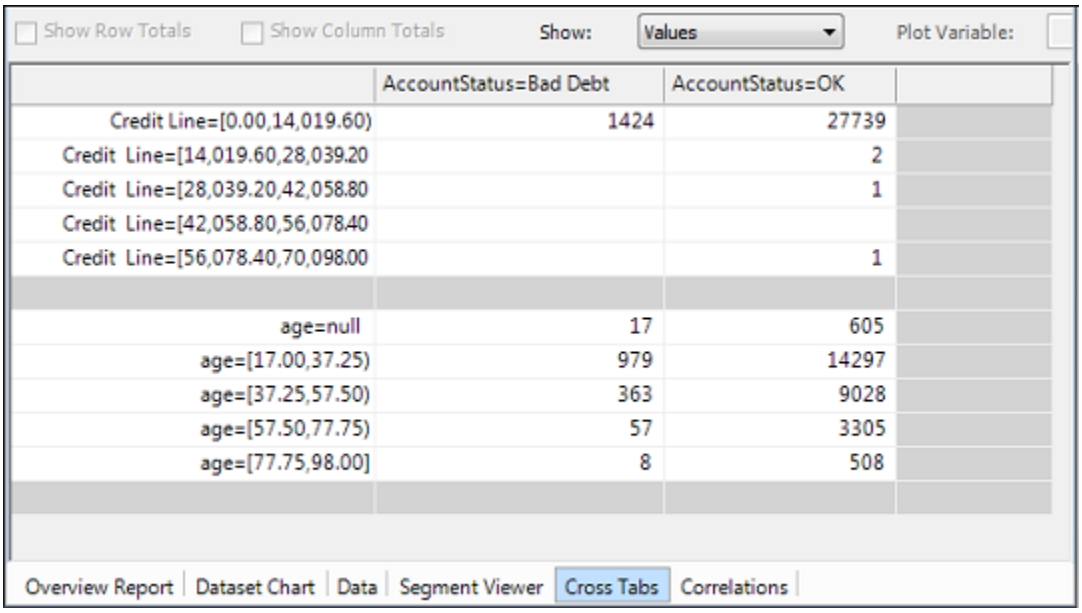
Cross Tabs: Report

Cross Tabs: Report

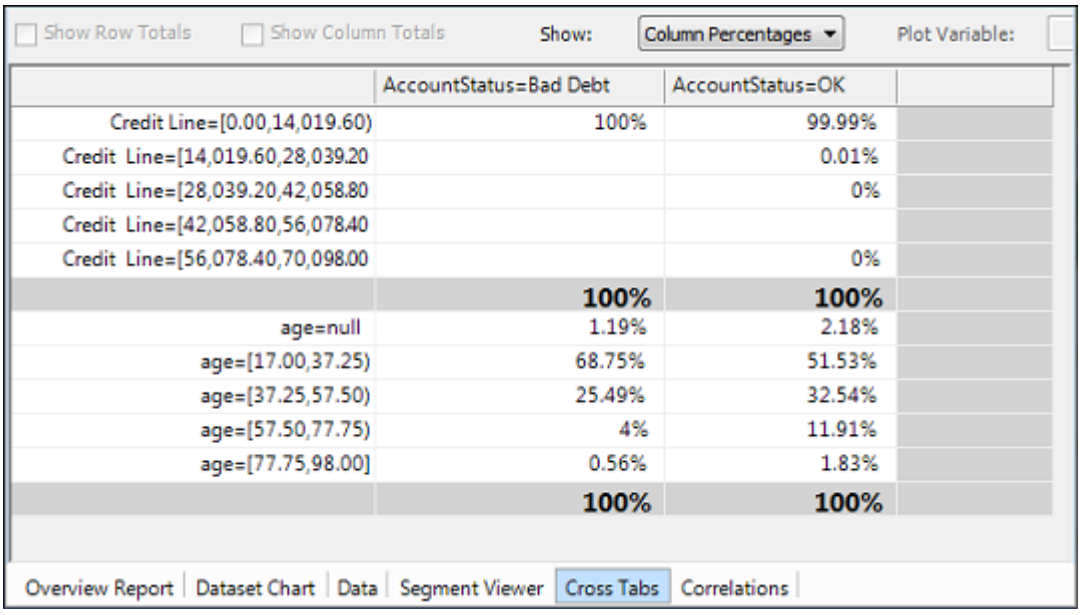
The Report mode of the **Cross Tabs** view displays the cross tables between two or more variables in the form of a tabular report based on the parameters specified in the Cross Tabs Options dialog.

> **Note:** You can switch between the report mode and the chart mode by clicking the Show As Report icon in the toolbar or selecting **View | Show As Report** from the menu.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



You can choose to show values, column percentages, or row percentages.



Characteristic Analysis

The Characteristic Analysis view shows the frequencies of each category of a dependent variable across various ranges (bins) of independent variables. This view helps to reveal relative trends in the distribution of the DV categories by plotting them side by side, making it easy to compare the trend of one category against the others.

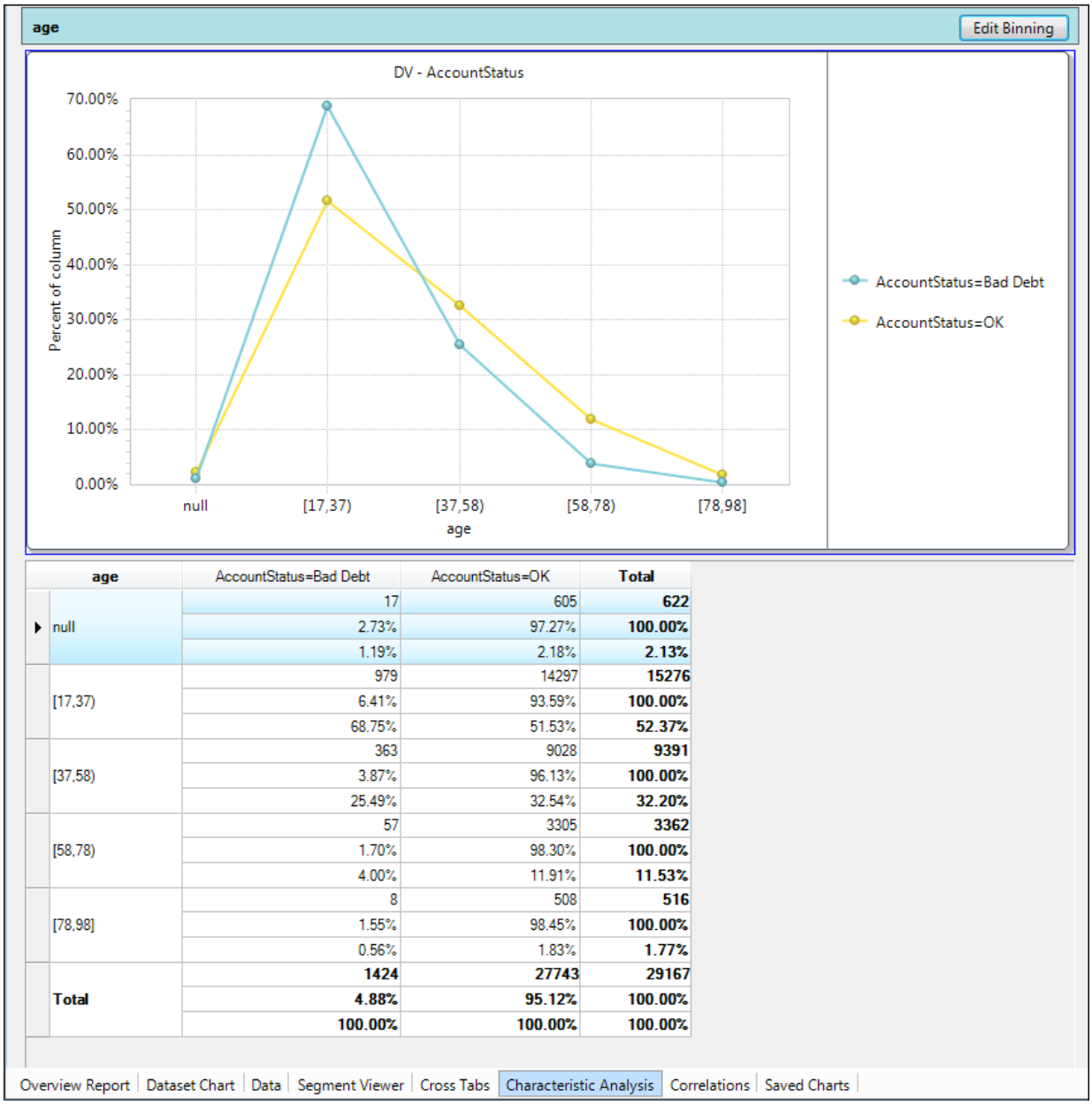
The options dialog provides a choice of displaying charts and report as percent of column or percent of rows. The default is percent of column.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The initial chart is displayed based on the dependent variable selected in Segment Viewer. You can change the dependent variable in the Options dialog.

All graphs are line charts by default.

You can change the grouping of the independent variable by clicking the Edit Binning button at the top right corner. This brings up the familiar Range Editor dialog.



age		AccountStatus=Bad Debt	AccountStatus=OK	Total
null		17	605	**622**
		2.73%	97.27%	**100.00%**
		1.19%	2.18%	**2.13%**
[17,37)		979	14297	**15276**
		6.41%	93.59%	**100.00%**
		68.75%	51.53%	**52.37%**
[37,58)		363	9028	**9391**
		3.87%	96.13%	**100.00%**
		25.49%	32.54%	**32.20%**
[58,78)		57	3305	**3362**
		1.70%	98.30%	**100.00%**
		4.00%	11.91%	**11.53%**
[78,98]		8	508	**516**
		1.55%	98.45%	**100.00%**
		0.56%	1.83%	**1.77%**
Total		**1424**	**27743**	**29167**
		4.88%	**95.12%**	**100.00%**
		100.00%	**100.00%**	**100.00%**

In the example above, we see consistently similar trends of both categories of the dependent variable (Account Status) across the selected age ranges.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

In the example below, we see that the High and Low income trends go against each other in the first two age groups [17,35) and [35,54), whereas they start to go together after age 54.

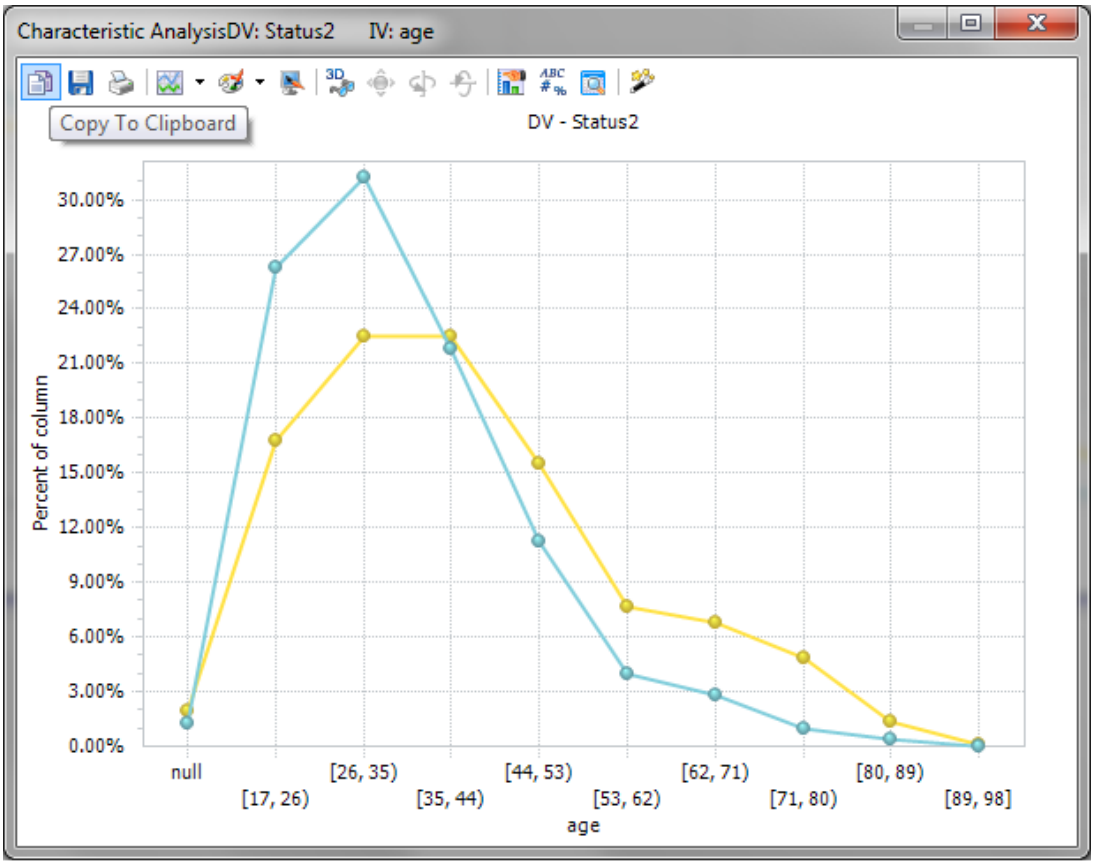


Thus, by looking closer into the behavior of the DV categories within various segments of a variable, Characteristic Analysis may help detect interesting relationships between this variable and the DV even when other methods do not rank this variable as highly significant.

All charts and tables in this view can be easily copied and pasted into any Microsoft Office application or saved as PDF.

- To copy a chart: double-click the desired chart to open it in a separate window and click the Copy icon in the toolbar at the top of the chart.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- To save the whole Characteristic Analysis view as PDF with all of its charts and tables, select the **Copy to PDF** command from the **File** menu.
- To copy a table, right-click on any cell and choose **Select All** from the context menu. Then right-click and select **Copy**.

age	Status2=Bad Debt	Status2=OK	Total
null	5682	195955	**201637**
	2.82%	97.18%	**100.00%**
	1.23%	1.92%	**1.89%**
[17, 44)	366277	6314085	**6680362**
	5.48%	94.52%	**100.00%**
	79.37%	61.73%	**62.49%**
[44, 71)	83219	3077004	**3160223**
	2.63%	97.37%	**100.00%**
	18.03%	30.08%	**29.56%**
[71, 98]	6276	641151	**647427**
	0.97%	99.03%	**100.00%**
	1.36%	6.27%	**6.06%**
Total	**461454**	**10228195**	**10689649**
	4.32%	**95.68%**	**100.00%**
	100.00%	**100.00%**	**100.00%**

Copy
Select All

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Correlations

The **Correlations** view of the dataset object allows you to explore how or to what extent fields in your dataset are associated with each other. It displays *correlation coefficients* (default option) or *covariances* for the fields you select.

You have an option of displaying these statistics in the matrix form, in pairs, or in a chart. See the topic Correlations View for more details.

Covariance and Pearson correlation coefficients measure the degree of linear association between pairs of variables.

The values of correlation coefficients are always between -1 and +1. The value of +1 indicates that two fields are perfectly related in a positive linear sense, -1 indicates that they are perfectly related in a negative linear sense, and 0 means there is no linear relationship between the two fields. The correlation coefficient of two variables is obtained by dividing their covariance by the product of their standard deviations.

Spearman's correlation is a special case of the Pearson correlation applied to the ranks of the variable values rather than the values themselves. It is a measure of monotone association between two variables.

Missing values are handled on a pairwise basis, i.e., the results of the correlations or covariance computation treats missing values in a pairwise mode. For a given row, as long as the two intersecting values are not missing, the numbers contribute to the calculation.

When you select the Correlations tab of a dataset for the first time, the **Options** dialog appears automatically, allowing you to select the desired fields, the type of statistics to be displayed, and the way to treat discrete and continuous variables (see Options) . You can also call up the Options dialog manually in any of the following ways:

- Select **Tools | Options...** from the menu
- Click the **Options** icon 

in the Toolbar

Handling Large Datasets

The calculation of correlations may take a very long time for large datasets with high-cardinality variables. The binning option is provided for the computations of correlations (all types except Pearson) in this case. It uses an approximation technique to improve performance and reduce the calculation time. If specified, the binning option is applied to the variables with cardinality exceeding the specified number of bins. This option is truned off by default.

When this option is turned on, binning is applied to high-cardinality variables and sampling is applied if the dataset has a very large number of records.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

See the topic Correlations View for detailed description of the view.

See also: Multicollinearity Analysis

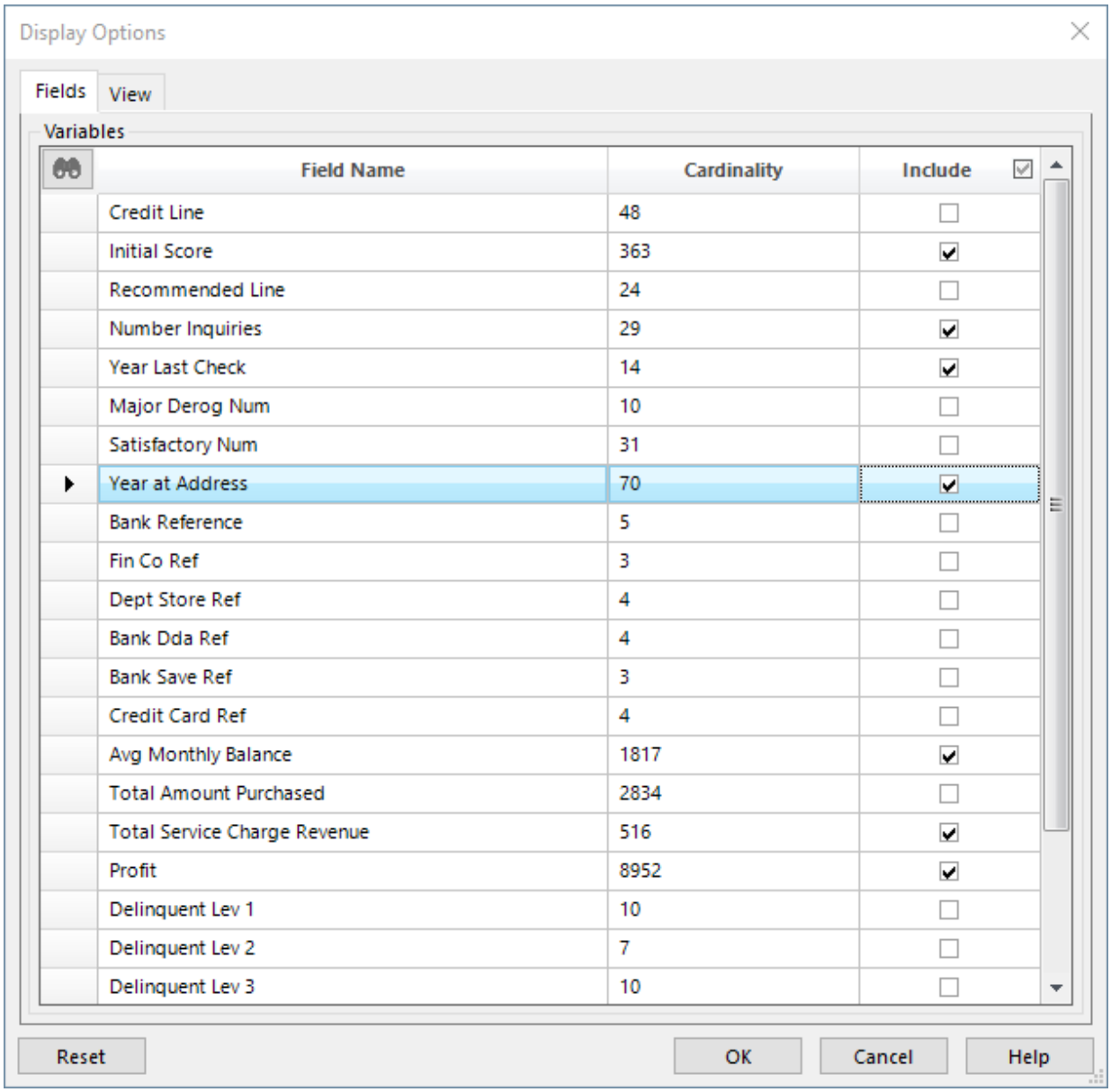
Correlation Options

The Display Options dialog for the Correlations view allows you to select the desired fields and the additional statistics to be displayed. For general information about the Correlations view, see the topic About the Correlations View.

Filed Selection

Select the fields to analyze for correlation. If you want to select or deselect all fields, click the checkbox in the **Include** column header. Note that the calculation of correlations for a very large number of variables may take substantial time and memory resources. To sort the list of variables by cardinality, click on the Cardinality column header, if necessary.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



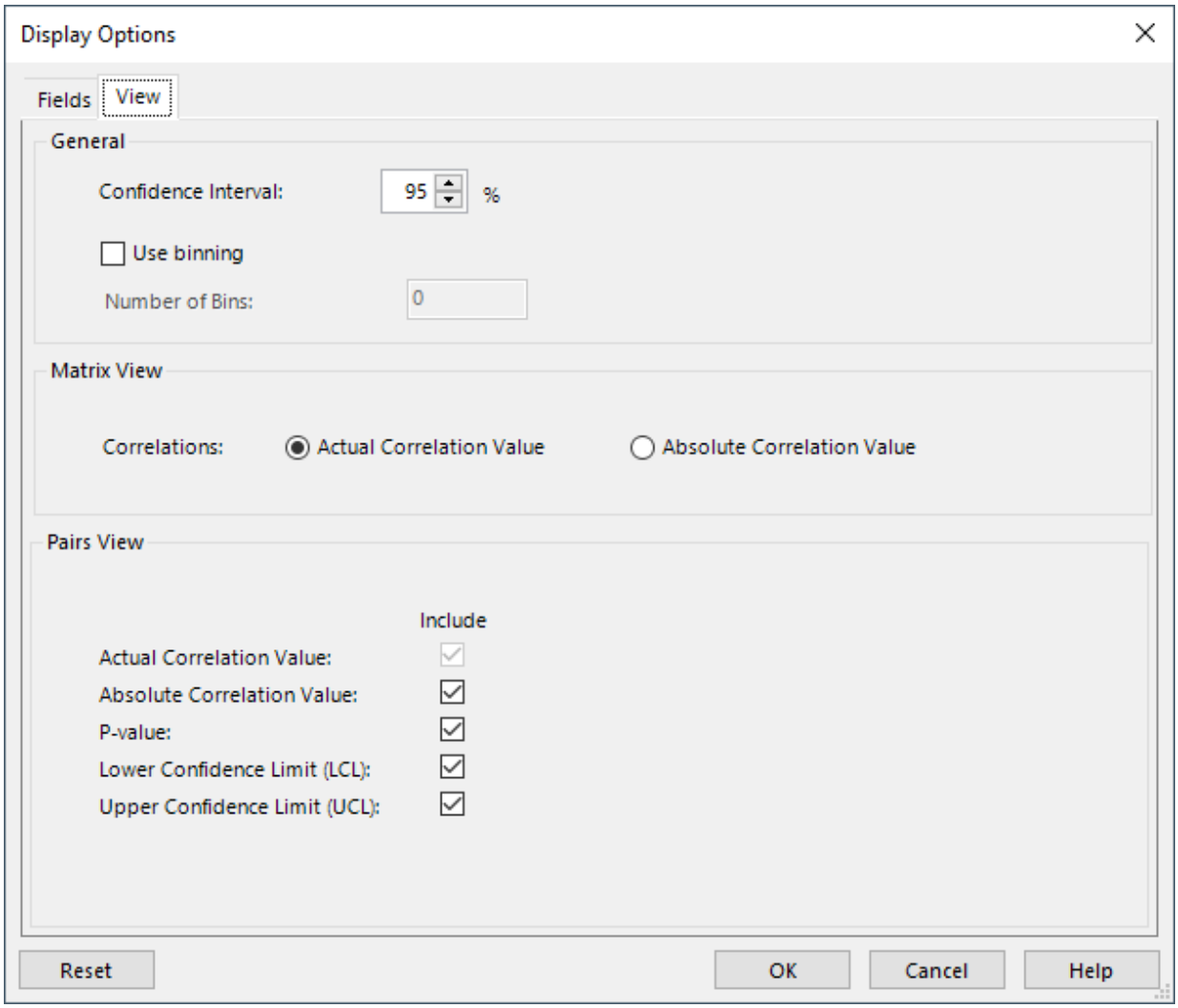
View Options

Click the **View** tab of the Display Options dialog to specify the confidence interval and the binning options, select whether to display absolute or actual correlation values in the Matrix View, and choose the statistics to display in the Pairs view. The available statistics for the Pairs view include the actual correlation value (always displayed), absolute correlation value, ***p***-value, and lower & upper confidence limits.

Use the **Binning** option for large datasets. The calculation of correlation coefficients, especially Kendall's Tau and Hoeffding's *D,* may take a very long time for large datasets with high-cardinality variables. The binning option is applicable to all correlation types except Pearson. It uses an approximation technique to improve performance and reduce the calculation time. If you select "Use binning", specify the number of bins. Binning is applied to high-cardinality variables whose cardinality

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

exceeds the specified number of bins, and sampling is applied if the dataset has a very large number of records.



Reset

The **Reset** button undoes the changes you have made in the Display Options dialog, i.e., resets the settings to those of the currently displayed Correlations view.

See also: About the Correlations View

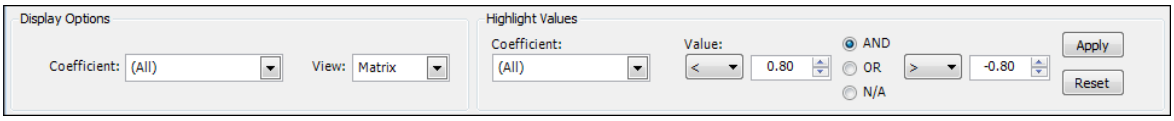
Correlations View

The correlation coefficients are shown based on the parameters specified in the Display Options dialog (see the topic Correlations: Options for details).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The **View** drop down list provides three ways to display correlation coefficients: *Matrix*, *Pairs*, and *Chart*.

From the **Coefficient** list, select the type of coefficients to display (Pearson, Spearman, Kendall's Tau-b, or Hoeffding's).



- **Matrix View**

  Shows the correlation or covariance matrix for the selected fields.

	WOE_Residential Status	WOE_Initial Score	WOE_Region	WOE_age
WOE_Residential Status	P: 1	P: 0.31931	P: 0.07141	P: 0.12701
	S: 1	S: 0.49013	S: 0.01134	S: 0.17247
	K: 1	K: 0.40827	K: 0.0104	K: 0.14673
	H: 0.27447	H: 0.05062	H: 5E-05	H: 0.0061
WOE_Initial Score	P: 0.31931	P: 1	P: 0.24811	P: 0.35921
	S: 0.49013	S: 1	S: 0.31464	S: 0.35671
	K: 0.40827	K: 1	K: 0.28344	K: 0.29778
	H: 0.05062	H: 0.41376	H: 0.00824	H: 0.02959
WOE_Region	P: 0.07141	P: 0.24811	P: 1	P: 0.03486
	S: 0.01134	S: 0.31464	S: 1	S: 0.04879
	K: 0.0104	K: 0.28344	K: 1	K: 0.04379
	H: 5E-05	H: 0.00824	H: 0.037	H: 0.00016
WOE_age	P: 0.12701	P: 0.35921	P: 0.03486	P: 1
	S: 0.17247	S: 0.35671	S: 0.04879	S: 1
	K: 0.14673	K: 0.29778	K: 0.04379	K: 1
	H: 0.0061	H: 0.02959	H: 0.00016	H: 0.46593

- **Pairs**

  Each pair of variables and their correlation or covariance is displayed in a single row.

			Pearson				
	Variable 1	Variable 2	Correlation	Absolute	P-value	LCL	UCL
▶	WOE_Residential Status	WOE_Initial Score	0.31931	0.31931	0	0.30896	0.32957
	WOE_Residential Status	WOE_Region	0.07141	0.07141	0	0.05998	0.08281
	WOE_Residential Status	WOE_age	0.12701	0.12701	0	0.1157	0.13828
	WOE_Initial Score	WOE_Region	0.24811	0.24811	0	0.23731	0.25885
	WOE_Initial Score	WOE_age	0.35921	0.35921	0	0.34917	0.36916
	WOE_Region	WOE_age	0.03486	0.03486	0	0.0234	0.04632

- **Chart**

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Correlation or covariance matrix is displayed in the form of a chart. The variables are displayed along the horizontal axes and their correlations or covariances are shown on the vertical axis. You can choose from different chart types and display options from the Chart menu or from the context menu brought up by right-clicking on the chart.

You have an option of displaying only the charts for a particular variable (chosen from the *Variable* drop-down list) or ignoring the correlation or covariance of a variable with itself (the Suppress Same Column checkbox).

1.

Having identified and highlighted the desired variables based on the correlation analysis in this view, you can run the **Insert | Variable List** command to save them in a variable list for future use. For example, if you want to use only the selected variables to train a model, use the field finder button  when selecting the independent variables for the model and import the saved variable list.

See also: About the Correlations View, Correlations: Options, Multicollinearity Analysis

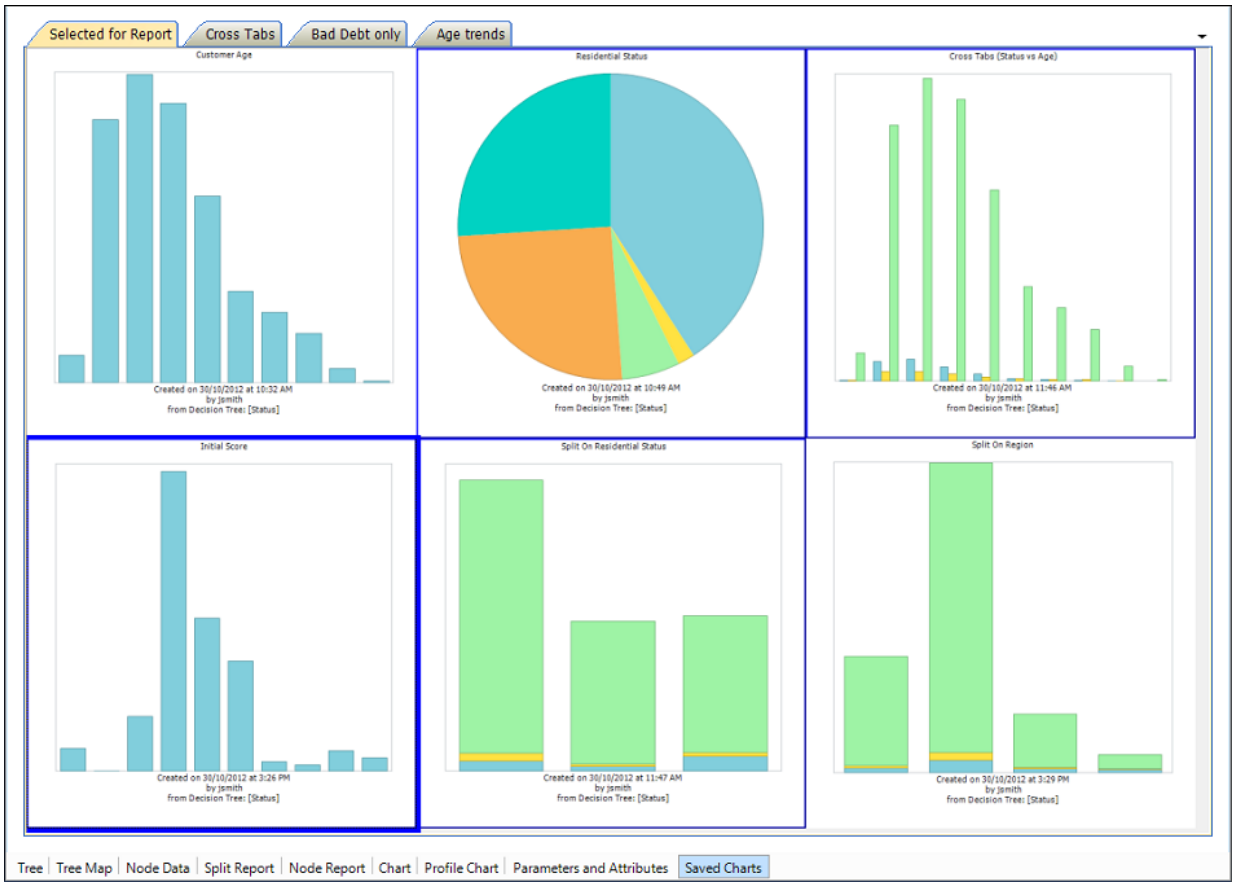
Saved Charts

The Saved Charts tab is available for datasets, decision trees, and strategy trees. This tab represents the library of charts you saved in other views of the object it belongs to.

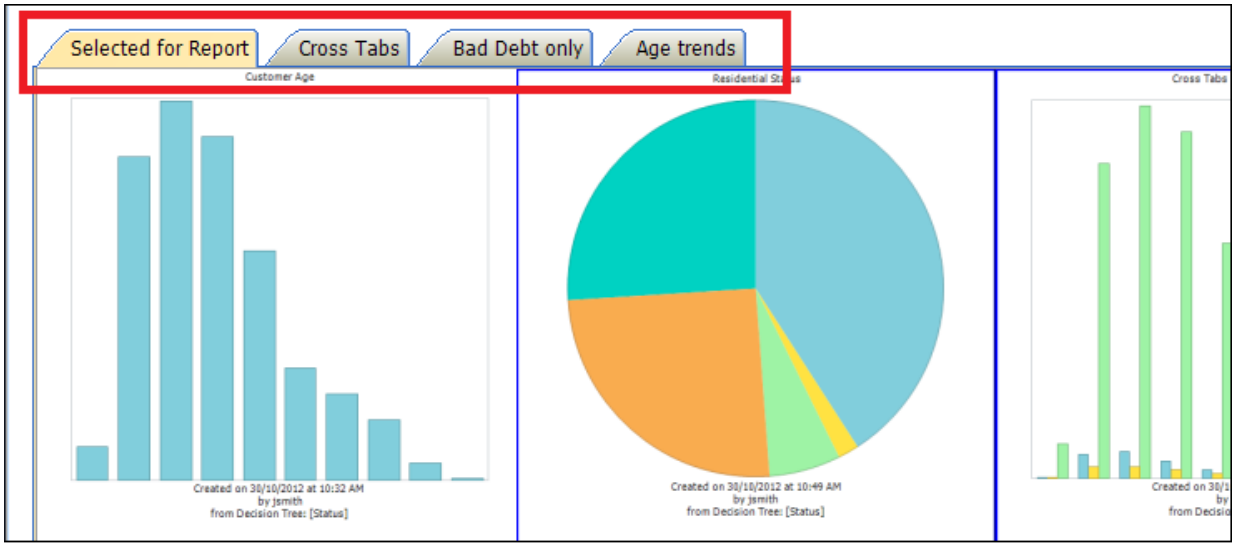
- The Saved Charts tab of a dataset contains charts saved in the Dataset Chart, Segment Viewer, Cross Tabs, and Correlations views of the dataset.
- The Saved Charts tab of a tree contains node charts saved in the Chart view of the tree.

In the Saved Charts tab you can view, organize, and print the charts, or copy them to Microsoft Office documents for reporting.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Charts are organized in groups. Each group is represented by a separate tab at the top of the chart grid. The tabs are highlighted by the red frame in the example screenshot below. Click on the tab names to switch between the chart groups.



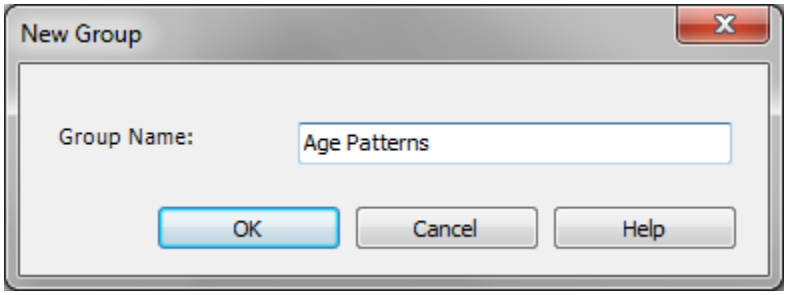
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To view any chart in detail in a separate window, double-click on it and resize its window as necessary. In this window you can view the chart details, copy or print it, etc. For more information, see the topic Single Chart View.

Chart Groups

You can create new chart groups and distribute the charts in these groups.

To create a new chart group, select the **New Group** command from the **Groups** menu, or right-click on the group tab name and select **New** from the context menu. Then specify the desired name in the dialog:



A new blank chart group with the new name will be created.

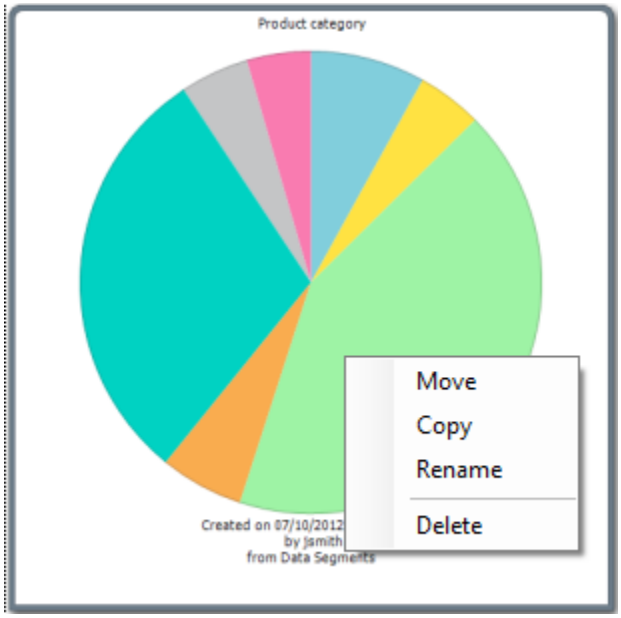
To rename an existing chart group, right-click on the group tab name and select the **Rename** command from the context menu. Enter the new name and click OK.

Operations with Charts

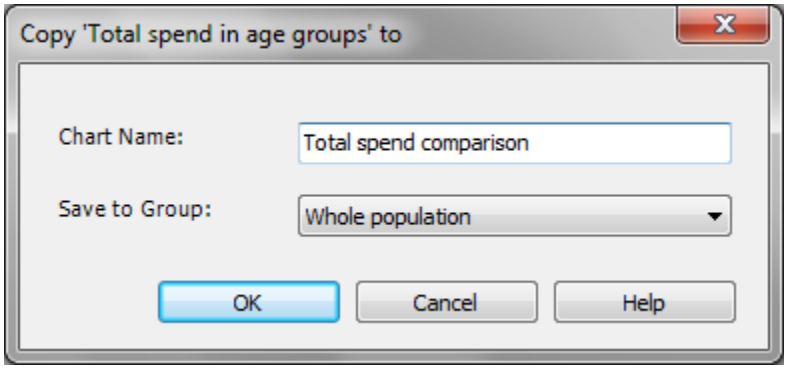
To change the position of any chart within its group, drag and drop it to the desired position.

To delete, copy, rename, or move a chart to a different group, right-click on the chart and select the appropriate command from the context menu.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- To delete a chart, select **Delete** from the context menu.
- To rename a chart, select **Rename** from the context menu, specify the new name for the chart, and click OK.
- Charts can be copied within the same chart group or to a different chart group. To copy a chart, right-click on it and select **Copy** from the context menu. Specify the name for the new chart and select the destination group. The current group is the default destination.



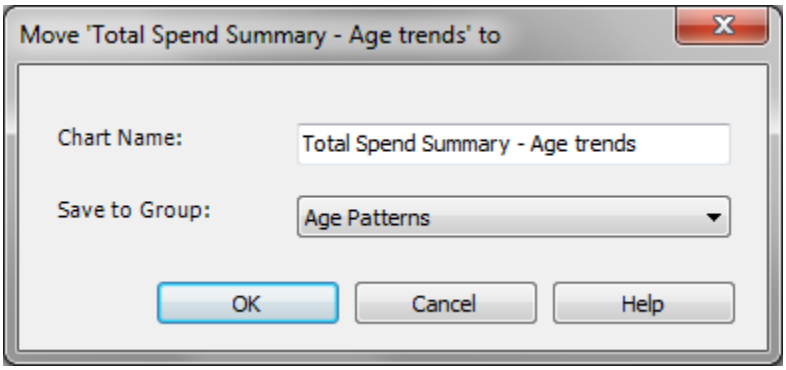
- There are two ways to move a chart to another group:

  a) drag and drop the chart to another group tab, or

  b) right-click on the chart and select **Move** from the context menu.

In the Move dialog, specify the new name for the chart and select the desired destination group from the Save to Group list

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also:

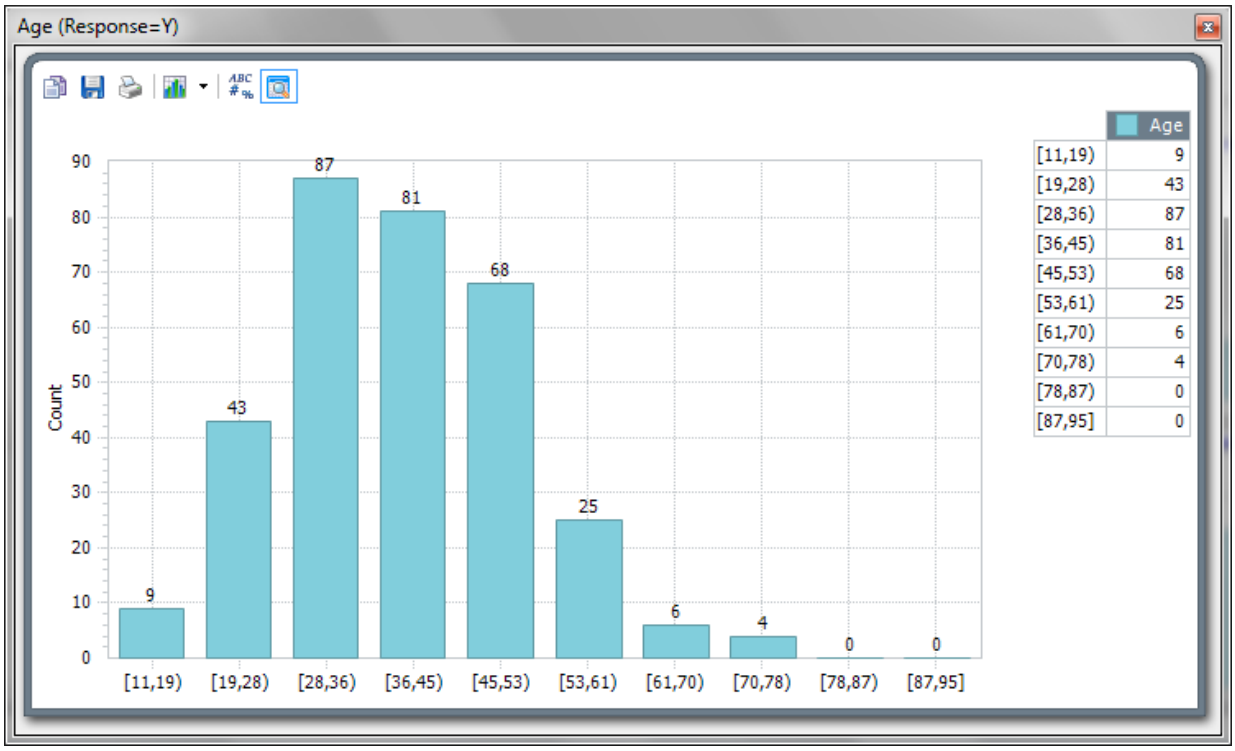
Single Chart View

Single Chart View

Several views in the application contain multiple charts: Segment Viewer in datasets, Chart in trees, and Saved Charts in both datasets and trees. In any of these views, you can open each individual chart in its own separate window, where you can view its details, copy or print it, etc.

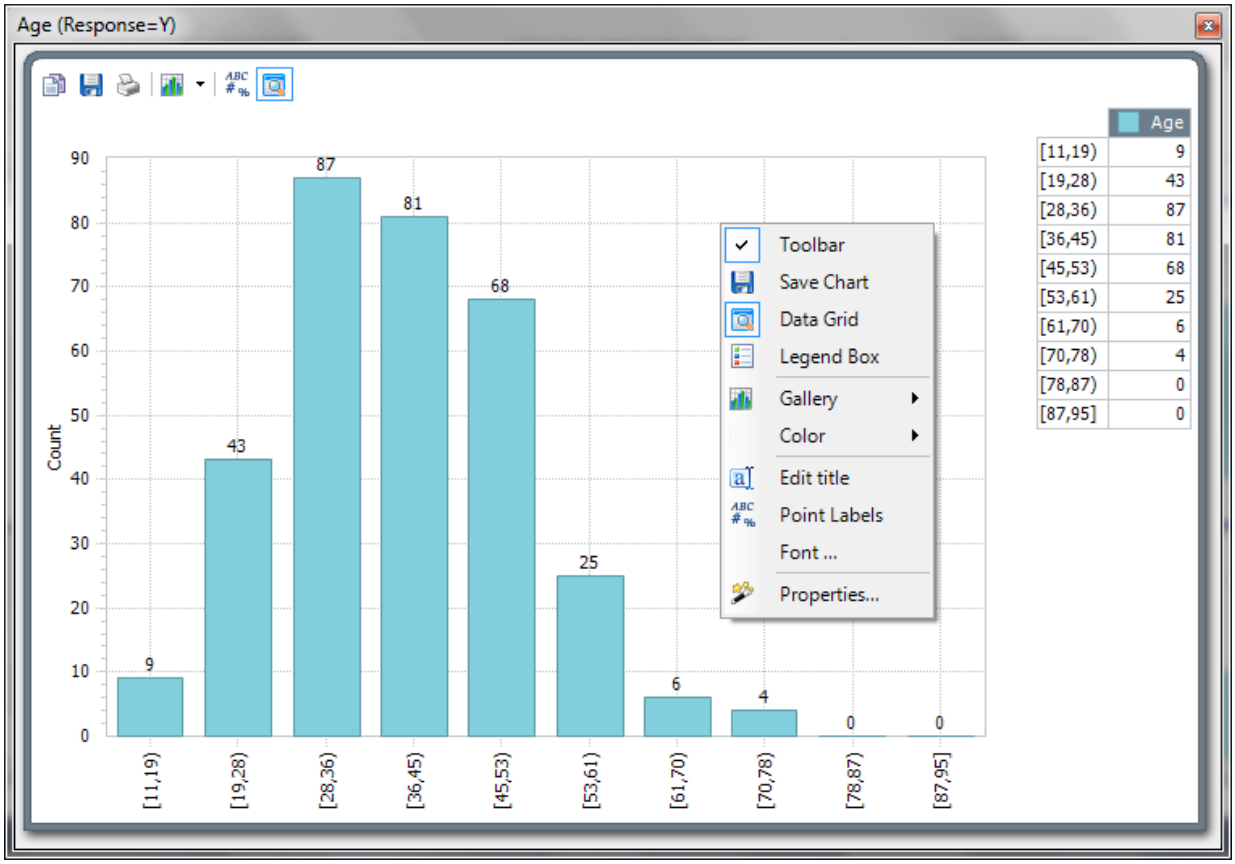
Double-click on the desired chart to view it in a separate window. The window provides a detailed view of the chart and shows the data grid representing the chart data. You can resize the window by clicking and dragging on its corners or borders.

Here is an example of a chart from Segment Viewer opened in its own window.



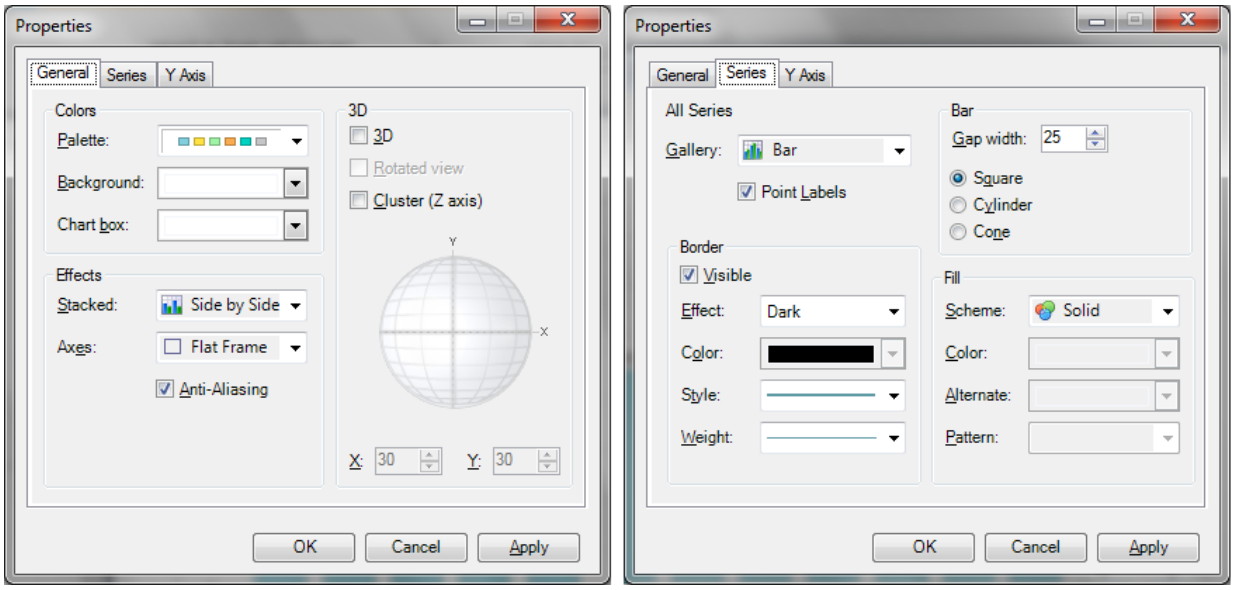
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To change the display properties of the chart, use the chart toolbar or right-click anywhere on the chart background. The following context menu appears when right-clicking on the chart background. For example, select Point Labels to show or hide the values on top of the bars; select Data Grid to show or hide the table with the chart data.

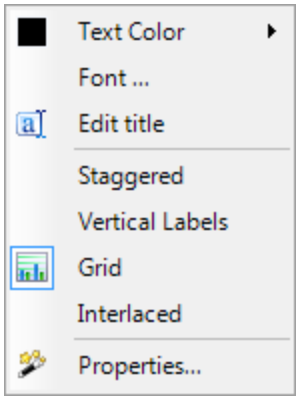


To change advanced properties of the chart, select Properties from the context menu. In the Properties dialog, you can choose the color palette, 3D or 2D view, effects, properties of the axis labels, show or hide grid lines, etc.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

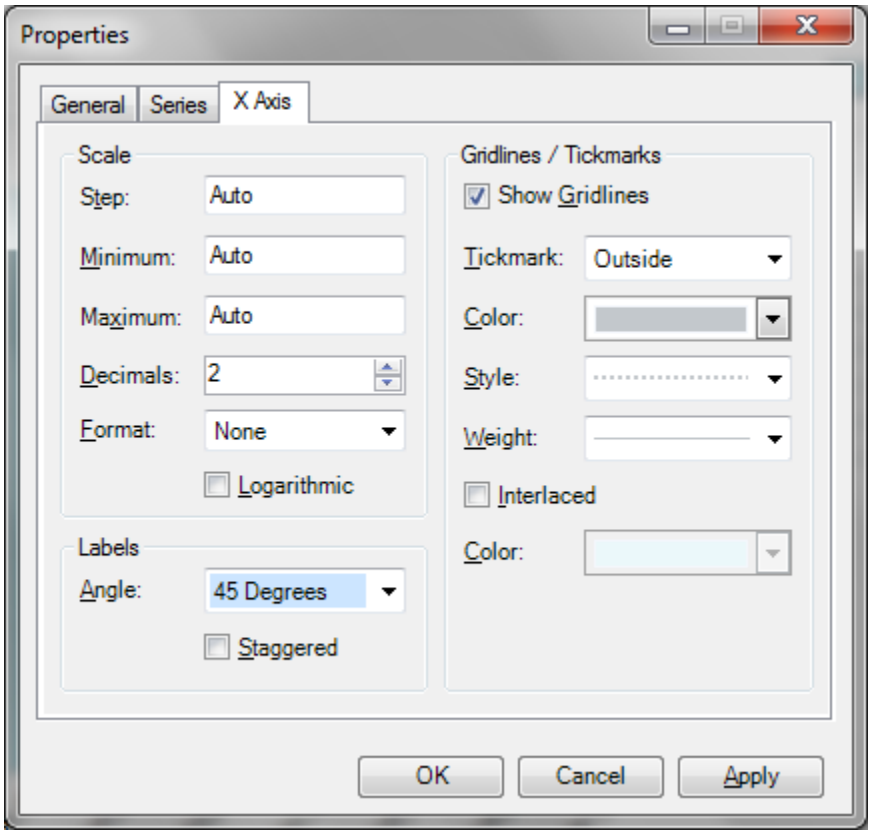


To change the orientation of the axis labels, right-click on any label of the desired axis (vertical or horizontal). The context menu will look like this:



Select *Staggered* or *Vertical Labels* if desired. If you want to place the labels at a 45-degree angle, select *Properties* and set *Angle* to 45 Degrees.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



As a result, the X-axis labels are displayed as follows:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

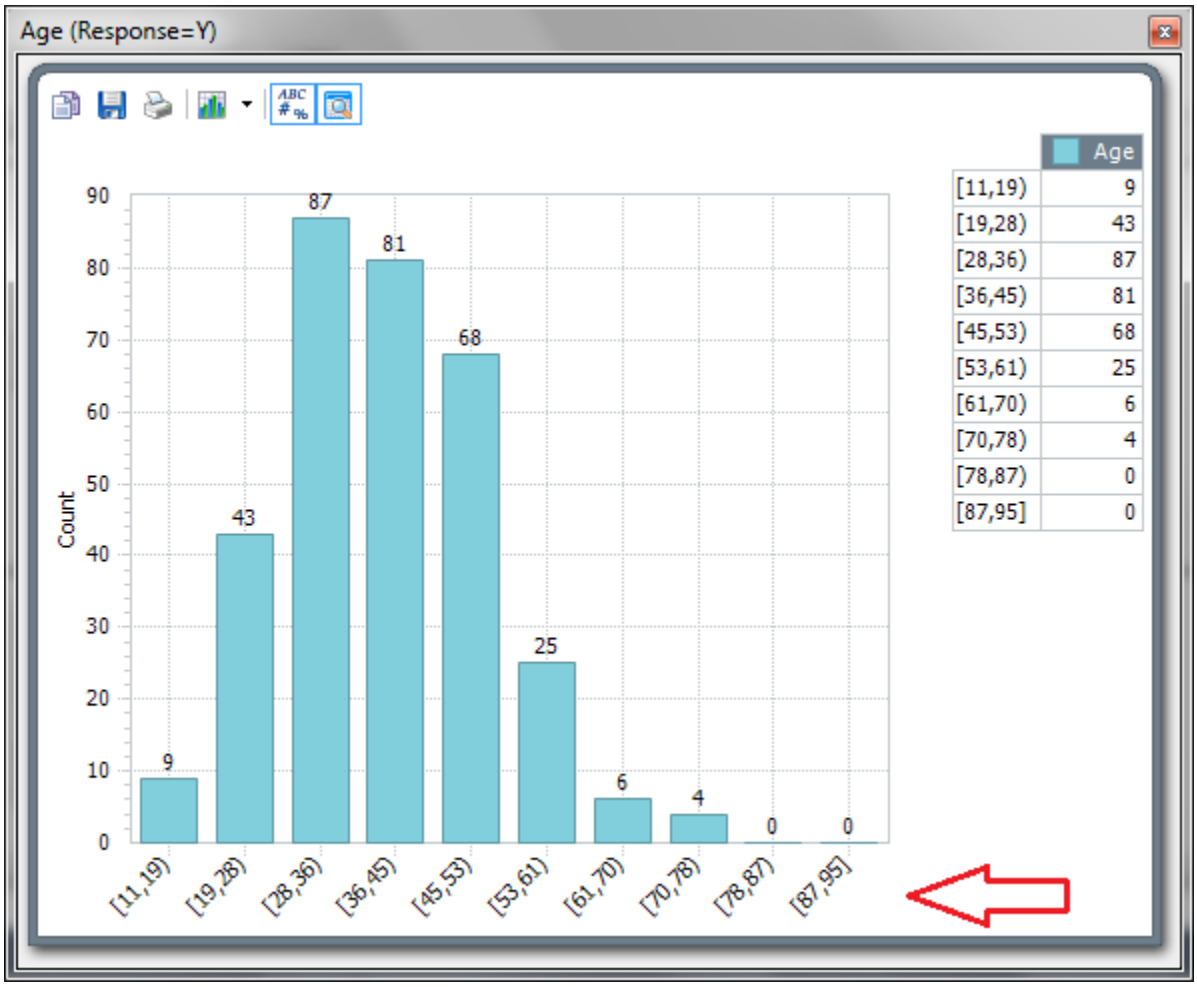


Chart Toolbar

Depending on the charts' parent view - Segment Viewer, Cross Tabs, Chart (in trees), etc., - its toolbar at the top left corner of the chart window can have one of the following two forms:

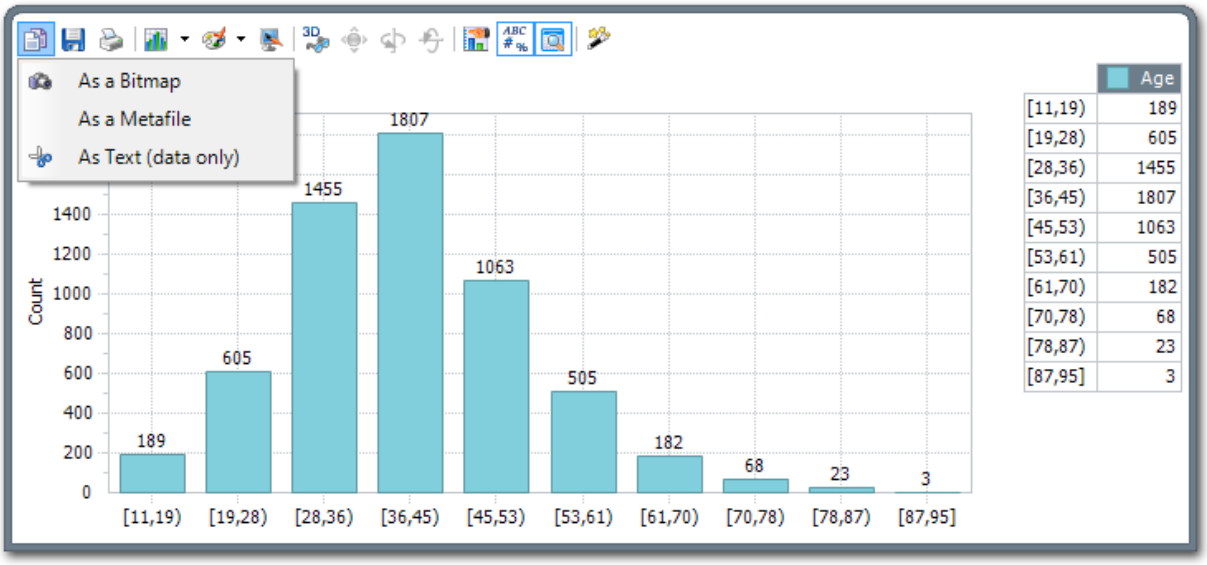




All icons in the Chart toolbar except the *Save*, *Copy*, and *Print* icons are shortcuts to the corresponding options and commands in the Properties dialog described above.

To copy a chart, click on the *Copy* icon in its toolbar, as shown in the screenshot below. Select *As a Bitmap* or *As a Metafile* from the context menu if you want to copy the chart as an image. Select *As Text* (*data only*) if you want to copy the underlying data, which is displayed in the data grid on the right-hand side. Then paste the copied image or text from the clipboard into the desired application.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If you are viewing a chart from the Segment Viewer or Node Chart, you can save it in the Chart Library by clicking on the Save button .

See also: Saved Charts

Interactive Filtering on Datasets

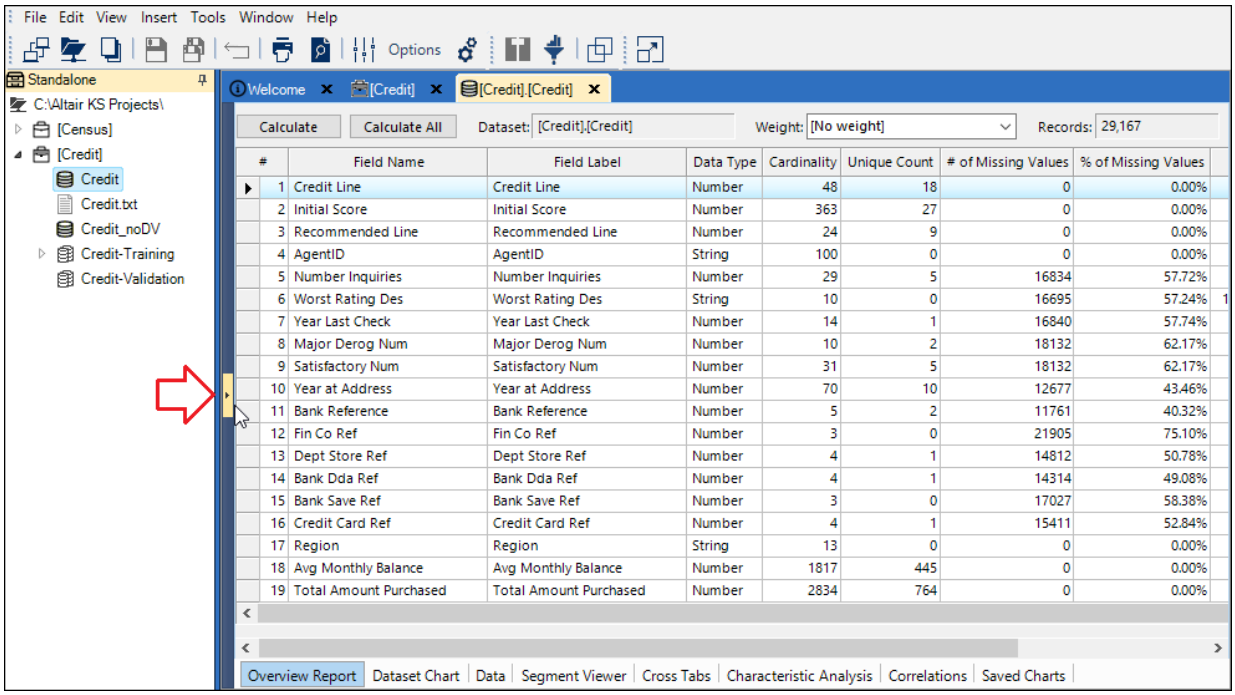
Aside from to the **Filter** node, which creates a new dataset, you can perform interactive filtering within the dataset views - without going back to the workflow and creating a new dataset. A dockable filter panel enabled in all dataset views (tabs) serves this purpose.

For example, when doing exploratory data analysis in a certain dataset view (such as Dataset Chart), you may notice some peculiarities in a subset of records and need to further explore that particular subset in other views (Data tab, etc.). An ideal tool for this task is an interactive filter functionality embedded within each view of the dataset to easily pick out specific records of interest and view them - without going back to the workflow and creating a new dataset with a Filter node, Partition node, or building a decision tree.

The interface of the Filter panel in dataset views is similar to that of the Filter node wizard. While exploring the dataset, you can easily switch between the filtered and unfiltered views with a single click. The Filter icon in the toolbar serves both as a toggle and an indicator that the filter is applied.

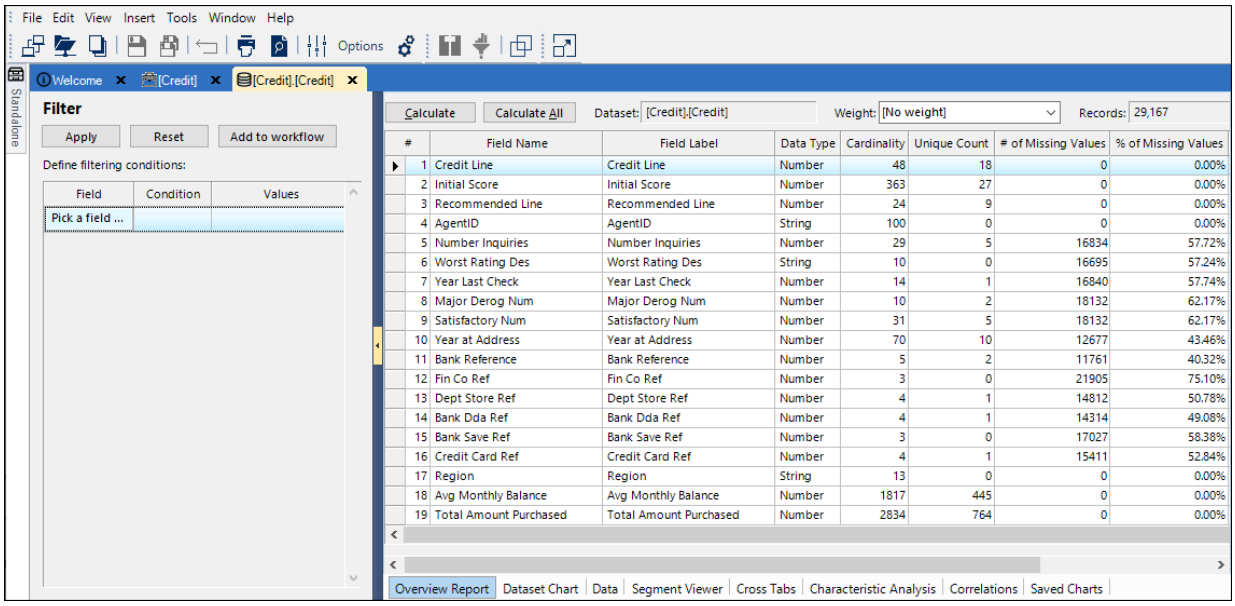
To open the interactive **Filter** panel in any dataset view, click on the small arrow in the middle of the separator bar between the Project Explorer panel and the current dataset view, as shown in the screenshot below.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



#	Field Name	Field Label	Data Type	Cardinality	Unique Count	# of Missing Values	% of Missing Values
1	Credit Line	Credit Line	Number	48	18	0	0.00%
2	Initial Score	Initial Score	Number	363	27	0	0.00%
3	Recommended Line	Recommended Line	Number	24	9	0	0.00%
4	AgentID	AgentID	String	100	0	0	0.00%
5	Number Inquiries	Number Inquiries	Number	29	5	16834	57.72%
6	Worst Rating Des	Worst Rating Des	String	10	0	16695	57.24%
7	Year Last Check	Year Last Check	Number	14	1	16840	57.74%
8	Major Derog Num	Major Derog Num	Number	10	2	18132	62.17%
9	Satisfactory Num	Satisfactory Num	Number	31	5	18132	62.17%
10	Year at Address	Year at Address	Number	70	10	12677	43.46%
11	Bank Reference	Bank Reference	Number	5	2	11761	40.32%
12	Fin Co Ref	Fin Co Ref	Number	3	0	21905	75.10%
13	Dept Store Ref	Dept Store Ref	Number	4	1	14812	50.78%
14	Bank Dda Ref	Bank Dda Ref	Number	4	1	14314	49.08%
15	Bank Save Ref	Bank Save Ref	Number	3	0	17027	58.38%
16	Credit Card Ref	Credit Card Ref	Number	4	1	15411	52.84%
17	Region	Region	String	13	0	0	0.00%
18	Avg Monthly Balance	Avg Monthly Balance	Number	1817	445	0	0.00%
19	Total Amount Purchased	Total Amount Purchased	Number	2834	764	0	0.00%

The Filter panel remains open when you switch between the dataset tabs (Overview Report, Dataset Chart, Data, Segment Viewer, etc.), and can be open or closed from any of these tabs.



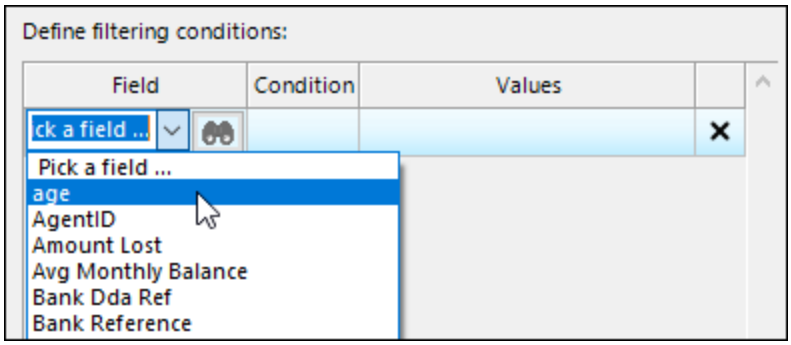
#	Field Name	Field Label	Data Type	Cardinality	Unique Count	# of Missing Values	% of Missing Values
1	Credit Line	Credit Line	Number	48	18	0	0.00%
2	Initial Score	Initial Score	Number	363	27	0	0.00%
3	Recommended Line	Recommended Line	Number	24	9	0	0.00%
4	AgentID	AgentID	String	100	0	0	0.00%
5	Number Inquiries	Number Inquiries	Number	29	5	16834	57.72%
6	Worst Rating Des	Worst Rating Des	String	10	0	16695	57.24%
7	Year Last Check	Year Last Check	Number	14	1	16840	57.74%
8	Major Derog Num	Major Derog Num	Number	10	2	18132	62.17%
9	Satisfactory Num	Satisfactory Num	Number	31	5	18132	62.17%
10	Year at Address	Year at Address	Number	70	10	12677	43.46%
11	Bank Reference	Bank Reference	Number	5	2	11761	40.32%
12	Fin Co Ref	Fin Co Ref	Number	3	0	21905	75.10%
13	Dept Store Ref	Dept Store Ref	Number	4	1	14812	50.78%
14	Bank Dda Ref	Bank Dda Ref	Number	4	1	14314	49.08%
15	Bank Save Ref	Bank Save Ref	Number	3	0	17027	58.38%
16	Credit Card Ref	Credit Card Ref	Number	4	1	15411	52.84%
17	Region	Region	String	13	0	0	0.00%
18	Avg Monthly Balance	Avg Monthly Balance	Number	1817	445	0	0.00%
19	Total Amount Purchased	Total Amount Purchased	Number	2834	764	0	0.00%

Defining the Filtering Conditions

Define the filtering conditions by adding them in the grid of the Filter panel, one row per field. This is similar to common query-building interfaces. First, click in the **Field** column to pick a field. Select

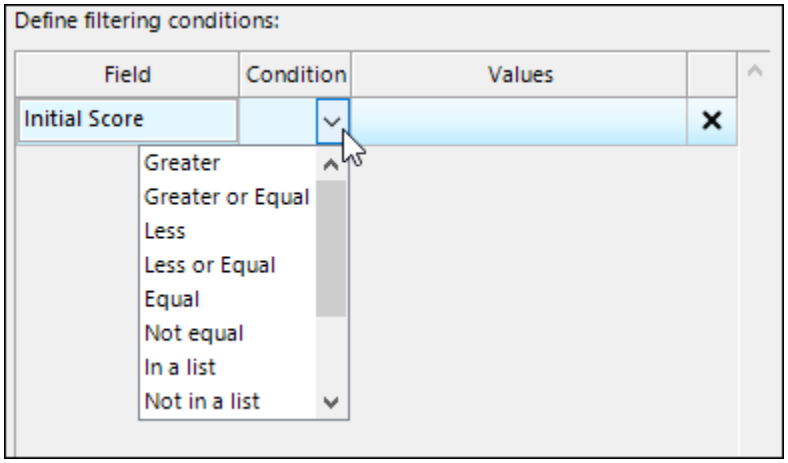
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

the desired field from the drop-down list. If there are hundreds of fields, you can use the binoculars button to open the Field Finder.



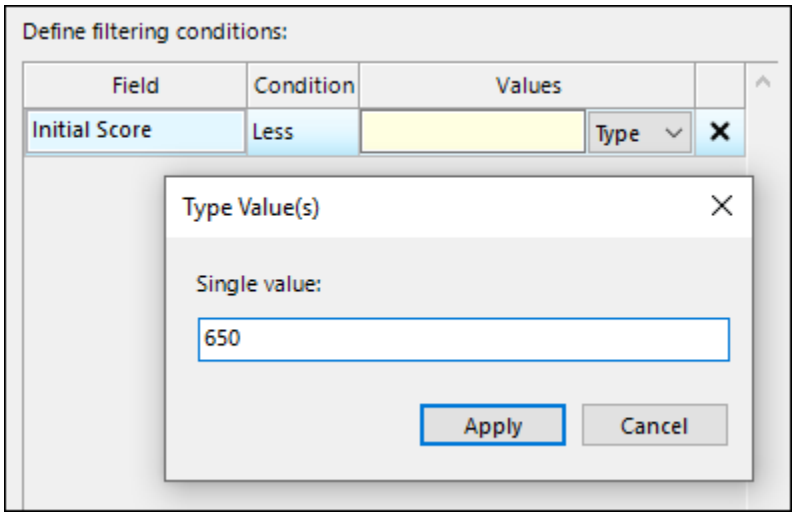
Click in the **Condition** column and select the desired condition from the drop-down list. The available conditions are:

- for numeric and Date/Time variables: *Greater / Greater or equal / Less / Less or equal / Equal / Not equal / In a list / Not in a list / Between / Not between / Is null / Is not null*
- for String and Boolean variables: *Equal / Not equal / In a list / Not in a list / Is null / Is not null*



Click in the **Values** column to specify the values for the condition. A drop-down list provides two options: "*Select*" and "*Type*". Choose "*Type*" to enter a value manually. Otherwise, choose "Select" to choose from the list of all values of the variable.

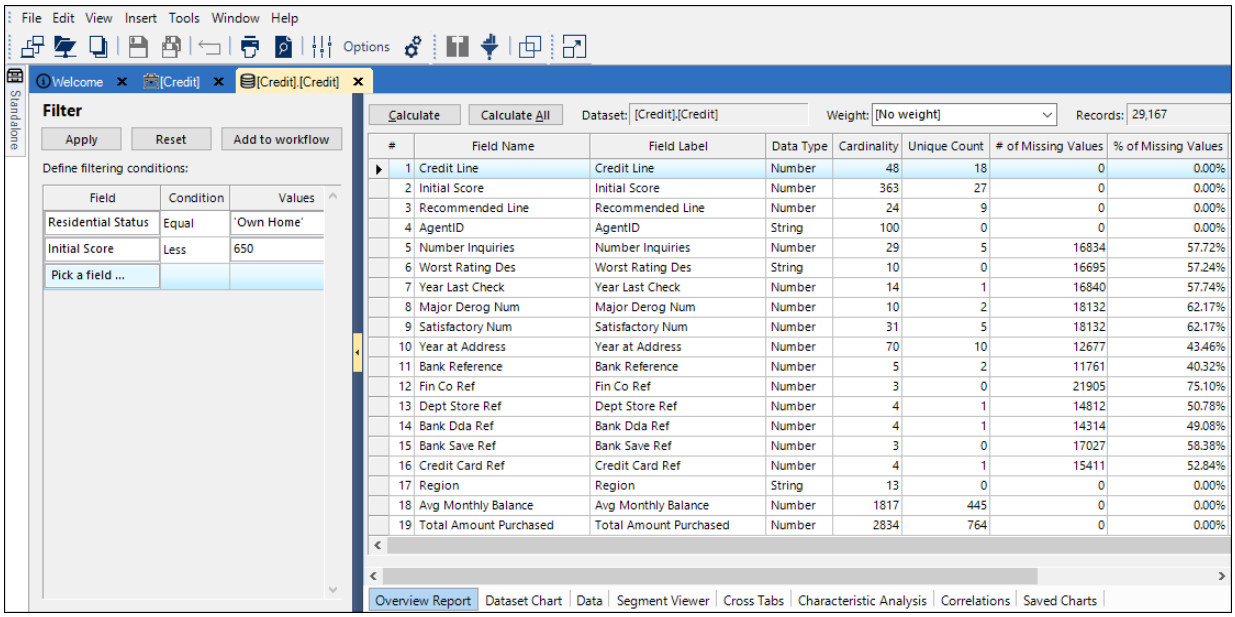
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Once the condition is specified, a new empty row will be added automatically to get the input for the next condition. Add more conditions if required.

"**AND**" is the implied operator for combining the conditions. Conditions like `((x < 10) OR (x > 100))` for numeric and Date/Time variables can be entered in a single row using the option "***Not between***".

To define filters with more complex conditions and arbitrary SQL expressions, use the Filter node in the **Manipulate** palette instead.

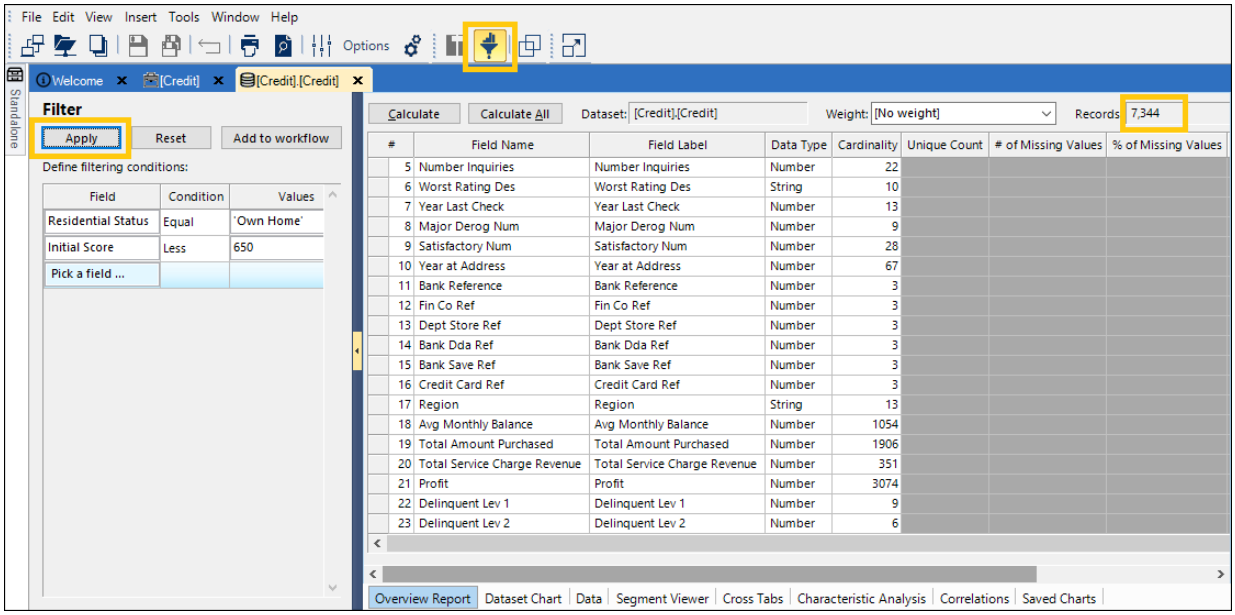


Applying the Filter

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Once you finish defining the conditions, clicks the **Apply** button. The filter will be applied to the data in all dataset tabs.

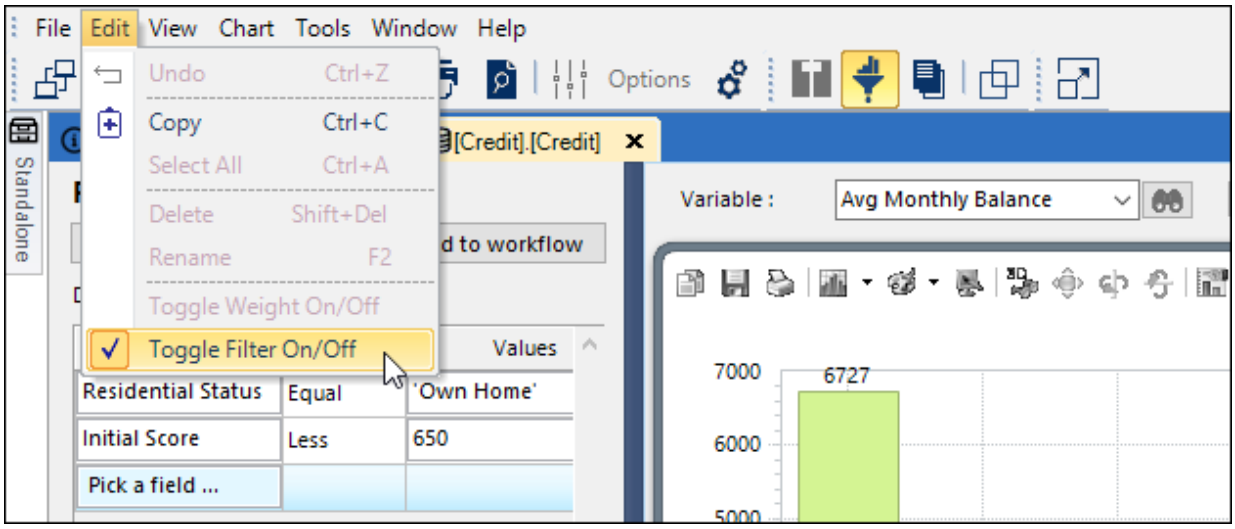
In the Overview Report tab, the cardinalities and the total number of records are updated, while any previously calculated summary statistics are reset, so you may need to click **Calculate All** to calculate them again if required. In other tabs, such as Cross Tabs, you may need to reapply their Options. The previously saved charts in the Saved Charts tab are always static and are not affected by the filter.



Once the filter is applied, the Filter icon  in the toolbar becomes highlighted:  . It serves both as a toggle and an indicator that the filter is applied. To toggle the filter on or off to switch between the original and filtered data in all views, click the Filter icon. When the filter is not applied, the icon is gray again. A tooltip over the Filter icon indicates that the "*Filter is on*" when hovering the mouse over it. When there are no filtering conditions defined, the Filter icon is disabled.

You can also switch between the original and filtered data using the command "**Toggle Filter On/Off**" in the **Edit** menu:

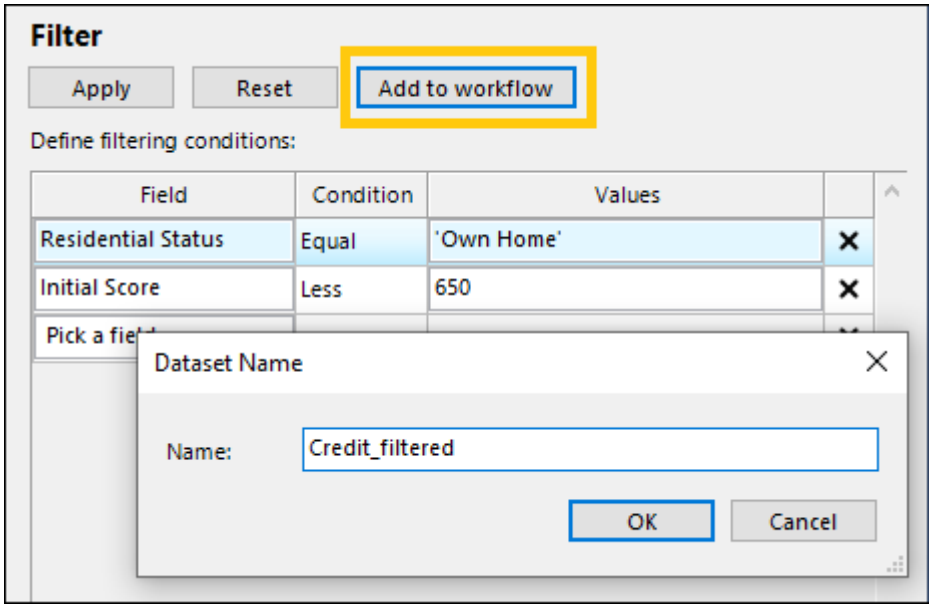
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



To clear the filter, click the **Reset** button. You will be prompted to confirm the reset, as the filter conditions will be permanently cleared.

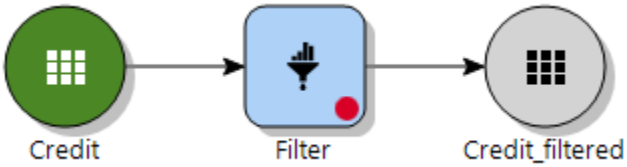
Adding the Filter to the Workflow

If you want to continue defining the filter in a separate workflow node (for example, if you need to specify more complex conditions using SQL expressions), click the button **Add to workflow** in the Filter panel. You will be prompted for the output dataset name. Specify the name for the output dataset and click **OK**.

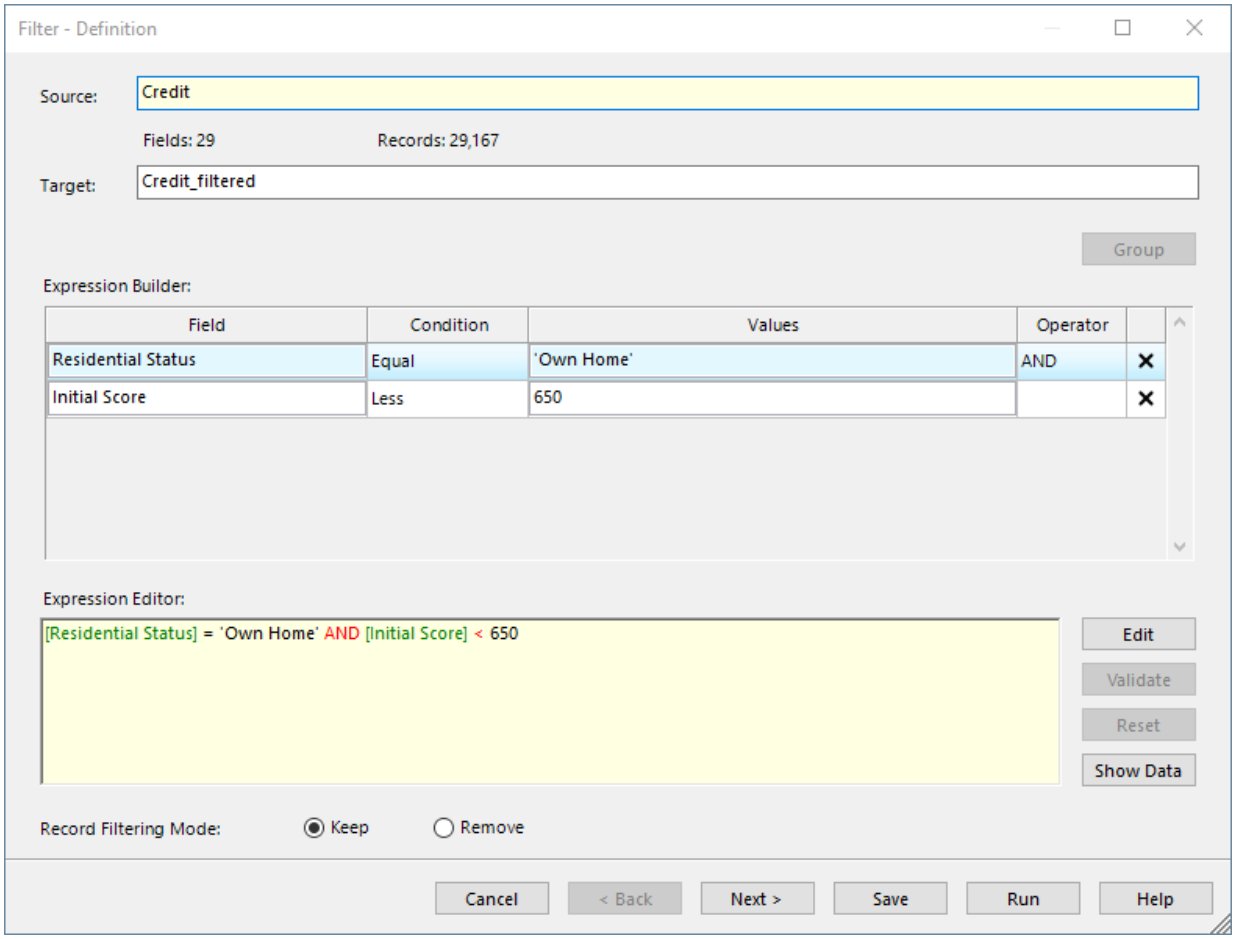


This will create a new **Filter** node on the workflow canvas, with the current dataset as the input node, and the conditions will be copied into the Filter node wizard. Note that the new Filter node will not be run yet.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



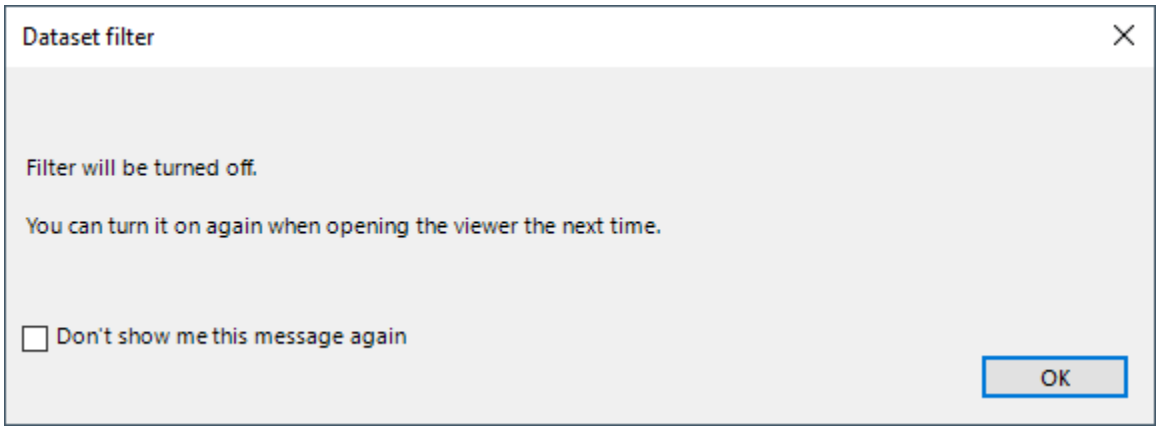
Double-click on the new **Filter** node to open the wizard and continue to specify the conditions. See the **Filter** node topic for details regarding its interface.



Filter Persistence

When you close the dataset, the project, or the application while the interactive filter is applied, the filter conditions are saved, but the filter will be automatically turned off if it is currently on. You will get a warning message, which you can choose to never show again. The next time you open the dataset, the filter will be off, but you can open it and turn it back on if necessary.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also: Filter

Outlier Detector

The **Outlier Detector** node in the **Profile** palette performs automatic detection of outliers in a dataset. An outlier record is a record with an extreme value in at least one variable. Several distance measures are provided for quantifying the degree to which values are outliers. In the output dataset, the *Outlier Flag* column with values *Yes / No* is created for each variable tested for outliers, as well as for the entire record. This can subsequently be used for data quality control, anomaly / fraud detection, and data manipulation to prevent a disproportionate effect of outliers on modeling.

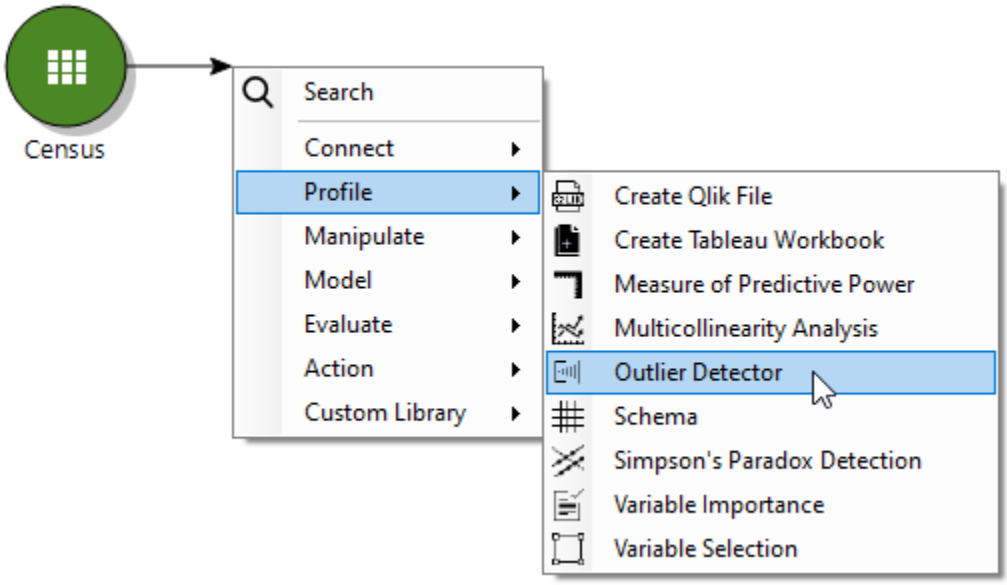
To add an Outlier Detector operation in your workflow:

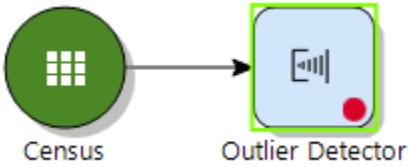
- Drag a connection arrow from a dataset node to any area on the Workflow canvas background. From the context menu, select **Profile --> Outlier Detector** as shown in the example below. A new **Outlier Detector** node will be created, and the wizard will open.

  OR:

- Drag the **Outlier Detector** node from the **Profile** palette onto the canvas and connect the input dataset to the node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



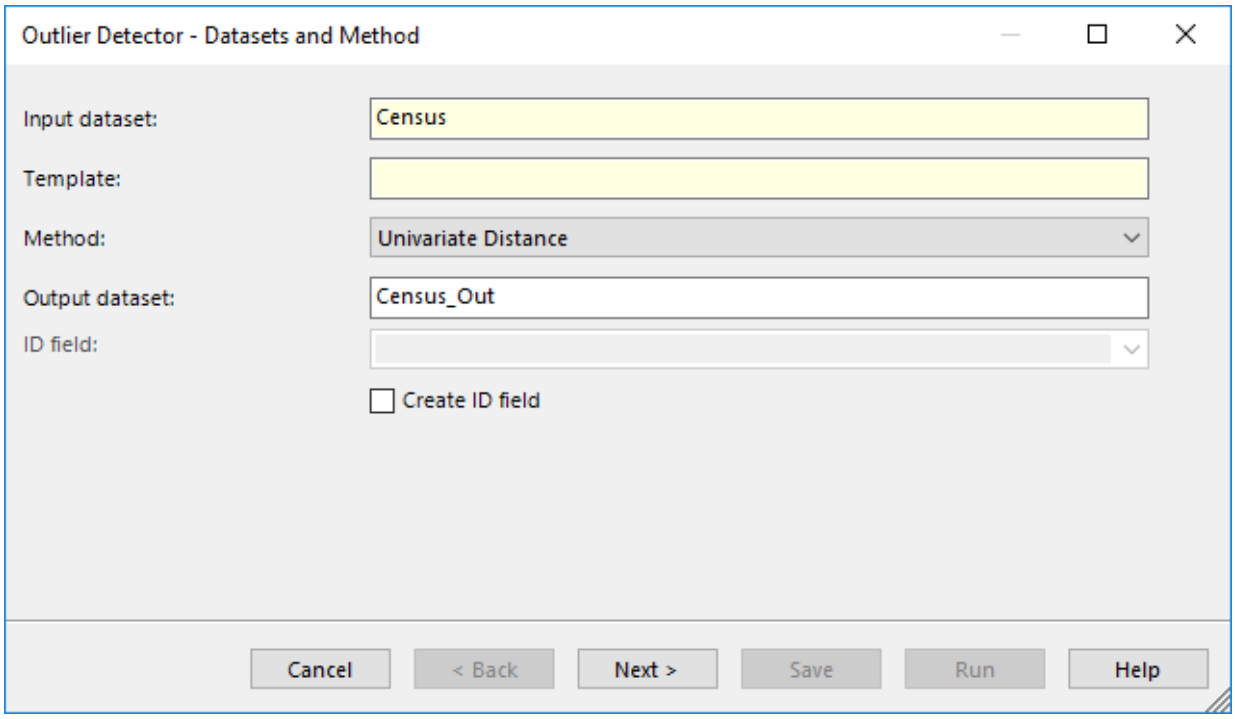


To open the node wizard, double-click the Outlier Detector node or right-click on it and select **Modify**.

In the first page of the wizard:

- specify the name for the output dataset, which will contain the original input dataset fields, outlier flag columns, and optional distance and reasons columns.
- select the outlier detection method (four methods are available; see the description of each method below)
  - *Univariate Distance* (default)
  - *Mahalanobis Distance*
  - *Tree*
  - *Clustering*
- if desired, identify the ID field in the input dataset or select the option "Create ID field"

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



A brief summary of available methods:

Univariate Distance	For each variable tested for outliers (*focal variables*), the user must first select the desired measure of univariate distance. A record will be flagged as an outlier if the value in any of the selected focal variables is greater than X distance units away from the centre of the distribution, where the cutoff value, X, is specified by the user. This method is selected by default. See the details below.
Mahalanobis Distance	A record will be flagged as an outlier if it is greater than X multivariate Mahalanobis distance units away from the centroid of the focal variables, where the cutoff value, X, is specified by the user in the next page.
Tree	A decision tree is automatically grown using the parameters specified by the user in the next wizard page. Then the method finds all terminal nodes with fewer than N records and flags all records in all such nodes as outliers. The number N is specified by the user in the last page of the wizard.
Clustering	A Cluster Analysis model is run using the parameters specified by the user in the next wizard page. Then the method finds all records that either belong to clusters with fewer than N records or do not belong to any cluster, and flags all such records as outliers. The number N is specified by the user in the last page of the wizard.

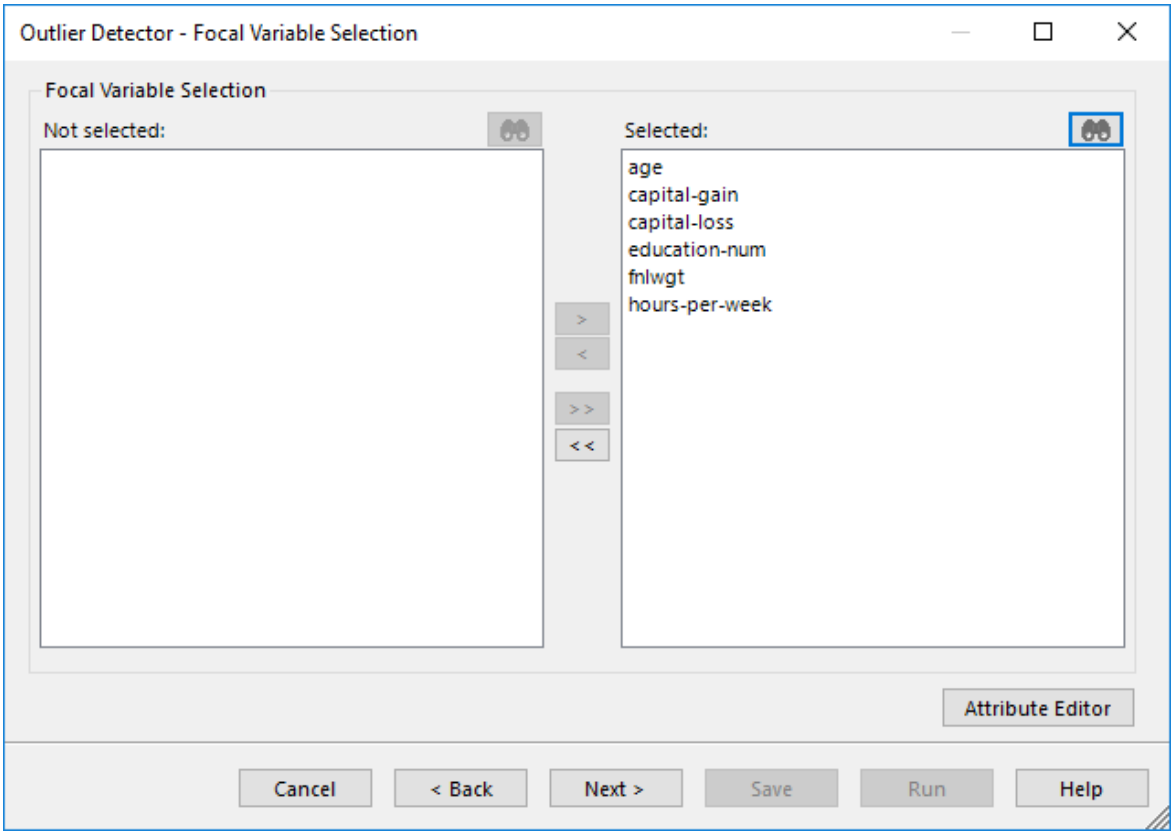
The next steps of the wizard depend on the selected method.

Univariate Distance

If you selected the *Univariate Distance* method (default), the next step is to select *focal variables* - the variables to be tested for outliers - and specify the distance measures and cut-off values. Outliers will be reported individually for each variable, and the input record will be

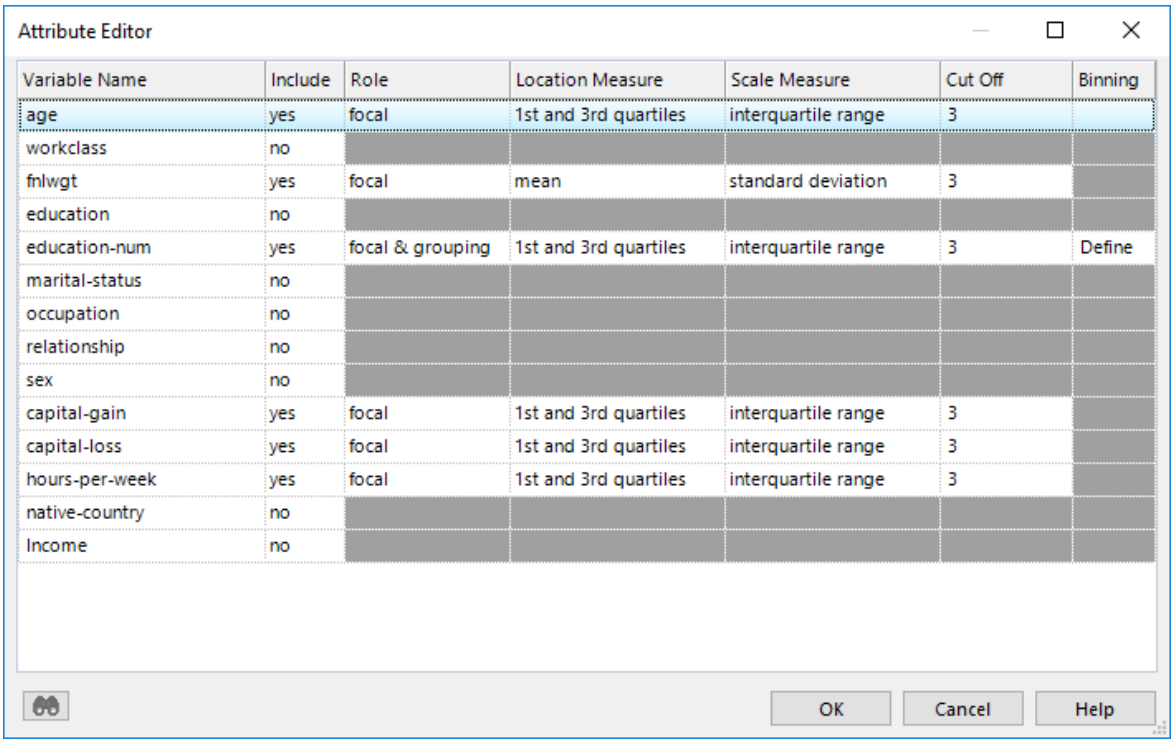
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

flagged as an outlier (or not) as a result. Only variables of number, date, and timestamp types can be selected as focal.



Click the **Attribute Editor** button to define the roles, measures, cut-off values and binning for the variables.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The following roles can be assigned to variables:

- *Focal*
- *Focal & grouping*
- *Grouping*

The roles *focal* and *focal & grouping* are applicable only to continuous variables (numeric, dates, and timestamps). The *grouping* role is applicable to all types of variables.

For focal variables you can define Location Measure, Scale Measure, and Cut Off. For grouping variables you can define the binning.

Location Measures:

- mean
- median
- 1st and 3rd quartiles (default)

  The choice of location measure determines how the centre of the distribution is quantified. The mean and median measures identify the centre with the values of these aggregation functions. The 1st and 3rd quartiles identify the centre as the entire interval between these two quartiles.

Scale Measures:

- standard deviation

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- median absolute deviation
- interquartile range (default)

The choice of scale measure determines the units with which to measure the distance from the centre of the distribution. The standard deviation is the most common scale measure, and is useful for determining outliers from normal distributions. The median absolute deviation and interquartile range are robust alternatives to the standard deviation, in that they are less sensitive to the presence of outliers. Given that the act of looking for outliers implies that outliers may be present, it is often recommended to use these robust scale measures. The interquartile range is the difference between the first and third quartiles, and is the most commonly recommended measure.

The default measures (*1st and 3rd quartiles* and *interquartile range*) correpond to a procedure for detecting the so-called box-plot outliers.

The outlier distance for each focal variable is defined as the absolute distance from the centre of the distribution (as quantified by the location measure), expressed in units of the scale measure. These distances can be optionally included in the output dataset of the Outlier Detector node.

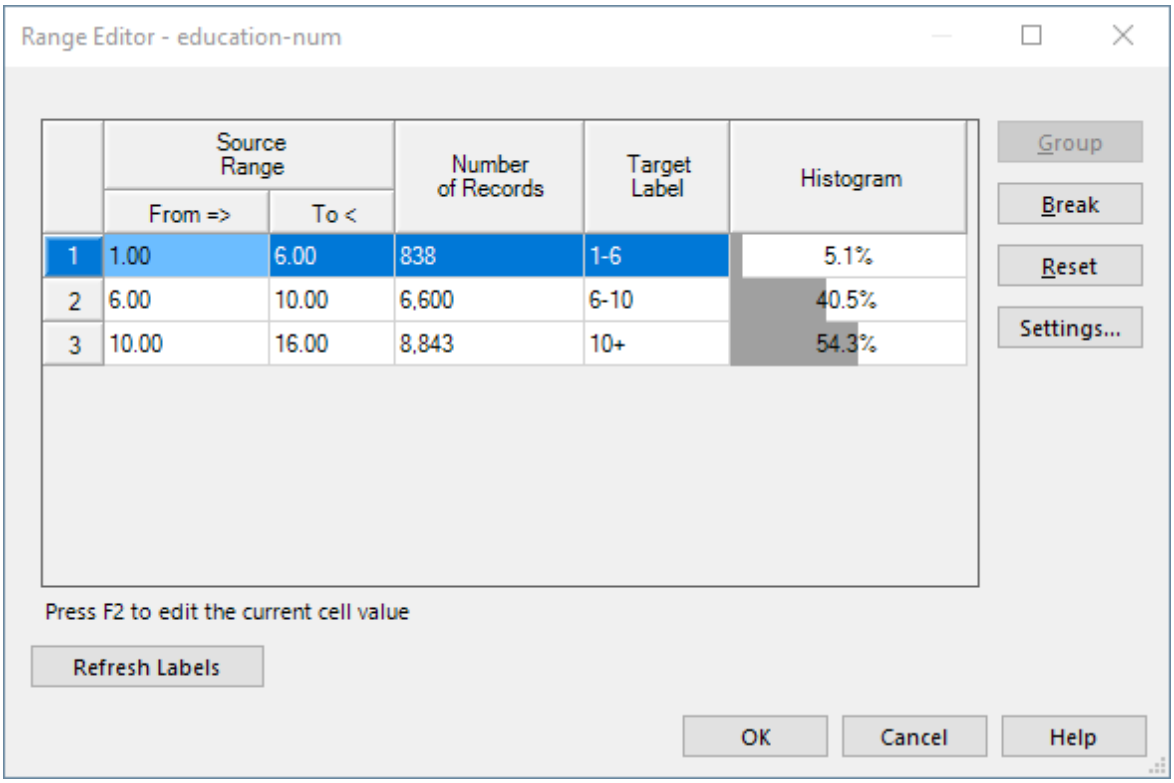
When selecting the location and scale measures, they can be mixed and matched as desired.

If any grouping variables are selected (see the details below), you can define their binnnig in the same way as in the Binning Helper and create group identification columns for the selected variables in the output dataset. For each record, these columns identifiy the group (interval or bin) that the record belongs to.

In the **Cut Off** column, specify the number of distance units in each of the selected variables for a record beyond this distance to be considered an outlier. The default is 3. Specify any non-negative number.

The **Define** button is available for variables whose role is '*grouping*' or '*focal & grouping*'. Clicking the **Define** button opens the *Range Editor* dialog. The default binning is the same as in the Binning Helper of the **Variable Transformations** node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

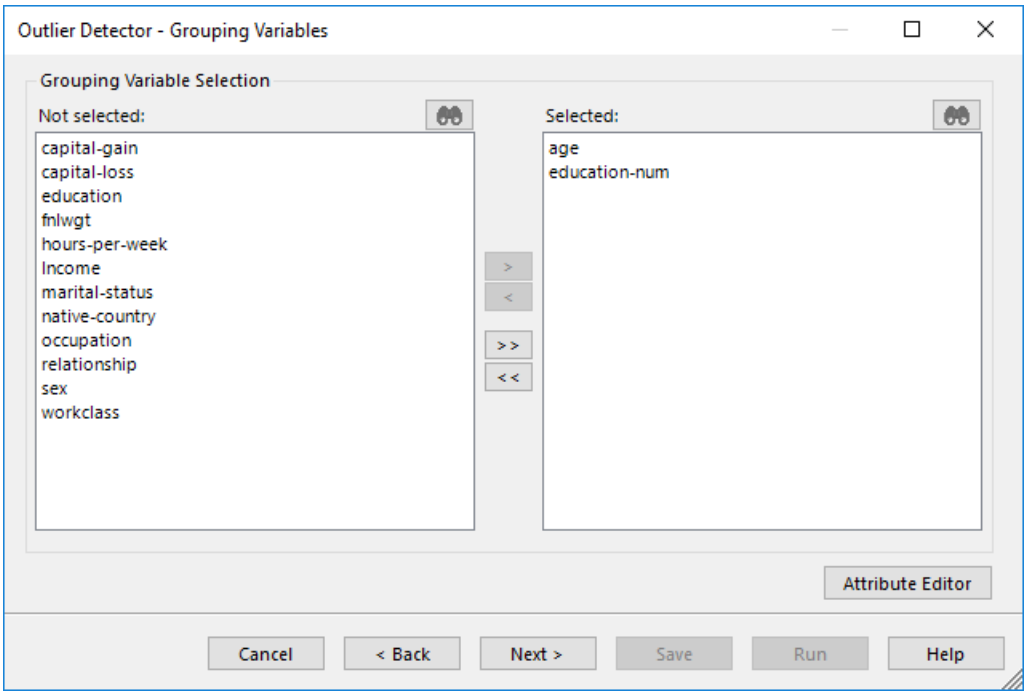


NOTE: If the data type of any grouping variable changes ater refreshing the input dataset, the following error may occur: *Failed to generate case statement for <variable_name>*. This means that the binning defined for this variable failed to be generated. To resolve this, open the Outlier Detector wizard, go to the second or third page, click the **Attribute Editor** button, click "Define" for the variable in question, edit the binning if necessary or just click OK, and then run the Outlier Detector node again.

Click **Next** to select the grouping variables if necessary.

The **Grouping Variables** page alows you to select one or more grouping variables and define their binnnig to create group identification columns for these variables in the output dataset. The columns identifiy the group that each record belongs to. This step is optional. None are selected by default. Attribute Editor can also be opened from this page to define the bins as described above.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

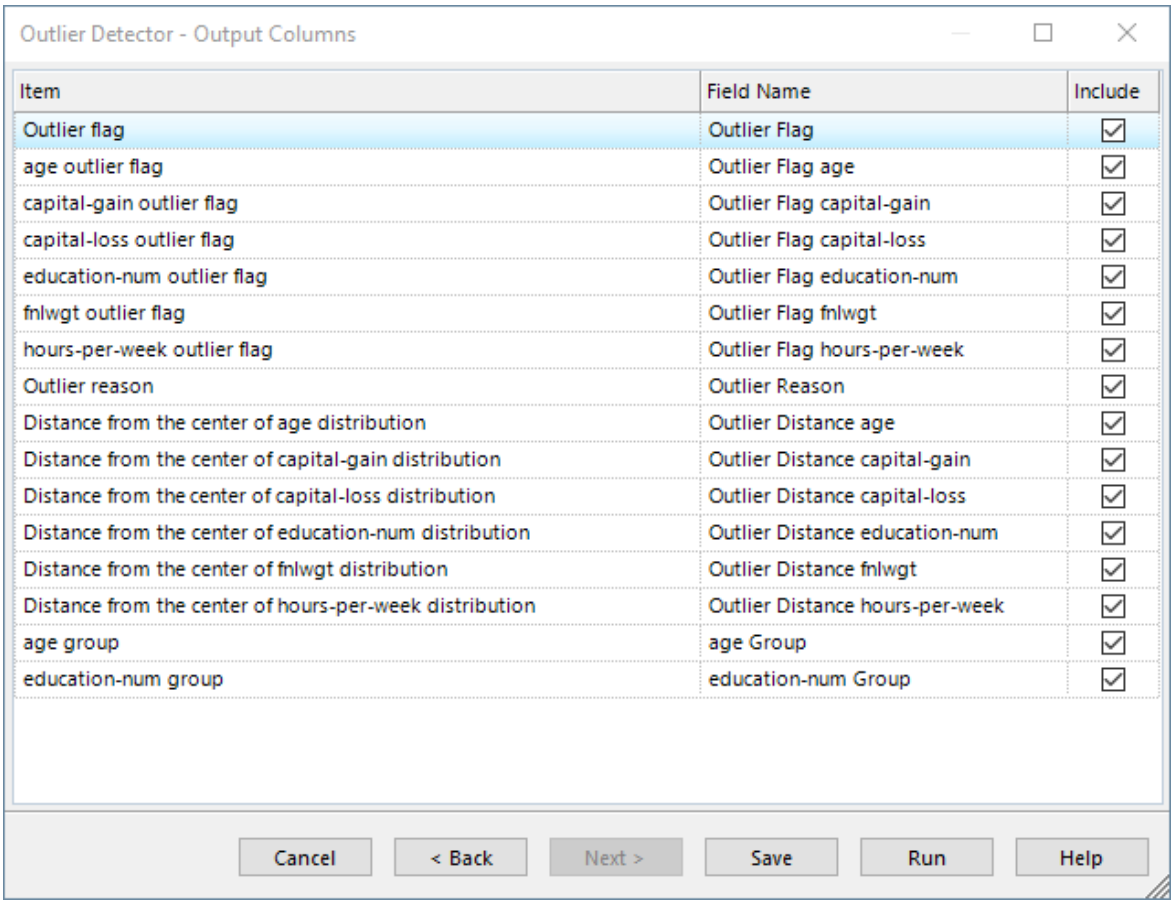


Click **Next** to select the output columns.

The items available here depend on the focal and grouping variables selected on the previous pages of the wizard. Modify the field names if desired.

The *Outlier reason* column lists all variables flagged as outliers for any given record.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

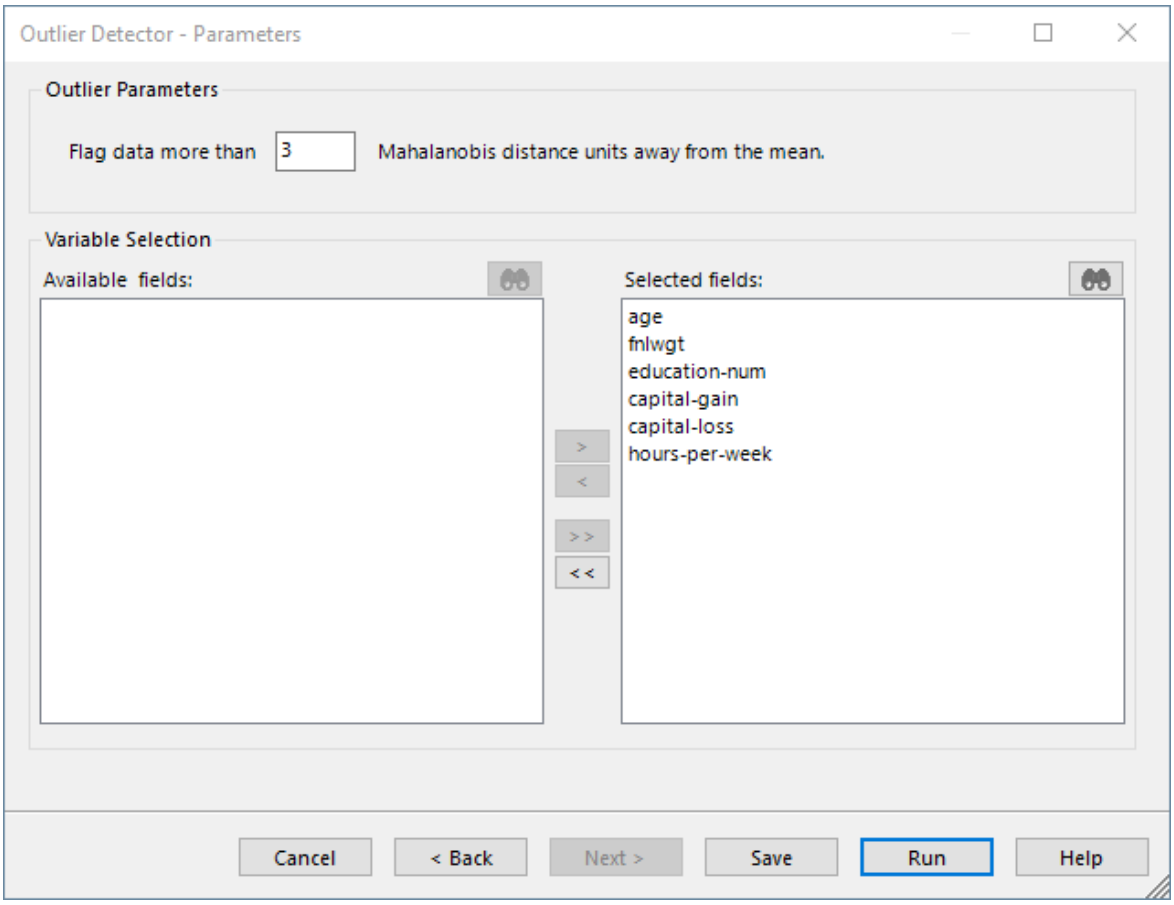


Mahalanobis Distance

If you selected the Mahalanobis Distance method, select the desired variables and specify the number of Mahalanobis Distance units away from the mean. The default is 3.

A record is an outlier if its robust Mahalanobis distance is greater than a threshold.

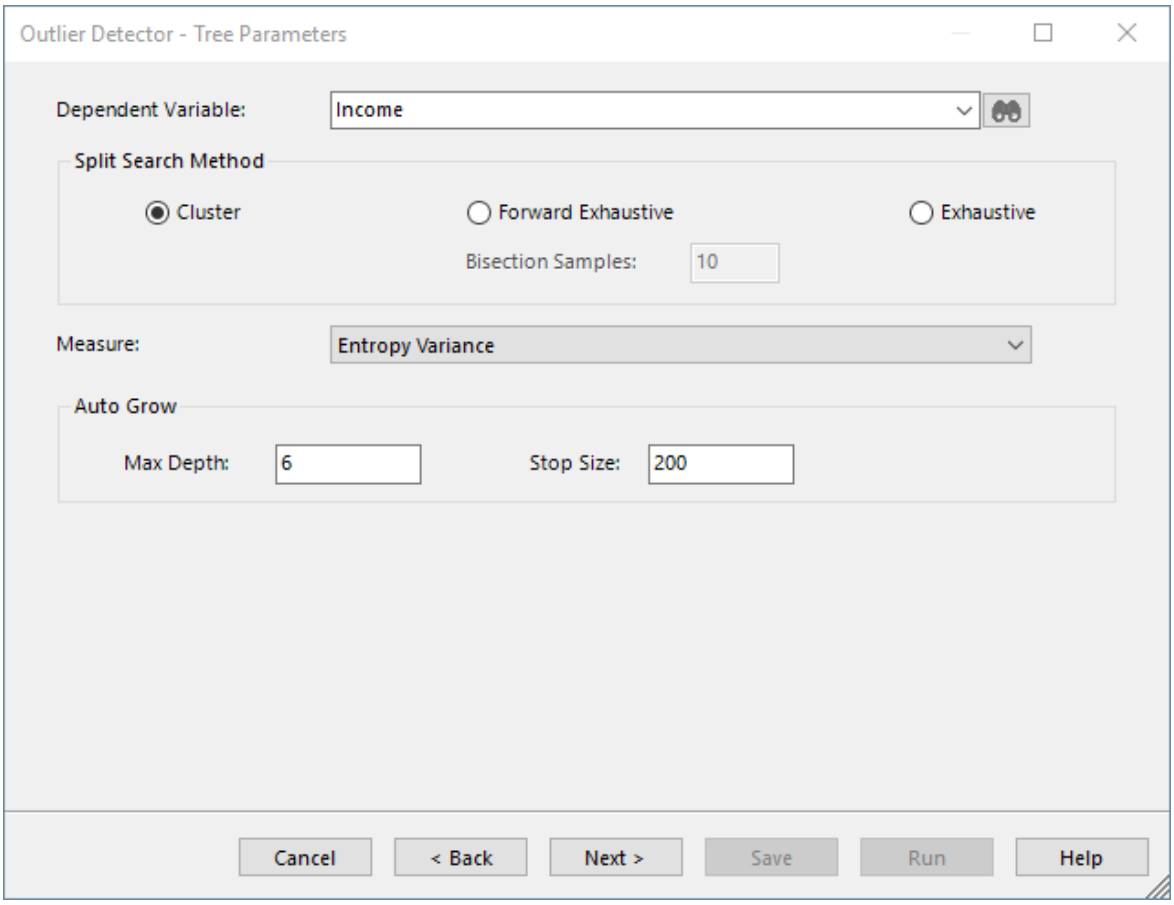
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Tree

If you selected the Tree method, specify the parameters for the decision tree. See the topics Decision Tree: Training Parameters and Auto Grow Parameters for the detailed description of these parameters.

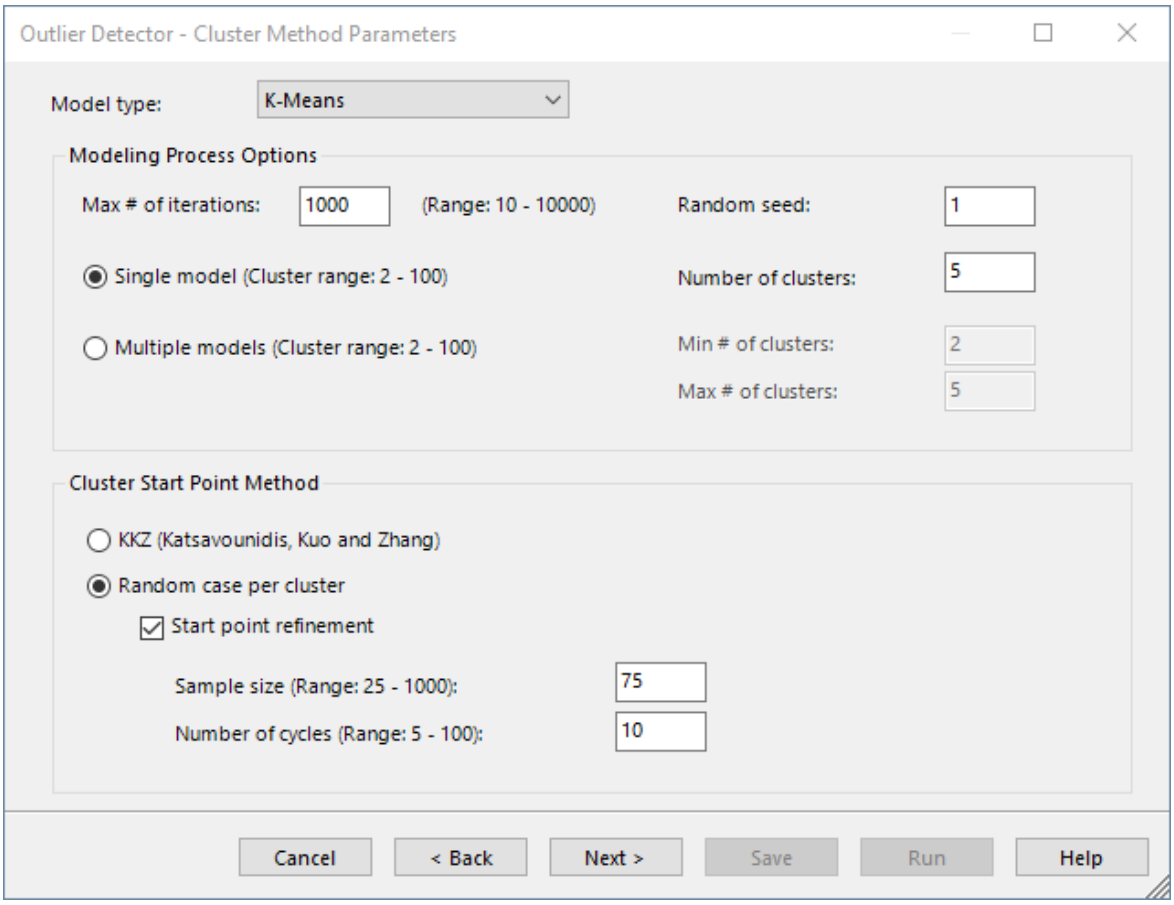
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Cluster

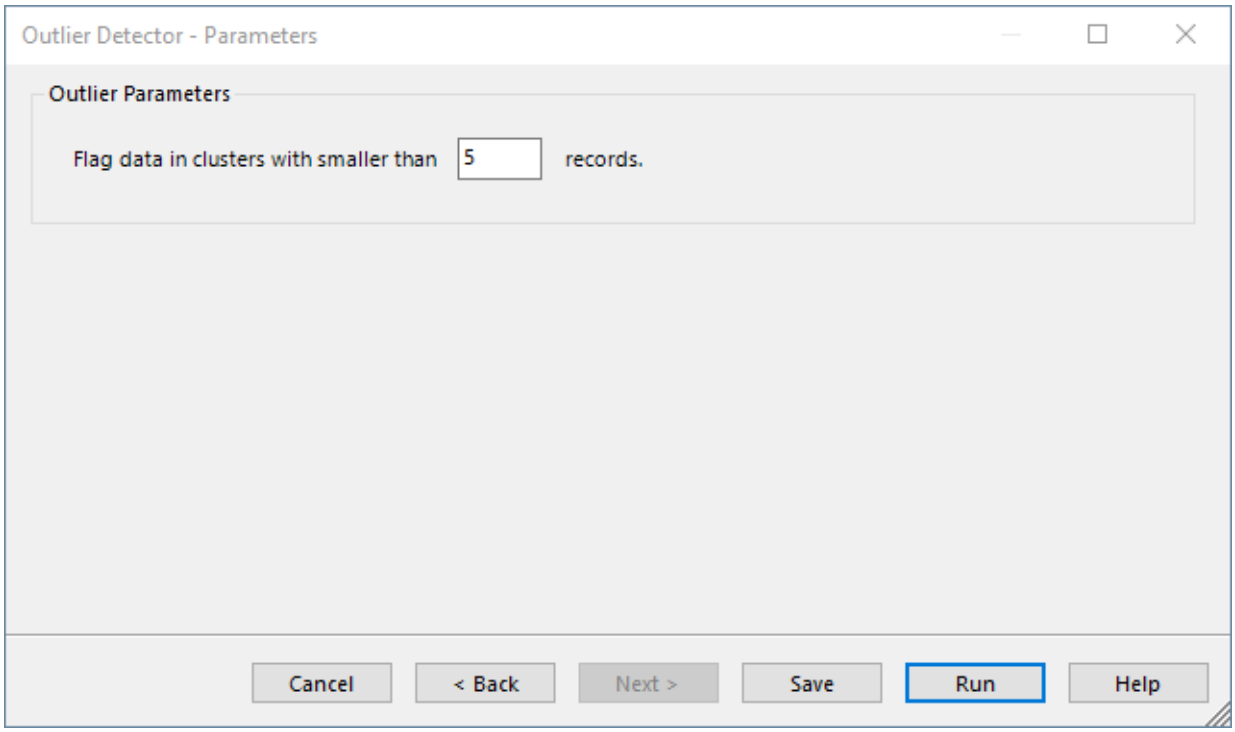
If you selected the Cluster method, specify the Cluster Analysis parameters. See the topic Inserting Cluster Analysis for details.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

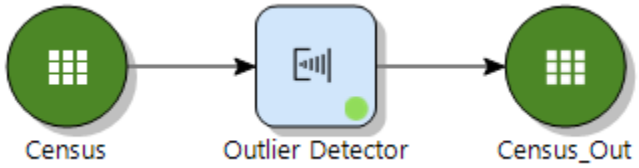


At the last page, if you selected the Tree or the Cluster method, specify the minimum number of records in a terminal node of the tree or a cluster to flag the data as an outlier. The default threshold is 5 records.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. The output dataset node will appear on the workflow canvas whne the process is finished..

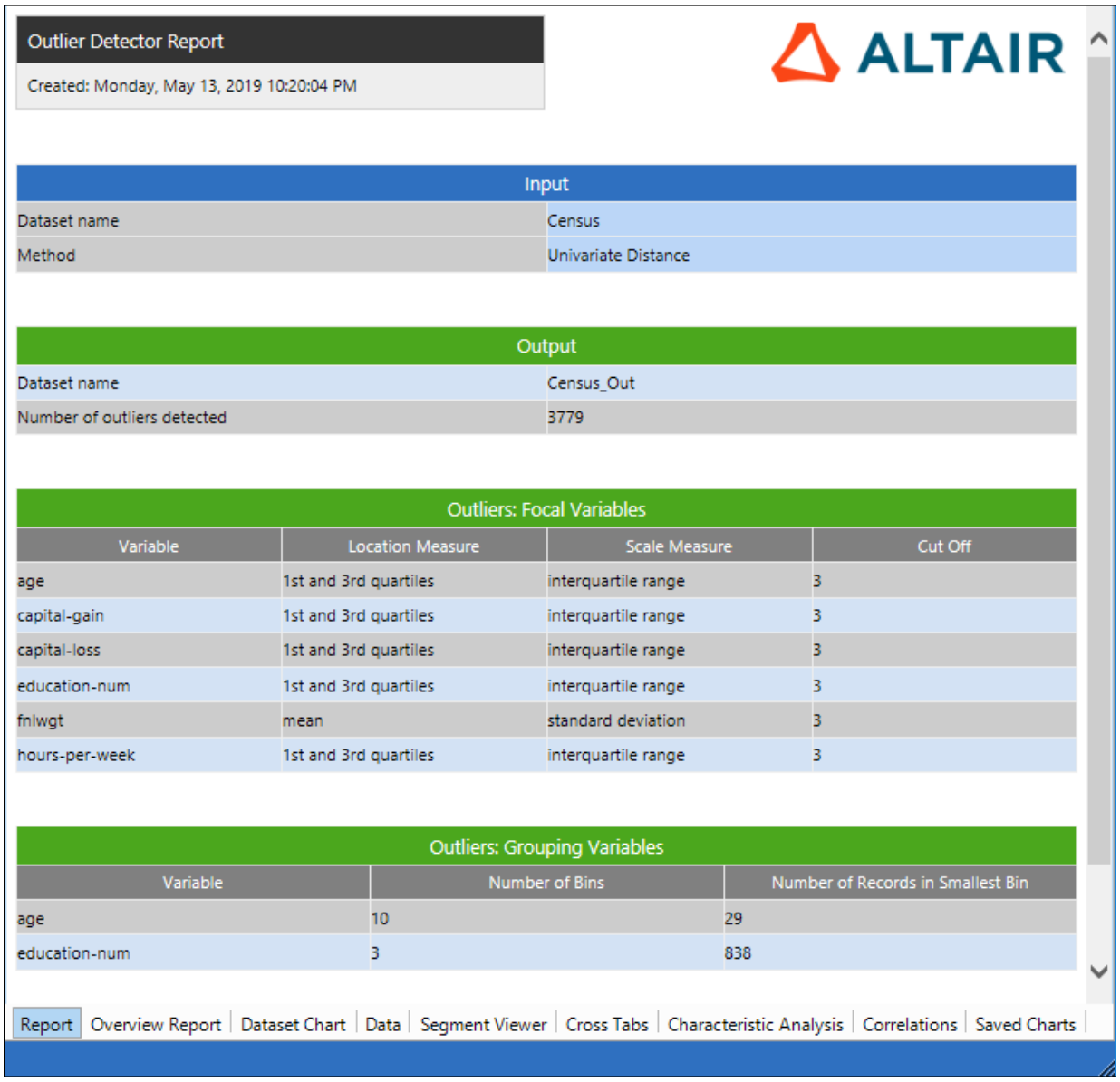


Outlier Detector Report and Dataset

Double-click the output dataset to open it. The default view is the **Report** tab, which shows the Outlier Detector Report. It provides the information on the input and output datasets, the parameters that were specified in the Outlier Detector wizard, and some general outlier statistics.

The number of outlier records detected is shown in the Output section. For each of the selected focal variables, the report shows the location measure, scale measure, and cut off. For each of the grouping variables, the report shows the number of bins and the number of records in the smallest bin.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Outlier Detector Report

Created: Monday, May 13, 2019 10:20:04 PM

Input	
Dataset name	Census
Method	Univariate Distance

Output	
Dataset name	Census_Out
Number of outliers detected	3779

Outliers: Focal Variables

Variable	Location Measure	Scale Measure	Cut Off
age	1st and 3rd quartiles	interquartile range	3
capital-gain	1st and 3rd quartiles	interquartile range	3
capital-loss	1st and 3rd quartiles	interquartile range	3
education-num	1st and 3rd quartiles	interquartile range	3
fnlwgt	mean	standard deviation	3
hours-per-week	1st and 3rd quartiles	interquartile range	3

Outliers: Grouping Variables

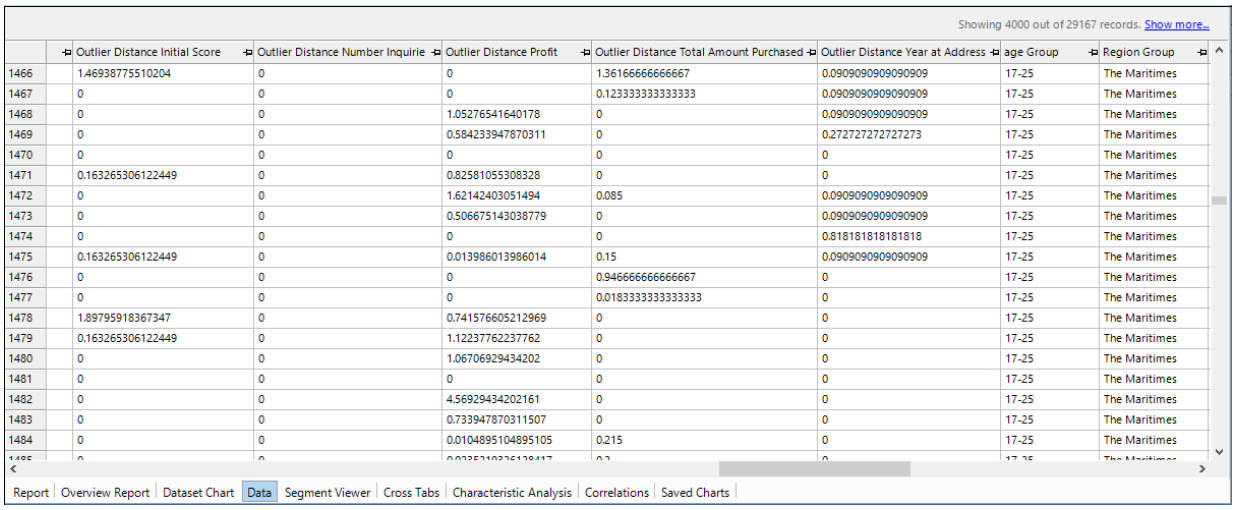
Variable	Number of Bins	Number of Records in Smallest Bin
age	10	29
education-num	3	838

Report | Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

Click the **Data** tab at the bottom of the dataset view to view the output dataset and check the outlier status of the records. The outlier indicator for an entire record is the *Outlier Flag* field, where the value is "Yes" if any of the focal variables in this record is flagged as an outlier according to the selected method.

For the selected grouping variables, the corresponding group identification columns display the bin label for each variable. In the example below, these are the *age Group* and the *Region Group* columns.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



	Outlier Distance Initial Score	Outlier Distance Number Inquirie	Outlier Distance Profit	Outlier Distance Total Amount Purchased	Outlier Distance Year at Address	age Group	Region Group
1466	1.46938775510204	0	0	1.36166666666667	0.0909090909090909	17-25	The Maritimes
1467	0	0	0	0.123333333333333	0.0909090909090909	17-25	The Maritimes
1468	0	0	1.05276541640178	0	0.0909090909090909	17-25	The Maritimes
1469	0	0	0.584233947870311	0	0.272727272727273	17-25	The Maritimes
1470	0	0	0	0	0	17-25	The Maritimes
1471	0.163265306122449	0	0.82581055308328	0	0	17-25	The Maritimes
1472	0	0	1.62142403051494	0.085	0.0909090909090909	17-25	The Maritimes
1473	0	0	0.506675143038779	0	0.0909090909090909	17-25	The Maritimes
1474	0	0	0	0	0.818181818181818	17-25	The Maritimes
1475	0.163265306122449	0	0.013986013986014	0.15	0.0909090909090909	17-25	The Maritimes
1476	0	0	0	0.946666666666667	0	17-25	The Maritimes
1477	0	0	0	0.0183333333333333	0	17-25	The Maritimes
1478	1.89795918367347	0	0.741576605212969	0	0	17-25	The Maritimes
1479	0.163265306122449	0	1.12237762237762	0	0	17-25	The Maritimes
1480	0	0	1.06706929434202	0	0	17-25	The Maritimes
1481	0	0	0	0	0	17-25	The Maritimes
1482	0	0	4.56929434202161	0	0	17-25	The Maritimes
1483	0	0	0.733947870311507	0	0	17-25	The Maritimes
1484	0	0	0.010489510489510 5	0.215	0	17-25	The Maritimes

If necessary, isolate the records flagged as outliers using any of the following methods:

- Connect the output dataset to a new Partition node. In the Partition wizard, use the **Stratified** option and select *Outlier* as the stratification field, then request all records with Outlier = "Yes".

  OR

- Insert a Strategy Tree for the output dataset. Force a split on the *Outlier* field, then click on the child node where Outlier = "Yes" and convert it into a new dataset by selecting the **Node Dataset** command from the Insert menu.

Measure of Predictive Power

The **Measure of Predictive Power** node helps you select the most predictive variables for your model.

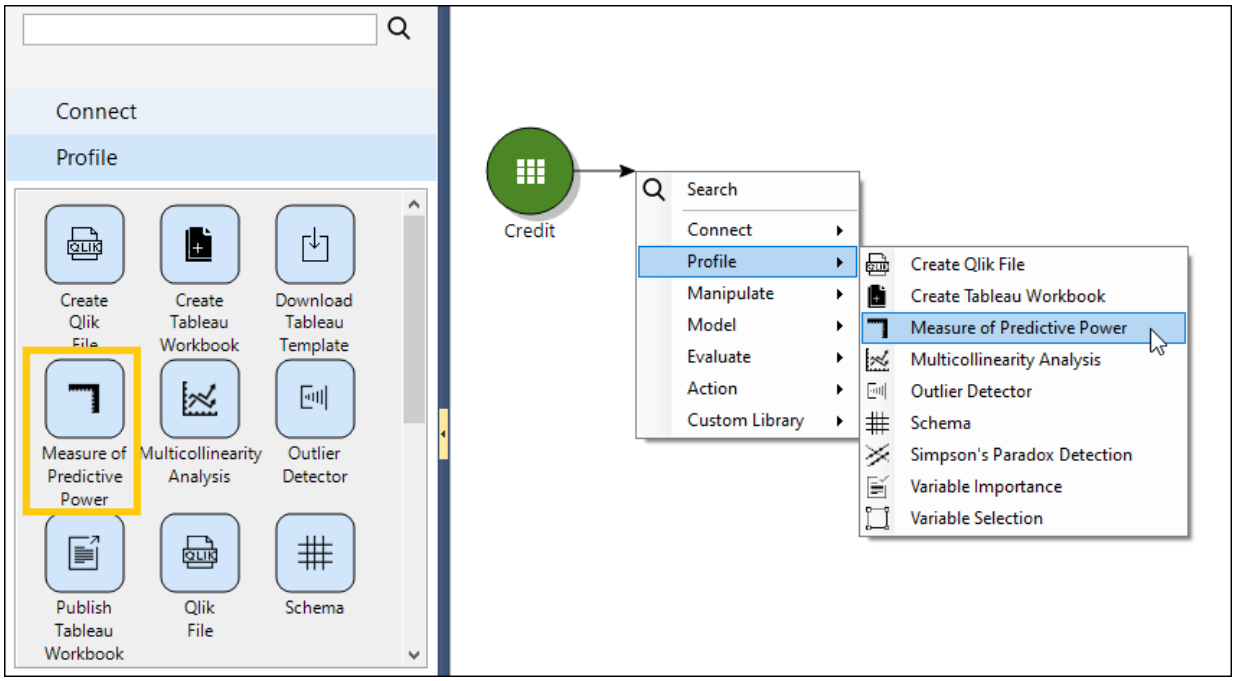
To add a Measure of Predictive Power node to your workflow:

- Drag a connection arrow from a dataset node to any area on the Workflow canvas background. This brings up a context menu that allows you to select the desired operation. From the context menu, select **Profile --> Measure of Predictive Power** as shown in the example below. A new **Measure of Predictive Power** process node will be created.

  OR:

- Drag the **Measure of Predictive Power** node from the **Profile** palette onto the canvas and connect the required input dataset to the node.

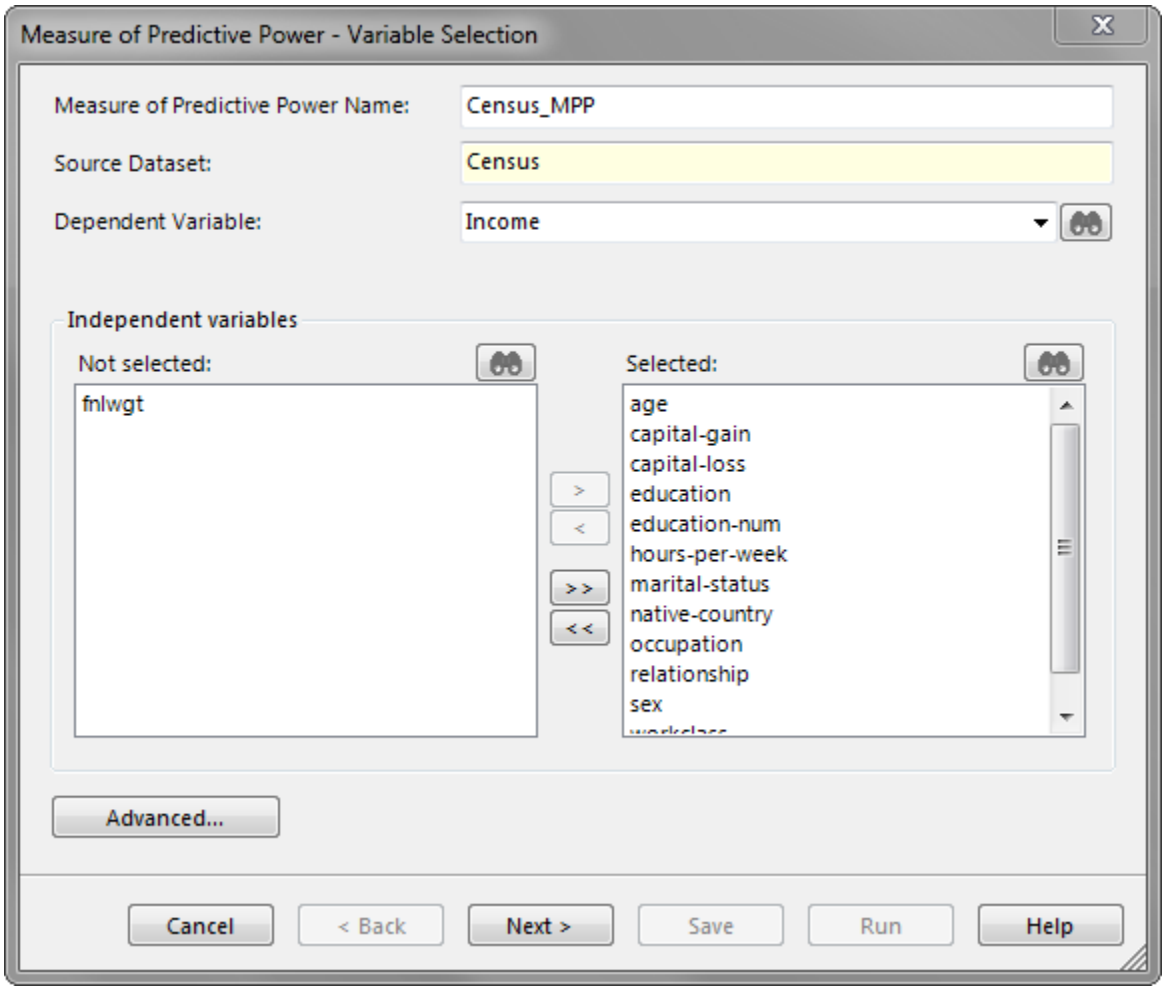
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If the wizard does not open automatically, double-click the new node or right-click on it and select **Modify** to open the wizard.

In the wizard, specify the name for the new report as it will appear in the Project Explorer and the workflow canvas. Specify the dependent variable you are interested in and select the independent variables.

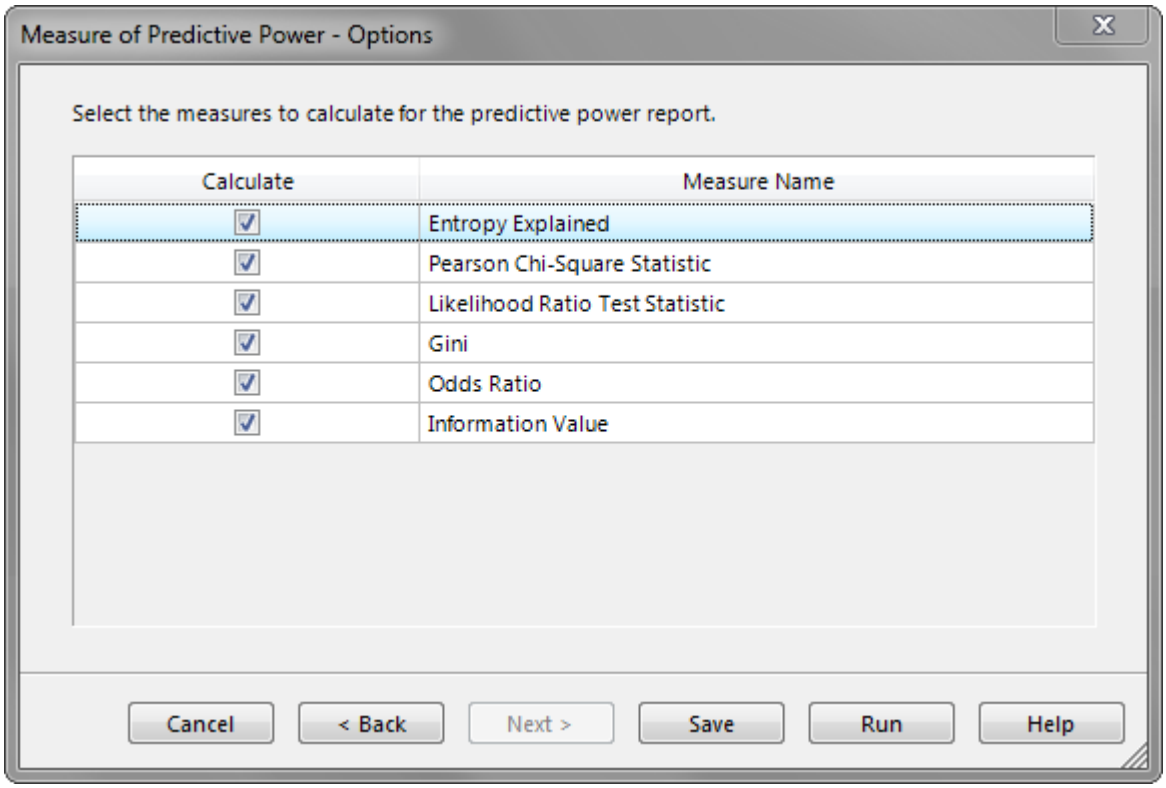
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next** to select the desired statistics (measures of predictive power). The list of available statistics at the next step depends on the type of the selected dependent variable.

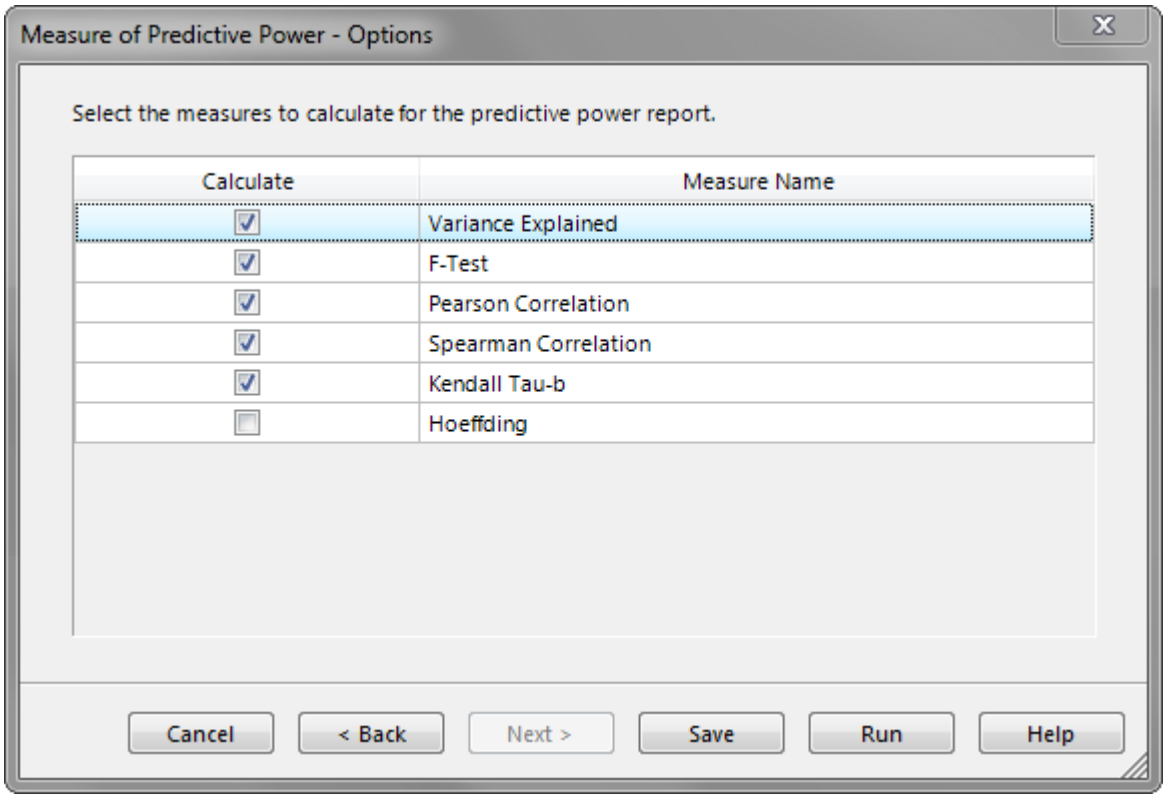
If the dependent variable is discrete, then the available measures are *Entropy Explained*, *Pearson Chi-Square Statistic*, *Likelihood Ratio Test Statistic*, *Gini* (*Gini raito*), *Odds Ratio*, and *Information Value*. Note that Odds Ratio and Information Value are only applicable in the case of binary dependent variables. See the topic Mathematical Formulations: Decision Tree for details on the Information Value and other statistics.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

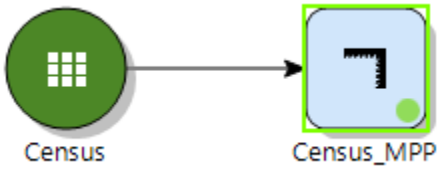


If the dependent variable is continuous, then the available measures are *Variance Explained*, *F-Test*, *Pearson Correlation*, *Spearman Correlation* (*Spearman's Rho*), *Kendall's Tau-b*, and *Hoeffding's D.* Note that Hoeffding's D is not selected by default because of its calculation complexity, as it may take a long time to calculate in case of a large dataset.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

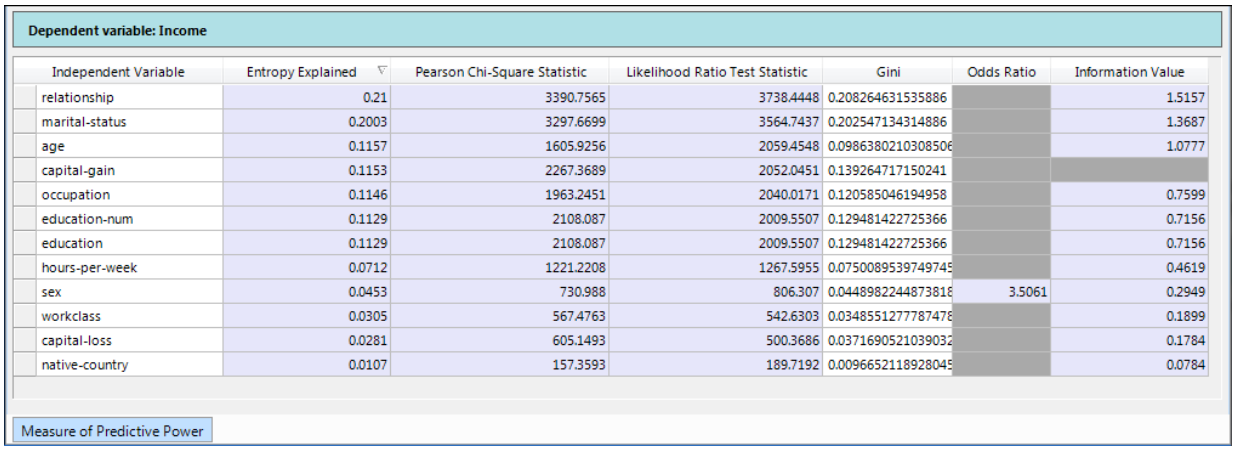


Click **Save** to save the settings, or click **Run** to save the settings and run the append process immediately.



Double-click the Measure of Predictive Power node on the canvas. The resulting report is very much similar to the Split Report in Decision Trees. In fact, some of its statistics such as Entropy Variance and Information value come from the Decision Tree split. Examine the report and choose the variables you would like to use in your modeling process based on the parameters most important to you.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Dependent variable: Income

Independent Variable	Entropy Explained	Pearson Chi-Square Statistic	Likelihood Ratio Test Statistic	Gini	Odds Ratio	Information Value
relationship	0.21	3390.7565	3738.4448	0.208264631535886		1.5157
marital-status	0.2003	3297.6699	3564.7437	0.202547134314886		1.3687
age	0.1157	1605.9256	2059.4548	0.0986380210308506		1.0777
capital-gain	0.1153	2267.3689	2052.0451	0.139264717150241		
occupation	0.1146	1963.2451	2040.0171	0.120585046194958		0.7599
education-num	0.1129	2108.087	2009.5507	0.129481422725366		0.7156
education	0.1129	2108.087	2009.5507	0.129481422725366		0.7156
hours-per-week	0.0712	1221.2208	1267.5955	0.0750089539749745		0.4619
sex	0.0453	730.988	806.307	0.0448982244873818	3.5061	0.2949
workclass	0.0305	567.4763	542.6303	0.0348551277787478		0.1899
capital-loss	0.0281	605.1493	500.3686	0.0371690521039032		0.1784
native-country	0.0107	157.3593	189.7192	0.0096652118928045		0.0784

Measure of Predictive Power

From this view you can insert a variable list and save the list variables you want to use for your modeling in the future. Select Insert | Variable List and follow the instructions in the Variable List topic.

Here is the list of statistics displayed depending on the dependent variable type:

- For all categorical variables: *Entropy Explained* (Entropy ratio), *Pearson Chi-square statistic*, *Likelihood ratio test statistic*, and *Gini* (Gini ratio).
- For binary variables: the same ones as for all categorical ones, plus: *Odds ratio* and *Information value*
- For ordinal variables (in addition to the above): *Spearman's Rho* and *Kendall Tau-b* correlation coefficients
- For continuous variables: *Variance Explained* (Variance ratio), four correlation coefficient types (*Pearson, Spearman, Kendall's Tau-b*, and *Hoeffding's D*), *Gini* (Gini ratio), and *F-test*

Variable Importance

The **Variable Importance** node in the **Profile** palette is used to estimate the importance of variables in terms of their contribution to a predictive model. The information from the model is used to calculate the variable importance metric. The output includes a ranked list of variables in the order of their importance and a histogram illustrating this in a graphical form.

The Variable Importance metric is computed based on accuracy drop for classification models and Mean Square Error (MSE) for regression models. The node is applicable to the following model types: Decision Tree, Random Forest, Bagging, Boosting, Deep Learning, Linear Regression, Logistic Regression, Constrained Regression, Regularization, Model Stacking, and Reject Inference.

The contribution of each variable is estimated by fitting a one-variable model and calculating the importance metric as follows: First, the model is scored on the dataset to calculate the metric *M* (accuracy or MSE value). Then, for each variable *X*, the values of the variable are permuted, the model is scored on the dataset with the values permuted, and the new value of *M* (accuracy or MSE) is calculated. The difference: (*M* - *Mx*) is taken as the importance of variable *X*.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Note: Indicator variables are kept together, i.e., the categories of a discrete independent variable are not treated separately.

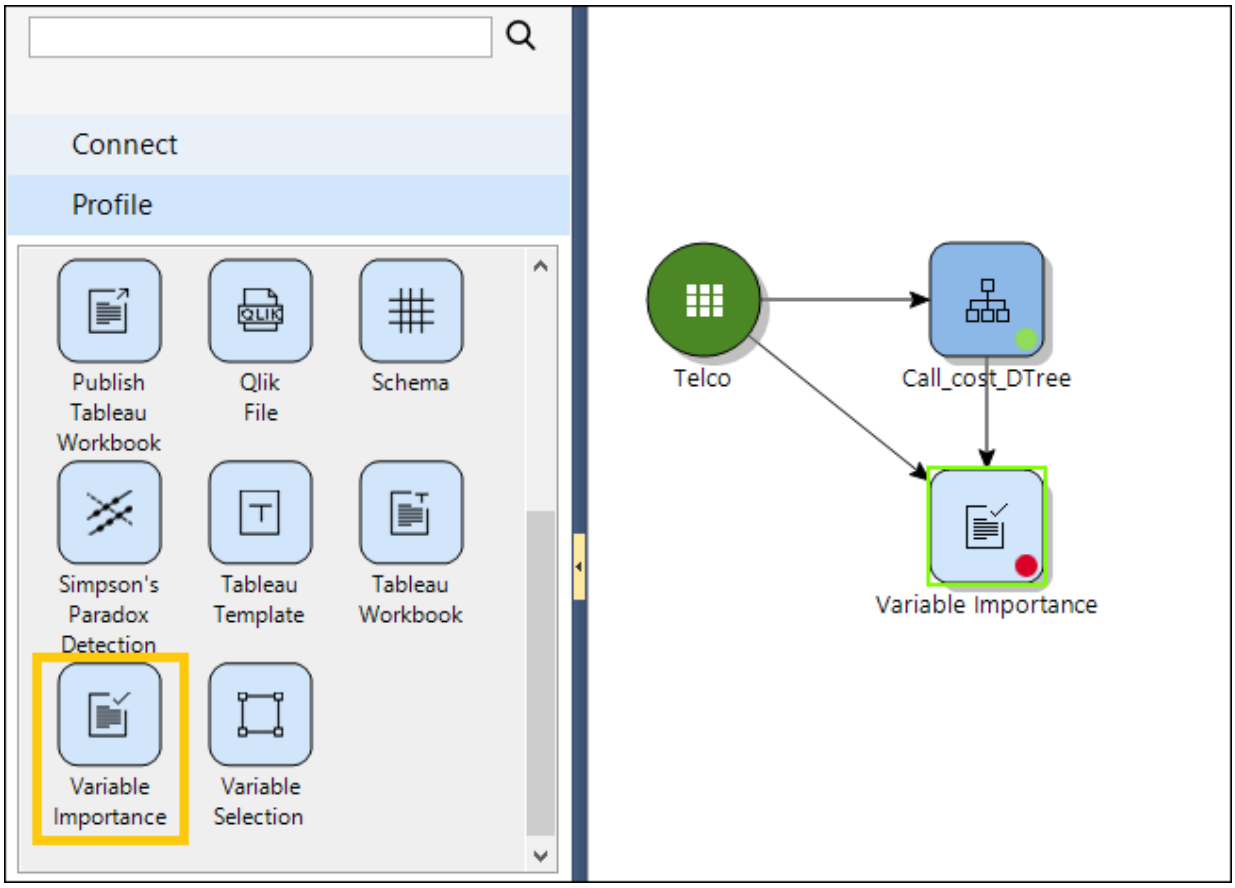
Prerequisites: A dataset node and a model that uses variables from this dataset.

To add a Variable Importance node to your workflow:

- Right-click on the Workflow canvas next to the dataset and model nodes. From the context menu, select **Profile --> Variable Importance**. A new **Variable Importance** process node will be created. Connect the dataset and the model to the new node as shown below.

OR:

- Drag the **Variable Importance** node from the **Profile** palette onto the canvas and connect the dataset and the model to the new node as shown below.

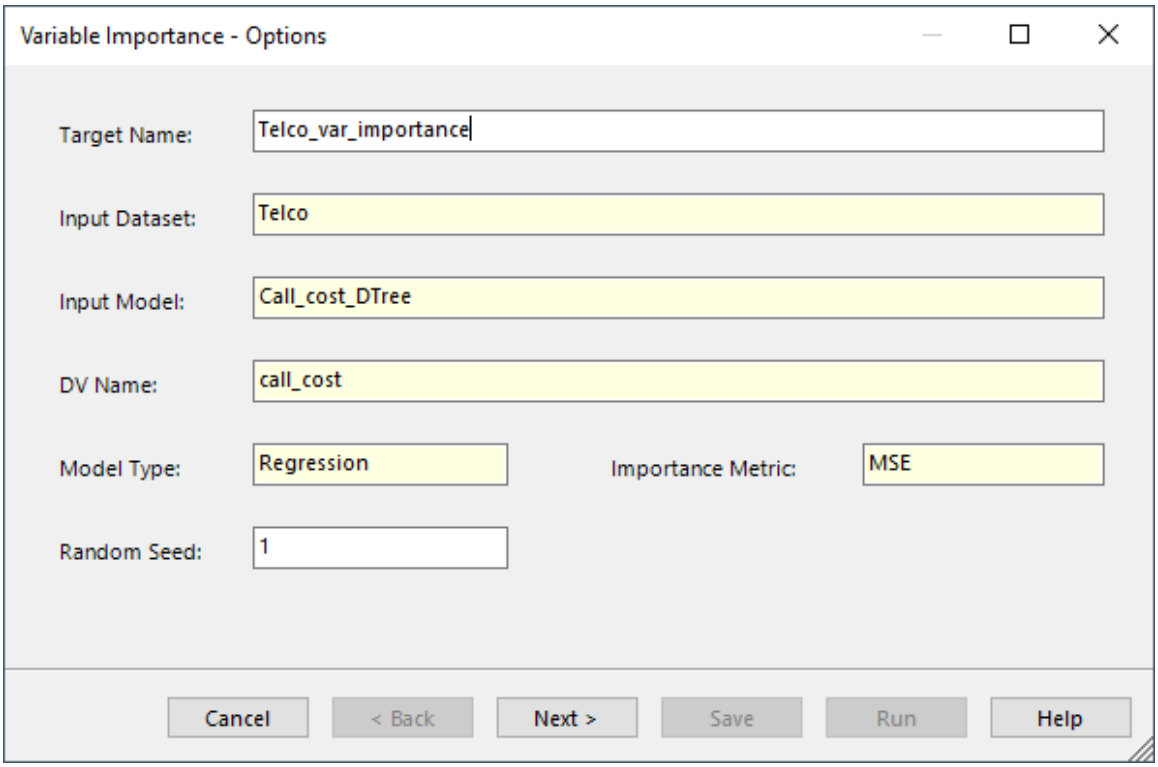


If the Variable Importance node is already on the Workflow canvas, double-click it or right-click on it and select **Modify**.

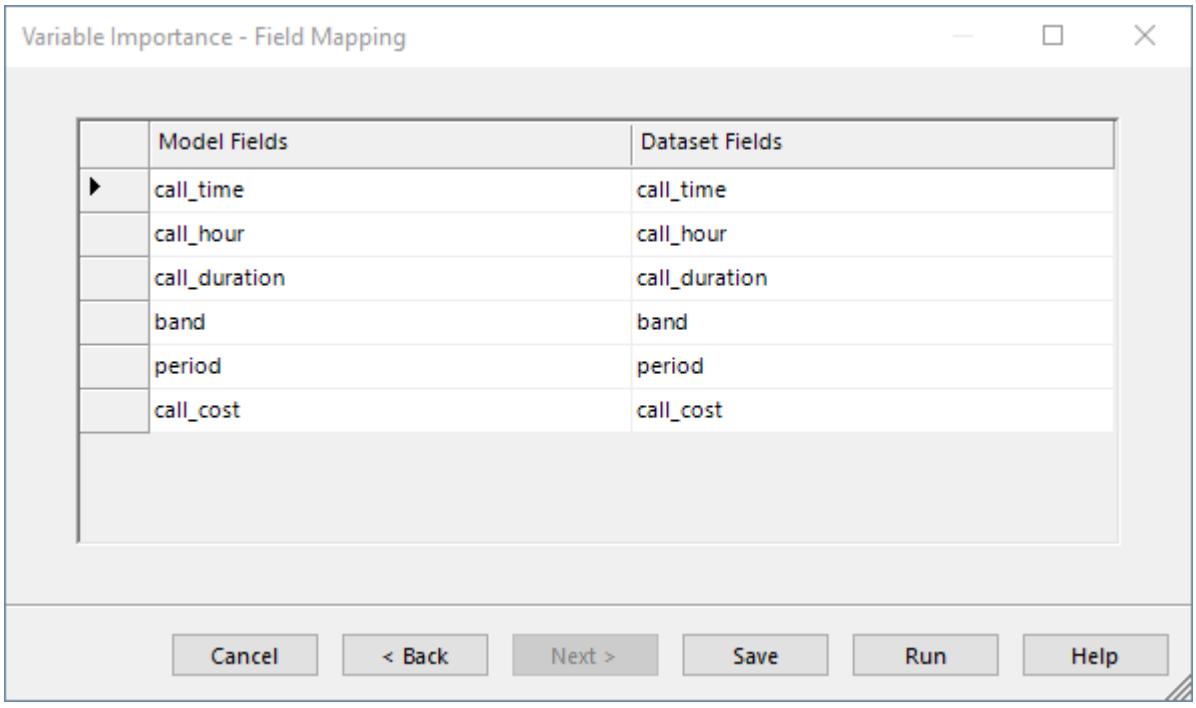
In the Variable Importance wizard:

- In the Target Name field, specify the name for the Variable Importance Report.
- Specify the Random Seed value for the repeatability of results, if necessary.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

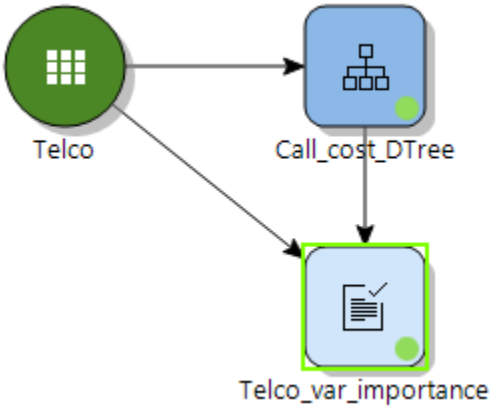
1. 

At the next step, select the input variables to be ranked:



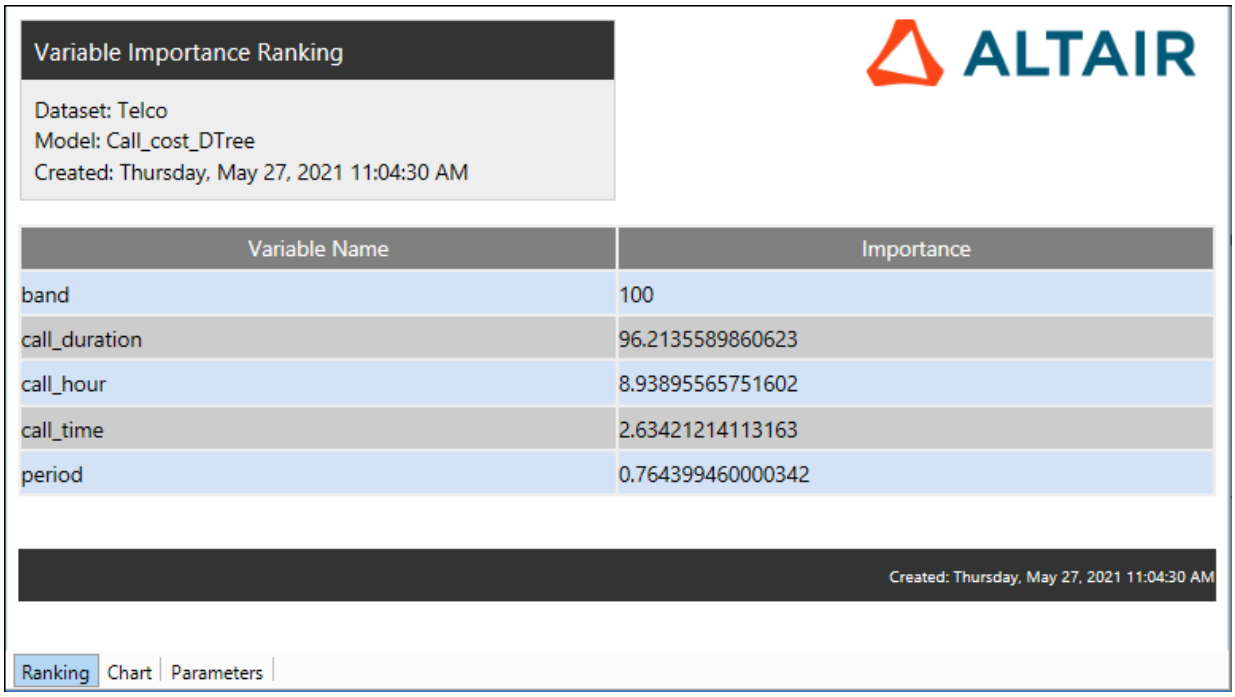
Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the node, double-click the Variable Importance node to view the results.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The Variable Importance report consists of three tabs: Ranking, Chart, and Parameters.

The **Ranking** tab displays the importance for each variable used by the model, in the descending order. You can sort by Variable Name and Importance by clicking on the column headers.



Variable Importance Ranking

Dataset: Telco
Model: Call_cost_DTree
Created: Thursday, May 27, 2021 11:04:30 AM

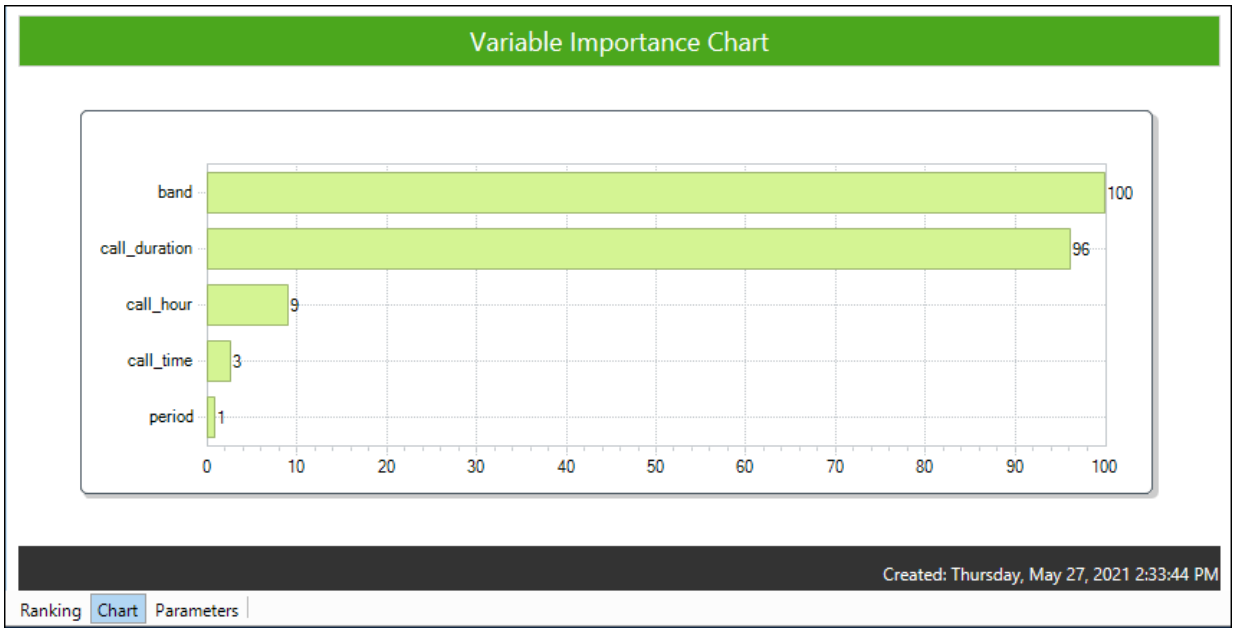
Variable Name	Importance
band	100
call_duration	96.2135589860623
call_hour	8.93895565751602
call_time	2.63421214113163
period	0.764399460000342

Created: Thursday, May 27, 2021 11:04:30 AM

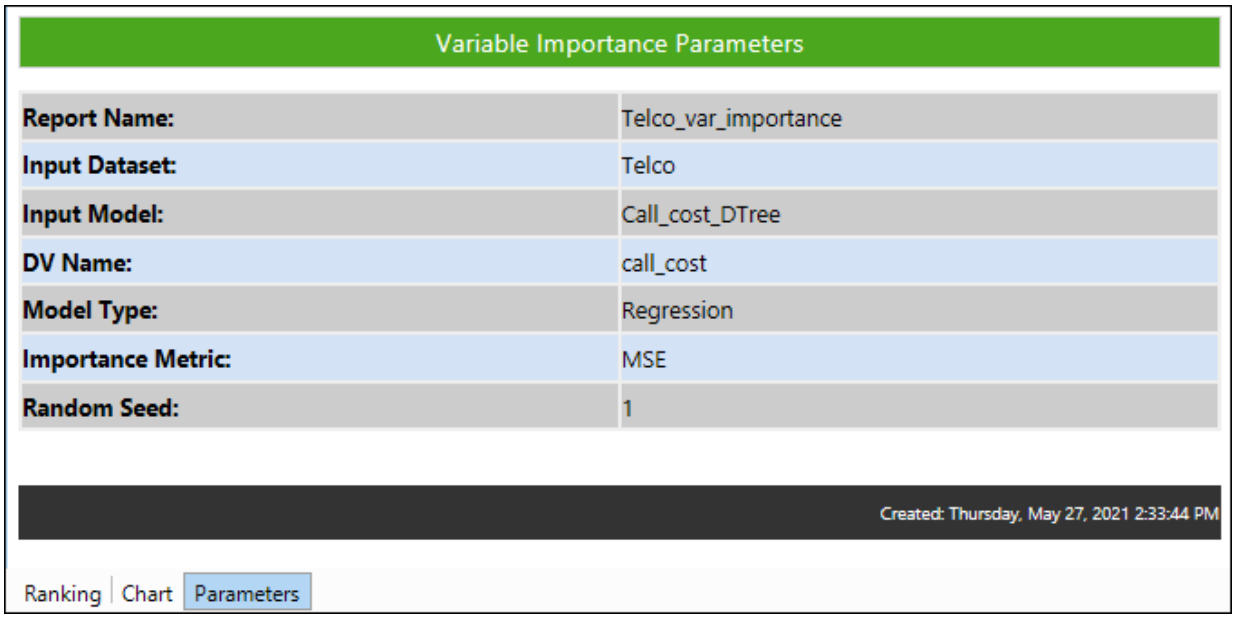
Ranking | Chart | Parameters

The **Chart** tab displays the same information in the form of a histogram:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The **Parameters** tab summarizes the input parameters that were specified in the Variable Importance node wizard:



See also: Measure of Predictive Power

Multicollinearity Analysis

The **Multicollinearity Analysis** node in the Profile palette detects multicollinearity among numeric variables and automatically removes highly correlated variables based on a user-defined cut-off value of the correlation coefficient.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Unlike the *Correlations* tab in datasets, which calculates the correlation coefficients for all variables and shows the correlation matrix, the **Multicollinearity Analysis** node also helps you decide which ones of the correlated variables should be removed. It is especially useful when the number of predictors is large, and it is hard to make decisions just by viewing the correlation matrix.

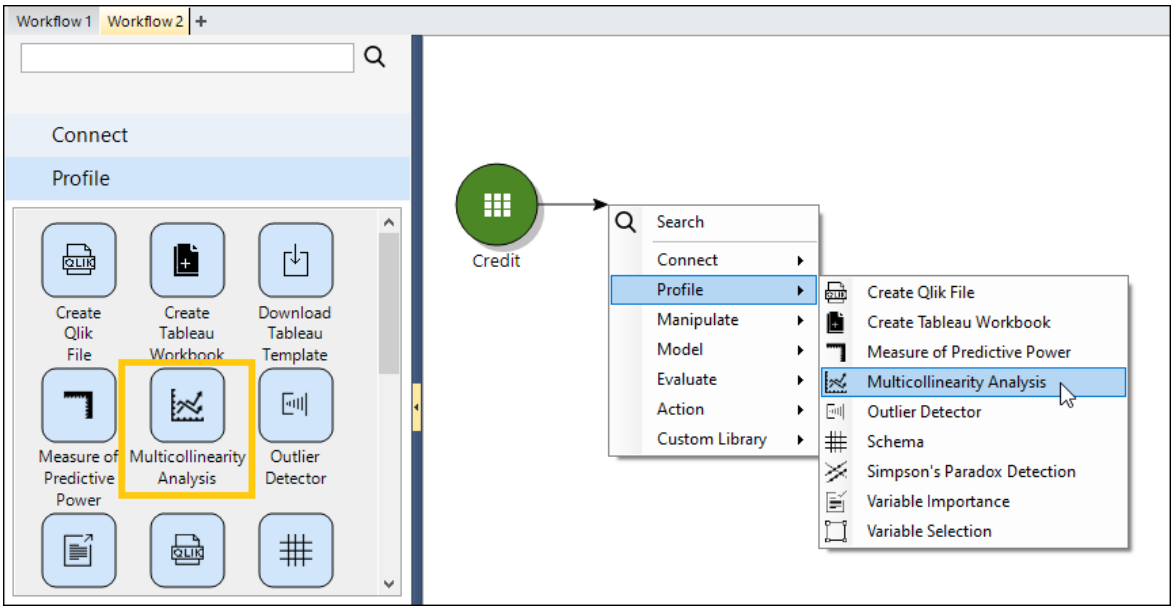
Only numeric variables are analyzed by this node.

To add a Multicollinearity Analysis node to your workflow:

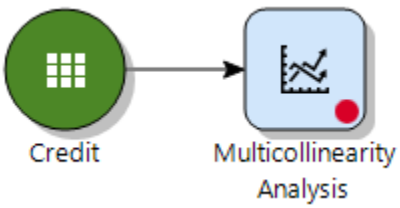
- Drag a connection arrow from a dataset node to any area on the Workflow canvas background. From the context menu, select **Profile --> Multicollinearity Analysis** as shown in the example below. A new Multicollinearity Analysis node will be created, and the wizard will open.

OR:

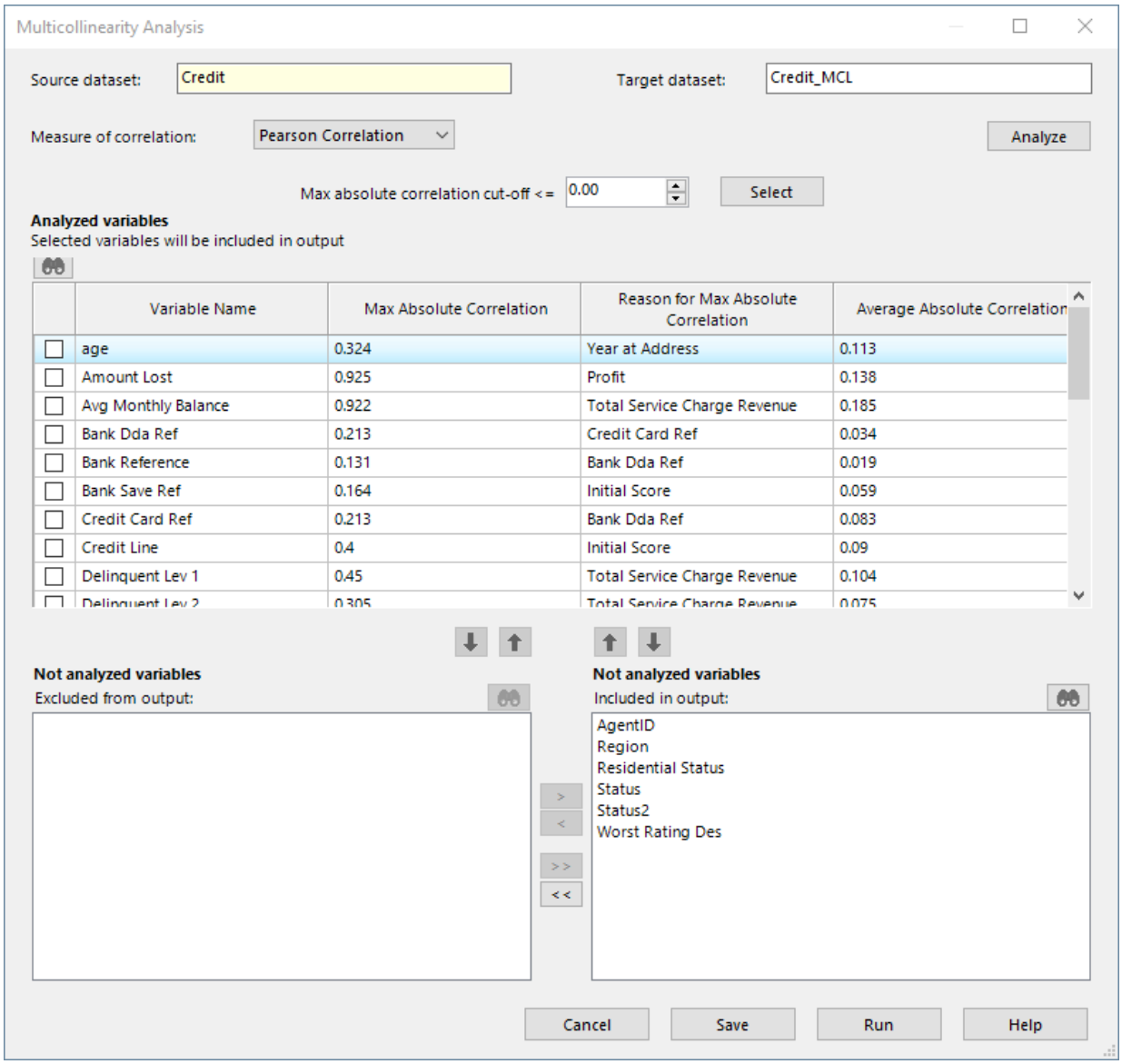
- Drag the **Multicollinearity Analysis** node from the **Profile** palette onto the canvas and connect the input dataset to the node.



If the Multicollinearity Analysis node is already on the canvas, double-click it or right-click on it and select **Modify** from the context menu to open the wizard.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.



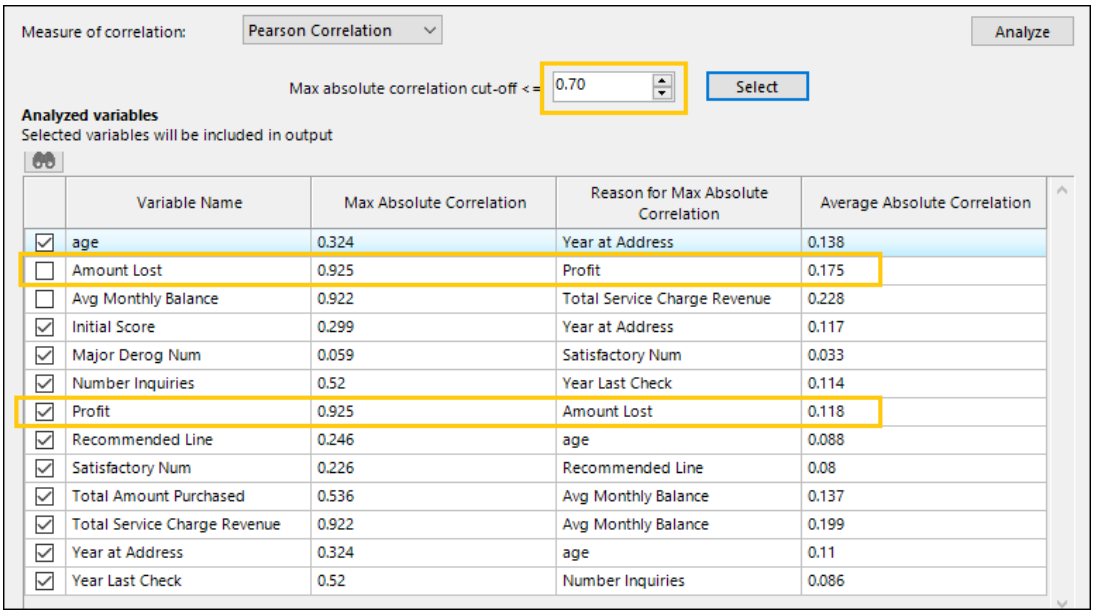
In the Multicollinearity Analysis wizard:

- Specify the name for the output dataset in the *Target dataset* field.
- Select the desired correlation type from the *Measure of correlation* drop-down list. The available types are: *Pearson Correlation* and *Spearman Correlation*.
- Click **Analyze** to calculate the correlations. Only numeric variables are analyzed.
    - For each variable, maximum and average absolute correlations are shown in the **Analyzed variables** frame.
        - *Maximum Absolute Correlation* is the maximum absolute value of the correlation coefficients of the given variable with all other variables).
        - *Average Absolute Correlation* is the average absolute value of the correlation coefficients of the given variable with all other variables).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Once you click **Analyze**, for each variable on the left, the highest-correlated variable is shown the column "*Reason for Max Absolute Correlation*".
- Click in the boxes in the leftmost column for any variables that you want to include in the output.
- To automatically select the variables whose Maximum Absolute Correlation does not exceed a certain threshold, enter the desired threshold value in the box "***Max absolute correlation cut-off***" and click **Select**.
    - Note: Some variables with Max absolute correlation exceeding the cut-off will also get selected according to the following rule: Suppose Variable *V1* has Max Absolute Correlation greater than the cut-off, and the corresponding highly correlated variable is *V2*. Then the value of Average Absolute Correlation will be compared for *V1* and *V2,* and the one with lower Average Absolute Correlation will get selected, while the other will get excluded.

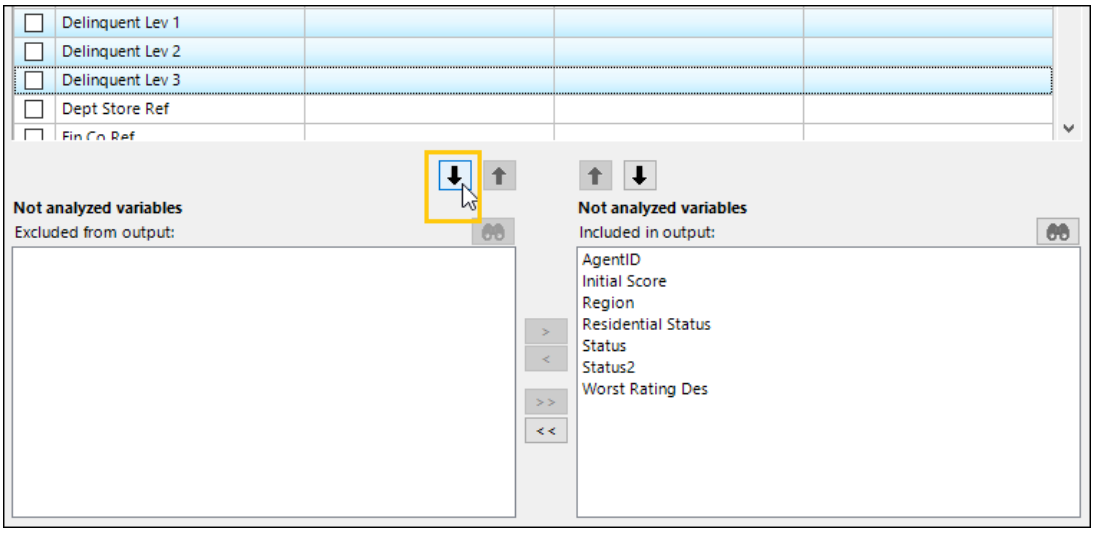
  Example: Assume the user-defined cut-off value is 0.7. After clicking **Select**, in the table below, the variables "*age*" and "*Year at Address*" are selected since their *Max Absolute Correlation* = 0.324, which is below the cut-off value of 0.7. The pair "*Amount Lost*" and "*Profit*" has Max Absolute Correlation = 0.925, which is above the cut-off value. Therefore, only one of these two variables are auto-selected as a result of comparing their *Average Absolute Correlation* values*:* For "*Profit*" it is 0.118, which is lower than for "*Amount Lost*" (0.175), and therefore "*Profit"* is selected, while "*Amount Lost"* is not.



	Variable Name	Max Absolute Correlation	Reason for Max Absolute Correlation	Average Absolute Correlation
☑	age	0.324	Year at Address	0.138
☐	Amount Lost	0.925	Profit	0.175
☐	Avg Monthly Balance	0.922	Total Service Charge Revenue	0.228
☑	Initial Score	0.299	Year at Address	0.117
☑	Major Derog Num	0.059	Satisfactory Num	0.033
☑	Number Inquiries	0.52	Year Last Check	0.114
☑	Profit	0.925	Amount Lost	0.118
☑	Recommended Line	0.246	age	0.088
☑	Satisfactory Num	0.226	Recommended Line	0.08
☑	Total Amount Purchased	0.536	Avg Monthly Balance	0.137
☑	Total Service Charge Revenue	0.922	Avg Monthly Balance	0.199
☑	Year at Address	0.324	age	0.11
☑	Year Last Check	0.52	Number Inquiries	0.086

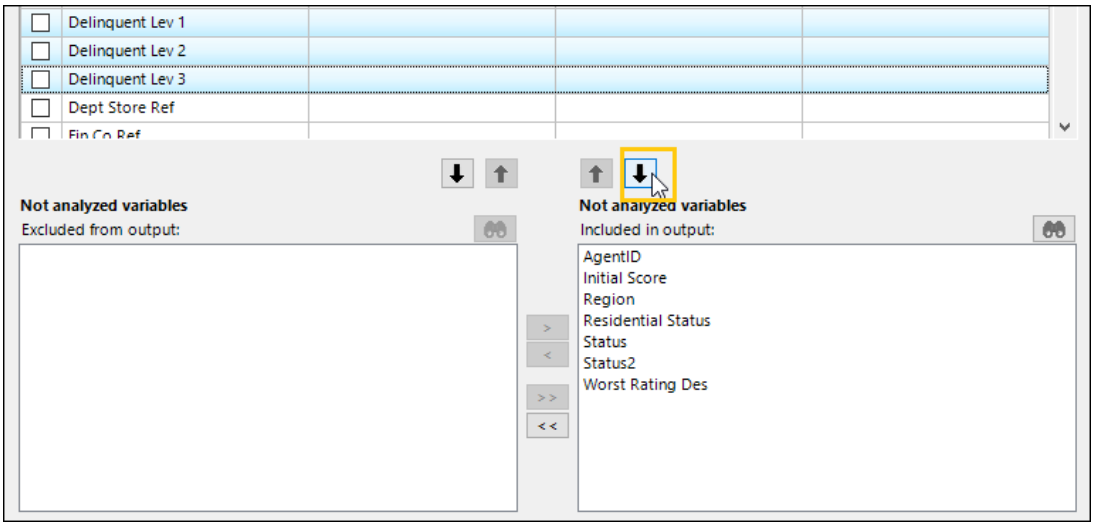
- All non-numeric variables are listed in the **Not analyzed variables** frame at the bottom-right. The variables in this list are not analyzed, but they are included in the output dataset. You can exclude them from the output using the left and right arrow buttons or just double-clicking on them.
- To exclude any numeric variables from analysis AND from the output, select them in the **Analyzed variables** list and click on the down arrow button on the left. In the example below, once you click on the highlighted arrow, the three selected variables *Delinquent Lev 1*, *Delinquent Lev 2*, and *Delinquent Lev 3* from analysis and from the output.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



To put any numeric variables back to the Analyzed list, select them and click the up arrow button.

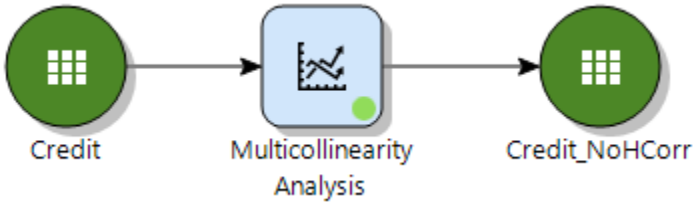
- To exclude any numeric variables from analysis, BUT still include them in the output, select them in the **Analyzed variables** list and click on the down-arrow button on the right. In the example below, once you click on the highlighted arrow, the three selected variables *Delinquent Lev 1*, *Delinquent Lev 2*, and *Delinquent Lev 3* will be excluded from analysis, but included in the output. To put any numeric variables back to the Analyzed list, select them and click the up-arrow button.



To put any numeric variables back to the Analyzed list, select them and click the up arrow button.

Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the node, the output dataset node appears on the workflow canvas. The output dataset contains only the variables selected in the wizard. Double-click on the output dataset to view the results.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also:

Correlations

Variable Selection

Variable Importance

Variable Selection

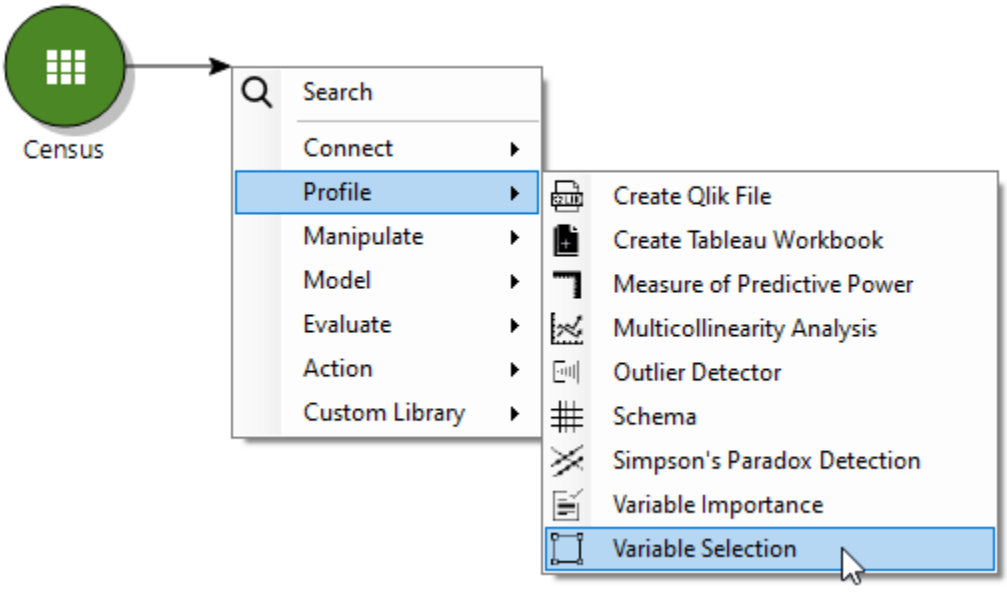
The **Variable Selection** node in the **Profile** palette provides a way to select variables based on measures of predictive power with respect to a dependent variable. After computing the necessary measures and ranking variables, the node produces the output dataset that contains only the variables selected by the user based on the statistics computed.

To add a variable selection operation to your workflow:

- Drag a connection arrow from a dataset node to any area on the Workflow canvas background. This brings up a context menu that allows you to select the desired operation. From the context menu, select **Profile --> Variable Selection** as shown in the example below. A new **Variable Selection** process node will be created.

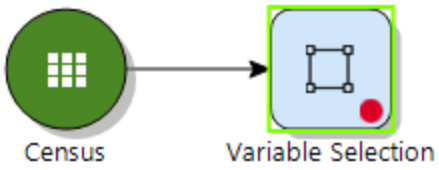
OR:

- Drag the **Variable Selection** node from the **Profile** palette onto the canvas and connect the input dataset to the node.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If the wizard does not open automatically, double-click the Variable Selection node or right-click on it and select **Modify**.



In the Variable Selection wizard:

- Specify the name for the output dataset with the variables to be selected.
- Select the dependent variable.
- Choose the measures you wish to use to rank and select the variables. The available measures depend on the type of the dependent variable (discrete or continuous). The formulas for most of these statistics can be found in the topic Mathematical Formulations: Decision Tree.

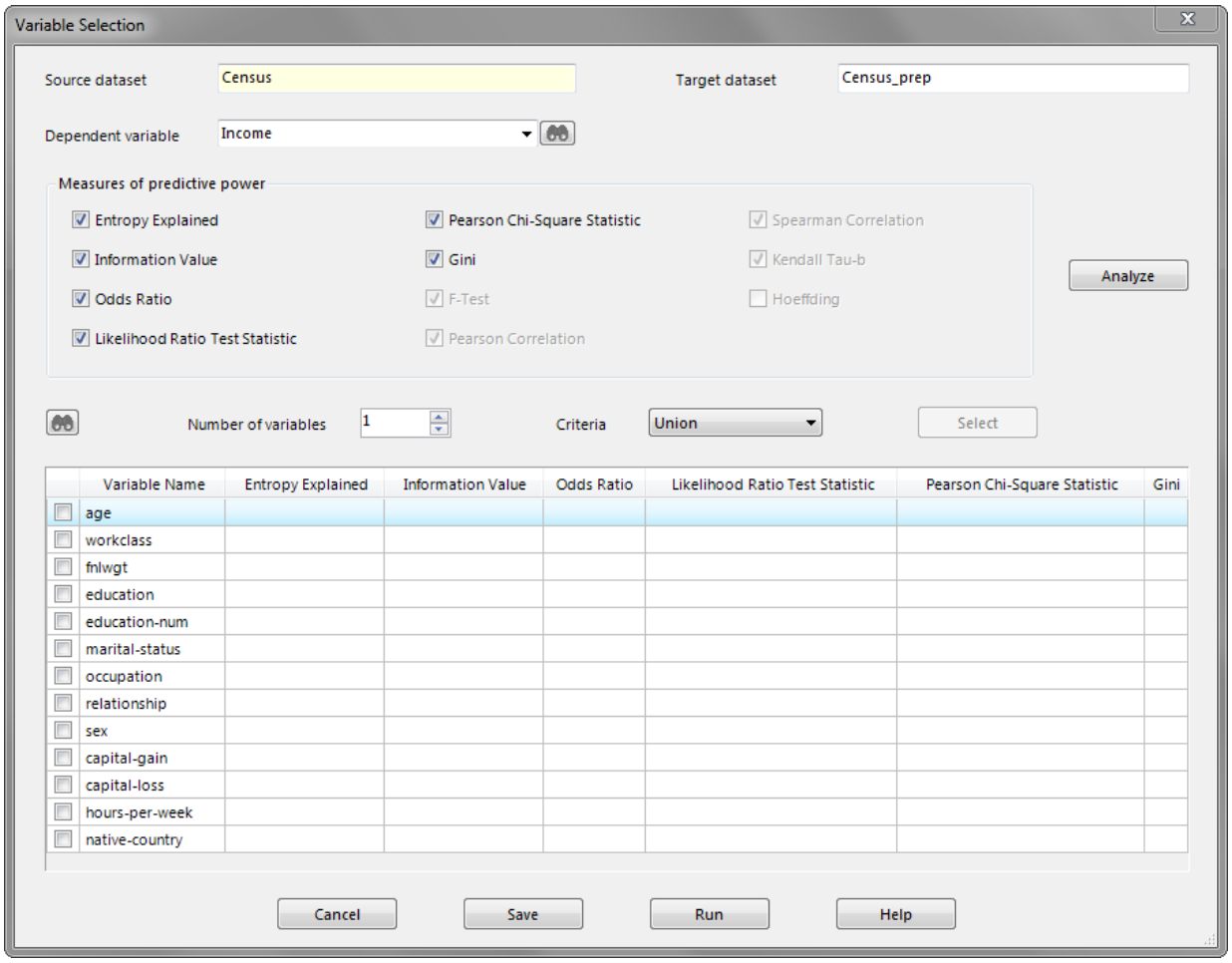
  Measures available in the case of a discrete DV:

  - *Entropy Explained*
  - *Information Value* (for binary DV only)
  - *Odds Ratio* (for binary DV only)
  - *Likelihood Ratio Test Statistic*
  - *Pearson Chi-Square Statistic*
  - *Gini*

  Measures available in the case of a continuous DV:

  - *Variance Explained*
  - *F-Test*
  - *Correlation:*
  - *Pearson Correlation*
  - *Spearman Correlation*
  - *Kendall's Tau-b*
  - *Hoeffding's D*

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- Click **Analyze** to compute the selected measures. The columns are sortable. Click on the column header for the desired measure to sort on it.
- Select the desired variables. If necessary, filter them by specifying the number of variables requested and the criterion to use (*Union* or *Intersection*).

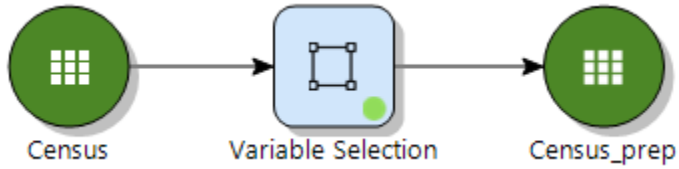
  Example: If you specify *Number of variables* = **5** and *Criteria* = **Union**, then click **Select**, then for each measure selected above, the wizard will pick top 5 variables with the highest values of this measure, then it will take the union of these sets of variables over all selected measures, and the resulting set will be selected in the lower frame. There may be 5 or more variables selected as a result.

  If you specify *Number of variables* = **5** and *Criteria* = **Intersection**, then click **Select**, then for each measure selected above, the wizard will pick top 5 variables with the highest values of this measure, then it will take the intersection of these sets of variables over all selected measures, and the resulting set will be selected in the lower frame. There may be 5 or fewer variables selected as a result.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

	Variable Name	Entropy Variance	Information Value	Odds Ratio	Likelihood Ratio Test Statistic	Pearson Chi-Square Statistic	Gini Variance
☑	relationship	0.21001	1.51571		3738.44479	3390.75647	0.20826
☑	marital-status	0.20025	1.36872		3564.74371	3297.66989	0.20255
☑	age	0.11569	1.07773		2059.45484	1605.92562	0.09864
☑	occupation	0.1146	0.75986		2040.01712	1963.24514	0.12059
☑	education	0.11289	0.71556		2009.55073	2108.08704	0.12948
☐	education-num	0.11289	0.71556		2009.55073	2108.08704	0.12948
☐	hours-per-week	0.07121	0.46187		1267.59555	1221.22078	0.07501
☐	sex	0.04529	0.29493	3.50612	806.30704	730.98799	0.0449
☐	workclass	0.03048	0.18992		542.63032	567.47634	0.03486
☐	capital-loss	0.02811	0.17844		500.36857	605.14934	0.03717
☐	native-country	0.01066	0.07839		189.71917	157.35931	0.00967
☐	fnlwgt	0.00027	0.00168		4.8835	4.78774	0.00029
☐	capital-gain	0.11527			2052.04506	2267.36886	0.13926

Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the operation, the node representing the resulting dataset appears on the workflow canvas, and the dataset also appears in the Project Explorer.



Double-click the output dataset to open it. The *output* dataset contains only the variables selected in the Variable Selection node wizard.

Simpson's Paradox Detection

The **Simpson's Paradox Detection** node in the **Profile** palette performs automatic detection of the Simpson's Paradox in your data with respect to the selected variable. The paradox manifests itself when a specific trend in the target variable is observed in subsets of data but the opposite trend is observed in the dataset as a whole.

Detecting this phenomenon in your data helps avoid the misinterpretation of trends and drawing wrong conclusions about biases or causal relationships when there are none or when the opposite ones exist. The results can be taken into account to make the necessary adjustments in subsequent data profiling and predictive modeling to avoid misleading results.

The Simpson's Paradox Detection node takes a single dataset as input and produces a report on the output.

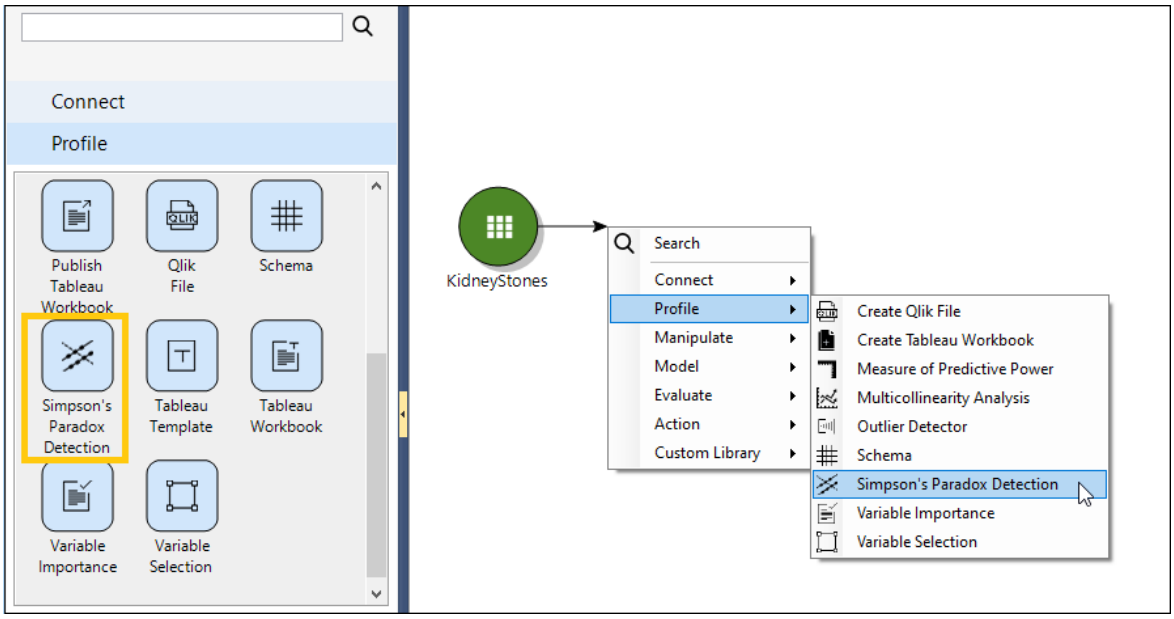
To add a Simpson's Paradox Detection operation in your workflow:

- Drag a connection arrow from the desired dataset node to any area on the Workflow canvas background. From the context menu, select **Profile --> Simpson's Paradox Detection** as shown in the example below. A new **Simpson's Paradox Detection** node will be created, and the wizard will open.

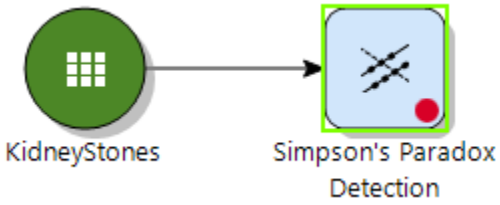
  OR:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Drag the **Simpson's Paradox Detection** node from the **Profile** palette onto the canvas and connect the input dataset to the node.



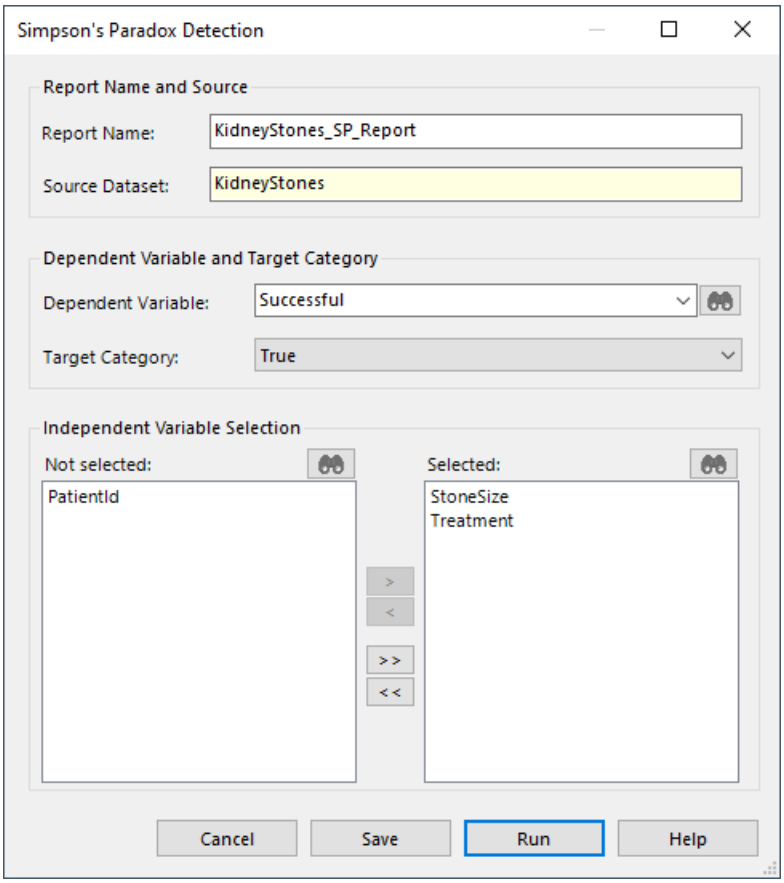
To open the node wizard, double-click the Simpson's Paradox Detection node or right-click on it and select **Modify**.



In the wizard page:

- Specify the name for the output report.
- Select the dependent variable (DV) to analyze trends for.
  - For a discrete dependent variable, select the target category whose frequency must be examined for Simpson's Paradox.
- Select the independent variables (IVs). Trends will be analyzed within the pairs of the selected independent variables.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.

The subsets analyzed for the Simpson's Paradox are determined by the splits on the selected independent variables (IVs). For each pair of independent variables (V1,V2), the DV trends with respect to V1 are examined in the whole dataset and in subgroups defined by V2, then they are compared. **Note**: The performance of the algorithm is proportional to the square of the number of independent variables. If you select more than 10 variables, a warning message will be displayed.

The algorithm steps:

For each pair of independent variables, a pair of decision trees with the selected dependent variable is built, and their "trend signs" are computed and compared.

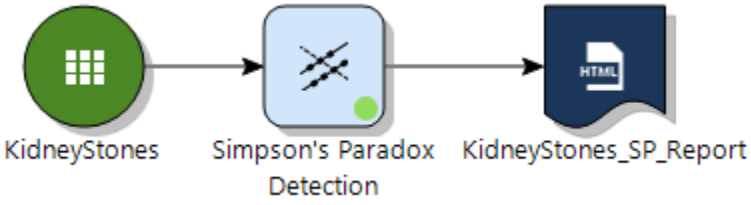
- For each independent variable V:
  - Build a decision tree DT1 with a single split on V
  - Let the remaining set of IVs be called *Conditioning Variables* (CVs). For each conditioning variable CV:
    - Build a decision tree DT2 with two levels: The root node is split on CV, then each child node is split on V. *Note: The same binning structure for V (the exact same split points and bins) is used for the single split in DT1 and all second-level splits in DT2.*

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

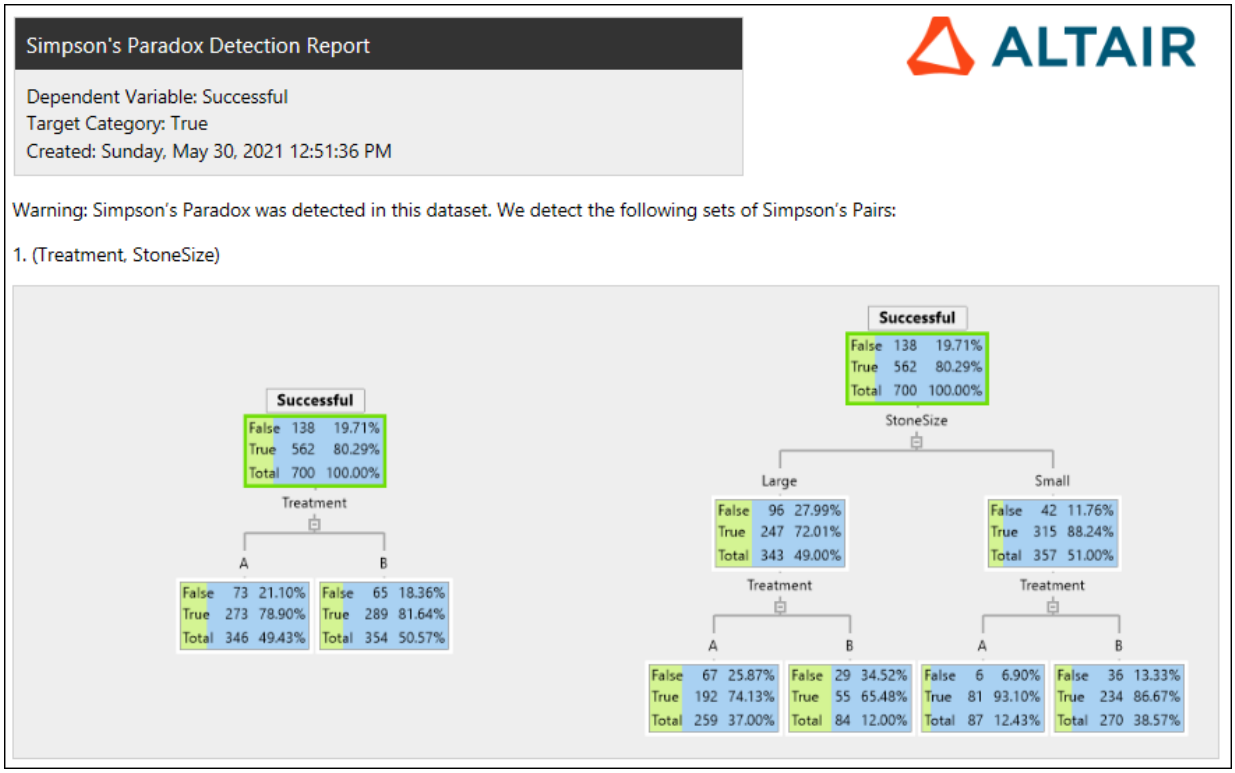
The trends in T1 and T2 are then evaluated and compared as follows:

- Calculate the "trend signs" of splits:
  - For each adjacent pair of terminal nodes in T1, the average value (for continuous DV) or the percentage of the target category (for discrete DV) in the left node is subtracted from the corresponding value in the right node. The trend sign of T1 is calculated by taking the sign of the average of these metrics for all adjacent pairs of terminal nodes.
  - In T2, the same is calculated separately for each split at the second level, then the sign of the average is taken as the trend sign of T2.
- The trend signs of the two trees are compared. If the trends have opposite signs, the paradox is detected, and the pair of independent variables (V, CV) is reported as a “Simpson’s Pair”.

The output report node appears on the workflow once the process is finished.

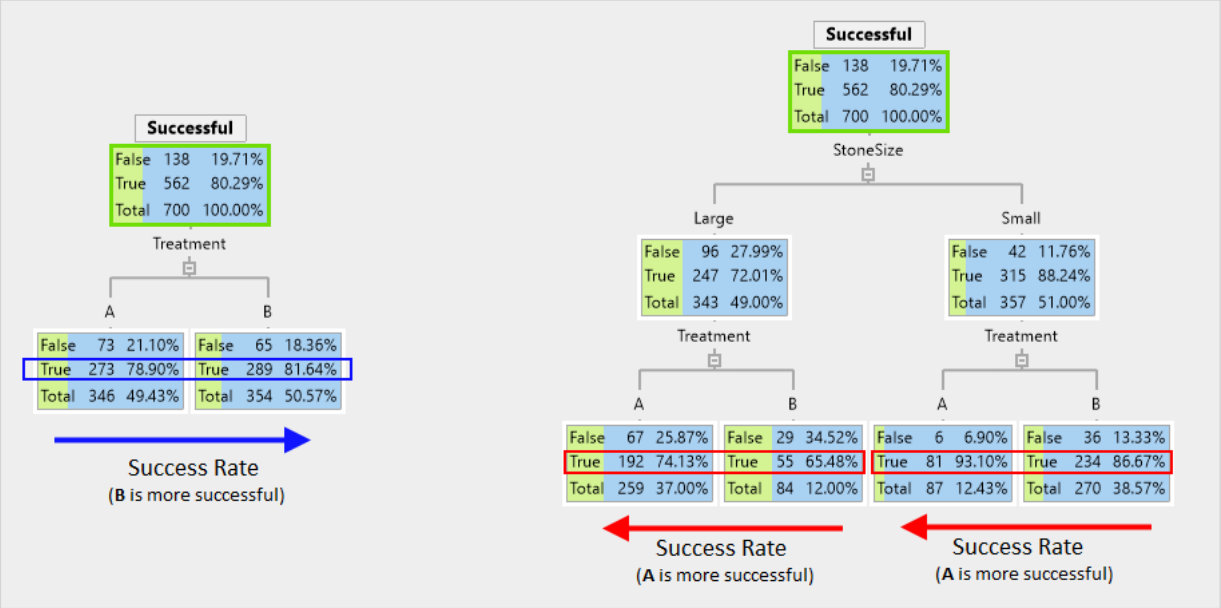


Double-click on the output report node to view the results.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

In this well-known example, Simpson's Paradox was detected for the pair of independent variables *Treatment* (treatment type) and *Stone Size* with respect to the dependent variable indicating treatment success or failure. The trend with respect to treatment type is reversed when considering subgroups defined by stone size. The opposite trend is recorded when considering treatment type and stone size versus the trend recorded when considering the treatment type only. As shown below: According to T1, when we look at the entire dataset as a whole, Treatment B seems to be more successful than treatment A (78.9% success rate for A versus 81.64% for B). However, within each subgroup defined by the stone size (large and small), the trend is the opposite: Treatment B is LESS successful than A for large stones (74.13% success rate for A versus 65.48% for B), and Treatment B is also LESS successful than Treatment A for small stones (93.1% success rate for A versus 86.67% for B). Hence the variables *Treatment* and *StoneSize* form a "Simpson's pair". In summary, the trend of success rate is reversed when you combine both subgroups of Large and Small stone size.



Schema

The **Schema** node in the **Profile** palette is provided to simplify the model development process by creating reusable schemas or templates for selecting input variables: the dependent variable, target category (if applicable), and independent variables. Similar to Variable Lists, the Schema node saves time by removing the need to select input variables from scratch for each model if multiple models need to use the same variables and targets.

The Schema is essentially a list of variables with roles (dependent or independent) associated with each variable. If the dependent variable is binary, it also indicates the target category selection.

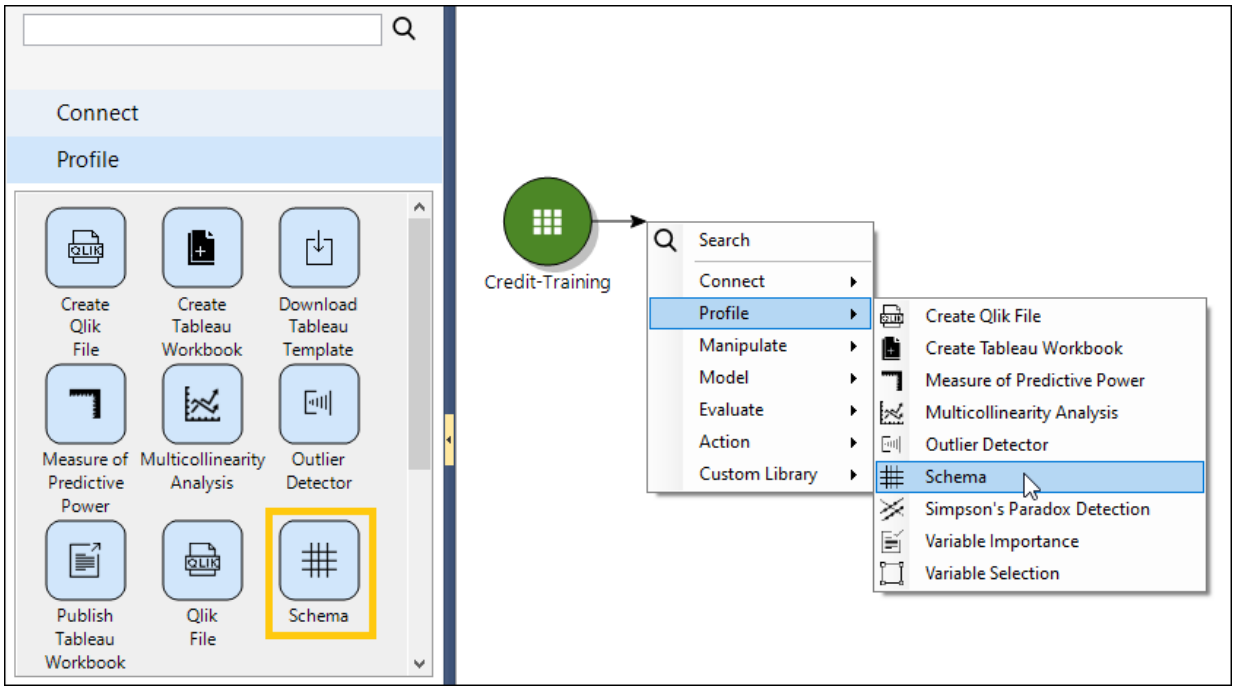
To add a Schema node to your workflow:

- Drag a connection arrow from a dataset node to any area on the Workflow canvas background. This brings up a context menu that allows you to select the desired operation. Select **Profile --> Schema** as shown in the example below. A new **Schema** node will be created.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

OR:

- Drag the **Schema** node from the **Profile** palette onto the canvas and connect the input dataset to the node.

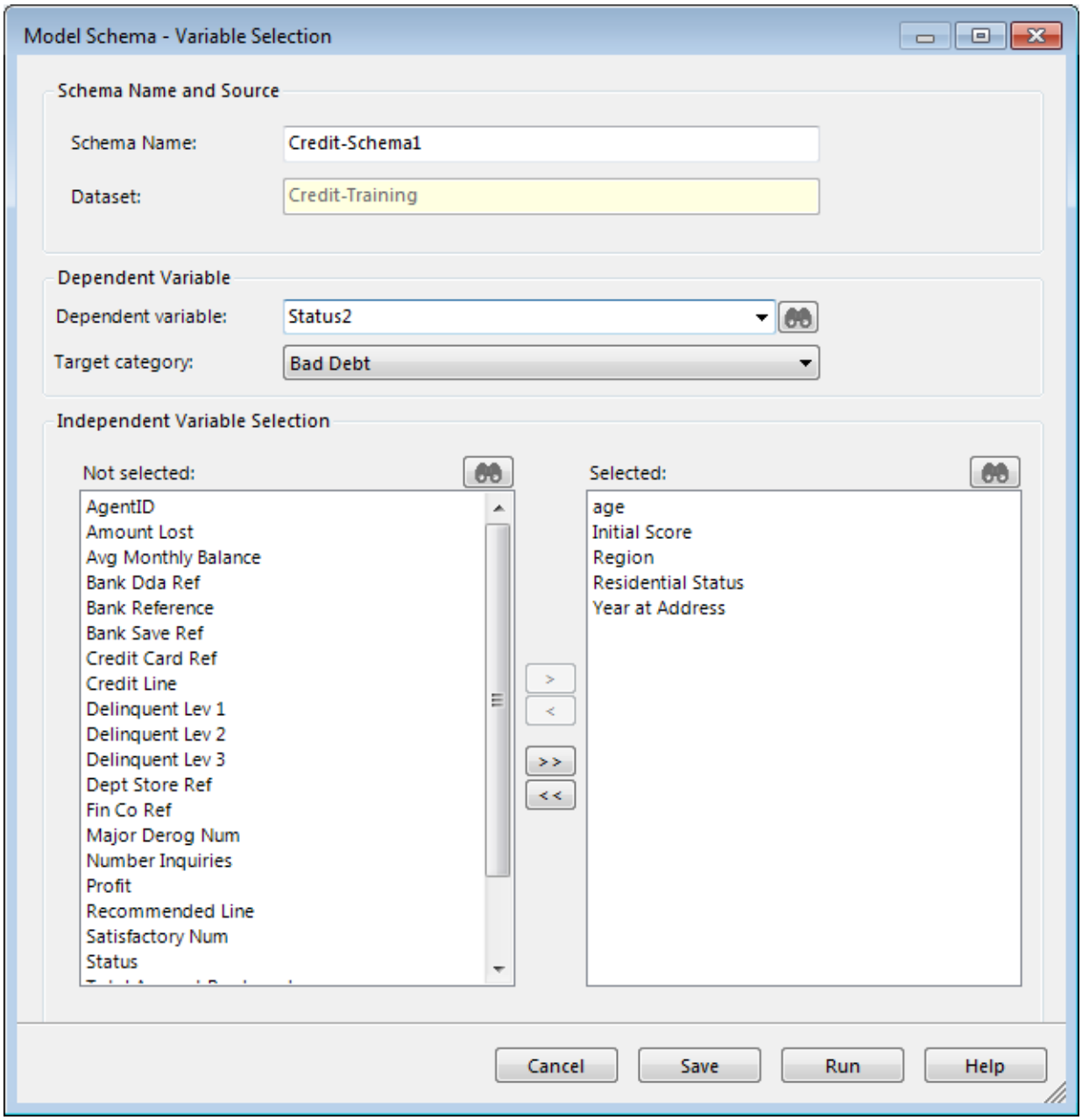


If the wizard does not open automatically, double-click the Schema node or right-click on it and select **Modify**.



- Specify the schema name
- Select the dependent variable. If the dependent variable is binary, select the target category
- Select the independent variables

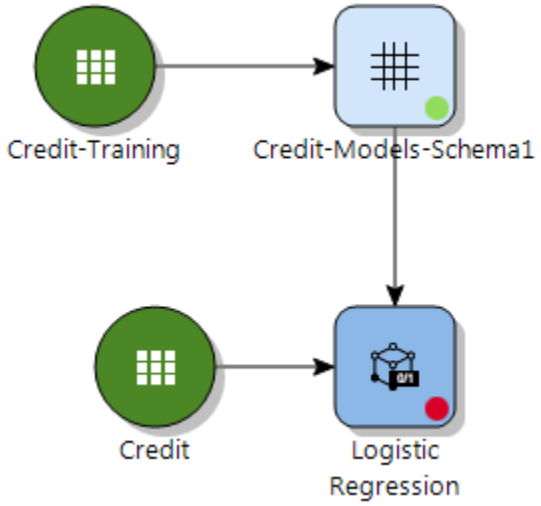
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



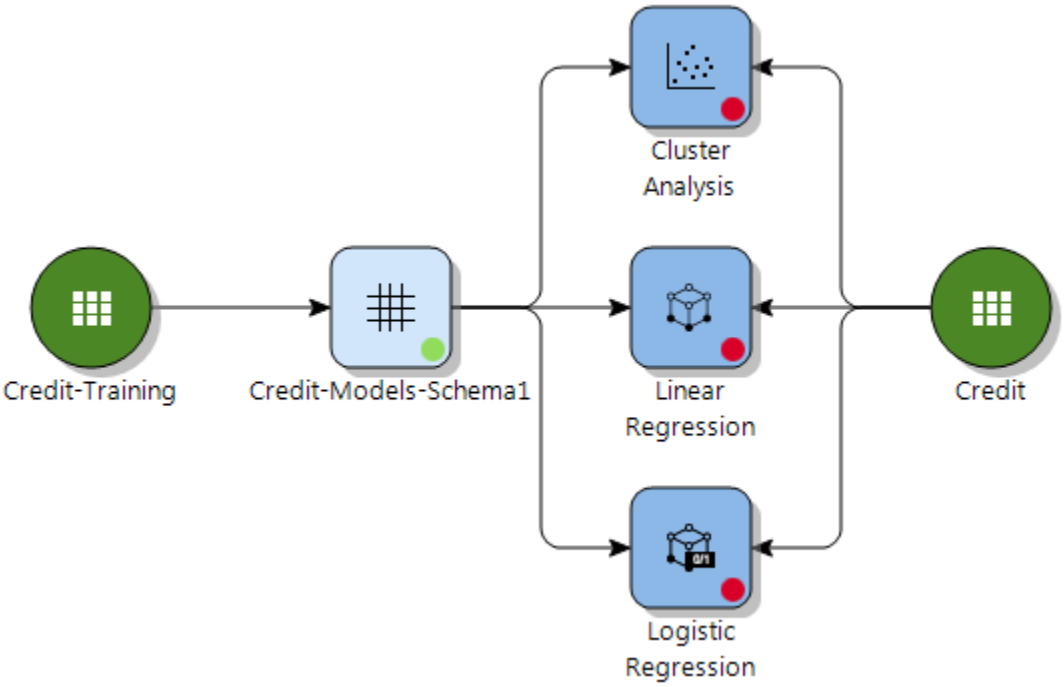
Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.

After adding a new model node to the workflow, attach the schema node to the model node. The model will inherit the input variable settings from the schema once you open the model wizard.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Although a Schema node should normally be used for models with the dependent variable (DV) of compatible type, the dependent variable or its type do not necessarily have to match. If they don't, the model wizard permits manual selection of any compatible DV. This also applies to models without a dependent variable (Cluster Analysis), in which case all variables in the Schema are inherited. In the example below, the same schema defined above is used for several models of different types (Logistic Regression requires a categorical DV, Linear Regression requires a continuous DV, and Cluster Analysis has no DV).



The selected / deselected variables in the wizards of models that use a Schema behave the same way as when a Base Model is connected. The independent variables selected in the Schema become included in the model wizard (provided that they are available in the model's parent dataset) just as they are inherited from the Base Model.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Exceptions: A Schema node cannot be connected to Decision Trees, Strategy Trees, and Market Basket Analysis.

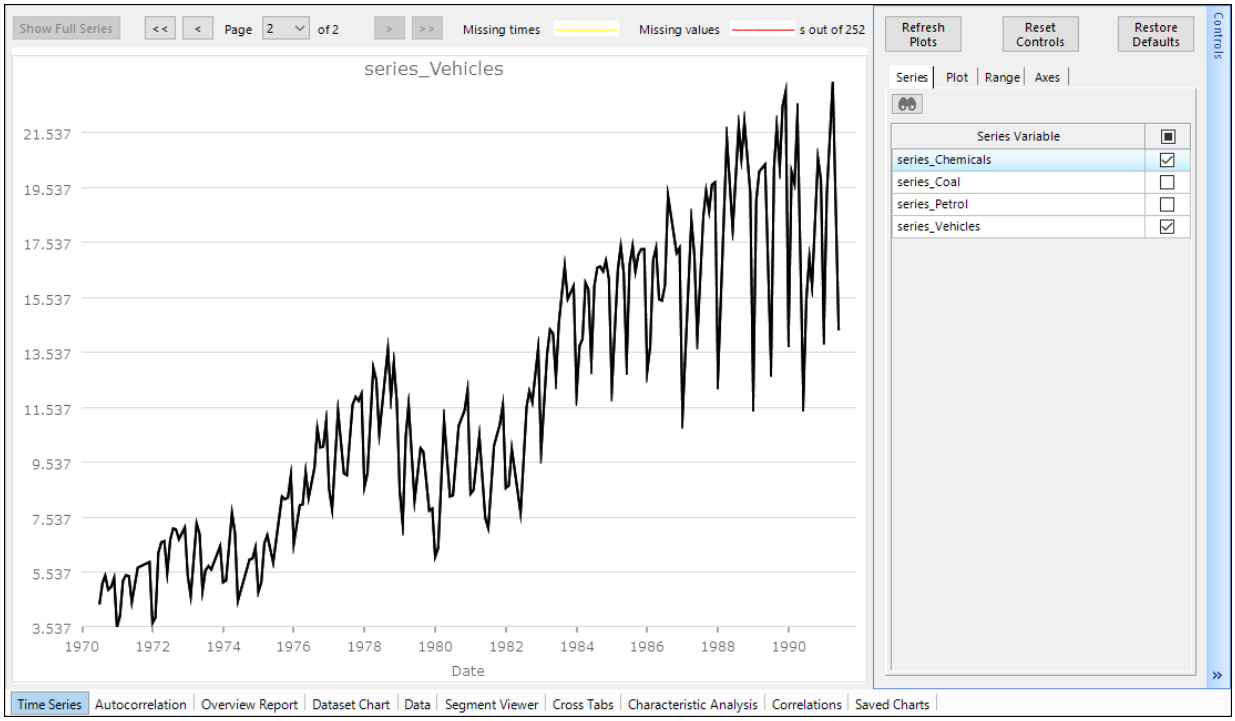
See also:

- Variable List
- Variable Selection

The **Time Series** tab is the first tab in datasets produced by the Make Time Series node of the **Manipulate** palette.

The tab consists of two panels: the Plot panel and the Control panel.

The first time you open the tab, the Plot panel is blank. Select the series to plot from the list in the *Series* tab of the Control panel and then press the **Refresh Plots** button.



Control Panel

The Control panel can be hidden to provide full view of the plot. To show or hide the Control panel, click on the vertical blue bar (labeled "**Controls**") on the right border of the Time Series tab.

The plot display and configuration controls are organized in four tabs: ***Series***, ***Plot***, ***Range***, and ***Axes***.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The top three buttons in the Control panel perform the following functions:

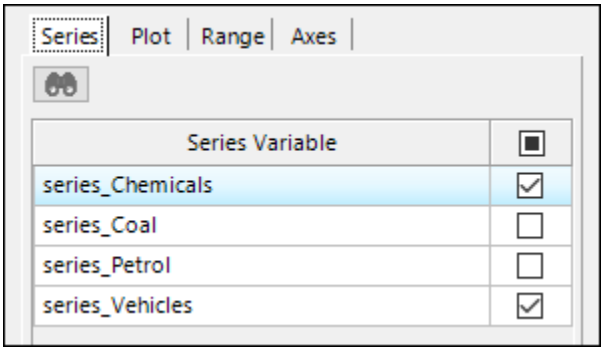
- **Refresh Plots**: redraw the plots selected in the Series tab in accordance with the currently defined configuration
- **Reset Controls**: return the controls to their state when the view was last opened and redraw the plots
- **Restore Defaults**: return the controls to their default state, and remove the plots

Series Control

The Series tab allows you to select the series variable to plot. Using the checkboxes on the right, select the series to display in the Plot panel. To select or deselect all series, click in the header checkbox.

Both columns in this grid are sortable, and the order of the rows can be changed by dragging rows to new positions.

If there are many variables, click the binoculars button above the grid to open the Field Finder tool.



Plot Control

Specify the Plot Configuration options:

- One series per plot: each series is plotted on its own panel with its own vertical axis
- All series on one plot: all series are plotted on the same panel, each with its own line type

If you choose one series per plot, specify the number of plots per page.

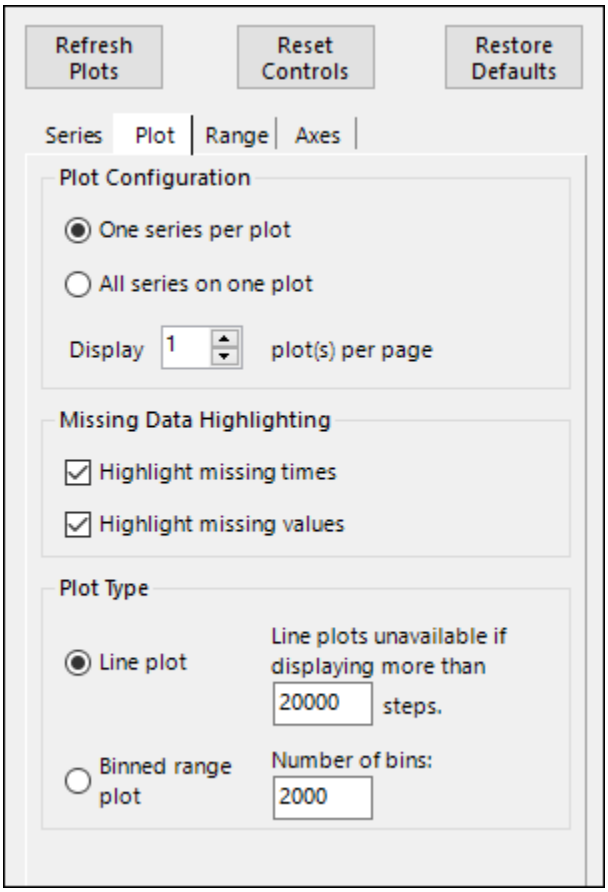
For the Missing Data Highlighting options, choose whether to plot coloured rectangles that highlight missing time points and missing data points.

Specify the Plot Type options:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- *Line plot* (the default number of steps is 20000)
- *Binned range plot* (the default number of bins is 2000)

If the number of time steps being displayed is greater than the specified number of steps for the Line Plot, the plot type automatically switches to the Binned range plot.



Range Control

The sections "*Enlarge Display Range*", "*Contract Display Range*", and "*Shift Display Range*" represent an interactive instruction to incrementally add or remove data to / from the display.

- Specify the number of time steps to add or remove (the default number is 100).
- The available time step types depend on the type of time variable: Aside from the default option "time steps" for all types of variables, Date and Timestamp variables additionally have years, quarters, months, weeks, and days; Timestamp variables additionally have minutes, hours, and seconds.
- Click the *Earlier* or *Later* button depending on whether the specified number of time steps should be subtracted from the start or added to the end of ranges in the Start (End) of Display Range group (see below).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- If the full range of the data is exceeded, the appropriate start and end times are set automatically. If the full series is plotted, all of these controls are disabled.

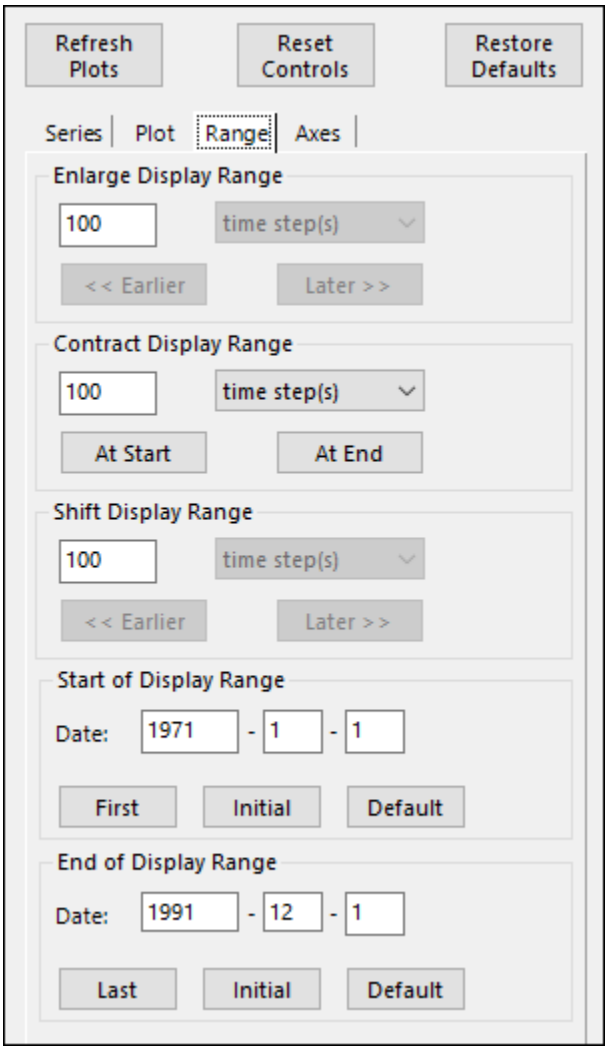
The sections "*Start / End of Display Range*" provide full control over the range of times displayed in the plots. Click **Refresh Plots** once you specify the desired start/end points.

- For Numeric variable, specify the desired start 'time' point value.
- For Date variable, specify the desired start date.
- For Timestamp variable, specify the desired start date and time.

The default values are determined automatically based on the first and last time points in the dataset and the rules specified above.

- The buttons **First / Last** set the Start (End) time to be the first (last) time point in the full dataset.
- The **Initial** button sets the Start (End) time to Start (End) time when the dataset viewer was opened.
- The **Default** button sets the Start (End) time to the default.

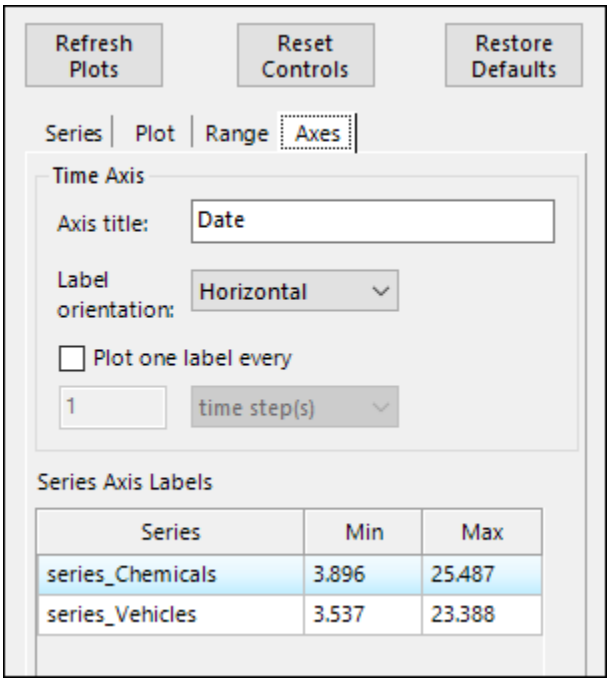
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Axes Control

- Specify the Axis title to be displayed for the horizontal axis. The default is the name of the Time Variable that was specified in the Make Time Series node.
- Select the label orientation for the marks on the horizontal axis (*Horizonal, Diagonal,* or *Vertical*). The default is *Horizontal*.
- Select whether to plot the marks and associated labels on the horizontal axis. If selected, specify at what interval they should be placed (e.g., every 5 time step(s)).
- In the Series Axis Labels grid, specify where to place marks and associated labels on the vertical axes for each series by indicating the Min and Max points.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Plot Panel

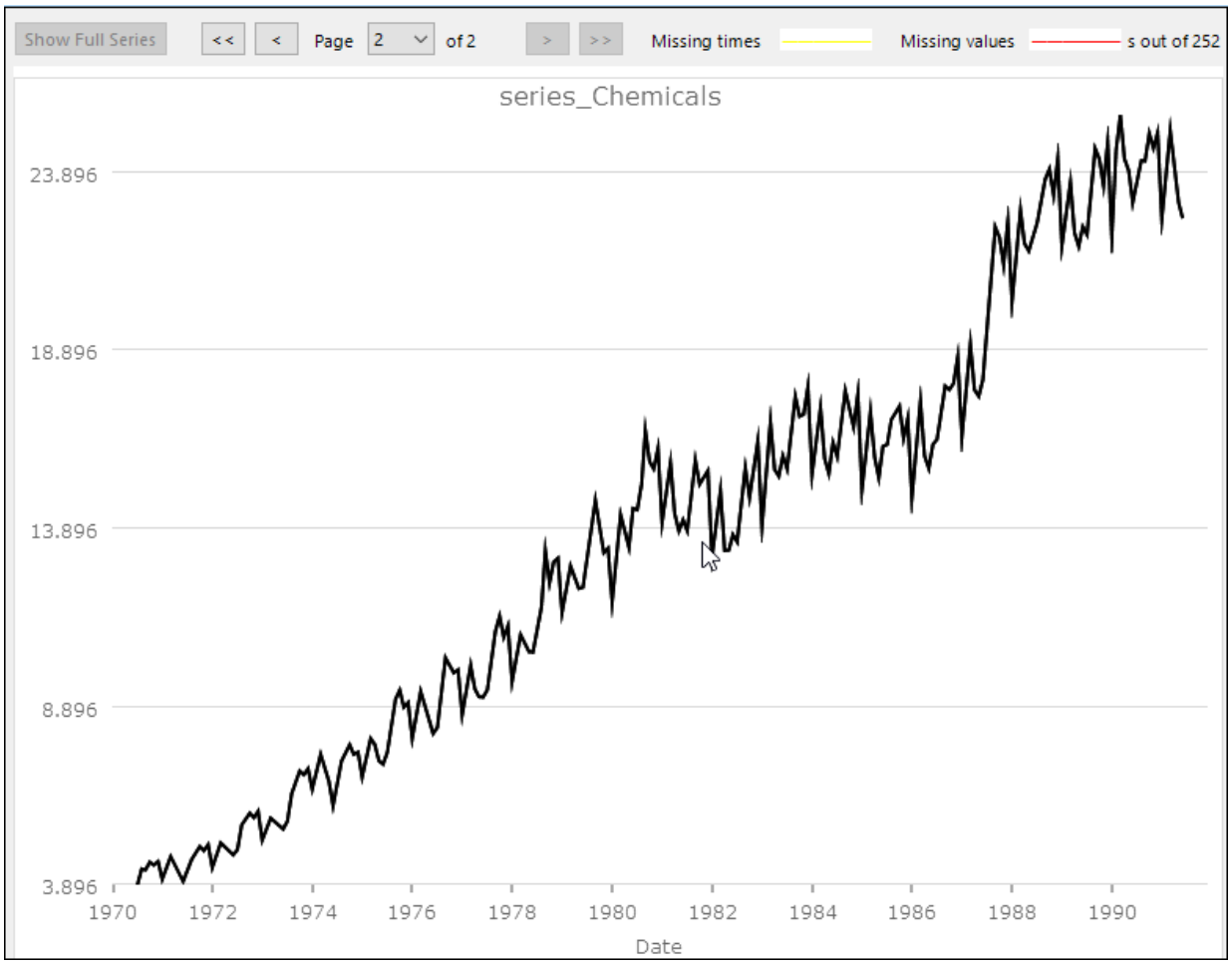
The plot navigation controls are located at the top of the plot panel.

When applicable, the **Show Full Series** button refreshes the plots with all the data shown. Note that this may involve an automatic change in the plot type from line plot to binned range plot, if the threshold number of time steps from the Plot tab is exceeded (see above).

The number of pages is equal to the number of series plotted - if the option "*One series per plot*" is selected in the Control panel. Use the arrows or the *Page* drop-down list to navigate to the desired plot.

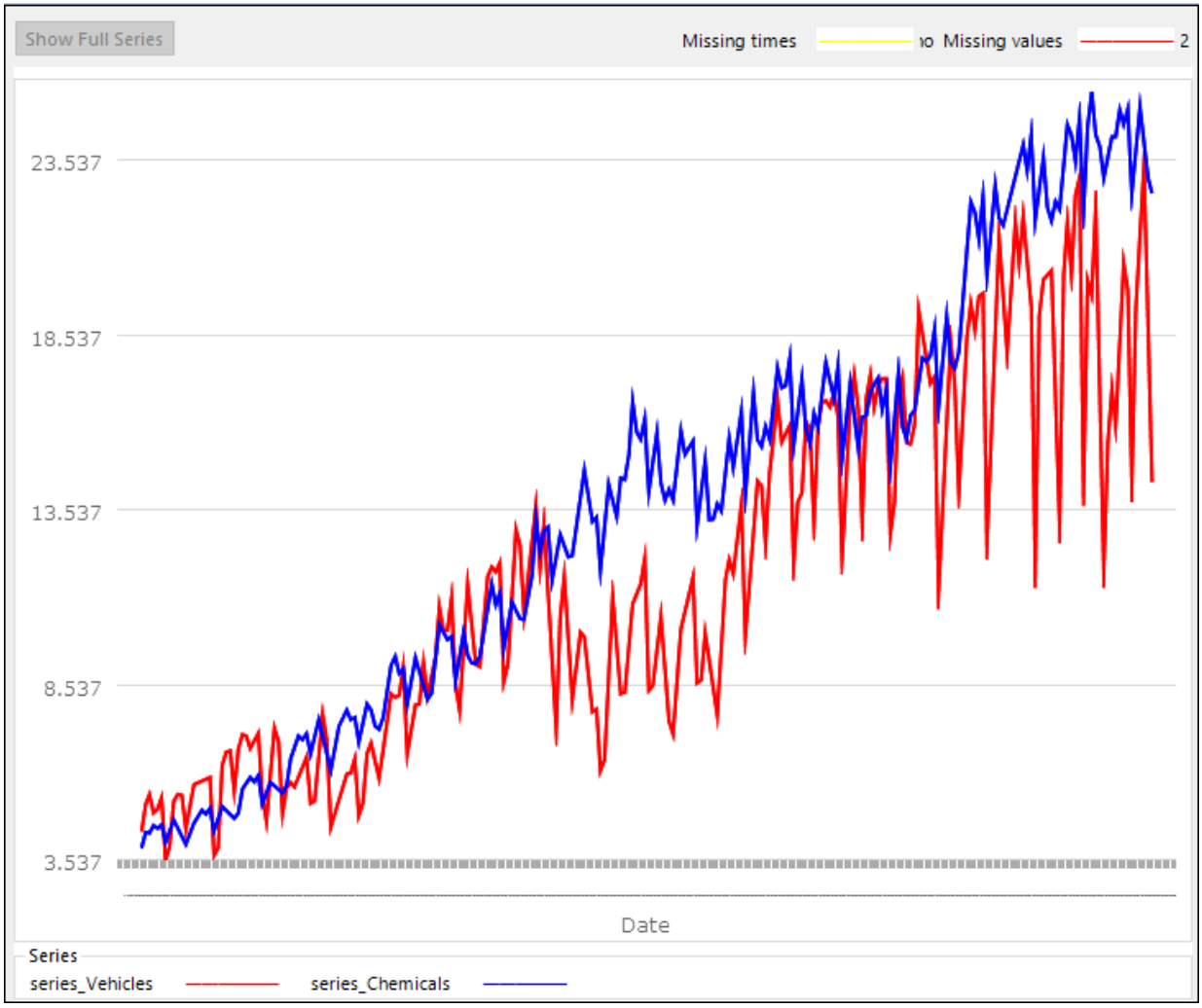
When the Missing Data Highlighting options are selected in the Control panel (see above), missing values in time points and data points (if any) are highlighted in different colors according to the legend at the top of the panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If all series are on one plot, a legend at the bottom of the tab identifies each plot on the screen.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Binned range plot

The time steps are binned into a user-defined number of bins, each with an equal number of steps. For each bin, a blank rectangle is plotted. The horizontal coordinates of the rectangle are given by the minimum and maximum (non-missing) time steps in the bin. The vertical coordinates are given by the minimum and maximum of the series variable.

See also:

- Make Time Series
- Autocorrelation (Dataset Tab)

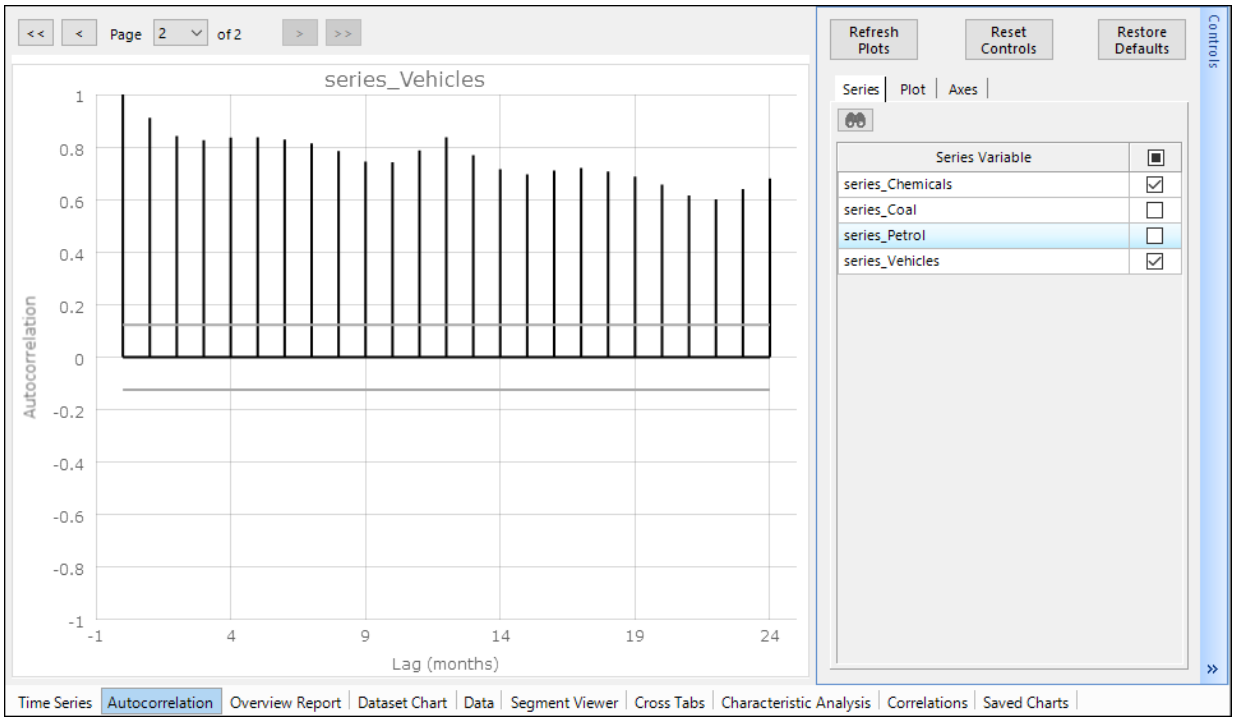
Autocorrelation (Dataset Tab)

The **Autocorrelation** tab is the second tab in datasets produced by the Make Time Series node of the **Manipulate** palette.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The tab consists of two panels: the Plot panel and the Control panel.

The first time you open the tab, the Plot panel is blank. Select the series to plot from the list in the *Series* tab of the Control panel and then press the **Refresh Plots** button.



Control Panel

The Control panel can be hidden to provide full view of the plot. To show or hide the Control panel, click on the vertical blue bar (labeled "**Controls**") on the right border of the Autocorrelation tab.

The plot display and configuration controls are organized in three tabs: ***Series***, ***Plot***, and ***Axes***.

The top three buttons in the Control panel perform the following functions:

- **Refresh Plots**: redraw the plots selected in the Series tab in accordance with the currently defined configuration
- **Reset Controls**: return the controls to their state when the view was last opened and redraw the plots
- **Restore Defaults**: return the controls to their default state, and remove the plots

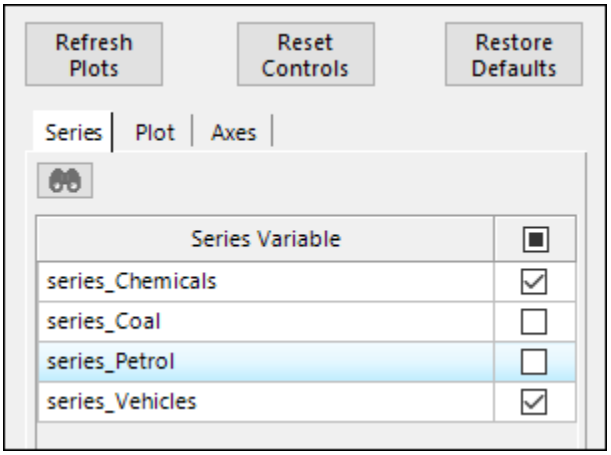
Series Control

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The *Series* tab allows you to select the series variable to plot. Using the checkboxes on the right, select the series to display in the Plot panel. To select or deselect all series, click in the header checkbox.

Both columns in this grid are sortable, and the order of the rows can be changed by dragging rows to new positions.

If there are many variables, click the binoculars button above the grid to open the Field Finder tool.



Plot Control

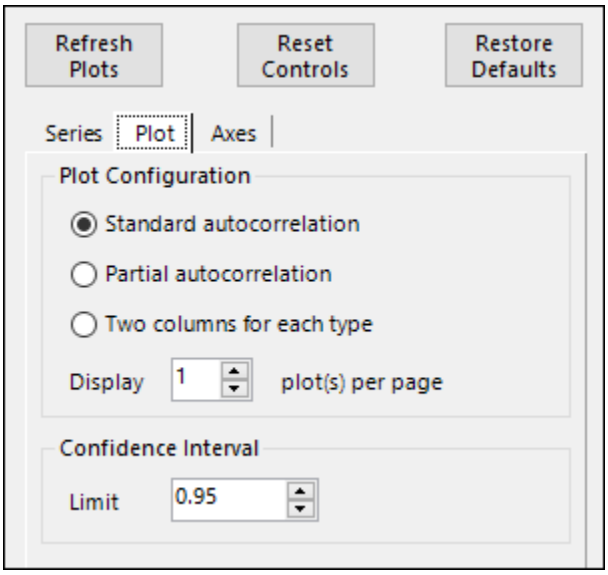
Specify the Plot Configuration options:

- Standard autocorrelation – one column of autocorrelation plots
- Partial autocorrelation – one column of partial autocorrelation plots
- Two columns for each type – one column for the standard autocorrelation plots and one for the partial autocorrelation plots

Specify the number of plots to display per page. In the two columns case, this means the number of rows as there will be two plots per row.

Specify the limit for the Confidence interval. The default is 0.95.

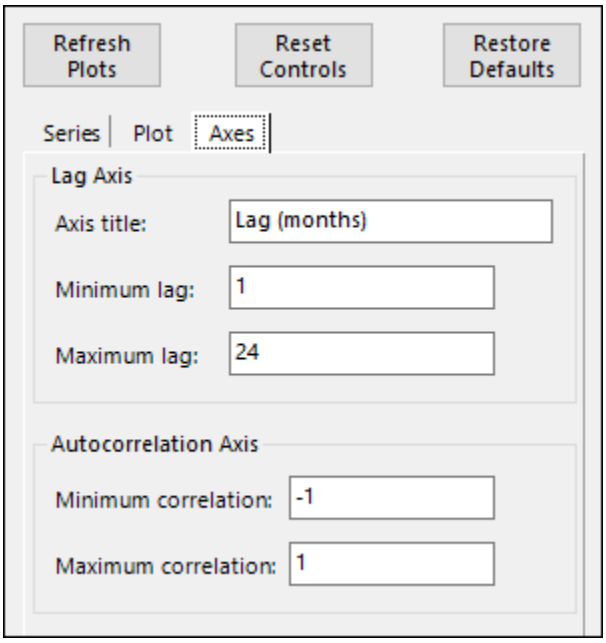
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Axes Control

- Specify the Axis title to be displayed for the horizontal axis. The default is Lag (<time_unit>), where time_unit was specified in the Make Time Series node.
- Specify the minimum lag (an integer number). The default is 0 for standard autocorrelation, and 1 for partial autocorrelation.
- Specify the maximum lag. The default depends on the type of the time unit:
  - Seconds and Minutes – 60
  - Hours – 48
  - Days – 28
  - Weeks – 52
  - Months – 24
  - Quarters – 20
  - Years – 20
  - No unit (integer) – 20

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

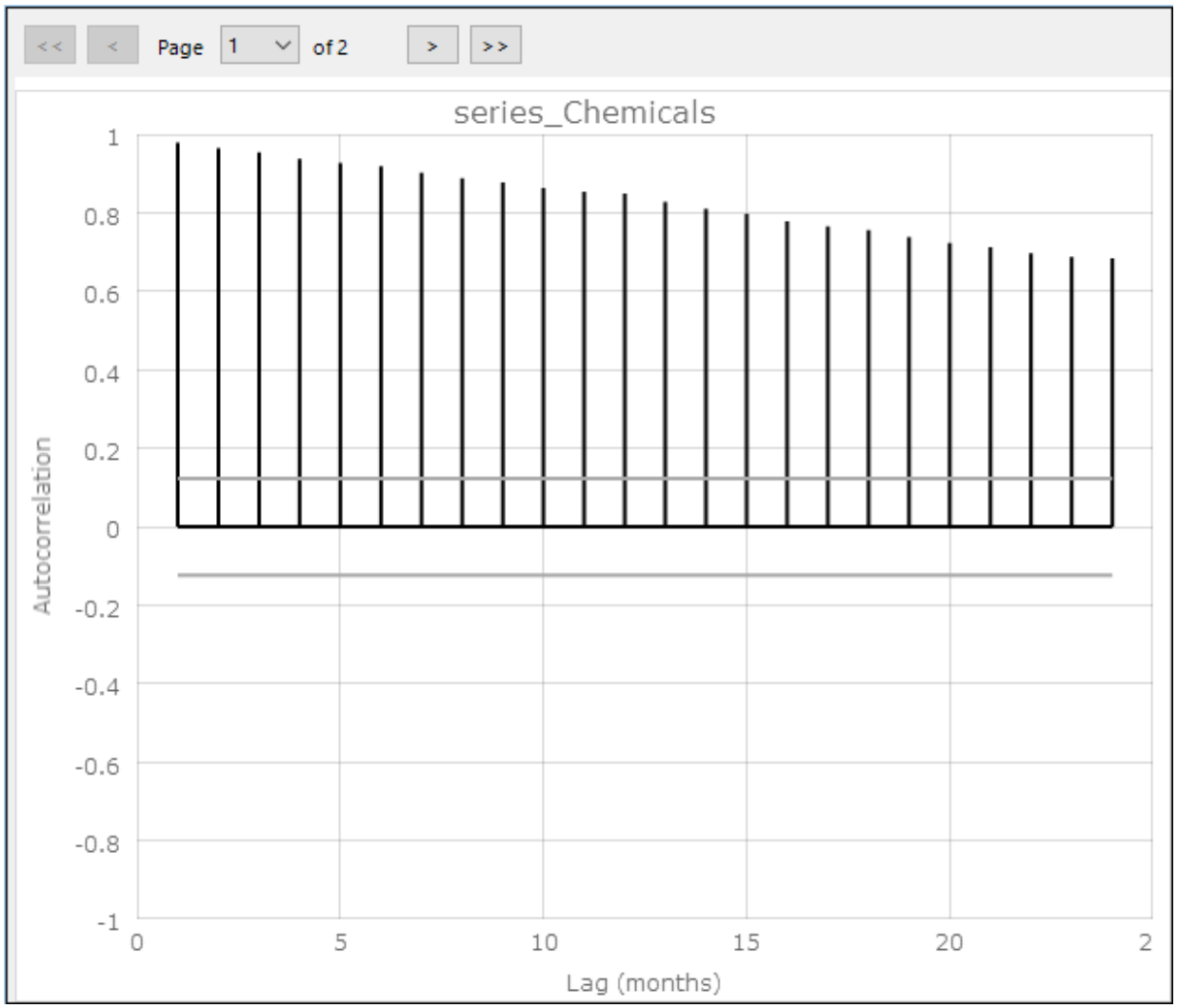


Plot Panel

The plot navigation controls are located at the top of the plot panel.

The number of pages is equal to the number of series plotted. Use the arrows or the *Page* drop-down list to navigate to the desired plot.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also:

- Make Time Series
- Time Series (Dataset Tab)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Modeling

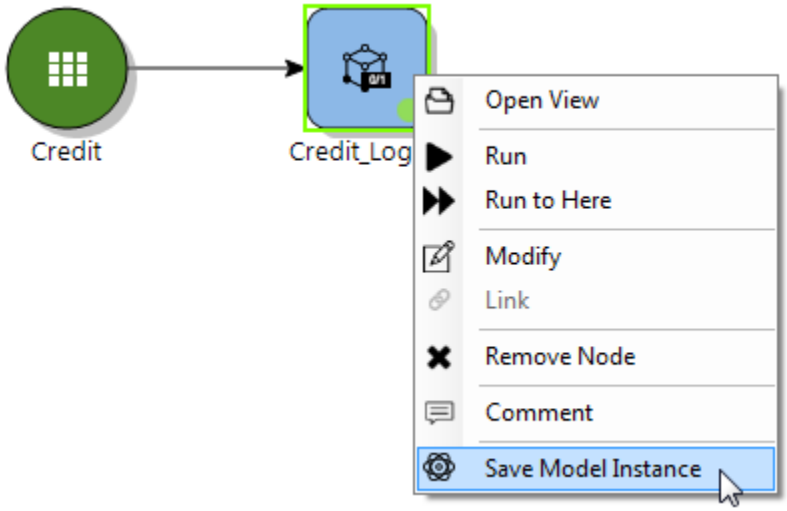
Model Instances

In any workflow, we distinguish **model process nodes** and **model instance nodes**.

There is an option for all model process nodes including trees to save a model instance that cannot be overwritten by re-running the model node.

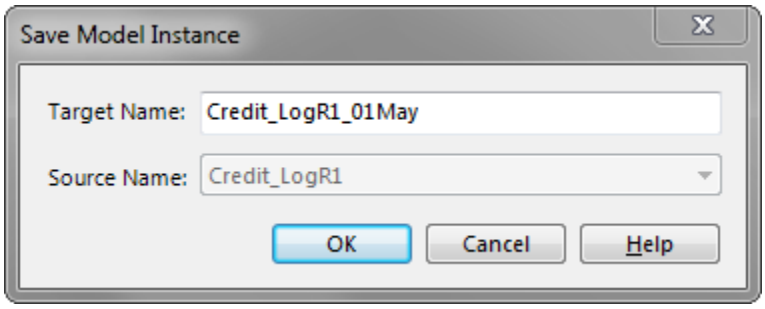
When you add a model to your workflow by dragging it from the Modelling group onto the workflow canvas, you create a **model process node.** It defines an *operation* of creating a model - a description of inputs, outputs, and training parameters - that can be re-run as part of running the workflow. The *model object* resulting from running a model process node and containing model results (the equation and model training statistics) is linked to the model process node and can be opened by right-clicking the model process node and selecting the **Open View** command. This model object is called a **model instance**.

Every time you re-run the model process node, the corresponding model instance behind it is overwritten. However, you can save the model instance for future use for scoring or other purposes. To save a model instance, right-click on the model process node and select the **Save Model Instance** command.

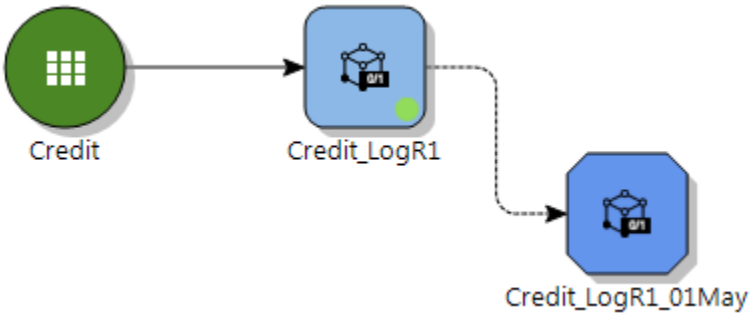


Specify the name for the new model instance. The name of the original model is shown in the Source Name field.

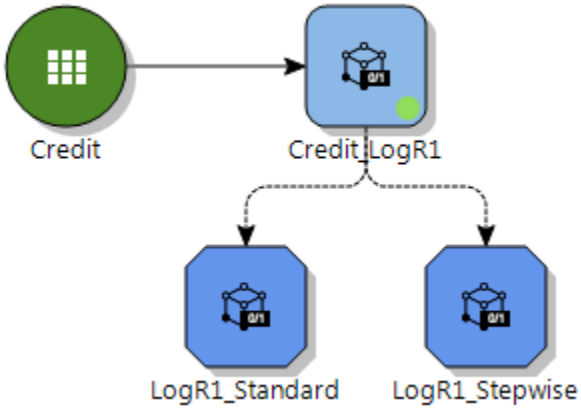
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



This will save a copy of the model instance resulting from the most recent run of the model process node. The new model instance is shown next to the model process node. It is connected to the process node by a dotted line indicating that this model instance cannot be overwritten by re-running the workflow.



You can save any number of model instance copies for the same model process node.



The connection between the source model node and the instance node can be deleted at any time, if desired. You will still be able to open and score the model instance after deleting the connection.

You can also save a model instance of another model instance. This is just creating a copy of a model instance. The copy is not connected to the original instance in the workflow.

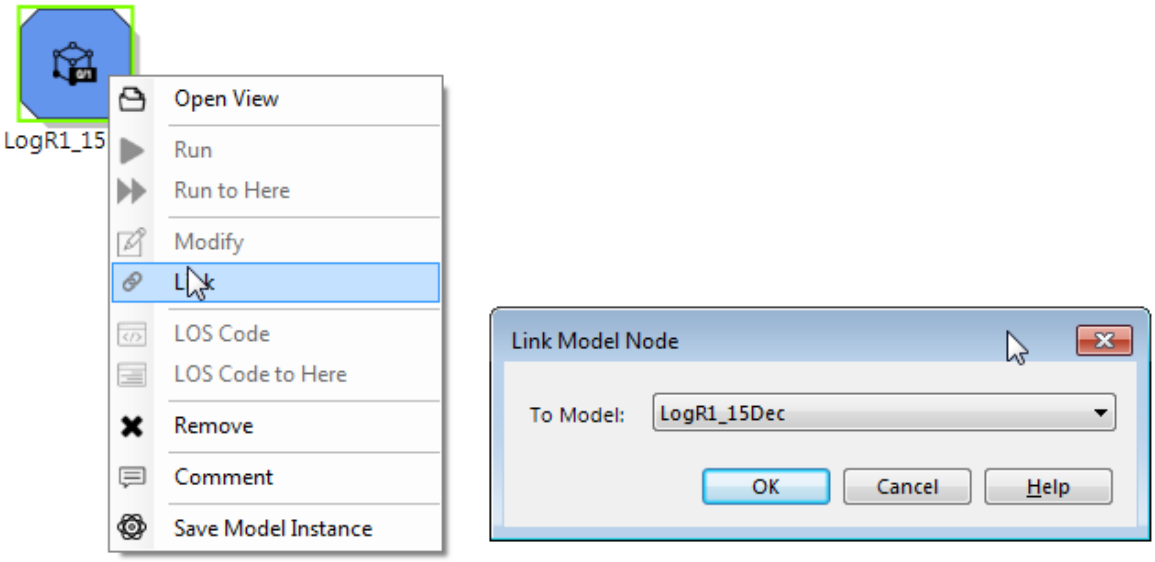
Operations with Model Instances

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

By definition, model instance nodes cannot be run and cannot be overwritten by re-running the workflow. The main operations defined for model instance nodes available in the right-click context menu are:

- **Open View:** Opens the model results view. For Decision and Strategy Trees, it opens the interactive tree view.

- **Save Model Instance:** Creates another model instance copy
- **Link:** Links the model instance node to a selected physical model in the Project Explorer. Any model instance node can be linked to any physical model - an output of a previous workflow execution available in the Project Explorer.

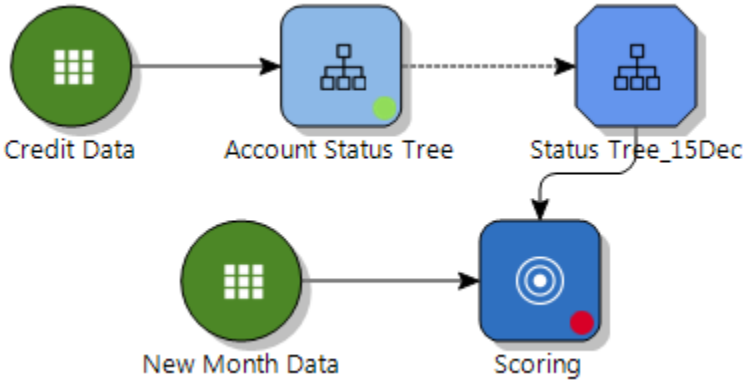


Scoring and Validation

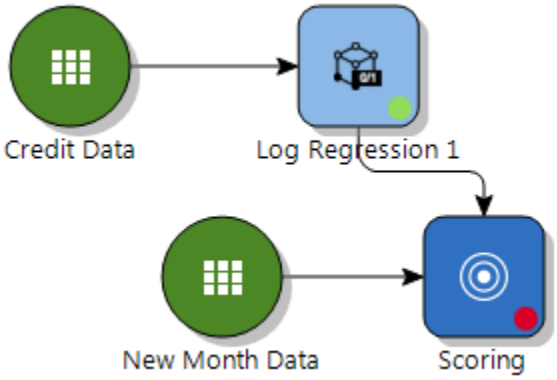
Model instances of all model types can be validated and scored. (Note: Validation does not apply to Market Basket Analysis). This means you can connect a Validation or Scoring node to any model instance node. In particular, for Decision Tree, Strategy Tree, and Market Basket Analysis models, validation and scoring are applicable ONLY to model instance nodes, and not applicable to model process nodes. So for these three model types you have to save model instances before scoring or validation. The reason is that manual user input is always required for scoring these model types. For example, trees are interactive and they need to be manually grown or modified before scoring, and in Market Basket Analysis you need to manually select the association rules to be deployed.

In the example below, the Account Status Tree node cannot be directly connected to the Scoring node. It can be scored only through creating a model instance node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



For Logistic Regression, however, the model process node can be directly connected to the scoring node:



See also: Workflows

Decision Trees

What is a Decision Tree

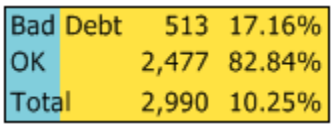
A ***decision tree*** is a statistical model used for classifying data. The information is represented in a tree-like structure where each node is either:

- a *terminal node*, also called a *leaf*, which is a classification of an instance, or
- a decision node, where some test is applied on a single attribute value, with one branch and sub-tree for each possible outcome of the test.

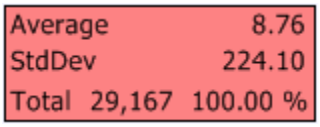
There are two types of decision trees, depending on the type of the dependent variable (*discrete* or *continuous DV*).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The nodes of a decision tree with *discrete* (*categorical*) DV show the distribution of the DV values (categories) in the form of frequencies and percentages. For example, for DV = AccountStatus (with values '*OK*' and '*Bad Debt*'), the tree nodes have the following form:



The nodes of a decision tree with *continuous* DV show the values of Average and Standard Deviation (StdDev) of that DV. For example, for DV = Profit (with values ranging from -8,600.00 to +1,700.00), the tree nodes have the following form:



If a continuous dependent variable contains singular values such as infinity or NaN, such values are treated the same way as missing values (*null*). For more information on infinities and NaN values, see Division and Infinity.

It is convenient to think of a decision tree as a set of rules, every split representing a condition. Using the menu command **Insert | Code Generation**, you can generate rules for any tree in several different languages (code types) such as SQL, SAS Language, SPSS, etc.

A detailed description of splitting algorithms used in decision trees can be found in the topic Supported Measures and in Mathematical Formulations (see Mathematical Formulations: Decision Tree)

See also:

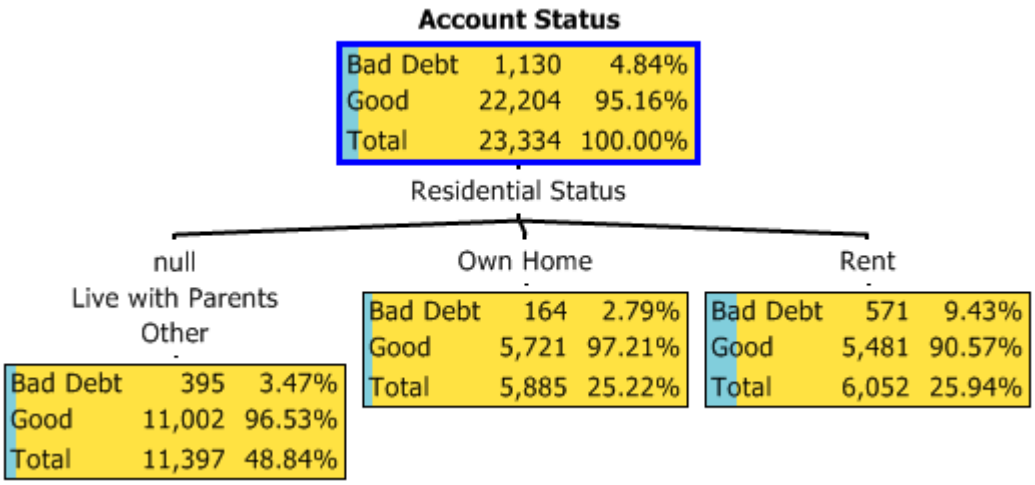
What is a Split

Inserting a Decision Tree

What is a Split

A split represents a rule. The rule determines how records from the parent node are to be distributed across the child nodes. The child nodes are the values (either ranges or categories) of a fixed independent variable (IV) which we refer to as the split variable. The rule is based on the split variable. Within the child node, the distribution of values of the variable of interest, which we refer to as the dependent variable (DV), are shown.

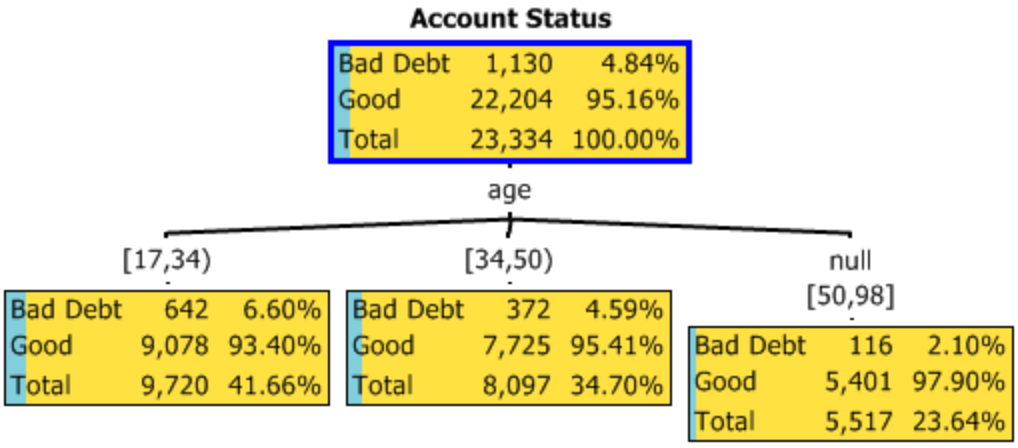
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



This example shows a parent node (which happens to be a root node) and a split. The data we are using is from the credit industry and our variable of interest, or the DV, is ***AccountStatus***.

The split variable is ***Residential Status***, and each child node represents a subset of the root node where the value of ***Residential Status*** is equal to the value (or one of the values) displayed above the node. For example, the rightmost node contains all records where *Residential Status* = 'Rent'. The condition *Residential Status* = 'Rent' is the *rule* of the rightmost node.

The split shown below is on a *continuous* independent variable, and its rules are therefore represented by ranges, or intervals, of the split variable ***age***.



In our example, there are 12119 records that satisfy the rule ***age*** = [17, 34), i.e., 17<= ***age*** <34.

See the topics in the Help section *Creating and Modifying Splits* for more details.

See also:

- Inserting a Decision Tree
- Find Split
- Split Report

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Automatic Grow
Measure
Tree View
Creating Calculated Column from Split

Inserting a Decision Tree

A decision tree node can be added to the workflow in one of the following ways:

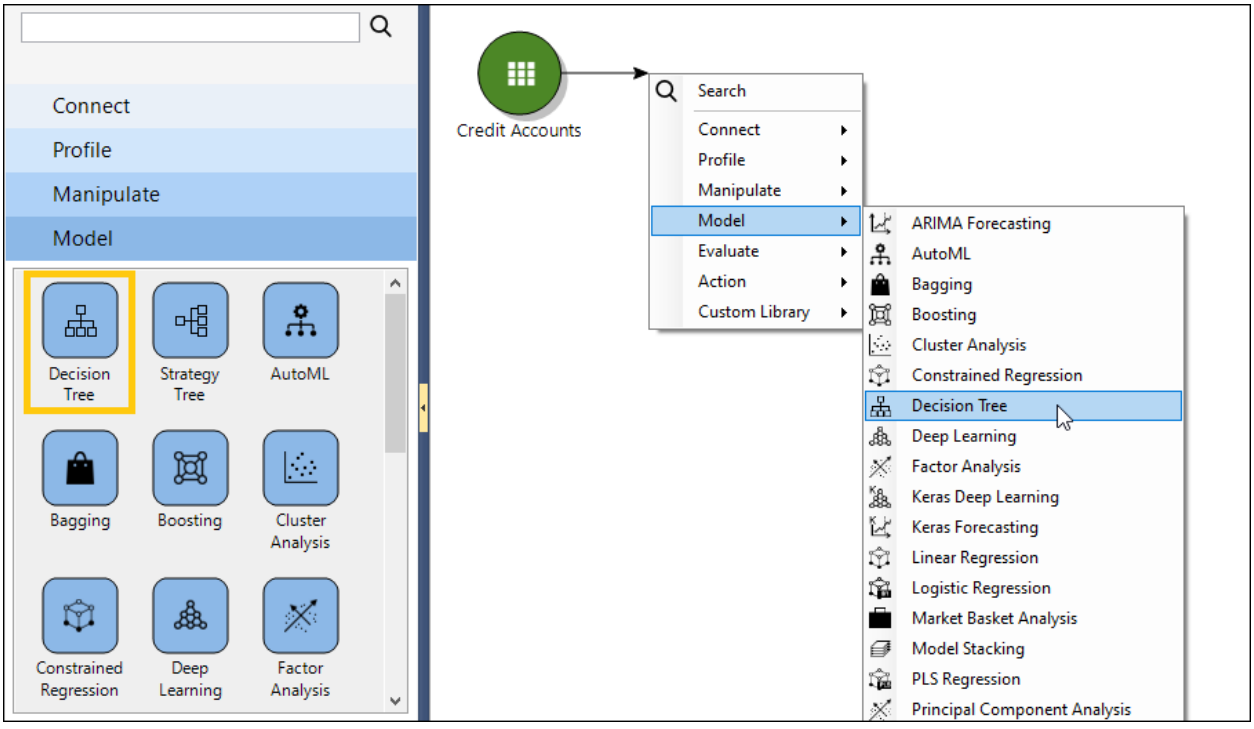
- Drag a connection arrow from a dataset node to any area on the Workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Model --> Decision Tree** as in the example screenshot below. A new **Decision Tree** process node will be created and the wizard will open.

OR

- Right-click on the Workflow canvas and select **Model --> Decision Tree** from the context menu. The new Decision Tree node is gray until its inputs are defined. Connect a dataset to the tree node.

OR

- Open the **Model** palette and drag the **Decision Tree** node onto the workflow canvas. The **Decision Tree** node is gray while its inputs are undefined. Connect the input dataset to the **Decision Tree** node by drawing a connection arrow from the dataset node to the tree node.



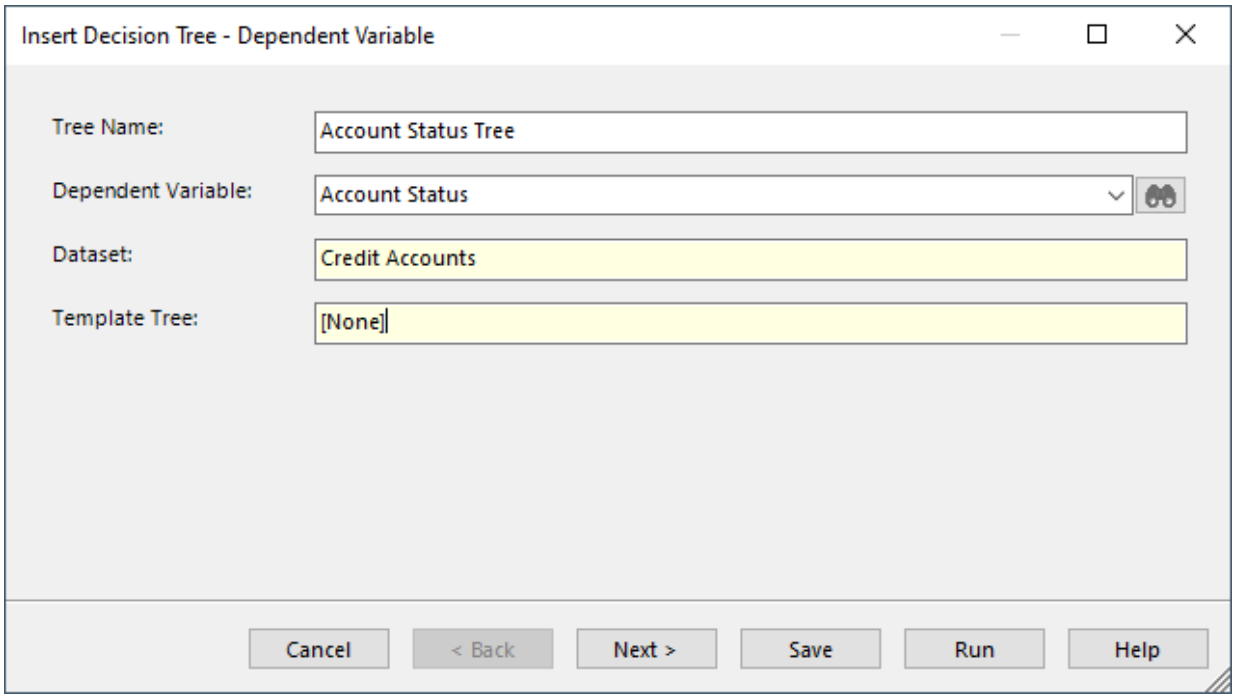
Double-click on the Decision Tree node added to the canvas, or right-click on it and select **Modify** from the context menu.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Proceed to Decision Tree: Select Dependent Variable

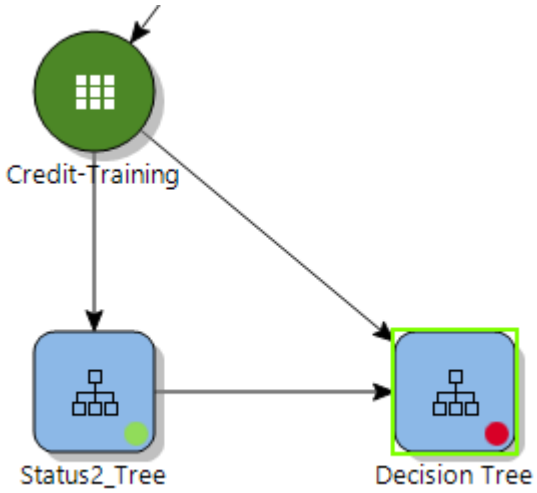
Select Dependent Variable

Select the dependent variable and specify the tree name.

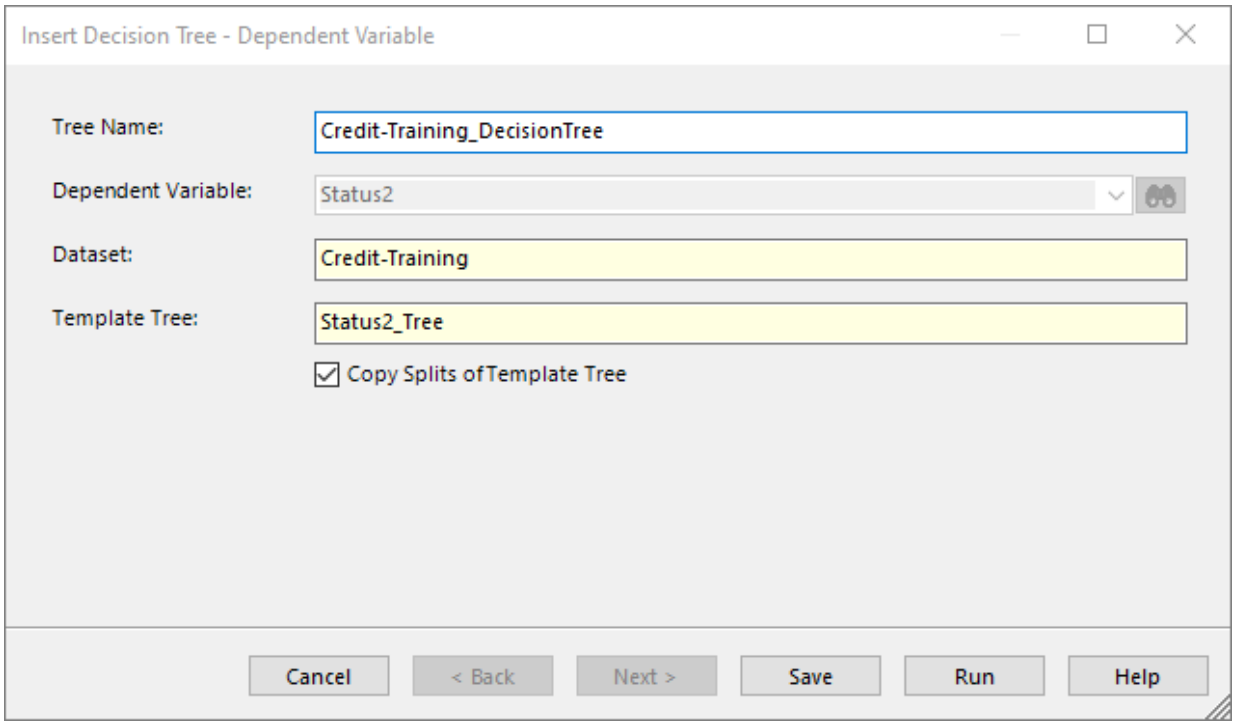


Tree Name	The name of the tree as it will appear in the project.
Dependent Variable	The dependent (target) variable for the analysis. The drop down list contains all available variables in the dataset.
Dataset	The name of the parent dataset of the tree.
Template Tree	The name of a tree to be used as a template for the purpose of copying its splits or just parameters and attributes. The template tree node can optionally be connected to the current tree node, as shown below:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If a **template tree** node is connected to the current tree node, and the dependent variable of the template tree is the same, the option *Copy Splits of Template Tree* becomes enabled. See the topic Copying a Tree for details.



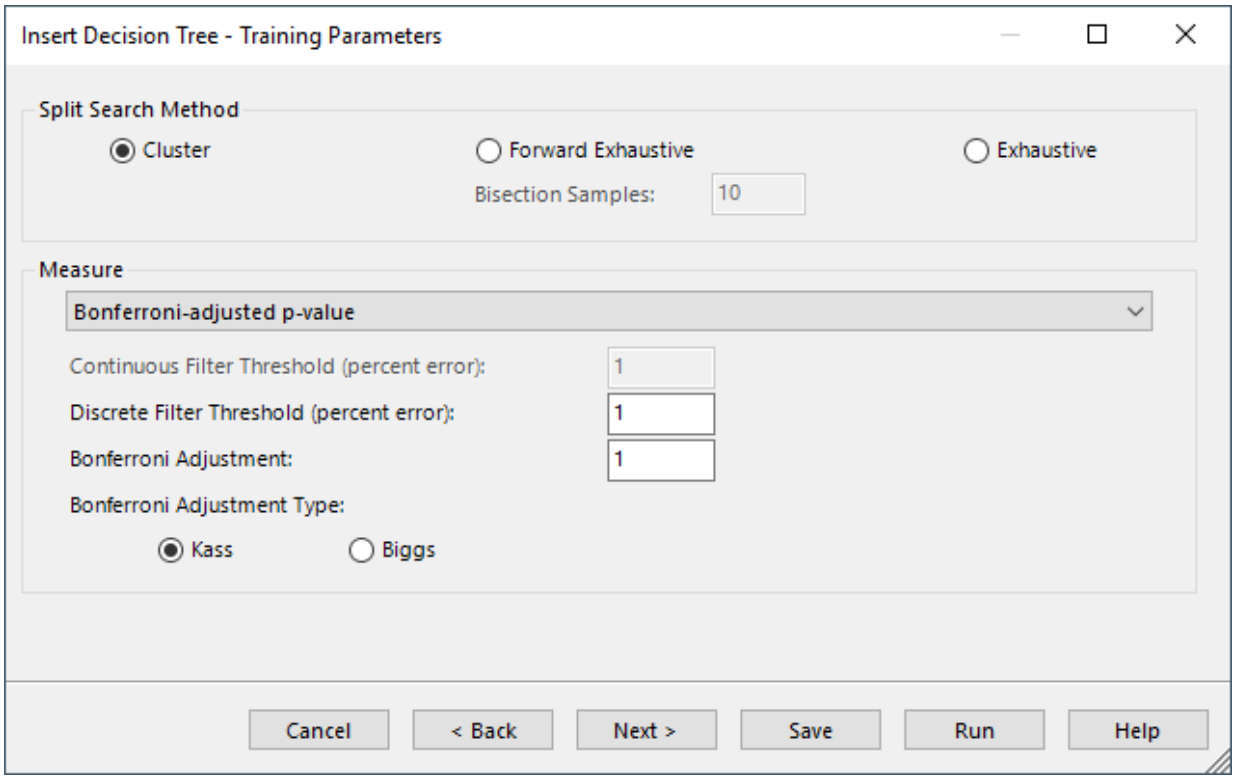
Click **Next** to specify the training parameters. If you would like accepting all other defaults and save your settings or insert the root node of the tree click **Save** or **Run**.

Proceed to the Training Parameters topic.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Training Parameters

The training parameters defined in this wizard page govern the splitting mechanism of the **Find Split** and **Automatic Grow** operations.



Split Search Method

The *Split Search Method* determines the iterative process of merging groups of values into bins until an optimal binning is achieved according to certain criteria of the split significance. This procedure locates the pairs of most similar groups and merges them together. Upon re-evaluation of the binning, this process is repeated. Merges are subject to the constraints defined by the type of independent variable (nominal, ordinal, or continuous) and whether or not the groups are in fact "similar" by comparison of the cost function to a merging threshold value. Special data structures are used to make the search for similarity between two groups as fast as possible.

Cluster	The *Cluster* method finds groups that maximize similarity within the groups and dissimilarity between the groups. Merging stops when either there is only one group left or there is no pair of groups that is "similar" enough with respect to the "merging threshold". The Cluster method tends to find more natural patterns than Exhaustive. This method is the application default. See the Find Split topic for more technical details.
Exhaustive	The *Exhaustive* method performs an exhaustive search to find groups that maximize statistical significance. Merging stops when there is only one group left. While merging, the sequence of merges is recorded in a special history buffer, noting the "best", global, solution found to date. When the merging

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

process is finished, a re-initialization to the start point is performed, and the sequence of merges recorded in the aforementioned "history buffer" is performed. This process stops as soon as the "best" history entry (i.e., the noted best solution) is encountered. Splits derived from the Exhaustive method display the most significant relationships. See the Find Split topic for more technical details.

Forward Exhaustive	This split search method is optimized for ordered variables. It begins with no split points, and adds them iteratively. It combines binary split search applied to the bins of the current (k-th) iteration and ternary splits on the bins obtained at the previous iteration (k-1). In addition, at each iteration it uses a discrete mesh of potential split points to increase the chances of achieving the global minimum p-value, rather than stopping at a local minimum. The **Bisection Samples** parameter determines the number of nodes in this mesh. The default is 10. The admissible values are nonnegative integers. Zero means no sampling (i.e., a complete sweep is performed). The higher the value, the more likely the method is to find the optimal global solution – at the expense of performance. These optimizations affect only ordered variables – the variables whose Usage is 'ordinal' or 'continuous'. For splits on nominal variables, the regular Exhaustive method is used.

See the Find Split topic for more technical details on the splitting process.

Selecting the Measure

The measure determines the method of evaluating the significance, or importance, of a split. The best split, or the most significant split, is the first split produced by a **Find Split** command. Choose the measure for your decision tree model. (The default is Entropy Variance, unless changed in Preferences.). The available measures are:

Entropy is the measure used to determine the importance of each split in a tree. Four statistics are calculated for this measure, ***input entropy***, ***output entropy***, ***ratio entropy*** and ***info***. See Mathematical Formulations for detailed formulas for these values.

For decision trees, entropy measures the variability of the distribution of the dependent variable (DV) values with respect to the independent variables (IV). The value of entropy is a number in the interval [0,1]. An entropy of one indicates high variability in the distribution and an entropy approaching zero indicates low variability. As the variability is reduced, there is a gain in information.

In general, the output entropy value will decrease (approach zero) as the importance of the split increases. The value of info shows the importance of a split expressed in a percentage. The higher the percentage, the more important the split.

Entropy Variance

This measure uses statistics based on entropy or variance to determine the importance of each split in a tree. The statistics calculated for this measure are:

- for discrete dependent variables:
  - ***information value*** (applicable to binary DV only)
  - ***input entropy***
  - ***output entropy***

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

    - ***ratio entropy***
    - ***info*** (ratio entropy expressed as a percentage)
- for continuous dependent variables:
    - ***input variance***
    - ***output variance***
    - ***ratio variance***
    - ***info*** (ratio variance expressed as a percentage)

For decision trees, entropy measures the variability of the distribution of the dependent variable (DV) values with respect to the independent variables (IV). The value of entropy is a number in the interval [0,1]. An entropy of one indicates high variability in the distribution and an entropy approaching zero indicates low variability. As the variability is reduced, there is a gain in information.

In general, the output entropy value will decrease (approach zero) as the importance of the split increases. The value of info shows the importance of a split expressed as a percentage. The higher the percentage, the more important the split.

See Decision Tree in Mathematical Formulations for detailed equations for these values.

Gini Variance

This measure uses the Gini coefficient, or Gini index, to determine the importance of each split in a tree. The statistics calculated for this measure are:

- for discrete dependent variables:
    - ***information value*** (applicable to binary DV only)
    - ***input index***
    - ***output index***
    - ***ratio index***
- for continuous dependent variables:
    - ***input variance***
    - ***output variance***
    - ***ratio variance***

Info is *ratio index* or *ratio variance* expressed as a percentage.

For decision trees, the Gini index compares the equality of the distributions of the dependent variable (DV) values with respect to the independent variables (IV). The value of the Gini index is a number in the interval [0,1]. A Gini index approaching 0 indicates that the distributions are equal, a Gini index approaching 1 indicates that the distributions are not equal.

See Decision Tree in Mathematical Formulations for detailed equations for these values.

Unadjusted p-value

In trees based on the *Unadjusted p-value* measure, the significance of splits is ranked using the p-value. The statistics calculated for this measure are:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- for discrete dependent variable:
    - ***information value*** (applicable to binary DV only)
    - ***p-value***
    - ***chi-square***
    - ***df*** (degrees of freedom)
    - ***confidence*** (1 - *p*)
- for continuous dependent variable:
    - ***p-value***
    - ***confidence*** (1 - *p*)
    - ***F-statistic***
    - ***df numerator***
    - ***df denominator***

Info is confidence expressed as a percentage: (1 - *p*) * 100%

p-value expresses the probability that the difference being observed occurs by chance. The higher the p-value the more likely the observed differences in the DV distribution are random.

chi-square: Chi-Squared Test uses the chi-square statistic measuring the statistical association between two categorical variables.

df – Degrees of freedom: The number of observations minus the number of necessary relations, which is obtainable from these observations (df = *n* - *r*). The definition of "necessary relationship" is simply defined as the relationship between a dependent variable (Y) and each independent variable (X) in the research.

The F-ratio statistic is calculated as part of the ANOVA procedure. The larger this number is, the greater the distance between the means or average values of the nodes in the split. ANOVA Procedure - (ANalysis Of VAriance). A procedure used to detect statistically significant effects induced by an independent variable on a continuous dependent variable. The ANOVA procedure employs an F-test to measure the differences between a given set of population means, where F = Mean Square for Treatments divided by the Mean Square for Error (F = MST/MSE).

See Decision Tree in Mathematical Formulations for detailed equations for these values.

Unadjusted log p-value

In trees based on the *Unadjusted log p-value* measure, the significance of splits is ranked using the logarithm of p-value. The statistics calculated for this measure are:

- for discrete dependent variable:
    - ***information value*** (applicable to binary DV only)
    - ***p-value***
    - ***log p-value*** (the natural logarithm of p-value)
    - ***chi-square***
    - ***df*** (degrees of freedom)
    - ***confidence*** (1 - *p*)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- for continuous dependent variable:
    - ***p-value***
    - ***log p-value*** (the natural logarithm of p-value)
    - ***confidence*** (1 - *p*)
    - ***F-statistic***
    - ***df numerator***
    - ***df denominator***

The split significance is ranked by ***-log(p)*** (negative logarithm of p-value).

The F-Ratio Statistic is calculated as part of the ANOVA procedure. The larger this number is, the greater the distance between the means or average values of the nodes in the split. ANOVA Procedure - (ANalysis Of VAriance). A procedure used to detect statistically significant effects induced by an independent variable on a continuous dependent variable. The ANOVA procedure employs an F-test to measure the differences between a given set of population means, where F = Mean Square for Treatments divided by the Mean Square for Error (F = MST/MSE).

The p-value is the probability that the difference being observed occurs by chance. The higher the p-value the more likely the observed differences in the DV distribution are random.

See Decision Tree in Mathematical Formulations for detailed equations for these values.

Bonferroni-adjusted p-value

In trees based on the *Bonferroni-adjusted p-value* measure, the significance of splits is ranked using the p-value with Bonferroni adjustment. The statistics calculated for this measure are:

- for discrete dependent variable:
    - ***information value*** (applicable to binary DV only)
    - ***p-value***
    - ***adjusted p-value*** (p-value with Bonferroni adjustment)
    - ***the Bonferroni factor***
    - ***chi-square***
    - ***df*** (degrees of freedom)
    - ***confidence*** (1 - *p*)
    - ***adjusted confidence*** (1 - adjusted *p*)
- for continuous dependent variable:
    - ***p-value***
    - ***adjusted p-value*** (p-value with Bonferroni adjustment)
    - ***the Bonferroni factor***
    - ***confidence*** (1 - *p*)
    - ***adjusted confidence*** (1 - adjusted *p*)
    - ***F-statistic***
    - ***df numerator***

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- ***df denominator***

Info is adjusted confidence expressed as a percentage: (1 - adjusted *p*) * 100%

The Bonferroni adjustment (B * *p*) uses multiple test comparisons and adjusts the significance level (p-value) accordingly. The p-value used as the cutoff is divided by the total number of comparisons to be made. Any single comparison would then be compared to this adjusted p-value. For example, for the cutoff, p<0.05 and using 10 comparisons, any single test statistic would be compared to p<0.005.

The Bonferroni adjustment factor for the p-value is used to minimize the discovery of chance relationships in a dataset. Bonferroni adjustments automatically adjust the level of significance used when testing the validity of a relationship.

See Decision Tree in Mathematical Formulations for detailed equations for these values.

Bonferroni-adjusted log p-value

In trees based on the *Bonferroni-adjusted log p-value* measure, the significance of splits is ranked using the logarithm of the adjusted p-value. The statistics calculated for this measure are:

- for discrete dependent variable:
    - ***information value*** (applicable to binary DV only)
    - ***p-value***
    - ***log p-value*** (the natural logarithm of p-value)
    - ***adjusted p-value*** (p-value with Bonferroni adjustment)
    - ***log adj p-value*** (the natural logarithm of the Bonferroni-adjusted p-value)
    - ***the Bonferroni factor***
    - ***chi-square***
    - ***df*** (degrees of freedom)
    - ***confidence*** (1 - *p*)
    - ***adjusted confidence*** (1 - adjusted *p*)
- for continuous dependent variable:
    - ***p-value***
    - ***log p-value*** (the natural logarithm of p-value)
    - ***adjusted p-value*** (p-value with Bonferroni adjustment)
    - ***log adj p-value*** (the natural logarithm of the Bonferroni-adjusted p-value)
    - ***the Bonferroni factor***
    - ***confidence*** (1 - *p*)
    - ***adjusted confidence*** (1 - adjusted *p*)
    - ***F-statistic***
    - ***df numerator***
    - ***df denominator***

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The split significance is ranked by ***-log(adj p)*** (negative logarithm of the Bonferroni-adjusted p-value).

The Bonferroni adjustment (B * *p*) uses multiple test comparisons and adjusts the significance level (p-value) accordingly. The p-value used as the cutoff is divided by the total number of comparisons to be made. Any single comparison would then be compared to this adjusted p-value. For example, for the cutoff, p<0.05 and using 10 comparisons, any single test statistic would be compared to p<0.005.

The Bonferroni adjustment factor for the p-value is used to minimize the discovery of chance relationships in a dataset. Bonferroni adjustments automatically adjust the level of significance used when testing the validity of a relationship.

See Decision Tree in Mathematical Formulations for detailed equations for these values.

See Supported Measures for more details.

Filter Thresholds

Continuous/Discrete Filter Threshold (percent error)

When analyzing data using the unadjusted or adjusted p-value measure, you run the risk that your data may display results obtained through chance, which can be potentially misleading. By adjusting the filter threshold value, you can lower the possibility that chance will play a part in your analysis - you determine how rigorously your data will be analyzed.

Continuous and discrete filter thresholds are for trees with continuous and discrete dependent variable, respectively. These thresholds are related to the selected measure as follows:

a) They are only applicable to the Bonferroni-adjusted measures.

b) The thresholds are based on the confidence level of a split which is described in the section “Mathematical Formulations" (Decision Trees). The *Info* parameter in Split Report (same as *Rank* in the Go To Split dialog) is the value of ***confidence*** (1 - *p*) or (*1* - *adjusted p*) expressed as a percentage. For example, if *confidence*=0.95, then Info = Rank = 95%.

> **Note:** If you want the *confidence* value to be displayed at every split of the current tree (for trees with a p-value measure), select **Tools | Options** from the main menu, go to the Tree Split Parameters tab, and select the *confidence* checkbox.

By specifying the filter threshold, you define the minimum value of Info (or Rank) that is of interest to you, so that Find Split or AutoGrow would not show the splits on independent variables whose rank is lower than 100% minus the specified error.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

For example, if you are interested only in the independent variables whose Rank is higher than 98%, specify 2 as the filter threshold (percent error). In this case, variables with Rank less than 98% (= 100% - 2%) will not appear in the Go To Split dialog and no statistics will be displayed for them in Split Report.

Bonferroni Adjustment

Bonferroni Adjustment

This option is applicable only to the Bonferroni-adjusted measures.

To minimize the discovery of chance relationships, Bonferroni adjustments let you adjust the level of significance used when testing the validity of a relationship.

This type of adjustment compensates for amount of redundant variables in your dataset. Normally, if a concept is captured by more than one variable, it has a disproportionate opportunity to enter the analysis by chance. Use the Bonferroni adjustment to compensate for this possibility.

Bonferroni adjustments may be quite conservative. Increasing the size of the adjustment increases the level of conservatism. It is not meaningful to set this value greater than the total number of variables in your dataset.

Bonferroni Adjustment Type (Kass / Biggs)

Two options are available for the Bonferroni adjustment type: **Kass** and **Biggs** (named after the authors of these methods, G.V. Kass and D. Biggs). The Kass type is the default. These options influence the number of child nodes in splits, and their effect depends on the type (nominal, ordinal, or continuous) and the cardinality of the split variable. For example, according to G. Ritschard, the Kass adjustment type tends to penalize high-cardinality predictors more than low-cardinality ones, and nominal predictors more than ordinal or continuous predictors. For details, see the articles by these authors listed in the References section of Mathematical Formulations.

Thus, for instance, if you increase the *Default number of bins* (the initial number of bins for the Find Split operation) in Attrubute Editor before growing the tree with Find Split or Automatic Grow, the Kass adjustment type may be more preferable as the Find Split algorithm will be less likely to pick variables with lower relevance to the target. The Biggs type may be preferable in other contexts. For guidance, see the articles cited above.

See also:

Find Split

Automatic Grow

Split Report

Mathematical Formulations: Decision Tree

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Measure

The measure determines the method of evaluating the significance, or importance, of a split. The best split, or the most significant split, is the one that is displayed after a Find Split command. Each method uses different statistics. The values of these statistics are displayed in the Split Report. A brief description of the available measures is given below. A more detailed description is available in the Decision Tree section of Mathematical Formulations.

Entropy Variance

This measure uses statistics based on entropy or variance to determine the importance of each split in a tree. The statistics calculated for this measure are:

- for discrete dependent variables:
    - ***information value*** (only applicable to binary dependent variable; see Mathematical Formulations: Decision Tree for details)
    - ***input entropy***
    - ***output entropy***
    - ***ratio entropy***
    - ***info*** (ratio entropy expressed as a percentage)
- for continuous dependent variables:
    - ***input variance***
    - ***output variance***
    - ***ratio variance***
    - ***info*** (ratio variance expressed as a percentage)

For decision trees, entropy measures the variability of the distribution of the dependent variable (DV) values with respect to the independent variables (IV). The value of entropy is a number in the interval [0,1]. An entropy of one indicates high variability in the distribution and an entropy approaching zero indicates low variability. As the variability is reduced, there is a gain in information.

In general, the output entropy value will decrease (approach zero) as the importance of the split increases. The value of info shows the importance of a split expressed as a percentage. The higher the percentage, the more important the split.

See Decision Tree in Mathematical Formulations for detailed equations for these values.

Gini Variance

This measure uses the Gini coefficient, or Gini index, to determine the importance of each split in a tree. The statistics calculated for this measure are:

- for discrete dependent variables:
    - ***information value*** (only applicable to binary dependent variable)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

    - ***input index***
    - ***output index***
    - ***ratio index***
- for continuous dependent variables:
    - ***input variance***
    - ***output variance***
    - ***ratio variance***

Info is *ratio index* or *ratio variance* expressed as a percentage.

For decision trees, the Gini index compares the equality of the distributions of the dependent variable (DV) values with respect to the independent variables (IV). The value of the Gini index is a number in the interval [0,1]. A Gini index approaching 0 indicates that the distributions are equal, a Gini index approaching 1 indicates that the distributions are not equal.

See Decision Tree in Mathematical Formulations for detailed equations for these values.

Unadjusted p-value

In trees based on the *Unadjusted p-value* measure, the significance of splits is ranked using the p-value. The statistics calculated for this measure are:

- for discrete dependent variable:
    - ***information value*** (only applicable to binary dependent variable)
    - ***p-value***
    - ***chi-square***
    - ***df*** (degrees of freedom)
    - ***confidence*** (1 - *p*)
- for continuous dependent variable:
    - ***p-value***
    - ***confidence*** (1 - *p*)
    - ***F-statistic***
    - ***df numerator***
    - ***df denominator***

Info is confidence expressed as a percentage: (1 - *p*) * 100%

p-value expresses the probability that the difference being observed occurs by chance. The higher the p-value the more likely the observed differences in the DV distribution are random.

chi-square: Chi-Squared Test uses the chi-square statistic measuring the statistical association between two categorical variables.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

df – Degrees of freedom: The number of observations minus the number of necessary relations, which is obtainable from these observations (df = *n* - *r*). The definition of "necessary relationship" is simply defined as the relationship between a dependent variable (Y) and each independent variable (X) in the research.

The F-ratio statistic is calculated as part of the ANOVA procedure. The larger this number is, the greater the distance between the means or average values of the nodes in the split. ANOVA Procedure - (ANalysis Of VAriance). A procedure used to detect statistically significant effects induced by an independent variable on a continuous dependent variable. The ANOVA procedure employs an F-test to measure the differences between a given set of population means, where F = Mean Square for Treatments divided by the Mean Square for Error (F = MST/MSE).

See Decision Tree in Mathematical Formulations for detailed equations for these values.

Unadjusted log p-value

In trees based on the *Unadjusted log p-value* measure, the significance of splits is ranked using the logarithm of p-value. The statistics calculated for this measure are:

- for discrete dependent variable:
    - ***information value*** (only applicable to binary dependent variable)
    - ***p-value***
    - ***log p-value*** (the natural logarithm of p-value)
    - ***chi-square***
    - ***df*** (degrees of freedom)
    - ***confidence*** (1 - *p*)
- for continuous dependent variable:
    - ***p-value***
    - ***log p-value*** (the natural logarithm of p-value)
    - ***confidence*** (1 - *p*)
    - ***F-statistic***
    - ***df numerator***
    - ***df denominator***

The split significance is ranked by ***-log(p)*** (negative logarithm of p-value).

The F-Ratio Statistic is calculated as part of the ANOVA procedure. The larger this number is, the greater the distance between the means or average values of the nodes in the split. ANOVA Procedure - (ANalysis Of VAriance). A procedure used to detect statistically significant effects induced by an independent variable on a continuous dependent variable. The ANOVA procedure employs an F-test to measure the differences between a given set of population means, where F = Mean Square for Treatments divided by the Mean Square for Error (F = MST/MSE).

The p-value is the probability that the difference being observed occurs by chance. The higher the p-value the more likely the observed differences in the DV distribution are random.

See Decision Tree in Mathematical Formulations for detailed equations for these values.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Bonferroni-adjusted p-value

In trees based on the *Bonferroni-adjusted p-value* measure, the significance of splits is ranked using the p-value with Bonferroni adjustment. The statistics calculated for this measure are:

- for discrete dependent variable:
    - ***information value*** (only applicable to binary dependent variable)
    - ***p-value***
    - ***adjusted p-value*** (p-value with Bonferroni adjustment)
    - ***the Bonferroni factor***
    - ***chi-square***
    - ***df*** (degrees of freedom)
    - ***confidence*** (1 - *p*)
    - ***adjusted confidence*** (1 - adjusted *p*)
- for continuous dependent variable:
    - ***p-value***
    - ***adjusted p-value*** (p-value with Bonferroni adjustment)
    - ***the Bonferroni factor***
    - ***confidence*** (1 - *p*)
    - ***adjusted confidence*** (1 - adjusted *p*)
    - ***F-statistic***
    - ***df numerator***
    - ***df denominator***

Info is adjusted confidence expressed as a percentage: (1 - adjusted *p*) * 100%

The Bonferroni adjustment (B * *p*) uses multiple test comparisons and adjusts the significance level (p-value) accordingly. The p-value used as the cutoff is divided by the total number of comparisons to be made. Any single comparison would then be compared to this adjusted p-value. For example, for the cutoff, $p<0.05$ and using 10 comparisons, any single test statistic would be compared to $p<0.005$.

The Bonferroni adjustment factor for the p-value is used to minimize the discovery of chance relationships in a dataset. Bonferroni adjustments automatically adjust the level of significance used when testing the validity of a relationship.

See Decision Tree in Mathematical Formulations for detailed equations for these values.

Bonferroni-adjusted log p-value

In trees based on the *Bonferroni-adjusted log p-value* measure, the significance of splits is ranked using the logarithm of the adjusted p-value. The statistics calculated for this measure are:

- for discrete dependent variable:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- ***information value*** (only applicable to binary dependent variable)
- ***p-value***
- ***log p-value*** (the natural logarithm of p-value)
- ***adjusted p-value*** (p-value with Bonferroni adjustment)
- ***log adj p-value*** (the natural logarithm of the Bonferroni-adjusted p-value)
- ***the Bonferroni factor***
- ***chi-square***
- ***df*** (degrees of freedom)
- ***confidence*** (1 - *p*)
- ***adjusted confidence*** (1 - adjusted *p*)

- for continuous dependent variable:
  - ***p-value***
  - ***log p-value*** (the natural logarithm of p-value)
  - ***adjusted p-value*** (p-value with Bonferroni adjustment)
  - ***log adj p-value*** (the natural logarithm of the Bonferroni-adjusted p-value)
  - ***the Bonferroni factor***
  - ***confidence*** (1 - *p*)
  - ***adjusted confidence*** (1 - adjusted *p*)
  - ***F-statistic***
  - ***df numerator***
  - ***df denominator***

The split significance is ranked by ***-log(adj p)*** (negative logarithm of the Bonferroni-adjusted p-value).

See Decision Tree in Mathematical Formulations for detailed equations for these values.

Split Report

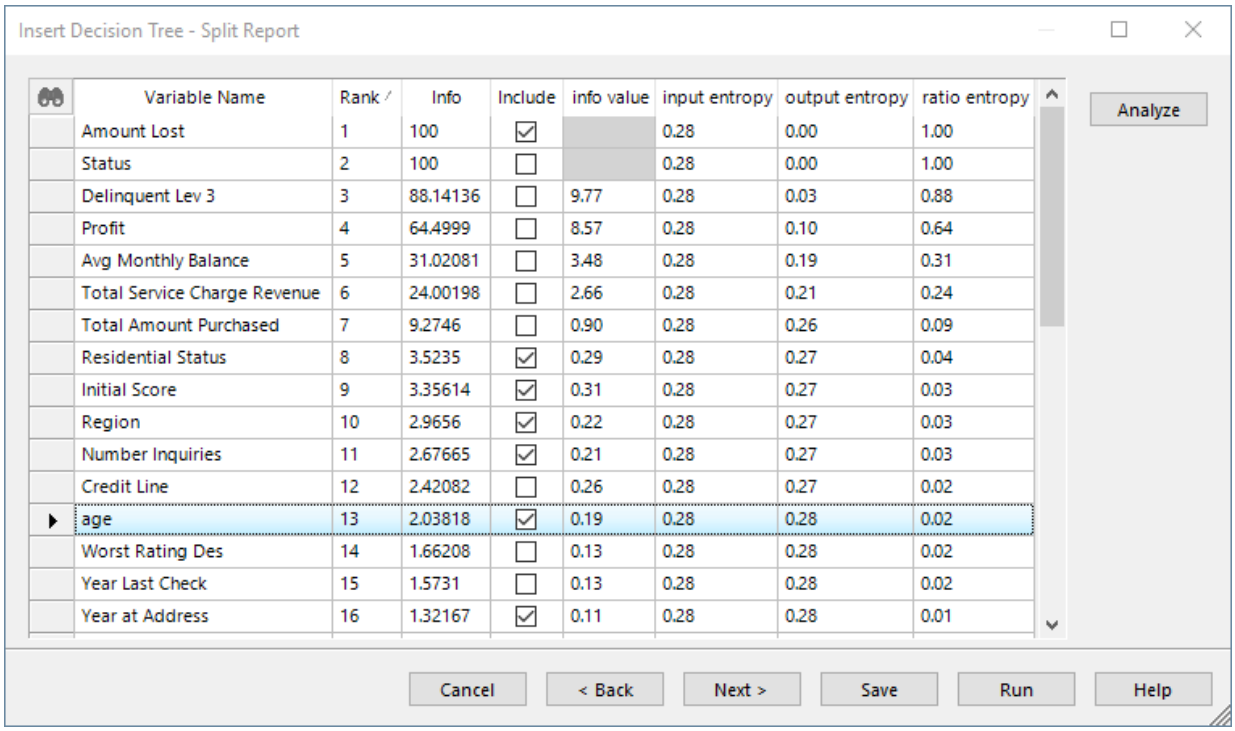
This step allows you to choose the independent variables to use in your model. By default, all variables are included, except those below the significance threshold (if applicable). To exclude a variable, deselect the corresponding box in the *Include* column. The columns are sortable, so you can sot by the desired statistic.

If desired, click the **Analyze** button to calculate and display the split statistics for each variable. This will rank the variables in order of their importance, or significance, with respect to the dependent variable. If a variable importance is below the minimum level, its ***Rank*** and ***Info*** values remain blank.

The split statistics calculated by the Analyze button depend on the measure sepected at the previous step of the wizard. For more information on the split statistics displayed in this report, see the topic Measure. The split report statistics can help you decide which variables to include in your

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

model. In addition to selecting significant variables, you may want to exclude variables that are obviously highly correlated with the dependent variable.

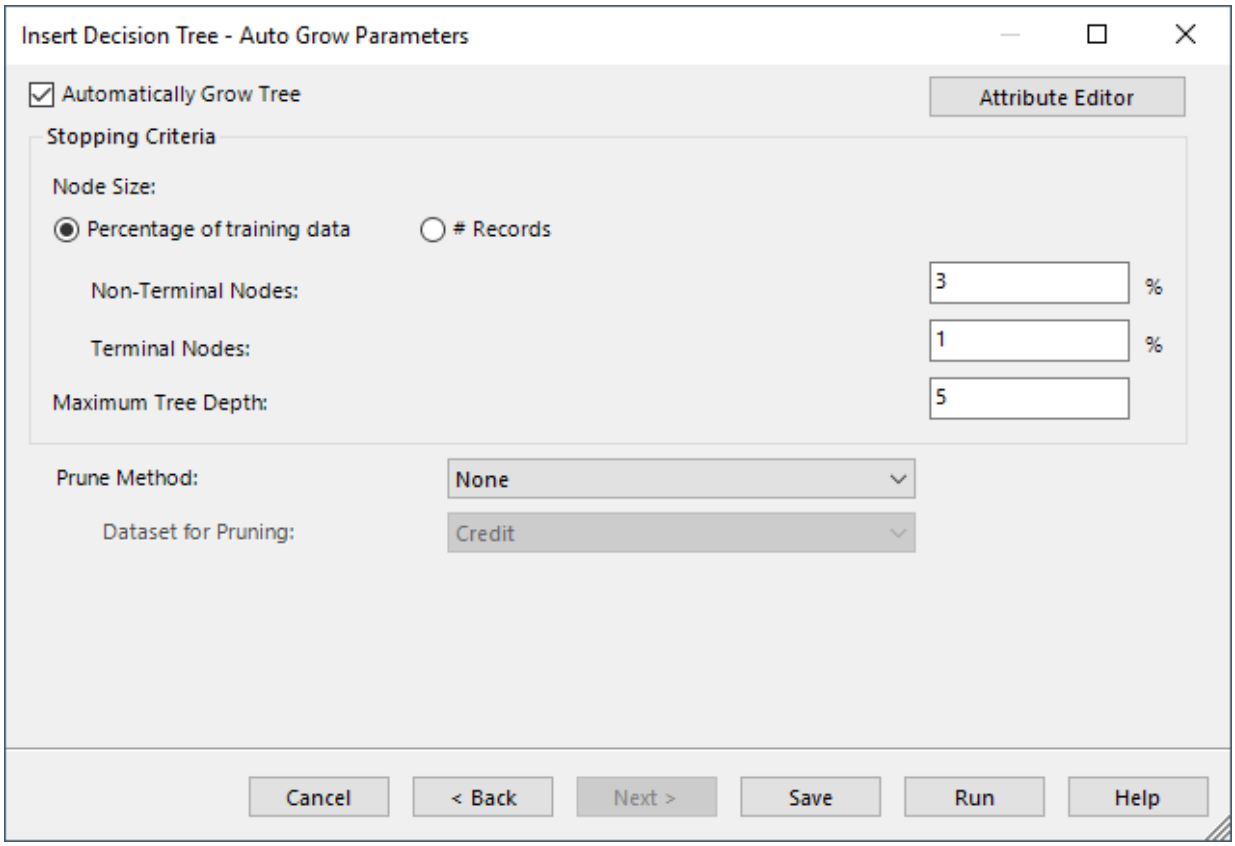


Variable Name	Rank	Info	Include	info value	input entropy	output entropy	ratio entropy
Amount Lost	1	100	☑		0.28	0.00	1.00
Status	2	100	☐		0.28	0.00	1.00
Delinquent Lev 3	3	88.14136	☐	9.77	0.28	0.03	0.88
Profit	4	64.4999	☐	8.57	0.28	0.10	0.64
Avg Monthly Balance	5	31.02081	☐	3.48	0.28	0.19	0.31
Total Service Charge Revenue	6	24.00198	☐	2.66	0.28	0.21	0.24
Total Amount Purchased	7	9.2746	☐	0.90	0.28	0.26	0.09
Residential Status	8	3.5235	☑	0.29	0.28	0.27	0.04
Initial Score	9	3.35614	☑	0.31	0.28	0.27	0.03
Region	10	2.9656	☑	0.22	0.28	0.27	0.03
Number Inquiries	11	2.67665	☑	0.21	0.28	0.27	0.03
Credit Line	12	2.42082	☐	0.26	0.28	0.27	0.02
age	13	2.03818	☑	0.19	0.28	0.28	0.02
Worst Rating Des	14	1.66208	☐	0.13	0.28	0.28	0.02
Year Last Check	15	1.5731	☐	0.13	0.28	0.28	0.02
Year at Address	16	1.32167	☑	0.11	0.28	0.28	0.01

Auto Grow Parameters

This step of the Decision Tree Wizard allows you to specify the Auto Grow parameters if you want the tree to be grown automatically upon insertion of the root node. You can also modify the attributes of the tree variables in the Tree Attribute Editor at this step using the **Advanced** button.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The box “Automatically Grow Tree” is unchecked by default (thus, by default, only the root node is created by the wizard). Check this box if you want the tree to be grown automatically upon insertion of the root node.

Node Size Limits in Percentage or Number of Records	If the **Percentage** option is selected, the node size limits must be specified in percentage of total. This option is selected by default and is recommended, since it is logical to set these limits relative to the size of the input dataset. Select the option **# Records** if you want to specify the node size limits in absolute numbers of records.
Minimum Node Size for Non-Terminal Nodes	The minimum number of records for a node to contain before the **Find Split** or **Automatic Grow** operations are allowed to split it. For the option "*Percentage of training data*", the default is 3%.
Minimum Node Size for Terminal Nodes	The minimum number of records allowed for terminal nodes. For the option "*Percentage of training data*", the default is 1%.
Maximum Tree Depth for Automatic Grow	The maximum number of levels allowed for the **Automatic Grow** command to grow the decision tree. The number of levels includes the root node, so the depth of a tree with a single split is 2. Enter the value of zero (0) for unlimited depth (in that case, the depth

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

is limited only by the minimum node size constraints defined in this dialog). The default is 5.

Pruning Options

In order to use the pruning feature, you must have at least one more dataset available in the project aside from the parent dataset of the new tree. Normally, you should first insert partitions of the original dataset and then insert your tree based on the first partition and specify the second partition as the dataset for pruning. The following pruning algorithms are available:

- Critical-Value
- Error-Complexity
- Reduced-Error

See the Decision Tree section of Mathematical Formulations for a detailed description of these pruning methods.

See also: Supported Measures

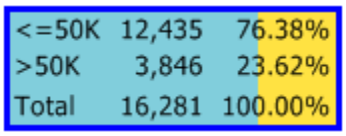
Creating and Modifying Splits

Find Split

The **Find Split** operation searches for the independent variables (IV) that are most significant to the dependent variable (DV) based on criteria defined by the measure. The independent variables that are found to be significant are called the split variable(s). The most significant independent variable is displayed with child nodes representing a category or range found in the independent variable. See more technical description of the splitting algorithm below.

The **Find Split** command may find multiple significant splits. The splits are ranked according to the measure and the first and most significant split is displayed in the tree view. The other splits can be viewed using Next Split, Previous Split, or Go To Split commands available through the right-click context menu or under the Reshape menu.

To do a **Find Split**, click on the node you want to split. The node becomes selected, which is indicated by a blue outline along the border.



From the menu:

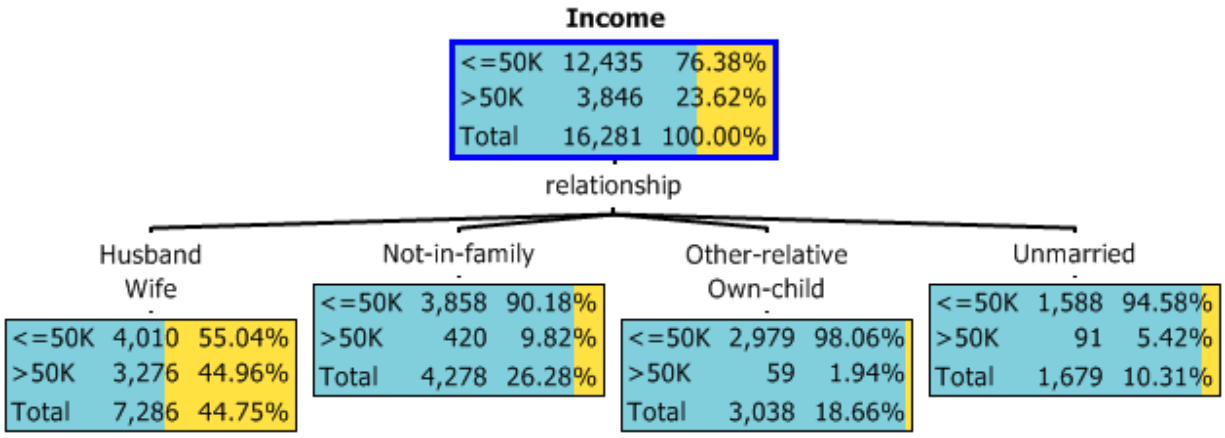
Grow | Find Split

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

From the context menu:

right-click on the node and select Find Split

For example, applying the Find Split operation to the node above may yield the following split:



For advanced options that define the characteristics of branches produced by Find Split, see the topic Using the Attribute Editor.

NOTE: You can undo any sequence of Find Split operations using the Undo command under the Edit menu, or pressing Ctrl Z on your keyboard.

Technical Details of the Splitting Algorithm

The underlying algorithm of the Find Split operation is as follows. (Note that it also governs the Automatic Grow operation, since Automatic Grow is just a recursive application of Find Split.)

The Knowledge Seeker tree growth algorithm (KSTGA) accessed via the **Find Split** or **Automatic Grow** functions attempts to solve a constrained combinatorial optimization problem. Given *N* input categories of a predictor, where each containing a measurement of the predictable, a cost function appropriate to the nature of the predictable (discrete vs. continuous), the algorithm finds *M* proper subsets of the *N* categories such that

(a) the subsets meet constraints imposed by the nature of the predictor (ordered, unordered, floating) as well as special properties of the solution (each output category must meet minimum 'size' requirements), and

(b) the cost function is minimized.

As an example, if *N* = 5 and *M* = 2, and we just use numbers for the categories (they can be any value -- strings, integers, or other types) for notational simplicity, then:

in = 1

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

2
3
4
5

are the input categories, and

out = 1 4 5
2 3

might be the solution. The "1 4 5" (and "2 3") can be called a "group", and the act of making a group from smaller groups is called "merging".

Each input category of the predictor, or IV, has some information about the predictable, or "DV". For discrete DV's, this information is the discrete distribution of the DV for that value of the IV. For a continuous DV, this would be the mean and variance of the DV for that value of the IV.

The cost function measures the statistical similarity of the DV for two or more groups of categories. For example, with a discrete DV, a chi-square test is used, and for a continuous DV, a simple analysis of variance is conducted. The larger the cost function value is, the more "similar" the groups are with respect to the DV.

The problem is one of finding a sequence of merges that produce groups that give the smallest value of the associated cost function -- that is to say, when the distribution of the DV in each group is *most different*, thus "best explains" the DV.

The search for the merges is conducted with respect to the class of the IV. If the IV is declared to be an 'ordered' variable, then merges are only conducted between "adjacent" groups after they have been put into "order". For the 'unordered' case, any set of groups can be merged without regard to 'order'.

The 'floating' case is a simple variation on the 'ordered': any missing value is removed from the input categories and held to the side. The problem is then considered to be 'ordered' and run normally. The last step of the algorithm will then take the missing category and search for the group which it "best fits" and place it there.

In addition, there are algorithm parameters that specify when a merge is actually performed, minimum 'group' size (as defined by the number of cases in a group, as opposed to the number of IV values), a merging threshold (below which groups are not merged), a break-apart threshold (above which groups are not dissociated), and whether or not an "exhaustive" search is conducted.

The basic structure of the algorithm is then:

- initialize-start-point

- merge-categories

if(exhaustive)

- initialize-start-point

- do-restore-exhaustive

else if(break_apart)

- do-break-apart

- handle-small-nodes

if(floating)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- handle-floating

Description of the steps:

INITIALIZE-START-POINT

The start point is N groups, each one consisting of a single input IV value. For the 'floating' constraint, at this point,

MERGE-CATEGORIES

The merge-phase locates the pairs of most-similar groups and merges them together. This process is then repeated. As noted above, merges are subject to the constraints defined by the type of IV (ordered, unordered) and whether or not the groups are in fact "similar" by comparison of the cost function to a merging threshold value.

For the non-exhaustive case (cluster):

Merging stops when either

a) there is only one group left,

OR

b) there is no pair of groups that is "similar" enough with respect to the "merging threshold".

For the exhaustive case:

Merging stops when

a) there is only one group left (i.e., the merging threshold is ignored)

AND

b) while merging, record the sequence of merges in a special history buffer, noting the "best", global, solution found to date.

Special data structures are used to make the search for similarity between two groups as fast as possible.

DO-RESTORE-EXHAUSTIVE

If the exhaustive analysis is selected, then after merging, a re-initialization to the start point is performed, and the sequence of merges recorded in the aforementioned "history buffer" is performed. This process stops as soon as the "best" history entry (i.e., the best solution) is encountered.

DO-BREAK-APART

During the merge phase, it is possible that groups were merged when, from the standpoint of the global solution, they should not have been merged. The "break apart" mechanism attempts to partially correct this tendency to be a bit greedy.

It works by examining each group, and chopping it into smaller groups (again, subject to the usual constraints), examining the broken group against a break-apart threshold (it won't break it apart if the broken pieces of the group are themselves still similar with respect to the DV),

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

and re-testing the solution for optimality (i.e., did the break-up actually make the solution better).

HANDLE-SMALL-NODES

Any group that has too few cases (or samples, or records) is merged with the group that is most similar to it with respect to the DV.

HANDLE-FLOATING

If the IV is 'floating', then the missing-value category is merged into the most similar group.

See also:

Mathematical Formulations: Decision Tree

Measure

Automatic Grow

Force Split

Using the Attribute Editor

Next Split

Next split displays the next significant split in the list of splits created by the Find Split command. This command is only enabled if there is more than one split created by the **Find Split** command.

From the menu:

Reshape | Next Split

From the popup menu:

Right click on parent node | Next split

Previous Split

Previous split displays the previous significant split in the list of splits created by the Find Split command. This command is only enabled if there is more than one split created by the **Find Split** command.

From the menu:

Reshape | Previous Split

From the popup menu:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Right click on parent node | Previous split

Go To Split

The **Go To Split** command allows you to go to a different split on the parent node. If there are no other significant splits calculated, this item will be disabled.

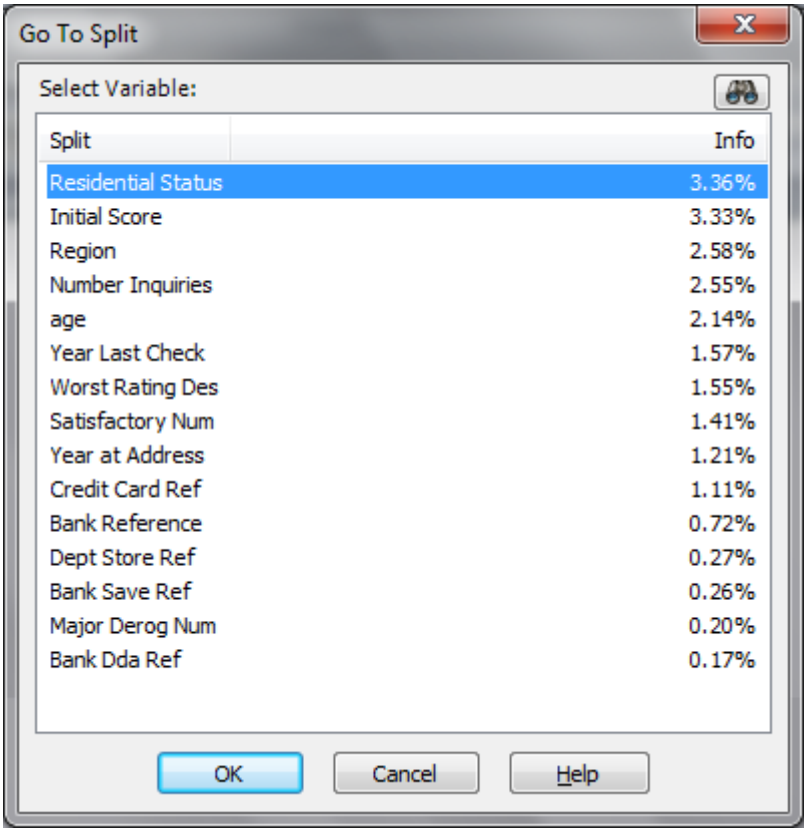
From the menu:

Grow | Go To Split

From the popup menu:

right-click on the parent node and select Go To Split

The **Go To Split** dialog displays the list of all the significant splits calculated for the given node in the order of their significance. The value displayed in the **Info** column is the same as in the Split Report for the current split. If the tree uses a ***log p-value*** measure, the splits are ranked by -log(*p*) or -log(adjusted *p*), and the corresponding header is shown instead of *Info*. See the Measures topic for more details.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

See also:

Next Split

Previous Split

Measure

Automatic Grow

The **Automatic Grow** command allows you to automatically grow a tree to the size specified in the automatic grow parameters. Each split on a node is the strongest statistical split for that node.

The **Automatic Grow** can be started on any node. Any previous **Find Split** or **Force Split** results above the current node will remain in the tree.

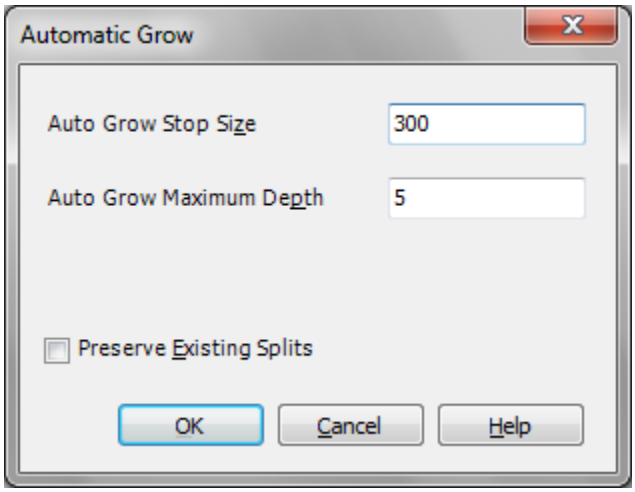
To use this command, you must have at least a root node of a tree and the tree must be active. The **Automatic Grow** command is called:

From the menu

Grow | Automatic Grow...

From the popup menu:

on the parent node, right-click | Automatic Grow...



Auto Grow Stop Size	Minimum number of records in a node needed before the **Automatic Grow** will attempt to split the node.
Auto Grow Maximum Depth	Determines the number of levels to grow in a tree. A value of **zero** leaves growth limits to other factors. The number of levels is measured from the current node.
Preserve Existing Splits	If checked, the existing tree is preserved and the growing starts from the terminal nodes. Otherwise, all splits are erased and Automatic Grow starts from scratch.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

You can change the default Auto Grow parameters at any time using the **Tools | Options** or **Tools | Preferences** menu commands (Tree Growth tab). Options change the defaults for the current tree only, whereas Preferences change them globally (for all trees to be inserted in the future). See the topic Preferences: Tree Growth for details.

If you change the Auto Grow Maximum Depth parameter to 0 (unlimited depth), the Automatic Grow command grows the tree until the other conditions are met (for Minimum Node Creation Size, Auto-Grow Stop Size, and Minimum DV Values) or there is no more significant splits found for any node (whichever happens first).

The Automatic Grow process is controlled by the parameters in the Tree Growth and Tree Training tabs of the **Options** (or **Preferences**) dialog box.

Limiting Automatic Grow based on DV values

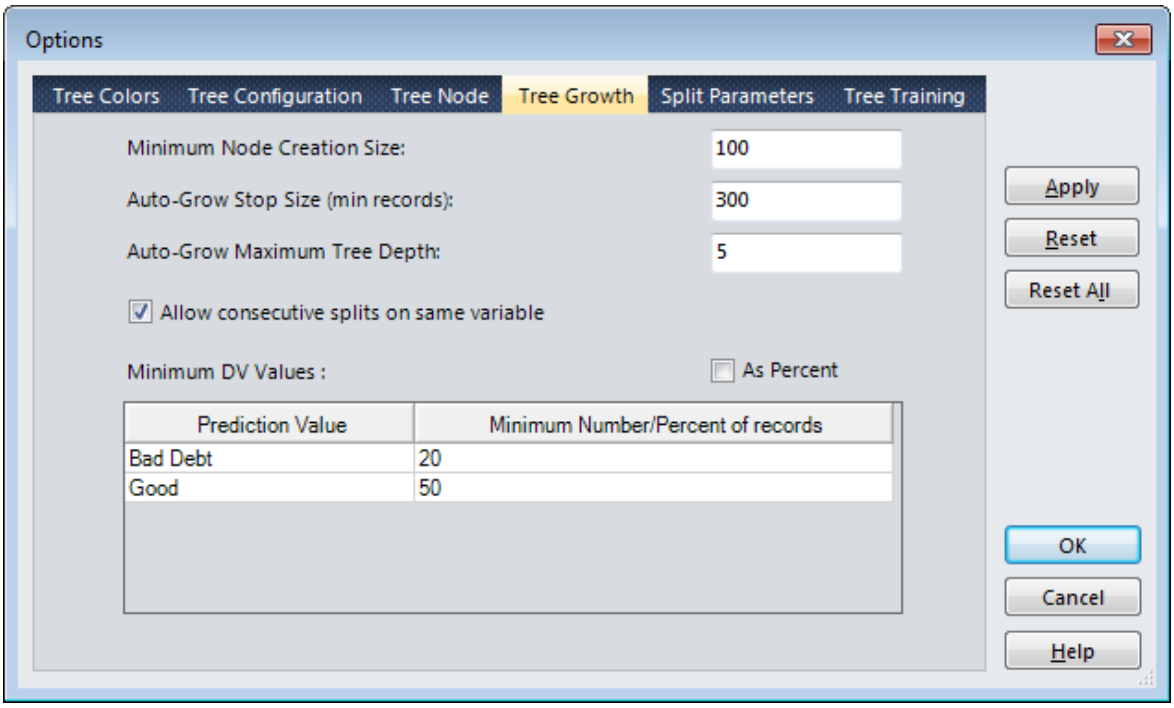
Aside from specifying the Auto-Grow Stop Size, Maximum Tree Depth, and Minimum Node Creation Size, you can also limit your Automatic Grow operation depending on the number (or percentage) of specific DV values in the nodes. This is done in the Tree Growth tab of the tree options dialog (**Tools | Options**).

For example, suppose that the dependent variable is AccountStatus, with values 'Bad Debt', 'Closed', and 'OK'.

If you specify the thresholds for Minimum DV Values as shown in the screenshot below, then the nodes where the number of records with AccountStatus = 'Bad Debt' is less than 150 or with AccountStatus = 'Closed' or 'OK' is less than 320 will **not** be created.

If you want to express your thresholds in terms of percentage of total number of records, select the checkbox 'As Percent' and enter the desired percentages.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also:

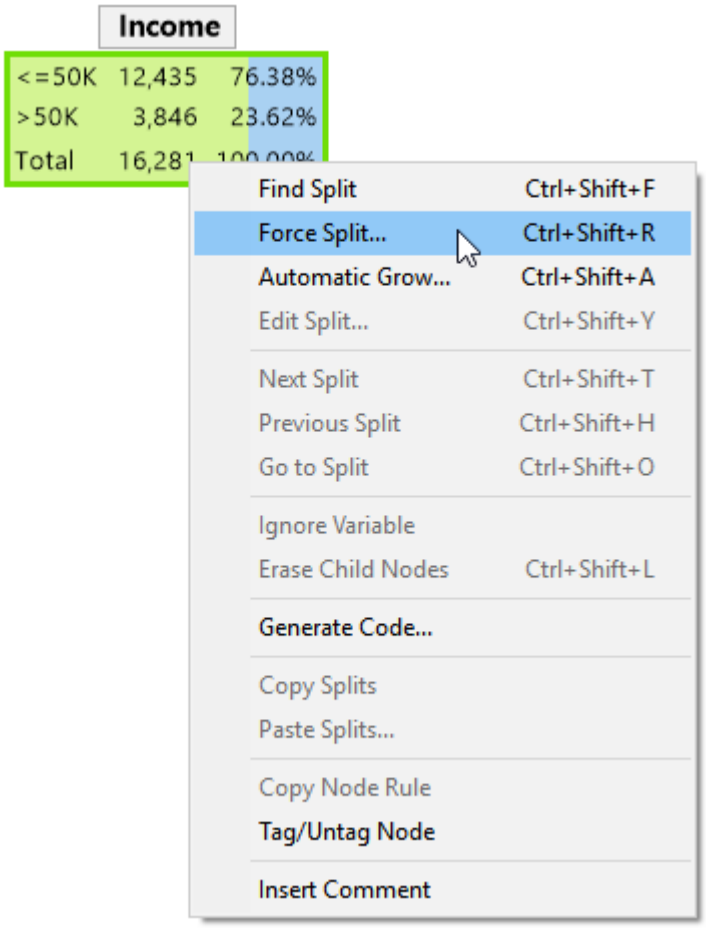
Find Split, Force Split

Force Split

Unlike Find Split, the **Force Split** command allows you to choose the independent variable to split on and gives you complete control over the bins of the split variable. In the default mode, Force Split does not use any algorithm to determine the most informative splits or rank independent variables. The split control is fully manual - the initial binning is "Equal Width" or "Equal Height", and the user is free to define their own custom bins starting from that point.

Right-click on the node of the tree that you want to split and select **Force Split** from the context menu. Alternatively, select the **Force Split** command from the **Grow** menu or use the keyboard shortcut CTRL + SHIFT + R.

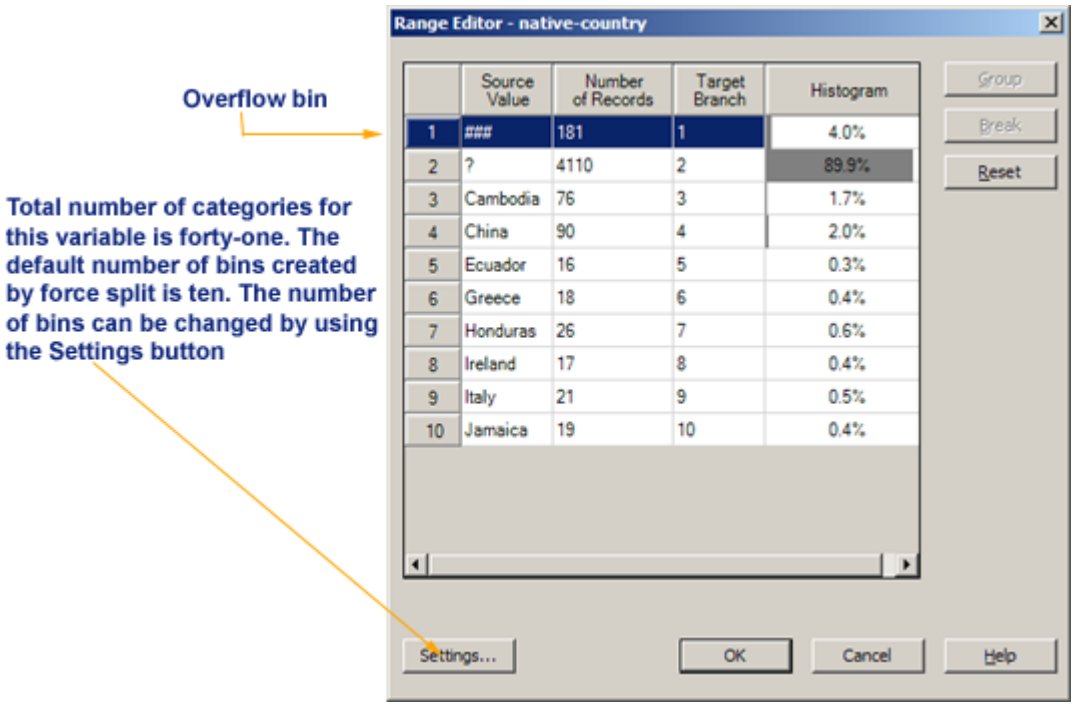
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



A series of steps takes you through the **Force Split** process. This includes choosing the split variable and defining the binning for it.

- The **Force Split** command does not use the decision tree algorithm and therefore ignores the settings in the Attribute Editor that are relevant to the Find Split algorithm.
- If the cardinality of a discrete (nominal) split variable is greater then the *Max Bins* threshold defined in the Tree Attributes tab of Preferences, then the split variable is mapped into the number of bins equal to *Max Bins* = *N* (N-1 most populous categories from the variable remain in their own category, while the others are mapped to the overflow bin, represented as ###). The default *Max Bins* is 50. For details, see the topic Preferences: Tree Attributes
- Example

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- The **Settings** command allows you to increase the number of bins (for example if you have move than 10 categories, you can show them all rather than using the overflow bin). A high cardinality split variable will be limited to one hundred bins/categories. There will be ninety-nine categories (or ranges) and the hundredth (100th) bin will be the overflow bin, represented as ###.
- If the selected variable is continuous, then it is initially mapped into a number of intervals defined by the *Max Intervals* parameter in the Tree Attributes tab of Preferences. The default is 10. For details, see the topic Preferences: Tree Attributes. These bins can be changed by editing the splits directly or using the Settings command.
- When performing a force split on a node that contains existing splits, all the splits will be replaced by the force split command. This is the same as performing an Erase All before doing a force split. If you would like to preserve some of the child node information, you may consider the **Copy and Paste splits** command.

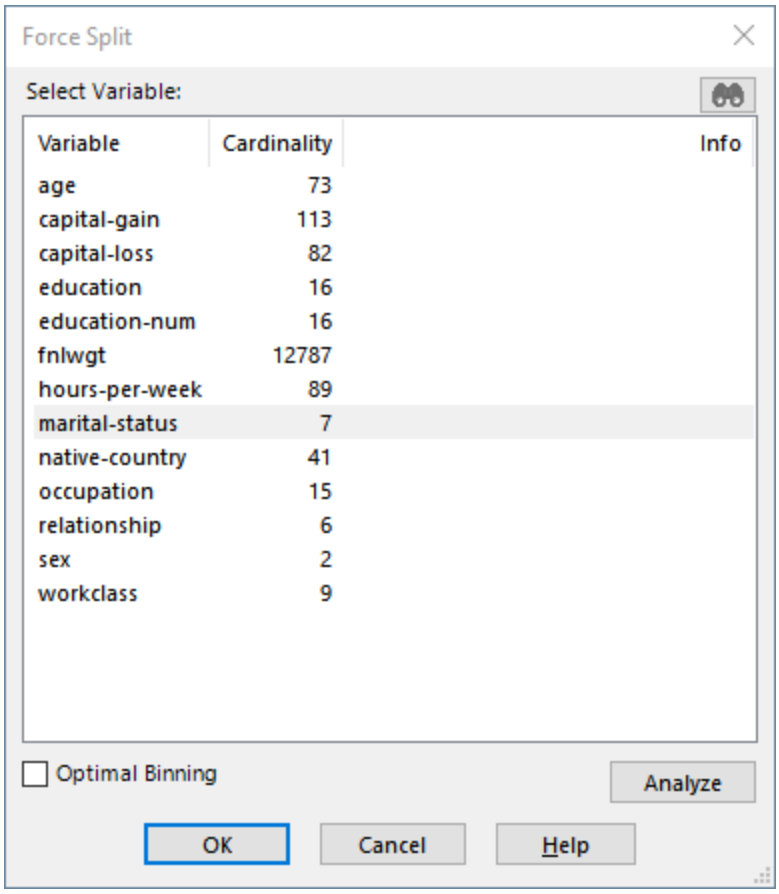
See the topic: Choose the Split Variable.

Choosing the split variable

The **Force Split** dialog allows you to choose the variable to split on. The variables are displayed in alphabetical order. Only the variables included in the Attribute Editor are shown in the list.

If the currently selected node already has child nodes, the existing split variable is highlighted in the Force Split dialog. Otherwise, the top variable in the list is highlighted.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

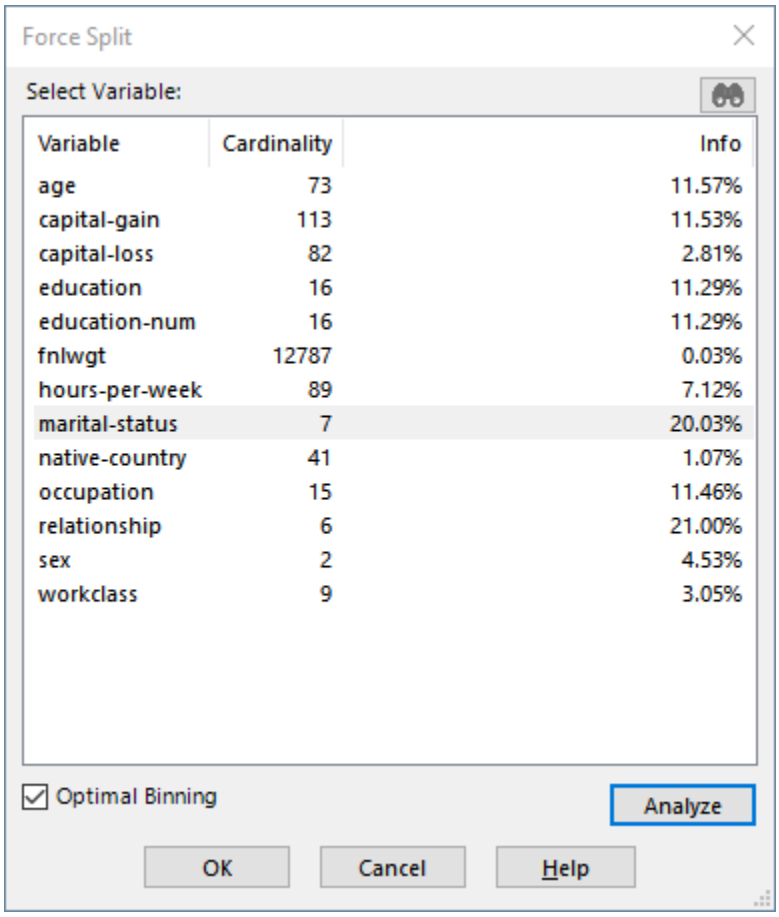


By default, Force Split does **not** use any algorithm to determine the most informative splits or rank independent variables. Unless you select the *Optimal Binning* option, the split control is fully manual - the initial binning is "Equal Width" or "Equal Height", and the user is free to define their own custom bins starting from that point.

Click **Analyze** if you want to compute the relative significance of the variables with respect to the dependent variable. It will be displayed in the Info column. The way Info is calculated depends on the selected measure. See the topic Measure for details.

Select the **Optimal Binning** option in this diallog if you want to apply an intelligent algorithm to automatically produce the optimal binning for the selected variable - the most informative binning with respect to the dependent variable distribution. This is equivalent to performing the **Find Split** command followed by the **Go To Split** command and selecting the desired variable. For the details of the optimal binning algorithm, see the Find Split topic.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

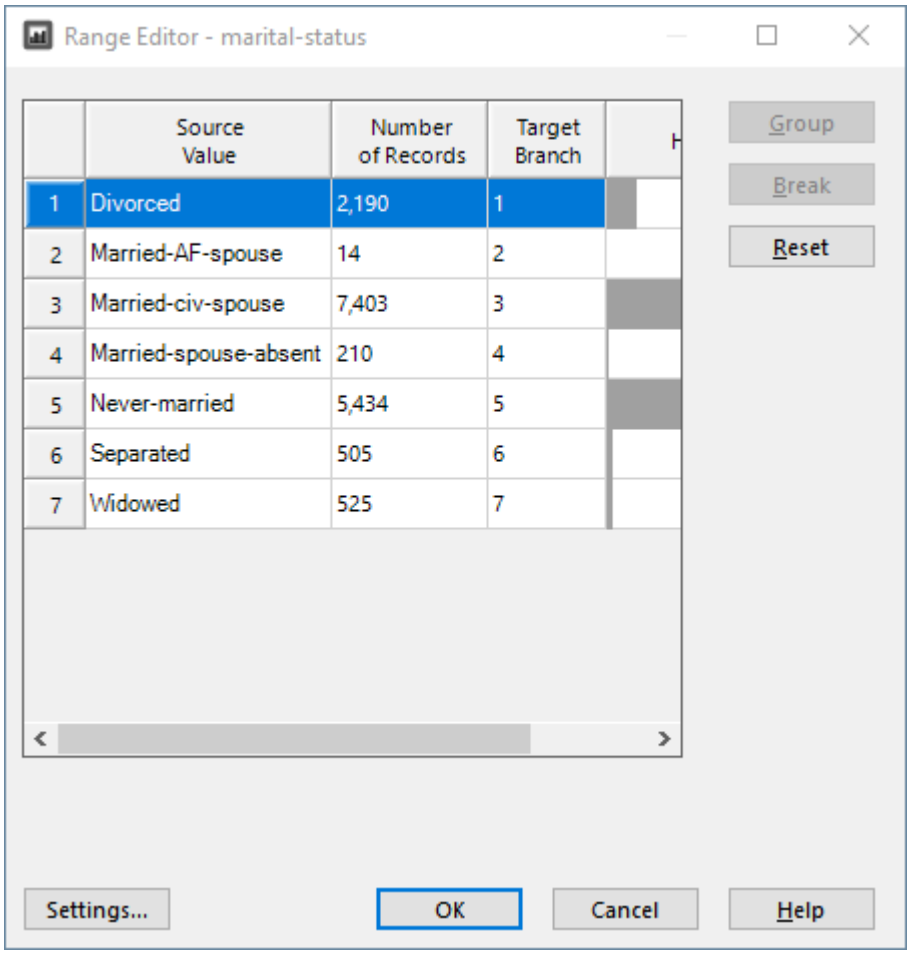


Select the desired variable and click **OK**. This will bring up the Range Editor dialog. The Range Editor functions depend on the type of the independent variable selected. See the topics Range Editor: Discrete and Range Editor: Continuous.

Range Editor for Discrete Variables

The **Range Editor** dialog of the Force Split function allows you to modify the bins of the selected variable: group or ungroup values, specify the maximum number of bins.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Source Value	Categories arranged as bins. Prior to any categories being grouped together, the number of categories displayed is equal the number of bins requested. The number of bins is set in the Settings. If there are more categories than requested bins, the remaining categories are grouped as ###.
Number of Records	Number of records associated with the bin.
Target Label	The Target Label is only a reference.
Histogram	Percentage of total records contained in the bin.

Bin Editing

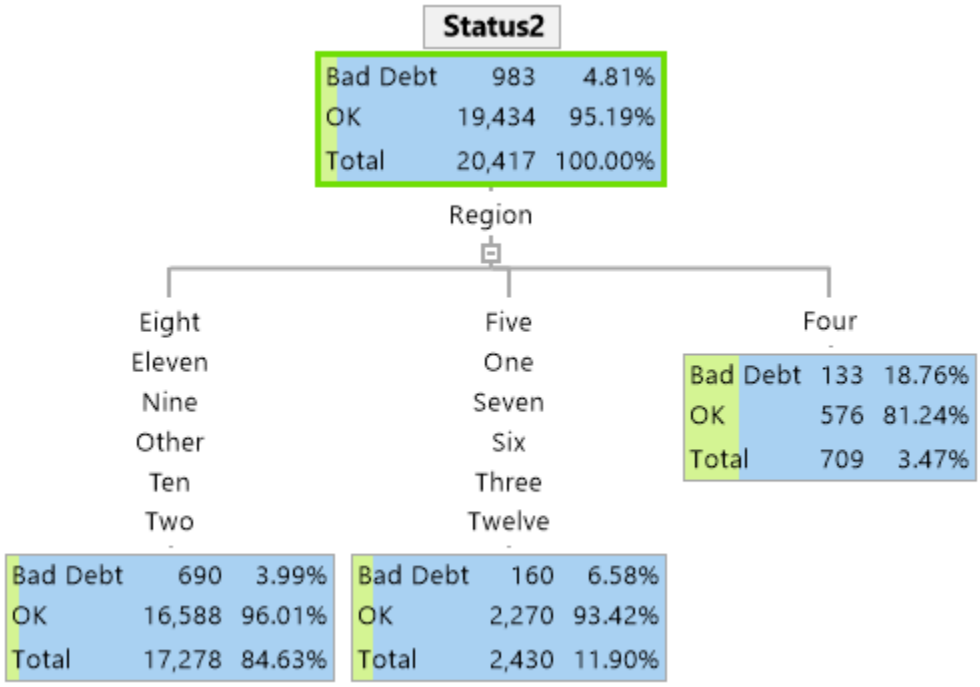
Group	Allows you to combine bins. Example
Break	Allows you to break apart grouped bins. Example

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Reset	The reset button allows you to return to the original bins displayed when the range editor was first called.
Settings	The settings button allows you to change the default bin settings. See Settings for Discrete Variables .

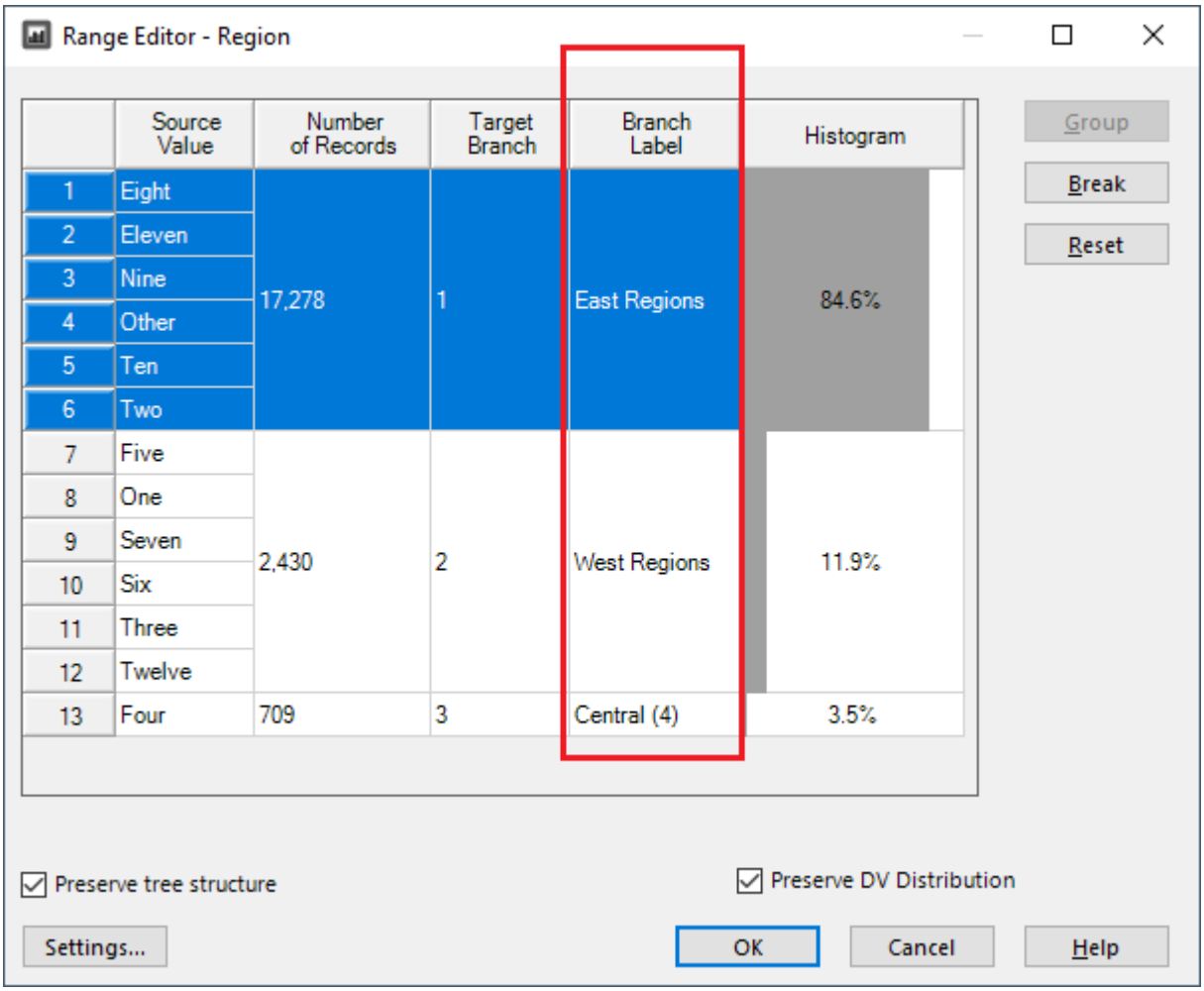
Branch Labels in Splits

When forcing or editing a split, after grouping a large number of discrete values of a nominal variable, you may want to label such groups and display the labels instead of stacked values that sometimes take too much space in the tree view. In the following example of a split on region codes, the two child nodes on the left have marginal rules with stacked values that take too much space:

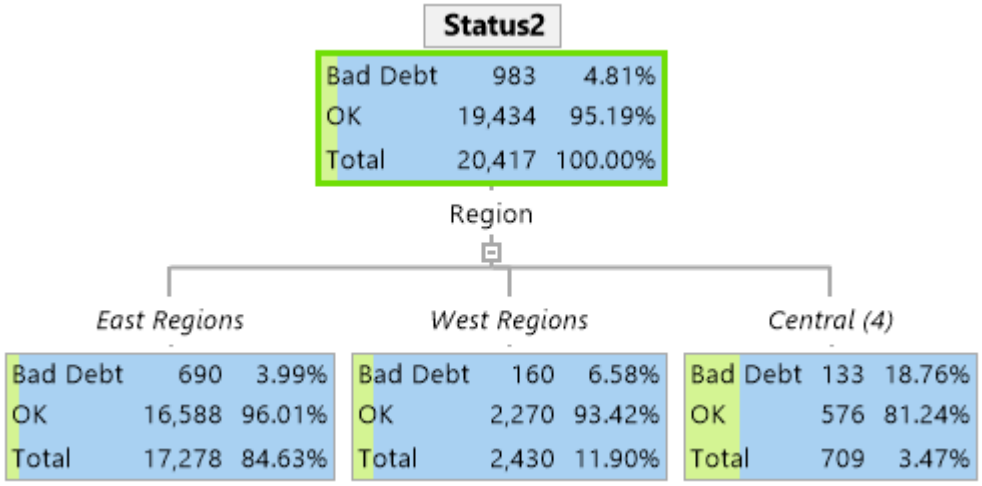


Label these groups in the Range Editor dialog of Force Split by entering the desired labels in the **Branch Label** column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



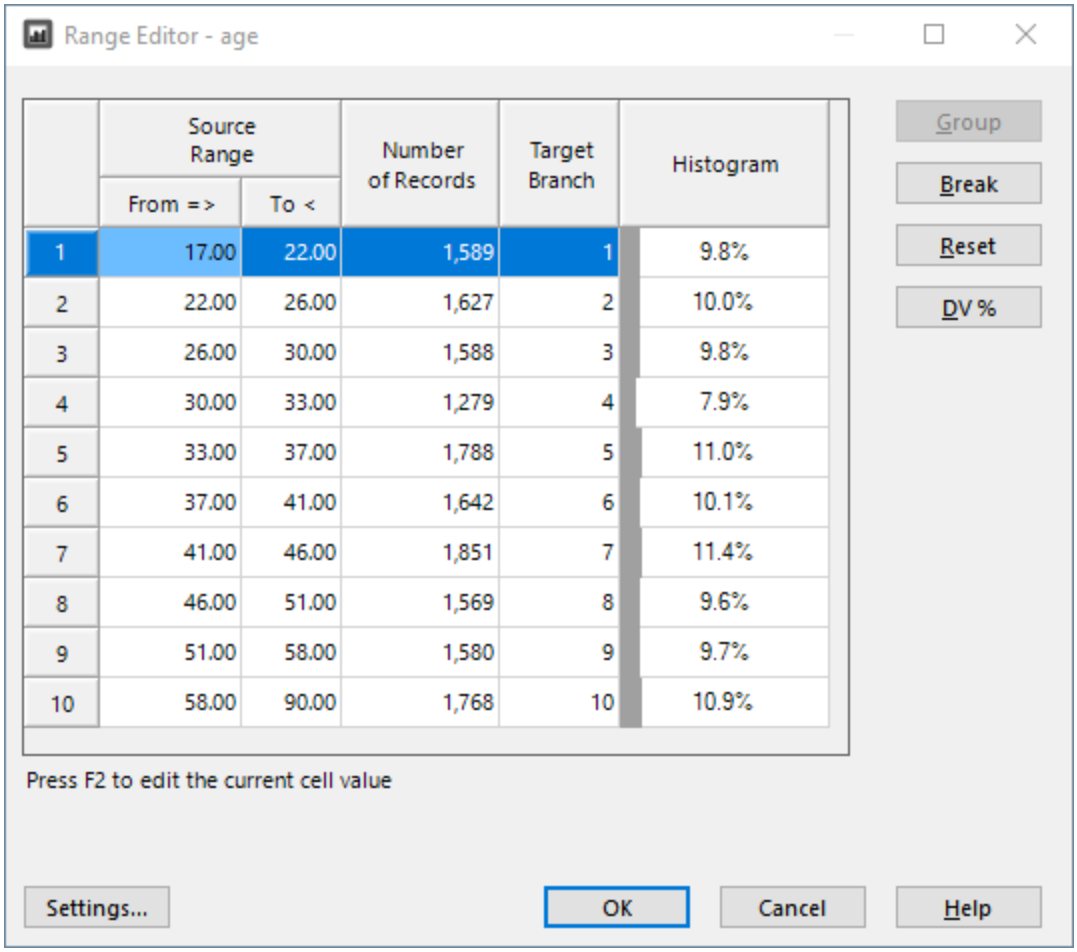
As a result, the split looks much more compact. The labels are dispayed in *cursive font* to distinguish them from regular individual values of the variable.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Range Editor for Continuous Variables

The **Range Editor** dialog of the Force Split function allows you to modify the bins of the selected variable: merge or break intervals, specify boundary values, specify the maximum number of intervals.



Range Display

Source Range From=> To<	Indicates the beginning and ending range of values that for each bin.
Number of Records	Number of records contained in the bin.
Target Label	Label for the bin as it will appear in the expression editor. The label can be modified by double-clicking on the cell and entering a new label.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Histogram	Percentage of records in the bin.

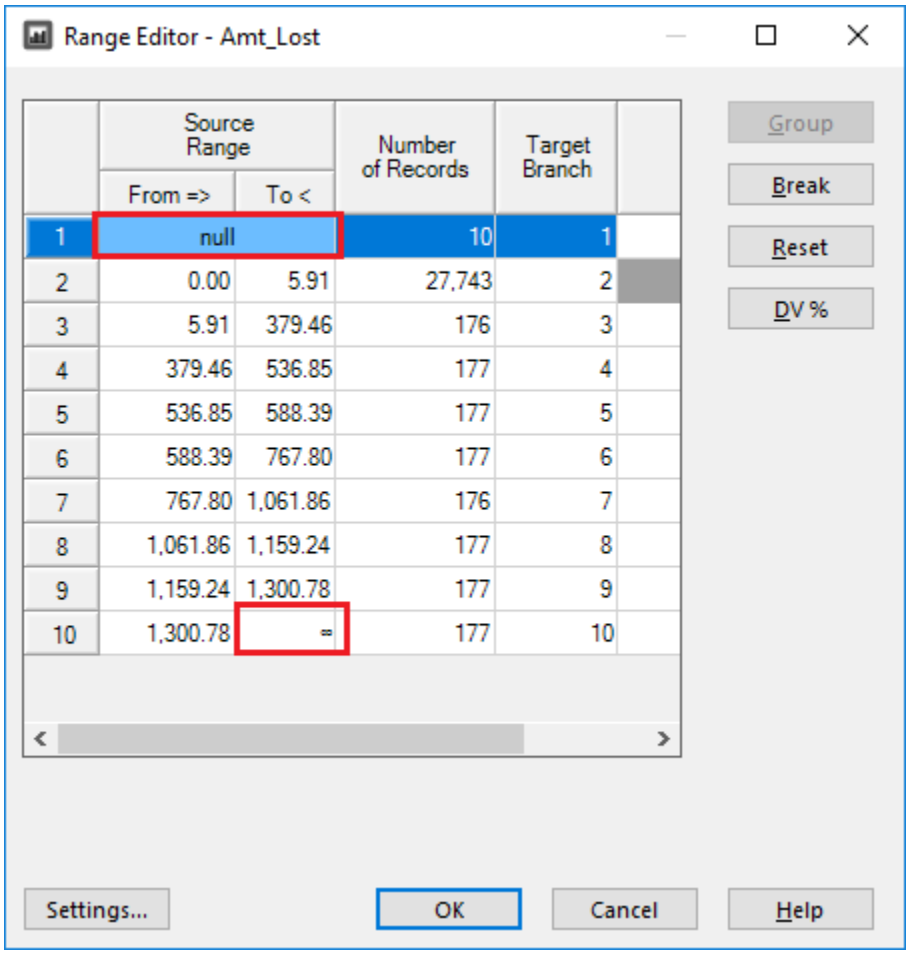
Range Editing

Group	Allows you to combine ranges. Only contiguous ranges can be grouped. Example
Break	Allows you to create new ranges by breaking the existing ranges. Example
Reset	The reset button allows you to return to the original ranges displayed when the range editor was first called.
DV %	Using the DV % button you can specify the desired percentage of each DV value in each child node of the split. See the topic Force/Edit Split with Requested DV Percentage
Settings	The settings button allows you to change the default range settings. See Settings.

Special and Singular Values (infinity, missing values)

If the variable contains special values such as positive or negative infinity +/-{INF} or missing values (*null*), the missing value is shown in a separate bin labeled ***null***, while infinity is naturally considered to be the maximum value in the variable and is shown as the right boundary of the last interval. In the following example, the variable Amt_Lost contains both missing values and infnity values.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



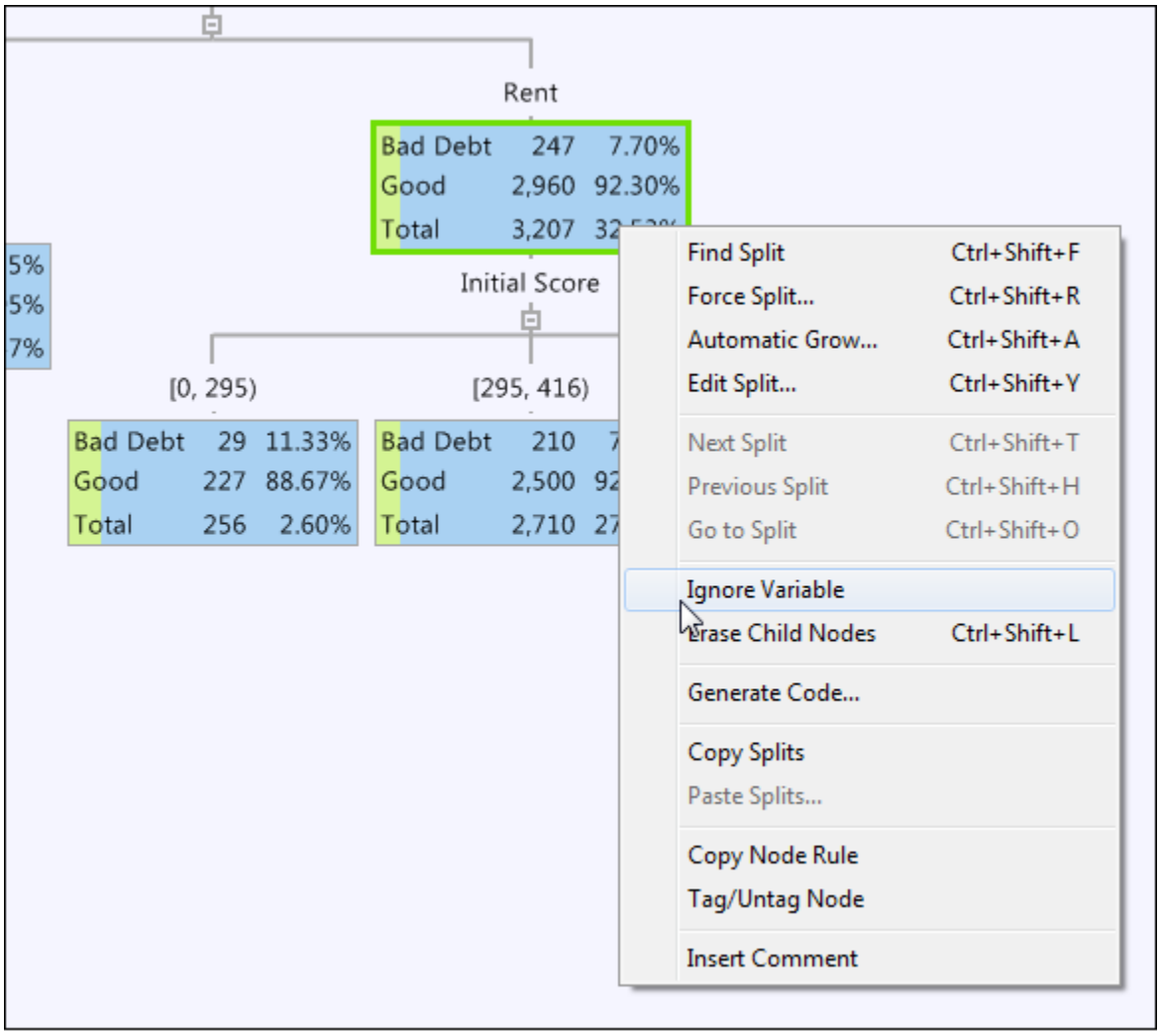
Ignore Variable

The **Ignore Variable** command removes the split on the independent variable displayed under the currently selected node. If there are any background splits at the same node (for example, as is usually the case after Find Split), then the next split is displayed.

You can also exclude variables from the Attribute Editor, but the next split is not displayed in that case.

In the example given below, to ignore the split on ***Initial Score***, right-click on the node and select **Ignore Variable** from the context menu.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Alternatively, click on the parent node and select the **Ignore Variable** command from the Reshape menu.

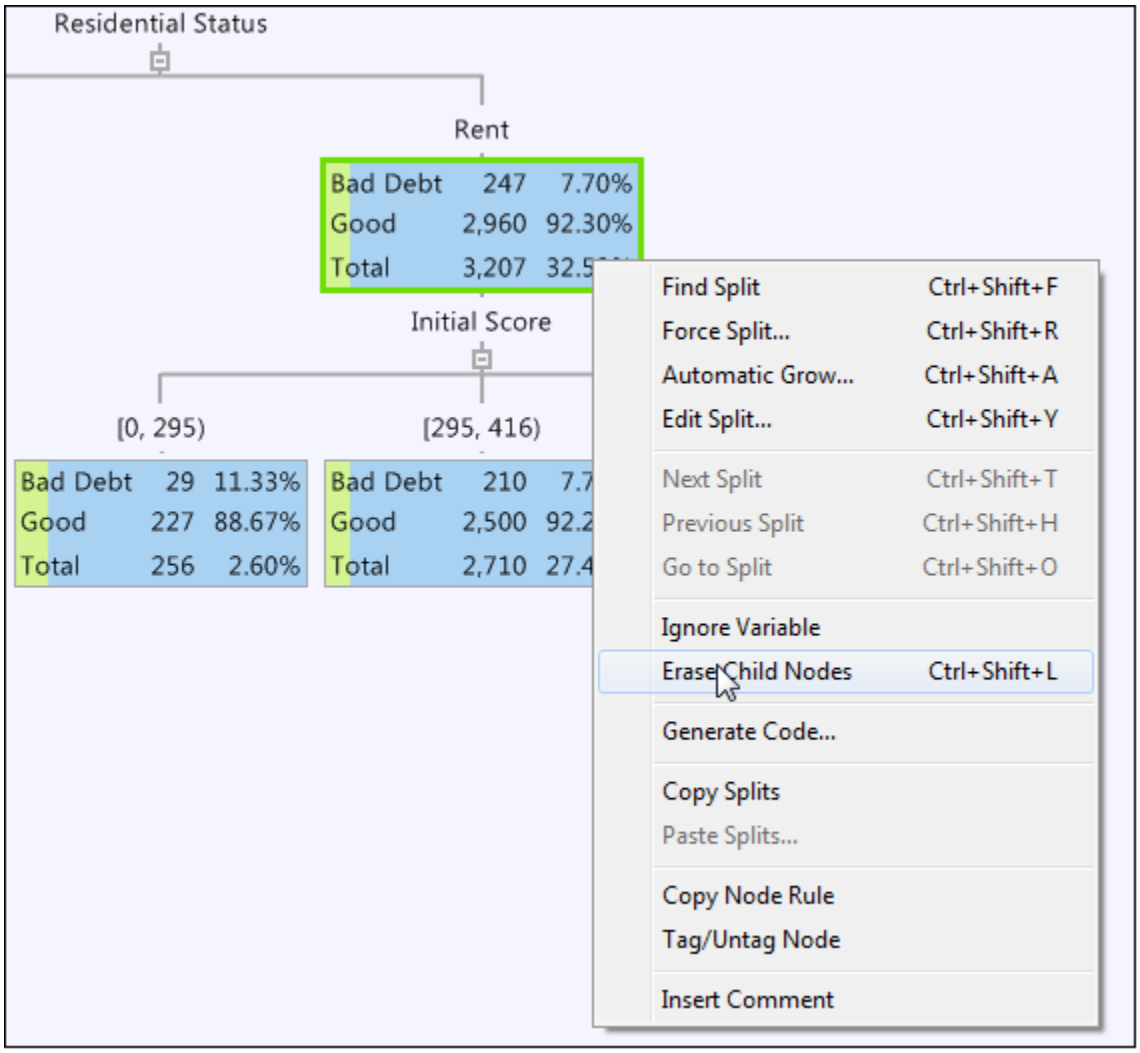
See also: Edit Split, Find Split, Force Split, Erase Child Nodes

Erase Child Nodes

The **Erase Child Nodes** command deletes all child nodes of the currently selected node. If there are several levels of child nodes under the current node, all levels are erased.

To erase the child nodes of a specific node, right-click on it and select the command **Erase Child Nodes** from the context menu, as shown below. Alternatively, click on the parent node and use the keyboard shortcut CTRL + SHIF + L.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



You can also use the **Erase child Nodes** command from the Reshape menu.

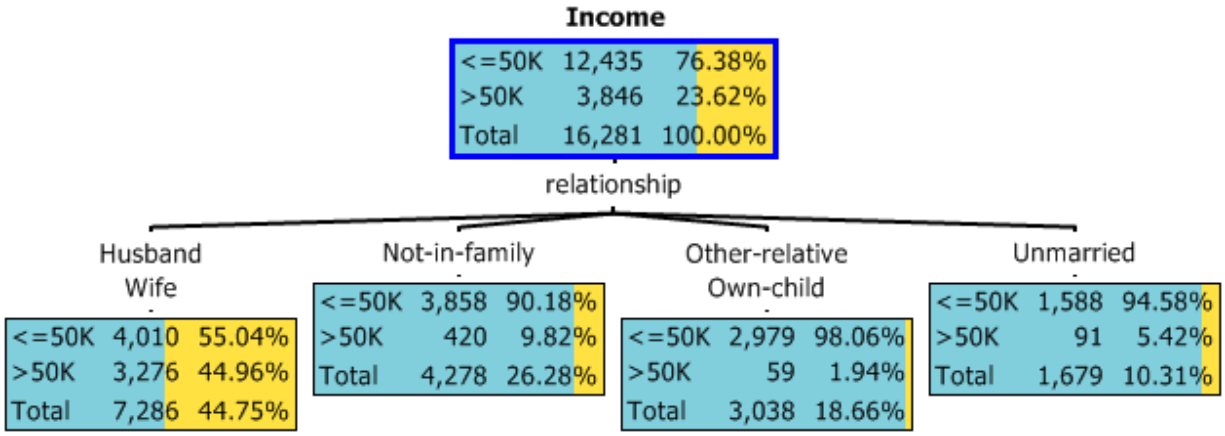
See also: Edit Split, Find Split, Force Split, Ignore Variable

Editing Splits

The **Edit Split** command allows you to modify the display of the ranges (for continuous variables) or bins (for discrete/categorical variables) in the currently selected split.

Note: If you have more than one significant split at the currently selected node (for example, resulting from a Find Split), then performing Edit Split on this node will affect only the visible split, and the other splits at this node will be preserved (the **Next/Previous/Go to Split** commands will still be enabled).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



From the context menu:

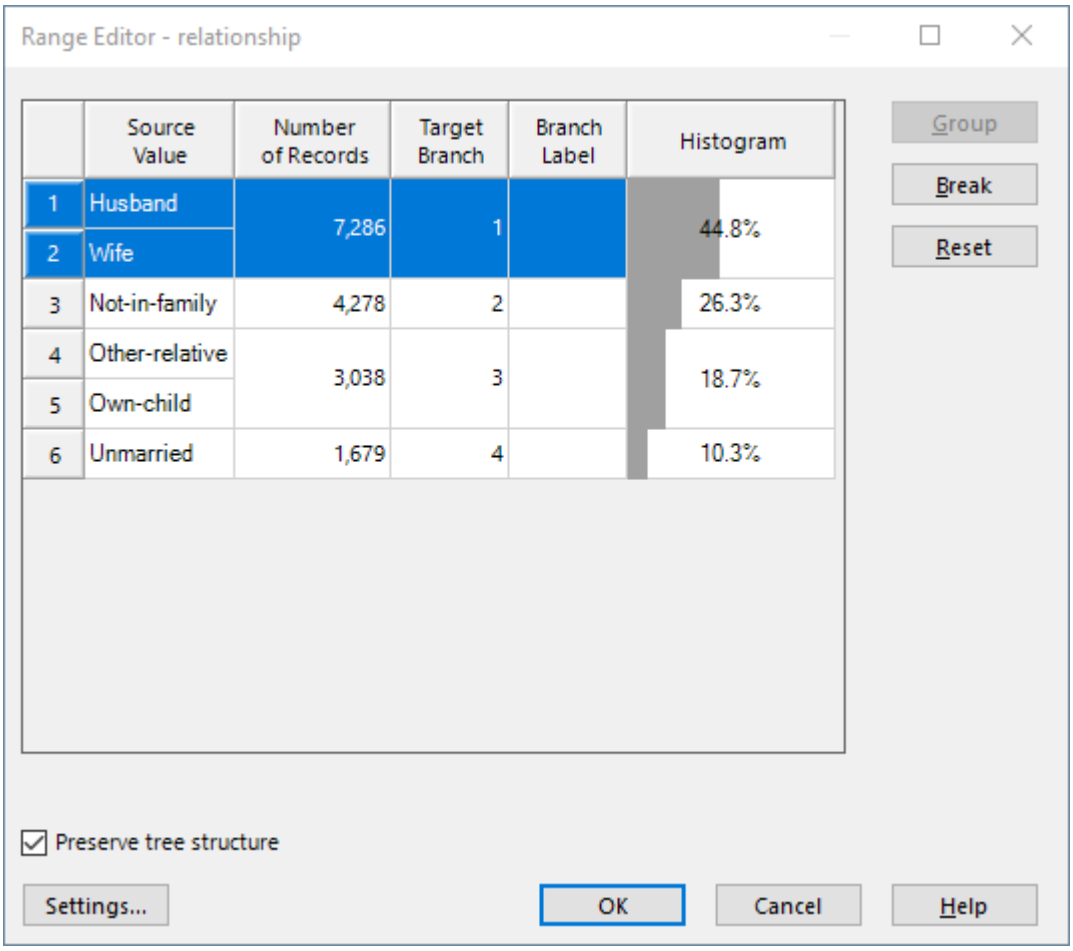
Right click on the parent node and select Edit Split

From the menu:

Grow | Edit Split

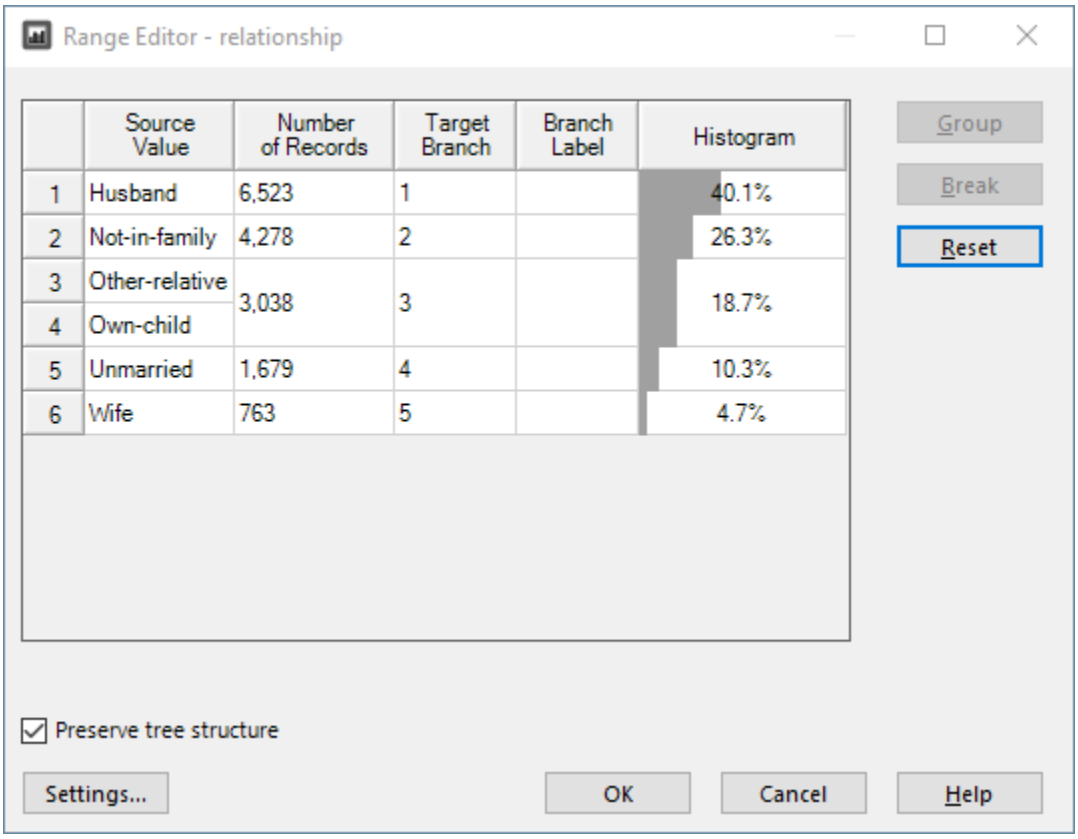
When you select the **Edit Split** command, it displays the distribution of bins/ranges at the current node in **Range Editor**, whose functions and options are exactly the same as in the case of the **Force Split** command. For example when editing the split shown above, we get:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



For example, if you want to ungroup the categories ***Husband*** and ***Wife*** in this case, highlight the group and click the **Break** button. As a result, you will get the following distribution:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Note: If you perform an Edit Split operation of a node with grandchildren, you have the option of preserving the splits unaffected by the Edit Split whenever possible. By default, the unaffected splits will be preserved (the **Preserve tree structure** checkbox in Range Editor is selected).

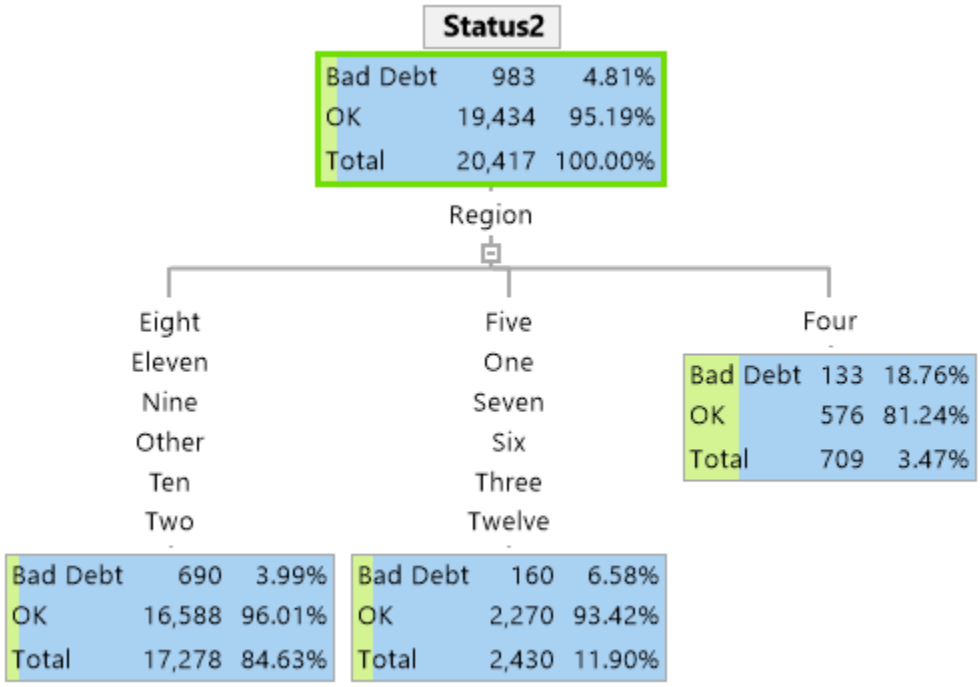
Refer to the topics Range Editor (Discrete) and Range Editor (Continuous) for details about using the Range Editor for discrete and continuous independent variables.

Note: In the case of a continuous independent variable, real numbers that represent interval boundaries in Range Editor are displayed with fixed precision (2 digits after the decimal point) if the cells are not in Edit mode. To enter the Edit mode, click on a cell or press the F2 key. This will display the actual value with the original precision.

Branch Labels in Splits

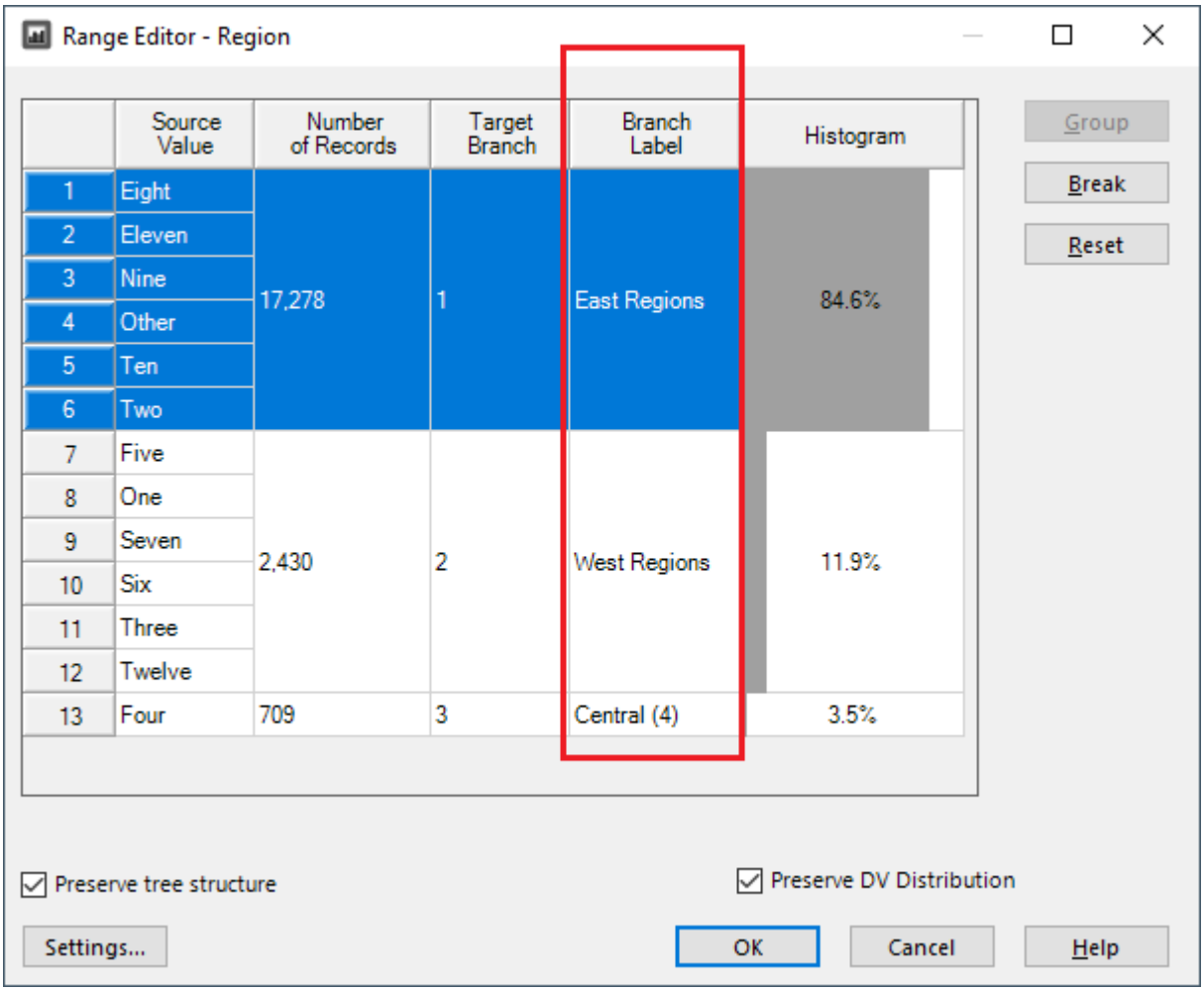
When editing or forcing a split, after grouping a large number of discrete values of a nominal variable, you may want to label such groups and display the labels instead of stacked values that sometimes take too much space in the tree view. In the following example of a split on region codes, the two child nodes on the left have marginal rules with stacked values that take too much space:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

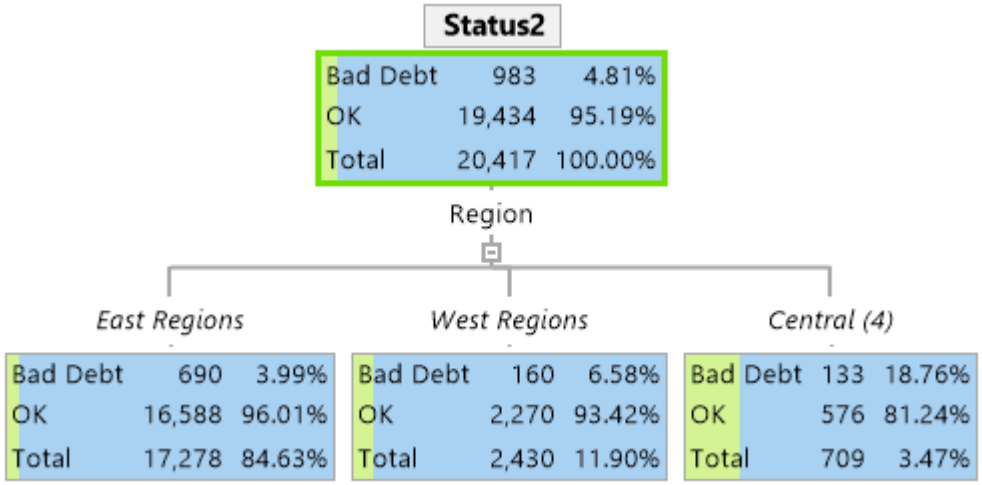


Using the Edit Split command, you can label these groups in the Range Editor dialog by entering the desired labels in the **Branch Label** column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



As a result, the split looks much more compact. The labels are dispayed in *cursive font* to distinguish them from regular individual values of the variable.

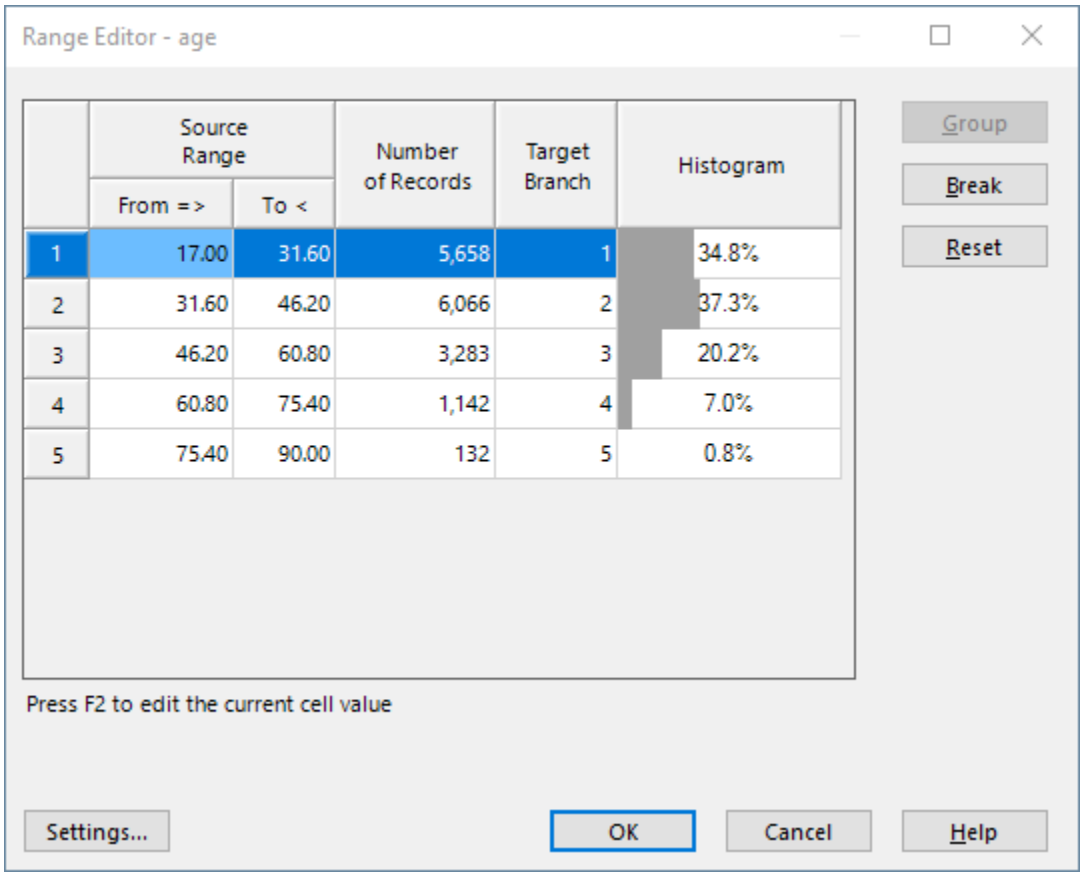


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Editing Split for Continuous Variable

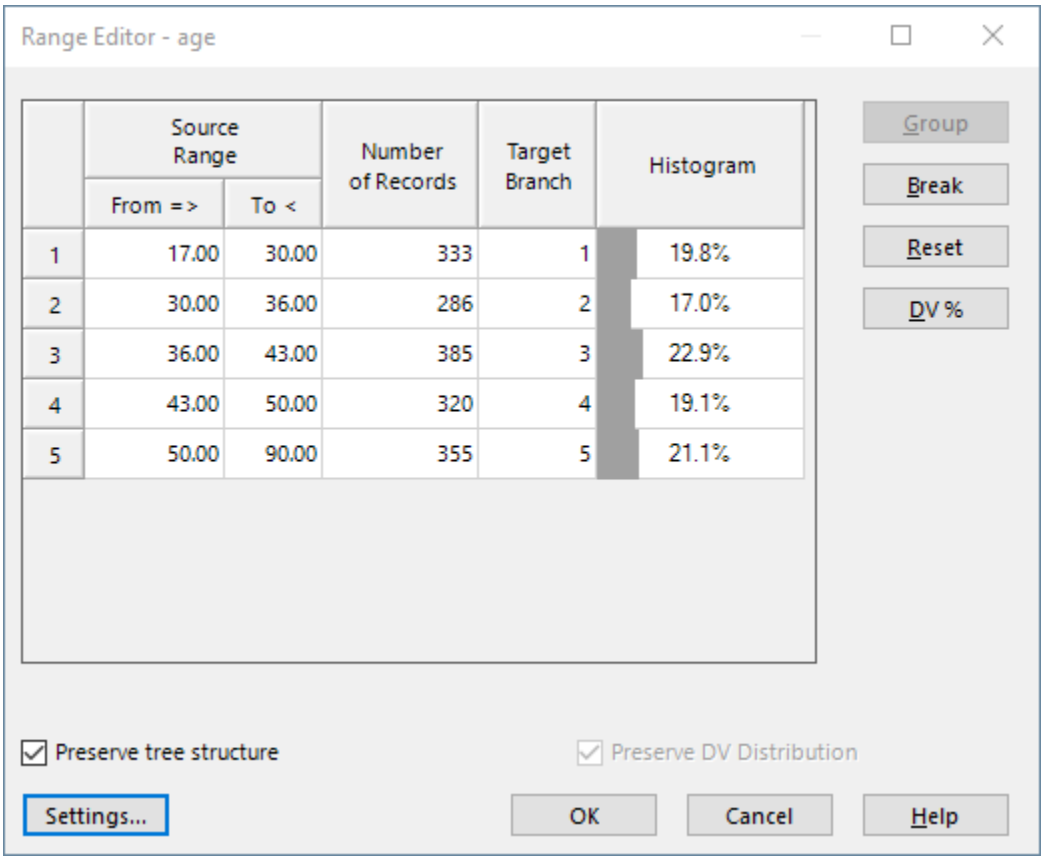
The range editor allows you to define how your continuous data will be grouped and distributed across intervals.

In Dataset Chart and Variable Transformations:



In Decision Trees or Strategy Trees:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Note: The option "Preserve DV Distribution" is applicable only to the Edit Split operation in Strategy Trees.

Range Display

Source Range From=> To<	Indicates the beginning and ending range of values that for each bin.
Number of Records	Number of records contained in the bin.
Target Branch	Label for the bin as it will appear in the expression editor. The label can be modified by double-clicking on the cell and entering a new label.
Histogram	Percentage of records in the bin.

Range Editing

Group	Allows you to combine ranges. Only contiguous ranges can be grouped. Example
Break	Allows you to create new ranges by breaking the existing ranges.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

	Example
Reset	The reset button allows you to return to the original ranges displayed when the range editor was first called.
DV %	Using the DV % button you can specify the desired percentage of each DV value in each child node of the split. See the topic Force/Edit Split with Requested DV Percentage
Settings	The settings button allows you to change the default range settings. See Settings.

Note: Real numbers in Range Editor are displayed with fixed precision (2 digits after the decimal point) if the cells are not in Edit mode. To enter the Edit mode, click on a cell or press the F2 key. This will display the actual value with the original precision.

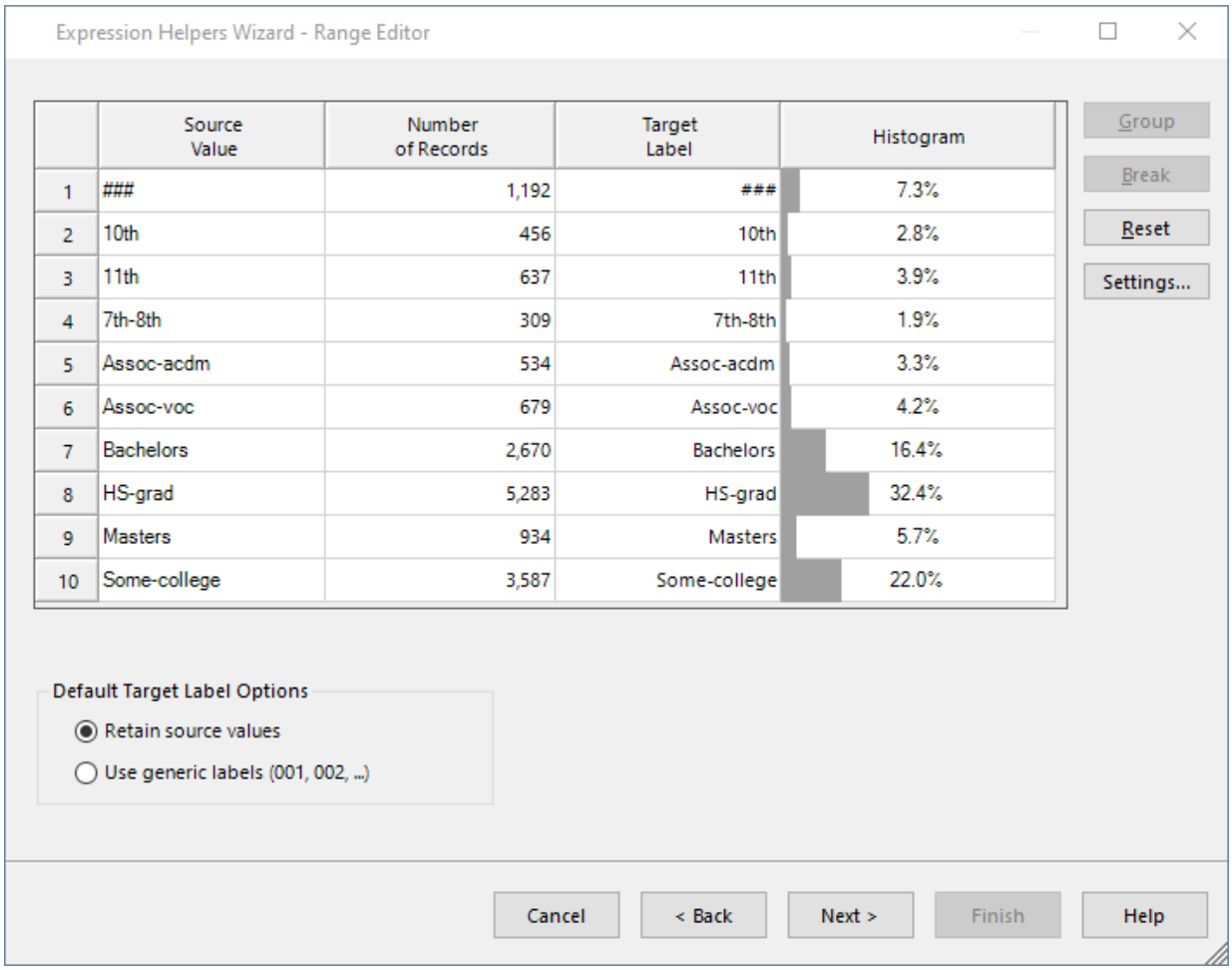
For Decision Trees and Strategy Trees: If you edit a split for a node with grandchildren, you have the option of preserving the branches unaffected by the Edit Split whenever possible. By default, the splits in the unaffected branches will be preserved (the **Preserve tree structure** checkbox is selected). For the description of the option "Preserve DV Distribution", see the Edit Split topic in the Strategy Tree section.

Editing Split for Discrete Variable

The Range Editor allows you to define how the categories will be grouped and distributed across the bins.

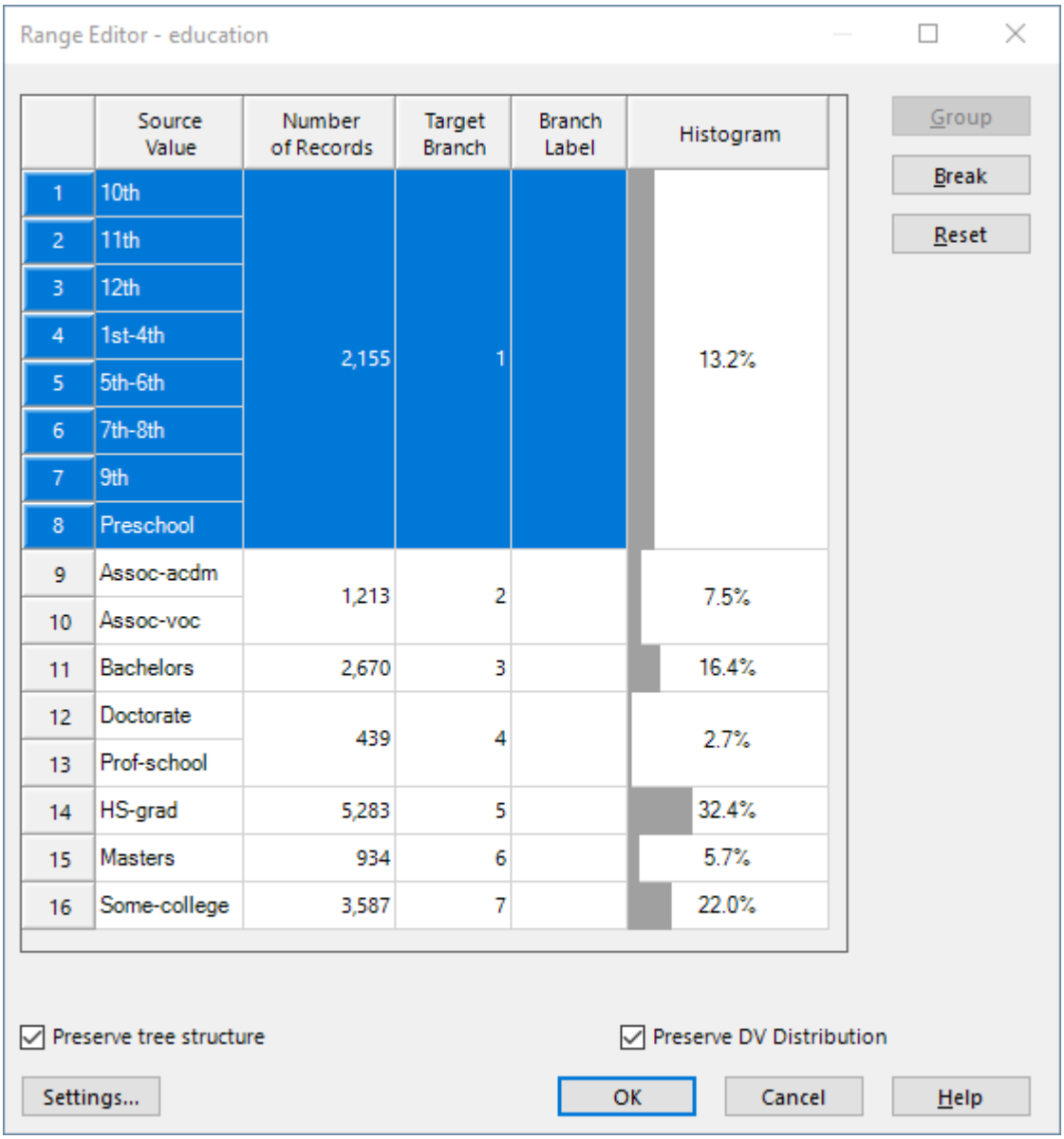
In the Binning Helper of Variable Transformations:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



In Decision Trees or Strategy Trees:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Note: The option "Preserve DV Distribution" is applicable only to the Edit Split operation in Strategy Trees.

Category Display

Source Value Categories arranged as bins. Prior to any categories being grouped together, the number of categories displayed is equal to the number of bins requested. The number of bins can be modified using the Settings button. If there are more categories than the number of requested bins, the categories with the highest frequencies remain as they are, and the remaining categories are grouped as ###.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Number of Records	Number of records contained in each bin.
Target Branch	Label for the bin as it will appear in the expression editor. The label can be modified by double-clicking on the cell and entering a new label.
Histogram	Percentage of total records contained in the bin.

Category Group Editing

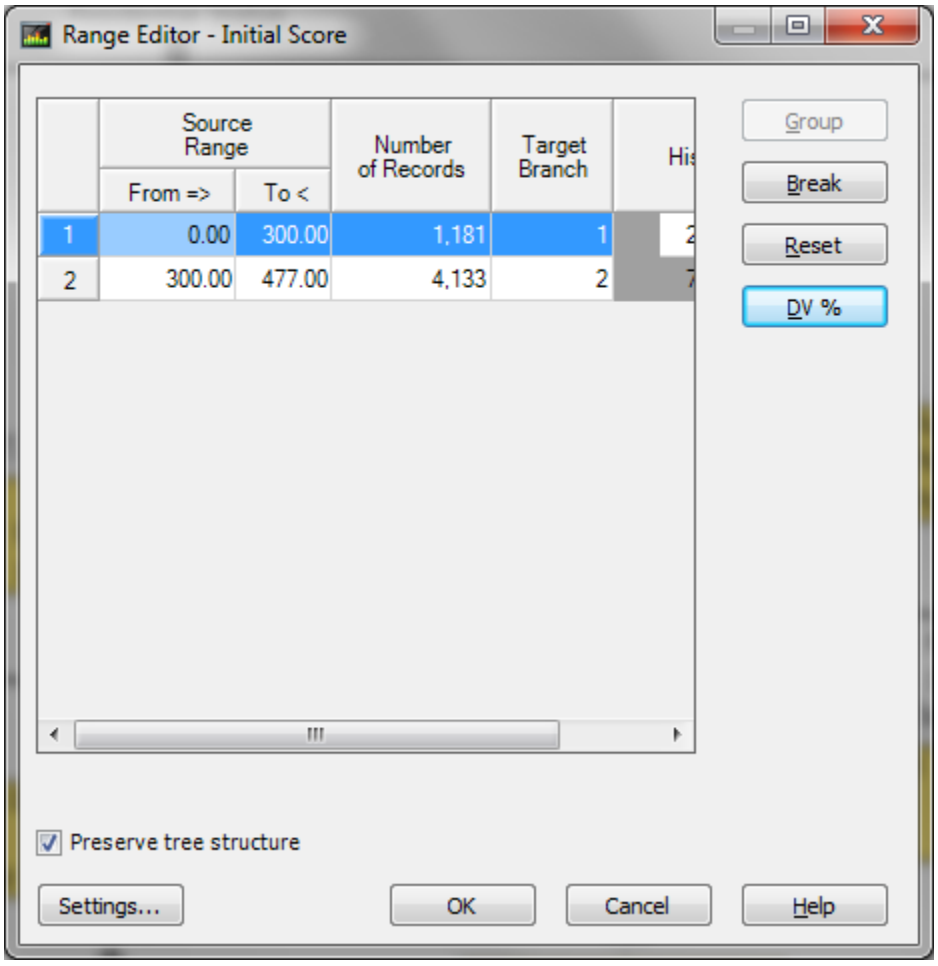
Group	Allows you to combine categories. When the categories are grouped, the weight column reflects the total number of records within the group. Example of Group
Break	Allows you to create new ranges by breaking the existing ranges. Example of Break
Reset	The reset button allows you to return to the original ranges displayed when the range editor was first called.
Settings	The settings button allows you to change the default number of bins.

For Decision Trees and Strategy Trees: If you edit a split for a node with grandchildren, you have the option of preserving the branches unaffected by the Edit Split whenever possible. By default, the splits in the unaffected branches will be preserved (the **Preserve tree structure** checkbox is selected). For the description of the option "Preserve DV Distribution", see the Edit Split topic in the Strategy Tree section.

Force/Edit Split with Requested DV Percentage

When forcing a split or editing a split on a continuous variable, the **DV %** button in the Range Editor provides the ability to specify the desired percentage of each category of the dependent variable in each of the child nodes in the split. This function is applicable only in Decision Trees with discrete dependent variable.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

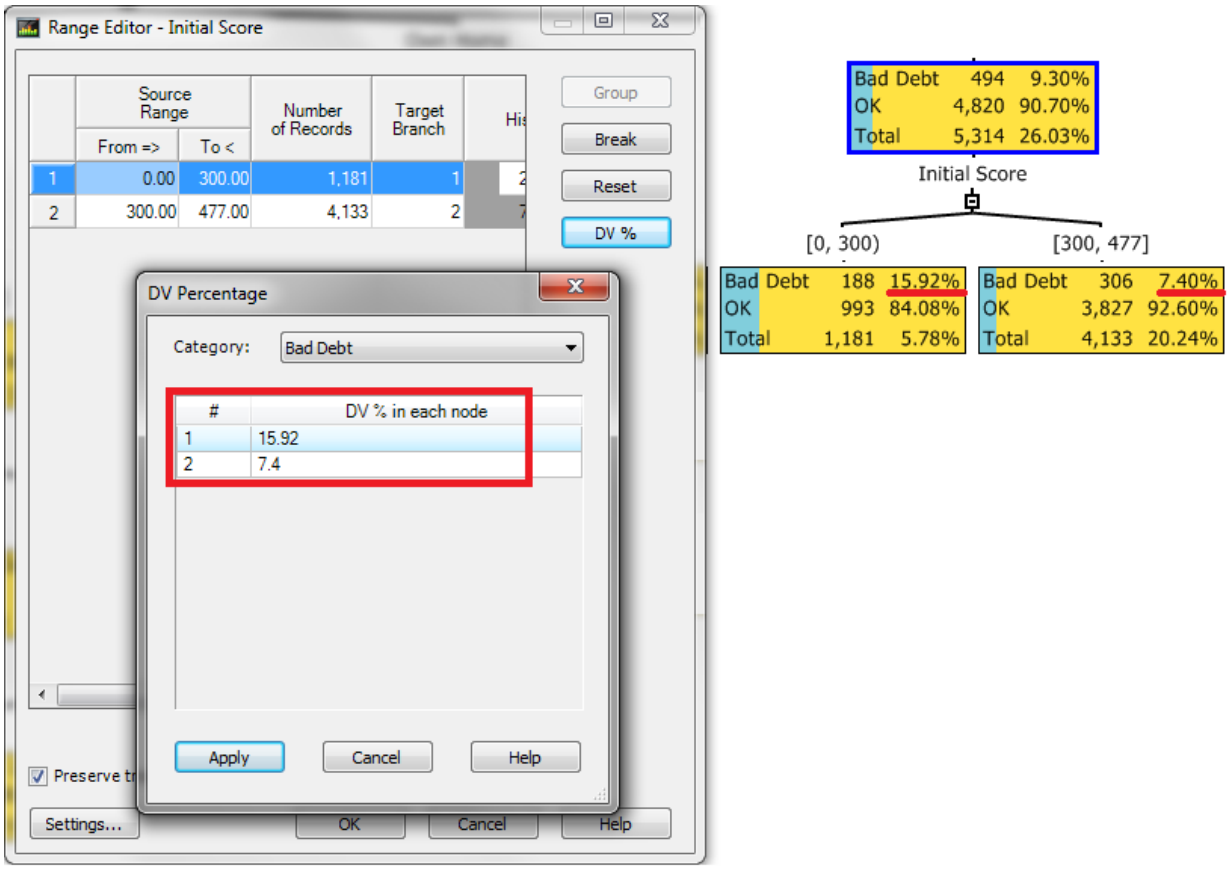


Clicking the **DV %** button brings up the **DV Percentage** dialog.

- First, choose the desired DV category from the Category drop-down list.
- In the grid below, specify the desired percentage for the chosen DV category in each child node of the split.

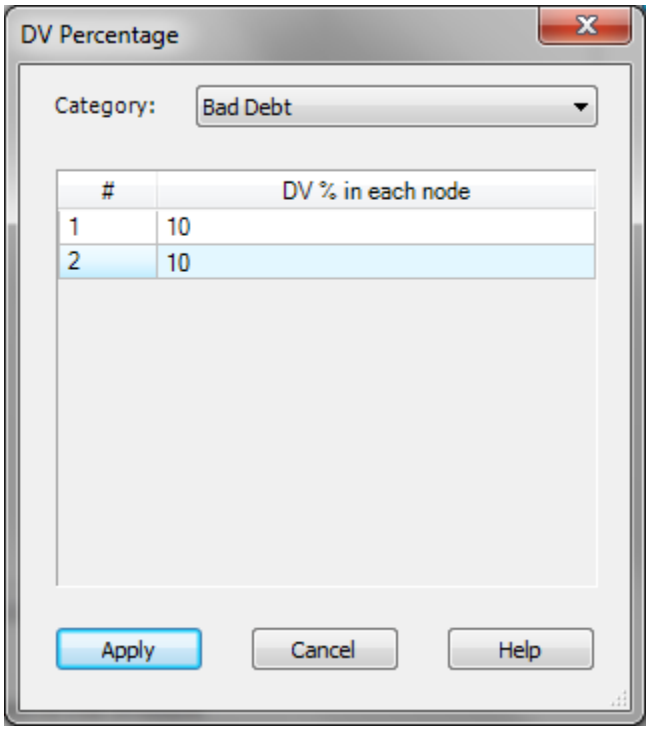
The example below illustrates the process of modifying the DV % when editing the split shown on the right-hand side. Initially, when the Bad Debt category is chosen, the dialog shows the current percentage of Bad Debt in each of the two child nodes of the split (15.92% and 7.4%).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



For example, if the percentages of *Bad Debt* in the two child nodes are required to be approximately 10% in each node, specify **10** instead of the current 15.92 and 7.4, as shown below:

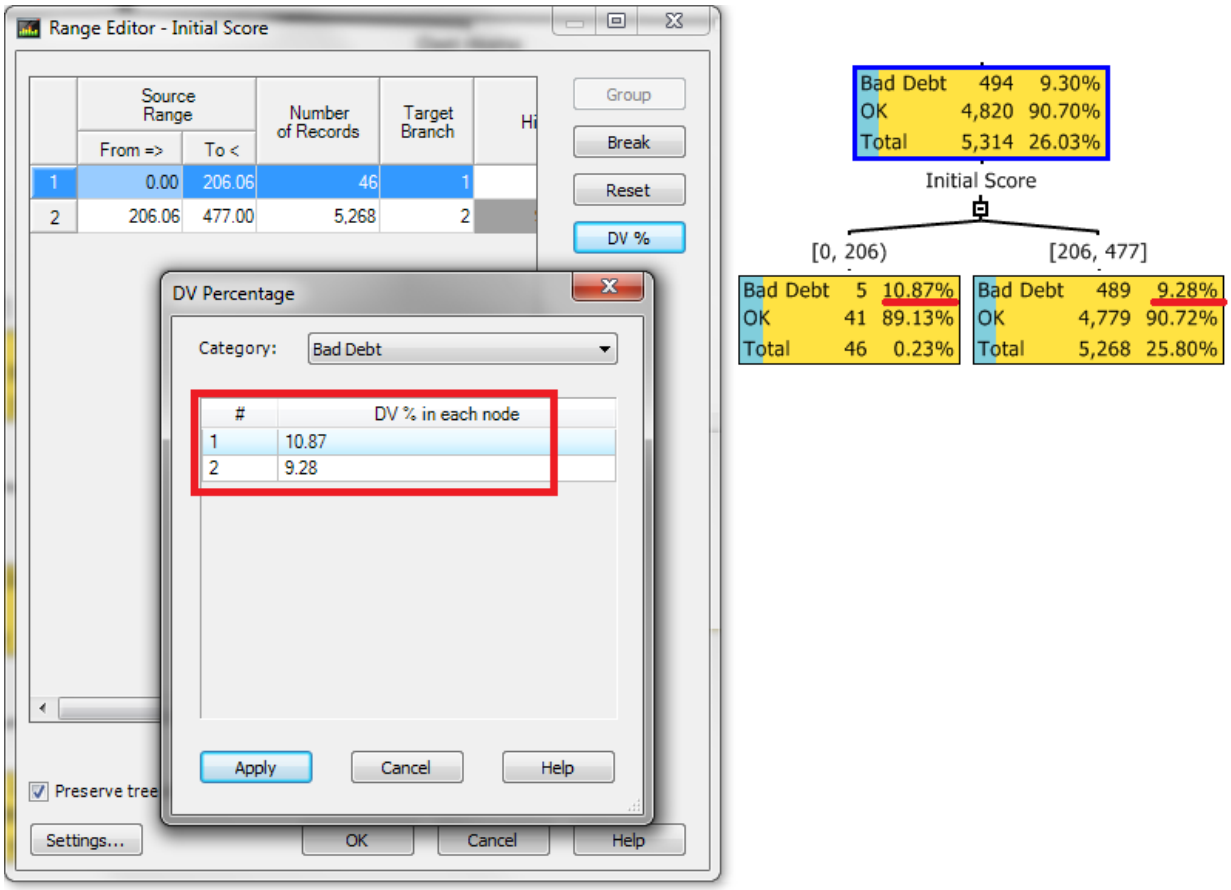
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Apply**. The boundaries of the intervals for the child nodes will be automatically adjusted to satisfy the new DV percentages specified. As shown below, the intervals [0, 300) and [300, 477] become [0, 206.06) and [206.06, 477]. Click OK in the Range Editor.

Note that it is not always possible to set the distribution exactly as specified. In this example, precisely 10% of *Bad Debt* in each child node is not achievable, so the percentages are automatically adjusted to 10.87 and 9.28, as can be seen in the resulting split.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also:

Force Split

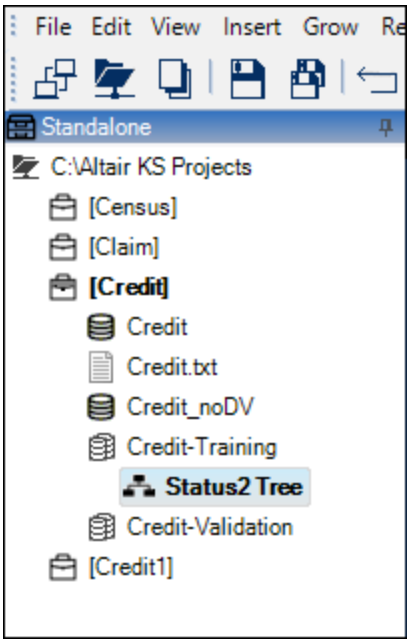
Edit Split

Saving Changes

The changes you made to your decision trees or strategy trees can be saved by selecting the **Save** or **Save All** command from the **File** menu or pressing **Ctrl+S** on your keyboard.

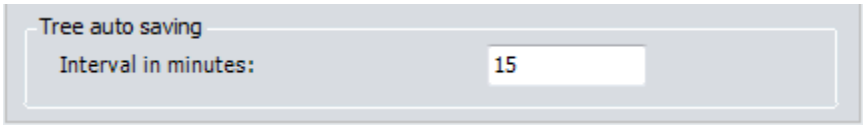
To indicate that a tree has been modified and the changes have not yet been saved, the name of the tree in Project Explorer is displayed in bold font. The names of the projects that have unsaved trees are also displayed in bold. For example, the Status2 tree in the Credit project in this screenshot has been modified and the changes have not been saved yet:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



When closing the tree or project, you will be prompted to save the trees you have modified.

The Auto-save feature is provided. You can set the time interval for auto saving trees using the **Tools | Preferences** menu command (in the ***General*** tab of the Preferences dialog). The default interval is 15 minutes.



Viewing a Decision Tree

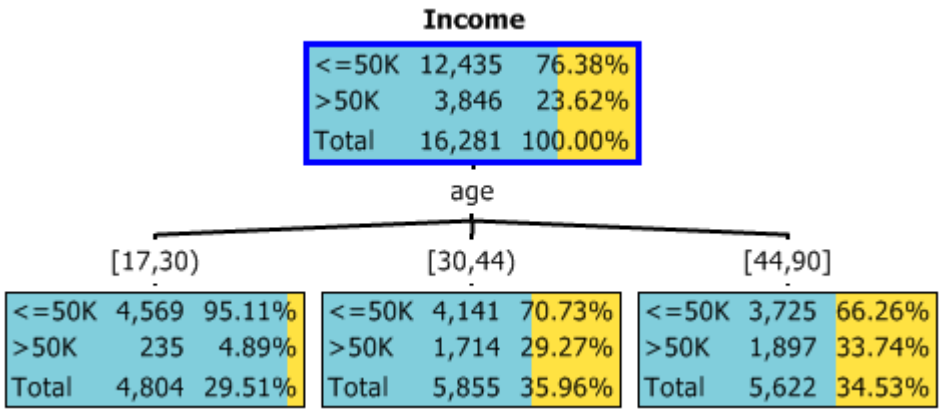
Tree View

For decision trees, this is the main analysis view.

Initially, the tree view displays the root node, which represents all the records in the dataset. From there, records are split into groups called child nodes. These nodes in turn are further split until terminal nodes or leaves - which can no longer be split - are created.

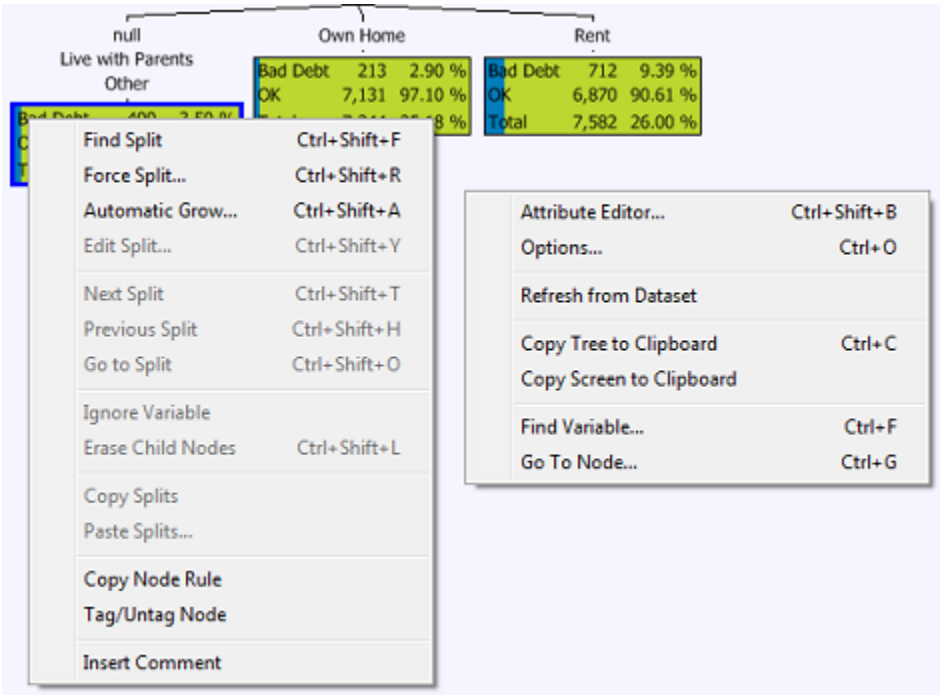
The tree view is interactive - you use it to split nodes one-by-one or to grow the tree automatically and then adjust it.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The name of the dependent variable (DV) is shown as a title at the top of the tree. If you want the DV name to be displayed in every node, use the **Tools | Options** command and select the *Show DV Name* option in the Tree Node tab of the Options dialog. The Options command is also available in the toolbar and in the context menu of the tree background.

Right-click menus in the Tree view are context-sensitive based on whether a node is clicked or the tree background. The context menu for the nodes contains only node-specific commands, such as *Find Split* and *Erase Child Nodes*. The commands applicable to the entire tree, such *Attribute Editor* and *Find Variable,* are in the context menu for the tree background. This also applies to the Tree Map view.



Usually the tree view is used to look for the most relevant (and statistically significant) variables that correlate to the values of the dependent variable. After running the **Find Split** command, the most significant split is displayed first. You can view other splits using the Next Split, Previous Split, and Go To Split commands. As you move down through the tree, you are building a series of rules such as this one:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
If a customer's age is between 17 and 30 then there is a 95.11 percent
chance that Income will be <=50K and a 4.89 percent chance that Income
will be >50K.
```

Tip: You can generate rules (code) of various types for your tree using the **Insert | Code Generation** menu command. See the topic About Code Generation for details.

When you click or hover the mouse pointer over a node, the tooltip displays the node rule. If you need to copy the node rule to the clipboard, just right click the node and select **Copy Node Rule** from the context menu (or select **Edit | Copy Node Rule** from the menu). For example, the rule for the middle child node in the tree above is ([age]>=30 AND [age]<41)

Notice that the further you traverse down the tree, the more you build on the rule set. After just a few splits, you may have a set of rules that classifies records quite tightly.

The primary tree commands are found on the **Grow**, **Reshape**, and **View** menus. The most common commands are also found on the context menu (right-click on a node to bring this menu up).

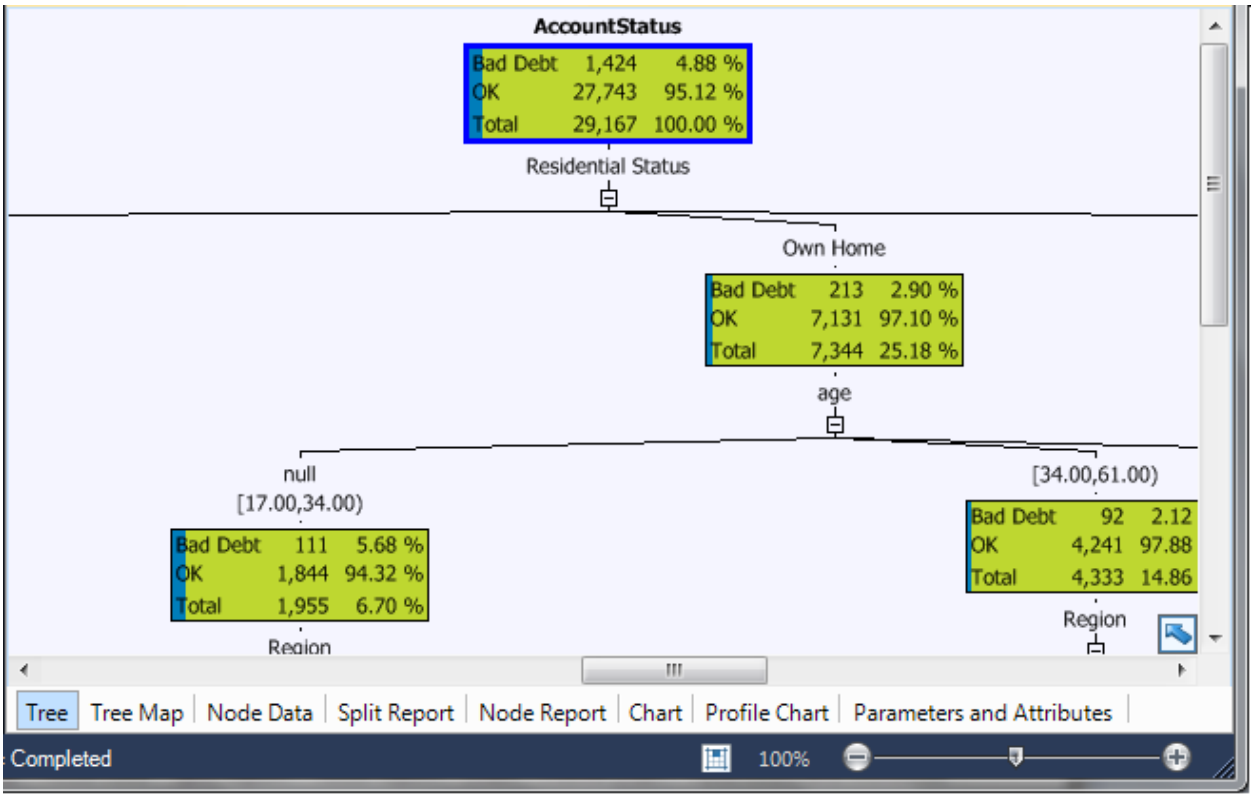
Pan and Zoom

You can easily navigate across large trees using pan and zoom features.

Pan control allows the user to click and drag the tree instead of using the scroll bars. Just click anywhere in the tree view, hold the left mouse button down and start dragging to move around the tree.

The Zoom control allows you to zoom in or out to view more of the tree. A standard zoom control with a slider is provided in the Status bar, at the bottom right corner, as seen in the screenshot below. Aside from zooming by moving the slider and clicking the +/- icons, you can also use mouse wheel scroll with the CTRL button, or various keyboard shortcuts (CTRL-+, CTRL--, CTRL-0).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The zoom commands are listed under the View menu: Zoom In / Zoom Out / Reset (to 100 %). The zoom percentage indicated by the slider position is relative to the current font size.

To the left of the zoom slider, there is a button to re-center the tree at the root node.

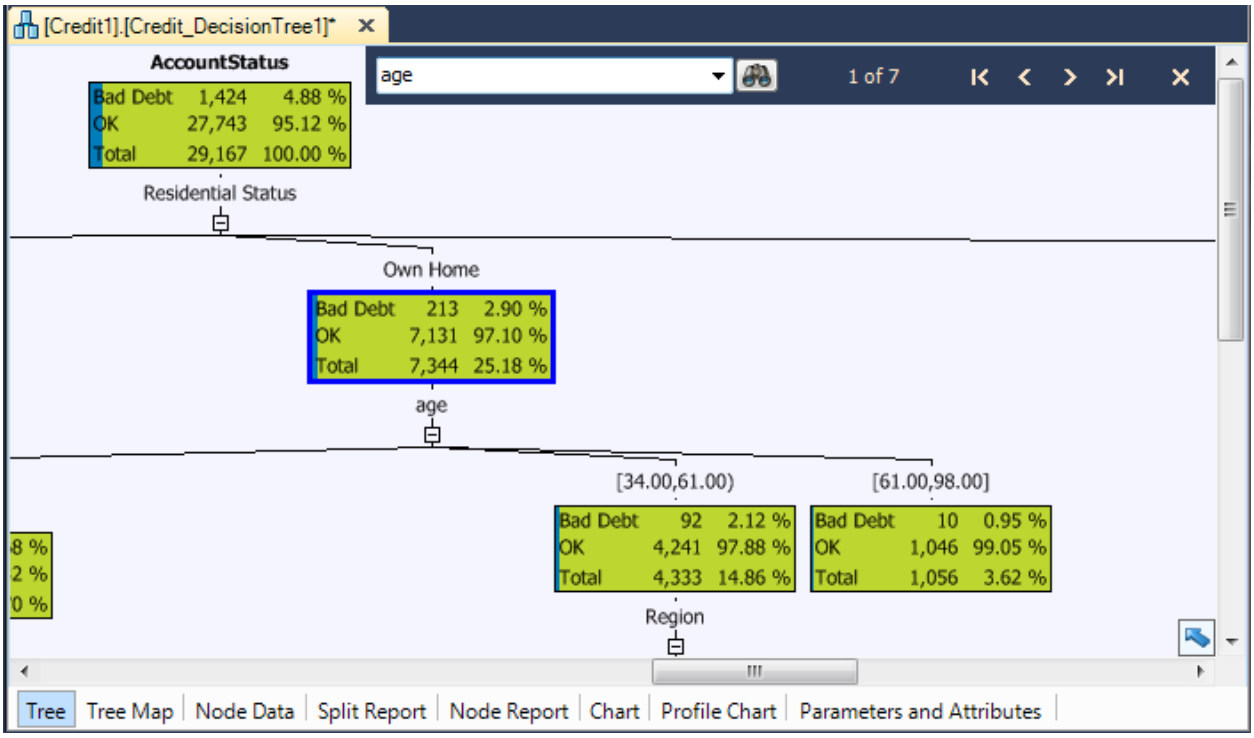
Searching for a Split Variable

The Find Variable command allows you to search for a split variable in the tree and navigate between splits, which is especially convenient in large trees with multiple splits on the same variable.

The Find Variable control can be opened with a CTRL-F keyboard shortcut or via the edit menu. See the screenshot below. You can enter the name of the variable using the keyboard or select it in the drop-down list. The control will automatically go to the first split on that variable, and show how many splits in the tree use that variable. The command finds the variables in the same order as the Node Numbers: from left to right, top to bottom.

The Previous and Next buttons in the form of arrows find the previous and next occurrence of the variable. Pressing Enter while the focus is on the Find control finds the next occurrence of the variable. Pressing F3 when the focus is not on the Find control does the same. Pressing the Escape key closes the Find box.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

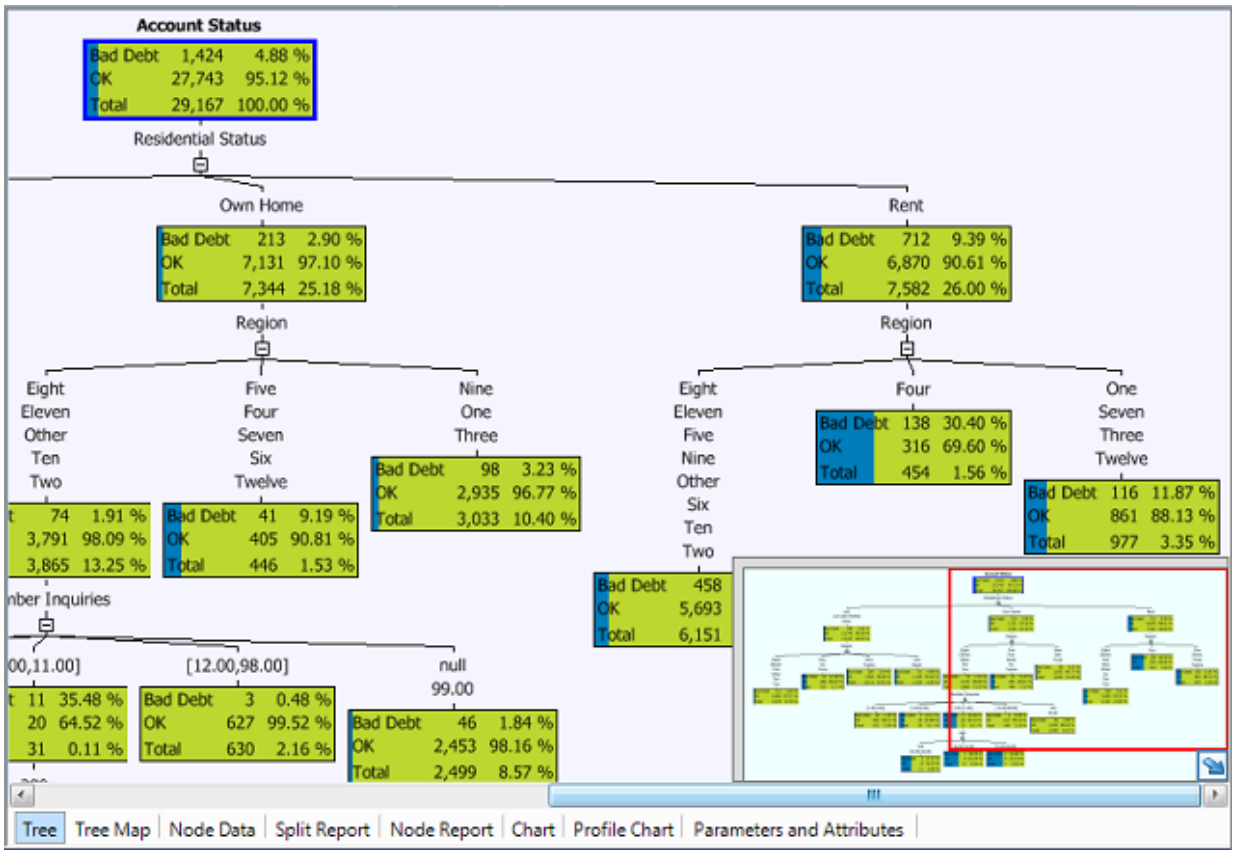


Mini-map

An interactive tree mini-map in the bottom right part of the Tree View shows the location of the current view of the tree relative to the whole tree and allows the user to jump easily to any part of the tree with a single click. The mini-map can be shown or hidden by clicking on the blue arrow in the bottom right corner. You can resize the mini-map by dragging its border or the top left corner.

A transparent rectangle with red border on the mini-map indicates the part of the tree shown in the Tree view. The fastest way to navigate to the desired part of the tree is to click on that part in the mini-map view. The Tree View will be updated immediately to show the part you clicked on.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Tree Options

You can specify various Tree options by clicking the *Options* icon in the toolbar or selecting **Tools | Options** from the main menu. They are applied to the current tree only, while the same settings specified via **Tools | Preferences** are global and will be applied to all *new* trees (not existing ones). See the Preferences section for details. Also, see the subsection "Precision and Format of Numbers Displayed in Trees" provided below.

Node Comments

In addition to node tags, node comments is a convenient way to mark some nodes of your interest and put your comments, explanations, or instructions (for example, specify the way to treat the observations in a particular node).

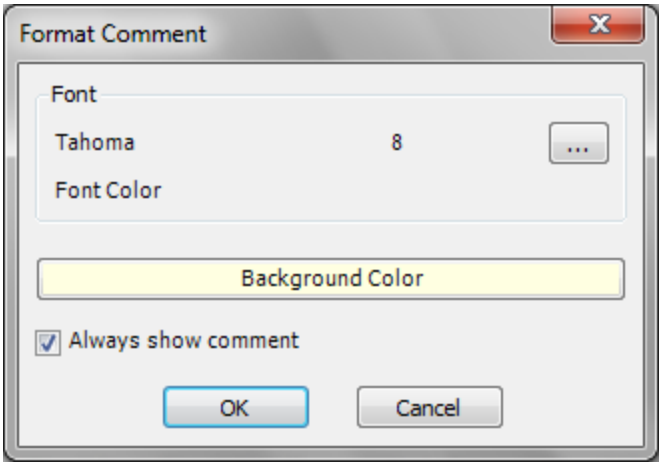
Node comments can be added for presentation purposes if you plan to print or send images of your tree or even share your project.

To insert a node comment, right-click the desired node and select **Insert Comment** from the context menu or select the node and choose **Edit | Insert Comment** from the main menu. Type your text in the comment window and click anywhere outside this window when finished. The comment will be saved with the node. This is indicated by a small red triangle at the top right

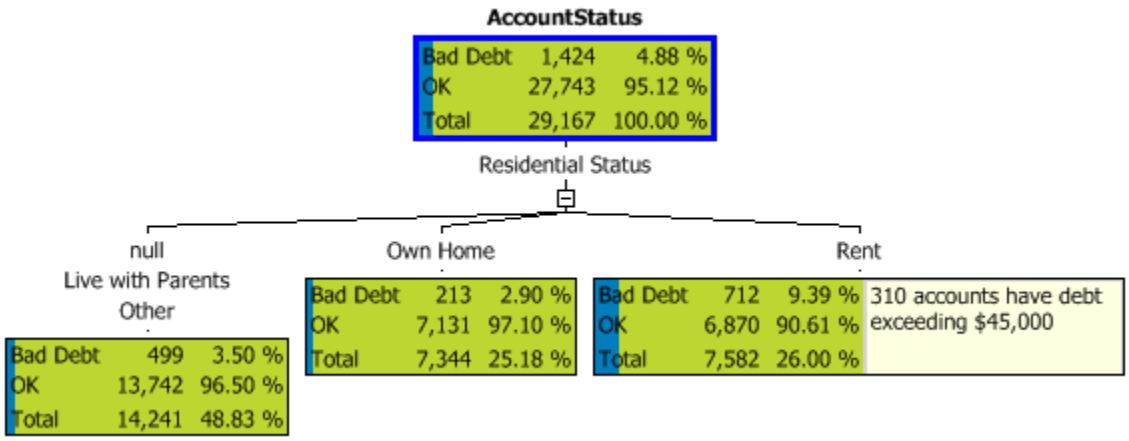
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

corner of the node. Now the commands **Edit Comment**, **Delete Comment**, and **Format Comment** become available in the context menu of the node and under the Edit menu.

The text of the node comment appears as a tooltip when your mouse pointer is over the top right corner of the node. You can also choose to always display the text of the comment next to the node by right-clicking it and selecting **Format Comment** from the context menu.



After you select the checkbox 'Always show comment', the comment will be displayed in a box to the right of the node as in this example:



Precision and Format of Numbers Displayed in Trees

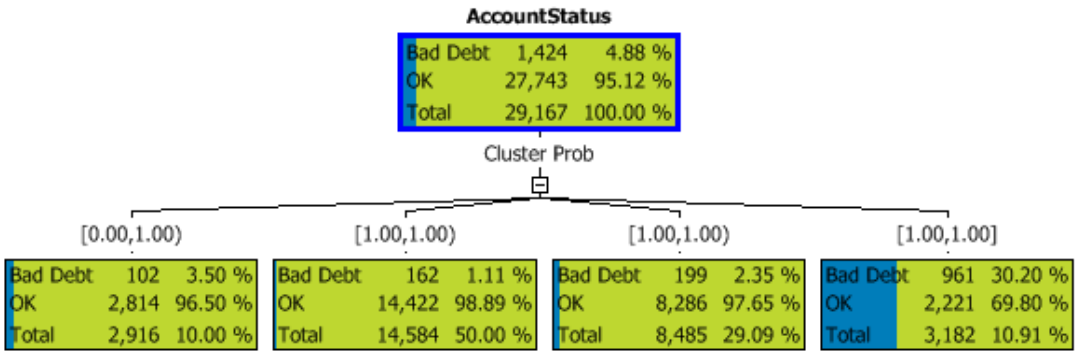
You can adjust the precision and format of all numerical values displayed in trees: value percentages of the discrete dependent variables, average and standard deviation of continuous dependent variables, values of independent variables in the split rules (displayed above the nodes), tree split parameters such as Ratio Entropy, Chi2, P value, etc. (if you chose the option to display them), and even the value frequencies, which may be non-integer if the tree is weighted.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To adjust the precision and format of numbers in the current tree only, use **Tools | Options** and Tree Attribute Editor (**Tools | Attribute Editor**).

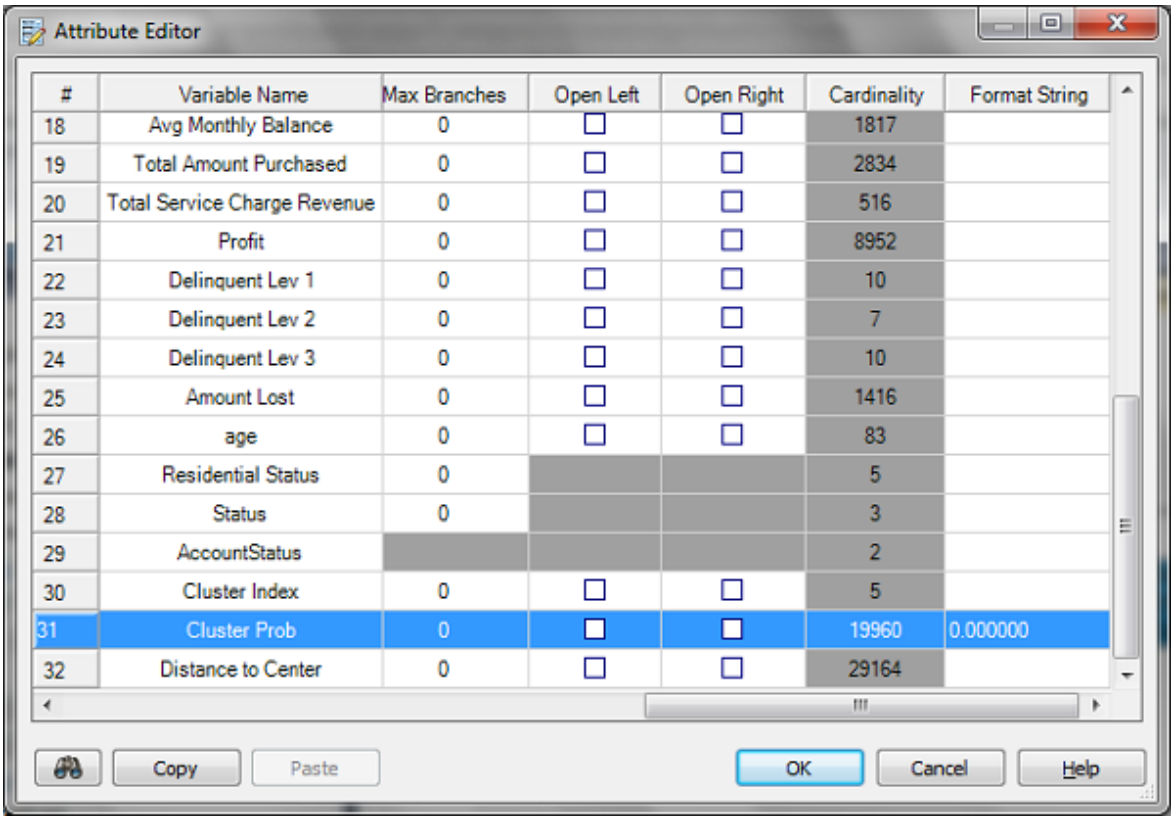
To set global precision and format preferences for all new trees to be created, use **Tools | Preferences** and Dataset Editor. Changes made in Dataset Editor can also be applied to existing trees using the menu command **Tools | Refresh From Dataset**.

To adjust the precision and format for independent variables in the current tree only, use the Tree Attribute Editor (**Tools | Attribute Editor**). For example, in the following split on Cluster Probability, the default precision of two digits after the decimal point is not sufficient, as many values of this variable may be close to 1 (such as 0.99998) and will therefore be displayed as 1.00 after rounding, so some intervals in marginal rules may look like this: [1.00, 1.00):

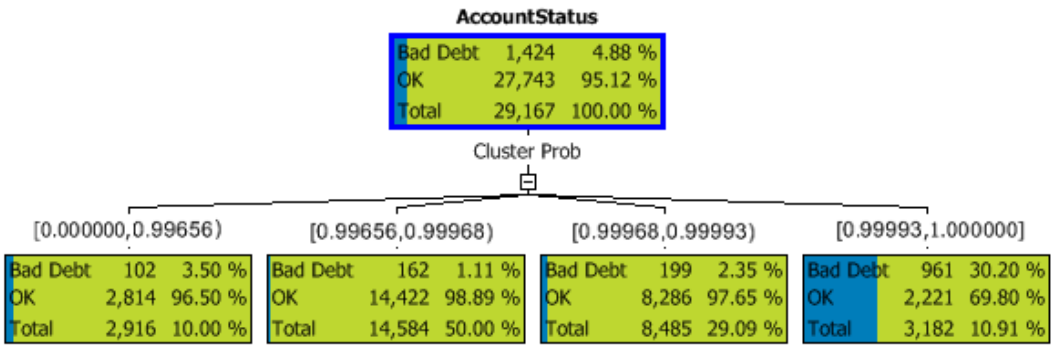


To increase the precision for the *Cluster Prob* variable, open the Tree Attribute Editor by selecting **Tools | Attribute Editor** from the menu, select the row for the Cluster Prob, scroll to the right and enter 0.000000 in the **Format String** field if you want 5 digits after the decimal point. (For details about the ways to specify format strings and precision, see the topic Specifying Format.)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Upon clicking OK in Attribute Editor, the values of the split variable in the tree will be displayed with requested precision:



If you need to change the default precision of an independent variable in all trees created for this dataset, use Dataset Editor instead. Open the parent dataset and select **Tools | Dataset Editor** from the menu. Select the desired variable and click the **Edit** button. Enter 0.00000 or another preferred format string in the Format field. NOTE: The format changes made in Dataset Editor will be applied to all new trees based on this dataset, but if you want them to be applied to an existing tree, you need to open the tree and select **Tools | Refresh From Dataset.**

In the above example, if you set the precision of *Cluster Prob* to 0.00000 in Dataset Editor, the values of *Cluster Prob* will be displayed with 5 digits after the decimal point in all new trees

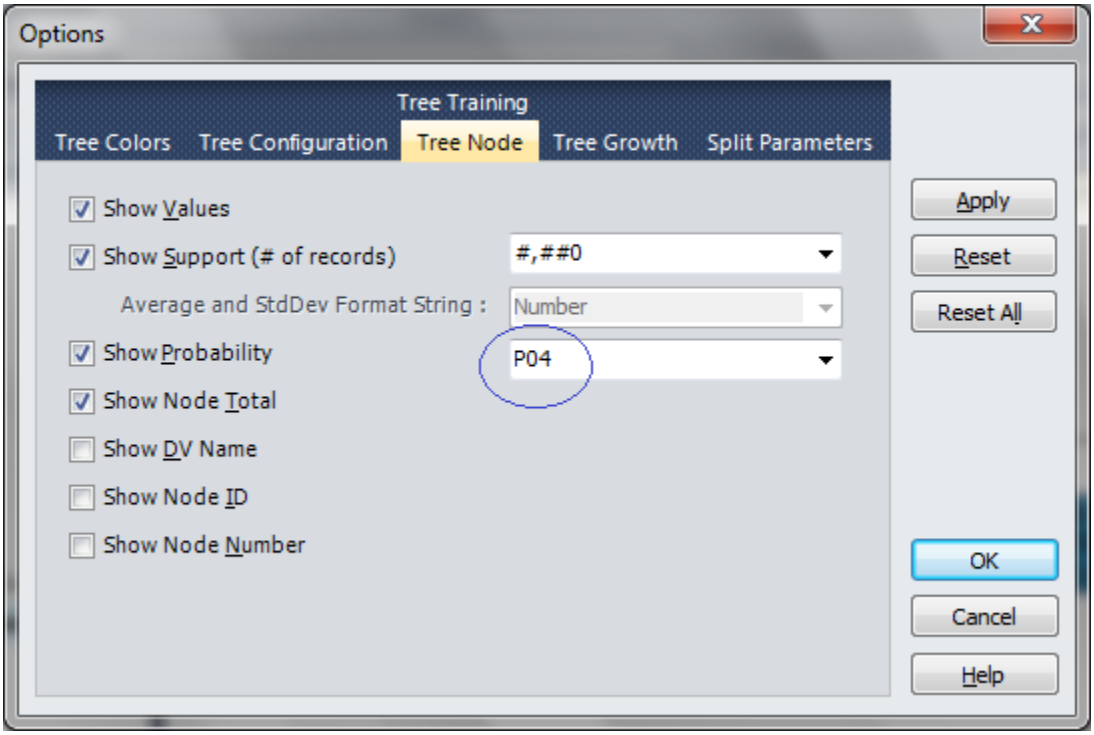
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

based on this dataset and also in the existing trees after you select **Tools | Refresh From Dataset** in the Tree view.

For details about the ways to specify format strings and precision, see the topic Specifying Format.

To change the format and precision of the DV value percentages displayed in the nodes of the current tree, use **Tools | Options** (the Tree Node Settings tab). To change the global default (for all new trees to be created), use **Tools | Preferences** instead

By default, DV percentages in tree nodes are rounded to two decimal places. For example, if you want the percentages of the values "Bad Debt" and "OK" in the above tree to be displayed with four digits after the decimal point, click the Options button in the toolbar. Type "P04" in the Show Probability box. (For details about the ways to specify format strings and precision, see the topic Specifying Format.)



Upon clicking OK, the percentages of the AccountStatus values will be displayed with four digits after the decimal point:

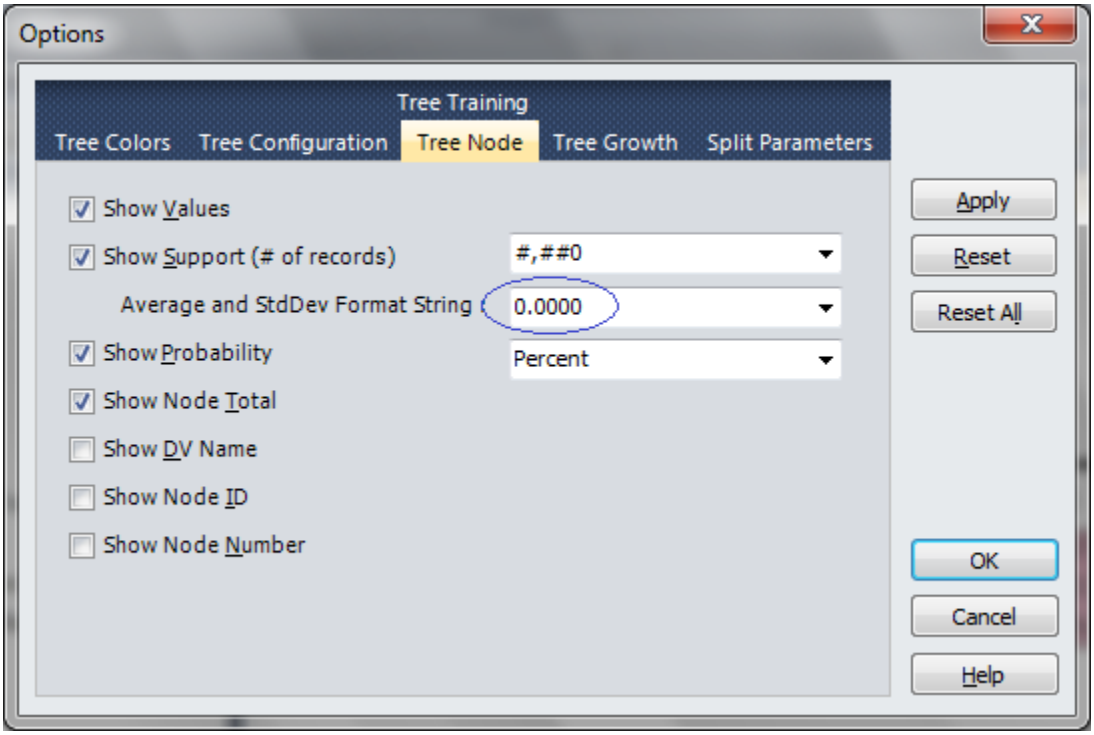
Bad Debt	1,424	4.8822	%
OK	27,743	95.1178	%
Total	29,167	100.0000	%

If the DV is continuous, the default precision of Average and Standard Deviation is two decimal places. For example:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Average		8.76
StdDev		224.10
Total	29,167	100.00 %

If you need to increase the precision of Average and StdDev to 4 decimal places, enter 0.0000 as Average and StdDev Format String in the **Options** dialog. (For this change to affect all new trees with continuous DVs to be created, use **Tools | Preferences**, Tree Node Settings tab.):

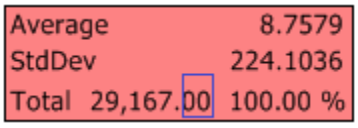


Upon clicking OK, the Average and StdDev for *Profit* will be displayed with four digits after the decimal point:

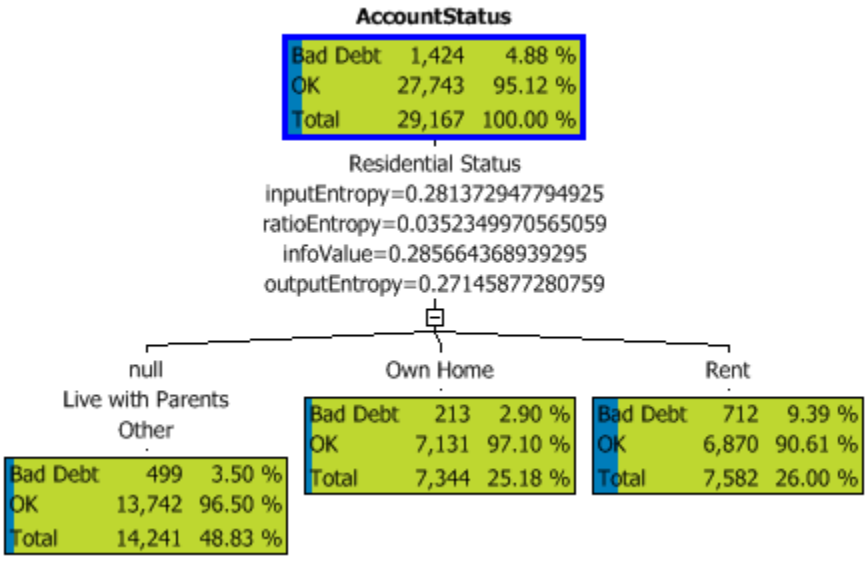
Average		8.7579
StdDev		224.1036
Total	29,167	100.00 %

If your tree is weighted (for details, see the topics Attribute Editor: About Weights and Weight), you may need to see the fractional parts of the node totals. By default, the total number of records in a node is displayed as an integer. To increase its precision, for example, to two decimal places, enter 0.00 in the **Show Support (# of records)** field in the Tree Node Settings tab of the **Options** dialog. If you need the thousands separator, enter #,##0.00. See the topic Specifying Format for information about format strings.

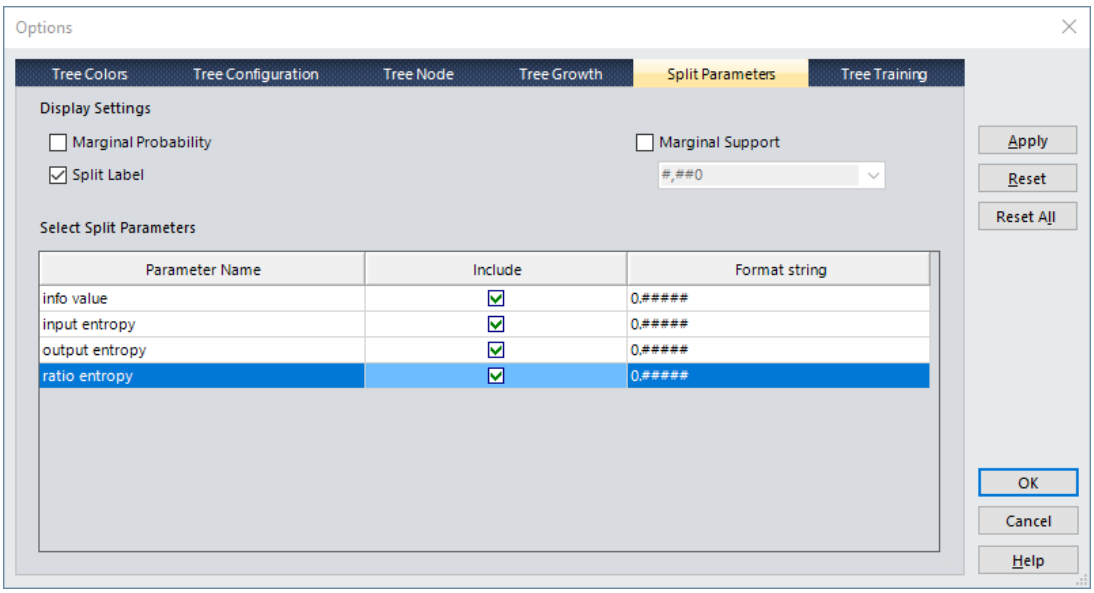
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



By default, the Split Parameters such as Ratio Entropy are displayed with up to 17 digits after the decimal point. For example:

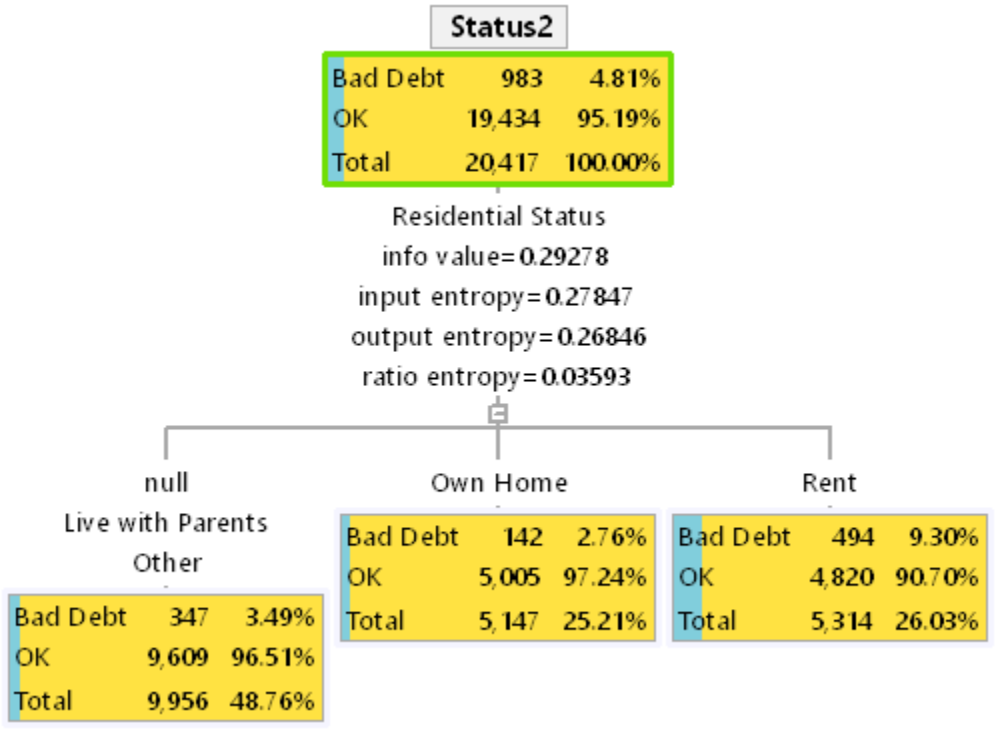


If you want them to be displayed with lower precision, select the Split Parameters tab of the **Tools | Options** dialog (or the **Tools | Preferences** dialog if you want the change to be applied to all trees to be created in the future) and enter the desired precision in the Format String field. For example, with the following settings, Info Value will be shown with three decimal places, while Input, Output, and Ratio Entropy will be shown with five:



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Upon clicking OK, the tree becomes as follows:



For the detailed information about the split parameters, see the Decision Tree section of Mathematical Formulations.

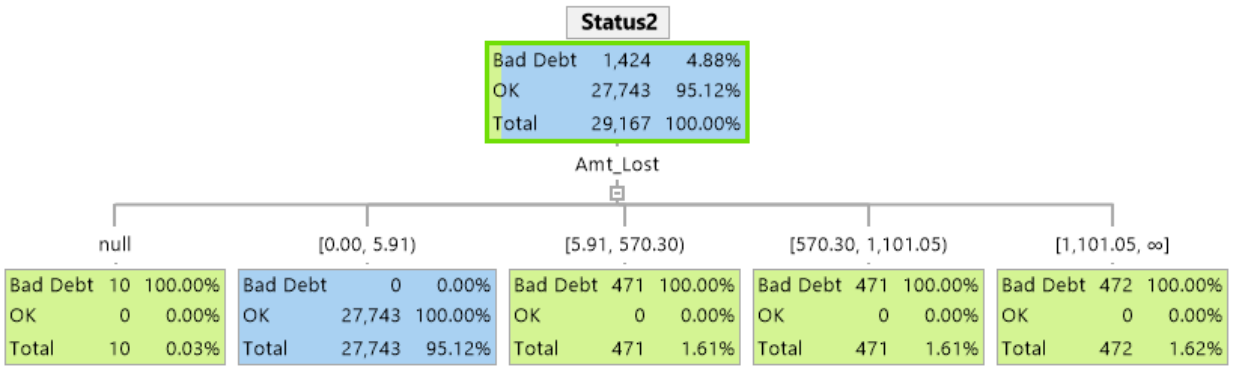
Singular Values in Continuous Independent Variables (infinity, missing values)

Continuous independent variables can contain missing values and singular values such as **infinity** and **NaN** (not a number). For details about infinity {INF} and NaN, see Division and Infinity.

If the variable contains missing values, they are shown in splits as **null** and either placed in a separate bin or grouped with other values according to the settings in the Missing Values column of Attribute Editor. NaN are treated the same way as missing values.

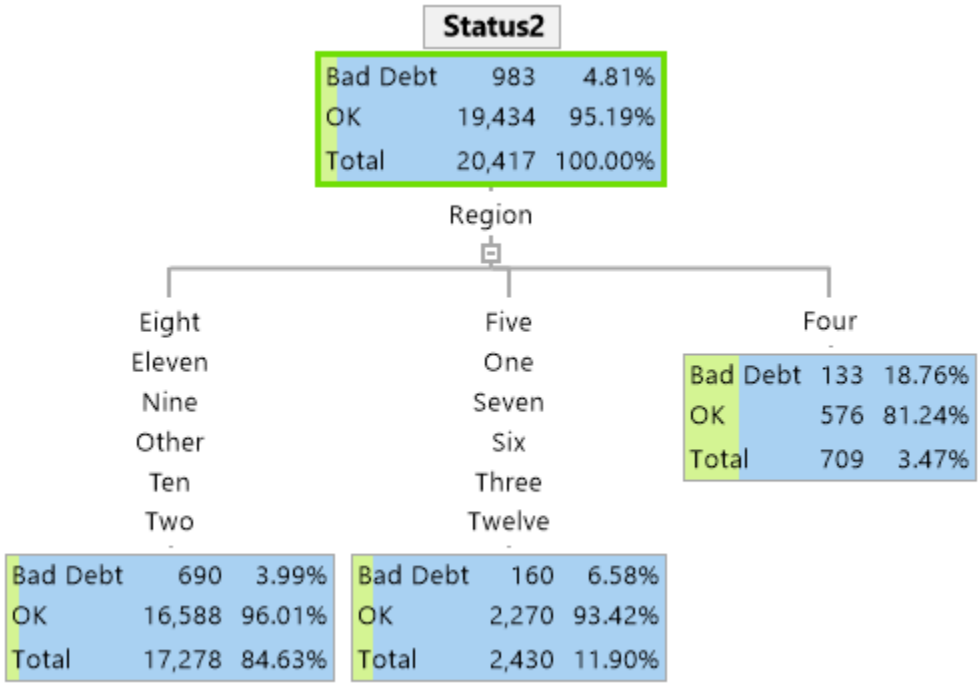
If the variable contains positive infinity {INF}, it is naturally considered to be the maximum value in the variable and is shown as the right boundary of the last interval. Similarly, negative infinity {INF} is naturally considered to be the minumum value in the variable and is shown as the left boundary of the first interval. In this example, the variable Amt_Lost contains both missing values and infnity values.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



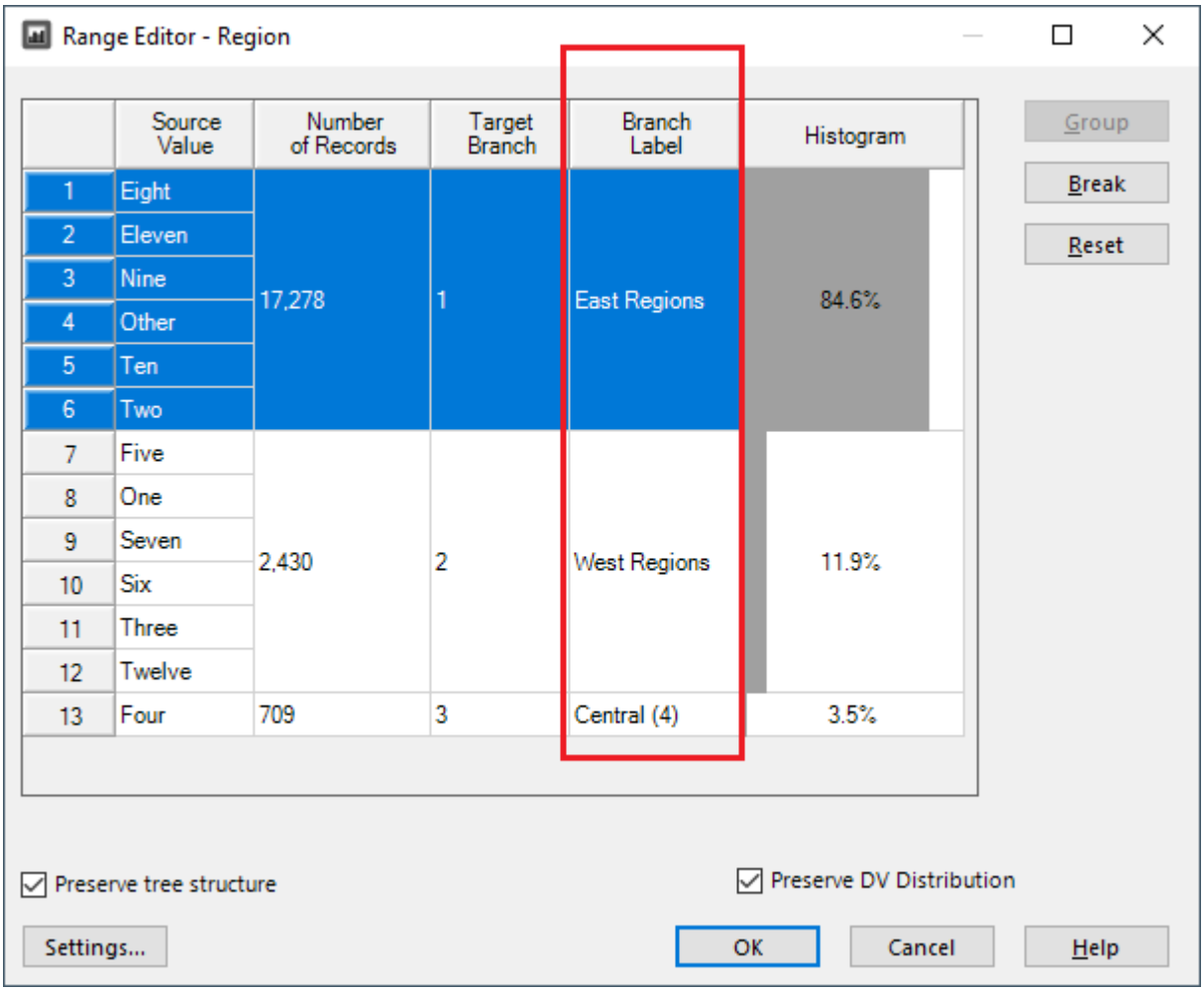
Branch Labels in Splits

When editing or forcing a split, after grouping a large number of discrete values of a nominal variable, you may want to label such groups and display the labels instead of stacked values that sometimes take too much space in the tree view. In the following example of a split on region codes, the two child nodes on the left have marginal rules with stacked values that take too much space:

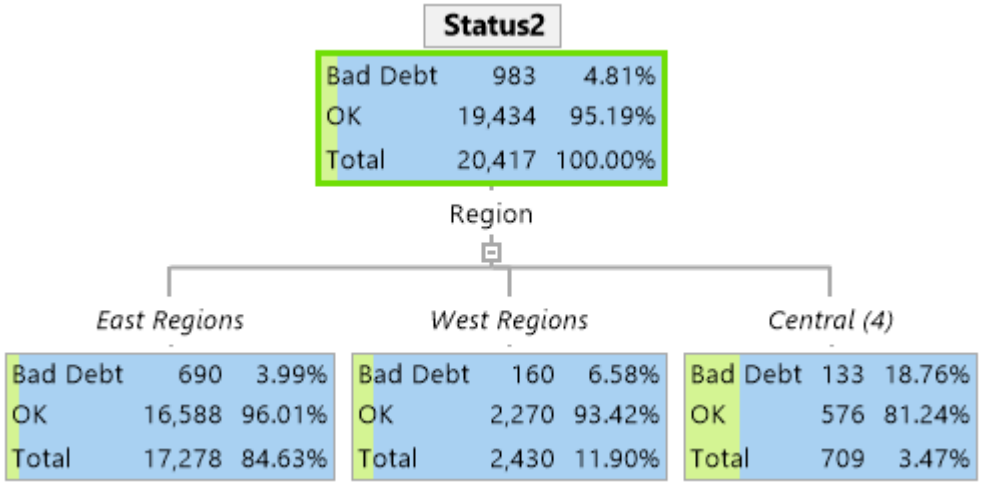


Using the Edit Split command, you can label these groups in the Range Editor dialog by entering the desired labels in the **Branch Label** column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



As a result, the split looks much more compact. The labels are dispayed in *cursive font* to distinguish them from regular individual values of the variable.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

See Also

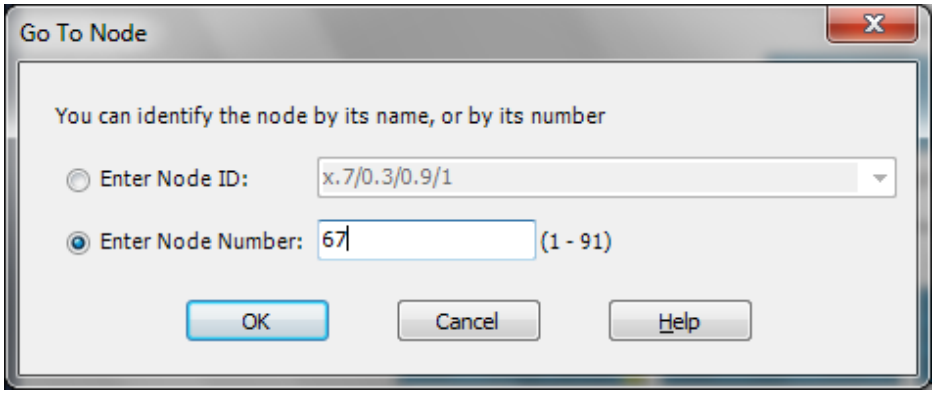
Tree Map view
Node Data view
Split Report view
Node Report view
Profile Chart
Find Split
Force Split
Edit Split

Go To Node

If you enabled the display of Node Numbers or Node IDs in your tree (in the *Tree Node Settings* tab of the **Options** or **Preferences** dialog), you may need to locate a specific node by its number or ID.

(For example, suppose you scored a tree model and included the Node IDs or Node Number column in the scoring result. Then at some point you need to know which tree node a certain record was assigned to.)

For this purpose, use the **Go To Node** command under the **Edit** menu. The same command is invoked by the keyboard shortcut **Ctrl+G**.

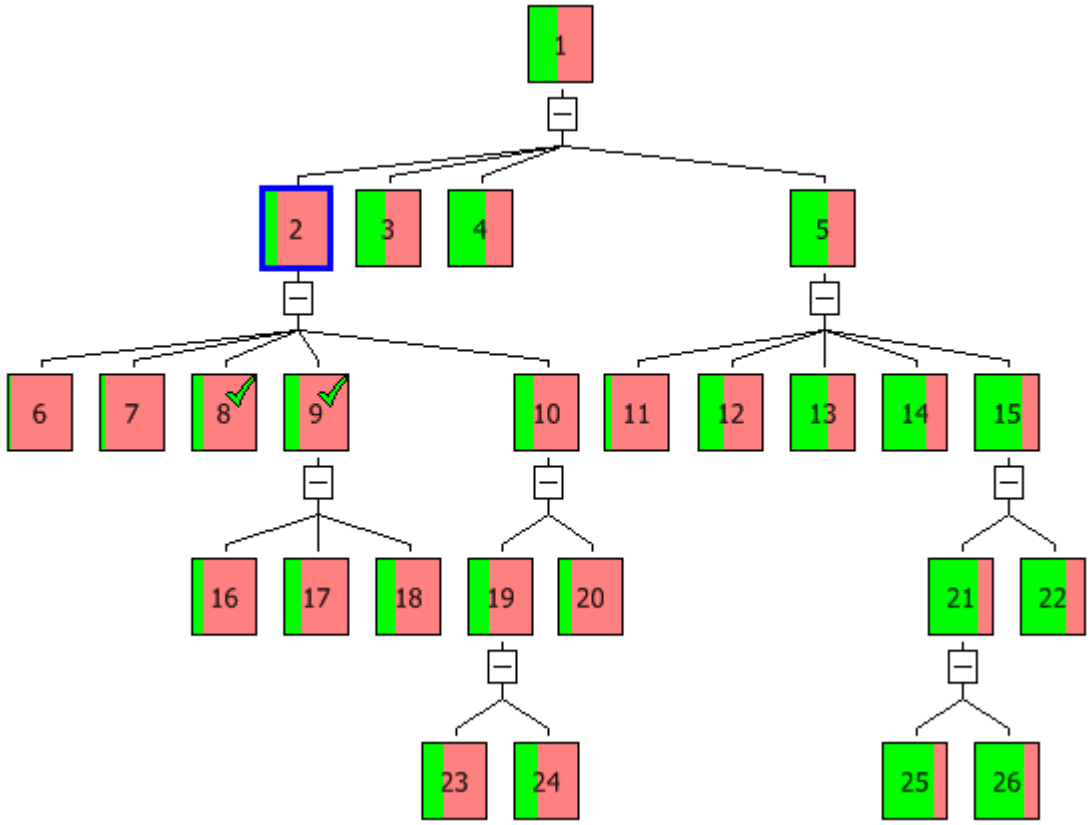


For more on Tree Node Settings and other display options, see the topic Preferences: Tree Node Settings.

Tree Map View

The **Tree Map** view displays the tree without the detailed information for each node. This view is good for navigating through a large tree. You can perform all the functions that are available for the **Tree** view.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See Also

Tree view
Node Data view
Split Report view
Node Report view
Find Split
Force Split
Edit Split

Node Data View

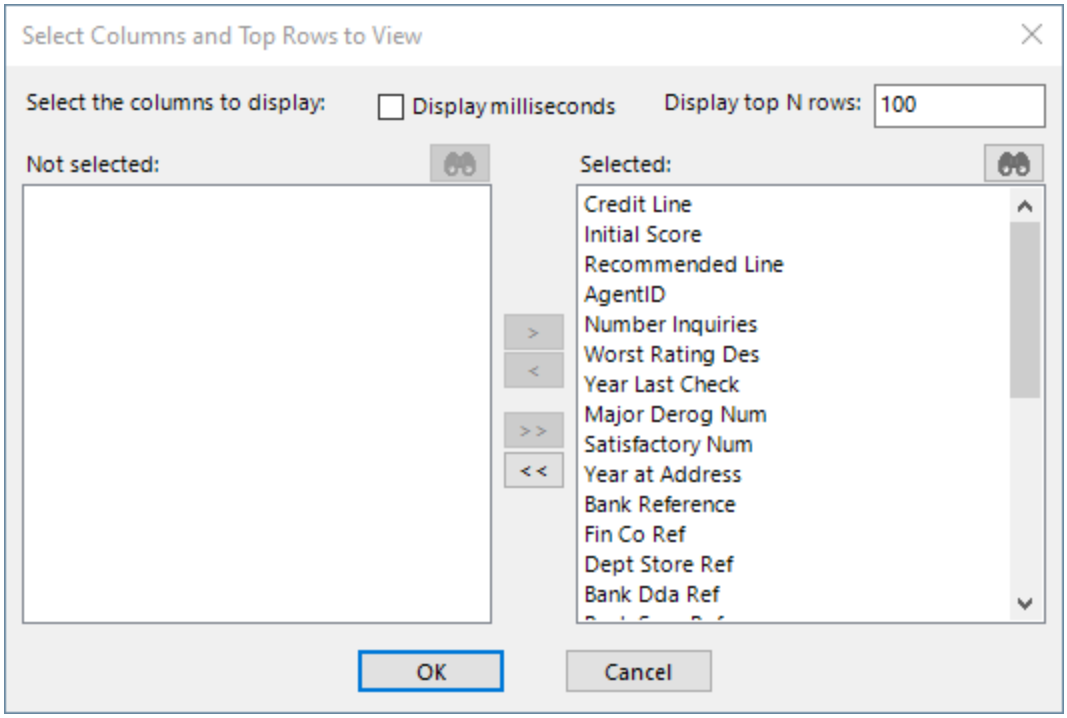
The **Node Data** view displays the data contained within the currently active tree node. The node is considered active when it is highlighted by a mouse click while in the Tree View or Tree Map view.

Missing values are displayed as `null`

The first time you click on the Node Data tab, you are prompted to select the variables (columns) to be displayed. All variables are selected by default.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

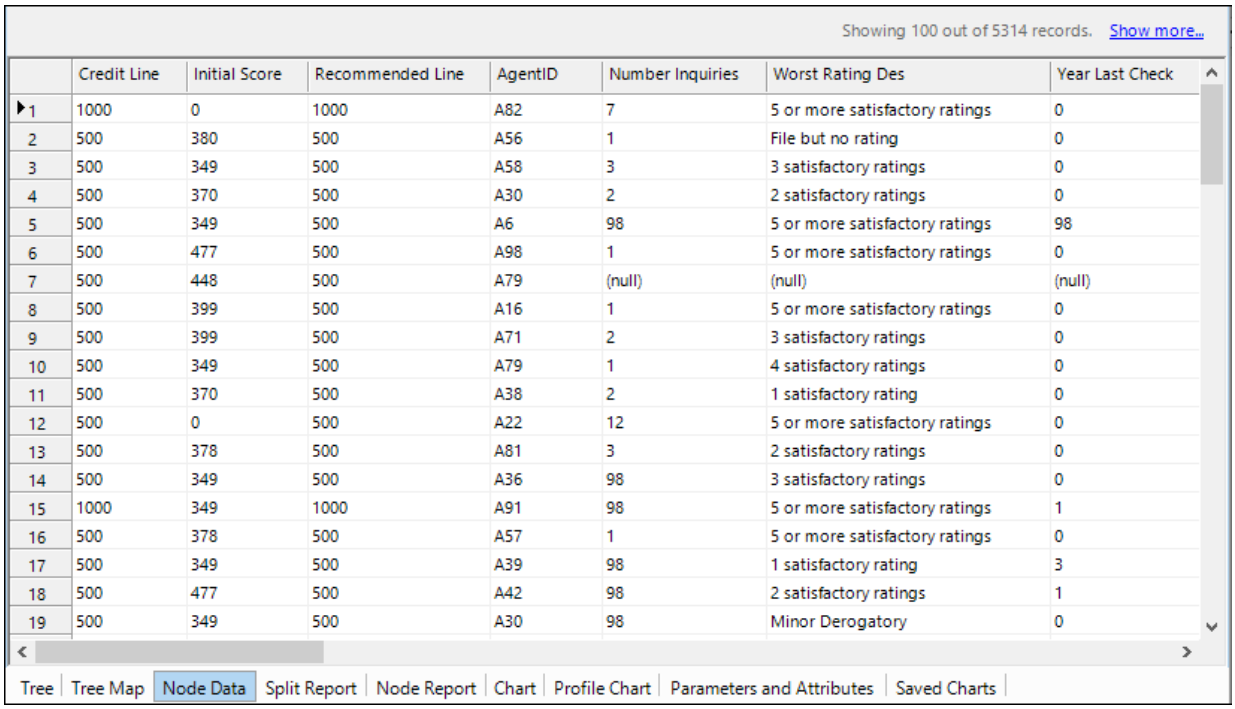
- Select or deselect the fields by clicking on the arrow buttons or double-clicking on the variable names.
- Specify the number of top rows in the dataset to display by default. Top 100 rows is the default. You will be able to load more rows if necessary.
- If your source data contains Time or Timestamp values with milliseconds, select the option ***Display milliseconds***.



Click the **Options** button in the toolbar to exclude or include any variables or display milliseconds for Time or Timestamp values, if any.

Click **Show more..** at the top right to load more rows if necessary.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Showing 100 out of 5314 records. Show more...

	Credit Line	Initial Score	Recommended Line	AgentID	Number Inquiries	Worst Rating Des	Year Last Check
1	1000	0	1000	A82	7	5 or more satisfactory ratings	0
2	500	380	500	A56	1	File but no rating	0
3	500	349	500	A58	3	3 satisfactory ratings	0
4	500	370	500	A30	2	2 satisfactory ratings	0
5	500	349	500	A6	98	5 or more satisfactory ratings	98
6	500	477	500	A98	1	5 or more satisfactory ratings	0
7	500	448	500	A79	(null)	(null)	(null)
8	500	399	500	A16	1	5 or more satisfactory ratings	0
9	500	399	500	A71	2	3 satisfactory ratings	0
10	500	349	500	A79	1	4 satisfactory ratings	0
11	500	370	500	A38	2	1 satisfactory rating	0
12	500	0	500	A22	12	5 or more satisfactory ratings	0
13	500	378	500	A81	3	2 satisfactory ratings	0
14	500	349	500	A36	98	3 satisfactory ratings	0
15	1000	349	1000	A91	98	5 or more satisfactory ratings	1
16	500	378	500	A57	1	5 or more satisfactory ratings	0
17	500	349	500	A39	98	1 satisfactory rating	3
18	500	477	500	A42	98	2 satisfactory ratings	1
19	500	349	500	A30	98	Minor Derogatory	0

Tree | Tree Map | Node Data | Split Report | Node Report | Chart | Profile Chart | Parameters and Attributes | Saved Charts

The data cells are not editable in this grid, but the data can be copied and pasted into Microsoft Office applications. Drag the mouse down the grid or use the CTRL + mouseclick to select the desired rows, then use the right-click context menu on the grid to copy the rows. You can also use the **Edit** menu commands **Select All** and **Copy.**

To paste the data into a Microsoft Office application such as Excel, open the application and paste the data from the clipboard.

See also:

- Node Datasets
- Tree View
- Node Data View
- Node Report View

Split Report

The **Split Report** displays all splits at the currently selected node of the decision tree. The default order is the order of the variable significance with respect to the dependent variable, depending on the measure. Although only the most statistically significant split is displayed in the tree, all statistically significant splits are calculated. The split report shows all the calculated splits in rank order. Variables with a significance less than the filter threshold will be displayed but will not have statistics computed. To adjust the filter threshold, select the menu command **Tools | Options** when Split Report is displayed.

Example 1: Split Report for the tree with the *Entropy Variance* measure (see Measure).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Variable Name	Rank	Info	info value	input entropy	output entropy	ratio entropy
relationship	1	21.00091	1.51571	0.78871	0.62307	0.21001
marital-status	2	20.02513	1.36872	0.78871	0.63077	0.20025
age	3	11.5691	1.07773	0.78871	0.69746	0.11569
capital-gain	4	11.52747		0.78871	0.69779	0.11527
occupation	5	11.4599	0.75986	0.78871	0.69832	0.1146
education-num	6	11.28876	0.71556	0.78871	0.69967	0.11289
education	7	11.28876	0.71556	0.78871	0.69967	0.11289
hours-per-week	8	7.12078	0.46187	0.78871	0.73255	0.07121
sex	9	4.52947	0.29493	0.78871	0.75298	0.04529
workclass	10	3.04825	0.18992	0.78871	0.76467	0.03048
capital-loss	11	2.81085	0.17844	0.78871	0.76654	0.02811
native-country	12	1.06576	0.07839	0.78871	0.7803	0.01066
fnlwgt	13	0.02743	0.00168	0.78871	0.78849	0.00027
Income						

Tree | Tree Map | Node Data | Split Report | Node Report | Chart | Profile Chart | Parameters and Attributes | Saved Charts

Example 2: Split Report for the tree with the *Bonferoni-adjusted log p-value measure* (see Measure).

Variable Name	Rank	-log(adj p)	info value	p-value	log p-value	adj p-value	log adj p-value	Bonferroni	chi-square	df	confidence	adj confidence
Amount Lost	1	14552.39674		0	-14552.39674	0	-14552.39674	1	29167	9	1	1
Delinquent Lev 3	2	12874.77085	9.78691	0	-12879.20166	0	-12874.77085	84	25794.87648	6	1	1
Profit	3	6798.25808		0	-6798.25808	0	-6798.25808	1	13653.40989	9	1	1
Avg Monthly Balance	4	2153.26048	3.59698	0	-2153.26048	0	-2153.26048	1	4355.41891	9	1	1
Total Service Charge Revenue	5	1425.08633	2.68469	0	-1425.08633	0	-1425.08633	1	2896.21611	9	1	1
Total Amount Purchased	6	505.36362	0.90055	0	-505.36362	0	-505.36362	1	1049.67474	9	1	1
Region	7	249.78371	0.21928	0	-253.11592	0	-249.78371	28	527.15813	6	1	1
Residential Status	8	220.79448	0.28591	0	-220.79448	0	-220.79448	1	452.44079	4	1	1
Number Inquiries	9	198.27053		0	-217.19133	0	-198.27053	164891025	498.11012	17	1	1
Initial Score	10	167.16519	0.31246	0	-167.16519	0	-167.16519	1	365.92632	9	1	1
Worst Rating Des	11	105.28434	0.13185	0	-105.28434	0	-105.28434	1	239.20945	9	1	1
age	12	92.89778	0.19174	0	-92.89778	0	-92.89778	1	213.6525	9	1	1
Credit Line	13	88.07803		0	-114.63506	0	-88.07803	341643774795	275.29649	14	1	1
Year Last Check	14	80.80524		0	-89.69538	0	-80.80524	7260	200.06643	7	1	1
Credit Card Ref	15	72.83179	0.09706	0	-74.44122	0	-72.83179	5	148.88245	2	1	1
Satisfactory Num	16	66.44032		0	-86.33163	0	-66.44032	435197490	230.10848	19	1	1
Year at Address	17	65.37857	0.11357	0	-65.37857	0	-65.37857	1	153.28741	8	1	1
Recommended Line	18	44.64437	0.08849	0	-53.73311	0	-44.64437	8855	115.61475	4	1	1
Bank Reference	19	42.55021	0.06631	0	-44.49612	0	-42.55021	7	84.0857	1	1	1
Delinquent Lev 2	20	42.05745		0	-44.7655	0	-42.05745	15	97.34184	4	1	1
Dept Store Ref	21	33.72283		0	-33.72283	0	-33.72283	1	71.28829	3	1	1
Delinquent Lev 1	22	29.79639		0	-34.2272	0	-29.79639	84	82.02092	6	1	1
Bank Save Ref	23	16.93505	0.02383	0	-16.93505	0	-16.93505	1	33.8701	2	1	1
Bank Dda Ref	24	8.56744	0.01216	0.00004	-10.17688	0.00019	-8.56744	5	20.35375	2	0.99996	0.99981
Major Derog Num	25	5.49959		0.00002	-11.02901	0.00409	-5.49959	252	24.89717	3	0.99998	0.99591
AgentID	26	0.44161	0.03445	0.643	-0.44161	0.643	-0.44161	1	44.82468	49	0.357	0.357
Fin Co Ref	27	0.40831	0.00108	0.22159	-1.50692	0.66477	-0.40831	3	1.49404	1	0.77841	0.33523
Status												
Account_Status												

Tree | Tree Map | Node Data | Split Report | Node Report | Chart | Profile Chart | Parameters and Attributes | Saved Charts

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

As you navigate the nodes in the other tree views, the split report is automatically updated.

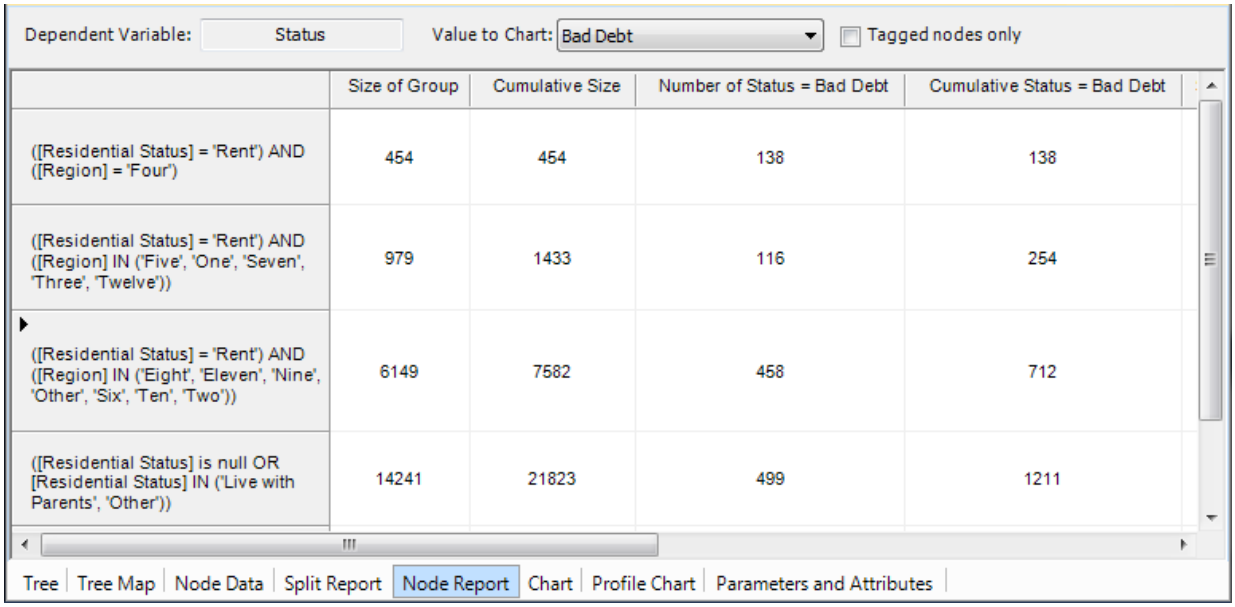
Split Report also displays statistics for each split. The type of statistics that are displayed in this report is dependent on the measure used when creating the decision tree. The splits in Split Report are ranked by significance, which is expressed by the *Info* parameter.

See Also

Tree view
Tree Map view
Node Data view
Node Report view

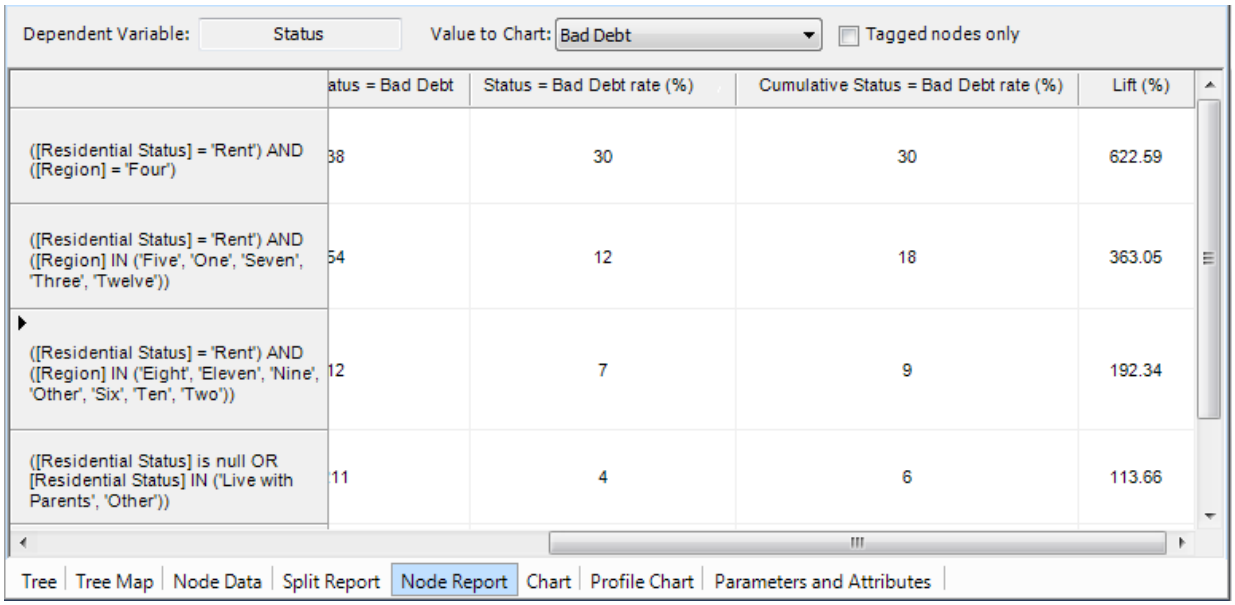
Node Report

The Node Report view provides you with information about the terminal nodes in the tree.



Node Rules	The rule for the terminal node in the decision tree.
Size of group	The number of observations in a given node.
Cumulative Size	The cumulative number of observations up to and including this node.
Number of <DV> = <DV Value>	The number of observations of this category of the dependent variable in this node.
Cumulative <DV> = <DV Value>	The cumulative number of observations of this category up to and including this node

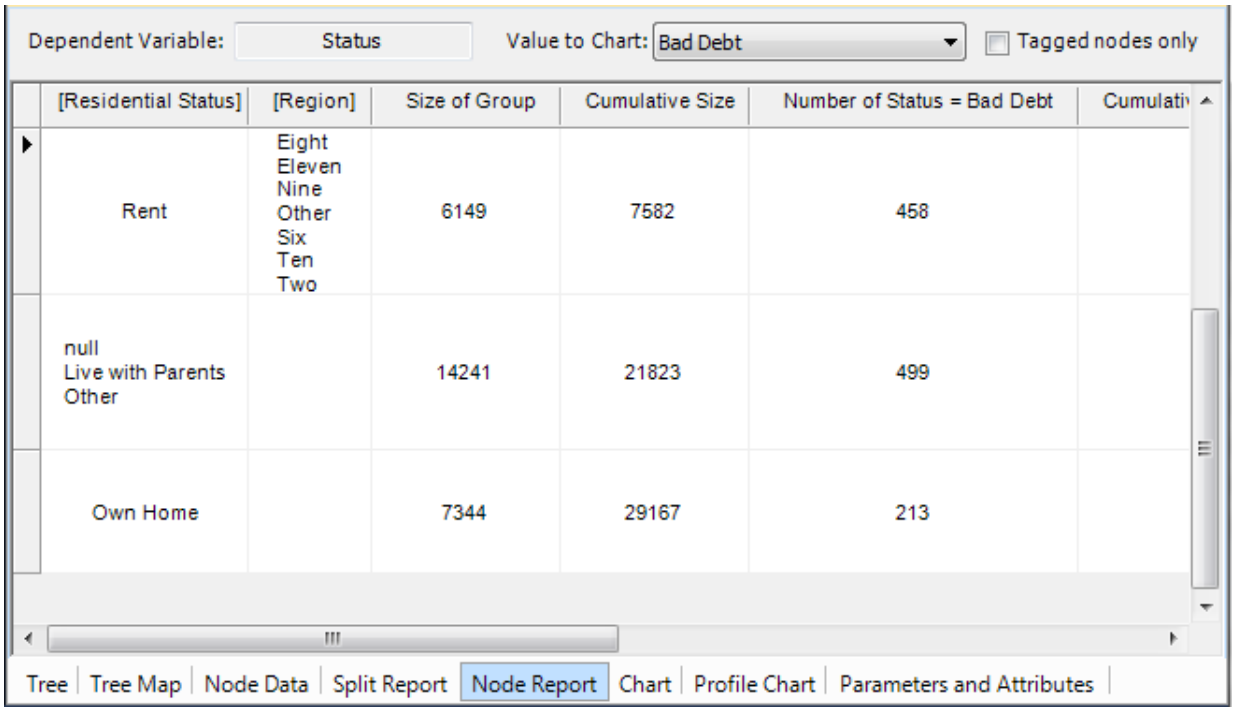
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



<DV> = <DV Value> Rate (%)	Ratio of the number of observations of the dependent variable category (Number of <DV>=<DV Value>) over the total number of observations in the node (Size of group).
Cumulative <DV> = <DV Value> Rate (%)	Ratio of the cumulative number of observations (cumulative <DV>=<DV Value>) over the cumulative number of observations up to and including this node (cumulative size).
Lift	The cumulative ratio of selected categories moving top-to-bottom, from node to node. It is computed relative to the overall percentage that the selected category represents in the entire dataset

Node Report also includes an option to display node rules with a column per Independent Variable (see Node Report Options below). In this mode, each independent variable occurring in at least one node rule is represented by a column in the node report. For any given rule (row), if a condition for the variable occurs in the rule, this condition (in the form of a value, a set of values, or an interval) is displayed in the column representing this variable.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

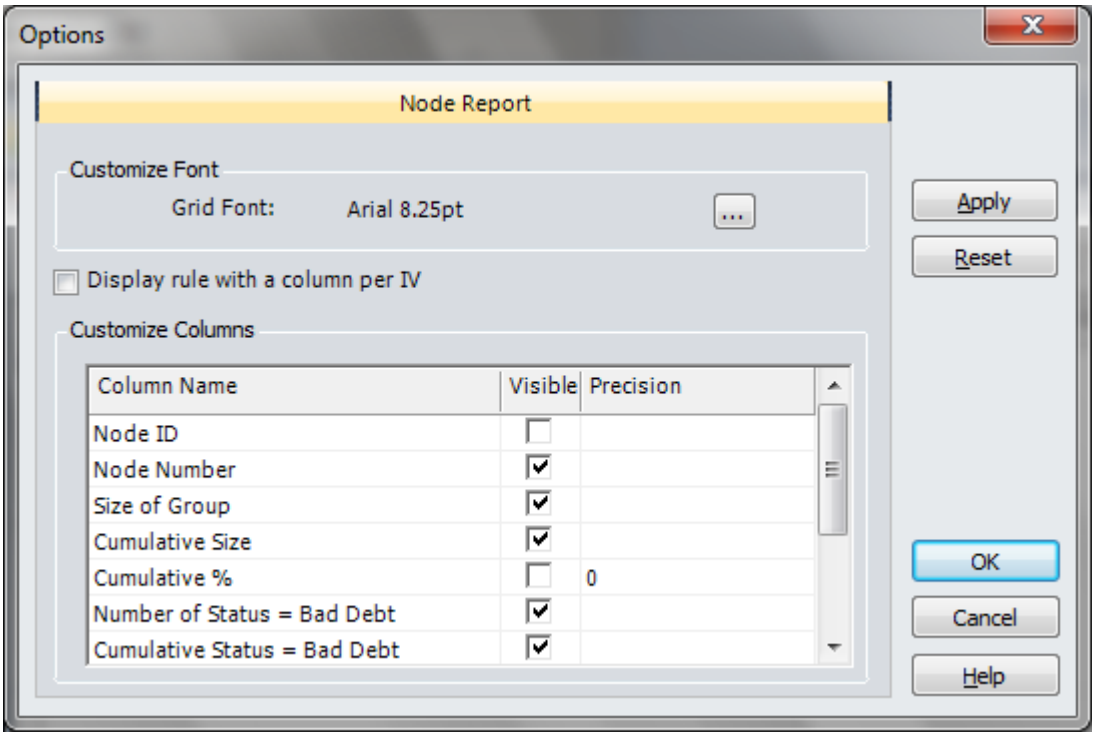


Node Report Options

To change the Node Report display options, click the 

icon in the toolbar or select **Tools | Options** from the menu. You can select columns to display or hide, specify precision for columns, change the font, and choose to display rules with a column per IV:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See Also

Preferences: Node Report
Tree view
Tree Map view
Node Data view
Split Report view

Node Chart

Node Chart

The Node Chart (the **Chart** tab of the Tree object) helps you to visualize the distributions of variables within a tree node.

The chart look and functionality is exactly the same as in the Dataset Chart (for the Single Variable option) or Cross Tabs (for the Cross Tab option), except that it shows the variable distributions in a single tree node rather than the whole dataset. Refer to the Dataset Chart topic for details.

One of the most convenient features of the tree chart is that it can be dynamic: you can traverse the tree clicking on different nodes and at the same time view the chart for the currently selected node.

Note that you can view the tree and the chart side-by-side by docking the Chart view next to the Tree view. To this end, click on the Chart tab name and drag the tab to the desired position on the screen.

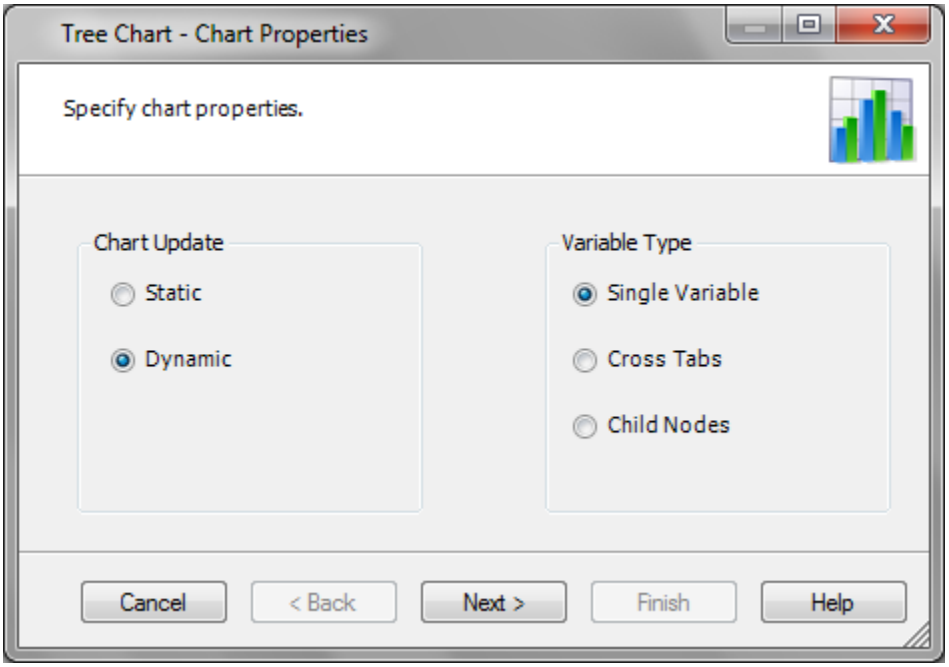
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The first time you click the **Chart** tab in a tree, the **Add Chart** wizard appears automatically. If it does not start automatically, then:

- Click the button 

in the toolbar, or
- Select **Tools | Add Chart** from the menu.

In the first page of the Add Chart Wizard, choose the Chart Properties.



With regard to the update type, the chart can be Static or Dynamic.

If you choose the **Dynamic** option (default), the Chart tab will be dynamically updated as you traverse the tree selecting different nodes in a manner completely synchronized with the Tree view. At any instant, it will display the chart for the node selected at that instant in the Tree view. To view the tree and the node chart side-by-side, dock the Chart view next to the Tree view. To this end, click on the Chart tab name and drag the tab to the desired position on the screen.

If you choose the **Static** option, the Chart tab will always display the chart for the node selected in the Tree view at the time of adding the chart. It will not be updated as you traverse the tree selecting different nodes, and will always show the chart for one fixed node.

Single Variable: this option means that the chart will be for a single variable (same type as the Dataset Chart object).

Cross Tabs: The chart will be of the same type as the Cross Tabs chart of the Dataset object.

Child Nodes: Each child of the selected node will be represented by a bar.

If the DV is discrete, the height of each bar represents the size of the corresponding child node (number of records), and the proportion of each DV value is shown in a specific color.

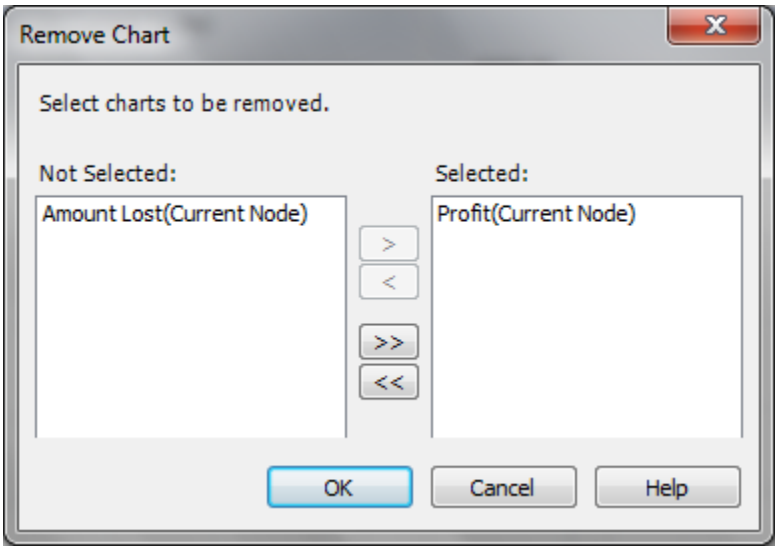
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If the DV is continuous, the height of each bar represents the average DV value in the corresponding child node.

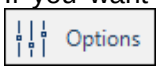
If you selected the Single Variable option, see the Dataset Chart topic in the Data Profiling section.

If you selected the Cross Tabs option, see the Cross Tabs topic in the Data Profiling section.

To remove a chart you added before, click the **Remove** button in the toolbar (or select **Tools | Remove Chart** from the menu). Double-click on the name of the chart to remove and click OK:



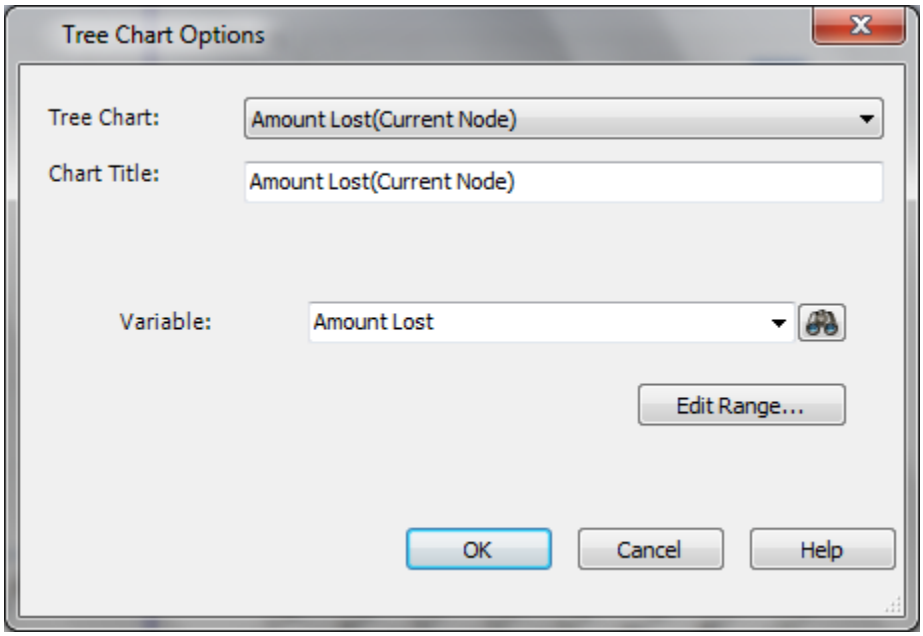
Node Chart Options

If you want to change the charted variable or ranges / bins in a Node Chart, click the button 

in the toolbar or select **Tools | Options** from the menu.

In the Tree Chart Options dialog, select the desired chart from the Tree Chart list.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



To change the variable for the selected chart, choose the desired variable from the drop-down list.

To change the binning of the selected variable, click the **Edit Range** button (or **Edit Grouping** button, if the variable is discrete).

Profile Chart

The profile chart shows the distribution of the dependent variable (DV) **in the terminal nodes** of the tree in the form of a bar chart. Each bar represents a terminal node in the tree, its width representing the size of the node (i.e., the number of records in the node).

Profile Chart is completely interactive and synchronized with the Decision Tree view, so you can click on the chart to navigate the tree, right-click on any node to choose a command from the same pop-up menu as in the Decision Tree view, so you can perform any split operation such as Find Split. If you tag a node in Profile Chart, it will be automatically tagged in the Decision Tree view, and vice versa.

Discrete DV

In the case of a discrete DV, you must specify the DV outcome of your interest using the drop-down menu at the top of the Profile Chart view.

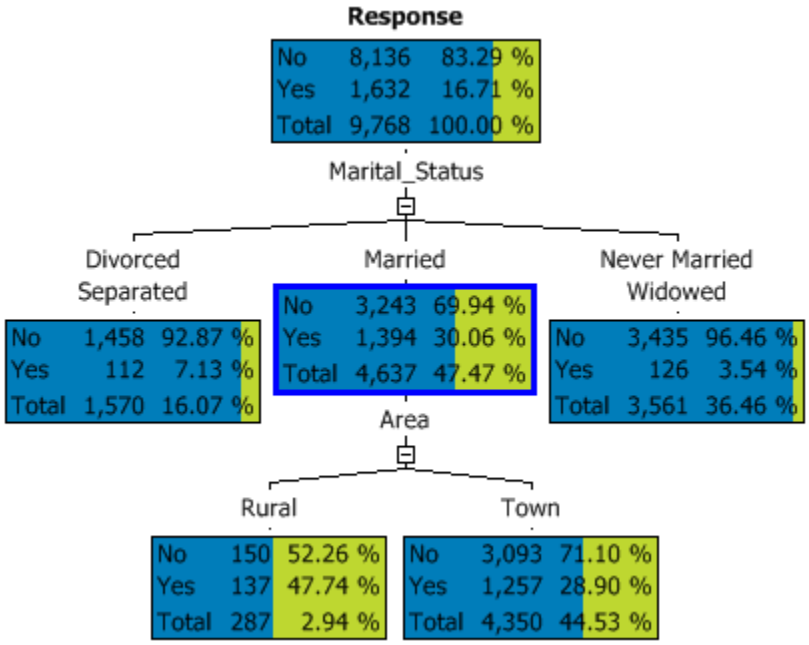
The height of each bar is the probability of the specified DV outcome for the corresponding node (equivalent to the percentage shown for this outcome in the Tree view). The width represents the size of the node relative to other terminal nodes.

The base line represents the probability of the specified DV outcome in the whole dataset (the root node of the tree). The Max line represents the maximum probability of the specified DV outcome over all terminal nodes.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

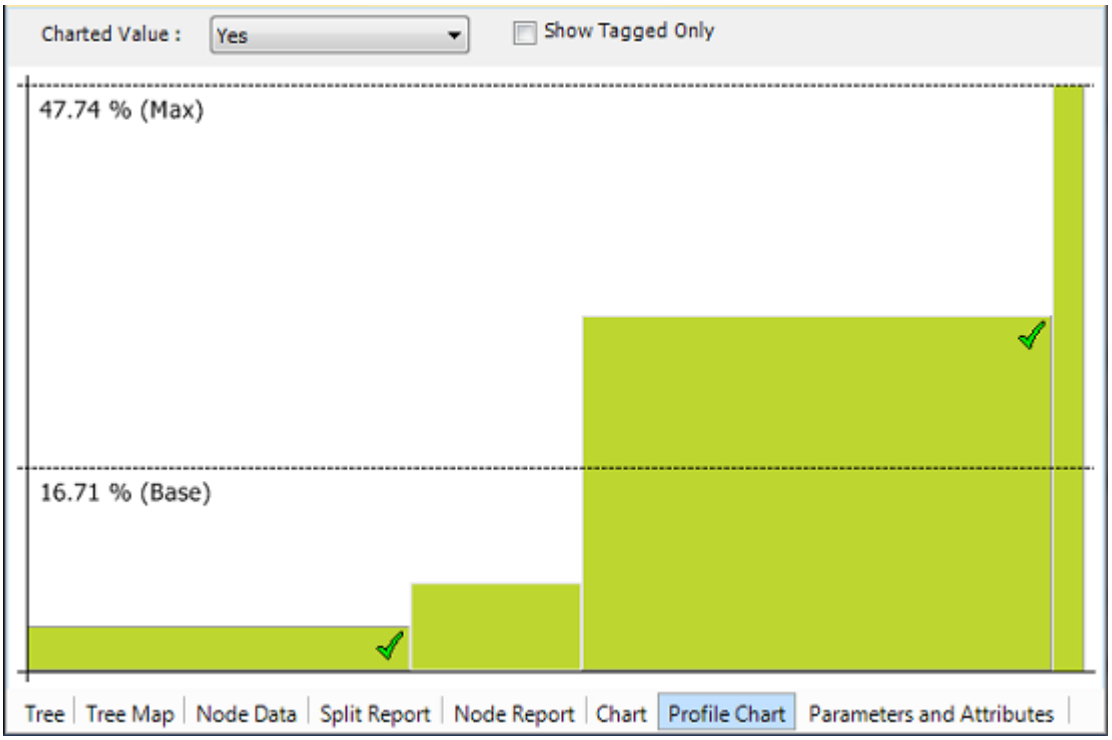
The nodes are sorted by the probability of the outcome. To the left of the screen are the nodes with the lowest probability of the specified outcome. To the right of the screen are the nodes with higher probability of the specified outcome.

For example, consider the following tree:



The Profile Chart for this tree is shown below (the charted DV value is 'Yes'). Note that two leaves are tagged in the tree view and Profile Chart shows these tags. You can choose to show only the tagged nodes by selecting the checkbox 'Show Tagged Only'. The base line (0.17) is the rounded probability of the 'Yes' outcome in the whole dataset (the root node). The Max line (0.48) represents the maximum probability of the 'Yes' outcome over all terminal nodes.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Continuous DV

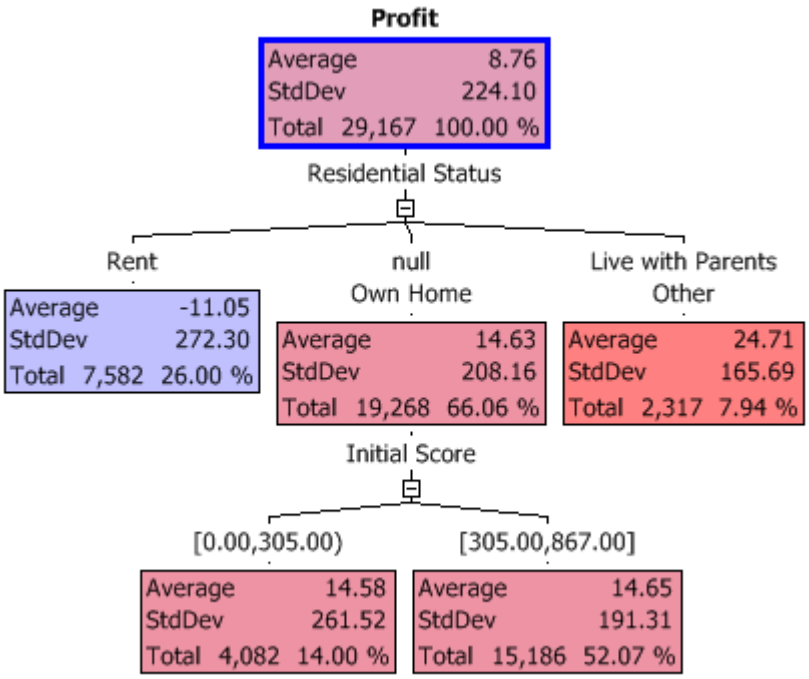
In the case of a continuous DV, the center line of each bar represents the average DV value in the corresponding node, the top and bottom of the box are average plus / minus one standard deviation in the node. As in the discrete case, the width of each bar represents the size of the corresponding node.

The center line of the chart represents the baseline, which is the average DV value in the whole dataset (the root node).

The Min and Max lines are the minimum of (Avg - StdDev) and the maximum of (Avg + StdDev), respectively, taken over all terminal nodes.

For example, consider the following tree with continuous DV called Profit.:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The Profile Chart for this tree is shown below. The average value of the DV is the only available charted value. Note that two leaves are tagged in the tree view and Profile Chart shows these tags. You can choose to show only the tagged nodes by selecting the checkbox 'Show Tagged Only'.

The base line (8.76) represents the average DV value in the root node of the tree. The Max line (287.83) and the Min line (-391.65) in this case correspond to (Avg +/- StdDev) for the node with the rule ([Residential Status] IN ('Live with Parents', 'Rent')) AND (0 <= [Initial Score] <305), which is represented by the leftmost bar.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Parameters and Attributes

The **Parameters and Attributes** tab displays a complete summary of the tree configuration:

- the settings you specified when inserting the tree in the Tree Wizard (also available under **Tools | Options** in the Tree view)
- the tree parameters from the Tree Attribute Editor
- expressions for calculated columns (if any are used in the tree)
- general tree statistics (the tree depth, the total number of nodes, the number of terminal nodes)

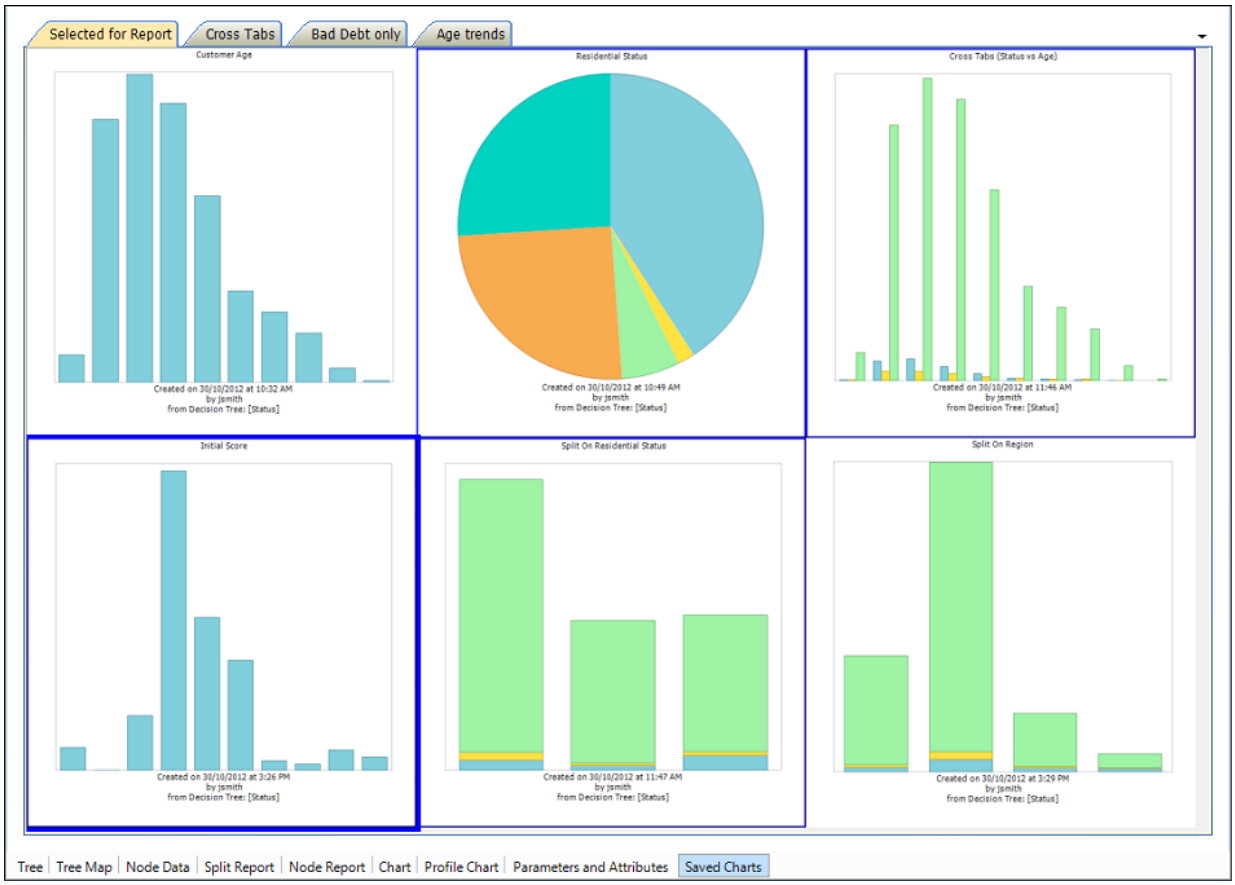
You can specify the precision for the numbers displayed in the Parameters and Attributes tab by selecting **Tools | Options** from the menu and choosing the desired notation (standard or scientific) and precision.

The Saved Charts tab is available for datasets, decision trees, and strategy trees. This tab represents the library of charts you saved in other views of the object it belongs to.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

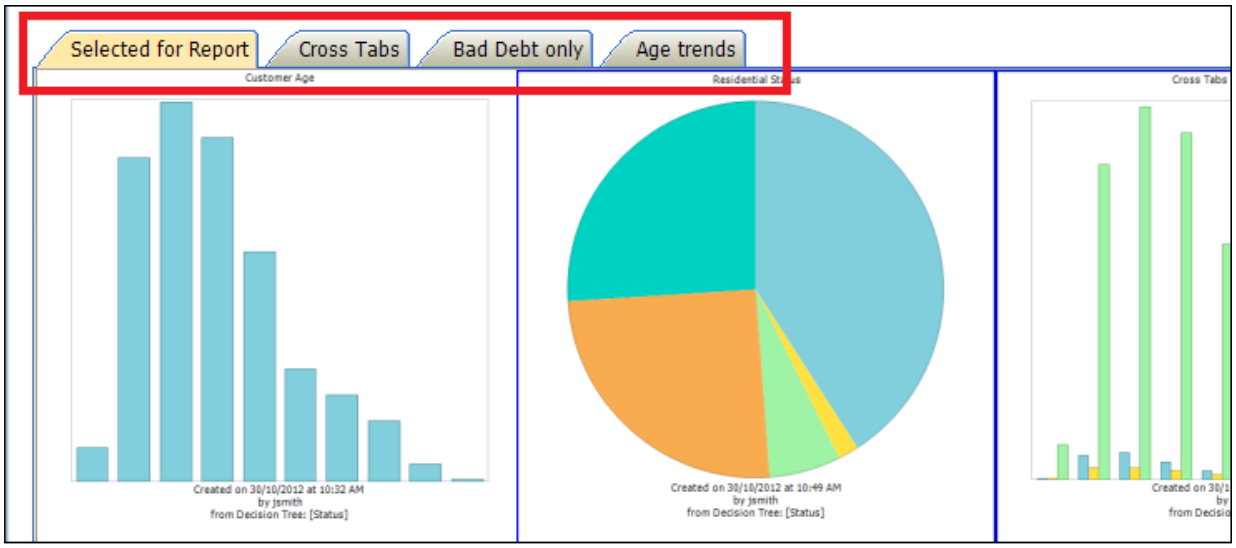
- The Saved Charts tab of a dataset contains charts saved in the Dataset Chart, Segment Viewer, Cross Tabs, and Correlations views of the dataset.
- The Saved Charts tab of a tree contains node charts saved in the Chart view of the tree.

In the Saved Charts tab you can view, organize, and print the charts, or copy them to Microsoft Office documents for reporting.



Charts are organized in groups. Each group is represented by a separate tab at the top of the chart grid. The tabs are highlighted by the red frame in the example screenshot below. Click on the tab names to switch between the chart groups.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

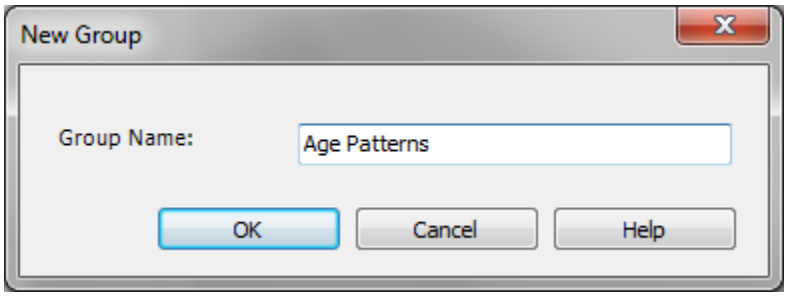


To view any chart in detail in a separate window, double-click on it and resize its window as necessary. In this window you can view the chart details, copy or print it, etc. For more information, see the topic Single Chart View.

Chart Groups

You can create new chart groups and distribute the charts in these groups.

To create a new chart group, select the **New Group** command from the **Groups** menu, or right-click on the group tab name and select **New** from the context menu. Then specify the desired name in the dialog:



A new blank chart group with the new name will be created.

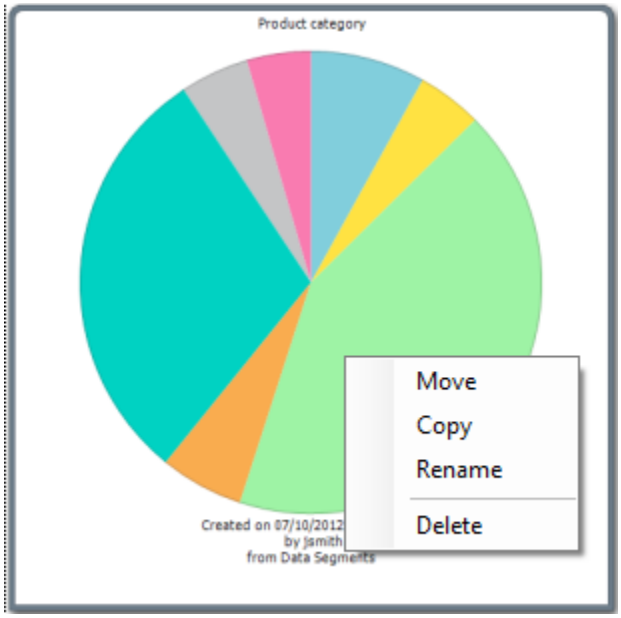
To rename an existing chart group, right-click on the group tab name and select the **Rename** command from the context menu. Enter the new name and click OK.

Operations with Charts

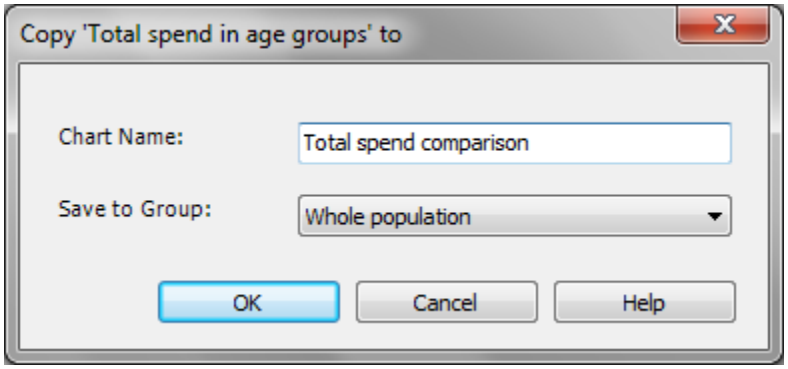
To change the position of any chart within its group, drag and drop it to the desired position.

To delete, copy, rename, or move a chart to a different group, right-click on the chart and select the appropriate command from the context menu.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- To delete a chart, select **Delete** from the context menu.
- To rename a chart, select **Rename** from the context menu, specify the new name for the chart, and click OK.
- Charts can be copied within the same chart group or to a different chart group. To copy a chart, right-click on it and select **Copy** from the context menu. Specify the name for the new chart and select the destination group. The current group is the default destination.



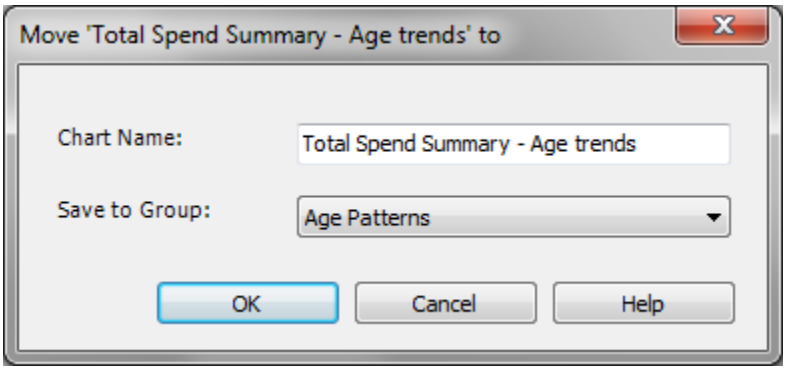
- There are two ways to move a chart to another group:

  a) drag and drop the chart to another group tab, or

  b) right-click on the chart and select **Move** from the context menu.

In the Move dialog, specify the new name for the chart and select the desired destination group from the Save to Group list

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also:

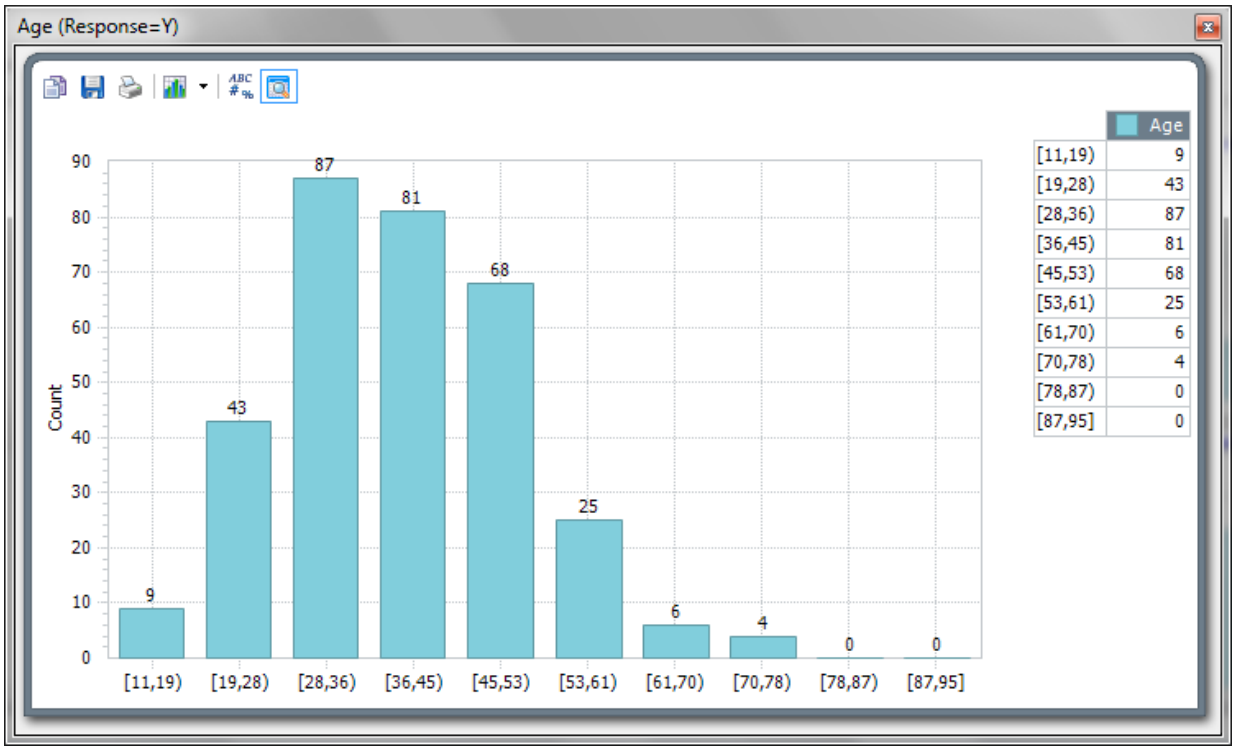
Single Chart View

Single Chart View

Several views in the application contain multiple charts: Segment Viewer in datasets, Chart in trees, and Saved Charts in both datasets and trees. In any of these views, you can open each individual chart in its own separate window, where you can view its details, copy or print it, etc.

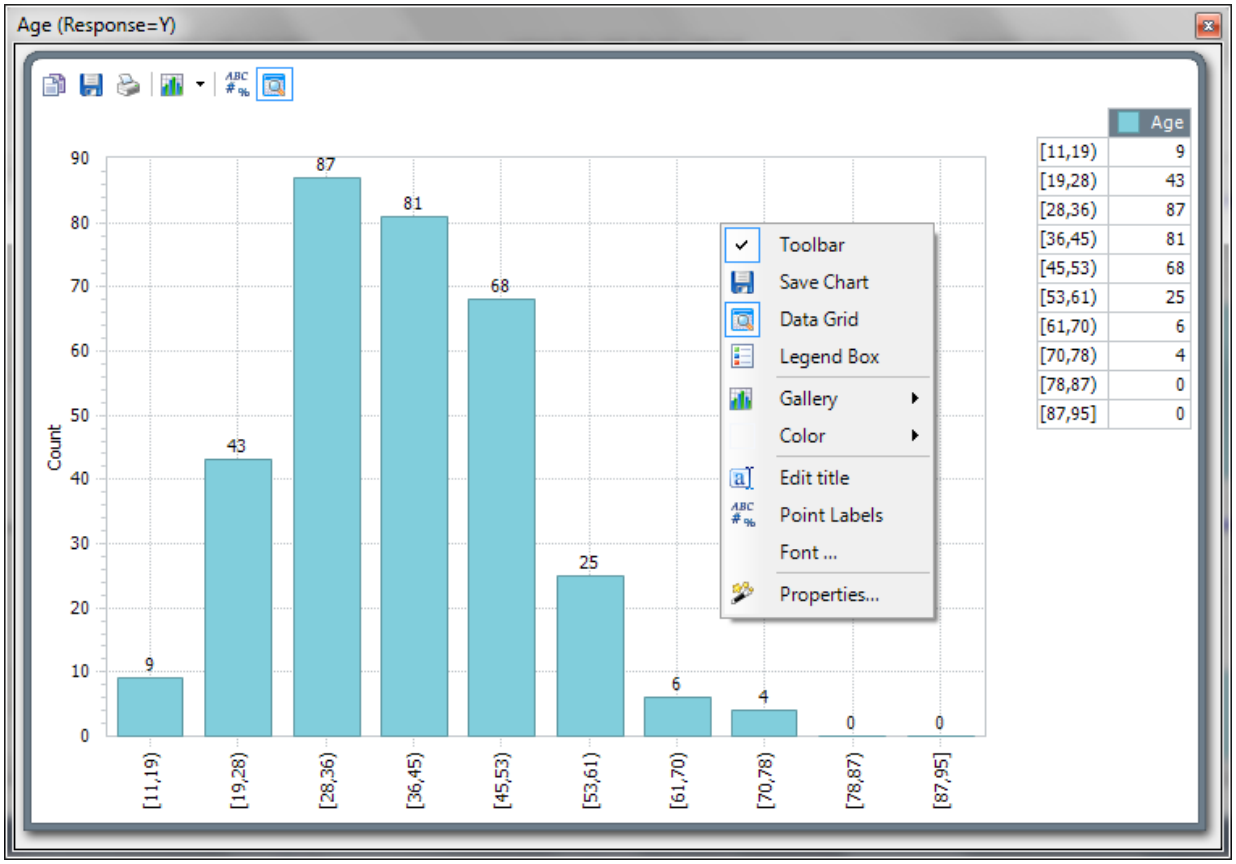
Double-click on the desired chart to view it in a separate window. The window provides a detailed view of the chart and shows the data grid representing the chart data. You can resize the window by clicking and dragging on its corners or borders.

Here is an example of a chart from Segment Viewer opened in its own window.



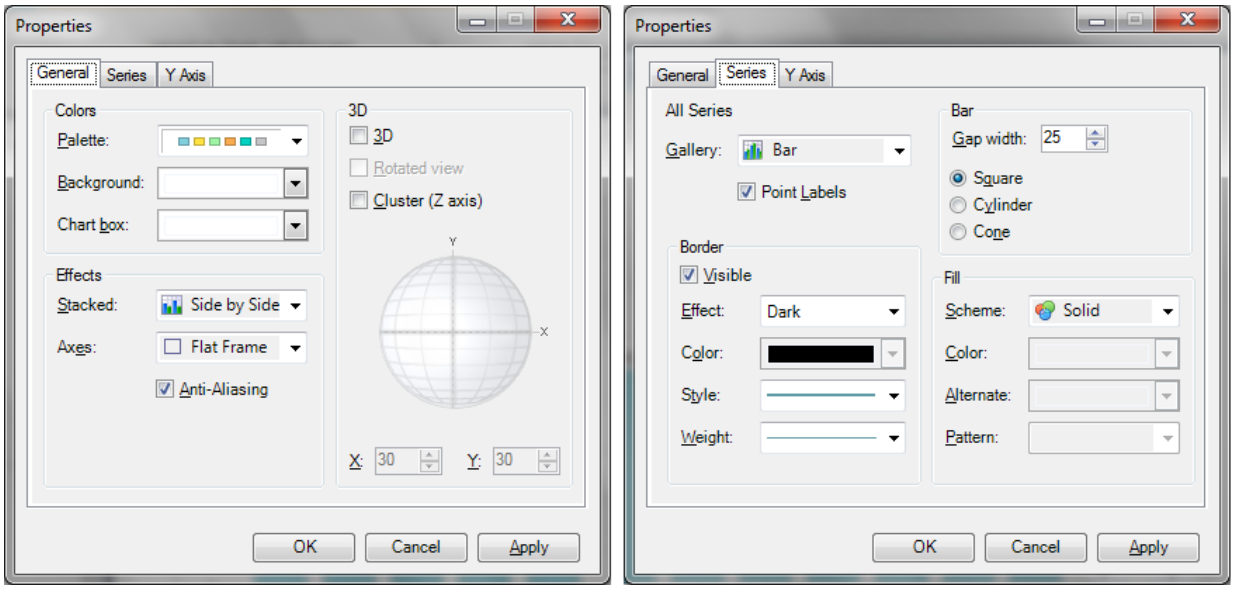
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To change the display properties of the chart, use the chart toolbar or right-click anywhere on the chart background. The following context menu appears when right-clicking on the chart background. For example, select Point Labels to show or hide the values on top of the bars; select Data Grid to show or hide the table with the chart data.

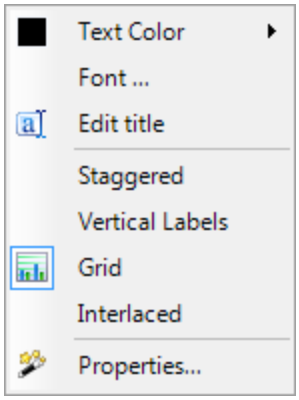


To change advanced properties of the chart, select Properties from the context menu. In the Properties dialog, you can choose the color palette, 3D or 2D view, effects, properties of the axis labels, show or hide grid lines, etc.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

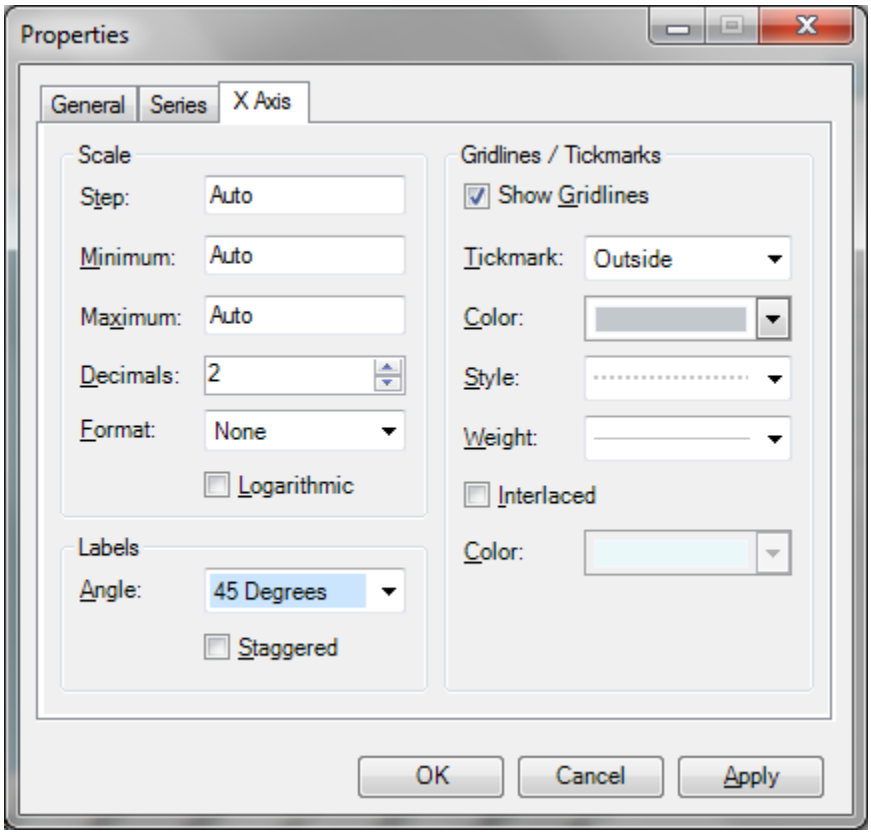


To change the orientation of the axis labels, right-click on any label of the desired axis (vertical or horizontal). The context menu will look like this:



Select *Staggered* or *Vertical Labels* if desired. If you want to place the labels at a 45-degree angle, select *Properties* and set *Angle* to 45 Degrees.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



As a result, the X-axis labels are displayed as follows:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

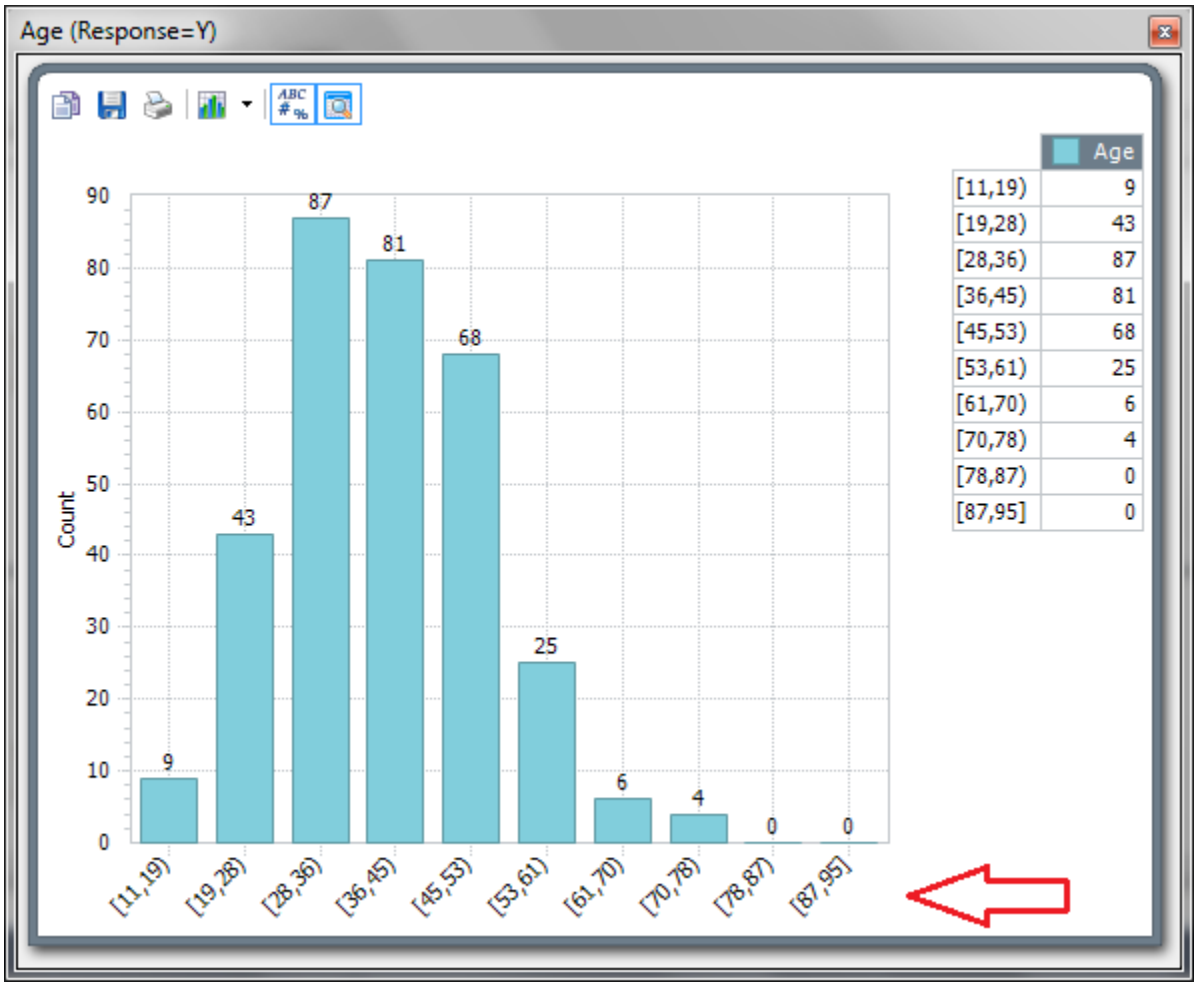


Chart Toolbar

Depending on the charts' parent view - Segment Viewer, Cross Tabs, Chart (in trees), etc., - its toolbar at the top left corner of the chart window can have one of the following two forms:

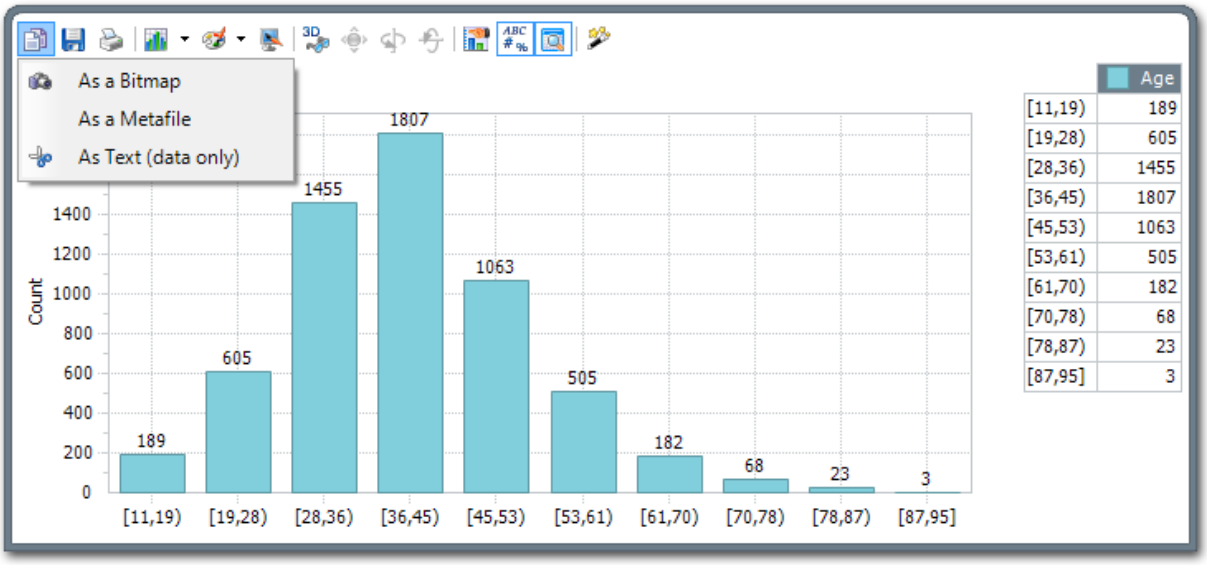




All icons in the Chart toolbar except the *Save*, *Copy*, and *Print* icons are shortcuts to the corresponding options and commands in the Properties dialog described above.

To copy a chart, click on the *Copy* icon in its toolbar, as shown in the screenshot below. Select *As a Bitmap* or *As a Metafile* from the context menu if you want to copy the chart as an image. Select *As Text* (*data only*) if you want to copy the underlying data, which is displayed in the data grid on the right-hand side. Then paste the copied image or text from the clipboard into the desired application.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If you are viewing a chart from the Segment Viewer or Node Chart, you can save it in the Chart Library by clicking on the Save button .

See also: Saved Charts

Copying Splits and Trees

The **Copy/Paste Splits** commands allow you to copy the splits from one tree and paste them to another. This feature can be useful in a variety of situations.

- See how the split in one node behaves when copied to another node of the same tree
- See how the split in one tree behaves when copied to the node of a different tree created in a different dataset.

Copy Splits

The **Copy Splits** command copies into memory all the splits of the tree, starting from the active node.

Highlight the node you want to copy

From the menu:

Edit | Copy Split

From the popup menu:

Right click on parent node | Copy Split

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Paste Splits

The **Paste Splits** command pastes splits that were copied to the clipboard with the **Copy Splits** command. The tree structure from the original Copy Split is maintained.

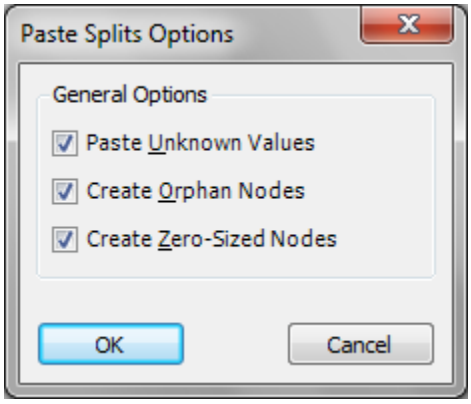
Highlight the node where you want to paste the split.

From the menu:

Edit | Paste Split

From the popup menu:

Right-click on the parent node
Select the Paste Split command



Choose the desired Paste Split options
Click OK to Paste the splits

Paste Split Options

Paste Unknown Values

This option applies to the independent variable (IV) value labels (predicate values of marginal rules) shown above each child node in splits. If some IV values in the source split are not present in the destination parent node (i.e., are 'unknown' to it), you can choose to show them anyway by selecting 'Paste unknown values'.

Create Orphan Nodes

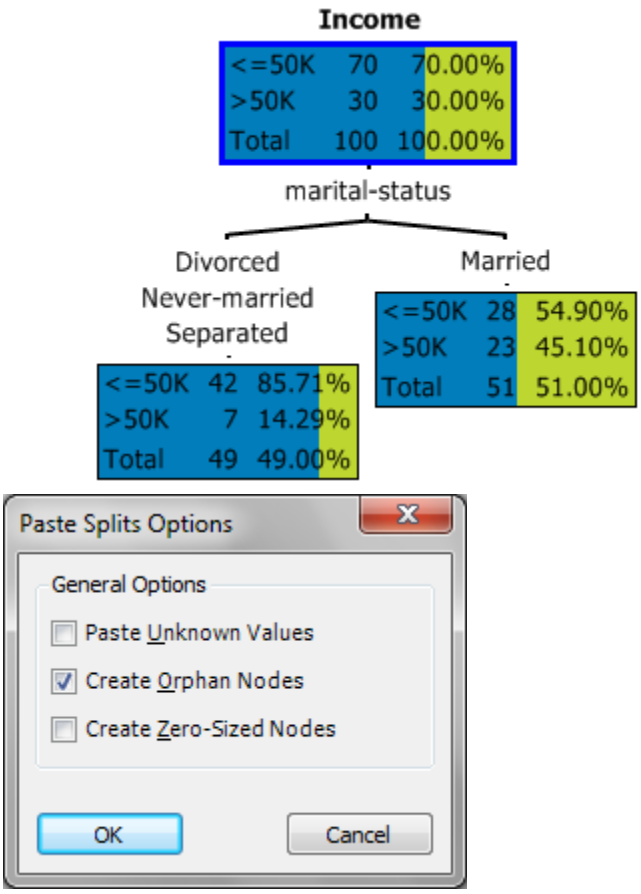
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Orphan nodes are created when a split is pasted to a tree whose dataset contains records that were not in the original dataset. This may occur when you copy splits from a tree created with a small dataset and paste to a tree created with a larger dataset.

For example, in **Tree A**, the split has three categories, color=***red, blue, green***. This split is pasted to **Tree B** but the dataset used to create **Tree B** contains four categories for color, color=***red, blue, green, yellow***. There is no rule in **Tree A** for color=***yellow***. To include the records for the rule where color=***yellow***, you must make sure the **Create Orphan Nodes** option is selected. If this option is not selected, the sum of records in the child nodes may appear to be less than the number of records in the parent node.

Example: The **Create Orphan Nodes** option
Pasting splits in the case where only the Create Orphan Nodes option is selected.

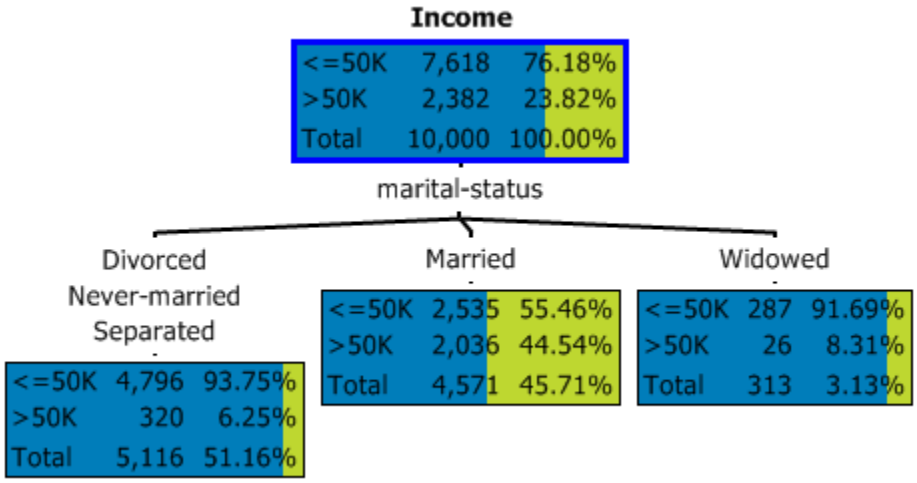
Tree A



We copied the split from the root node from **Tree A** and pasted the split to **Tree B**. The dataset for **Tree B** has the same data structure but contains more records. The split copied from **Tree A** does not have any records where ***marital-status=Widowed***. However, **Tree B** contains data where ***marital-status=Widowed***. If you did not select Create Orphan Nodes, records where ***marital-status=Widowed*** would never be displayed after the Paste Split operation. By selecting Create Orphan Nodes, these records will appear after the Paste Split operation in **Tree B**.

Tree B

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Create Zero-sized Nodes

Zero-sized nodes are nodes that contain no records. They are created when a split is pasted to a tree created with a dataset that does not contain records that were in the original dataset. This may occur when you copy splits from a tree created in a large dataset and pasted to a tree created with a smaller dataset.

For example, in **Tree A**, the split has four categories, color=***red, blue, green, yellow***. This split is pasted to **Tree B** but the dataset used to create **Tree B** only has three categories for color, color=***red, blue, green***. If **Create zero-sized nodes** option is selected, **Tree B** will create a node with the rule color=***yellow*** but node will contain zero records.

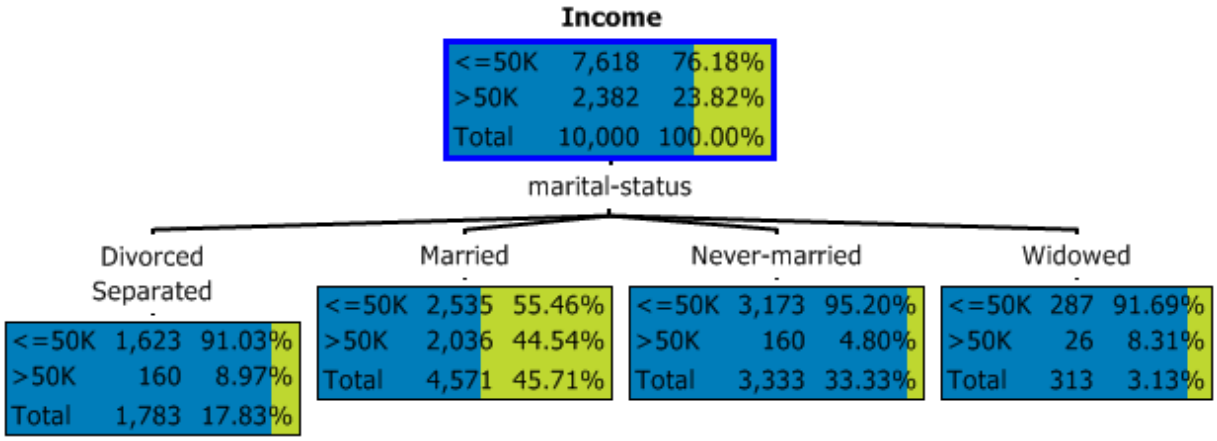
- If this option is not selected, the split will only contain nodes for color= ***red, blue, green***.
- If this option is selected but option Paste Unknown values is not selected, some zero-sized nodes may not be created because they correspond to unknown categories in the destination node.

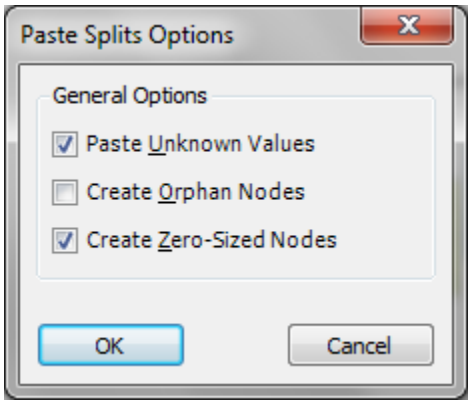
Example: The **Create Zero-sized Nodes** option

Pasting splits with the options Create Zero-sized Nodes and Paste Unknown Values.

Tree C

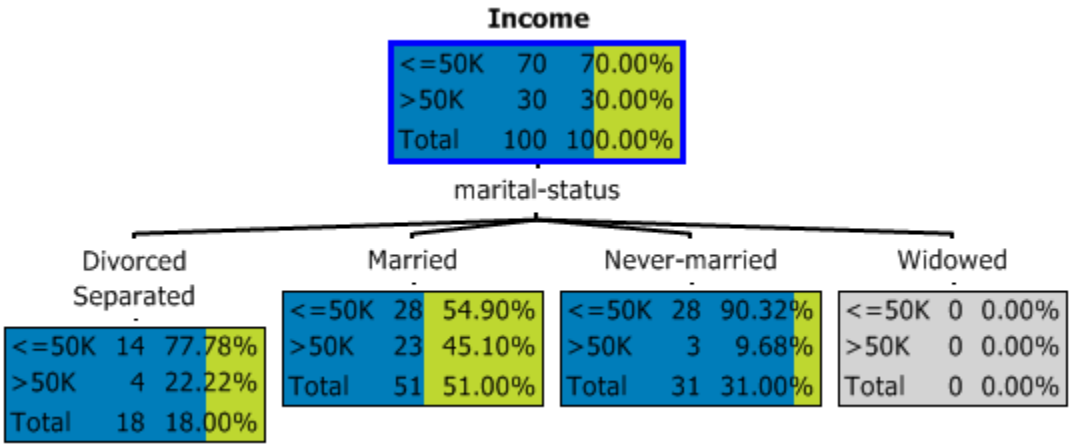
Altair Engineering, Inc. Copyright © 2022. All rights reserved.





We copied the split from the root node from **Tree C** and pasted the split to **Tree D**. The dataset for **Tree D** has the same data structure but contains fewer records. This dataset does not have any records where marital-status=Widowed. However, the split from **Tree C** contains a rule marital-status=Widowed. **Tree D** shows the rule and the pasted node with the rule marital-status=Widowed as a zero-sized node.

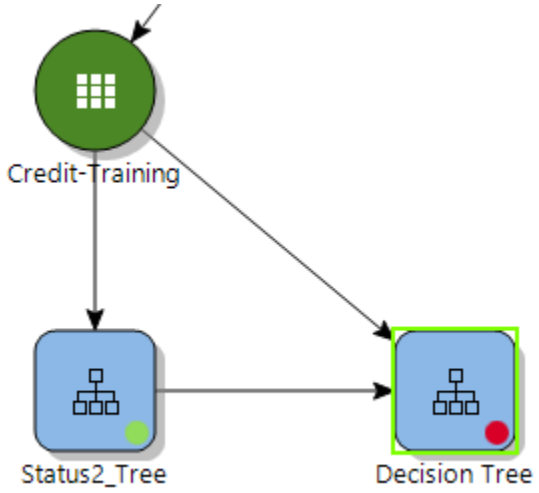
Tree D



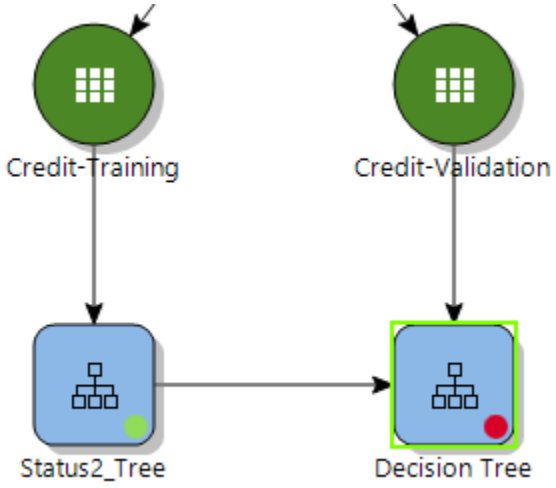
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Copying a Tree

To copy a decision tree or to use it as a template for creating a new one, add a new **Decision Tree** node to the workflow and connect the source tree to the new tree node. Then connect the dataset that the new tree should be based on. In the following example, the new tree is based on the same dataset as the original (template) tree.



You can also connect a different dataset (other than the parent of the template tree), as shown in the second example below. This is equivalent to copying all splits of the original tree to a new tree (root node) based on the new dataset. Note that the splits on the variables that do not exist in the other dataset will not appear in the new tree. It is assumed that the same dependent variable exists in the second dataset, and its data type is compatible with that in the template tree.



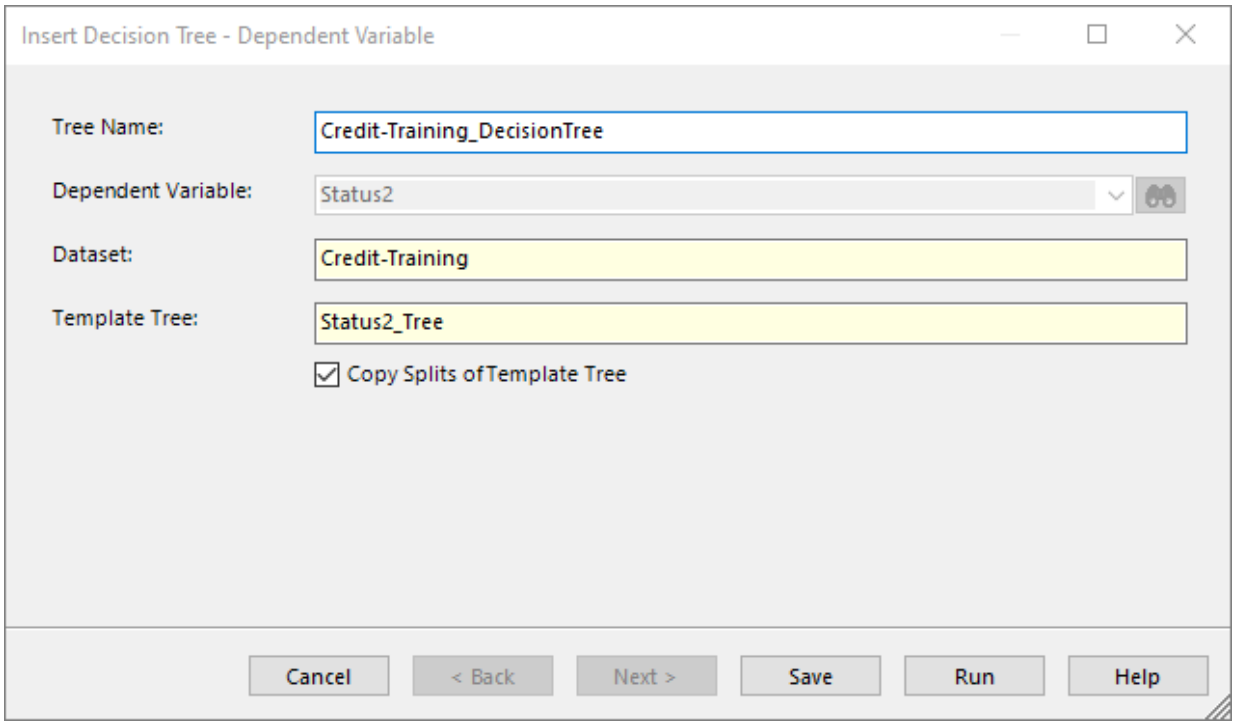
Double-click on the new tree node or right-click on it and select **Modify** to open the wizard.

The name of the original tree is displayed in the *Template Tree* field.

- Specify the name of the new decision tree in the *Tree Name* field.

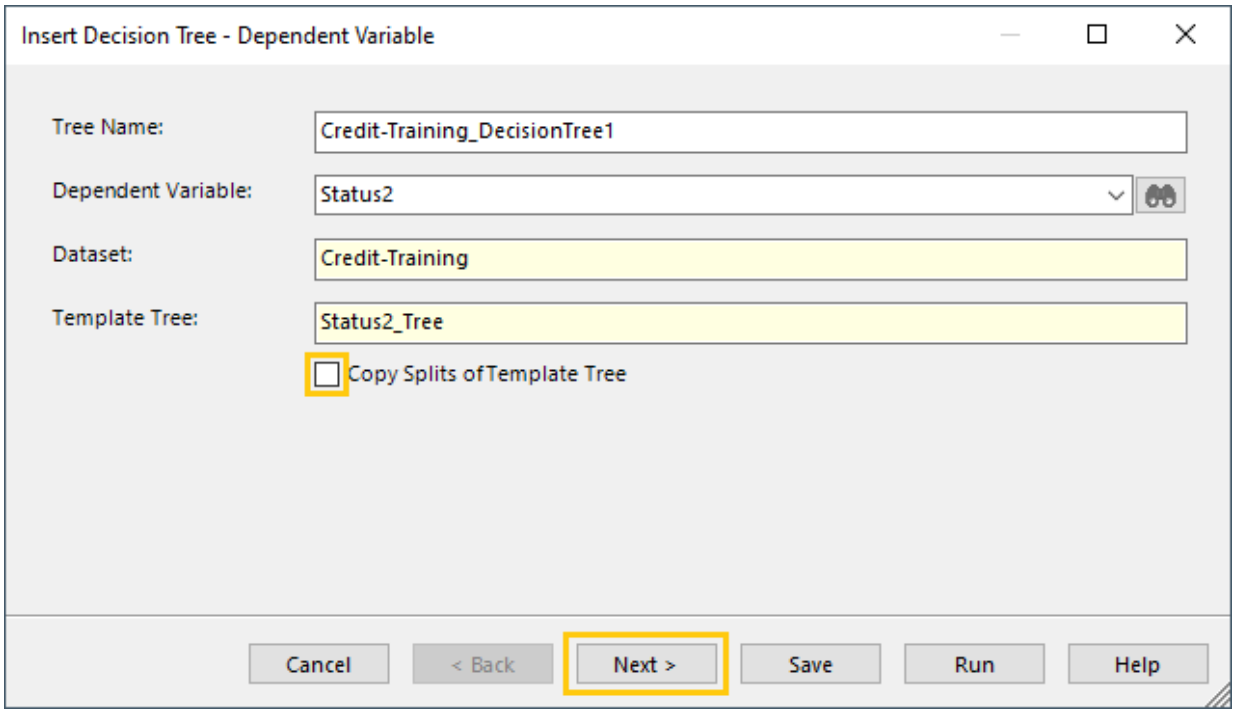
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- The option "*Copy Splits of Template Tree*" is selected by default. With this option selected, running the node will create a copy of the template tree along with its attributes. If the new tree and the template tree are based on different datasets, the operation is equivalent to copying all splits of the template tree to the root node of the new tree. Click **Save** to save the settings or **Run** to save the settings and run the process.



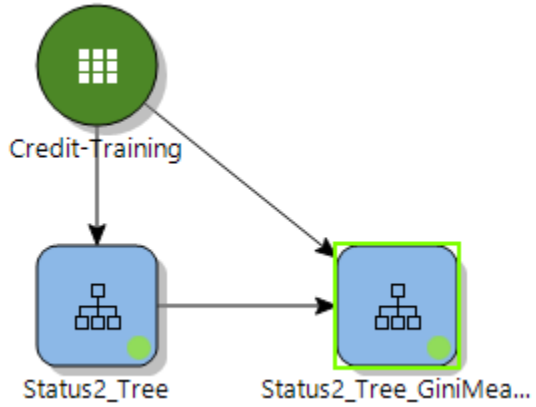
- To modify the properties of new tree before creating it from the template, deselect the box "*Copy Splits of Template Tree*" and click **Next.** This allows you to go through all steps of the Decision Tree wizard and change the tree properties and attributes if necessary. The default properties and attributes are inherited from the template tree, but you can change them as needed.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If you chose to modify the tree, see the topic Decision Tree: Training Parameters for the description of the next steps.

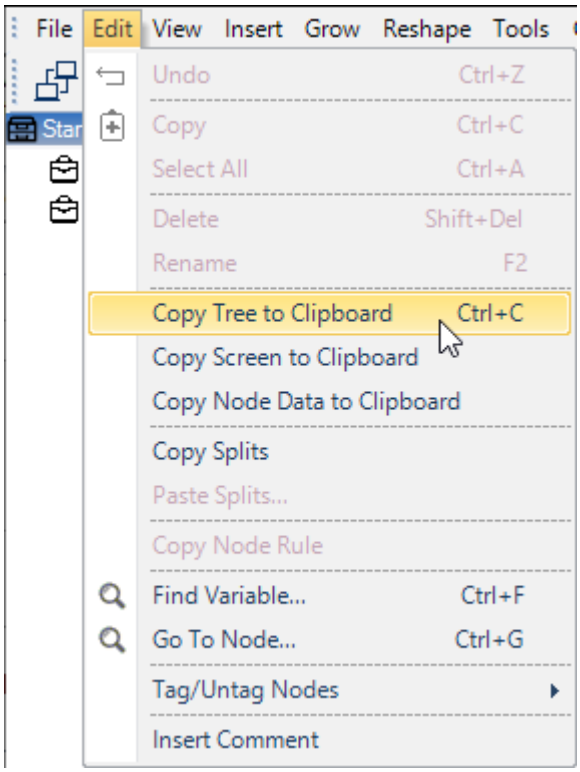
Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.



Copying a Tree as an Image

To copy the tree as an image in order to paste it to a presentation or another document, press CTRL+C or use the command **Edit | Copy Tree to Clipboard**. This command is also available on the right-click context manu of the tree background. Right-click anywhere on the tree background and select "Copy Tree to Clipboard". Then paste it to the desired document.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Copy Node Data to Clipboard

The command **Copy Node Data to Clipboard** copies the data, in a tabular form, from the active tree node and copies it to the clipboard. The copied data can then be pasted into another application, such as Microsoft Excel spreadsheet.

Select the node whose data you want to copy

From the main menu:

Edit | Copy Node Data to Clipboard

From the background context menu:

Right click anywhere on the tree background and select "Copy Node Data to Clipboard"

This command is only available when the decision tree model is the active view.

Copy Screen to Clipboard

The command **Copy Screen to Clipboard** copies a bitmap image of the visible tree to the clipboard. This image can then be pasted into another application, such as Microsoft Word or Excel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

From the main menu:

Edit | Copy Screen to Clipboard

From the background context menu:

Right click anywhere on the tree background and select "Copy Screen to Clipboard"

This command is only available when the decision tree model is the active view.

Modifying Tree Attributes

Using the Attribute Editor

The Attribute Editor allows you to modify variable attributes for the tree. The attribute values can be set and modified when inserting the tree in the tree wizard or modified after the tree has been inserted. Some of the tree parameters determined by the values set in the Attribute Editor are:

- whether a variable is included in the model
- the role of variables in the model (dependent, independent, or weight)
- the maximum number of nodes in splits produced by Find Split and AutoGrow
- the maximum number of bins or intervals in the Force Split Range Editor
- whether the values of a variable are treated as ordered or not
- whether grouped values of numeric variables are displayed in intervals or as discrete values
- the way missing values are treated

While a decision tree is open, the Attribute Editor can be invoked as follows:

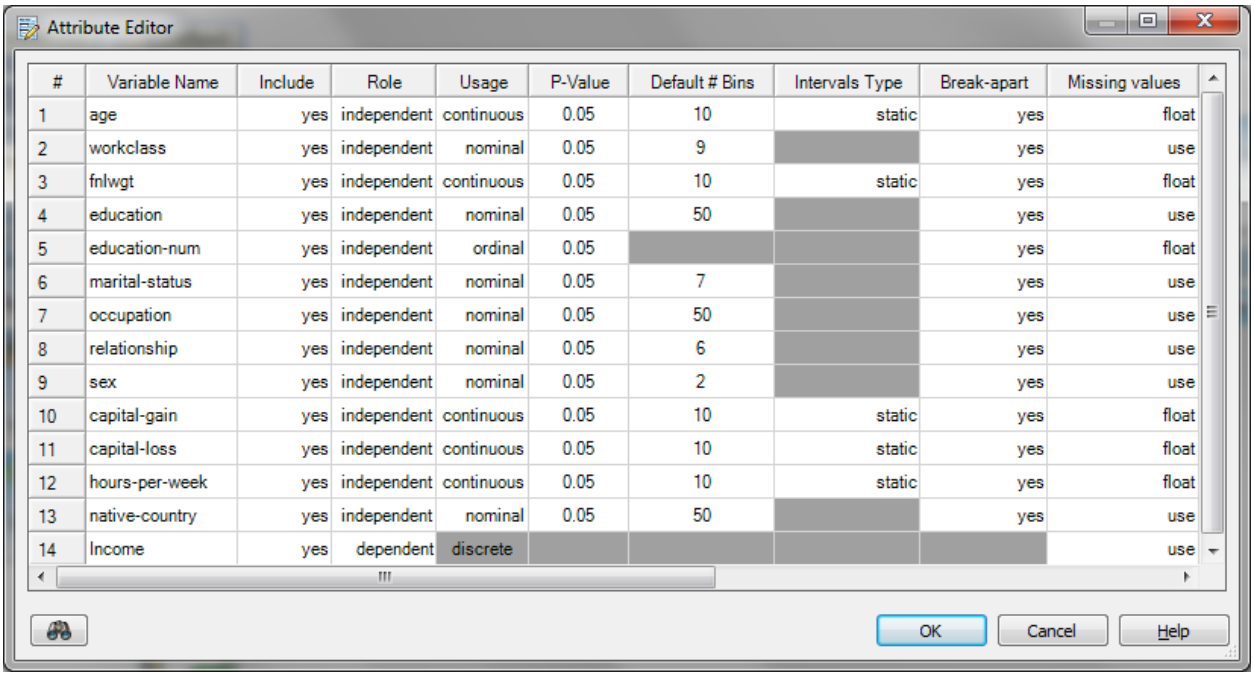
From the menu:

Tools | Attribute Editor...

From the background context menu:

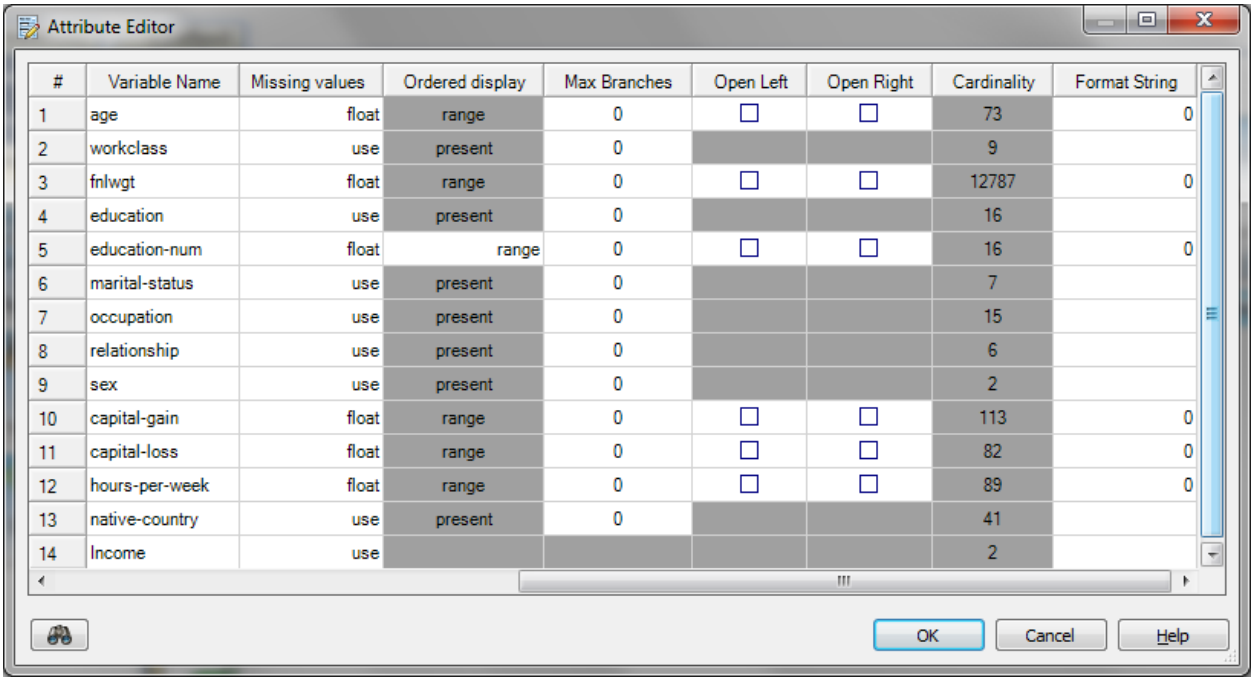
Right-click anywhere on the tree background and select Attribute Editor...

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



#	Variable Name	Include	Role	Usage	P-Value	Default # Bins	Intervals Type	Break-apart	Missing values
1	age	yes	independent	continuous	0.05	10	static	yes	float
2	workclass	yes	independent	nominal	0.05	9		yes	use
3	fnlwgt	yes	independent	continuous	0.05	10	static	yes	float
4	education	yes	independent	nominal	0.05	50		yes	use
5	education-num	yes	independent	ordinal	0.05			yes	float
6	marital-status	yes	independent	nominal	0.05	7		yes	use
7	occupation	yes	independent	nominal	0.05	50		yes	use
8	relationship	yes	independent	nominal	0.05	6		yes	use
9	sex	yes	independent	nominal	0.05	2		yes	use
10	capital-gain	yes	independent	continuous	0.05	10	static	yes	float
11	capital-loss	yes	independent	continuous	0.05	10	static	yes	float
12	hours-per-week	yes	independent	continuous	0.05	10	static	yes	float
13	native-country	yes	independent	nominal	0.05	50		yes	use
14	Income	yes	dependent	discrete					use

Scroll to the right to see more attributes:



#	Variable Name	Missing values	Ordered display	Max Branches	Open Left	Open Right	Cardinality	Format String
1	age	float	range	0	☐	☐	73	0
2	workclass	use	present	0			9	
3	fnlwgt	float	range	0	☐	☐	12787	0
4	education	use	present	0			16	
5	education-num	float	range	0	☐	☐	16	0
6	marital-status	use	present	0			7	
7	occupation	use	present	0			15	
8	relationship	use	present	0			6	
9	sex	use	present	0			2	
10	capital-gain	float	range	0	☐	☐	113	0
11	capital-loss	float	range	0	☐	☐	82	0
12	hours-per-week	float	range	0	☐	☐	89	0
13	native-country	use	present	0			41	
14	Income	use					2	

The global defaults for the tree attributes can be set in the Tree Attributes tab of the Preferences dialog accessible from the Tools menu. See the topic Preferences: Tree Attributes for details.

Tip: you can change attribute values for more than one variable at a time. To this end, click in the attribute column of your choice and drag your mouse down while holding down the left mouse button to highlight several cells in the column. (Alternatively, while holding down the SHIFT or CTRL key on your keyboard, click on several cells whose values you want to change in the selected column.)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

When finished selecting multiple cells, set the value for one of them (the last one you highlighted). This will set the same value for all highlighted cells in the selected column.

You can also click the Field Finder (binoculars) button to quickly find and select the desired variable and use the search capability.

Tree attributes can be copied from one tree to another. Open the tree whose attributes you want to copy and open its Attribute Editor. Click the Copy button and close the Attribute Editor. Then open the other tree, open its Attribute Editor and click the Paste button. All attributes for the variables that are common for both trees will be copied. Click OK to apply the changes and close the Attribute Editor.

NOTE: After you apply changes made in Attribute Editor and return to the tree view, you can still undo them using the Undo command, provided that you haven't made changes to a different tree or dataset.

Tree Attributes

Variable Name	Name of the variable in the decision tree.
Include	Determines if a variable is to be included in the analysis.
Role	Determines how a variable is used in an analysis. A number of options are available: **Dependent**: Indicates that this variable is the dependent variable, or the variable that you want to explore. There can only be one dependent variable (DV). **Independent**: Indicates that this variable should be compared to the dependent variable to determine if it has predictive value. **Weight:** Indicates that this variable should be used as a weight that controls how much emphasis each record should be given in the analysis. If this option is selected, you must also select the type of weighting in the Weight Type column. Example
Usage	Algorithms can treat values in a number of ways. In general, if the values of a field are ranked (like 1, 2, 3, or 1-low, 2- medium and 3-high) then they are 'ordinal'. If there is no ordering, i.e., if they are just categories or qualities, they are considered to be 'nominal' values (red, blue and green). **No binning**: Each category within the variable will form its own separate node in the tree. The search for similar groupings is turned off. **Ordinal:** Treats values as an ordered set (e.g., 1-low, 2-medium, 3-high or 1, 2, 3). Only adjoining values will be grouped together when an algorithm is looking for similarities. For example, 1-low can be paired with 2-medium, or 2-medium can be paired with 3-high but 1-low cannot be paired with 3-high. **Nominal**: Values combine freely. For example, the category 'red' in a String variable may combine with 'blue' or 'green'. All possible groupings of field categories are examined so that similar items are grouped together to become a node in the tree. **Continuous**:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

a) For *independent* variables: The variable is binned into intervals, which can be calculated in a number of different ways, based on your selections in the Intervals Type column.

b) For numeric *dependent* variables: The numeric DV is considered as a continuous quantity, and instead of its individual values the tree nodes show the average and standard deviation of its values.

Discrete: Available only for the dependent variable. This option allows you to treat a numeric dependent variable as discrete: Show all its values as discrete categories instead of treating it as continuous (showing Average and Standard Deviation in the nodes).

Example

p-value Defines the level of confidence required to merge values into groupings. This setting affects the bushiness of the decision tree - a low setting (0.001) will tend to produce few branches (splits) from each node, whereas a high setting (0.20) will produce a bushy tree with more branches. The default setting is 0.05.With a p-value setting of 0.05 you would typically only be wrong considering values as category components 5 times out of 100. Similarly, a setting of 0.001 will assure you that you have a 99.9% accuracy in the category assignments but that you will also have fewer resulting categories. A setting of 0.20 would result in more branches but with less certainty of the accuracy (80% confidence).

Default # of Bins - If ***Usage*** is `continuous`, the value in this column determines the number of intervals for the selected variable according to the following rule: To form splits in a decision tree, continuous values are initially divided into discrete intervals where each interval has approximately the same number of observations. These discrete intervals are then treated as ordered and combined accordingly. The number specified here determines how many **initial** intervals are formed for this continuous variable prior to the ordered grouping. Therefore, the resulting number of nodes may be less than or equal to the value in the Interval column. For example, if Interval=8, the resulting number nodes can be less than or equal to (<=) 8.

This value also determines the default nimber of intervals shown in the Range Editor of Force Split.

For more details, see the topic "Attribute Editor: Default Number of Bins".

- If ***Usage*** is `nominal`, the value in this column determines the maximum number of categories (bins) to display for the selected variable. The default value is equal to the cardinality or 50 (whichever is less). You may enter any integer number between 1 and the cardinality of the variable. If the maximum number of bins **N** is equal to the cardinality of the variable, then each of its values is represented as a separate category. If **N** is less than the cardinality, the following mapping is applied: the top **N-1** values ordered by frequency remain as they are, while the rest are mapped into one category denoted by triple number sign ("###").

The default maximum number of bins for nominal variables can be changed in the *Tree Attributes* tab of the **Preferences** dialog accessible from the **Tools** menu.

Intervals Type This option affects the Find Split and Automatic Grow operations and applies only to the variables with *continuous* Usage (see above). It defines the level at which the binning maps for continuous independent variables are calculated.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Static: The binning map for the variable is precomputed at the root node. The intervals remain static throughout the tree.

Dynamic: The binning map for the variable is calculated dynamically at every node.

Although the Static type is the default, the Dynamic type provides more intelligent mapping and is therefore recommended in most cases. Note that splitting a variable with dynamic interval type is up to two times slower than with the static type, but this may not be noticeable on fast computers.

Break-apart Allows you to optimize the growing of the tree by choosing either speed or accuracy. There are two options:

Yes: this setting gives the most accurate results. **Yes** is the default.

No: this setting is used when you have variables with many categories and you want to speed up the analysis. However, the results may not be optimal.

Missing Values Allows you to specify how missing values in your data are handled in the tree. The effect and applicability of the missing value options depend on the variable *role* (independent, dependent, or weight) and *usage* (see *Usage* above).

1. For the dependent variable (DV):

Use (applicable to discrete DV only): Missing values are treated like any other value of the DV. More precisely, a missing value will be considered a separate DV category designated as ***null*** and will be shown in every node together with other DV values. This option is not applicable to continuous DV.

***Ignore*:** All records containing missing values in the dependent variable will be excluded from the analysis. Note that in this case the root node will contain fewer records than the parent dataset. If the DV of a Decision Tree is continuous, the records with missing values in the DV are always excluded, so *Ignore* is the default and the only available option for continuous DV in Decision Trees. This does not apply to Strategy Trees.

Note that if you ignore missing values in the DV, it will affect all tree views except for the Data tab. which will still show records with missing values in the DV. (The Data tab always shows the actual raw data for any selected node, which is determined by the node rule only, not by the settings in Attribute Editor).

2. For the independent variables (IVs):

***Use*:**

a) For variables whose *Usage* is "nominal" or "no binning", the ***use*** option is the default. The missing value is considered one of the regular categories of the variable and will be used just like any other discrete category when calculating the split. Missing values are displayed as ***null*** in the tree view, the Range Editor of Force Split and Edit Split, and other views and dialogs.

During the iterative process of the Find Split algorithm, the missing value (*null*) category bin can be grouped with any other category bin starting from the initial iteration when the algorithm starts to merge the bins in the initial split, in accordance with the test of significance at the current level of *p-value* (see above).

b) For variables whose *Usage* is "ordinal" or "continuous", the missing value is mapped to a separate value (token) of the variable and will be used just like any other value when calculating the split. Missing values are displayed as ***null*** in the tree view, the Range Editor of Force Split and Edit Split, and other views and dialogs.

During the iterative process of the Find Split algorithm, the missing value (*null*) bin can be grouped with any other bin (interval) starting from the initial iteration when the algorithm starts to merge the bins in the initial split, in accordance with the test of significance at the current level of *p-value* (see above).

***Ignore*:**

Missing values in the variable will be ignored. Any record with a missing value in this variable will be dropped from the analysis. Any split on this variable will not include the condition "`IF <variable_name> is NULL`", so the child nodes in this split will not contain records with missing values in this variable. Note that in this

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

case the sum of the numbers of records in the child nodes of a split will not necessarily add up to the number of records in their parent node.

Float:

Before the Find Split algorithm starts the iterative process of calculating the split, the bin for missing values in the variable is removed from the initial set of bins for other values and is held to the side. With the regular values remaining, the Find Split algorithm then performs the normal iterative process of merging the bins in accordance with the test of significance at the current level of *p-value* (see above).

The last step of the algorithm will then take the missing value bin and search for the bin which it "best fits" according to the test of significance, and place it there if it is satisfies the criterion. Otherwise the missing value bin remains separate in the final split.

Missing values are displayed as ***null*** in the tree view, the Range Editor of Force Split and Edit Split, and other views and dialogs.

The *Float* option is the default for continuous and ordinal independent variables.

Separate bin:

Missing values will not be grouped with other data in the binning process of the Find Split algorithm. All records with missing values for the given variable (if any exist) will be put in a separate child node.

The behavior is the same as the ***float*** option except at the last step the missing value bin is not merged with any other bin at the end. Before the Find Split algorithm starts the iterative process of calculating the split, the bin for missing values in the variable is removed from the initial set of bins for other values and is held to the side. With the regular values remaining, the Find Split algorithm then performs the normal iterative process of merging the bins in accordance with the test of significance at the current level of p-value. At the last step of the algorithm, the missing value bin remains in its own separate node in the final split.

3. For variables used as weight: Not applicable.

Ordered Display

Allows you to define how ordered variables are displayed. There are three options:

Present: displays all values which are present in the node. If a particular value of the variable is not present in the node, it will not be listed.

All: displays all values that fit within the range in the node, whether they are present in that particular node or not. For example, if a node contains the values 1, 2, 4, and 5 (and the value 3 exists somewhere in the dataset), the value 3 will be listed for the node.

Range: displays values as a range. For example, if a split groups the values 1, 2, 3, and 4 together, the values will appear as [1, 4].

Max Branches

The **Max Branches** value determines the maximum number of branches (leaves) produced by Find Split on the given variable. For example, if you want Find Split to produce binary splits for a specific independent variable, specify "2" in the Max Branches column for this variable. If you want to grow a binary tree (i.e., a tree with binary splits only), enter "2" in the Max Branches column for all variables. See above for the tip on changing attributes for more than one variable at a time.

Note that this setting does not apply to the Force Split operation. Also, note the difference from the **Default # of Bins** attribute. The Find Split algorithm *starts* calculating the split with the number of branches equal to Default # of Bins. As opposed to Default # of Bins, the Max Branches limit never leads to the creation of the 'other' ("###") category for nominal variables.

Open Left

The splits created by the independent variable are not bound by the minimum data values found in the dataset.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Open Right	The splits created by the independent variable are not bound by the maximum data values found in the dataset.
Cardinality	The cardinality (number of unique values) of the variable.
Format String	Allows you to specify various formats (such as currency, percent, etc.) and precision for displaying values in the tree. You can either choose one of the standard formats from the Format String drop-down list or enter a custom format. For detailed description of format strings, see Specifying Format.

Weight in Decision Trees

The type of weight that can be defined in datasets and models is a generalized frequency weight (the weight values don't have to be integer). A weight can be set at the dataset level, but it can also be set and changed independently in trees and models.

See the Weight topic for general information about weights and on setting weight at the dataset level.

Weights in Datasets and Dependent Models

The weights in a dataset and its dependent models can be changed or reset independently. When you toggle weight on or off in the dataset Overview Report, this does not affect existing trees and other models depending on the dataset.

By default, any new model created on the basis of a weighted dataset inherits the weight from it. More precisely, when inserting a model for a weighted dataset, the weight field defined in the dataset is automatically listed in the model's Attribute Editor as "weight" (in the Usage column). You can override this in the Insert Tree (or Predictive / Cluster Model) Wizard and redefine the field as an analysis variable by changing the usage from "weight" to "independent" if desired. Once the model is inserted, it is independent of the parent dataset in the sense that any changes of weights in the model will not affect the parent dataset and vice versa.

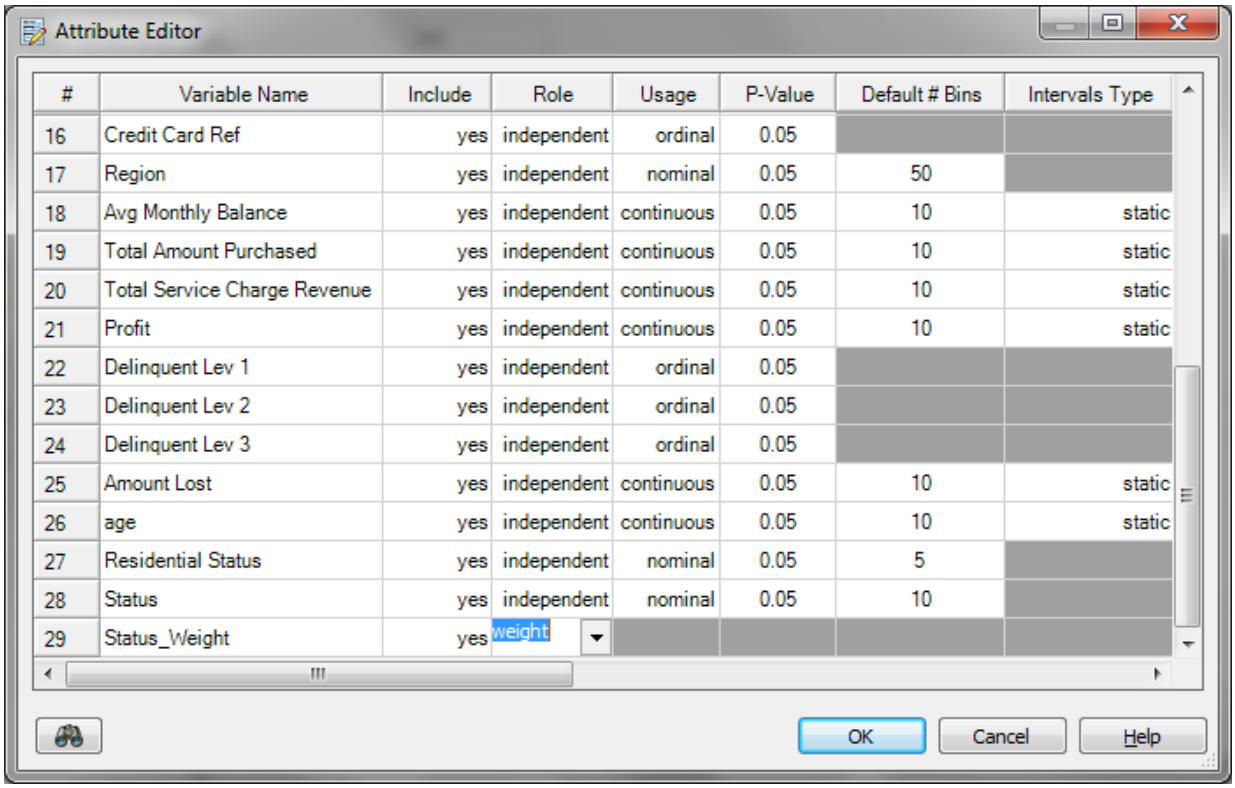
Weight in Decision Trees

To specify weight for an existing decision tree, open the tree and select **Tools | Attribute Editor** from the menu.

To specify weight for a new decision tree in the Decision Tree Wizard, open Attribute Editor by clicking the Advanced button in the last page of the wizard. (It is assumed that the field you want to set as weight already exists in the parent dataset. See the Weight and Weight Helper topics for information on available tools for creating a weight field.)

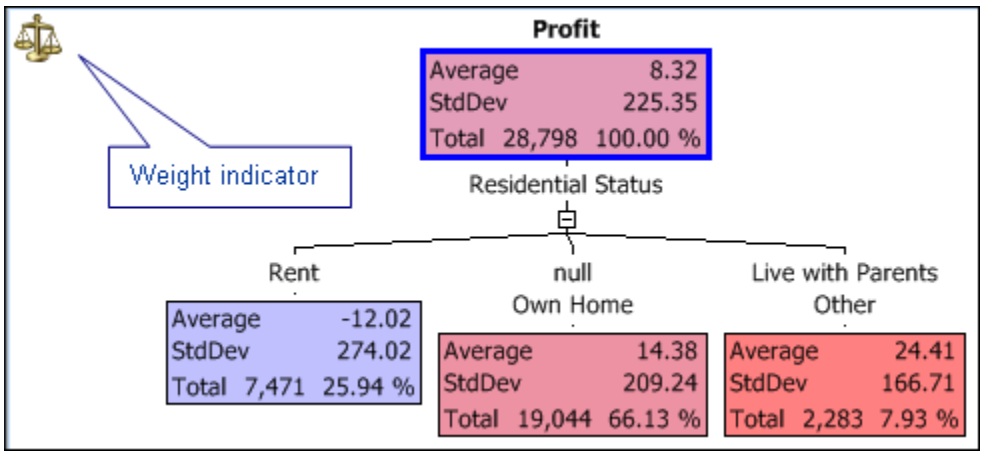
Click in the ***Usage*** column for the variable you want to set as weight and select "**weight**" from the drop-down list. Note that this usage option is available only for nonnegative numeric variables. The weight variable does not have to be integer.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Note that any existing splits in the tree will be deleted after you specify weight.

If the weight is defined, the scales icon displayed at the top-left corner of the Tree view serves as an indicator that the tree is weighted. In addition to Attribute Editor, the information about the weight is available in the Parameters and Attributes tab of the tree.



Since weight is treated as a (generalized) frequency weight, the effect of setting a weight in trees is as follows:

The number of records in all tree nodes is recalculated based on the defined weight (equals the sum of the weight field over all records in each node).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

NOTE: Non-integer node totals are possible, since the sum of the weight field over a node is not necessarily integer. By default, the node totals displayed in tree nodes are formatted as integer numbers, but you can change this setting by selecting Tools | Options and changing the format in the *Show Support (# of records)* field, for example, to 0.00 or #,##0.00 (for two digits after the decimal point).

The Dependent Variable distribution in all nodes is recalculated based on the defined weight. The weighted distribution in a given node is as follows:

If W is the weight variable, F_i is the i th value of the discrete dependent variable F, and SUM(W_i) is the sum of the weight field over all records in the node where $F=F_i$, then

- The weighted total number of records = SUM(W) (over the node)
- The weighted frequency for the value F_i = SUM(W_i) (over the node)
- The weighted percentage of the value F_i = 100 * SUM(W_i) / SUM(W) (sums taken over the node)

The effect in other tree views: All charts in the Tree Chart and Profile Chart are calculated based on the defined weight. Also, note that the columns of Node Report may display non-integer values.

NOTE: All of the above also applies to trees built in the Optimal Binning and WOE Helper Wizard.

Variable Usage Types

By setting the ***Usage,*** you can define the way in which the values of a field are treated by the tree algorithm and, therefore, the way they will be displayed in the splits (how they will be grouped, whether they will be represented in ranges, etc). For example, if the values of a field are ranked (like 1, 2, 3, or low, medium and high) then they are 'ordinal'. If there is no ordering, i.e., if they are just categories or qualities, they are considered to be 'nominal' values (red, blue and green).

Usage set to 'no binning'

When set to ***no binning***, each value of the independent variable forms a separate branch, i.e., individual values are presented as unique nodes in the tree.

Example

Usage set to 'ordinal'

Values of this Usage are treated as an ordered set (e.g., 1, 2, 3, or a,b,c...). Only values immediately succeeding one another will be grouped together when an algorithm is looking for similarities. For example, 'a' can be paired with 'b', but not with 'c'.

This Usage is the default for independent variables of types Number, Integer, Date, Time, or Timestamp if their cardinality is less than 50.

You can assign the Usage 'ordinal' to String variables (which are 'nominal' by default). In this case, when creating the split, the algorithm will order the values alphabetically and then try to group contiguous values together to create a branch. If your data is not alphabetically ordered, you may want to create a calculated column that transforms it. For example, use the Binning helper in Dataset Editor to transform the category 'low' to '***1-low'***, 'medium' to '***2-medium'*** and 'high' to '***3-high'.***

Example

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Usage set to 'nominal'

Values of this Usage are treated as an unordered set. All possible groupings of field categories are examined so that similar items are grouped together. This Usage is the default for String and Boolean variables.

Example

Usage set to 'continuous'

This Usage is available for independent variables of types Number, Integer, Date, Time, or Timestamp and it is the default if the cardinality is higher than 50. A variable of this Usage is split into intervals that are calculated in a number of different ways based on your selections in the Intervals Type column and the ***Intervals/Bins*** column. See also the topic Using the Attribute Editor for more information about the Intervals/Bins column.

Usage set to 'discrete'

This Usage is only available for dependent variables. It keeps the variable ungrouped. Each value is represented as an individual category. Dependent variables of types String and Boolean are always discrete. Dependent variables of types Number, Integer, Date, Time, or Timestamp are considered discrete by default if their cardinality is less than or equal to 10. Otherwise, if their cardinality is greater than 10, their default Usage is continuous.

Default Number of Bins

The ***Default # Bins*** in Attribute Editor affects both Find Split and Force Split commands.

Effect on Find Split

- If the Usage of the variable is **`continuous`**, the Default # Bins determines the number of intervals for the selected variable according to the following rule: To form splits in a decision tree, continuous values are initially divided into discrete intervals where each interval has approximately the same number of observations. These discrete intervals are then treated as ordered and combined accordingly. The number specified here determines how many **initial** intervals are formed for this continuous variable prior to the ordered grouping. Therefore, the resulting number of nodes may be less than or equal to the value in the Interval column. For example, if Interval=8, the resulting number nodes can be less than or equal to (<=) 8.

- If the Usage of the variable is `nominal`, the value in this column determines the maximum number of categories (bins) to display for the selected variable. The default value is equal to the cardinality or 50 (whichever is less). You may enter any integer number between 1 and the cardinality of the variable. If the maximum number of bins **N** is equal to the cardinality of the variable, then each of its values is represented as a separate category. If **N** is less than the cardinality, the following mapping is applied: the top **N-1** values ordered by frequency remain as they are, while the rest are mapped into one category denoted by triple number sign ("###").

The default maximum number of bins for nominal variables can be changed in the *Tree Attributes* tab of the **Preferences** dialog accessible from the **Tools** menu.

Effect on Force Split

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

For **Force Split**, the ***Default # Bins*** attribute determines the default number of bins for the selected variable in the **Range Editor** dialog of Force Split. The actual number of bins or intervals in the Range Editor for a given variable is the lesser of the *Default # Bins* value and the *cardinality* of the variable.

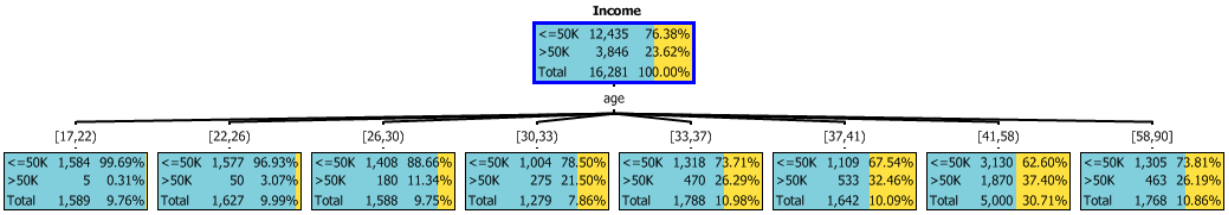
Effect on Scoring

If the Default # of Bins threshold for a nominal variable leads to the creation of the ### map as described above, the tree scoring is affected as follows:

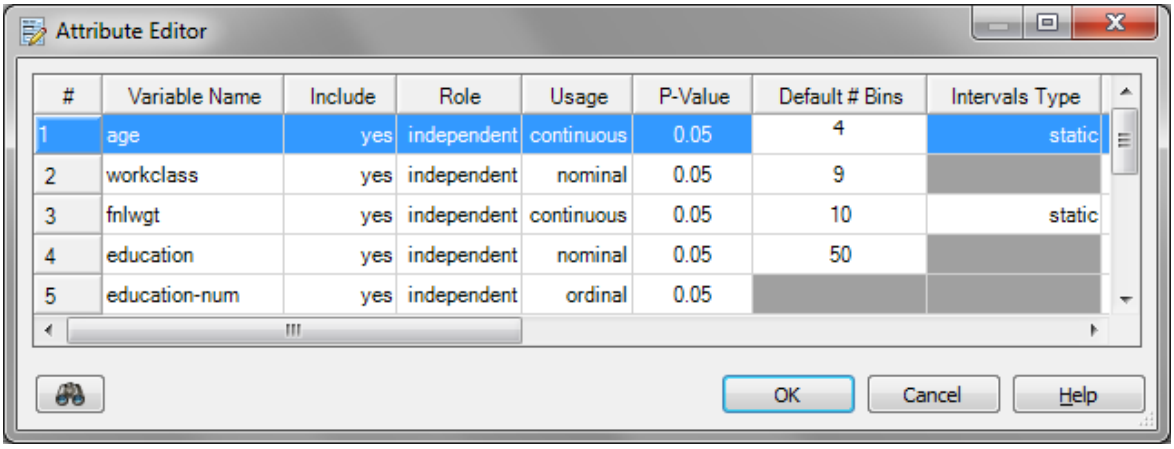
- Records with original values grouped in the ### category will be assigned the probability defined by the ### node.
- Records with 'unknown' values that do not occur in this variable in the training dataset will also satisfy the ### node rule and will therefore be assigned the same probability. If you do not want these 'unknown' values to be treated the same way, you may want to increase the Default # of Bins either in Attrubute Editor (for the current tree only) or globally in the *Tree Attributes* tab of **Preferences**. See Example 3 below.

Examples

Example 1: The Find Split operation produces a split on the variable *age*. The Default # Bins for *age* in Attribute Editor is set to 10. As shown below, Find Split produces eight child nodes. The algorithm started with 10 bins and merged several bins in the process to form 8 bins at the end.

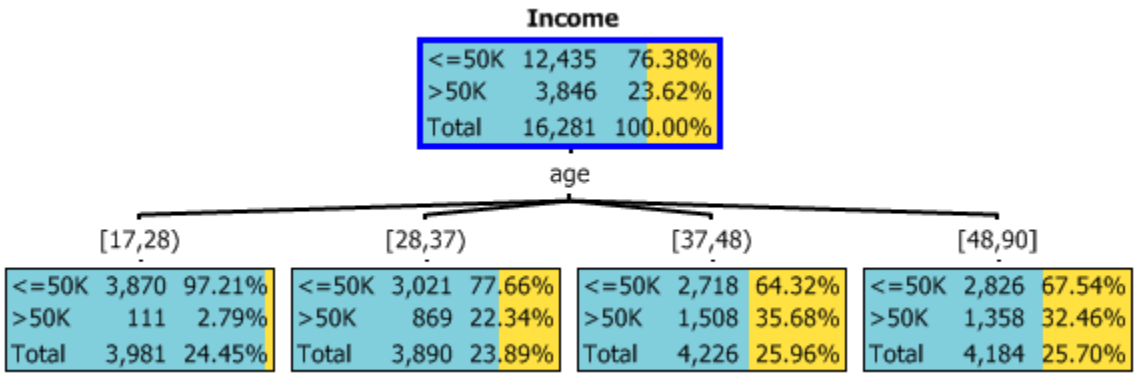


In Attribute Editor, change the Default # Bins value for *age* to 4:

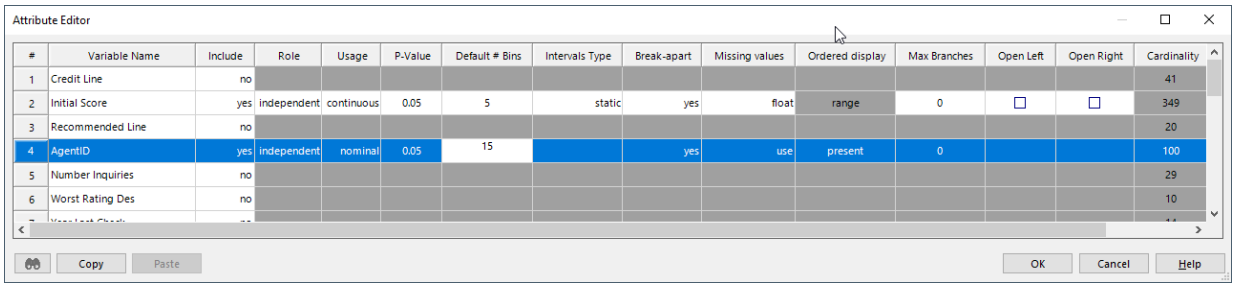


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

After applying the change, the split on *age* will be removed from the tree. Performing Find Split after this change will produce four child nodes:

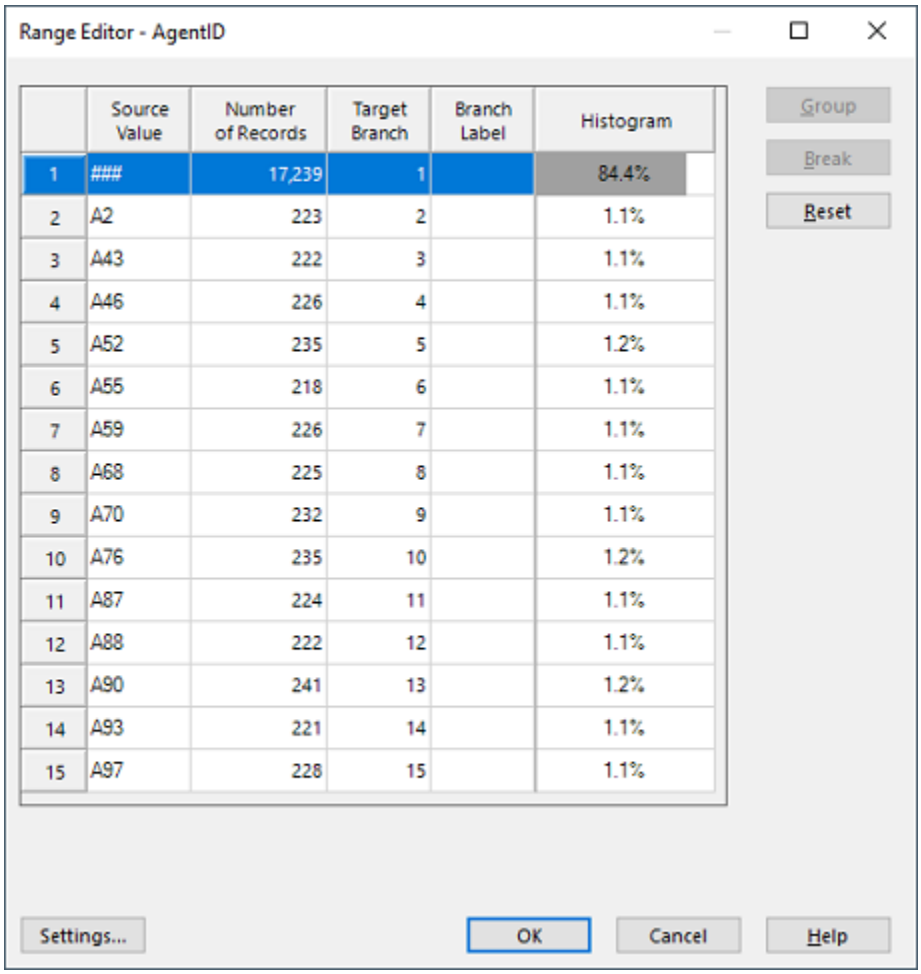


Example 2: For the nominal variable *Agent ID,* the Default # Bins in Attribute Editor is 50 - the application default for nominal variables defined in **Tools | Preferences** (the Tree Attributes tab). We change it to 15 as shown below:



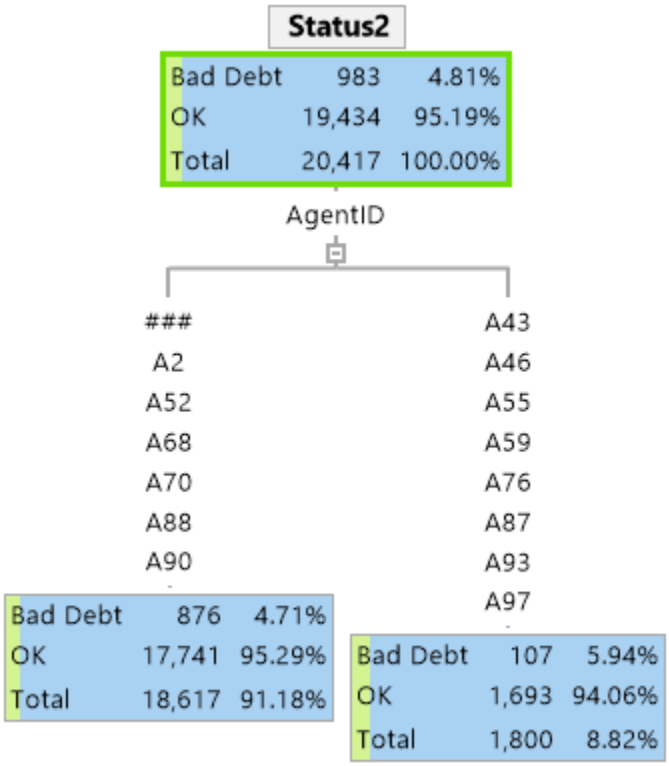
Since the cardinality of Agent ID is 100, and the Default # Bins is 15, the first 14 values with the highest frequency will remain as they are, while the remaining 86 will be mapped to the category ### ('other'). When performing Force Split on Agent ID, the Range Editor displays the ### category at the top:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



When doing Find Split, the result is as follows:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Example 3:

In the conditions of Example 2, suppose the tree in the above screenshot is scored on a dataset where some records have *Agent ID* value 'A500', which does not occur in the training dataset. Since the ### category means '*all other values*', the records with Agent ID = 'A500' will end up in the left leaf of the tree, just as those with 'known' values of Agent ID grouped in ###, and will be assinged the Bad Debt probability of 0.0471 (4.71%) and the OK probability of 0.9529 (95.29%).

If you do not want these 'unknown' values to be treated the same way as the 'known' ones, you may want to eliminate the ### map by increasing the Default # of Bins either in Attrubute Editor (for the current tree only) or globally in the *Tree Attributes* tab of **Preferences**. Then, during scoring, in our example the records with 'unknown' values of Agent ID will end up in the root node and will be assigned the probabilities 0.0481 and 0.9519.

In any tree, you can select one or several nodes and create a separate dataset that would contain the records from the selected node(s) only. The resulting dataset is called a node dataset.

Creating a node dataset can be useful in many cases. For example, if you used a decision tree to isolate a certain segment and you are not interested in analyzing any other data, then you may want to insert a node dataset based on this segment.

A node dataset has the same properties as any other dataset in the project. It is only referred to as a node dataset because of where the data originated. In the Project Outline, a node dataset is shown as a child object of the tree it is based on.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

NOTE: A node dataset inserted from a node of a weighted tree inherits the weight from the tree. If necessary, you can turn the weighting off in the node dataset after it is created.

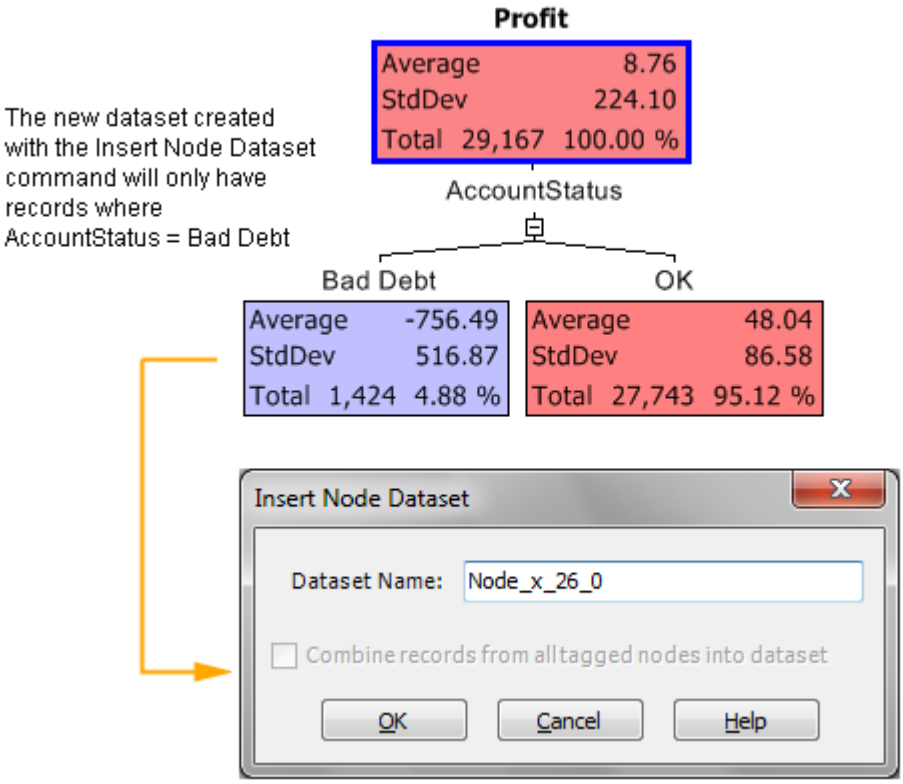
To insert a node dataset:

- Click on the node you want to use to create a node dataset
- To merge data from more than one node, tag each of the desired nodes (right-click and select Tag/Untag Node)

From the menu:

Insert | Node Dataset

The Insert Node Dataset command can be used to stratify the data, as an alternative to the Partition Wizard. For example, if your dataset contains credit account data and you want to analyze only the bad debt accounts, insert a decision tree with any dependent variable (other than *AccountStatus*) and do a **Force Split** on the variable *AccountStatus*.



The default name for the node dataset is its Node ID (note that you can view Node IDs for each node if you include Node ID in the Tree Node Settings tab of the Tree Options dialog via **Tools | Options**). You can specify a different name, if desired.

The new dataset will appear as a child object of the decision tree in the Project Outline. In our example, in the Overview Report for the new dataset, the total number of records equals the number

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

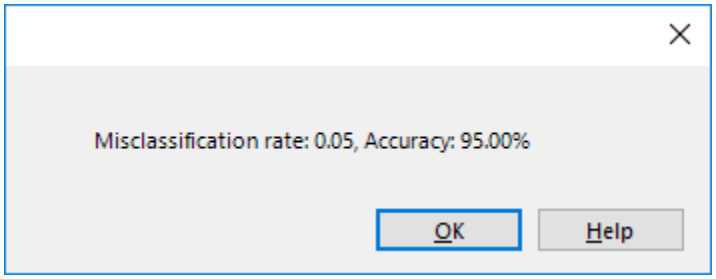
of records in the source node (1424), and it contains only the records where AccountStatus = 'Bad Debt'.

Accuracy Estimate: Resubstitution

The **Tools | Resubstitution** command can be used to quickly estimate the level of accuracy for the current tree. Note that the Resubstitution command performs a static analysis of the model. Generally, the error rate will be underestimated, since no new data is used to test the tree. For comprehensive validation that compares the model with data, use Model Analyzer or the commands **Tools | Tree Validation** or **Tools | Validate** (if available in your configuration).

Resubstitution error results are presented in one of two ways - based on the type of the dependent variable (DV) used.

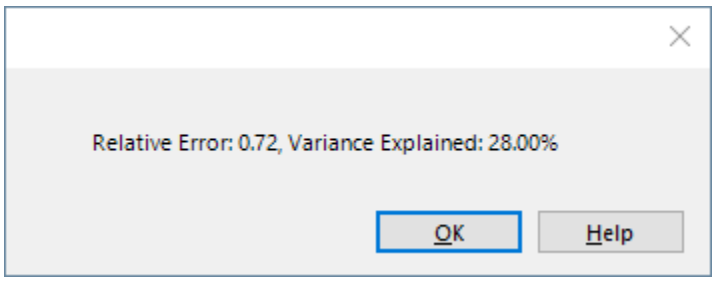
Discrete DV



The value of Accuracy tells you how often you will be correct in assigning records from a sample into the DV's modal (predominant) category. For example, if the reported accuracy is 60%, then you will correctly assign, about 60% of the time, an unclassified record in the modal category of the node that corresponds to the unclassified record's attributes.

You can verify the accuracy of the calculation by looking at the percentage of observations in each of the ***terminal*** nodes of the decision tree that fall outside the modal category. The weighted sum of these percentages over all terminal nodes of the tree is the overall error rate (the error rate for a given node is weighted by the number of observations in the node relative to the total size of the tree).

Continuous DV



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The value of Variance Explained tells you how much better off you are using the decision tree to predict some outcome compared to using the average value as a guess of the dependent variable. For example, if the variance is 8%, you will likely guess a given sample's outcome 8% more accurately using the tree than you would by predicting that it will have the average value shown in the root node of the tree.

See also:

- Model Analyzer
- Validation

Creating Calculation from Split

Every split on an independent variable V in a Decision Tree is a rule that can be represented by a conditional expression

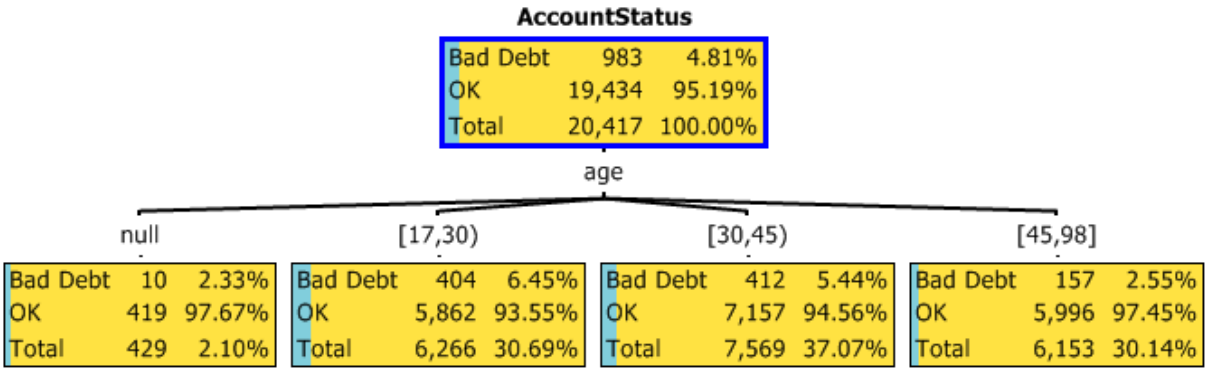
```
    CASE  WHEN  V  in  V_1  THEN  <Node  1>  WHEN  V  in  V_2  THEN  <Node
2>  ....  WHEN V in V_N THEN <Node N>  END
```

where V_1, V_2, ... V_N are subsets of the range of values of the variable V at the node being split.

Once you created a split with any tree growing command such as Automatic Grow, Find Split or Force Split, you can use the split as a binning transformation for creating a new expression for a calculated column with the conditional expression defined by the split. Thus, it can be a convenient alternative to the Binning Helper in the Dataset Editor.

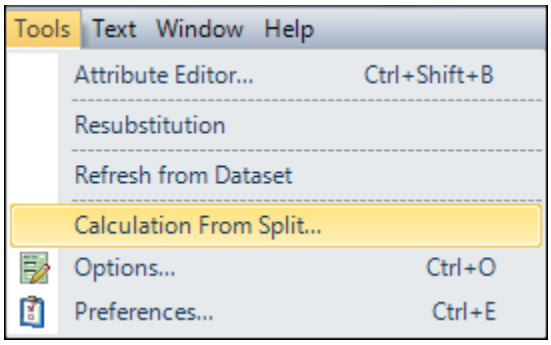
The command Calculaton from Split creates an expression based on the split and writes it to an expression file that can be subsequently used in any Data Transformation step in your workflow simply by importing it into the Dataset Editor.

For example, suppose you have a Decision Tree with the dependent variable AccountStatus with values *Bad Debt* and *OK*. If you created a split on ***age*** such as the one shown below, you may want to bin the ***age*** variable (create discrete categories for it) exactly the way it is mapped by the split, and then work with the new, binned age variable instead.

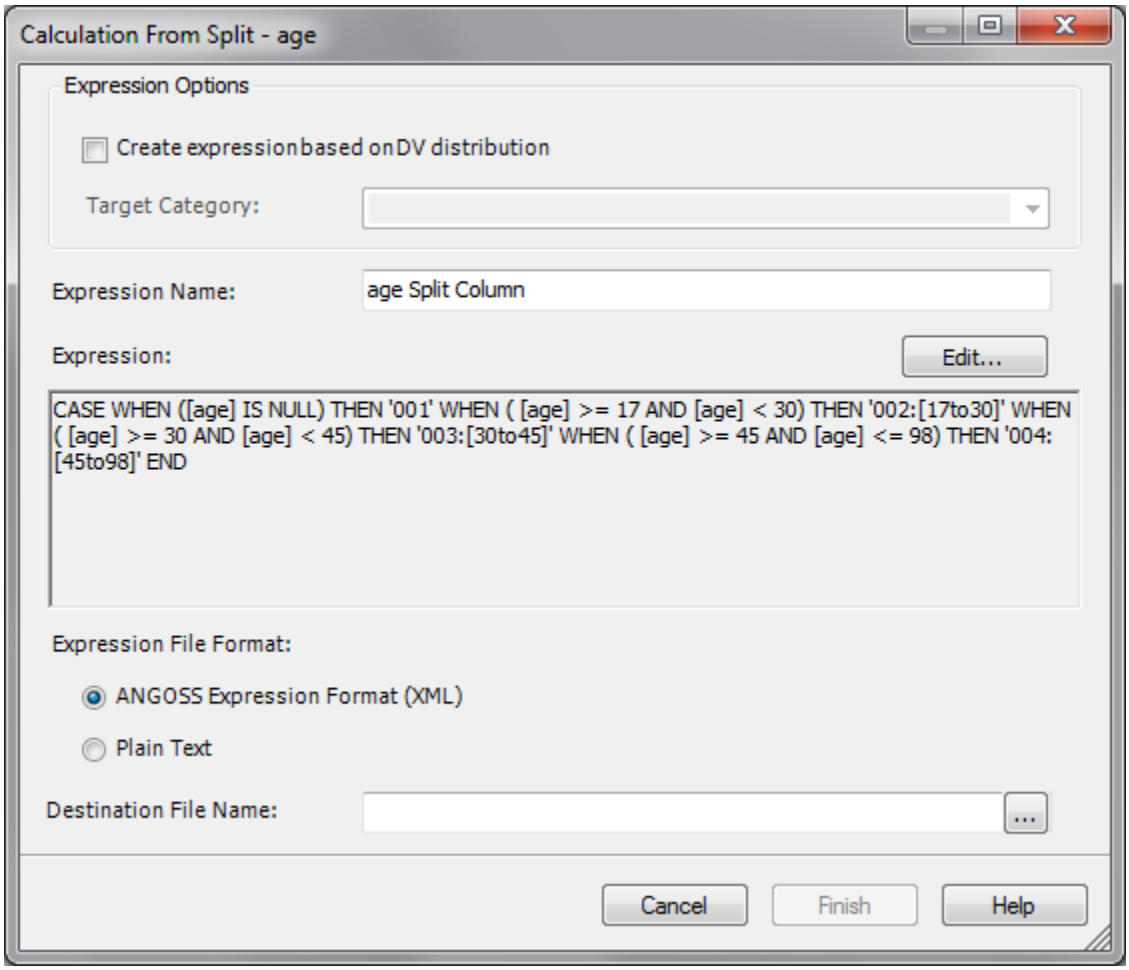


Click on the parent node in the split and select the menu command **Tools | Calculation From Split**.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



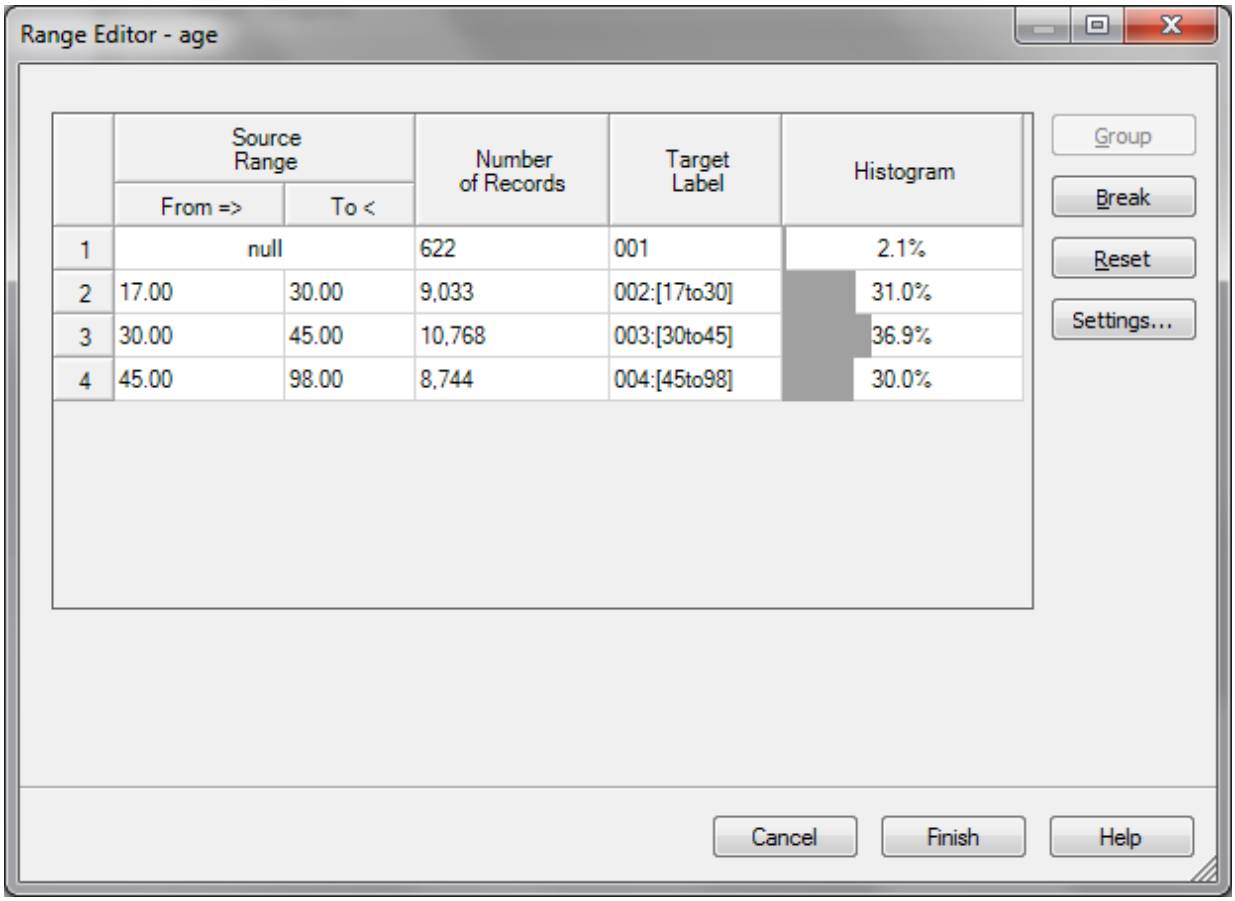
This brings up the Calculation from Split dialog. It allows you to choose the expression options, edit bins, specify the expression name, and save it in the an expression file.



By default, the new calculation is a simple binning, assigning bin labels to each bin in the split (the option *Create expression based on DV distribution* is not selected).

Click the **Edit** button if you would like to change the binning before creating the expression. This brings up the Range Editor dialog, where you can group the bins, edit their boundaries, and edit the bin labels in the Target Label column for the final expression.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



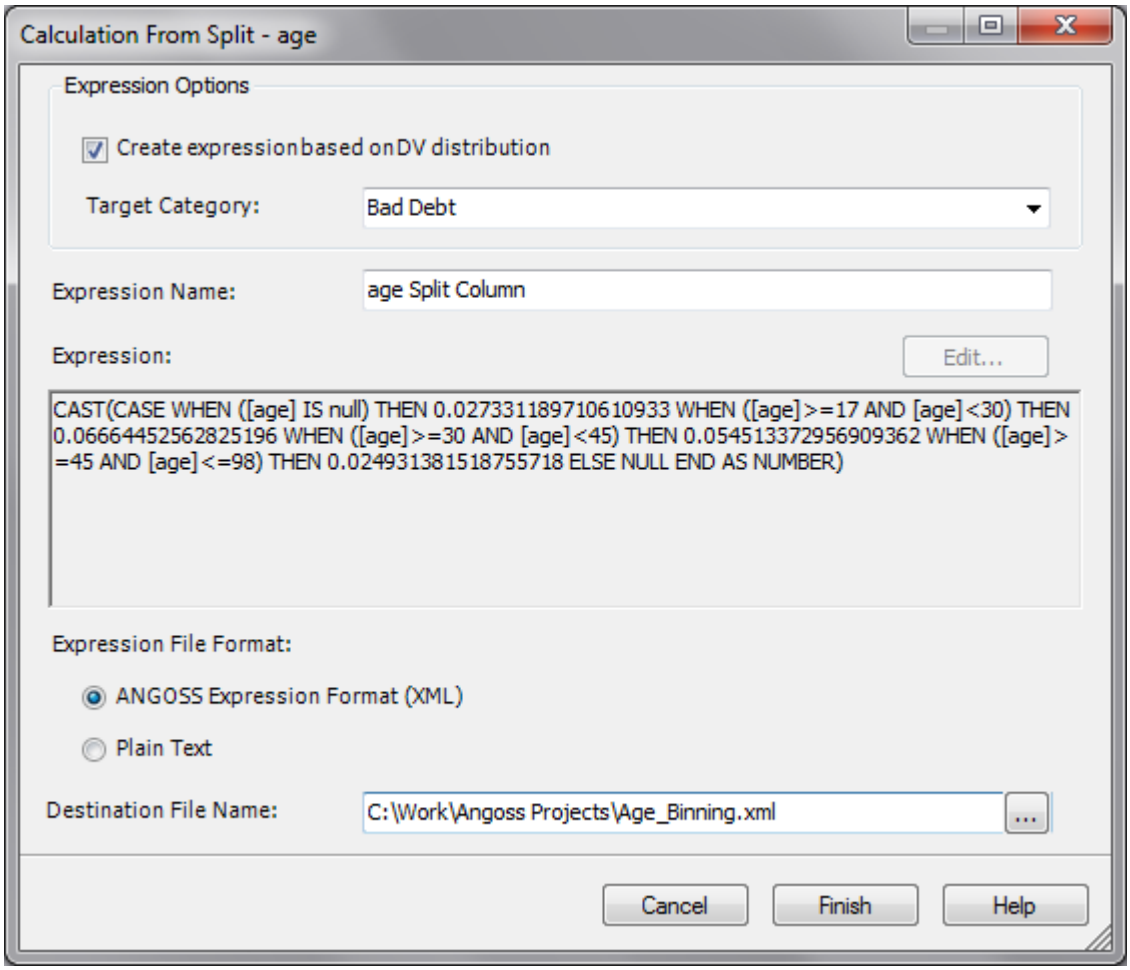
Expressions based on the DV distribution

If you want the expression to reflect the distribution of the dependent variable (represent the scoring rule of the split), choose the option ***Create expression based on DV distribution***.

Discrete DV

If the dependent variable is discrete, this option creates a calculation whose value for each bin is the probability of one of the dependent variable outcomes. Select the target category. For example, given the split shown above, the value *Bad Debt* is selected as Target Category in the screenshot below. The resulting CASE WHEN expression produces the probabilities of Bad Debt in each of the bins (child nodes) in the selected split.

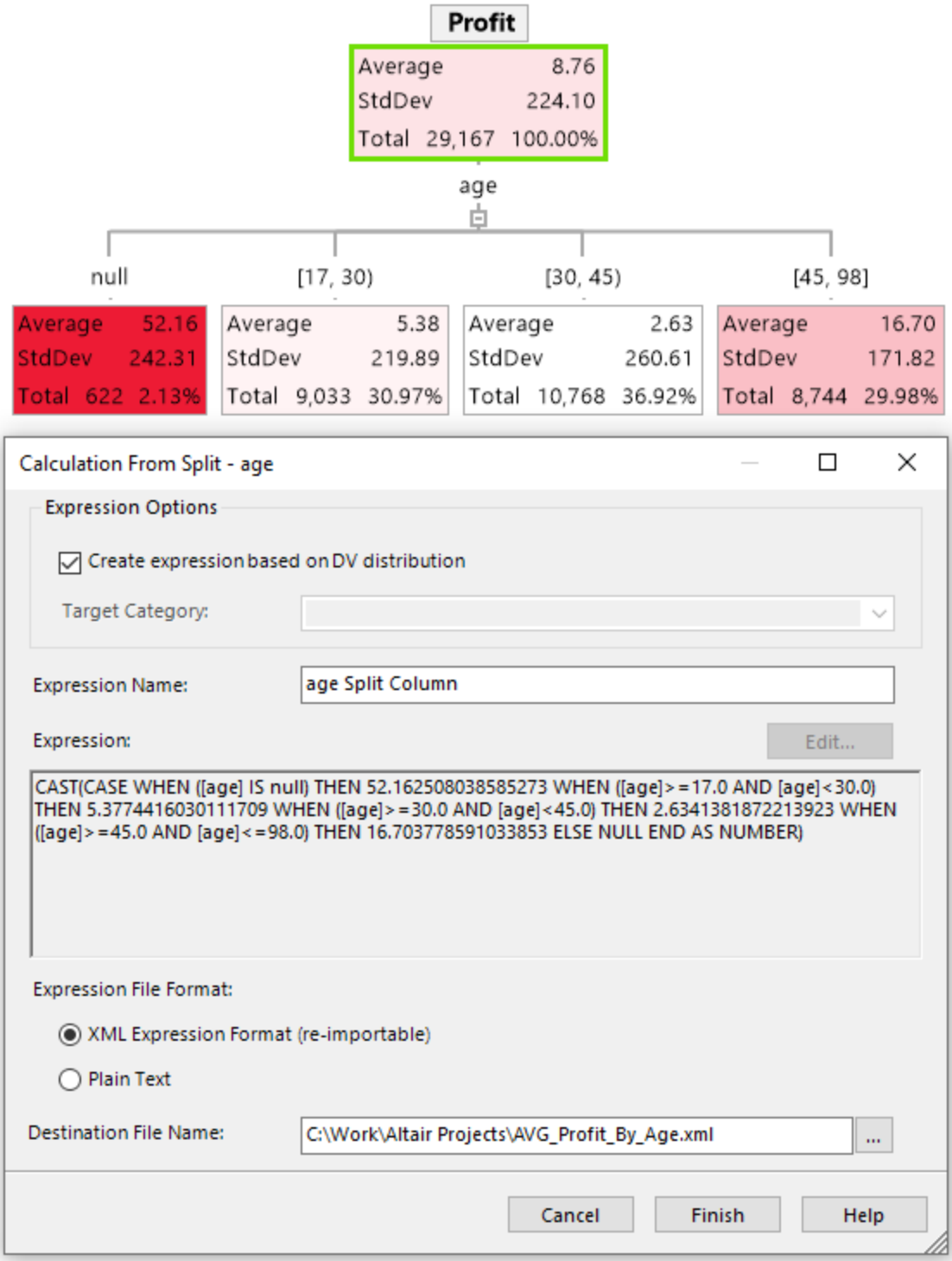
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Continuous DV

If the dependent variable is continuous, the command creates a calculation whose value for each bin is the average value of the dependent variable.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Choose the expression file format and specify the name of the output expression file. The default is the XML Expression Format (re-importable). It allows you to reuse expressions by importing the expressions file into the Dataset Editor for Data Transformation or into Node Calculations for Strategy Trees.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Click **Finish** to create the expression file. You can use it in your workflow by importing it by means of the **Import Expressions** button available in the Dataset Editor for Data Transformation or in the Calculations dialog of Strategy Trees.

See also:

- Find Split
- Force Split
- Edit Split
- Variable Transformations Node
- Function Syntax

Ensemble Trees

Instead of creating a single decision tree model, you can create multiple decision trees in one go and combine them in a single “ensemble” tree model. An ensemble of tree models can perform better than individual trees. The trees within ensembles are built on multiple samples, and their predictions are aggregated at the scoring stage. Ensemble models are therefore less prone to overfit than individual models (the problem of a model performing well on its training sample but poorly on other datasets).

Three types of Ensemble Trees are available:

- Bagging (Bootstrap Aggregating)
- Random Forest
- Boosting

Of these three types, **Bagging** models are the simplest. In Bagging models, a number of decision trees are grown using automatic growth features. The decision trees are not all identical because the data that are used to train the trees are not the same for each tree. Suppose there are N records in the training data being used to build the Bagging model. Then for each decision tree, there will be N records chosen (sampled) with replacement from the total N records. That means that some records may appear more than once, and some records won’t appear at all. This technique is known as Bootstrapping. Bagging stands for Boostrap Aggregating. Note that the Bagging wizard allows the user to modify the sample size and to turn off Sampling with Replacement. If Sample Size (%) is not set to 100%, Sample with Replacement can be deselected. The predictions of these trees are averaged to create the final Bagging model.

Boosting models begin with a single model and iteratively create new models based on the previous model, assigning a greater weight to those records incorrectly predicted in the previous iteration. In practice, after each tree in the Boosting model is trained, the error from the previous model is computed. A weight vector is assigned which depends on how well the previous model performed; where the previous model made a correct prediction the weight for the next model is decreased, and where the previous model made an error the weight for the next model is increased.

New trees are trained on the weighted data, meaning that each tree will be more likely to do a good job on the subsets of the data where the previous tree did poorly. However, each tree in the process is more specialized and should therefore have less weight than previous trees when the predictions are all averaged. This is handled by a shrinkage factor, which is user-editable in the wizard– larger

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

values of the shrinkage factor result in a more aggressive model. The end result is a single Boosting model with the ability to make predictions on new data. New data is scored by averaging the predictions across all trees.

Random Forest models are similar to Bagging models, but slightly more complicated. Bagging involves the creation of a large number of decision trees using bootstrapped data, and variables are the same for all trees. In Random Forest models, the variables included in the model are also sampled. Each decision tree will be trained with only a subset of all the variables available to that specific tree.

While this sampling of variables makes the individual trees in a Random Forest model simpler, and likely weaker on average, it can also mean that there is more variation between the trees in the model, and in many cases this can lead to better predictions on new data used when the model is actually being deployed. In addition, the user can specify the proportion of variables to use for each sample. The defaults - sampling with replacement with a record sample size equal to the total number of records, and a variable sample size of 33% - are almost always good practice, and usually understood to be the situation when discussing Bagging.

As in Bagging, the final model is obtained by averaging the results of the large number of simple trees.

There are two tabs in final results:

- Results: A simple overview of the variables used
- Parameters and Attributes: Basic information about how the parameters used in the model

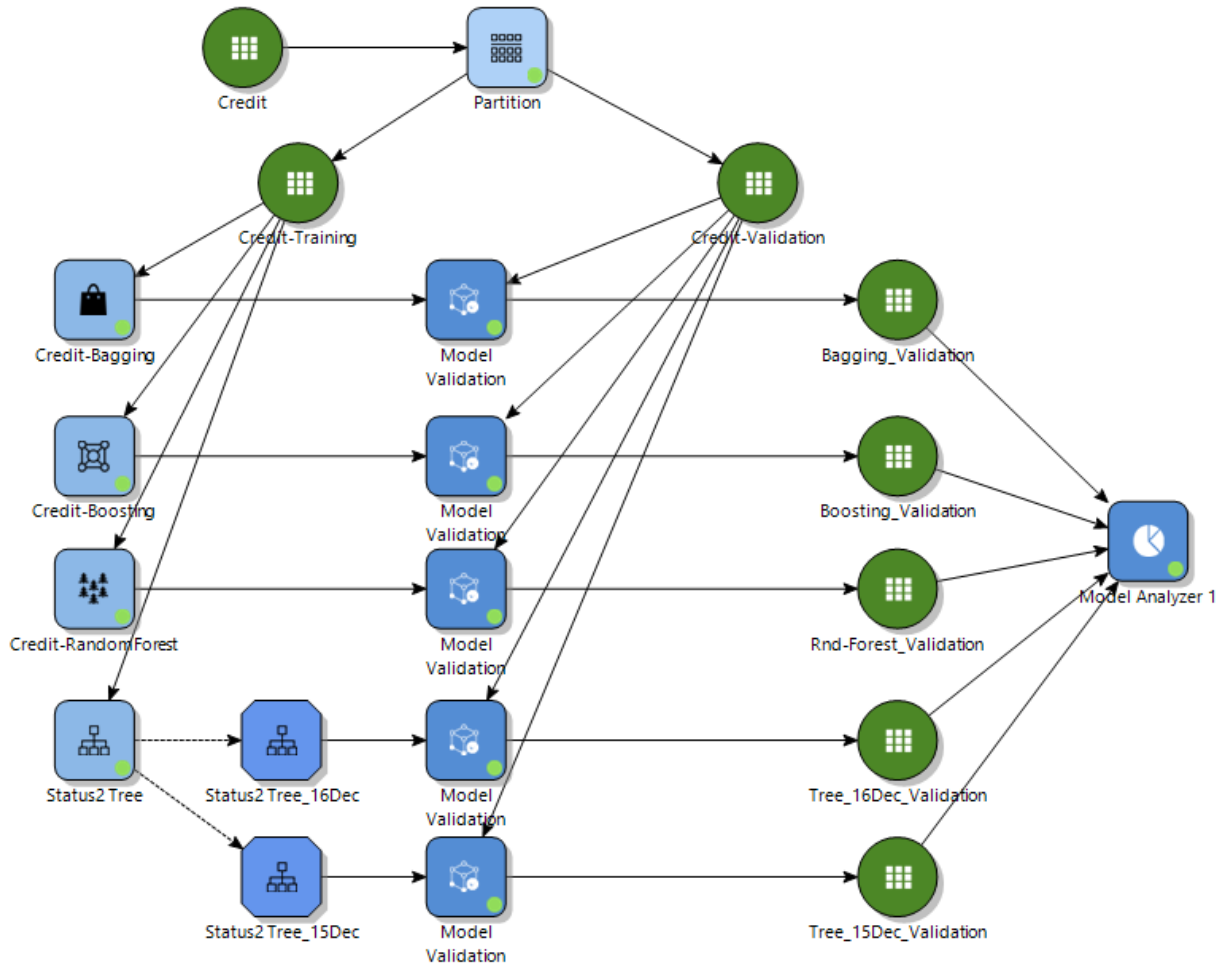
Both sets of output results are simply reflecting the model set up prior to running. As a direct result of building many models and combining results, no additional output is created.

In fact the output from ensemble models is minimal to such an extent that scoring and model evaluation are the only recourse to assess model performance.

Scoring the model is accomplished by using a Scoring node from the Deployment palette or Model Validation Node from the **Evaluate** palette. You can assess and compare the performance of models using the **Model Analyzer** node from the **Evaluate** palette.

An example workflow is shown below:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Proceed to the topics: Bagging (Bootstrap Aggregating), Random Forest, Boosting

Bagging (Bootstrap Aggregating)

Bagging (Bootstrap aggregating) is one of the simplest ensemble models. It creates many different bootstrapped samples of the data and grows a decision tree on each one of them. At the deployment stage, the predictions of the trees within the ensemble are averaged to generate the final probabilities.

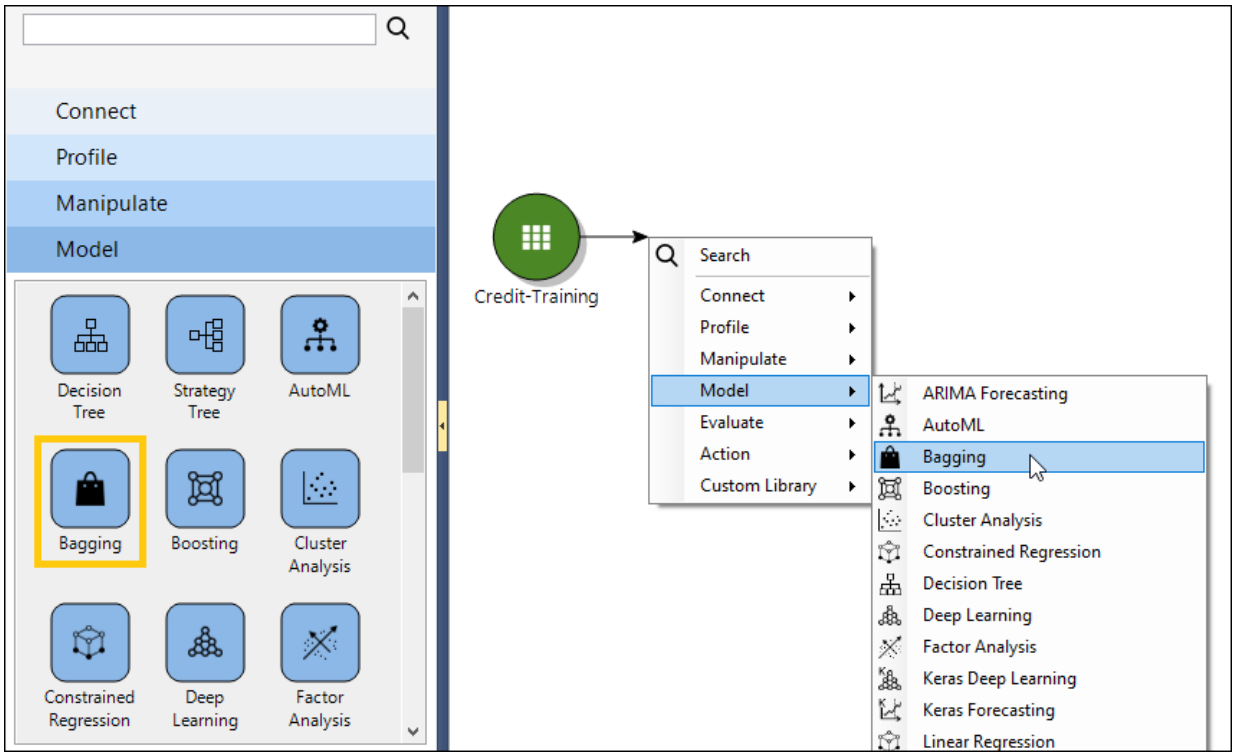
To insert a Bagging operation in your workflow:

- Drag a connection arrow from a dataset node to any area on the Workflow canvas background. This brings up a context menu that allows you to select the desired operation. From the context menu, select **Model --> Bagging** as shown in the example below. A new **Bagging** process node will be created.

  OR:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Drag the **Bagging** node from the **Model** palette onto the canvas and connect the required input dataset to the node.



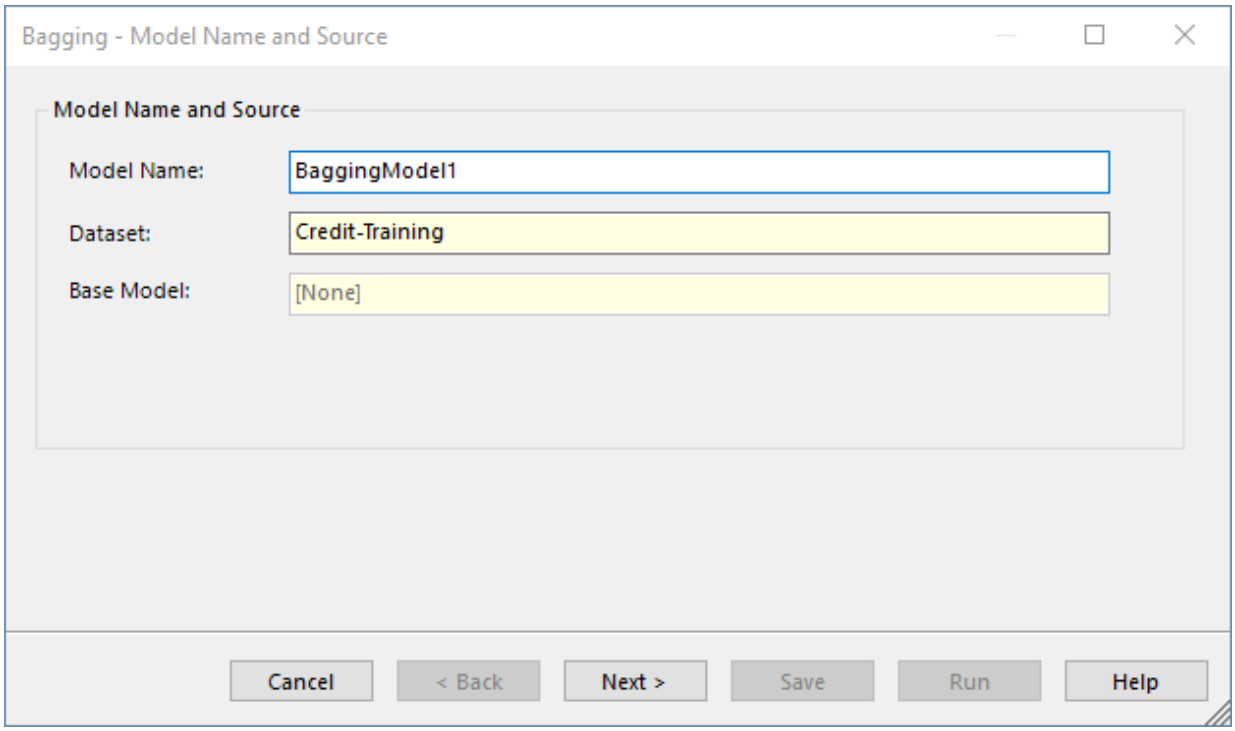
Double-click the Bagging node or right-click on it and select **Modify** to open the wizard.



In the first page, specify the model name and select the base model if applicable (in that case the base model must be connected to the Bagging node).

If you connect a base model as a template, the input variables and all applicable training parameters and attributes of the base model will be copied and used for the new model.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

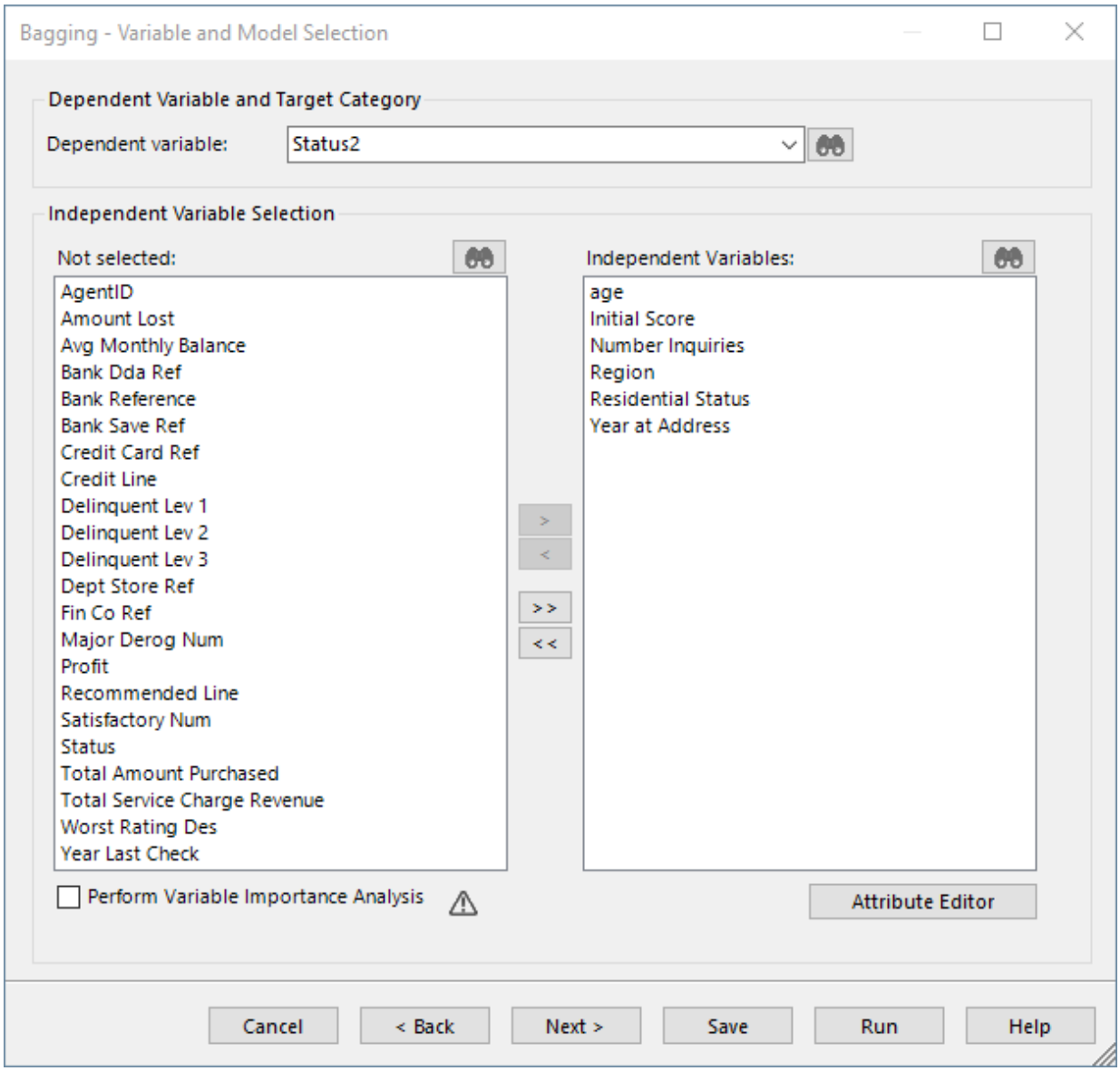


Click Next to select the variables and specify variable selection options.

In the Variable Selection page:

- Select the dependent variable and the independent variables for your model
- Select the option **Perform Variable Importance Analysis** if necessary. Variable importance is analyzed using accuracy (in the classification case), MSE (in the regression case), Entropy Variance and Gini Variance. The importance of a variable is checked by permuting its values on the input data (the out-of-bag sample of each tree in the ensemble), scoring the resulting records and seeing how much worse or better the model performed. By comparing key model metrics on the original records with those same metrics on the records with permuted data, the algorithm determines if a variable is important to the model.
- Use the **Attribute Editor** button to open the Tree Attribute Editor to adjust model attributes such as missing value substitution options, default binning, etc.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

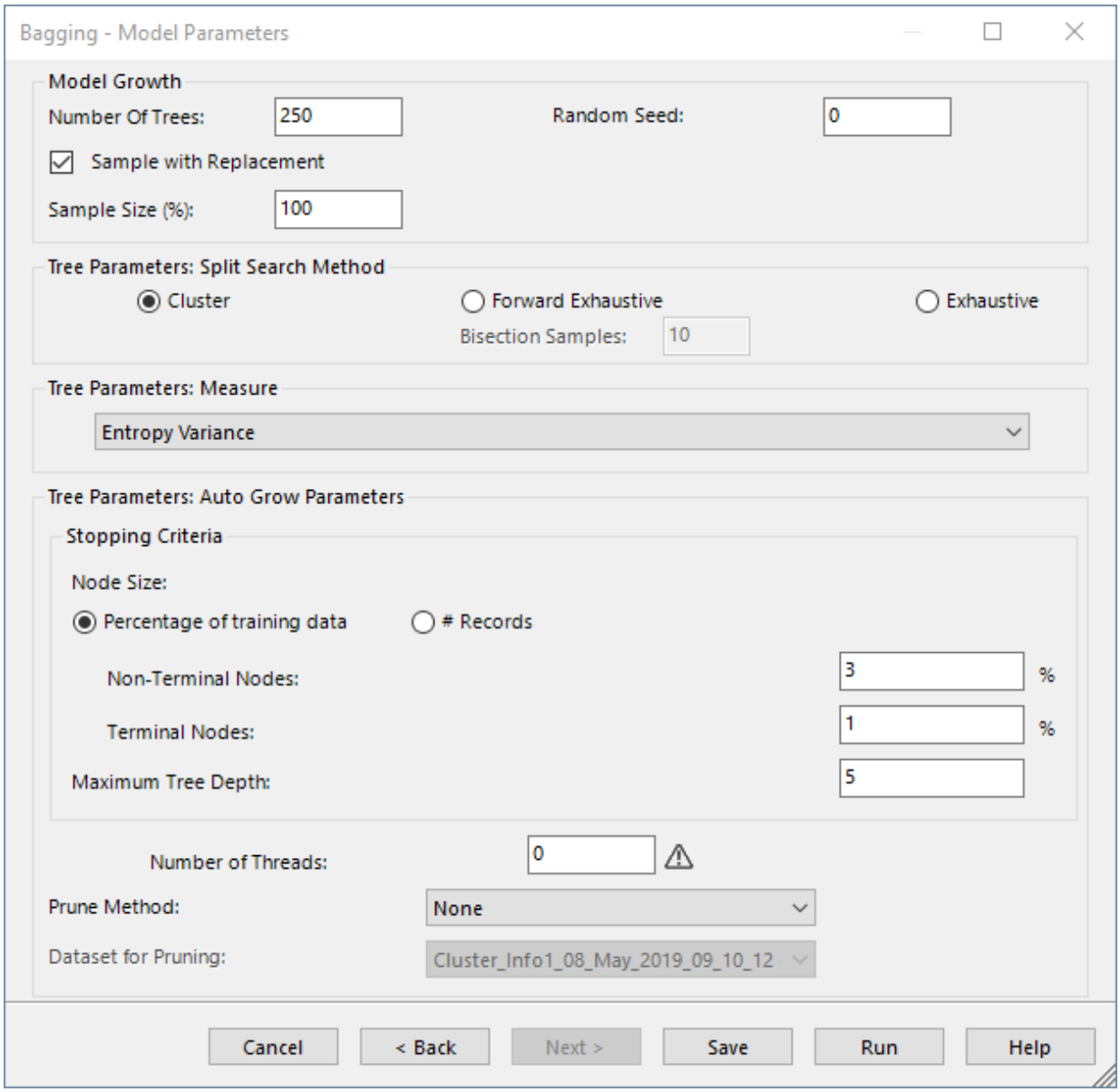


In the next page, select the following parameters:

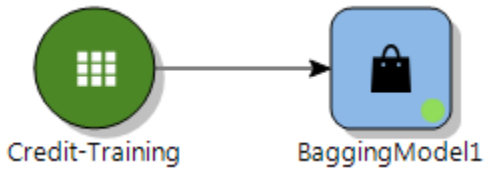
- Number of trees (the default is 250)
- Sample with Replacement option (selected by default)
    - If you specify 100% for the Sample Size, you must select Sample with Replacement.
- Decision Tree growing parameters:
    - Split search method
    - Measure
    - Auto-Grow stop size (minimum node size to split)
    - Minimum node size
    - Maximum depth of the tree

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

For detailed information on the training parameters, see the topics Measure and Decision Tree: Training Parameters.

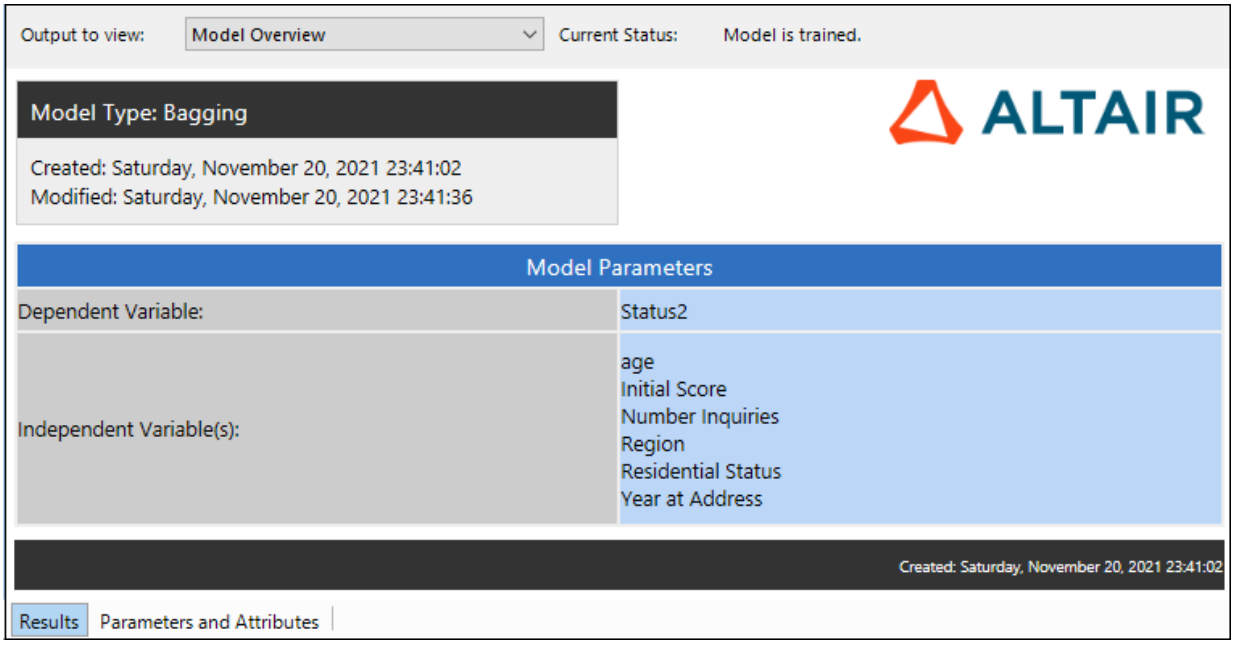


Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.

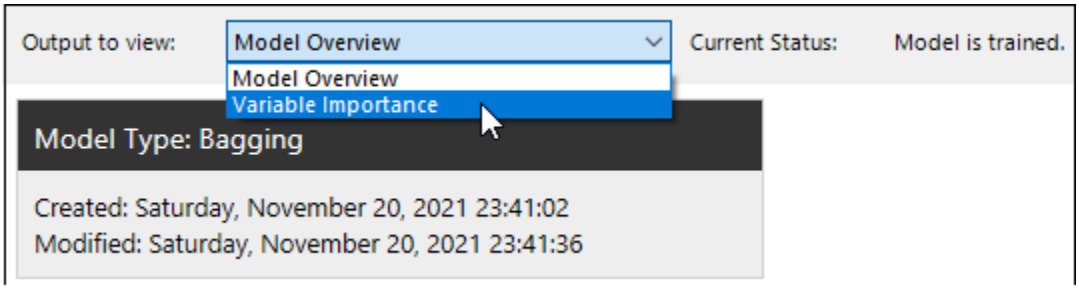


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

When the model training has completed, double-click the model node to view the report. The **Results** tab shows the dependent variable and the independent variables of the model in the Model Overview.

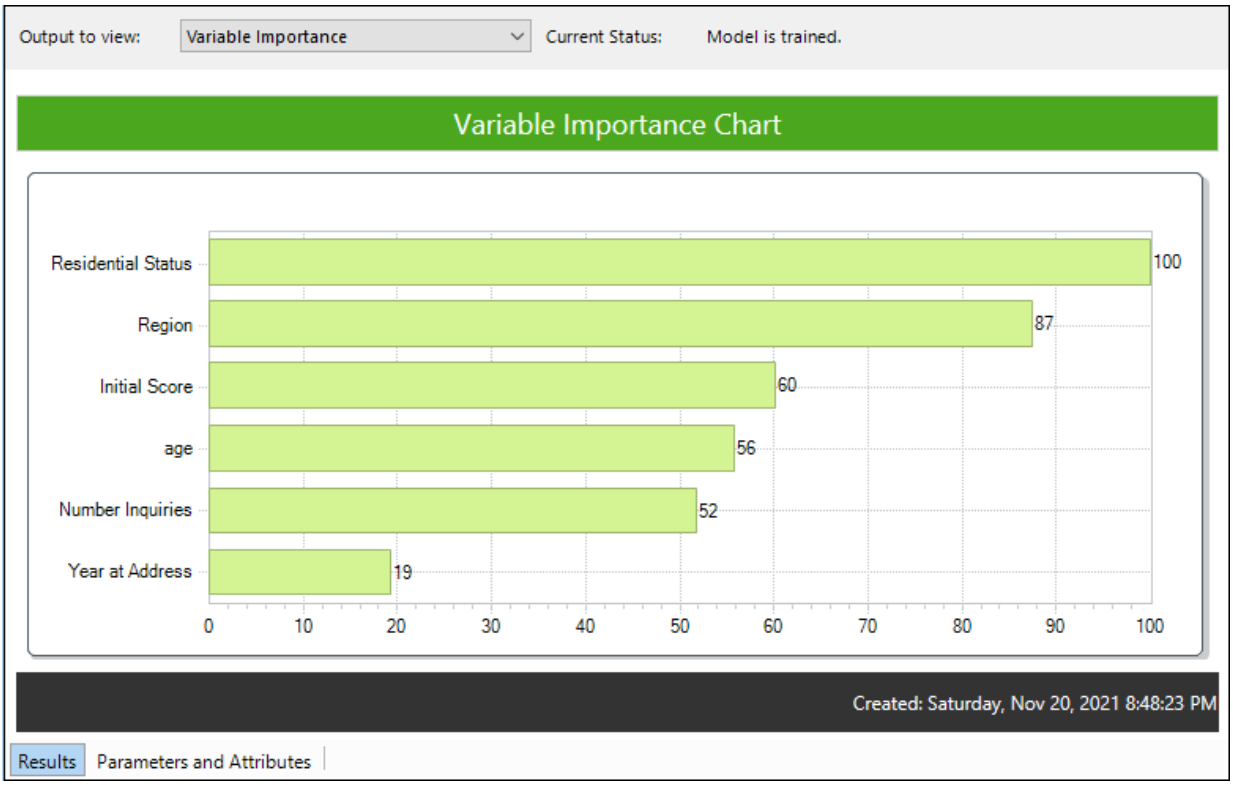


If you selected the **Perform Variable Importance Analysis** option in the model wizard, select **Variable Importance** from the *Output to view* drop-down list at the top of the Results tab:



The Variable Importance Chart will be displayed as a result. The variables are ranked by scaling the average losses of the key metrics over the ensemble for each variable.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Random Forest

The Random Forest technique is an evolution of the bagging algorithm. The algorithm creates many different trees using changing features and different samples of the data, grows a tree in each sample, scores the probabilities and averages the results to generate the final probabilities.

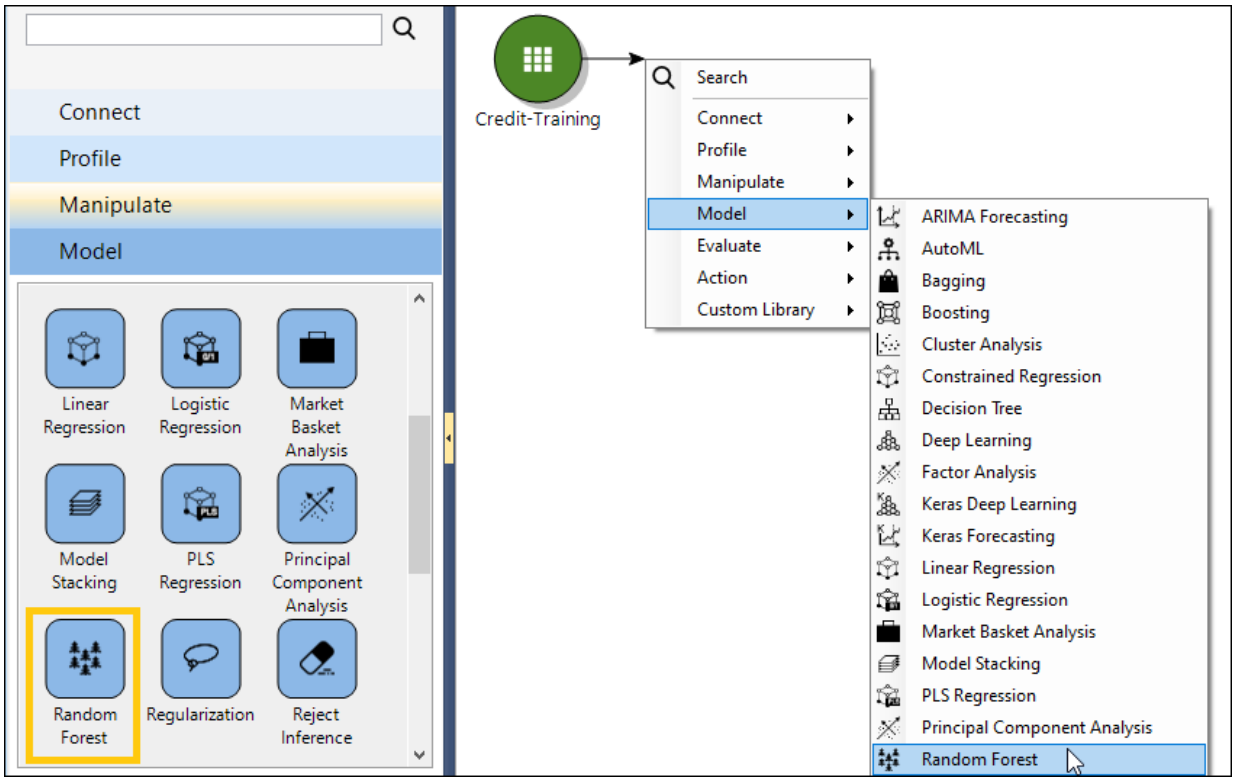
To insert a Random Forest operation in your workflow:

- Drag a connection arrow from a dataset node to any area on the Workflow canvas background. This brings up a context menu that allows you to select the desired operation. From the context menu, select **Model --> Random Forest** as shown in the example below. A new **Random Forest** process node will be created.

  OR:

- Drag the **Random Forest** node from the **Model** palette onto the canvas and connect the required input dataset to the node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



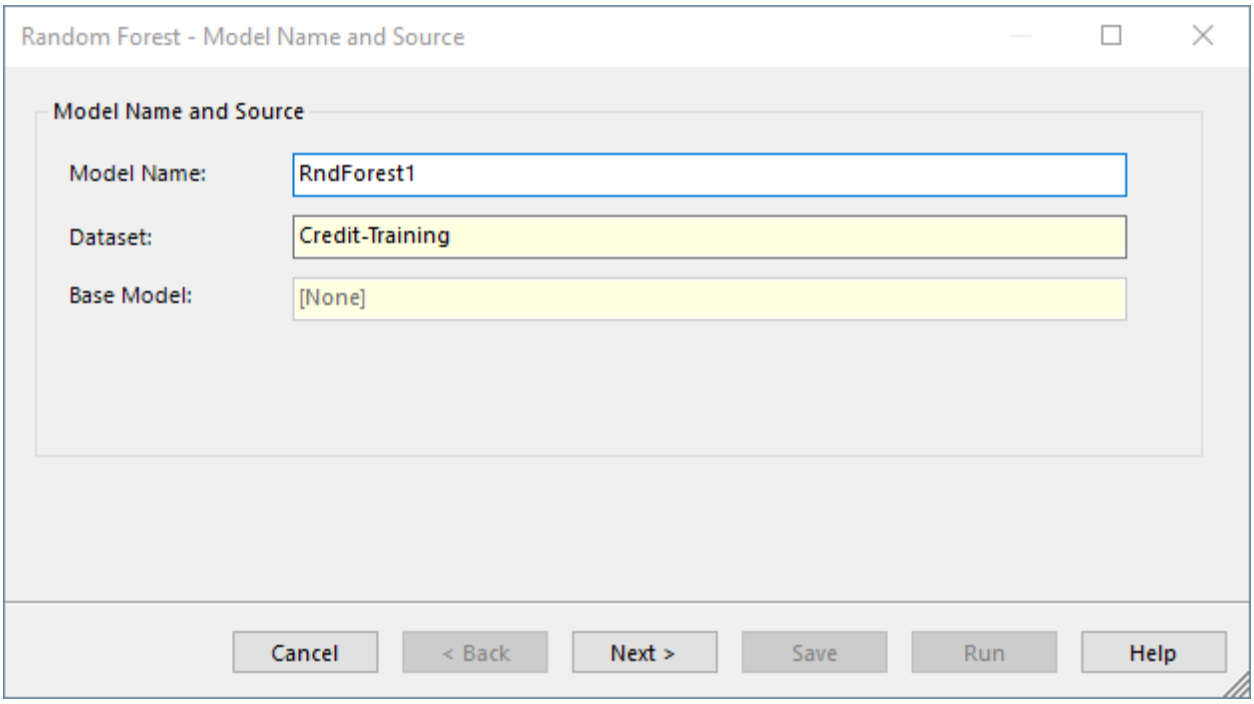
Double-click the Random Forest node or right-click on it and select **Modify** to open the wizard.



In the first page, specify the model name and select the base model if applicable (in that case the base model must be connected to the Random Forest node).

If you connect a base model as a template, the input variables and all applicable training parameters and attributes of the base model will be copied and used for the new model.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

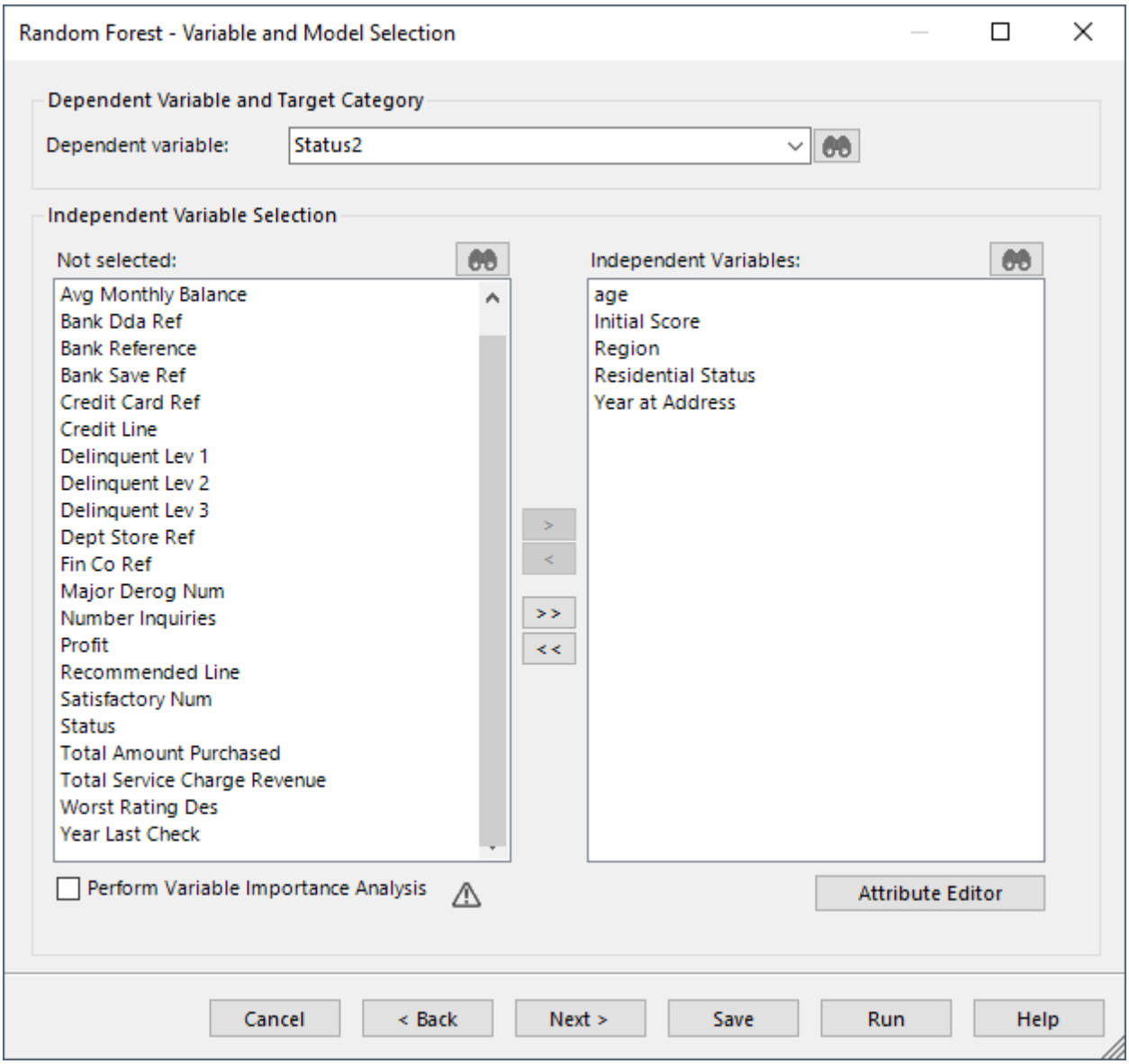


Click Next to select the variables and specify variable selection options.

In the Variable Selection page:

- Select the dependent variable and the independent variables for your model
- Select the option **Perform Variable Importance Analysis** if necessary. Variable importance is analyzed using accuracy (in the classification case), MSE (in the regression case), Entropy Variance and Gini Variance. The importance of a variable is checked by permuting its values on the input data (the out-of-bag sample of each tree in the forest), scoring the resulting records and seeing how much worse or better the model performed. By comparing key model metrics on the original records with those same metrics on the records with permuted data, the algorithm determines if a variable is important to the model.
- Use the **Attribute Editor** button to open the Tree Attribute Editor to adjust model attributes such as missing value substitution options, default binning, etc.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



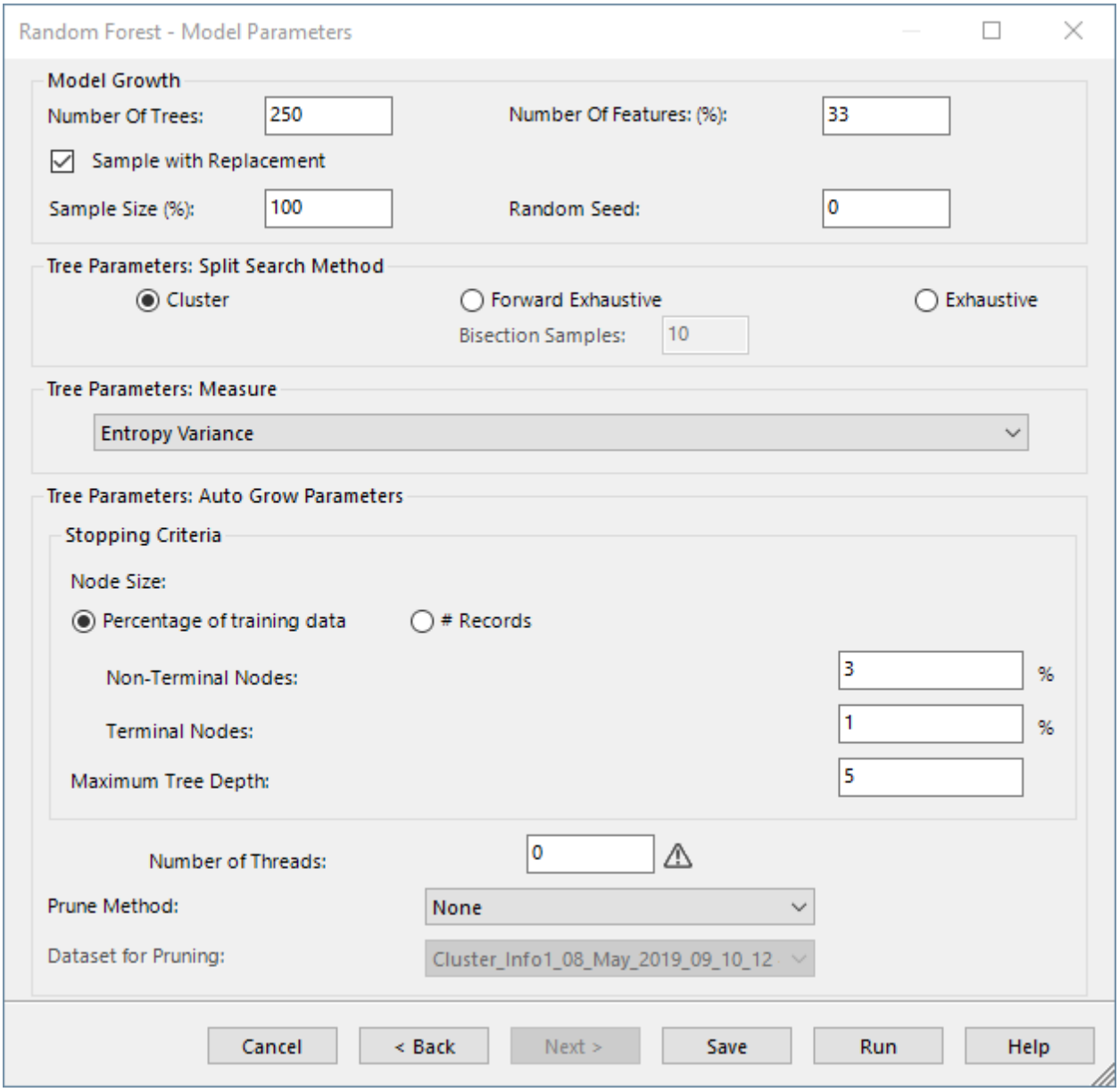
Click Next. In the next page, select the following parameters:

- Number of trees (the default is 250)
- Number of features (the default is 33%)
- Sample with replacement (selected by default)
    - If you specify 100% for Sample Size, you must select Sample with Replacement.
- Decision Tree training parameters:
    - Split search method
    - Measure
    - Auto-Grow stop size (minimum node size to split)
    - Minimum node size

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Maximum depth of the tree

For detailed information on the training parameters, see the topics Measure and Decision Tree: Training Parameters.

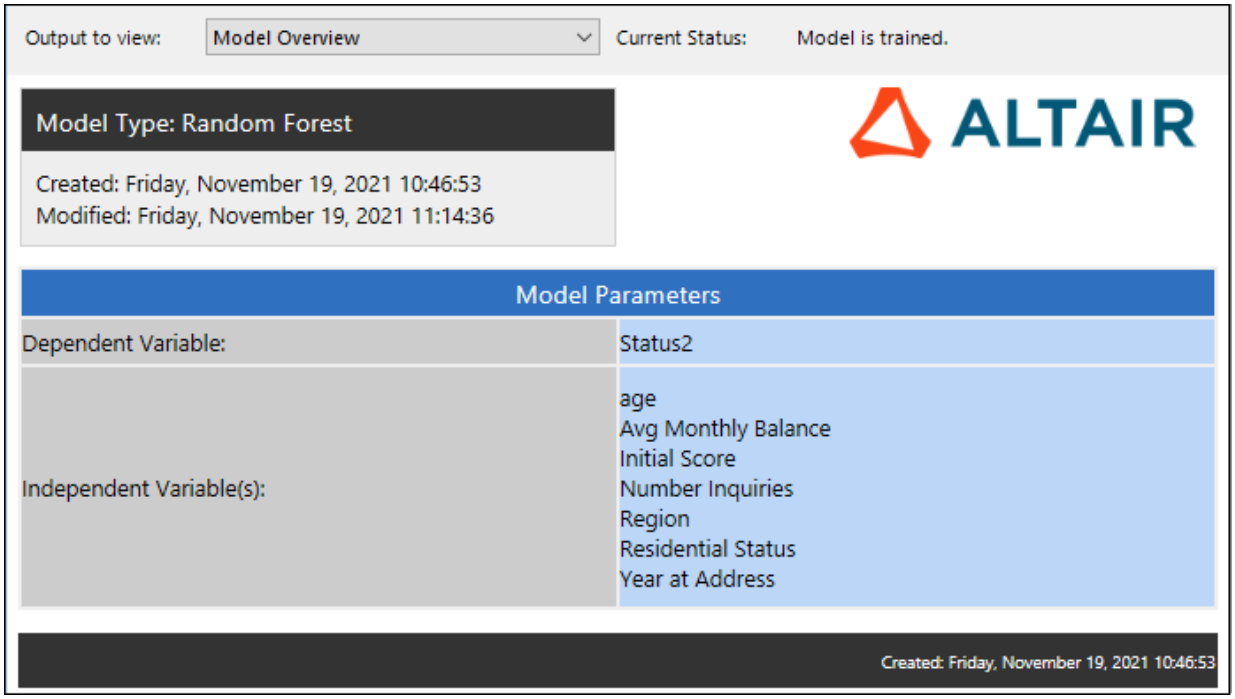


Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.

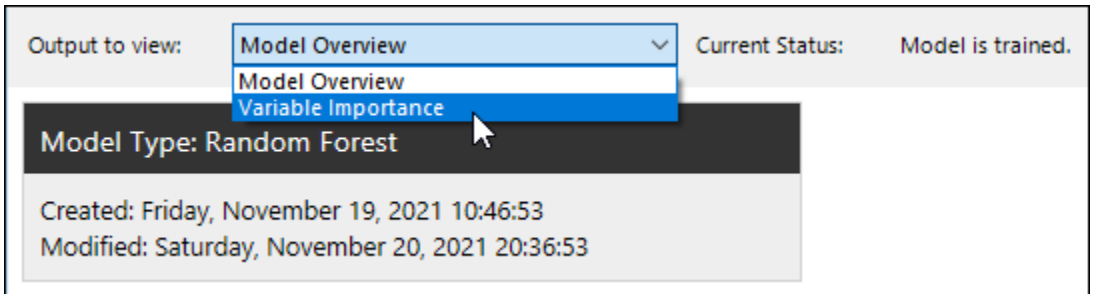


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

When the model training has completed, double-click the model node to view the report. The **Results** tab shows the dependent variable and the independent variables of the model in the Model Overview.

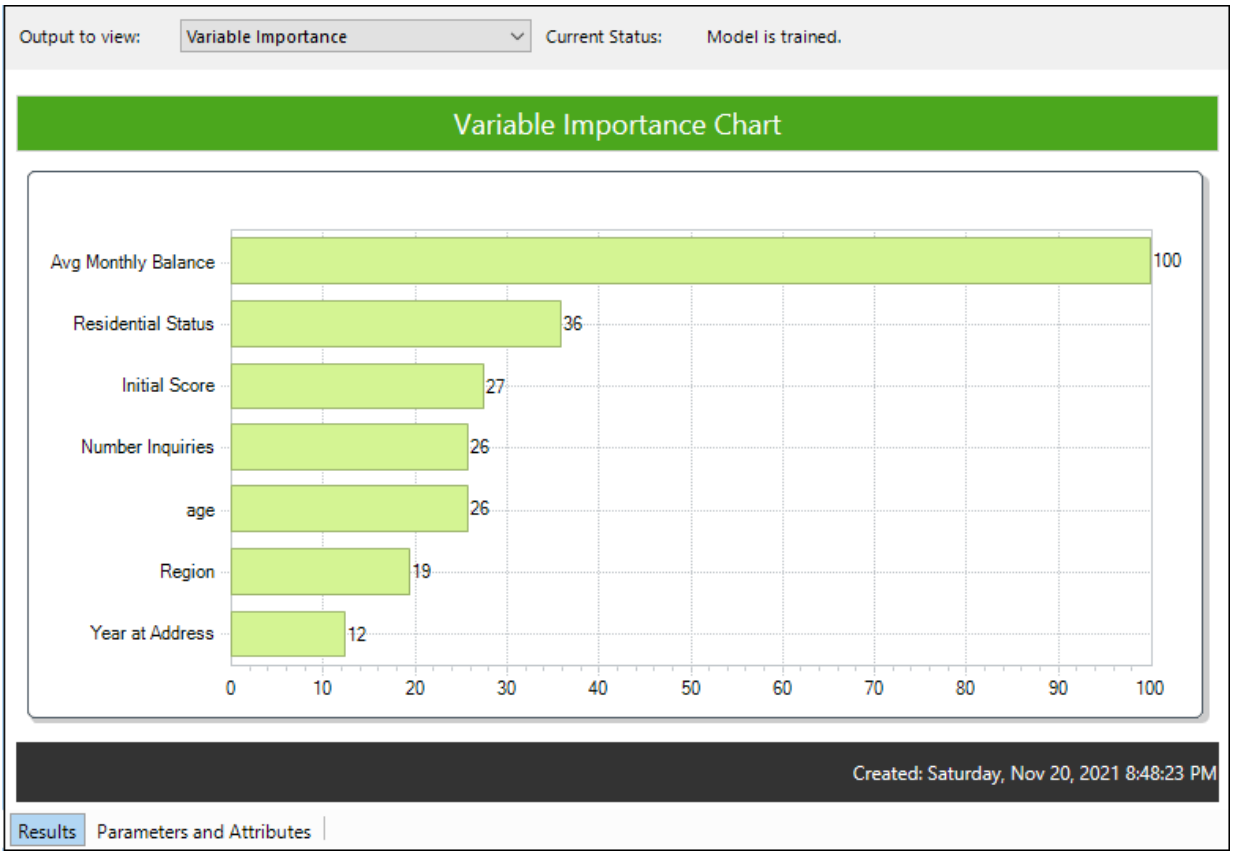


If you selected the **Perform Variable Importance Analysis** option in the model wizard, select **Variable Importance** from the *Output to view* drop-down list at the top of the Results tab:



The Variable Importance Chart will be displayed as a result. The variables are ranked by scaling the average losses of the key metrics over the forest for each variable.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Boosting

Unlike Bagging and Random Forest, in **Boosting** the iterations are not independent from each other. Every tree is grown based on the input from the previous tree.

The general outline of the algorithm is as follows:

1. Generate a vector of weights
2. Generate a vector of alphas that is going to weight every tree
3. For i=1 to the maximum number of iterations specified:
    - grow tree i weighted by the weight vector;
    - score the tree;
    - update the weight vector increasing the importance of the misclassified points;
    - generate the alpha for the tree based on the error ratio.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

4. Return a set of trees and their corresponding alphas for weighting

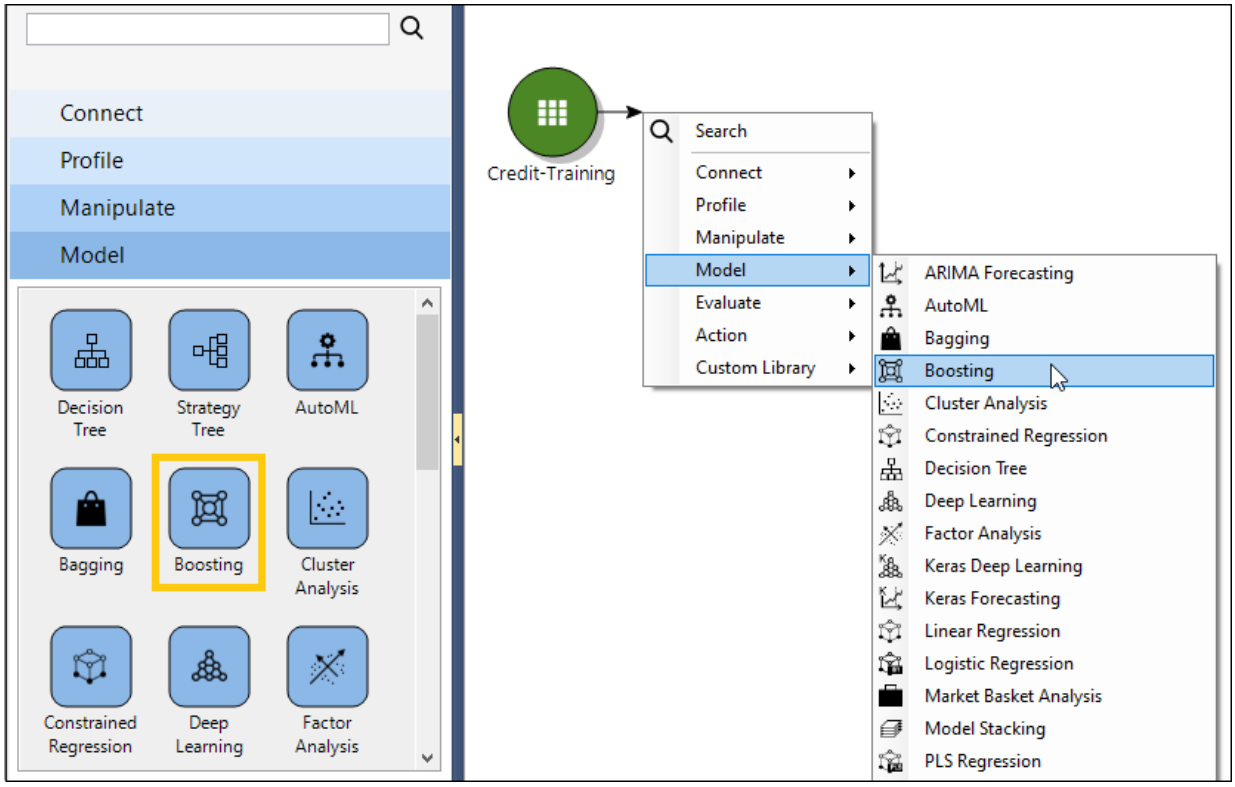
The implementation is based on the AdaBoost loss function, where AdaBoost stands for "Adaptive Boosting".

To insert a Boosting operation in your workflow:

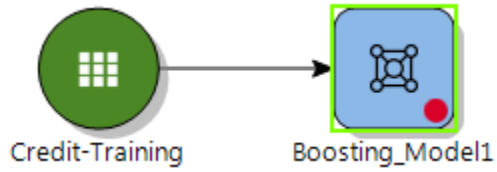
- Drag a connection arrow from a dataset node to any area on the Workflow canvas background. This brings up a context menu that allows you to select the desired operation. From the context menu, select **Model --> Boosting** as shown in the example below. A new **Boosting** process node will be created.

OR:

- Drag the **Boosting** node from the **Model** palette onto the canvas and connect the required input dataset to the node.



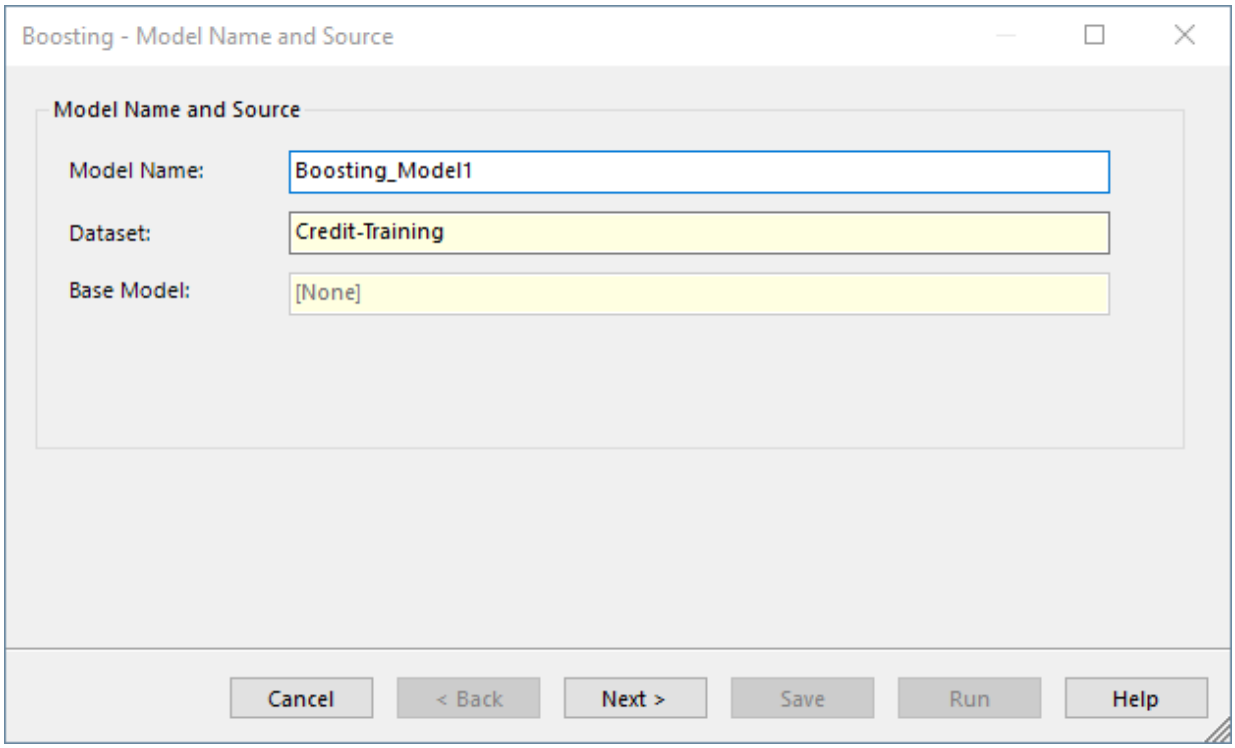
Double-click the Boosting node or right-click on it and select **Modify** to open the wizard.



In the first page, specify the model name and select the base model if applicable (in that case the base model must be connected to the Boosting node).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If you connect a base model as a template, the input variables and all applicable training parameters and attributes of the base model will be copied and used for the new model.

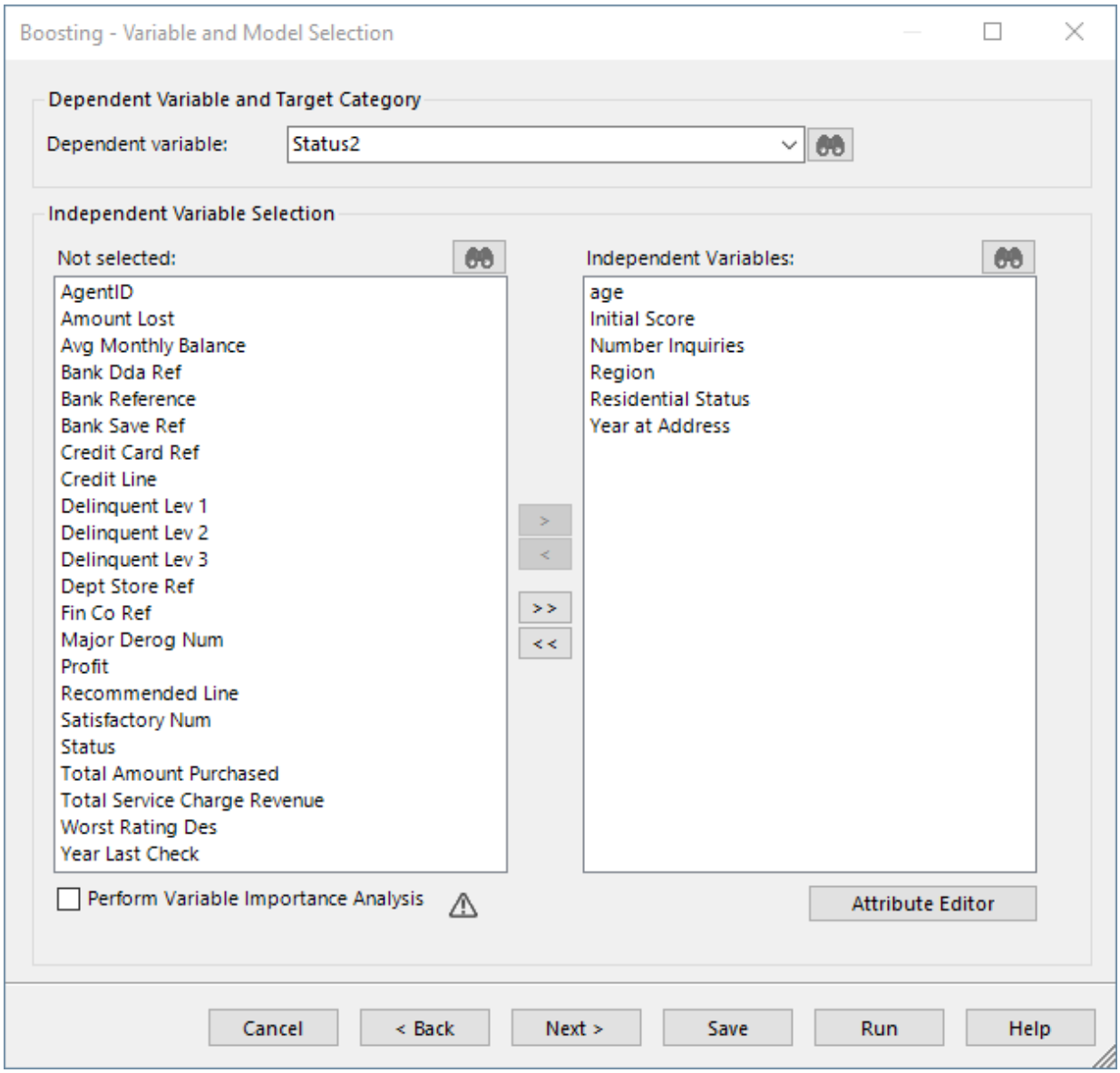


Click Next to select the variables and specify variable selection options.

In the Variable Selection page:

- Select the dependent variable and the independent variables for your model
- Select the option **Perform Variable Importance Analysis** if necessary. Variable importance is analyzed using Entropy variance and Gini variance measures.
- Use the **Attribute Editor** button to open the Tree Attribute Editor to adjust model attributes such as missing value substitution options, default binning, etc.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



In the next page, select the following parameters:

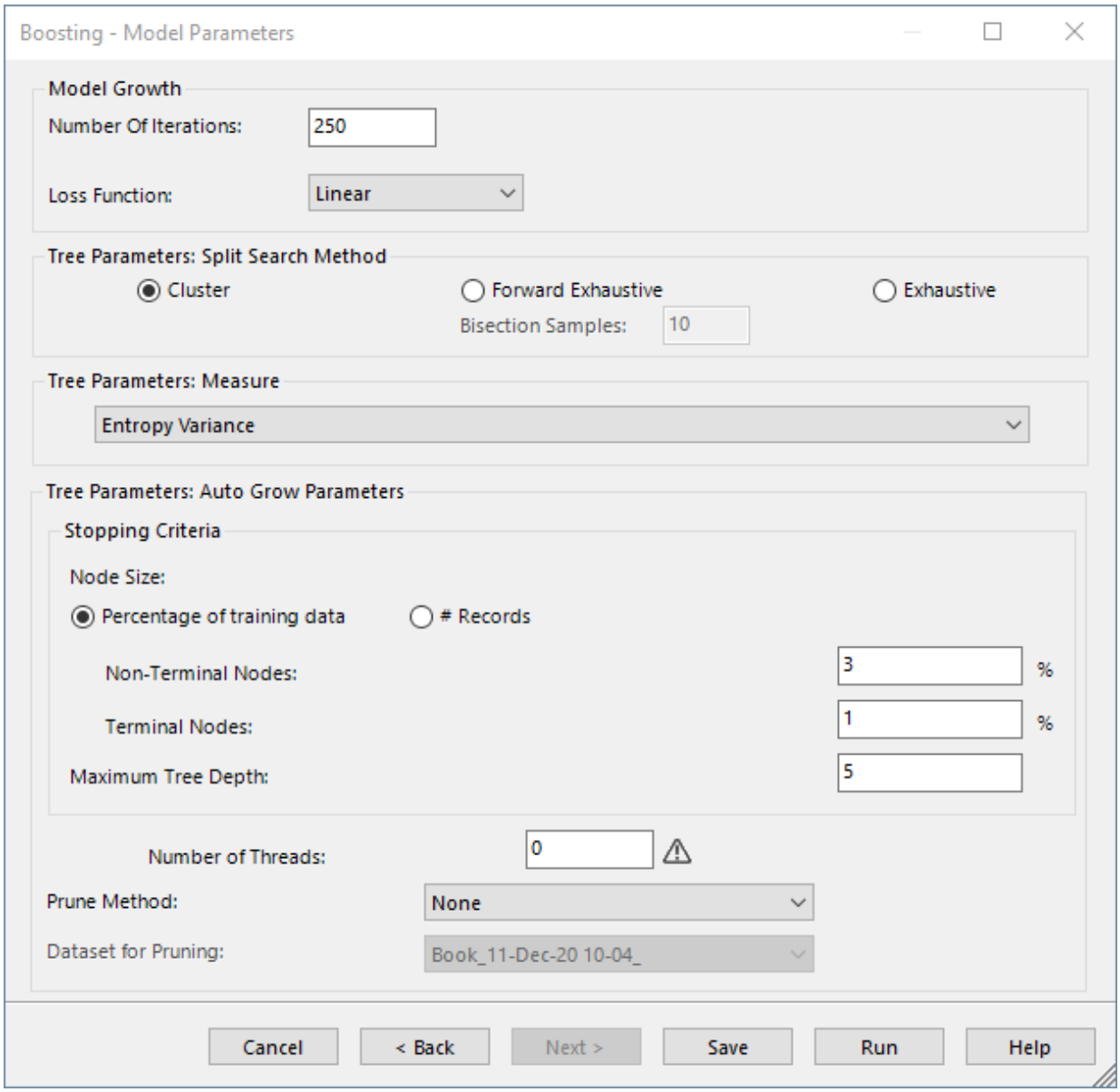
- Number of iterations (the default is 250)
- Loss function (*linear*, *square*, or *exponential*)
- Decision Tree growing parameters:
  - Split search method
  - Measure
  - Auto-Grow stop size (minimum node size to split)
  - Minimum node size
  - Maximum depth of the tree
  - Number of threads (to improve performance by parallelizing the computations).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- The default value of 0 means automatic allocation of the number of threads deemed optimal and most efficient for this task and hardware. Specify the required number of threads if necessary. Note that the number of threads exceeding the number of CPU cores does not necessarily provide performance benefits.

- Prune method and the dataset for pruning.

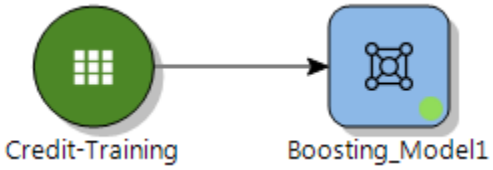
For detailed information on the training parameters, see the topics Measure and Decision Tree: Training Parameters.



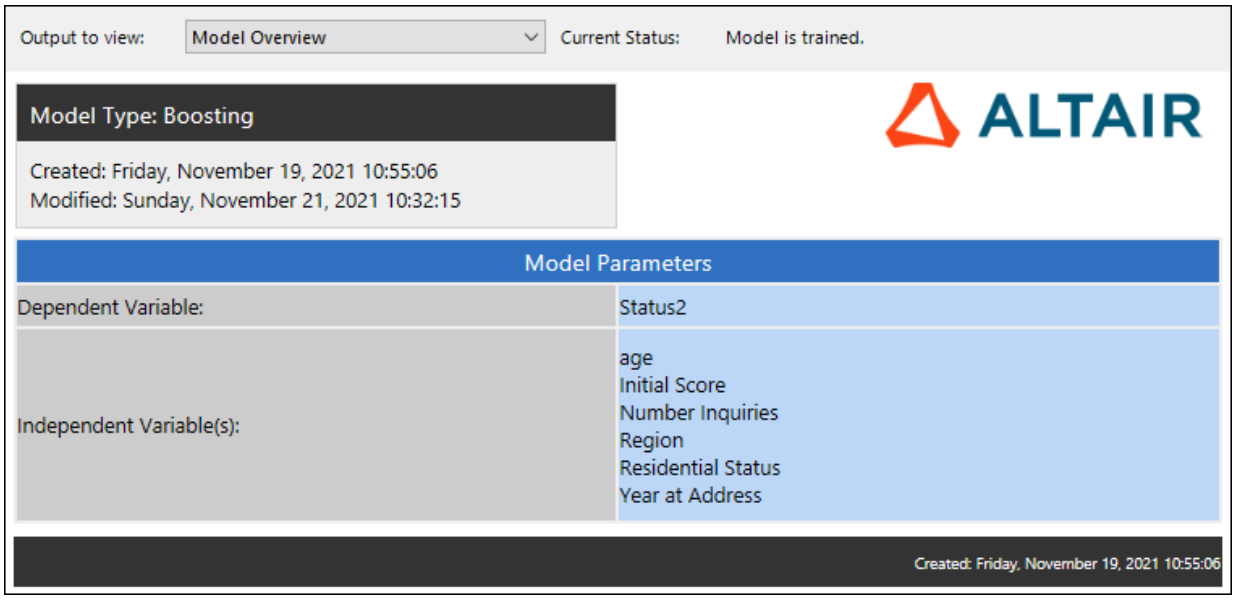
Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.

The resulting boosting model is created in the workflow after running the node. Double-click the resulting node to view the results.

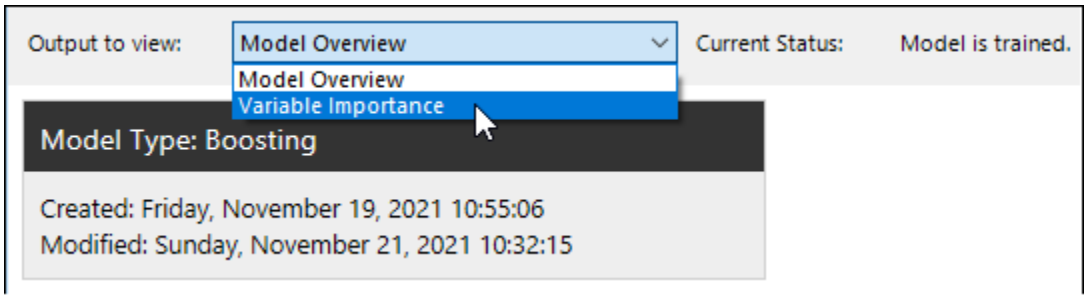
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



When the model training has completed, double-click the model node to view the report. The **Results** tab shows the dependent variable and the independent variables of the model in the Model Overview.

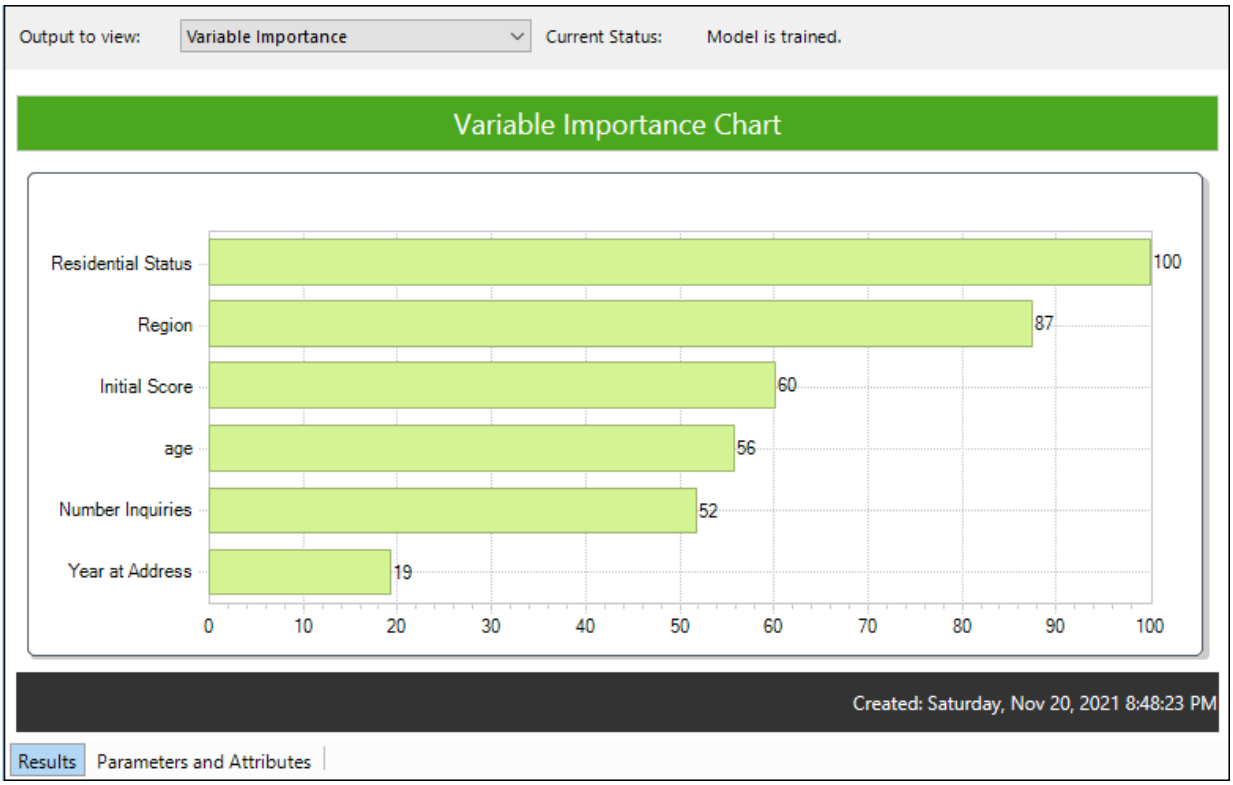


If you selected the **Perform Variable Importance Analysis** option in the model wizard, select **Variable Importance** from the *Output to view* drop-down list at the top of the Results tab:



The Variable Importance Chart will be displayed as a result. The variables are ranked by scaling the average losses of the key metrics over the ensemble for each variable.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Predictive Models

Aside from Decision Trees and Ensemble Trees, Knowledge Studio provides the following predictive models (models that require a *dependent variable*):

- Linear Regression
- Logistic Regression
- Scorecards (based on Logistic Regression)
- Deep Learning (Multi-Level Neural Network)
- PLS (Partial Least Squares) Regression
- Regularization
- Constrained Regression
- Keras Deep Learning Model (implemented in Python)
- Generalized Linear Model (Sklearn GLM) (implemented in Python)
- Novelty & Outlier Detector Model (Sklearn Novelty & Outlier) (implemented in Python)
- XGBoost Model (Sklearn XGB) (implemented in Python)

In addition, the AutoML node (Automated Machine Learning) automatically builds a modeling workflow involving multiple types of models and selects the best one.

Logistic Regression

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Logistic regression is a model that allows you to predict an outcome for a discrete dependent variable by determining the odds that the observed values of the dependent variable can be predicted from the observed values of the independent variables.

Logistic regression calculates the probability of success over the probability of failure. The results of the analysis are in the form of an odds ratio.

Characteristics of logistic regression are:

- logistic regression does not assume linearity of relationship between the independent and dependent variables.
- it does not require normally distributed variables
- the dependent variable can be categorical {yes, no} or numeric {0, 1, 2}.
- the set of independent variables can be discrete, continuous or a mixture of both types.

A detailed description of a logistic regression model and the resulting statistics can be found in the topic Predictive Models of the Mathematical Formulations section.

Scorecard

Knowledge Studio Scorecard is a type of predictive model that associates points with a set of characteristics that are summed for a number of variables to produce a score. As a simple illustration:

Residential Status	Live with Parents Other	66
	Own Home	70
	Rent	39
Years at Address	0 to <1	51
	1 to <3	56
	3 to <5	58
	5 to <10	66
	>=10	71

An individual can be assigned a score by determining the points related to those characteristics he or she has, and by summing the values so obtained across the characteristics contained in the scorecard. So, as a simple example, if I rent (which is “worth” 39 points) and have been at my address for less than one year (which is “worth” 51 points) my score will be 90.

If I wish to make a decision based on this scorecard (for example to approve an individual for a loan), I may decide to reject all applicants below a specified minimum threshold score (referred to as a cut-off point), and above this level I may approve the applicant and grant different credit lines or modify the interest rates charged based on the score.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Knowledge Studio® is an enabling tool – it is designed to make it easier to produce scorecards. However, note that there may be many considerations which impact on the scorecard design and use, including laws, regulations and internal policies and procedures, as well as practical considerations about what information is or may not be available at the time of using a scorecard. Factors such as these should be taken into account in using this capability.

See the topic Inserting a Scorecard Model. A more detailed introduction to Scorecards that discusses them from the point of view of business users in financial organizations can be found in Knowledge Studio Tutorial, which can be accessed from the Welcome screen or from the Help menu. It is recommended that you use the Tutorial to learn a typical workflow for creating a scorecard.

Linear Regression

Linear regression, also called Multiple Linear Regression, is a method to model the linear relationship between a dependent variable and one or more independent variables. In its most simple form, the linear equation to solve is:

$$Y = b0 + b1X1 + b2X2 + ... + bkXk$$

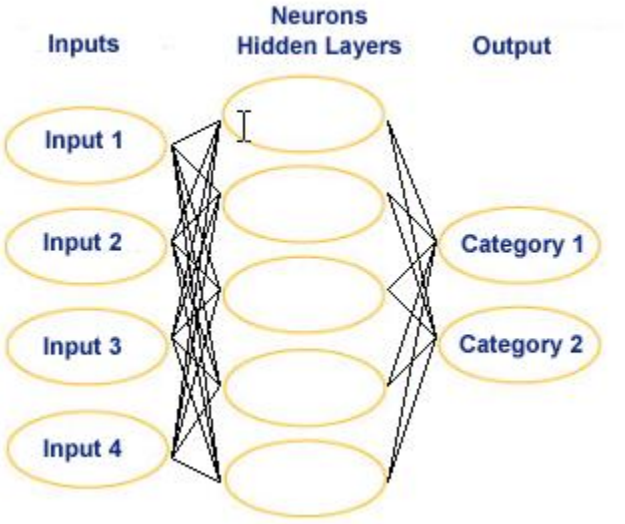
where Y is the dependent variable, b0 is the intercept coefficient and bi is the regression coefficients for each of the independent variables used in the equation.

A detailed description of a linear regression model and the resulting statistics can be found in the topic Predictive Models of the Mathematical Formulations section.

Deep Learning (Multi-Level Neural Network)

Neural Networks at the core of the Deep Learning models are based on a simplified model of the human brain. They use an internal construct called a “neuron”. Each neuron has a number of inputs that are combined to produce a single output. Outputs from a layer of neurons are fed to another layer called the hidden layer.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The effect of these neural pathways is that they produce pattern recognition.

Like the human brain, neural networks do not explain how they arrive at a particular decision - they simply recognize patterns. If you need to understand your model, build or rebuild it as a decision tree.

Results of Deep Learning Models (Multi-Layer Neural Networks)

The statistics for each layer can be seen if you select Currently Selected Sequence from the drop down list on the Predictive Model Results tab after training is complete.

Deep Learning models can use discrete or continuous variables as dependent variables. The results will be displayed differently depending on whether the DV is continuous or discrete.

The independent variable statistic show the parameters for the hidden layers. The hidden layer statistics are shown for each layer. The layers are labeled as H_N1_N2, where N1 is the layer index and N2 is the neuron index. For example, the following fragment of the model results report shows the information about the model parameters for the fifth neuron in the first layer (H_1_5):

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

H1_5		
Variable Name / Value		**Model Parameters**
age		1.856643
education	10th	-0.405961
	11th	-0.839169
	12th	0.119026
	1st-4th	0.527439
	5th-6th	0.179783
	7th-8th	-0.205366
	9th	0.139670
	Assoc-acdm	-1.018960
	Assoc-voc	0.723240
	Bachelors	-0.906371
	Doctorate	-0.898373
	Masters	-0.397652

For more details about the Multi-Level Neural Network, see the topic Deep Learning (Multi-Level Neural Network) in the **Mathematical Formulations** section.

Weight in Predictive Models

Starting from version 7, there is only one type of weight in datasets and models, which can be viewed as a generalized frequency weight (the weight values don't have to be integer). A weight can be set at the dataset level, but it can also be set and changed independently in the dependent models.

See the Weight topic for general information about weights and on setting weight at the dataset level.

Relationship between Weights in Datasets and Dependent Models

The weights in a dataset and its dependent models can be changed or reset independently. When you toggle weight on or off in the dataset Overview Report, this does not affect the existing models based on this dataset.

By default, any new model created on the basis of a weighted dataset inherits the weight from it. More precisely, when inserting a model for a weighted dataset, the weight field defined in the

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

dataset is automatically listed in the model's Attribute Editor as "weight" (in the Usage column). You can override this in the Predictive Model Wizard and redefine the field as an analysis variable by changing the usage from "weight" to "independent" if desired.

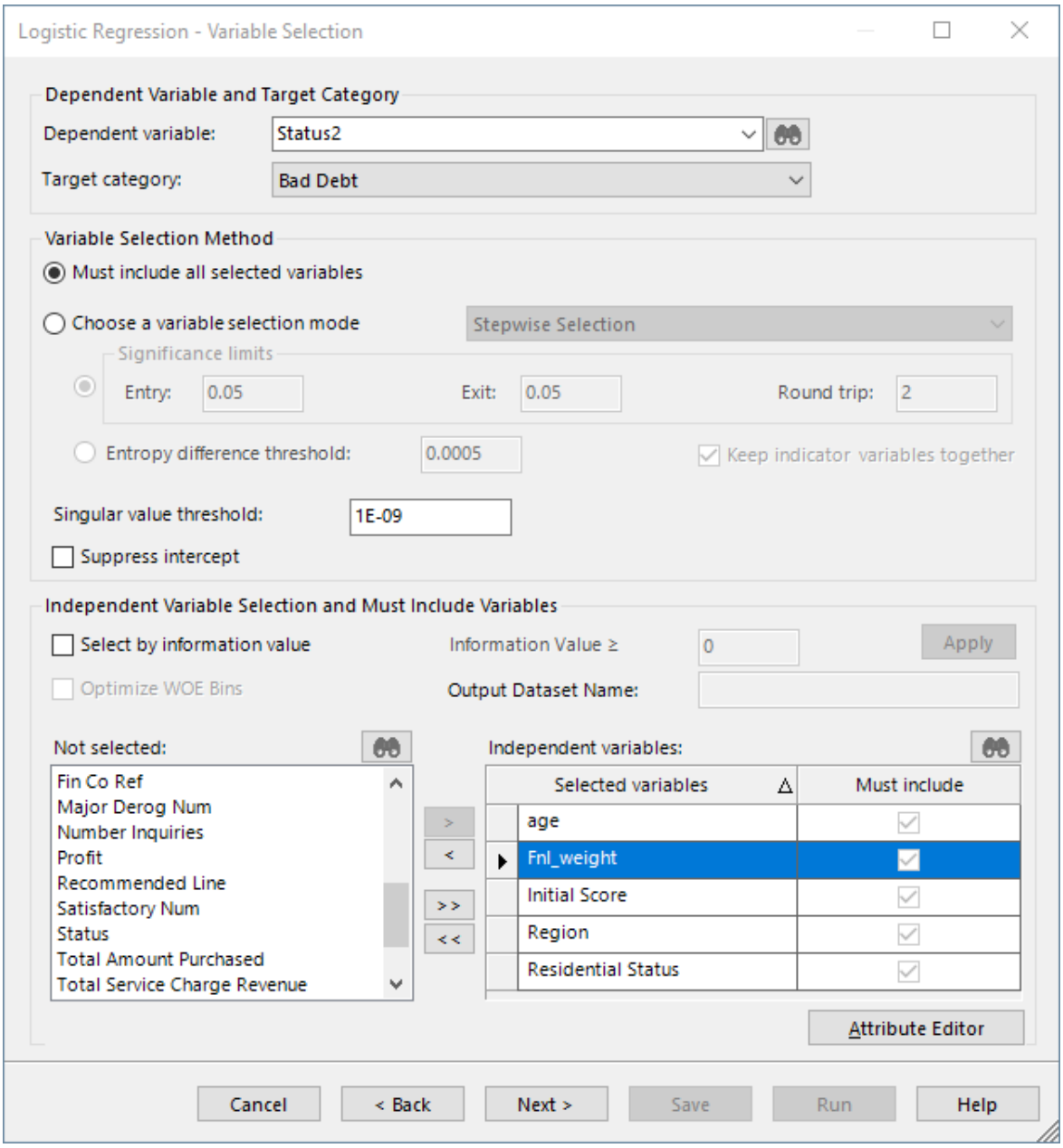
Weight in Predictive Models

It is assumed that the field you want to set as weight already exists in the parent dataset (although you don't have to set it as weight in Overview Report). See the Weight and Weight Helper topics for information on available tools for creating a weight field. If the parent dataset already has weight specified in Overview Report when you were inserting the model, the same weight is set in the model by default.

If you want to specify or change a weight for a Predictive Model, it must be done in the Predictive Model Wizard when inserting the model.

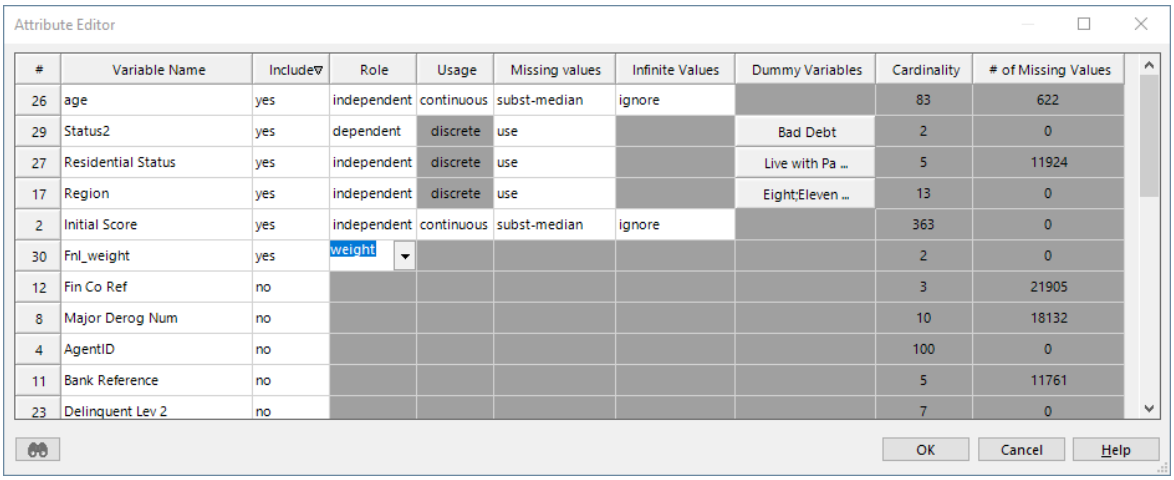
In the second page of the predictive model wizard, when choosing the analysis variables for your model, make sure to add your weight variable to the list of Independent Variables and click the **Attribute Editor** button.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



When the Attribute Editor window opens, click in the ***Role*** column for the variable you want to set as weight and select "**weight**" from the drop-down list. Note that the weight option is available only for nonnegative numeric variables. The weight variable does not have to be integer.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



After the model is inserted, the information about the weight is available in the Parameters and Attributes tab of the model.

Linear Regression Model

A Linear Regression node can be added to the workflow in one of the following ways:

- Drag a connection arrow from a dataset node to any area on the Workflow canvas background. From the context menu that appears when you release the mouse button, select **Model --> Linear Regression** as in the example screenshot below. A new **Linear Regression** process node will be created.

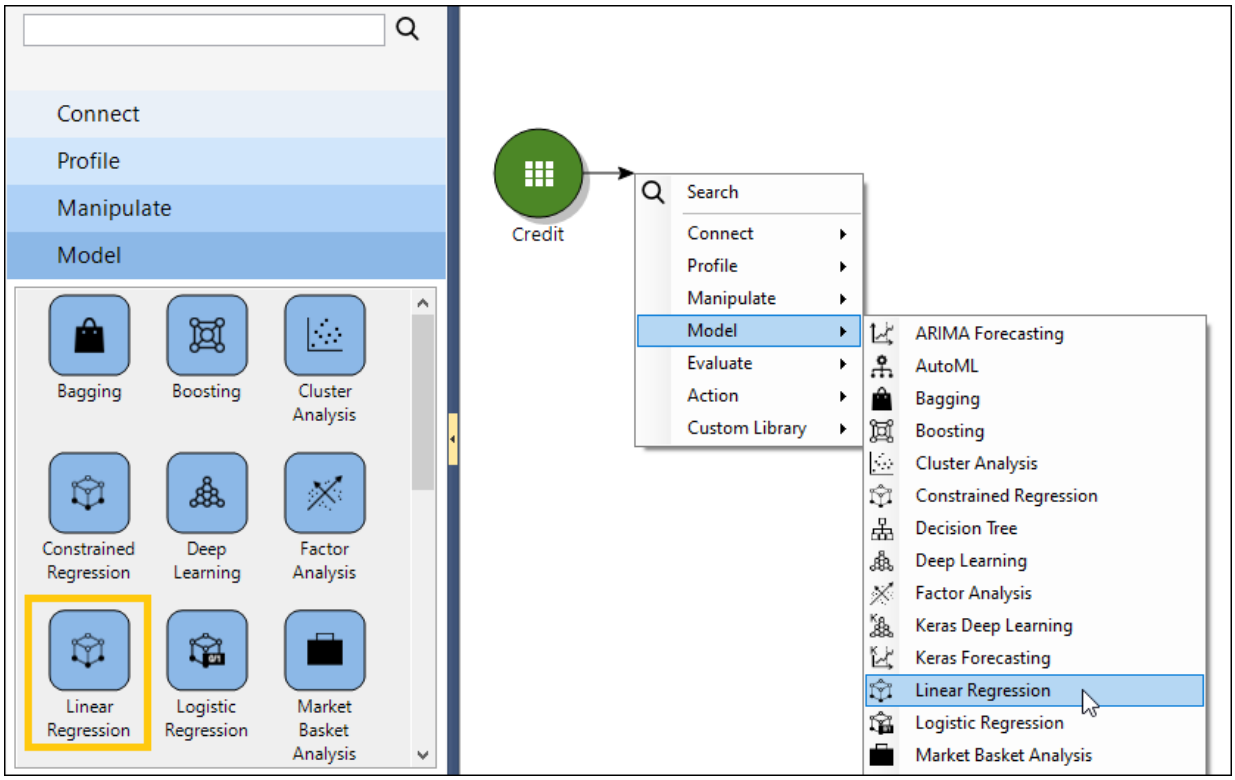
OR

- Right-click on the Workflow canvas and select **Model --> Linear Regression** from the context menu. The new Linear Regression node is gray until its inputs are defined. Connect a dataset to the tree node.

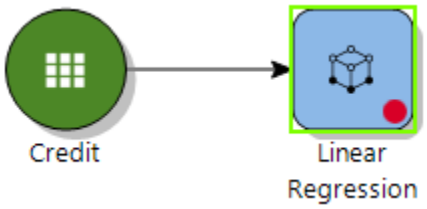
OR

- Open the **Model** palette and drag the **Linear Regression** node onto the workflow canvas. The **Linear Regression** node is gray while its inputs are undefined. Connect the input dataset to the **Linear Regression** node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

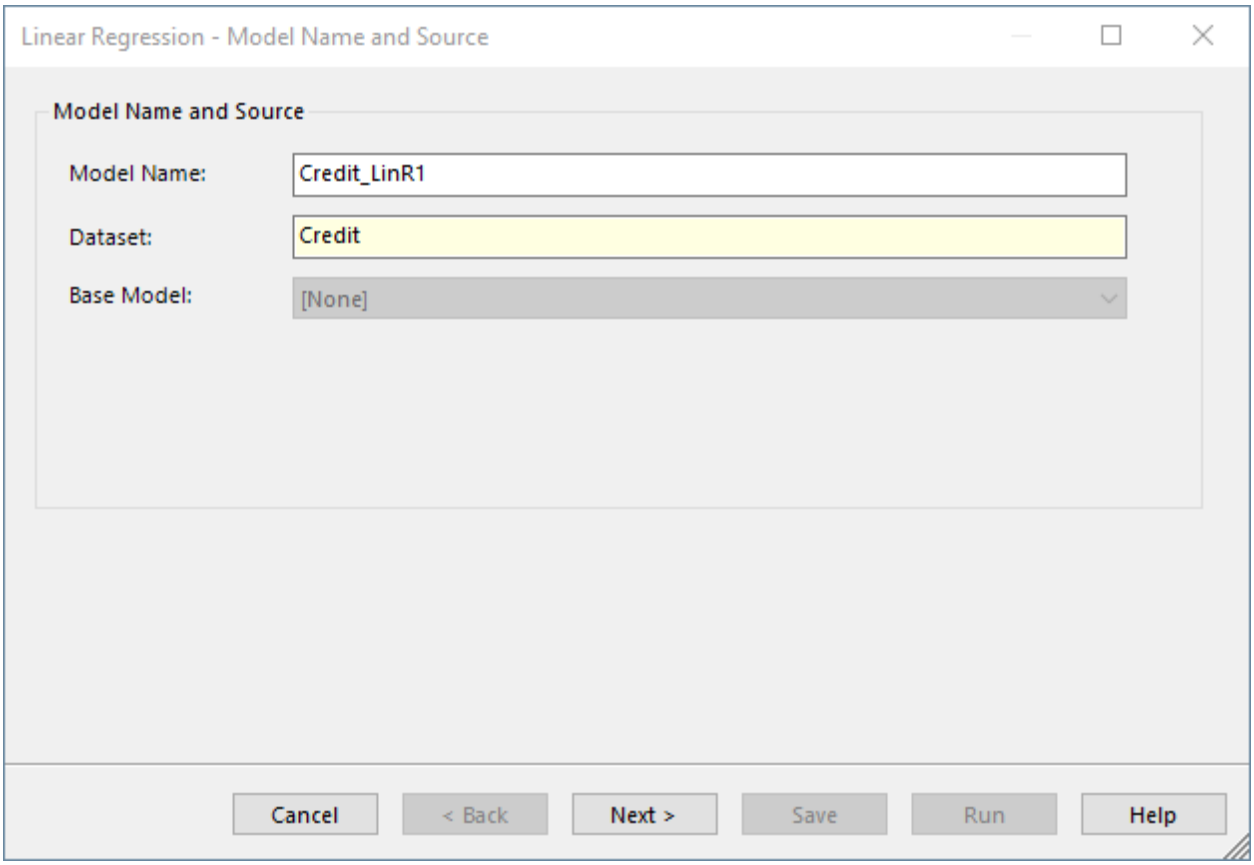


While it is not connected to other nodes, the regression model node is gray, as its inputs are undefined. Connect the input dataset to the regression node. A binary dependent variable should be present in your input dataset.



Double-click on the Linear Regression node added to the canvas, or right-click on it and select **Modify** from the context menu. Specify the model name and select the base model if applicable.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Model Nan The name of the model that will be produced when running the model node. The model object with this name will appear in the Project Explorer after running the node.

Dataset The name of the input dataset for the model.

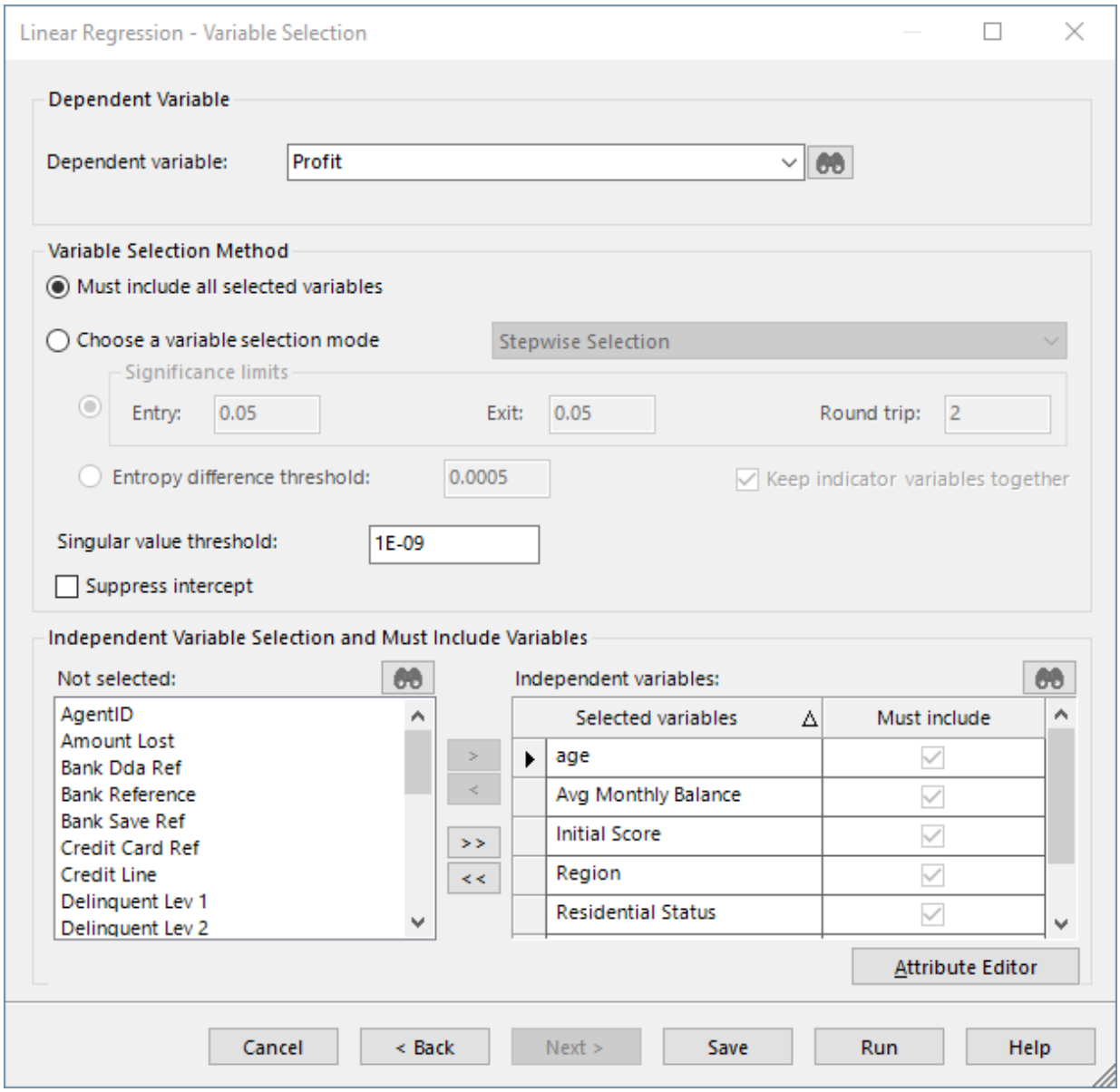
Base Mod The model definition to use as a template for the new predictive model. This requires connecting the model you intend to use as a template to the current Linear Regression node. In this case the current regression will have two inputs: a dataset and another model. All applicable training parameters and attributes of the template model will be copied and used for the new model. For a template, you can select any other linear regression model node in the workflow. This function is convenient for retraining models and inserting new models that inherit some or all of the model parameters of an existing model (at least the set of independent variables).

Click Next to select the variables and specify variable selection options.

Variable Selection

In the Variable Selection page, select the dependent variable (DV) and the independent variables (IVs) you want to use in your model. Only numeric variables can be selected as DV. The DV list will be empty if you don't have any numeric variable in the input dataset.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Dependent Variable	The variable of interest for the analysis. The drop down list contains the available variables. For linear regression, only numeric variables can be selected as DV. Namely, variables of type Number or Integer.
Target Category	Not applicable to models with continuous dependent variable.

Note: The DV can also be changed in the Predictive Model Attribute Editor. Click the Attribute Editor button to open the attribute editor and set the Role to "dependent" for the desired variable.

Must include all selected variables

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If this method is checked, then all the variables selected in the previous wizard step will be used when training the model, regardless of their significance.

Variable Selection Method

Stepwise	Include variables by considering the Entry and Exit Significance Levels.
Forward Selection	Starts with no variables and then includes variables based on the rank order of variables from the best to the worst. The rank order is determined by a log-likelihood ratio test.
Backward Selection	Starts with all the variables and then excludes variables based on the rank order of variables from the worst to the best. The rank order is determined by a log-likelihood ratio test.
R-Squared Selection	Check through all possible combinations of coefficients and produce a ranked list with r-square increasing as various combinations go in.
Variable Sequence My Choice	Include variables based on the user's custom selection. The variables can be chosen on the basis of their significance (the P-value computed for each variable). See Variable Sequence Selection for details.

Significance Limits

Entry	The entry significance level determines when a variable is qualified to be included in the model. This parameter is used for the **Stepwise** and **Forward** selection methods.
Exit	The exit significance level determines when a variable is disqualified and excluded from the model. This parameter is used for the **Stepwise** and **Backward** selection methods.
Round Trip	The number of entry-exit-entry-exit sequences before a variable is disqualified and excluded from the model.

Entropy Difference Threshold

"Entropy difference threshold" defines a stopping criterion for model training with the *Stepwise*, *Backward*, and *Forward* variable selection methods. If this option is selected, the model training is stopped when the change in Entropy Explained in consecutive models in the iterative process is less than the user-defined level. The default threshold is 0.0005.

This option is not applicable when Significance Limits are specified or when the variable selection method is "*R-Squared Selection*" or "*Variable Sequence Of My Choice*".

After the training is finished, you can check the Entropy Explained values in the Results tab of the model in the sequence.

Keep indicator variables together

When this option is selected, the categories of each discrete variable are treated as a group. When a discrete variable is included or excluded from the model all the categories within the variable are included or excluded together as a group. For example, ***marital-status*** contains seven categories: **Single**, **Married**, etc. If

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

marital-status is excluded, all the categories are excluded from the model. When this option is unchecked, each category is evaluated separately and will be included or excluded individually. This option is selected by default.

This option is not applicable when "*Variable Sequence Of My Choice*" is selected.

Singular Value Threshold

Used to adjust the collinearity detection sensitivity. The threshold determines when any two independent variables are considered to be collinear and one of them must be ignored (removed from the model) to avoid the risk of biasing the model.

If there is an exact linear dependency between any variables, the corresponding singular value is equal to zero. A small singular value (eigenvalue) indicates collinearity issues, so when small singular values occur, this threshold determines how small the singular values may be without causing any problems to the least squares estimation.

The default value for this threshold is 1.0E-09.

Independent Variable Selection and Must-Include Variables

Variables on the left are not included in the model. Variables listed on the right will be included. You can move the variables to and from the Independent Variables list by highlighting them and using the arrow buttons. Double-clicking the individual variables will also move them from one list to the other.

To select more than one variable in the lists, use mouse dragging, or mouse clicks while holding down the SHIFT key or the CTRL key. If there are a lot of variables (more than a hundred), it is convenient to use Field Finder (the binoculars button ).

If you chose a particular variable selection method, you have an option of selecting the variables that must be forced into the model regardless of the selection method. The **Must include** column is only enabled if you have chosen a particular Variable Selection Method. During the variable selection process, the algorithm will exclude independent variables from the model if they do not affect the model. However, you may want to make sure that specific variables are kept in the model regardless of the algorithm. The variables that are in the **Must include** list will be forced into the model regardless of the method selected.

Attribute Editor

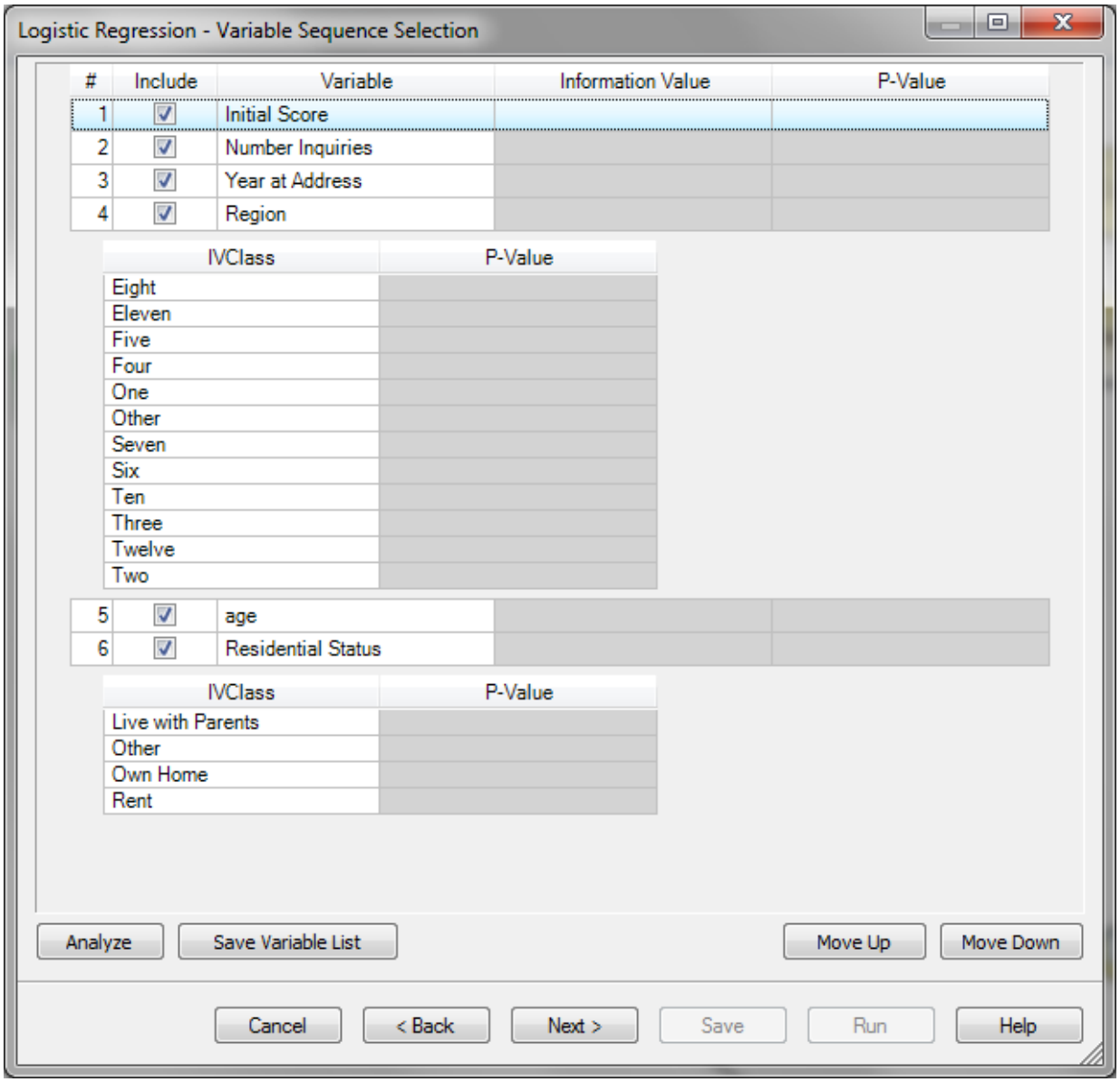
The **Attribute Editor** button brings up the predictive model attribute editor. The attribute editor allows you to modify variable attributes such as missing value substitution options, etc. For details, see the topic Model Attribute Editor.

Custom Sequencing

Custom Sequencing is applicable to Regression models and Deep Learning models.

Selecting **Variable Sequence Of My Choice** as the variable selection mode allows you to define a custom order of including variables in the model. The variables can be chosen on the basis of their significance (the *p*-value computed for each variable) and Information Value. For details on these and otehr statistics, see Mathematical Formulations: Decision Tree.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



More precisely, the model under 'Variable Sequence Of My Choice' starts with no variables and then includes variables based on their order in the list of variables in this page of the wizard. The buttons "**Move Up**" and "**Move Down**" allow you to change the order of inclusion of variables in the sequence. It starts with the topmost variable in the list, then adds subsequent variables one by one until it adds all of them, and finally chooses the best sequence as the "Currently Selected Sequence".

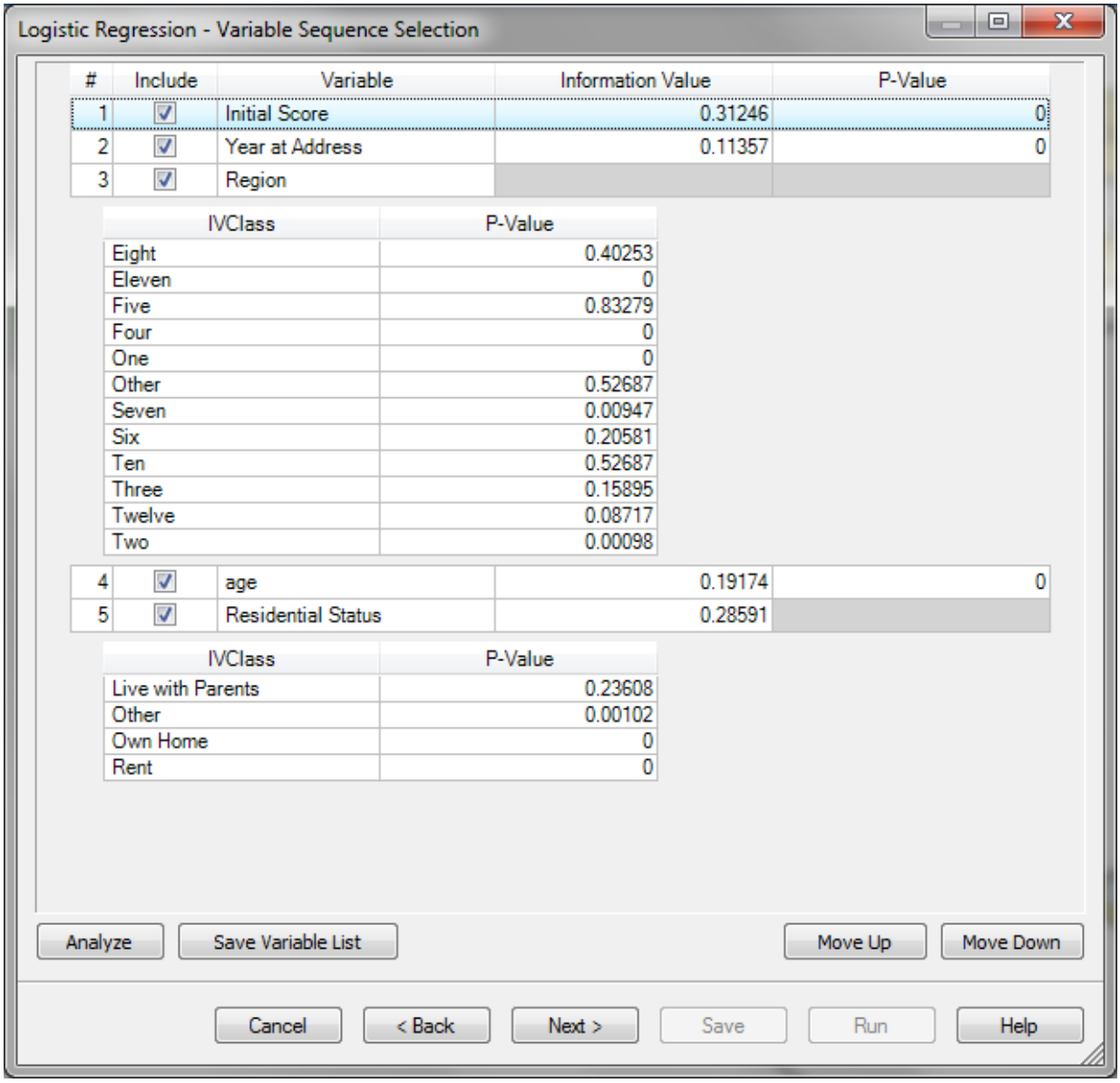
The details in the statistics for each sequence can be found in the Results tab of the model when it is trained (open the drop-down list "Output To View" at the top of the Results tab).

The significance for each variable is computed by building a single-independent-variable regression model with this variable and then computing the P-value.

Click the **Analyze** to compute the *p*-value and Information Value in this page.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Based on the computed parameters, decide which variables to include and move up or down.



Linear Regression model training results are illustrated here using an example model built with Stepwise selection method. It contains the output of the final model as well as all intermediate models built in the stepwise variable selection process.

The default view displayed after training is a short summary report for the final model (Model Overview). It shows the list of the input variables, and the value of Variance Explained.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



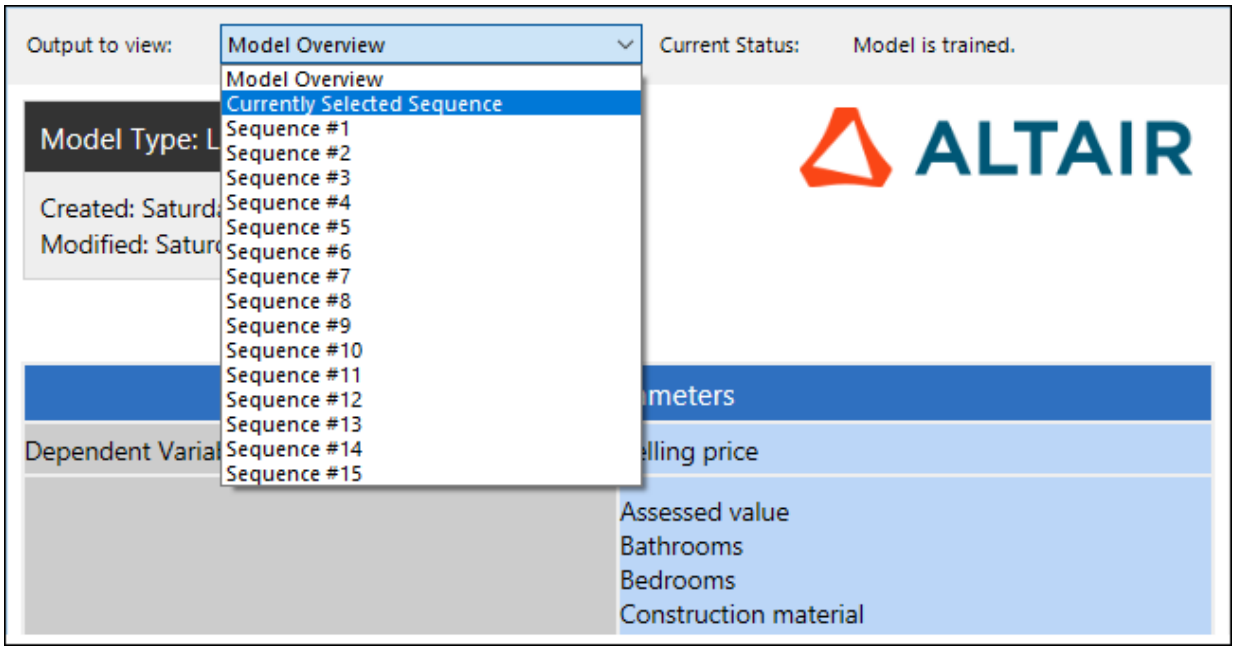
Note: The number of records correctly predicted is more relevant in the Model Validation wizard, where you can specify the *Valid hit range* - the maximum distance between the predicted and actual value for the prediction to be considered correct. For example, if Valid hit range is set at 1000 in the Model Validation wizard, the Selling price prediction will be considered correct if the predicted value is within $1000 of the actual value.

The list of independent variables in the Model Parameters section of the Results view includes all input variables selected in the second page of the training wizard. Not all of these variables are included in the final model. Some are discarded during the training process with Stepwise selection technique. Even without variable selection techniques some variables may be discarded at the end because of collinearity or other reasons, in which case a solution advisory is provided where these

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

reasons are explained. The variables ultimately selected into the final model are listed in the Sequence view of the final model.

In the same view, select Currently Selected Sequence from the **Output to view** drop-down list. The detailed Sequence Statistics report includes detailed output: model performance statistics, model parameters, and model equation, which provides the formula for predicting the dependent variable values. In this example, 15 models have been trained in the iterative process of model training.



Choose the Currently Selected Sequence, which is the final model selected in the training process. The top of the view indicates the sequence number that corresponds to it.

The Analysis of Variance section shows performance statistics for the whole model. The Independent Variable Statistics section shows model parameters - coefficients in the model equation - and various statistics for each parameter. From the Independent Variable Statistics section of the Currently Selected Sequence, you can see that one of the independent variables - Frontage (ft) - is not included in the final model as it was deemed not significant enough and discarded during the variable selection process.

Most of the model statistics, such as P-value, AIC, and BIC have the same meaning as in Logistic Regression Results.

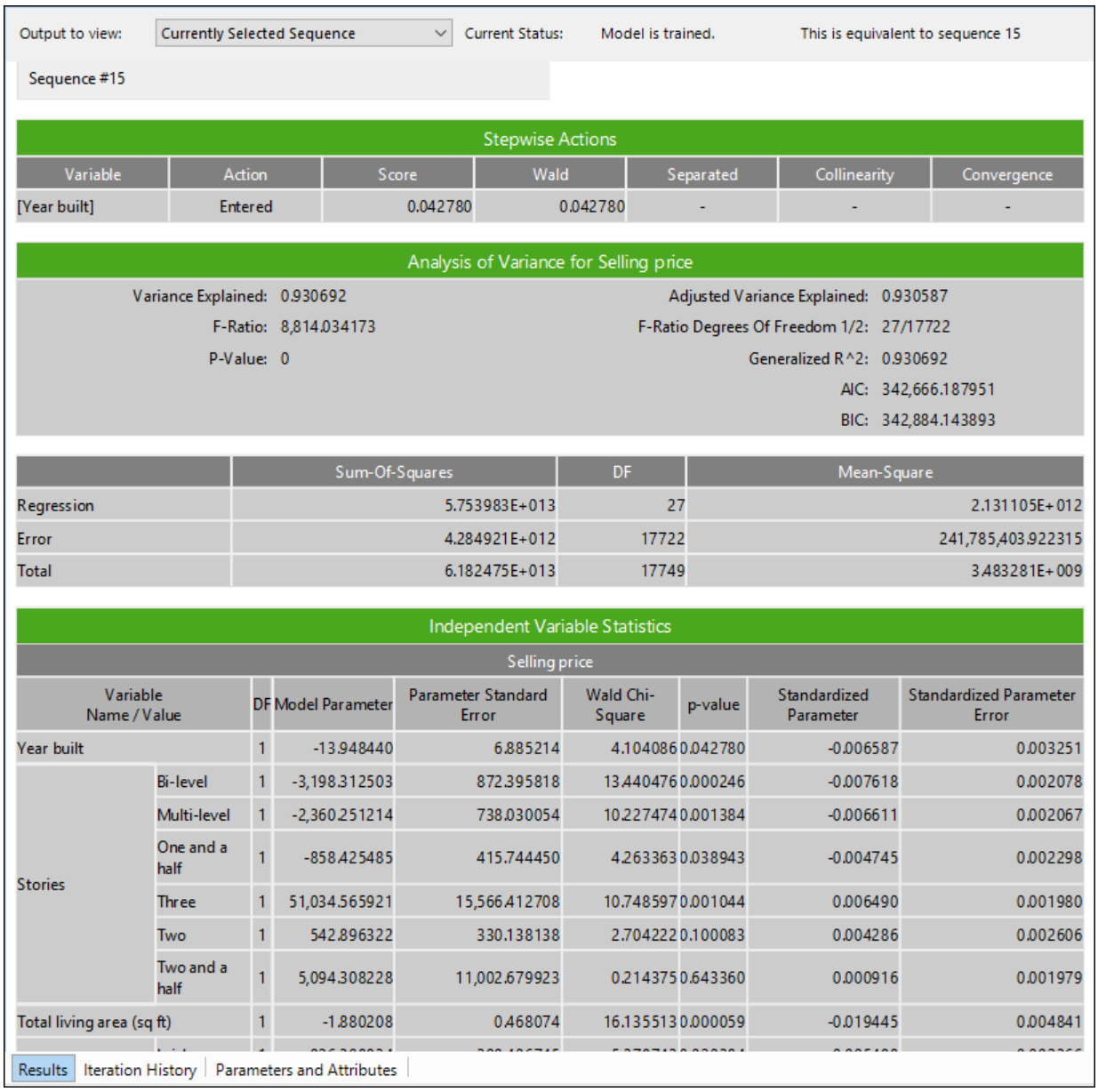
First, examine the statistics in the Analysis of Variance section at the top.

Variance Explained is a measure that shows how much variation of the dependent variable was explained by the model. The higher its value, the better. In the final model in our example, it is 0.930692 - the highest among all 15 models. It is an important, but not sufficient parameter characterizing the model performance. Evaluate it in combination with the values of Sum of Squares, AIC (Akaike Information Criterion), BIC (Bayesian Information Criterion), Chi-Square, P-Value, independent variable statistics (model parameters, significance, standardized parameter), and Variance Inflation Factors.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

AIC is an important measure of the goodness of fit which penalizes excessive model complexity at the same time. The lower the AIC value, the better. The best model is selected based on this criterion. If you compare Variance Explained and AIC in all 15 sequences, you will notice that the best model has the highest value of Variance Explained (0.930692) and the lowest AIC value (342,666.187951).

BIC is a model selection criterion similar to AIC, but it imposes higher penalty for the number of model parameters. In our example, Sequence #15 is the best model according to both AIC and BIC criteria, since it has the smallest AIC and BIC of all models (342,666.187951 and 342,884.143893, respectively).



Output to view: Currently Selected Sequence | Current Status: Model is trained. | This is equivalent to sequence 15

Sequence #15

Stepwise Actions

Variable	Action	Score	Wald	Separated	Collinearity	Convergence
[Year built]	Entered	0.042780	0.042780	-	-	-

Analysis of Variance for Selling price

Variance Explained:	0.930692	Adjusted Variance Explained:	0.930587
F-Ratio:	8,814.034173	F-Ratio Degrees Of Freedom 1/2:	27/17722
P-Value:	0	Generalized R^2:	0.930692
		AIC:	342,666.187951
		BIC:	342,884.143893

	Sum-Of-Squares	DF	Mean-Square
Regression	5.753983E+013	27	2.131105E+012
Error	4.284921E+012	17722	241,785,403.922315
Total	6.182475E+013	17749	3.483281E+009

Independent Variable Statistics

Selling price

Variable Name / Value		DF	Model Parameter	Parameter Standard Error	Wald Chi-Square	p-value	Standardized Parameter	Standardized Parameter Error
Year built		1	-13.948440	6.885214	4.104086	0.042780	-0.006587	0.003251
Stories	Bi-level	1	-3,198.312503	872.395818	13.440476	0.000246	-0.007618	0.002078
	Multi-level	1	-2,360.251214	738.030054	10.227474	0.001384	-0.006611	0.002067
	One and a half	1	-858.425485	415.744450	4.263363	0.038943	-0.004745	0.002298
	Three	1	51,034.565921	15,566.412708	10.748597	0.001044	0.006490	0.001980
	Two	1	542.896322	330.138138	2.704222	0.100083	0.004286	0.002606
	Two and a half	1	5,094.308228	11,002.679923	0.214375	0.643360	0.000916	0.001979
Total living area (sq ft)		1	-1.880208	0.468074	16.135513	0.000059	-0.019445	0.004841

Results | Iteration History | Parameters and Attributes

F-Ratio shows whether the model is of significant value in explaining the dependent variable. More precisely, it shows how much higher the amount of variation explained by the model is than the variation explained by the mean of the DV. The higher the value of F-Ratio, the better. In our final

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

model, F-Ratio = 8814.034173, which is not the highest among all models, as sequences 4-17 have higher F-Ratio, but this is not the main criterion of the model quality.

P-value is the probability of obtaining current model results using a null (random) model. More formally, it is the probability of rejecting the null hypothesis when it is actually valid. The smaller the P-value, the better. In our example, it is equal to 0 with precision of up to 6 decimal points.

The number of Degrees of Freedom is the number of independent parameters in the model. Generally, it is greater than the number of input variables because of the normalization of discrete variables: each String variable with N values is transformed into a set of N-1 indicator variables with values 0 and 1. In our example there are 16 variables initially included in the model, while the number of degrees of freedom (DF) is 27 because of the normalization of three String variables: Stories, Construction material, and Garage type.

Sum of Squares Regression (SSR) shows how much the independent variables contribute to the explanation of the dependent variable variance. It is equal to the sum of the squared differences between the predicted and mean values of the DV over all records. The higher the SSR value, the better. The SSR value in our final model is the highest among all 15 models.

Sum of Squared Errors (SSE) shows how much variance in the dependent variable remains unexplained by the independent variables. It is equal to the sum of the squared residuals (prediction errors) over all records. The smaller the SSE value, the better. The SSR value in our final model is the smallest among all 15 models.

Now examine the Independent Variable Statistics section:

To determine whether there is a significant relationship between the dependent variable (DV) and an independent variable (IV) in the model, look at the value in the Significance column. The value of Significance expresses the likelihood of the null hypothesis that the model parameter is zero, i.e., that the corresponding variable is statistically insignificant for the DV. Thus, the closer the Significance value to zero, the more significant the variable is to the DV. For example, the variables Assessed value and Lot size have Significance = 0 with precision of 6 digits after the decimal point. This means they are highly significant for the DV Selling price, which is also intuitively clear. Most other variables in our example are fairly significant for Selling price, since their Significance values are well below 0.05 (5%), which is a cutoff value for considering the variable not significant enough. However, Year built has Significance = 0.04278, which is close to 0.05, and there are about 6 indicator variables (components produced by the String variable normalization discussed above) with Significance values much higher than 0.05, and therefore statistically insignificant to the DV when taken separately. All String variables in the final model sequence #15, however, include components statistically significant for the DV.

Standardized Parameters are Model Parameters rescaled to a common basis by dividing each by the standard deviation of the model parameters. From the Standardized Parameter values it can be easily seen which variable contributes the most to the model equation. In our example, as one may expect, Assessed value contributes the most, as it has the largest Standardized Parameter value (0.939545). Lot size is the second largest contributor, with Standardized Parameter = 0.029761.

Scroll down to examine Variance Inflation Factors and the model equation. See the next screenshot below. The Variance Inflation Factors help to estimate the degree of multicollinearity between the independent variables. The square root of the variance inflation factor for a given variable tells you

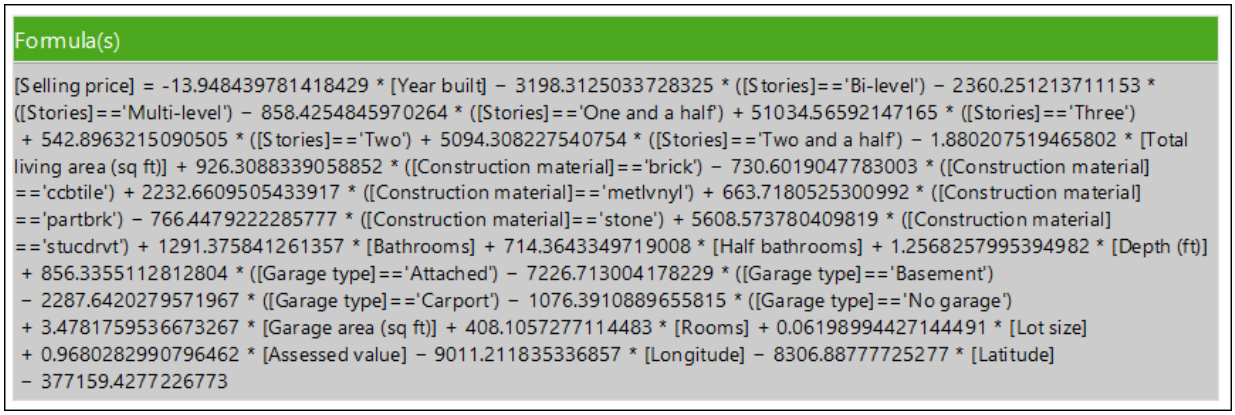
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

how much larger its standard error is, compared with what it would be if that variable were uncorrelated with the other predictor variables in the model. Usually the degree of multicollinearity is considered to be high if VIF is greater than 5. In our case only Total living area (sq ft) has VIF slightly exceeding 5 (5.992). Assessed value has the second highest VIF (4.398).

Variance Inflation Factors	
Variable	Value
[Year built]	2.703
([Stories]=='Bi-level')	1.104
([Stories]=='Multi-level')	1.093
([Stories]=='One and a half')	1.350
([Stories]=='Three')	1.002
([Stories]=='Two')	1.737
([Stories]=='Two and a half')	1.001
[Total living area (sq ft)]	5.992
([Construction material]=='brick')	1.432
([Construction material]=='ccbtile')	1.013
([Construction material]=='metlvnyl')	1.169
([Construction material]=='partbrk')	1.859
([Construction material]=='stone')	1.011
([Construction material]=='stucdrvt')	1.020
[Bathrooms]	2.447
[Half bathrooms]	1.785
[Depth (ft)]	1.477
([Garage type]=='Attached')	2.113
([Garage type]=='Basement')	1.018
([Garage type]=='Carport')	1.028
([Garage type]=='No garage')	2.252
[Garage area (sq ft)]	2.591
[Rooms]	2.919
[Lot size]	1.713
[Assessed value]	4.398
[Longitude]	1.519
[Latitude]	1.200

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The model formula is displayed at the bottom of the Sequence Statistics page. It expresses the dependent variable Selling price as a linear combination of the independent variables plus a constant term. The numeric coefficients of the independent variables in the formula are the values in the Model Parameters column of the Independent Variable Statistics table above. The last (constant) term is the intercept (-377159.4275095398). At the model deployment stage when the model is scored, the values of the required variables in the input dataset are substituted for the model parameters in the formula, and the predicted value of Selling price is calculated.



More details about the linear regression algorithm and the statistics displayed in this view can be found in the Regression section of Mathematical Formulations.

Generalized Linear Model

Overview

The **Sklearn GLM** node in the **Model** palette of Knowledge Studio allows you to build regression models with a dependent variable that does not necessarily follow the normal distribution. The node wizard is similar to that of the Linear Regression node, but it works through integration with Python libraries. It uses regressors of the linear model methods of **`scikit-learn`** - a popular Python library for machine learning. For details, see the scikit-learn documentation for Generalized Linear Regression.

The node wizard provides a convenient graphical interface to specify the options and parameters for linear model functions in the `scikit-learn (sklearn)` library. The wizard automatically translates the inputs into Python code, which can be previewed before running the node. The code then runs in a container with the necessary Python packages and their dependencies. The input dataset is converted from the Altair KDD format to pandas dataframes in the process. The resulting dataframe is converted back to produce the output Altair dataset.

After training, the model can be validated on test data using the Sklearn GLM Validation node or used for scoring new data with the Sklearn GLM Scoring node.

Building an Sklearn GLM Model

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

An **Sklearn GLM** model requires a single dataset as input. It can optionally take another predictive model as the second input (*base model*), in which case it inherits the dependent variable and independent variable selection from the base model.

An **Sklearn GLM** node can be added to the workflow in one of the following ways:

- Drag a connection arrow from the training dataset node to any area on the Workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Model --> Sklearn GLM** as in the example screenshot below. A new **Sklearn GLM** node will be created, and the wizard will open.

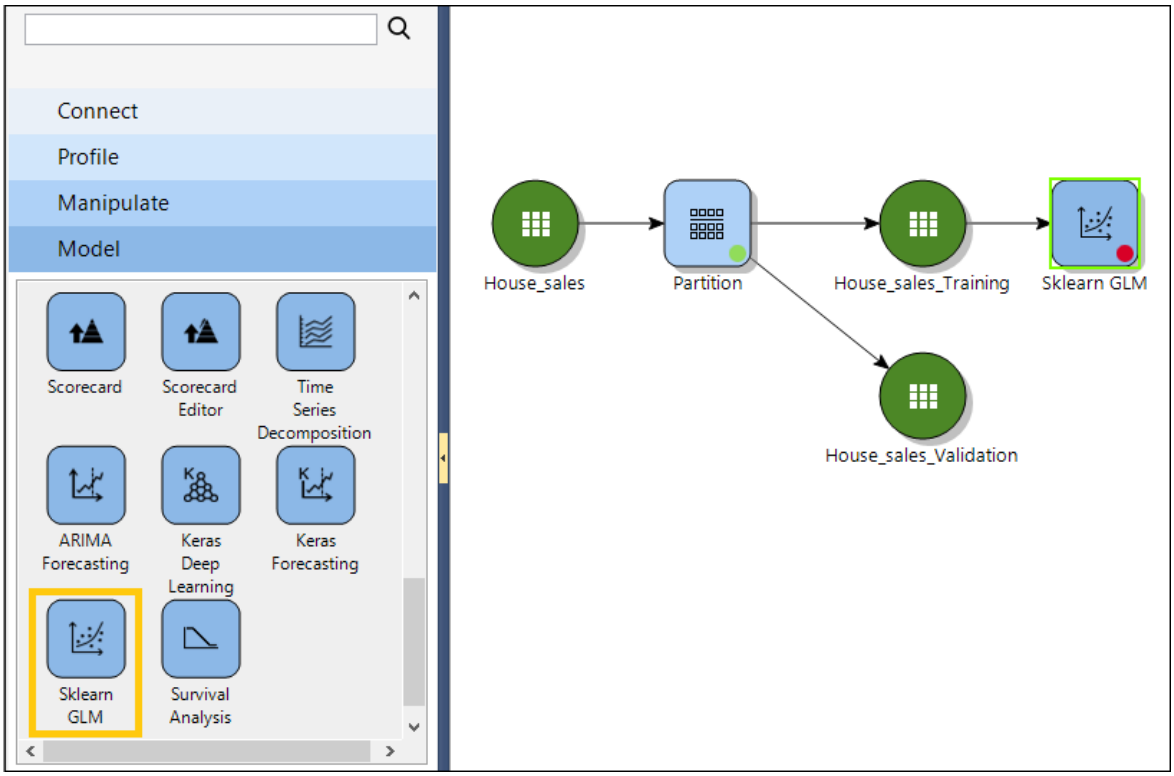
OR

- Right-click on the Workflow canvas and select **Model --> Sklearn GLM** from the context menu. Connect the training dataset. Optionally, connect another predictive model node as a base model to use as a template.

OR

- Open the **Model** palette and drag the **Sklearn GLM** node onto the workflow canvas. Connect the training dataset. Optionally, connect another predictive model node as a base model to use as a template.

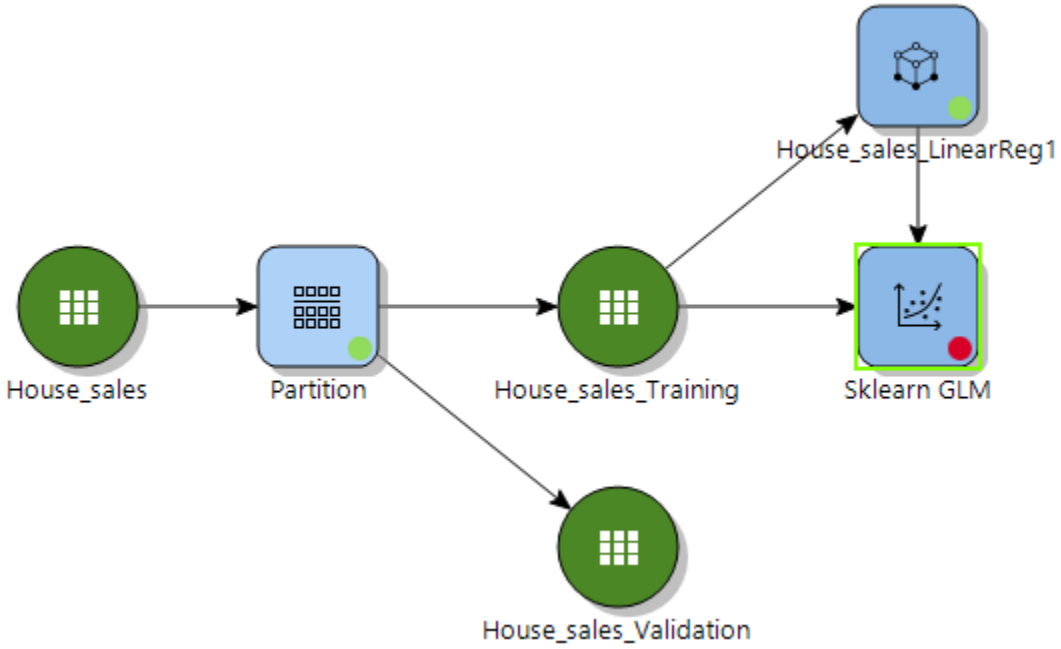
Example 1:



Example 2 (with base model)

In this example, an existing Linear Regression model is connected to the new Sklearn GLM model as a base model.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



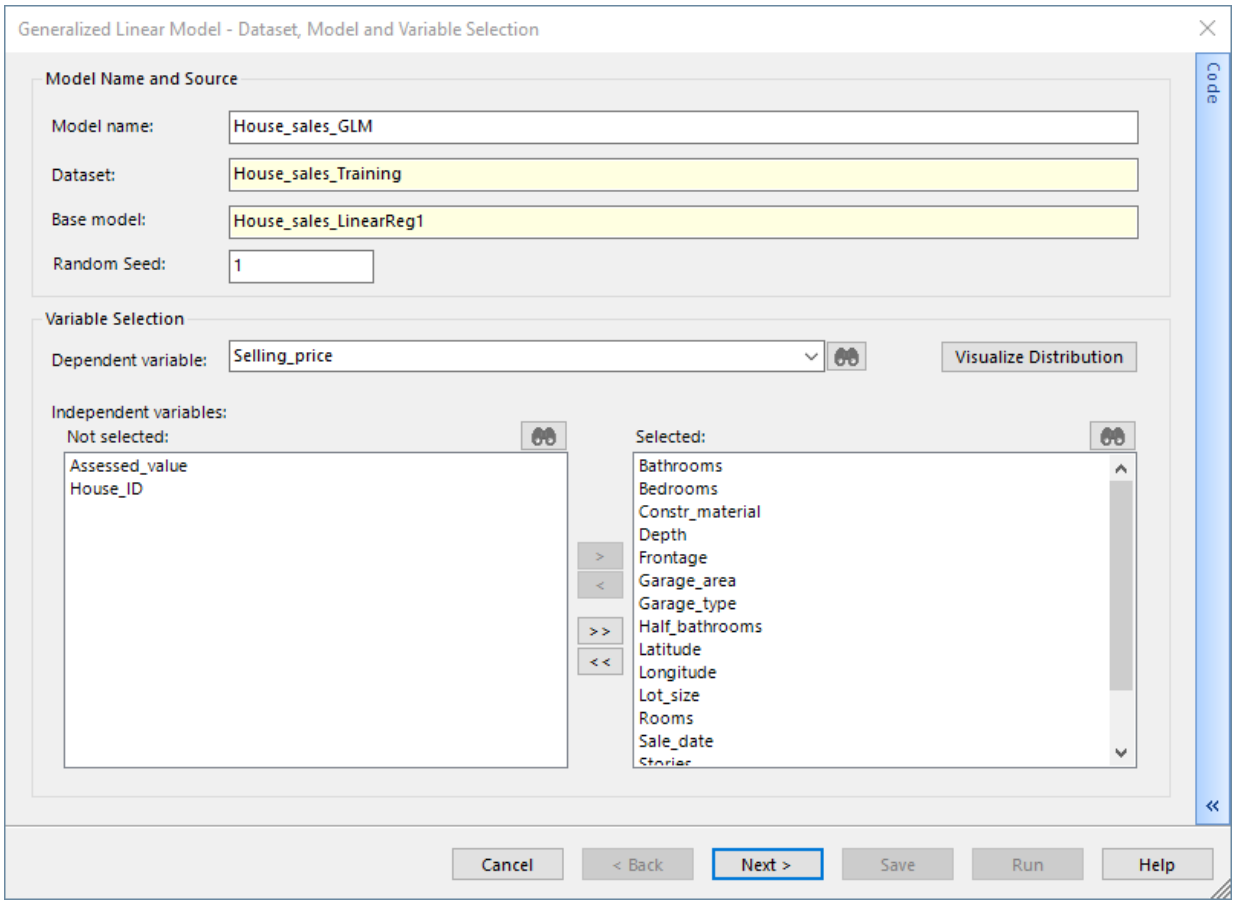
Double-click the Sklearn GLM node or right-click on it and select **Modify** to open the wizard.

In the first page of the wizard:

- Specify the model name
- Change the Random Seed value if required for repeatability.
- Select the dependent variable. Only numeric variables are available for selection.
- Select the independent variables.
    - To select or deselect variables, use the arrow buttons or double-click on the variable names. Advanced field finding and multi-selection capabilities are also available via the Field Finder dialog, which can be opened by clicking the binoculars button.

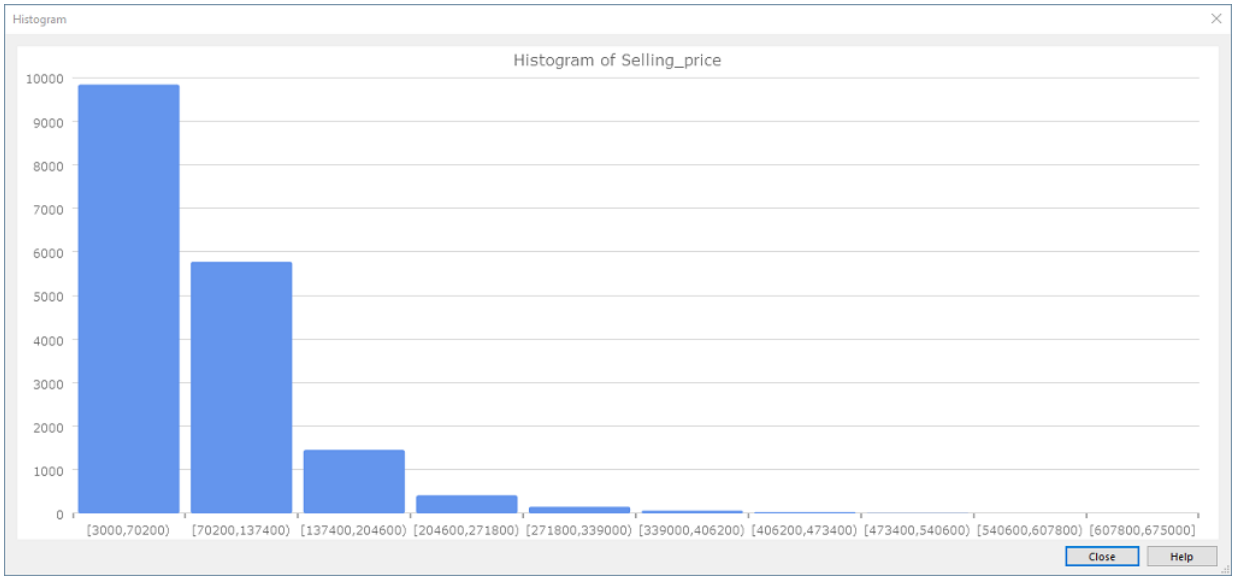
  Note: The dependent variable and independent variables are automatically inherited from the Base model if it is connected.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- Click the button **Visualize Distribution** to see the histogram representing the distribution of the dependent variable.

.



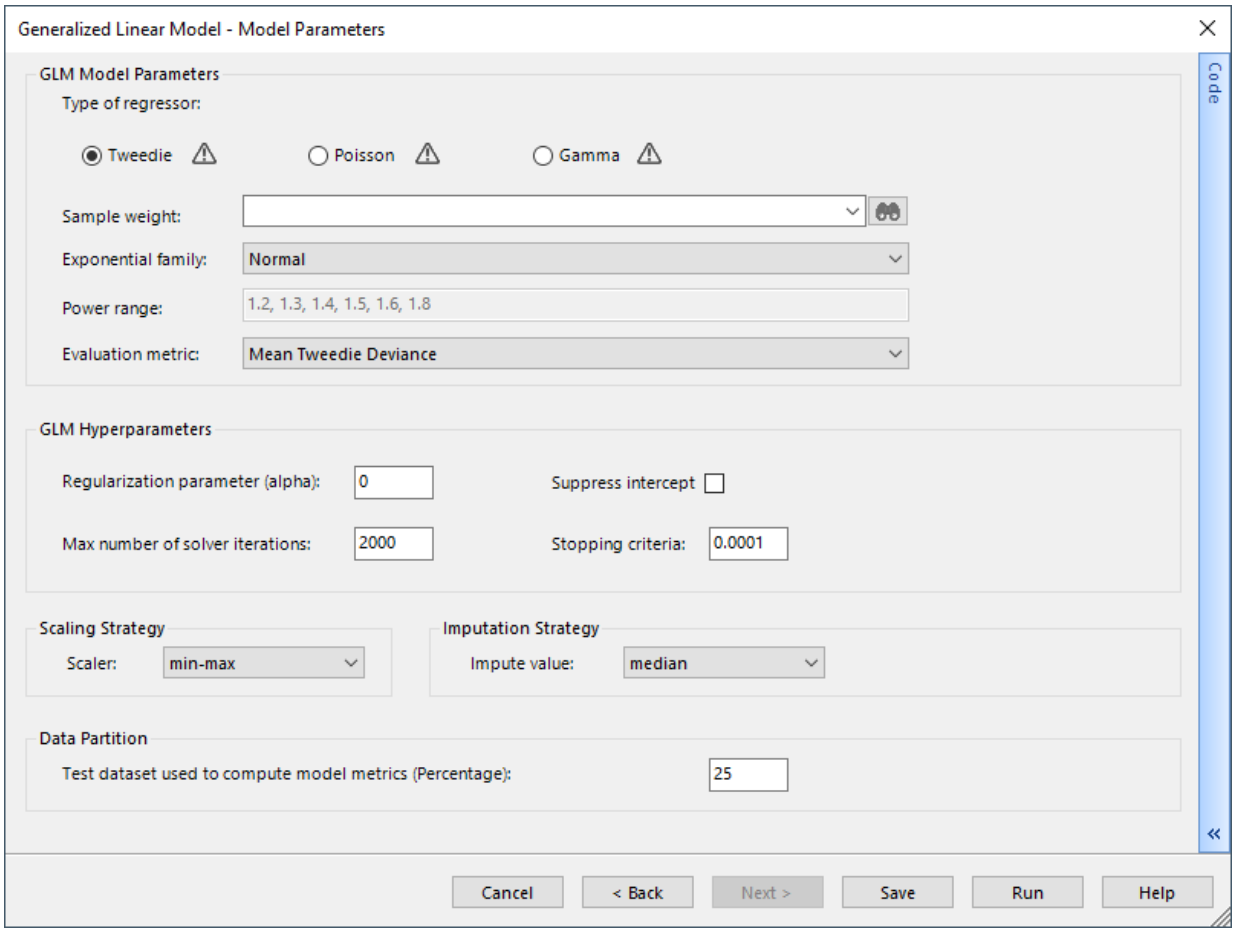
Click **Next**.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

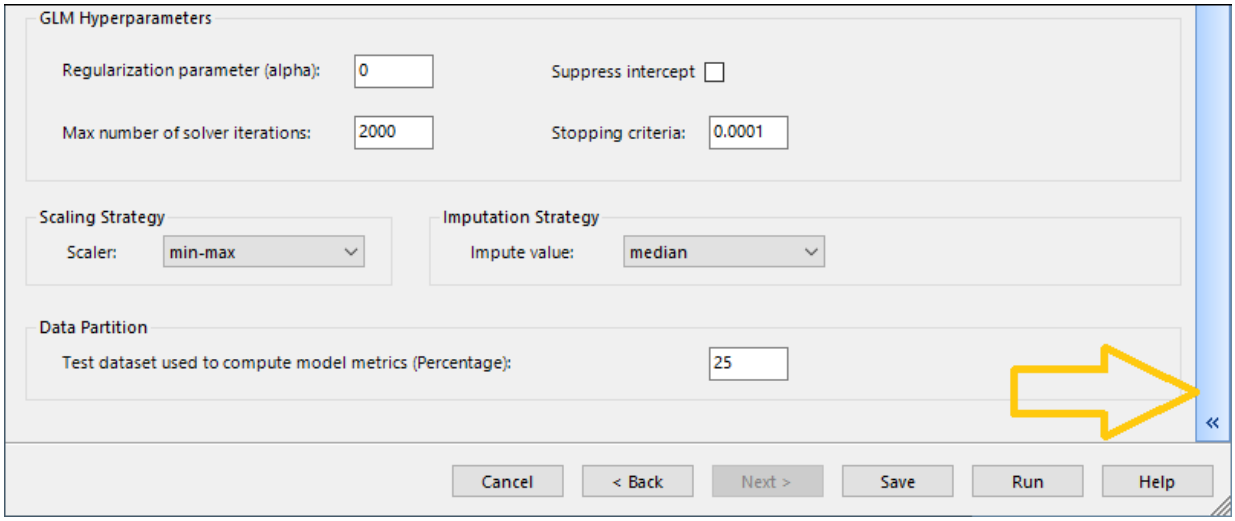
In the Model Parameters page, specify the following GLM model parameters as needed (See the scikit-learn documentation for Generalized Linear Regression for the description of regressor types, evaluation metrics, and other parameters.)

- **Type of regressor**:
  - Tweedie - most commonly used in insurance for modeling Pure Premium. This is the default regressor.
  - Poisson - most commonly used for count data
  - Gamma - used to model positive-valued dependent variables (continuous, positive, right-skewed, and where the variance is near-constant on the log-scale)
- Select **Sample weight** if applicable.
- If the Tweedie regressor is selected:
  - Select the **Exponential family**: *Normal, Poisson, Compound Poisson Gamma, Gamma,* or *Inverse Gaussian*. The default is *Normal*.
    - For *Compound Poisson Gamma*, specify the **Power range** as a comma-separated list. The default is "1.2, 1.3, 1.4, 1.5, 1.6, 1.8".
  - Choose the **Evaluation metric**: Mean Tweedie Deviance, MAE (Mean Average Error), or MSE (Mean Square Error).
- Specify **GLM Hyperparameters**:
  - Regularization parameter (alpha). The default is 0.
  - Select the option to suppress intercept, if necessary.
  - Specify the maximum number of solver iterations. The default is 2000.
  - Specify the evaluation metric threshold for the stopping criterion. The default is 0.0001.
- Specify the **Scaling strategy**:
  - Select the scaler option: *min-max* or *standard*. The default is *min-max*.
  - Select the imputation strategy: impute *median*, *mode*, or *mean* value. The default is *median*.
- For **Data Partition**, specify the percentage of the training dataset to be allocated to compute model metrics. The default is 25% allocated for testing.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

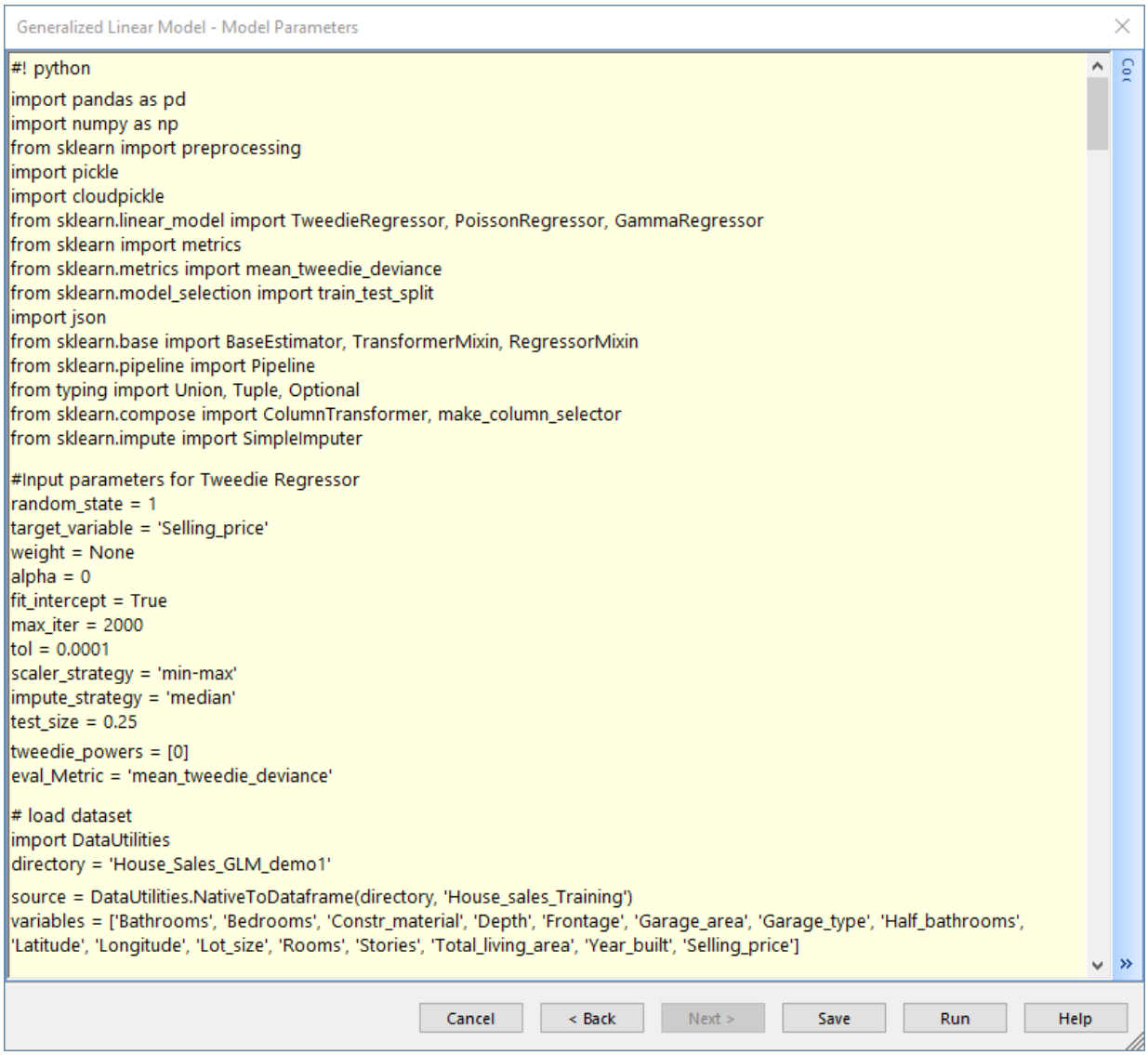


To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog.



This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

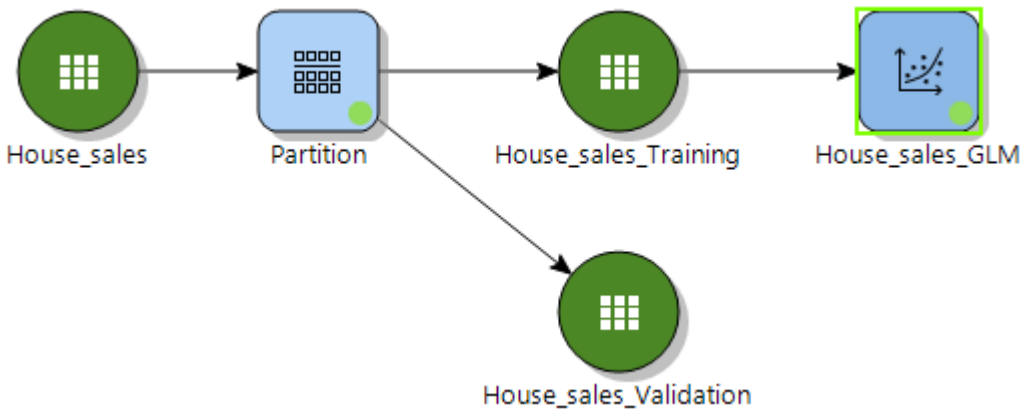
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. The input dataset is converted to a Pandas dataframe, and the linear model functions of the `sklearn` library are applied. The training stops when the total number of iterations reaches the specified limit or when the metric threshold in the stopping criterion is reached.

After running the node, double-click the model node to view the results.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The output dataset can be connected to a Model Analyzer node to analyze the model performance.

Double-click on the Sklearn GLM node to view the training results report. It displays the information on the dependent variable and independent variables, and the values of the evaluation metrics selected in the wizard.

Model Overview

Model Type: Tweedie Regressor

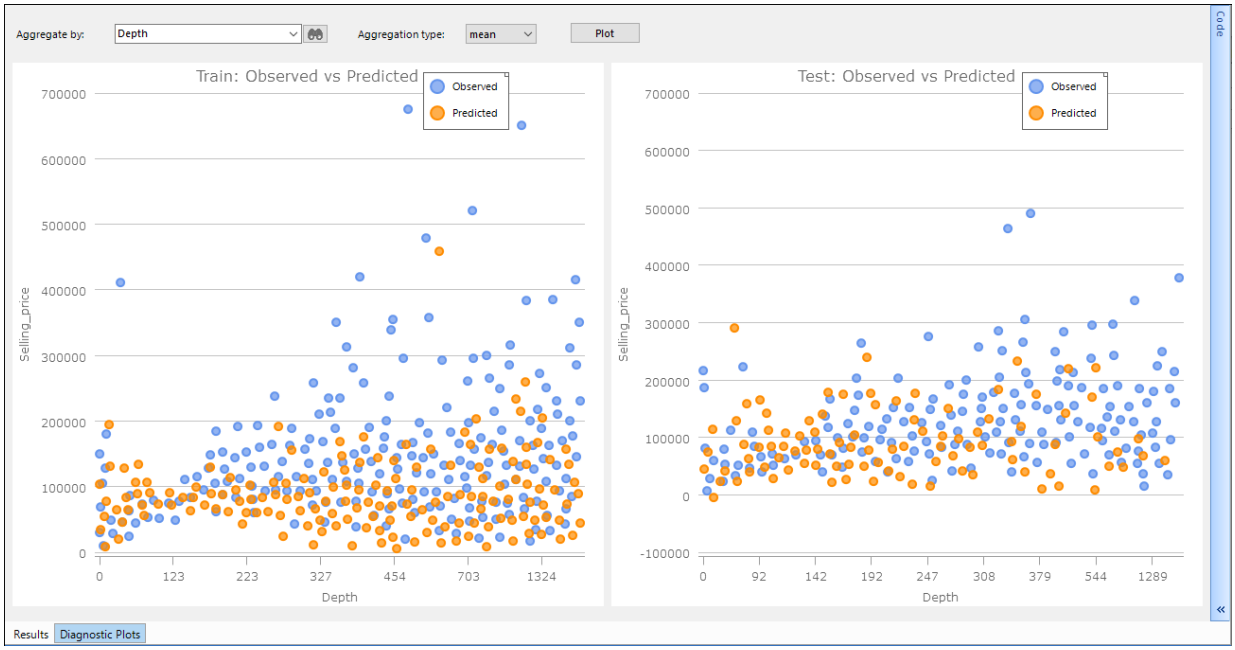
Model Parameters	
Dependent Variable:	Selling_price
Independent Variable(s):	Bathrooms Bedrooms Constr_material Depth Frontage Garage_area Garage_type Half_bathrooms Latitude Longitude Lot_size Rooms Stories Total_living_area Year_built

Training Results	Train	Test
Deviance:	1	1
Tweedie Deviation Regression Loss:	N/A	N/A
MAE:	18573.8969192997	18729.4528098179
MSE:	911021508.6766	984171657.829598
RMSE:	30183.1328505939	31371.5102892672

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The **Diagnostic Plots** tab shows the observed and predicted values of the dependent variables aggregated by a selected independent variable on the training and test samples.

- Select the desired independent variable to aggregate by.
- Select the aggregation type (*mean* or *sum*).
- Select the weight, if applicable.
- Click the **Plot** button to build the plots.



After training, the model can be validated on test data or used for scoring new data. See also the Sklearn GLM Validation and Sklearn GLM Scoring topics.

See also: Model Analyzer

Logistic Regression Model

A **Logistic Regression** model node can be added to the workflow in any of the following ways:

- Drag a connection arrow from a dataset node to any area on the Workflow canvas background. From the context menu that appears when you release the mouse button, select **Model --> Logistic Regression** as in the example screenshot below. A new **Logistic Regression** node will be created.

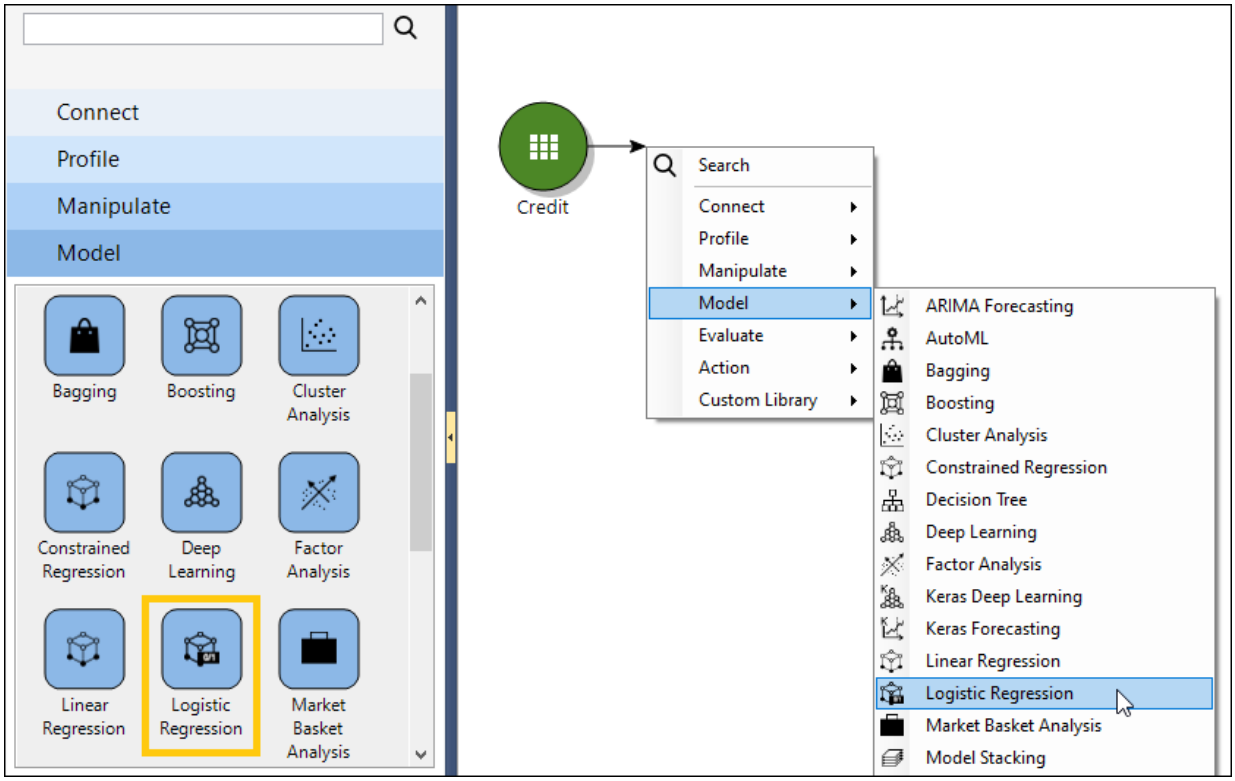
OR

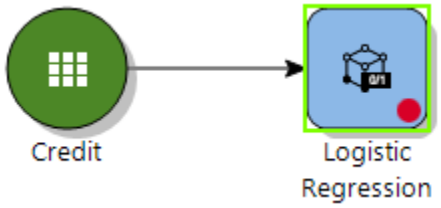
- Right-click on the Workflow canvas and select **Model --> Logistic Regression** from the context menu. Connect a dataset to the **Logistic Regression** node.

OR

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

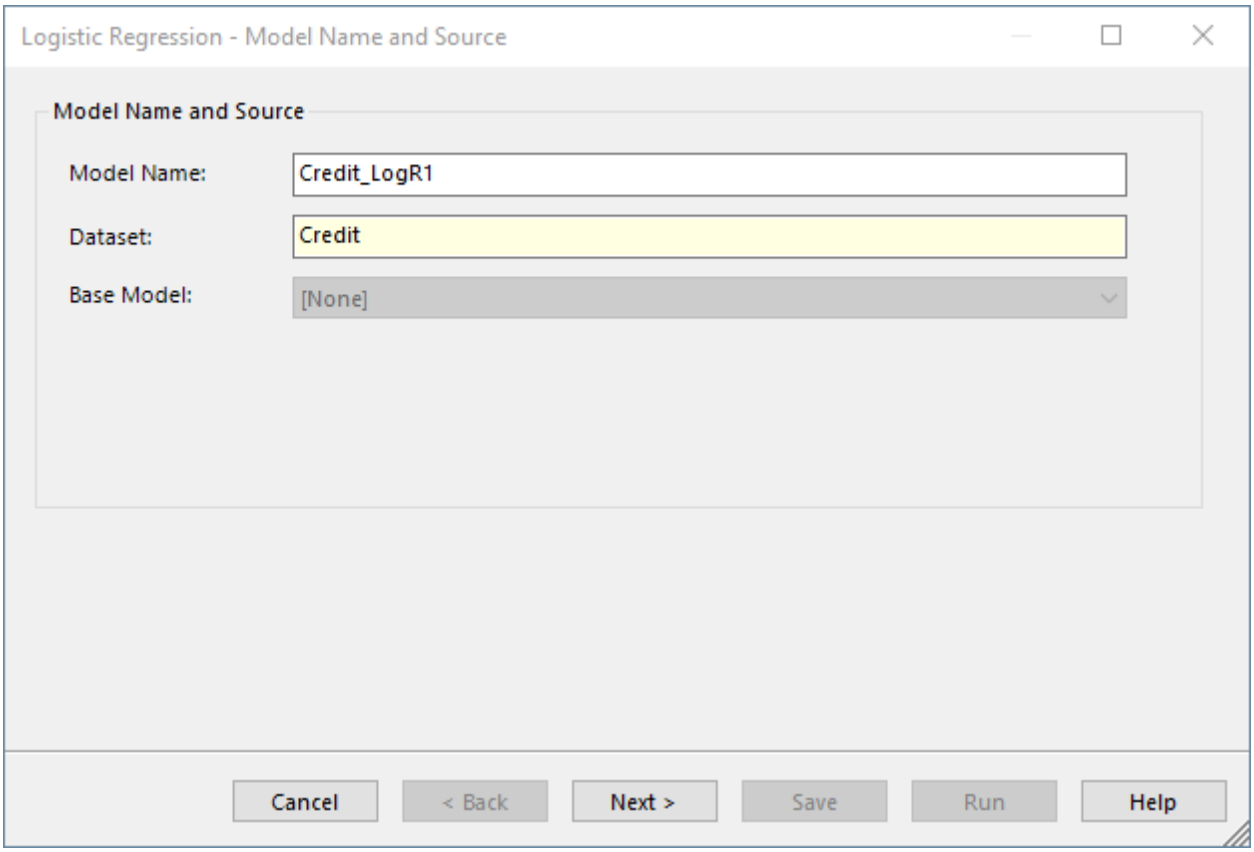
- Open the **Model** palette and drag the **Logistic Regression** node onto the workflow canvas. The **Logistic Regression** node is gray while its inputs are undefined. Connect the input dataset to the **Logistic Regression** node.





Double-click on the Logistic Regression node added to the canvas, or right-click on it and select **Modify** from the context menu. Specify the model name and select the base model if applicable.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Model Nar The name of the model that will be produced when running the model node. The model object with this name will appear in the Project Explorer after running the node.

Dataset The name of the input dataset for the model.

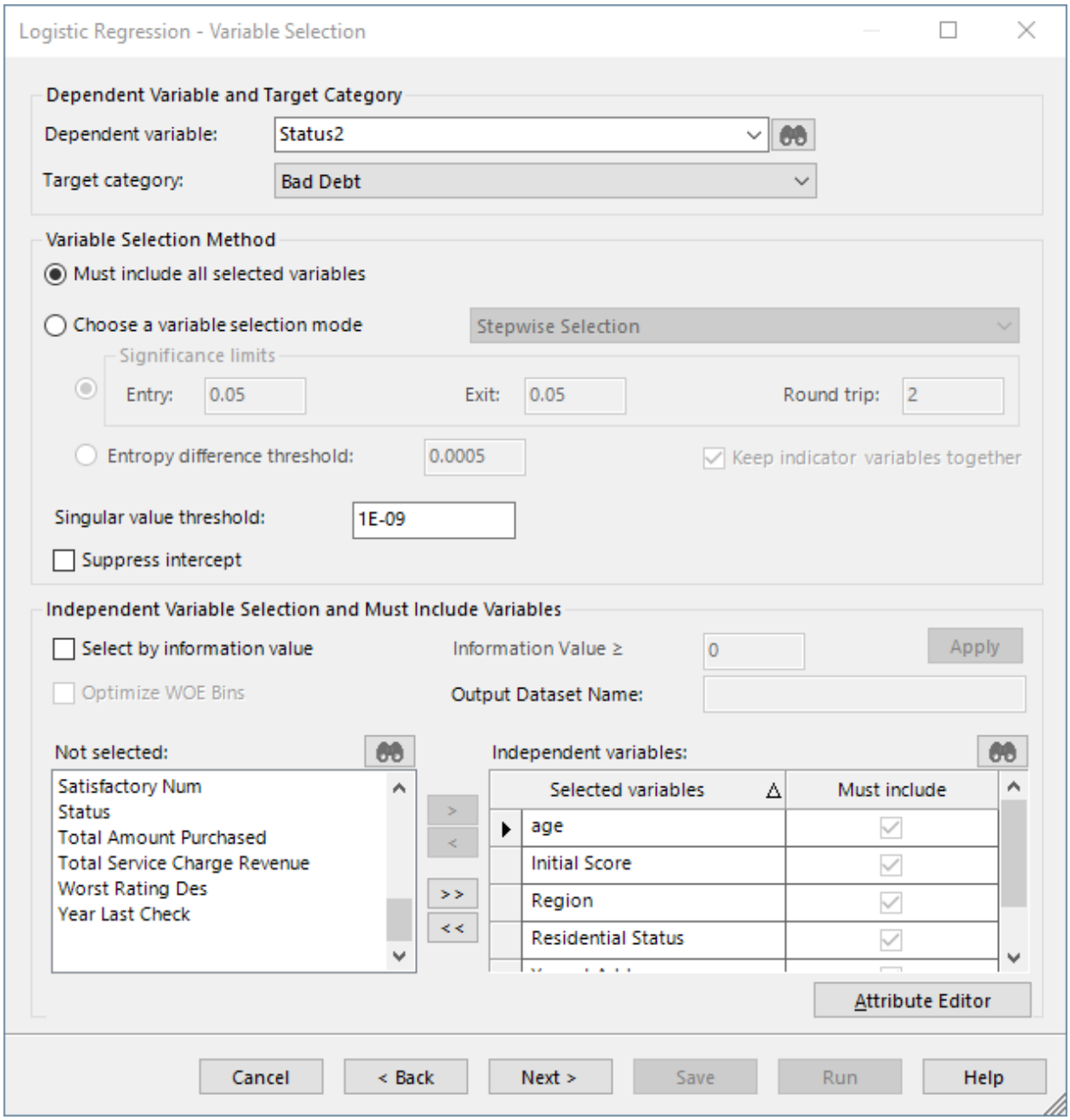
Base Mod The model definition to use as a template for the new predictive model. This requires connecting the model you intend to use as a template to the current Logistic Regression node. In this case the current regression will have two inputs: a dataset and another model. All applicable training parameters and attributes of the template model will be copied and used for the new model. For a base model, you can select any other logistic regression model node in the workflow. This function is convenient for retraining models and inserting new models that inherit some or all of the model parameters of an existing model (at least the set of independent variables).

Click Next to select the variables and specify variable selection options.

Variable Selection

In the Variable Selection page, select the dependent variable (DV) and the independent variables (IVs) for your model:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Dependent Variable

The variable whose outcome is to be modeled. The Dependent Variable drop-down list contains variables admissible for this role. For logistic regression, these are discrete variables of type Number, Integer, Date, Time, or Timestamp with cardinality not exceeding 5, and any variables of type String or Boolean.

Note: If you want to create a multinomial logistic regression with a numeric or date/time DV of cardinality higher than 5, you can override the above restriction. For example, if you want to choose a variable of type "Number" or "Integer" with cardinality 7 as the dependent variable, click the **Attribute Editor** button, change the ***Role*** of this variable to '*dependent*', and click OK.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Alternatively, convert this variable to the String type prior to adding the model node. This can be done using the Variable Transformations node.

Target Catego	The value of the dependent variable whose probability the model should try to predict.

Must include all selected variables

If this option is selected, then all the variables selected as **Independent variables** in this wizard page will be used when training the model, regardless of their significance.

Choose a variable selection method

Stepwise	Include variables by considering the Entry and Exit Significance Levels.
Forward Selection	Starts with no variables and then includes variables based on the rank order of variables from the best to the worst. The rank order is determined by a log-likelihood ratio test.
Backward Selection	Starts with all the variables and then excludes variables based on the rank order of variables from the worst to the best. The rank order is determined by a log-likelihood ratio test.
R-Squared Selection	Check through all possible combinations of coefficients and produce a ranked list with R-squared increasing as various combinations go in.
Variable Sequence My Choice	Include variables based on the user's custom selection. The variables can be chosen on the basis of their significance (the P-value computed for each variable). See Variable Sequence Selection for details.

Significance Limits

Entry	The entry significance level determines when a variable is qualified to be included in the model. This parameter is used for the **Stepwise** and **Forward** selection methods.
Exit	The exit significance level determines when a variable is disqualified and excluded from the model. This parameter is used for the **Stepwise** and **Backward** selection methods.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Round Trip	The number of entry-exit-entry-exit sequences before a variable is disqualified and excluded from the model.

Entropy Difference Threshold

"Entropy difference threshold" defines a stopping criterion for model training with the *Stepwise*, *Backward*, and *Forward* variable selection methods. If this option is selected, the model training is stopped when the change in Entropy Explained in consecutive models in the iterative process is less than the user-defined level. The default threshold is 0.0005.

This option is not applicable when Significance Limits are specified or when the variable selection method is "*R-Squared Selection*" or "*Variable Sequence Of My Choice*".

After the training is finished, you can check the Entropy Explained values in the Results tab of the model in the sequence.

Keep indicator variables together

When this option is selected, the categories of each discrete variable are treated as a group. When a discrete variable is included or excluded from the model all the categories within the variable are included or excluded together as a group. For example, ***marital-status*** contains seven categories: **Single**, **Married**, etc. If ***marital-status*** is excluded, all the categories are excluded from the model. When this option is unchecked, each category is evaluated separately and will be included or excluded individually. This option is selected by default.

This option is not applicable when "*Variable Sequence Of My Choice*" is selected.

Singular Value Threshold

Used to adjust the collinearity detection sensitivity. The threshold determines when any two independent variables are considered to be collinear and one of them must be ignored (removed from the model) to avoid the risk of biasing the model.

If there is an exact linear dependency between any variables, the corresponding singular value is equal to zero. A small singular value (eigenvalue) indicates collinearity issues, so when small singular values occur, this threshold determines how small the singular values may be without causing any problems to the least squares estimation.

The default value for this threshold is 1.0E-09.

Independent Variable Selection and Must-Include Variables

Variables on the left are not included in the model. Variables listed on the right will be included. You can move the variables to and from the Independent Variables list by highlighting them and using the arrow buttons. Double-clicking the individual variables will also move them from one list to the other.

To select more than one variable in the lists, use mouse dragging, or mouse clicks while holding down the SHIFT key or the CTRL key. If there are a lot of variables (hundreds or thousands), it is convenient to use Field Finder (the binoculars button ).

If you chose a particular variable selection method, you have an option of selecting the variables that must be forced into the model regardless of the selection method. The **Must include** column is only enabled if you have chosen a particular Variable Selection Method. During the variable selection process, the algorithm will exclude independent variables from the model if they do not affect the model. However, you may want to make sure that specific variables are kept in the model regardless of the algorithm. The variables that are in the **Must include** list will be forced into the model regardless of the method selected.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Select by information value

This option is applicable only if the dependent variable is binary. Choose this option and specify the **Information Value** threshold if you would like to auto-select independent variables based on their Information value. Clicking the **Apply** button on the right will select all variables with Information value higher than the specified threshold and move them to the **Independent variables** list. The default threshold value is 0.

Information value is defined in terms of the *Kullback-Leibler (KL) divergence*. See the formula in Subsection 1.5 "*Information Value in Trees with Binary DV*" in the Decision Tree section of Mathematical Formulations.

Optimize WOE Bins

This option applies only to variables produced by Weight of Evidence transformations in the WOE node. It is applicable only under the following conditions:

- The dependent variable is binary
- The input dataset contains at least one WOE variable
- The option '*Must include all selected variables'* is chosen

Under these conditions, selecting this option turns on the ***WOE binning optimization algorithm*** that automatically adjusts the boundaries of WOE bins for the WOE variables in the process of model training. The number of bins are determined from the WOE expressions.

In the field "**Output Dataset Name**", specify the name of the new dataset with optimized WOE variables. The output dataset is the same as the input except that the WOE variables have modified WOE expressions with interval boundaries adjusted during the model training.

For each WOE variable included in the *Independent variables* list, three columns are enabled: "**Monotonic**", "**No Pure Nodes**", and "**Equal Size**". Use the first two columns ("Monotonic" and "No Pure Nodes") to indicate whether to optimize the selected parameter for them or not. In the third one - "Equal Size" - you can specify a numeric value from 0 to 1, which means the degree of enforcing equal size bins (degree of the penalty for unequal size bins). 0 means "not enforced", 1 means "fully enforced with high priority". **0.5** is the default. The "Monotonic" and "No Pure Nodes" options are selected by default, turning on the corresponding optimizations for the WOE variables.

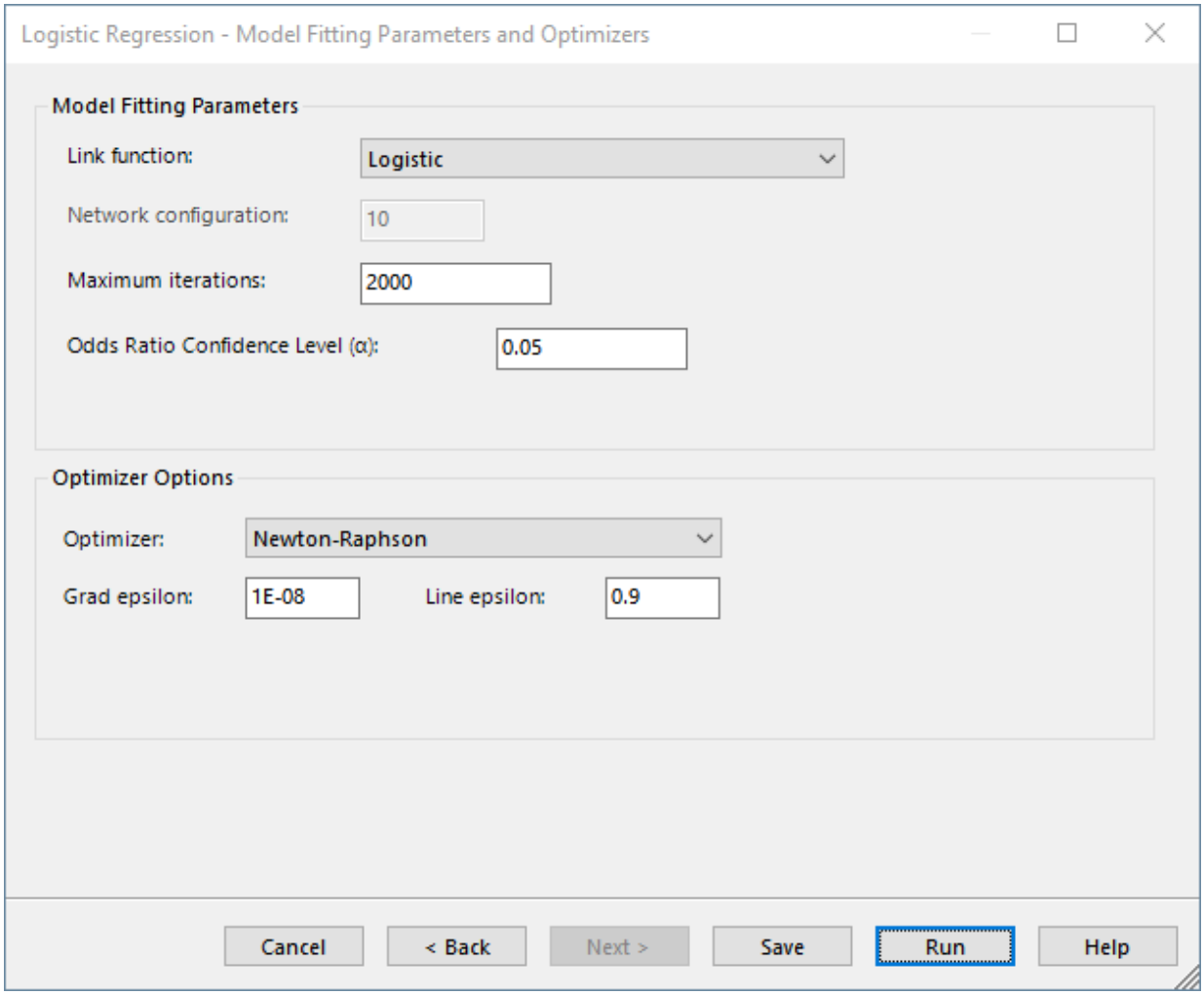
Attribute Editor

The Attribute Editor button brings up the predictive model attribute editor. The attribute editor allows you to modify variable attributes such as missing value substitution options, etc. For details, see the topic Model Attribute Editor.

Click **Next** to specify the model fitting parameters.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Model Fitting Parameters and Optimizers



Maximum Iterations	The number of times the model calculates the error, adjusts the weights, and feeds the error back through the model.
Link function	The link function specifies a nonlinear transformation of the predicted values so that the distribution of predicted values is one of several special members of the exponential family of distributions (e.g., gamma, Possion, binomial, etc.). The link function is used to model responses when a dependent variable is assumed to be nonlinearly related to the predictors. The available link functions are: ***Logistic*** (most commonly used, selected by default), ***Inverse Cumulative Normal, Complementary Log-Log***, and ***Log-Log***.
Compute Error	If you select this option, the algorithm will compute errors and modify the training of the predictive model as it trains.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Model Optimizers

Choose the optimizer and specify parameters for it.

***Newton-Raphson* (*default*)**	The Newton-Raphson optimizer (selected by default). A method that finds the slope of the function at the current point and uses the zero of the tangent line as the next reference point. The process is repeated until the root is found.
Levenberg-Marquardt	A method of finding the minimum of a function that is a sum of squares of nonlinear functions. The **Levenberg-Marquardt** optimizer is only defined for the sum squared error functions.

Optimizer Parameters:

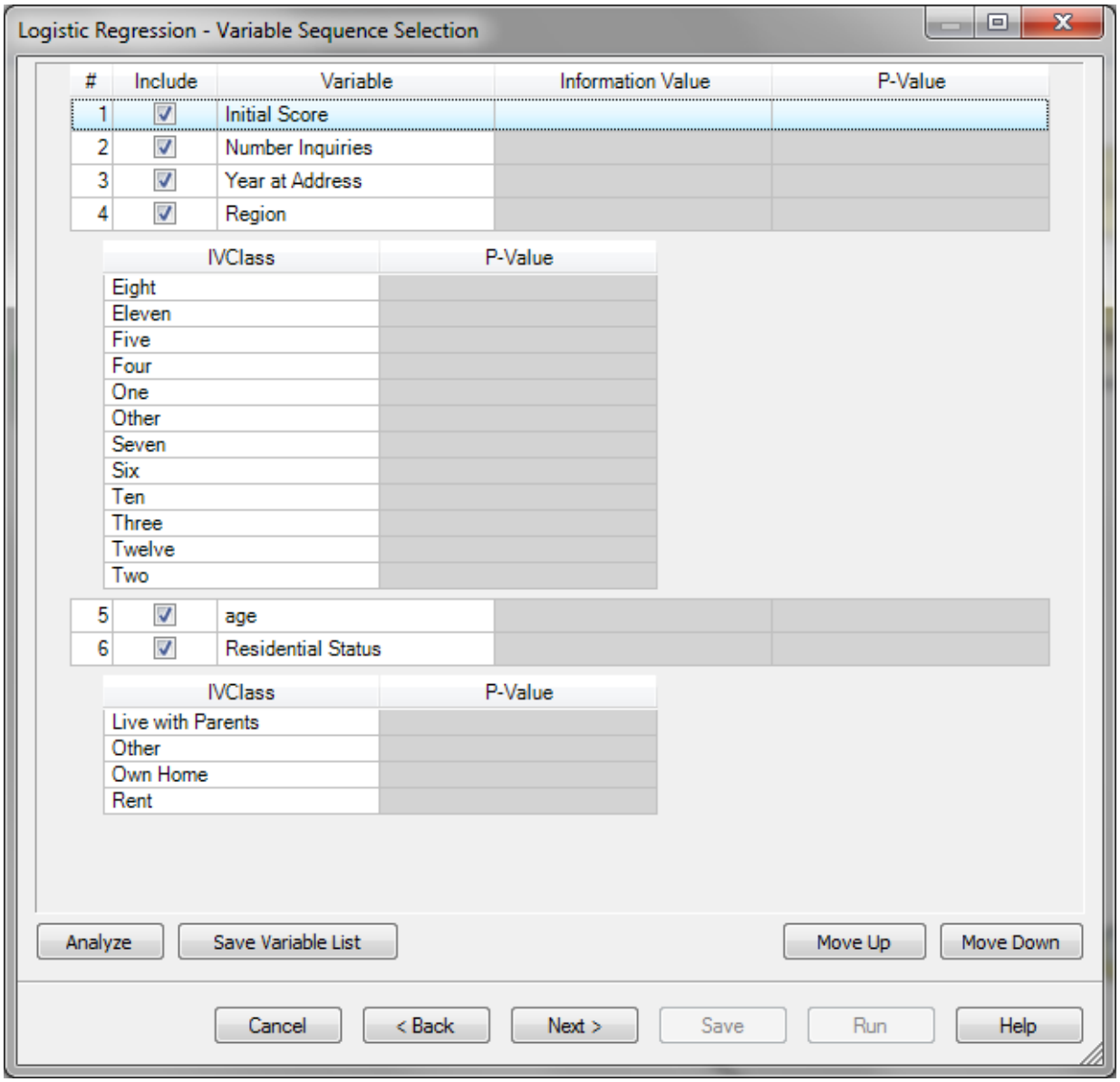
Grad Epsilon	Convergence epsilon for gradient.
Line Epsilon	Similar to Grad Epsilon, but used by line optimizer.

Custom Sequencing

Custom Sequencing is applicable to Regression models and Deep Learning models.

Selecting **Variable Sequence Of My Choice** as the variable selection mode allows you to define a custom order of including variables in the model. The variables can be chosen on the basis of their significance (the *p*-value computed for each variable) and Information Value. For details on these and otehr statistics, see Mathematical Formulations: Decision Tree.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



More precisely, the model under 'Variable Sequence Of My Choice' starts with no variables and then includes variables based on their order in the list of variables in this page of the wizard. The buttons "**Move Up**" and "**Move Down**" allow you to change the order of inclusion of variables in the sequence. It starts with the topmost variable in the list, then adds subsequent variables one by one until it adds all of them, and finally chooses the best sequence as the "Currently Selected Sequence".

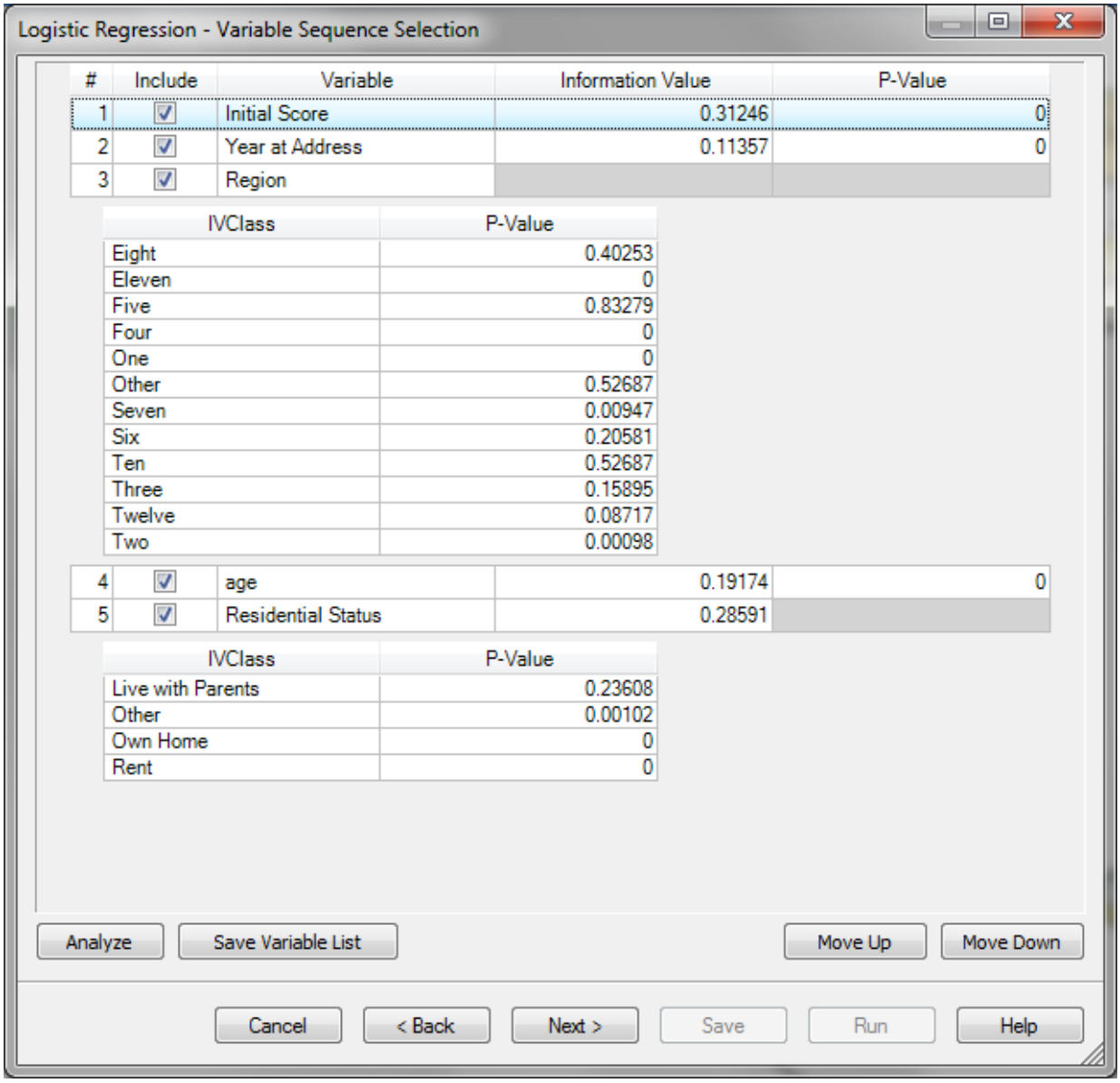
The details in the statistics for each sequence can be found in the Results tab of the model when it is trained (open the drop-down list "Output To View" at the top of the Results tab).

The significance for each variable is computed by building a single-independent-variable regression model with this variable and then computing the P-value.

Click the **Analyze** to compute the *p*-value and Information Value in this page.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Based on the computed parameters, decide which variables to include and move up or down.

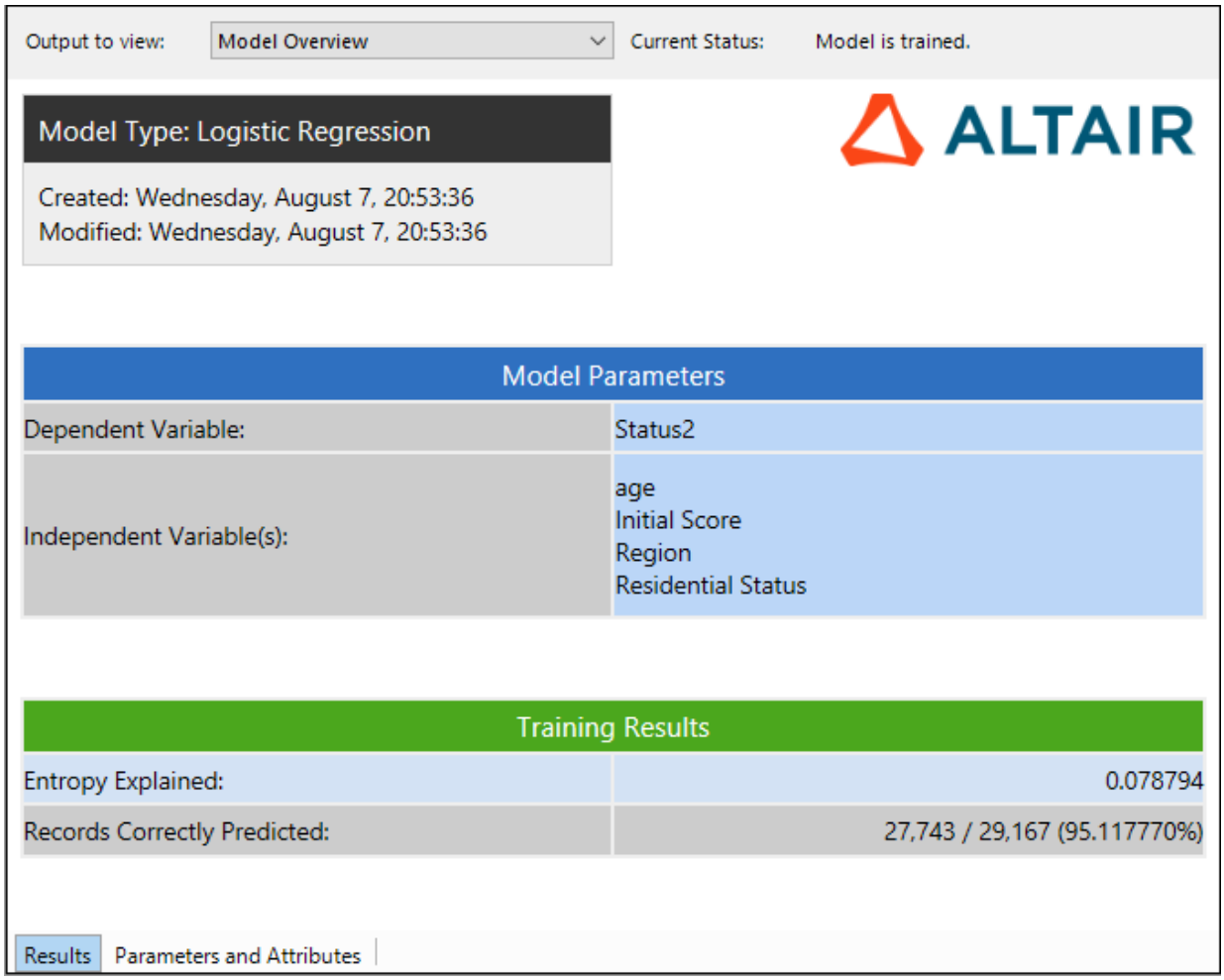


Logistic Regression Results

To view the model training results, double-click on the model node or right-click on it and select the **Open View** command from the context menu.

The default model results view displays a short summary report for the final model (*Model Overview*). It shows the list of the input variables, the value of *Entropy Explained* and the number and percentage of records for which the dependent variable outcome was correctly predicted during the resubstitution test.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



In the same view, select **Currently Selected Sequence** from the **Output to view** drop-down list.

The ***Sequence Statistics*** view is the detailed view of the Logistic Regression results. It includes model fitting summary statistics, null hypothesis testing results, independent variable statistics with model parameters, variance inflation factors, and the actual model equation.

Note that the result of model training performed with a variable selection method (such as Stepwise) contains the output of the final model as well as all intermediate models built in the variable selection process. In this case, the Output to View drop-down list contains multiple sequences (models).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Output to view: Currently Selected Sequence Current Status: Model is trained. This is equivalent to sequence 1

[Credit].[Credit_LogR1]

Sequence #1

Model Fitting Summary for Status2

Chi-Square:	896.441707	Chi-Square Degrees Of Freedom:	18
P-Value:	0.000000	Generalized R^2:	0.030267
Entropy Explained:	0.078794	AIC:	10,518.605466
		BIC:	10,675.940537
Percent Concordant:	71.776044	Somer's D:	0.436402
Percent Discordant:	28.135850	Gamma:	0.436787
Percent Tied:	0.088106	tau a:	0.040533
Total Pairs:	39,506,032	c:	0.718201

Global Null Hypothesis Testing (BETA=0)

Statistic	Chi-Square	DF	p-value
Likelihood Ratio	896.441707	18	0.000000
Score	1,082.629497	18	0.000000
Wald	896.234060	18	0.000000

	Negative 2(Log-Likelihood)	DF
Null Model	11,377.047173	-
Full Model	10,480.605466	18

Independent Variable Statistics

Status2 = Bad Debt

Variable Name / Value		DF	Model Parameter	Parameter Standard Error	Wald Chi-Square	p-value	-95%	Odds	+95%	Standardized Parameter	Standardized Parameter Error
Initial Score		1	-0.001371	0.000302	20.561908	0.000006	0.998038	0.998630	0.999222	-0.181442	0.040013
	Eight	1	-9.784017	246.019686	0.001582	0.968277	0.000000	0.000056	1.456101E+205	-0.151557	3.810907
	Eleven	1	-0.079588	0.086378	0.848950	0.356850	0.779669	0.923497	1.093858	-0.031561	0.034254
	Five	1	0.163090	1.030481	0.025048	0.874247	0.156200	1.177143	8.871080	0.004773	0.030157
	Four	1	1.443566	0.097675	218.425309	0.000000	3.497772	4.235776	5.129493	0.266571	0.018037
	One	1	0.233295	0.094536	6.089979	0.013595	1.049179	1.262754	1.519805	0.067752	0.027454

Results | Parameters and Attributes

The Currently Selected Sequence represents the best (final) model in the case of Stepwise or other variable selection method used. The number of the best model is indicated by the label at the top of the view, for example: "This is equivalent to sequence 5".

The Independent Variable Statistics section shows all model parameters. The **Model Parameters** are coefficients of the variables in the model, represented by the model equation discussed below.

Entropy Explained is a measure that shows how much variation of the dependent variable was explained by the model. Its values are between 0 and 1. The higher its value, the better. Entropy Explained is an important, but not sufficient parameter characterizing the model performance. Evaluate it in combination with the values of -2 Log likelihood, AIC (Akaike Information Criterion), BIC (Bayesian Information Criterion), Chi-Square, P-Value, independent variable statistics (model parameters, significance, standardized error), and Variance Inflation Factors.

AIC is an important measure of the goodness of fit which penalizes excessive model complexity at the same time. The lower the AIC value, the better. The best model is selected based on this criterion. For example, if you compare Entropy Explained and AIC in all sequences of a Stepwise

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

model, you will notice that the best model has the highest value of Entropy Explained and the lowest AIC value.

BIC is a model selection criterion similar to AIC, but it imposes higher penalty for the number of model parameters.

P-value is the probability of obtaining current model results using a null (random) model. More formally, it is the probability of rejecting the *null hypothesis* when it is actually valid. The lower this value, the better.

The number of **Degrees of Freedom** is the number of independent parameters in the model. Note that, as a result of normalization, all String variables *are* represented by sets of dummy (indicator) variables assuming values 0 and 1.

The **-2Log likelihood** is a measure of how well the model fits the data. The original likelihood is the probability of the observed model results given the model parameters. The likelihood close to 1 means the goodness of fit close to perfect, which means the smaller the value of -2 log likelihood, the better the model fit.

The value of **Significance** helps to determine whether there is a significant relationship between the dependent variable (DV) and an independent variable (IV) in the model. It expresses the likelihood of the *null hypothesis* that the model parameter is zero, i.e., that the corresponding variable is *statistically insignificant* for the DV. Thus, the closer the Significance value to zero, the more significant the variable is to the DV. Significance of 0.05 (5%), is a typical cutoff value for considering the variable not significant enough.

Standardized Parameters are Model Parameters rescaled to a common basis by dividing each by the standard deviation of the model parameters. From the Standardized Parameter values it can be easily seen which variable contributes the most to the model equation.

The **Variance Inflation Factors** help to estimate the degree of multicollinearity between the independent variables. The VIF for the ith independent variable is defined as $VIF_i = \frac{1}{1-R_i^2}$

The square root of the variance inflation factor for a given variable tells you how much larger its standard error is, compared with what it would be if that variable were uncorrelated with the other predictor variables in the model. Usually the degree of multicollinearity is considered to be high if VIF is greater than 5.

The **model formula** for the probability of the target DV value is displayed at the bottom of the Sequence Statistics page. The formula is **exp(Z0)/(1 + exp(Z0))**, where Z0 is the log odds expression shown below. The coefficients of the linear expression Z0 are the values in the *Model Parameters* column of the Independent Variable Statistics table above. The last term is the *intercept*. At the model deployment stage when the model is *scored*, the values of the required variables in the input dataset are substituted for the model parameters in the formula, and the probability is calculated.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Constrained Regression

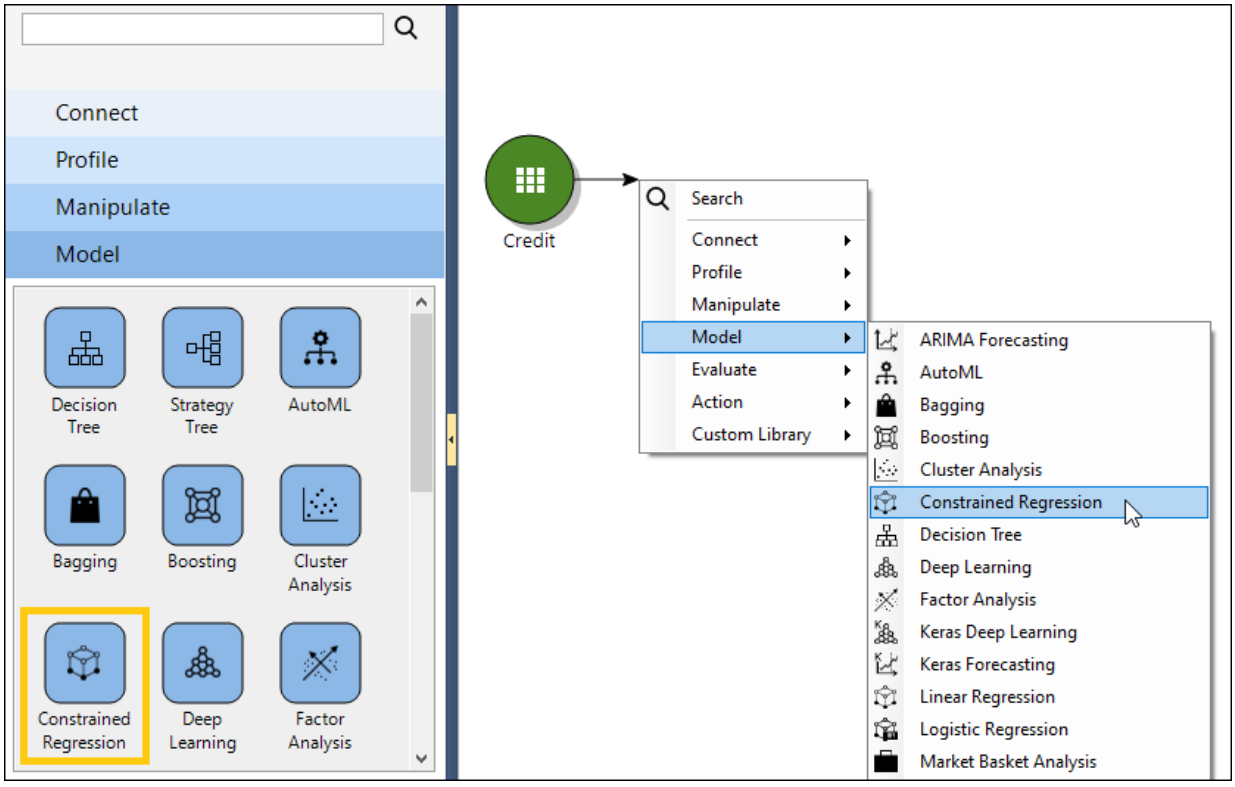
The **Constrained Regression** model node in the Modeling palette provides the capability to build a Logistic Regression model with constraints imposed on model parameters. The constraints are defined in the form of inequalities. This feature can be especially useful in building scorecards, which often need to comply with various business constraints and external regulations.

To add a Constrained Regression model to your workflow:

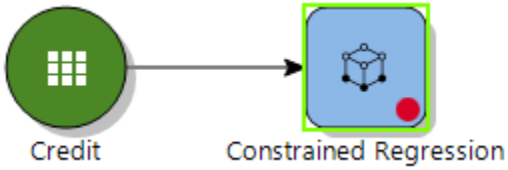
- Drag a connection arrow from a dataset node to any area on the Workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Model --> Constrained Regression**, as shown in the example below. A new **Constrained Regression** model node will be created.

OR:

- Drag the **Constrained Regression** node from the **Model** palette onto the canvas and connect the required input dataset to the node.

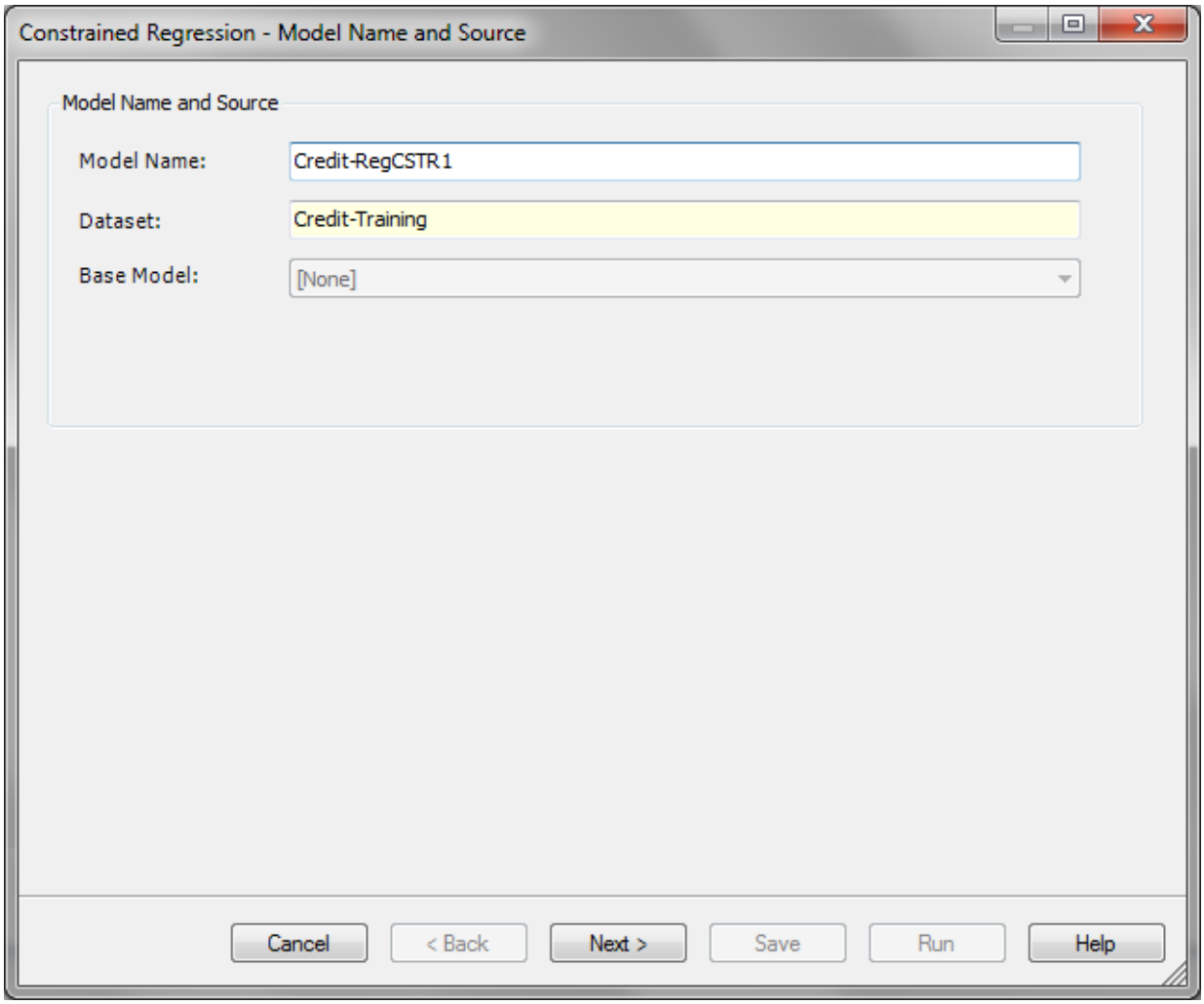


If the wizard does not open automatically, double-click the new node or right-click on it and select **Modify**.



Specify the model name in the first page of the wizard:

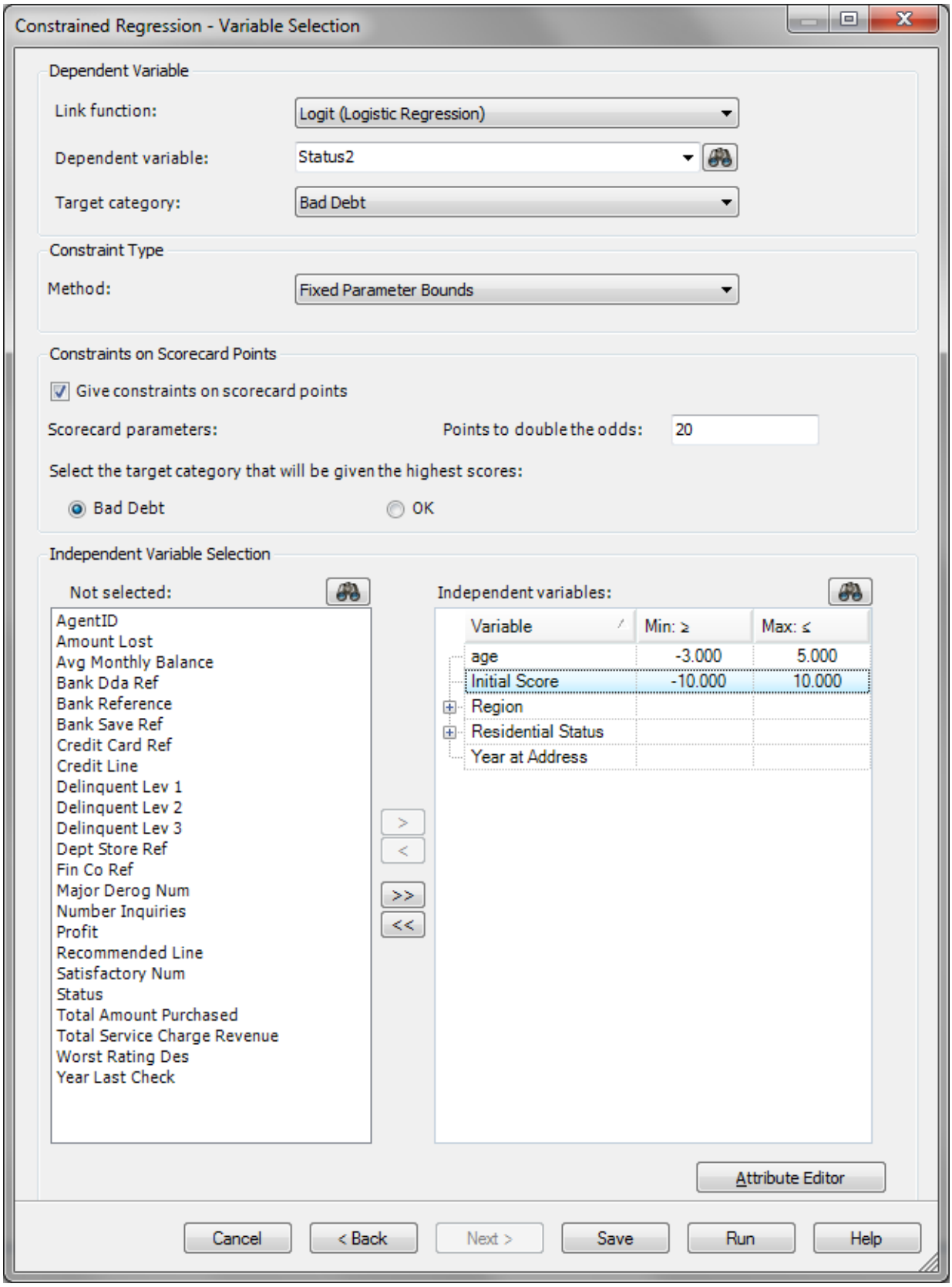
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



In the second page:

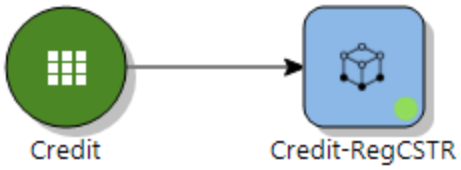
- Choose the link function (Logit for Logistic Regression).
- Select the dependent variable. Only binary variables are available for selection.
- Select the constraint type ("Fixed Parameter Bounds" is the only method available in this version).
- Select the option "Give constraints on Scorecard points" if necessary. In that case you will need to specify some scorecard scaling parameters: Points to double the odds and select the target category that will be given the highest scores.
- Select the independent variables. For each of the selected independent variables you can define parameter boundaries in the Min and Max columns, which define the range of the coefficients of the corresponding variables in the regression model. For a discrete variable, you can define the parameter boundaries for every dummy variable created for it.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the operation, double-click the model node to view the results.



In the sequence statistics view of your model results, the model parameters will satisfy the constraints you entered in the wizard,

Regularization

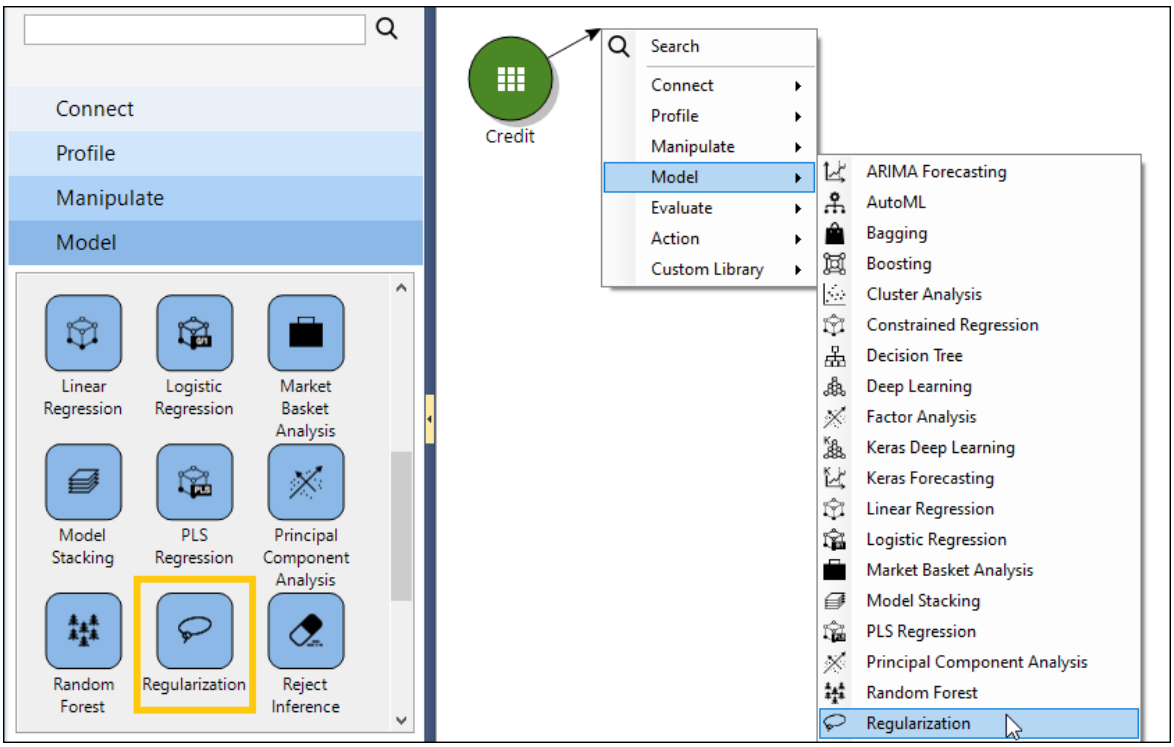
The **Regularization** node in the Modeling palette provides the techniques of Regularized least squares (RLS), including the Lasso, Ridge Regression, and Elastic Net methods.

To add a Regularization operation to your workflow:

- Drag a connection arrow from a dataset node to any area on the Workflow canvas background. This brings up a context menu that allows you to select the desired operation. From the context menu, select **Model --> Regularization** as shown in the example below. A new **Regularization** process node will be created.

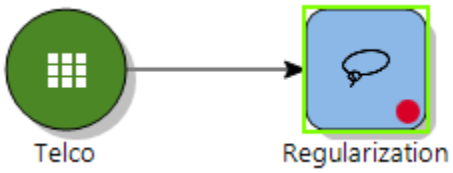
OR:

- Drag the **Regularization** node from the **Model** palette onto the canvas and connect the required input dataset to the node.

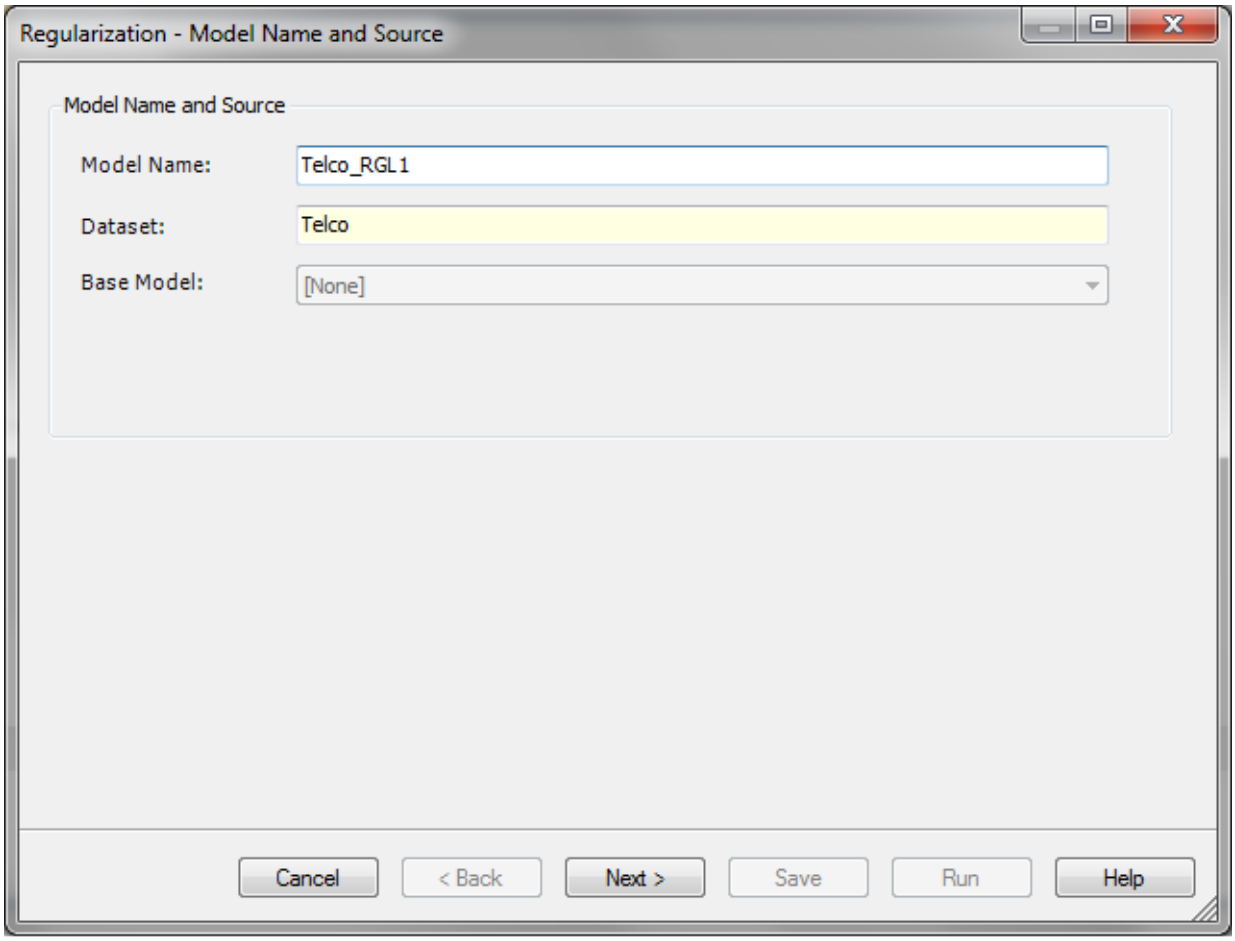


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If the wizard does not open automatically, double-click the new node or right-click on it and select **Modify**.



Specify the model name in the first page of the wizard:



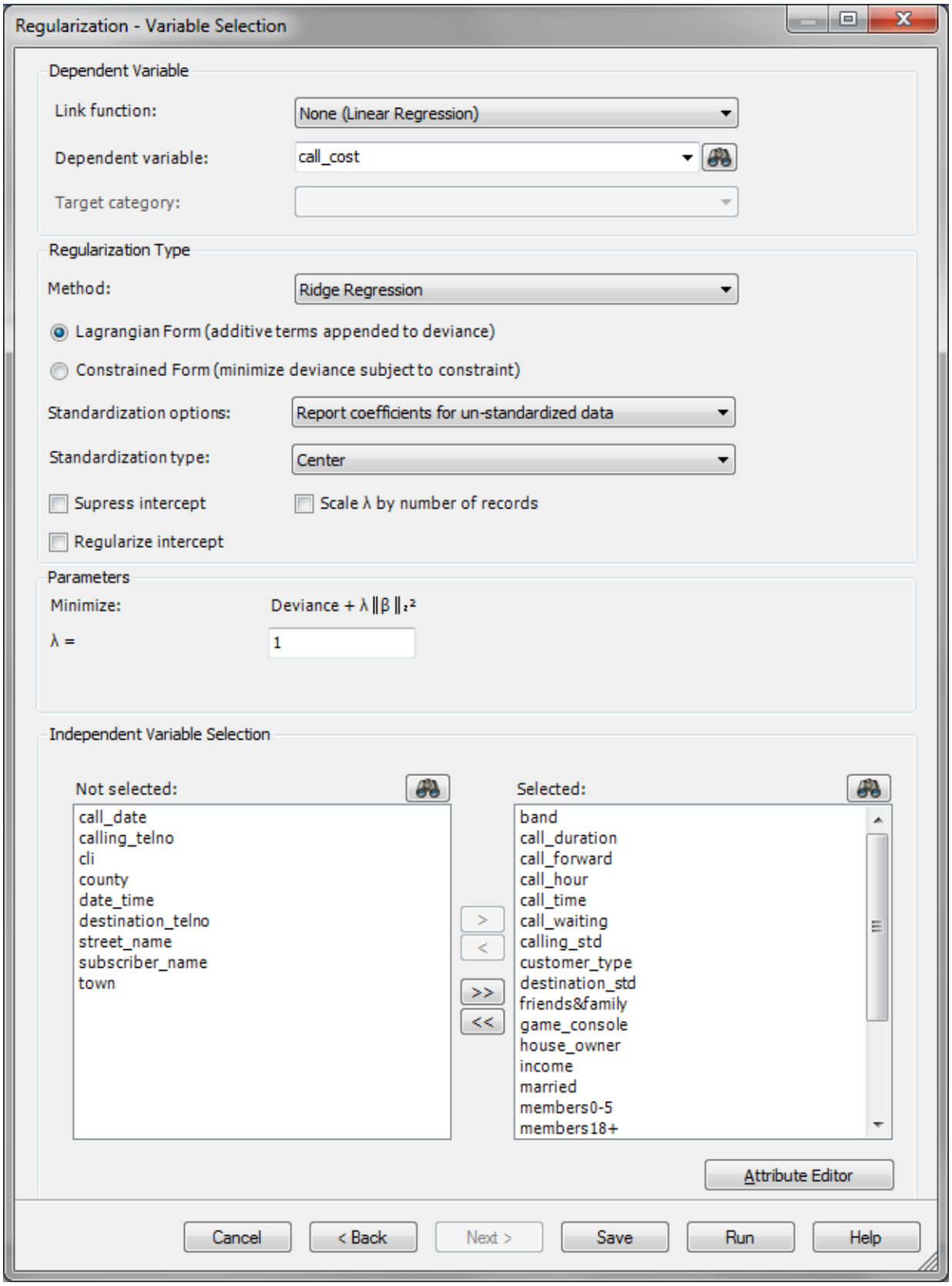
In the second page:

- Choose the link function (*None* for Linear Regression, or *Logit* for Logistic Regression).
- Select the dependent variable.
  - Only numeric variables are available for selection if you selected Linear Regression above.
  - Only binary variables are available for selection if you select Logit (Logistic Regression).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Select the constraint / regularization type and parameters.
- Select the independent variables..

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the operation, double-click the model node to view the results.



PLS Regression Model

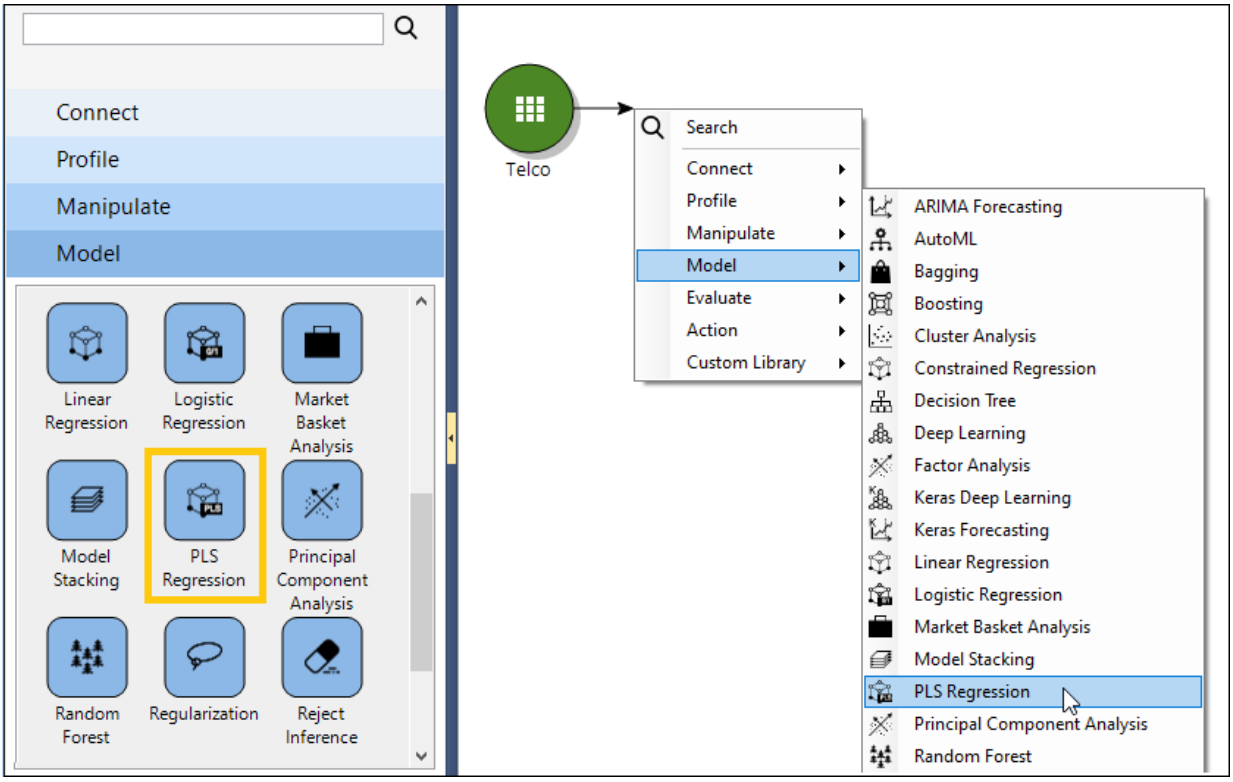
The **PLS Regression** node in the Model palette represents ***Partial Least Squares Regression***. It provides the ability to choose factors for regression (similar to Principal Component Analysis), specify how many of these factors to use, and perform a linear regression.

To add a PLS Regression operation to your workflow:

- Drag a connection arrow from a dataset node to any area on the Workflow canvas background. This brings up a context menu that allows you to select the desired operation. From the context menu, select **Model --> PLS Regression** as shown in the example below. A new **PLS Regression** process node will be created.

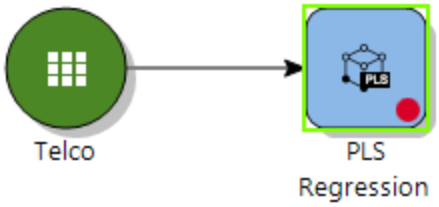
OR:

- Drag the **PLS Regression** node from the **Model** palette onto the canvas and connect the required input dataset to the node.



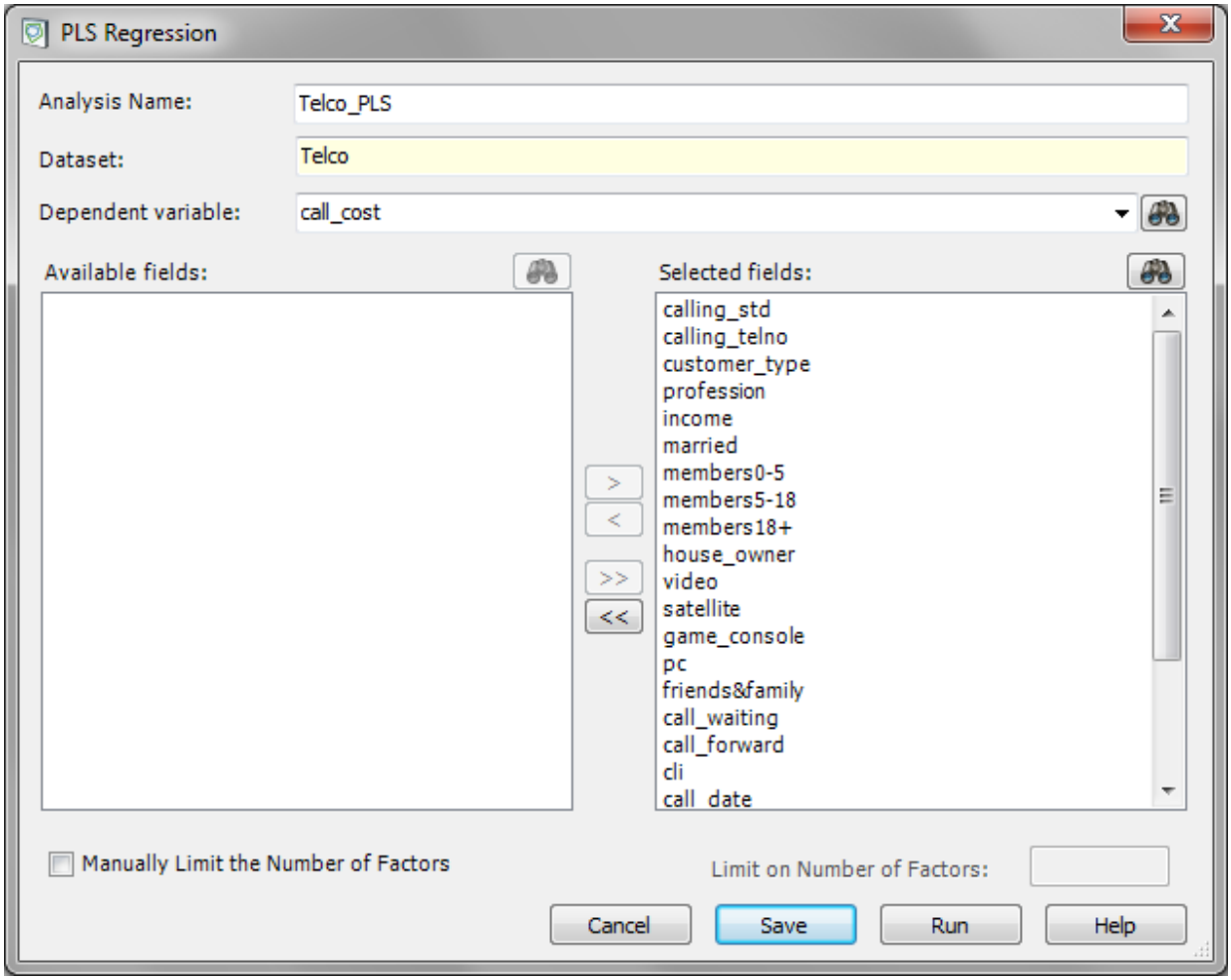
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If the wizard does not open automatically, double-click the PLS Regression node or right-click on it and select **Modify**.



In the PLS Regression wizard:

- Specify the name for the model ("Analysis Name").
- Select the dependent variable. Only numeric variables are available for selection.
- Select the independent variables. Only numeric variables are available for selection.
- Select the option to Manually Limit the Number of Factors, if necessary, and specify the limit.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

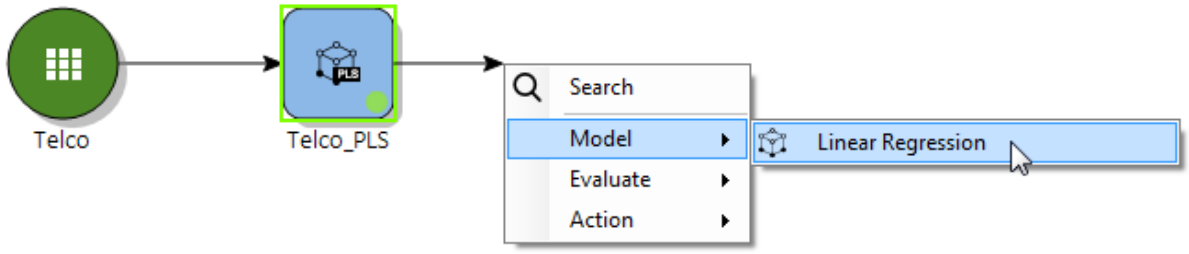
Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the operation, double-click the PLS Regression node to view the results.

To build a linear regression based on the factors produced by the PLS regression node,

- Drag a connection arrow from the **PLS Regression** node to any area on the Workflow canvas background. From the context menu, select **Model --> Linear Regression** as shown in the example below. A new **Linear Regression** process node will be created.

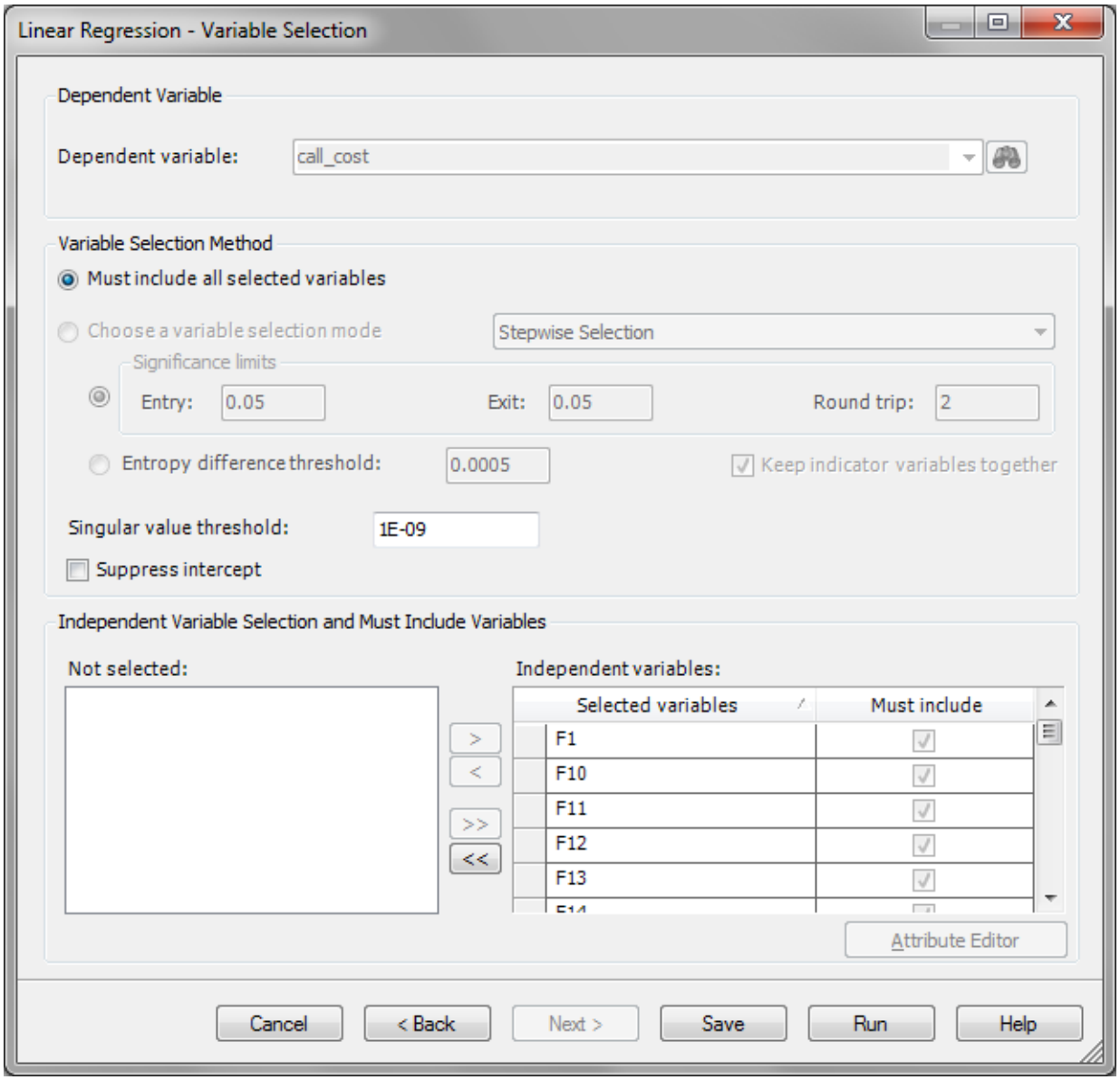
OR:

- Drag the **Linear Regression** node from the **Model** palette onto the canvas and connect the **PLS Regression** node to it.



There is no need to connect the input dataset to the Linear Regression node in this case. In the Linear Regression wizard, the dependent variable is inherited from the PLS Regression node. Select or deselect the required factors and choose other parameters as needed (see the Linear Regression topic for details).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

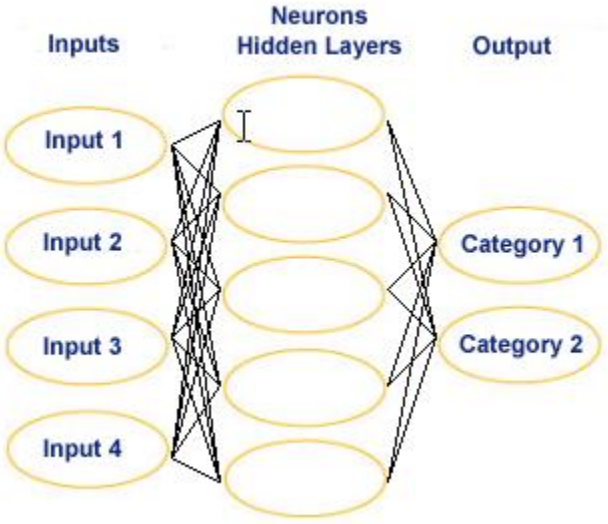


Deep Learning (Neural Network)

Deep Learning

Deep Learning models (Multi-Level Neural Networks) are based on a simplified model of the human brain. They use an internal construct called a “neuron”. Each neuron has a number of inputs that are combined to produce a single output. Outputs from a layer of neurons are fed to another layer called the hidden layer.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The effect of these neural pathways is that they produce pattern recognition.

Like the human brain, neural networks in the Deep Learning models do not explain how they arrive at a particular decision - they simply recognize patterns. If you need to understand your model, build or rebuild it as a decision tree.

Results of the Deep Learning Model

The statistics for each layer can be seen if you select **Currently Selected Sequence** from the drop down list on the Predictive Model Results tab after training is complete.

Deep Learning models can use discrete or continuous variables as dependent variables. The results will be displayed differently depending on whether the DV is continuous or discrete.

Currently Selected Sequence Statistics

Models with Discrete DVs

Sequence Statistics			
Dependent Variable:	**Income**		
Entropy Explained:	**0.345254246461**		
Chi-Square:	**1894.62957408**	Chi-Square Degrees Of Freedom:	**371**
P-Value:	**2.30288368419e-202**		

Entropy Explained	A goodness of fit measure. *Proportion of the variability in the response that is accounted for by the model.*

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Chi-Square P-Value

The P values tell us whether a variable has statistically significant predictive capability in the presence of the other variables, that is, whether it adds something to the equation. A low p-value means that there is evidence to believe that there is a statistically significant linear relationship between the two variables.

Chi-Square Degrees of Freedom

Model Fitting Summary

Model Fitting Summary

Source	Negative 2 Log-Likelihood	DF
Intercept Only	5487.63583215	-
Full Model	3593.00625808	371

Negative 2 Log-Likelihood

The -2LL statistic is the "scaled deviance" statistic for logistic regression and is also called "deviation chi-square". This statistic measures how poorly the model predicts the decisions. The smaller the statistic the better the model.

Models with Continuous DVs

Sequence Statistics

Dependent Variable:	**age**		
Variance Explained:	**0.465324342407**	Adjusted Variance Explained:	**0.422471745958**
F-Ratio:	**10.8587199133**	F-Ratio Degrees Of Freedom 1 / 2:	**371 / 4629**
P-Value:	**0**		

Variance Explained

A goodness of fit measure. Proportion of the variability in the response that is accounted for by the model.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Adjusted Variance Explained	As additional variables are added to a regression equation, the value for ***variance explained*** increases even when the new variables have no real predictive capability. The ***adjusted variance explained*** measure avoids this issue. When variables are added to the equation, ***adjusted variance explained*** does not increase unless the new variables have additional predictive capability.
F-Ratio	Test statistic used to decide whether the model as a whole has statistically significant predictive capabilities.
P-Value	*The P values tell us whether a variable has statistically significant predictive capability in the presence of the other variables, that is, whether it adds something to the equation. A low p-value means that there is evidence to believe that there is a statistically significant linear relationship between the two variables.*
F-Ratio Degrees of Freedom 1/2	

Analysis of Variance

Source	Sum-Of-Squares	DF	Mean-Square
REGRESSION	440247.104444	371	1186.64987721
ERROR	505860.941755	4629	109.280825611
TOTAL	946108.0462	5000	189.22160924

The statistics that are produced for the multi-layer neural network model with a continuous dependent variable form a standard ANOVA table.

Independent Statistics

The independent variable statistic show the parameters for the hidden layers. The hidden layer statistics are shown for each layer. The layers are labeled as:

hidden_layer#_neuron#

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

H1_5		
Variable Name / Value		**Model Parameters**
age		1.856643
education	10th	-0.405961
	11th	-0.839169
	12th	0.119026
	1st-4th	0.527439
	5th-6th	0.179783
	7th-8th	-0.205366
	9th	0.139670
	Assoc-acdm	-1.018960
	Assoc-voc	0.723240
	Bachelors	-0.906371
	Doctorate	-0.898373
	Masters	-0.397652

Income = >50K	
Variable Name / Value	**Model Parameters**
hidden_001_001	1.66330312908
hidden_001_002	1.17525049159
hidden_001_003	-0.163397668088
hidden_001_004	1.14727884546
hidden_001_005	-1.83303772025
hidden_001_006	-1.83565377469
hidden_001_007	-0.0460596520072
hidden_001_008	-1.70900192799
hidden_001_009	0.736393234601
hidden_001_010	1.09305274794
#INTERCEPT#	-2.86366428467

For more details about the Multi-Level Neural Network, see the topic Deep Learning (Multi-Level Neural Network) in Mathematical Formulations.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Inserting a Deep Learning Model

Prerequisite: Before inserting a Deep Learning model (Multi-Layer Neural Network), if you want to guard it against overfit, partition your training dataset into two partitions. Then you will be able to use the second partition for testing to prevent model overfit. See the topic Model Overfit for details.

A **Deep Learning** model node can be added to the workflow in one of the following ways:

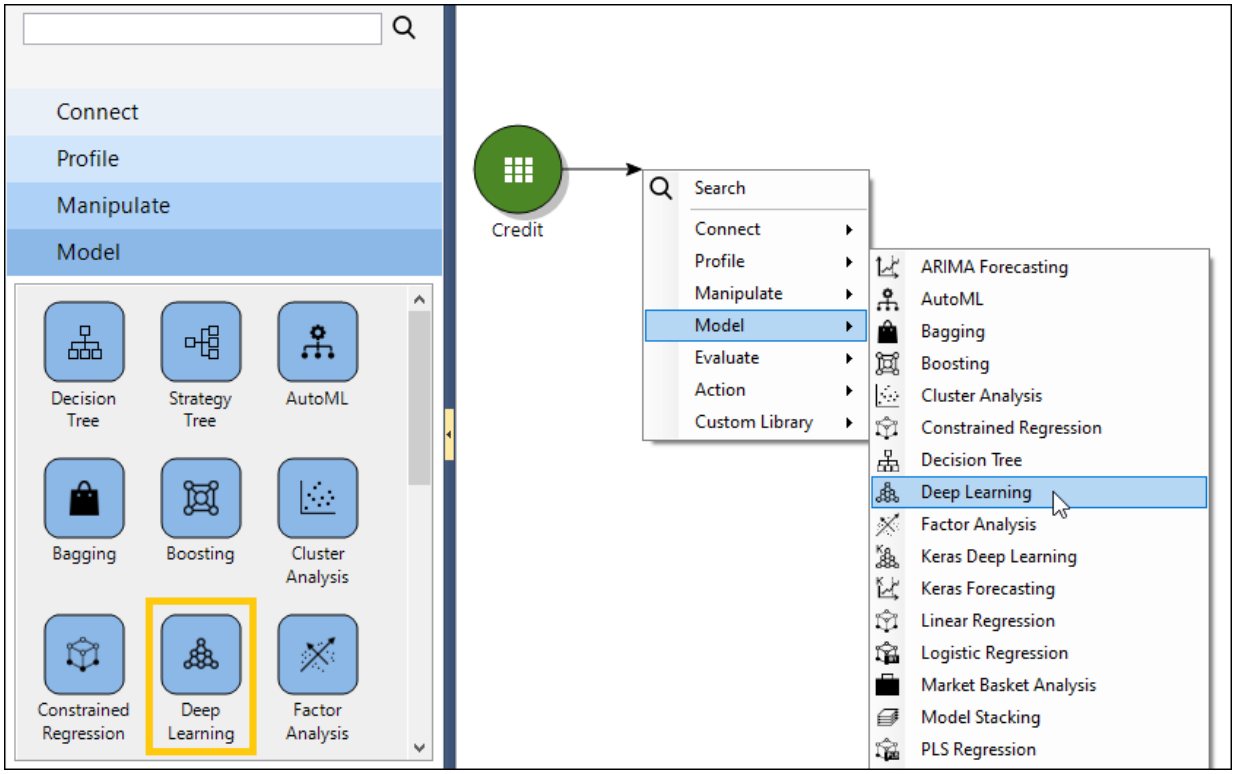
- Drag a connection arrow from a dataset node to any area on the Workflow canvas background. From the context menu that appears when you release the mouse button, select **Model --> Deep Learning** as in the example screenshot below. A new **Deep Learning** process node will be created.

OR

- Right-click on the Workflow canvas and select **Model --> Deep Learning** from the context menu. The new Deep Learning node is gray until its inputs are defined. Connect a dataset to the model node.

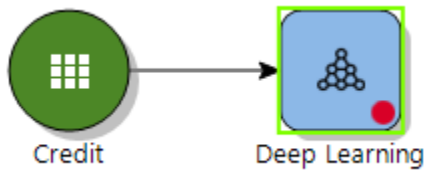
OR

- Open the **Model** palette group and drag the **Deep Learning** node onto the workflow canvas. The **Deep Learning** node is gray while its inputs are undefined. Connect the input dataset(s) to the **Deep Learning** node.

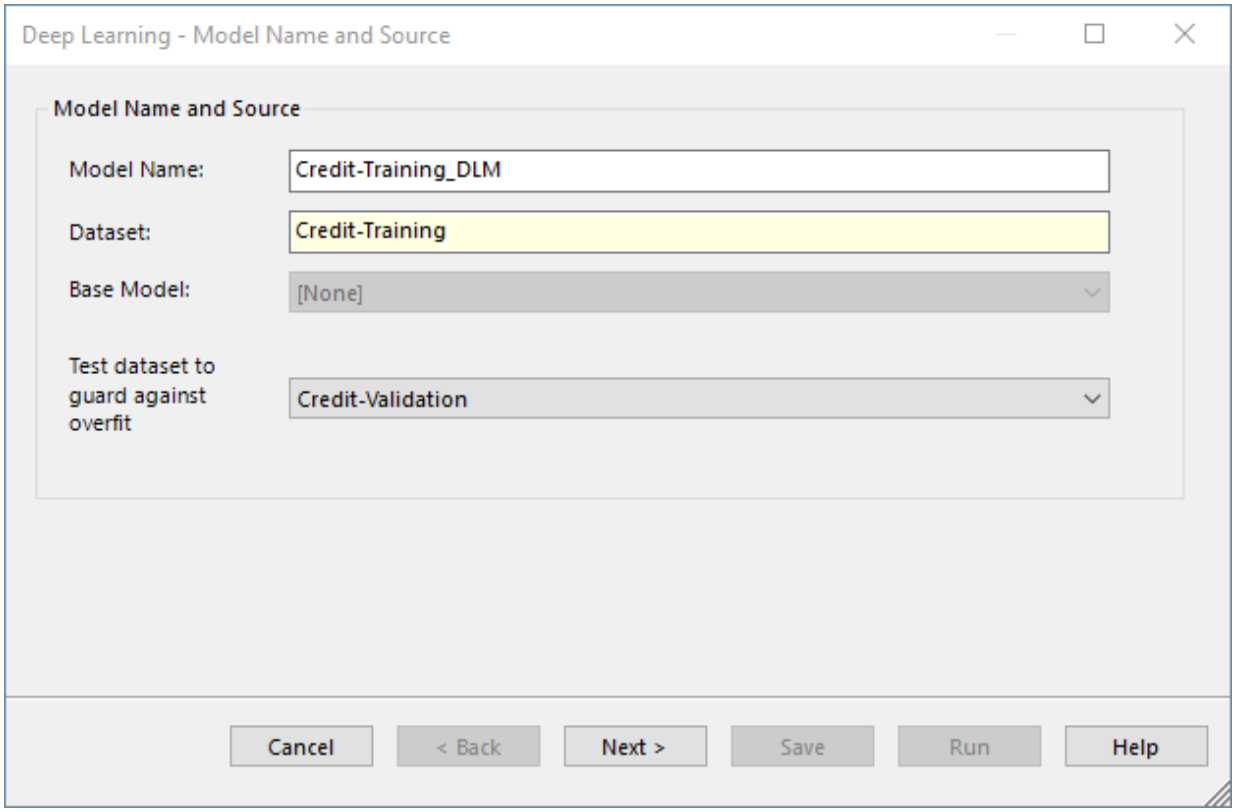


While it is not connected to other nodes, the model node is gray, as its inputs are undefined. Connect the input dataset to the model node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Double-click on the Deep Learning node added to the canvas, or right-click on it and select **Modify** from the context menu.



Model Name The name of the model that will be produced when running the model node. The model object with this name will appear in the Project Explorer after running the node.

Dataset The name of the input dataset for the model.

Base Model The model to use as a base model for the new predictive model. This requires connecting the intended base model to the current model node being. In this case the current model will have two inputs: a dataset and another model. All applicable training parameters and attributes of the template model will be copied and used for the new model. For a template, you can select any other model node in the workflow. This function is convenient for retraining models and inserting new models that inherit some or all of the model parameters of an existing model (at least the set of independent variables).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Test data to guard against ove

If necessary, select a test dataset to guard against model overfit from the list of datasets in the project.

Overfitting occurs when a model is finding patterns in the training dataset that may not occur in other datasets. Overfitting is dangerous because it can lead to predictions that are far beyond the range of the training data. This problem of neural networks is also known as or memorizing. Overfit networks model the patterns of the training dataset very well, but are not trained to be general-purpose models.

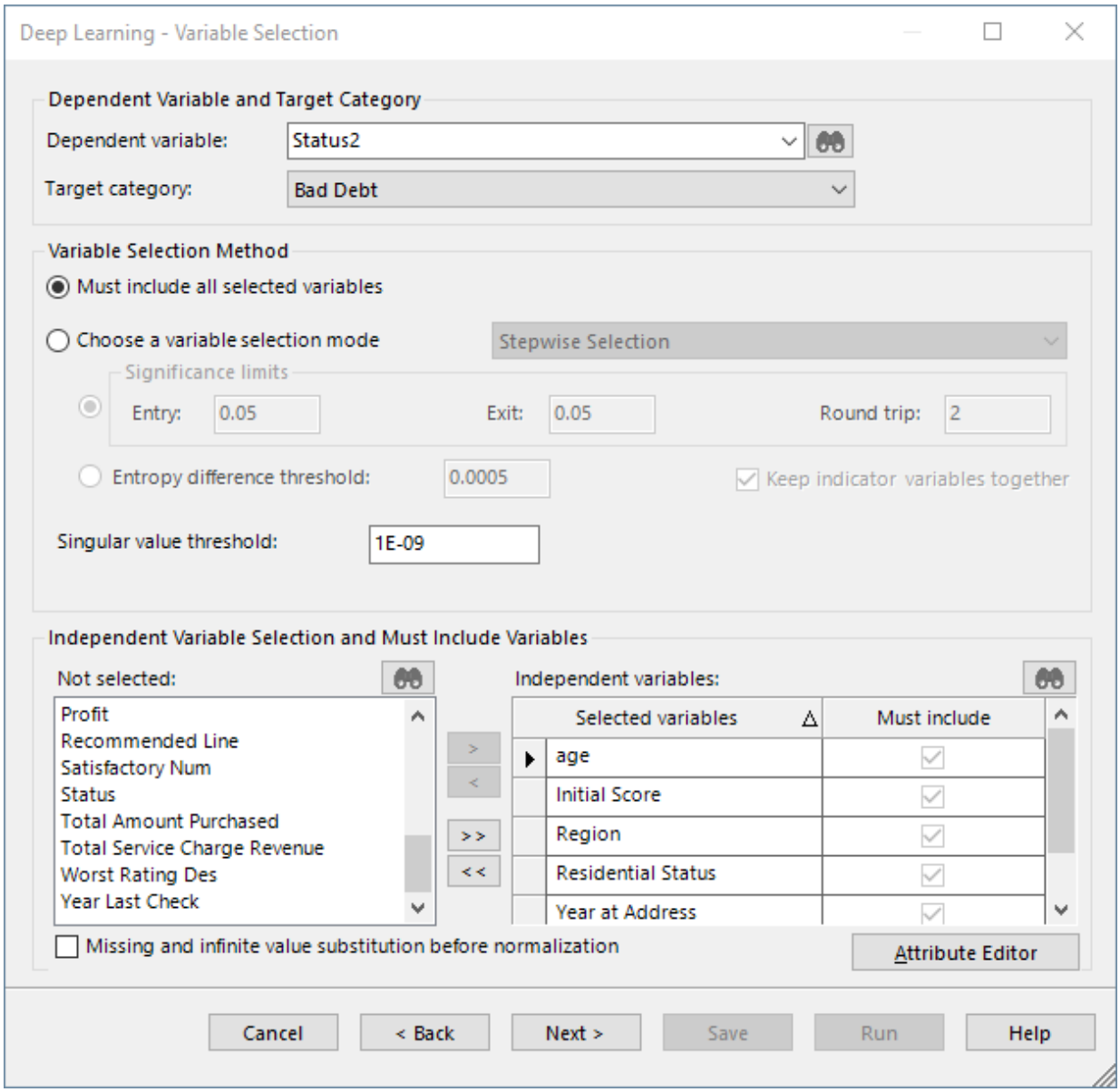
To protect against overfitting, the model is trained on one dataset partition and validated on another. Periodically, the neural net stops learning and tries to predict the records kept aside in the testing partition. If testing accuracy degrades too much, the training process is terminated. This can be controlled by the parameter "Number of degraded results to terminate" in the last page of the wizard.

Click Next to select the variables and specify variable selection options.

Variable Selection

In the Variable Selection page, select the dependent variable (DV) and the independent variables (IVs) you want to use in your model.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Dependent Variable	The variable of interest for the analysis. The drop down list contains the available variables. Both discrete and continuous types of variables can be selected as DV.
Target Category	If the selected dependent variable is discrete, for the Target Category, select the value of the dependent variable whose probability the model should try to predict. Not appliable if the selected dependent variable is continuous.

Note: The DV can also be changed in the Predictive Model Attribute Editor. Click the Attribute Editor button to open the attribute editor and set the Role to "dependent" for the desired variable.

Must include all selected variables

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If this method is checked, then all the variables selected in the previous wizard step will be used when training the model, regardless of their significance.

Choose a variable selection method

Stepwise	Include variables by considering the Entry and Exit Significance Levels.
Forward Selection	Starts with no variables and then includes variables based on the rank order of variables from the best to the worst. The rank order is determined by a log-likelihood ratio test.
Backward Selection	Starts with all the variables and then excludes variables based on the rank order of variables from the worst to the best. The rank order is determined by a log-likelihood ratio test.
R-Squared Selection	Check through all possible combinations of coefficients and produce a ranked list with r-square increasing as various combinations go in.
Variable Sequence My Choice	Include variables based on the user's custom selection. The variables can be chosen on the basis of their significance (the P-value computed for each variable). See Variable Sequence Selection for details.

Significance Limits

Entry	The entry significance level determines when a variable is qualified to be included in the model. This parameter is used for the **Stepwise** selection method.
Exit	The exit significance level determines when a variable is disqualified and excluded from the model. This parameter is used for the **Stepwise** selection method.
Round Trip	The number of entry-exit-entry-exit sequences before a variable is disqualified and excluded from the model.

Singular Value Threshold

Used to adjust the collinearity detection sensitivity. The threshold determines when any two independent variables are considered to be collinear and one of them must be ignored (removed from the model) to avoid the risk of biasing the model.

If there is an exact linear dependency between any variables, the corresponding singular value is equal to zero. A small singular value (eigenvalue) indicates collinearity issues, so when small singular values occur, this threshold determines how small the singular values may be without causing any problems to the least squares estimation.

The default value for this threshold is 1.0E-09.

Keep indicator variables together

When this option is selected, the categories of the discrete variable are treated as a group. When the discrete variable is included or excluded from the model all the categories within the variable are included or excluded as a group. For example, ***marital-status*** contains seven categories: **Single**, **Married**, etc. If ***marital-status*** is excluded, all the categories are excluded from the model. When this option is unchecked, each category is evaluated separately and will be included or excluded individually.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Independent Variable Selection and Must-Include Variables

Variables on the left are not included in the model. Variables listed on the right will be included. You can move the variables to and from the Independent Variables list by highlighting them and using the arrow buttons. Double-clicking the individual variables will also move them from one list to the other.

To select more than one variable in the lists, use mouse dragging, or mouse clicks while holding the SHIFT key or the CTRL key. If there are a lot of variables (more than a hundred, for example), it is convenient to use Field Finder (the binoculars button ).

If you chose a particular variable selection mode, select variables that must be forced into the model regardless of the selection method.

This option is only enabled if you have chosen a particular Variable Selection Method. During the variable selection process, the algorithm will exclude independent variables from the model if they do not affect the model. However, you may want to make sure that specific variables are kept in the model regardless of the algorithm. The variables that are in the **Must Include variables** list will always be included.

Select the variables(s) you want to always force into the model.

Attribute Editor

The Attribute Editor button brings up the predictive model attribute editor. The attribute editor allows you to modify variable attributes such as missing value substitution options, etc. For details, see the topic Model Attribute Editor.

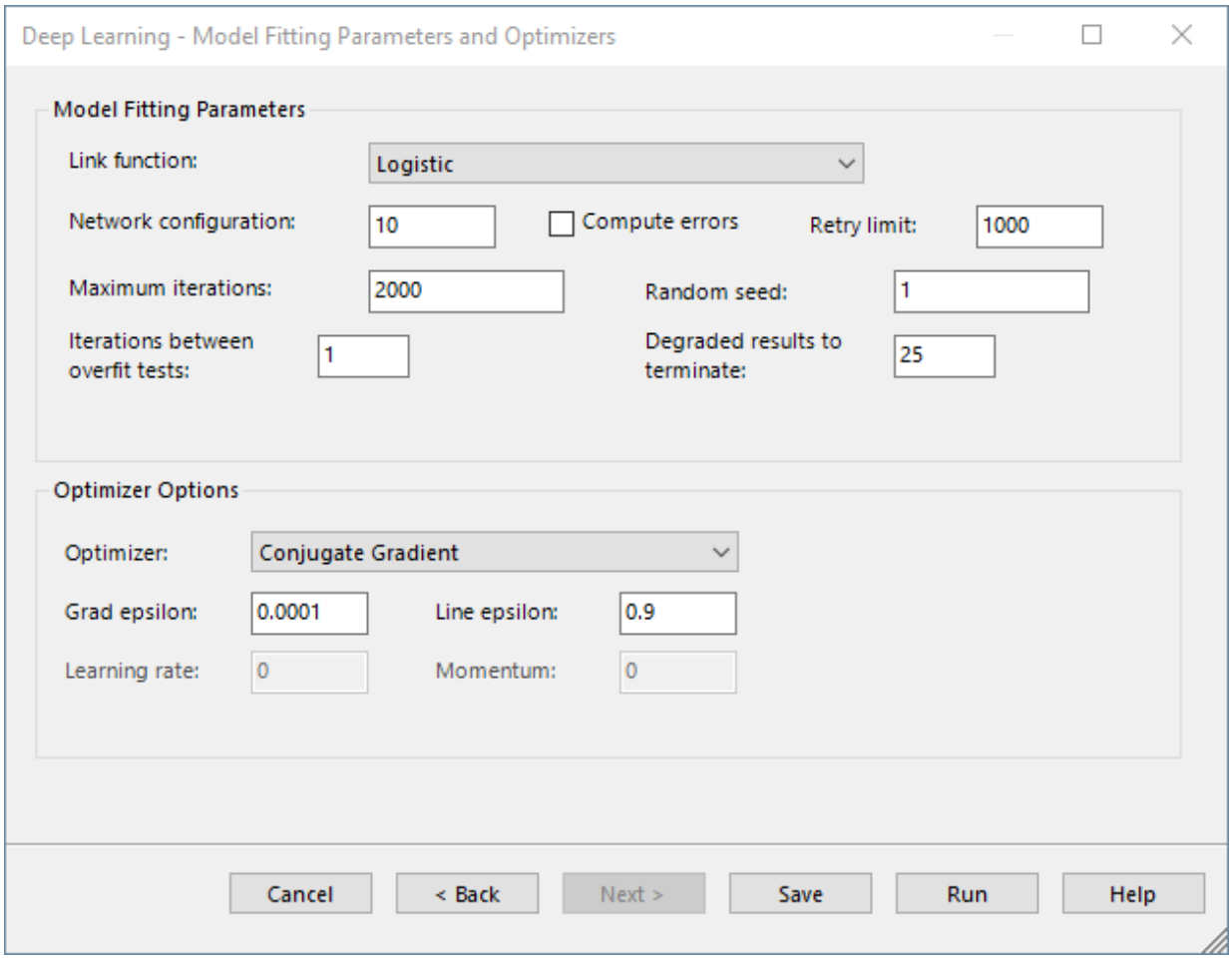
Substitution and Normalization Order

Variables included in the model are normalized according to their Usage defined in the Attribute Editor. The automatic substitution of missing and infinite values (if any) is also defined there. By default, normalization is performed **before** the substitution of missing and infinite values. If any of the variables to be included in your model contain missing and/or infinite values, and if you prefer to substitute them before normalization, select the option "**Missing and infinite value substitution before normalization**" in this wizard page. Alternatively, you can use the Variable Transformations node to perform the necessary substitutions before building the model.

Click **Next** to specify the model fitting parameters.

Model Fitting Parameters and Optimizers

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Link function The link function specifies a nonlinear transformation of the predicted values so that the distribution of predicted values is one of several special members of the exponential family of distributions (e.g., gamma, Possion, binomial, etc.). The link function is used to model responses when a dependent variable is assumed to be nonlinearly related to the predictors. The available link functions are: ***Logistic*** (most commonly used, selected by default), ***Inverse Cumulative Normal, Complementary Log-Log***, and ***Log-Log***.

Network Configuration This option allows you to specify the number of hidden layers in the network. You can enter a series of integers separated with a "/". Each integer represents a hidden layer and the number of neurons for that layer. For example, if you entered the following into the Network Configuration: 4/ 6/ 8, this would indicate 3 hidden layers where Hidden Layer #1 would use 4 neurons, Hidden Layer #2 would use 6 neurons and Hidden Layer #3 would use 8 neurons for the calculations.

Compute Errors If this option is selected, the ***Score test*** and ***Wald's test*** will be performed when testing the Global Null Hypothesis in the model training process. The corresponding Score and Wald statistics will be computed and shown in the Sequence view of the model results report. This option is not selected by default.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Retry Limit	The number of times to try to retrain the model if it fails to converge to an acceptable solution before reaching the Maximum Iterations limit. The default is 1000. The model is initialized randomly (based on the *random seed* value specified in the same page). After the model converges on a solution or reaches the Maximum Iterations limit, the value of Variance Explained is calculated. If it is positive, the model solution is accepted. Otherwise, the training process is repeated with another random initial model. The Retry Limit value determines how many times it is repeated. Reduce this limit if the model training takes too much time. For example, if it takes several hours and results in the message "No significant model found", or if it appears to run indefinitely, change the Retry Limit to 10 or another value less than 1000, depending on how much time the training takes.
Maximum Iteratio	The number of times the model calculates the error, adjusts the weights, and feeds the error back through the model.
Random seed	The model is initialized randomly based on the random seed value. The option to specify the random seed value is provided for repeatability of results.
# Iteratio between ove tests	Number of iterations before each overfit test. Although the safest approach is to test the neural network after every iteration, this may take a lot of time. This option allows you to skip a number of iterations before testing for overfit again.
# Degraded resu to terminate	Allows you to control how long the model continues to train as the results continue to degrade before terminating training and rolling back to the model with the best results. The default number is 25.

Model Optimizers

Choose the optimizer and specify the desired parameters.

***Congugate Gradient* (*default*)**	The Conjugate Gradient method is the default optimizer. For larger datasets with more than a few thousand weights, Conjugate Gradient is faster than BFGS, and uses less memory.
BFGS	BFGS stands for the Broyden–Fletcher–Goldfarb–Shanno method. For the datasets with few variables, BFGS is usually faster than Conjugate Gradient. However, for networks with more than a few thousand weights, BFGS requires a lot of storage and could lead to a shortage in memory.
Classic Ba propagation	The Classic Back-propagation, which is a gradient descent algorithm, tries to adjust the weights based on the previous weights. At each step, it iteratively updates the weights by moving a short distance in the direction of the greatest rate of decrease of the error. It is not as accurate as the other two algorithms.

Optimizer Parameters

Grad Epsilon	Converge epsilon for gradient.
Line Epsilon	Similar to Grad Epsilon, but used by line optimizer.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

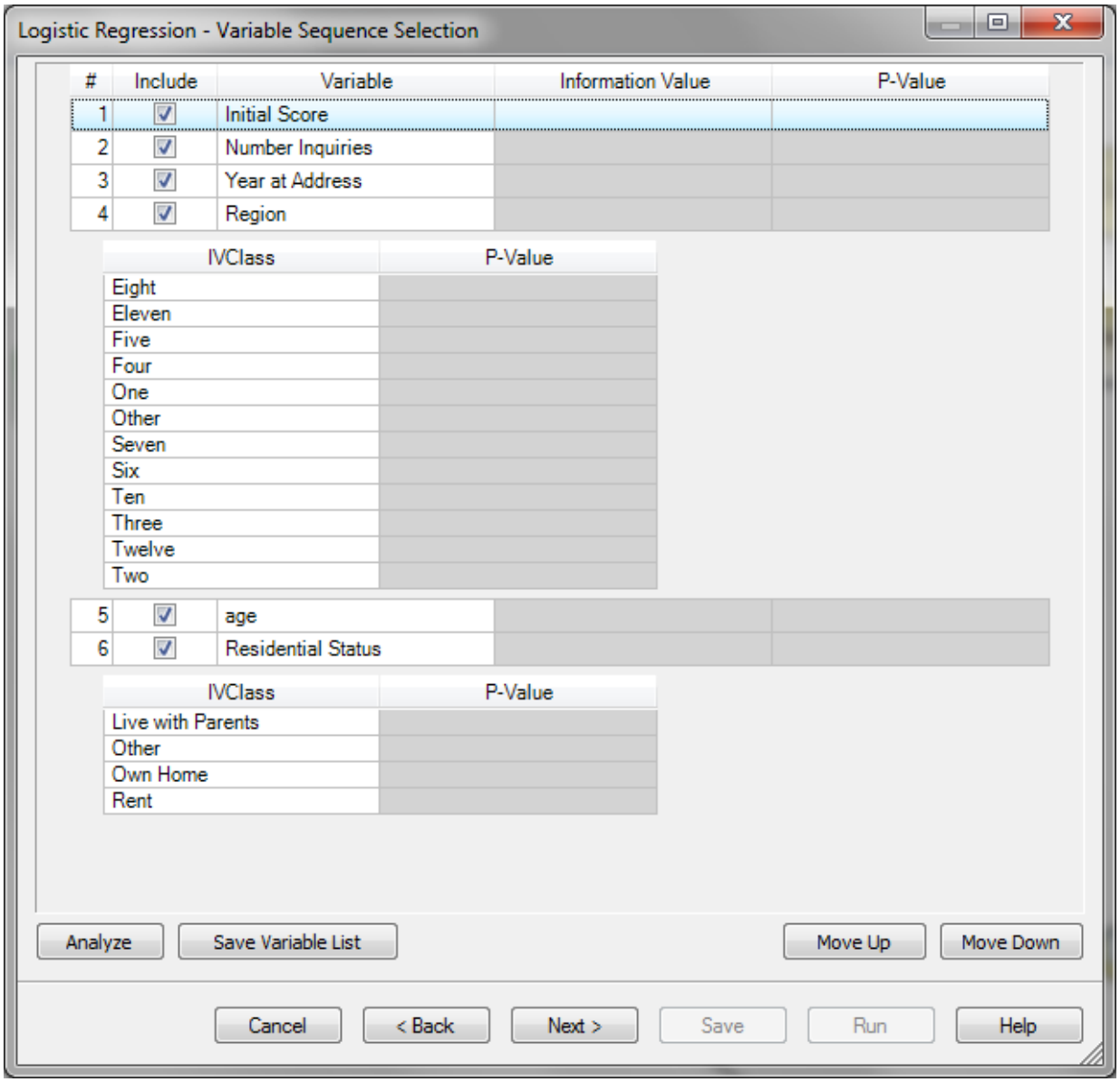
Learning Ra *and* ***Momentu***	Optimizer parameters applicable to the Classic Back-Propagation optimizer only

Custom Sequencing

Custom Sequencing is applicable to Regression models and Deep Learning models.

Selecting **Variable Sequence Of My Choice** as the variable selection mode allows you to define a custom order of including variables in the model. The variables can be chosen on the basis of their significance (the *p*-value computed for each variable) and Information Value. For details on these and otehr statistics, see Mathematical Formulations: Decision Tree.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



More precisely, the model under 'Variable Sequence Of My Choice' starts with no variables and then includes variables based on their order in the list of variables in this page of the wizard. The buttons "**Move Up**" and "**Move Down**" allow you to change the order of inclusion of variables in the sequence. It starts with the topmost variable in the list, then adds subsequent variables one by one until it adds all of them, and finally chooses the best sequence as the "Currently Selected Sequence".

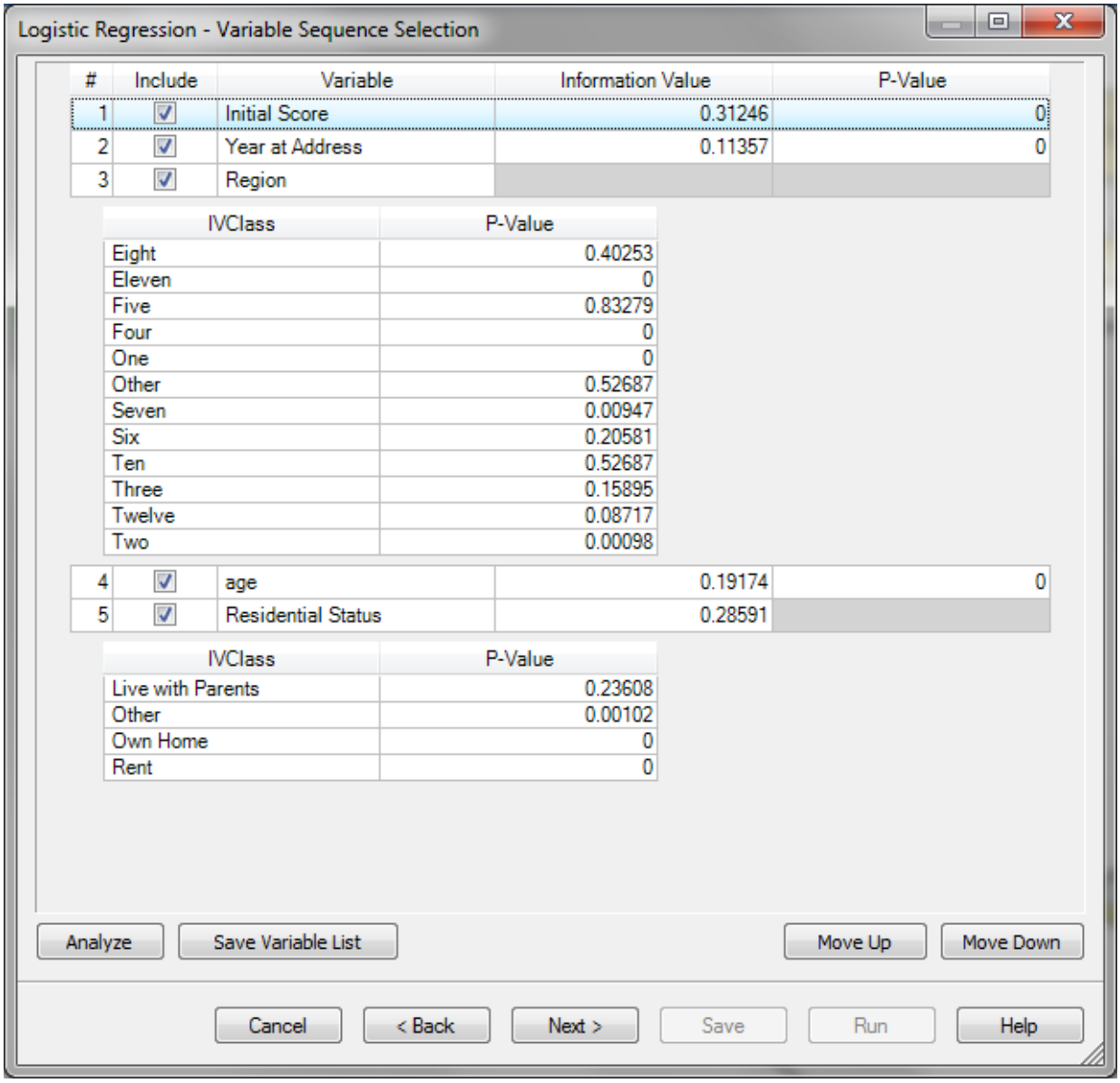
The details in the statistics for each sequence can be found in the Results tab of the model when it is trained (open the drop-down list "Output To View" at the top of the Results tab).

The significance for each variable is computed by building a single-independent-variable regression model with this variable and then computing the P-value.

Click the **Analyze** to compute the *p*-value and Information Value in this page.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Based on the computed parameters, decide which variables to include and move up or down.



Keras Deep Learning

The **Keras Deep Learning** node in the **Model** palette of Knowledge Studio provides an interface to the *Keras* library via integration with Python. Keras is a Python Deep Learning library designed to enable fast experimentation with deep neural networks. For details see Keras documentation at https://keras.io/.

The **Keras Deep Learning** node wizard allows you to specify the necessary parameters and define the neural network configuration for Keras using graphical user interface. The wizard automatically translates the inputs into Python code that can be previewed before running the node. It also converts the input Altair KDD dataset(s) to Keras-readable format.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The model performance can be validated against the test data if any test data is selected. After training, the model can be scored on new data or validated using the Scoring node or Model Validation node.

Building a Keras Deep Learning Model

A Keras Deep Learning model requires one input dataset for model training. It can also use an optional second input dataset to guard the model against overfit. Training and test datasets can be created, for example, using the Partition node. The data types of the variables in the training and test datasets must match.

A **Keras Deep Learning** node can be added to the workflow in one of the following ways:

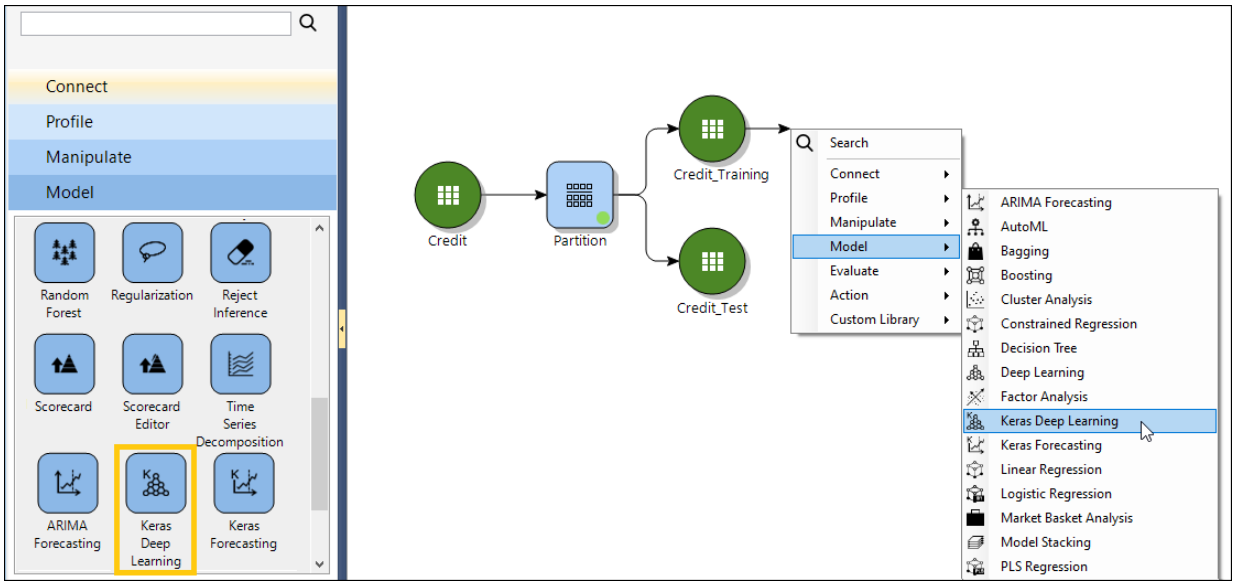
- Drag a connection arrow from the training dataset node to any area on the Workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Model --> Keras Deep Learning** as in the example screenshot below. A new **Keras Deep Learning** node will be created. Optionally, connect a test dataset to the new Keras Deep Learning node.

OR

- Right-click on the Workflow canvas and select **Model --> Keras Deep Learning** from the context menu. Connect the training dataset to the Keras Deep Learning node. Optionally, connect a test dataset to the same model node.

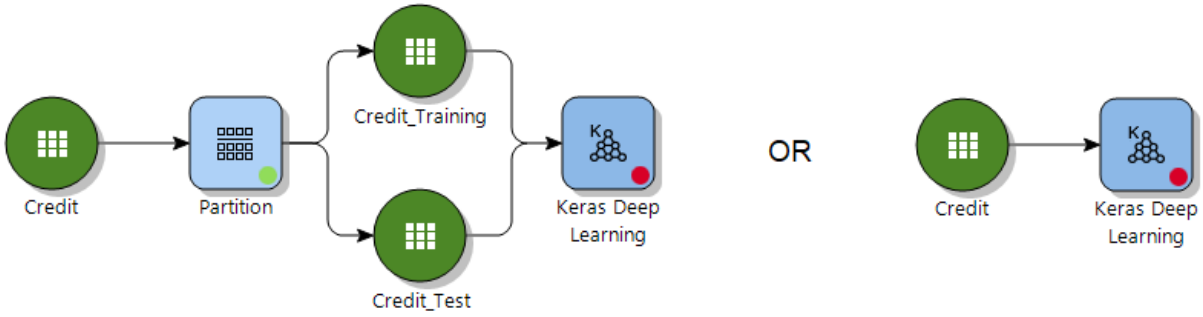
OR

- Open the **Model** palette and drag the **Keras Deep Learning** node onto the workflow canvas. Connect the training dataset to the Keras Deep Learning node. Optionally, connect a test dataset to the same model node.



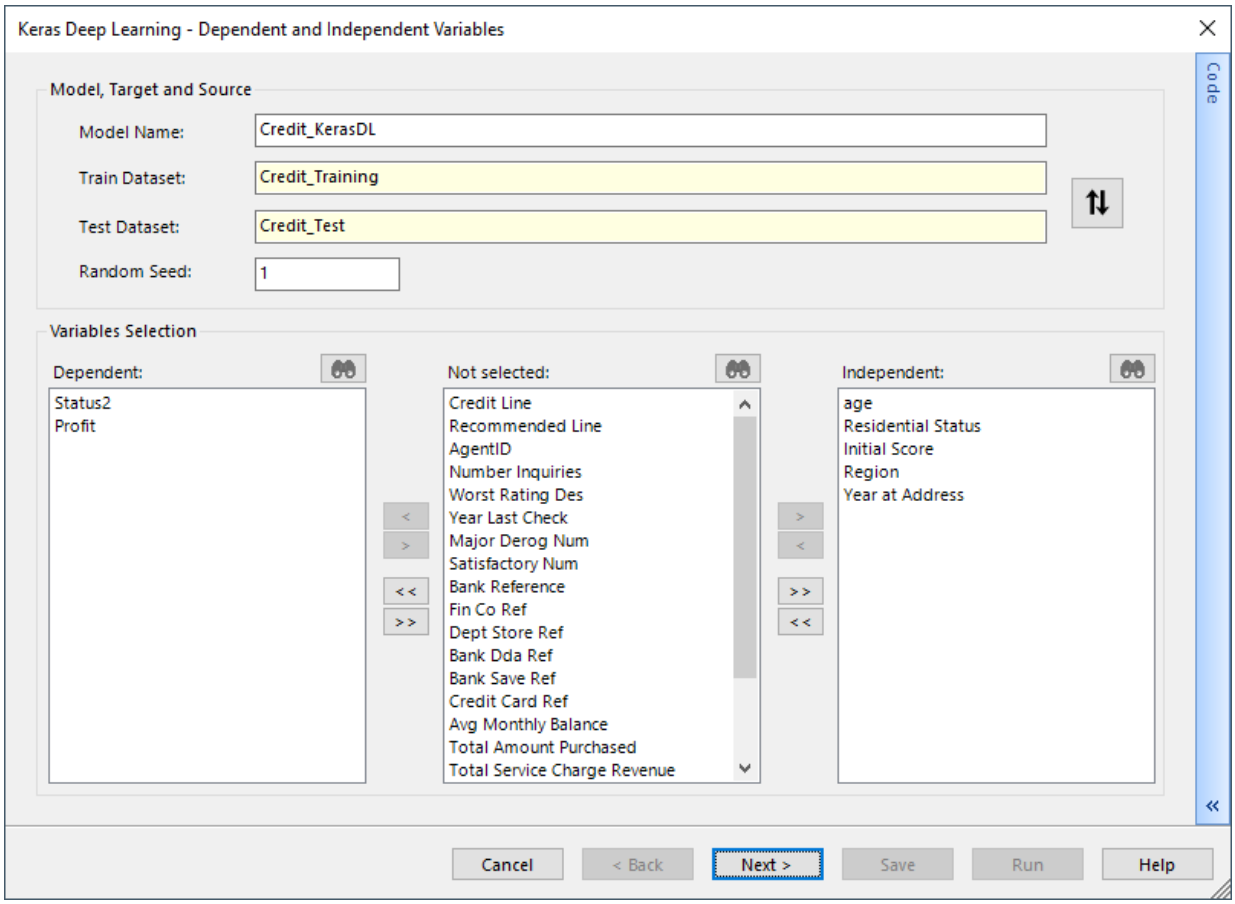
Double-click the Keras Deep Learning node or right-click on it and select **Modify** to open the wizard. This example shows two options for connecting the nodes:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



In the first page of the wizard:

- Specify the model name
- To change the order of the Train and Test input datasets (if applicable), click the arrows button on the right-hand side.
- Change the Random Seed value if required for repeatability.
- Select the dependent variable(s) using the arrow buttons to move them to the *Dependent* list on the left.
- Select the independent variables using the arrow buttons to move them to the *Independent* list on the right.



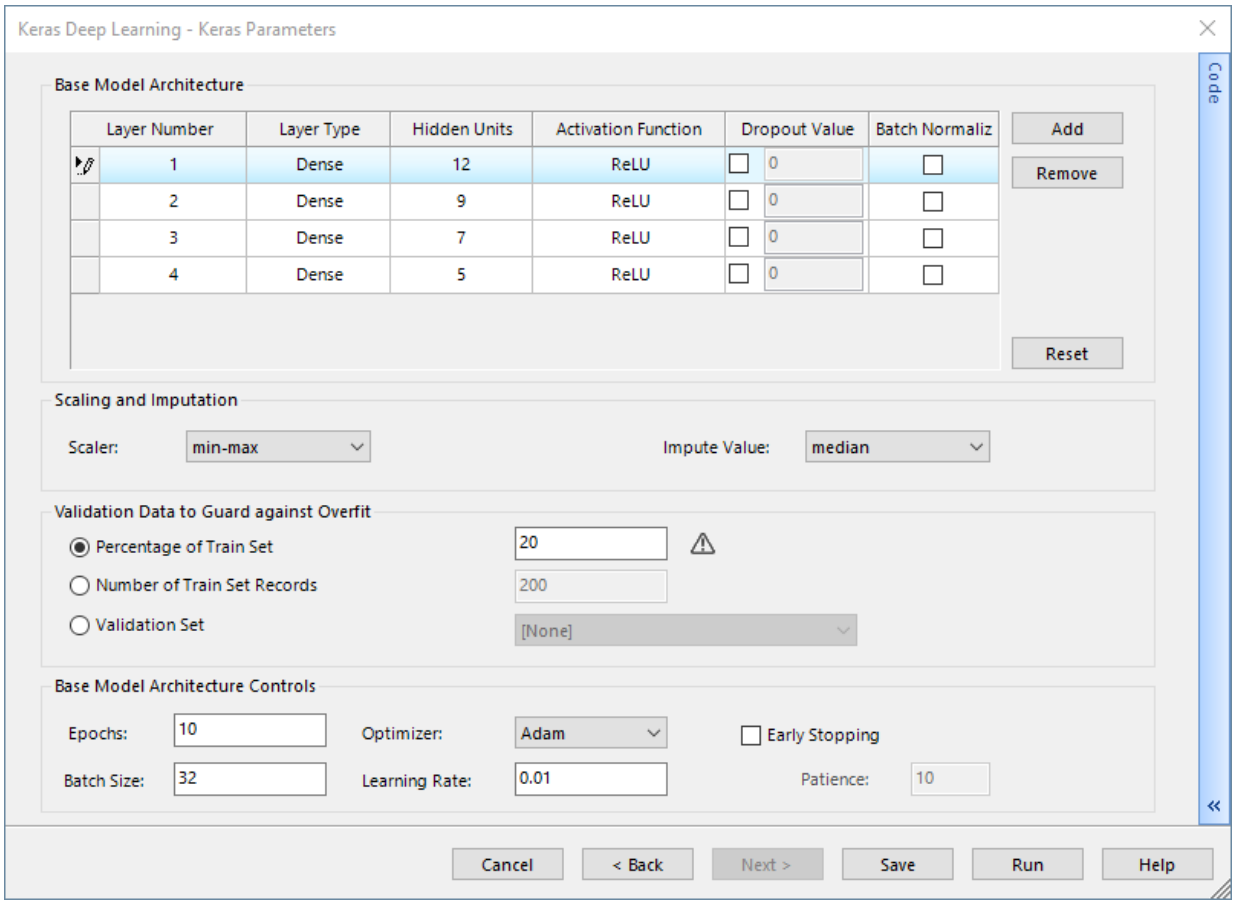
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

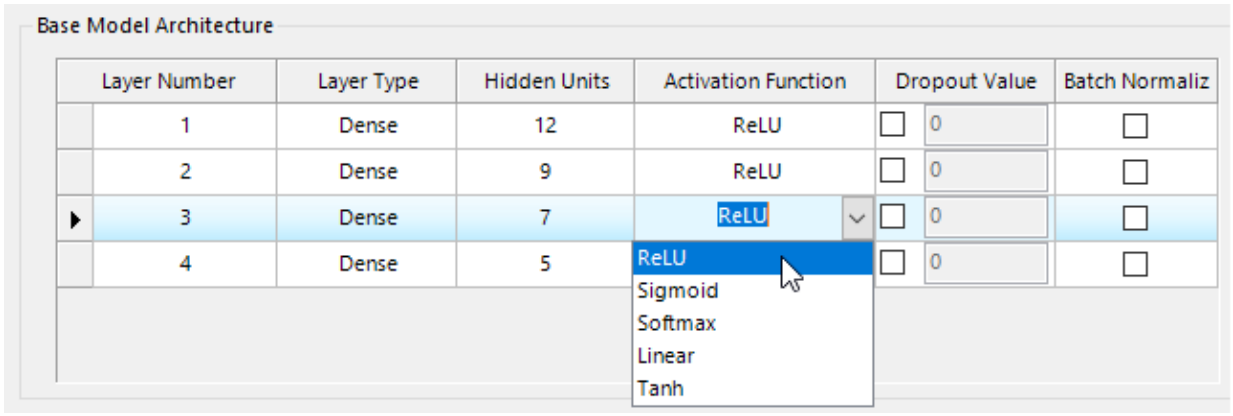
Click **Next**.

In the Keras Parameters page, configure the model and specify the following parameters as needed:

- Define the neural network configuration (Base Model Architecture) by adding layers and specifying their properties. To remove a layer, select it and click **Remove**. The **Reset** button removes all layers except the first one. The layer properties are:
  - ***Layer type*** (*Dense*, *LSTM*, *GRU*, or *Bidirec*)
  - The number of ***hidden units***
  - ***Activation function***. The available activation functions are *ReLU (rectified linear unit)*, *Sigmoid*, *Softmax*, *Linear*, and *Tanh*. Select the desired function from the drop-down list for each layer.
  - ***Dropout value***. Specify the dropout rate (a number between 0 and 1) - the frequency at which the input units are randomly set to 0.
  - ***Batch normalization***
- In the section *Validation Data to Guard against Overfit*, specify the portion of the training dataset to be allocated for testing to guard against overfit (percentage or the exact number of records). Note that this decreases the number of records for training. The default is 20% allocated for testing. Alternatively, select the Validation Set option to use the Test dataset (if connected to the model node).
- In the *Base Model Architecture Controls* section, specify:
  - Number of epochs (training iterations); the default is 10
  - Batch size; the default is 32
  - Optimizer. Available optimizers are *Adam, RMSprop,* and *SGD. T*he default is *Adam*.
  - Learning Rate. The default is 0.01.
  - Early Stopping. If this option is selected, specify the *Patience* parameter value (the default is 10).

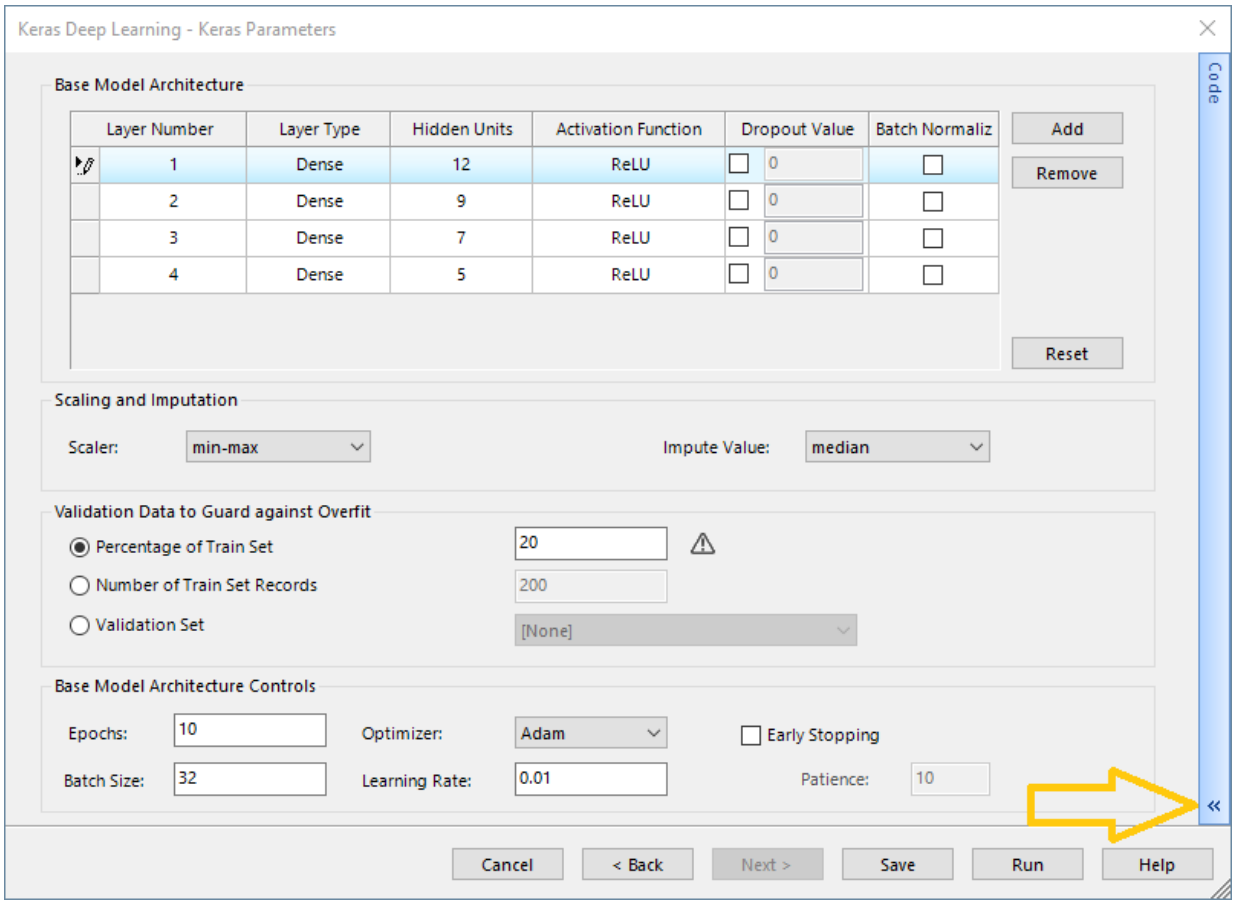
Altair Engineering, Inc. Copyright © 2022. All rights reserved.





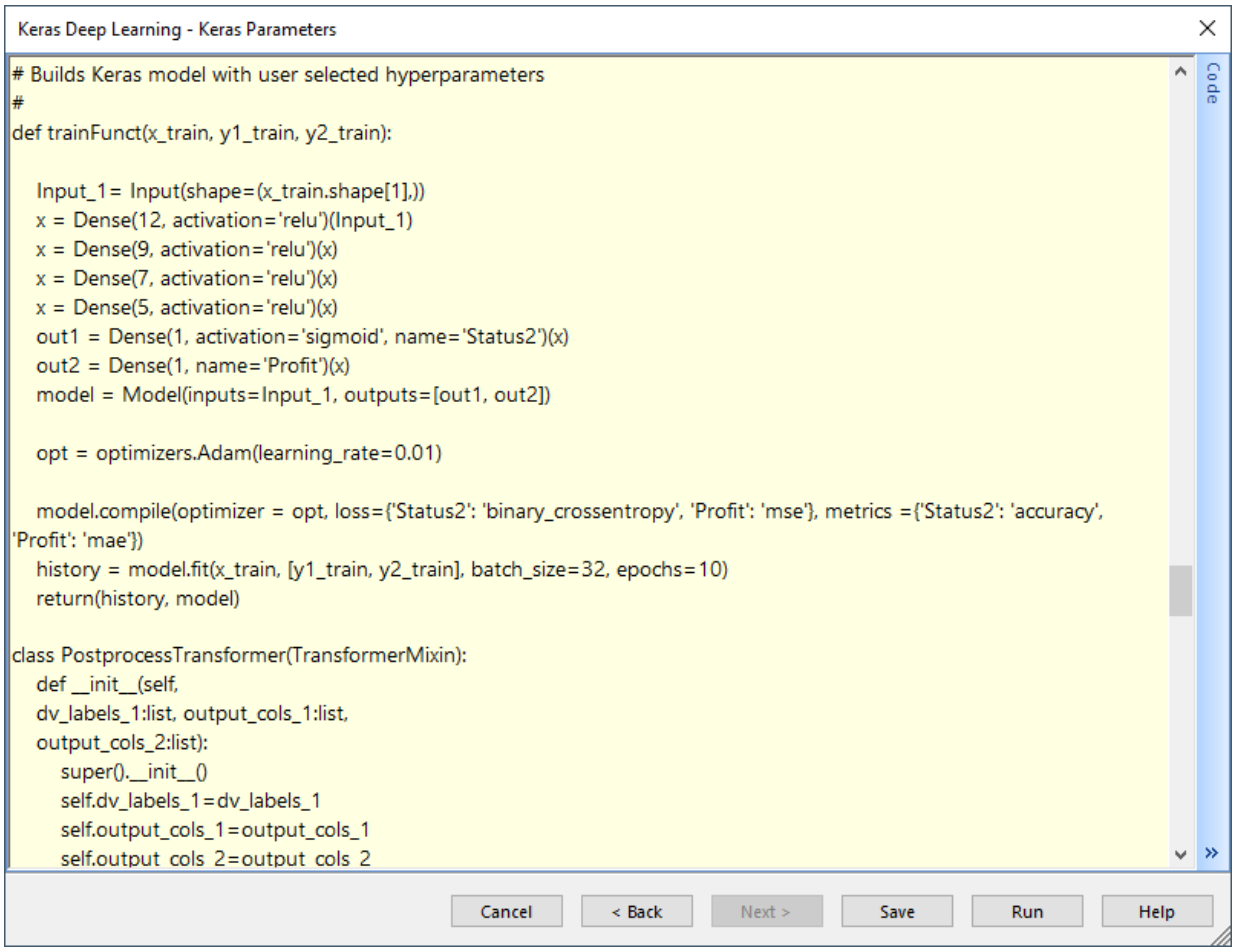
To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

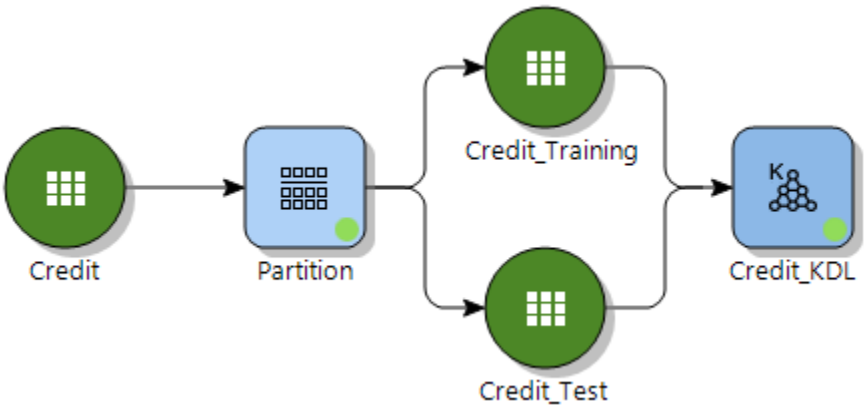


This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

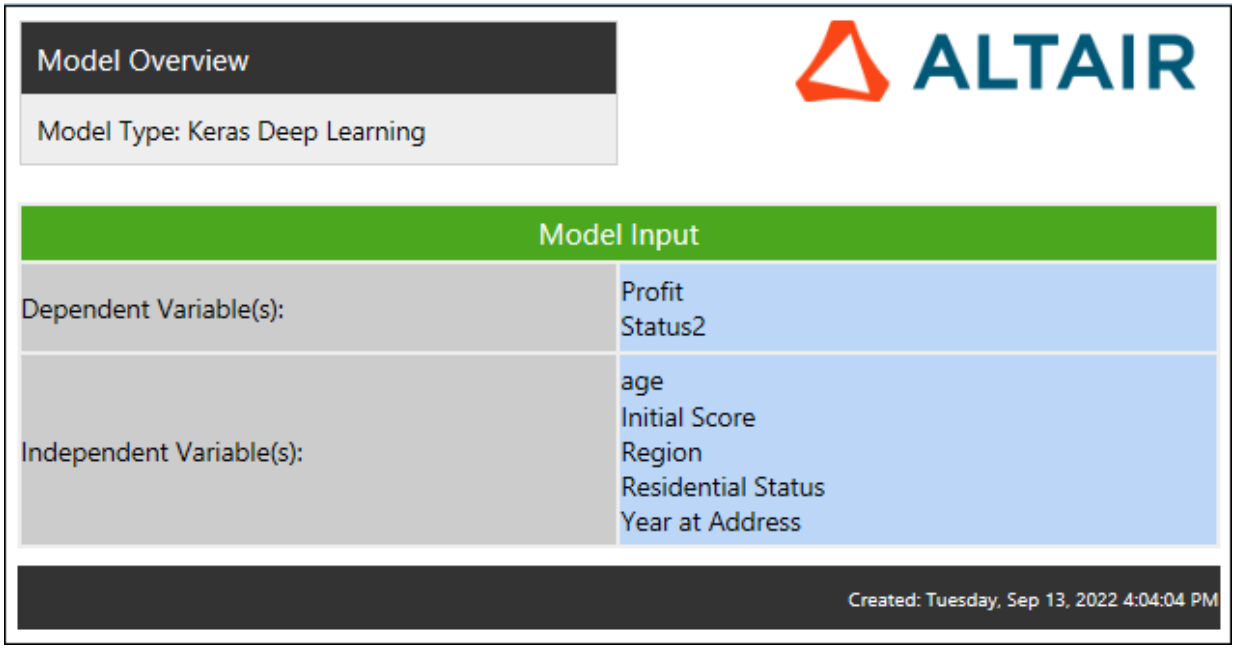


Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the node, double-click the model node to view the results.



Double-click on the Keras Deep Learning node to view the training results report and model performance plots. The **Results** tab shows the list of dependent and independent variables.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Model Overview

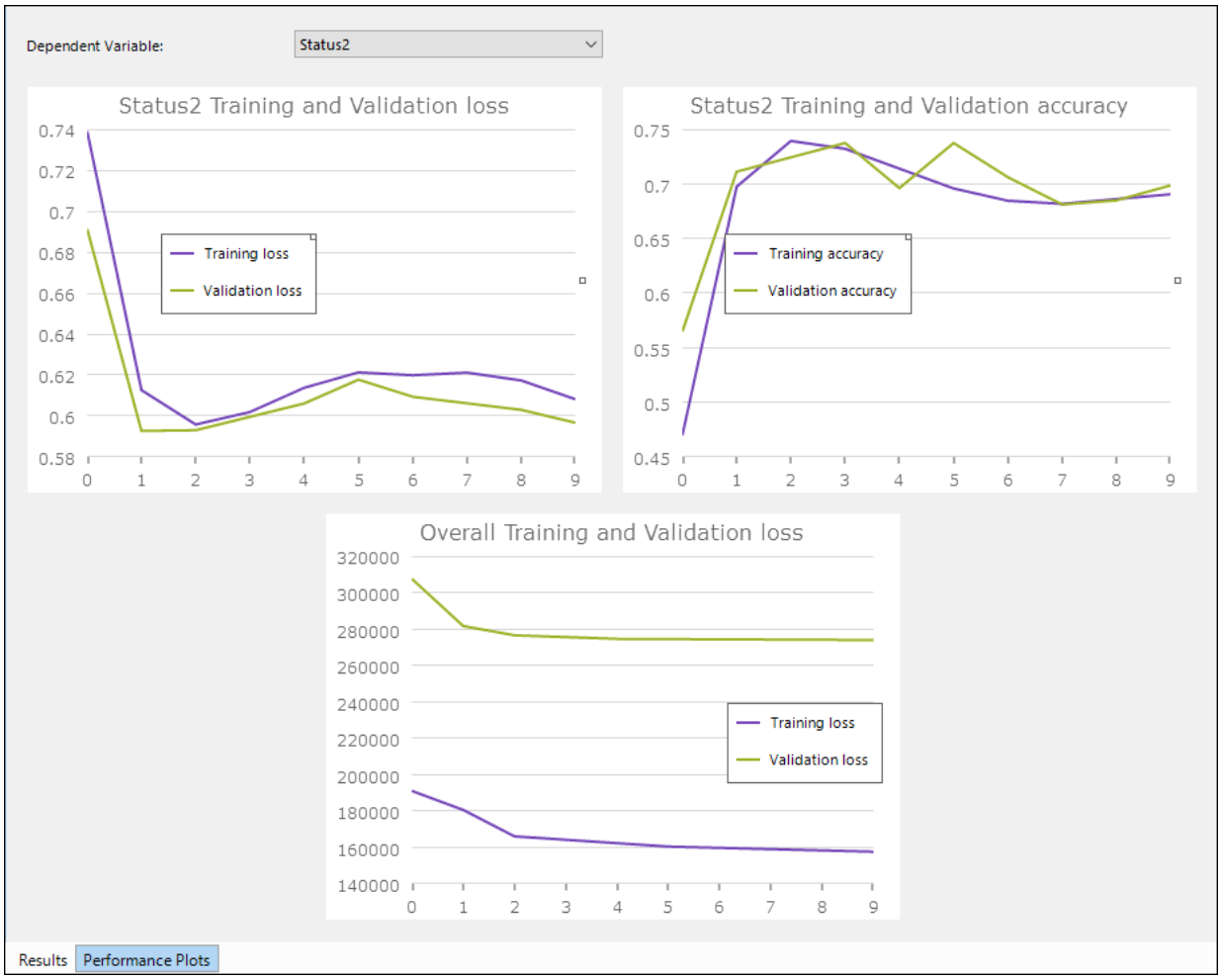
Model Type: Keras Deep Learning

Model Input	
Dependent Variable(s):	Profit Status2
Independent Variable(s):	age Initial Score Region Residential Status Year at Address

Created: Tuesday, Sep 13, 2022 4:04:04 PM

The **Performance Plots** tab shows the training loss and accuracy plots. Validation loss and accuracy plots are shown if any data was specified for *Validation Data to Guard against Overfit* in the Keras Parameters page. If you trained the model with multiple dependent variables, select the desired variable from the drop-down list at the top of the Performance Plots frame.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



After training, the model can be used for scoring new data or validating test data. See also the Scoring topic.

See also: Model Analyzer

Keras Deep Learning Scoring

A Keras Deep Learning model is scored using the **Scoring** node in the Action palette. For general information about scoring, see the topic About Scoring.

Prerequisites for Keras model scoring: a **Keras Deep Learning** node and an input dataset to score. The input Keras Deep Learning node stores the Keras Deep Learning in HDF5 format.

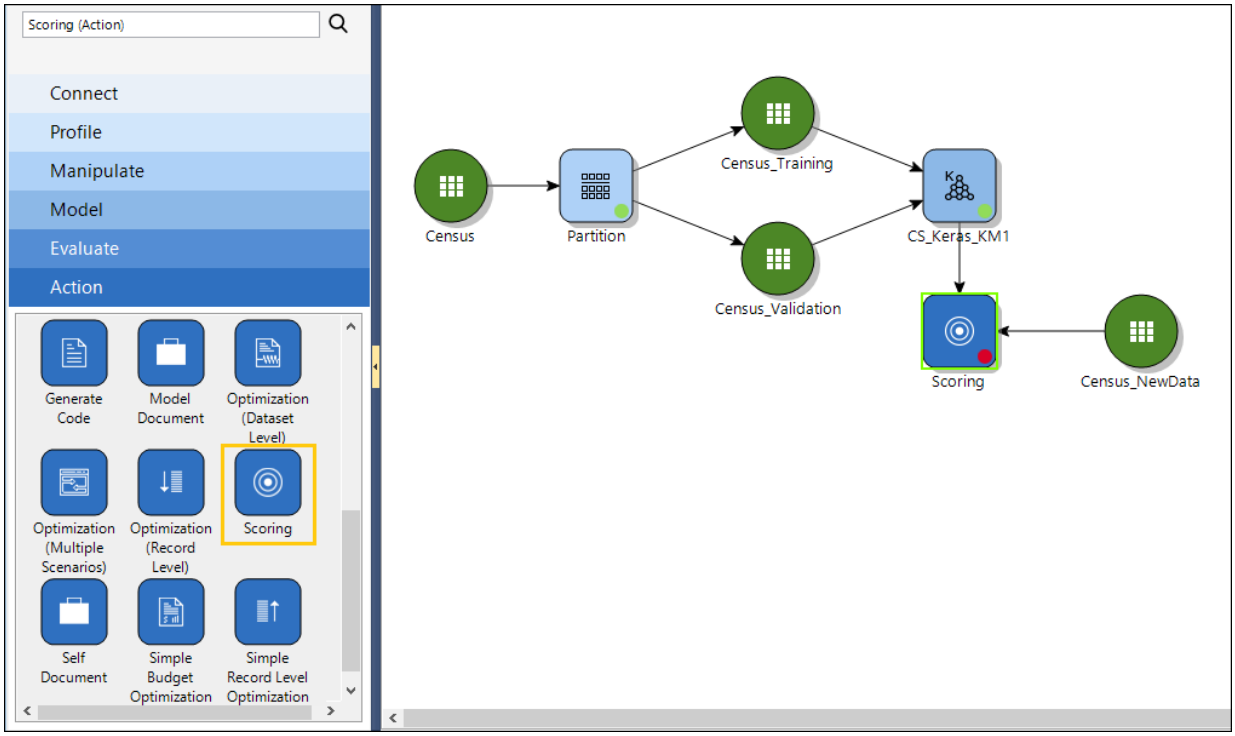
To add a Scoring operation to your workflow:

- Drag the **Scoring** node from the **Action** palette onto the Workflow canvas and connect the inputs: a **Keras Deep Learning** node and a dataset node that contains the data to score. See the example below.

OR

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Right-click anywhere on the workflow canvas and select the context menu command **Action --> Scoring**. This will create a new Scoring node. Connect the inputs: a Keras Deep Learning model node and a dataset node that contains the data to score.

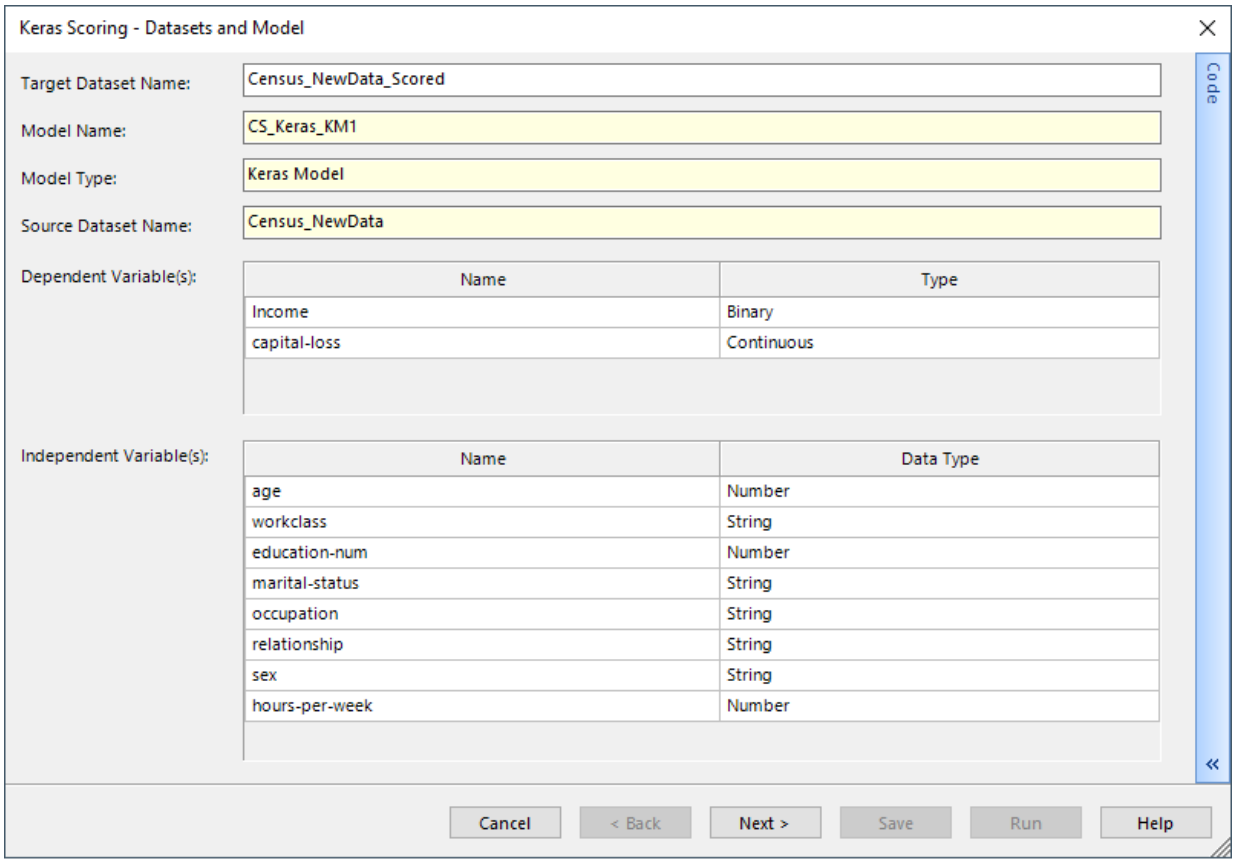


Double-click the **Scoring** node or right-click on it and select **Modify**.

The Scoring wizard opens. It displays the names of the input model and dataset, the list of dependent and independent variables and their data types.

In the *Target Dataset Name* field, specify the name for the output dataset that will contain the scores.

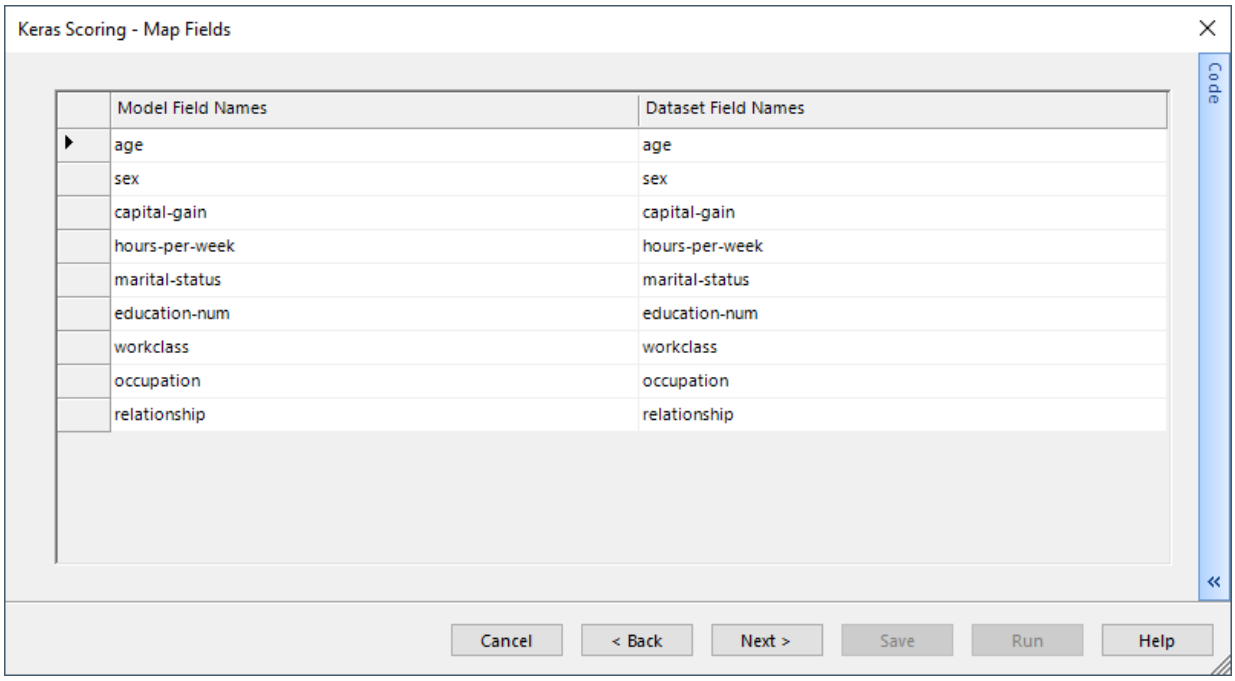
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

The next step provides the option of mapping the field names in the model to the ones in the scoring input dataset. In some situations, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map fields by clicking in the right column (Dataset Field Names) and selecting the right fields to match the model variables in the left column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

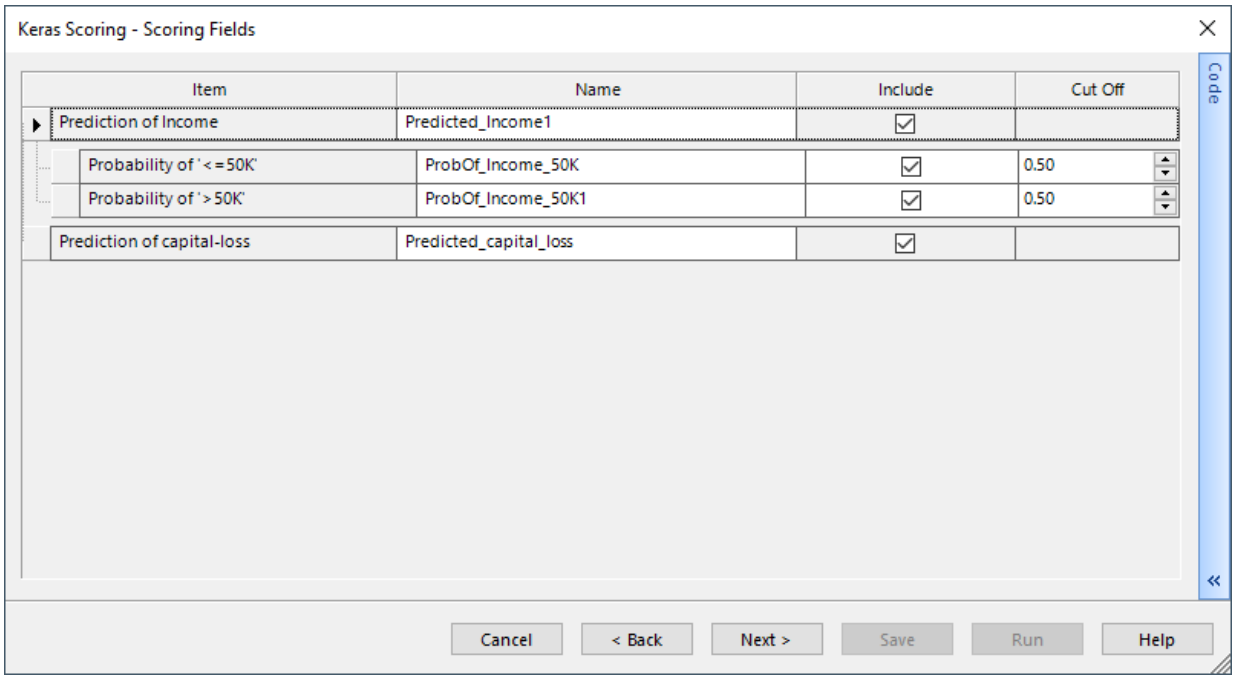


Note: When scoring Keras Deep Learning models that use derived variables (also called calculated columns) with expressions, the input dataset must include these variables. This is different from the case of native Altair models with derived variables, which can be scored on a dataset with only *original* variables that the derived ones depend on, while the derived ones are not required.

Click **Next**.

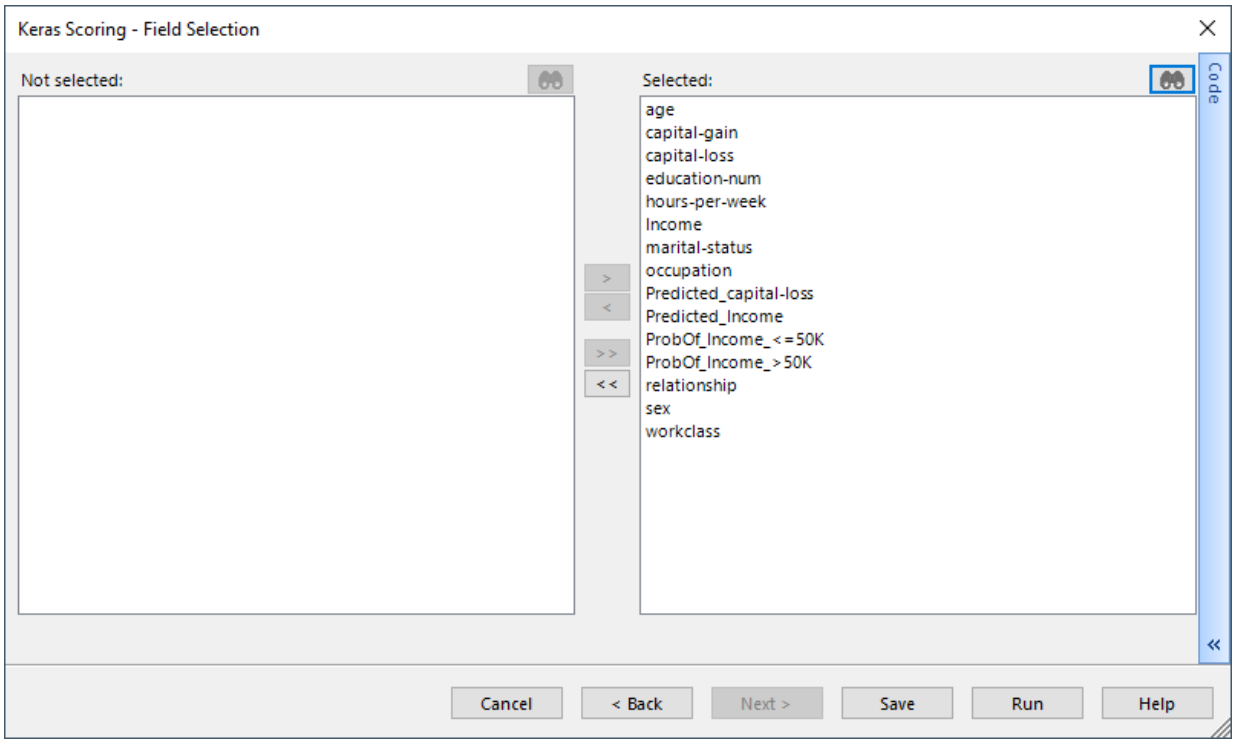
In the **Scoring Fields** page, select the scoring fields to be included in the scoring output dataset and modify their names if necessary. For discrete dependent variables, specify the probability cut-off values. See the general Scoring Wizard topic for more information on cut-off values.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



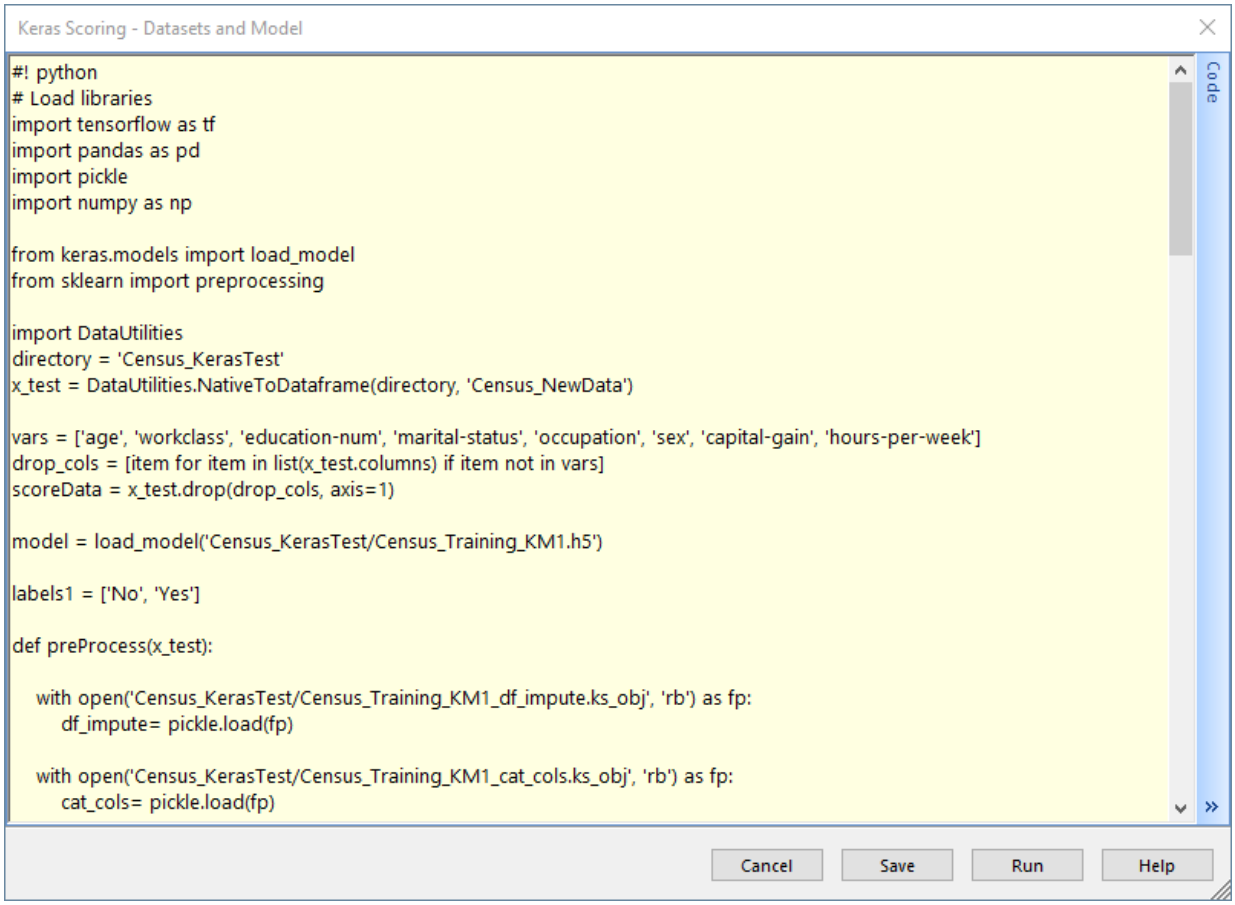
Click **Next**.

In the **Field Selection** page, choose the fields from the input dataset that you would like to include in the scoring output in addition to the new score / probability fields. By default, all fields in the scoring input dataset are selected to be included in the output.



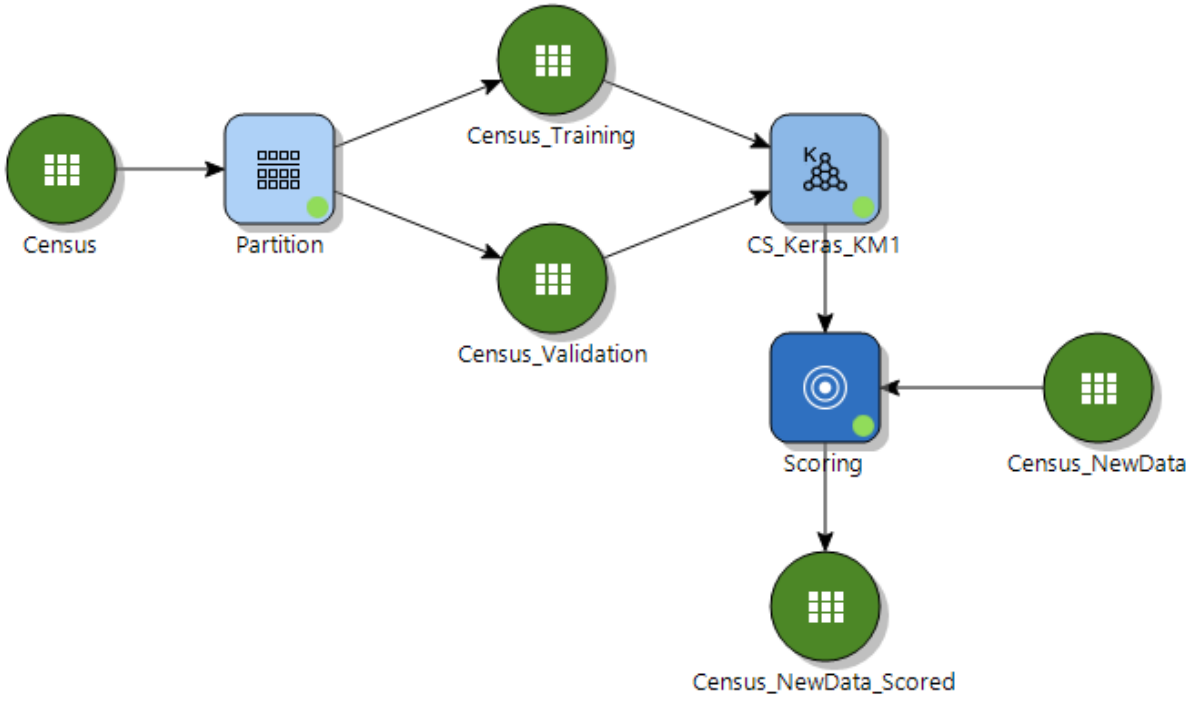
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.



Click **Save** to save the settings, or click **Run** to save the settings and run the scoring process immediately. The model is scored in a Python session with TensorFlow backend against the input data and the generated probabilities / scores are converted back from Pandas to Altair KDD format.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Double-click on the output dataset to view the scoring results.

- For discrete dependent variables, the output dataset contains the probabilities of the DV values and the prediction (the value that has the highest probability) for each record.
- For continuous dependent variables, the output dataset contains the predicted value for each record.

	Predicted_capital-loss	Predicted_Response	ProbOf_Response_No	ProbOf_Response_Yes
25	20.051362991333	No	0.818814516067505	0.181185454130173
26	113.580047607422	No	0.529841303825378	0.470158696174622
27	104.981788635254	No	0.542384803295135	0.457615196704865
28	223.476135253906	Yes	0.353432238101959	0.646567761898041
29	89.8815460205078	No	0.66495019197464	0.33504980802536
30	134.846542358398	Yes	0.478796482086182	0.521203517913818
31	165.005187988281	Yes	0.457728087902069	0.542271912097931
32	63.8199729919434	No	0.66535758972168	0.334642440080643
33	41.4115982055664	No	0.784708797931671	0.215291202068329
34	61.854076385498	No	0.626626670360565	0.373373329639435
35	121.861389160156	No	0.664023637771606	0.335976392030716

See also: Keras Deep Learning

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Keras Deep Learning Validation

A Keras Deep Learning model is validated using the **Model Validation** node in the **Evaluate** palette. For general information about validation, see the topic About Validation.

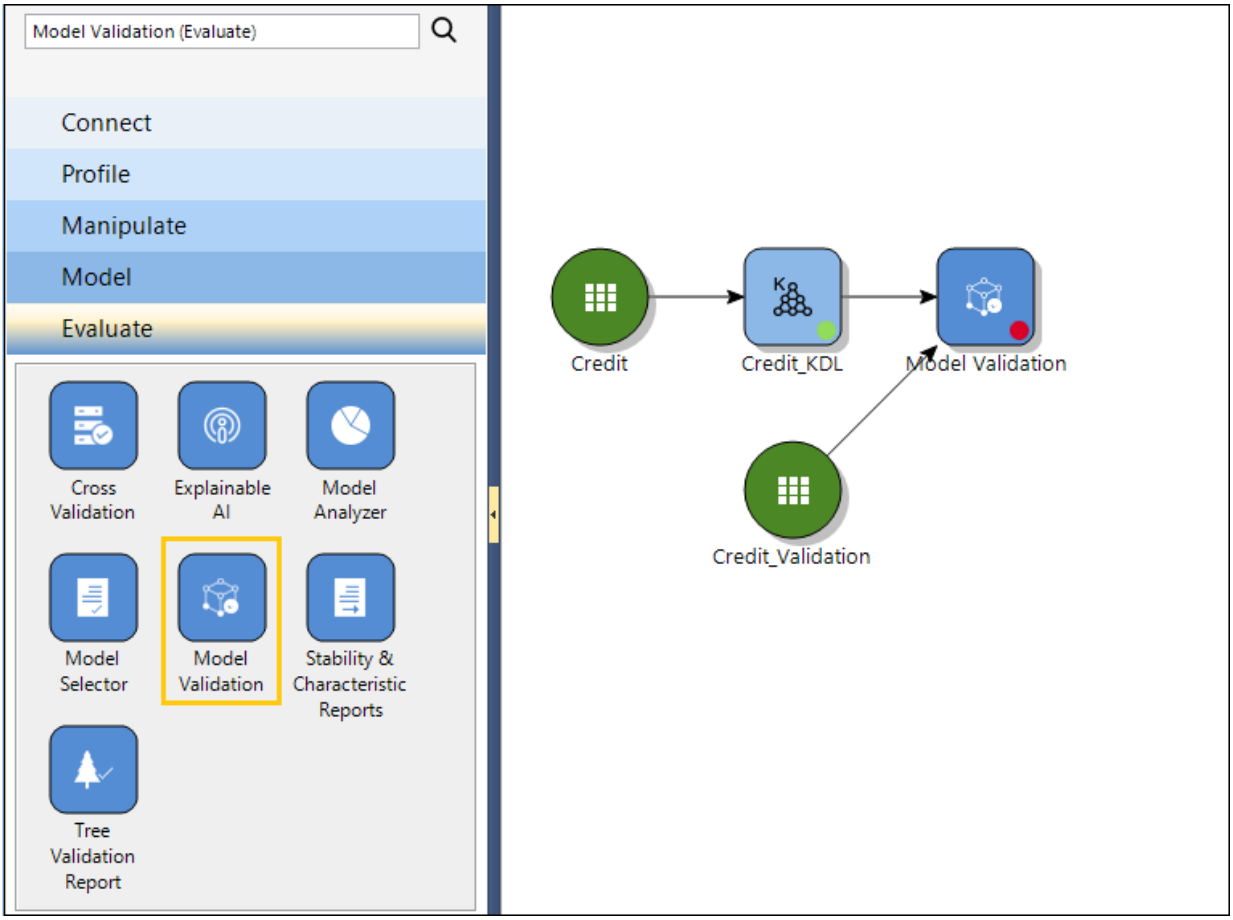
Prerequisites for Keras Deep Learning model validation: a **Keras Deep Learning** node and an input dataset to validate.

To add a Model Validation operation to your workflow:

- Drag the **Model Validation** node from the **Evaluate** palette onto the Workflow canvas and connect the inputs: a **Keras Deep Learning** node and a dataset node that contains the data to validate.

OR

- Right-click anywhere on the workflow canvas and select the context menu command **Evaluate --> Model Validation**. This will create a new Model Validation node. Connect the inputs: a **Keras Deep Learning** model node and a dataset node that contains the data to validate.

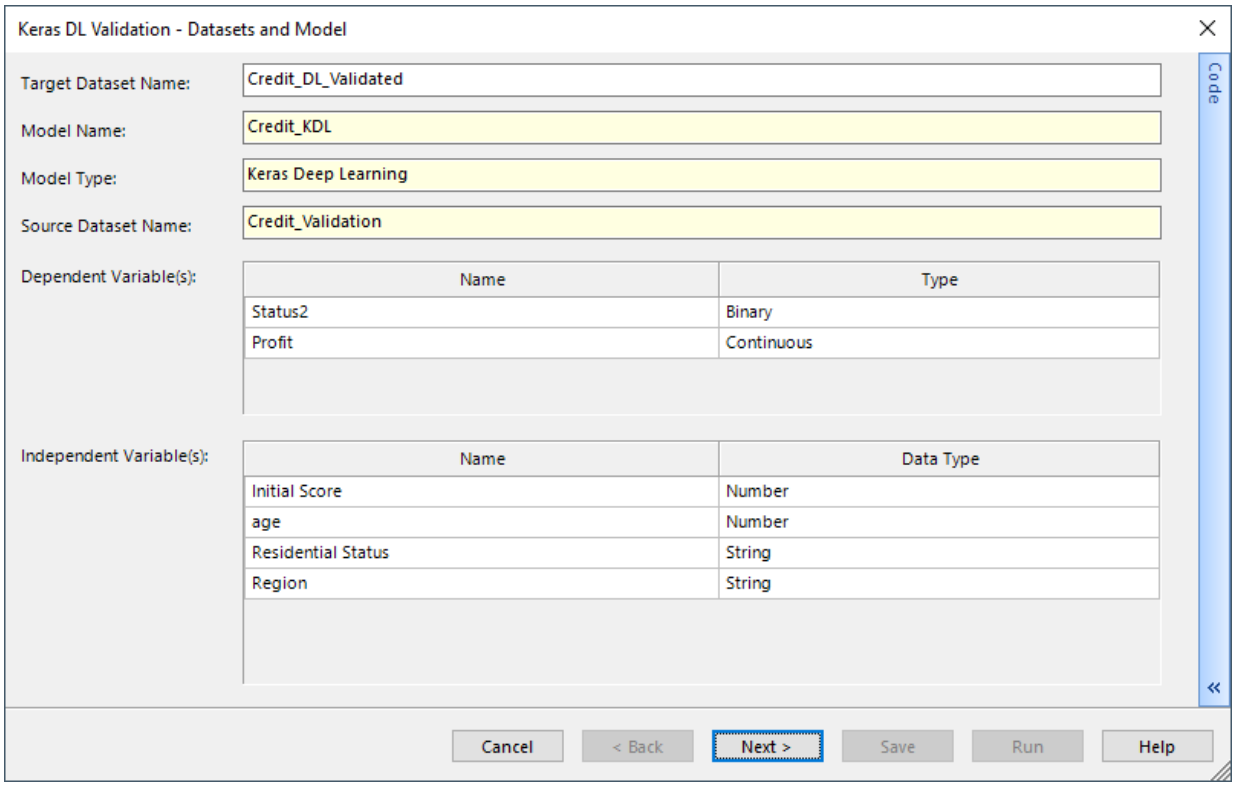


Double-click the **Model Validation** node or right-click on it and select **Modify**.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The Keras Deep Learning Validation wizard opens. It displays the names of the input model and dataset, the names and types of the dependent and independent variables of the model.

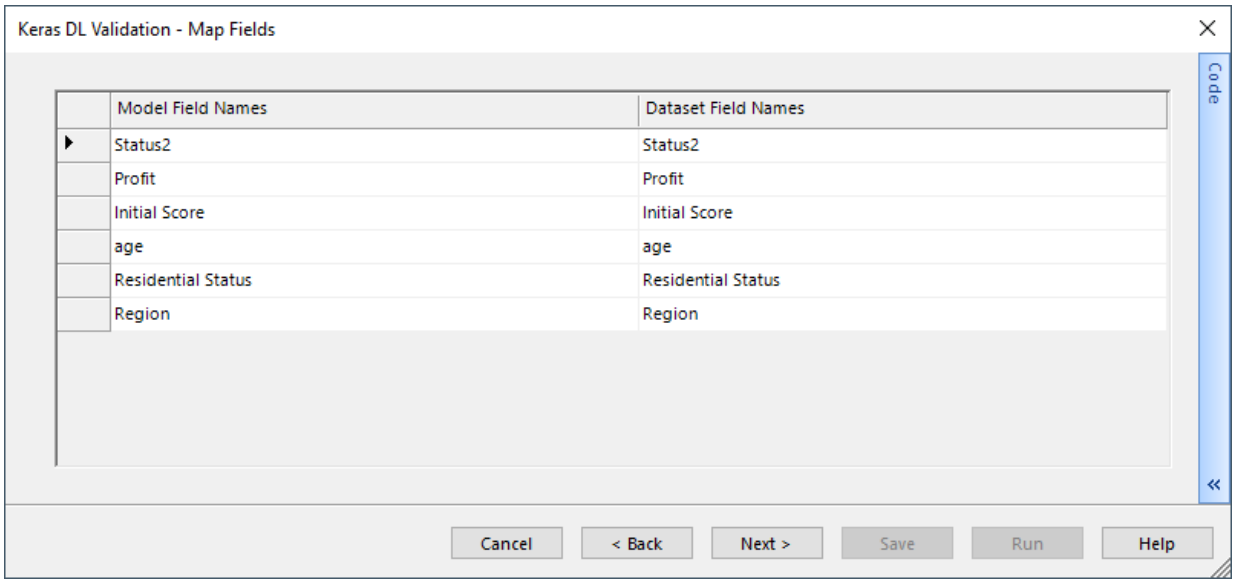
In the *Target Dataset Name* field, specify the name for the output dataset that will contain the validation results.



Click **Next**.

The next step provides the option of mapping the field names in the model to the ones in the validation input dataset. In some cases, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map some fields by clicking in the Dataset Field Names column and selecting the right fields to match the model variables in the left column:

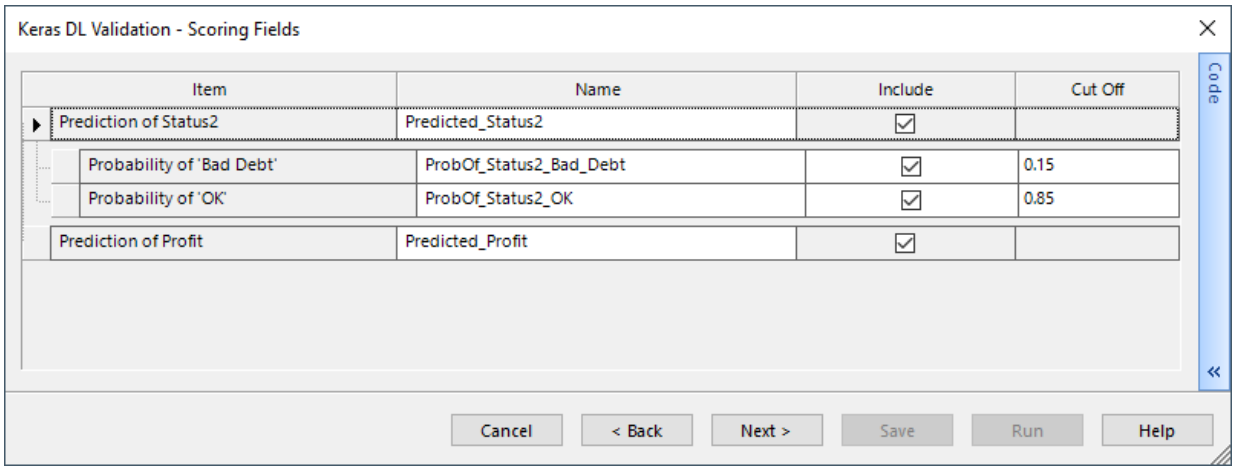
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Note: When validating Keras Deep Learning models that use derived variables (also called calculated columns) with expressions, the input dataset must include these variables. This is different from the case of native Altair models with derived variables, which can be validated on a dataset with only *original* variables that the derived ones depend on, while the derived ones are not required.

Click **Next**.

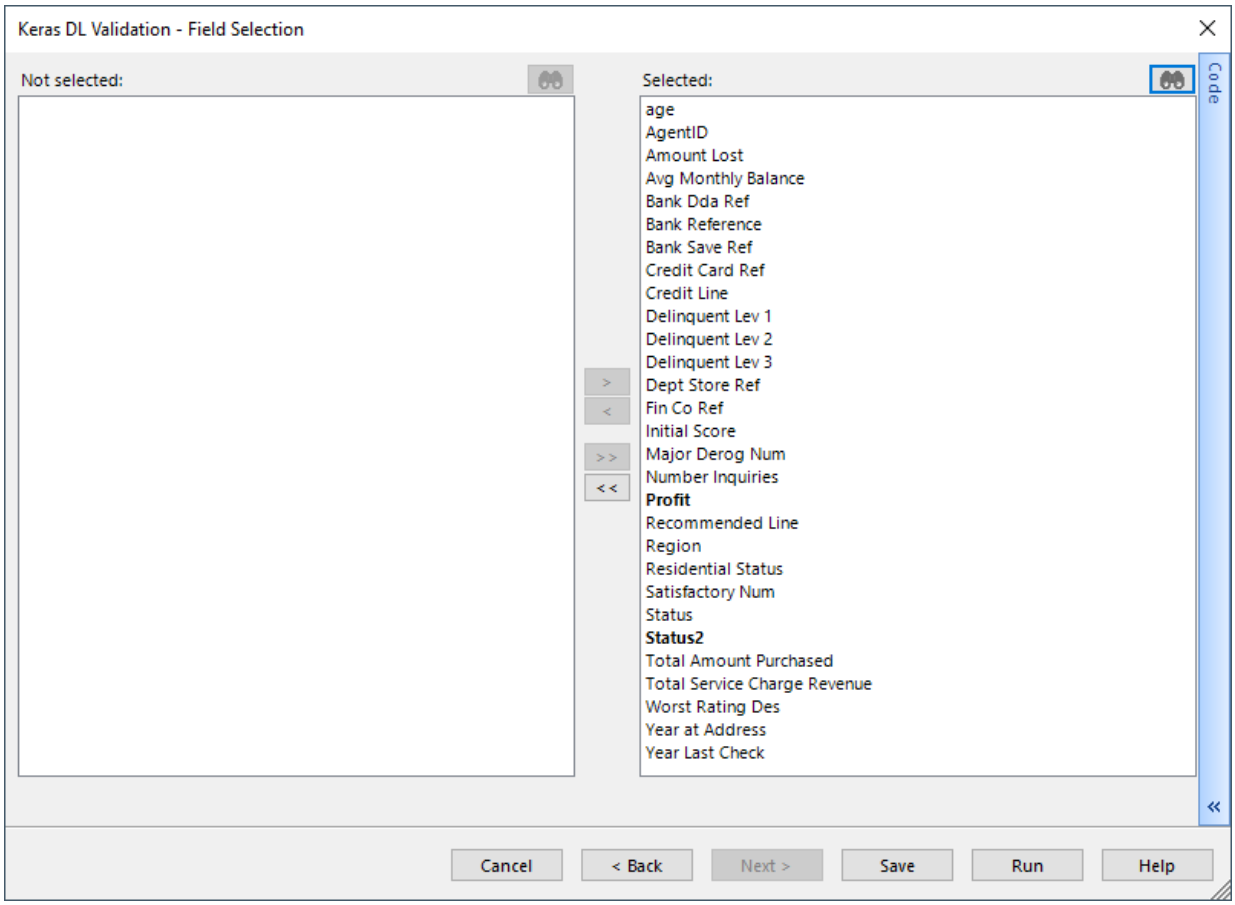
In the **Scoring Fields** page, select the scoring fields to be included in the scoring output dataset and modify their names if necessary. For discrete dependent variables, specify the probability cut-off values. See the general Validation Wizard topic for more information on cut-off values.



Click **Next**.

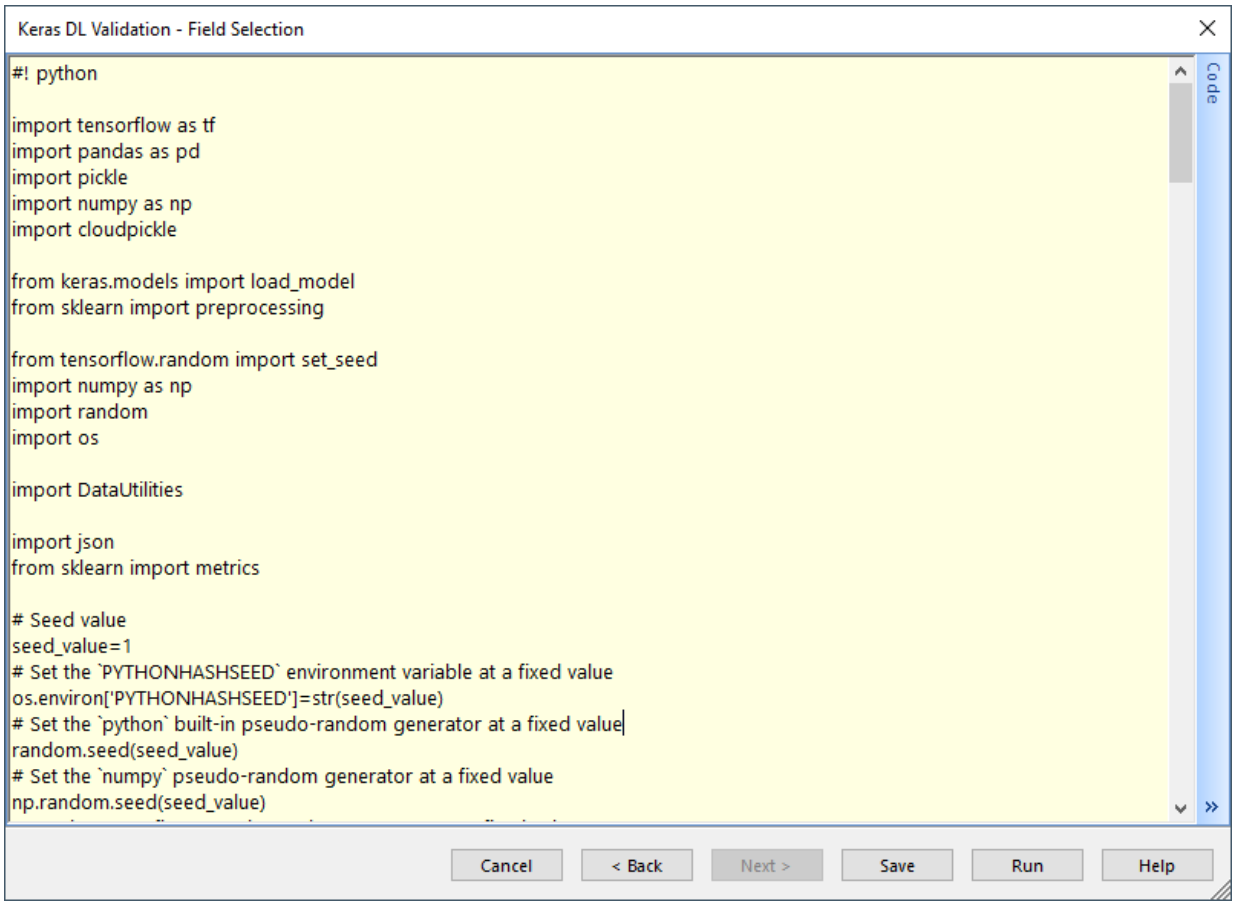
In the **Field Selection** page, choose the fields from the input dataset that you would like to include in the validation output in addition to the new score / probability fields. The field(s) representing the dependent variable(s) are highlighted in bold font and cannot be excluded. By default, all fields in the validation input dataset are selected to be included in the output.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

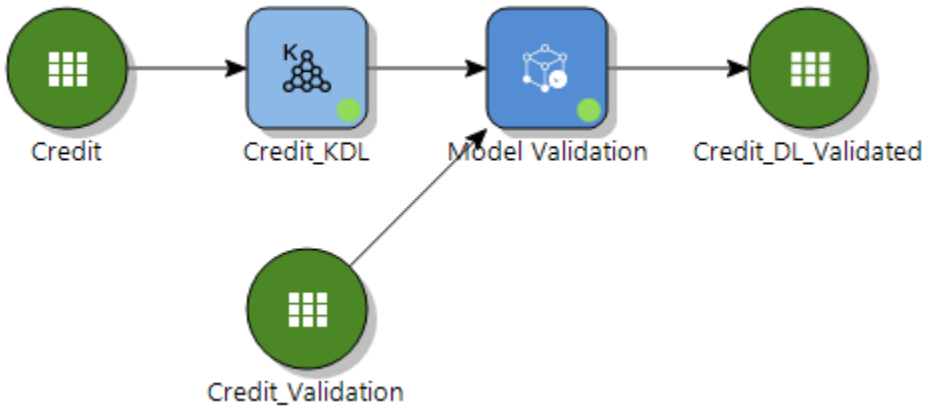


To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. You can also use the right-click context menu to copy the code, save it in a file, or search for specific code fragments. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the validation process immediately.

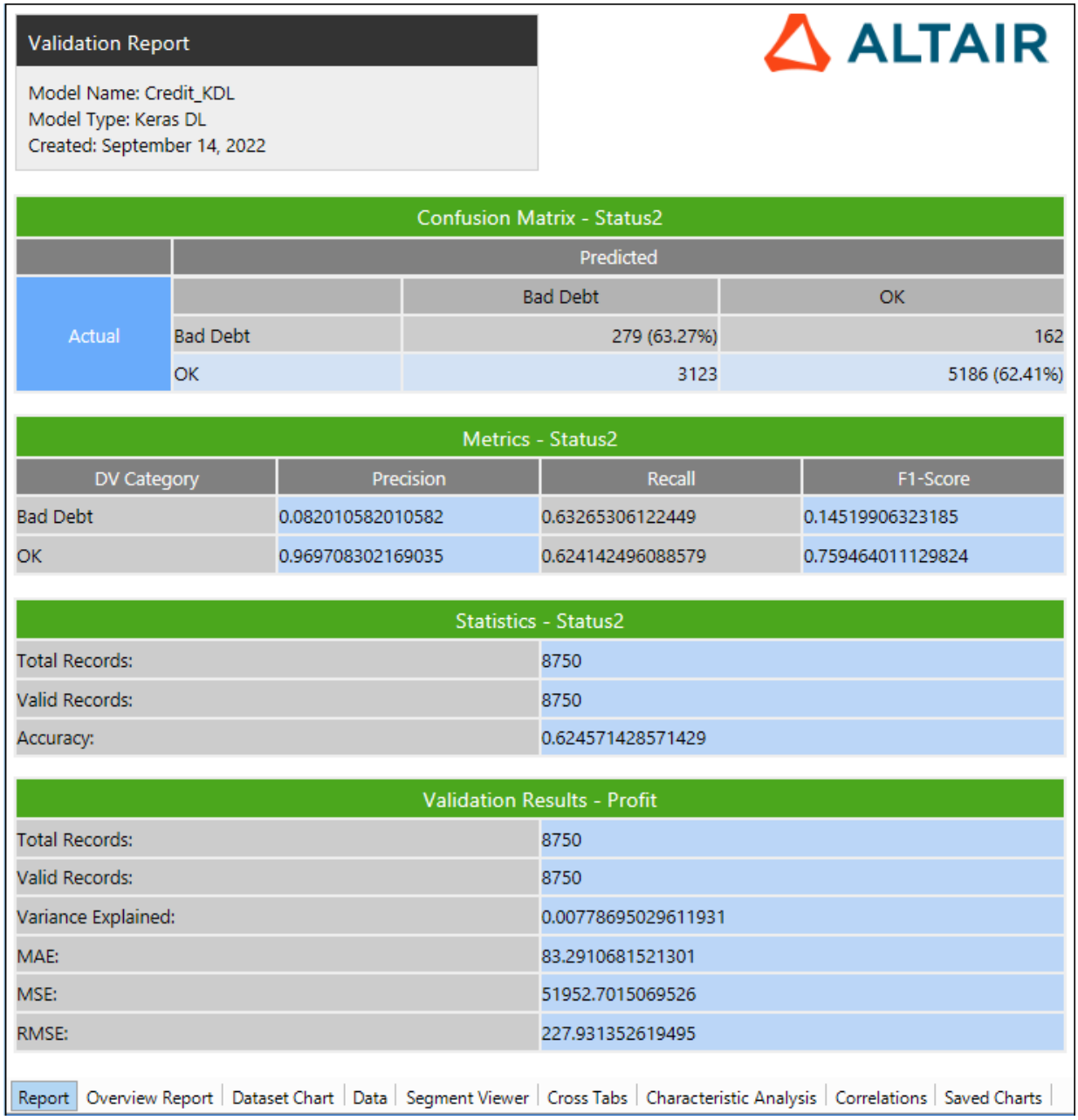


Double-click on the output dataset to view the validation results. The first tab of the output dataset is the Validation Report. It displays the values of the model evaluation metrics.

- For discrete dependent variables, along with the model accuracy, the Report tab shows the Confusion Martix, the table showing *Precision*, *Recall*, and *F1-Score.*

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- For continuous dependent variables, the Report tab shows Variance Explained (R-Squared), Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE).



Validation Report

Model Name: Credit_KDL
Model Type: Keras DL
Created: September 14, 2022

ALTAIR

Confusion Matrix - Status2

		Predicted	
		Bad Debt	OK
Actual	Bad Debt	279 (63.27%)	162
	OK	3123	5186 (62.41%)

Metrics - Status2

DV Category	Precision	Recall	F1-Score
Bad Debt	0.082010582010582	0.63265306122449	0.14519906323185
OK	0.969708302169035	0.624142496088579	0.759464011129824

Statistics - Status2

Total Records:	8750
Valid Records:	8750
Accuracy:	0.624571428571429

Validation Results - Profit

Total Records:	8750
Valid Records:	8750
Variance Explained:	0.00778695029611931
MAE:	83.2910681521301
MSE:	51952.7015069526
RMSE:	227.931352619495

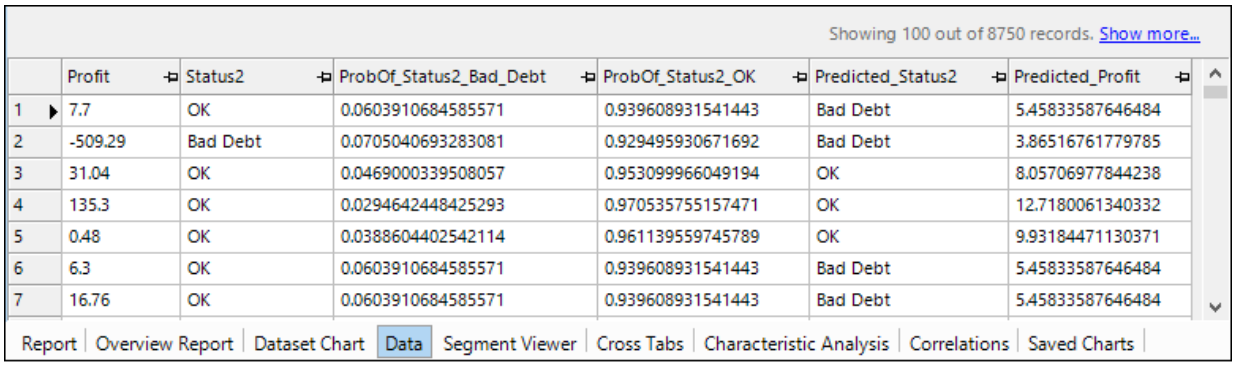
Report | Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

Click the **Data** tab of the output dataset and scroll to the rightmost columns to view the predictions and probabilities for the dependent variables.

- For discrete dependent variables, the output dataset contains the probabilities of the dependent variable values and the prediction (the value that has the highest probability) for each record.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- For continuous dependent variables, the output dataset contains the predicted value for each record.



Showing 100 out of 8750 records. Show more...

	Profit	Status2	ProbOf_Status2_Bad_Debt	ProbOf_Status2_OK	Predicted_Status2	Predicted_Profit
1	7.7	OK	0.0603910684585571	0.939608931541443	Bad Debt	5.45833587646484
2	-509.29	Bad Debt	0.0705040693283081	0.929495930671692	Bad Debt	3.86516761779785
3	31.04	OK	0.0469000339508057	0.953099966049194	OK	8.05706977844238
4	135.3	OK	0.0294642448425293	0.970535755157471	OK	12.7180061340332
5	0.48	OK	0.0388604402542114	0.961139559745789	OK	9.93184471130371
6	6.3	OK	0.0603910684585571	0.939608931541443	Bad Debt	5.45833587646484
7	16.76	OK	0.0603910684585571	0.939608931541443	Bad Debt	5.45833587646484

Report | Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

See also:

Keras Deep Learning

Keras Deep Learning Scoring

Reject Inference

Overview

Purpose and Benefits of Reject Inference

Reject inference methods are used in credit scoring to reduce the bias and increase the relevance of scorecards to the whole population by inferring the behavior of rejected applicants.

Models are usually developed based on accounts with known performance – samples with known account status (Good or Bad). These are samples of customers that have been previously approved. Some of them ended up in debt, some have been in good standing since the time of their approval. However, application scorecards are intended to predict the behavior of ALL applicants. Using a model based on only previously approved applicants can lead to inaccurate results. The results will be even less accurate if the previous approval/rejection decisions were made according to certain rules rather than at random, so the population of accepted applicants is a biased sample that does not adequately represent all applicants.

Reject inference involves analyzing the performance of previously rejected applicants to estimate their potential behavior, i.e., infer whether they would end up with the Good or Bad status. Just as there are some of the approved accounts that will turn out to be Bad, there are some of the rejected ones that would have turned out to be Good had they been accepted. The purpose is therefore to predict the odds of Good/Bad status for the whole population, assuming the approval rate of 100%. The results can also be used to form a swap set – exchange known bad performers for unknown (inferred) good performers, while rejecting both known and inferred bad performers.

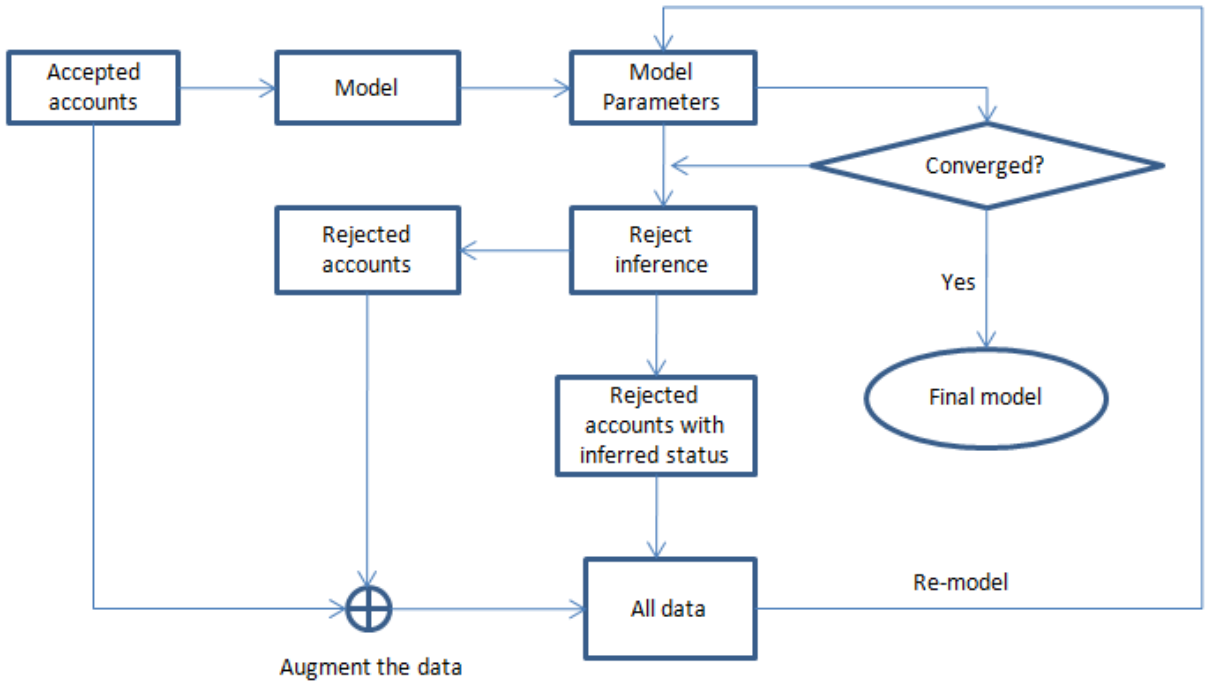
If a bank considers approving some of the previously rejected applicants, reject inference also helps to understand what to expect from them. For example, when deciding to reduce the acceptance threshold by a number of points, it makes sense to estimate the incremental risk taken

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

by decreasing the threshold. By allowing to estimate the bad rates for those who were previously rejected, reject inference helps determine the safe margin for decreasing the acceptance threshold.

Reject Inference Methods

The general reject inference implementation methodology is represented in the following diagram:



Knowledge Studio provides the choice of three common reject inference methods:

Proportional Assignment

In this method, the Good and Bad status is assigned to rejected population at random, keeping the Good/Bad ratio at a fixed level, which is usually set to be the same as in the accepted population. In Knowledge Studio, the default is 50%, which means that half of the rejected population is expected to end up with Bad status. The value can be changed by the user. When the rate is kept the same as in the accepted population, this means the original acceptance process is assumed to be random or inconsistent enough to be approximated by random selection.

Hard Cutoff

The Hard Cutoff method, also known as Simple Augmentation, relies on the results of scoring the rejected population using the model developed on the accepted population. A fixed cutoff score is set so that any rejected applicant with a score below this cutoff is classified as ‘Bad’, and any applicant above the cutoff is classified as ‘Good’. The cutoff score is not arbitrary, since the rejected population is of higher risk than the accepted population, assuming the acceptance process is not random. It is recommended that the cutoff be the current bad rate (probability of "Bad") in the accepted population multiplied by a factor, usually between 2 and

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

5. In this method, it is usually assumed that the first model based on the accepted accounts is known to be fairly "strong".

After the assignment of status values to the rejected accounts, a new model is developed based on ALL accounts and used for the final scorecard.

Parceling

The Parceling method is a hybrid between the hard cutoff and proportional assignment methods. Instead of classifying all rejected applicants at a certain fixed score as Good or Bad, this method assigns goods and bads in proportion to the *expected* bad rate at that score. Bad status values are distributed in ranges according to the score generated by the first model based on the accepted population.

The parceling method takes into account that the potential scores of rejected accounts are spread over the whole range of scores (probabilities) produced by the model based on the accepted population. It is based on assigning the Bad status to rejects in each score range so that the distribution of Bad in the rejected population is the same as in the accepted population. Assignment of rejects in each range is done at random or using the value of the scores in the corresponding range. After assigning the status to the rejects, the two datasets are mixed and a new model is developed on the resulting dataset. Since the Bad rate is expected to be higher among rejected applicants, rejects in lower score ranges are assigned higher Bad rates than those observed for the accepted applicants. These higher rates are often adjusted over several iterations to achieve an overall Bad rate among the rejects 2-5 times the Bad rate for accepted applicants.

Fuzzy Augmentation

The Fuzzy Augmentation method is similar to the Hard Cutoff method. Rather than assigning a "Good" or "Bad" status, it assigns to each rejected applicant a partial "Good" and a partial "Bad" status. The process involves classification and subsequent augmentation according within the following procedure:

Step 1 (Classification):

- Rejects are scored with the known Good/Bad model, and the probabilities of Good and Bad are determined for each reject based on expected bad rates.
- Each reject is assigned a partial Good and a partial Bad status by creating two weighted cases from each reject record.
- Rejected goods are weighted with the probability of Good, and rejected Bads are weighted with the probability of Bad.

Step 2 (Augmentation):

- Inferred rejects and accepts are combined, adjusting for the approval rate. For example, the frequency of a "Good" from rejects is the probability of Good multiplied weight, where the weight is the probability of a reject to be included in the combined (augmented) dataset.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The benefit of the method is that it takes into account the probability of a reject to be accepted in the first place, therefore assigning some measure of importance to a reject in the final dataset.

Reject Inference Wizard

For general information on reject inference and its implementation methodology, see the topic Reject Inference: Overview.

As follows from the overview, you must have two input datasets and a logistic regression model as prerequisites to the reject inference process:

- a dataset containing data for previously accepted applicants, including the column indicating their account status information - usually "Good" or "Bad". This dataset should also contain columns with Weight Of Evidence - the result of coarse classing, which is one of the first stages in the scorecard development process. These columns are usually produced in the Dataset Editor using the WOE Transformation.
- a dataset representing rejected applicants' information
- a Logistic Regression predictive model based on the first dataset (accepted applicants), with the binary dependent variable representing the account status.

Note: For information on coarse classing using the Weight Of Evidence and building a regression model in the scorecard development process, see the topics Overview: Scorecards, WOE Transformation, and the Scorecard section in the Tutorial.

In the example screenshot below, the *Credit Accounts* dataset contains the accepted applicants' data, and the *Rejected* dataset has the rejected applicants' data.

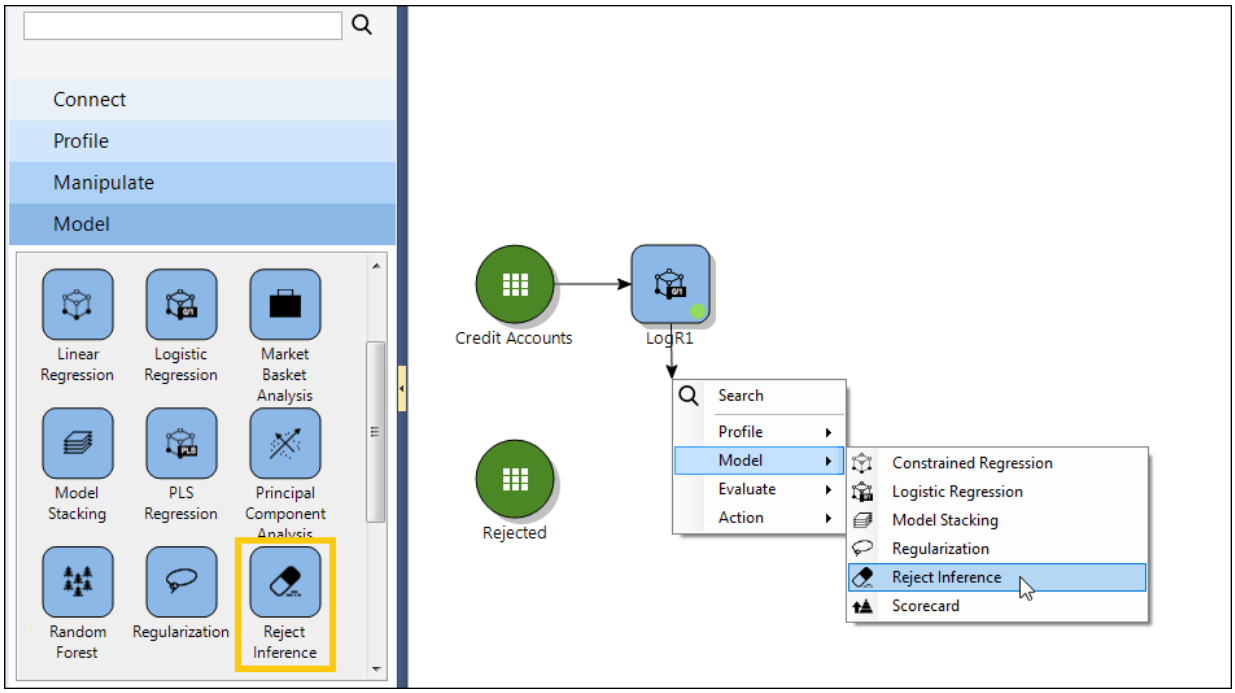
To add a Reject Inference operation to your workflow:

- Drag a connection arrow from the input Logistic Regression model to any area on the Workflow canvas background. Releasing the mouse button brings up a context menu. From the context menu, select **Model --> Reject Inference** as shown in the example below. A new **Reject Inference** node will be created. Then connect the input datasets to the Reject Inference node.

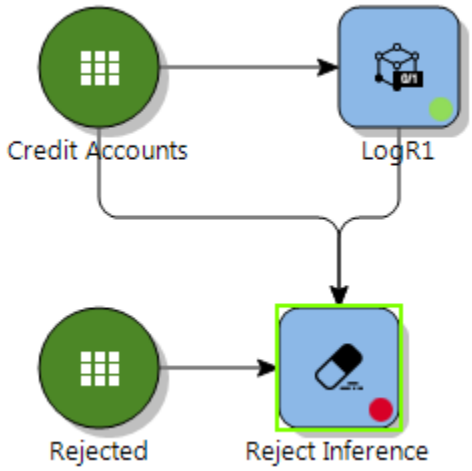
OR:

- Drag the **Reject Inference** node from the **Model** palette onto the canvas and connect the inputs (Logistic Regresion model and two datasets).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Double-click the **Reject Inference** node or right-click on it and select **Modify** from the context menu.



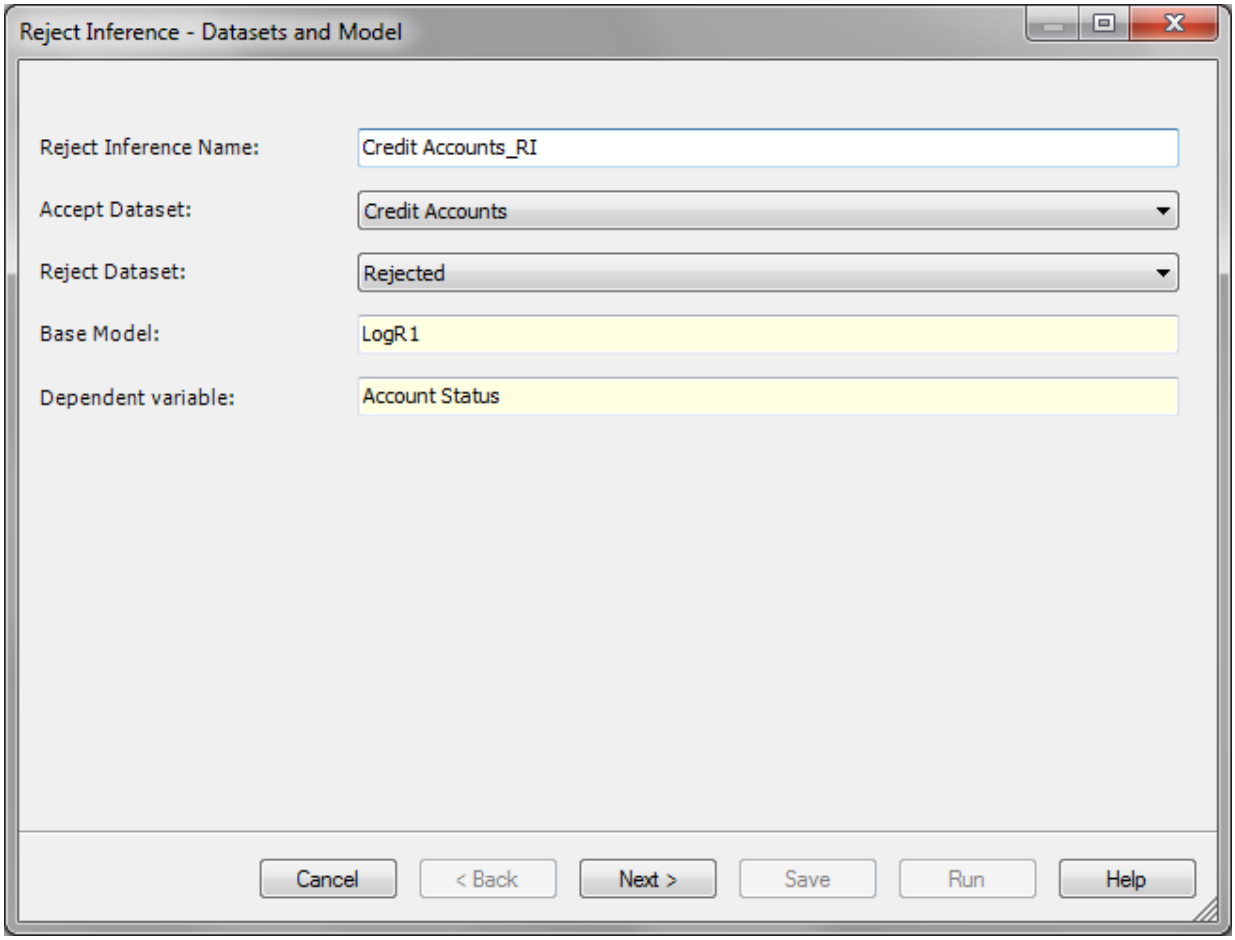
In the first page of the Reject Inference Wizard, specify the Reject Inference Name. This is the reject inference result, which is an object containing a report and including child datasets and/or models with inferred Good/Bad rates for the rejected applicants.

For the *Accept Dataset*, Select the dataset with the accepted applicants' data that satisfies the prerequisite requirements above. In particular, it should contain the Weight Of Evidence columns.

For the *Reject Dataset*, select the dataset with rejected applicants' information. It obviously contains no account status information.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

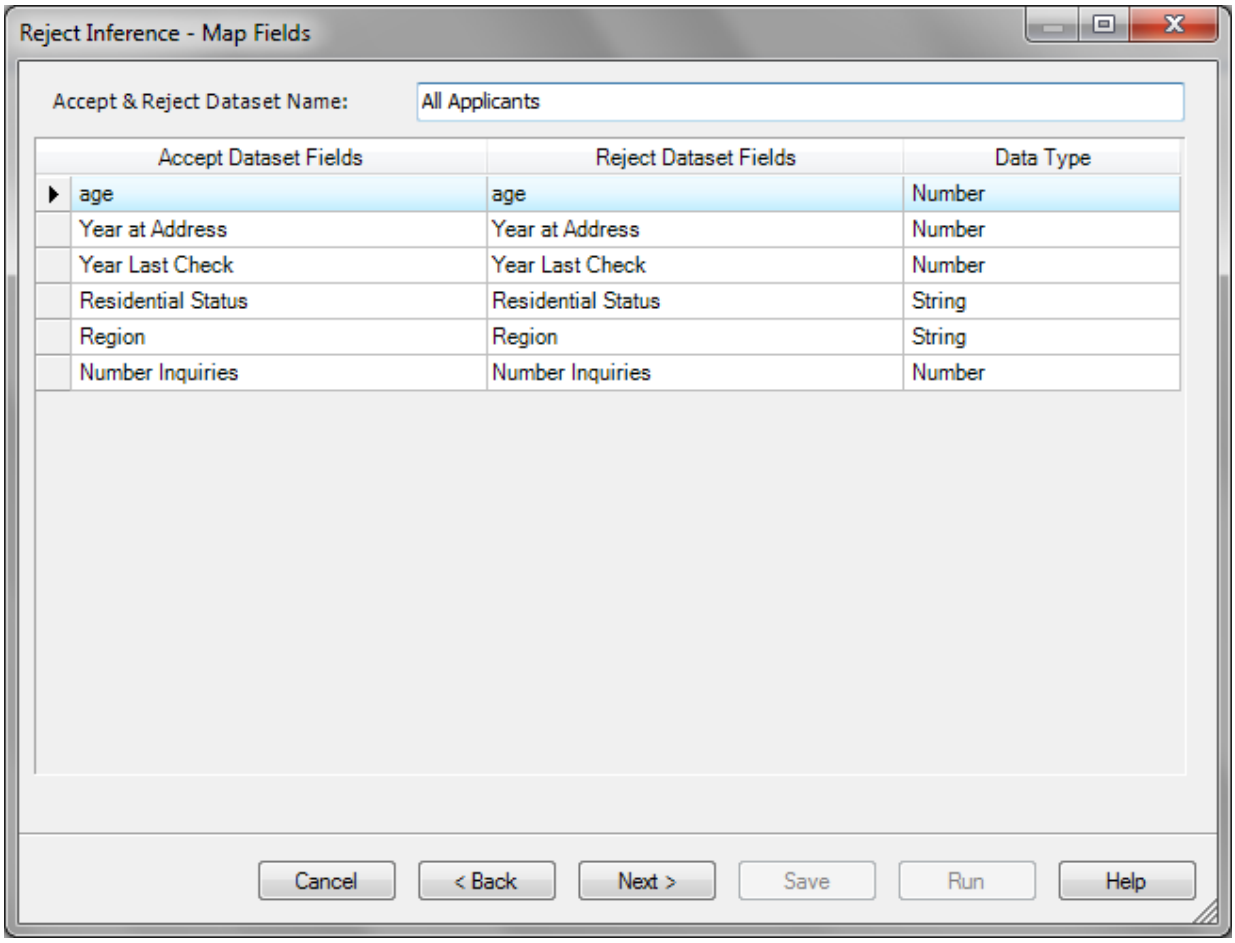
For the *Base Model*, Select the Logistic Regression predictive model satisfying the prerequisite requirements above. Its independent variables should be the Weight of Evidence columns of the Accept Dataset. The dependent variable name will be displayed below as soon as you select the Base Model.



If the inputs don't satisfy the prerequisite requirements indicated above, you will get a warning message "The Base Model is not trained on the Accept Dataset or independent variable(s) is not a WOE expression." In this case, fix these problems and try again.

Click Next. In the Map Fields page, specify the name for the combined Accept & Reject dataset, which will be the result of appending the Reject Dataset to the Accept Dataset. Then verify that the corresponding columns match. If any of them don't match, click in the *Reject Dataset Fields* column and select the correct column. Verify that the data types of the corresponding columns match, which is indicated by the Data Type column.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



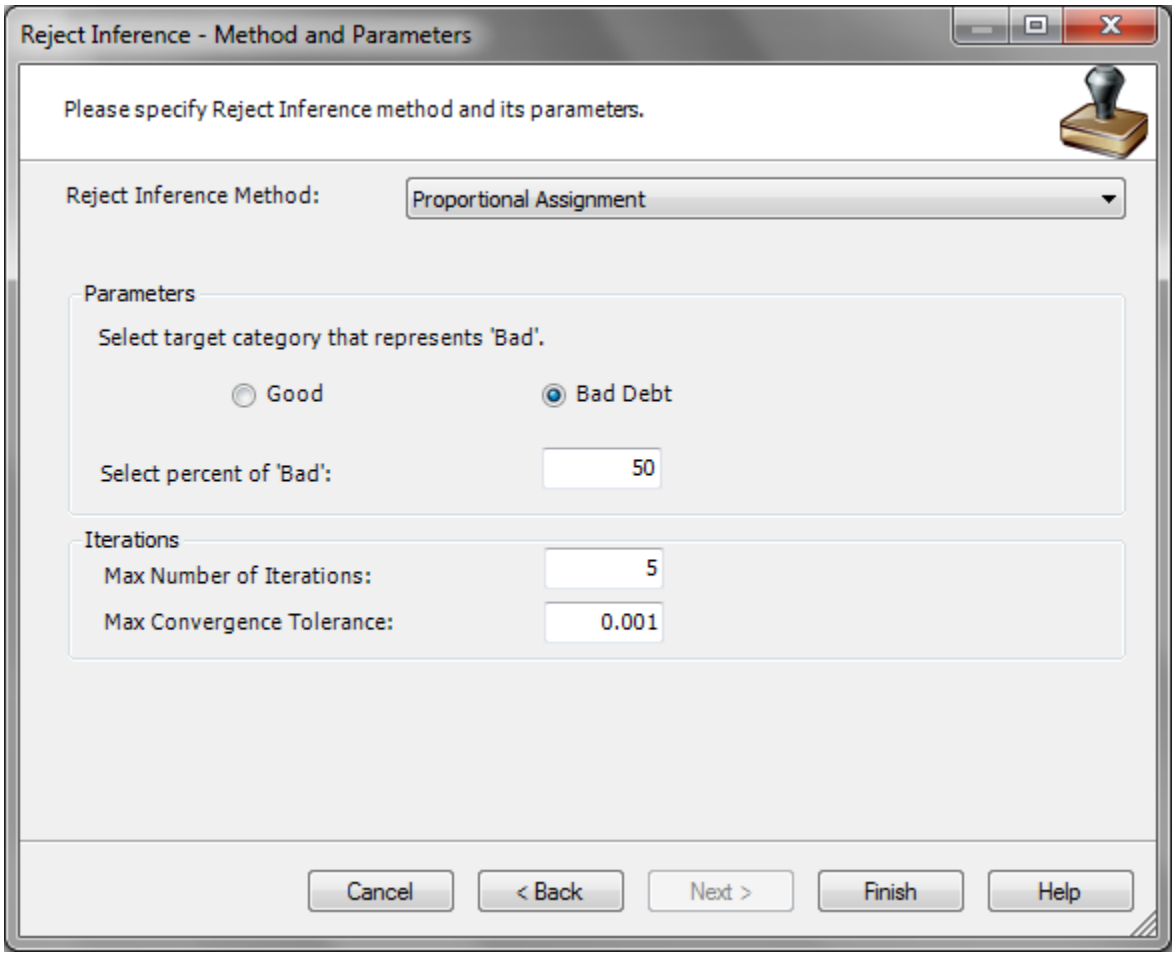
Click Next. In the Method and Parameters page, select the desired reject inference method and specify its parameters. The available methods are: *Proportional Assignment*, *Hard Cutoff*, and *Parceling*. See the topic Reject Inference: Overview for reference.

The parameters to specify depend on the selected method.

Proportional Assignment

If you selected the Proportional Assignment method, which is the default, the method parameters to specify are as follows:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



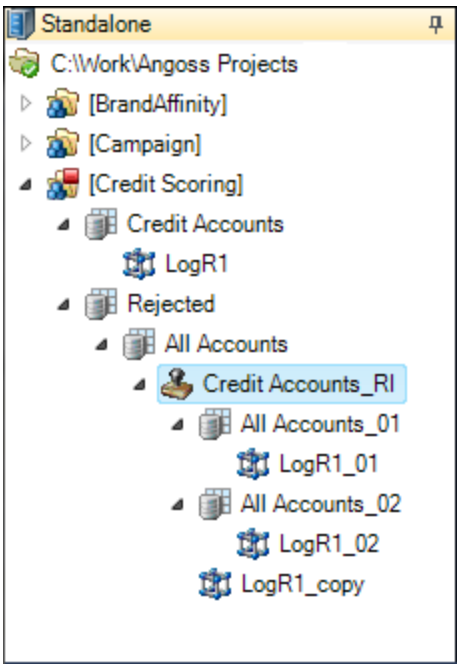
Select target category that represents 'Bad'	The value of the dependent variable that represents the "Bad" status of an account. The actual values, which are determined from the base model, are displayed next to the radio buttons. Select the appropriate radio button. In the example screenshot above, the selected target value is "Bad Debt".
Select percent of 'Bad'	Specify the Bad rate (in percentage) to be assigned to the rejected population. The Bad status will be assigned at random. The default is 50%, which means that half of the rejected population are expected to end up with Bad status. See Reject Inference: Overview for details. If you are not sure what value to specify, check the actual Bad rate in the Dataset Chart tab of the Accept Dataset and estimate how much higher you expect it to be for the rejects.
Max Number of Iterations	The maximum number of iterations in the process. During each iteration, a new

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

	regression model is created based on the combined dataset - Accept Dataset plus Reject Dataset with inferred status. The default is 5 iterations.
Max Convergence Tolerance	The convergence threshold for the iterative process. This is the threshold for Root Mean Square Error (RMSE) at each step. The default threshold is 0.001. $RMSE = \sqrt{\frac{1}{n}\sum_i (dx_i)^2}$ where dx_i are the parameter estimate errors.

Click **Run**. As a result, the wizard produces the Reject Inference object with child objects including the combined datasets - Accept Dataset plus Reject Dataset with inferred status - and the logistic regression models built on them at each iteration. See the Reject Inference Results topic for information about the reject inference output for each method.

Note: Unlike the usual regression models, which substitute median for missing values by default, the regression models built in the iterative process of Reject Inference ***ignore*** missing values by default More precisely, the wizard makes a copy of the base model and changes the missing value attribute to *ignore* for all scalar variables. This is done to ensure that outliers in the Reject Dataset, i.e., records with "unknown" values that are not observed in the Accept dataset, get the *null* score by default. The copy of the base model can be seen as *LogR1_copy* at the bottom of the example screenshot of the Project Explorer below.



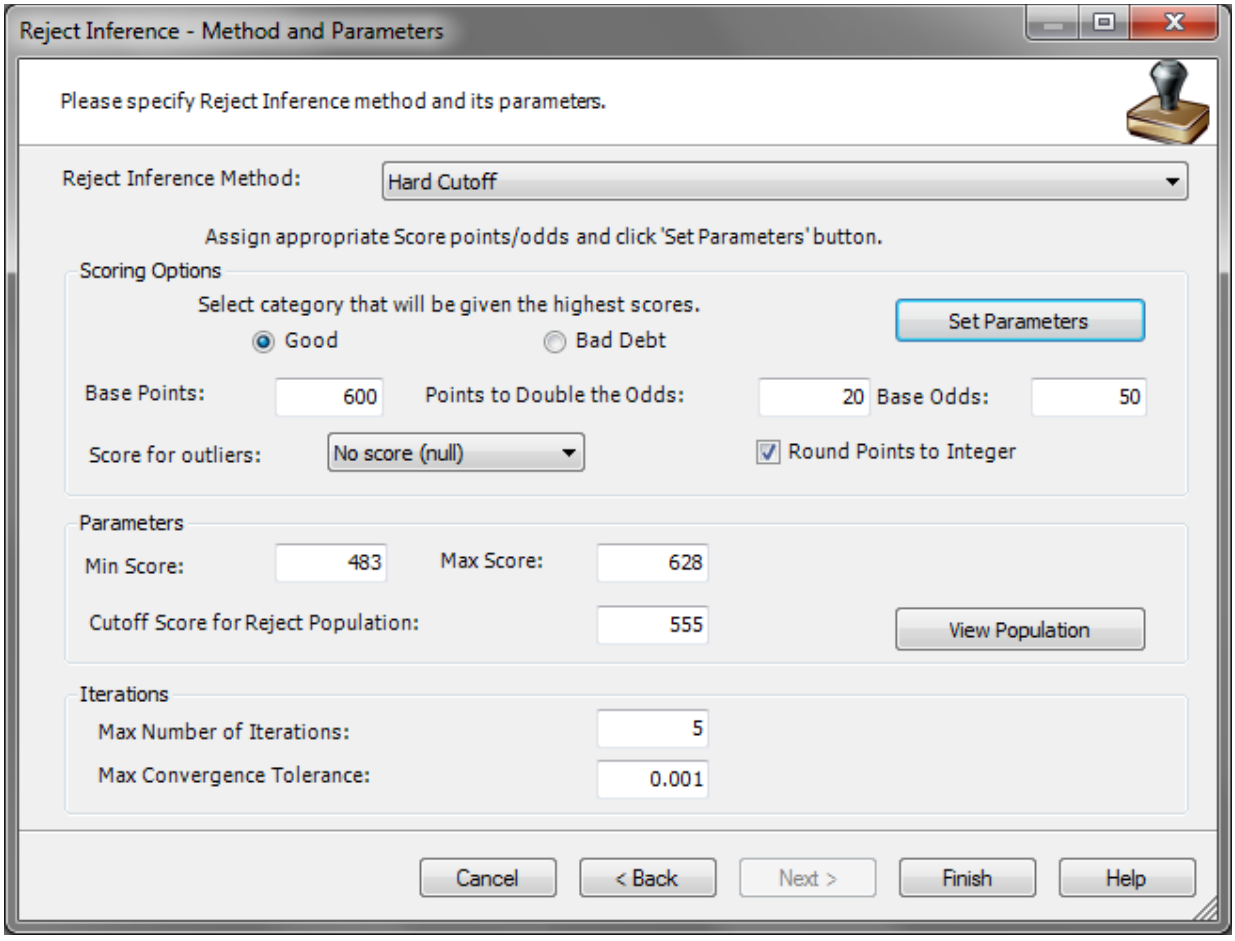
Hard Cutoff

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If you selected the Hard Cutoff method, the parameters to specify are described below.

First, in the Scoring Options section, specify the scaling parameters for the scorecards to be created implicitly at every iteration of the method. As follows from the method description in Reject Inference: Overview, the first of these scorecards is built on the Base Model and then scored on the Reject Dataset. The Bad status is then assigned to all rejects with scores below the user-defined Cutoff Score, and the Good status is assigned to the remaining rejects. The Reject Dataset augmented with Good/Bad status values inferred this way is then appended to the Accept Dataset, and a new logistic regression model is trained on the resulting combined dataset. This process is repeated in accordance with the iteration settings specified in this wizard page.

As in the Proportional Assignment method, the regression models built in the iterative process of the Hard Cutoff method ignore missing values by default rather than substitute median. The wizard makes a copy of the base model and changes the missing value attribute to *ignore* for all scalar variables. See the note in the description of the Proportional Assignment method.



To estimate the appropriate cutoff score for the rejected population, click the **Set Parameters** button upon specifying the scoring options. This will calculate Min Score, Max Score, and the suggested Cutoff Score for Reject Population. Adjust the suggested Cutoff Score as necessary. When this is done, click the **View Population** button, which will show the corresponding frequencies and percentages of the status values (Good/Bad rates) in the Accept Dataset, the Reject Dataset with

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

inferred status, and in the combined dataset that includes the rejects with inferred status. Review the rates and adjust the cutoff score again if necessary.

Select category that will be given the highest scores	This is the category of the dependent variable that is supposed to get higher scores than the other category. More precisely, higher probability of that category produced by the regression model will result in higher score of the scorecard. It is therefore natural to select the value indicating good status. The DV categories, which are determined from the base model, are displayed next to the radio buttons. Select the desired category. In the example screenshot above, the selected category is "Good".
Base Points	Scorecard scaling option for the scorecards implicitly created in each iteration of the method. The number of points for the specified value of Base Odds. The default is 600 points.
Points to Double the Odds	The number of points to double the odds for the scorecard. If this value is set to N, then the odds will be doubling at every N points. The default is 20.
Base Odds	Base odds for the scorecard. The default is 50:1.
Score for Outliers	The score for the records not recognized by the model, i.e., records with values that are not observed in the Accept dataset, which the model was trained on. By default, such records get no score (null). Alternatively, you can assign a custom score or the sum of scorepoint averages (*Avg score*).
Round Points to Integer	If this option is selected, the scorecard is rounded to the nearest integer number. Otherwise, real numbers are used. Note that slight variations between the results of the original logistic regression model and the generated Scorecard model can be observed if you choose to round the points to integer values (which is the default). If you do not round the points, the Scorecard model and the original logistic model reveal exactly the same lift.
Min Score	The minimum score of the scorecard defined by the scaling parameters specified above. This field is not editable.
Max Score	The maximum score of the scorecard defined by the scaling parameters specified above. This field is not editable.
Cutoff Score for Reject Population	The cutoff score such that any rejected applicant with a score below this cutoff is classified as 'Bad', and any applicant above this cutoff is classified as 'Good'. The cutoff score is calculated automatically upon clicking the Set Parameters

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

button, but you can specify a custom cutoff value in this field. See the description of the Hard Cutoff method in the topic Reject Inference: Overview.

Max Number of Iterations The maximum number of iterations in the process. During each iteration, a new regression model is created based on the combined dataset - Accept Dataset plus Reject Dataset with inferred status. The default is 5 iterations.

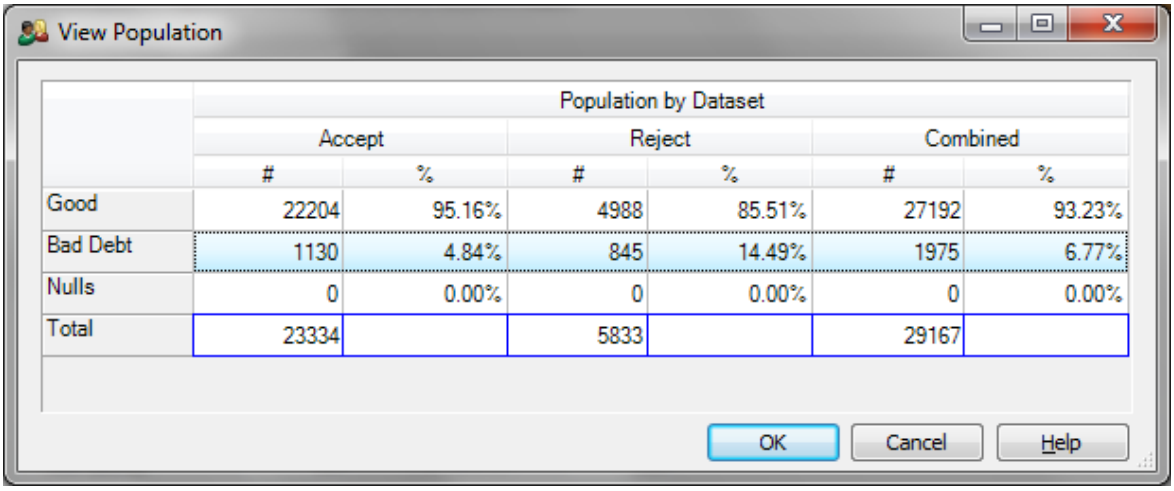
Max Convergence Tolerance The convergence threshold for the iterative process. This is the threshold for Root Mean Square Error (RMSE) at each step. The default threshold is 0.001.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i}(\mathrm{d}x_i)^2}$$

where $\mathrm{d}x_i$ are the parameter estimate errors.

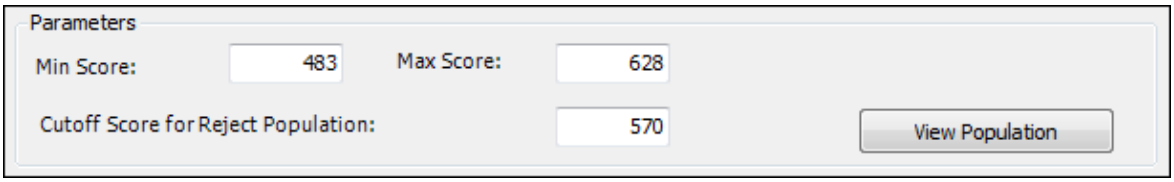
The View Population window brought up by the **View Population** button shows the frequencies and percentages of the dependent variable values (Good and Bad rates) in the Accept Dataset, the Reject Dataset with inferred status, and in the combined dataset that includes the rejects with inferred status. It also shows the number and percentage of records with unassigned status (Nulls), which may happen if there are outliers in the Reject Dataset that got null scores.

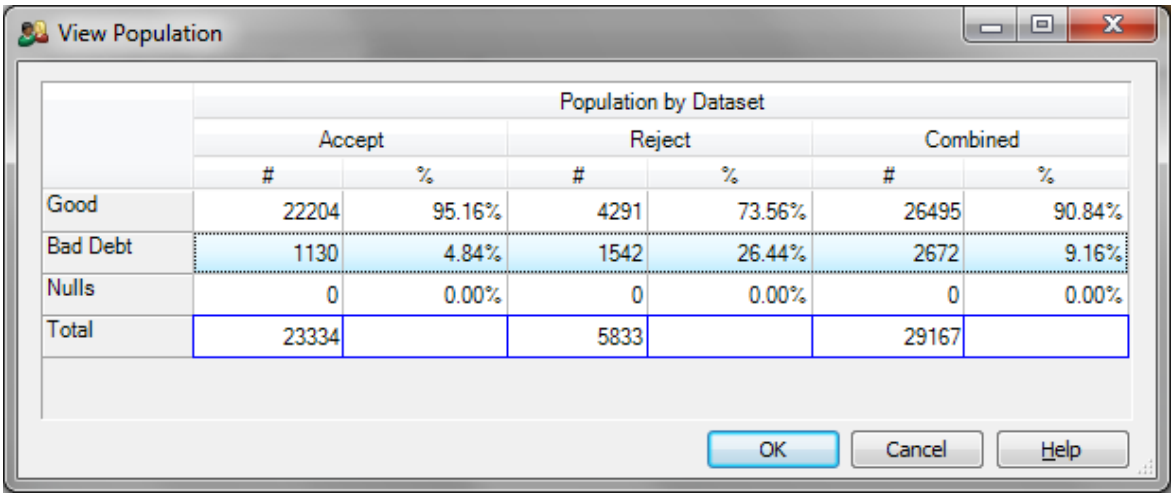
In our example, with the specified Cutoff Score of 555 (see the wizard screenshot above), the inferred Bad Debt rate is 14.49% in the Reject Dataset and 6.77% in the combined dataset of all applicants.



If you expect the Bad rate among rejected applicants to be higher, increase the Cutoff Score and click the View Population button again to check the inferred Bad rates for the new cutoff. In our example, if we increase the Cutoff Score to 570, the inferred Bad Debt rate will be 26.44% in the Reject Dataset and 9.16% in the combined dataset of all applicants:

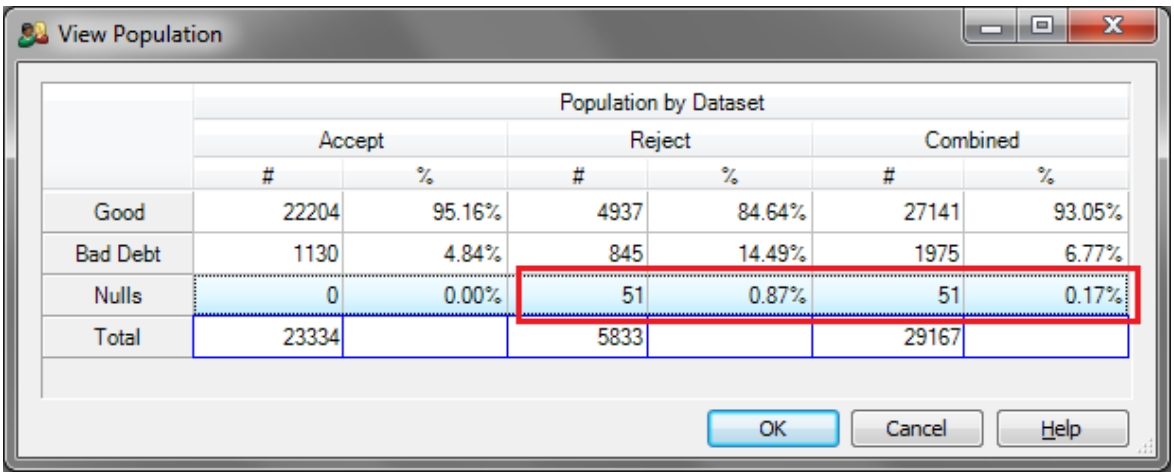
Altair Engineering, Inc. Copyright © 2022. All rights reserved.





Outliers (Records with Unknown Values) in the Reject Dataset

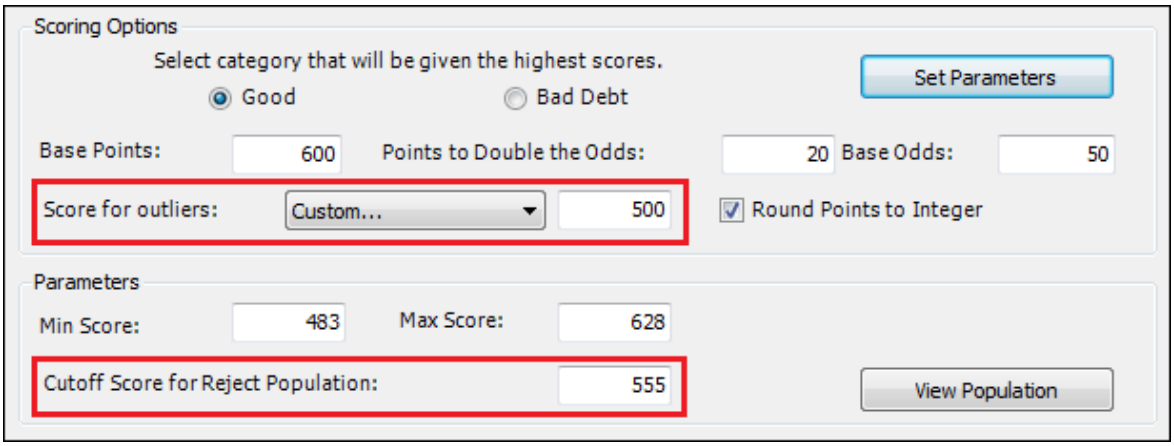
The Reject Dataset may contain records with values that are not observed in the Accept Dataset, in which case these values will not be recognized by the regression model based on the Accept dataset. Such records get no score (null) unless you assign a custom score or the sum of scorepoint averages using the *Score for outliers* option in the Reject Inference Wizard as described above. If you leave the default setting (Score for outliers is null), then after clicking View Population you will see nonzero numbers and percentages in the ***Null*** row for the Reject and Combined dataset, as shown below:



If you set the score for outliers to *Avg score* (the sum of scorepoint averages) or a custom score, then, depending on whether this score is above or below the *Cutoff Score for Reject Population,* all records with values not observed in the Accept Dataset will be assigned to Bads or Goods, respectively. For example, here we set the *Score for outliers* to 500, and after

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

clicking the Set Parameters button the automatically calculated Cutoff Score for Reject Population is 555:



Since 500 < 555, all the outlier records will be classified as *Bad*. If you don't change the cutoff and set the Score for outliers to 556 or higher, all outliers will be classified as *Good*.

Parceling

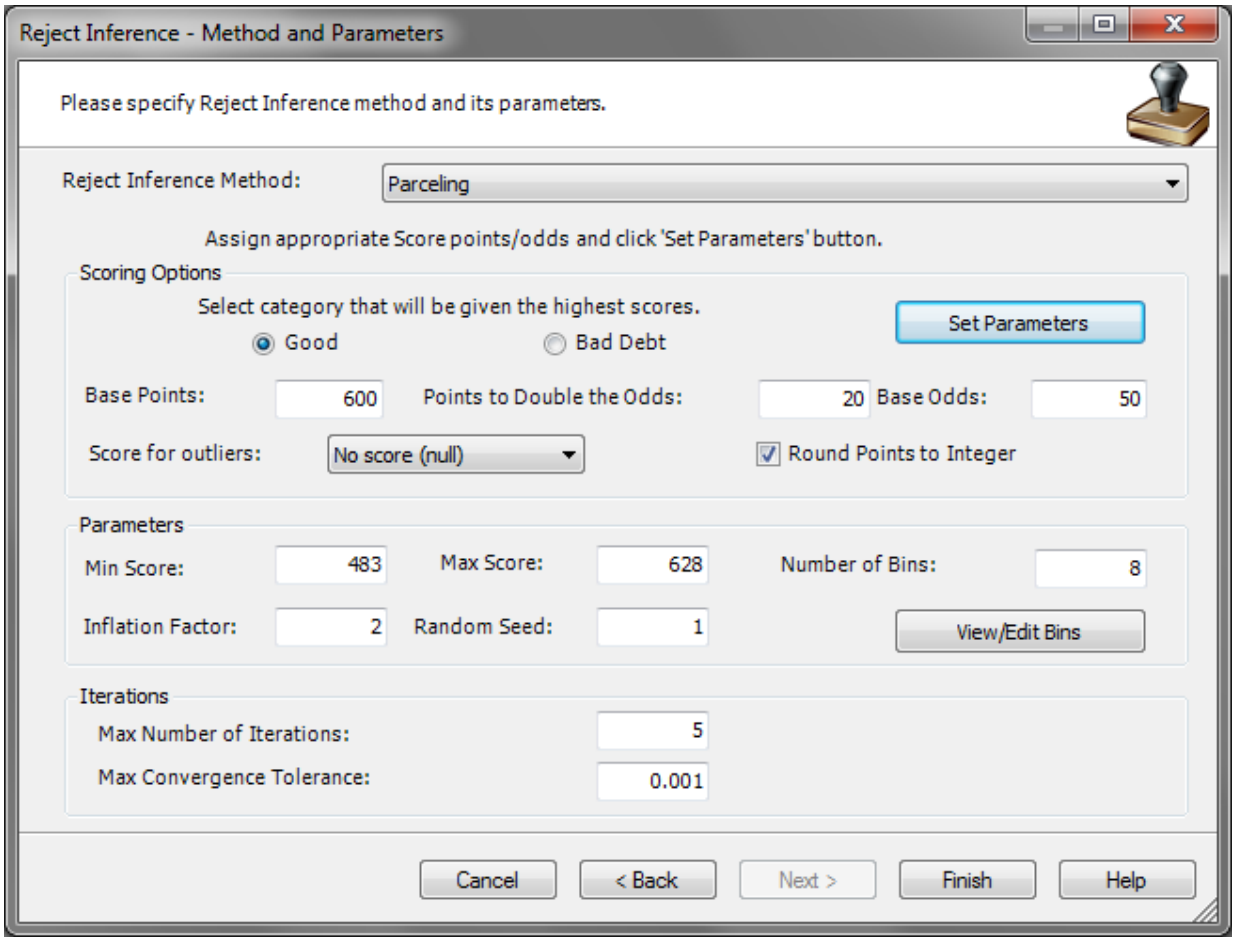
The general description of the Parceling method is provided in the topic Reject Inference: Overview. As in the Hard Cutoff method, first the Reject Dataset is scored with a scorecard built on the Base Model. The next stage is different: the dataset is subdivided into segments corresponding to a number of score ranges, and the Bad status values are distributed separately within each segment.

The Parceling method parameters are the same as for the Hard Cutoff method, except for ***Inflation Factor, Number of Bins,*** and ***Random Seed***. Again, the first step is to select the category of the dependent variable that is supposed to get higher scores than the other category. More precisely, higher probability of the selected category generated by the regression model will result in higher score of the scorecard. It is therefore natural to select the value indicating good status. See the description of other parameters common to both methods in the Hard Cutoff section above.

As in the Proportional Assignment and Hard Cutoff methods, the regression models built in the iterative process of the Parceling method ignore missing values by default rather than substitute median. See the note in the description of the Proportional Assignment method.

The *Random Seed* parameter is provided for the repeatability of results. You will get consistent results if the random seed is the same each time you run the wizard.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click the **Set Parameters** button upon specifying the scoring options. This will calculate Min Score, Max Score, and the suggested Inflation Factor and Number of Bins.

The *Inflation Factor* determines how much higher you expect the target category rate to be for the rejected population than for the accepted population. For example, Inflation Factor = 4 means that you expect the target category rate (e.g., Bad rate) for rejected applicants to be four times higher than that for the accepted ones.

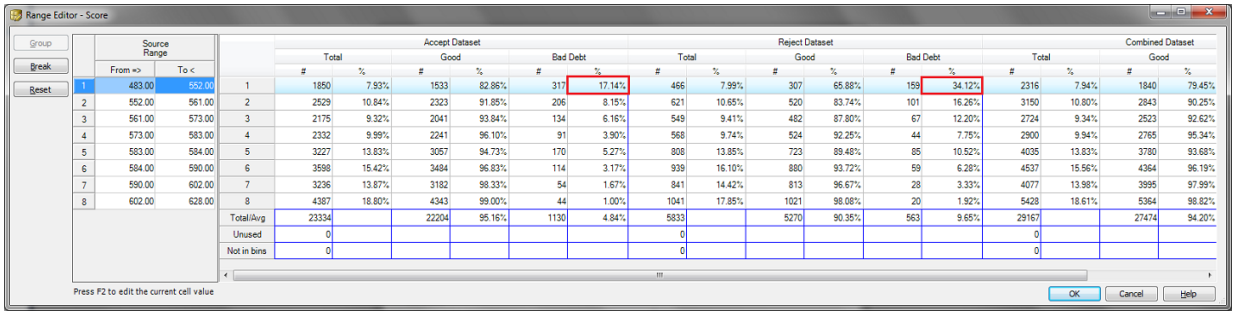
Change the suggested Number of Bins and Inflation Factor if necessary. When this is done, click the **View/Edit Bins** button, which will show the frequencies and percentages of the status values (Good/Bad rates) in the segments corresponding to the score ranges in the Accept Dataset, the Reject Dataset with inferred status, and in the combined dataset that includes the rejects with inferred status. Review the rates and adjust the Inflation Factor again if necessary.

In the Score Range Editor window, you can group and break the ranges and edit their boundaries the same way you would do in any Range Editor dialog for continuous variables, such as the one in the Dataset Chart. The left frame shows the score ranges, and the right frame shows the frequencies and percentages of the dependent variable values in the Accept, Reject, and Combined datasets.

In our example, the specified Inflation Factor is 2. Look at the first row in the screenshot below. In accordance with the Inflation Factor, the inferred Bad rate in the Reject Dataset segment for the

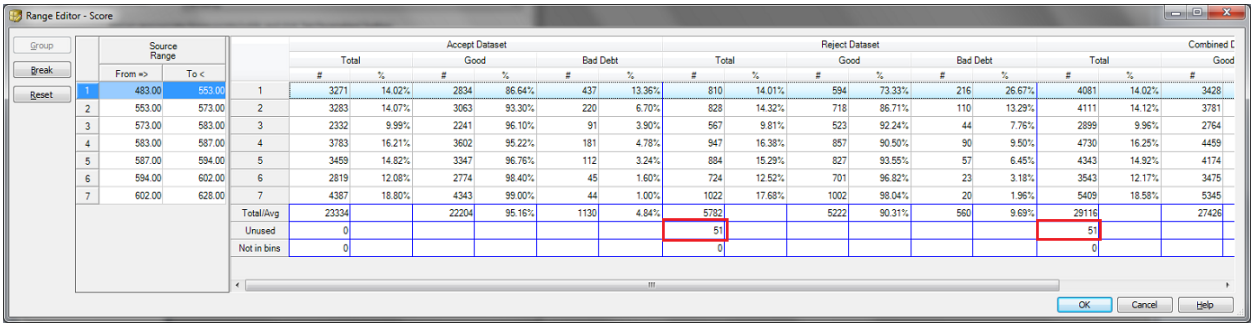
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

first score range [483, 552] is 34.12%, which is about twice as high as 17.14% - the Bad rate in the Accept Dataset segment for the same score range. Click OK to close the Range Editor and adjust the Inflation Factor if necessary.



Outliers (Records with Unknown Values) in the Reject Dataset

If the Reject Dataset contains records with "unknown" values that are not observed in the Accept Dataset, then these values will not be recognized by the regression model based on the Accept dataset. Such records get no score (null) unless you assign a custom score or the sum of scorepoint averages using the *Score for outliers* option in the Reject Inference Wizard as described above. If the score for outliers is set to null, which is the default, then after clicking View/Edit Bins you will see the number of outlier records shown as ***Unused*** in the *Total #* column for the Reject and Combined datasets as in the example below, which means that they will not be included in the Combined dataset.

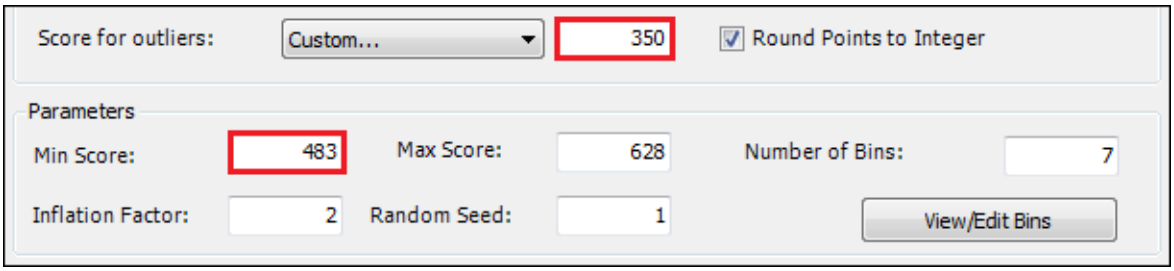


If you set the score for outliers to *Avg score* (the sum of scorepoint averages) or a custom score, then:

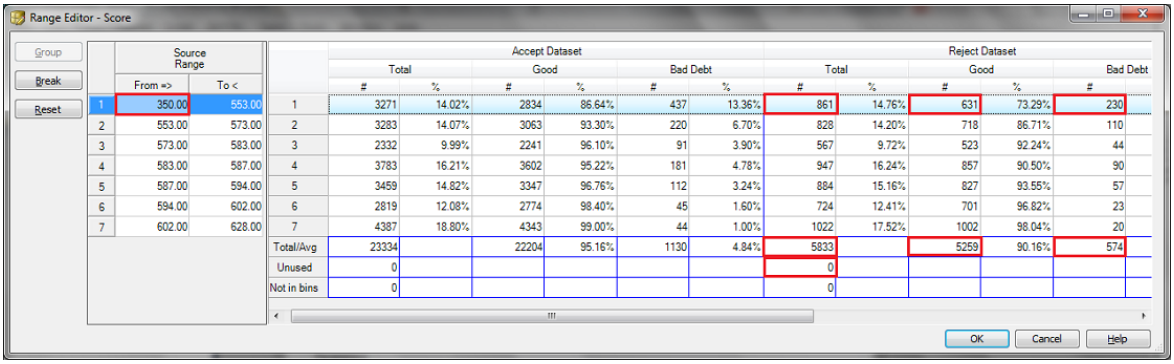
a) if this score falls within the binned score range for the Accept Dataset (the *Source Range* in the left panel), depending on the specific bin that this score falls into and the proportion of Goods and Bads in this bin, a given outlier record will be assigned to Goods or Bads, respectively.

b) if this score is outside the binned score range for the Accept Dataset, the outlier records in the Reject Dataset will be listed as *Unused* and will not be included in the Combined dataset, just as in the case of null score for outliers described below. In the above example, this will happen, for instance, if you set the *Score for outliers* to 350, which is below the minimum score of 483 (the lower boundary of the overall score range).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

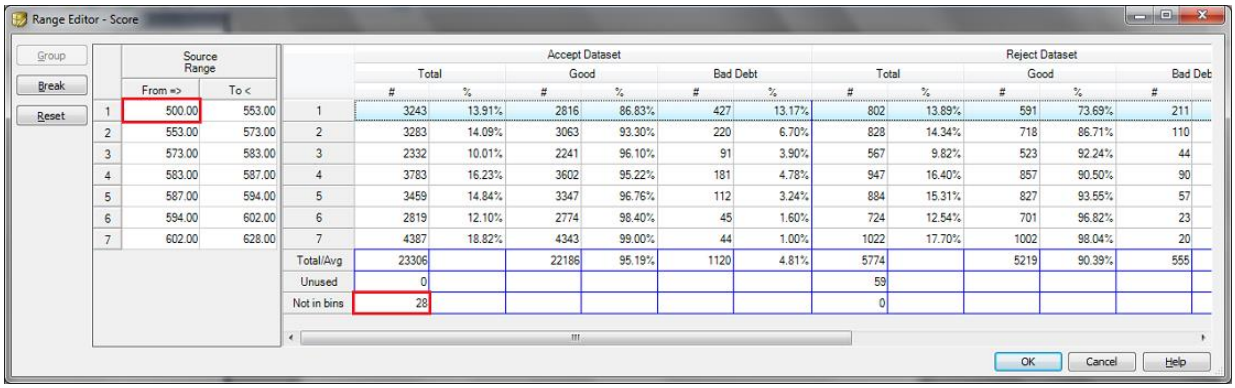


If you do want to use them or look at the Good/Bad rates with these records included, just set the lower boundary of the overall score range in the Range Editor window to 350 or lower. Then the outliers will fall into the first interval. In the Range Editor example below, after changing 483 to 350, they fall into the interval [350,553), and the numbers in the highlighted cells change accordingly. The same happens for the Combined Dataset. The case of the custom outlier score that is higher than the maximum score range for the Accept Dataset is similar.



	From =>	To <	Accept Total #	Accept Total %	Accept Good #	Accept Good %	Accept Bad Debt #	Accept Bad Debt %	Reject Total #	Reject Total %	Reject Good #	Reject Good %	Reject Bad Debt #
1	350.00	553.00	3271	14.02%	2834	86.64%	437	13.36%	861	14.76%	631	73.29%	230
2	553.00	573.00	3283	14.07%	3063	93.30%	220	6.70%	828	14.20%	718	86.71%	110
3	573.00	583.00	2332	9.99%	2241	96.10%	91	3.90%	567	9.72%	523	92.24%	44
4	583.00	587.00	3783	16.21%	3602	95.22%	181	4.78%	947	16.24%	857	90.50%	90
5	587.00	594.00	3459	14.82%	3347	96.76%	112	3.24%	884	15.16%	827	93.55%	57
6	594.00	602.00	2819	12.08%	2774	98.40%	45	1.60%	724	12.41%	701	96.82%	23
7	602.00	628.00	4387	18.80%	4343	99.00%	44	1.00%	1022	17.52%	1002	98.04%	20
Total/Avg			23334		22204	95.16%	1130	4.84%	5833		5259	90.16%	574
Unused			0						0				
Not in bins			0						0				

If you set the lower limit in the Range Editor to be greater than Min Score specified in the Method and Parameters page of the wizard, the Accept Dataset records whose score is below the new lower boundary will be shown as ***Not in bins***. If we change the lower limit in the same Range Editor example from 483 to 500, we will see that there are now 28 records in the Accept Dataset that are "Not in bins", since their score was in the excluded range [483, 500).



	From =>	To <	Accept Total #	Accept Total %	Accept Good #	Accept Good %	Accept Bad Debt #	Accept Bad Debt %	Reject Total #	Reject Total %	Reject Good #	Reject Good %	Reject Bad Debt #
1	500.00	553.00	3243	13.91%	2816	86.83%	427	13.17%	802	13.89%	591	73.69%	211
2	553.00	573.00	3283	14.09%	3063	93.30%	220	6.70%	828	14.34%	718	86.71%	110
3	573.00	583.00	2332	10.01%	2241	96.10%	91	3.90%	567	9.82%	523	92.24%	44
4	583.00	587.00	3783	16.23%	3602	95.22%	181	4.78%	947	16.40%	857	90.50%	90
5	587.00	594.00	3459	14.84%	3347	96.76%	112	3.24%	884	15.31%	827	93.55%	57
6	594.00	602.00	2819	12.10%	2774	98.40%	45	1.60%	724	12.54%	701	96.82%	23
7	602.00	628.00	4387	18.82%	4343	99.00%	44	1.00%	1022	17.70%	1002	98.04%	20
Total/Avg			23306		22186	95.19%	1120	4.81%	5774		5219	90.39%	555
Unused			0						59				
Not in bins			28						0				

If you want to return to the default source ranges, just click the **Reset** button to set the original range boundaries.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

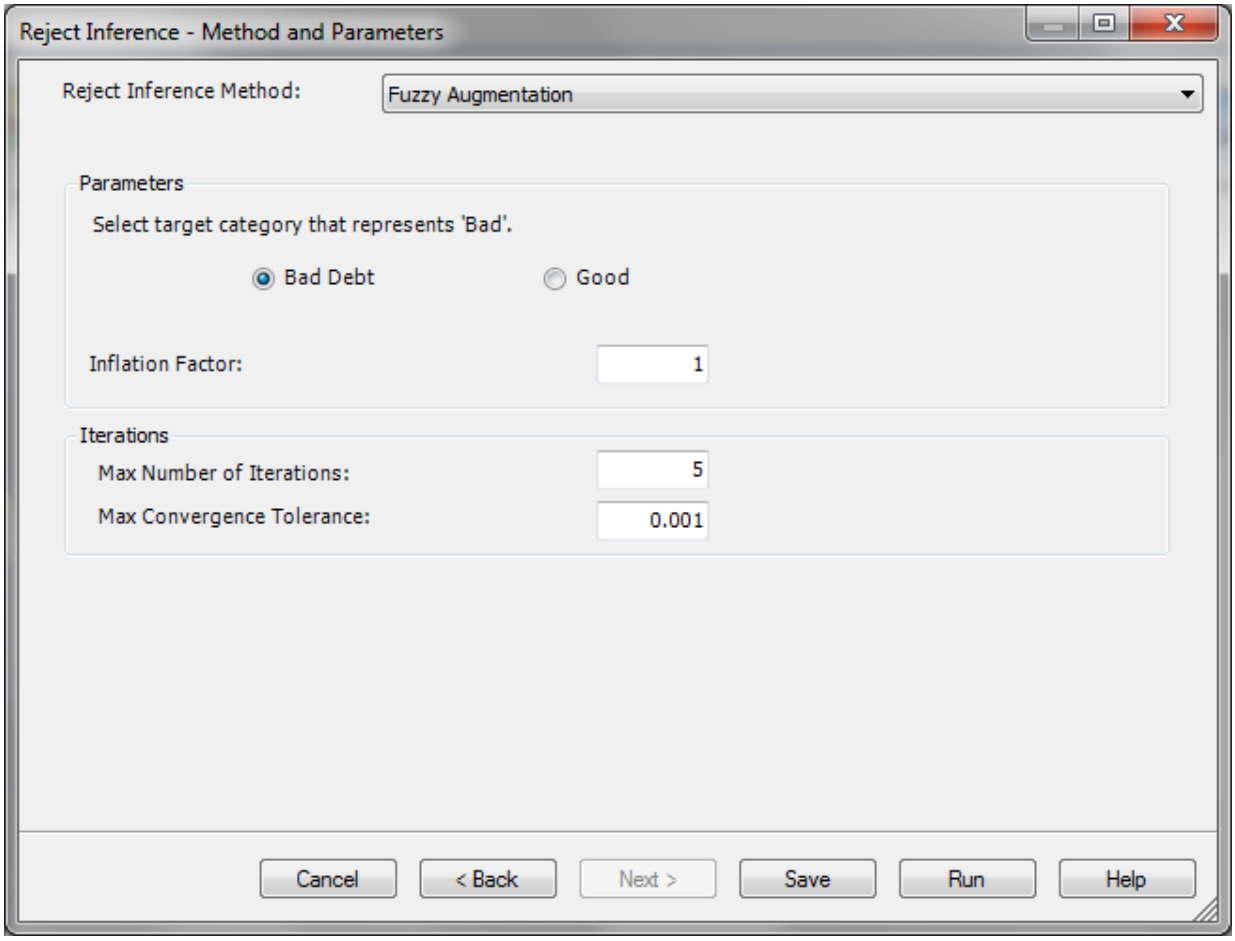
After clicking **Finish** in the Methods and Parameters page, the wizard produces the Reject Inference object with child objects including the combined datasets - Accept Dataset merged with the Reject Dataset with inferred status - and the models built on them at each iteration. See the Reject Inference Results topic for information about the reject inference output.

Fuzzy Augmentation

The general description of the Fuzzy Augmentation method is provided in the topic Reject Inference: Overview. First, the Reject Dataset is scored with the input **logistic regression** model. The next stage is weighting rejected goods and bads with the respective probabilities and then combining the Rejects and Accepts, together with the associated weights, and adjusting for the approval probability.

The Inflation Factor is the coefficient to inflate the weight of the "Bad" status in order to be conservative with assigning the "Good" status to known rejects.

The Iterations parameters have the same meaning as in the methods above.



See also:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Reject Inference: Overview
Reject Inference Results
About Scorecards

Reject Inference Results

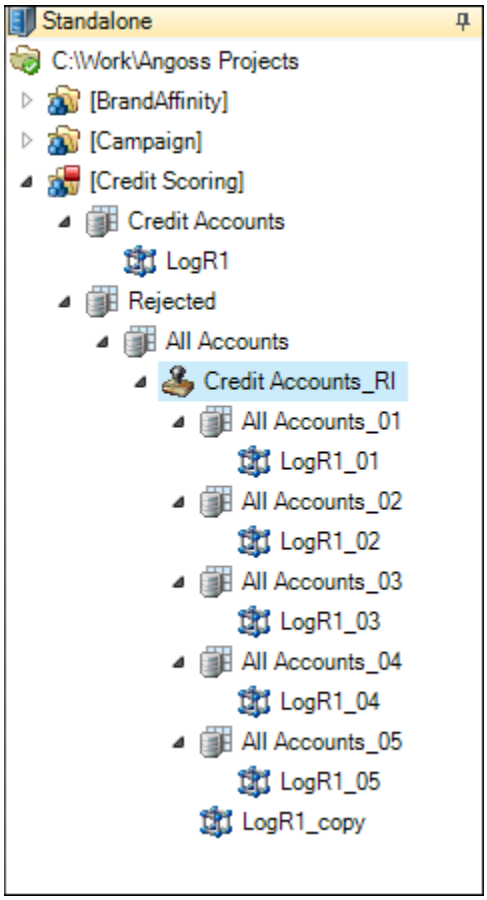
The Reject Inference Wizard produces a series of datasets resulting from combining the Accept dataset and the Reject dataset with inferred status at each iteration, as well as models based on the datasets.

The main result is a dataset representing the whole population including accepted applicants and rejected applicants with appropriately inferred Good/Bad status, which you can use for developing the overall scorecard. Examine the results to choose the output of the best iteration and use it for building the final scorecard.

Here is an example output of a Reject Inference using the Proportional Assignment method with 5 iterations. The process followed by the wizard is illustrated by the diagram in the topic Reject Inference: Overview. In the example screenshot below, *Credit Accounts_RI* is the resulting Reject Inference report; the *All Applicants* dataset is the combined dataset of approved accounts and rejected accounts with inferred Good/Bad status. The *All Applicants_N* datasets are the combined datasets of approved and rejected accounts with Good/Bad status inferred at iteration *N*.

As noted in the Reject Inference Wizard topic, the regression models built in the iterative process of Reject Inference ignore missing values by default rather than substitute median. The wizard makes a copy of the base model and changes the missing value attribute to *ignore* for all scalar variables. The wizard makes a copy of the base model and changes the missing value attribute to *ignore* for all scalar variables. The copy of the base model is shown in the project as <*Base model name*>*_copy*. For example, in the screenshot below, it is *LogR1_copy* at the bottom.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The Reject Inference overview report is displayed after the process has finished. It summarizes the settings specified in the Reject Inference Wizard, provides the training statistics, the details of the final target category distribution in the resulting dataset, and iteration details.

The first three sections of the report: Input Details, Training Details, and Scoring Options, summarize the inputs specified by the user in the Reject Inference Wizard. The Training Details section also shows the number of iterations and the Root Mean Square Error at the final iteration.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Input Details	
Source Table:	Credit Accounts (23334 records)
Reject Table:	Rejected (5833 records)
Base Model:	LogR1
Base Model Copy:	LogR1_copy

Training Details	
Method:	HardCutoff
Iterations:	5
Root Mean Square Error:	0.1466
Stop Point:	Iterations
Cutoff Score:	555

Scoring Options	
Target outcome:	'Good'
Base Points:	600
Points to Double Odds:	20
Base Odds:	50
Score for Outliers:	Null
Round Points to Integer:	Yes

The remaining sections show the details of the Final Distribution and each iteration.

The Final Distribution table shows the Good and Bad rates (frequency and percentage) in the final Reject Dataset and Combined Dataset, as well as the original Accept Dataset.

For each iteration, the iteration details include

- the names of the regression model and dataset;
- the Root Mean Square Error (RMSE) computed at the given iteration; see the formula in the Reject Inference Wizard topic;
- the number of records used in the Accept Dataset and the Reject Dataset;
- the number of unused records (if any); this is the number of records in the Reject Dataset with values that are not observed in the Accept Dataset and get null scores or custom scores outside of the automatically calculated score range; see the Reject Inference Wizard topic for details

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Final Distribution

	Accepts		Rejects		Combined	
	Total	%	Total	%	Total	%
Good	22204	95.2%	4173	72.2%	26377	90.6%
Bad Debt	1130	4.8%	1609	27.8%	2739	9.4%

Iteration 1

Model	Dataset	RMSE	Accept records used	Reject records used	Reject records unused
LogR1_01	All Accounts_01	0.3988	23334	5782	51

Iteration 2

Model	Dataset	RMSE	Accept records used	Reject records used	Reject records unused
LogR1_02	All Accounts_02	0.2343	23334	5782	51

Iteration 3

Model	Dataset	RMSE	Accept records used	Reject records used	Reject records unused
LogR1_03	All Accounts_03	0.1836	23334	5782	51

In addition, the iteration details for the Parceling method also contain Good and Bad rates - frequency (#) and percentage (%) - in the Accept, Reject, and Combined datasets **for all score ranges** at each iteration:

Iteration 1

Model	Dataset	RMSE	Accept records used	Reject records used	Reject records unused
LogR1_01	All Accounts_01	0.1147	23334	5782	51

		Accept Dataset						Reject Dataset						Combined Dataset					
		Total		Good		Bad Debt		Total		Good		Bad Debt		Total		Good		Bad Debt	
Bin	Range	#	%	#	%	#	%	#	%	#	%	#	%	#	%	#	%	#	%
1	[Score] >= 483 AND [Score] < 553	3271	14.02%	2834	86.64%	437	13.36%	810	14.01%	594	73.33%	216	26.67%	4081	14.02%	3428	84.00%	653	16.00%
2	[Score] >= 553 AND [Score] < 573	3283	14.07%	3063	93.30%	220	6.70%	828	14.32%	718	86.71%	110	13.29%	4111	14.12%	3781	91.97%	330	8.03%
3	[Score] >= 573 AND [Score] < 583	2332	9.99%	2241	96.10%	91	3.90%	567	9.81%	523	92.24%	44	7.76%	2899	9.96%	2764	95.34%	135	4.66%
4	[Score] >= 583 AND	3783	16.21%	3602	95.22%	181	4.78%	947	16.38%	857	90.50%	90	9.50%	4730	16.25%	4459	94.27%	271	5.73%

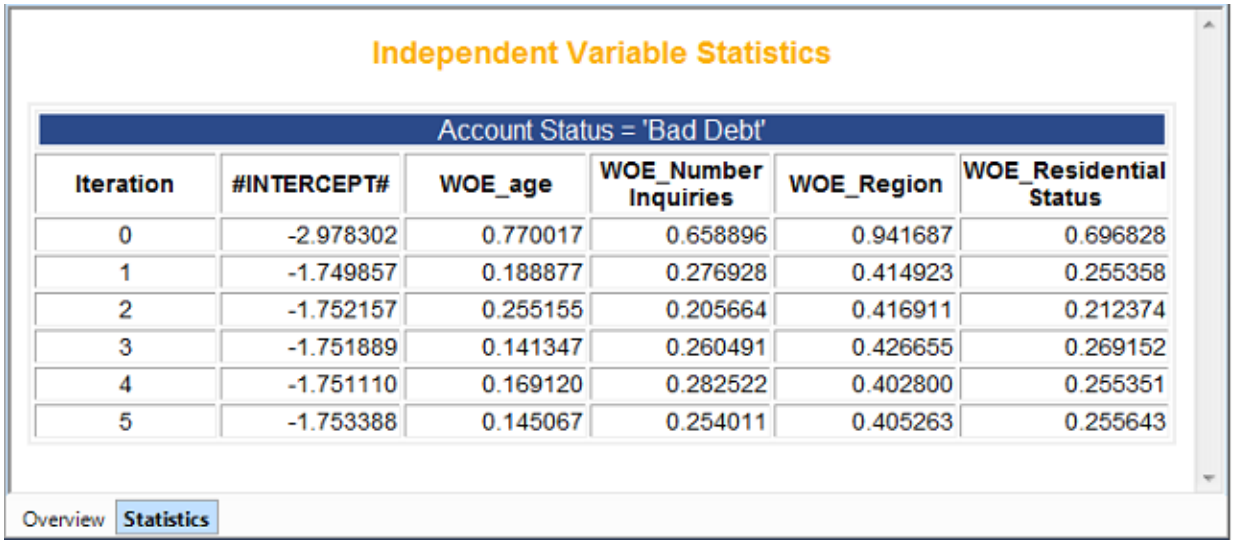
The tables with Good and Bad rates for each iteration also show the totals summary, including the numbers of unused records and records not in bins for the Reject dataset (if any). See the Reject Inference Wizard topic for the description of such records.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

		Accept Dataset						Reject Dataset						Combined Dataset					
		Total		Good		Bad Debt		Total		Good		Bad Debt		Total		Good		Bad Debt	
Bin	Range	#	%	#	%	#	%	#	%	#	%	#	%	#	%	#	%	#	%
Total*		23334		22204	95.16%	1130	4.84%	5782		5222	90.31%	560	9.69%	29116		27426	94.20%	1690	5.80%
*																			
Records unused		0						51						51					
Records not in bins		0						0						0					

Statistics

The Statistics tab of the Reject Inference report shows the independent variable statistics at each iteration. These are model parameters for the logistic regression models created at each iteration.



Independent Variable Statistics

Account Status = 'Bad Debt'

Iteration	#INTERCEPT#	WOE_age	WOE_Number Inquiries	WOE_Region	WOE_Residential Status
0	-2.978302	0.770017	0.658896	0.941687	0.696828
1	-1.749857	0.188877	0.276928	0.414923	0.255358
2	-1.752157	0.255155	0.205664	0.416911	0.212374
3	-1.751889	0.141347	0.260491	0.426655	0.269152
4	-1.751110	0.169120	0.282522	0.402800	0.255351
5	-1.753388	0.145067	0.254011	0.405263	0.255643

Overview | Statistics

Detailed description of the Logistic Regression model parameters and other statistics can be found in the Logistic Regression Tutorial.

See also:

- Reject Inference: Overview
- Reject Inference Wizard
- About Scorecards

Sklearn XGBoost Model

Overview

The **Sklearn XGB** node in the **Model** palette of Knowledge Studio allows you to build models using the Python implementation of XGBoost (Extreme Gradient Boost) algorithm. It uses gradient boosting functions for classification and regression in **`scikit-learn`** - a popular Python library

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

for machine learning. For details, see the scikit-learn documentation for Gradient Boosting Classifier and Gradient Boosting Regressor (Copyright © 2007 - 2021, scikit-learn developers, BSD License).

The node wizard provides a convenient graphical interface to specify the options and parameters for the required functions in the `scikit-learn (sklearn)` library. The wizard automatically translates the inputs into Python code, which can be previewed before running the node. The code then runs in a built-in Python environment, loading all required packages and their dependencies. The input dataset is converted from the Altair KDD format to pandas dataframes in the process.

After training, the model can be validated using the Model Validation node and scored on new data using the Scoring node.

Building an Sklearn XGB Model

An **Sklearn XGB** model requires a single dataset as input. An **Sklearn XGB** node can be added to the workflow in one of the following ways:

- Drag a connection arrow from the input dataset node to any area on the Workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Model --> Sklearn XGB** as in the example screenshot below. A new **Sklearn XGB** node will be created, and the wizard will open.

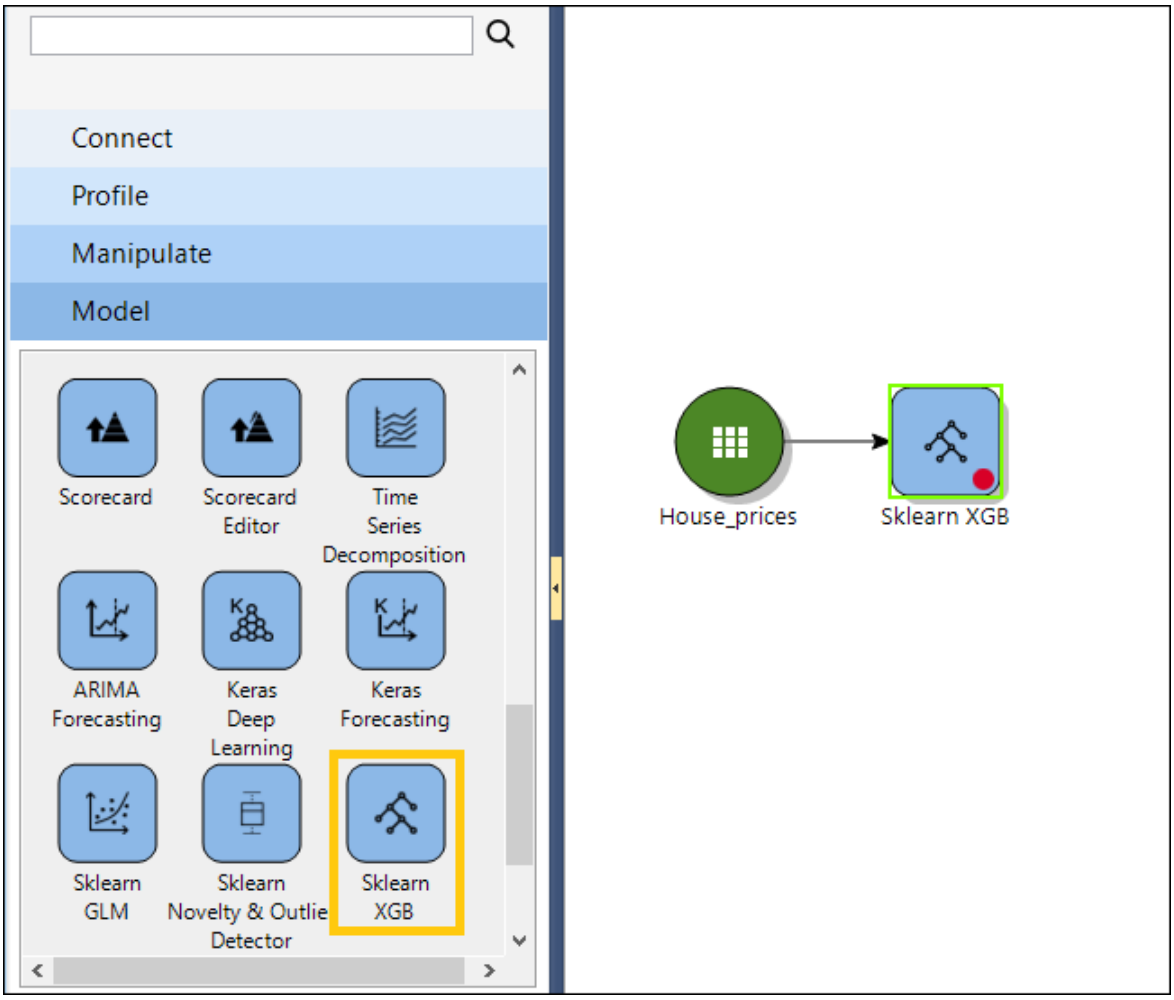
  OR

- Right-click on the Workflow canvas and select **Model --> Sklearn XGB** from the context menu. Connect the input dataset. Optionally, connect another predictive model node as a base model to use as a template.

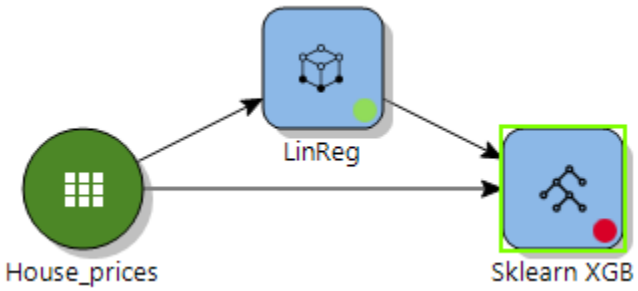
  OR

- Open the **Model** palette and drag the **Sklearn XGB** node onto the workflow canvas. Connect the input dataset. Optionally, connect another predictive model node as a base model to use as a template.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



In the following example, an existing Linear Regression model is used as a base model for the new Sklearn XGB model.



Double-click the Sklearn XGB node or right-click on it and select **Modify** to open the wizard.

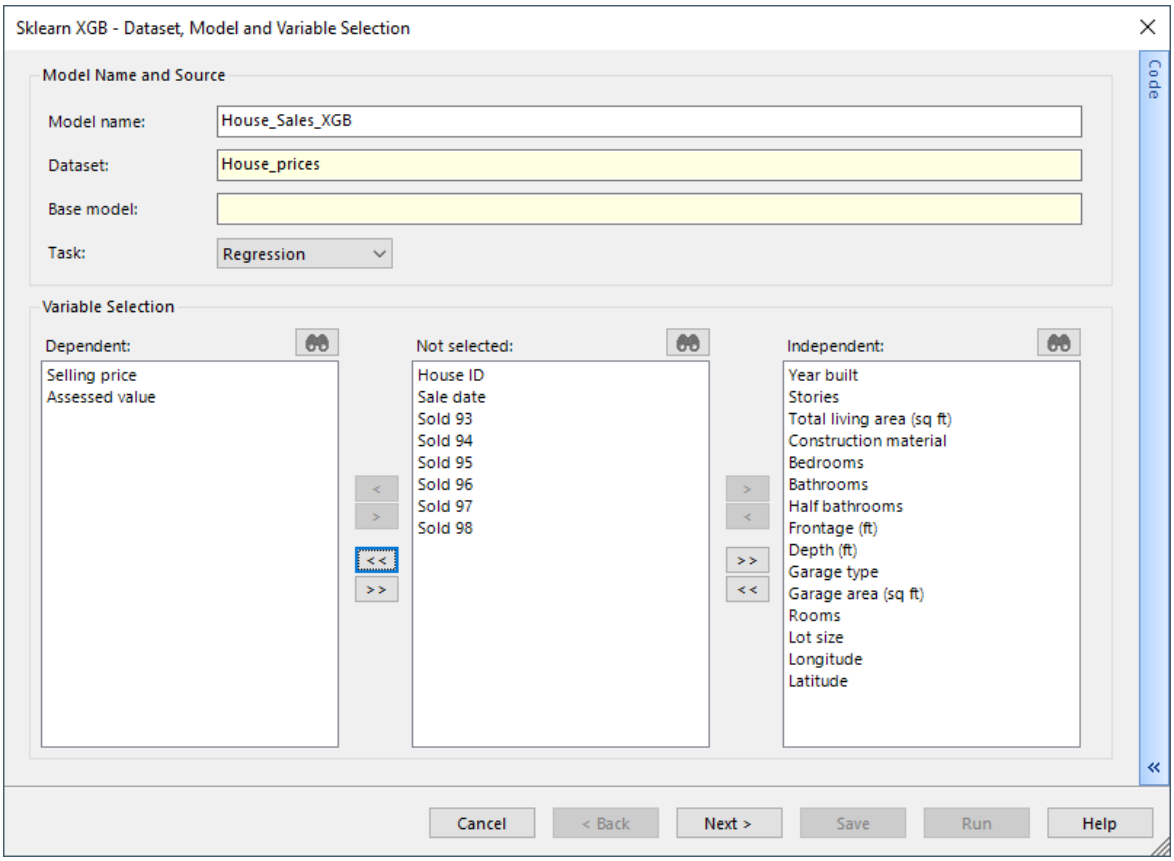
In the first page of the wizard:

- Specify the model name.
- Specify the Task type: *Regression* or *Classification*

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Select at least one dependent variable. If multiple dependent variables are selected, all of them must be either continuous or binary. Multiple binary dependent variables must be of the same data type.
- Select the independent variables.
  - To select or deselect variables, use the arrow buttons or double-click on the variable names. Advanced field finding and multi-selection capabilities are also available via the Field Finder dialog, which can be opened by clicking the binoculars button.

Note: If a Base model is used, the dependent variable and independent variables are automatically inherited from the Base model.



Click **Next**.

In the Preprocessing Parameters page, specify the following parameters for imputation and scaling. See the scikit-learn documentation for Gradient Boosting Classifier and Gradient Boosting Regressor for the detailed description of the function parameters.

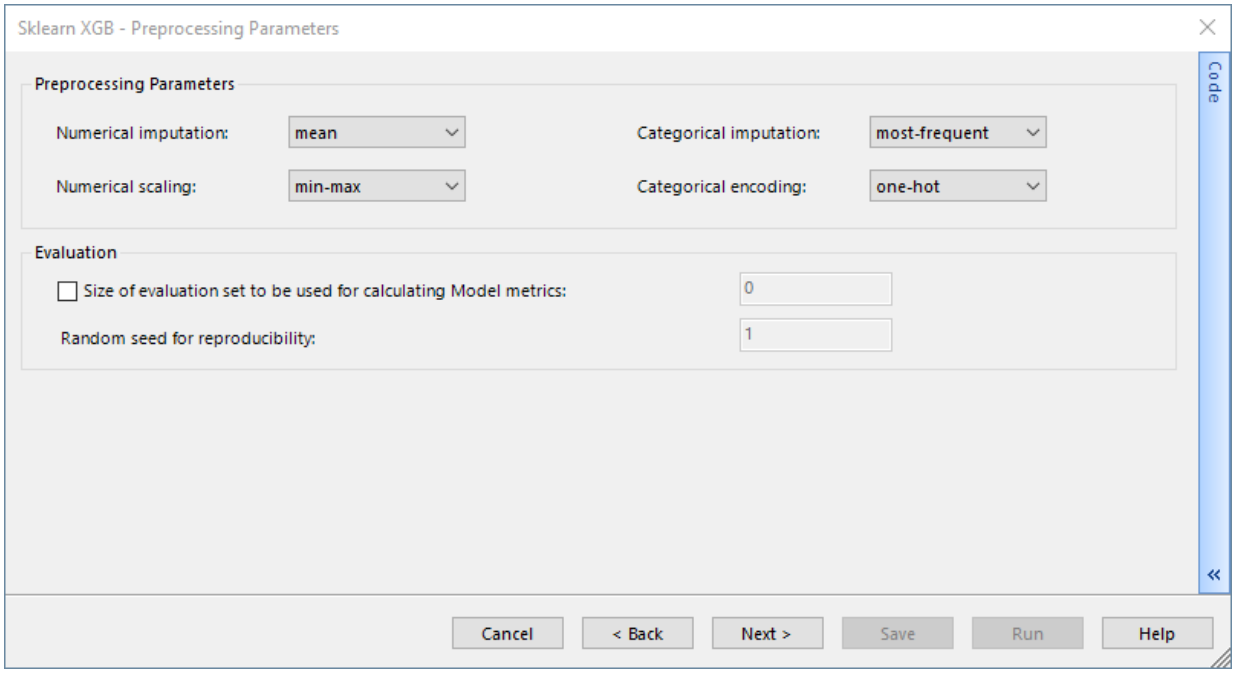
- Specify the *imputation* method for numerical variables. Available methods are: *mean, median, most-frequent, zero, min, max*. The default is *mean*.
- Specify the *scaling* method for numerical variables. Available methods are: *min-max* and *standard*. The default limit is *min-max*.

For categorical variables (classification problems), the imputation method is set to *most-frequent*, and the encoding method is set to *one-hot* encoder.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Imputation is applied to missing and NaN (undefined) values.

- Specify the size of the evaluation sample to be used for calculating model performance metrics during training. Select the checkbox and enter a number between 0 and 1 to indicate the sample size with respect to the training dataset.
    - Note: For Classification tasks, when splitting the dataset for model evaluation, each value of the dependent variable must occur in both partitions. If the dependent variable contains a value that occurs only once in the entire dataset, the following error may occur: "*ValueError: The least populated class in y has only 1 member, which is too few. The minimum number of groups for any class cannot be less than 2*." In such cases, disable evaluation by deselecting the checkbox in the Preprocessing Parameters page of the wizard.



Click **Next**.

In the Model Parameters page, specify the following model parameters. See the scikit-learn documentation for Gradient Boosting Classifier and Gradient Boosting Regressor for the detailed description of the function parameters.

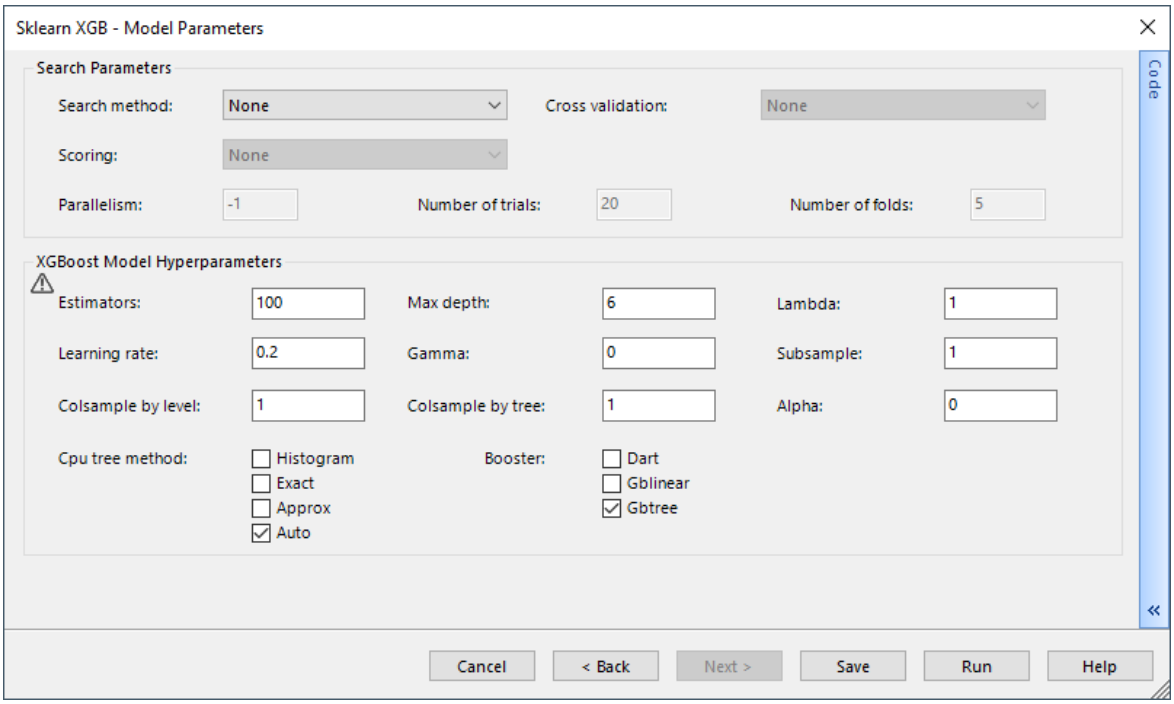
- Specify the **Search Parameters**:
    - The *Search method* (None, *GridSearchCV,* or *OptunaSearchCV*). The default is None.
    - The *Cross validation* option (None, *KFold*, or *StratifiedKFold*). The default is None. Not applicable if Search method is None.
    - The *Scoring* option
        - For Regression tasks, the Scoring options are: None, *Neg mean absolute error*, *Neg mean squared error*, or *Neg mean squared log error*. The default is None.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

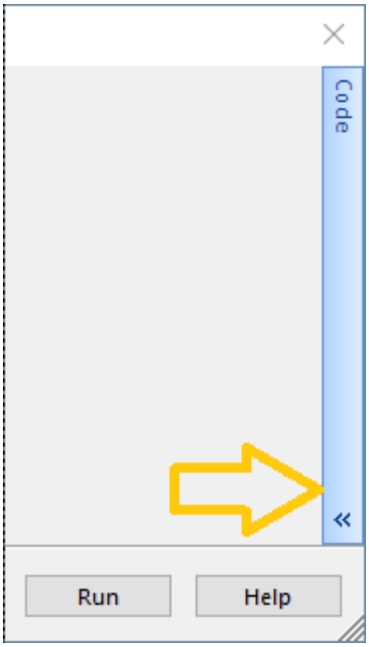
        - For Classification tasks, the Scoring options are: None, *Accuracy*, *F1 macro, F1 micro, Precision macro, Precision micro, Recall macro,* or *Recall micro*. The default is None.
    - The *Parallelism* parameter. The default is -1.
    - The *Number of trials.* The default is 20.
    - The *Number of folds*. The default is 5.
- Specify the **XGBoost Model Hyperparameters**:
    - The number of *estimators* in the ensemble. The default is 100.
    - The *maximum depth* of the tree estimators in the ensemble. The default limit is 6 levels.
    - The *learning rate*. The default is 0.2.
    - The *Subsample* size. Specify a positive number between 0 and 1. The default is 1.
    - The *alpha*, *lambda*, and *gamma* parameter values (the defaults are 0, 1, and 0, respectively)
    - *Colsample by level* and *Colsample by tree* (the default is 1 for both)
    - *Cpu tree method* (one or more, depending on the selected *Search method*)
    - *Booster* (one or more, depending on the selected Search method)

    When specifying the hyperparameter values:
        - If the *Search method* is not specified (*None* is selected), you can specify a single value per parameter.
        - If the selected Search method is *GridSearchCV*, you can specify
            - -- a single value per parameter (static during the search).
            - -- three slash-separated values for interval creation (start, end, step). For example, **100/120/5** will create the interval values to be tested [100, 110, 115] (last value not included).
            - -- multiple *Cpu tree methods* and *Booster* choices.
        - If the selected Search method is *OptunaSearchCV*, you can specify
            - -- a single value per parameter (static during the search)
            - -- two comma-separated values for interval search (lower bound, upper bound).
            - -- multiple *Cpu tree methods* and *Booster* choices.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

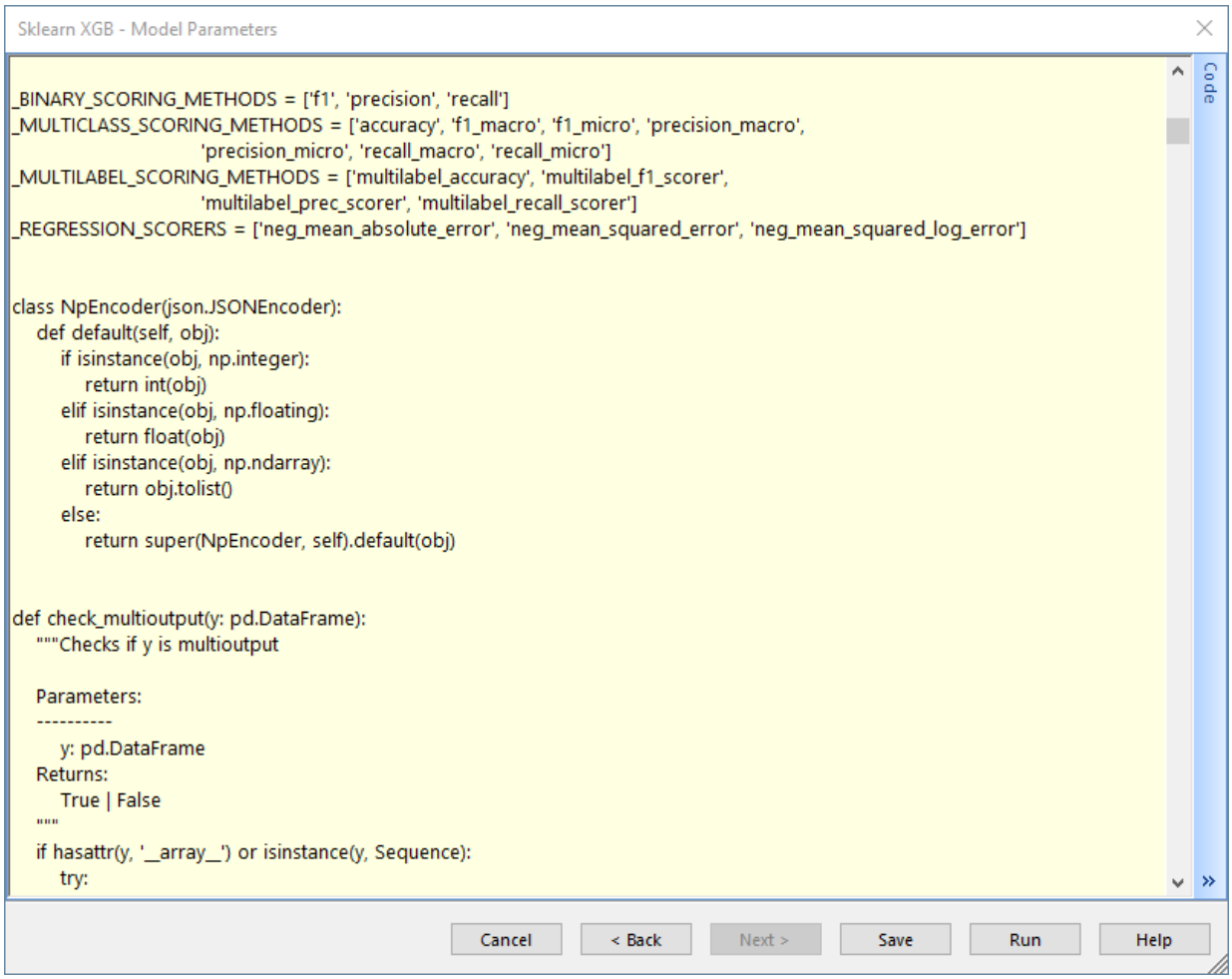


To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog.



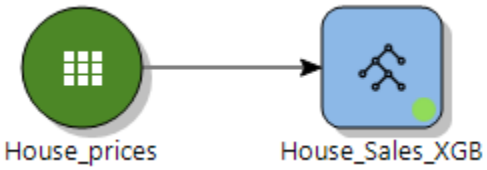
This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



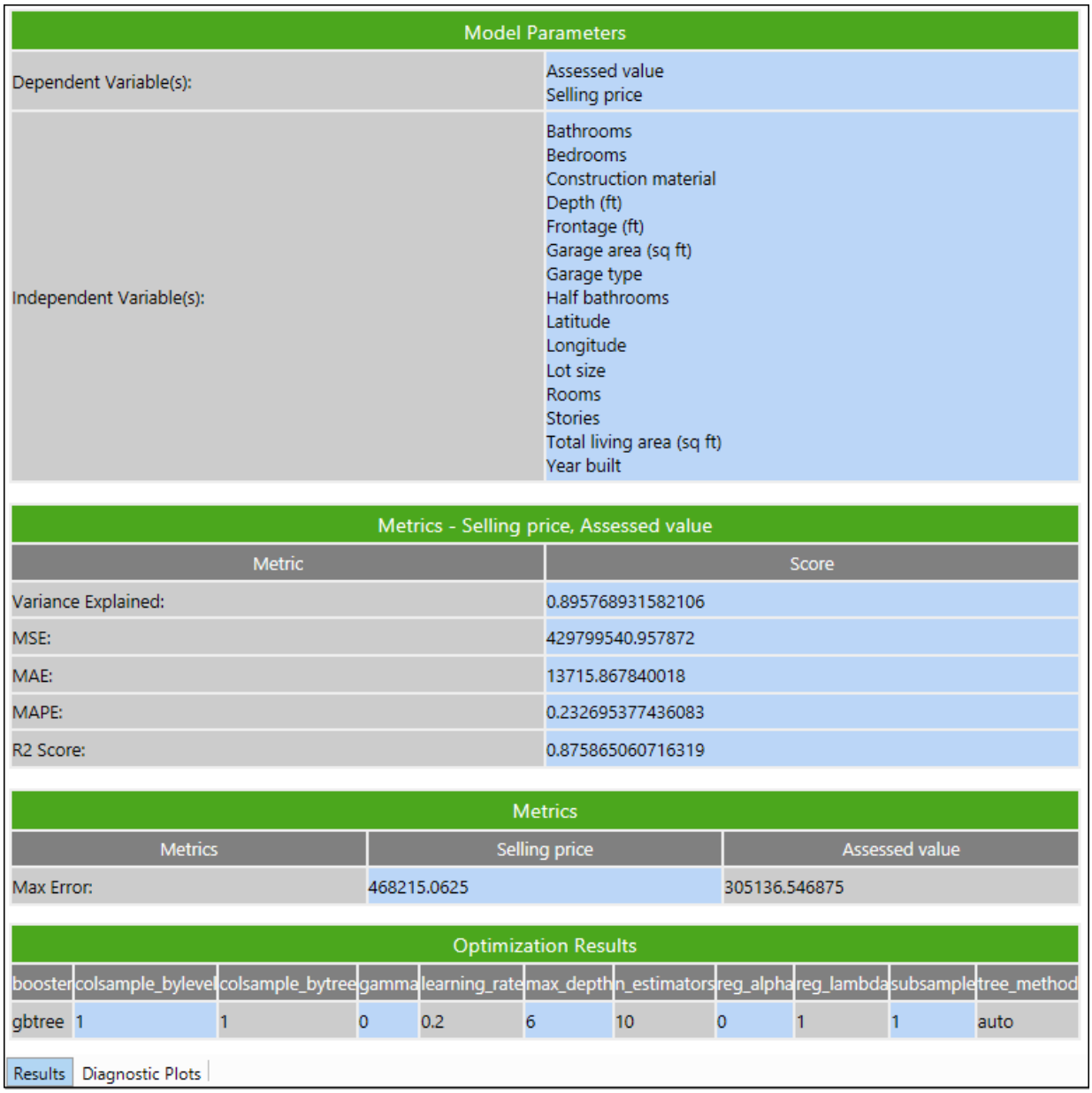
Click **Save** to save the settings or click **Run** to save the settings and run the process immediately. The input dataset is converted to a Pandas dataframe, and the required functions of `sklearn` and other libraries are applied. Then the Gradient Boosting Classifier or Regressor functions with specified parameters are applied.

After running the node, double-click the model node to view the results.



The model training report displays the list of the dependent variable(s) and independent variables, and the model accuracy and performance metrics.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Model Parameters	
Dependent Variable(s):	Assessed value Selling price
Independent Variable(s):	Bathrooms Bedrooms Construction material Depth (ft) Frontage (ft) Garage area (sq ft) Garage type Half bathrooms Latitude Longitude Lot size Rooms Stories Total living area (sq ft) Year built

Metrics - Selling price, Assessed value	
Metric	Score
Variance Explained:	0.895768931582106
MSE:	429799540.957872
MAE:	13715.867840018
MAPE:	0.232695377436083
R2 Score:	0.875865060716319

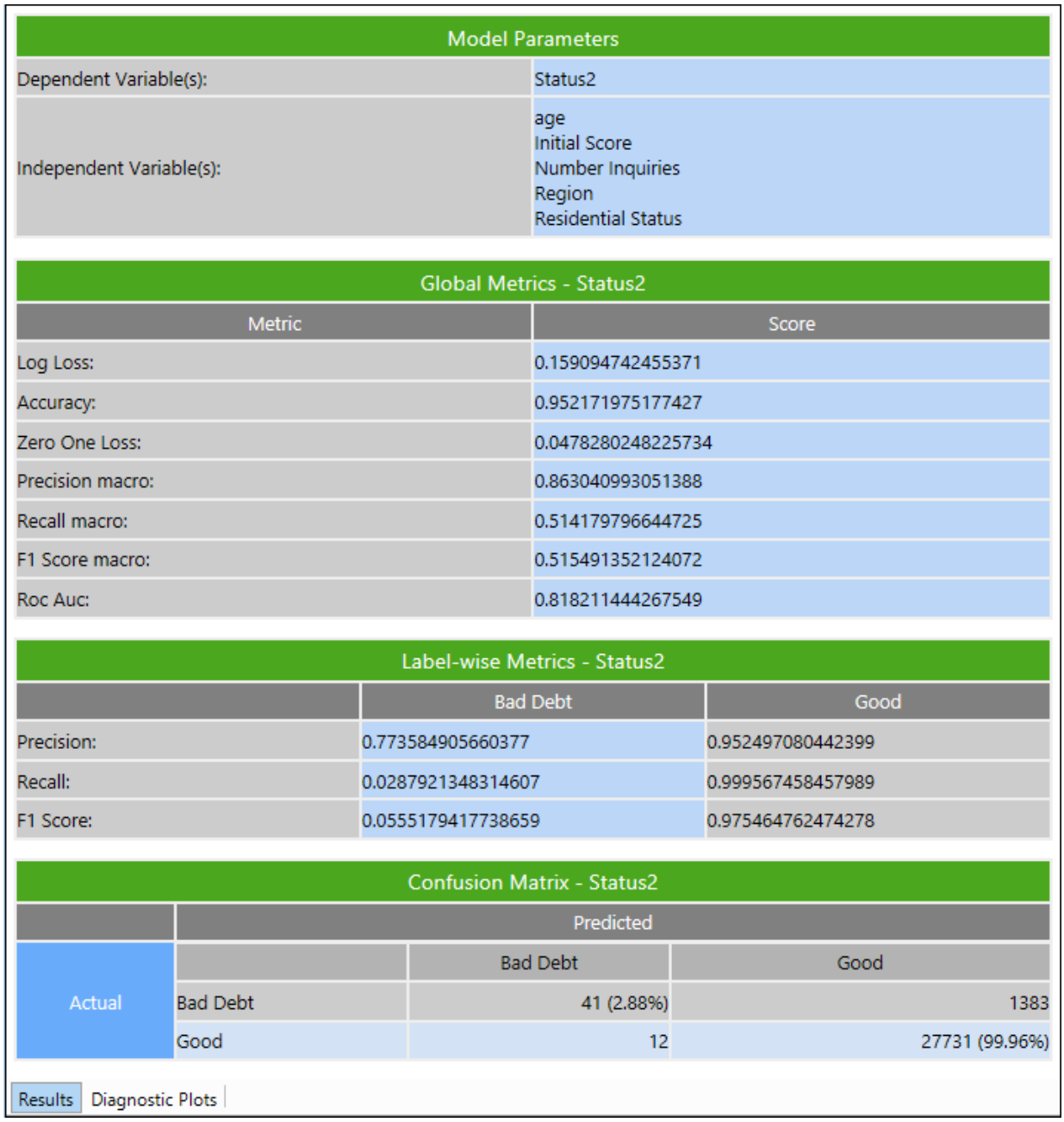
Metrics		
Metrics	Selling price	Assessed value
Max Error:	468215.0625	305136.546875

Optimization Results										
booster	colsample_bylevel	colsample_bytree	gamma	learning_rate	max_depth	n_estimators	reg_alpha	reg_lambda	subsample	tree_method
gbtree	1	1	0	0.2	6	10	0	1	1	auto

Results | Diagnostic Plots

In the case of a categorical dependent variable, the Results report shows Confusion Matrices for the model performance on the training and test partitions. Here is an example of the Results report for an XGB model with a single binary dependent variable:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Model Parameters	
Dependent Variable(s):	Status2
Independent Variable(s):	age Initial Score Number Inquiries Region Residential Status

Global Metrics - Status2	
Metric	Score
Log Loss:	0.159094742455371
Accuracy:	0.952171975177427
Zero One Loss:	0.0478280248225734
Precision macro:	0.863040993051388
Recall macro:	0.514179796644725
F1 Score macro:	0.515491352124072
Roc Auc:	0.818211444267549

Label-wise Metrics - Status2		
	Bad Debt	Good
Precision:	0.773584905660377	0.952497080442399
Recall:	0.0287921348314607	0.999567458457989
F1 Score:	0.0555179417738659	0.975464762474278

Confusion Matrix - Status2			
	Predicted		
		Bad Debt	Good
Actual	Bad Debt	41 (2.88%)	1383
	Good	12	27731 (99.96%)

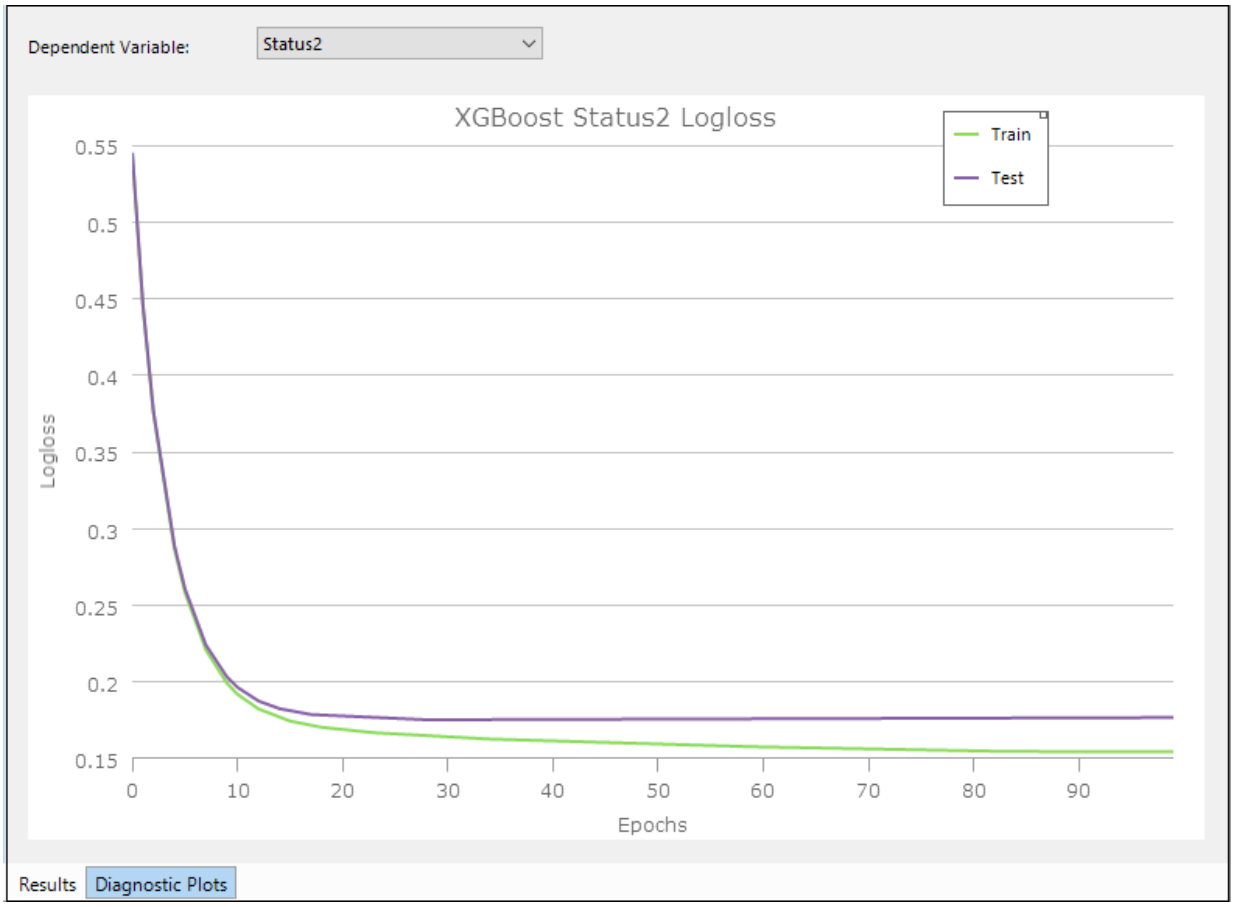
Results | Diagnostic Plots

The **Diagnostic Plots** tab shows diagnostic plots that depend on the type of the dependent variable. If the model has multiple dependent variables, select the desired variable from the drop-down list at the top of the Plots view.

For a discrete dependent variable, it shows the *Logloss* plot for both Training and Testing partitions for all epochs. The X axis shows the number of epochs, while the Y axis shows the Classification error rate on the left chart and Logloss on the right chart.

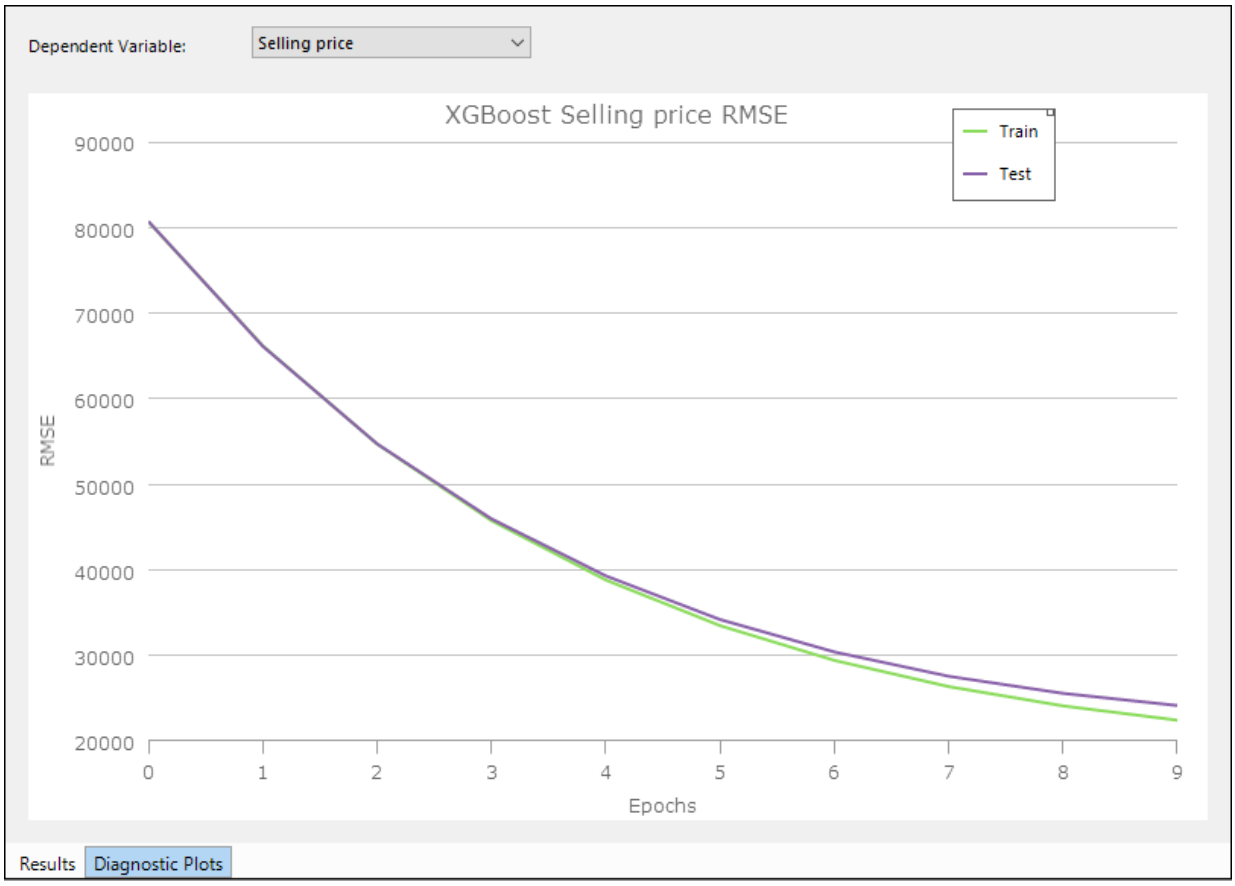
> *Note*: Diagnostic plots and confusion matrix are not supported in the case of multiple categorical dependent variables.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



For a continuous dependent variable, the **Diagnostic Plots** tab shows the Root Mean Square Error (RMSE) on the Training and Test partitions. The X axis shows the epochs.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

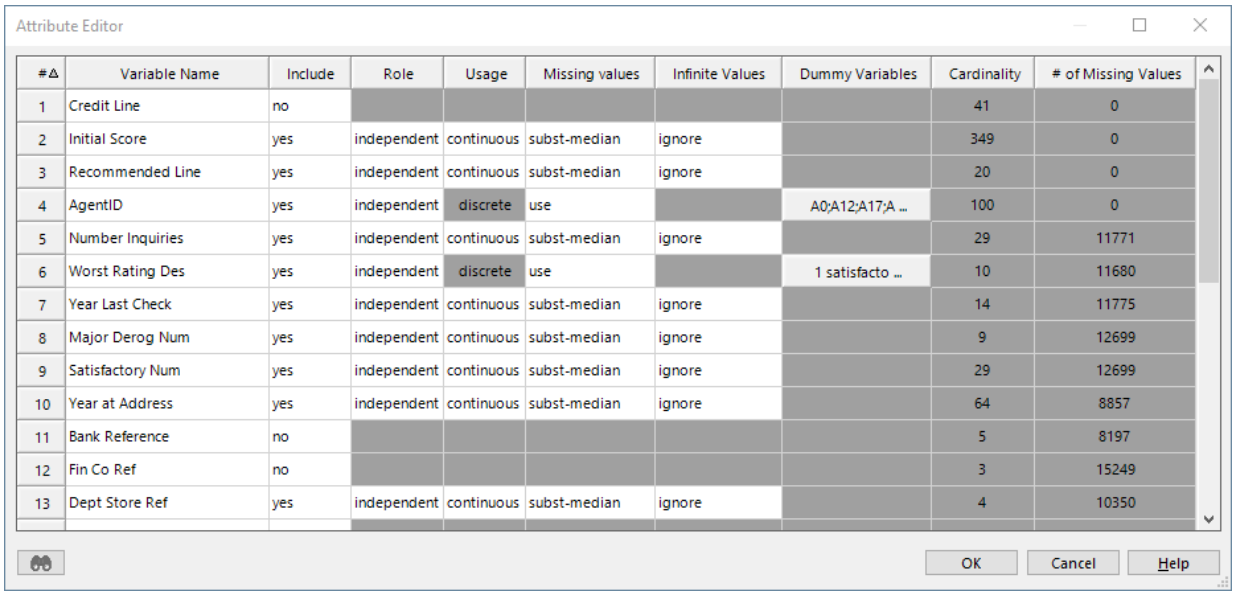


After training, the model can be used for scoring new data using the Scoring node. If the model is scored on the training dataset or another dataset with known outcomes of the dependent variable, the resulting dataset can be used to evaluate the model performance with Model Analyzer.

Model Attribute Editor

The Attribute Editor for predictive models and cluster models allows you to modify variable attributes for the predictive model being inserted. The variables can be modified during the insert wizard. The attribute editor dialog is the same for all model types. To open the attribute editor, click the Advanced button in the Variable Selection step of the Model Wizard.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



#△	Variable Name	Include	Role	Usage	Missing values	Infinite Values	Dummy Variables	Cardinality	# of Missing Values
1	Credit Line	no						41	0
2	Initial Score	yes	independent	continuous	subst-median	ignore		349	0
3	Recommended Line	yes	independent	continuous	subst-median	ignore		20	0
4	AgentID	yes	independent	discrete	use		A0;A12;A17;A ...	100	0
5	Number Inquiries	yes	independent	continuous	subst-median	ignore		29	11771
6	Worst Rating Des	yes	independent	discrete	use		1 satisfacto ...	10	11680
7	Year Last Check	yes	independent	continuous	subst-median	ignore		14	11775
8	Major Derog Num	yes	independent	continuous	subst-median	ignore		9	12699
9	Satisfactory Num	yes	independent	continuous	subst-median	ignore		29	12699
10	Year at Address	yes	independent	continuous	subst-median	ignore		64	8857
11	Bank Reference	no						5	8197
12	Fin Co Ref	no						3	15249
13	Dept Store Ref	yes	independent	continuous	subst-median	ignore		4	10350

Variable Name The name of the variable.

Include Determines if a variable must be included in the model.

Role Determines the role of the variable in the analysis.

Dependent: In the case of Predictive Models, indicates that this variable is the dependent (target) variable that you want to explore. A model can have only one dependent variable (DV). Not applicable in the case of Cluster Models.

Independent: Indicates that this variable is an independent variable (IV) whose predictive value with respect to the dependent variable is to be determined.

Weight: Indicates that the variable is set as weight. The weight option is available only for nonnegative numeric variables. For details, see Weight in Predictive Models.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Usage

Depending on the selected option, values in a given variable can be left unchanged, scaled (normalized) in a certain way, or the variable may be transformed into a set of dummy variables. Note that the meaning of the continuous normalization type depends on the model type.

Continuous: This is the default normalization type for numeric variables.

In Logistic and Linear Regression models, continuous variables are not normalized.

In Multi-Level Neural Network models (MLN) and Cluster Analysis models, continuous variables are normalized as follows:

Normalized Value = (Original Value - Mean) / StdDev

Note: By default, the mean and the standard deviation used for normalization are calculated **before** the substitution of missing and infinite values described below. If any of the variables to be included in your model contain missing and/or infinite values, and if you prefer to substitute them prior to normalization, select the option "**Missing and infinite value substitution before normalization**" in this wizard page. Alternatively, you can use the Variable Transformations node to perform the necessary substitutions before building the model.

Ordinal: Linear interpolation into the interval [0,1]:

- Select a number of bins L (=10 by default);

- bin the data in L bins, and let x_0 < x_1 < x_2 < …< x_L be the cut-off points;

- let f_j = [weighted] fraction of records with x < x_j

Then the normalized value corresponding the original value x is defined by the rules

if x < x_0 then 0;

if x_L<= x then 1;

if x_j <= x < x_(j+1) for some j=0,1,..,L-1 then f_j + (x – x_j)/(x_(j+1) – x_j)*(f_(j+1) – f_j)

if x is null (missing), then it is replaced by the [weighted average] of the variable, and the above rule is applied

Discrete: Used for independent variables of type String. A variable with N unique values is mapped to N-1 indicator variables with values 1 or 0. If the original variable assumes the value V_k, the value of the k-th indicator variable is 1 and the others (if any) are 0.

Note that you can also generate indicator variables explicitly using the Expression Helper for the generation of dummy variables.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Missing Values

Only applies to independent variables. This column allows you to specify how missing values in your data are handled.

NaN values (which stands for "*Not a Number*") in continuous independent variables are treated the same way as missing values.

The missing value handling options are as follows:

use: Missing values will be treated like any other value. (This option is not available for continuous variables.)

ignore: Records containing missing values in this variable will be excluded from the analysis.

subst-median: The median value of the variable is substituted for every missing value. (This is the default option.)

subst-mode: The mode (most common value) of the variable is substituted for every missing value.

The following options are applicable for numeric or date/time IVs only:

subst-mean: The mean value of the variable is substituted for every missing value.

subst-min: The minimum value of the variable is substituted for every missing value.

subst-max: The maximum value of the variable is substituted for every missing value.

subst-zero: Zero is substituted for every missing value.

Note: The mean and the standard deviation used for the normalization of numeric variables (see the *Usage* section above) are calculated **before** the substitution of missing values. If you prefer to substitute missing values in your model variables prior to normalization, use the Variable Transformations node to perform the substitution before building the model.

Infinite Values

Only applies to numeric independent variables. This column allows you to specify how infinities are handled in numeric variables whose **Usage** is set to '*continuous*' . For more information on infinities and NaN, see Division and Infinity.

The options for handling infinities are:

ignore: Records containing infinities in this variable will be excluded from the analysis.

clip-to-finite: Infinities will be replaced with finite values. Positive infinities will be replaced with the maximum value of the variable; Negative infinities will be replaced with the minimum value of the variable.

Note: The mean and the standard deviation used for the normalization of numeric variables (see the *Usage* section above) are calculated **before** the substitution of infinite values. If any of the variables to be included in your model contain infinities, and if you prefer to substitute them prior to normalization, use the Variable Transformations node to perform the substitution before building the model.

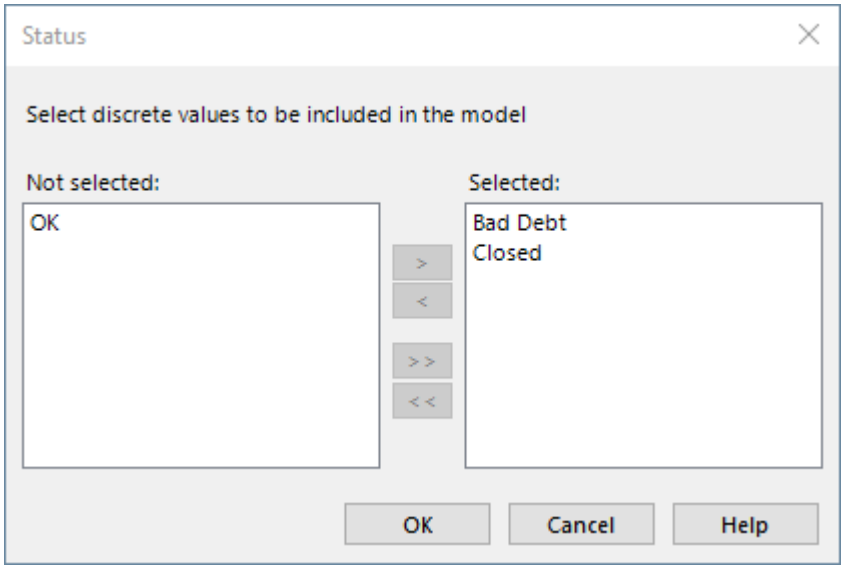
Dummy Variables

For any variable whose **Usage** is set to '*discrete*', this determines the representation of the variable as a set of *dummy variables*. Click in this column to select the *reference value* and the values to be

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

explicitly used by the model. For a variable with *N* unique values, *N-1* will be explicitly used in the training and one will be chosen as a reference value. By default, the largest category is set as a reference value (put in the *Not selected* list). To change the reference value, use the left & right arrow buttons to include or exclude values from the *Selected* list.

In a multinomial logistic regression, separate odds ratios are determined for independent variables for each dummy variable defined for the dependent variable, with the reference value excluded. In the example below, the Status variable has three values: 'OK', 'Bad Debt', and 'Closed'. The 'OK' category is set as a reference value:



Cardinality	Displays the cardinality (the number of distinct values) of each variable.
# of Missing Values	Displays the number of missing values for each variable.

The **Parameters and Attributes** tab displays two tables.

The first table contains the information about all model variables and their attributes. It is a summary of all settings in the Model Attribute Editor for the model (variable role, the treatment of missing values, etc.).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Parameters and Attributes

Model Type: Logistic Regression Model

Attributes

Name	Include	Scoring Input	Training Input	Role	Active	Value Type	Usage	Dummy Variables	Expression	Missing Values	Marginal Information Values	Infinite Values
Status2	yes	no	yes	dependent	no	string	discrete	'Bad Debt'	yes	use	N/A	
Credit Line	yes	no	yes	independent	no	real	continuous		no	subst-median	N/A	
Initial Score	yes	no	yes	independent	no	real	continuous		no	subst-median	N/A	
Recommended Line	yes	no	yes	independent	no	real	continuous		no	subst-median	N/A	
Number Inquiries	yes	no	yes	independent	no	real	continuous		no	subst-median	N/A	
Worst Rating Des	yes	no	yes	independent	no	string	discrete	'1 satisfactory rating' '2 satisfactory ratings' '3 satisfactory ratings' '4 satisfactory ratings' '5 or more satisfactory ratings' 'File but no rating' 'Major Derogatory '9 worst rating'' 'Minor Derogatory' 'No File'	no	use	N/A	
Major Derog Num	yes	no	yes	independent	no	real	continuous		no	subst-median	N/A	
Satisfactory Num	yes	no	yes	independent	no	real	continuous		no	subst-median	N/A	
Year at Address	yes	no	yes	independent	no	real	continuous		no	subst-median	N/A	

If the model includes any derived variables, the second table displays the expressions for these variables:

Expressions

Name	Expression
Status2	CASE WHEN [Status] IN('Bad Debt') THEN 'Bad Debt' WHEN [Status] IN('Closed', 'OK') THEN 'Good' END

The last table summarizes the model training parameters that were specified by the user in the model wizard. The example below is for a Logistic Regression model.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

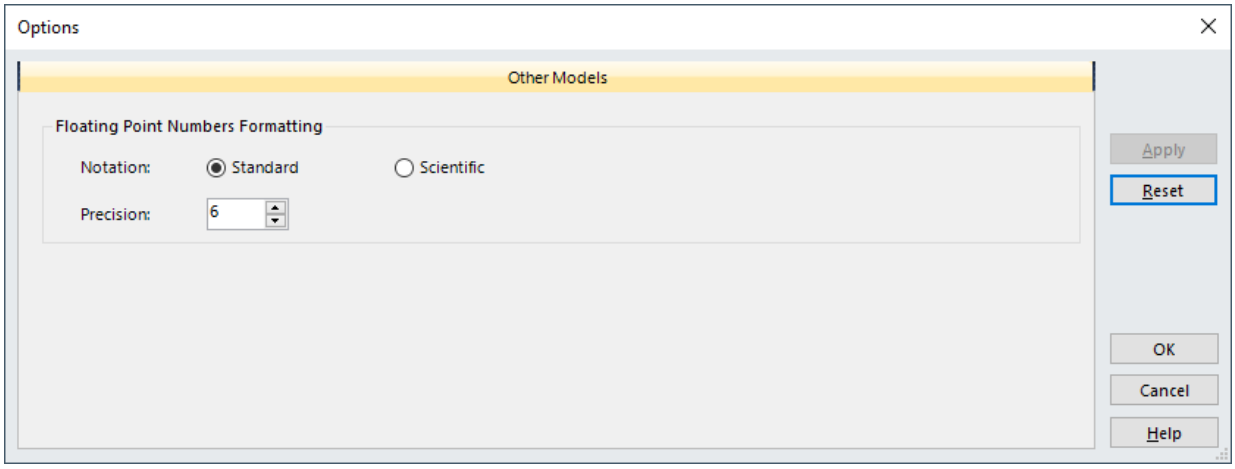


Parameters

Description	Parameter Name	Value	Value Description
Perform resubstitution test	Resubstitute	true	
Valid Hit Range for resubstitution test	HitRange	0	
Break indicator variables apart	SubSequencer	false	
Random Seed	RandomSeed	1	
Maximum number of iterations	MaxIteration	2000	
Variable Sequencing Method	SequenceMethod	Stepwise	
Entry Significance	EntrySignificance	0.05	
Exit Significance	ExitSignificance	0.05	
Round Trip Limit	RoundTripLimit	2	
Suppress intercept	Intercept	false	
Odds Ratio Confidence Level	OddsConfidence	0.05	
Compute errors	ComputeErrors	true	
Logistic Link Function	LinkFunction	logistic	N/A
Optimizer	Optimizer	NewtonRaphson	NewtonRaphson
Singular value threshold	ExcludeIVCollinearity	0	
Data Source	DataSource	[Credit_2021_3].[Credit]	
Relative convergence parameter	GradEpsilon	0	
Line search termination parameter	LineEpsilon	0.9	
# of records used for training	TrainedRecordCount	29167	
Training correct count	CorrectTrainCount	29163	
Entropy Explained	TrainEntropyExplained	0.995150	
Number of Degeneracies found	TrainDegenerate	0	

Results | Iteration History | Parameters and Attributes

To change the format and precision of the parameter values displayed in the Parameters and Attributes view, click the **Options** button in the toolbar or select the **Options** command from the **Tools** menu.



Scorecards

About Scorecards

A Scorecard in Knowledge Studio is a type of predictive model that associates points with a set of characteristics that are summed for a number of variables to produce a score. As a simple illustration:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Residential Status	Live with Parents	66
	Other	
	Own Home	70
	Rent	39
Years at Address	0 to <1	51
	1 to <3	56
	3 to <5	58
	5 to <10	66
	>=10	71

Note: A much more detailed introduction to Scorecards that discusses them from the point of view of business users in financial organizations can be found in Knowledge Studio Tutorial, which can be accessed from the Welcome screen or from the Help menu. It is recommended that you use the Tutorial to learn a typical workflow for creating a scorecard.

Creating a Scorecard in Knowledge Studio provides a simple extension to the workflows already used within the program for portfolio segmentation, univariate analysis, and predictive modeling.

The high-level steps are as follows:

Prerequisites for creating a scorecard:

- Data Profiling & Decision Trees for Discovery.
- Create coarse classing of the candidate input variables. This is usually done using the Optimal Binning and WOE Helper in Dataset Editor.
- Variable Selection. Determine which variables should be used as inputs.
- Predictive Modeling with Regression. Train and select a logistic regression model to base the Scorecard on.
- Perform Reject Inference to infer the behavior of rejected applicants (optional)

Build, validate, and deploy a scorecard

- Building a Scorecard. Convert the logistic regression model to a Scorecard.
- Validation. Validate the Scorecard.
- Scoring. Deploy (and monitor) the Scorecard.

Proceed to Inserting a Scorecard Model.

Inserting a Scorecard Model

Before inserting a scorecard model, make sure you have taken the prerequisite steps listed in the topic About Scorecards. The most detailed description of these prerequisites can be found in the Knowledge Studio Tutorial, which can be accessed from the Welcome screen or from the Help menu.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

It is assumed that you have prepared for inserting a Scorecard model and have all the prerequisites described in the topic About Scorecards and created the logistic regression model to be converted to a Scorecard model.

You may also want to go through the Reject Inference stage prior to developing your final scorecard in order to take into account the behavior of rejected applicants.

For a logistic regression model to be convertible to a Scorecard, it must have a binary DV and use the **logit** activation function.

A Scorecard node can be added to the workflow in one of the following ways:

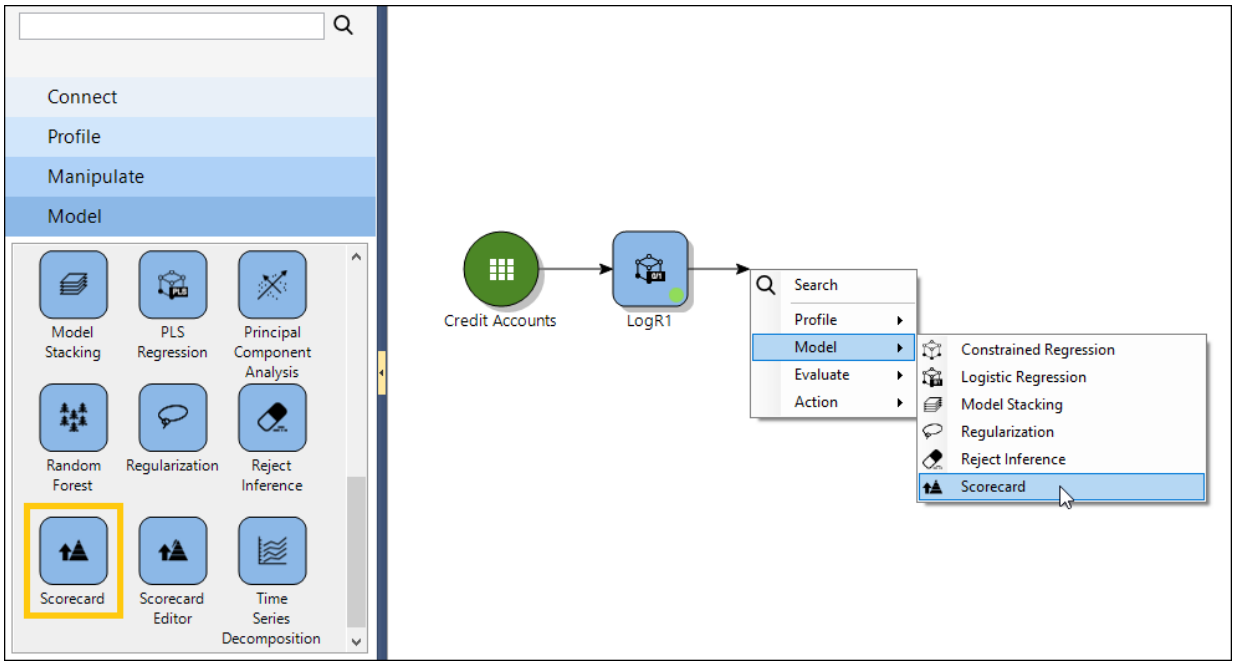
- Drag a connection arrow from a **Logistic Regression** node to any area on the Workflow canvas background. From the context menu that appears when you release the mouse button, select **Model --> Scorecard** as in the example screenshot below. A new **Scorecard** process node will be created.

OR

- Right-click on the Workflow canvas and select **Model --> Scorecard** from the context menu. The new Scorecard node is gray until its input is defined. Connect a logistic regression node to the Scorecard node.

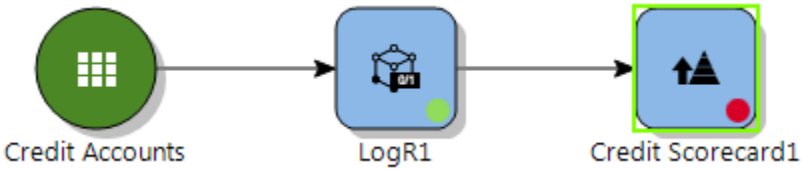
OR

- Open the **Model** palette and drag the **Scorecard** node onto the workflow canvas. The **Scorecard** node is gray while its input is undefined. Connect a logistic regression node to the Scorecard node.



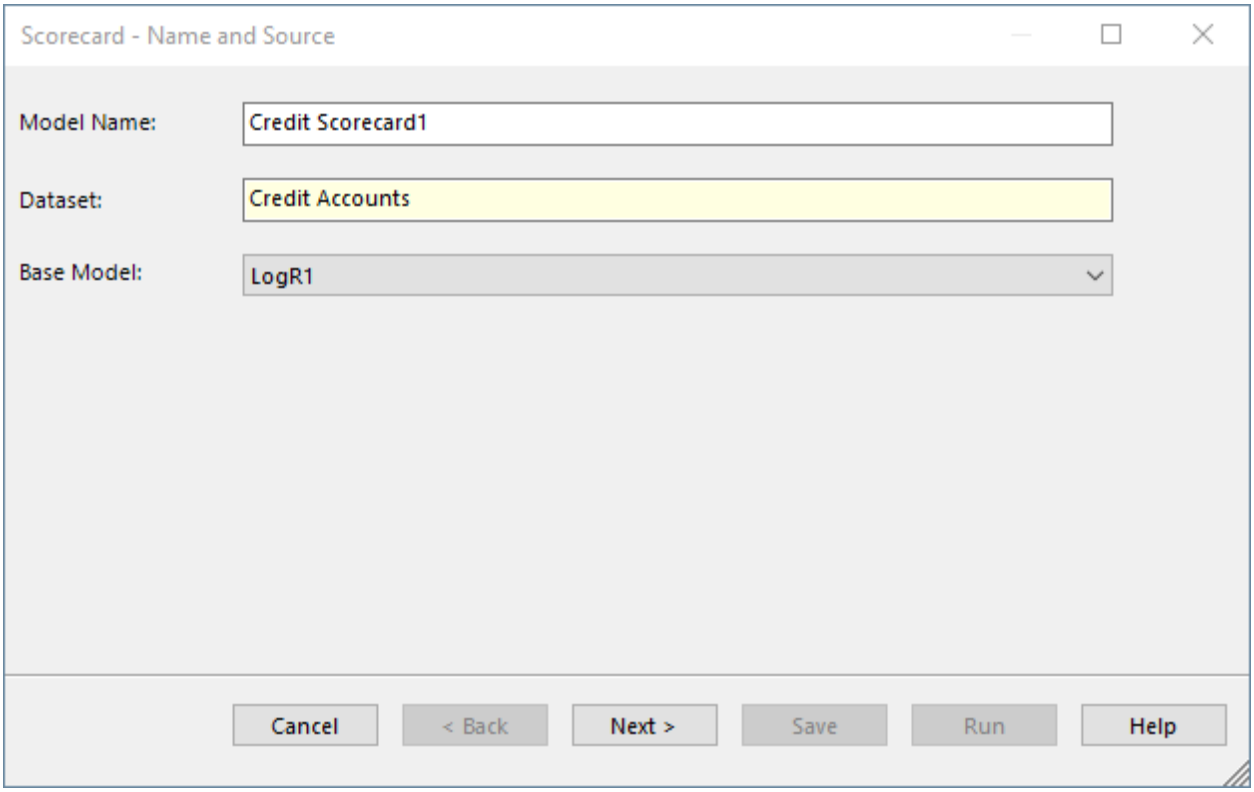
While it is not connected to other nodes, the scorecard node is gray, as its inputs are undefined. Connect the input logistic regression model to the scorecard node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Double-click on the scorecard node added to the canvas, or right-click on it and select **Modify** from the context menu.

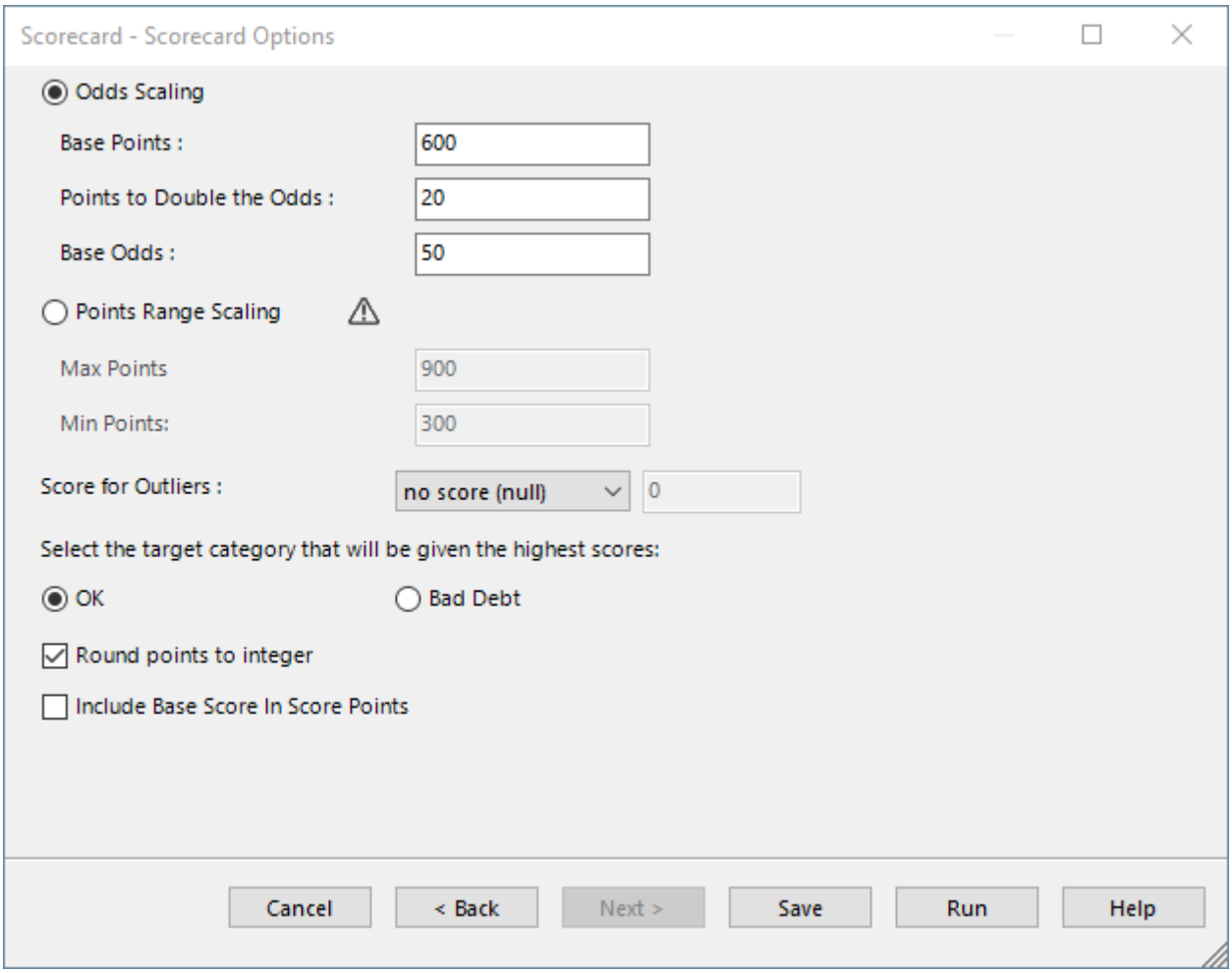
The Dataset field shows the parent dataset for the selected Base Model and is not editable.



Click **Next** to proceed to the Scorecard Options page.

The final step of the wizard lets you specify **scorecard scaling** options. There are two scaling options to choose from: ***Odds Scaling*** and ***Points Range Scaling***.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Odds Scaling

Base Points	The number of points for the specified value of Base Odds. The default is 600 points.
Points Double t Odds	If this value is set to N, then the odds will be doubling at every N points. The default is 20.
Base Od	Base odds. The default is 50:1

The scores are scaled logarithmically. The defaults represent the most common scaling options, with base odds of 50:1 at 600 points, doubling at every 20 points. The relationship between odds and scores is as follows:

```
Score = Offset + Factor * ln(odds)
```

where *ln* is the natural logarithm, `Factor = (Points to Double the Odds)/ln(2)`, and *Offset* can be found from the above equation with *Base Points* and *Base Odds* substituted for *Score* and *odds*, respectively.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Points Range Scaling

Choose this option if you want the scorecard scaled with specific maximum and minimum number of points. The wizard determines the scorecard formula from the specified *Max Points* and *Min Points* values.

Note: This option provides linear scaling between min and max points. The differences between score points may be less meaningful than with Odds Scaling.

Max Points	The maximum number of points for the scorecard. The default is 900 points.
Min Poi	The minimum number of points for the scorecard. The default is 300 points.

If `P_max` and `P_min` are the Max and Min Points values specified in the wizard, `beta_0` are the regression model parameters in the log odds formula, `w` are the WOE values, then assume

```
M = max { beta_0 + beta_1 * w_{1,k1} + beta_2 * w_{2,k2} + ...}

m = min { beta_0 + beta_1 * w_{1,k1} + beta_2 * w_{2,k2} + ...}
```

where the max and min are taken over the values of `k` (the max/min possible value of the regression model that can be obtained by choosing, for each variable, the category with the max/min WOE coefficient).

Then the Scorecard coefficients are:

```
Offset = (M*P_min - m*P_max)/(M-m)

Factor = (P_max - P_min)/(M-m)
```

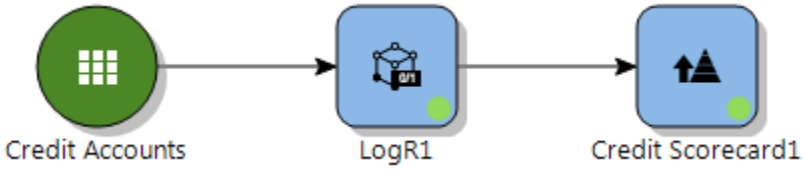
Common Options

Score for Outlier	The score for the records not recognized by the model (i.e., records with values that did not occur in the dataset the model was trained on). By default, such records get no score (null). Alternatively, you can assign a custom score or the sum of scorepoint averages.
Target catego that will be giv the highest score	This is the category of the dependent variable that is supposed to get higher scores than the other category. More precisely, higher probability of that category produced by the base regression model will result in higher score of the scorecard. It is therefore natural to select the value indicating good status. The

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

	DV categories, which are determined from the base model, are displayed next to the radio buttons. Select the appropriate category.
Round points integer	If this option is selected, the scorecard is rounded to the nearest integer number. Otherwise, real numbers are used. Note that slight variations between the results of the original logistic regression model and the generated Scorecard model can be observed if you choose to round the points to integer values (which is the default). If you do not round the points, the Scorecard model and the original logistic model reveal exactly the same lift.

Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the operation, double-click the Scorecard model node to view the results.



The scorecard expression is based on model coefficients and the parameters specified in the Scorecard Scaling Options dialog.

Note: If there is a CASE statement expressions associated with a continuous independent variable (usually representing the Weight of Evidence expression created with the Optimal Binning and WOE Helper), the scorecard assigns appropriate scores to each bin in the expression. Otherwise, for raw continuous variables (without expressions), the scorecard assigns a multiplier.

Scorecard Editor

The **Scorecard Editor** node in the Model palette provides the ability to edit an existing scorecard, manually changing the score points and the base score. It also illustrates the performance of the edited scorecard by showing the Model Analyzer charts (Cumulative Lift, etc.).

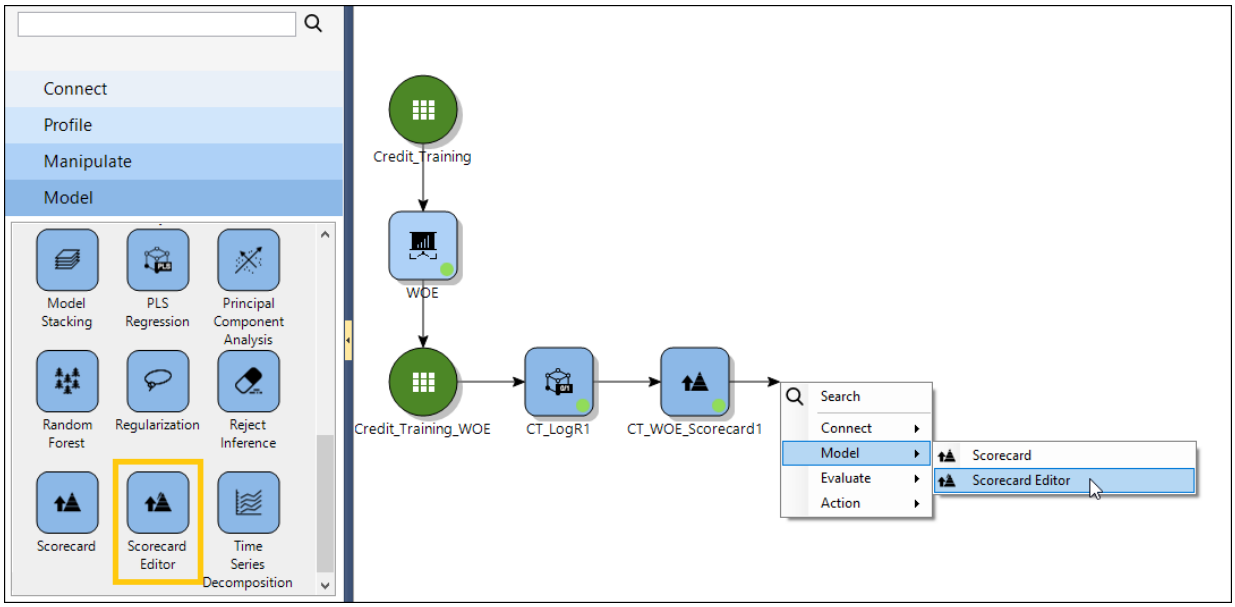
To edit an existing scorecard, add a Scorecard Editor node to your workflow as follows:

- Drag a connection arrow from a Scorecard node to any area on the Workflow canvas background. This brings up a context menu that allows you to select the desired operation. From the context menu, select **Model --> Scorecard Editor** as shown in the example below. A new **Scorecard Editor** process node will be created.

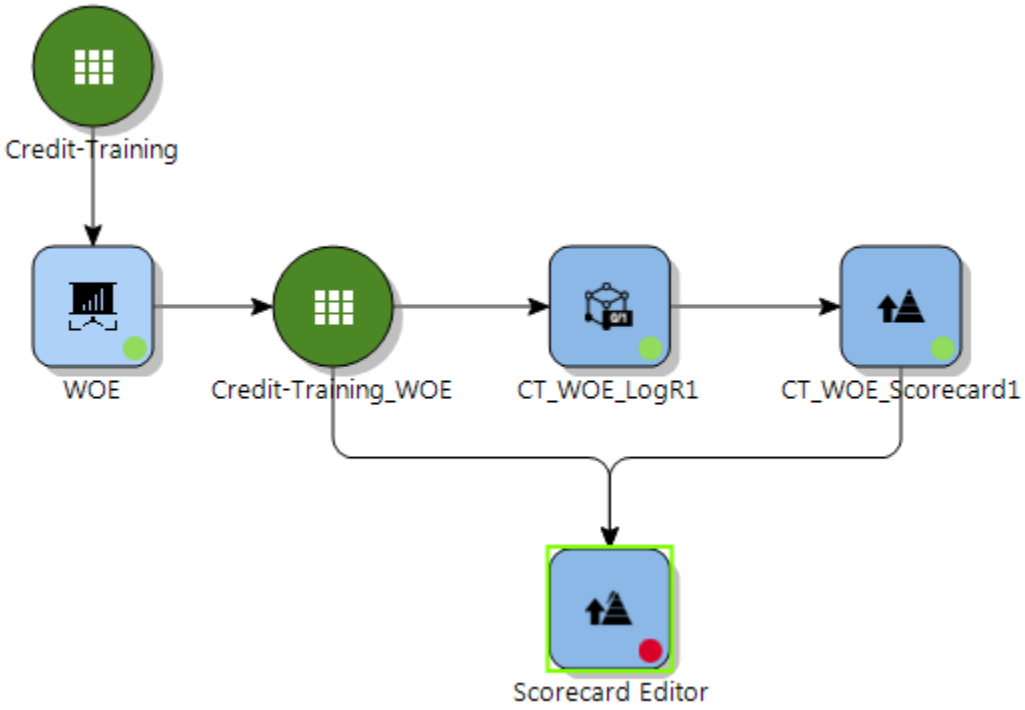
  OR:

- Drag the **Scorecard Editor** node from the **Model** palette onto the canvas and connect the required Scorecard node to it.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The new Scorecard Editor node is created, but it still requires a dataset input in addition to the Scorecard input node. Connect the desired input dataset to the Scorecard Editor. In the example below, we connect the same dataset as the one used for the source Scorecard. You can use any dataset as long as it contains the information about the actual values of the dependent variable.

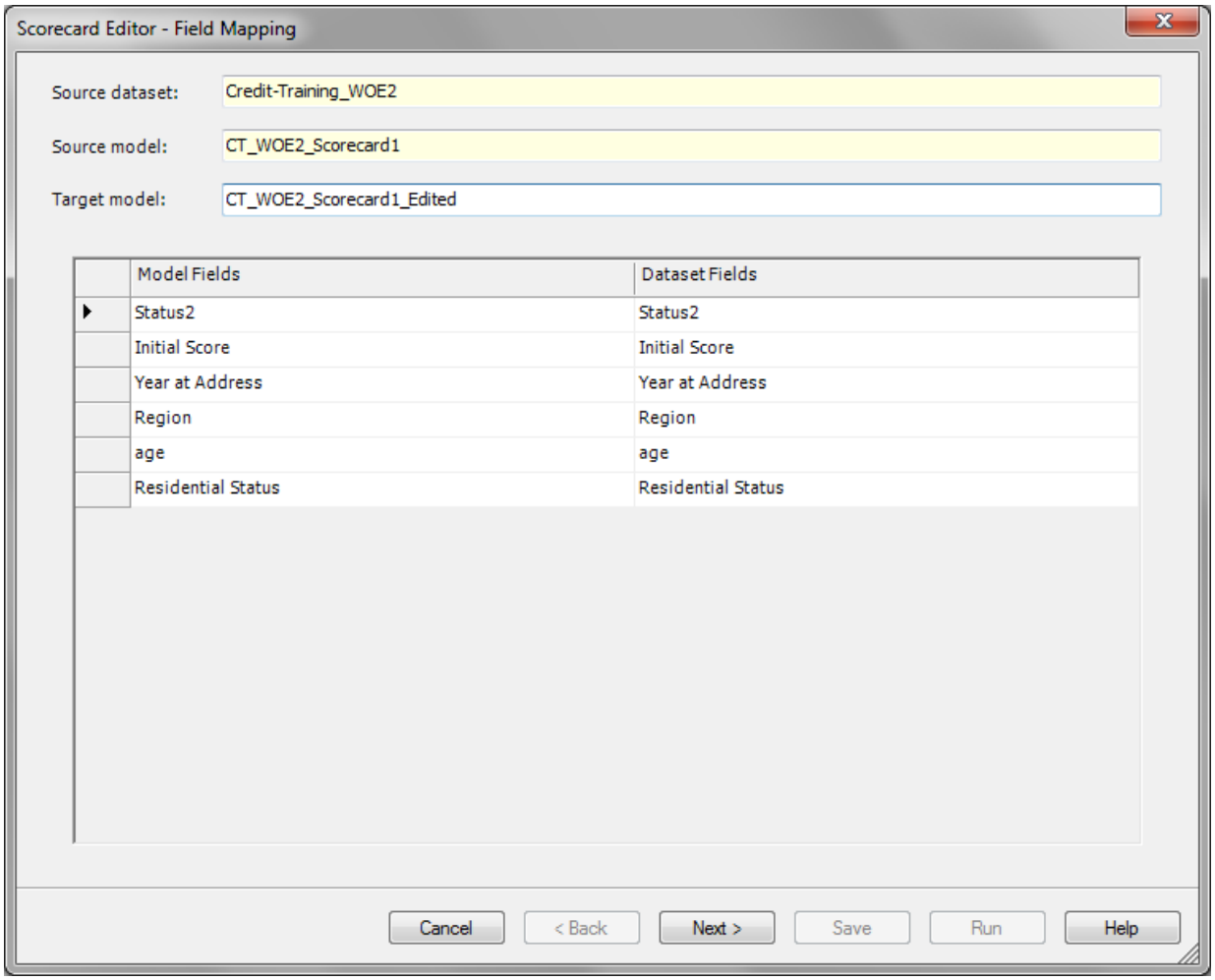


Double-click the Variable Selection node or right-click on it and select **Modify**.

- Specify the name for the new (edited) Scorecard model.
- This page also provides the option of mapping the field names in the model to the ones in the validation input dataset. In some situations, the names of model variables may not match

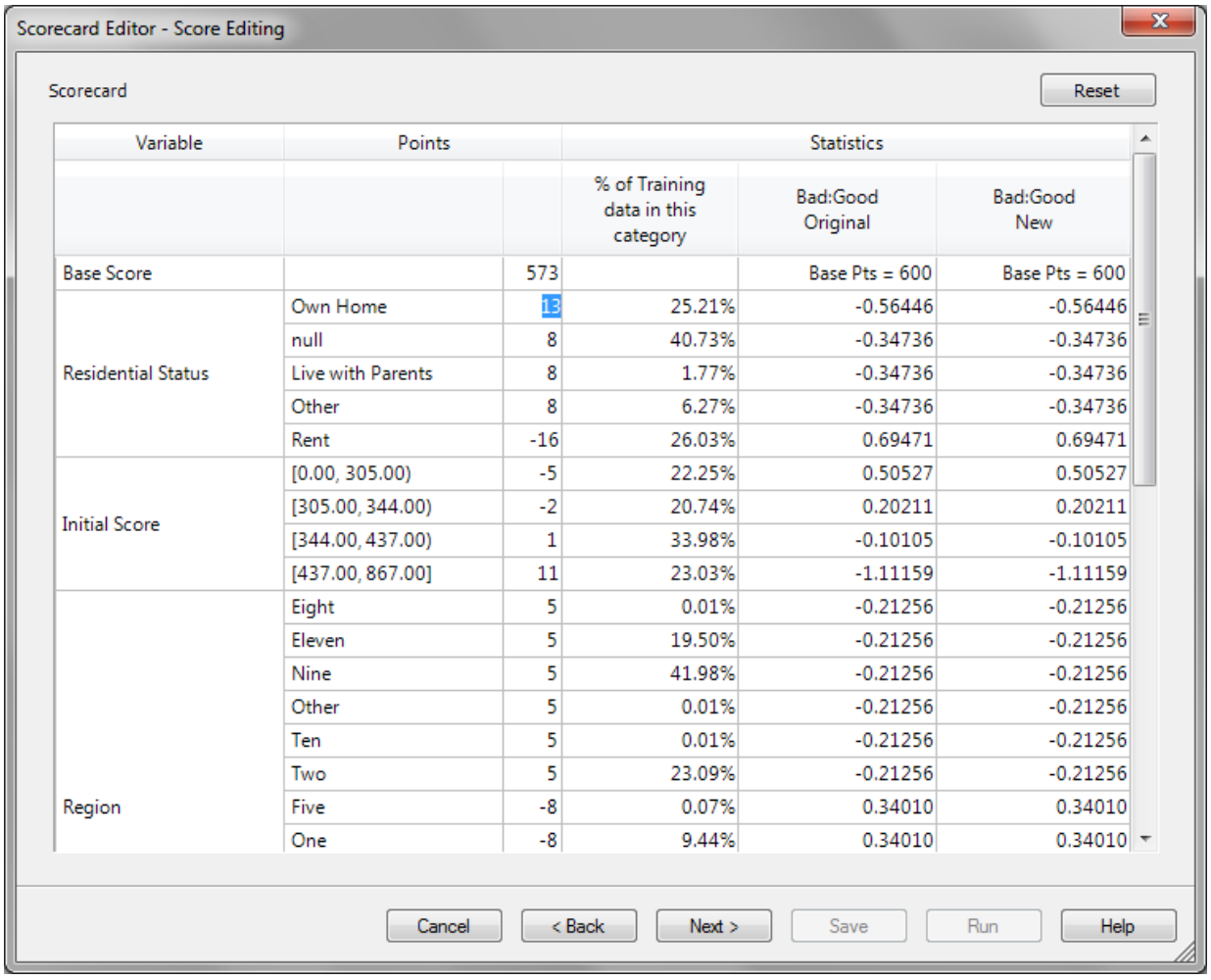
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

the names of the corresponding fields in the input dataset, so you will need to manually map fields by clicking in the right column (Dataset Field Names) and selecting the right fields to match the model variables in the left column:



Click Next. In the next page you can change the values of Score Points and the Base Score.

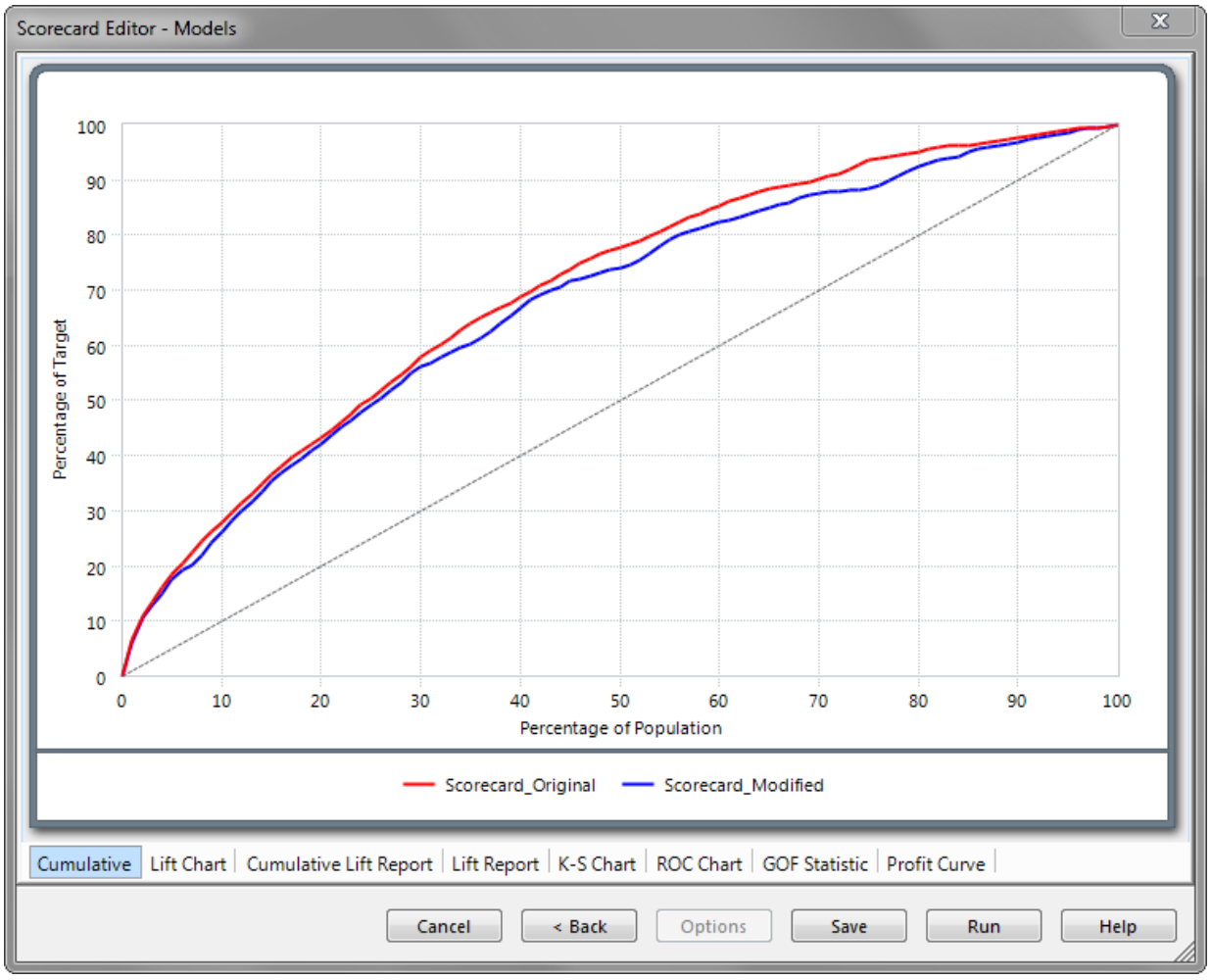
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Variable	Points		Statistics		
			% of Training data in this category	Bad:Good Original	Bad:Good New
Base Score		573		Base Pts = 600	Base Pts = 600
Residential Status	Own Home	13	25.21%	-0.56446	-0.56446
	null	8	40.73%	-0.34736	-0.34736
	Live with Parents	8	1.77%	-0.34736	-0.34736
	Other	8	6.27%	-0.34736	-0.34736
	Rent	-16	26.03%	0.69471	0.69471
Initial Score	[0.00, 305.00)	-5	22.25%	0.50527	0.50527
	[305.00, 344.00)	-2	20.74%	0.20211	0.20211
	[344.00, 437.00)	1	33.98%	-0.10105	-0.10105
	[437.00, 867.00]	11	23.03%	-1.11159	-1.11159
Region	Eight	5	0.01%	-0.21256	-0.21256
	Eleven	5	19.50%	-0.21256	-0.21256
	Nine	5	41.98%	-0.21256	-0.21256
	Other	5	0.01%	-0.21256	-0.21256
	Ten	5	0.01%	-0.21256	-0.21256
	Two	5	23.09%	-0.21256	-0.21256
	Five	-8	0.07%	0.34010	0.34010
	One	-8	9.44%	0.34010	0.34010

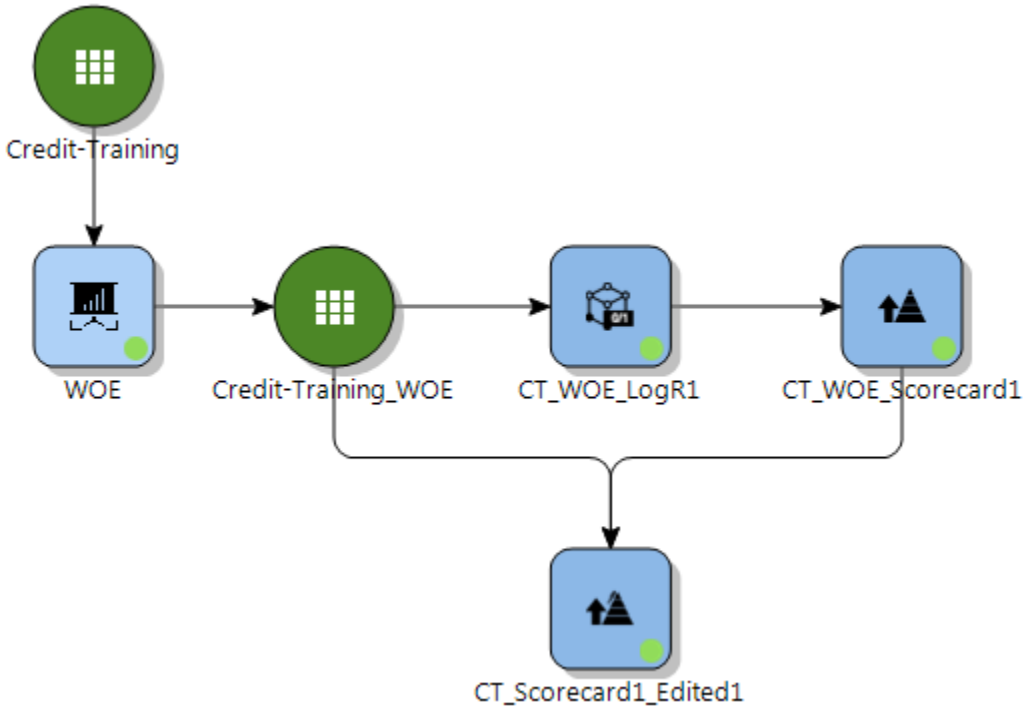
While editing, click Next and Back if you need to see the effect of your edits on the performance of the scorecard on the input dataset.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the operation, double-click the Scorecard Editor node to view the results. This node can be used as any other Scorecard node. If you need to modify it to edit and run again, right-click on it and select **Modify** from the context menu.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



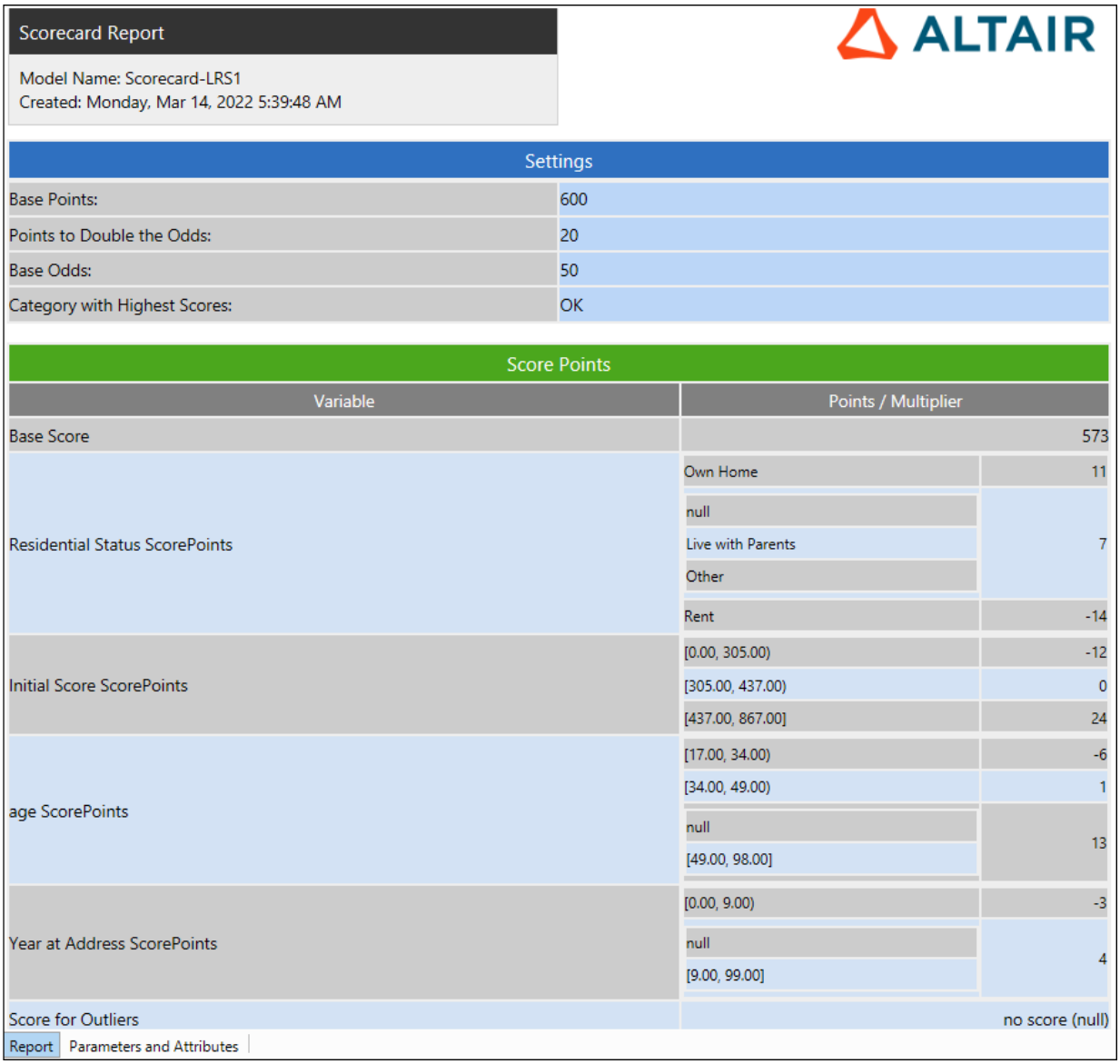
The **Results** tab of a trained Scorecard model displays the Scorecard Report.

The Settings section of the report displays the scorecard scaling parameters that were specified by the user in the Scorecard Scaling Options dialog.

The Score Points section displays the scorecard in a tabular form with *Base Score* (if applicable), score points for each variable, and the score for outliers.

Note: If there is a CASE statement expressions associated with a continuous independent variable (usually representing the Weight of Evidence expression created with the Optimal Binning and WOE Helper), the scorecard assigns appropriate scores to each bin in the expression. Otherwise, for raw continuous variables (without expressions), the scorecard assigns a multiplier.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Scorecard Report

Model Name: Scorecard-LRS1
Created: Monday, Mar 14, 2022 5:39:48 AM

Settings	
Base Points:	600
Points to Double the Odds:	20
Base Odds:	50
Category with Highest Scores:	OK

Score Points

Variable	Points / Multiplier	
Base Score		573
Residential Status ScorePoints	Own Home	11
	null	7
	Live with Parents	
	Other	
	Rent	-14
Initial Score ScorePoints	[0.00, 305.00)	-12
	[305.00, 437.00)	0
	[437.00, 867.00]	24
age ScorePoints	[17.00, 34.00)	-6
	[34.00, 49.00)	1
	null	13
	[49.00, 98.00]	
Year at Address ScorePoints	[0.00, 9.00)	-3
	null	4
	[9.00, 99.00]	
Score for Outliers		no score (null)

Report | Parameters and Attributes

Model Selector

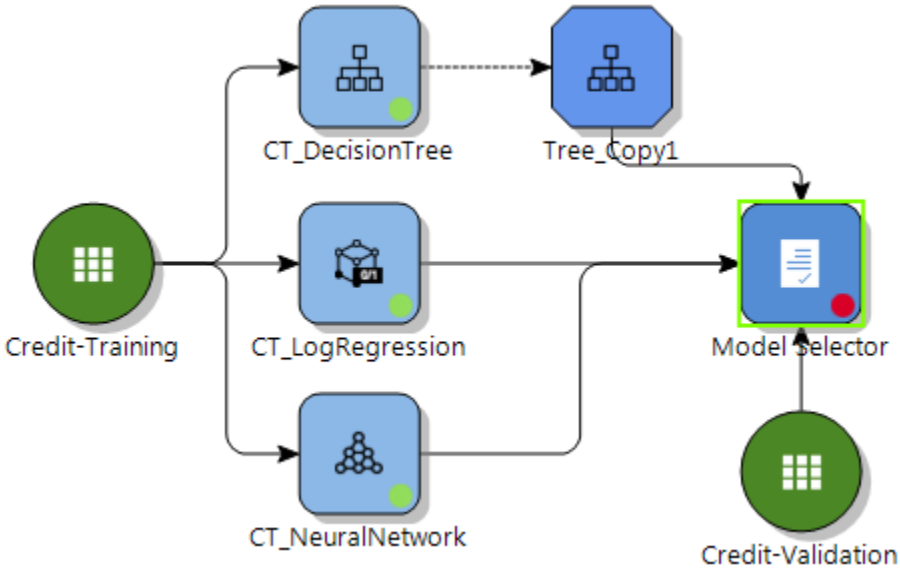
The **Model Selector** node provides the ability to select the best model based on one of the available model performance metrics. Prerequisites: two or more predictive models and a dataset for model validation.

To automatically select the best model from several models on your workflow, drag the **Model Selector** node from the **Evaluate** palette onto the canvas and connect the required input models to the node. Connect the validation dataset to Model Selector node.

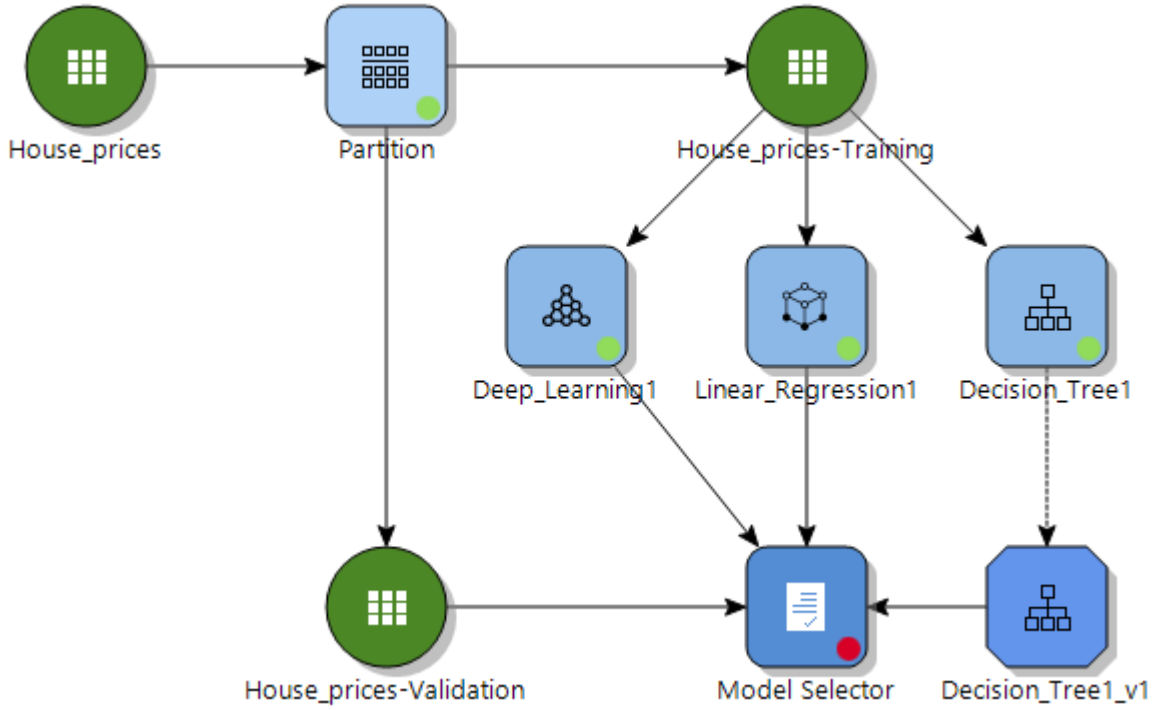
- All input models must have compatible model schemas: they should have the same dependent variable, and its data type must be the same in all models.
- The input validation dataset must contain all variables required by the input models, and their data types must match those of the model variables.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Here is an example of a workflow with three models, which include a Decision Tree. Note: When connecting a Decision Tree node to Model Selector, the tree model instance is created automatically.



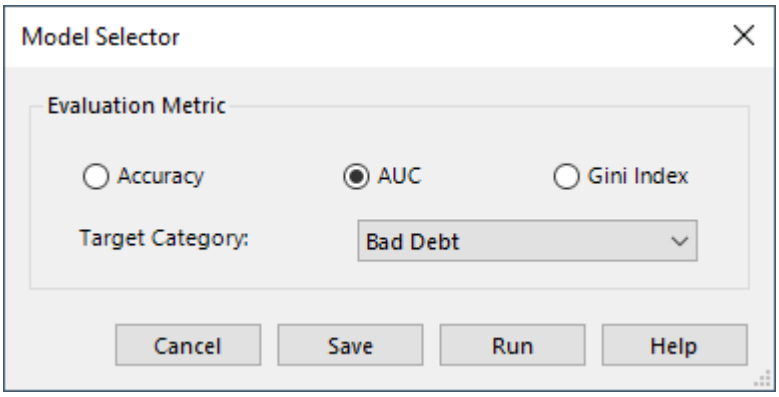
An example of a workflow with three models with continuous dependent variable, including linear regression:



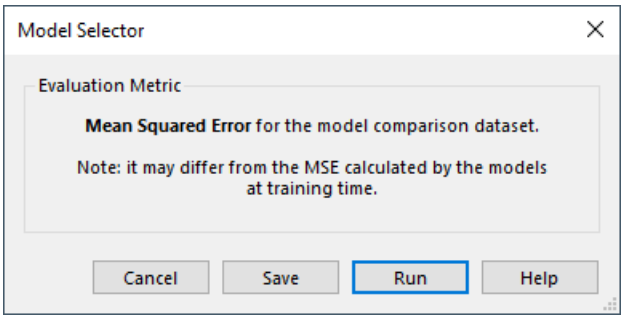
Double-click the Model Selector node or right-click on it and select **Modify**. Choose the desired metric as the criterion to select the best model.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- If the dependent variable is discrete, the evaluation metric options are: *Accuracy*, *AUC* (*the area under the ROC curve*), and *Gini Index*.



- If the dependent variable is continuous, the evaluation metric is Mean Squared Error (MSE). This is the mean of the squared errors calculated on the input dataset used for model validation and comparison. This MSE value may differ from the MSE calculated by the input models at training time.

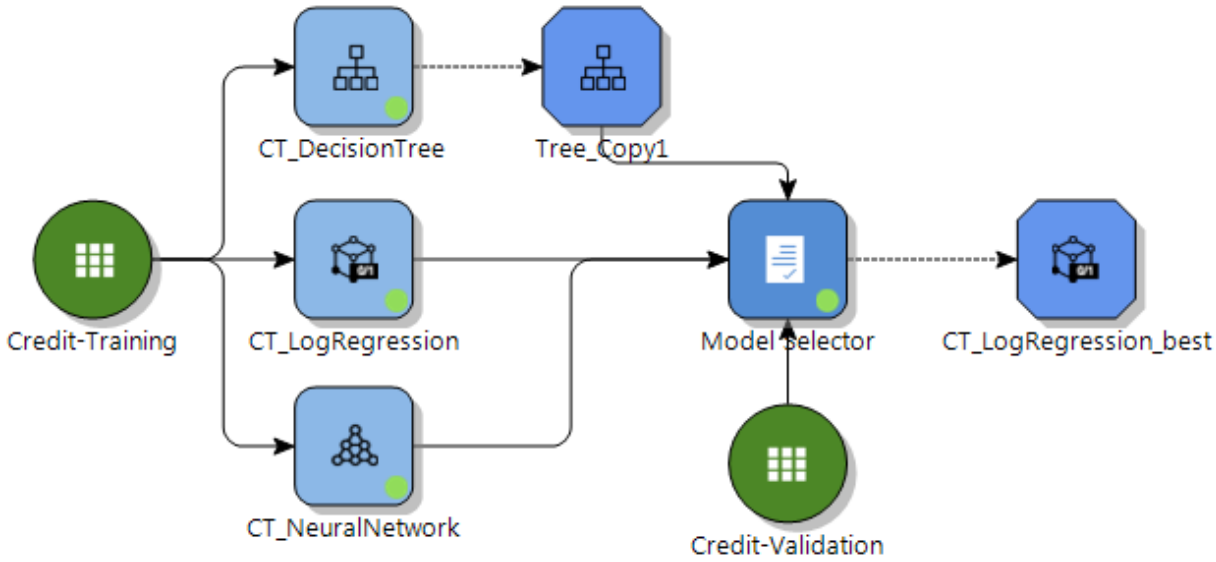


Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.

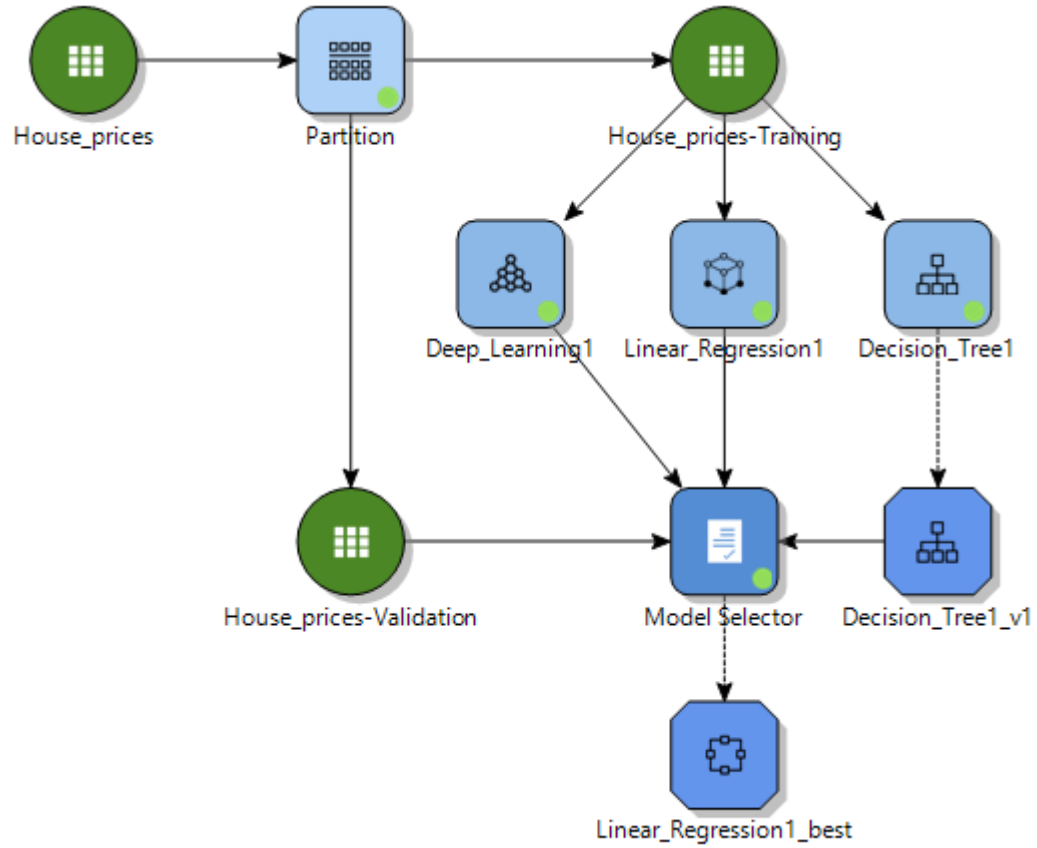
For every model connected to the node, the attached dataset will be scored with that model, and the appropriate evaluation metric will be calculated. For the selected evaluation metric, the node will determine the model with the best value for that statistic (the highest Accuracy, AUC, or Gini in the discrete case; the lowest MSE in the continuous case). The output is a copy of the trained model node that has the best value for that statistic.

In the example below, the best model is the logistic regression model:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



In the example below, the best model is the linear regression model:

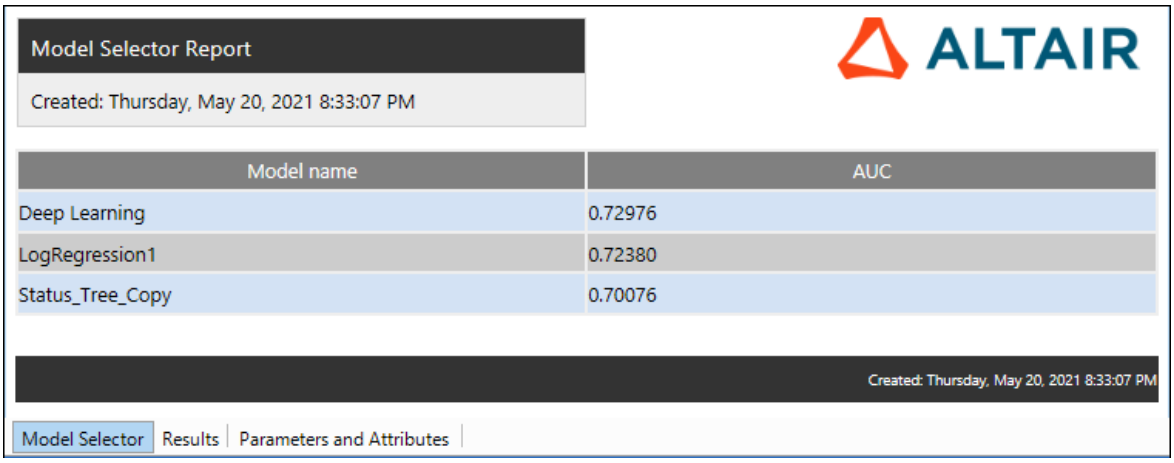


Double-click the output model node labeled as the best one.

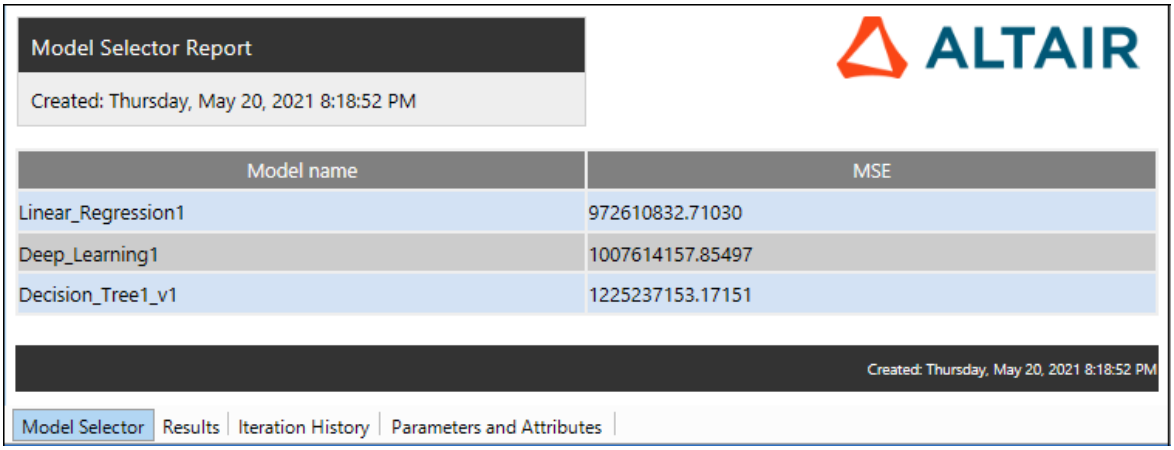
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The first tab is the **Model Selector** Report, which lists the values of the selected evaluation metric for all input models. The tabs **Results** and **Parameters and Attributes** show the model training results and model attributes of the best model.

An example of the Model Selector report in the case of a discrete dependent variable:



An example of the Model Selector report in the case of a continuous dependent variable:



Model Stacking

The **Model Stacking** node in the **Model** palette of Knowledge Studio represents an ensemble model that combines predictions produced by multiple predictive models to build the final model. Unlike Random Forest, Bagging or Boosting, which are ensembles of models of the same type (Decision Trees), the Model Stacking node can handle different model types. For example, a Model Stacking node can be based on a set of models including Deep Learning (Neural Network), Logistic Regression, Random Forest, etc.

The general workflow is to create several models from a dataset, score the same dataset or a test dataset with these models, and use the generated scores as new predictors to build the final (stacked) model. By automating cumbersome steps required to stack multiple models, combine and use their scores, Model Stacking saves time in the modeling process.

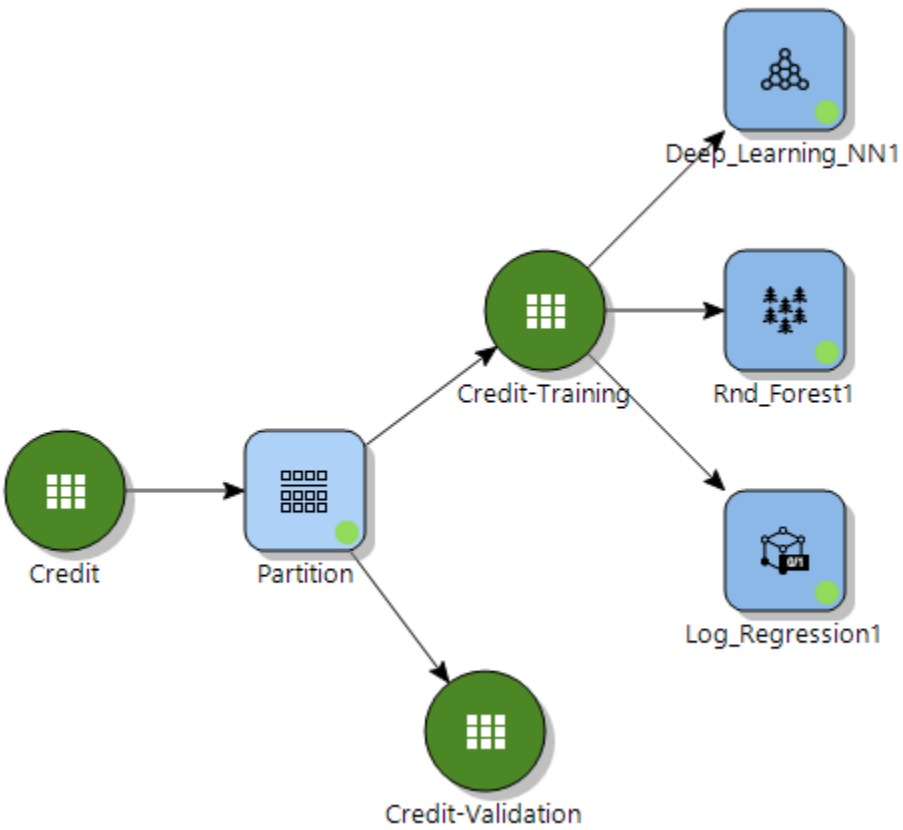
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Prerequisites

The following are prerequisites for Model Stacking:

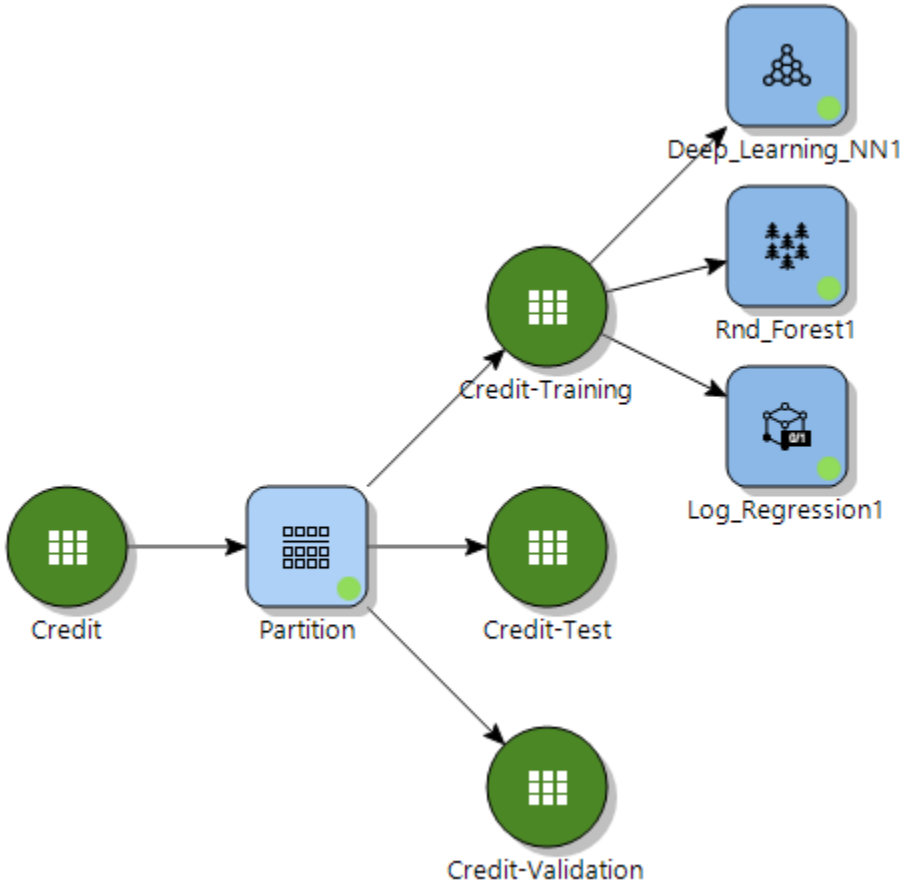
1. A training dataset
2. Two or more predictive models of any types built on the training dataset.
    - *Note*: *In a more general case, the models may be built on different datasets, as long as the name and data type of the dependent variable are the same. For models with discrete dependent variable, it is also required that the cardinality of the dependent variable be the same*.
3. [Optional]: A separate test dataset (or partition) to score with the models to produce inputs for the stacked model. The scoring is performed automatically by the Model Stacking node.
4. [Optional]: A validation dataset (or partition) to validate the stacked model after it has been built.

Prerequisites | Option 1: The original data is partitioned into two datasets: *training* and *validation*. The training dataset is used to build several predictive models, and the same dataset is then used as an input for the Model Stacking node. (The stacked model is built based on the scores of the models on the training dataset). The validation partition is set aside for subsequent validation of the stacked model. See the example setup below.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Prerequisites | Option 2: The original data is partitioned into three datasets: *training, testing,* and *validation*. The training dataset is used to build several predictive models. The testing dataset is then used as an input for the Model Stacking node. (The stacked model is built based on the scores of the models on the testing dataset). As in the first option, the *validation* partition is set aside for subsequent validation of the stacked model. See the example setup below.



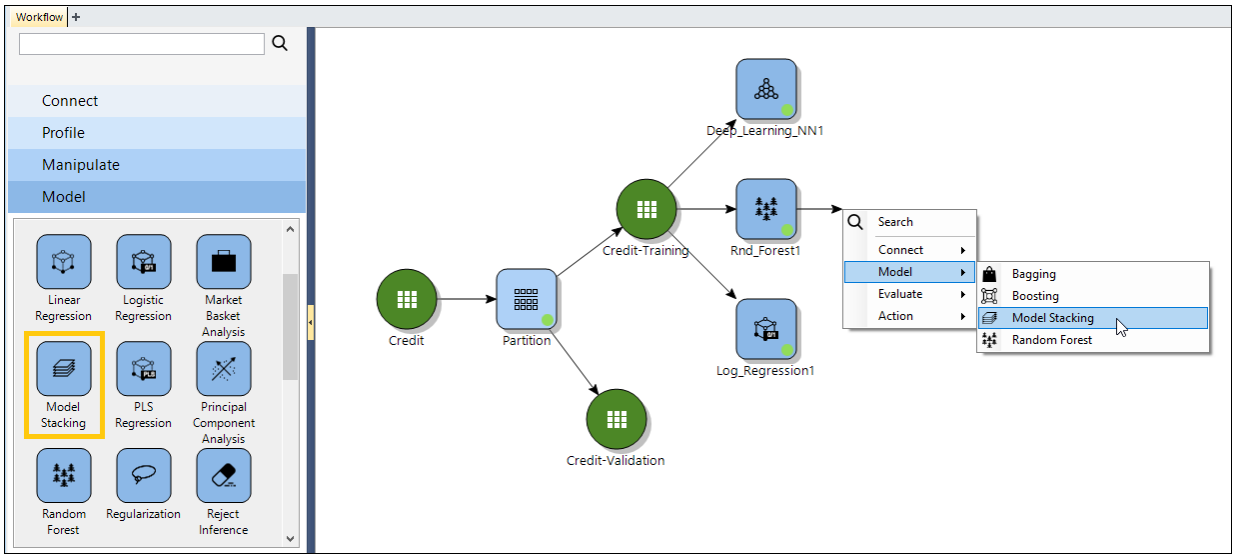
A **Model Stacking** node can be added to the workflow in one of the following ways:

- Drag a connection arrow from any of the model nodes to any area on the Workflow canvas background. From the context menu that opens when the mouse button is released, select **Model --> Model Stacking**. A new **Model Stacking** node will be created. Connect the other input model nodes and the training dataset (or the testing dataset) to the Model Stacking node.

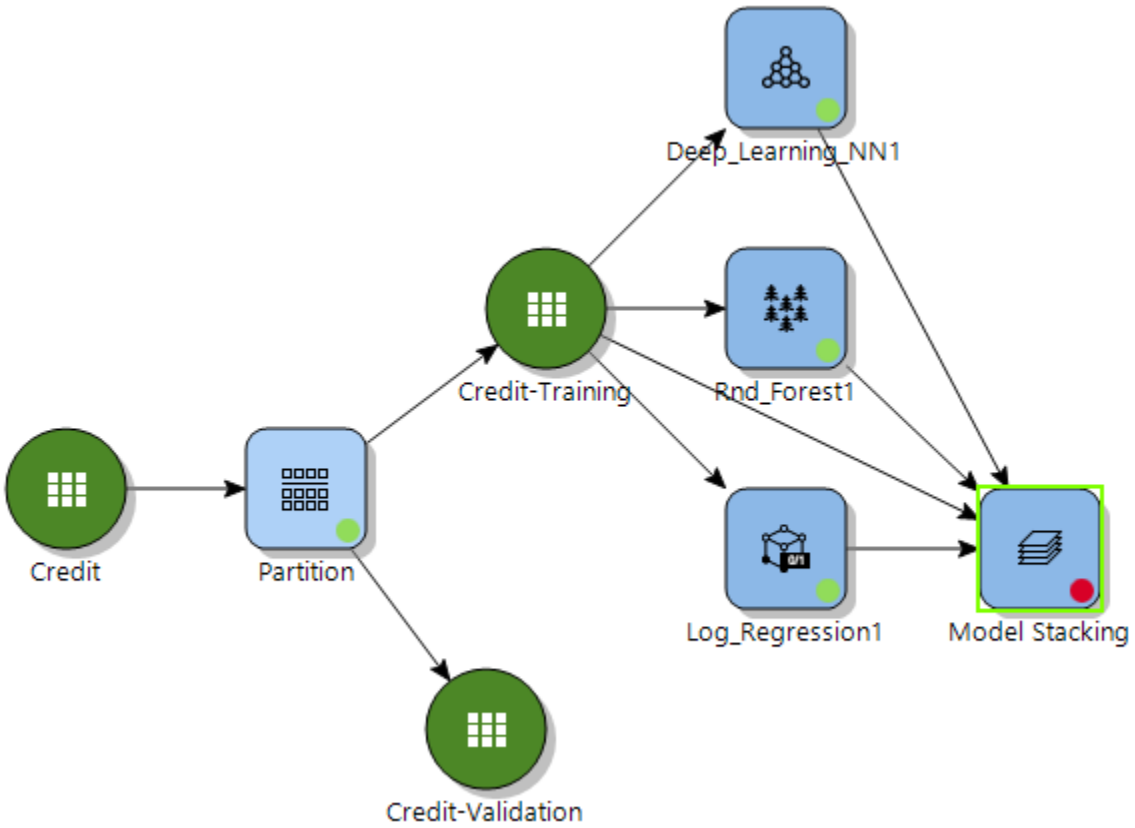
OR

- Right-click on the Workflow canvas and select **Model --> Model Stacking** from the context menu. Alternatively, drag the **Model Stacking** node from the Model palette onto the workflow canvas. A new **Model Stacking** node will be created. Connect all input model nodes and the training dataset (or the testing dataset) to the Model Stacking node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

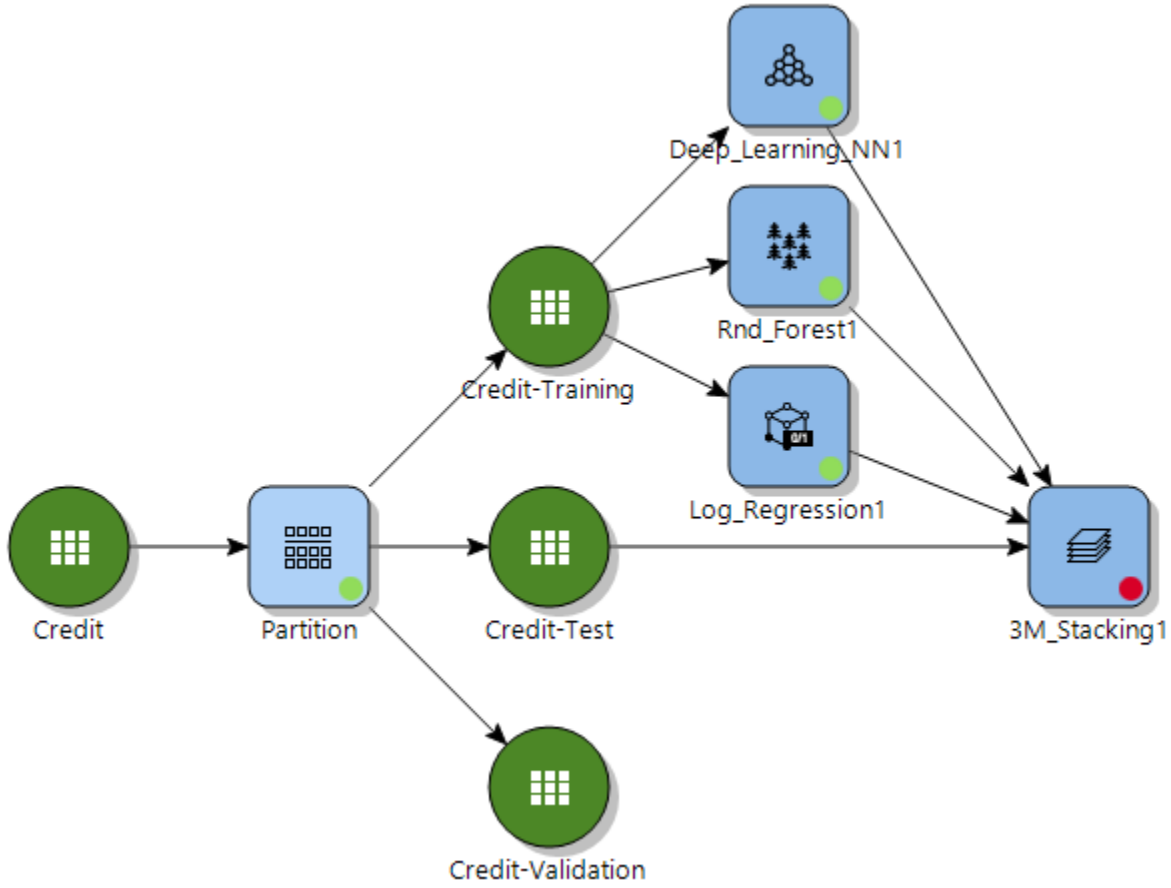


The example below shows one of the ways to connect the model and dataset nodes to the Model Stacking node. This corresponds to Prerequisites | Option 1.



The following example shows another way to connect the model and dataset nodes to the Model Stacking node. This corresponds to Prerequisites | Option 2. The input dataset for the Model Stacking node is not the same as the one used for building the individual models.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



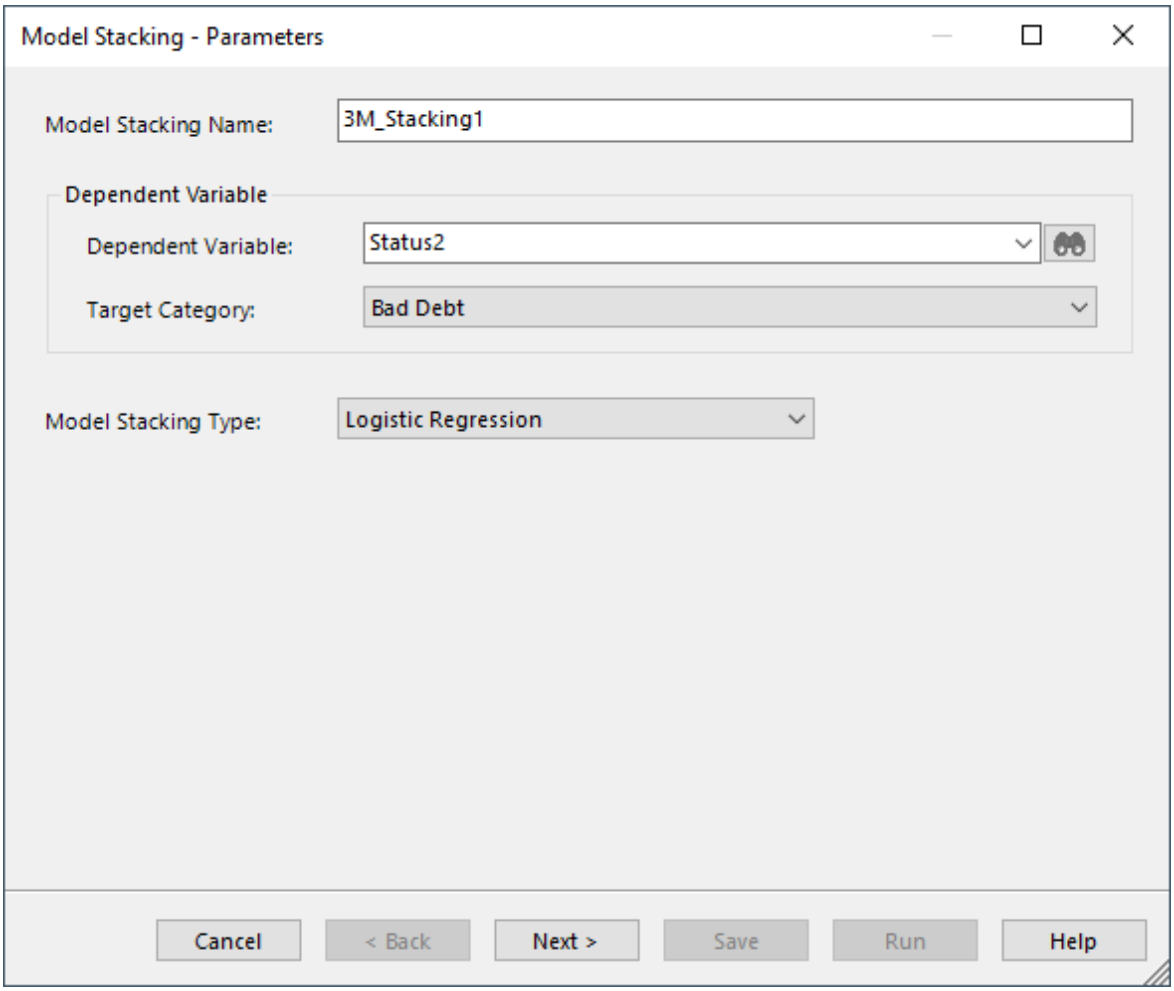
Double-click on the **Model Stacking** node or right-click on it and select **Modify** from the context menu. The Model Stacking wizard opens.

In the first page of the wizard:

- Specify the model name
- Select the dependent variable. If the dependent variable is discrete, select the Target Category. The Target Category selection is not applicable for continuous dependent variables.
- Select the Model Stacking Type. For discrete dependent variables, the available types are:
  - *Logistic Regression* (for discrete dependent variables only)
  - *Linear Regression* (for continuous dependent variables only)
  - *Average* : The average of the probability scores of the individual models (base learners) in the stacked model

Example 1: Discrete Dependent Variable

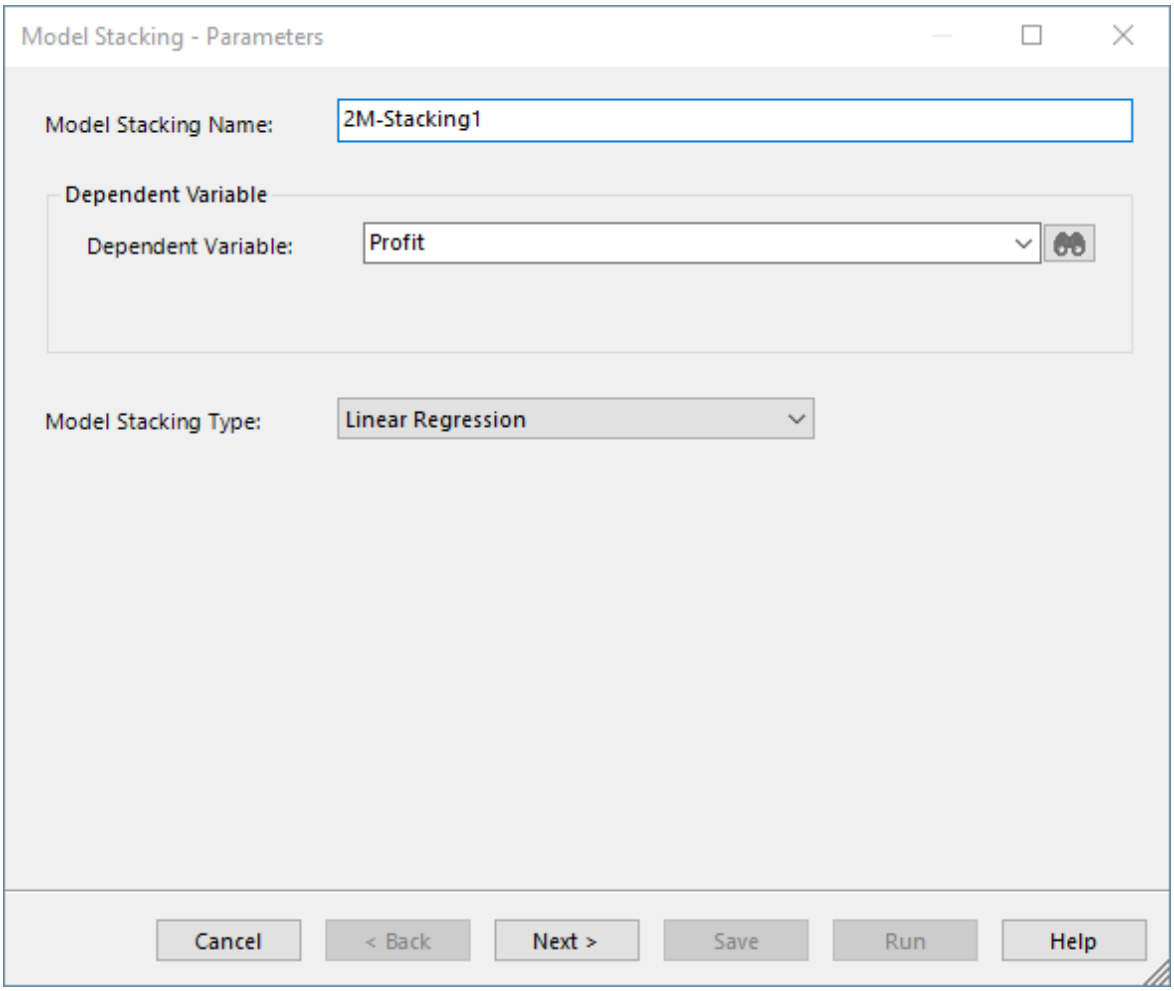
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



-

Example 2: Continuous Dependent Variable

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

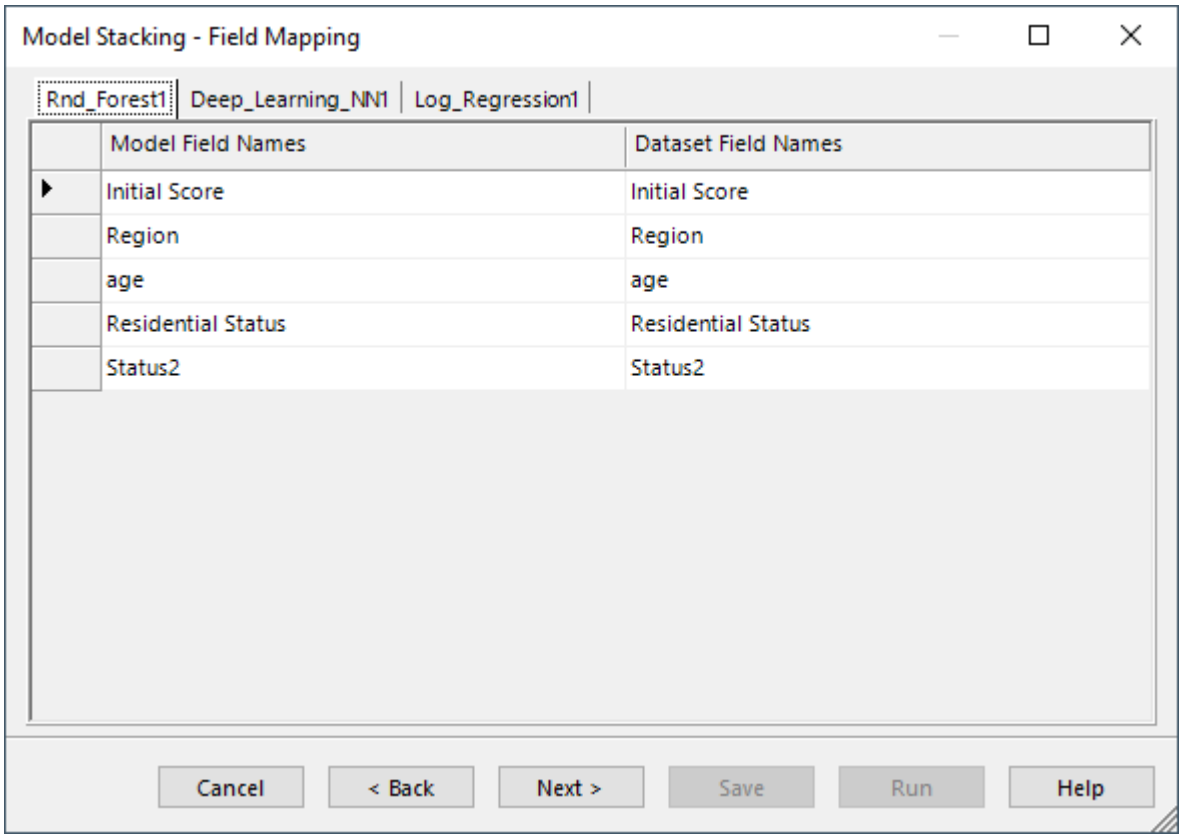


Click **Next** to proceed to the Model Selection page.

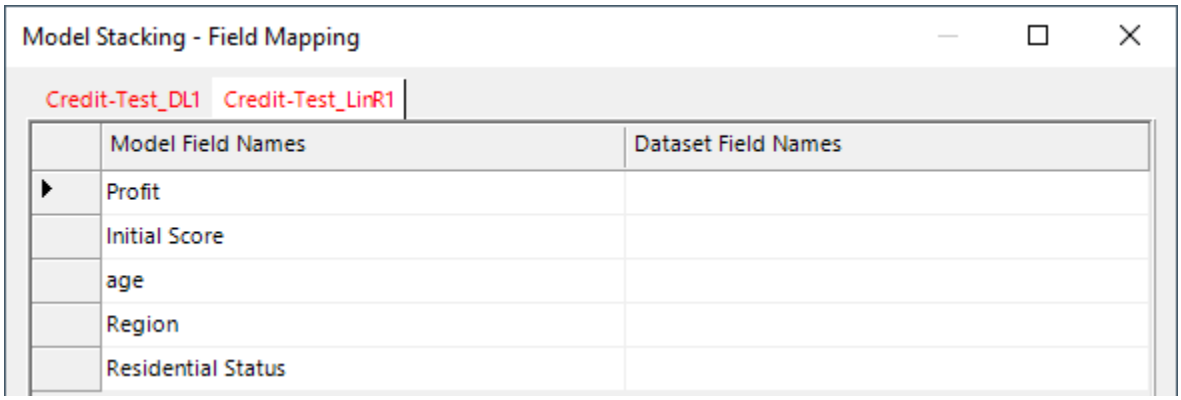
Just as in the Scoring wizard, this page provides the option of mapping the variables in each input model to the fields in the input dataset of the Model Stacking node. In some cases, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map fields by clicking in the right column (Dataset Field Names) and selecting the right fields to match the model variables in the left column.

There is a separate field mapping tab for each component model ("base learner"). The “Model Field Names” column in each tab lists the independent variables in the corresponding model. In general, the component models may have different sets of independent variables.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



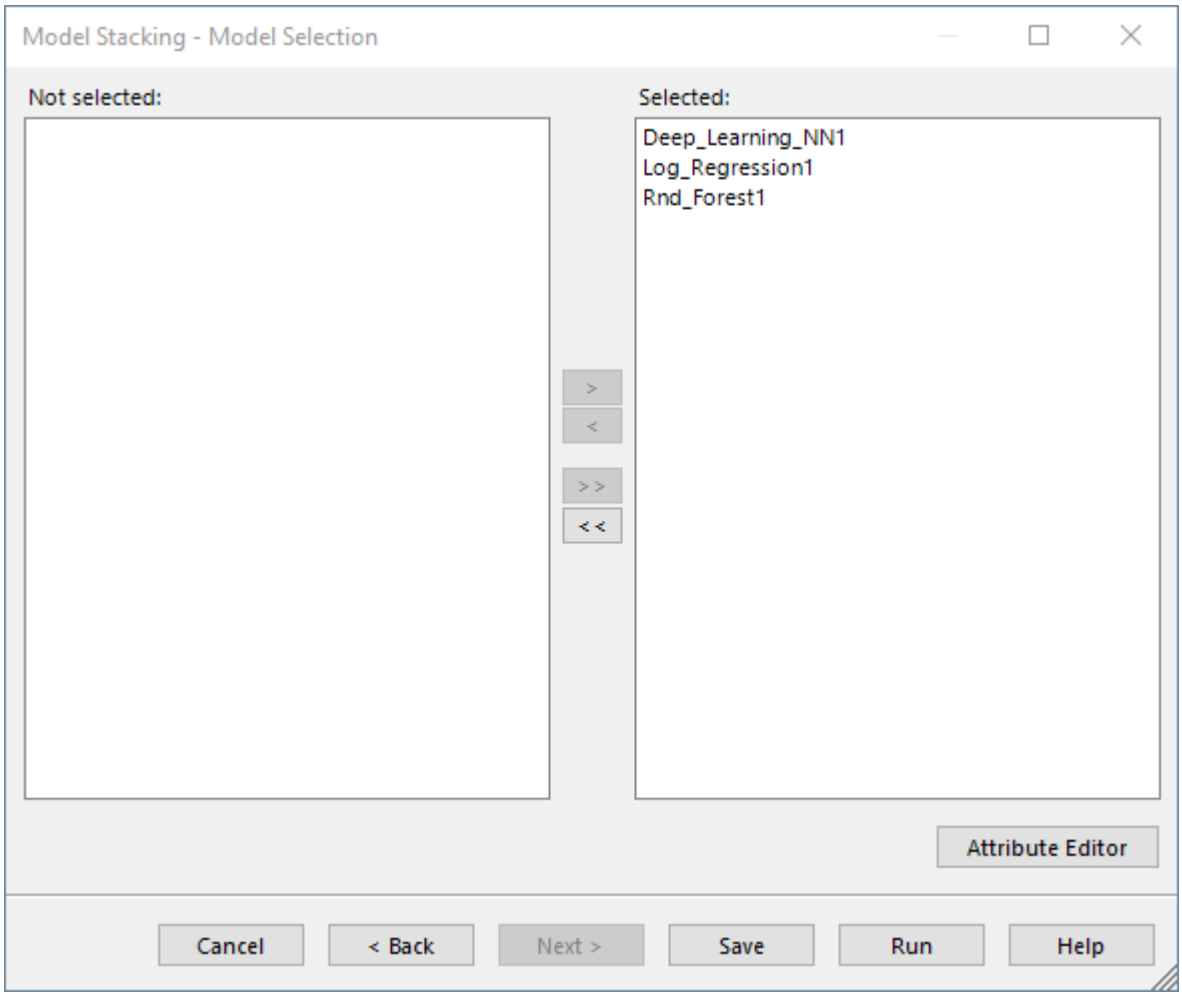
If any of the variables cannot be mapped automatically (for example, if the field names in the input dataset do not match any of the model variable names) the corresponding tab name is highlighted in red, indicating that the user needs to go to that tab and map the fields manually. The *Next* button is disabled until all fields are mapped.



Click **Next** to proceed to the Model Selection page.

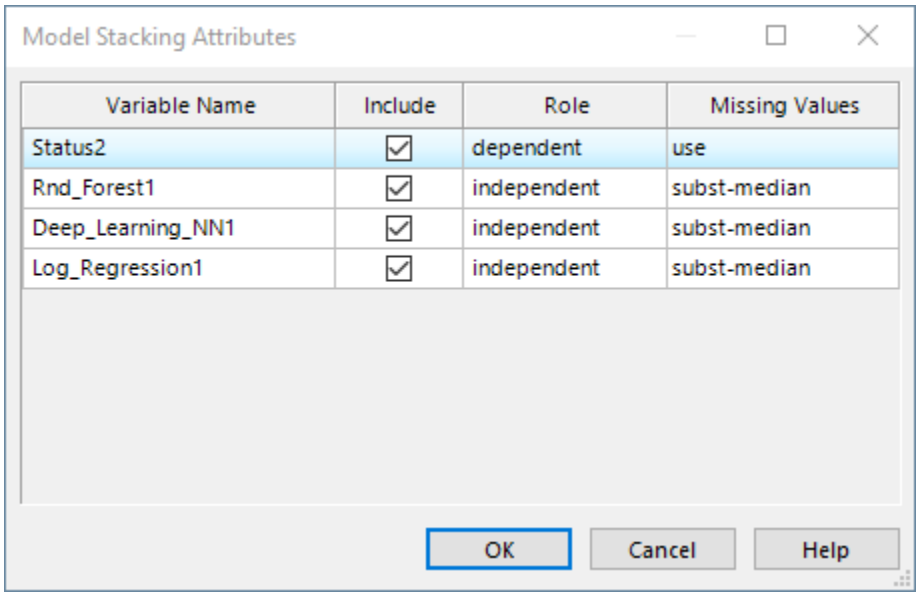
Select the models whose scores you want to be used for building the stacked model. By default, all input models are selected, which means that the scores (DV probability variables) of all models will be used. At least two models must be selected.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click the **Attribute Editor** button if you would like to change the role and missing value substitution options for the scores of the input models and the dependent variable. The models are treated as "variables" here, meaning that their probability scores are the inputs. The available Missing Values options are the same as in any other predictive model: *use, ignore, subst-median, subst-mean, subst-mode, subst-max, subst-min, subst-zero*. See the topic Model Attribute Editor for details.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

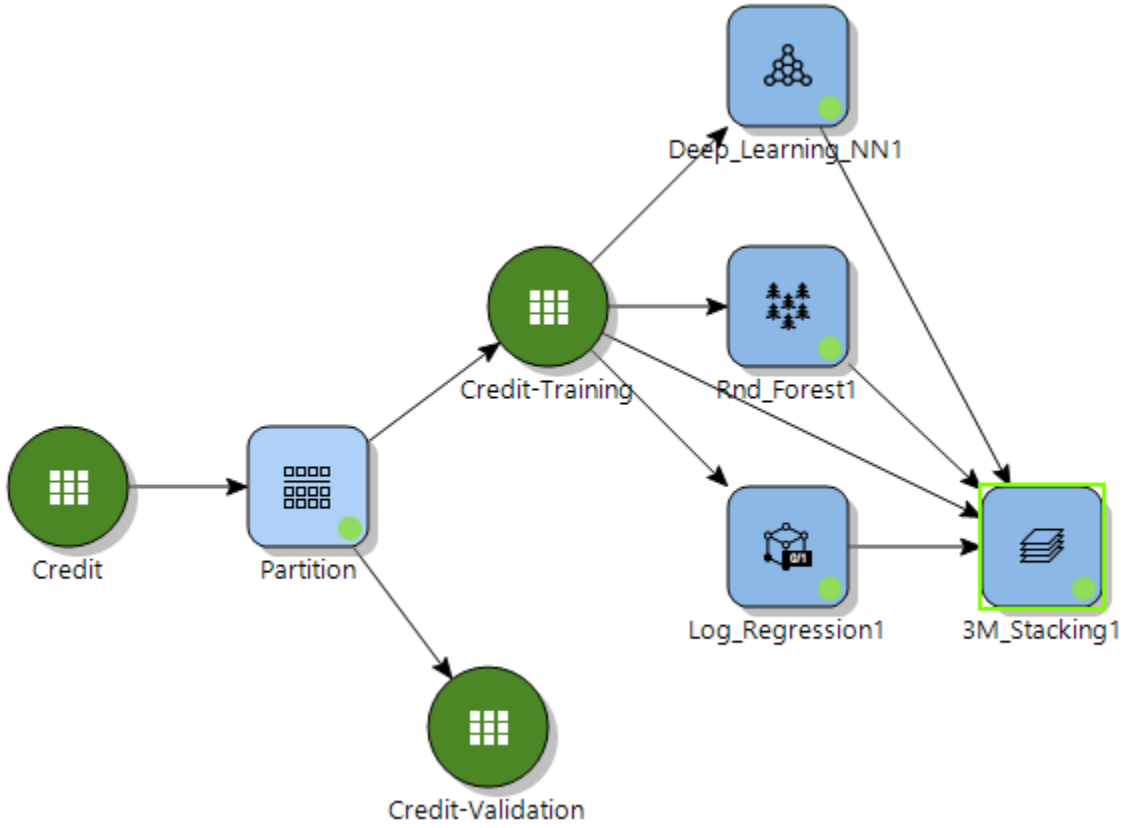


Click **Save** to save the settings, or click **Run** to save the settings and run the scoring process immediately.

The wizard will generate scores for the target category for each base model and build the second-level model on top of the scores (logistic or linear regression, depending on the DV type) to produce the stacking model. For the **Average** model stacking type, it just takes the average of the base learner scores.

Double-click the resulting model node to view the results.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The ***Results*** tab of the output stacking model shows the following information:

1. In the case of a discrete dependent variable:

- Model Fitting Summary for the dependent variable
- Variance Inflation Factors (applicable only for the ***Logistic Regression*** model stacking type)
- Model equation (Formula)

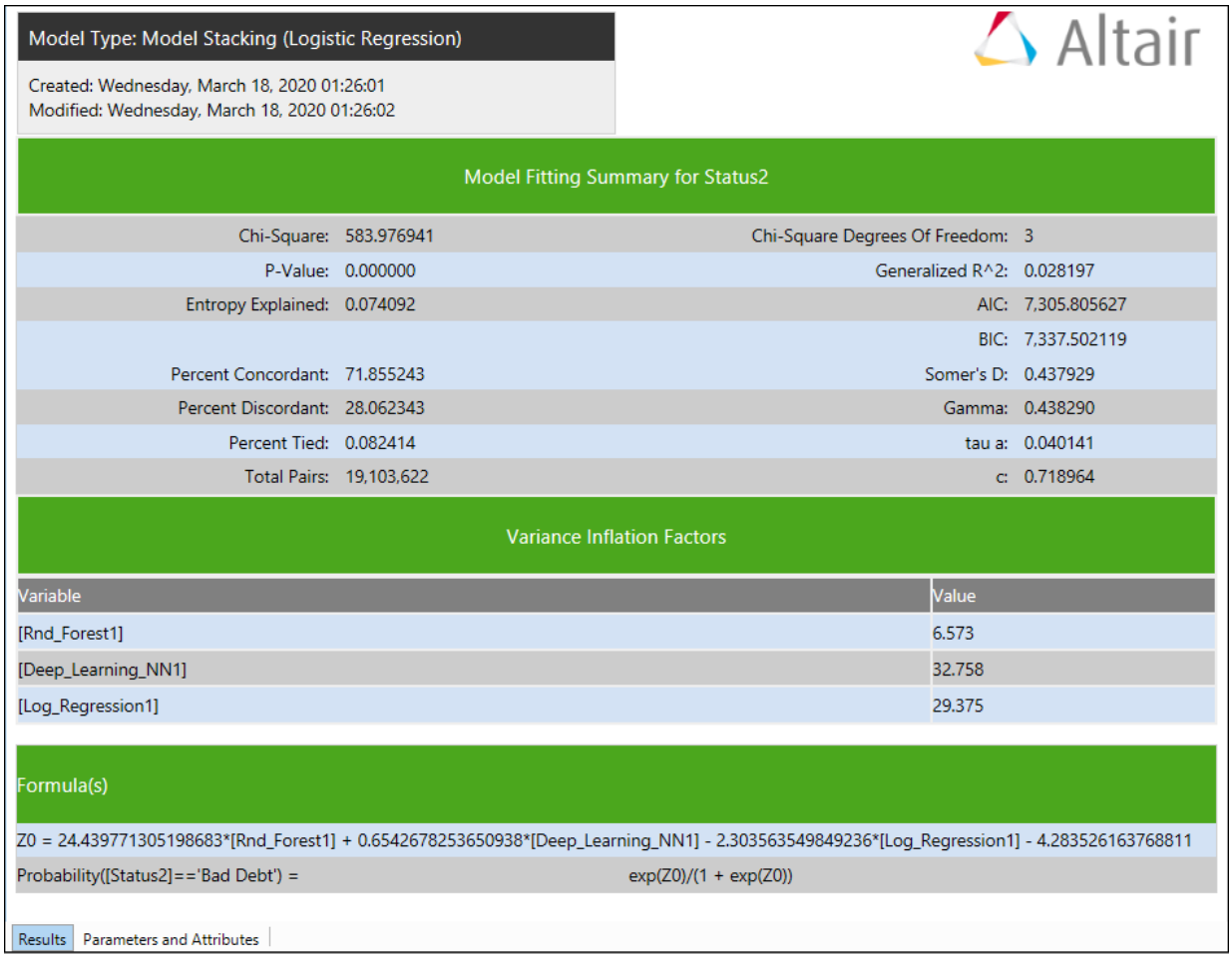
The report lists the input model names as "variables" in the final stacking model, meaning the probability scores of the respective models.

> Note: In the example below, if the ***Average*** model stacking type were selected, the formula would be the following:

```
Probability([Status2]=='Bad Debt') = (RndForest1 + Deep_Learning_NN1 + Log_Regression1) / 3
```

Example 1:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Model Type: Model Stacking (Logistic Regression)

Created: Wednesday, March 18, 2020 01:26:01
Modified: Wednesday, March 18, 2020 01:26:02

Model Fitting Summary for Status2

Chi-Square:	583.976941	Chi-Square Degrees Of Freedom:	3
P-Value:	0.000000	Generalized R^2:	0.028197
Entropy Explained:	0.074092	AIC:	7,305.805627
		BIC:	7,337.502119
Percent Concordant:	71.855243	Somer's D:	0.437929
Percent Discordant:	28.062343	Gamma:	0.438290
Percent Tied:	0.082414	tau a:	0.040141
Total Pairs:	19,103,622	c:	0.718964

Variance Inflation Factors

Variable	Value
[Rnd_Forest1]	6.573
[Deep_Learning_NN1]	32.758
[Log_Regression1]	29.375

Formula(s)

Z0 = 24.439771305198683*[Rnd_Forest1] + 0.6542678253650938*[Deep_Learning_NN1] - 2.303563549849236*[Log_Regression1] - 4.283526163768811

Probability([Status2]=='Bad Debt') = exp(Z0)/(1 + exp(Z0))

Results | Parameters and Attributes

2. In the case of a continuous dependent variable:

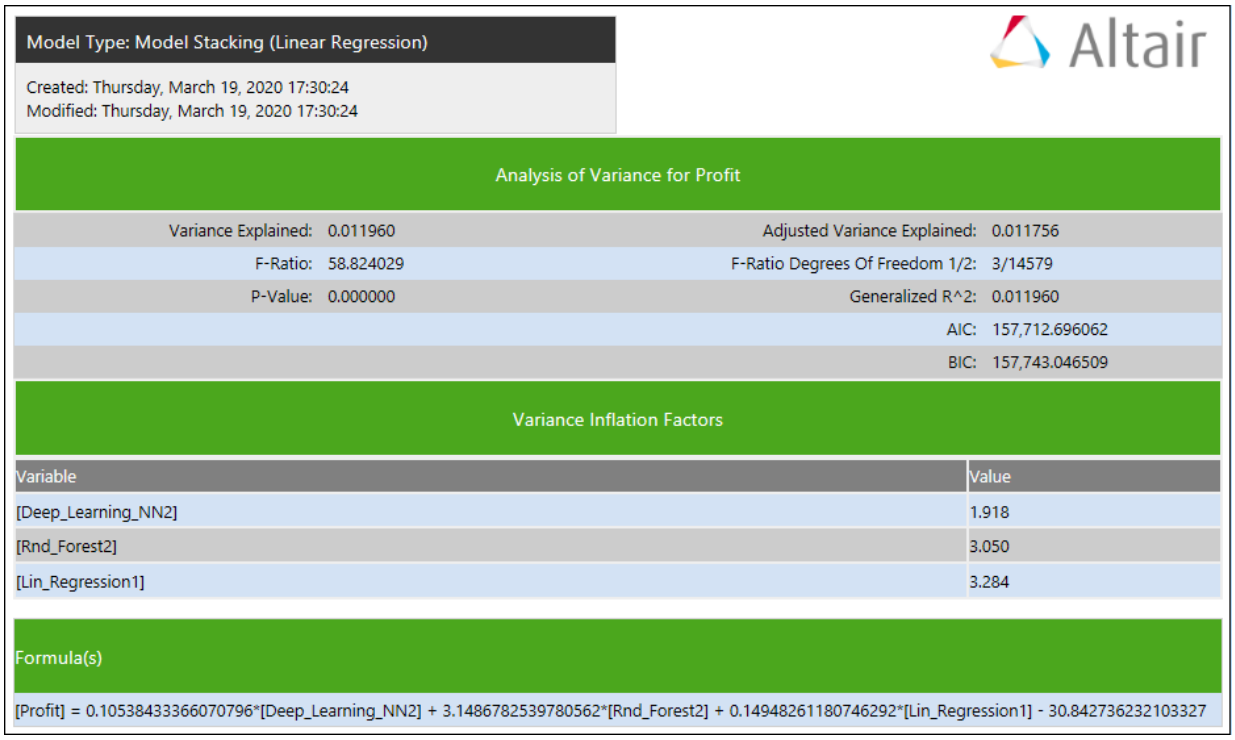
- Analysis of variance for the dependent variable
- Variance Inflation Factors (applicable only for the ***Linear Regression*** model stacking type)
- Model equation (Formula)

The report lists the input model names as "variables" in the final stacking model, meaning the probability scores of the respective models.

> Note: In the example below, if the ***Average*** model stacking type were selected, the formula would be the following:
>
> **`Probability([Status2]=='Bad Debt') = (Deep_Learning_NN2 + RndForest2 + Lin_Regression1) / 3`**

Example 2:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Model Type: Model Stacking (Linear Regression)

Created: Thursday, March 19, 2020 17:30:24
Modified: Thursday, March 19, 2020 17:30:24

Analysis of Variance for Profit

Variance Explained:	0.011960	Adjusted Variance Explained:	0.011756
F-Ratio:	58.824029	F-Ratio Degrees Of Freedom 1/2:	3/14579
P-Value:	0.000000	Generalized R^2:	0.011960
		AIC:	157,712.696062
		BIC:	157,743.046509

Variance Inflation Factors

Variable	Value
[Deep_Learning_NN2]	1.918
[Rnd_Forest2]	3.050
[Lin_Regression1]	3.284

Formula(s)

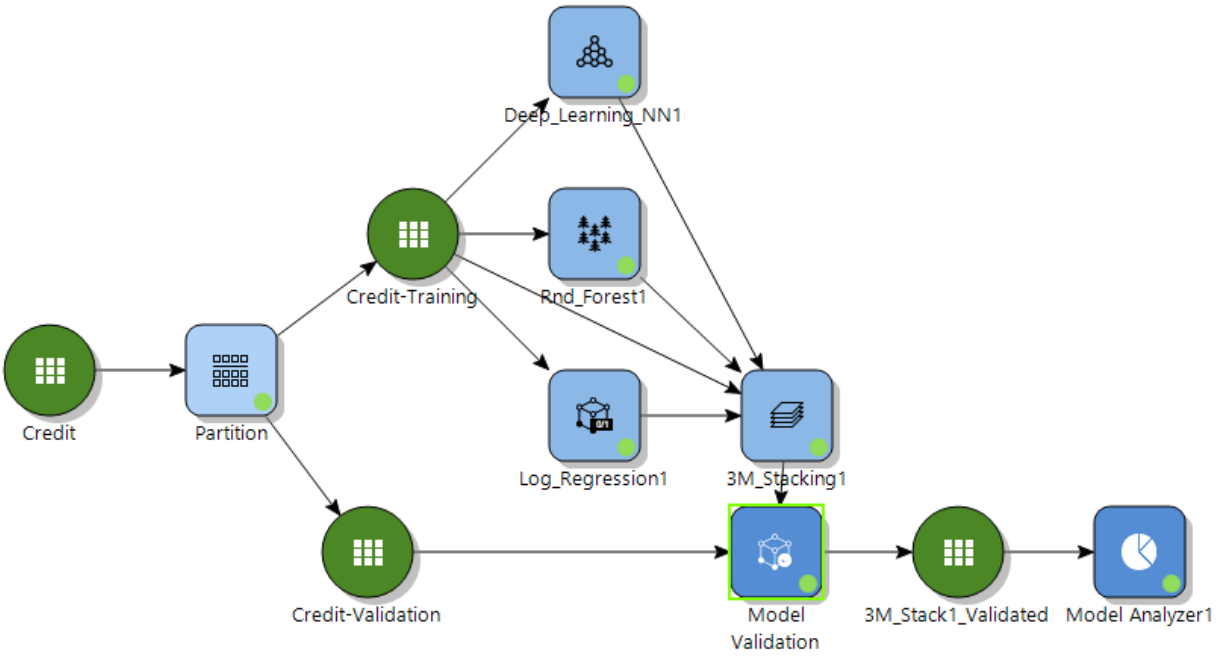
[Profit] = 0.1053843366070796*[Deep_Learning_NN2] + 3.1486782539780562*[Rnd_Forest2] + 0.14948261180746292*[Lin_Regression1] - 30.842736232103327

Scoring and Validation for Model Stacking

Model Validation and **Scoring** nodes can be applied to the Model Stacking node just like any other predictive model type. The base model formulas are preserved and substituted into the model formula of the Model Stacking node when scoring or validating a new data set. For example, if a Deep Learning model and Random Forest models were used for Model Stacking with Logistic Regression, then, when scoring or validating a new dataset, the Model Stacking node recalls the formulas of its base models and uses them in its formula to apply to the new dataset and produce the final scores.

Example 3: Model Validation and Model Analyzer applied to a Model Stacking node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also:

- Model Selector
- About Scoring

Novelty & Outlier Detector

Overview

The **Sklearn Novelty & Outlier Detector** node in the **Model** palette of Knowledge Studio helps determine whether a set of observations belong to the same distribution as the other observations in a dataset or should be classified as outliers. This is useful for data cleansing to prepare data for modeling. An observation is considered to be an outlier if its distance (in terms of a specific metric) from the other observations in a dataset exceeds a certain threshold. Outlier detection estimators try to fit the regions where the data in the training dataset is the most concentrated, ignoring the deviant observations.

When the training data is assumed to be not polluted by outliers, the purpose is to detect whether a **new** observation is an outlier. In this context, an outlier is also called a ***novelty***.

The **Sklearn Novelty & Outlier Detector** node requires a single input dataset and produces one or two output datasets (depending on the method used) with columns containing anomaly metrics that determine the outlier status. In the cases where two outputs are produced, he first output dataset contains all original records, while the second one contains only non-outliers.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

This feature works through integration with Python libraries. It uses functions of **`scikit-learn`** - a popular Python library for machine learning. For details, see the scikit-learn documentation for Isolation Forest, Local Outlier Factor, and One Class SVM algorithms (Copyright © 2007 - 2021, scikit-learn developers, BSD License). The node wizard provides a convenient graphical interface to specify the options and parameters for the required `sklearn` functions. The wizard automatically translates the inputs into Python code, which can be previewed before running the node. The code then runs in a container with the necessary Python packages and their dependencies. The input dataset is converted from the Altair KDD format to pandas dataframes in the process. The resulting dataframe is converted back to produce the output Altair dataset.

After training, the model can be applied to the observations in a new dataset using the Sklearn Novelty & Outlier Scoring node.

Inserting a Novelty & Outlier Detector node

An **Sklearn Novelty & Outlier Detector** model requires a single dataset as input and can be added to the workflow in one of the following ways:

- Drag a connection arrow from the input dataset node to any area on the Workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Model --> Sklearn Novelty & Outlier Detector**. A new **Sklearn Novelty & Outlier Detector** node will be created, and the wizard will open.

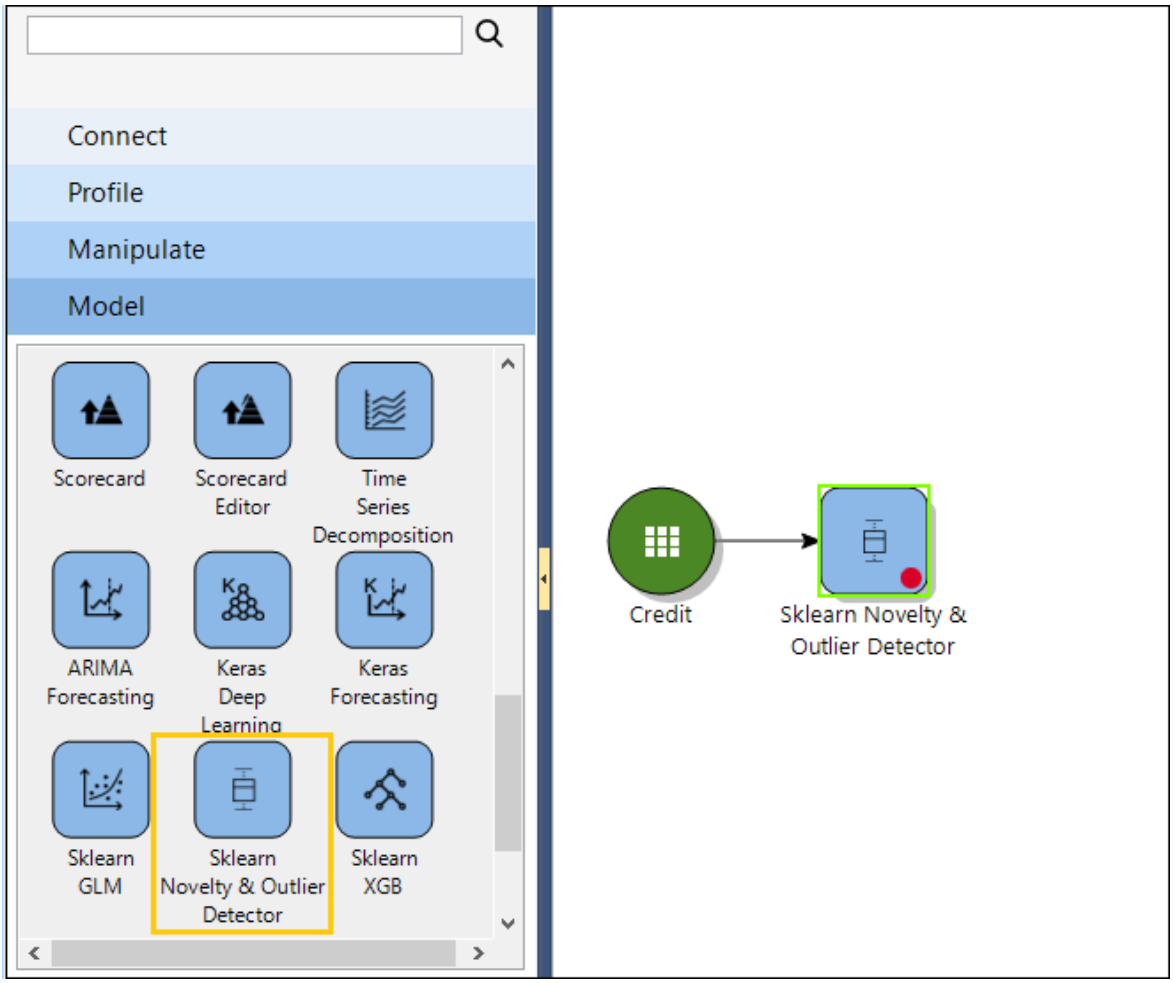
  OR

- Right-click on the Workflow canvas and select **Model --> Sklearn Novelty & Outlier Detector** from the context menu. Connect the training dataset.

  OR

- Open the **Model** palette and drag the **Sklearn Novelty & Outlier Detector** node onto the workflow canvas. Connect the input dataset to the new node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



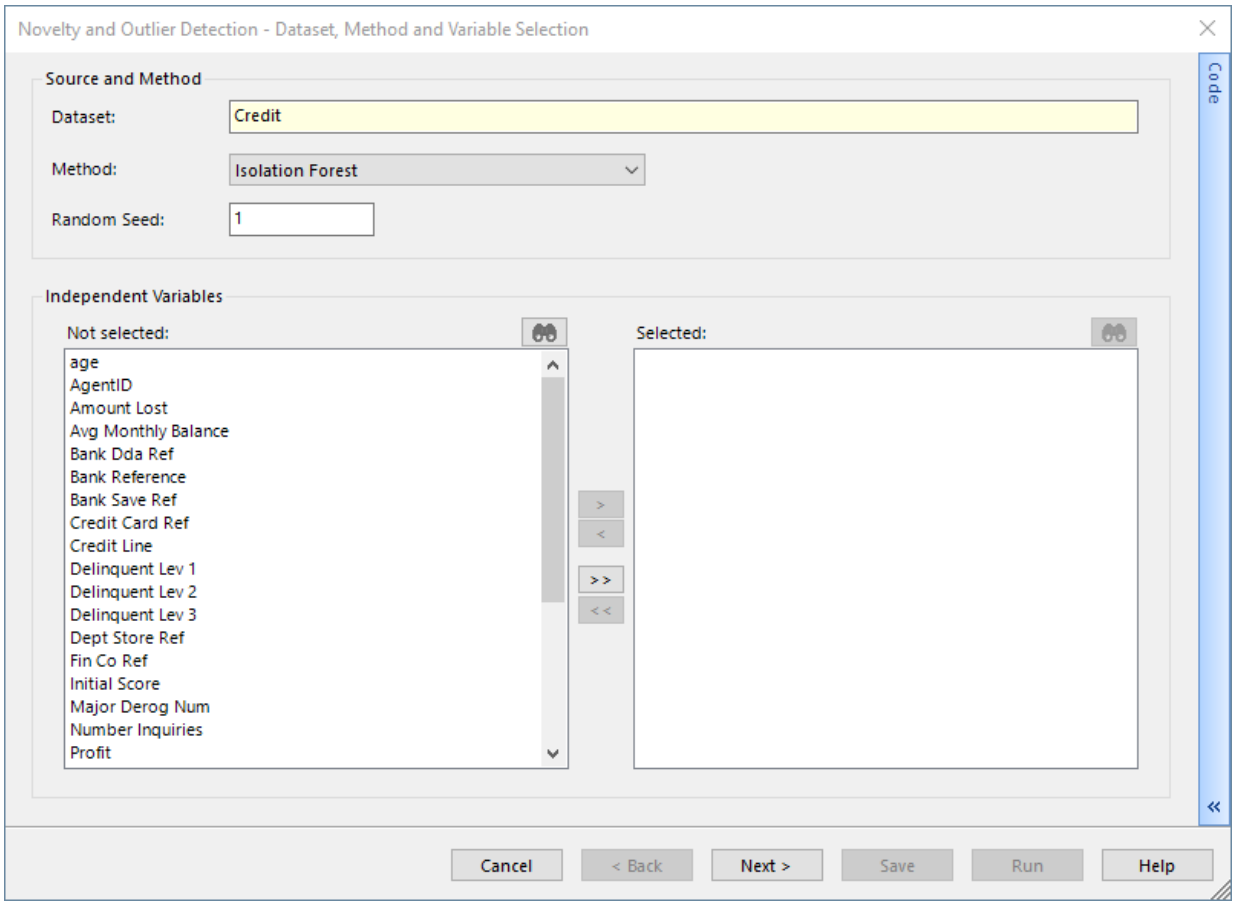
If the **Sklearn Novelty & Outlier Detector** node is already on the canvas, double-click it or right-click on it and select **Modify** from the context menu to open the wizard.

In the first page of the wizard:

- Specify the method for novelty & outlier detection. The available methods are *Isolation Forest*, *Local Outlier Factor*, and *One Class SVM*.
- Specify the Random Seed value for the repeatability of results, if necessary.
- Select the desired independent variables to analyze for outliers.

Tip: Advanced field finding and multi-selection capabilities are also available via the Field Finder dialog, which can be opened by clicking the binoculars button.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



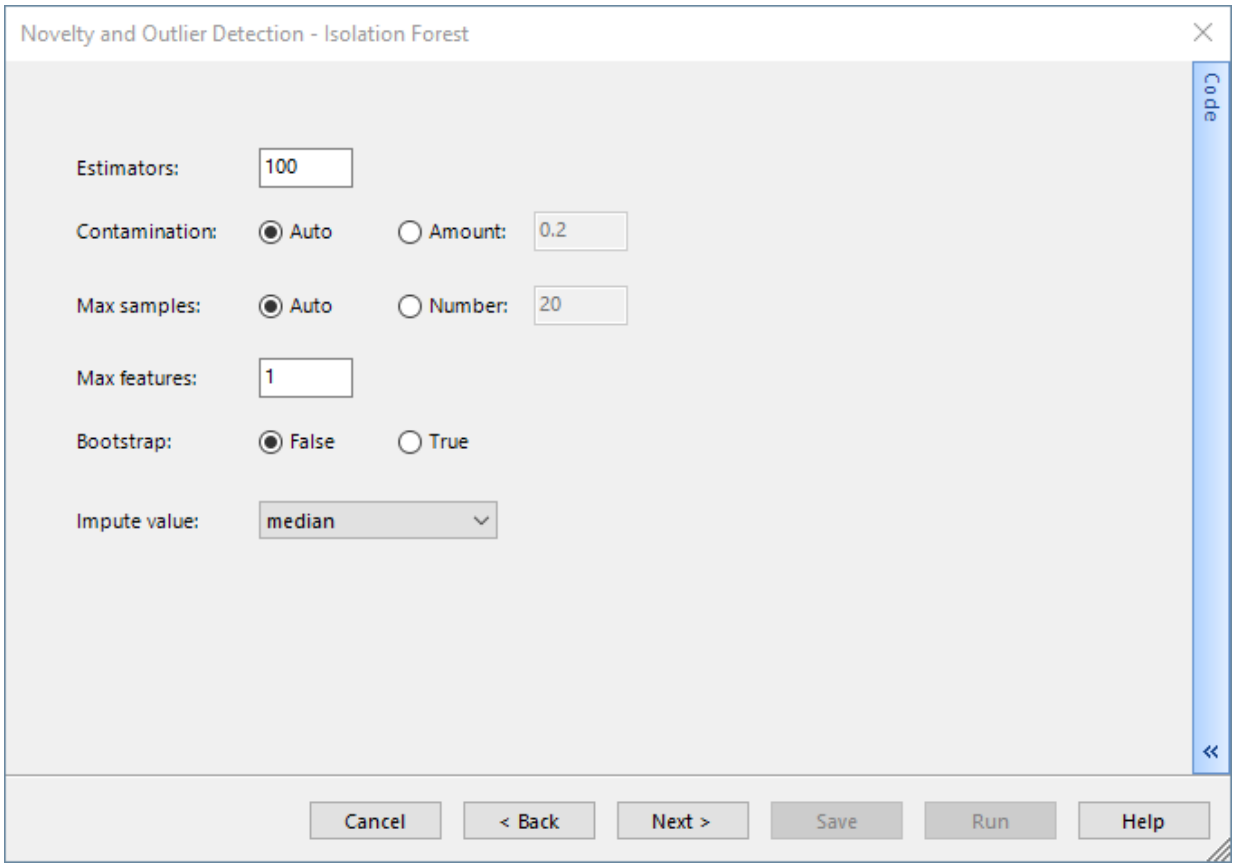
Click **Next**.

The options in the next page depend on the chosen method.

For the ***Isolation Forest*** method, specify the following options and parameters. For details, see the description of the Isolation Forest Algorithm parameters in the Scikit-learn documentation.

- Specify the number of *estimators* in the ensemble. The default is 100.
    - Specify the amount of *contamination* (the proportion of outliers) in the dataset: choose Auto (automatic) or specify the desired amount.
    - Select the *maximum number of samples*: choose Automatic (Auto) or specify the desired number.
- Specify the *maximum number of features* to draw to train each base estimator. The default is 1.
- Set the *Bootstrap* option to True if you need sampling with replacement, otherwise set it to False.
- Select the *imputation strategy* for preprocessing. The available imputation options are: *median*, *mean*, and *mode*. The default value is *median*.

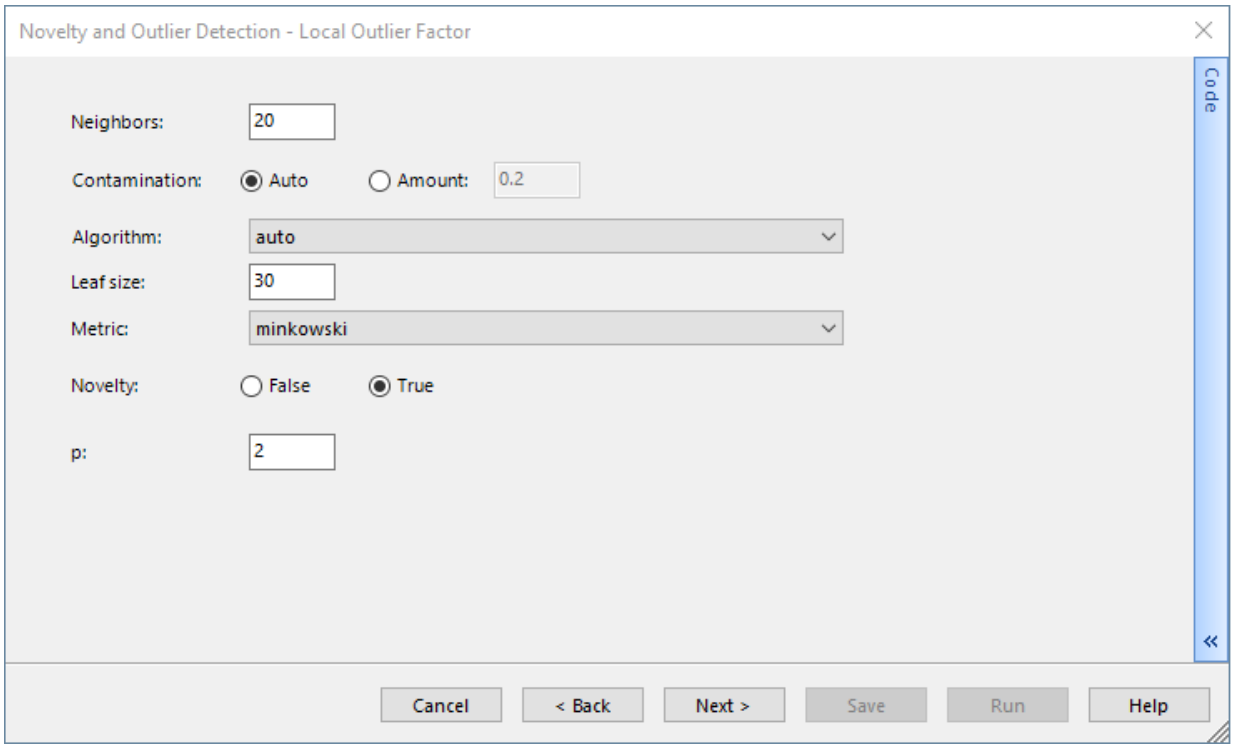
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



For the ***Local Outlier Factor*** method, specify the following options and parameters. For details, see the description of the Local Outlier Factor Algorithm parameters in the Scikit-learn documentation.

- Specify the number of *neighbors* for samples. The default is 20.
- Specify the amount of *contamination* (the proportion of outliers) in the dataset: choose Auto (automatic) or specify the desired amount.
- Choose the *algorithm* used to compute the nearest neighbors. The available options are *ball_tree*, *kd_tree*, *brute* (brute-force search), and *auto.* The default is *auto.*
- Specify the *leaf size* passed to ball_tree or kd_tree. The default is 30.
- Choose the m*etric* used to compute the distance. The available metrics are: *minkowski*, *euclidean*, and *manhattan.* The default is *minkowski.*
- Set the *Novelty* option to True if you want to use the method for novelty detection. In this case it is assumed that the training data is not polluted by outliers, and you are interested in detecting whether a new observation is an outlier..
- Specify the value of the parameter *p* for the Minkowski metric from sklearn.metrics.pairwise.pairwise_distances. The default is 2.

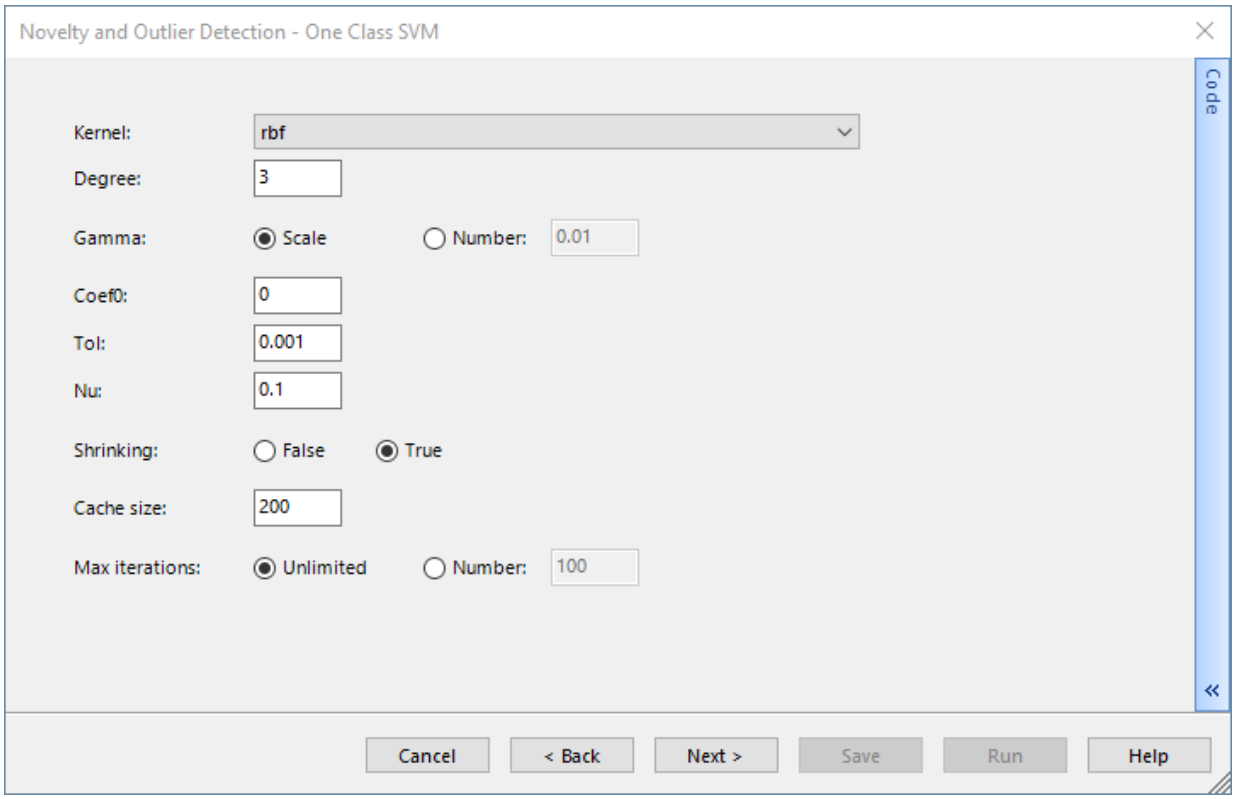
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



For the ***One Class SVM*** method, specify the following options and parameters. For details, see the description of the One Class SVM algorithm parameters in the Scikit-learn documentation.

- Select the *kernel type*. The available options are *rbf*, *linear*, *poly*, *sigmoid*, and *precomputed*. The default is *rbf*.
    - Specify the *degree* of the polynomial kernel function. The default is 3.
- Specify the *Gamma* coefficient for rbf, poly or sigmoid kernel types. 'Scale' is selected by default.
- Specify *Coef0* - the Independent term in the kernel function (for ‘poly’ or ‘sigmoid’ kernel). The default is 0.
- Specify the value of tolerance *tol* for the stopping criterion. The default is 0.001.
- Specify the value of *Nu* - the upper bound on the fraction of training errors and a lower bound of the fraction of support vectors. The default is 0.1
- For the *shrinking* option, select True to use the shrinking heuristic. Otherwise, select False.
- Specify the kernel *cache size* (in MB). The default is 200 MB.
- Specify the *maximum number of iterations* within the solver. The default is *Unlimited*.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

- Select the original fields to be included in the output dataset(s).
- Specify which output datasets are to be created. The output depends on the detection method.

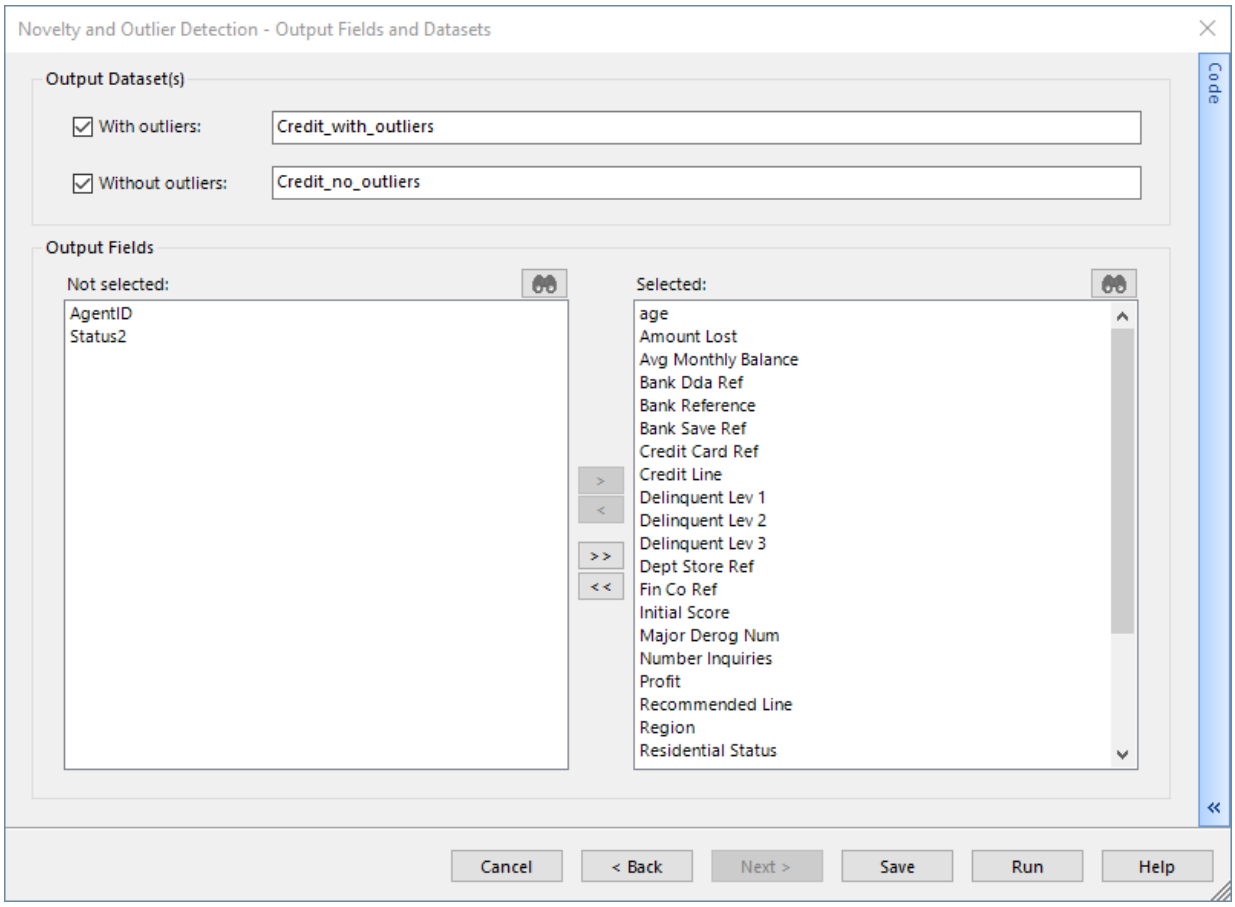
    - For the Isolation Forest method, two output datasets are available:
        - The dataset with outliers contains all records of the original dataset, the ones classified as outliers being flagged according to their computed *Anomaly_Score* and *Oposite_Anomaly_Score,* shown in two additional fields.
            - The *Anomaly_Score* column contains the computed anomaly score values for each record, followed by the *Opposite_Anomaly_Score* column. The *Anomaly_Flag* column displays -1 for outliers and 1 for non-outliers.
        - The second dataset contains only non-outlier records. It includes the *Anomaly_Score* and *Oposite_Anomaly_Score* columns.

    - For the *One Class SVM* method, two output datasets are available:
        - The dataset with outliers contains all records of the original dataset and three additional columns.
            - The *Anomaly_Flag* column displays -1 for outliers and 1 for non-outliers.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

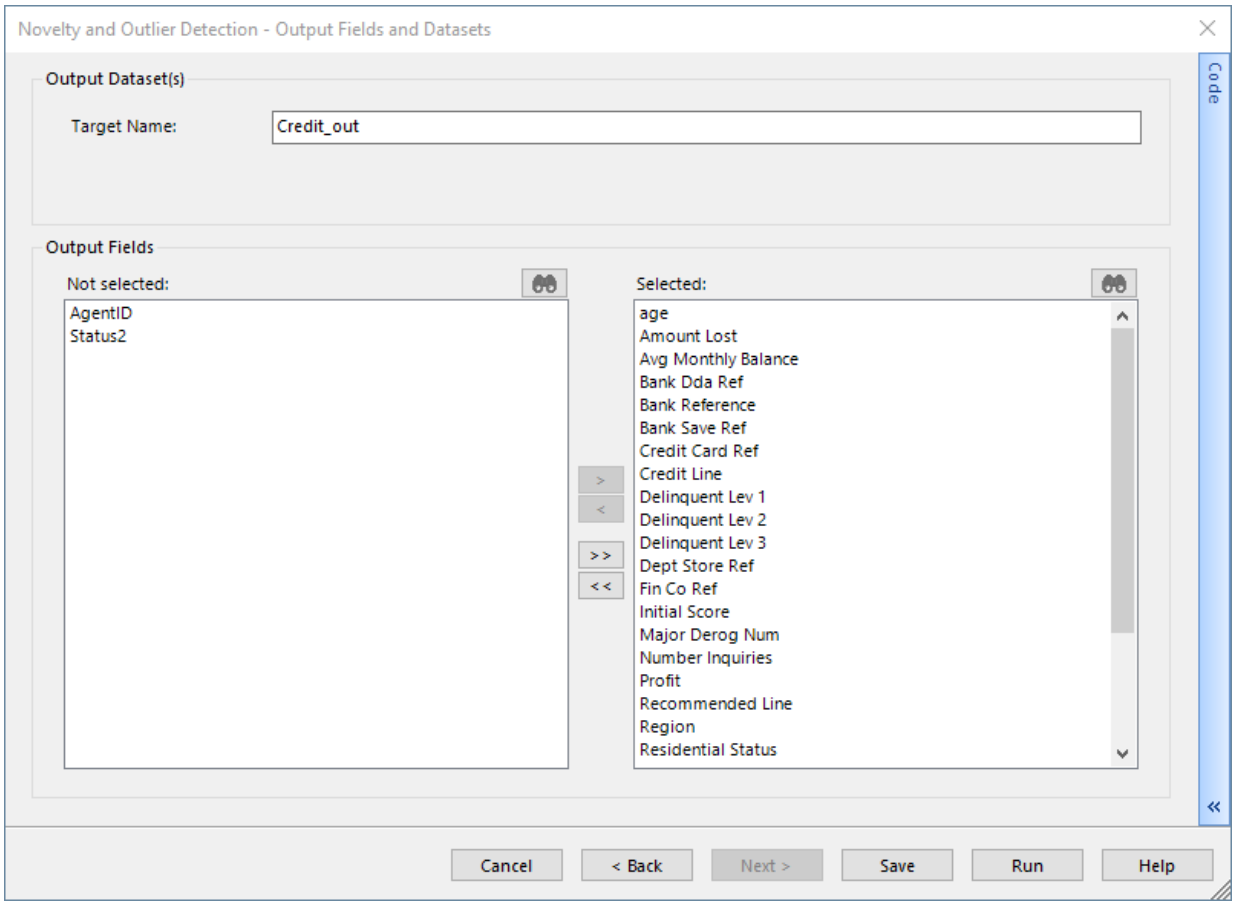
- The *Separating_hyperplane_distance* contains the computed score values for each record.
- The *Scoring_Function* column.

- The second dataset contains only non-outlier records. It includes two additional columns: *Separating_hyperplane_distance* and *Scoring_Function*.

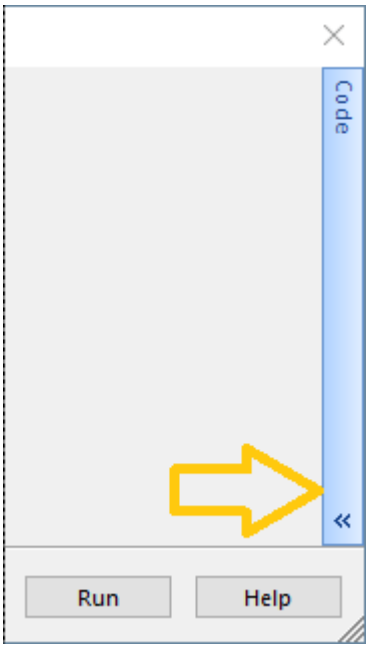


- For the Local Outlier Factor method, there is a single output dataset.
  - If the *Novelty* option was set to True, the output dataset will have three new columns: the *Anomaly_Score* column contains the computed anomaly score values for each record, followed by the *Opposite_Anomaly_Score* column. The *Anomaly_Flag* column displays -1 for anomaly and 1 for not an anomaly.
  - If the *Novelty* option was set to False, the output dataset has one new column: *Negative Outlier Factor*.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

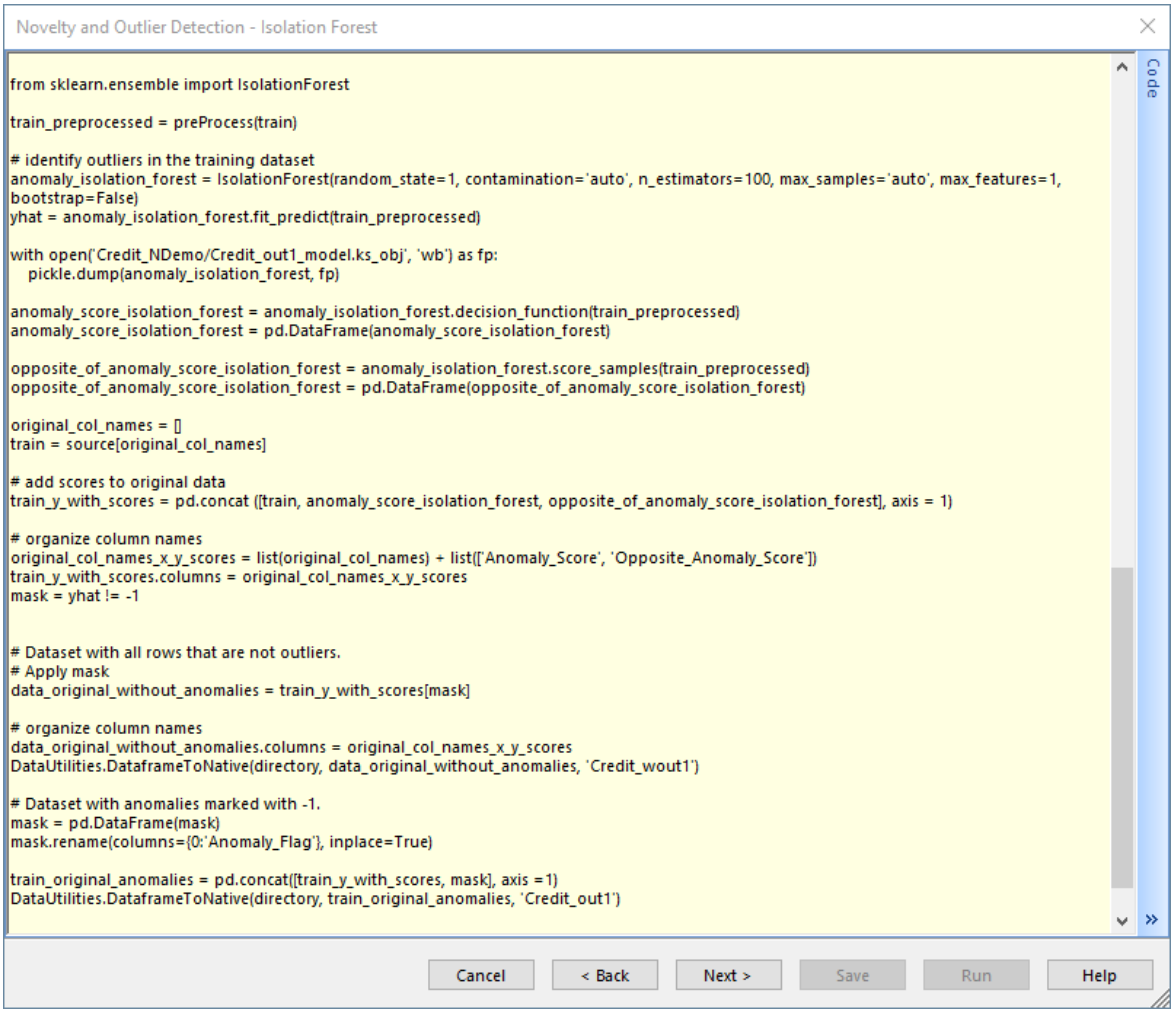


To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

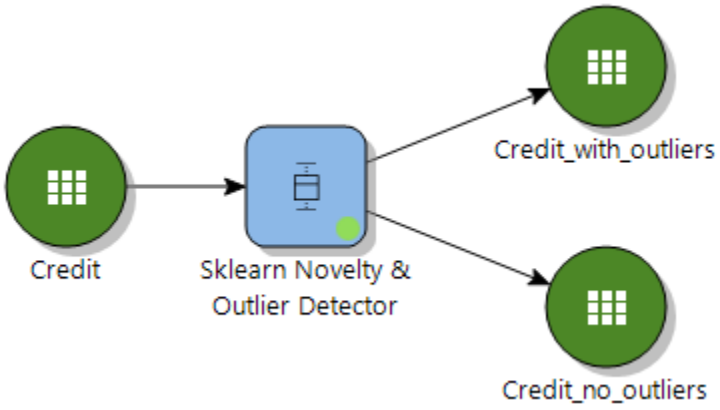
This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key. Click on the blue Code bar again to close the code preview panel.



Click **Save** to save the settingsor click Run to save the settings and run the process immediately. The code generated from your inputs runs in a Python container session. The input dataset is converted to a Pandas dataframe, and the required functions of the `sklearn` library are applied. The output data is converted from Pandas to the Altair KDD format and is shown as the output dataset node at the end.

Double-click the output dataset node(s) to view the results.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click the Data tab to view the new columns added.

Example 1: Output dataset with outliers (Isolation Forest method). The *Anomaly_Flag* field is the outlier indicator based on the values of *Anomaly_Score* and *Opposite_Anomaly_Score*.

	Total Service	Worst Rating Des	Year at Address	Year Last Check	Anomaly_Score	Opposite_Anomaly_Score	Anomaly_Flag
1	1	3 satisfactory ratings	10	1	0.000867930621737345	-0.499132069378263	☑
2	2	(null)	(null)	(null)	0.0313802700330305	-0.468619729966969	☑
3	3	1 satisfactory rating	23	1	0.0132539822796954	-0.486746017720305	☑
4	6	(null)	(null)	(null)	0.0255671819827542	-0.474432818017246	☑
5	0	(null)	(null)	(null)	0.0273426656560493	-0.472657334343951	☑
6	111	5 or more satisfactory ratings	(null)	0	-0.0243173989998243	-0.524317398999824	☐
7	284	(null)	10	(null)	0.00673247165971619	-0.493267528340284	☑
8	0	(null)	(null)	(null)	0.0292755414470232	-0.470724458552977	☑
9	47	File but no rating	1	0	-0.00098775065024248	-0.500987750650242	☐
10	38	File but no rating	3	98	-0.0157806793598609	-0.515780679359861	☐
11	0	2 satisfactory ratings	(null)	1	0.00701388289900789	-0.492986117100992	☑
12	115	2 satisfactory ratings	1	0	0.0154828208525022	-0.484517179147498	☑
13	157	No File	(null)	99	0.00136647916434529	-0.498633520835655	☑
14	0	(null)	(null)	(null)	0.0274657975381126	-0.472534202461887	☑
15	0	(null)	(null)	(null)	0.0246972324736272	-0.475302767526373	☑

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

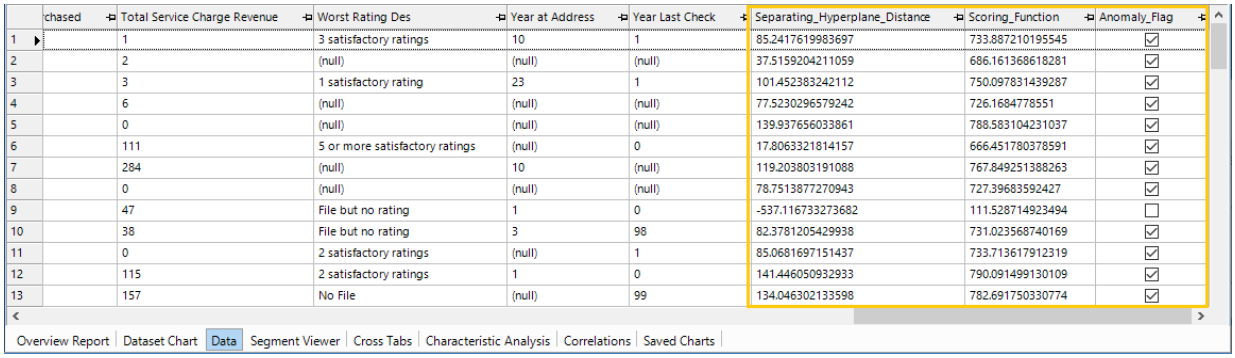
Example 2: Output dataset without outliers (Isolation Forest method). The records classified as outliers are removed. The fields *Anomaly_Score* and *Opposite_Anomaly_Score* show the anomaly metrics for each record.

	hased	Total Service Charge Revenue	Worst Rating Des	Year at Address	Year Last Check	Anomaly_Score	Opposite_Anomaly_Score
1		1	3 satisfactory ratings	10	1	0.000867930621737345	-0.499132069378263
2		2	(null)	(null)	(null)	0.0313802700330305	-0.468619729966969
3		3	1 satisfactory rating	23	1	0.0132539822796954	-0.486746017720305
4		6	(null)	(null)	(null)	0.0255671819827542	-0.474432818017246
5		0	(null)	(null)	(null)	0.0273426656560493	-0.472657334343951
6		284	(null)	10	(null)	0.00673247165971619	-0.493267528340284
7		0	(null)	(null)	(null)	0.0292755414470232	-0.470724458552977
8		0	2 satisfactory ratings	(null)	1	0.00701388289900789	-0.492986117100992
9		115	2 satisfactory ratings	1	0	0.0154828208525022	-0.484517179147498
10		157	No File	(null)	99	0.00136647916434529	-0.498633520835655
11		0	(null)	(null)	(null)	0.0274657975381126	-0.472534202461887
12		0	(null)	(null)	(null)	0.0246972324736272	-0.475302767526373
13		0	(null)	(null)	(null)	0.026059262103508	-0.473940737896492
14		0	(null)	(null)	(null)	0.034842179753759	-0.465157820246241
15		0	(null)	(null)	(null)	0.0306606035987099	-0.46933939640129

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Example 3: Output dataset with outliers (One Class SVM method).



	chased	Total Service Charge Revenue	Worst Rating Des	Year at Address	Year Last Check	Separating_Hyperplane_Distance	Scoring_Function	Anomaly_Flag
1		1	3 satisfactory ratings	10	1	85.2417619983697	733.887210195545	☑
2		2	(null)	(null)	(null)	37.5159204211059	686.161368618281	☑
3		3	1 satisfactory rating	23	1	101.452383242112	750.097831439287	☑
4		6	(null)	(null)	(null)	77.5230296579242	726.1684778551	☑
5		0	(null)	(null)	(null)	139.937656033861	788.583104231037	☑
6		111	5 or more satisfactory ratings	(null)	0	17.8063321814157	666.451780378591	☑
7		284	(null)	10	(null)	119.203803191088	767.849251388263	☑
8		0	(null)	(null)	(null)	78.7513877270943	727.39683592427	☑
9		47	File but no rating	1	0	-537.116733273682	111.528714923494	☐
10		38	File but no rating	3	98	82.3781205429938	731.023568740169	☑
11		0	2 satisfactory ratings	(null)	1	85.0681697151437	733.713617912319	☑
12		115	2 satisfactory ratings	1	0	141.446050932933	790.091499130109	☑
13		157	No File	(null)	99	134.046302133598	782.691750330774	☑

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

The Isolation Forest and One Class SVM methods can be used to clean the input dataset from anomalies. The output dataset without outliers / anomalies can be used as a clean dataset for modeling.

The output of the Local Outlier Factor method is a single dataset that includes all records of the original data. It is assumed that all records are not outliers (clean dataset). Use the Sklearn Novelty & Outlier Scoring node to check a new dataset for anomalies.

See also: Novelty & Outlier Scoring.

AutoML

AutoML automates machine learning by creating a complete modeling workflow starting from data preparation (variable selection and data transformations), to modeling, validation, and selection of the best model for deployment. AutoML makes it possible for users of all skill levels to easily build deployable predictive models.

An **AutoML** node can be added to the workflow in one of the following ways:

- Drag a connection arrow from a dataset node to any area on the Workflow canvas background. Select **Model --> AutoML** from the context menu that opens upon releasing the mouse button, as in the example screenshot below. A new **AutoML** node will be created.

OR

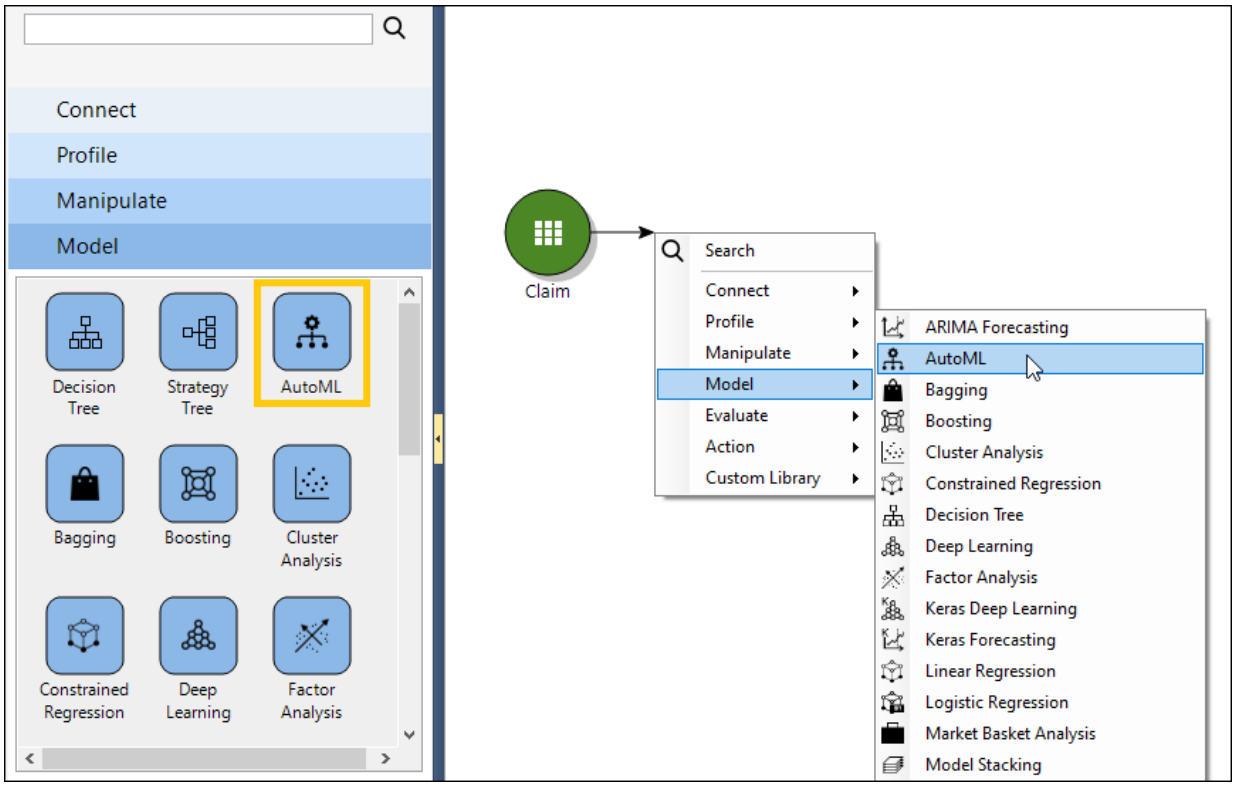
- Right-click on the Workflow canvas and select **Model --> AutoML** from the context menu. Connect a dataset to the **AutoML** node.

OR

- Open the **Model** palette and drag the **AutoML** node onto the workflow canvas. Connect the input dataset to the **AutoML** node.

You can also click on the canvas and use the **CTRL + I** keyboard shortcut to search and insert the "AutoML" node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



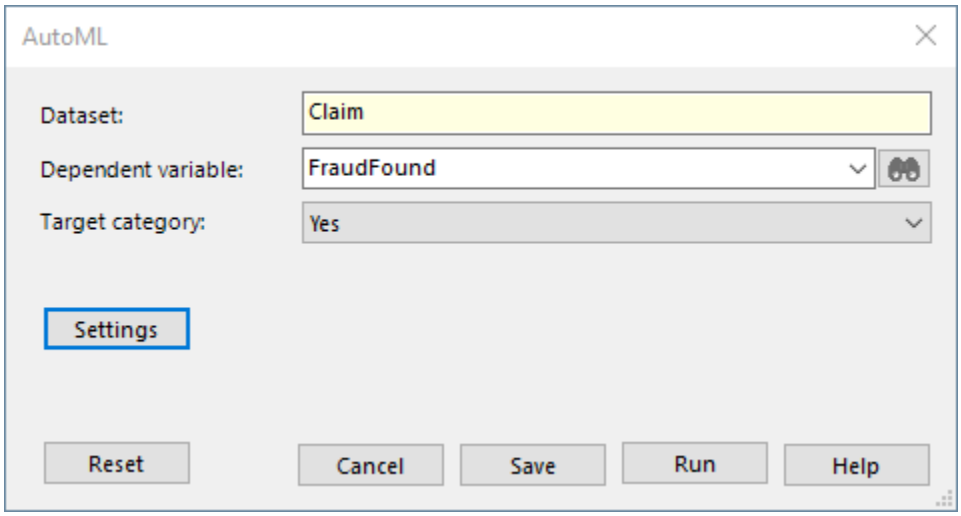
If the AutoML node is already on the canvas, double-click it or right-click on it and select **Modify** from the context menu to open the wizard.



In the AutoML wizard:

- Select the **dependent variable**.
- If the selected dependent variable is discrete, select the **target category**. The target category is required for the stages of model validation, model analysis and comparison (Model Analyzer), and best model selection (Model Selector).
    - *Note: Numeric variables with cardinality higher than 10 are considered continuous. Target category is not applicable for continuous variables.*
- Click **Settings** to define the data preparation and modeling options for automated machine learning.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



AutoML Settings

The **Settings** dialog allows you to define the variable selection options and choose the types and parameters of the models to be built.

1. Variable Selection

The functions of the **Variable Selection** tab allow the user to configure the Variable Selection node settings in the workflow to be built by AutoML. You can either manually select the desired independent variables here or configure the automatic selection mode.

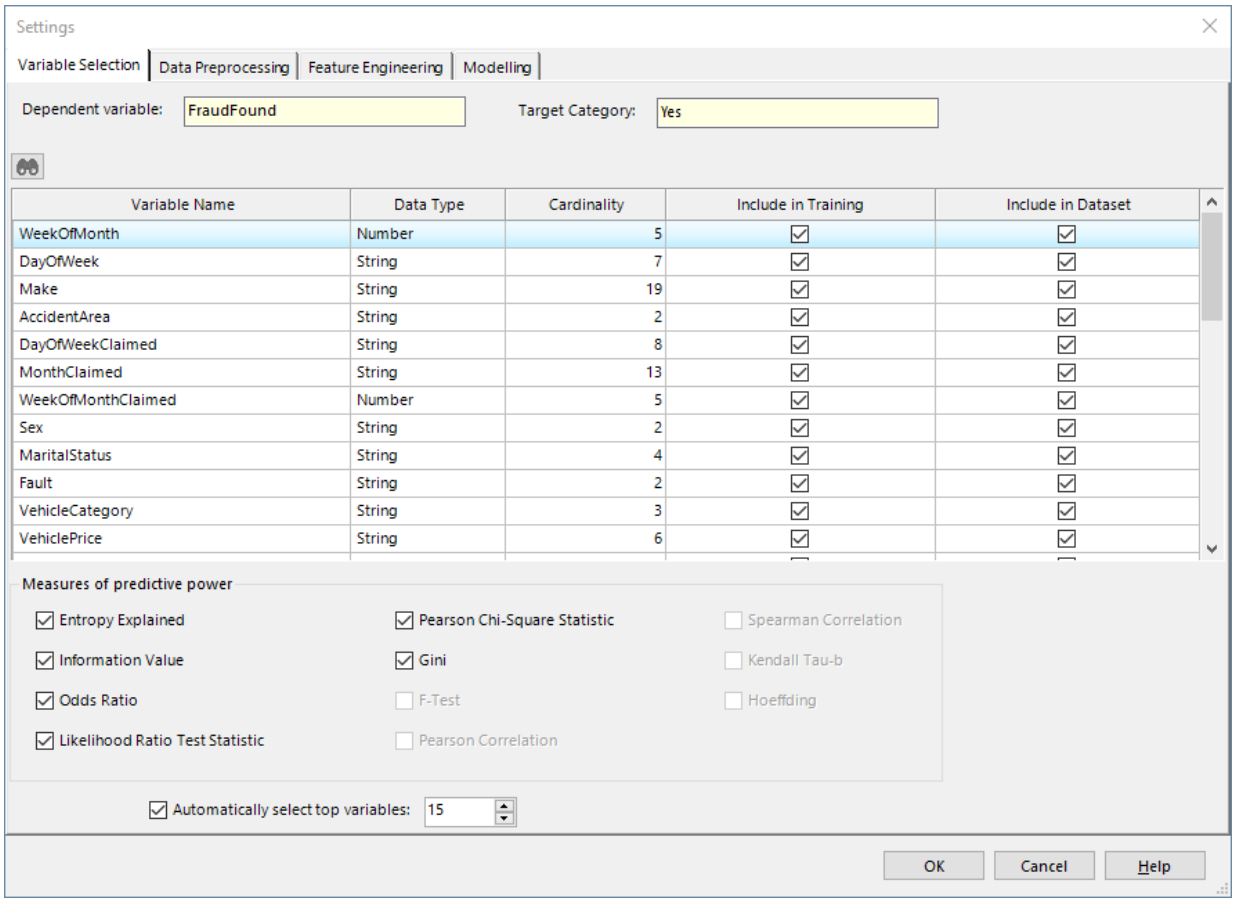
- Any variable selected in the column **Include in Training** will be included in the data preparation steps, model training, and in the final output datasets. The option "**Automatically select top variables**" overrides this setting.
- If you want a variable to be included in the final output datasets, but excluded from the data preparation and model training process, deselect it in the column **Include in Training** and select it in the column **Include in Dataset**.

Note that the measures of predictive power that you select here are not analyzed and calculated in this tab, but will be calculated in the Variable Selection node to be built in the AutoML process.

- Select the desired measures of predictive power to be calculated in the Variable Selection node to be created by AutoML. If the automatic selection option is chosen (see below), the calculated values of these measures will be used to automatically select the independent variables.
- If desired, select the option "**Automatically select top variables:**" and specify the number of variables requested. If more than one measure of predictive power is chosen, then the ***Union*** criterion will be applied in the Variable Selection node to be built. For example, if you request to automatically select 5 variables, then for each measure selected above, the Variable Selection node will pick top 5 variables with the highest values of this measure, then it will take the union of these quintets over all selected measures, and the resulting set will be selected. There may be 5 or more variables selected as a result. See the Variable Selection topic for details.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Any variable excluded from training in the automatic selection process is still included in the output datasets unless it is deselected in the column **Include in Dataset**.



Note: If there are a lot of variables (hundreds or thousands), click the binoculars button  *to use the* Field Finder*.*

2. Data Preprocessing

The **Data Preprocessing** tab allows the user to define the missing values substitution and the outliers treatment options for the Variable Transformations nodes to be built by AutoML.

The functions for the missing values substitution are the same as the existing Substitute Missing Values Helper. Choose whether to enable the substitution for continuous variables, discrete variables, or both.

- For *Continuous*, the options are: Mean, Median, Min, Max, Mode, and Zero. The default is *Substitute with Mean.*
- For *Discrete*, the options are: Mode, Max, and Min. The default is *Substitute with Mode*.

If the **Outliers Treatment** option is enabled:

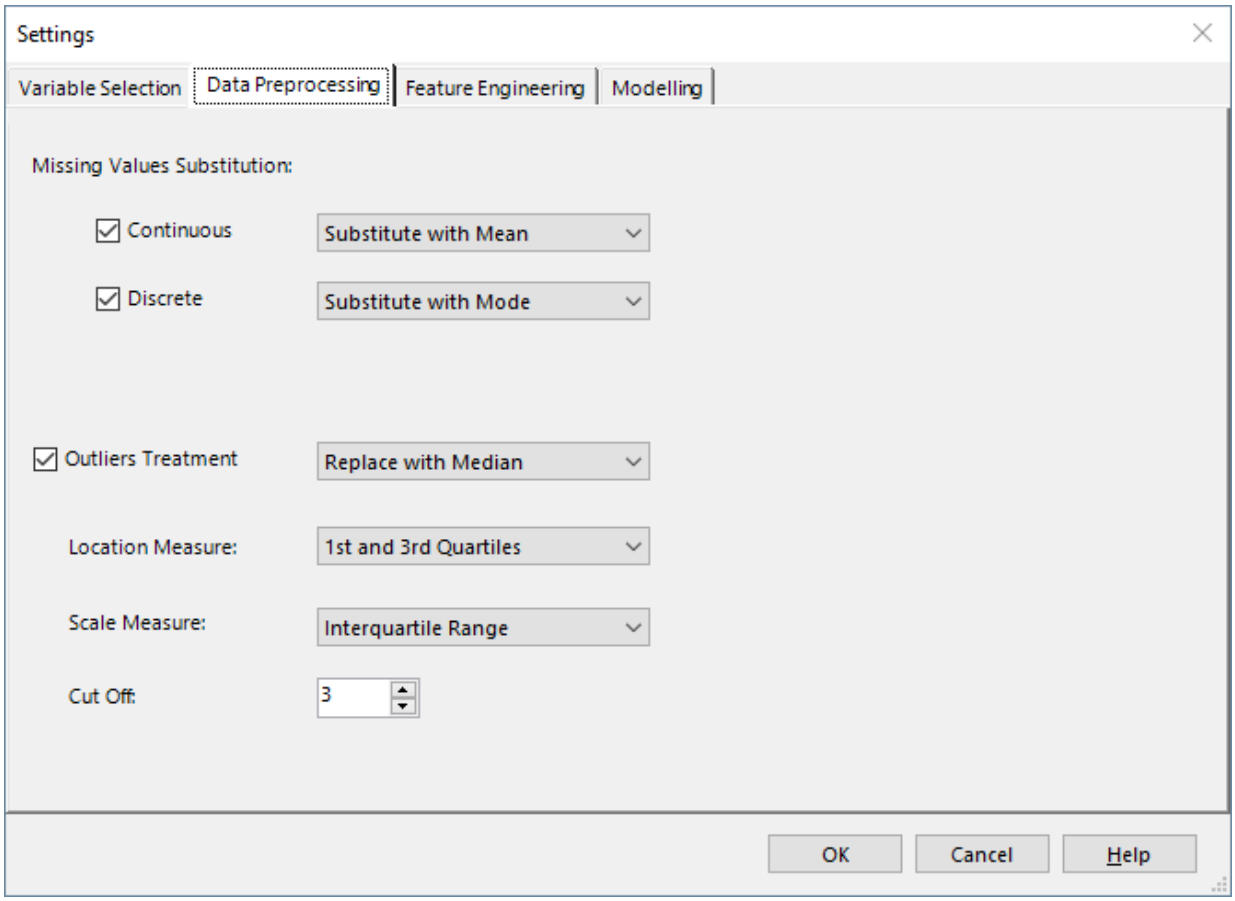
1. Select the replacement options for the outlier values:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- *Replace with Median* (default)
- *Replace with Mean*
- *Remove Record* (any record with an outlier value will be removed)

2. Define an outlier. A record will be flagged as an outlier if the value in any of the selected variables is greater than N scale measures away from the centre of the distribution (location measure), where N is a user-defined cut-off value.

- Select the *Location Measure.* The options are: *Mean*, *Median*, *1st and 3rd Quartiles*. The default is *1st and 3rd Quartiles*
- Select the *Scale measure.* The options are: Interquartile Range, Mean Absolute Deviation, and Standard Deviation. The default is *Interquartile Range*.
- Specify the cut-off value. The default is 3.

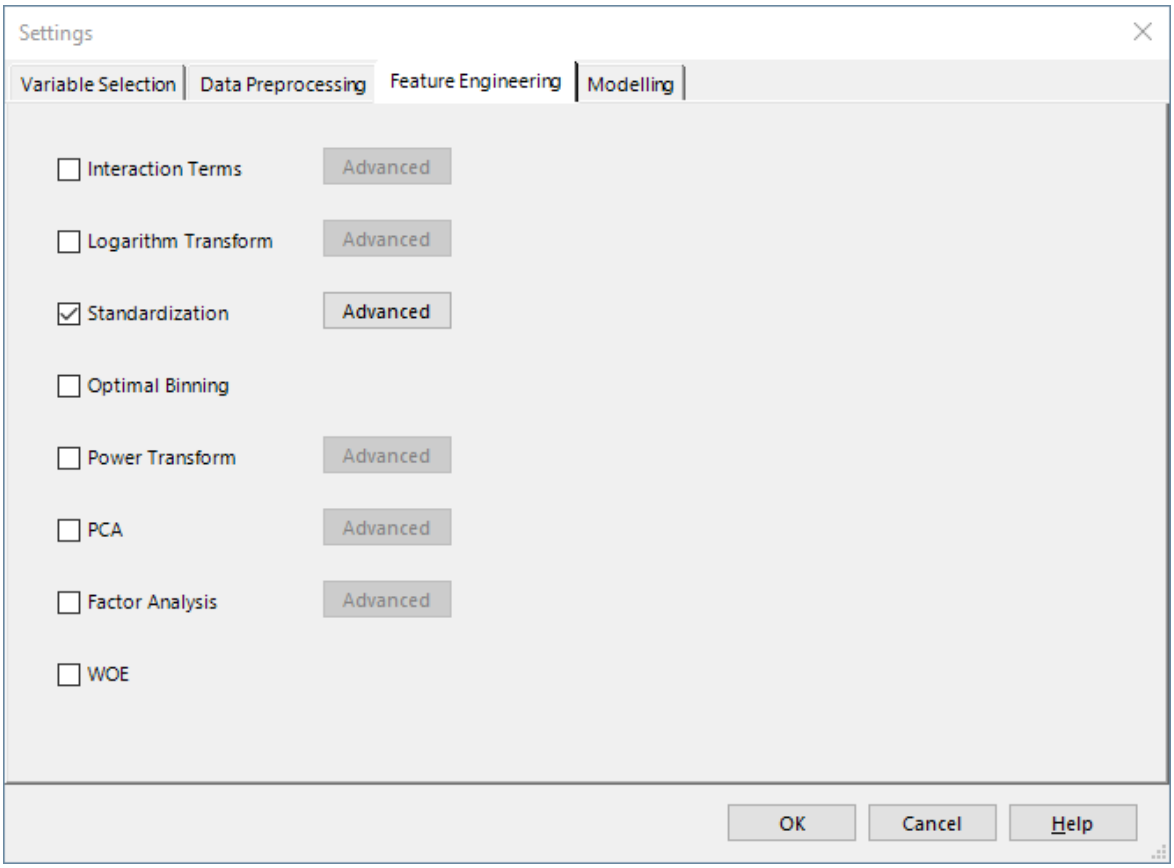


3. Feature Engineering

The **Feature Engineering** tab defines variable transformations that apply to all variables selected for AutoML and go between the Data Preprocessing and Modelling stages of the AutoML process. They are applied to the training dataset only (after the partition) and evaluated on the validation dataset.

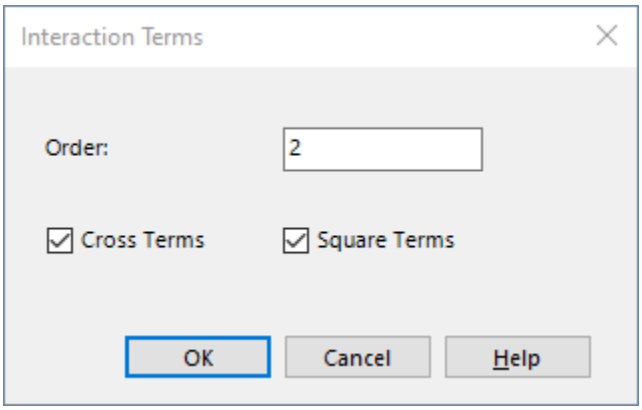
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Select the desired transformation types. The available types are: *Interaction Terms*, *Logarithm Transform*, *Standardization*, *Optimal Binning*, *Power Transform*, *Principal Component Analysis (PCA)*, *Factor Analysis*, and *WOE Transformation*. None of the transformations are selected by default.



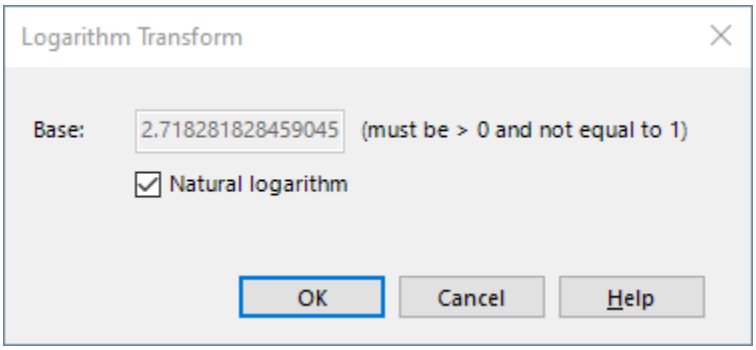
Advanced settings are available for all transformations except Optimal Binning and WOE. Click **Advanced** for any selected transformation to specify its parameters.

- For Interaction Terms, specify the order and type (Cross Terms and / or Square Terms). See the *Interaction Terms* topic for details.

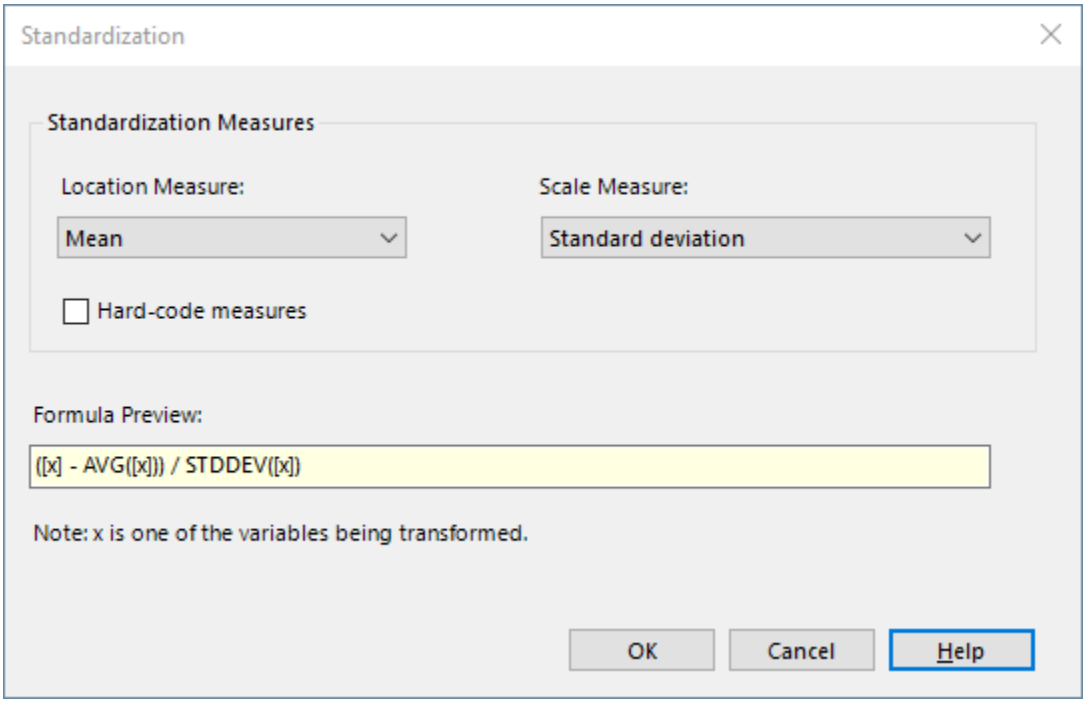


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- For Logarithm Transform, select the natural logarithm or specify the base of the logarithm. The base must be a positive number that is not equal to 1. Natural logarithm is based on the constant e (= approx. 2.718281828459045).

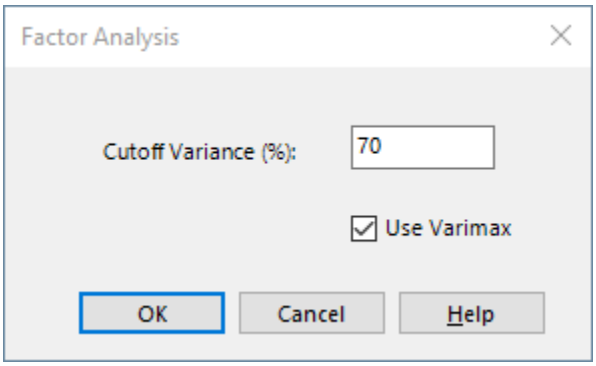


- For Standardization, select the desired Location Measure and Scale Measure to generate the standardization formula. The Formula Preview field shows the corresponding expression to be applied to all variables. The formula is (x - selected location measure) / (selected scale measure), where x stands for any of the variables selected for AutoML. For example, if you selected Mean and Standard deviation, then the formula is (x - AVG(x))/STDDEV(x). See the Standardization topic for details.



- The Optimal Binning and WOE transformations are applied with the default settings. See the topics *Helper: Optimal Binning* and *WOE Transformation* for the detailed description.
- For Power Transform, specify the exponent. See the topic Helper: Power Transform for details.
- For PCA. specify the Cutoff Variance (%). See the Principal Component Analysis topic for details.
- For Factor Analysis, specify the Cutoff Variance (%) and Varimax rotation option. See the Factor Analysis topic for details.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

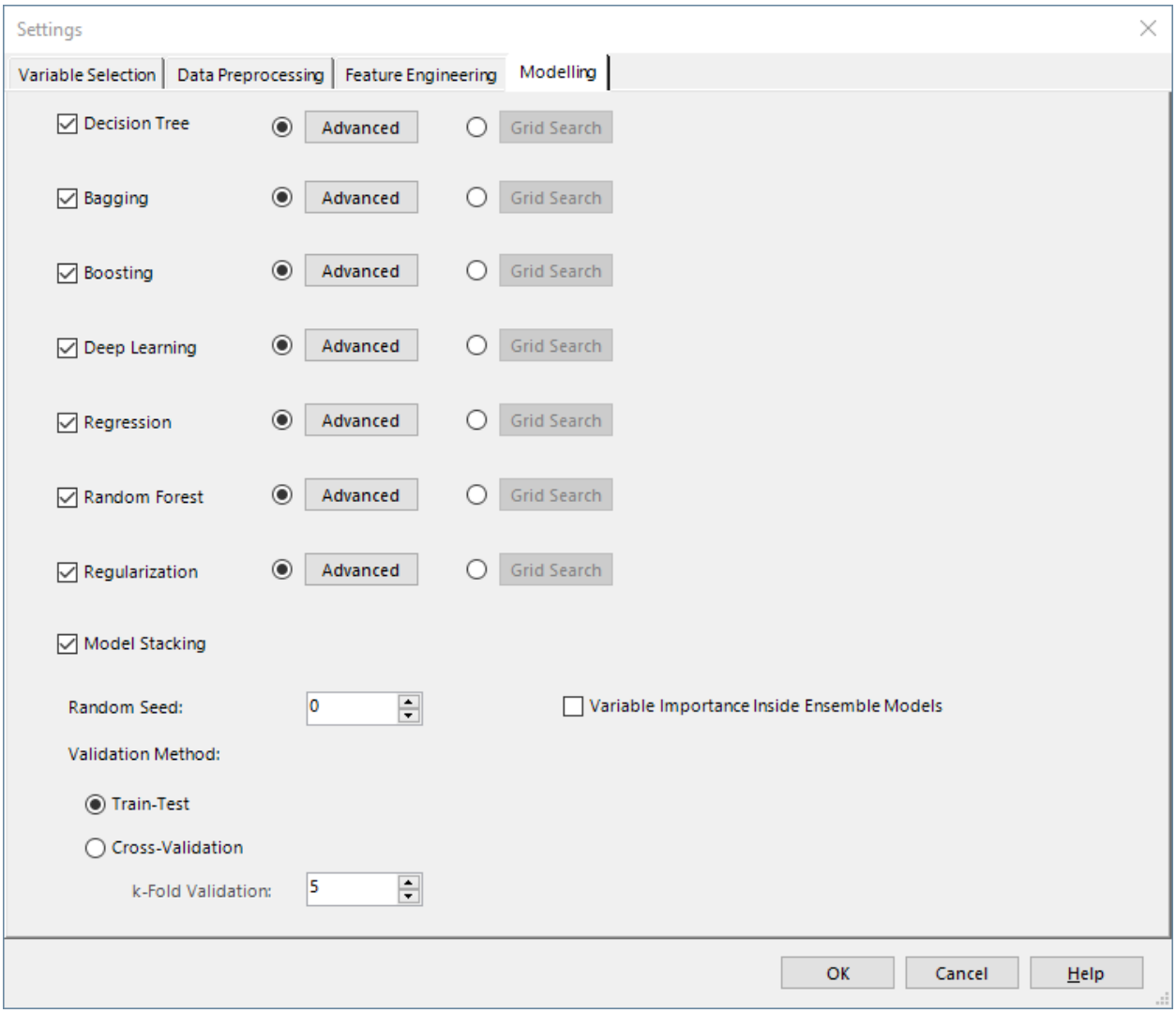


4. Modeling

The **Modeling** tab allows you to select which models will be built at the modeling stage of the AutoML process.

- Select the desired model types. The available types are: *Decision Tree*, *Bagging*, *Boosting*, *Deep Learning*, *Regression* (logistic or linear, depending on the dependent variable type), *Random Forest*, *Regularization*, and *Model Stacking*. At least one model must be selected.
- For each selected model type, click **Advanced** and specify the model parameters. The model wizards launched by the Advanced button are the same as those for the corresponding model nodes, except that they don't have the Run button. Click **Save** in each wizard after specifying the parameters.
    - Optionally, select **Grid Search** to specify a set of hyperparameters in a grid allows specifying all possible combinations of model settings for hyperparameter tuning. See an example below.
- The option **Variable Importance Inside Ensemble Models** is applicable if you selected *Bagging*, *Boosting*, or *Random Forest* model. Select it if necessary. See the Help topics for the respective models for the details on Variable Importance calculation and charts displayed in the model report.
- Specify the *Random Seed* value for repeatable results if necessary.
- Choose the Validation Method for model evaluation and comparison:
    - *Train-Test*: regular validation using a simple Train/Test partition
    - *Cross-Validation:* Select this option for *k-Fold Cross-Validation* and specify the value of the parameter *k* (the number of subsamples of the input dataset). The default is 5. This function is the same as that of the Cross Validation node in the Evaluate palette. Data preparation steps are performed within the cross-validation loop. Note: Modeling with Cross-Validation is not applicable to ensemble models (Bagging, Boosting, Random Forest) and Model Stacking. These model types will be automatically deselected if Cross-Validation is enabled.

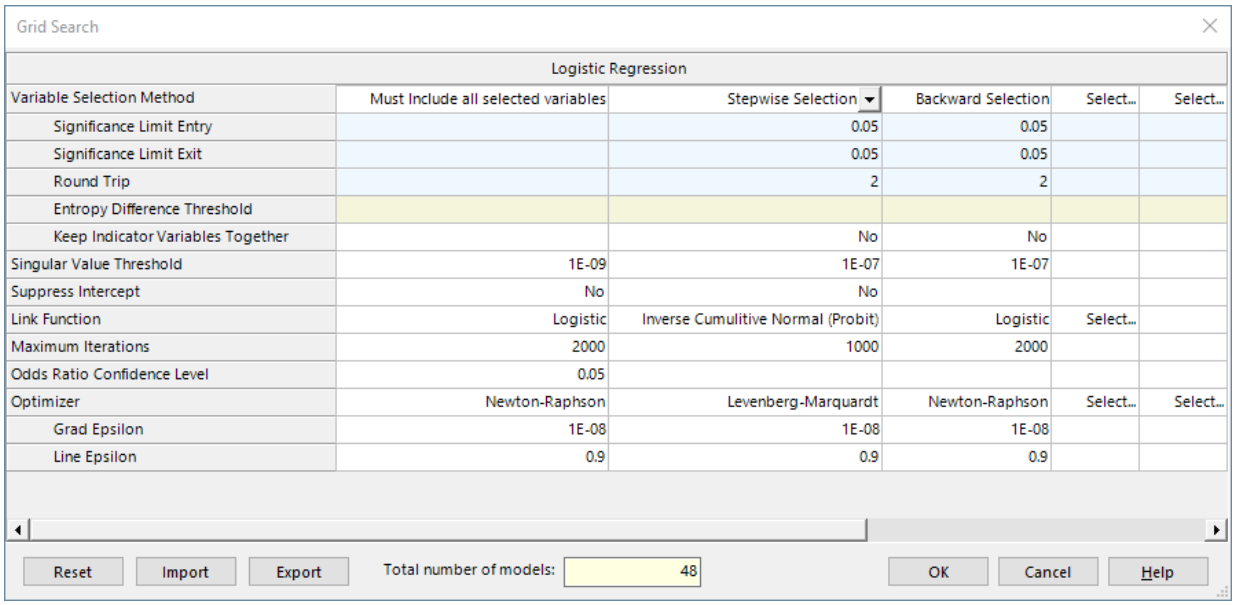
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



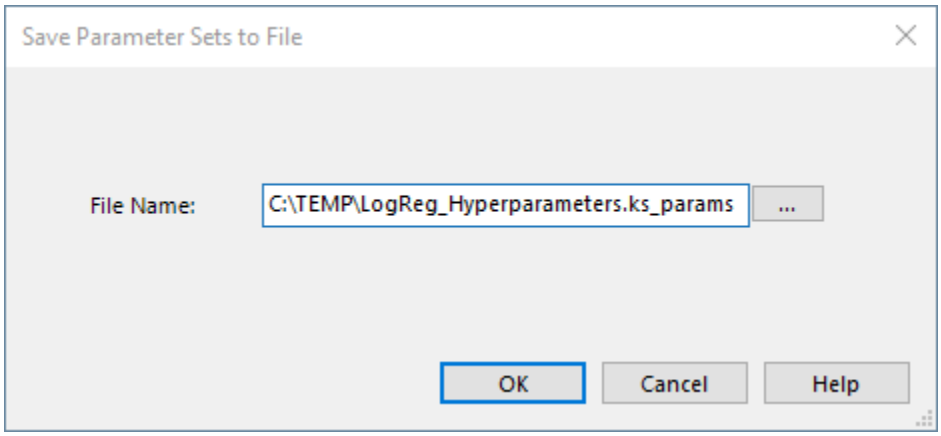
If you select the **Grid Search** option for any model, the Grid Search window opens for the selected model type. It allows you to specify a set of model parameters in a grid for hyperparameter tuning. All possible combinations of settings can be entered in the grid. The parameter categories are listed in the leftmost column. Specify the desired parameters in each column. The AutoML node will use the **grid search** method for hyperparameter tuning - an exhaustive search through the specified subset of the hyperparameter space of the models.

Note that using the **Grid Search** option may significantly increase the time required for model building within AutoML. The total number of models defined by the grid data is indicated at the bottom of the Grid Search window. In the example below, the specified subset of the hyperparameter space for Logistic Regression defines 48 models.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



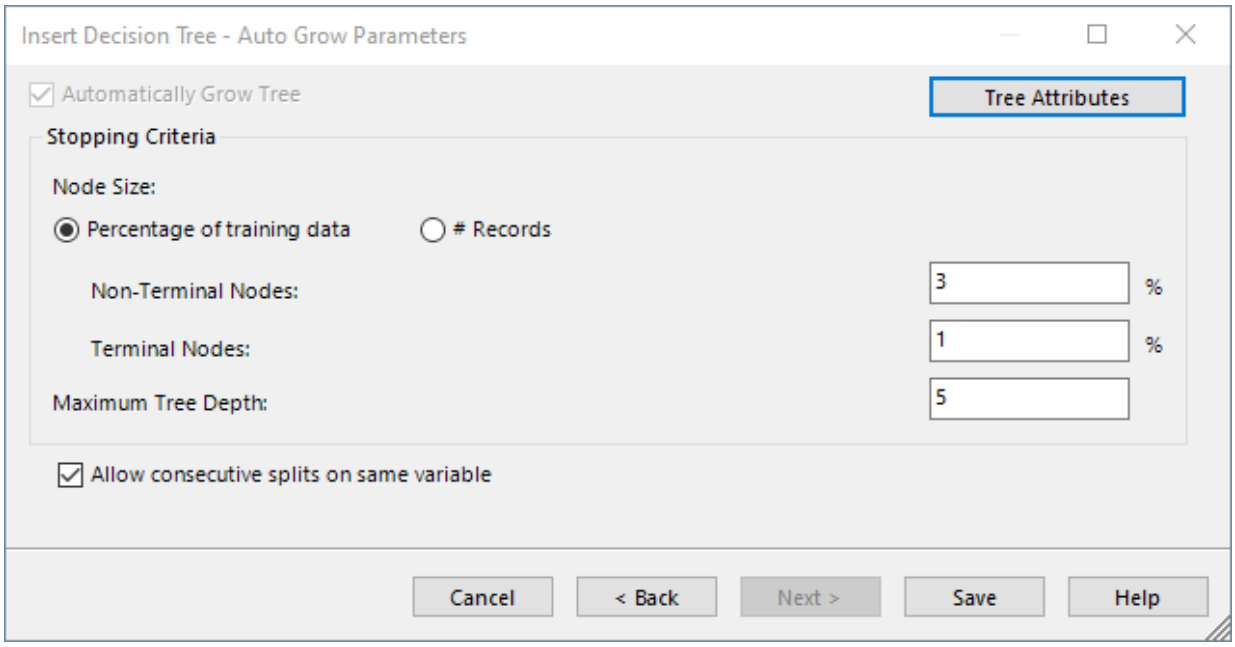
After defining the model hyperparameters in the Grid Search window, you can save the specified configuration in a file for reuse. Click the Export button and specify the name of the destination file. This will be saved in a re-importable XML file with extension ***.ks_params.**



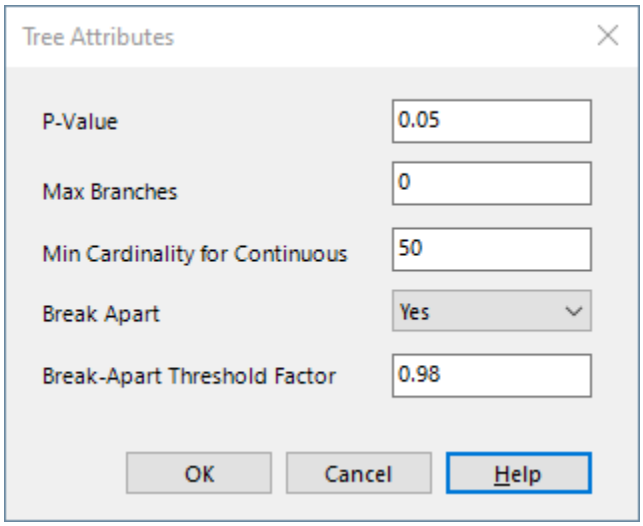
To import a previously saved set of hyperparameters, click Import and browse for the desired *.ks_params file.

For all model wizards launched by the Advanced button, the Dependent Variable is fixed (it is the one specified by the user in the main AutoML dialog). The default Tree Attributes are taken from Preferences. Here is the last page of the Decision Tree wizard in AutoML:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Clicking Tree Attributes button launches a modified Attribute Editor dialog, where you can only set a subset of attributes available in a regular Decision Tree. These attributes apply to all variables and cannot be set at the individual variable level like in a regular Decision Tree. See Tree Attributes for AutoML.



When finished specifying all settings, click **OK** in the Settings dialog.

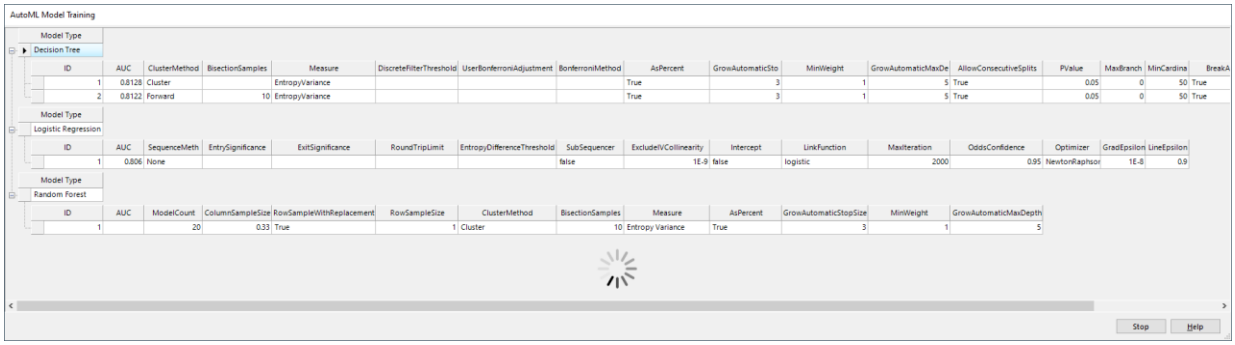
If you would like to clear all previously defined settings and reset the wizard to its default state, click the **Reset** button in the main AutoML dialog.

Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.

Since the AutoML process may take a long time, especially if the Grid Search option is used, at the model training stage the AutoML node provides a training progress window to display the list of

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

models being built, their training parameters, and current training status. You can stop the process at any time by clicking the **Stop** button in this window. In the example below, the AutoML Training Window shows four models being built.



5. AutoML Workflow and Results

As a result, the AutoML node produces a workflow with two groups of nodes (*Supernodes*). One supernode is called **AutoML - DATAPREP**, the other is **AutoML - MODELLING**. The best model of all models trained within the AutoML workflow is shown as the output of the is **AutoML - MODELLING** supernode. By default, the best model is selected using the **AUC criterion** in the Model Selector node. You can open this node and run it with a different criterion, if necessary. Double-click on the output model to view its training results. You can also view the model comparison results by opening the **Model Analyzer** node.

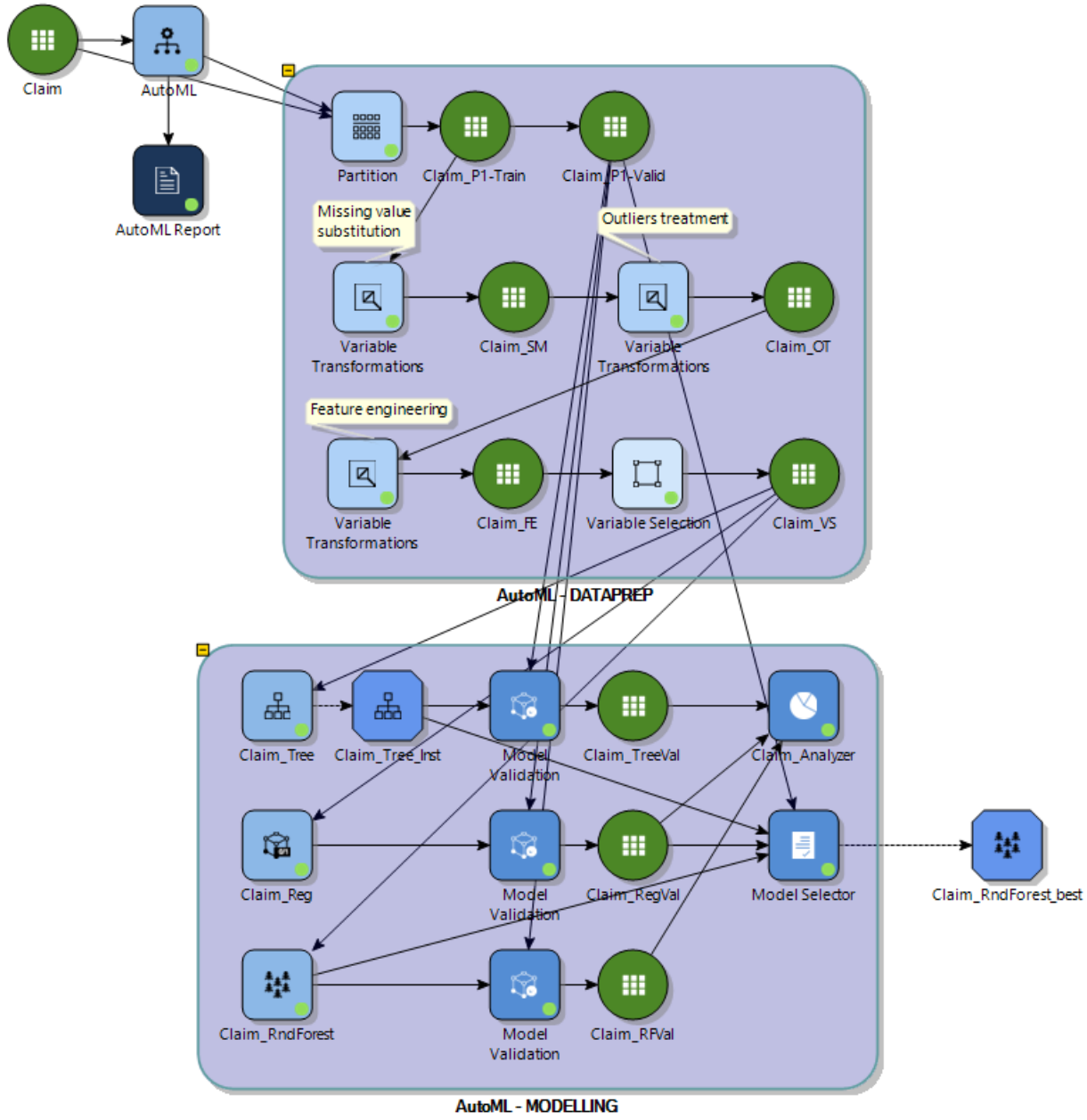
In the **AutoML - DATAPREP** group, the first step is **partitioning** of the input dataset into two subsets - **60%** for training and **40%** for validation - unless the Deep Learning model is included in the Modeling tab. If the Deep Learning model is included, the input dataset is partitioned into three subsets: **60%** for training, **30%** for validation, and **10%** for testing the Deep Learning model to guard it against overfit. The latter is indicated with the "*-Fit*" suffix in its name.

All data preparation steps are applied after partitioning: Data Preprocessing, Feature Engineering, and Variable Selection are applied to the Training subset; If the Deep Learning model is included, Feature Engineering is also applied to the Test subset for overfit prevention.

After the models have been built on the preprocessed and transformed training partition, they are validated on the Validation subset. The validation results are fed into the Model Analyzer. Finally, the Model Selector node chooses the best model of all by comparing their performance on the Validation subset.

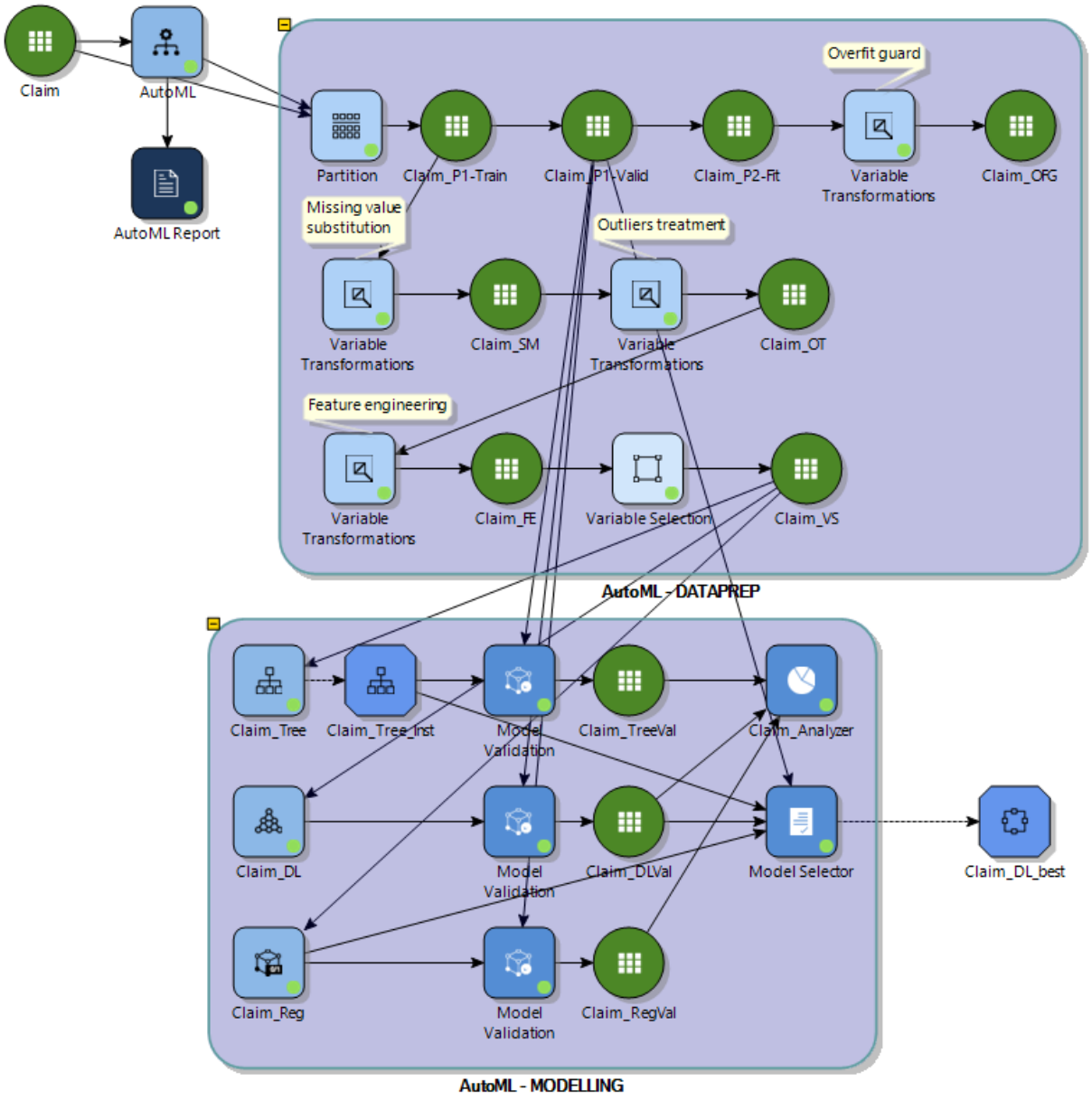
Example 1: AutoML with three model types included (Decision Tree, Regression, and Random Forest). Two-way train/validation partitioning (60% / 40%) is performed in the beginning. The best model in this case turns out to be the Random Forest model, since *Claim_RndForest_best* is the output of the Model Selector node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



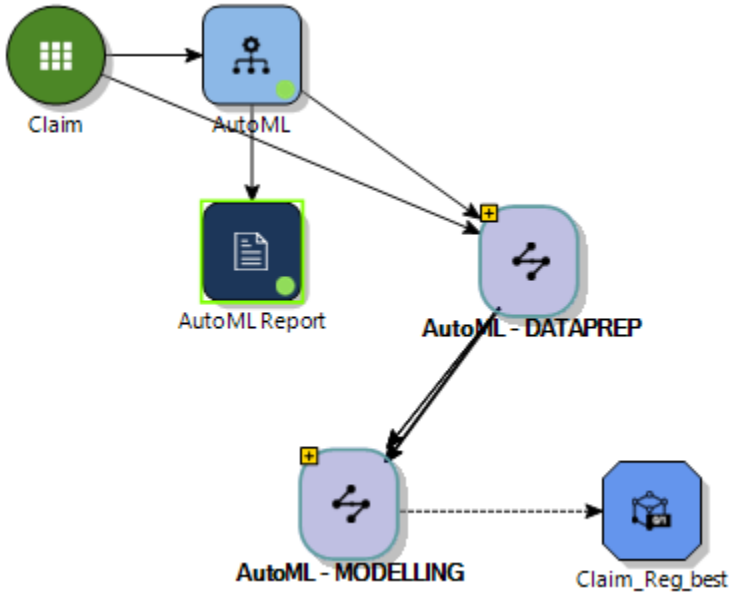
Example 2: AutoML with three model types, including Deep Learning. As noted above, in this case a three-way partitioning is performed in the beginning, *Claim_P2-Fit* being the test subset (10%) to guard the Deep Learning model against overfit.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



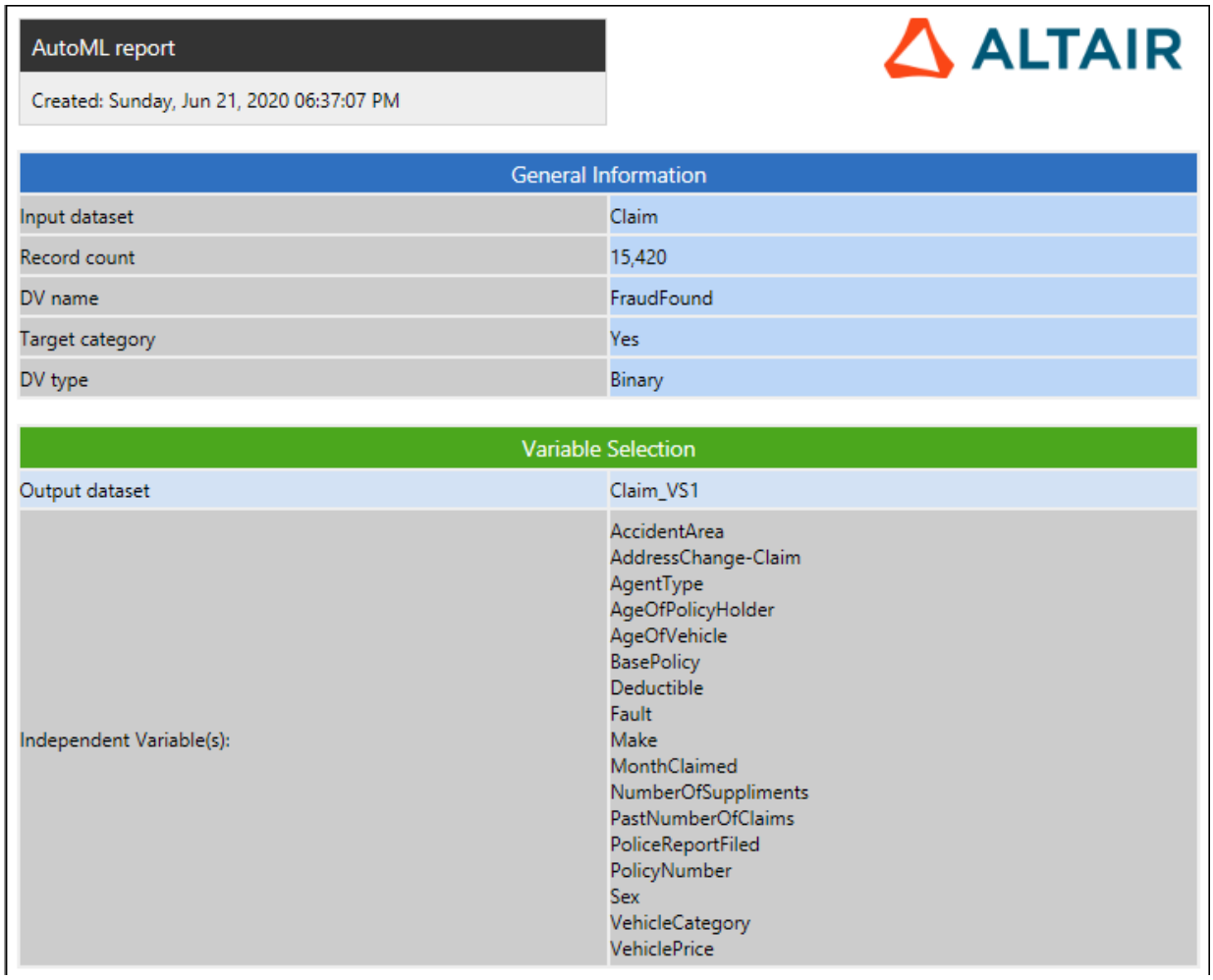
To merge (collapse) the supernode group into a single node, click on the "minimize" control at the upper-left corner of the supernode frame. For more details on handling Supernodes on the workflows, see the **Supernode** section of the Workflows topic.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The AutoML node also produces an AutoML Report in the HTML format, which contains the information on all settings defined for AutoML. Double-click on the **AutoML_Report** node to view the report. You can save it by selecting the command **Save Report As** from the **File** menu.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



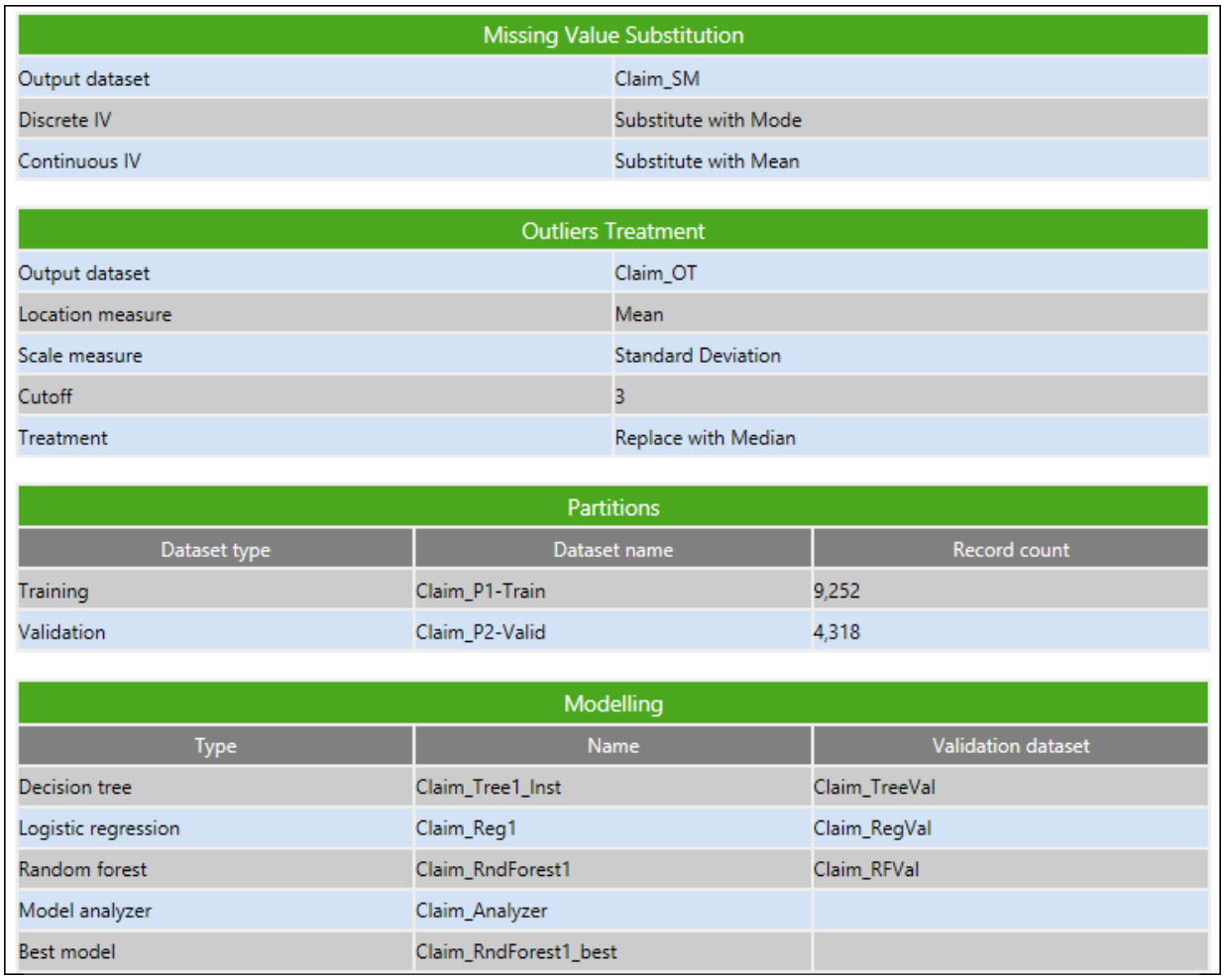
AutoML report

Created: Sunday, Jun 21, 2020 06:37:07 PM

General Information	
Input dataset	Claim
Record count	15,420
DV name	FraudFound
Target category	Yes
DV type	Binary

Variable Selection	
Output dataset	Claim_VS1
Independent Variable(s):	AccidentArea AddressChange-Claim AgentType AgeOfPolicyHolder AgeOfVehicle BasePolicy Deductible Fault Make MonthClaimed NumberOfSuppliments PastNumberOfClaims PoliceReportFiled PolicyNumber Sex VehicleCategory VehiclePrice

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Missing Value Substitution	
Output dataset	Claim_SM
Discrete IV	Substitute with Mode
Continuous IV	Substitute with Mean

Outliers Treatment	
Output dataset	Claim_OT
Location measure	Mean
Scale measure	Standard Deviation
Cutoff	3
Treatment	Replace with Median

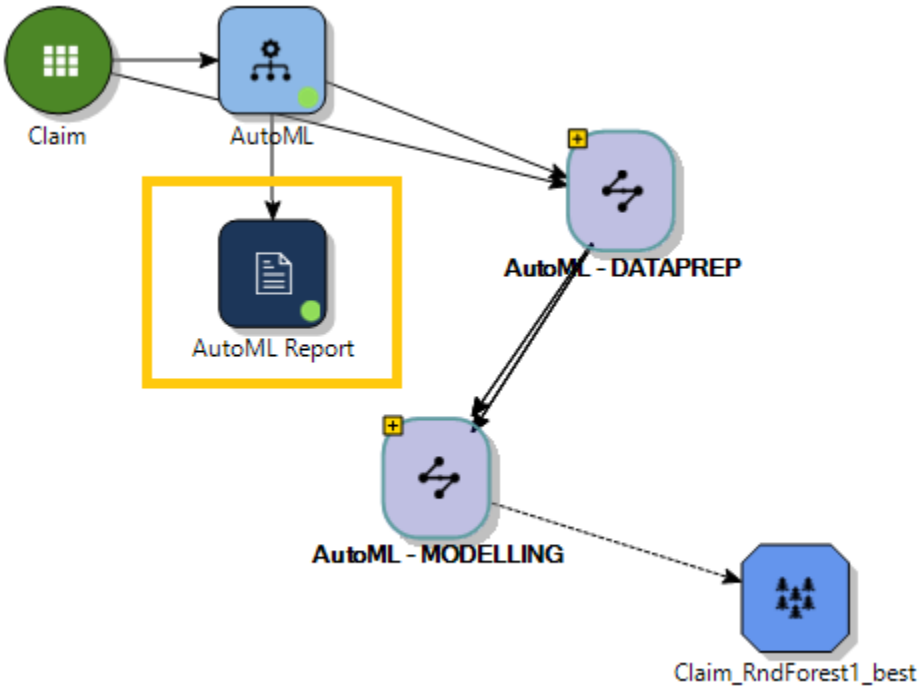
Partitions		
Dataset type	Dataset name	Record count
Training	Claim_P1-Train	9,252
Validation	Claim_P2-Valid	4,318

Modelling		
Type	Name	Validation dataset
Decision tree	Claim_Tree1_Inst	Claim_TreeVal
Logistic regression	Claim_Reg1	Claim_RegVal
Random forest	Claim_RndForest1	Claim_RFVal
Model analyzer	Claim_Analyzer	
Best model	Claim_RndForest1_best	

The **AutoML Report** node is a report in the HTML format that is a part of the AutoML node output. It lists the settings specified in the AutoML node wizard and summarizes the results. See the topic AutoML (Automated Machine Learning) for details.

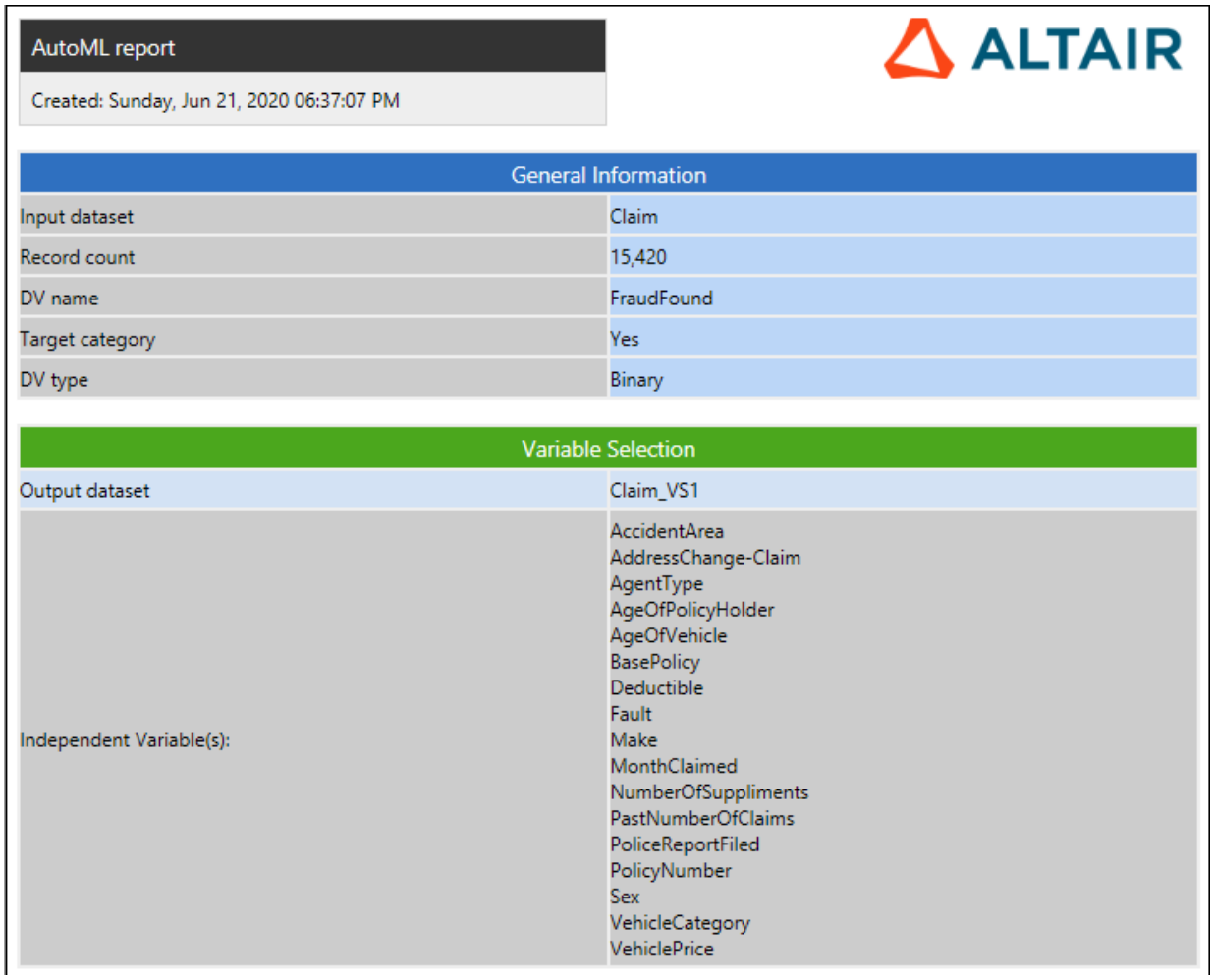
After running the AutoML node, double-click the **AutoML Report** node to view the report.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



For each stage of the AutoML-defined workflow, the report summarizes the settings: Selected Dependent Variable and its target category, selected variables, data preprocessing settings (missing value substitution and outliers treatment), training/validation partition info, the list of models built, and the name of the best model selected as a result.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



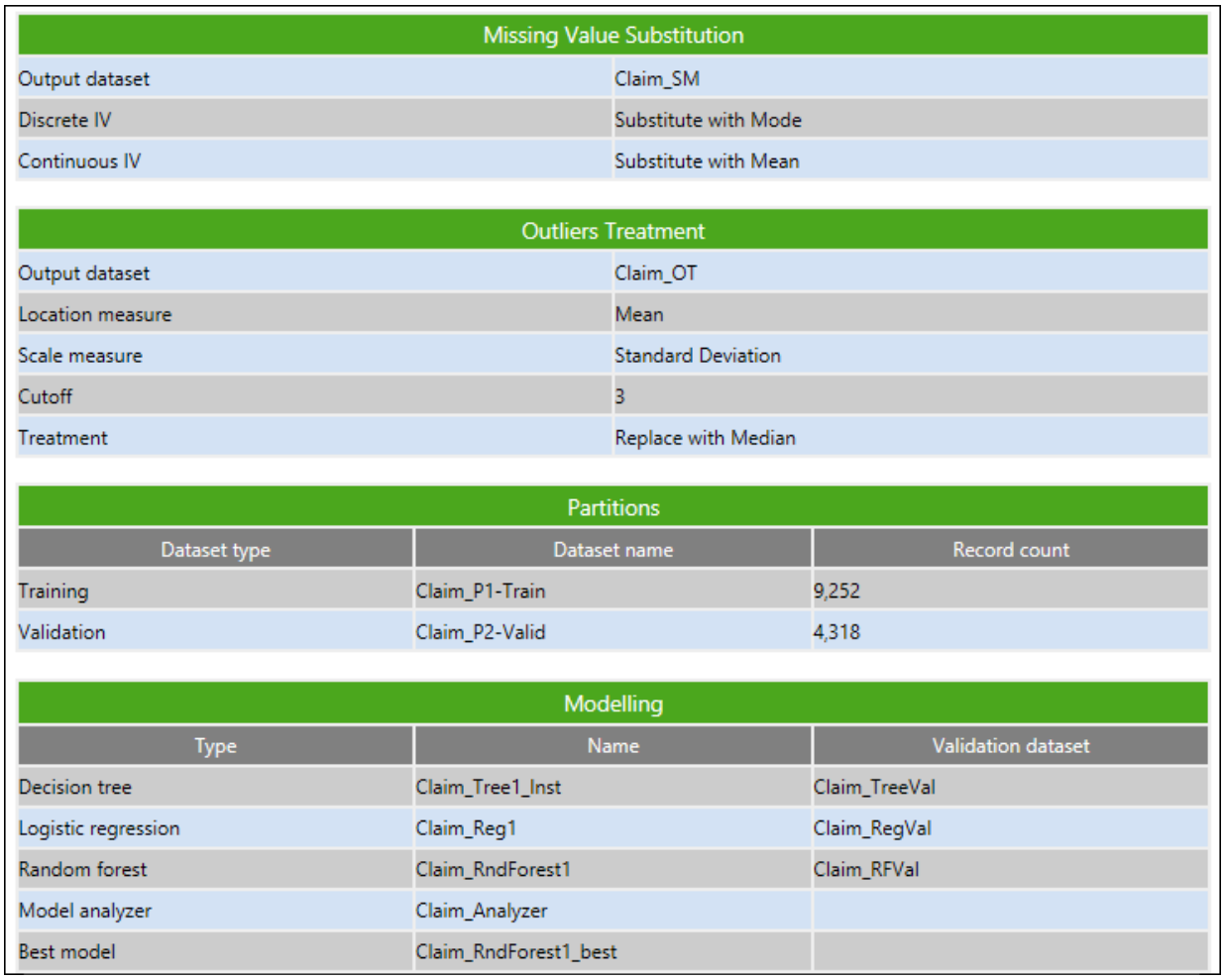
AutoML report

Created: Sunday, Jun 21, 2020 06:37:07 PM

General Information	
Input dataset	Claim
Record count	15,420
DV name	FraudFound
Target category	Yes
DV type	Binary

Variable Selection	
Output dataset	Claim_VS1
Independent Variable(s):	AccidentArea AddressChange-Claim AgentType AgeOfPolicyHolder AgeOfVehicle BasePolicy Deductible Fault Make MonthClaimed NumberOfSuppliments PastNumberOfClaims PoliceReportFiled PolicyNumber Sex VehicleCategory VehiclePrice

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Missing Value Substitution	
Output dataset	Claim_SM
Discrete IV	Substitute with Mode
Continuous IV	Substitute with Mean

Outliers Treatment	
Output dataset	Claim_OT
Location measure	Mean
Scale measure	Standard Deviation
Cutoff	3
Treatment	Replace with Median

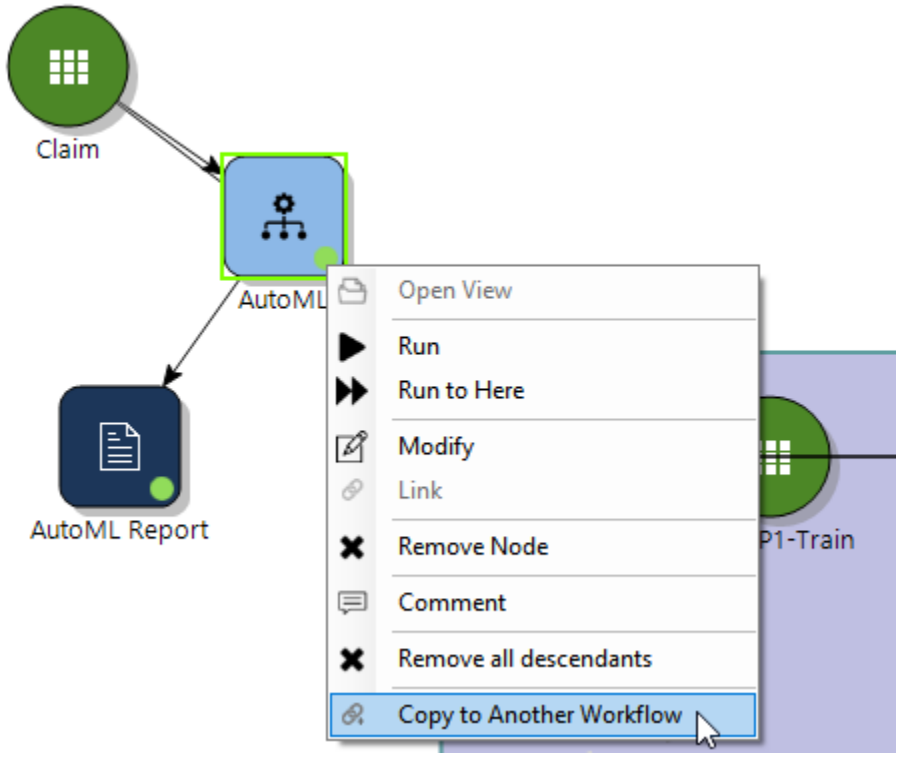
Partitions		
Dataset type	Dataset name	Record count
Training	Claim_P1-Train	9,252
Validation	Claim_P2-Valid	4,318

Modelling		
Type	Name	Validation dataset
Decision tree	Claim_Tree1_Inst	Claim_TreeVal
Logistic regression	Claim_Reg1	Claim_RegVal
Random forest	Claim_RndForest1	Claim_RFVal
Model analyzer	Claim_Analyzer	
Best model	Claim_RndForest1_best	

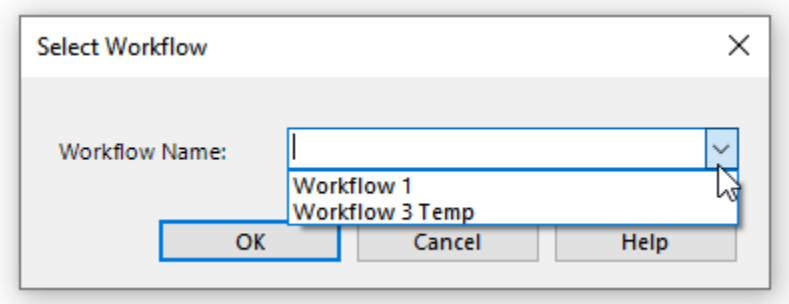
The context menu command **Copy AutoML to Another Workflow** provides an easy way to copy an AutoML node to another workflow while inheriting all its settings. The copied node is a new AutoML node - it produces new objects, without overwriting the original outputs. All the settings of the new node are inherited from the source AutoML node, the same way the settings of *Template models* are inherited when connecting a model node to a new one. You can modify the node and modify its settings before running it.

To copy an AutoML node, right-click on it and select the command **Copy to Another Workflow** from the context menu, as shown below.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The **Select Workflow** dialog opens. Select the desired destination workflow from the drop-down list or specify the name of a new workflow tab to create.



Click **OK**. The destination workflow becomes active, with the new (copied) AutoML node displayed. The input dataset for the source AutoML node is also linked on the destination workflow tab and is connected to the new AutoML node.

You can modify the new new AutoML node and run it when finished modifying the settings, or run it as is. The new node creates objects with new unique names rather than overwriting the objects created by running the source AutoML on the original workflow.

See also: Workflows

Insert Model from XML/PMML

The node **Model From XML/PMML** available in the Connect palette allows you to:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- import models from other projects
- share and exchange models created in Knowledge Seeker and Knowledge Studio
- import PMML models generated in other analytics environments, such as *R*.

In the case of PMML, note that not all statistics of the imported model can be displayed in its Results tab. The model's Sequence Statistic view in the Results tab will only display the model coefficients.

All native models, including trees, are stored as XML files in the file system. The model files normally have the *kdm* extension. For example, if your project contains a model called Response_DecisionTree, it is represented by the file *Response_DecisionTree.kdm* in the project folder. You can view the contents of the project folder by right-clicking the project in the Project Outline and selecting Open in Explorer from the context menu.

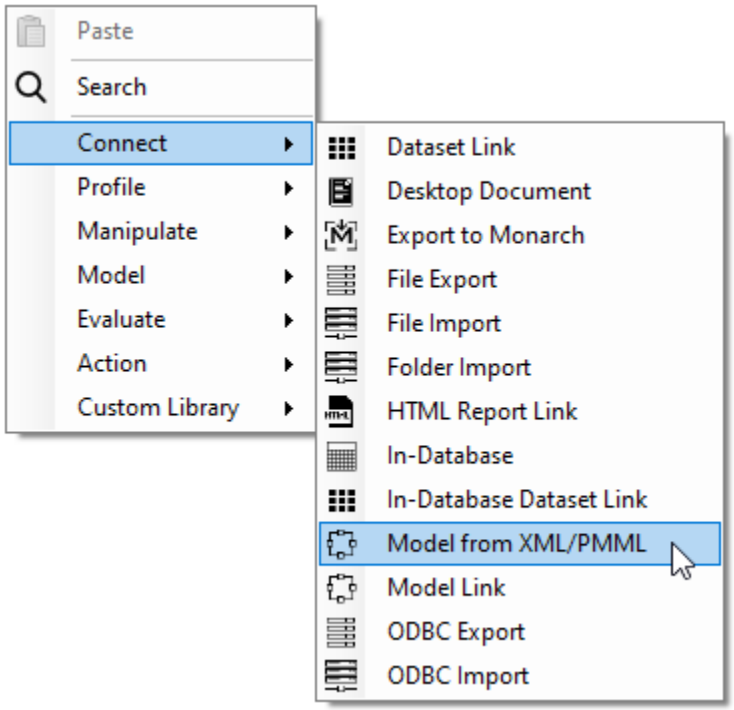
Any native model can be exported as an XML or PMML file by generating XML code for it and then saving the code in a file. This is done using the **Insert | Code Generation** command with the XML or PMML option. See the topic About Code Generation for details.

To import a model from a XML or PMML file, use one of the following ways:

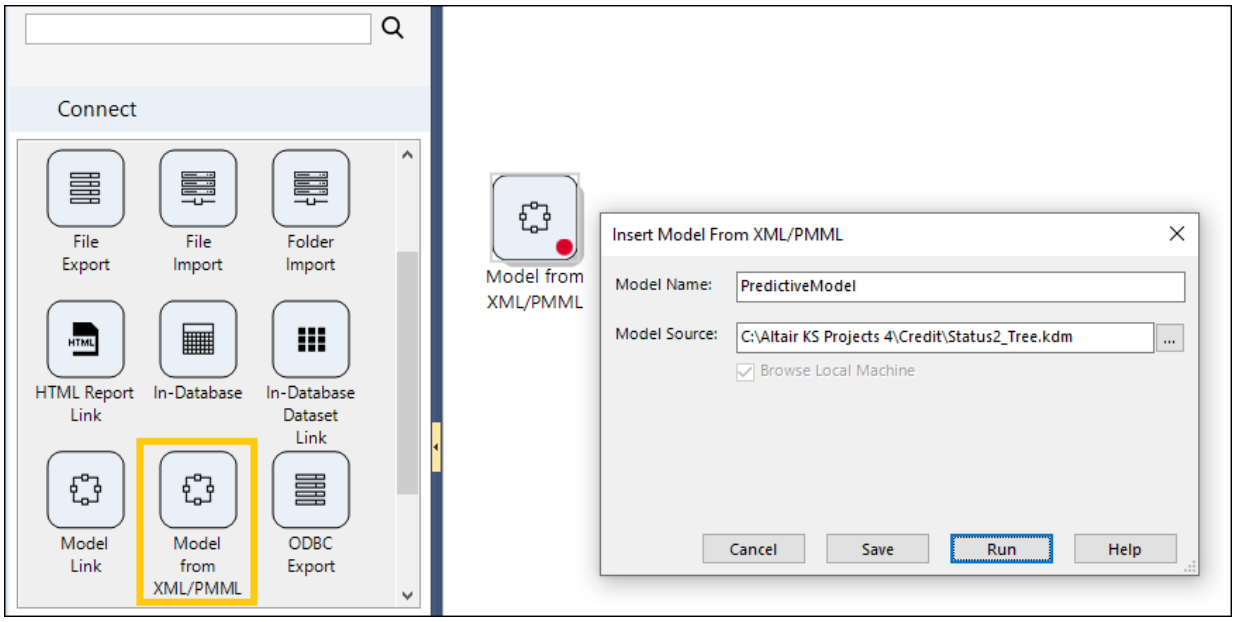
- Right-click on the Workflow canvas and select **Connect --> Model From XML/PMML** from the context menu.

OR

- Open the **Connect** palette and drag the **Model From XML/PMML** node onto the workflow canvas, then double-click on it.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Specify the name of the new model as it will appear in the workflow. Use the browse button to browse for the XML, KDM, or PMML file of the model you wish to import. If you want to import a model from another project, browse the Working Directory that contains the project folder, open the folder and select the desired model.

If you are working in the client/server mode, you have an option of browsing the local machine or the KS Server host. Uncheck the **Browse Local Machine** checkbox if your source model is on the server.

See also: About Code Generation

Strategy Trees

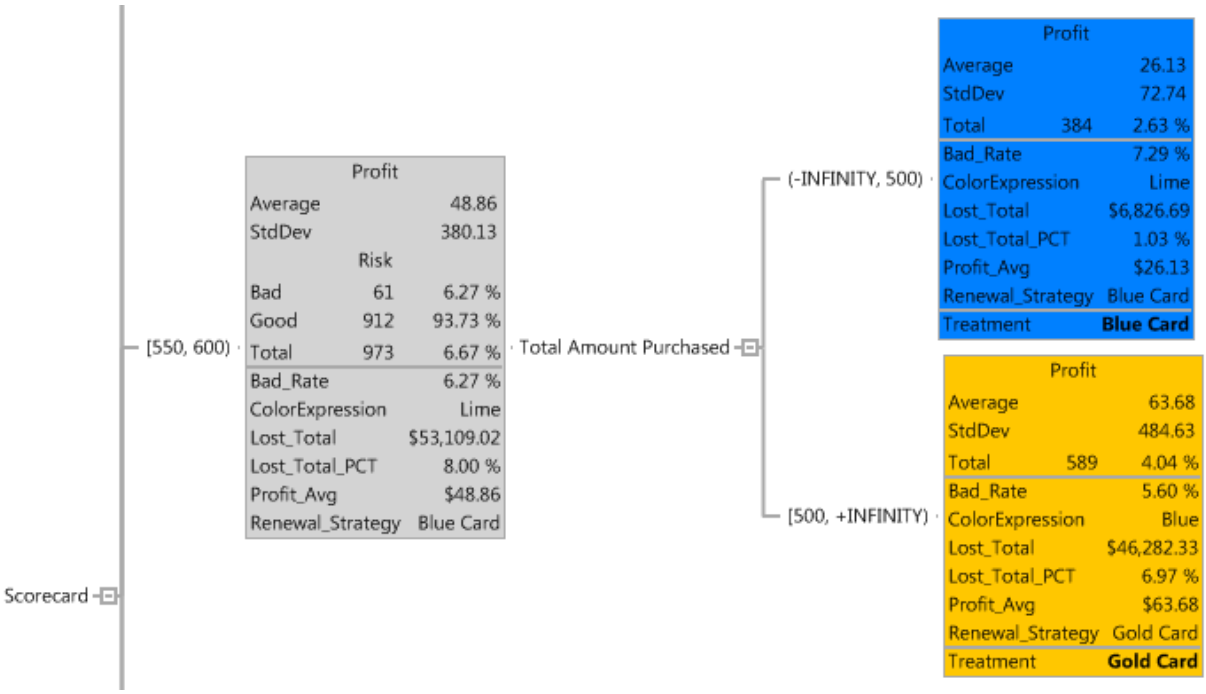
Strategy Trees: Overview

A **strategy tree** is a series of rules and actions that are developed to help achieve a business objective when interacting with a customer. A tree metaphor, very similar to decision trees, is used to define conditions and associated actions. Each segment (or node) can also display tree calculations that can compute key performance indicators such as profit, losses, purchasing behaviors or other important measures.

Figure1 shows a fragment of a strategy tree with five calculations, such as Profit_Avg = AVG([Profit]) and treatments *Gold Card*, *Blue Card*, and *Cancel Card* assigned to three terminal nodes.

Figure 1 - Example strategy tree (fragment)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Strategy Trees Compared to Other Model Types

The Strategy tree model differ from other models. To understand strategy tree models it is helpful to compare them to other model types:

- **Strategy Trees are created manually based on business rules supplied by the user.**
  By contrast, predictive models are generated automatically by applying a mathematical algorithm to historic data.

- **Strategy Trees prescribe an action.**

  By contrast, a classifying predictive model predicts the likelihood of a particular outcome. A strategy tree may use this likelihood to prescribe an action. For example, a Strategy tree may state, “if the probability of a customer responding to offer A is greater than 10%, then send then offer A”. Consider Strategy Tress to be Prescriptive Models.

- **Strategy Trees do not need dependent variable but can have dependent variables assigned locally.**
  By contrast, a decision tree has a single dependent variable that is used for each and every split. To help optimize the business rules, a dependent variable can be used when constructing Strategy Trees, but this is only to help find optimal groupings and the dependent variable can vary throughout the tree.

- **Strategy Trees can display the distribution of any variable.**

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

By contrast, a decision tree has a single dependent variable and its distribution is always displayed in each and every node. With Strategy Trees, you can display the value distribution for any variable in your dataset and apply a great variety of calculations.

Inserting a Strategy Tree

A strategy tree is a series of rules and actions developed to achieve a business objective depending on certain performance indicators.

A Strategy Tree node can be added to the workflow in one of the following ways:

- Drag a connection arrow from a dataset node to any area on the Workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Model --> Strategy Tree** as in the example screenshot below. A new **Strategy Tree** process node will be created and the wizard will open.

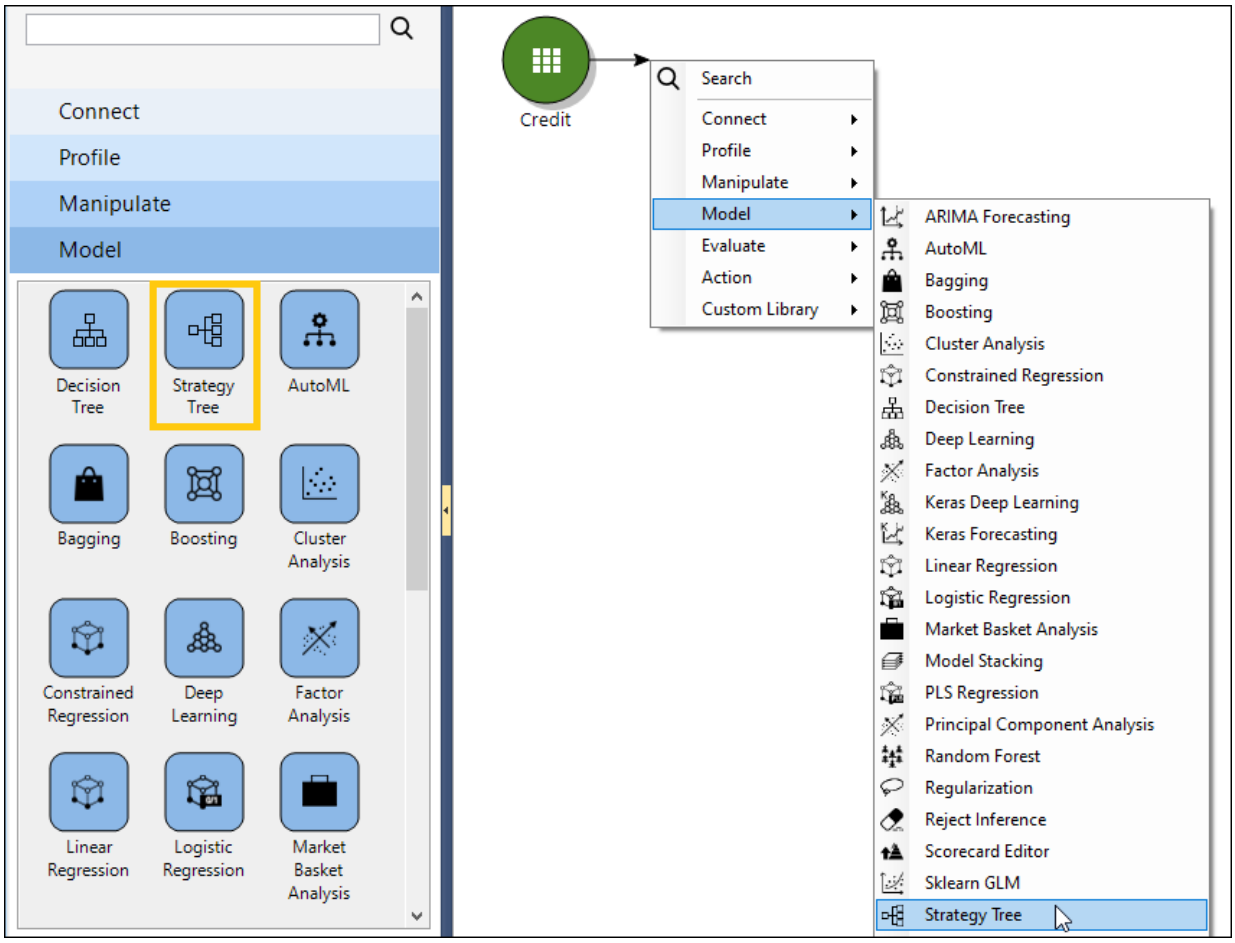
  OR

- Right-click on the Workflow canvas and select **Model --> Strategy Tree** from the context menu. Connect a dataset node to the tree node.

  OR

- Open the **Model** palette and drag the **Strategy Tree** node onto the workflow canvas. Connect a dataset node to the **Strategy Tree** node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



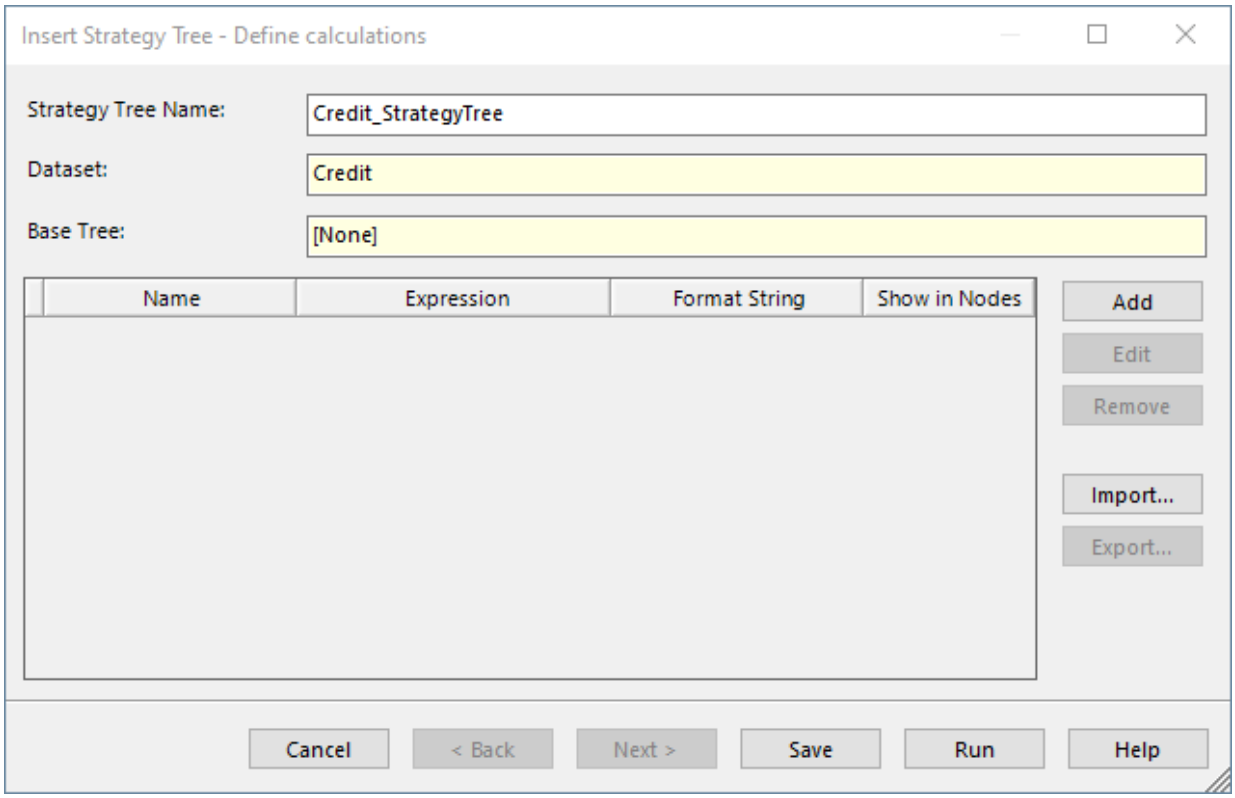
If the wizard does not open automatically, double-click on the Strategy Tree node or right-click on it and select **Modify** from the context menu.



In the Insert Strategy Tree wizard:

- Specify the name for the new tree. The name can contain spaces, but cannot contain the following characters: " * + , . / : ; < = > ? \ |
- Add calculations or import calculations if necessary. This is optional. You may add calculations later, after the tree has been inserted.
- The Base Tree name is specified only if another Strategy Tree node is connected to the new tree node. In this case the current tree has two inputs: a dataset and another tree. All applicable training parameters and attributes of the base tree will be copied and used for the new tree..

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Calculations can also be imported from another Strategy Tree or from a previously saved Expressions file.

To edit or remove a calculation, select it and click **Edit** or **Remove**. When you click Add or Edit, this opens the Expression Editor for Node Calculations.

After you have finished adding or editing calculations, click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the operation, double-click the strategy tree node to open the interactive tree view.

See also:

- About Node Calculations
- Inserting a Node Calculation

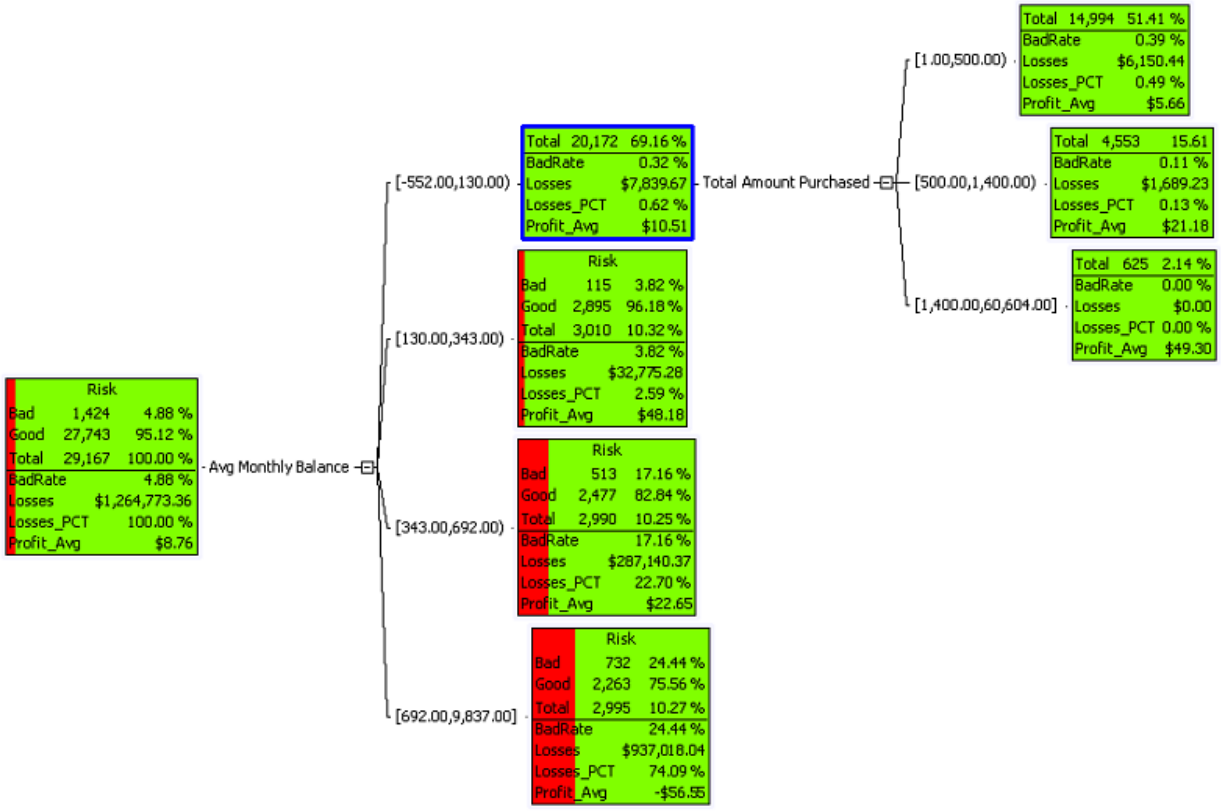
Viewing a Strategy Tree

Tree View

The **Tree** view displays the graphical representation of the tree.

Strategy Trees are oriented horizontally by default, but you can change the orientation using the **Tools | Options** command.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

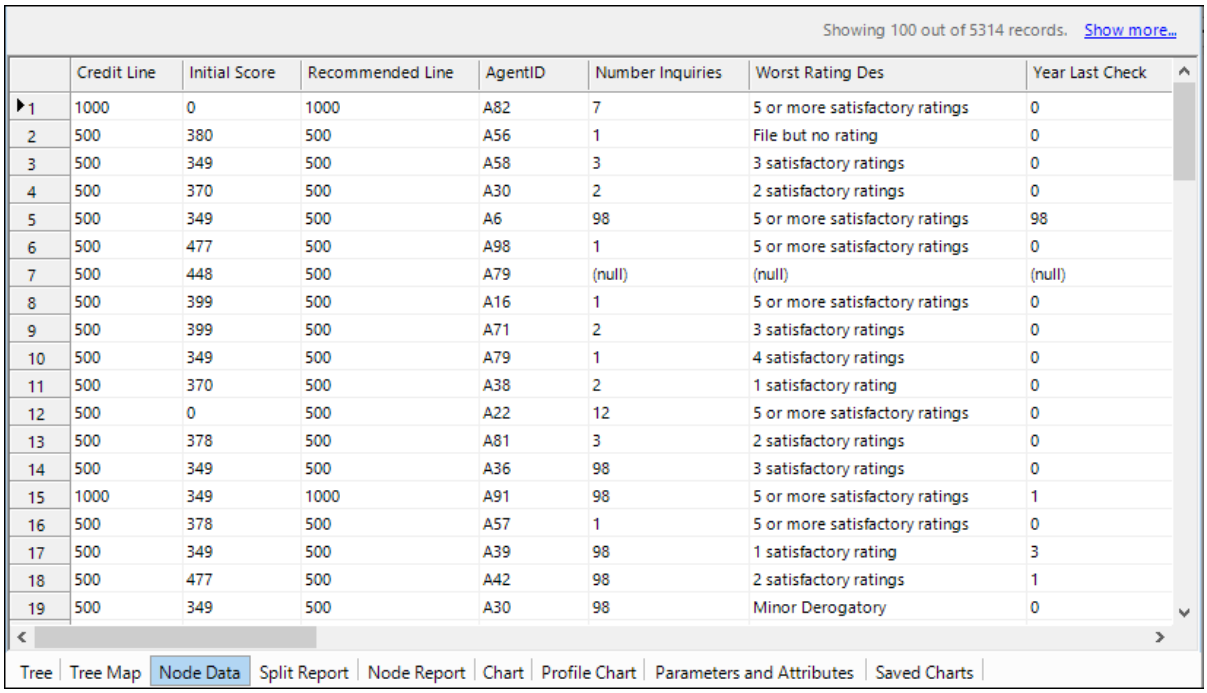


The Tree view for Strategy Trees has the same capabilities as that for Decision Trees. See the Tree topic for Decision Trees for details.

Node Data View

The **Node Data** view displays the data contained within the active tree node. The node is considered active when it is highlighted by a mouse click while in the **tree** view or **tree map** view.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Showing 100 out of 5314 records. Show more...

	Credit Line	Initial Score	Recommended Line	AgentID	Number Inquiries	Worst Rating Des	Year Last Check
1	1000	0	1000	A82	7	5 or more satisfactory ratings	0
2	500	380	500	A56	1	File but no rating	0
3	500	349	500	A58	3	3 satisfactory ratings	0
4	500	370	500	A30	2	2 satisfactory ratings	0
5	500	349	500	A6	98	5 or more satisfactory ratings	98
6	500	477	500	A98	1	5 or more satisfactory ratings	0
7	500	448	500	A79	(null)	(null)	(null)
8	500	399	500	A16	1	5 or more satisfactory ratings	0
9	500	399	500	A71	2	3 satisfactory ratings	0
10	500	349	500	A79	1	4 satisfactory ratings	0
11	500	370	500	A38	2	1 satisfactory rating	0
12	500	0	500	A22	12	5 or more satisfactory ratings	0
13	500	378	500	A81	3	2 satisfactory ratings	0
14	500	349	500	A36	98	3 satisfactory ratings	0
15	1000	349	1000	A91	98	5 or more satisfactory ratings	1
16	500	378	500	A57	1	5 or more satisfactory ratings	0
17	500	349	500	A39	98	1 satisfactory rating	3
18	500	477	500	A42	98	2 satisfactory ratings	1
19	500	349	500	A30	98	Minor Derogatory	0

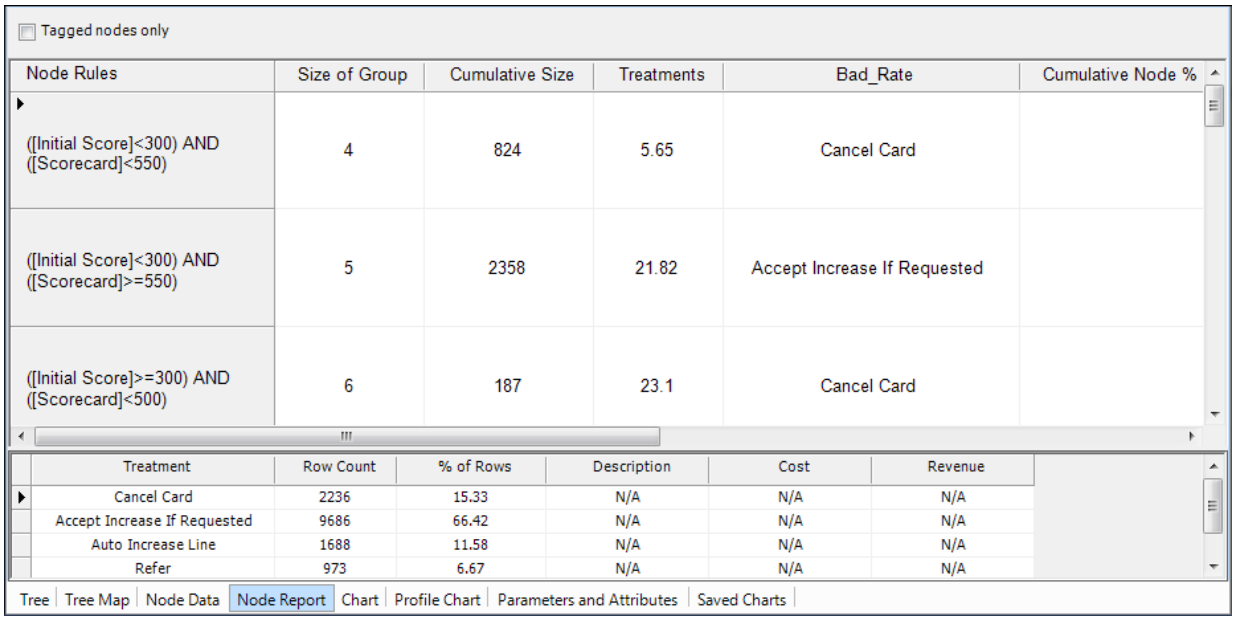
Tree | Tree Map | Node Data | Split Report | Node Report | Chart | Profile Chart | Parameters and Attributes | Saved Charts

Data is listed in a worksheet-style window where the fields appear as columns and the records appear as rows. This information cannot be changed.

Node Report

The strategy tree node report provides you with information about the tree calculations in the terminal nodes of the tree.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Node ID	The node ID. By default this is not displayed and must be turned on in the Options command.
Node Rules	The rules for the terminal node in the tree.
Size of group	The number of observations in a given terminal node.
Cumulative Size	The cumulative number of observations up to and including this node.
Treatment	The treatment that is assigned to the given node. You can highlight one or more rows and assign them treatments by right clicking or Treatments command.
Tree Calculation Values	Each of the tree calculations is displayed in the node report. The title of the column matches the title of the label when the expression was defined. You can sort the node report by clicking on calculation headers.
Cumulative Calculation Values	If you turn this option on (it is off by default), then the calculation values is cumulatively summed. This is useful for selecting cut off points and managing resources. The amounts are accumulated by the current sort order.
Node Percent	If you turn this option on (it is off by default), then node calculation is displayed a percentage of the sum node calculation for all nodes. This will only make sense in some circumstances, namely when the calculation is a SUM or COUNT.
Cumulative Node Percent	If you turn this option on (it is off by default), then node calculation is displayed a cumulative percentage of the sum node calculation for all nodes. As with Node Percent, This will only make sense in some circumstances, namely when the calculation is a SUM or COUNT.

The report can be sorted on a particular column. For example, you want to see all the nodes that have a treatment.

To sort the columns, click on the column heading.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Treatment Summary

There is a summary of the treatment nodes located at the bottom of the report. This gives you a quick report on the total number of records that have been labeled with a given treatment.

Chart View

The **chart** helps you to visualize the distributions of the independent variables within a node of a tree model

The first time the **chart** is viewed for a tree, the **Add Chart** wizard appears. You can start the **Add Chart** wizard manually:

From the menu:

Tools | Add Chart...

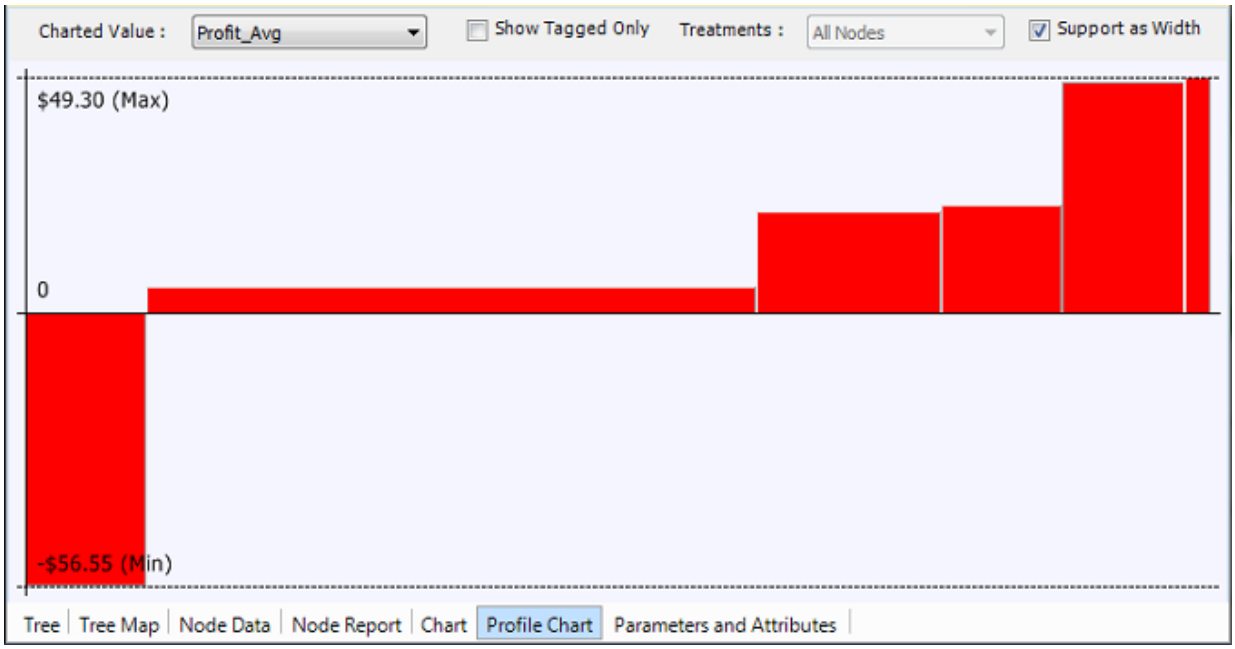
Using the Tree Chart

The tree chart provides useful information while traversing the tree. Splitting the screen and docking the dynamic chart with the tree view allows you to traverse the tree and immediately see the distributions of the specified variable displayed in the chart.

Profile Chart

The **Profile Chart** view displays each terminal node in tree. Its purpose is to display the tree in a single chart.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Before the profile chart can be displayed the tree must be grown at least one level and you have at least one tree calculation.

Clicking on a bar will cause the tree to navigate to that node. You can also right click to access commands such as assigning treatments and tagging nodes.

Charted Value	The charted value presents a list of each of the tree calculations. Choose the value that you would like charted.
Show Tagged	If this is checked, only nodes that are tagged are displayed.
Treatments by Default	By default, all terminal nodes are displayed. Changing the treatments option allows you to plot only nodes that are assigned to a specific treatment.
Support as Width	When checked, the width of the bar represents the number of observations in the node. If unchecked, each bar is the same width regardless of the number of rows.

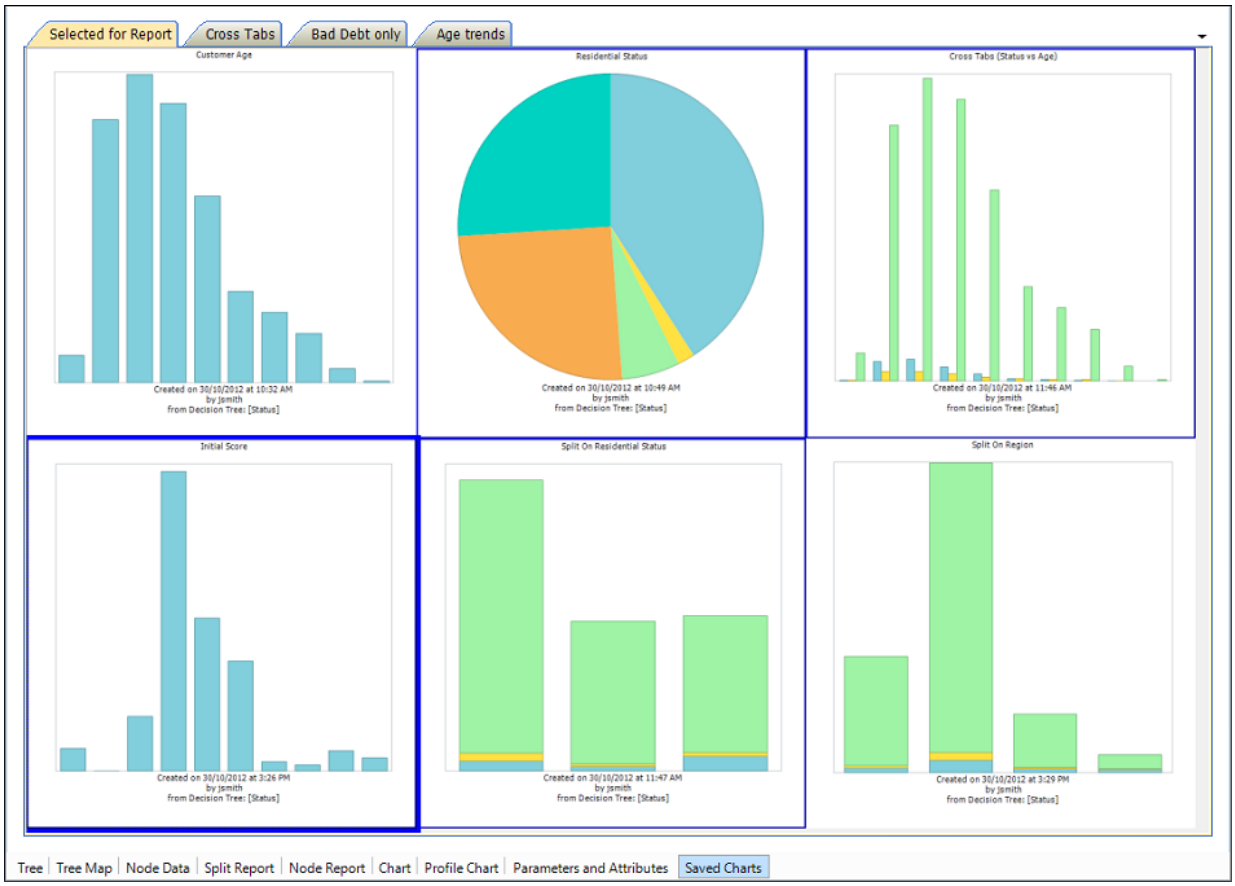
If the tree calculations are not up-to-date, then they will not be charted. You may need to use the Recalculate command for the profile chart to plot all of the terminal nodes.

The Saved Charts tab is available for datasets, decision trees, and strategy trees. This tab represents the library of charts you saved in other views of the object it belongs to.

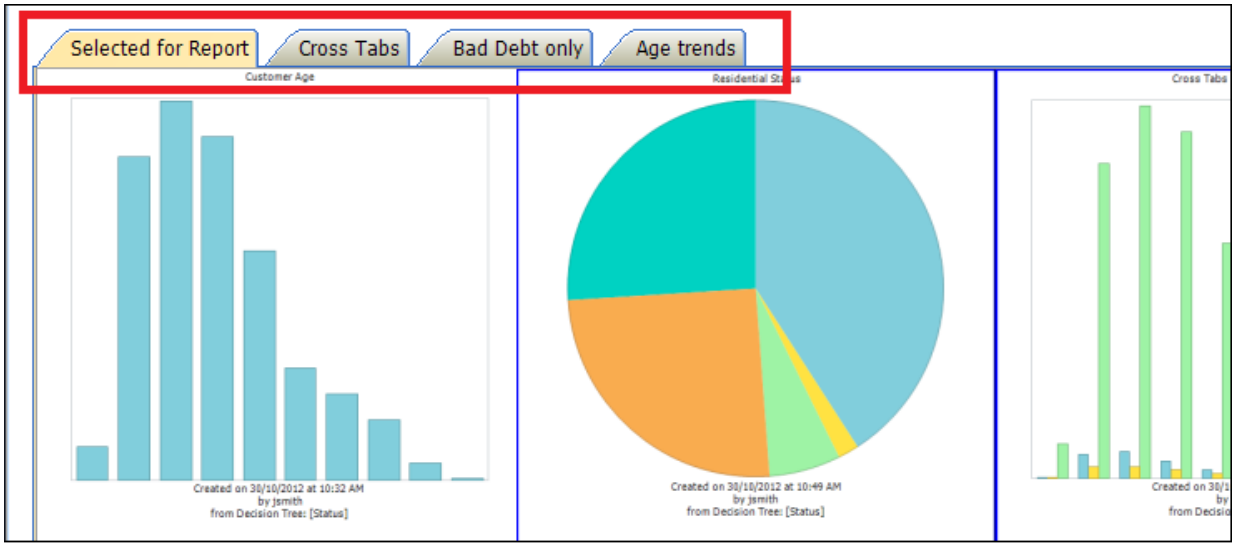
- The Saved Charts tab of a dataset contains charts saved in the Dataset Chart, Segment Viewer, Cross Tabs, and Correlations views of the dataset.
- The Saved Charts tab of a tree contains node charts saved in the Chart view of the tree.

In the Saved Charts tab you can view, organize, and print the charts, or copy them to Microsoft Office documents for reporting.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Charts are organized in groups. Each group is represented by a separate tab at the top of the chart grid. The tabs are highlighted by the red frame in the example screenshot below. Click on the tab names to switch between the chart groups.



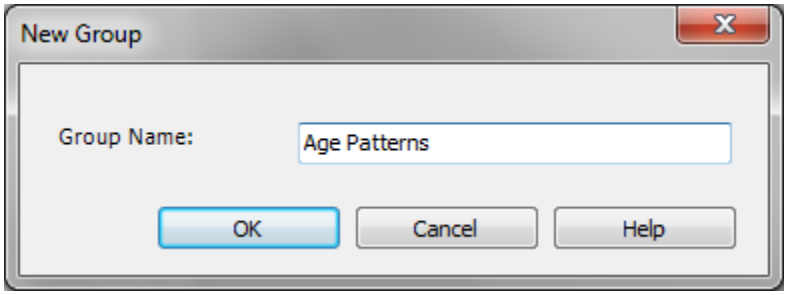
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To view any chart in detail in a separate window, double-click on it and resize its window as necessary. In this window you can view the chart details, copy or print it, etc. For more information, see the topic Single Chart View.

Chart Groups

You can create new chart groups and distribute the charts in these groups.

To create a new chart group, select the **New Group** command from the **Groups** menu, or right-click on the group tab name and select **New** from the context menu. Then specify the desired name in the dialog:



A new blank chart group with the new name will be created.

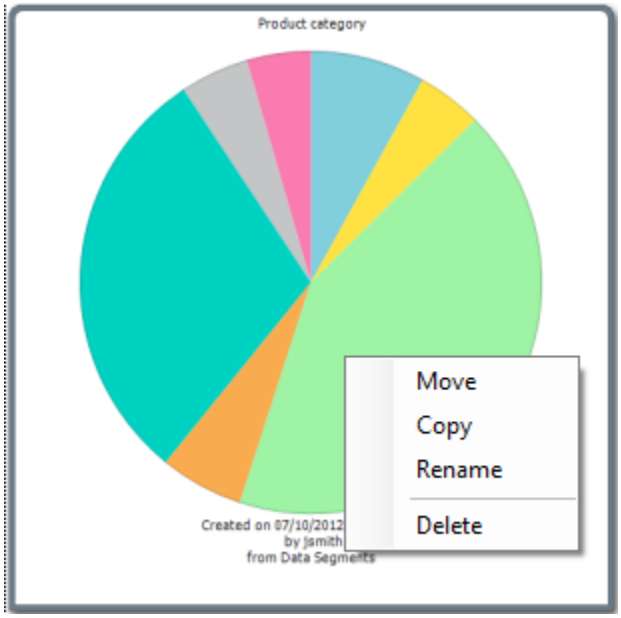
To rename an existing chart group, right-click on the group tab name and select the **Rename** command from the context menu. Enter the new name and click OK.

Operations with Charts

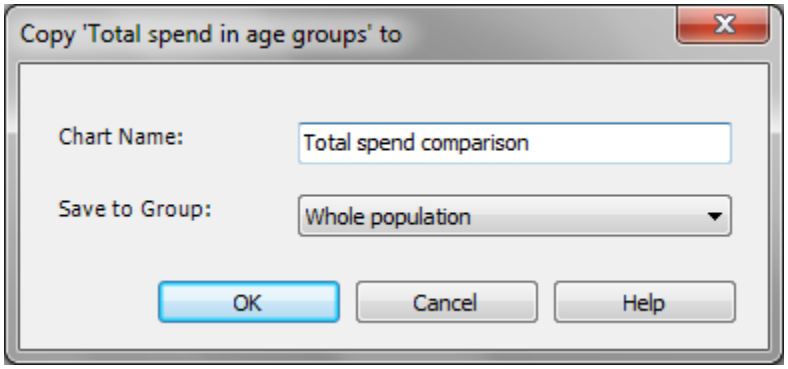
To change the position of any chart within its group, drag and drop it to the desired position.

To delete, copy, rename, or move a chart to a different group, right-click on the chart and select the appropriate command from the context menu.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- To delete a chart, select **Delete** from the context menu.
- To rename a chart, select **Rename** from the context menu, specify the new name for the chart, and click OK.
- Charts can be copied within the same chart group or to a different chart group. To copy a chart, right-click on it and select **Copy** from the context menu. Specify the name for the new chart and select the destination group. The current group is the default destination.



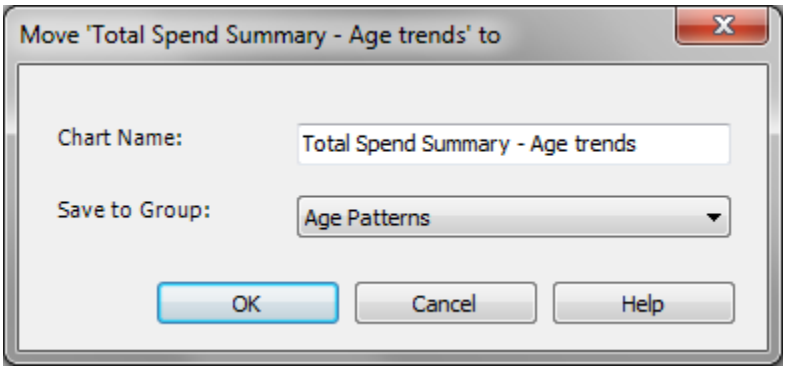
- There are two ways to move a chart to another group:

  a) drag and drop the chart to another group tab, or

  b) right-click on the chart and select **Move** from the context menu.

In the Move dialog, specify the new name for the chart and select the desired destination group from the Save to Group list

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also:

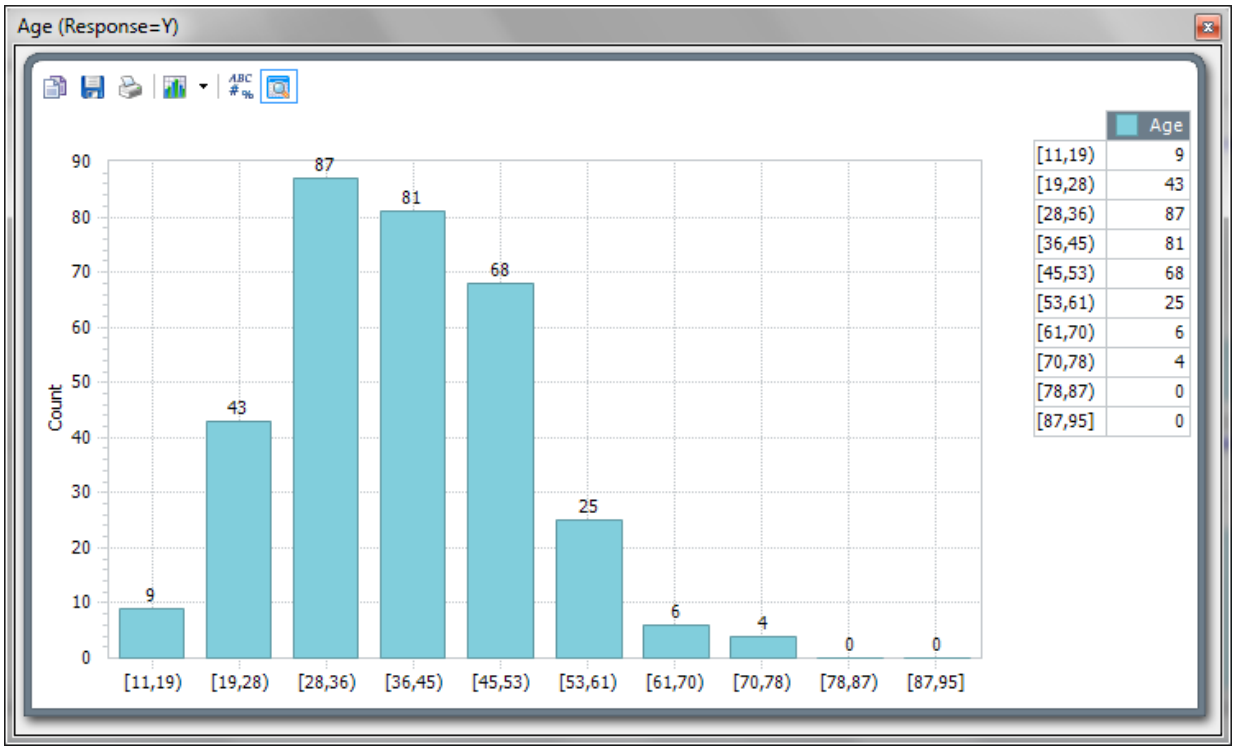
Single Chart View

Single Chart View

Several views in the application contain multiple charts: Segment Viewer in datasets, Chart in trees, and Saved Charts in both datasets and trees. In any of these views, you can open each individual chart in its own separate window, where you can view its details, copy or print it, etc.

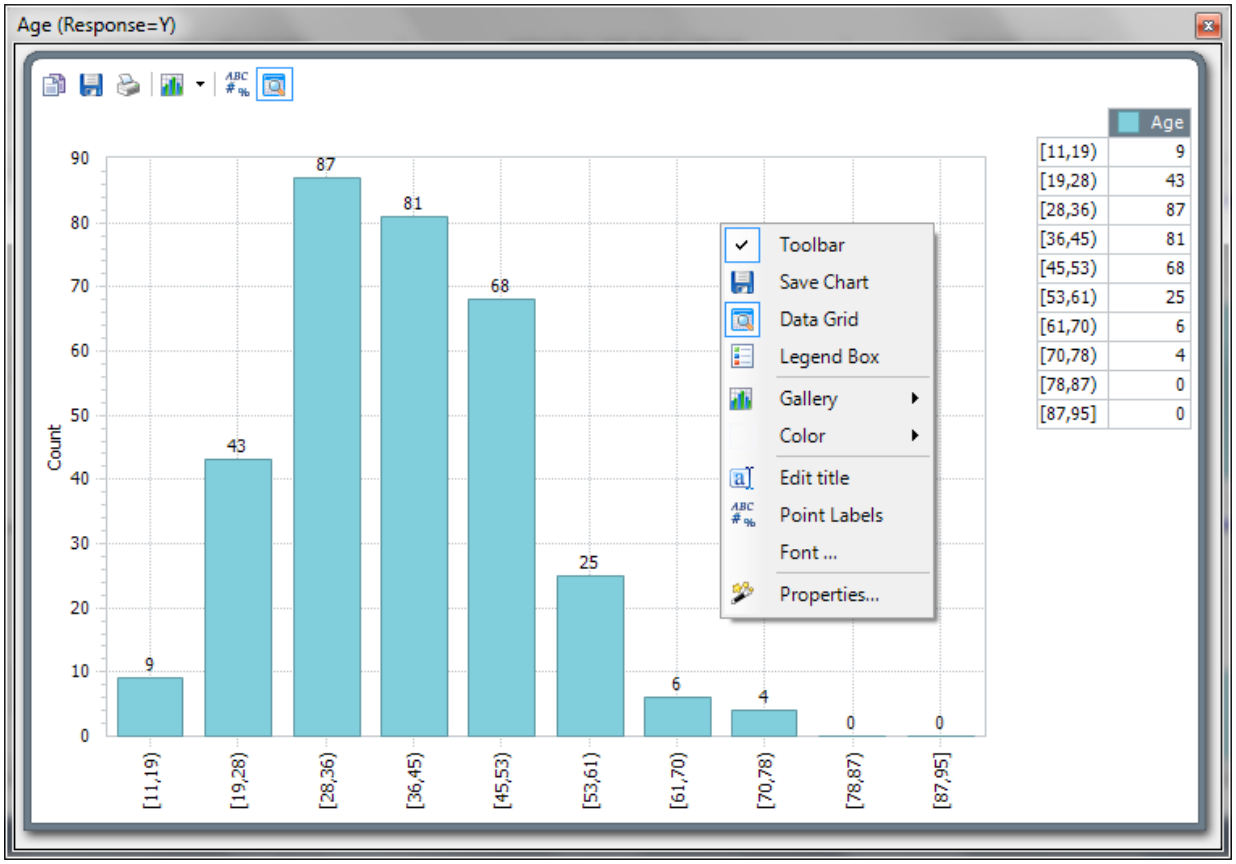
Double-click on the desired chart to view it in a separate window. The window provides a detailed view of the chart and shows the data grid representing the chart data. You can resize the window by clicking and dragging on its corners or borders.

Here is an example of a chart from Segment Viewer opened in its own window.



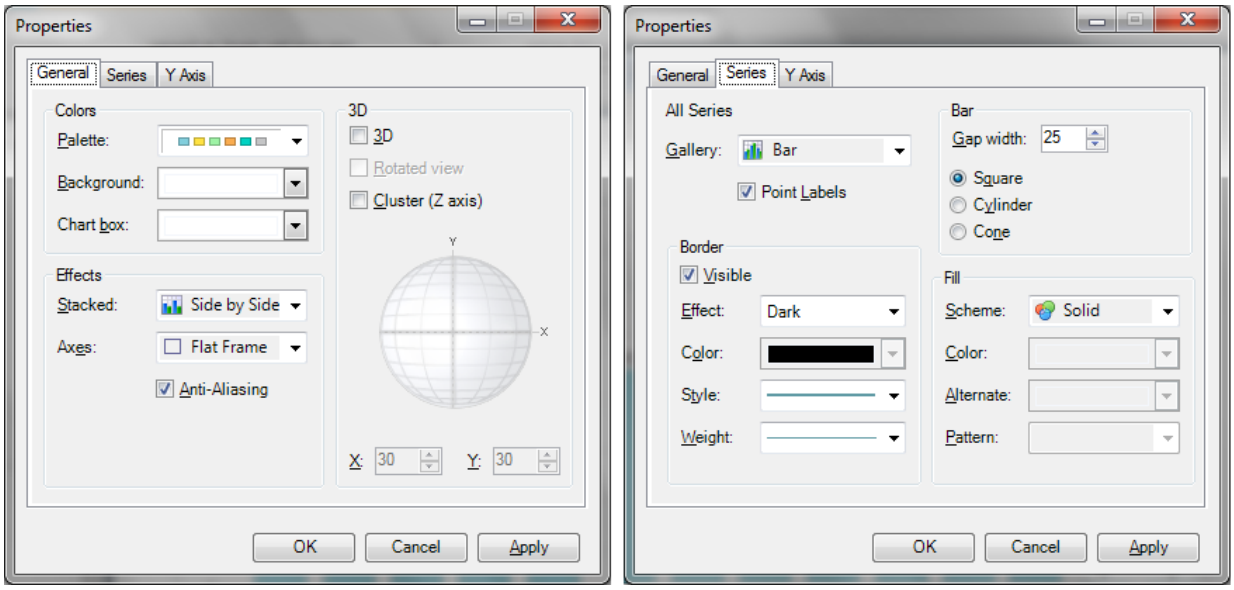
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To change the display properties of the chart, use the chart toolbar or right-click anywhere on the chart background. The following context menu appears when right-clicking on the chart background. For example, select Point Labels to show or hide the values on top of the bars; select Data Grid to show or hide the table with the chart data.

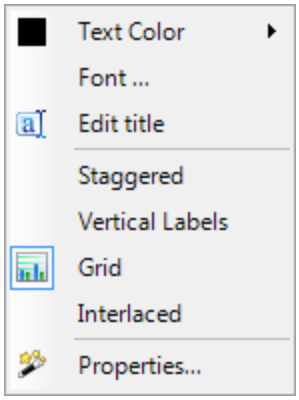


To change advanced properties of the chart, select Properties from the context menu. In the Properties dialog, you can choose the color palette, 3D or 2D view, effects, properties of the axis labels, show or hide grid lines, etc.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

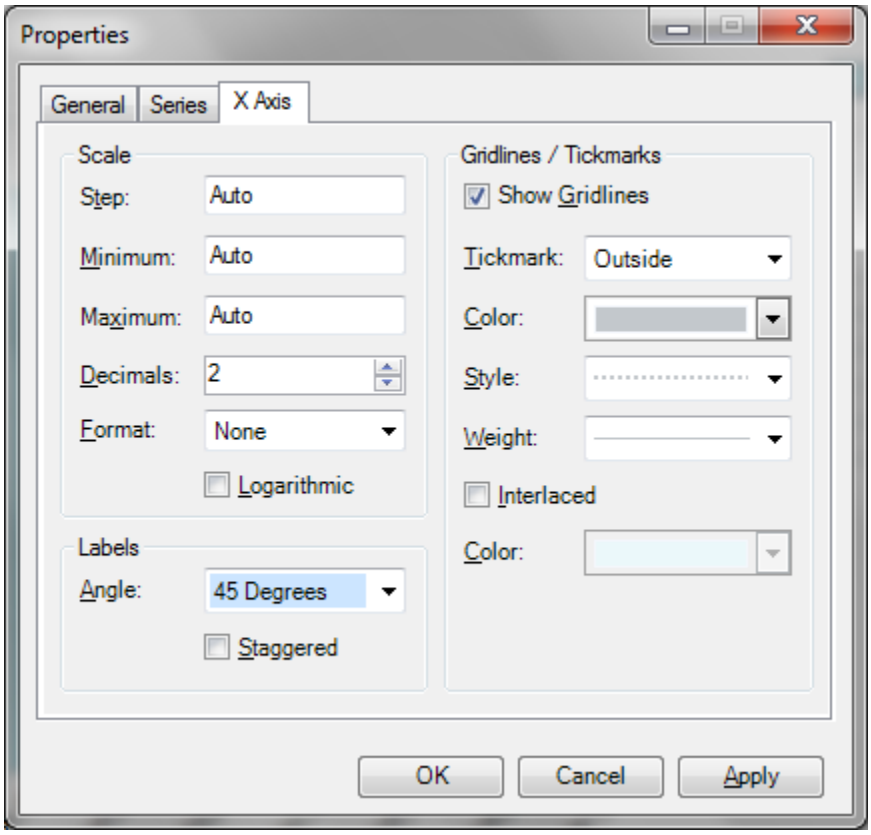


To change the orientation of the axis labels, right-click on any label of the desired axis (vertical or horizontal). The context menu will look like this:



Select *Staggered* or *Vertical Labels* if desired. If you want to place the labels at a 45-degree angle, select *Properties* and set *Angle* to 45 Degrees.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



As a result, the X-axis labels are displayed as follows:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

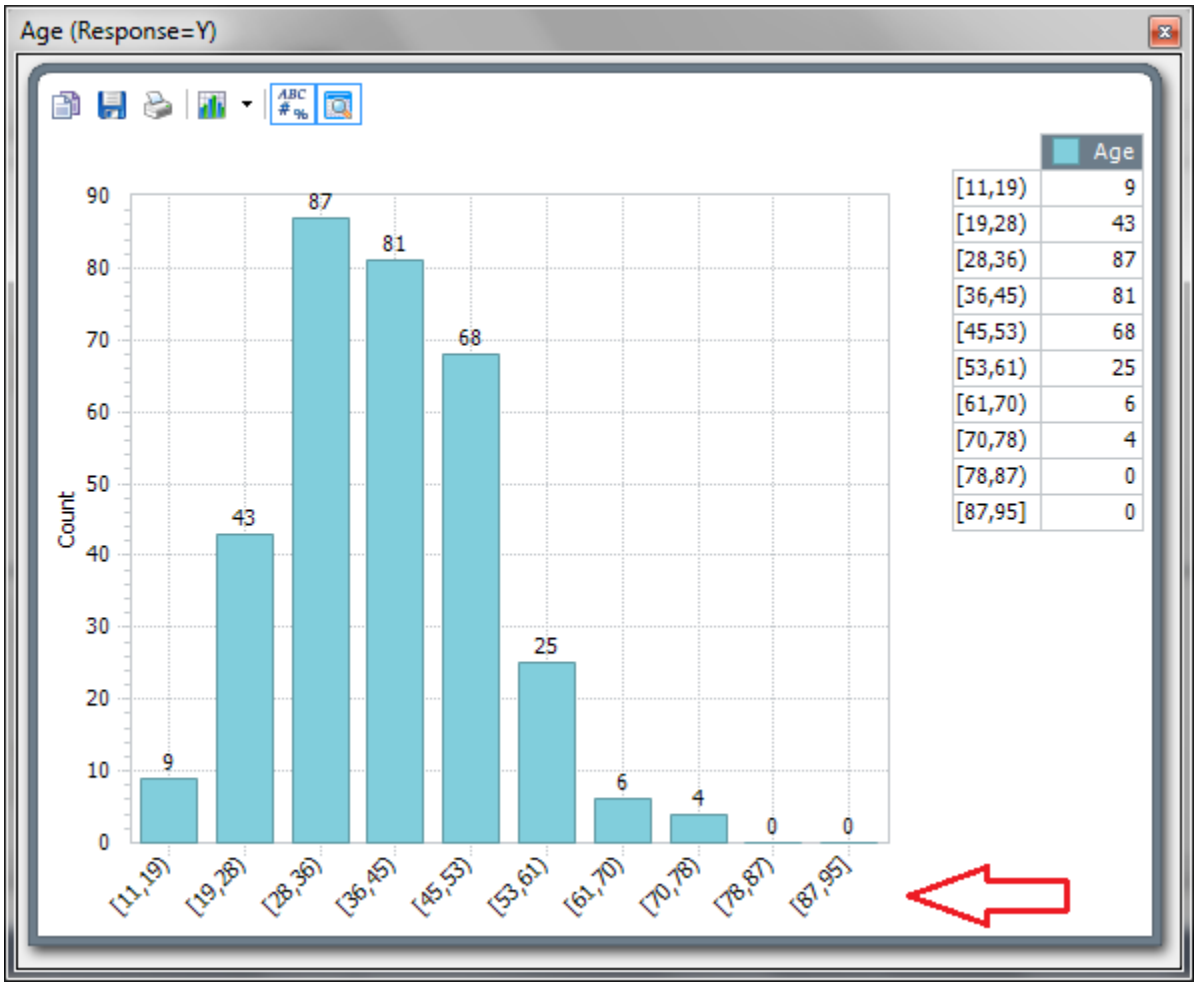


Chart Toolbar

Depending on the charts' parent view - Segment Viewer, Cross Tabs, Chart (in trees), etc., - its toolbar at the top left corner of the chart window can have one of the following two forms:

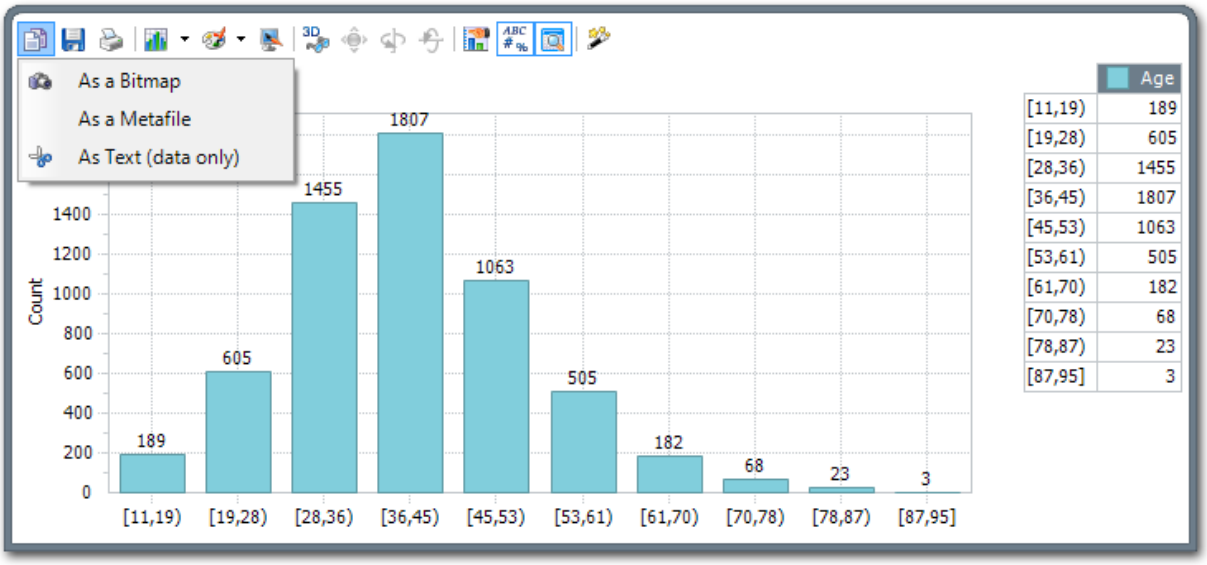




All icons in the Chart toolbar except the *Save*, *Copy*, and *Print* icons are shortcuts to the corresponding options and commands in the Properties dialog described above.

To copy a chart, click on the *Copy* icon in its toolbar, as shown in the screenshot below. Select *As a Bitmap* or *As a Metafile* from the context menu if you want to copy the chart as an image. Select *As Text* (*data only*) if you want to copy the underlying data, which is displayed in the data grid on the right-hand side. Then paste the copied image or text from the clipboard into the desired application.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

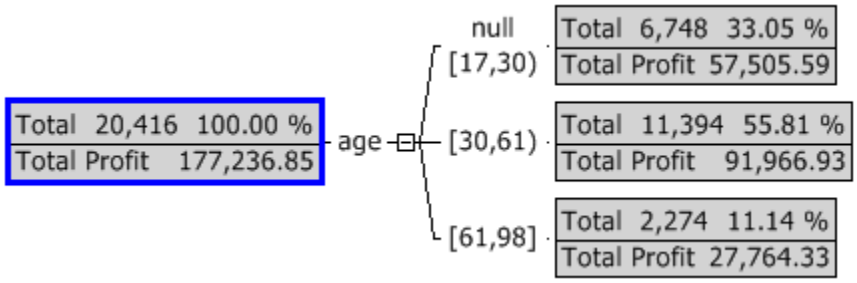


If you are viewing a chart from the Segment Viewer or Node Chart, you can save it in the Chart Library by clicking on the Save button .

See also: Saved Charts

Adding Node Calculations

Node calculations are expressions that are calculated and applied to each node in the strategy tree. Expressions can include various functions, operators, constants, and field names as variables.



In this example, **Total Profit** is a calculation from the expression **SUM*[Profit]*** where ***[Profit]*** is a field in the dataset that the strategy tree is based on.

Node calculations can be added in two ways:

- in the **Insert Strategy Tree** wizard
- through the **Tools | Calculations...** menu command
- by right-clicking any node and selecting **Calculations** from the context menu

When you add or edit node calculations, they are added or updated for all nodes of the tree. After tree growing operations such as Find Split, Force Split, and others, the existing calculations are automatically added to the newly created nodes.

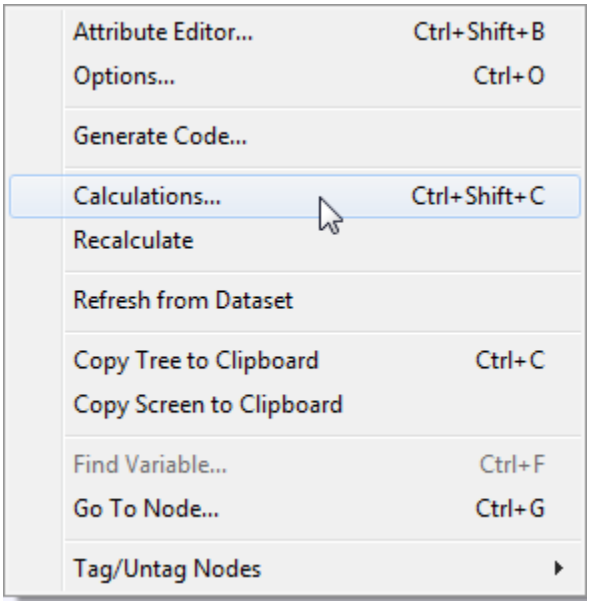
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Inserting a Node Calculation

Calculations inserted for a Strategy Tree are calculated and displayed for every tree node.

To insert a calculation, open the Strategy Tree (double-click on the Strategy Tree node on the workflow canvas or in the Project Explorer panel) and use one of the following ways:

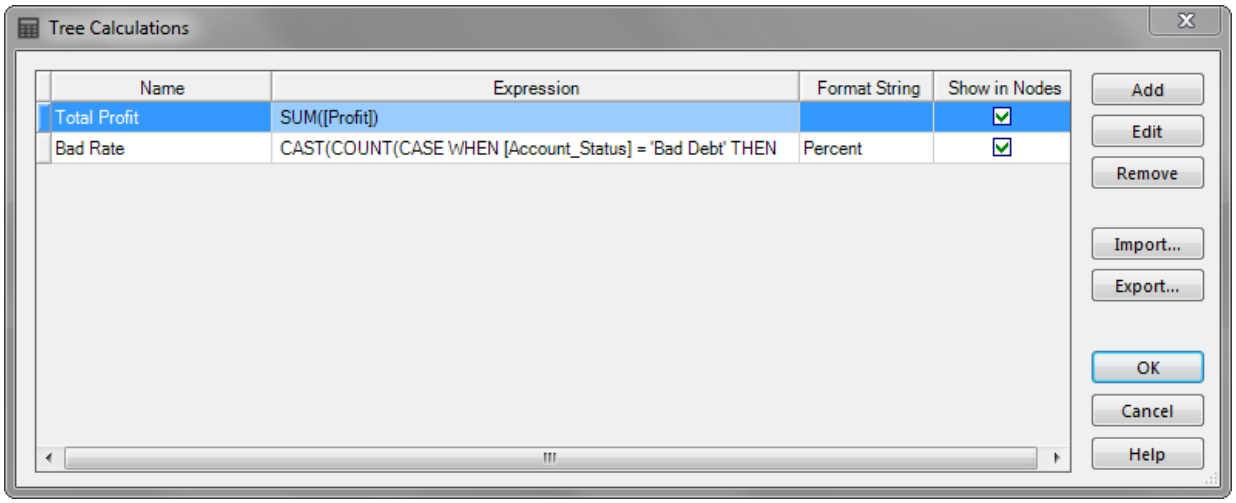
- Right-click anywhere in the Strategy Tree view and select **Calculations** from the context menu:



- Click the icon  in the toolbar
- Select the **Calculations** command from the **Tools** menu.
- Use the keyboard shortcut CTRL + SHIFT + C.

The Tree Calculations window opens. It displays existing node calculations (if any) and provides a set of tools for adding and managing node calculations.

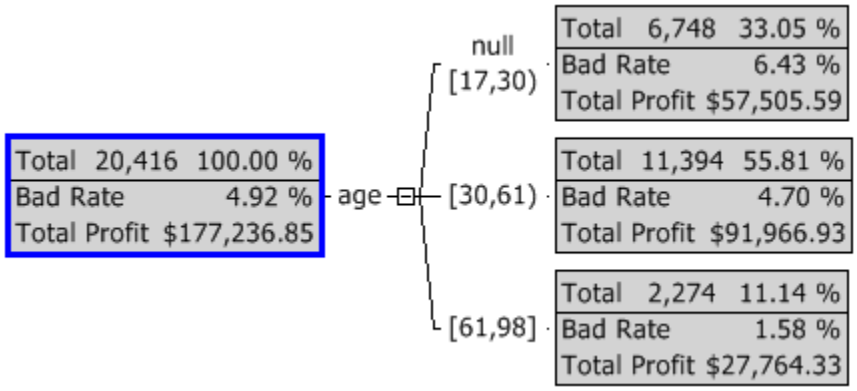
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Add	Add a calculation to the node. This button calls the expression editor to help in creating the calculations.
Edit	Edit an existing node calculation
Remove	Remove a node calculation

In this example, two calculations already exist in the strategy tree. See Figure 1.

Figure 1 - Node Calculation for strategy tree



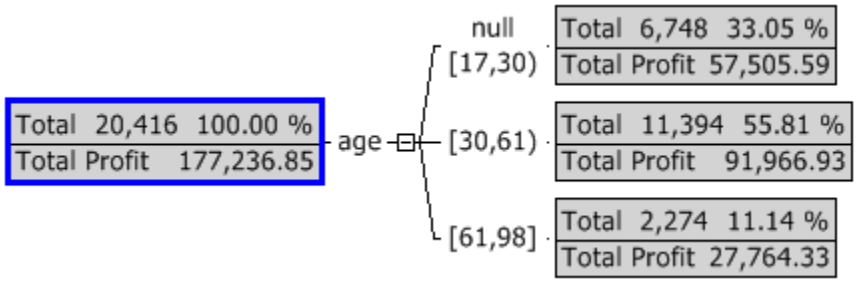
Click OK to calculate the expressions and apply them to the nodes in the strategy tree.

Expression Editor

Expression Editor for Node Calculations

Node calculations are SQL expressions that are calculated and applied to each node in the strategy tree. Expressions can include various functions, operators, constants, and field names as variables.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



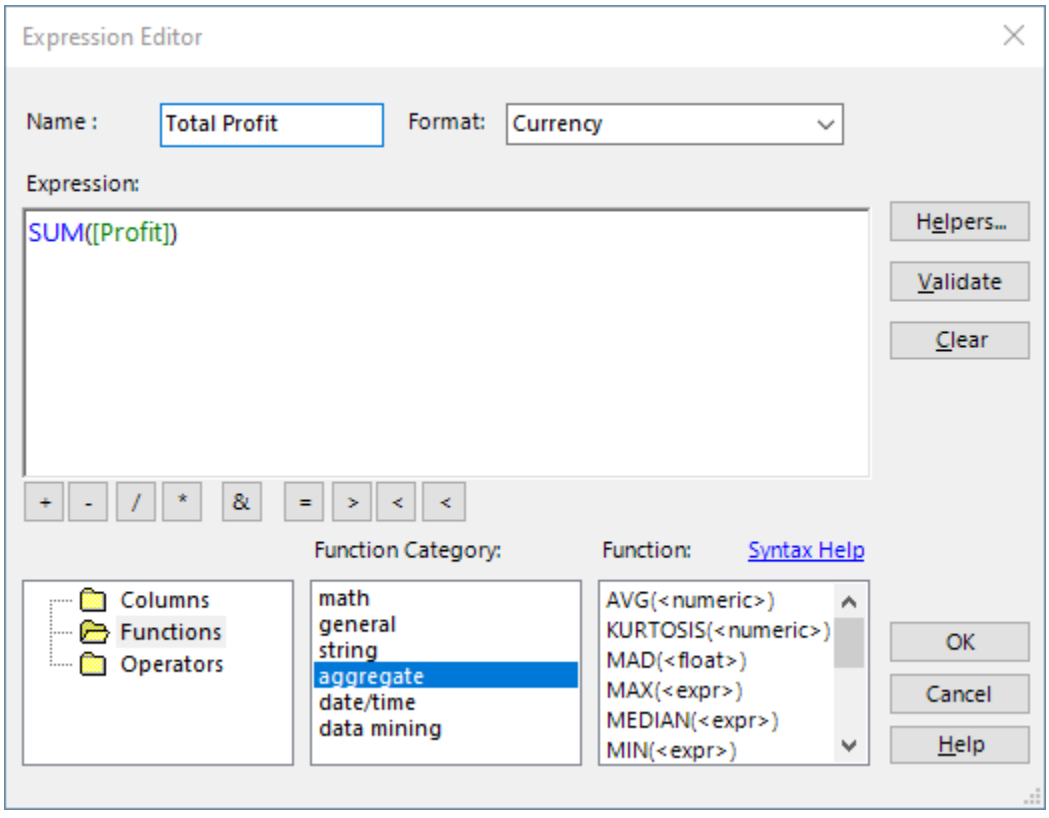
In this example, Total Profit is a calculation from the expression SUM[Profit] where [Profit] is a field in the dataset that the strategy tree is based on.

Node calculations can be added in the following ways:

- in the Insert Strategy Tree Wizard
- using the **Tools | Calculations** menu command
- by right-clicking any node and selecting **Calculations** from the context menu

When you add or edit node calculations, they are added or updated for all nodes of the tree. After tree growing operations such as Find Split, Force Split, and others, the existing calculations are automatically added to the newly created nodes.

To add node calculations after the strategy tree is inserted, right-click any node and select **Calculations** from the context menu. Then click Add. This brings up the Expression Editor for node calculations:



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The calculations created by the expressions will appear in all nodes in the strategy tree.

Helpers	Helpers are wizards that help you create the most common expressions for the node calculations. Each Helper is described in detail in a separate topic.
Validate	Allows you to validate the syntax of the expression entered in the expression box.
Clear	Clears the expression from the expression box.

The expression syntax for node calculations is very flexible. Note that you can also use *node-specific* syntax to refer to the value of any expression at the root node or parent node.

To refer to the root or parent node, prefix an identifier of an existing expression with **[$root$]** or **[$parent$]**, respectively. For example:

A = sum([field_name1])

ratio = [A]/[$parent$].[A]

global = [A]/[$root$].[A]

At any node, "ratio" is the value of expression 'A' at this node divided by the value of A at its parent node, while "global" divides A by the value of A at the root node of the tree.

Also possible is [$self$]. You can also chain them up like this:

[$parent$].[$parent$].[a]

which would refer to the value of 'a' in the grand-parent node.

Other examples:

a = sum([Profit])

b = a/[$root$].[a]

c = a/[$parent$].[a]

d = a/[$parent$].[$parent$].[a]

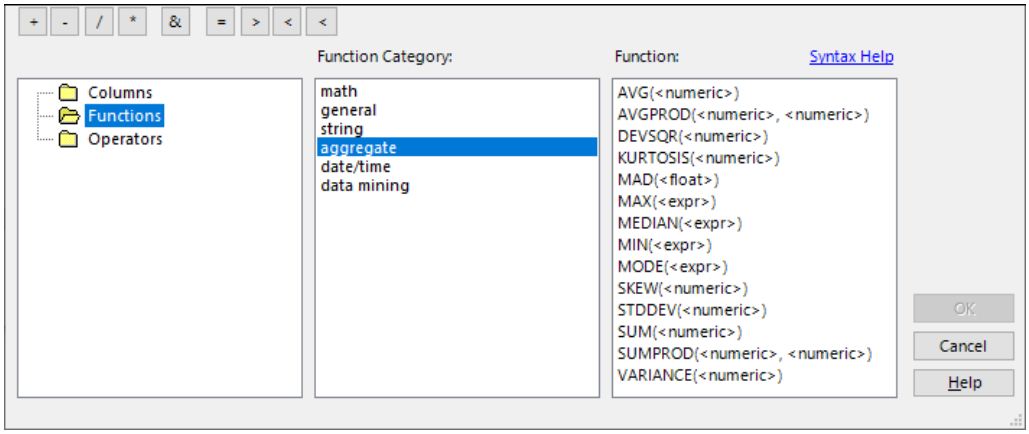
A = SUM([Amount_Lost])

Loss_Index=[A]/[$root$].[A]

Note the flexibility: you can reuse these calculations in other trees. For example, if you copy your tree to another dataset, these calculations will still be valid as it will refer to the root or parent nodes in the new tree rather than using constants from the source tree.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Columns, Functions and Operators

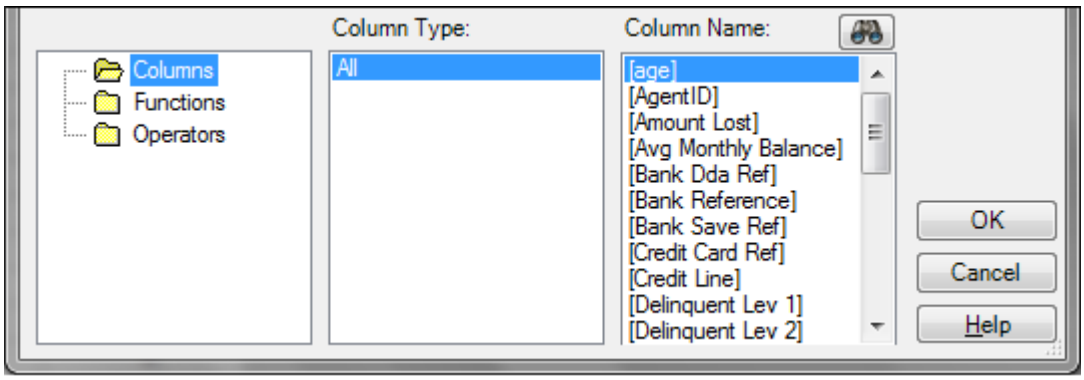


The Columns, Functions, and Operators folders provide help in building your expressions. Columns will display all the columns in the dataset the tree is based on. The Functions and Operators lists provide standard functions and operators for SQL expressions.

See also:

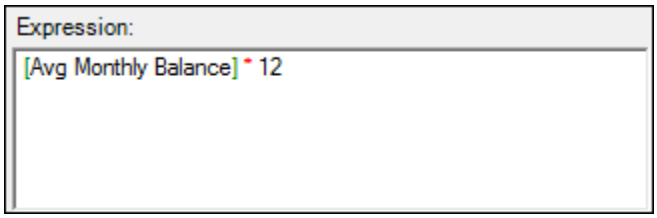
Columns
Functions and Operators
Function Syntax
Expression Helpers

By highlighting the **Columns** folder, a list of columns from the dataset will appear under **Column Name**. The SQL expression must used the exact name of the column in your data.



If you want to perform a calculation using a column, double-click on the column name and it will appear in the expression box. The column can now be used as part of the calculation.

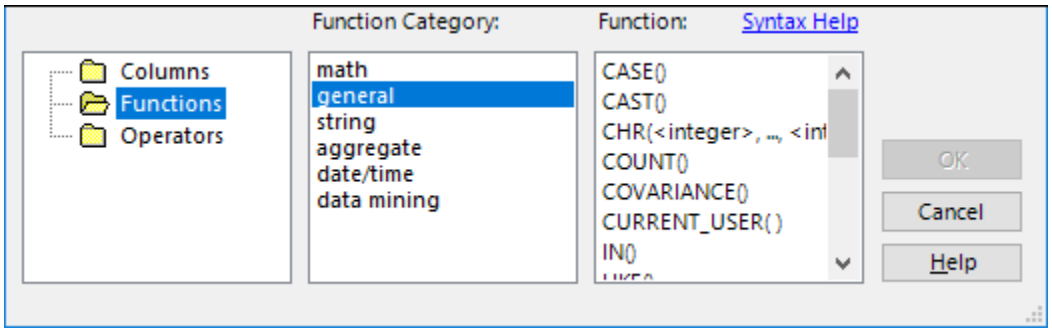
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



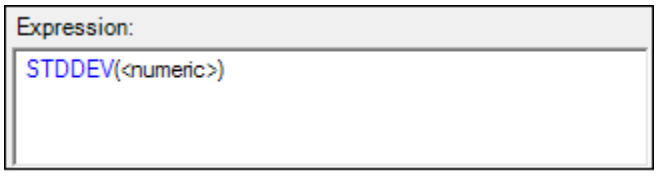
Expressions can always be typed manually into the expression box. When doing this, be aware that the column name must always be surrounded by the square "[" brackets.

Functions

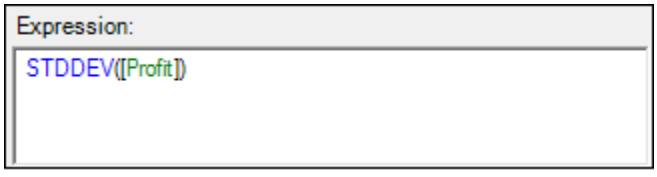
The **Functions** folder contains most commonly used mathematical, string, date/time, data mining, and other SQL functions. The **Function Syntax** box lists the functions of each category. The type of their arguments is indicated in parentheses. For information and examples on these functions, see the Function Syntax topic.



Double-click on a function, and it will appear in the expression box.



Replace the type in the parentheses with a value or a column name. A column name can be entered in the expression box by selecting the Columns folder and double-clicking the desired column name. The type of the entered value or column must be compatible with the type of the function argument. For example:



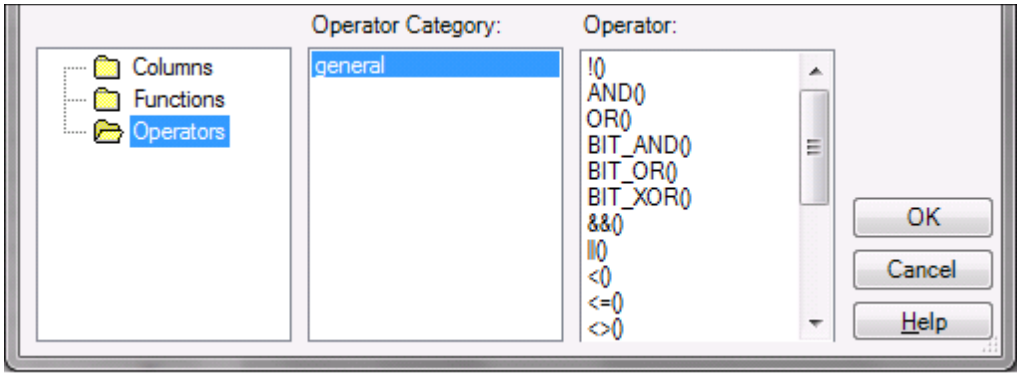
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

where [Profit] is a column in the dataset.

As with all expressions, these functions can be typed manually into the expression box.

Operators

The operators list contains most common operators used in SQL Queries. They include addition, subtraction, multiplication and division operators, comparison operators (<,>,=,>=,<=, etc.), logical operators AND, OR, NOT, etc., and bitwise operators.

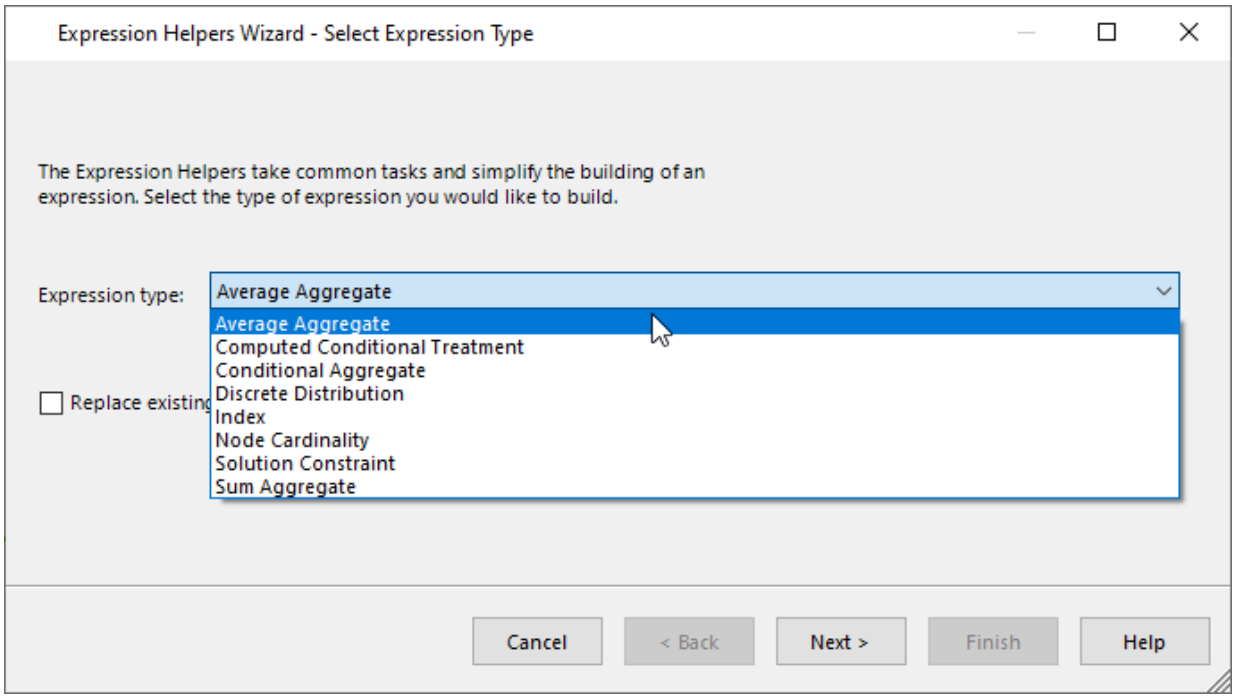


Expression Helpers

Expression Helpers

The purpose of the helpers is to help you create most commonly used expressions using a simple wizard-like process. After you construct an expression using a helper and click Finish in the helper wizard, the expression appears in Expression Editor, so you can edit it further, if necessary, before the calculation is added.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Average Aggregate and Sum Aggregate

The Average Aggregate and Sum Aggregate helpers allow you to create a new node calculation that is an aggregate function (sum or average) of a numeric variable. Example expressions create by these helpers are:

AVG([Age])

SUM([Profit])

where **Age** and **Profit** are variables in the parent dataset of the strategy tree.

For details, see the topic Average and Sum Aggregate Helpers.

Conditional Aggregate

A conditional aggregate allows you to display a node calculation involving aggregate functions that is based on a particular condition imposed on the nodes. For example, if you would like to see the average account balance only for the nodes where the average age is greater than 50, the expression constructed by the helper is

CASE WHEN AVG([Age]) > 50 THEN AVG([Account Balance]) ELSE 0 END

For details, see the topic Conditional Aggregate.

Computed Conditional Treatment

Allows you to assign any alphanumeric expression to the nodes depending on a condition you specify. The condition must include an aggregate function (SUM or AVG) and is applicable to numeric variables only.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Note: the result of applying this helper is not a treatment but a calculation whose result is a string value. If you want to transform this result into a treatment, use the menu command **Tools | Treatment From Calculation...** after adding the calculation.

For details, see Computed Conditional Treatment.

Index

This type of node calculation is the normalized aggregate function (sum or average) of a selected numeric variable. The normalization coefficient is the inverse of the sum (average) of the variable at the root node. Thus, the value of the expression at a given node is the sum (average) of the selected variable at this node divided by the sum (average) of the same variable at the root node. It follows from the definition that the value of the index expression at the root node is always equal to 1.

For example, suppose you have your sales data over several regions and the question is how well each region is doing relative to other regions as well as overall (i.e., you want to see only relative sales performance rather than exact dollar amounts). You have a strategy tree with the split forced on the Region variable. Using the index helper, you can represent the overall sales as 1 (shown at the root node) and the regional sales as numbers between 0 and 1 (shown in the child nodes).

For details, see Index Helper.

Discrete Distribution

Allows you to show the percentage, or count, of a discrete variable category within each node.

For details, see Discrete Distribution.

Node Cardinality

Shows the cardinality (the number of distinct values) of the specified variable within each node.

For details, see Node Cardinality.

Solution Constraint

The Solution Constraint Helper is provided to assist with the optimization task workflow. It allows you to assign constant values to specific nodes selected for optimization. These constant values will serve as the upper or lower limits for the values of the optimized solution in the corresponding nodes. These values can be used in the Optimization Helper.

For details, see the topics Solution Constraint and Optimization in Strategy Trees.

Option: Replace Existing Expressions

If you are editing an expression and this option is unchecked (which is the default), the expression created by the helper will be appended to the existing expression in the expression builder. Otherwise, the existing expression displayed in the expression builder will be overwritten.

Average and Sum Aggregate Helpers

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

An aggregate function is a single-valued function that performs a computation on a set of values rather than on a single argument. The simplest examples are MIN(), MAX(), SUM(), and AVG(). When used in a node calculation of a Strategy Tree, aggregate functions perform a computation on all records within the given node.

For example, if Profit is a column in your dataset, the expressions to calculate the average and total Profit within each node is AVG([Profit]) and SUM([Profit]), respectively.

	Status2	
Bad Debt	1,005	4.92 %
OK	19,411	95.08 %
Total	20,416	100.00 %
Average Profit		$8.68
Total Profit		$177,236.85

The Average Aggregate and Sum Aggregate helpers assist in creating expressions containing sum and average aggregate functions. You can edit the output of these helpers to substitute any other supported aggregate function for SUM or AVG or append custom statements.

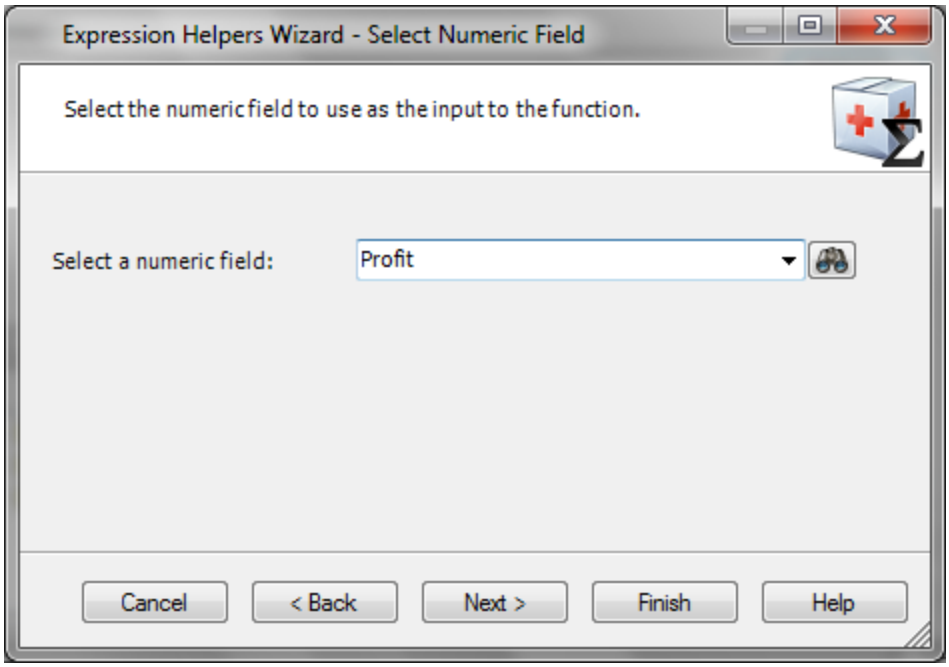
The expression syntax is standard SQL. Column names should be enclosed in square brackets. See Function Syntax for details.

Supported Aggregate Functions

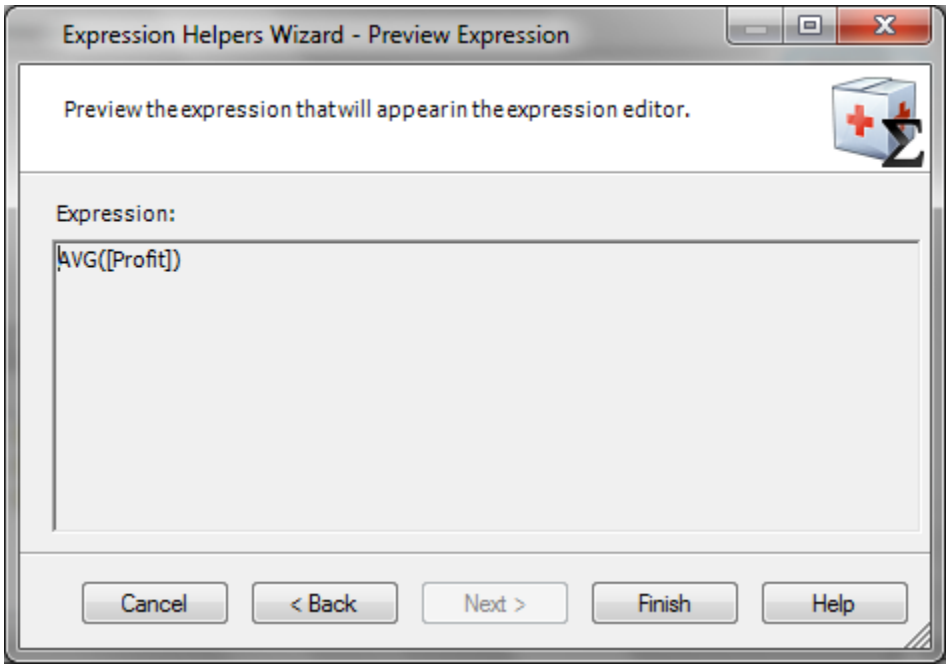
AVG(<numeric>)	Allows you to take the average of a group of values. For example, AVG(Age) where Age is a field in the dataset
MAX(<expr>)	Gives you the maximum value of a group of values. The value can be numeric or string. If the value is of type string, the ordering is alphabetical.
MEDIAN(<expr>)	Gives you the median value that occurs within a group of values. The value can be numeric or string. If the value is of type string, the ordering is alphabetical.
MIN(<expr>)	Gives you the minimum value that occurs within a group of values. The value can be numeric or string. If the value is of type string, the ordering is alphabetical.
MODE(<expr>)	Gives you the mode of the values that occurs within a group of values. The value can be numeric or string. If the value is of type string, the ordering is alphabetical.
STDDEV(<numeric>)	Calculates the standard deviation of a group of numeric values.
SUM(<numeric>)	Calculates the sum of a group of numeric values.
VARIANCE(<expr>)	Calculates the variance within a group of numeric values

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

After choosing either the *Average Aggregate* or *Sum Aggregate* as Expression type, click Next and select the desired numeric field as the argument for your aggregate function.



Click Next to preview the constructed expression:



Click Finish to add the expression to the Expression Editor and edit it if necessary. See the topic Preview Expression for details.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

See also: Expression Helpers: Overview, Conditional Aggregate, Computed Conditional Treatment.

Conditional Aggregate

The Conditional Aggregate Helper allows you to add an aggregate calculation at the nodes that meet a certain condition.

The helper produces an expression of the form **CASE WHEN A THEN B ELSE C**, where A is a relational expression, B is a simple aggregate expression, and C is a constant. If desired, you can edit the resulting expression to replace A, B, and C with custom expressions.

Condition A can be of two types:

- Based on an aggregate function of a continuous numeric field. For example: *"If the sum of the field X at a given node is greater than R, then ... "*.

  Example: **CASE WHEN SUM([Amount Lost]) > 5000 THEN ...**

- Based on the frequency of a value of a discrete field. For example: *"If the percentage of the value V of the field X at a given node is less than R, then ...".*

  Example: **CASE WHEN CAST(COUNT(CASE WHEN [Status] = 'Bad Debt' THEN 1 ELSE null END) AS DOUBLE) / CAST(COUNT(*) AS DOUBLE) >= 0.8 THEN ...**

  Note: The CAST ... AS DOUBLE operator takes care of the data type compatibility and conversion. It is inserted automatically whenever necessary.

Expression B is a simple aggregate. For example, **SUM ([Profit])** or **AVG ([Profit])**

Examples:

1) In the case of a continuous condition field, an example of the expression *Exp1* built by the helper is

CASE WHEN SUM([Amount Lost]) > 5000 THEN AVG([Age]) ELSE 0

In this example, the average of Age will be displayed as the value of *Exp1* at all nodes such that the total of Amount Lost exceeds 5000. At all other nodes, the value of *Exp1* will be zero.

2) In the case of a discrete condition field, an example of the expression *Exp1* built by the helper is

CASE WHEN CAST(COUNT(CASE WHEN [Status] = 'Bad Debt' THEN 1 ELSE null END) AS DOUBLE) / CAST(COUNT(*) AS DOUBLE) > 0.8 THEN AVG([Age]) ELSE 0

In this example, the average of Age will be displayed as the value of *Exp1* at all nodes such that the percentage of records with Status = 'Bad Debt' exceeds 80%. At all other nodes, the value of *Exp1* will be zero.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Note:

- all String and Boolean fields are discrete;

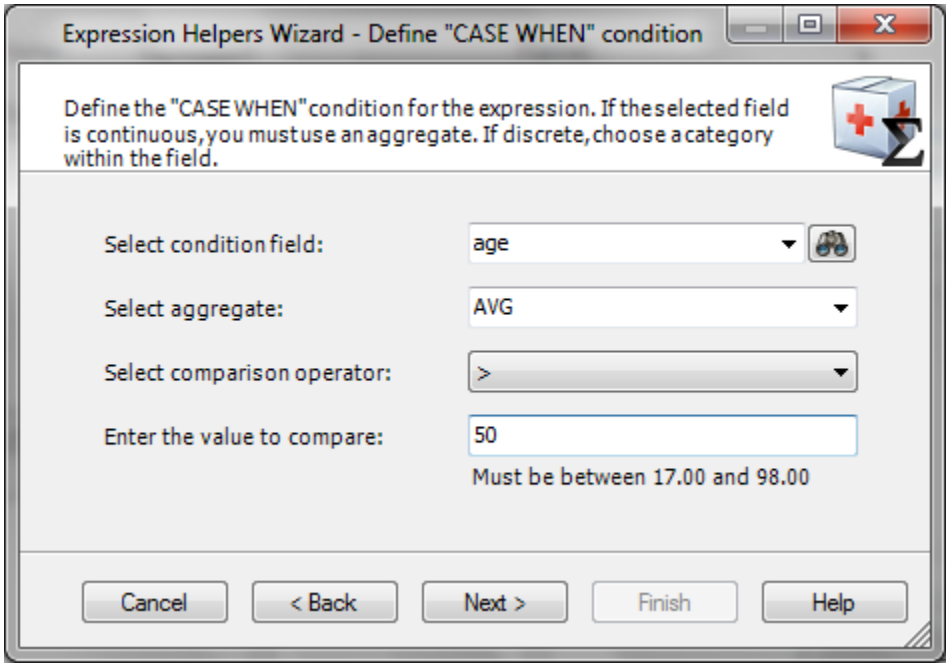
- in this helper, variables of the types Number, Integer, Time, or Timestamp whose cardinality is less than 50 are considered to be discrete.

The wizard allows you to specify the calculation in two steps:

1. specify the condition imposed on the nodes
2. specify the calculation to be displayed depending on the condition

The case of a continuous condition field.

In this case, the condition is based on an aggregate function of a continuous numeric field at a given node.

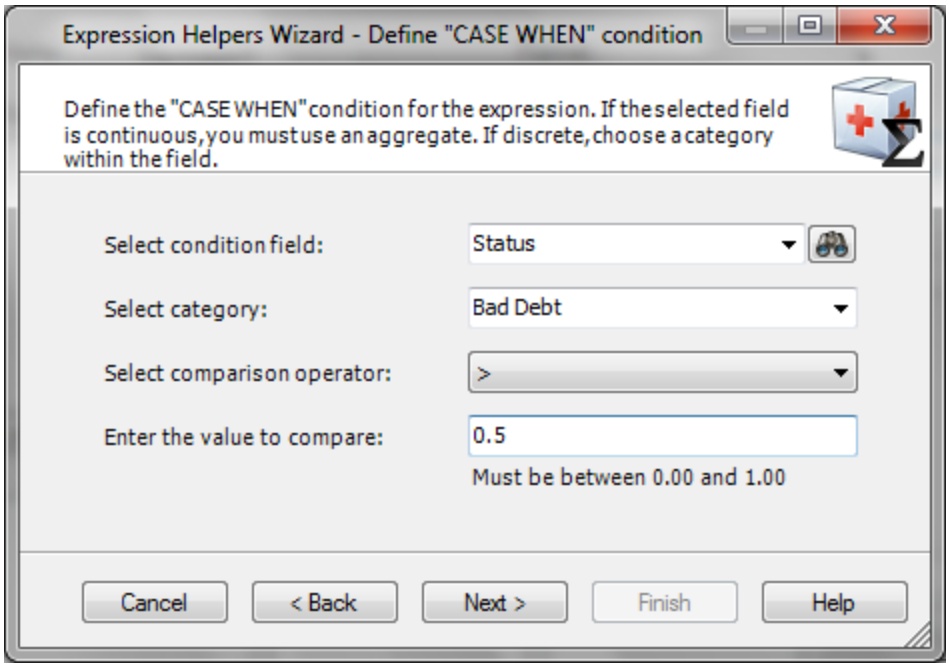


In the above example, the condition constructed at this step is "*If the average age at a given node is greater than 50...*", or **CASE WHEN AVG([Age]) > 50 THEN ...**

The case of a discrete condition field.

In this case, the IF condition is based on the percentage of a value of a discrete field at a given node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



In the above example, the IF condition constructed at this step is *"If the percentage of Bad Debt in the field Status at a given node is greater than 50%, then ...".*

The resulting expression will look like this:

CASE WHEN CAST(COUNT(CASE WHEN [Status] = 'Bad Debt' THEN 1 ELSE null END) AS DOUBLE) / CAST(COUNT(*) AS DOUBLE) > 0.8 THEN AVG([Age]) ELSE 0

Note: The CAST ... AS DOUBLE operators take care of the type compatibility and conversion and are inserted automatically whenever necessary.

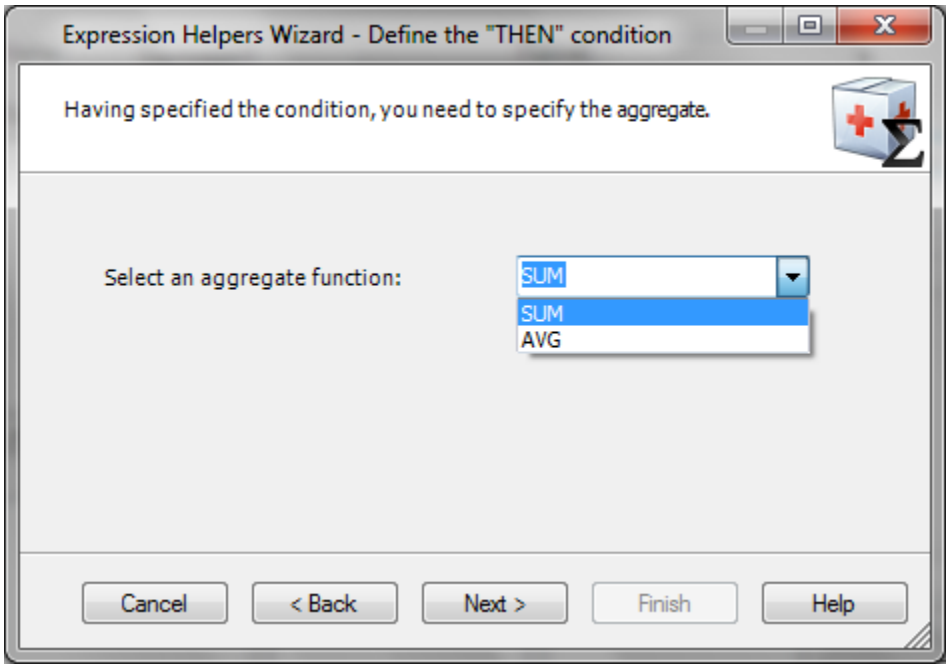
Summary:

Select condition field	Select the field to be used for the CASE WHEN part of the expression. In the above example, the field **age** is selected.
Select aggregate	Only available when the condition field is continuous. Choose the aggregate function for the condition. In the above example, it is **AVG[age]**.
Select category	Only enabled if the selected condition field is discrete. This list contains all categories of the field. Pick the category on which to base the condition.
Select comparison operator	Choose a comparison operator for the condition.
Enter the value to compare	Enter the threshold value for your comparison condition. If the selected field is continuous, this must be the value to compare the average or the sum of the selected variable. If the condition field is discrete, the threshold value must be a number between 0 and

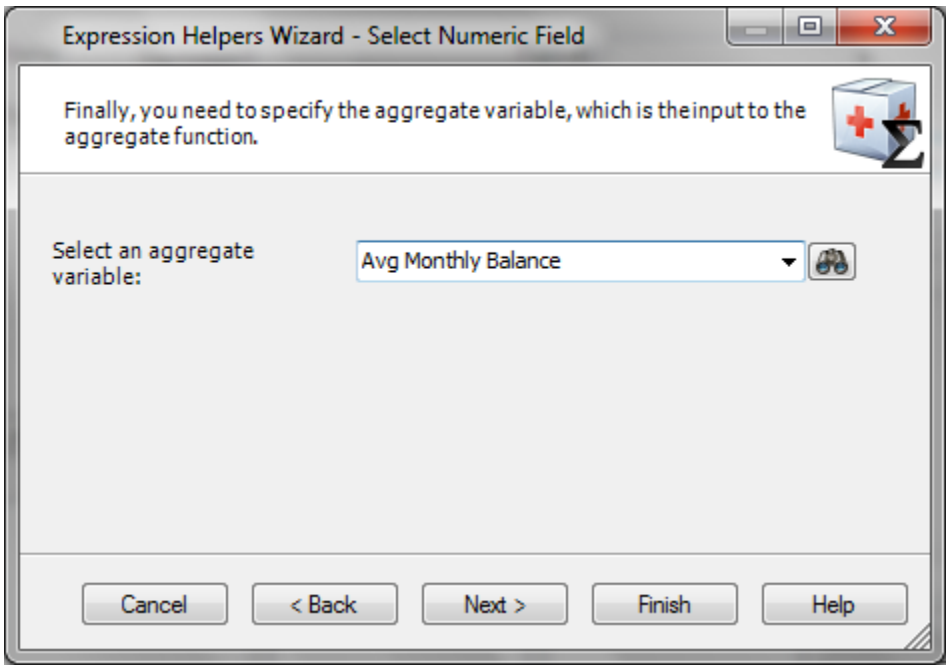
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

1 (the percentage of the selected category divided by 100 is compared to this threshold).

Click Next. At the next step, choose the aggregate function for the calculation to be displayed if the condition specified at the previous step **holds true**.



Click Next. Select a field to be the argument for the aggregate function you selected at the preceding step:



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Assuming that we chose SUM as an aggregate function and Avg Monthly Balance as the argument, we obtain the expression **CASE WHEN SUM([Amount Lost]) > 5000 THEN SUM([Avg Monthly Balance]) ELSE 0**

Click Next to preview the resulting expression and add it to the Expression Editor. See Preview Expression for details.

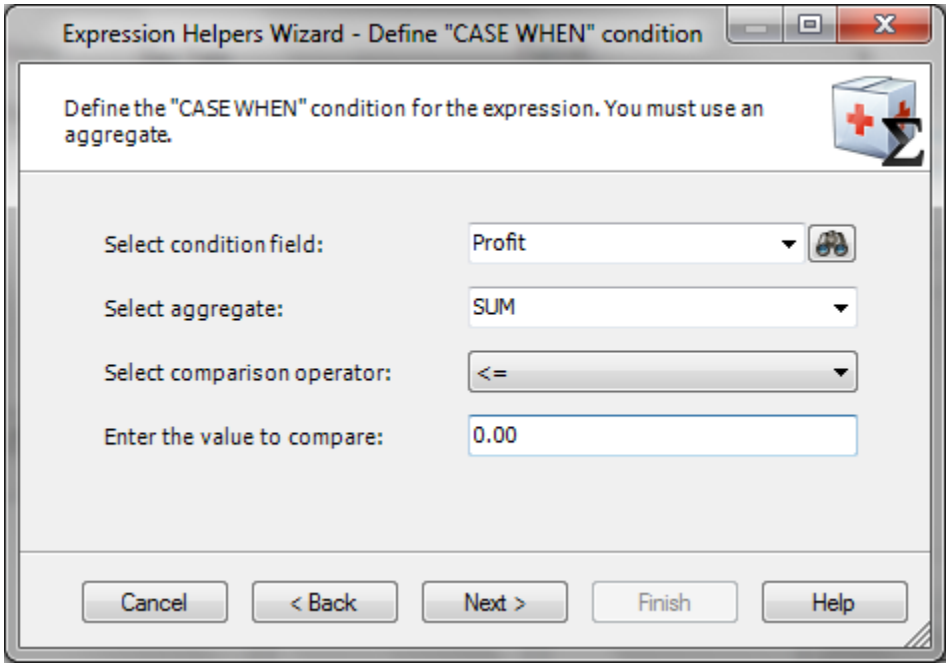
See also:

Computed Conditional Treatment

Computed Conditional Treatment

The Computed Conditional Treatment helper allows you to assign any string-valued expression to the nodes depending on a condition you specify.

The first step of the wizard prompts for the condition field, the aggregate function to be applied to it (SUM or AVG), the comparison operator, and the constant value to compare.



In the above example, this step of the wizard constructs the following condition:

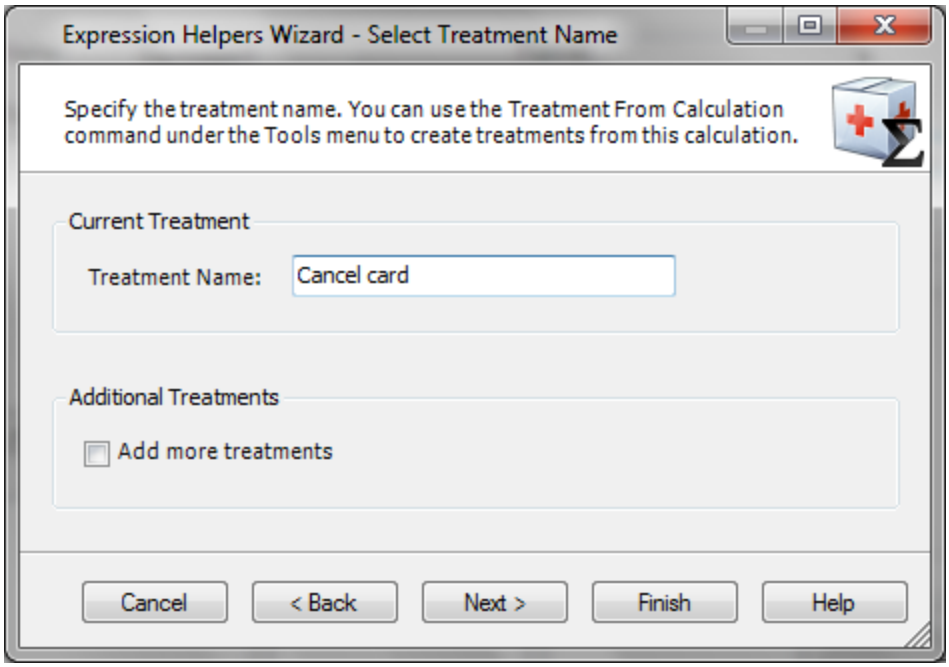
CASE WHEN SUM([Profit]) <= 0 THEN ...

Click **Next >** to specify the expression text.

At the second page of the Computed Conditional Treatment Wizard, enter the string value to be displayed at the nodes that satisfy the condition specified at the previous step. For example, it can be a treatment such as "Cancel card", "Platinum Card", "Increase Credit Line by 5000", etc.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Selecting the **Add more treatments** option enables you to define more than one condition and assign text expressions to nodes based on the other conditions you specify.



Click the **Next >** button to preview the resulting expression or define a new condition (if the **Add more treatments** checkbox is checked).

Note: the result of applying this helper is not a treatment but a calculation whose result is a string value. If you want to transform this into a treatment, use the menu command **Tools | Treatment From Calculation...** after adding the calculation.

See also:

Expression Helpers: Overview

Index Helper

This type of node calculation is the normalized aggregate function (sum or average) of a selected numeric variable. See the Expression Helpers topic for the summary.

The meaning of the expression being inserted is either

(SUM (Variable X) over the node) / (SUM (Variable X) over the whole dataset)

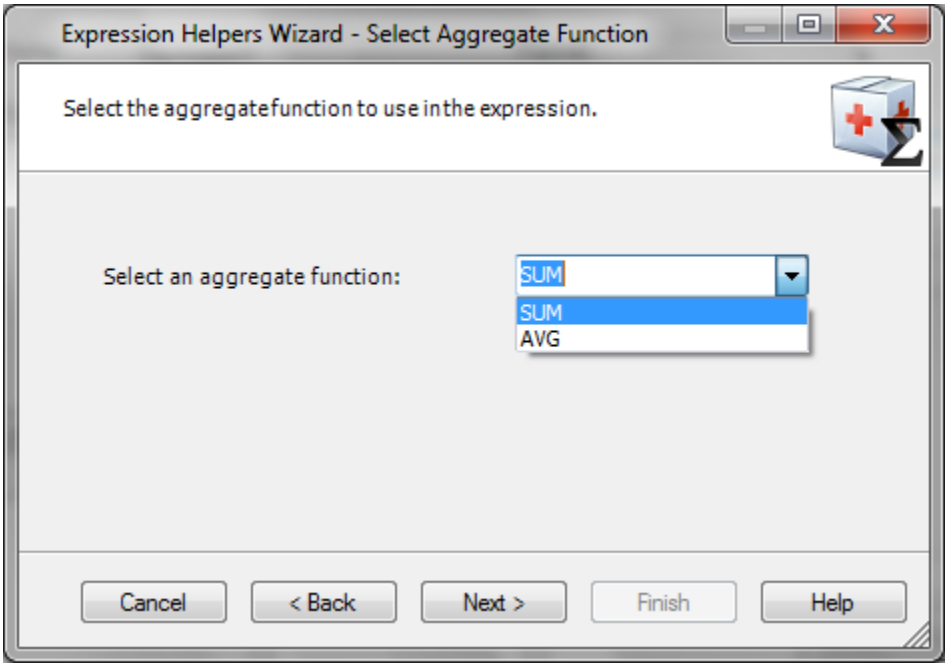
or

(AVG (Variable X) over the node) / (AVG (Variable X) over the whole dataset)

By the whole dataset we mean the parent dataset of the tree. The whole dataset is represented by the root node of the tree.

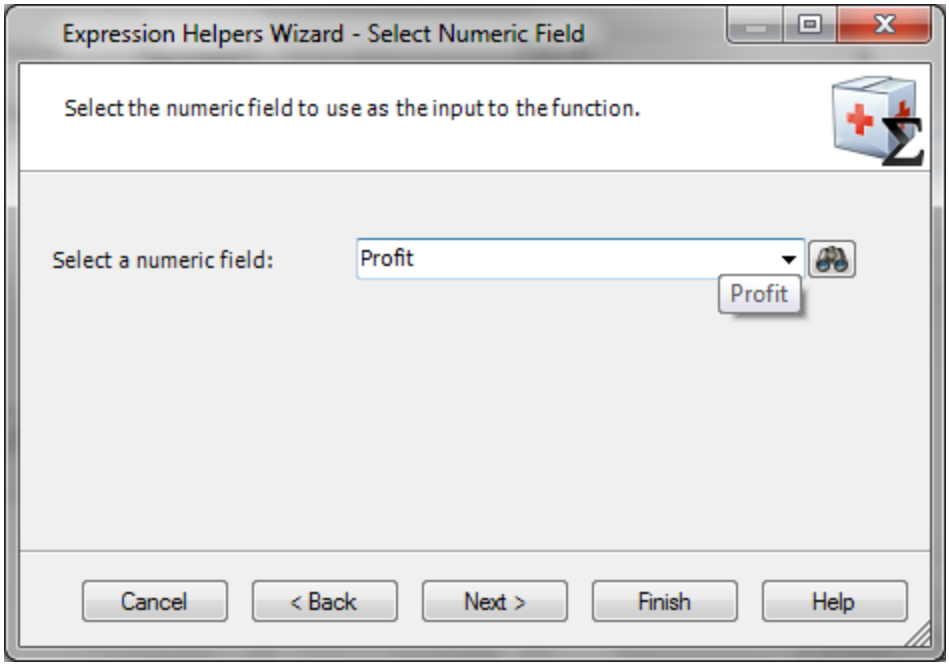
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

At the first step of the Index Helper wizard, select the desired aggregate function:



Select the desired aggregate function for the expression. Note that you can always edit the expression after it is inserted in the Expression Editor if you need to construct a more complex aggregate function.

Select a numeric field to be the argument for the aggregate function chosen at the previous step.



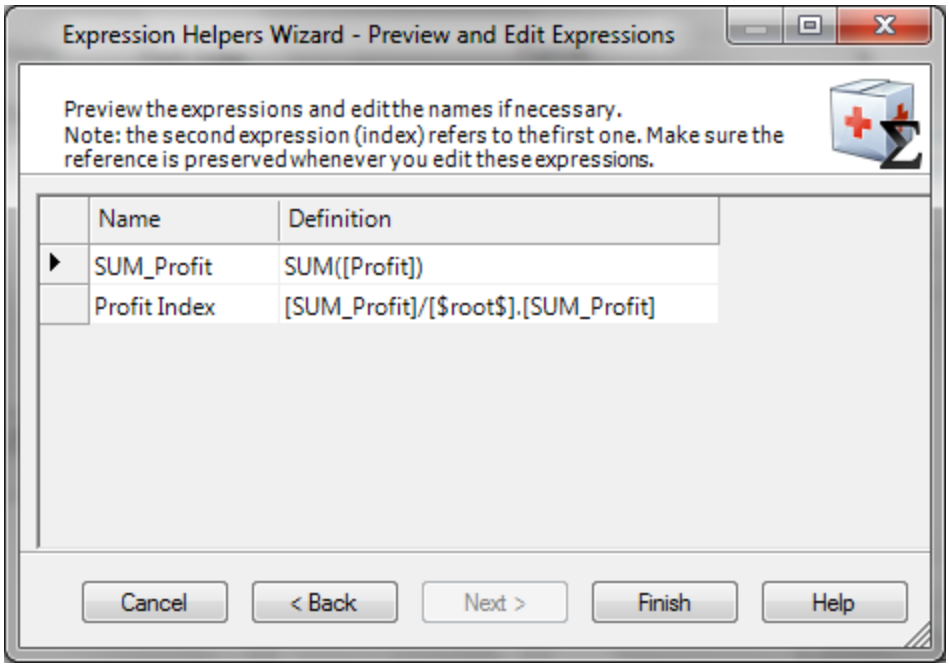
In this example, the Helper computes the ratio

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

(SUM(Profit) over the node) / (SUM (Profit) over the whole dataset)

The constructed expression is shown in the Preview window. Note that two expressions are created.

This example shows the portion of total profit represented by each node. The denominator of the Profit Index expression is the reference to the value of SUM_Profit at the root node.



Because the expression [$root$].[SUM_Profit] is used, if you copy the tree to a different dataset, the Index expression will be calculated with respect to the target dataset rather than the source. Press Finish to return to Expression Editor and edit the expression there if necessary.

In general, the expression syntax allows you to refer to the value of any expression at the root node. For example, you can manually create two expressions

A = SUM([Profit])

Profit_Index=[A]/[$root$].[A]

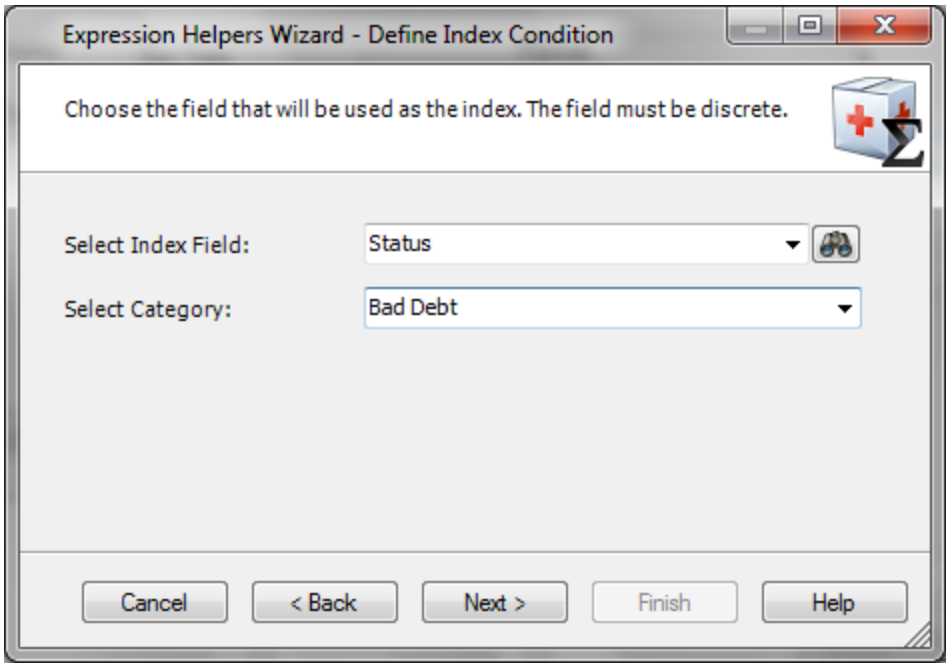
In some cases, this is a more preferable way to specify calculations because of its flexibility: you can reuse these calculations in other trees. If you copy them to another dataset they will still be valid as they will refer to the root node of the new tree.

See also: Expression Helpers: Overview

Discrete Distribution Helper

At the first step of the Discrete Distribution Helper Wizard, select a discrete field and the category whose frequency you want to be displayed in the nodes.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Note:

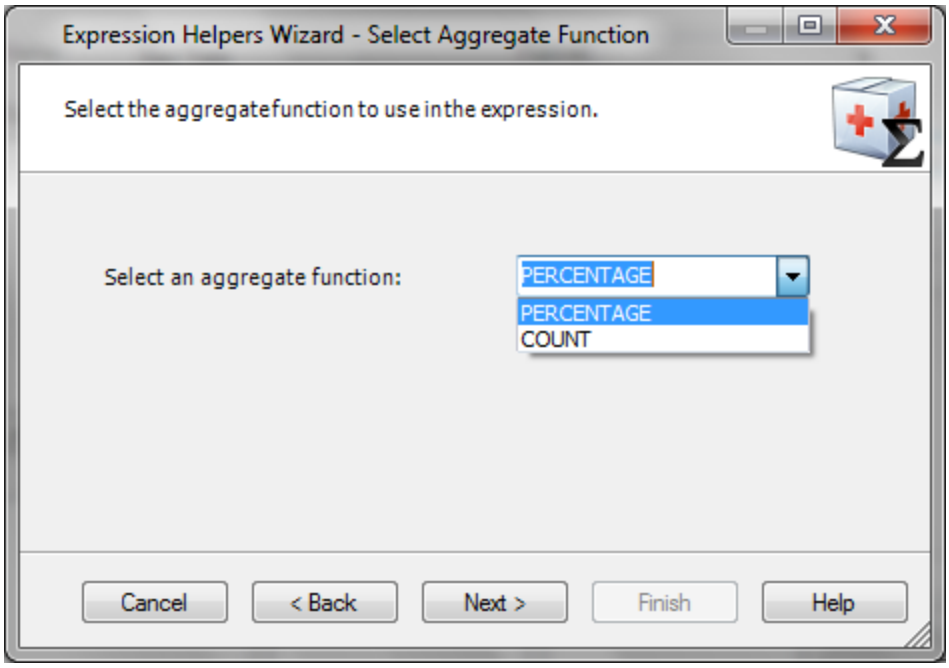
- all String and Boolean fields are discrete;

- in this helper, variables of the types Number, Integer, Time, or Timestamp whose cardinality is less than 50 are considered to be discrete.

Select whether you want to display the percentage or the count of the category selected at the previous step. The default option is PERCENTAGE.

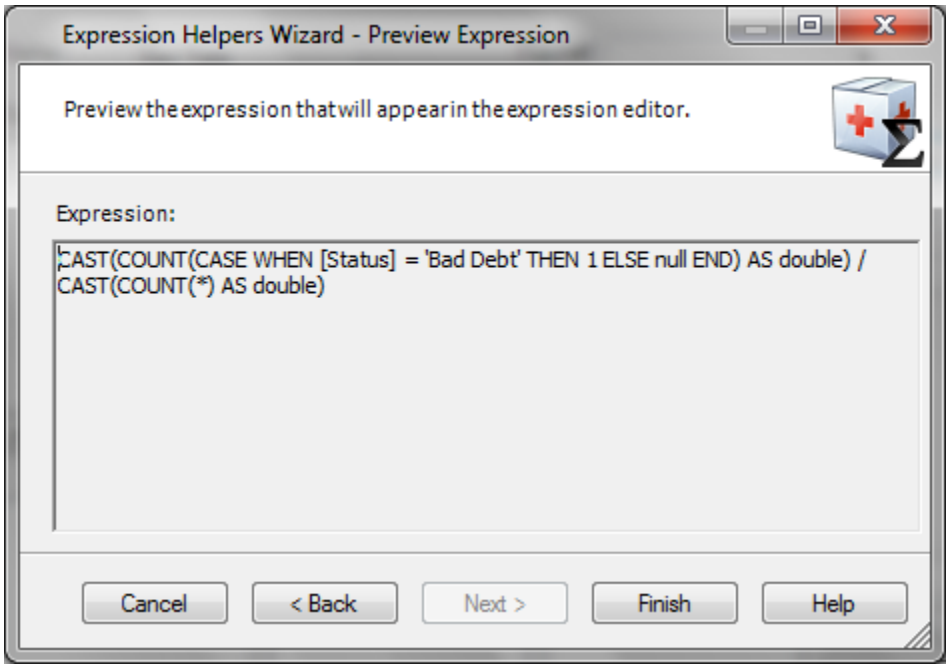
If the PERCENTAGE option is selected, the Format of the expression in the Expression Editor window is automatically set to **Percent**.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If PERCENTAGE is selected, then the result is as follows.

Note: The CAST ... AS DOUBLE operator takes care of the data type compatibility and conversion. It is inserted automatically whenever necessary.



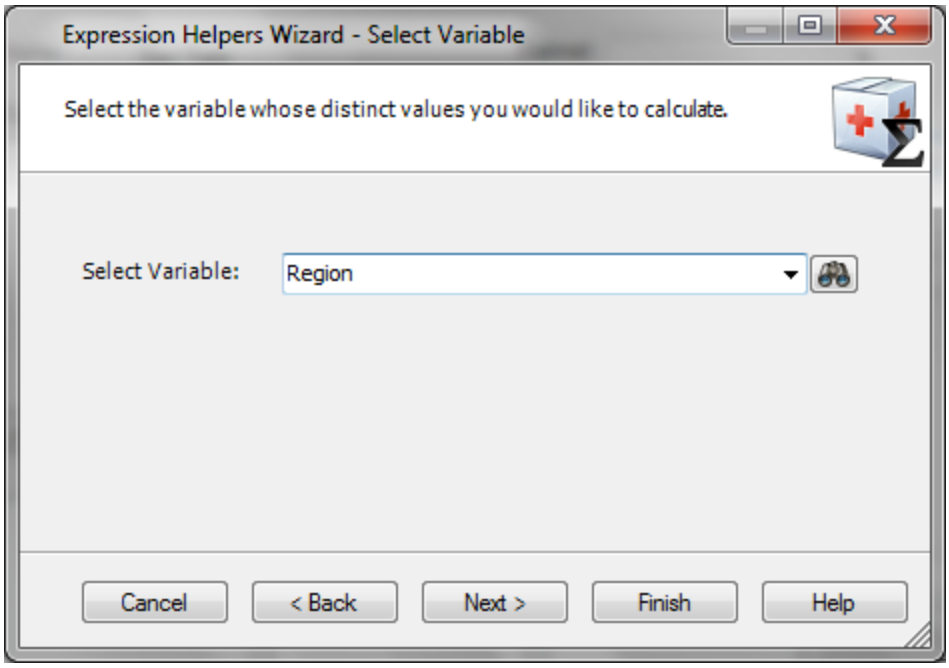
See also:

Expression Helpers: Overview

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Node Cardinality Helper

In the Node Cardinality helpers wizard, select a variable whose cardinality you want to be displayed in the tree nodes. This is the number of distinct values of the selected variable in a given node.



Click **Next** to preview the resulting expression. See the topic Preview Expression for details.

Click **Finish** to add it to the Expression Editor. For example, if the selected variable is "Region", the helper produces the expression **COUNT(DISTINCT [Region])**

Note that missing values are treated as a separate category when calculating the cardinality of a field.

See also: Expression Helpers: Overview.

Solution Constraint Helper

The Solution Constraint Helper is provided to assist you in defining constraints for the solutions to the optimization problem (see the topic Optimization in Strategy Trees for details).

The helper allows you to assign specific constant values to the nodes you selected for optimization. These constant values will serve as the upper or lower boundaries in the inequalities limiting the values of the solution in the corresponding nodes. You will use them later in the Optimization Helper.

Thus, the helper allows you to enter expressions with conditions on specific Node IDs of the form:

```
CASE
    WHEN [$node_id$] = <Node ID 1> THEN <constant 1>
    WHEN [$node_id$] = <Node ID 2> THEN <constant 2>
```

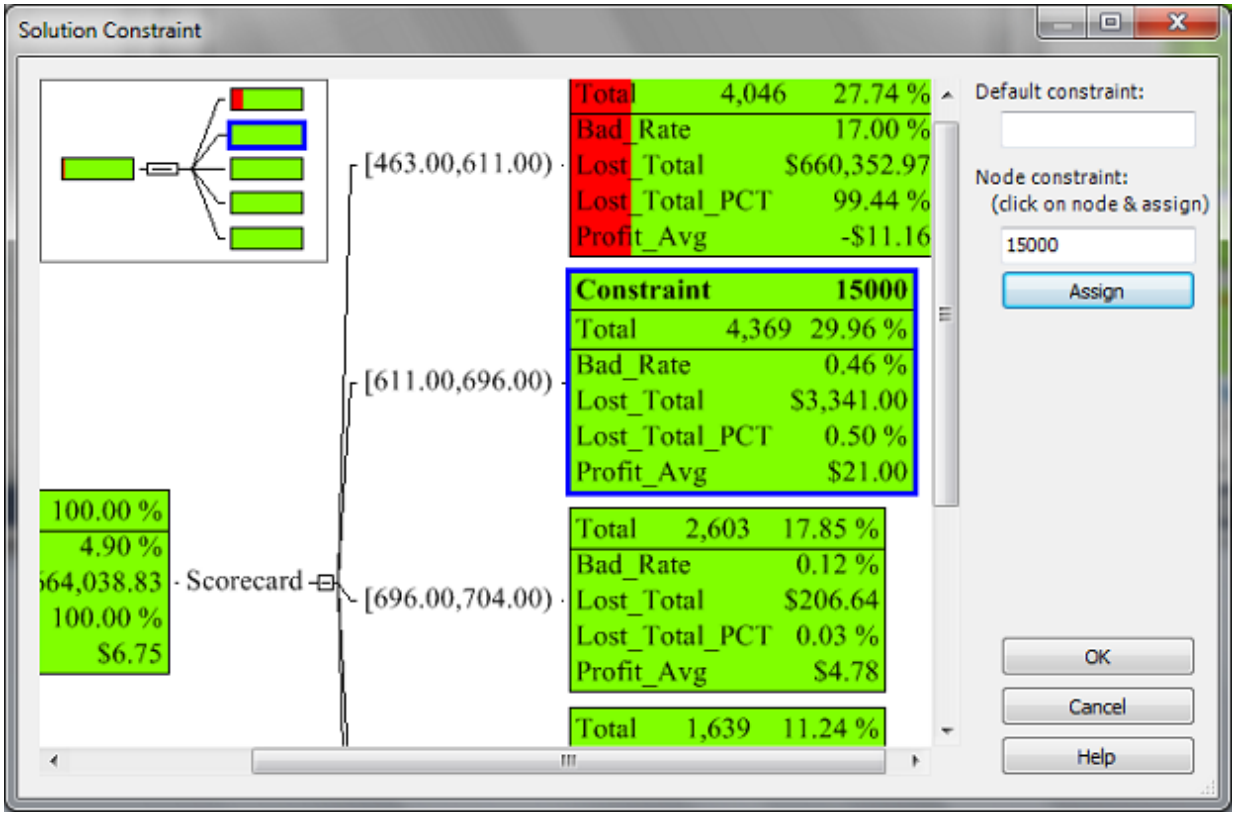
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
        ......
        ELSE <constant N>
END
```

You can specify:

- different constraints for specific nodes
- one default constraint for all other nodes (for the ELSE condition in the above expression)

When you click the Helpers button, select Solution Constraint as the Expression type, and click Next, the Solution Constraint dialog opens. It shows both the Strategy Tree and its map in the same window.



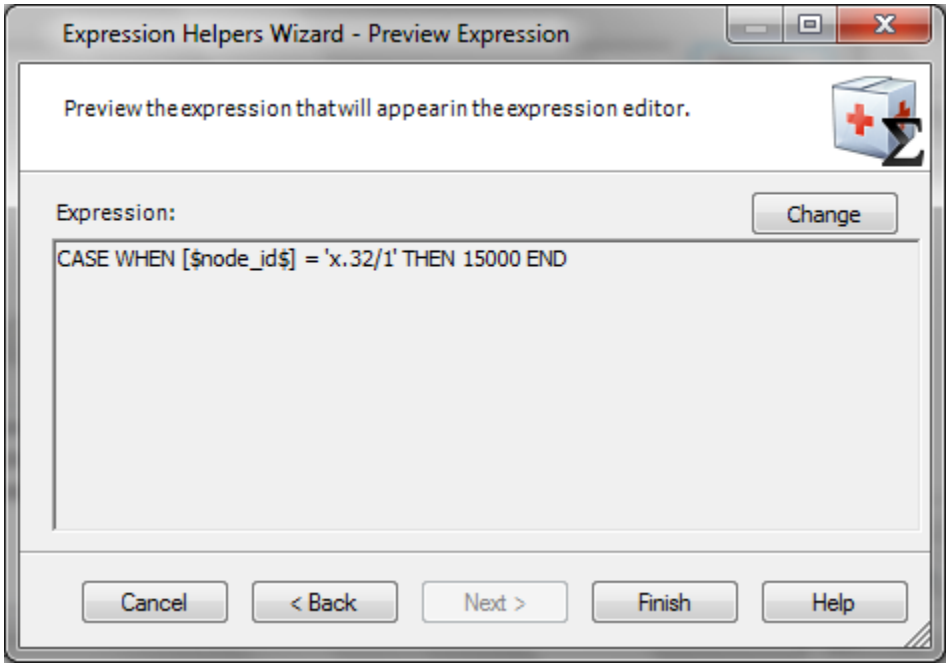
To assign a constraint to a specific node:

- click on the desired node in the tree (or in the tree map in the top left corner)
- enter the desired value in the Node constraint field and click Assign. If your constraint is the same for all nodes, you can enter its value in the Default constraint field instead, and click Assign. The assigned value will appear in the Constraint line at the top of the selected node.
- once you are finished defining different constraints for specific nodes, you can define a default constraint for all other nodes. Enter its value in the Default constraint field instead, and click Assign.

Click OK when done.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The next screen shows the resulting expression. For example:



Click **Change** if you want to go back to the Solution Constraint dialog and change anything. Otherwise, click **Finish**. The expression will be displayed in the Expression Editor, where you can manually edit it as you wish before adding it to the strategy tree. Specify the name of your new constraint expression in the Expression Name field. When done, click **OK** in the Expression Editor.

For details on the Optimization feature, see the topic Optimization in Strategy Trees.

See also: Expression Helpers: Overview

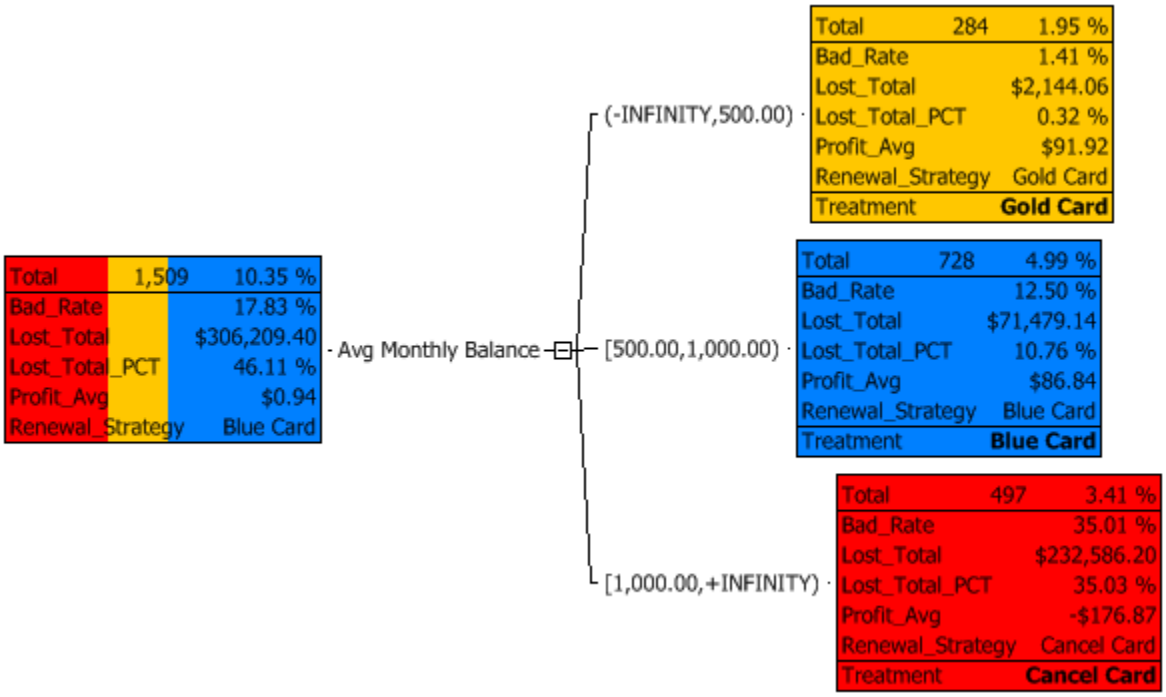
Treatments

A ***treatment*** is an action, output, or goal associated with a terminal node of a Strategy Tree as defined by the StrategyBUILDER user. Only one treatment can be associated with a given terminal node. You can create and assign new treatments dynamically or assign previously used treatments.

A single treatment may in fact mean a sequence of regular actions. For example, a treatment named "Repeated offers" could mean the sending of certain promotional materials to a certain group of customers at regular intervals.

A treatment is expressed in the form of an alphanumeric text string, which is assigned using a special command.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



StrategyBUILDER has the following capabilities in dealing with treatments:

- Color can be assigned to any treatment to improve the visualization and clarity. For example, you can assign blue color to the "Accept" treatment and red to "Reject".
- A monetary amount (cost or revenue) can be assigned to each treatment.
- Your strategy can be summarized based on treatments and view profiles and statistics of nodes with various treatments assigned. For example, treatments can be viewed and assigned not only on the main tree view, but also in Node Report and Profile Chart.
- A Treatments Report showing the list of treatments (including costs) can be generated.
- The command **Tools | Treatments from Calculation** allows you to convert the result of a calculation to a treatment and assign it.
- Treatments can be imported from and exported to an external file.
- When copying and pasting splits, you have an option of pasting treatments.

Note that splitting a node (e.g., using Find Split or Force Split) will remove assigned treatments from the current node (if any) since it will no longer be a terminal node after this operation.

See also: Managing Treatments

Managing Treatments

For general information about treatments, see the topic About Treatments.

To assign, remove, or edit a treatment, select a terminal node and click the Treatments icon  in the toolbar or select the Treatments command from the Tools menu:

From the menu:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

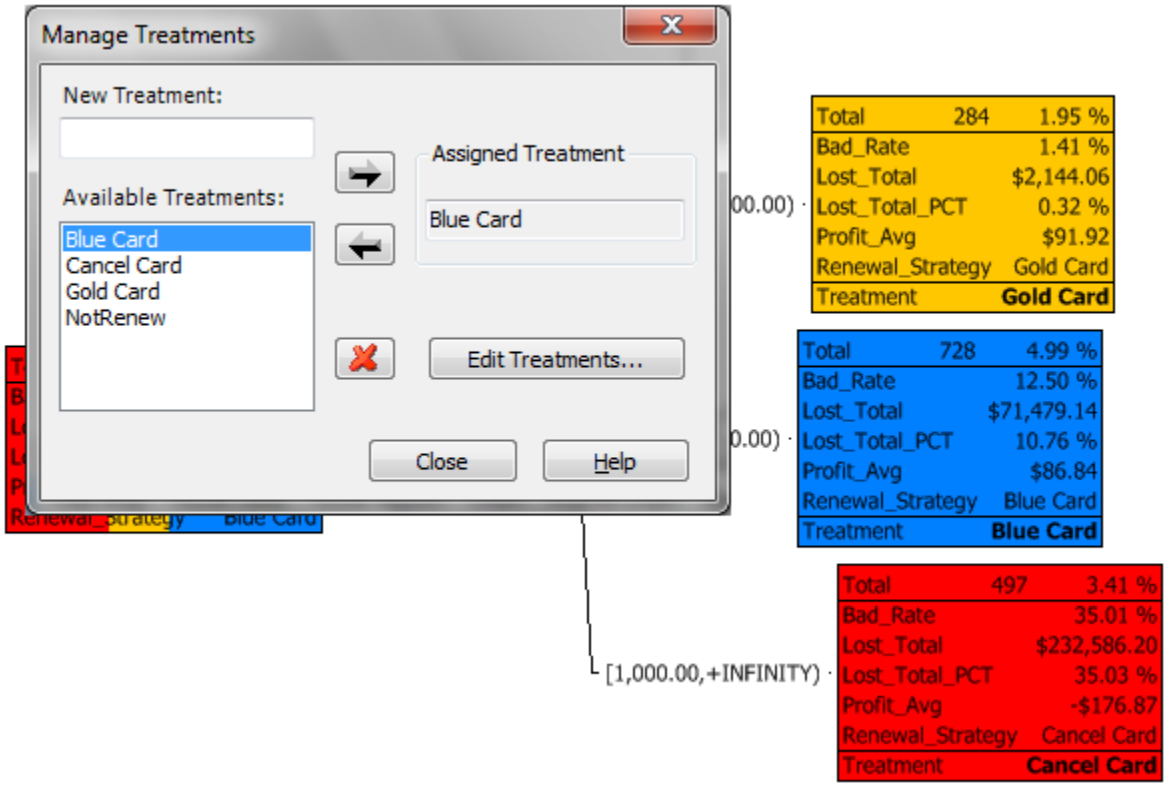
Tools | Treatments...

From the toolbar:

click the icon 

This will bring up the Manage Treatments dialog box. If the desired treatment already exists in the list of available treatments, select it and click the right arrow button to assign it to the selected node. Otherwise, enter the name of the new treatment and click the right arrow button.

In this example, the "Blue Card" treatment is created and assigned to the middle leaf:

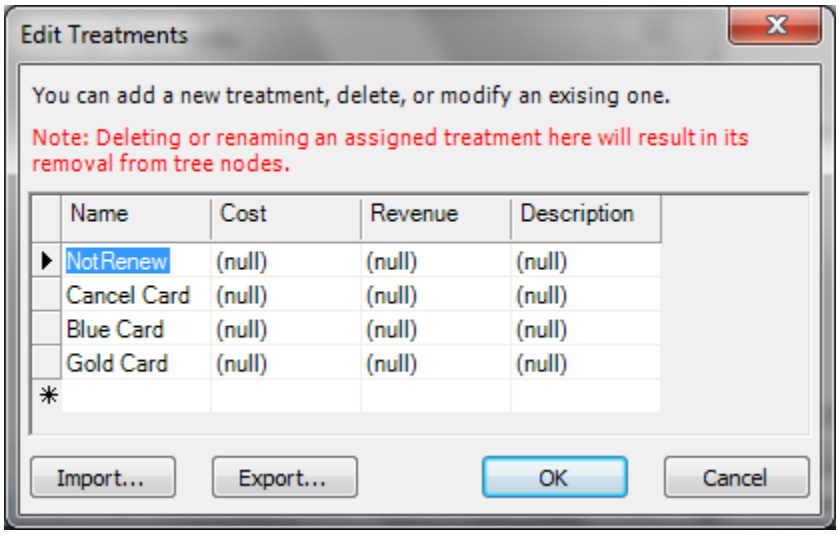


To **unassign** a treatment from a node, select this node first (its treatment is displayed in the Assigned Treatment box) and click the left arrow. Note that the treatment will remain in the Available Treatments list.

> *Note that splitting a node (e.g., using Find Split or Force Split) will remove the assigned treatment from the current node (if any) because it will no longer be a terminal node.*

To **edit** a treatment (rename, or assign cost and revenue, or enter a description), click the **Edit Treatments** button.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



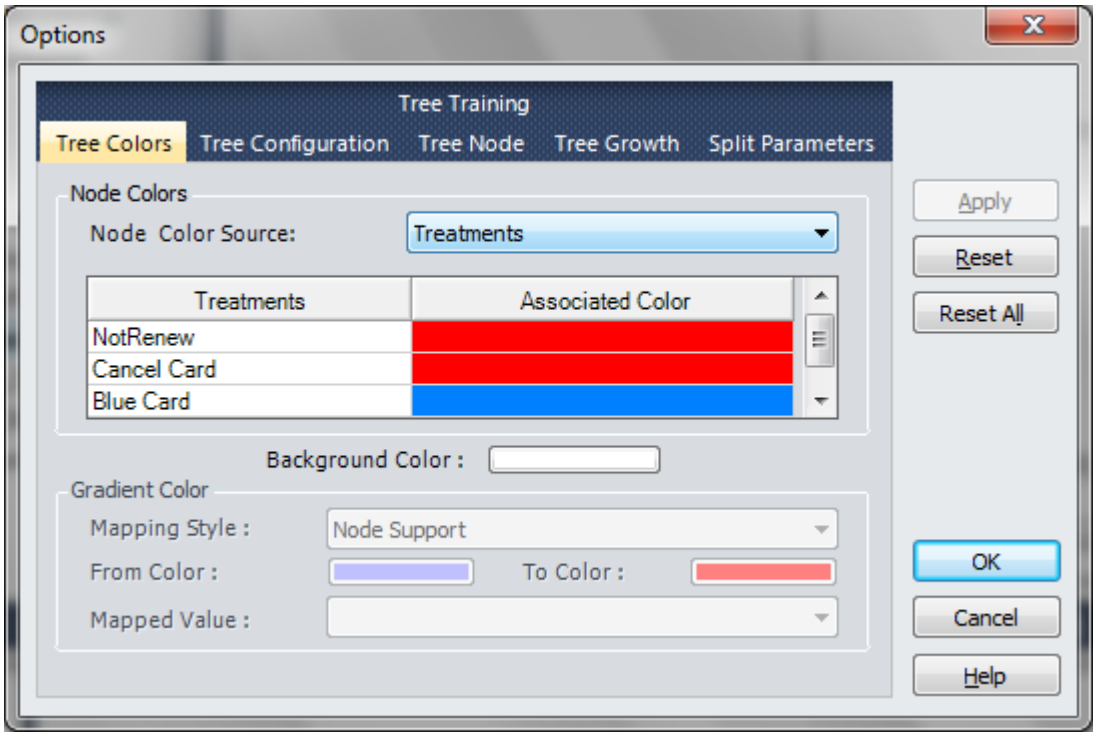
Note that renaming a treatment is equivalent to replacing it with a new one, so it will be removed from all nodes (if it is assigned to any).

To delete a treatment completely (i.e., to remove it from all nodes and the list of available treatments), click the Edit Treatment button, select the treatment and press the Delete key.

To delete all available treatments, click the button  .

You can assign a color to a treatment using the Options command. Select **Tools | Options** from the menu and click the Tree Colors tab in the Options dialog. Choose 'Treatments' as Node Color Source and assign the desired color to your treatment. The node that has this treatment will have the corresponding color.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also: About Treatments

Generating Treatments From Calculations

In some cases, your treatments may consist in assigning numerical values (for example, dollar amounts to each record in the selected node). If you already have those values calculated in the Strategy Tree nodes, you may want to convert them to treatments.

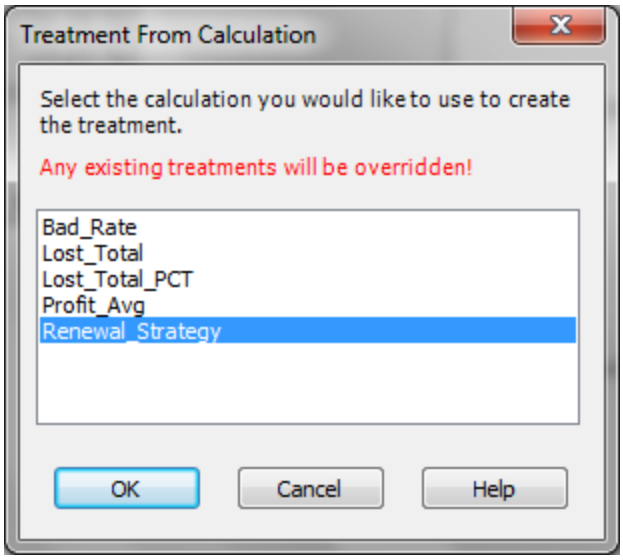
For this purpose,

- Click on any node. In this example, the "Blue Card" Renewal_Strategy is a string-valued calculation (which may have resulted, for example, from the Computed Conditional Treatment Helper). We need to convert it to an actual treatment.

Total	728	4.99 %
Bad_Rate		12.50 %
Lost_Total		$71,479.14
Lost_Total_PCT		10.76 %
Profit_Avg		$86.84
Renewal_Strategy		Blue Card

- select **Tools | Treatment From Calculation** from the menu

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- in the Treatment From Calculation dialog, select the calculation you would like to convert to a treatment and click **OK**.

Since calculations are displayed in all nodes of the tree, the corresponding treatment will be assigned to every terminal node.

As a result, your nodes will look like this (in this case, we generated a treatment from the *Renewal_Strategy* calculation, so our treatment coincides with the value of *Renewal_Strategy* at this node:

Total	728	4.99 %
Bad_Rate		12.50 %
Lost_Total		$71,479.14
Lost_Total_PCT		10.76 %
Profit_Avg		$86.84
Renewal_Strategy		Blue Card
Treatment		**Blue Card**

The treatment in this case is "Blue Card". You can rename the resulting treatment by editing it. To rename a treatment,

- click the Treatments button  in the toolbar, or select **Tools | Treatments** from the menu.

- click **Edit Treatments** in the Manage Treatments dialog

- rename the treatment you have just generated

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

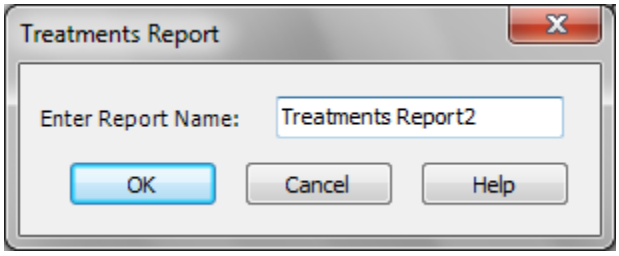
Treatment Report

The **Treatment Report** is a simple report that shows all assigned treatments in the Strategy Tree.

In addition, it also shows the following node information for any node with a treatment:

- the node size (the number and percentage of records)
- the values of all node calculations
- all the treatment attributes from the Edit Treatments dialog (cost and revenue, if defined)

To generate a treatment report, select **Tools | Treatment Report** from the menu.



Enter the desired name for the report and click OK.

An example of a treatment report:

Treatments Report - [Project3].[Credit_Strategy]

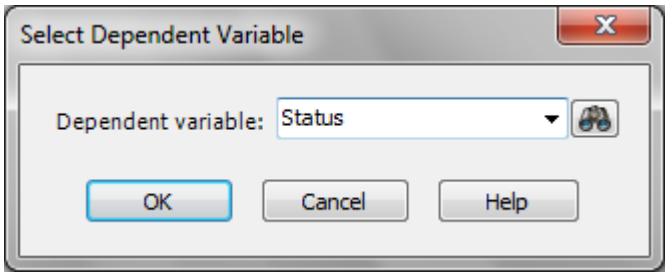
Treatment	Size(Count)	Size(Percent)	Description	Cost	Revenue	Amt_Lost_Total	AvailableLineTotal	AvgProfit	Bad_Rate
No increase	3,024.00	10.37%	N/A	N/A	N/A	$740,551.82	$2,453,935.00	-35.18	23.74 %
$2000 Line Increase	13,044.00	44.72%	N/A	N/A	N/A	$15,231.42	$13,096,712.00	11.44	0.66 %
$5000 Line Increase	1,510.00	5.18%	N/A	N/A	N/A	$44,268.90	$1,034,850.00	90.07	5.30 %
$3000 Line increase	5,199.00	17.82%	N/A	N/A	N/A	$4,764.58	$4,876,954.00	20.97	0.37 %

Creating and Modifying Splits

Find Split

The **Find Split** command prompts you to choose a dependent variable

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



and then searches for the independent variables (IV) that are most significant to this dependent variable (DV) based on a criteria defined by the measure. The Independent variables that are found to be significant are called the split variable(s). The most significant split variable is displayed as child nodes where each child node represents a category or range found in the independent variable. The Find Split algorithm in Strategy Trees is the same as in Decision Trees. See the Find Split topic in the Decision Tree section for details.

The **Find Split** command may find multiple significant splits. The splits are ranked according to the measure and the first and most significant split is displayed in the tree. The other splits can be viewed using Next Split or Go To Split

Strategy trees do not require dependent variables. However, the results of the **Find Split** command will produce a split that contains the distribution of the dependent variable.



To perform a **Find Split**:

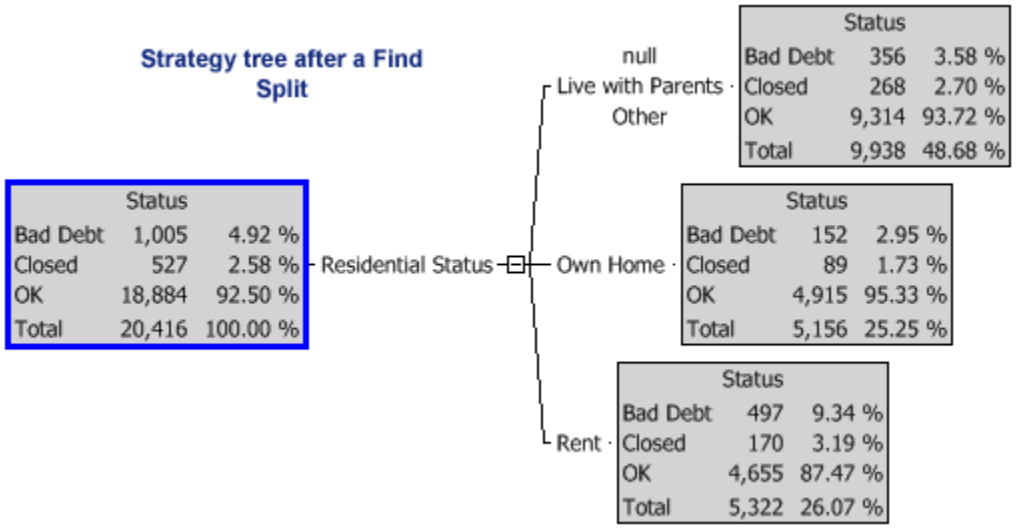
From the menu:

Grow | Find Split

From the popup menu:

right-click on the parent node and select Find Split

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Force Split

Unlike **Find Split**, **Force Split** allows you to choose the variable that you want as your split variable. Like **Find Split**, the node to split must be active:



From the menu:

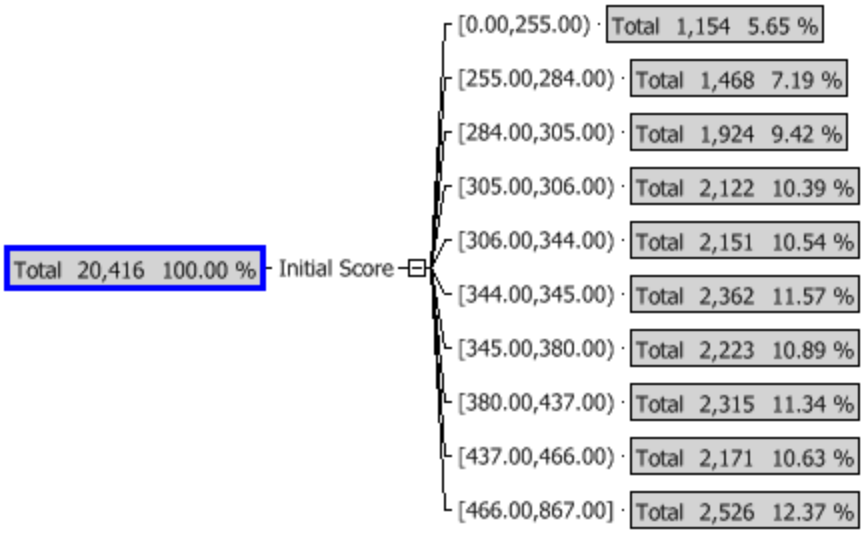
Grow | Force Split...

From the popup menu:

on the parent node, right-click | Force Split...

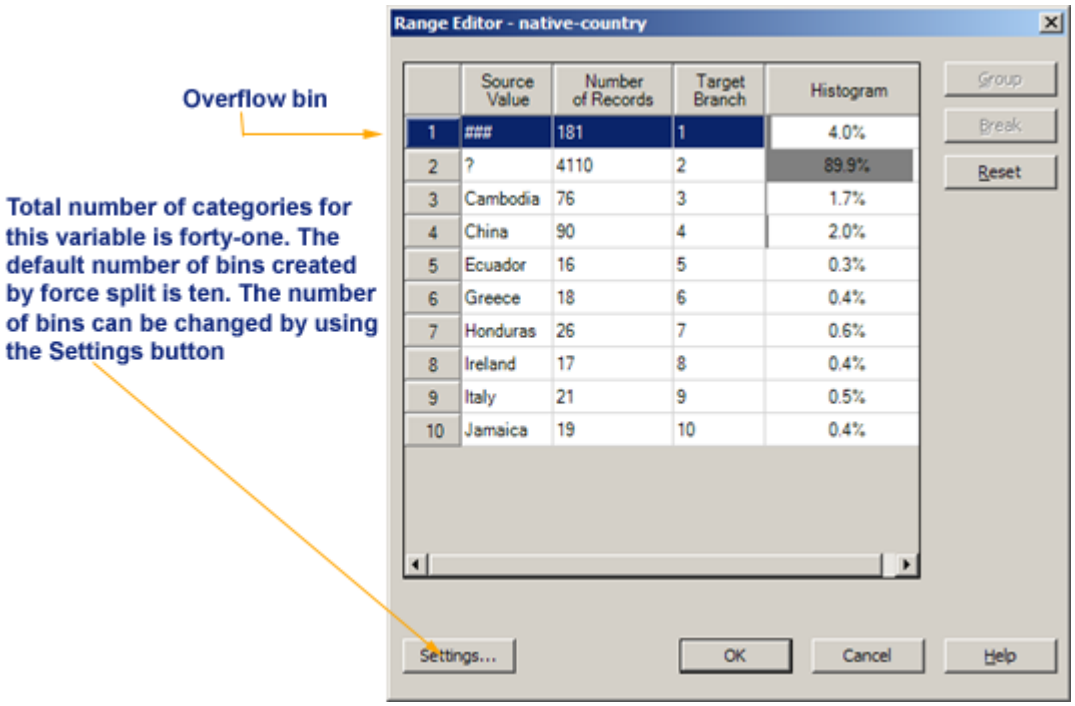
A series of steps takes you through the **Force Split** process. This includes choosing the split variable and defining the binning of the variable as it will appear in the tree. The resulting split of the above example is:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



About Force Split

- The **Force Split** command does not use the settings in the attribute editor.
- If the cardinality of the split variable is greater then ten (>10), then by default the split variable is divided into ten bins, nine of the bins will be categories (or ranges) from the variable and the tenth bin will be the overflow bin, represented as ###.
- Example



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- A high cardinality split variable will be limited to one hundred bins/categories. There will be ninety-nine categories (or ranges) and the hundredth (100th) bin will be the overflow bin, represented as ###.
- When performing a force split on a node that contains existing splits, all the splits will be replaced by the force split command. This is the same as performing an Erase All before doing a force split.

Next / Previous / Go To Split / Copy & Paste Splits

Next split displays the next significant split in the list of splits created by the Find Split command. This command is only enabled if there is more than one split created by the **Find Split** command.

From the menu:

Reshape | Next Split

From the popup menu:

Right click on parent node | Next split

Previous split displays the previous significant split in the list of splits created by the Find Split command. This command is only enabled if there is more than one split created by the **Find Split** command.

From the menu:

Reshape | Previous Split

From the popup menu:

Right click on parent node | Previous split

The **Go To Split** command allows you to go to a different split on the parent node. If there are no other significant splits calculated, this item will be disabled.

From the menu:

Grow | Go To Split

From the popup menu:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

on the parent node, right-click | Go To Split

The **Go To Split** dialog will appear and display a list of all the significant splits calculated for the given node in rank order. The rank corresponds to the **Info** column in the split report.

The **Copy/Paste Splits** commands allow you to copy the splits from one tree and paste them to another. This feature can be useful in a variety of situations.

- See how the split in one node behaves when copied to another node of the same tree
- See how the split in one tree behaves when copied to the node of a different tree created in a different dataset.

Copy Splits

The **Copy Splits** command copies into memory all the splits of the tree, starting from the active node.

Highlight the node you want to copy

From the menu:

Edit | Copy Split

From the popup menu:

Right click on parent node | Copy Split

Paste Splits

The **Paste Splits** command pastes splits that were copied to the clipboard with the **Copy Splits** command. The tree structure from the original Copy Split is maintained.

Highlight the node where you want to paste the split.

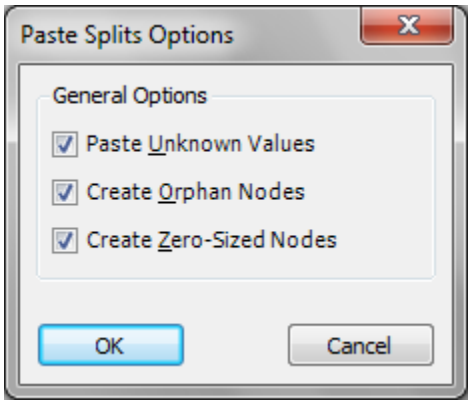
From the menu:

Edit | Paste Split

From the popup menu:

Right-click on the parent node
Choose Paste Split

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Choose the desired Paste Split Options
Click OK to Paste the splits

Paste Split Options

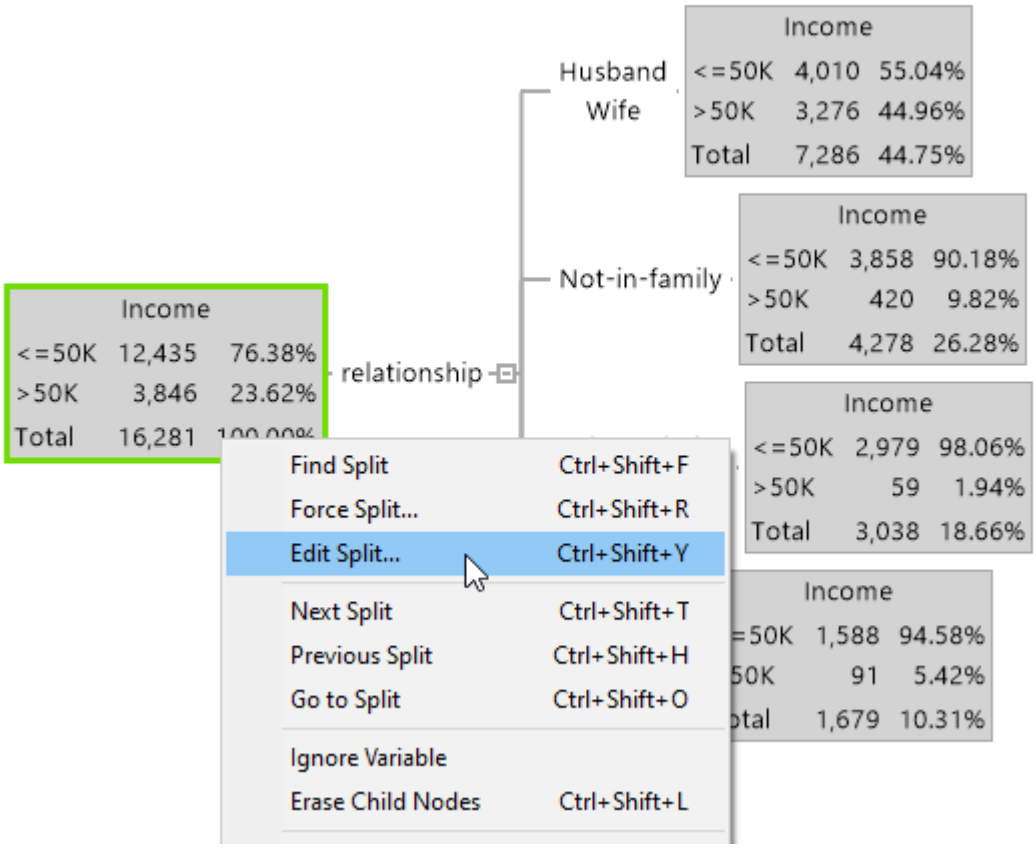
The meaning of Paste Split Options in Strategy Trees is the same as in Decision Trees. For details, see the Copy Splits / Paste Splits topic in the Decision Tree section.

Edit Split

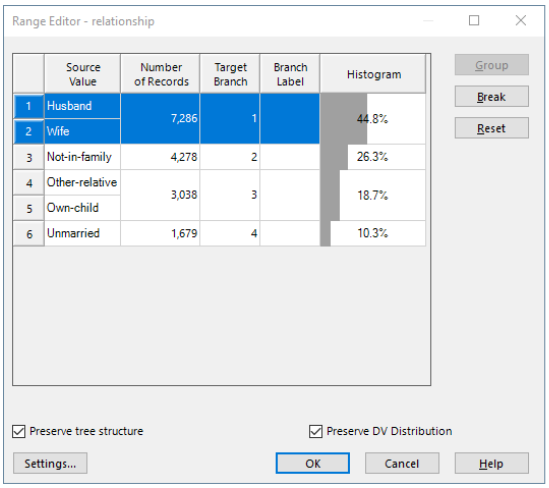
The **Edit Split** command allows you to modify the split on the currently selected node by grouping or splitting the child nodes using the Range Editor.

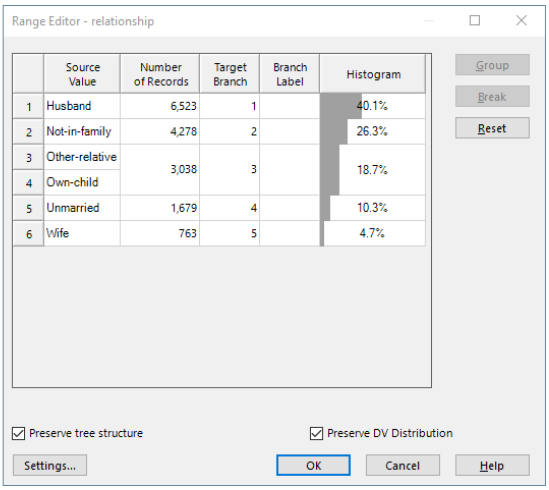
To edit a split, righ-click on its parent node and select **Edit Split** from the context menu. Alternatively, click on the parent node to select it and press **Ctrl + Shift + Y** on your keyboard, or select **Edit Split** from the **Grow** menu.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The **Range Editor** displays the bins of the split variable corresponding to the child nodes of the split. You can modify the binning the same way as in a decision tree. For example, to separate the categories ***Husband*** and ***Wife***, select the top two (grouped) rows and click the **Break** button.





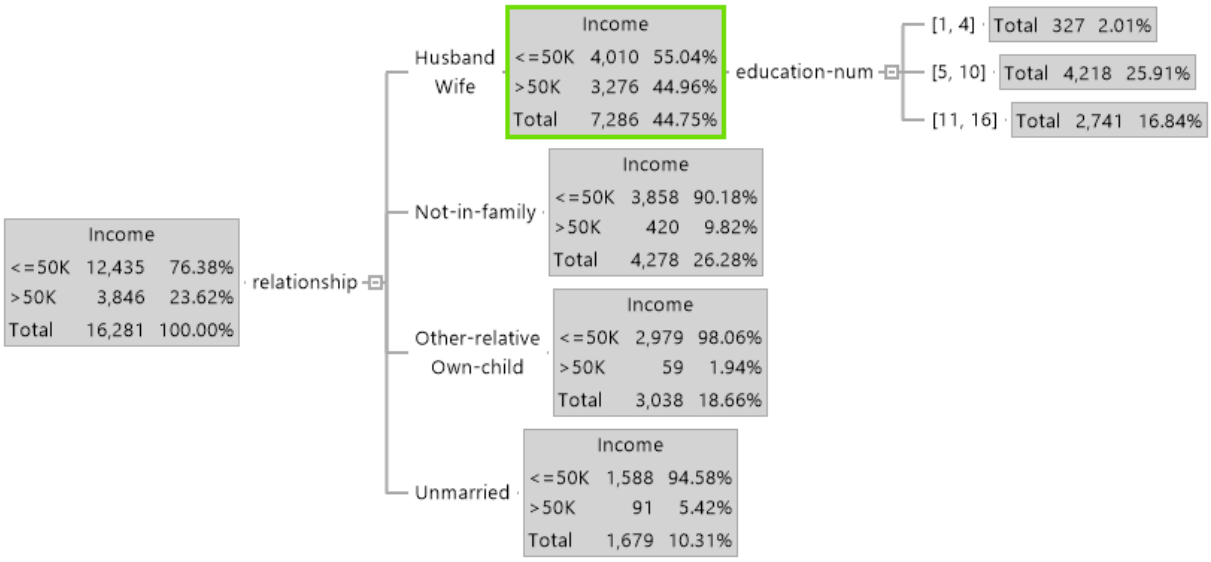
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If you edit a split for a node with grandchildren, you have the option of preserving the branches unaffected by the Edit Split whenever possible. By default, the splits in the unaffected branches will be preserved (the **Preserve tree structure** checkbox is selected).

Editing splits for discrete and continuous variables in a Strategy Tree is the same as in Decision Trees. See the topics Range Editor (Discrete) and Range Editor (Continuous).

Note: In the case of a continuous independent variable, real numbers that represent interval boundaries in Range Editor are displayed with fixed precision (2 digits after the decimal point) if the cells are not in Edit mode. To enter the Edit mode, click on a cell or press the F2 key. This will display the actual value with the original precision.

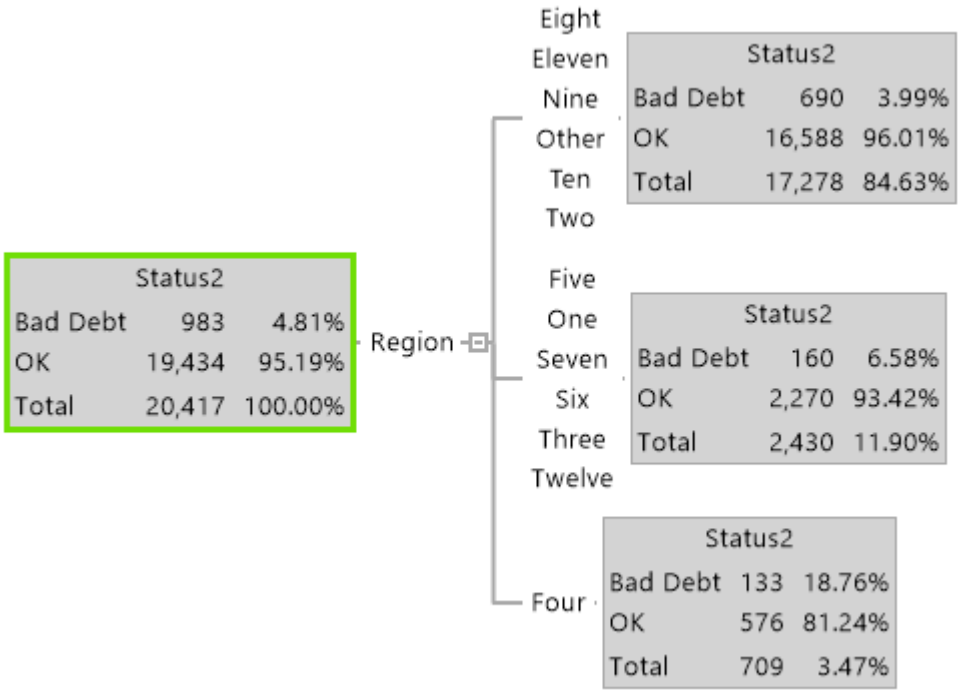
The option "**Preserve DV Distribution**" is selected by default and applies to both the selected node and its child nodes if they have a DV distribution displayed. This option is useful when editing splits produced by the Find Split operation. If you deselect this option, the DV distribution in the child nodes and their sub-branches (if any) will be erased. The DV distribution in the selected node will also be erased unless it is inherited from its parent node. In the example below, the split on the *education-num* variable was edited with the "Preserve DV Distribution" option turned off. The leaves in this split showed the Income distribution before editing the split. The Income distribution in its parent node, however, is inherited from the root, and is therefore preserved in this case.



Branch Labels in Splits

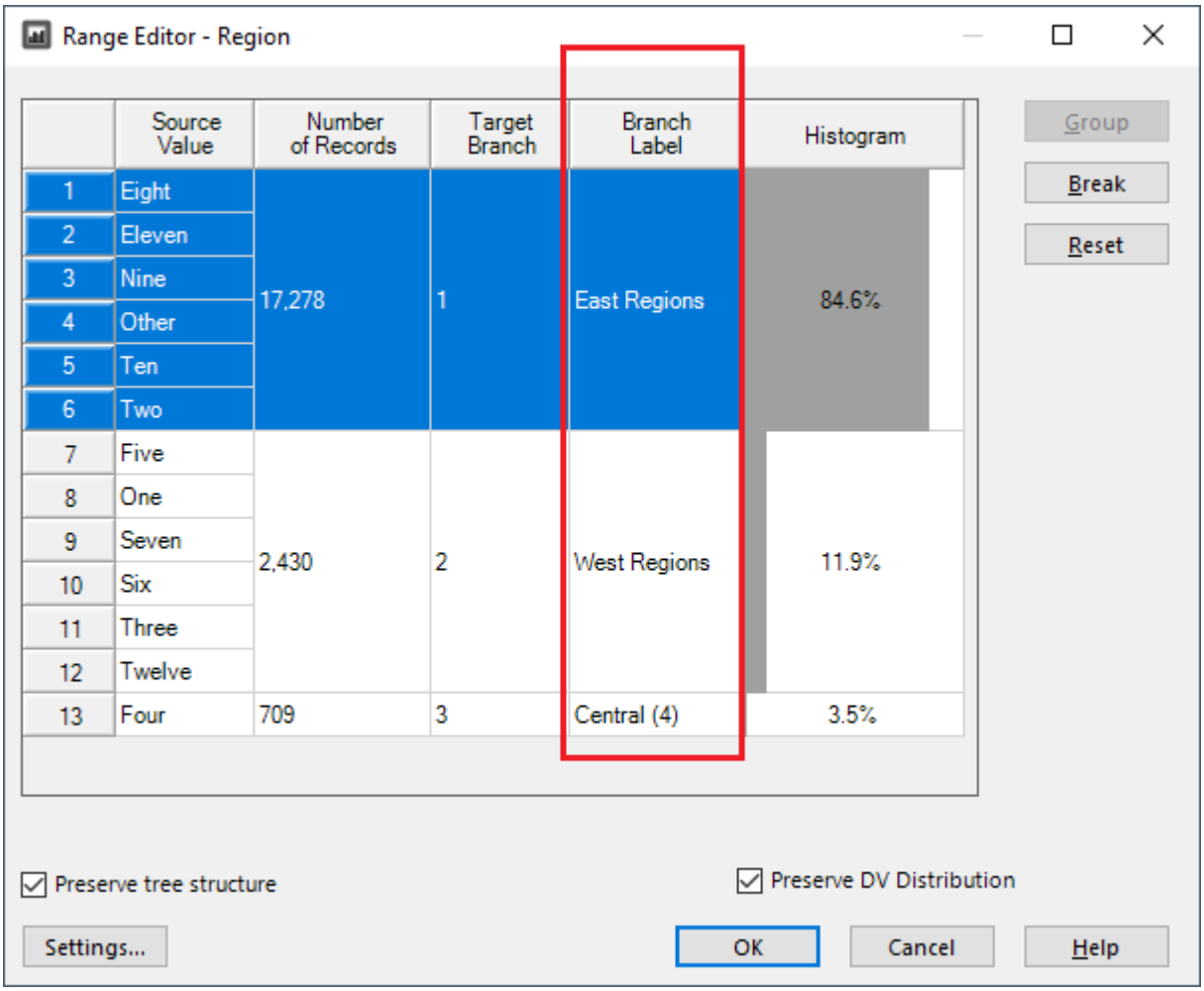
When editing or forcing a split, after grouping a large number of discrete values of a nominal variable, you may want to label such groups and display the labels instead of stacked values that sometimes take too much space in the tree view. In the following example of a split on region codes, the two child nodes on the left have marginal rules with stacked values that take too much space:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



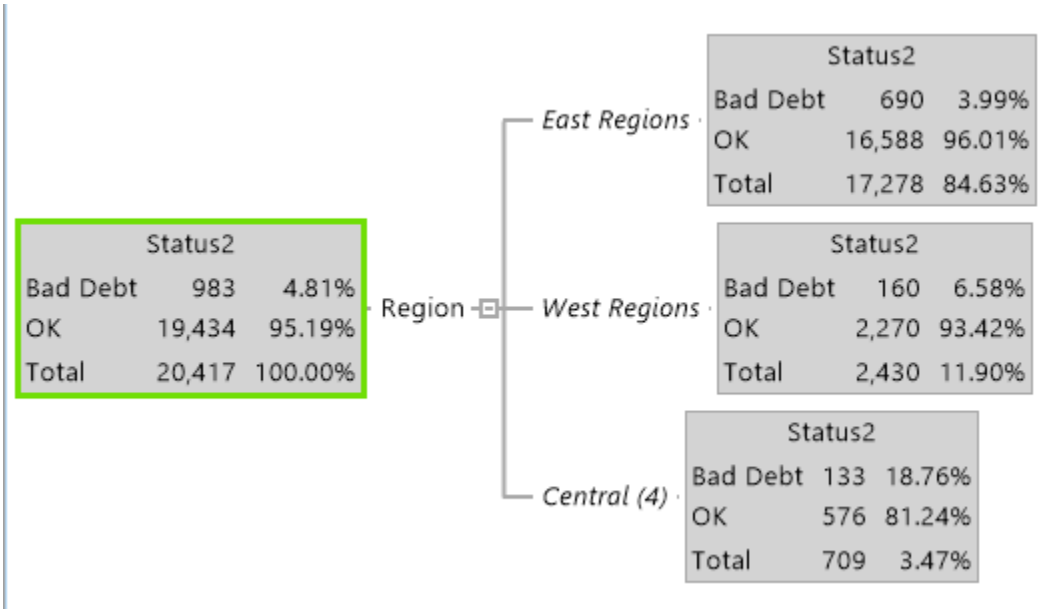
Using the Edit Split command, you can label these groups in the Range Editor dialog by entering the desired labels in the **Branch Label** column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



As a result, the split looks much more compact. The labels are dispayed in *cursive font* to distinguish them from regular individual values of the variable.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Erase Child Nodes / Ignore Variable

The **Erase Child Nodes** command deletes all nodes beneath the current node. First, select a node whose child nodes you want to delete.

From the main menu:

Grow | Erase Child Nodes

OR

Right click on the node and select Erase Child Nodes

The **Ignore Variable** command removes the variable of the current split from the analysis. The same effect can be accomplished by ignoring the variable in the Attribute Editor.

From the menu:

Grow | Ignore Variable

From the popup menu:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Right click on parent node | Ignore Variable

Modifying Tree Attributes

Using the Attribute Editor

The Attribute Editor allows you to modify variable attributes for the tree. The attribute values can be set and modified when inserting the tree in the tree wizard or modified after the tree has been inserted. Some of the tree parameters determined by the values set in the Attribute Editor are:

- whether a variable is included in the model
- the role of variables in the model (dependent, independent, or weight)
- the maximum number of nodes in splits produced by Find Split and AutoGrow
- the maximum number of bins or intervals in the Force Split Range Editor
- whether the values of a variable are treated as ordered or not
- whether grouped values of numeric variables are displayed in intervals or as discrete values
- the way missing values are treated

While a decision tree is open, the Attribute Editor can be invoked as follows:

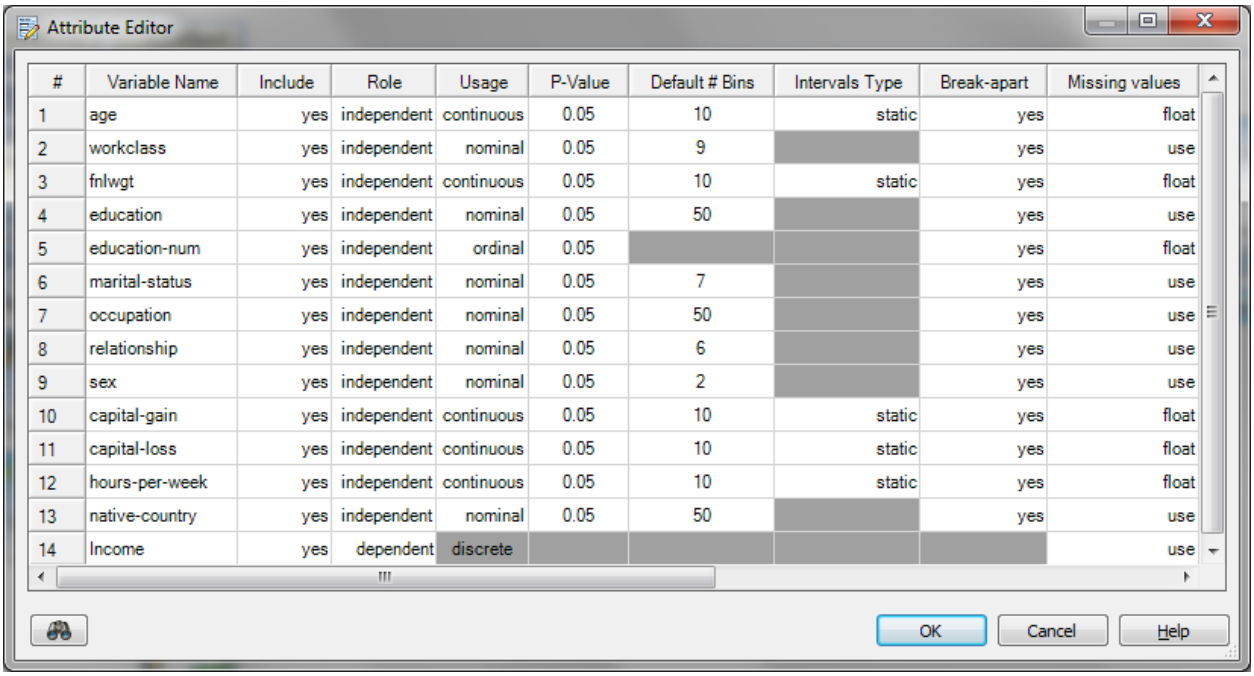
From the menu:

Tools | Attribute Editor...

From the background context menu:

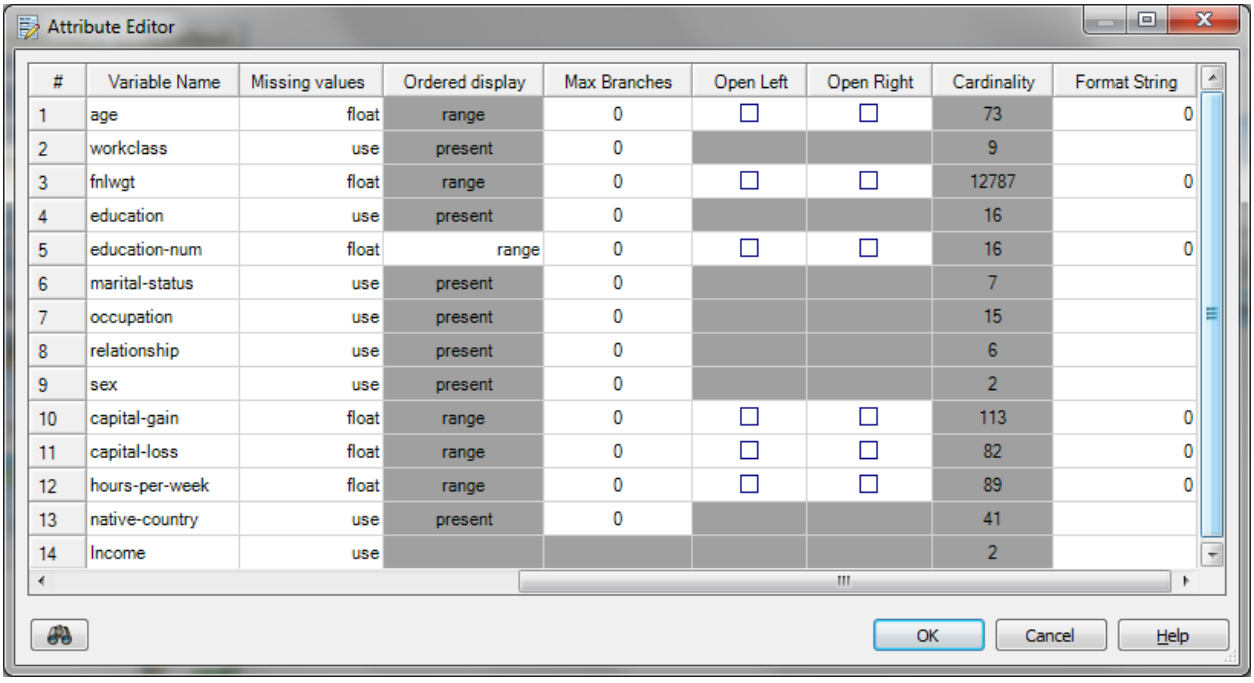
Right-click anywhere on the tree background and select Attribute Editor...

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



#	Variable Name	Include	Role	Usage	P-Value	Default # Bins	Intervals Type	Break-apart	Missing values
1	age	yes	independent	continuous	0.05	10	static	yes	float
2	workclass	yes	independent	nominal	0.05	9		yes	use
3	fnlwgt	yes	independent	continuous	0.05	10	static	yes	float
4	education	yes	independent	nominal	0.05	50		yes	use
5	education-num	yes	independent	ordinal	0.05			yes	float
6	marital-status	yes	independent	nominal	0.05	7		yes	use
7	occupation	yes	independent	nominal	0.05	50		yes	use
8	relationship	yes	independent	nominal	0.05	6		yes	use
9	sex	yes	independent	nominal	0.05	2		yes	use
10	capital-gain	yes	independent	continuous	0.05	10	static	yes	float
11	capital-loss	yes	independent	continuous	0.05	10	static	yes	float
12	hours-per-week	yes	independent	continuous	0.05	10	static	yes	float
13	native-country	yes	independent	nominal	0.05	50		yes	use
14	Income	yes	dependent	discrete					use

Scroll to the right to see more attributes:



#	Variable Name	Missing values	Ordered display	Max Branches	Open Left	Open Right	Cardinality	Format String
1	age	float	range	0	☐	☐	73	0
2	workclass	use	present	0			9	
3	fnlwgt	float	range	0	☐	☐	12787	0
4	education	use	present	0			16	
5	education-num	float	range	0	☐	☐	16	0
6	marital-status	use	present	0			7	
7	occupation	use	present	0			15	
8	relationship	use	present	0			6	
9	sex	use	present	0			2	
10	capital-gain	float	range	0	☐	☐	113	0
11	capital-loss	float	range	0	☐	☐	82	0
12	hours-per-week	float	range	0	☐	☐	89	0
13	native-country	use	present	0			41	
14	Income	use					2	

The global defaults for the tree attributes can be set in the Tree Attributes tab of the Preferences dialog accessible from the Tools menu. See the topic Preferences: Tree Attributes for details.

Tip: you can change attribute values for more than one variable at a time. To this end, click in the attribute column of your choice and drag your mouse down while holding down the left mouse button to highlight several cells in the column. (Alternatively, while holding down the SHIFT or CTRL key on your keyboard, click on several cells whose values you want to change in the selected column.)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

When finished selecting multiple cells, set the value for one of them (the last one you highlighted). This will set the same value for all highlighted cells in the selected column.

You can also click the Field Finder (binoculars) button to quickly find and select the desired variable and use the search capability.

Tree attributes can be copied from one tree to another. Open the tree whose attributes you want to copy and open its Attribute Editor. Click the Copy button and close the Attribute Editor. Then open the other tree, open its Attribute Editor and click the Paste button. All attributes for the variables that are common for both trees will be copied. Click OK to apply the changes and close the Attribute Editor.

NOTE: After you apply changes made in Attribute Editor and return to the tree view, you can still undo them using the Undo command, provided that you haven't made changes to a different tree or dataset.

Tree Attributes

Variable Name	Name of the variable in the decision tree.
Include	Determines if a variable is to be included in the analysis.
Role	Determines how a variable is used in an analysis. A number of options are available: **Dependent**: Indicates that this variable is the dependent variable, or the variable that you want to explore. There can only be one dependent variable (DV). **Independent**: Indicates that this variable should be compared to the dependent variable to determine if it has predictive value. **Weight:** Indicates that this variable should be used as a weight that controls how much emphasis each record should be given in the analysis. If this option is selected, you must also select the type of weighting in the Weight Type column. Example
Usage	Algorithms can treat values in a number of ways. In general, if the values of a field are ranked (like 1, 2, 3, or 1-low, 2- medium and 3-high) then they are 'ordinal'. If there is no ordering, i.e., if they are just categories or qualities, they are considered to be 'nominal' values (red, blue and green). **No binning**: Each category within the variable will form its own separate node in the tree. The search for similar groupings is turned off. **Ordinal:** Treats values as an ordered set (e.g., 1-low, 2-medium, 3-high or 1, 2, 3). Only adjoining values will be grouped together when an algorithm is looking for similarities. For example, 1-low can be paired with 2-medium, or 2-medium can be paired with 3-high but 1-low cannot be paired with 3-high. **Nominal**: Values combine freely. For example, the category 'red' in a String variable may combine with 'blue' or 'green'. All possible groupings of field categories are examined so that similar items are grouped together to become a node in the tree. **Continuous**:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

a) For *independent* variables: The variable is binned into intervals, which can be calculated in a number of different ways, based on your selections in the Intervals Type column.

b) For numeric *dependent* variables: The numeric DV is considered as a continuous quantity, and instead of its individual values the tree nodes show the average and standard deviation of its values.

Discrete: Available only for the dependent variable. This option allows you to treat a numeric dependent variable as discrete: Show all its values as discrete categories instead of treating it as continuous (showing Average and Standard Deviation in the nodes).

Example

p-value Defines the level of confidence required to merge values into groupings. This setting affects the bushiness of the decision tree - a low setting (0.001) will tend to produce few branches (splits) from each node, whereas a high setting (0.20) will produce a bushy tree with more branches. The default setting is 0.05.With a p-value setting of 0.05 you would typically only be wrong considering values as category components 5 times out of 100. Similarly, a setting of 0.001 will assure you that you have a 99.9% accuracy in the category assignments but that you will also have fewer resulting categories. A setting of 0.20 would result in more branches but with less certainty of the accuracy (80% confidence).

Default # of Bins - If ***Usage*** is `continuous`, the value in this column determines the number of intervals for the selected variable according to the following rule: To form splits in a decision tree, continuous values are initially divided into discrete intervals where each interval has approximately the same number of observations. These discrete intervals are then treated as ordered and combined accordingly. The number specified here determines how many **initial** intervals are formed for this continuous variable prior to the ordered grouping. Therefore, the resulting number of nodes may be less than or equal to the value in the Interval column. For example, if Interval=8, the resulting number nodes can be less than or equal to (<=) 8.

This value also determines the default nimber of intervals shown in the Range Editor of Force Split.

For more details, see the topic "Attribute Editor: Default Number of Bins".

- If ***Usage*** is `nominal`, the value in this column determines the maximum number of categories (bins) to display for the selected variable. The default value is equal to the cardinality or 50 (whichever is less). You may enter any integer number between 1 and the cardinality of the variable. If the maximum number of bins **N** is equal to the cardinality of the variable, then each of its values is represented as a separate category. If **N** is less than the cardinality, the following mapping is applied: the top **N-1** values ordered by frequency remain as they are, while the rest are mapped into one category denoted by triple number sign ("###").

The default maximum number of bins for nominal variables can be changed in the *Tree Attributes* tab of the **Preferences** dialog accessible from the **Tools** menu.

Intervals Type This option affects the Find Split and Automatic Grow operations and applies only to the variables with *continuous* Usage (see above). It defines the level at which the binning maps for continuous independent variables are calculated.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Static: The binning map for the variable is precomputed at the root node. The intervals remain static throughout the tree.

Dynamic: The binning map for the variable is calculated dynamically at every node.

Although the Static type is the default, the Dynamic type provides more intelligent mapping and is therefore recommended in most cases. Note that splitting a variable with dynamic interval type is up to two times slower than with the static type, but this may not be noticeable on fast computers.

Break-apart Allows you to optimize the growing of the tree by choosing either speed or accuracy. There are two options:

Yes: this setting gives the most accurate results. **Yes** is the default.

No: this setting is used when you have variables with many categories and you want to speed up the analysis. However, the results may not be optimal.

Missing Values Allows you to specify how missing values in your data are handled in the tree. The effect and applicability of the missing value options depend on the variable *role* (independent, dependent, or weight) and *usage* (see *Usage* above).

1. For the dependent variable (DV):

Use (applicable to discrete DV only): Missing values are treated like any other value of the DV. More precisely, a missing value will be considered a separate DV category designated as ***null*** and will be shown in every node together with other DV values. This option is not applicable to continuous DV.

***Ignore*:** All records containing missing values in the dependent variable will be excluded from the analysis. Note that in this case the root node will contain fewer records than the parent dataset. If the DV of a Decision Tree is continuous, the records with missing values in the DV are always excluded, so *Ignore* is the default and the only available option for continuous DV in Decision Trees. This does not apply to Strategy Trees.

Note that if you ignore missing values in the DV, it will affect all tree views except for the Data tab. which will still show records with missing values in the DV. (The Data tab always shows the actual raw data for any selected node, which is determined by the node rule only, not by the settings in Attribute Editor).

2. For the independent variables (IVs):

***Use*:**

a) For variables whose *Usage* is "nominal" or "no binning", the ***use*** option is the default. The missing value is considered one of the regular categories of the variable and will be used just like any other discrete category when calculating the split. Missing values are displayed as ***null*** in the tree view, the Range Editor of Force Split and Edit Split, and other views and dialogs.

During the iterative process of the Find Split algorithm, the missing value (*null*) category bin can be grouped with any other category bin starting from the initial iteration when the algorithm starts to merge the bins in the initial split, in accordance with the test of significance at the current level of *p-value* (see above).

b) For variables whose *Usage* is "ordinal" or "continuous", the missing value is mapped to a separate value (token) of the variable and will be used just like any other value when calculating the split. Missing values are displayed as ***null*** in the tree view, the Range Editor of Force Split and Edit Split, and other views and dialogs.

During the iterative process of the Find Split algorithm, the missing value (*null*) bin can be grouped with any other bin (interval) starting from the initial iteration when the algorithm starts to merge the bins in the initial split, in accordance with the test of significance at the current level of *p-value* (see above).

***Ignore*:**

Missing values in the variable will be ignored. Any record with a missing value in this variable will be dropped from the analysis. Any split on this variable will not include the condition "`IF <variable_name> is NULL`", so the child nodes in this split will not contain records with missing values in this variable. Note that in this

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

case the sum of the numbers of records in the child nodes of a split will not necessarily add up to the number of records in their parent node.

Float:

Before the Find Split algorithm starts the iterative process of calculating the split, the bin for missing values in the variable is removed from the initial set of bins for other values and is held to the side. With the regular values remaining, the Find Split algorithm then performs the normal iterative process of merging the bins in accordance with the test of significance at the current level of *p-value* (see above).

The last step of the algorithm will then take the missing value bin and search for the bin which it "best fits" according to the test of significance, and place it there if it is satisfies the criterion. Otherwise the missing value bin remains separate in the final split.

Missing values are displayed as ***null*** in the tree view, the Range Editor of Force Split and Edit Split, and other views and dialogs.

The *Float* option is the default for continuous and ordinal independent variables.

Separate bin:

Missing values will not be grouped with other data in the binning process of the Find Split algorithm. All records with missing values for the given variable (if any exist) will be put in a separate child node.

The behavior is the same as the ***float*** option except at the last step the missing value bin is not merged with any other bin at the end. Before the Find Split algorithm starts the iterative process of calculating the split, the bin for missing values in the variable is removed from the initial set of bins for other values and is held to the side. With the regular values remaining, the Find Split algorithm then performs the normal iterative process of merging the bins in accordance with the test of significance at the current level of p-value. At the last step of the algorithm, the missing value bin remains in its own separate node in the final split.

3. For variables used as weight: Not applicable.

Ordered Display

Allows you to define how ordered variables are displayed. There are three options:

Present: displays all values which are present in the node. If a particular value of the variable is not present in the node, it will not be listed.

All: displays all values that fit within the range in the node, whether they are present in that particular node or not. For example, if a node contains the values 1, 2, 4, and 5 (and the value 3 exists somewhere in the dataset), the value 3 will be listed for the node.

Range: displays values as a range. For example, if a split groups the values 1, 2, 3, and 4 together, the values will appear as [1, 4].

Max Branches

The **Max Branches** value determines the maximum number of branches (leaves) produced by Find Split on the given variable. For example, if you want Find Split to produce binary splits for a specific independent variable, specify "2" in the Max Branches column for this variable. If you want to grow a binary tree (i.e., a tree with binary splits only), enter "2" in the Max Branches column for all variables. See above for the tip on changing attributes for more than one variable at a time.

Note that this setting does not apply to the Force Split operation. Also, note the difference from the **Default # of Bins** attribute. The Find Split algorithm *starts* calculating the split with the number of branches equal to Default # of Bins. As opposed to Default # of Bins, the Max Branches limit never leads to the creation of the 'other' ("###") category for nominal variables.

Open Left

The splits created by the independent variable are not bound by the minimum data values found in the dataset.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Open Right	The splits created by the independent variable are not bound by the maximum data values found in the dataset.
Cardinality	The cardinality (number of unique values) of the variable.
Format String	Allows you to specify various formats (such as currency, percent, etc.) and precision for displaying values in the tree. You can either choose one of the standard formats from the Format String drop-down list or enter a custom format. For detailed description of format strings, see Specifying Format.

By setting the ***Usage,*** you can define the way the values of a field are treated by the tree algorithm and, therefore, the way they will be displayed in the splits: how they will be grouped, whether they will be represented in ranges, etc. For example, if the values of a field are ranked (like 1, 2, 3, or low, medium and high) then they are *ordinal*. If there is no order, i.e., if they are just categories or qualities, they are considered to be *nominal* values (e.g., "red", "blue" and "green").

Usage set to 'no binning'

When set to ***no binning***, each value of the independent variable forms a separate branch, i.e., individual values are presented as unique nodes in the tree.

Example

Usage set to 'ordinal'

Values of this Usage are treated as an ordered set (e.g., 1, 2, 3, or a,b,c...). Only values immediately succeeding one another will be grouped together when an algorithm is looking for similarities. For example, 'a' can be paired with 'b', but not with 'c'.

This Usage is the default for independent variables of types Number, Integer, Date, Time, or Timestamp if their cardinality is less than 50.

You can assign the Usage 'ordinal' to String variables (which are 'nominal' by default). In this case, when creating the split, the algorithm will order the values alphabetically and then try to group contiguous values together to create a branch. If your data is not alphabetically ordered, you may want to create a calculated column that transforms it. For example, use the Binning helper in Dataset Editor to transform the category 'low' to '***1-low'***, 'medium' to '***2-medium'*** and 'high' to '***3-high'.***

Example

Usage set to 'nominal'

Values of this Usage are treated as an unordered set. All possible groupings of field categories are examined so that similar items are grouped together. This Usage is the default for String and Boolean variables.

Example

Usage set to 'continuous'

This Usage is available for independent variables of types Number, Integer, Date, Time, or Timestamp and it is the default if the cardinality is higher than 50. A variable of this Usage is split into intervals that are calculated in a number of different ways based on your selections in the Intervals Type column and the ***Intervals/Bins*** column. See also the topic Using the Attribute Editor for more information about the Intervals/Bins column.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

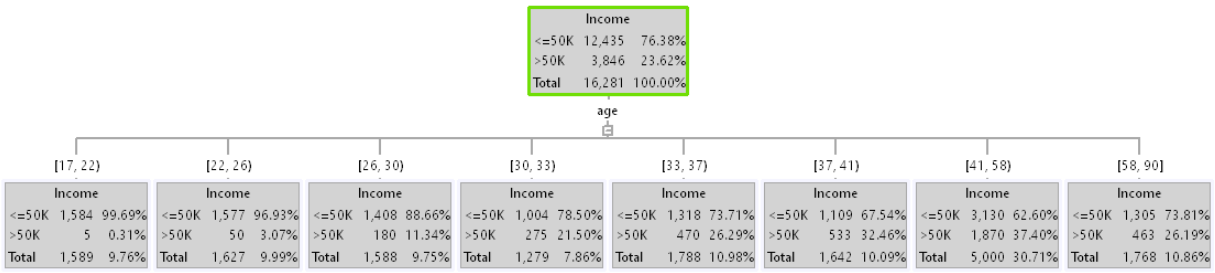
The ***Default # Bins*** attribute affects both Find Split and Force Split commands.

For **Find Split**, this attribute affects the maximum number of child nodes that can be produced by Find Split for a given independent variable (IV). More precisely, it determines the maximum **initial** number of bins created at the initial stage of the Find Split operation. During the steps of the Find Split algorithm, some bins may be merged, so the number of bins (child nodes) in the resulting split cannot exceed the value of this attribute.

For **Force Split**, the ***Default # Bins*** attribute determines the default number of bins in the Force Split Range Editor dialog. The actual number of bins or intervals in the Range Editor for a given IV is the lesser of the *Default # Bins* value and the *cardinality* of the variable.

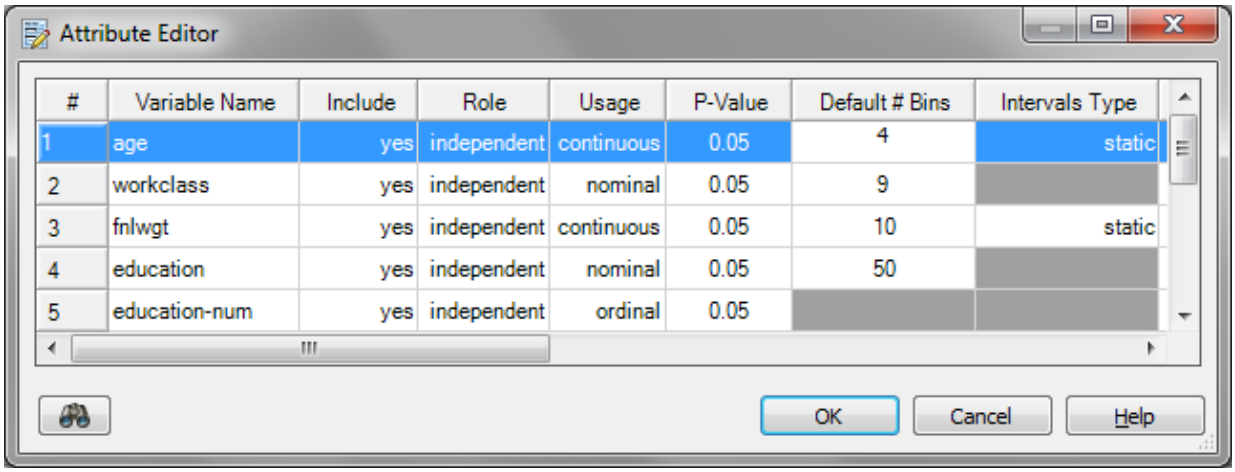
Example: The Find Split operation produces a split on the variable *age*. The Default # Bins value for *age* was set to 10. As shown below, Find Split produces eight child nodes. The algorithm started with 10 bins and merged several bins in the process.

Figure 1



Through the attribute editor, change ***Interval*** to ***"8"***.

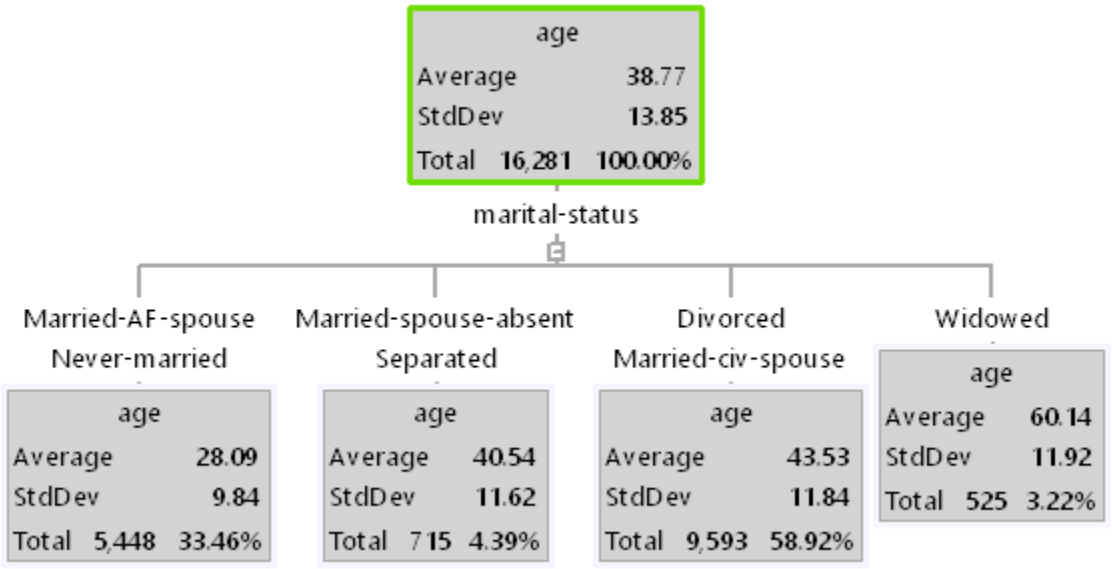
Figure 2



The resulting number of nodes will be four (4).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Figure 3



Weight in Strategy Trees

The type of weight that can be defines in datasets and models is a generalized frequency weight (the weight values don't have to be integer). A weight can be set at the dataset level, but it can also be set and changed independently in trees and models.

See the Weight topic for general information about weights and on setting weight at the dataset level.

Weights in Datasets and Dependent Models

The weights in a dataset and its dependent models can be changed or reset independently. When you toggle weight on or off in the dataset Overview Report, this does not affect existing trees and other models depending on the dataset.

By default, any new model created on the basis of a weighted dataset inherits the weight from it. More precisely, when inserting a model for a weighted dataset, the weight field defined in the dataset is automatically listed in the model's Attribute Editor as "weight" (in the Usage column). You can redefine the field as an analysis variable in the Attribute Editor by changing the usage from "weight" to "independent" if desired. Once a model is inserted, it is independent of the parent dataset in the sense that any changes of weights in the model will not affect the parent dataset and vice versa.

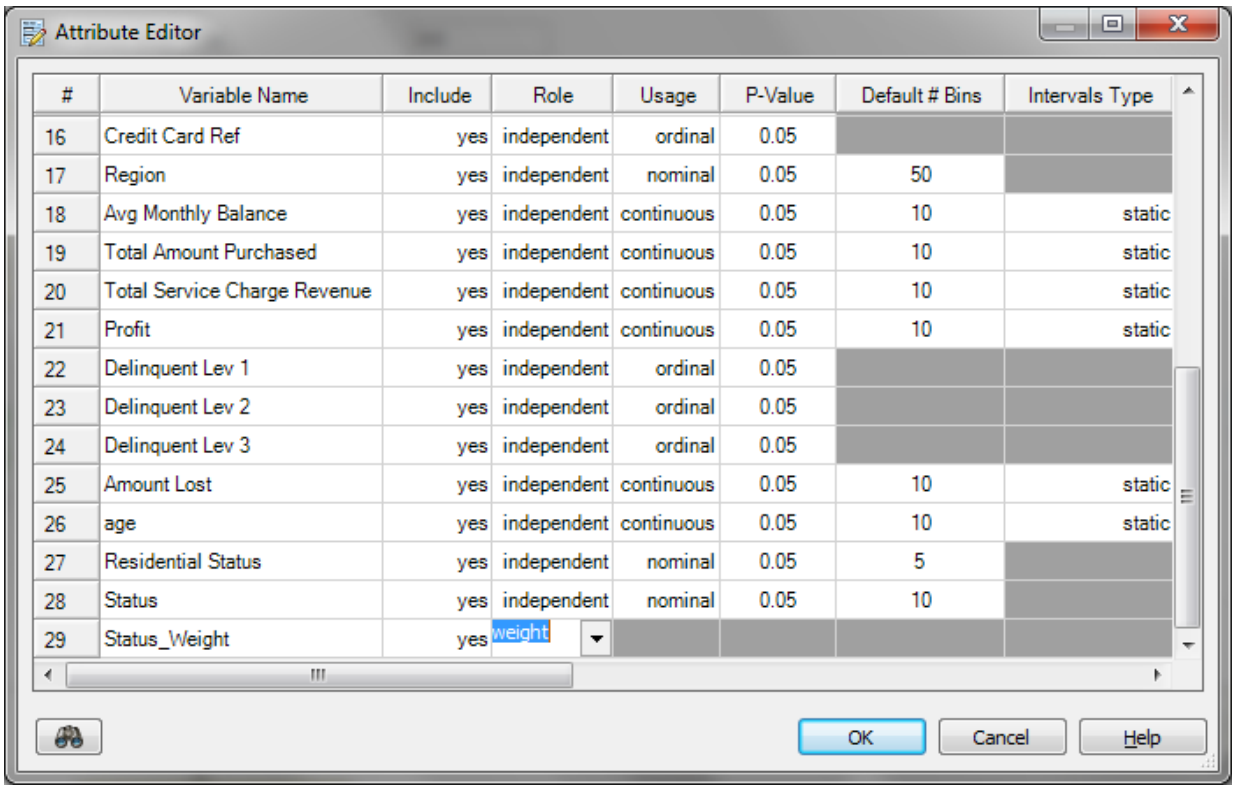
Weight in Strategy Trees

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To specify weight for a Strategy Tree, open the tree and select **Tools | Attribute Editor** from the menu.

It is assumed that the field you want to set as weight already exists in the parent dataset. See the Weight and Weight Helper topics for information on available tools for creating a weight field. If the parent dataset already had weight when you were inserting the tree, the same weight is set in the tree by default.

Click in the ***Usage*** column for the variable you want to set as weight and select "**weight**" from the drop-down list. Note that this usage option is available only for nonnegative numeric variables. The weight variable does not have to be integer.

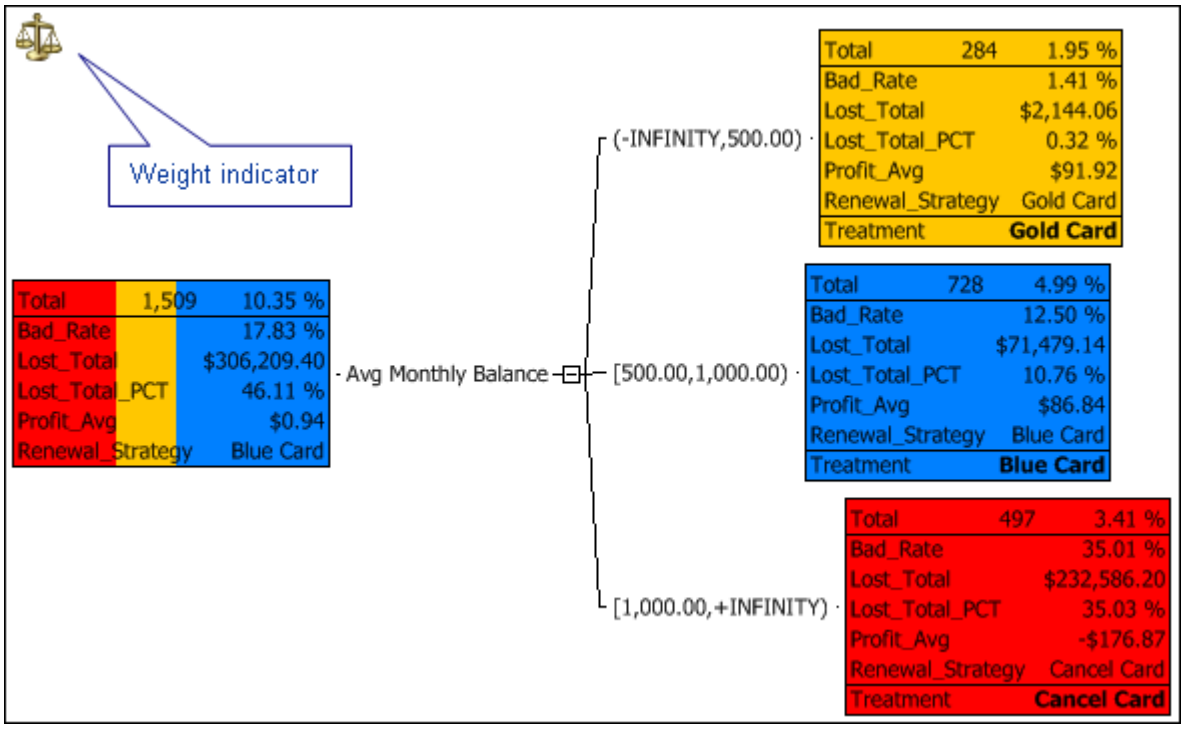


#	Variable Name	Include	Role	Usage	P-Value	Default # Bins	Intervals Type
16	Credit Card Ref	yes	independent	ordinal	0.05		
17	Region	yes	independent	nominal	0.05	50	
18	Avg Monthly Balance	yes	independent	continuous	0.05	10	static
19	Total Amount Purchased	yes	independent	continuous	0.05	10	static
20	Total Service Charge Revenue	yes	independent	continuous	0.05	10	static
21	Profit	yes	independent	continuous	0.05	10	static
22	Delinquent Lev 1	yes	independent	ordinal	0.05		
23	Delinquent Lev 2	yes	independent	ordinal	0.05		
24	Delinquent Lev 3	yes	independent	ordinal	0.05		
25	Amount Lost	yes	independent	continuous	0.05	10	static
26	age	yes	independent	continuous	0.05	10	static
27	Residential Status	yes	independent	nominal	0.05	5	
28	Status	yes	independent	nominal	0.05	10	
29	Status_Weight	yes	weight				

Note that any existing splits in the tree will be deleted after you specify weight.

If a weight is defined, the scales icon at the top-left corner of the Tree view serves as an indicator that the tree is weighted. In addition to the Attribute Editor, the information about the weight is available in the Parameters and Attributes tab of the tree.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Since weight in Altair Data Analytics applications is treated as a (generalized) frequency weight, the effect of setting a weight in trees is as follows:

The number of records in all tree nodes is recalculated based on the defined weight (equals the sum of the weight field over all records in each node).

> NOTE: Non-integer node totals are possible, since the sum of the weight field over a node is not necessarily integer. By default, the node totals displayed in tree nodes are formatted as integer numbers, but you can change this setting by selecting Tools | Options and changing the format in the *Show Support (# of records)* field, for example, to 0.00 or #,##0.00 (for two digits after the decimal point).

If a node is produced by a Find Split (and therefore shows a DV distribution), the weighted distribution of the dependent variable is as follows:

If W is the weight variable, F_i is the i th value of the discrete dependent variable F, and SUM(W_i) is the sum of the weight field over all records in the node where $F=F_i$, then

- The weighted total number of records = SUM(W) (over the node)
- The weighted frequency for the value F_i = SUM(W_i) (over the node)
- The weighted percentage of the value F_i = 100 * SUM(W_i) / SUM(W) (sums taken over the node)

The effect in other tree views: All charts in the Tree Chart and Profile Chart are calculated based on the defined weight. Also, note that the columns of Node Report may display non-integer values.

If the dependent variable is continuous, the average and standard deviation are weighted.

All node calculations are also calculated based on the defined weight.

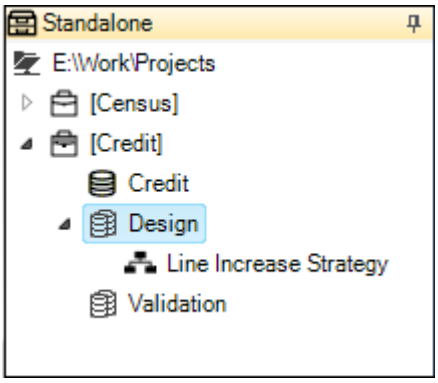
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Validating and Scoring a Strategy

Strategy Validation

As a prerequisite for strategy development and subsequent validation, the dataset is usually partitioned into **Design** and **Validation** samples. The strategy is designed on the **Design** partition and then validated on the **Validation** partition.

The purpose of **Strategy Validation** is to monitor the changes in KPIs in the data segments defined by a given Strategy Tree. More precisely, it is to compare the values of KPIs expressed by node calculations in the development dataset and a validation (testing) dataset. The operation produces a *Strategy Validation Tree* whose nodes show the original KPI values, the KPI values calculated on the validation sample, and the comparison metric requested by the user.

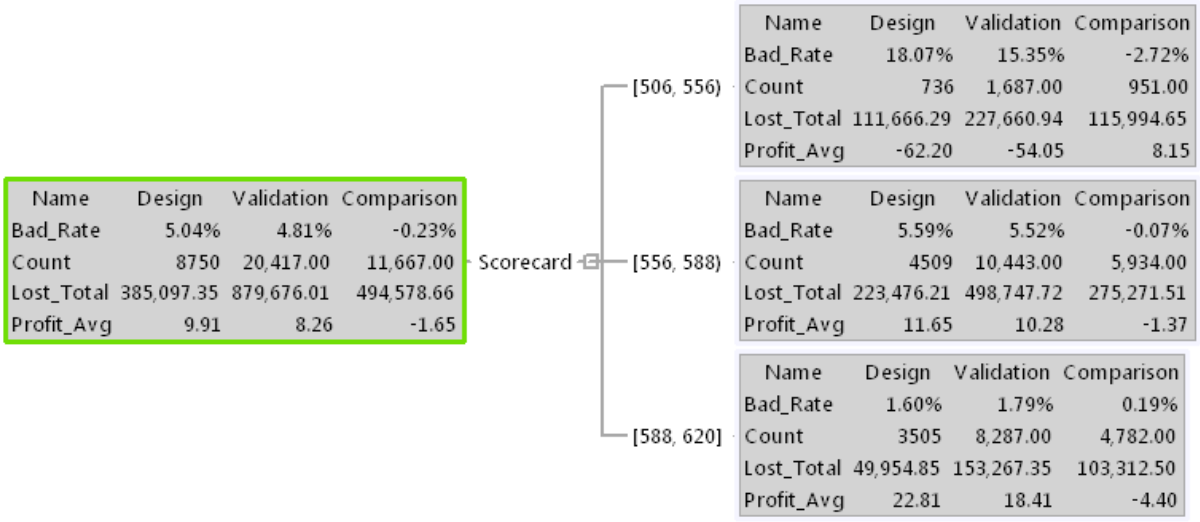


Strategy validation is used to:

- assess the statistical validity of the strategy design before going into production
- assess if a strategy that has passed the testing or early stage of production meets the intent of the original design
- assess if the original strategy design assumptions are still valid after using the strategy in production for some time
- assess the performance of a strategy on a node-by-node basis

Strategy validation results are displayed in the form of a validation tree that combines information from the strategy trees applied to the design and validation samples along with user-defined comparison metrics.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The results of each of the node calculations from the **design** dataset and the **validation** dataset are displayed on the nodes. The comparison column shows you the results of the comparison, where the method of comparison is defined in the validation wizard. In the above example, the comparison method is the difference (**validation - design)**.

Proceed to the topic Starting Strategy Validation.

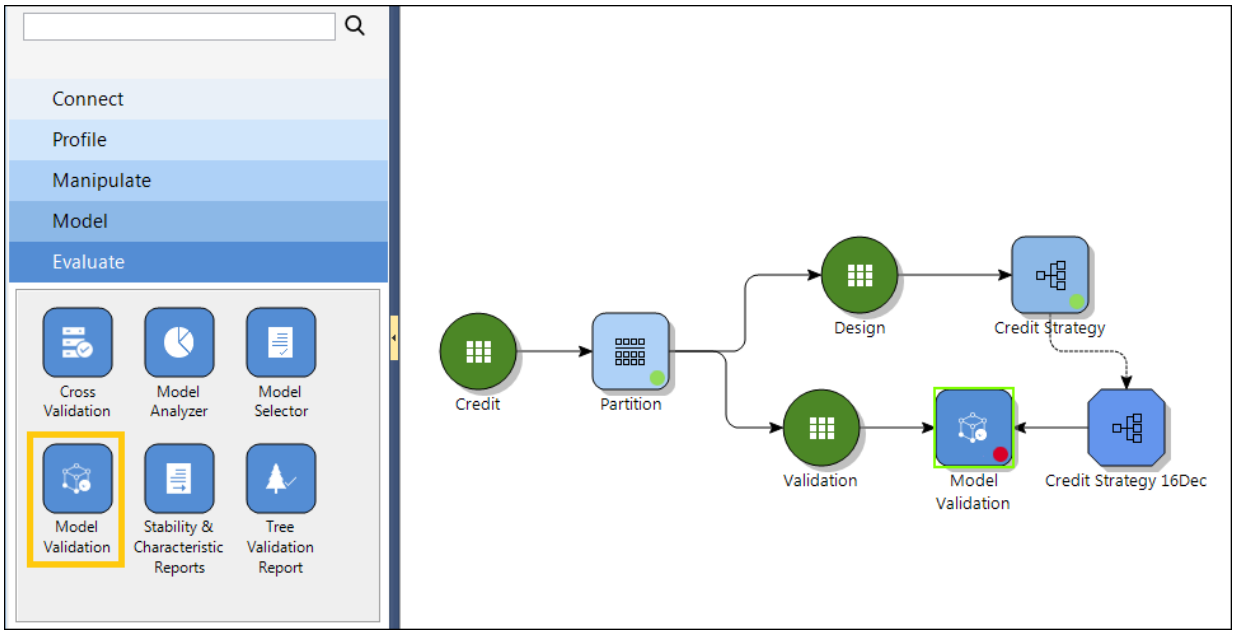
The purpose of **Strategy Validation** is to monitor the changes in KPIs in the data segments defined by a given Strategy Tree. More precisely, it is to compare the values of KPIs expressed by node calculations in the development dataset and a validation (testing) dataset. The operation produces a *Strategy Validation Tree* whose nodes show the original KPI values, the KPI values calculated on the validation sample, and the comparison metric requested by the user.

Strategy Validation is applied to Strategy Tree instances, since it is important to know what the state of the strategy tree was at the time of validation. If you apply validation to a Strategy Tree *process* node, a model instance will be automatically saved before validation, and the validation will be applied to the instance.

To validate a strategy, drag the **Model Validation** node from the **Evaluate** palette onto the Workflow canvas and connect the Strategy Tree node and the input dataset to it. If the Strategy Tree node is a process node (rather than a model instance node), you will be prompted to create a model instance first.

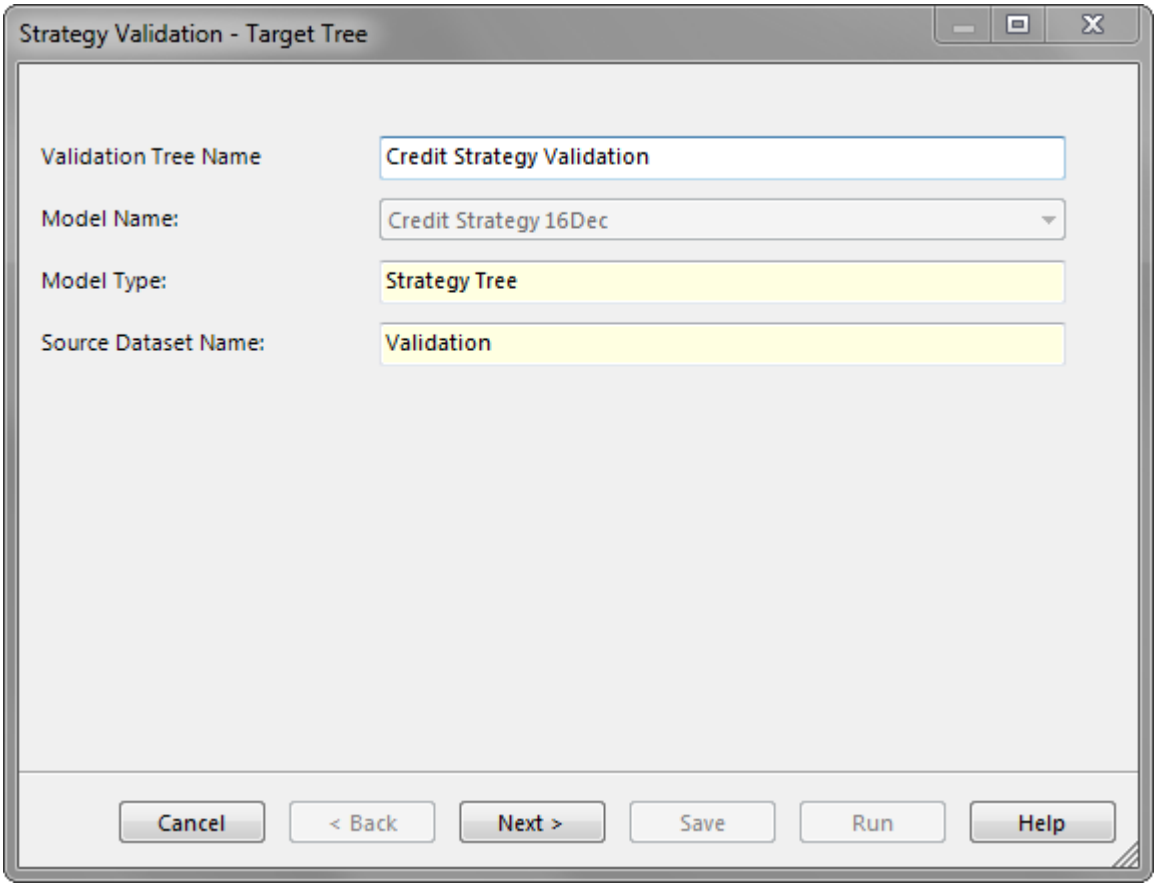
Alternatively, right-click anywhere on the workflow canvas and select the context menu command **Evaluate --> Model Validation**. This will create a new Model Validation node. Connect the input dataset and the Strategy Tree to it.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Double-click the **Model Validation** node or right-click on it and select **Modify** to open the wizard.

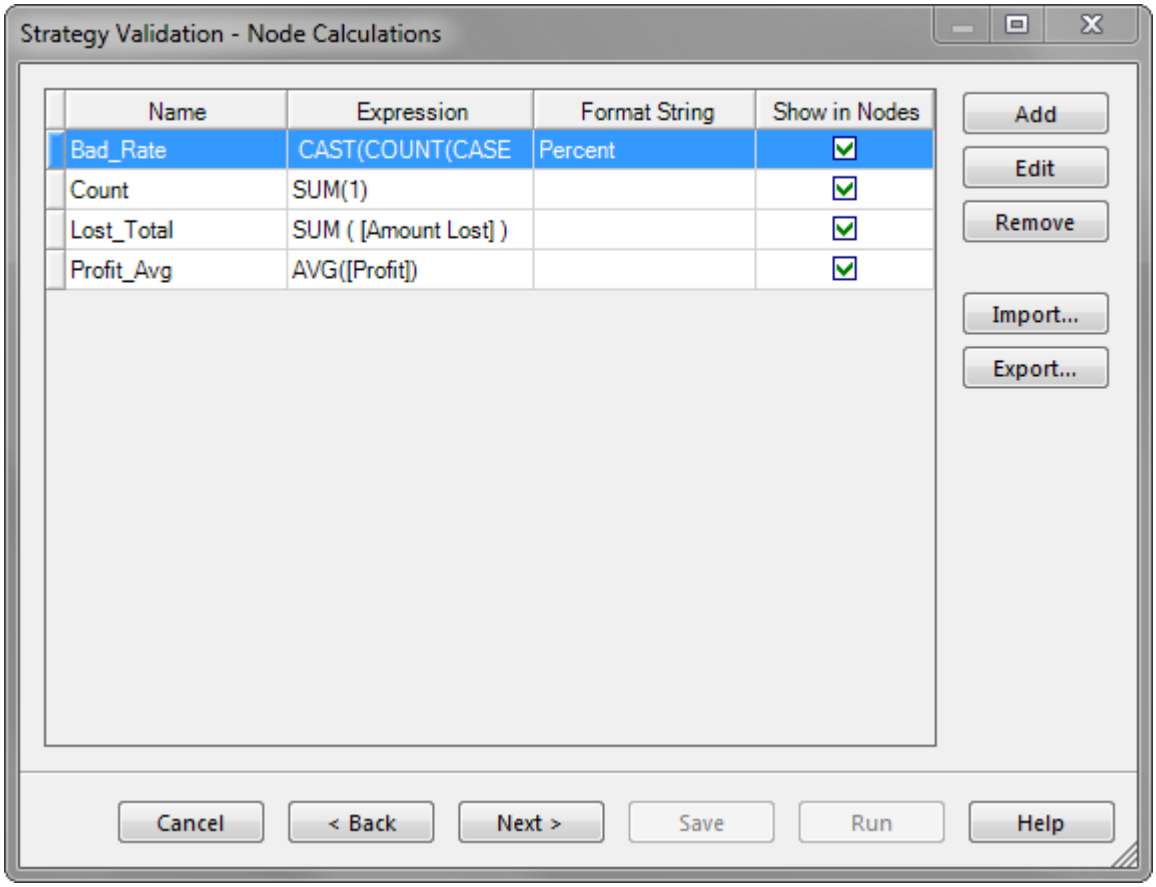
Specify the name for the output Strategy Validation Tree, which will be embedded in the Model Validation node:



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Click **Next** to proceed to the Node Calculations page to review or edit the existing KPI calculations or add new ones.

In the Node Calculations page, review the existing calculations, edit them or add new ones if necessary. You can also import calculations from other strategy trees or previously saved expression files.



Name	The name of the calculation as it appears in the node of the strategy model.
Expression	SQL expression representing the node calculation.
Format String	The format of the expression to be displayed in the node. See the topic Specifying Format for details.
Show in Nodes	Choose whether to show or hide the calculation values in the tree nodes.

Operations:

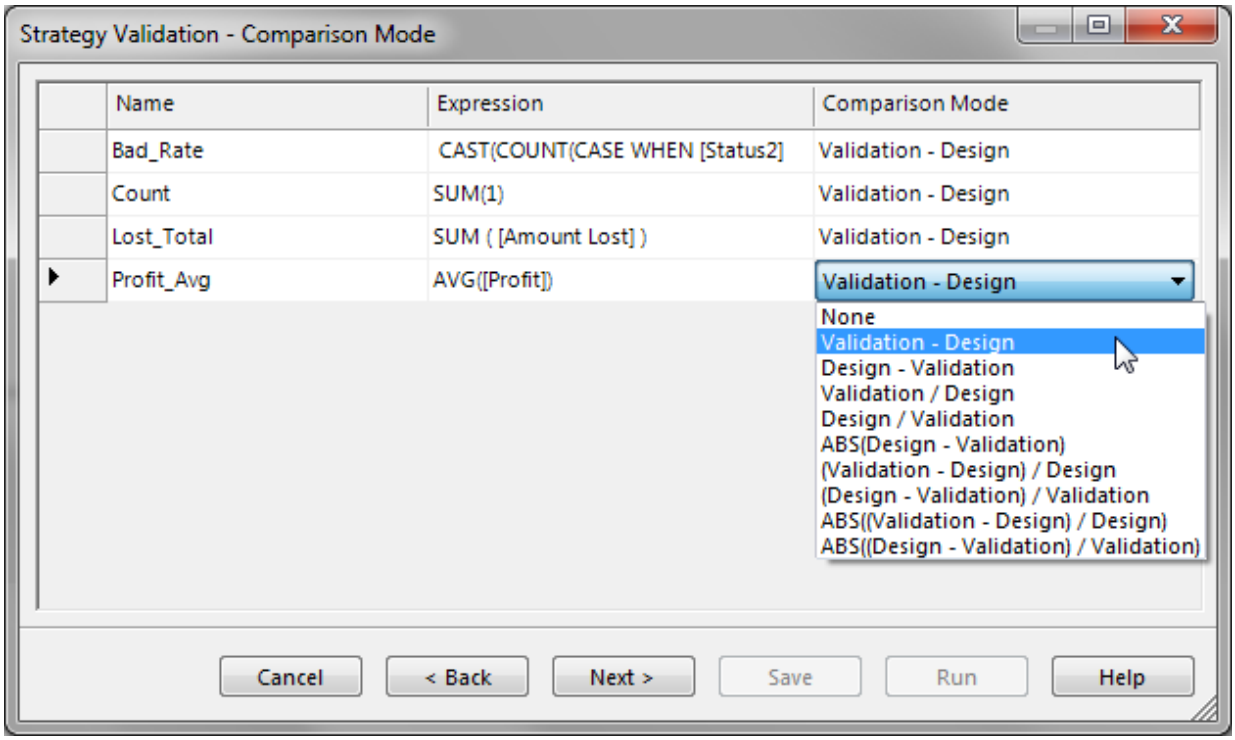
Add	Add a calculation to the node. This button calls the expression editor to help in creating the calculations.
Edit	Edit an existing node calculation in the expression editor.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Remove Remove a node calculation.

Click **Next** to proceed to the Comparison Mode page to define the type of comparison metric for the KPIs.

In the Comparison Mode page of the wizard, choose the desired comparison metric.



Name Name of the calculation as it appears in the node of the strategy model.

Expression SQL expression representing the node calculation.

Comparison Mode The following comparison modes are available:

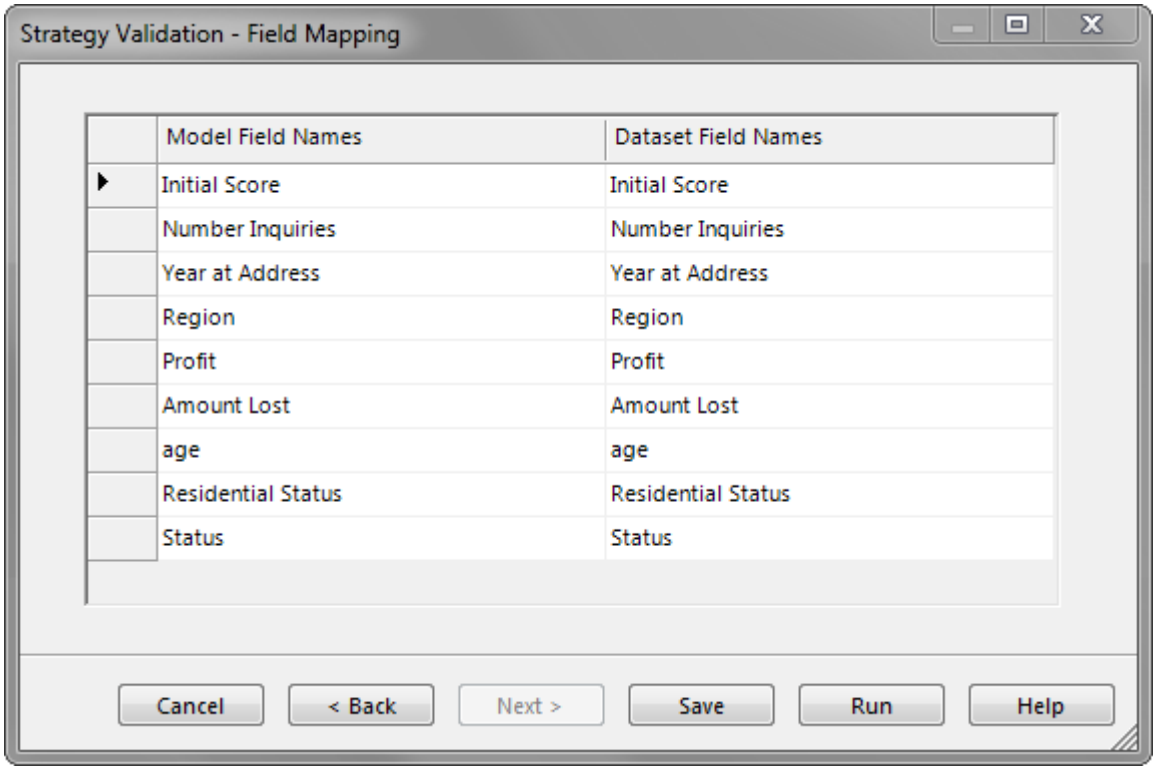
- "None" does not perform any comparison
- "Validation - Design" subtracts the data in the learning dataset (Design) from the data in the test dataset (Validation)
- "Design - Validation" subtracts the data in the test dataset (Validation) from the data in the learning dataset (Design)
- "Validation / Design" divides the data in the test dataset (Validation) by the data in the learning dataset (Design)
- "Design / Validation" divides the data in the learning dataset (Design) by the data in the test dataset (Validation)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- "ABS(Design - Validation)" takes the absolute value of the difference between the data in the learning dataset (Design) and the data in the test dataset (Validation)
- "(Validation - Design) / Design": provides the ratio of the difference between the data in the test dataset (Validation) and in the learning dataset (Design), and the data in the learning dataset (Design).
- "ABS ((Validation - Design) / Design)" is the absolute value of the preceding expression.
- "(Design - Validation) / Validation": provides the ratio of the difference between the data in the learning dataset (Design) and in the test dataset (Validation), and the data in the test dataset (Validation).
- "ABS ((Desing - Validation) / Validation)" is the absolute value of the preceding expression.

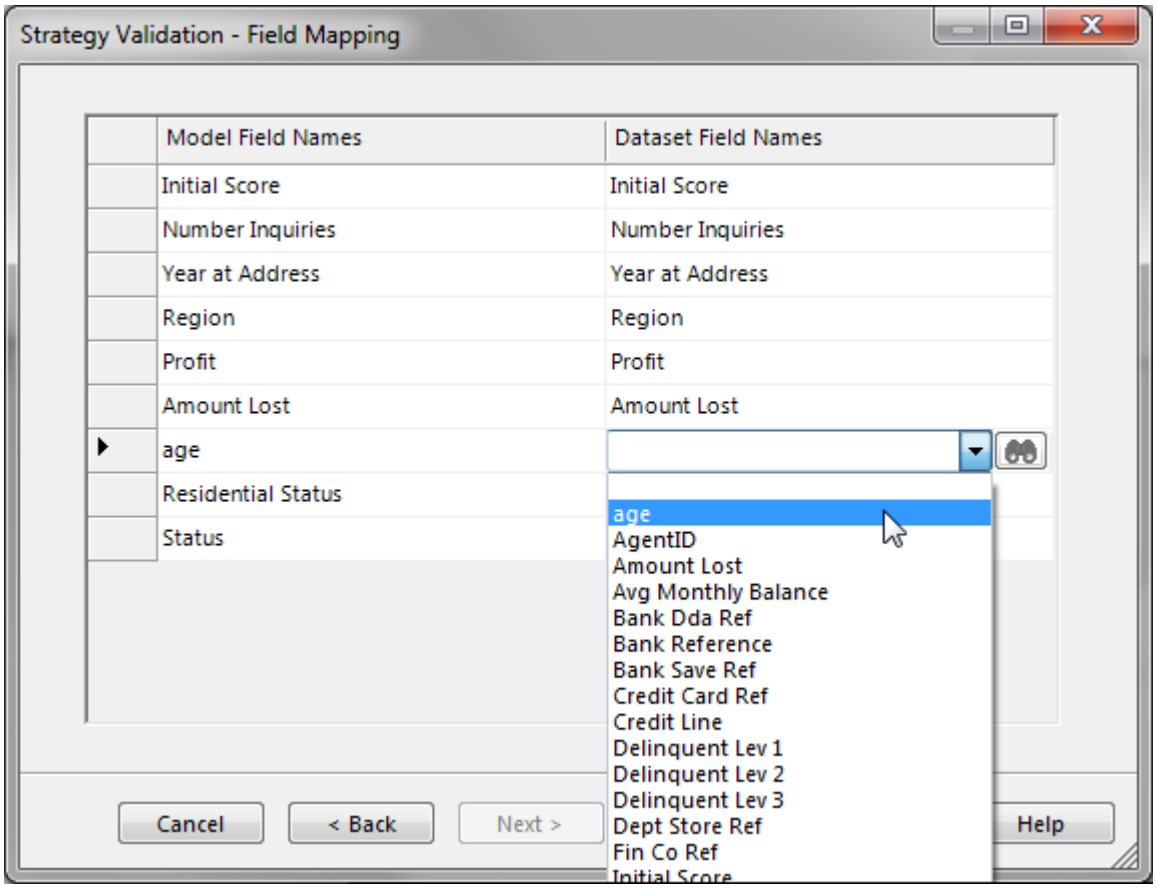
Click **Next** to proceed to the Field Mapping page to map the strategy variables to the input dataset fields if necessary.

In the general case, the field names in the input dataset for strategy validation or scoring may be different from those in the original design dataset used to build the strategy tree. This step of the wizard allows you to map the fields if necessary. If identical or similar field names are found, they are matched automatically.

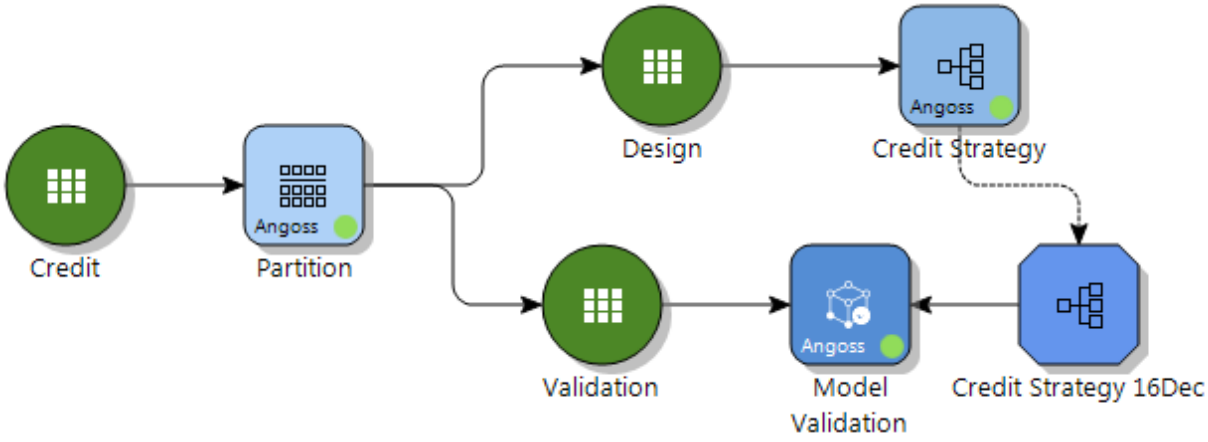


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If there is a dataset field that is not mapped or is mapped incorrectly, click in the corresponding cell and select the right field as shown below.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the operation, double-click the **Model Validation** node to view the results.

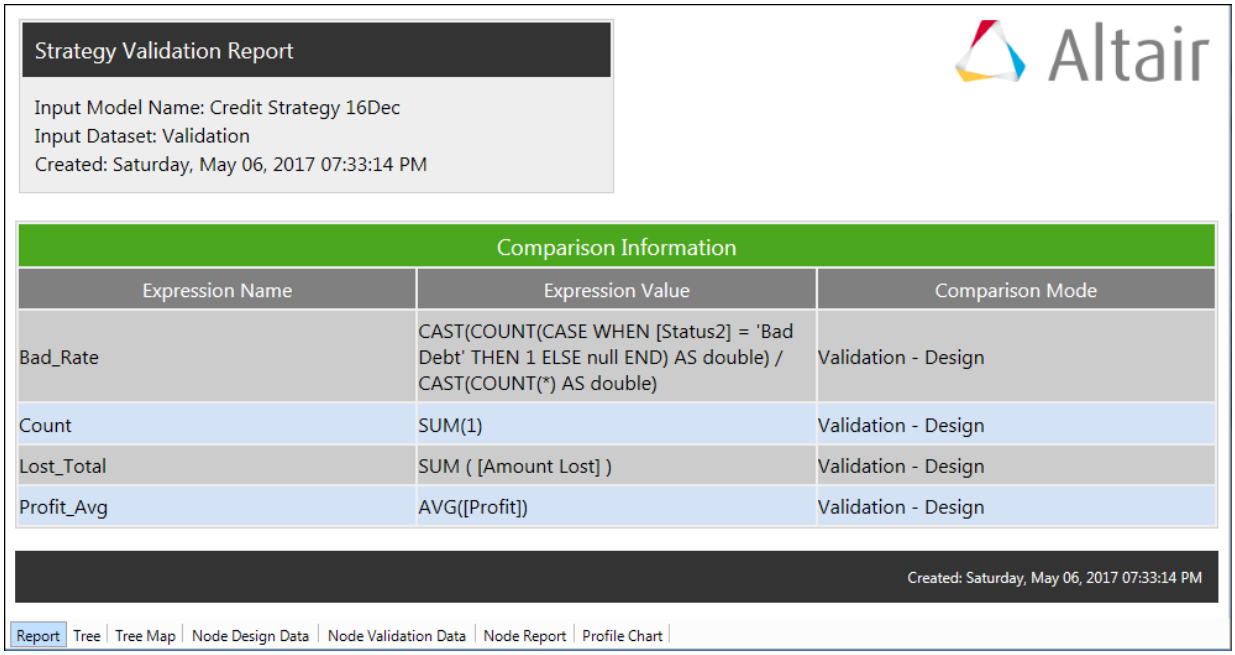


Proceed to the Validation Tree View topic.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The default view displayed when you open the Model Validation node after Strategy Validation is a report that summarizes the validation inputs:

- Date and time of execution
- Input model name
- Input dataset name
- KPIs (expressions) defined
- Comparison metric (mode) for each KPI



Strategy Validation Report

Input Model Name: Credit Strategy 16Dec
Input Dataset: Validation
Created: Saturday, May 06, 2017 07:33:14 PM

Comparison Information

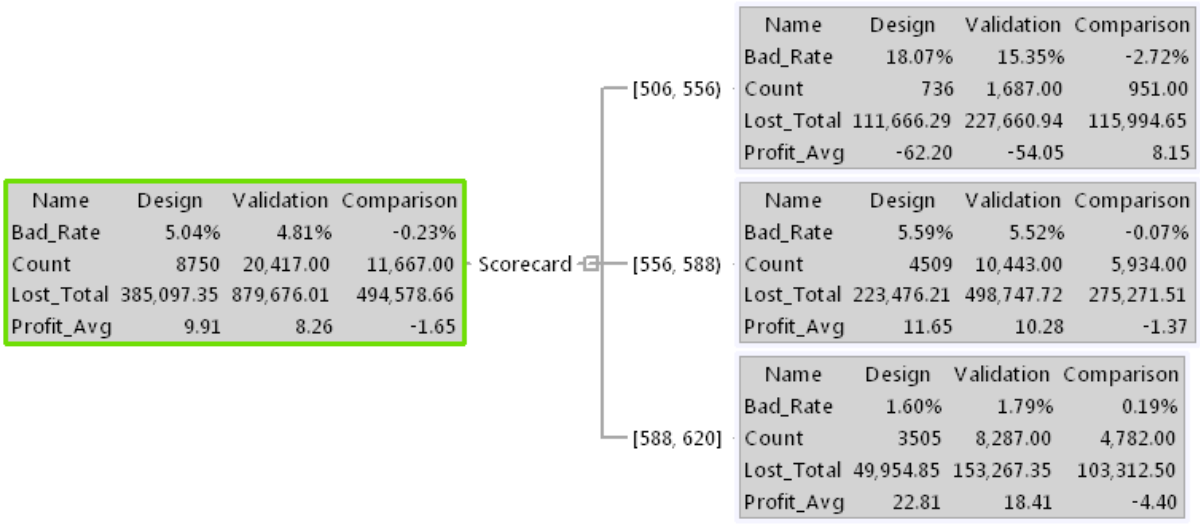
Expression Name	Expression Value	Comparison Mode
Bad_Rate	CAST(COUNT(CASE WHEN [Status2] = 'Bad Debt' THEN 1 ELSE null END) AS double) / CAST(COUNT(*) AS double)	Validation - Design
Count	SUM(1)	Validation - Design
Lost_Total	SUM ([Amount Lost])	Validation - Design
Profit_Avg	AVG([Profit])	Validation - Design

Created: Saturday, May 06, 2017 07:33:14 PM

Report | Tree | Tree Map | Node Design Data | Node Validation Data | Node Report | Profile Chart

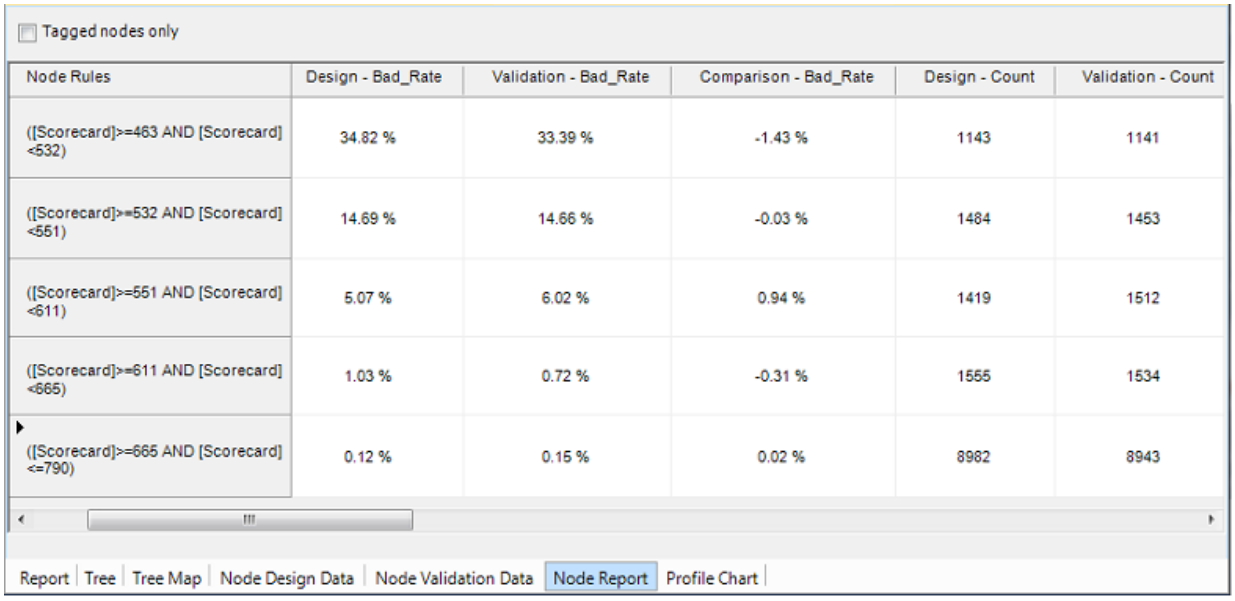
The *Tree* view of the Model Validation node, also referred to as the Strategy Validation Tree, displays the results of the node calculations from both the Design and Validation datasets, along with the values of the KPI comparison metric.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



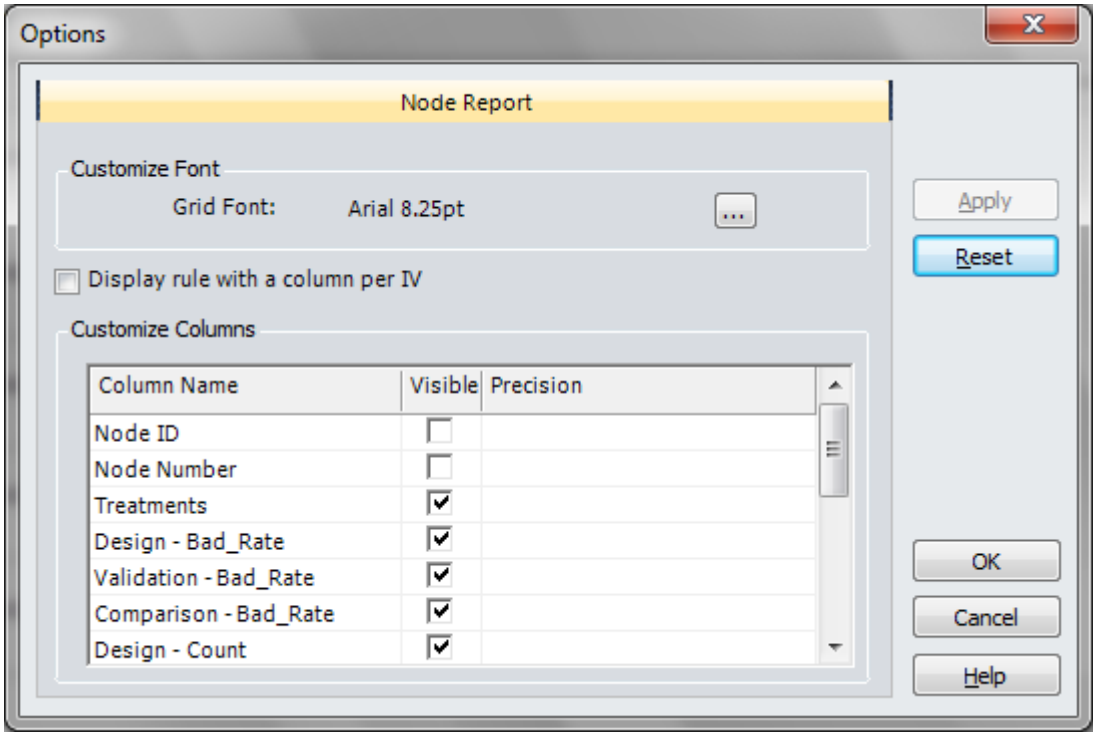
Name	The node calculation name (KPI name).
Design	The value of the node calculation in the original strategy model based on the Design dataset.
Validation	The value of the node calculation in the strategy model based on the Validation dataset.
Comparison	The value of the comparison metric (comparison between the KPIs calculated in the segments of the Design and Validation datasets), as defined in the Strategy Validation wizard.

The validation tree node report provides you with information about the validation results. As with the strategy tree node report, only the results for the terminal nodes are displayed



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Columns can be added and removed from the report by using the Options.



The report can be sorted on a particular column. For example, you want to see all the nodes that have a treatment.

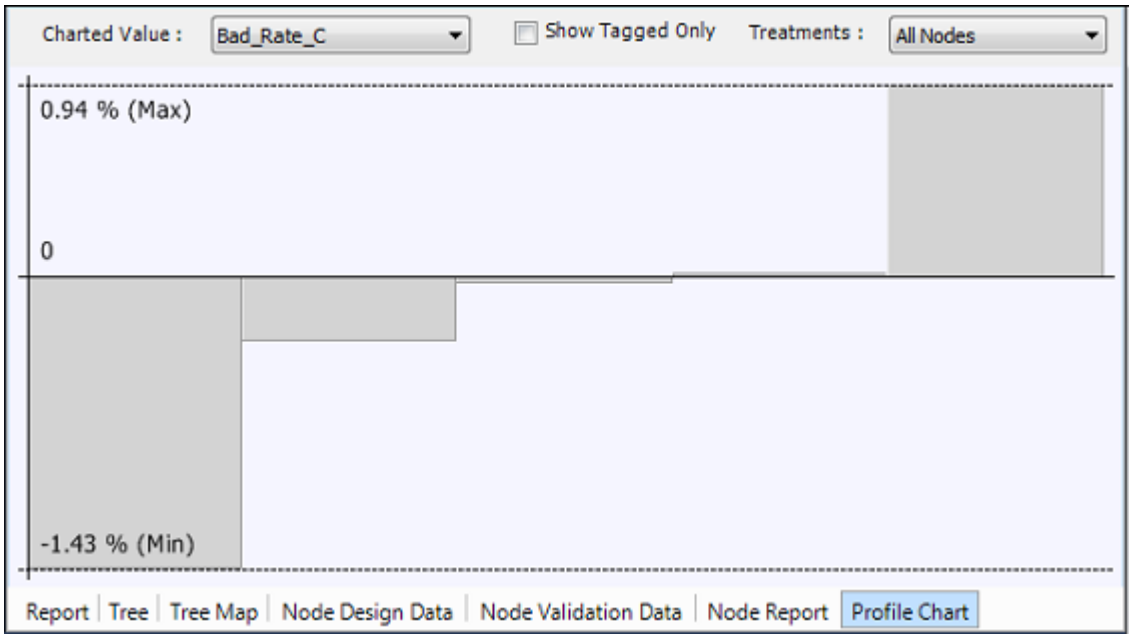
To sort the columns, click on the column heading.

Treatment Summary

There is a summary of the treatment nodes located at the bottom of the report. This gives you a quick report on the total number of records that have been labeled with a given treatment.

The strategy validation **Profile Chart** view displays each terminal node in tree. Its purpose is to display the tree in a single chart. It differs from the design strategy tree model profile chart in that the charted values available are the node calculations from both the design and the validation.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Clicking on a bar will cause the tree to navigate to that node. You can also right click to access commands such as assigning treatments and tagging nodes.

Charted Value	The charted value presents a list of each of the tree calculations from both the design and the validation trees. Choose the value that you would like charted.
Show Tagged	If this is checked, only nodes that are tagged are displayed.
Treatments by Default	By default, all terminal nodes are displayed. Changing the treatments option allows you to plot only nodes that are assigned to a specific treatment.
Support as Width	When checked, the width of the bar represents the number of observations in the node. If unchecked, each bar is the same width regardless of the number of rows.

Strategy Scoring

Scoring a strategy is the process of assigning treatments to the records of a new dataset using the rules and treatment assignments in a Strategy Tree.

If the Strategy Tree has node calculations, you also have an option to assign the calculation values to each record being scored.

The values of treatments and calculations assigned to the input records in the process of strategy scoring are written to the corresponding new fields of the output dataset.

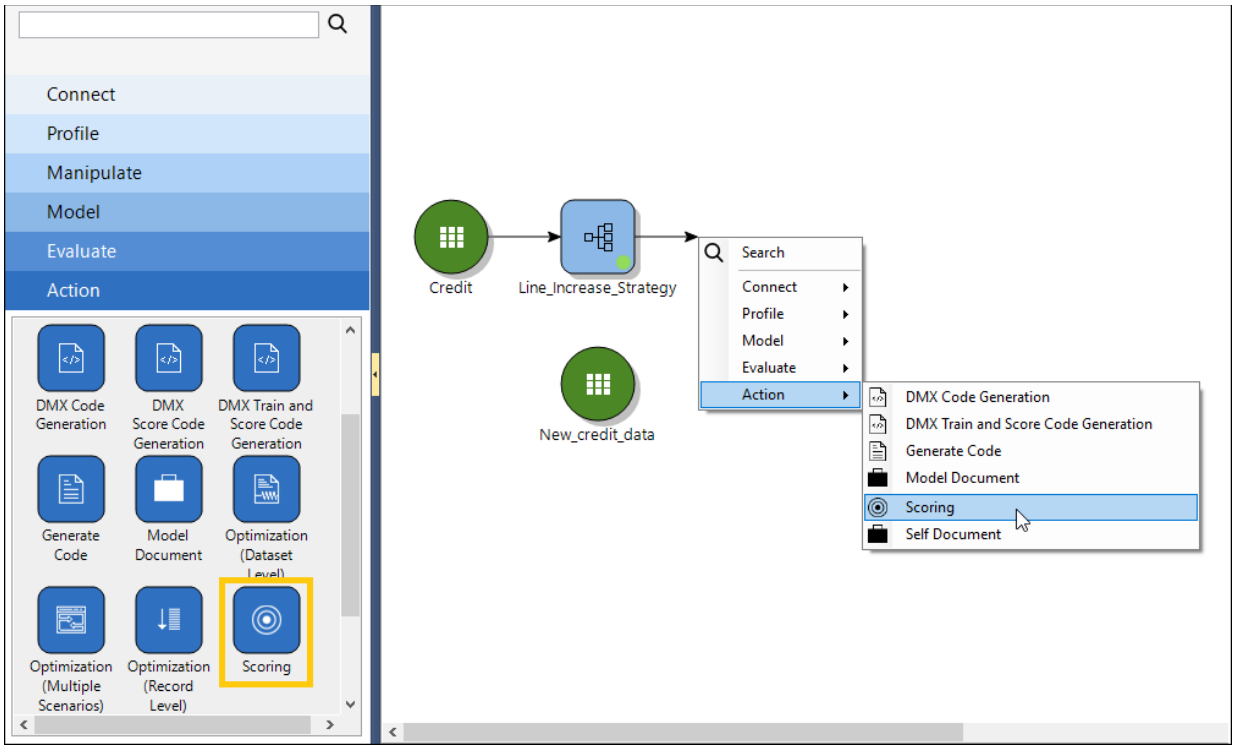
Use case examples:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Strategy scoring can be used to apply strategies to test data for analysis and reporting with the purpose of estimating their effects. For example, after applying a strategy that assigns credit limit increase amounts, you can compute new derived variables such as the new (revised) credit limit, estimated future utilization, interest income and write-offs, and then display them in a new strategy tree or simply report the total effect. This may be helpful in explaining the benefits and risks of a new strategy.
- Treatment lists generated by scoring a strategy can be used by an operational application. A list of actions generated by deploying a collections strategy can be processed by a call center operator. A list of actions generated by deploying a marketing campaign strategy can help build separate lists of customers or prospects to be contacted via specific channels (Email, phone, mail) defined by the strategy.

The strategy scoring process reads the input dataset and associates each input record with a strategy tree node according to the rules defined by the tree splits. The treatment and the KPI found in the node (if any) are assigned to the record.

Strategy tree scoring is performed using the **Scoring** node from the **Action** palette. See the topic Starting Strategy Scoring.



In-Database Strategy Scoring

You can perform in-Database strategy scoring if the In-Database Analytics feature is included in your product. In this mode, strategy scoring commands are executed directly on the database server and the generated treatments are stored in the database. Altair Data Mining Engine sends the strategy scoring commands to the database engine, which reads the records from the input table and writes the results (treatments) to the target table. No data extraction occurs in this process, and the commands are performed by the database engine rather than the Altair Data Mining Engine.

See also: Starting Strategy Scoring

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

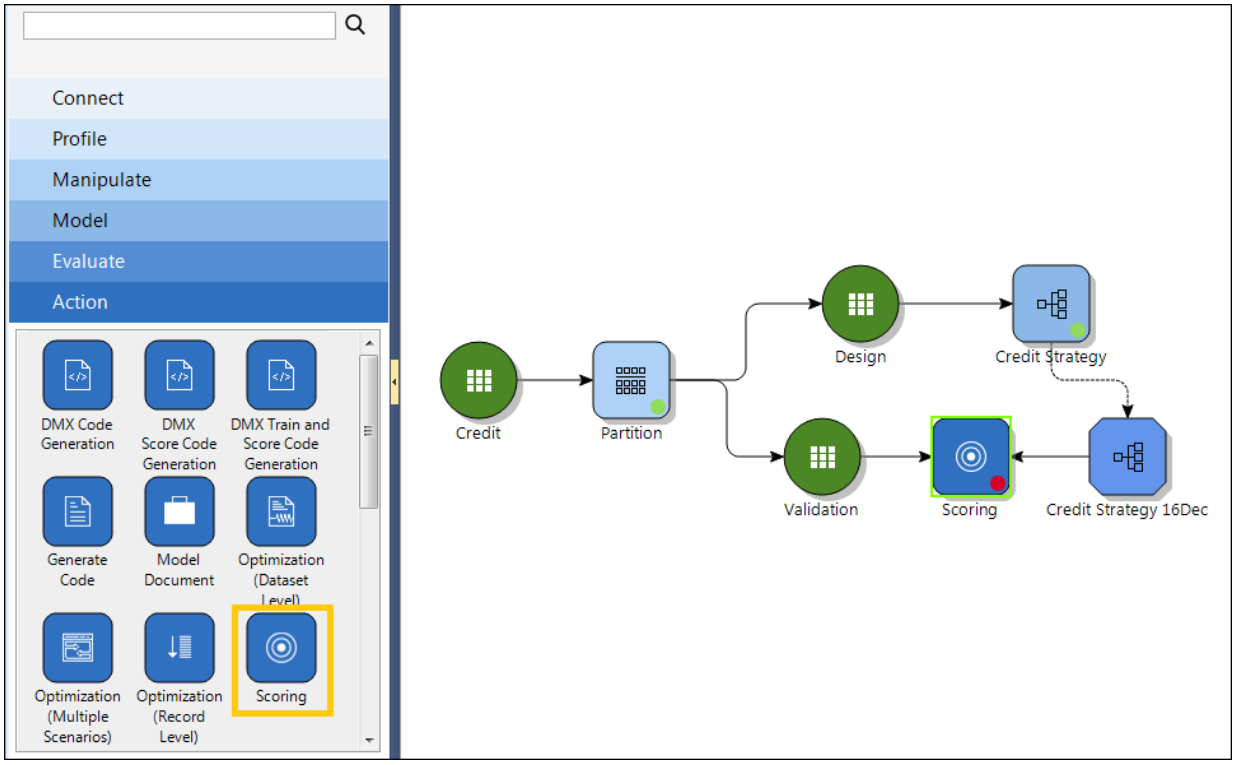
For general information about the notion and the purpose of strategy scoring, see the topic Scoring a Strategy.

Strategy tree scoring is performed using the **Scoring** node from the **Action** palette.

Strategy scoring is applied to Strategy Tree instances, since it is important to know what the state of the strategy tree was at the time of scoring. If you apply scoring to a Strategy Tree *process* node, a model instance will be automatically saved before scoring, and the scoring will be applied to the instance.

To score a strategy, drag the **Scoring** node from the **Action** palette onto the Workflow canvas and connect the Strategy Tree node and the input dataset to it. If the Strategy Tree node is a process node (rather than a model instance node), you will be prompted to create a model instance first, in which case you need to specify the name for the tree instance.

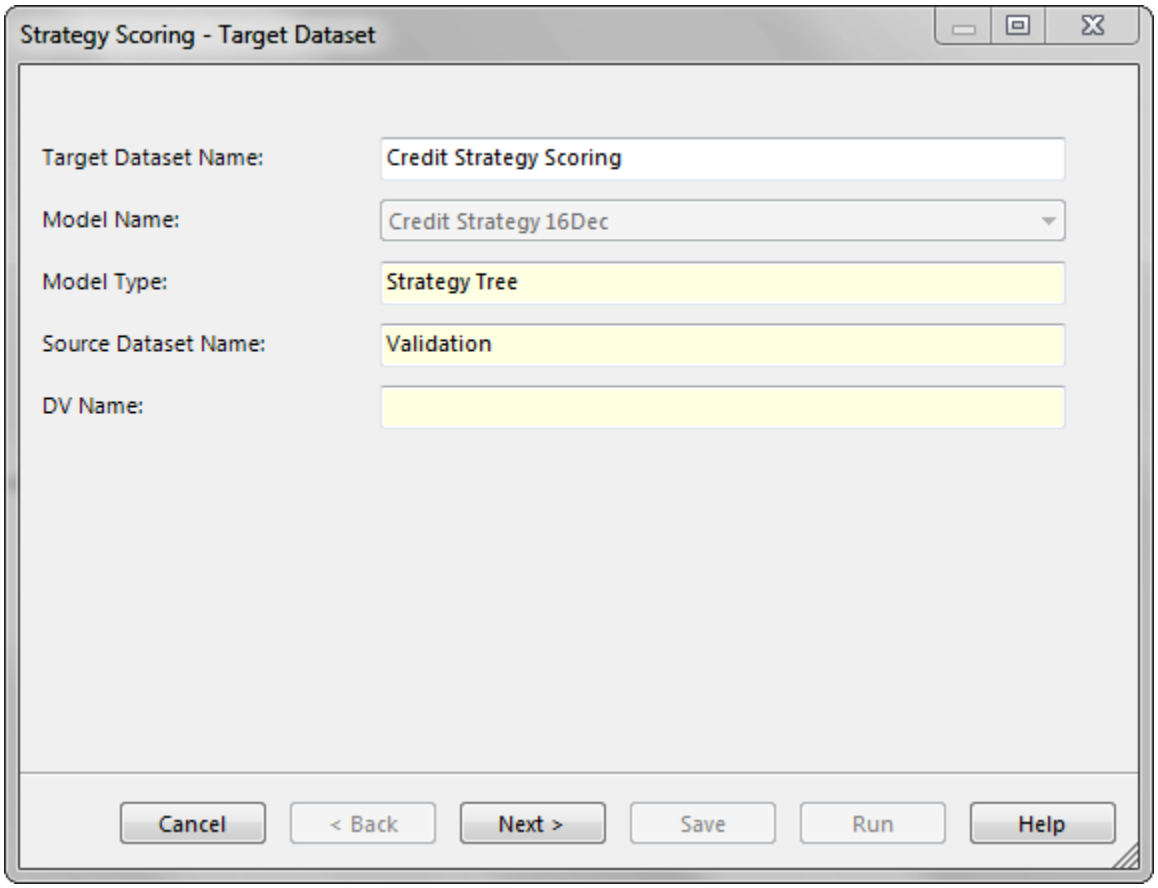
Alternatively, right-click anywhere on the workflow canvas and select the context menu command **Action --> Scoring**. This will create a new Scoring node. Connect the input dataset and the Strategy Tree to it.



Double-click the **Scoring** node or right-click on it and select **Modify** to open the wizard.

Specify the name for the output dataset.

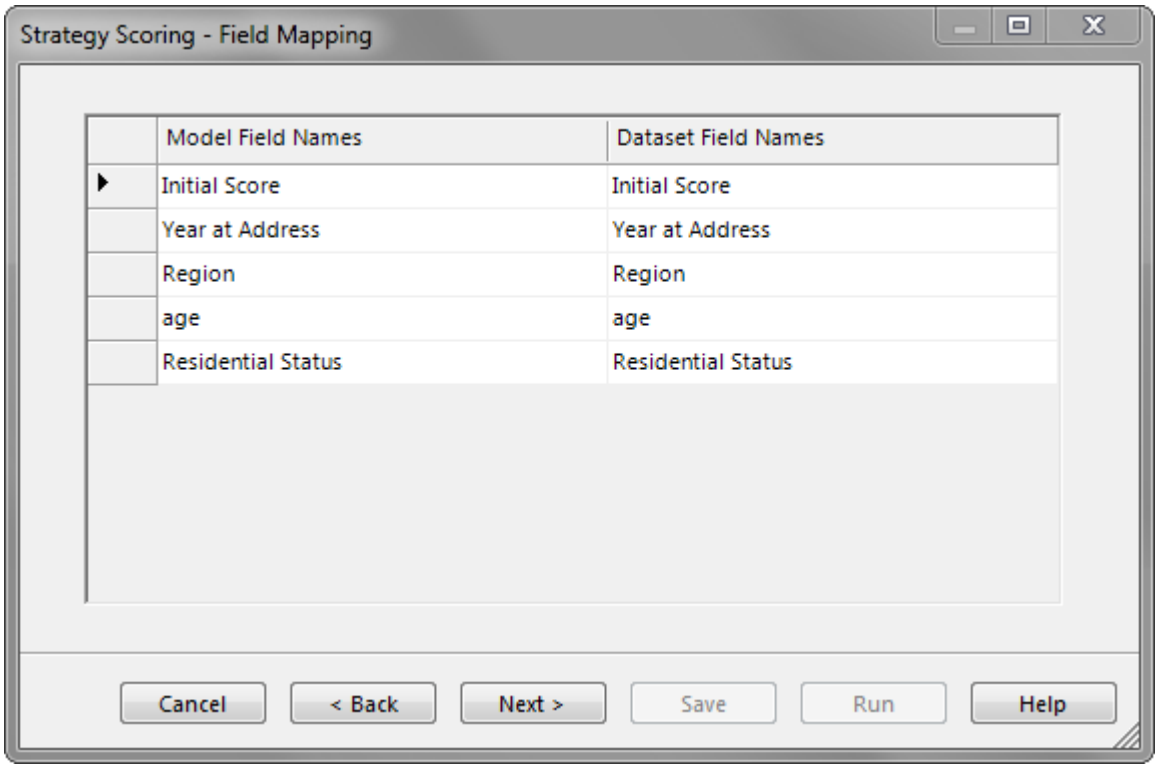
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Proceed to the topic Score Strategy: Field Mapping.

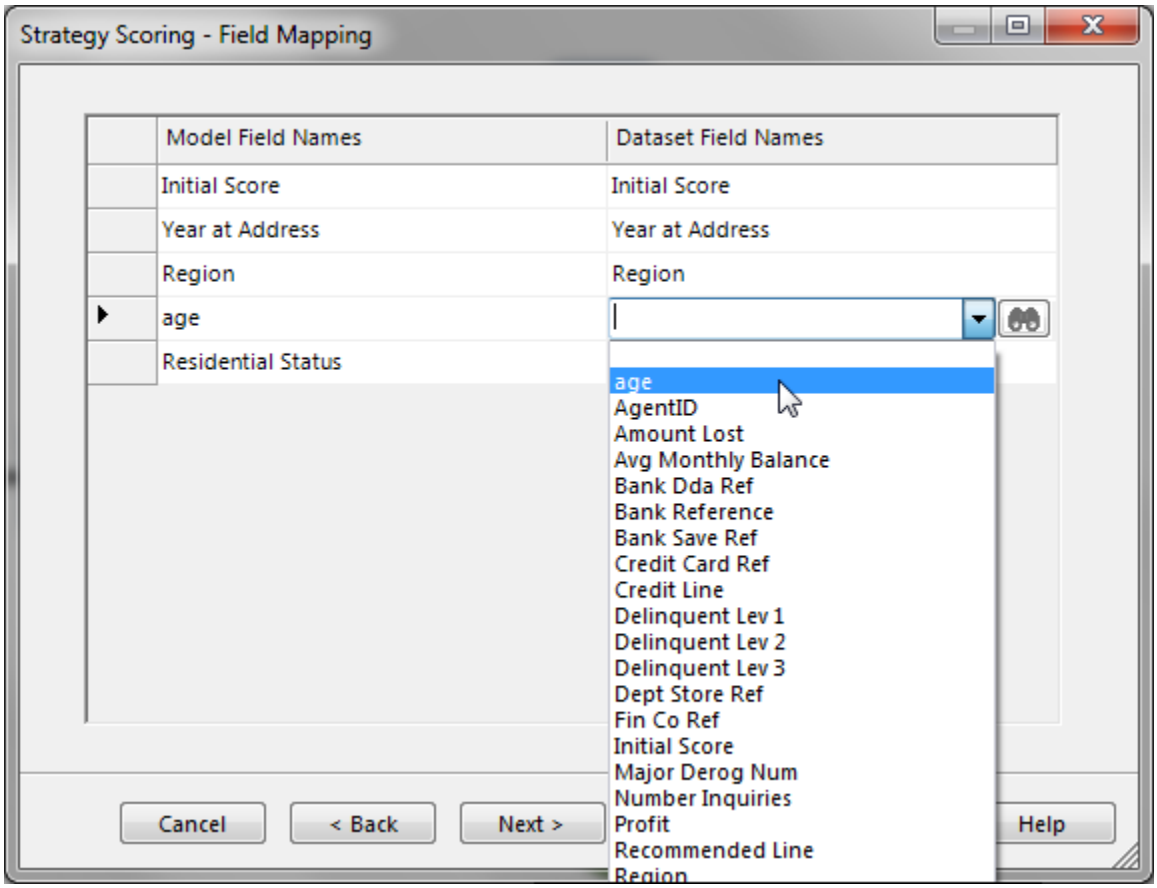
In the general case, the field names in the input dataset for strategy scoring may be different from those in the original design dataset used to build the strategy tree. This step of the wizard allows you to map the fields if necessary. If identical or similar field names are found, they are matched automatically.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If there is a dataset field that is not mapped or is mapped incorrectly, click in the corresponding cell and select the right field as shown below.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

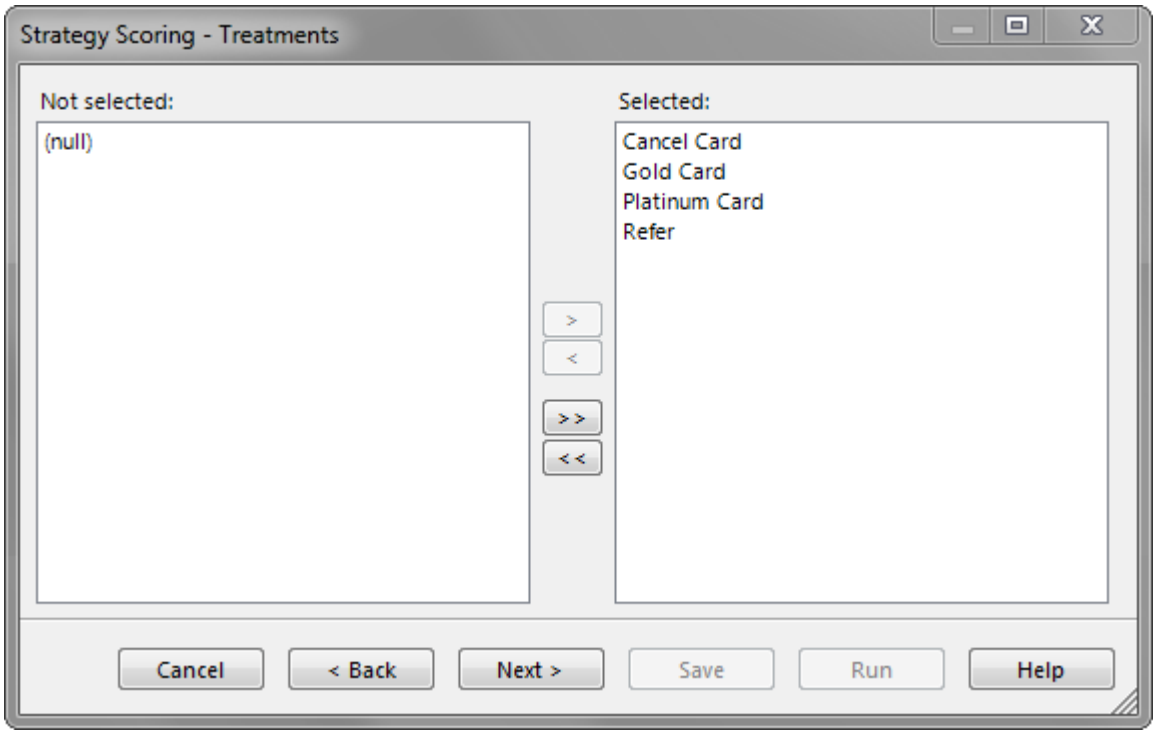


Click **Next** to proceed to the Treatments page to select the desired Treatments.

Select one or more treatments from the list of all treatments found in the tree. By default, all treatments are selected for list generation. You can move a treatment from one list to the other using arrow buttons or double-clicking on the treatment name.

Note: (null) is one of the options for treatments. It is not selected by default. If it is selected, then nodes without a treatment will also be written in the output with a NULL in the treatment column.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



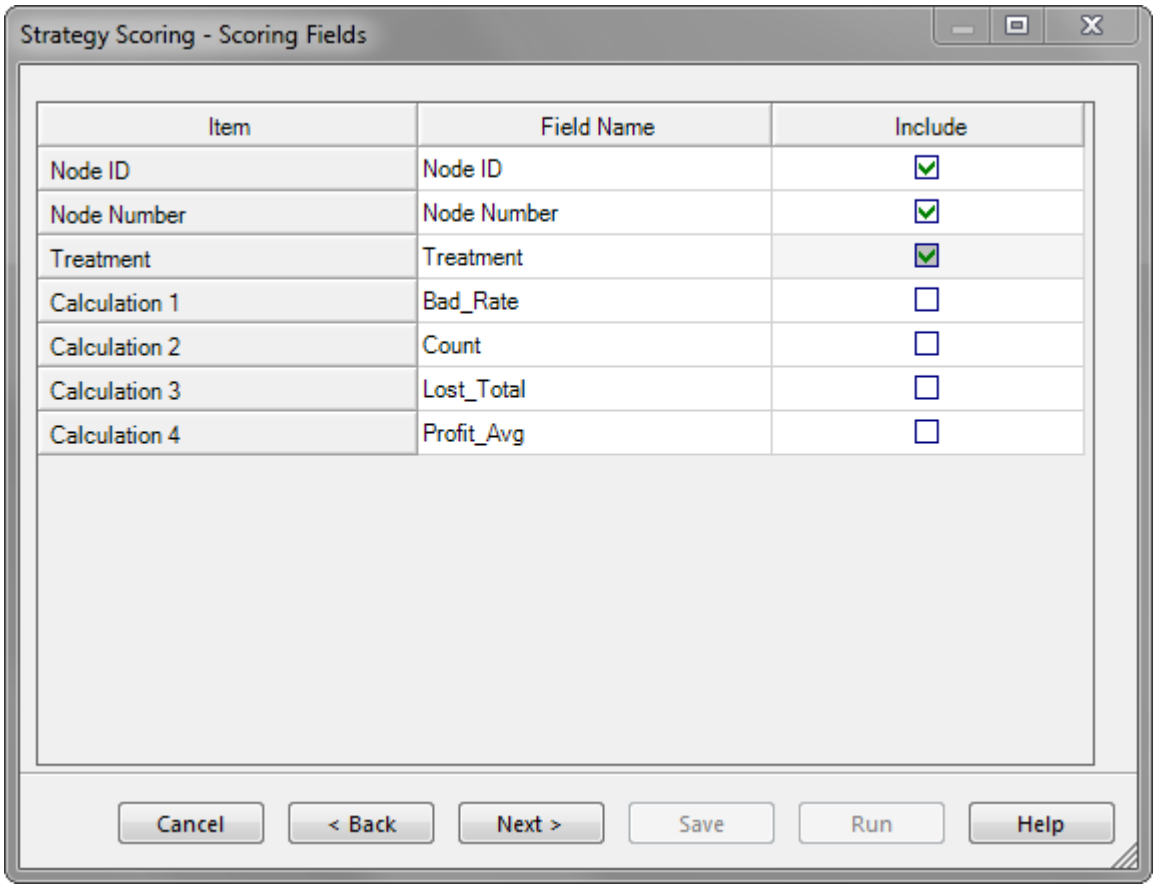
Click **Next** to proceed to the Scoring Fields page to select the new fields to be generated for the output dataset in addition to the Treatment field.

In the Scoring Fields page, select the fields to include in the scoring output dataset. The fields included by default are Treatment, Node ID, and Node Number.

Calculations are not included by default. Select the ones required in the output.

You can rename the fields if necessary. To rename a field, just edit it in the Field Name column.

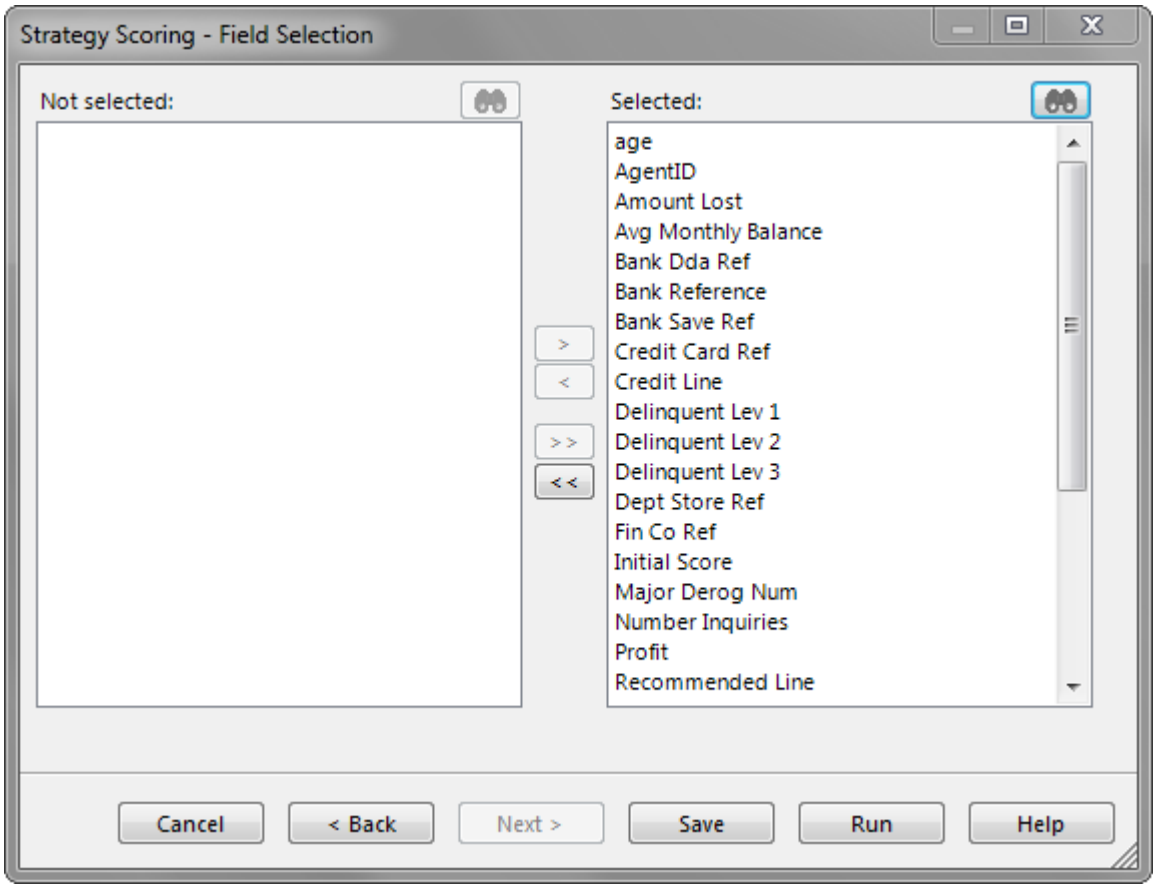
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



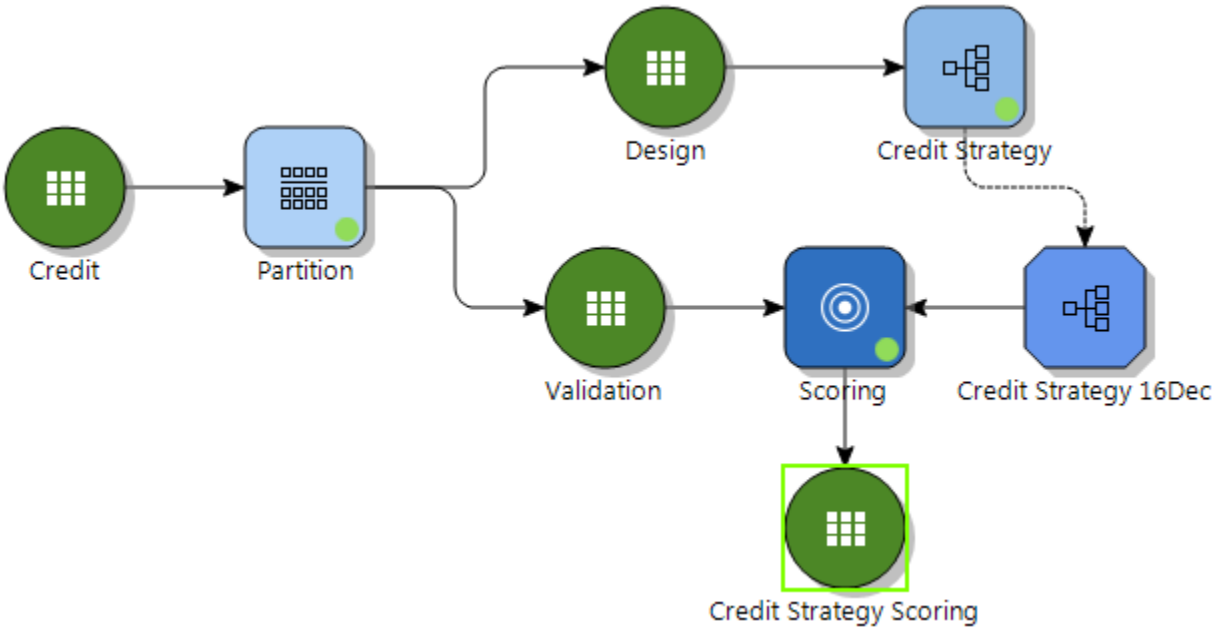
Click **Next** to proceed to the page Fields To Include In The Output to select the fields from the input dataset to include in the output.

At the last step of the Score Strategy Wizard, select the fields from the input dataset to include in the output. If you don't select any, the output will contain only the Treatment field and the new scoring fields you may have selected in the Scoring Fields page.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the operation, double-click the output dataset node to view the results.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Optimization in Strategy Trees

The Optimization feature in Strategy Trees allows you to find and assign optimal solutions (treatments) that maximize or minimize objective functions across the segments of interest identified by Strategy Trees.

The optimizer solves a ***linear programming problem*** maximizing or minimizing a metric while meeting constraints on other metrics. The optimization algorithm is CLP (COIN-OR LP) - an open-source linear programming solver. For more information, see https://projects.coin-or.org/Clp/wiki/FAQ and https://projects.coin-or.org/Clp/wiki/SuccessStories.

Suppose your objective is to maximize or minimize a certain amount (for example, revenue or risk) that depends on treatments to be assigned to some segments of your data (for example, credit line increase amounts). The purpose is to find and assign the optimal values of the treatments that maximize (or minimize) your objective quantity.

The complete workflow is as follows:

1. Build a Strategy Tree to identify and mark the groups (tree nodes) eligible for your treatments.
2. Express the objective quantity in terms of the treatments to be assigned. The objective must be a linear function of the treatments. Specify whether it is to be minimized or maximized.
3. Specify constraints for the treatments (for example, each treatment value must be within a certain range).
4. Run optimization. As a result, optimal treatment values will be assigned to the marked segments (tree nodes). These values provide the maximum (minimum) of your objective quantity.
5. (Optional): Add a new field with assigned optimal treatments to your dataset in two easy steps.

Node Calculations in Strategy Trees can be used to specify formulas for the coefficients of the objective function and the constraints.

Optimal solutions are assigned directly to the strategy tree nodes representing the eligible groups.

After completing optimization, you can easily add a new field to your dataset that indicates the optimal solution (treatment) for each record.

See the Example Workflow topic for details.

Example Workflow

For general information about the optimization feature in Strategy Trees, see the topic Optimization in Strategy Trees. The present topic describes an example workflow of a typical optimization task, which can be used as a tutorial.

Example Task Overview

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

A dataset with credit account data is used in this example. The main fields are: Credit Line, Initial Score, Delinquent Level 1, Average Monthly Balance, Profit, Amount Lost, Region, Residential Status, age, Risk_Indicator ('OK' or 'Bad Debt').

Several groups (segments) of interest are identified in the segmentation process using a Strategy Tree. We are interested in assigning line increase amounts to these groups.

Purpose: Determine and assign optimal line increase amounts to the identified groups providing maximum revenue under specific constraints

Workflow Steps

1. Build a strategy tree to identify and mark the groups (nodes) eligible for line increase
2. Express the expected revenue in terms of the line increase amounts
3. Specify the coefficients of the total revenue function
4. Specify the constraints for the line increase amounts
5. Run optimization. As a result, optimal line increase amounts will be assigned to the marked segments
6. (Optional): Add a new field with assigned optimal treatments to your dataset in two easy steps

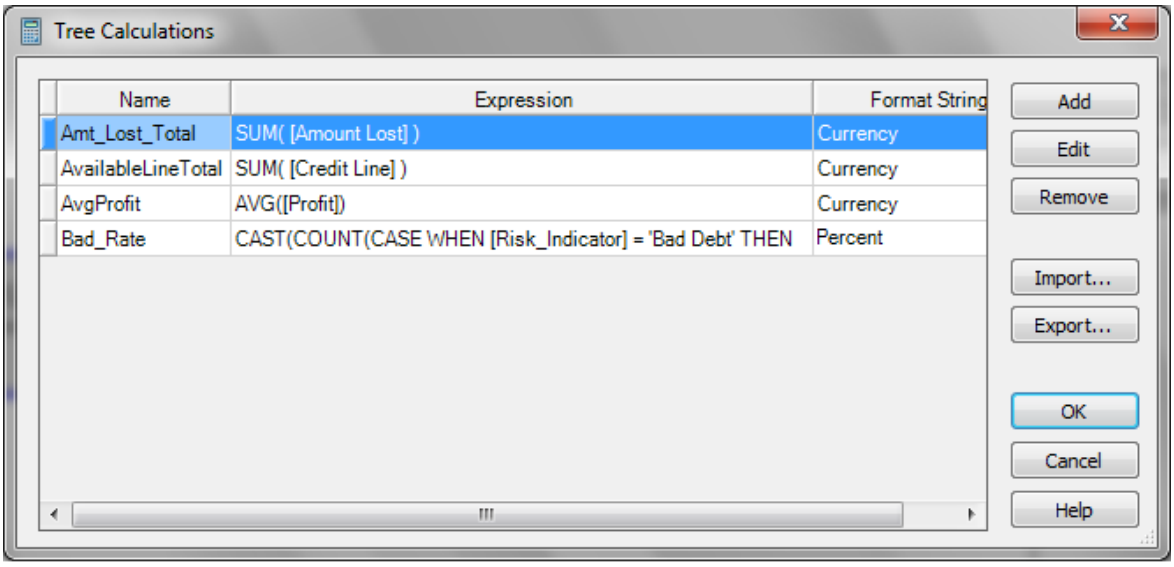
Step 1: Identify groups eligible for line increase

Insert a strategy tree

- After inserting the root node, add the calculations relevant to your strategy. In our example, we add the following metrics:
    - Total Amount Lost: SUM([Amount Lost])
    - Total Available Credit Line: SUM([Credit Line])
    - Average Profit: AVG([Profit])
    - Bad Rate: <The fraction of accounts with status 'Bad Debt'>. Can be added using Helpers (Discrete Distribution) and displayed as Percentage.

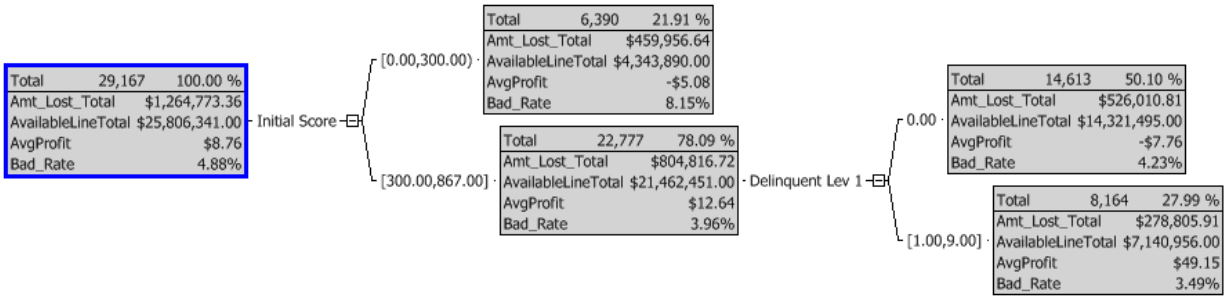
Total	29,167	100.00 %
Amt_Lost_Total		$1,264,773.36
AvailableLineTotal		$25,806,341.00
AvgProfit		$8.76
Bad_Rate		4.88 %

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



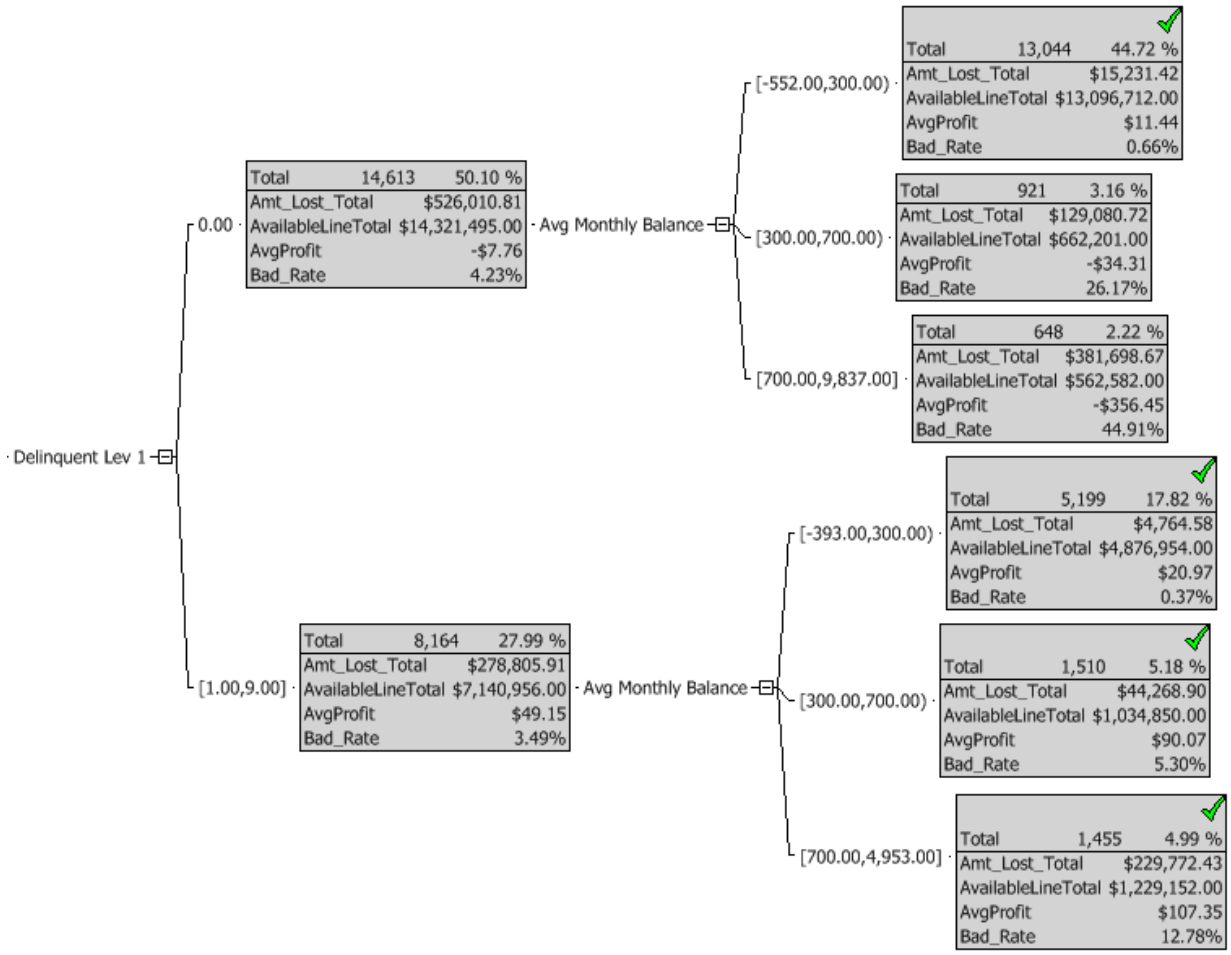
Build splits to identify the groups eligible for line increase

- Force Split on Initial Score with break at 300
- Ignore the group with Initial Score < 300, as it is not eligible.
- Select the group with Initial Score >=300 and split on Delinquent Level 1



- Split each of the resulting leaves on Average Monthly Balance
- Look at Bad Rate in each leaf. Select and tag only leaves with Bad Rate < 0.13 (13%).
- The four tagged nodes represent the groups eligible for line increase.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

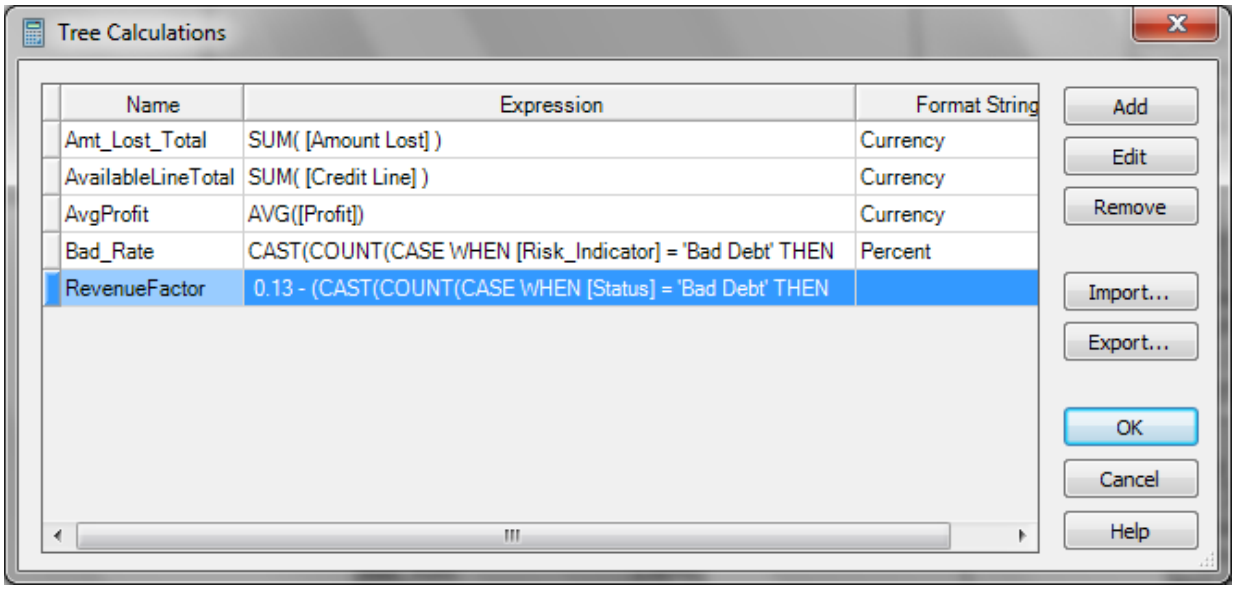


We have Group 1 with Line increase X(1), Group 2 with Line increase X(2), Group 3 with Line increase X(3), and Group 4 with Line increase X(4).

Step 2: Express the revenue in terms of the line increase amount

- The unknowns (optimal treatments to assign):
  - X(i) is the line increase to be assigned to every account in Group # i (i = 1, …, 4).
- The expected revenue for each group must be a linear function of the line increase assigned to it: R = K * X. Determine the factor K at this stage. The expected revenue for each segment (taking losses into account) can be expressed, for example, as
  - (0.13 * Line Increase) – (Bad Rate * Line Increase) = X * (0.13 – Bad Rate)
  - Thus, Revenue Factor = K = (0.13 – Bad Rate)
- Enter the Revenue Factor as a Node Calculation (use the existing expression for Bad_Rate)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Step 3: Specify the coefficients of the total revenue function

- The calculated values of RevenueFactor are shown in the tree nodes
- The total revenue is the sum of RevenueFactor(i)*X(i) for the four tagged groups i=1,…,4. By adding the RevenueFactor calculation to each node we specified the coefficients of the objective function (total revenue) to be maximized

Total	5,199	17.82 %
Amt_Lost_Total		$4,764.58
AvailableLineTotal		$4,876,954.00
AvgProfit		$20.97
Bad_Rate		0.37%
RevenueFactor		0.12

Step 4: Determine the Constraints

Determine the solution constraints. For example:

- Each line increase must be between $500 and $5,000:

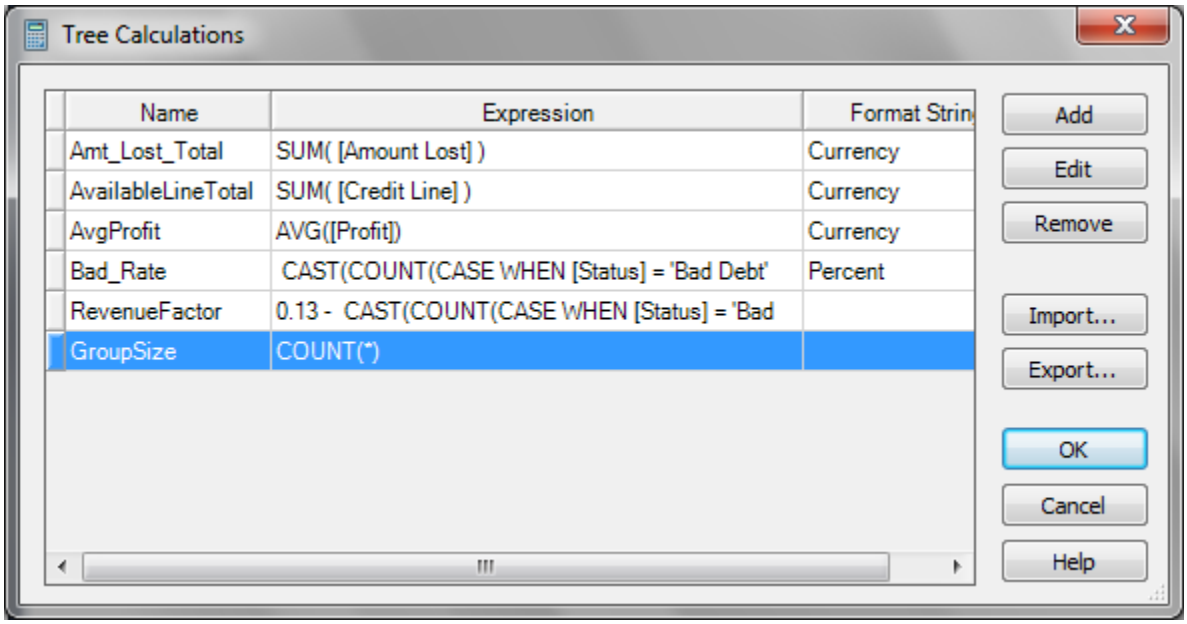
  500 <= X <= 5,000 (applies to both X(1) and X(2))

- the sum of all line increase amounts must not exceed $ 30 mln:

  SUM (Size of Group i) * X(i) (over i=1,…4) <= 30,000,000

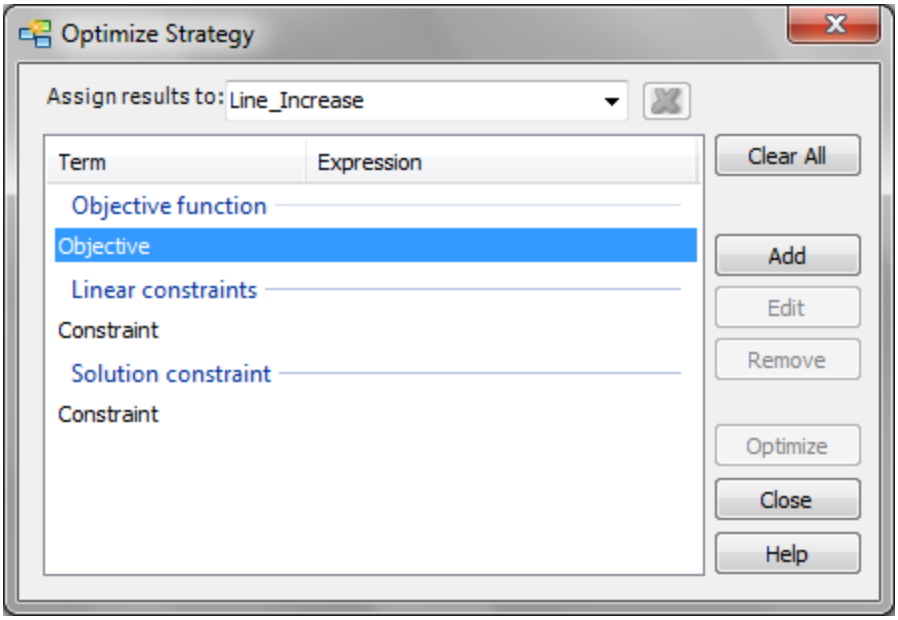
  (In this case, add Group Size as a Node Calculation, since it is a coefficient for this linear constraint)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



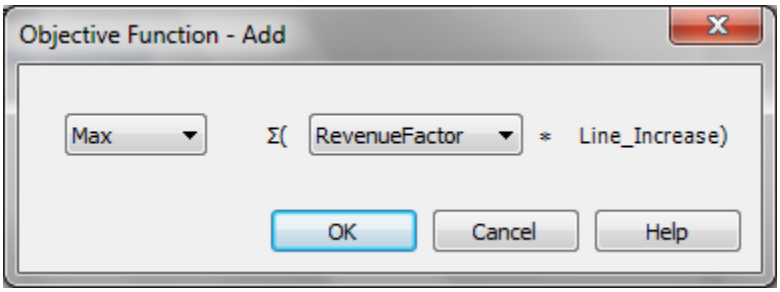
Step 5: Run Optimization Helper

- Select the menu command **Tools | Optimization Helper**.

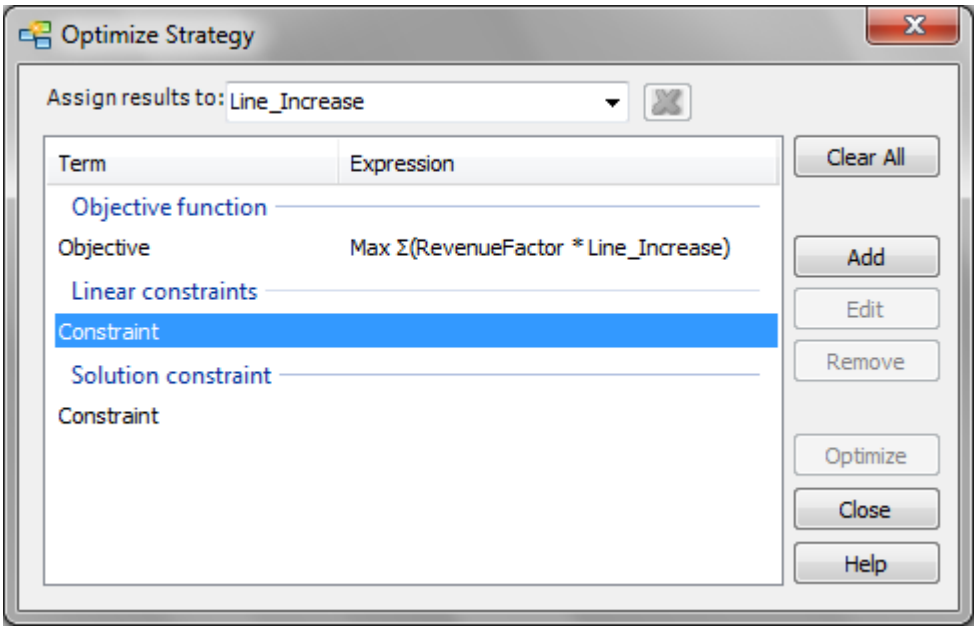


- In the Optimize Strategy dialog, enter the name of the treatment to assign.
- To choose your objective function, double-click on Objective, or just select it and click the **Add** button.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



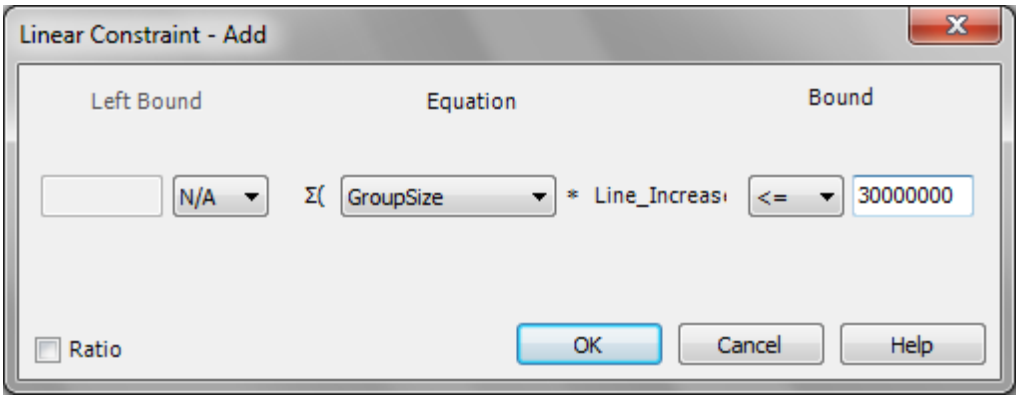
- Choose whether to maximize or minimize the objective function.
- Choose RevenueFactor from the list of existing node calculations to indicate the coefficients of your objective function. Click **OK**. The expression for the objective function will be displayed in the Optimize Strategy dialog:



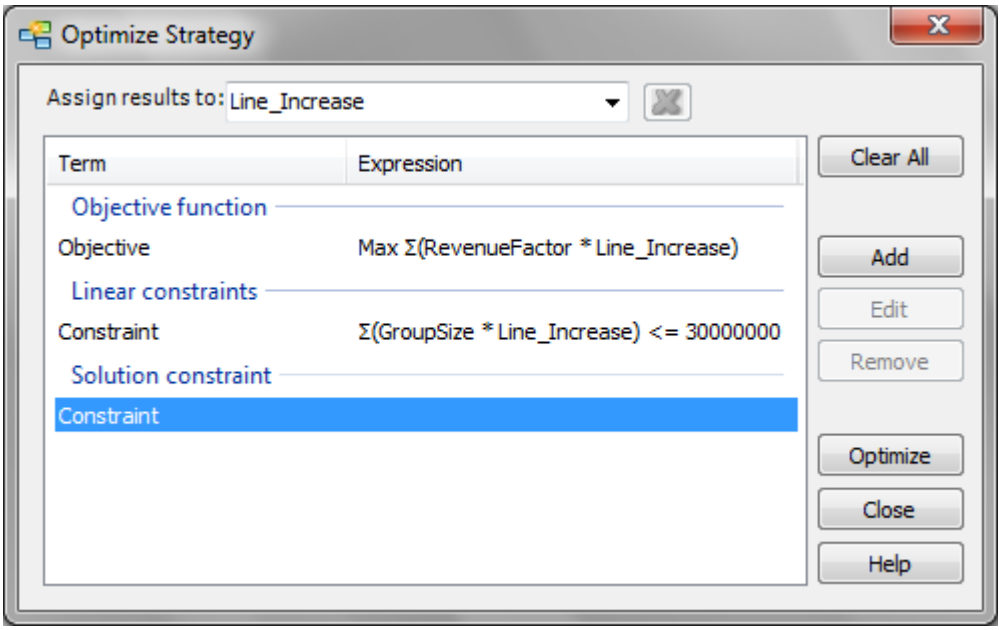
Add your constraints:

- Double-click on *Constraint* under the Linear Constraints line.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



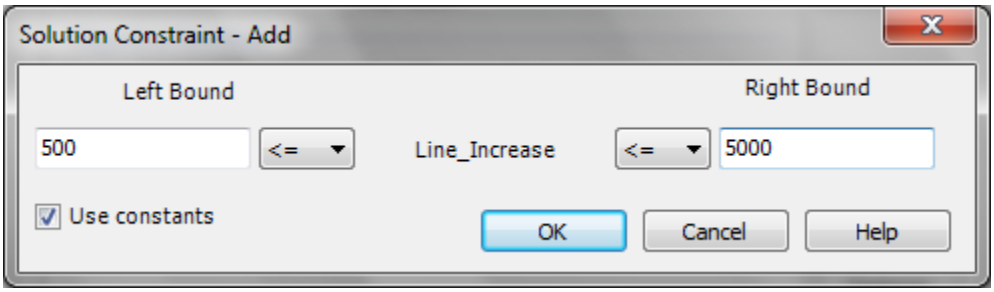
- Choose GroupSize from the list of existing node calculations to indicate the constraint coefficients
- Enter the right boundary for the linear constraint. Click OK. Your linear constraint will be displayed in the Optimize Strategy dialog:



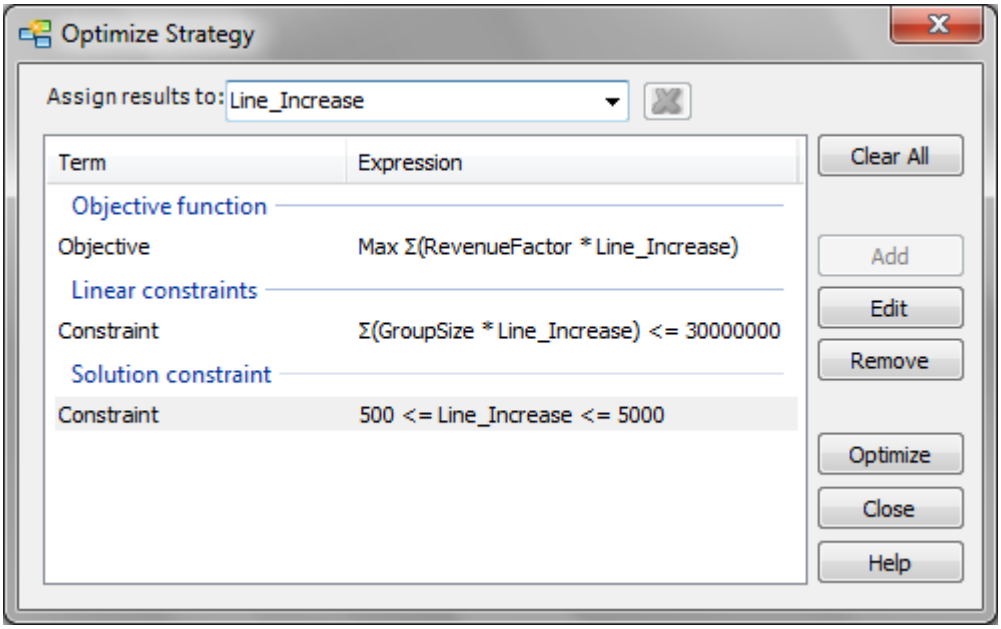
- To add a constraint for an individual solution, double-click on *Constraint* under the Solution constraint line (or select it and click Add).
- Select the Use Constants option and enter the boundaries for the individual solution constraint, for example:

  $500<= Line_Increase <= $5,000

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



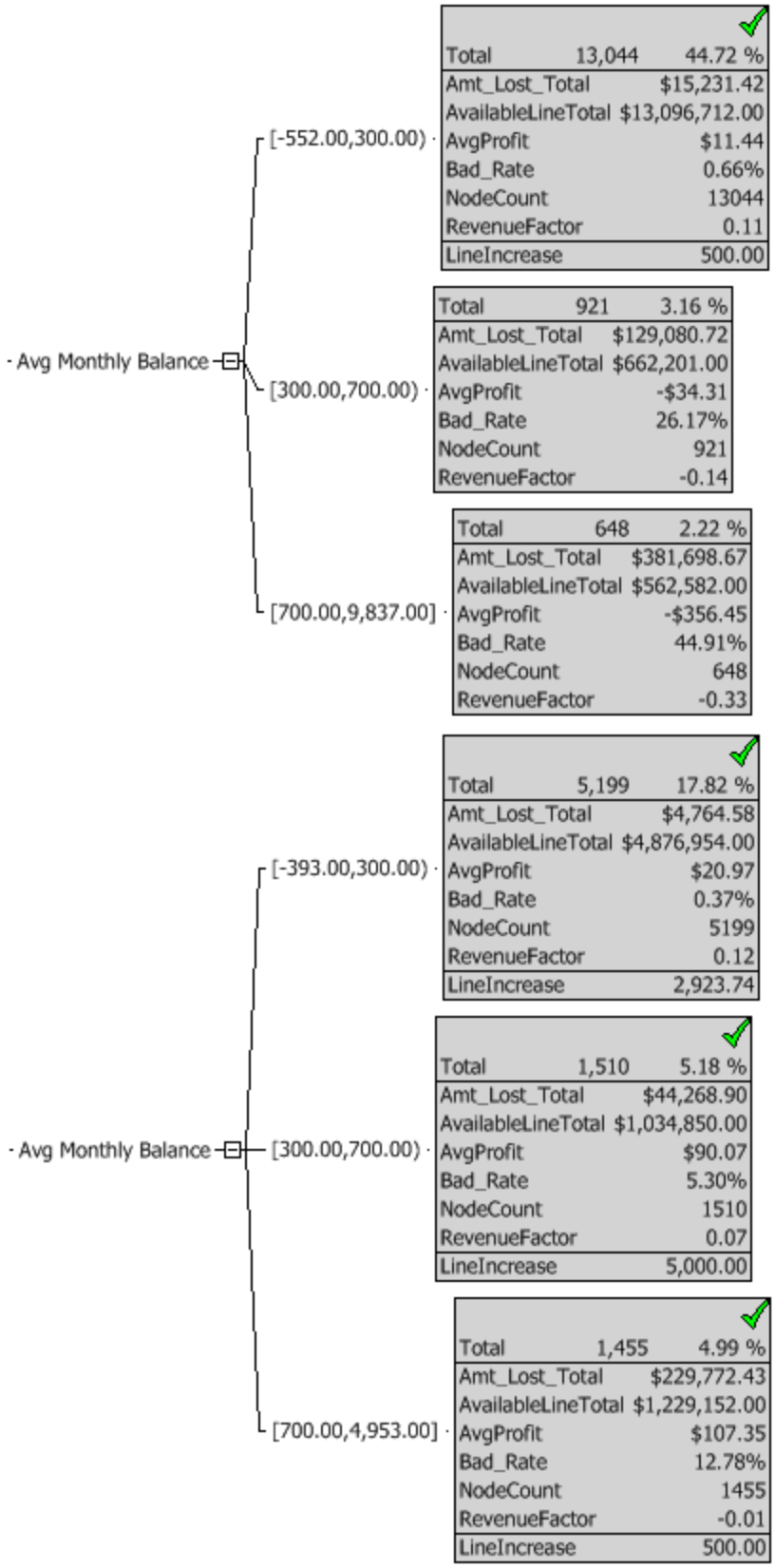
Click **OK**. You are now ready to run optimization.



Click the **Optimize** button. The optimal solution will be assigned to the tagged nodes.

In this example, the optimal Line Increase for $500.00 for Group 1, $2923.74 for Group 2, $5000 for Group 3, and $500 for Group 4:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

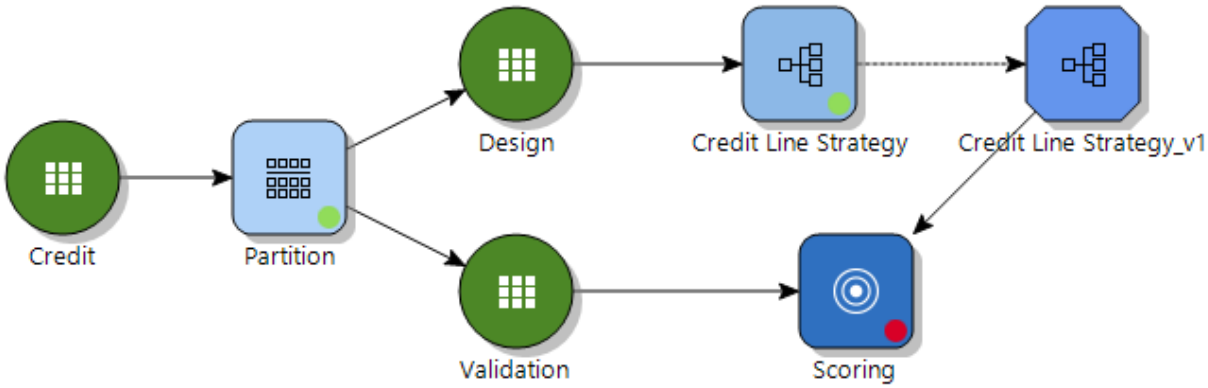


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Step 6 (optional): Add Optimal Treatments To Your Data

You can easily add a new field to your dataset that will indicate the optimal treatments found by the strategy optimization for every record. Just score the Strategy Tree on the new dataset the same way as you would score any other model.

Add the scoring operation to your workflow by dragging the **Scoring** node from the **Action** palette onto the workflow canvas and connecting the Strategy Tree node to the Scoring node. You will first be prompted to save a **model instance.** Then the instance will be connected to the Scoring node.



Double-click the Scoring node to open the wizard. In the Scoring Wizard, select the *Line_Increase* treatment. After the output dataset has been created, you can export it to a SAS System file or another format using the nodes from the Data Export palette.

See also: Record-Level Optimization

Cluster Analysis

Cluster analysis is a method for grouping objects with similar characteristics into respective categories. In data mining, a cluster analysis model is used as a strategic data analysis tool that groups data in a way to minimize the variability within clusters and maximize the variability between clusters.

Two algorithms are available when creating a cluster analysis model.

K-Means Algorithm

K-means is a simple, unsupervised learning algorithm that classifies a dataset into *k* clusters, where *k* is determined by the user.

K-means algorithm can be described as follows:

1. The initial centroids for the *k* clusters are calculated from the data.
2. Each record in the data is assigned to the cluster with the closest centroid based on the Euclidean distance between the data point and the centroid.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

3. When all the records are assigned, the centroids for each of the *k* clusters are recalculated.
4. Based on the new centroid calculated, the records may now belong to different clusters. Steps 2 and 3 are repeated until there is no change of records between the clusters.

For details on the K-Means clustering algorithm and other Cluster Analysis options, see the topic Cluster Analysis Models in the section "Mathematical Formulations".

Expectation Maximization (EM) Clustering

EM clustering algorithm computes probabilities of cluster memberships based on one or more probability distributions. The goal is to maximize the overall probability or likelihood of the data belonging to a particular cluster.

For details on the EM clustering algorithm and other Cluster Analysis options, see the topic Cluster Analysis Models in the section "Mathematical Formulations".

The Mathematical Formulations section also contains detailed description of the Goodness of Cluster metric and the cluster initialization methods (such as KKZ) used by the application.

Inserting Cluster Analysis

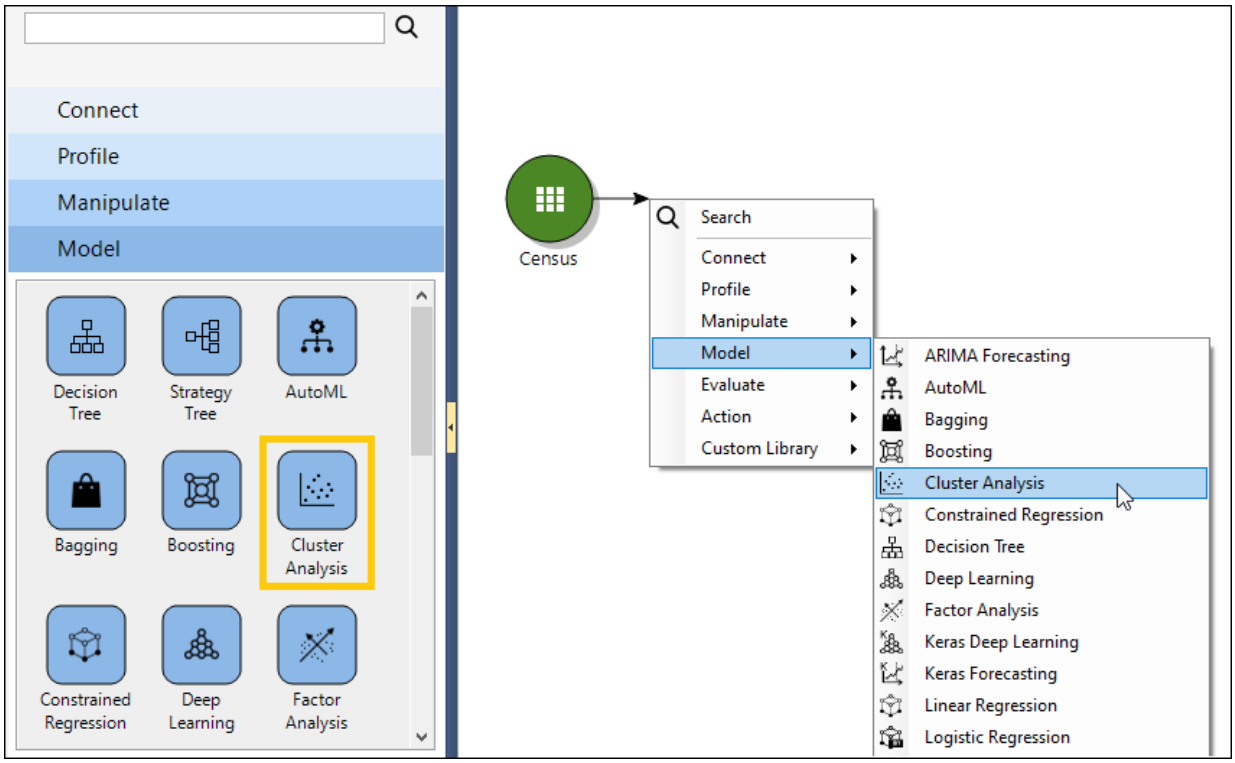
To build a Cluster Analysis model:

- Drag a connection arrow from the desired dataset node to any area on the Workflow canvas background. From the context menu that appears when you release the mouse button, select **Model --> Cluster Analysis** as in the example screenshot below. A new **Cluster Analysis** node will be created and the wizard will open.

OR

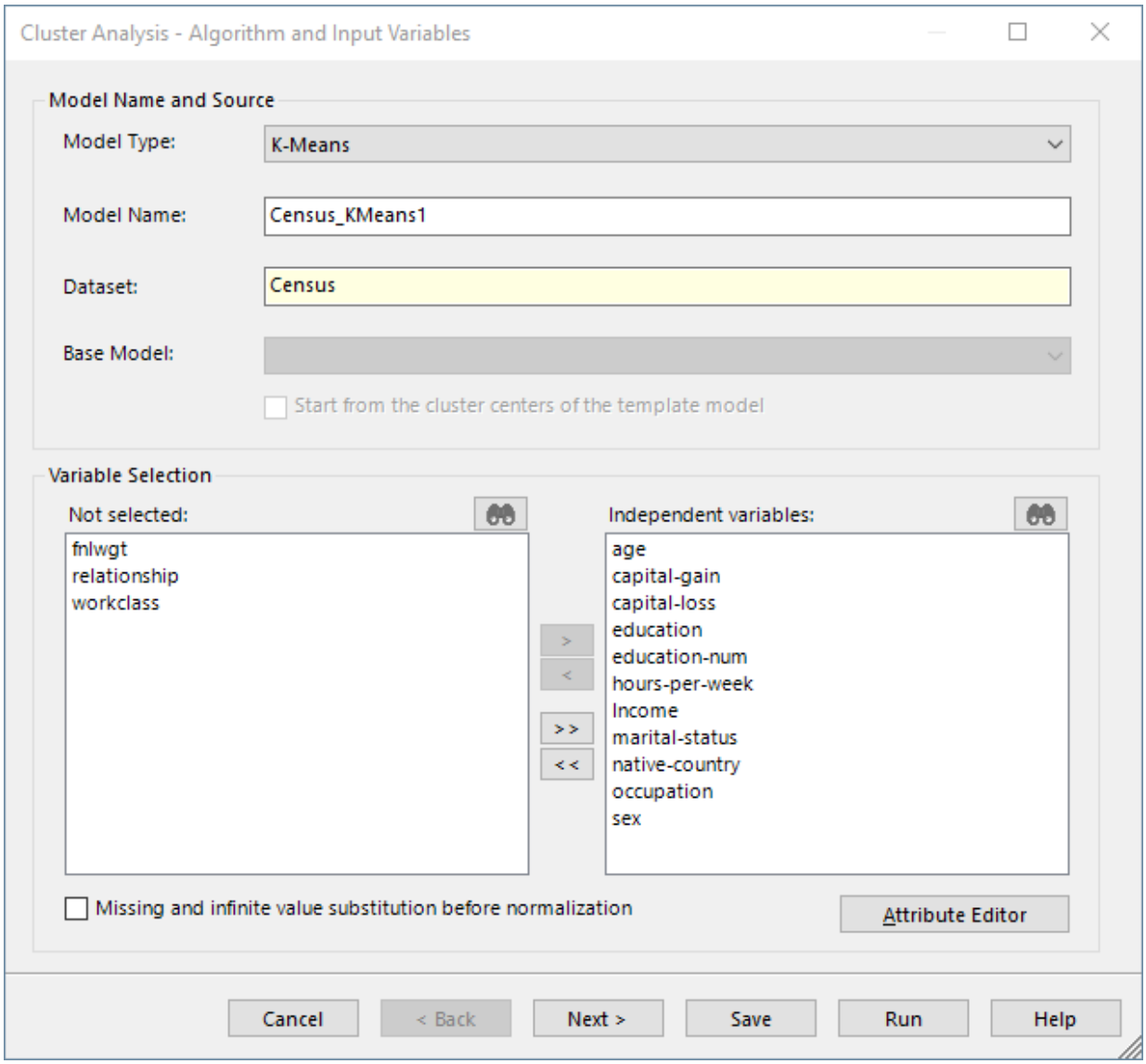
- Open the **Model** palette and drag the **Cluster Analysis** node onto the workflow canvas. Connect the input dataset to the new node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Then double-click the Cluster Analysis node or right-click it and select **Modify** to open the wizard.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Model Ty Select the desired cluster algorithm.

Model Name Specify the name for the new model as it should appear on the workflow canvas and in the Project Explorer panel.

Dataset The input dataset used to train the cluster model. This is the dataset currently connected to the Cluster Analysis node.

Base Moc Specifies the model to use as a template for the new one being inserted. The training parameters of the template model (such as the dependent and independent variables, etc) will be used when creating your new model. For the template, you can choose any of the models available in the project.

If you start this wizard when no model is currently in focus, then no template will be used (the default template model is set to [None]). If any model is currently being viewed, then this model will be selected by default as the template model.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The template model does not have to be the same model type as the model being created. For example, when creating a cluster model, you can pick a linear regression model to use as a template. Any parameters that are not relevant to the new cluster model will be disregarded.

If you select a template model, the option *Start from the cluster centers of the template model* becomes available. In that case you would normally choose a different dataset than the one the template model was trained on.

Select the variables for the model.

You can move the variables to and from the 'Selected' list using the arrow buttons or double-clicking their names. To select more than one variable in the lists, use mouse dragging or left-click while holding down the SHIFT key or the CTRL key. If there are hundreds or thousands of variables, it is convenient to use the **Field Finder** opened by clicking the binoculars button  .

Adjust Model Attributes

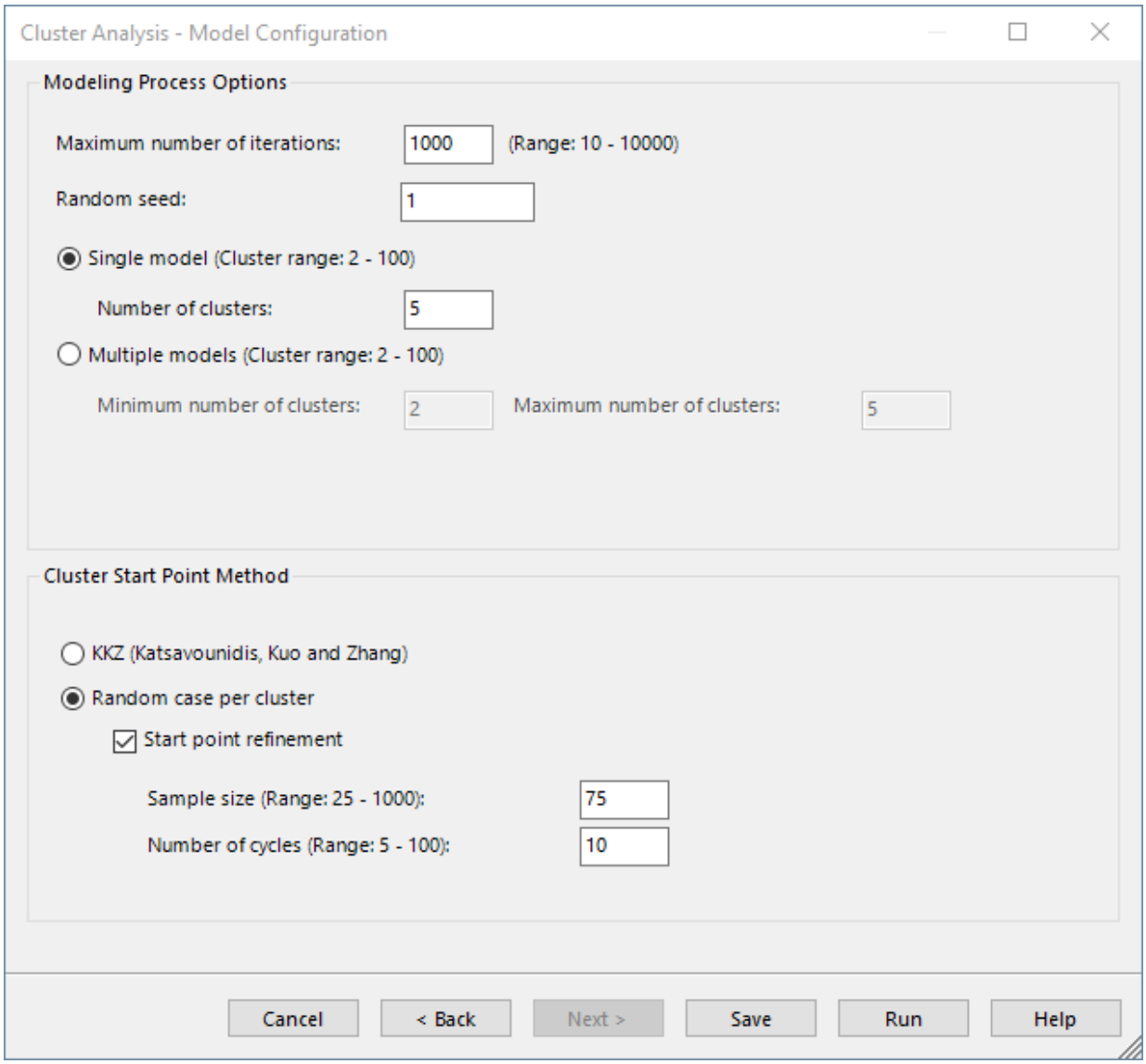
To change the attributes of the variables such as normalization type, the way missing values are treated, etc., click the **Attribute Editor** button. The Cluster Analysis Attribute Editor is the same as the Predictive Model Attribute Editor. See the topic Predictive Model Attribute Editor for more information.

Define Substitution and Normalization Order

Variables included in the model are normalized according to their Usage defined in the Attribute Editor. The automatic substitution of missing and infinite values (if any) is also defined there. By default, normalization is performed **before** the substitution of missing and infinite values. If any of the variables to be included in your model contain missing and/or infinite values, and if you prefer to substitute them before normalization, select the option "**Missing and infinite value substitution before normalization**" in this wizard page. Alternatively, you can use the Variable Transformations node to perform the necessary substitutions before building the model.

Click Next to define the model configuration. At this step, you can choose the modeling process options and the start point method options.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Maximum number of iterations

This is the number of times the algorithm calculates the error, adjusts the weights, and feeds the error back through the algorithm.

Random Seed

In the case of Random Case Per cluster (see the preceding step), records are chosen at random to be the start point for the centres of the clusters. If you want to insert and compare several Cluster Analysis Models, specifying the same random seed for all cluster models will ensure the repeatability of results.

Single Cluster Model

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

A single cluster model is created. Enter the desired number of clusters in the Number of Clusters field. The model will be trained to find the specified number of clusters. The default is 5. You can request from 2 to 100 clusters.

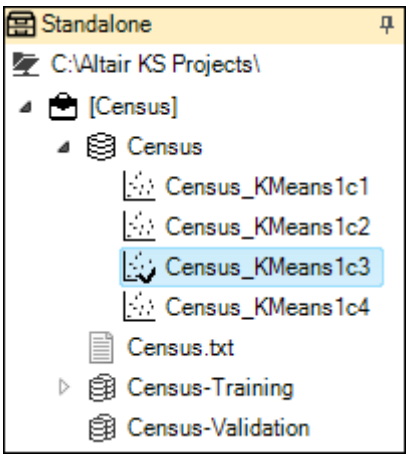
Note that, depending on the data, the cluster algorithm may not necessarily be able to find as many clusters as requested, so in some cases it may find fewer clusters than the number you specify in the wizard.

Multiple models

If the Multiple Models option is selected, a range of cluster models will be created automatically. The first model created will have the number of clusters specified in the ***Minimum number of clusters*** field. The number of clusters will be incremented by one for the next model and so on until the last model is created with the ***Maximum number of clusters***. For example, if Minimum number of clusters = 2 and Maximum number of clusters = 5, four cluster models will be created, with 2, 3, 4, and 5 clusters, respectively.

Minimum number of clusters	Minimum number of clusters to be discovered (in the first model). Starting from this number, the number of clusters increases by one for each subsequent model up to the Maximum number of clusters.
Maximum number of clusters	Maximum number of clusters to be discovered. The model with the maximum number of clusters will be the last model created.

When all requested models are built, they are compared and the best one is selected based on the goodness of clusters metric (see the Cluster Analysis Models section in Mathematical Formulations). The best model has the highest value of *Pseudo F Statistic* in the model results view.



Note that, depending on the data, the cluster algorithm may not necessarily be able to find the requested number of clusters, so in some cases it may find fewer clusters than the number you specify in the wizard.

Cluster Start Point Method Options

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

KKZ (Katsavounidis, Kuo and Zhang)

Optimizes the distance between the seed clusters for inter-cluster separation. This method utilizes the sorted pairwise distances between the cluster centroids.

For more details about the method, see the topic Cluster Analysis Models in the section "Mathematical Formulations".

This method was first described in I. Katsavounidis, C. Kuo, and Z. Zhang, "A new initialization technique for generalized Lloyd iteration". *IEEE Signal Processing Letters*, **1**(10):144–146, 1994.

Random case per cluster

k records are chosen at random to be the start point for the centres of the clusters.

Start Point Refinement

The start point refinement algorithm attempts to find good starting points for the cluster centres.

k-means clustering is somewhat sensitive to the cluster centre start points, and it is not hard to find examples where the algorithm will find objectively poor clusters given bad start points. Start point refinement attempts to reduce the chance of this happening by choosing start points in a more intelligent way than random selection.

The algorithm begins by taking a number of subsamples of the data and running the standard clustering model on each of these subsamples (with random start points). The algorithm stores the final cluster centres of each subsample run as new datapoints, and ultimately runs a clustering algorithm on these new datapoints to find the centres it will use as the start points in the full clustering algorithm.

Sample Size

This parameter determines how many subsamples are defined. The appropriate value is very data-dependent but generally you should use a lower value when you have few variables and many records. Conversely, use a higher value when you have many variables and relatively few records.

Number of Cycles

The default setting of 10 is usually sufficient for the refinement process.

Click **Save** to save the settings, or click **Run** to save the settings and run the model immediately.

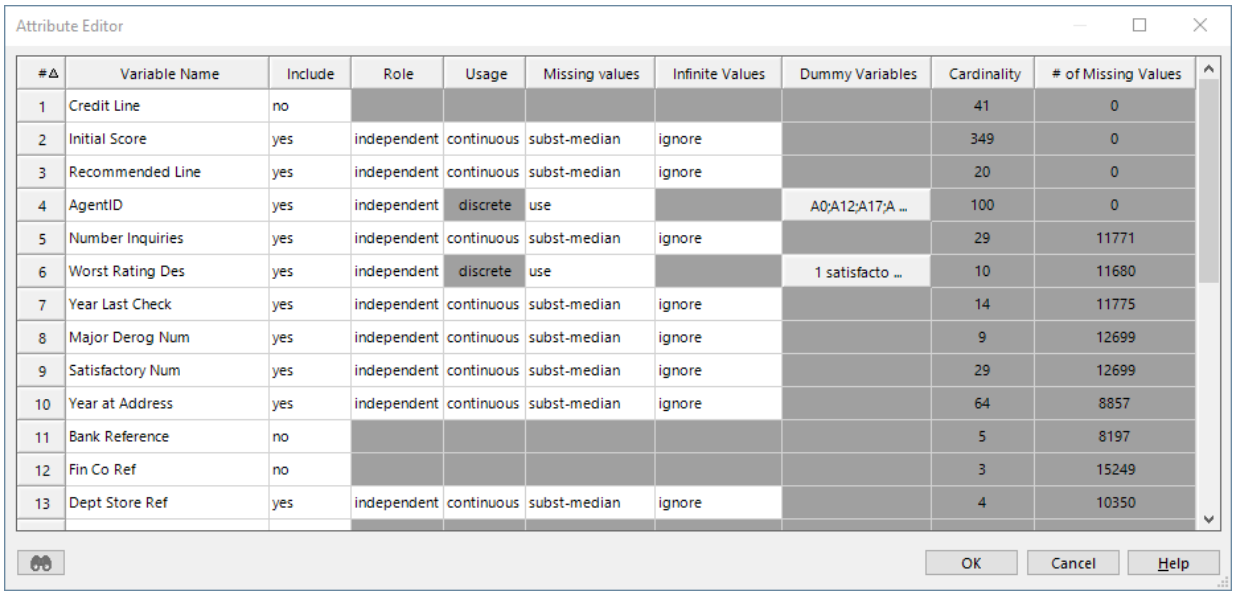
To view the resulting Cluster Analysis model, right-click on the Cluster Analysis node and select the **Open View** command from the context menu.

See the topic Cluster: Results View

Model Attribute Editor

The Attribute Editor for predictive models and cluster models allows you to modify variable attributes for the predictive model being inserted. The variables can be modified during the insert wizard. The attribute editor dialog is the same for all model types. To open the attribute editor, click the Advanced button in the Variable Selection step of the Model Wizard.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



#△	Variable Name	Include	Role	Usage	Missing values	Infinite Values	Dummy Variables	Cardinality	# of Missing Values
1	Credit Line	no						41	0
2	Initial Score	yes	independent	continuous	subst-median	ignore		349	0
3	Recommended Line	yes	independent	continuous	subst-median	ignore		20	0
4	AgentID	yes	independent	discrete	use		A0;A12;A17;A ...	100	0
5	Number Inquiries	yes	independent	continuous	subst-median	ignore		29	11771
6	Worst Rating Des	yes	independent	discrete	use		1 satisfacto ...	10	11680
7	Year Last Check	yes	independent	continuous	subst-median	ignore		14	11775
8	Major Derog Num	yes	independent	continuous	subst-median	ignore		9	12699
9	Satisfactory Num	yes	independent	continuous	subst-median	ignore		29	12699
10	Year at Address	yes	independent	continuous	subst-median	ignore		64	8857
11	Bank Reference	no						5	8197
12	Fin Co Ref	no						3	15249
13	Dept Store Ref	yes	independent	continuous	subst-median	ignore		4	10350

Variable Name The name of the variable.

Include Determines if a variable must be included in the model.

Role Determines the role of the variable in the analysis.

Dependent: In the case of Predictive Models, indicates that this variable is the dependent (target) variable that you want to explore. A model can have only one dependent variable (DV). Not applicable in the case of Cluster Models.

Independent: Indicates that this variable is an independent variable (IV) whose predictive value with respect to the dependent variable is to be determined.

Weight: Indicates that the variable is set as weight. The weight option is available only for nonnegative numeric variables. For details, see Weight in Predictive Models.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Usage

Depending on the selected option, values in a given variable can be left unchanged, scaled (normalized) in a certain way, or the variable may be transformed into a set of dummy variables. Note that the meaning of the continuous normalization type depends on the model type.

Continuous: This is the default normalization type for numeric variables.

In Logistic and Linear Regression models, continuous variables are not normalized.

In Multi-Level Neural Network models (MLN) and Cluster Analysis models, continuous variables are normalized as follows:

Normalized Value = (Original Value - Mean) / StdDev

Note: By default, the mean and the standard deviation used for normalization are calculated **before** the substitution of missing and infinite values described below. If any of the variables to be included in your model contain missing and/or infinite values, and if you prefer to substitute them prior to normalization, select the option "**Missing and infinite value substitution before normalization**" in this wizard page. Alternatively, you can use the Variable Transformations node to perform the necessary substitutions before building the model.

Ordinal: Linear interpolation into the interval [0,1]:

- Select a number of bins L (=10 by default);

- bin the data in L bins, and let x_0 < x_1 < x_2 < …< x_L be the cut-off points;

- let f_j = [weighted] fraction of records with x < x_j

Then the normalized value corresponding the original value x is defined by the rules

if x < x_0 then 0;

if x_L<= x then 1;

if x_j <= x < x_(j+1) for some j=0,1,..,L-1 then f_j + (x – x_j)/(x_(j+1) – x_j)*(f_(j+1) – f_j)

if x is null (missing), then it is replaced by the [weighted average] of the variable, and the above rule is applied

Discrete: Used for independent variables of type String. A variable with N unique values is mapped to N-1 indicator variables with values 1 or 0. If the original variable assumes the value V_k, the value of the k-th indicator variable is 1 and the others (if any) are 0.

Note that you can also generate indicator variables explicitly using the Expression Helper for the generation of dummy variables.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Missing Values

Only applies to independent variables. This column allows you to specify how missing values in your data are handled.

NaN values (which stands for "*Not a Number*") in continuous independent variables are treated the same way as missing values.

The missing value handling options are as follows:

use: Missing values will be treated like any other value. (This option is not available for continuous variables.)

ignore: Records containing missing values in this variable will be excluded from the analysis.

subst-median: The median value of the variable is substituted for every missing value. (This is the default option.)

subst-mode: The mode (most common value) of the variable is substituted for every missing value.

The following options are applicable for numeric or date/time IVs only:

subst-mean: The mean value of the variable is substituted for every missing value.

subst-min: The minimum value of the variable is substituted for every missing value.

subst-max: The maximum value of the variable is substituted for every missing value.

subst-zero: Zero is substituted for every missing value.

Note: The mean and the standard deviation used for the normalization of numeric variables (see the *Usage* section above) are calculated **before** the substitution of missing values. If you prefer to substitute missing values in your model variables prior to normalization, use the Variable Transformations node to perform the substitution before building the model.

Infinite Values

Only applies to numeric independent variables. This column allows you to specify how infinities are handled in numeric variables whose **Usage** is set to '*continuous*' . For more information on infinities and NaN, see Division and Infinity.

The options for handling infinities are:

ignore: Records containing infinities in this variable will be excluded from the analysis.

clip-to-finite: Infinities will be replaced with finite values. Positive infinities will be replaced with the maximum value of the variable; Negative infinities will be replaced with the minimum value of the variable.

Note: The mean and the standard deviation used for the normalization of numeric variables (see the *Usage* section above) are calculated **before** the substitution of infinite values. If any of the variables to be included in your model contain infinities, and if you prefer to substitute them prior to normalization, use the Variable Transformations node to perform the substitution before building the model.

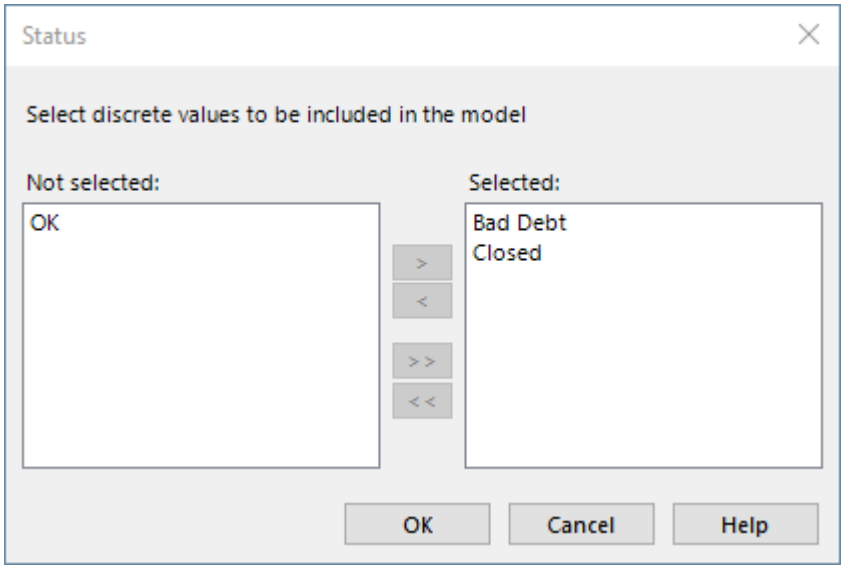
Dummy Variables

For any variable whose **Usage** is set to '*discrete*', this determines the representation of the variable as a set of *dummy variables*. Click in this column to select the *reference value* and the values to be

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

explicitly used by the model. For a variable with *N* unique values, *N-1* will be explicitly used in the training and one will be chosen as a reference value. By default, the largest category is set as a reference value (put in the *Not selected* list). To change the reference value, use the left & right arrow buttons to include or exclude values from the *Selected* list.

In a multinomial logistic regression, separate odds ratios are determined for independent variables for each dummy variable defined for the dependent variable, with the reference value excluded. In the example below, the Status variable has three values: 'OK', 'Bad Debt', and 'Closed'. The 'OK' category is set as a reference value:



Cardinality	Displays the cardinality (the number of distinct values) of each variable.
# of Missing Values	Displays the number of missing values for each variable.

Weight in Cluster Models

Weight in Knowledge Studio datasets and models is a generalized frequency weight (the weight values do not have to be integer). A weight can be set at the dataset level, but it can also be set and changed independently in the dependent models.

See the Weight topic for general information about weights and on setting weight at the dataset level.

Relationship between Weights in Datasets and Dependent Models

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The weights in a dataset and its dependent models can be changed or reset independently. When you toggle weight on or off in the dataset Overview Report, this does not affect the existing models based on this dataset.

By default, any new model created on the basis of a weighted dataset inherits the weight from it. More precisely, when inserting a model for a weighted dataset, the weight field defined in the dataset is automatically listed in the model's Attribute Editor as "weight" (in the Usage column). You can override this in the Predictive / Cluster Model Wizard and redefine the field as an analysis variable by changing the usage from "weight" to "independent" if necessary.

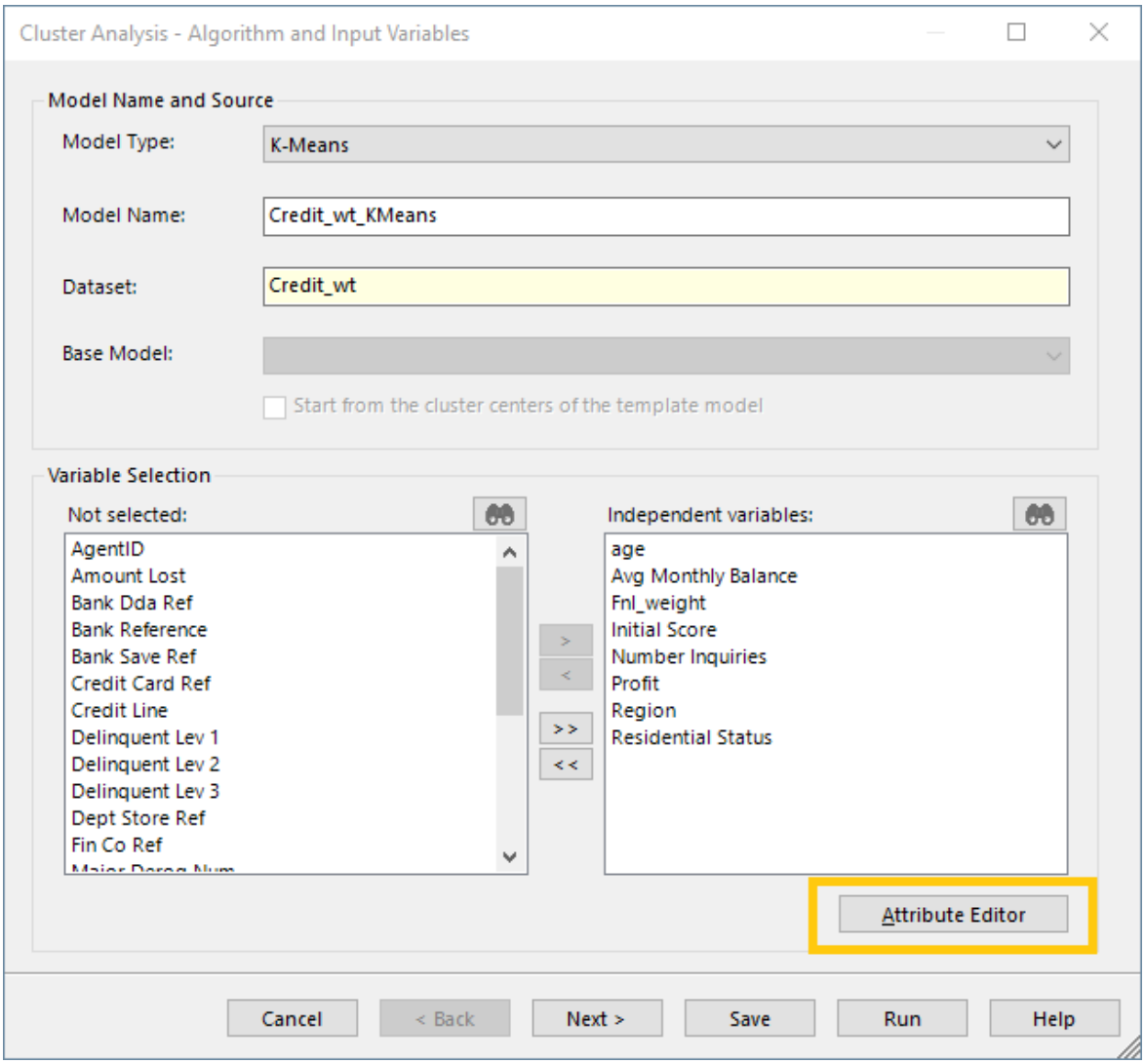
Weight in Cluster Models

It is assumed that the field you want to set as weight already exists in the parent dataset (although you don't have to set it as weight in Overview Report). See the Weight and Weight Helper topics for information on available tools for creating a weight field. If the parent dataset already has weight specified in Overview Report when you were inserting the model, the same weight is set in the model by default.

If you want to specify or change a weight for a Cluster Model, it must be done in the Cluster Model Wizard when inserting the model.

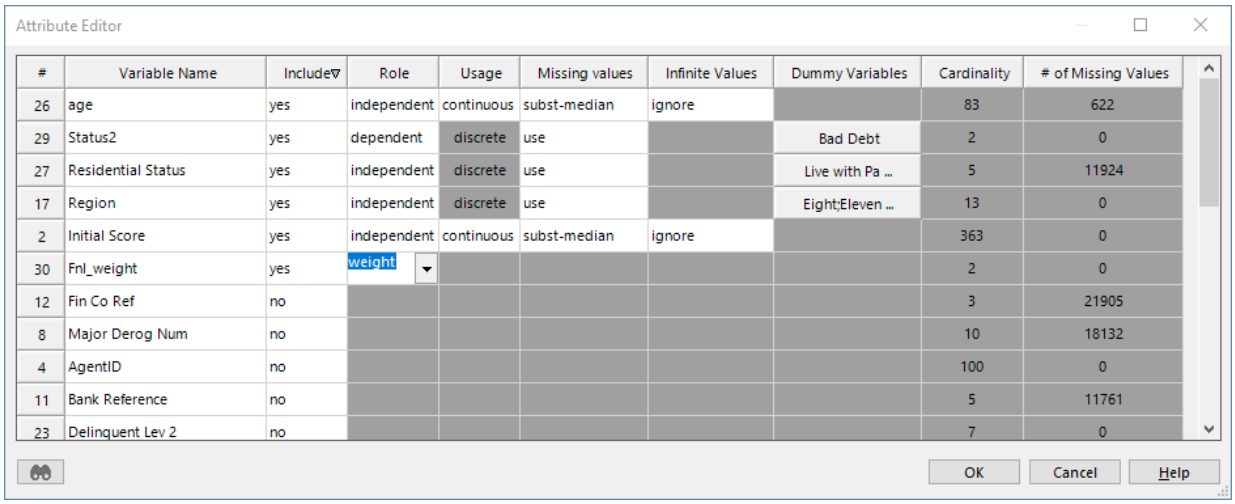
Click the **Attribute Editor** button in the second page of the Cluster Model Wizard when choosing the analysis variables for your model.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



When the Attribute Editor window opens, click in the ***Usage*** column for the variable you want to set as weight and select "**weight**" from the drop-down list. Note that the weight option is available only for nonnegative numeric variables. The weight variable does not have to be integer.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

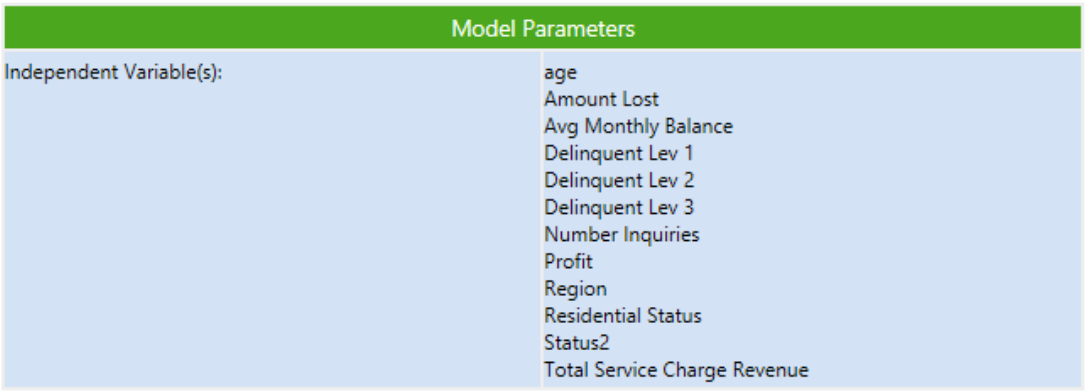


After the model is inserted, the information about the weight is available in the Parameters and Attributes tab of the model.

Viewing a Cluster Model

The **Results** view of a Cluster Analysis model provides the following information about the model:

- ***Independent Variable(s)*:** The list of variables included in the analysis.



- ***Training Results* (*Summary Statistics*):**
  - ***Sample size*:** The number of records used in the model
  - ***Number of clusters*:** The number of clusters found.
  - ***Maximum number of iterations.***
  - ***Goodness Of Clusters*:** This value is simply a normalized Calinski-Harabasz pseudo F statistic (see *Pseudo F Statistic* below). It is equal to Pseudo F divided by (*N* - *K*), where *N* is the total number of observations and *K* is the number of clusters.
  - ***Cubic Clustering Criterion* (*CCC*):** The Cubic Clustering Criterion (CCC) is a measure of the deviation of the clusters from the distribution expected for data sampled from a uniform distribution. The larger the values of the *CCC*, the better the solution. The maximum value of CCC across the hierarchy levels can be used to

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

indicate the optimal number of clusters in the data. For details, see the topic Cluster Analysis Models in the Mathematical Formulations section.

- ***R-squared:*** The proportion of variance accounted for by the clusters. For details, see Cluster Analysis Models in the Mathematical Formulations section.
- ***Pseudo F Statistic:*** Reflects the "tightness" of clusters as the ratio of the mean sum of squares between groups to the mean sum of squares within group. Larger values of the Pseudo-F indicate better clustering solution. Can be used as a criterion for the best number of clusters. For more details, see the topic Cluster Analysis Models in the Mathematical Formulations section.

Example:

K-Means Cluster Analysis

Created: Tuesday, June 16, 2020 22:37:29
Modified: Tuesday, June 16, 2020 22:37:30

Training Results	
Sample Size:	29,167
Number of Clusters:	5
Max Number of Iterations:	1000
Goodness Of Clusters:	0.252516932534584
Cubic Clustering Criterion:	666.690050732814
R^2:	0.502504324903349
Pseudo F Statistic:	7363.89875147384

- ***Cluster Distances:*** The Euclidean distances between the cluster centroids. Each cluster is denoted by K_i. ***Variance*** represents the intra-cluster measure of tightness.

Example:

Cluster Distances						
	1	2	3	4	5	Variance
1	0.000000	2.811545	2.805995	2.786381	2.742446	19.800759
2	2.811545	0.000000	1.888993	2.063174	1.933358	8.769325
3	2.805995	1.888993	0.000000	2.112013	2.008416	10.317936
4	2.786381	2.063174	2.112013	0.000000	1.368400	3.095536
5	2.742446	1.933358	2.008416	1.368400	0.000000	3.405902
Variance	19.800759	8.769325	10.317936	3.095536	3.405902	-

- ***Cluster Sizes***: The number of records in each cluster.

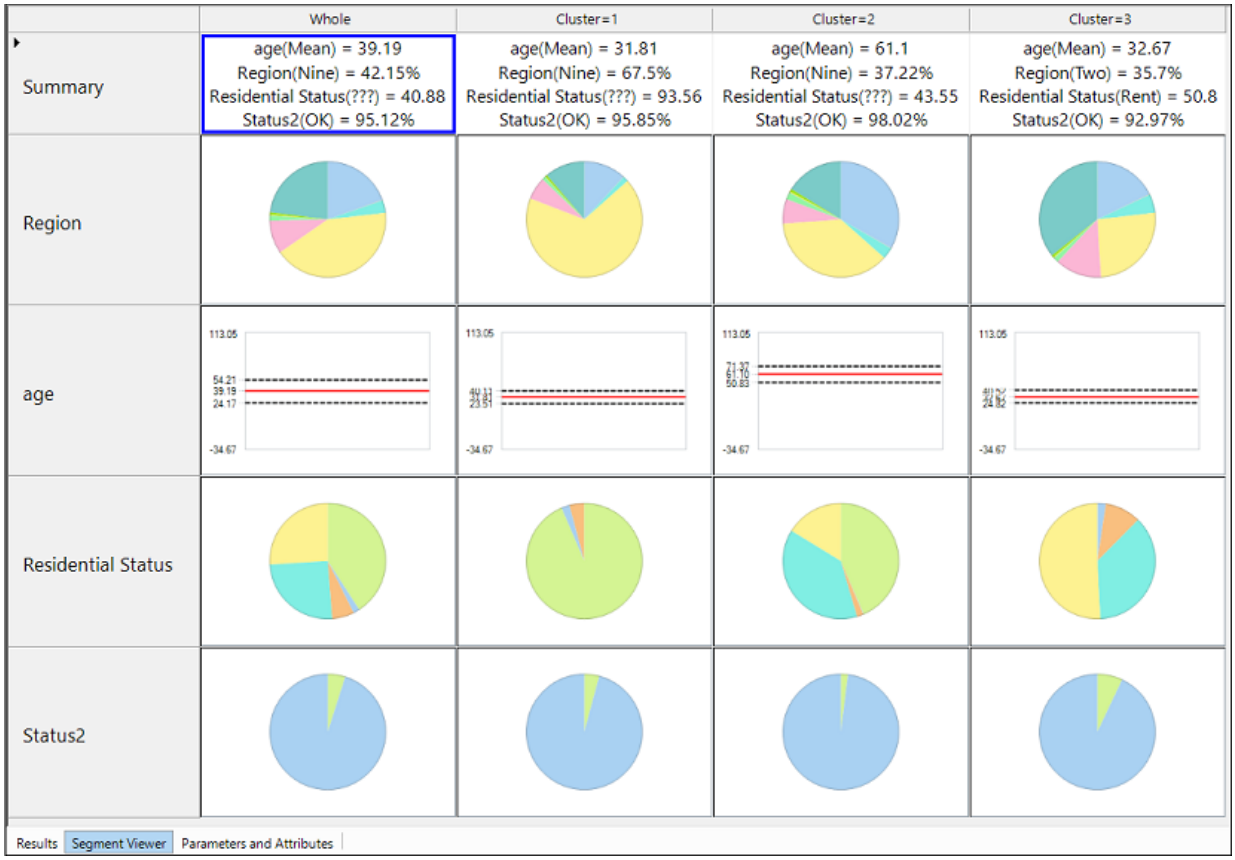
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Example:

Cluster Sizes	
Cluster	Record Count
1	1,310
2	1,851
3	2,560
4	7,199
5	16,247

See also: Cluster: Segment Viewer, Parameters and Attributes

The **Segment Viewer** tab of a Cluster Analysis model shows the distribution of each variable's data within each cluster in the form of charts. The charts are similar to those in the dataset segment viewer tab.



Columns represent clusters and rows represent variables. The first column represents the whole dataset.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The first row (Summary) contains short cluster descriptions, i.e., the summary of each variable's distribution within each cluster. For discrete variables, only the value that has the highest percentage within a given cluster is shown (for example, Status(OK)=95.85% in Cluster #1). For any continuous variable, its mean value over the given cluster is shown.

- The charts for discrete variables

  The distribution of a discrete variable is represented in the form of a pie chart. Double-click any chart to enlarge it and view it in a separate window. In this mode, you can change the chart type by right-clicking it and selecting the Gallery command.

- The charts for continuous variables

  The distribution of a continuous variable is displayed as a solid red line that represents the mean value of the variable over a given cluster, and dashed black lines represent the mean plus/minus standard deviation. Double-click any chart to enlarge it and view it in a separate window. In this mode, you can change the chart type by right-clicking it and selecting the Gallery command.

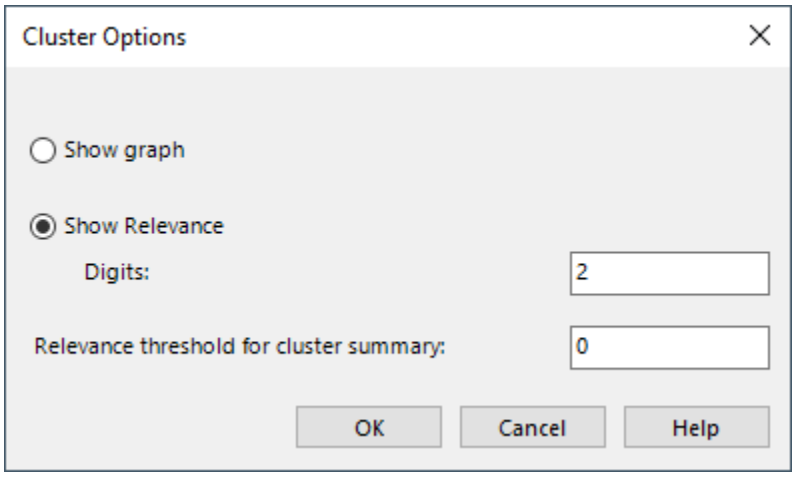
All charts for a variable use the same scale for convenient visual comparison.

When you view a chart in a separate window, right-click it to change the display options. The available options and commands are the same as in the dataset segment viewer charts. For example, you can copy any chart to the clipboard and paste into a Microsoft Office application.

The columns can be sorted according to the ***relevance*** of variables to a particular cluster. This is done by clicking on the header bar for the cluster you want to sort by.

The value of relevance is the K-L divergence (also called *information divergence*, or *relative entropy*). It measures the degree of relevance of a variable to each cluster. For details, see the topic Cluster Analysis Models in Mathematical Formulations.

To see the values of relevance for each variable, select **Tools | Options** from the menu. In the Cluster Options dialog, select the option "Show Relevance".



You can specify the necessary precision and the relevance threshold. Only the values with relevance greater than the specified threshold will be shown.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The **Parameters and Attributes** tab displays two tables.

The first table contains the information about all model variables and their attributes, i.e., the list of variables included in the model and all settings in the Cluster Analysis Attribute Editor (normalization type, the treatment of missing values, etc.).

Parameters and Attributes

Model Type: K-Means Cluster Analysis

Attributes

Name	Include	Scoring Input	Training Input	Role	Active	Value Type	Usage	Dummy Variables	Expression	Missing Values	Infinite Values
Region	yes	yes	yes	independent	yes	string	discrete	'Eight' 'Eleven' 'Five' 'Four' 'Nine' 'One' 'Other' 'Seven' 'Six' 'Ten' 'Three' 'Twelve' 'Two'	no	use	
age	yes	yes	yes	independent	yes	real	continuous		no	subst-median	
Residential Status	yes	yes	yes	independent	yes	string	discrete	'Missing' 'Live with Parents' 'Other' 'Own Home' 'Rent'	no	use	
Status2	yes	no	yes	independent	yes	string	discrete	'Bad Debt' 'OK'	yes	use	
Number Inquiries	yes	yes	yes	independent	yes	real	continuous		no	subst-median	
Avg Monthly Balance	yes	yes	yes	independent	yes	real	continuous		no	subst-median	
Total Service Charge Revenue	yes	yes	yes	independent	yes	real	continuous		no	subst-median	
Profit	yes	yes	yes	independent	yes	real	continuous		no	subst-median	
Delinquent Lev 1	yes	yes	yes	independent	yes	real	continuous		no	subst-median	
Delinquent Lev 2	yes	yes	yes	independent	yes	real	continuous		no	subst-median	
Delinquent Lev 3	yes	yes	yes	independent	yes	real	continuous		no	subst-median	
Amount Lost	yes	yes	yes	independent	yes	real	continuous		no	subst-median	
Credit Line	no	no	no								
Initial Score	no	no	no								

The second table summarizes all the training parameters entered in the Cluster Analysis Wizard when the model was inserted.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Parameters			
Description	Parameter Name	Value	Value Description
Perform resubstitution test	Resubstitute	true	
Valid Hit Range for resubstitution test	HitRange	0	
Break indicator variables apart	SubSequencer	false	
Random Seed	RandomSeed	1	
Requested number of clusters	NCluster	5	
Maximum number of iterations	MaxIteration	1000	
Normalize	Normalize	true	
Name of start point method	ClusterStartMethod	refinecluster	
Sample size	ClusterRefineSample	75	
Number of cycles	ClusterRefineCycles	10	
Overlap	Overlap	1	
Data Source	DataSource	[Credit2].[Credit]	
MultipleModels	MultipleModels	false	
# of records used for training	TrainedRecordCount	29167	
R-Squared	R-Squared	0.5025043249033486	
Pseudo-F	Pseudo-F	7363.898751473834	
CubicClusteringCriterion	CubicClusteringCriterion	666.6900507328138	
Expected-R-Squared	Expected-R-Squared	0.2196374702698939	

Market Basket Analysis

Overview

Market Basket Analysis is a data mining technique for discovering associations between items and deriving rules that indicate the likelihood of items to occur together in groups of specific types.

A typical Market Basket Analysis problem is to determine which products or product categories are likely to be purchased together. The resulting association rules can be used to build strategies for product promotions, placement, cross-sell, and so on. Other possible applications of association rules include health sciences, fraud detection, and many other areas where identifying patterns of events or behavior from transactional data is required.

Market Basket Analysis in Knowledge Studio includes the following features:

- deriving association rules from historical data
- visualizing the degree of attraction or repellence between items
- graphical representation of the top items most strongly associated with a specific basket
- graphical representation of the rank of association rules with respect to their lift or confidence
- applying association rules to new data to produce recommendations
- automatic generation of SQL code for association rules

Input Data for Market Basket Analysis

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The input data for a Market Basket Analysis model can be organized in any way that allows the application to group items for analysis. For example, this can be customer transaction data from a retail store, with item groups represented by product baskets in purchase transactions.

The data must satisfy certain criteria. Namely, it should contain at least the following columns:

- one or several ID columns that uniquely identify groups of items, such as transaction ID;
- a column used to uniquely identify items in your dataset, such as item name, item ID, or item category.

For example, in a dataset that contains data from several retail stores, a transaction may be uniquely identified by the pair <Store ID, Transaction ID>.

If your transactional data is distributed across several tables in a database, it must be consolidated into a single table before importing it and building a Market Basket Analysis model. One transaction can be represented by multiple records. An example of the required structure for the input dataset is given below:

Customer ID	Transaction ID	Item ID	Sale date	Quantity	Item Name	Item Category
2474	3723404	1014235	2001-04-08	1	HERALD AMERICAN	MAGAZINES AND BOOKS
2476	3723406	1020868	2001-04-08	1	BG ROAST SIRLOIN DNNR 9 OZ.	FROZEN FOODS
2480	3723410	1028084	2001-04-08	2	SINGLE DONUT	BAKERY
2484	3723414	1014235	2001-04-08	1	HERALD AMERICAN	MAGAZINES AND BOOKS
2485	3723415	1014235	2001-04-08	1	HERALD AMERICAN	MAGAZINES AND BOOKS
2485	3723415	1028083	2001-04-08	1	ONE DOZEN DONUTS	BAKERY
2487	3723417	1015807	2001-04-08	1	HALLS TABS MENTHO 9CT	CANDY
2487	3723417	1028042	2001-04-08	1	BYRNE 2% MILK GAL	DAIRY
2487	3723417	1028084	2001-04-08	5	SINGLE DONUT	BAKERY
2488	3723418	1014235	2001-04-08	1	HERALD AMERICAN	MAGAZINES AND BOOKS
2488	3723418	1018829	2001-04-08	1	OM LITTLE LINKS	MEAT
2492	3723422	1014502	2001-04-08	1	PC SQZ KETCHUP 28OZ	OTHER
2492	3723422	1017310	2001-04-08	1	ROAST BEEF	COLD CUTS

Main Concepts

Item: any item, such as a product in a market basket. The choice of the item identifier depends on the level of item detail in associations you want to analyze. In the above example, you may be interested in associations between product categories, products, or even specific product IDs. Note that there may be multiple products with the same name and different IDs or SKUs. For example, 2% Milk of different brands can be represented by different Item IDs with the same item name "2% Milk".

Item set: a set of items, such as a set of products in a market basket. In the above example, an item set {HALLS TABS MENTHO 9CT, SINGLE DONUT} is a subset of the basket with *Transaction ID*=3723417. At the first stage, before building association rules, a Market Basket Analysis model analyzes the frequency of different item sets.

Association Rule: An association rule indicates that a certain item set is likely to occur together with a certain other item set in the groups of items in the dataset. For example, the rule {FLOUR,

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

2% MILK} => {EGGS} indicates that customers who buy both FLOUR and 2% MILK are likely to also buy EGGS.

The main likelihood measures for association rules are Confidence and Lift, which are explained below.

Support: the frequency of an item set expressed in percentage, i.e., the measure of how often the items in the set occur together in groups in your dataset. For example, if the item set {SINGLE DONUT, 2% MILK} occurs in 250 out of 10000 transactions, then its support is 2.5%. The support of a rule A=>B is the support of the item set {A,B} that contains both item sets A and B.

Confidence: the confidence of a rule A=>B is the ratio of the support of the item set {A,B} containing both A and B and the support of A. For example, if SINGLE DONUT occurs in 5% of all transactions and the pair {SINGLE DONUT, 2% MILK} occurs in 2.5% of all transactions, then the confidence of the rule {SINGLE DONUT} => {2% MILK} is 2.5 / 5 = 0.5.

Lift: the lift of a rule A=>B, sometimes also called its *importance*, is Support ({A,B})/Support(A)*Support(B). In the above example, if 2% MILK occurs in 4% of all transactions, then the lift of the rule {SINGLE DONUT} => {2% MILK} is 2.5 / (5 * 4) = 0.125.

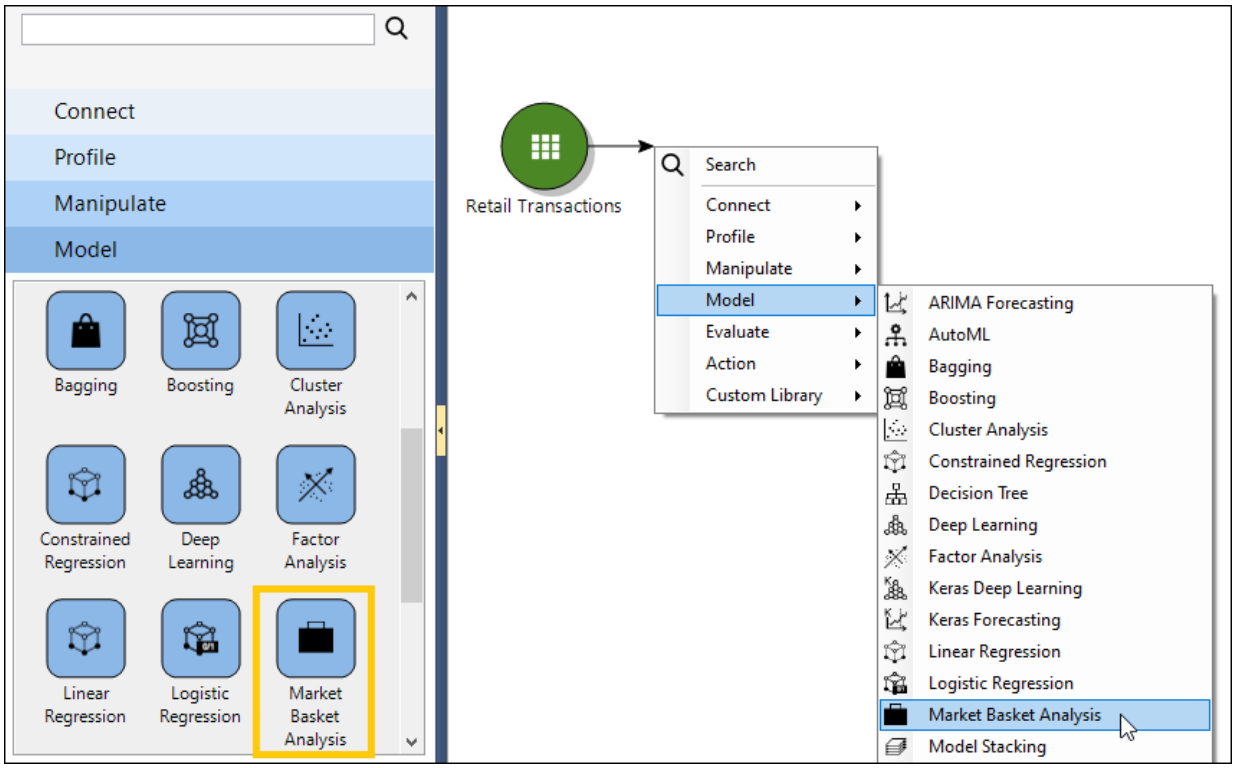
See also:

- Inserting Market Basket Analysis
- Deploying Association Rules: Overview
- Scoring a Market Basket Analysis Model

Inserting Market Basket Analysis

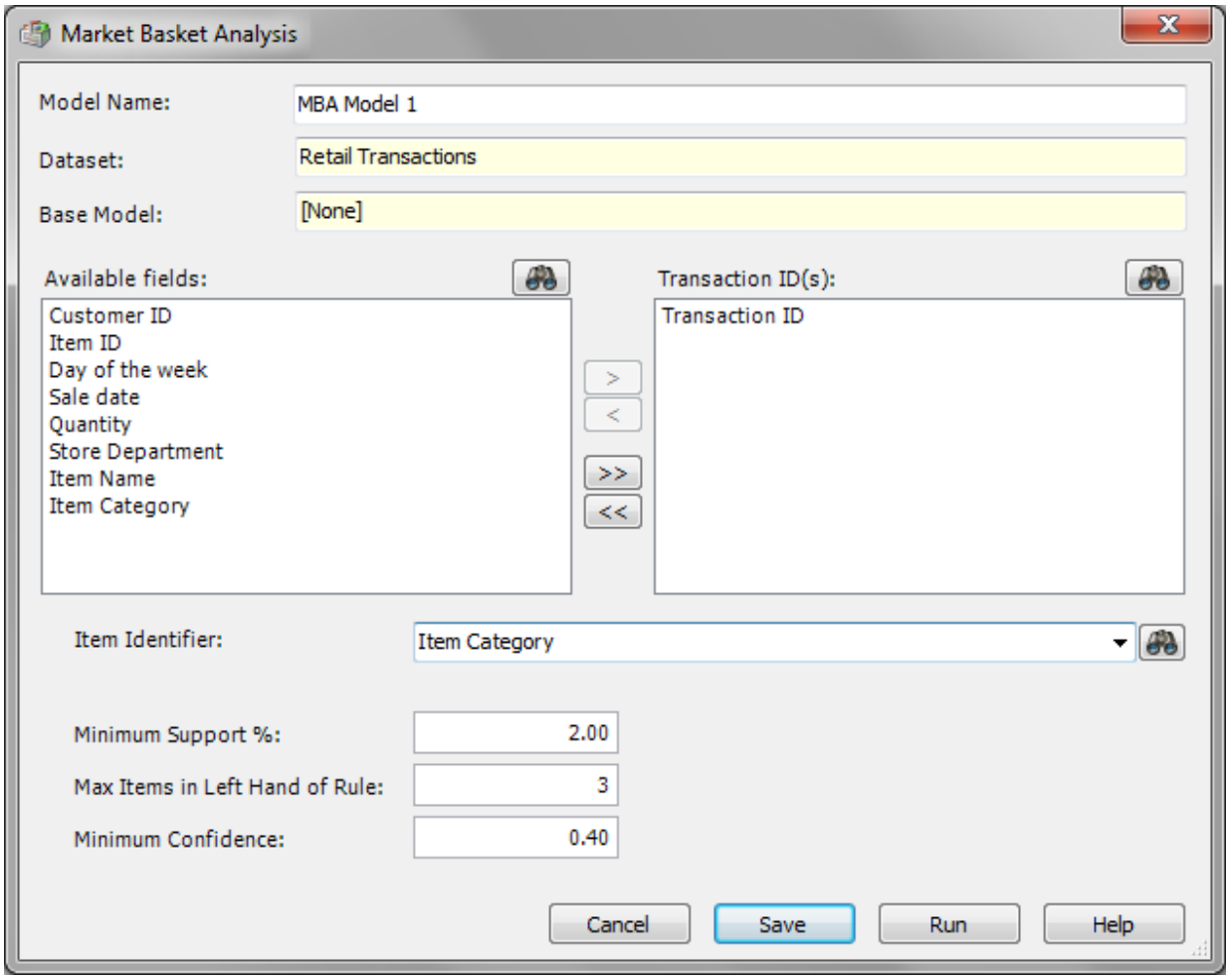
To insert a Market Basket Analysis model, drag the **Market Basket Analysis** node from the **Model** palette to the workflow canvas and connect a dataset node to it.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Then double-click the Market Basket Analysis node or right-click it and select **Modify** to open the wizard.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



In the Market Basket Analysis Wizard:

- Enter the model name.
- Select a template model If you want to copy its settings for the new model, namely, the columns selected as transaction and item identifiers, the minimum support and confidence thresholds, and the maximum size of left hand of rule.
- Select one or several ID fields that uniquely identify transactions in the input dataset. For example, if the dataset contains transactional data from several stores, you may need to select both the Store ID and Transaction ID field.
- For Item Identifier, select a field that contains item names. If your dataset contains a field where items are grouped into categories and you would rather explore associations between the categories, select that field as Item Identifier. In that case, a set of different items that belong to the same category will be treated as a single item. For example, both ROAST BEEF and COOKED HAM belong to the Item Category COLD CUTS, so if you select it as the Item Category column as Item Identifier, the model will treat the two items as one.
- Specify model parameters. These are the thresholds that affect the number of item sets and association rules produced by the model.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Note: Finding frequent item sets is the most computation-intensive phase of the task. Market Basket Analysis models based on some datasets may constitute complex problems that require a lot of memory. Such cases will be detected based on the information about the number of generated item sets, and a warning will be provided. If you choose to proceed with building the model and your system runs out of memory, you may want to rebuild your model with different parameters that result in fewer item sets and rules (for example, specify a higher support or confidence thresholds).

Minimum Support %:	Support threshold. The model will not consider item sets that occur in fewer than the specified percentage of transactions. The default value is 2%. Valid values are between 0.01 and 100.
Max Items in Left Hand Of Rule:	The maximum number of items in the left-hand side of the association rules. The default value is 3. Admissible values are between 1 and 100.
Minimum Confidence:	The model will only produce association rules with confidence higher than this threshold. The default value is 0.4. Valid values are between 0.01 and 1.

Click **Save** to save the settings, or click **Run** to save them and run the model.

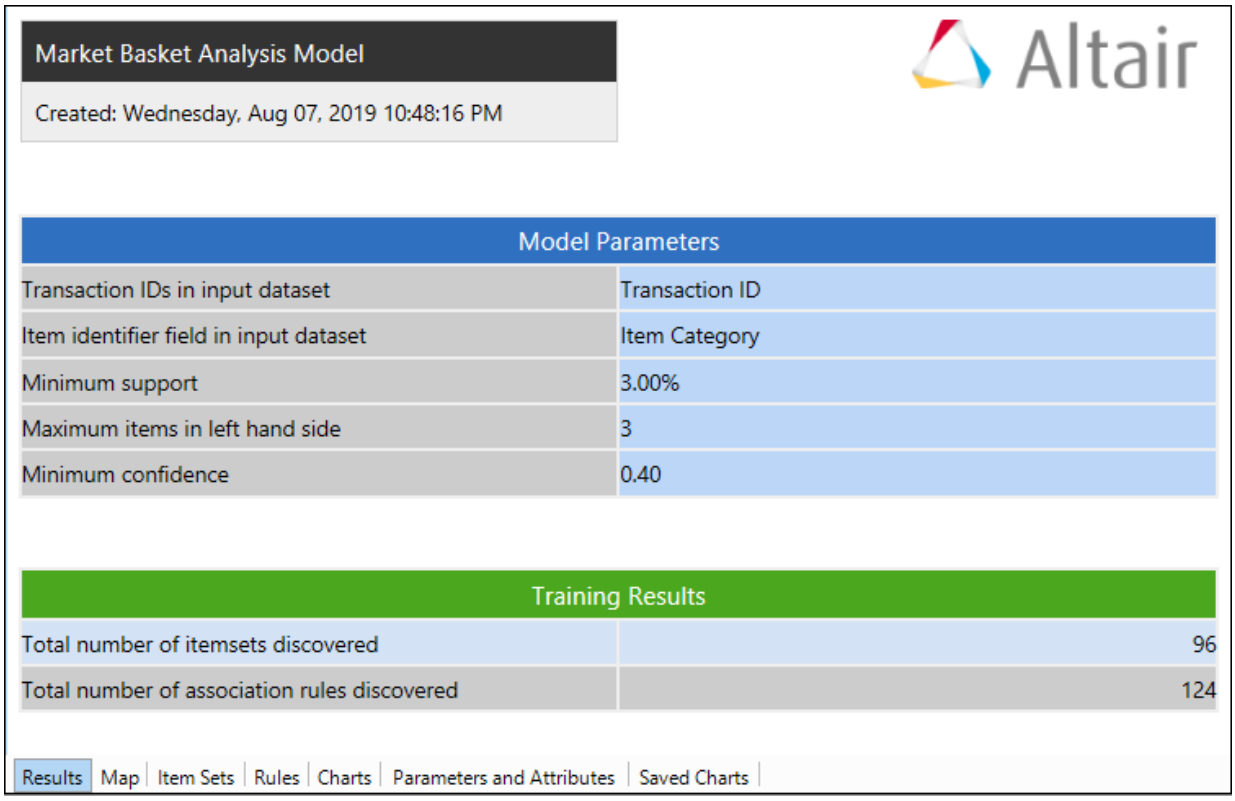
The model starts to search for frequent item sets satisfying the specified model parameters and construct the association rules.

See also: Model Results.

Market Basket Analysis Model Views

The **Results** tab is the first view displayed after the training of a Market Basket Analysis model. It provides the summary of model parameters and training results.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Item Attraction Map

The **Item Attraction Map** represents the degree of attraction or repellence between items in the sense of their likelihood to occur together in the same basket.

For example, in the case of retail store transactions, for each pair of products the map shows how likely they are to be purchased together. The closer they are, the more likely they are to be purchased together, and vice versa.

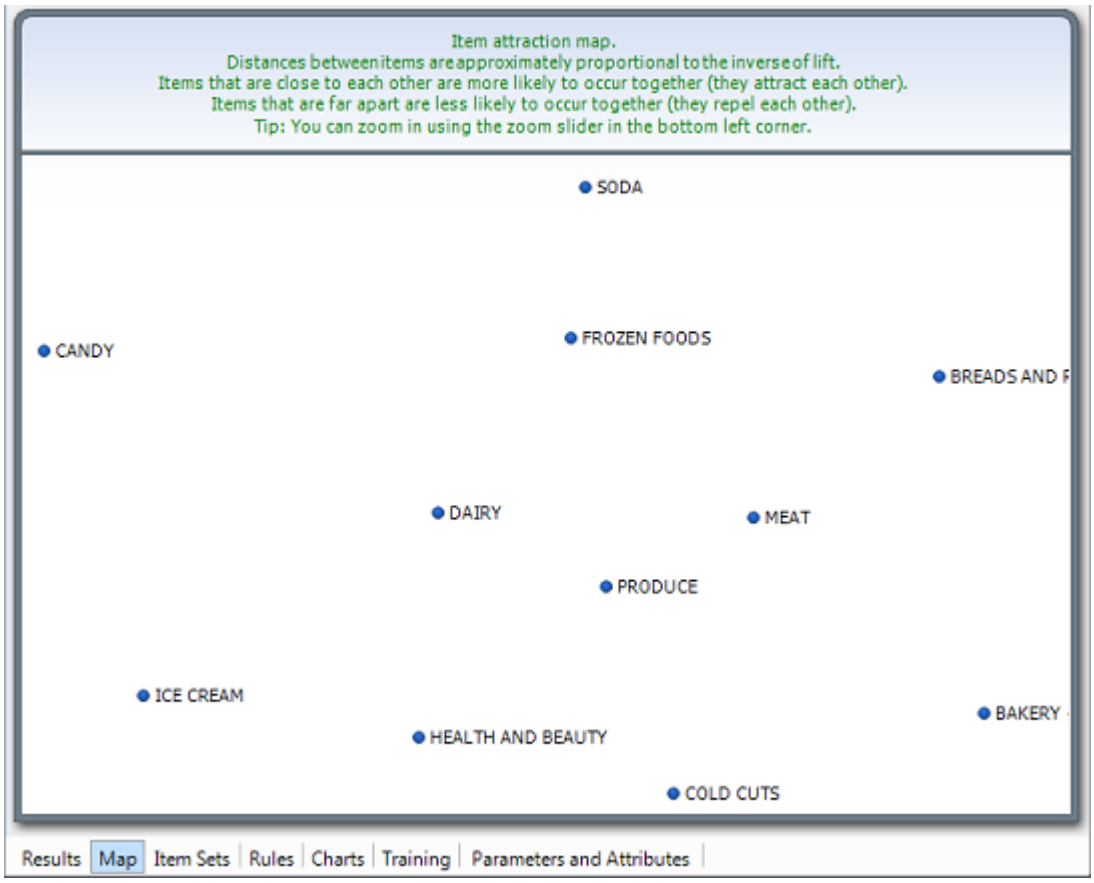
The distance between any two items A and B is approximately proportional to the inverse of lift for the rule A=>B (or B=>A). The distance is calculated from lift using a standard multidimensional scaling algorithm.

Note that the Item Attraction Map shows only the items from the rules that satisfy the settings in the Market Basket Analysis Wizard. In the **Rules** tab, you can view the list of rules represented in the Item Attraction Map if you reset the filter and set *Left Hand Size Min* = *Max* = 1.

If there are too many items in the map, or if they are too dense, you can zoom in and out on the map using the slider control in the bottom right corner the same way as you do for trees.

In the example below, it is clearly seen that PRODUCE is purchased together with MEAT much more frequently than CANDY is purchased with any other product.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Item Sets

The Item Sets view lists frequent item sets satisfying the parameters that were specified in the Market Basket Analysis wizard. The item sets are shown with their size and support.

The Size and Support columns are sortable.

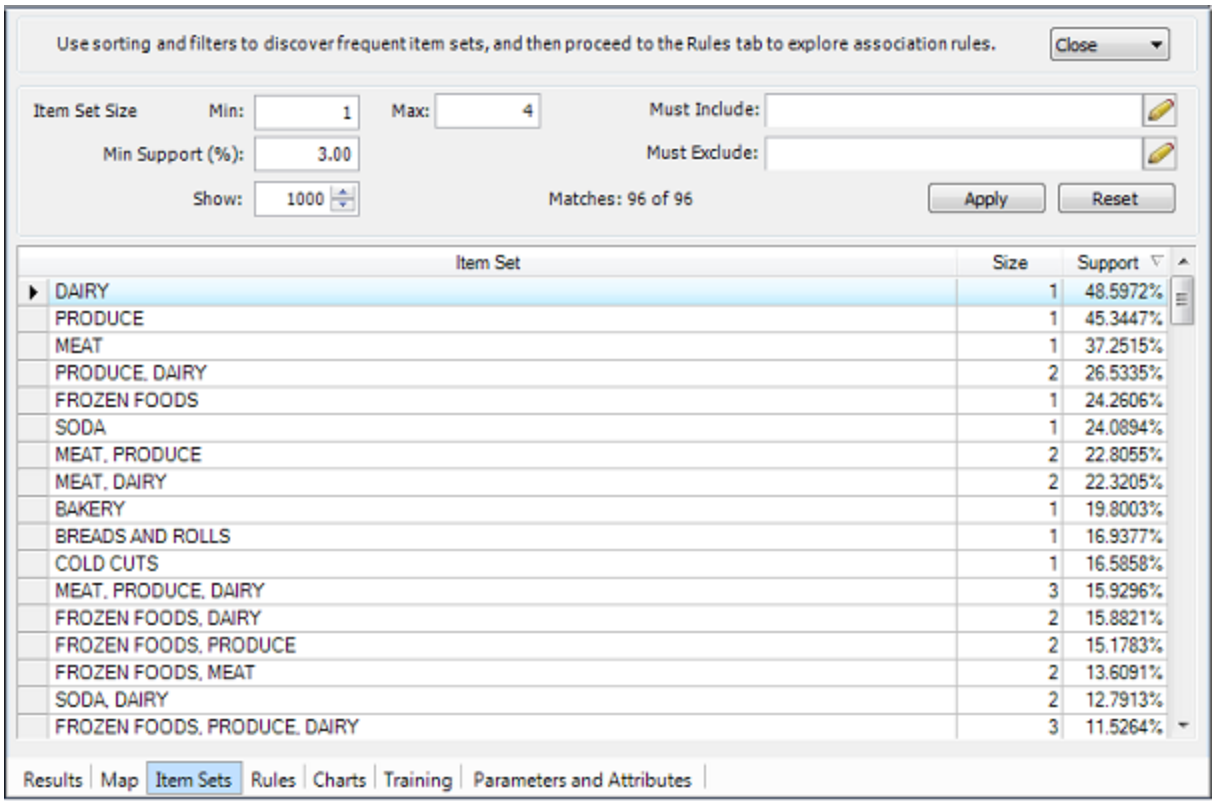
By default, data is sorted by Support; items with highest support are on top.

You can filter the item sets based on:

- Minimum Support threshold. The default is the value that was specified in the Market Basket Analysis wizard. Admissible values are between the default value and 100%)
- Minimum and maximum size of item sets (default Min = 1, default Max is the value specified in the training wizard)
- Names of specific items in the item sets (one or more items to be included or excluded)
- Maximum number of rows to show (default = 1000)

After you have specified filter conditions, click the Apply button to apply the filter. The Reset button resets the filter settings to defaults.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



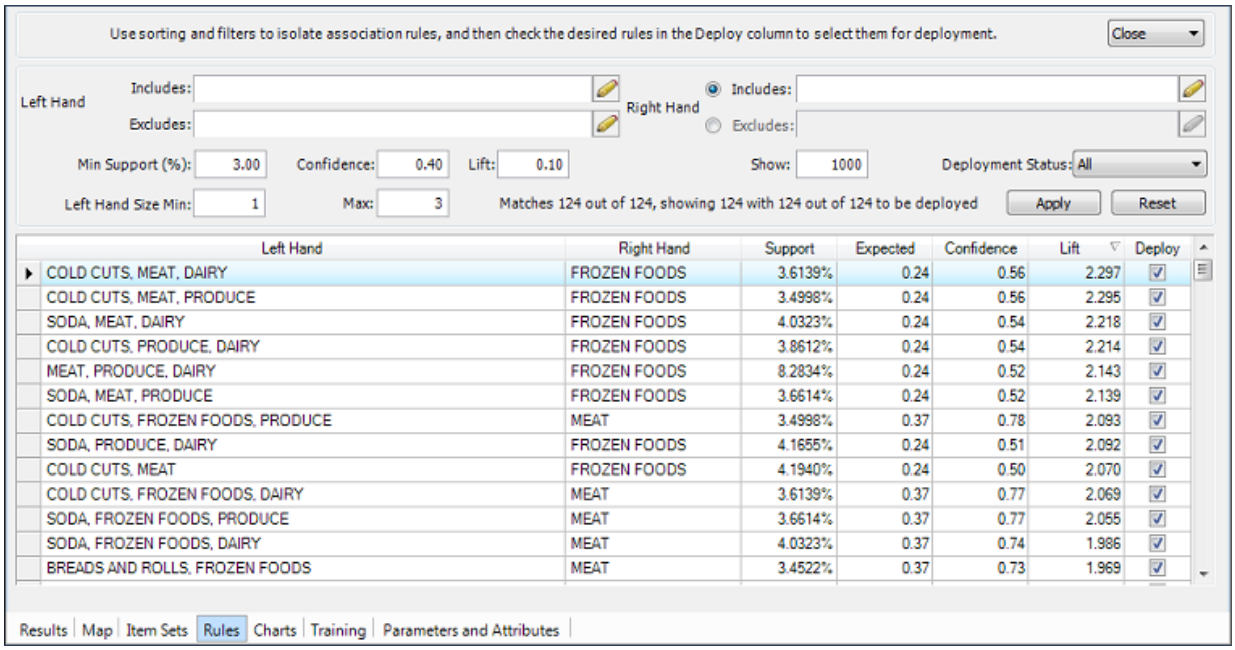
After analyzing frequent item sets, proceed to the Rules view.

Rules

The Rules tab of the Market Basket Analysis model shows association rules discovered by the model. The Rules table is sorted by Lift by default; items with highest Lift are on top. All columns that show numeric parameters are sortable.

Sorting and filters can be used to isolate the association rules of interest.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The left-hand and right-hand sides of the rules, called "left hand" and "right hand" for brevity, are shown in separate columns.

The Support column shows the support of the item set consisting of the left and right hand side of the rule.

The Expected column shows the frequency of the right-hand side in the input dataset and is intended for comparing with Confidence.

Filters

You can filter the rules based on:

- Minimum support, confidence, and lift thresholds (the default values are the values specified in the Market Basket Analysis wizard). Note that you cannot specify values smaller than those you specified in the wizard.
- Minimum and maximum size of item sets (default Min = 1; default Max = the value specified in the Market Basket Analysis wizard)
- Specific items to be included in or excluded from the left-hand and right-hand sides of rules
- Maximum number of rows to show (default = 1000)
- Rules checked for deployment

After specifying filter conditions, click the *Apply* button to apply the filter. The *Reset* button resets the filter settings to defaults.

The status message in the middle, just above the rules grid, informs you how many rules match the current filter out of the total number of rules discovered by the Market Basket Analysis model. It also indicates how many of the rules selected for deployment are visible. See below for details regarding deployment status.

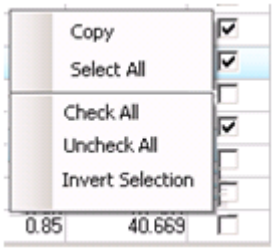
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Selecting Rules for Deployment

Deploying association rules consists in generating recommendations based on these rules. Recommendations are produced by applying the rules to the new data. For details, see the topic Deploying Association Rules: Overview

Only the rules marked for deployment in the Rules tab serve as input for the scoring command or code generation command. Make sure you select al least one rule for deployment before using the commands *Tools* | *Score* or *Insert* | *Code Generation*. In the above screenshot, all rules are selected for deployment.

The Deploy column is intended for selecting rules for deployment. After isolating the rules of your interest using sorting and filters, check the desired rules in the Deploy column to select them for deployment. You can check several rules at a time by highlighting them first, and then placing a checkmark for any one of the highlighted rules. To highlight several rules, drag the mouse over them or clicking on them while holding down the SHIFT or CTRL keys. To check or uncheck all displayed rules, or to invert your selection, right-click in the Deploy column and select the appropriate command from the context menu.



Deployment Status

Note that the deployment status of any rule is not affected by applying filters. In other words, the rules selected for deployment remain selected even if they are not visible in the Rules table as a result of applying filters. For example, if there are any rules with Support < 10% selected for deployment, and then you specify Min Support = 15% and apply the filter, no rules with Support < 10% will be visible, but they will remain selected for deployment.

Use the Deployment Status options just above the Reset button to quickly isolate rules selected for deployment, or only unselected rules. Remember to use the Apply button after changing the Deployment Status options.

Charts

The Chart view provides graphical illustration of association rules. There are two types of charts: Association Rank chart and Rule Comparison chart.

- The Association Rank chart shows items most strongly associated with one or several given items in the form of a bar chart.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

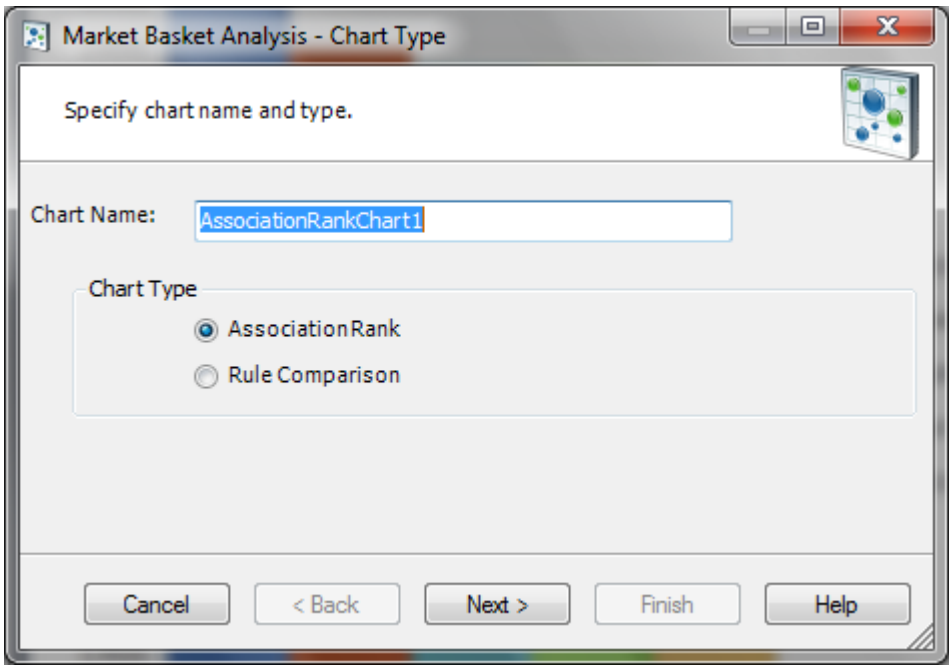
- The Rule Comparison chart ranks association rules with respect to their lift or confidence, while also showing their support. It is a bubble chart where the color of a bubble represents the lift or confidence and its size represents the support of the corresponding rule.

Multiple charts can be added to this view.

Adding Charts

Clicking on the Charts tab automatically brings up the Add Chart wizard. While in the Charts tab, you can also use the *Add* and *Remove* buttons in the toolbar or the *Add Chart ... / Remove Chart ...* commands under the *Tools* menu to add or remove a chart.

Choose the type and specify the name for the new chart.



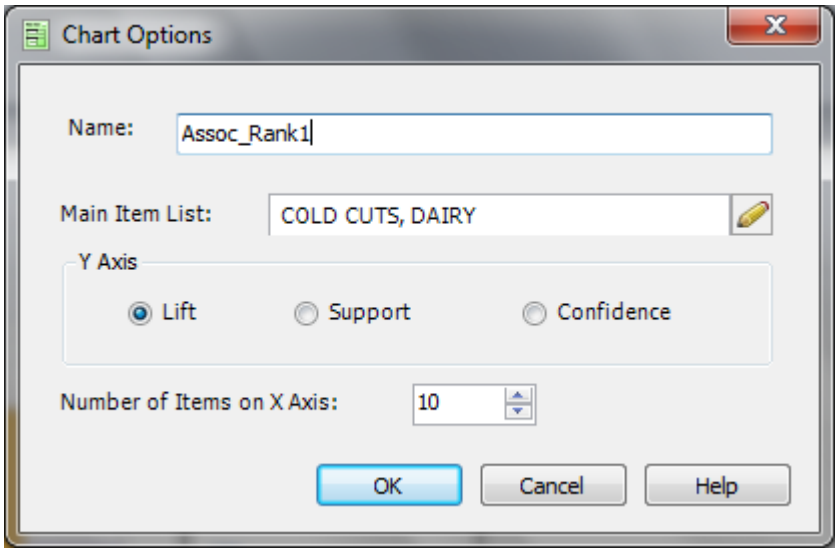
Association Rank Chart

If the Association Rank chart type was selected when adding a chart, the next step of the wizard provides the following options:

Main Items List	Contains all items occurring in the left-hand side of the rules. Select one or more items from the list.
Y Axis	The metric to show on the Y axis of the chart. The options are Lift, Support, and Confidence. The default is Lift.
Number of Items on X Axis	The maximum number of items to show on the X axis. The default is 10. These are the top items associated with the Main List basket in terms of

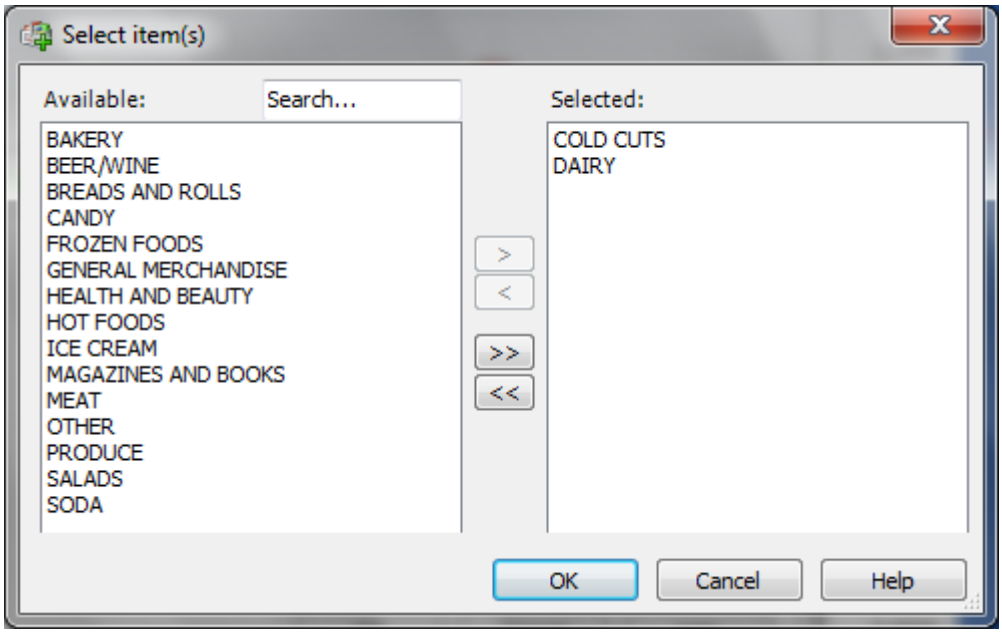
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

the metric selected for the Y axis (Lift, Confidence, or Support).



Specify the chart name.

To add items to the *Main Items List*, click anywhere in the field or on the button to the right. This brings up the Select Items dialog.



Select the items by double-clicking on them or pressing the arrow buttons. If you need to quickly locate an item in the long list, start typing the item name in the Search window.

Specify the number of top items to show as *Number of Items on X Axis*, and select Lift, Support, or Confidence for the ranking metric.

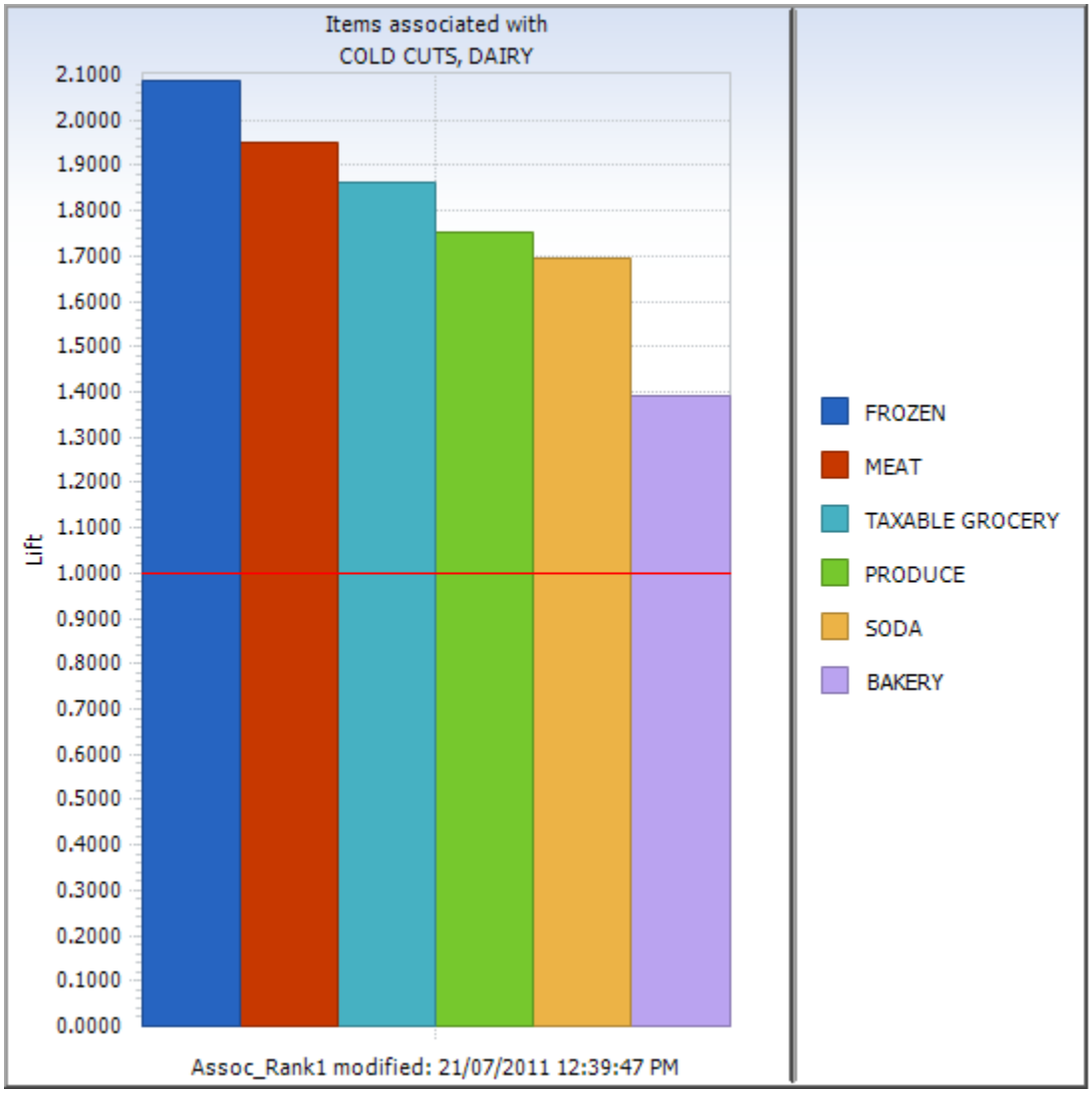
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Click OK.

The chart is based on the rules with the Main List items on the left-hand side. First, these rules are ranked with respect to the chosen metric, and top N of them are selected, where N is the number of items on X axis specified by the user. Then the items on the right-hand side of these rules are shown in the chart. The height of the bars corresponds to the value of the selected metric.

If Lift is selected for the Y axis, then a horizontal line will be shown at 1 to indicate the reference point of item independence. Items or item sets A and B in the rule A=>B are said to attract each other if Lift (A,B)>1, and repel each other if Lift (A,B)<1. If Lift (A,B)=1, they are considered to be independent.

In the example below, there are six rules with COLD CUTS and DAIRY on the left-hand side. The product category most likely to be purchased together with COLD CUTS and DIARY is FROZEN FOODS, represented by the blue bar.

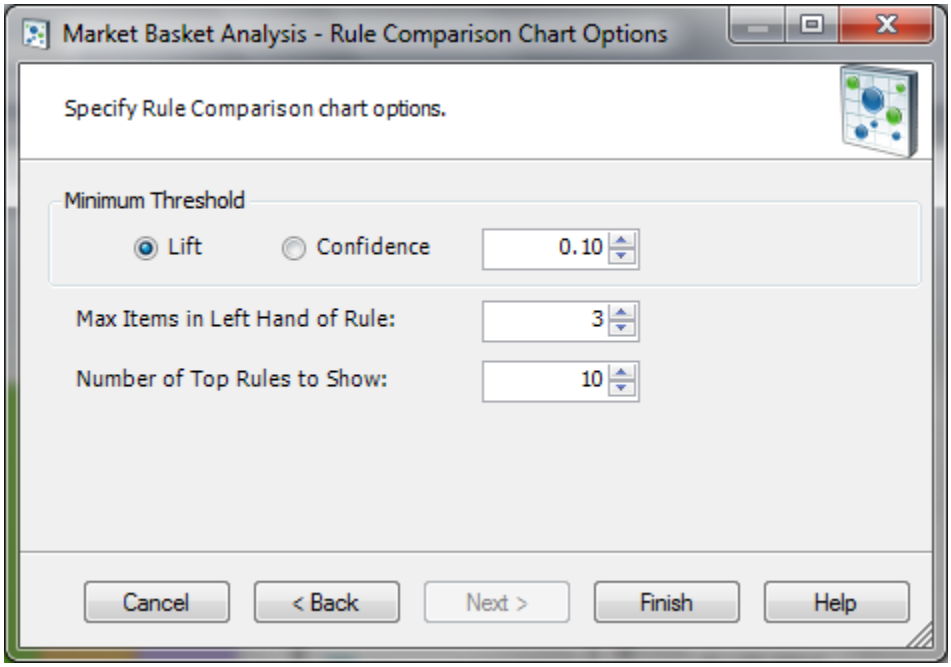


Rule Comparison Chart

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The following options are available:

Minimum Threshold for Lift or Confidence	Only the rules with Lift or Confidence above the specified threshold will be displayed. The default values are 0.1 and 0.4 for Lift and Confidence, respectively.
Max Items in Left Hand Of Rule:	The maximum number of items in the left-hand side of the rules. The default value is 3. This value cannot exceed the value specified in the Market Basket Analysis wizard.
Number of Top Rules to Show	No more than the specified number of rules will be displayed. The default value is 10.



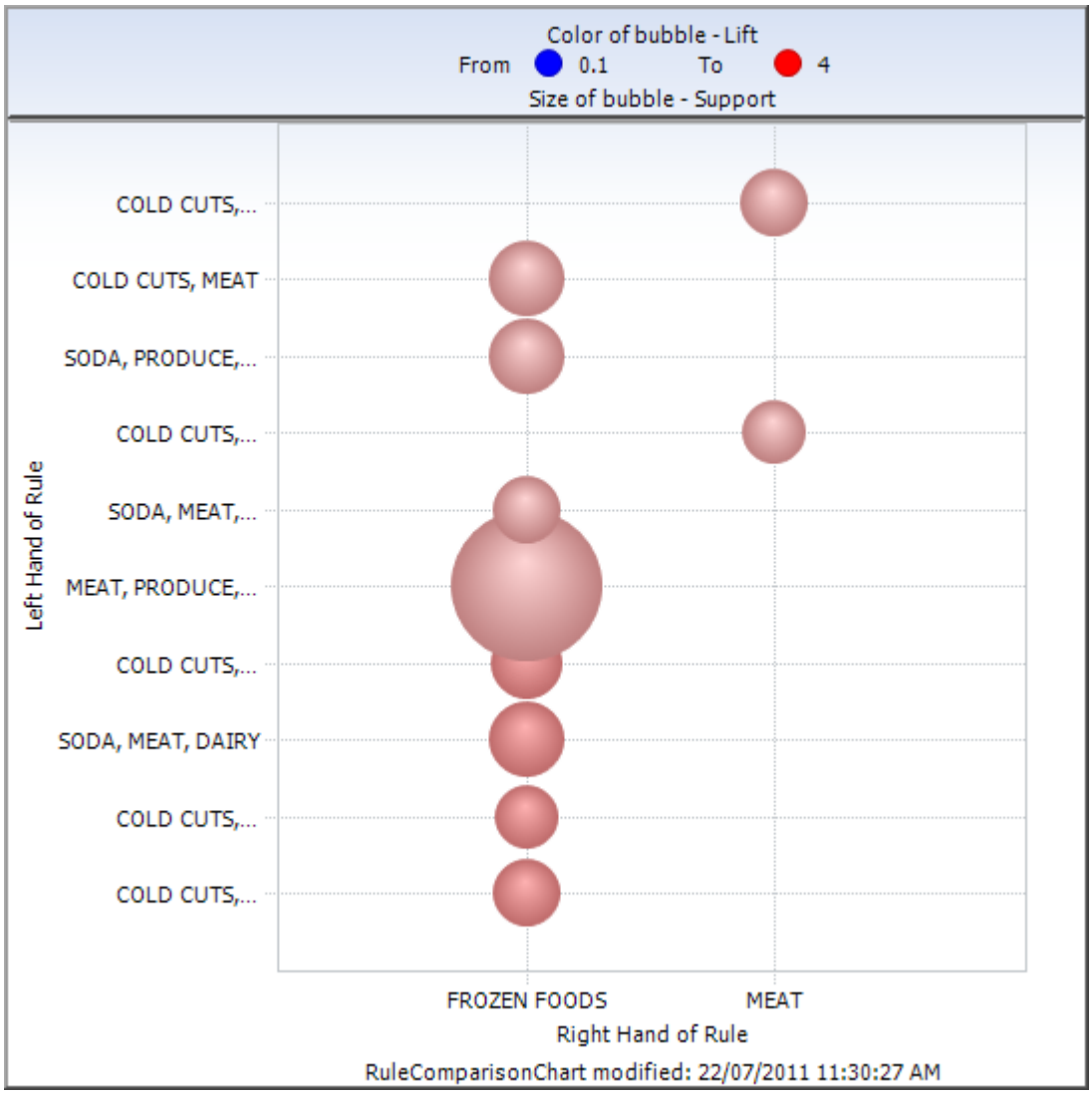
Click Finish. The result is a 2D bubble chart where each bubble represents a rule. The left hand of rules are shown on the Y axis, the corresponding right hands are on the X axis, the color of bubbles represents lift or confidence, and the size of bubbles represent the support of the corresponding rule.

If lift is selected, then shades of red represent lift values higher than 1, which means attraction. Shades of blue represent lift values smaller than 1, which means repellence. The value of 1 is the reference point indicating item independence, represented by color white.

If confidence is selected, then 0 is the reference point represented by color white, with shades of red representing other values.

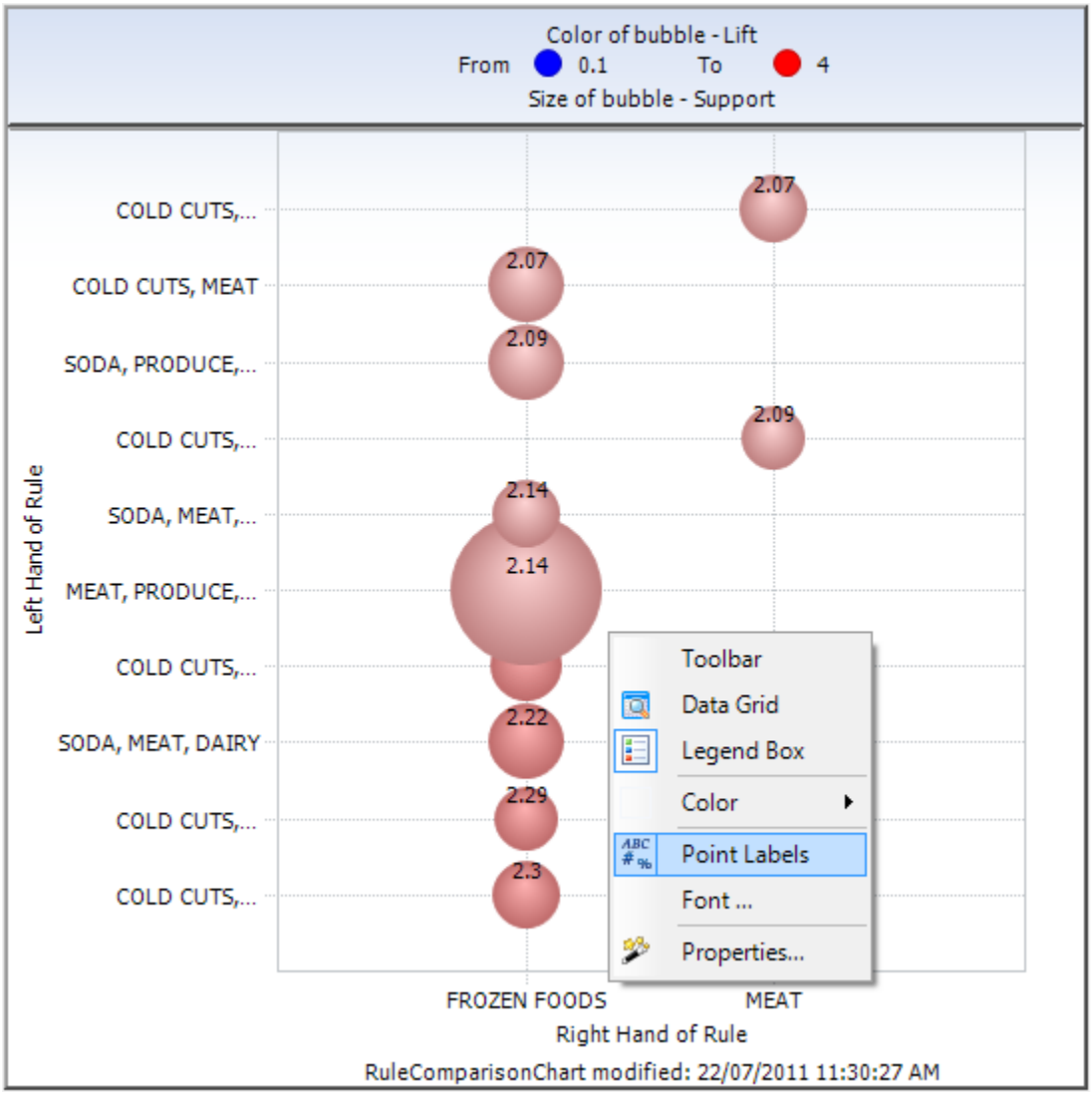
Darker shades correspond to higher parameter values, while lighter shades correspond to lower values.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Note that you can view the values of lift or confidence on each bubble by turning on the Point Labels option. Right-click anywhere on the chart and select Point Labels in the context menu.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

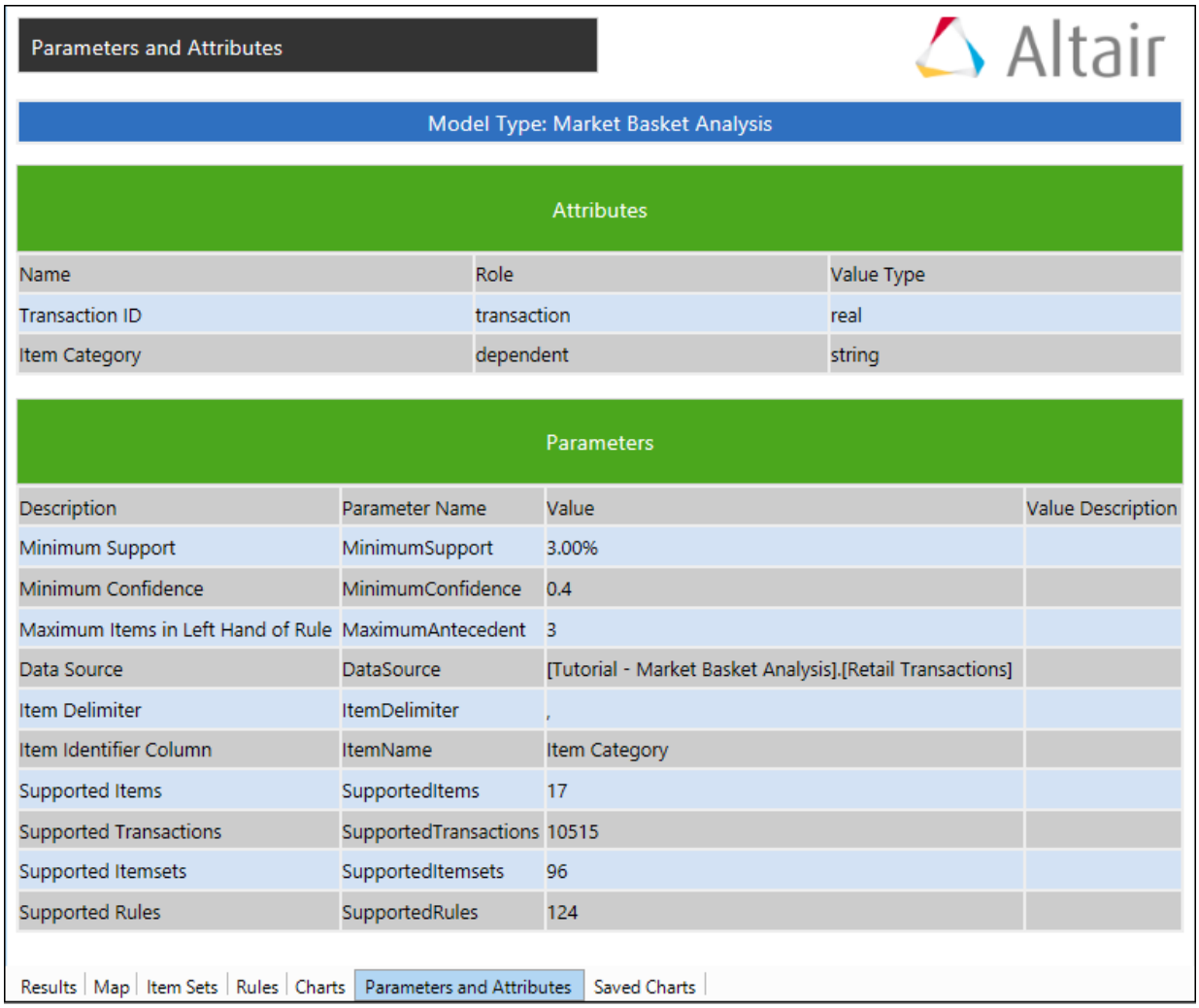


The settings of any chart can be changed by clicking on it and selecting the *Tools* | *Options* command or the Options toolbar button.

Parameters and Attributes

The Parameters and Attributes tab shows the summary of the model properties and parameters such as the date it was created, the data source name, columns selected for the transaction ID and item identifier, minimum threshold parameters, and basic item set statistics.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Parameters and Attributes

Model Type: Market Basket Analysis

Attributes

Name	Role	Value Type
Transaction ID	transaction	real
Item Category	dependent	string

Parameters

Description	Parameter Name	Value	Value Description
Minimum Support	MinimumSupport	3.00%	
Minimum Confidence	MinimumConfidence	0.4	
Maximum Items in Left Hand of Rule	MaximumAntecedent	3	
Data Source	DataSource	[Tutorial - Market Basket Analysis].[Retail Transactions]	
Item Delimiter	ItemDelimiter	,	
Item Identifier Column	ItemName	Item Category	
Supported Items	SupportedItems	17	
Supported Transactions	SupportedTransactions	10515	
Supported Itemsets	SupportedItemsets	96	
Supported Rules	SupportedRules	124	

Results | Map | Item Sets | Rules | Charts | Parameters and Attributes | Saved Charts

Principal Component Analysis

Principal Component Analysis (PCA) is normally used for variable reduction and understanding the relationships between the variables. It is a data reduction method that transforms a number of correlated variables into a smaller set of uncorrelated variables called ***principal components***. The first principal component is the combination of variables that explains the greatest amount of variation. The second principal component defines the next largest amount of variation and is independent of the first principal component, and so on. There can be as many possible principal components as there are variables.

Factors which are generated are thought to be representative of the underlying processes that have created the correlations among variables.

PCA can also be used as a precursor to running Cluster Analysis as it can help to generate better cluster models.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

See the Principal Component Analysis section of Mathematical Formulations for more details.

Proceed to the description of the PCA node wizard.

See also: Viewing PCA Results.

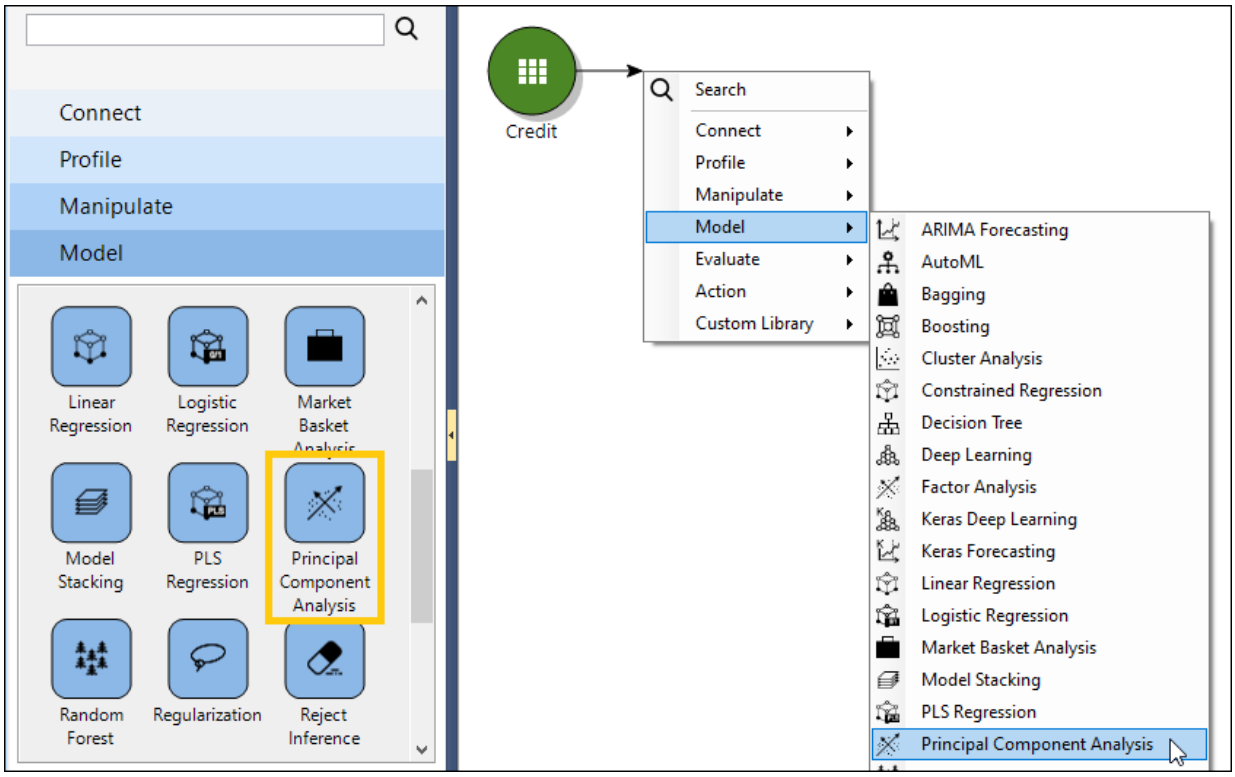
Inserting Principal Component Analysis and Viewing Results

To add a **Principal Component Analysis** node to the workflow:

- Drag a connection arrow from the desired dataset node to any area on the Workflow canvas background. From the context menu that appears when you release the mouse button, select **Model --> Principal Component Analysis** as in the example screenshot below. A new **Principal Component Analysis** node will be created and the wizard will open.

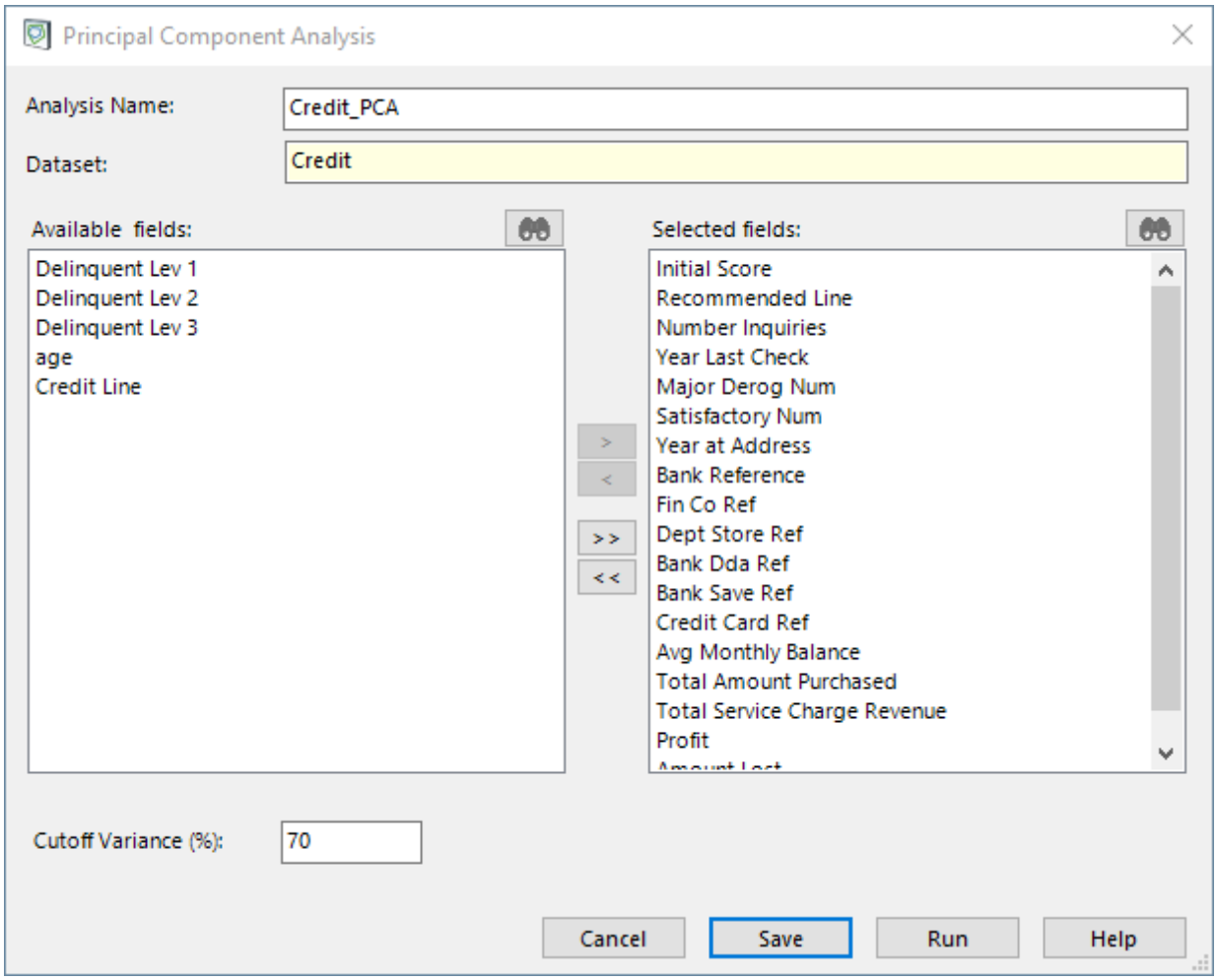
OR

- Open the **Model** palette and drag the **Principal Component Analysis** node onto the workflow canvas. The **Principal Component Analysis** node is gray while its inputs are undefined. Connect the input dataset to the new node.



Double-click the Principal Component Analysis node or right-click it and select **Modify** to open the wizard.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Analysis Name	Specify the name for the Principal Component Analysis node.
Dataset	The input dataset for Principal Component Analysis.

Select the fields to be included in the Principal Component Analysis (PCA). Only numeric variables can be used for the PCA. If the correlations tab in the dataset is active and initiated, only the variables used in the correlations tab will be selected by default.

Variables listed on the right will be included in the analysis. You can move the variables to and from the 'Selected' list by highlighting them and using the arrow buttons. Double-clicking the individual variables will also move them from one list to the other.

To select more than one variable in the lists, use mouse dragging, or mouse clicks while holding the SHIFT key or the CTRL key. If there are a lot of variables (more than a hundred, for example), it is convenient to use **Field Finder** (the binoculars button).

Cutoff (%)	***Variance***	Specifies the percent of variance to explain (the defaults is 70%). Changing this number can affect the number of components found.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Click **Save** to save the settings, or click **Run** to save the settings and run the model immediately.

To view the resulting Principal Component Analysis model, right-click on the Principal Component Analysis node and select the **Open View** command from the context menu.

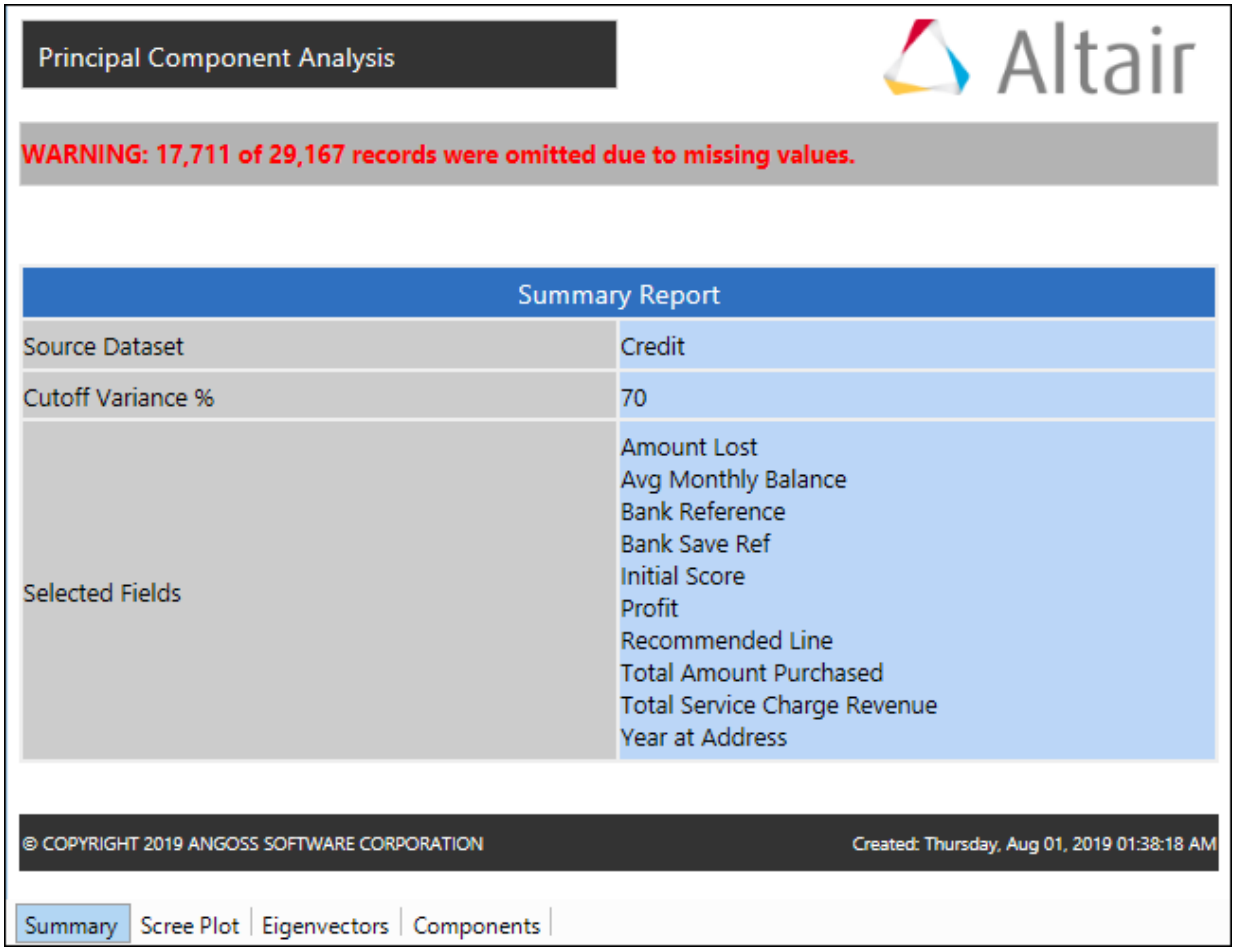
See the topic PCA: Summary

See also:

- PCA: Scree Plot
- PCA: Eigenvectors
- PCA: Components

The **Summary** tab displays the report of the PCA settings that were specified in the PCA Wizard: the source dataset, cutoff variance, and selected fields.

Note that records with missing values are excluded from the analysis. A warning in the report indicates the number of records were omitted due to missing values.



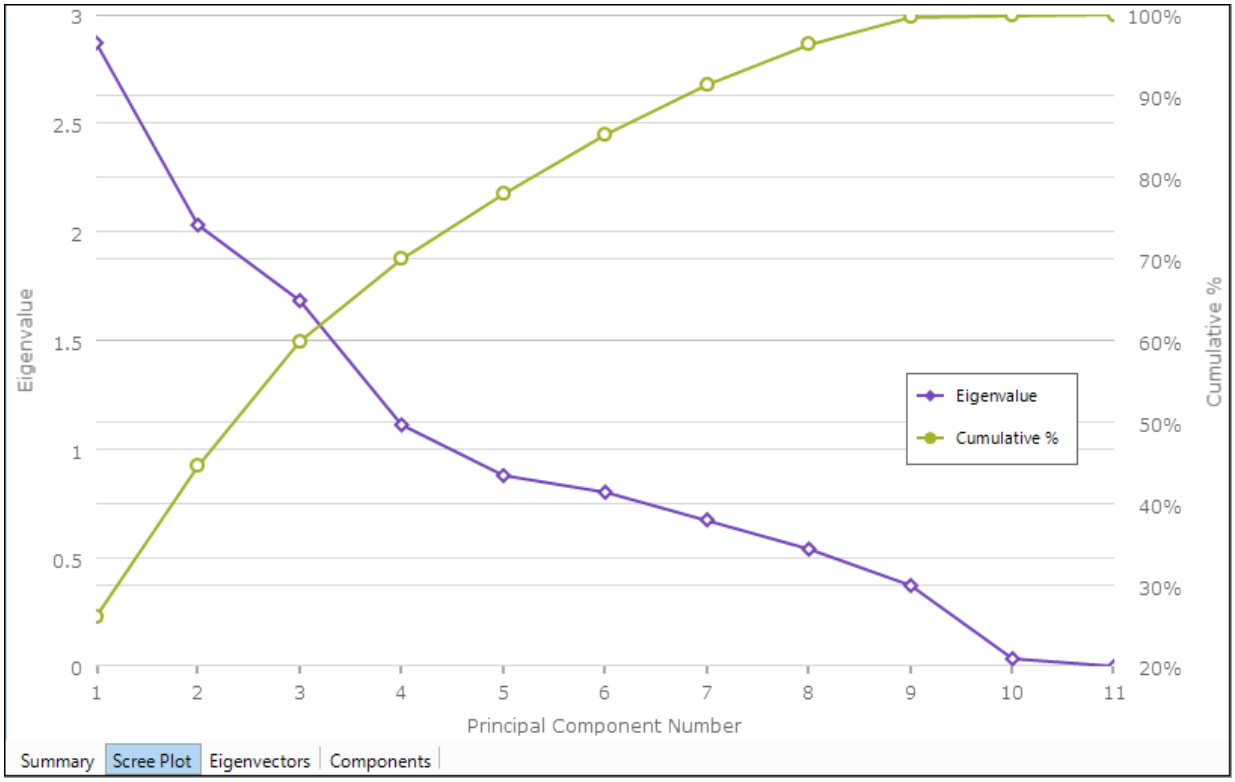
See also:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

PCA: Scree Plot
PCA: Components

The **Scree Plot** tab of Principal Component Analysis shows the line plot of the eigenvalues of principal components and the line plot of the cumulative proportion of variance explained.

- The *Eigevalue* plot helps determine how many principal components should be retained as significant. The general rule is to retain the components in the steep segment of the eigenvalue curve.
- The *Cumulative %* curve is the cumulative proportion of variance explained by consecutive components. Use it in combination with the eigenvalue plot to retain the components explaining the required level of variance.



The plot can be copied as an image using the **Copy** command under the **Edit** menu and pasted in any Office document or image editor.

See also:

PCA: Eigenvectors

PCA: Components

The **Eigenvectors** tab displays a table of eigenvectors for each of the variables selected in the PCA node wizard along with the associated eigenvalues and variance percentages.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

	Eigen1	Eigen2	Eigen3	Eigen4	Eigen5	Eigen6	Eigen7
age	0.04314	0.38044	0.35776	-0.07684	-0.26235	0.00771	-0.19127
Amount Lost	-0.57199	0.26920	-0.23027	0.01491	0.00723	0.02237	0.02355
Avg Monthly Balance	-0.51808	-0.14814	0.25253	0.19234	0.12057	-0.00028	0.01781
Bank Reference	-0.00305	-0.02921	-0.04771	-0.11604	0.54533	0.59191	-0.50972
Credit Card Ref	0.10504	0.15130	0.02606	-0.38081	0.55577	0.00012	0.44816
Credit Line	-0.02337	0.32031	0.40799	-0.15714	-0.00465	-0.16755	-0.16233
Delinquent Lev 1	-0.14495	-0.38740	0.29758	0.22017	0.11983	0.03938	0.07002
Major Derog Num	-0.03935	-0.04937	0.11220	-0.03769	-0.44579	0.76450	0.34518
Number Inquiries	0.19612	0.35500	0.04994	0.44208	0.16304	0.11506	0.17482
Profit	0.44315	-0.37215	0.38250	0.06114	0.04243	-0.02859	-0.01919
Satisfactory Num	-0.06650	0.14192	0.31946	-0.48148	0.01353	0.04134	0.27805
Total Amount Purchased	-0.32375	-0.10050	0.43983	0.13314	0.15455	-0.06685	0.03254
Year at Address	0.08506	0.29679	0.19641	0.08928	-0.06275	0.10297	-0.39228
Year Last Check	0.13078	0.31488	0.04220	0.51683	0.20204	0.06452	0.29715
Eigenvalues	2.36019	1.75666	1.66163	1.27644	1.07319	1.00296	0.94218
% Variance	16.85847	12.54756	11.86881	9.11741	7.66565	7.16399	6.72983

Summary | Scree Plot | Eigenvectors | Components

The columns **Eigen1, Eigen2**, **Eigen3**..., **EigenN** are eigenvectors ordered by eigenvalue. The eigenvector with the largest eigenvalue is the first column.

The **eigenvalues** are displayed in the second row from the bottom. An eigenvalue gives you an idea of how many of the original variables are explained by each component. There are different criteria for retaining principal components with the largest eigenvalues. For example, according to the K1 or Kaiser criterion, only the components with eigenvalues greater than 1 are selected for modeling.

Eigenvalues measure variance along the principal components. If you perform principal component analysis on the correlation matrix, the variables will be standardized, which means that each variable would have a variance of 1, and the total variance would be equal to the number of variables used in the analysis.

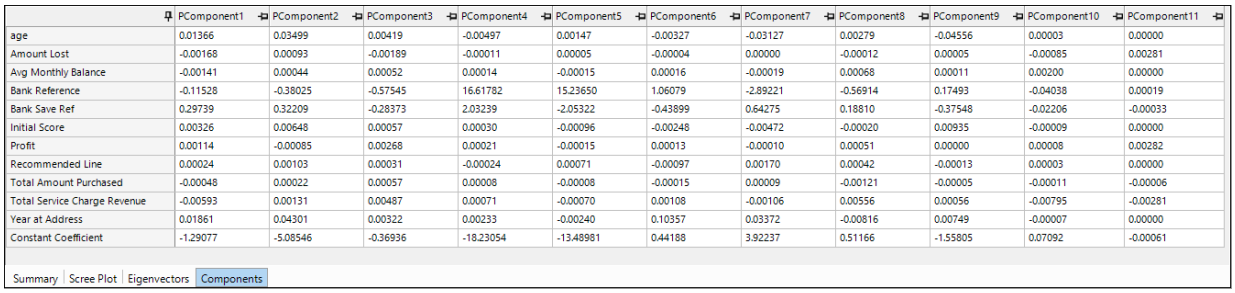
The last row **% Variance** shows the proportion of the variance represented by each eigenvector. For each cell, the value is calculated by dividing the eigenvalue by the number of variables and multiplying by 100%.

See also:

- PCA: Scree Plot
- PCA: Components

The **Components** tab of PCA displays the coefficients of each variable in the linear combinations that express the resulting components.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



	PComponent1	PComponent2	PComponent3	PComponent4	PComponent5	PComponent6	PComponent7	PComponent8	PComponent9	PComponent10	PComponent11
age	0.01366	0.03499	0.00419	-0.00497	0.00147	-0.00327	-0.03127	0.00279	-0.04556	0.00003	0.00000
Amount Lost	-0.00168	0.00093	-0.00189	-0.00011	0.00005	-0.00004	0.00000	-0.00012	0.00005	-0.00085	0.00281
Avg Monthly Balance	-0.00141	0.00044	0.00052	0.00014	-0.00015	0.00016	-0.00019	0.00068	0.00011	0.00200	0.00000
Bank Reference	-0.11528	-0.38025	-0.57545	16.61782	15.23650	1.06079	-2.89221	-0.56914	0.17493	-0.04038	0.00019
Bank Save Ref	0.29739	0.32209	-0.28373	2.03239	-2.05322	-0.43899	0.64275	0.18810	-0.37548	-0.02206	-0.00033
Initial Score	0.00326	0.00648	0.00057	0.00030	-0.00096	-0.00248	-0.00472	-0.00020	0.00935	-0.00009	0.00000
Profit	0.00114	-0.00085	0.00268	0.00021	-0.00015	0.00013	-0.00010	0.00051	0.00000	0.00008	0.00282
Recommended Line	0.00024	0.00103	0.00031	-0.00024	0.00071	-0.00097	0.00170	0.00042	-0.00013	0.00003	0.00000
Total Amount Purchased	-0.00048	0.00022	0.00057	0.00008	-0.00008	-0.00015	0.00009	-0.00121	-0.00005	-0.00011	-0.00006
Total Service Charge Revenue	-0.00593	0.00131	0.00487	0.00071	-0.00070	0.00108	-0.00106	0.00556	0.00056	-0.00795	-0.00281
Year at Address	0.01861	0.04301	0.00322	0.00233	-0.00240	0.10357	0.03372	-0.00816	0.00749	-0.00007	0.00000
Constant Coefficient	-1.29077	-5.08546	-0.36936	-18.23054	-13.48981	0.44188	3.92237	0.51166	-1.55805	0.07092	-0.00061

Summary | Scree Plot | Eigenvectors | Components

Principal Component Analysis computes all the **eigenvectors** for the final purpose of producing new columns on the basis of the input variables. These columns are linear combinations of the original columns, plus a constant term. This constant represents the mean subtracted from the data points in the beginning of the PCA process.

Example of a component representation for PComponent1 from the example screenshot above:

```
PComponent1 = 0.0136619059462795 * [age] + -0.00167810377868361 *
[Amount Lost] + -0.00141145969613312 * [Avg Monthly Balance] + -
0.115275649171164 * [Bank Reference] + 0.297386288008992 * [Bank Save
Ref] + 0.00326355587815566 * [Initial Score] + 0.00113736141132766 *
[Profit] + 0.000235046906524128 * [Recommended Line] + -
0.0004756128142604 * [Total Amount Purchased] + -0.00593459549593711
* [Total Service Charge Revenue] + 0.0186107528507943 * [Year at
Address] + -1.29076781759777
```

You can take note of variables with the highest component scores and see if this is in agreement with your expectations according to the business logic.

From this view, you can generate the expression for each factor in terms of the original variables using the command **Calculation From FA** from the **Tools** menu. It produces expressions such as the one given above, which can be imported into any dataset with the same original fields using the **Import Expressions** feature in the **Variable Transformations** node wizard.

See also:

PCA: Scree Plot

PCA: Eigenvectors

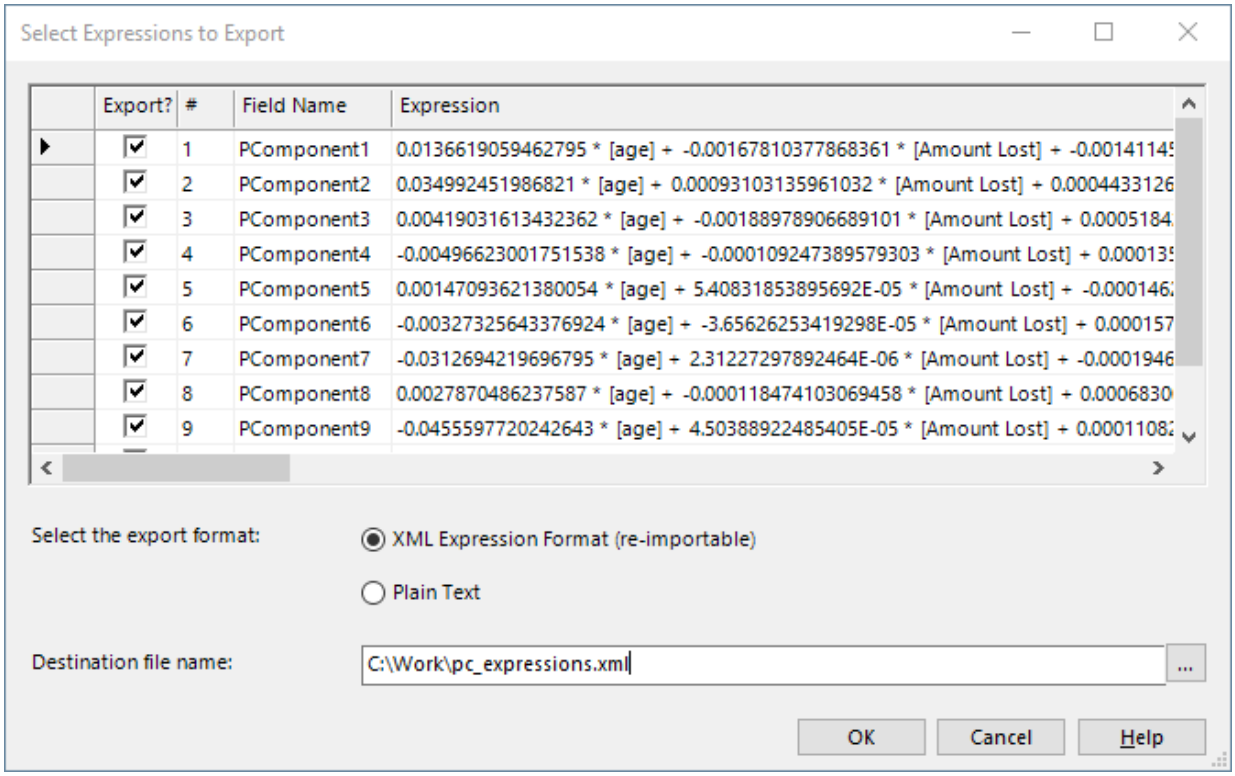
Exporting Calculations From PCA and Factor Analysis

Once the principal components or factors are computed by the PCA or Factor Analysis node, you can export their calculated expressions to a file. They can be exported either as plain text or re-importable XML Expression Format. The expressions in XML format can be used in a transformation node to create component columns or factor columns in any dataset that contains the variables referenced by the expressions.

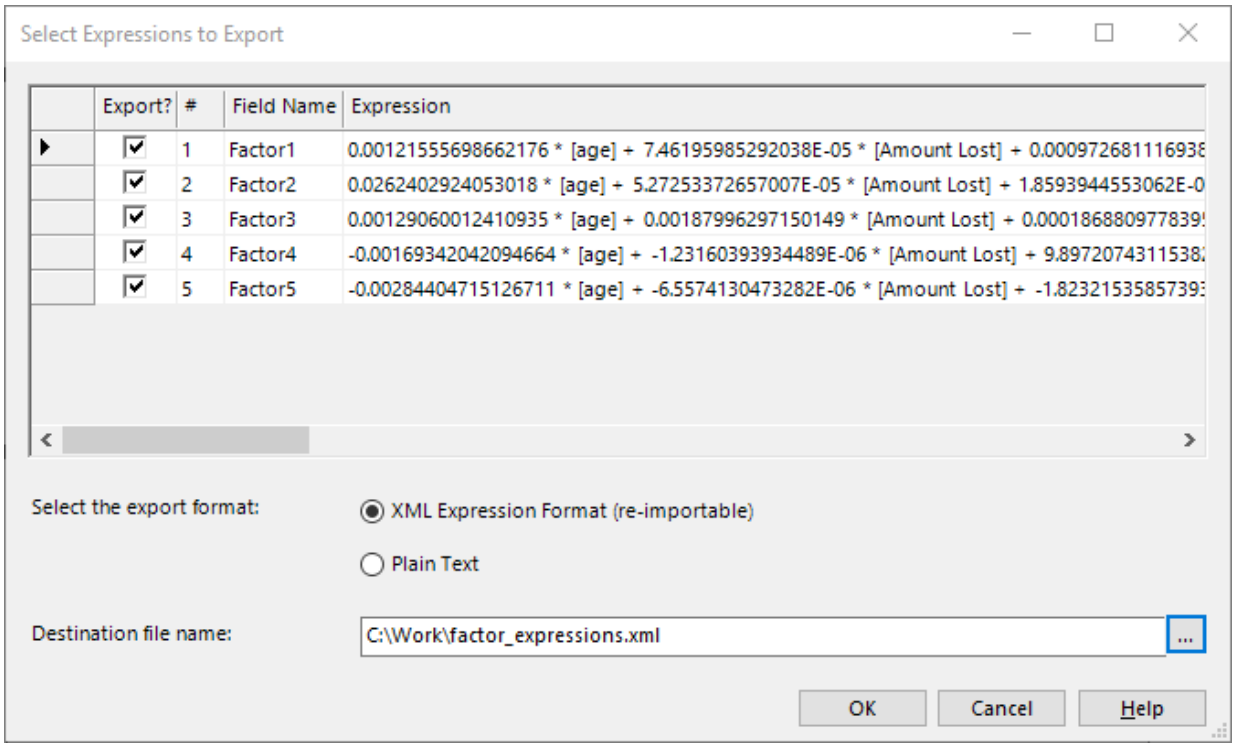
To generate and export the calculations, double-click the PCA or Factor Analysis node in your workflow. Then select the command **Calculation From PCA..** or **Calculation From FA..** from the **Tools** menu.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Example 1: Exporting PCA expressions



Example 2: Exporting factor expressions



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The export dialog displays the generated expressions:

- For principal components, the coefficients in the expressions are taken from the columns of the **Components** table of the PCA node (see PCA: Components).
- For factors, the coefficients in the expressions are taken from the columns of the **Scores** table of the Factor Analysis node (see Factor Analysis: Scores).

Select the expressions to export.

The *Field Name* and *Expression* columns are editable. If required, rename the variables or edit the corresponding expressions.

Choose the export format (plain text or XML) and specify the destination file name. Only files in the re-importable XML Expression Format can be imported into the Variable Transformations node for creating new columns.

See also:

Scoring a Factor Analysis Model

Scoring a Principal Component Analysis Model

Scoring a Principal Component Analysis Model

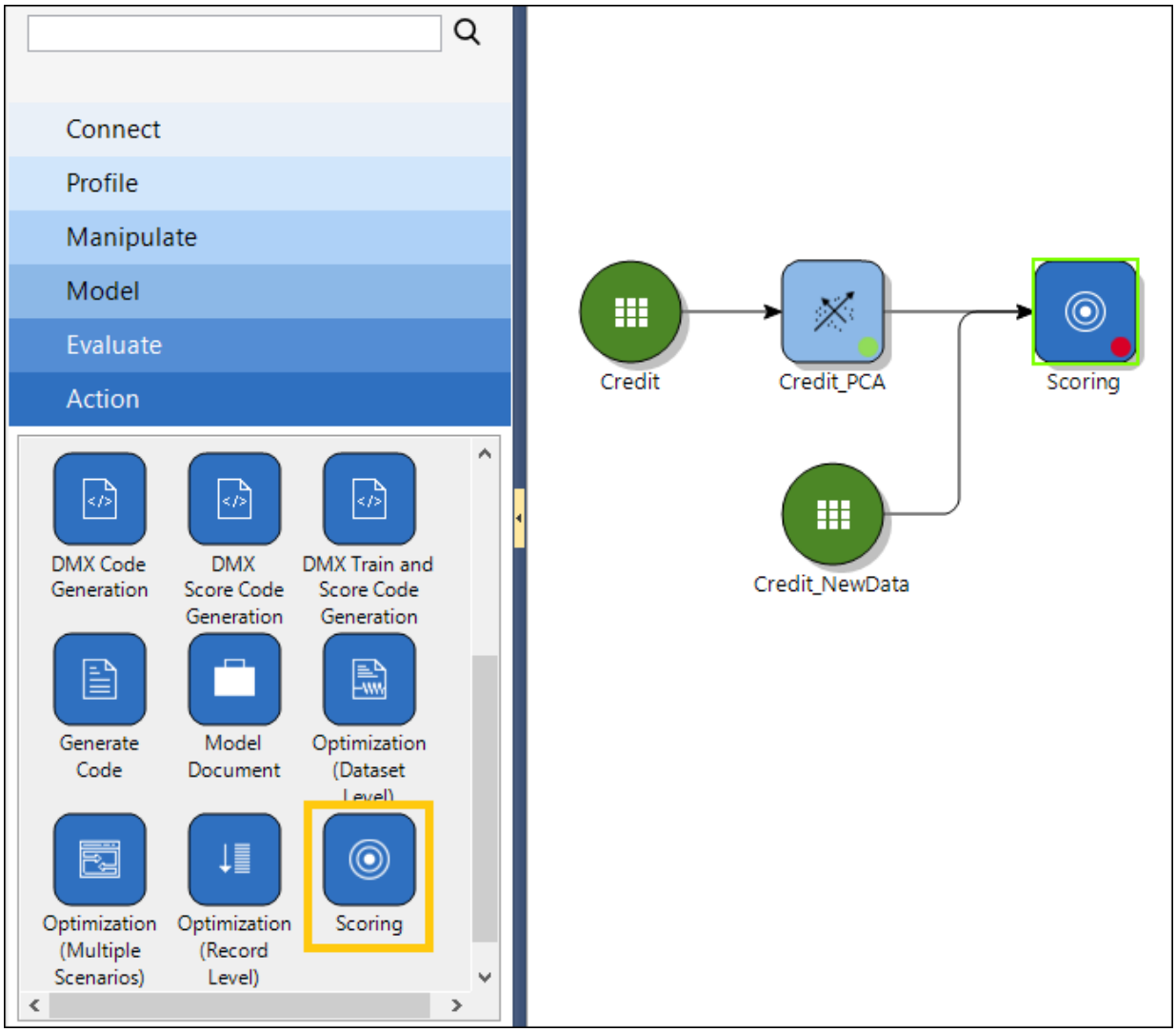
By ***Scoring*** a Principal Component Analysis model (PCA) we understand applying the PCA transformations defined by the model to an input dataset and generating a new dataset with columns based on the PCA scores. See the topic PCA Scores view. Within the workflow concept, the use of the term "scoring" for PCA models is consistent with its usual meaning of model deployment.

Code generation for PCA models for deploying in other environments is discussed in the Gode Generation topic.

To score a PCA model (apply PCA transformations), connect the PCA model and an input dataset to a **Scoring** node and run it. The input dataset can be the same as the one used to build the PCA model or a different one - as long as it has the necessary input fields.

First, drag the **Scoring** node from the **Action** palette to the Workflow canvas. Then connect the PCA model and the input dataset to the Scoring node:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

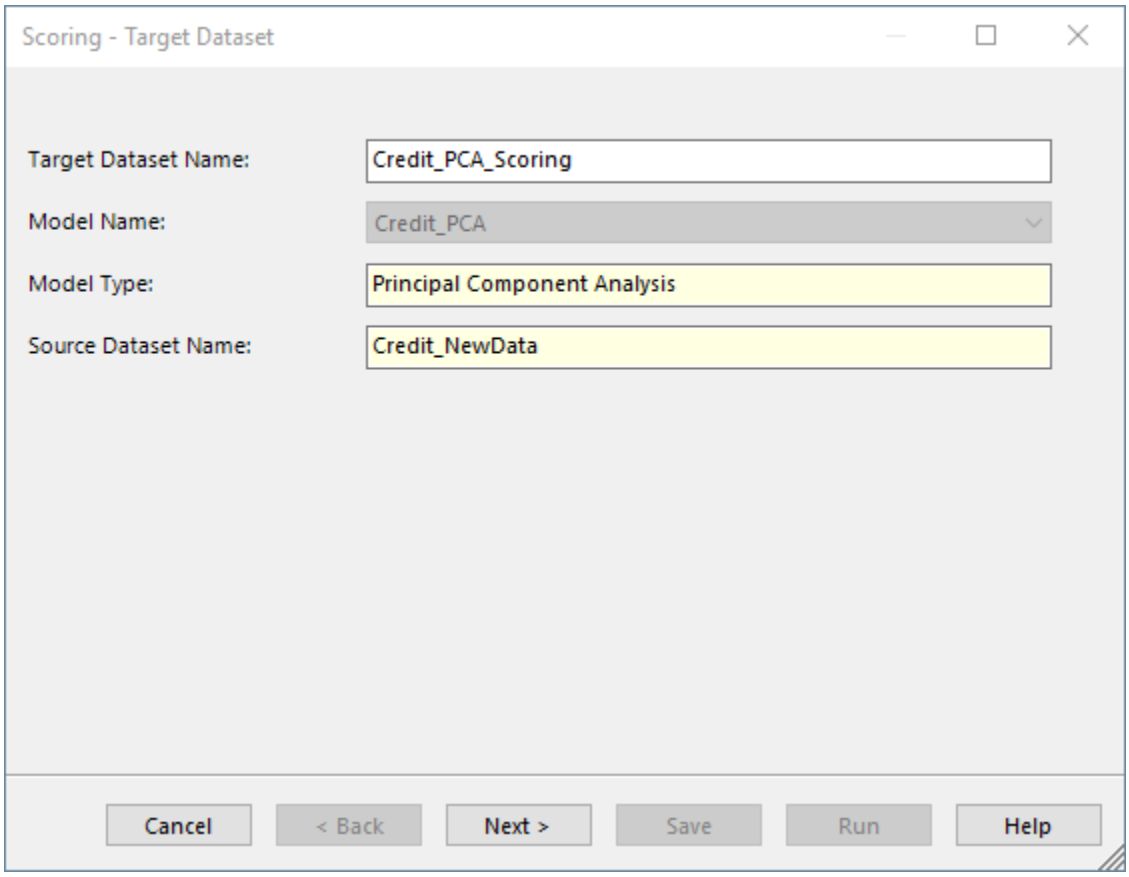


Double-click on the Scoring node or right-click on it and select **Modify**. This opens the Scoring wizard for PCA.

Step 1

In the first page of the wizard, specify the name for the output dataset in the Target Dataset Name field.

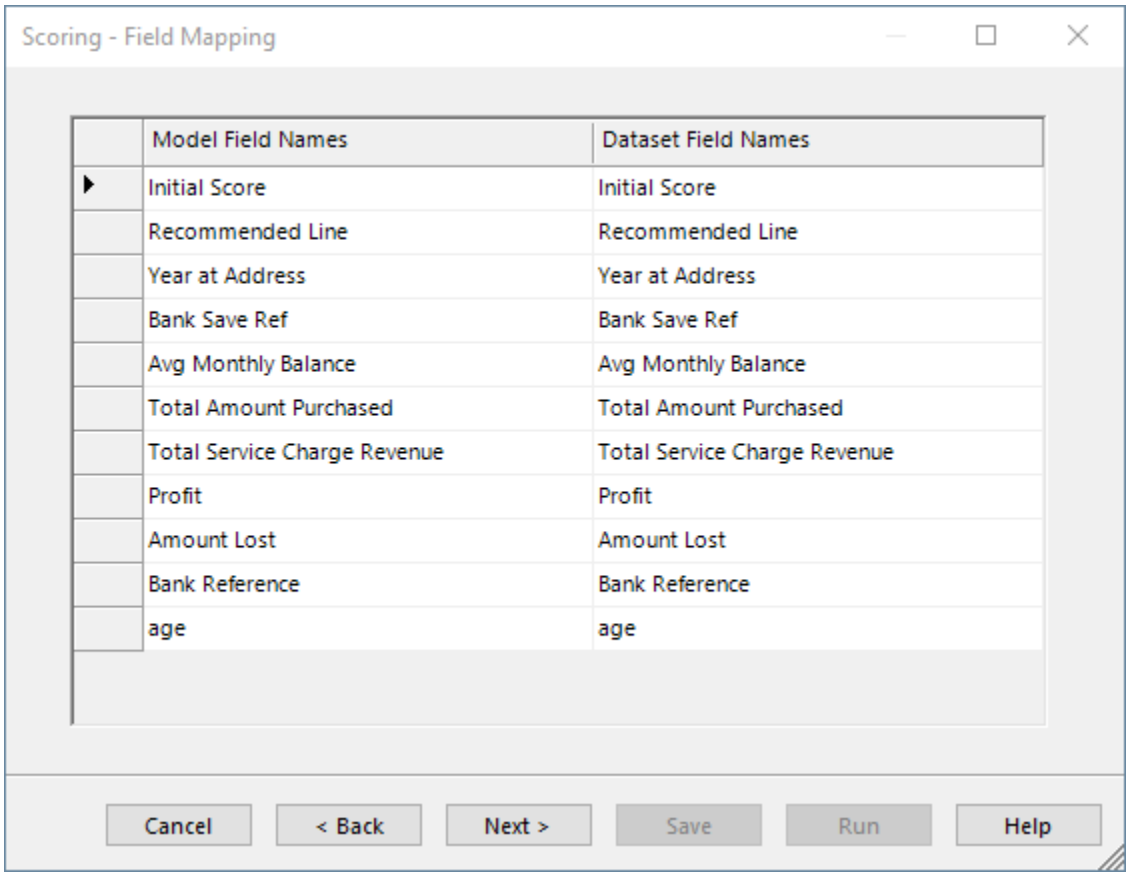
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Step 2

The next step provides the option of mapping the field names in the model to the ones in the dataset to be scored. In some situations, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map fields by clicking in the right column (Dataset Field Names) and selecting the right fields to match the model variables in the left column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



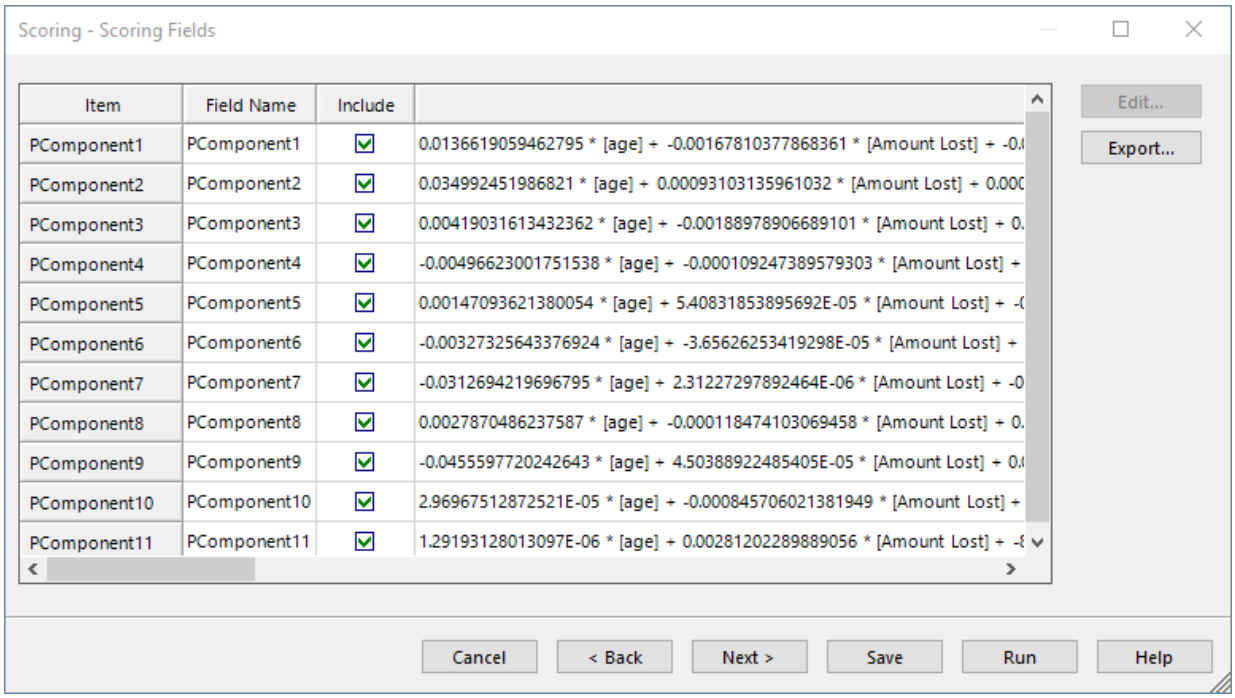
Step 3

At the next step, choose the new Principal Component Fields to be generated and included in the output dataset. The default field names are *PComponent1, ..., PComponentN*. These are calculated columns with expressions defined by the coefficients (scores) displayed in the Scores view of the input Principal Component Analysis model.

You can rename the new fields in the Field Name column and even edit any of the expressions by selecting it and clicking the **Edit** button. The Edit button opens the Expression Editor for the selected expression.

To save the expressions in a file, click the Export button and specify the file name. For details on exporting the PCA expressions, see the topic Inserting Calculations From PCA.

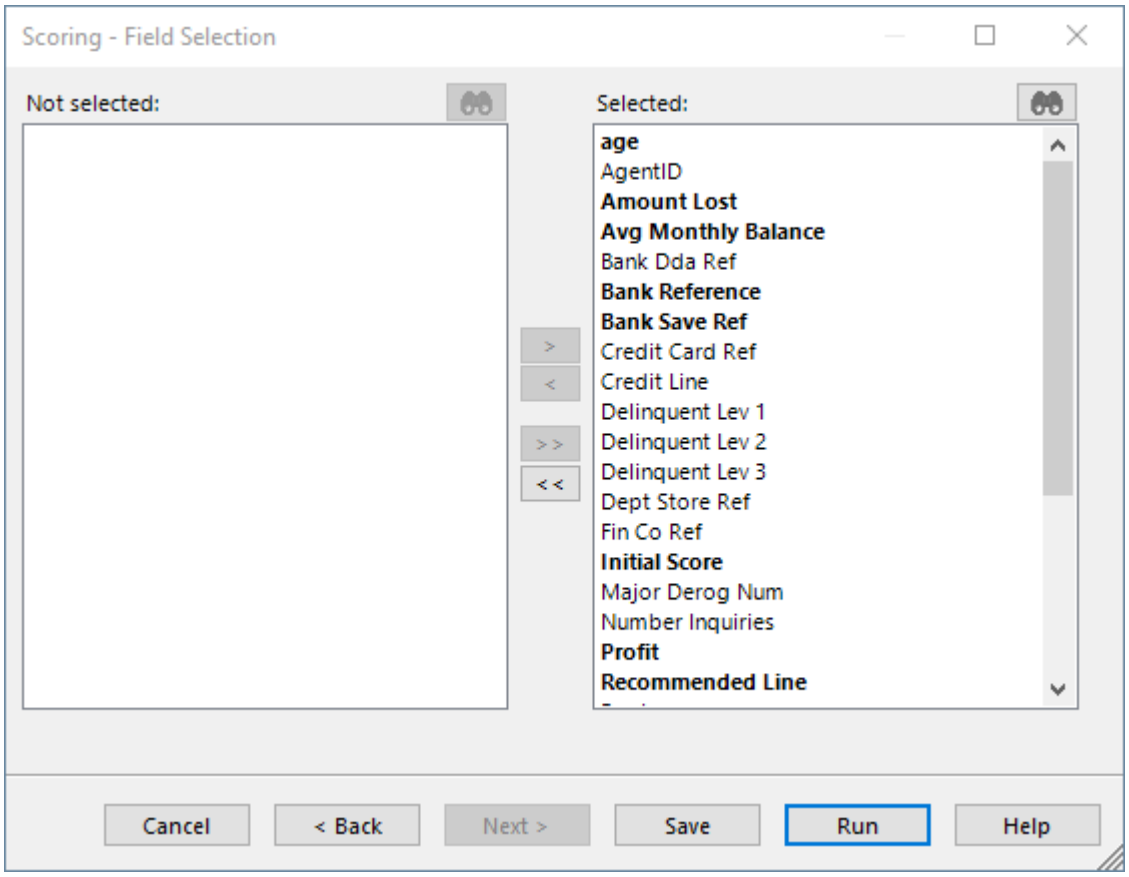
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Step 4

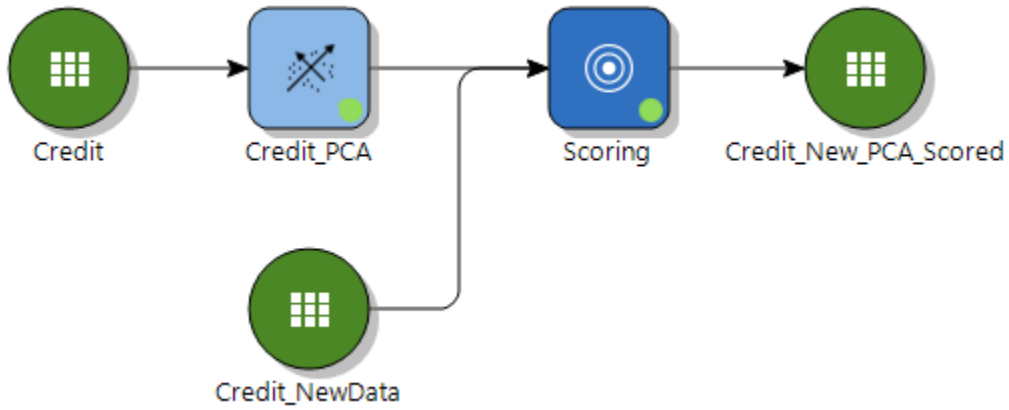
At the **Field Selection** step, choose the fields from the input dataset that you would like to include in the output dataset in addition to the generated Principal Component fields. The fields required by the model are highlighted in bold font and cannot be excluded. By default, all fields in the scoring input dataset are selected to be included in the output.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the scoring process immediately.

The output dataset node will be created in the workflow after running the scoring node. Double-click the output dataset to view the results..



See also:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Exporting Calculations From PCA and Factor Analysis

Code Generation (Section 4: "*Code Generation for Factor Analysis and Principal Component Analysis*")

Factor Analysis

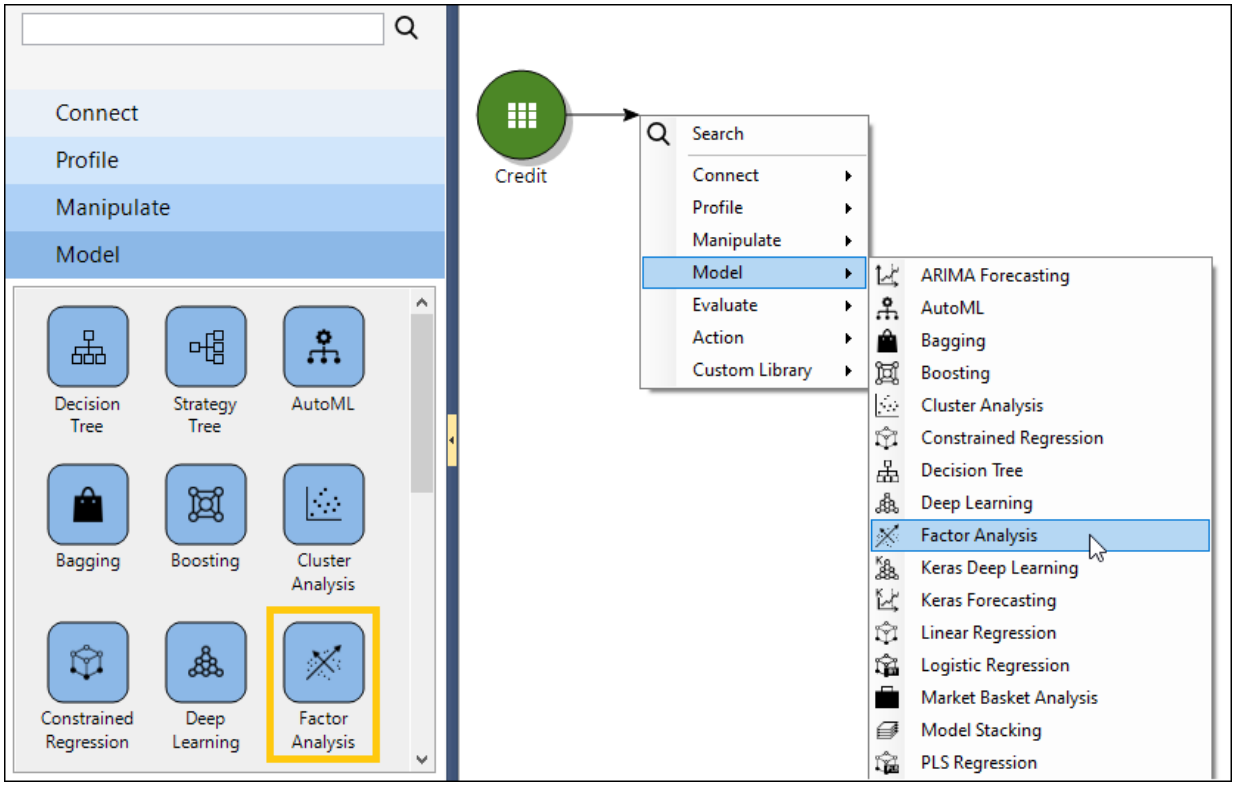
Inserting Factor Analysis

To add a **Factor Analysis** node to the workflow:

- Drag a connection arrow from a dataset node to any area on the Workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Model --> Factor Analysis** as in the example screenshot below. A new **Factor Analysis** node will be created and the wizard will open.

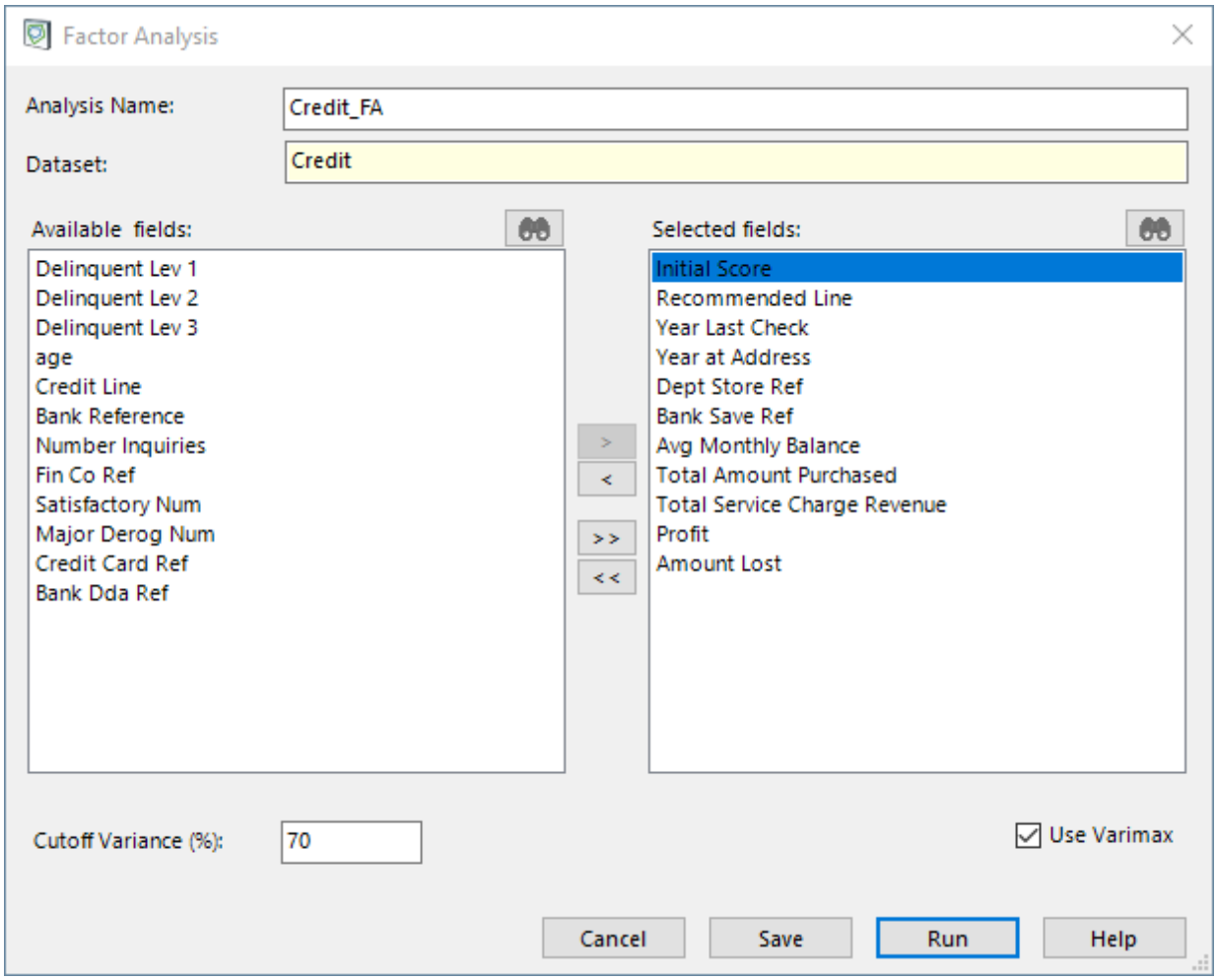
OR

- Open the **Model** palette and drag the **Factor Analysis** node onto the workflow canvas. Connect the input dataset to the new node.



Double-click the Factor Analysis node or right-click it and select **Modify** to open the wizard.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Analysis Name	Specify the name for the Factor Analysis node.
Dataset	The name of the input dataset connected to the Factor Analysis node.

Select the fields to be included in the Factor Analysis (PCA). Only numeric variables can be used for the PCA. If the correlations tab in the dataset is active and initiated, only the variables used in the correlations tab will be selected by default.

Variables listed on the right will be included in the analysis. You can move the variables to and from the 'Selected' list by highlighting them and using the arrow buttons. Double-clicking the individual variables will also move them from one list to the other.

To select more than one variable in the lists, use mouse dragging, or mouse clicks while holding the SHIFT key or the CTRL key. If there are a lot of variables (more than a hundred, for example), it is convenient to use **Field Finder** (the binoculars button).

Choose whether to include ***Varimax*** rotation (checked by default) and specify the cutoff variance.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Cutoff Variance (%)	Specifies the percent of variance to explain. The default is 70%. Changing this number can affect the number of factors found.
Use Varimax	Specifies whether to use Varimax rotation. It is turned on by default.

Click **Save** to save the settings, or click **Run** to save the settings and run the model immediately.

To view the resulting Factor Analysis model, right-click on the Factor Analysis node and select the **Open View** command from the context menu.

See the topic Factor Analysis: Summary Report

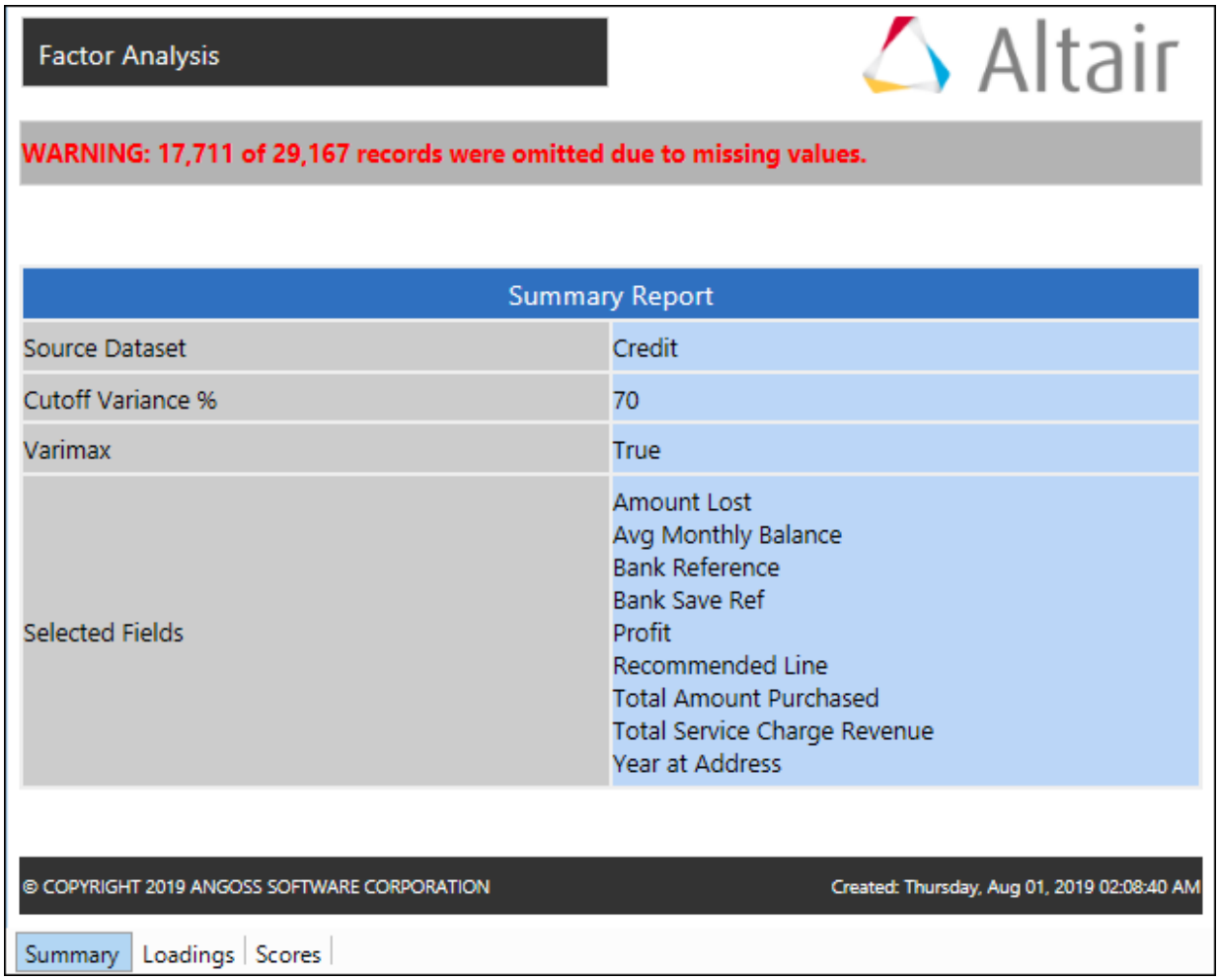
See also:

- Factor Analysis: Loadings
- Factor Analysis: Scores

The **Summary** tab displays the report of the Factor Analysis settings that were specified in the Factor Analysis node wizard: the source dataset, cutoff variance, Varimax rotation indicator, and selected fields.

Note that records with missing values are excluded from the analysis. A warning in the report indicates the number of records were omitted due to missing values.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also:
Factor Analysis: Loadings
Factor Analysis: Scores

The table in the **Loadings** tab is the matrix of correlations between the original variables and the factors.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Absolute Display Threshold: 0

	Factor1	Factor2	Factor3	Factor4	Factor5
age	-0.04250	0.82170	-0.00292	0.02368	-0.05814
Amount Lost	0.27499	-0.03766	0.95954	-0.00709	0.00201
Avg Monthly Balance	0.90744	-0.09476	0.32295	-0.02412	0.00371
Bank Reference	-0.00182	-0.01757	0.00256	0.07171	0.99074
Bank Save Ref	-0.04629	0.08542	0.00102	0.96494	0.07598
Initial Score	-0.10991	0.80486	-0.04282	0.16510	-0.04502
Profit	0.05782	0.00795	-0.99357	-0.00732	-0.00134
Recommended Line	0.08975	0.67232	-0.01319	-0.21835	0.09882
Total Amount Purchased	0.78106	0.04666	-0.06120	-0.01712	-0.00243
Total Service Charge Revenue	0.92774	-0.10488	0.04287	-0.04287	0.00276
Year at Address	-0.05229	0.56982	-0.00342	0.09546	-0.00867
Variance	2.39999	2.13095	2.01982	1.02367	1.00262
% Variance	21.81813	19.37223	18.36196	9.30613	9.11469

Loadings relate the specific association between the factors and the original variables. Therefore, it is necessary to find the loadings, then solve for the factors, which will approximate the relationship between the original variables and underlying factors. The loadings are derived from the magnitude of eigenvalues associated to individual variables.

Display option: Absolute Display Threshold

The Loadings table displays the values whose absolute values are greater than or equal to the Absolute Display Threshold. By default, the threshold value is zero (i.e., all the values are displayed).

See also: Factor Analysis: Scores

The **Scores** view displays the coefficients of each variable in the linear combinations that are the resulting factors.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

	Factor1	Factor2	Factor3	Factor4	Factor5
age	0.00122	0.02624	0.00129	-0.00169	-0.00284
Amount Lost	0.00007	0.00005	0.00188	0.00000	-0.00001
Avg Monthly Balance	0.00097	0.00000	0.00019	0.00010	-0.00002
Bank Reference	-0.21074	0.21777	-0.00245	-0.96694	22.62360
Bank Save Ref	0.14161	-0.06591	-0.02024	2.93382	-0.10256
Initial Score	0.00001	0.00482	0.00004	0.00148	-0.00059
Profit	0.00055	-0.00006	-0.00219	0.00004	0.00000
Recommended Line	0.00013	0.00082	0.00000	-0.00065	0.00034
Total Amount Purchased	0.00052	0.00008	-0.00017	0.00005	-0.00001
Total Service Charge Revenue	0.00520	-0.00010	-0.00101	0.00032	-0.00008
Year at Address	0.00087	0.03159	0.00165	0.00681	-0.00075
Constant Coefficient	-0.80781	-3.86210	-0.04511	-1.74422	-22.37859

Factor Analysis computes all the **loadings** for the final purpose of producing new columns on the basis of the input variables. These columns are linear combinations of the original columns, plus a constant term. This constant represents the mean subtracted from the data points in the beginning of the **Factor Analysis** process.

Example of a factor representation:

```
Factor1 = 0.00121555698662176 * [age] + 7.46195985292038E-05 * [Amount Lost] + 0.000972681116938105 * [Avg Monthly Balance] + -0.2107351552992 * [Bank Reference] + 0.141609168095386 * [Bank Save Ref] + 9.13868640834794E-06 * [Initial Score] + 0.000550516945929114 * [Profit] + 0.00013001784033818 * [Recommended Line] + 0.000518594427807931 * [Total Amount Purchased] + 0.00520305354938283 * [Total Service Charge Revenue] + 0.000874942287725498 * [Year at Address] + -0.807810598321399
```

You can take note of variables with the highest factor scores and see if this is in agreement with your expectations according to the business logic.

From this view, you can generate the expression for each factor in terms of the original variables using the command **Calculation From FA** from the **Tools** menu. See the topic Inserting Calculations From Factor Analysis.

See also:
Factor Analysis: Loadings

Exporting Calculations From PCA and Factor Analysis

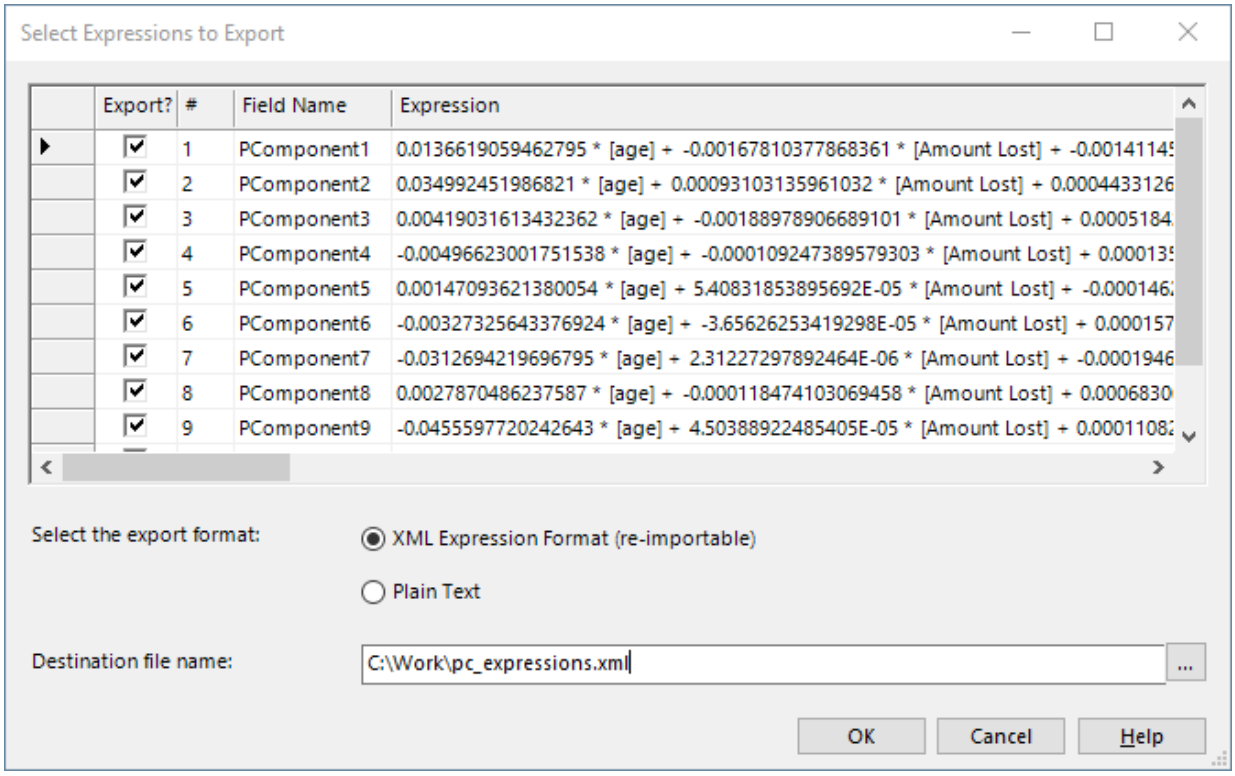
Once the principal components or factors are computed by the PCA or Factor Analysis node, you can export their calculated expressions to a file. They can be exported either as plain text or re-importable XML Expression Format. The expressions in XML format can be used in a

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

transformation node to create component columns or factor columns in any dataset that contains the variables referenced by the expressions.

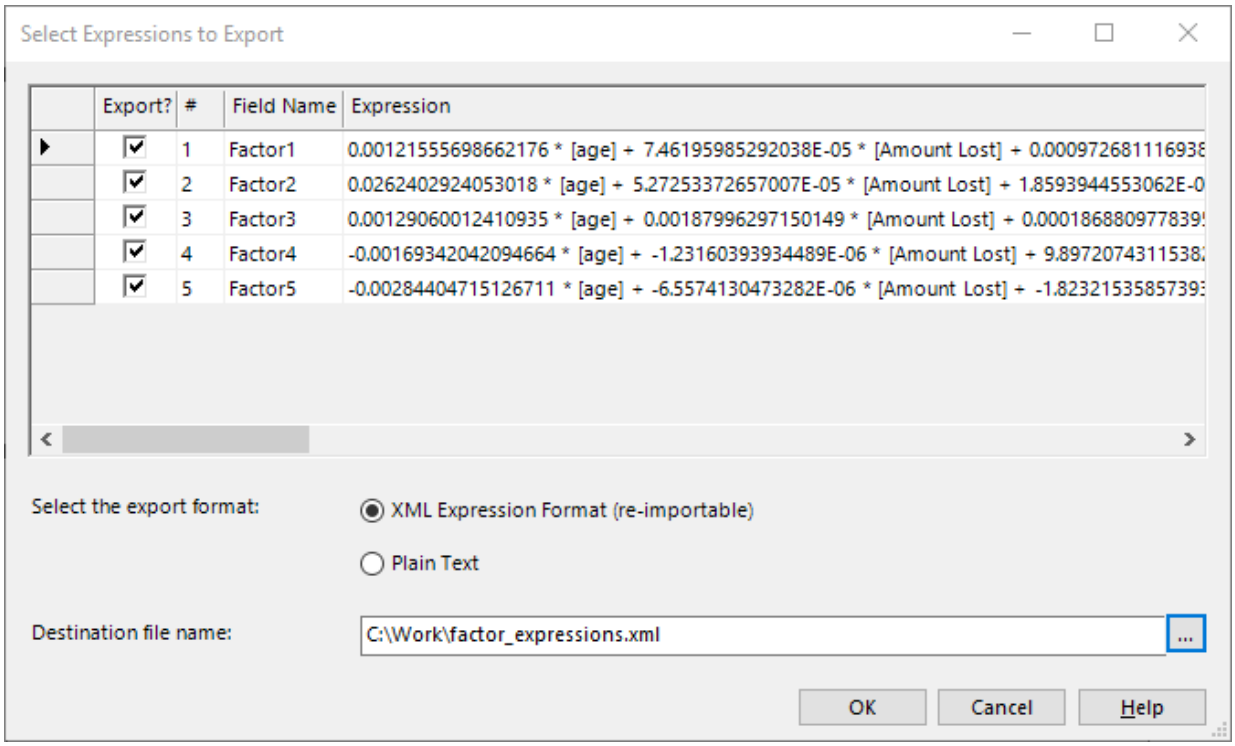
To generate and export the calculations, double-click the PCA or Factor Analysis node in your workflow. Then select the command **Calculation From PCA..** or **Calculation From FA..** from the **Tools** menu.

Example 1: Exporting PCA expressions



Example 2: Exporting factor expressions

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The export dialog displays the generated expressions:

- For principal components, the coefficients in the expressions are taken from the columns of the **Components** table of the PCA node (see PCA: Components).
- For factors, the coefficients in the expressions are taken from the columns of the **Scores** table of the Factor Analysis node (see Factor Analysis: Scores).

Select the expressions to export.

The *Field Name* and *Expression* columns are editable. If required, rename the variables or edit the corresponding expressions.

Choose the export format (plain text or XML) and specify the destination file name. Only files in the re-importable XML Expression Format can be imported into the Variable Transformations node for creating new columns.

See also:

Scoring a Factor Analysis Model

Scoring a Principal Component Analysis Model

Scoring a Factor Analysis Model

By ***Scoring*** a Factor Analysis model we understand applying the Factor Analysis transformations defined by the model to an input dataset and generating a new dataset with columns based on the Factor Analysis scores. See the topic Factor Analysis: Scores. Within the workflow concept, the

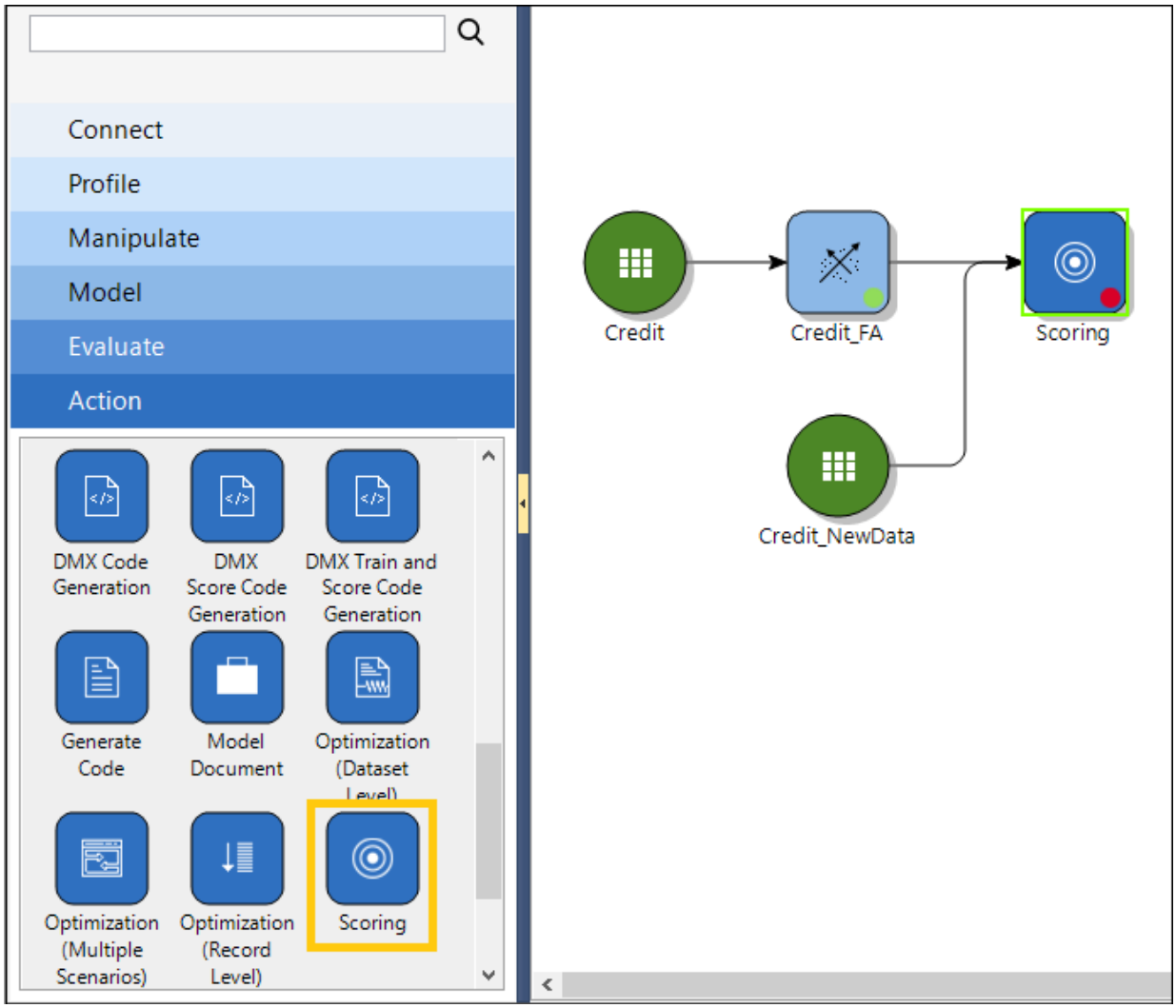
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

use of the term "scoring" for Factor Analysis models is consistent with its usual meaning of model deployment.

Code generation for PCA models for deploying in other environments is discussed in the Gode Generation topic.

To score a Factor Analysis model (apply Factor Analysis transformations), connect the Factor Analysis model and an input dataset to a Scoring node and run it. The input dataset can be the same as the one used to build the PCA model or a different one - as long as it has the necessary input fields.

First, drag a **Scoring** node from the **Action** palette to the Workflow Canvas. Then connect the Factor Analysis model and the input dataset to the Scoring node:

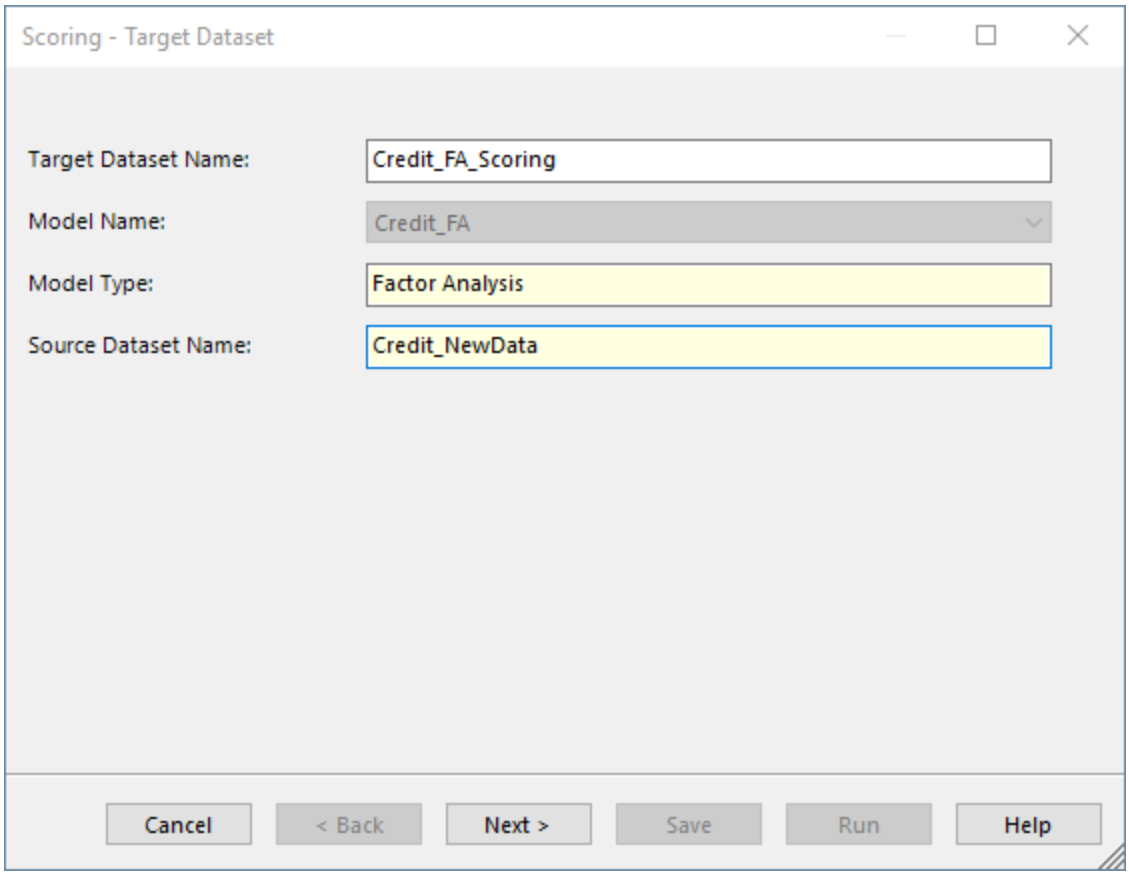


Double-click on the Scoring node or right-click on it and select **Modify**. This opens the Scoring wizard for Factor Analysis.

Step 1

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

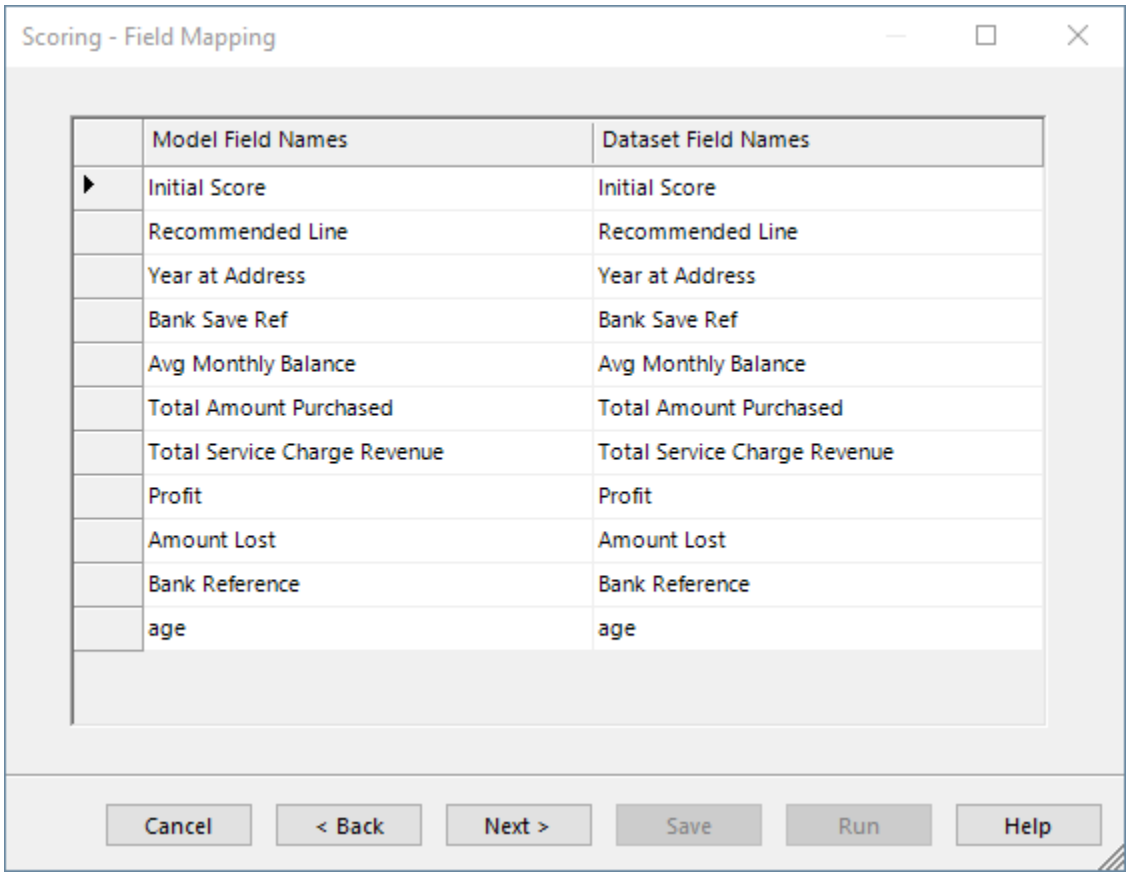
In the first page of the wizard, specify the name for the output dataset in the Target Dataset Name field.



Step 2

The next step provides the option of mapping the field names in the model to the ones in the dataset to be scored. In some situations, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map fields by clicking in the right column (Dataset Field Names) and selecting the right fields to match the model variables in the left column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



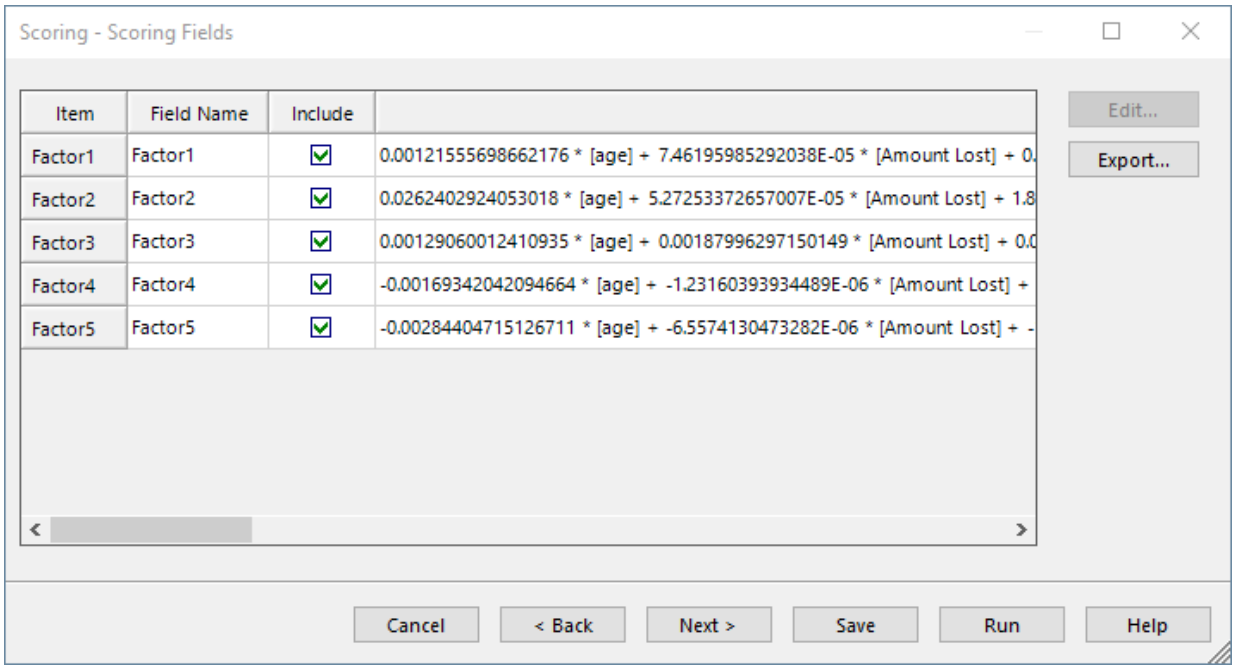
Step 3

At the next step, choose the new factor fields to be generated and included in the output dataset. The default field names are *Factor1, ..., FactorN*. These are calculated columns with expressions defined by the coefficients (scores) displayed in the Scores view of the input Factor Analysis model.

You can rename the new fields in the Field Name column and even edit any of the expressions by selecting it and clicking the **Edit** button. The Edit button opens the Expression Editor for the selected expression.

To save the expressions in a file, click the Export button and specify the file name.

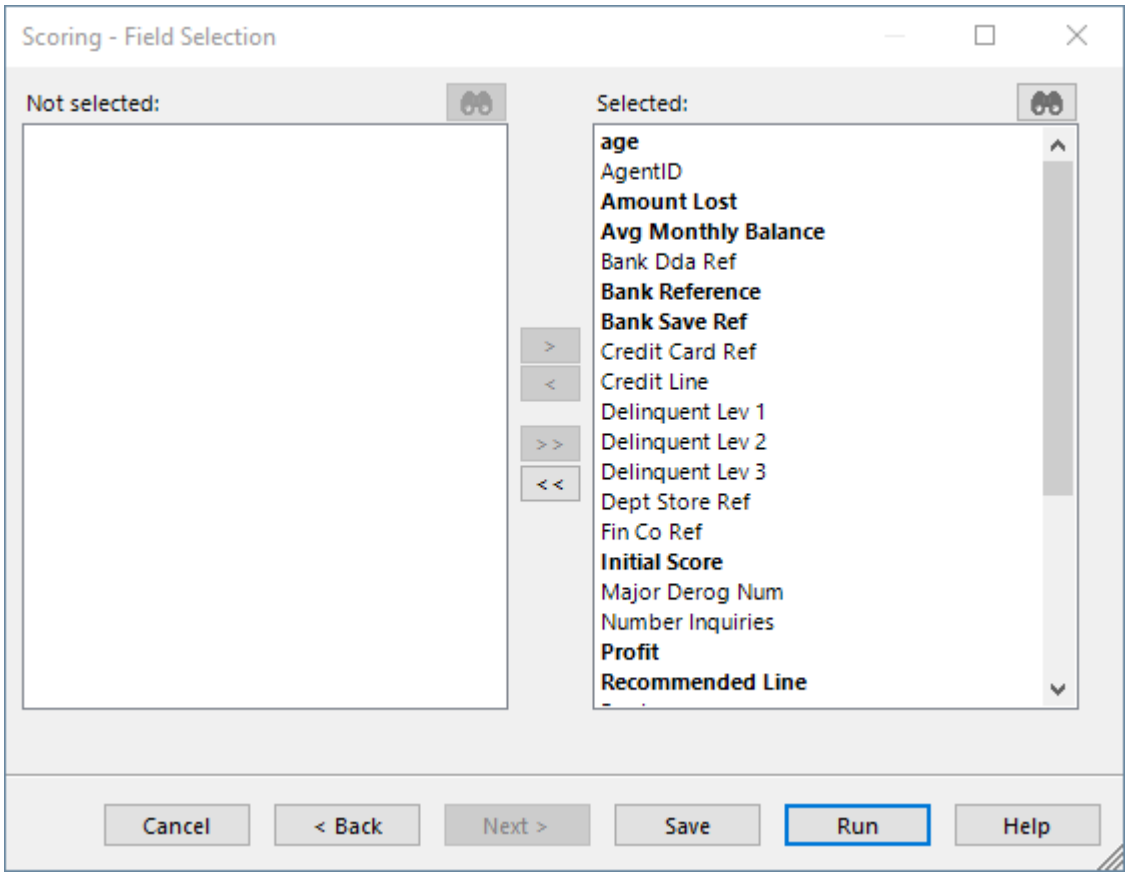
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Step 4

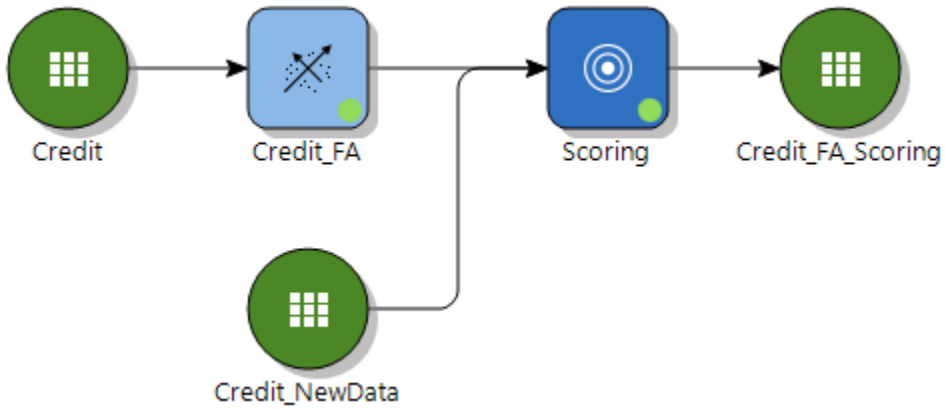
At the **Field Selection** step, choose the fields from the input dataset that you would like to include in the output dataset in addition to the generated Factor fields. The fields required by the model are highlighted in bold font and cannot be excluded. By default, all fields in the scoring input dataset are selected to be included in the output.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the scoring process immediately.

The output dataset node will be created in the workflow after running the scoring node. Double-click the output dataset to view the results..



See also:

[Exporting Calculations From PCA and Factor Analysis]

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Code Generation (Section 4: "*Code Generation for Factor Analysis and Principal Component Analysis*")

Time Series

Time Series Decomposition

The **Time Series Decomposition** node performs transformation of time series data into several component series: Series, Trend, Seasonal, and Residuals (noise). The forecast is performed on the user-defined train data partition and out-of-sample data partition of the input dataset.

It is assumed that the input dataset for Time Series Decomposition is a ***time series dataset*** containing metadata that identifies one column as the time index (series ID) and another column as the time label. Such datasets can be created, for example, using the Make Time Series node in Knowledge Studio. These datasets have special views (integrated tabs) where time series charts and autocorrelation charts van be plotted.

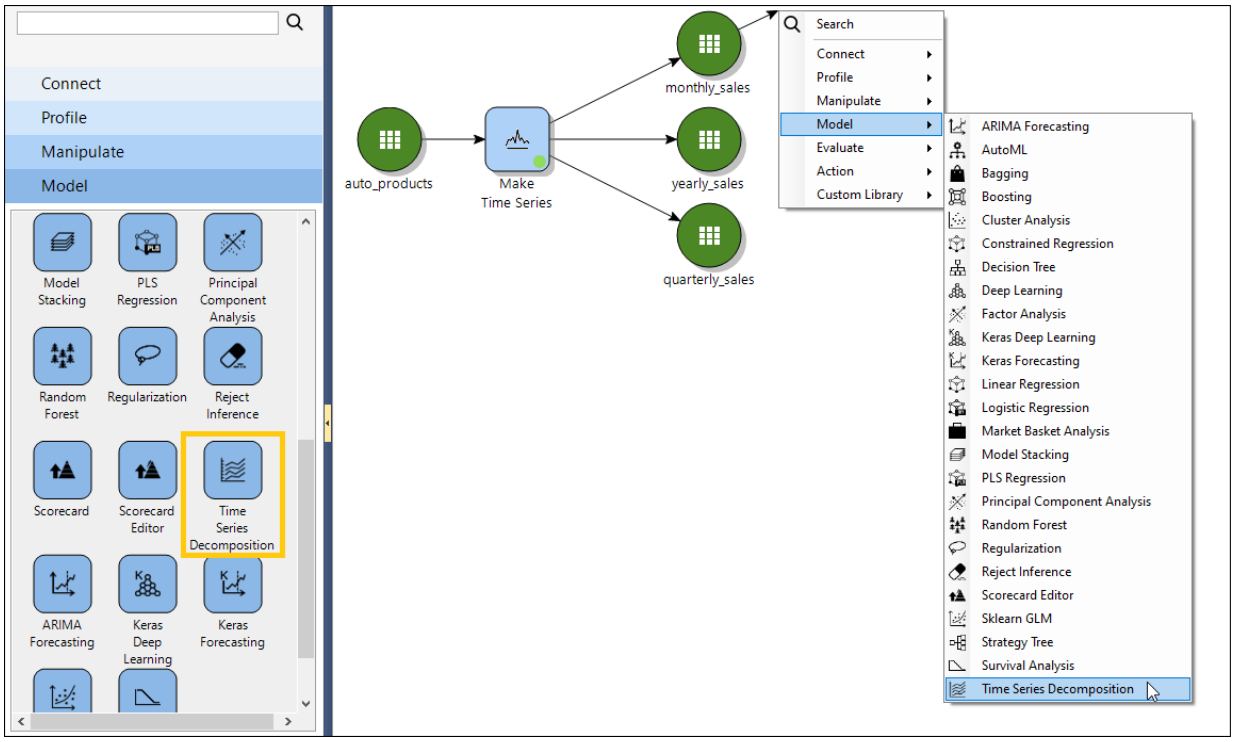
To add a **Time Series Decomposition** node to the workflow:

- Drag a connection arrow from a dataset node that represents a time series dataset to any area on the Workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Model --> Time Series Decomposition** as in the example screenshot below. A new **Time Series Decomposition** node will be created, and the wizard will open.

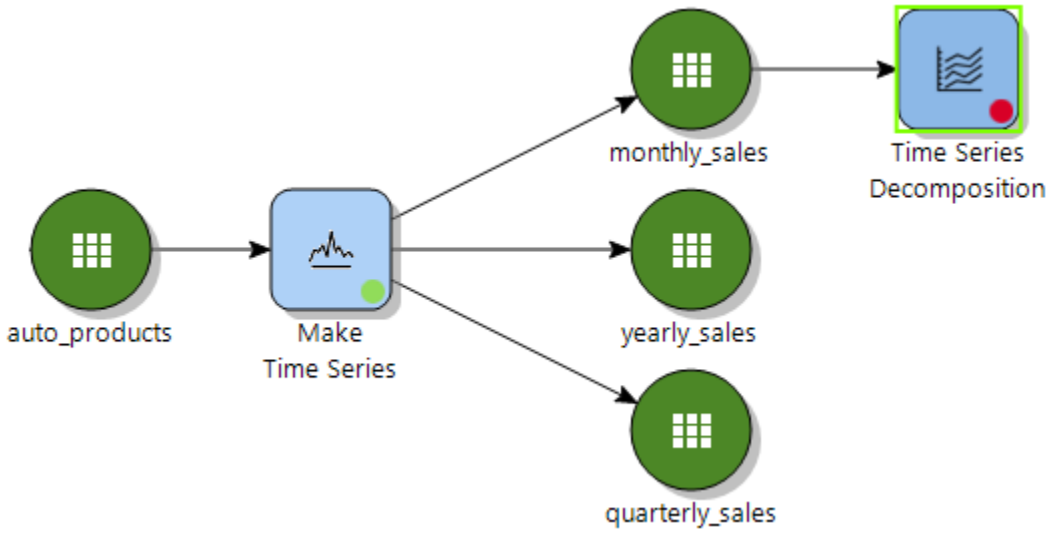
  OR

- Open the **Model** palette and drag the **Time Series Decomposition** node onto the workflow canvas. Connect the input dataset that represents a time series dataset to the Time Series Decomposition node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

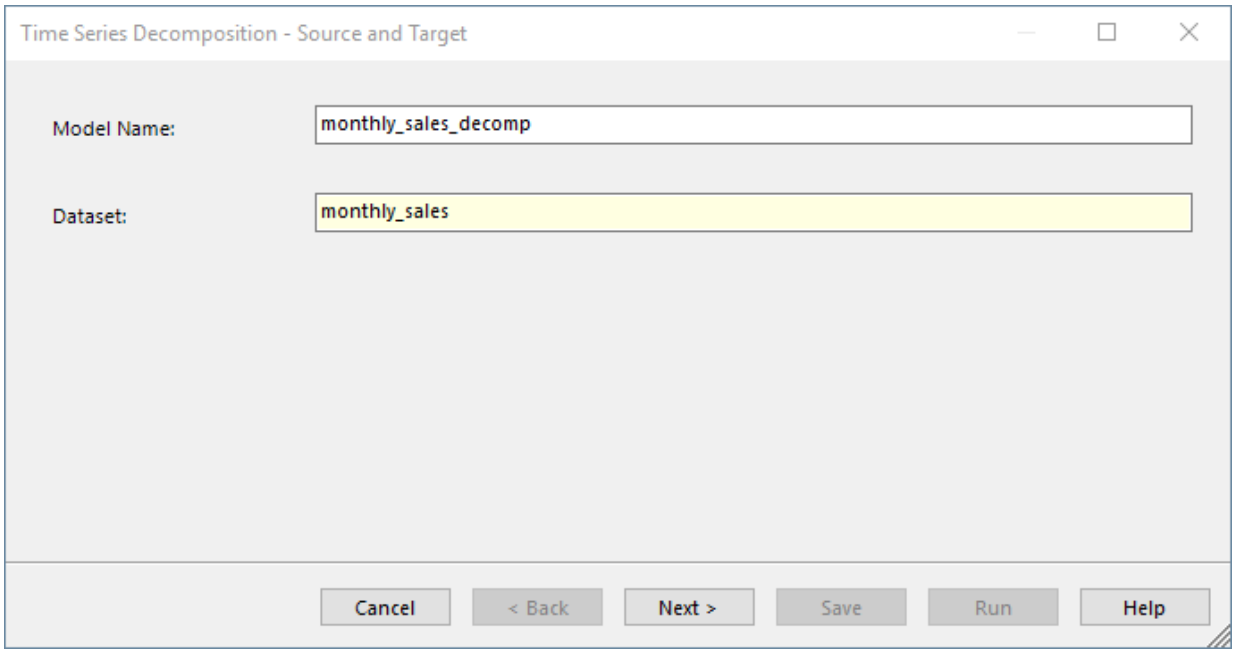


If the Time Series Decomposition node is already on the canvas, double-click it or right-click on it and select **Modify** from the context menu to open the wizard.



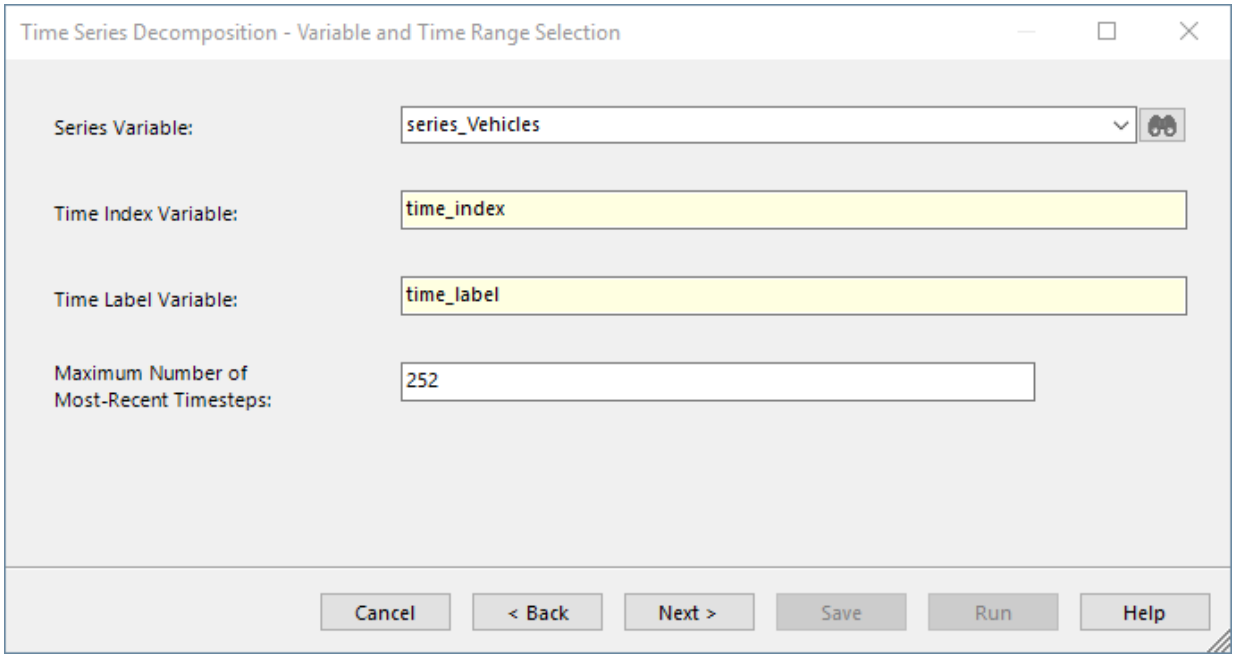
In the *Model Name* field, specify the name for the Time Series Decomposition node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



In the second page of the wizard:

- Select the desired series variable for decomposition.
- Specify the maximum number of most recent timesteps (the number of time steps from the end of the series - the rows with data to be used for building the models).
- The time index variable (series ID) and time label variable to be forecast are selected automatically, assuming the input data is in the correct format.



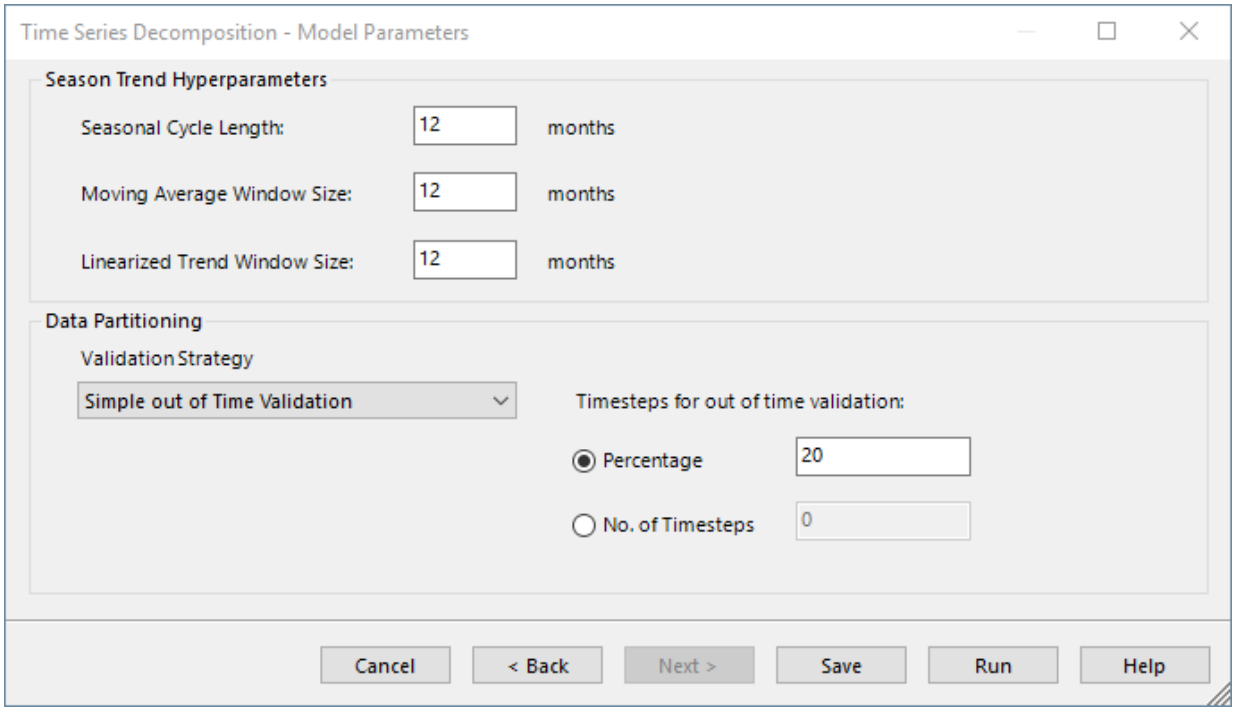
Click **Next**.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

In the Model Parameters page:

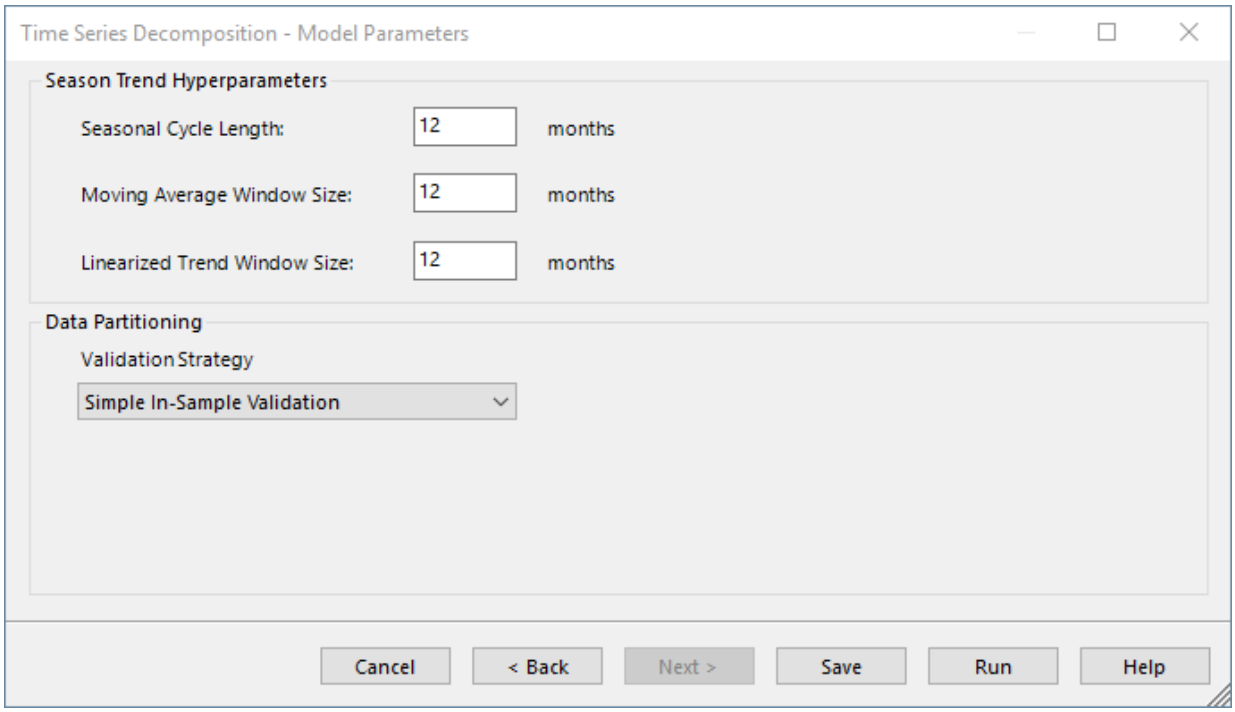
- Specify the season trend hyperparameters:
  - Seasonal cycle length. The units depend on whether you are working on quarterly data, monthly data, etc. The smallest supported time unit is millisecond.
  - Moving average window size. The units are the same as for the seasonal cycle length.
  - Linearized trend window size. For example, for monthly data, enter 12 if you want to go back 12 months to establish the trend.
- Specify the data partitioning parameters:
  - Select the validation strategy. There are three options: "*Simple out-of-time validation*", "*Simple in-sample validation*", and "*Flexible Split*". The default is "*Simple out-of-time validation*".

If you selected "*Simple out-of-time validation*", specify either the percentage of data or the number of records to be set as out-of-time validation. The example below is for monthly data.



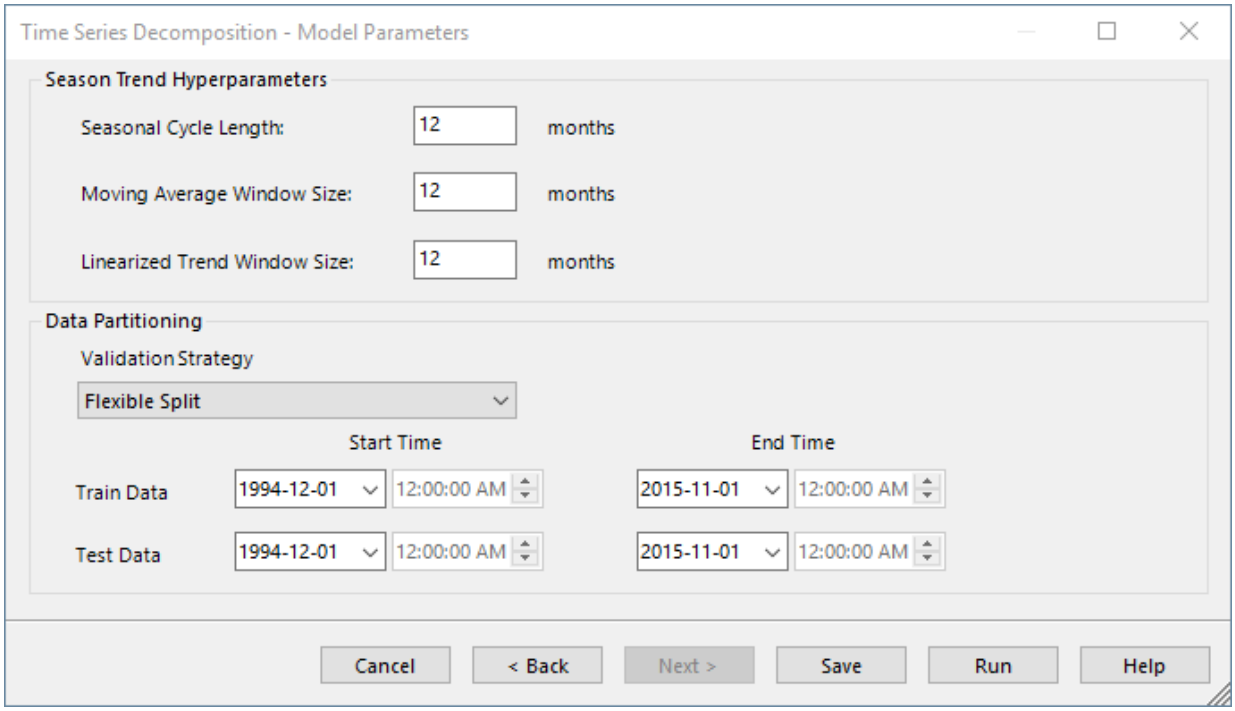
If you chose "*Simple in-sample validation*", then all available data is used for both training and validation.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



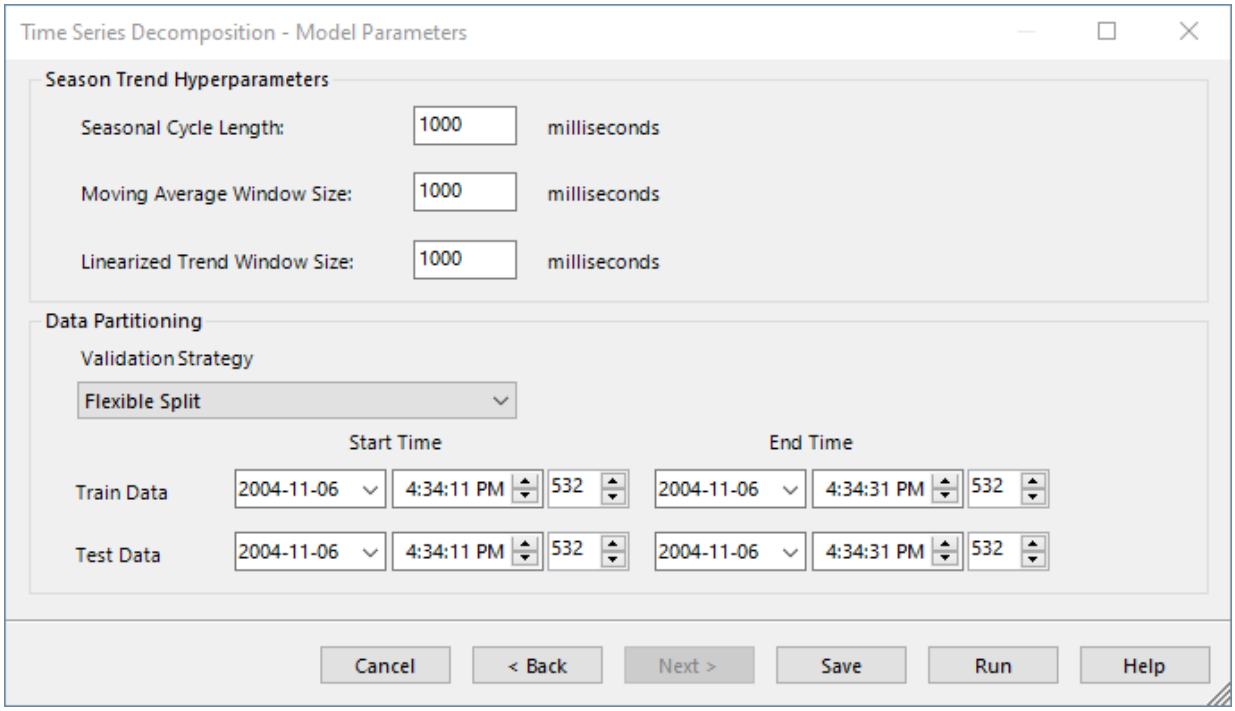
If you chose the "*Flexible Split*" validation strategy, specify the exact start and end times for the train data sample and the test data sample. If the series has numeric time steps rather than date/time, then enter numeric values.

Note: The start and end times of the test (validation) window must not be before the start and end times of the training window, respectively.

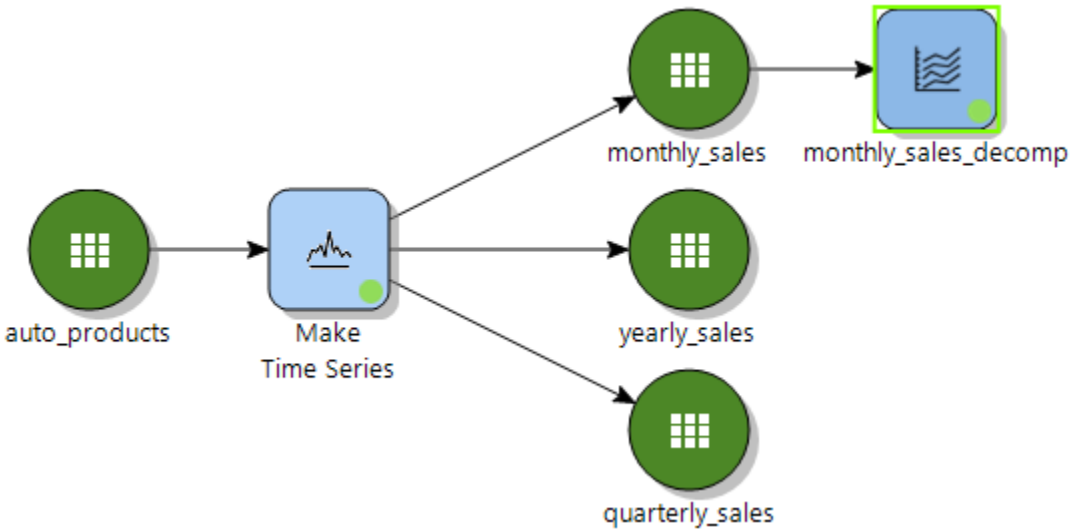


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If your source data contains Time or Timestamp values with milliseconds, the start and end times for the train and test data samples can be specified at the millisecond level. The millisecond part is specified in a separate field. An example is shown below.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the node, double-click on it to view the plots.



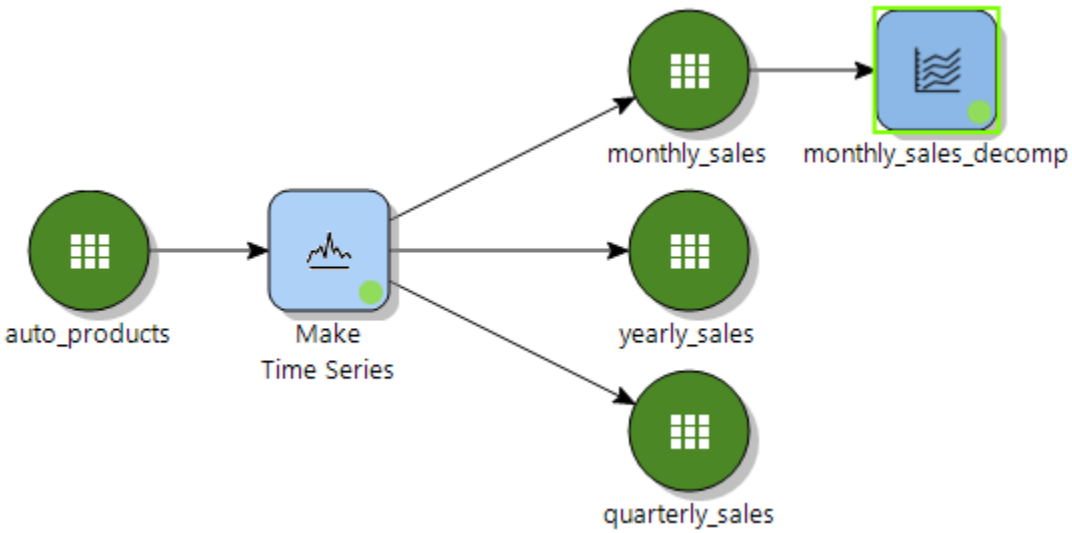
Proceed to the topic Time Series Decomposition Plots.

See also: Make Time Series

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Time Series Decomposition Plots

After you have built and run the Time Series Decomposition node, double-click on it to view the interactive plots.



The Time Series Decomposition object has three tabs - **Fit**, **Evaluate**, and **Forecast**.

Fit

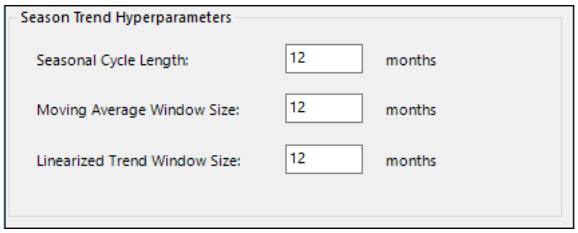
The **Fit** tab shows interactive plots for Series, Trend, Seasonal, and Residuals for visualizing the forecast on the training data. Here you can change the hyperparameters for the model and fit a new model.

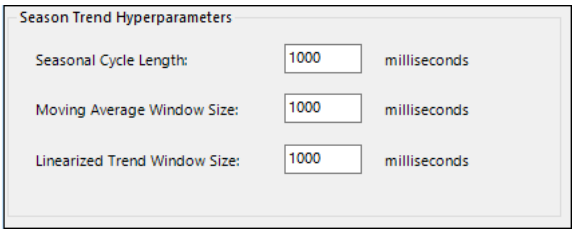
The model hyperparameters are:

- Seasonal cycle length. This is the number of time steps in one seasonal cycle (e.g., 4 for quarterly data, etc.) The units depend on whether you are working on quarterly data, monthly data, etc. The smallest supported time unit is millisecond.
- Moving average window size. The units are the same as for the seasonal cycle length. Enter any integer between 2 and a quarter of the number of time steps in the input dataset.
- Linearized trend window size. For example, enter 12 if you want to go back 12 months to establish the trend. Enter any integer between 1 and the length of the moving average trend window.

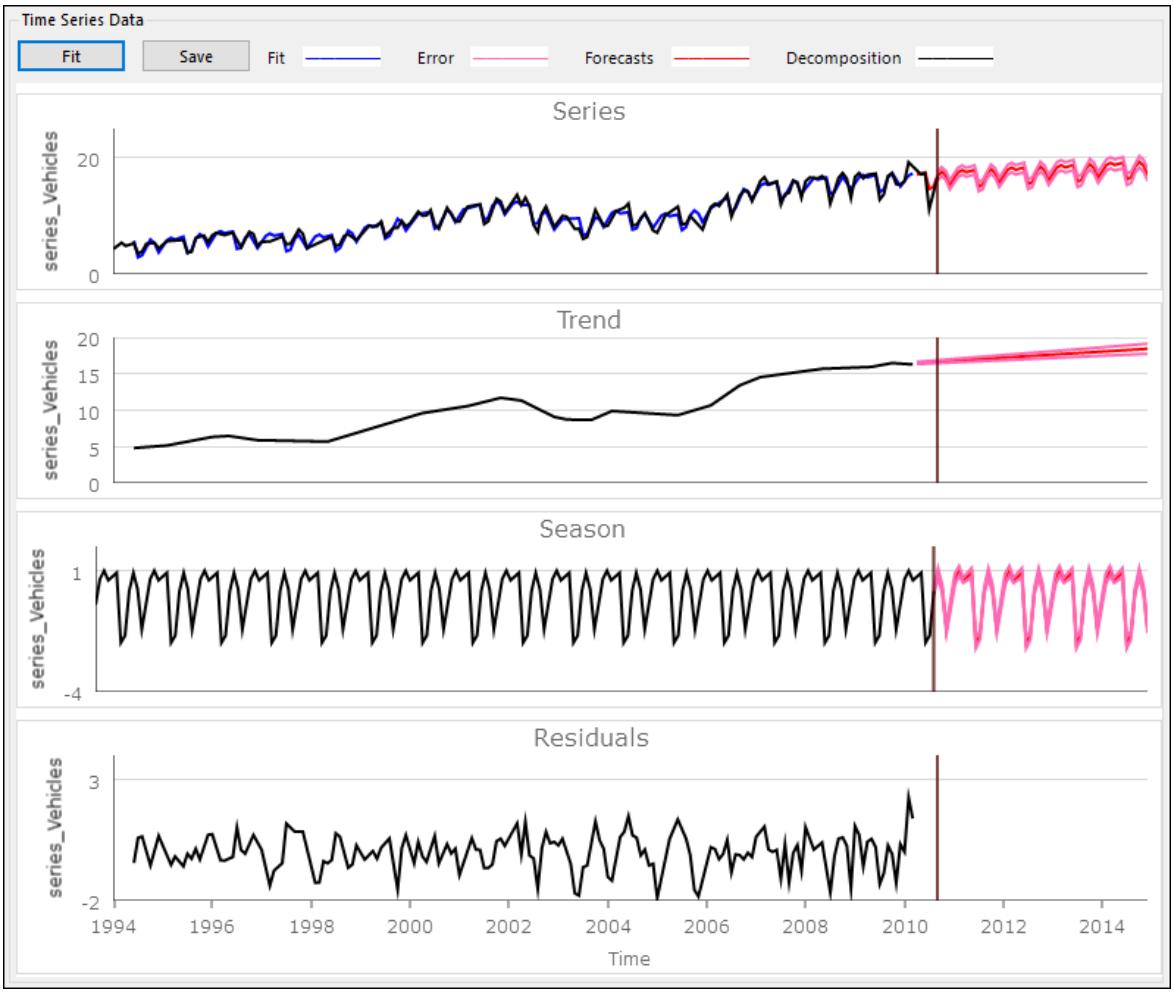
Examples for diferent time units:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.





To change the model, change the hyperparameter values, click **Fit**, then click **Save**. The model will be saved and placed in the **Evaluation** tab for model comparison.



Evaluate

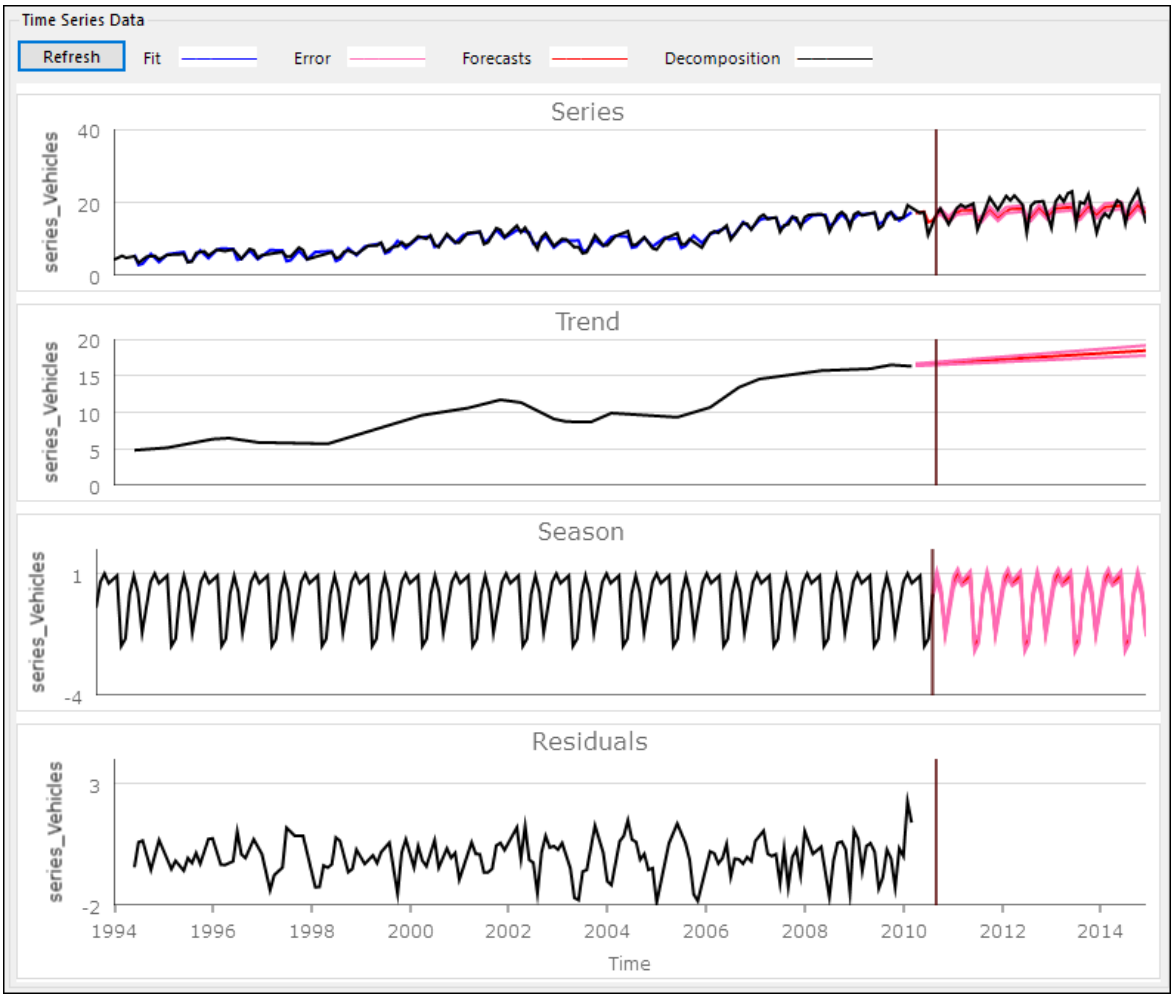
The **Evaluate** tab allows you to evaluate the fit using the values of Mean Absolute Percentage Error (MAPE) and Mean Squared Error (MSE) from the table.

Visually evaluate the fit. The graph shows the forecast on the out-of-time validation sample. The series variable is plotted throughout the entire time range.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Select the desired model and click **Refresh**.

Model ID	Forecast Model	MAPE	MSE	Moving Average Window Size	Linearized Trend Window Size	Seasonal Cycle Length
1	◉	12.41	6.41	12	12	12
2	○	11.95	5.92	24	24	24
3	○	13.1	7.79	6	12	6



Forecast

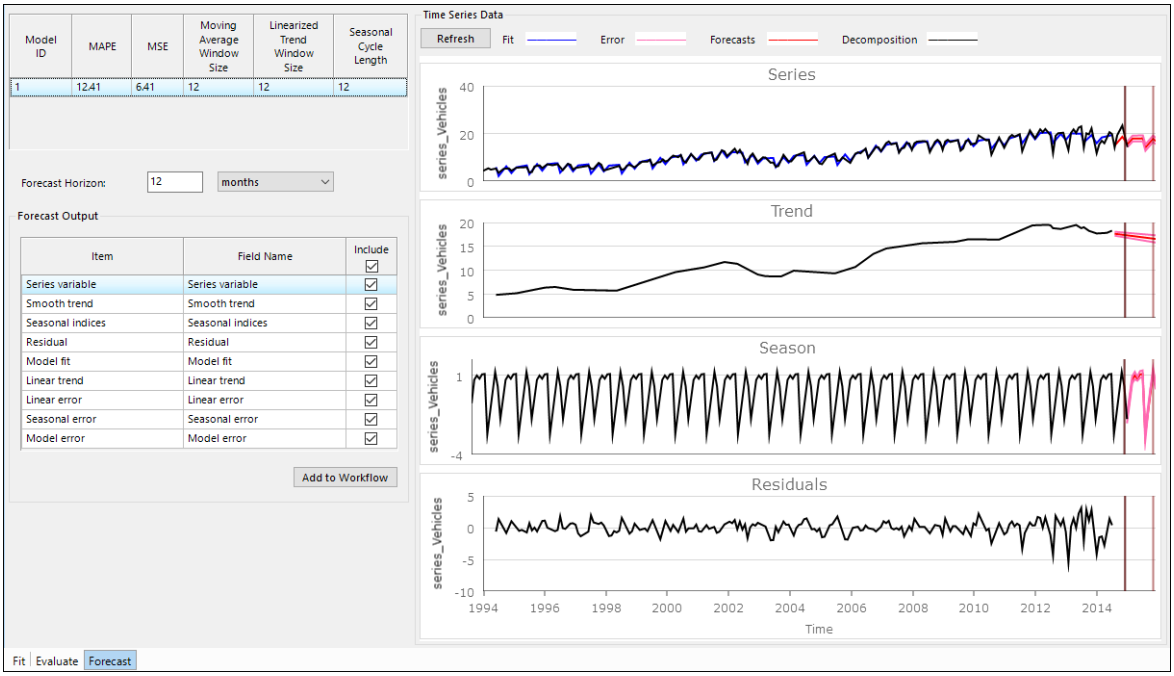
The **Forecast** tab shows the series, trend, season and residuals plotted on the full data for the model selected in the **Evaluate** tab. To see the forecast for a different model, go back to the Evaluate tab and select the desired model using the radio buttons in the frame on the left.

Enter the **forecast horizon** to forecast on the full data and select the units.

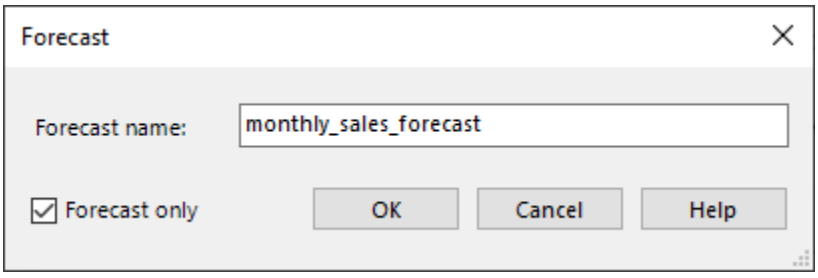
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

In the **Forecast Output** frame on the left, select the type of data to be included in a new time series dataset node on the workflow. The available types are: *series variable, smooth trend, seasonal indices, residual, model fit, linear trend, linear error, seasonal error,* and *model error*.

Click **Add to Workflow** to create a new dataset on the workflow that contains the selected types of data.

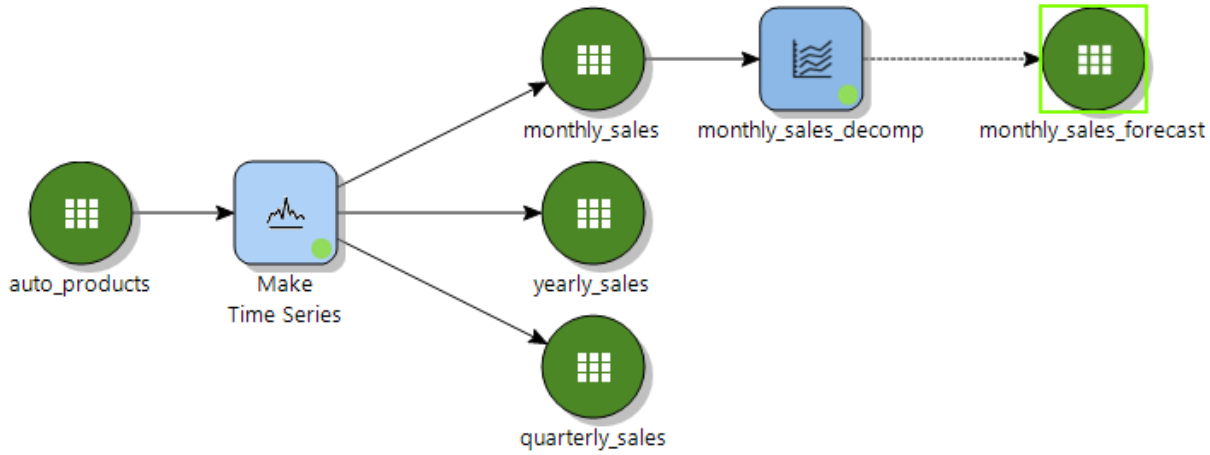


Upon clicking **Add to Workflow**, specify the name for the new time series dataset with your forecasted data.



Go back to the workflow. The output forecast dataset is connected to the Time Series Decomposition node with a dashed line. Double-click the resulting dataset node to view the forecasted data. The forecast dataset is a *time series dataset* with the Time Series and Autocorrelation tabs that satisfies the conditions indicated in the Make Time Series topic.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also:

- Time Series Decomposition
- Make Time Series

Keras Forecasting

Overview

The **Keras Forecasting** node in the **Model** palette of Knowledge Studio performs time-series forecasting with ***Keras*** - a deep learning API written in Python, running on top of the machine learning platform TensorFlow. For details, see Keras documentation at https://keras.io/. Keras copyright and license information is available at https://github.com/keras-team/keras.

The **Keras Forecasting** node wizard provides a convenient interface to specify the parameters and define the neural network configuration for Keras time-series forecasting. The wizard automatically translates the inputs into Python code, which can be previewed before running the node. The input Altair datasets are converted from the native KDD format to Keras-readable format in the process.

Building a Keras Forecasting Model

The input dataset should contain at least one Numeric, Integer, Date, or Timestamp field with no missing values to serve as time index. It should also contain at least one field with the series data. If necessary, use the Make Time Series node to aggregate the data to the desired period level to prepare it for Keras Forecasting.

In the examples below, data aggregated at the monthly level is used as input to the Keras Forecasting model.

Use one of the following ways to add a Keras Forecasting node to your workflow:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

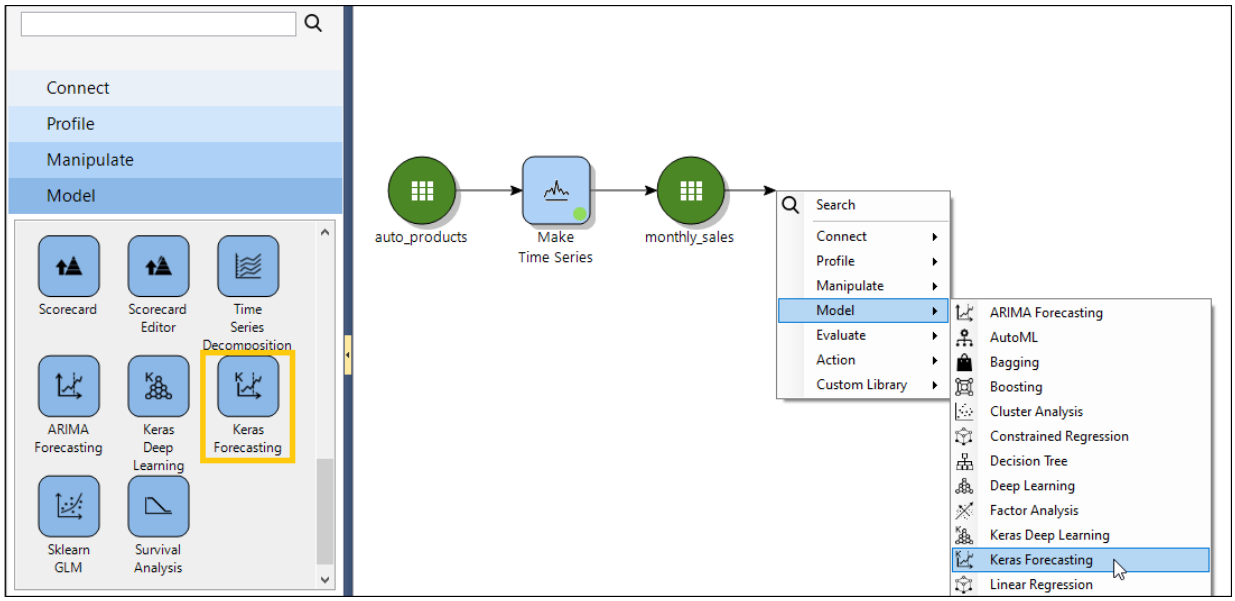
- Drag a connection arrow from the input dataset to any area on the Workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Model --> Keras Forecasting** as in the example screenshot below. A new **Keras Forecasting** node will be created, and the wizard will open.

OR

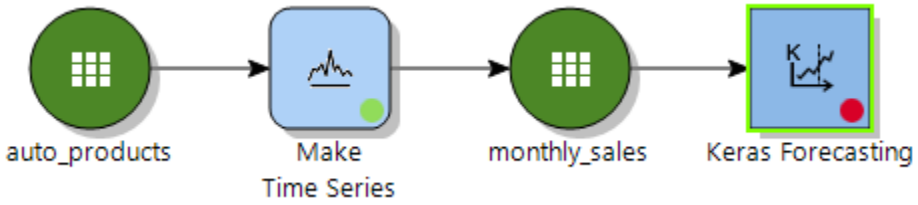
- Right-click on the Workflow canvas and select **Model --> Keras Forecasting** from the context menu. Connect the input dataset node to the Keras Forecasting node.

OR

- Open the **Model** palette and drag the **Keras Forecasting** node onto the workflow canvas. Connect the input dataset node to the Keras Forecasting node.



Double-click the Keras Forecasting node or right-click on it and select **Modify** to open the wizard.



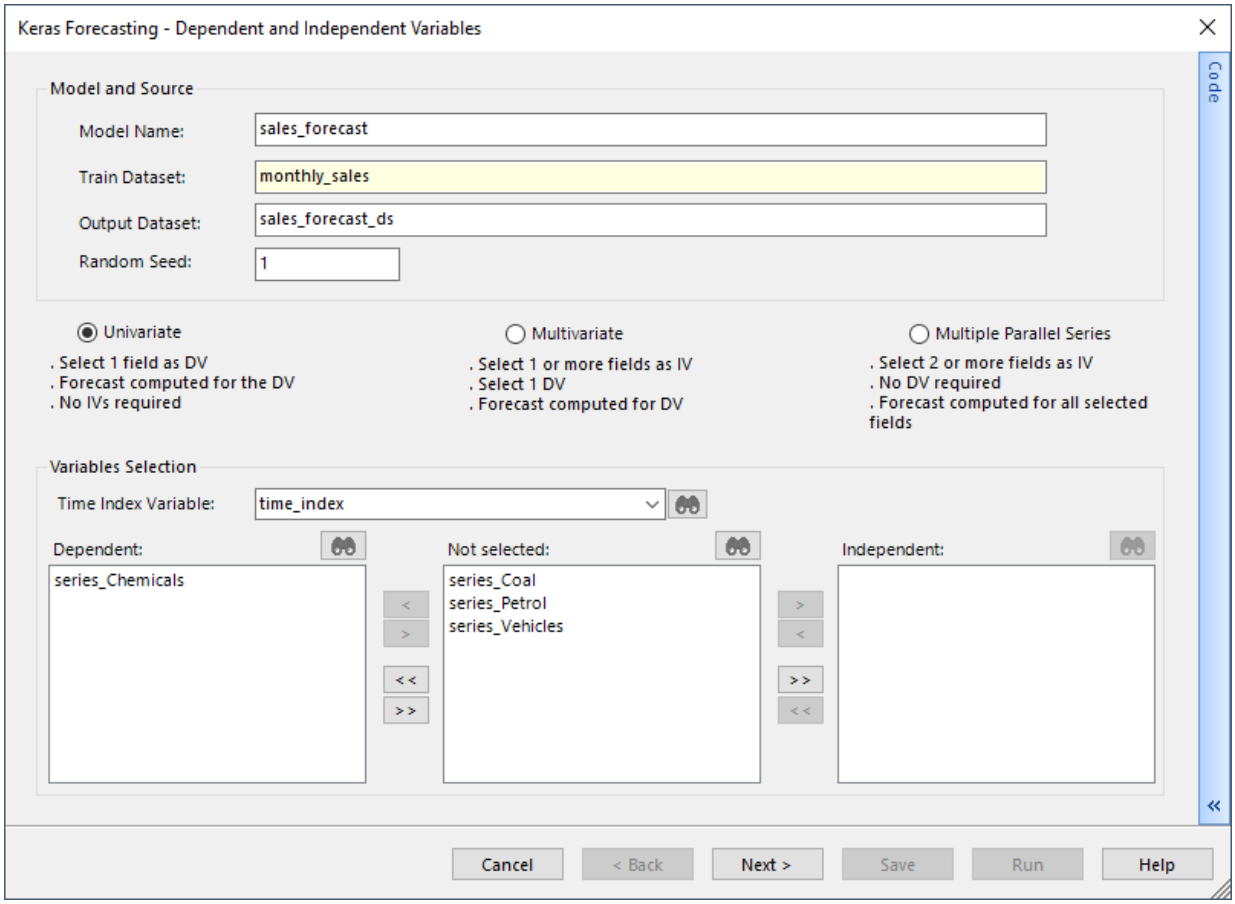
In the first page of the wizard:

- Specify the model name
- Specify the name for the output dataset
- Change the Random Seed value if required for repeatability.
- Select the forecast type:
  - *Univariate*: Only one dependent variable can be selected to be forecasted for one or multiple steps into the future.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- *Multivariate*: One or more independent variables (IV) and one dependent variable (DV) can be selected to forecast the DV for one or multiple steps into the future.
- *Multiple Parallel Series*: One or more independent variables can be forecasted for one or multiple steps into the future. No dependent variable required. (Any variables placed in the *Dependent* list will be ignored.)

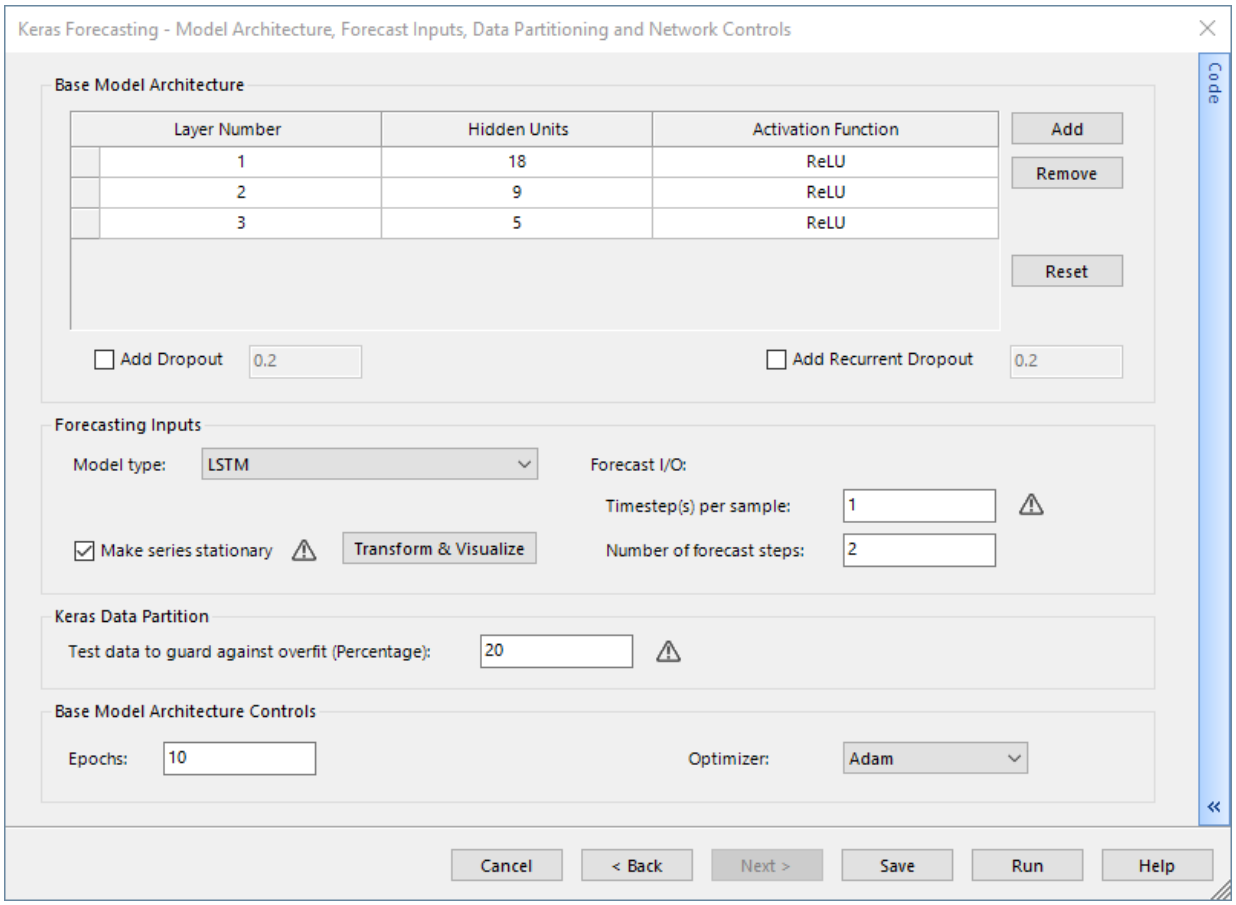
- Select the *Time Index Variable.* This must be a Date, Timestamp, Number or Integer variable with no missing values.
- Select the dependent variable(s) using the arrow buttons to move them to the *Dependent* list on the left.
- Select the independent variables using the arrow buttons to move them to the *Independent* list on the right.



Click **Next**.

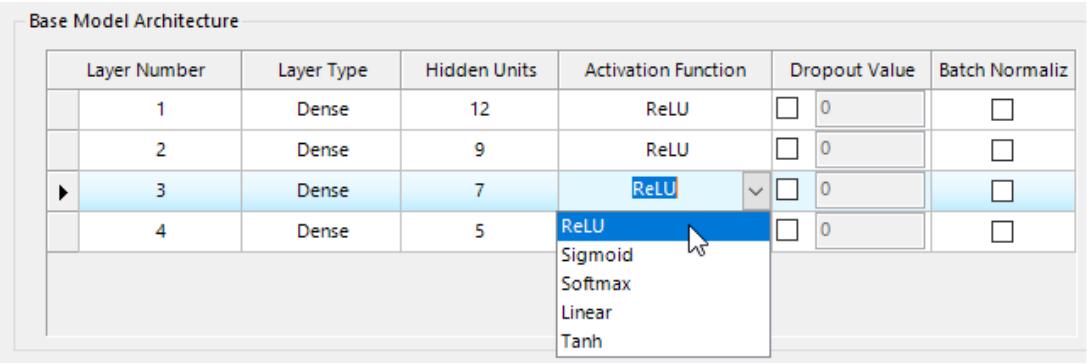
In the next page, specify the model architecture, forecast parameters, data partitioning, and network controls:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- Define the neural network configuration (Base Model Architecture) by adding layers and specifying their properties: the number of **hidden units** and **activation function**. The available activation functions are *ReLU (rectified linear unit)*, *Sigmoid*, *Softmax*, and *Linear*. Select the desired function from the drop-down list for each layer. The **Remove** button removes the selected layer. The **Reset** button removes all layers except the first one.

  Activation function options:

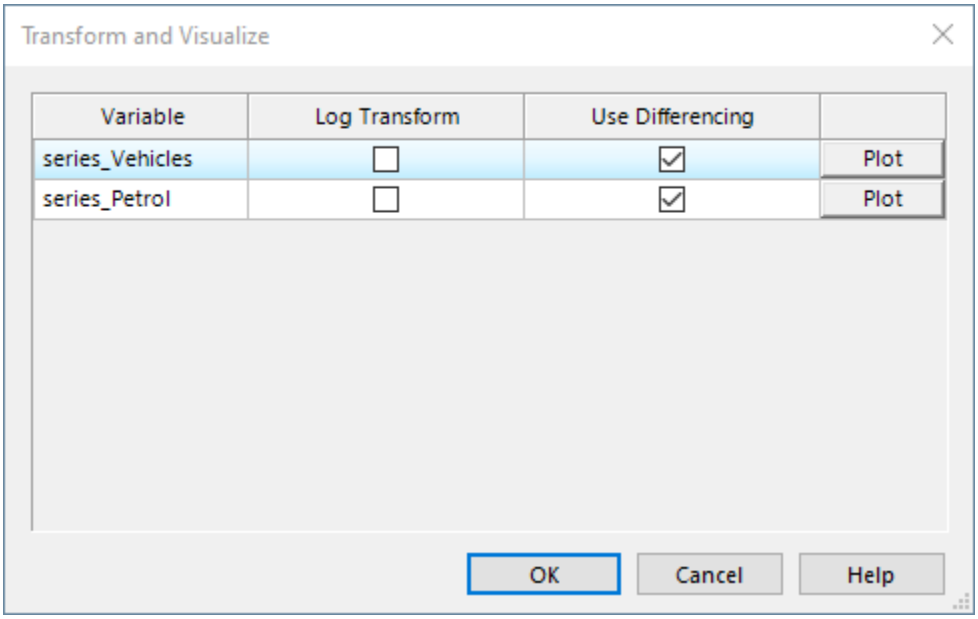


- Select *Add Dropout* and specify the dropout rate - the frequency at which the input units are randomly set to 0 to prevent overfitting. Enter a number between 0 and 1. The default is 0.2.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Select *Add Recurrent Dropout* to improve overfit prevention if necessary. Unlike regular dropout, the recurrent dropout only removes inputs between time steps rather than between layers. Enter a number between 0 and 1. The default is 0.2.

- In the ***Forecasting Inputs*** section:
  - Choose the forecast model type:
    - *LSTM* (Long Short-Term Memory)
    - *Bidirectional LSTM*
    - *GRU* (Gated Recurrent Unit)
  - *Make series stationary*: If you deselect this option, no transformations will be applied. This option uses the augmented Dickey Fuller test to decide if the series must be made stationary.
    - Note: For the *Multivariate* forecast type, no transformations are applied to the independent variables.
  - Click *Transform & Visualize* to define transformations (applicable to the *Univariate* and *Multiple Parallel Series* forecast types).

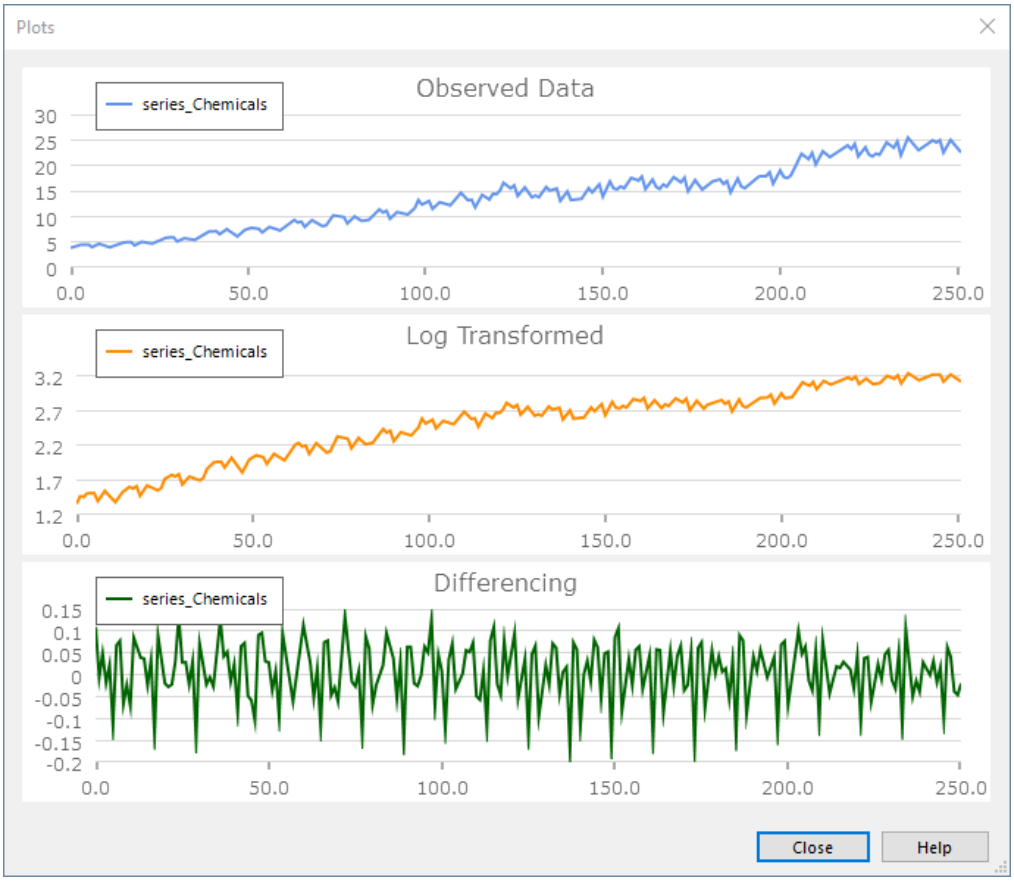


    - The *Differencing* option is selected by default. If this function does not return a plot, you will get a message “Data is already stationary“. Otherwise the observed data and differenced data are plotted one below the other in a pop-up window when you click the **Plot** button. If the *Log Transform* option was selected, the log-transformed plot is displayed as well.
    - Selecting the *Log Transform* option also automatically enables *Differencing*. The log transformation is applied first, then differencing is automatically applied.
    - Click the **Plot** button for the desired series to plot the observed data, log-transformed data, and differenced data one below the other. If the log transformation is not applicable because the data is outside of the range of

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

admissible values, then the message "*Data is not suitable for log transformation*" is displayed. In this case only differencing is applied.

- The output from this step is used as the input for modelling. If a plot is available after differencing, then the differenced data is used as input for modelling, otherwise the log transformed data is used. If you deselect the option "Make series stationary", then the observed data is used for modelling.

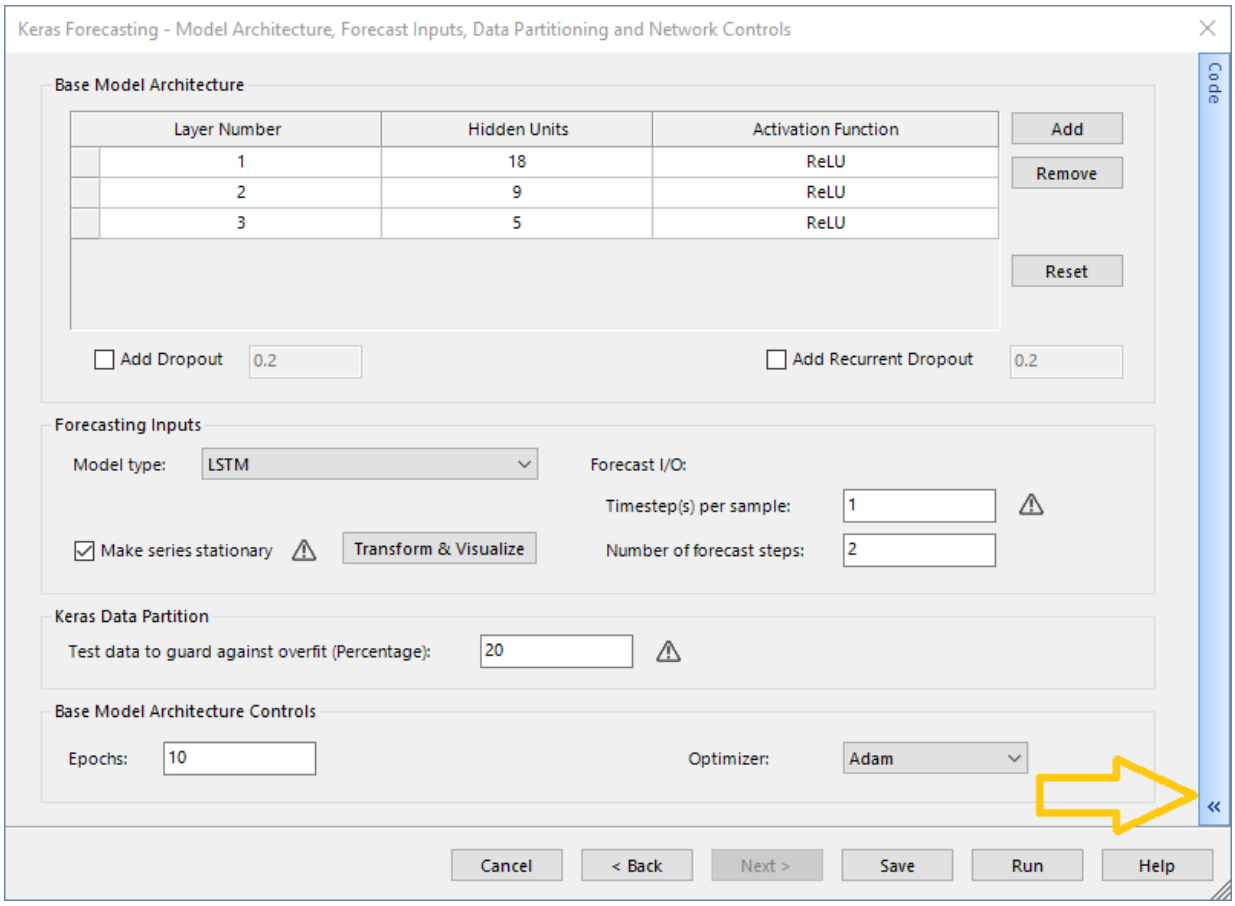


- Forecast I/O options:
  - *Timesteps per sample*: Specify the number of rows (lag values) used per sample. The default is 1.
  - *Number of forecast steps:* Specify the number of time steps produced as the forecast output. The default is 2.

- In the ***Keras Data Partition*** section, specify the percentage of the input dataset to be allocated for testing to guard against overfit. Note that this decreases the number of records for training. The default is 20% allocated for testing.
- In the ***Base Model Architecture Controls*** section, specify:
  - Number of epochs (training iterations); the default is 10
  - Optimizer (the available optimizers are *Adam* and *RMSprop;* the default is *Adam*).

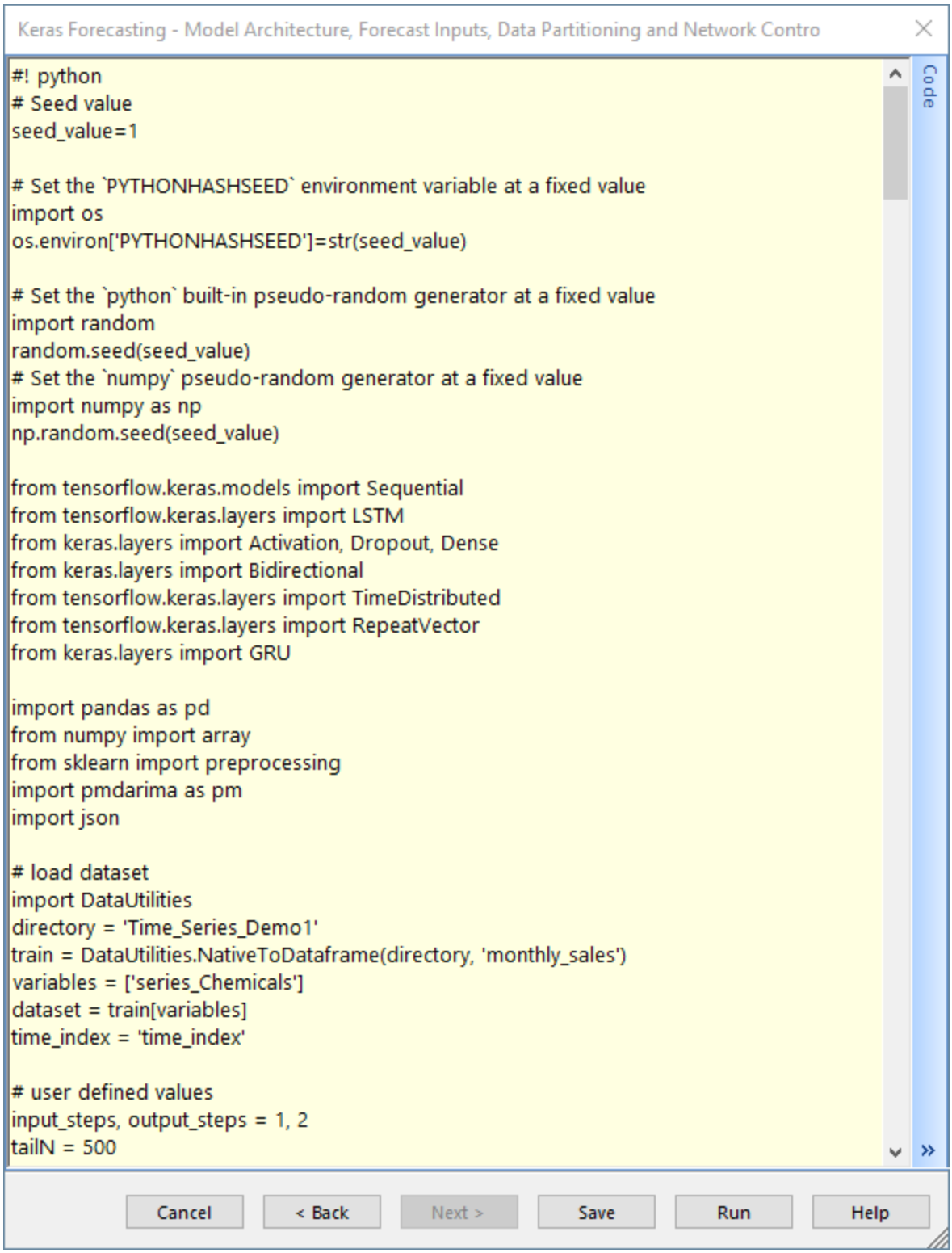
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog.



This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

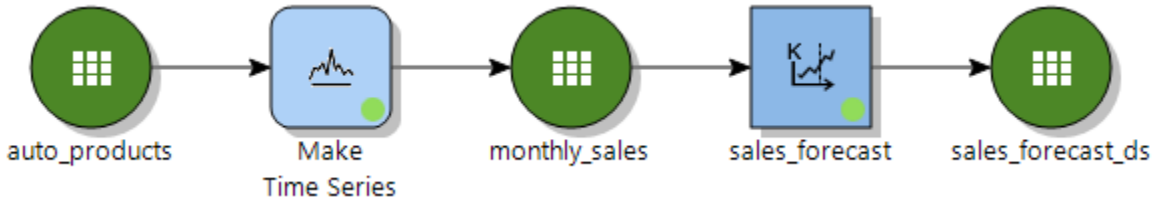


Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. The input dataset is converted to a Pandas dataframe and placed in a TensorFlow session. The defined Keras Forecasting model runs in a built-in Python environment, loading all required

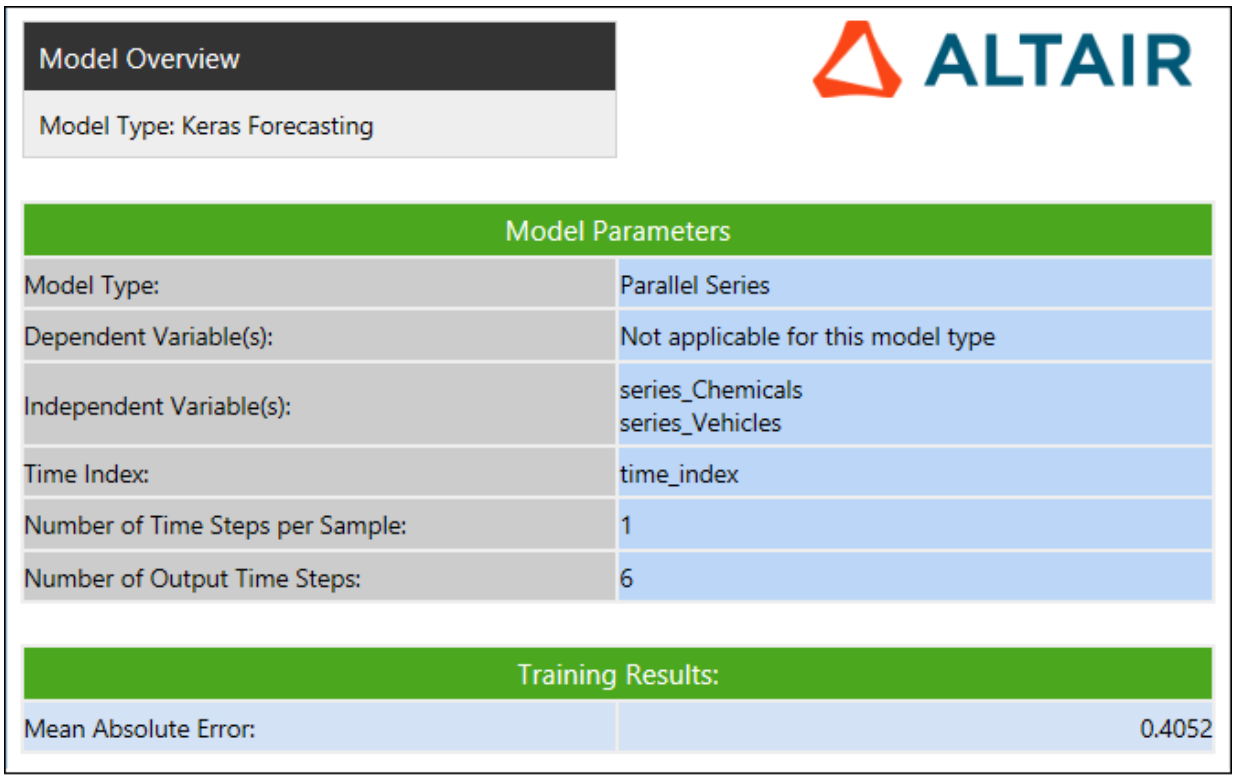
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

packages and their dependencies. The output dataset is converted from Pandas to the Altair KDD format.

After running the node, double-click the forecast node and the output dataset node to view the results.



Double-click on the Keras Forecasting node to view the summary report, performance plots, and forecast plots. The following examples are for Multiple Parallel Series.



Model Overview

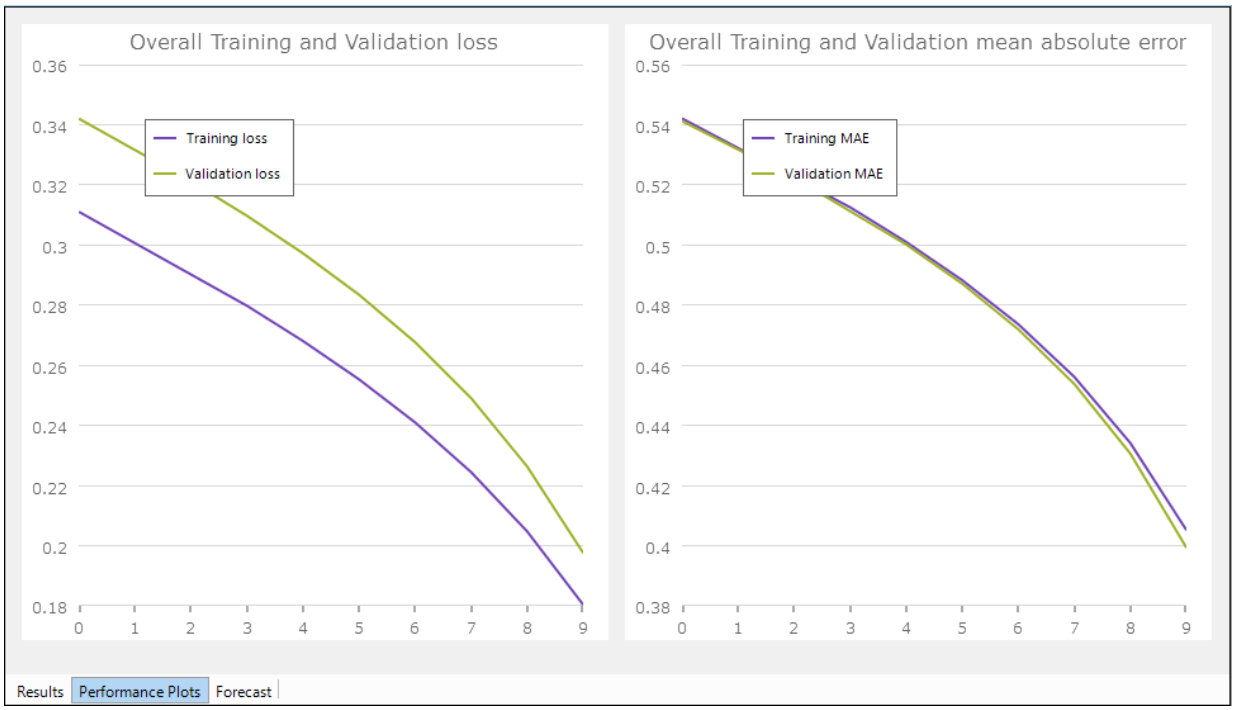
Model Type: Keras Forecasting

Model Parameters	
Model Type:	Parallel Series
Dependent Variable(s):	Not applicable for this model type
Independent Variable(s):	series_Chemicals series_Vehicles
Time Index:	time_index
Number of Time Steps per Sample:	1
Number of Output Time Steps:	6

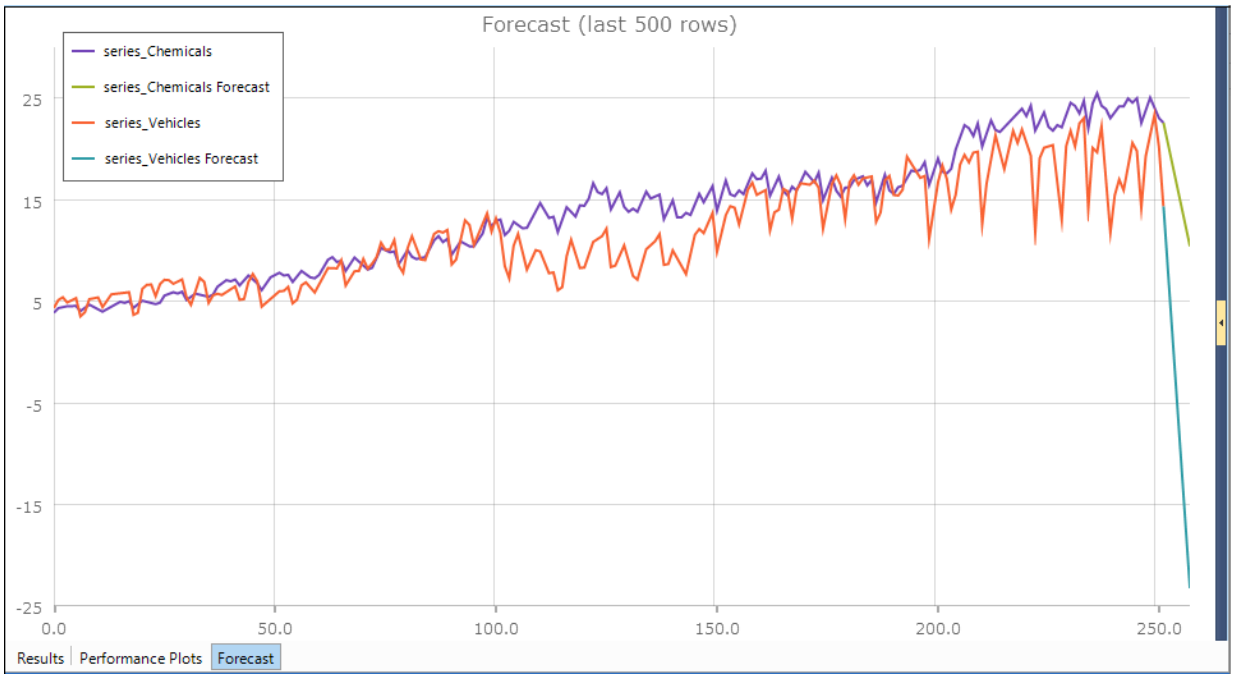
Training Results:	
Mean Absolute Error:	0.4052

Performance plots provide a way to validate the time series model. Click the **Performance Plots** tab to view the training and validation loss and accuracy plots. The second plot shows the mean absolute error (MAE) of the model on the training and validation sets.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

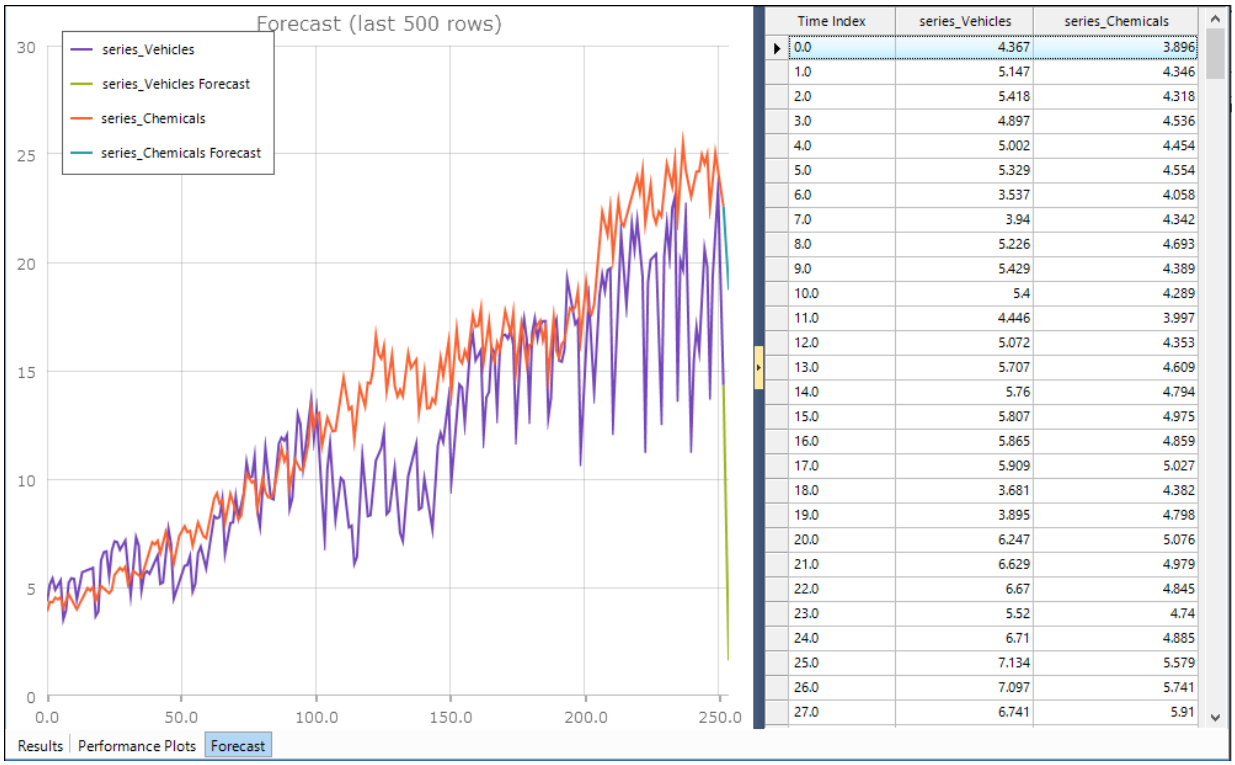


Click the **Forecast** tab. The plot shows the input data points extended with forecast data for the number of forecast steps specified in the wizard. The legend at the top left shows the color codes for the input data and forecast data for each series.



To view the data table that the forecast plot is based on, click on the small arrow in the middle of the blue bar on the right-hand side. The table will open in a frame on the right. Click the arrow again to hide the frame.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Time Index	series_Vehicles	series_Chemicals
0.0	4.367	3.896
1.0	5.147	4.346
2.0	5.418	4.318
3.0	4.897	4.536
4.0	5.002	4.454
5.0	5.329	4.554
6.0	3.537	4.058
7.0	3.94	4.342
8.0	5.226	4.693
9.0	5.429	4.389
10.0	5.4	4.289
11.0	4.446	3.997
12.0	5.072	4.353
13.0	5.707	4.609
14.0	5.76	4.794
15.0	5.807	4.975
16.0	5.865	4.859
17.0	5.909	5.027
18.0	3.681	4.382
19.0	3.895	4.798
20.0	6.247	5.076
21.0	6.629	4.979
22.0	6.67	4.845
23.0	5.52	4.74
24.0	6.71	4.885
25.0	7.134	5.579
26.0	7.097	5.741
27.0	6.741	5.91

See also: Keras Deep Learning, Keras Scoring.

ARIMA Forecasting

Overview

Autoregressive Integrated Moving Average (ARIMA) is a time series modeling technique used to explore and forecast time series data points.

The **ARIMA Forecasting** node in the **Model** palette of Knowledge Studio performs ARIMA forecasting with **`pmdarima`** - a Python statistical library for time series forecasting. For details, see the pmdarima documentation at the Python Package Index repository: https://pypi.org/project/pmdarima/. The node wizard provides a convenient graphical interface to specify the options and parameters for `pmdarima`. The wizard automatically translates the inputs into Python code, which can be previewed before running the node. The input dataset is converted from the Altair KDD format to a Python dataframe in the process, then back to produce the output dataset with forecasted series.

Building an ARIMA Forecasting Model

The input dataset should contain at least one Numeric, Integer, Date, or Timestamp field with no missing values to serve as time index. It should also contain at least one field with the series data. If necessary, use the Make Time Series node to aggregate the data to the desired period level to prepare it for ARIMA Forecasting.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

In the examples below, data aggregated at the monthly level is used as input to the ARIMA Forecasting model.

Use one of the following ways to add an ARIMA Forecasting node to your workflow:

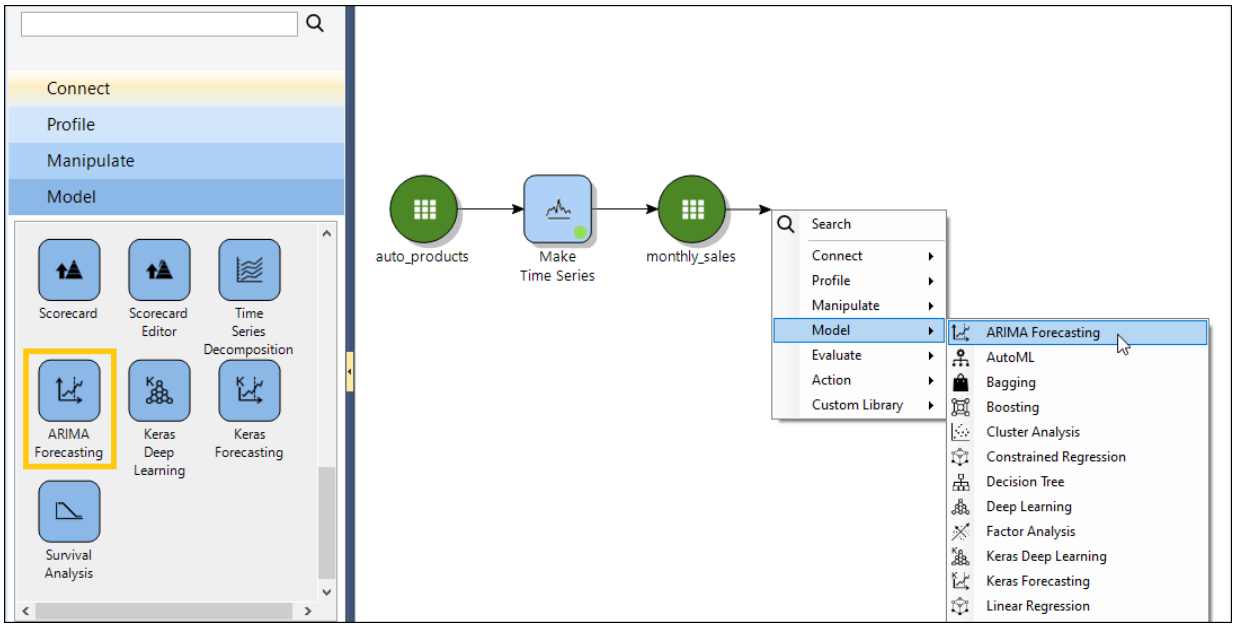
- Drag a connection arrow from the input dataset to any area on the Workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Model --> ARIMA Forecasting** as shown below. A new **ARIMA Forecasting** node will be created, and the wizard will open.

  OR

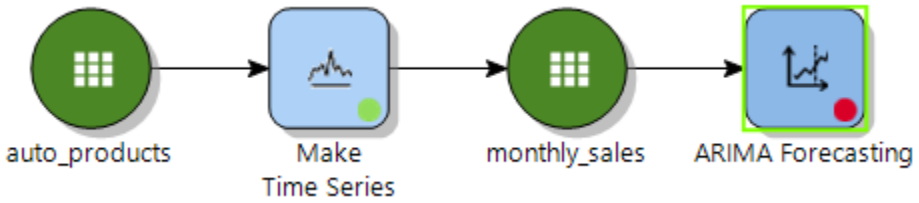
- Right-click on the Workflow canvas and select **Model --> ARIMA Forecasting** from the context menu. Connect the input dataset node to the ARIMA Forecasting node.

  OR

- Open the **Model** palette and drag the **ARIMA Forecasting** node onto the workflow canvas. Connect the input dataset node to the ARIMA Forecasting node.



Double-click the ARIMA Forecasting node or right-click on it and select **Modify** to open the wizard.

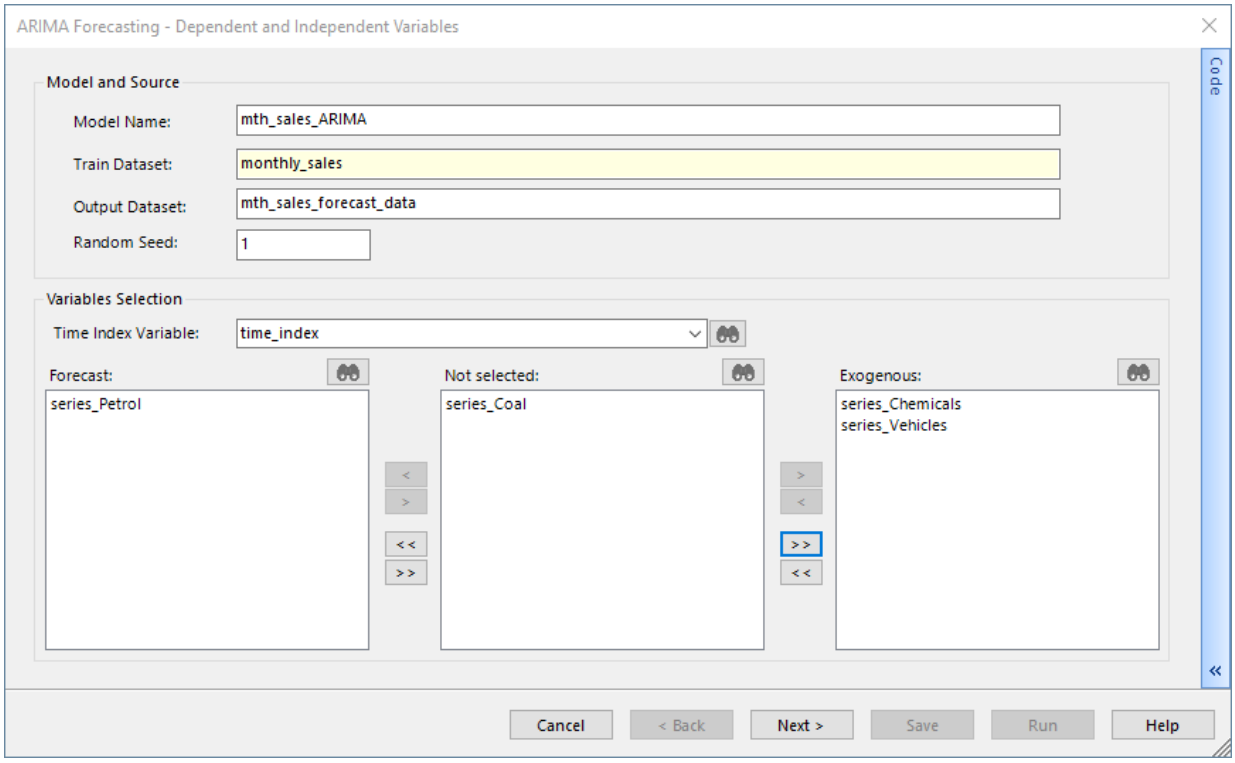


In the first page of the wizard:

- Specify the names for the ARIMA model and the output dataset.
- Change the Random Seed value if required for repeatability when performing multiple runs.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

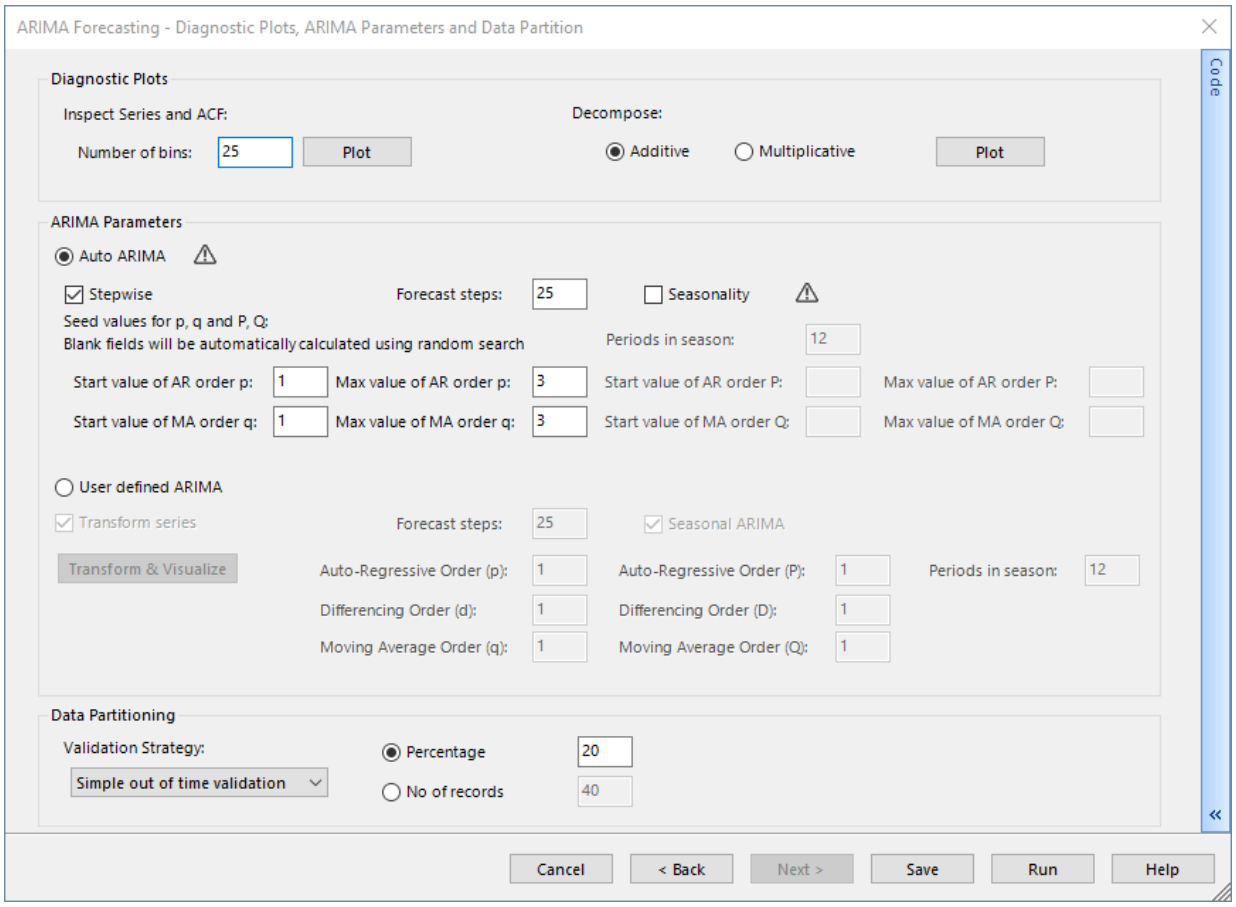
- Select the ***Time Index Variable***. This must be a Date, Timestamp, Number or Integer variable with no missing values.
- Select one or more variables to forecast by moving them to the *Forecast* list on the left using the arrow buttons.
- Select the exogenous variables by moving them to the *Exogenous* list on the right using the arrow buttons.



Click **Next**.

In the next page, specify the model parameters, options for diagnostic plots, and define data partitioning.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



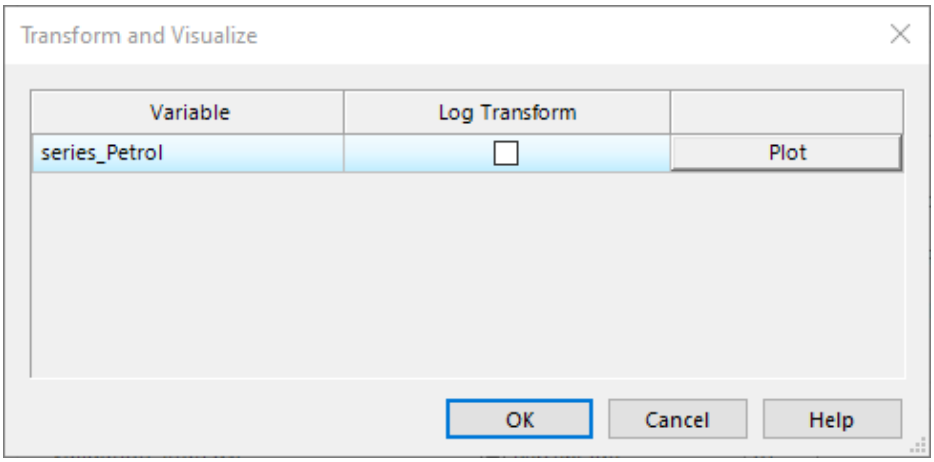
- For the diagnostic plots:
  - specify the number of bins for the series plot and the autocorrelation function (ACF) plot of the series. The default is 25 bins. Click the **Plot** button to preview the plot
  - choose the type of error for time series decomposition: *Additive* or *Multiplicative*. The default type is *Additive*. Click the **Plot** button to preview the plot.
- For the ARIMA model parameters:
  - Select the ***Auto ARIMA*** option for automatic handling of differencing. Select the ***User-defined ARIMA*** option to define differencing parameters manually.
  - If the ***Auto ARIMA*** option is selected:
    - The *Stepwise* mode of the auto_arima function is selected by default. Deselect it to use the Parallelized mode. Note that the Parallelized mode is slower.
    - Specify the number of *Forecast steps* (periods ahead to forecast). The default is 25.
    - Select the *Seasonality* option to specify seasonal model parameters if required. Note that using seasonality may significantly increase the run time. This option is not selected by default.
    - Specify the initial and maximum values for the search of the ARIMA terms `p` and `q` - the orders of the auto-regressive (AR) model and the moving average (MA) model, respectively. If not specified, they will be automatically

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

calculated using random search. The default is 1 for the start values and 3 for the max values.

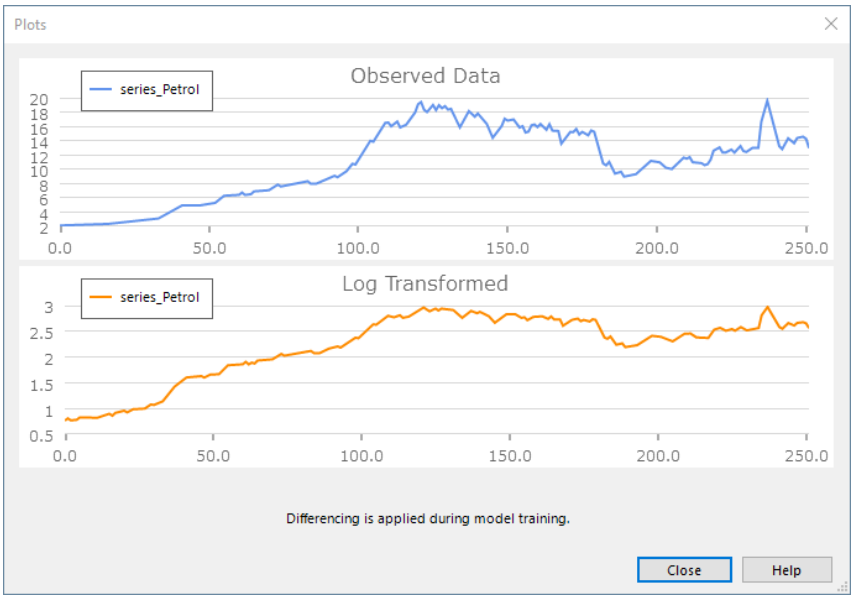
- If the *Seasonality* option is selected, define the seasonal component of the model:
  - Specify the *number of periods per season*. The default is 12.
  - Specify the initial and maximum values for the search of the ARIMA terms P and Q - the orders of the *seasonal components* of the auto-regressive (AR) model and the moving average (MA) model, respectively. If not specified, they will be automatically calculated using random search.

- If the ***User-defined ARIMA*** option is selected:
  - Select the *Transform series* option to apply log transform to the series to enforce stationarity. This option is enabled by default.
  - Click **Transform & Visualize** to define transformations and view the plots for the observed and transformed data.



  - Select the Log Transform option if desired and click the **Plot** button to plot the observed data and log-transformed data. The plots are shown in a pop-up window. If the log transformation is not applicable because the data is outside of the range of admissible values, then the message "*Data is not suitable for log transformation*" is displayed.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

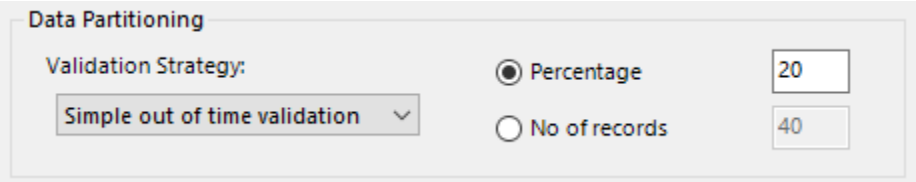


- The output from this step is used as the input for modelling. If you deselect the option "Transform series", then the observed data is used for modelling.

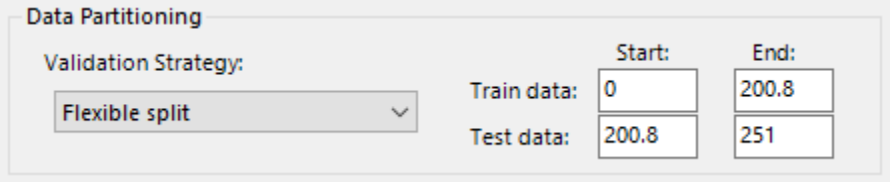
- Specify the number of *Forecast steps* (periods ahead to forecast). The default is 25.
- Specify the values for the ARIMA terms `p`, `d`, and `q` - the orders of the auto-regressive (AR) model, differencing, and the moving average (MA) model, respectively.
- Select the *Seasonal ARIMA* option to specify seasonal model parameters if required.
- If the ***Seasonal ARIMA*** option is selected, define the seasonal component of the model:
  - Specify the number of periods per season. The default is 12.
  - Specify the ARIMA terms `P`, `D`, and `Q`, where `P` and `Q` are the orders of the *seasonal components* of the auto-regressive (AR) model and the moving average (MA) model, respectively, and `D` is the integration order of the seasonal process.

- Specify the Data Partitioning parameters for model validation:
  - Choose a ***Validation strategy***: *Simple out-of-time validation*, *Simple in-sample validation*, or *Flexible split*.
    - For *Simple out-of-time validation*, specify the size of the partition in percentage of the input dataset (the default is 20%) or the number of records. Note that this decreases the number of records for training.

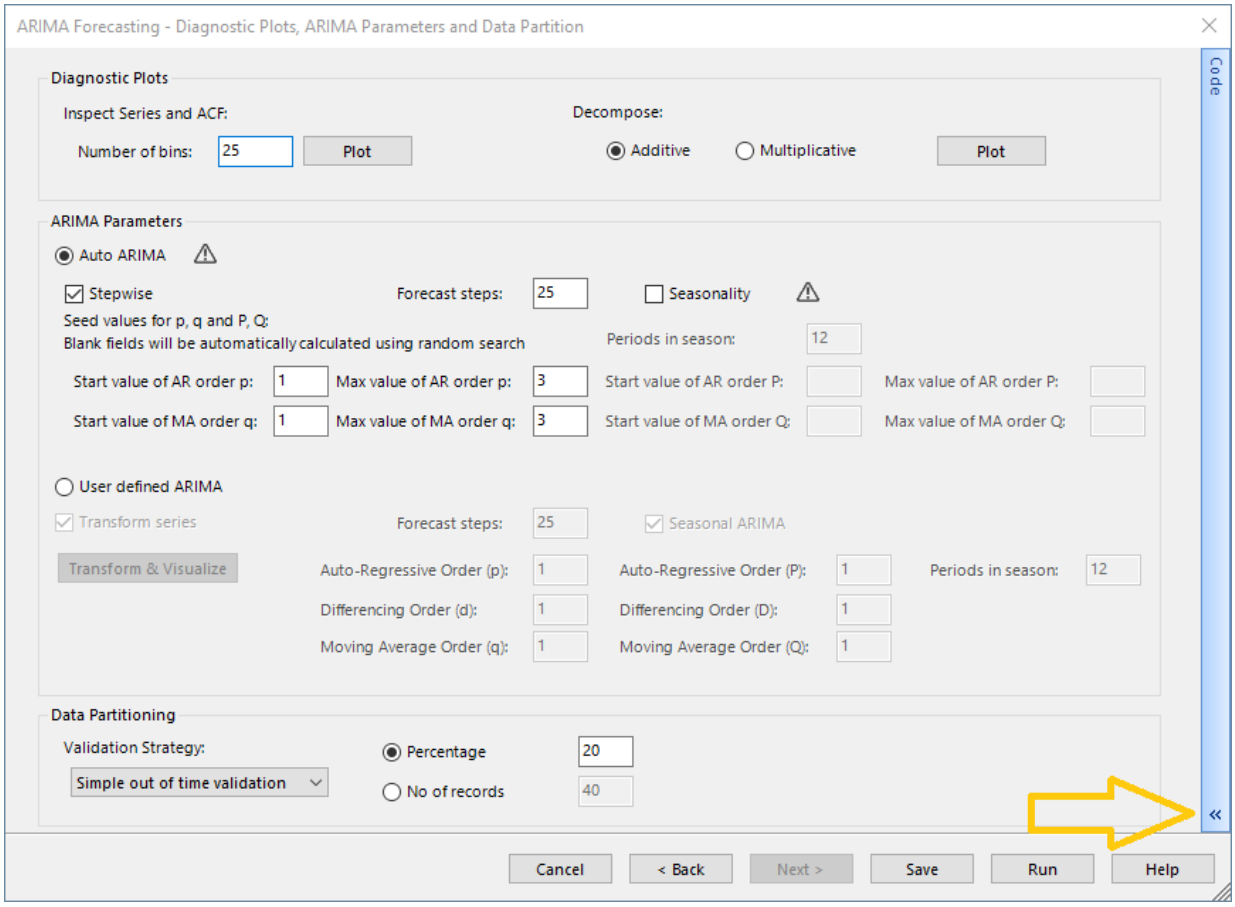
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- For *Flexible split*, specify the start and end points for the Train data and Test data partitions. (These are the data points of the forecasted series.)

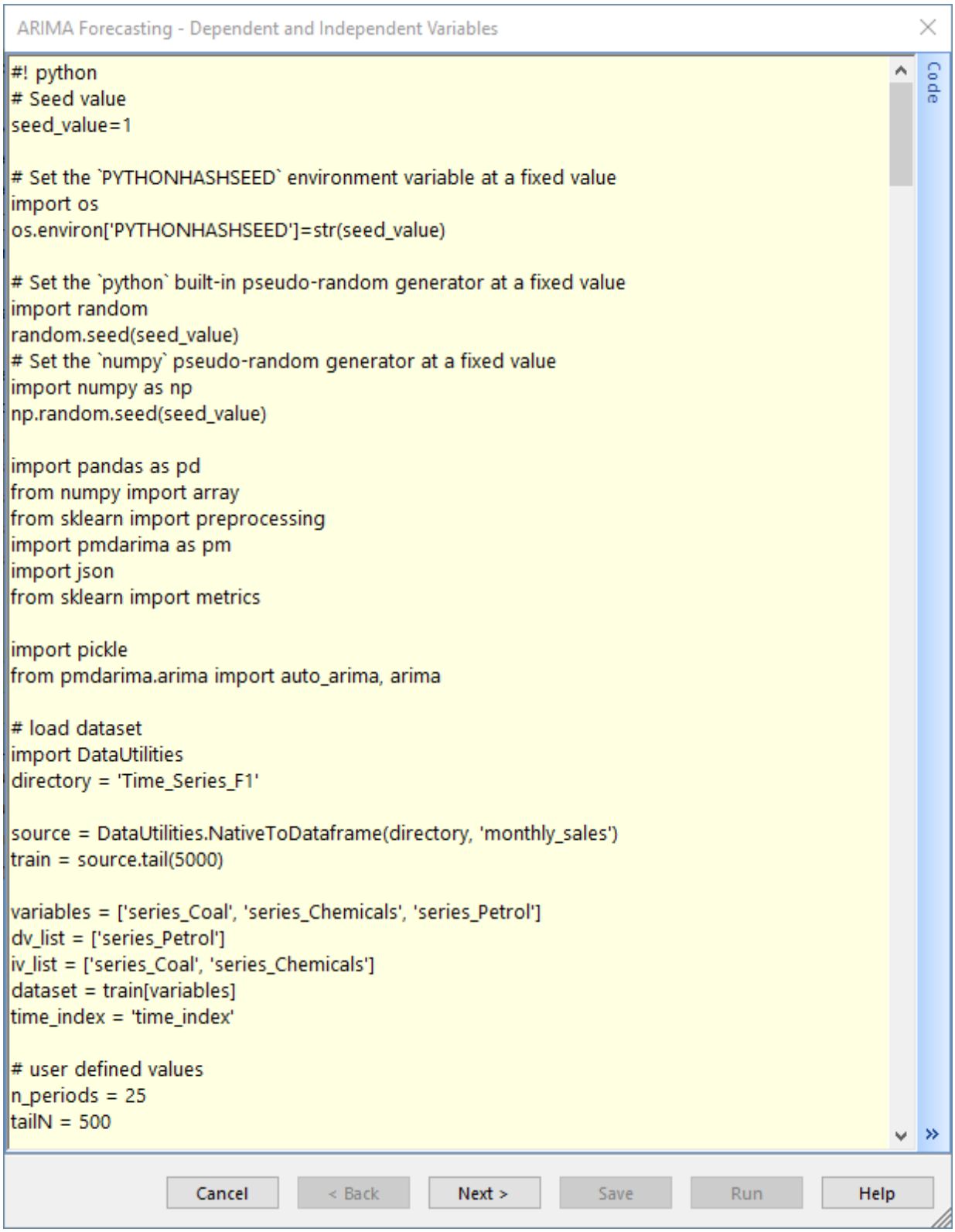


To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog.



This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

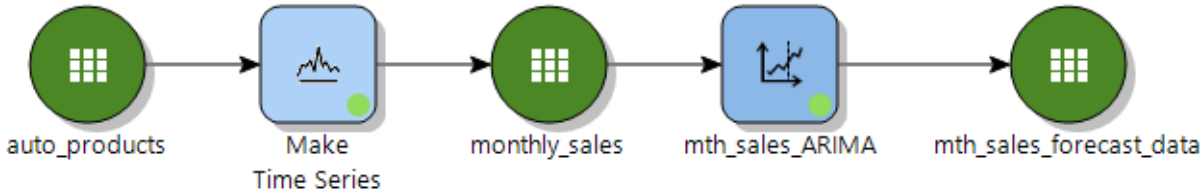


Click **Save** to save the settingsor click Run to save the settings and run the process immediately. The input dataset is converted to a Pandas dataframe. The defined ARIMA Forecasting model runs

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

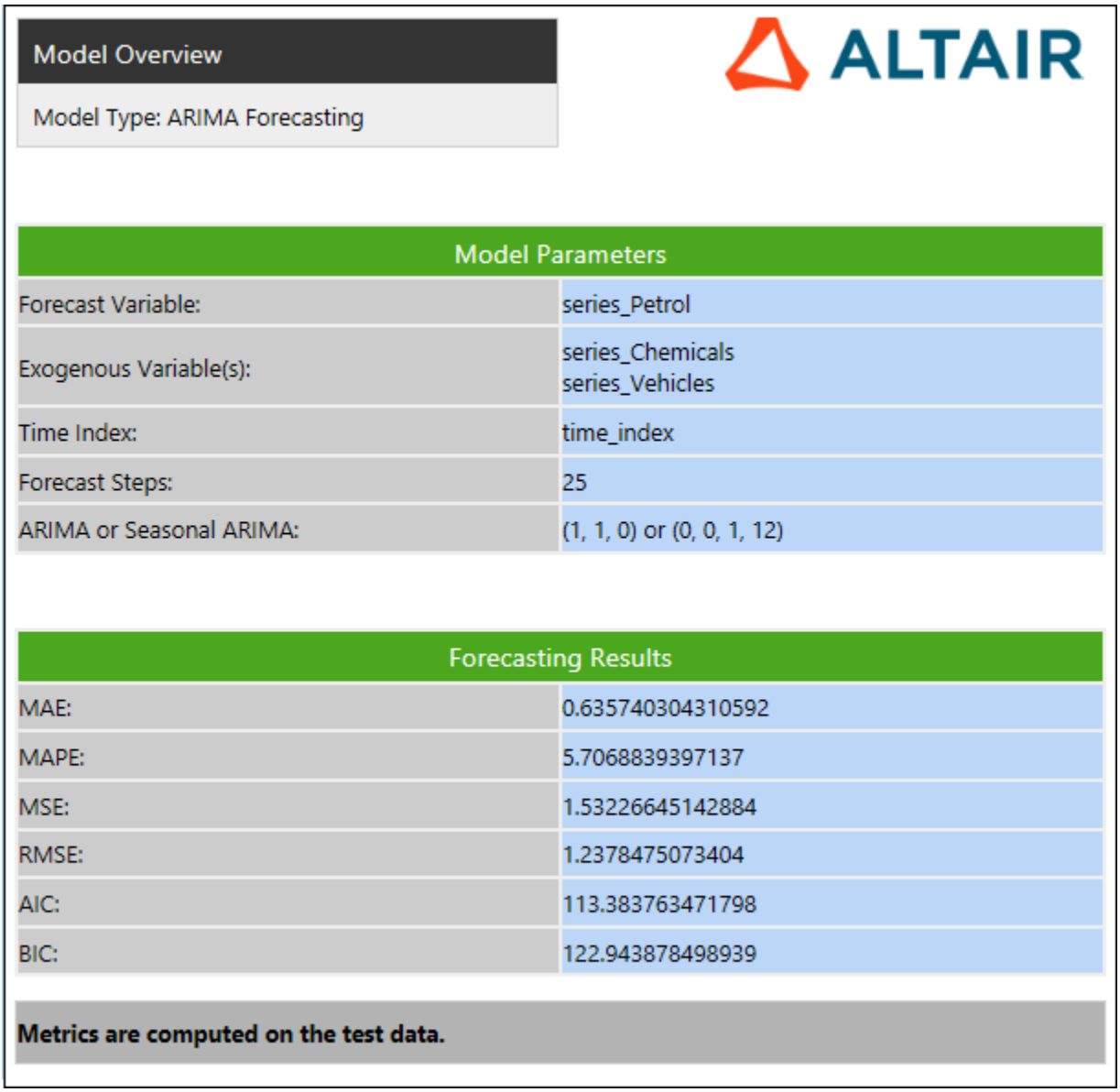
in a Python container session. The output dataset is converted from Pandas to the Altair KDD format.

After running the node, double-click the forecast node and the output dataset node to view the results.



Double-click on the ARIMA Forecasting node to view the summary report, diagnostic plots, and forecast plots.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Model Overview

Model Type: ARIMA Forecasting

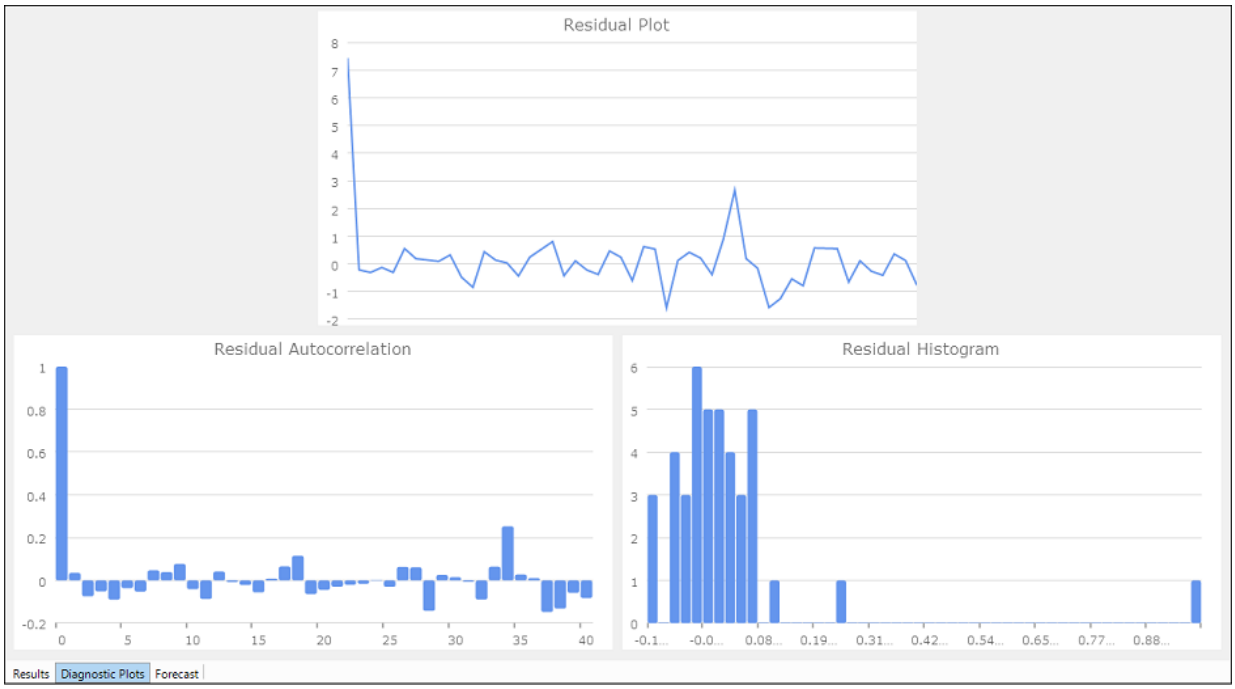
Model Parameters	
Forecast Variable:	series_Petrol
Exogenous Variable(s):	series_Chemicals series_Vehicles
Time Index:	time_index
Forecast Steps:	25
ARIMA or Seasonal ARIMA:	(1, 1, 0) or (0, 0, 1, 12)

Forecasting Results	
MAE:	0.635740304310592
MAPE:	5.7068839397137
MSE:	1.53226645142884
RMSE:	1.2378475073404
AIC:	113.383763471798
BIC:	122.943878498939

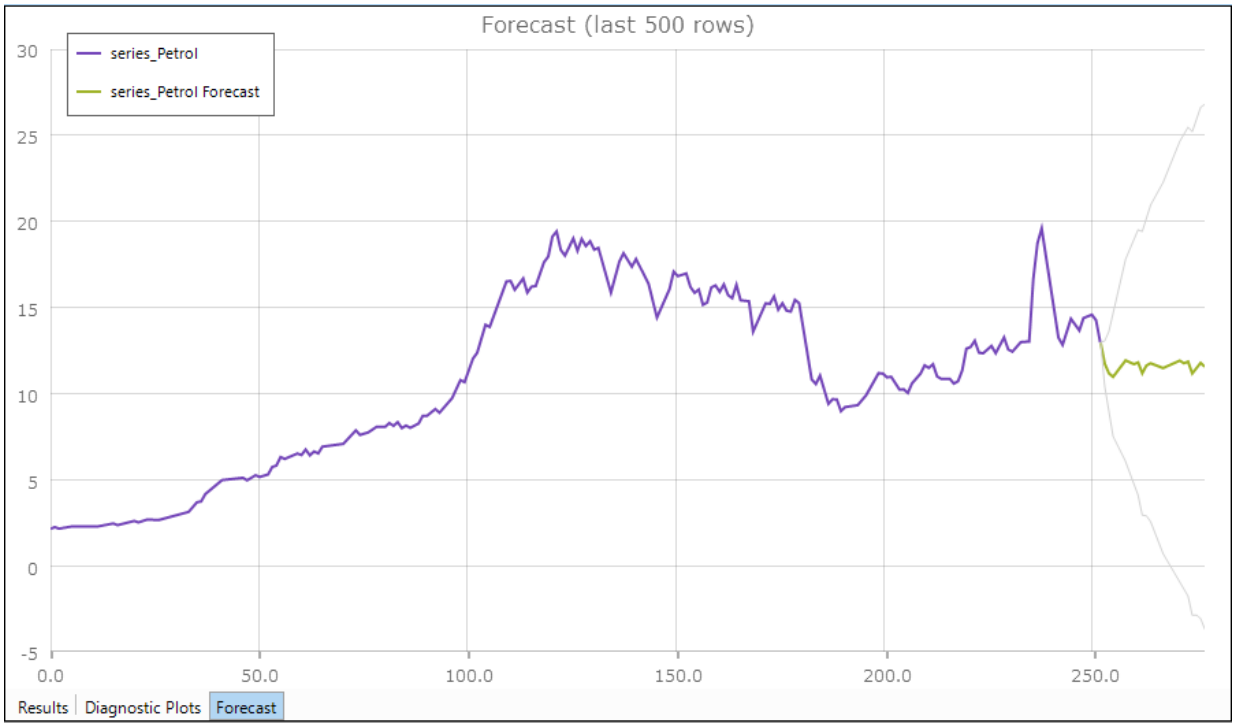
Metrics are computed on the test data.

Diagnostic plots provide a way to validate the time series model. Click the **Diagnostic Plots** tab to view the plots for Residual, Residual Autocorrelation, and Residual Histogram.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

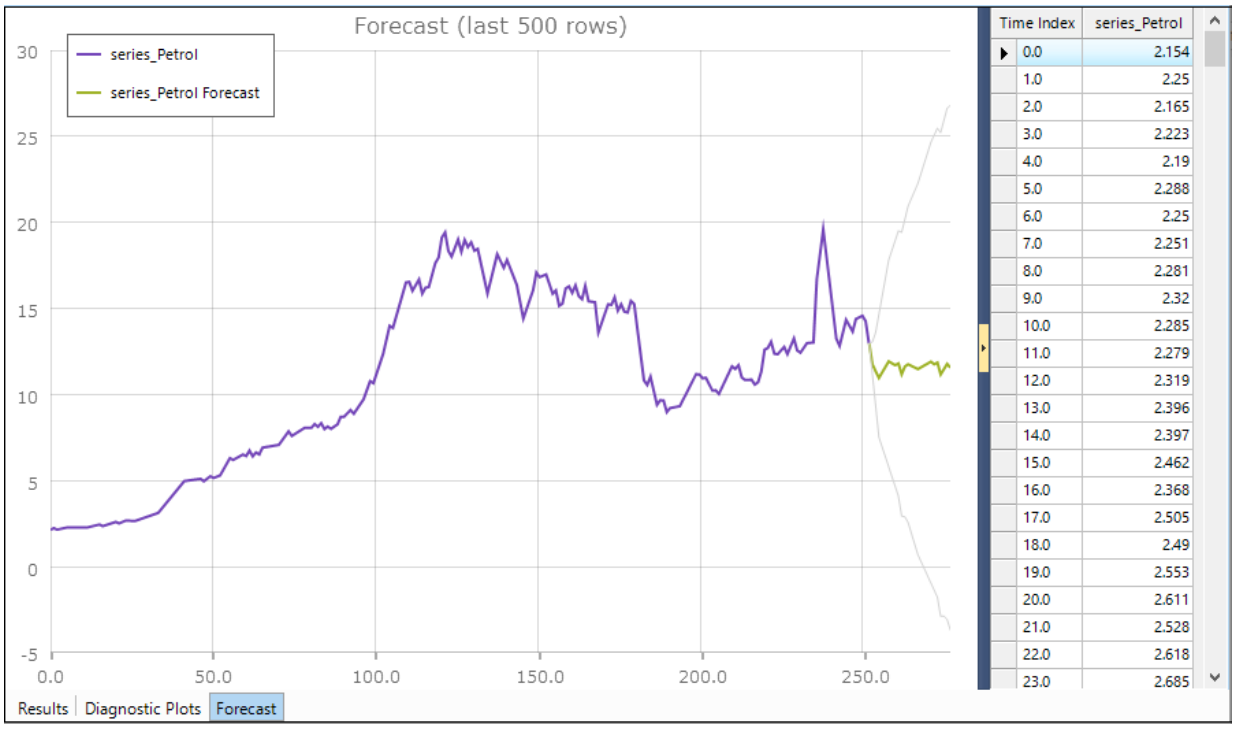


Click the **Forecast** tab. The X axis label shows input data points and the forecasted data. The legend at the top left shows the color codes for the input data and forecasted data for each series.



To view the data table that the forecast plot is based on, click on the small arrow in the middle of the blue bar on the right-hand side. The table will open in a frame on the right. Click the arrow again to hide the frame.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also: ARIMA Scoring, Keras Forecasting.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Model Evaluation

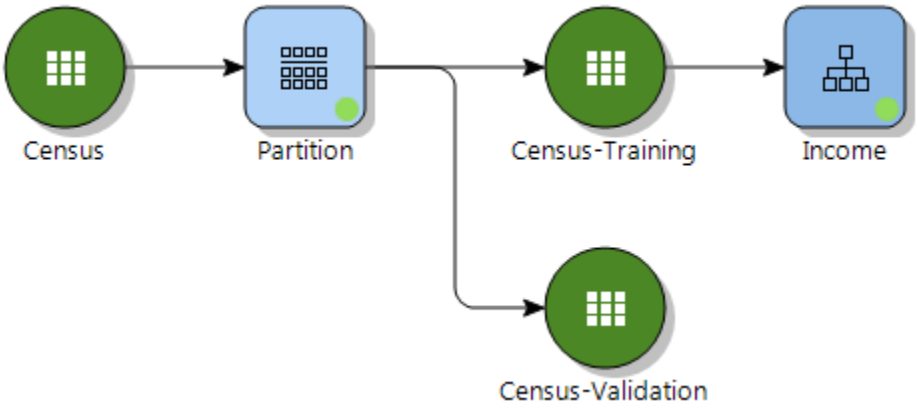
Model Validation

After creating a predictive model, it is reasonable to test the accuracy of its predictions before deploying it into production. This can be accomplished by applying the model to other datasets (different from the one the model was trained on) with known values of the dependent variable. Model predictions are then compared to these known values, and certain statistics are calculated to measure the model accuracy.

This process is called **validation**.

As a standard modeling practice, prior to creating models, the dataset is split into two partitions: **Training** and **Validation**. The model is trained on the **Training** partition and then validated on the **Validation** partition.

In the example project below, the Census dataset is split into Training and Validation partitions.



To learn how to partition a dataset, see the topic About Partitioning.

Validation Fields

The validation process creates a new dataset that contains model predictions and comparison fields. Creating some of these fields is optional. During the validation wizard setup, you choose which fields to include.

The output of the validation process depends on whether the input models have discrete or continuous dependent variable (dependent variable).

Validation Fields for Models with Discrete dependent variable

In Figure 2, the *Income* field is a discrete dependent variable with two possible values: **<=50K** and **>50K**. This field (also referred to as the *known outcome*) already exists in the dataset. Notice that the generated field names are prefixed with the dependent variable name, *Income*.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Figure 2 - Fields Created for a Discrete dependent variable

-c	Income	Income Prediction	Income <=50	Income >50K Prob	Income Predict Prob	Income Correct
	<=50K	<=50K	1	0	1	☑
	>50K	>50K	0	1	1	☑
	>50K	>50K	0.090909090	0.90909090909090	0.909090909090909	☑
	<=50K	<=50K	1	0	1	☑
	>50K	<=50K	0.701492537	0.29850746268656	0.701492537313433	☐
	<=50K	<=50K	0.713414634	0.28658536585365	0.713414634146341	☑
	>50K	>50K	0.271186440	0.72881355932203	0.728813559322034	☑

In this example, the generated fields are:

Income Prediction	The predicted value for a given record.
Income <=50K Prob **and** ***Income >50K Prob***	The probability of the specified outcome (value). For example, the field *income >50K Prob* contains the probability of the value "*>50K*". Usually, the probability of the value of interest is the only field necessary for the analysis. The probabilities of other dependent variable values are optional.
Income Predict Probability	The probability of the predicted value (i.e., of the value in the *Income Prediction* field in this case).
Income Correct	Indicates whether the predicted value coincides with the actual dependent variable value. Provides a quick way to determine if the prediction is correct for a given record.

Note that the *Income Correct* column in Figure 2 shows one record that is not checked. This means the prediction does not match the actual *Income* value. You can verify this directly by comparing the values in the corresponding columns (*Income* and *Income Prediction*).

Validation Fields for Models with Continuous dependent variable

Because the "correctness" of a prediction on a continuous dependent variable is not as discernible as it is for a discrete dependent variable, a hit range is specified during the validation wizard. Predicted values within that range are assessed as correct.

In Figure 3, the *age* field is a continuous dependent variable. As with the discrete dependent variable example above, this field already exists in the dataset. Similarly, the generated field names are prefixed with the dependent variable name, in this case, *age*.

Figure 3 - Fields Created for a Continuous dependent variable

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

age	age Prediction	age Correct	age Std Dev	age Error	age Standardi	ageAbsolute	age Squared
30	34.6918489066	☐	13.04705219	-4.691848906	0	4.691848906	22.01344616
26	38.1526586621	☐	13.03777367	-12.15265866	-1	12.15265866	147.6871125
53	48.3490566038	☐	18.63485850	4.650943396	0	4.650943396	21.63127447
22	40.9773584906	☐	11.72860142	-18.97735849	-2	18.97735849	360.1401352
38	38.1526586621	☑	13.03777367	-0.152658662	0	0.152658662	0.023304667
18	40.9484873262	☐	14.47908737	-22.94848732	-2	22.94848732	526.6330705

In this example, the generated fields are:

age Prediction	The predicted value for a given record.
age Correct	Indicates whether the predicted value is close enough to the actual dependent variable value to be considered a correct prediction. This depends on the user-defined **valid hit range** specified in the validation wizard. See the topic Validation Wizard for more information.
age Std Dev	The standard deviation of the dependent variable based on the distribution in the entire dataset.
age Error	The difference between the actual value and the predicted value.
age Standardized Error	The standard deviation of the error.
age Absolute Error	The absolute value of the error.
age Squared Error	The squared value of the error.
age Node ID	For Decision Tree models only, this is the node ID of the tree node to which the record is assigned.

Further Analysis

Once the validation is complete, there are a number of ways to analyze the results. Charts created in the Model Analyzer, such as cumulative lift, K-S and ROC charts, will help you determine how well the model performed during validation.

Special Notes on Validating Decision Trees

Validating a decision tree is a little different from validating other models due to the interactive character of trees and the fact that the validation fields are calculated columns depending on the validated models.

If you validate a tree model, the calculated columns created with the validation operation refer to a read-only copy of the tree being validated, not to the original tree. This read-only version of the original tree is created automatically by the validation process and is called a “snapshot tree”. This prevents a variety of problems that can occur if a referenced model is changed after validation (especially if validation fields were appended to the parent dataset) and allows

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

the user to delete and modify trees without breaking any dependencies. In addition, the user always has a copy of a tree as it was at the time of validation.

In-Database Validation

Validation can be performed within a database if your product includes the In-Database Analytics feature. The purpose of In-Database validation is to execute the model validation commands directly on the database server and store the resulting validation data there. The Altair Data Mining Engine sends the model validation commands to the database engine, which reads the records from the input table and writes the results to a new table. No data extraction occurs in this process, and the validation is performed by the database engine rather than the Altair Data Mining Engine.

Proceed to Validation Wizard.

See also: About the Model Analyzer

Validation can be performed for all predictive models, decision trees, and strategy trees. Validation is not applicable to Cluster Analysis and Market Basket Analysis. For general information about model validation, please see the topic About Validation.

This topic describes model validation that produces a dataset with predictions and comparison metrics. Model performance evaluation and comparison tasks that produce charts such as cumulative lift, K-S and ROC, are based on the validation results. They are described in the topic About Model Analyzer.

For interactive models - Decision Trees, Strategy Trees, and Market Basket Analysis - Model Validation is applied to model instances, since it is important to know what the state of the model was at the time of validation. If you apply validation to a tree *process* node or a market basket analysis process node, a model instance will be automatically saved before validation, and the validation will be applied to the instance.

Note: The context and meaning of Strategy Tree validation is different from those of other models. For validating Strategy Trees, see the topic About Strategy Validation.

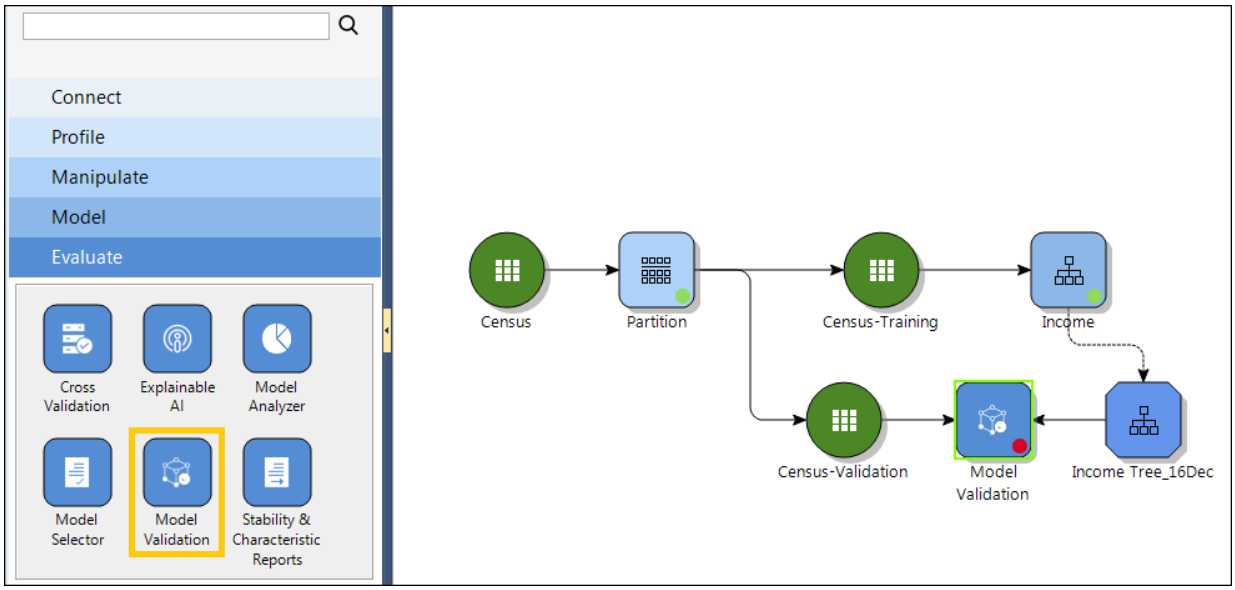
To add a model validation operation to your workflow:

- Drag the **Model Validation** node from the **Evaluate** palette onto the Workflow canvas and connect the input model and dataset nodes to this node. If your model is a Decision Tree, a Strategy Tree, or a Market Basket Analysis process node (rather than a model instance node), you will be prompted to create a model instance first.

OR

- Right-click anywhere on the workflow canvas and select the context menu command **Evaluate --> Model Validation**. This will create a new Model Validation node. Connect the input dataset and model nodes to it.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

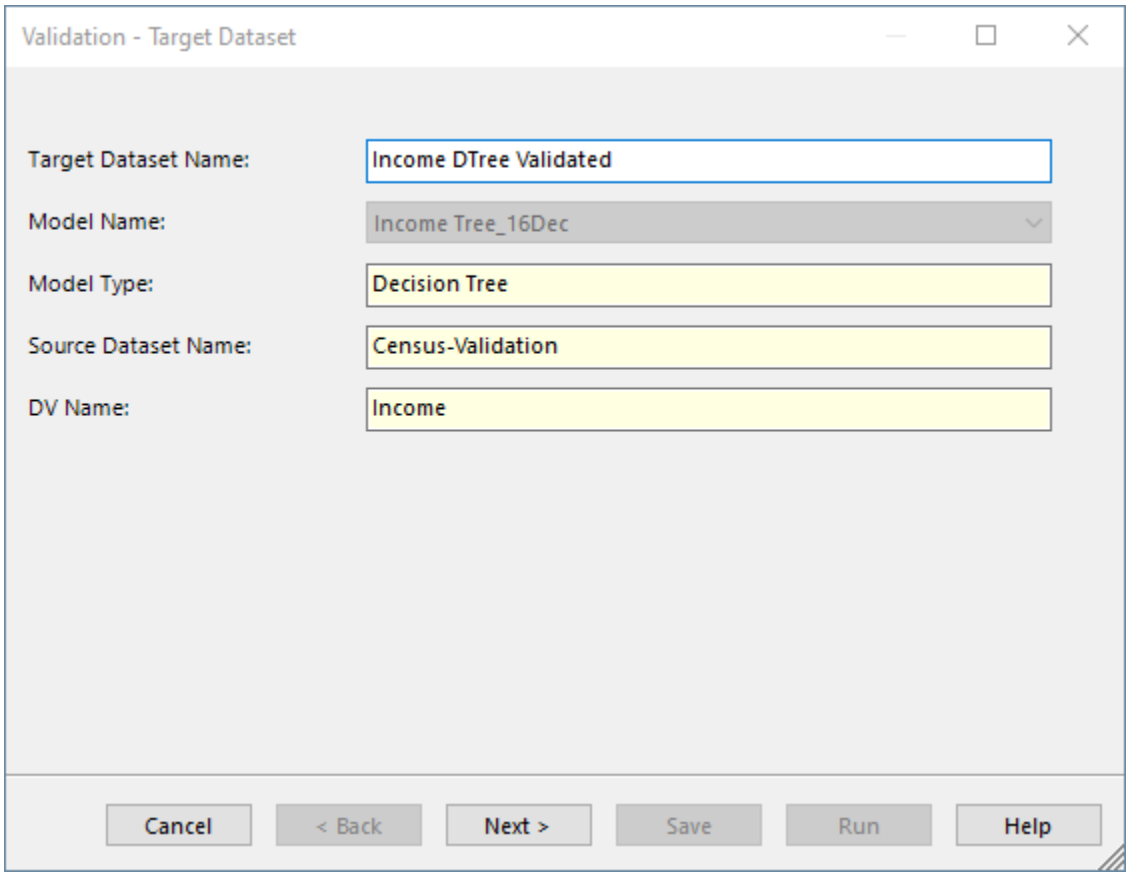


To define the validation parameters, double-click the **Model Validation** node or right-click on it and select **Modify**.

Step 1

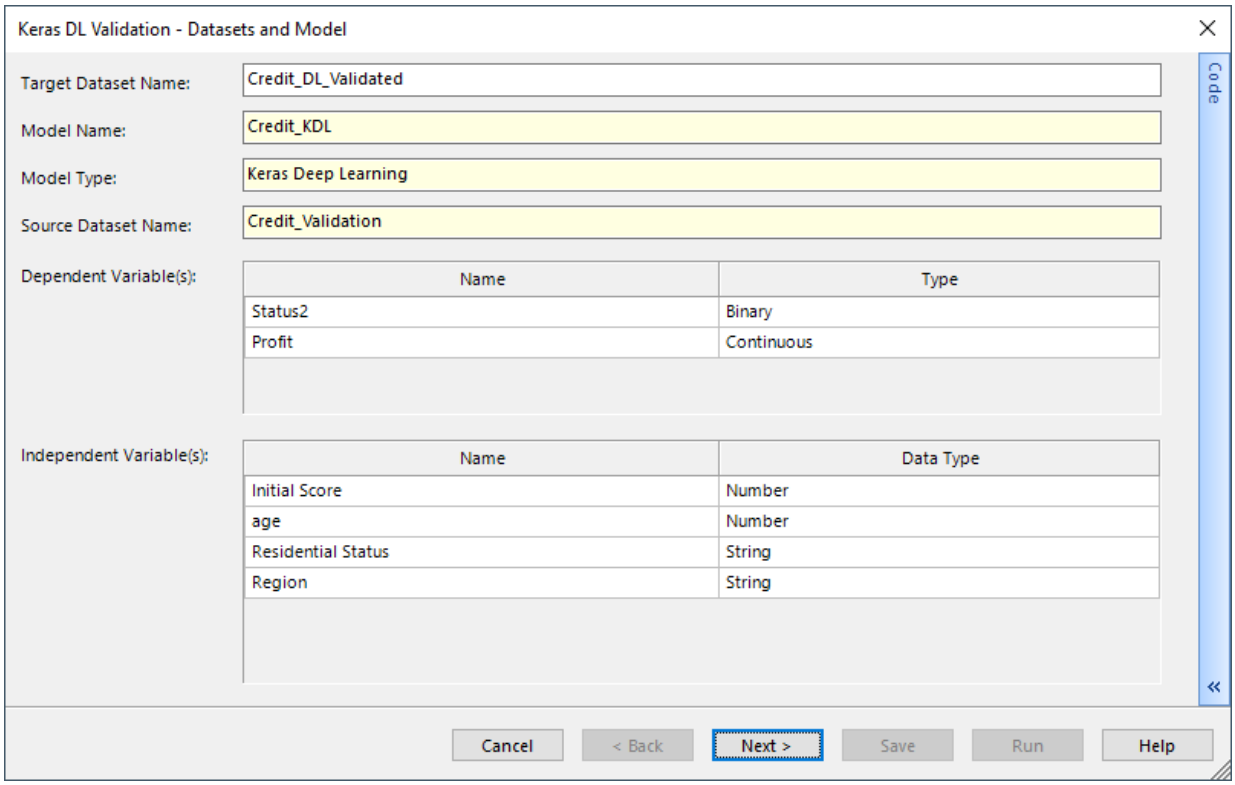
Specify the name of the output dataset in the Target Dataset name field. The name of the input model, input dataset, and the name of the dependent variable are also displayed in the first page.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



For models implemented in Python, such as the Keras Deep Learning model, the first page of the validaion wizard may also contain information about their dependent and independent variables:

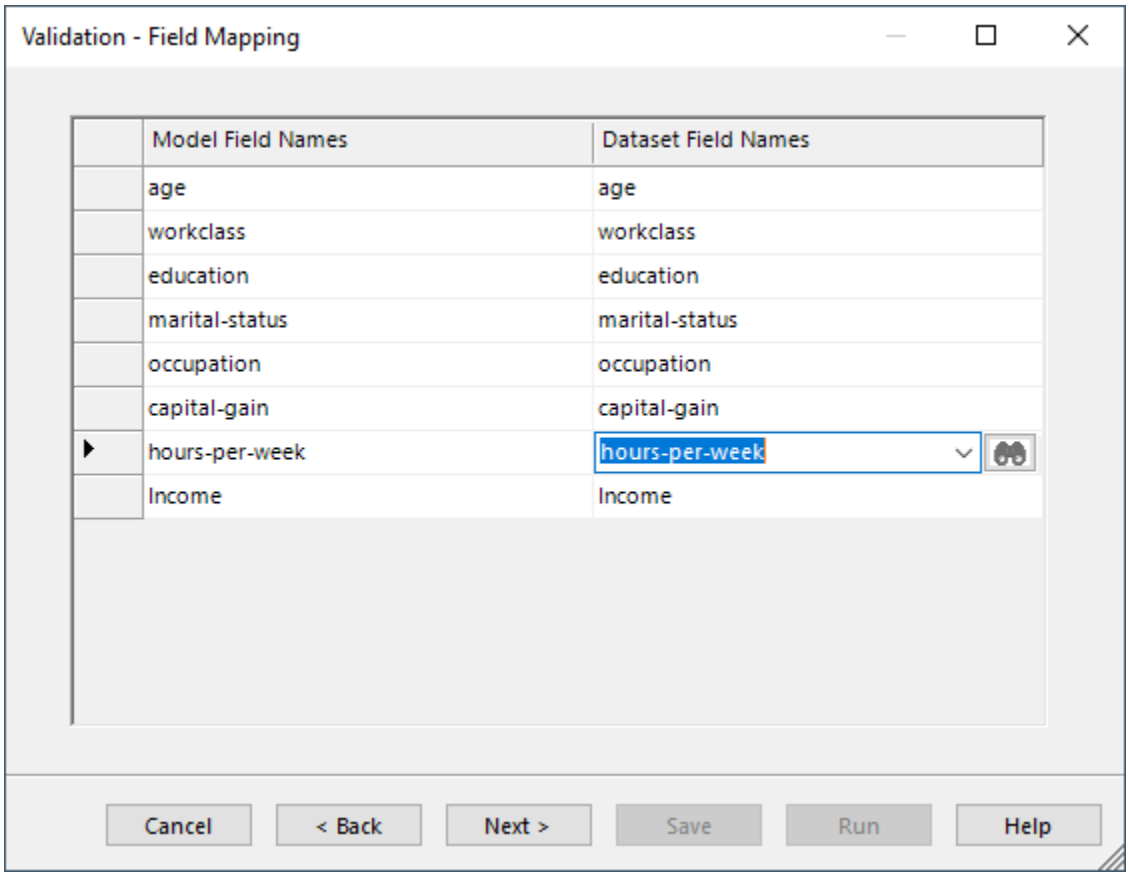
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Step 2

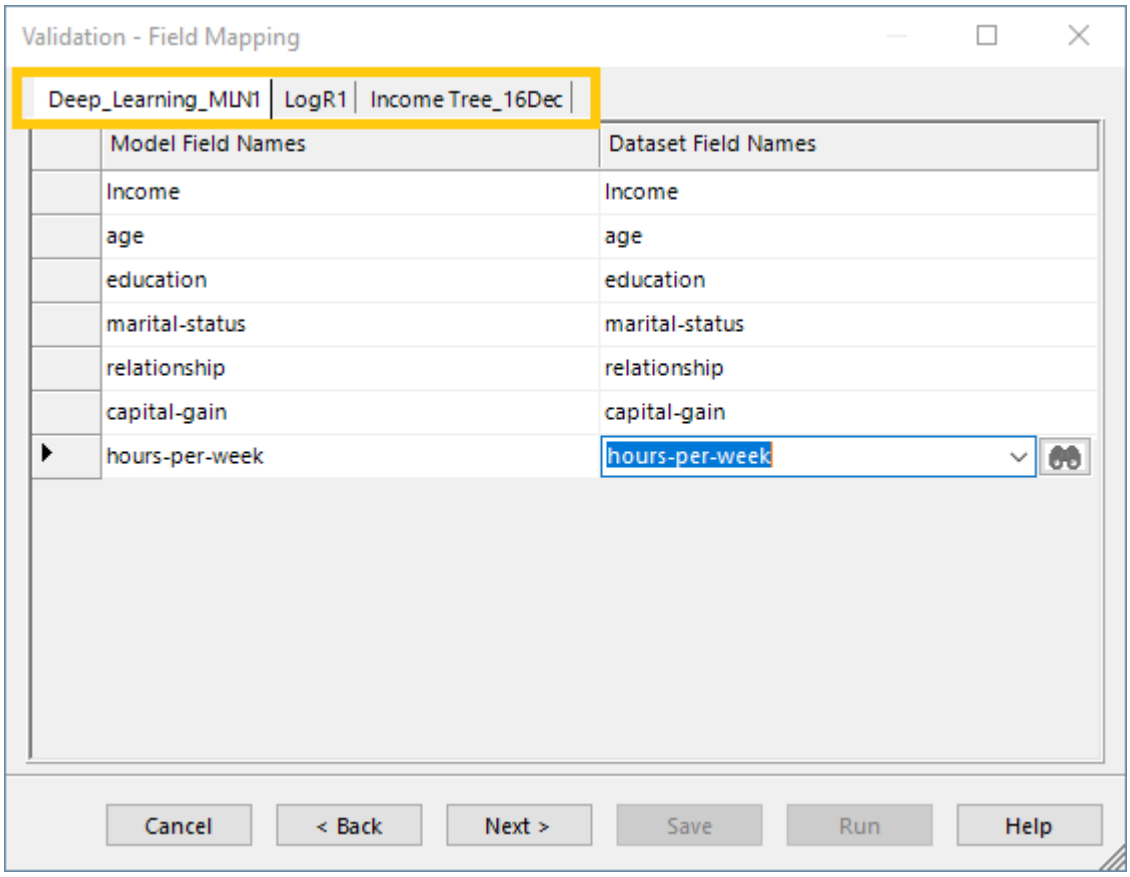
The next step provides the option of mapping the field names in the model to the ones in the validation input dataset. In some situations, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map fields by clicking in the right column (Dataset Field Names) and selecting the right fields to match the model variables in the left column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



In the special case where the input model is a **Model Stacking** node, the field mapping needs to be defined for all models, so the Field Mapping page will have multiple tabs - one tab per each model within the Model Stacking node. In the following example, there are 3 models in the input Model Stacking node. Click on each tab to verify or change the mapping for each model.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



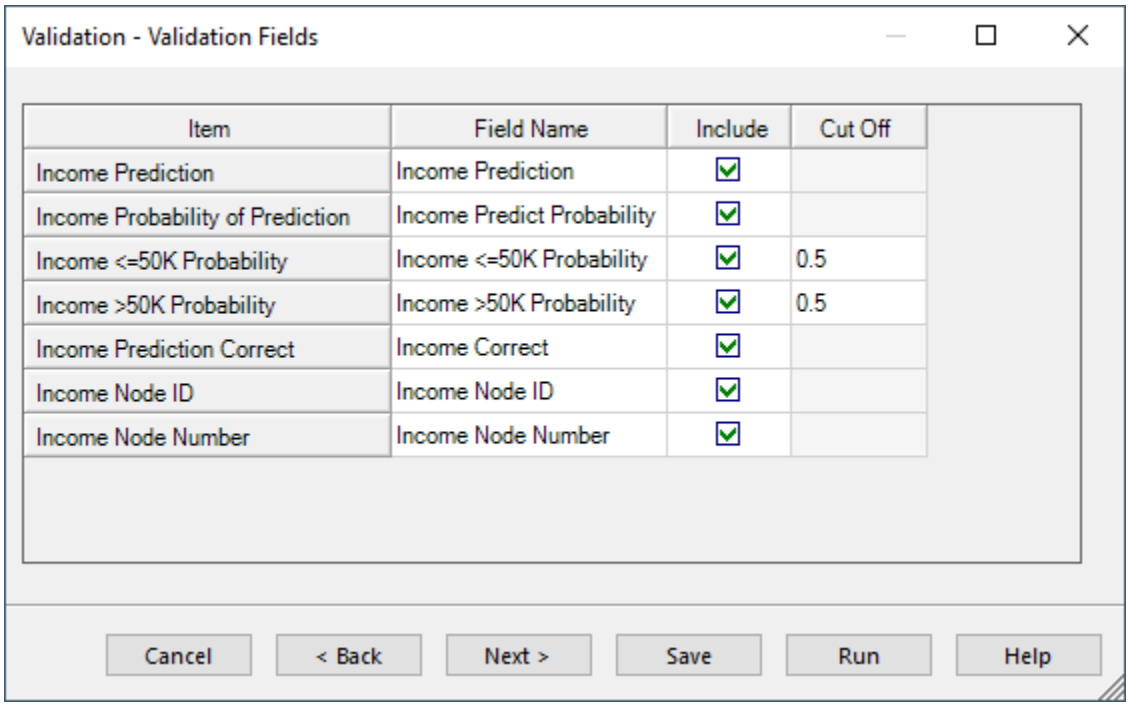
For models implemented in Python, such as Keras Deep Learning, GLM, and XGB, see also the description of the corresponding variants of the Model Validation wizard in the Help section **Model Evaluation | Model Validation**.

Step 3

At the next step, choose the new validation fields to be generated for the output dataset. Generated field names are prefixed with the name of the dependent variable. In the case of a binary DV (a DV with two values), you can also specify a **cut off** value (see the explanation below).

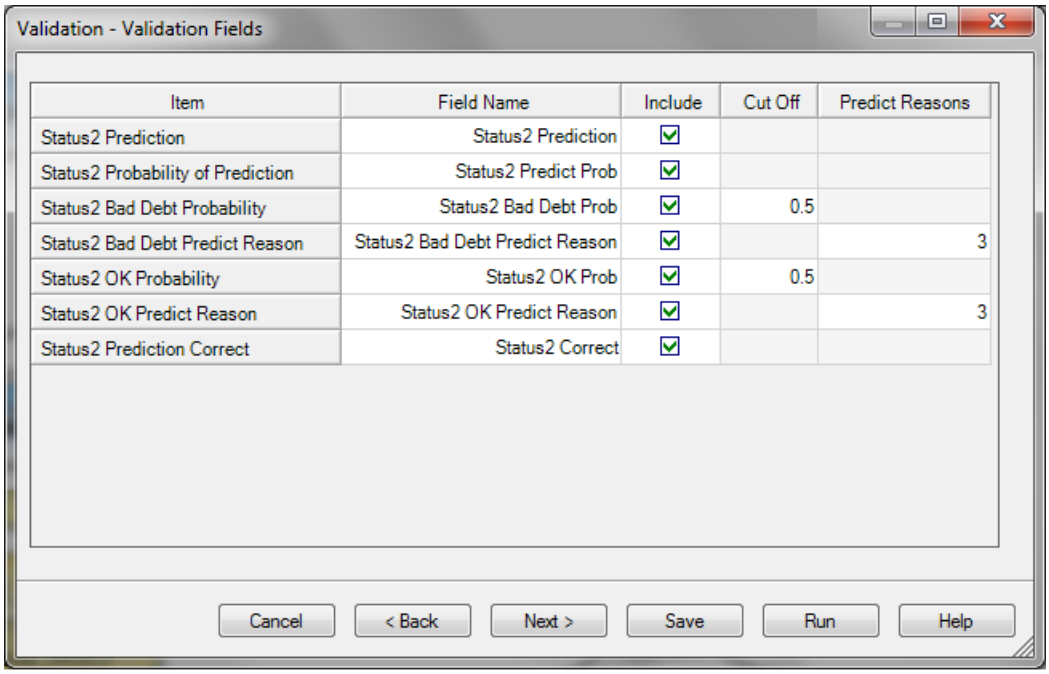
The validation fields produced by validating models with discrete dependent variable (DV) are different from those produced by models with continuous DV. For example, for a Decision Tree model with discrete DV, the scoring fields are: DV Prediction, DV Probability of Prediction, DV Value Probability (for each DV value), DV Prediction Correct, Node ID, and Node Number.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



For a Logistic Regression model with discrete DV, the validation fields are: DV Prediction, DV Probability of Prediction, and DV Value Probability for each DV value. For models with binary DV, there are also Predict Reasons that can be included for each DV value. Predict Reasons explain why the model is making a particular prediction. This a comma-separated list of variable names that contributed the most to the prediction. In the Predict Reasons field you can specify the maximum number of reasons (contributing variables) to include. For example, in the example screenshot below, the field Status2 OK Predict Reason will include no more than 3 reasons for predicting the value "OK" for the dependent variable Status2. Predict Reasons are also available for Linear Regression. For details on the calculations behind Predict Reasons, see the section Predictive Models: Regression in Mathematical Formulations.

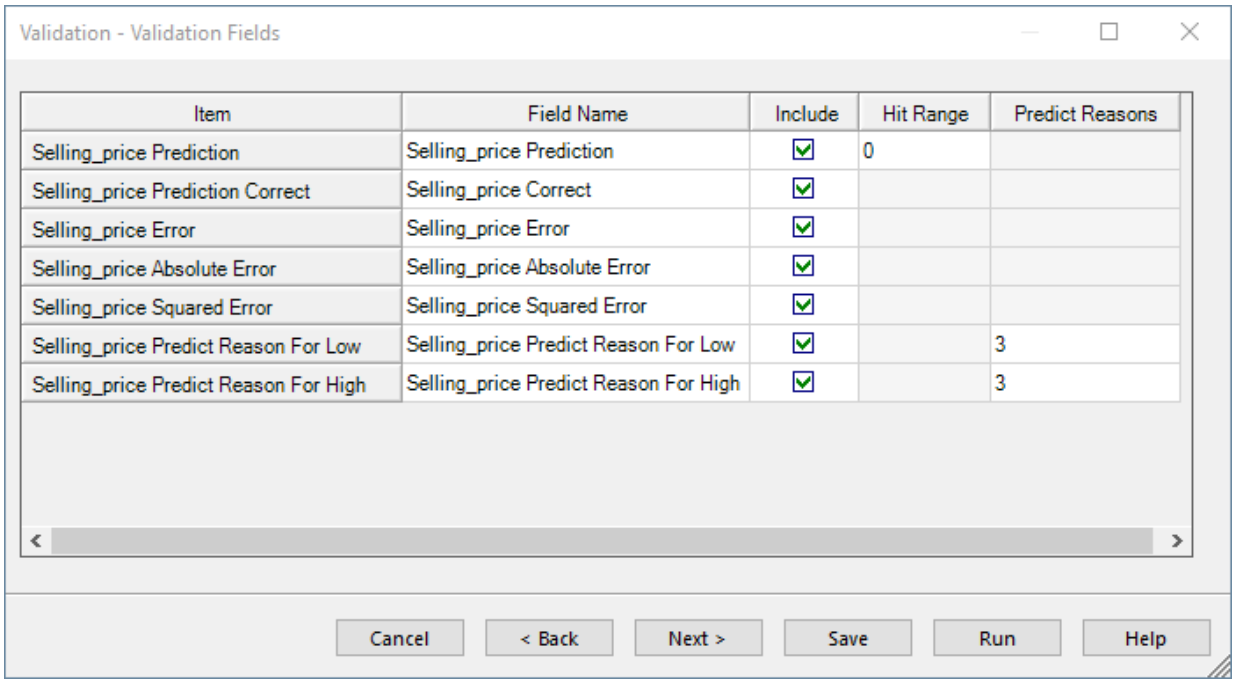
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Prediction	The predicted value for a given record.
DV Value Probability	The probability of the specified outcome (value). For example, the field *Income >50K Prob* contains the probability with which the model predicts the value "*>50K*". Usually, the probability of the value of interest is the only field necessary for the analysis. The probabilities of other DV values are optional. If the dependent variable is binary, you can specify the probability cut-off point to define the prediction of the DV value (set in the DV Prediction field). For example, if your dependent variable is *Income*, with two string values, ">50K" and "<=50K", and you set the cut-off for *Income <=50K Probability* to **0.7**, the *Income Prediction* field will be set to "<=50K" when the *Income <=50K Probability* value is **0.7** or greater. Changing the cut-off for the probability of one the DV values will automatically adjust the cut-off for the other one to provide the total of 1.
Probability of Prediction	The probability of the DV value predicted by the model. This is the maximum of the probabilities of all DV values calculated by the model.
Prediction Correct	Indicates whether the predicted value coincides with the actual DV value. Provides a quick way to determine if the prediction is correct for a given record.
Predict Reason	Applicable only to Logistic Regression with binary dependent variable and to Linear Regression. Predict Reasons explain why the model is making a particular prediction. This a comma-separated list of variable names that contributed the most to the prediction. Specify the maximum number of reasons to include.

The Validation fields produced by models with continuous DV are different:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



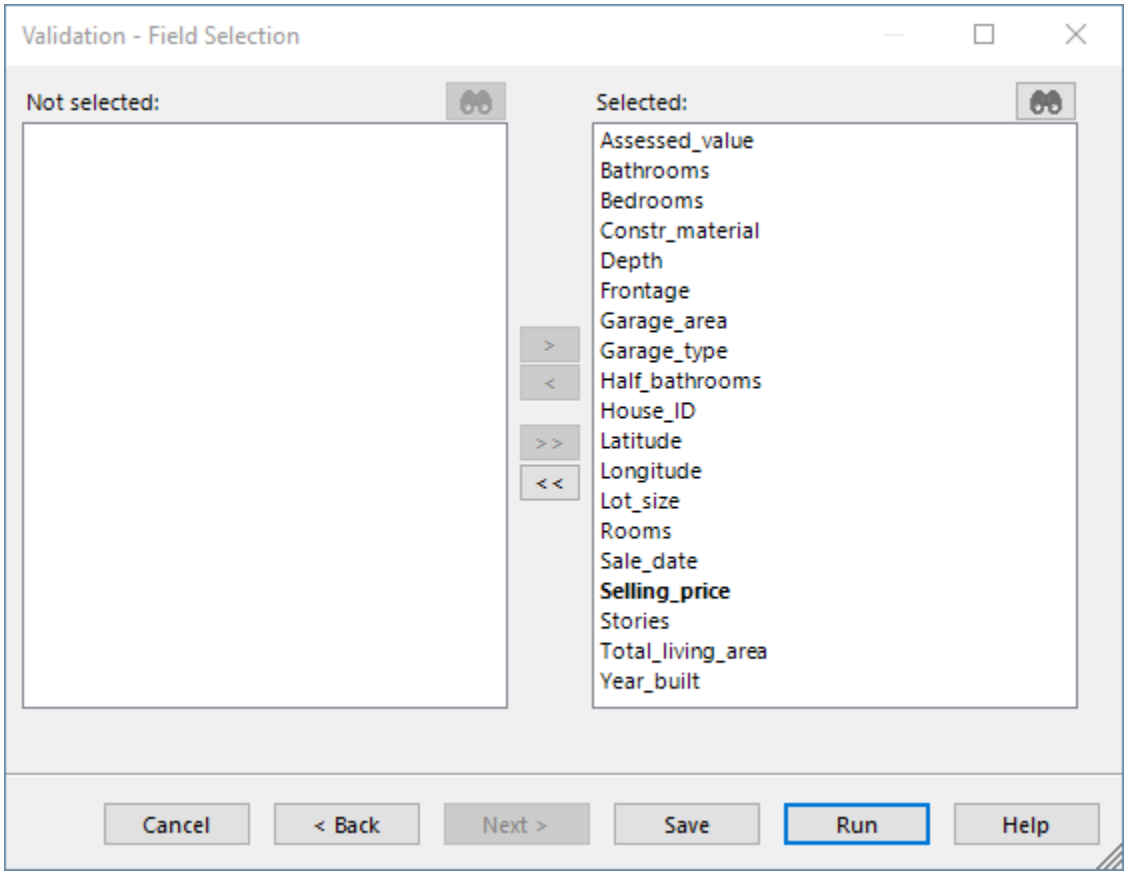
Prediction	The predicted value for a given record.
Correct	Indicates whether the predicted value is close enough to the actual DV value to be considered a correct prediction. This depends on the user-defined valid ***hit range*** that can be specified in the Hit Range column on the right. If you want a prediction to be considered correct whenever it falls into the interval [*x-r*, *x+r*], where x is the actual DV value, specify ***r*** as the Hit Range value.
Std Dev	The standard deviation of the DV based on the distribution in the entire dataset.
Error	The difference between the actual value and the predicted value.
Standardized Error	The standard deviation of the error.
Absolute Error	The absolute value of the error.
Squared Error	The squared value of the error.
Predict Reason	Applicable only to Linear Regression and Logistic Regression with binary dependent variable. Predict Reasons explain why the model is making a particular prediction. This a comma-separated list of variable names that contributed the most to the prediction. Specify the maximum number of reasons to include.

For models implemented in Python, such as Keras Deep Learning, GLM, and XGB, see also the description of the corresponding variants of the Model Validation wizard in the Help section **Model Evaluation | Model Validation**.

Step 4

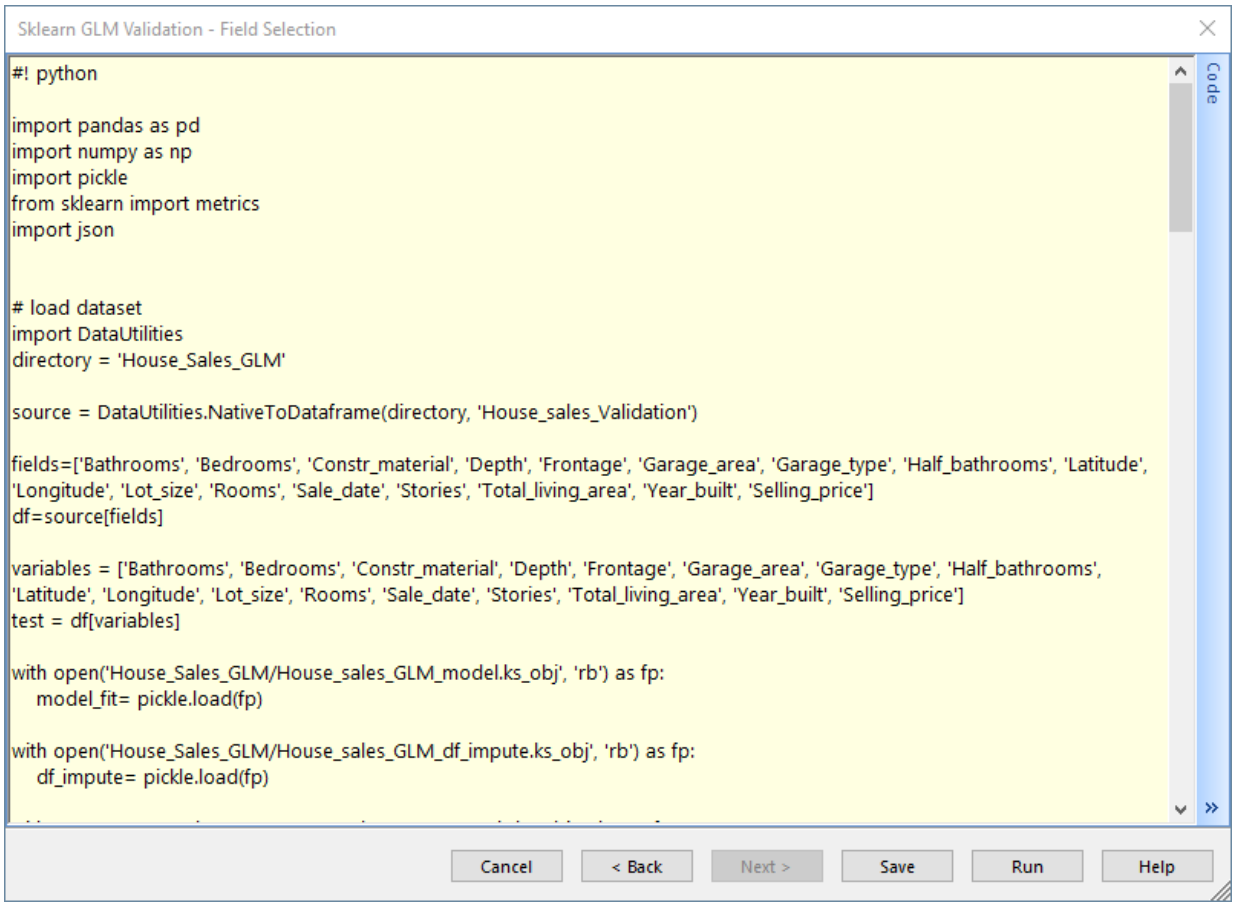
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

At the **Field Selection** step, choose the fields from the input dataset that you would like to include in the validation output in addition to the new score / probability fields. The fields required by the model are highlighted in bold font and cannot be excluded. By default, all fields in the validation input dataset are selected to be included in the output.



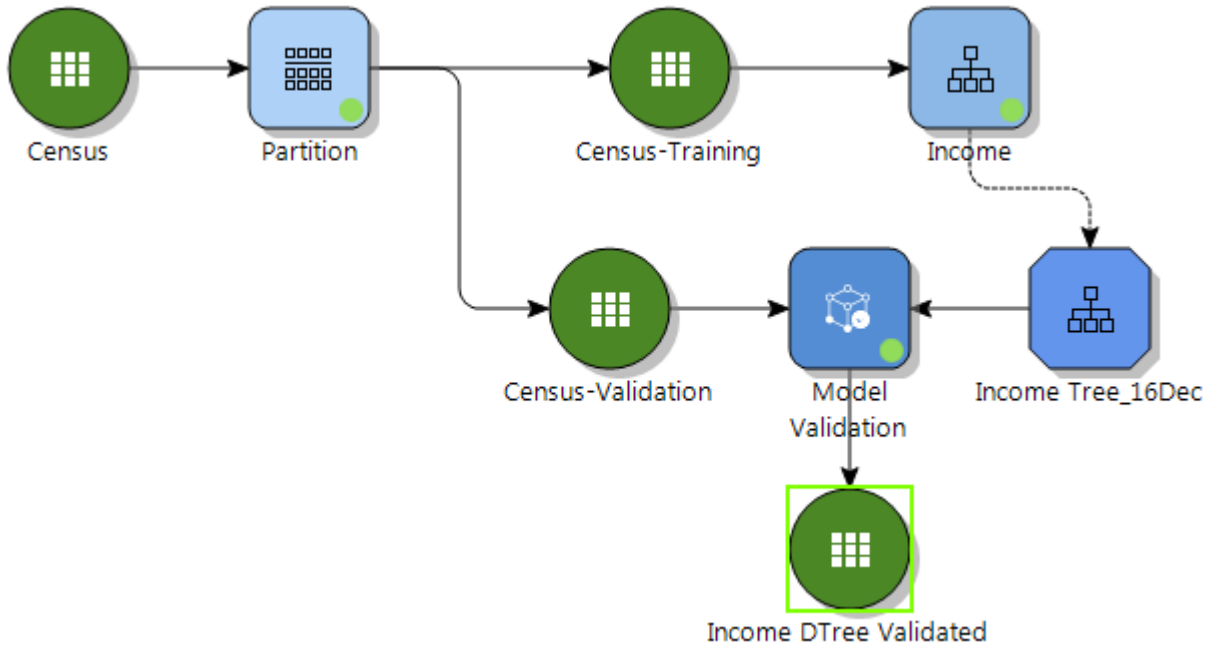
When validating a model implemented in Python, such as Keras Deep Learning, GLM, or XGBoost, you can preview the Python code generated for your current inputs by clicking on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel. For details, see the description of the corresponding variants of the Model Validation wizard in the Help section **Model Evaluation | Model Validation**.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the validation process immediately. The output dataset node will be created in the workflow after running the validation node. It can be viewed and manipulated like any other dataset.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Proceed to the Validation Report topic.

Keras Deep Learning Validation

A Keras Deep Learning model is validated using the **Model Validation** node in the **Evaluate** palette. For general information about validation, see the topic About Validation.

Prerequisites for Keras Deep Learning model validation: a **Keras Deep Learning** node and an input dataset to validate.

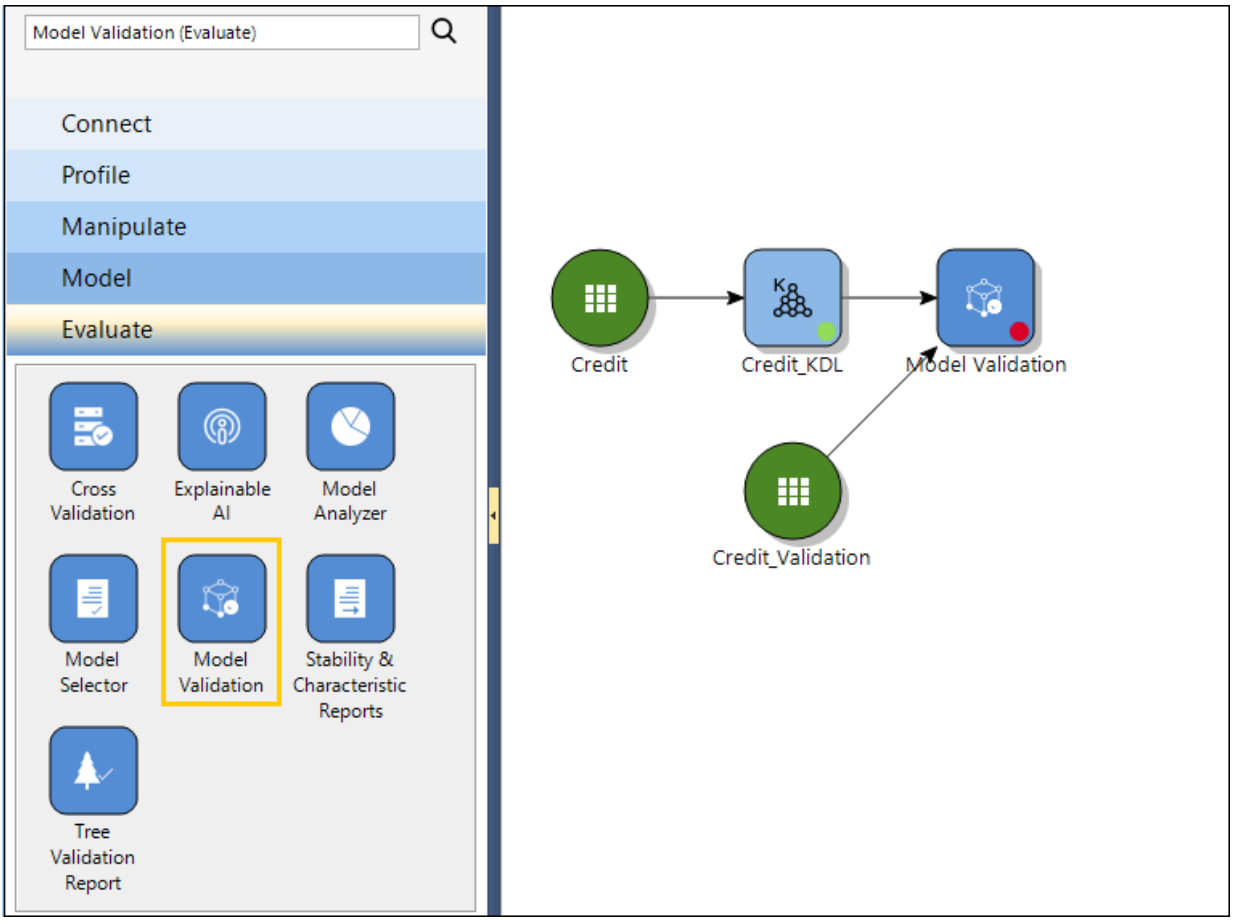
To add a Model Validation operation to your workflow:

- Drag the **Model Validation** node from the **Evaluate** palette onto the Workflow canvas and connect the inputs: a **Keras Deep Learning** node and a dataset node that contains the data to validate.

OR

- Right-click anywhere on the workflow canvas and select the context menu command **Evaluate --> Model Validation**. This will create a new Model Validation node. Connect the inputs: a **Keras Deep Learning** model node and a dataset node that contains the data to validate.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

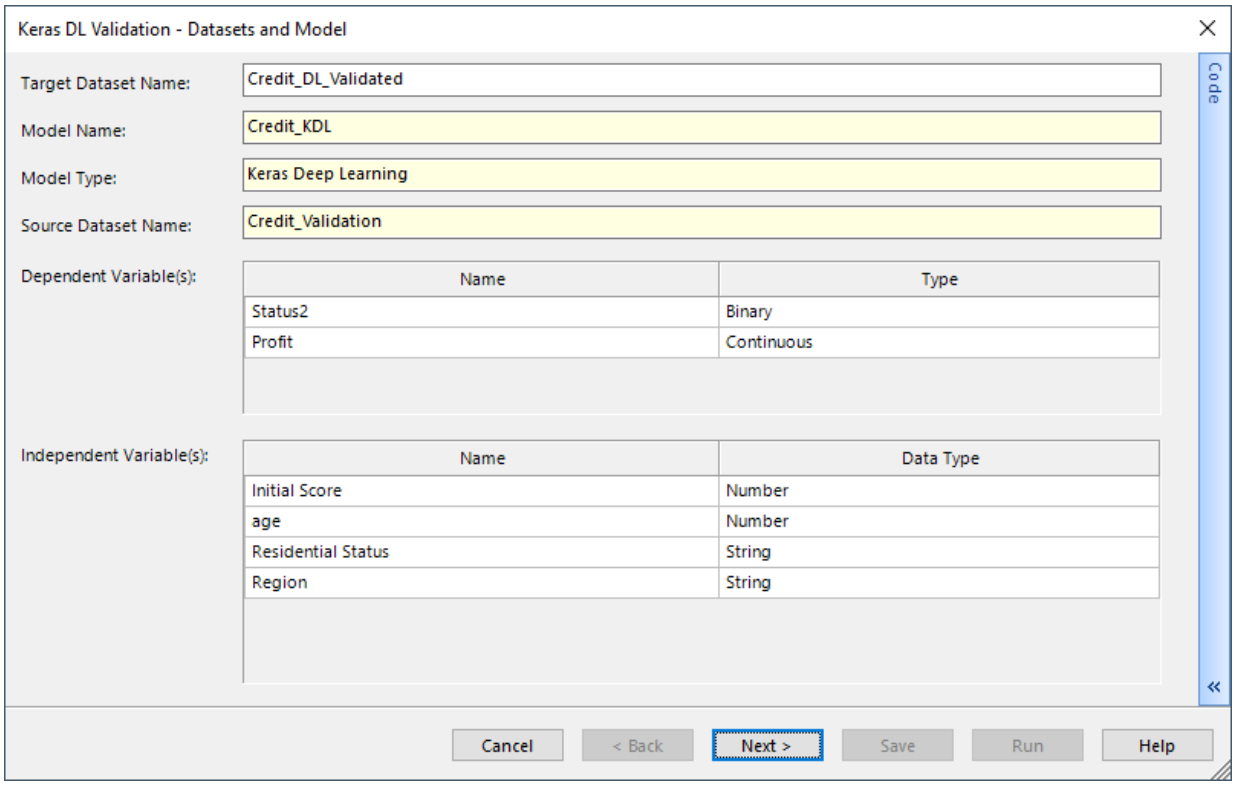


Double-click the **Model Validation** node or right-click on it and select **Modify**.

The Keras Deep Learning Validation wizard opens. It displays the names of the input model and dataset, the names and types of the dependent and independent variables of the model.

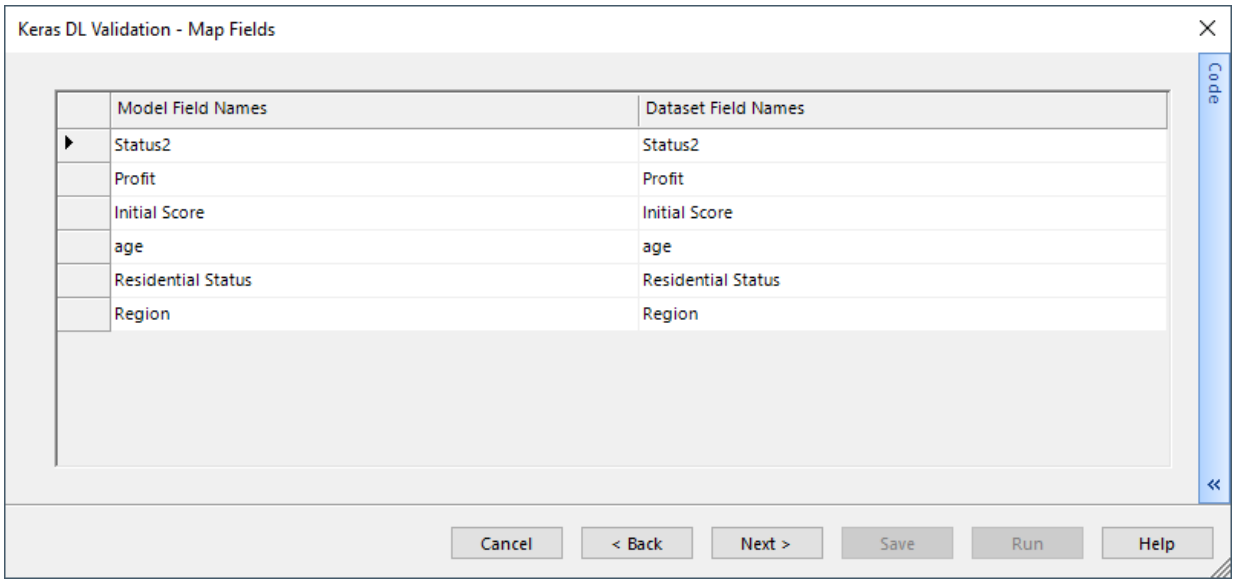
In the *Target Dataset Name* field, specify the name for the output dataset that will contain the validation results.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

The next step provides the option of mapping the field names in the model to the ones in the validation input dataset. In some cases, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map some fields by clicking in the Dataset Field Names column and selecting the right fields to match the model variables in the left column:

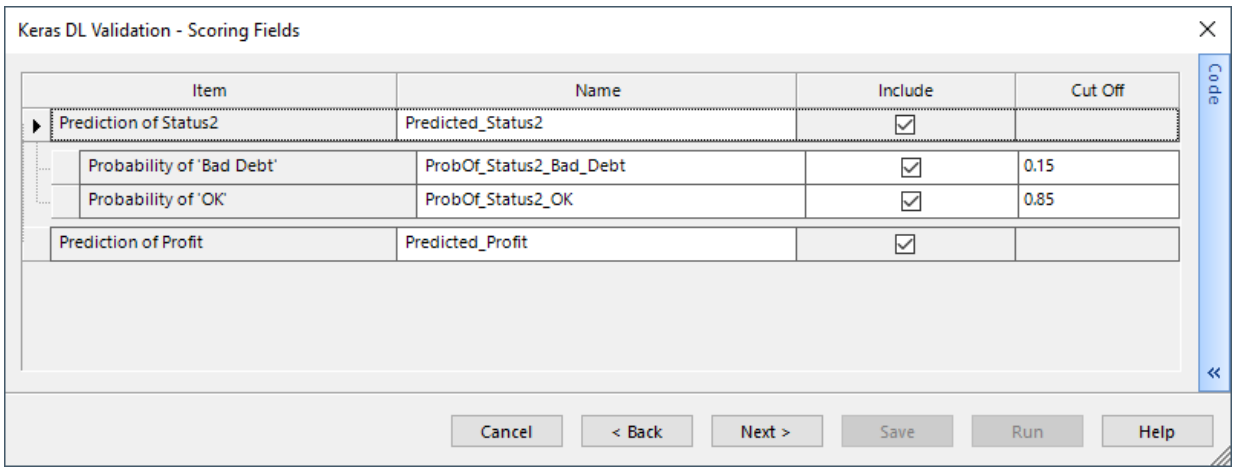


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Note: When validating Keras Deep Learning models that use derived variables (also called calculated columns) with expressions, the input dataset must include these variables. This is different from the case of native Altair models with derived variables, which can be validated on a dataset with only *original* variables that the derived ones depend on, while the derived ones are not required.

Click **Next**.

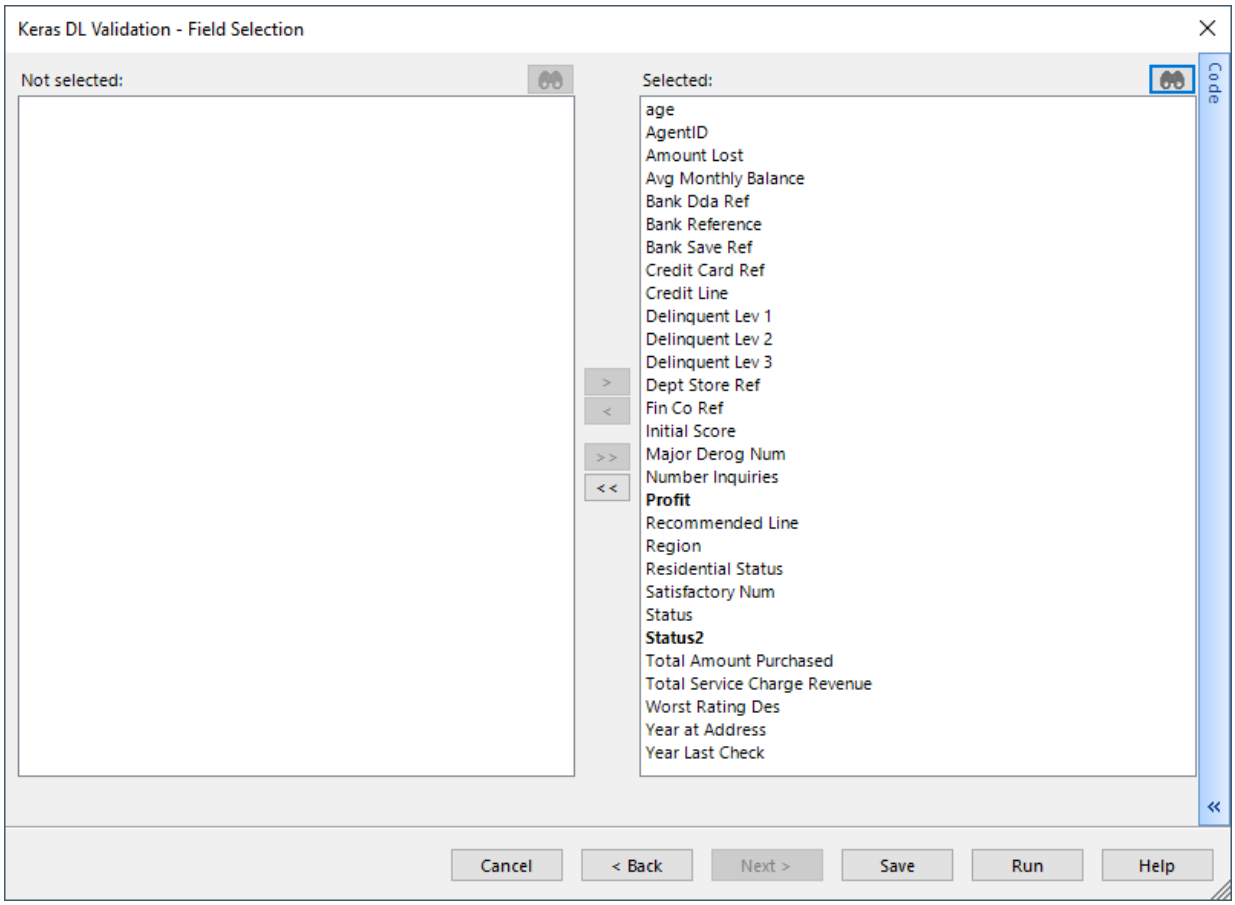
In the **Scoring Fields** page, select the scoring fields to be included in the scoring output dataset and modify their names if necessary. For discrete dependent variables, specify the probability cut-off values. See the general Validation Wizard topic for more information on cut-off values.



Click **Next**.

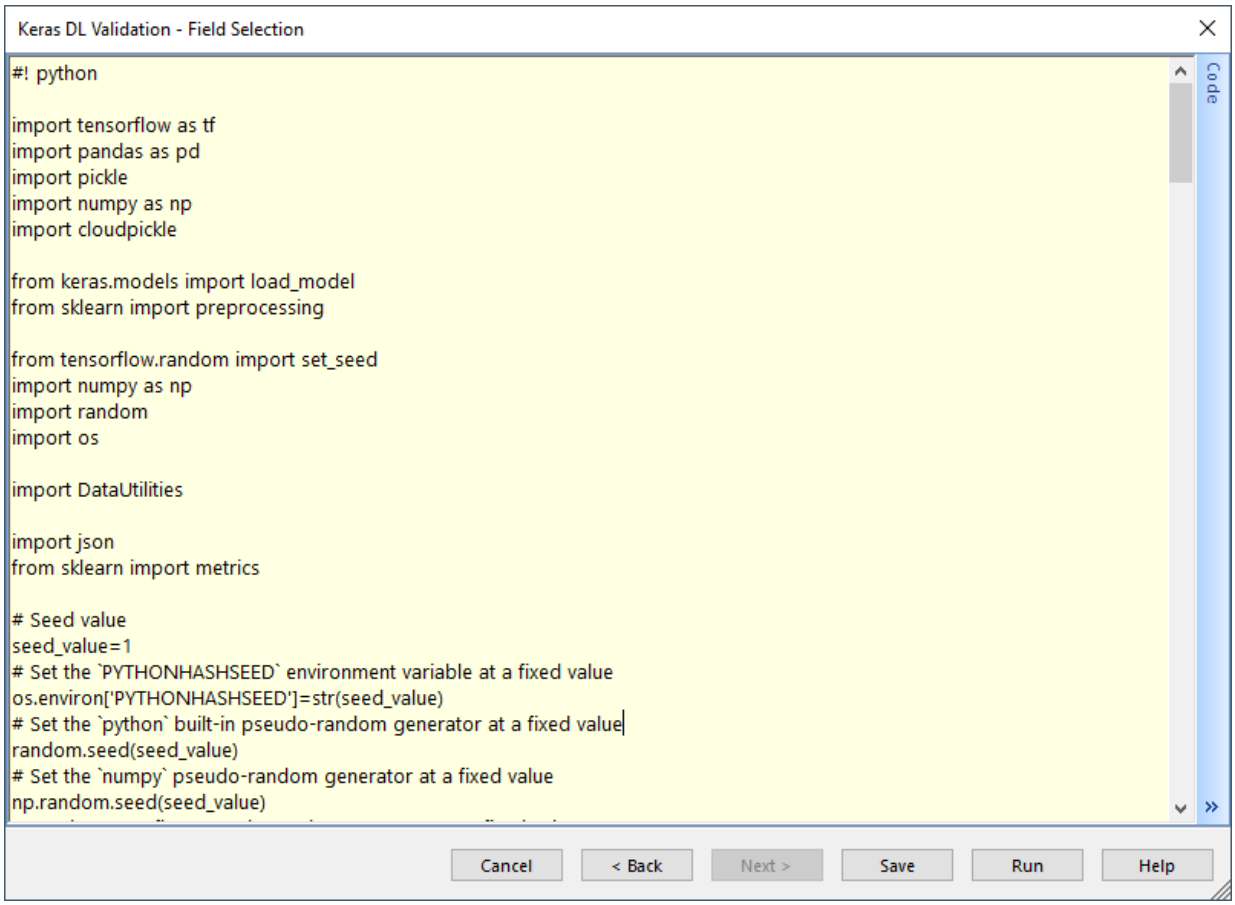
In the **Field Selection** page, choose the fields from the input dataset that you would like to include in the validation output in addition to the new score / probability fields. The field(s) representing the dependent variable(s) are highlighted in bold font and cannot be excluded. By default, all fields in the validation input dataset are selected to be included in the output.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

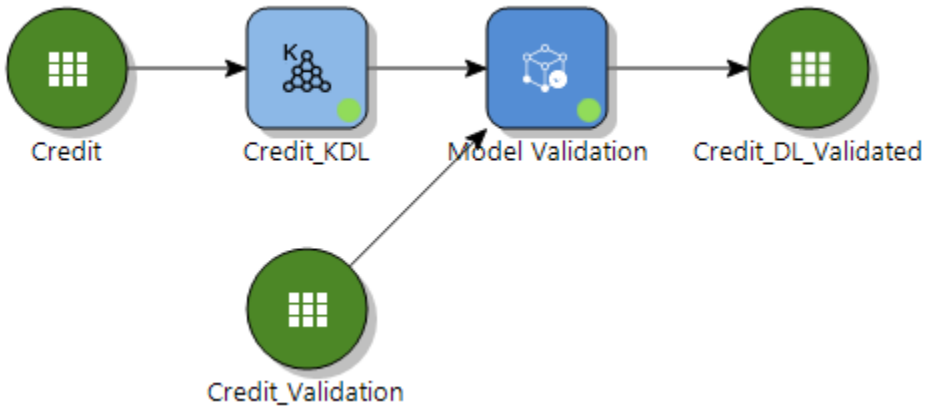


To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. You can also use the right-click context menu to copy the code, save it in a file, or search for specific code fragments. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the validation process immediately.

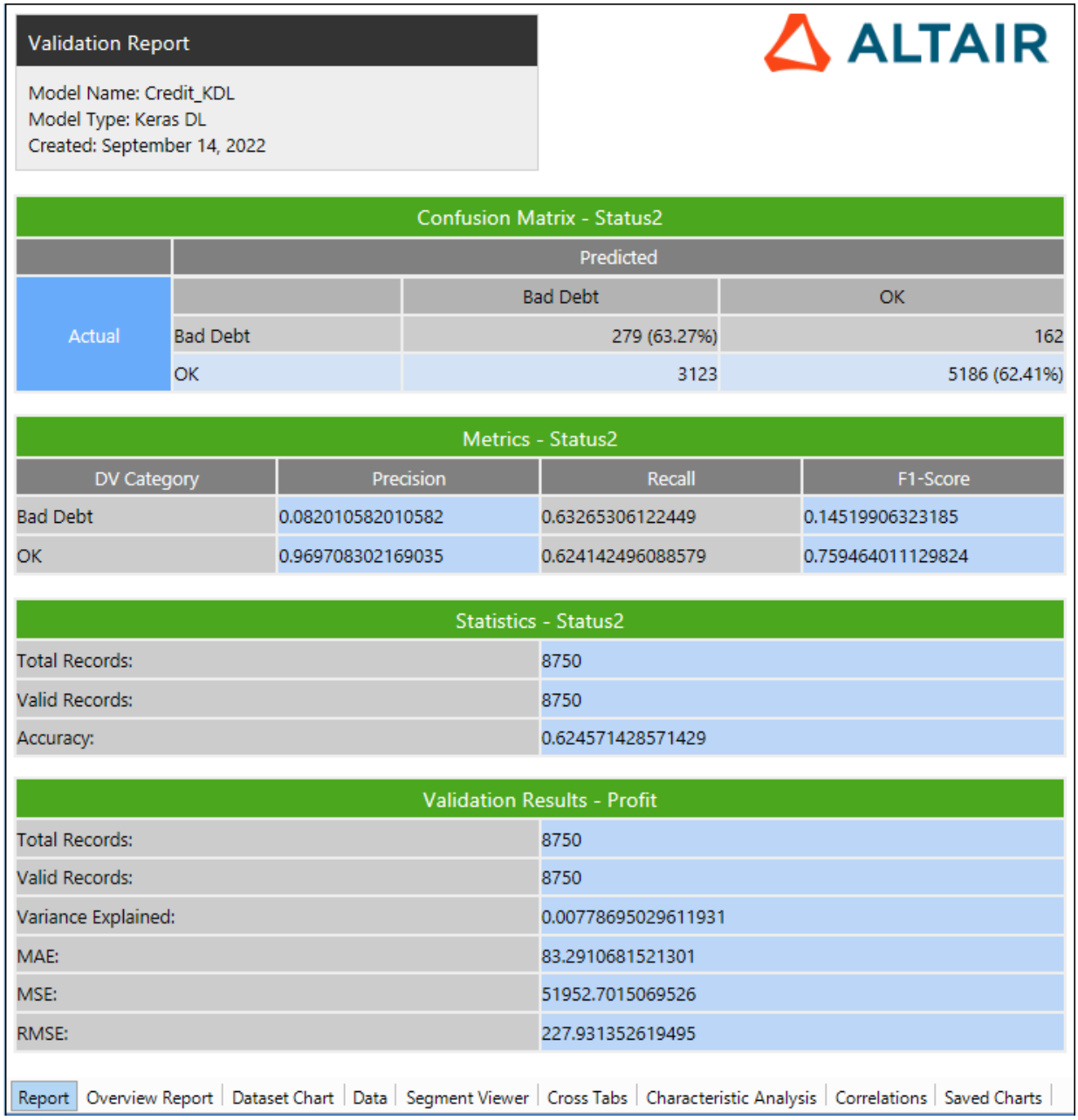


Double-click on the output dataset to view the validation results. The first tab of the output dataset is the Validation Report. It displays the values of the model evaluation metrics.

- For discrete dependent variables, along with the model accuracy, the Report tab shows the Confusion Martix, the table showing *Precision*, *Recall*, and *F1-Score.*

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- For continuous dependent variables, the Report tab shows Variance Explained (R-Squared), Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE).



Validation Report

Model Name: Credit_KDL
Model Type: Keras DL
Created: September 14, 2022

ALTAIR

Confusion Matrix - Status2			
	Predicted		
		Bad Debt	OK
Actual	Bad Debt	279 (63.27%)	162
	OK	3123	5186 (62.41%)

Metrics - Status2			
DV Category	Precision	Recall	F1-Score
Bad Debt	0.082010582010582	0.63265306122449	0.14519906323185
OK	0.969708302169035	0.624142496088579	0.759464011129824

Statistics - Status2	
Total Records:	8750
Valid Records:	8750
Accuracy:	0.624571428571429

Validation Results - Profit	
Total Records:	8750
Valid Records:	8750
Variance Explained:	0.00778695029611931
MAE:	83.2910681521301
MSE:	51952.7015069526
RMSE:	227.931352619495

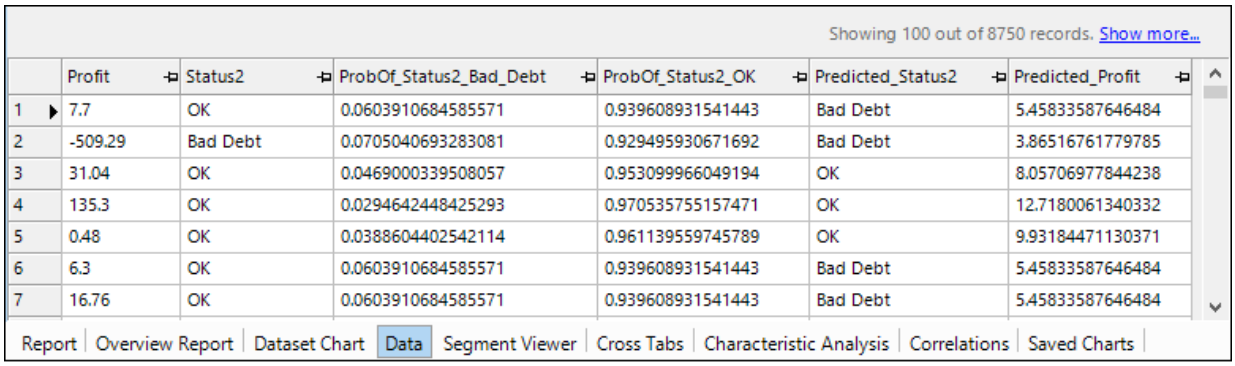
Report | Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

Click the **Data** tab of the output dataset and scroll to the rightmost columns to view the predictions and probabilities for the dependent variables.

- For discrete dependent variables, the output dataset contains the probabilities of the dependent variable values and the prediction (the value that has the highest probability) for each record.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- For continuous dependent variables, the output dataset contains the predicted value for each record.



Showing 100 out of 8750 records. Show more...

	Profit	Status2	ProbOf_Status2_Bad_Debt	ProbOf_Status2_OK	Predicted_Status2	Predicted_Profit
1	7.7	OK	0.0603910684585571	0.939608931541443	Bad Debt	5.45833587646484
2	-509.29	Bad Debt	0.0705040693283081	0.929495930671692	Bad Debt	3.86516761779785
3	31.04	OK	0.0469000339508057	0.953099966049194	OK	8.05706977844238
4	135.3	OK	0.0294642448425293	0.970535755157471	OK	12.7180061340332
5	0.48	OK	0.0388604402542114	0.961139559745789	OK	9.93184471130371
6	6.3	OK	0.0603910684585571	0.939608931541443	Bad Debt	5.45833587646484
7	16.76	OK	0.0603910684585571	0.939608931541443	Bad Debt	5.45833587646484

Report | Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

See also:

Keras Deep Learning

Keras Deep Learning Scoring

Sklearn GLM Validation

Validation of Sklearn GLM models is performed using the **Model Validation** node in the **Evaluate** palette.

For general information about validation, see the topic About Validation.

Prerequisites for Sklearn GLM model validation: Sklearn GLM node and an input dataset to validate.

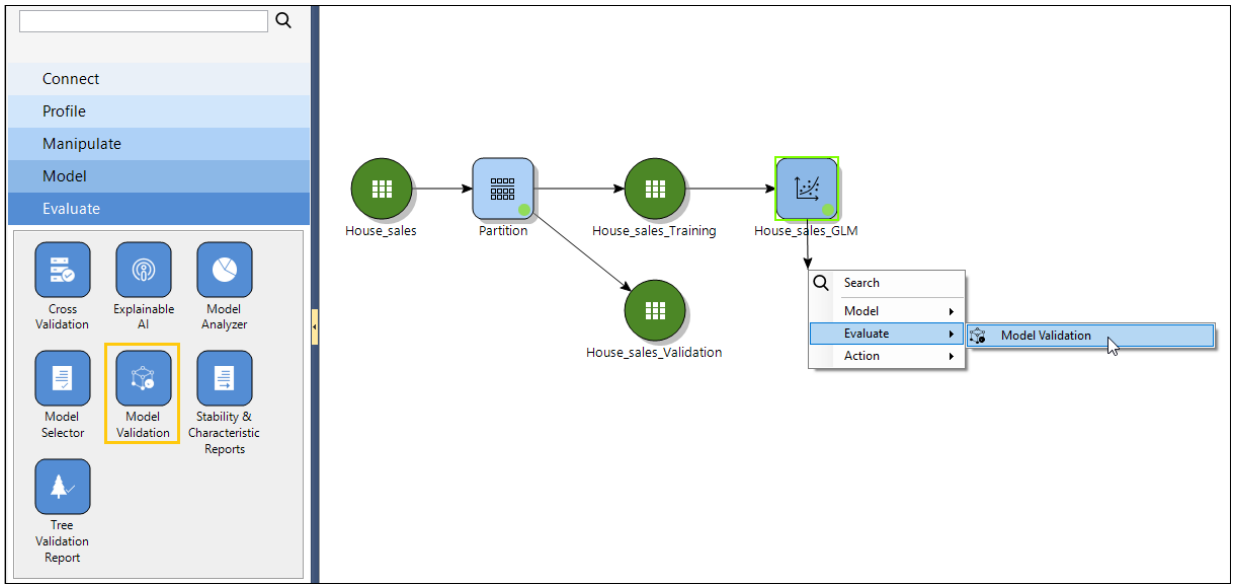
To add a Model Validation operation to your workflow:

- Drag the **Model Validation** node from the **Evaluate** palette onto the Workflow canvas and connect the inputs: an **Sklearn GLM** node and a dataset node that contains the data to validate.

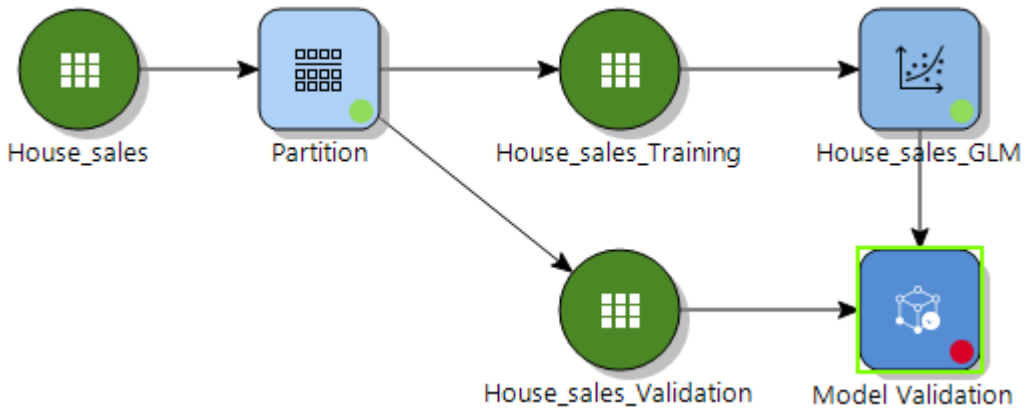
  OR

- Right-click anywhere on the workflow canvas and select the context menu command **Evaluate --> Model Validation**. This will create a new Model Validation node. Connect the inputs: an **Sklearn GLM** model node and a dataset node that contains the data to validate.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



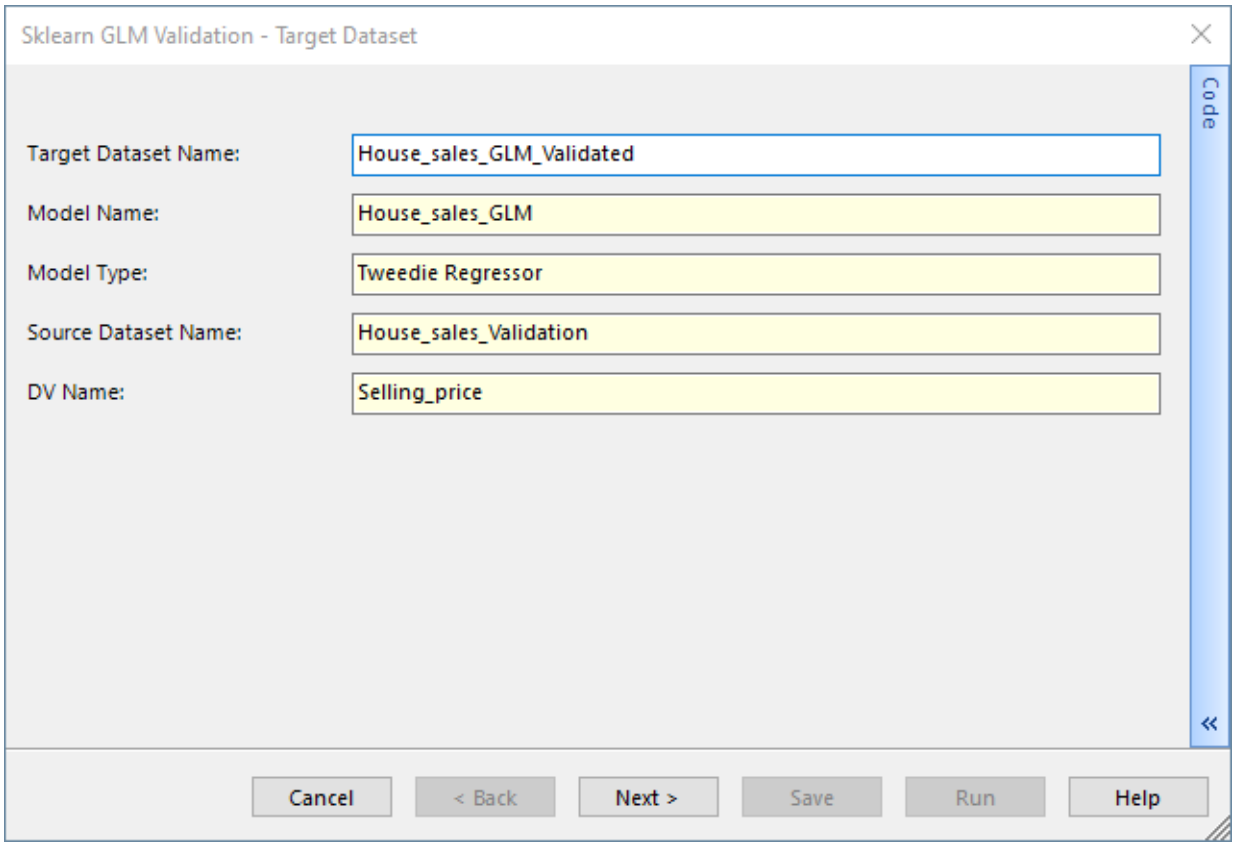
Double-click the **Model Validation** node or right-click on it and select **Modify**.



The Sklearn GLM Validation wizard opens. It displays the names of the input model and dataset, the regressor type, and the name of the dependent variable.

In the *Target Dataset Name* field, specify the name for the output dataset that will contain the validation information.

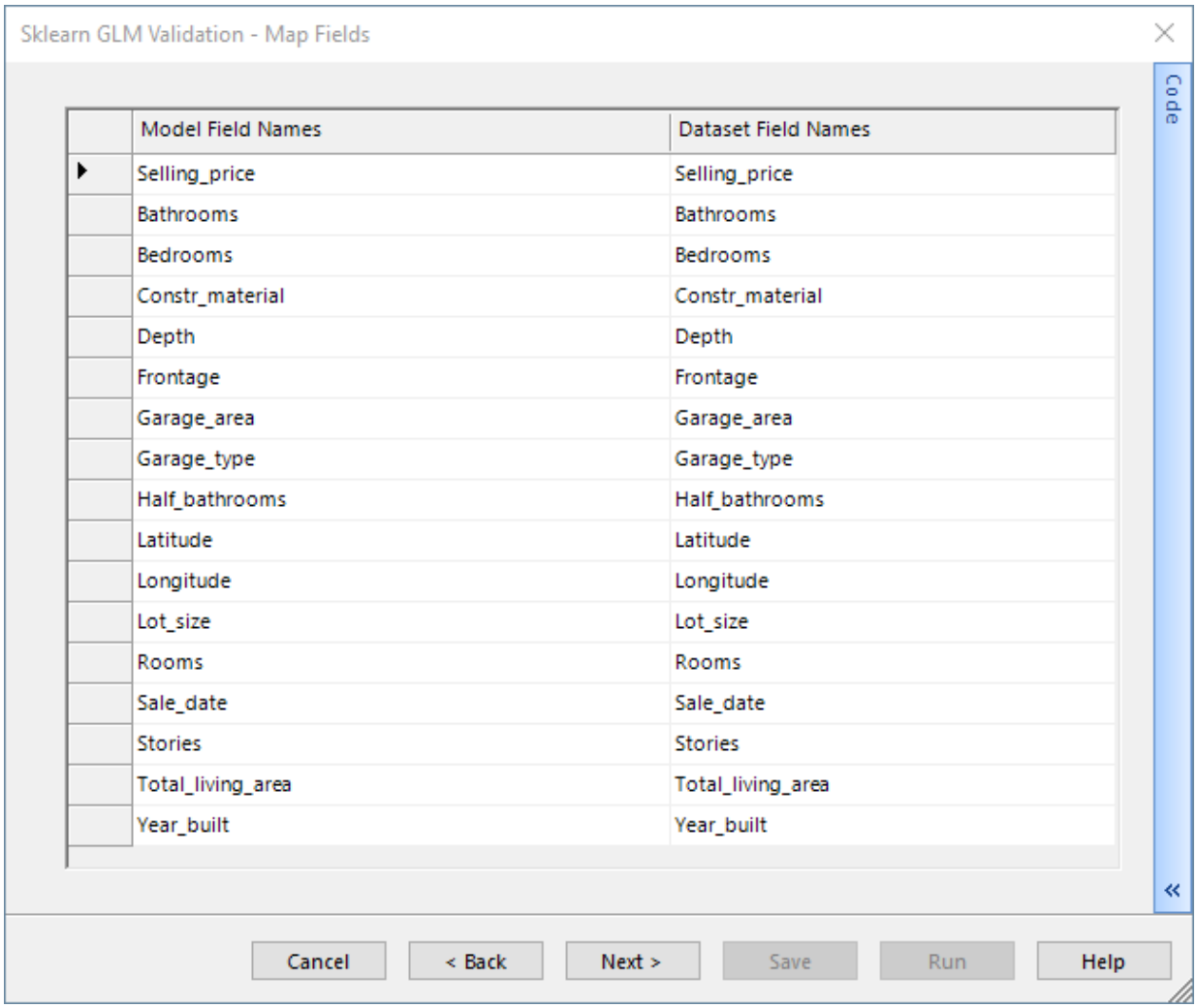
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

The next step provides the option of mapping the field names in the model to the ones in the validation input dataset. In some situations, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map fields by clicking in the right column (Dataset Field Names) and selecting the right fields to match the model variables in the left column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

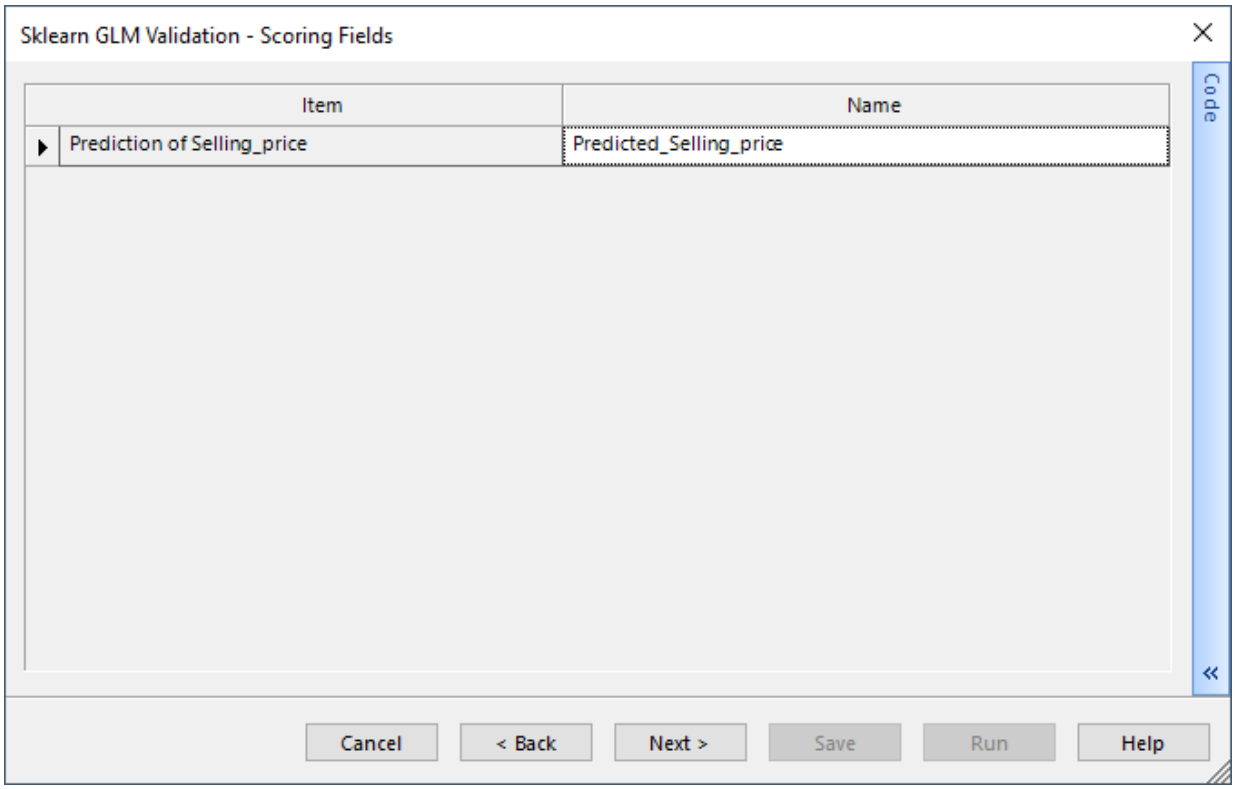


Note: When validating GLM models that use derived variables (also called calculated columns) with expressions, the input dataset must include these variables. This is different from the case of native Altair models with derived variables, which can be validated on a dataset with only *original* variables that the derived ones depend on, while the derived ones are not required.

Click **Next**.

In the **Scoring Fields** page, review the scoring fields to be produced by the model and modify their names if necessary.

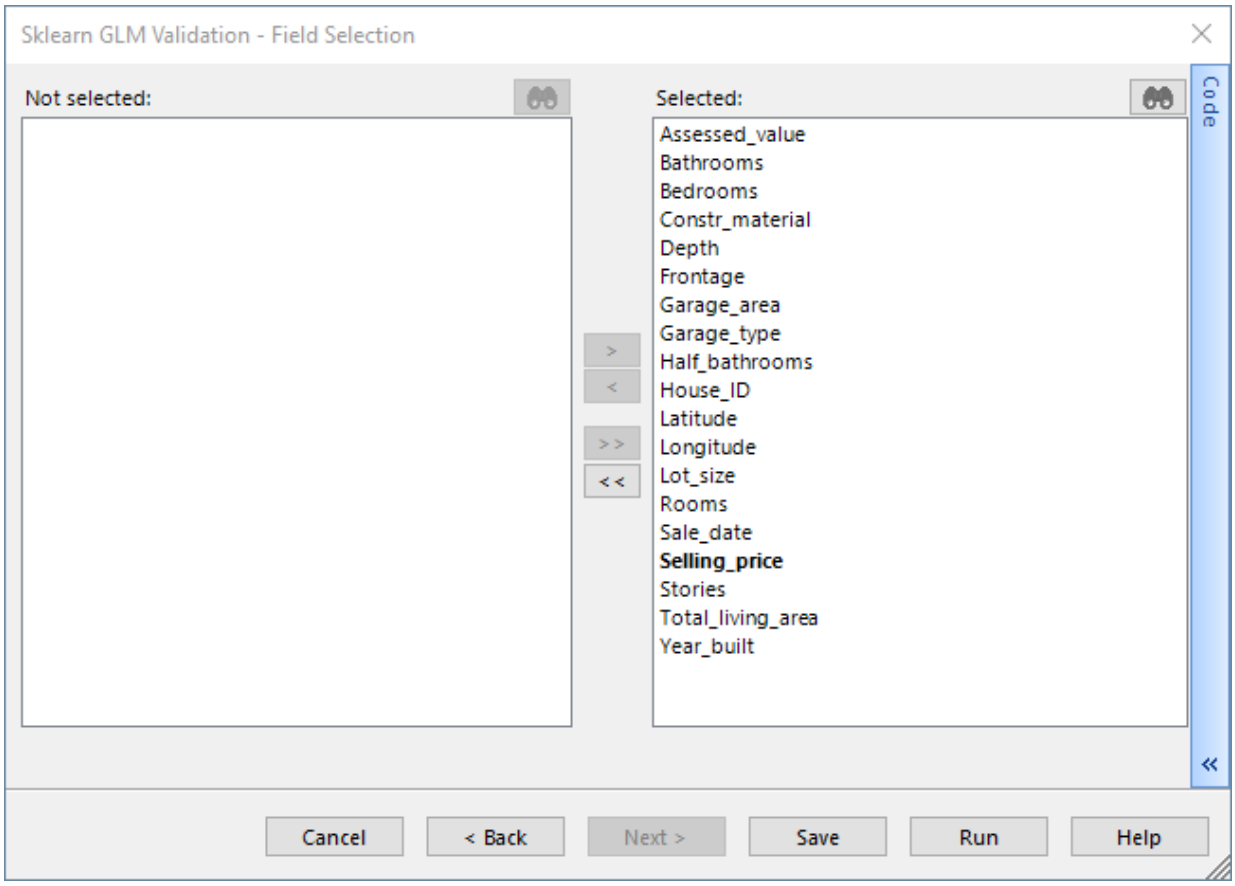
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

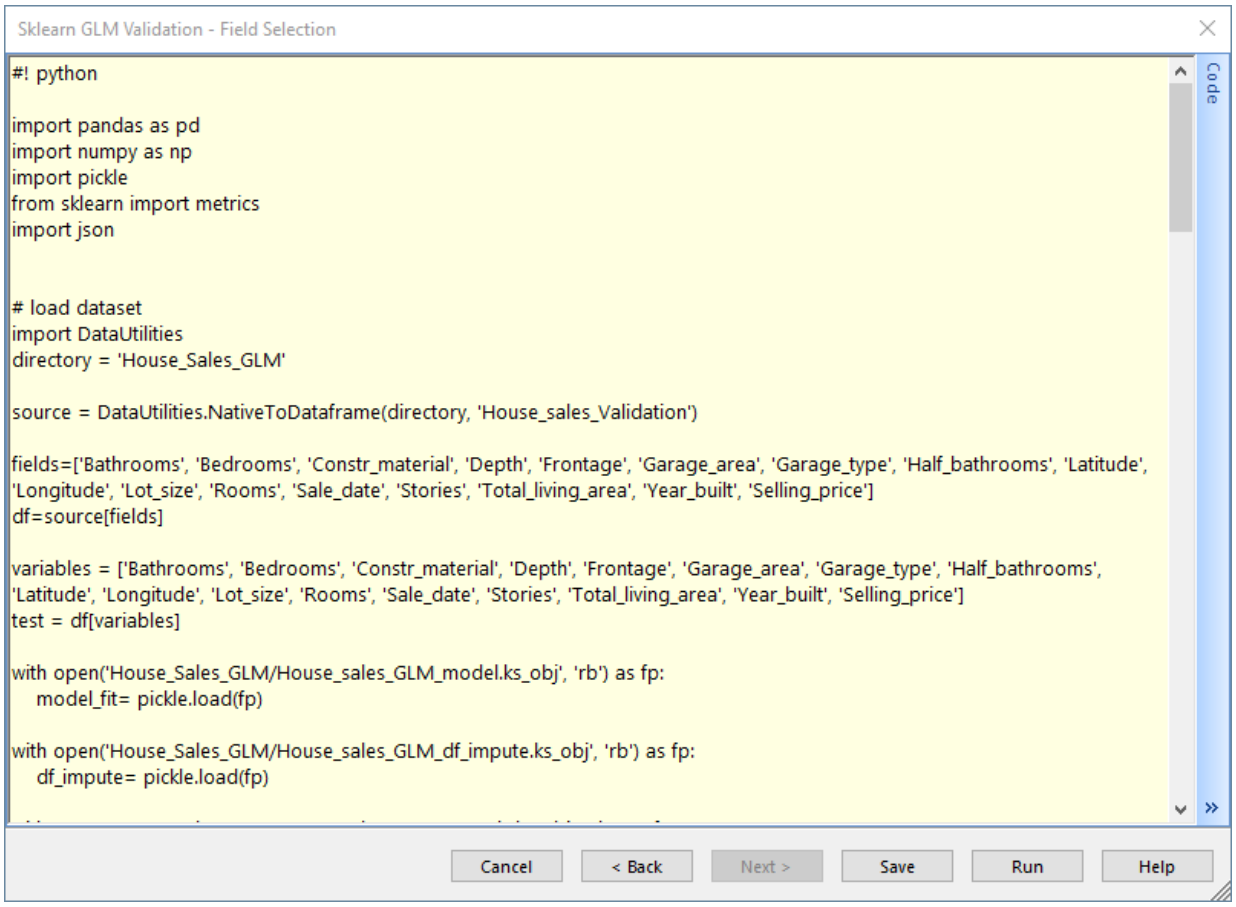
In the **Field Selection** page, choose the fields from the input dataset that you would like to include in the validation output in addition to the new score / probability fields. The fields required by the model are highlighted in bold font and cannot be excluded. By default, all fields in the validation input dataset are selected to be included in the output.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

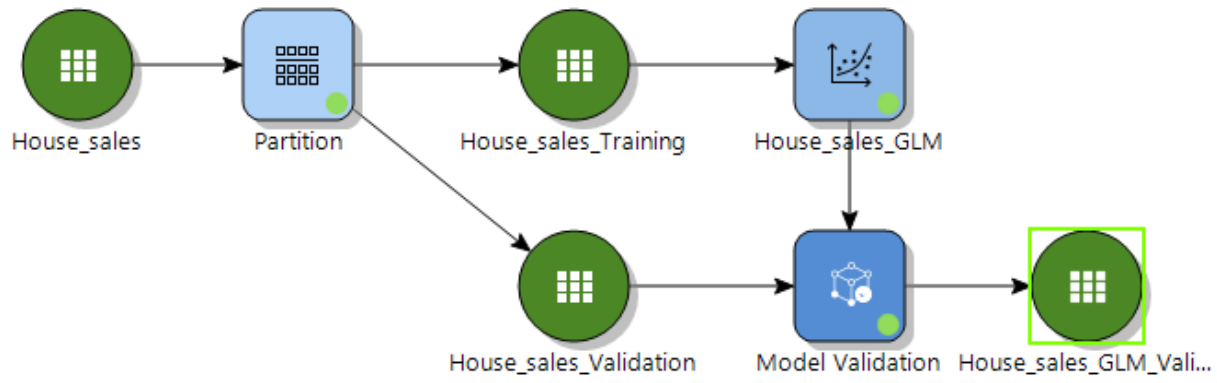


To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

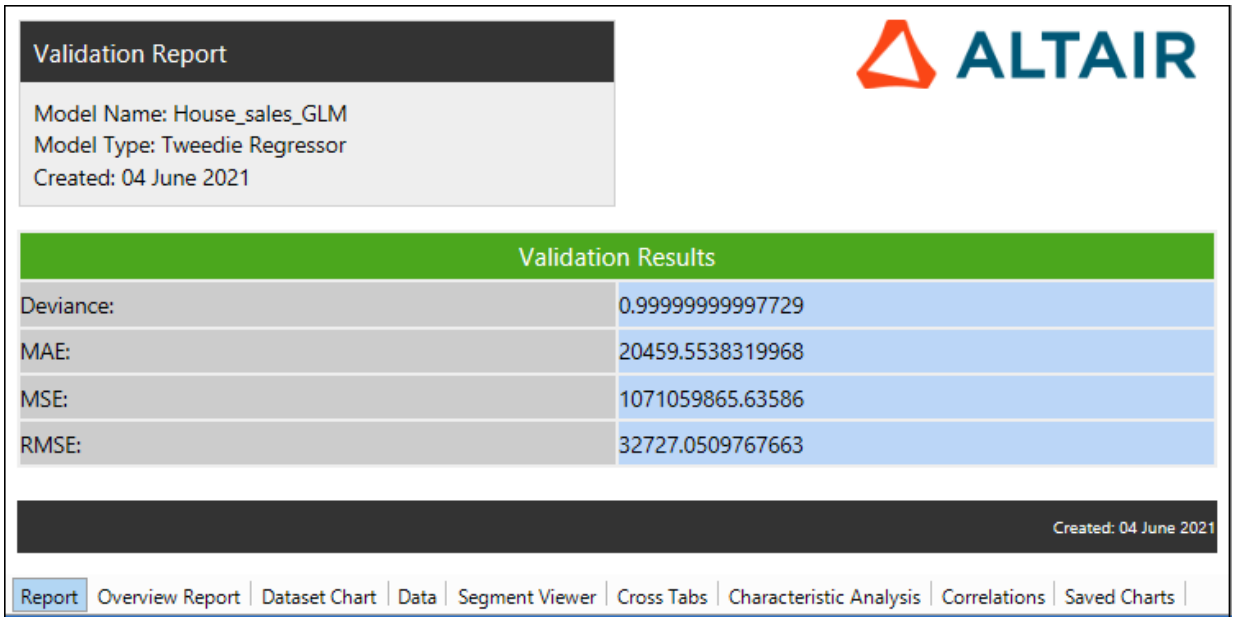


Click **Save** to save the settings, or click **Run** to save the settings and run the validation process immediately. The model is validated in a Python container session, and then the dataframe with generated probabilities / scores is converted back from Pandas to Altair KDD format. The output dataset node is created in the workflow. Double-click the output dataset to view the results.



The first tab of the output dataset is the Validation Report. It displays the values of the model evaluation metrics.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also:

- Generalized Linear Model

Sklearn XGB Validation

Validation of an Sklearn XGBoost model is performed using the **Model Validation** node available the **Evaluate** palette. For general information about validation, see the topic About Validation.

Prerequisites for Sklearn XGB model validation: An Sklearn XGB model node and a dataset with the dependent variable and independent variables required by the model.

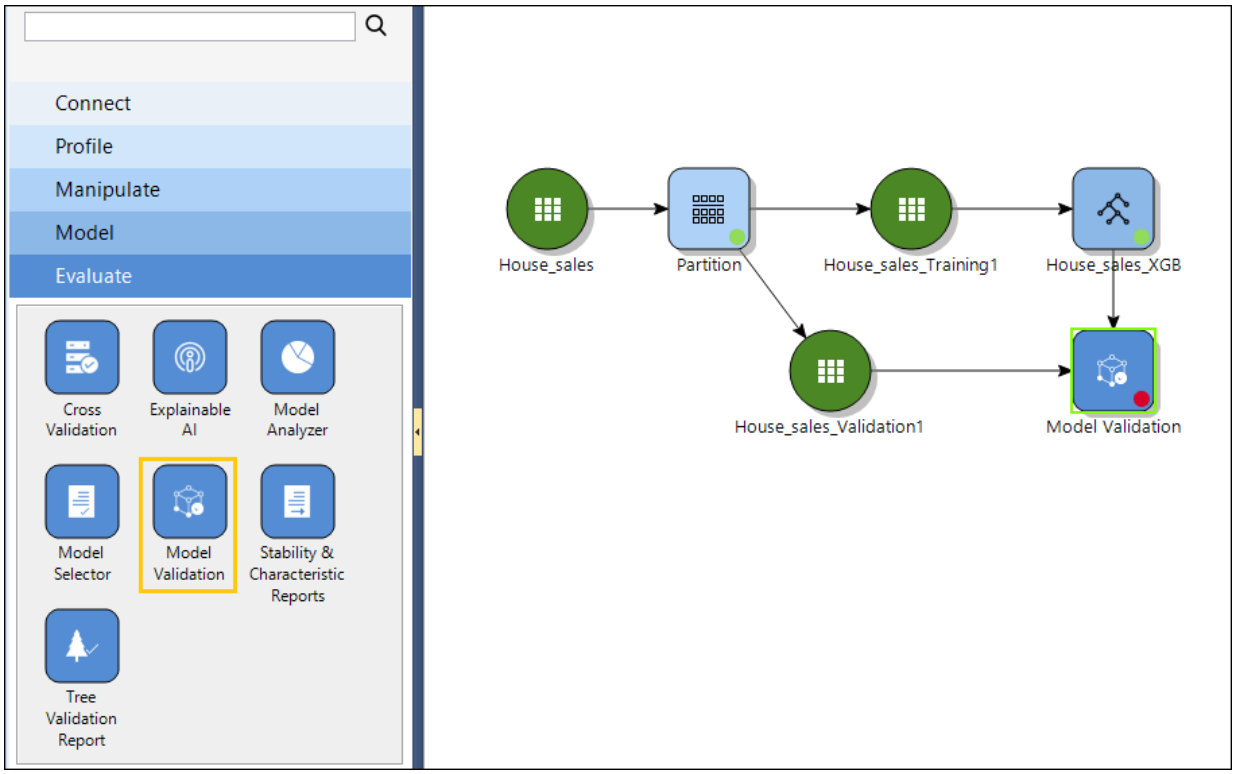
To add a Model Validation operation to your workflow:

- Drag the **Model Validation** node from the **Evaluate** palette onto the Workflow canvas and connect the inputs: an **Sklearn XGB** node and a dataset node that contains the data to validate.

  OR

- Right-click anywhere on the workflow canvas and select the context menu command **Evaluate --> Model Validation**. This will create a new Model Validation node. Connect the inputs: an **Sklearn XGB** model node and a dataset node that contains the data to validate.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

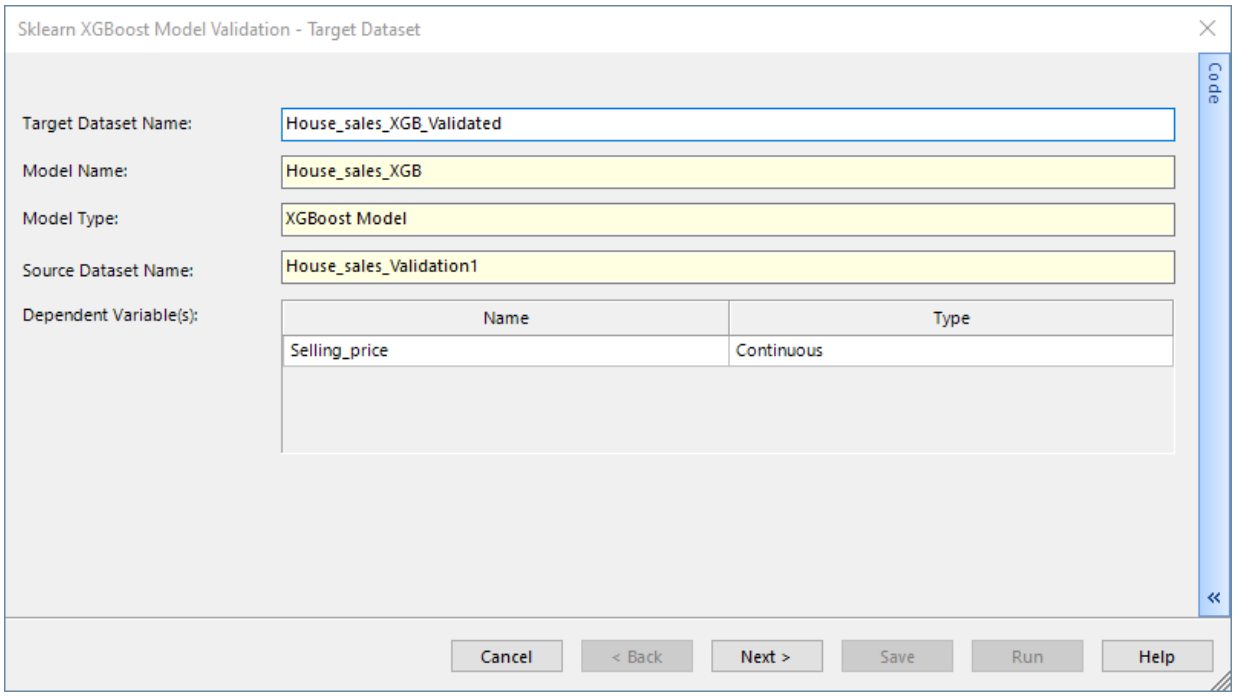


Double-click the **Model Validation** node or right-click on it and select **Modify** from the context menu.

The Sklearn XGB Validation wizard opens. It displays the names of the input model and dataset, the name and type of the dependent variable (*Continuous* or *Discrete*).

In the *Target Dataset Name* field, specify the name for the output dataset that will contain the validation data.

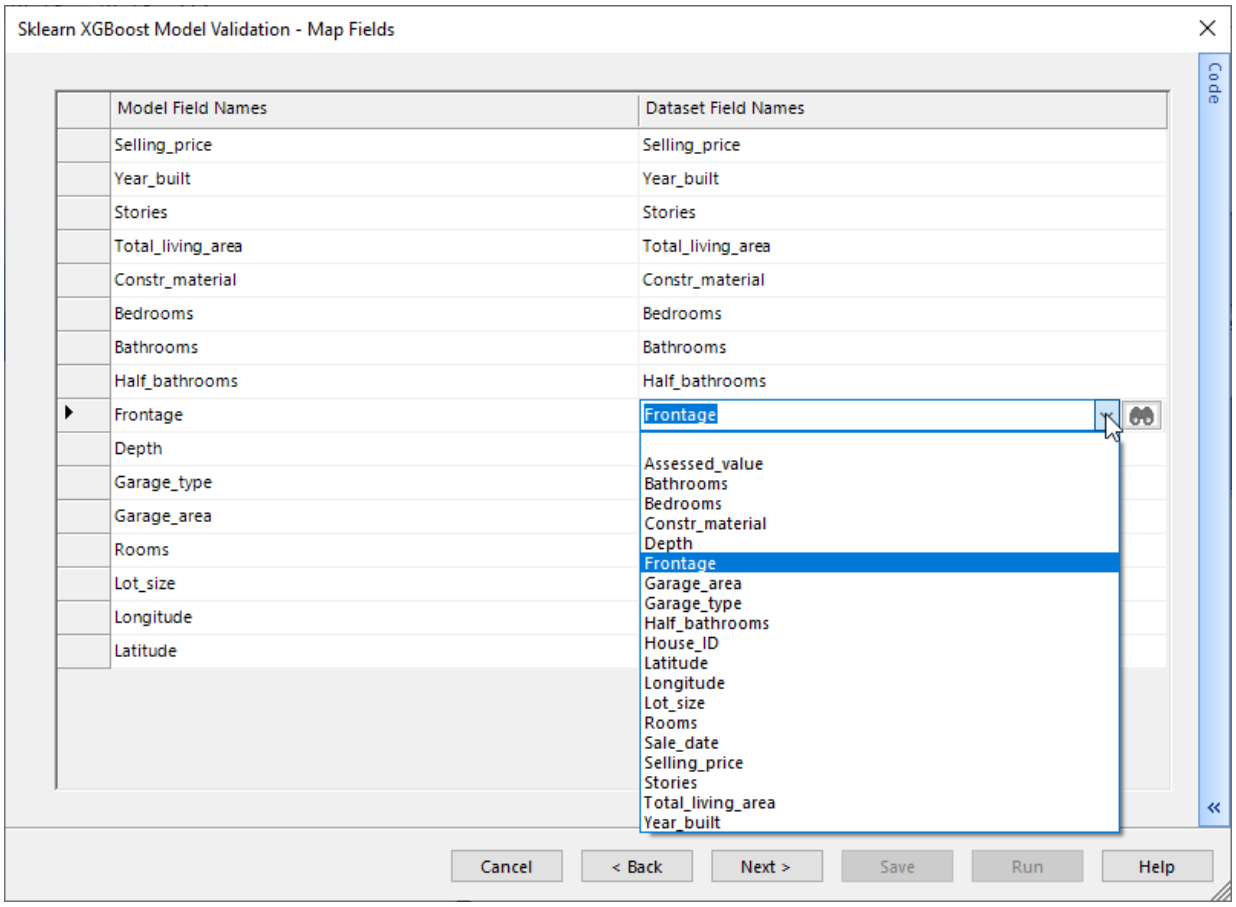
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

The next step provides the option of mapping the field names in the model to the ones in the validation input dataset. In some cases, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map fields by clicking in the right column (Dataset Field Names) and selecting the right fields to match the model variables in the left column:

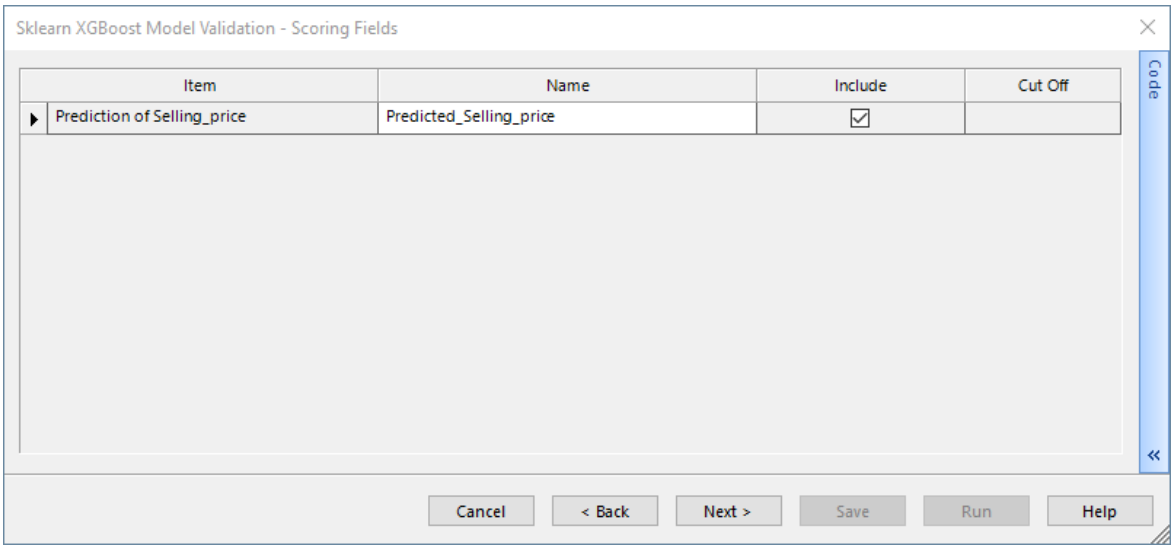
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

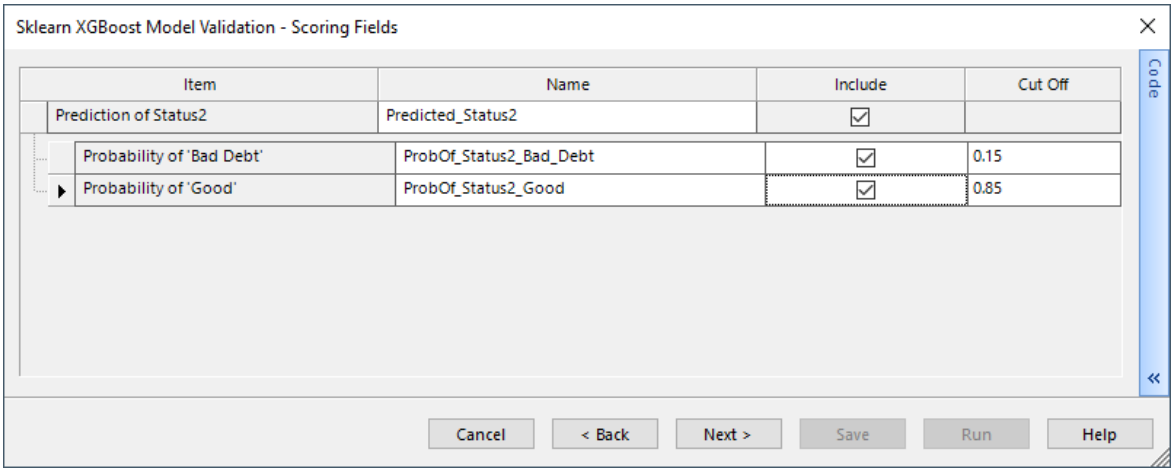
In the **Scoring Fields** page, review the scoring fields to be produced by the model and modify their names if necessary..

Example 1: Scoring fields for a model with a continuous dependent variable:



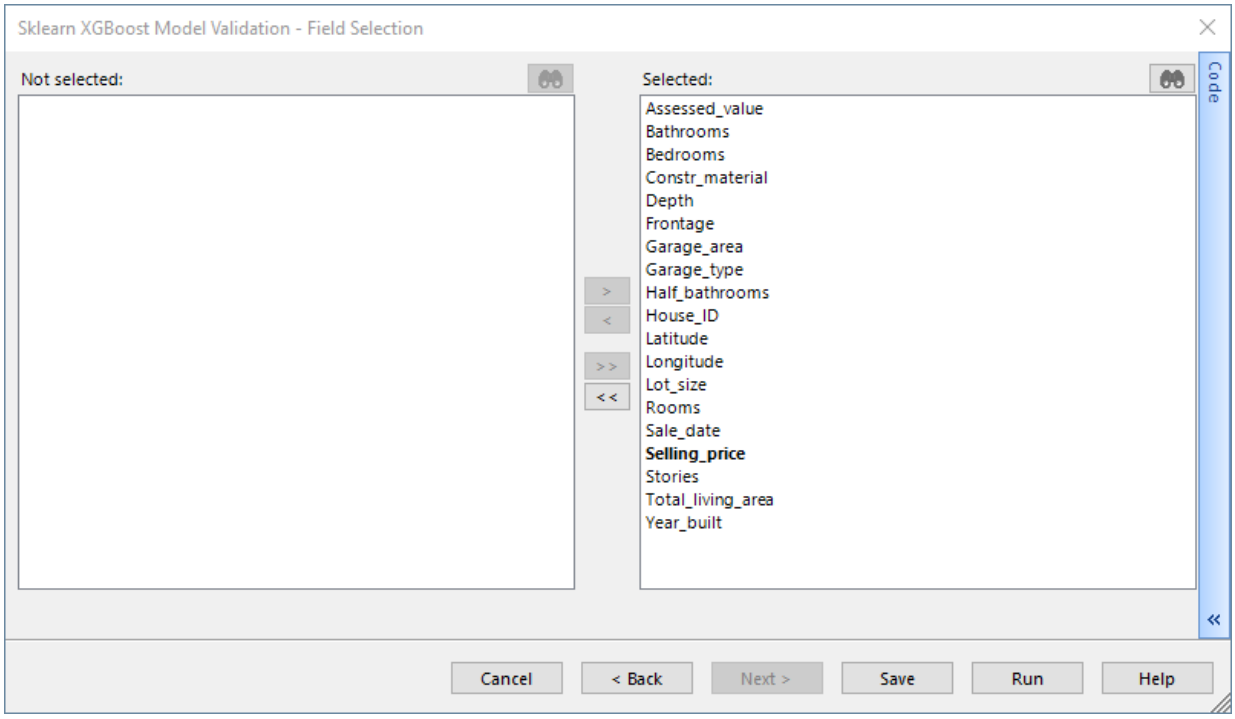
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Example 2: Scoring fields for a model with a discrete (categorical) dependent variable. If the dependent variable is binary, specify the cut-off value for the probability of the target outcome to classify the predictions.



Click **Next**.

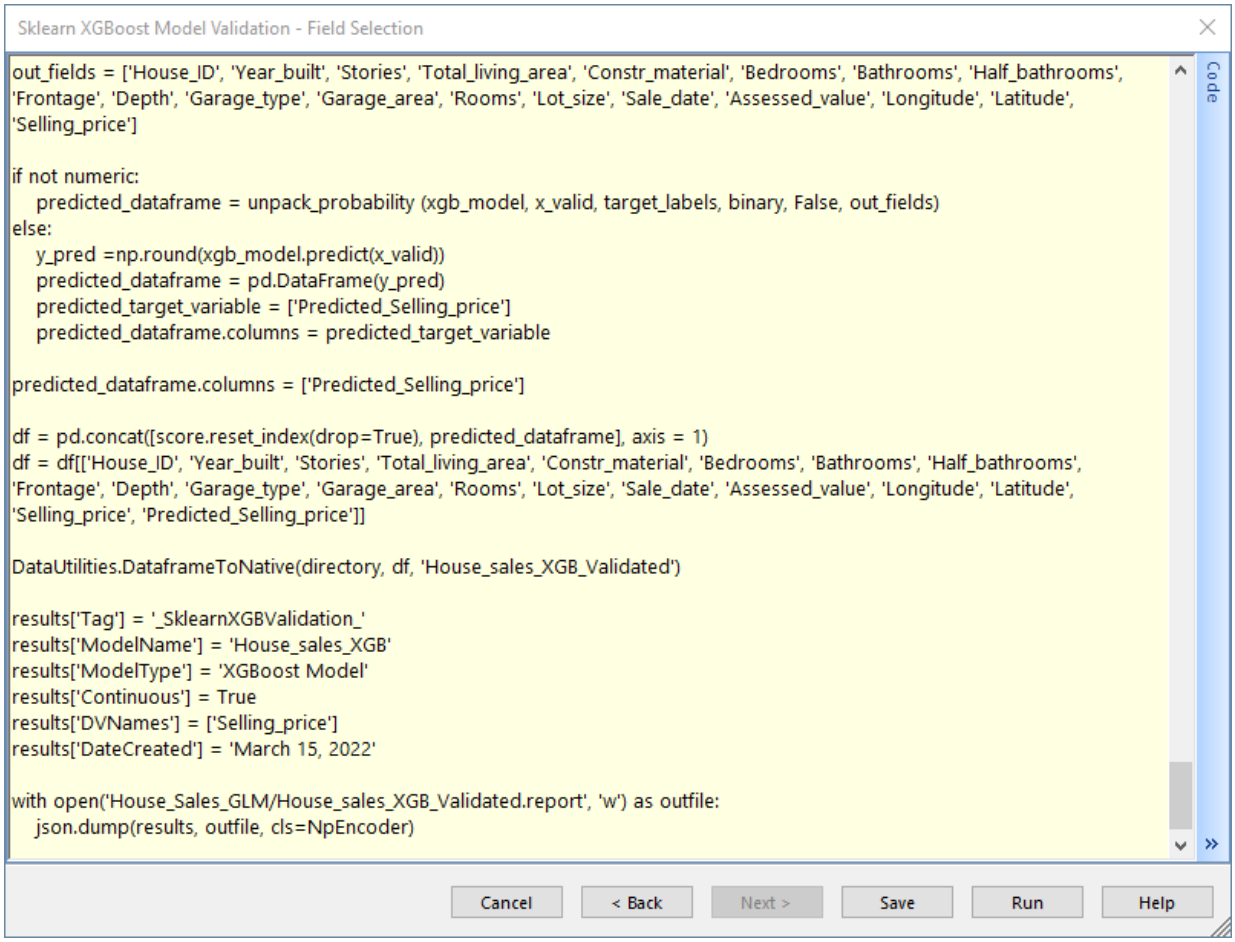
In the **Field Selection** page, choose the fields from the input dataset to include in the validation output in addition to the new scoring fields. The fields required by the model are highlighted in bold font and cannot be excluded. By default, all fields in the validation input dataset are selected to be included in the output.



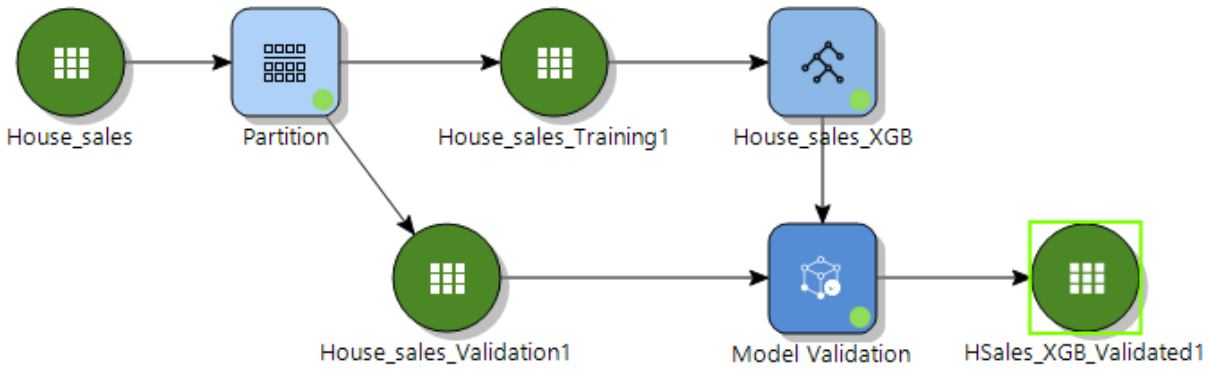
To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.



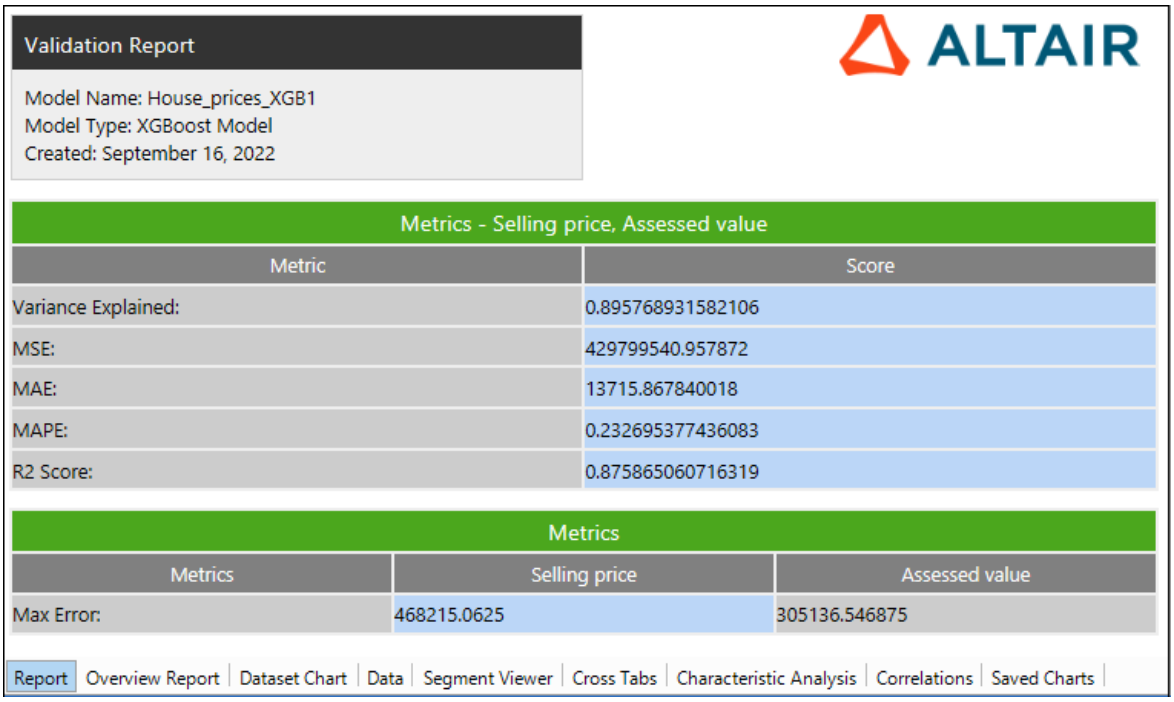
Click **Save** to save the settings, or click **Run** to save the settings and run the validation process immediately. The model is validated in a built-in Python environment, then the dataframe with generated probabilities / scores is converted back from Pandas to Altair KDD format. The output dataset node is created in the workflow. Double-click the output dataset to view the results.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The first tab of the output dataset is the Validation Report. It displays the values of the model evaluation metrics.

Example 1: Validation Report for a regression model with two continuous dependent variables.



Validation Report

Model Name: House_prices_XGB1
Model Type: XGBoost Model
Created: September 16, 2022

Metrics - Selling price, Assessed value

Metric	Score
Variance Explained:	0.895768931582106
MSE:	429799540.957872
MAE:	13715.867840018
MAPE:	0.232695377436083
R2 Score:	0.875865060716319

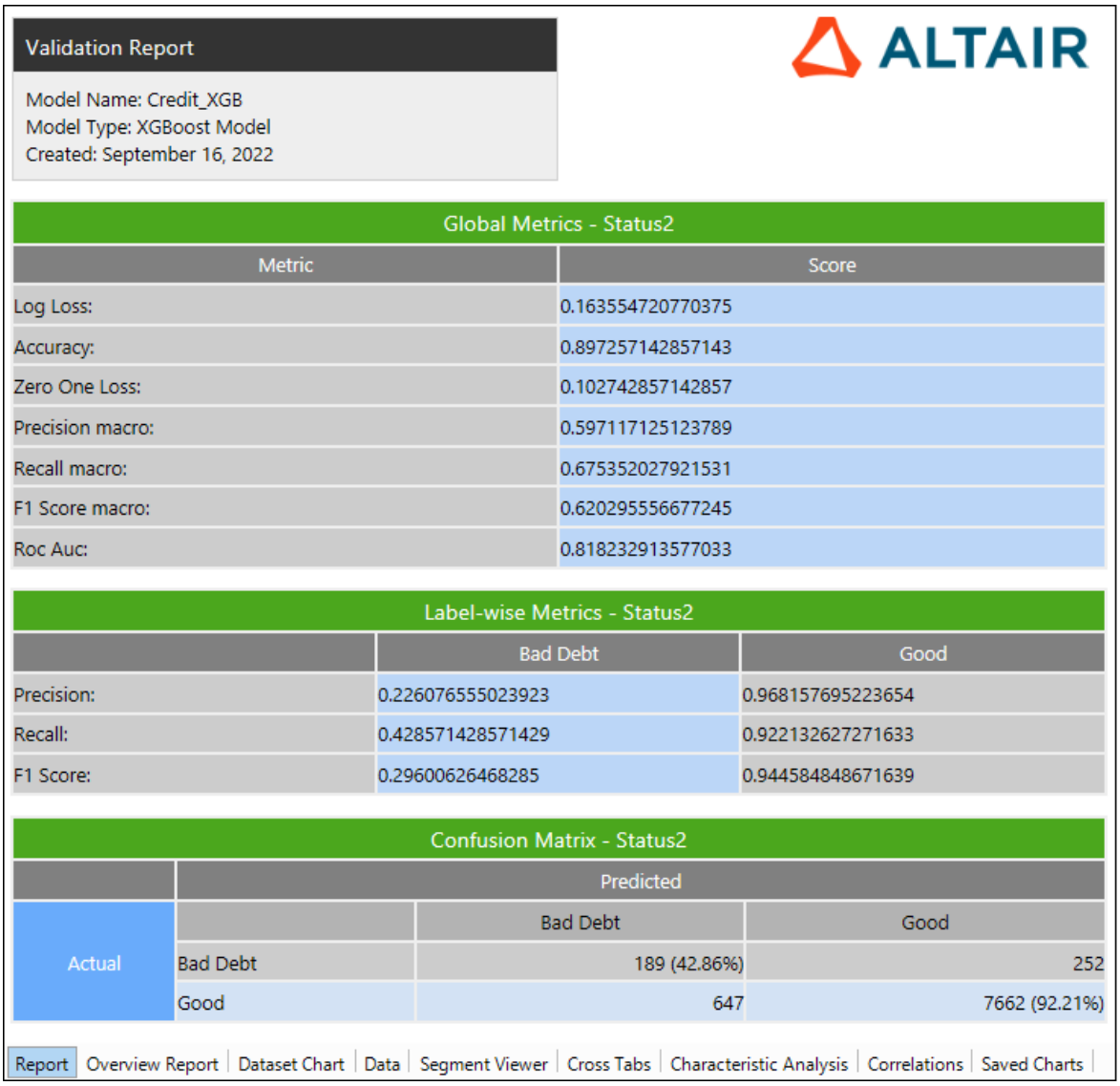
Metrics

Metrics	Selling price	Assessed value
Max Error:	468215.0625	305136.546875

Report | Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

Example 2: Validation Report for a classification model with a single discrete dependent variable.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Validation Report

Model Name: Credit_XGB
Model Type: XGBoost Model
Created: September 16, 2022

Global Metrics - Status2

Metric	Score
Log Loss:	0.163554720770375
Accuracy:	0.897257142857143
Zero One Loss:	0.102742857142857
Precision macro:	0.597117125123789
Recall macro:	0.675352027921531
F1 Score macro:	0.620295556677245
Roc Auc:	0.818232913577033

Label-wise Metrics - Status2

	Bad Debt	Good
Precision:	0.226076555023923	0.968157695223654
Recall:	0.428571428571429	0.922132627271633
F1 Score:	0.29600626468285	0.944584848671639

Confusion Matrix - Status2

		Predicted	
		Bad Debt	Good
Actual	Bad Debt	189 (42.86%)	252
	Good	647	7662 (92.21%)

Report | Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

The output dataset can be connected to a Model Analyzer node to further analyze the model performance.

See also: Sklearn XGBoost Model, Model Analyzer.

The top section of the Validation Report displays the name of the input dataset and the input model.

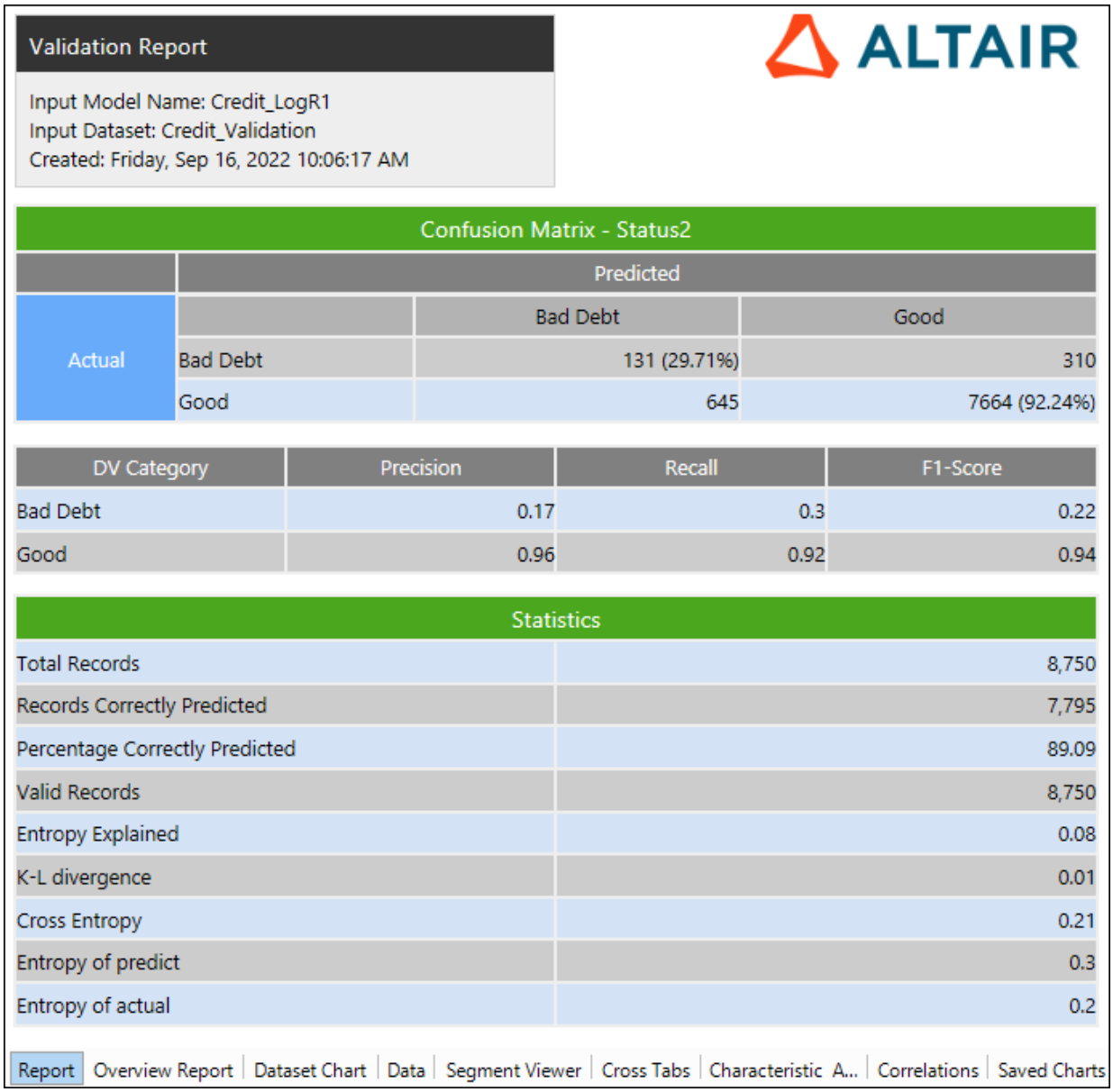
1. Validation Report for models with discrete dependent variable

The report includes the following metrics characterizing the model relevance, accuracly, and performance:

- *Confusion Matrix*

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- The table showing *precision*, *recall*, and *F1-score* (the harmonic mean of the precision and recall).
- The *Statistics* table with prediction accuracy metrics such as *Kullback-Leibler Divergence, Cross Entropy*, and other applicable metrics.



Validation Report

Input Model Name: Credit_LogR1
Input Dataset: Credit_Validation
Created: Friday, Sep 16, 2022 10:06:17 AM

ALTAIR

Confusion Matrix - Status2

		Predicted	
		Bad Debt	Good
Actual	Bad Debt	131 (29.71%)	310
	Good	645	7664 (92.24%)

DV Category	Precision	Recall	F1-Score
Bad Debt	0.17	0.3	0.22
Good	0.96	0.92	0.94

Statistics	
Total Records	8,750
Records Correctly Predicted	7,795
Percentage Correctly Predicted	89.09
Valid Records	8,750
Entropy Explained	0.08
K-L divergence	0.01
Cross Entropy	0.21
Entropy of predict	0.3
Entropy of actual	0.2

Report | Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic A... | Correlations | Saved Charts

The **Confusion Matrix** shows how many of the predictions produced by model validation were correct and how many were wrong, i.e., did and did not coincide with actual DV values.

In the Validation Report example shown above, the validated model has the dependent variable *Status2* with values 'Bad Debt' and 'OK'. The Confusion Matrix shows that the actual 'Bad Debt' value was correctly predicted in 127 cases (28.8% of all Bad Debt values) and wrongly predicted as 'OK' in 314 cases. The 'OK' value was correctly predicted in 7561 cases (91% of all OK values) and wrongly predicted as 'Bad Debt' in 748 cases.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Note that for models with binary dependent variable, all the numeric results in the report depend on the ***cut-off*** value specified in the Validation Fields page of the Validation Wizard, since the predictions, i.e., the way the probabilities are interpreted, will be different depending on the cut-off. If we re-validate the model with the reduced 'Bad Debt' probability, the results will be slightly better for 'Bad Debt' prediction accuracy. See the description of the Validation Fields page in the Validation Wizard for details on the cut-off value.

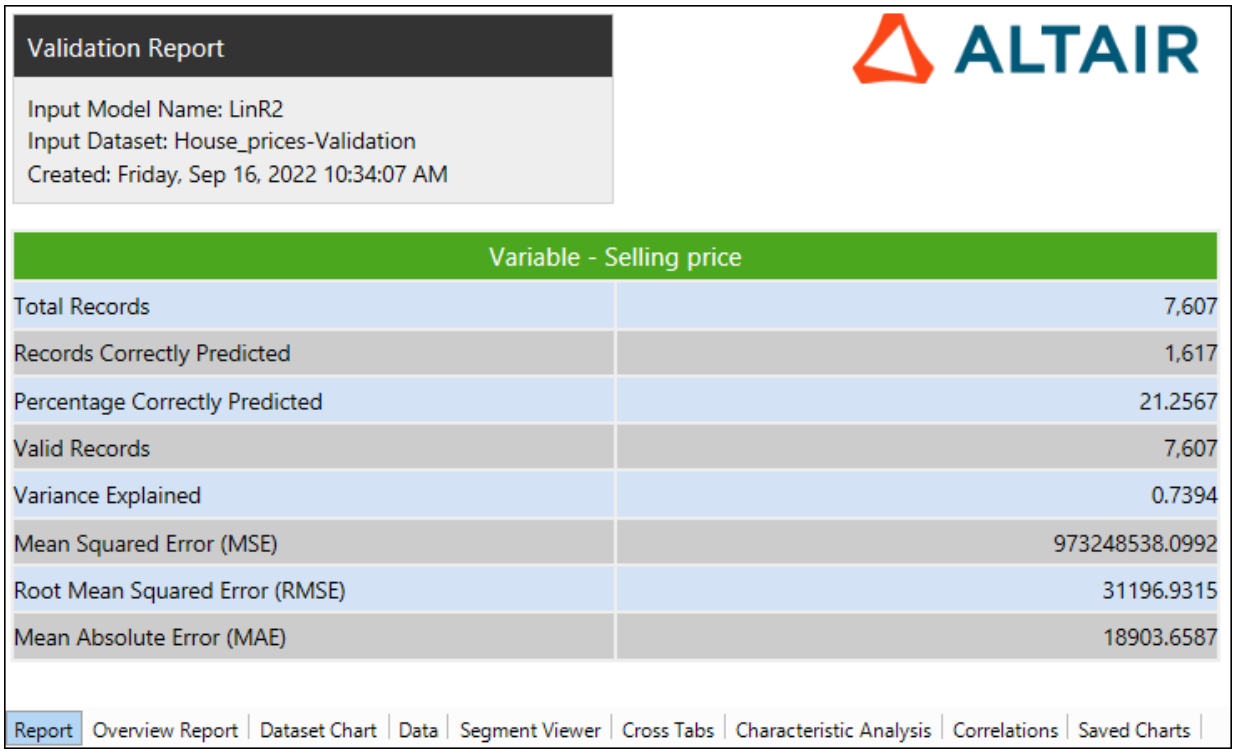
The **Statistics** table provides the following information:

- ***Total Records***: The total number of input records.
- ***Records Correctly Predicted***: The number of records with correctly predicted target outcome. Note: This depends on the cut-off value specified in the Model Validation wizard.
- ***Percentage Correctly Predicted***: The Percentage of records with correctly predicted target outcome.
- ***Valid Records***: This is the number of records for which the probabilities could be successfully calculated and therefore predictions were produced. Invalid records are the records in the input dataset for which the probabilities could not be calculated because the values in some fields were unknown to the model.
- ***Entropy Explained:*** The same as *pseudo R-squared*.
- ***K-L Divergence***: The Kullback-Leibler (K-L) divergence, or relative entropy, is a quantity which measures the difference between two probability distributions.
- ***Cross Entropy***: Cross entropy between two probability distributions measures the overall difference between the two distributions. This measures how well a distribution approximates another distribution.
- ***Entropy of predict:*** For the dependent variable values predicted by the model.
- ***Entropy of actual:*** For the actual values of the dependent variable.

2. Validation Report for models with continuous dependent variable

The report displays the name of the dependent variable and the *Statistics* table with the metrics characterizing the prediction accuracy of the model.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Validation Report

Input Model Name: LinR2
Input Dataset: House_prices-Validation
Created: Friday, Sep 16, 2022 10:34:07 AM

Variable - Selling price	
Total Records	7,607
Records Correctly Predicted	1,617
Percentage Correctly Predicted	21.2567
Valid Records	7,607
Variance Explained	0.7394
Mean Squared Error (MSE)	973248538.0992
Root Mean Squared Error (RMSE)	31196.9315
Mean Absolute Error (MAE)	18903.6587

The **Statistics** table provides the following information:

- ***Total Records***: The total number of input records.
- ***Records Correctly Predicted***: The number of records with correctly predicted target outcome. Note: This depends on the *Hit Range* value specified in the Model Validation wizard.
- ***Percentage Correctly Predicted***: The Percentage of records with correctly predicted target outcome.
- ***Valid Records***: The number of records for which the probabilities could be successfully calculated, and therefore predictions were produced. Invalid records are the records in the input dataset for which the probabilities could not be calculated because the values in some fields were unknown to the model.
- ***Variance Explained:*** The same as the c*oefficient of determination* or *R-Squared*
- ***Mean Squared Error* (*MSE*)**
- ***Root Mean Squared Error* (*RMSE*)**
- ***Mean Absolute Error* (*MAE*)**

See also: Scoring

Cross Validation

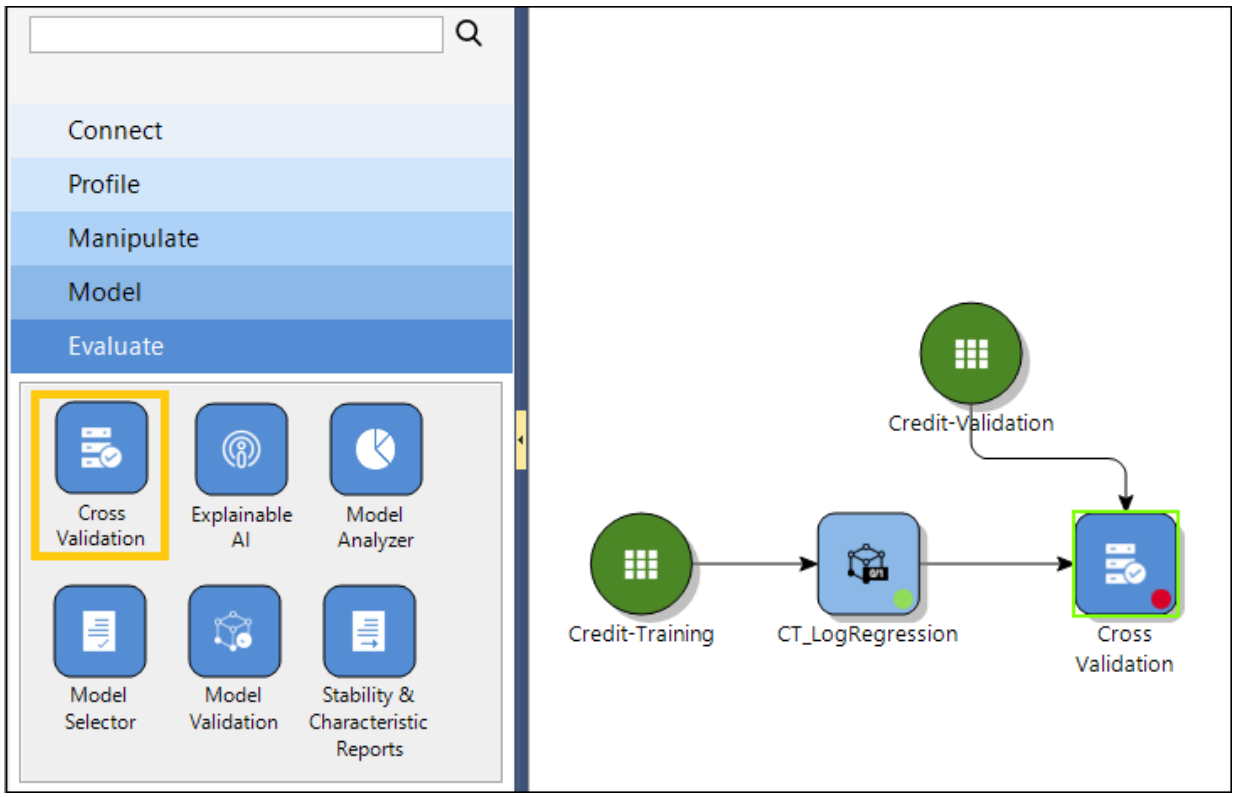
The **Cross Validation** node performs a function similar to the usual Model Validation. It is commonly used as a step in PLS Regression. As an input, in addition to a model and a dataset, cross validation requires a number k for k-fold validation.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The data is automatically separated into k folds. For each fold, the model is trained on the other k-1 folds, and then scored on the *k*th fold. The validation includes the computation of a likelihood function, depending on the model.

The output report includes a list of the maximized values of the likelihood function and the average, max, and min of the likelihood function over all folds.

To add a Cross Validation operation to your workflow, drag the **Cross Validation** node from the **Evaluate** palette onto the canvas and connect the required input model and dataset to the node. If you connect a Decision Tree node, a copy of the tree model (model instance) will be created automatically.

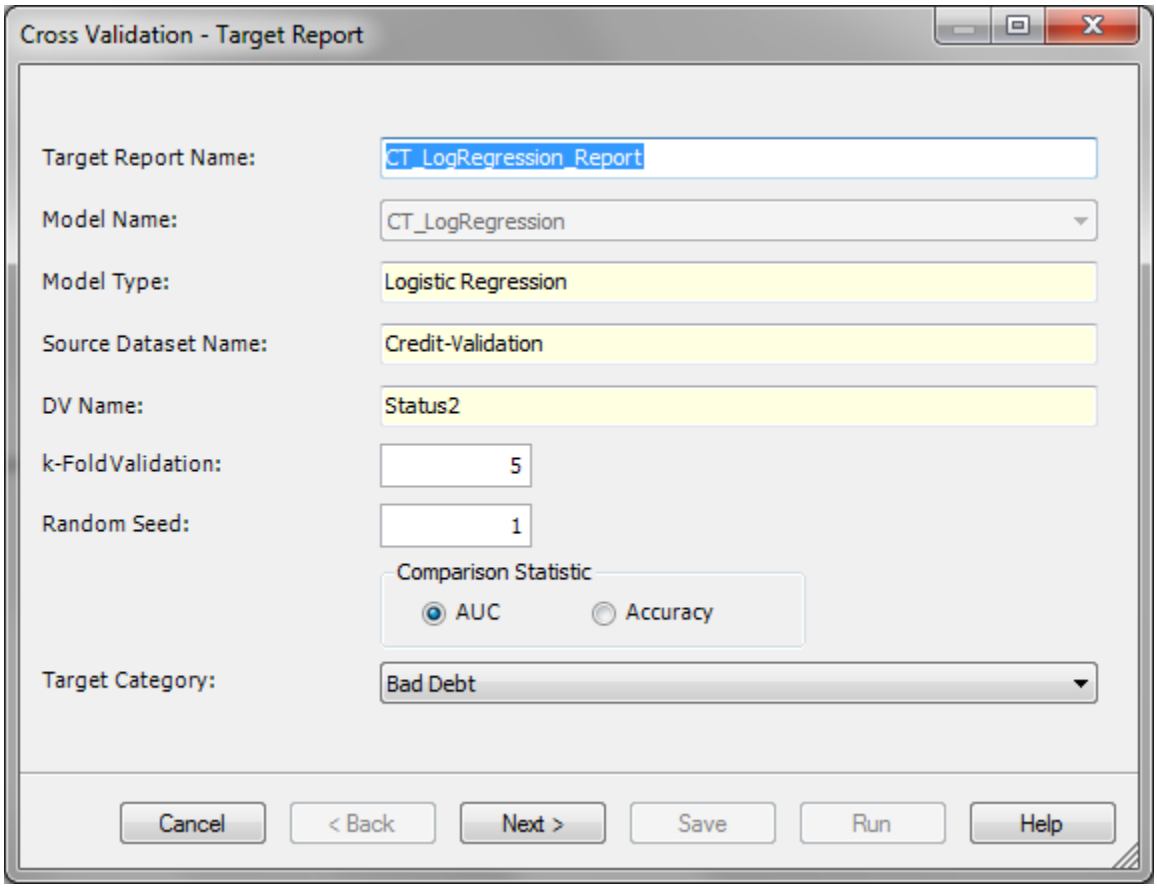


Double-click the Cross Validation node or right-click on it and select **Modify**.

In the Cross Validation wizard:

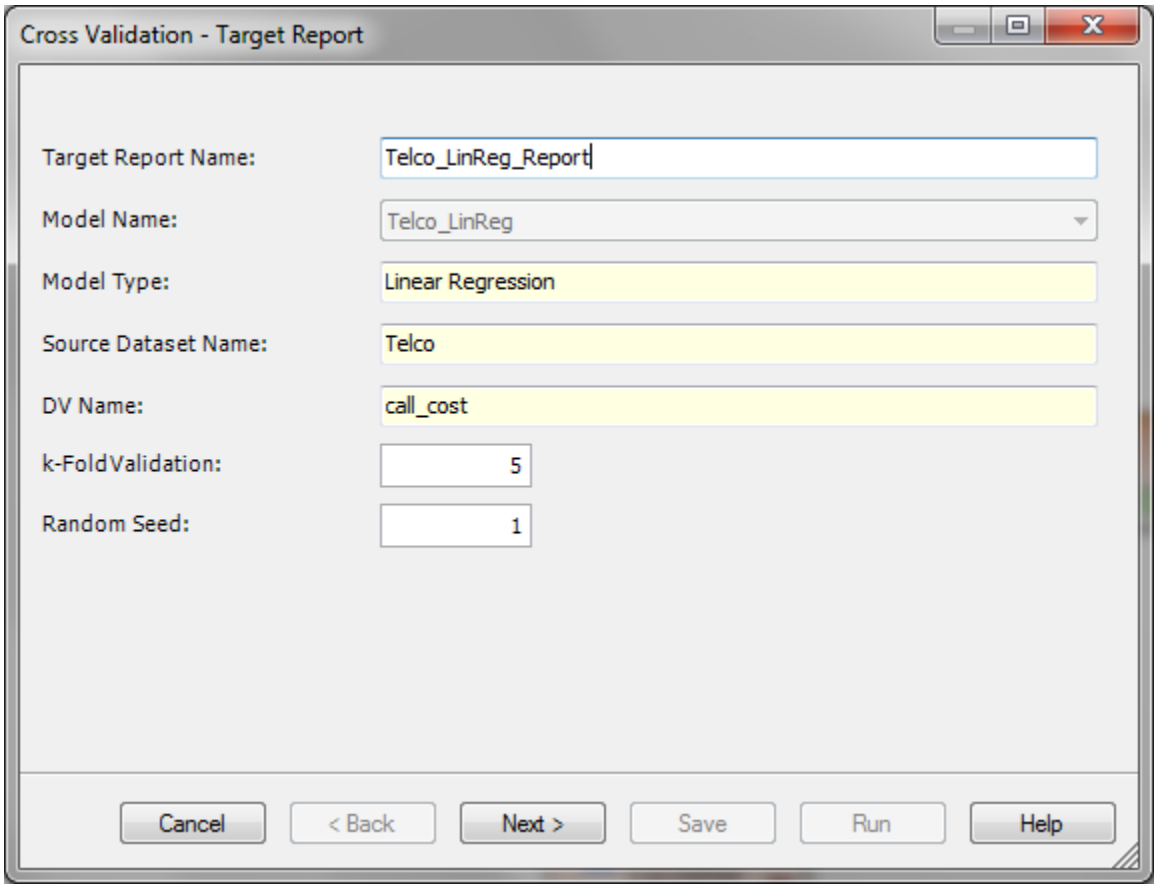
- Enter the target report name
- Specify the parameters: the number k for k-Fold Validation (the default is 5), Random Seed for repeatability.
- If the input model has a discrete dependent variable, also select the comparison statistics (AUC or Accuracy) and the target category of the dependent variable.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



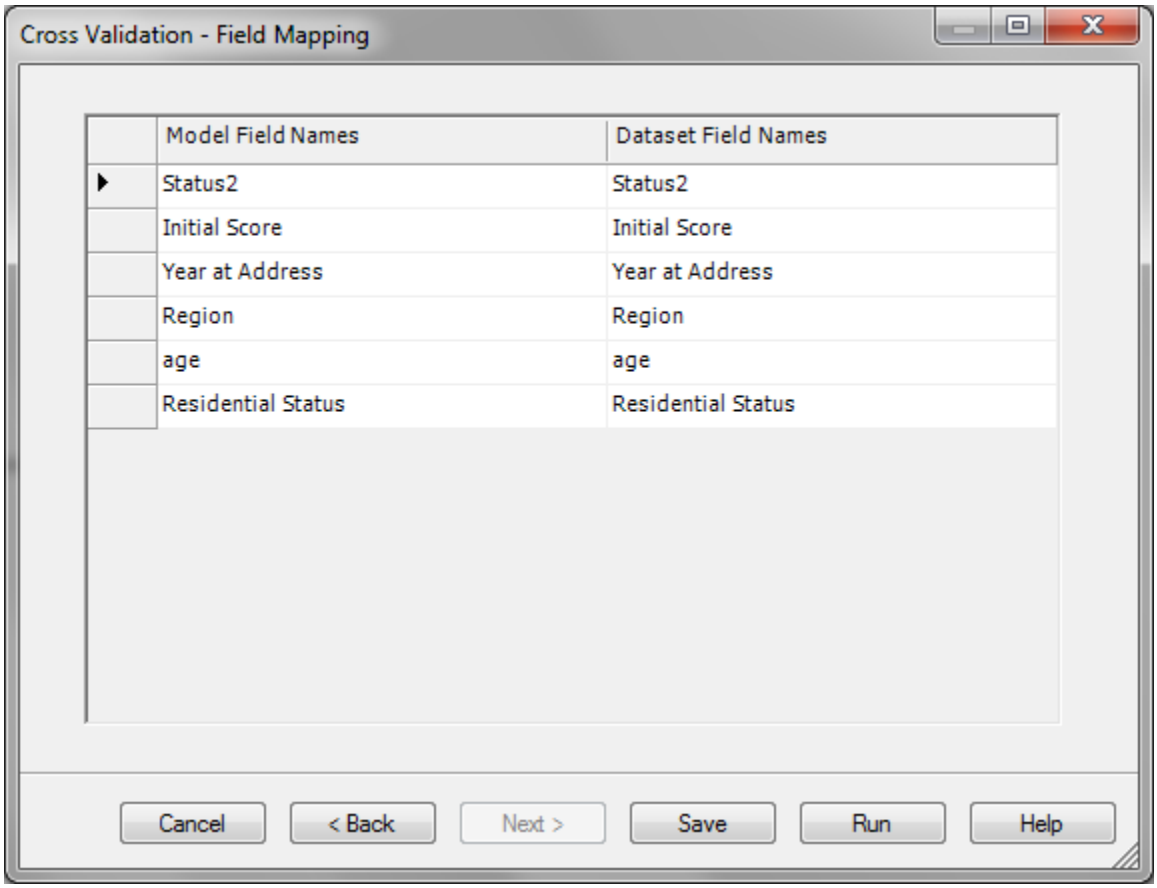
If the input model has a continuous dependent variable, specify the same parameters except the comparison statistics. In this case the statistic is MSE (Mean Squared Error).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If necessary, map the fields at the next page:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.

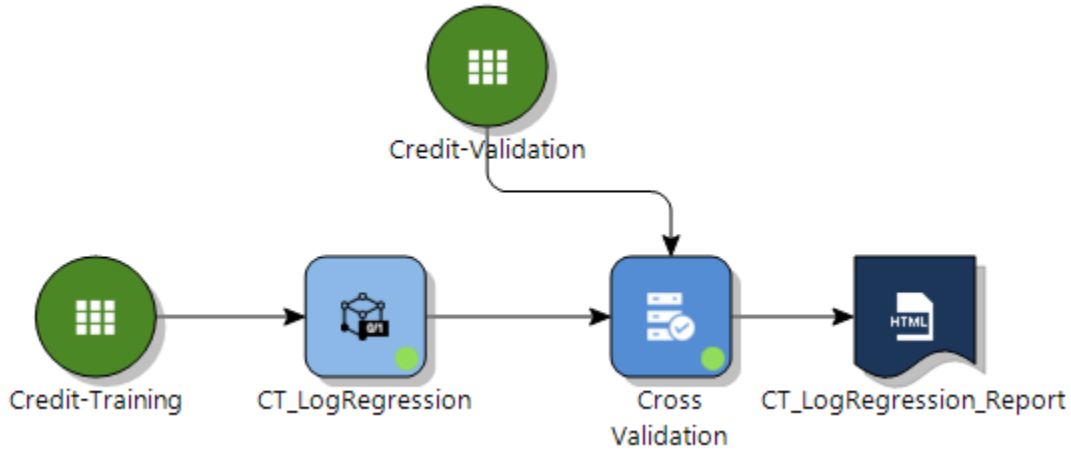
If the input is any model except decision tree or PLS, the node performs the following:

- 1. Create a random partition of the data into k "folds".
- 2. For each fold: leave that fold out, and re-train the model on the set of data which is the collection of the other k-1 folds.
- 3. Score the data from the final fold using the newly trained model.
- 4. Generate a score (from a comparison statistic) on the final fold. The score will depend on the model connected, and on the user's choice in option 3 above. If the DV in the connected model is categorical, the user had the choice of AUC or Accuracy. If the DV is continuous, the comparison statistic is Mean Squared Error (MSE).
- 5. Report the scores for each fold, as well as the average, max and min.

If a PLS model is connected: Since PLS "model" actually contains M different models, where M is the number of factors in the model, the node does the above for all M models.

If a decision tree, strategy tree or xml/pmml model is connected, the behaviour is the same as case 1, except that there is no retraining on the other k-1 folds. The trained model is just scored on the left out (kth) fold.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



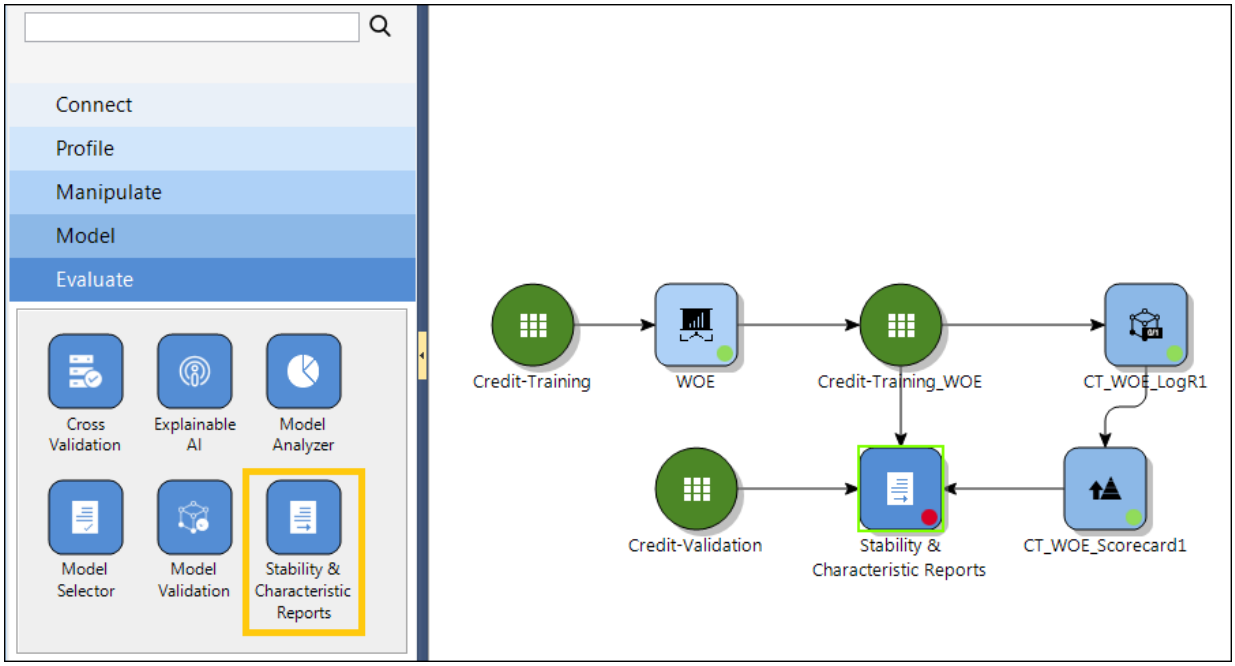
Double-click the output report node.

Stability & Characteristic Reports

The **Stability & Characteristic Reports** node in the **Evaluate** palette is used for model performance monitoring especially useful for Scorecards. The reports include Characteristic Analysis Report (expected vs actual), Population Stability Report, and Model Performance Statistics for score deciles.

To add a Stability & Characteristic Reports node to your workflow:

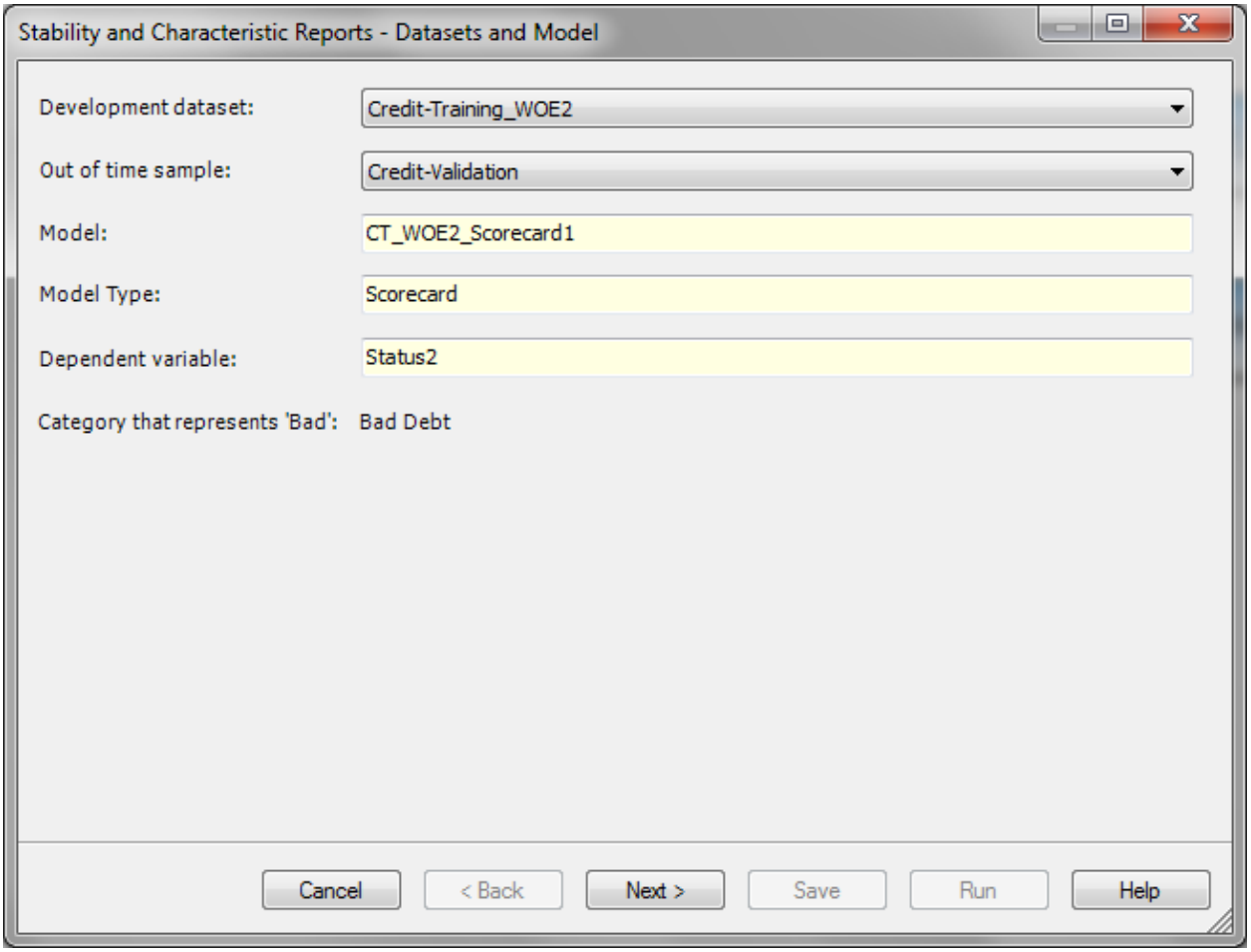
- Drag the **Stability & Characteristic Reports** node from the **Evaluate** palette onto the canvas and connect one model and two datasets (development and validation). The model must have a binary DV or be a Scorecard.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Double-click the Stability & Characteristic Reports node or right-click on it and select **Modify**.

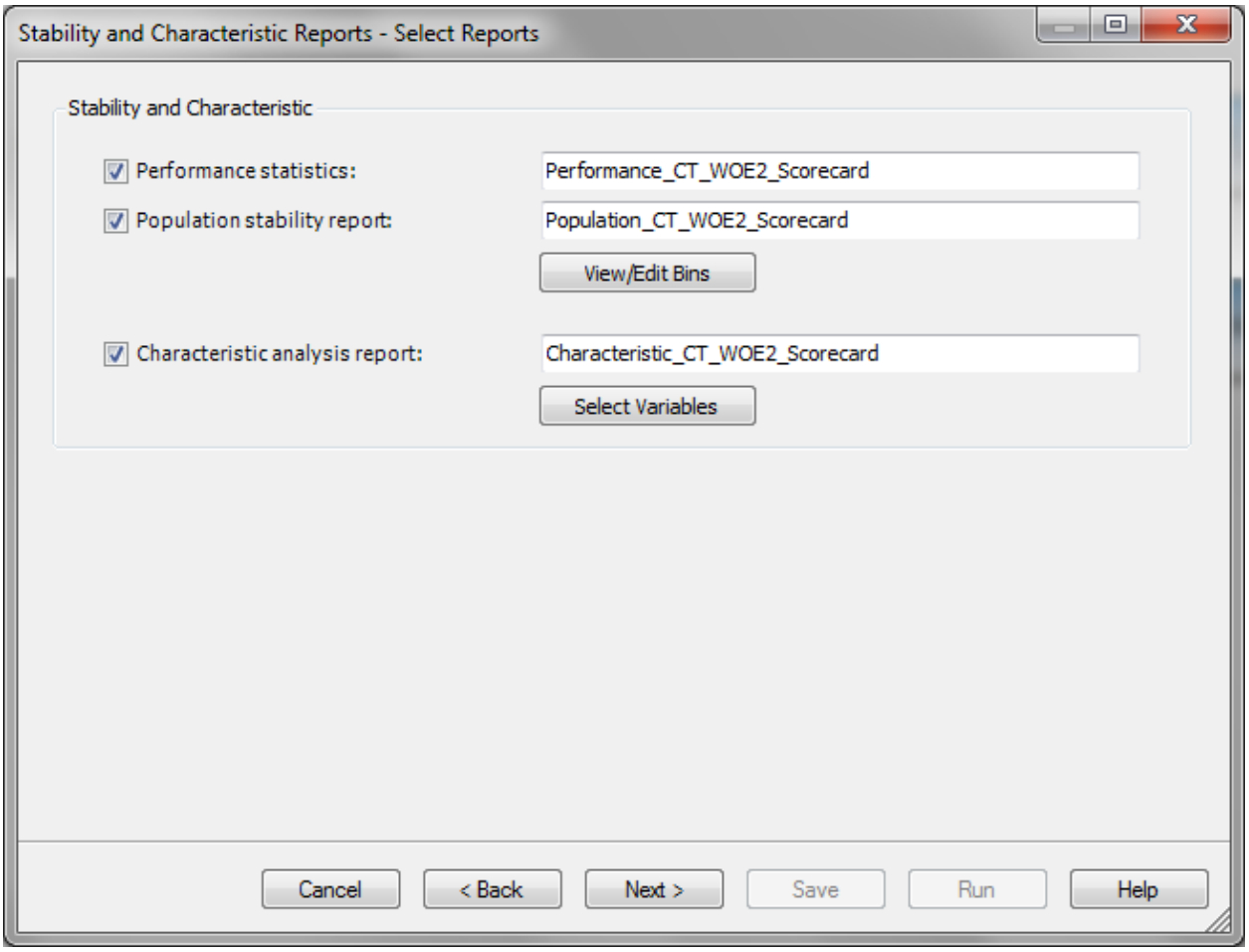
In the Stability & Characteristic Reports wizard: select the datasets for the appropriate roles - Development and Out of time samples.



Choose the required reports and specify their names:

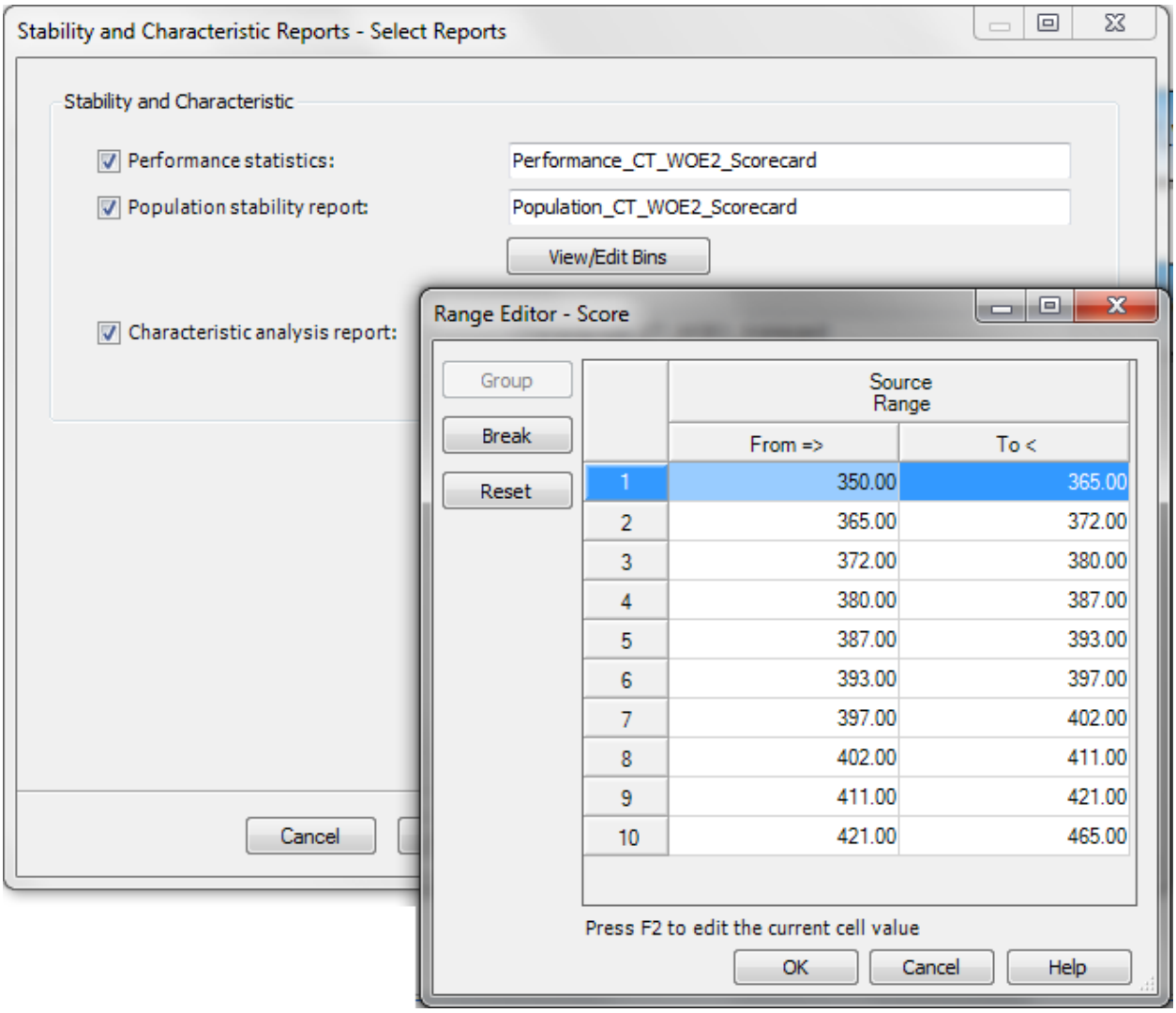
- Performance statistics report
- Population stability report
- Characteristic Analysis report

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



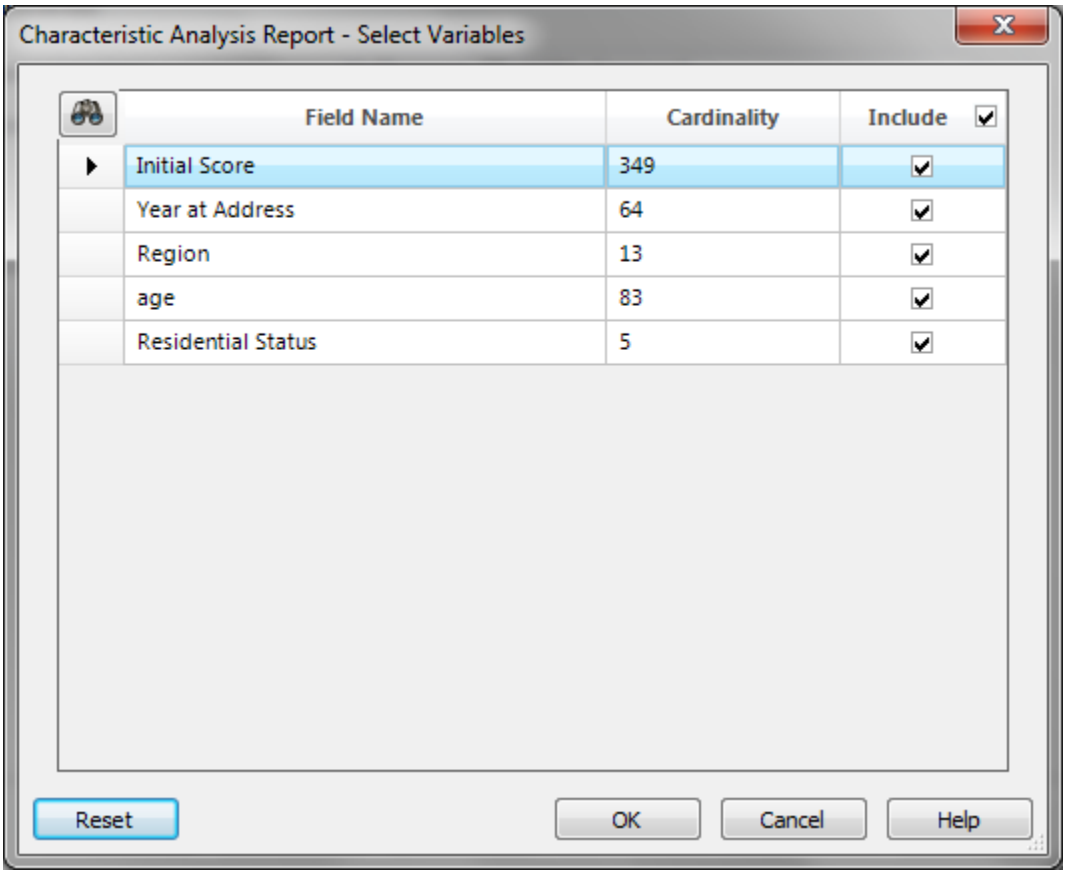
If necessary, click View/Edit Bins to edit the score bins.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

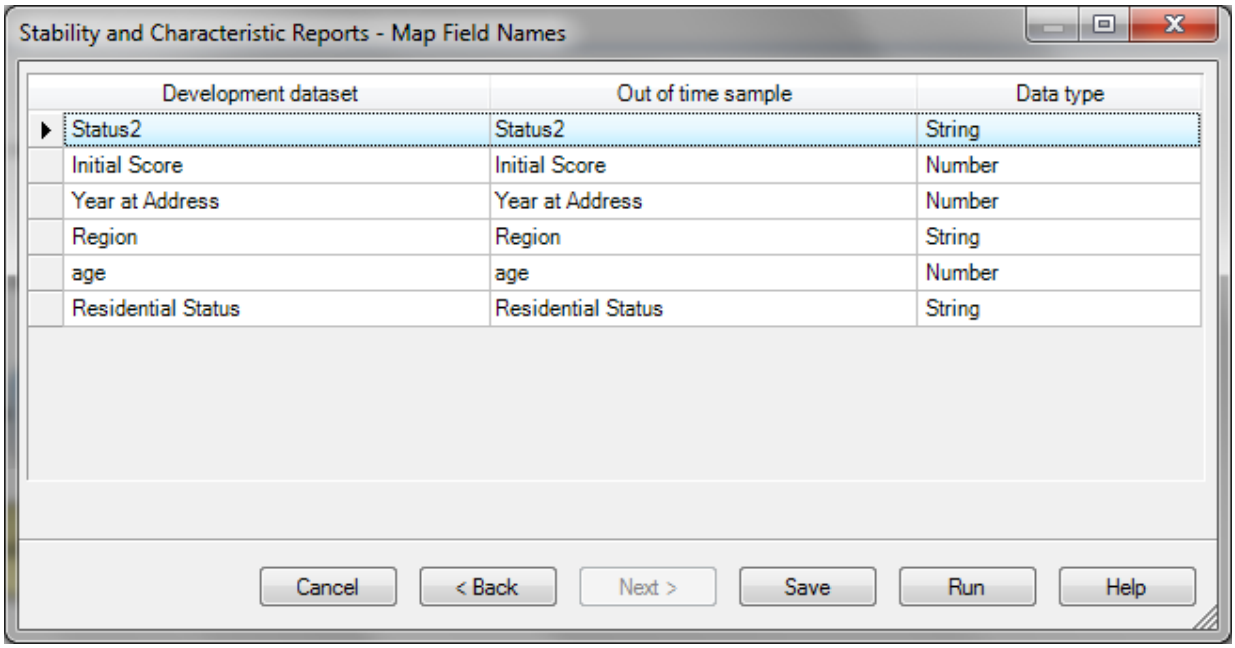


Click the Select Variables button to select or deselect the variables for the characteristic reports:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

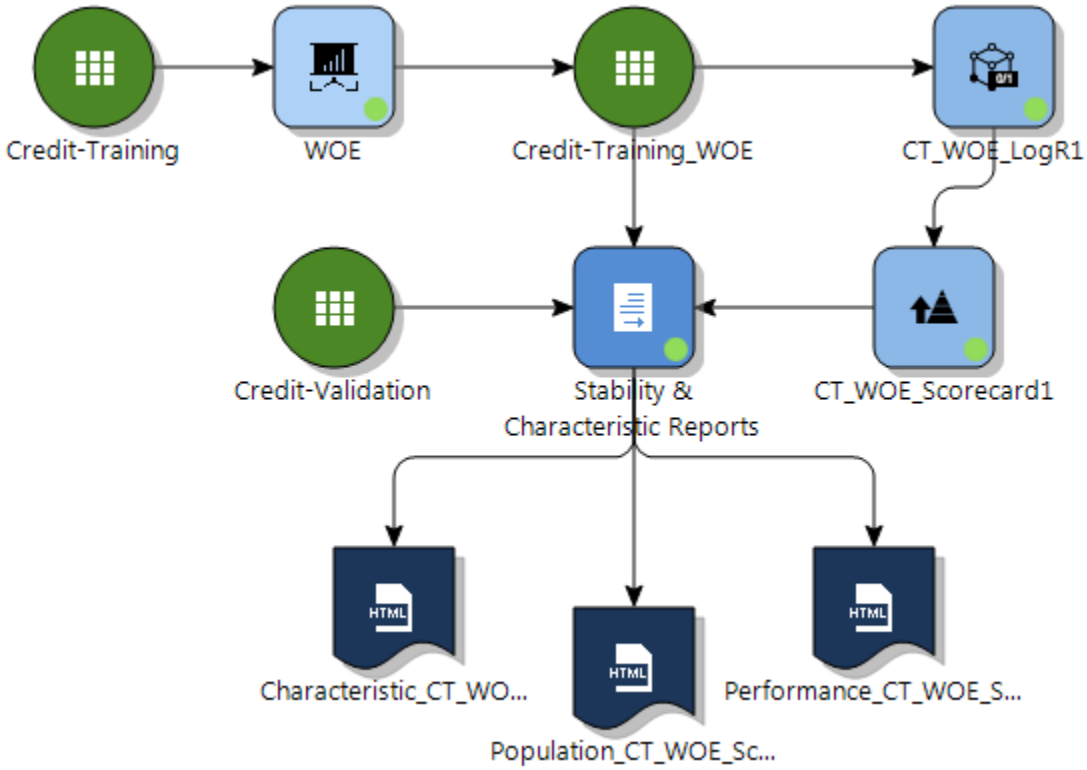


At the next step, map the field names of the development dataset and out of time sample if necessary:



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the operation, double-click the outputs of the Stability & Characteristic Reports node to view the results.



Variable Importance

The **Variable Importance** node in the **Profile** palette is used to estimate the importance of variables in terms of their contribution to a predictive model. The information from the model is used to calculate the variable importance metric. The output includes a ranked list of variables in the order of their importance and a histogram illustrating this in a graphical form.

The Variable Importance metric is computed based on accuracy drop for classification models and Mean Square Error (MSE) for regression models. The node is applicable to the following model types: Decision Tree, Random Forest, Bagging, Boosting, Deep Learning, Linear Regression, Logistic Regression, Constrained Regression, Regularization, Model Stacking, and Reject Inference.

The contribution of each variable is estimated by fitting a one-variable model and calculating the importance metric as follows: First, the model is scored on the dataset to calculate the metric *M* (accuracy or MSE value). Then, for each variable *X*, the values of the variable are permuted, the model is scored on the dataset with the values permuted, and the new value of *M* (accuracy or MSE) is calculated. The difference: (*M* - *Mx*) is taken as the importance of variable *X*.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Note: Indicator variables are kept together, i.e., the categories of a discrete independent variable are not treated separately.

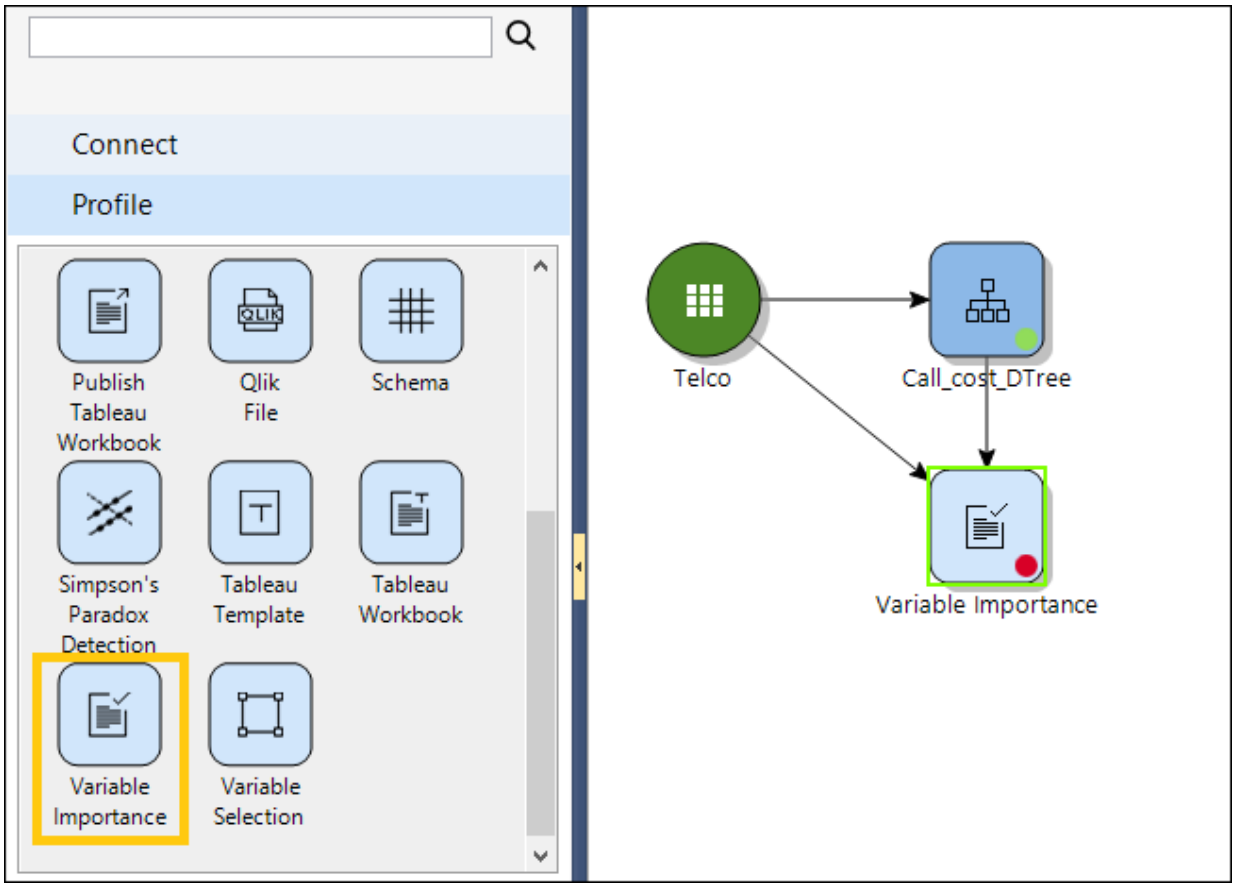
Prerequisites: A dataset node and a model that uses variables from this dataset.

To add a Variable Importance node to your workflow:

- Right-click on the Workflow canvas next to the dataset and model nodes. From the context menu, select **Profile --> Variable Importance**. A new **Variable Importance** process node will be created. Connect the dataset and the model to the new node as shown below.

OR:

- Drag the **Variable Importance** node from the **Profile** palette onto the canvas and connect the dataset and the model to the new node as shown below.

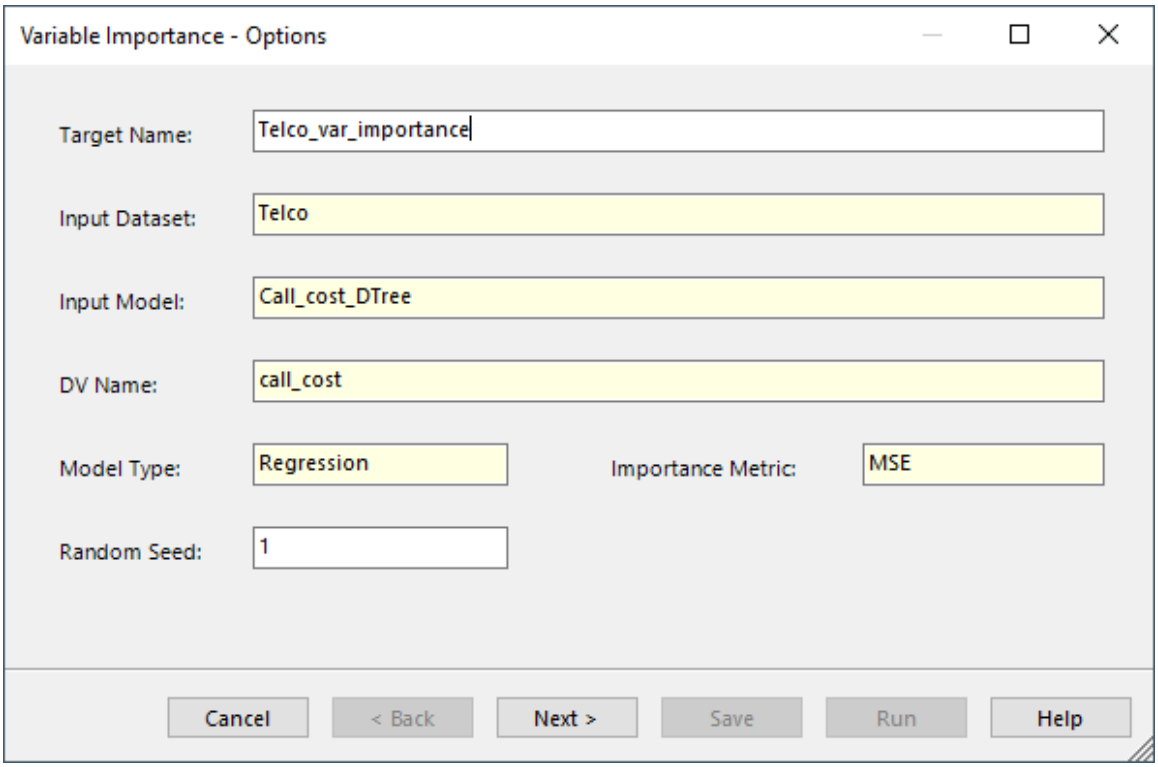


If the Variable Importance node is already on the Workflow canvas, double-click it or right-click on it and select **Modify**.

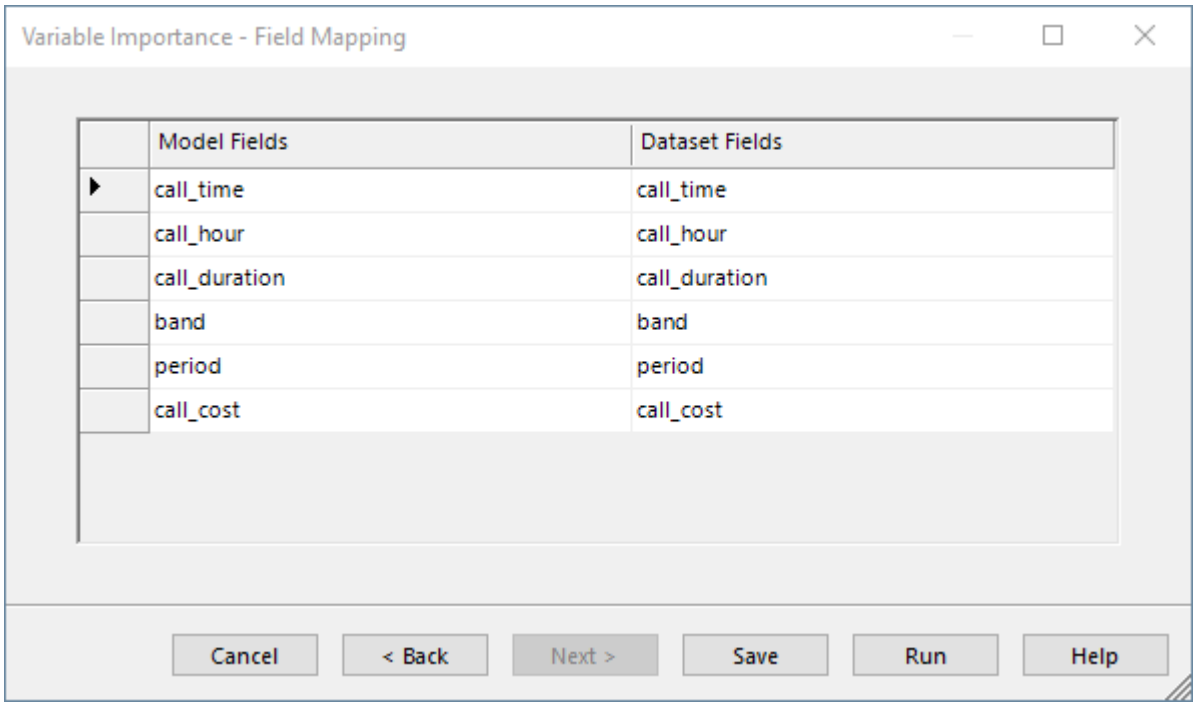
In the Variable Importance wizard:

- In the Target Name field, specify the name for the Variable Importance Report.
- Specify the Random Seed value for the repeatability of results, if necessary.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

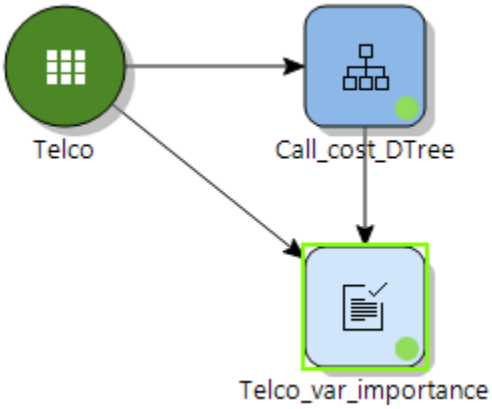
1. 

At the next step, select the input variables to be ranked:



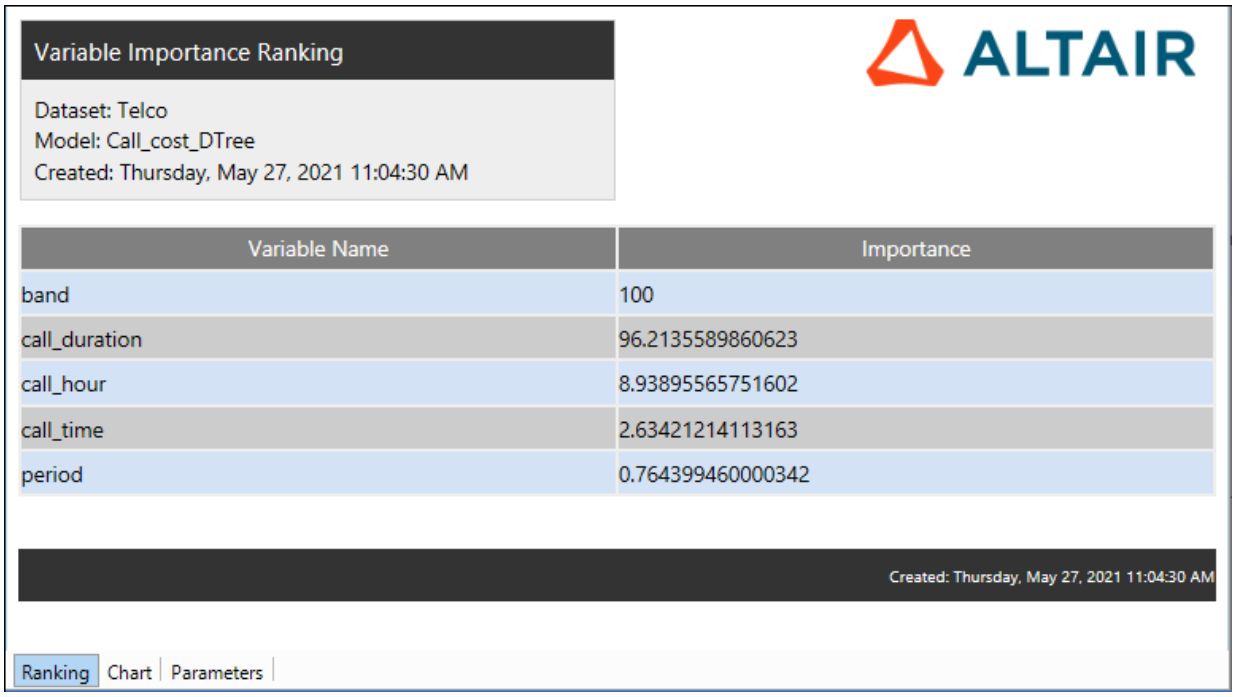
Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the node, double-click the Variable Importance node to view the results.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The Variable Importance report consists of three tabs: Ranking, Chart, and Parameters.

The **Ranking** tab displays the importance for each variable used by the model, in the descending order. You can sort by Variable Name and Importance by clicking on the column headers.



Variable Importance Ranking

Dataset: Telco
Model: Call_cost_DTree
Created: Thursday, May 27, 2021 11:04:30 AM

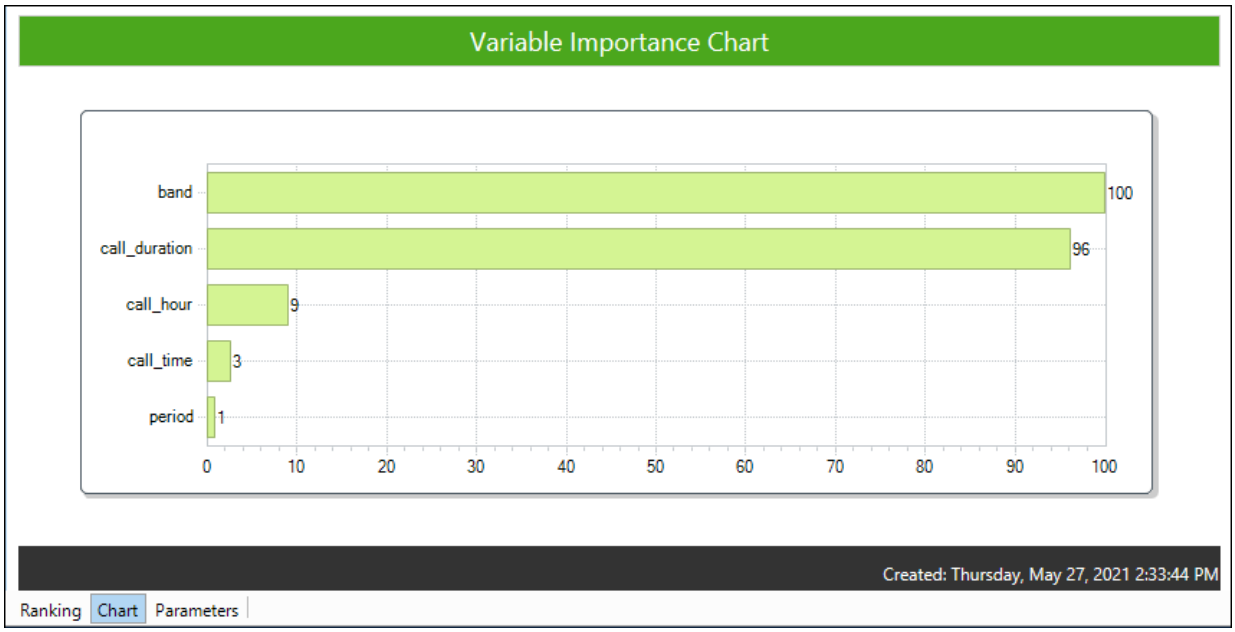
Variable Name	Importance
band	100
call_duration	96.2135589860623
call_hour	8.93895565751602
call_time	2.63421214113163
period	0.764399460000342

Created: Thursday, May 27, 2021 11:04:30 AM

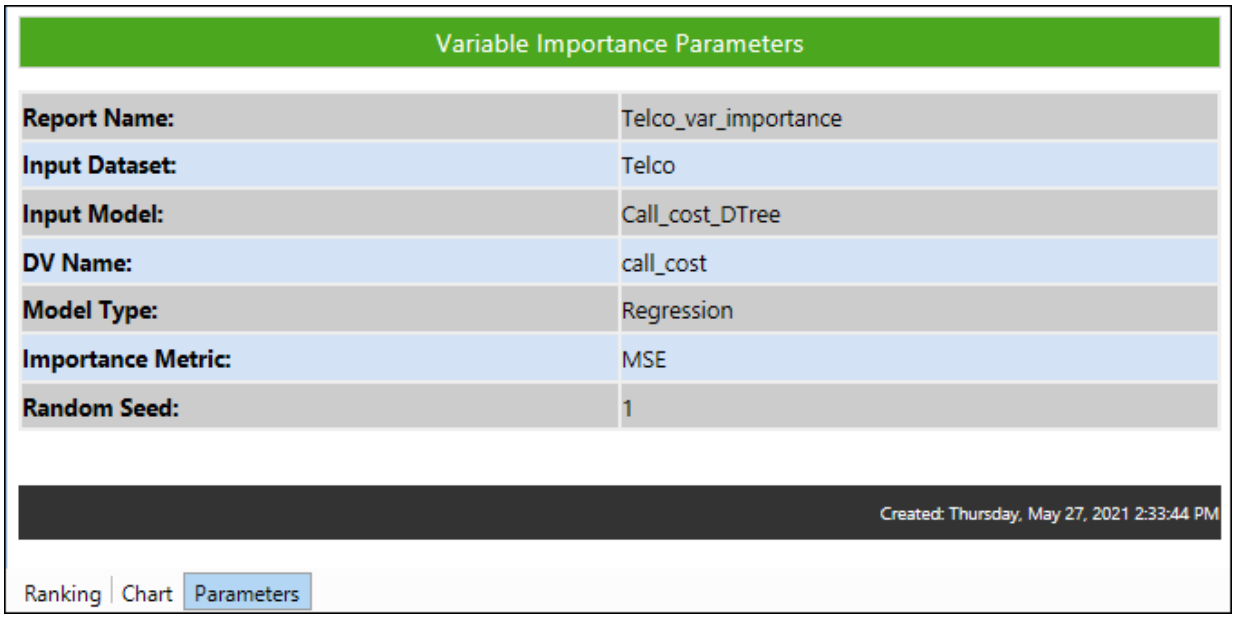
Ranking | Chart | Parameters

The **Chart** tab displays the same information in the form of a histogram:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The **Parameters** tab summarizes the input parameters that were specified in the Variable Importance node wizard:



See also: Measure of Predictive Power

Explainable AI

The **Explainable AI** node in the **Evaluate** palette helps explain and evaluate the results of "black-box" models such as Random Forest and Deep Learning (Neural Networks). Since the solution process of these models is less transparent than that of simpler model techniques, extracting

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

understandable rules from them by means of Explainable AI makes the model results more interpretable by humans.

An **Explainable AI** node is applied to one dataset and one model. Once the node is run, it provides five views to help interpret the input model:

- Decision Tree
- Partial Dependence Plots (PDP)
- Individual Conditional Expectation plots (ICE)
- Local Interpretable Model-Agnostic Explanations (LIME plots)
- Shapley Additive Explanations (SHAP)

Prerequisites:

1. A model of any of these types:
    - Deep Learning
    - Random Forest
    - Bagging
    - Boosting
2. A dataset that includes the variables required by the model.

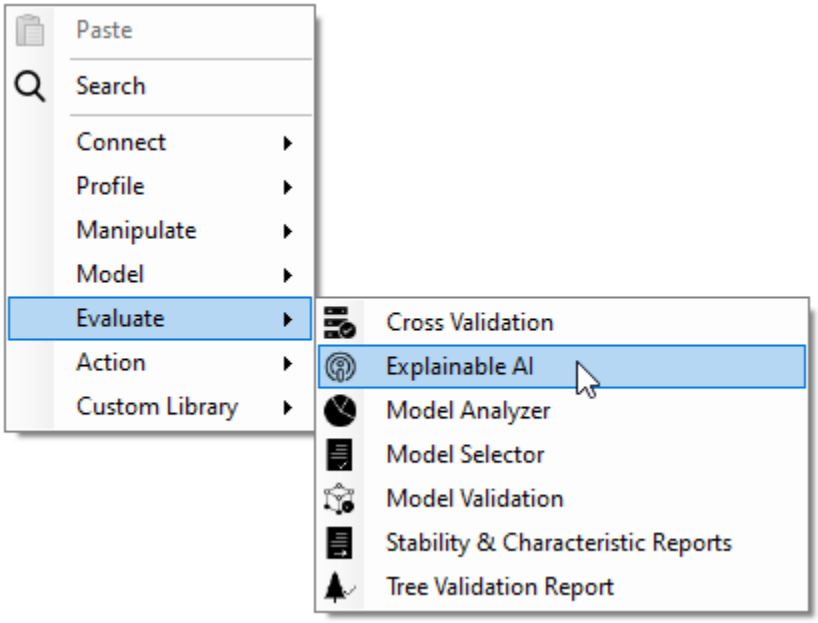
To add an Explainable AI node to your workflow:

- Right-click on the spot on the workflow canvas where you want to place the node. From the context menu, select **Evaluate --> Explainable AI** as shown in the example below. A new **Explainable AI** node will be created. Connect the input dataset node and model node to the Explainable AI node.

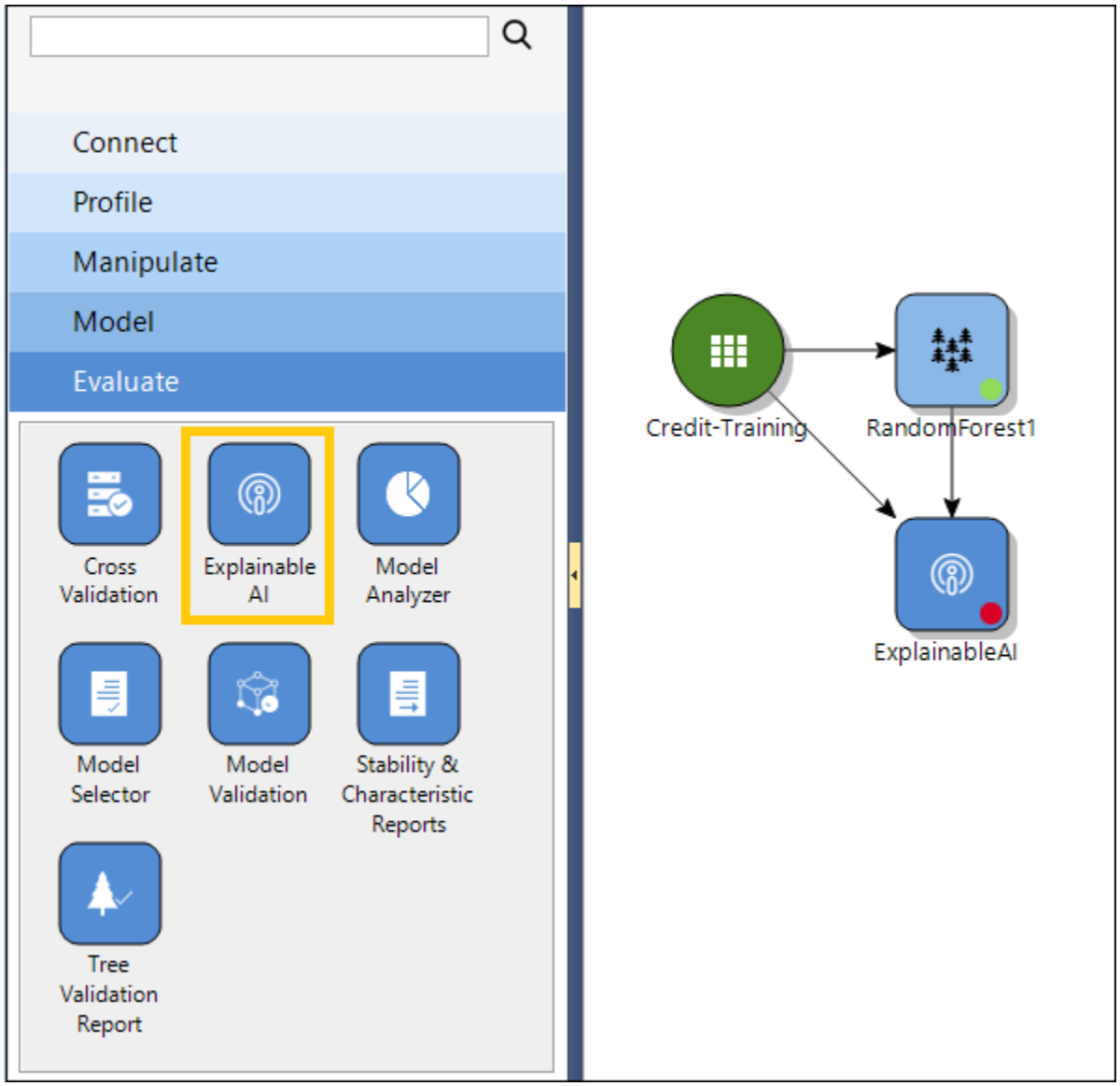
  OR:

- Drag the **Explainable AI** node from the **Evaluate** palette onto the canvas and connect the required input dataset and model nodes to the Explainable AI node.

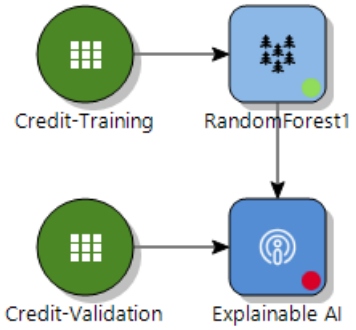
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.



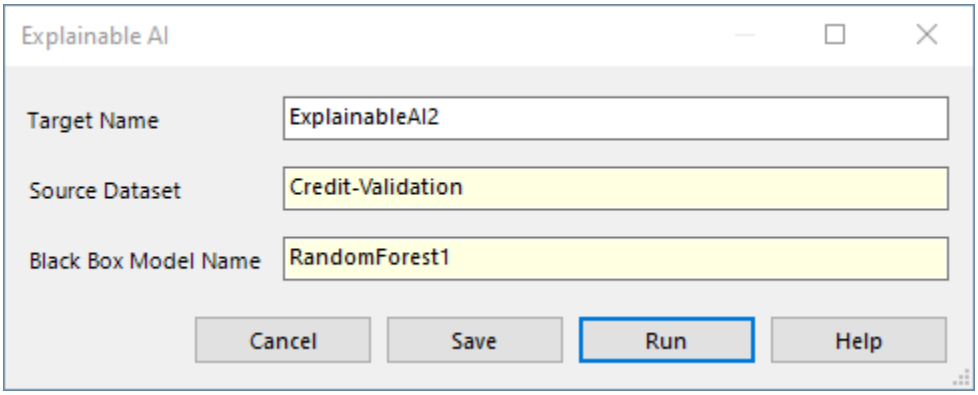
In the general case, the input dataset can be different from the one that was used to train the input model. For example, it can be a validation partition, as shown below:



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Double-click the **Explainable AI** node or right-click on it and select **Modify**.

Specify the name for the new Explainable AI node (*Target Name*). The dialog also shows the names of the input dataset and the input model name ("*Black Box Model Name*").



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the operation, double-click the Explainable AI node to view the results.

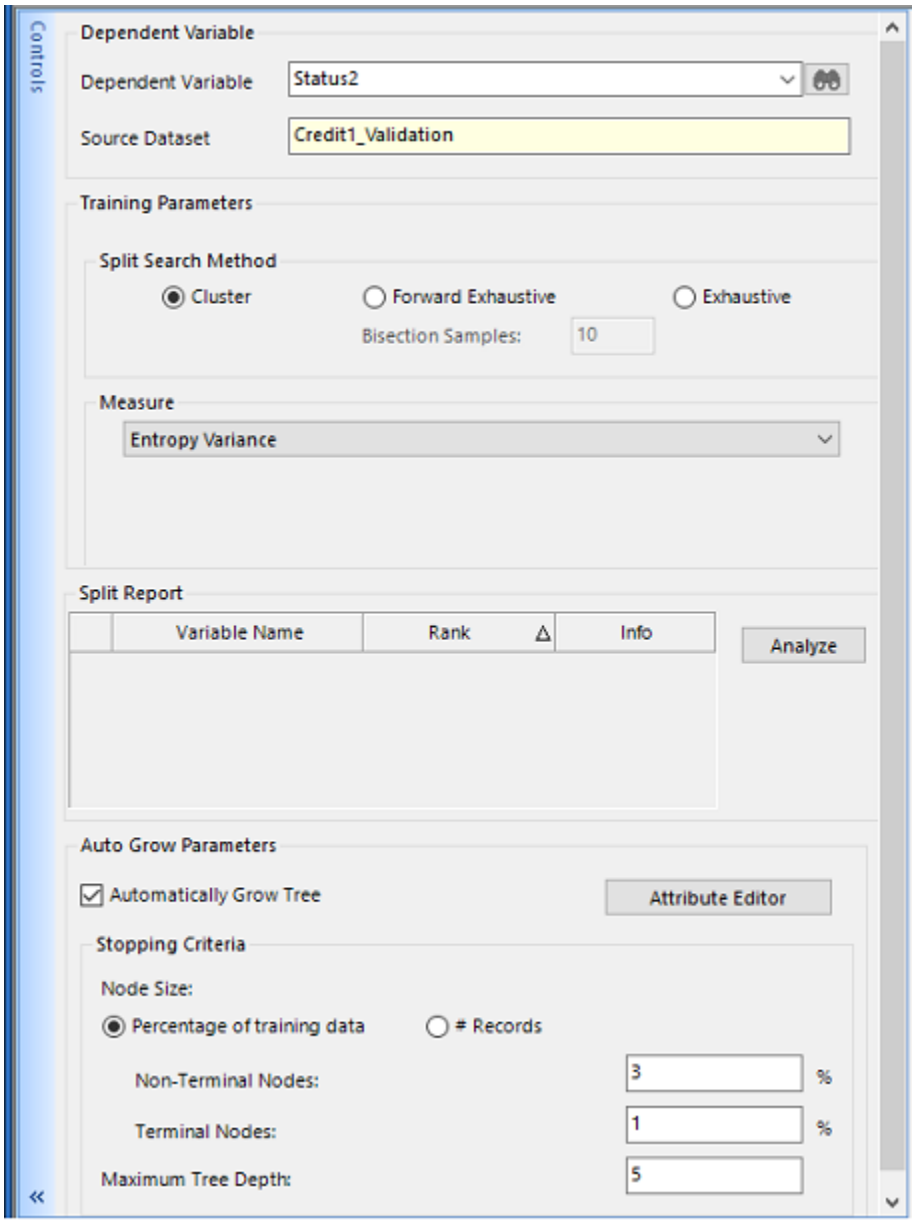
The resulting Explainable AI object consists of five views, which can be accessed using the tabs at the bottom of the page:

1. Decision Tree

The first tab is a Decision Tree. The Controls frame on the left shows Decision Tree settings, which are the same as in the Decision Tree node wizard.

Select the Dependent Variable and specify other parameters just as you do in the Decision Tree wizard. These include Training Parameters (split search method and measure), the Automatic Grow option, and constraints (stopping criteria, which affect AutoGrow and Find Split operations).

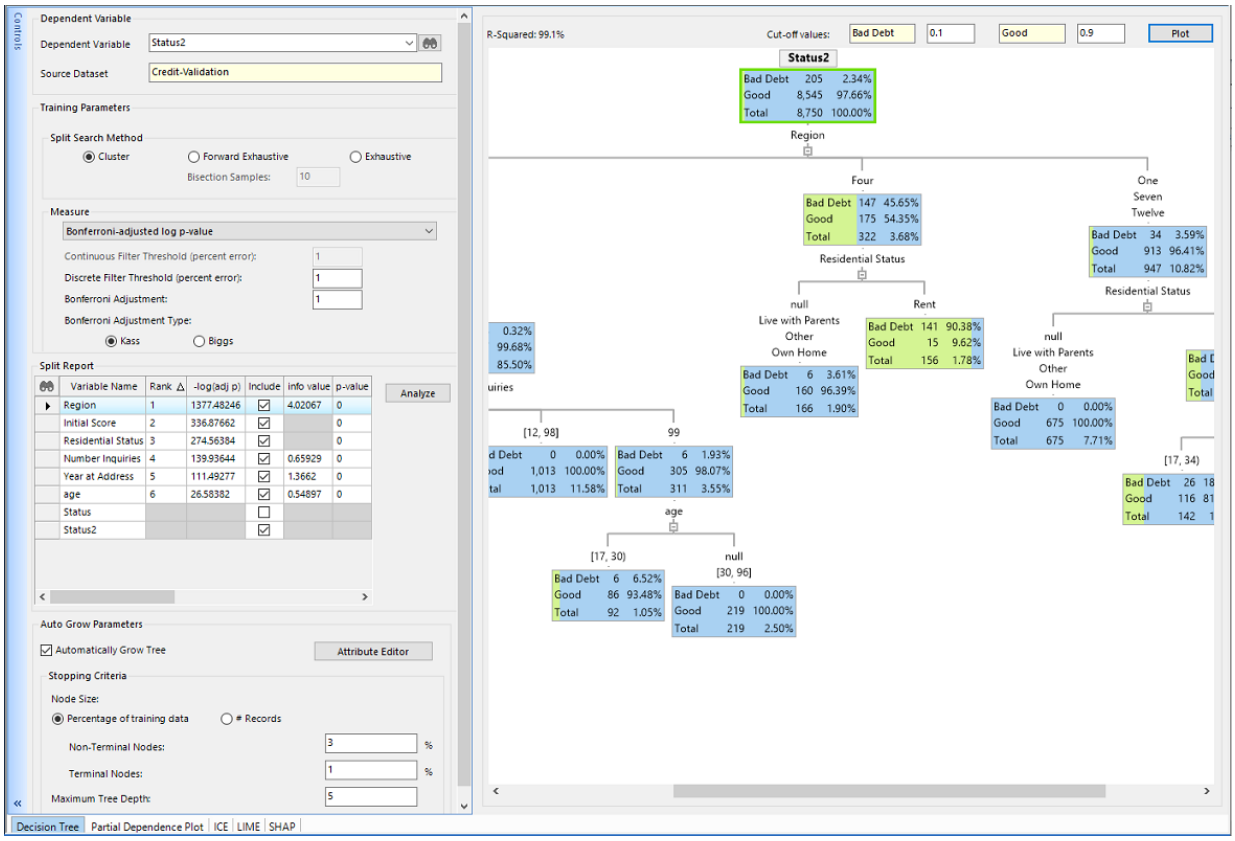
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click the **Plot** button at the top-right corner of the frame on the right to build the Decision Tree on the input dataset scored with the original black-box model. The dependent variable of the tree is the variable with predictions produced by the model scoring. If the original dependent variable of the black-box model is binary, specify the probability cut-off for its values. This is the same cut-off that you normally specify in the Scoring wizard. The default cut-off values are 0.5.

The decision tree will be built with the parameters specified in the left frame and displayed in the frame on the right. You can explore and reshape the Decision Tree splits just like in a regular interactive Decision Tree model node. See the Tree topic for details. The constraints of interactive Automatic Grow override those in the left-hand frame. The R-squared statistic for the Decision Tree model is displayed in the upper-left corner of the tree frame.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



2. Partial Dependence Plot

The second tab is Partial Dependence Plot (PDP).

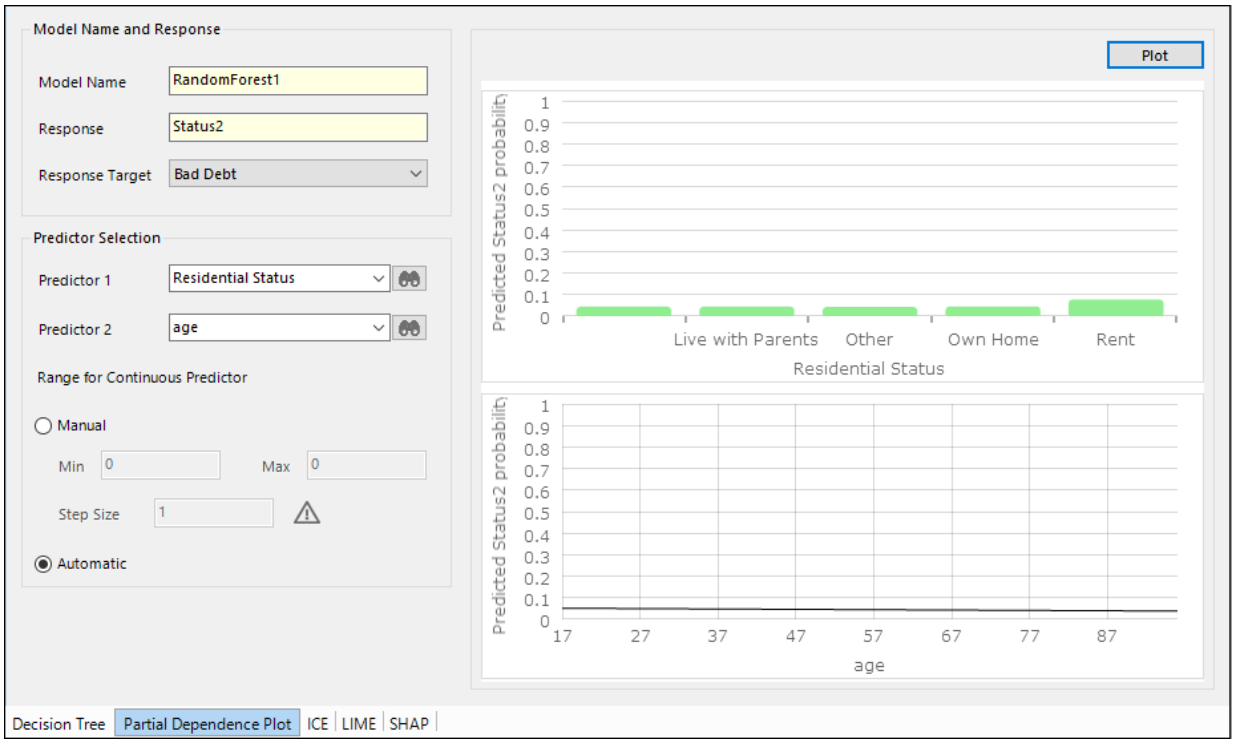
- For *Response Target,* select the desired value of the dependent variable.
- Select the predictors to which the PDP has to be plotted.
- If at least one predictor is continuous, select the range option for continuous predictor (*Manual* or *Automatic*).

    If the Manual option is selected, enter the minimum, maximum, and the step size.

    Note: By default, in the automatic mode, the step size is chosen so that 100 evenly spaced values across the range are used. If the step size is chosen so that a much larger number of values is required, computation time will increase significantly.

Click the **Plot** button at the top-right corner. This will present the plots for the selected predictors, with predicted DV value probability plotted across the categories or ranges of the predictors.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

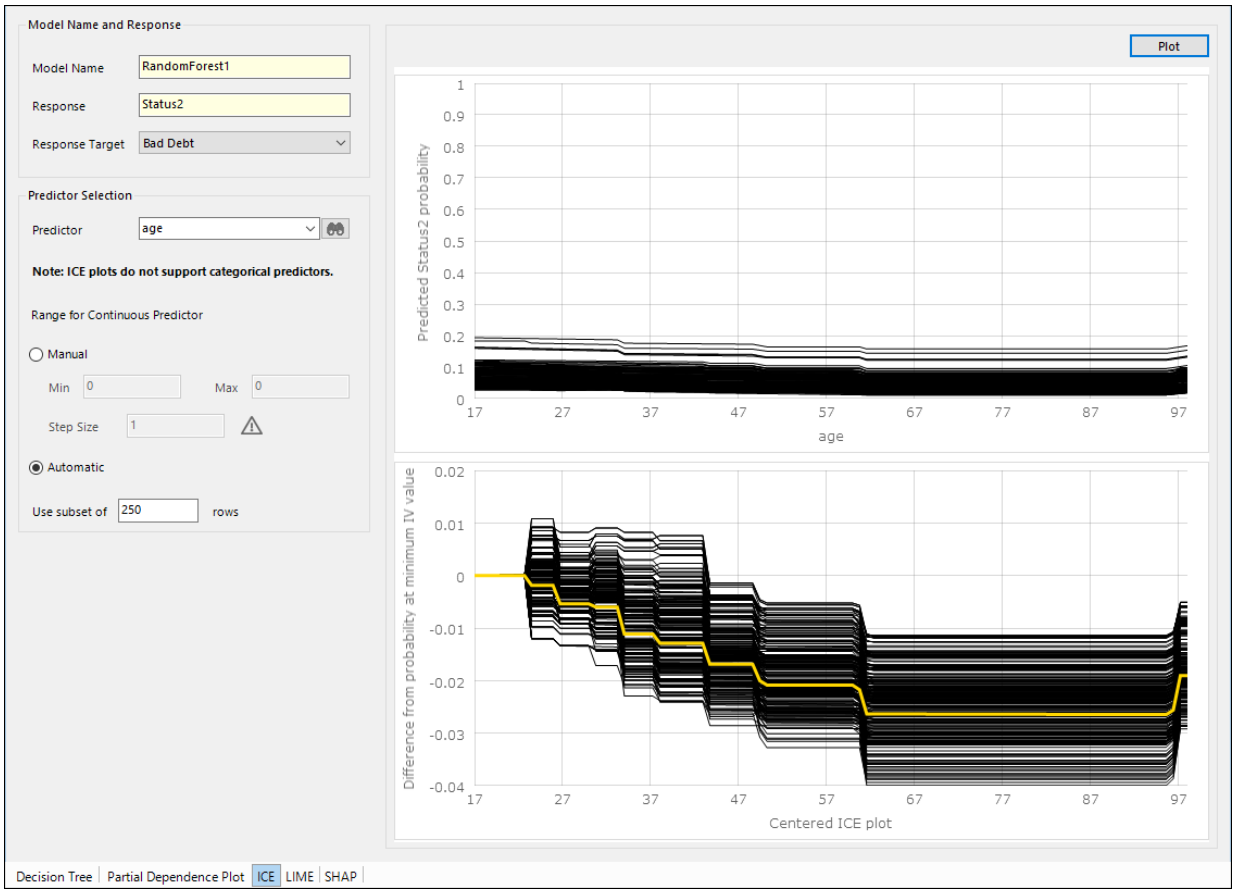


3. Individual Conditional Expectation plots (ICE)

The third tab presents Individual Conditional Expectation plots (ICE). This is the same as PDP except a line is generated for each row in the input dataset or its subset.

- For *Response Target,* select the desired value of the dependent variable.
- Select the predictor to which the ICE has to be plotted. The predictor variable must be continuous. Discrete (categorical) predictors are not supported.
- Choose the range option (*Manual* or *Automatic*) for the selected continuous predictor. For each row, a distribution is created for the range of the selected variable, and the values are sorted.
    - If the ***Automatic*** option is selected, enter the number of rows in the sample to be used. The default is 250 rows.
    - If the ***Manual*** option is selected, specify the range minimum and maximum, and the step size.
        - Note: By default, in the automatic mode, the step size is chosen so that 100 evenly spaced values across the range are used. If the step size is chosen manually so that a much larger number of values is required, computation time will increase significantly.
- Click the **Plot** button at the top-right corner. This will produce the plots for the selected predictor: the difference between each probability of the dependent variable on each curve and the probability corresponding to the minimum value of the selected predictor for that curve. The yellow line in the Centered ICE plot is the average of all the centered ICE lines, similar to PDP.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



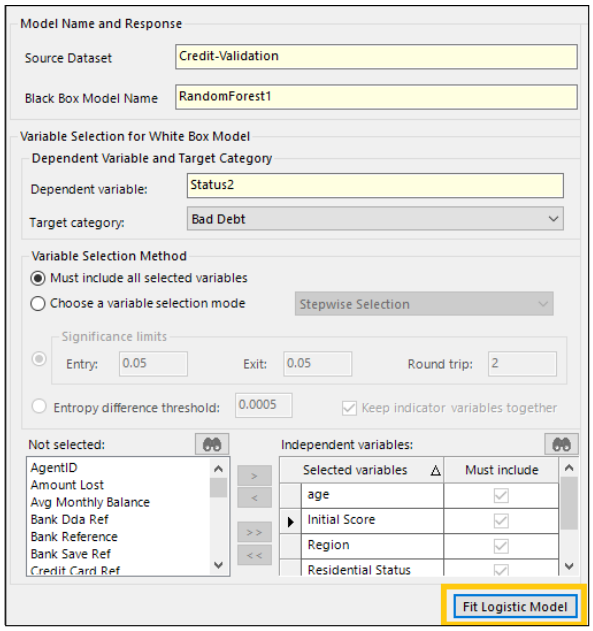
4. Local Interpretable Model-Agnostic Explanations (LIME plots)

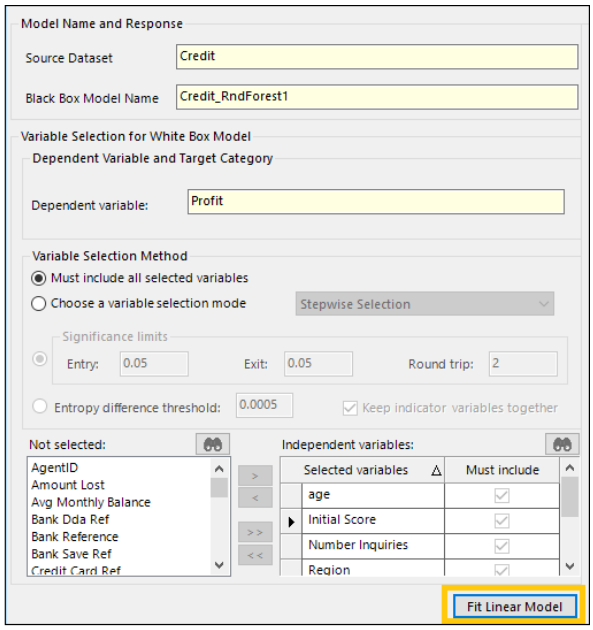
The fourth tab presents the plots for ***Local Interpretable Model-Agnostic Explanations*** (**LIME**). These plots help determine the most important contributors to the model prediction and the extent of their contribution. By selecting a subset of predictors and varying their values, the user can test different model inputs and observe the effect of the changes on the predictions. Thus, the model predictions can be interpreted without knowing the model structure, and conclusions can be made regarding the adequacy or "sensibility" of the model from the user's viewpoint.

More precisely, the LIME Plots view provides a local approximation of the input Black Box model's behavior using linear or logistic regression. The regression model is fit around the user-specified data point (observation). N records are sampled around the specified data point X and weighted according to their proximity to X. A regression model that approximates the Black Box model is fit in the vicinity of X.

The *Model Name and Response* section displays the input dataset and model names. Specify your inputs in the section ***Variable Selection for White Box Model***, which is the same as in the Logistic Regression or Linear Regression model wizard, depending on whether the dependent variable is discrete or continuous.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.





- If the dependent variable is binary, select the *Target category*.
- Choose the *Variable selection method.* The options are similar to those in the Linear and Logistic Regression model wizards:
  - *Must include all selected variables:* All the variables selected as *Independent variables* will be used when training the model, regardless of their significance.
  - *Choose a **variable selection method**:*

Stepwise	Include variables depending on the Entry and Exit Significance Limits.
Forward Selection	Starts with no variables and then includes variables based on their rank order from the best to the worst. The rank order is determined by a log-likelihood ratio test.
Backward Selection	Starts with all the variables and then excludes variables based on the rank order of variables from the worst to the best. The rank order is determined by a log-likelihood ratio test.
R-Squared Selection	Check through all possible combinations of coefficients and produce a ranked list with R-squared increasing as various combinations go in.

  - Significance limits*:*

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Entry	The entry significance level determines when a variable is qualified to be included in the model. This parameter is used for the **Stepwise** and **Forward** selection methods.
Exit	The exit significance level determines when a variable is disqualified and excluded from the model. This parameter is used for the **Stepwise** and **Backward** selection methods.
Round Trip	The number of entry-exit-entry-exit sequences before a variable is disqualified and excluded from the model.

- ***Entropy Difference Threshold*** defines a stopping criterion for model training with the Stepwise, Backward, and Forward variable selection methods. If this option is selected, the model training is stopped when the change in Entropy Explained in consecutive models in the iterative process is less than the user-defined level. The default threshold is 0.0005. This option is not applicable when Significance Limits are specified or when the variable selection method is "R-Squared Selection". After the training is finished, you can check the Entropy Explained values in the Results tab of the model in the sequence.
- **K*eep indicator variables together****:* When this option is selected, the categories of each discrete variable are treated as a group. When a discrete variable is included or excluded from the model all the categories within the variable are included or excluded together as a group. For example, **marital-status** contains seven categories: *Single, Married***,** etc. If **marital-status** is excluded, all the categories are excluded from the model. When this option is unchecked, each category is evaluated separately and will be included or excluded individually. This option is selected by default.

Independent Variable Selection and Must-Include Variables

Variables on the left are not included in the model. Variables listed on the right in the "*Independent Variables*" list will be included.

To select more than one variable in the lists, use mouse dragging or mouse clicks while holding down the SHIFT key or the CTRL key. If there are a lot of variables (hundreds or thousands), it is convenient to use Field Finder (the binoculars button ).

If you chose a particular variable selection method, you have an option of selecting the variables that must be forced into the model regardless of the selection method (the **Must include** column will be enabled). To force specific variables into the model regardless of the selection method criteria, select them in the **Must include** column.

When finished, click the **Fit Logistic Model** or **Fit Linear Model**, depending on the type of the dependent variable.

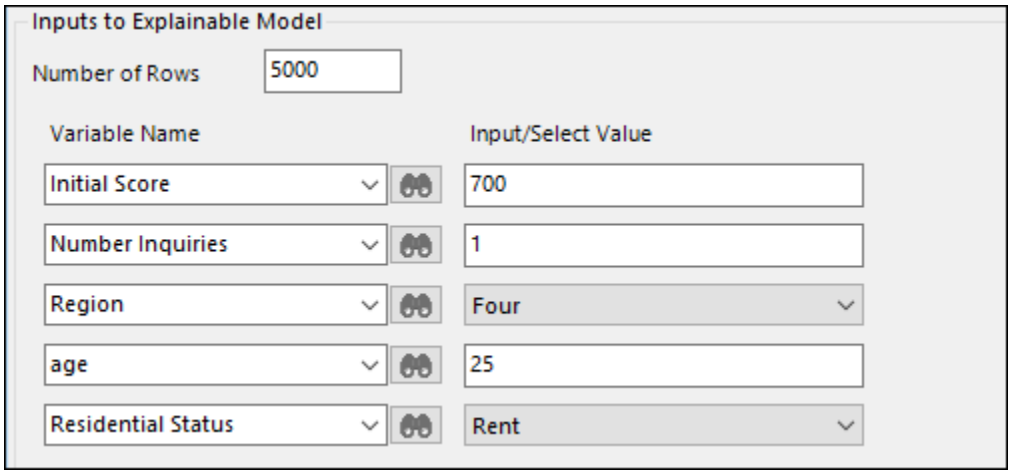
Once the model is fit, the fields in the section **Inputs to Explainable Model** become enabled.

Inputs to Explainable Model

In the section *Inputs to Explainable Model:*

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Specify the sample size (*Number of Rows*) to use for training the regression model. The default is 5000 or the input dataset size, whichever is smaller.
- Select the predictor variables to be used for the explainable model and enter their values (scoring inputs).



Building the LIME Plot

If necessary, specify the Random Seed value for repeatability of results.

Click the **Plot** button at the top-right corner to build the "White Box" model and produce the LIME plot. If the dependent variable is discrete, the White Box model is logistic regression. Otherwise, it is linear regression. A regression model is fit for the single data point defined by the specified values of the variables, which will approximate the predictive behaviour of the Black Box model on that data point.

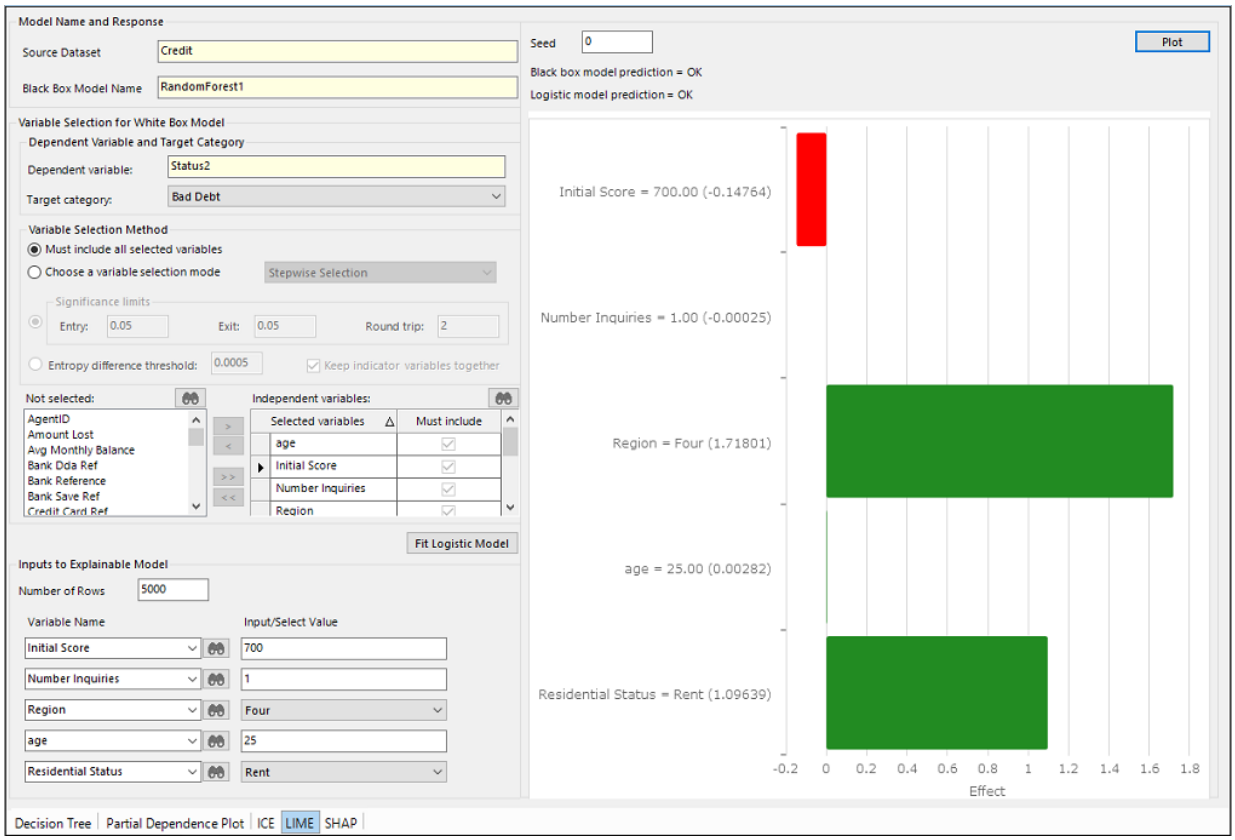
For each of the selected variables, random normal distribution is generated, and the mean and standard deviation are computed for all records in the training sample. A randomized version of each variable is produced, then the distance of every row from the mean of the respective variable is computed. The rows are weighted based on the distance.

At the top of the plot frame, the predictions produced by the input Black Box model and the Explainable model are shown.

The selected variables and their values are shown on the Y-axis.

The X axis (*Effect*) shows the weighted response for each selected predictor. For a discrete dependent variable, the X axis shows the weighted probability of the target category. For a continuous dependent variable, the X axis has the weighted prediction value.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

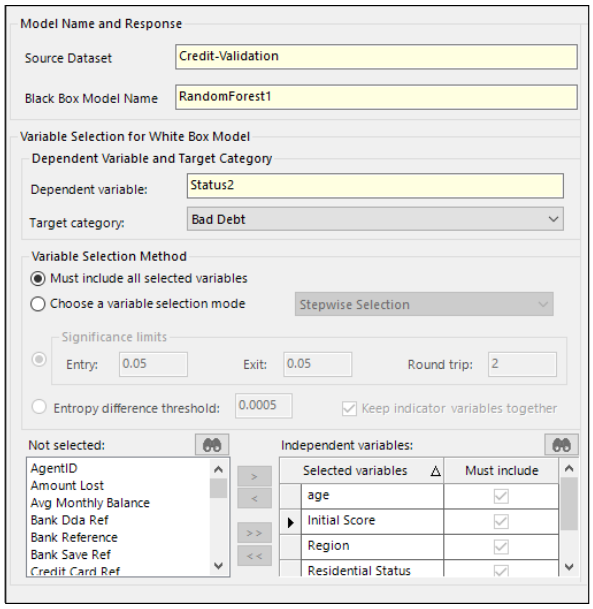


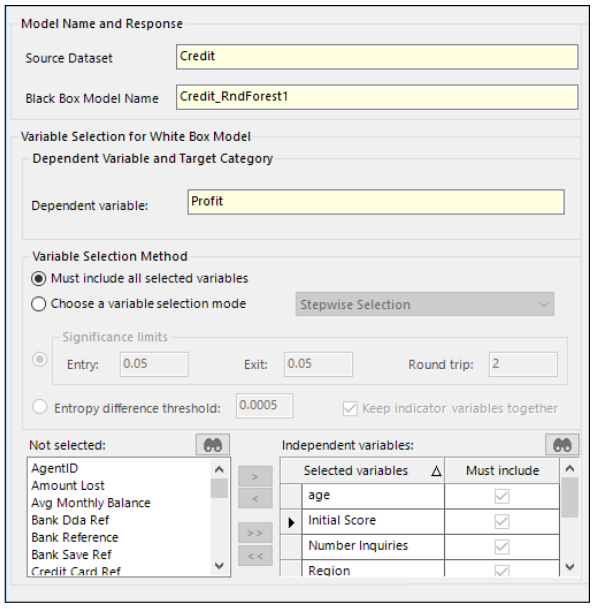
5. Shapley Additive Explanations (SHAP)

The **SHAP** tab represents the SHAP (SHapley Additive exPlanations) plots, which illustrate the contribution of individual variables to the model prediction using their *Shapley values*. The Shapley value of a variable is the expected value of the linear model minus the coefficient of the variable in Shapley value calculation. SHAP values are approximated using the *Kernel SHAP* method, which is similar to LIME, with a different weighting function. For reference on the original SHAP method, see Lundberg, S. M. and Lee, S. (2017). A Unified Approach to Interpreting Model Predictions. *Advances in Neural Information Processing Systems 30* (*NIPS 2017*).

The *Model Name and Response* section displays the input dataset and model names. Specify your inputs in the section ***Variable Selection for White Box Model***, which is the same as in the Logistic Regression or Linear Regression model wizard, depending on whether the dependent variable is discrete or continuous.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.





- If the dependent variable is binary, select the *Target category*.
- Choose the *Variable selection method.* The options are similar to those in the Linear and Logistic Regression model wizards:
  - *Must include all selected variables:* All the variables selected as *Independent variables* will be used when training the model, regardless of their significance.
  - *Choose a **variable selection method**:*

Stepwise	Include variables depending on the Entry and Exit Significance Limits.
Forward Selection	Starts with no variables and then includes variables based on their rank order from the best to the worst. The rank order is determined by a log-likelihood ratio test.
Backward Selection	Starts with all the variables and then excludes variables based on the rank order of variables from the worst to the best. The rank order is determined by a log-likelihood ratio test.
R-Squared Selection	Check through all possible combinations of coefficients and produce a ranked list with R-squared increasing as various combinations go in.

  - Significance limits*:*

Entry	The entry significance level determines when a variable is qualified to be included in the model. This parameter is used for the **Stepwise** and **Forward** selection methods.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Exit	The exit significance level determines when a variable is disqualified and excluded from the model. This parameter is used for the **Stepwise** and **Backward** selection methods.
Round Trip	The number of entry-exit-entry-exit sequences before a variable is disqualified and excluded from the model.

- ***Entropy Difference Threshold*** defines a stopping criterion for model training with the Stepwise, Backward, and Forward variable selection methods. If this option is selected, the model training is stopped when the change in Entropy Explained in consecutive models in the iterative process is less than the user-defined level. The default threshold is 0.0005. This option is not applicable when Significance Limits are specified or when the variable selection method is "R-Squared Selection". After the training is finished, you can check the Entropy Explained values in the Results tab of the model in the sequence.
- **K*eep indicator variables together****:* When this option is selected, the categories of each discrete variable are treated as a group. When a discrete variable is included or excluded from the model all the categories within the variable are included or excluded together as a group. For example, **marital-status** contains seven categories: *Single, Married***,** etc. If **marital-status** is excluded, all the categories are excluded from the model. When this option is unchecked, each category is evaluated separately and will be included or excluded individually. This option is selected by default.

Independent Variable Selection and Must-Include Variables

Variables on the left are not included in the model. Variables listed on the right in the "*Independent Variables*" list will be included.

To select more than one variable in the lists, use mouse dragging or mouse clicks while holding down the SHIFT key or the CTRL key. If there are a lot of variables (hundreds or thousands), it is convenient to use Field Finder (the binoculars button ).

If you chose a particular variable selection method, you have an option of selecting the variables that must be forced into the model regardless of the selection method (the **Must include** column will be enabled). To force specific variables into the model regardless of the selection method criteria, select them in the **Must include** column.

Inputs to Explainable Model

In the section *Inputs to Explainable Model,* specify the sample size (*Number of Rows*) to use for training the explainable model. The default is 5000 or the input dataset size, whichever is smaller.



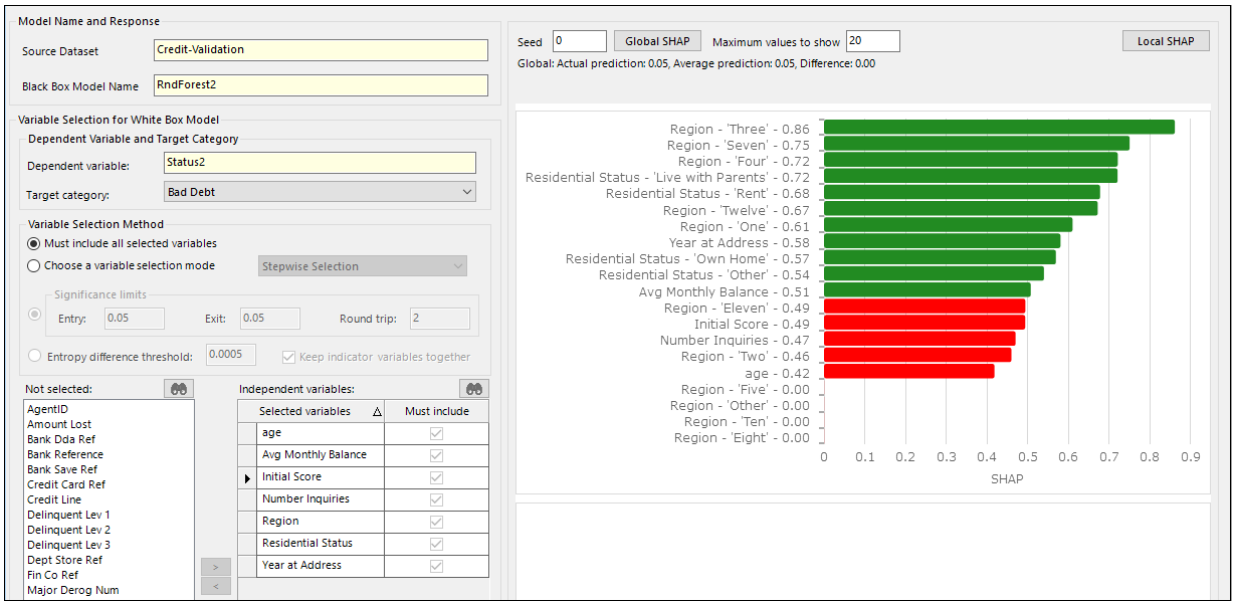
Building the SHAP Plots

- Specify the Random Seed value for the repeatability of results, if necessary. The default value is 0.
- Specify the maximum number of values to show. The default is 10.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Click the **Global SHAP** button to plot the average Shapley value of each variable across the dataset (*global* Shapely values). The selected independent variables are on the Y axis, and their Shapley values are plotted on the X axis. The bars are colored based on the sign of the coefficient of the variable in the surrogate model. The Shapley values are also displayed for each variable on the Y axis at the left end of each bar (coefficients in the case of a continuous dependent variable; probabilities in the case of a discrete dependent variable).

The values of actual prediction, average prediction, and the difference are shown at the top of the chart frame. Double-click on the chart to open it in a separate resizeable window.



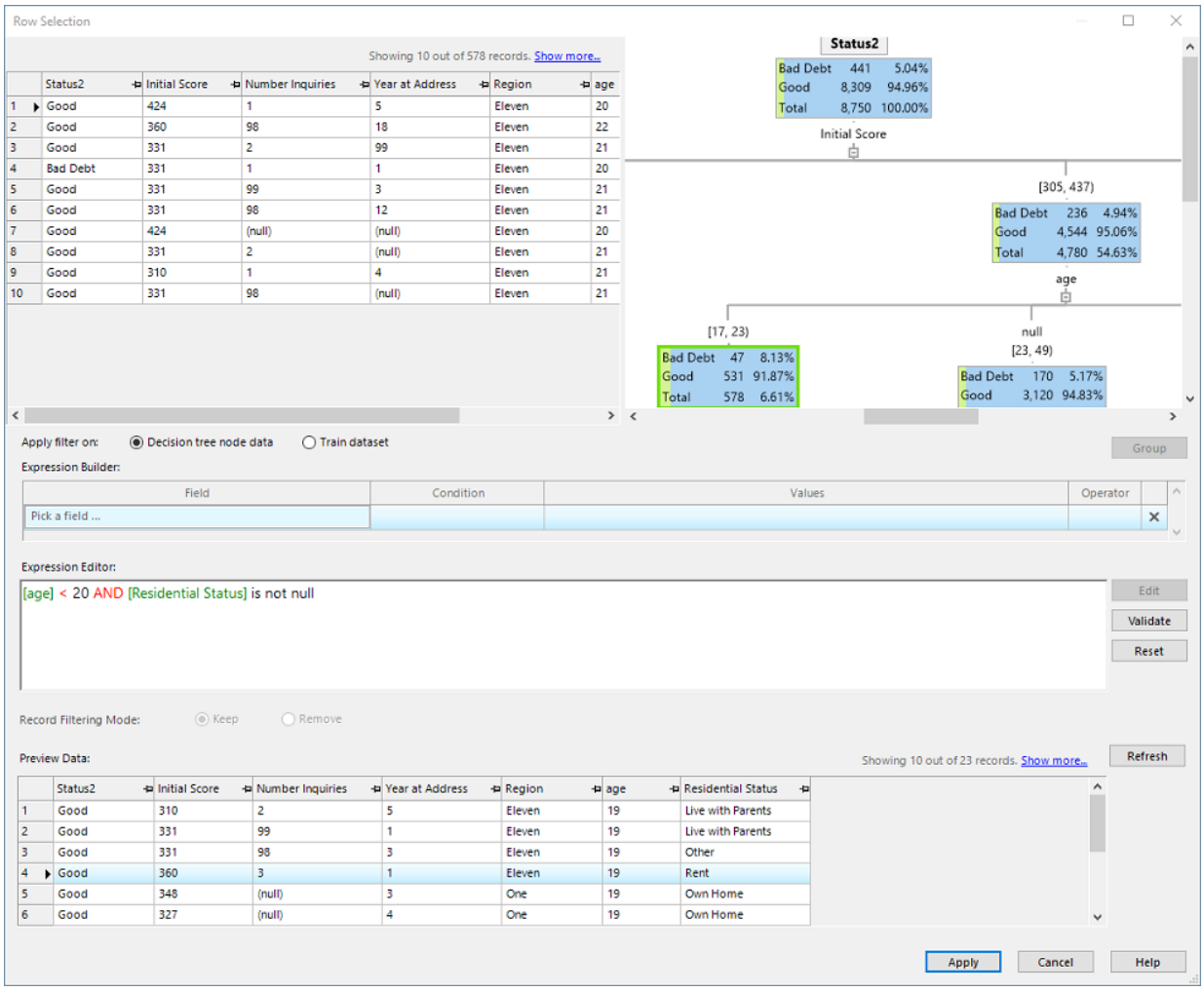
Click the **Local SHAP** button to select the row for which local Shapley values must be calculated for each variable. The **Row Selection** window opens. The top-right frame displays the Decision Tree model built with the selected dependent and independent variables on the unaltered training set. As you traverse the decision tree by clicking on the tree nodes, the top-left frame displays the data in the currently selected node. You can split and reshape the tree in this frame just as you would normally do in a regular decision tree node. The frame can be resized by dragging on the horizontal separator.

Define the desired filter condition using the **Expression Builder** interface or by entering an expression using the **Expression Editor**.

You can apply the filter to the records in the currently selected node only (the **Decision tree node data** option) or on the entire training dataset (the **Train dataset** option).

Click the **Refresh** button on the right to preview the filtered data in the **Preview Data** frame at the bottom of the window.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



	Status2	Initial Score	Number Inquiries	Year at Address	Region	age
1	Good	424	1	5	Eleven	20
2	Good	360	98	18	Eleven	22
3	Good	331	2	99	Eleven	21
4	Bad Debt	331	1	1	Eleven	20
5	Good	331	99	3	Eleven	21
6	Good	331	98	12	Eleven	21
7	Good	424	(null)	(null)	Eleven	20
8	Good	331	2	(null)	Eleven	21
9	Good	310	1	4	Eleven	21
10	Good	331	98	(null)	Eleven	21

	Status2	Initial Score	Number Inquiries	Year at Address	Region	age	Residential Status
1	Good	310	2	5	Eleven	19	Live with Parents
2	Good	331	99	1	Eleven	19	Live with Parents
3	Good	331	98	3	Eleven	19	Other
4	Good	360	3	1	Eleven	19	Rent
5	Good	348	(null)	3	One	19	Own Home
6	Good	327	(null)	4	One	19	Own Home

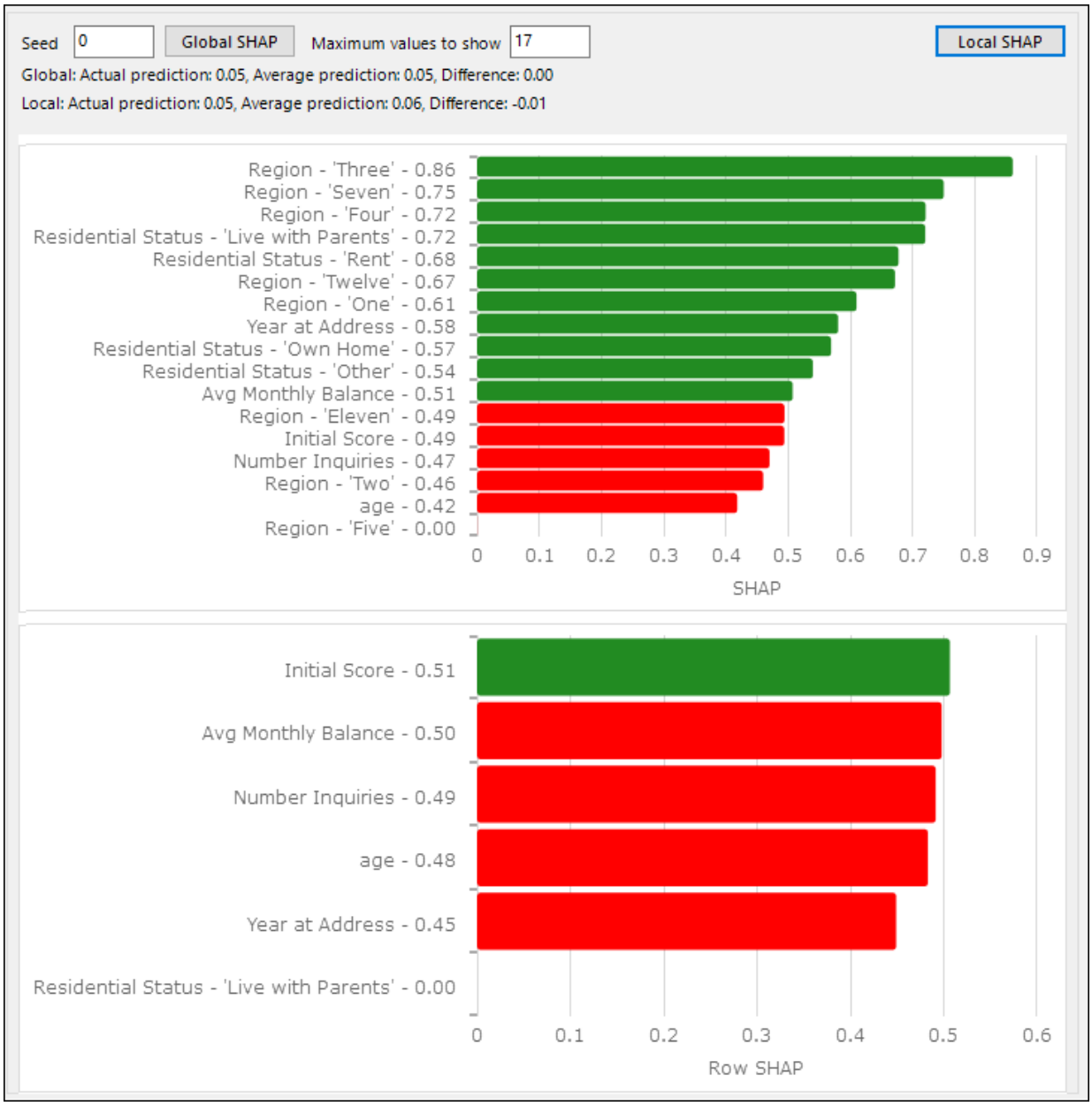
Once you have narrowed down the set of records to select from, click on the desired row in the Preview Data frame and click **Apply**. The SHAP plot will show *local* Shapley values for a single row (the row selected above in the Preview Data frame) for each of the selected variables.

The design and color code for the local SHAP plot is similar to that of LIME plot (positive values are shown in green color, while negative ones are shown in red). The predicted value, actual value and the average value are displayed at the top.

For continuous dependent variable, the Shapley values plotted on the X axis are the coefficients of the linear model built on the randomized dataset created for each row. For discrete dependent variable, the Shapley values are probabilities of the coefficients `prob = exp(beta/(1+exp(beta))`

Double-click on any of the charts to open it in a separate resizeable window, if necessary.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Model Analyzer

Model Analyzer allows you to extract information about the models you created and then display the information in the form that can be used for analysis. You can create charts and statistics for evaluating discrete and continuous models and use these charts to compare the performance of different models. Also, you can compare model performance using validated datasets (i.e., datasets obtained as a result of model validation).

In a nutshell, a model analyzer allows you to:

- evaluate how well your model can classify the data

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- validate how well you model works with different data
- compare different models to see which model provides the best lift.

Before using the model analyzer, you must create, or already have, a decision tree or a predictive model in your project. Once a model is added to the model analyzer, information is calculated and displayed in the form of charts.

If you wish to compare several models in Model Analyzer, you can add another model at any time using the menu command **Tools | Add Model**. To remove a model from Model Analyzer, select **Tools | Remove Model.**

Proceed to Inserting a Model Analyzer.

See also: About Validation.

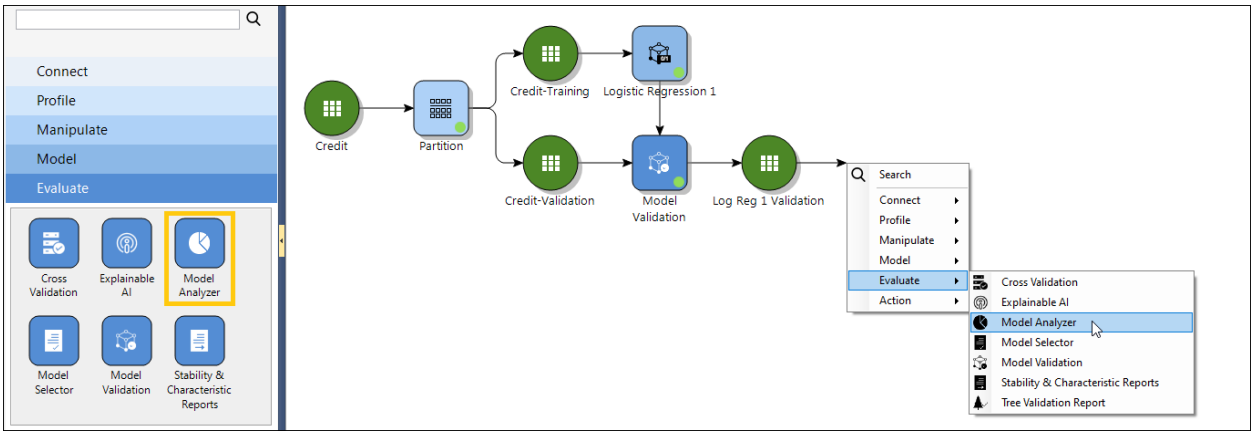
If you have a validated dataset (a dataset that is a result of validating a model), Model Analyzer can estimate the performance of the model on the basis of this dataset. The necessary condition is that the dataset must have a field with actual values of the variable of interest and the field with values predicted by the model (DV prediction field or Scorecard field). In Altair Data Analytics applications, datasets with these fields are generated as a result of validating a model (see the Model Validation topic). However, the origin of these datasets does not really matter. You can even use datasets with prediction fields generated in other applications. The most important thing is the presence of the fields with predicted values to compare to actual values.

To add a Model Analyzer operation to your workflow:

- Drag the **Model Analyzer** node from the **Evaluate** palette and connect the input dataset to it.

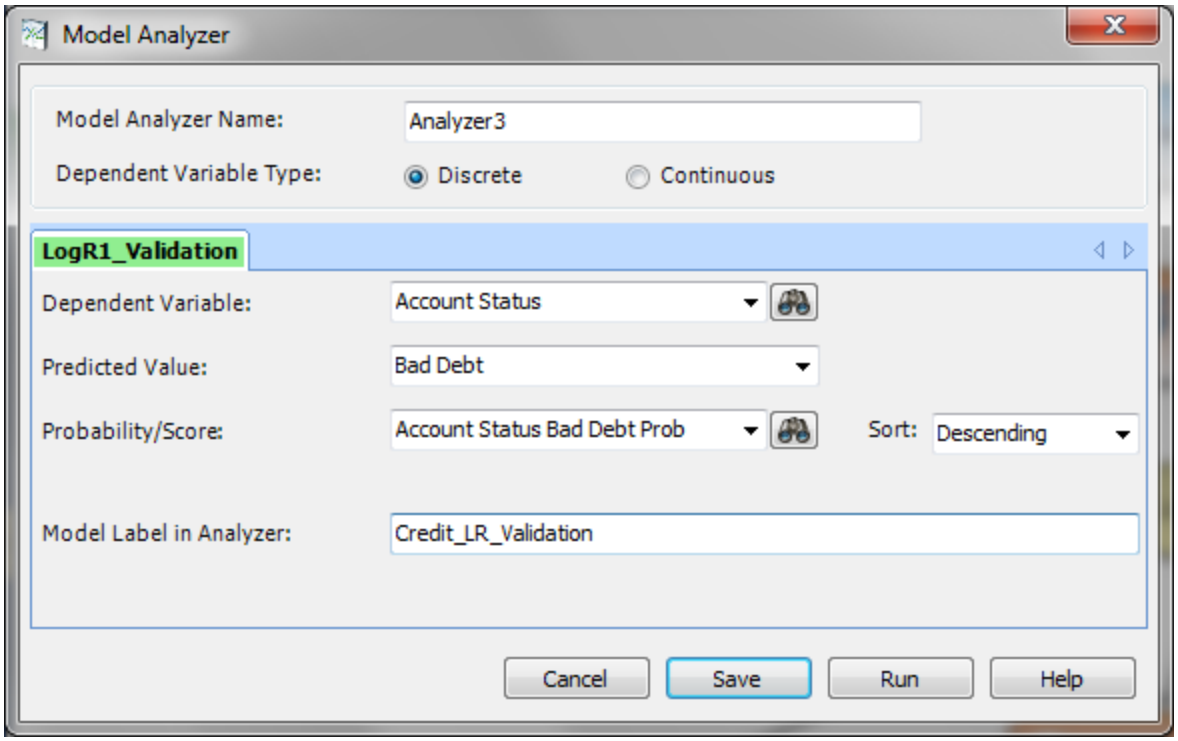
  OR

- Drag a connection arrow from the input dataset node to any area on the Workflow canvas background. From the context menu that appears when you release the mouse button, select **Model --> Model Analyzer** as in the example screenshot below. A new **Model Analyzer** node will be created and the wizard will open.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If the wizard does not open automatically, double-click on the Model Analyzer node or right-click on it and select **Modify** from the context menu.



Dependent Variable	The field containing the values of the dependent variable of the validated model.
Predicted Value	The target value predicted by the model that produced the input validation dataset.
Probability/Score	The field that contains the probability of the predicted value or the scores of a scorecard.

Choose the appropriate sorting depending on how the scores are sorted in your input dataset with respect to the target value (selected Predicted Value).

The typical structure of a dataset produced by the Model Validation node is as follows (only the fields relevant to the model prediction are shown):

Figure 1 - Validated model

Income	Income Prediction	Income Predict Prob	Income <=50K Prob	Income >50K Prob	Income Correct
<=50K	<=50K	0.99752662935081	0.99752662935081	0.00247337064919	☑
<=50K	<=50K	0.85896117393076	0.85896117393076	0.14103882606924	☑
>50K	<=50K	0.59785440122996	0.59785440122996	0.40214559877004	☐
>50K	>50K	0.816749257712649	0.18325074228735	0.81674925771264	☑
<=50K	<=50K	0.997994121929173	0.99799412192917	0.00200587807082	☑

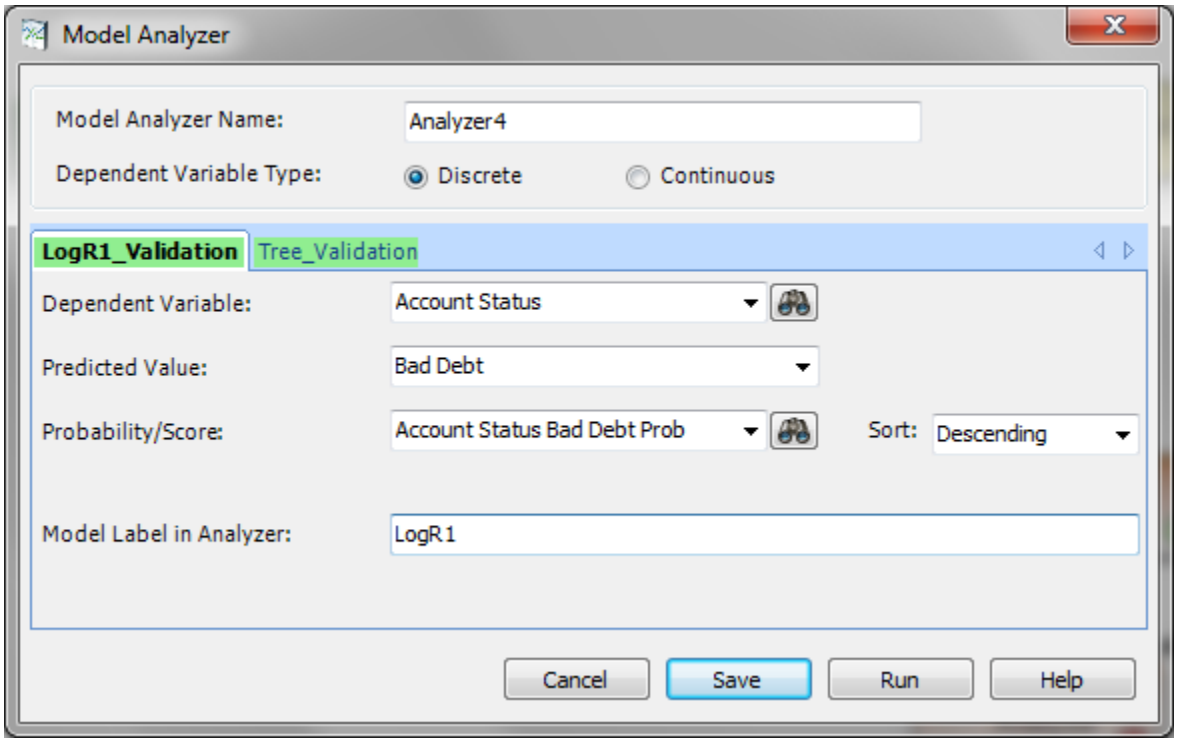
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

In this example, the DV is ***Income***. It has two categories, "***<=50K"*** and "***>50K"***. The ***Income Prediction*** field contains values predicted by the model (either "***<=50K"*** or ***">50K"***). The probability of the model predicting the value "**<=50**" or "**>50**" is displayed in the fields ***Income <=50 Prob*** and ***Income >50K Prob,*** respectively. The ***Income Predict Prob*** field contains the probability of the predicted value (i.e., the probability of the value in the ***Income Prediction*** field).

See also: Adding and Removing Models

You can add more than one dataset with model validation results to the model analyzer to compare their performance. To add another input dataset, connect it to the Model Analyzer node. Each input dataset is represented by a separate tab. Click on each tab to define the settings for each input..

In the example below, two input datasets - LogR1_Validation and Tree_Validation - resulting from the validation of two different models are represented by two tabs in the Model Analyzer.



To remove an input dataset from a Model Analyzer, just remove the connection from the dataset node to the analyzer node in the workflow by clicking on the connection arrow and pressing the Delete key.

Models Analyzer Views

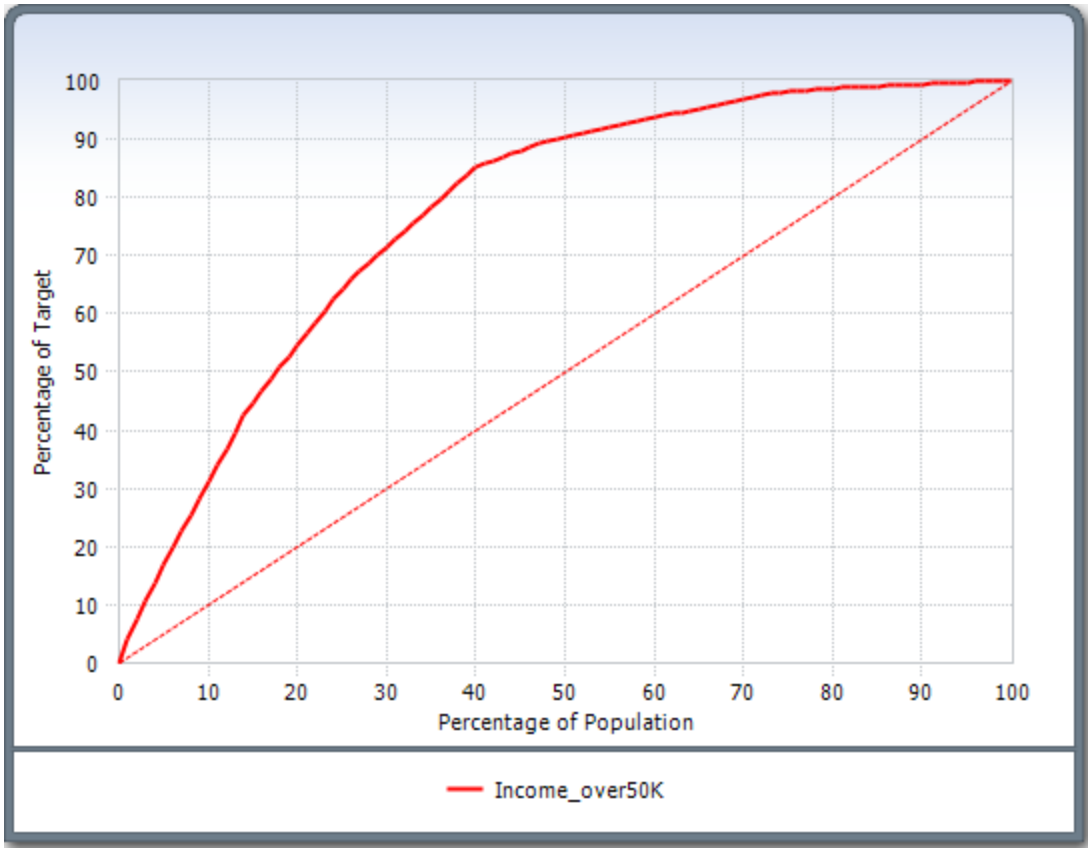
Cumulative Lift

A cumulative lift chart is a measurement tool that helps you analyze the effectiveness of your model. You would use this chart when you want to:

- determine if the model has any value to you

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- compare different models to determine which is the best
- see how the model behaves on new data



The cumulative lift is based on the probabilities of the category of interest plotted against the percentage of the total population.

Proceed to Calculating Cumulative Lift.

See also:

Cumulative Lift Report

The cumulative lift is based on the probabilities of the category of interest plotted against the percentage of the total population.

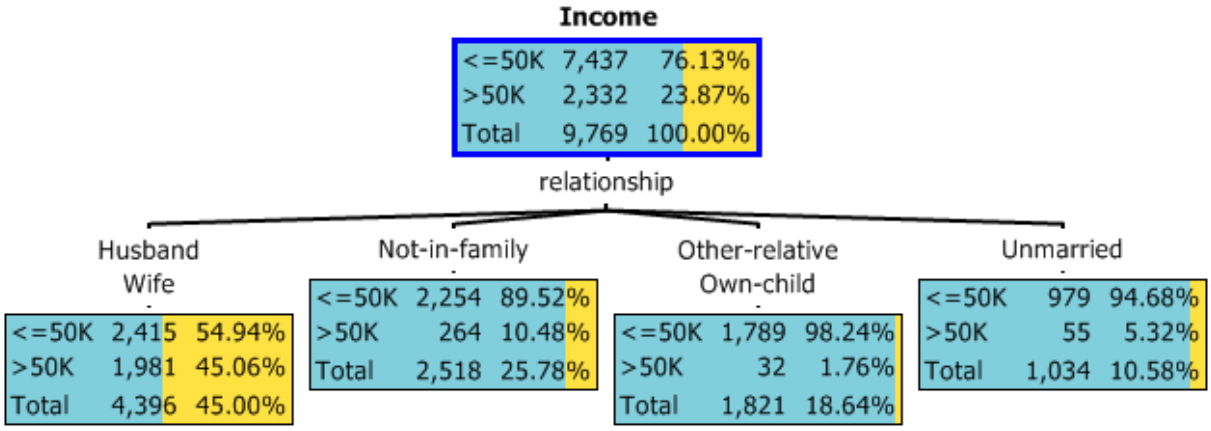
In our example, we insert a decision tree with DV=Income.

- Income has two categories, <=50K and >50K.
- We are interested in the category >50K.
- The total population is the total number of records used in the tree. In this example, we inserted a tree with 9769 records so our total population=9769.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Decision trees typically have more then one level and many more terminal nodes. To keep the example simple, we only grew the tree to one level beyond the root node. This yielded four terminal nodes.

Figure 1 - Decision Tree



The table below shows the normal and cumulative frequency and percentage of the category of interest (Income is ">50K") at each child node. The table is sorted by the probability of Income > 50K, which is the second column. Therefore, in terms of the node color, the child nodes are ordered by the size of their yellow portion, which indicates the percentage of >50K.

Table 1 - Cumulative Lift Calculation

Node rule	Probability	Total ">50" in Node	Cumulative Total >50K	Cumulative Lift (Y-axis)	Total Records in Node	Cumulative Total of Nodes	Cumulativ % of Population axis)
relationship = Husband, Wife	45.06	1981	1981	**84.95%**	4396	4396	45.0
relationship = Not-in-family	10.48	264	2245	**96.27%**	2518	6914	70.7
relationship = Unmarried	5.32	55	2300	**98.63%**	1034	7948	81.3
relationship = Other-relative, Own-child	1.76	32	2332	**100.00%**	1821	9769	100.0

Calculating the coordinates for the X-axis

The X-axis represents the cumulative percentage of the total population. Our total population is 9769 records. For each node, the calculation is as follows:

(Cumulative Total of Nodes / Total Population) * 100

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

For the first point representing the node [relationship = Husband, Wife], this is (4396 / 9769) * 100 = 0.4499949 * 100 = **45** (after rounding)

For the second point representing the first two nodes, this is (6914 / 9769) * 100 = 0.7077490 * 100 = **70.77** (after rounding)

Similar calculations yield **81.36%** for the first three child nodes, and obviously 100% for all four.

Calculating the coordinates for the Y-axis

The **Y-axis** represents the cumulative percentage of the records where the DV category of interest is **>50K**. The total number of records with Income >50K is 2332, as displayed in the root node. For each node, the calculation is as follows:

(Cumulative Total **>50K** / Total records with Income **>50K**) * 100

For the first point representing the node [relationship = Husband, Wife], the cumulative lift is (1981 / 2332) * 100 = 0.8494854 * 100 = **84.95** (after rounding)

For the second point representing the first two nodes, the cumulative lift is (2245 / 2332) * 100 = 0.9626929 * 100 = **96.27** (after rounding)

Similar calculations yield **98.63%** for the first three child nodes, and finally 100% for all four.

The Chart

From the above calculations, we have four points for the cumulative lift chart for the decision tree in Figure 1 above:

(45.00, 84.95)

(70.77, 96.27)

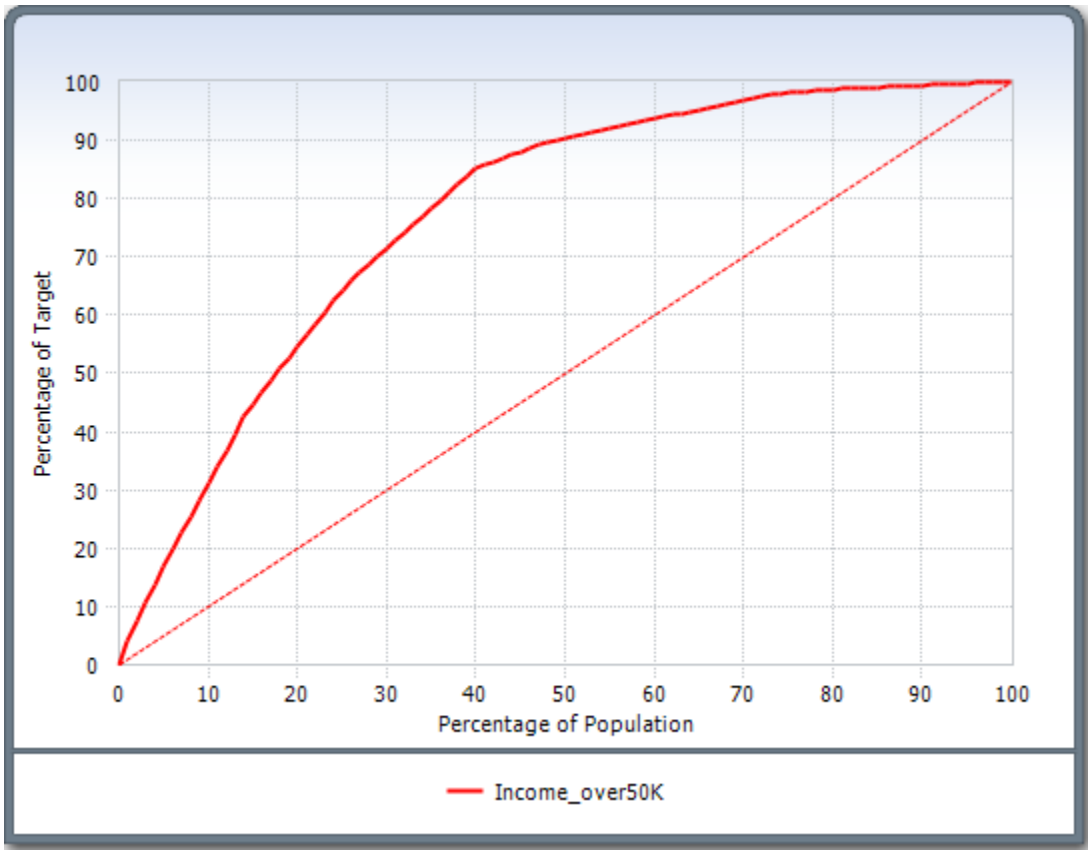
(81.36, 98.63)

(100, 100).

Plotting these points, we obtain the following chart:

Figure 2 - Cumulative Lift Chart

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Suppose our business objective is to create a mail campaign to target people who have an annual income >50K.

The chart shows that if we target approximately 45% of the population, we will reach 84.95% of the people with Income >50K.

By default, the y-axis shows the percentage of target values. You can choose to display absolute values instead of percentages by selecting **View | Show Absolute** from the main menu.

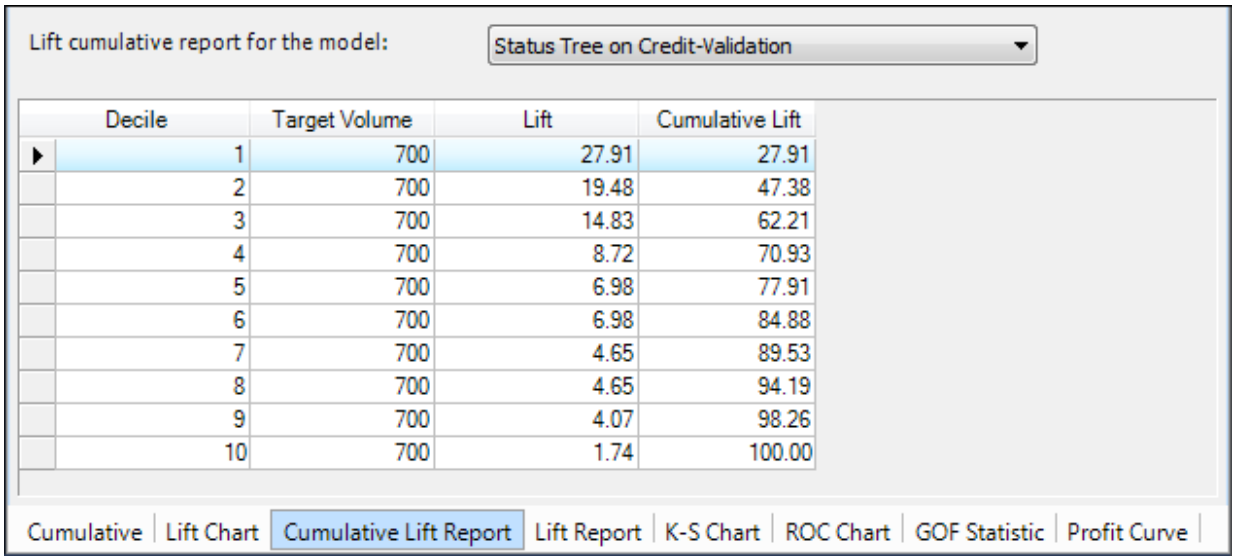
The *Cumulative Lift Report* tab shows the values of lift and cumulative lift at the decile level for each model in the Model Analyzer. If more than one model was added to the Model Analyzer, select the desired model label from the drop-down list at the top of the view. For example, in the screenshot below, the selected model label is "Status Tree on Credit-Validation".

See the topic Calculating Cumulative Lift for the detailed information on how to calculate lift and cumulative lift.

The *Target Volume* column shows the number of records in each decile of the validation dataset for the selected model.

The values in the *Cumulative Lift* column are obtained by adding up the values in the *Lift* column as you go down the deciles.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Decile	Target Volume	Lift	Cumulative Lift
1	700	27.91	27.91
2	700	19.48	47.38
3	700	14.83	62.21
4	700	8.72	70.93
5	700	6.98	77.91
6	700	6.98	84.88
7	700	4.65	89.53
8	700	4.65	94.19
9	700	4.07	98.26
10	700	1.74	100.00

See also:

Lift Report

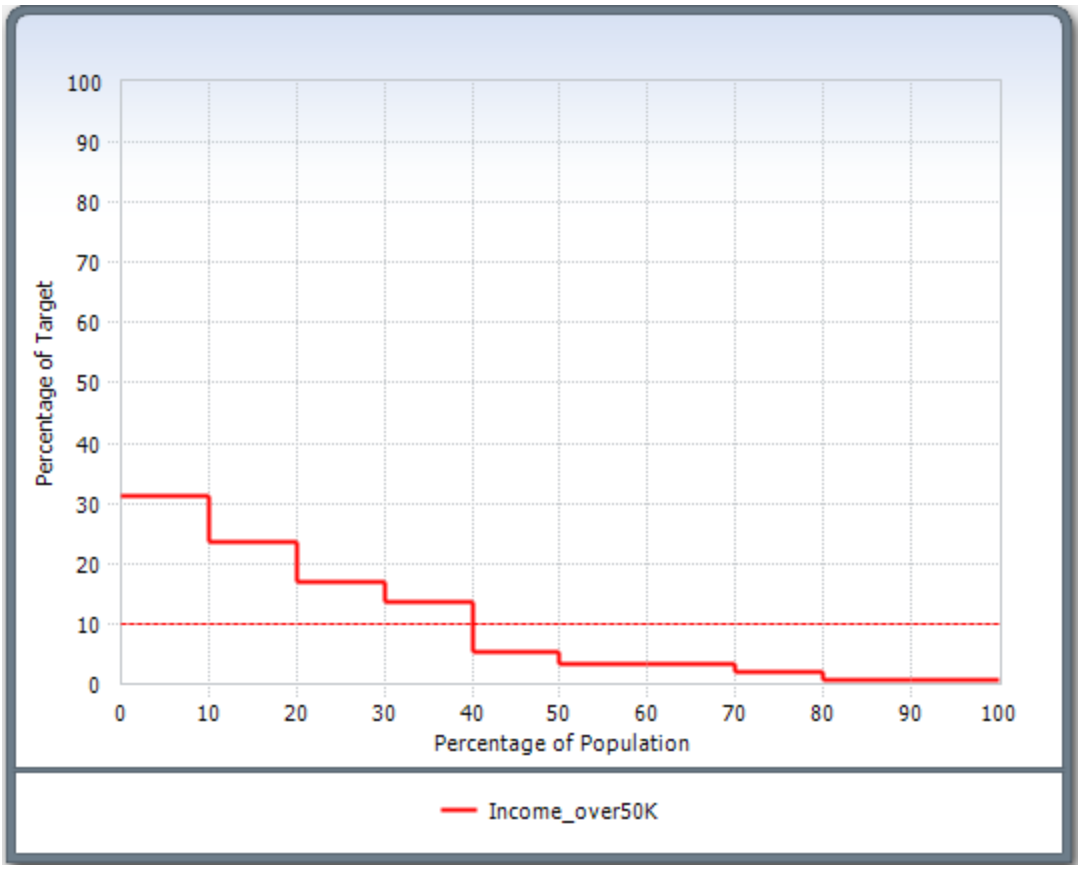
Cumulative Lift Chart

Lift Chart

Lift

Lift is a measure of the effectiveness of a model calculated as the ratio between the result obtained with the model and the result obtained without the model (i.e., by random record selection). The result obtained without a model is based on randomly selected records and is represented by the random curve in the lift chart (the horizontal line).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The x-axis of the Lift Chart shows the percentage of the total number of records sorted in descending order by the score, i.e., by the probability that the model assigns to a prediction of the selected value of the DV (the target value).

When we select a sample, we want to know **what portion of the total number of records with the target value is contained in the sample**. For example, if the sample coincides with the whole dataset, this portion is obviously equal to 100%; if we select half of the dataset so that all records with the target value turn out to be in our sample, the above portion is also 100%.

The Lift Chart shows the change in the percentage representing this portion when the model is used to select a sample from the general population. The size of the samples is fixed and is assumed to be 10%, which is the x-axis step. Thus, the y-coordinate of the Lift Chart shows the percentage of all records with the target value contained in the 10% sample.

If you select samples without using a model (i.e., random samples), the above portion of the target value will be approximately the same for all the samples. In particular, if you select a 10% random sample of the population you can expect that your samples will contain 10% of all records with target value of the DV. For this reason, the random curve in the lift chart is represented by the horizontal line y=10%.

For example, if your dataset contains 1000 records where the value of the Response field is "Yes" for 20% of the records and "No" for 80%, you would expect approximately the same 20 / 80 proportion of the responses in any random sample. So if you take a 10% random sample (100

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

records), about 20% of it (20 records) will have Response = "Yes", which amounts to 10% of all records with positive response in the dataset (200 records).

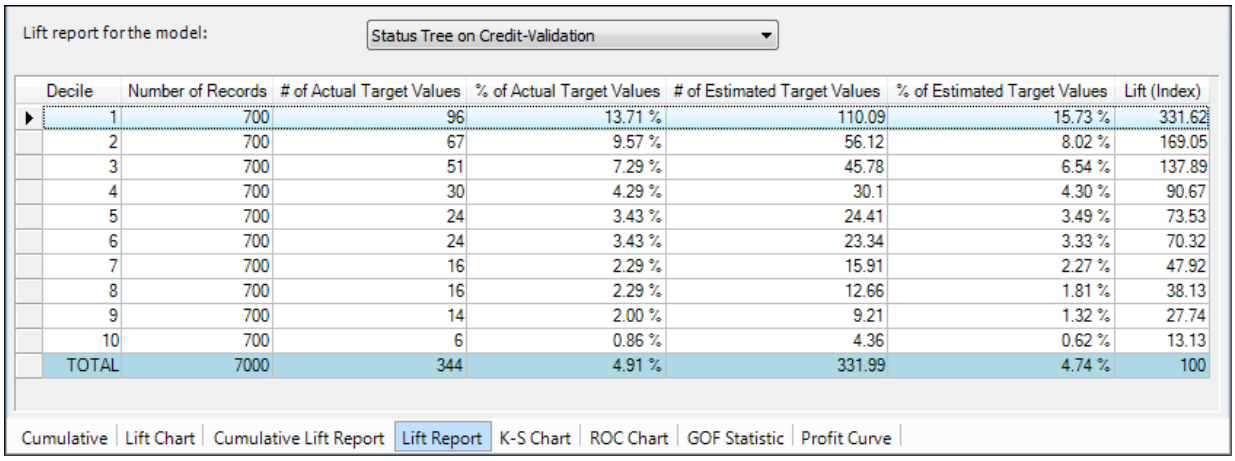
At the points where the model curve is below the random line, the lift factor is smaller than 1, which means that the records of the model contain fewer target values than a random sample with equally distributed target values.

By default, the y-axis shows the percentage of target values. You can specify to display absolute values instead of percentages by selecting **View | Show Absolute** from the main menu.

The *Lift Report* tab shows the values of lift at the decile level for each model in the Model Analyzer. If more than one model was added to the Model Analyzer, select the desired model label from the drop-down list at the top of the view. For example, in the screenshot below, the selected model label is "Status Tree on Credit-Validation".

See the topic Calculating Cumulative Lift for the detailed information on how to calculate lift and cumulative lift.

Target Value is the value of the dependent variable selected by the user when the model was added to the Model Analyzer. For example, if a Decision Tree model you added to the Model Analyzer has the dependent variable AccountStatus with two values: "Good" and "Bad Debt", and you chose "Bad Debt" when adding it to the Model Analyzer, then "Bad Debt" is the target value.



Decile	Number of Records	# of Actual Target Values	% of Actual Target Values	# of Estimated Target Values	% of Estimated Target Values	Lift (Index)
1	700	96	13.71 %	110.09	15.73 %	331.62
2	700	67	9.57 %	56.12	8.02 %	169.05
3	700	51	7.29 %	45.78	6.54 %	137.89
4	700	30	4.29 %	30.1	4.30 %	90.67
5	700	24	3.43 %	24.41	3.49 %	73.53
6	700	24	3.43 %	23.34	3.33 %	70.32
7	700	16	2.29 %	15.91	2.27 %	47.92
8	700	16	2.29 %	12.66	1.81 %	38.13
9	700	14	2.00 %	9.21	1.32 %	27.74
10	700	6	0.86 %	4.36	0.62 %	13.13
TOTAL	7000	344	4.91 %	331.99	4.74 %	100

Decile	Decile index
Number of Records	The number of records in each decile of the validation dataset for the selected model
# of Actual Target Values	The number of records with the target value of the dependent variable in the decile. For example, if the target value is "Bad Debt", that's the number of records with DV="Bad Debt" in the decile.
% of Actual Target Values	The percentage of records with the target value of the dependent variable in the decile. For example, if the target value is "Bad Debt", that's the percentage of records with DV="Bad Debt" in the decile.
# of Estimated Target Values	The estimated expected number of records where the value predicted by the model is the target value. This is

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

	the sum of the probabilities of the target values for each decile.
% of Estimated Target Values	The percentage of records where the value predicted by the model is the target value. Note: the number of estimated target values is calculated by taking the sum of the probabilities of the target values for each decile.
Lift (index)	Lift expressed as a ratio.

See also:

Cumulative Lift Report

Cumulative Lift Chart

Lift Chart

K-S Chart

K-S Chart

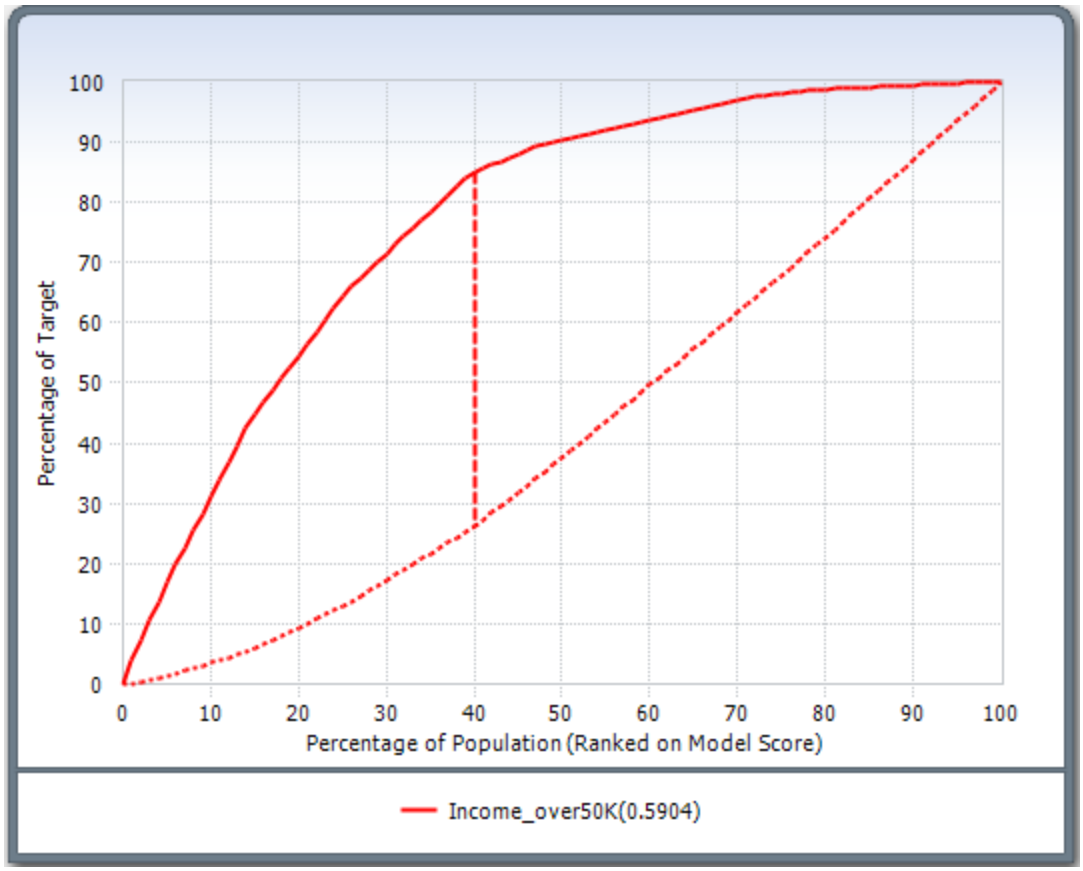
The K-S chart shows the maximum spread between two curves, where the first curve (the upper curve) represents the dependent variable (DV) category of interest and the second curve (the lower curve) represents the remaining dependent variable categories.

The upper curve is the cumulative lift of the DV category of interest and is identical to what is displayed in the Cumulative Lift Chart view. The vertical line between the two curves graphically shows the point where the maximum difference, or spread, occurs.

The calculated value at the point of maximum difference is the **K-S statistic**.

Figure 2 - K-S Chart

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

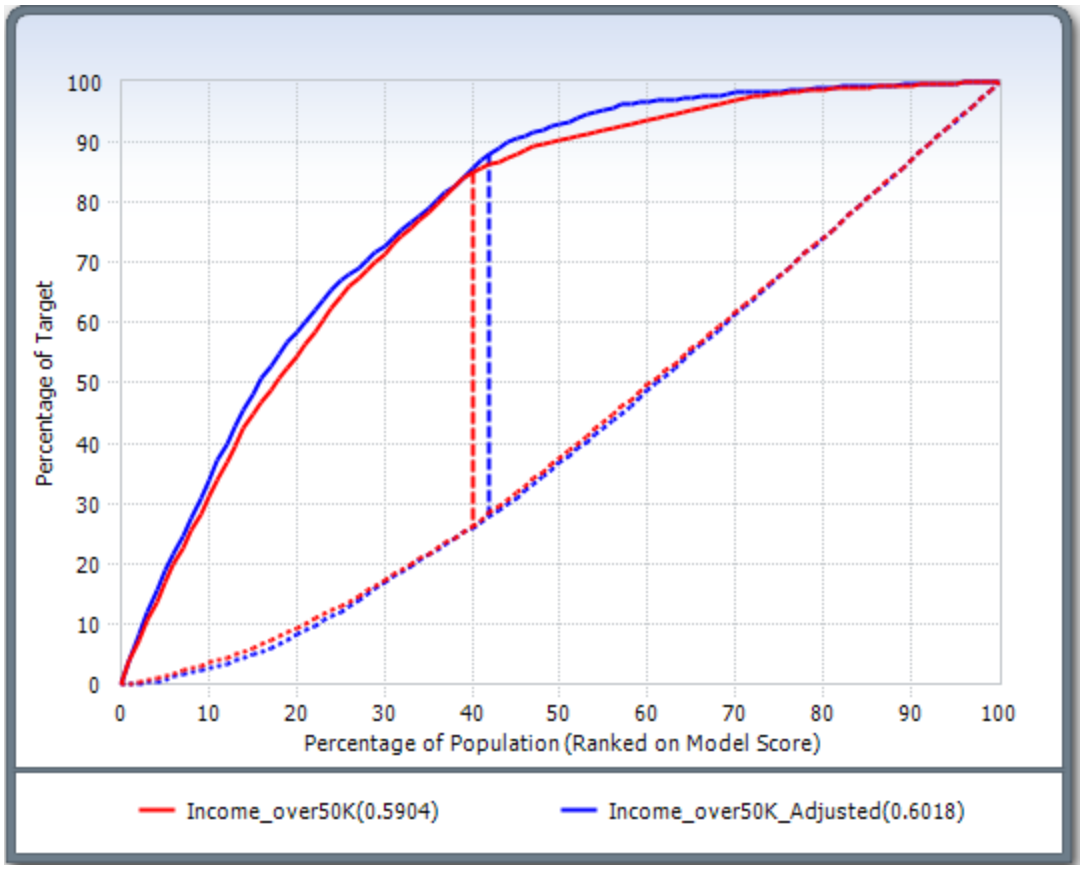


The K-S chart is used to compare different models to see which model is better.

In this example, "better" means "which model better distinguishes between high income earners (>50K) and low income earners (>=50K)".

Figure 2 - K-S Chart with two models

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The model with the maximum separation provides the better model to differentiate between high and low income earners. For this example, it would be better to use the model **Income_over50K_Adjusted** with a K-S statistic of 0.6018, because your chances of identifying high income earners is greater with this model.

However, for the **Income_over50K_Adjusted** model, the maximum separation is achieved at a higher percentage of population, which should also be taken into account.

ROC Chart

The ROC (Receiver Operating Characteristic) curve, also referred to as Relative Operating Characteristic, allows you to evaluate the model accuracy by estimating the sensitivity and specificity of a model in terms of the true positive rate and the false positive rate.

Suppose you have a model with a binary dependent variable, for example, AccountStatus = 'Bad Debt' or 'OK'. When validating the model using the **Tools | Validate** command, in one of the Validation Wizard pages you choose the columns to be created in the Validation output dataset. For example,

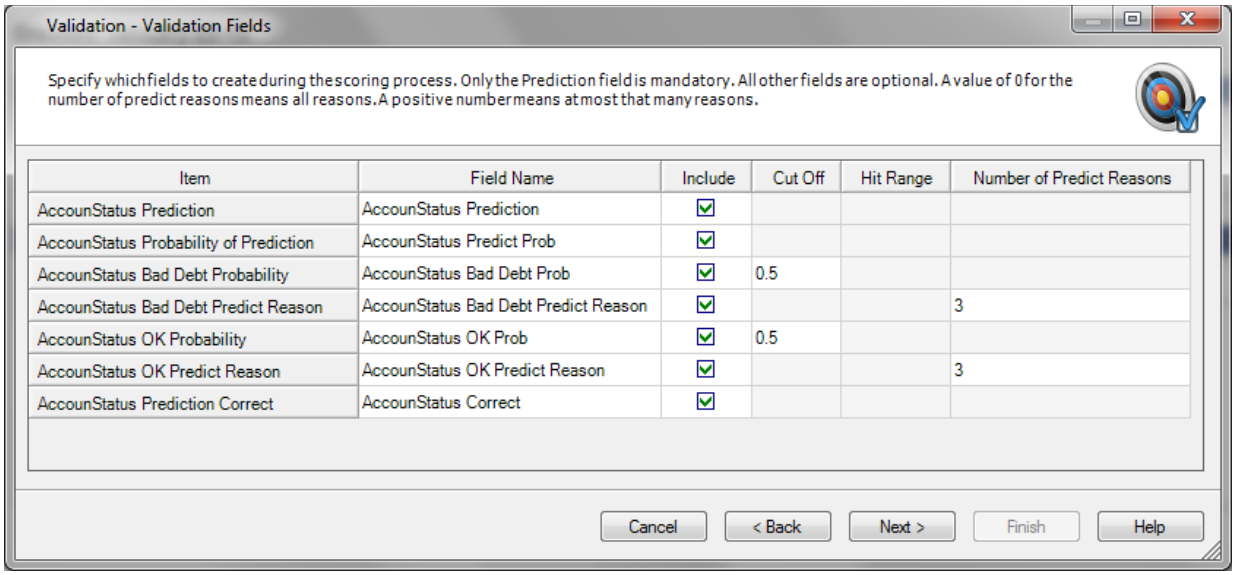
- Probability of AccountStatus = 'Bad Debt'
- Probability of AccountStatus = 'OK'
- Prediction
- Prediction Correct (Yes/No)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Also, in the same page you can specify a cut-off value for the probability of your outcome at which the Prediction value will be set. For example, if you set the cut-off for the Bad Debt Probability to 0.7 (70%), then the Prediction value will be 'Bad Debt' whenever Probability of AccountStatus = 'Bad Debt' is greater or equal to 0.7. At the same time, the probability cut-off for the other value ('OK') will be set to (1 - 0.7) = 0.3 (30%). Thus you are specifying a condition

IF PROBABILITY ('Bad Debt') >= 0.7 THEN PREDICTION = 'BAD DEBT' ELSE PREDICTION = 'OK'

The default cut-off value is 0.5 (50%).



You will get different validation results depending on the cut-off value. More precisely, the probabilities of Bad Debt and OK will be the same, but the Predictions, i.e., the way these probabilities are interpreted, will be different depending on the cut-off.

The Confusion Matrix is a result of model validation that shows how many of the predictions were correct and how many were wrong (did and did not coincide with actual DV values). It also shows how many actual 'Bads' were misinterpreted as 'OK' and vice versa. These rates obviously depend on the cut-off specified in the Validation wizard.

Example Confusion Matrix:

		Predicted value of AccountStatus	
		'Bad Debt'	'OK'
Actual value of AccountStatus	'Bad Debt'	2	417
	'OK'	2	8329

Here you can see that 'Bad Debt' is correctly predicted in 2 cases and misinterpreted as 'OK' in 417 cases. 'OK' is correctly predicted in 8329 cases and misinterpreted as 'Bad Debt' in 2 cases.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

These are the results with Cut-off = 0.5. If you re-validate the model with Cut-off = 0.3, the results are slightly better for ‘Bad Debt’ prediction accuracy:

		Predicted value of AccountStatus	
		'Bad Debt'	'OK'
Actual value of AccountStatus	'Bad Debt'	2	417
	'OK'	2	8329

The numbers in these cells are called True Negatives, True Positives (for correct predictions) and False Negatives and False Positives (for incorrect predictions)

		Predicted value	
		'Bad Debt'	'OK'
Actual value	'Bad Debt'	True Negatives (TN)	False Positives (FP)
	'OK'	False Negatives (FN)	True Positives (TP)

The ROC chart essentially shows the variation of the prediction accuracy for different cut-off values. The prediction accuracy is expressed by the following metrics in terms of TN, TP, FN, FP:

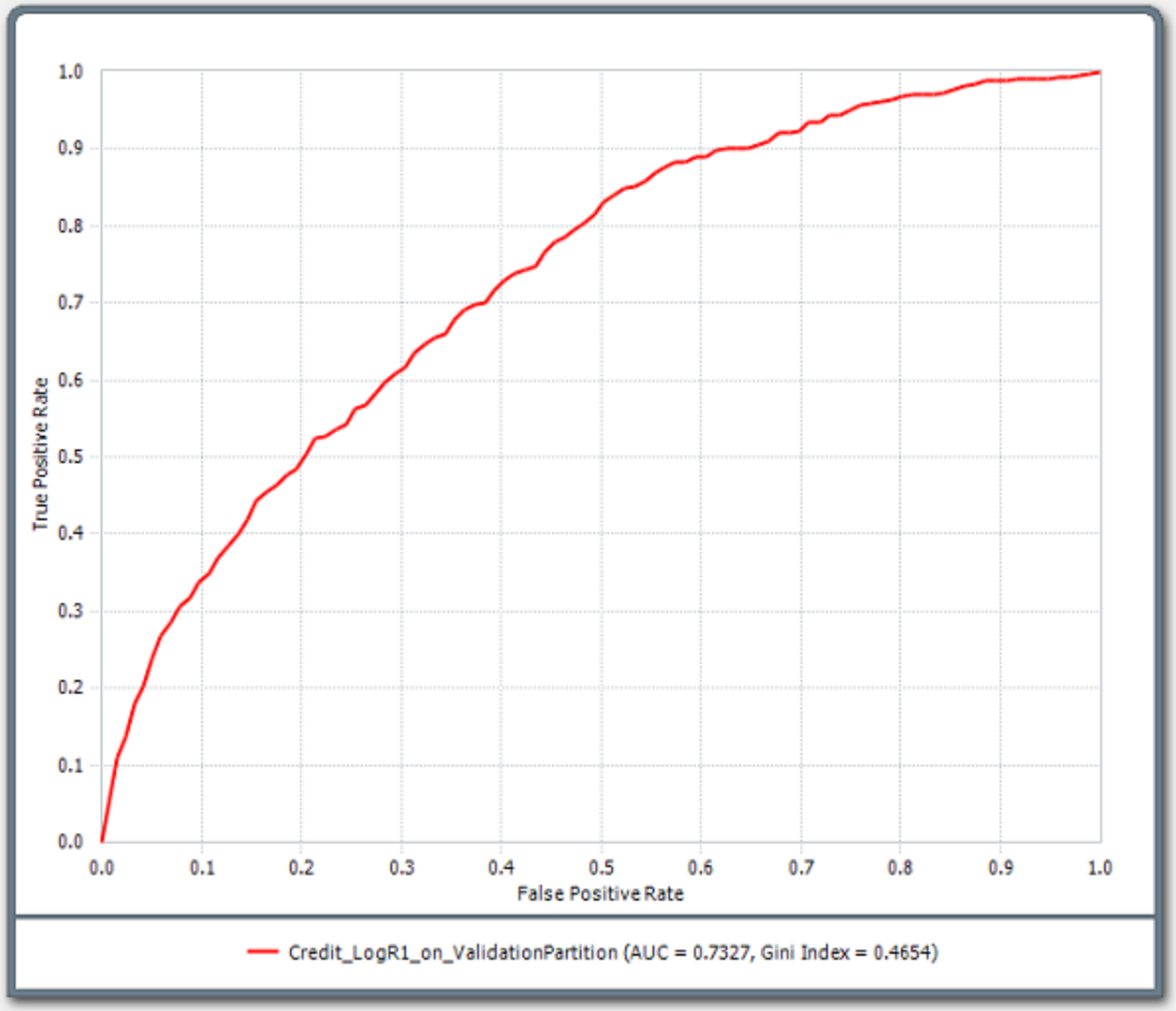
Sensitivity = True Positive Rate = TP / (TP + FN)

Specificity = TN / (TN + FP)

False Positive Rate = FPR = FP / (TN +FP) = 1 - Specificity

The ROC curve is created by plotting the True Positive Rate (TPR, or Sensitivity, on the vertical axis) versus the False Positive Rate (FPR, or 1 – Specificity, on the horizontal axis) while the cutoff used to construct the Confusion Matrix is varied from 0 to 1. Both TPR and FPR obviously vary within the interval [0,1]. See the example chart below.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The best case (a perfect classification) would occur for a point plotted at (0,1), i.e., the upper left corner, where the Sensitivity is the highest (=1) and the False Positive Rate is the lowest (=0). In general, it is better if a point is above the diagonal line (where the True Positive Rate is higher than the False Positive Rate) than below the diagonal line (where the False Positive Rate is higher). The higher the curve is above the diagonal, the better the model is.

AUC and Gini Index

The area under the ROC curve is known as the AUC statistic or the *c*-statistic.

AUC is related to the Gini index as follows: AUC = 0.5 (G+1), or G = 2*AUC - 1.

Goodness of Fit (GOF)

The GOF Statistic view of the Model Analyzer displays the results of the ***Hosmer-Lemeshow test*** for goodness of fit. It is applied to Logistic Regression models. The statistic, however, can also be calculated for decision trees and other models with binary dependent variable.

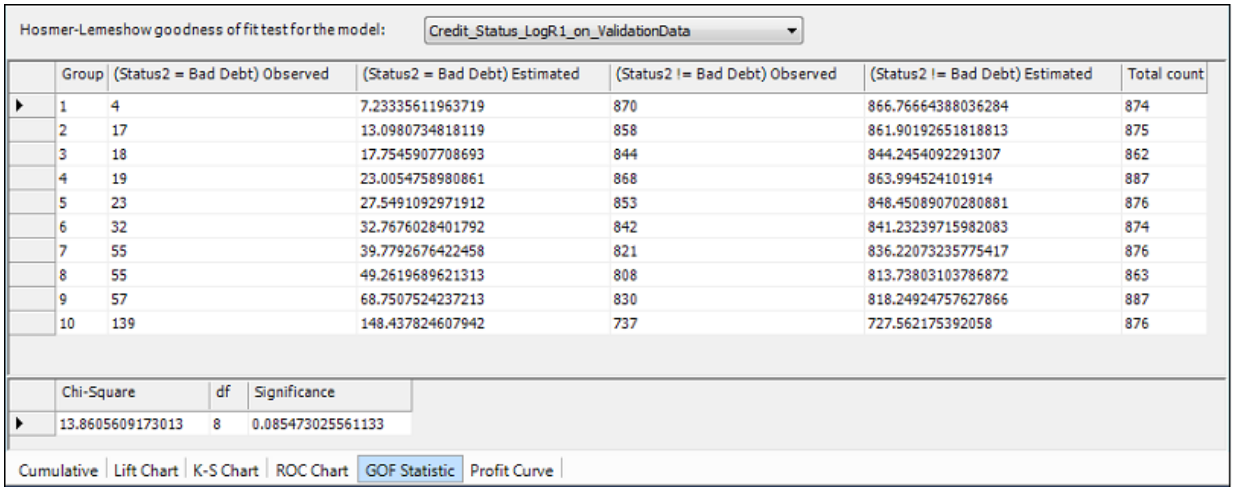
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The test compares the rates of the actual (*observed*) and predicted (*estimated*) target outcome in the subgroups of the validated dataset that are the deciles of the target outcome probabilities. The closer the predicted and observed outcome rates in subgroups, the better the model performance.

In addition, a statistic similar to Chi2 is calculated and its significance is evaluated using the Chi2 distribution.

If more than one model was added to the Model Analyzer, select the desired model from the list at the top of the view.

The following example illustrates the Goodness of Fit test for a Logistic Regression model with credit account status as the dependent variable. The target outcome is *Bad Debt.* (Note: Regression models are available only in Knowledge Studio).


Hosmer-Lemeshow goodness of fit test for the model: Credit_Status_LogR1_on_ValidationData

	Group	(Status2 = Bad Debt) Observed	(Status2 = Bad Debt) Estimated	(Status2 != Bad Debt) Observed	(Status2 != Bad Debt) Estimated	Total count
▶	1	4	7.23335611963719	870	866.76664388036284	874
	2	17	13.0980734818119	858	861.90192651818813	875
	3	18	17.7545907708693	844	844.2454092291307	862
	4	19	23.0054758980861	868	863.994524101914	887
	5	23	27.5491092971912	853	848.45089070280881	876
	6	32	32.7676028401792	842	841.23239715982083	874
	7	55	39.7792676422458	821	836.22073235775417	876
	8	55	49.2619689621313	808	813.73803103786872	863
	9	57	68.7507524237213	830	818.24924757627866	887
	10	139	148.437824607942	737	727.562175392058	876

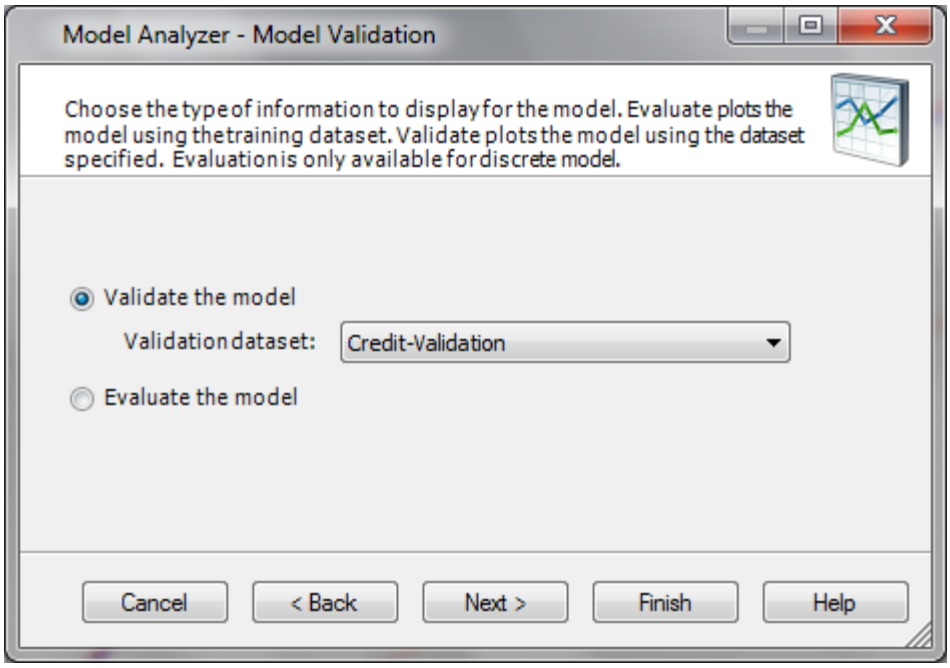
	Chi-Square	df	Significance
▶	13.8605609173013	8	0.085473025561133

Cumulative | Lift Chart | K-S Chart | ROC Chart | GOF Statistic | Profit Curve

Calculation of Target Outcome Rates for the Goodness of Fit Test

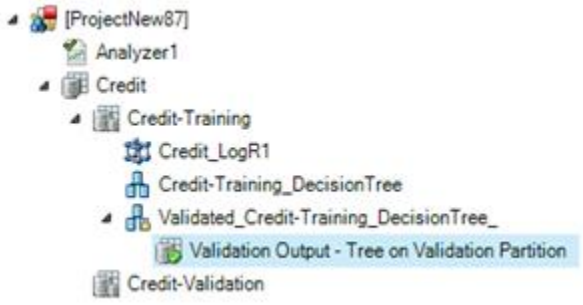
First, the Model Analyzer validates the model on the dataset specified by the user in the Model Validation step of the wizard:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



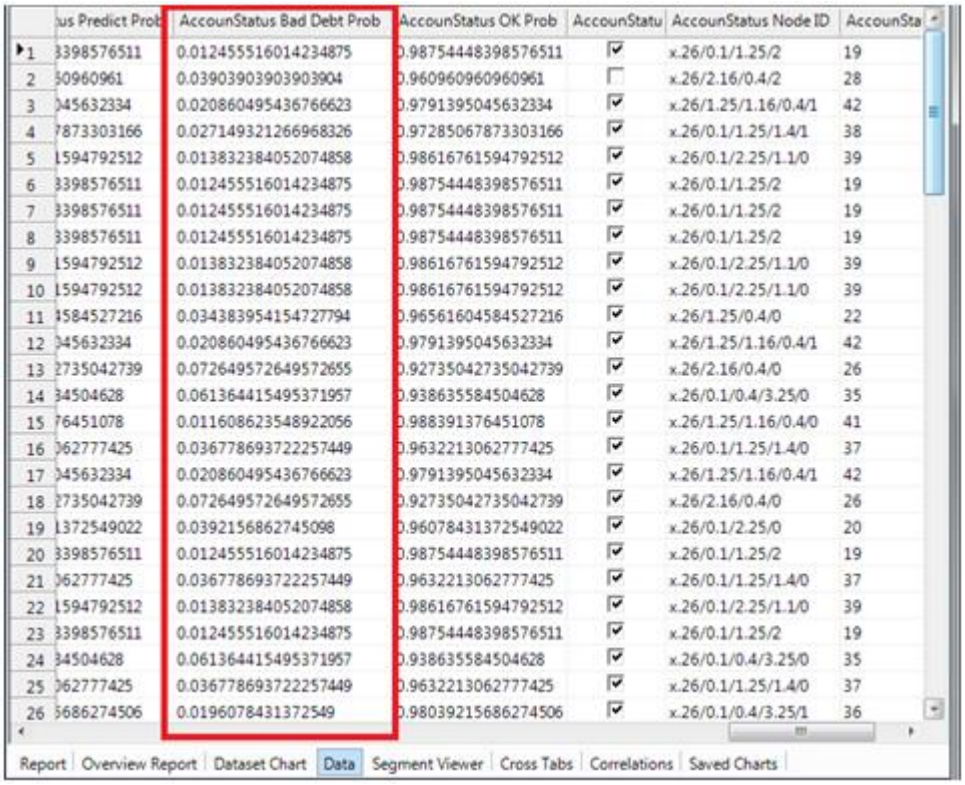
The dataset resulting from the model validation contains the column with the probability of the target outcome (for example, probability of Account Status = 'Bad Debt').

Note that Model Analyzer does validation in the background and does not explicitly show the output dataset in the project. To see the validation output dataset, you can use the menu command **Tools | Validate**. Open your model, select **Tools | Validate** from the menu, choose the input dataset for validation, and click Next until the Finish button is enabled, then click Finish. This will produce the validation output dataset in your project. In the example below, it is the object highlighted in blue - the child object of the validated tree model copy:



You will see the Validation report with the Confusion Matrix and other validation statistics. Click on the Data tab at the bottom. In the Data tab, scroll to the right to see the probability columns and find the probability of your target outcome, for example, Account Status = 'Bad Debt'

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



	us Predict Prob	AccounStatus Bad Debt Prob	AccounStatus OK Prob	AccounStatu	AccounStatus Node ID	AccounSta
1	3398576511	0.012455516014234875	0.9875444839857651 1	☑	x.26/0.1/1.25/2	19
2	60960961	0.03903903903903904	0.960960960960961	☐	x.26/2.16/0.4/2	28
3	045632334	0.020860495436766623	0.9791395045632334	☑	x.26/1.25/1.16/0.4/1	42
4	7873303166	0.027149321266968326	0.97285067873303166	☑	x.26/0.1/1.25/1.4/1	38
5	1594792512	0.013832384052074858	0.98616761594792512	☑	x.26/0.1/2.25/1.1/0	39
6	3398576511	0.012455516014234875	0.9875444839857651 1	☑	x.26/0.1/1.25/2	19
7	3398576511	0.012455516014234875	0.9875444839857651 1	☑	x.26/0.1/1.25/2	19
8	3398576511	0.012455516014234875	0.9875444839857651 1	☑	x.26/0.1/1.25/2	19
9	1594792512	0.013832384052074858	0.98616761594792512	☑	x.26/0.1/2.25/1.1/0	39
10	1594792512	0.013832384052074858	0.98616761594792512	☑	x.26/0.1/2.25/1.1/0	39
11	3584527216	0.034383954154727794	0.96561604584527216	☑	x.26/1.25/0.4/0	22
12	045632334	0.020860495436766623	0.9791395045632334	☑	x.26/1.25/1.16/0.4/1	42
13	2735042739	0.07264957264957265 5	0.92735042735042739	☑	x.26/2.16/0.4/0	26
14	84504628	0.061364415495371957	0.938635584504628	☑	x.26/0.1/0.4/3.25/0	35
15	76451078	0.011608623548922056	0.988391376451078	☑	x.26/1.25/1.16/0.4/0	41
16	062777425	0.036778693722257449	0.9632213062777425	☑	x.26/0.1/1.25/1.4/0	37
17	045632334	0.020860495436766623	0.9791395045632334	☑	x.26/1.25/1.16/0.4/1	42
18	2735042739	0.07264957264957265 5	0.92735042735042739	☑	x.26/2.16/0.4/0	26
19	1372549022	0.0392156862745098	0.96078431372549022	☑	x.26/0.1/2.25/0	20
20	3398576511	0.012455516014234875	0.9875444839857651 1	☑	x.26/0.1/1.25/2	19
21	062777425	0.036778693722257449	0.9632213062777425	☑	x.26/0.1/1.25/1.4/0	37
22	1594792512	0.013832384052074858	0.98616761594792512	☑	x.26/0.1/2.25/1.1/0	39
23	3398576511	0.012455516014234875	0.9875444839857651 1	☑	x.26/0.1/1.25/2	19
24	84504628	0.061364415495371957	0.938635584504628	☑	x.26/0.1/0.4/3.25/0	35
25	062777425	0.036778693722257449	0.9632213062777425	☑	x.26/0.1/1.25/1.4/0	37
26	6686274506	0.01960784313725 49	0.98039215686274506	☑	x.26/0.1/0.4/3.25/1	36

Report | Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Correlations | Saved Charts

In the Hosmer-Lemeshow test, first, the dataset resulting from the model validation is sorted into 10 bins by the predicted probability of target outcome (e.g., Account Status = 'Bad Debt'), in increasing order. *Note that the number of bins may be smaller than 10 if there are not enough distinct values of the target probability*. Then the average predicted probability of the target outcome is calculated in each bin, as well as the number of actual target outcomes in each bin. The number of bins ***g*** depends on the total number of records in the input dataset and the number of different data patterns in the independent variables in the input dataset. The number of groups is determined in the same way as in the Range Editor of the Force Split command performed on a continuous variable with the *Equal Height* option.

Then the Hosmer-Lemeshow statistic is defined as

$$\sum_{i=1}^{g} \frac{(O_i - N_i \bar{p}_i)^2}{N_i \bar{p}_i (1 - \bar{p}_i)}$$

where

O_i is the frequency of the target outcome in the ith bin;

N_i is the total number of records in the ith bin;

$\bar{p}_i$ is the average predicted probability of the target outcome in the ith bin.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

In the table such as the one shown below, the *Observed* column shows O_i, while the *Estimated* column shows $N_i * \bar{p}_i$

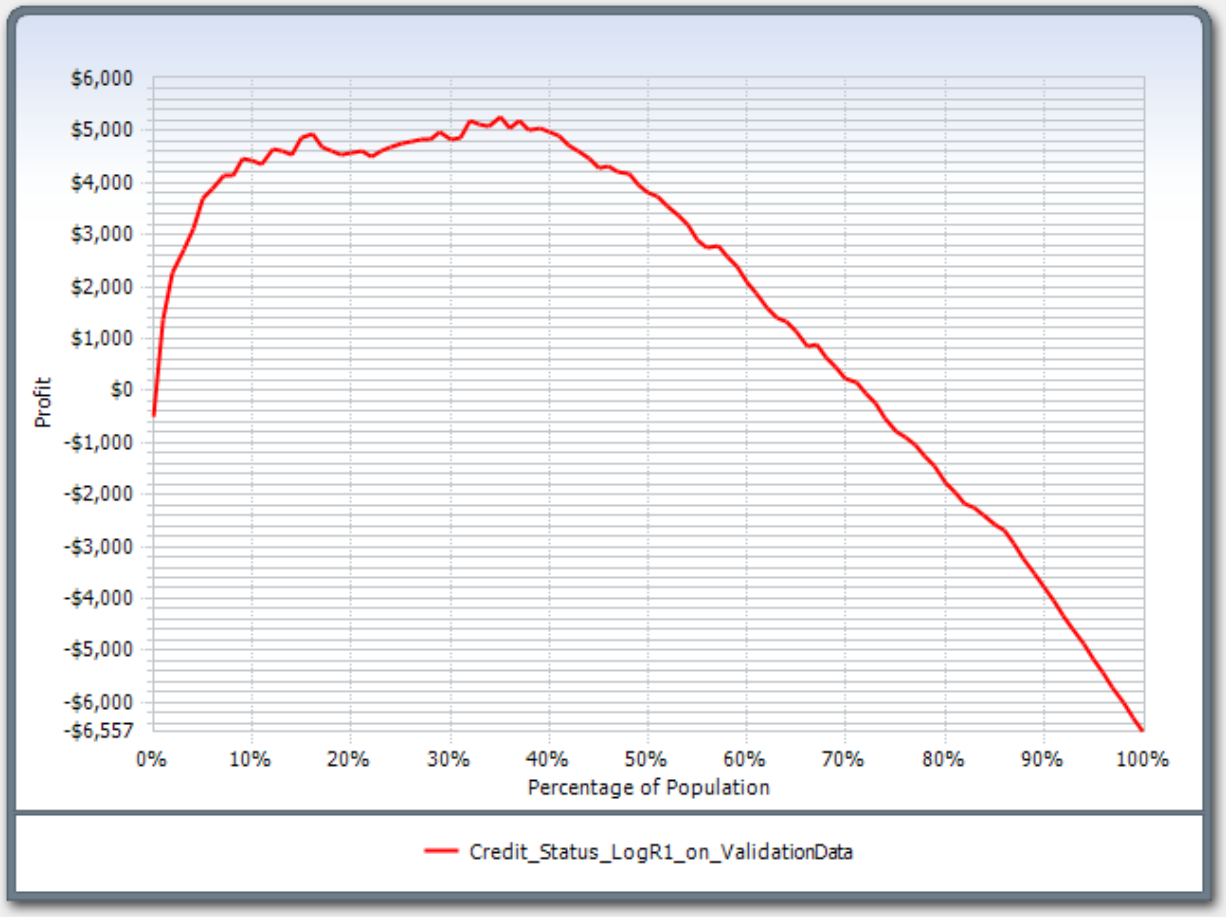
The Significance of the statistic is the *p*-value calculated for a Chi2 distribution with (*g* - 2) degrees of freedom, where *g* is the number of groups.

For more information, see Hosmer, D.W. and Lemeshow, S. (2000) *Applied Logistic Regression*, *2nd Edition*, J. Wiley & Sons, New York.

Profit Curve

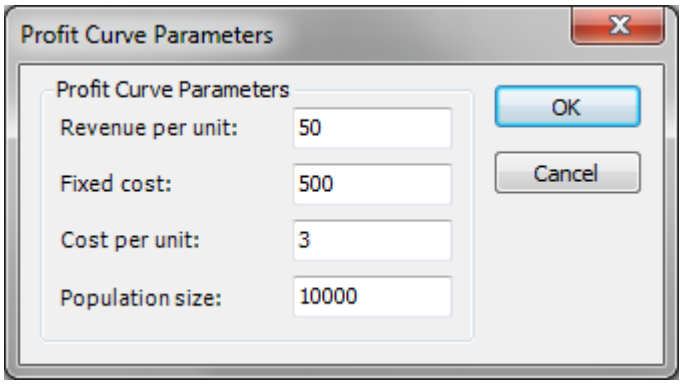
The **Profit Curve** is a view within the model analyzer that provides a quick Return-on-Investment (ROI) calculation based model performance (cumulative lift chart) and business specific costs and expected returns.

Figure 1



The first time the Profit Curve tab is used, the options dialog appears automatically. Enter the parameter values.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Revenue per unit	revenue generated for each response.
Fixed cost	fixed cost for the campaign
Cost per unit	associated cost for each response
Population size	the population involved in the campaign.

To invoke this dialog at point after the first profit curve is added,

From the menu

Tools | Options...

From the toolbar

Click on the Options icon

Model Selector

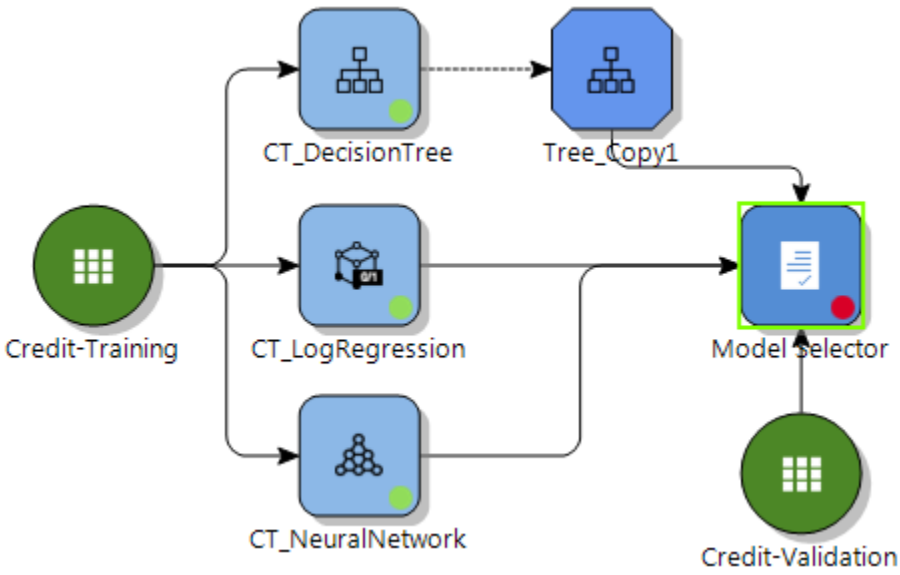
The **Model Selector** node provides the ability to select the best model based on one of the available model performance metrics. Prerequisites: two or more predictive models and a dataset for model validation.

To automatically select the best model from several models on your workflow, drag the **Model Selector** node from the **Evaluate** palette onto the canvas and connect the required input models to the node. Connect the validation dataset to Model Selector node.

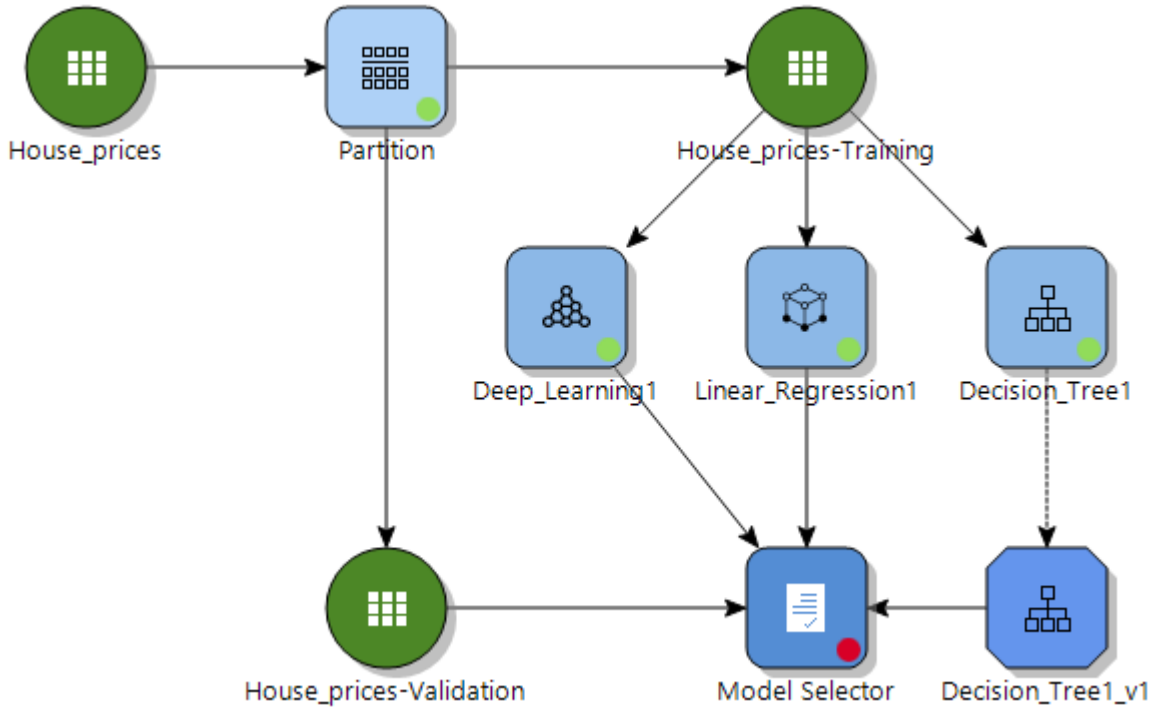
- All input models must have compatible model schemas: they should have the same dependent variable, and its data type must be the same in all models.
- The input validation dataset must contain all variables required by the input models, and their data types must match those of the model variables.

Here is an example of a workflow with three models, which include a Decision Tree. Note: When connecting a Decision Tree node to Model Selector, the tree model instance is created automatically.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



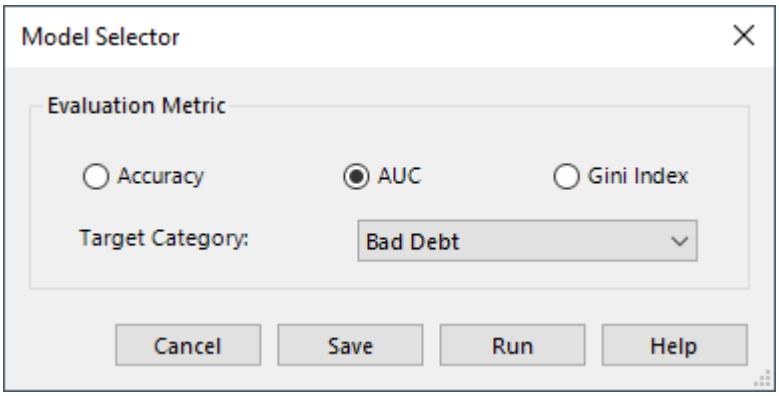
An example of a workflow with three models with continuous dependent variable, including linear regression:



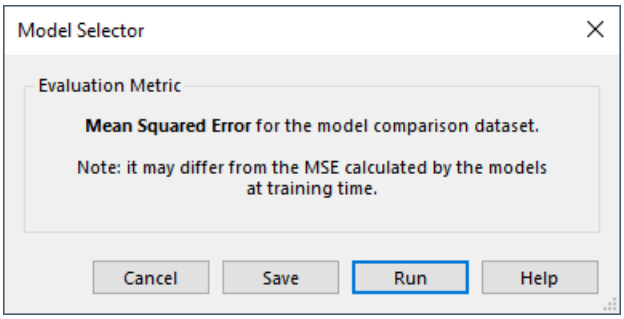
Double-click the Model Selector node or right-click on it and select **Modify**. Choose the desired metric as the criterion to select the best model.

- If the dependent variable is discrete, the evaluation metric options are: *Accuracy*, *AUC* (*the area under the ROC curve*), and *Gini Index*.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- If the dependent variable is continuous, the evaluation metric is Mean Squared Error (MSE). This is the mean of the squared errors calculated on the input dataset used for model validation and comparison. This MSE value may differ from the MSE calculated by the input models at training time.

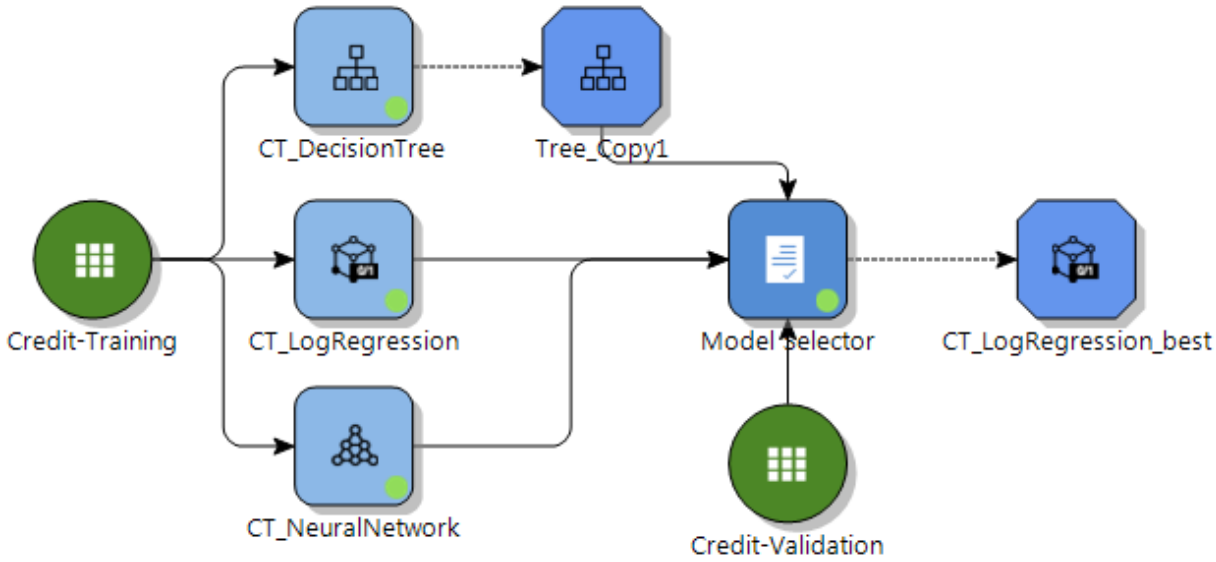


Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.

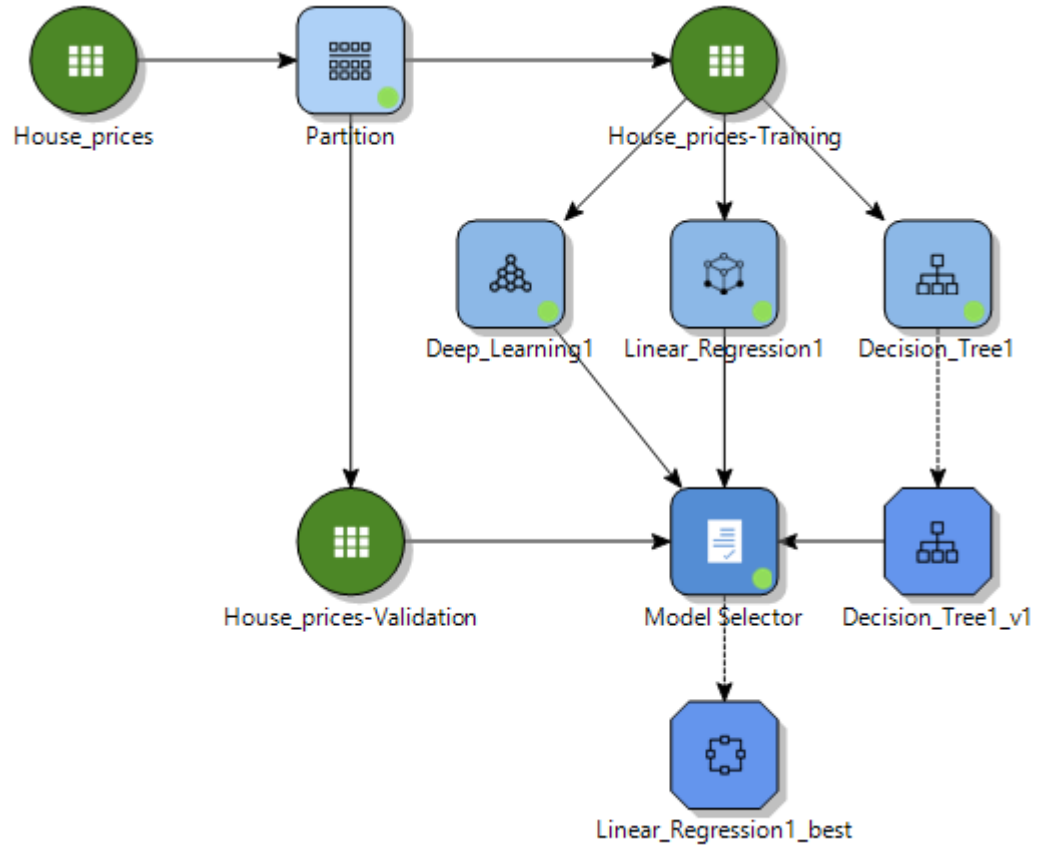
For every model connected to the node, the attached dataset will be scored with that model, and the appropriate evaluation metric will be calculated. For the selected evaluation metric, the node will determine the model with the best value for that statistic (the highest Accuracy, AUC, or Gini in the discrete case; the lowest MSE in the continuous case). The output is a copy of the trained model node that has the best value for that statistic.

In the example below, the best model is the logistic regression model:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



In the example below, the best model is the linear regression model:

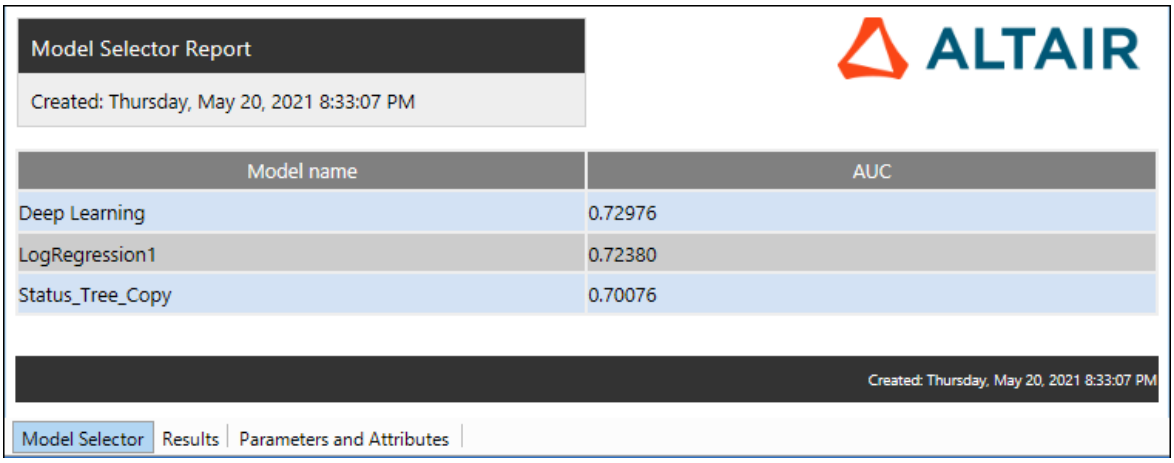


Double-click the output model node labeled as the best one.

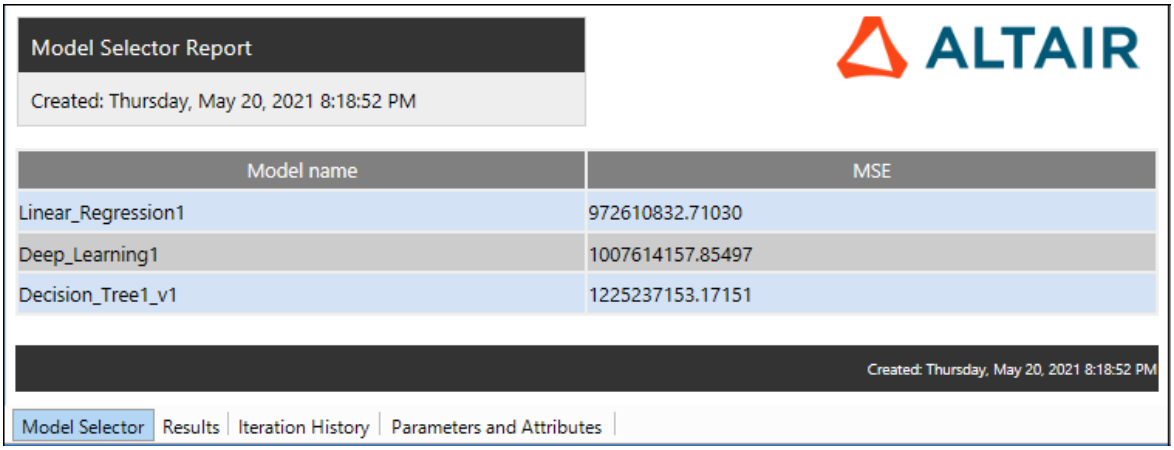
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The first tab is the **Model Selector** Report, which lists the values of the selected evaluation metric for all input models. The tabs **Results** and **Parameters and Attributes** show the model training results and model attributes of the best model.

An example of the Model Selector report in the case of a discrete dependent variable:



An example of the Model Selector report in the case of a continuous dependent variable:



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Model Deployment

Scoring

For Decision Trees and Predictive Models, scoring is the process of predicting the values of the dependent variable (DV) for records in a new dataset. Scoring is usually done after completing the validation process, in which you assess how well your model performs on test data.

The **score** is the probability that the predicted outcome will occur. In contrast to validation, in scoring the target dataset is assumed to have no DV field, the correct outcome is not known, so no comparison of actual vs. predicted outcome takes place.

For Cluster Analysis models, scoring is assigning a record to a cluster, computing the probability for each record to belong to a cluster, and computing distances to the corresponding cluster center.

For a Market Basket Analysis model, scoring produces *recommendations*. This type of scoring is covered in a separate topic Scoring a Market Basket Analysis Model.

For a Principal Component Analysis, scoring produces *principal components.* This type of scoring is covered in a separate topic Scoring a Principal Component Analysis Model.

Scoring Fields

During the scoring process, additional fields are created. These fields can be either part of a newly generated dataset, appended to the target dataset, or exported to an external file or database table. Not all of the generated fields are mandatory. During the scoring wizard setup, you choose which additional fields you want to include.

Models with discrete DV produce fields different from those produced by models with continuous DV. The scoring of Cluster Analysis models produces fields specific for clustering (Cluster Index, Cluster Probability, Distance to Center).

Scoring Fields for Models with Discrete DV

Figure 1

Income Prediction	Income <=50	Income >50K Prob	Income Predict Prob
<=50K	1	0	1
>50K	0	1	1
>50K	0.090909090	0.90909090909090	0.909090909090909
<=50K	1	0	1
<=50K	0.701492537	0.29850746268656	0.701492537313433
<=50K	0.713414634	0.28658536585365	0.713414634146341
>50K	0.271186440	0.72881355932203	0.728813559322034

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The score is the probability of the DV value of interest for a given record. For example, if you are interested in the records where the value ">50K" is predicted for your dependent variable *Income*, the values in the field ***Income >50K Prob*** represent the scores.

The scores are compared with the probability **cut off** value (specified in the scoring wizard) which determines the predicted value in the ***DV Prediction*** field (***Income Prediction*** in this example). If the cut off was defined as **0.5** (default), then any score value on or above **0.5** will result in the predicted value "**>50K**" and a score value below **0.5** will result in the predicted value "**<=50K**".

Income Prediction	The predicted value for a given record.
Income <=50K Prob **and** ***Income >50K Prob***	The probability of the specified outcome (value). For example, the field *Income >50K Prob* contains the probability of the value "*>50K*". Usually, the probability of the value of interest is the only field necessary for the analysis. The probabilities of other DV values are optional.
Income Predict Probability	The probability of the predicted value (i.e., of the value in the Income Prediction field in this case).

Scoring Fields for Models with Continuous DV

In Figure 2, the continuous DV ***age*** is predicted. In the continuous case, the **score** is the value in the ***DV Prediction*** field (***age Prediction*** in this example).

Figure 2

age Prediction	age Std Dev	age Node ID
42.569767441	13.975222357	x.5/0.12/3
54.755208333	18.364188744	x.5/2
54.755208333	18.364188744	x.5/2
54.755208333	18.364188744	x.5/2
42.569767441	13.975222357	x.5/0.12/3
42.569767441	13.975222357	x.5/0.12/3

age Prediction	The predicted value for a given record.
age Std Dev	The standard deviation of the DV can be used as the estimated precision of the prediction as estimated by the model.
age Node ID	For Decision Tree models only, this is the node ID of the tree node to which the record is assigned.

Special Notes on Scoring Decision Trees

Scoring a decision tree is a little different from scoring other models due to the interactive character of trees and the fact that the scoring fields are calculated columns depending on the validated models.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If you validate a tree model, the calculated columns created with the scoring operation refer to a read-only copy of the tree being validated, not to the original tree. This read-only version of the original tree is created automatically by the scoring process and is called a "*snapshot tree*". This prevents a variety of problems that can occur if a referenced model is changed after scoring (especially if scoring fields were appended to the parent dataset) and allows the user to delete and modify trees without breaking any dependencies. In addition, the user always has a copy of a tree as it was at the time of scoring.

In-Database Scoring

Scoring can be performed within a database if your product includes the In-Database Analytics feature. The purpose of In-Database scoring is to execute the model scoring commands directly on the database server and store the resulting scores there. The Altair Data Mining Engine sends the model scoring commands to the database engine, which reads the records from the input table and writes the results to a new table. No data extraction occurs in this process, and the scoring is performed by the database engine rather than the Altair Data Mining Engine.

Proceed to Scoring Wizard.

Scoring can be performed for all types of models - Decision Trees, Regression Models, Cluster Analysis, Scorecards, Market Basket Analysis, Keras Deep Learning, etc. See the topic About Scoring for an overview.

For interactive models - Decision Trees, Strategy Trees, and Market Basket Analysis - scoring is applied to model instances, since it is important to know what the state of the model was at the time of scoring. If you apply scoring to a tree *process* node or a market basket analysis process node, a model instance will be automatically saved before scoring, and the scoring will be applied to the instance.

Note: The context and meaning of Strategy Tree scoring is different from those of other models. See the topic Scoring a Strategy for details. The context and meaning of scoring for Market Basket Analysis is different from those of other models and is therefore covered in a separate topic Scoring a Market Basket Analysis Model.

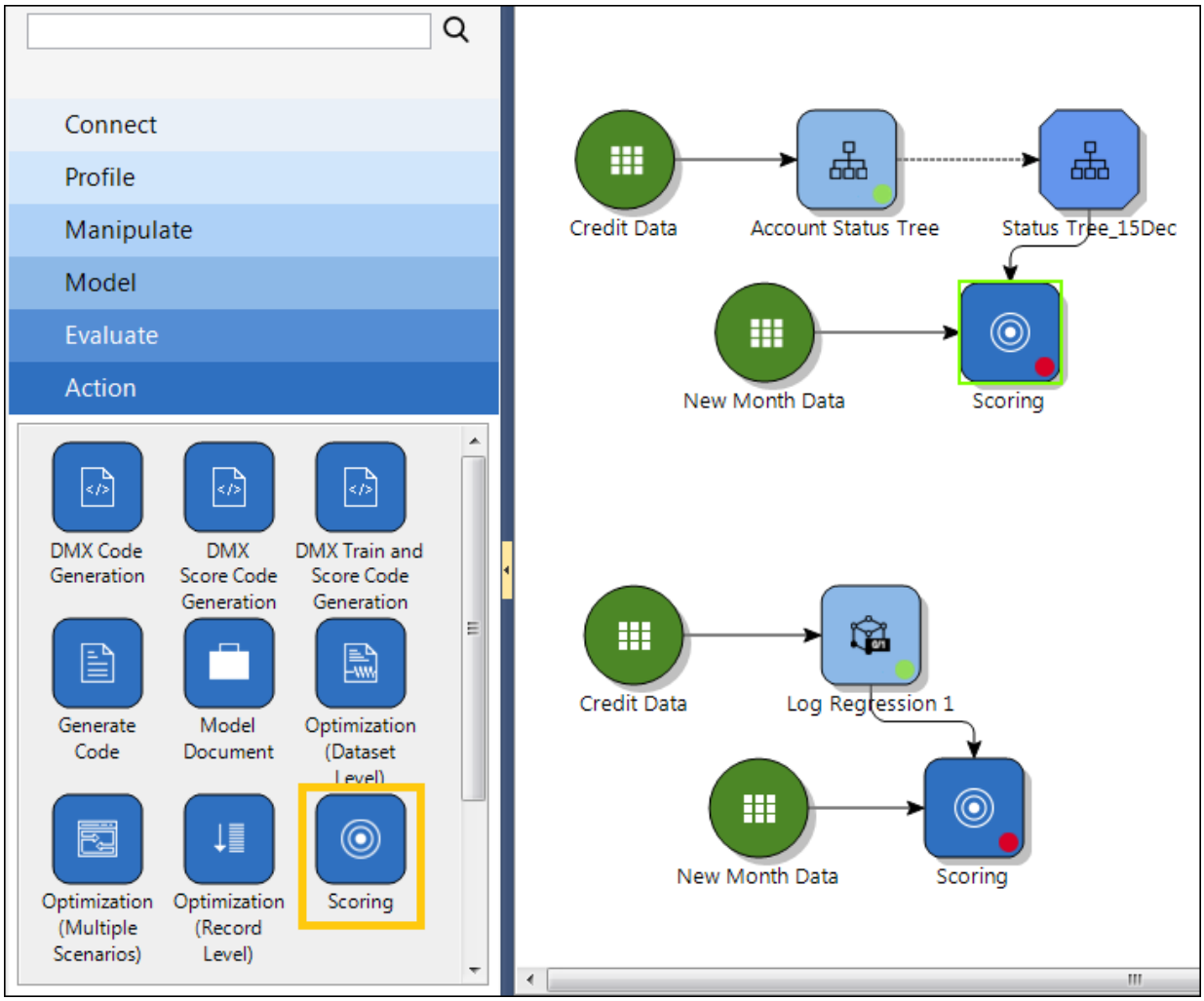
To add a model scoring operation to your workflow:

- Drag the **Scoring** node from the **Action** palette onto the Workflow canvas and connect the input model and dataset nodes to this node. If your model is a Decision Tree, a Strategy Tree, or a Market Basket Analysis process node (rather than a model instance node), you will be prompted to create a model instance first.

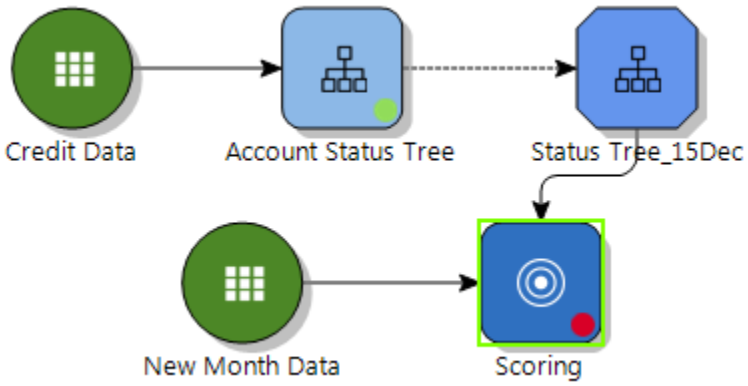
OR

- Right-click anywhere on the workflow canvas and select the context menu command **Action --> Scoring**. This will create a new scoring node. Connect the input dataset and model nodes to it.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



To define the scoring parameters, double-click the **Scoring** node or right-click on it and select **Modify**.



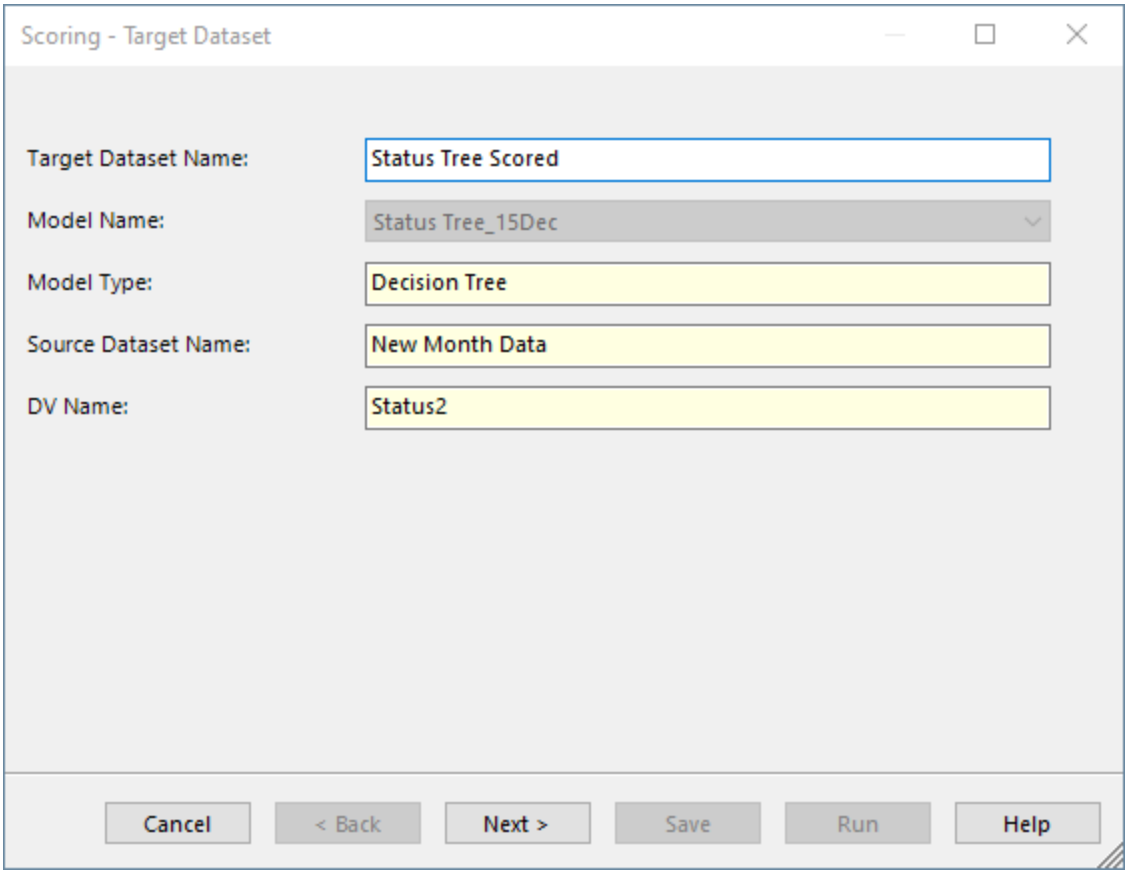
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Step 1

Specify the name for the output dataset in the Target Dataset Name field.

A) Native Altair Models

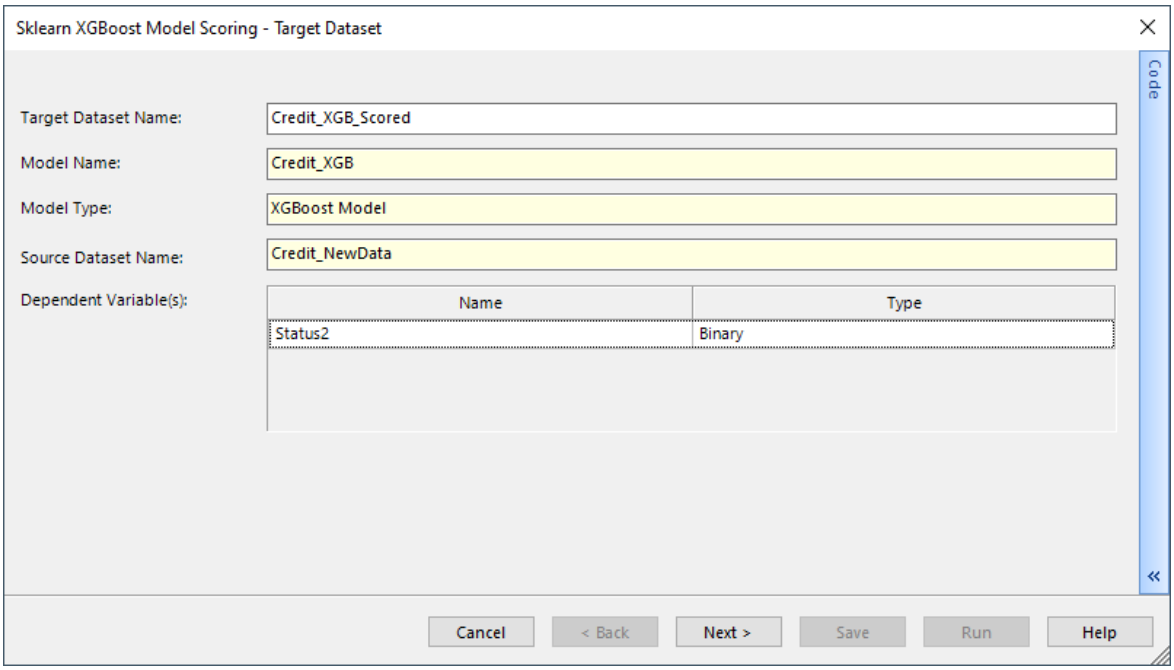
The first page of the Scoring wizard for native models (as opposed to those implemented in Python) looks as follows:



B) Models Implemented in Python

For models Implemented in Python, the first page depends on the model type. For example, for an XGBoost model:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

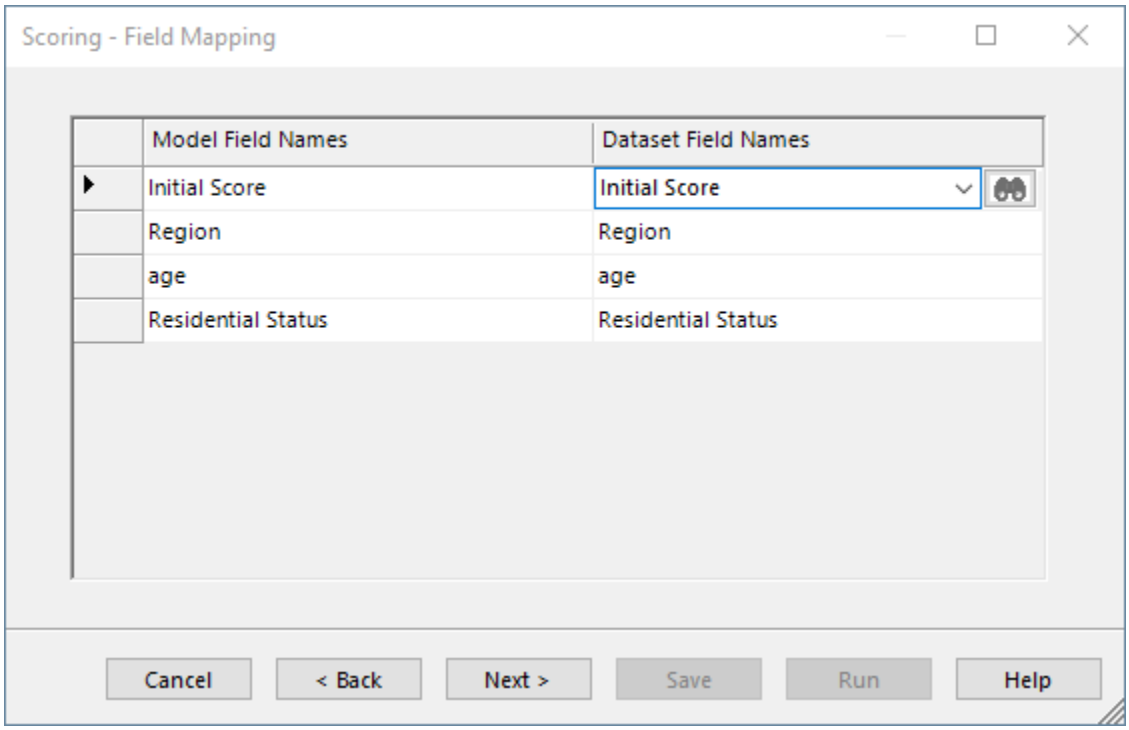


Scoring wizard interface for models that use Python libraries is described in the topics Sklearn GLM Scoring, Sklearn XGB Scoring, Keras Deep Learning Scoring, Keras Forecasting Scoring, Novelty & Outlier Scoring, ARIMA Scoring.

Step 2

The next step provides the option of mapping the field names in the model to the ones in the dataset to be scored. If the names of model variables do not match the names of the corresponding fields in the input dataset, map the fields manually by clicking in the right column (Dataset Field Names) and selecting the right fields to match the model variables in the left column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

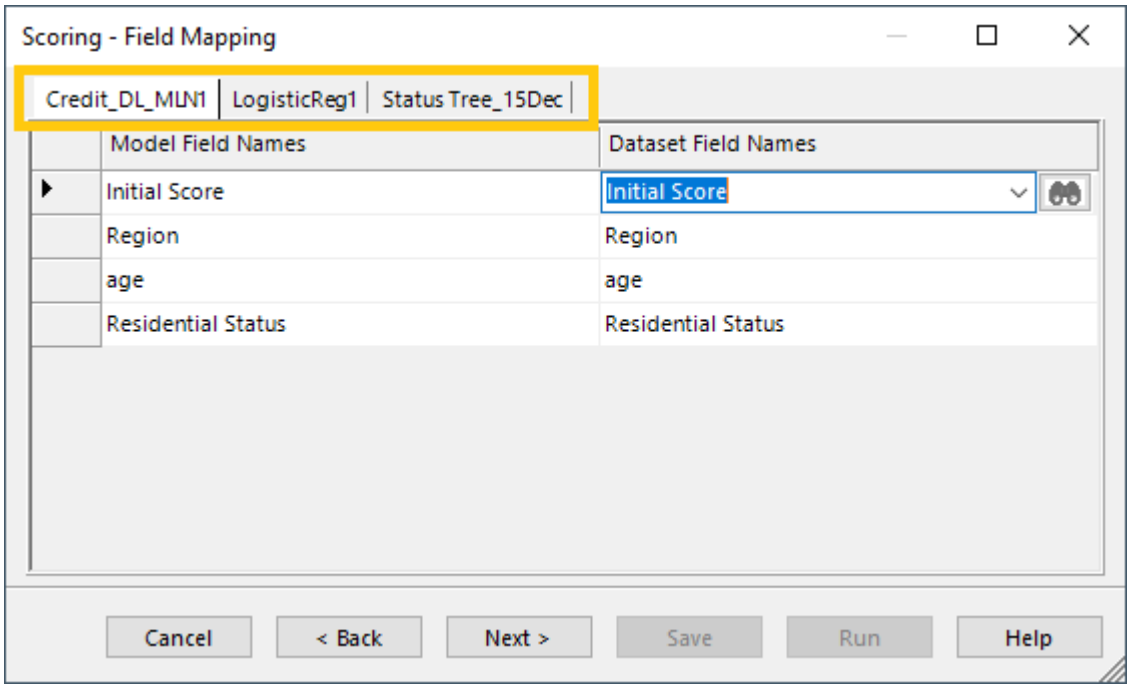


Note: When scoring or validating native Altair models that use derived variables (also called calculated columns), the input dataset must have the original raw variables, while the derived ones are not required. The reason is that the expressions for the derived variables, which are stored in the model, are automatically applied to the input variables during scoring. Therefore, when scoring or validating a native Altair model, not the derived variables have to be mapped, but the original variables from which they were derived, thus allowing to use a dataset with only original independent variables as a scoring input.

In the case of models Implemented in Python, however, the independent variables are used as is, without resolving expressions for any derived variables. Therefore, scoring and validation for such models is possible only against datasets with the derived fields (since they have to be mapped).

In the special case where the input model is a **Model Stacking** node, the field mapping needs to be defined for all models, so the Field Mapping page will have multiple tabs - one tab per each model within the Model Stacking node. In the following example, there are 3 models in the input Model Stacking node. Click on each tab to verify or change the mapping for each model.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Step 3

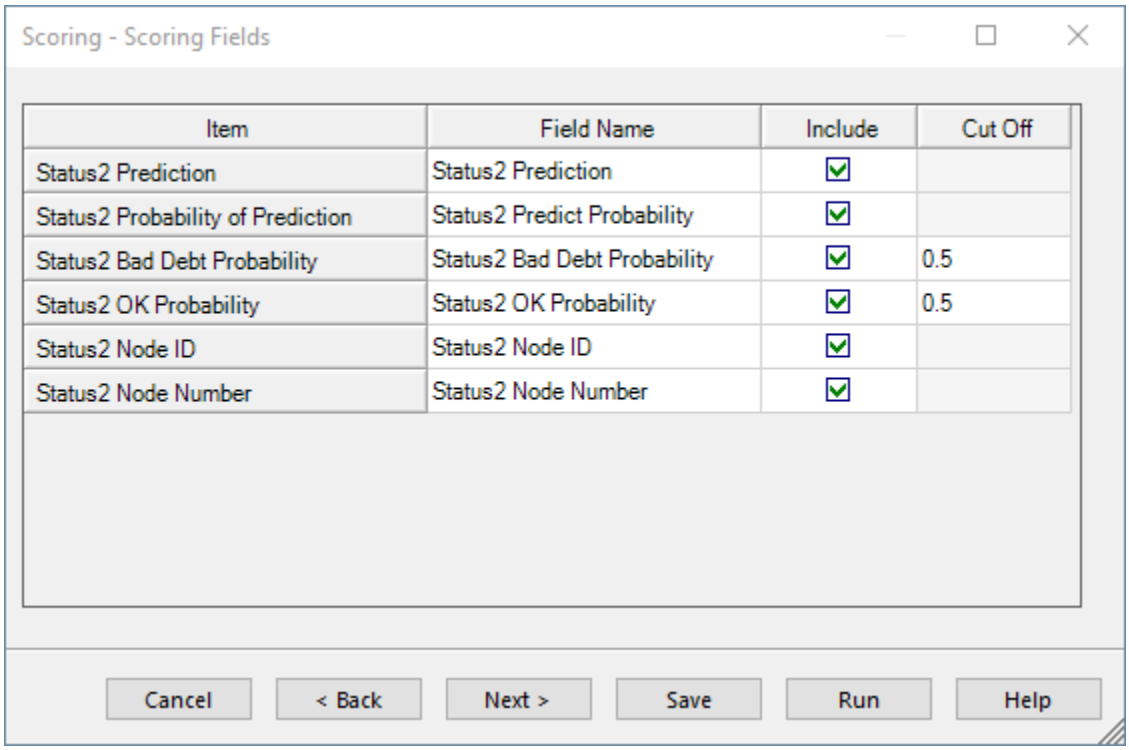
At the next step, choose the new Scoring Fields to be generated for the output dataset. For trees and other predictive models, generated field names are prefixed with the name of the dependent variable.

For Cluster Analysis models, the scoring fields to be created are Cluster Index, Cluster Probability, Distance to Center, 2nd Nearest Cluster Index, 2nd Nearest Cluster Probability, and Distance to the 2nd Nearest Cluster Center.

A) Decision Trees and Native Predictive Models

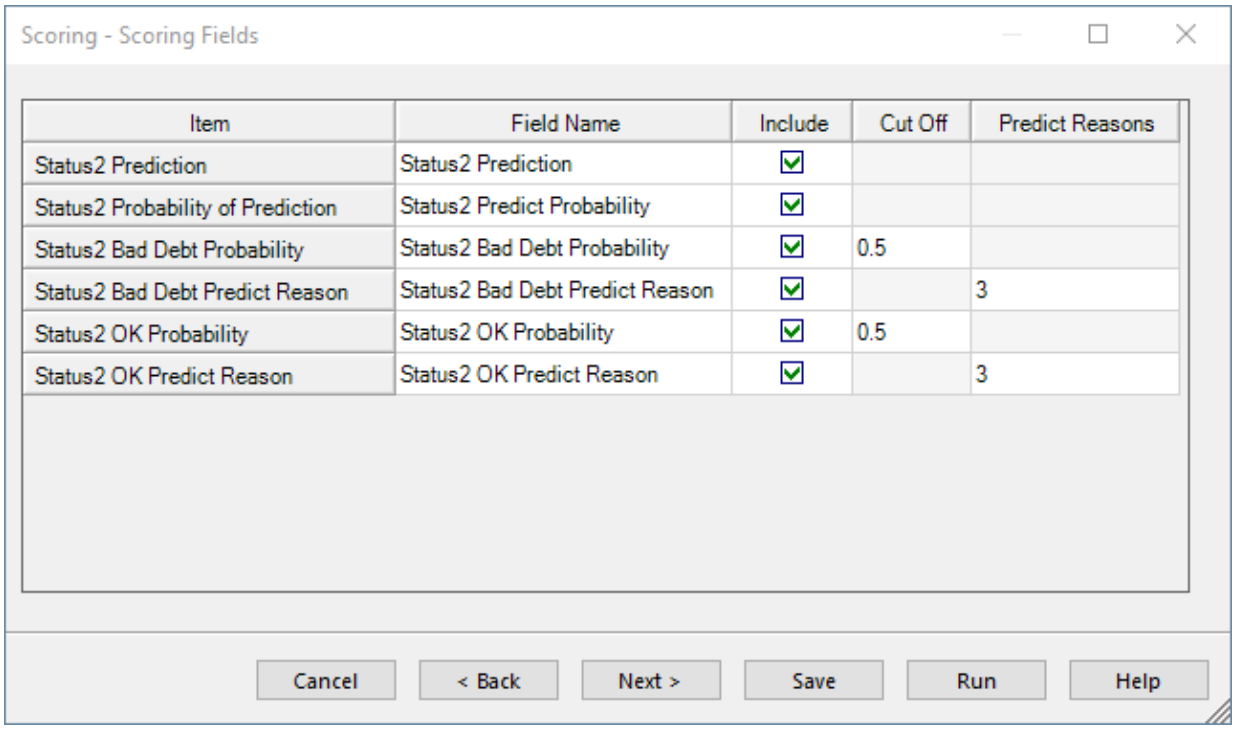
The scoring fields produced by scoring models with discrete dependent variable (DV) are different from those produced by models with continuous DV. For example, for a Decision Tree model with discrete DV, the scoring fields are: DV Prediction, DV Probability of Prediction, DV Value Probability (for each DV value), Node ID, and Node Number.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



For a Logistic Regression model with discrete DV, the scoring fields are: DV Prediction, DV Probability of Prediction, and DV Value Probability for each DV value. For models with binary DV, there are also Predict Reasons that can be included for each DV value. Predict Reasons explain why the model is making a particular prediction. This a comma-separated list of variable names that contributed the most to the prediction. In the Predict Reasons field you can specify the maximum number of reasons (contributing variables) to include. For example, in the example screenshot below, the field Status2 OK Predict Reason will include no more than 3 reasons for predicting the value "OK" for the dependent variable Status2. Predict Reasons are also available for Linear Regression. For details on the calculations behind Predict Reasons, see the section Predictive Models: Regression in Mathematical Formulations.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Prediction The predicted value of the dependent variable for a given record.

DV Value Probability The probability of the specified DV value. For example, the field *Account Status Bad Debt Prob* contains the probability with which the model predicts the value '*Bad Debt*'. Usually, the probability of the value of interest is the only field necessary for the analysis. The probabilities of other DV values are optional.

In the case of a binary DV (a DV with two values), you can also specify the probability **cut off** point at which the prediction will be set in the *DV Prediction* field. For example, if your dependent variable is *Account Status*, with two values, '*Bad Debt*' and '*Good*', and you set the cut-off for *Bad Debt Probability* to 0.7, the *Account Status Prediction* field will be set to '*Bad Debt*' when the *Account Status Bad Debt Probability* is 0.7 or greater. Note that the other dependent variable value probability cut-offs will automatically adjust to the total of 1 when you change the probability cut-off for one value.

Probability of Prediction The probability that the predicted outcome will occur. This field does not distinguish between the categories of the DV and will have a probability regardless of the category.

Node ID Applicable to Decision Tree models only. This is the node ID of the tree node that the record in the input dataset is assigned to during scoring.

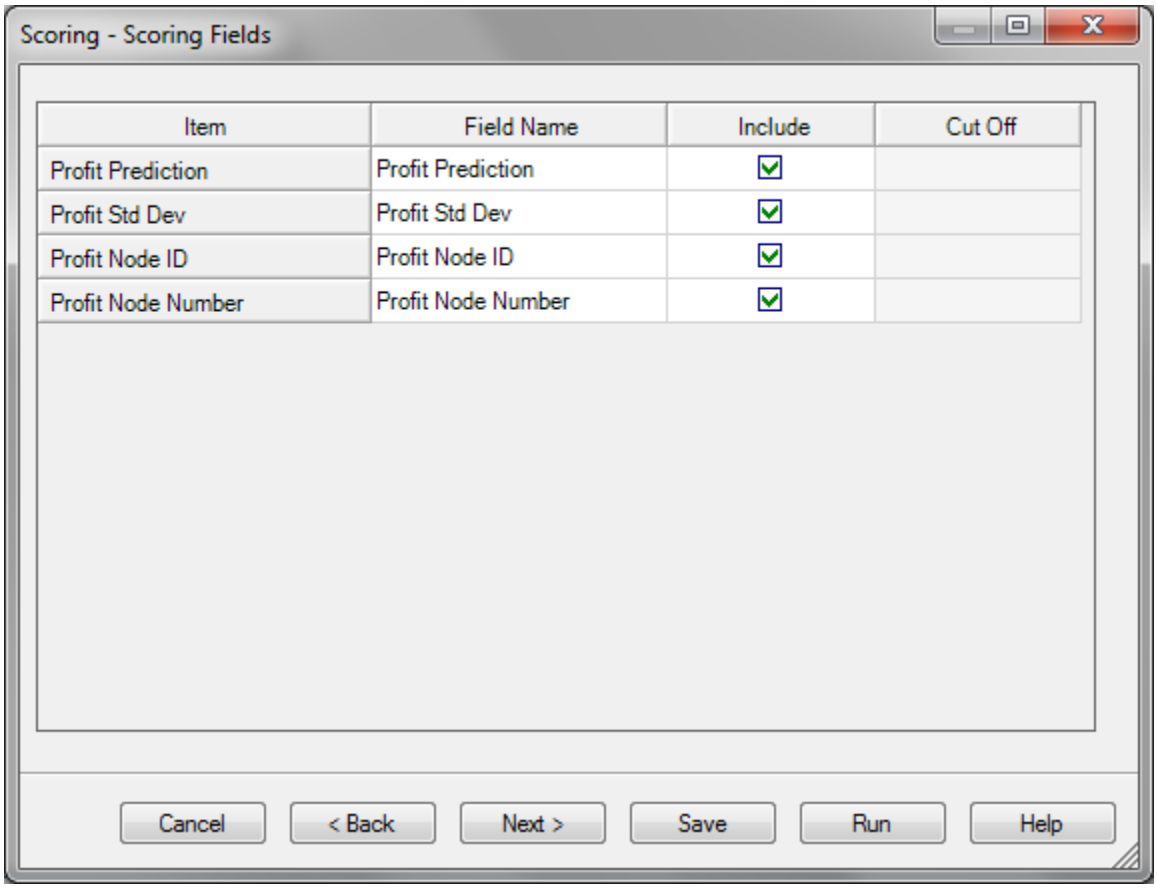
Node Number Applicable to Decision Tree models only. This is the node number of the tree node that the record in the input dataset is assigned to during scoring.

Predict Reason Applicable only to Logistic Regression with binary dependent variable and to Linear Regression. Predict Reasons explain why the model is making a particular prediction. This a comma-separated list of variable

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

names that contributed the most to the prediction. Specify the maximum number of reasons to include.

The scoring fields generated when scoring a model with continuous DV are as follows (the Node ID and Node Number fields are generated for trees only):



Prediction	The predicted value for a given record.
Std Dev	The standard deviation of the DV can be used as the estimated precision of the prediction as estimated by the model.
Node ID	Applicable to Decision Tree models only. T his is the node ID of the tree node that the record is assigned to.
Node Number	For Decision Tree models only, this is the node number of the tree node that the record is assigned to.
Predict Reason	Applicable only to Linear Regression and Logistic Regression with binary DV. Predict Reasons explain why the model is making a particular prediction. This a comma-separated list of variable names that contributed the most to the prediction. Specify the maximum number of reasons to include.

B) Cluster Analysis Models

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The scoring fields produced by scoring Cluster Models depend on the type of the cluster model.

For K-Means Clustering, the scoring fields are:

- Cluster Index (the index of the cluster that the record is assigned to).
- Distance to Center (the distance to the center of the cluster that the record is assigned to)
- 2nd Nearest Cluster Index (called *2nd Cluster Index* for short)
- Distance to 2nd Nearest Center (the distance to the center of the 2nd nearest cluster)

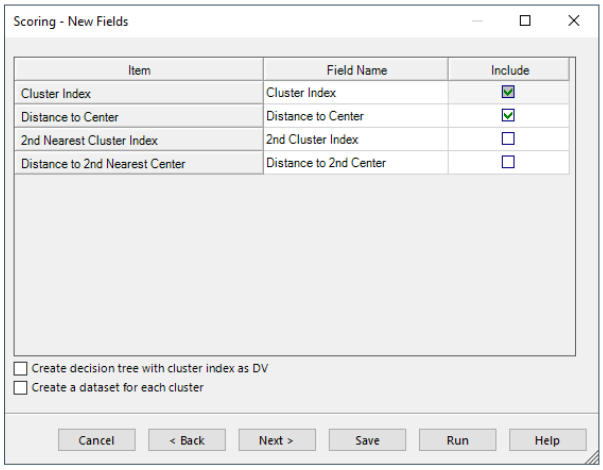
For Expectation Maximization Clustering, the scoring fields are:

- Cluster Index
- Cluster Probability
- 2nd Nearest Cluster Index (called *2nd Cluster Index* for short)
- 2nd Nearest Cluster Probability (called *2nd Cluster Probability* for short)

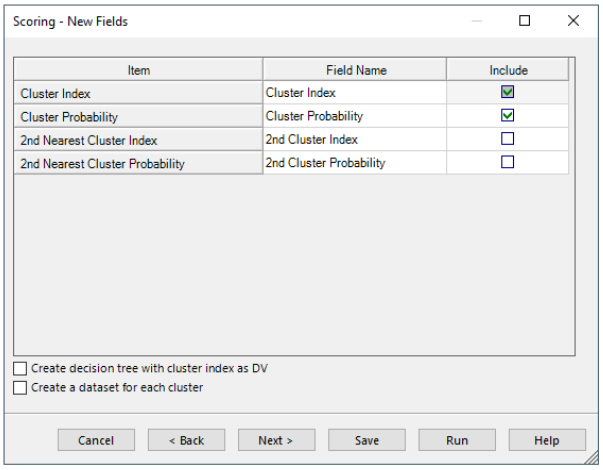
Cluster Probability indicates the probability for the record to be assigned to the corresponding cluster. For details on the Cluster Analysis algorithm and the calculation of Cluster Probability, see the Cluster Analysis Models section of Mathematical Formulations in this Help or the User Guide.

The Scoring Fields page of the Scoring Wizard for Cluster Analysis models looks as follows:

K-Means Clustering



Expectation Maximization Clustering



By selecting any of the options *Create a decision tree with cluster index as DV* or *Create a dataset for each cluster*, you choose to automatically create the following objects in your project:

- a Decision Tree with *Cluster Index* as the dependent variable. Find Split will be performed automatically in this tree to show the most significant factors in the cluster association.
- a separate dataset for each cluster detected in the input dataset by scoring.

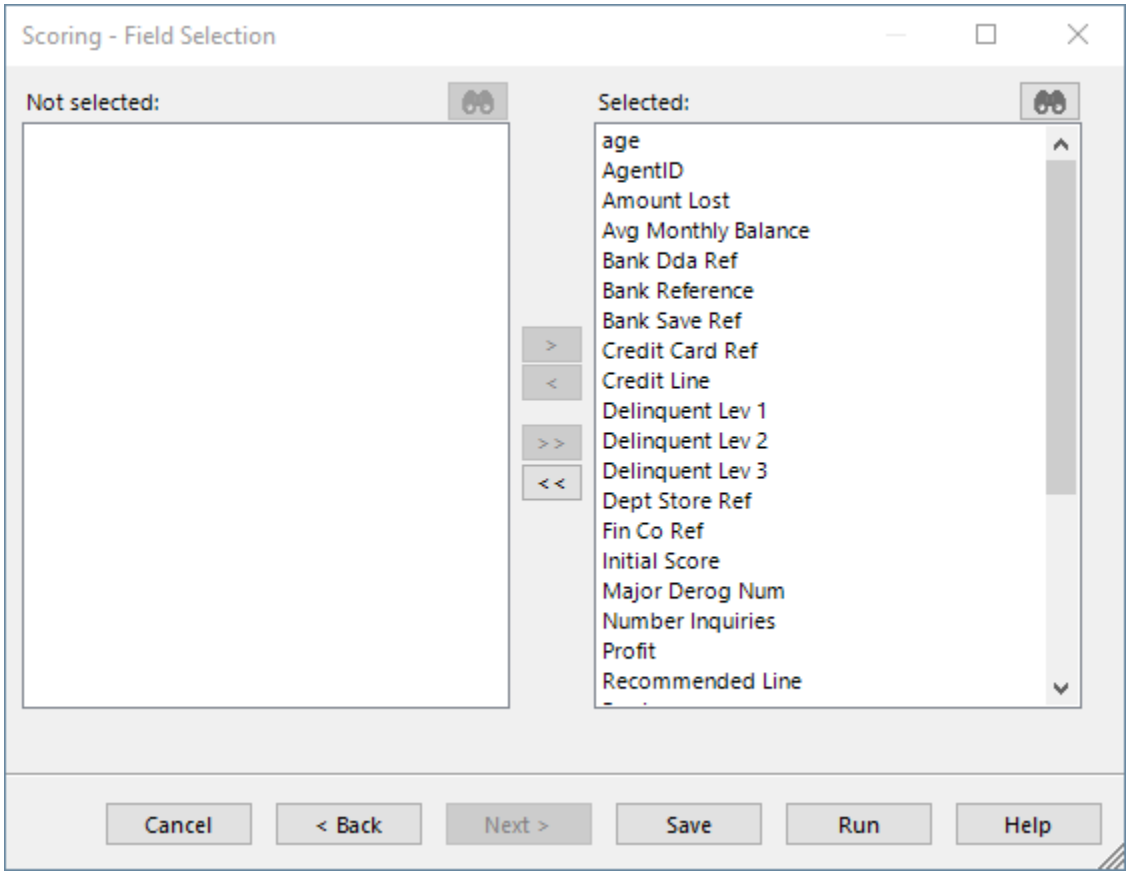
C) Models implemented in Python

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Scoring wizard interface for models that use Python libraries is described in the topics Sklearn GLM Scoring, Sklearn XGB Scoring, Keras Deep Learning Scoring, Keras Forecasting Scoring, Novelty & Outlier Scoring, ARIMA Scoring.

Step 4

At the **Field Selection** step, choose the fields from the input dataset that you would like to include in the scoring output in addition to the new score / probability fields. By default, all fields in the scoring input dataset are selected to be included in the output.

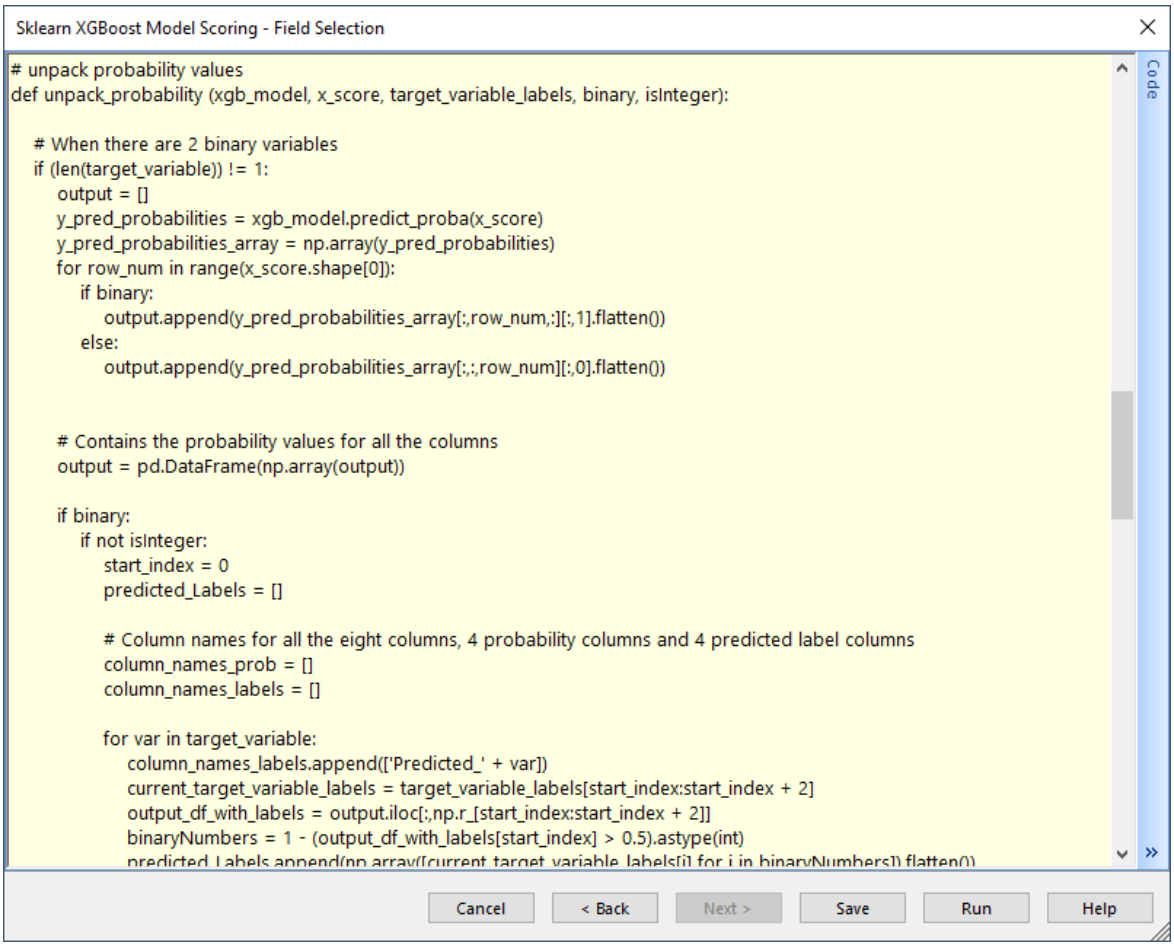


Note: If you do not include the fields required by the model, scoring will still be performed, but the score / probability fields in the output dataset will be instantiated (i.e., values only, without scoring expressions).

For models implemented in Python:

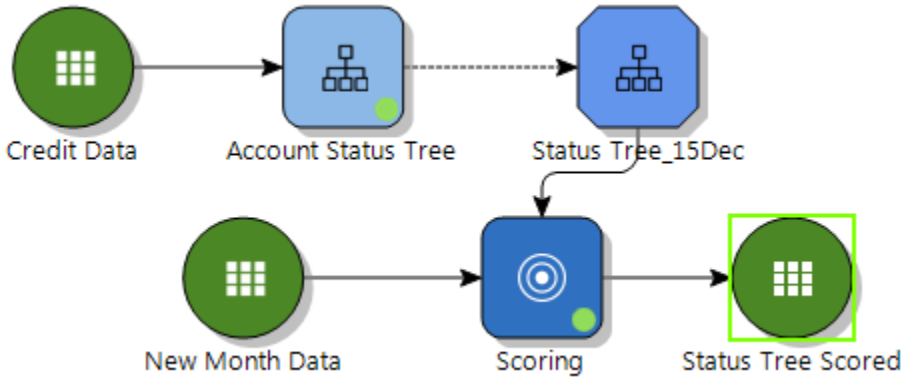
To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the scoring process immediately.

The output dataset node will be created in the workflow after running the scoring node. It can be viewed and manipulated like any other dataset.



Proceed to Score Report.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Scoring Wizard for Python-based Models

Novelty & Outlier Scoring

Scoring a Novelty & Outlier Detector model is similar to scoring other models, but it produces anomaly estimation metrics and outlier indicators instead of probabilities. A Novelty & Outlier Detector model is scored using the **Scoring** node in the Action palette. For general information about scoring, see the topic About Scoring.

Prerequisites for Novelty & Outlier model scoring: a **Novelty & Outlier Detector** node and an input dataset with new data that has no outlier or anomaly information.

> ***Note***: For the Local Outlier Factor method, scoring is not applicable when the *Novelty* option is set to False.

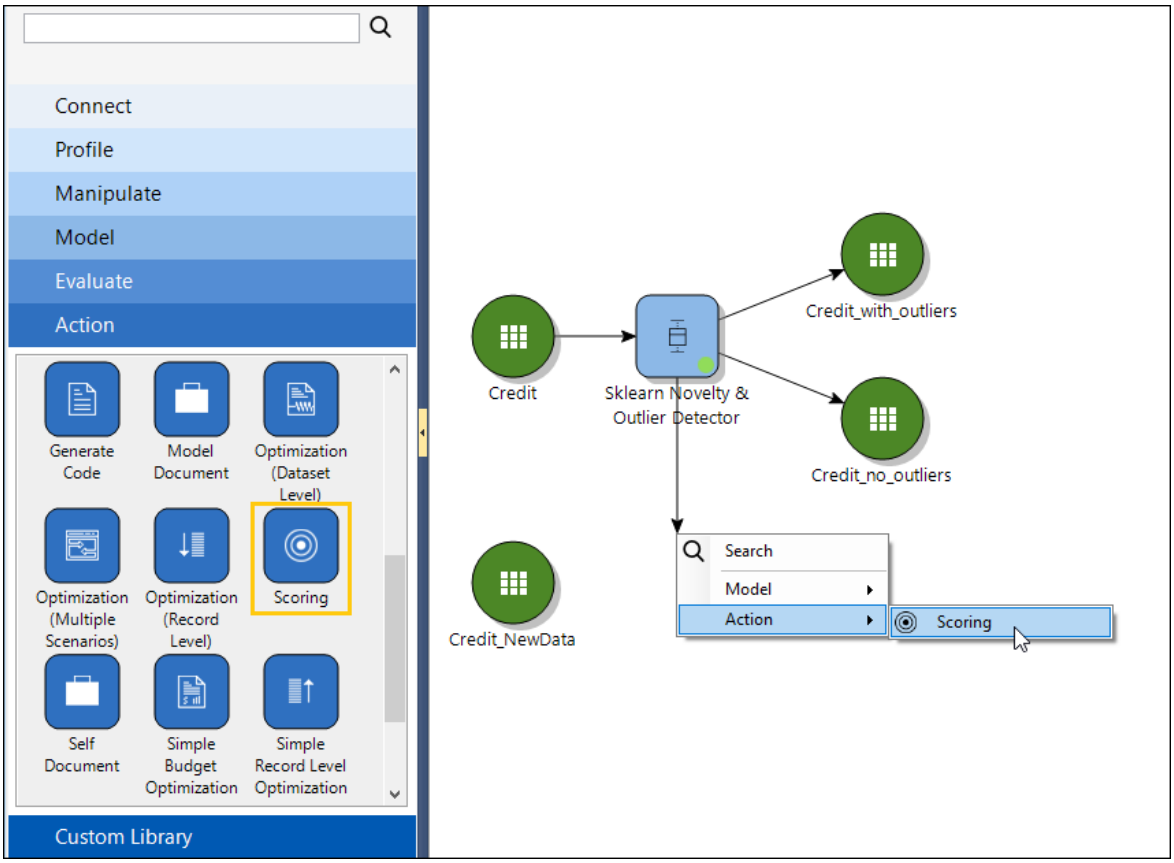
To add a Scoring operation to your workflow:

- Drag the **Scoring** node from the **Action** palette onto the Workflow canvas and connect the inputs: **a Novelty & Outlier Detector** node and a dataset node that contains the new data to score.

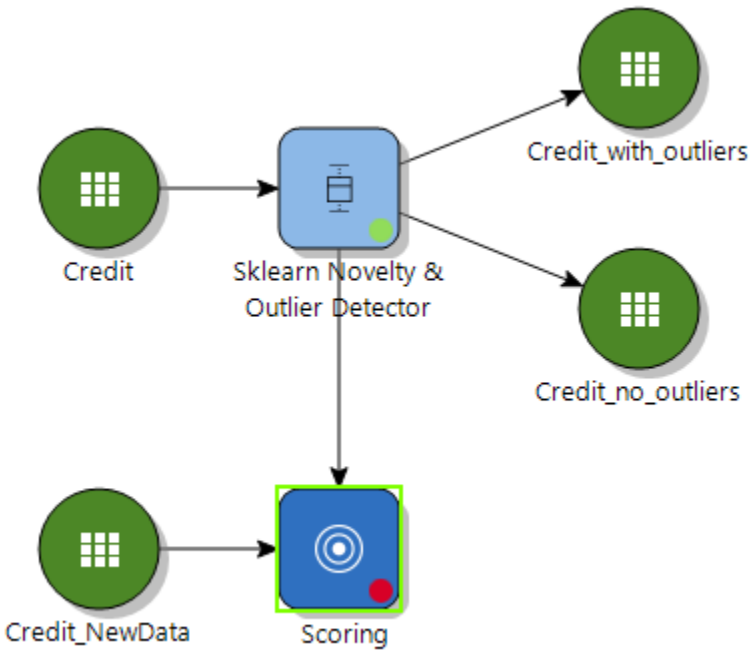
  OR

- Right-click anywhere on the workflow canvas and select the context menu command **Action --> Scoring**. This will create a new Scoring node. Connect the inputs a **Novelty & Outlier Detector** node and a dataset node that contains the data to score.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



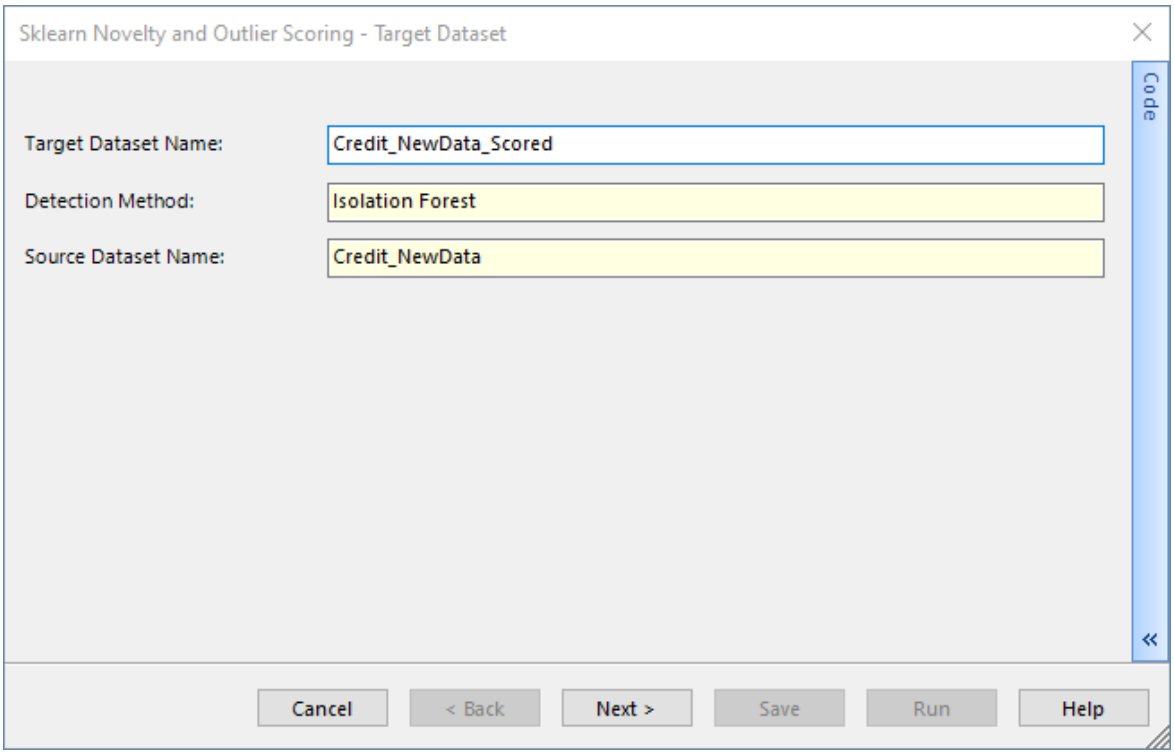
Double-click the **Scoring** node or right-click on it and select **Modify**.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The Scoring wizard opens. It displays the name of the input dataset and the method used in the input Novelty & Outlier Detection model node.

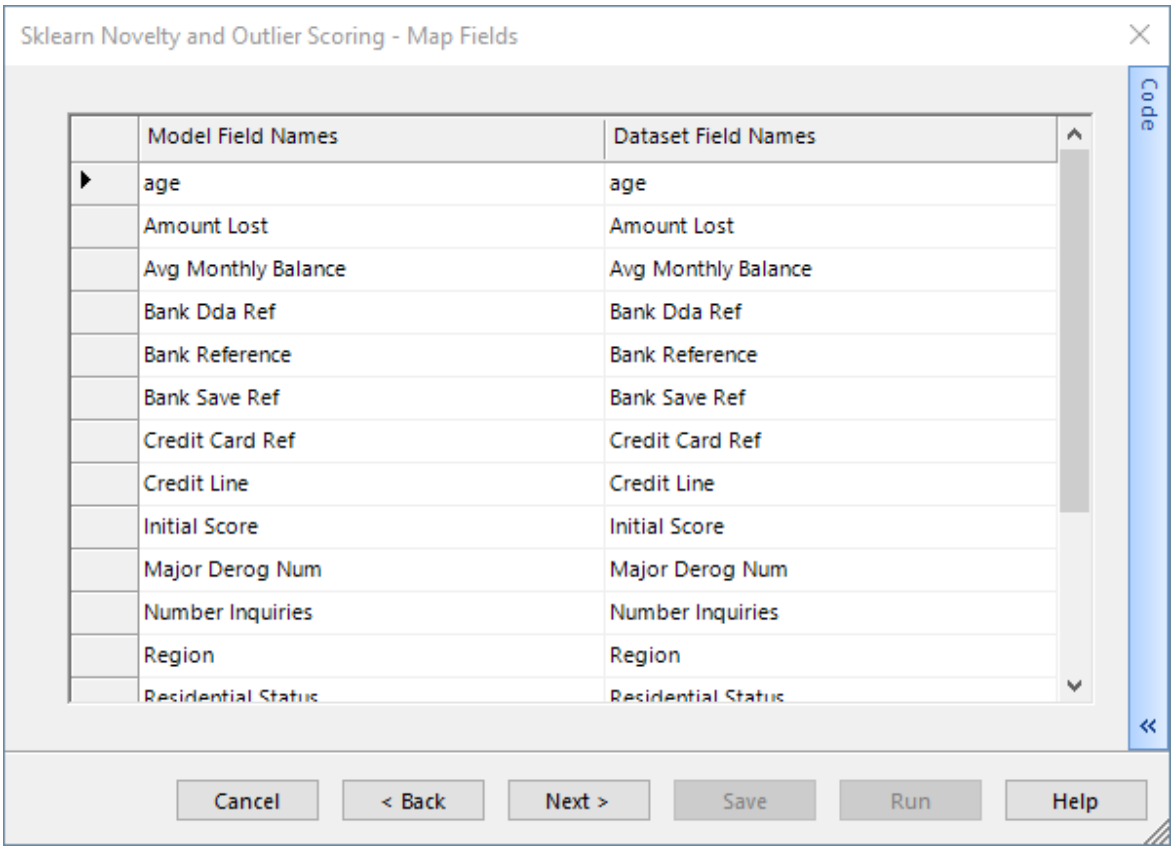
In the *Target Dataset Name* field, specify the name for the output dataset that will contain the scores.



Click **Next**.

The next step provides the option of mapping the field names in the model to the ones in the validation input dataset. In some situations, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map fields by clicking in the right column (Dataset Field Names) and selecting the right fields to match the model variables in the left column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

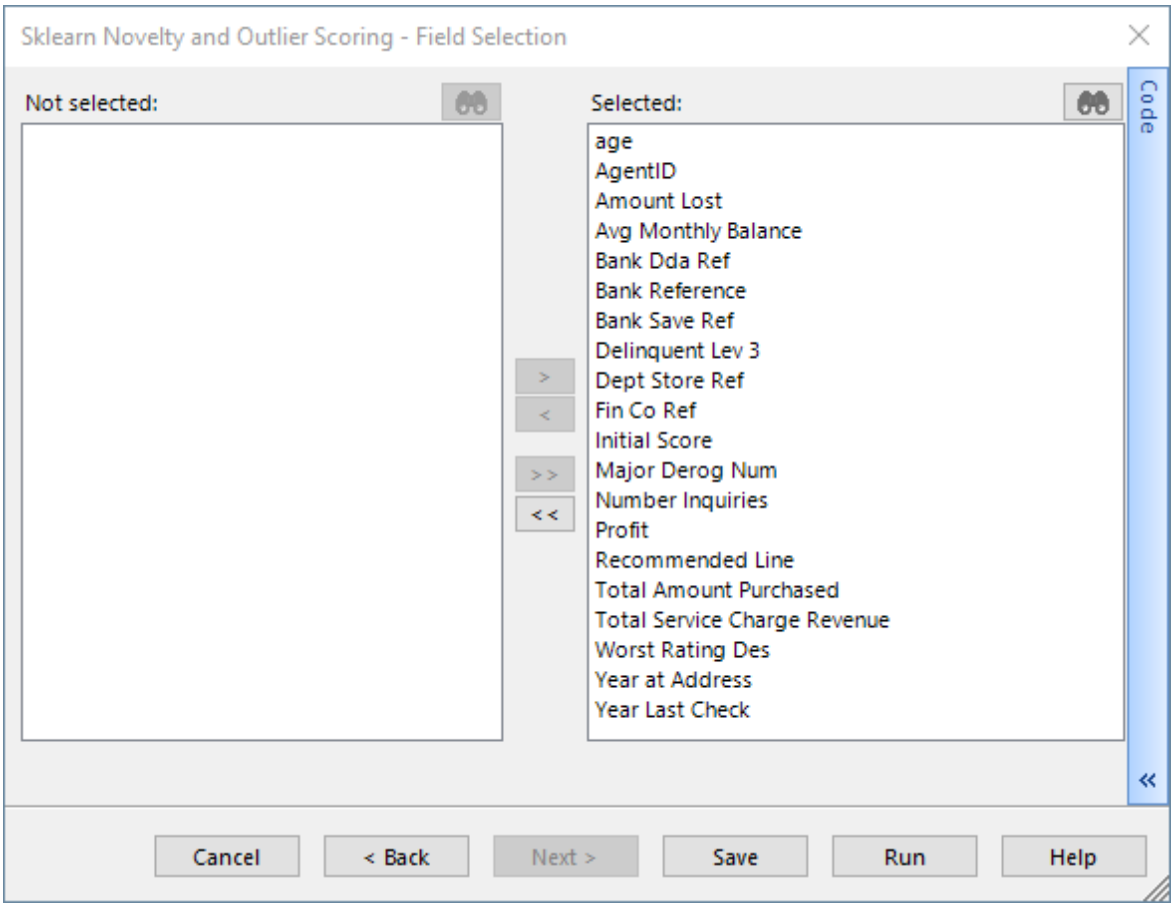


Note: When scoring Novelty & Outlier Detector models that use derived variables (also called calculated columns) with expressions, the input dataset must include these variables. This is different from the case of native Altair models with derived variables, which can be scored on a dataset with only *original* variables that the derived ones depend on, while the derived ones are not required.

Click **Next**.

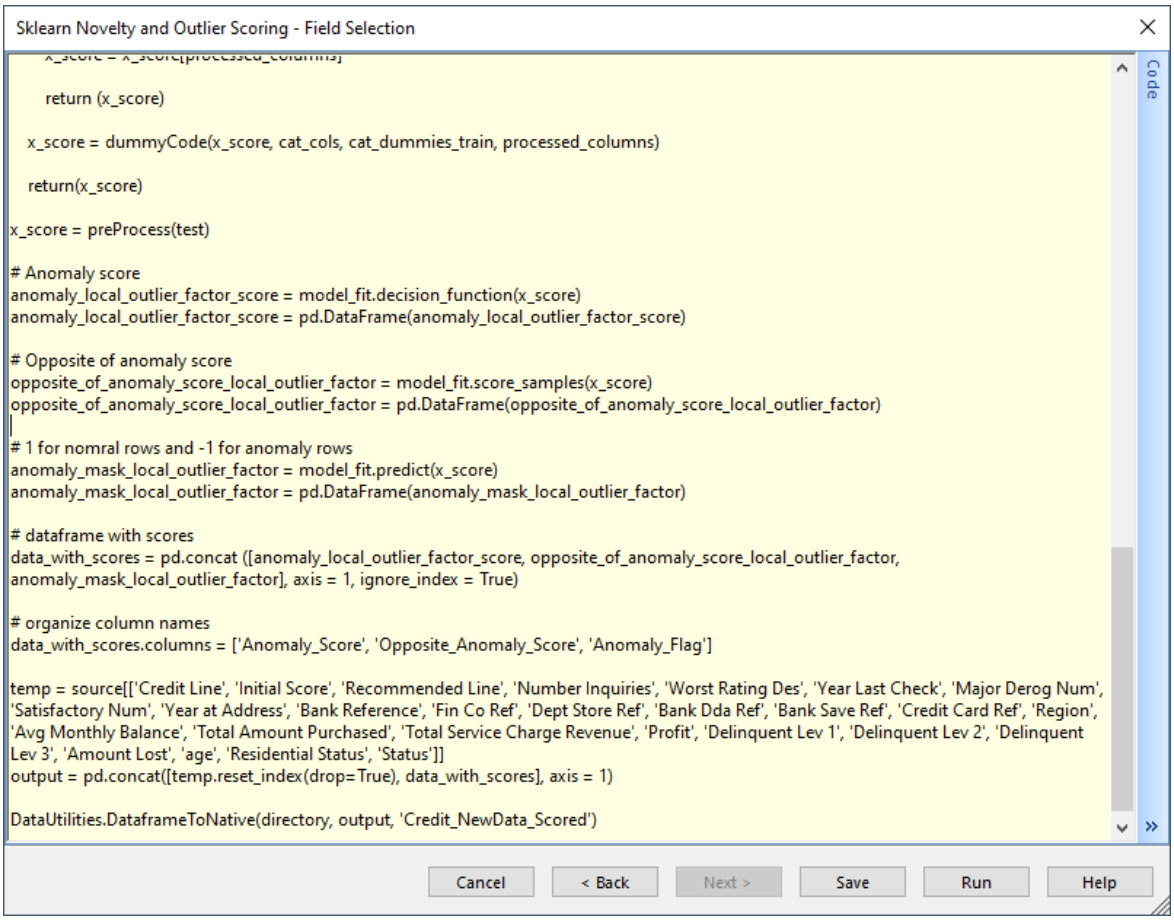
In the **Field Selection** page, choose the fields from the input dataset that you would like to include in the scoring output in addition to the new anomaly score fields. By default, all fields in the scoring input dataset are selected to be included in the output.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



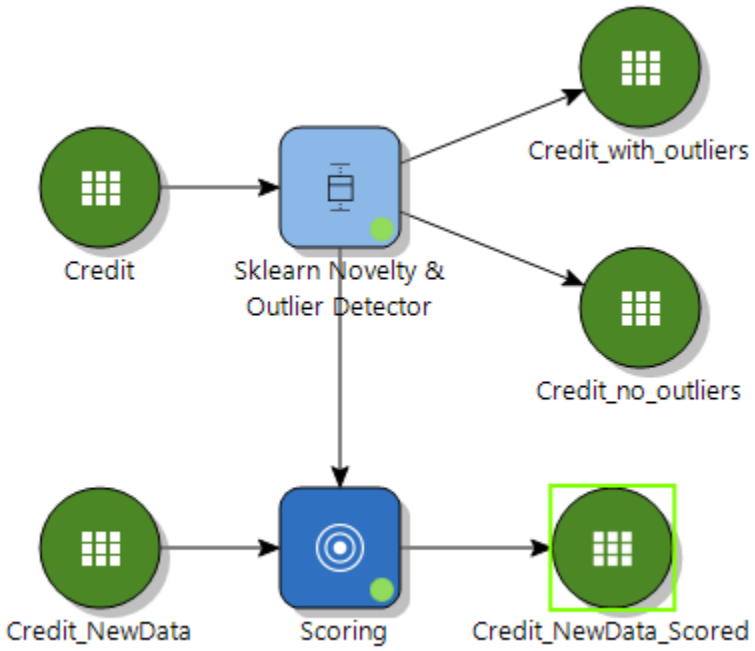
To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the scoring process immediately. The model is scored in a Python container session, and then the dataframe with generated scores is converted back from Pandas to Altair KDD format. The output dataset node is created in the workflow. Double-click the output dataset to view the results.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click the Data tab of the output dataset and scroll to the rightmost columns. In this example from scoring a model that uses the Isolation Forest method, the scored dataset has the *Anomaly_Flag* column showing the outlier indicator based on the values of *Anomaly_Score* and *Opposite_Anomaly_Score*.

	Delinquent	Amount Lost	age	Residential Status	Status	Anomaly_Score	Opposite_Anomaly_Score	Anomaly_Flag
1	0	0	(null)	(null)	OK	0.0306769841368138	-0.469323015863186	1
2	3	557.67	22	Rent	Bad Debt	-0.0157806793598609	-0.515780679359861	-1
3	0	0	37	Own Home	OK	0.00701388289900789	-0.492986117100992	1
4	0	0	25	Other	OK	0.0154828208525022	-0.484517179147498	1
5	0	0	81	(null)	OK	0.0239839169398195	-0.47601608306018	1
6	0	0	(null)	(null)	OK	0.0341440893542848	-0.465855910645715	1
7	0	0	(null)	(null)	OK	0.0299562376556643	-0.470043762344336	1
8	0	0	62	(null)	OK	0.0245103050179628	-0.475489694982037	1
9	1	0	67	(null)	OK	0.0175292749437517	-0.482470725056248	1
10	0	0	30	Other	OK	0.0200072406322655	-0.479992759367735	1
11	0	0	20	Own Home	OK	0.0108821254141328	-0.489117874585867	1
12	0	0	61	Own Home	OK	0.00165246899136623	-0.498347531008634	1
13	0	0	35	Rent	OK	0.0118635398366399	-0.48813646016336	1
14	0	0	21	(null)	OK	0.0256281662644814	-0.474371833735519	1
15	0	0	59	Own Home	OK	0.000304559304685592	-0.499695440695314	1

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

See also: Novelty & Outlier Detector

Keras Deep Learning Scoring

A Keras Deep Learning model is scored using the **Scoring** node in the Action palette. For general information about scoring, see the topic About Scoring.

Prerequisites for Keras model scoring: a **Keras Deep Learning** node and an input dataset to score. The input Keras Deep Learning node stores the Keras Deep Learning in HDF5 format.

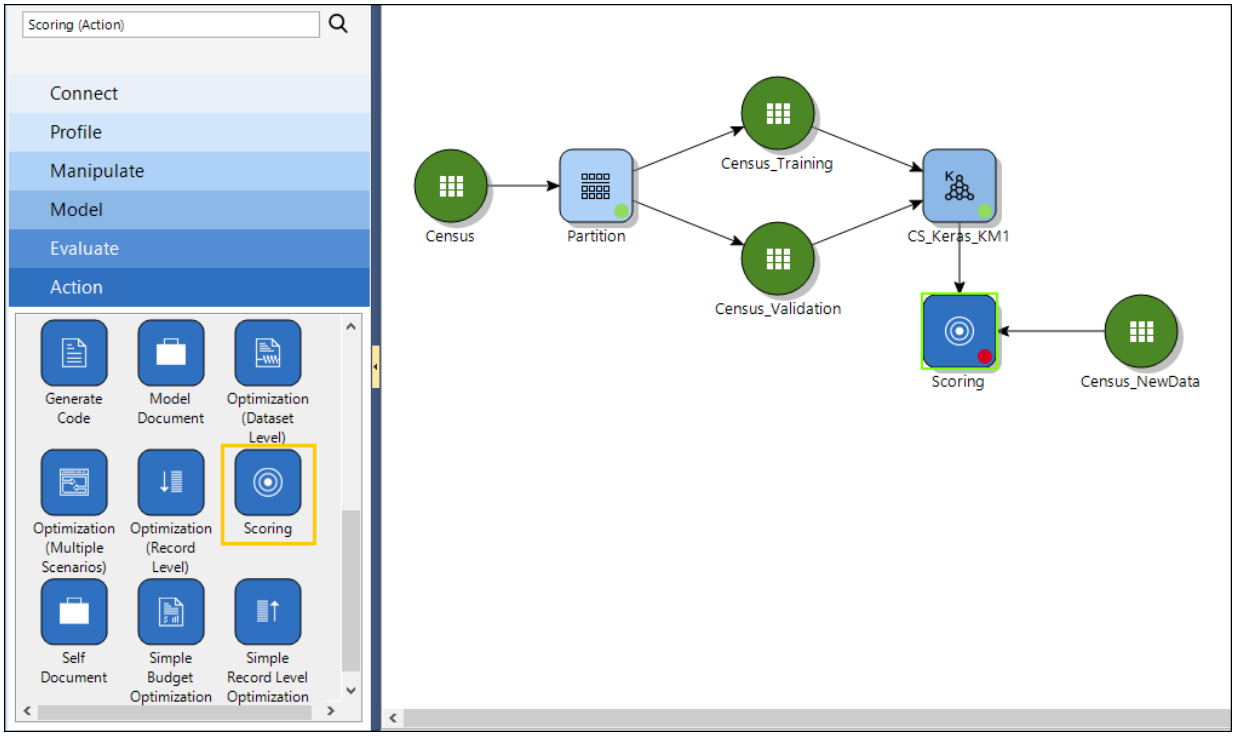
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To add a Scoring operation to your workflow:

- Drag the **Scoring** node from the **Action** palette onto the Workflow canvas and connect the inputs: a **Keras Deep Learning** node and a dataset node that contains the data to score. See the example below.

OR

- Right-click anywhere on the workflow canvas and select the context menu command **Action --> Scoring**. This will create a new Scoring node. Connect the inputs: a Keras Deep Learning model node and a dataset node that contains the data to score.

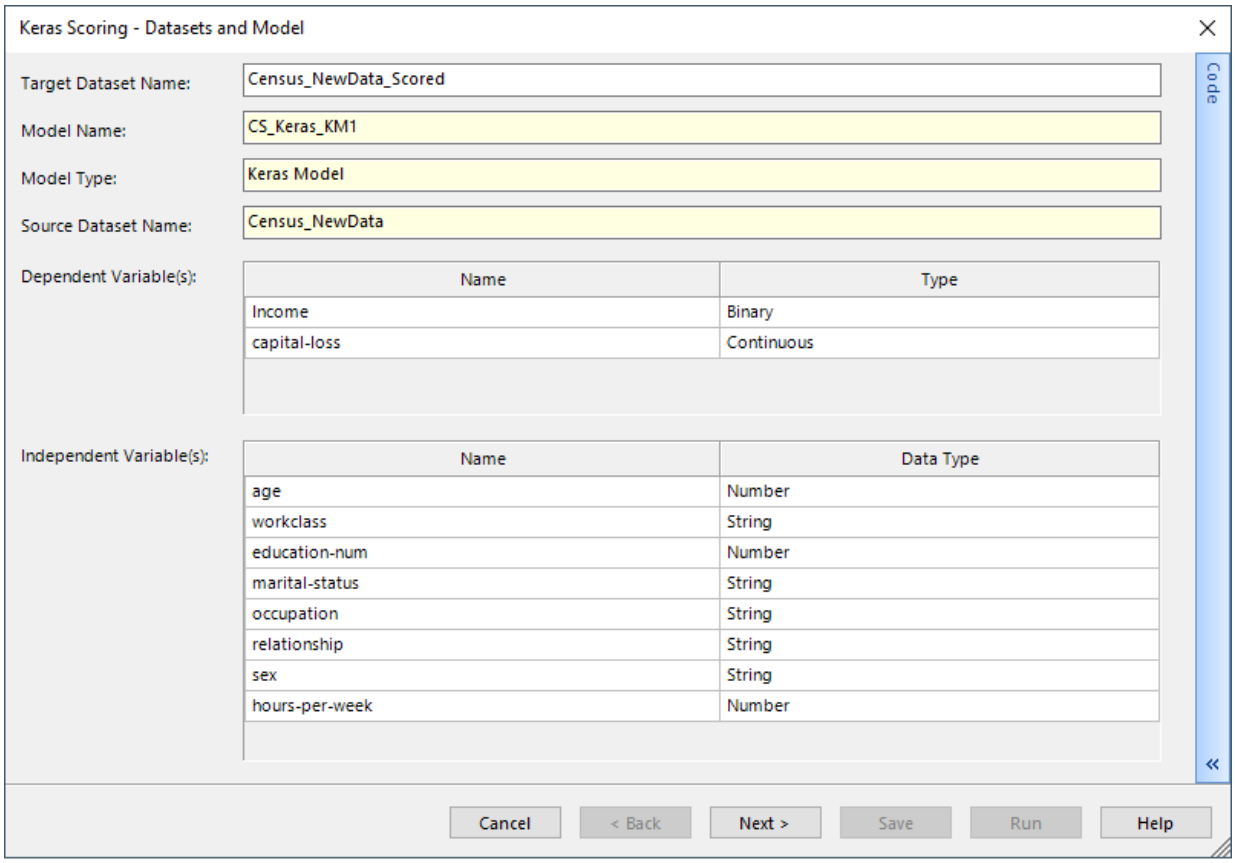


Double-click the **Scoring** node or right-click on it and select **Modify**.

The Scoring wizard opens. It displays the names of the input model and dataset, the list of dependent and independent variables and their data types.

In the *Target Dataset Name* field, specify the name for the output dataset that will contain the scores.

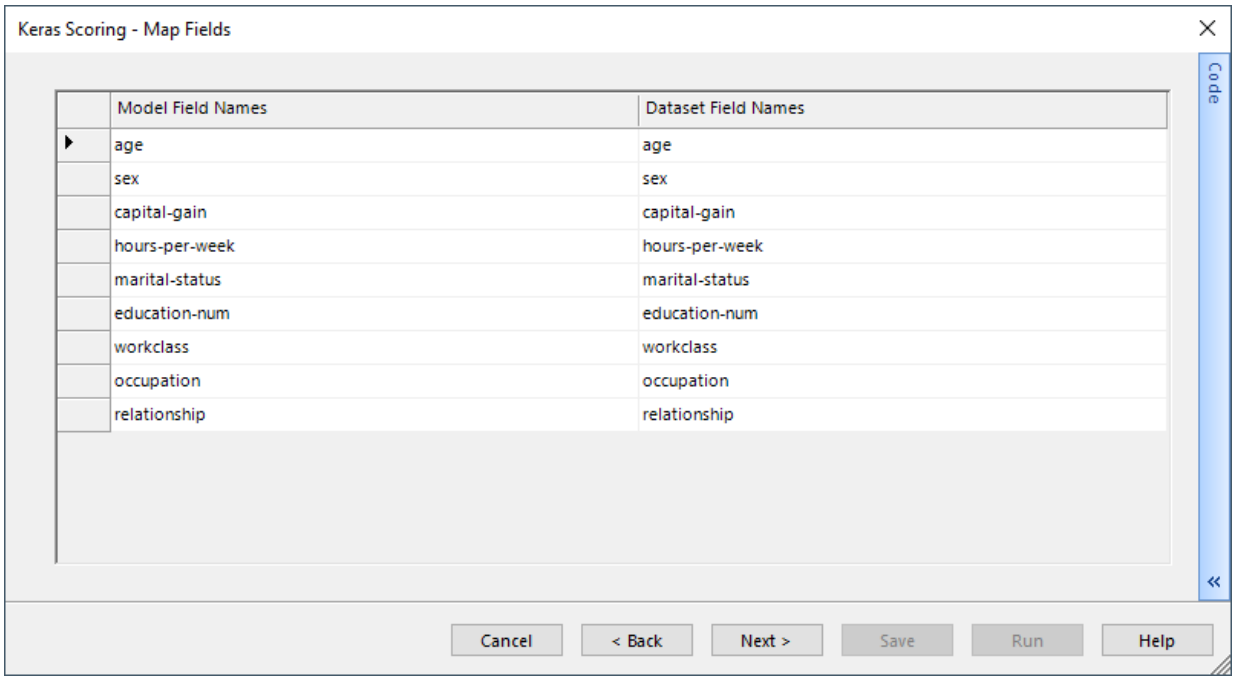
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

The next step provides the option of mapping the field names in the model to the ones in the scoring input dataset. In some situations, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map fields by clicking in the right column (Dataset Field Names) and selecting the right fields to match the model variables in the left column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

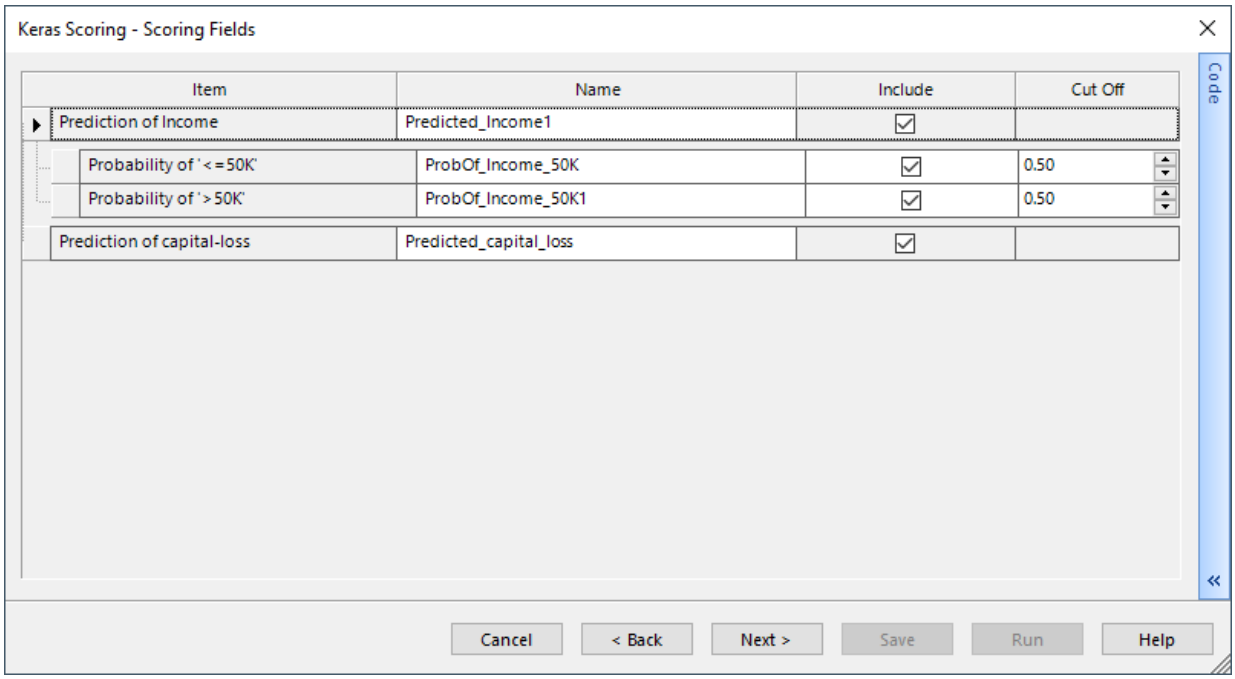


Note: When scoring Keras Deep Learning models that use derived variables (also called calculated columns) with expressions, the input dataset must include these variables. This is different from the case of native Altair models with derived variables, which can be scored on a dataset with only *original* variables that the derived ones depend on, while the derived ones are not required.

Click **Next**.

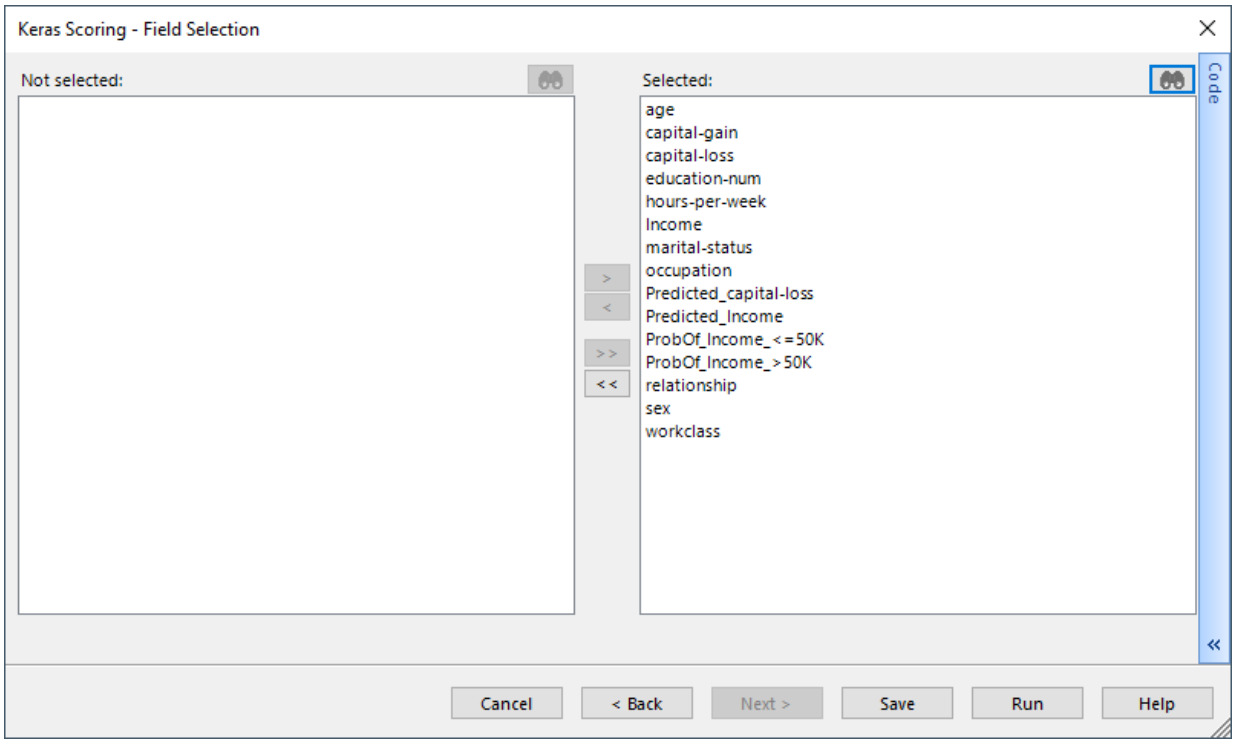
In the **Scoring Fields** page, select the scoring fields to be included in the scoring output dataset and modify their names if necessary. For discrete dependent variables, specify the probability cut-off values. See the general Scoring Wizard topic for more information on cut-off values.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



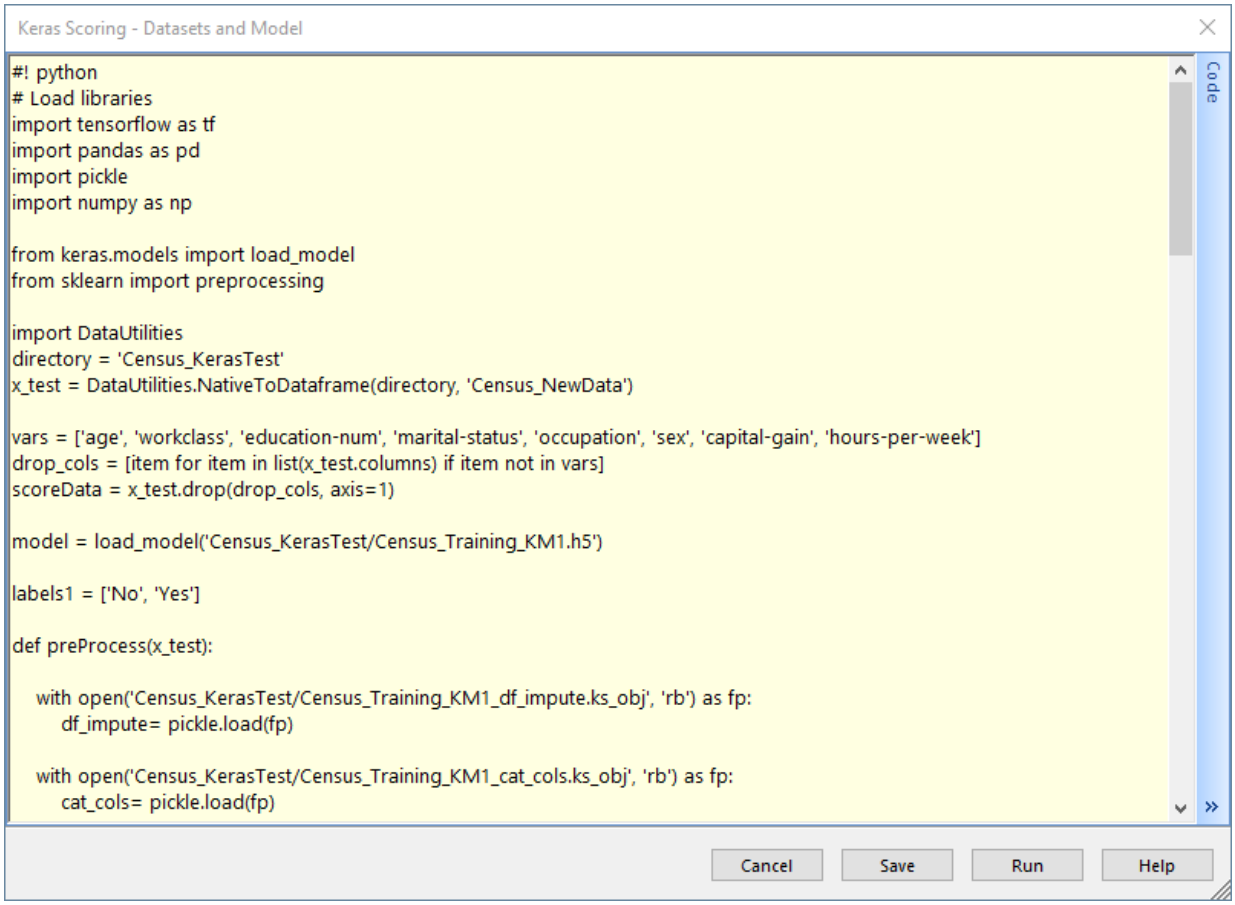
Click **Next**.

In the **Field Selection** page, choose the fields from the input dataset that you would like to include in the scoring output in addition to the new score / probability fields. By default, all fields in the scoring input dataset are selected to be included in the output.



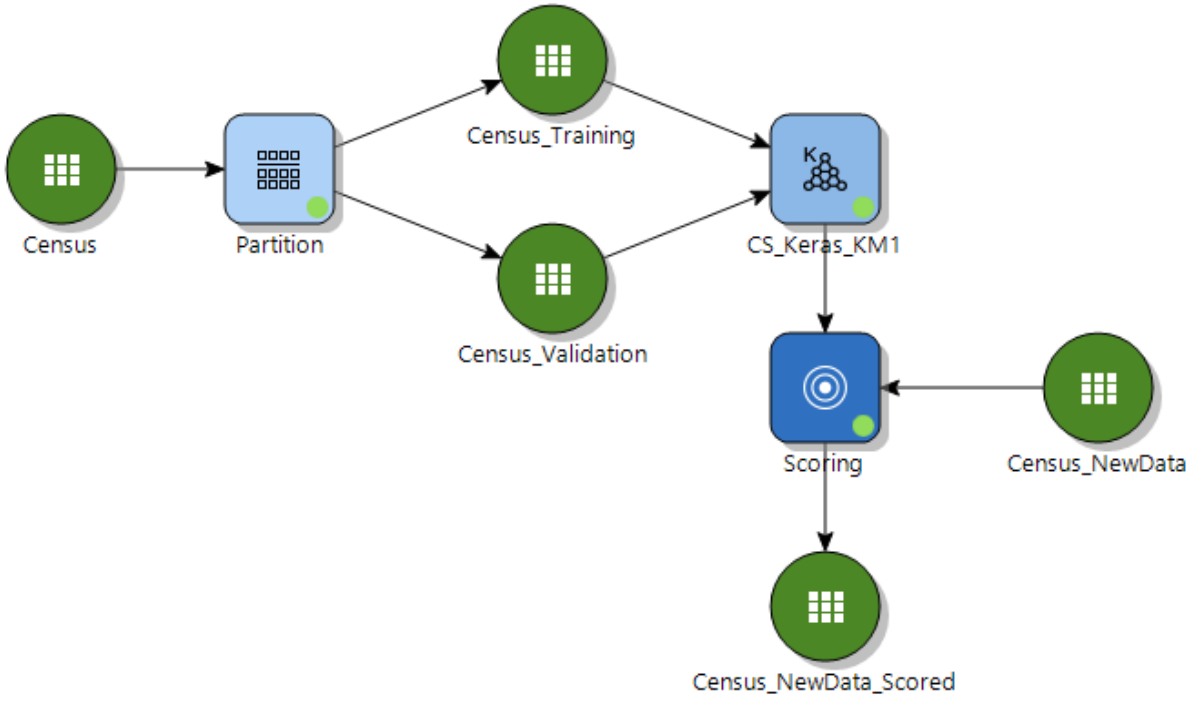
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.



Click **Save** to save the settings, or click **Run** to save the settings and run the scoring process immediately. The model is scored in a Python session with TensorFlow backend against the input data and the generated probabilities / scores are converted back from Pandas to Altair KDD format.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Double-click on the output dataset to view the scoring results.

- For discrete dependent variables, the output dataset contains the probabilities of the DV values and the prediction (the value that has the highest probability) for each record.
- For continuous dependent variables, the output dataset contains the predicted value for each record.

	Predicted_capital-loss	Predicted_Response	ProbOf_Response_No	ProbOf_Response_Yes
25	20.051362991333	No	0.818814516067505	0.181185454130173
26	113.580047607422	No	0.529841303825378	0.470158696174622
27	104.981788635254	No	0.542384803295135	0.457615196704865
28	223.476135253906	Yes	0.353432238101959	0.646567761898041
29	89.8815460205078	No	0.66495019197464	0.33504980802536
30	134.846542358398	Yes	0.478796482086182	0.521203517913818
31	165.005187988281	Yes	0.457728087902069	0.542271912097931
32	63.8199729919434	No	0.66535758972168	0.334642440080643
33	41.4115982055664	No	0.784708797931671	0.215291202068329
34	61.854076385498	No	0.626626670360565	0.373373329639435
35	121.861389160156	No	0.664023637771606	0.335976392030716

See also: Keras Deep Learning

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Keras Forecasting Scoring

A Keras Forecasting model is scored using the **Scoring** node in the Action palette. For general information about scoring, see the topic About Scoring.

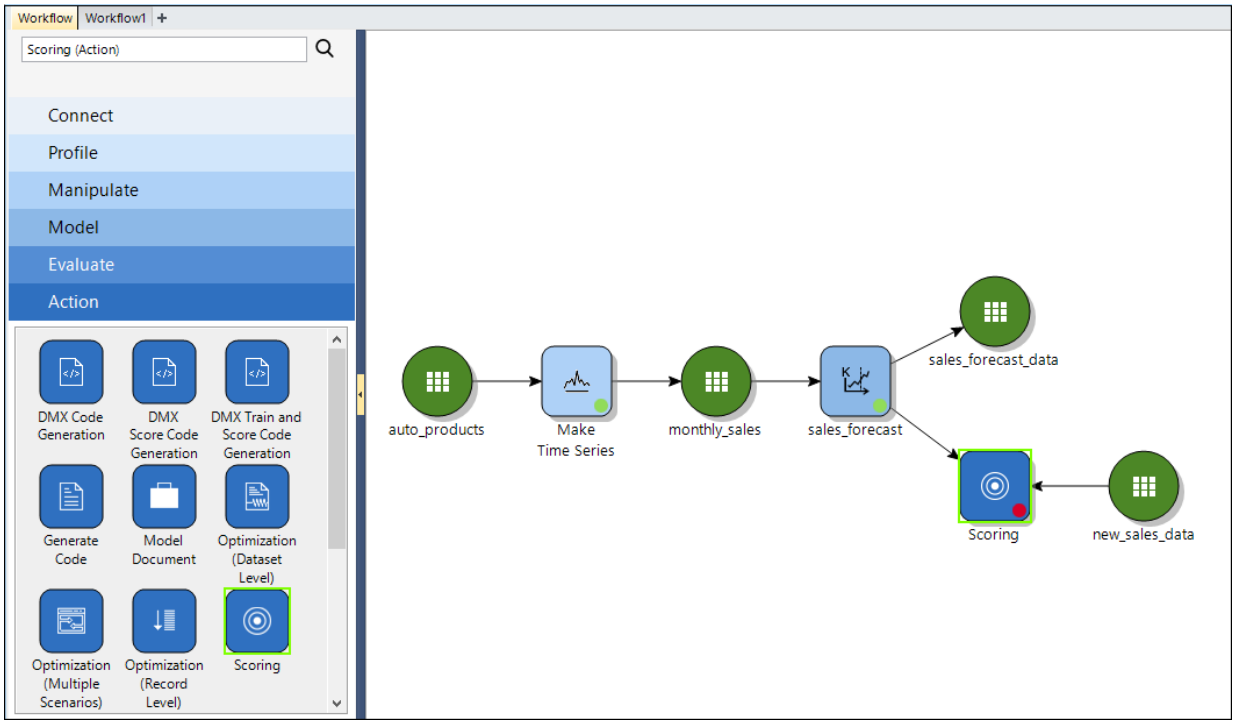
Prerequisites: a **Keras Forecasting** node and an input dataset node to score.

To add a Scoring node and apply it to a Keras Forecasting model on your workflow:

- Drag the **Scoring** node from the **Action** palette onto the Workflow canvas and connect the inputs: a **Keras Forecasting** node and a dataset node that contains the data to score. See the example below.

OR

- Right-click anywhere on the workflow canvas and select the context menu command **Action --> Scoring**. This will create a new Scoring node. Connect the inputs: a Keras Forecasting model node and a dataset that contains the data to score.

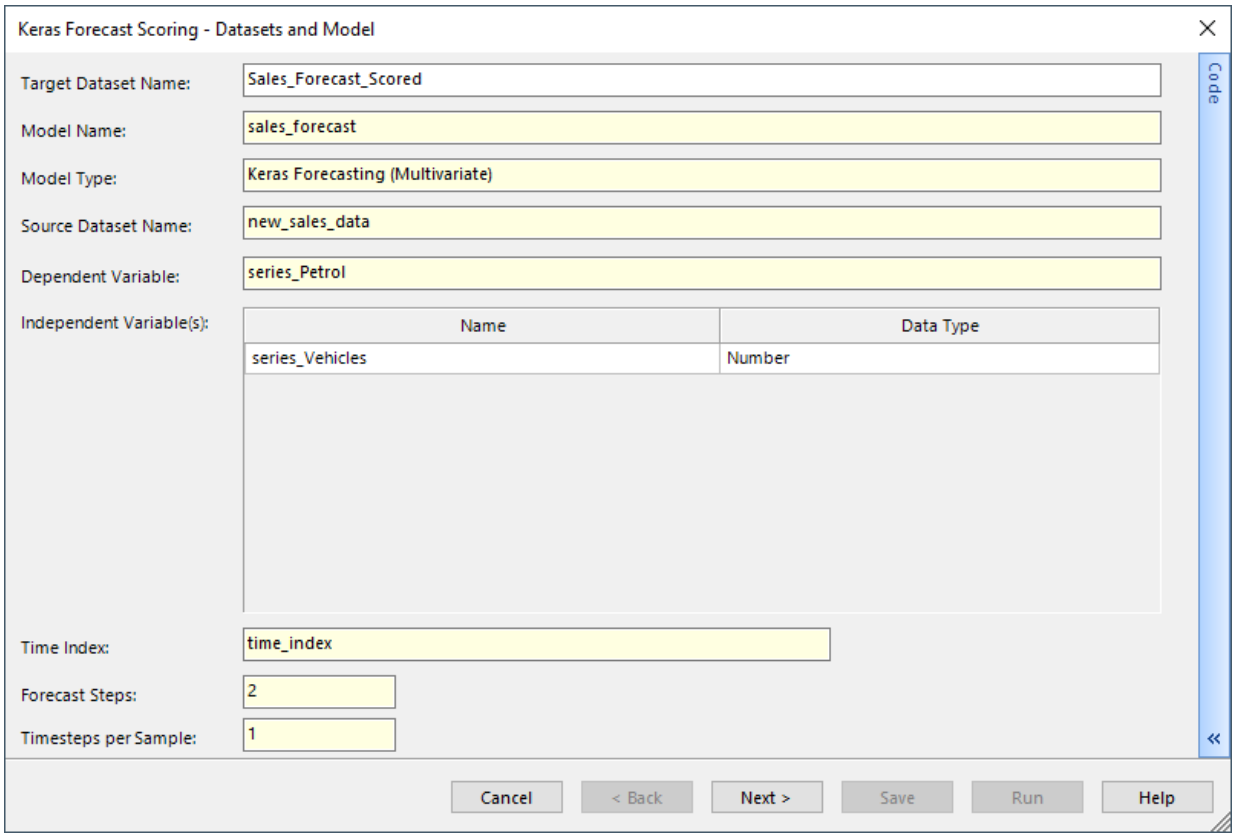


Double-click the **Scoring** node or right-click on it and select **Modify**.

The Scoring wizard opens. It displays the names of the input model and dataset, the list of dependent and independent variables and their data types.

In the *Target Dataset Name* field, specify the name for the output dataset that will contain the scores.

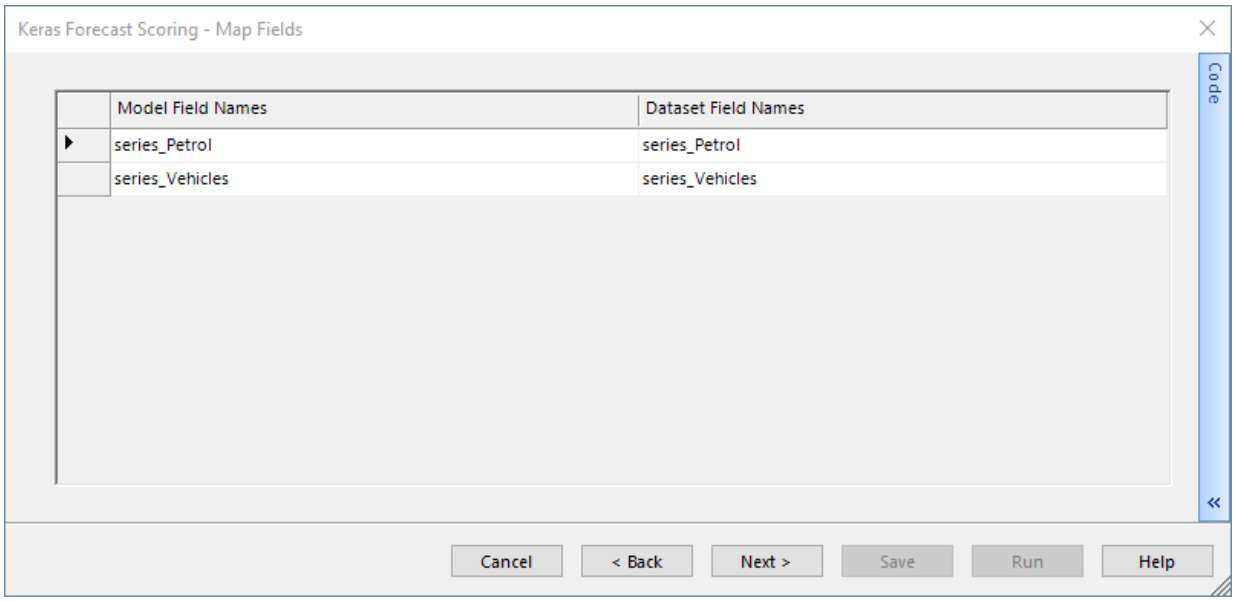
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

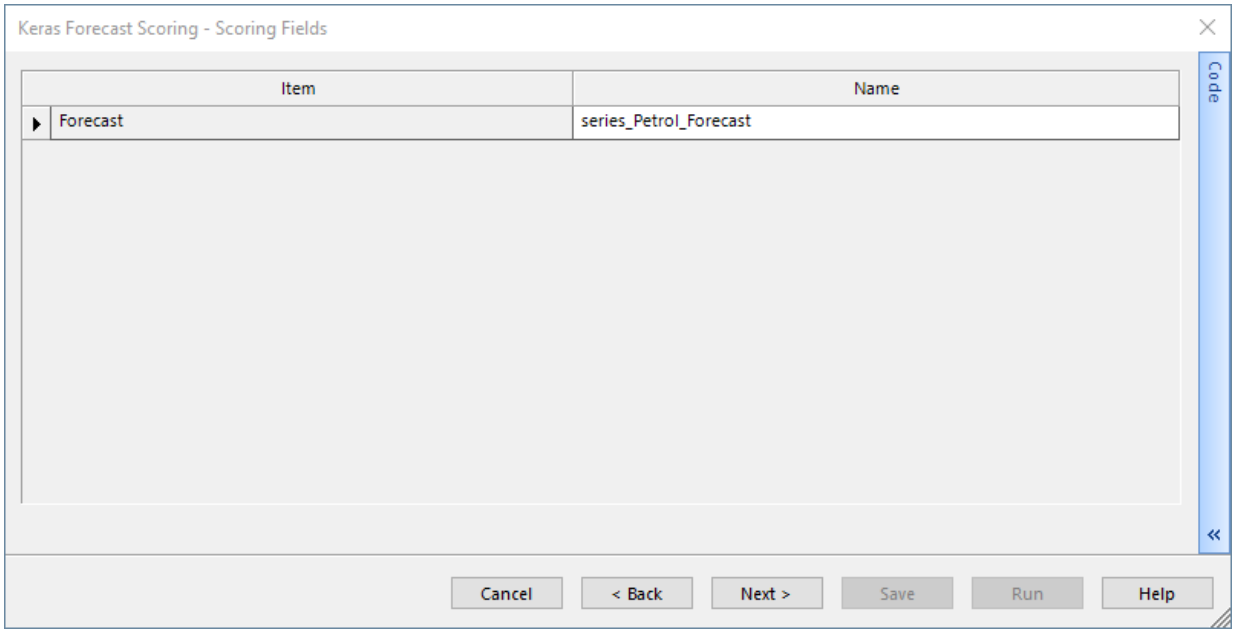
The next step provides the option of mapping the field names in the model to the ones in the scoring input dataset. In some situations, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map fields by clicking in the right column (Dataset Field Names) and selecting the right fields to match the model variables in the left column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

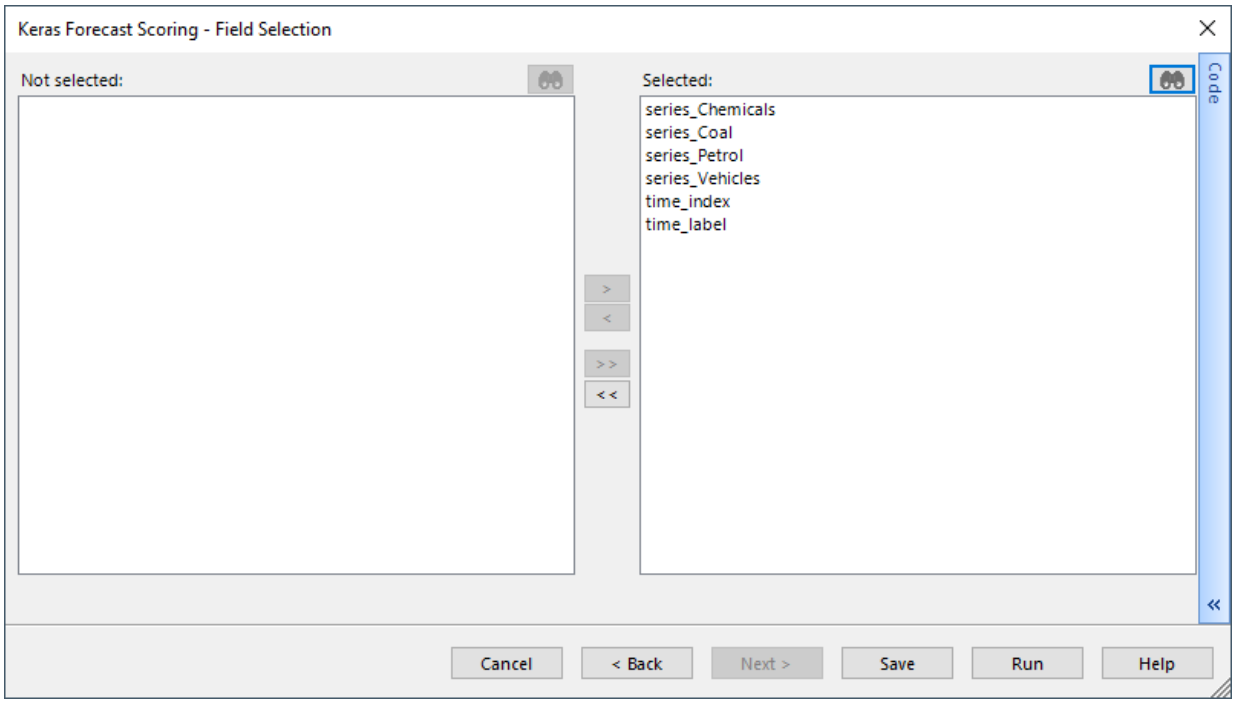
In the **Scoring Fields** page, select the scoring fields to be included in the scoring output dataset and modify their names if necessary. The default name for each dependent variable is *<DV_name>_Forecast.*



Click **Next**.

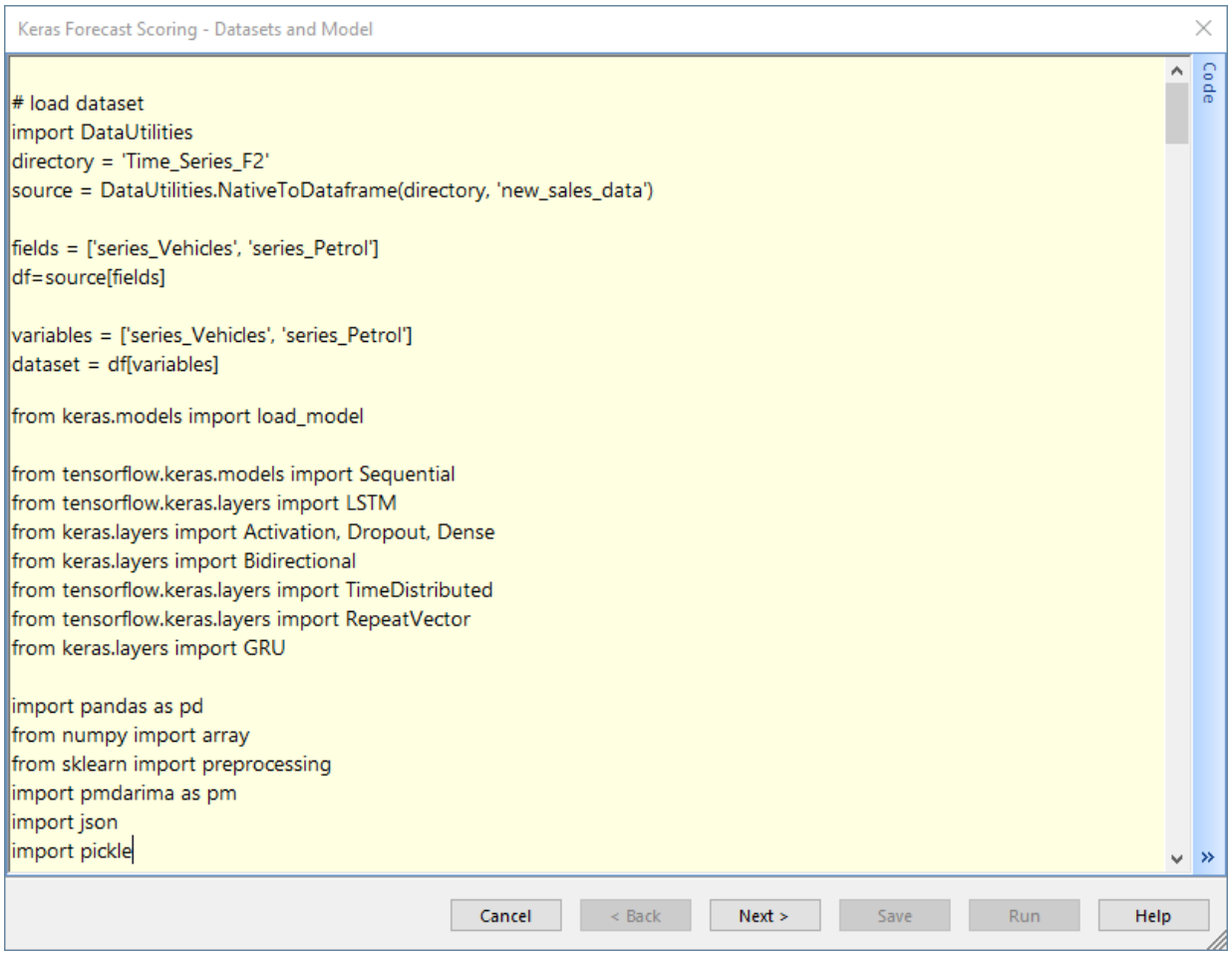
In the **Field Selection** page, choose the fields from the input dataset that you would like to include in the scoring output in addition to the new score / probability fields. By default, all fields in the scoring input dataset are selected to be included in the output.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



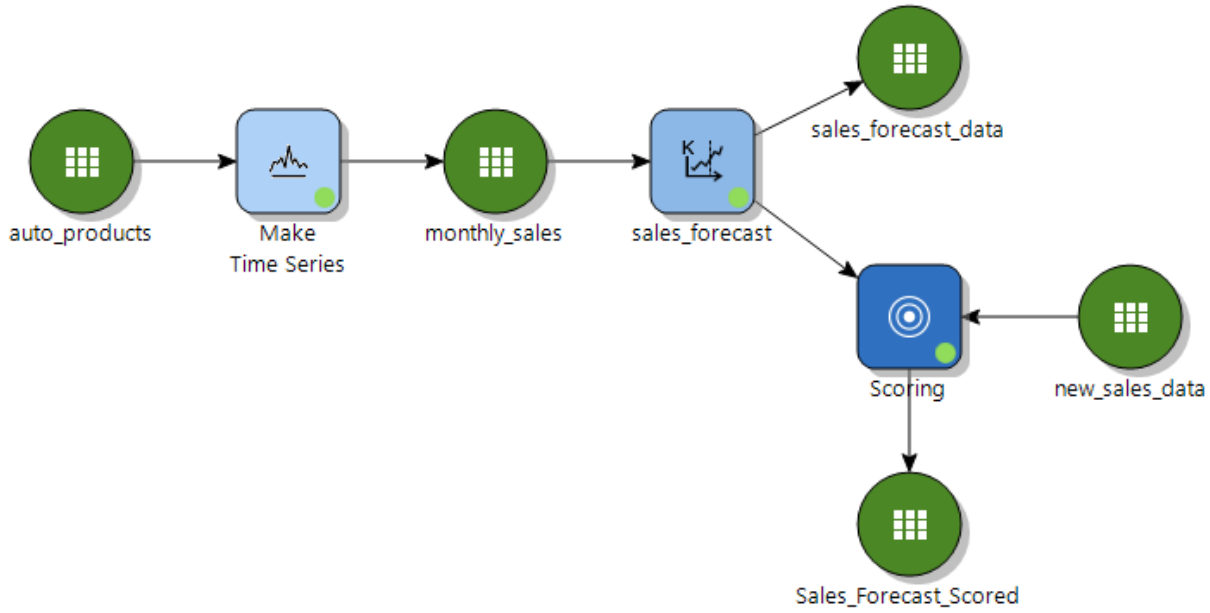
To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



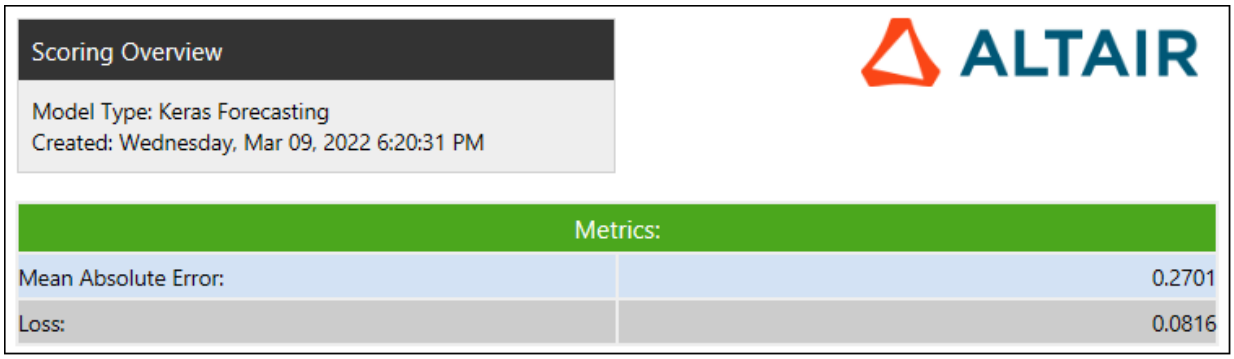
Click **Save** to save the settings, or click **Run** to save the settings and run the scoring process immediately. The model is scored in a Python session with TensorFlow backend against the input data, and the generated data with forecast scores are converted back from Pandas to Altair KDD format.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Double-click on the output dataset to view the scoring results.

The **Metrics** tab of the scoring output dataset displays an overview that includes two metrics: *Mean Absolute Error* and *Loss*:



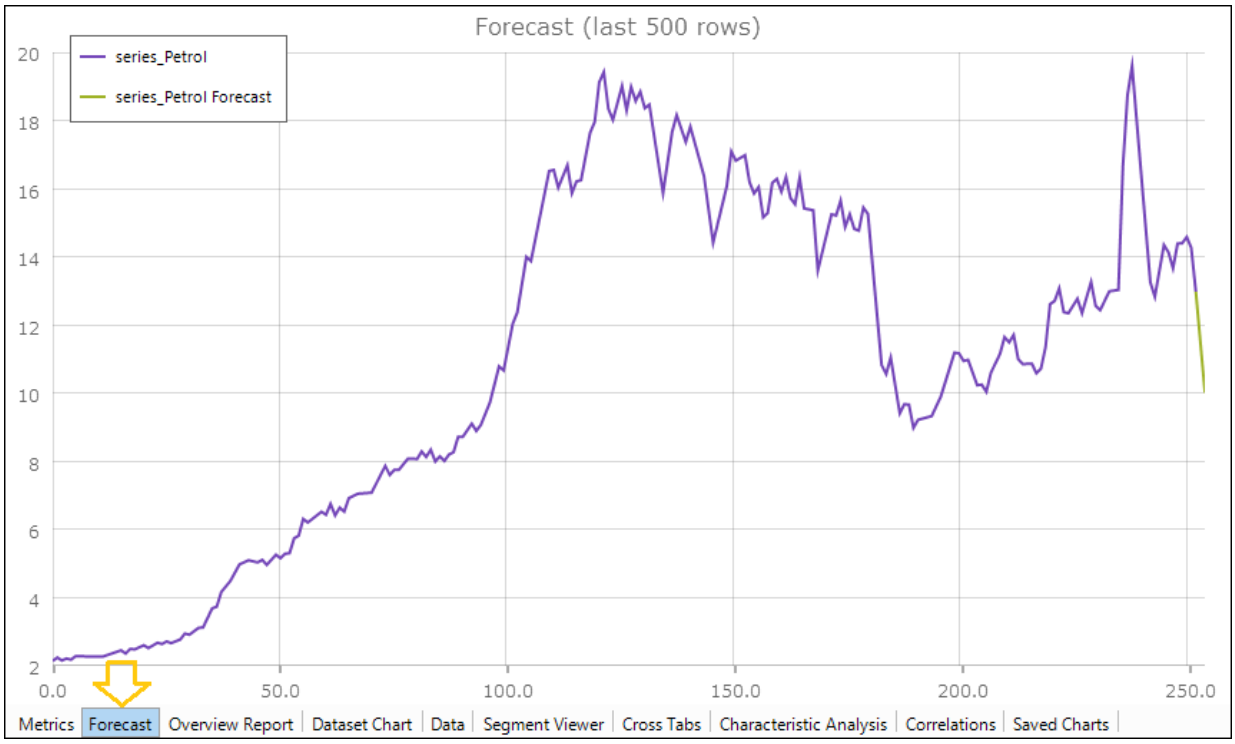
Scoring Overview

Model Type: Keras Forecasting
Created: Wednesday, Mar 09, 2022 6:20:31 PM

Metrics:	
Mean Absolute Error:	0.2701
Loss:	0.0816

Click the **Forecast** tab of the scoring output dataset to view the plot of the forecasted data produced by the scoring process.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also: Keras Forecasting, ARIMA Forecasting, ARIMA Scoring

Scoring a GLM Model

A Generalized Linear Model is scored using the **Scoring** node in the Action palette. For general information about scoring, see the topic About Scoring.

Prerequisites for Scoring: an **Sklearn GLM** model node and an input dataset to score.

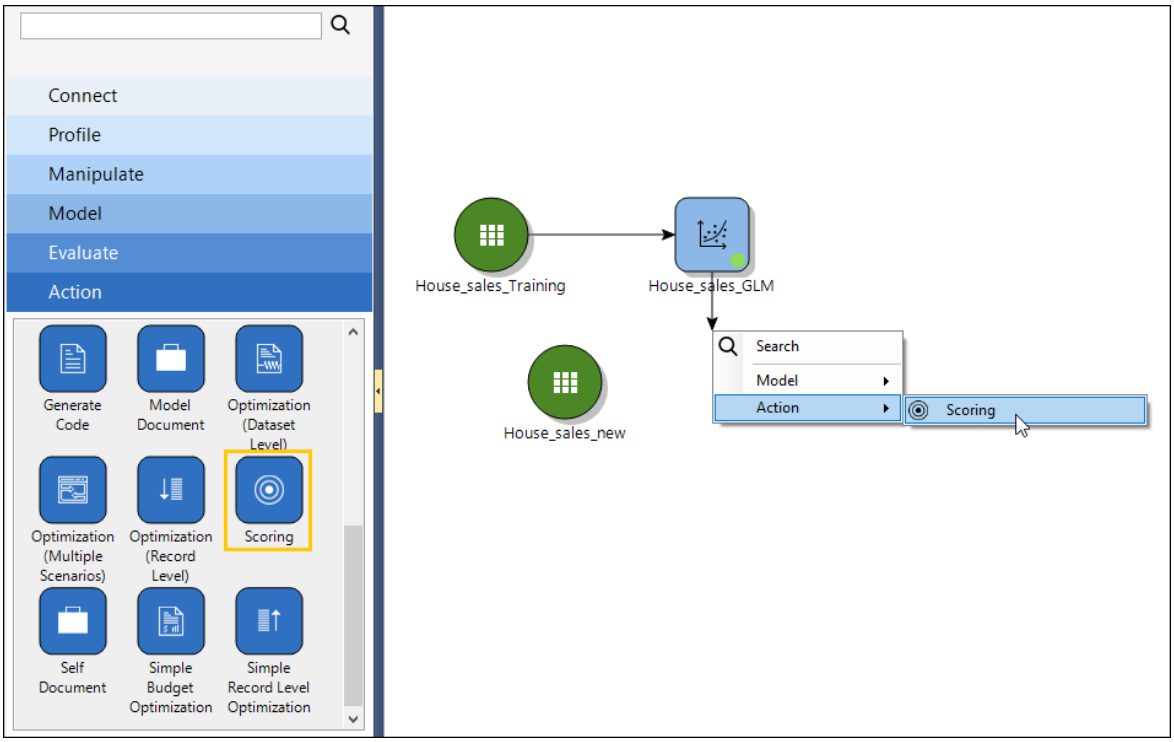
To add an Scoring operation to your workflow:

- Drag the **Scoring** node from the **Action** palette onto the Workflow canvas and connect the inputs: an **Sklearn GLM** model node and a dataset node that contains the data to score.

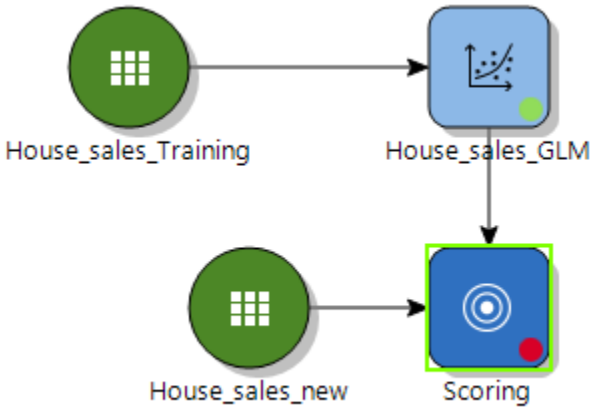
  OR

- Right-click anywhere on the workflow canvas and select the context menu command **Action --> Scoring**. This will create a new Scoring node. Connect the inputs: an **Sklearn GLM** node and a dataset node that contains the data to score.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



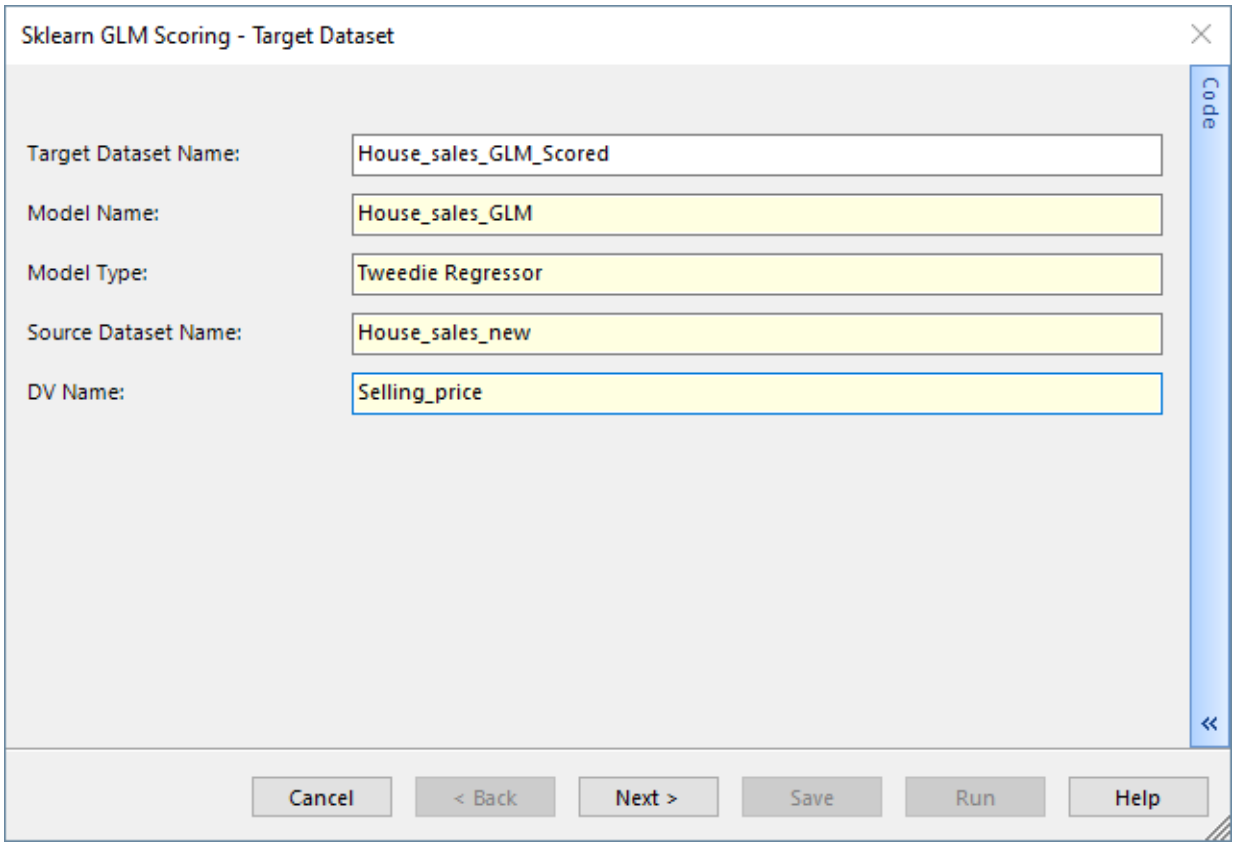
Double-click the **Scoring** node or right-click on it and select **Modify**.



The Scoring wizard opens. It displays the names of the input model and dataset, the regressor type, and the name of the dependent variable.

In the *Target Dataset Name* field, specify the name for the output dataset that will contain the scores.

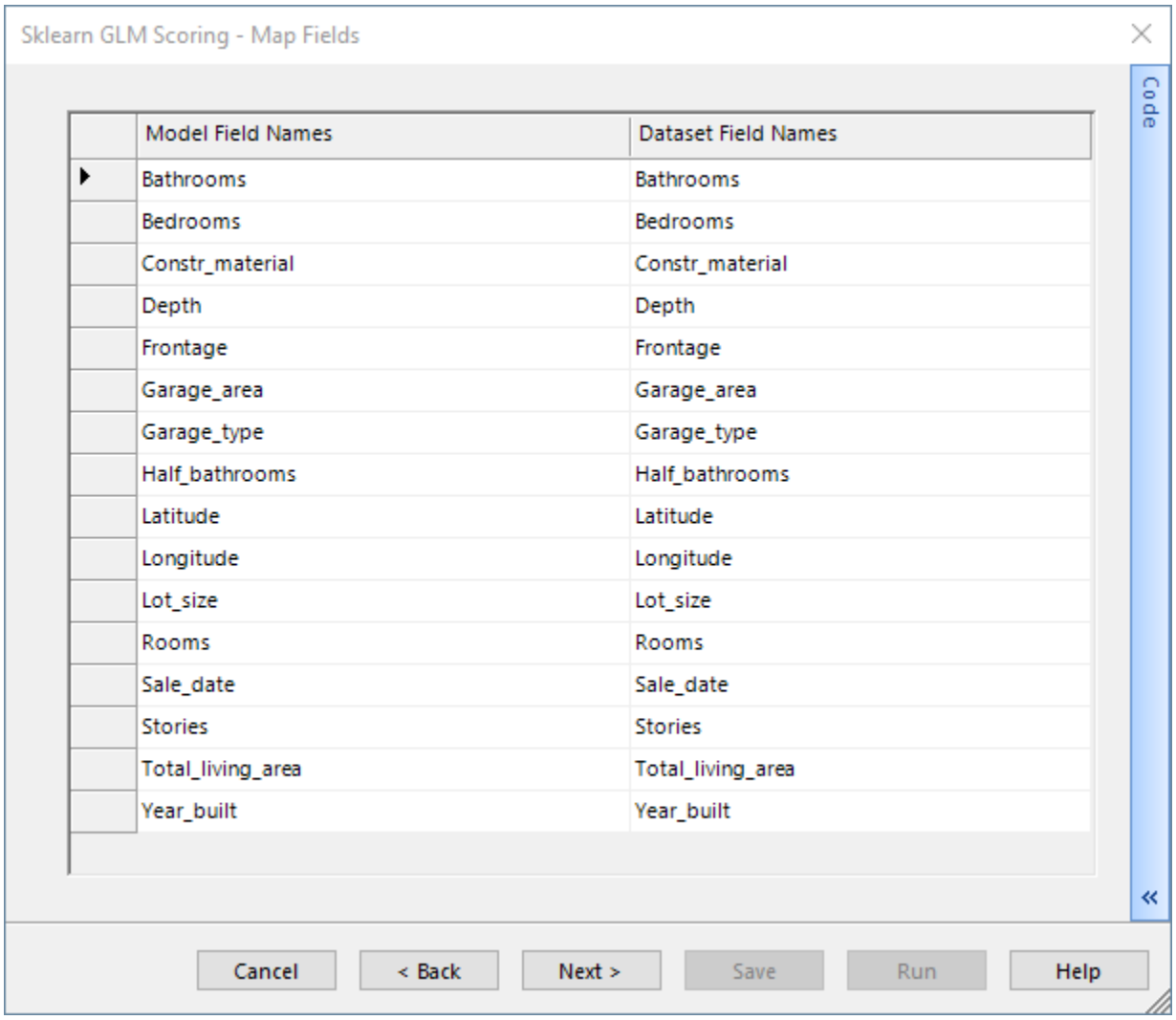
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

The next step provides the option of mapping the field names in the model to the ones in the scoring input dataset. In some situations, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map fields by clicking in the right column (Dataset Field Names) and selecting the right fields to match the model variables in the left column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

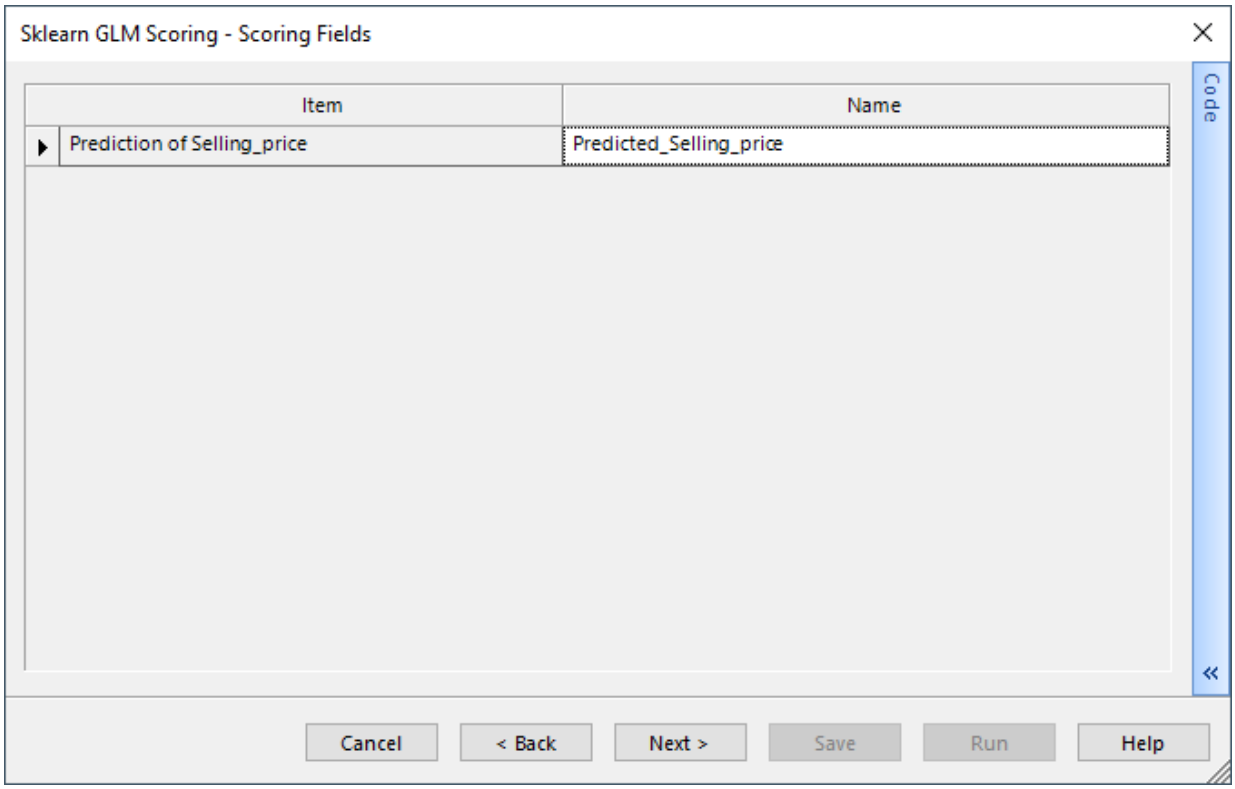


Note: When scoring GLM models that use derived variables (also called calculated columns) with expressions, the input dataset must include these variables. This is different from the case of native Altair models with derived variables, which can be scored on a dataset with only *original* variables that the derived ones depend on, while the derived ones are not required.

Click **Next**.

In the **Scoring Fields** page, review the scoring fields to be produced by the model and modify their names if necessary.

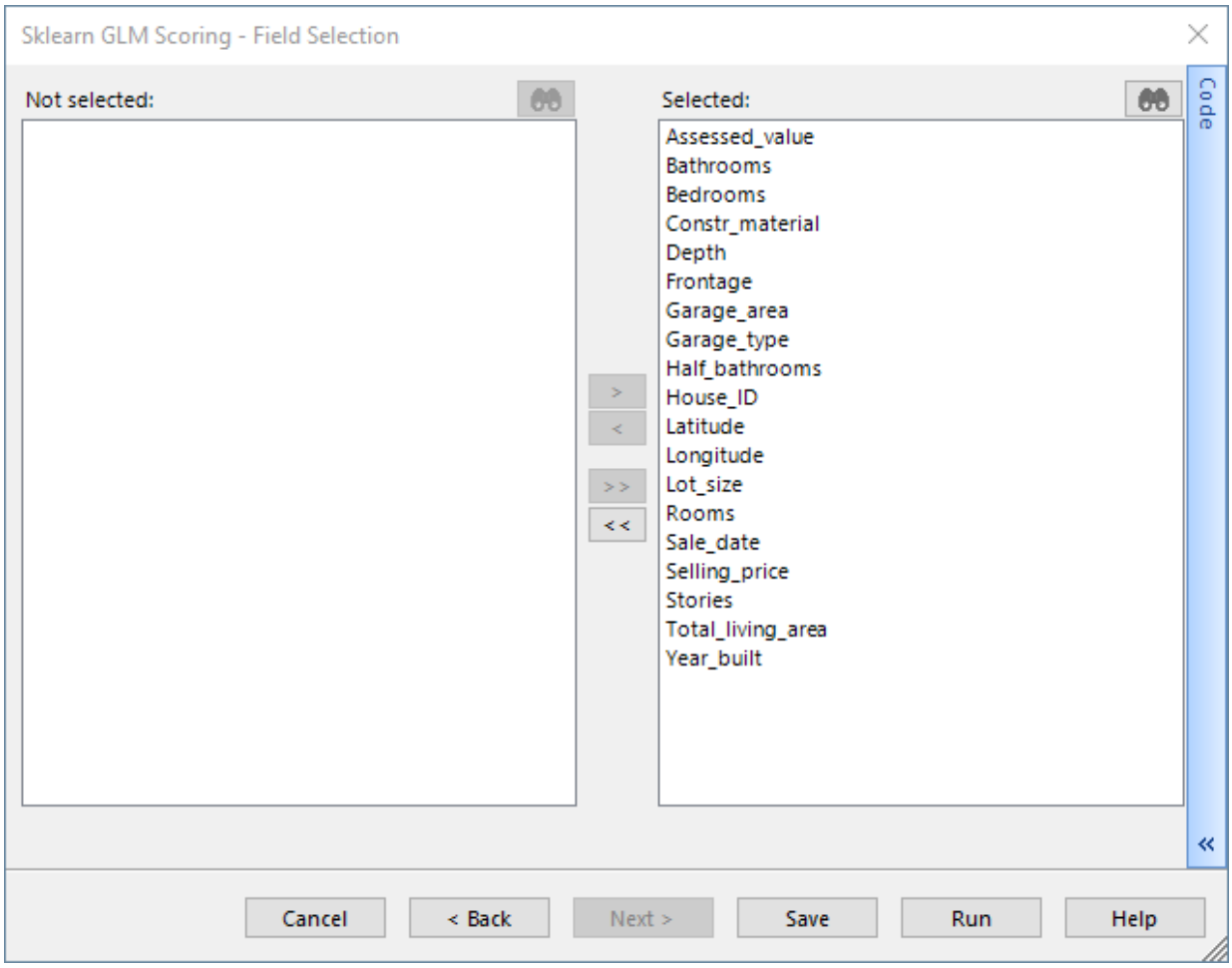
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

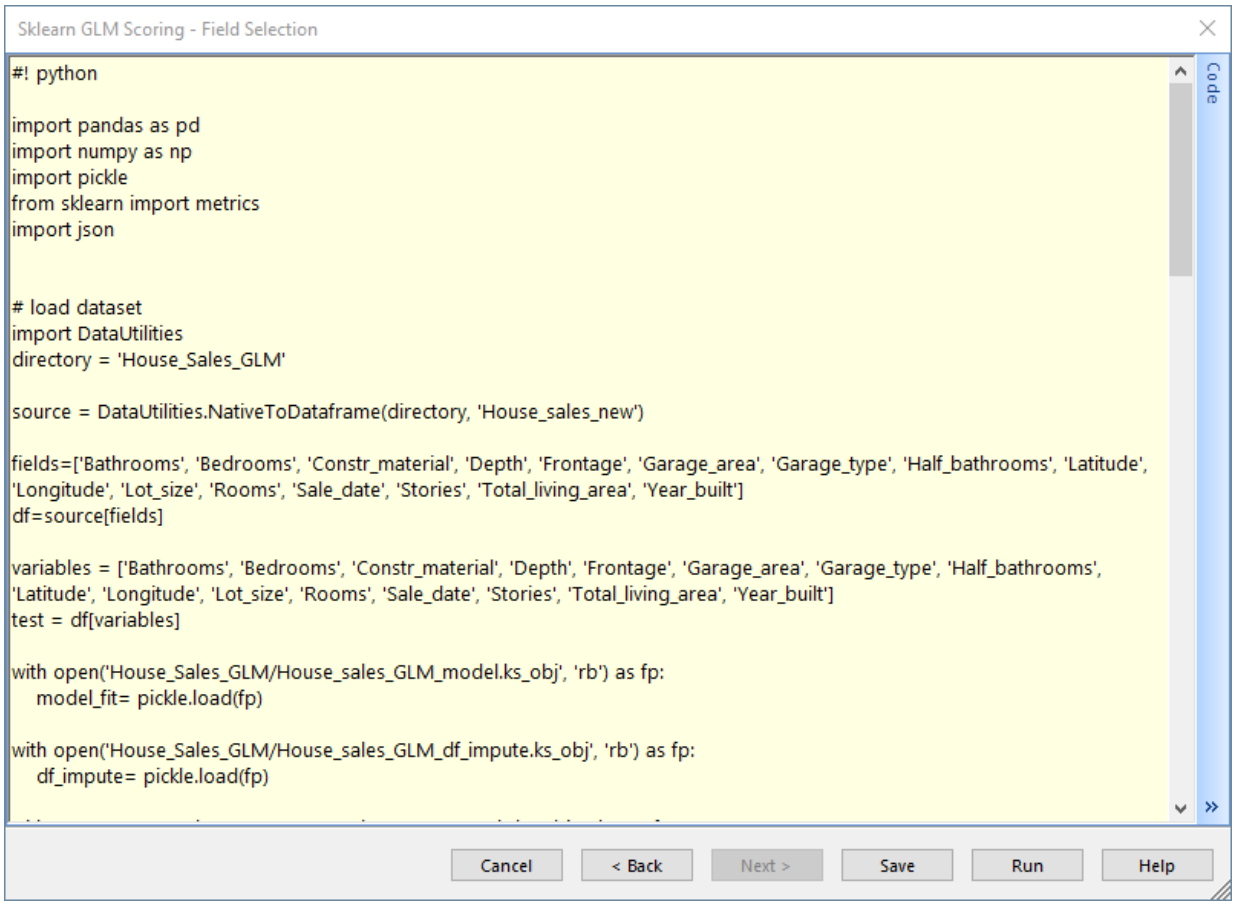
In the **Field Selection** page, choose the fields from the input dataset that you would like to include in the scoring output in addition to the new score / probability fields. By default, all fields in the scoring input dataset are selected to be included in the output.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

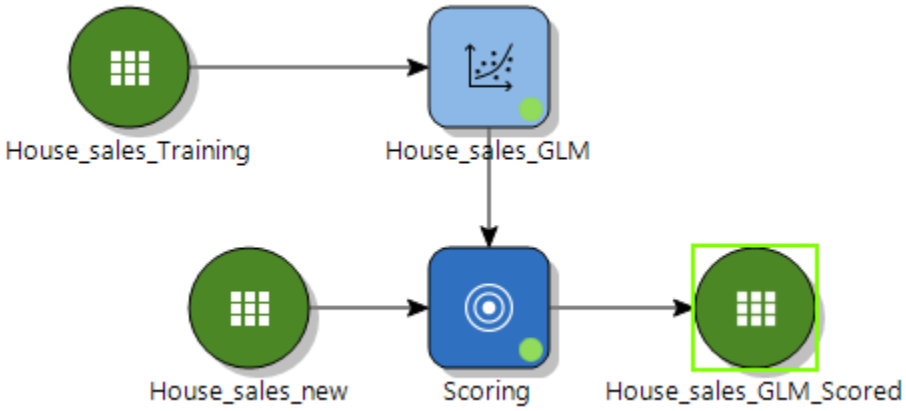


To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

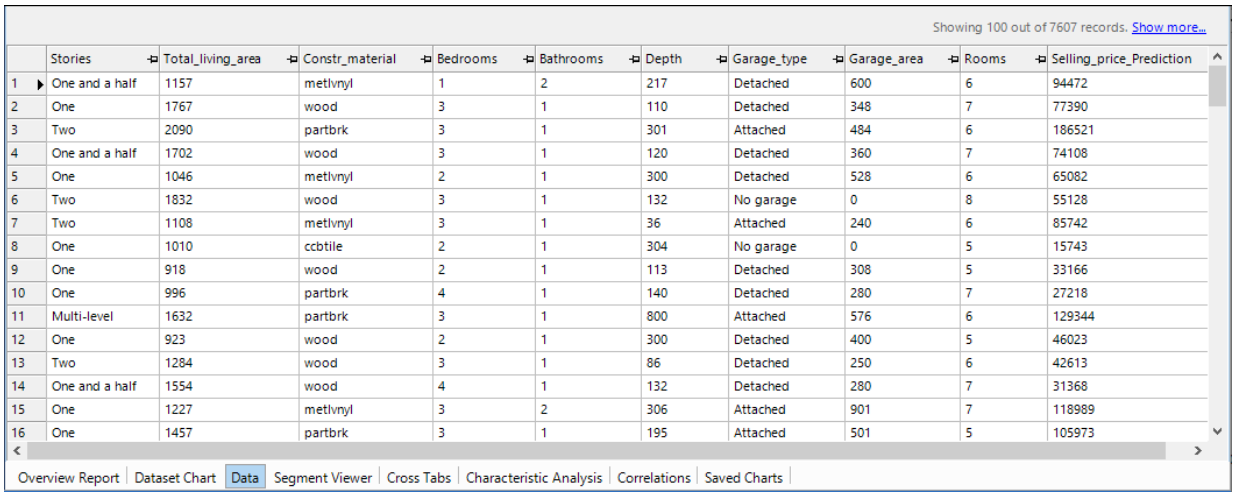


Click **Save** to save the settings, or click **Run** to save the settings and run the scoring process immediately. The model is scored in a Python container session, and then the dataframe with generated predictions (scores) is converted back from Pandas to Altair KDD format. The output dataset node is created in the workflow. Double-click the output dataset to view the results.



Click the Data tab of the output dataset to see the model predictions in the last column.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Showing 100 out of 7607 records. Show more...

	Stories	Total_living_area	Constr_material	Bedrooms	Bathrooms	Depth	Garage_type	Garage_area	Rooms	Selling_price_Prediction
1	One and a half	1157	metlvnyl	1	2	217	Detached	600	6	94472
2	One	1767	wood	3	1	110	Detached	348	7	77390
3	Two	2090	partbrk	3	1	301	Attached	484	6	186521
4	One and a half	1702	wood	3	1	120	Detached	360	7	74108
5	One	1046	metlvnyl	2	1	300	Detached	528	6	65082
6	Two	1832	wood	3	1	132	No garage	0	8	55128
7	Two	1108	metlvnyl	3	1	36	Attached	240	6	85742
8	One	1010	ccbtile	2	1	304	No garage	0	5	15743
9	One	918	wood	2	1	113	Detached	308	5	33166
10	One	996	partbrk	4	1	140	Detached	280	7	27218
11	Multi-level	1632	partbrk	3	1	800	Attached	576	6	129344
12	One	923	wood	2	1	300	Detached	400	5	46023
13	Two	1284	wood	3	1	86	Detached	250	6	42613
14	One and a half	1554	wood	4	1	132	Detached	280	7	31368
15	One	1227	metlvnyl	3	2	306	Attached	901	7	118989
16	One	1457	partbrk	3	1	195	Attached	501	5	105973

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

See also:

- Generalized Linear Model
- Sklearn GLM Validation

Scoring an XGBoost Model

An Sklearn XGBoost Model is scored using the **Scoring** node in the Action palette. For general information about scoring, see the topic About Scoring.

Prerequisites for Scoring: an **Sklearn XGB** model node and an input dataset to score.

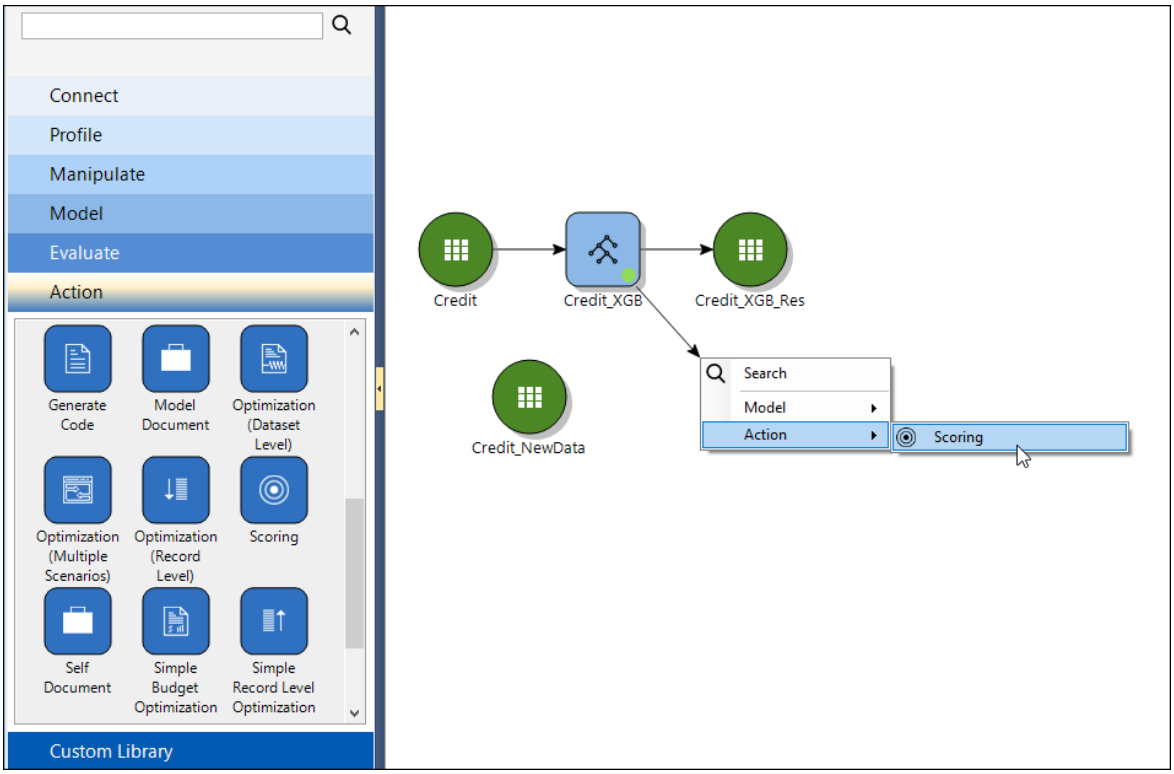
To add a Scoring operation to your workflow:

- Drag the **Scoring** node from the **Action** palette onto the Workflow canvas and connect the inputs: an **Sklearn XGB** model node and a dataset node that contains the data to score.

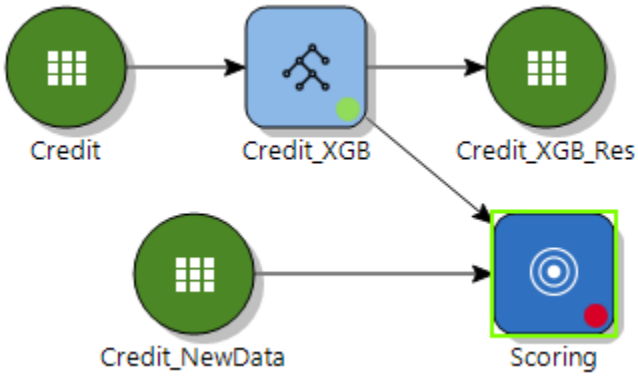
  OR

- Right-click anywhere on the workflow canvas and select the context menu command **Action --> Scoring**. This will create a new Scoring node. Connect the inputs: an **Sklearn XGB** node and a dataset node that contains the data to score.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



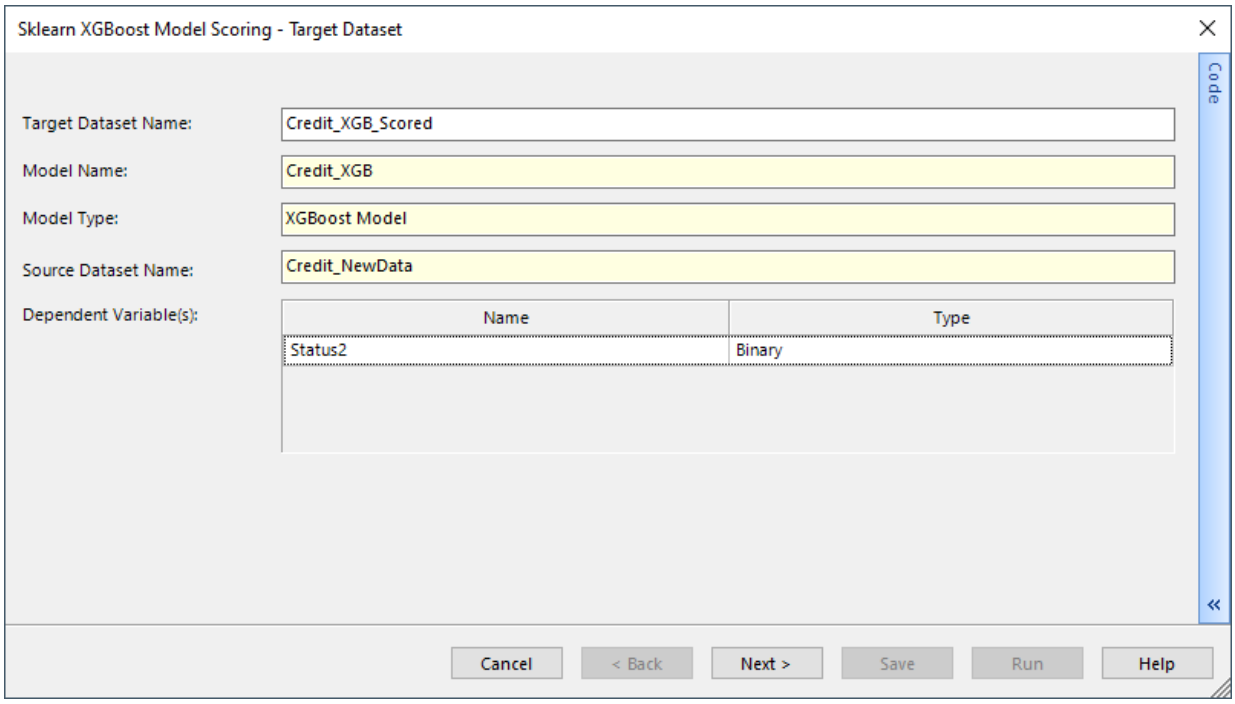
Double-click the **Scoring** node or right-click on it and select **Modify**.



The Scoring wizard opens. It displays the names of the input model and dataset, the regressor type, and the name of the dependent variable.

In the *Target Dataset Name* field, specify the name for the output dataset that will contain the scores.

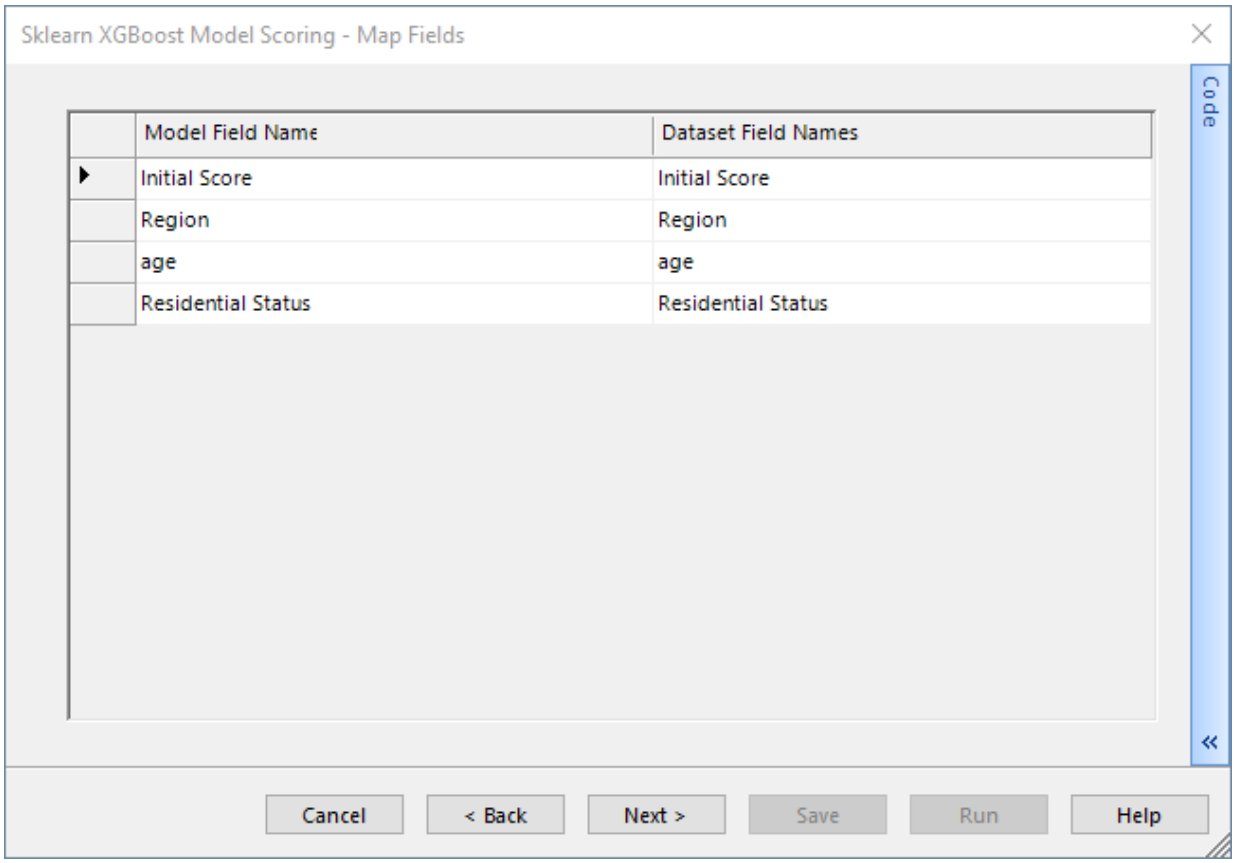
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

The next step provides the option of mapping the field names in the model to the ones in the validation input dataset. In some situations, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map fields by clicking in the right column (Dataset Field Names) and selecting the right fields to match the model variables in the left column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

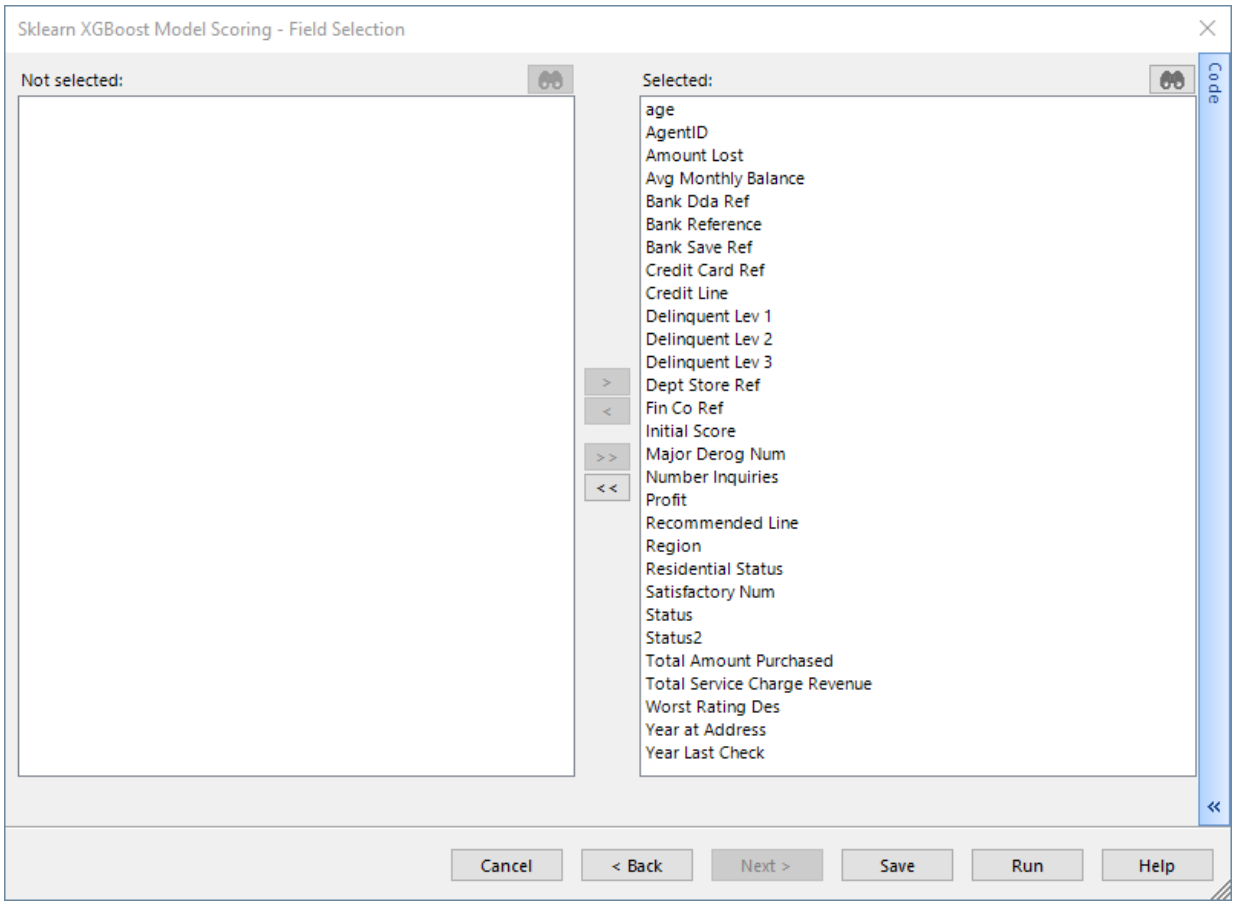


Note: When scoring XGB models that use derived variables (also called calculated columns) with expressions, the input dataset must include these variables. This is different from the case of native Altair models with derived variables, which can be scored on a dataset with only *original* variables that the derived ones depend on, while the derived ones are not required.

Click **Next**.

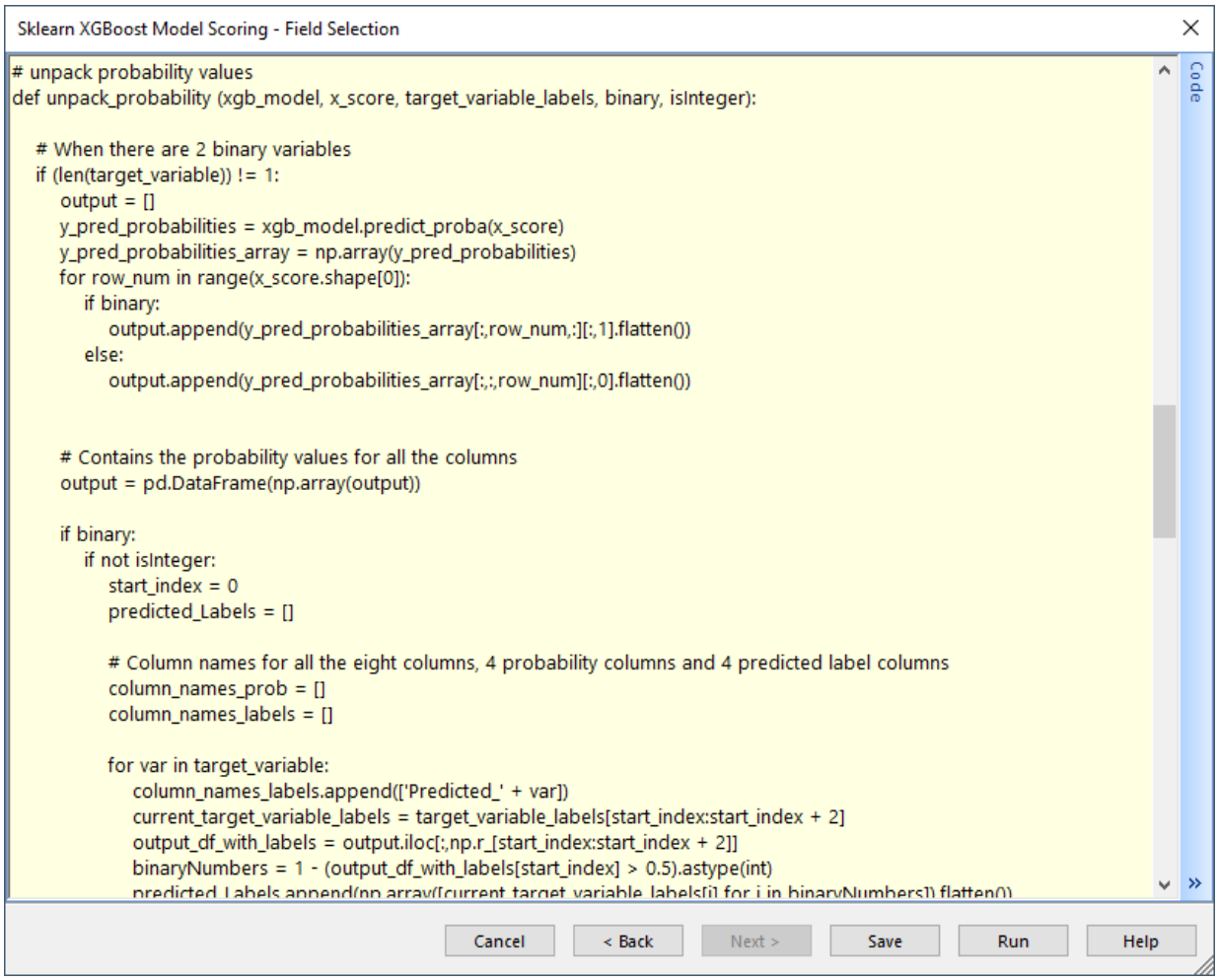
In the **Field Selection** page, choose the fields from the input dataset that you would like to include in the scoring output in addition to the new score fields. By default, all fields in the scoring input dataset are selected to be included in the output.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

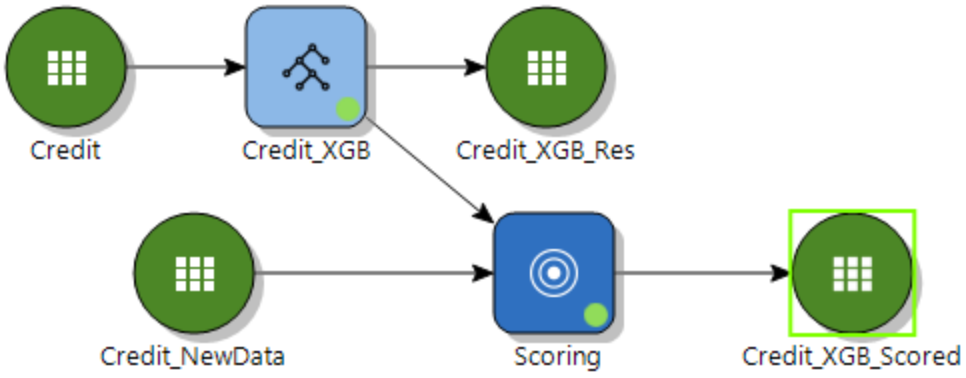


To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

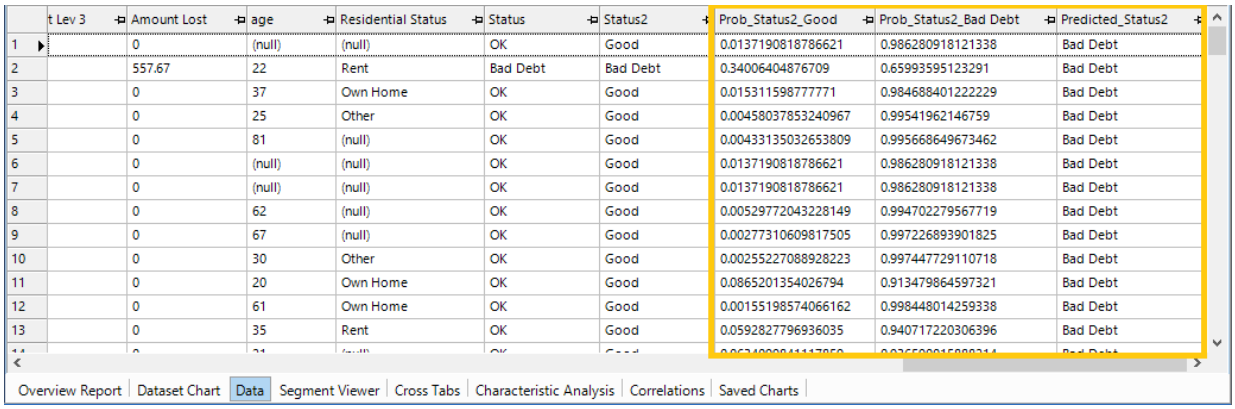


Click **Save** to save the settings, or click **Run** to save the settings and run the scoring process immediately. The model is scored in a Python container session, and then the dataframe with generated predictions (scores) is converted back from Pandas to Altair KDD format. The output dataset node is created in the workflow. Double-click the output dataset to view the results.



Click the Data tab of the output dataset to see the probabilities of the dependent variable(s) and model predictions.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



	t Lev 3	Amount Lost	age	Residential Status	Status	Status2	Prob_Status2_Good	Prob_Status2_Bad Debt	Predicted_Status2
1		0	(null)	(null)	OK	Good	0.0137190818786621	0.986280918121338	Bad Debt
2		557.67	22	Rent	Bad Debt	Bad Debt	0.34006404876709	0.65993595123291	Bad Debt
3		0	37	Own Home	OK	Good	0.015311598777771	0.984688401222229	Bad Debt
4		0	25	Other	OK	Good	0.00458037853240967	0.99541962146759	Bad Debt
5		0	81	(null)	OK	Good	0.00433135032653809	0.995668649673462	Bad Debt
6		0	(null)	(null)	OK	Good	0.0137190818786621	0.986280918121338	Bad Debt
7		0	(null)	(null)	OK	Good	0.0137190818786621	0.986280918121338	Bad Debt
8		0	62	(null)	OK	Good	0.00529772043228149	0.994702279567719	Bad Debt
9		0	67	(null)	OK	Good	0.00277310609817505	0.997226893901825	Bad Debt
10		0	30	Other	OK	Good	0.00255227088928223	0.997447729110718	Bad Debt
11		0	20	Own Home	OK	Good	0.0865201354026794	0.913479864597321	Bad Debt
12		0	61	Own Home	OK	Good	0.00155198574066162	0.998448014259338	Bad Debt
13		0	35	Rent	OK	Good	0.0592827796936035	0.940717220306396	Bad Debt

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

See also: Sklearn XGBoost Model

ARIMA Scoring

Scoring an ARIMA Forecasting model is similar to scoring other models, but it produces forecasted data points instead of probabilities. An ARIMA Forecasting model is scored using the **Scoring** node in the Action palette. For general information about scoring, see the topic About Scoring.

Prerequisites for ARIMA model scoring: an **ARIMA Forecasting** node and an input dataset to score. It is assumed that the input dataset is a ***time series dataset*** containing metadata that identifies one column as the *time index* (series ID) and another column as the time label. Such datasets can be created, for example, using the Make Time Series node in Knowledge Studio. These datasets have special views (integrated tabs) where time series charts and autocorrelation charts van be plotted.

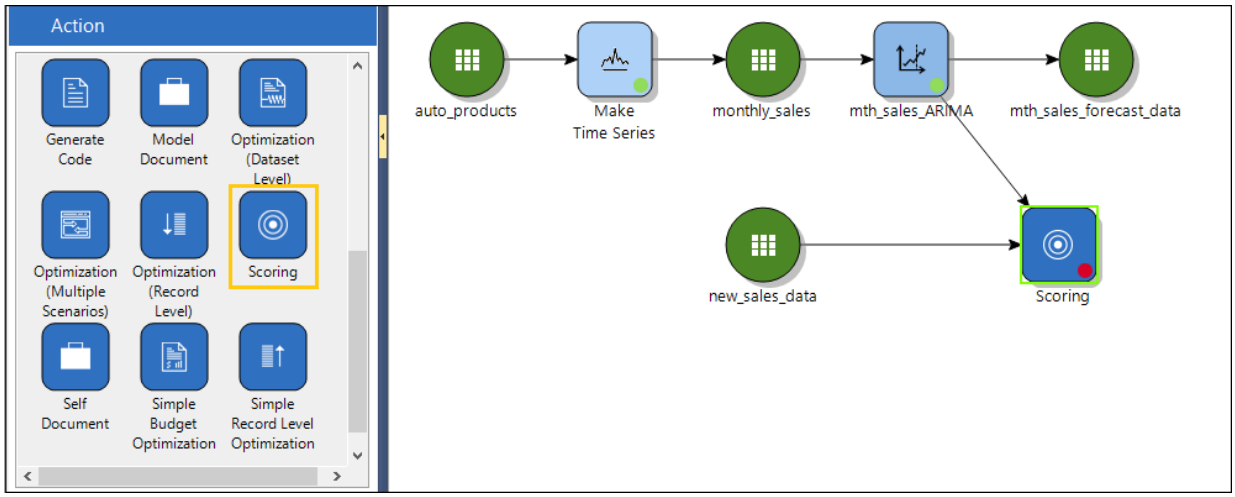
To add a Scoring operation to your workflow:

- Drag the **Scoring** node from the **Action** palette onto the Workflow canvas and connect the inputs: an **ARIMA Forecasting** node and a dataset node that contains the data to score, as shown below.

  OR

- Right-click anywhere on the workflow canvas and select the context menu command **Action --> Scoring**. This will create a new Scoring node. Connect the inputs: an **ARIMA Forecasting** node and a dataset node that contains the data to score.

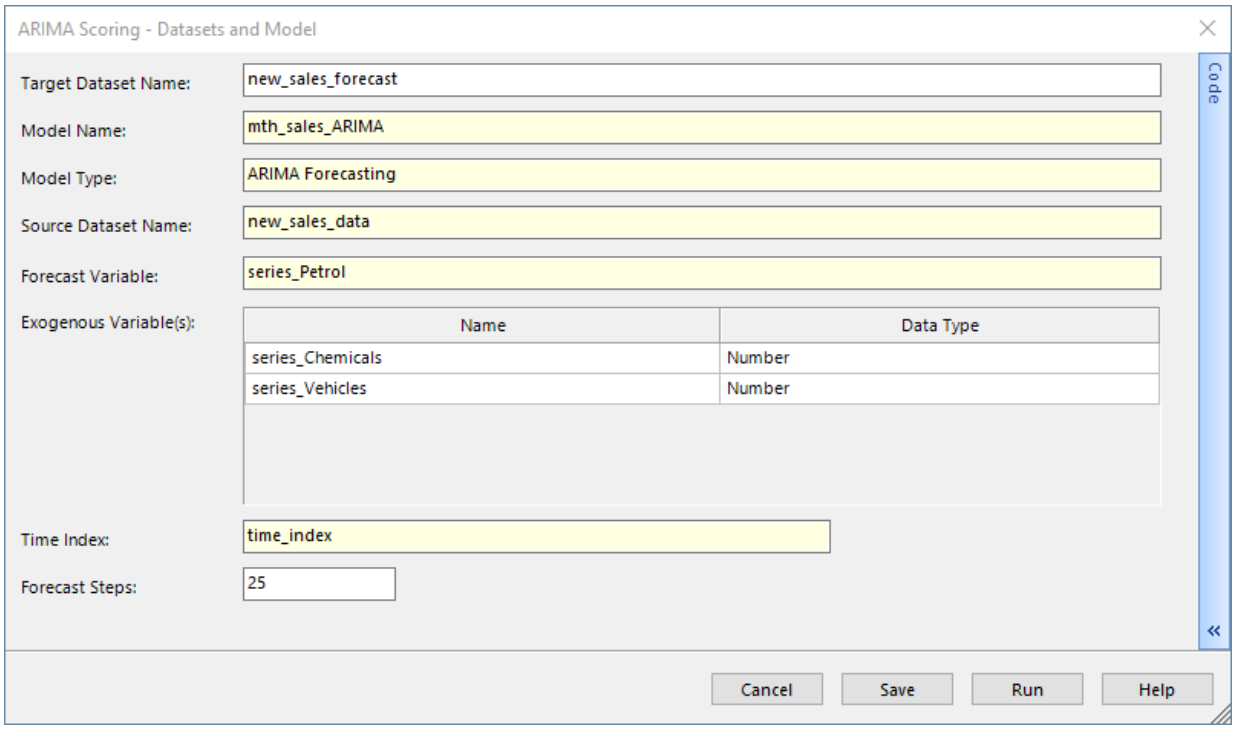
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Double-click the **Scoring** node or right-click on it and select **Modify**.

The Scoring wizard opens. It displays the names of the input model and dataset, the forecast variable, the list of exogenous variables and their data types.

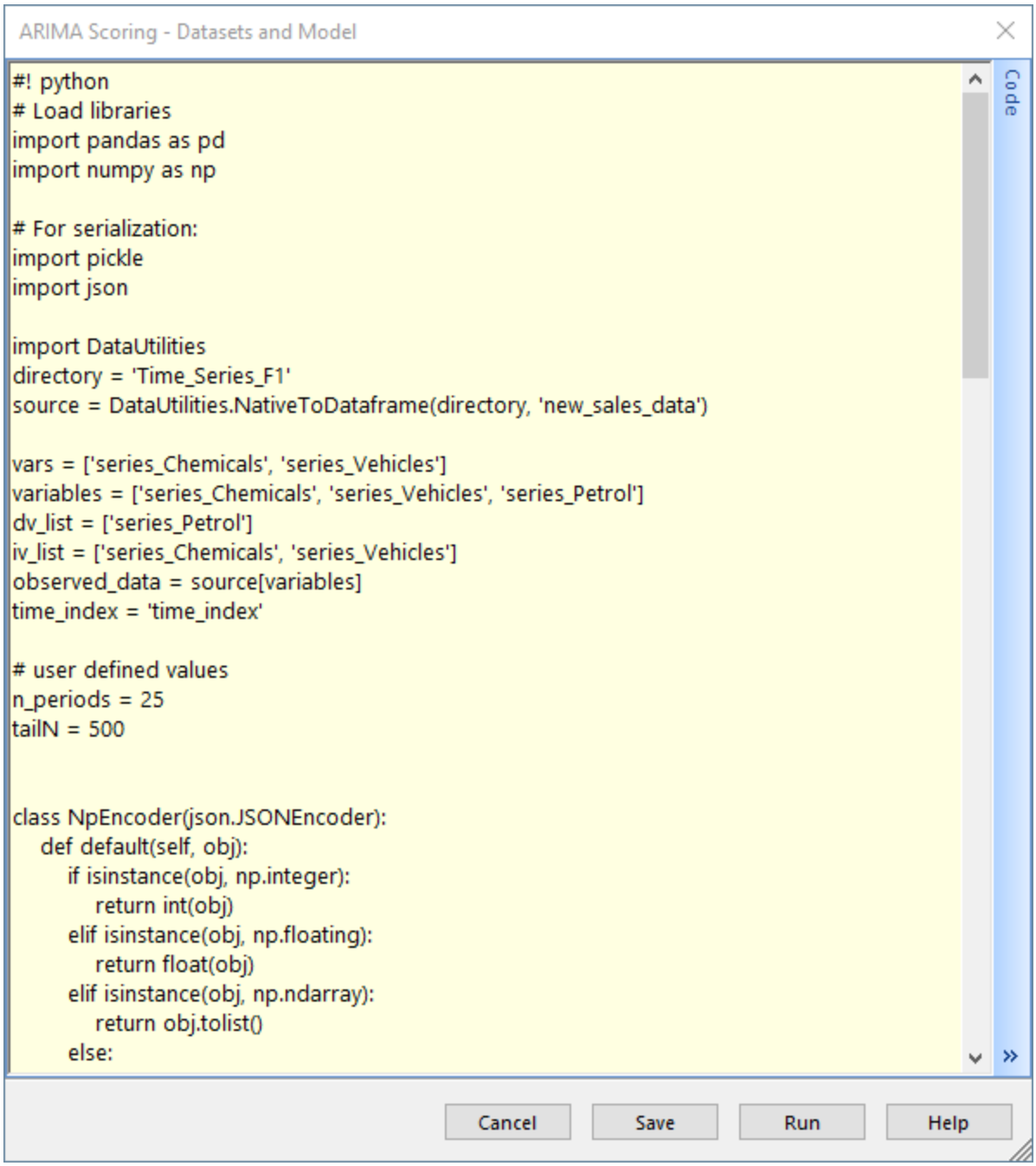
- In the *Target Dataset Name* field, specify the name for the output dataset that will contain the forecasted data.
- Specify the number of *Forecast Steps* (periods ahead to forecast). The default is 25.



To preview the Python code generated for your inputs, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview frame. When viewing the

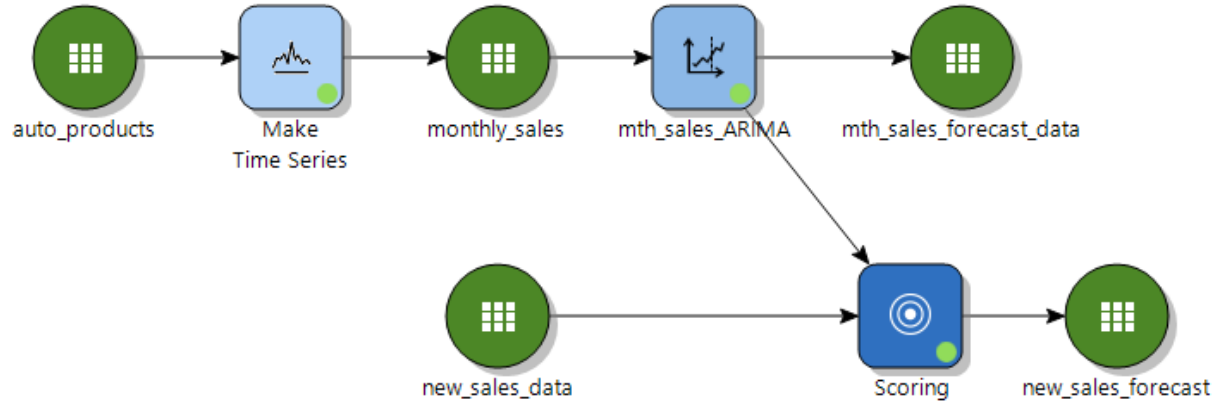
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview frame.



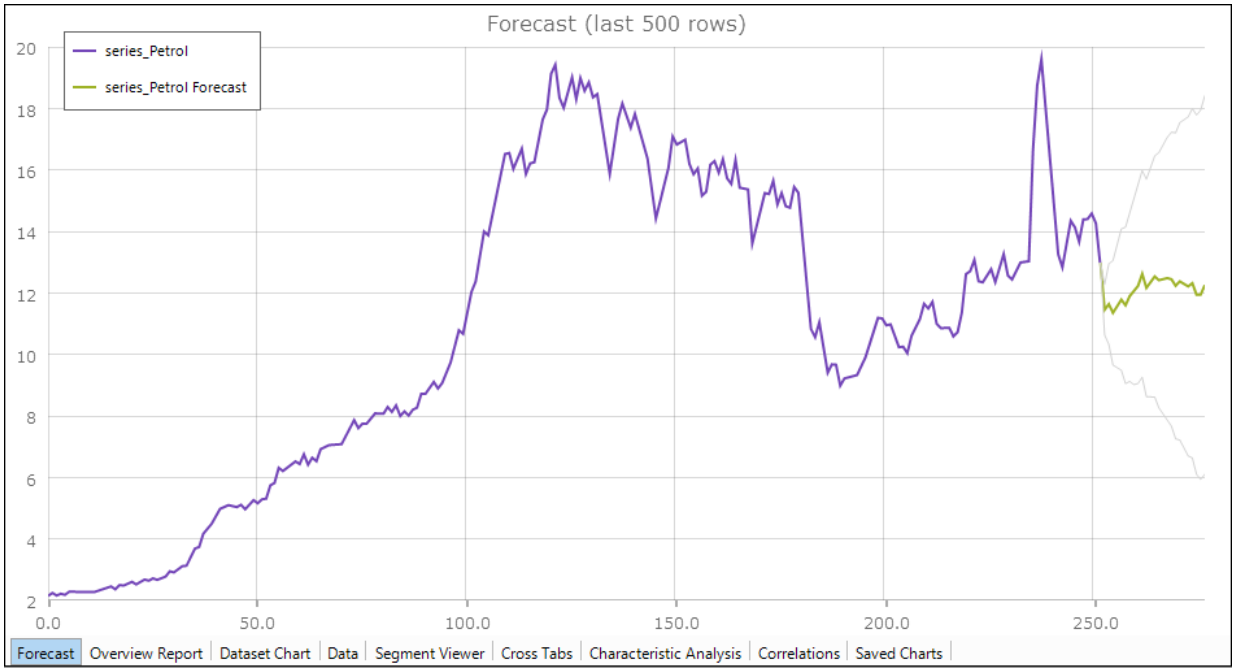
Click **Save** to save the settings, or click **Run** to save the settings and run the scoring process immediately. The model is scored in a Python container session against the input data, the generated forecast data is converted back from Pandas to Altair KDD format and shown as the output dataset node on the workflow canvas.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



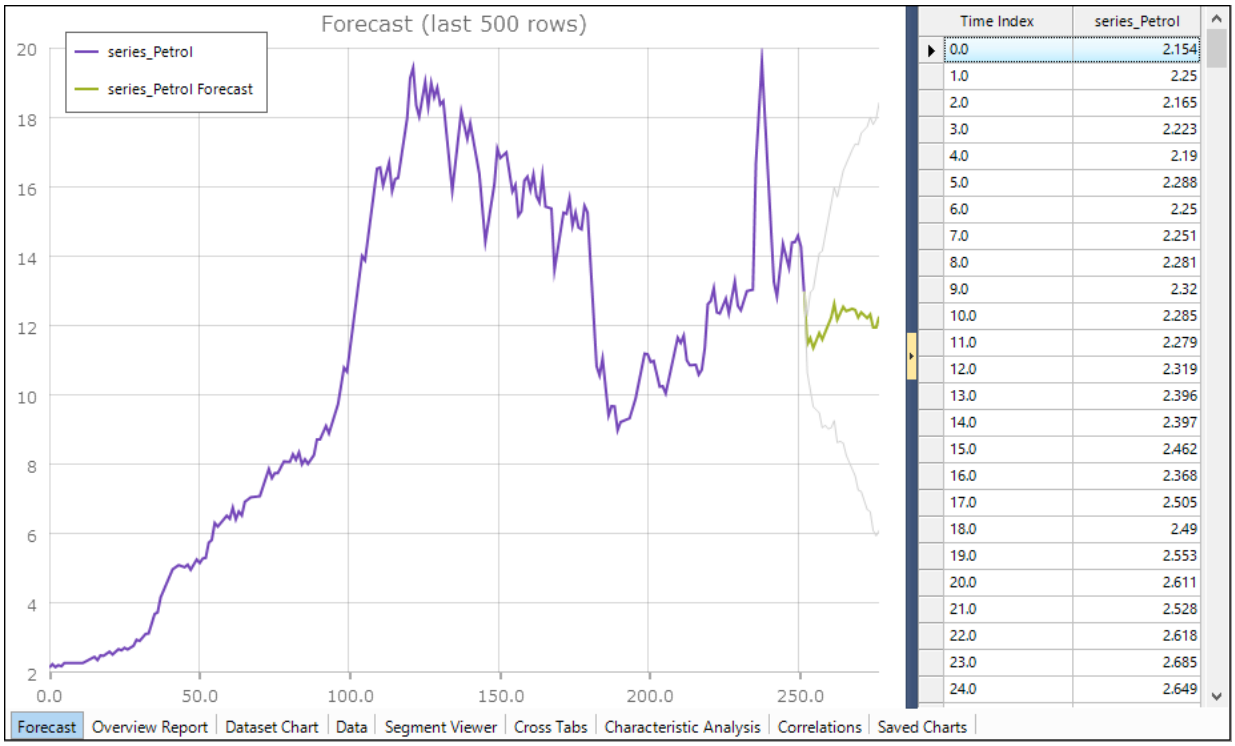
Double-click on the output dataset to view the forecast plot and data.

Click the **Forecast** tab. The plot shows the input data points extended with forecast data for the number of forecast steps specified in the wizard. The legend at the top left shows the color codes for the input data and forecasted data for each series.



To view the underlying data of the forecast plot, click on the small arrow in the middle of the blue bar on the right-hand side. The data table will open in a frame on the right. Click the arrow again to hide the frame.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Time Index	series_Petrol
0.0	2.154
1.0	2.25
2.0	2.165
3.0	2.223
4.0	2.19
5.0	2.288
6.0	2.25
7.0	2.251
8.0	2.281
9.0	2.32
10.0	2.285
11.0	2.279
12.0	2.319
13.0	2.396
14.0	2.397
15.0	2.462
16.0	2.368
17.0	2.505
18.0	2.49
19.0	2.553
20.0	2.611
21.0	2.528
22.0	2.618
23.0	2.685
24.0	2.649

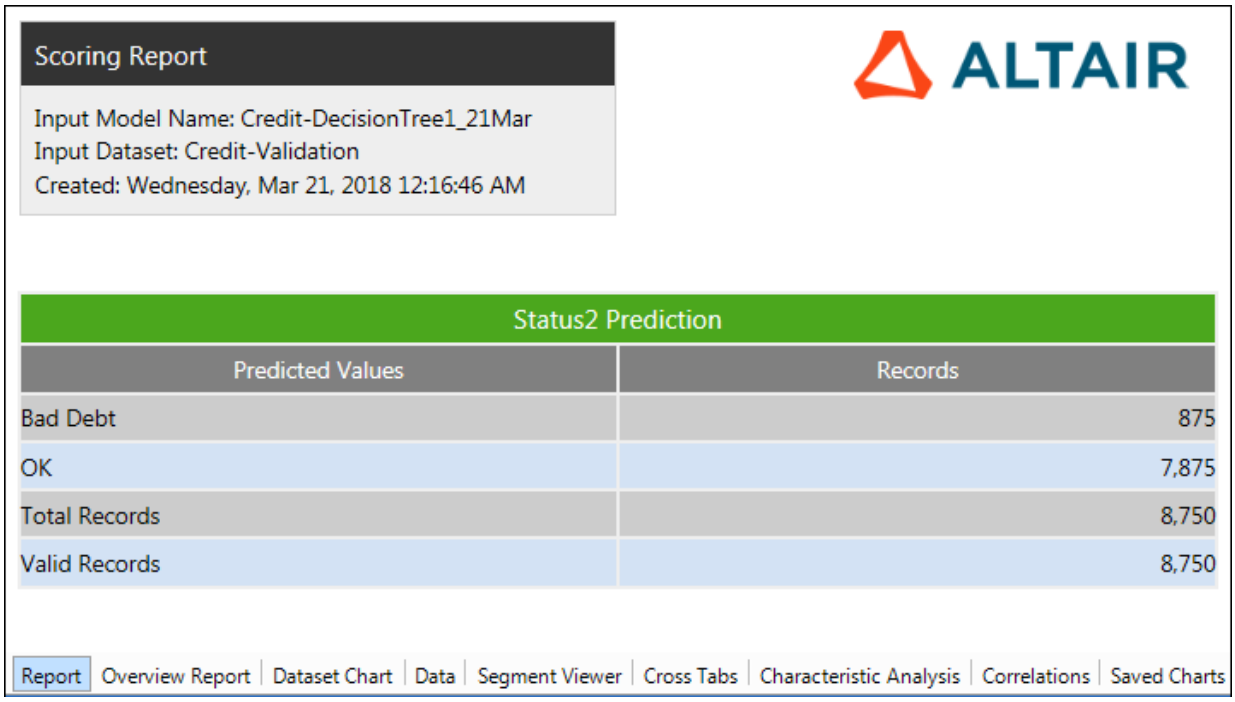
See also: ARIMA Forecasting

The top section of the Score Report displays basic information such as the name of the input dataset, the name of the input model, and the date scoring was performed.

It also shows the total number of records provided as the scoring input ("***Total Number of Scored Records***") and the number of records for which a valid score was produced ("***Valid Records***"). The latter may be less than the former in the case where a required field in the scoring dataset has values that did not occur in the corresponding field of the training dataset (in this case, the produced score is a missing value).

Note: Score Report is not applicable to the models implemented in Python.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Scoring Report

Input Model Name: Credit-DecisionTree1_21Mar
Input Dataset: Credit-Validation
Created: Wednesday, Mar 21, 2018 12:16:46 AM

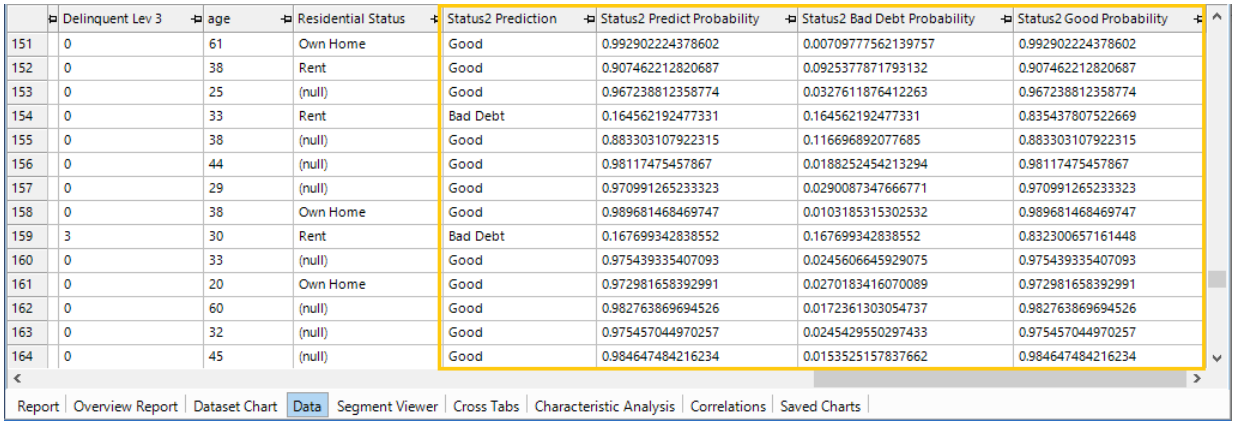
Status2 Prediction

Predicted Values	Records
Bad Debt	875
OK	7,875
Total Records	8,750
Valid Records	8,750

Report | Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

Click the **Data** tab of the output dataset to see the probabilities of the dependent variable(s) and model predictions.

Example A: Score fields for a native model (Random Forest):

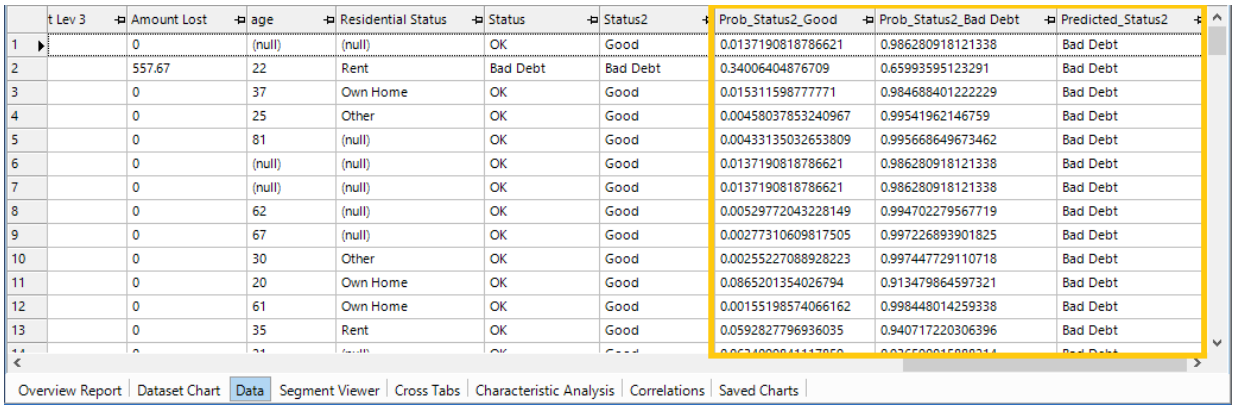


	Delinquent Lev 3	age	Residential Status	Status2 Prediction	Status2 Predict Probability	Status2 Bad Debt Probability	Status2 Good Probability
151	0	61	Own Home	Good	0.992902224378602	0.00709777562139757	0.992902224378602
152	0	38	Rent	Good	0.907462212820687	0.0925377871793132	0.907462212820687
153	0	25	(null)	Good	0.967238812358774	0.0327611876412263	0.967238812358774
154	0	33	Rent	Bad Debt	0.164562192477331	0.164562192477331	0.835437807522669
155	0	38	(null)	Good	0.883303107922315	0.116696892077685	0.883303107922315
156	0	44	(null)	Good	0.98117475457867	0.0188252454213294	0.98117475457867
157	0	29	(null)	Good	0.970991265233323	0.0290087347666771	0.970991265233323
158	0	38	Own Home	Good	0.989681468469747	0.0103185315302532	0.989681468469747
159	3	30	Rent	Bad Debt	0.167699342838552	0.167699342838552	0.832300657161448
160	0	33	(null)	Good	0.975439335407093	0.0245606645929075	0.975439335407093
161	0	20	Own Home	Good	0.972981658392991	0.0270183416070089	0.972981658392991
162	0	60	(null)	Good	0.982763869694526	0.0172361303054737	0.982763869694526
163	0	32	(null)	Good	0.975457044970257	0.0245429550297433	0.975457044970257
164	0	45	(null)	Good	0.984647484216234	0.0153525157837662	0.984647484216234

Report | Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

Example B: Score fields for a model implemented in Python (XGBoost):

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



	t Lev 3	Amount Lost	age	Residential Status	Status	Status2	Prob_Status2_Good	Prob_Status2_Bad Debt	Predicted_Status2
1		0	(null)	(null)	OK	Good	0.0137190818786621	0.986280918121338	Bad Debt
2		557.67	22	Rent	Bad Debt	Bad Debt	0.34006404876709	0.65993595123291	Bad Debt
3		0	37	Own Home	OK	Good	0.015311598777771	0.984688401222229	Bad Debt
4		0	25	Other	OK	Good	0.00458037853240967	0.99541962146759	Bad Debt
5		0	81	(null)	OK	Good	0.00433135032653809	0.995668649673462	Bad Debt
6		0	(null)	(null)	OK	Good	0.0137190818786621	0.986280918121338	Bad Debt
7		0	(null)	(null)	OK	Good	0.0137190818786621	0.986280918121338	Bad Debt
8		0	62	(null)	OK	Good	0.00529772043228149	0.994702279567719	Bad Debt
9		0	67	(null)	OK	Good	0.00277310609817505	0.997226893901825	Bad Debt
10		0	30	Other	OK	Good	0.00255227088928223	0.997447729110718	Bad Debt
11		0	20	Own Home	OK	Good	0.0865201354026794	0.913479864597321	Bad Debt
12		0	61	Own Home	OK	Good	0.00155198574066162	0.998448014259338	Bad Debt
13		0	35	Rent	OK	Good	0.0592827796936035	0.940717220306396	Bad Debt

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

Deploying Association Rules

Deploying association rules of a Market Basket Analysis model consists in applying the rules to the new data to generate recommendations.

One of the typical use cases is where a marketing analyst needs to generate product recommendations for customers based on the transactional data from a retail store for a certain time period.

Examples of tasks that can be solved by deploying association rules are as follows:

- For each customer, find top 5 recommended products and rank them
- Generate product recommendations based on rules with the highest lift

An example of the output is a list of customers and recommended products in the following form:

Customer ID	Recommendations	Rank
1022	PEPSI 16OZ 6PK	1
1022	COLOMBO YOGHURT PEACH	2
3067	KRAFT CHEDDAR SINGLES 12OZ	1
3067	BREYERS CHERRY ICE CREAM	2
5499	BYRNE 2% MILK 1GAL	1
5570	DOLE SALAD MIX 1#	1

Formally, in terms of association rules, the task of producing product recommendations can be formulated as follows: For each customer, find all association rules whose left-hand sides are

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

subsets of at least one basket purchased by this customer. Among these rules, select top 5 most important ones and recommend their right-hand side items to the customer.

Example: If there is an association rule: {a,b,c} => z and a basket {a,b,c,d,e,f}, then z would be a recommendation for this basket. Recommendations can be grouped by Customer ID or another field in the target dataset.

To deploy a Market Basket Analysis model directly in Knowledge Studio, see the topic Scoring a Market Basket Analysis Model

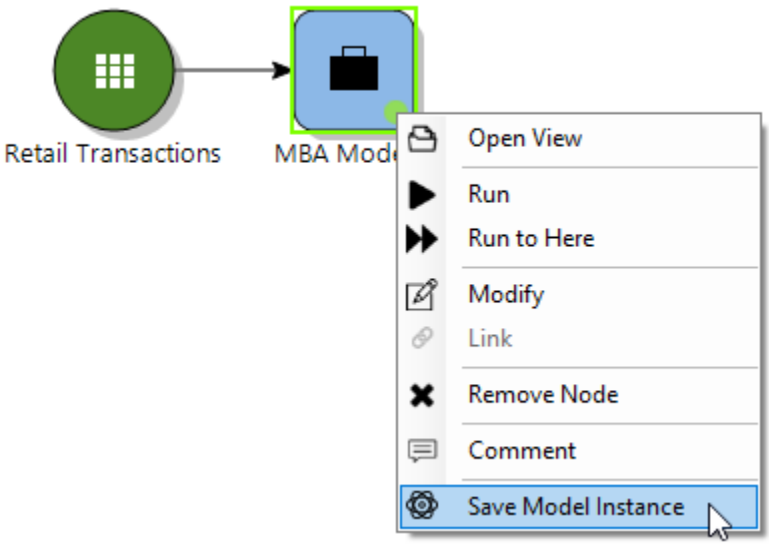
Alternatively, you can generate SQL code for a Market Basket Analysis model and run it in a database environment outside of Knowledge Studio. See Code Generation for Market Basket Analysis for details.

Deploying, or scoring, a Market Basket Analysis model consists in applying selected association rules to the new transactional data to produce *recommendations*.

The right-hand side of a rule represents a recommendation for a group of items if the rule's left-hand side is contained in this group. For example, if the purpose is to produce product recommendations for customer N, the first step is to determine the set of items he/she purchased. If there is a rule A=>B such that the purchases of customer N include the set of items A, then B will be a recommendation for this customer.

As follows from the Model Instances topic, for Market Basket Analysis models and scoring are applicable ONLY to model instance nodes, and not applicable to model process nodes. To score a Market Basket Analysis model, you should first **save a model instance** for it, since you always need to manually select the association rules to be deployed.

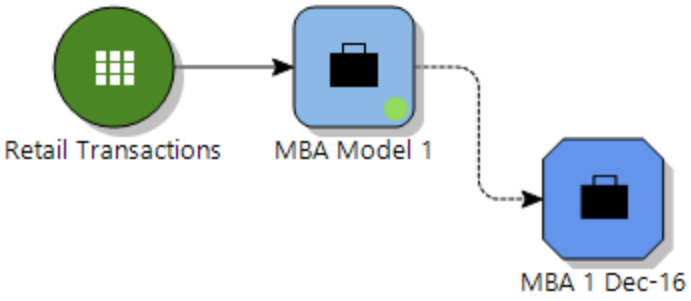
To save a model instance, right-click on the Market Basket Analysis process node in your workflow and select the **Save Model Instance** command.



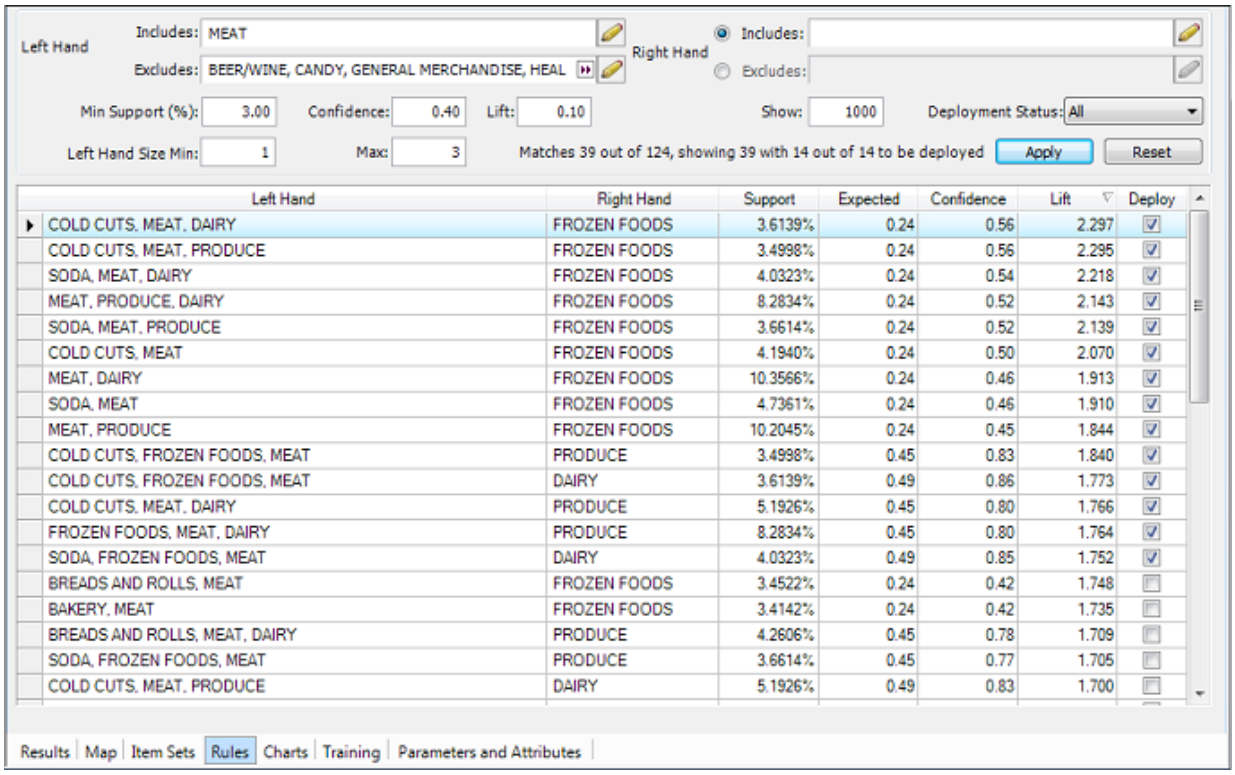
Specify the name for the new model instance and click OK. This will save a copy of the model instance resulting from the most recent run of the model process node. The new model instance is

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

connected to the process node by a dotted line indicating that this model instance cannot be overwritten by re-running the workflow.



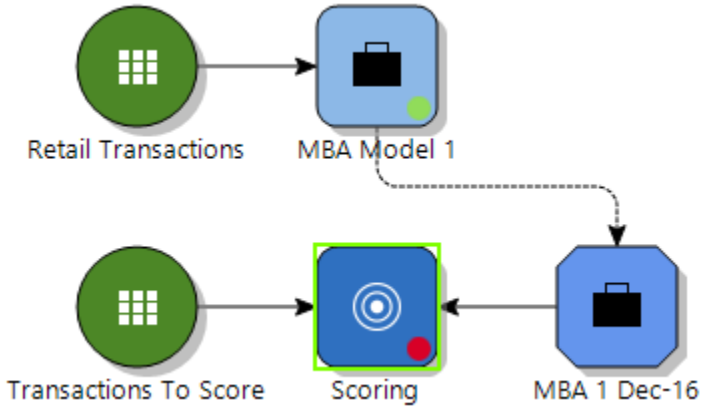
Double-click the model instance node to open it. Open the **Rules** tab of the model and select the association rules you want to deploy to generate recommendations. Only the rules marked for deployment in the Rules tab of Market Basket Analysis will be the inputs of the Scoring operation. Make sure you select al least one rule for deployment before starting the scoring wizard. In the example below, 14 top rules in the Rules tab are selected for deployment. Only these 14 rules will be scored.



Left Hand	Right Hand	Support	Expected	Confidence	Lift	Deploy
COLD CUTS, MEAT, DAIRY	FROZEN FOODS	3.6139%	0.24	0.56	2.297	☑
COLD CUTS, MEAT, PRODUCE	FROZEN FOODS	3.4998%	0.24	0.56	2.295	☑
SODA, MEAT, DAIRY	FROZEN FOODS	4.0323%	0.24	0.54	2.218	☑
MEAT, PRODUCE, DAIRY	FROZEN FOODS	8.2834%	0.24	0.52	2.143	☑
SODA, MEAT, PRODUCE	FROZEN FOODS	3.6614%	0.24	0.52	2.139	☑
COLD CUTS, MEAT	FROZEN FOODS	4.1940%	0.24	0.50	2.070	☑
MEAT, DAIRY	FROZEN FOODS	10.3566%	0.24	0.46	1.913	☑
SODA, MEAT	FROZEN FOODS	4.7361%	0.24	0.46	1.910	☑
MEAT, PRODUCE	FROZEN FOODS	10.2045%	0.24	0.45	1.844	☑
COLD CUTS, FROZEN FOODS, MEAT	PRODUCE	3.4998%	0.45	0.83	1.840	☑
COLD CUTS, FROZEN FOODS, MEAT	DAIRY	3.6139%	0.49	0.86	1.773	☑
COLD CUTS, MEAT, DAIRY	PRODUCE	5.1926%	0.45	0.80	1.766	☑
FROZEN FOODS, MEAT, DAIRY	PRODUCE	8.2834%	0.45	0.80	1.764	☑
SODA, FROZEN FOODS, MEAT	DAIRY	4.0323%	0.49	0.85	1.752	☑
BREADS AND ROLLS, MEAT	FROZEN FOODS	3.4522%	0.24	0.42	1.748	☐
BAKERY, MEAT	FROZEN FOODS	3.4142%	0.24	0.42	1.735	☐
BREADS AND ROLLS, MEAT, DAIRY	PRODUCE	4.2606%	0.45	0.78	1.709	☐
SODA, FROZEN FOODS, MEAT	PRODUCE	3.6614%	0.45	0.77	1.705	☐
COLD CUTS, MEAT, PRODUCE	DAIRY	5.1926%	0.49	0.83	1.700	☐

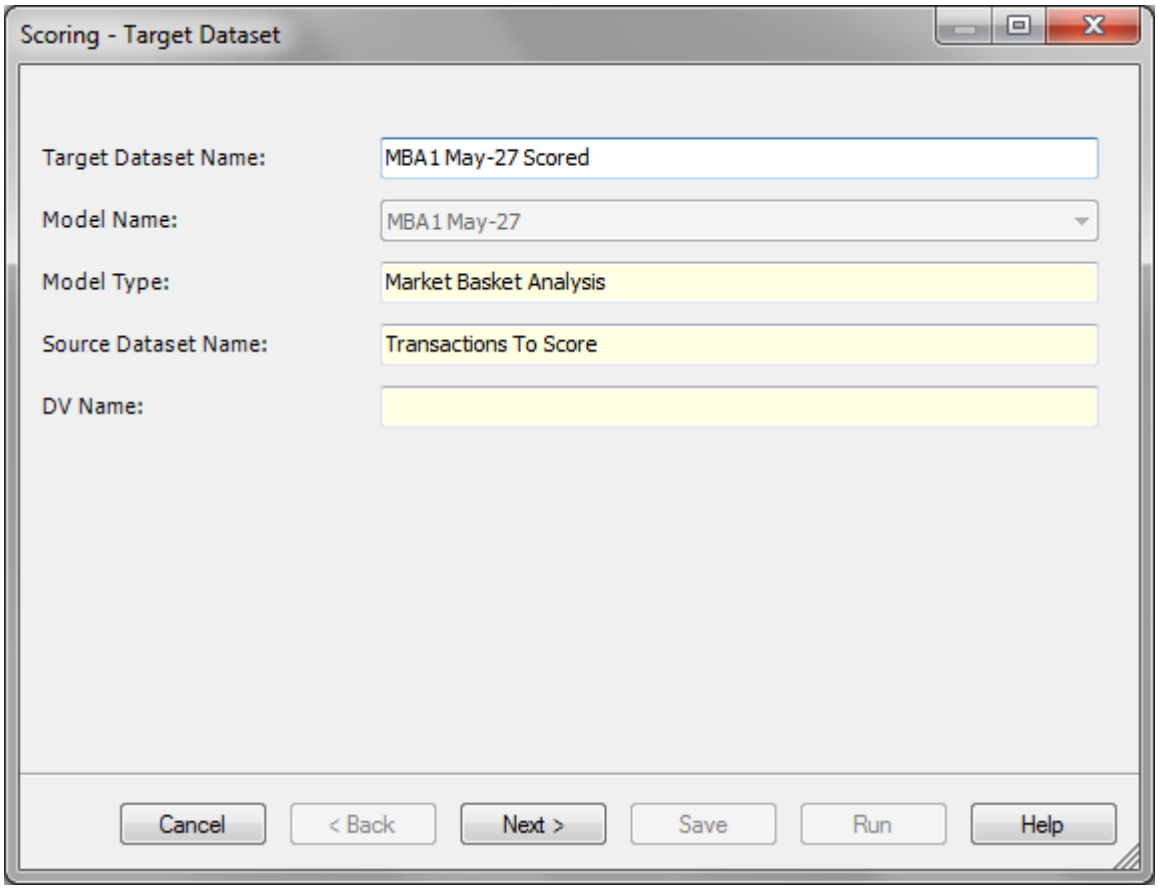
Drag the **Scoring** node from the **Action** palette to the workflow canvas. Connect the Market Basket Analysis model instance to the Scoring node. Connect an input dataset to score to the scoring node, as shown in the example screenshot below.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Double-click the Scoring node to open the wizard.

In the first page of the wizard, enter the name for the scoring output dataset in the Target Dataset Name field:

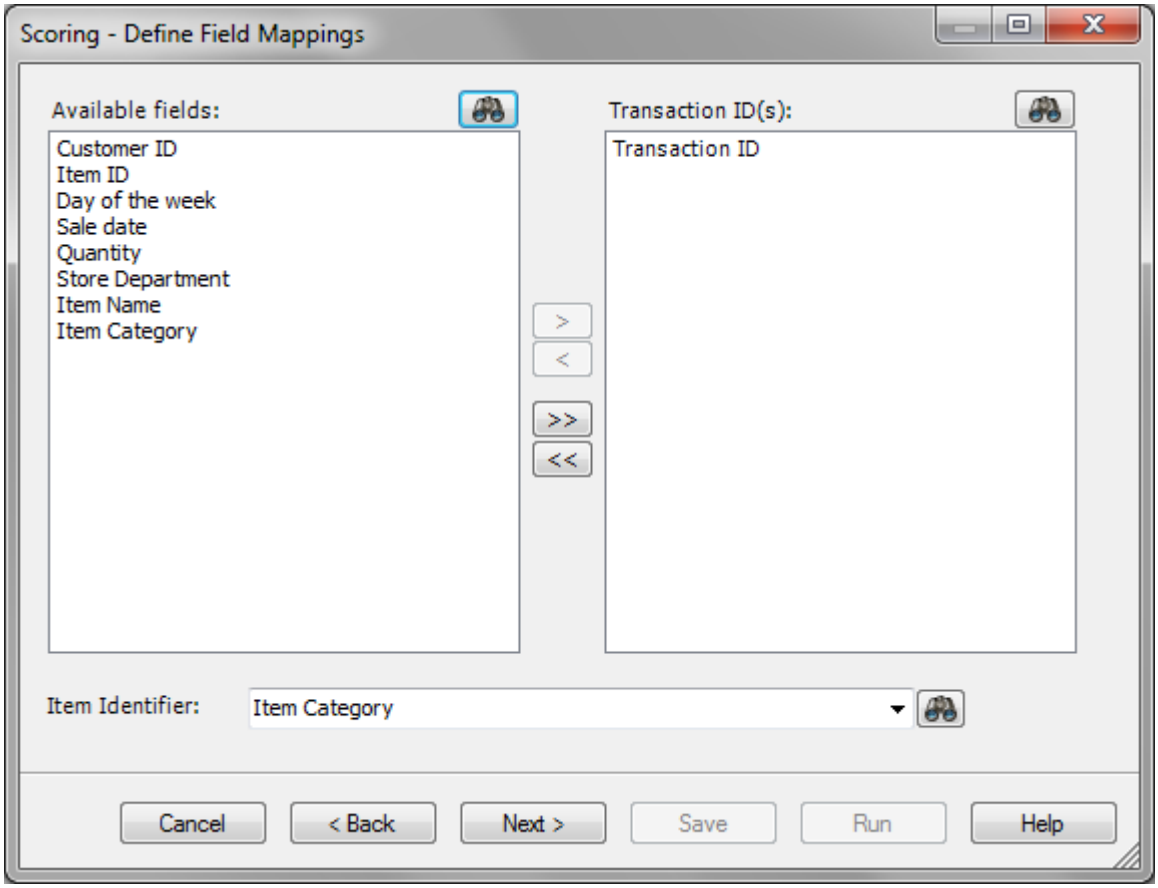


The input dataset must contain at least two fields:

- the identifier of the target of recommendation
- the item identifier

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

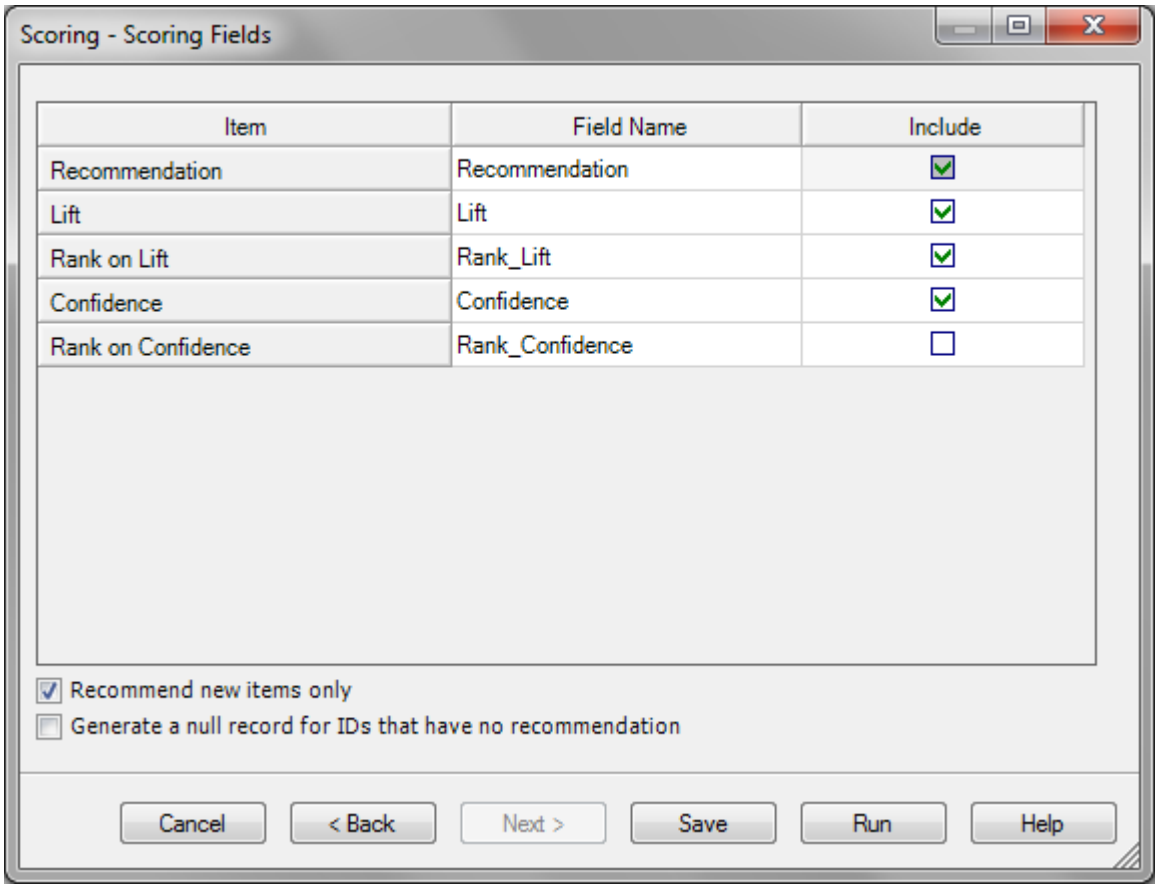
Select these fields in the next page of the wizard.



Transaction ID(s)	The identifier of the target of recommendation (one or more columns). For example, if product recommendations are to be produced at the customer level, it can be Customer ID. If they are to be produced for individual transactions, it can be Transaction ID. In general, the target of recommendations can be chosen to be any column(s) by which to group the items when producing recommendations. For example, you can choose two columns: Customer ID and Transaction Month, so personalized recommendations will also depend on the month of purchase.
Item Identifier	The item identifier field. This is assumed to be the same identifier type as the one used for items in the association rules. For example, if your association rules deal with product categories rather than product names, then choose the product category column as *Item identifier* in the input dataset.

At the next page, select the new fields to be created as part of the scoring output and specify the scoring options.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



New fields to create:

Recommendation	The Recommendation field is the only mandatory field, so it is selected by default and cannot be unselected.
Lift	The lift of the association rule that was applied to produce the recommendation. This is an optional field selected by default.
Rank on Lift	The recommendation rank based on Lift. In the case of multiple recommendations generated for the same ID, recommendations with higher lift have higher rank, starting from Rank=1 for the highest lift. This is an optional field selected by default.
Confidence	The confidence of the association rule that was applied to produce the recommendation. This is an optional field selected by default.
Rank on Confidence	In the case of multiple recommendations generated for the same Transaction ID, this is the recommendation rank based on confidence (recommendations with higher confidence have higher rank). This field is not selected by default.

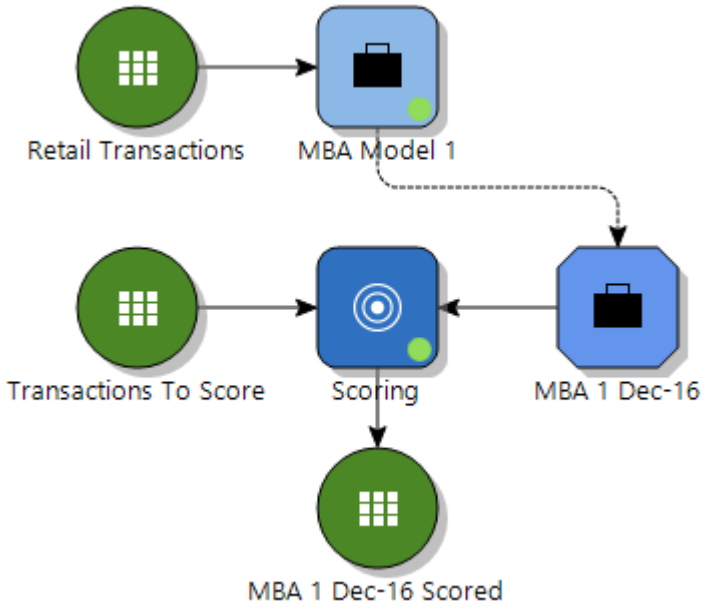
Options:

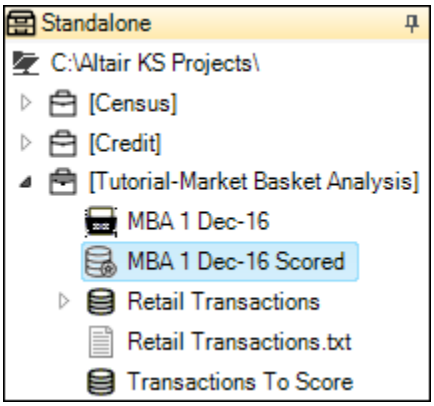
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Recommend new items only	This option is also called "Restrict to cross-sell", i.e., do not recommend any item that already appeared in the target basket. This option is selected by default.
Generate a null record for IDs that have no recommendation	This option determines whether to drop or retain records with no recommendation. Select this option if you want the output to include records with no recommendation. In this case, such records will have a null value in the Recommendation column. By default, this option is not selected, which means that records with no recommendation are omitted from the output.

Click **Save** to save the settings, or click **Run** to save the settings and run the node immediately.

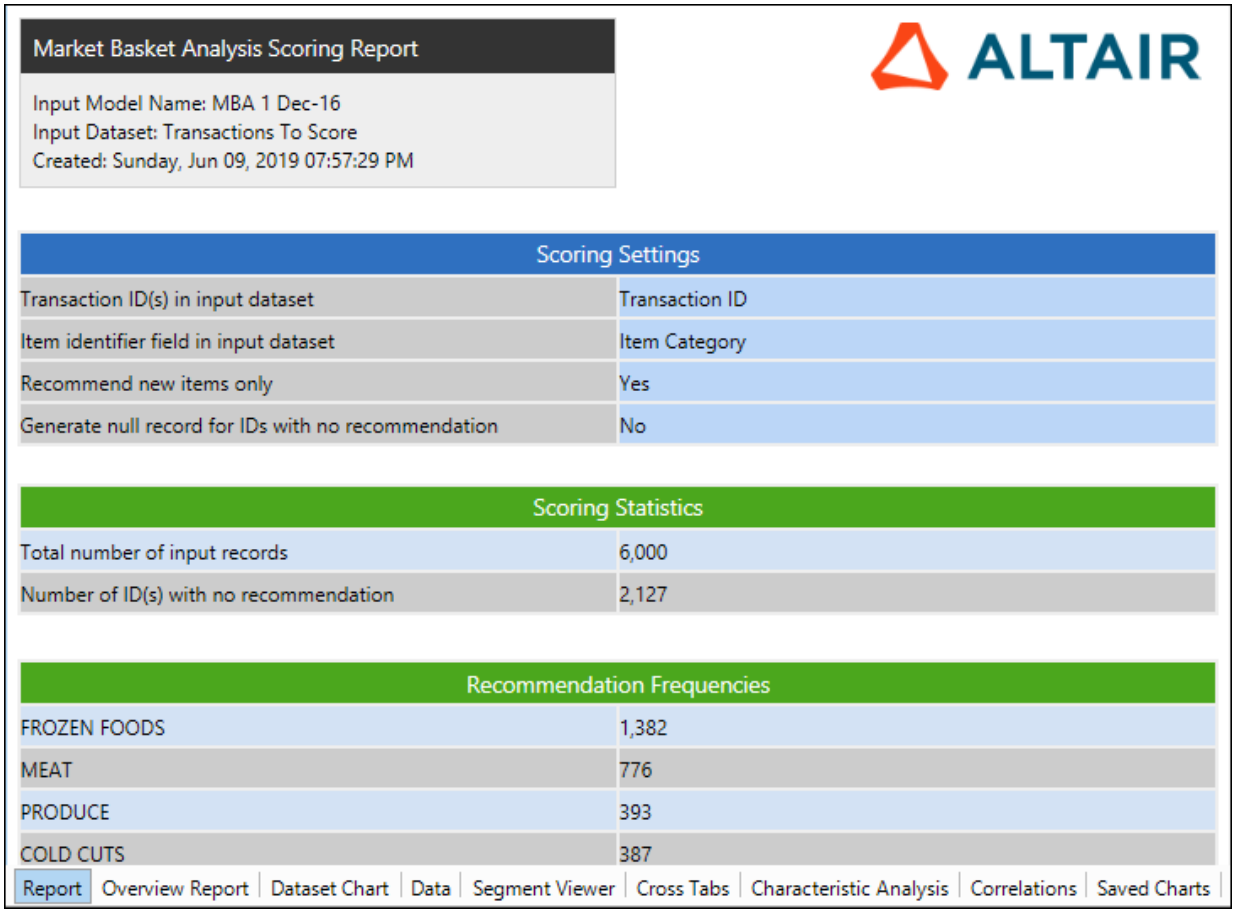
When the scoring process is finished, the output dataset node that contains scored records with recommendations is shown in the workflow, and the underlying dataset is dispplayed in the Project Explorer panel.





Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The *Results* view of this object contains a summary of the scoring settings and basic scoring statistics, including recommendation frequencies. The report shows up to 100 most frequent recommendations ordered by frequency.



Market Basket Analysis Scoring Report

Input Model Name: MBA 1 Dec-16
Input Dataset: Transactions To Score
Created: Sunday, Jun 09, 2019 07:57:29 PM

Scoring Settings	
Transaction ID(s) in input dataset	Transaction ID
Item identifier field in input dataset	Item Category
Recommend new items only	Yes
Generate null record for IDs with no recommendation	No

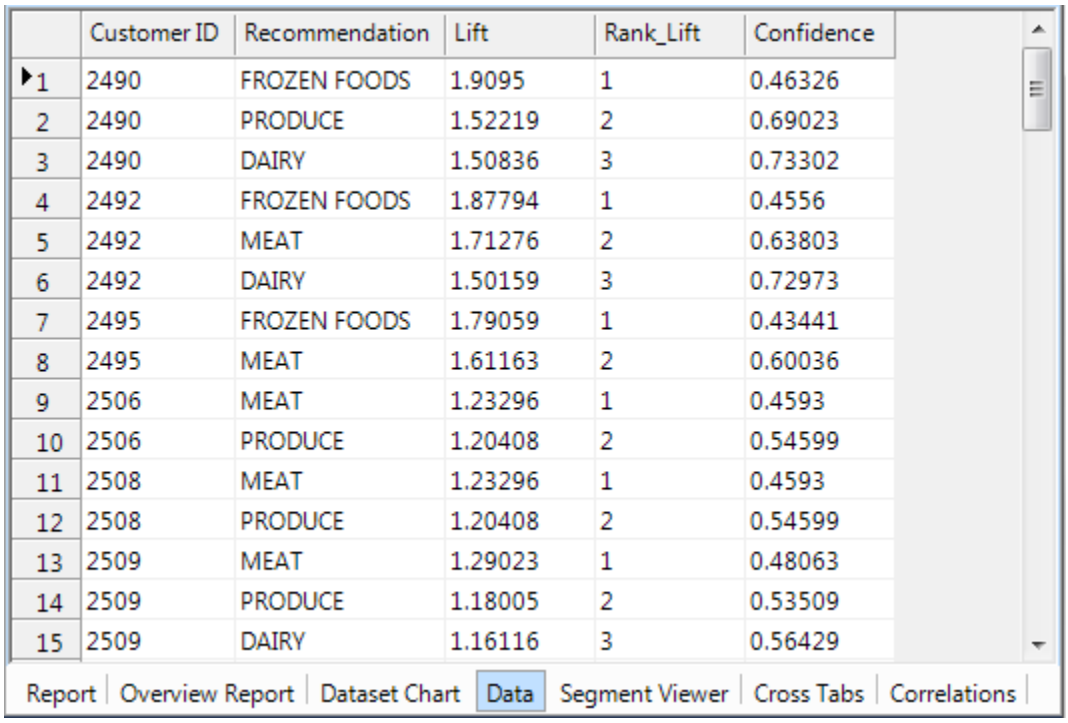
Scoring Statistics	
Total number of input records	6,000
Number of ID(s) with no recommendation	2,127

Recommendation Frequencies	
FROZEN FOODS	1,382
MEAT	776
PRODUCE	393
COLD CUTS	387

Report | Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

If the New Dataset option was selected in the wizard, the output includes a dataset with generated recommendations and the related metrics. It has all the standard views of a dataset object, such as Overview Report and Segment Viewer, and can be manipulated as any other dataset. Click the Data tab to view the results. The metrics displayed here depend on your choice in the New Fields page of the scoring wizard (see above). As shown in the example screenshot below, if there are multiple recommendations generated for a single ID, they are ordered by Lift, and the Rank_Lift column shows their rank with respect to Lift.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



	Customer ID	Recommendation	Lift	Rank_Lift	Confidence
1	2490	FROZEN FOODS	1.9095	1	0.46326
2	2490	PRODUCE	1.52219	2	0.69023
3	2490	DAIRY	1.50836	3	0.73302
4	2492	FROZEN FOODS	1.87794	1	0.4556
5	2492	MEAT	1.71276	2	0.63803
6	2492	DAIRY	1.50159	3	0.72973
7	2495	FROZEN FOODS	1.79059	1	0.43441
8	2495	MEAT	1.61163	2	0.60036
9	2506	MEAT	1.23296	1	0.4593
10	2506	PRODUCE	1.20408	2	0.54599
11	2508	MEAT	1.23296	1	0.4593
12	2508	PRODUCE	1.20408	2	0.54599
13	2509	MEAT	1.29023	1	0.48063
14	2509	PRODUCE	1.18005	2	0.53509
15	2509	DAIRY	1.16116	3	0.56429

Report | Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Correlations

Another way to deploy a Market Basket Analysis model is to generate SQL code and run it in a database environment outside of the Altair application. See Code Generation for Market Basket Analysis for details.

Two types of code can be generated for Market Basket Analysis: English language rules and SQL code.

English language rules are simply a natural language expression of association rules. They are generated for illustrative purposes rather than deployment.

For example, for the association rule {*COLD CUTS, MEAT, DAIRY*} => *FROZEN FOODS* with Support=3.61%, Confidence=0.5572, Lift=2.3 and Expected frequency=24.26%, the English language rule is

> *Items "COLD CUTS" and "MEAT" and "DAIRY" are associated with item "FROZEN FOODS" with a confidence of 55.72%, which is 2.30 times more likely than the expected frequency of 24.26%. This combination occurred in 3.61% of all transactions (380)*.

The SQL code is intended to produce recommendations and can be deployed in a database system. For an introduction to deploying a Market Basket Analysis model, see the topic Deploying Association Rules: Overview

When run in a database environment, the SQL code implements the same actions as the scoring command for Market Basket Analysis, i.e., it generates recommendations based on association rules and creates a new table with the Recommendation, Lift, and Confidence columns. The SQL code syntax follows Microsoft SQL Server standards, but can be easily modified to be run in other database systems.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

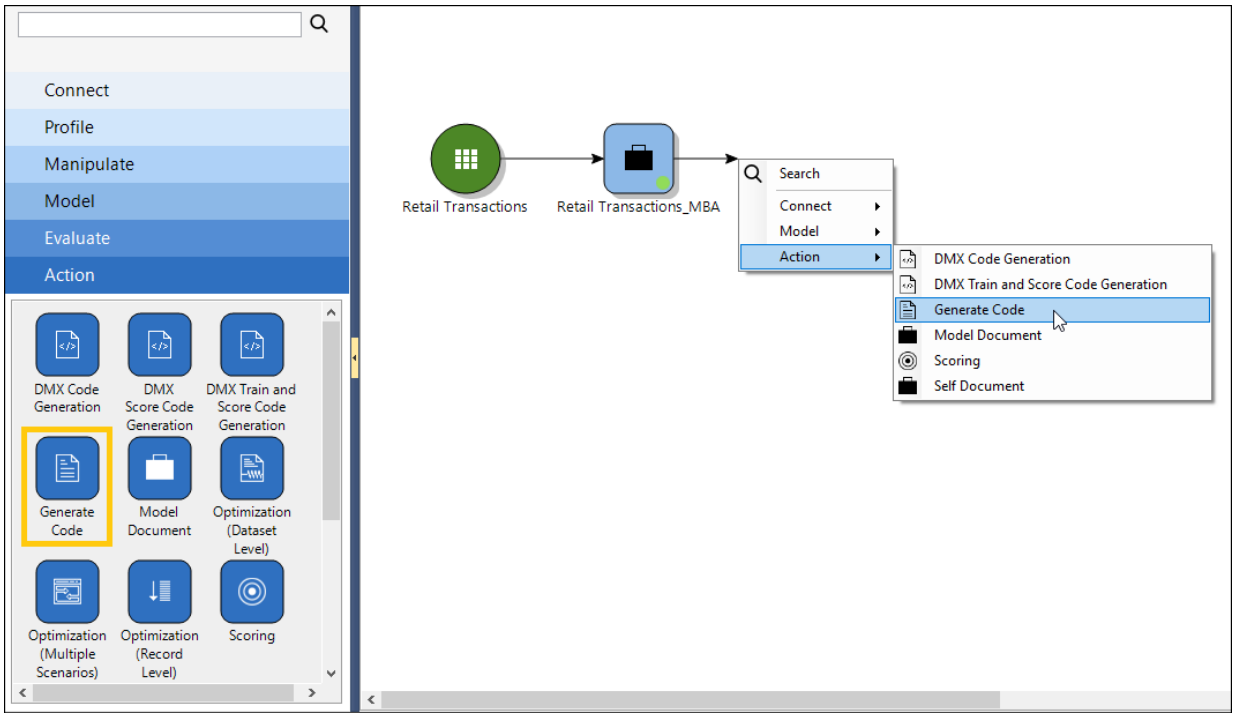
Code is generated only for the association rules selected for deployment in the **Rules** tab of Market Basket Analysis. Make sure at least one rule is selected in the Deploy column of the Rules tab before generating code. See the Rules topic for details.

Once you selected the desired rules for deployment, add the **Generate Code** node to your workflow as follows:

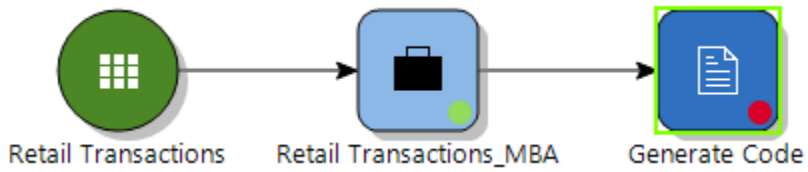
- Drag a connection arrow from your Market Basket Analysis model node to any area on the Workflow canvas background. From the context menu, select **Action --> Generate Code.** A new **Generate Code** node will be created, and the wizard will open.

OR:

- Drag the **Generate Code** node from the **Action** palette onto the workflow canvas and connect the Market Basket Analysis model to it.



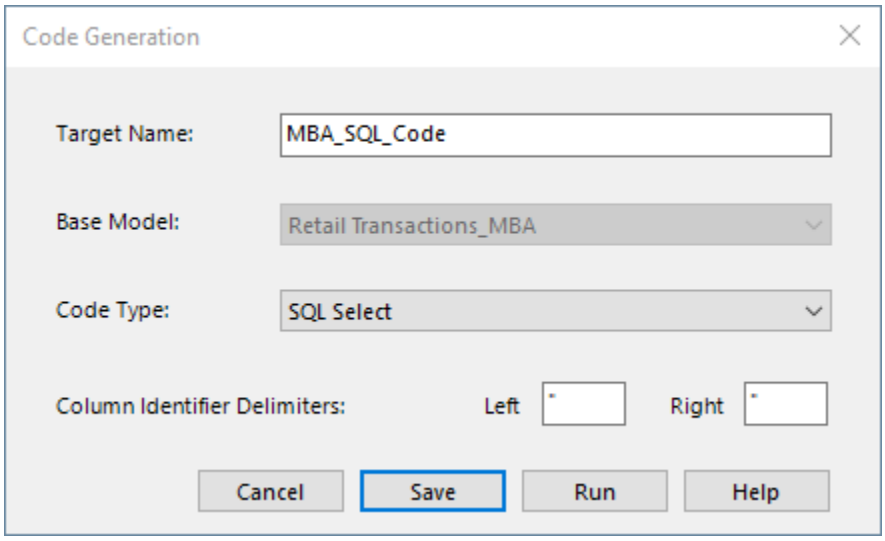
If the Generate code node is already on the canvas, double-click on it or right-click it and select **Modify**.



Choose the desired code type in the code Generation dialog. The available code types are *English*, *SAS Language*, and *SQL Select*.

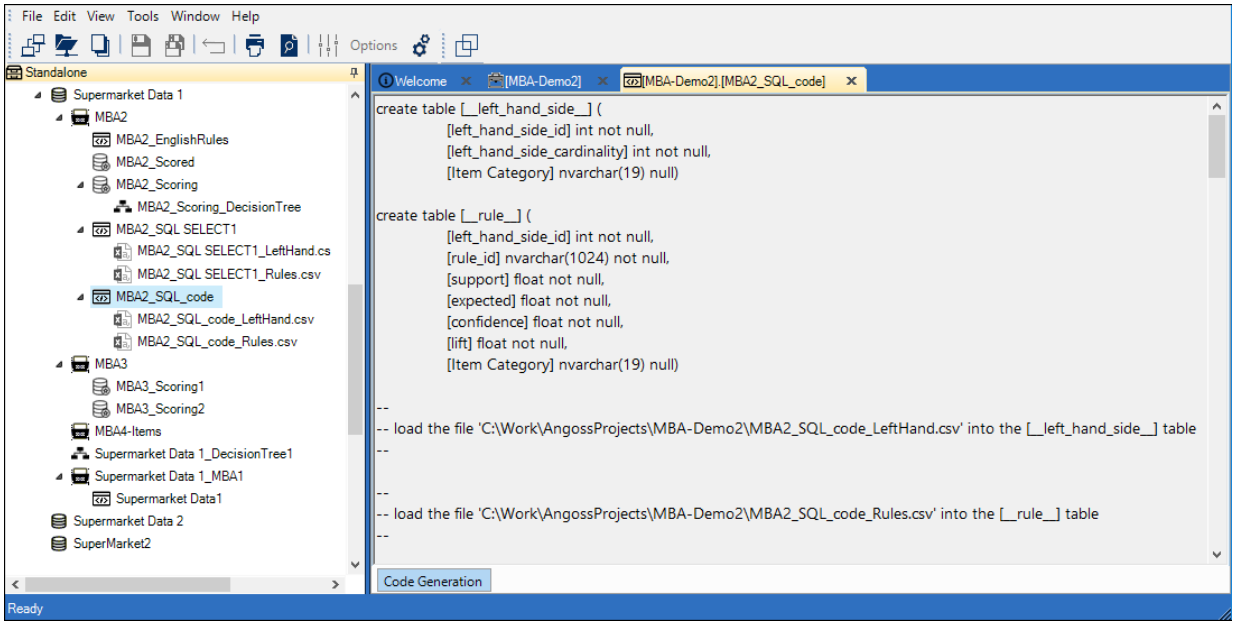
If you choose the *SQL Select* code type, specify the column identifier delimiters if necessary.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the node immediately.

In the case of SQL code: The code will be generated along with two comma-separated text files that represent the association rules selected for deployment. Together with the scoring code, the content of these files is necessary to deploy the rules in a database. The files are called *<Model_name>_SQL_code_LeftHand.csv* and *<Model_name>_SQL_code_Rules.csv.* They are shown in the Project Outline as child objects of the SQL code under the parent Market Basket Analysis model.



The first part of the code contains the commands to load the content of the generated csv files into database tables. The second part contains the scoring code that uses these tables to produce recommendations.

Detailed comments in the scoring code provide useful guidance and explanation of the process.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

In the first part, the CREATE TABLE statements create the schema for two tables that together represent the association rules to be deployed. The task of populating these tables by loading the generated text files is left to the user.

To this end, right-click the project name in the Project Outline and select *Open In Explorer* from the context menu. This will open the project folder in Windows Explorer. Copy the files *<Model_name>_SQL_code_LeftHand.csv* and *<Model_name>_SQL_code_Rules.csv* from the project folder to a preferred location and load their content into the database tables.

The second part contains an example scoring code that converts the first table to a set of input transactions and produces recommendations for them. It serves as an example code snippet that the user can modify as necessary to target other transaction tables.

As in the direct scoring using the *Tools* | *Score* command, the code searches for all rules whose left-hand side matches the set of items in a target transaction and produces their right-hand sides as recommendations. For each input ID, the SQL code only produces the top-ranked recommendation - the one with the highest lift.

Note that the scoring options "Recommend new items only" and "Generate a null record for IDs that have no recommendation" are not used in the SQL code generation. You can add the necessary code for these options if necessary.

See also: Scoring a Market Basket Analysis Model

Code Generation

Using ***automatic code generation***, native Altair models can be exported to other formats for deploying outside of the Altair Data Analytics applications and for presentation purposes. The model deployment code in SQL, SAS Language, R, C, Python, SPSS, Java, or PMML can be run in the analytic environments, databases or decision engines that can interpret these languages.

Code generation is performed using the **Generate Code** node in the **Action** palette.

> Note: The following nodes are implemented in R or Python, and their source code is readily available in their respective node wizards. The Generate Code node is therefore not applicable to them.
>
> - *Survival Analysis* is implemented in the language of R. The code for the Survival Analysis node can be viewed in the Code Preview panel of its wizard. R statistical software is required to run the Survival Analysis node.
> - *Keras Deep Learning, Keras Scoring*, *Keras Forecasting*, *ARIMA Forecasting, ARIMA Scoring*, *Auto Feature Engineering*, *Handle Class Imbalance*, *Sklearn GLM, Sklearn GLM Scoring, Sklearn Novelty & Outlier Detector, Sklearn Novelty & Outlier Scoring, Sklearn XGB*, and *Sklearn XGB Scoring* are implemented in the language of Python. The Python code for these nodes can be viewed in the Code Preview panel of their respective wizards.

The code generation types available for each model type are listed in the following table:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

	Decision Trees Strategy Trees	Ensemble Trees (Random Forest, Bagging, Boosting)	Regression, Regularization, Deep Learning (Neural Network) Cluster Analysis	Scorecards	Market Basket Analysis	Principal Component Analysis Factor Analysis
English	x				x	
Generic pseudocode	x					
Java	x					
SAS Language	x	x	x	x	x	x
PMML	x		x	x		
Python	x	x	x	x	x	x
R	x	x	x	x	x	x
C	x	x	x	x		x
SPSS	x					
SQL (CASE)	x			x	x	
SQL Function	x		x			
SQL (SELECT)	x					
XML	x	x	x	x		

Note:

- **English code** (tree representation in natural language) and **Generic pseudocode** are provided for illustration and presentation purposes. The other types of generated code can be used for model deployment in systems that can interpret these languages.
- Generated **XML code** can only be consumed by Knowledge Seeker and Knowledge Studio applications. It is provided for exporting models from one project to another, shaing models with other users, or saving models as files for backup or storage.
- Code generation is not applicable to the PLS Regression model node, but it is applicable to Linear Regression based on PLS.
- **SQL code** generated for Market Basket Analysis is conceptually different from SQL code for other models. It is covered in a separate topic Code Generation for Market Basket Analysis
- **PMML code** has limitations: Not all of the advanced attribute settings in Knowledge Seeker and Knowledge Studio models are supported in the PMML standard. For example, the PMML

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

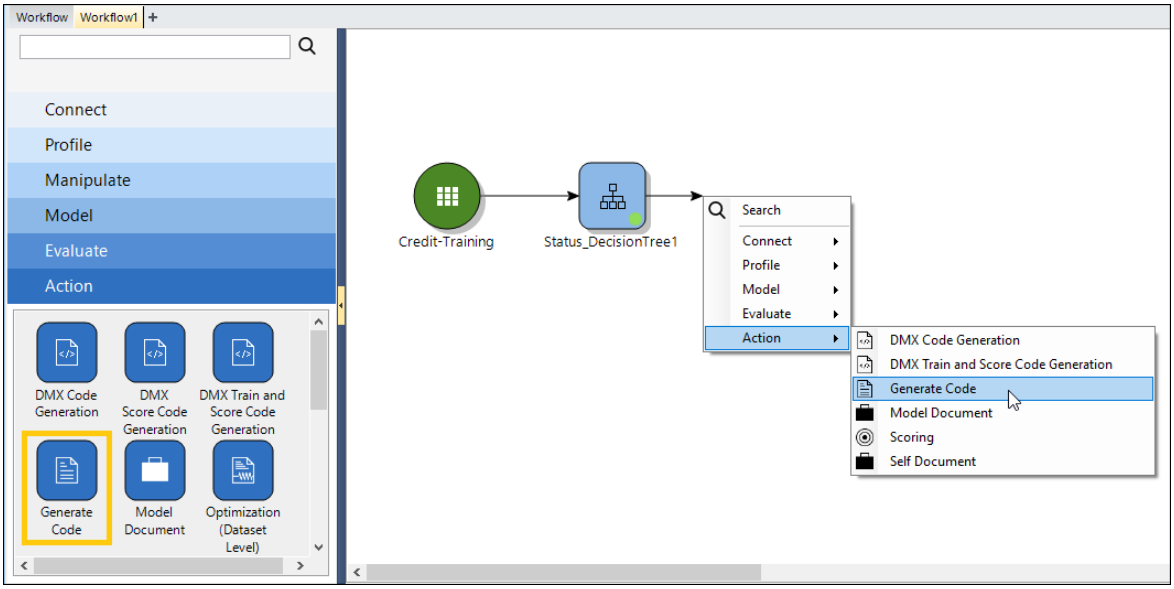
code generator ignores functions like ROUND(), which have no analog in PMML. If you generate PMML code from a Scorecard with the option "Round points to integer", this option will be ignored.

To generate code for any model node in the workflow:

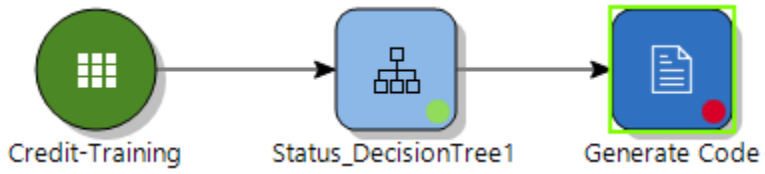
- Drag a connection arrow from the model node to any area on the Workflow canvas background. From the context menu, select **Action --> Generate Code.** A new **Generate Code** node will be created, and the wizard will open.

OR:

- Drag the **Generate Code** node from the **Action** palette onto the workflow canvas and connect the model to it.



If the **Generate Code** node is already on the canvas, double-click it or right-click on it and select **Modify**.

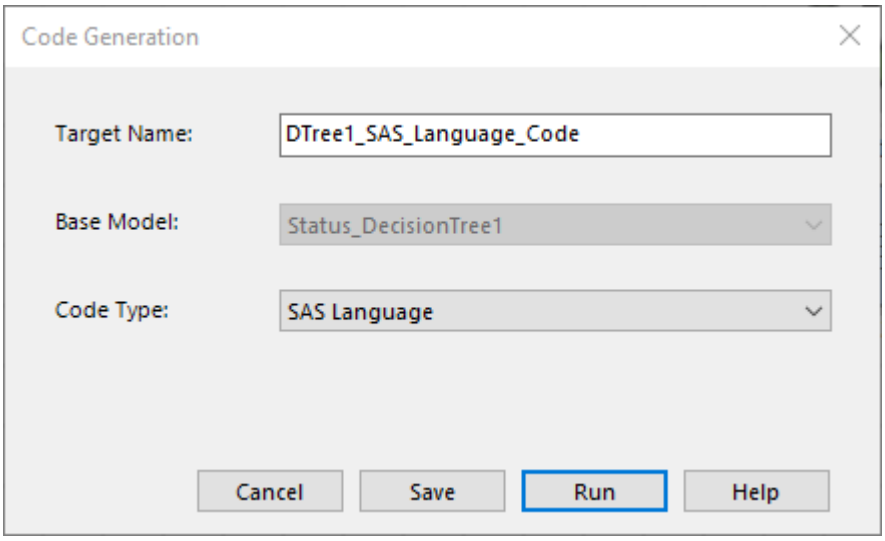


In the Code Generation dialog:

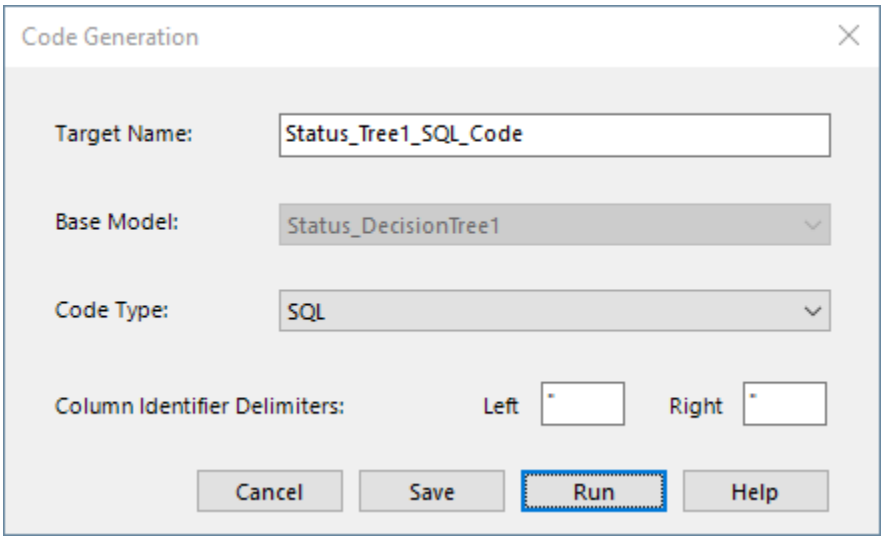
- Specify the name for the new node.
- Select the type of code to generate. Available code types depend on the input model type. See the table above.

  Note: To generate SAS Language code, select *SAS Language*.

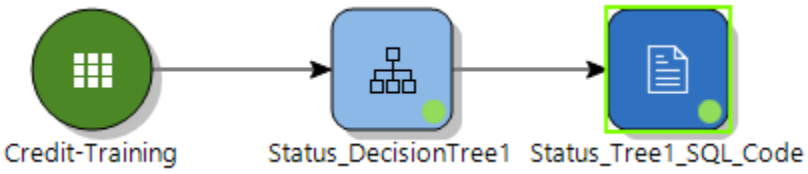
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If you select the ***SQL*** or ***SQL Select*** code type, you have an additional option of specifying column identifier delimiters. This option may be useful depending on the type of the target database for deploying the SQL code.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.



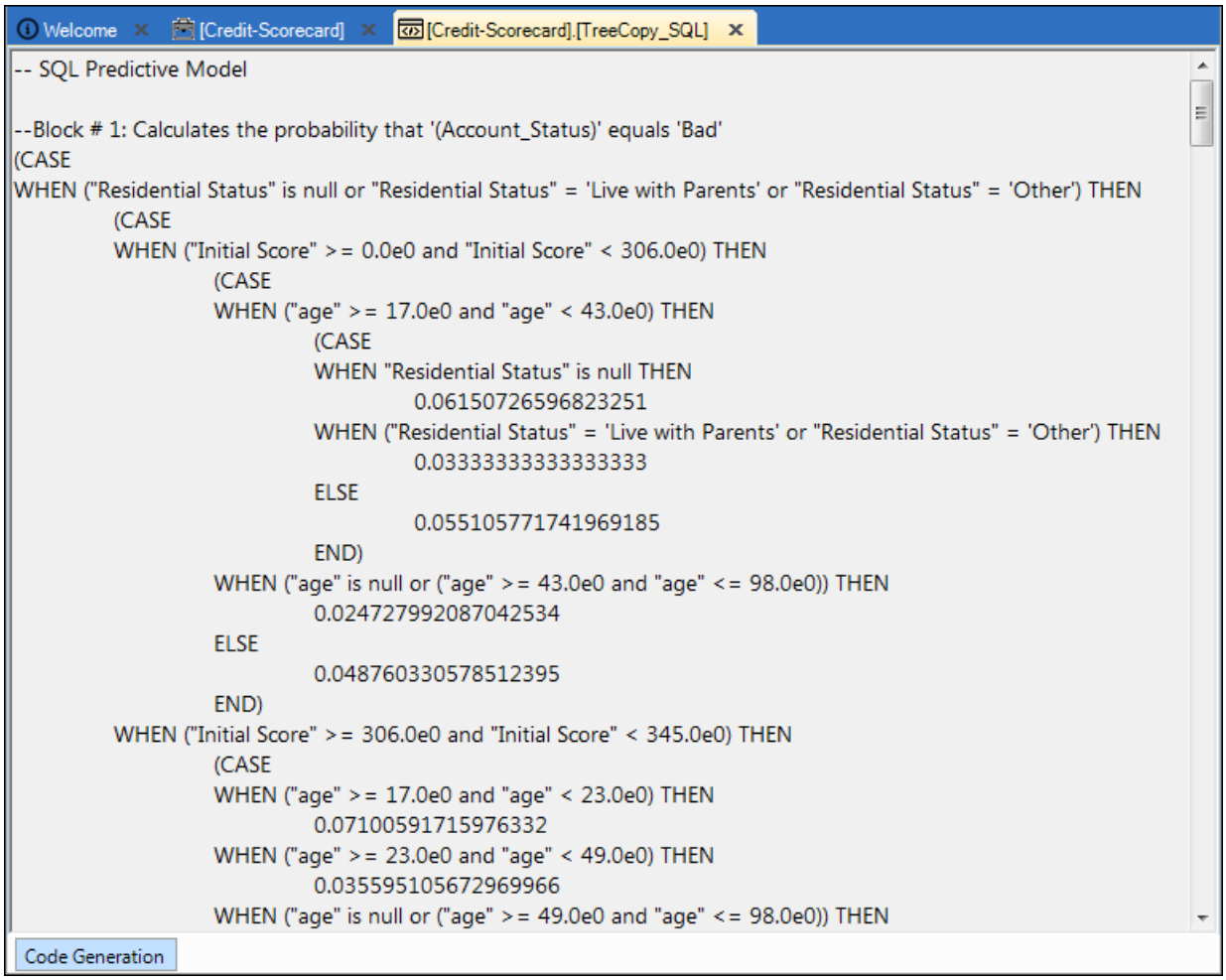
Double-click the code node to view the generated code or right-click on it and select **Open View**.

Within the frame that displays the code, you can use the context menu as well as common keyboard shortcuts CTRL+F, CTRL+A, and CTRL+C to find, select and copy text. The font size can be

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

increased using CTRL + mouse wheel zoom (scrolling the mouse wheel while holding down the CTRL key).

To save the code in a file, select the menu command **File | Save As...**

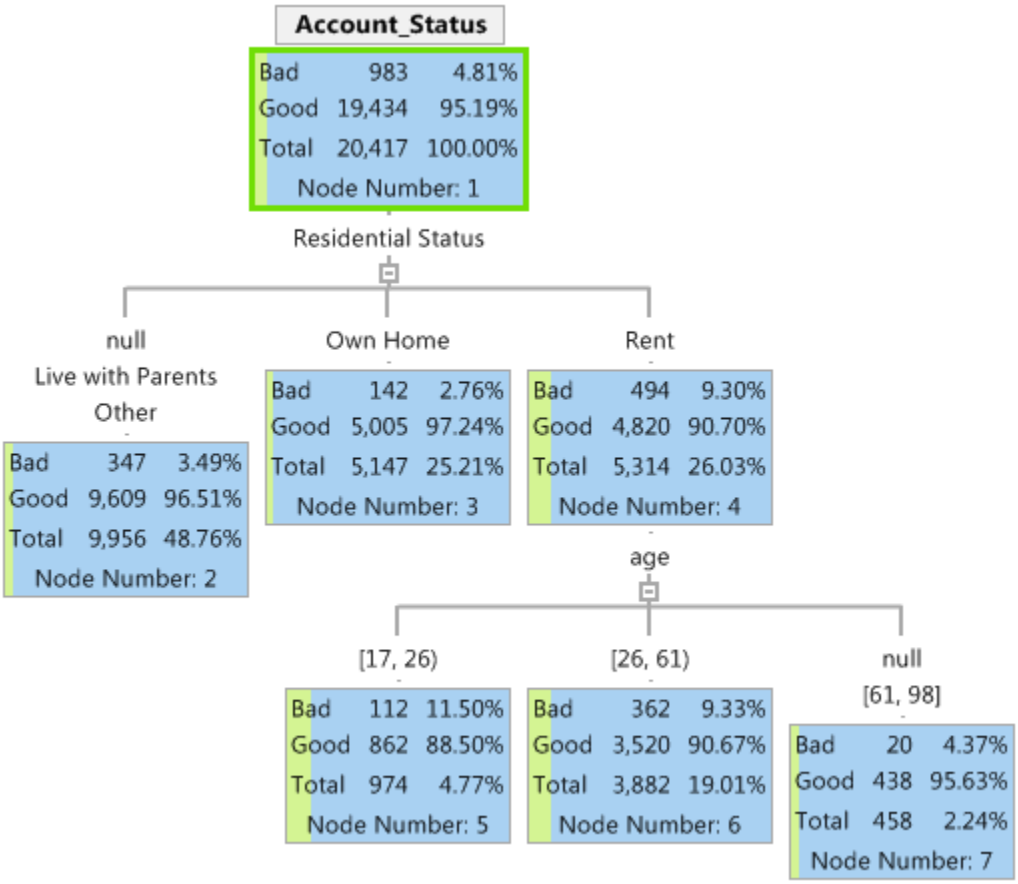


The generated deployment code in SQL, SAS Language, Python, R, SPSS, Java, or PMML can be deployed in the environments that can interpret these languages to generate model scores, cluster assignments, or treatments, depending on the model type. Just use the saved file or copy and paste the entire code or a selected part of it to the target environment. You may need to modify the code to specify the scoring input data and the destination for the scores.

1. Code Generation for Decision Trees and Strategy Trees

Brief descriptions of code types available for Decision Trees and Strategy Trees are given below. Examples are provided.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



English Natural language rules in English. Select *English* for Code Type in the Generate Code node wizard.

An example of an English rule for a specific node: "*If Residential Status is equal to Rent then there is a 9.3 percent chance that Account_Status will be Bad Debt and a 90.7 percent chance that Account_Status will be OK*."

Generic Generic pseudocode in structured English. Select *Generic* for Code Type in the Generate Code node wizard.

For each value of the dependent variable in a given node, its probability is shown on the right-hand side.

An example fragment of Generic pseudocode:

if Residential Status = Rent

Account_Status = Bad Debt 0.093

Account_Status = OK 0.907

Java ***Java*** code. Select *Java* for Code Type in the Generate Code node wizard.

You may need additional Java classes for this type of code generation. Please contact Customer Support if you have any problems with running the code.

An example fragment of Java code:

```
if(Residential Status.matches("Rent")) {
    r.set("Status2.Bad Debt", Value.create(0.093));
    r.set("Status2.OK", Value.create(0.907));
```

PMML ***PMML*** code (Predictive Model Markup Language). Select *PMML* for Code Type in the Generate Code node wizard.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Note that some advanced properties of trees, such as open-ended intervals in splits on continuous variables, may not be expressed in PMML.

Also, note that the scoring output of a PMML model may be different from the scoring output of the original model in some cases. For example, PMML 4.2 specifies that the <CategoricalPredictor> element is to never match a missing value: all the dummy variables end up as zero. Hence, if you do not carry out any missing value substitution, PMML scoring will not agree with scoring performed with the Score node, which allows for missing value matches.

Python

(for Decision Trees and Strategy Trees)

Code in the ***Python*** programming language. Select *Python* for Code Type in the Generate Code node wizard.

The Python code for the first split in the above tree example is given below.

- The first part of the code is the TreeModel class that traverses the tree branches and returns probabilities of target values of the DV along with the node numbers / IDs and node sizes.
- The second part of the code - starting with `colNodeNumber = []` - is the actual deployment (scoring) code that scans the records of the input dataset over the columns representing the model variables and produces the output scores (model predictions) along with node numbers. In the example below, the scoring input dataset is called "Credit_DF". When using the code for deployment, this string is to be replaced with the actual input dataset name.

```
class TreeModel:
        DV = 'Account_Status'
        IVs = ['Residential Status']
        DVCategories = ['Bad', 'Good']
        Weight = 1
        @staticmethod
        def predict(Residential_Status):
                if Residential_Status is None \
                or Residential_Status == 'Live with Parents' \
                or Residential_Status == 'Other':
                        return ['x.26/0', 2, 0.034853354, 0.96514664, 9956]
                if Residential_Status == 'Own Home':
                        return ['x.26/1', 3, 0.02758888, 0.9724111, 5147]
                if Residential_Status == 'Rent':
                        return ['x.26/2', 4, 0.09296198, 0.90703801, 5314]
                return ['x', 1, 0.048146152, 0.951853847, 20417]

colNodeNumber = []
colValues = []
for n in range(len(TreeModel.DVCategories)):
        colValues.append([])
colPrediction = []
for index, row in Credit_DF.iterrows():
        values = TreeModel.predict(*[row[column] for column in TreeModel.IVs])
        colNodeNumber.append(values[1])
        probability = 0
        prediction = None
        for n in range(len(TreeModel.DVCategories)):
                colValues[n].append(values[n + 2])
                if values[n + 2] > probability:
                        probability = values[n + 2]
                        prediction = TreeModel.DVCategories[n]
        colPrediction.append(prediction)
Credit_DF[TreeModel.DV + ' Node Number'] = colNodeNumber
for n in range(len(TreeModel.DVCategories)):
        Credit_DF[TreeModel.DV + ' ' + TreeModel.DVCategories[n] + ' Probability'] =
colValues[n]
Credit_DF[TreeModel.DV + ' Prediction'] = colPrediction
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

SAS Language

Code in the SAS Language for statistical programming. Select *SAS Language* for Code Type in the Generate Code node wizard.

Generated SAS Language code can be run in SAS Systems that include the packages called "Base SAS" and "SAS/STAT".

R

Code in the ***R*** language for statistical programming. For more information on R, see https://www.r-project.org/. Select *R* for Code Type in the Generate Code node wizard.

The R code for the above example tree is given below.

- The first part of the code - `converters = list(...)` - provides converter functions for date, time, and datetime variables. These functions are used in the main part of the code if the tree contains splits on variables of these types.
- The main part of the code is the *score_decision_tree* function. It describes the tree rules as a list, with the node_scores dataframe defining the dependent variable probabilities for each node. The *scoring_fields* dataframe defines the fields to be created as a result of scoring, which will contain the probabilities and node numbers assigned to the input records.
- The last part of the code, which consists of the three lines at the end, is the actual deployment (scoring) code that applies the instructions defined by *score_decision_tree* to the input dataset and produces the output scores and node numbers, writing them to *scoring_fields*, the output dataframe being *scored_data*. When using the code for deployment in R, substitute the required input and output dataframe names. In this example, the scoring input dataset is Data_New, and the output is Data_New_Scored_R:

```
scoring_fields = score_decision_tree(Data_New)
scored_data = cbind(Data_New, scoring_fields)
scored_data -> Data_New_Scored_R
```

The R code for the above example tree:

```
converters = list(
  to_date = function(x, format = "%Y-%m-%d") as.Date(x, format = format),
  to_time = function(x, format = "%H:%M:%S") as.numeric(as.POSIXct(x, format = format,
tz = "UTC")) %% 86400,
  to_datetime = function(x, format = "%Y-%m-%dT%H:%M:%S") as.POSIXct(x, format =
format, tz = "UTC")
)
score_decision_tree = structure(
 function(data) {
  n = nrow(data)
  node_membership = with(data,
   lapply(
    list(
     rep(TRUE, n),
     ((is.na(`Residential_Status`)) | (`Residential_Status` %in% c("Live with
Parents", "Other"))),
     (`Residential_Status` %in% c("Own Home")),
     (`Residential_Status` %in% c("Rent")),
     (`Residential_Status` %in% c("Rent")) & (`age` >= 17.0 & `age` < 26.0),
     (`Residential_Status` %in% c("Rent")) & (`age` >= 26.0 & `age` < 61.0),
     (`Residential_Status` %in% c("Rent")) & ((is.na(`age`)) | (`age` >= 61.0 & `age`
<= 98.0))
    ),
    function(x) ifelse(is.na(x), FALSE, x)
   )
  )
  node_scores = data.frame(
   node_id = c("x", "x.26/0", "x.26/1", "x.26/2", "x.26/2.25/0", "x.26/2.25/1",
"x.26/2.25/2"),
   node_predict_prob_Bad = c(0.04814615, 0.03485335, 0.0275888, 0.0929619, 0.11498973,
0.0932509, 0.043668122),
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
   node_predict_prob_Good = c(0.9518538, 0.96514664, 0.9724111, 0.907038, 0.88501026,
0.906749, 0.9563318),
   node_prediction = c("Good", "Good", "Good", "Good", "Good", "Good", "Good"),
   node_prediction_probability = c(0.95185384, 0.96514664, 0.97241111, 0.907038,
0.88501, 0.90674909, 0.95633187),
   node_size = c(20417.0, 9956.0, 5147.0, 5314.0, 974.0, 3882.0, 458.0),
   stringsAsFactors = FALSE
  )
  scoring_fields = data.frame(
   node_id = character(n),
   node_number = integer(n),
   node_predict_prob_Bad = numeric(n),
   node_predict_prob_Good = numeric(n),
   node_prediction = character(n),
   node_prediction_probability = numeric(n),
   node_size = numeric(n),
   stringsAsFactors = FALSE
  )
  for(node in seq_along(node_membership)) {
   scoring_fields[node_membership[[node]], "node_id"] = node_scores[node, "node_id"]
   scoring_fields[node_membership[[node]], "node_number"] = node
   scoring_fields[node_membership[[node]], "node_predict_prob_Bad"] =
node_scores[node, "node_predict_prob_Bad"]
   scoring_fields[node_membership[[node]], "node_predict_prob_Good"] =
node_scores[node, "node_predict_prob_Good"]
   scoring_fields[node_membership[[node]], "node_prediction"] = node_scores[node,
"node_prediction"]
   scoring_fields[node_membership[[node]], "node_prediction_probability"] =
node_scores[node, "node_prediction_probability"]
   scoring_fields[node_membership[[node]], "node_size"] = node_scores[node,
"node_size"]
  }
  return(scoring_fields)
 },
 weight = 1,
 dv_categories = c("Bad", "Good")
)
scoring_fields = score_decision_tree(Data_New)
scored_data = cbind(Data_New, scoring_fields)
scored_data -> Data_New_Scored_R
```

C

Code in the ***C*** programming language. Select *C* for Code Type in the Generate Code node wizard.

The C code for the above example tree is given below.

```
char* dv = "Account_Status";
char* ivs[2] = {"age", "Residential Status"};
char* dvCategories[2] = {"Bad", "Good"};
double weight = 1;
char* getColumnValue(char* column, char** row, int columnCount,
char** columns)
{
for (int n = 0; n < columnCount; ++n)
{
if (strcmp(column, columns[n]) == 0)
return row[n];
}
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
fprintf(stderr, "Column %s not found in table!\n", column);
return NULL;
}
void predict(char** row, int ColumnCount, char** columns, char**
treeNodeID, int* treeNodeNumber, double* probabilities, double*
treeNodeWeight)
{
char* Residential_Status = getColumnValue("Residential Status",
row, ColumnCount, columns);
char* age = getColumnValue("age", row, ColumnCount, columns);
double age1 = (age != NULL) ? strtod(age, NULL) : 1E308;
if (Residential_Status == NULL ||
(Residential_Status != NULL && strcmp(Residential_Status, "Live
with Parents") == 0) ||
(Residential_Status != NULL && strcmp(Residential_Status, "Other")
== 0))
{
*treeNodeID = "x.26/0";
*treeNodeNumber = 2;
probabilities[0] = 0.03485335476094817;
probabilities[1] = 0.9651466452390518;
*treeNodeWeight = 9956;
return;
}
if ((Residential_Status != NULL && strcmp(Residential_Status, "Own
Home") == 0))
{
*treeNodeID = "x.26/1";
*treeNodeNumber = 3;
probabilities[0] = 0.027588886730134057;
probabilities[1] = 0.972411113269866;
*treeNodeWeight = 5147;
return;
}
if ((Residential_Status != NULL && strcmp(Residential_Status,
"Rent") == 0))
{
if (age == NULL ||
(age != NULL && (age1 >= 61 && age1 <= 98)))
{
*treeNodeID = "x.26/2.25/2";
*treeNodeNumber = 7;
probabilities[0] = 0.043668122270742356;
probabilities[1] = 0.9563318777292577;
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
*treeNodeWeight = 458;
return;
}
if ((age != NULL && (age1 >= 26 && age1 < 61)))
{
*treeNodeID = "x.26/2.25/1";
*treeNodeNumber = 6;
probabilities[0] = 0.09325090159711488;
probabilities[1] = 0.9067490984028851;
*treeNodeWeight = 3882;
return;
}
if ((age != NULL && (age1 >= 17 && age1 < 26)))
{
*treeNodeID = "x.26/2.25/0";
*treeNodeNumber = 5;
probabilities[0] = 0.11498973305954825;
probabilities[1] = 0.8850102669404517;
*treeNodeWeight = 974;
return;
}
*treeNodeID = "x.26/2";
*treeNodeNumber = 4;
probabilities[0] = 0.0929619872036131;
probabilities[1] = 0.9070380127963868;
*treeNodeWeight = 5314;
return;
}
*treeNodeID = "x";
*treeNodeNumber = 1;
probabilities[0] = 0.048146152715874024;
probabilities[1] = 0.9518538472841259;
*treeNodeWeight = 20417;
return;
}
```

SPSS

SPSS programming language. Select *SPSS* for Code Type in the Generate Code node wizard.

Generated SPSS code can be run in SPSS Statistics packages.

SQL Case

Tree rules in SQL code. Select *SQL* for Code Type in the Generate Code node wizard.

This is a nested {CASE WHEN} code - a single scoring instruction representing the tree model down to its terminal nodes ("decision nodes"). The generated code can be run in any system that supports SQL. The code follows Microsoft SQL Server standards (Transact SQL). Some adjustments in SQL syntax may be necessary if the code is deployed in database systems that have their own SQL dialects, such as Oracle or DB2.

An example fragment of a SQL Case code fragment:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
(CASE
     WHEN ("Residential Status" is null OR "Residential Status"
= 'Live with Parents') THEN  0.0361
     WHEN  ("Residential  Status"  =  'Other'  OR  "Residential
Status" = 'Own Home') THEN 0.030
     WHEN "Residential Status" = 'Rent' THEN 0.0934
ELSE 0.0492
END)
```

SQL Function

The SQL Function code creates a SQL function that takes attributes as arguments and returns a score or prediction. This makes it easier to deploy your tree models into a production environment. The code follows Microsoft SQL Server standards (Transact SQL). Some adjustments in SQL syntax may be necessary if the code is deployed in database systems that have their own SQL dialects, such as Oracle or DB2.

Select *SQL Function* for Code Type in the Generate Code node wizard.

An example fragment of SQL Function code:

```
CREATE   FUNCTION   DBO.SOLN_STATUS_X0020_TREE_PREDICTPROBABILITY(
@Residential_Status nvarchar(17))
  RETURNS float
     BEGIN
     RETURN  CASE
        WHEN  @Residential_Status  IN  ('Live  with  Parents',  'Own
Home') THEN  0.9697
         WHEN @Residential_Status = 'Rent' THEN 0.9066
  ELSE 0.9507
 END
END;
```

SQL SELECT

The SQL SELECT code provides rules for each node of the tree. It does not include probabilities (in the case of Decision Trees) or treatments (in the case of Strategy Trees). Choose *SQL Select* for Code Type in the Generate Code node wizard.

An example fragment of SQL SELECT code:

```
SELECT * FROM %table%
WHERE "Residential Status" = 'Rent' AND
("age" >= 30 AND "age" < 45);
```

XML

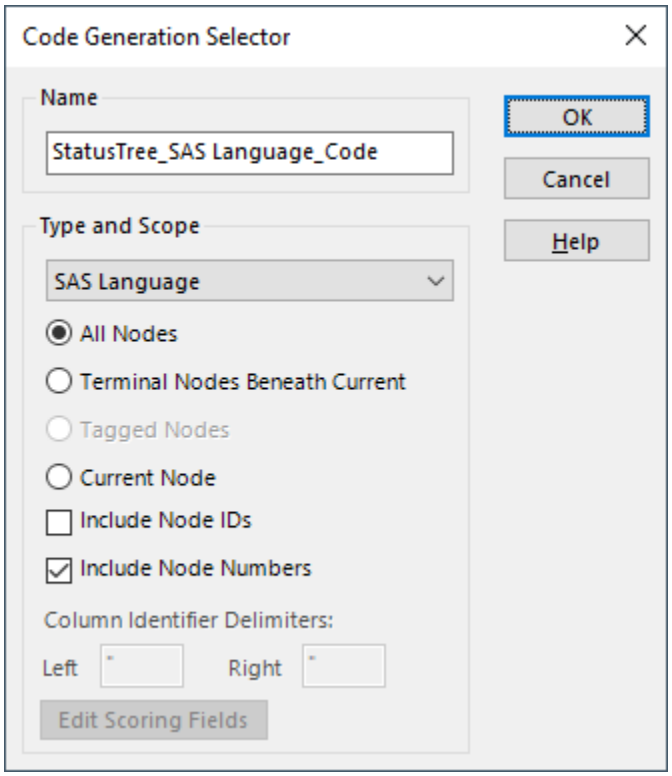
All types of models, including decision trees and strategy trees, are stored in the file system as XML files. Generating XML code for a model is therefore equivalent to displaying the contents of the model file. Note that the generated XML code can be imported using the node **Model from XML/PMML**, so this type of code can be used, for example, for sharing tree models with other users.

Select *XML* for Code Type in the Generate Code node wizard.

Scope Options

These options are available only when generating code from the interactive tree view using the menu command **Insert | Code Generation.** Open a tree and select the **Code Generation** command from the **Insert** menu. This brings up the Code Generation Selector dialog, where you can choose the desired code type and scope (the subset of tree nodes to generate the code for).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



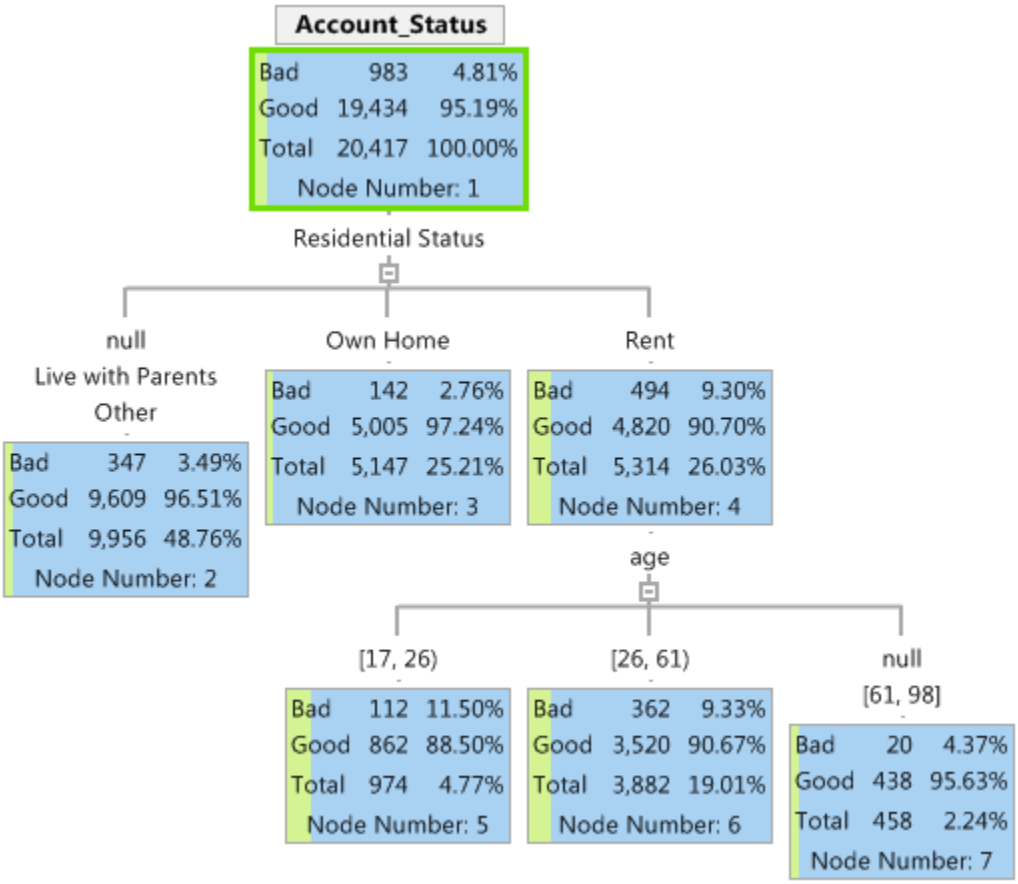
All Nodes	Generates code for all nodes in the tree.
Terminal Nodes Beneath Current	Generates code for all *terminal* nodes (leaves) in the branch beneath the currently selected node. If the selected node is the root node, code will be generated for all terminal nodes in the tree.
Tagged Nodes	Generates code for the tagged nodes only. This option is enabled only if at least one node in the tree is tagged. To tag a node, right-click on it and select "**Tag/Untag Node**" from the context menu.
Current Node	Generates code for the currently selected node only.

You can also include Node Numbers or Node IDs in the generated code. Note: To display node numbers or node IDs for all nodes in the tree view, select them in the **Tree Node Settings** tab of the Tree Options dialog available via the **Tools | Options** command.

Example 1: SQL Code for Decision Trees

A fragment of SQL Case code for the following Decision Tree is given below.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.




```
-- SQL Predictive Model
--Block # 1: Calculates the probability that '(Account_Status)' equals 'Bad'
(CASE
    WHEN ("Residential Status" is null or "Residential Status" = 'Live with Parents'
    or "Residential Status" = 'Other') THEN
          0.03485335476094817
    WHEN "Residential Status" = 'Own Home' THEN
          0.027588886730134057
    WHEN "Residential Status" = 'Rent' THEN
          (CASE
          WHEN ("age" >= 17.0e0 and "age" < 26.0e0) THEN
                0.11498973305954825
          WHEN ("age" >= 26.0e0 and "age" < 61.0e0) THEN
                0.09325090159711488
          WHEN ("age" is null or ("age" >= 61.0e0 and "age" <= 98.0e0)) THEN
                0.043668122270742356
          ELSE
          0.0929619872036131
    END)
    ELSE
    0.048146152715874024
END)
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

For Strategy Trees, the SQL code is generated for both treatments and node calculations (if they are defined). SAS Language code is generated for treatments only.

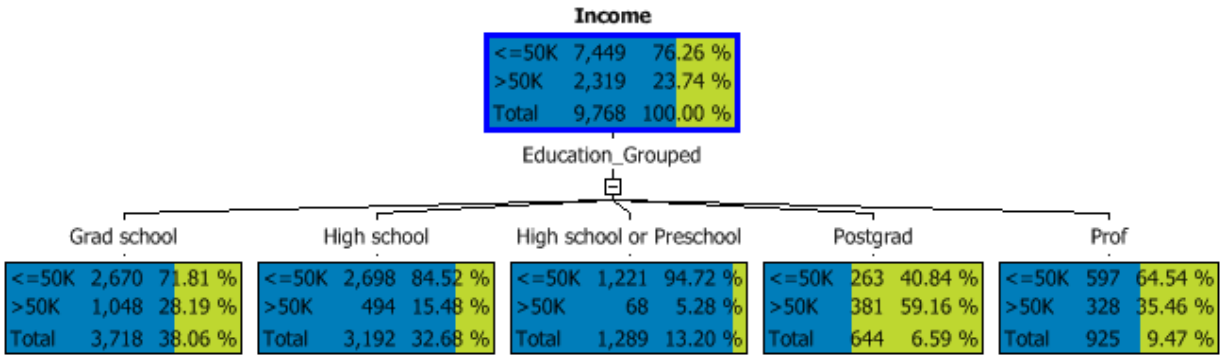
Calculated columns in generated SQL code

If a tree has splits on calculated columns, SQL Code generation uses SQL functions for calculated columns instead of inline substitution of their expressions. This is especially convenient in the cases where there are many splits on calculated columns and their expressions are long and complex.

For example, suppose you have a calculated column called Education_Grouped obtained by merging categories in your original education column:

CASE WHEN ([education] = '10th' or [education] = '11th' or [education] = '12th' or [education] = '1st-4th' or [education] = '5th-6th' or [education] = '7th-8th' or [education] = '9th' or [education] = 'Preschool') THEN 'High school or Preschool' WHEN ([education] = 'Assoc-acdm' or [education] = 'Assoc-voc' or [education] = 'Prof-school') THEN 'Prof' WHEN ([education] = 'Bachelors' or [education] = 'Some-college') THEN 'Grad school' WHEN ([education] = 'Doctorate' or [education] = 'Masters') THEN 'Postgrad degree' WHEN ([education] = 'HS-grad') THEN 'High school grad' END

Suppose your tree with Income as the dependent variable has a split on the new column Education_Grouped:



Having the whole expression substituted in the SQL code for every split on Education_Grouped that may occur in the tree could make the code extremely long, so the code always uses SQL functions. The initial fragment of the SQL code generated for this tree is as follows:

```
create function EducationGrouped
(
      @education nvarchar(50)
)
returns nvarchar(50)
as
begin
      return CASE WHEN @education = '10th' OR @education = '11th' OR @education = '12th' OR @education = '1st-4th' OR @education = '5th-6th' OR @education = '7th-8th' OR @education = '9th' OR @education = 'Preschool' THEN 'High school or Preschool' WHEN @education = 'Assoc-acdm' OR @education = 'Assoc-voc' OR @education = 'Prof-school' THEN 'Prof' WHEN @education = 'Bachelors' OR @education = 'Some-college' THEN 'Grad school' WHEN @education = 'Doctorate' OR @education = 'Masters' THEN 'Postgrad degree' WHEN @education = 'HS-grad' THEN 'High school grad' END
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

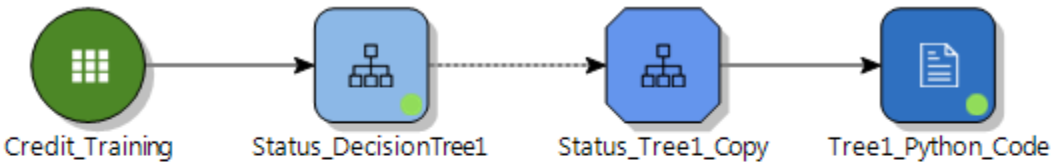
```
end
go

--Block # 1: Calculates the probability that '(Income)' equals '<=50K'
(CASE
WHEN (dbo.EducationGrouped([education])) = 'Grad school' THEN
        0.7174364711523095
WHEN (dbo.EducationGrouped([education])) = 'High school grad' THEN
        0.8432708688245315
WHEN (dbo.EducationGrouped([education])) = 'High school or Preschool' THEN
        0.9433874709976798
WHEN (dbo.EducationGrouped([education])) = 'Postgrad degree' THEN
        0.43946188340807174
WHEN (dbo.EducationGrouped([education])) = 'Prof' THEN
        0.6580557443915703
Else
        0.7637737239727289
END)
```

Example 2: Deploying Python Code Generated for a Decision Tree

To score a dataset with Python code generated for a decision tree model, you can use the **Python Code** node from the Action palette:

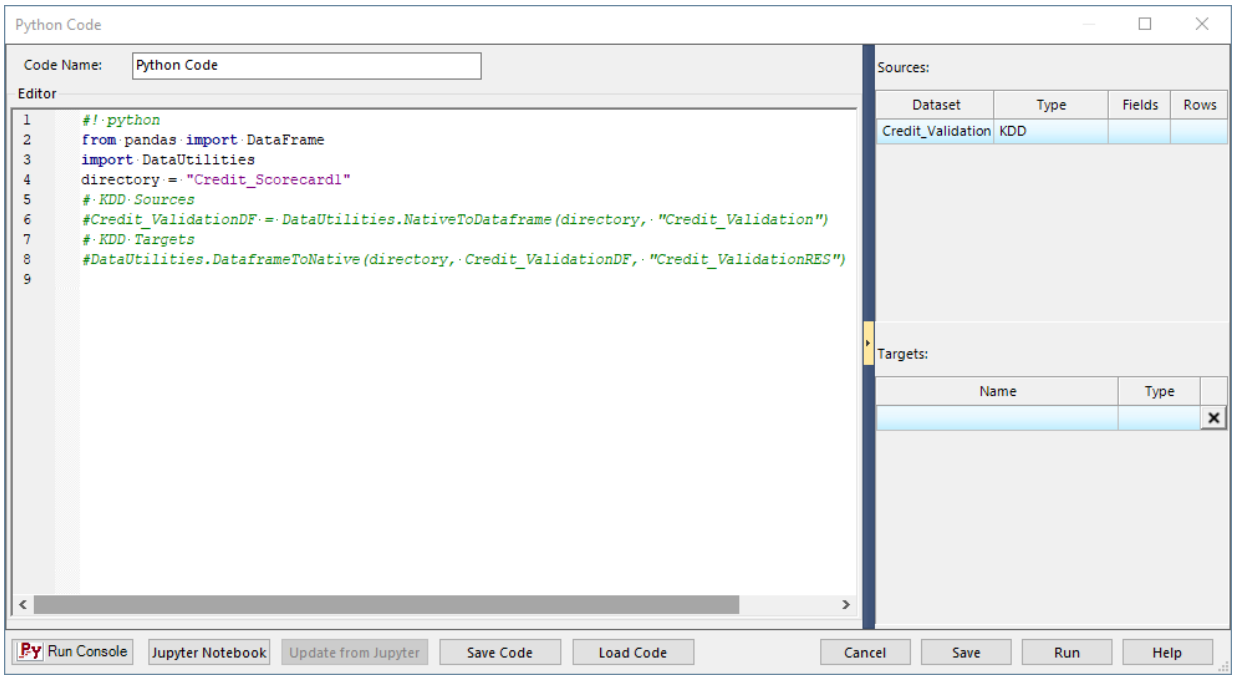
1. Connect the input dataset that you need to score to the **Python Code** node or use an unattached **Python Code** node and obtain the input Pandas dataframe by other means.





2. In the **Python Code** node, uncomment the lines following “# KDD sources” and “# KDD targets”.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

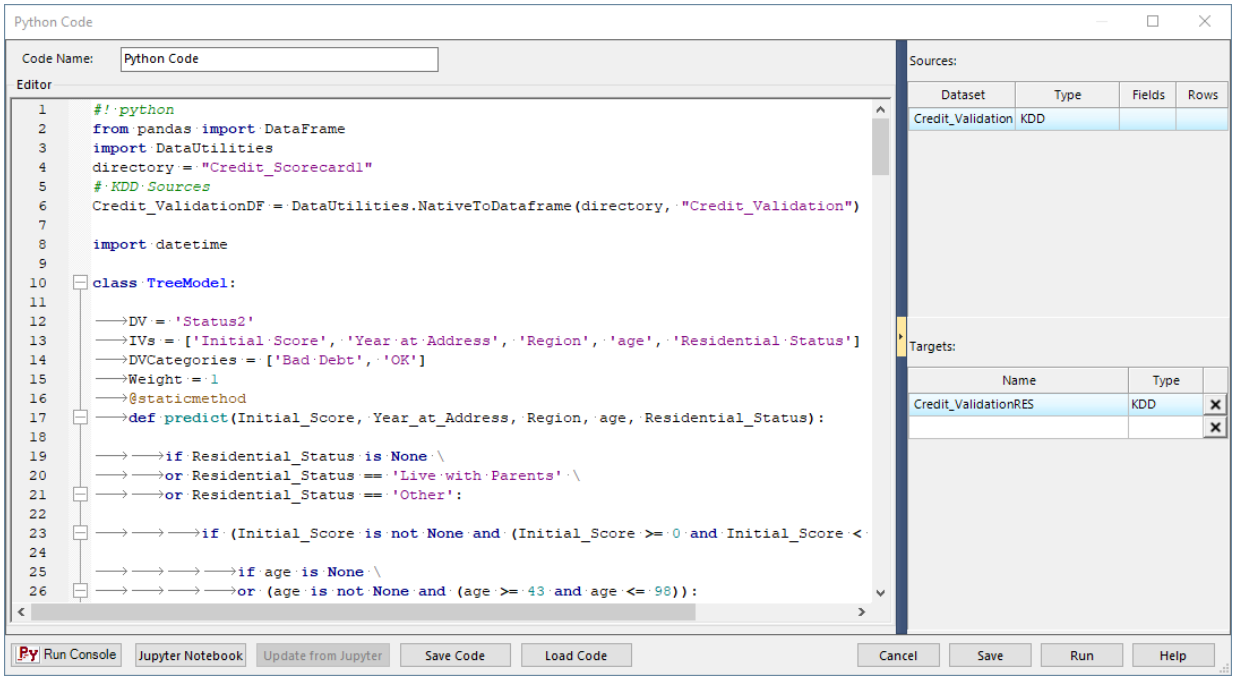


3. Paste the entire generated code between the blocks “# KDD sources” and “# KDD targets”.

4. If necessary, change the name of the source dataset to match the actual input dataset name.

5. Change the target (output) dataset name in the last line of the code, if desired. This is the name of the output dataset that will contain the scores.

6. Specify the target (output) dataset name at the bottom-right in the Targets panel.



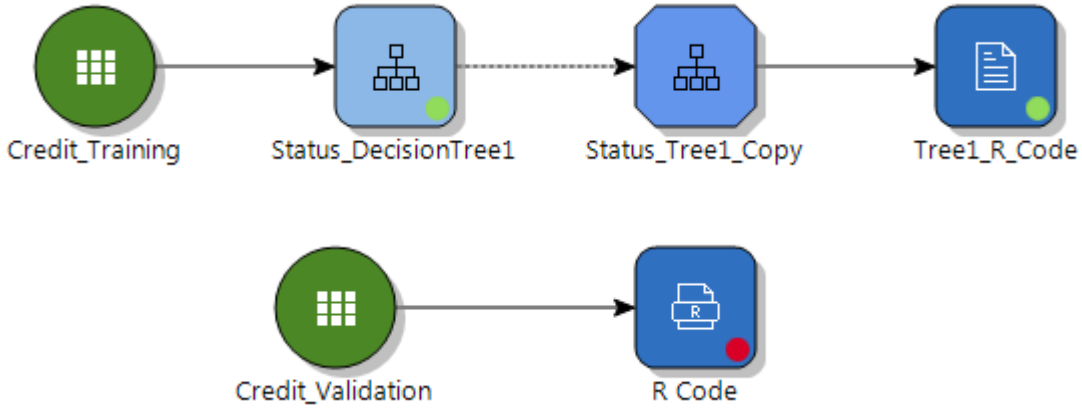
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

7. Run the Python code node.

Example 3: Deploying R Code Generated for a Decision Tree

To score a dataset with R code generated for a decision tree model, you can use the **R Code** node from the Action palette:

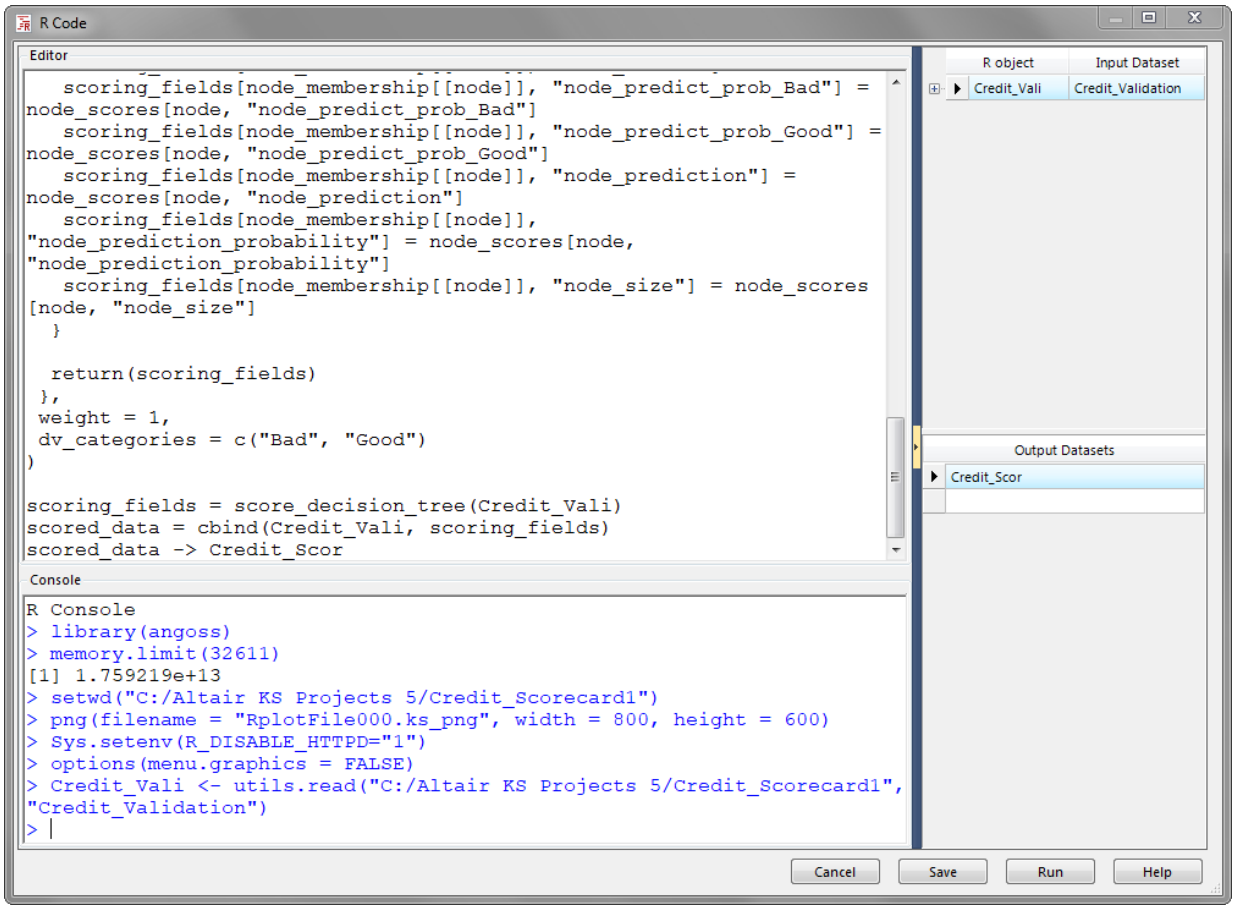
1. Connect the input dataset that you need to score to the **R Code** node or use an unattached **R Code** node and obtain the input R dataframe by other means.



2. Double-click the **R Code** node to open the console interface. The Console frame contains the executed code that read your input dataset into an R dataframe.

3. Paste the entire generated R code to the Editor frame of the R Code window:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

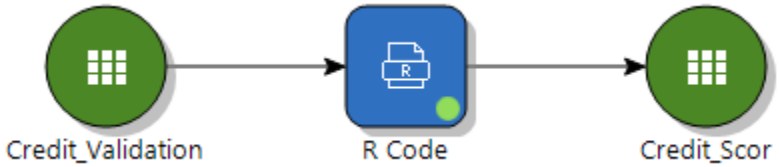


4. If necessary, change the name of the source dataset in the last lines of the code to match the actual input dataframe name. (In the above example, it is "Credit_Vali".)

5. Change the output dataset name in the last line of the code, if desired. This is the name of the output dataset that will contain the scores. (In the above example, it is "Credit_Scor".)

6. Specify the output dataset name at the bottom-right in the **Targets** panel.

7. Run the R code node. The output dataset contains the scores.

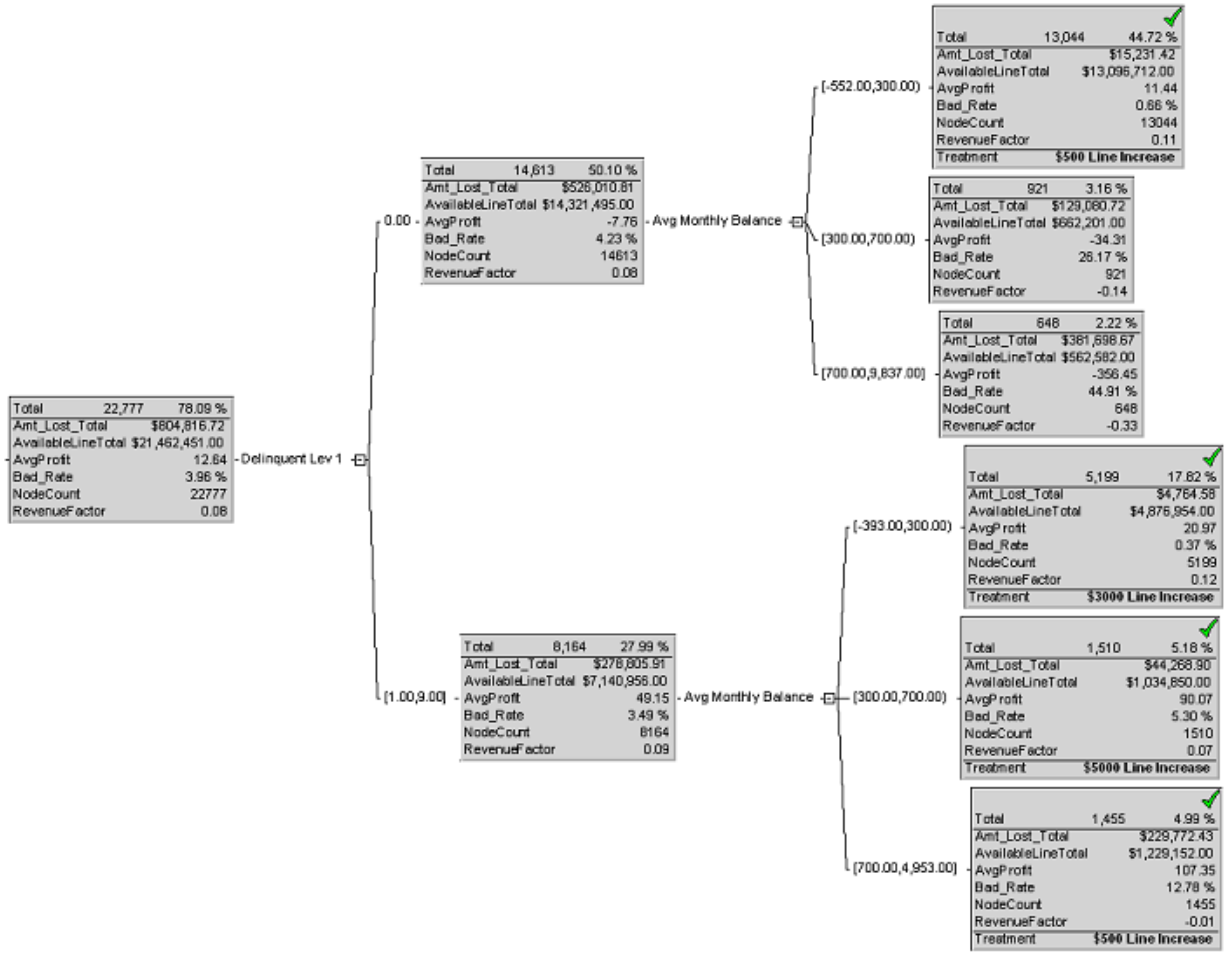


Example 4: SAS Language Code for Strategy Trees

A fragment of SAS Language code generated for the following Strategy Tree is given below. After the code is generated, you can use it for scoring in the corresponding business intelligence application or database system. For example, if you generated SAS Language code, you can save

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

it by selecting **File | Save As...** command from the menu. The code is saved as a SAS Language file (a text file with extension *.sas), which you can run in SAS System. You can also copy and paste any part of the generated code to a SAS System program window. Note that you may need to change the names of the input and output datasets appropriately.



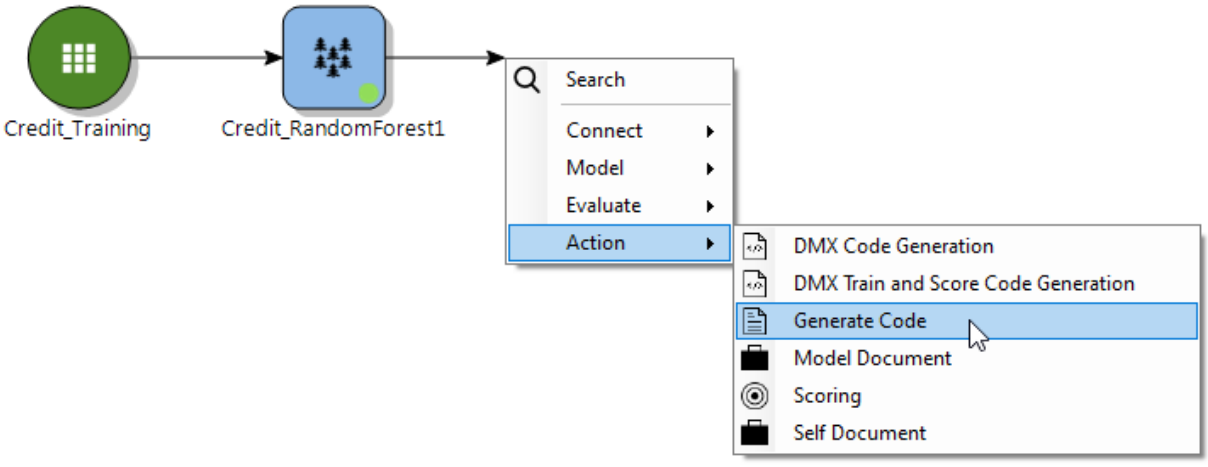

```
IF DELINQUENT_LEV_1 = 0 THEN
DO;
      IF AVG_MONTHLY_BALANCE >= -552 AND AVG_MONTHLY_BALANCE < 300 THEN
      DO;
            Treatment = '$500 Line Increase';
      END;
END;
IF DELINQUENT_LEV_1 >= 1 AND DELINQUENT_LEV_1 <= 9 THEN
DO;
      IF AVG_MONTHLY_BALANCE >= -393 AND AVG_MONTHLY_BALANCE < 300 THEN
      DO;
            Treatment = '$3000 Line Increase';
      END;
END;
IF DELINQUENT_LEV_1 >= 1 AND DELINQUENT_LEV_1 <= 9 THEN
DO;
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

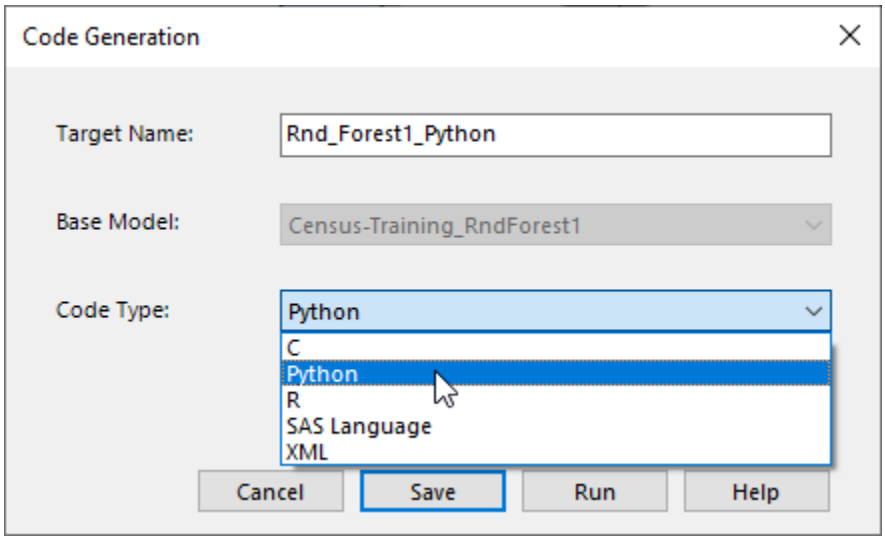
```
        IF AVG_MONTHLY_BALANCE >= 300 AND AVG_MONTHLY_BALANCE < 700 THEN
        DO;
              Treatment = '$5000 Line Increase';
        END;
END;
IF DELINQUENT_LEV_1 >= 1 AND DELINQUENT_LEV_1 <= 9 THEN
DO;
        IF AVG_MONTHLY_BALANCE >= 700 AND AVG_MONTHLY_BALANCE <= 4953 THEN
        DO;
              Treatment = '$500 Line Increase';
        END;
END;
```

2. Python and R Code Generation for Ensemble Trees (Random Forest, Bagging, and Boosting)

2.1. Python Code for Ensemble Trees



Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Python code generated for ensemble trees (Random Forest, Bagging, or Boosting) is a nested code that comprises the code modules for each individual tree in the ensemble. For example, the code structure for a random forest consisting of only 3 trees, for the sake of simplicity, is as follows:

```
class TreeModels:
        class TreeModel1:
                <TreeModel1 code> (as described in Section 1 above)
        class TreeModel2:
                 <TreeModel2 code>
        class TreeModel3:
                 <TreeModel2 code>
```

The **scoring part** of the code performs the following actions:

For each input row:

- calculate the average of the probabilities for the dependent variable values over all trees in the ensemble;
- set the prediction to the dependent variable value with the highest average probability.

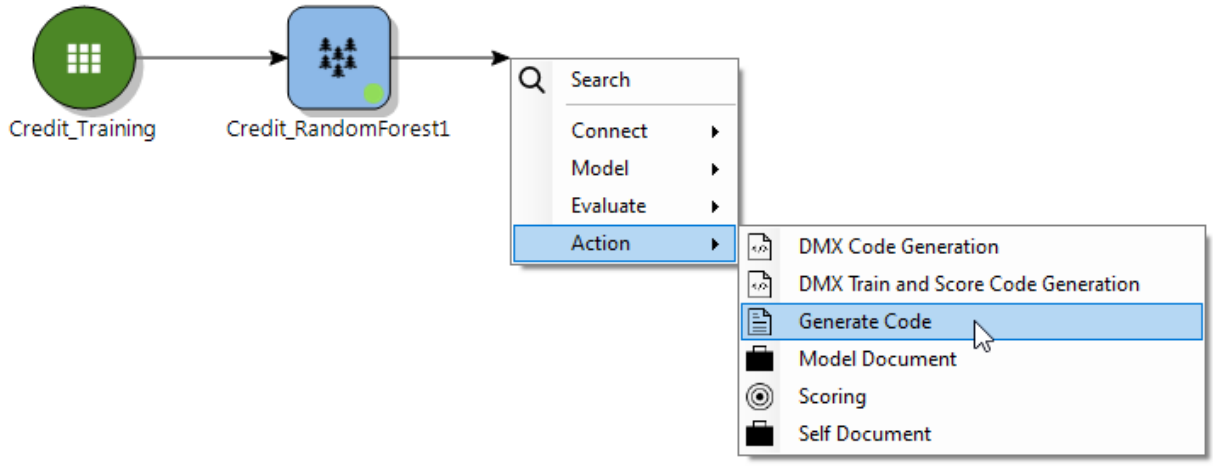
```
classNames = []
for key in TreeModels.__dict__.keys():
              if key.startswith("TreeModel"):
                            classNames.append(key)
numTrees = len(classNames)
colValues = []
colPrediction = []
dv = None
dvCategories = None
for index, row in Credit_TrainingDF.iterrows():
              avgValues = []
```

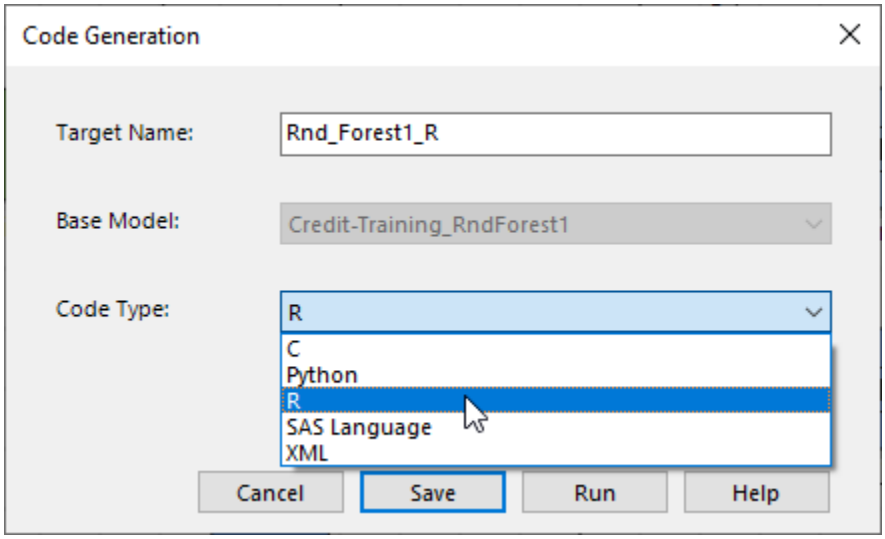
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
            for className in classNames:
                        classItem = getattr(TreeModels, className)
                        if dv is None:
                                    dv = classItem.DV
                                    dvCategories = classItem.DVCategories
                                    for n in range(len(dvCategories)):
                                                colValues.append([])
                        if len(avgValues) == 0:
                                    for n in range(len(dvCategories)):
                                                avgValues.append(0)
                        values = classItem.predict(*[row[column] for column in classItem.IVs])
                        for n in range(len(dvCategories)):
                                    avgValues[n] += values[n + 2]
              for n in range(len(dvCategories)):
                        avgValues[n] /= numTrees
            probability = 0
            prediction = None
            for n in range(len(dvCategories)):
                        colValues[n].append(avgValues[n])
                        if avgValues[n] > probability:
                                    probability = avgValues[n]
                                    prediction = dvCategories[n]
               colPrediction.append(prediction)
for n in range(len(dvCategories)):
                        Credit_TrainingDF[dv + ' ' + dvCategories[n] + ' Probability'] = colValues[n]
Credit_TrainingDF[dv + ' Prediction'] = colPrediction
```

2.2. R Code for Ensemble Trees

Altair Engineering, Inc. Copyright © 2022. All rights reserved.





R code generated for ensemble trees (Random Forest, Bagging, or Boosting) comprises the code modules for the member trees and the calculation of scores for the ensemble model. The structure of the code is described below.

Just as in the R code for a single decision tree, the first part of the code provides converter functions for date, time, and datetime variables. These functions are used in the main code whenever a split on such a variable occurs in any of the member trees.

The ensemble function that describes all the tree rules for calculating the DV probabilities and defines the score fields to be created is a **list** containing the corresponding parts of the code for the member trees. It is followed by the part that generates the scores for the ensemble.

For example, the code structure for a random forest consisting of 3 trees (for the sake of simplicity) is as follows:

```
decision_tree_scoring_functions = list(
 score_decision_tree1 = structure( ... )
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
    score_decision_tree2 = structure( ... )
    score_decision_tree3 = structure( ... )
  )
  <..scoring code..>
```

For the details of the code for *score_decision_tree* module for each individual tree, see the R code description in the table in the beginning of Section 1.

The **scoring part** of the code performs the following actions:

- for each input row:
  - calculate the average of the probabilities for the dependent variable values over all trees in the ensemble;
  - set the prediction to the dependent variable value with the highest average probability.

Here is an example of the scoring part of the code (with comments). In the last 3 lines, Cr_Scor_In and Cr_Scor_Out are the scoring input and output dataframes, respectively.

```
#' Function for averaging scores over trees
#' @param data -- dataframe to score
#' @param scoring_functions -- list of scoring functions -- one function
for each tree
#' @return dataframe with averaged scoring fields

averaging_function = function(data, scoring_functions) {

  #' apply each of a list of functions to the same objects
  #' @param FUNS -- list of functions
  #' @param ... -- arguments to pass to the functions
  #' @return list of results -- one result for each function

  fapply = function(FUNS, ...) lapply(FUNS, function(func) func(...))

  #' average a scoring column over all trees in a forest
  #' @param column -- name of column to extract
  #' @return vector containing the average scores

  average_column = function(column) {
      (sapply(forest_scores,   `[[`,   column)   %*%   tree_weights)   /
sum_tree_weights
  }

  #' extract an attribute from all scoring functions
  #' @param attribute -- name of attribute to extract
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
  extract_attribute = function(attribute) sapply(scoring_functions,
attr, attribute)

  # compute the scores for the entire forest

  forest_scores = fapply(scoring_functions, data)

  # extract the scoring column names
  # -- they're in position 3 to 2 + the number of scoring fields

  score_col_names =
names(forest_scores[[1]])[3:(ncol(forest_scores[[1]]) - 3)]

  # extract and sum tree weights

  tree_weights = extract_attribute("weight")
  sum_tree_weights = sum(tree_weights)

  # compute the averages and reshape to dataframe

  scored_data = as.data.frame(
      setNames(
          lapply(score_col_names, average_column),
          score_col_names
      )
    )

  # compute predictions

  dv_categories = attr(scoring_functions[[1]], "dv_categories")
  scored_data$prediction = dv_categories[apply(scored_data, 1,
which.max)]
  return(scored_data)
}

scoring_fields = averaging_function(Cr_Scor_In,
decision_tree_scoring_functions)
scored_data = cbind(Cr_Scor_In, scoring_fields)
scored_data -> Cr_Scor_Out
```

3. Code Generation for Other Predictive Models and Cluster Analysis

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The following code types can be generated for *Linear Regression*, *Logistic Regression*, *Constrained Regression*, *Regularization*, *Deep Learning (Neural Network)*, and *Cluster Analysis*.

PMML	PMML (Predictive Model Markup Language). Note that PMML code generated in Altair Data Analytics or other analytics applications, such as *R*, can be imported into Knowledge Seeker or Knowledge Studio using the node **Model from XML/PMML**, so this type of code can be used, for example, for sharing models between users. See the topic Insert Model from XML/PMML for details. Note: The scoring output of a PMML model may be different from the scoring output of the original model in some cases. For example, PMML 4.2 specifies that the <CategoricalPredictor> element is to never match a missing value: all the dummy variables end up as zero. Hence, if you do not carry out any missing value substitution, PMML scoring will not agree with scoring performed with the Score node, which allows for missing value matches.
SAS Language	Code in the SAS Language for statistical programming. Generated SAS Language code can be run in SAS Systems that include the packages called "Base SAS" and "SAS/STAT".
Python	Code in the Python programming language. For models that include automatic variable normalization, the normalization code is generated as well and included before the scoring equations code. Attribute Editor settings such as missing value substitution are also automatically translated and included in the code.
R	Code in the *R* programming language. For models that include automatic variable normalization, the normalization code is generated as well and included before the scoring equations code. Attribute Editor settings such as missing value substitution are also automatically translated and included in the code.
C	Code in the *C* programming language. For models that include automatic variable normalization, the normalization code is generated as well and included before the scoring equations code. Attribute Editor settings such as missing value substitution are also automatically translated and included in the code.
SQL Function	The SQL Function code creates a SQL function that takes attributes as arguments and returns a score or prediction. This makes it easier to deploy your models into a production environment. The code follows the Microsoft SQL Server standards. Some adjustments in SQL syntax may be necessary if the code is deployed in other database systems, such as Oracle. SQL Function code generation is available for all model types except Scorecards.
XML	All types of models except Survival Analysis are stored in the file system as XML files. Generating XML code for a model is therefore equivalent to displaying the contents of the model file. Note that the generated XML code can be imported into Knowledge Studio using the node **Model from XML/PMML**, so this type of code can be used, for example, for sharing models between users. See the topic Insert Model from XML/PMML for details.

After the code is generated, you can use it for scoring in the corresponding business intelligence application or database system. For example, if you generated SAS Language code, you can save it by selecting **File | Save As...** command from the menu. The code is saved as a SAS System program (a text file with extension *.sas), which you can run in SAS System. You can also copy and paste any part of the generated code to a SAS System program window. Note that you may need to change the names of the input and output datasets appropriately.

4. Code Generation for Factor Analysis and Principal Component Analysis

The code generation types supported for *Factor Analysis* and *Principal Component Analysis* are *SAS Language*, *Python,* and *R.*

The generated SAS Language code can be run in SAS Systems that include the packages called "Base SAS" and "SAS/STAT".

The **Generate Code** node produces expressions for factors (for the Factor Analysis node) and principal components (for the Principal Component Analysis node).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Example:

Suppose a Factor Analysis model with eight input variables *age, Amount_Lost, Avg_Monthly_Balance, Initial_Score, Number_Inquiries, Profit, Year_at_Address,* and *Year_Last_Check* produces four factors, whose scores are given in the Scores tab of the model object:

	Factor1	Factor2	Factor3	Factor4
age	0.00158	0.04500	-0.00116	-0.00610
Amount Lost	0.00163	0.00003	0.00010	0.00012
Avg Monthly Balance	0.00073	0.00004	-0.00025	-0.00025
Initial Score	-0.00002	0.00829	-0.00057	-0.00112
Number Inquiries	-0.00030	-0.00050	0.01172	0.00138
Profit	-0.00157	-0.00001	-0.00028	-0.00029
Year at Address	0.00070	-0.01180	-0.00446	0.13874
Year Last Check	0.00044	-0.00078	0.01596	-0.00311
Constant Coefficient	-0.30815	-4.42158	-0.46030	-0.07500

Summary | Loadings | Scores

Then the SAS Language code generated for this model will be as follows:

```
%MACRO KST_MODEL(DSInput, DSOutput);
DATA &DSOutput;
SET &DSInput;
  /*
  * Factor Analysis Expressions
  */
  Factor1 = 0.00158483254087207 * age + 0.00162652280305986 *
Amount_Lost + 0.000727739565747815 * Avg_Monthly_Balance
   +    -2.02307629953659E-05    *    Initial_Score    +    -
0.000300777367557212 * Number_Inquiries + -0.00157446715566619 *
Profit
   +     0.000695563391904633     *     Year_at_Address     +
0.000439534292050606 * Year_Last_Check + -0.30814735783863;
  Factor2 = 0.0450045064573354 * age + 2.95975622636913E-05 *
Amount_Lost + 3.83282937623245E-05 * Avg_Monthly_Balance
   + 0.00829353164442716 * Initial_Score + -0.000496537356636761
* Number_Inquiries + -1.13469876347307E-05 * Profit
   +    -0.0118020599753628    *    Year_at_Address    +    -
0.000783501151826687 * Year_Last_Check + -4.42157786231676;
  Factor3 = -0.00115517939712653 * age + 0.000103151256262668 *
Amount_Lost + -0.000248468282180936 * Avg_Monthly_Balance
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
   + -0.000568442311100854 * Initial_Score + 0.0117162766115178
* Number_Inquiries + -0.000276857658972099 * Profit
   + -0.00446251681993336 * Year_at_Address + 0.0159618361380709
* Year_Last_Check + -0.460295149168303;
  Factor4 = -0.00609574104712766 * age + 0.000118721533499808 *
Amount_Lost + -0.000246777430582782 * Avg_Monthly_Balance
   + -0.00111597439289469 * Initial_Score + 0.00137847595554339
* Number_Inquiries + -0.000291964428164577 * Profit
   + 0.13874098700079 * Year_at_Address + -0.0031142075720121 *
Year_Last_Check + -0.0750011163340014;
RUN;
%mend KST_MODEL;

/* Example use */
   /* DSinput: dataset to be scored */
   %let DSinput = xxx;
   /* DSOutput: output dataset with scores */
   %let DSOutput = yyy;
   %KST_MODEL(&DSInput, &DSOutput);
```

The R code generated for the same model is as follows:

```
converters = list(
  to_date = function(x, format = "%Y-%m-%d") as.Date(x, format =
format),
  to_time = function(x, format = "%H:%M:%S") as.numeric(as.POSIXct(x,
format = format, tz = "UTC")) %% 86400,
  to_datetime = function(x, format = "%Y-%m-%dT%H:%M:%S")
as.POSIXct(x, format = format, tz = "UTC")
)
# Model summary
# Variables
# [age] -> age
# [Amount Lost] -> Amount_Lost
# [Avg Monthly Balance] -> Avg_Monthly_Balance
# [Initial Score] -> Initial_Score
# [Number Inquiries] -> Number_Inquiries
# [Profit] -> Profit
# [Year at Address] -> Year_at_Address
# [Year Last Check] -> Year_Last_Check
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
# Factor Analysis Expressions
getPrediction = function(table) {
 # Variables
 # Expressions
 table$Factor1 = with(table, {
  0.00158483254087207 * age + 0.00162652280305986 * Amount_Lost +
0.000727739565747815 * Avg_Monthly_Balance +
  -2.02307629953659E-05 * Initial_Score + -0.000300777367557212 *
Number_Inquiries + -0.00157446715566619 * Profit +
  0.000695563391904633 * Year_at_Address + 0.000439534292050606 *
Year_Last_Check + -0.30814735783863
 })
 table$Factor2 = with(table, {
  0.0450045064573354 * age + 2.95975622636913E-05 * Amount_Lost +
3.83282937623245E-05 * Avg_Monthly_Balance +
  0.00829353164442716 * Initial_Score + -0.000496537356636761 *
Number_Inquiries + -1.13469876347307E-05 * Profit +
  -0.0118020599753628 * Year_at_Address + -0.000783501151826687 *
Year_Last_Check + -4.42157786231676
 })
 table$Factor3 = with(table, {
  -0.00115517939712653 * age + 0.000103151256262668 * Amount_Lost + -
0.000248468282180936 * Avg_Monthly_Balance +
  -0.000568442311100854 * Initial_Score + 0.0117162766115178 *
Number_Inquiries + -0.000276857658972099 * Profit +
  -0.00446251681993336 * Year_at_Address + 0.0159618361380708 *
Year_Last_Check + -0.460295149168303
 })
 table$Factor4 = with(table, {
  -0.00609574104712766 * age + 0.000118721533499808 * Amount_Lost + -
0.000246777430582782 * Avg_Monthly_Balance +
  -0.00111597439289469 * Initial_Score + 0.00137847595554339 *
Number_Inquiries + -0.000291964428164577 * Profit +
  0.13874098700079 * Year_at_Address + -0.0031142075720121 *
Year_Last_Check + -0.0750011163340014
 })
 table = table[c('Factor1', 'Factor2', 'Factor3', 'Factor4')]
 return(table)
}
scoringFields = getPrediction(Credit_Training)
scoredData = cbind(Credit_Training, scoringFields)
```

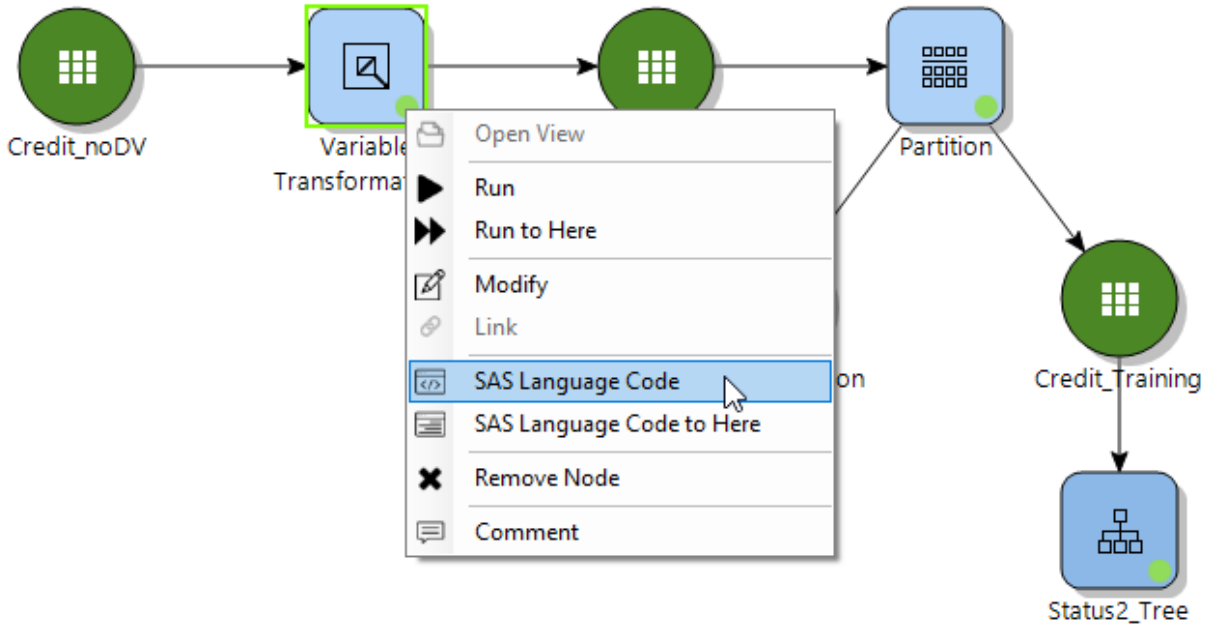
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
scoredData -> Credit_TrainingR
```

5. SAS Language Code Generation for Data Manipulation Nodes

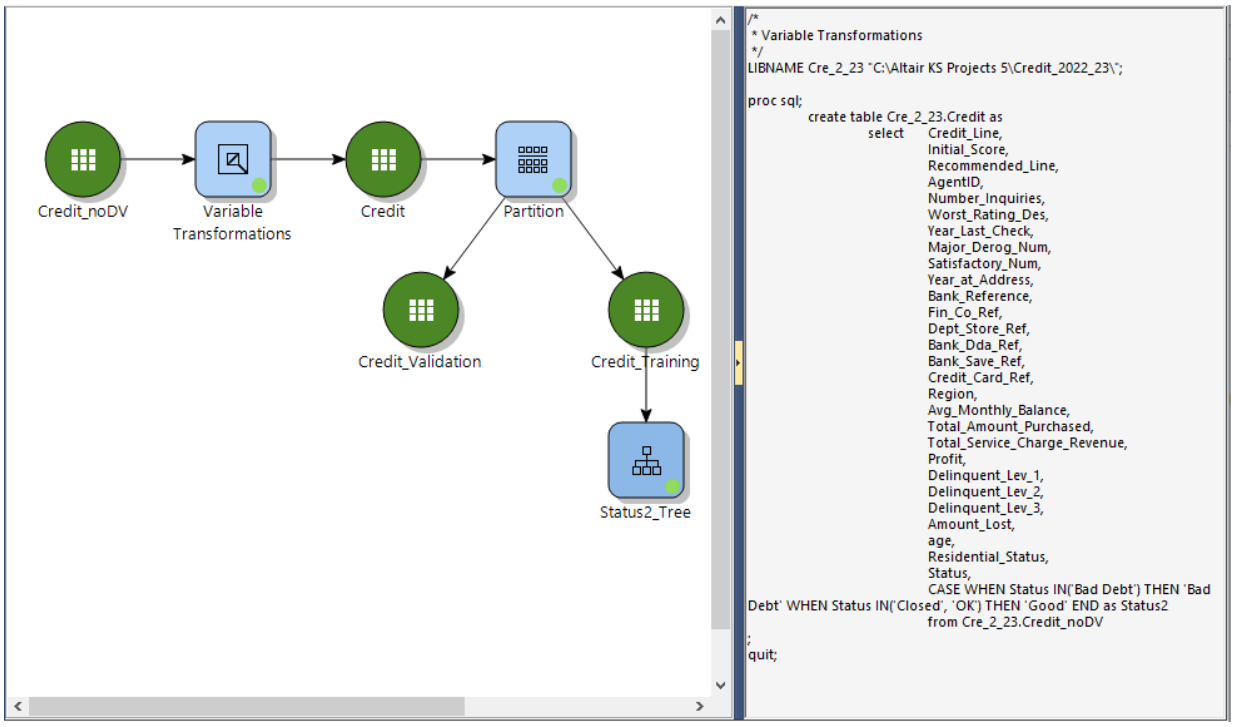
SAS Language code can be generated using the context menu for the following data profiling and manipulation nodes: Multicollinearity Analysis, Variable Selection, Aggregate, Append, Deduplicate, Filter, Filtering Join, Join, Partition, Variable Transformations, and WOE. The SAS Language Code context menu command is also enabled for the Scoring node.

To view SAS Language code for any of these nodes, right-click on it and select the **SAS Language Code** command from the context menu.

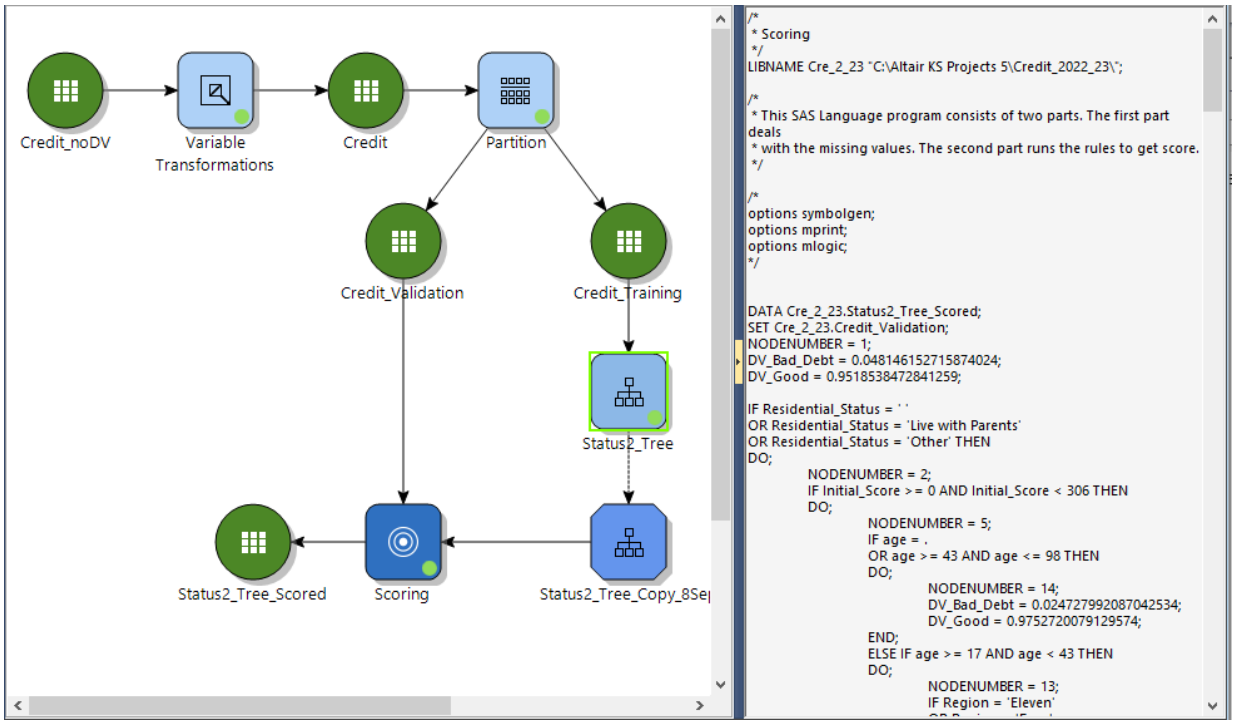


The SAS Language code generated for the selected node will be displayed in a new frame that opens on the right-hand side of the Workflow canvas. In this example, the code generated for the Variable Transformation node is displayed in the frame to the right of the workflow. To hide the frame, click on the arrow in the middle of the separator bar.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

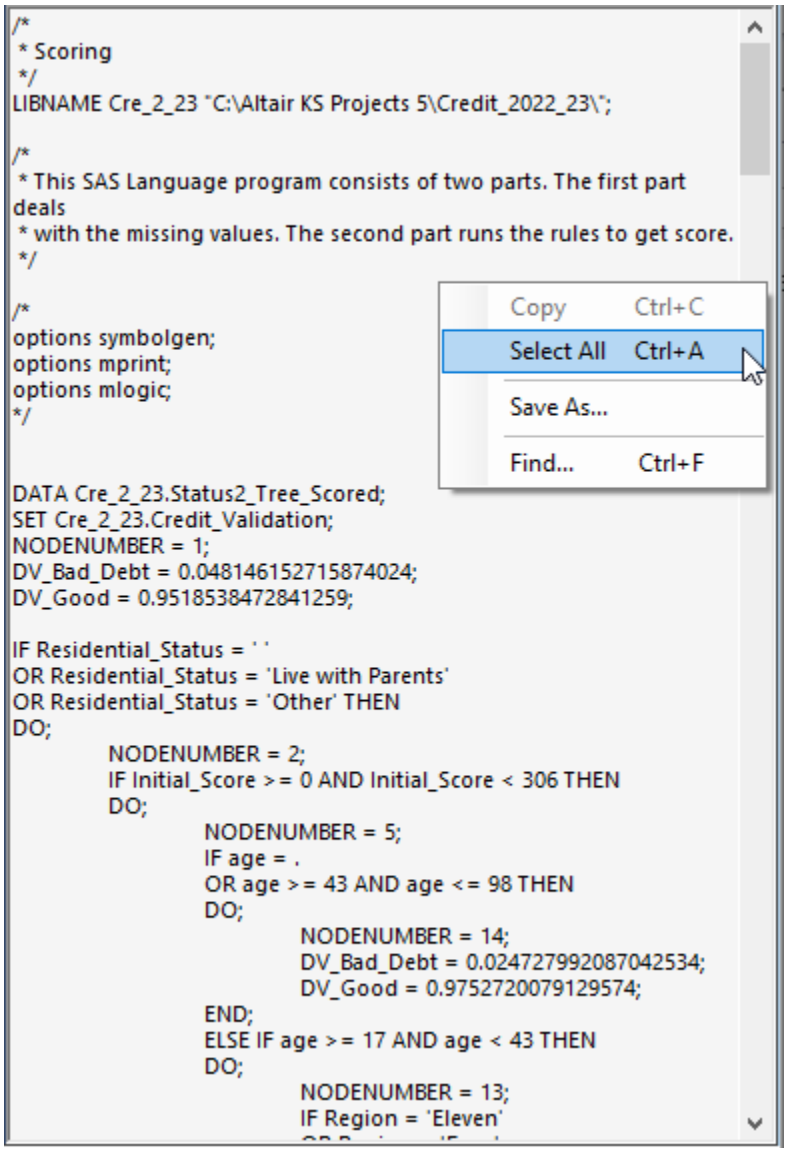


When applicable, the context menu also contains the command **SAS Language Code to Here**. Selecting this command will generate SAS Language code for all nodes in the workflow path leading up to the selected node. In the following example, the code is generated for the whole workflow leading up to the Scoring node, including the scoring operation.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Right-click anywhere within the frame that displays the code and use the context menu commands to copy the code, save it in a file, or search for specific test in the code.



See also:

- Code Generation for Market Basket Analysis
- Insert Model from XML/PMML

Upload Model to SmartWorks Analytics

Altair® SmartWorks™ Analytics is an end-to-end platform that enables you to combine enterprise expertise with data and artificial intelligence (AI) to build analytics applications and scalable automation – with or without coding.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If you use SmartWorks Analytics at your organization and have a user account and execution profile for it, you can upload models to SmartWorks Analytics directly from the project workflow.

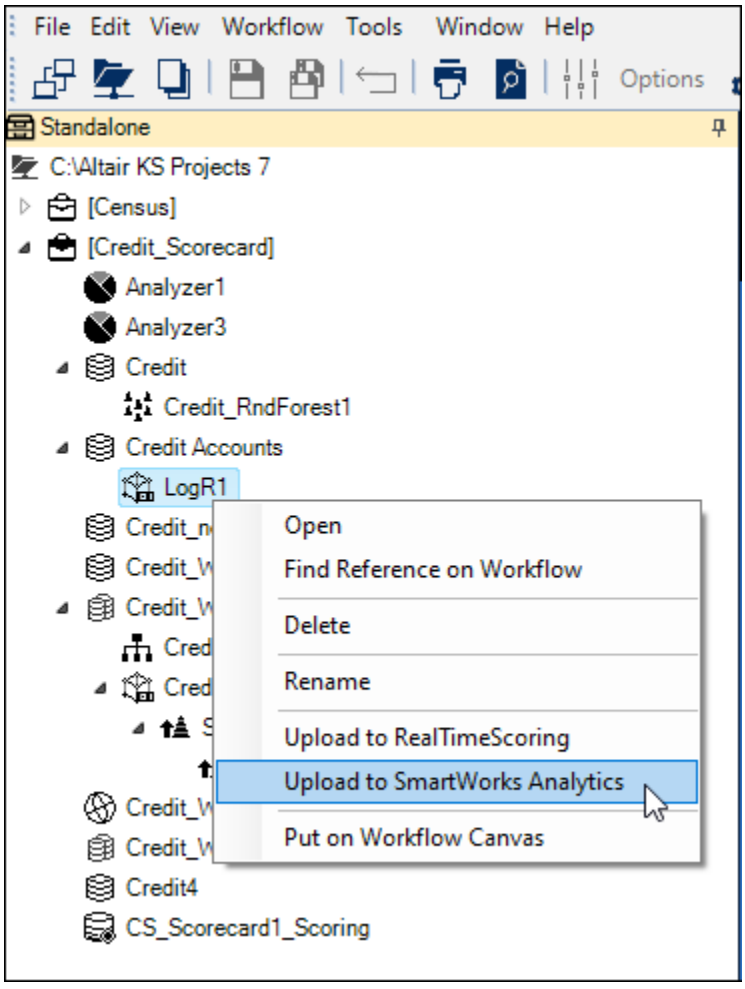
All native Altair models except Market Basket Analysis and PLS Regression Model can be uploaded to SmartWorks Analytics. For the open-source models implemented in Python (such as GLM or XGB models), this functionality will be implemented in one of the next versions of Knowledge Studio.

Prerequisites

The properties of the execution profile of your SmartWorks Analytics account must be specified in the configuration file `Angoss.KSMain.exe.config` in `C:\Program Files\Altair Data Analytics\KS Workstation 2022.3.0\bin`. Please refer to the SmartWorks Analytics documentation or contact a SmartWorks Analytics administrator for the required information.

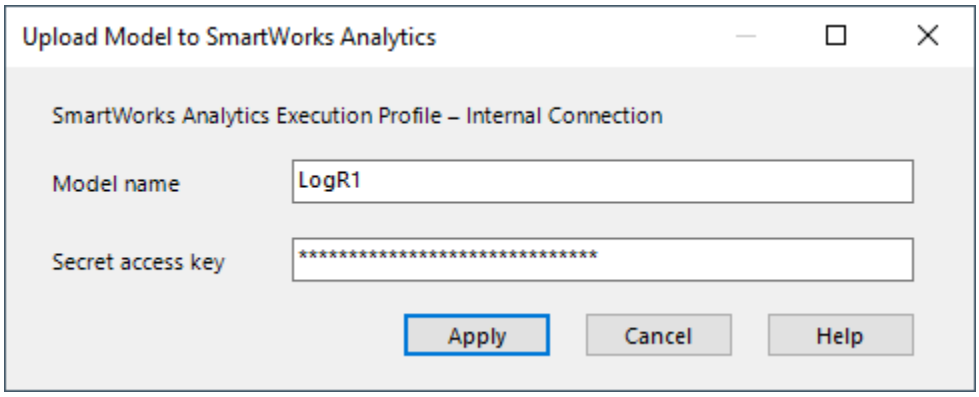
Uploading Models

To upload a model to SmartWorks Analytics, open Project Explorer and right-click on the model you wish to upload. From the context menu, select the command **Upload to SmartWorks Analytics**.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

In the upload dialog, enter the model name as you want it to appear in the SmartWorks Analytics Library and enter the secret access key for your account.



Click **Apply** to upload the model.

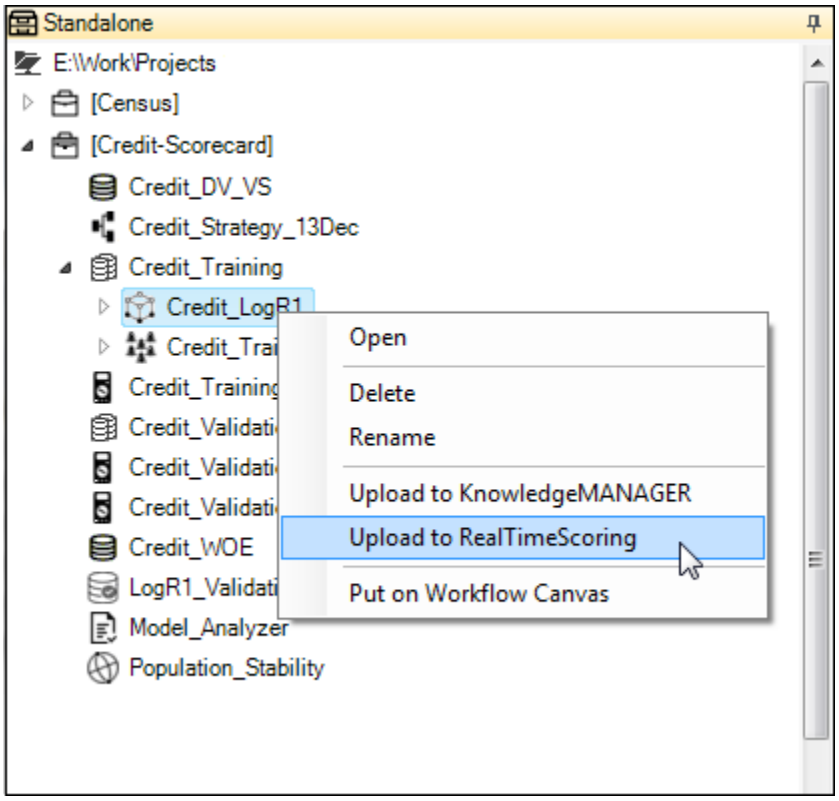
The uploaded model will appear in the SmartWorks Analytics library defined by the execution profile configuration. Please refer to the SmartWorks Analytics documentation for further details.

Upload Model to Real-Time Scoring Service

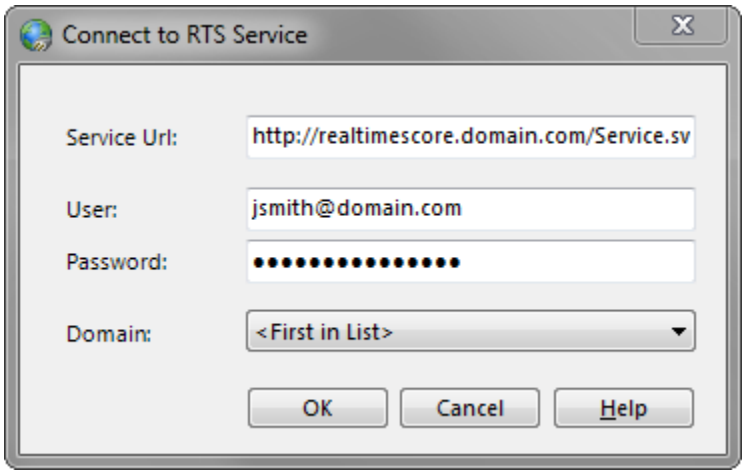
Altair Real-Time Scoring Engine integrates with operational systems to provide the ability to deploy models and strategies in real-time and produce on-demand scores, decisions, or recommendations as well as apply real-time actions or treatments.

If you use the Real-Time Scoring Engine at your organization and have a user account for it, you can upload models to the Real-Time Scoring Server directly from the project workflow. To upload a model, open Project Explorer and right-click on the model you wish to upload. From the context menu, select the command **Upload to RealTimeScoring**.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



This will bring up the dialog for connecting to the Real Time Scoring (RTS) Service. Enter the Service URL and specify your credentials (your user name and password for the RTS Server), select the domain and click OK.



Please refer to the Real-Time Scoring Engine documentation for further details.

If you do not currently use this product and would like to learn more, please contact Customer Support.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Optimization

Simple Budget Optimization

Linear Optimization on the dataset level handles user-defined linear objective functions with linear constraints. The decision variables to optimize can be continuous or discrete. The range of optimization problems covered are those on the level of aggregating the values over a given dataset. The objective function parameters and the constraints are composed using the values specified by the user in the optimization wizard. The decision variables are defined to be new variables in each record.

For this type of optimization, Knowledge Studio uses the open-source linear programming solver COIN-OR LP based on the revised Simplex method.

The optimizer is capable of solving optimization problems that arise in credit risk management and in marketing/CRM applications. Special numerical algorithms are implemented in order to handle large problems while avoiding numerical problems. Problems that involve objective functions and constraints resulting in unbounded or non-existent solution are automatically diagnosed and the user is presented with appropriate warnings.

The optimization engine (as an out-of-the-box feature, without custom programming) cannot handle multi-objective and hierarchical optimization problems. However, using the R Integration node in the workflow these problems can be solved using the appropriate R packages with some programming in R.

The input for the optimization operation is a dataset that contains the **probabilities of the outcomes that drive your decisions (treatments) to be optimized**.

> *For example, suppose you plan a marketing campaign based on the past campaign response models. You would like to contact customers using several channels, and you want to find optimal channel for each customer, subject to constraints (cost, total budget, etc.). Then the input should be a dataset that contains the predictions of response models for each channel.*

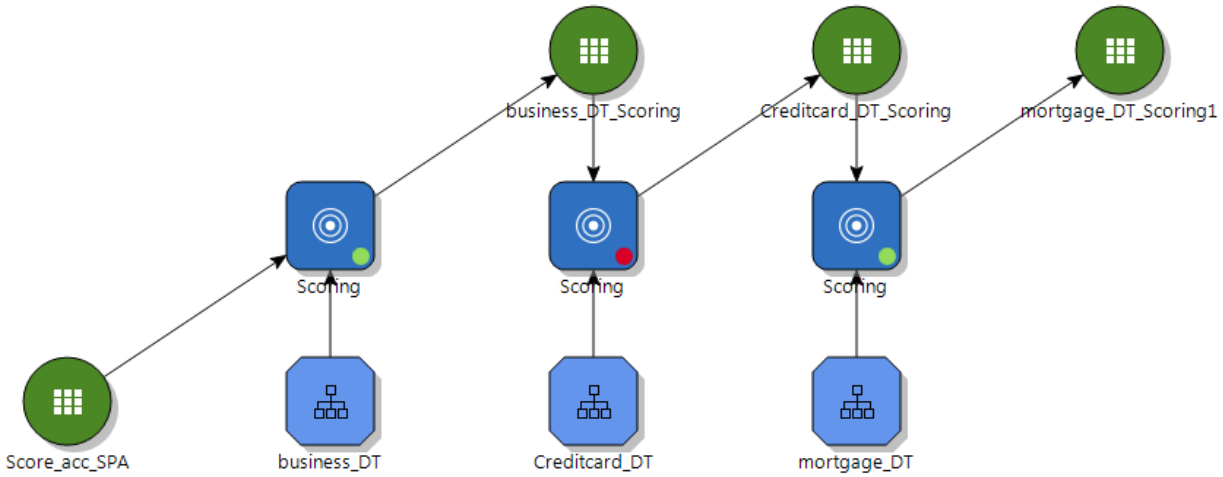
This dataset can be the result of scoring Knowledge Studio models or any other models. In the case of model scores produced in other analytics environments (SAS System, SPSS, etc.), the datasets can be imported from external sources (Text, Excel, SAS System, SPSS, R data files or database tables).

To add an optimization operation in your workflow:

Step 1. Prepare the input dataset. As noted above, the input dataset should contain the probabilities of the outcomes that drive the treatments you need to optimize. If these probabilities are in external files or tables, import them using import nodes from the **Connect** palette.

Example: Three models of customer credit risk are scored. Each scoring output is an input for the next model scoring, so the final dataset contains the scores of all three models. It will serve as an input for the Optimization operation.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



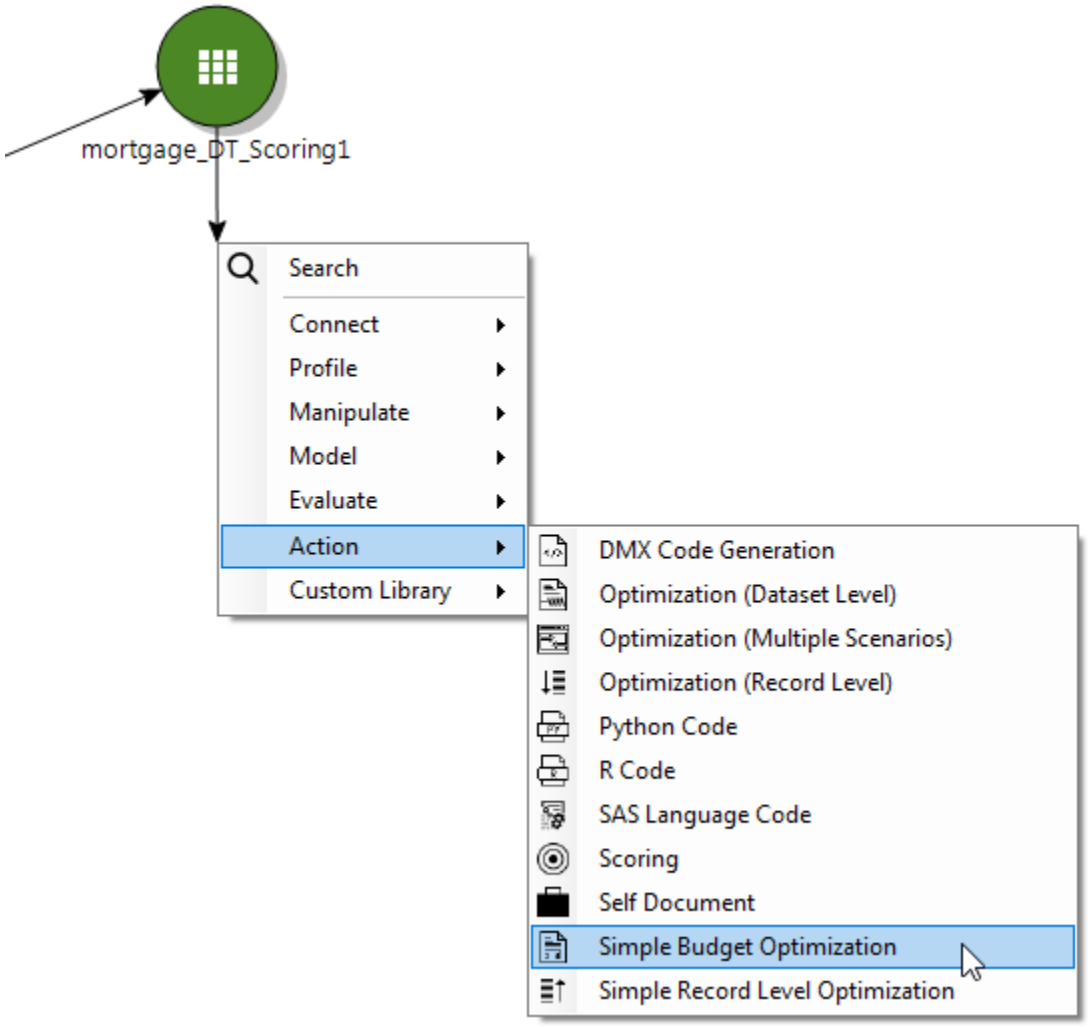
Step 2. Perform the optimization:

- Drag a connection arrow from the input dataset node to any area on the Workflow canvas background. This brings up a context menu that allows you to select the desired operation. From the context menu, select **Action --> Simple Budget Optimization** as in the example screenshot below. A new **Simple Budget Optimization** process node will be created.

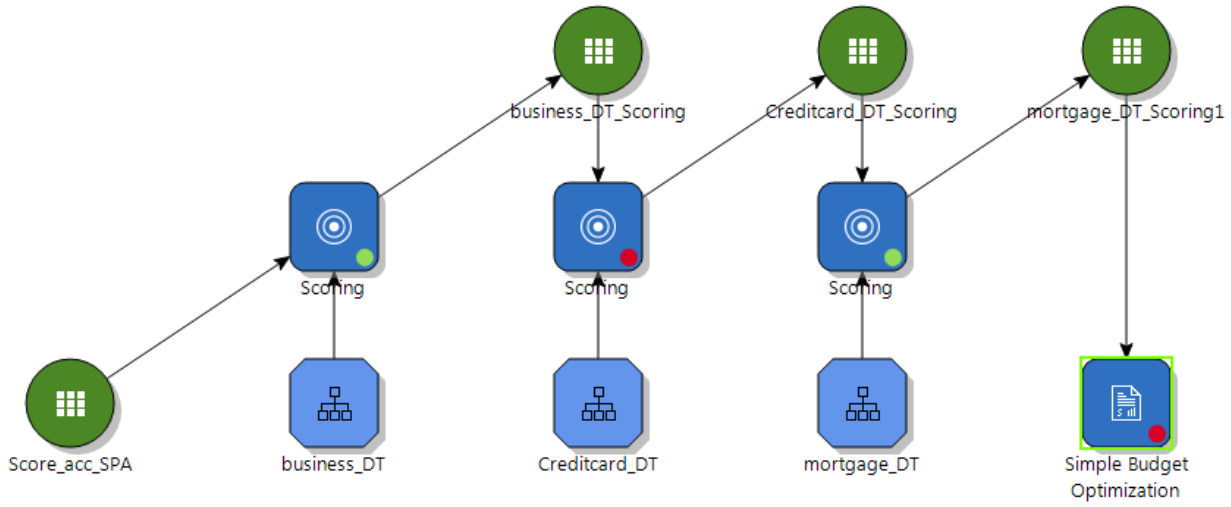
OR:

- Drag the **Simple Budget Optimization** node from the **Action** palette onto the canvas and connect the input dataset to the node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



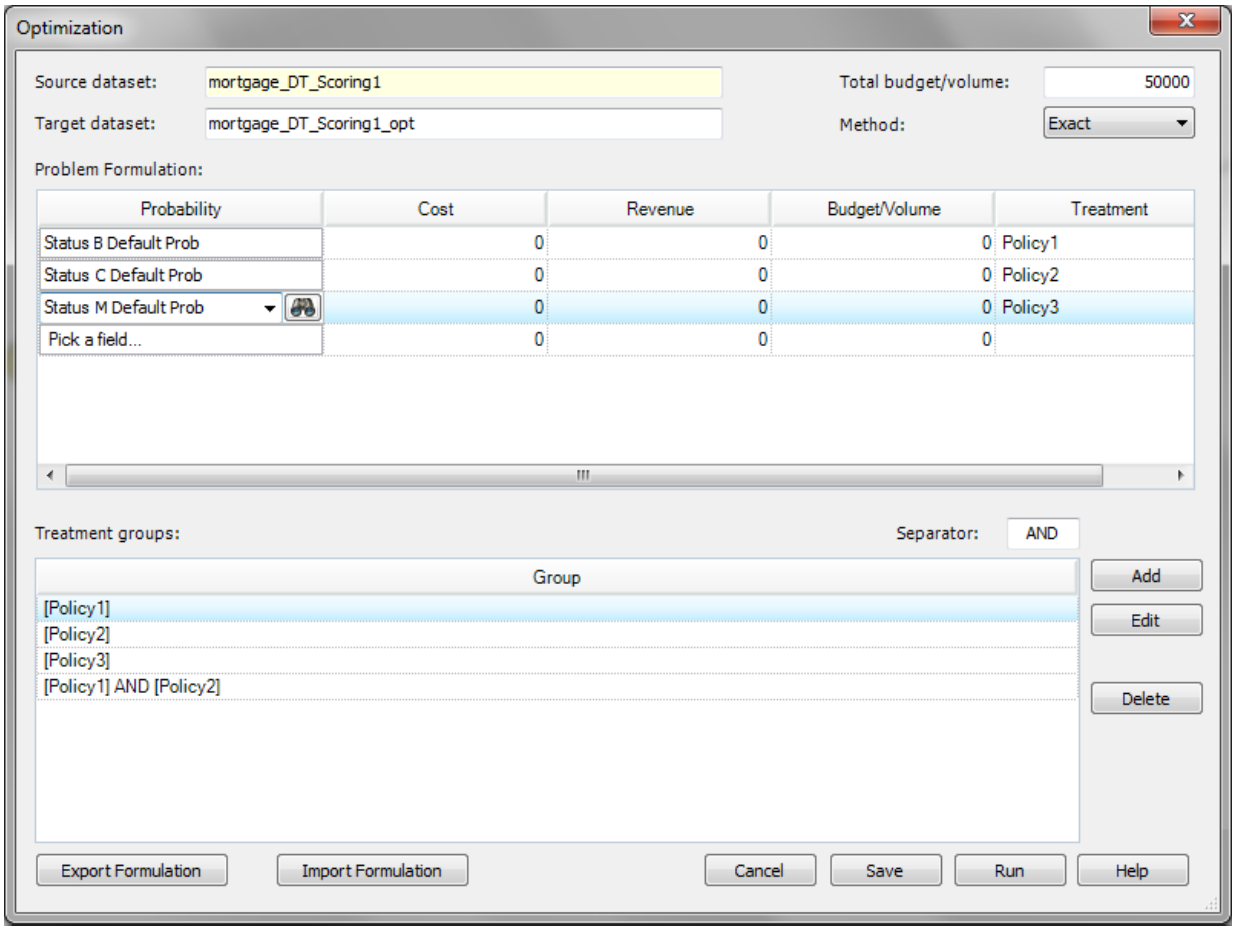
Unless you used the context menu, double-click the **Simple Budget Optimization** node or right-click on it and select **Modify** to open the wizard.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

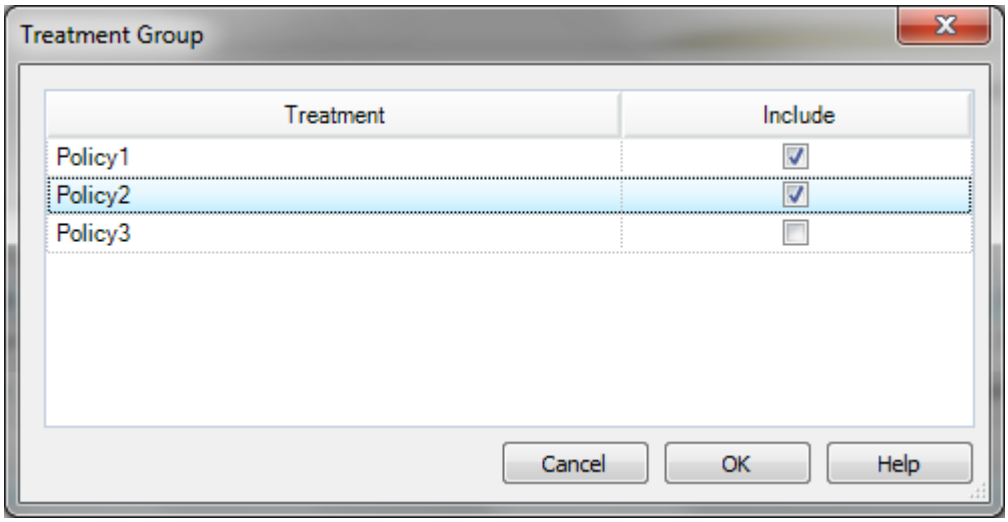
Specify the options and parameters for the operation:

- Enter the name for the output (target) dataset that will contain optimized treatments
- Enter the total budget amount or total volume.
- Specify the method for the optimization algorithm (*Exact*, *Greedy*, or *Approximate*)
- Specify the fields expressing the probability of the outcomes that drive the treatments to optimize. To add a new field, click on the new row in the ***Field*** column. To delete a probability field, select the field and press the Delete key.
- Specify the treatment in the Treatment column. These are the treatments you need to optimize.
- Specify the cost, revenue, and budget (or volume) for each treatment specified.
- Click **Add** to specify the treatments or treatment groups (combinations) to assign. If you define any treatment group, specify the desired separator. The default separator is the multiplication symbol (*).



Example: adding the treatment group "Policy1" AND "Policy2"

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If you need to copy the optimization problem formulation to an optimization node within the same workflow canvas, or to a node in another workflow or project, or to send it to a colleague, see the topic Copy, Export, and Import Problem Formulations and Variables.

Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the operation, double-click the output dataset to view the results.

Simple Record-Level Optimization

The Simple Record-Level Optimization is a solver that supports linear and nonlinear objective functions with linear constraints. It assigns treatments (values of decision variables) to every record in the input dataset. As opposed to the general Record-Level Optimization node, this node provides an easier and way to define linear constraints.

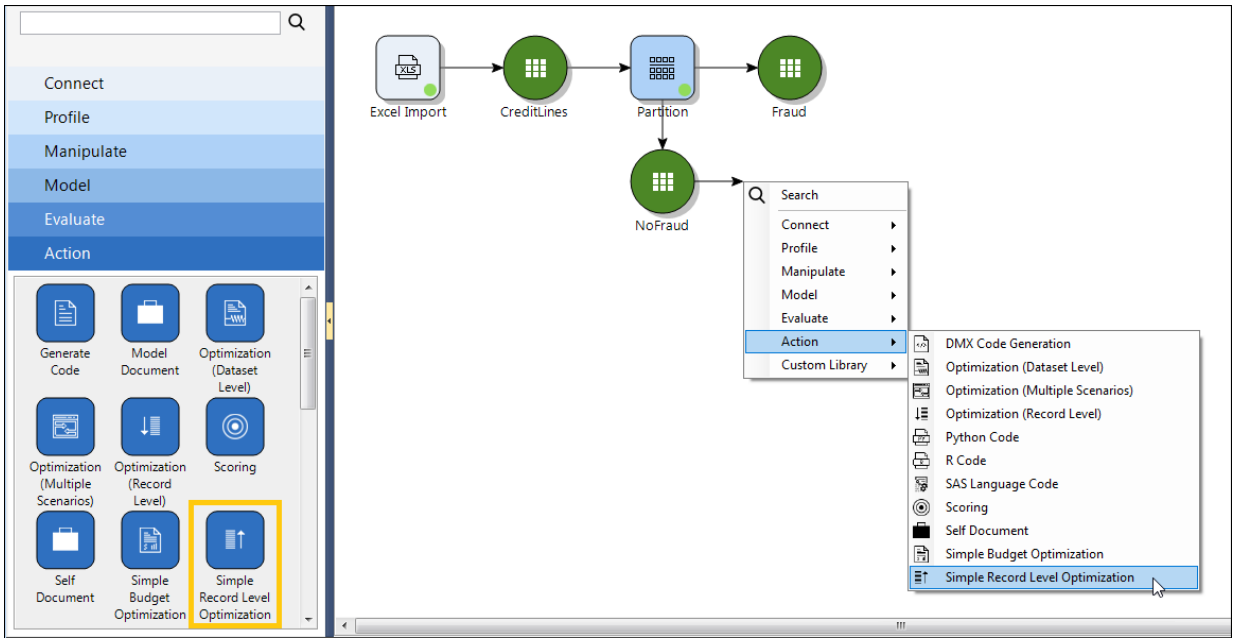
To add a simple record-level optimization operation to the workflow:

- Drag a connection arrow from the input dataset node to any area on the Workflow canvas background. This brings up a context menu that allows you to select the desired operation. From the context menu, select **Action --> Simple Record-Level Optimization** as in the example screenshot below. A new **Simple Record-Level Optimization** process node will be created.

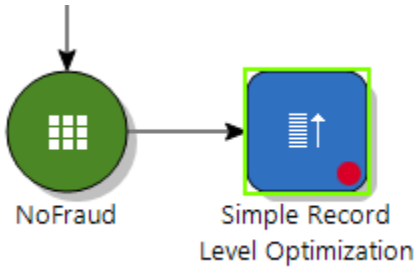
  OR:

- Drag the **Simple Record-Level Optimization** node from the **Action** palette onto the canvas and connect the input dataset to the node.

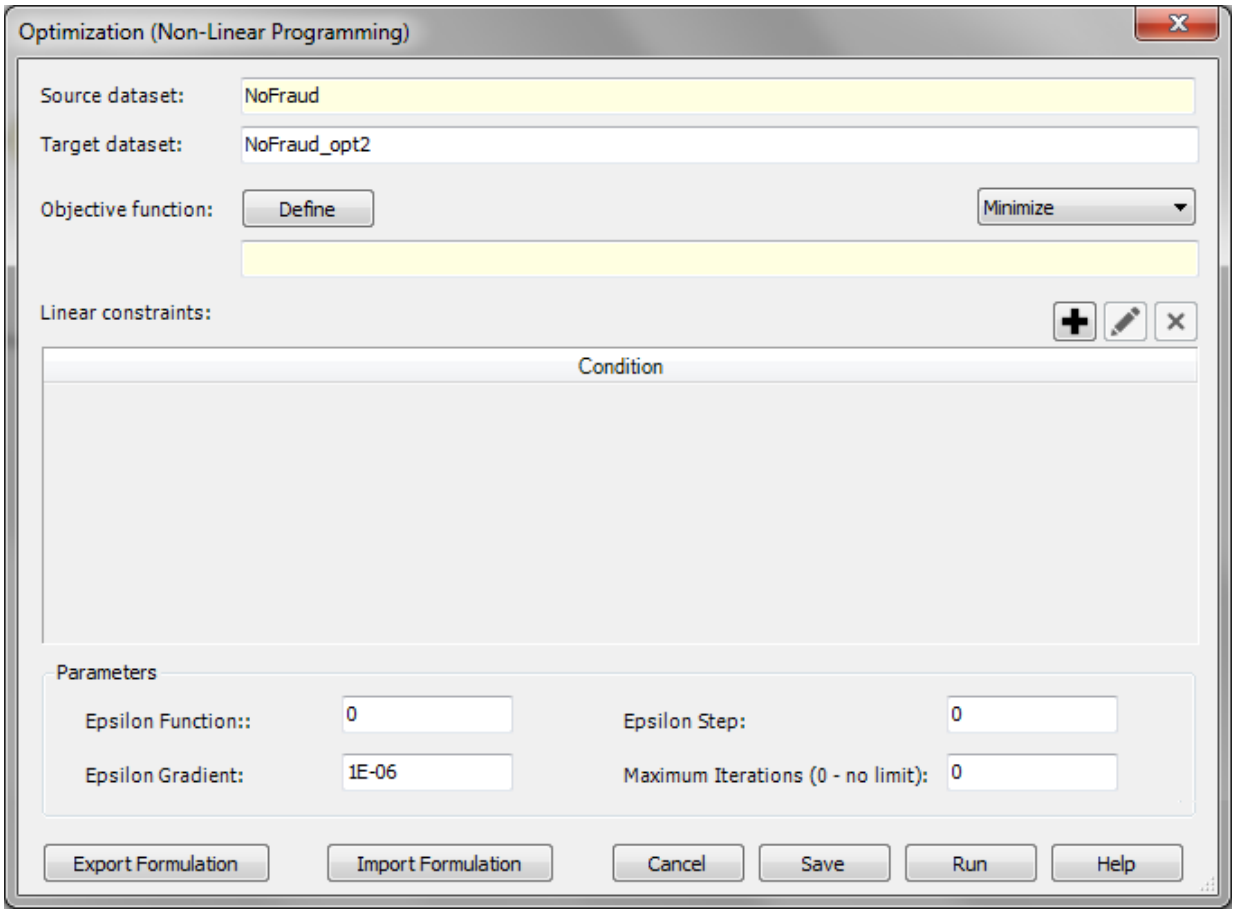
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If the wizard does not open automatically, double-click the **Simple Record-Level Optimization** node or right-click on it and select **Modify** to open the wizard.



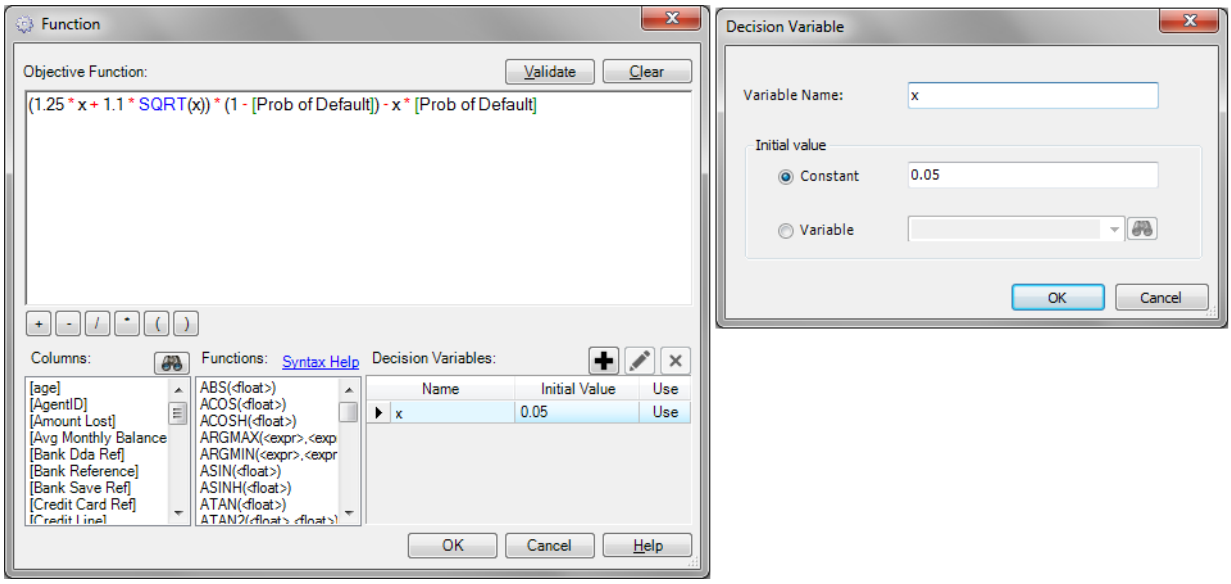
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Specify the options and parameters for the operation:

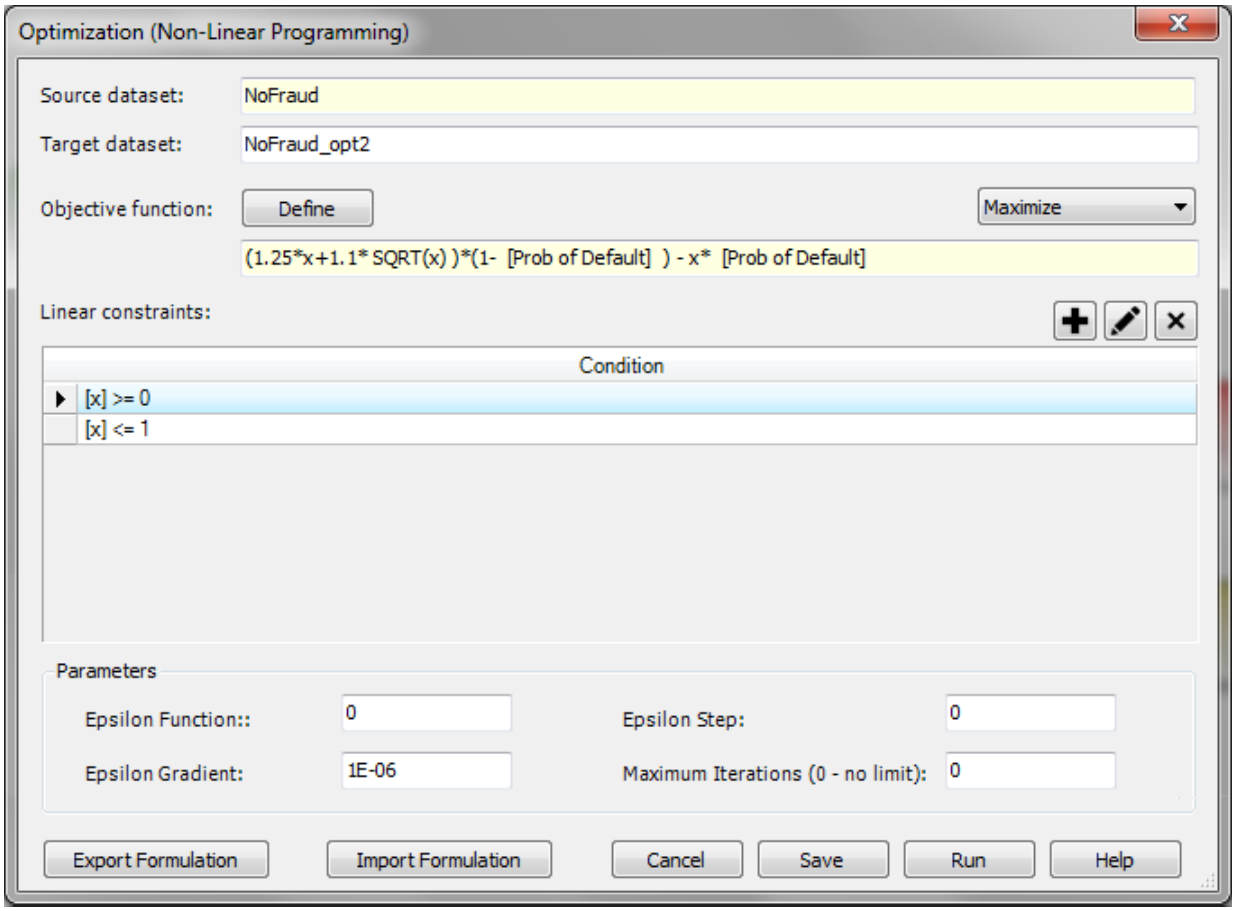
- Enter the name for the output (target) dataset that will contain optimized treatments
- Choose whether to minimize or maximize the objective function
- Click the **Define** button to define the decision variables and the objective function
- In the Function dialog, before entering your expression for the objective function, add decision variables by clicking the Add button .

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



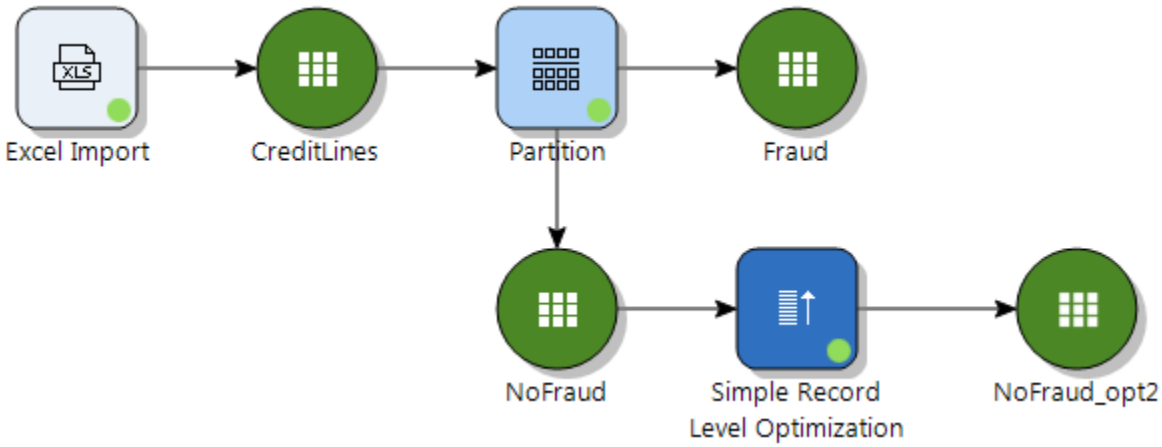
- Enter the expression for the objective function (a formula expressing the quantity to maximize or minimize in terms of the decision variables and other variables or their functions). You can use aggregate functions in the Functions list, choose variables form the Columns list, or enter a custom expression.
- To modify the name or initial value of a decision variable, click in the cell and edit the name or value directly within the cell. Alternatively, click in the selector column on the left side of the desired decision variable to select the whole row, and then click the Edit button .
- To delete a decision variable, click in the selector column on the left side of the desired decision variable to select the whole row, and then click the Delete button .
- To use an existing decision variable in the expression, either type its name or click the Use button in the rightmost column ("Use") of the Decision Variables list. Clicking "Use" will cause the variable to appear in the expression box at the cursor position.
- Click Validate to check the expression syntax if necessary.
- Click OK. This brings you back to the main dialog, where you can add constraints and specify parameters.
- To add linear constraints, click the Add button above the Condition list. To edit or delete an existing constraint, select it and click the Edit button or the Delete button .

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If you need to copy the optimization problem formulation to an optimization node within the same workflow canvas, or to a node in another workflow or project, or to send it to a colleague, see the topic Copy, Export, and Import Problem Formulations and Variables.

Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the operation, double-click the output dataset to view the results.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Record-Level Optimization

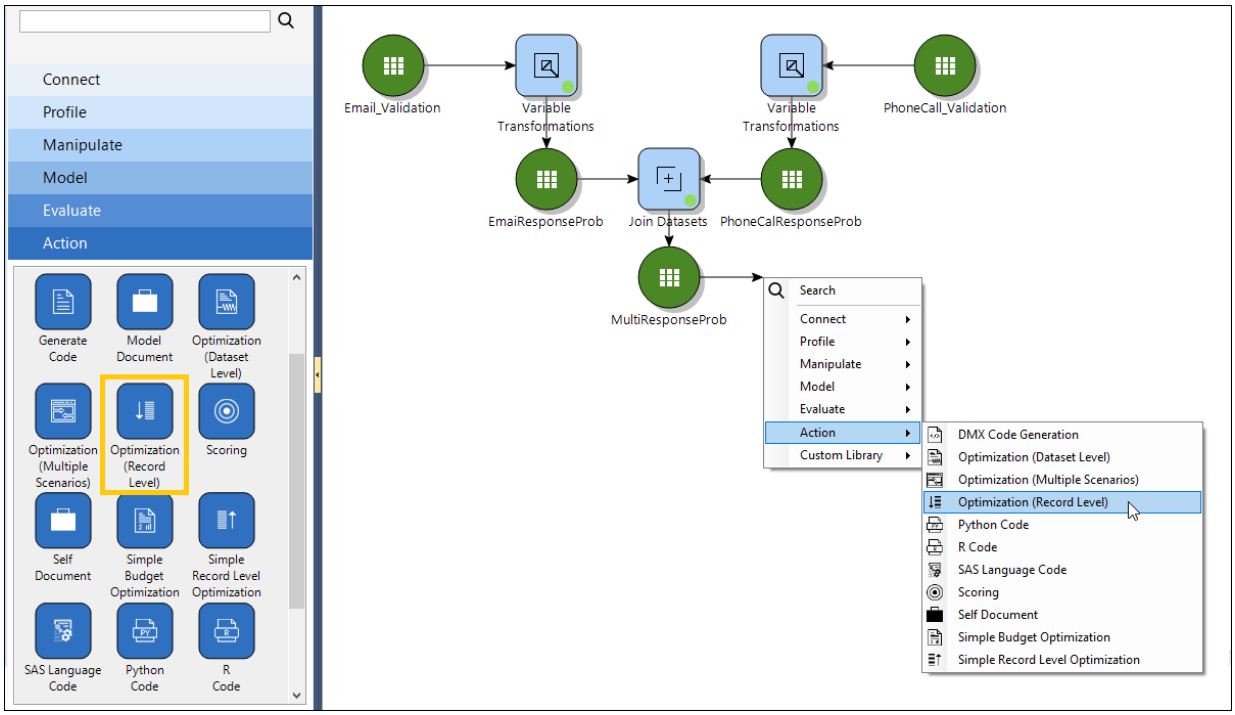
The Record-Level Optimization node in the Action palette provides a solver that supports both linear and nonlinear objective functions and constraints. Decision variables can be of any type (categorical, continuous, or integer). The optimizer assigns treatments (values of the decision variables) to every record in the input dataset. The node wizard allows specifying constraints in the form of arbitrary expressions.

To add a general record-level optimization operation to the workflow:

- Drag a connection arrow from the input dataset node to any area on the Workflow canvas background. Releasing the mouse button brings up the context menu. Select **Action --> Optimization (Record Level)** as shown below. A new **Optimization (Record Level)** node will be created, and the wizard will open.

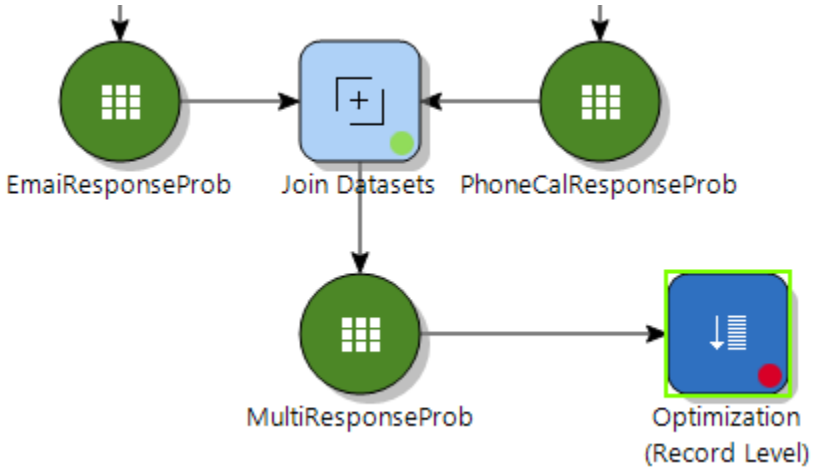
  OR:

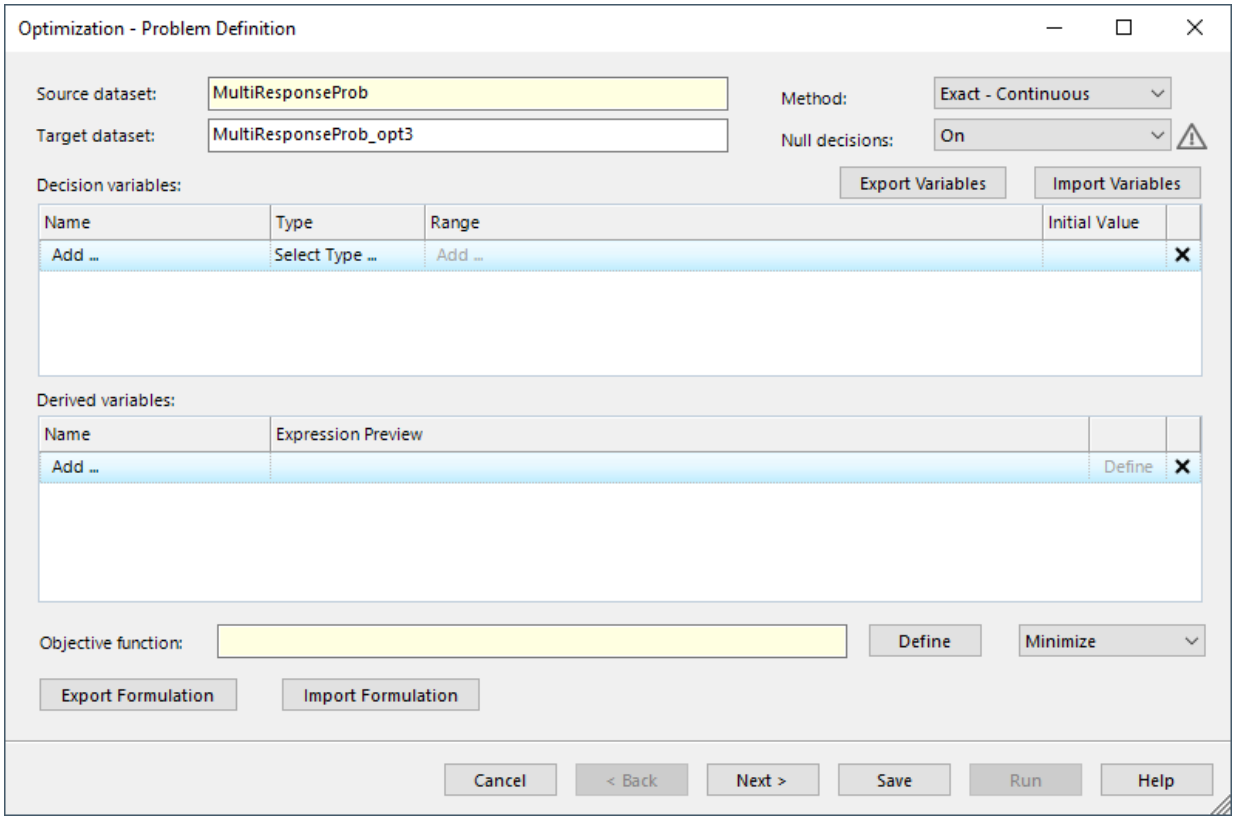
- Drag the **Optimization (Record Level)** node from the **Action** palette onto the canvas and connect the input dataset to the Optimization node.



Once the Optimization node is on the canvas, double-click or right-click on it and select **Modify** to open the wizard.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

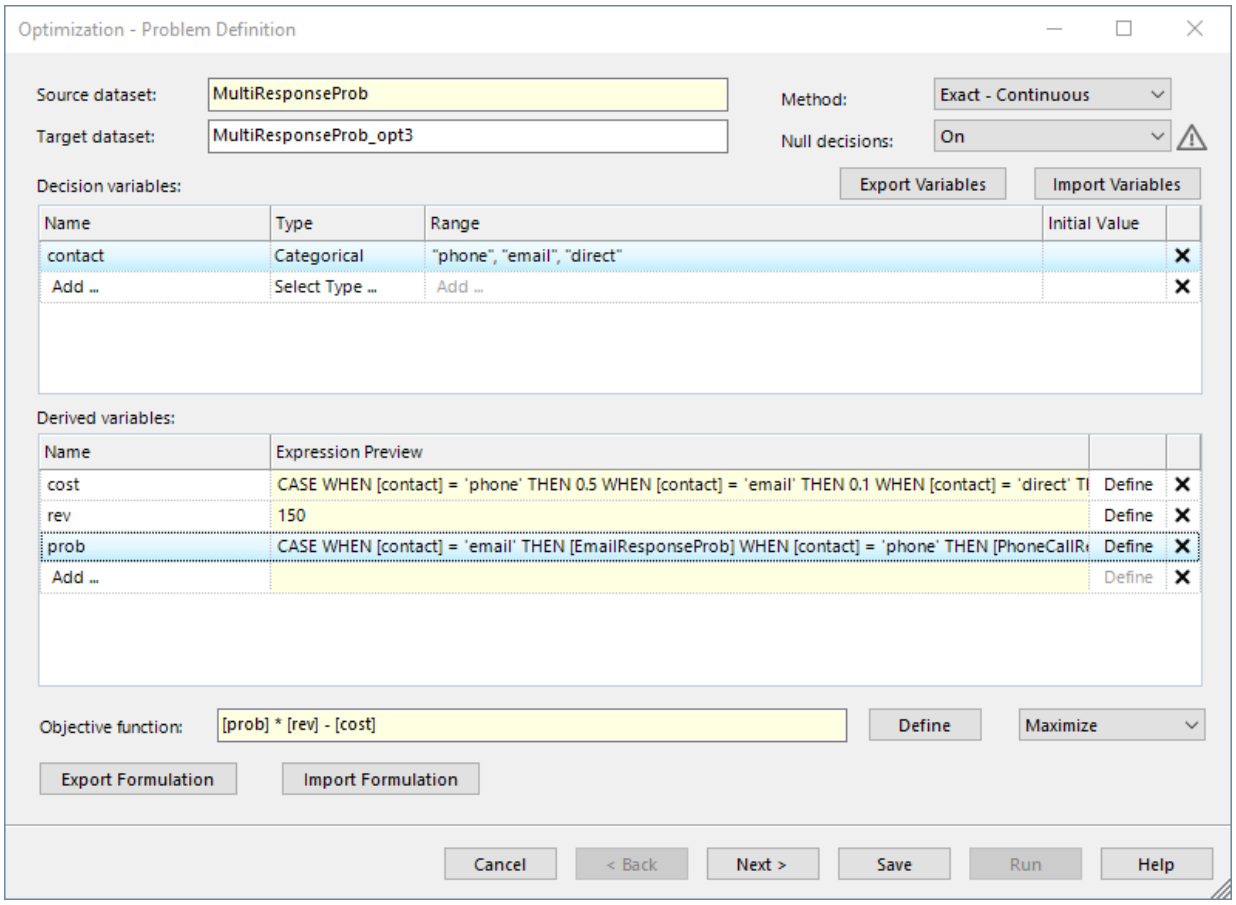




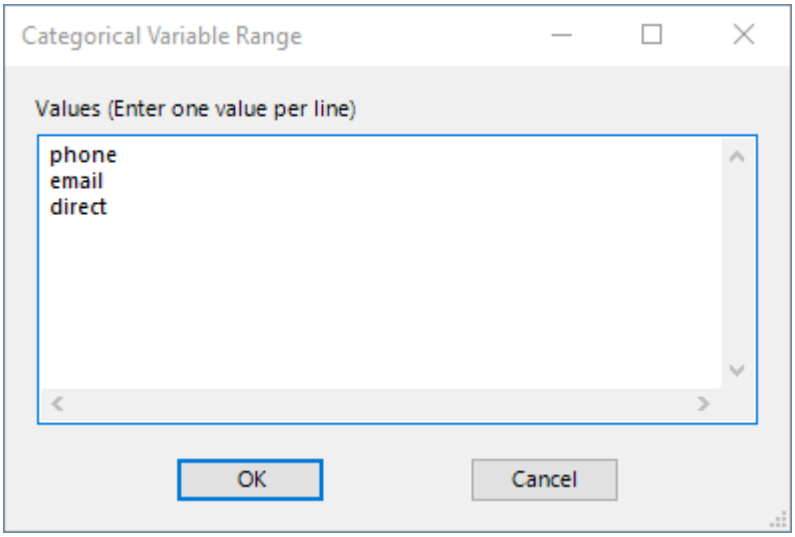
Specify the options and parameters for the operation:

- Enter the name for the output (target) dataset that will contain optimized treatments
- Specify the method for the optimization algorithm: *Exact - Continuous* or *Exact - Discrete*
- Specify whether to produce null values for decisions (solutions). Note: Turning off the option for null decisions greatly increases the possibility of an infeasible problem (a problem which has no solution). The optimizer will automatically recognize and give warnings about infeasible problems, however, it may take a long time to detect infeasibility and return an error. We recommend that only experienced users turn off null decisions.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- Define the ***decision variables*** - the ones whose values are optimized to maximize or minimize the objective function. Enter the decision variable names and specify their **type** (*Categorical*, *Continuous*, or *Integer*) and **range**. When specifying the range for categorical ones, enter their possible values - one value per line, as in this example:

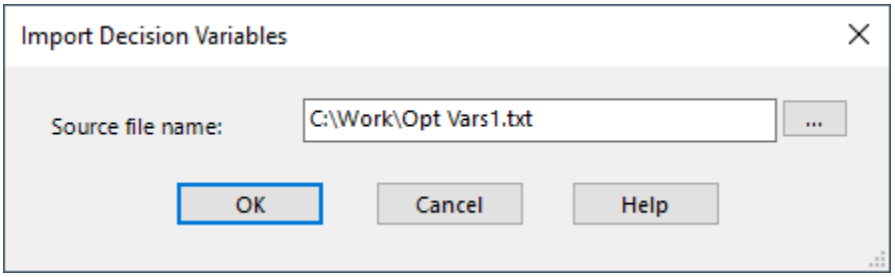


Alternatively, you can import the definition of decision variables from a file. This must be a text file with one line per variable, with the following syntax (note the double quotes):

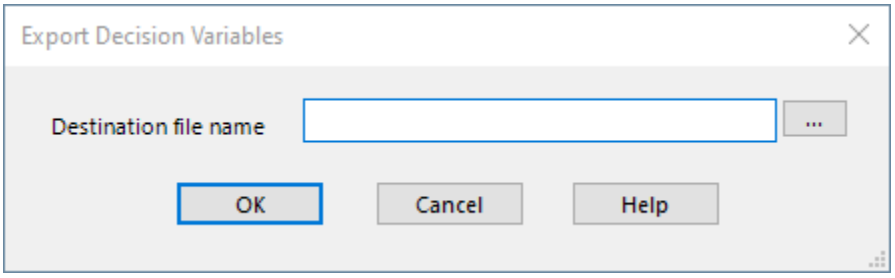
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
"<variable_name>", "<variable_type>", "<value1>", "<value2>",
....  "<valueN>"
```

To import the definition from a file, click the **Import Variables** button, browse for the source file and click OK.

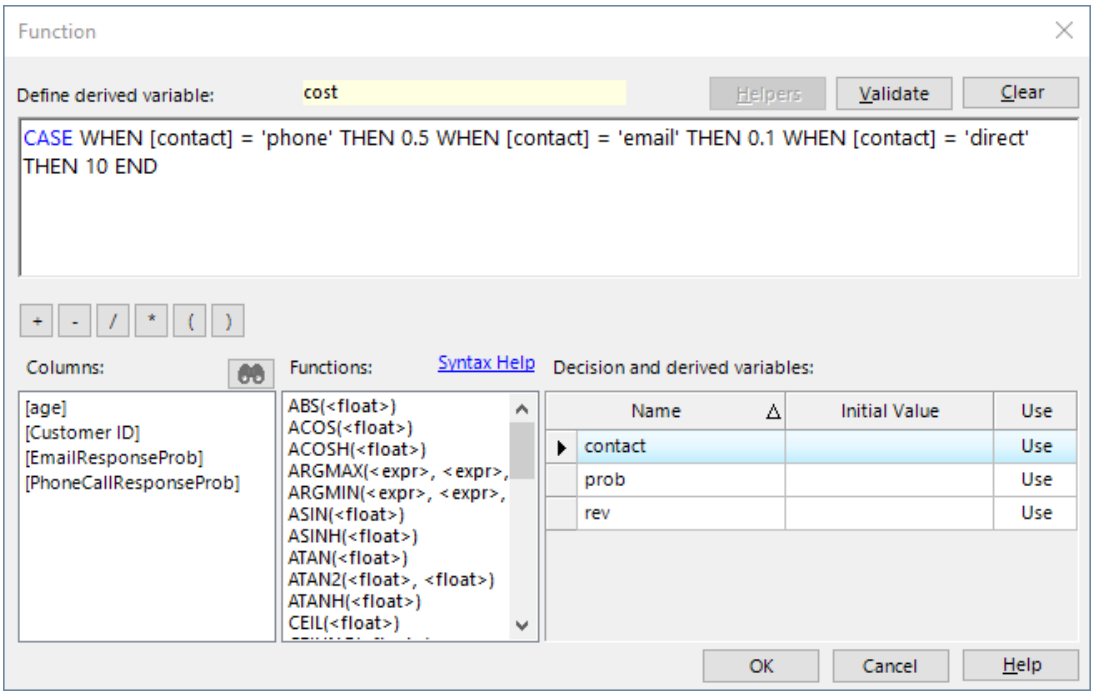


Once finished definining the decision variables, you can export the definitions for future use, if necessary. Click **Export Variables**, browse for the desired destination folder, and specify the file name.

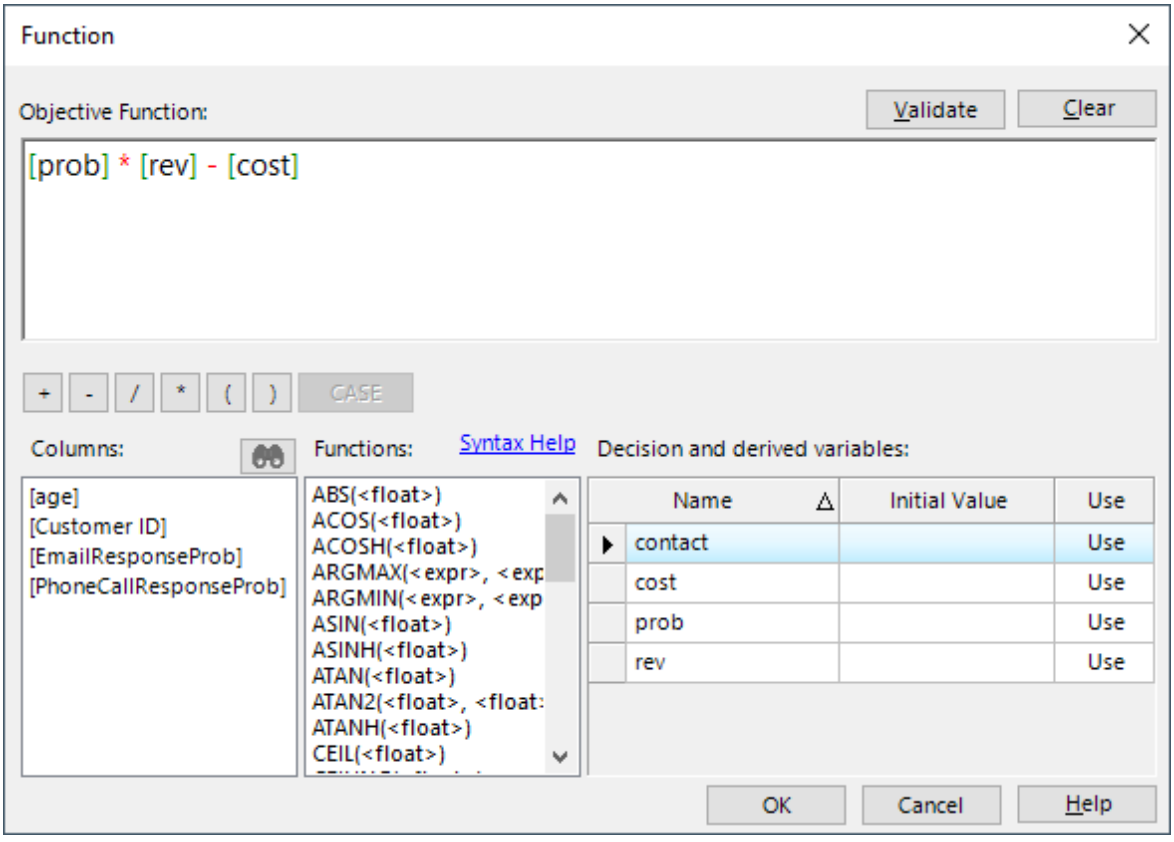


- Add **Derived variables** in the lower frame if necessary. These are new variables that depend on the original variables in the input dataset and/or on the decision variables. Adding derived variables helps to simplify the representation of the objective function.
    - Enter the names for the derived variables. Click the **Define** button on the right to enter the expression for each derived variable in the Expression Editor window. An example is shown below. The interface of the Expression Editor is similar to that in the Variable Transformations node, except the additional list of *Decision and derived variables* on the right. To use the name of a decision or derived variable in the expression, select it and click **Use** in the rightmost column. You can also type it in square brackets.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



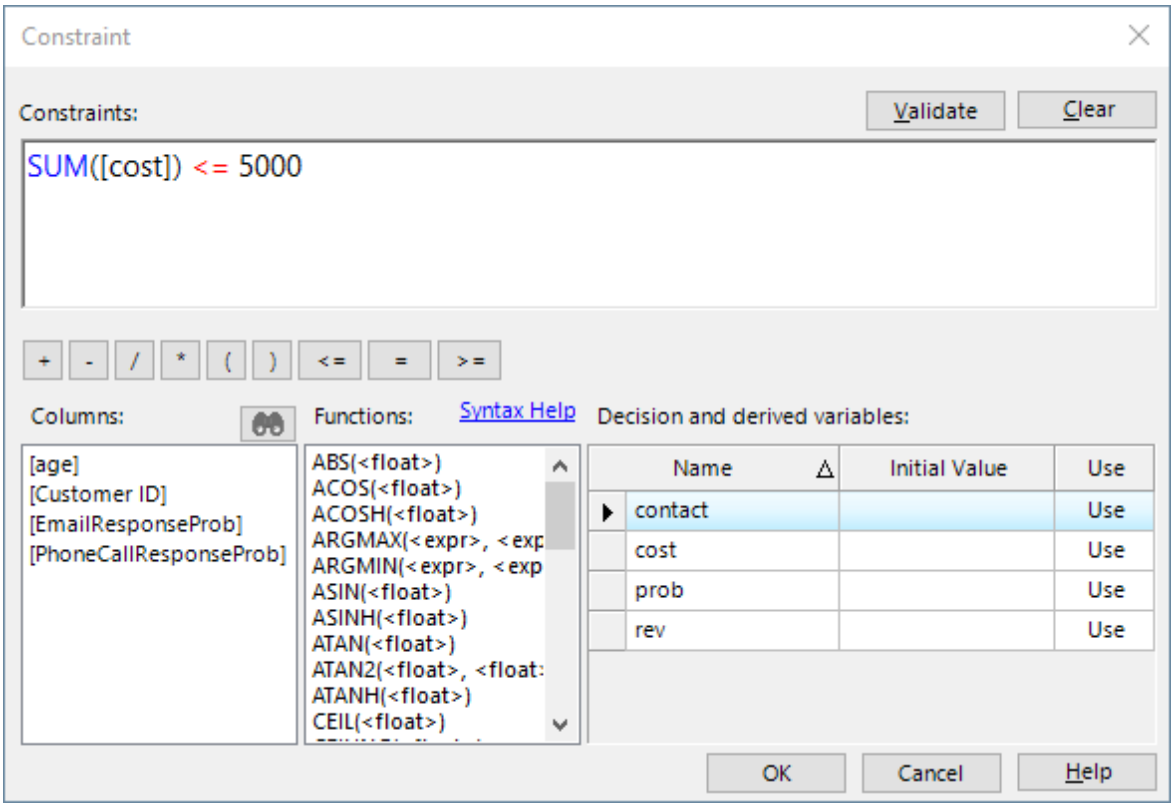
- Upon defining all necessary variables, define the **objective function** in terms of these variables. Click the **Define** button to the right of the Objective function field at the bottom. This opens the Expression Editor, which has the same features as that for Derived Variables.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

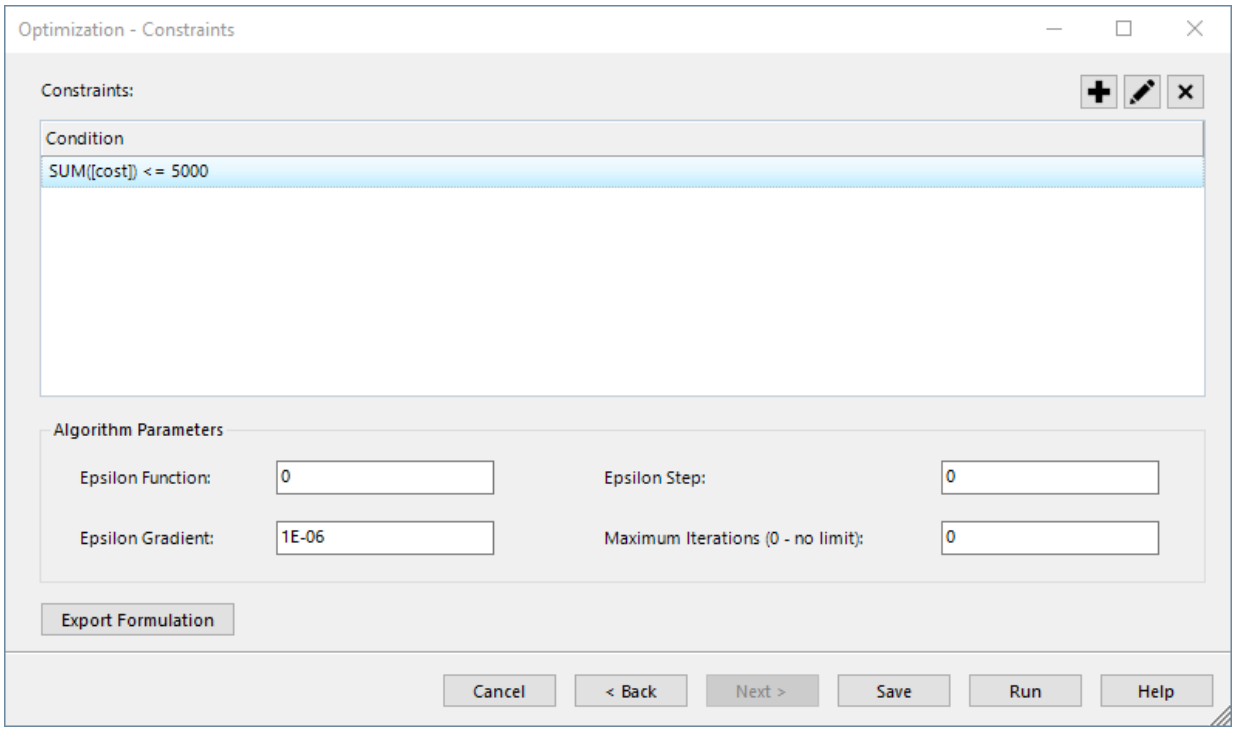
Once finished specifying the decision variables, derived variables, and the objective function, click **Next** to define the constraints.

- In the next page, click the plus sign (**Add new**) to add a constraint. Specify the expression for the new constraint in the Expression Editor window.



- To edit an existing constraint, select it and click the **Edit** button . To delete a constraint, select it and click the **Delete** button .
- Adjust the algorithm parameters at the bottom of the page, if necessary.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If you need to copy the optimization problem formulation to an optimization node within the same workflow canvas, or to a node in another workflow or project, or to send it to a colleague, see the topic Copy, Export, and Import Problem Formulations and Variables.

Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the operation, double-click the output dataset to view the results.

Dataset-Level Optimization

The Dataset-Level Optimization node in the Action palette provides a solver that supports both linear and nonlinear objective functions and only linear constraints (overall and per-channel). Decisions variables can be of any type (categorical, continuous, or integer). The solutions are assigned at the dataset (segment) level as opposed to record level. The node wizard allows specifying constraints in the form of arbitrary expressions.

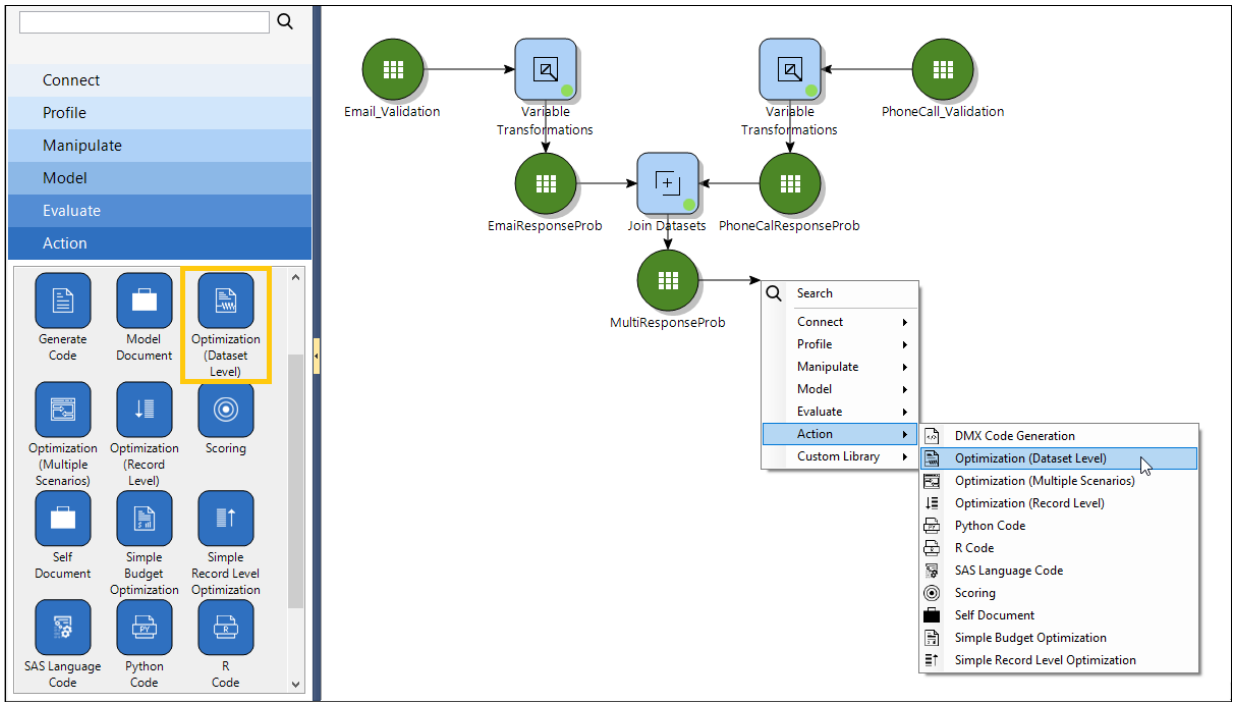
To add a general dataset-level optimization operation to the workflow:

- Drag a connection arrow from the input dataset node to any area on the Workflow canvas background. This brings up a context menu that allows you to select the desired operation. From the context menu, select **Action --> Optimization (Dataset Level)** as in the example screenshot below. A new **Optimization (Dataset Level)** process node will be created.

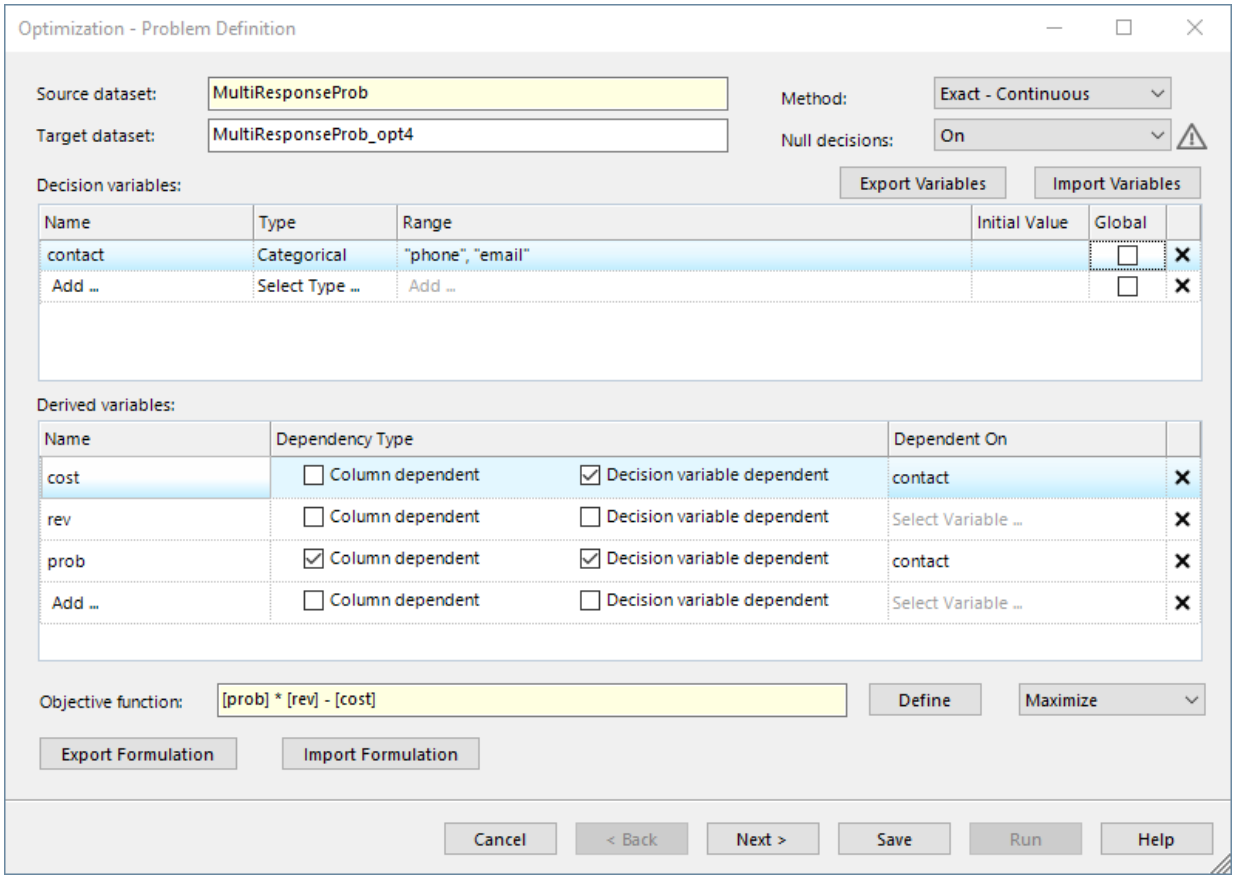
  OR:

- Drag the **Optimization (Dataset Level)** node from the **Action** palette onto the canvas and connect the input dataset to the node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



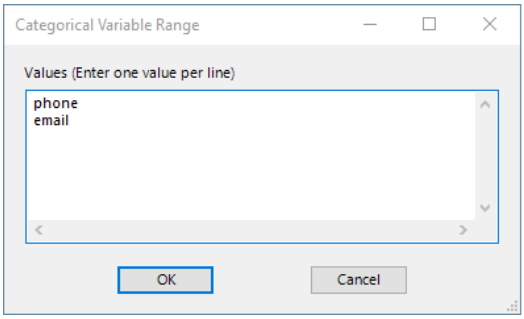
Double-click the node or right-click on it and select **Modify** to open the wizard.

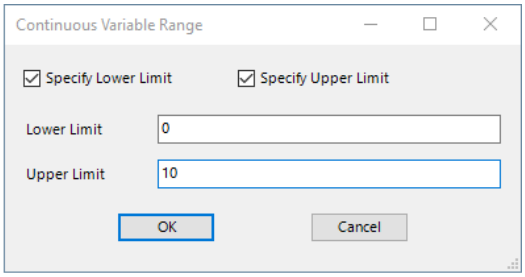


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Specify the options and parameters to define the problem:

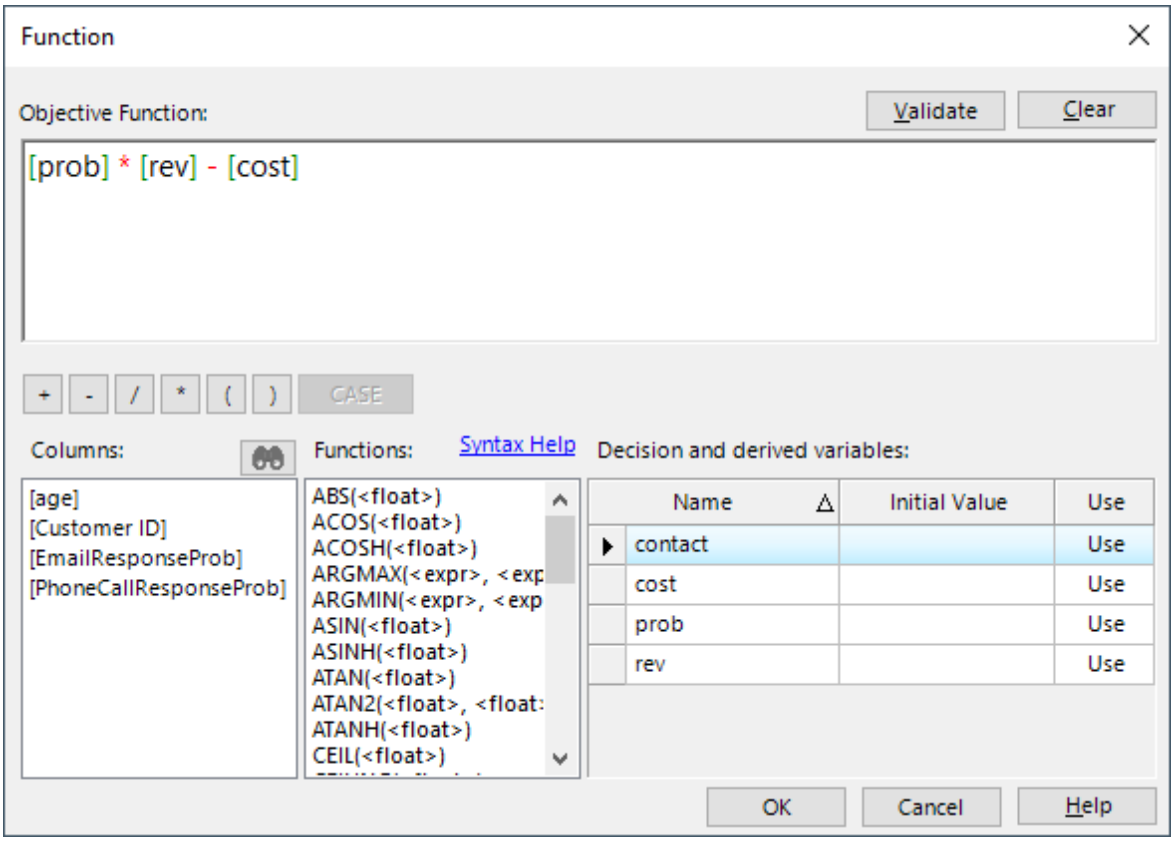
- Enter the name for the output (target) dataset that will contain optimized treatments.
- Specify the method for the optimization algorithm: *Exact - Continuous* or *Exact - Discrete*
- Choose whether to produce null values for decisions (solutions). Note: Turning off the option for null decisions greatly increases the possibility of an infeasible problem (a problem that has no solution). The optimizer will automatically recognize and give warnings about infeasible problems, however, it may take a long time to detect infeasibility and return an error. We recommend that only experienced users turn off null decisions.
- Define the *Decision variables*:
  - Specify the type: *Categorical*, *Continuous*, or *Integer*.
  - Specify the range: Click *Add* in the Range column and enter all possible values for the categorical variables - one value per line, as shown below - and specify the lower and/or upper limits for the values of the continuous variables:



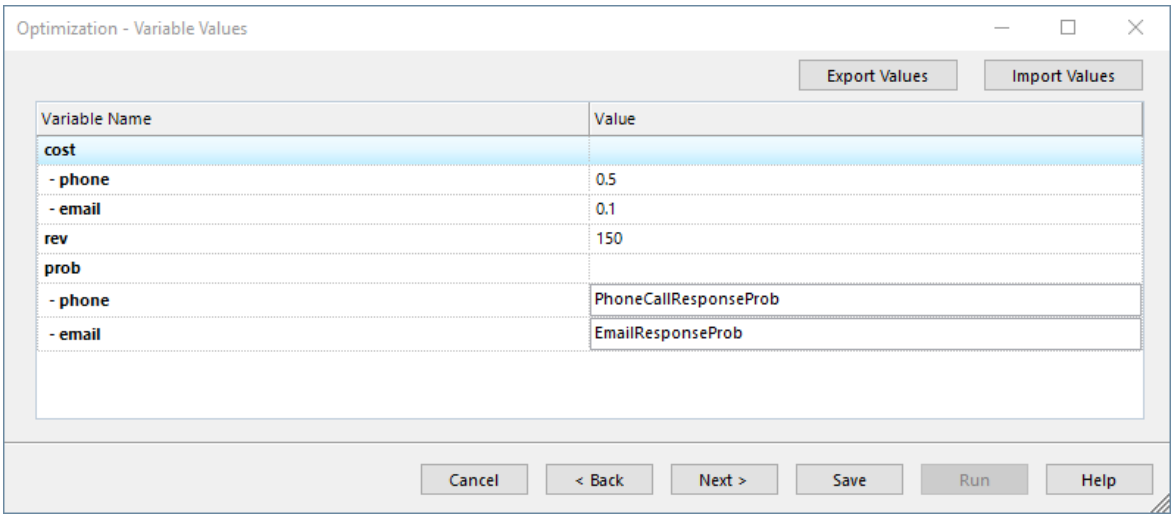


  - Specify the initial value for continuous variables in the *Initial Value* column.
  - Select the ***Global*** option if you need a single (global) decision variable to be used with the same single value across every row of the dataset. Otherwise the variable will be automatically translated to a family of decision variables - one for each row in the dataset.
- For derived variables (variables that depend on other variables/columns in the dataset and/or on the decision variables), specify their names and types of dependence. If any one depends on a decision variable, specify which one.
- Click the **Define** button to define the objective function in terms of all variables defined above. The expression editor is provided for this purpose:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

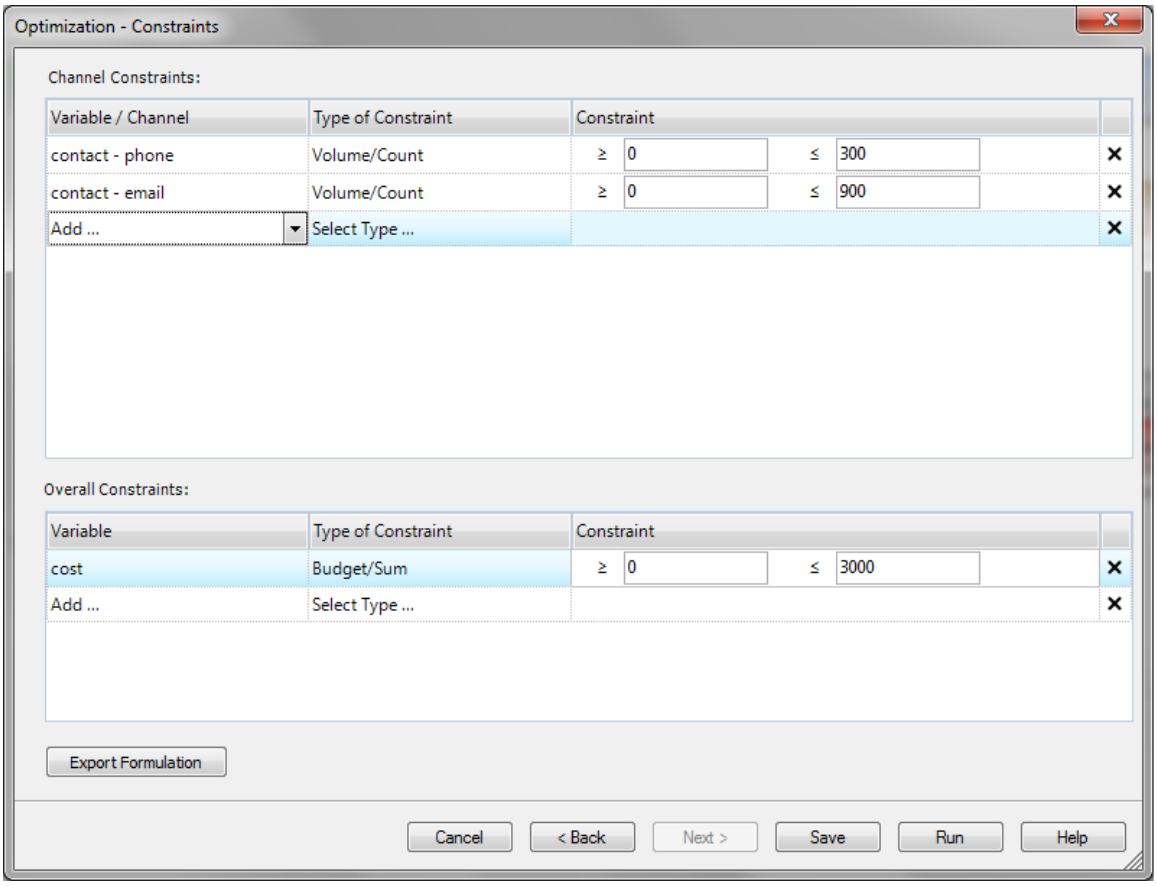


- Click **Next**. In the Variable Values page, select the probability fields for the probabilities of each discrete decision, and specify the values for the derived variables where applicable. You can import the variable values from a previously saved definition file or export them to use in another optimization node within the same workflow or a different project. See the topic Copy, Export, and Import Problem Formulations and Variables.



- Specify the constraints at the next step. This wizard page allows you to specify per-channel constraints and Overall constraints (linear only).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If you need to copy the optimization problem formulation to an optimization node within the same workflow canvas, or to a node in another workflow or project, or to send it to a colleague, see the topic Copy, Export, and Import Problem Formulations and Variables.

Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the operation, double-click the output dataset to view the results.

Multiple Scenario Optimization

The Multiple Scenario Optimization node titled "**Optimization (Multiple Scenarios)**" is located in the **Action** palette together with the other optimization nodes. It provides the capability to define multiple optimization tasks.

A task scheduler option in its wizard provides the ability to schedule the optimization task, which can run even in unattended mode when the application is closed and the user is logged out. The scheduled task properties can be modified if necessary using the Task Scheduler (on Windows) or *cron* (on Linux).

> *Note*: On Red Hat Enterprise Linux 8 and CentOS 8, task scheduling requires the **at** utility and the **atd** job scheduler daemon. To install and enable them, use the commands
>
> ```
> sudo yum install at
> ```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
sudo systemctl start atd
sudo systemctl enable atd
```

When using the scheduling option, note that the scheduled task launches a separate instance of the application executable in the background, and therefore it will perform a separate license check when it starts.

- In the case of a named-user license, if you modify the task properties in Task Scheduler or cron, make sure the task is configured to run under the account of a licensed user.
- In the case of a concurrent license, the instance launched by the scheduled task will use a new connection, so it will run only if connections are available at the scheduled start time. Since tasks are usually scheduled to run in unattended mode when the application is closed, one user ID would not normally take more than one connection. However, the concurrent connections limit should be taken into account when scheduling multiple tasks.

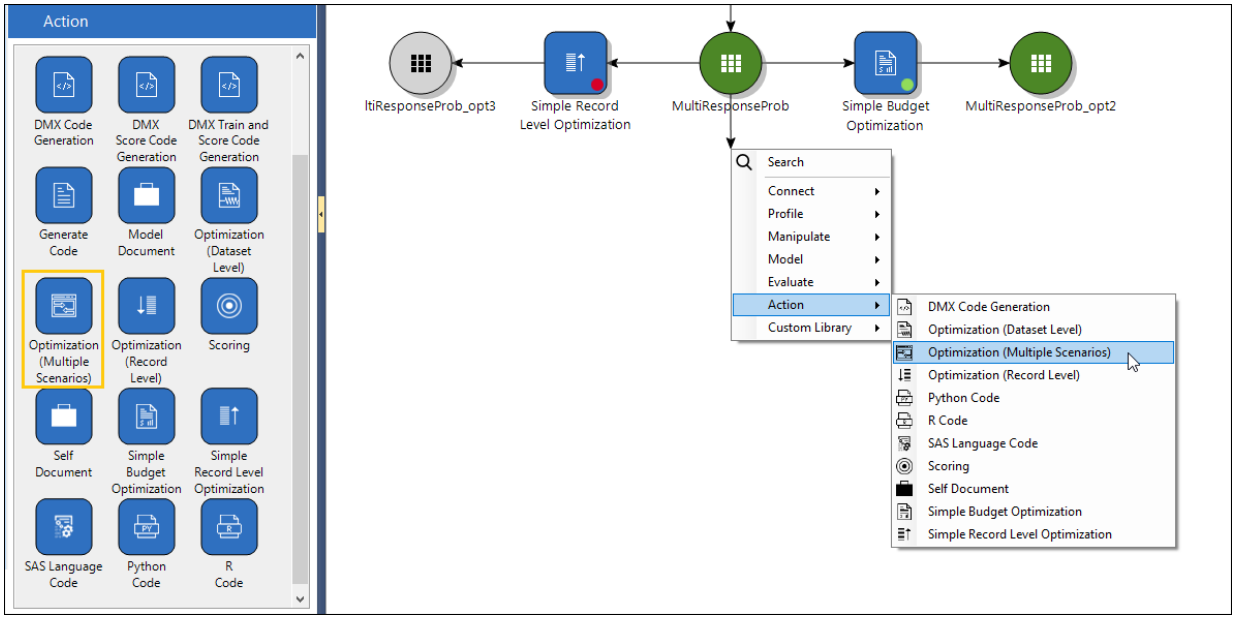
See the Task Scheduler topic for details.

To add a Multiple Scenario Optimization operation to the workflow:

- Drag a connection arrow from the input dataset node to any area on the Workflow canvas background. Releasing the mouse button brings up a context menu. From the context menu, select **Action --> Optimization (Multiple Scenarios)** as shown in the example screenshot below. A new **Optimization (Multiple Scenarios)** node will be created, and the wizard will open.

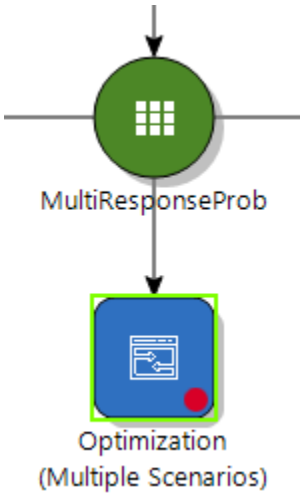
OR:

- Drag the **Optimization (Multiple Scenarios)** node from the **Action** palette onto the canvas and connect the input dataset to the node.

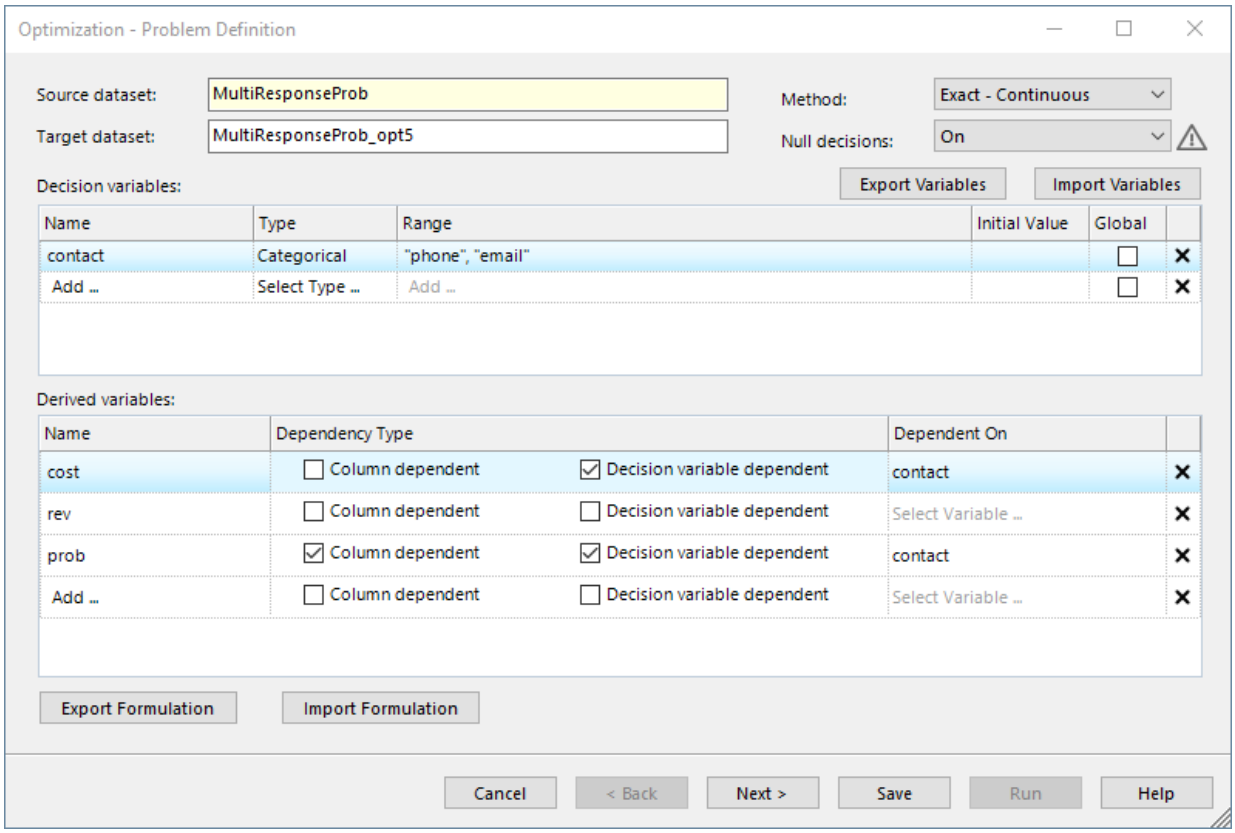


To open the wizard, you can either double-click the node or right-click on it and select **Modify**.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The first page of the wizard (*Problem Definition*) is similar to the one for Dataset-Level Optimization, except that the objective function is not defined in this page (it is defined later in the Define Scenarios page).

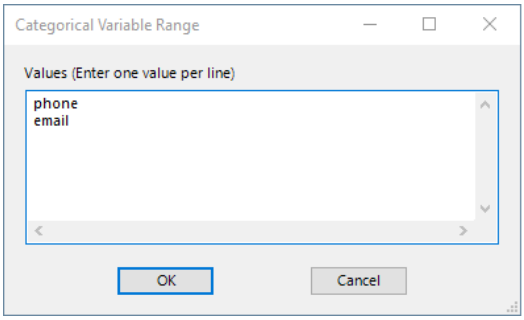


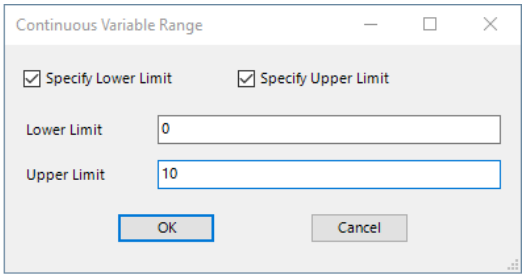
Specify the options and parameters for the operation:

- Enter the name for the output (target) dataset that will contain optimized treatments
- Specify the method for the optimization algorithm: *Exact - Continuous* or *Exact - Discrete*

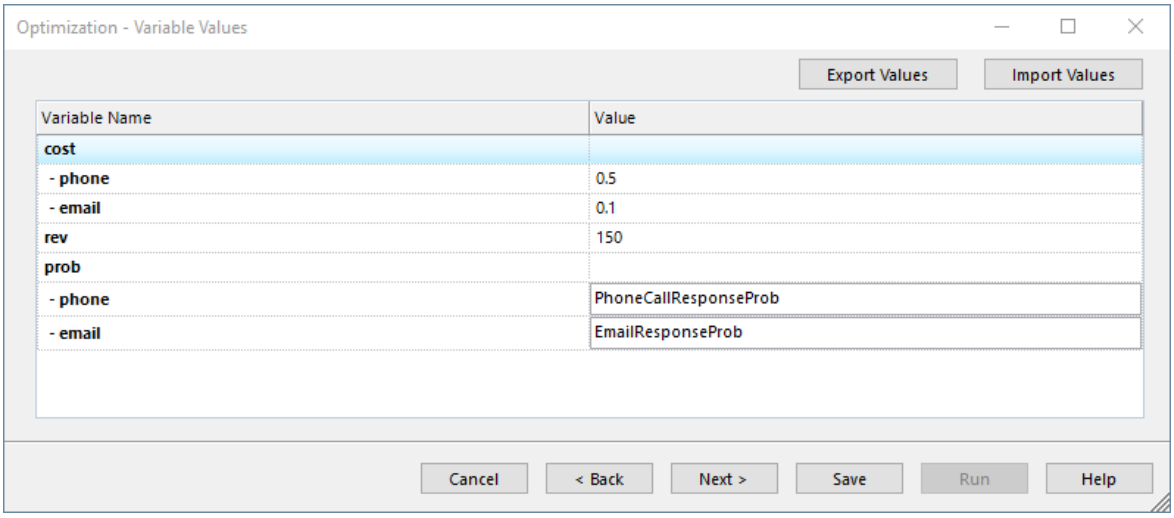
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Specify whether to produce null values for decisions (solutions). **Note**: Turning off the option for null decisions greatly increases the possibility of an infeasible problem (a problem that has no solution). The optimizer will automatically recognize and give warnings about infeasible problems, however, it may take a long time to detect infeasibility and return an error. We recommend that only experienced users turn off null decisions.
- Define the *Decision variables*:
  - Specify the type: *Categorical*, *Continuous*, or *Integer*.
  - Specify the range: Click *Add* in the Range column and enter all possible values for the categorical variables - one value per line, as shown below - and specify the lower and/or upper limits for the values of the continuous variables:



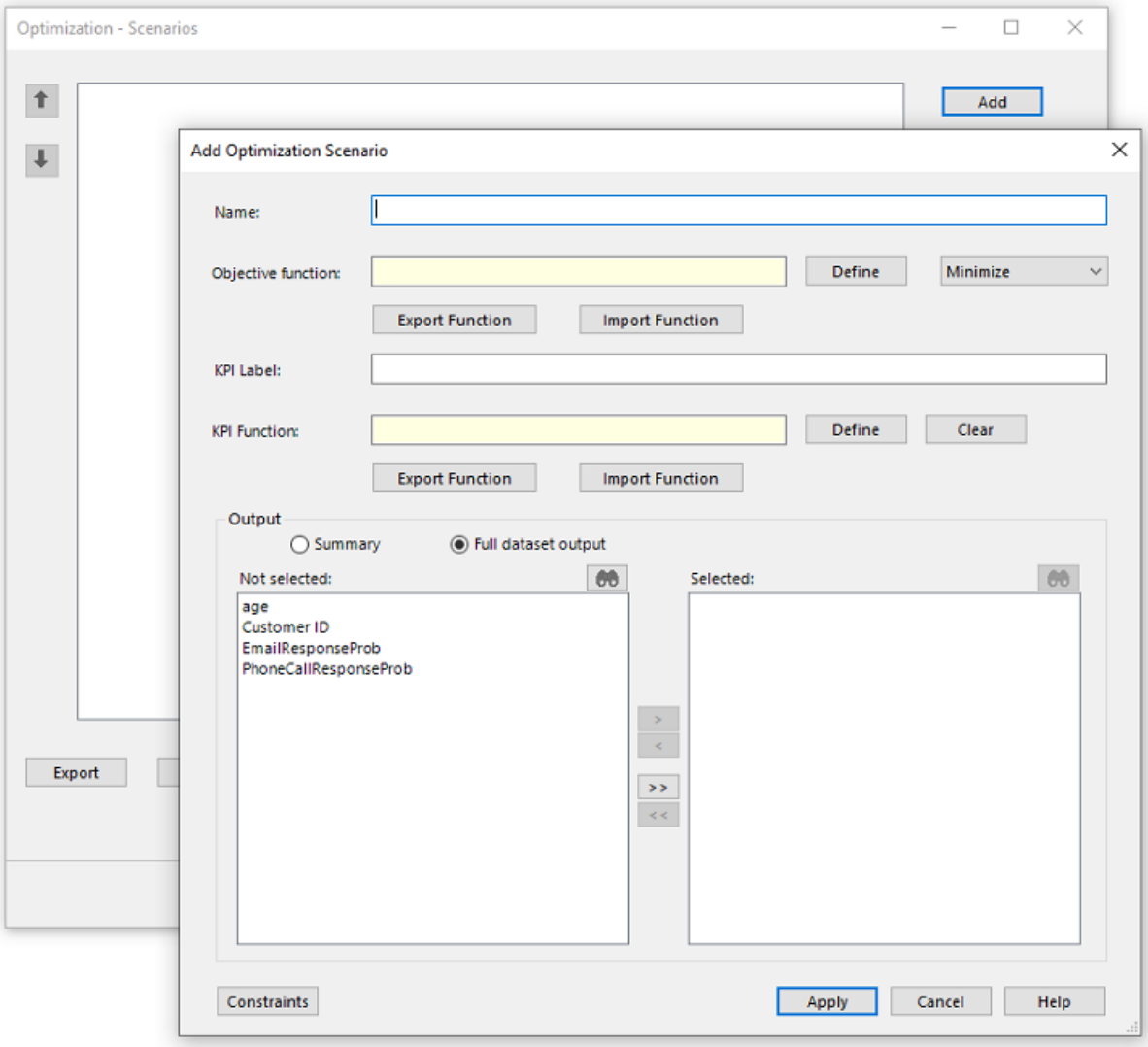


  - Specify the initial value for continuous variables in the *Initial Value* column.
  - Select the ***Global*** option if you need a single (global) decision variable to be used with the same single value across every row of the dataset. Otherwise the variable will be automatically translated to a family of decision variables - one for each row in the dataset
- For *Derived variables* (variables that depend on other variables/columns in the dataset and/or on the decision variables), specify their names and types of dependence. If any one depends on a decision variable, specify which one.
- Click **Next**. In the next page, select the probability fields for the probabilities of each discrete decision, and specify the values for the derived variables where applicable.



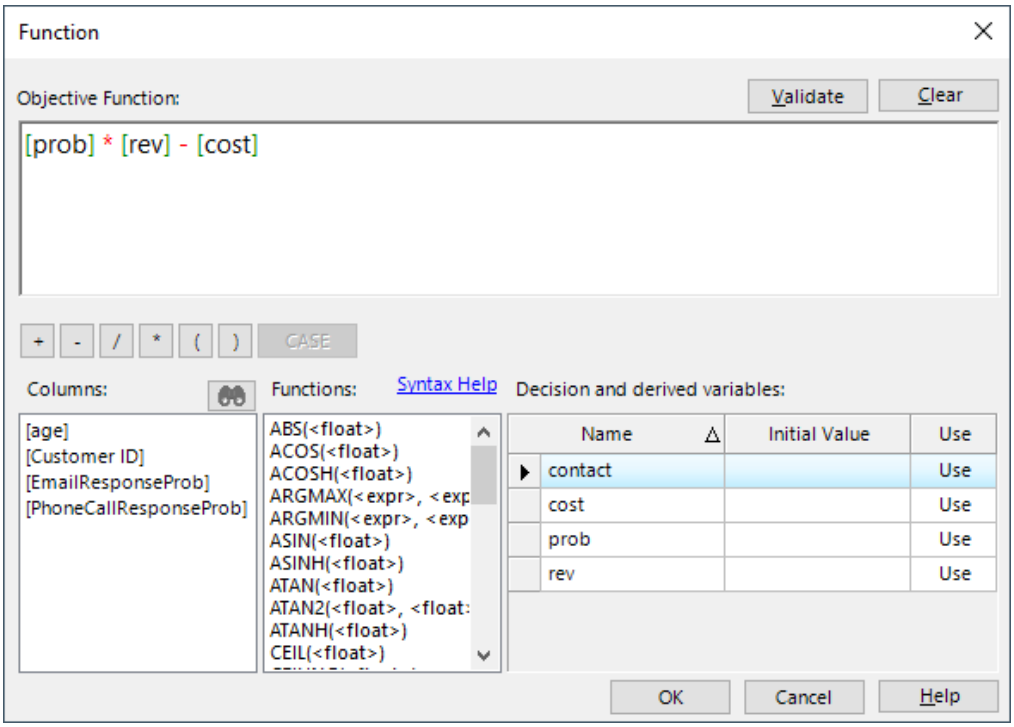
- Click **Next**. In the next page [Optimization - Scenarios], click **Add** to add a new scenario or click **Import** to import a problem definition from a previously saved definition file.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

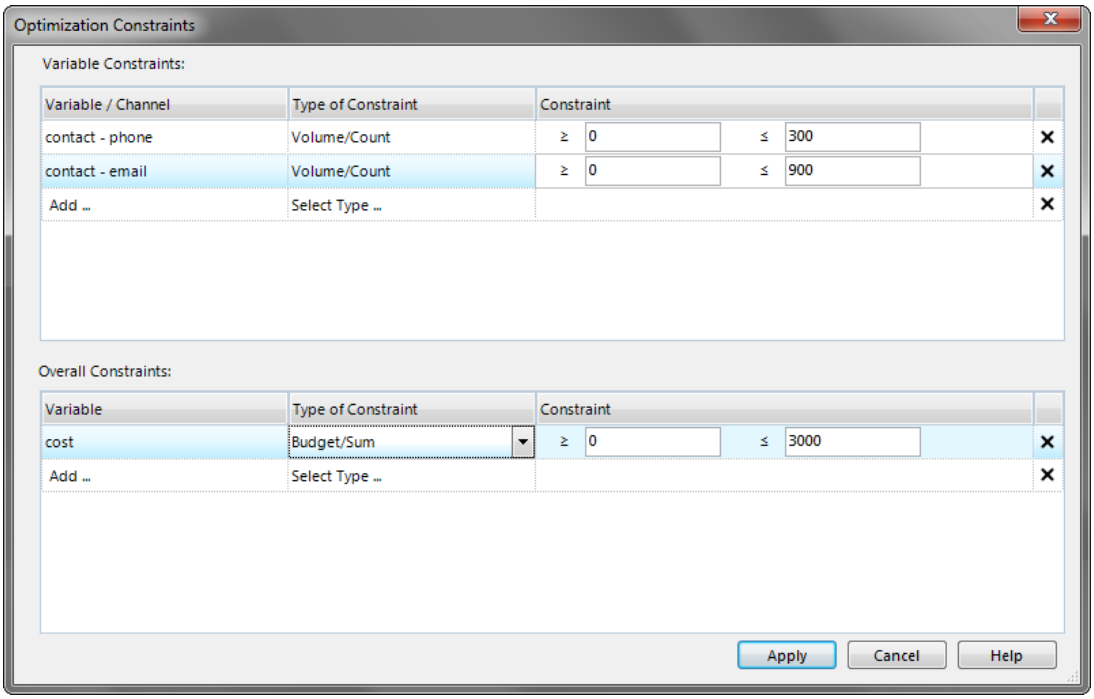


- The [Add Optimization Scenario] page opens.
  - Enter the name for the new scenario
  - Define the objective function. Use the **Import Function** button if you wish to import the function formulas from a previously saved expression file. The Expression Editor for defining the functions is the same as in the other Optimization node wizards.
  - Define the KPI function (if required). Use the **Import Function** button if you wish to import the function formulas from a previously saved expression file.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

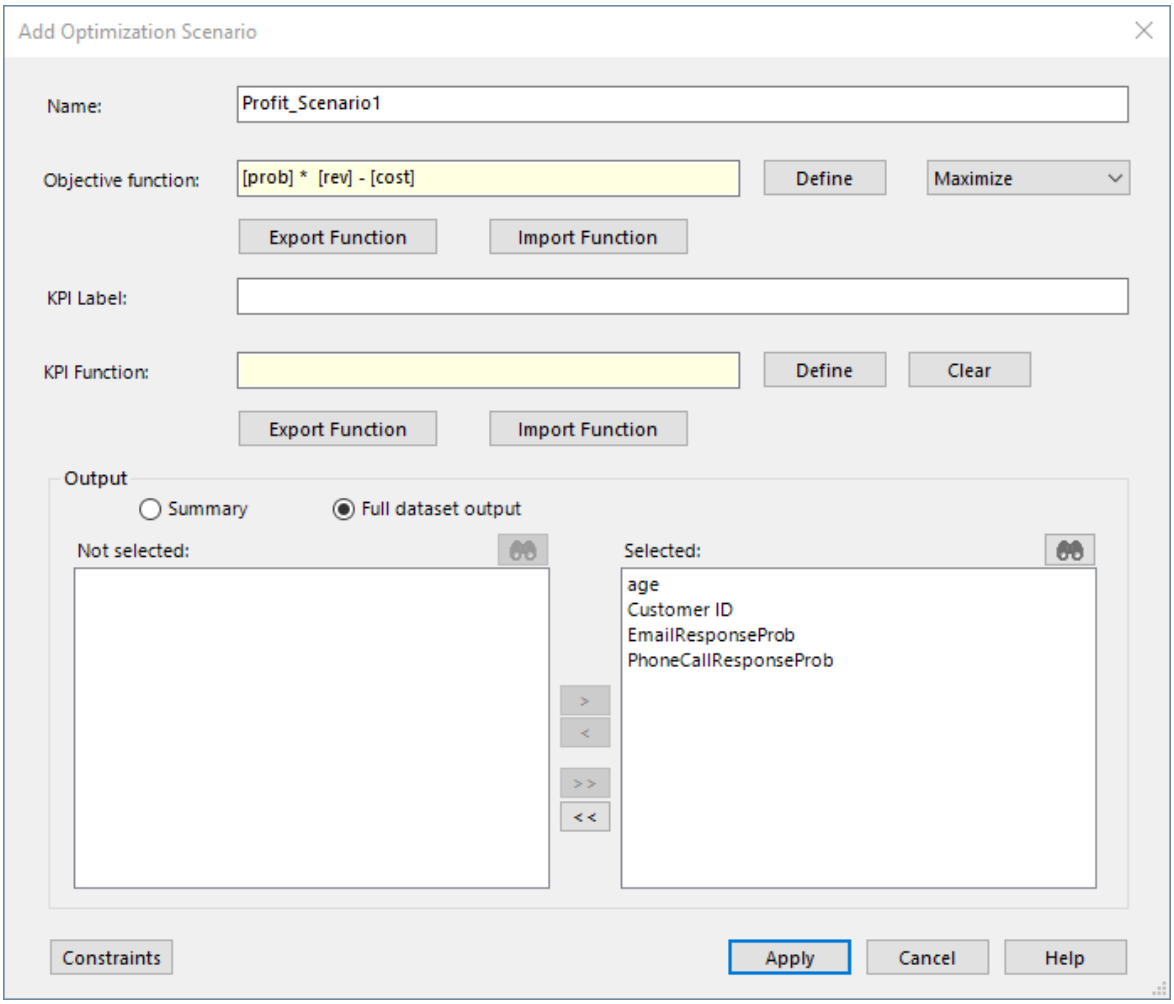


- Select the original fields to be included in the output
- Click the **Constraints** button to define constraints. The [Optimization - Constraints] dialog allows you to specify per-channel constraints and Overall constraints.



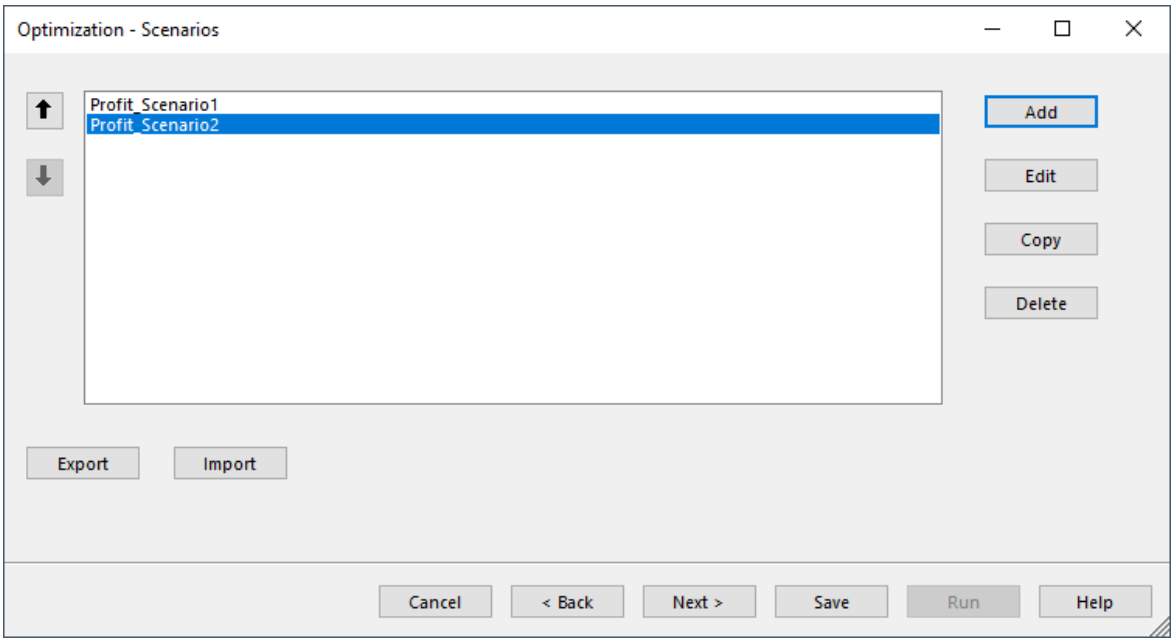
- Click **Apply** when finished defining the constraints. You will be brought back to the [Edit Optimization Scenario] page.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

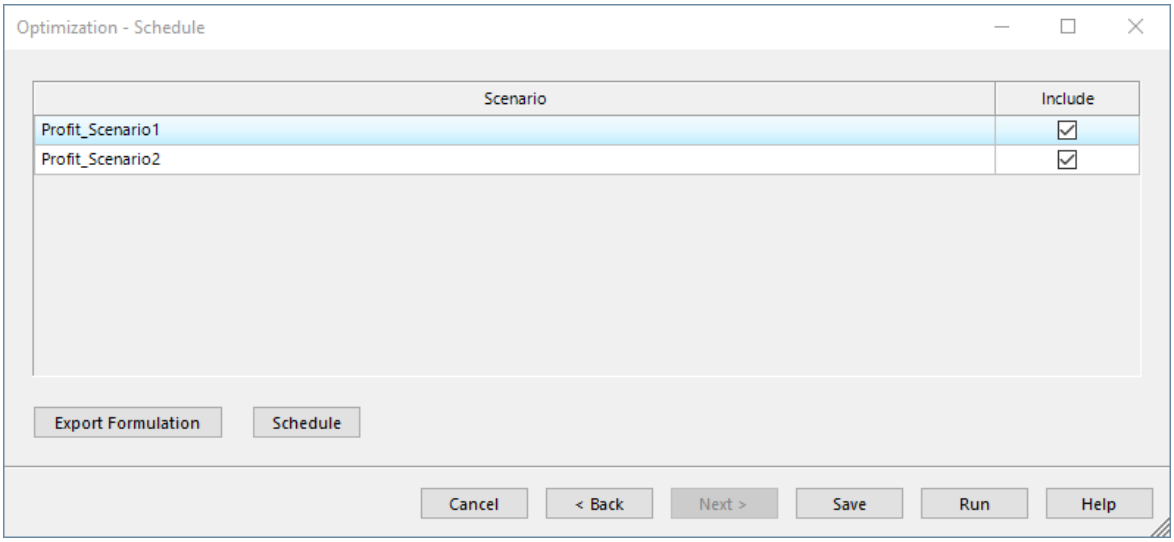


- Click **Apply** in the Add Optimization Scenario dialog when finished defining the scenario. Once you are finished defining all scenarios, they are displayed in the Scenarios page, where you can change their order, edit or copy them, export their definitions, or click Next proceed to the Scheduling page.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next** to proceed to the **Scheduling** page, where you can include or exclude the defined scenarios for execution, or schedule them for running at a specific time. See the Task Scheduler topic for details.



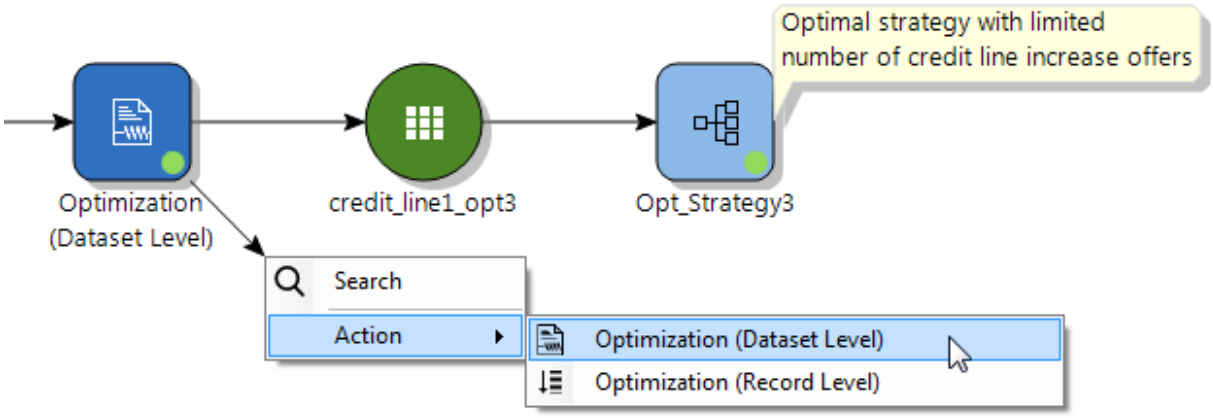
Copy, Export, and Import Problem Formulations and Variables

Optimization problem formulations can take a while to set up. If you want to try a few similar schemes, you can avoid wasting time setting up the same variables, functions, and constraints multiple times by saving the formulation and variables from one optimization node for use in another node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

1. Copying problem formulations using node connector

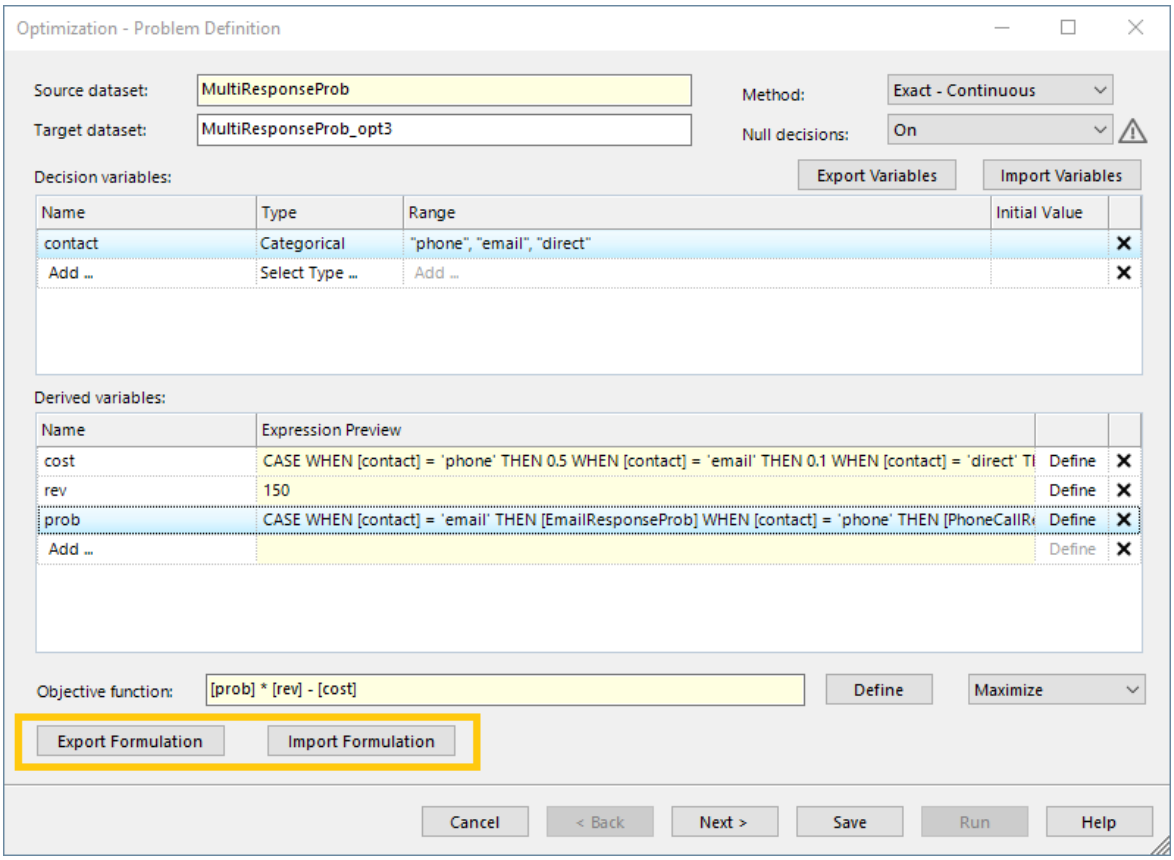
To copy the optimization problem formulation to another node within the same workflow canvas, just connect the source node to the target node. If the target node has not been created yet, then drag a connection arrow from the source node to any area on the Workflow canvas background. From the context menu, select **Action --> Optimization** (choose the required type), as in the example screenshot below. A new Optimization node will be created, which will inherit all applicable parameters from the source node.



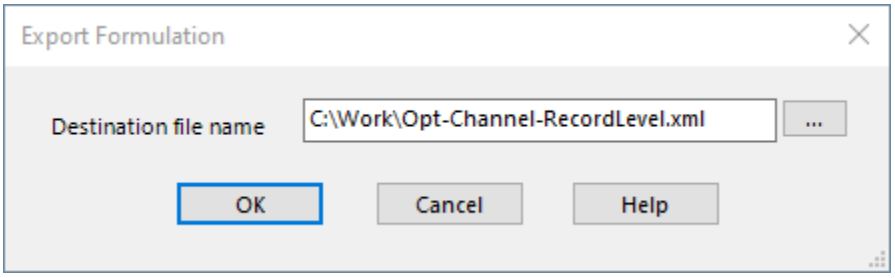
2. Exporting and importing problem formulations

To copy the optimization problem formulation to an optimization node in another workflow or project, or to send it to a colleague, use the buttons "**Export Formulation**" and "**Import Formulation**" in the Optimization wizard. The former saves the problem formulation in an XML file that can be reused for another node. The latter imports the problem formulation from a file (if it is compatible with the target optimization node type).

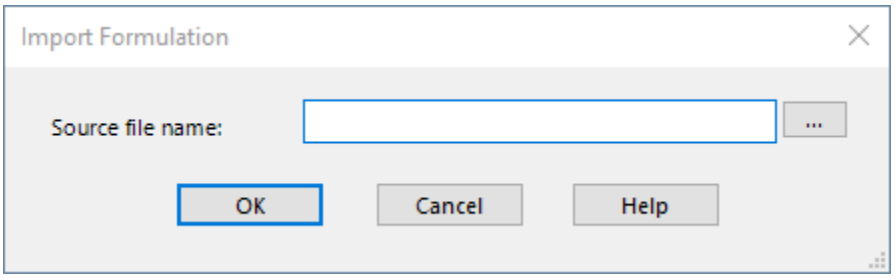
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The "**Export Formulation**" button is enabled in any optimization node that has been saved or run successfully. Click **Export Variables**, browse for the desired destination folder, and specify the file name.



The "**Import Formulation**" button opens the browse dialog. Browse and select the XML file that contains the formulation to be imported.

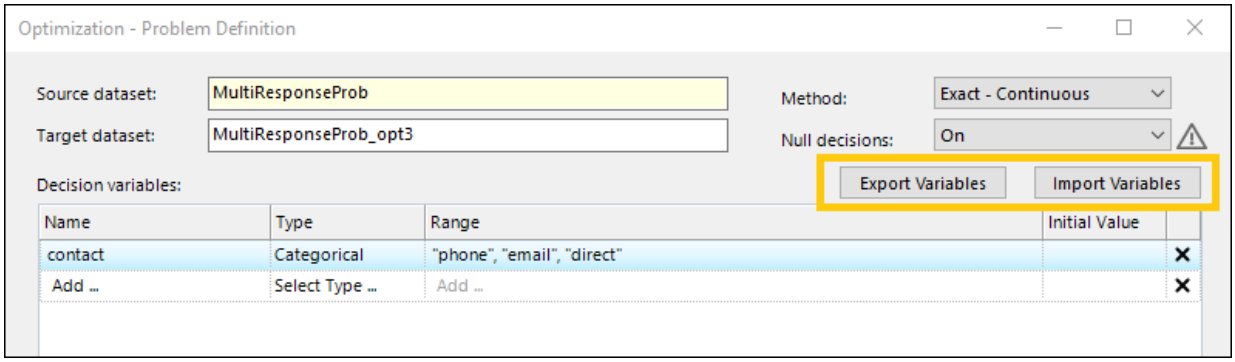


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- It is possible to import formulations exported from a Dataset-Level Optimization node into a Record-Level Optimization node, or vice versa. In these cases, the objective function, decision variables, and derived variables are copied, but no information about the constraints is applied from the file.
- Any defined decision variables are carried over with all the same saved settings.
- The objective function is carried over, but if it includes columns from the dataset which don't exist in the dataset connected to the new optimization node, the objective function remains, but you need to edit its expression before running the node.
- Derived variables dependent on columns that don't exist in the dataset connected to the target node are still carried over, but have blank fields for these columns.
- All constraints remain the same, if applicable. If there is an incompatibility (e.g., if a constraint makes reference to a column that does not exist in the dataset connected to the new node), a warning is provided before running the node. In that case you need to manually change the affected constraints as needed.

3. Exporting and importing variable definitions

Decision variables are the variables whose values are optimized to maximize or minimize the objective function. Their definition includes their names, types (*Categorical*, *Continuous*, or *Integer*) and ranges of values. To save time and efficiently reuse variable definitions or share them, use the buttons "**Export Variables**" and "**Import Variables**" in the Optimization wizard.

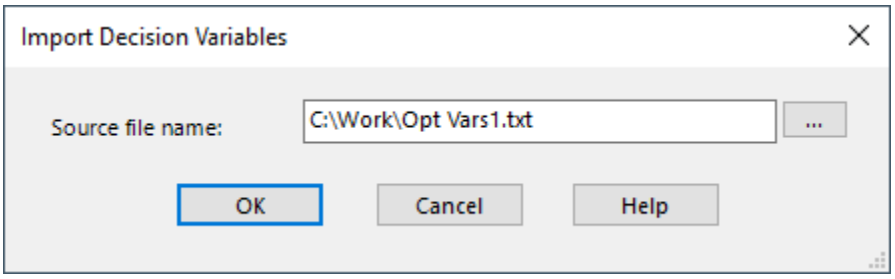


A decision variable definition file must be a text file with one line per variable, with the following syntax (note the double quotes):

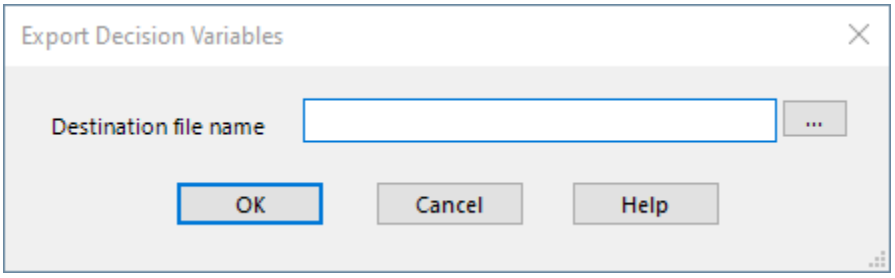
```
"<variable_name>", "<variable_type>", "<value1>", "<value2>",
....  "<valueN>"
```

To import the definition from a file, click the **Import Variables** button, browse for the source file and click OK.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



To export the variable definitions for future use, click **Export Variables**, browse for the desired destination folder, and specify the file name.



Task Scheduler

The **Task Scheduler** allows you to schedule the execution of one or more functional nodes in Altair workflows. Scheduled tasks can run even in unattended mode when the application is closed and the user is logged out. This can be especially beneficial for complex tasks or very large data volumes that take a lot of time to process.

The task scheduler is available only for the **Multiple Scenario Optimization** node in Knowledge Studio.

The scheduler can be invoked in three ways:

- from the node wizard, using the **Schedule** button in one of the wizard pages
- from the context menu when right-clicking on the desired node
- from the main menu, using the **Scheduled Tasks** icon  (in this context it only shows the list of existing scheduled tasks and does not allow to schedule new tasks).

If scheduled tasks on your computer can only be created by users with administrative privileges, you may need to start Knowledge Studio **as administrator** before using the scheduling feature.

On Windows 10, 11, and Windows Server 2016, 2019, 2022, you will also need to edit the properties of the newly created task in Windows Task Scheduler. See the section *Modifying Scheduled Task Properties* below.

Scheduled Tasks and Licensing

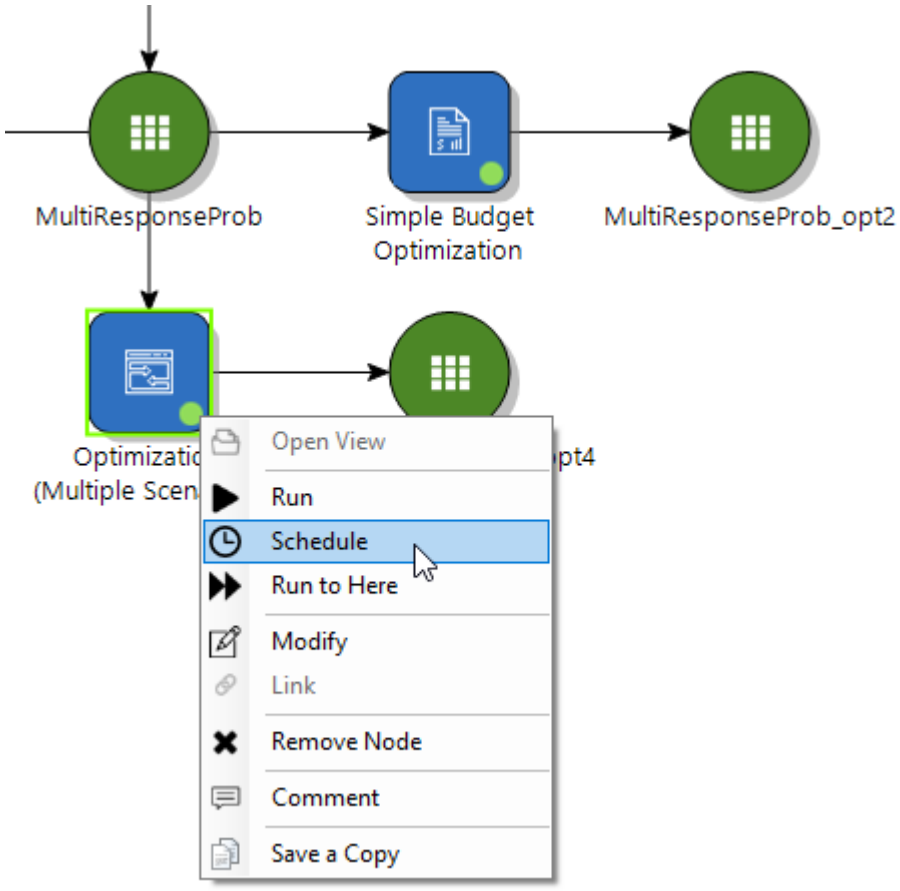
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

A scheduled task launches a separate instance of the application executable in the background, and therefore it will perform a separate license check when it starts.

- In the case of a named-user license, if you modify the task properties in Task Scheduler or cron, make sure the task is configured to run under the account of a licensed user.
- In the case of a concurrent license, the instance launched by the scheduled task will use a new connection, so it will run only if connections are available at the scheduled start time. Since tasks are usually scheduled to run in unattended mode when the application is closed, one user ID would not normally take more than one connection. However, the concurrent connections limit should be taken into account when scheduling multiple tasks.

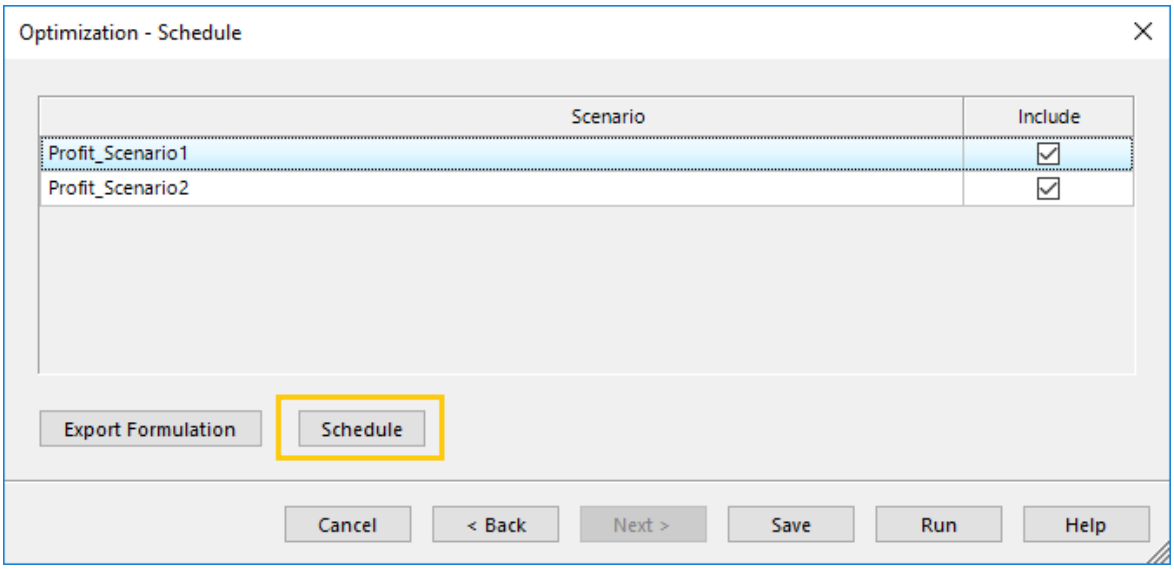
Multiple Scenario Optimization node

In Knowledge Studio, you can schedule the execution of optimization tasks for Multiple Scenario Optimization. If all settings in the node wizard have been defined, right-click on the node and select the **Schedule** command from the context menu:

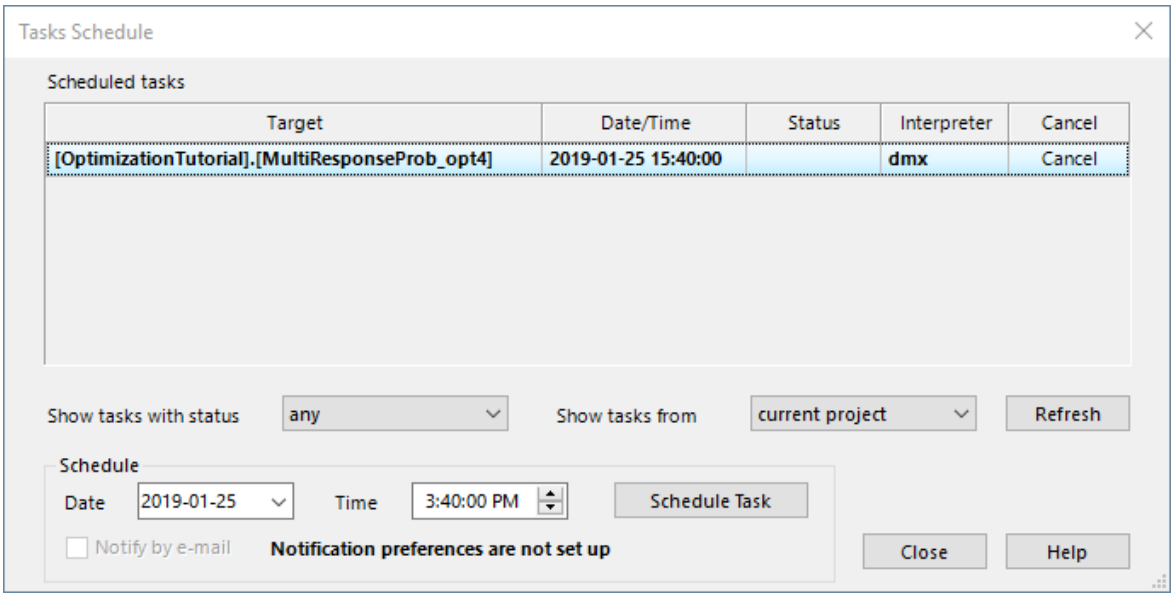


Alternatively, double-click the Multiple Scenario Optimization node to open the wizard and proceed to the last page. Select the required scenarios in the *Include* column and click the **Schedule** button.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



This opens the Tasks Schedule dialog, where you can view the previously defined tasks (if any) along with their status and properties, and schedule a new task to run the selected scenarios.



- To schedule a new task, specify the desired date and time to run it, then click the **Schedule Task** button. A new line representing this task will appear in the list above.
- To cancel an existing task, click **Cancel** for this task in the Cancel column on the right.

Once the task has been scheduled, close the Tasks Schedule dialog. The Multiple Scenario Optimization node will have a clock icon indicating that the task defined by the node is scheduled to run.

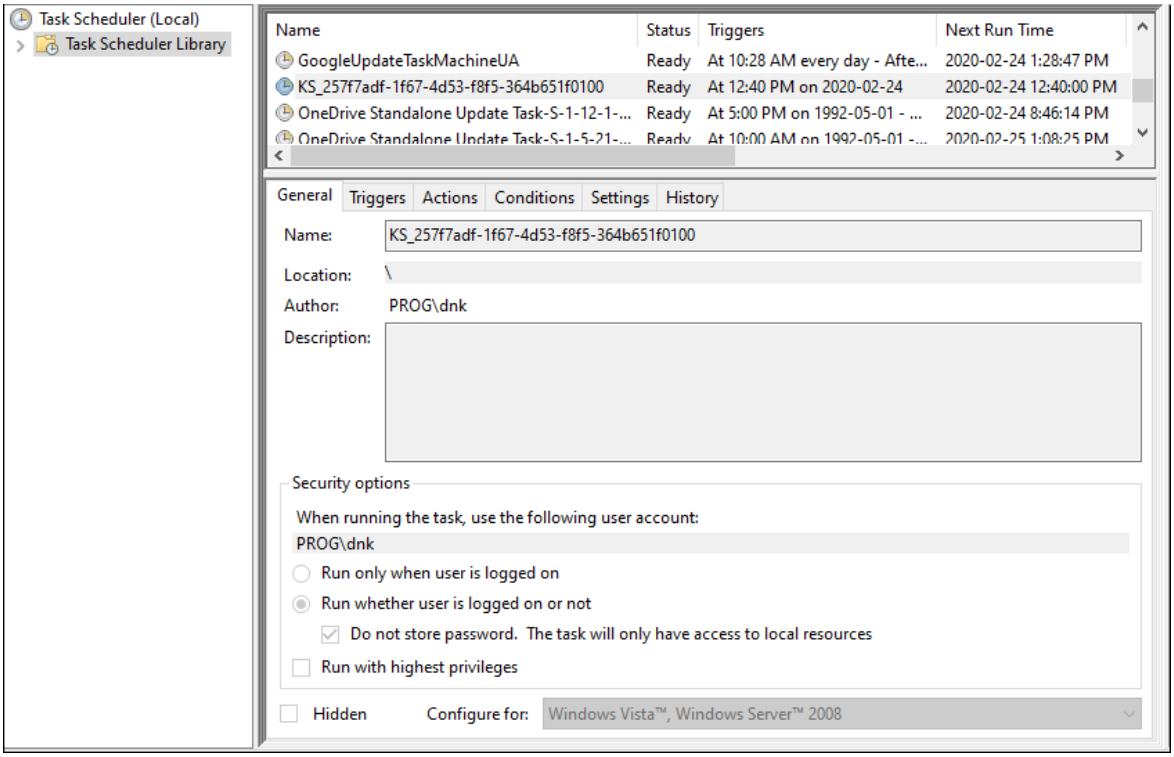
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Modifying Scheduled Task Properties

The scheduled task properties can be modified if necessary using Windows Task Scheduler (on Windows) or *cron* (on Linux).

For example, to modify the task properties on Windows, open Task Scheduler from Windows Administrative Tools and select *Task Scheduler Library* in the leftmost panel. The name of the task created by Knowledge Studio starts with "**KS**_". Double click on it and modify the settings in the *General*, *Triggers*, *Conditions*, and other tabs if necessary.

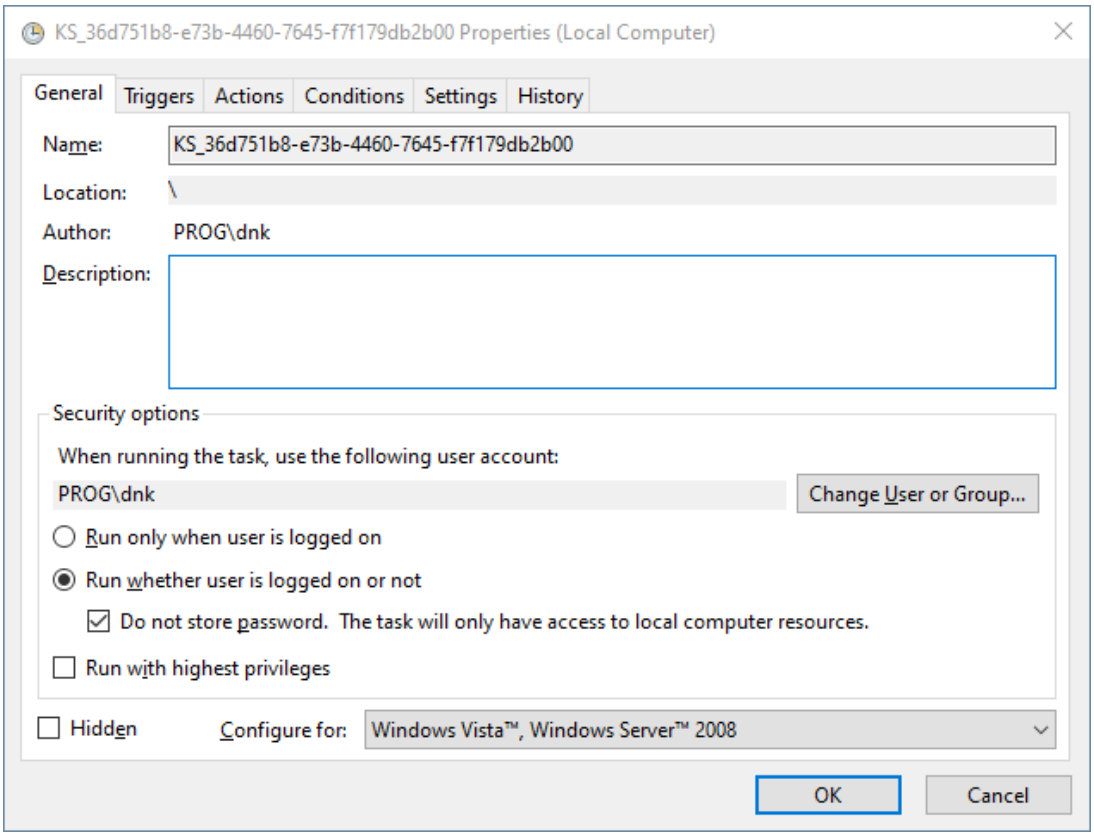


On Windows 10 and 11, Windows Server 2016, 2019, 2022, the following modifications may be necessary:

- In the ***General*** tab:
  - You may need to deselect the option "*Do not store password. The task will only have access to local resources*". When applying this change, you will need to specify the password for the account you chose for running the task.

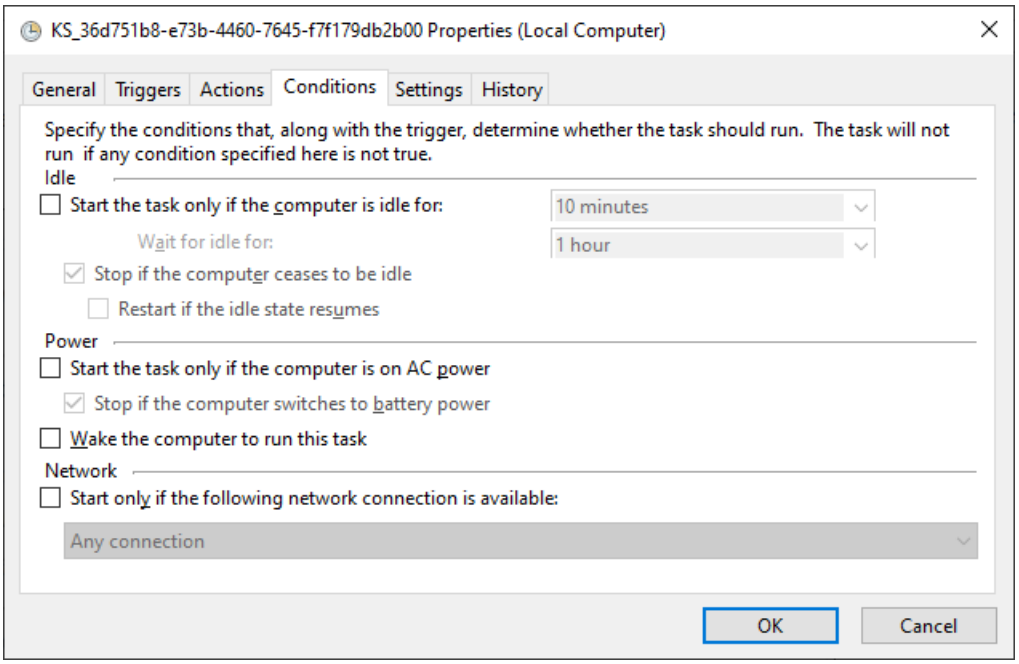
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- If necessary, click the button "*Change User or Group*" and specify a different user account for running the task. This must be an account of a licensed user. In the case of a concurrent license or Altair Units license, you can choose any user account as long as it has sufficient privileges to start the program as defined in the *Actions* tab. See the note on Scheduled Task and Licensing at the beginning of this topic.



- In the ***Conditions*** tab, specify the desired power and network availability options. Note: On laptops, the option "*Start the task only of the computer is on AC power*" is selected by default, so you may want to deselect it.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

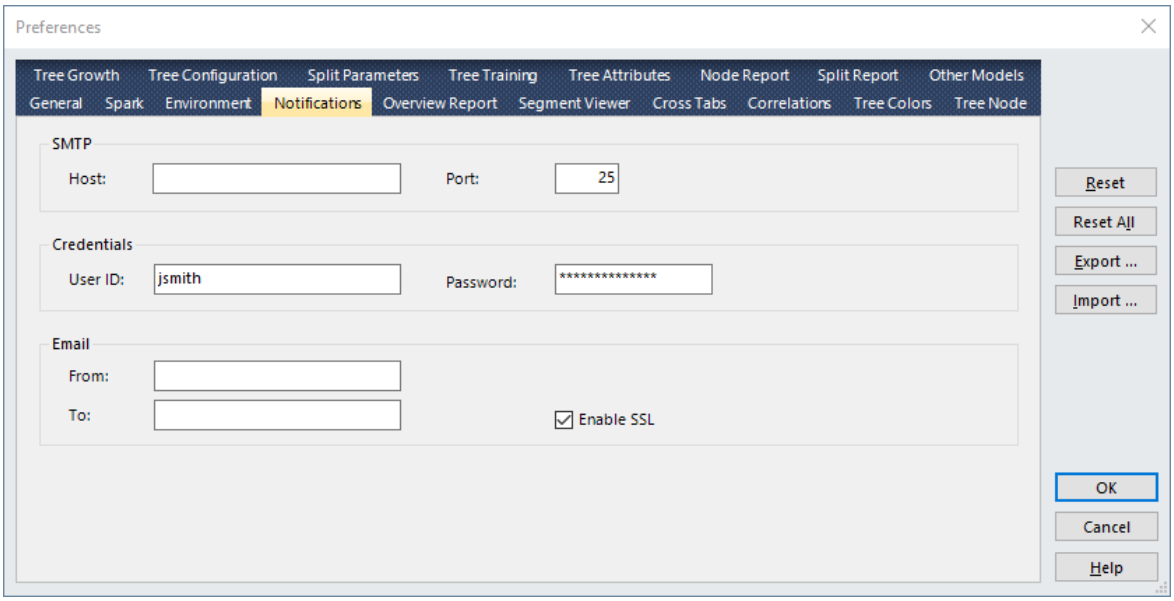


Email Notification Options

If you need to set up Email notifications in the scheduler to inform any user(s) of the task status changes, you need to set up Notification preferences in the application preferences under the **Tools** menu. After that, the checkbox **Notify by email** will be enabled at the bottom of the Tasks Schedule dialog, and you can select it if necessary. When setting the Email Notification options using **Tools | Preferences**, specify the SMTP host name or IP address, SMTP port, User ID and password (optional), and the *From/To* Email addresses. The option **Enable SSL** is selected by default, but you may need to disable it (depending on the security settings in your Email client). If you are not getting notifications after configuring these settings, deselect the Enable SSL option and try again.

The settings shown in the image below are given as an example template. Ask your IT team for assistance if you are not sure what parameters to specify.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



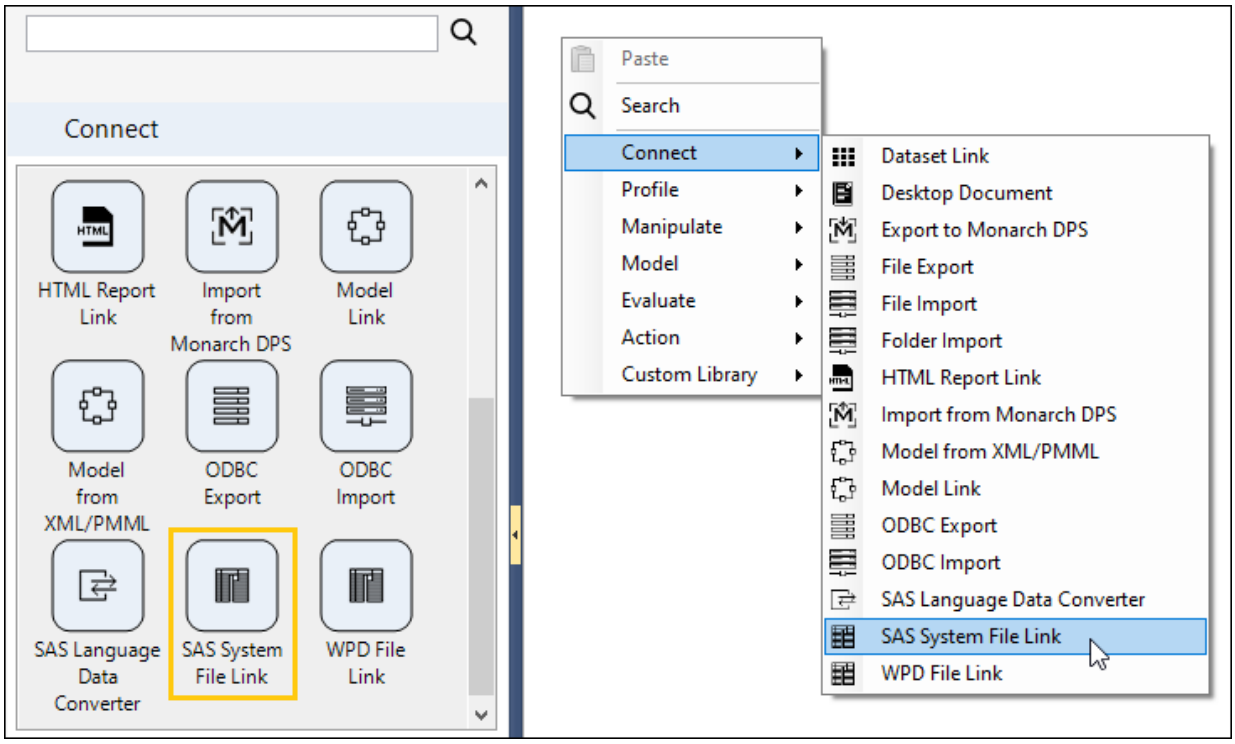
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Integration using SAS Language

SAS System files (*.sas7bdat) can be used as data nodes in the Workflow. They can be converted to Altair KDD and WPD formats using the SAS Language Data Converter node.

To create a dataset node linked to a SAS System file:

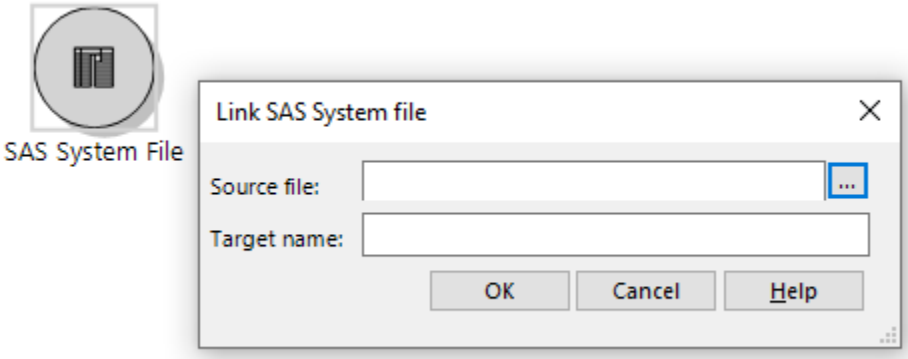
- Right-click anywhere on the Workflow canvas and select the context menu command **Connect --> SAS System File Link** as shown below. A new **SAS System File** will be created, and the file link dialog will open.



OR:

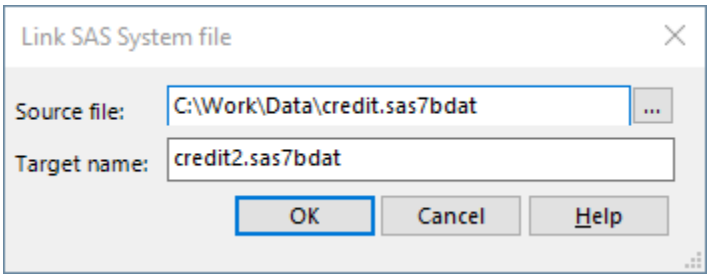
- Drag the **SAS System File Link** node from the **Connect** palette onto the canvas. Then double-click the node or right-click on it and select **Link** from the context menu.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



In the Link SAS System File dialog:

- Browse for the source file (*.sas7bdat) and specify its name in the *Source file* field
- In the *Target name* field, specify the name for the new dataset node that the file will be linked to. If you specify the target name without extension, the *.sas7bdat extension will be appended automatically.

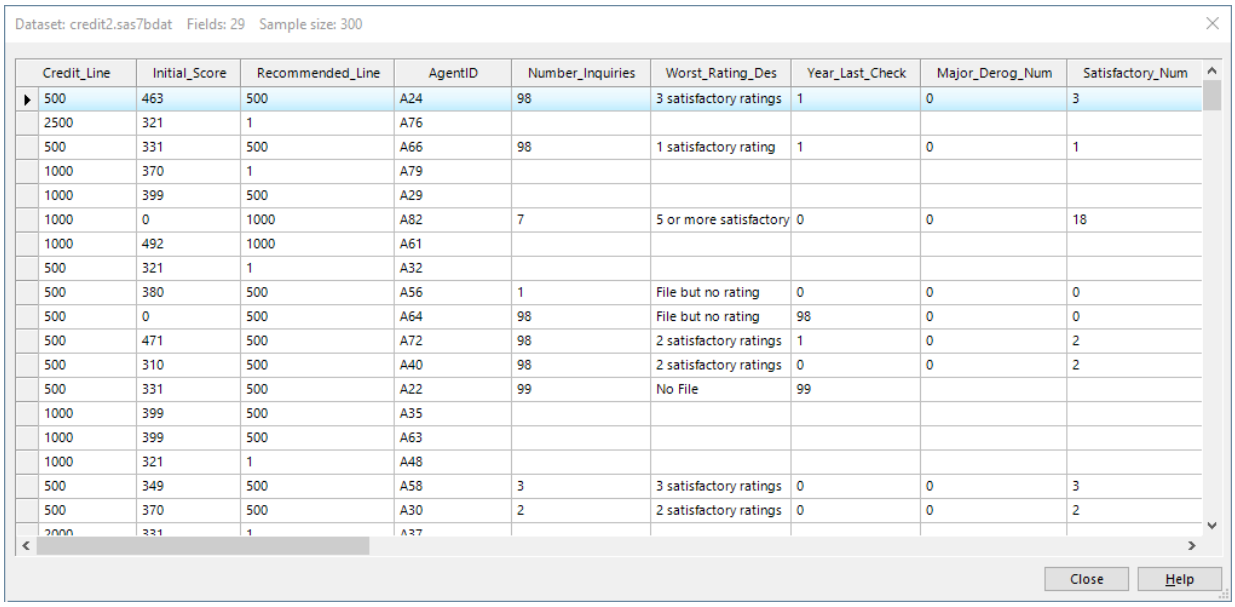


- Click **OK**. The source file will be copied to the current project folder, shown in the Project Explorer pane, and linked as a new dataset node on the Workflow canvas.



Double-click the dataset node to view its contents if necessary. This will open the data preview window, which shows the top 300 rows of the dataset. The data cells in the grid are not editable.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

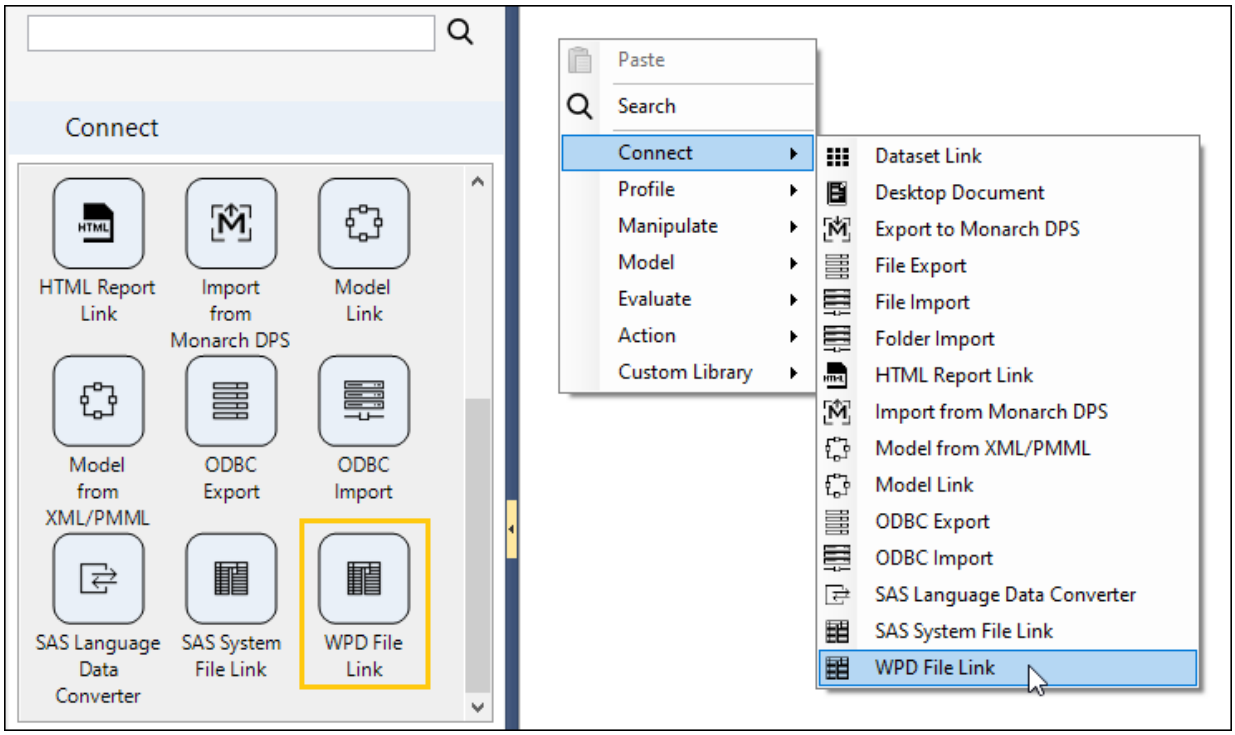


Credit_Line	Initial_Score	Recommended_Line	AgentID	Number_Inquiries	Worst_Rating_Des	Year_Last_Check	Major_Derog_Num	Satisfactory_Num
500	463	500	A24	98	3 satisfactory ratings	1	0	3
2500	321	1	A76					
500	331	500	A66	98	1 satisfactory rating	1	0	1
1000	370	1	A79					
1000	399	500	A29					
1000	0	1000	A82	7	5 or more satisfactory	0	0	18
1000	492	1000	A61					
500	321	1	A32					
500	380	500	A56	1	File but no rating	0	0	0
500	0	500	A64	98	File but no rating	98	0	0
500	471	500	A72	98	2 satisfactory ratings	1	0	2
500	310	500	A40	98	2 satisfactory ratings	0	0	2
500	331	500	A22	99	No File	99		
1000	399	500	A35					
1000	399	500	A63					
1000	321	1	A48					
500	349	500	A58	3	3 satisfactory ratings	0	0	3
500	370	500	A30	2	2 satisfactory ratings	0	0	2

WPD files (*.wpd) are data files in the format of Altair Analytics Workbench (WPS Workbench). They can be converted to Altair KDD and SAS System file formats using the [SAS Language Data Converter] node.

To create a dataset node linked to a WPD file:

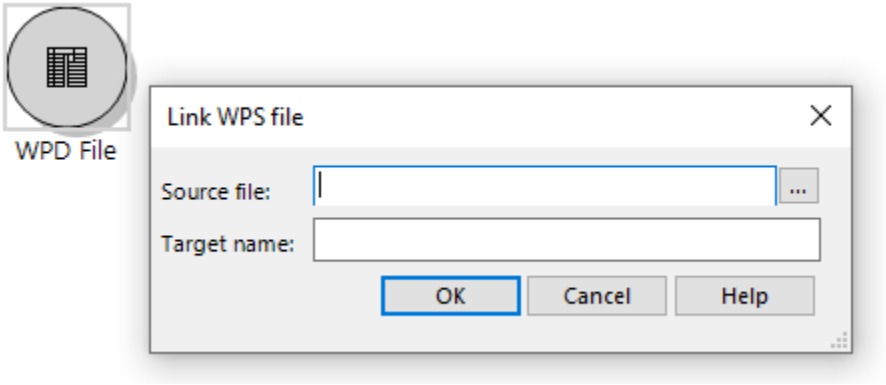
- Right-click anywhere on the Workflow canvas and select the context menu command **Connect --> WPD File Link** as shown below. A new **WPD File** will be created, and the file link dialog will open.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

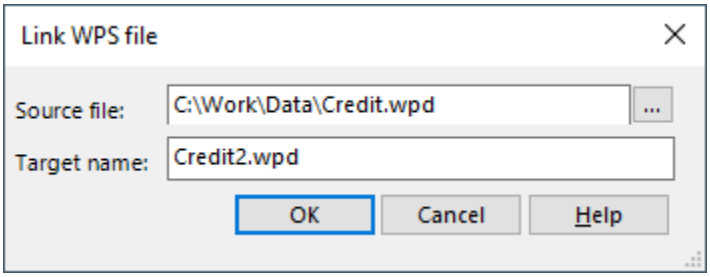
OR:

- Drag the **WPD File Link** node from the **Connect** palette onto the canvas. Then double-click the node or right-click on it and select **Link** from the context menu.



In the Link WPD File dialog:

- Browse for the source file (*.sas7bdat) and specify its name in the *Source file* field
- In the *Target name* field, specify the name for the new dataset node that the file will be linked to. If you specify the target name without extension, the *.sas7bdat extension will be appended automatically.

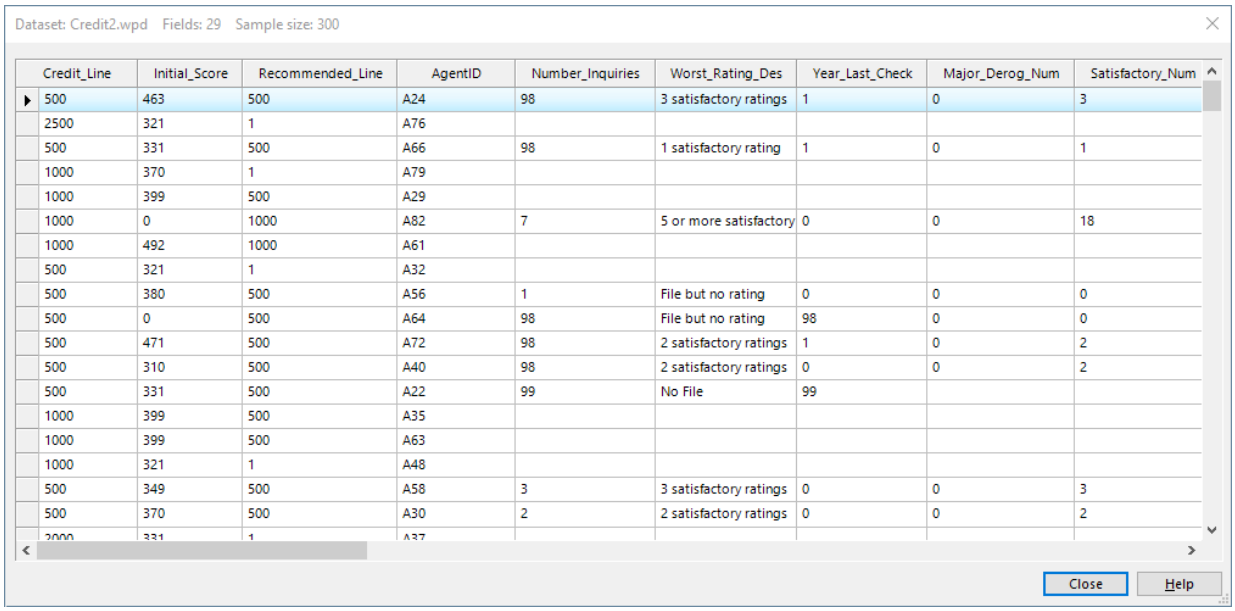


- Click **OK**. The source file will be copied to the current project folder, shown in the Project Explorer pane, and linked as a new dataset node on the Workflow canvas.



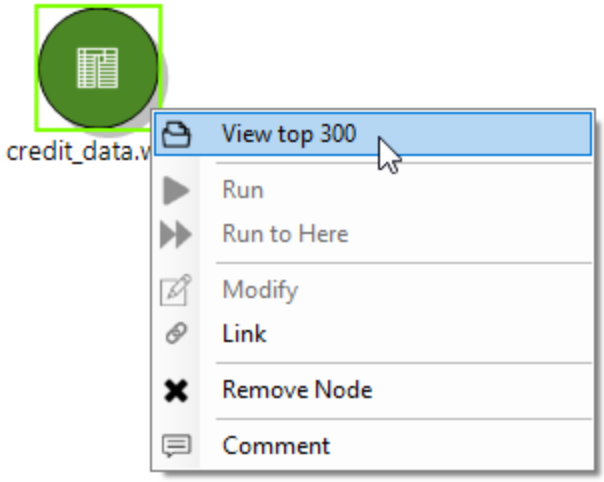
Double-click the dataset node to view its contents if necessary. This will open the data preview window, which shows the top 300 rows of the dataset. The data cells in the grid are not editable.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



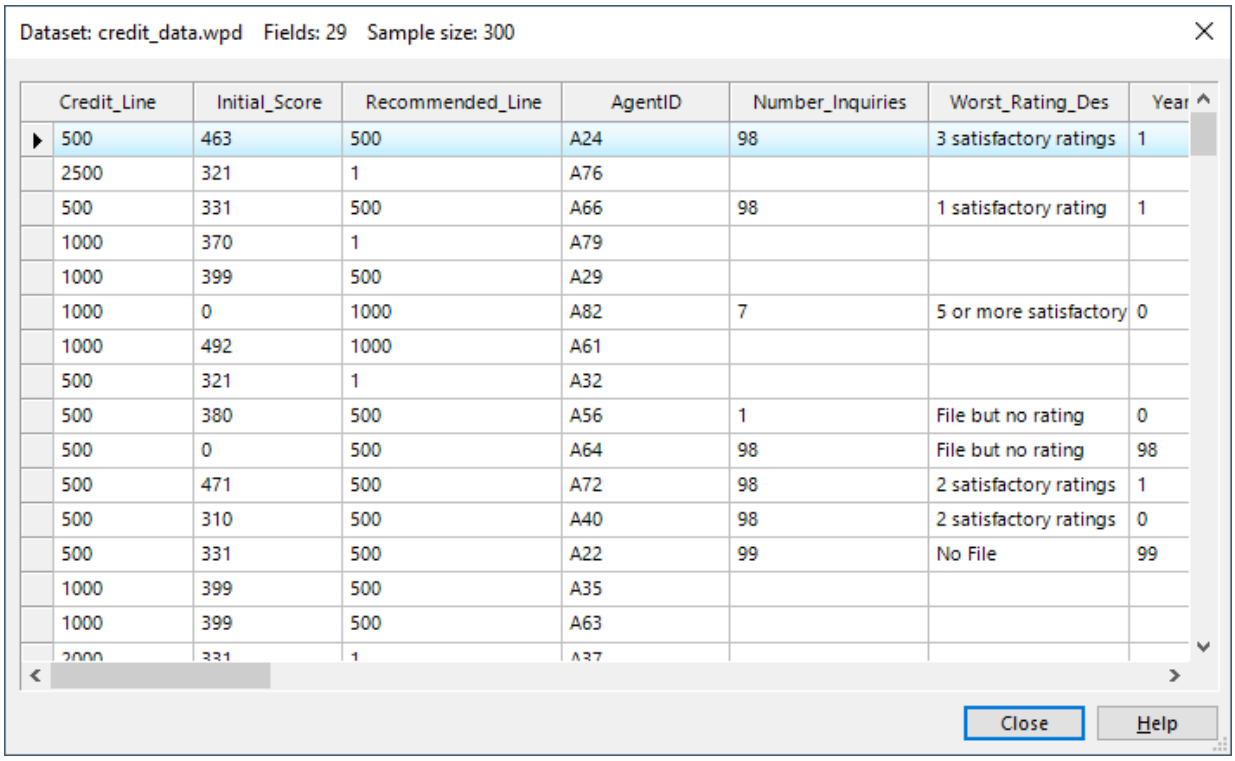
Credit_Line	Initial_Score	Recommended_Line	AgentID	Number_Inquiries	Worst_Rating_Des	Year_Last_Check	Major_Derog_Num	Satisfactory_Num
500	463	500	A24	98	3 satisfactory ratings	1	0	3
2500	321	1	A76					
500	331	500	A66	98	1 satisfactory rating	1	0	1
1000	370	1	A79					
1000	399	500	A29					
1000	0	1000	A82	7	5 or more satisfactory	0	0	18
1000	492	1000	A61					
500	321	1	A32					
500	380	500	A56	1	File but no rating	0	0	0
500	0	500	A64	98	File but no rating	98	0	0
500	471	500	A72	98	2 satisfactory ratings	1	0	2
500	310	500	A40	98	2 satisfactory ratings	0	0	2
500	331	500	A22	99	No File	99		
1000	399	500	A35					
1000	399	500	A63					
1000	321	1	A48					
500	349	500	A58	3	3 satisfactory ratings	0	0	3
500	370	500	A30	2	2 satisfactory ratings	0	0	2

To view top 300 records in a WPD or SAS System file, double-click on the corresponding dataset node on the workflow canvas or right-click on it and select **View top 300** from the context menu.



This opens the data preview window, which shows top 300 records of the dataset:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Credit_Line	Initial_Score	Recommended_Line	AgentID	Number_Inquiries	Worst_Rating_Des	Year
500	463	500	A24	98	3 satisfactory ratings	1
2500	321	1	A76			
500	331	500	A66	98	1 satisfactory rating	1
1000	370	1	A79			
1000	399	500	A29			
1000	0	1000	A82	7	5 or more satisfactory	0
1000	492	1000	A61			
500	321	1	A32			
500	380	500	A56	1	File but no rating	0
500	0	500	A64	98	File but no rating	98
500	471	500	A72	98	2 satisfactory ratings	1
500	310	500	A40	98	2 satisfactory ratings	0
500	331	500	A22	99	No File	99
1000	399	500	A35			
1000	399	500	A63			
2000	331	1	A37			

This view is read-only. The data cells are not editable.

To sort the data in any column, click on the column header. Note that only the currently displayed top 300 records will be sorted.

To open the dataset in the WPS Workbench, double-click on the icon representing this dataset in the Project Explorer pane rather than in the workflow. This will launch WPS Workbench, which will open the dataset.

Three types of datasets can be used In Knowledge Studio workflows: Altair KDD datasets, SAS System files (*.sas7bdat), and WPD files (Altair Analytics Workbench, formerly known as WPS Workbench).

Conversion between these dataset formats can be performed using the **SAS Language Data Converter** node in the Connect palette.

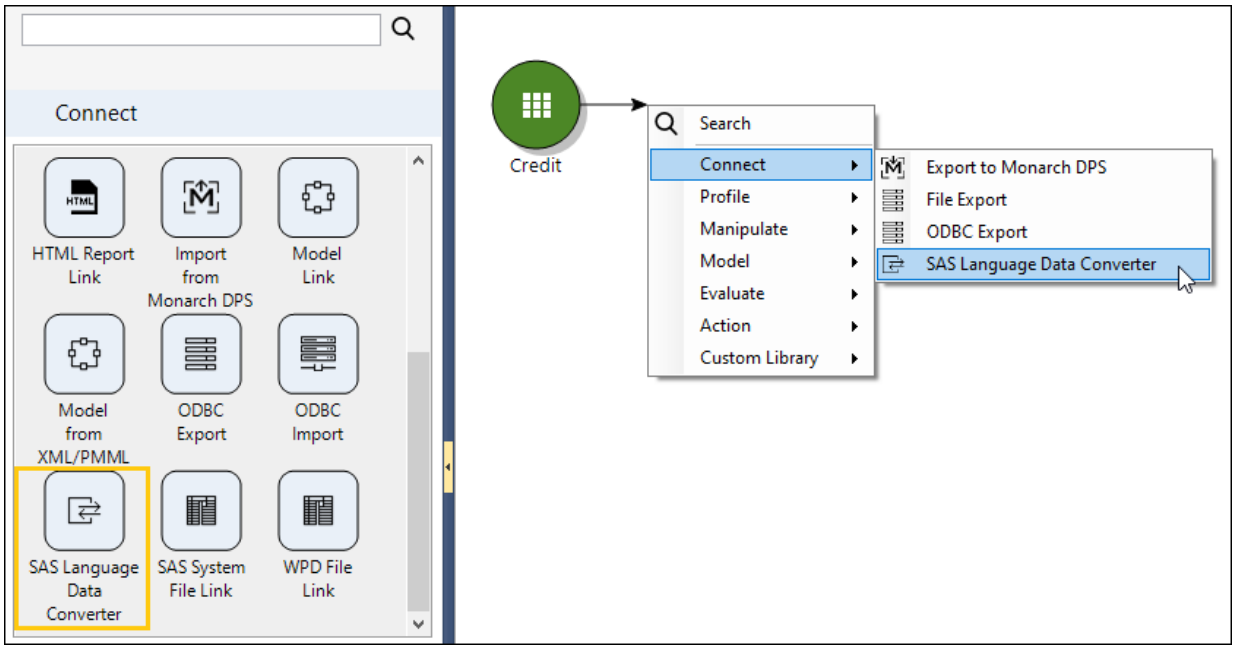
To convert a KDD, SAS System, or WPD dataset to any of these formats:

- Drag a connection arrow from the dataset node to any area on the Workflow canvas. Releasing the mouse brings up a context menu. From the context menu, select **Connect --> SAS Language Data Converter** as shown below. A new Converter node will be created, and the wizard will open.

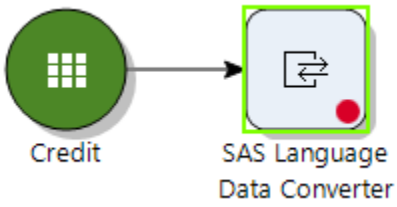
OR:

- Drag the **SAS Language Data Converter** node from the **Connect** palette onto the Workflow canvas and connect the input dataset to the node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



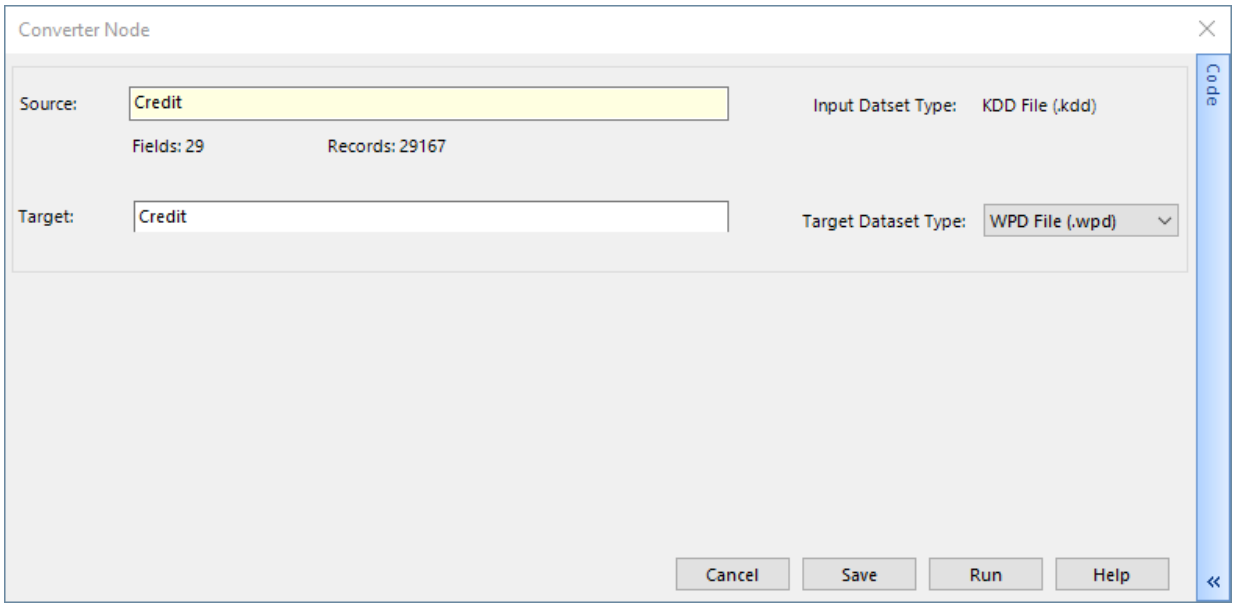
Double-click the **SAS Language Data Converter** node or right-click on it and select **Modify** to open the wizard.



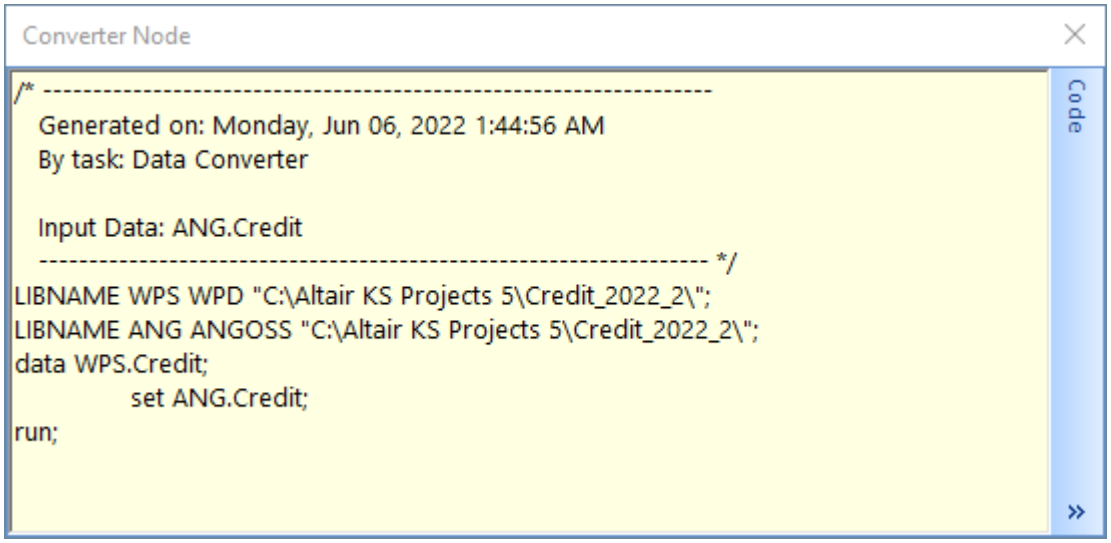
In the SAS Language Data Converter Node wizard:

- Specify the name for the output dataset in the Target field.
- Choose the target dataset type: SAS System file (*.sas7bdat), WPD file (*.wpd), or KDD file (*.kdd).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



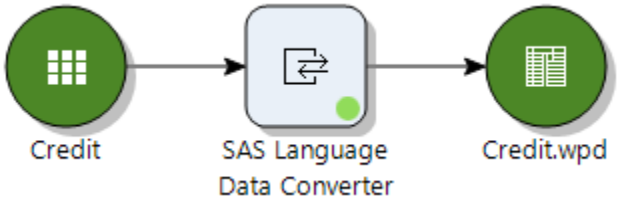
You can preview the auto-generated SAS Language code at any time by clicking the arrow at the bottom of the Code bar - the blue vertical bar on the right side of the window. Click the arrow again to close the preview.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.

The output dataset is created in the workflow after running the node. In this example, a KDD dataset is converted to a WPD dataset.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Double-click the output dataset to view the results.

Double-click the SAS Language Data Converter node to view the code and the log if necessary.

```
5            Input Data: ANG.Credit
6            ------------------------------------------------------------------- */
7         LIBNAME WPS WPD "C:\Altair KS Projects 5\Credit_2022_2\";
NOTE: Library WPS assigned as follows:
      Engine:        WPD
      Physical Name: C:\Altair KS Projects 5\Credit_2022_2\

8         LIBNAME ANG ANGOSS "C:\Altair KS Projects 5\Credit_2022_2\";
NOTE: Library ANG assigned as follows:
      Engine:        ANGOSS
      Physical Name: C:\Altair KS Projects 5\Credit_2022_2\

9         data WPS.Credit;
10          set ANG.Credit;
11        run;

NOTE: 29167 observations were read from "ANG.Credit"
NOTE: Data set "WPS.Credit" has 29167 observation(s) and 29 variable(s)
NOTE: The data step took :
      real time : 0.416
      cpu time  : 0.421

NOTE: Submitted statements took :
      real time : 1.040
      cpu time  : 0.859
```

Code | Log

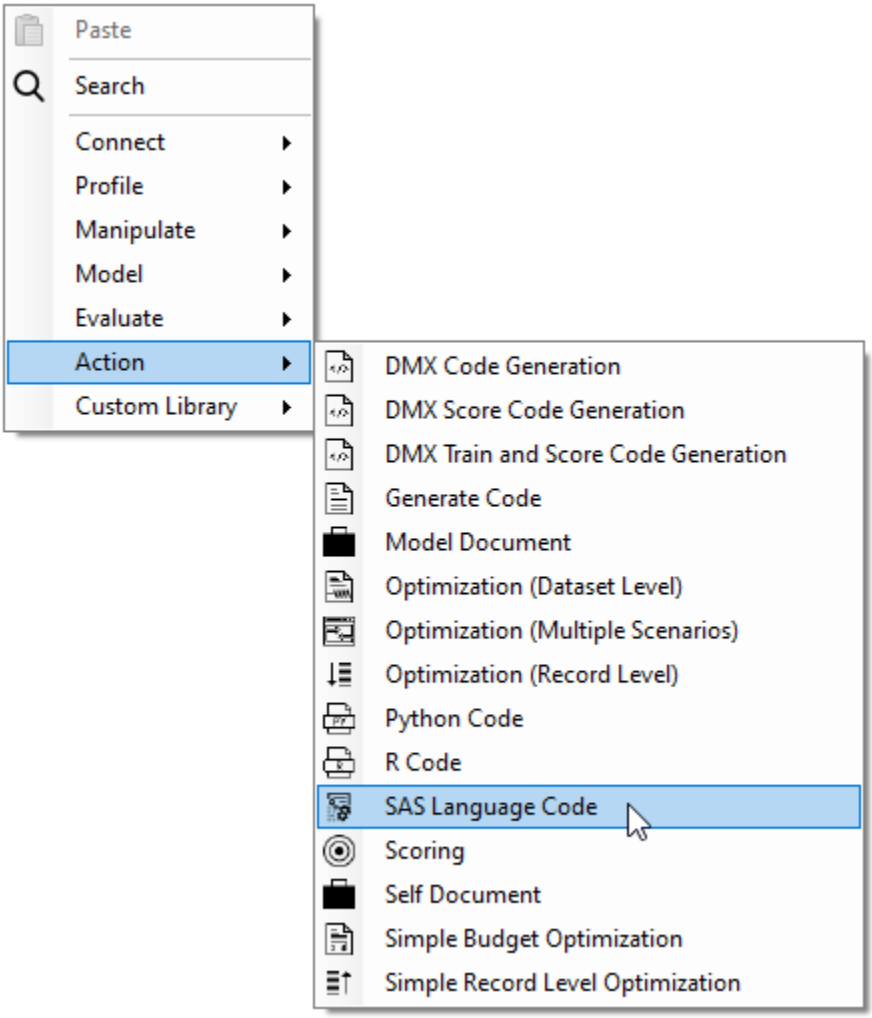
For importing data from other file formats or databases into an Altair KDD dataset, see the topic Accessing Data Sources.

Writing and executing programs in the SAS Language in Knowledge Studio workflows is powered by the World Programming System software (WPS). The WPS software package is included with the Knowledge Studio package.

To write and execute custom programs and macros in the SAS Language, use the **SAS Language Code** node in the **Action** palette:

- Right-click anywhere on the Workflow canvas and select the context menu command **Action --> SAS Language Code** as shown below. A new **SAS Language Code** node will be created, and the wizard will open.

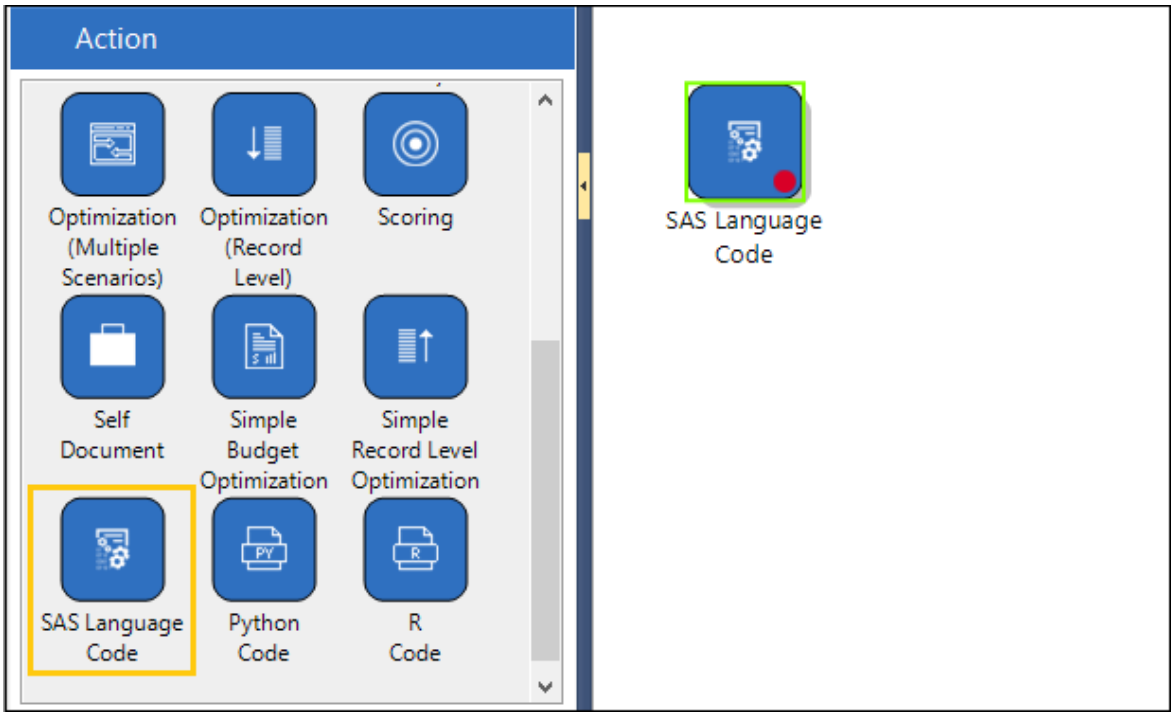
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



OR:

- Drag the **SAS Language Code** node from the **Action** palette onto the canvas. If the code you intend to write will use an existing dataset in the current project (a KDD, WPD, or SAS System file), connect the desired dataset node to the SAS Language Code node. Double-click the node or right-click on it and select **Modify** from the context menu.

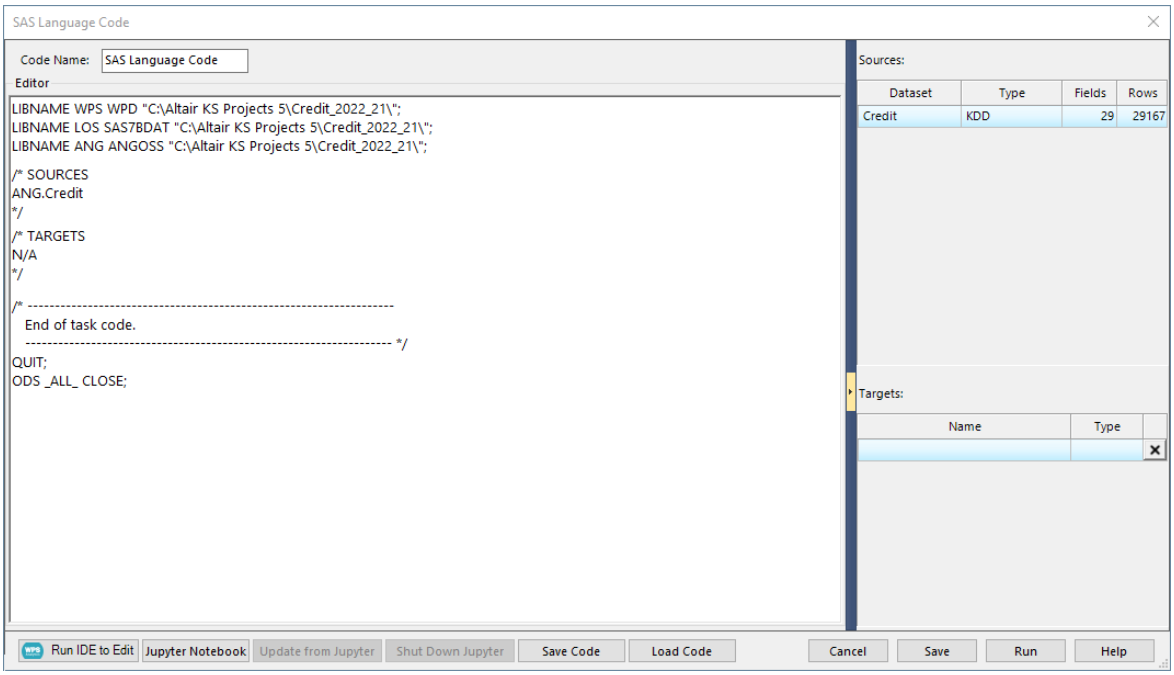
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



This opens the SAS Language Code window, where you can write and execute any program and macro in the SAS Language, launch a powerful IDE to edit the code, or launch Jupyter Notebook to code using WPS Analytics Jupyter Kernel.

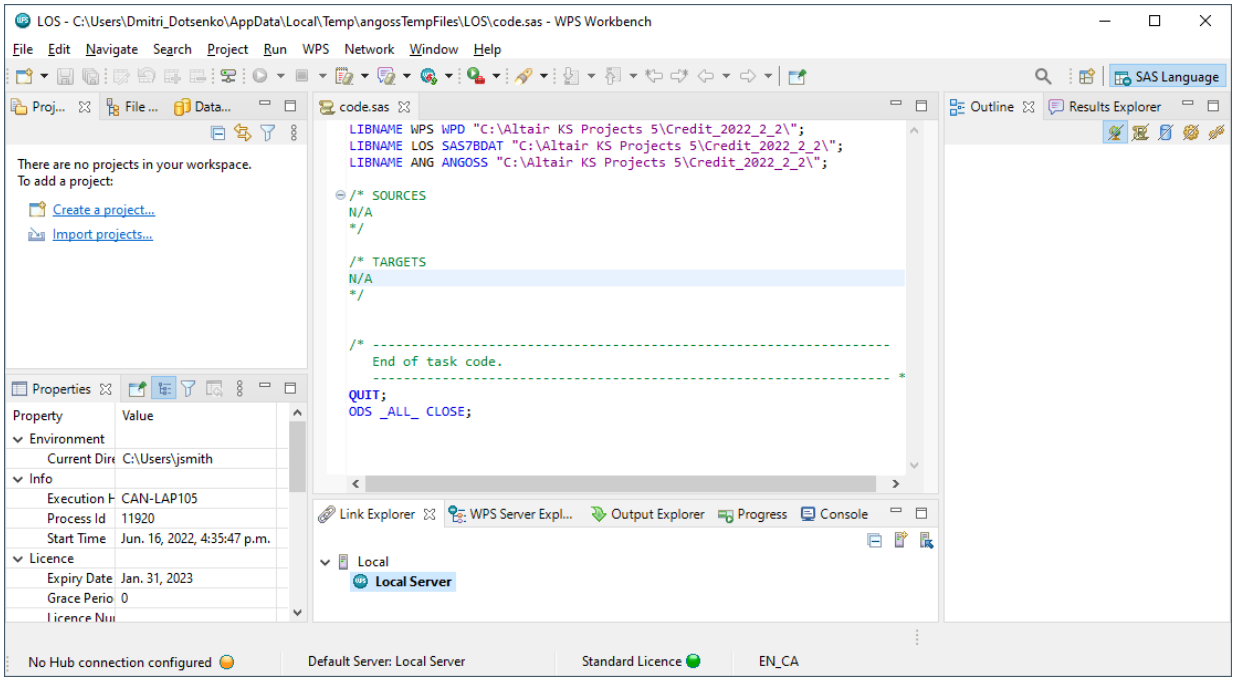
- The **Sources** section on the right-hand side lists the input datasets currently connected to the SAS Language Code node in the workflow.
- The **Targets** section on the right-hand side lists the output datasets to be produced by the SAS Language Code node in the workflow. Specify the names and types of the output datasets here, if necessary.
- Clicking the button **Run IDE to Edit** launches WPS Workbench - a powerful integrated development environment with rich functionality for additional code editing and validation.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The Sources & Targets pane on the right-hand side can be hidden by docking it to the right to expand the area for writing code. Click on the small arrow on the border between the Editor frame and the Sources & Targets pane to dock it. Click the arrow again to restore the pane.

Clicking **Run IDE to Edit** launches WPS Workbench in a separate window. Use the rich code editing features of WPS Workbench to write your code, then save and close it when finished. Your code updates will be automatically saved in the SAS Language code editor window.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. The datasets specified in the Targets section (if any) will be created on the workflow canvas as the outputs of the SAS Language Code node.



Supported Functions and Procedures of the SAS Language

For the complete list of functions, routines, procedures, and other language elements supported in the WPS version supplied with your Knowledge Studio product, see the language syntax description in the PDF document "*WPS Reference for Language Elements*" available at https://www.worldprogramming.com/support/wps/documentation

Jupyter Notebook Integration

The WPS Analytics Jupyter Kernel supports using the SAS Language in Jupyter Notebooks. The SAS Language Code node allows you to launch Jupyter Notebook, load your SAS Language code in the notebook session, and work with it using WPS Analytics Jupyter Kernel (if it is installed). For more information about Jupyter Notebook, see https://jupyter.org/.

Clicking the **Jupyter Notebook** button in the SAS Language Code node opens a Jupyter Notebook session in your default web browser. The code from the SAS Language Code node editor will be placed in the Jupyter Notebook window.

To use this feature, the path to the Jupyter executable must be specified in the configuration file **`KDSProvider.conf`**. Use any text editor to create a new file called KDSProvider.conf in the directory that contains the application binaries. The default location is:

- Windows Workstation (in the standalone configuration): `C:\Program Files\Altair Data Analytics\KS Workstation 2022.3.0\bin`
- Windows Server (in the client/server configuration): `C:\Program Files\Altair Data Analytics\KS Server 2022.3.0\bin`
- Linux: `/opt/Altair-Data-Analytics/KS-Server-2022.3/lib`

In KDSProvider.conf, specify the path to the Jupyter executable as a value of the jupyter_path parameter as follows:

```
jupyter_path <path>
```

For example:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Windows:

```
jupyter_path
"C:\ProgramData\Python3\Scripts\jupyter.exe"
```

Linux:

```
jupyter_path /opt/python3/bin/jupyter
```

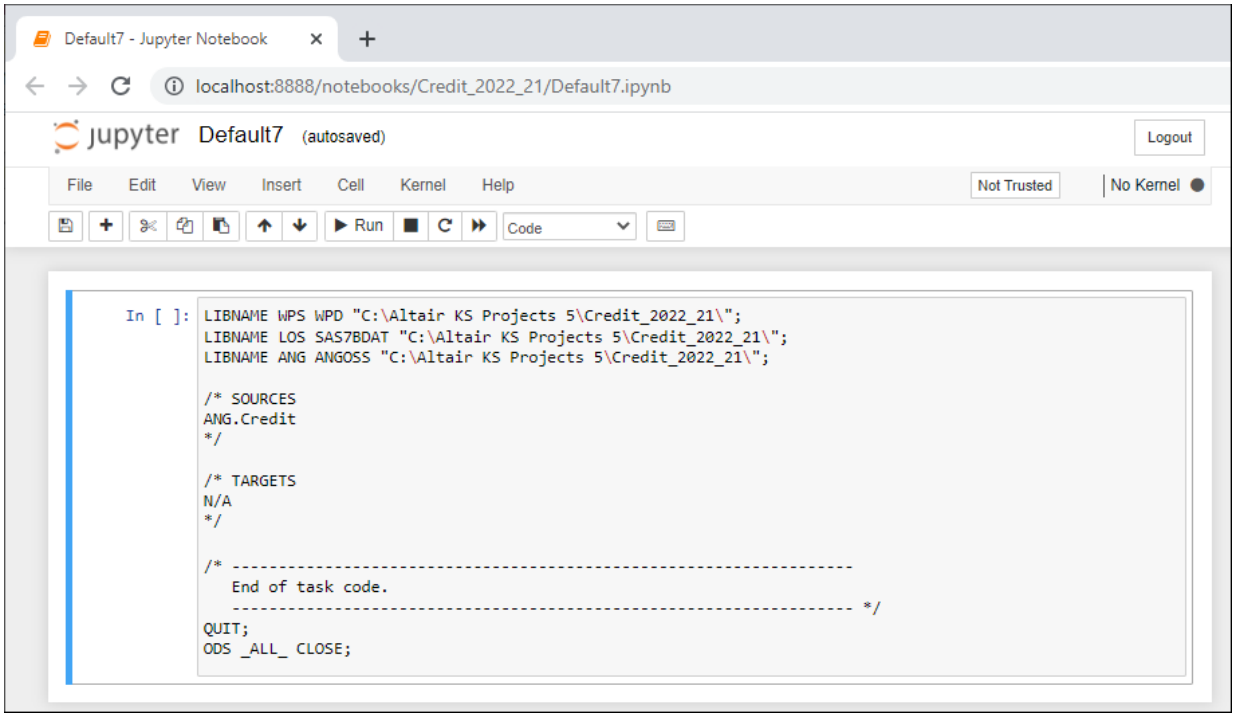
Save the changes and restart the application.

Open the SAS Language Code node and click the **Jupyter Notebook** button. This will open the Jupyter connection dialog.

Specify the following inputs:

- Jupyter server name. The default is *localhost* in the standalone mode, and http://<server_name> in the client/server mode.
- Port number. The default is 8888
- Notebook file name and path. The default is <current_project_folder>\Default.ipynb. Browse for the desired location and change the notebook name if necessary. If the specified notebook file already exists and is not empty, you will be prompted to replace the existing code in it.

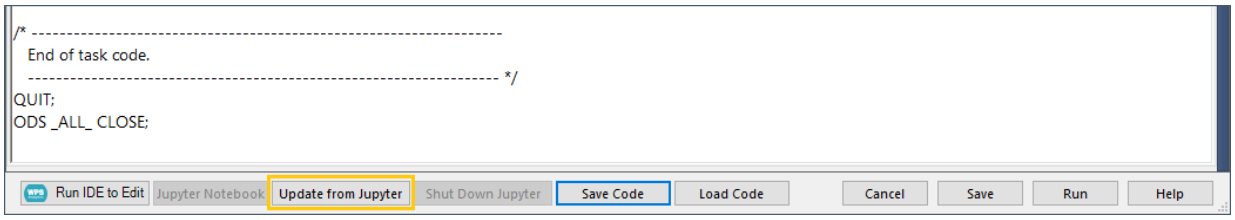
The Jupyter Notebook is launched in your default web browser, and the code from the SAS Language Code node is loaded in the new notebook session. Select the WPS Analytics Jupyter Kernel if it is not set by default.



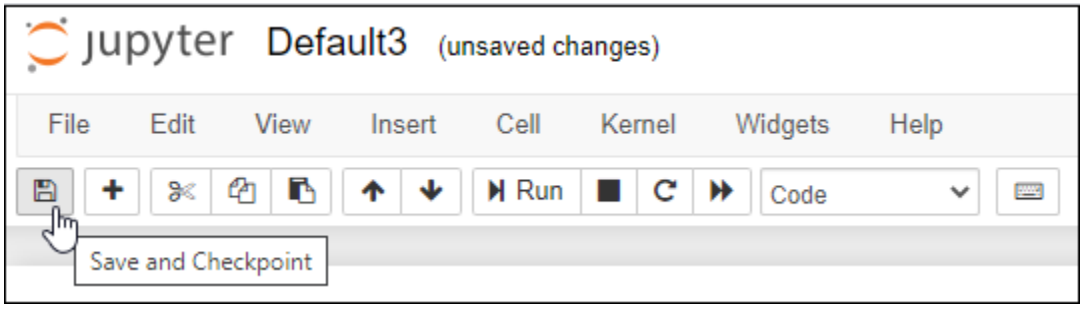
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Once Jupyter Notebook is launched, it creates notebook files (*.ipynb) in the project directory. The SAS Language Code node monitors the running notebook process.

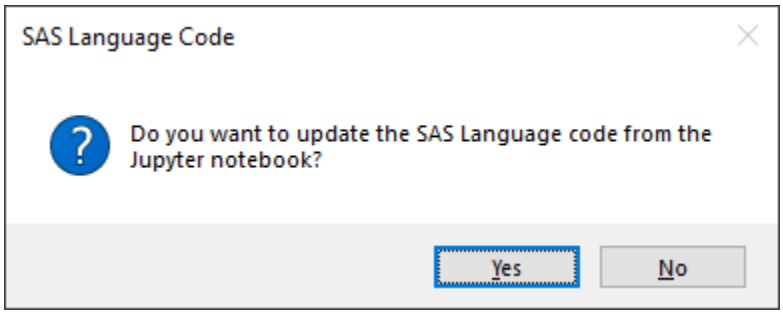
When Jupyter notebook opens in the browser, the **Jupyter Notebook** button in the SAS Language Code node becomes disabled, so you cannot launch another notebook session from node wizard. At the same time, the button **Update from Jupyter** becomes enabled, so that you could refresh the content of the SAS Language Code node editor from the linked notebook. Thus you can synchronize the code in the SAS Language Code editor and the notebook to view the code changes without having to click **Save** or **Run** (which would close the node wizard).



The button **Update from Jupyter** refreshes the code entered in the notebook up to the last checkpoint. If there is any unsaved code in the notebook, first click **Save and Checkpoint** in the notebook toolbar, then click **Update from Jupyter** in the SAS Language Code editor.



Once the notebook session is finished and you switch back to the SAS Language Code node, you will be prompted to update the code from the notebook.



Click **Yes** if you want to update the code, then continue working with it in the SAS Language Code node. The **Jupyter Notebook** button will become enabled again.

Code Editor Features

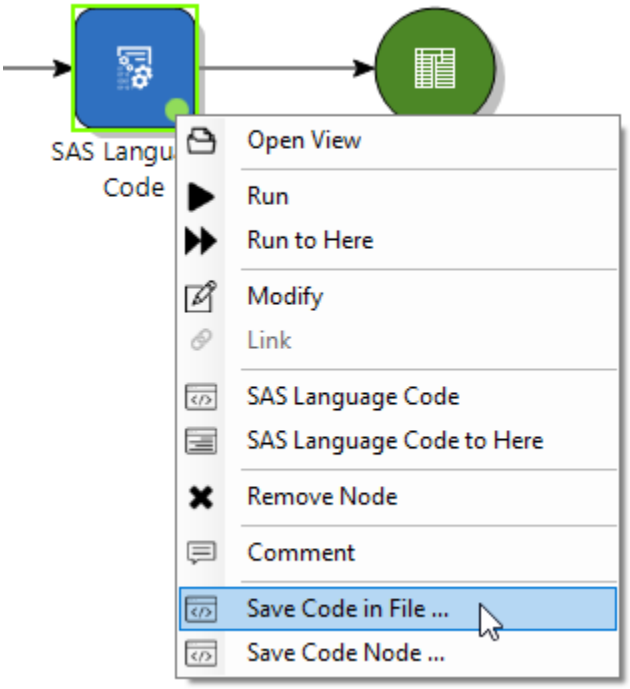
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

1. Saving code as a file

The code from any SAS Language Code node in your workflow can be saved as a file in two ways:

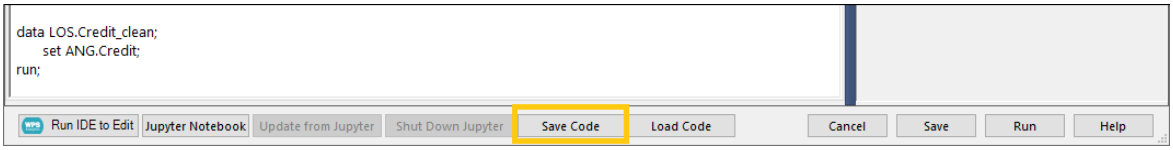
a) Using the context menu of the node:

Right-click on the SAS Language Code node and select the command **Save Code in File ...**



In the *Save SAS Language Code* window, browse to the desired destination folder and specify the name for the output file. By default, it is saved as SAS Language program type (with extension ***.sas***).

b) Using the **Save Code** button in the SAS Language Code editor window:



Click **Save Code** at the bottom of the SAS Language Code editor window.

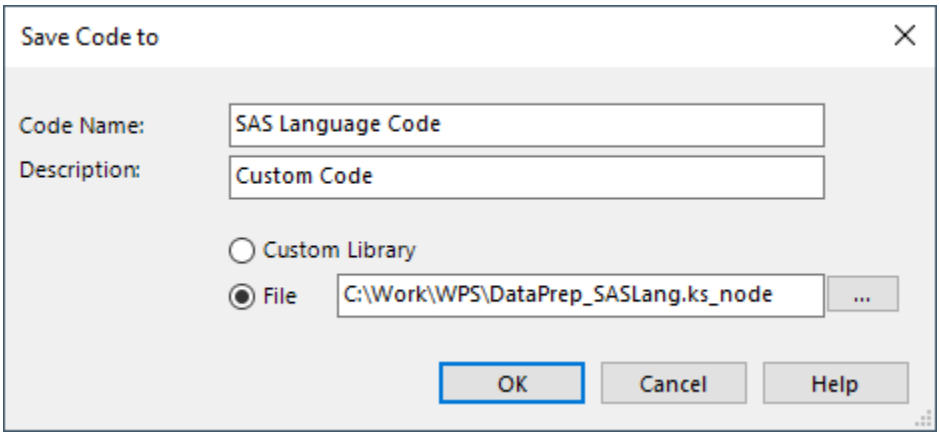
In the *Save Code* dialog:

- Select the **File** option, browse to the desired destination folder, and specify a file name. The default extension is ***.ks_node***. This is a re-importable XML file format that allows importing the code into any other SAS Language Code node using the **Load Code**

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

button. The code saved in this format can be easily shared with other Knowledge Studio users.

- Specify the code name and description. These attributes will be stored together with the code in the saved file.



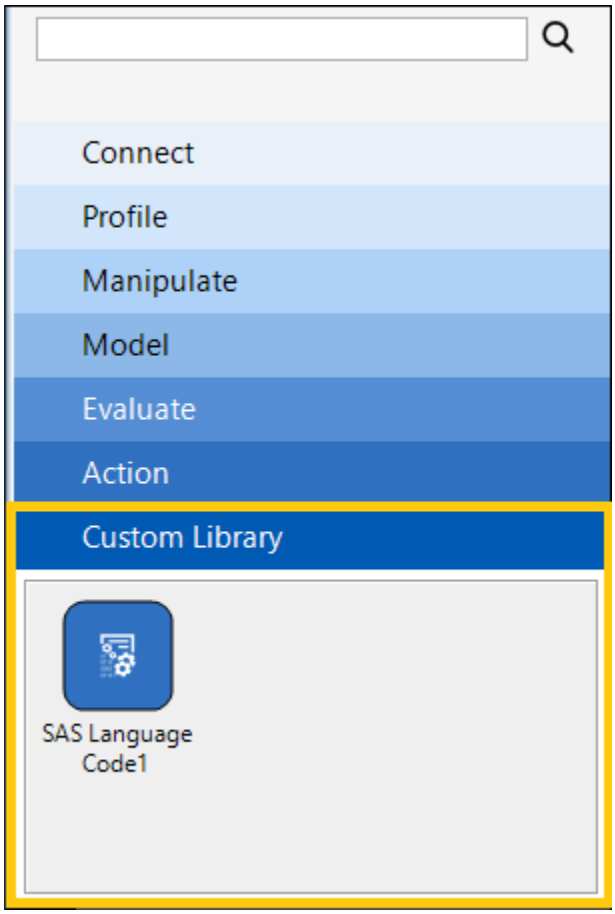
Click **OK**. You will get a message confirming that the code has been saved successfully. For the information on the **Load Code** feature, see the subsection *Loading code from file or from Custom Library*.

c) Using the **Save As** command in the right-click context menu of the Editor window or the **Ctrl+S** keyboard shortcut.

2. Saving code in the Custom Library palette

You can reuse the code from any SAS Language Code node by saving it as a new node in a separate palette - the **Custom Library** palette. Any custom code node saved in the Custom Library palette can be renamed or removed. *Note: The* **Custom Library** *palette is not shown unless it contains at least one node that was saved or imported previously*.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

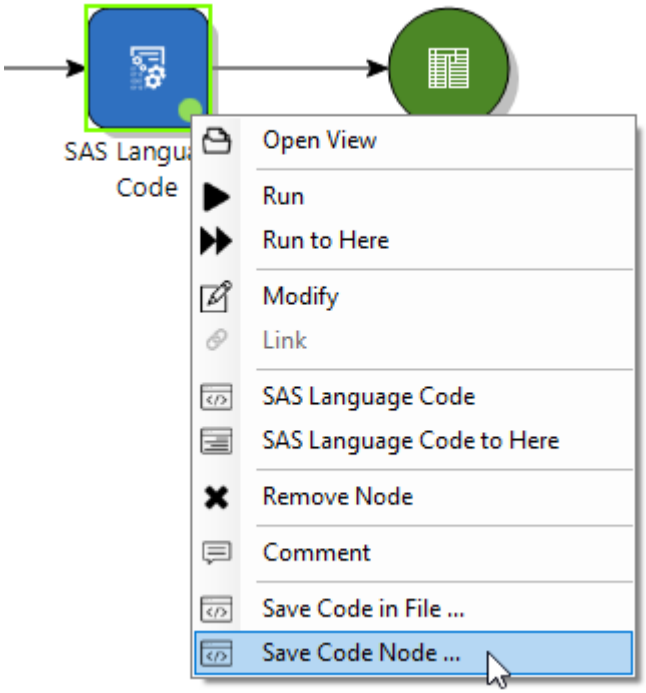


There are two ways to save a SAS Language Code node in your workflow under a new name in the Custom Library palette:

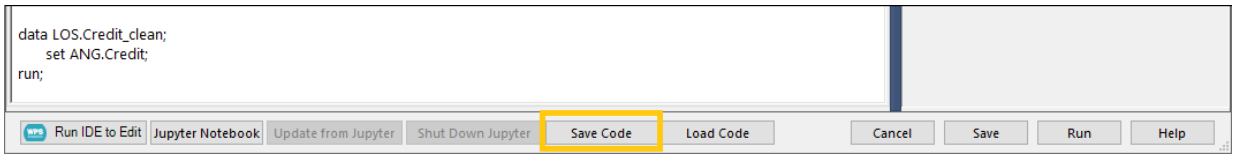
a) Using the context menu of the node:

Right-click on the SAS Language Code Code node and select the command **Save Code Node ...**

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



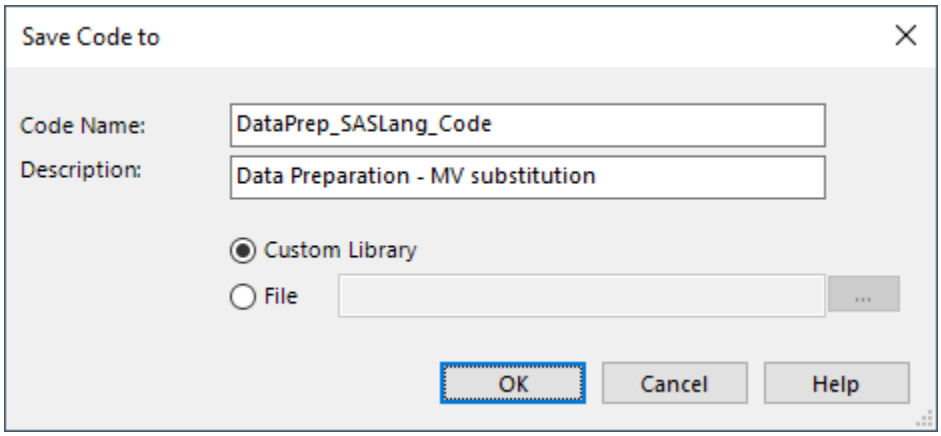
right-click on the code node on the workflow and select **Modify** to open the editor. Edit the code if necessary and click the **Save Code** button at the bottom of the editor window.



The "*Save Code to*" dialog opens.

- Select the **Custom Library** option.
- Specify the name for the new custom code node as it should appear in the Custom Library palette. The default name is that of the original node.
- Enter the description text (optional). The description will be shown as the tooltip when hovering over the new custom code node in the Custom Library palette.

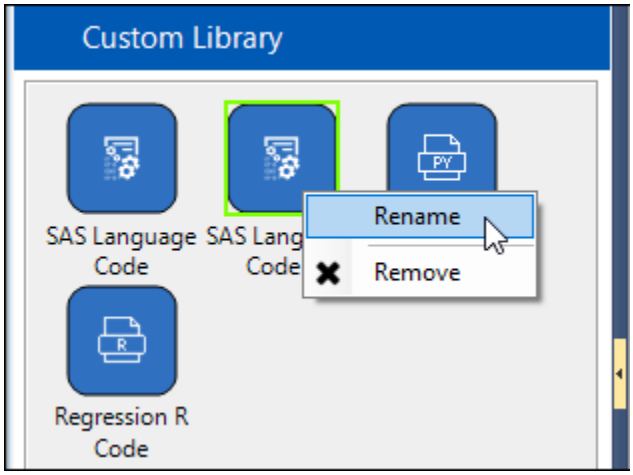
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **OK**. The new custom code node will be placed in the Custom Library palette.

> *Note: The code is saved "as is". When reusing the custom code nodes from the Custom Library palette, check the code for the dependencies on the source and target datasets in the new project context, link source dataset(s), specify target(s) and modify the code appropriately.*

To rename or remove any node in the Custom Library palette, right-click on the node and select **Rename** or **Remove** from the context menu:

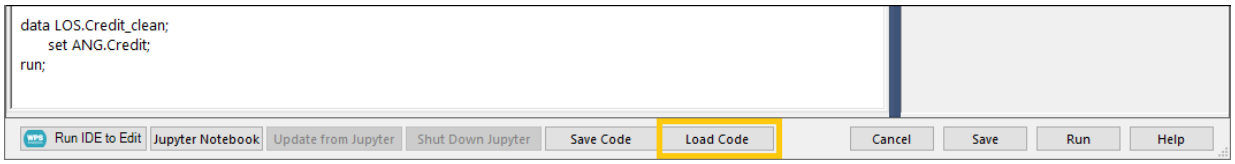


3. Loading code from file or from Custom Library

The **Load Code** button in the SAS Language Code editor window loads previously saved custom code from the **Custom Library** or from a ***.ks_node** file (see above). Using this function, users can reuse and share their code as custom SAS Language Code nodes. For example, they can be exchanged via email, loaded from disk, then saved to the Custom Library palette if desired.

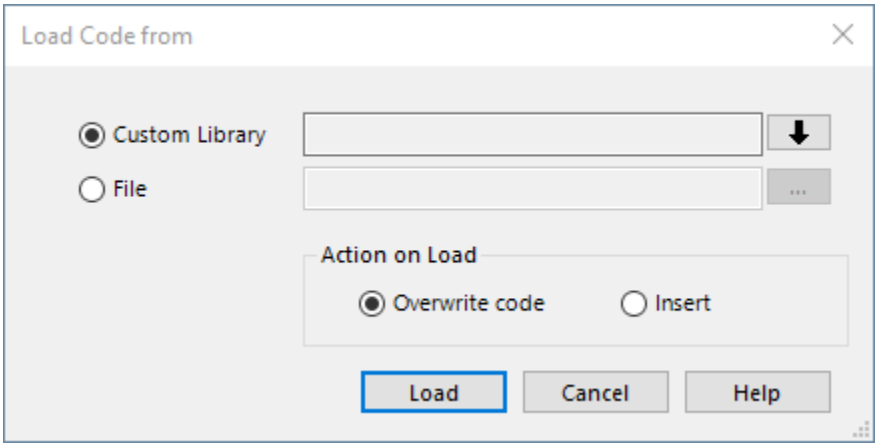
Add a new SAS Language Code node to your workflow or open an existing one on the workflow canvas. In the SAS Language Code editor window, click the **Load Code** button:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The "*Load Code from*" dialog opens.

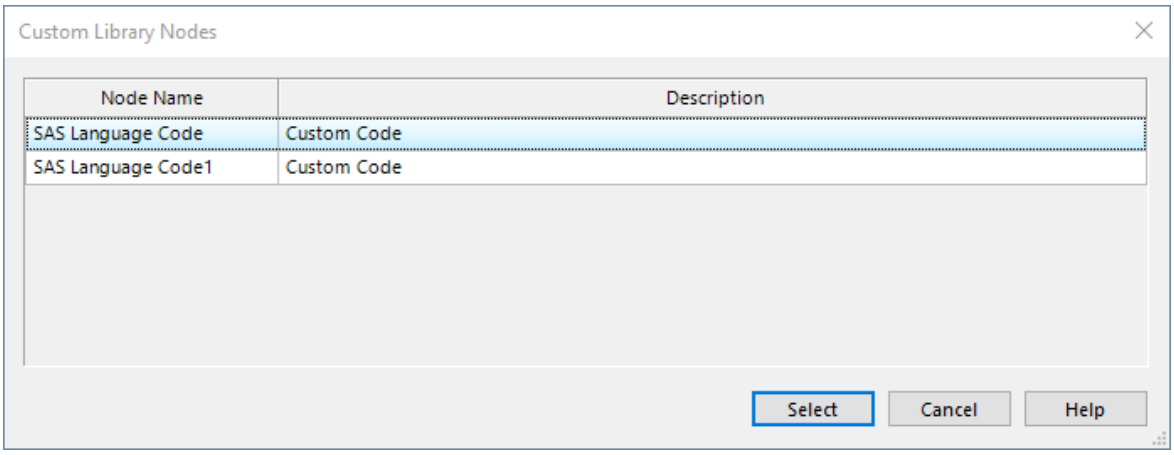
- Select the **Custom Library** option if you want to load code from one of the nodes in the Custom Library palette.
- Select the **File** option if you want to load code from a file. The source file must be in the "custom node" file format (with extension ***.ks_node**) - a re-importable XML file format used by the **Save Code** feature. Browse for the source *.ks_node file, select it and click **Open**.
- Choose whether to **overwrite** any existing code in the editor window or to **insert** the new code at the current cursor position.



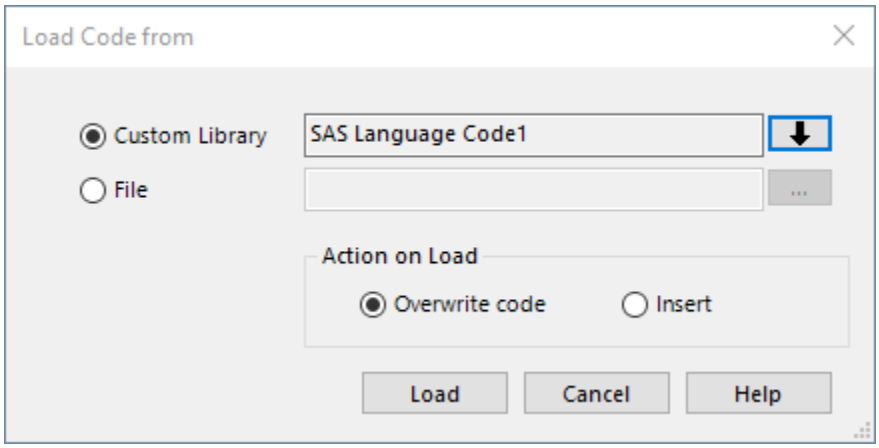
If you selected the **Custom Library** option, click the down arrow on the right to open the Custom Library Nodes list. It displays the custom code nodes available in the Custom Library palette. Choose the desired source node click **Select**.

> Note: The default location for the Custom node templates in the file system is the folder `%APPDATA%\Altair\CustomLibrary` of the current user profile. When the application is started, this folder is scanned for valid templates, which are then displayed as nodes in the Custom Library palette.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



This brings you back to the "*Load Code from*" dialog. Click **Load** to load the code from the selected source.



Note: the code is loaded "as is". When reusing the code, check the code for the dependencies on the source and target datasets, link source dataset(s), specify target(s) and modify the code appropriately.

Limitations

ODBC Import from Oracle:

ODBC import using the **Import ODBC** node or a script in the **SAS Language Code** node will fail for any Oracle table that has lowercase characters in the table name or in any column name. If you need to import data from Oracle using any of these nodes, make sure the source table or view has only uppercase letters and no special characters in its name and its column names.

In particular, the problem may occur when importing a table originally created by exporting data to Oracle from Angoss or another application. If the export query uses quotes around the table name or a column name, the name is created as is (e.g., lowercase letters and special characters are preserved); otherwise, Oracle converts table and column names to uppercase.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

See also: R Code Node, Python Code Node

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Integration with R and Python

R Code Node

Prerequisites

R Software

Integration is supported with 64-bit R versions 3.5.0 and higher. For the Standalone (desktop) configuration of Knowledge Seeker or Knowledge Studio, R must be installed on the computer(s) that host KS Workstation. For the Client/Server configuration, R is only required on the server side.

- For Windows desktop: If R is not installed, download the latest version of R from https://www.r-project.org/ unless you are required to install a specific version of R approved in your organization.
- If you are using the client/server configuration for Linux or Windows Server, please contact your server administrator regarding the installation of R on the server.

Rserve package

The Rserve package must be installed in the common R library location, which depends on the R installation directory. On Windows, the default R library folder is usually `C:\Program Files\R\R-x.x.x\library`, where x.x.x is the R version. On Linux, it is usually `/usr/lib64/R/library`. For KS Server on Linux, R home directory must be specified by the system administrator in the KS Server startup script `ks_start`.

For the desktop configuration of Knowledge Studio or Knowledge Seeker, follow the steps below. For the client/server configuration, please ask your server administrator to install the Rserve package on the server using the instructions in the Deployment Guide.

1. Make sure you have Read & Write permissions in the common R library folder. Open your file browser, navigate to `C:\Program Files\R\R-x.x.x\`, right-click on the **library** folder, select *Properties*, go to the Security tab, click **Edit**, and add the **Write** permission if it is not already there.

2. Run the R console. In the R command line, run the following command, where R-x.x.x is your version of R:

```
install.packages("Rserve", "C:/Program Files/R/R-x.x.x/library")
```

You will be prompted to select an R Archive Network site to download the package. Choose any suitable mirror and click OK. The package will be downloaded from the selected site and installed.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

KS Library for R

R integration also requires ***KS Library for R.*** The KS Library packages for different versions of R are available on the Altair software download site. The complete requirements and installation instructions are provided in the deployment guide available at the Altair downlioad site. Quick installation instructions are also given in the License Administrator's email that has your registration key.

- For Windows platforms:
  - For R version 4.0.0 or higher, you need KS Library for R 4.0.0 or higher.
  - For R version 3.5.0 - 3.6.3, you need KS Library for R 3.5.0 - 3.6.3.
- For Linux:
  - For R version 3.5.0 - 3.6.3, KS Library for R on Linux is included in the KS Server package. The installation instructions are provided in the Deployment Guide.
  - For R version 4.0.0 or higher, KS Library for R 4.0.0 or higher on Linux is provided separately. The download and installation instructions are provided in the Deployment Guide.

Feature Overview

Knowledge Seeker and Knowledge Studio integrate with R software environment for statistical computing. R language integration enables embedding R programs into Altair workflows. Users can write R code within the **R Code** node in workflows and use its outputs in further operations within the same workflows. Data outputs of R programs embedded in the workflow can be used as regular datasets. Thus, all the versatile data manipulation functions and data science algorithms of R can be used within Knowledge Seeker and Knowledge Studio.

The main benefits are as follows:

1. The native Knowledge Seeker and Knowledge Studio functionality is expanded by means of R functions and algorithms. All the data manipulation functions and data science algorithms available in the base R distribution and its numerous libraries can be used within these Altair applications. Load any desired R packages representing any algorithms you need and write R programs within your workflows, then use the outputs as regular datasets.

2. Workflow continuity: Users of R do not need to switch to R and back when working on their projects in Knowledge Seeker or Knowledge Studio. For instance, if you prefer to use R for data preparation, instead of exporting your dataset to R, writing an R program transforming the data, generating a new R dataset and importing it back into the Altair application to continue modelling, you can write the data transformation code in the **R Code** node within the workflow and use its outputs in further workflow operations.

3. Decision Trees and Ensemble Tree models can be automatically translated into R programs using the code generation feature.

4. Workflows prepared with the use of R can be translated into production-ready deployment code in the SAS Language, SQL, or PMML

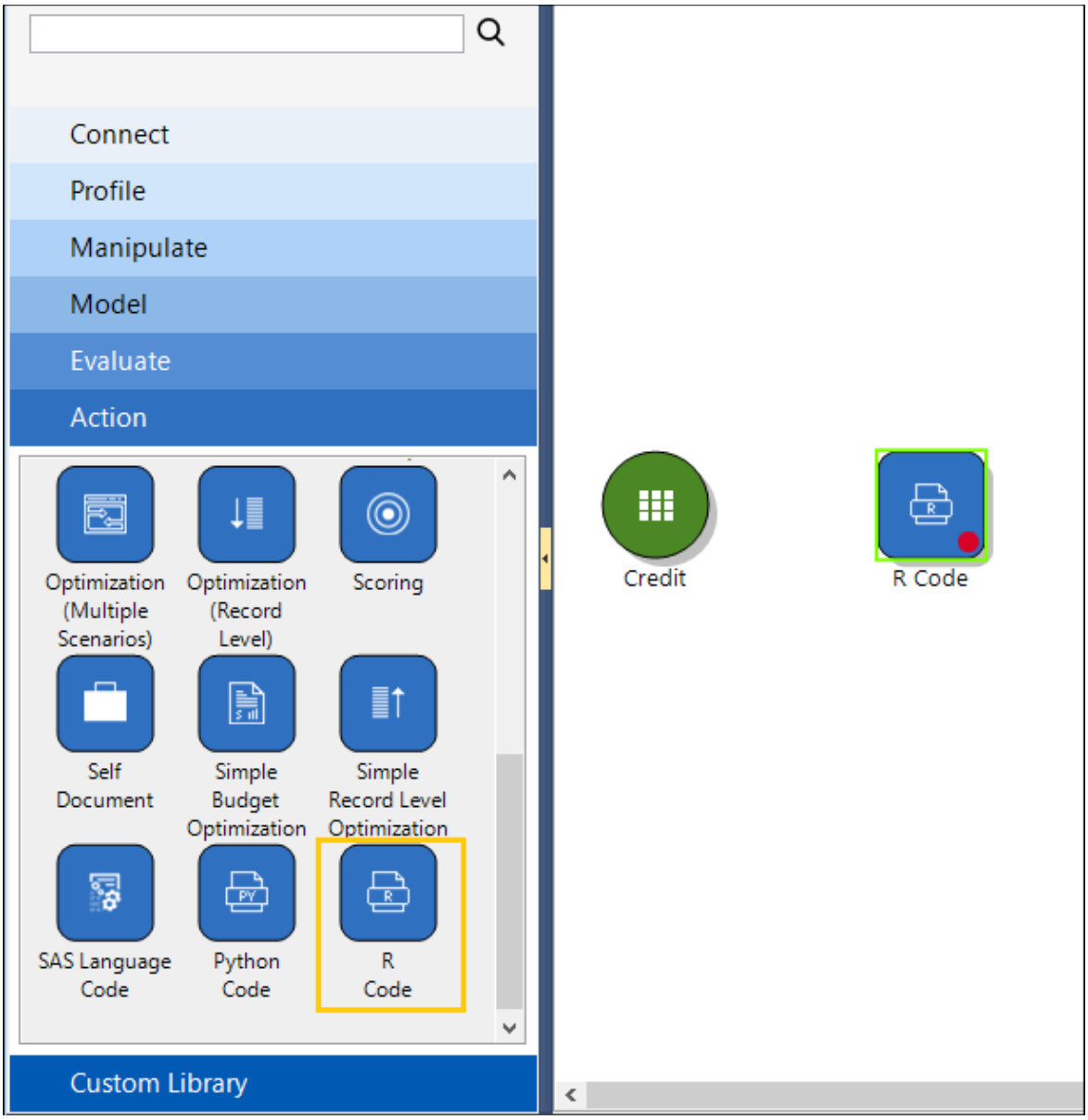
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Embedding R Programs in Workflows Using the R Code Node

For information about projects and workflows, see the topics Projects and Workflows.

> NOTE: *Because of the **Rserve** package limitations, the **R Code** node does not work if the Working Directory path or the project name contains non-ASCII characters such as Chinese, Cyrillic, Arabic, or other characters whose encoding requires more than 8 bits. If your Working Directory path or the project name contains such characters, you will get an R server connection timeout error when running the R Code node. To resolve this, rename your project using only Latin characters and move it to a folder whose path contains only ASCII characters.*

Open the **Action** palette and drag the **R Code** node onto the Workflow canvas. If the code must use any datasets on the workflow, connect these dataset nodes to the R Code node. Double-click the R Code node or right-click it and select **Modify**.

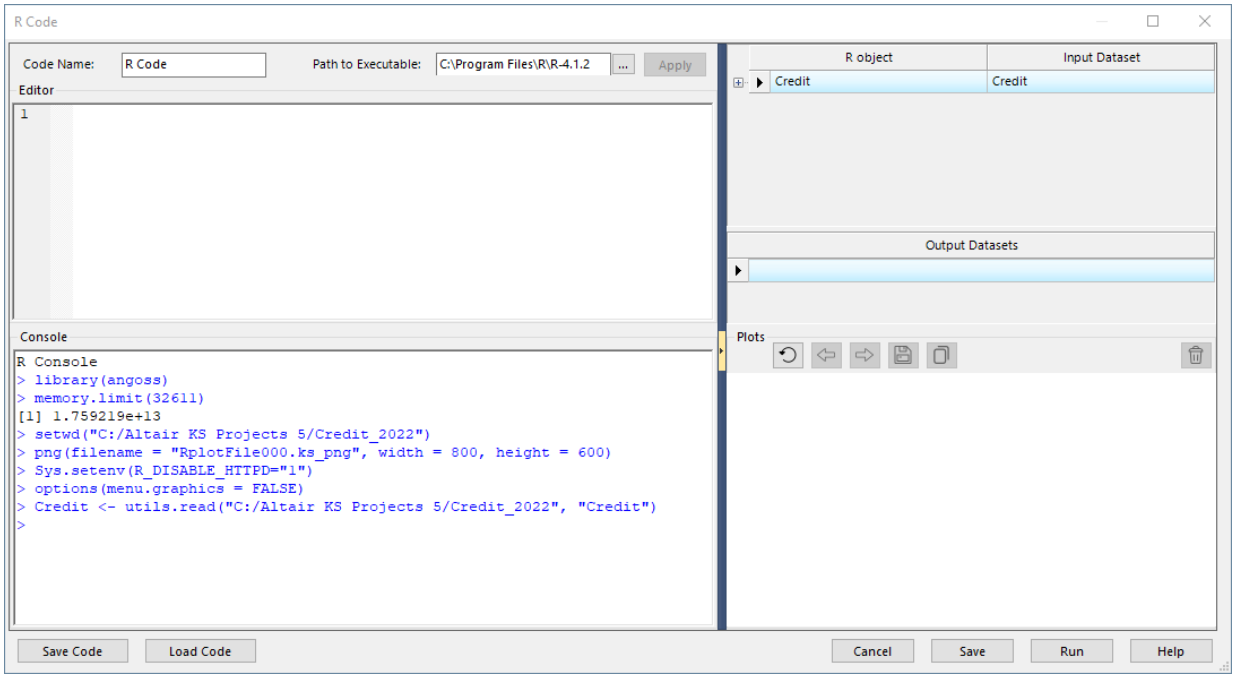


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The R Code editor window opens.

- In the **Code Name** field, specify the name for the R Code node - the label to be displayed under the node on the workflow canvas.
- **Path to Executable** shows the location of the R **bin** folder where **`R.exe`** resides, as well as the location of the R **library** folder.
    - In the Standalone configuration, the default path is defined by Environment Preferences. In the standalone mode, you can change this path in the R Code node at any time (for example, if you have two versions of R and would like to use a different one). After changing the path, click **Apply** for the change to take effect. The change can also be applied by clicking **Save** and reopening the R Code node.
    - Update the default path in Environment Preferences if you change your R environment (e.g., after upgrading R) or if you need to use a different version of R when two or more versions are installed.
    - In the Client/Server configuration, Path to Executable is read-only. It is defined by the configuration on the KS Server host machine. Please contact your server administrator if you need to change the R environment used by KS Server.

Start writing your R program in the R Code editor as you normally would in R or R Studio.



Several lines of initialization code are automatically executed in the Console frame when the node is opened. Here is an example of the code displayed in a new R Code node with one input dataset:

```
> library(angoss)
> memory.limit(32611)
[1] 1.759219e+13
> setwd("C:/Altair KS Projects/Credit")
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
> png(filename = "RplotFile000.ks_png", width = 800, height =
600)
> Sys.setenv(R_DISABLE_HTTPD="1")
> options(menu.graphics = FALSE)
> Credit <- utils.read("C:/Altair KS Projects/Credit",
"Credit")
>
```

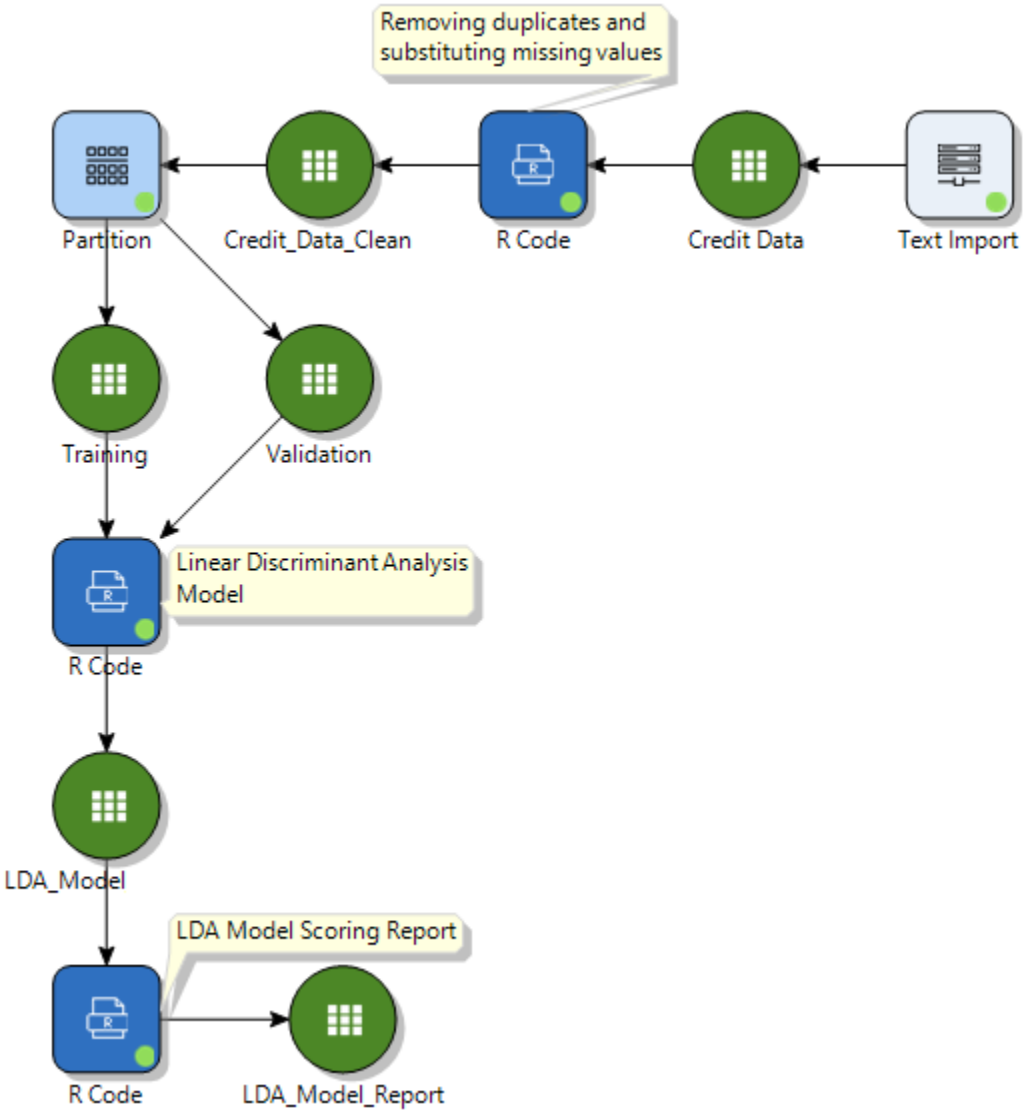
First, the command `library(angoss)` loads the integration library. The `setwd` command sets the R workspace to your current project folder. The next three commands set several environment options. The `utils.read` command reads the input datasets into R objects.

- To drop any dataset, use the `rm()` command, which deletes the R object in the argument.
- To export datasets, use the **`write.table`** command, for example, `write.table(census, "C:\\Data\\census.txt")`
- To load any dataset from the same project or any other KDD datasets from other folders, use the **`utils.read`** command as in the example above.

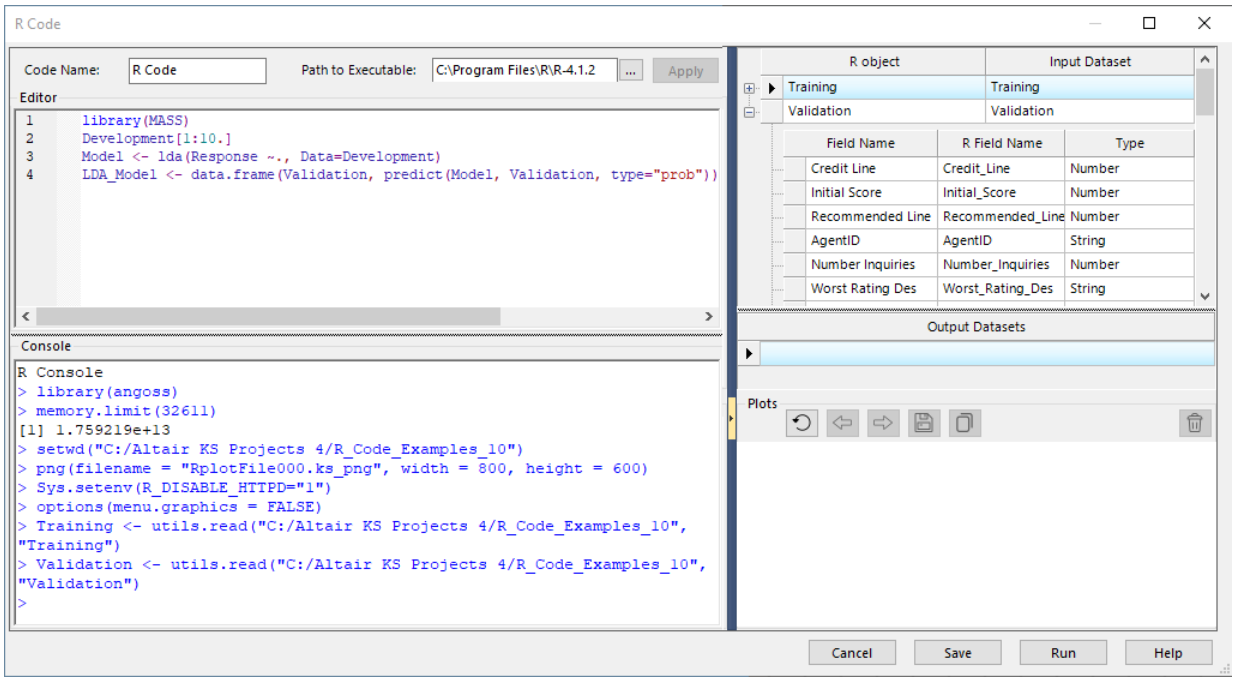
The inputs and outputs of the R Code node are listed in the panel on the right-hand side. Note that the node does not produce any output dataset nodes on the workflow unless you explicitly specify their names in the **Output Datasets** section of the right pane. The names of any intermediate datasets created in your code with no intention of using them in the subsequent workflow do not need to be specified there.

In the example below, the input datasets for the R Code node are Validation and Development. The output dataset is called *lda_Model*. Note that object names are case-sensitive.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.



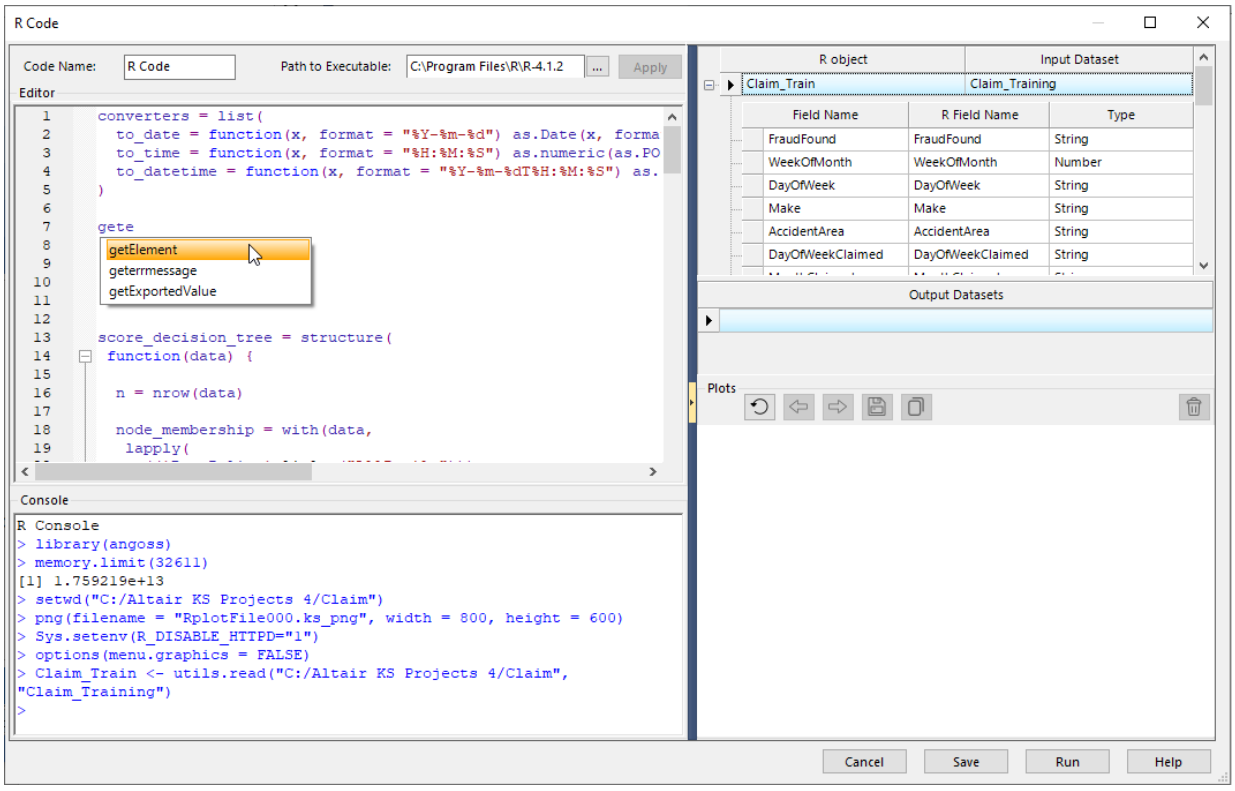
Code editing tools and enhancers

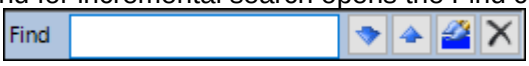
The R Code editor window provides basic IDE features to enhance code editing:

- line numbers
- collapsing / expanding blocks of code
- syntax highlighting displays code elements in different colours and fonts according to their category
- the auto-complete function predicts and suggests the commands as you type (after typing at least two characters); when multiple suggestions appear in a list, select the desired one with single mouse click or using the arrow keys and pressing Enter.

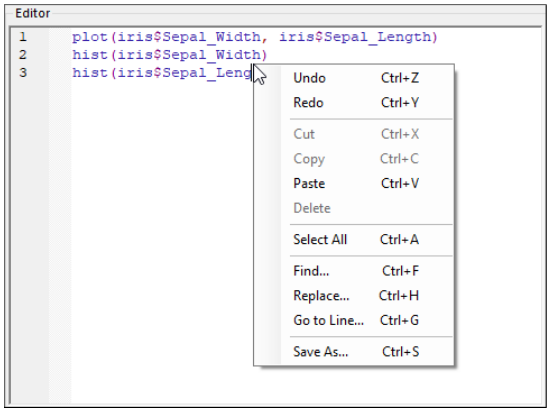
To execute the code entered in the Editor frame, highlight the lines to be executed and press CTRL+Enter or CTRL+R on the keyboard. The commands will be executed in the R Console frame. If none are highlighted, by default, the line where the cursor is placed will be executed.

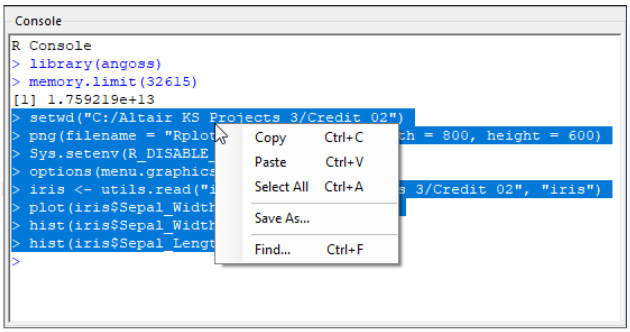
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- **Undo**, **Redo**, **Find**, **Replace**, and **Go to Line** functions are available in the context menu and with keyboard shortcuts in the Editor frame.
- The code in the Editor frame and Console frame can be saved as a file using the **Save As** command in the context menu.
- The keyboard shortcuts are common for business applications (Ctrl+Z and CTRL+Y for Undo and Redo; Ctrl+F and Ctrl+H for Find and Replace; F3 and Shift+F3 for Find next / Find previous; Ctrl+G for Go to line; Ctrl+C and Ctrl+V for Copy and Paste; Ctrl+X for Cut; Ctrl+A for Select All).
- The **Ctrl+I** command for incremental search opens the Find control at the top-right corner of the Editor window: 

. Enter your search string, then use the Next/Previous and the Highlight All Matches buttons in this control. To exit the search, just click anywhere else within the R Code window.

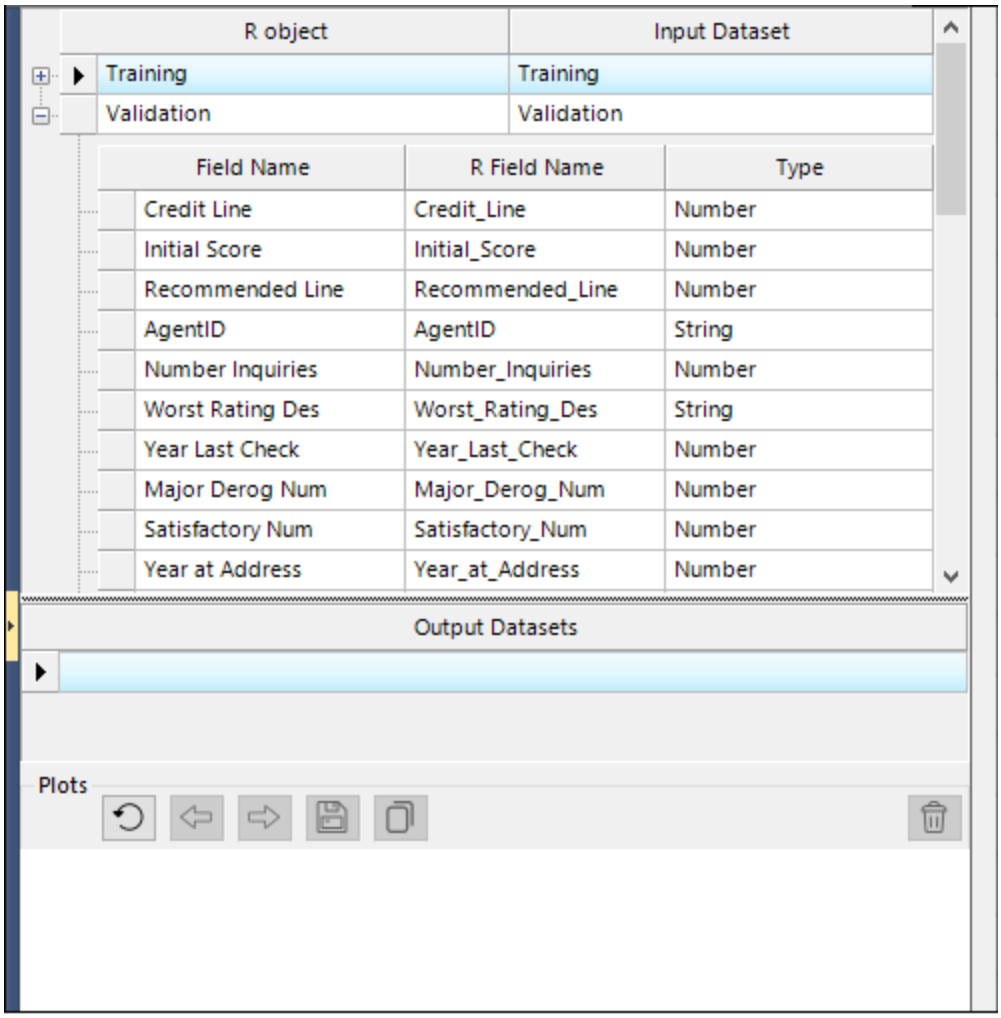




Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The right panel of the R Code window can be hidden by docking it to the right whenever necessary to save some space for writing code. For this purpose, click on the arrow on the border between the Editor and Console frame and the Inputs and Output Datasets frame.

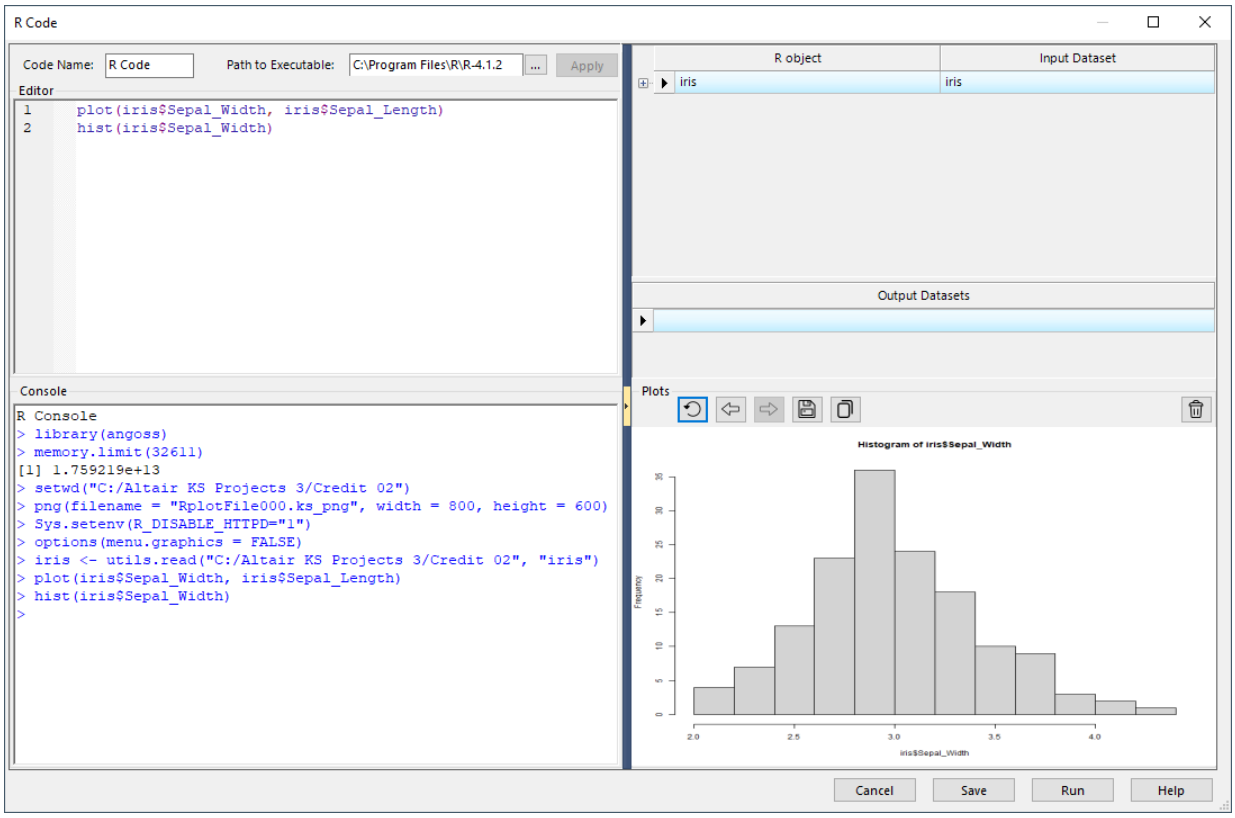
Expand the Input dataset item to view its fields (variables). Note that the naming convention for R objects is more restrictive than the one for native objects. For example, spaces are not allowed in R. By default, R object names are the same as native object names, but native object names longer than 12 characters are truncated, and all characters that are not allowed in R are replaced with periods. The field names in the R object that the input dataset is converted to are displayed in a separate column *R Field Name*. In the example below, spaces in the original field names are replaced with underscores in the R object.



Viewing R Plots

To view the R plots, histograms, or other graphs that you create in your code, use the **Plots** frame in the right panel of the R Code window. After executing a command that creates a plot, click the Refresh button  at the top of the Plots frame to view the result.

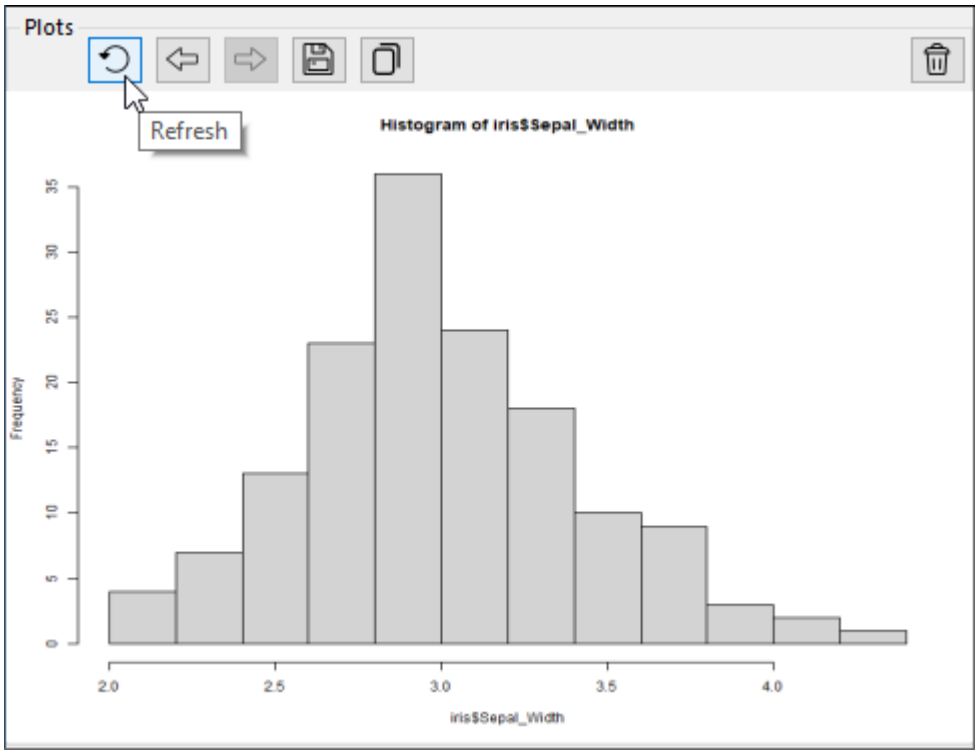
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Use the toolbar at the top of the Plots frame to manage the plots created in your code:

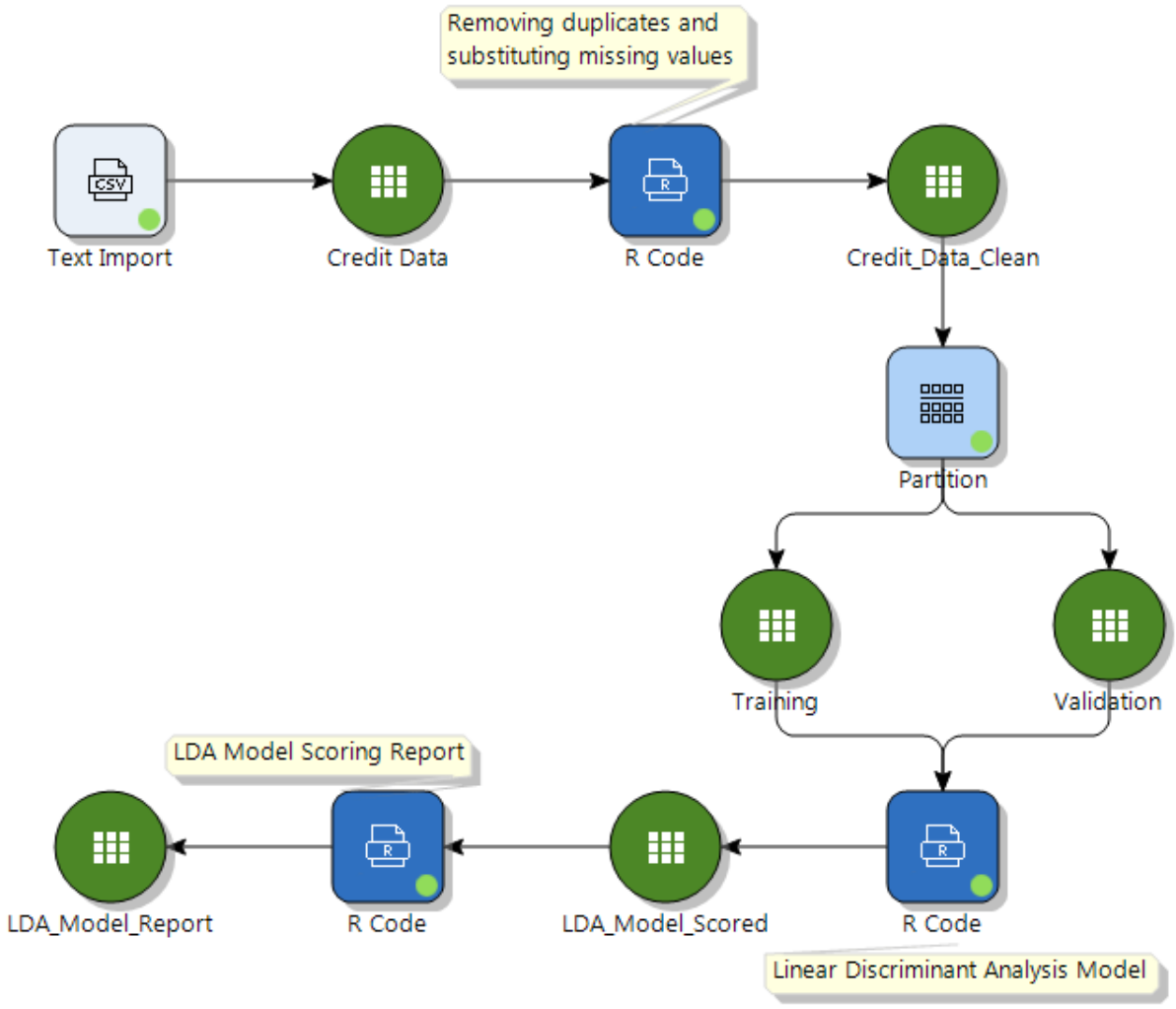
- After creating new plots in your code, click Refresh  to view the latest plot.
- Use the back and forward arrow buttons   to view the previous plots.
- Use the Copy button  to copy the plot to the clipboard.
- Use the Save button  to save the plot as an image file (PNG format).
- To clear all plots from the memory, click the recycle bin button  on the right.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Here is an example of a workflow with embedded R programs. Here the first R Code node performs data transformation, the second builds a predictive model (linear discriminant analysis), and the third performs reporting functions (builds plots for the model scoring result dataset).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Note: All R code that you write in an R node will be saved even If you disconnect all inputs and outputs.

Specific Features and Limitations of the R Code Node Interface

- Getting R help using the **?** command (for example, **`?glm`**) within the R Editor or Console will produce the warning messages "httpd server disabled by R_DISABLE_HTTPD; HTML help is unavailable ". If you execute the same command again, Help will appear in a new window, but it can only be closed by closing the R Editor and Console window.
- Read commands in R programs will not produce a new KS dataset. You need to use the import nodes instead. For example, the **`read.csv`** command **`Data_new <- read.csv("C:\\Data\\new_accounts.txt", header=T)`** would import a text file into an R object, but would not create a new KS dataset "Data_new".

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- New fields can be added in an existing R object, but these fields would not automatically show up in the corresponding dataset. For example, the command **`credit$X1 <- credit$field1 * 2.5`** generates a new field called [X1] in the credit dataset object, but you have to explicitly specify the new output dataset in the Output Datasets section of the R Code window to actually produce the dataset with the new field.
- Because of the **Rserve** package limitations, the R Code node does not work if the Working Directory path or the project name contains non-ASCII characters such as Chinese, Cyrillic, Arabic, or other characters whose encoding requires more than 8 bits. If your Working Directory path or the project name contains such characters, you will get an R server connection timeout error when running the R Code node. To resolve this, rename your project using only Latin characters and move it to a folder whose path contains only ASCII characters.

See also:

Survival Analysis

Python Code Node

Survival Analysis

Survival Analysis deals with predicting time-to-event data. A standard example of an "event" in business is customer churn. Survival Analysis also has applications in insurance, quality control, health sciences, and many other areas.

The **Survival Analysis** node in the **Model** palette uses integration with *R* software for statistical computing. The **Survival Analysis** node is enabled in the **Model** palette. For details about *R* language Integration, see the R Code Node topic. The prerequisites for the Survival Analysis node are the same as those for the **R Code** node.

The function uses open-source *R* packages for survival analysis:

- "***survival***" is the standard *R* package for survival analysis used as the code template for the Survival Analysis feature.
- the "***xtable***" package
- the custom package "***KnowledgeSurvival***" developed by Altair

KS Library for *R*, which enables *R* Language integration, installs all three packages listed above. The KnowledgeSurvival package is installed first. The "survival" and "xtable" packages are installed next from https://cran.r-project.org/. If any of the packages fail to install, this will be recorded in a log file. Users can install them separately from the **R Code** node available in the **Action** palette.

> ***Note***: If more than one version of *R* is installed on the host machine, the Survival Analysis node uses the most recent version. To run the model in a different version of *R*, copy the code from the Survival Analysis node wizard to a new *R* Code node and specify the path to the desired R version there.

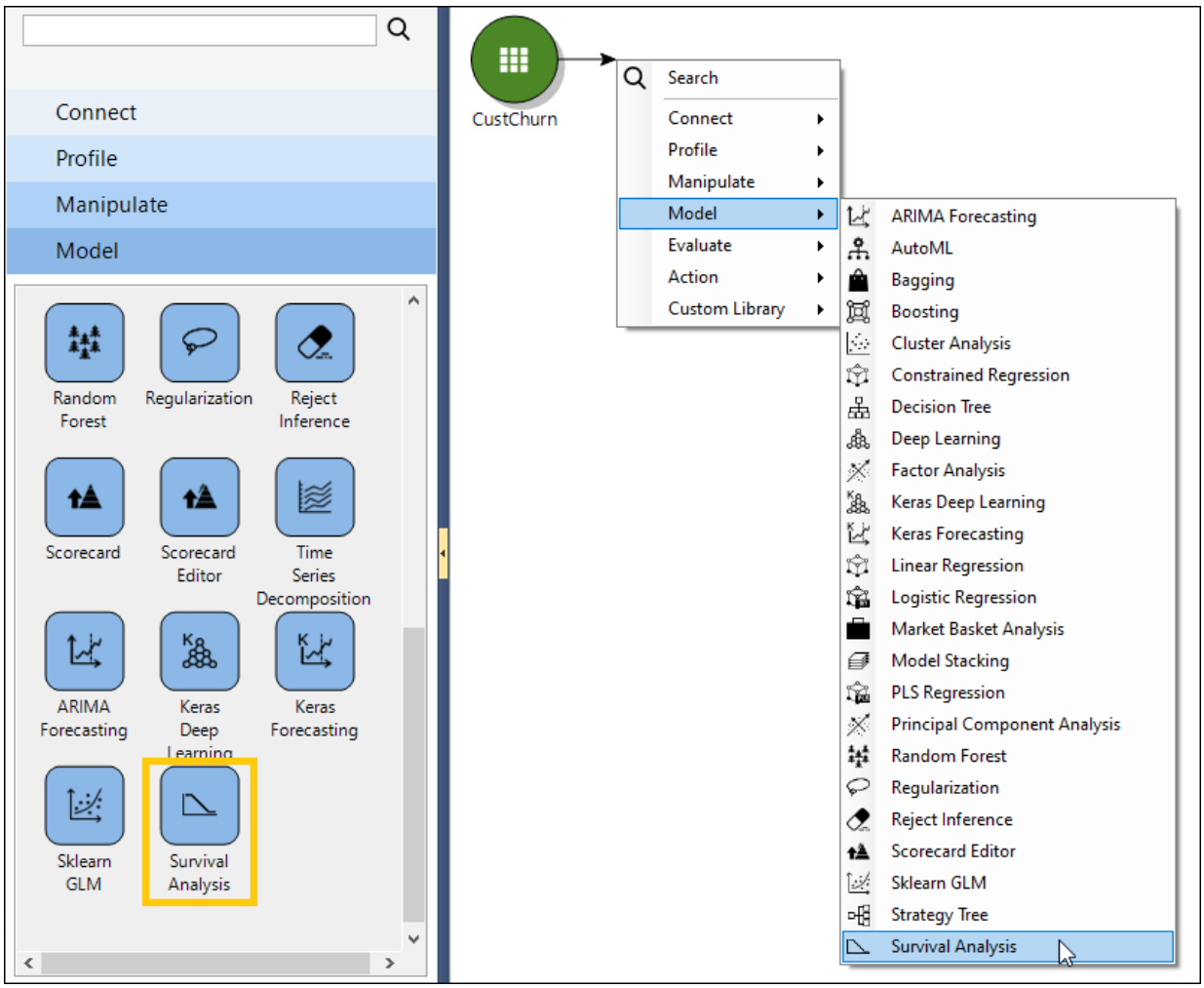
To insert a Survival Analysis operation in your workflow:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

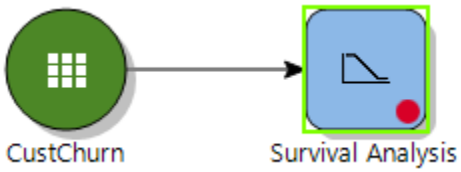
- Drag a connection arrow from a dataset node to any area on the Workflow canvas background. A context menu opens when you release the mouse button. From the context menu, select **Model --> Survival Analysis** as shown in the example below. A new **Survival Analysis** node will be created, and the wizard will open.

OR:

- Drag the **Survival Analysis** node from the **Model** palette onto the canvas and connect the required input dataset to the node.



If the Survival Analysis node is already on the canvas, double-click it or right-click on it and select **Modify** to open the wizard.

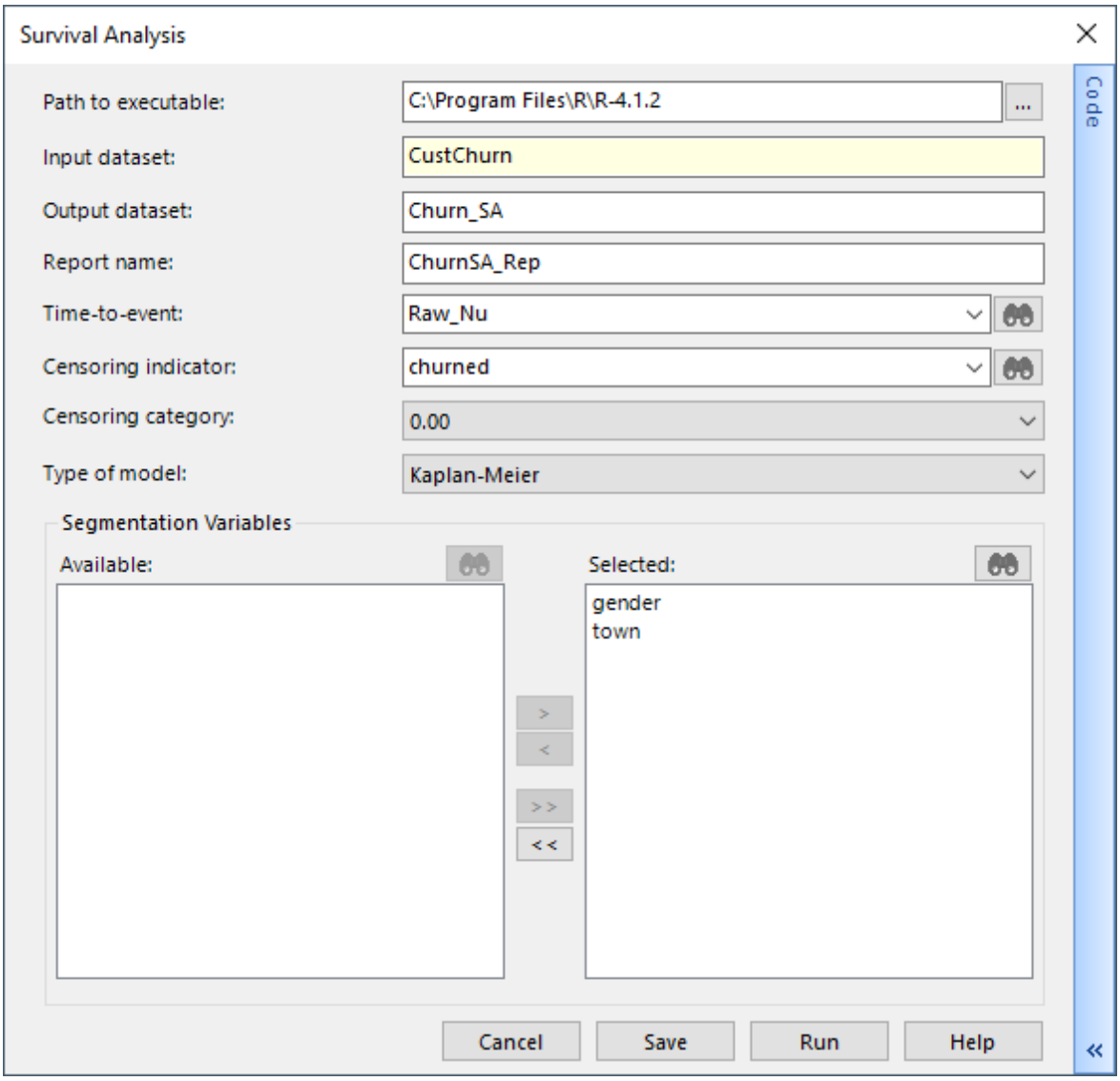


The Survival Analysis dialog opens. Specify the following inputs:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- **Path to Executable** shows the location of the R **bin** folder where **`R.exe`** resides, as well as the location of the R **library** folder.
    - In the Standalone configuration, the default path is defined by Environment Preferences. In the standalone mode, you can change this path in the Survival Analysis node at any time (for example, if you have two versions of R and would like to use a different one). The change is applied when clicking **Save** or **Run**.
    - Update the default path in Environment Preferences if you change your R environment (e.g., after upgrading R) or if you need to use a different version of R when two or more versions are installed.
    - In the Client/Server configuration, Path to Executable is read-only. It is defined by the configuration on the KS Server host machine. Please contact your server administrator if you need to change the R environment used by KS Server.
- Specify the names for the output dataset and report.
- Select the field in the input dataset that indicates the time (time-to-event).
- Select the censoring indicator field in the input dataset. Only fields with cardinality 2 are available for selection.
- Select the censoring category (the value indicating that the event did **not** occur during the observation time)
- Choose the model type (Kaplan-Meier, Cox proportional hazards, or Weibull regression)
- Select the segmentation variables to include

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The node generates R code that applies the functions of the survival, xtable, and KnowledgeSurvival packages with the input parameters specified in the wizard. You can preview the auto-generated R code at any time by clicking the arrow at the bottom of the Code bar - the blue vertical bar on the right side of the window. Click the arrow again to close the preview.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Survival Analysis

```
library(angoss)
memory.limit(32611)
setwd("C:/Altair KS Projects 5/Churn_Analysis/")
png(filename = "RplotFile000.ks_png", width = 800, height = 600)
Sys.setenv(R_DISABLE_HTTPD="1")
options(menu.graphics = FALSE)
CustChurn <- utils.read("C:/Altair KS Projects 5/Churn_Analysis/", "CustChurn")
library(KnowledgeSurvival)
library(survival)
library(xtable)
data(CustChurn)
(mod <- survMod(data = CustChurn,
            timeToEvent = "`Raw_Nu`",
            censoringIndicator = "`churned`",
            censoringCategory = "0.00",
            segmentationVariables = c("`gender`",
"`town`"),
            typeOfModel = "kaplan"))
report(mod, file = "#report.html")
Churn_SA <- survGraph(mod)
utils.write(Churn_SA, "C:/Altair KS Projects 5/Churn_Analysis/", "Churn_SA")
```

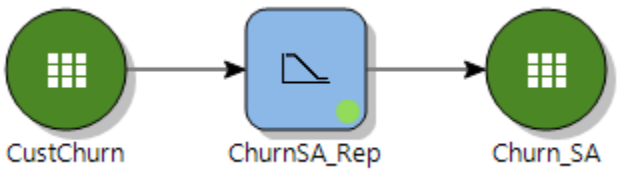
Code

The code in the preview window cannot be modified, but you can copy and paste it in the R Code node and edit it as necessary.

Before the code is passed to the R engine for execution, the input data is converted in the background from the native Altair data format (KDD) to R data.

Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.

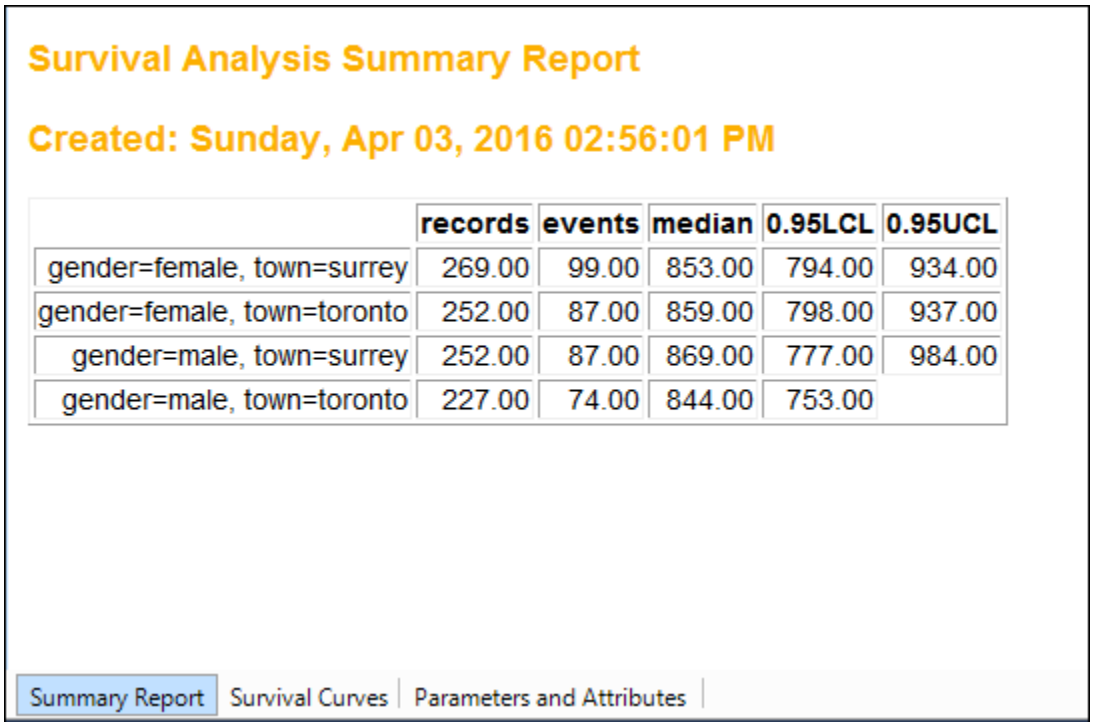
The output dataset and report are created after running the node.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To view the report, double-click on the Survival Analysis node or right-click it and select **Open View**. The report has three tabs: Summary Report, Survival Curves, and Parameters and Attributes.

The Summary report shows the event statistics for each segment defined by the segmentation variables selected in the wizard.



Survival Analysis Summary Report

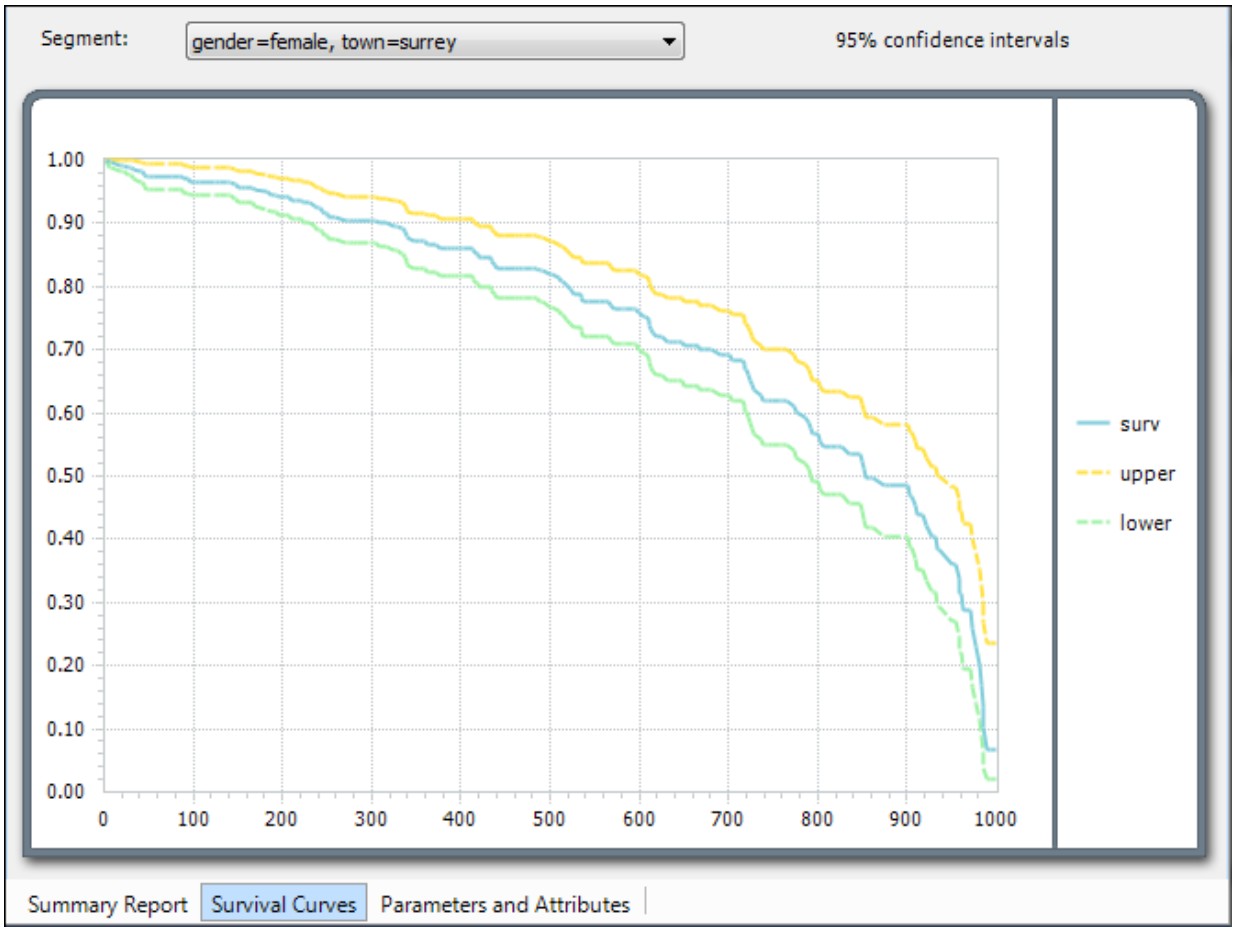
Created: Sunday, Apr 03, 2016 02:56:01 PM

	records	events	median	0.95LCL	0.95UCL
gender=female, town=surrey	269.00	99.00	853.00	794.00	934.00
gender=female, town=toronto	252.00	87.00	859.00	798.00	937.00
gender=male, town=surrey	252.00	87.00	869.00	777.00	984.00
gender=male, town=toronto	227.00	74.00	844.00	753.00	

Summary Report | Survival Curves | Parameters and Attributes

The Survival Curves tab shows the survival curve and the 95% confidence interval curves for the segments defined by the segmentation variables selected in the wizard. Choose the desired segment from the drop-down list at the top of the tab view.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also: R Code Node

Python Code Node

Overview

The Python language integration feature allows you to embed Python programs in your workflows. For more information on Python, see the Python project website https://www.python.org/.

Users can write and execute Python code within a **Python Code** node in the project workflow and use its outputs in further operations. The outputs of your Python programs embedded in the project workflow can be used as regular Altair KDD datasets. Thus, all Python functions and algorithms can be used within the application.

Integration of Python programs into Knowledge Studio and Knowledge Seeker projects has several benefits:

1. The functionality of Altair Data Analytics applications is expanded with Python functions and libraries. Data manipulation and modeling algorithms present in Python packages for mathematics and data science can be used within the Altair Data Analytics applications. Load

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

any Python library that contains the required algorithms and write Python programs in Python Code nodes, then use their outputs as regular datasets to continue building your workflow.

2. Workflow continuity: Python users do not need to switch from a Altair Data Analytics application to Python and back when working on their projects.

3. Workflows prepared with the use of Python can be translated into production-ready deployment code in the SAS Language, SQL, or PMML

Prerequisites

Supported Python versions are 3.7.0 and higher. The required Python packages include Python Data Analysis Library (*pandas*), which provides data structures and analysis tools for Python, the *SciPy* package, and the *reports* package. See the Prerequisites section of the Deployment Guide. For the client/server configuration of Knowledge Seeker or Knowledge Studio, Python needs to be installed on the server side only.

For KS Server on Linux, the KS Server startup script `ks_start` must be configured to be aware of the Python environment. The PYTHON_HOME environment variables must be set. Please see the Deployment Guide for details.

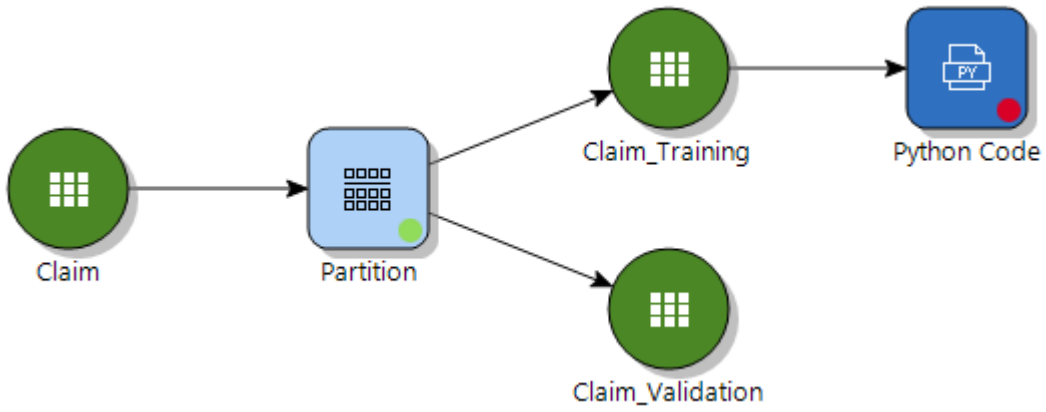
Embedding Python Programs in Project Workflows

For information about projects and workflows, see the topics Projects and Workflows.

To add a Python Code node to your workflow:

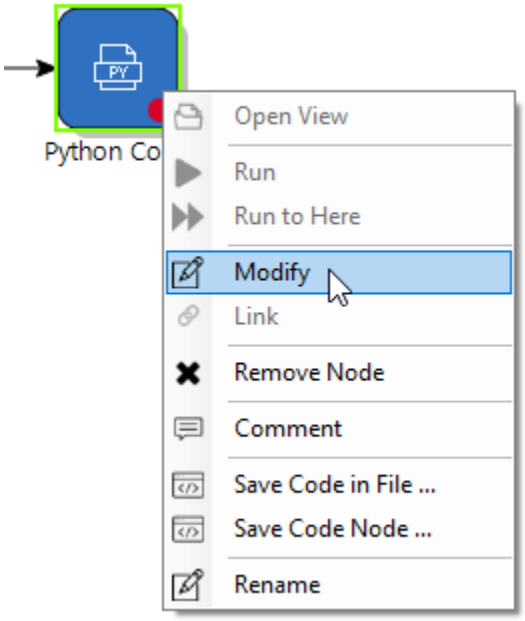
Open the **Action** palette and drag the **Python Code** node onto the Workflow canvas. Alternatively, right-click anywhere on the canvas background and select **Action --> Python Code**.

Connect one or more input datasets to the new Python Code node.



Double-click on the Python Code node to open the Python Code Editor window. Alternatively, right-click on the Python Code node and select **Modify**.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

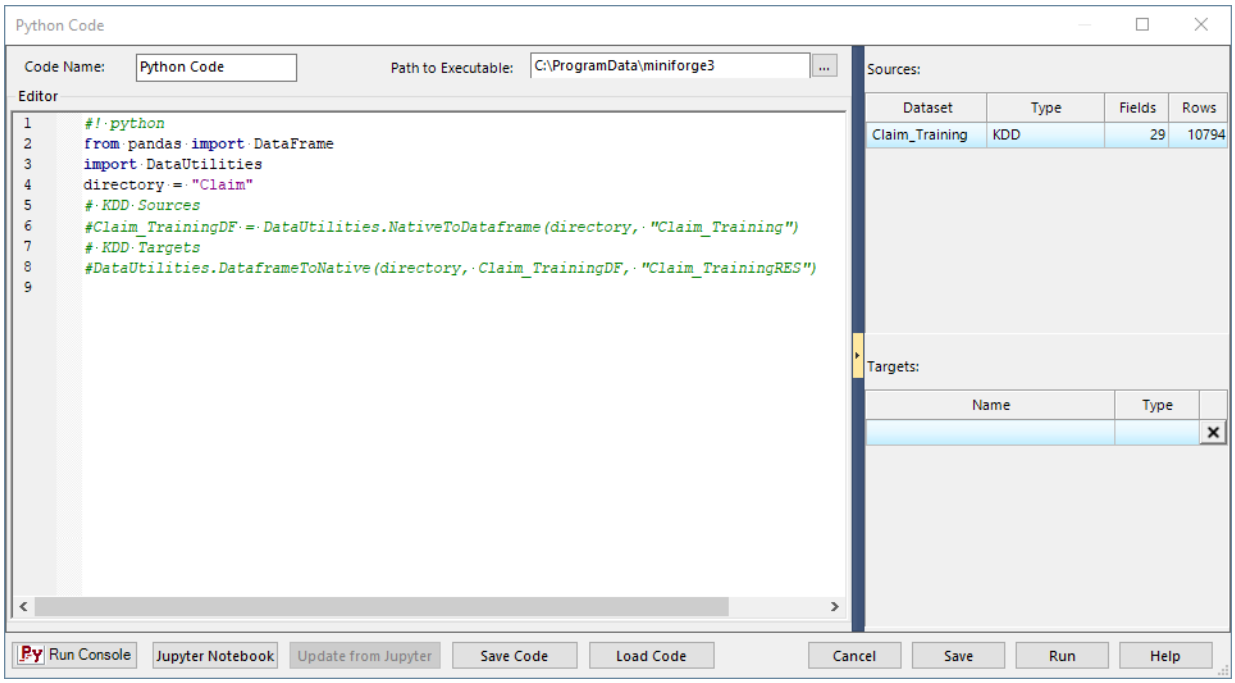


The **Python Code** editor window opens.

- In the **Code Name** field, specify the name for the Python Code node - the label to be displayed under the node on the workflow canvas. The default name is "*Python Code*".
- **Path to Executable** shows the location of the folder that contains **`python.exe`**, as well as the Python packages. The default path is determined by Environment Preferences. You can change this path at any time (for example, if you have two versions of Python and would like to use a different one). After changing the path, click **Save** and reopen the Python Code node for the change to take effect.
    - Change the default path in the Environment Preferences if you change your Python environment (e.g., after upgrading Python) or if you need to use a different version of Python when more than one version is installed.

Start writing your Python program as you normally would in any Python development environment editor.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The first time you open a new Python Code node that has at least one input dataset connected to it, several lines of necessary initialization code are automatically executed. You will see it in the Console part of the Python Code window. Here is an example, with source and target dataset conversion commands uncommented:

```
#! python
from pandas import DataFrame
import DataUtilities
directory = "Claim"
# KDD Sources
Claim_TrainingDF     =     DataUtilities.NativeToDataframe(directory,
"Claim_Training")
# KDD Targets
DataUtilities.DataframeToNative(directory,          Claim_TrainingDF,
"Claim_TrainingRES")
```

First, the command **`import DataUtilities`** loads the Python integration library. Then the **`directory`** command sets the Python workspace to your current project folder. The **`DataUtilities.NativeToDataframe`** command reads all the input datasets of the current Python Code node into Python (pandas) dataframes.

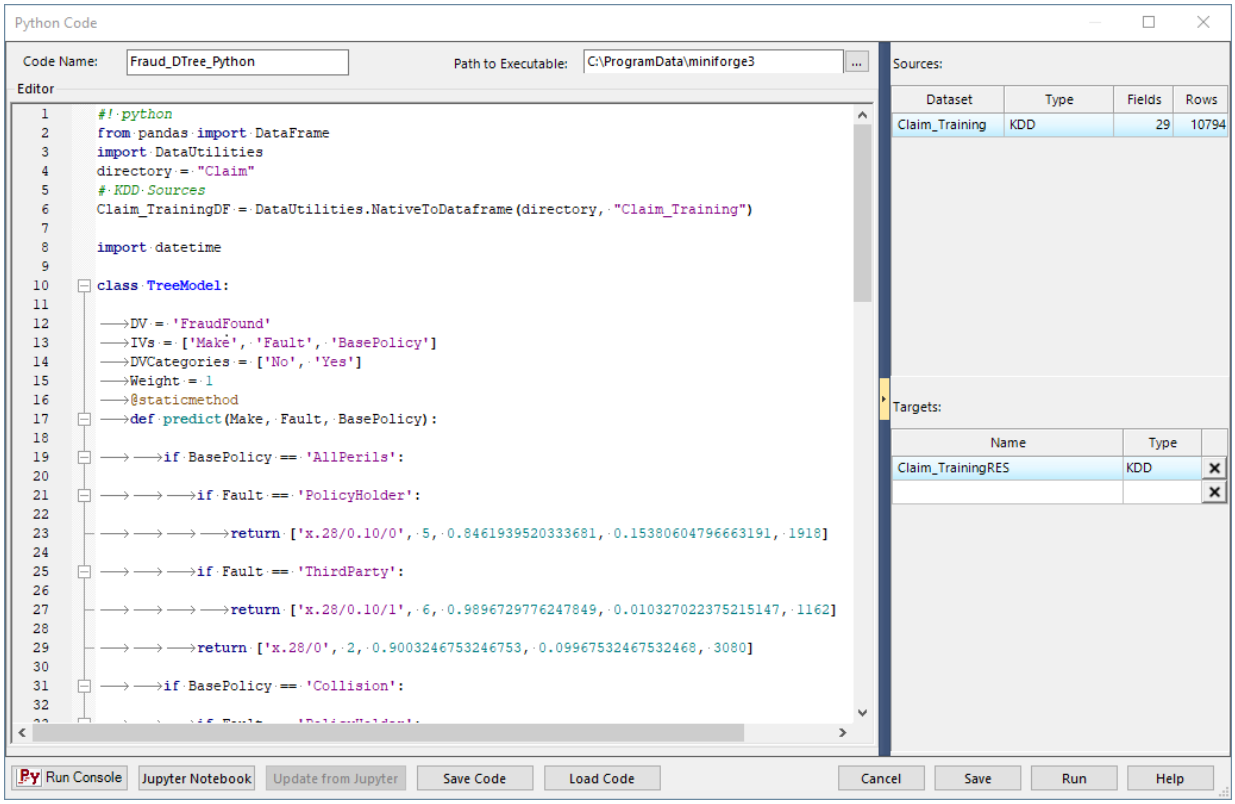
You can export pandas dataframes to the KS dataset format using the **`DataUtilities.DataframeToNative`** command, for example, **`DataUtilities.DataframeToNative(directory, ClaimDF, "ClaimKDD")`**.

At any point in your program, you can load any other libraries and other KS datasets from the same project using the commands as in the above example.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

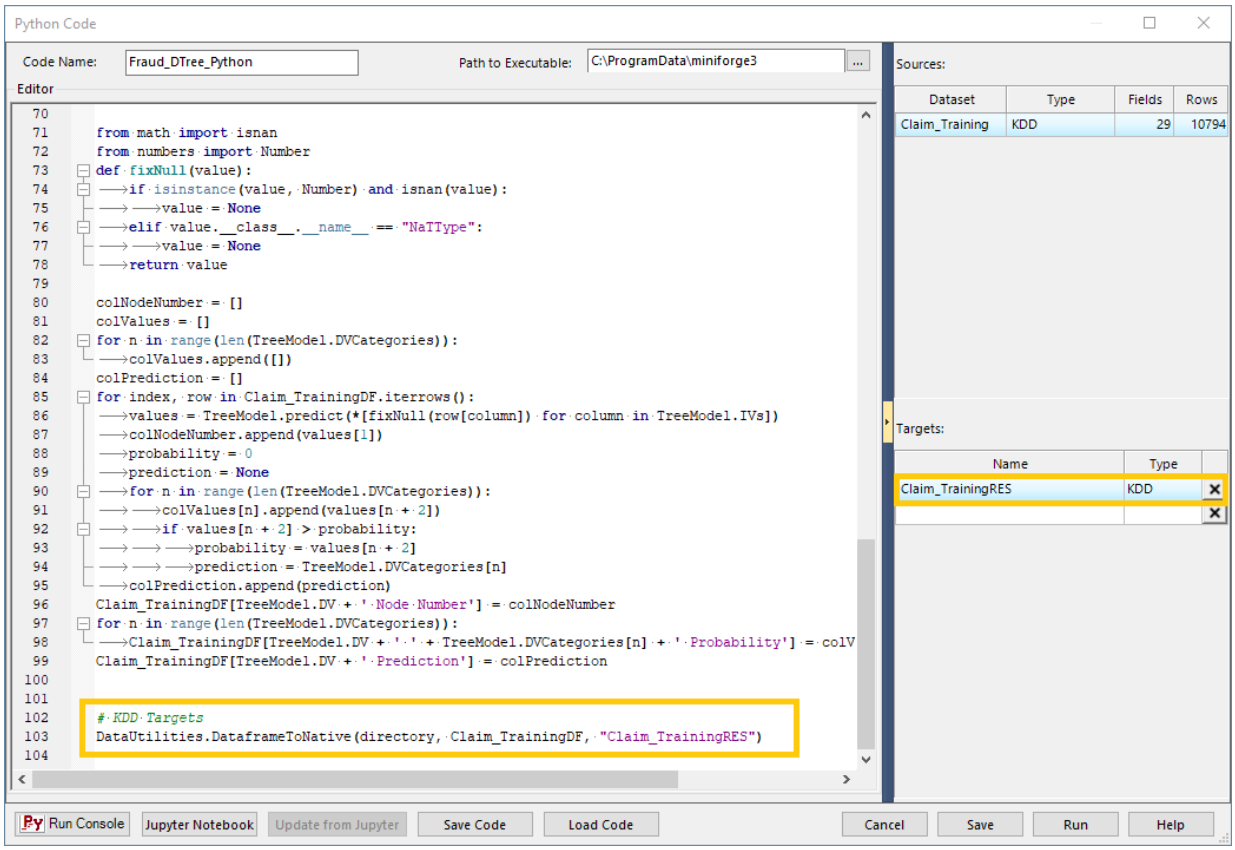
The inputs and outputs are listed in the panel on the right-hand side of the Python Code window. Note that the Python node does not produce any output dataset nodes unless you specify the names of the output datasets you want to be created. Some of the datasets you want to create as intermediate datasets, without using them in the subsequent workflow, and some of them will be used later. You need to enter the names of these datasets in the **Targets** section of the panel on the right-hand side.

In the example below, the input dataset for the Python Code node is *Claim_Training*. First, it is converted to a pandas dataframe. The code that follows represents a decision tree model.



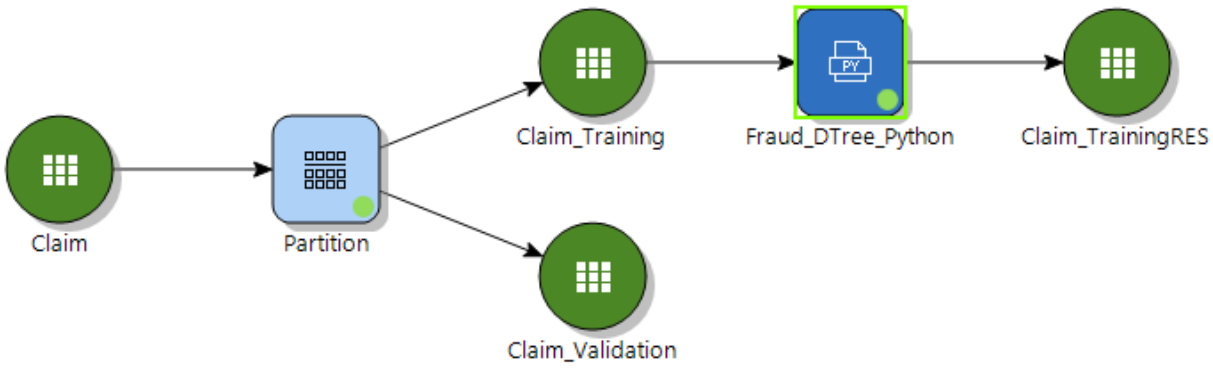
The decision tree rules in this example are followed by the scoring code. The scores (probabilities) produced by the model are written to the dataframe *Claim_TrainingDF*, which is then converted to an Altair KDD dataset *Claim_TrainingRES.* Adding its name in the **Targets** section of the panel on the right, we designate it as the output dataset of the Python Code node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



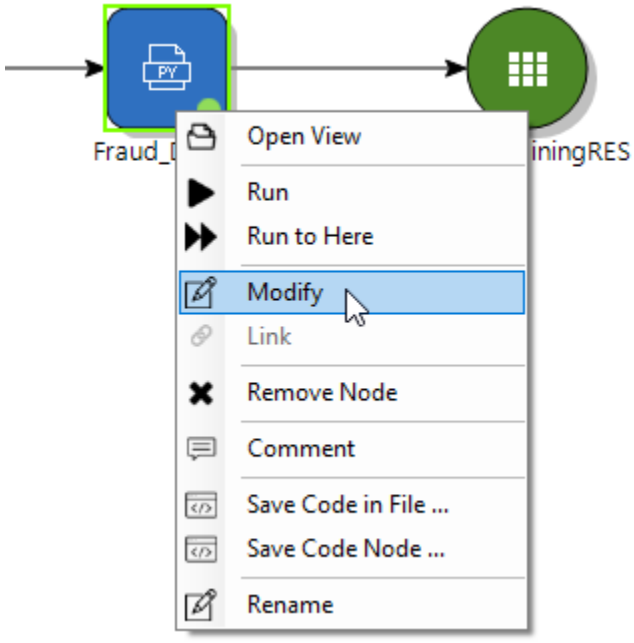
The Sources & Targets panel of the Python Code window on the right-hand side can be hidden by docking it to the right to expand the area for writing code. Click on the small arrow on the border between the editor frame and the Sources & Targets panel to dock it. Click the arrow again to restore the panel.

Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. The datasets specified in the Targets section (if any) will be created on as the outputs of the Python Code node on the canvas:



To rename the code node or to modify the code and run it again, right-click on the Python Code node and select **Modify**:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Jupyter Notebook Integration

From the Python Code node, you can launch Jupyter Notebook and load the code in the notebook session. For more information about Jupyter Notebook, see https://jupyter.org/.

Clicking the **Jupyter Notebook** button in the Python Code node opens a Jupyter Notebook session in your default web browser. The code from the Python Code node editor will be placed in the Jupyter Notebook window.

To use this feature, the path to the Jupyter executable must be specified in the configuration file **`KDSProvider.conf`**. Use any text editor to create a new file called KDSProvider.conf in the directory that contains the application binaries. The default location is:

- Windows Workstation (in the standalone configuration): `C:\Program Files\Altair Data Analytics\KS Workstation 2022.3.0\bin`
- Windows Server (in the client/server configuration): `C:\Program Files\Altair Data Analytics\KS Server 2022.3.0\bin`
- Linux: `/opt/Altair-Data-Analytics/KS-Server-2022.3/lib`

In KDSProvider.conf, specify the path to the Jupyter executable as a value of the jupyter_path parameter as follows:

```
jupyter_path <path>
```

For example:

Windows:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

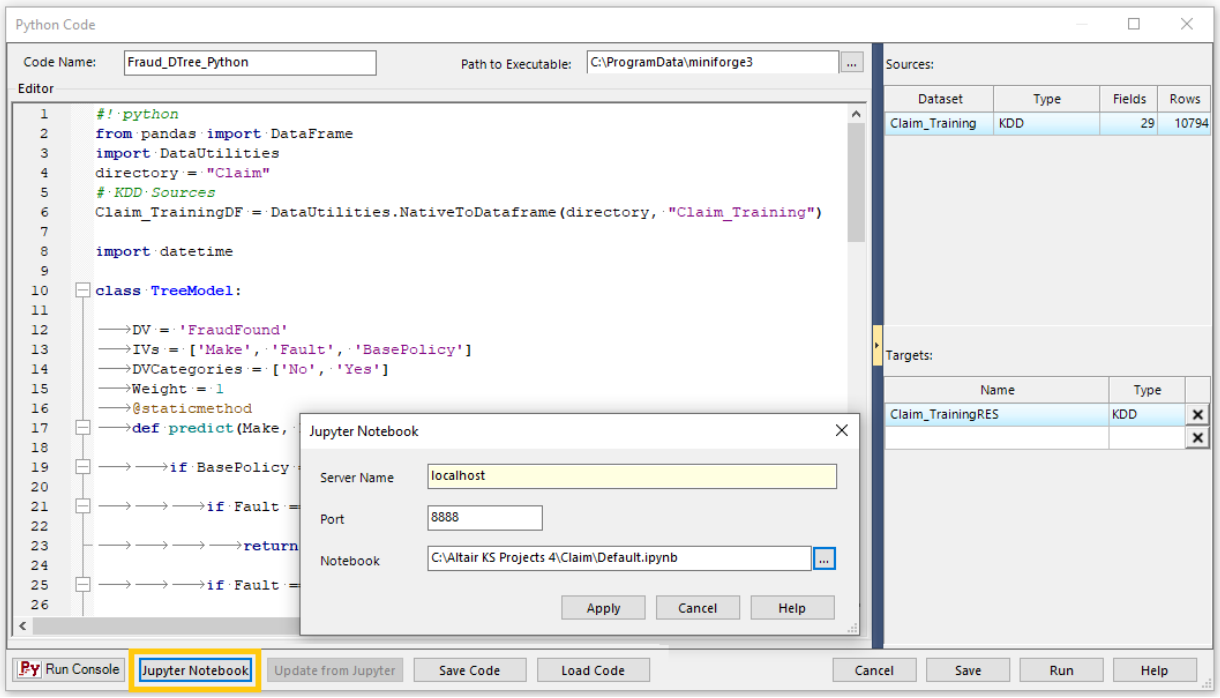
```
jupyter_path
"C:\ProgramData\Python3\Scripts\jupyter.exe"
```

Linux:

```
jupyter_path /opt/python3/bin/jupyter
```


Save the changes and restart the application.

Open the desired Python Code and click the **Jupyter Notebook** button. This will open the Jupyter connection dialog.

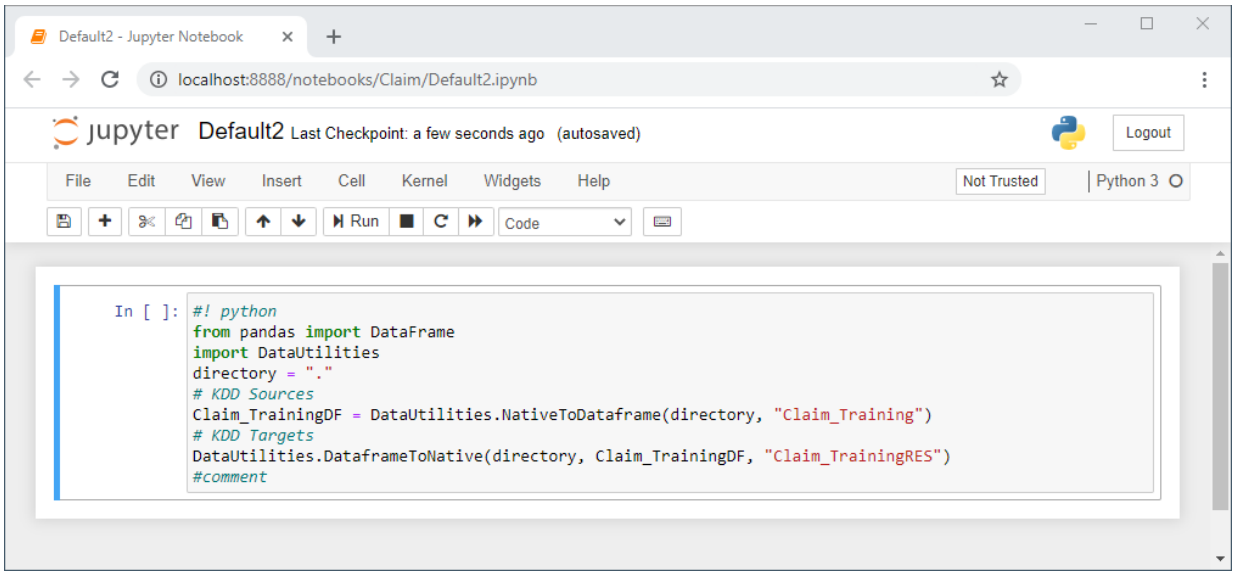


Specify the following inputs:

- Jupyter server name. The default is *localhost* in the standalone mode, and *http://<server_name>* in the client/server mode.
- Port number. The default is 8888
- Notebook file name and path. The default is <current_project_folder>\Default.ipynb. Browse for the desired location and change the notebook name if necessary. If the specified notebook file already exists and is not empty, you will be prompted to replace the existing code in it.

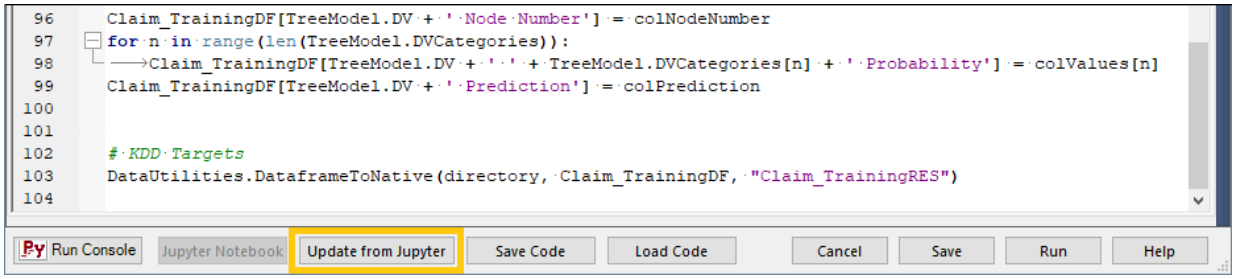
The Jupyter Notebook is launched in your default web browser, and the code from the Python Code node is loaded in the new notebook session.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

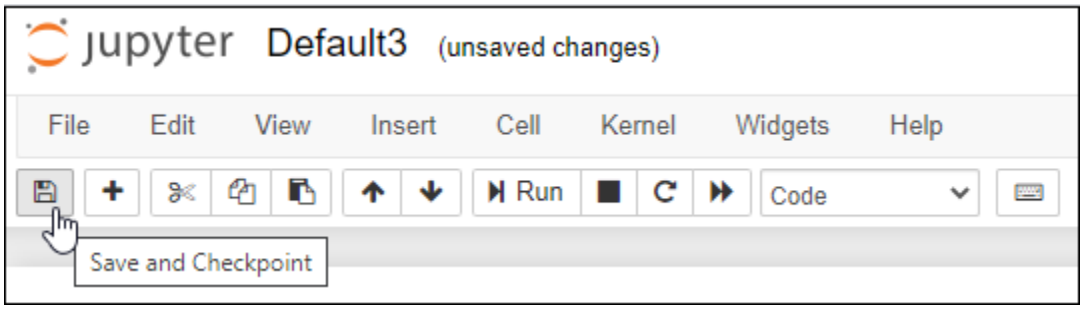


Once Jupyter Notebook is launched, it creates notebook files (*.ipynb) in the project directory. The Python Code node monitors the running notebook process.

When Jupyter notebook opens in the browser, the **Jupyter Notebook** button in the Python Code node becomes disabled, so you cannot launch another notebook session from node wizard. At the same time, the button **Update from Jupyter** becomes enabled, so that you could refresh the content of the Python Code node editor from the linked notebook. Thus you can synchronize the code in the Python Code editor and the notebook to view the code changes without having to click **Save** or **Run** (which would close the node wizard).

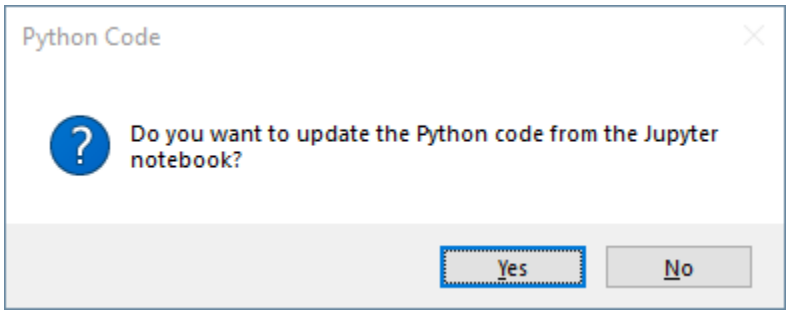


The button **Update from Jupyter** refreshes the code entered in the notebook up to the last checkpoint. If there is any unsaved code in the notebook, first click **Save and Checkpoint** in the notebook toolbar, then click **Update from Jupyter** in the Python Code editor.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Once the notebook session is finished and you switch back to the Python Code node, you will be prompted to update the code from the notebook.



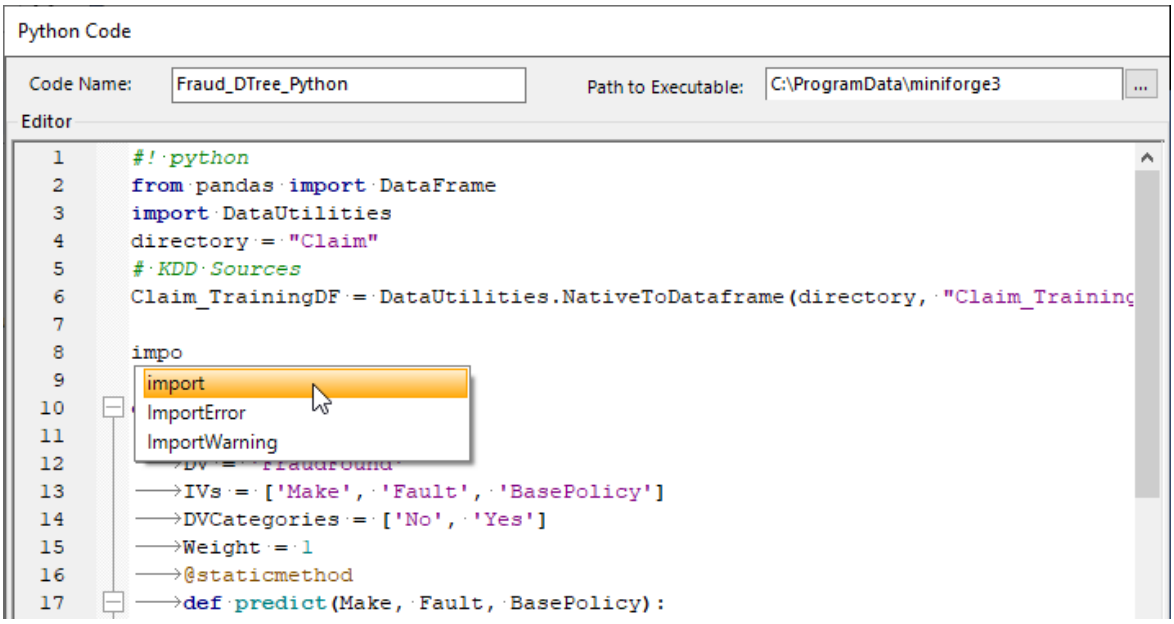
Click **Yes** if you want to update the code, then continue working with it in the Python Code node. The **Jupyter Notebook** button will become enabled again.

Code Editor Features

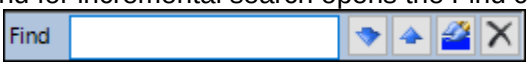
1. Code editing enhancers

The Python Code editor window provides basic IDE features to enhance code editing:

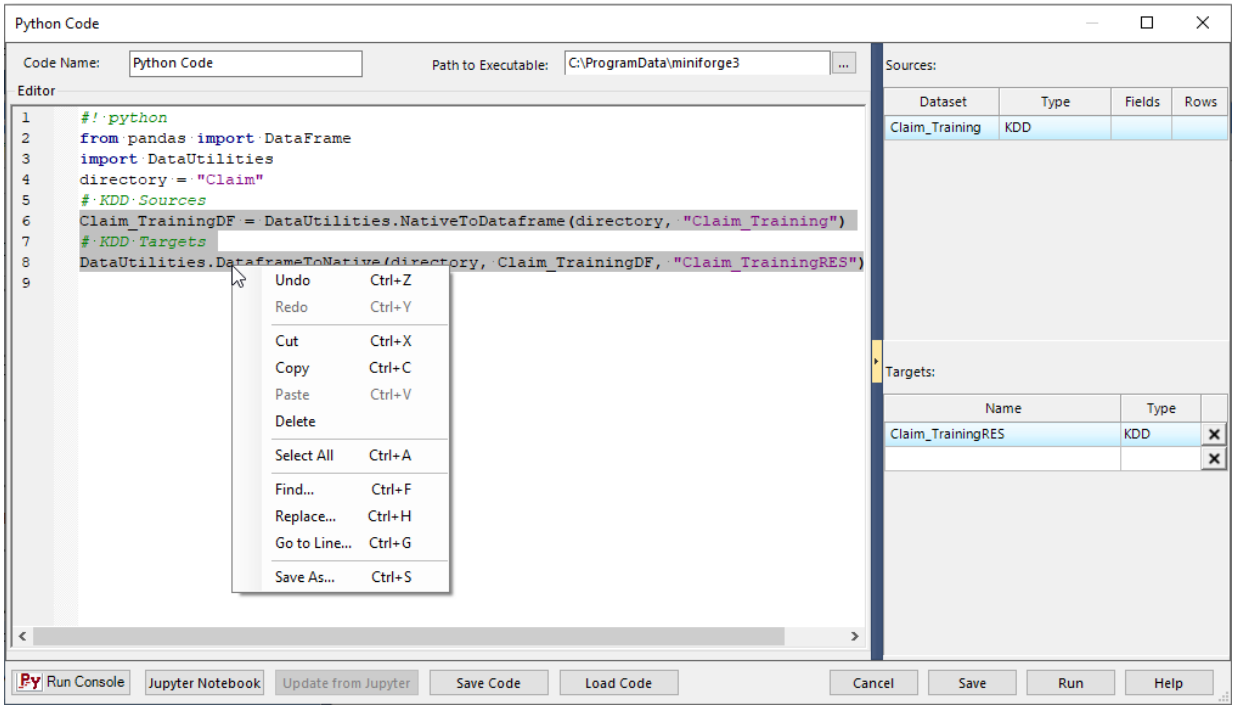
- line numbers
- collapsing / expanding blocks of code
- syntax highlighting displays code elements in different colours and fonts according to their category
- the auto-complete function predicts and suggests the commands as you type (after typing at least two characters); when multiple suggestions appear in a list, select the desired one with single mouse click or using the arrow keys and pressing Enter.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- **Undo**, **Redo**, **Find**, **Replace**, and **Go to Line** functions are available in the context menu and with keyboard shortcuts in the Editor frame.
- The code in the Editor frame and Console frame can be saved as a Python program file (*.py) or a text file using the **Save As** command in the context menu.
- Common keyboard shortcuts for business applications are provided: Ctrl+Z and CTRL+Y for Undo and Redo; Ctrl+F and Ctrl+H for Find and Replace; F3 and Shift+F3 for Find next / Find previous; Ctrl+G for Go to line; Ctrl+C and Ctrl+V for Copy and Paste; Ctrl+X for Cut; Ctrl+A for Select All).
- The **Ctrl+I** command for incremental search opens the Find control at the top-right corner of the Editor window: 

. Enter your search string, then use the Next/Previous and the Highlight All Matches buttons in this control. To exit the search, just click anywhere else within the Python Code window.



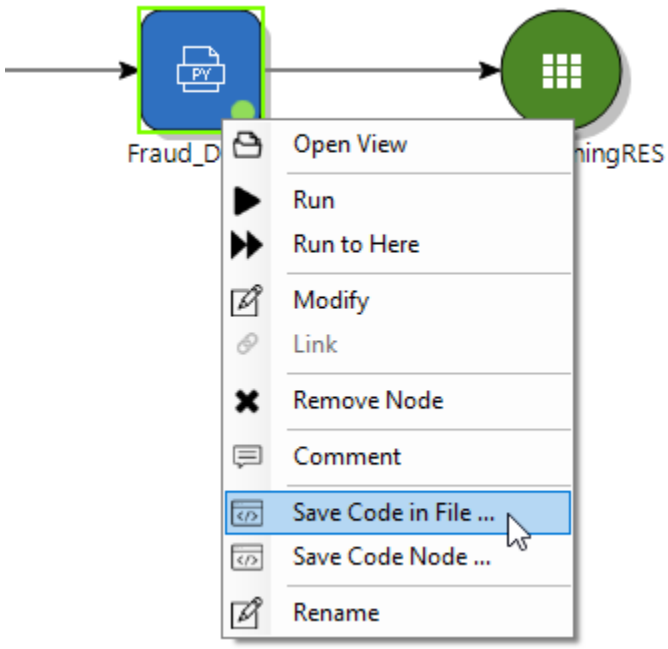
2. Saving code as a file

The code from any Python Code node in your workflow can be saved as a file in two ways:

a) Using the context menu of the node:

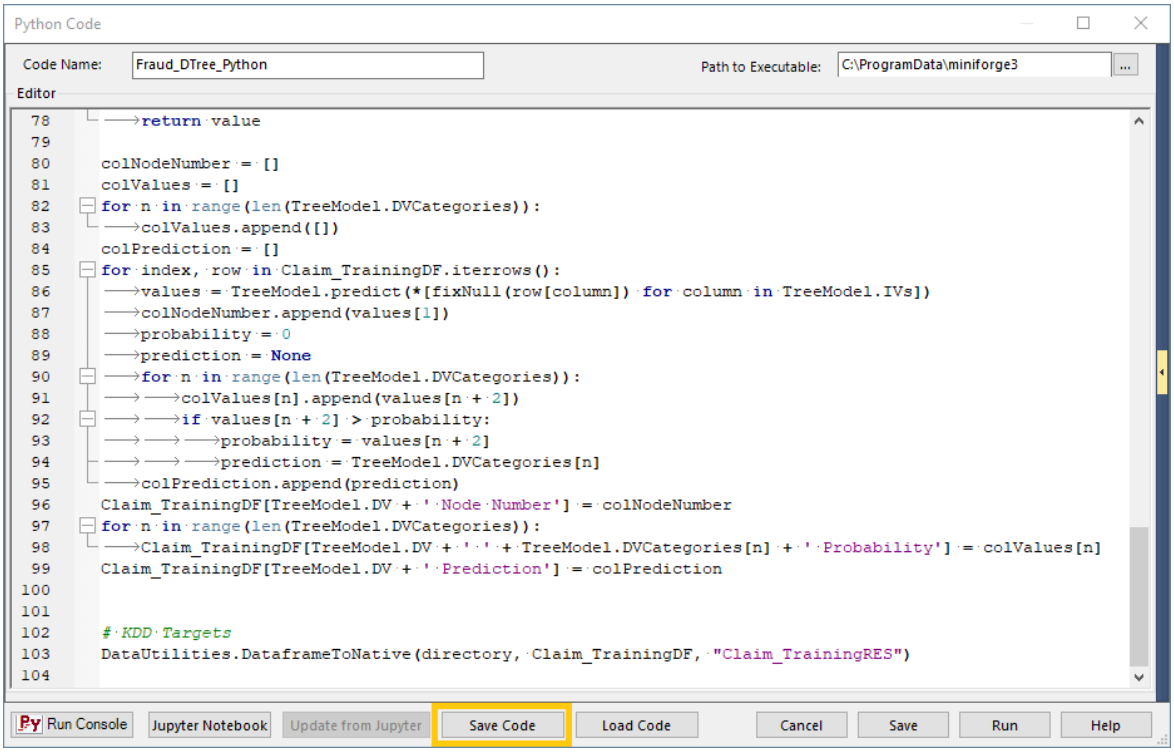
Right-click on the Python Code node and select the command **Save Code in File ...**

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



In the *Save Python Code* window, browse to the desired destination folder and specify the name for the output file. By default, it is saved as Python program type (with extension ***.py***). The default file name is PythonCode.py.

b) Using the **Save Code** button in the Python Code editor window:

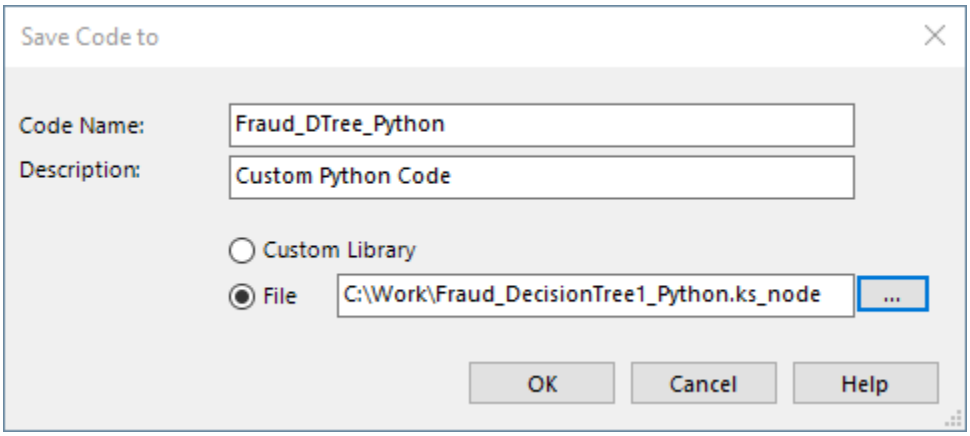


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Click **Save Code** at the bottom of the Python Code editor window.

In the *Save Code* dialog:

- Select the **File** option, browse to the desired destination folder, and specify a file name. The default extension is ***.ks_node**. This is a re-importable XML file format that allows importing the code into any other Python Code node using the **Load Code** button. The code saved in this format can be easily shared with other users of Knowledge Studio and Knowledge Seeker.
- Specify the code name and description. These attributes will be stored together with the code in the saved file.



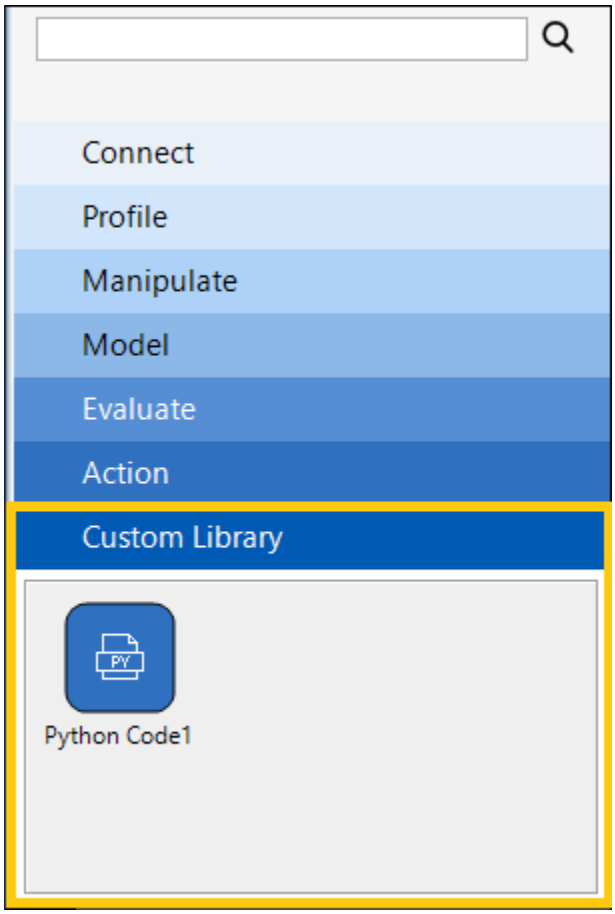
Click **OK**. You will get a message confirming that the code has been saved successfully. For the information on the **Load Code** feature, see item 4 below.

c) Using the **Save As** command in the right-click context menu of the Editor window or the **Ctrl+S** keyboard shortcut.

3. Saving code in the Custom Library palette

You can reuse the code from any Python Code node by saving it as a new node in a separate palette - the **Custom Library** palette. Any custom Python code node saved in the Custom Library palette can be renamed or removed. *Note: The* **Custom Library** *palette is not shown unless it contains at least one node that was saved or imported previously*.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

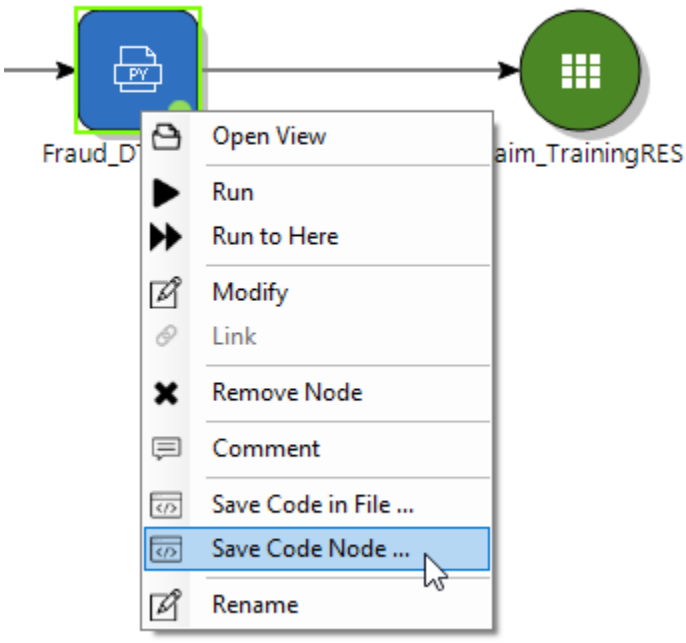


There are two ways to save a Python Code node in your workflow under a new name in the Custom Library palette:

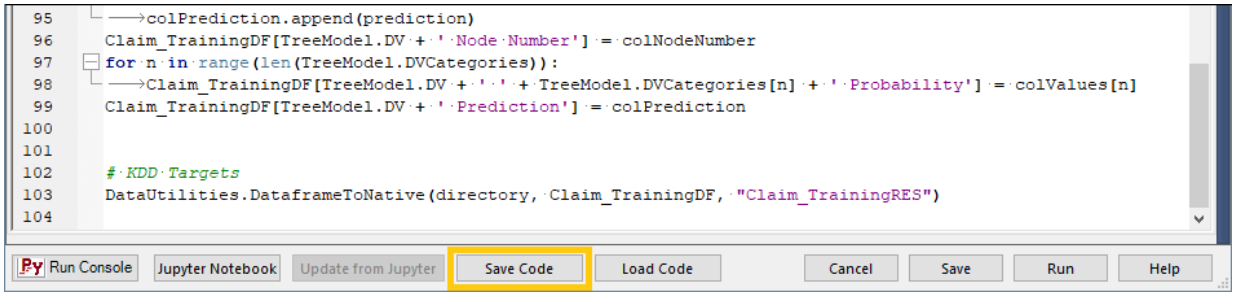
a) Using the context menu of the node:

Right-click on the Python Code node and select the command **Save Code Node ...**

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



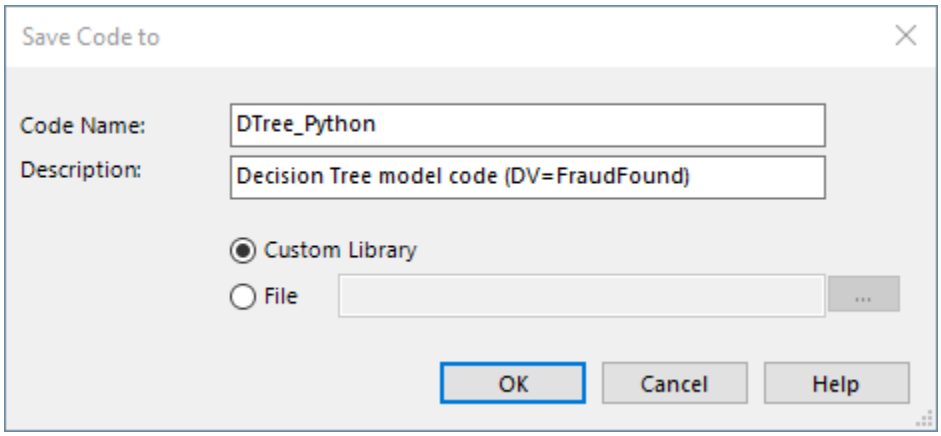
right-click on the code node on the workflow and select **Modify** to open the editor. Edit the code if necessary and click the **Save Code** button at the bottom of the editor window.



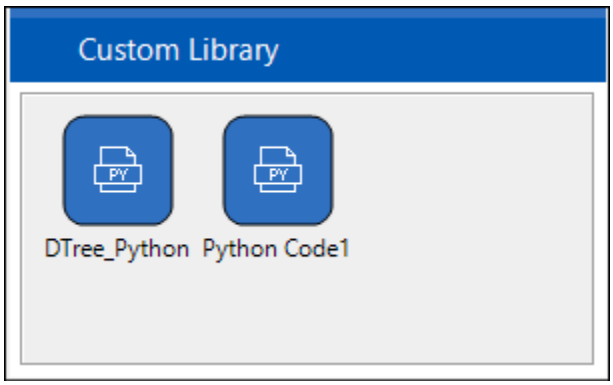
The "*Save Code to*" dialog opens.

- Select the **Custom Library** option.
- Specify the name for the new custom code node as it should appear in the Custom Library palette. The default name is that of the original node.
- Enter the description text (optional). The description will be shown as the tooltip when hovering over the new custom code node in the Custom Library palette.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

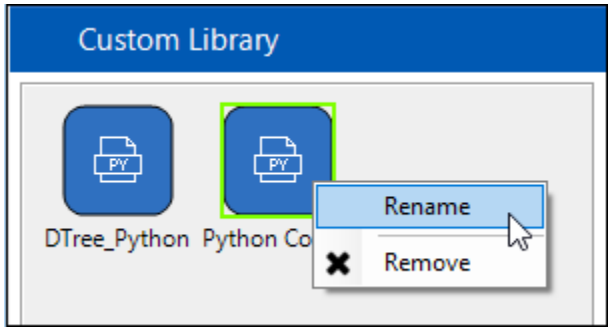


Click **OK**. The new custom code node will be placed in the Custom Library palette.



Note: The code is saved "as is". When reusing the custom code nodes from the Custom Library palette, check the code for the dependencies on the source and target datasets in the new project context, link source dataset(s), specify target(s) and modify the code appropriately.

To rename or remove any node in the Custom Library palette, right-click on the node and select **Rename** or **Remove** from the context menu:

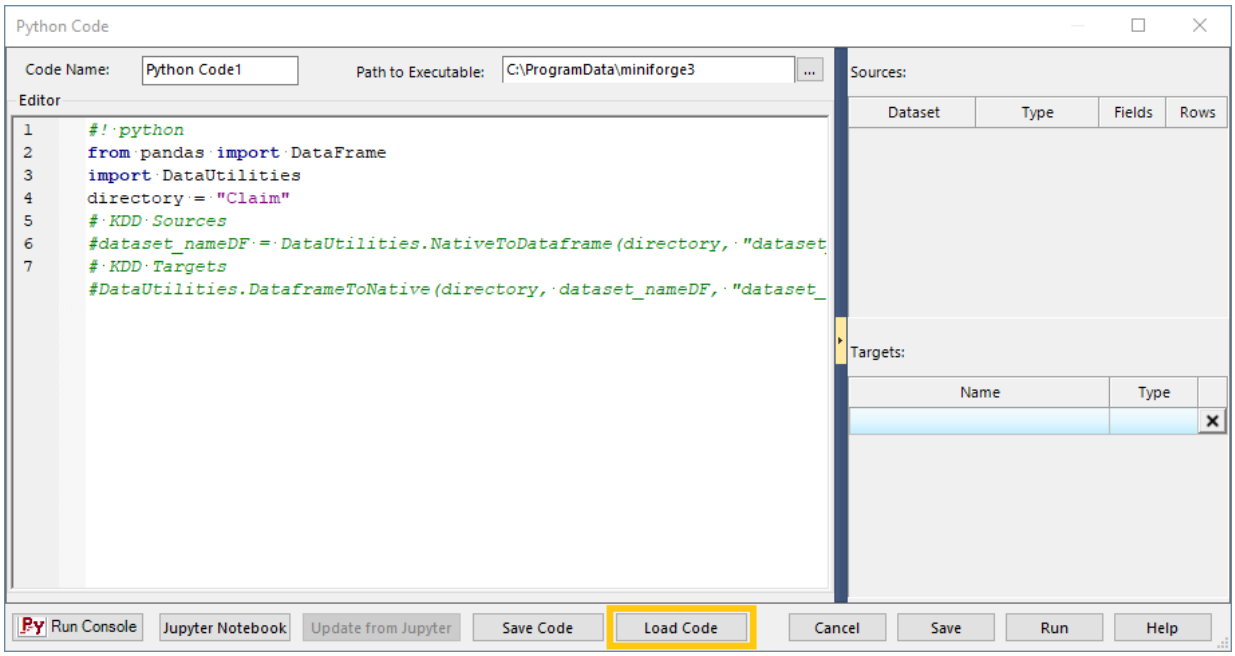


4. Loading code from file or from Custom Library

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The **Load Code** button in the Python Code editor window loads previously saved custom code from the **Custom Library** or from a ***.ks_node** file (see above). Using this function, users can reuse and share their code as custom Python Code nodes. For example, they can be exchanged via email, loaded from disk, then saved to the Custom Library palette if desired.

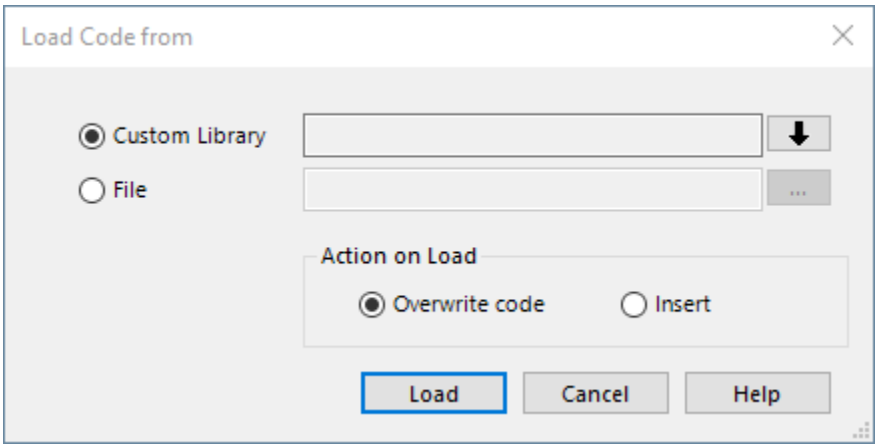
Add a new Python Code node to your workflow or open an existing one on the workflow canvas. In the Python Code editor window, click the **Load Code** button:



The "*Load Code from*" dialog opens.

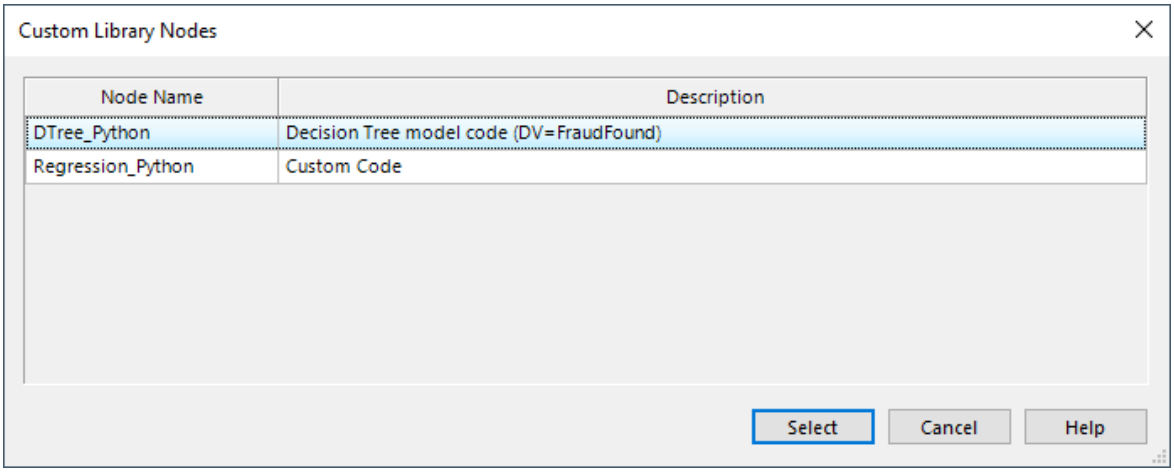
- Select the **Custom Library** option if you want to load code from one of the nodes in the Custom Library palette.
- Select the **File** option if you want to load code from a file. The source file must be in the "custom node" file format (with extension ***.ks_node**) - a re-importable XML file format used by the **Save Code** feature. Browse for the source *.ks_node file, select it and click **Open**.
- Choose whether to **overwrite** any existing code in the editor window or to **insert** the new code at the current cursor position.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



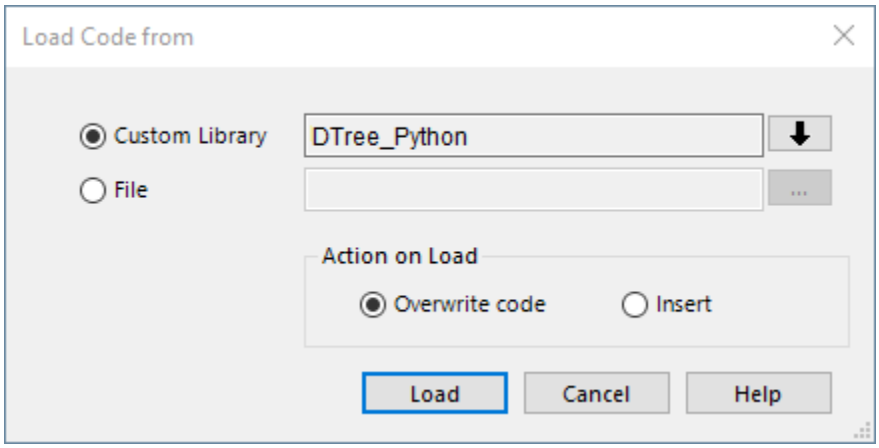
If you selected the **Custom Library** option, click the down arrow on the right to open the Custom Library Nodes list. It displays the custom code nodes available in the Custom Library palette. Choose the desired source node click **Select**.

Note: The default location for the Custom node templates in the file system is the folder `%APPDATA%\Altair\CustomLibrary` of the current user profile. When the application is started, this folder is scanned for valid templates, which are then displayed as nodes in the Custom Library palette.



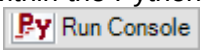
This brings you back to the "*Load Code from*" dialog. Click **Load** to load the code from the selected source.

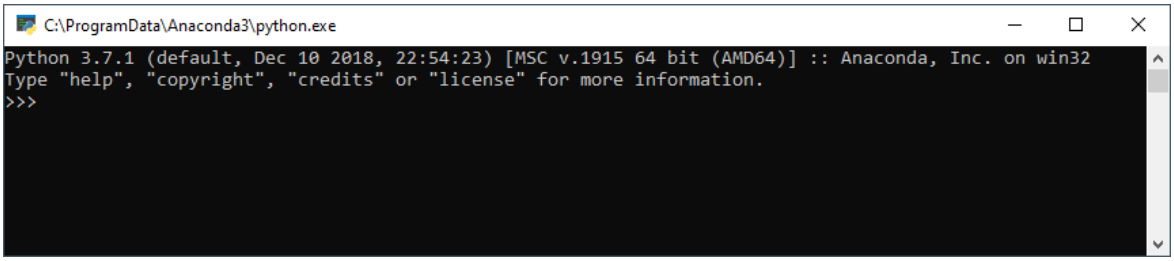
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Note: the code is loaded "as is". When reusing the code, check the code for the dependencies on the source and target datasets, link source dataset(s), specify target(s) and modify the code appropriately.

5. Launching the Python console

If necessary, the Python console can be launched from within the Python Code editor window by clicking the **Run Console** button at the bottom-left corner: 



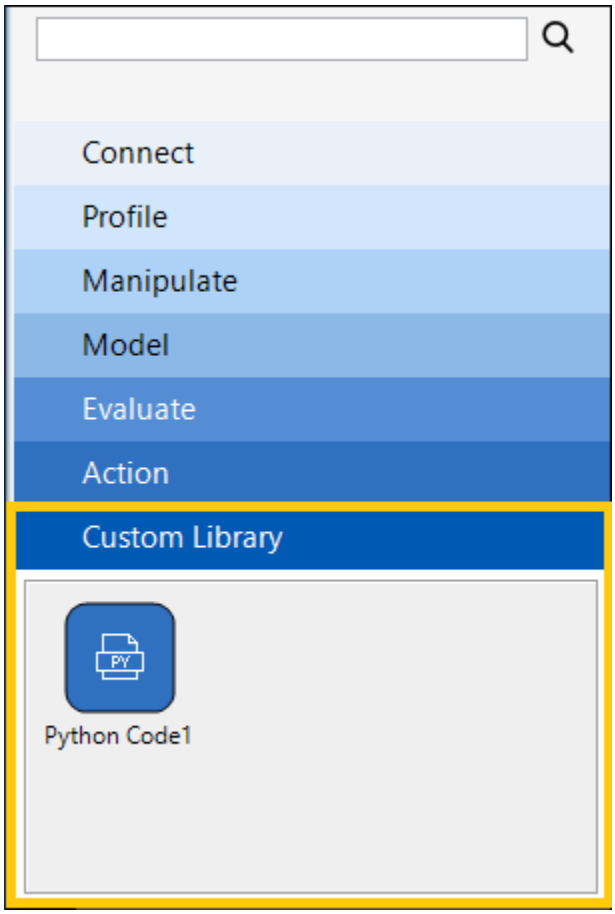
See also: R Code Node, SAS Language Code Node

Custom Library Palette

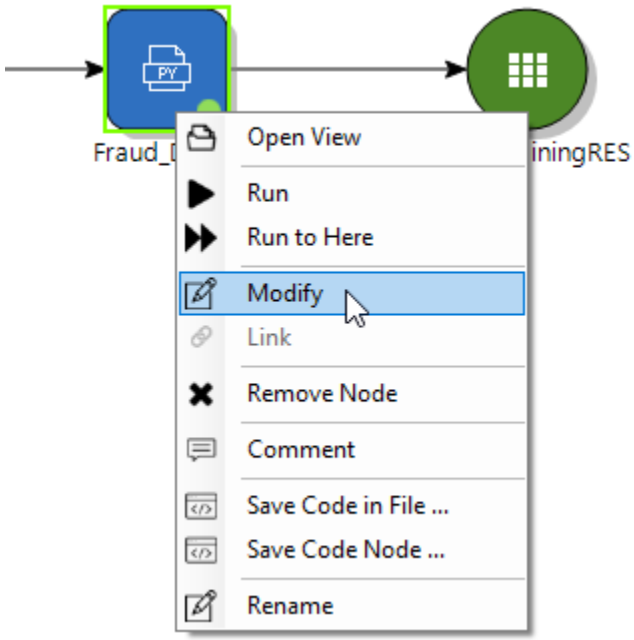
The code from any Python Code node, R Code node, and SAS Language Code node can be reused by saving it as a new node in a separate palette - the **Custom Library** palette. The nodes saved in the Custom Library palette can be renamed or removed.

The **Custom Library** palette is beneath the Action palette. It is not shown unless it contains at least one custom code node that was saved or imported previously.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

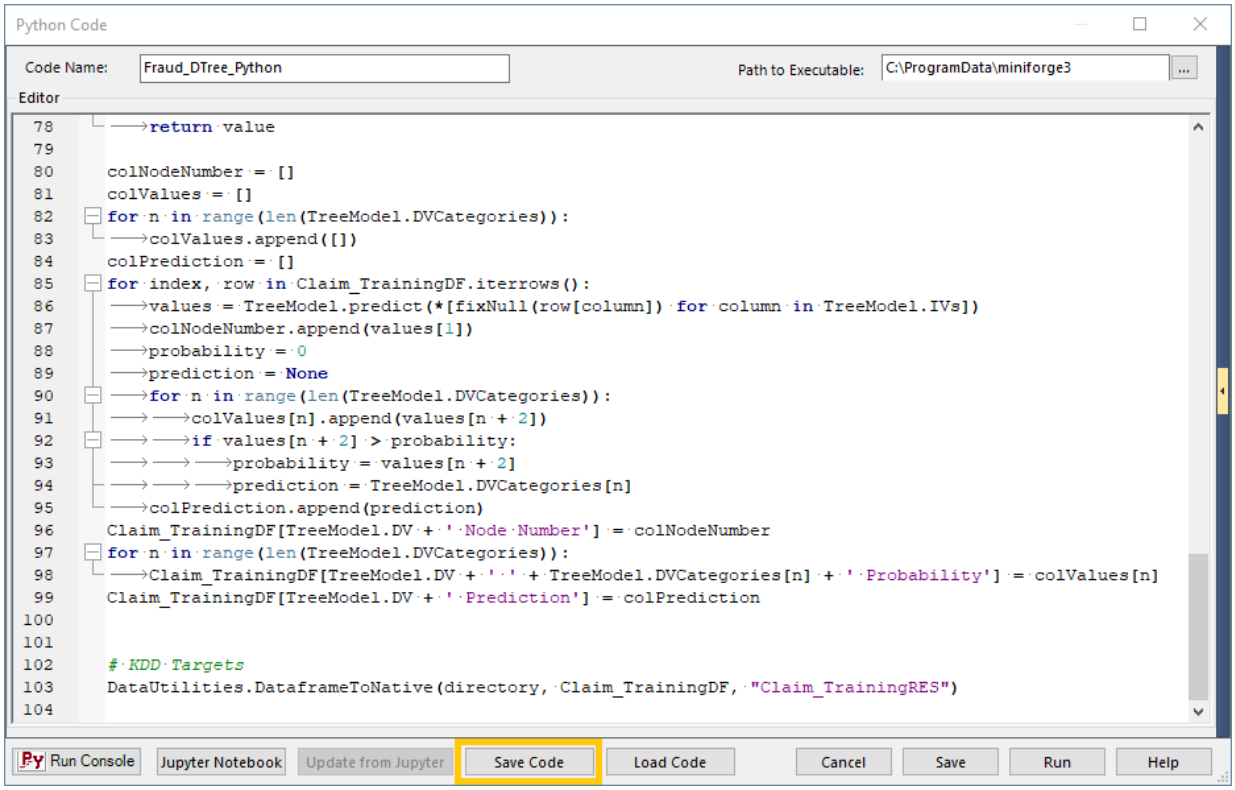


For example, to save a Python Code node under a new name in the Custom Library palette, right-click on the code node in the workflow and select **Modify** to open the editor.



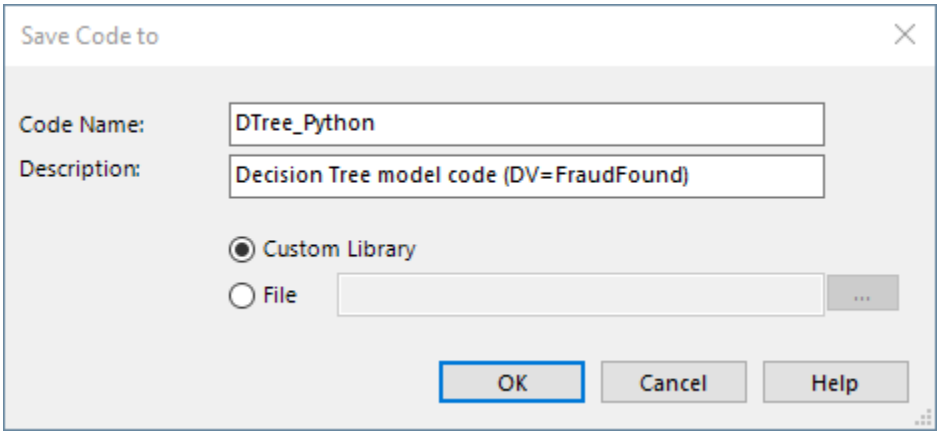
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Edit the code if necessary and click the **Save Code** button at the bottom of the editor window.



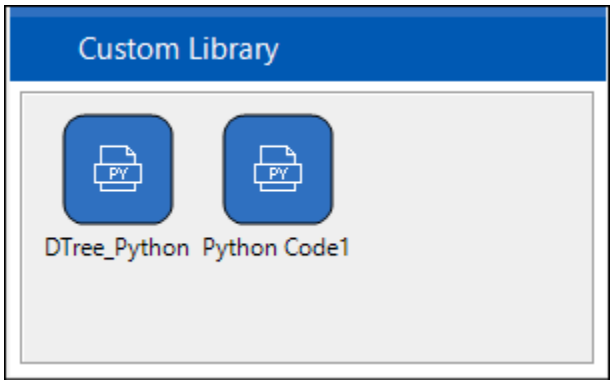
The "*Save Code to*" dialog opens.

- Select the **Custom Library** option.
- Specify the name for the new custom code node as it should appear in the Custom Library palette. The default name is the same as the original node.
- Enter the description text (optional). The description will be shown in the tooltip when hovering over the node in the Custom Library palette.



Click **OK**. The new code node will be placed in the Custom Library palette.

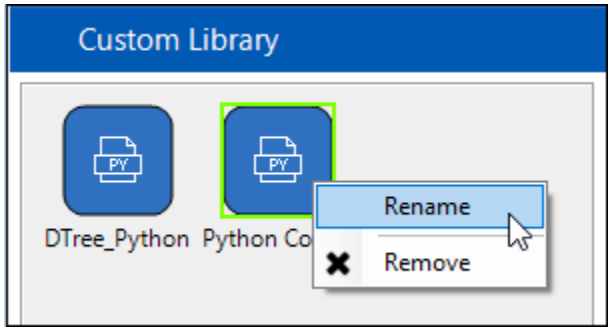
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The new node can now be used on the workflow canvas just as any other node from other palettes. It is also available in the context menu and in the search function on the canvas.

> *Note: The code is saved "as is". When reusing the custom code nodes from the Custom Library palette, check the code for the dependencies on the source and target datasets in the new project context, link source dataset(s), specify target(s) and modify the code appropriately.*

To rename or remove any node in the Custom Library palette, right-click on the node and select **Rename** or **Remove** from the context menu:

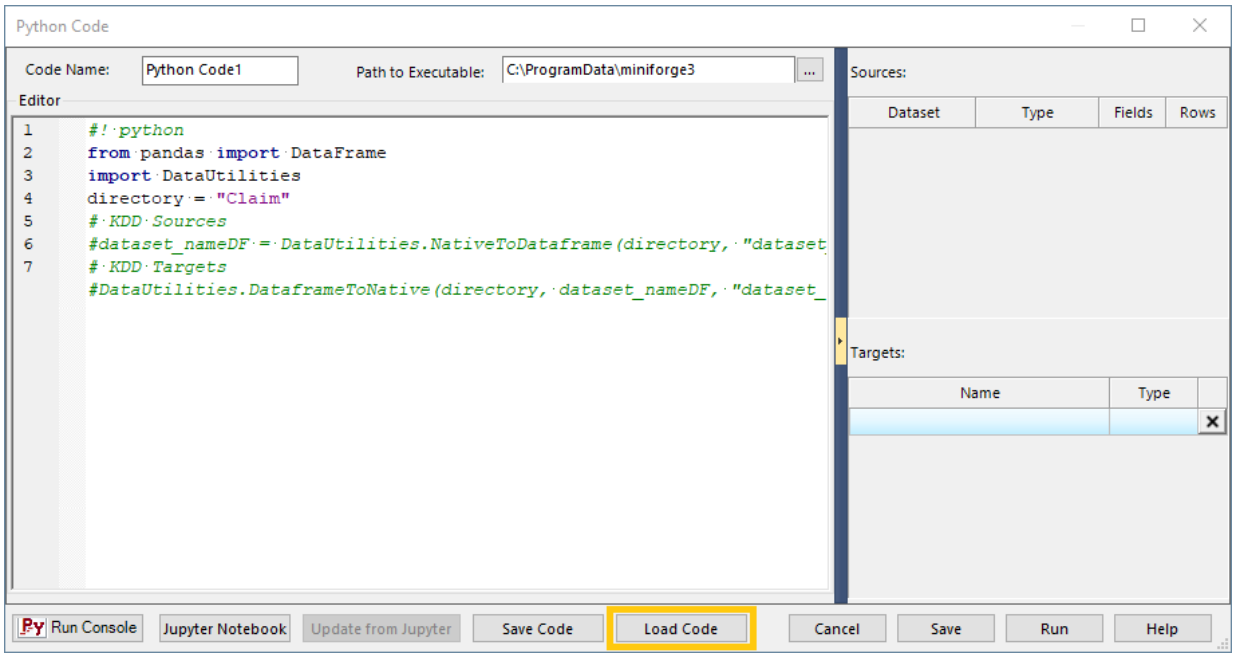


Loading code from Custom Library

The **Load Code** button in the code editor window loads previously saved custom code from the **Custom Library** or from a ***.ks_node** file (see, for example, the Python Code Node topic). This function can also be used to load code that others shared with you.

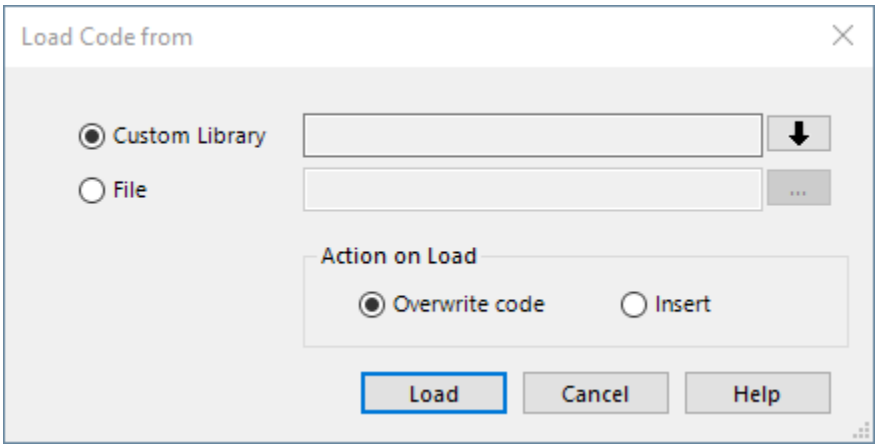
Add a new Python, R, or SAS Language Code node to your workflow or open an existing one on the workflow canvas. In the code editor window, click the **Load Code** button:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The "*Load Code from*" dialog opens.

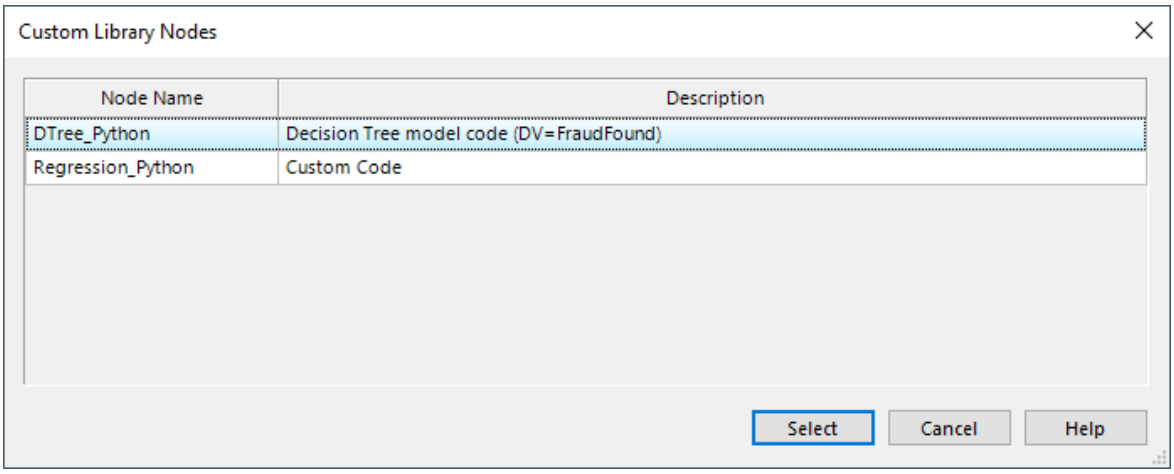
Select the **Custom Library** option if you want to load code from one of the nodes in the Custom Library palette.



Click the down arrow on the right to open the Custom Library Nodes list. It displays the code nodes available in the Custom Library palette. Choose the desired source node click **Select**.

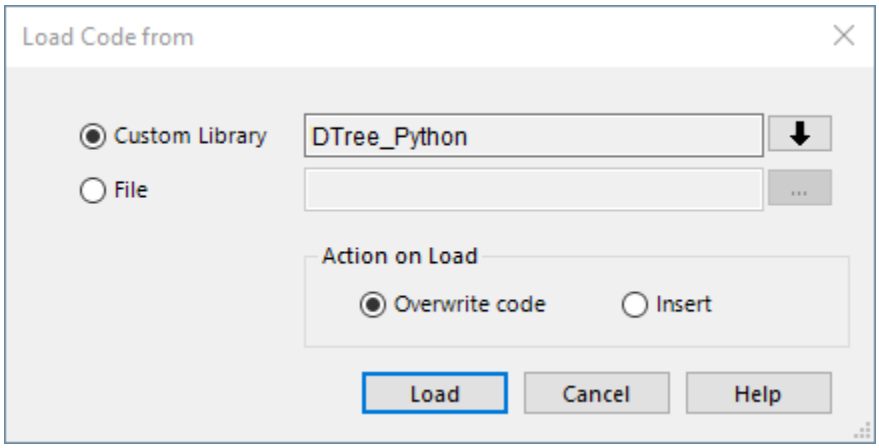
> Note: The default location for the Custom node templates in the file system is the folder `%APPDATA%\Altair\CustomLibrary` of the current user profile. When the application is started, this folder is scanned for valid templates, which are then displayed as nodes in the Custom Library palette.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



This brings you back to the "*Load Code from*" dialog.

Choose whether to **overwrite** any existing code in the editor window or to **insert** the new code at the current cursor position.



Click **Load** to load the code from the selected source.

Note: the code is loaded "as is". When reusing the code, check the code for the dependencies on the source and target datasets, link source dataset(s), specify target(s) and modify the code appropriately.

See also:

Python Code Node

R Code Node

SAS Language Code Node

Keras Model

The **Keras Deep Learning** node in the **Model** palette of Knowledge Studio provides an interface to the *Keras* library via integration with Python. Keras is a Python Deep Learning library designed to

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

enable fast experimentation with deep neural networks. For details see Keras documentation at https://keras.io/.

The **Keras Deep Learning** node wizard allows you to specify the necessary parameters and define the neural network configuration for Keras using graphical user interface. The wizard automatically translates the inputs into Python code that can be previewed before running the node. It also converts the input Altair KDD dataset(s) to Keras-readable format.

The model performance can be validated against the test data if any test data is selected. After training, the model can be scored on new data or validated using the Scoring node or Model Validation node.

Building a Keras Deep Learning Model

A Keras Deep Learning model requires one input dataset for model training. It can also use an optional second input dataset to guard the model against overfit. Training and test datasets can be created, for example, using the Partition node. The data types of the variables in the training and test datasets must match.

A **Keras Deep Learning** node can be added to the workflow in one of the following ways:

- Drag a connection arrow from the training dataset node to any area on the Workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Model --> Keras Deep Learning** as in the example screenshot below. A new **Keras Deep Learning** node will be created. Optionally, connect a test dataset to the new Keras Deep Learning node.

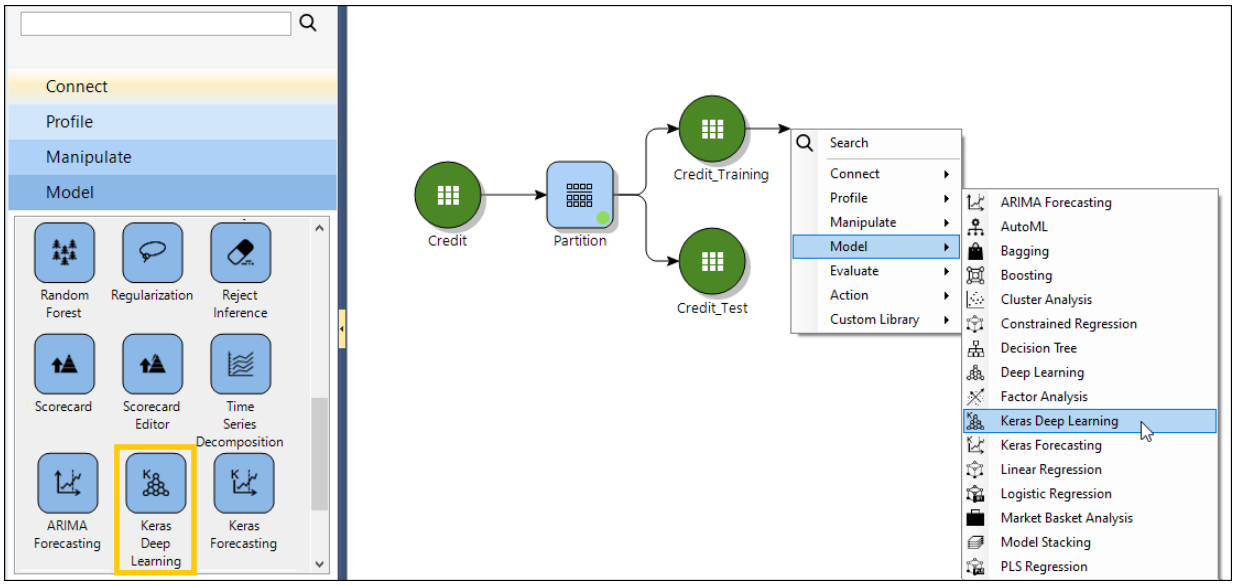
OR

- Right-click on the Workflow canvas and select **Model --> Keras Deep Learning** from the context menu. Connect the training dataset to the Keras Deep Learning node. Optionally, connect a test dataset to the same model node.

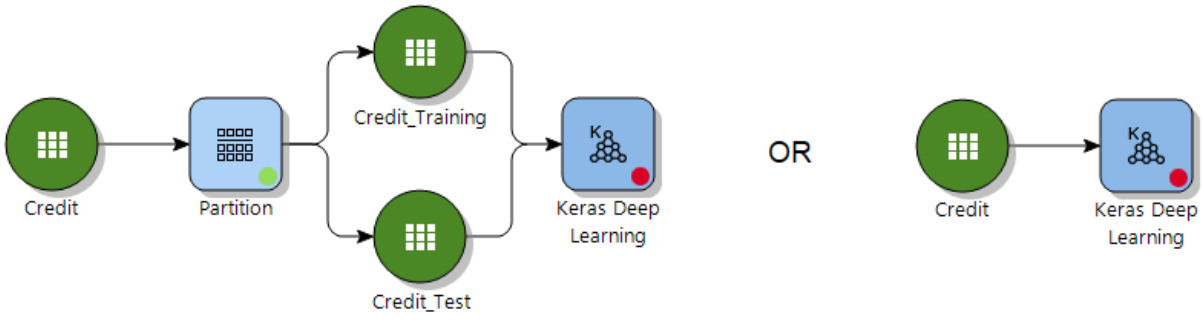
OR

- Open the **Model** palette and drag the **Keras Deep Learning** node onto the workflow canvas. Connect the training dataset to the Keras Deep Learning node. Optionally, connect a test dataset to the same model node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



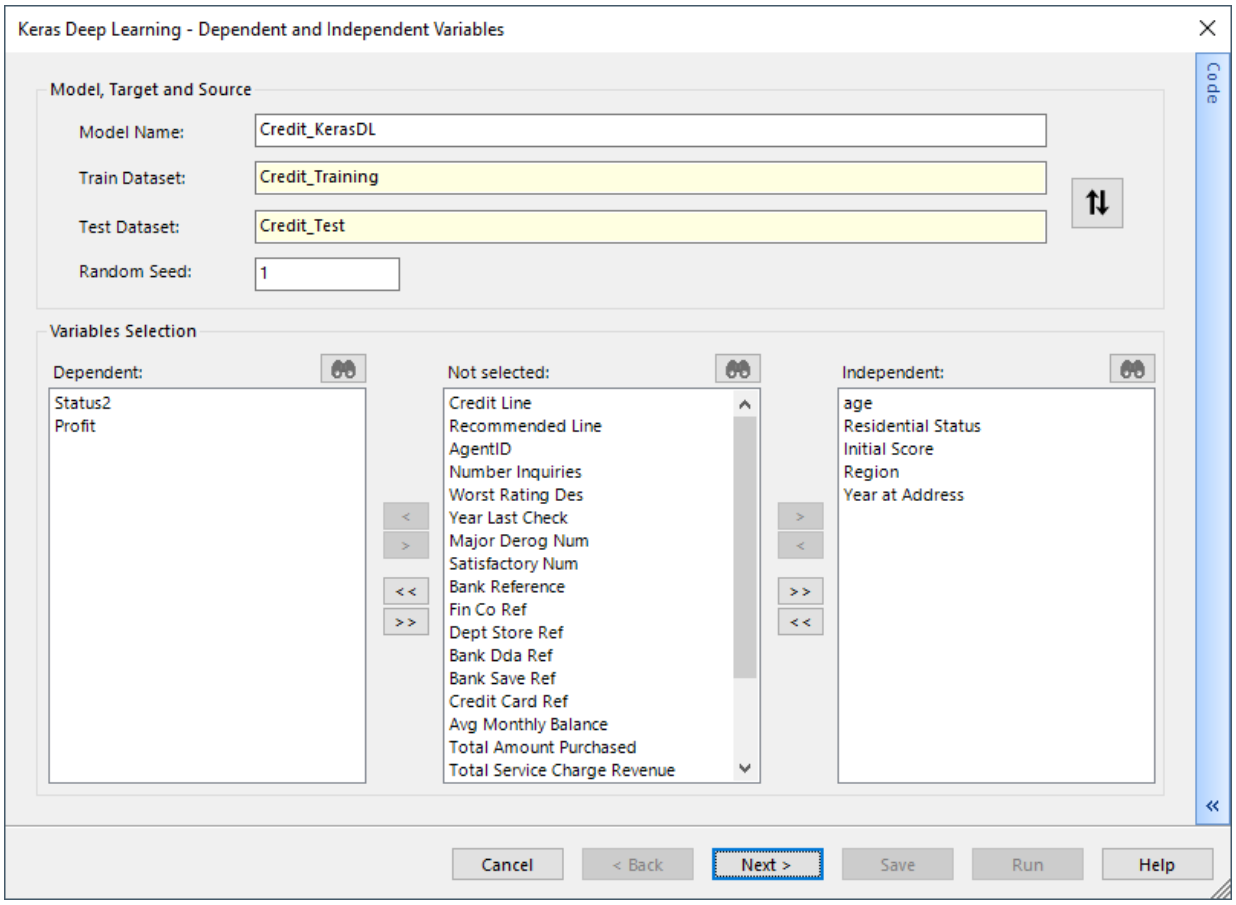
Double-click the Keras Deep Learning node or right-click on it and select **Modify** to open the wizard. This example shows two options for connecting the nodes:



In the first page of the wizard:

- Specify the model name
- To change the order of the Train and Test input datasets (if applicable), click the arrows button on the right-hand side.
- Change the Random Seed value if required for repeatability.
- Select the dependent variable(s) using the arrow buttons to move them to the *Dependent* list on the left.
- Select the independent variables using the arrow buttons to move them to the *Independent* list on the right.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



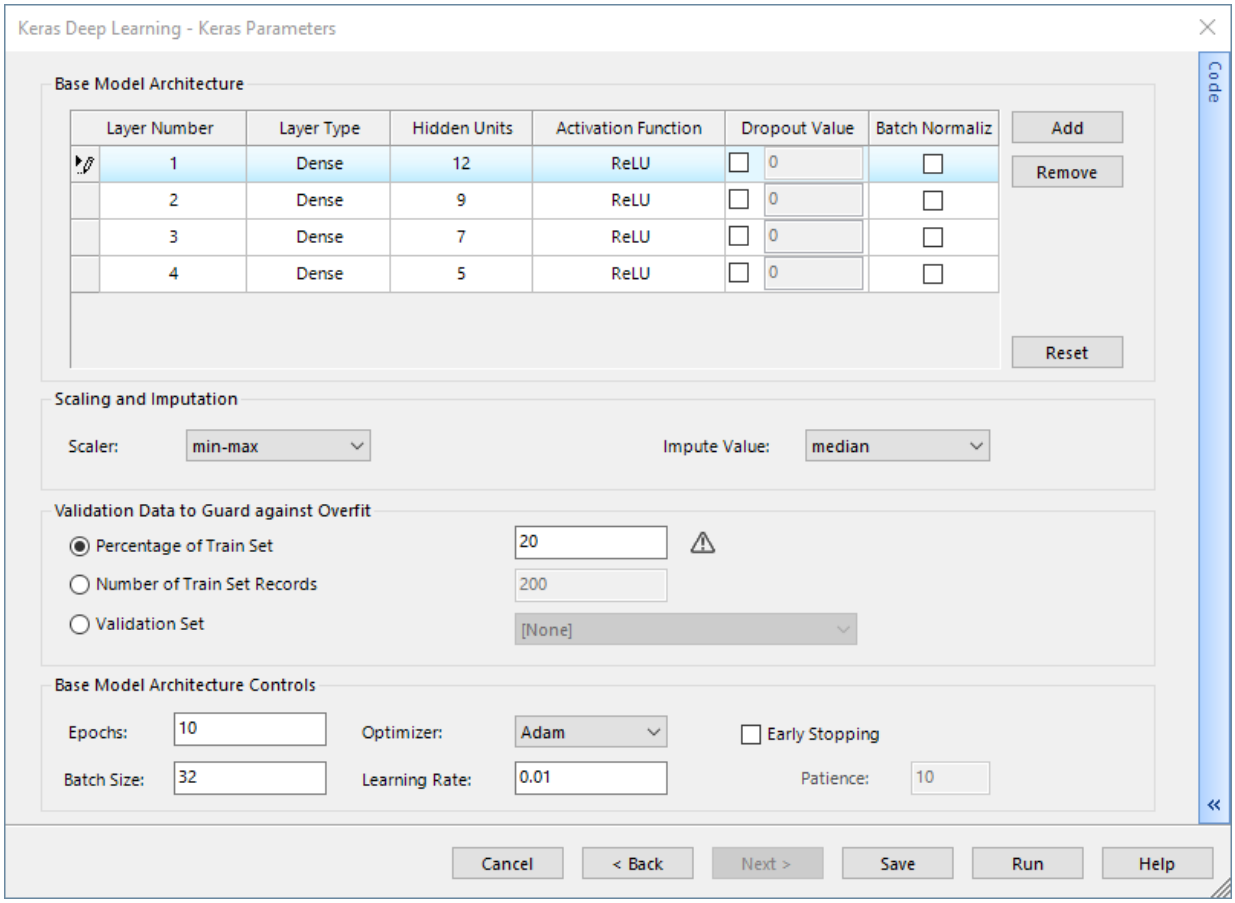
Click **Next**.

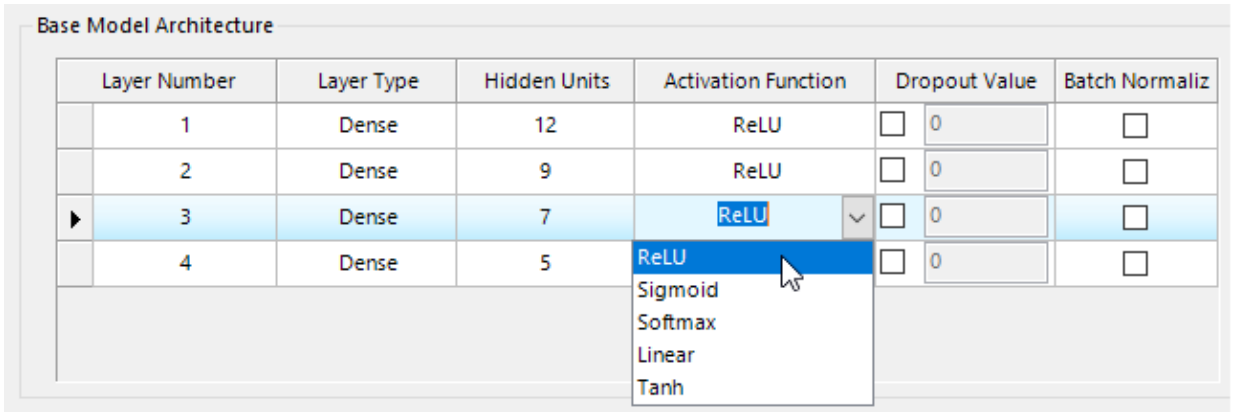
In the Keras Parameters page, configure the model and specify the following parameters as needed:

- Define the neural network configuration (Base Model Architecture) by adding layers and specifying their properties. To remove a layer, select it and click **Remove**. The **Reset** button removes all layers except the first one. The layer properties are:
  - ***Layer type*** (*Dense*, *LSTM*, *GRU*, or *Bidirec*)
  - The number of ***hidden units***
  - ***Activation function***. The available activation functions are *ReLU (rectified linear unit)*, *Sigmoid*, *Softmax*, *Linear*, and *Tanh*. Select the desired function from the drop-down list for each layer.
  - ***Dropout value***. Specify the dropout rate (a number between 0 and 1) - the frequency at which the input units are randomly set to 0.
  - ***Batch normalization***
- In the section *Validation Data to Guard against Overfit*, specify the portion of the training dataset to be allocated for testing to guard against overfit (percentage or the exact number of records). Note that this decreases the number of records for training. The default is 20% allocated for testing. Alternatively, select the Validation Set option to use the Test dataset (if connected to the model node).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

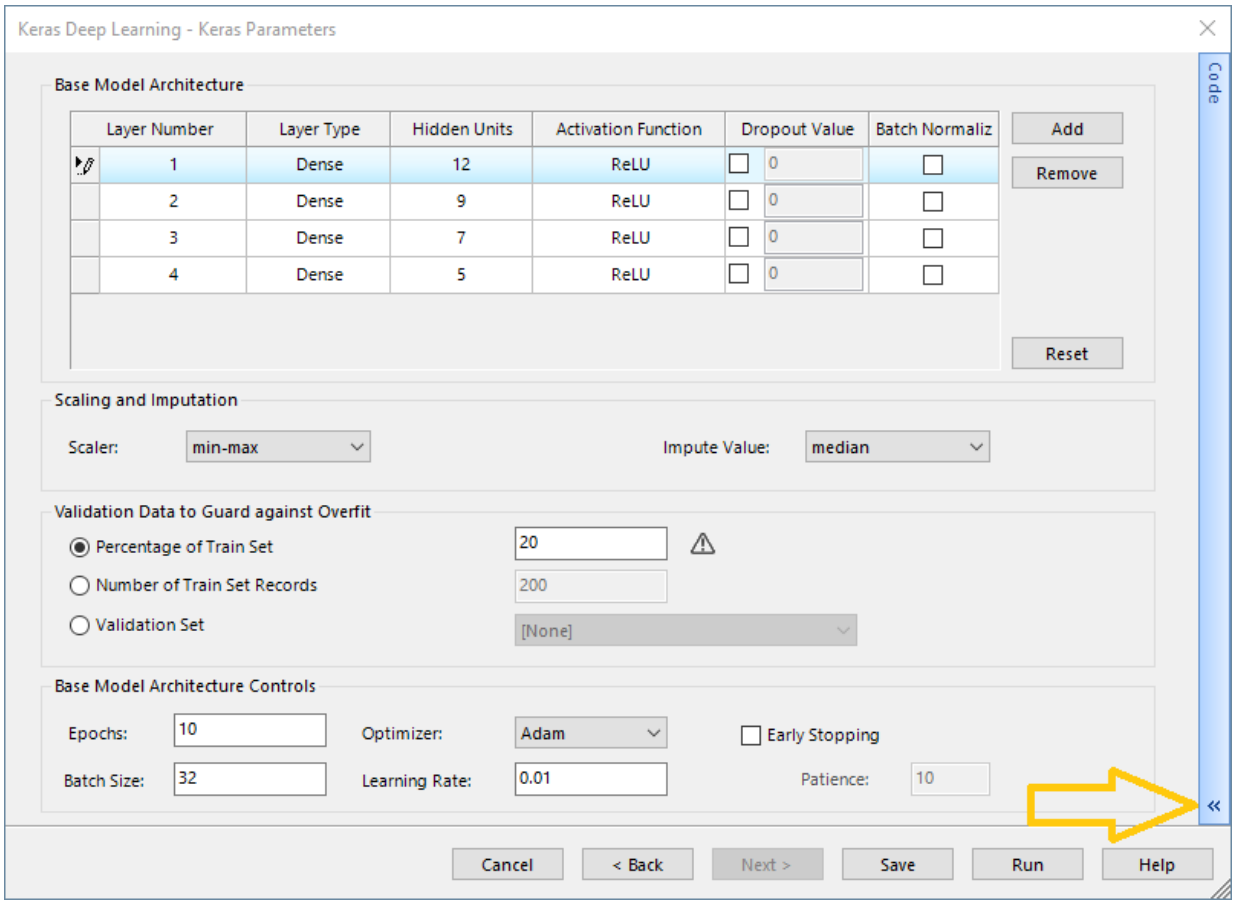
- In the *Base Model Architecture Controls* section, specify:
  - Number of epochs (training iterations); the default is 10
  - Batch size; the default is 32
  - Optimizer. Available optimizers are *Adam, RMSprop,* and *SGD. T*he default is *Adam*.
  - Learning Rate. The default is 0.01.
  - Early Stopping. If this option is selected, specify the *Patience* parameter value (the default is 10).





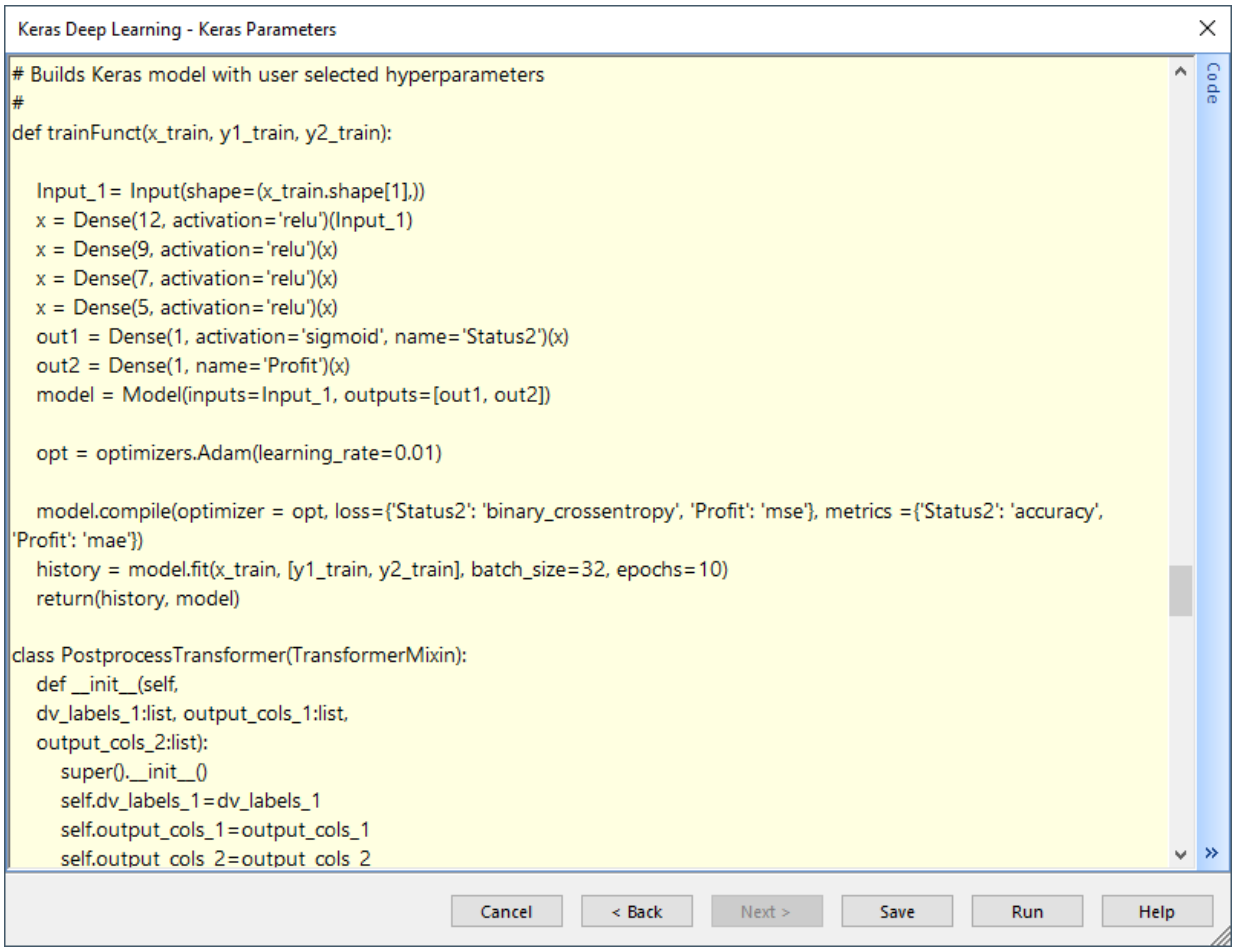
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog.

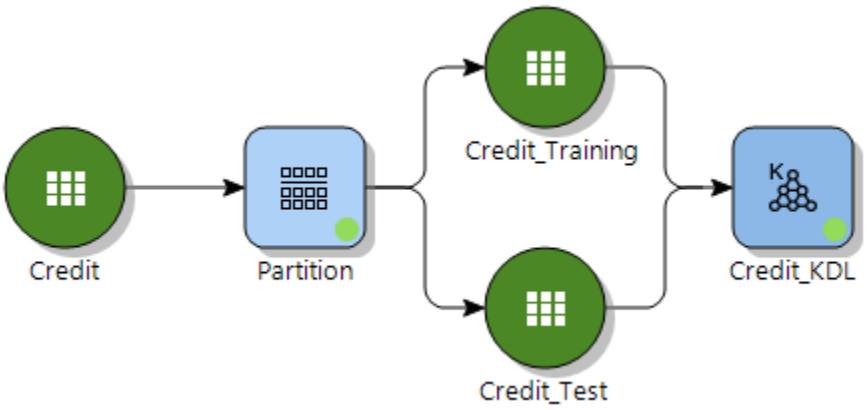


This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

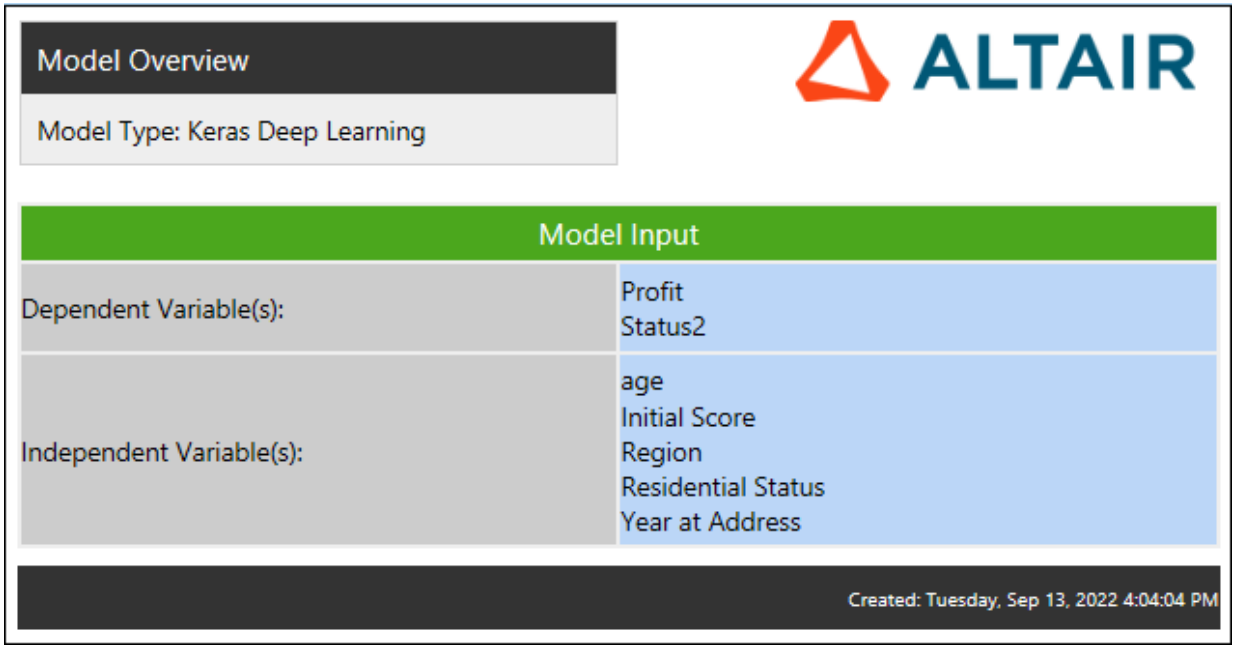


Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. After running the node, double-click the model node to view the results.



Double-click on the Keras Deep Learning node to view the training results report and model performance plots. The **Results** tab shows the list of dependent and independent variables.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Model Overview

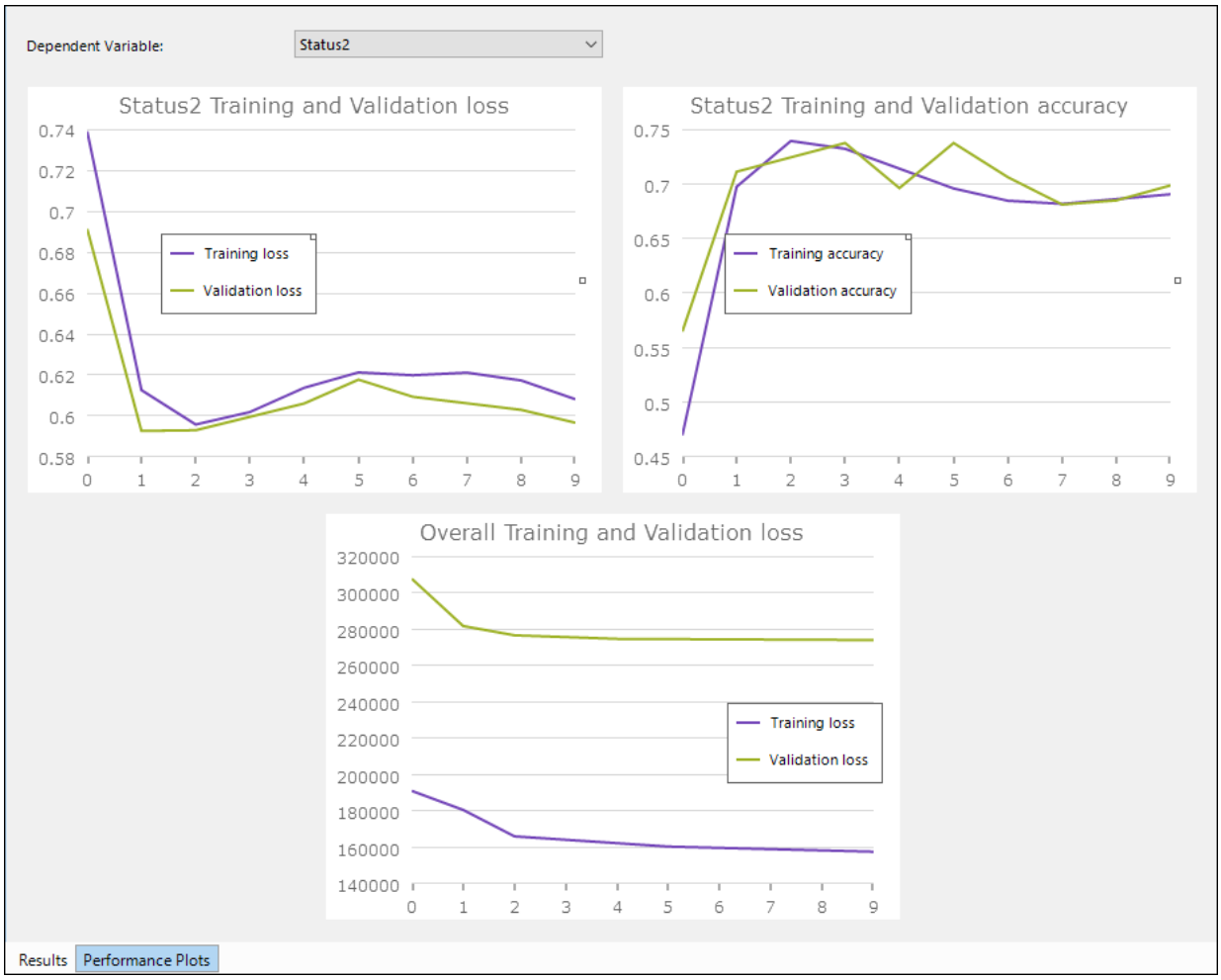
Model Type: Keras Deep Learning

Model Input	
Dependent Variable(s):	Profit Status2
Independent Variable(s):	age Initial Score Region Residential Status Year at Address

Created: Tuesday, Sep 13, 2022 4:04:04 PM

The **Performance Plots** tab shows the training loss and accuracy plots. Validation loss and accuracy plots are shown if any data was specified for *Validation Data to Guard against Overfit* in the Keras Parameters page. If you trained the model with multiple dependent variables, select the desired variable from the drop-down list at the top of the Performance Plots frame.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



After training, the model can be used for scoring new data or validating test data. See also the Scoring topic.

See also: Model Analyzer

Keras Forecasting

Overview

The **Keras Forecasting** node in the **Model** palette of Knowledge Studio performs time-series forecasting with ***Keras*** - a deep learning API written in Python, running on top of the machine learning platform TensorFlow. For details, see Keras documentation at https://keras.io/. Keras copyright and license information is available at https://github.com/keras-team/keras.

The **Keras Forecasting** node wizard provides a convenient interface to specify the parameters and define the neural network configuration for Keras time-series forecasting. The wizard automatically translates the inputs into Python code, which can be previewed before running the node. The input Altair datasets are converted from the native KDD format to Keras-readable format in the process.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Building a Keras Forecasting Model

The input dataset should contain at least one Numeric, Integer, Date, or Timestamp field with no missing values to serve as time index. It should also contain at least one field with the series data. If necessary, use the Make Time Series node to aggregate the data to the desired period level to prepare it for Keras Forecasting.

In the examples below, data aggregated at the monthly level is used as input to the Keras Forecasting model.

Use one of the following ways to add a Keras Forecasting node to your workflow:

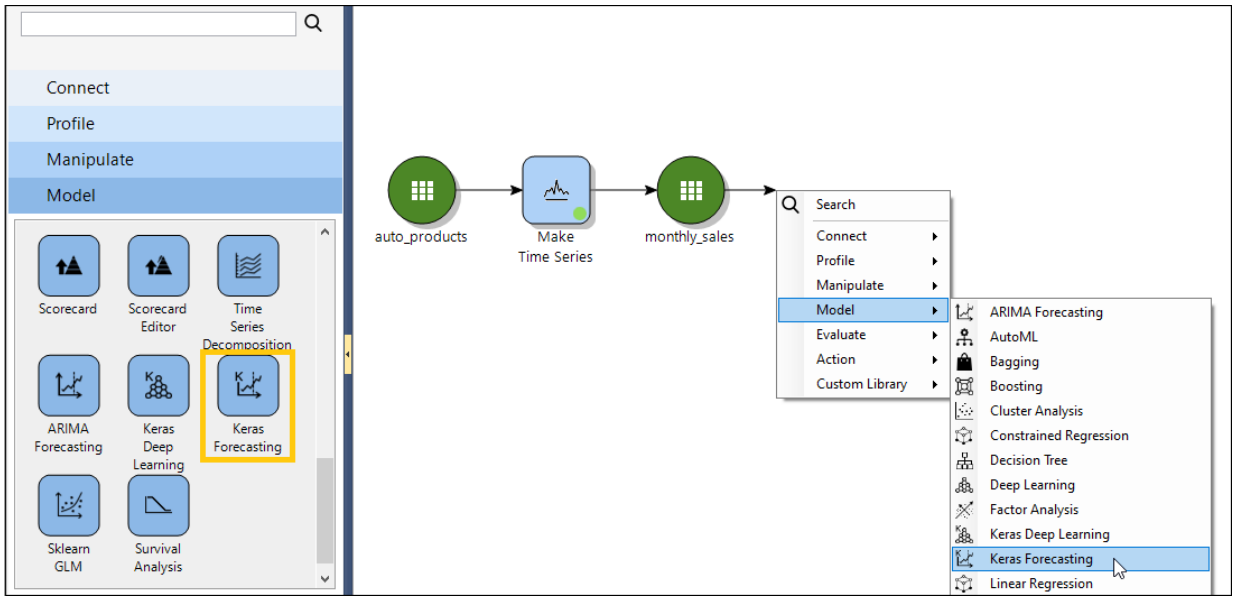
- Drag a connection arrow from the input dataset to any area on the Workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Model --> Keras Forecasting** as in the example screenshot below. A new **Keras Forecasting** node will be created, and the wizard will open.

  OR

- Right-click on the Workflow canvas and select **Model --> Keras Forecasting** from the context menu. Connect the input dataset node to the Keras Forecasting node.

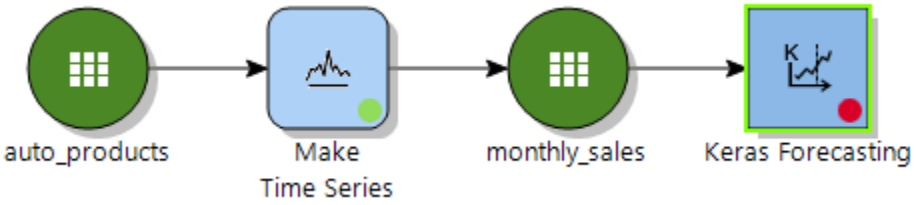
  OR

- Open the **Model** palette and drag the **Keras Forecasting** node onto the workflow canvas. Connect the input dataset node to the Keras Forecasting node.



Double-click the Keras Forecasting node or right-click on it and select **Modify** to open the wizard.

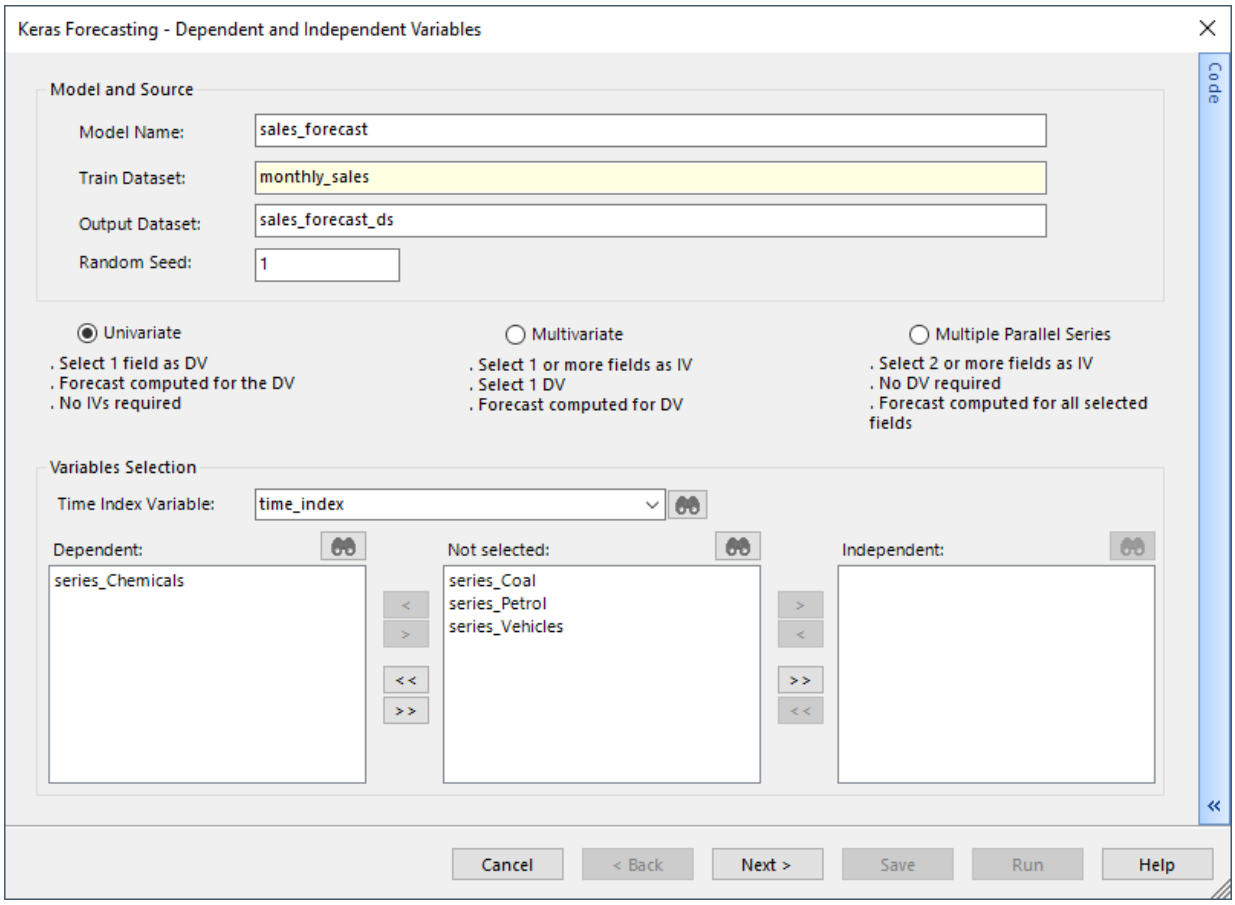
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



In the first page of the wizard:

- Specify the model name
- Specify the name for the output dataset
- Change the Random Seed value if required for repeatability.
- Select the forecast type:
  - *Univariate*: Only one dependent variable can be selected to be forecasted for one or multiple steps into the future.
  - *Multivariate*: One or more independent variables (IV) and one dependent variable (DV) can be selected to forecast the DV for one or multiple steps into the future.
  - *Multiple Parallel Series*: One or more independent variables can be forecasted for one or multiple steps into the future. No dependent variable required. (Any variables placed in the *Dependent* list will be ignored.)
- Select the *Time Index Variable.* This must be a Date, Timestamp, Number or Integer variable with no missing values.
- Select the dependent variable(s) using the arrow buttons to move them to the *Dependent* list on the left.
- Select the independent variables using the arrow buttons to move them to the *Independent* list on the right.

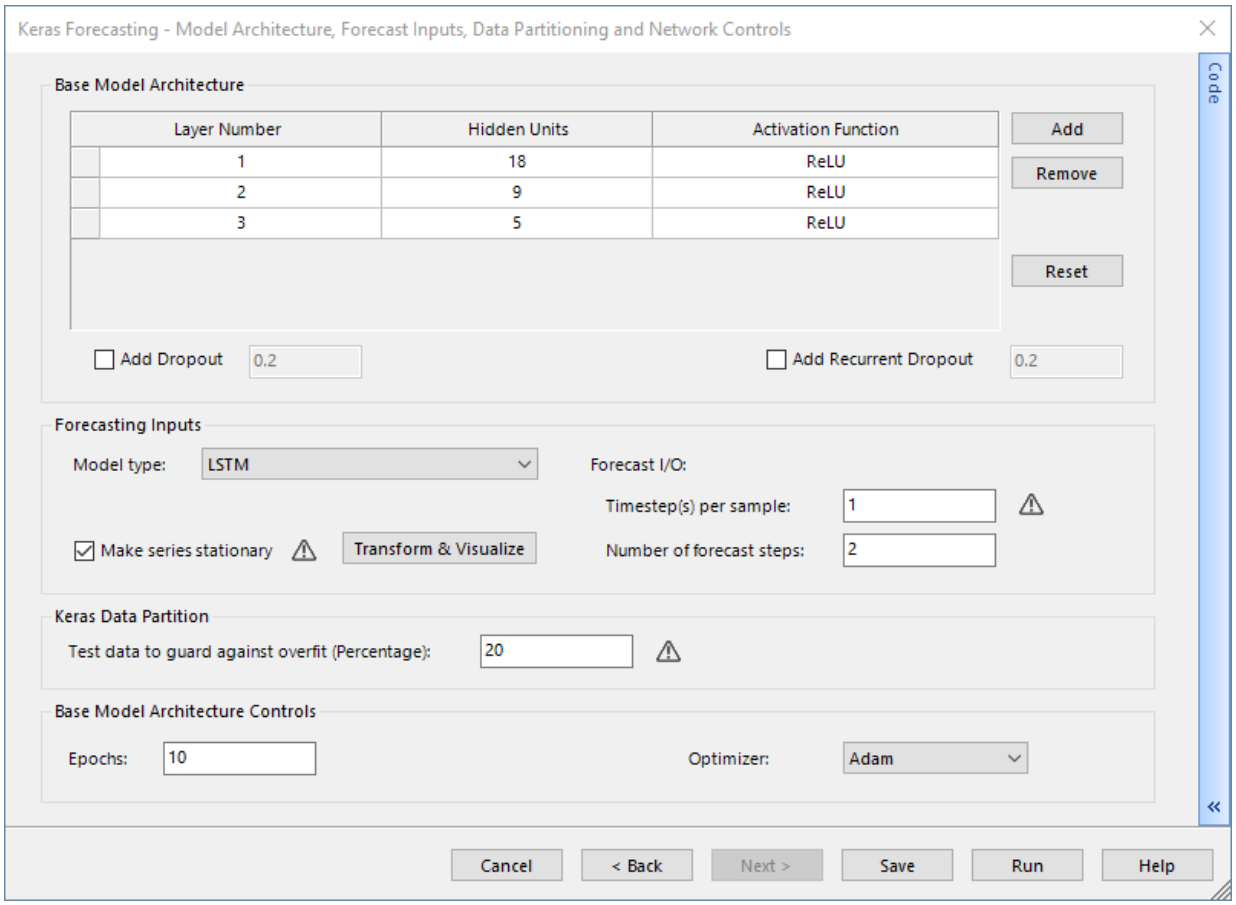
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

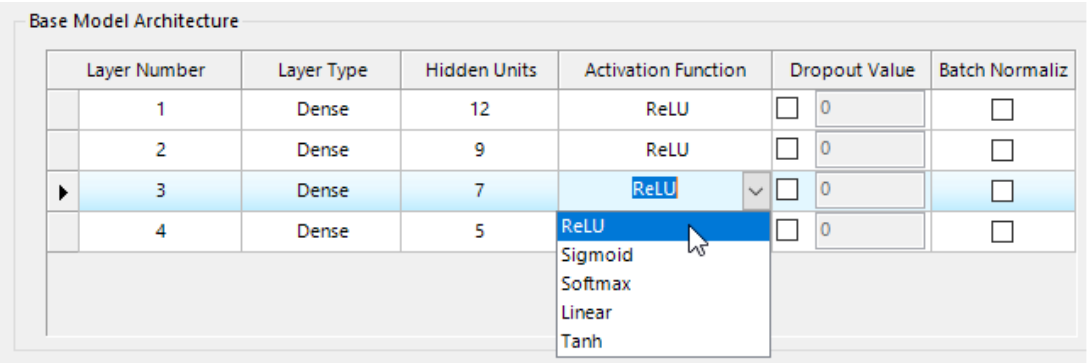
In the next page, specify the model architecture, forecast parameters, data partitioning, and network controls:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- Define the neural network configuration (Base Model Architecture) by adding layers and specifying their properties: the number of **hidden units** and **activation function**. The available activation functions are *ReLU (rectified linear unit)*, *Sigmoid*, *Softmax*, and *Linear*. Select the desired function from the drop-down list for each layer. The **Remove** button removes the selected layer. The **Reset** button removes all layers except the first one.

  Activation function options:

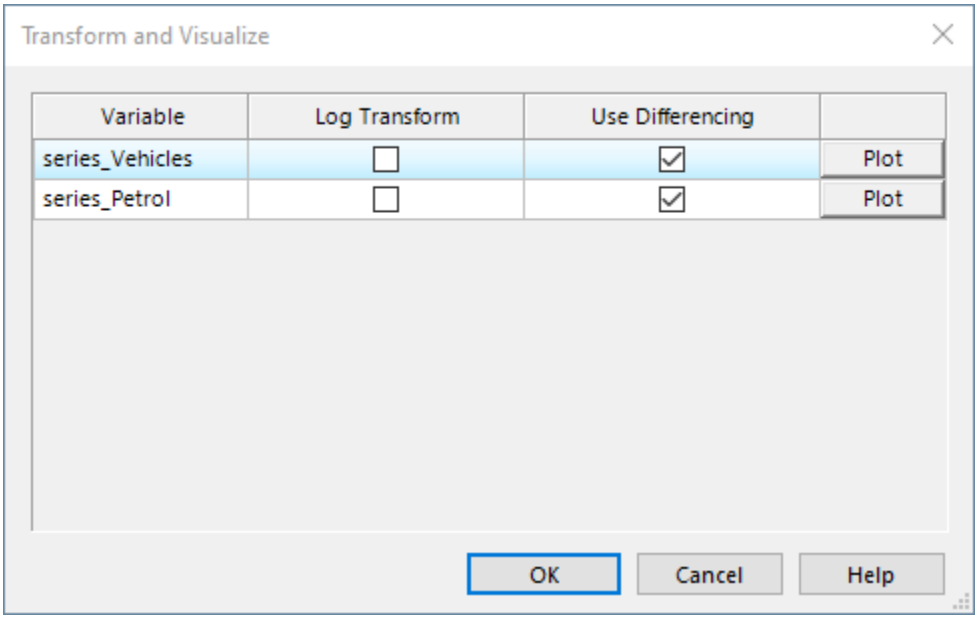


- Select *Add Dropout* and specify the dropout rate - the frequency at which the input units are randomly set to 0 to prevent overfitting. Enter a number between 0 and 1. The default is 0.2.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Select *Add Recurrent Dropout* to improve overfit prevention if necessary. Unlike regular dropout, the recurrent dropout only removes inputs between time steps rather than between layers. Enter a number between 0 and 1. The default is 0.2.

- In the ***Forecasting Inputs*** section:
  - Choose the forecast model type:
    - *LSTM* (Long Short-Term Memory)
    - *Bidirectional LSTM*
    - *GRU* (Gated Recurrent Unit)
  - *Make series stationary*: If you deselect this option, no transformations will be applied. This option uses the augmented Dickey Fuller test to decide if the series must be made stationary.
    - Note: For the *Multivariate* forecast type, no transformations are applied to the independent variables.
  - Click *Transform & Visualize* to define transformations (applicable to the *Univariate* and *Multiple Parallel Series* forecast types).

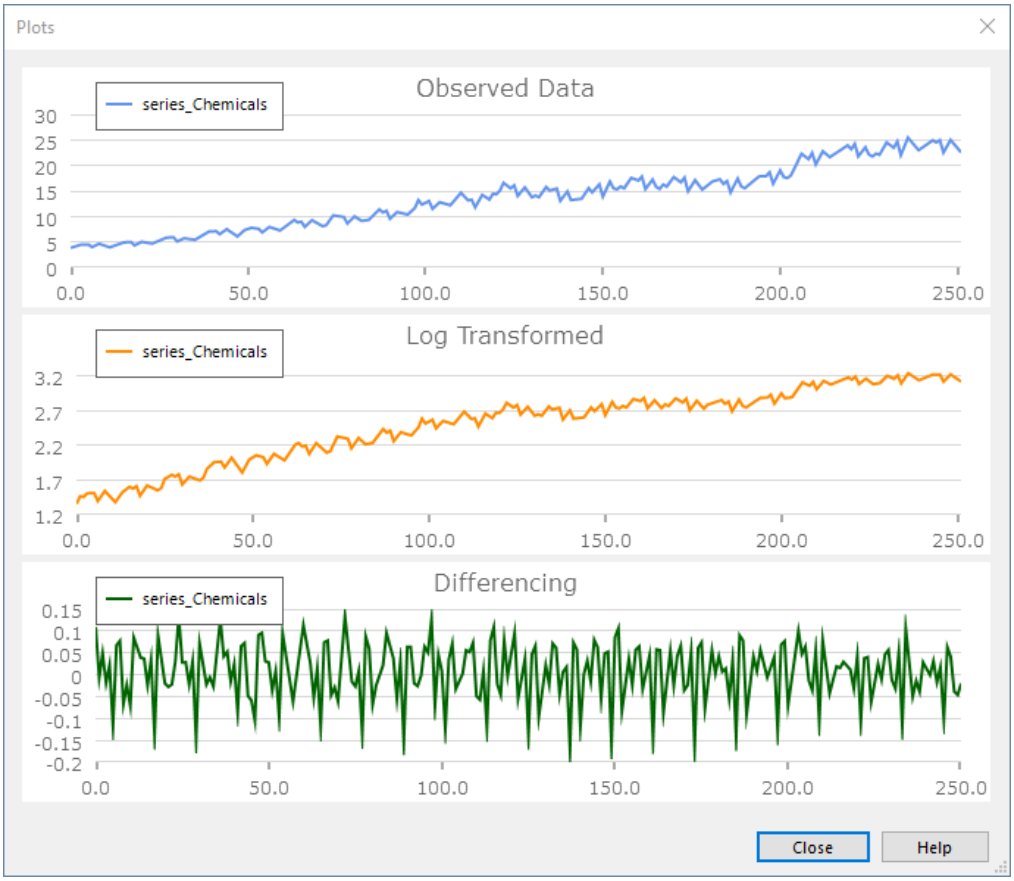


    - The *Differencing* option is selected by default. If this function does not return a plot, you will get a message “Data is already stationary“. Otherwise the observed data and differenced data are plotted one below the other in a pop-up window when you click the **Plot** button. If the *Log Transform* option was selected, the log-transformed plot is displayed as well.
    - Selecting the *Log Transform* option also automatically enables *Differencing*. The log transformation is applied first, then differencing is automatically applied.
    - Click the **Plot** button for the desired series to plot the observed data, log-transformed data, and differenced data one below the other. If the log transformation is not applicable because the data is outside of the range of

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

admissible values, then the message "*Data is not suitable for log transformation*" is displayed. In this case only differencing is applied.

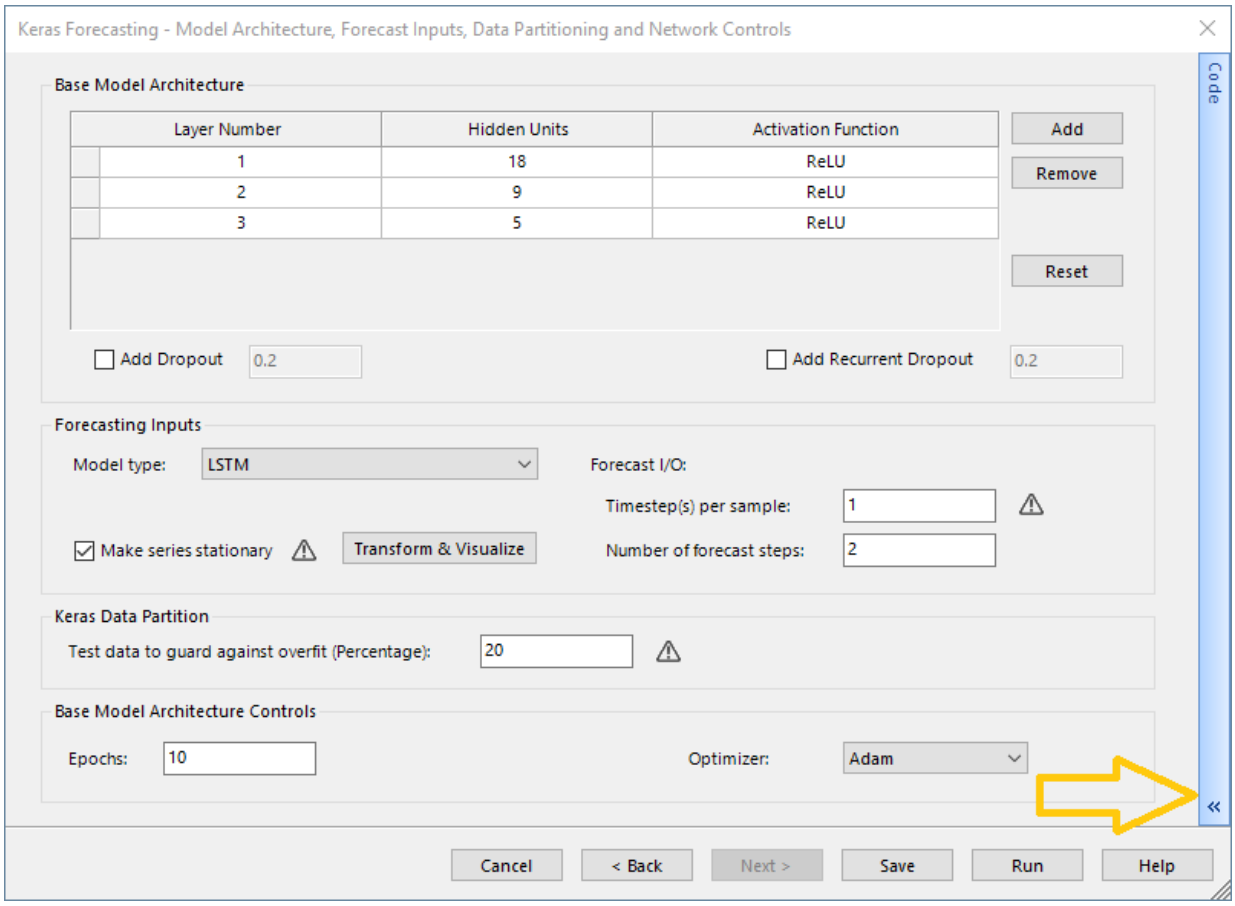
- The output from this step is used as the input for modelling. If a plot is available after differencing, then the differenced data is used as input for modelling, otherwise the log transformed data is used. If you deselect the option "Make series stationary", then the observed data is used for modelling.



  - Forecast I/O options:
    - *Timesteps per sample*: Specify the number of rows (lag values) used per sample. The default is 1.
    - *Number of forecast steps:* Specify the number of time steps produced as the forecast output. The default is 2.
- In the ***Keras Data Partition*** section, specify the percentage of the input dataset to be allocated for testing to guard against overfit. Note that this decreases the number of records for training. The default is 20% allocated for testing.
- In the ***Base Model Architecture Controls*** section, specify:
  - Number of epochs (training iterations); the default is 10
  - Optimizer (the available optimizers are *Adam* and *RMSprop;* the default is *Adam*).

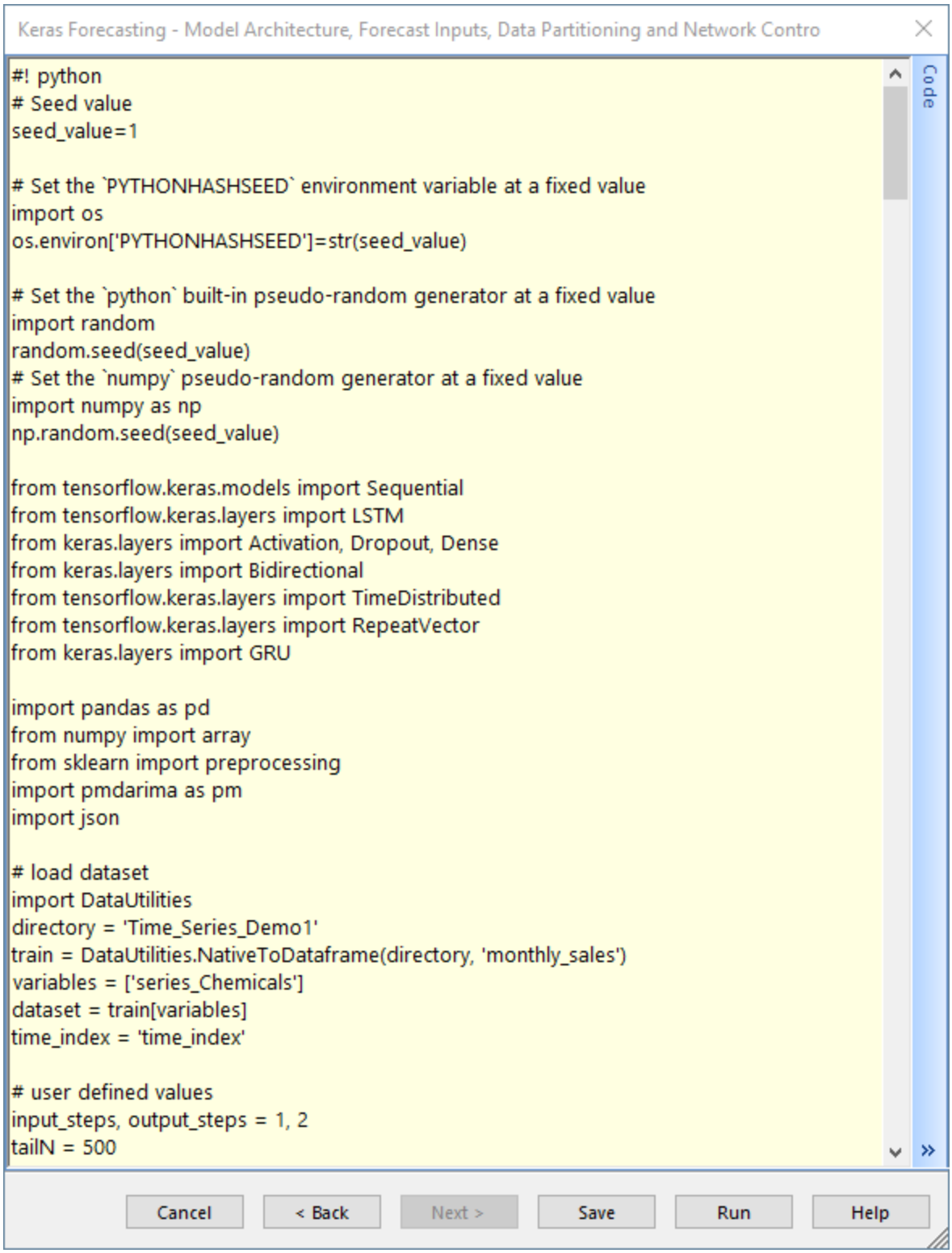
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog.



This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

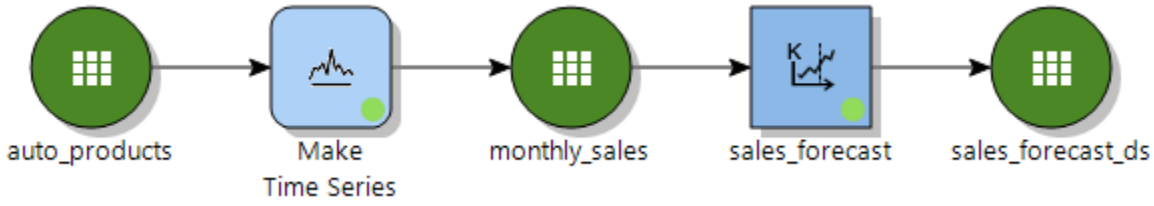


Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. The input dataset is converted to a Pandas dataframe and placed in a TensorFlow session. The defined Keras Forecasting model runs in a built-in Python environment, loading all required

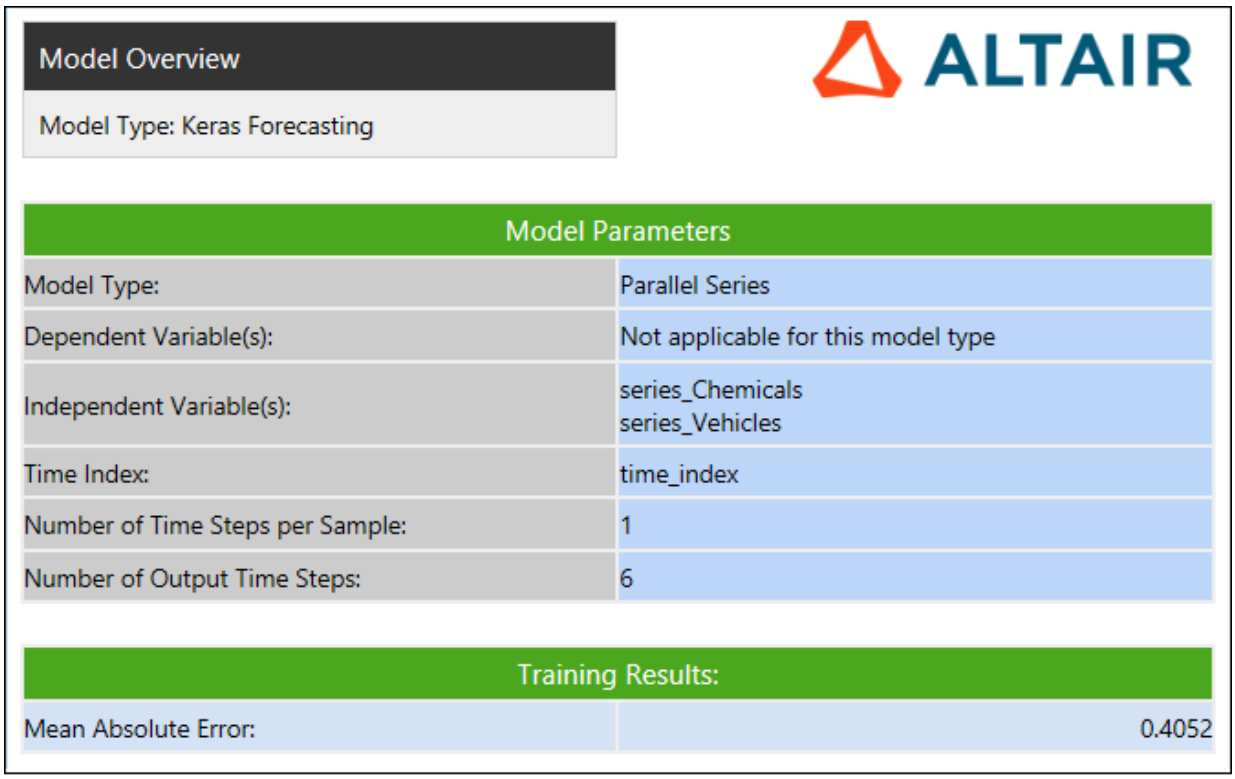
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

packages and their dependencies. The output dataset is converted from Pandas to the Altair KDD format.

After running the node, double-click the forecast node and the output dataset node to view the results.



Double-click on the Keras Forecasting node to view the summary report, performance plots, and forecast plots. The following examples are for Multiple Parallel Series.



Model Overview

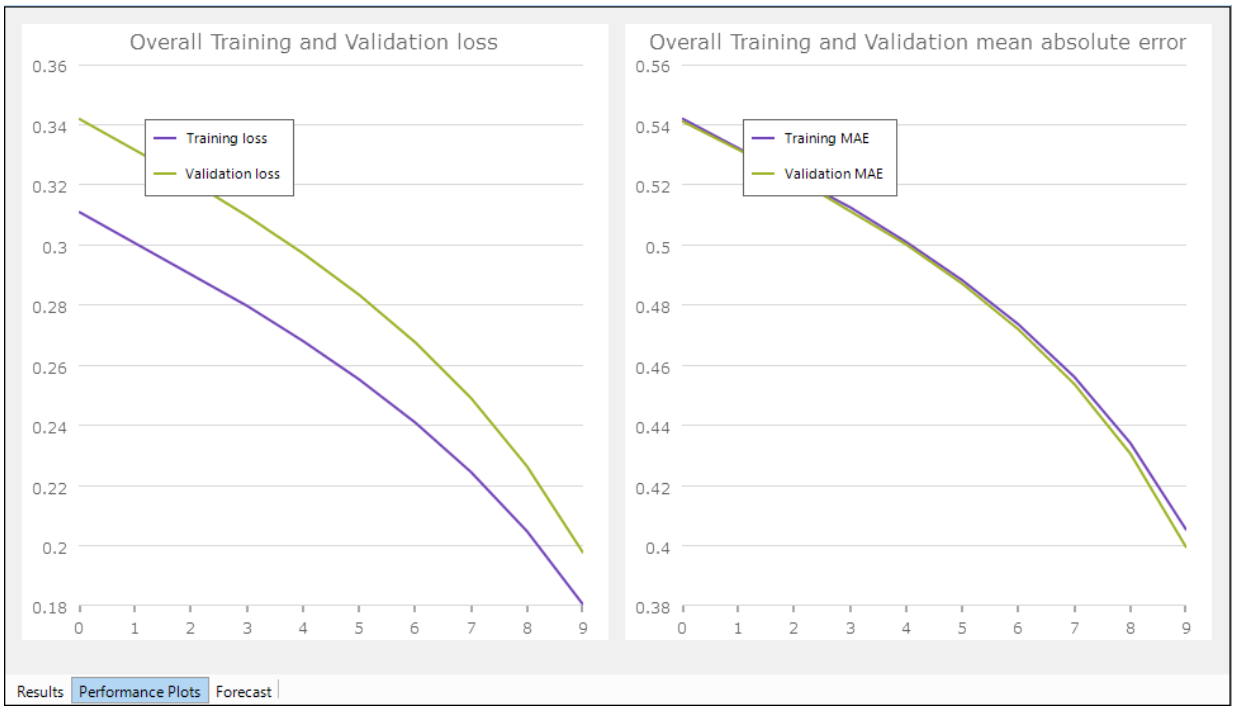
Model Type: Keras Forecasting

Model Parameters	
Model Type:	Parallel Series
Dependent Variable(s):	Not applicable for this model type
Independent Variable(s):	series_Chemicals series_Vehicles
Time Index:	time_index
Number of Time Steps per Sample:	1
Number of Output Time Steps:	6

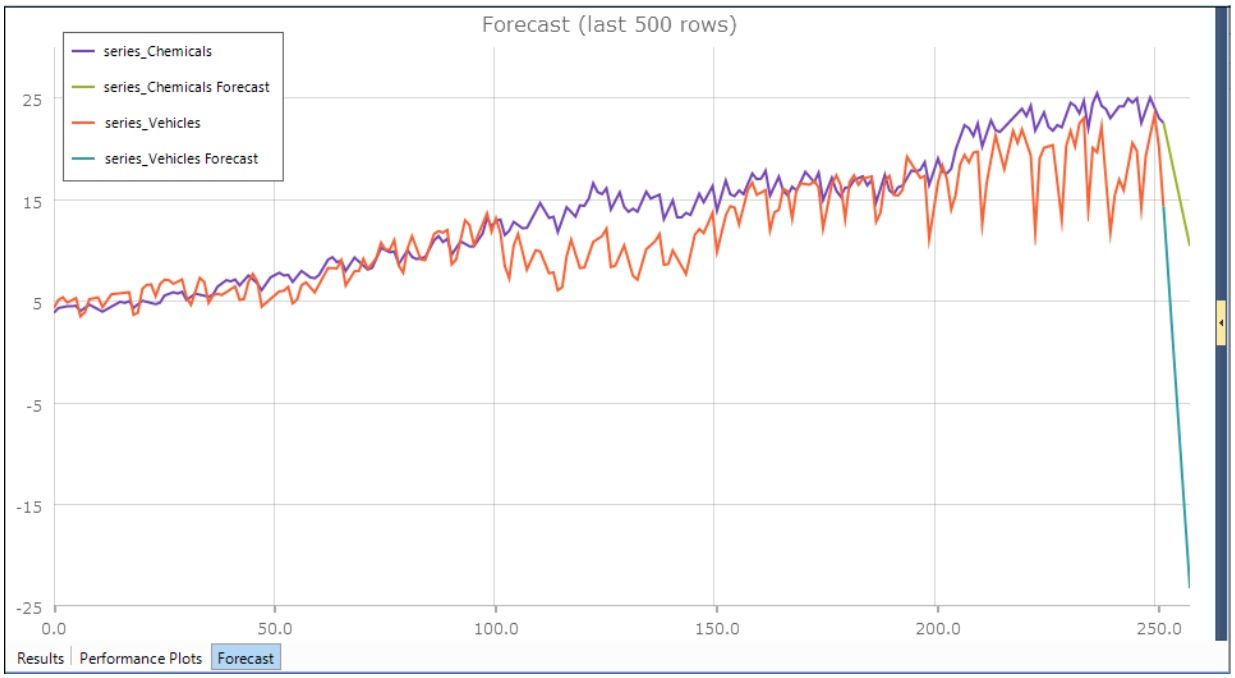
Training Results:	
Mean Absolute Error:	0.4052

Performance plots provide a way to validate the time series model. Click the **Performance Plots** tab to view the training and validation loss and accuracy plots. The second plot shows the mean absolute error (MAE) of the model on the training and validation sets.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

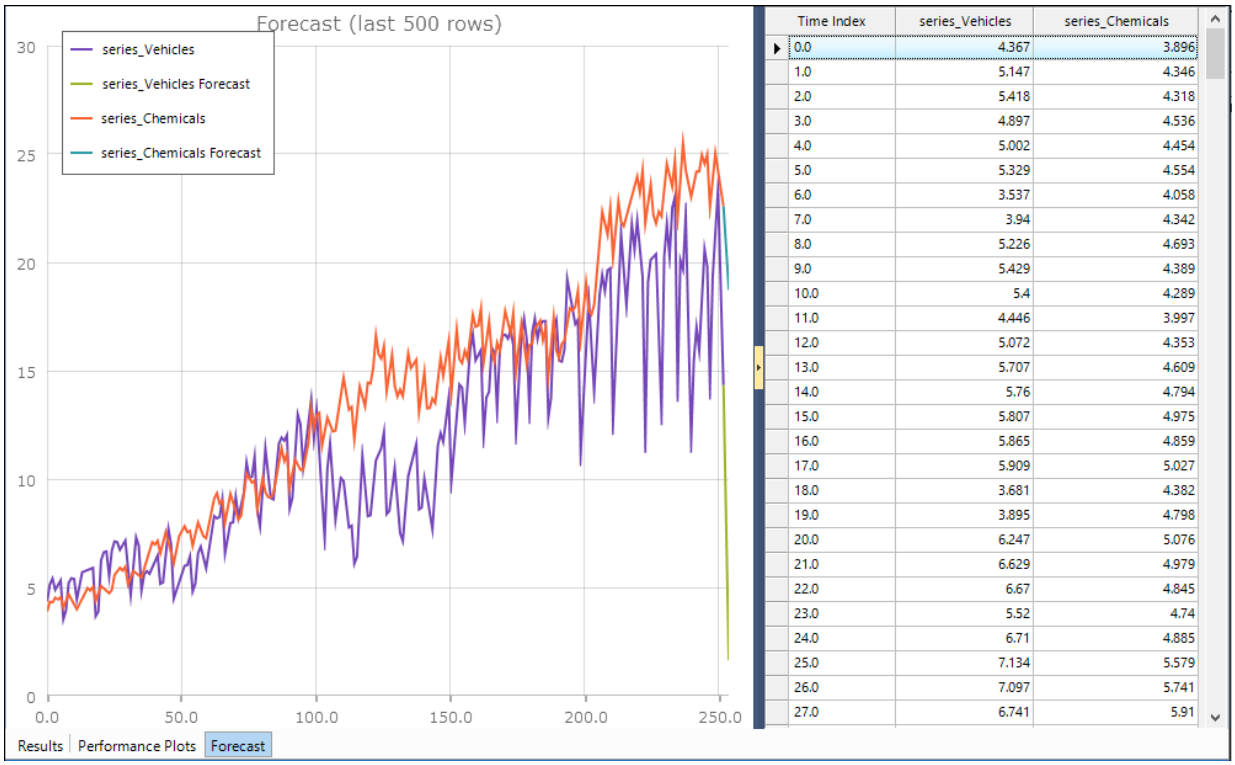


Click the **Forecast** tab. The plot shows the input data points extended with forecast data for the number of forecast steps specified in the wizard. The legend at the top left shows the color codes for the input data and forecast data for each series.



To view the data table that the forecast plot is based on, click on the small arrow in the middle of the blue bar on the right-hand side. The table will open in a frame on the right. Click the arrow again to hide the frame.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Time Index	series_Vehicles	series_Chemicals
0.0	4.367	3.896
1.0	5.147	4.346
2.0	5.418	4.318
3.0	4.897	4.536
4.0	5.002	4.454
5.0	5.329	4.554
6.0	3.537	4.058
7.0	3.94	4.342
8.0	5.226	4.693
9.0	5.429	4.389
10.0	5.4	4.289
11.0	4.446	3.997
12.0	5.072	4.353
13.0	5.707	4.609
14.0	5.76	4.794
15.0	5.807	4.975
16.0	5.865	4.859
17.0	5.909	5.027
18.0	3.681	4.382
19.0	3.895	4.798
20.0	6.247	5.076
21.0	6.629	4.979
22.0	6.67	4.845
23.0	5.52	4.74
24.0	6.71	4.885
25.0	7.134	5.579
26.0	7.097	5.741
27.0	6.741	5.91

See also: Keras Deep Learning, Keras Scoring.

Keras Deep Learning Scoring

A Keras Deep Learning model is scored using the **Scoring** node in the Action palette. For general information about scoring, see the topic About Scoring.

Prerequisites for Keras model scoring: a **Keras Deep Learning** node and an input dataset to score. The input Keras Deep Learning node stores the Keras Deep Learning in HDF5 format.

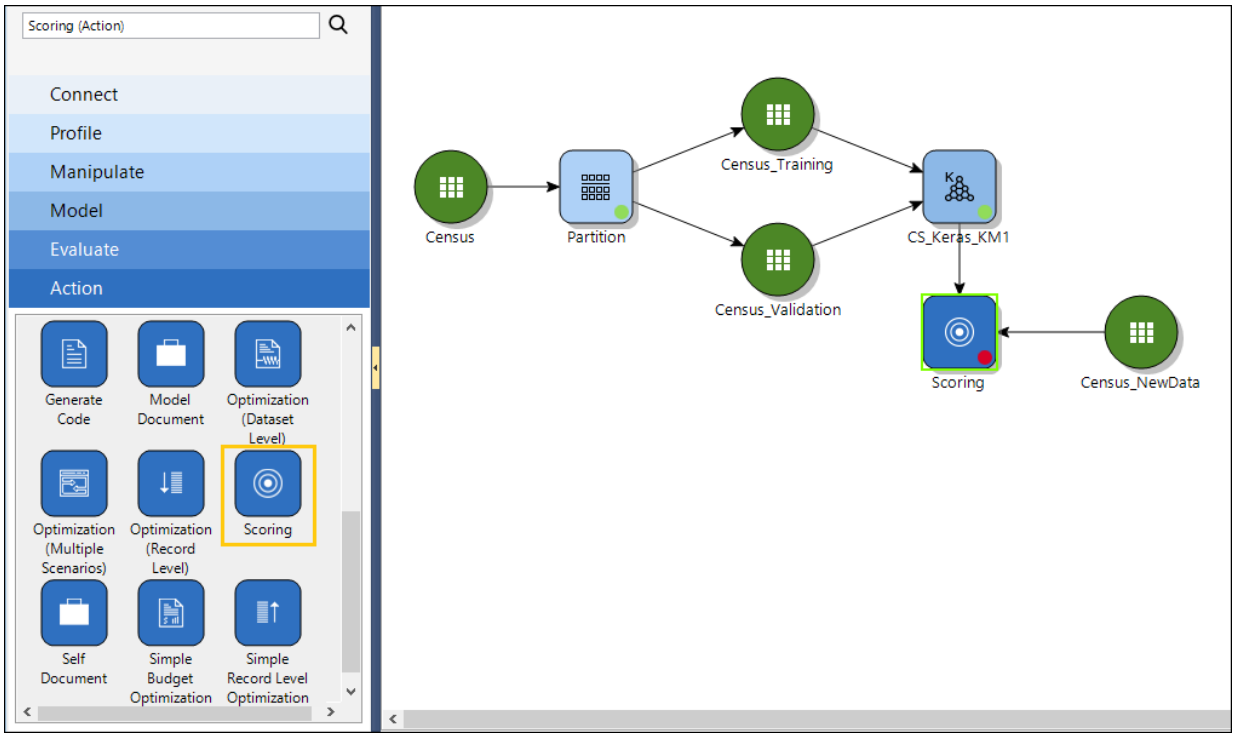
To add a Scoring operation to your workflow:

- Drag the **Scoring** node from the **Action** palette onto the Workflow canvas and connect the inputs: a **Keras Deep Learning** node and a dataset node that contains the data to score. See the example below.

OR

- Right-click anywhere on the workflow canvas and select the context menu command **Action --> Scoring**. This will create a new Scoring node. Connect the inputs: a Keras Deep Learning model node and a dataset node that contains the data to score.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

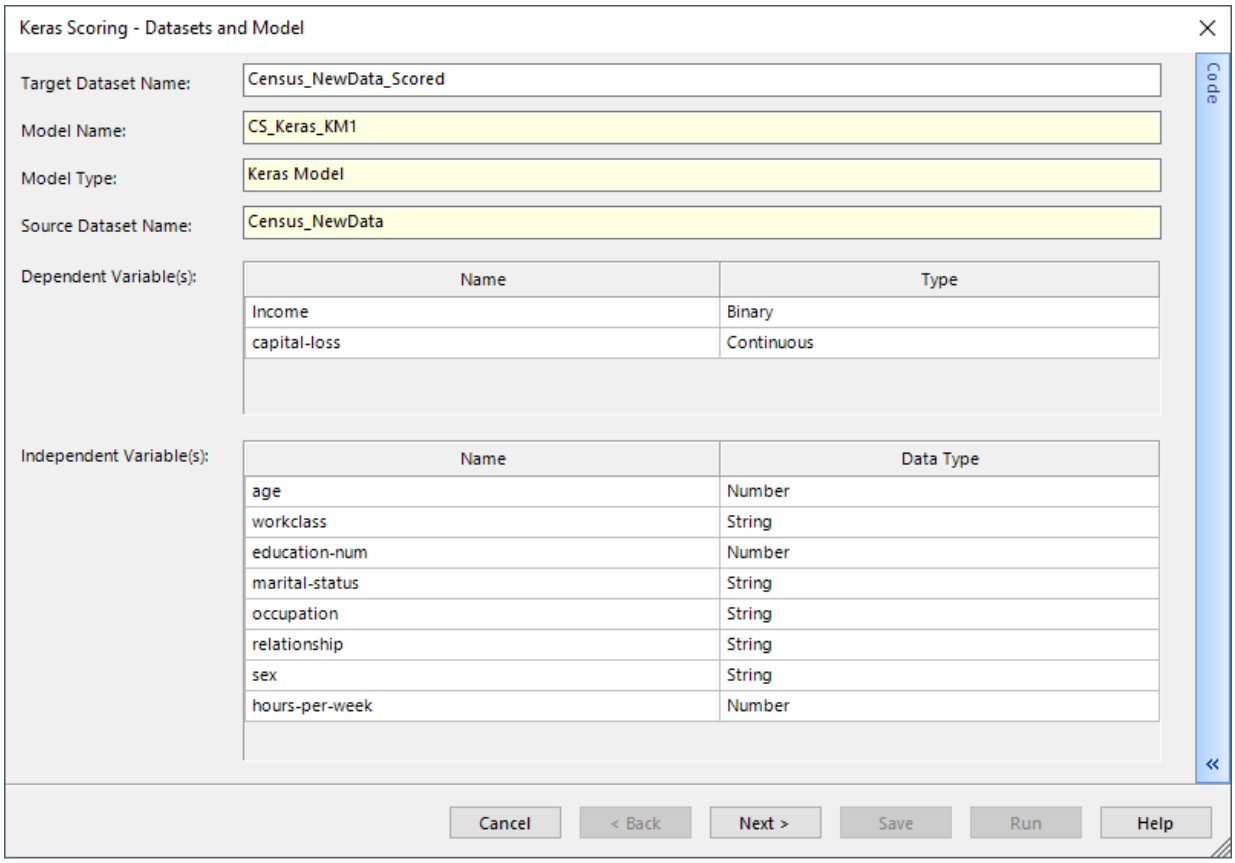


Double-click the **Scoring** node or right-click on it and select **Modify**.

The Scoring wizard opens. It displays the names of the input model and dataset, the list of dependent and independent variables and their data types.

In the *Target Dataset Name* field, specify the name for the output dataset that will contain the scores.

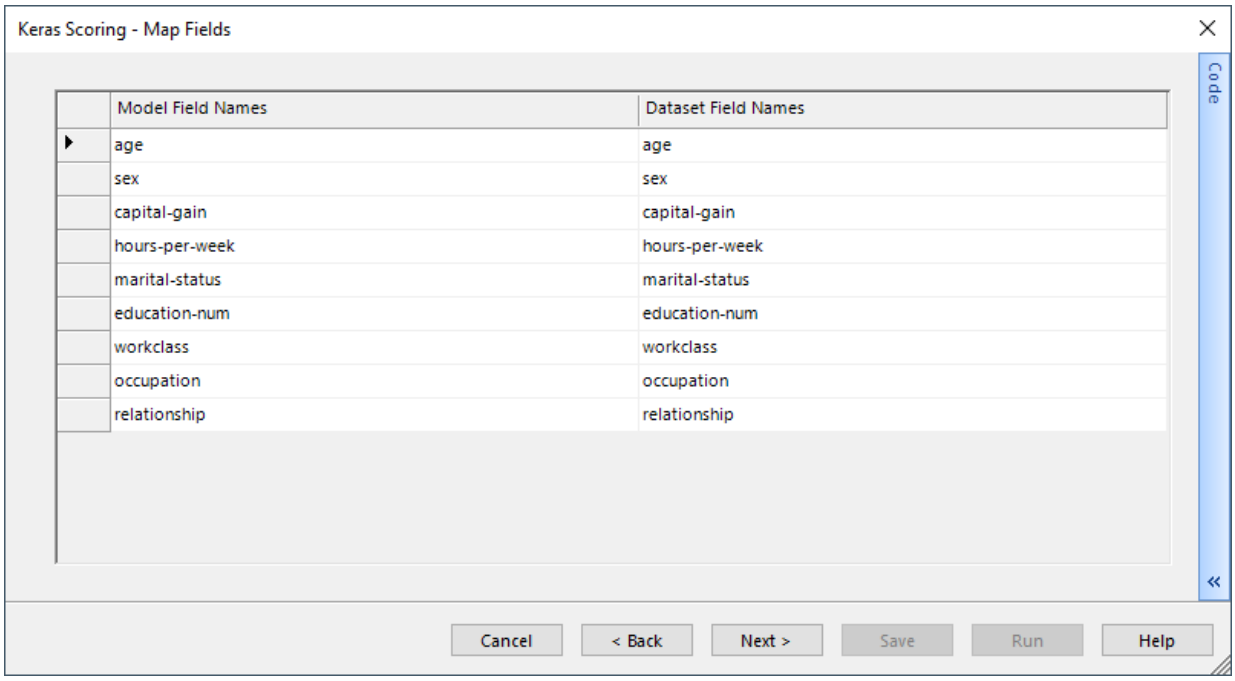
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

The next step provides the option of mapping the field names in the model to the ones in the scoring input dataset. In some situations, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map fields by clicking in the right column (Dataset Field Names) and selecting the right fields to match the model variables in the left column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

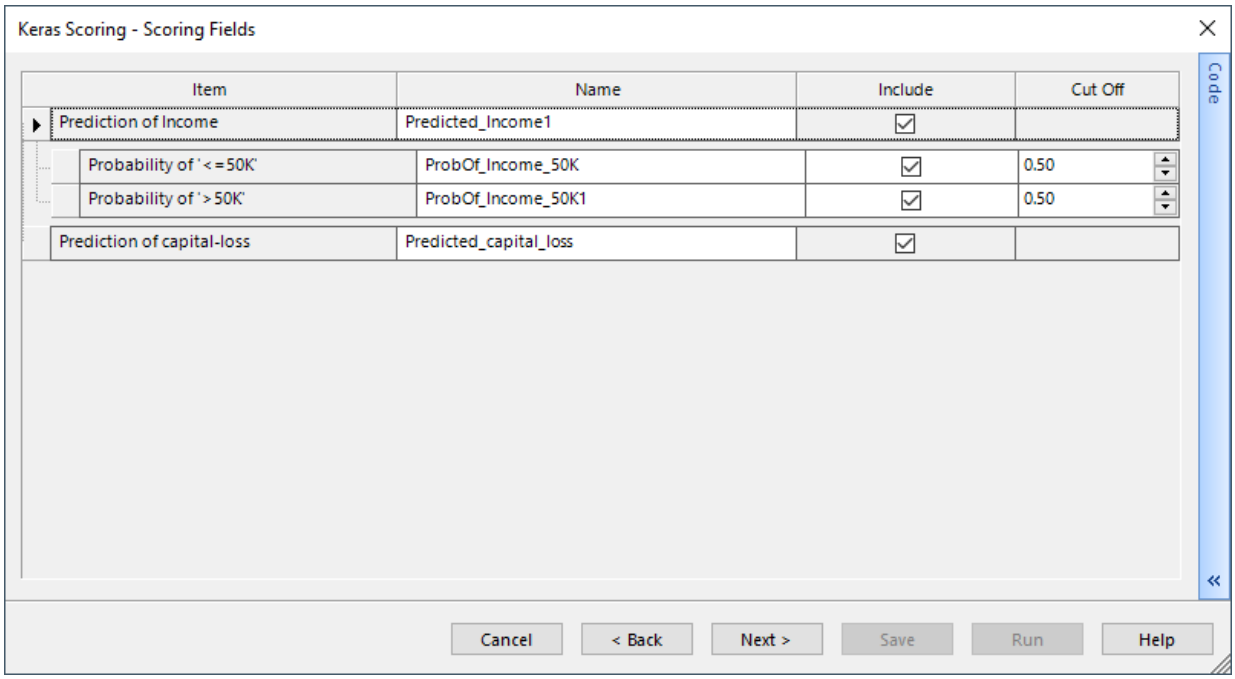


Note: When scoring Keras Deep Learning models that use derived variables (also called calculated columns) with expressions, the input dataset must include these variables. This is different from the case of native Altair models with derived variables, which can be scored on a dataset with only *original* variables that the derived ones depend on, while the derived ones are not required.

Click **Next**.

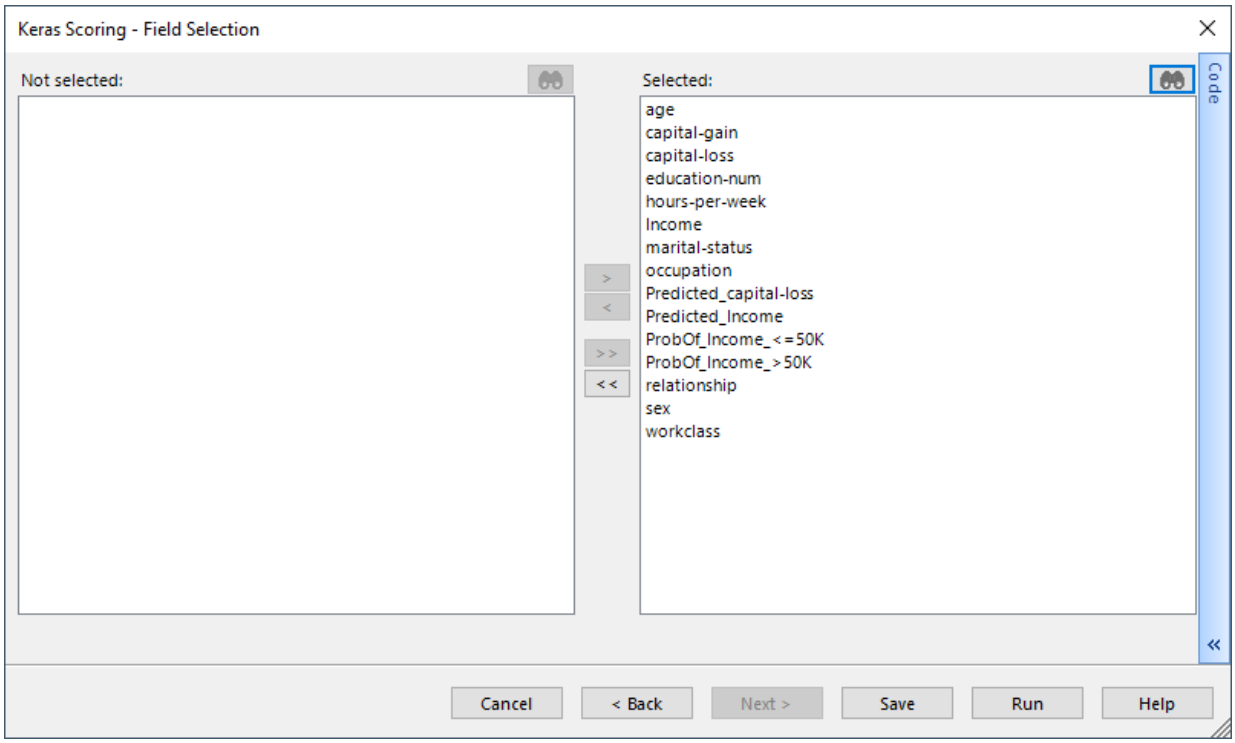
In the **Scoring Fields** page, select the scoring fields to be included in the scoring output dataset and modify their names if necessary. For discrete dependent variables, specify the probability cut-off values. See the general Scoring Wizard topic for more information on cut-off values.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



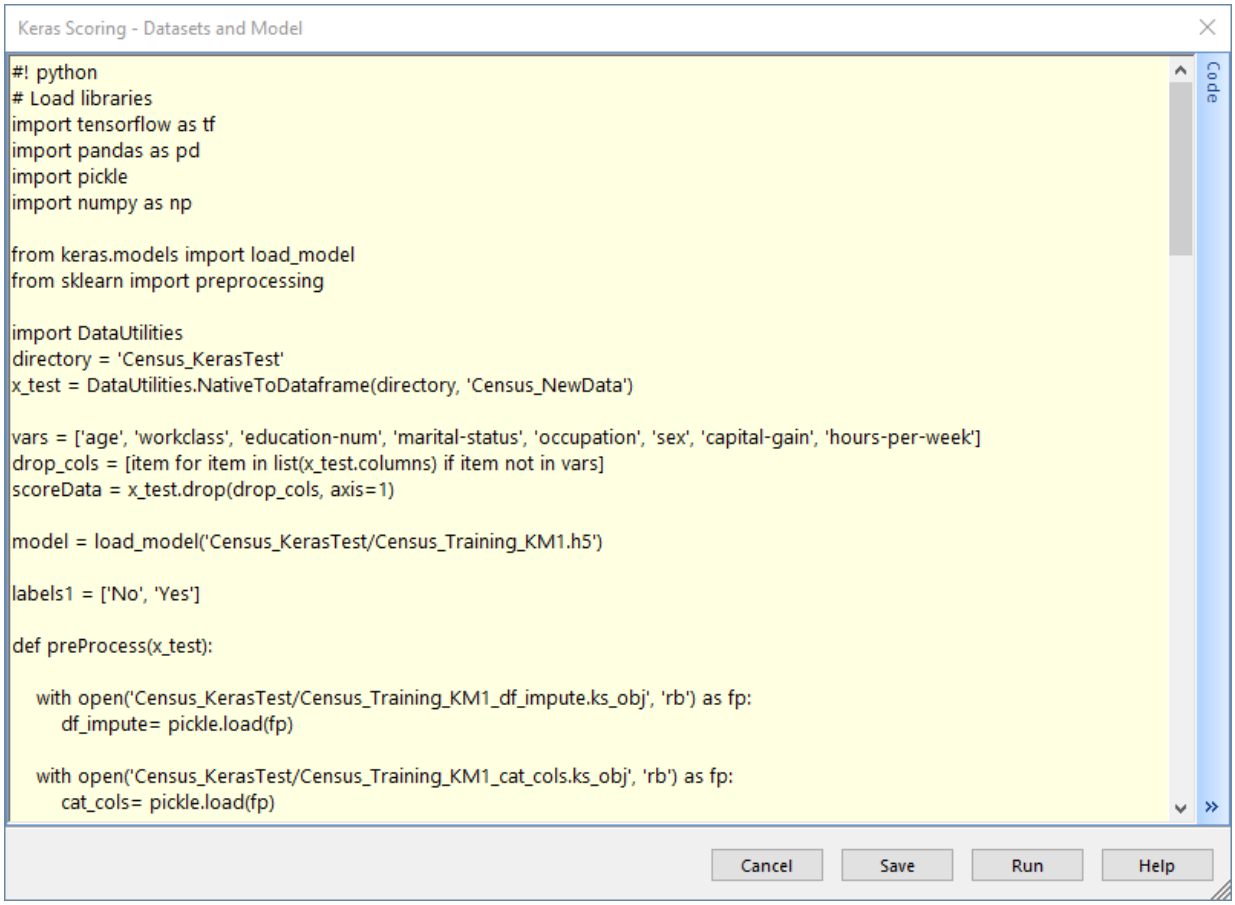
Click **Next**.

In the **Field Selection** page, choose the fields from the input dataset that you would like to include in the scoring output in addition to the new score / probability fields. By default, all fields in the scoring input dataset are selected to be included in the output.



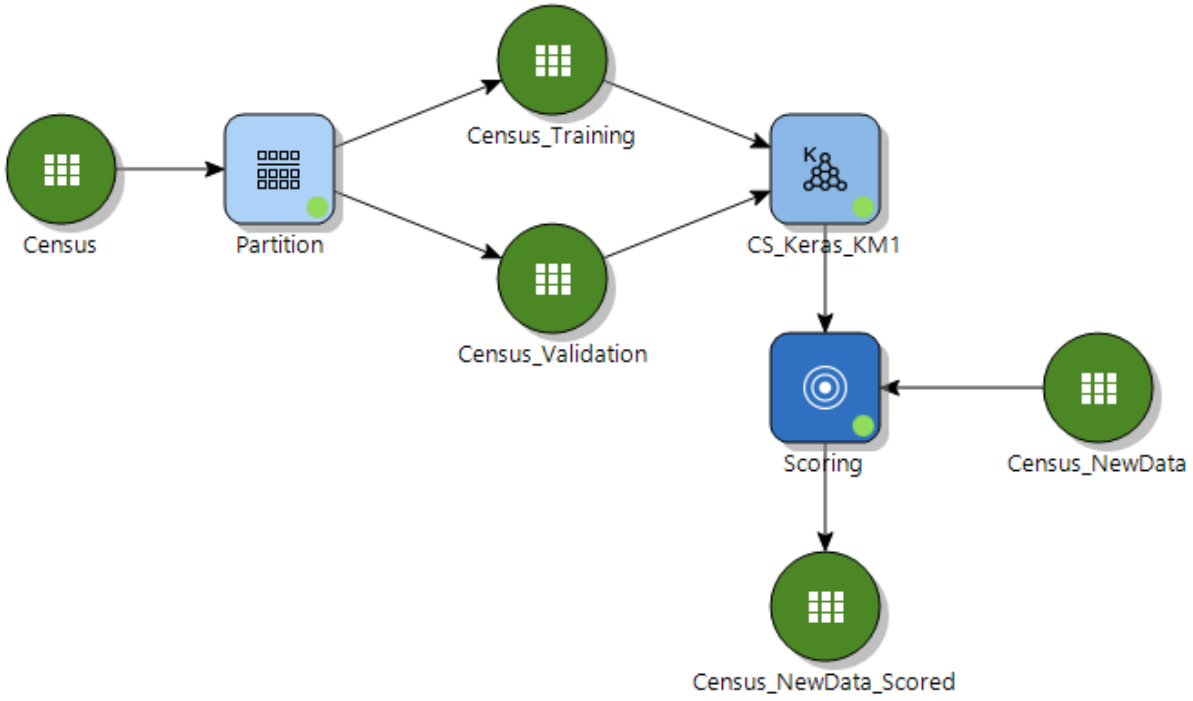
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.



Click **Save** to save the settings, or click **Run** to save the settings and run the scoring process immediately. The model is scored in a Python session with TensorFlow backend against the input data and the generated probabilities / scores are converted back from Pandas to Altair KDD format.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Double-click on the output dataset to view the scoring results.

- For discrete dependent variables, the output dataset contains the probabilities of the DV values and the prediction (the value that has the highest probability) for each record.
- For continuous dependent variables, the output dataset contains the predicted value for each record.

	Predicted_capital-loss	Predicted_Response	ProbOf_Response_No	ProbOf_Response_Yes
25	20.051362991333	No	0.818814516067505	0.181185454130173
26	113.580047607422	No	0.529841303825378	0.470158696174622
27	104.981788635254	No	0.542384803295135	0.457615196704865
28	223.476135253906	Yes	0.353432238101959	0.646567761898041
29	89.8815460205078	No	0.66495019197464	0.33504980802536
30	134.846542358398	Yes	0.478796482086182	0.521203517913818
31	165.005187988281	Yes	0.457728087902069	0.542271912097931
32	63.8199729919434	No	0.66535758972168	0.334642440080643
33	41.4115982055664	No	0.784708797931671	0.215291202068329
34	61.854076385498	No	0.626626670360565	0.373373329639435
35	121.861389160156	No	0.664023637771606	0.335976392030716

See also: Keras Deep Learning

Keras Deep Learning Validation

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

A Keras Deep Learning model is validated using the **Model Validation** node in the **Evaluate** palette. For general information about validation, see the topic About Validation.

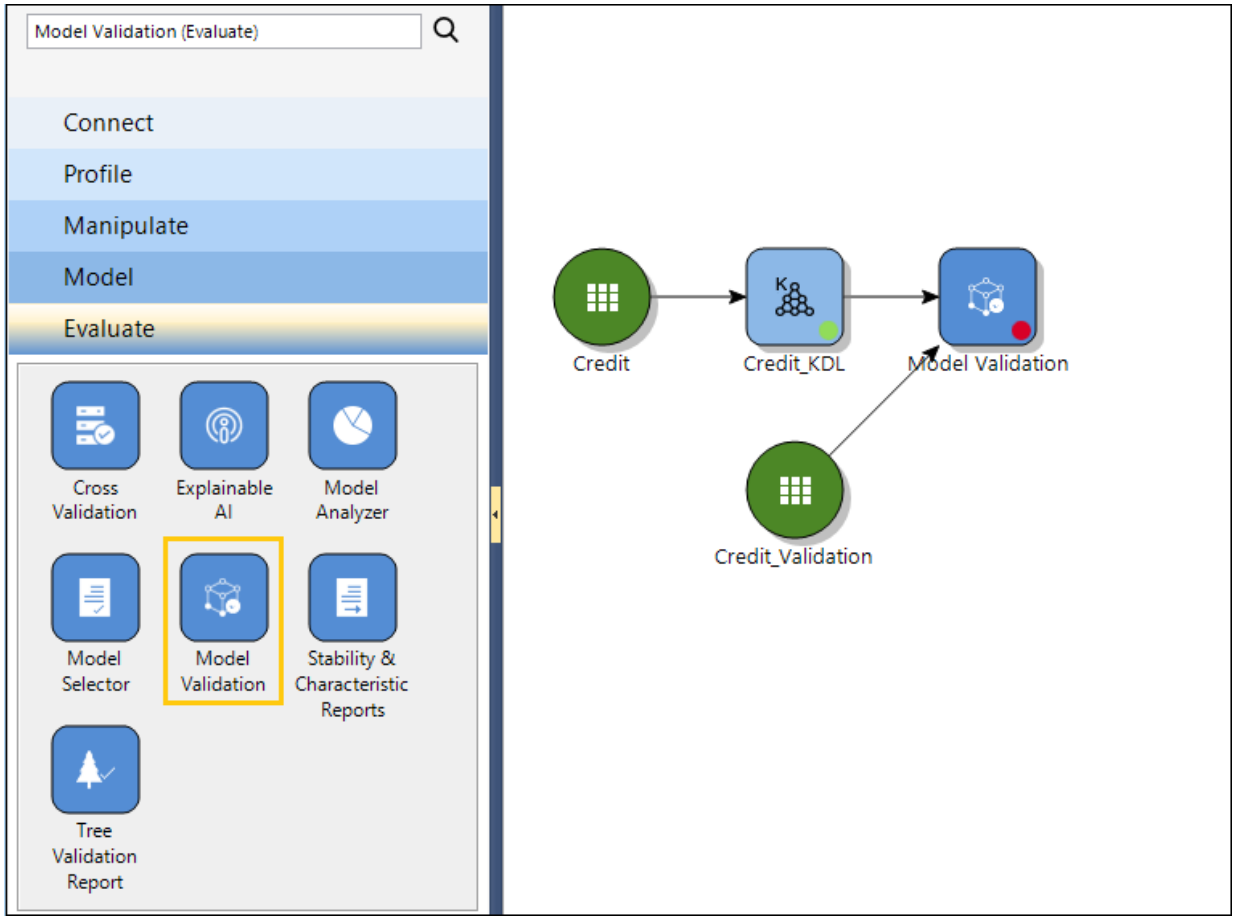
Prerequisites for Keras Deep Learning model validation: a **Keras Deep Learning** node and an input dataset to validate.

To add a Model Validation operation to your workflow:

- Drag the **Model Validation** node from the **Evaluate** palette onto the Workflow canvas and connect the inputs: a **Keras Deep Learning** node and a dataset node that contains the data to validate.

OR

- Right-click anywhere on the workflow canvas and select the context menu command **Evaluate --> Model Validation**. This will create a new Model Validation node. Connect the inputs: a **Keras Deep Learning** model node and a dataset node that contains the data to validate.

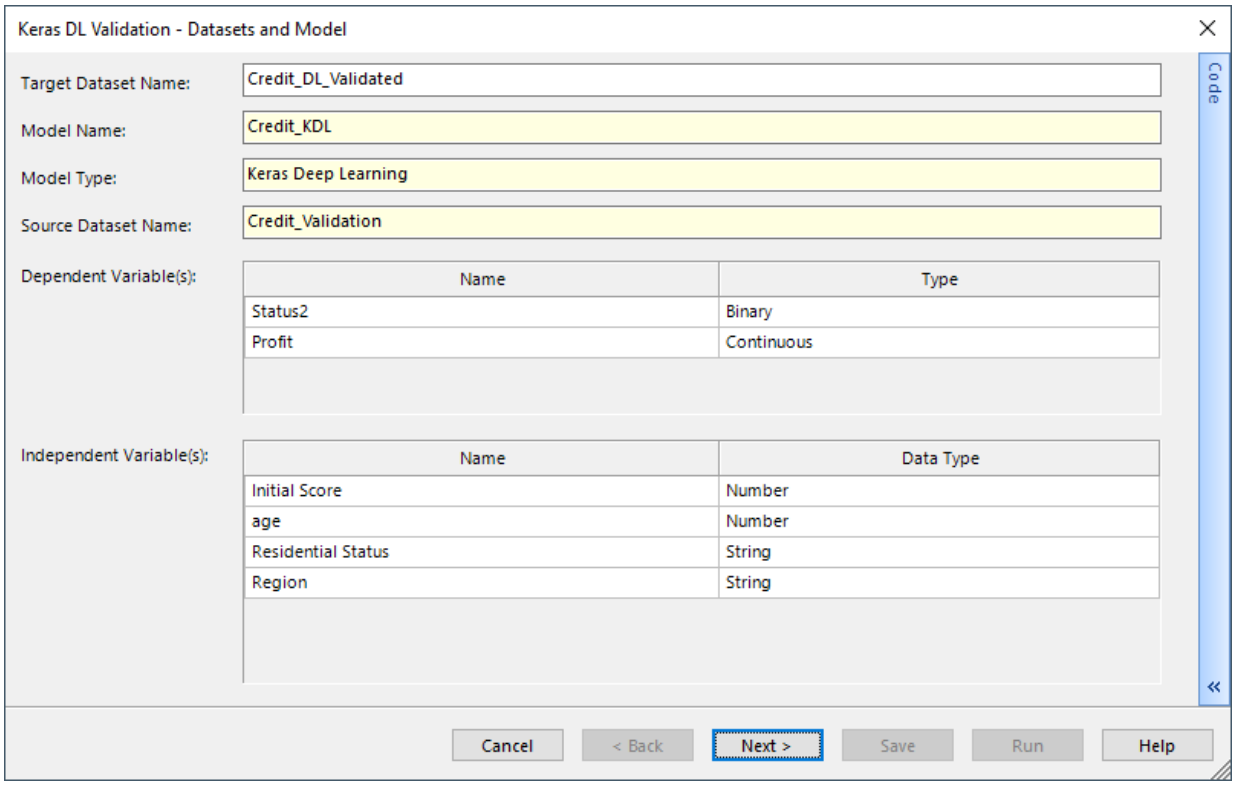


Double-click the **Model Validation** node or right-click on it and select **Modify**.

The Keras Deep Learning Validation wizard opens. It displays the names of the input model and dataset, the names and types of the dependent and independent variables of the model.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

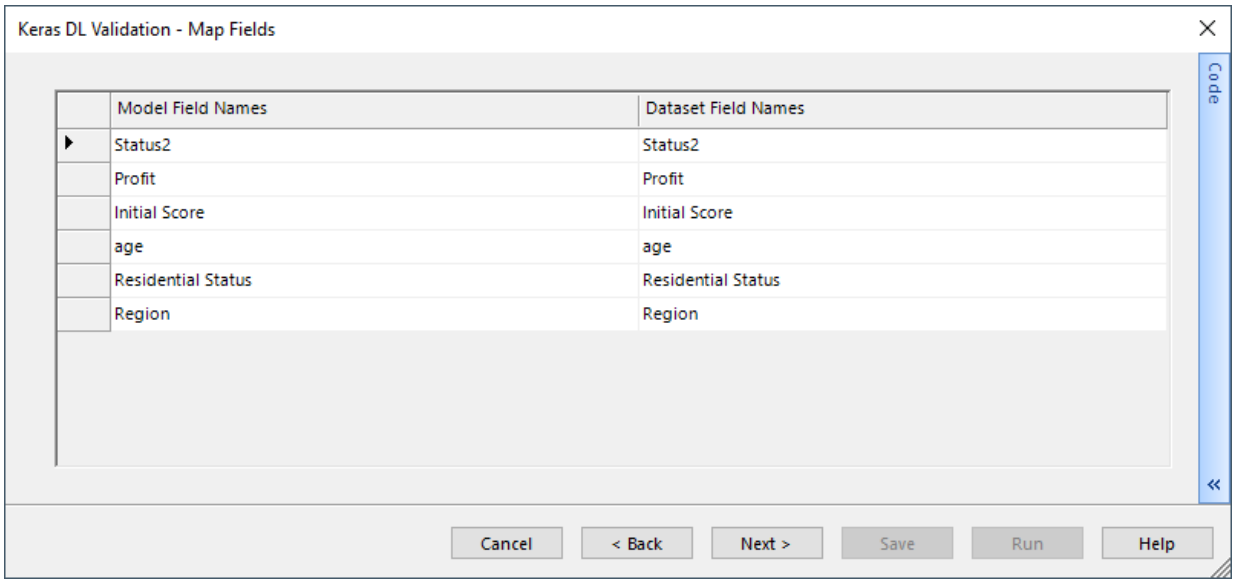
In the *Target Dataset Name* field, specify the name for the output dataset that will contain the validation results.



Click **Next**.

The next step provides the option of mapping the field names in the model to the ones in the validation input dataset. In some cases, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map some fields by clicking in the Dataset Field Names column and selecting the right fields to match the model variables in the left column:

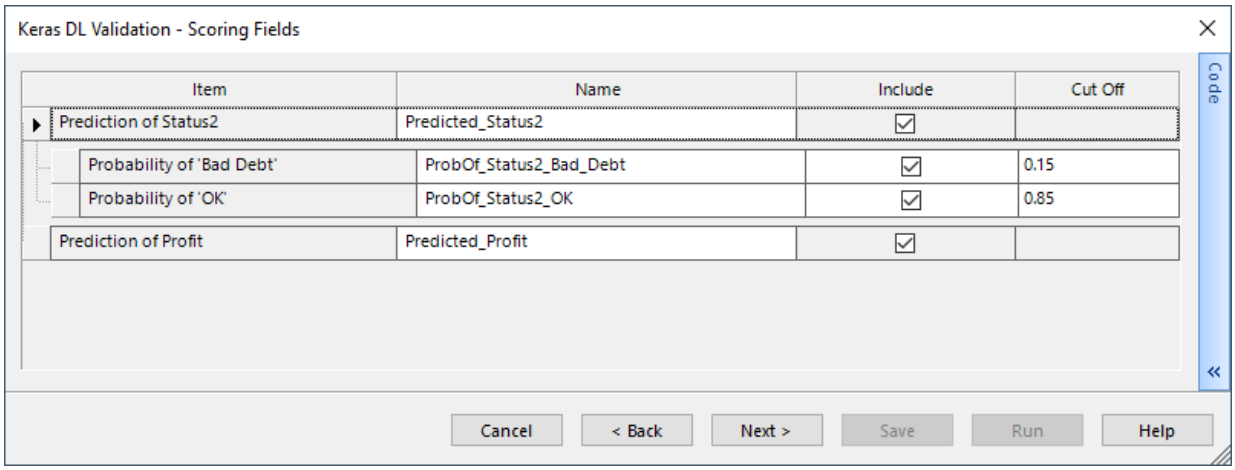
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Note: When validating Keras Deep Learning models that use derived variables (also called calculated columns) with expressions, the input dataset must include these variables. This is different from the case of native Altair models with derived variables, which can be validated on a dataset with only *original* variables that the derived ones depend on, while the derived ones are not required.

Click **Next**.

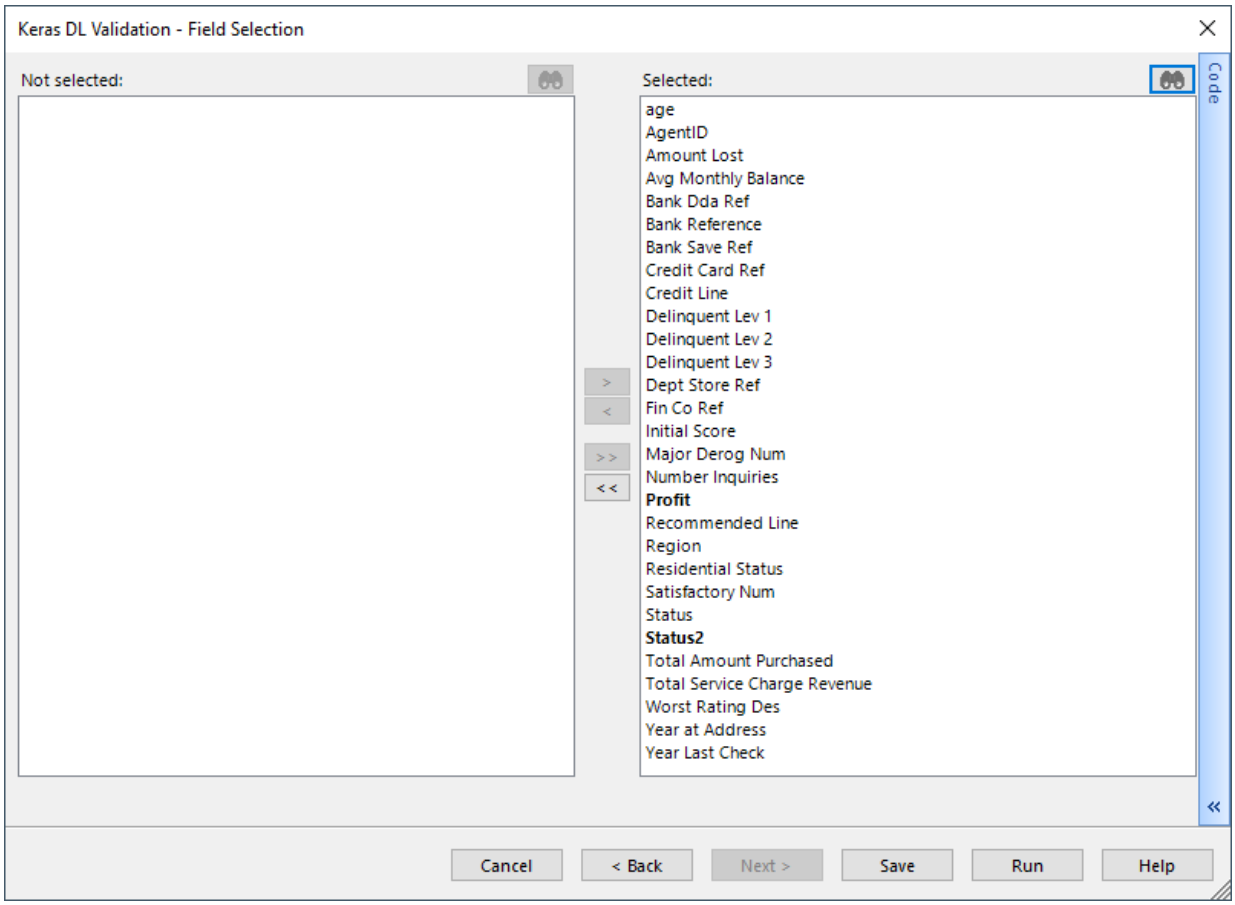
In the **Scoring Fields** page, select the scoring fields to be included in the scoring output dataset and modify their names if necessary. For discrete dependent variables, specify the probability cut-off values. See the general Validation Wizard topic for more information on cut-off values.



Click **Next**.

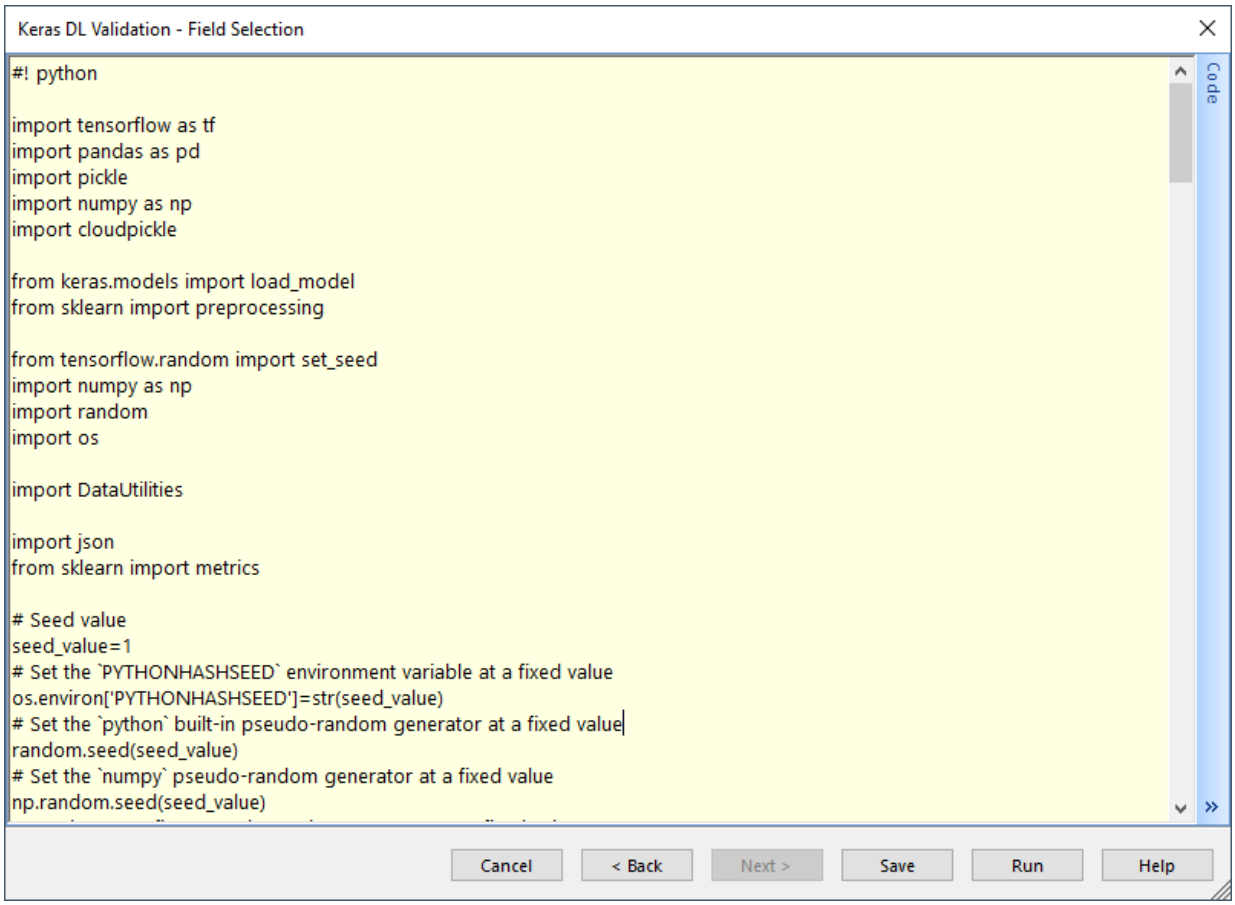
In the **Field Selection** page, choose the fields from the input dataset that you would like to include in the validation output in addition to the new score / probability fields. The field(s) representing the dependent variable(s) are highlighted in bold font and cannot be excluded. By default, all fields in the validation input dataset are selected to be included in the output.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

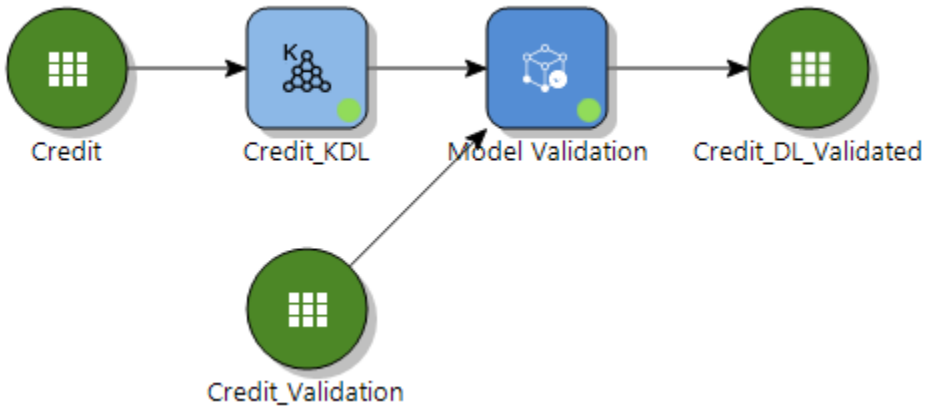


To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. You can also use the right-click context menu to copy the code, save it in a file, or search for specific code fragments. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the validation process immediately.

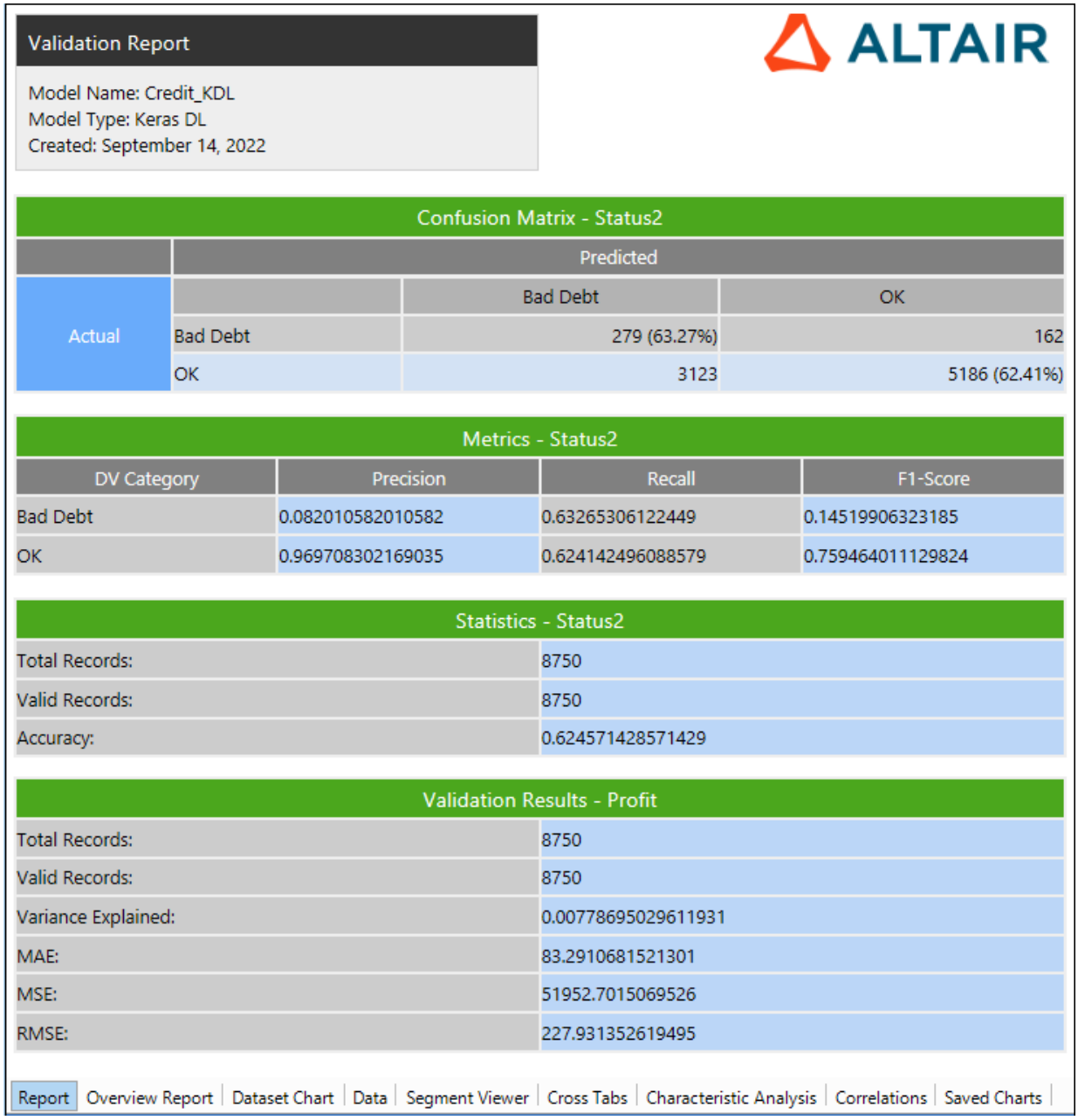


Double-click on the output dataset to view the validation results. The first tab of the output dataset is the Validation Report. It displays the values of the model evaluation metrics.

- For discrete dependent variables, along with the model accuracy, the Report tab shows the Confusion Martix, the table showing *Precision*, *Recall*, and *F1-Score.*

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- For continuous dependent variables, the Report tab shows Variance Explained (R-Squared), Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE).



Validation Report

Model Name: Credit_KDL
Model Type: Keras DL
Created: September 14, 2022

ALTAIR

Confusion Matrix - Status2			
	Predicted		
		Bad Debt	OK
Actual	Bad Debt	279 (63.27%)	162
	OK	3123	5186 (62.41%)

Metrics - Status2			
DV Category	Precision	Recall	F1-Score
Bad Debt	0.082010582010582	0.63265306122449	0.14519906323185
OK	0.969708302169035	0.624142496088579	0.759464011129824

Statistics - Status2	
Total Records:	8750
Valid Records:	8750
Accuracy:	0.624571428571429

Validation Results - Profit	
Total Records:	8750
Valid Records:	8750
Variance Explained:	0.00778695029611931
MAE:	83.2910681521301
MSE:	51952.7015069526
RMSE:	227.931352619495

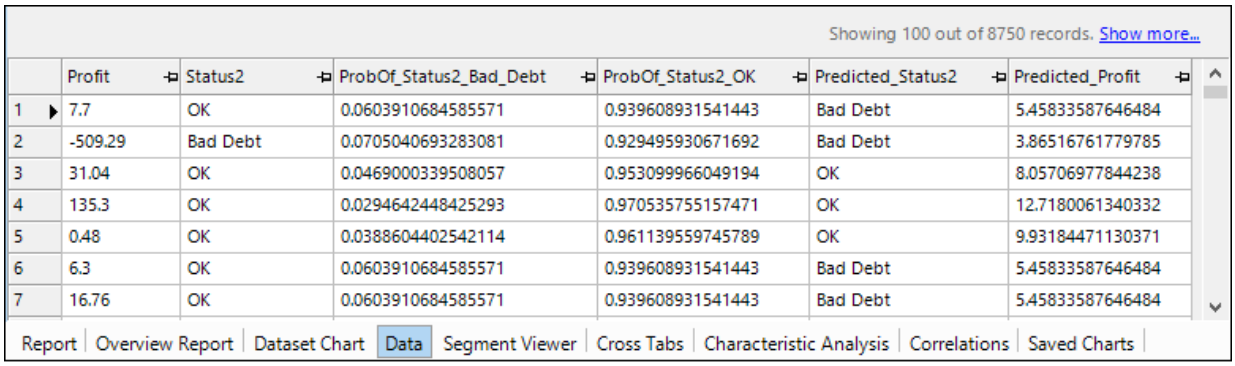
Report | Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

Click the **Data** tab of the output dataset and scroll to the rightmost columns to view the predictions and probabilities for the dependent variables.

- For discrete dependent variables, the output dataset contains the probabilities of the dependent variable values and the prediction (the value that has the highest probability) for each record.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- For continuous dependent variables, the output dataset contains the predicted value for each record.



Showing 100 out of 8750 records. Show more...

	Profit	Status2	ProbOf_Status2_Bad_Debt	ProbOf_Status2_OK	Predicted_Status2	Predicted_Profit
1	7.7	OK	0.0603910684585571	0.939608931541443	Bad Debt	5.45833587646484
2	-509.29	Bad Debt	0.0705040693283081	0.929495930671692	Bad Debt	3.86516761779785
3	31.04	OK	0.0469000339508057	0.953099966049194	OK	8.05706977844238
4	135.3	OK	0.0294642448425293	0.970535755157471	OK	12.7180061340332
5	0.48	OK	0.0388604402542114	0.961139559745789	OK	9.93184471130371
6	6.3	OK	0.0603910684585571	0.939608931541443	Bad Debt	5.45833587646484
7	16.76	OK	0.0603910684585571	0.939608931541443	Bad Debt	5.45833587646484

Report | Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

See also:

Keras Deep Learning

Keras Deep Learning Scoring

Auto Feature Engineering

Overview

The **Auto Feature Engineering** node in the **Manipulate** palette automatically creates new features (variables) using transformations and aggregations of different types from two or more input datasets and makes them into a new dataset.

This node is integrated with Python and uses **`Featuretools`** – a popular Python library for feature engineering which creates features from temporal and relational datasets to prepare data for modeling. For details, see the **`featuretools`** documentation at https://www.featuretools.com. The node wizard provides a convenient graphical interface to specify the options and parameters for `featuretools`. The wizard automatically translates the inputs into Python code, which can be previewed before running the node. The input datasets are converted from the Altair KDD format to pandas dataframes in the process. The resulting dataframe is converted back to produce the output Altair dataset.

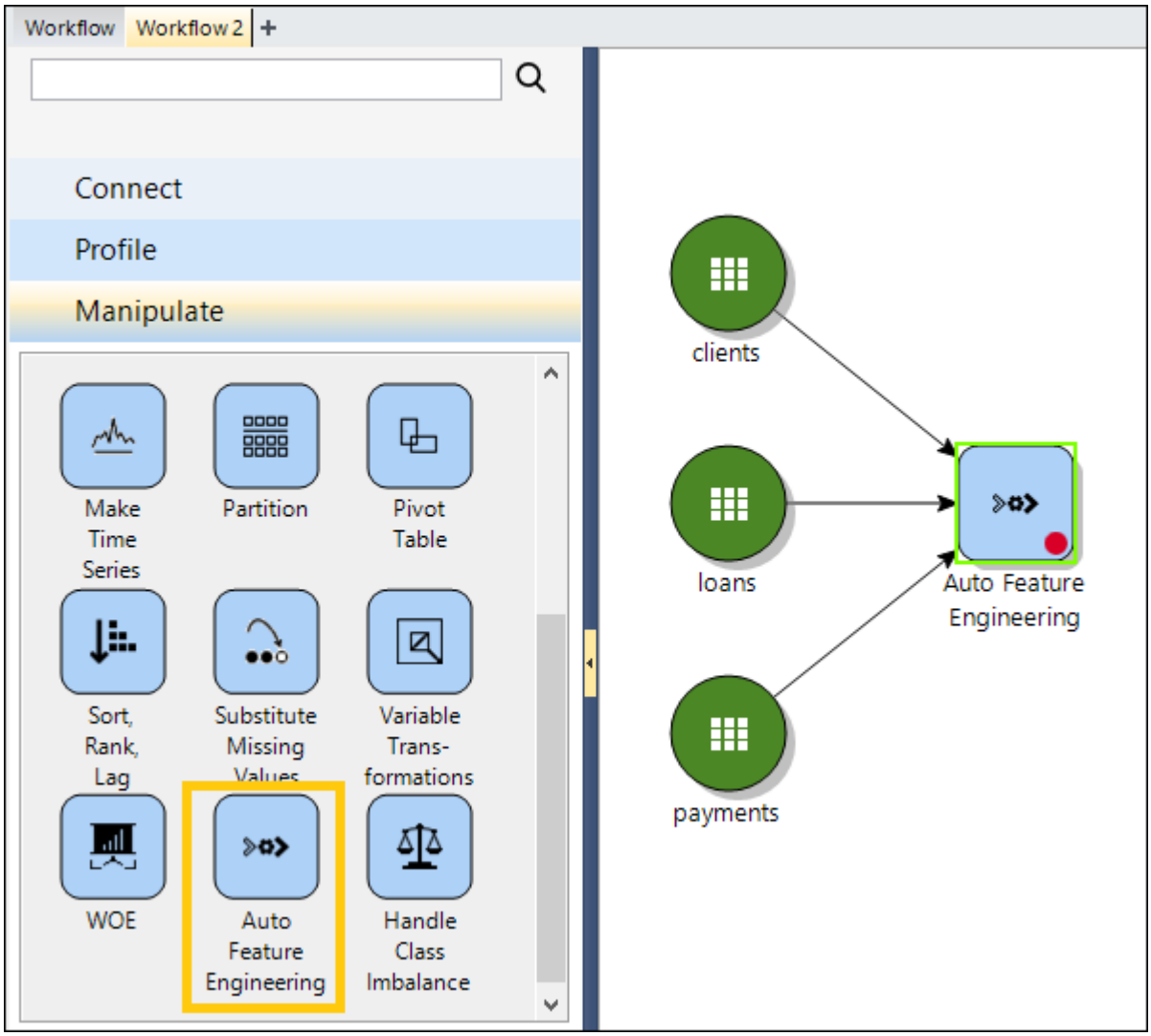
Data Manipulation with Auto Feature Engineering

Prerequisites: The Auto Feature Engineering requires at least two datasets as inputs. All input datasets must have ID fields and time index fields and represent tables linked in parent-child relationships.

To add an Auto Feature Engineering node to your workflow:

Drag the **Auto Feature Engineering** node from the **Manipulate** palette onto the workflow canvas. Alternatively, right-click anywhere on the canvas background and select **Manipulate --> Auto Feature Engineering** from the context menu. Connect the input datasets to the Auto Feature Engineering node. An example with three input datasets is shown below.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

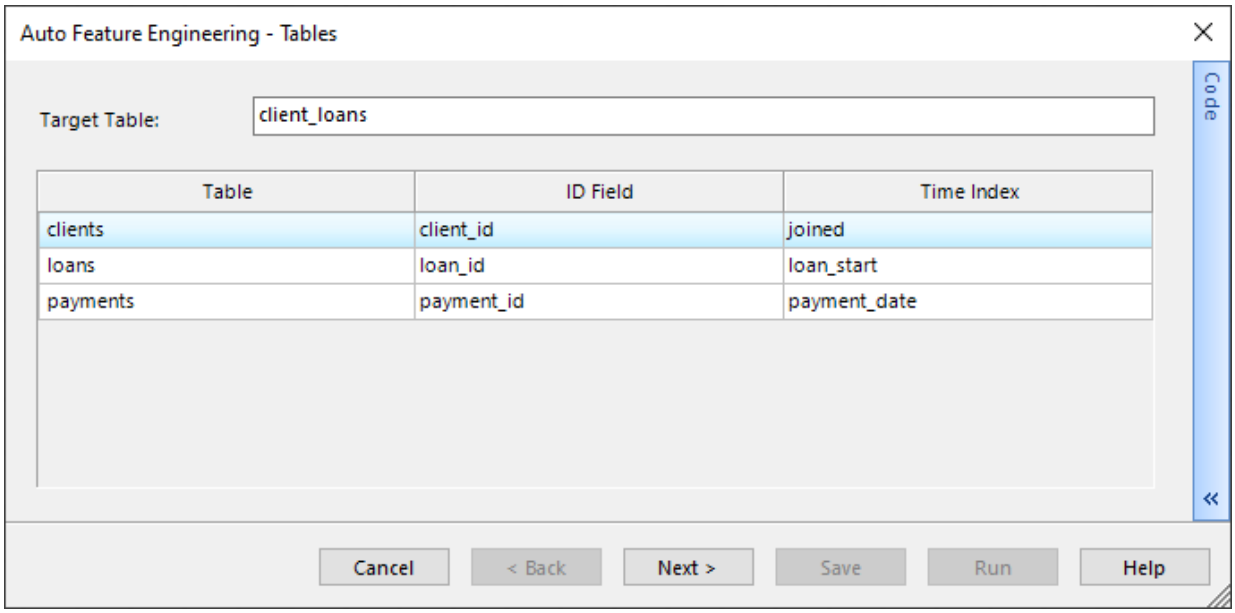


Double-click the Auto Feature Engineering node or right-click on it and select **Modify** to open the wizard.

In the first page of the wizard:

- Specify the name for the output dataset in the *Target Table* field.
- Select the ID field for each input dataset in the *ID Field* column.
- Select the *Time Index* field for each input dataset. This must be a Timestamp, Number or Integer field with no missing values.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

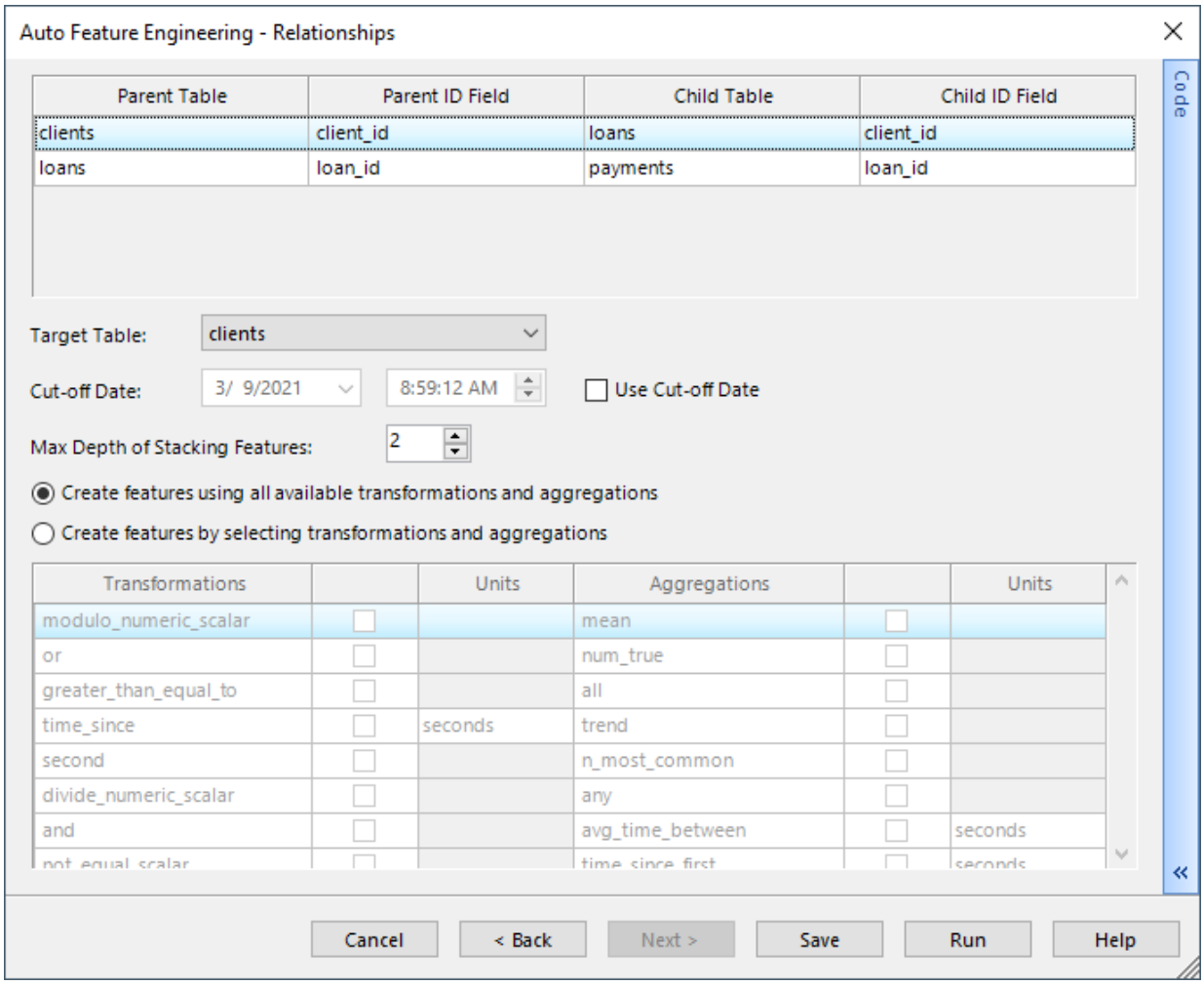


Click **Next**.

In the Relationships page:

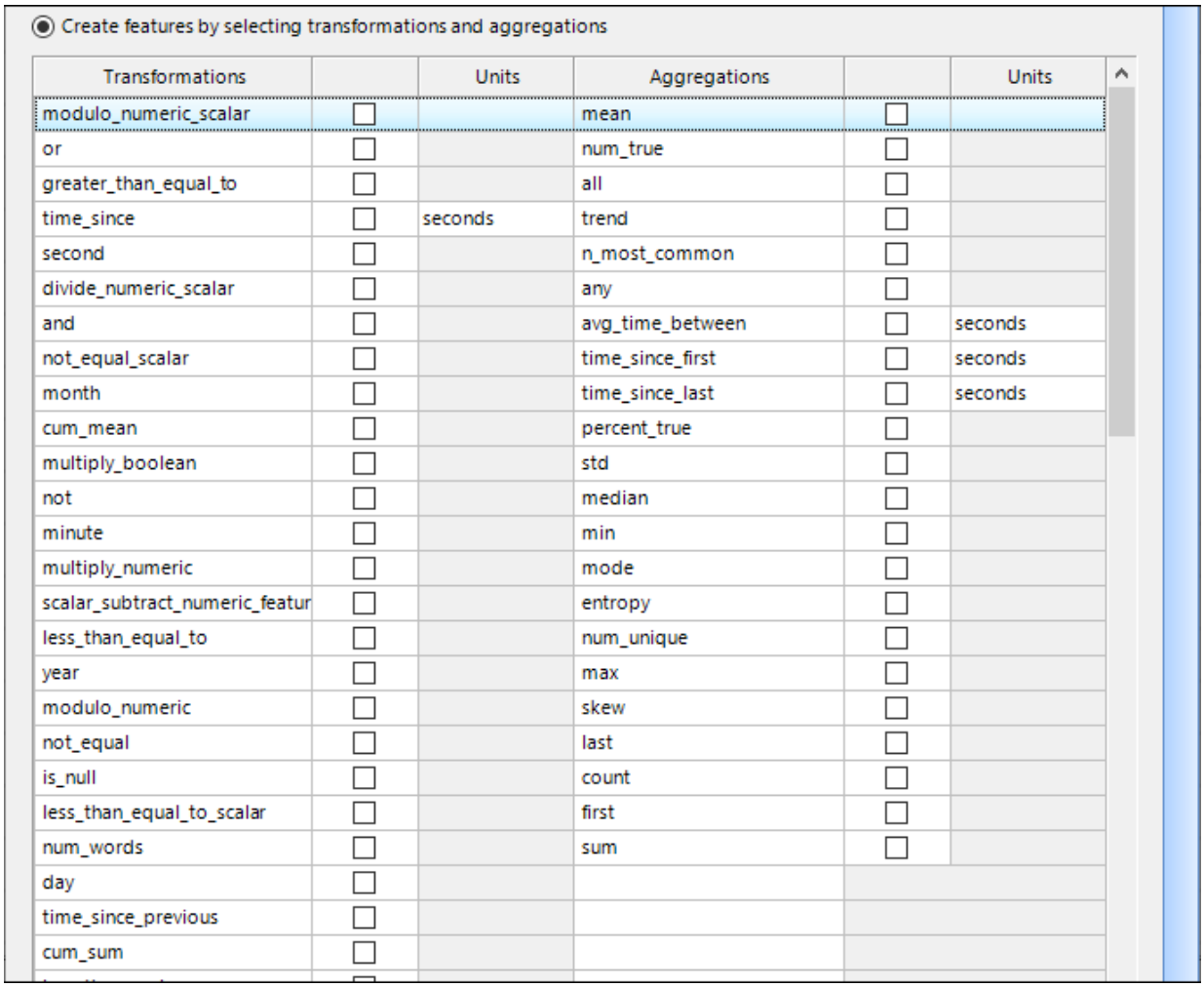
- Define the parent-child relationships for the tables representing the input datasets: For each Parent Table, select its Child Table. For the *Child ID Field,* specify the ID in the child table that links it to the parent table. In the example below, the clients table is a parent of the loans table, which is a parent of the payments table. The client_id in the loans table links it to the parent clients table, while the loan_id in the payments table links it to the parent loans table. In the general case, the link field names may be different in the parent and child tables.
- For *Target Table*, select the dataset that the output will be based on (usually the one that represents the base table in the hierarchy).
- Specify the cut-off date if necessary. Select the "*Use Cut-off Date*" box first, then enter the date and time.
- Specify the maximum depth of stacking features. The default is 2. Up to 3 levels are allowed.
- Available transformations and aggregations are listed in the table at the bottom of the wizard page. Choose whether to automatically create all of them or select the desired transformations and aggregations manually. The full list with descriptions is provided in the Appendix at the end of this Help topic.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



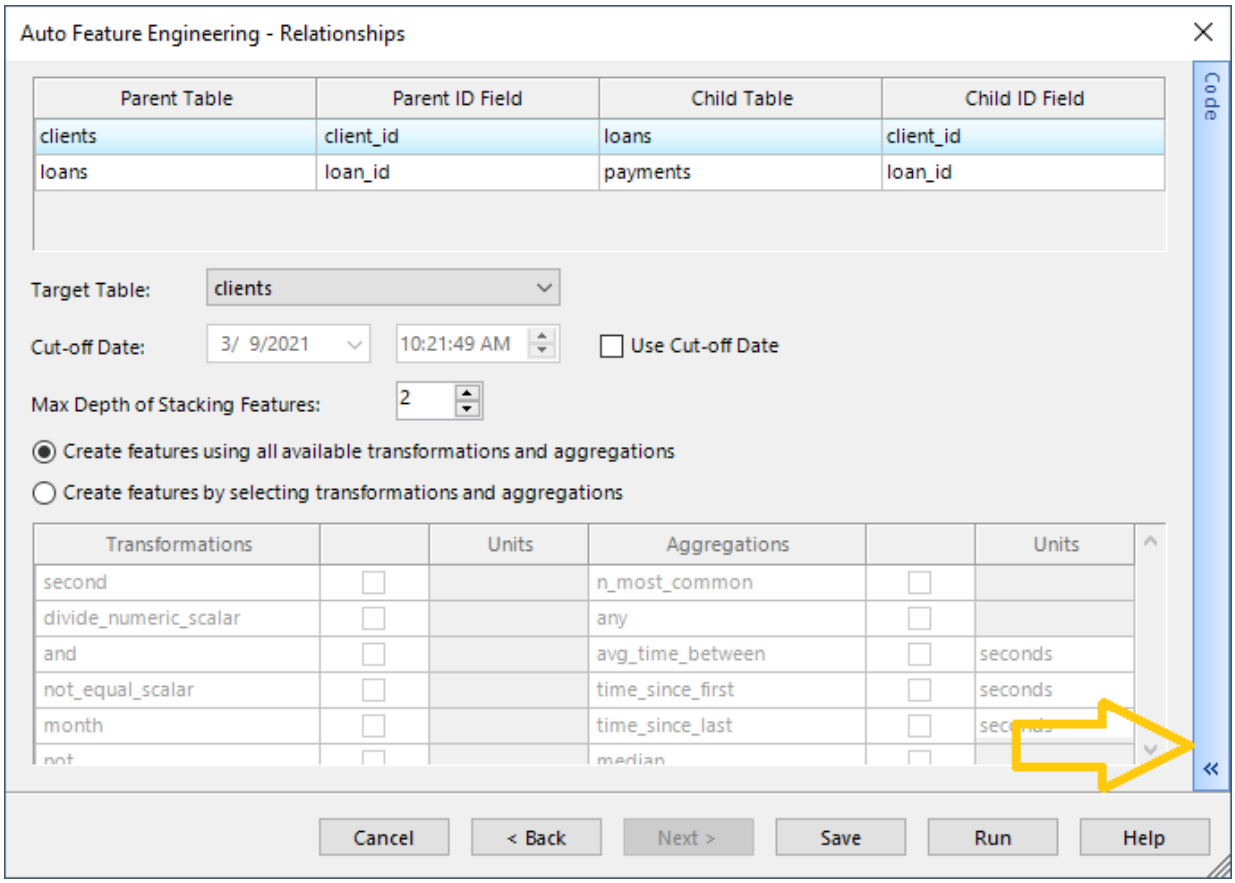
When selecting the desired transformations and aggregations manually, define the units for time and date function values.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



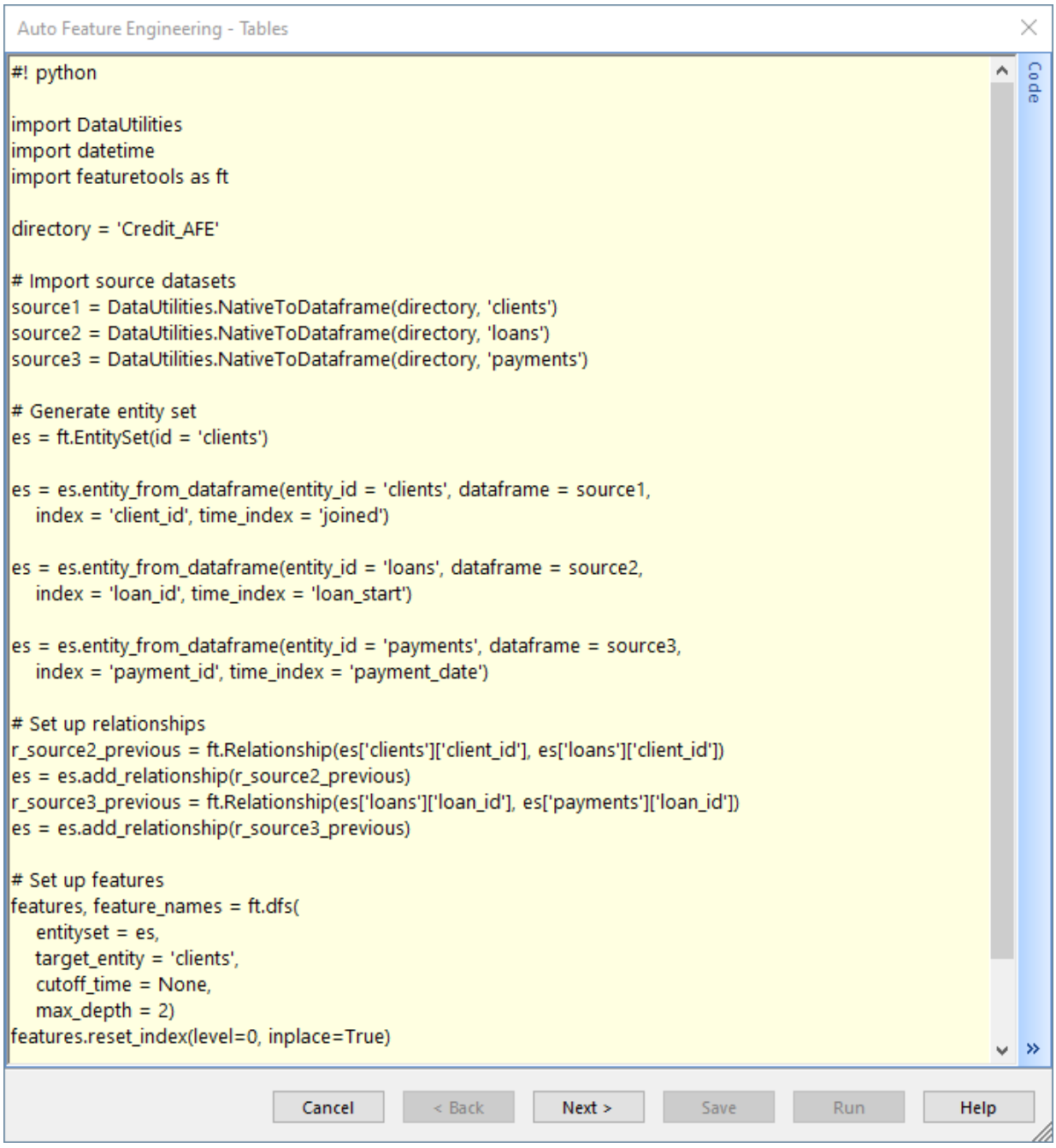
To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



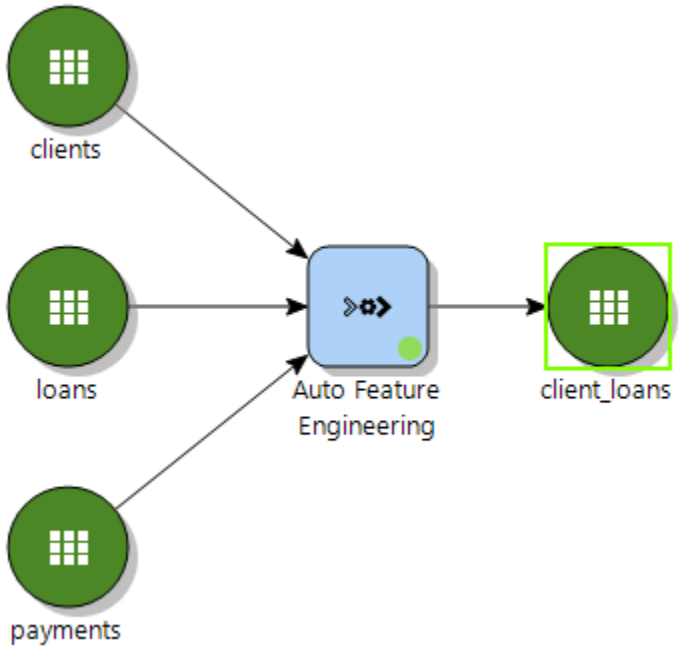
This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. The code generated from your inputs runs in Python. The input datasets are converted to Pandas dataframes, and the functions of the `featuretools` library are applied. The output data is converted from Pandas to the Altair KDD format and is shown as the output dataset node at the end.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Double-click the output dataset node to view the results and explore the generated features.

	client_id	income	credit_score	COUNT(loans)	MAX(loans.loan_amount)	MAX(loans.rate)	MAX(loans.repaid)	MEAN(loans.loan_amount)	MEAN(loans.rate)
1	42320	229481	563	15	13887	6.74	1	7062.06666666667	2.45733333333333
2	39384	191204	617	19	14654	9.23	1	7865.47368421053	3.53842105263158
3	26945	214516	806	15	14593	5.65	1	7125.93333333333	2.85533333333333
4	41472	152214	638	16	13657	9.82	1	7510.8125	3.98125
5	46180	43851	562	20	14081	9.26	1	7700.85	3.5025
6	46109	172677	527	20	14049	9.48	1	8951.6	3.1525
7	32885	58955	642	15	14162	9.11	1	9920.4	2.436
8	29841	38354	523	18	14837	6.76	1	9813	3.445
9	38537	127183	643	17	14804	8.01	1	8986.35294117647	2.38941176470588
10	35214	95849	696	18	14767	8.44	1	7173.55555555556	3.10833333333333
11	49068	128813	603	18	13910	6.78	1	7617.88888888889	3.09555555555556
12	26326	227920	633	16	13464	6.73	1	7270.0625	2.5175
13	26695	174532	680	18	14865	6.51	1	7824.72222222222	2.46611111111111
14	44601	156341	518	20	14104	9.22	1	6869.25	4.151
15	32726	235705	730	19	14802	9.1	1	6633.26315789474	3.05894736842105
16	25707	211422	621	20	13913	9.44	1	7963.95	3.477
17	49545	104564	770	20	14971	6.52	1	10289.3	2.684

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

Supported Transformations and Aggregations

#	Name	Type	Description
0	mean	aggregation	Computes the average for a list of values.
1	num_true	aggregation	Counts the number of `True` values.
2	all	aggregation	Calculates if all values are 'True' in a list.
3	trend	aggregation	Calculates the trend of a variable over time.
4	n_most_common	aggregation	Determines the `n` most common elements.
5	any	aggregation	Determines if any value is 'True' in a list.
6	time_since_first	aggregation	Calculates the time elapsed since the first datetime (in s

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

7	percent_true	aggregation	Determines the percent of `True` values.
8	std	aggregation	Computes the dispersion relative to the mean value, igno
9	time_since_last	aggregation	Calculates the time elapsed since the last datetime (defa
10	median	aggregation	Determines the middlemost number in a list of values.
11	min	aggregation	Calculates the smallest value, ignoring `NaN` values.
12	mode	aggregation	Determines the most commonly repeated value.
13	entropy	aggregation	Calculates the entropy for a categorical variable.
14	num_unique	aggregation	Determines the number of distinct values, ignoring `NaN
15	max	aggregation	Calculates the highest value, ignoring `NaN` values.
16	avg_time_between	aggregation	Computes the average number of seconds between con
17	skew	aggregation	Computes the extent to which a distribution differs from a
18	last	aggregation	Determines the last value in a list.
19	count	aggregation	Determines the total number of values, excluding `NaN`.
20	first	aggregation	Determines the first value in a list.
21	sum	aggregation	Calculates the total addition, ignoring `NaN`.
22	modulo_numeric_scalar	transform	Return the modulo of each element in the list by a scalar
23	or	transform	Element-wise logical OR of two lists.
24	greater_than_equal_to	transform	Determines if values in one list are greater than or equal
25	time_since	transform	Calculates time from a value to a specified cutoff datetim
26	second	transform	Determines the seconds value of a datetime.
27	divide_numeric_scalar	transform	Divide each element in the list by a scalar.
28	and	transform	Element-wise logical AND of two lists.
29	not_equal_scalar	transform	Determines if values in a list are not equal to a given sca
30	month	transform	Determines the month value of a datetime.
31	cum_mean	transform	Calculates the cumulative mean.
32	multiply_Boolean	transform	Element-wise multiplication of two lists of Boolean values
33	not	transform	Negates a Boolean value.
34	minute	transform	Determines the minutes value of a datetime.
35	multiply_numeric	transform	Element-wise multiplication of two lists.
36	scalar_subtract_numeric_feature	transform	Subtract each value in the list from a given scalar.
37	less_than_equal_to	transform	Determines if values in one list are less than or equal to
38	year	transform	Determines the year value of a datetime.
39	modulo_numeric	transform	Element-wise modulo of two lists.
40	not_equal	transform	Determines if values in one list are not equal to another l
41	is_null	transform	Determines if a value is null.
42	less_than_equal_to_scalar	transform	Determines if values are less than or equal to a given sc
43	num_words	transform	Determines the number of words in a string by counting t
44	day	transform	Determines the day of the month from a datetime.
45	time_since_previous	transform	Compute the time since the previous entry in a list.
46	cum_sum	transform	Calculates the cumulative sum.
47	less_than_scalar	transform	Determines if values are less than a given scalar.
48	diff	transform	Compute the difference between the value in a list and th
49	is_weekend	transform	Determines if a date falls on a weekend.
50	subtract_numeric_scalar	transform	Subtract a scalar from each element in the list.
51	subtract_numeric	transform	Element-wise subtraction of two lists.
52	greater_than_scalar	transform	Determines if values are greater than a given scalar.
53	longitude	transform	Returns the second tuple value in a list of LatLong tuples
54	negate	transform	Negates a numeric value.
55	week	transform	Determines the week of the year from a datetime.
56	cum_max	transform	Calculates the cumulative maximum.
57	divide_numeric	transform	Element-wise division of two lists.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

58	greater_than_equal_to_scalar	transform	Determines if values are greater than or equal to a given
59	hour	transform	Determines the hour value of a datetime.
60	cum_count	transform	Calculates the cumulative count.
61	multiply_numeric_scalar	transform	Multiply each element in the list by a scalar.
62	weekday	transform	Determines the day of the week from a datetime.
63	modulo_by_feature	transform	Return the modulo of a scalar by each element in the list
64	add_numeric	transform	Element-wise addition of two lists.
65	greater_than	transform	Determines if values in one list are greater than another
66	haversine	transform	Calculates the approximate haversine distance between
67	less_than	transform	Determines if values in one list are less than another list.
68	add_numeric_scalar	transform	Add a scalar to each value in the list.
69	absolute	transform	Computes the absolute value of a number.
70	equal_scalar	transform	Determines if values in a list are equal to a given scalar.
71	isin	transform	Determines whether a value is present in a provided list.
72	percentile	transform	Determines the percentile rank for each value in a list.
73	divide_by_feature	transform	Divide a scalar by each value in the list.
74	cum_min	transform	Calculates the cumulative minimum.
75	equal	transform	Determines if values in one list are equal to another list.
76	age	transform	Calculates the age in years as a floating-point number gi
77	latitude	transform	Returns the first tuple value in a list of LatLong tuples.
78	num_characters	transform	Calculates the number of characters in a string.

Generalized Linear Model

Overview

The **Sklearn GLM** node in the **Model** palette of Knowledge Studio allows you to build regression models with a dependent variable that does not necessarily follow the normal distribution. The node wizard is similar to that of the Linear Regression node, but it works through integration with Python libraries. It uses regressors of the linear model methods of **`scikit-learn`** - a popular Python library for machine learning. For details, see the scikit-learn documentation for Generalized Linear Regression.

The node wizard provides a convenient graphical interface to specify the options and parameters for linear model functions in the `scikit-learn (sklearn)` library. The wizard automatically translates the inputs into Python code, which can be previewed before running the node. The code then runs in a container with the necessary Python packages and their dependencies. The input dataset is converted from the Altair KDD format to pandas dataframes in the process. The resulting dataframe is converted back to produce the output Altair dataset.

After training, the model can be validated on test data using the Sklearn GLM Validation node or used for scoring new data with the Sklearn GLM Scoring node.

Building an Sklearn GLM Model

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

An **Sklearn GLM** model requires a single dataset as input. It can optionally take another predictive model as the second input (*base model*), in which case it inherits the dependent variable and independent variable selection from the base model.

An **Sklearn GLM** node can be added to the workflow in one of the following ways:

- Drag a connection arrow from the training dataset node to any area on the Workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Model --> Sklearn GLM** as in the example screenshot below. A new **Sklearn GLM** node will be created, and the wizard will open.

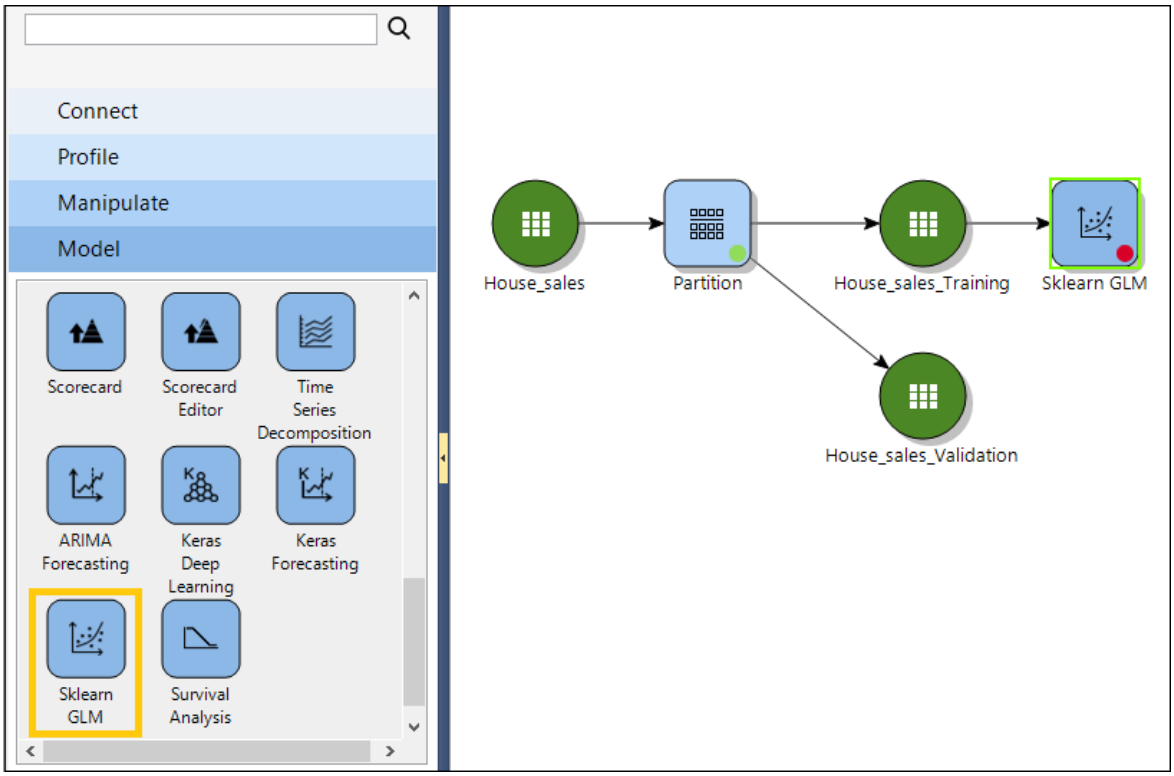
OR

- Right-click on the Workflow canvas and select **Model --> Sklearn GLM** from the context menu. Connect the training dataset. Optionally, connect another predictive model node as a base model to use as a template.

OR

- Open the **Model** palette and drag the **Sklearn GLM** node onto the workflow canvas. Connect the training dataset. Optionally, connect another predictive model node as a base model to use as a template.

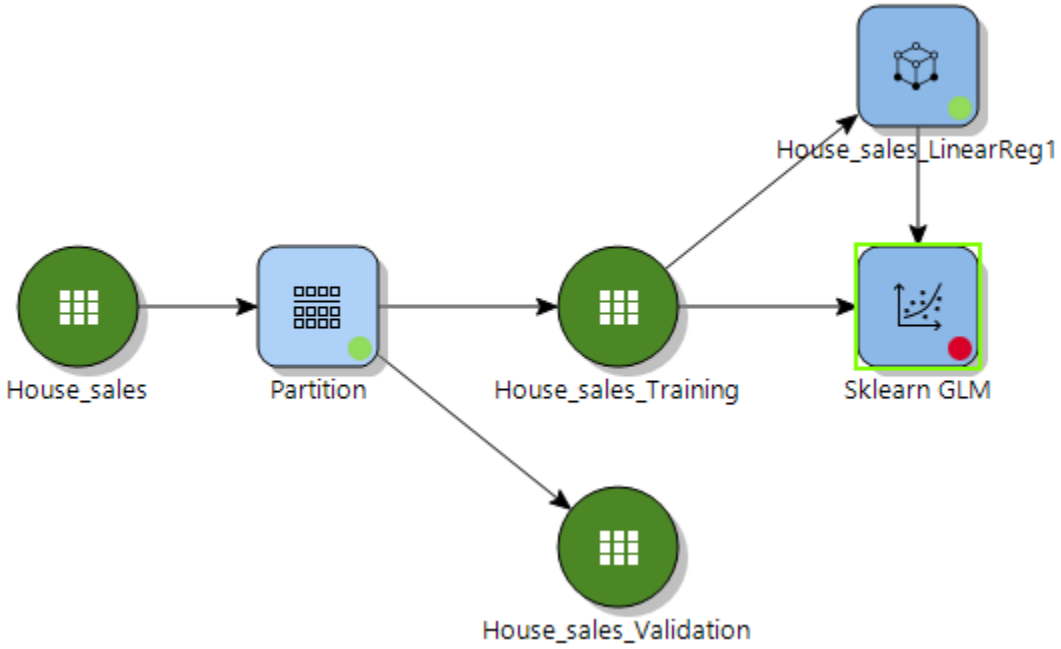
Example 1:



Example 2 (with base model)

In this example, an existing Linear Regression model is connected to the new Sklearn GLM model as a base model.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



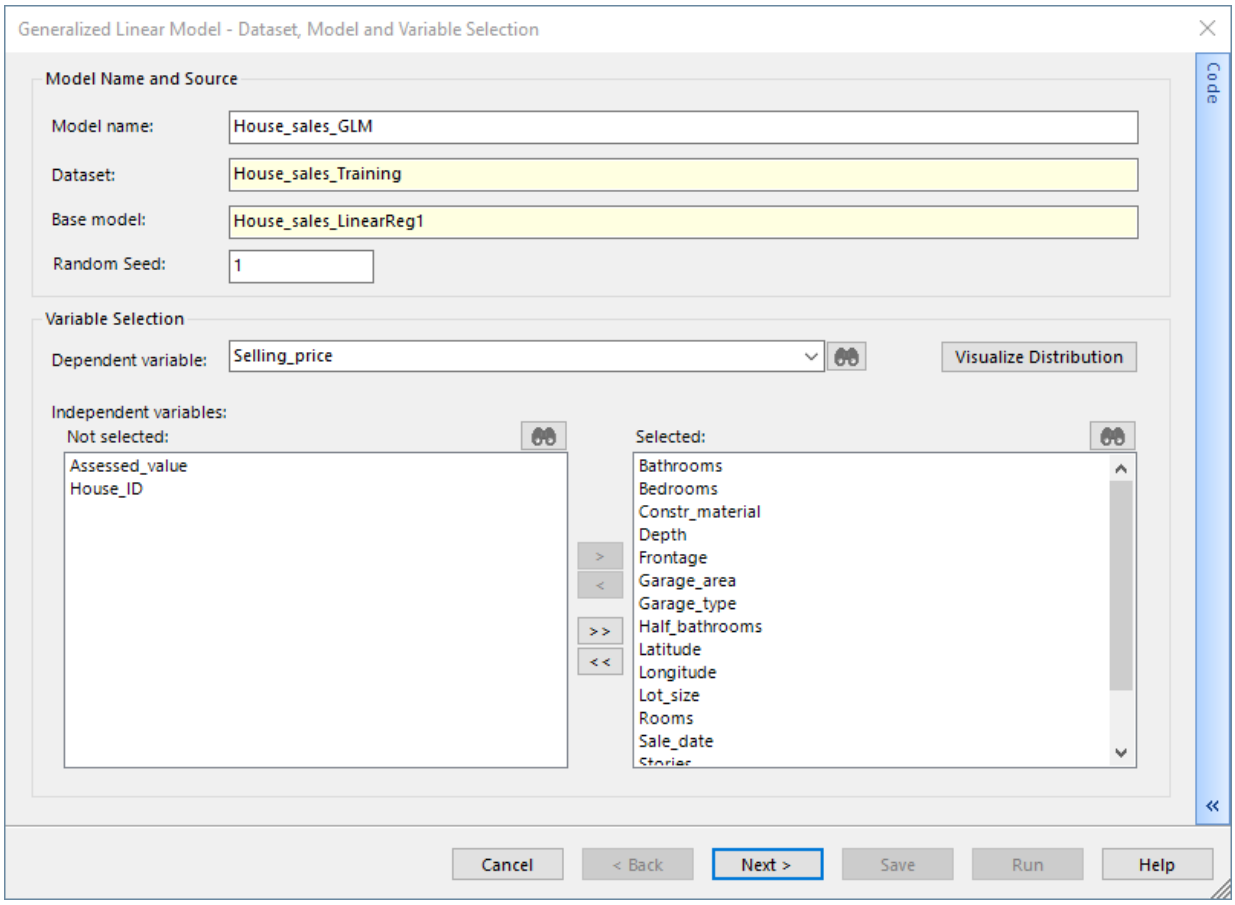
Double-click the Sklearn GLM node or right-click on it and select **Modify** to open the wizard.

In the first page of the wizard:

- Specify the model name
- Change the Random Seed value if required for repeatability.
- Select the dependent variable. Only numeric variables are available for selection.
- Select the independent variables.
    - To select or deselect variables, use the arrow buttons or double-click on the variable names. Advanced field finding and multi-selection capabilities are also available via the Field Finder dialog, which can be opened by clicking the binoculars button.

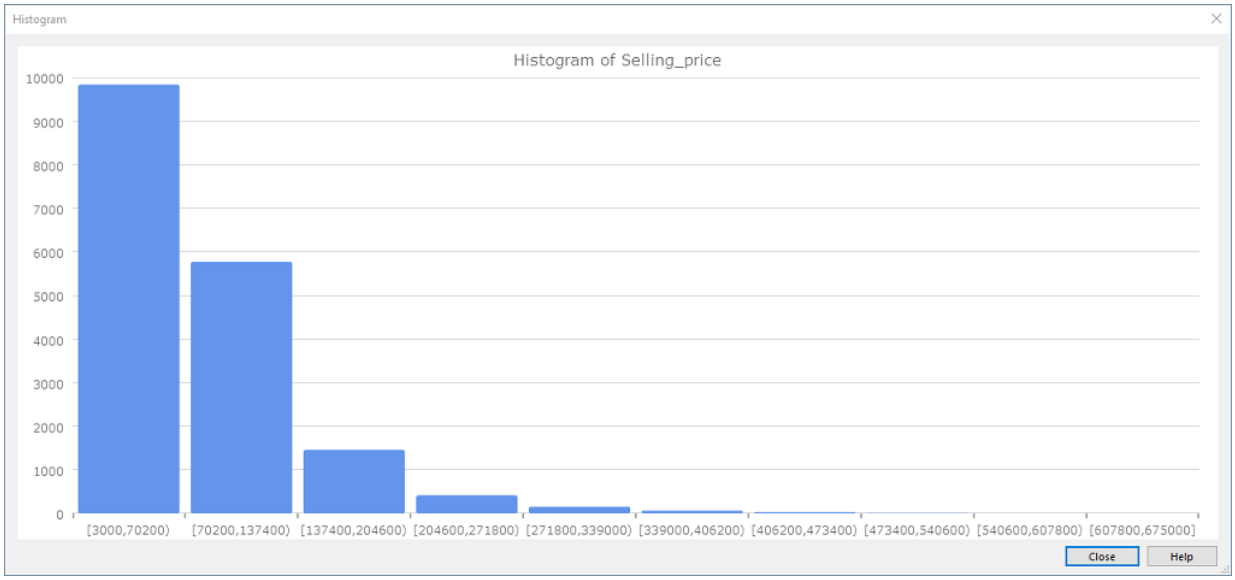
  Note: The dependent variable and independent variables are automatically inherited from the Base model if it is connected.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- Click the button **Visualize Distribution** to see the histogram representing the distribution of the dependent variable.

.



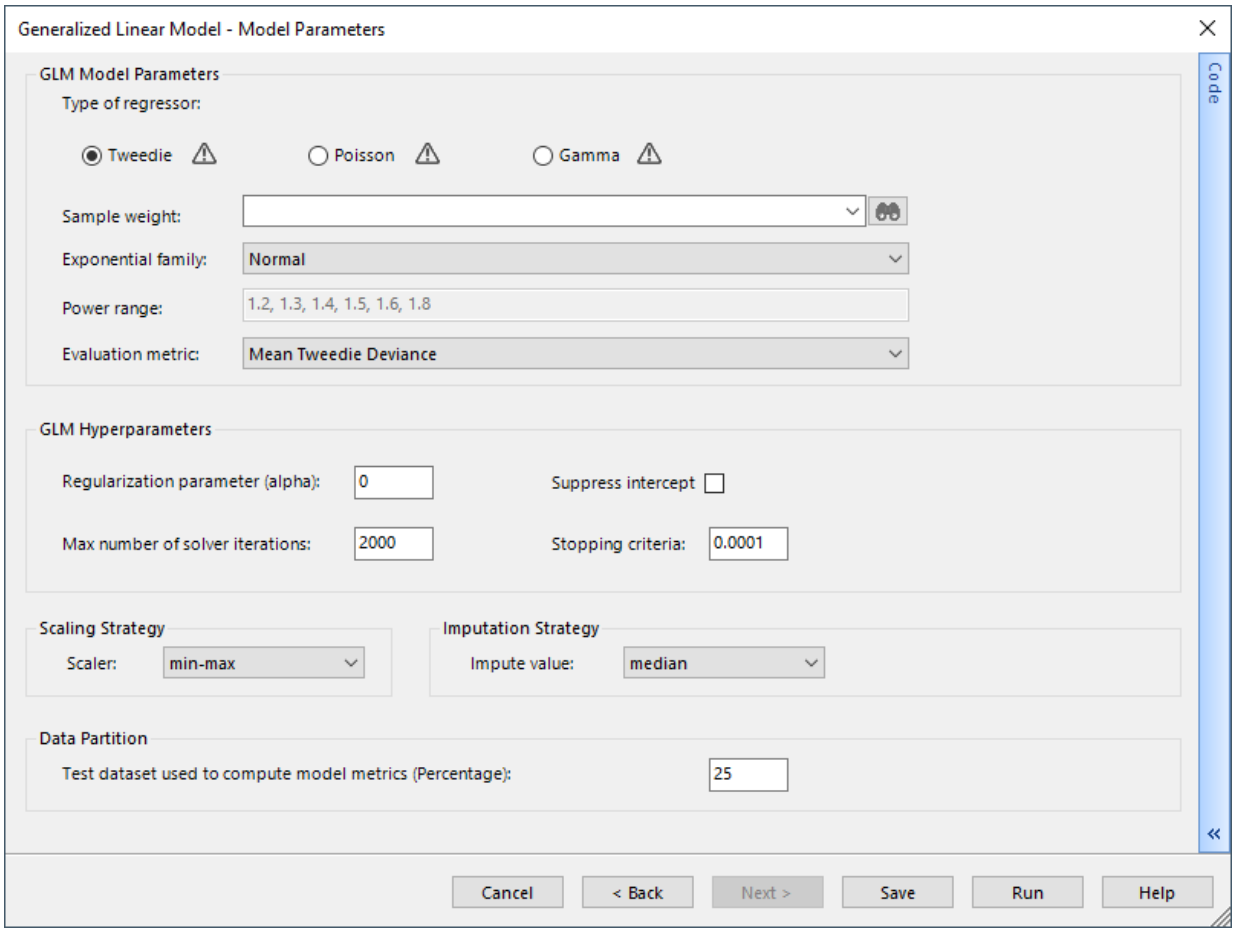
Click **Next**.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

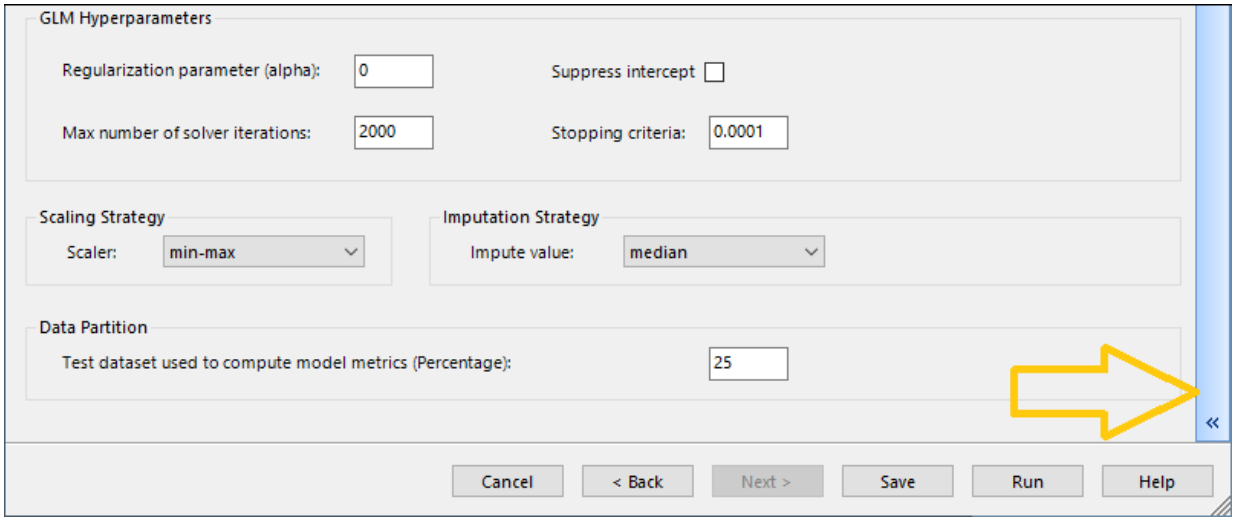
In the Model Parameters page, specify the following GLM model parameters as needed (See the scikit-learn documentation for Generalized Linear Regression for the description of regressor types, evaluation metrics, and other parameters.)

- **Type of regressor**:
  - Tweedie - most commonly used in insurance for modeling Pure Premium. This is the default regressor.
  - Poisson - most commonly used for count data
  - Gamma - used to model positive-valued dependent variables (continuous, positive, right-skewed, and where the variance is near-constant on the log-scale)
- Select **Sample weight** if applicable.
- If the Tweedie regressor is selected:
  - Select the **Exponential family**: *Normal, Poisson, Compound Poisson Gamma, Gamma,* or *Inverse Gaussian*. The default is *Normal*.
    - For *Compound Poisson Gamma*, specify the **Power range** as a comma-separated list. The default is "1.2, 1.3, 1.4, 1.5, 1.6, 1.8".
  - Choose the **Evaluation metric**: Mean Tweedie Deviance, MAE (Mean Average Error), or MSE (Mean Square Error).
- Specify **GLM Hyperparameters**:
  - Regularization parameter (alpha). The default is 0.
  - Select the option to suppress intercept, if necessary.
  - Specify the maximum number of solver iterations. The default is 2000.
  - Specify the evaluation metric threshold for the stopping criterion. The default is 0.0001.
- Specify the **Scaling strategy**:
  - Select the scaler option: *min-max* or *standard*. The default is *min-max*.
  - Select the imputation strategy: impute *median*, *mode*, or *mean* value. The default is *median*.
- For **Data Partition**, specify the percentage of the training dataset to be allocated to compute model metrics. The default is 25% allocated for testing.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

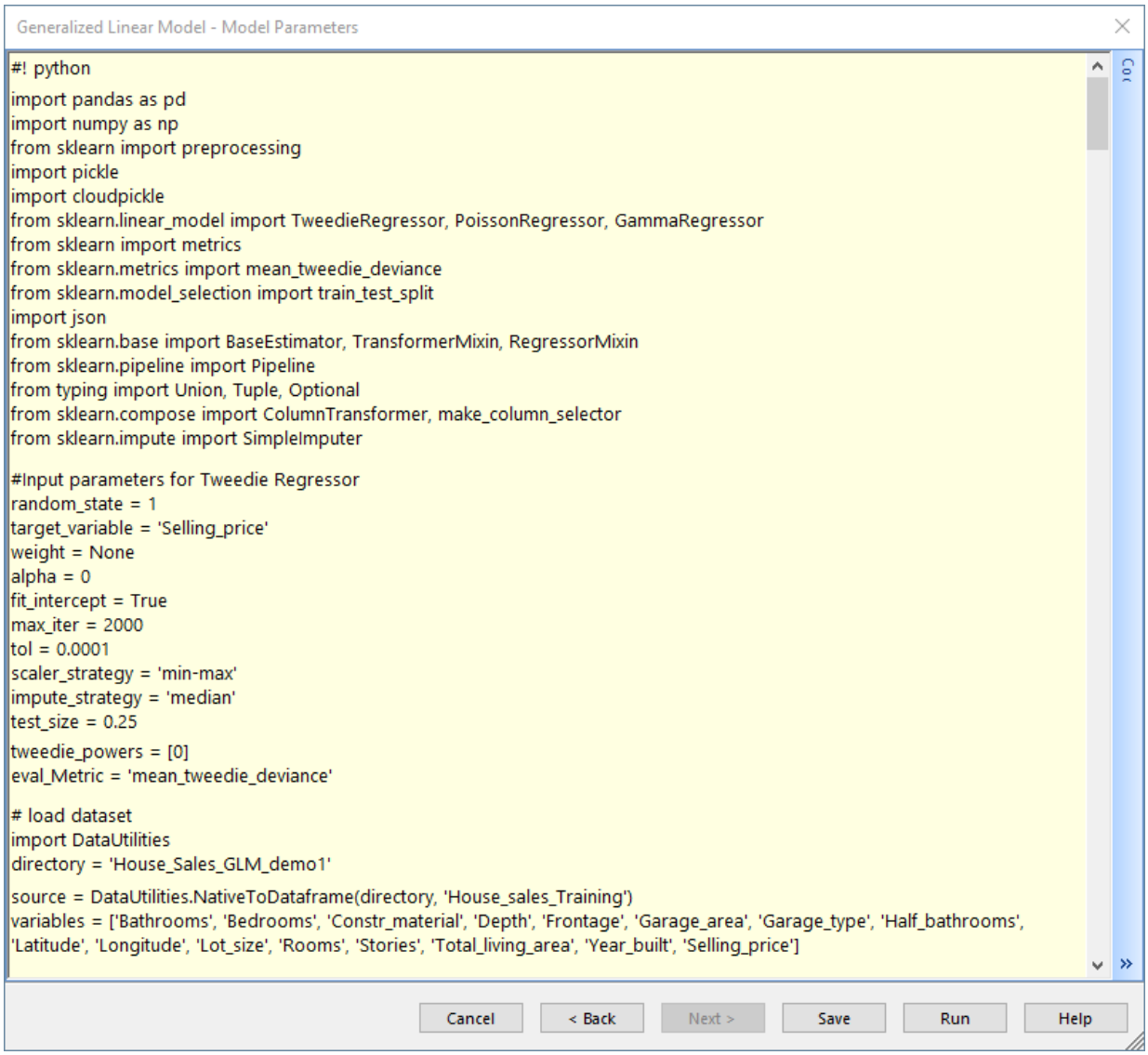


To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog.



This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

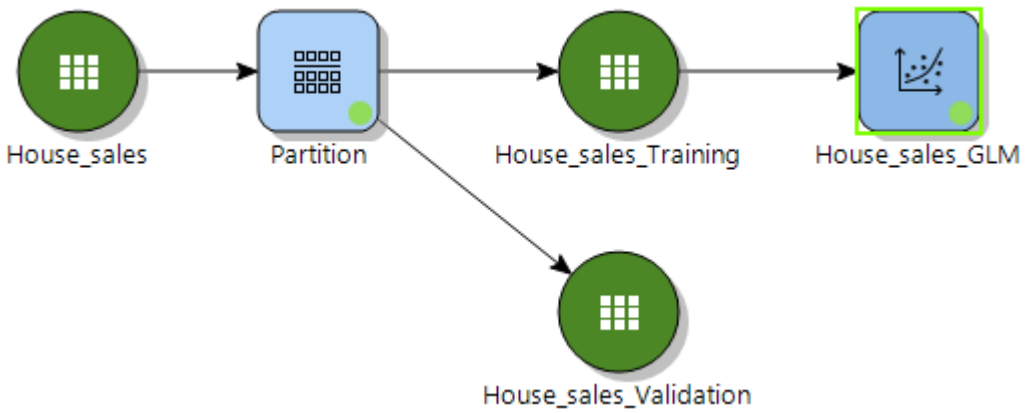
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. The input dataset is converted to a Pandas dataframe, and the linear model functions of the `sklearn` library are applied. The training stops when the total number of iterations reaches the specified limit or when the metric threshold in the stopping criterion is reached.

After running the node, double-click the model node to view the results.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The output dataset can be connected to a Model Analyzer node to analyze the model performance.

Double-click on the Sklearn GLM node to view the training results report. It displays the information on the dependent variable and independent variables, and the values of the evaluation metrics selected in the wizard.

Model Overview

Model Type: Tweedie Regressor

ALTAIR

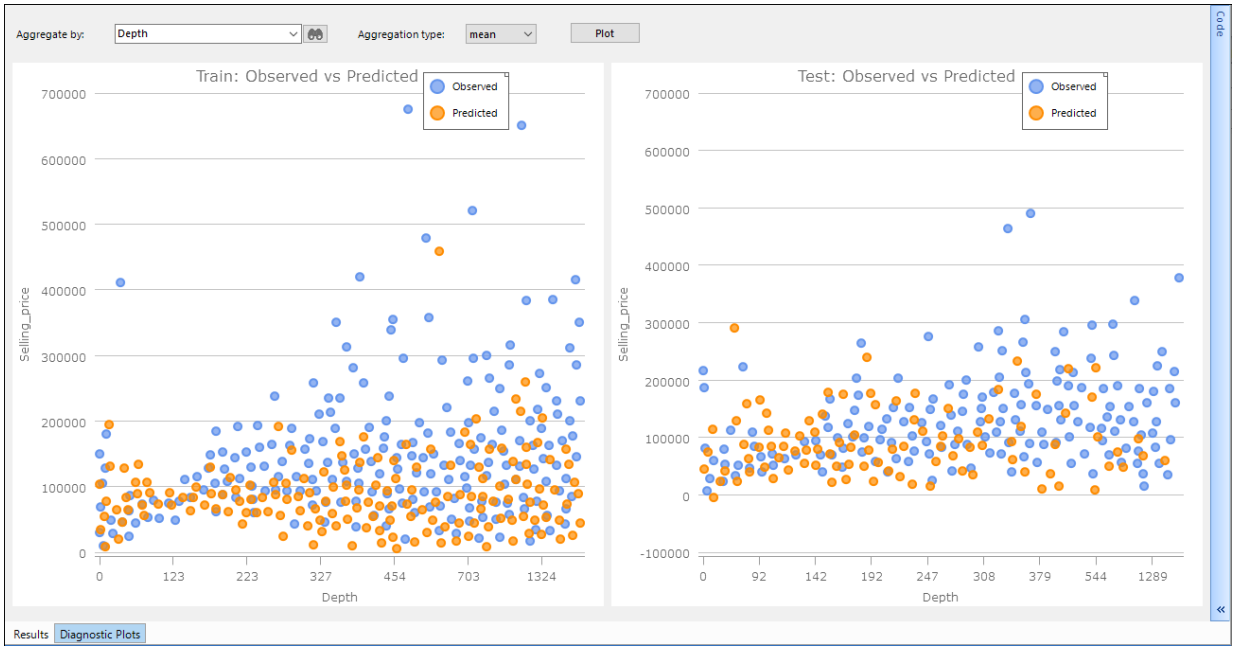
Model Parameters	
Dependent Variable:	Selling_price
Independent Variable(s):	Bathrooms Bedrooms Constr_material Depth Frontage Garage_area Garage_type Half_bathrooms Latitude Longitude Lot_size Rooms Stories Total_living_area Year_built

Training Results		
	Train	Test
Deviance:	1	1
Tweedie Deviation Regression Loss:	N/A	N/A
MAE:	18573.8969192997	18729.4528098179
MSE:	911021508.6766	984171657.829598
RMSE:	30183.1328505939	31371.5102892672

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The **Diagnostic Plots** tab shows the observed and predicted values of the dependent variables aggregated by a selected independent variable on the training and test samples.

- Select the desired independent variable to aggregate by.
- Select the aggregation type (*mean* or *sum*).
- Select the weight, if applicable.
- Click the **Plot** button to build the plots.



After training, the model can be validated on test data or used for scoring new data. See also the Sklearn GLM Validation and Sklearn GLM Scoring topics.

See also: Model Analyzer

ARIMA Forecasting

Overview

Autoregressive Integrated Moving Average (ARIMA) is a time series modeling technique used to explore and forecast time series data points.

The **ARIMA Forecasting** node in the **Model** palette of Knowledge Studio performs ARIMA forecasting with **`pmdarima`** - a Python statistical library for time series forecasting. For details, see the pmdarima documentation at the Python Package Index repository: https://pypi.org/project/pmdarima/. The node wizard provides a convenient graphical interface to specify the options and parameters for `pmdarima`. The wizard automatically translates the inputs into Python code, which can be previewed before running the node. The input dataset is converted from the Altair KDD format to a Python dataframe in the process, then back to produce the output dataset with forecasted series.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Building an ARIMA Forecasting Model

The input dataset should contain at least one Numeric, Integer, Date, or Timestamp field with no missing values to serve as time index. It should also contain at least one field with the series data. If necessary, use the Make Time Series node to aggregate the data to the desired period level to prepare it for ARIMA Forecasting.

In the examples below, data aggregated at the monthly level is used as input to the ARIMA Forecasting model.

Use one of the following ways to add an ARIMA Forecasting node to your workflow:

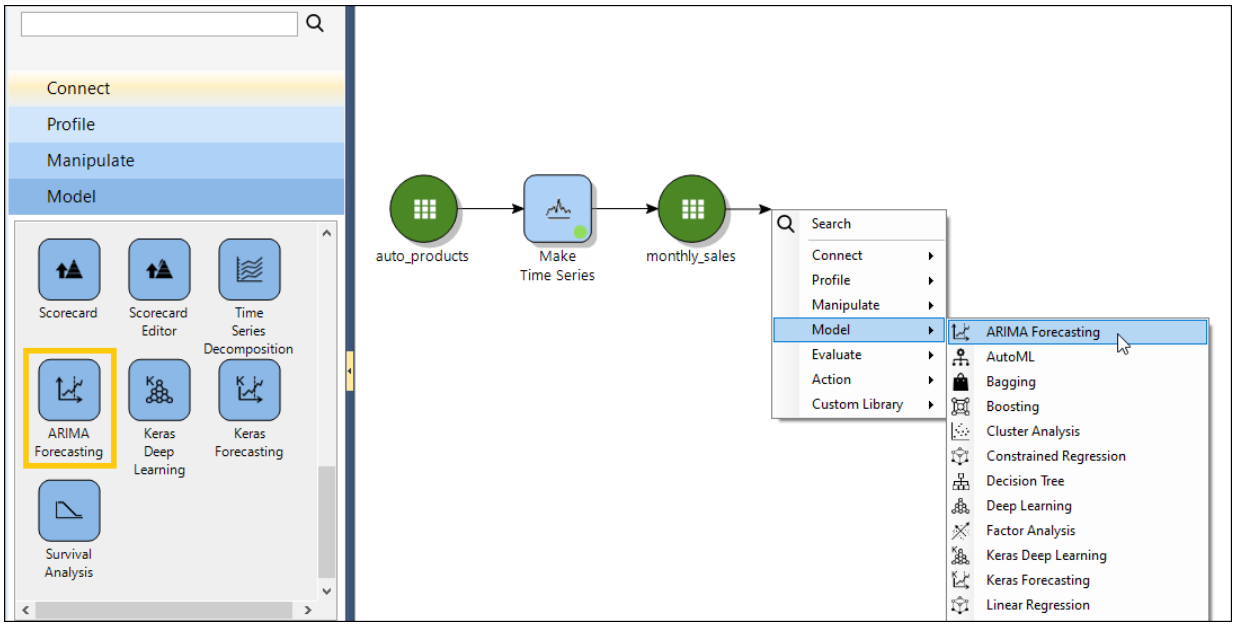
- Drag a connection arrow from the input dataset to any area on the Workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Model --> ARIMA Forecasting** as shown below. A new **ARIMA Forecasting** node will be created, and the wizard will open.

  OR

- Right-click on the Workflow canvas and select **Model --> ARIMA Forecasting** from the context menu. Connect the input dataset node to the ARIMA Forecasting node.

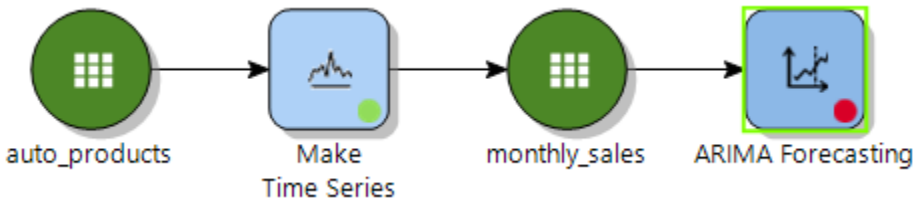
  OR

- Open the **Model** palette and drag the **ARIMA Forecasting** node onto the workflow canvas. Connect the input dataset node to the ARIMA Forecasting node.



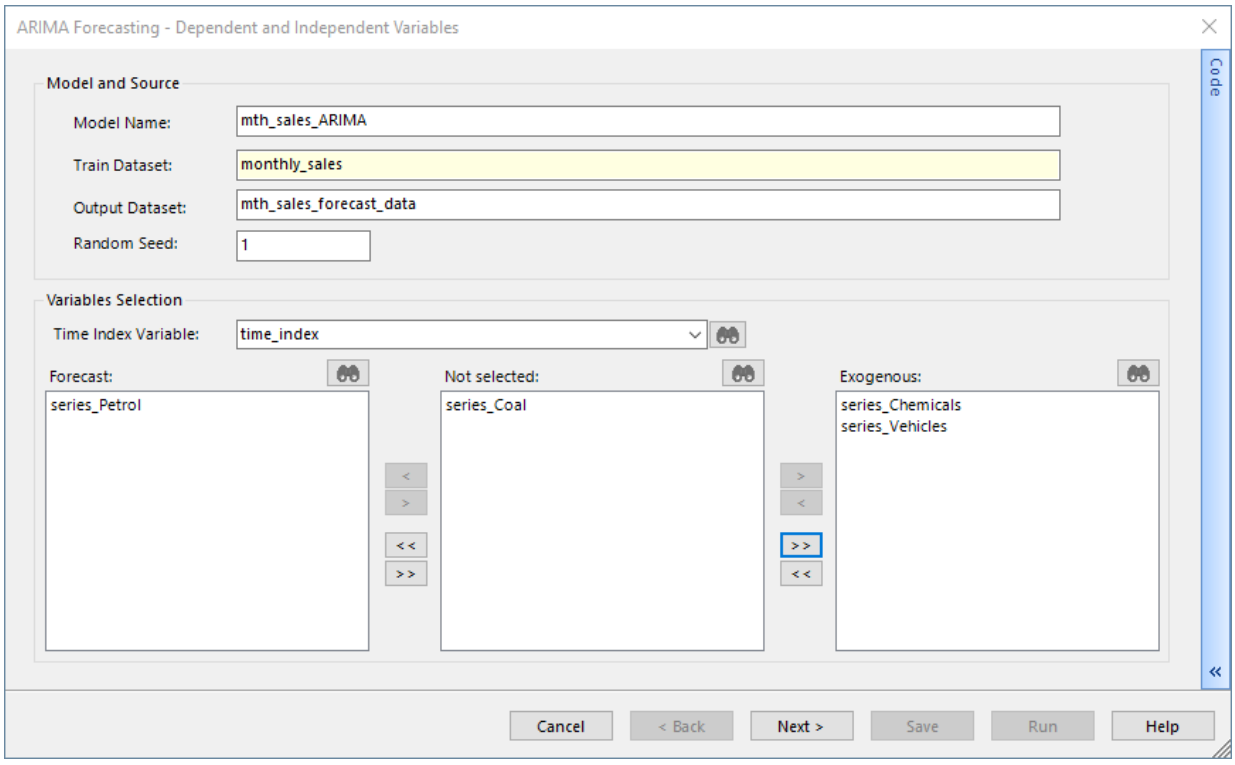
Double-click the ARIMA Forecasting node or right-click on it and select **Modify** to open the wizard.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



In the first page of the wizard:

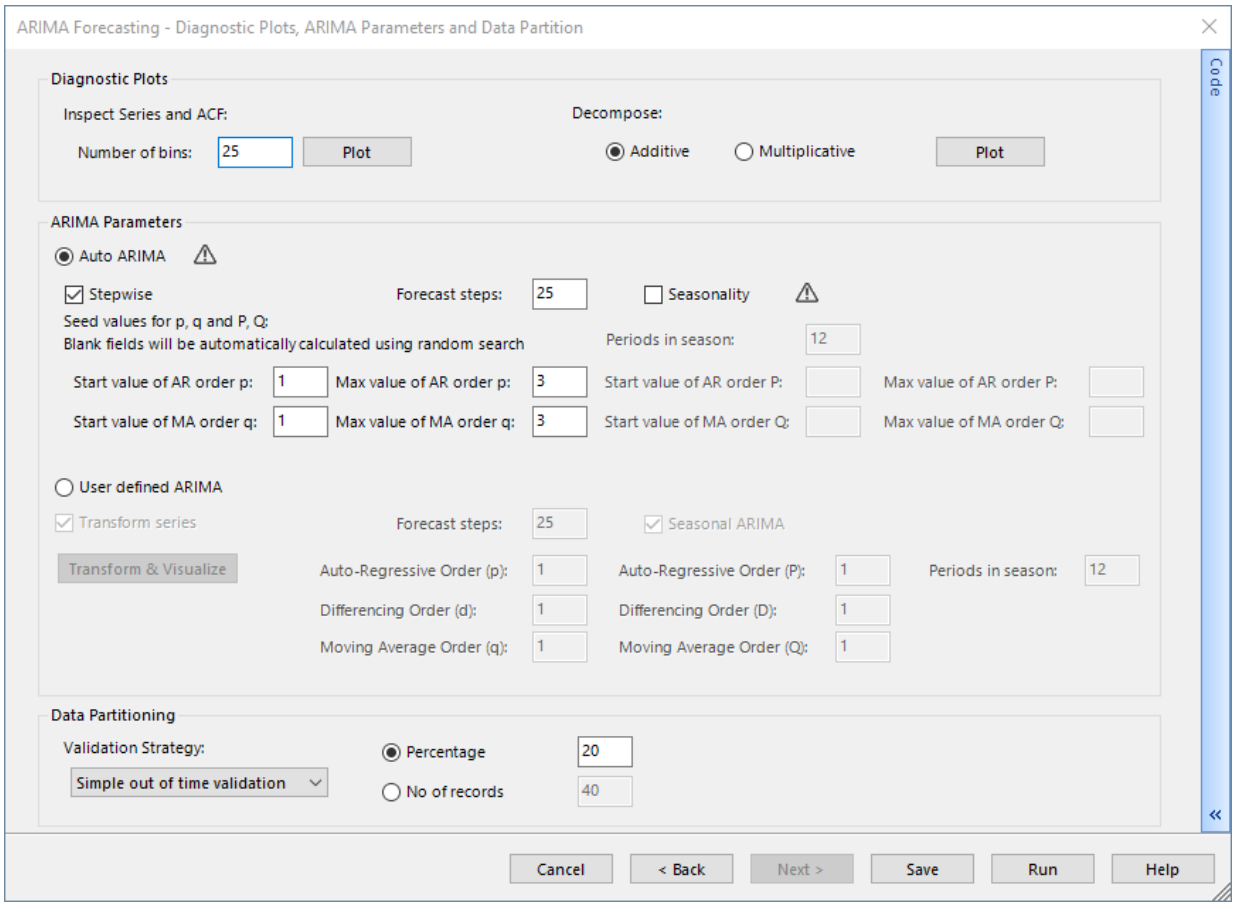
- Specify the names for the ARIMA model and the output dataset.
- Change the Random Seed value if required for repeatability when performing multiple runs.
- Select the ***Time Index Variable***. This must be a Date, Timestamp, Number or Integer variable with no missing values.
- Select one or more variables to forecast by moving them to the *Forecast* list on the left using the arrow buttons.
- Select the exogenous variables by moving them to the *Exogenous* list on the right using the arrow buttons.



Click **Next**.

In the next page, specify the model parameters, options for diagnostic plots, and define data partitioning.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



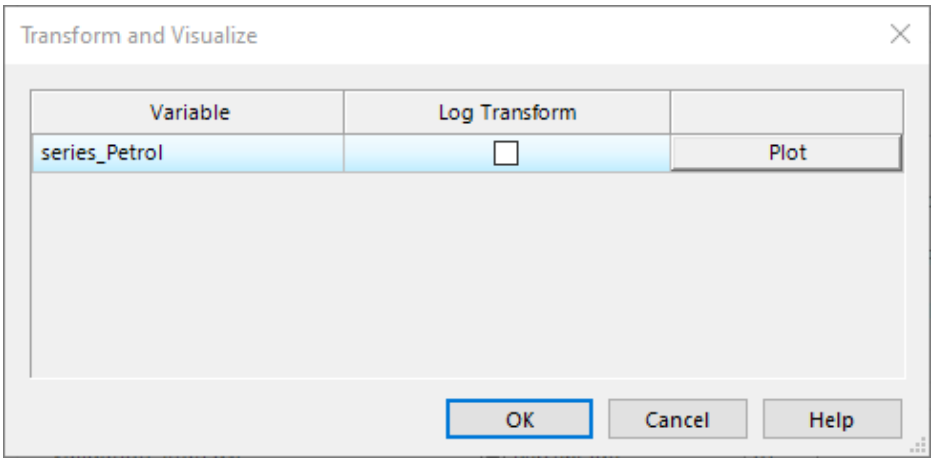
- For the diagnostic plots:
  - specify the number of bins for the series plot and the autocorrelation function (ACF) plot of the series. The default is 25 bins. Click the **Plot** button to preview the plot
  - choose the type of error for time series decomposition: *Additive* or *Multiplicative*. The default type is *Additive*. Click the **Plot** button to preview the plot.
- For the ARIMA model parameters:
  - Select the ***Auto ARIMA*** option for automatic handling of differencing. Select the ***User-defined ARIMA*** option to define differencing parameters manually.
  - If the ***Auto ARIMA*** option is selected:
    - The *Stepwise* mode of the auto_arima function is selected by default. Deselect it to use the Parallelized mode. Note that the Parallelized mode is slower.
    - Specify the number of *Forecast steps* (periods ahead to forecast). The default is 25.
    - Select the *Seasonality* option to specify seasonal model parameters if required. Note that using seasonality may significantly increase the run time. This option is not selected by default.
    - Specify the initial and maximum values for the search of the ARIMA terms `p` and `q` - the orders of the auto-regressive (AR) model and the moving average (MA) model, respectively. If not specified, they will be automatically

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

calculated using random search. The default is 1 for the start values and 3 for the max values.

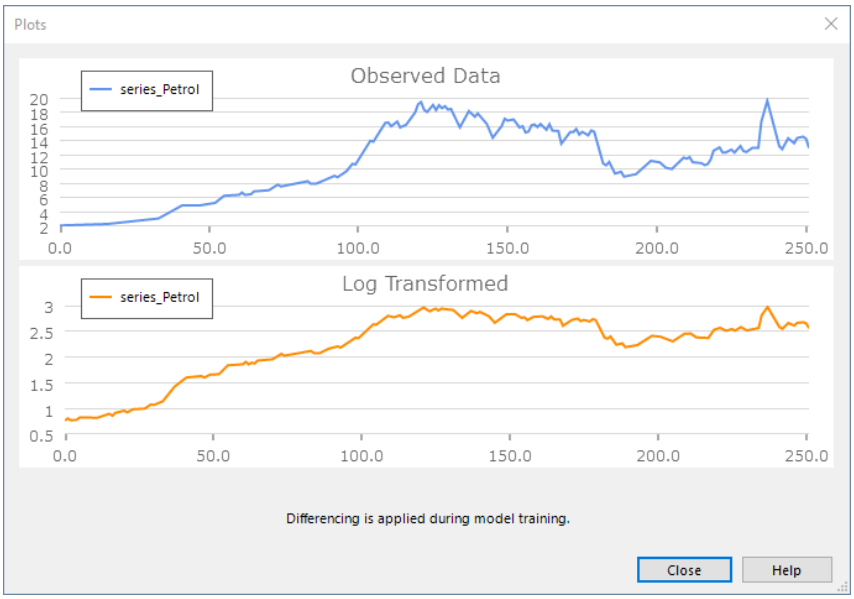
- If the *Seasonality* option is selected, define the seasonal component of the model:
  - Specify the *number of periods per season*. The default is 12.
  - Specify the initial and maximum values for the search of the ARIMA terms P and Q - the orders of the *seasonal components* of the auto-regressive (AR) model and the moving average (MA) model, respectively. If not specified, they will be automatically calculated using random search.

- If the ***User-defined ARIMA*** option is selected:
  - Select the *Transform series* option to apply log transform to the series to enforce stationarity. This option is enabled by default.
  - Click **Transform & Visualize** to define transformations and view the plots for the observed and transformed data.



  - Select the Log Transform option if desired and click the **Plot** button to plot the observed data and log-transformed data. The plots are shown in a pop-up window. If the log transformation is not applicable because the data is outside of the range of admissible values, then the message "*Data is not suitable for log transformation*" is displayed.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

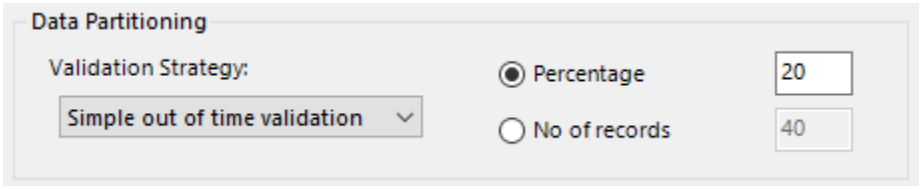


- The output from this step is used as the input for modelling. If you deselect the option "Transform series", then the observed data is used for modelling.

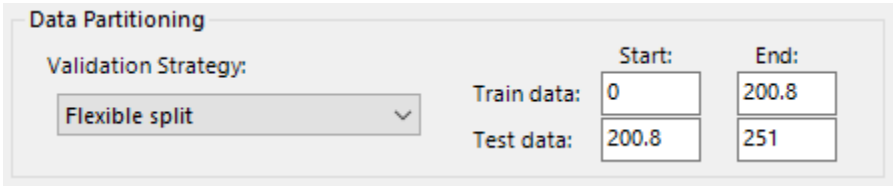
- Specify the number of *Forecast steps* (periods ahead to forecast). The default is 25.
- Specify the values for the ARIMA terms `p`, `d`, and `q` - the orders of the auto-regressive (AR) model, differencing, and the moving average (MA) model, respectively.
- Select the *Seasonal ARIMA* option to specify seasonal model parameters if required.
- If the ***Seasonal ARIMA*** option is selected, define the seasonal component of the model:
  - Specify the number of periods per season. The default is 12.
  - Specify the ARIMA terms `P`, `D`, and `Q`, where `P` and `Q` are the orders of the *seasonal components* of the auto-regressive (AR) model and the moving average (MA) model, respectively, and `D` is the integration order of the seasonal process.

- Specify the Data Partitioning parameters for model validation:
  - Choose a ***Validation strategy***: *Simple out-of-time validation*, *Simple in-sample validation*, or *Flexible split*.
    - For *Simple out-of-time validation*, specify the size of the partition in percentage of the input dataset (the default is 20%) or the number of records. Note that this decreases the number of records for training.

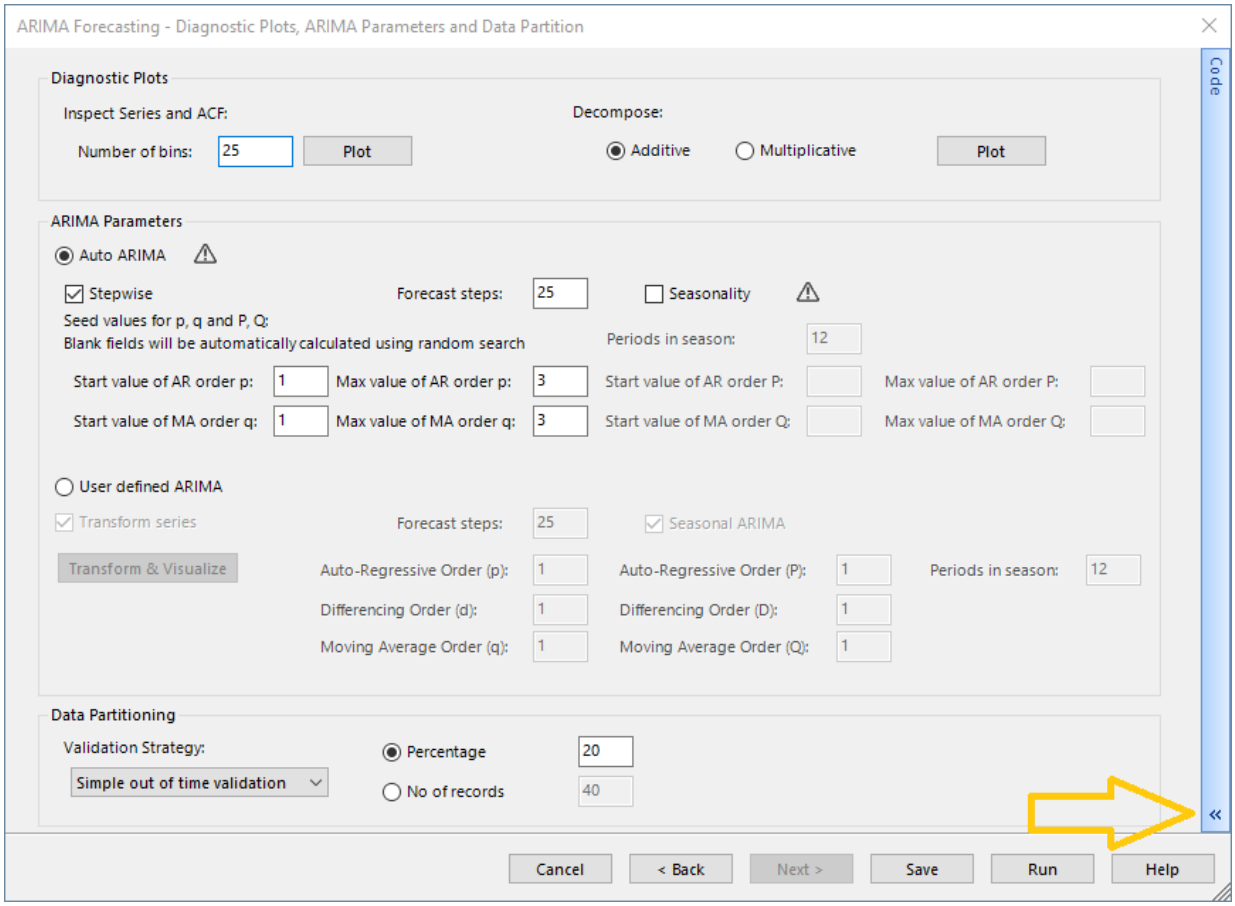
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- For *Flexible split*, specify the start and end points for the Train data and Test data partitions. (These are the data points of the forecasted series.)

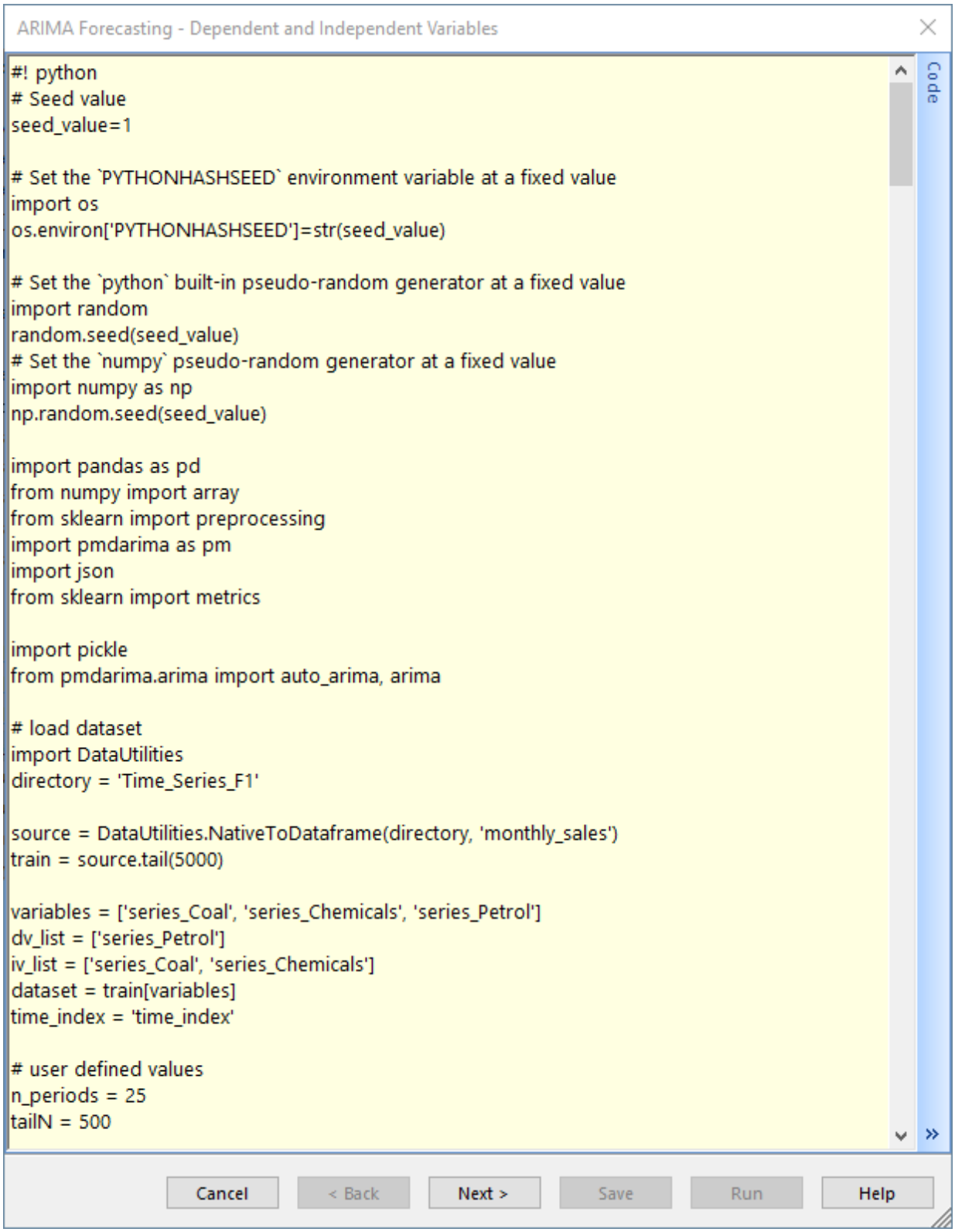


To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog.



This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

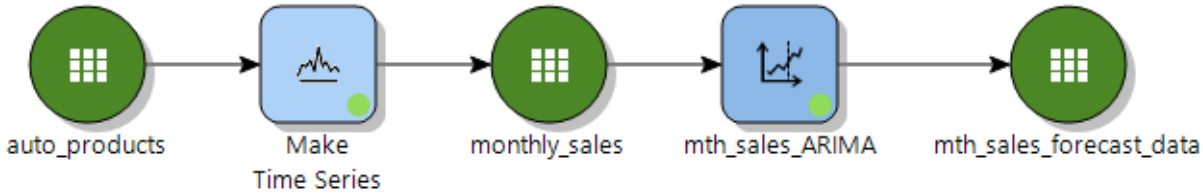


Click **Save** to save the settingsor click Run to save the settings and run the process immediately. The input dataset is converted to a Pandas dataframe. The defined ARIMA Forecasting model runs

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

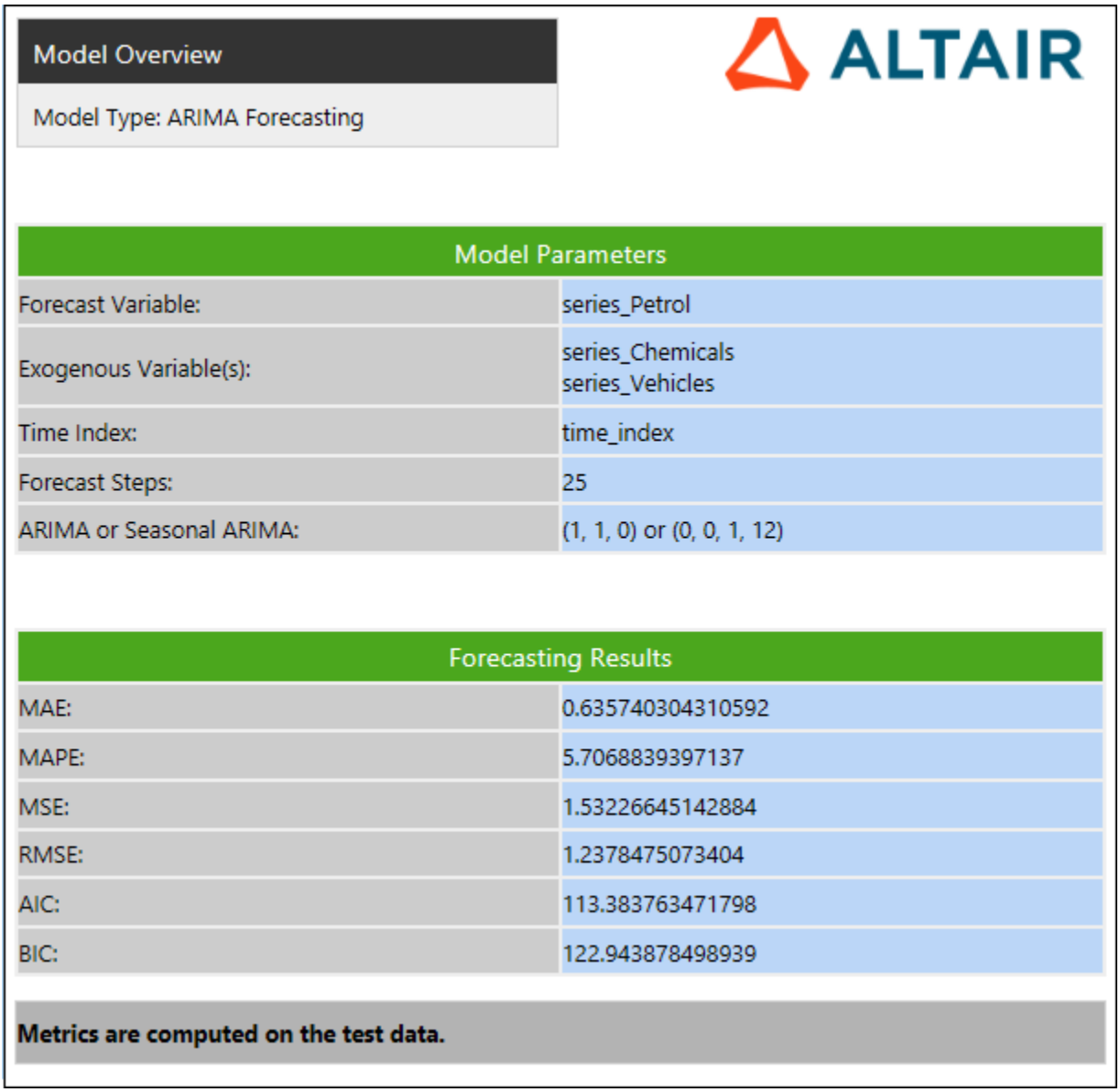
in a Python container session. The output dataset is converted from Pandas to the Altair KDD format.

After running the node, double-click the forecast node and the output dataset node to view the results.



Double-click on the ARIMA Forecasting node to view the summary report, diagnostic plots, and forecast plots.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Model Overview

Model Type: ARIMA Forecasting

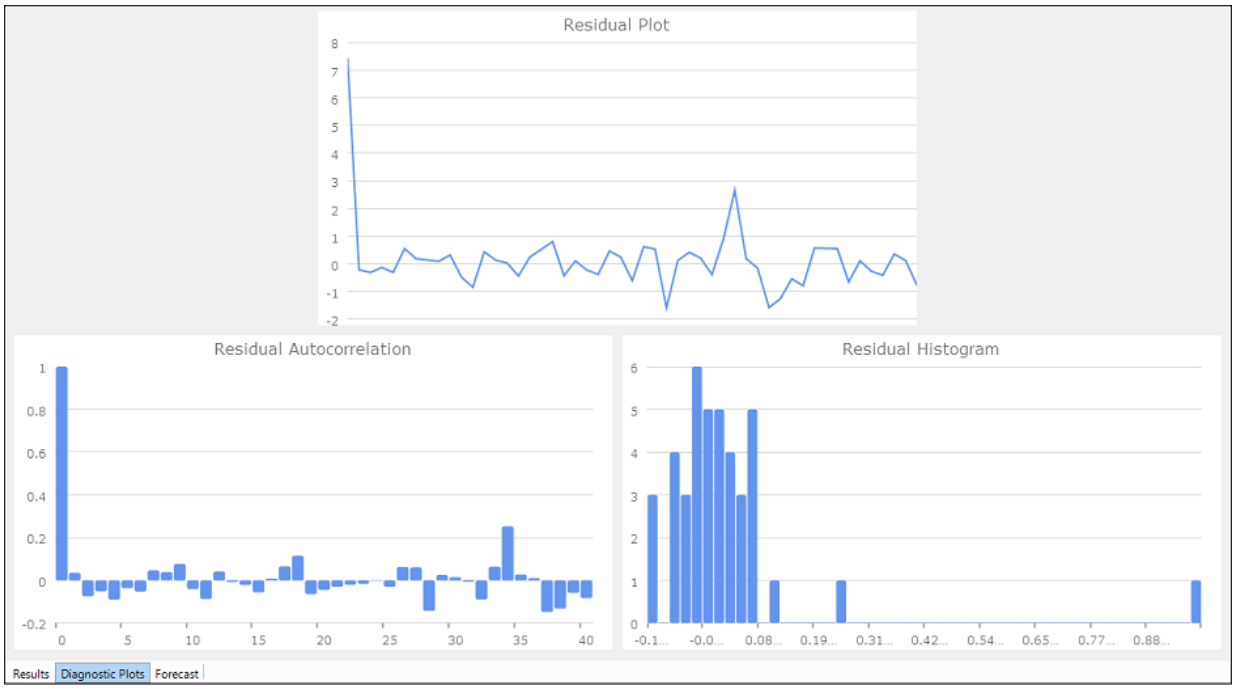
Model Parameters	
Forecast Variable:	series_Petrol
Exogenous Variable(s):	series_Chemicals series_Vehicles
Time Index:	time_index
Forecast Steps:	25
ARIMA or Seasonal ARIMA:	(1, 1, 0) or (0, 0, 1, 12)

Forecasting Results	
MAE:	0.635740304310592
MAPE:	5.7068839397137
MSE:	1.53226645142884
RMSE:	1.2378475073404
AIC:	113.383763471798
BIC:	122.943878498939

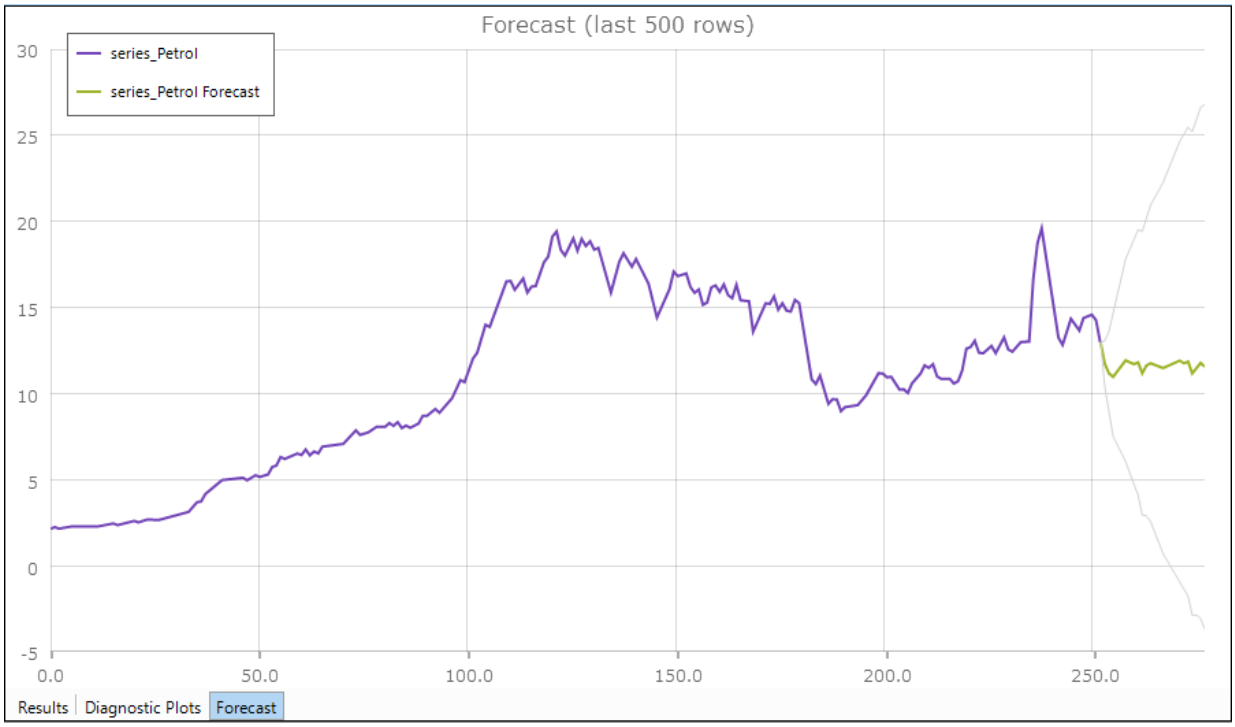
Metrics are computed on the test data.

Diagnostic plots provide a way to validate the time series model. Click the **Diagnostic Plots** tab to view the plots for Residual, Residual Autocorrelation, and Residual Histogram.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

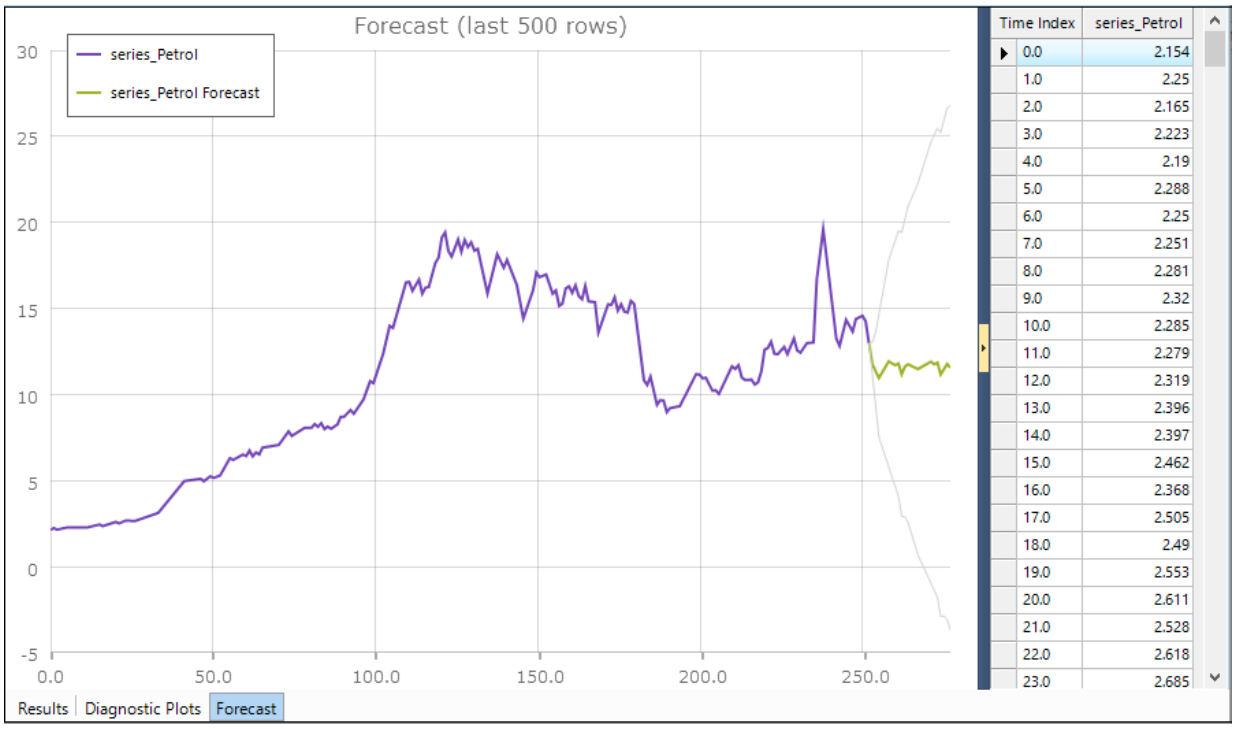


Click the **Forecast** tab. The X axis label shows input data points and the forecasted data. The legend at the top left shows the color codes for the input data and forecasted data for each series.



To view the data table that the forecast plot is based on, click on the small arrow in the middle of the blue bar on the right-hand side. The table will open in a frame on the right. Click the arrow again to hide the frame.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also: ARIMA Scoring, Keras Forecasting.

ARIMA Scoring

Scoring an ARIMA Forecasting model is similar to scoring other models, but it produces forecasted data points instead of probabilities. An ARIMA Forecasting model is scored using the **Scoring** node in the Action palette. For general information about scoring, see the topic About Scoring.

Prerequisites for ARIMA model scoring: an **ARIMA Forecasting** node and an input dataset to score. It is assumed that the input dataset is a ***time series dataset*** containing metadata that identifies one column as the *time index* (series ID) and another column as the time label. Such datasets can be created, for example, using the Make Time Series node in Knowledge Studio. These datasets have special views (integrated tabs) where time series charts and autocorrelation charts van be plotted.

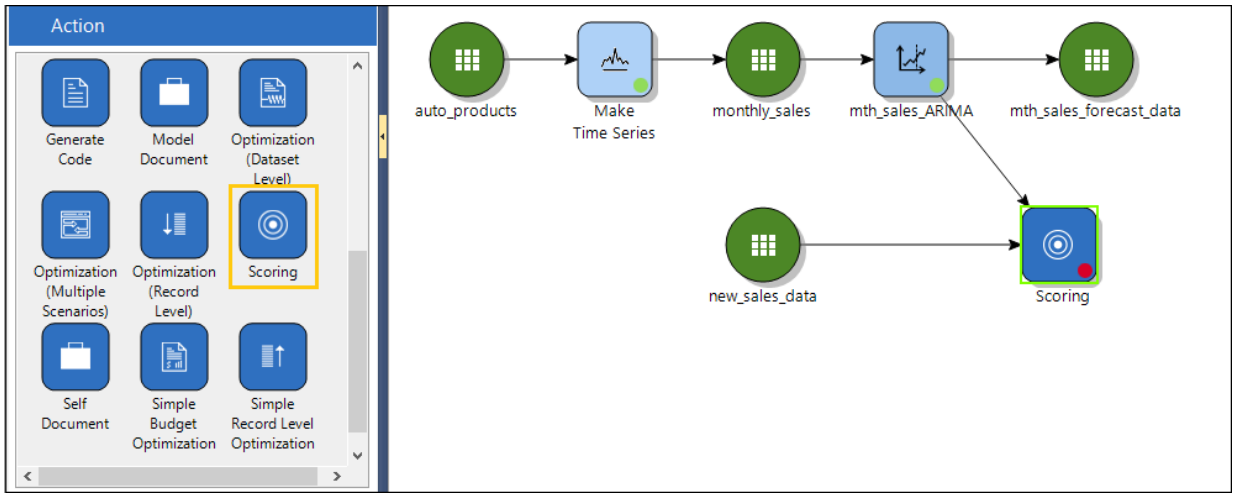
To add a Scoring operation to your workflow:

- Drag the **Scoring** node from the **Action** palette onto the Workflow canvas and connect the inputs: an **ARIMA Forecasting** node and a dataset node that contains the data to score, as shown below.

  OR

- Right-click anywhere on the workflow canvas and select the context menu command **Action --> Scoring**. This will create a new Scoring node. Connect the inputs: an **ARIMA Forecasting** node and a dataset node that contains the data to score.

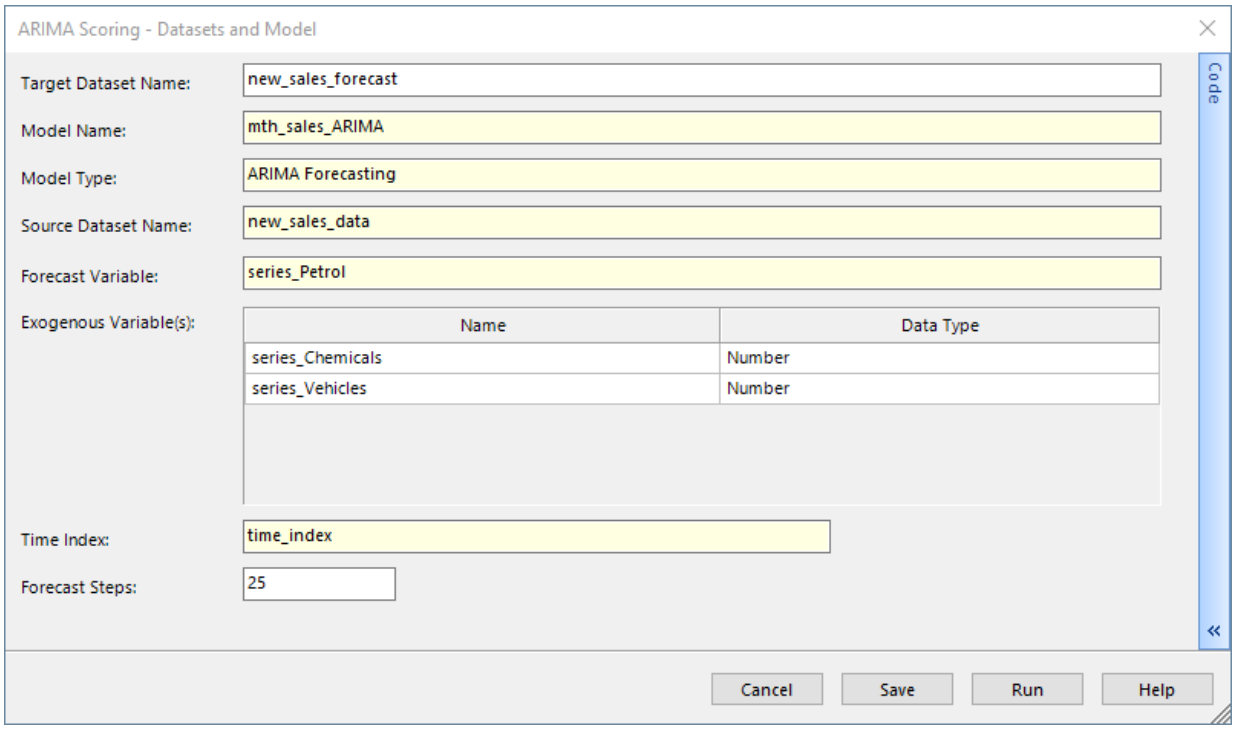
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Double-click the **Scoring** node or right-click on it and select **Modify**.

The Scoring wizard opens. It displays the names of the input model and dataset, the forecast variable, the list of exogenous variables and their data types.

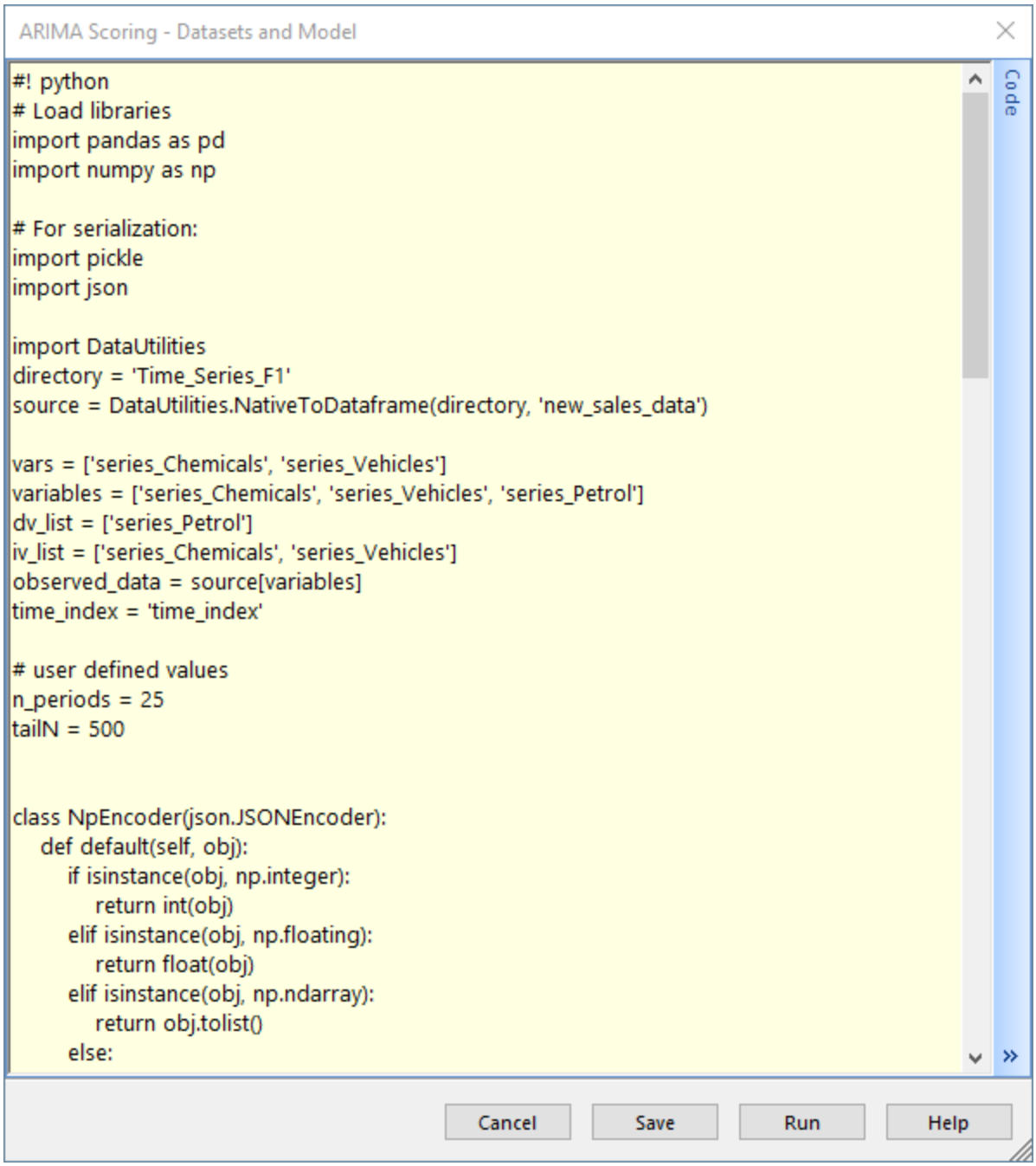
- In the *Target Dataset Name* field, specify the name for the output dataset that will contain the forecasted data.
- Specify the number of *Forecast Steps* (periods ahead to forecast). The default is 25.



To preview the Python code generated for your inputs, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview frame. When viewing the

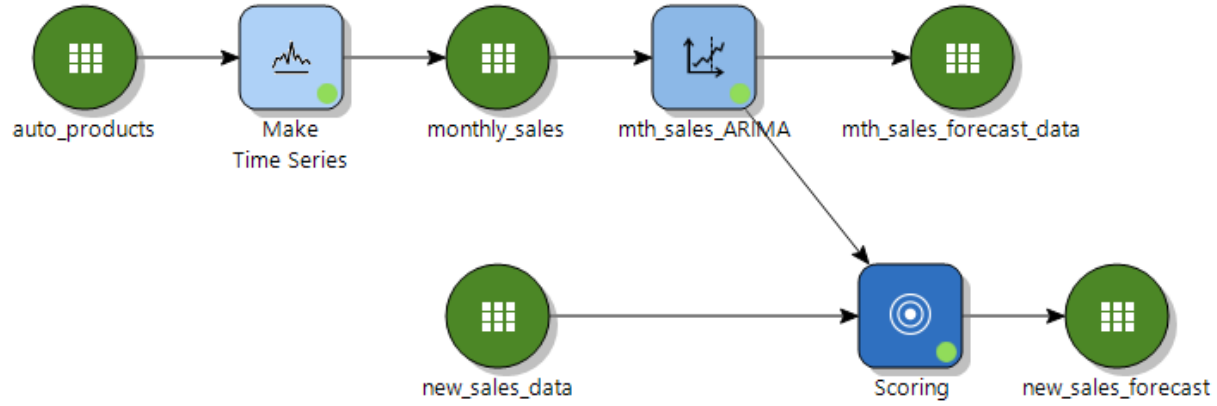
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview frame.



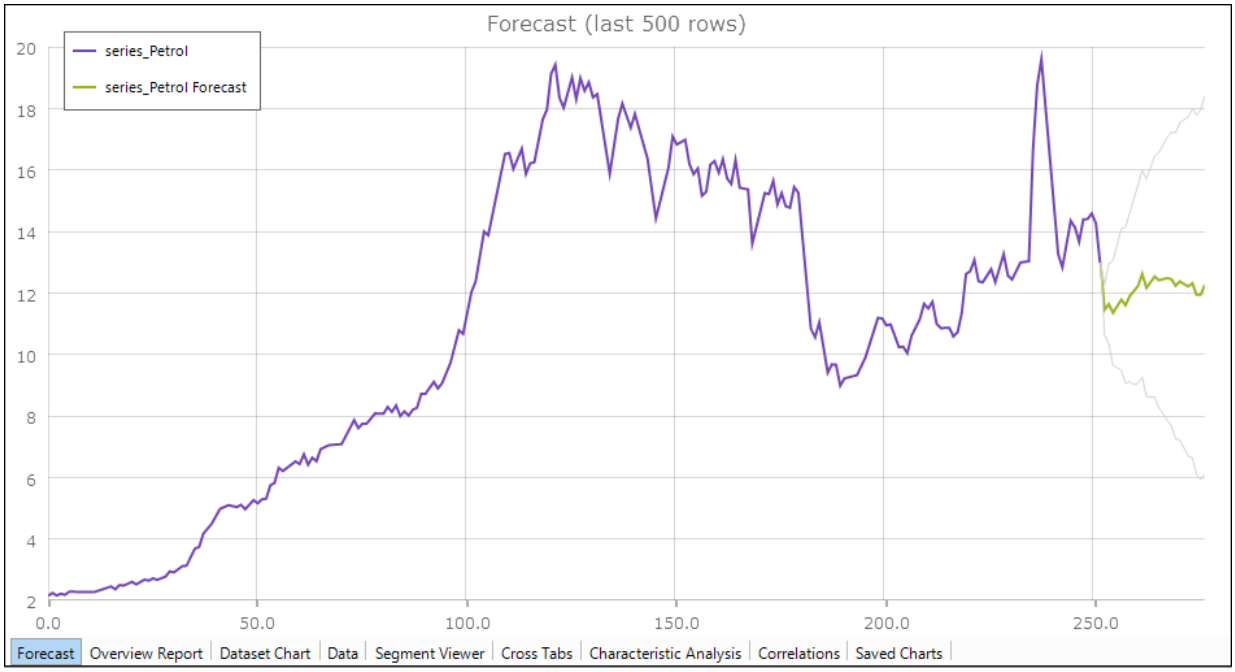
Click **Save** to save the settings, or click **Run** to save the settings and run the scoring process immediately. The model is scored in a Python container session against the input data, the generated forecast data is converted back from Pandas to Altair KDD format and shown as the output dataset node on the workflow canvas.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



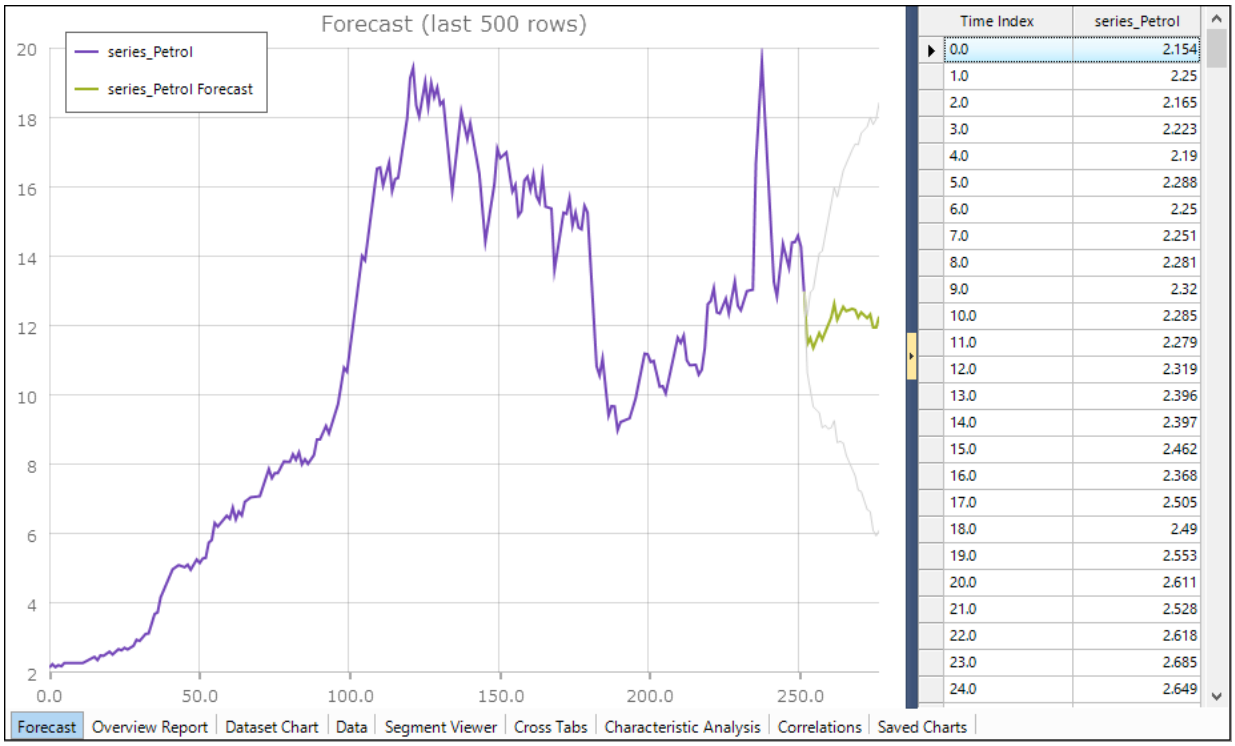
Double-click on the output dataset to view the forecast plot and data.

Click the **Forecast** tab. The plot shows the input data points extended with forecast data for the number of forecast steps specified in the wizard. The legend at the top left shows the color codes for the input data and forecasted data for each series.



To view the underlying data of the forecast plot, click on the small arrow in the middle of the blue bar on the right-hand side. The data table will open in a frame on the right. Click the arrow again to hide the frame.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Time Index	series_Petrol
0.0	2.154
1.0	2.25
2.0	2.165
3.0	2.223
4.0	2.19
5.0	2.288
6.0	2.25
7.0	2.251
8.0	2.281
9.0	2.32
10.0	2.285
11.0	2.279
12.0	2.319
13.0	2.396
14.0	2.397
15.0	2.462
16.0	2.368
17.0	2.505
18.0	2.49
19.0	2.553
20.0	2.611
21.0	2.528
22.0	2.618
23.0	2.685
24.0	2.649

See also: ARIMA Forecasting

Generalized Linear Model

Overview

The **Sklearn GLM** node in the **Model** palette of Knowledge Studio allows you to build regression models with a dependent variable that does not necessarily follow the normal distribution. The node wizard is similar to that of the Linear Regression node, but it works through integration with Python libraries. It uses regressors of the linear model methods of **`scikit-learn`** - a popular Python library for machine learning. For details, see the scikit-learn documentation for Generalized Linear Regression.

The node wizard provides a convenient graphical interface to specify the options and parameters for linear model functions in the `scikit-learn (sklearn)` library. The wizard automatically translates the inputs into Python code, which can be previewed before running the node. The code then runs in a container with the necessary Python packages and their dependencies. The input dataset is converted from the Altair KDD format to pandas dataframes in the process. The resulting dataframe is converted back to produce the output Altair dataset.

After training, the model can be validated on test data using the Sklearn GLM Validation node or used for scoring new data with the Sklearn GLM Scoring node.

Building an Sklearn GLM Model

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

An **Sklearn GLM** model requires a single dataset as input. It can optionally take another predictive model as the second input (*base model*), in which case it inherits the dependent variable and independent variable selection from the base model.

An **Sklearn GLM** node can be added to the workflow in one of the following ways:

- Drag a connection arrow from the training dataset node to any area on the Workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Model --> Sklearn GLM** as in the example screenshot below. A new **Sklearn GLM** node will be created, and the wizard will open.

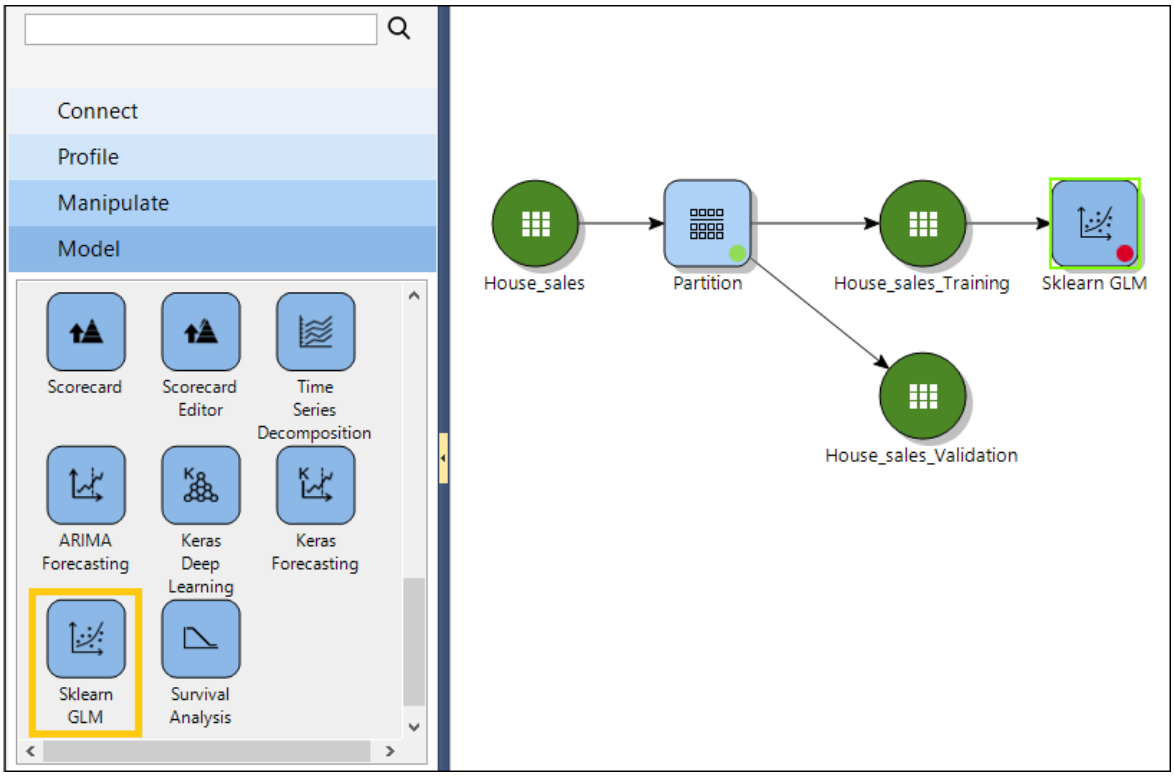
OR

- Right-click on the Workflow canvas and select **Model --> Sklearn GLM** from the context menu. Connect the training dataset. Optionally, connect another predictive model node as a base model to use as a template.

OR

- Open the **Model** palette and drag the **Sklearn GLM** node onto the workflow canvas. Connect the training dataset. Optionally, connect another predictive model node as a base model to use as a template.

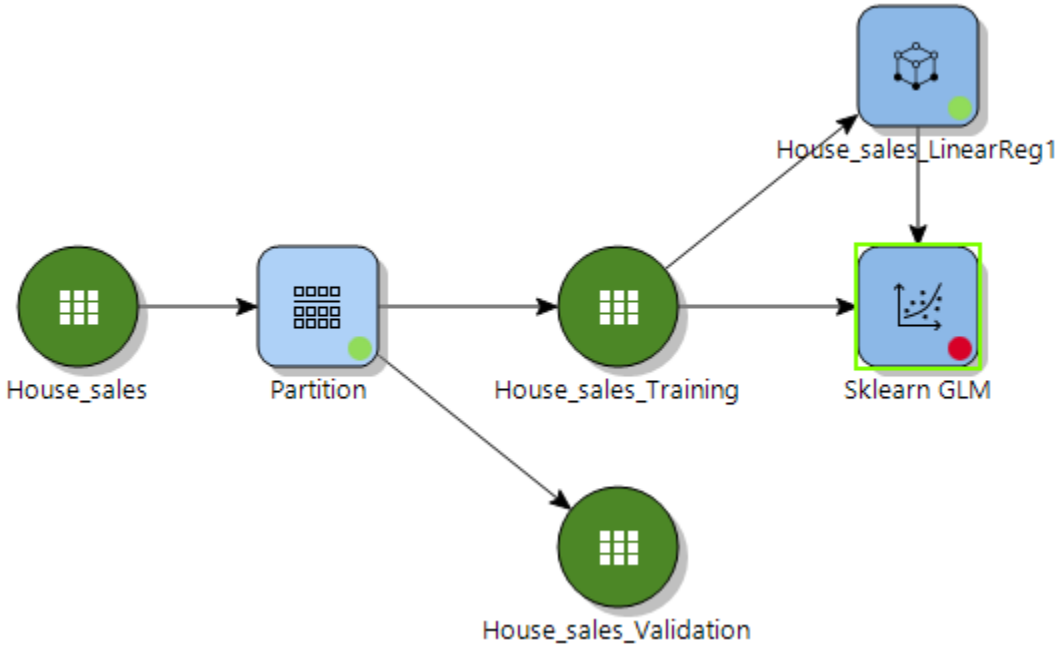
Example 1:



Example 2 (with base model)

In this example, an existing Linear Regression model is connected to the new Sklearn GLM model as a base model.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



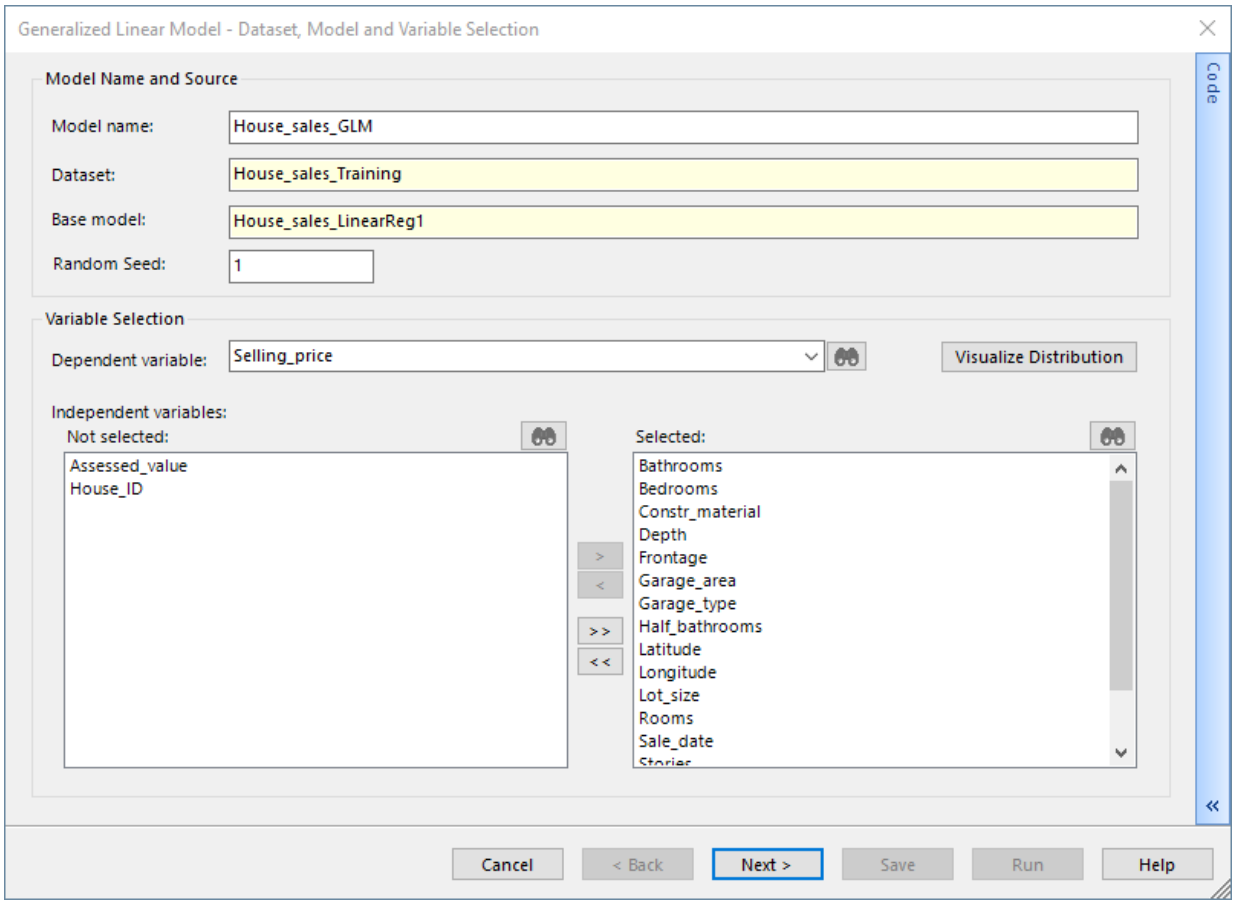
Double-click the Sklearn GLM node or right-click on it and select **Modify** to open the wizard.

In the first page of the wizard:

- Specify the model name
- Change the Random Seed value if required for repeatability.
- Select the dependent variable. Only numeric variables are available for selection.
- Select the independent variables.
    - To select or deselect variables, use the arrow buttons or double-click on the variable names. Advanced field finding and multi-selection capabilities are also available via the Field Finder dialog, which can be opened by clicking the binoculars button.

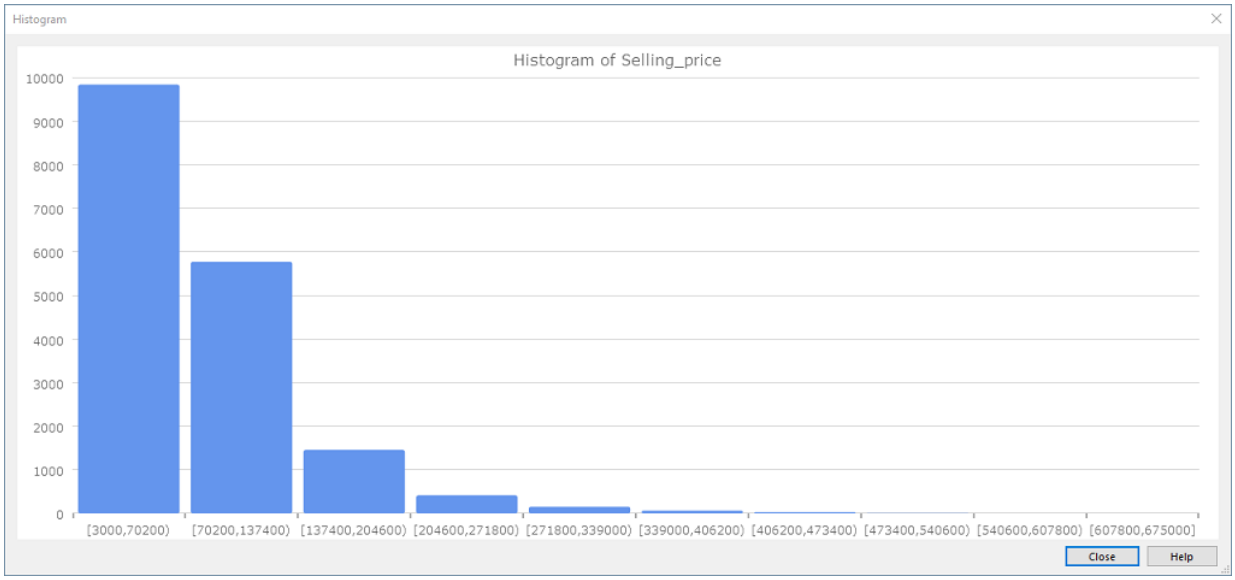
  Note: The dependent variable and independent variables are automatically inherited from the Base model if it is connected.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- Click the button **Visualize Distribution** to see the histogram representing the distribution of the dependent variable.

.



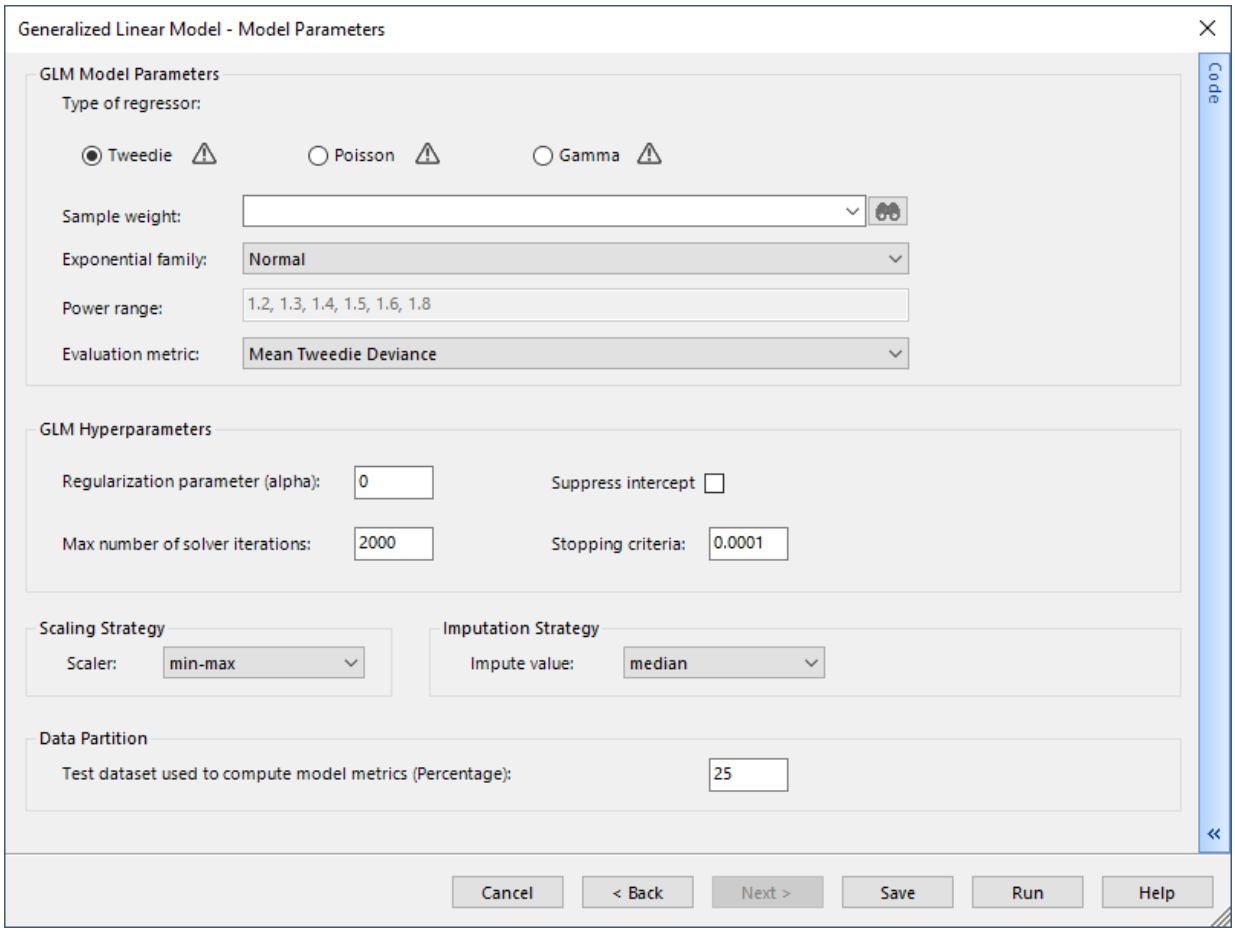
Click **Next**.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

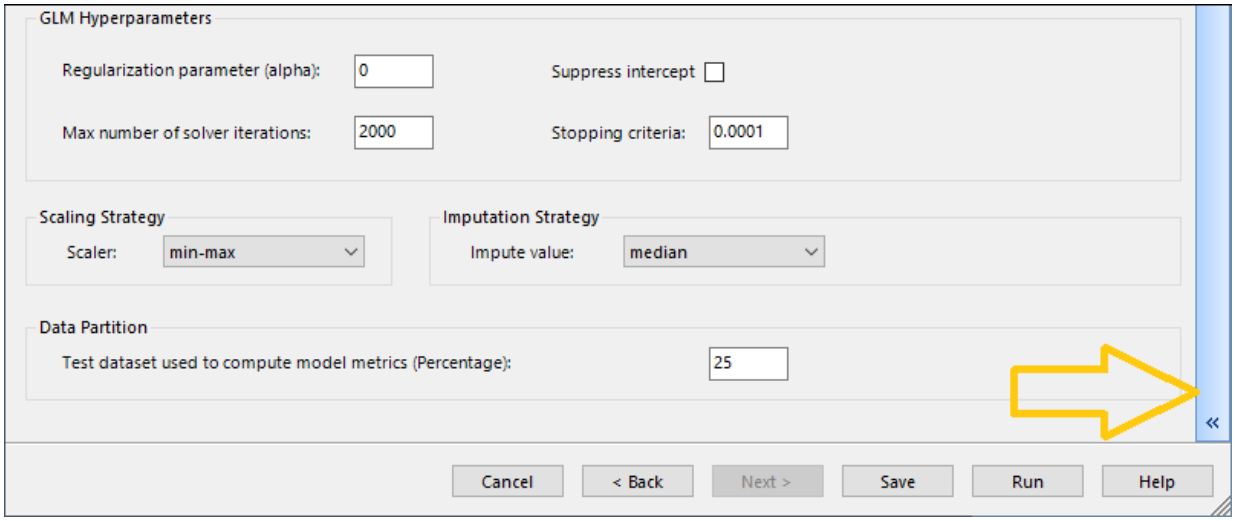
In the Model Parameters page, specify the following GLM model parameters as needed (See the scikit-learn documentation for Generalized Linear Regression for the description of regressor types, evaluation metrics, and other parameters.)

- **Type of regressor**:
  - Tweedie - most commonly used in insurance for modeling Pure Premium. This is the default regressor.
  - Poisson - most commonly used for count data
  - Gamma - used to model positive-valued dependent variables (continuous, positive, right-skewed, and where the variance is near-constant on the log-scale)
- Select **Sample weight** if applicable.
- If the Tweedie regressor is selected:
  - Select the **Exponential family**: *Normal, Poisson, Compound Poisson Gamma, Gamma,* or *Inverse Gaussian*. The default is *Normal*.
    - For *Compound Poisson Gamma*, specify the **Power range** as a comma-separated list. The default is "1.2, 1.3, 1.4, 1.5, 1.6, 1.8".
  - Choose the **Evaluation metric**: Mean Tweedie Deviance, MAE (Mean Average Error), or MSE (Mean Square Error).
- Specify **GLM Hyperparameters**:
  - Regularization parameter (alpha). The default is 0.
  - Select the option to suppress intercept, if necessary.
  - Specify the maximum number of solver iterations. The default is 2000.
  - Specify the evaluation metric threshold for the stopping criterion. The default is 0.0001.
- Specify the **Scaling strategy**:
  - Select the scaler option: *min-max* or *standard*. The default is *min-max*.
  - Select the imputation strategy: impute *median*, *mode*, or *mean* value. The default is *median*.
- For **Data Partition**, specify the percentage of the training dataset to be allocated to compute model metrics. The default is 25% allocated for testing.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

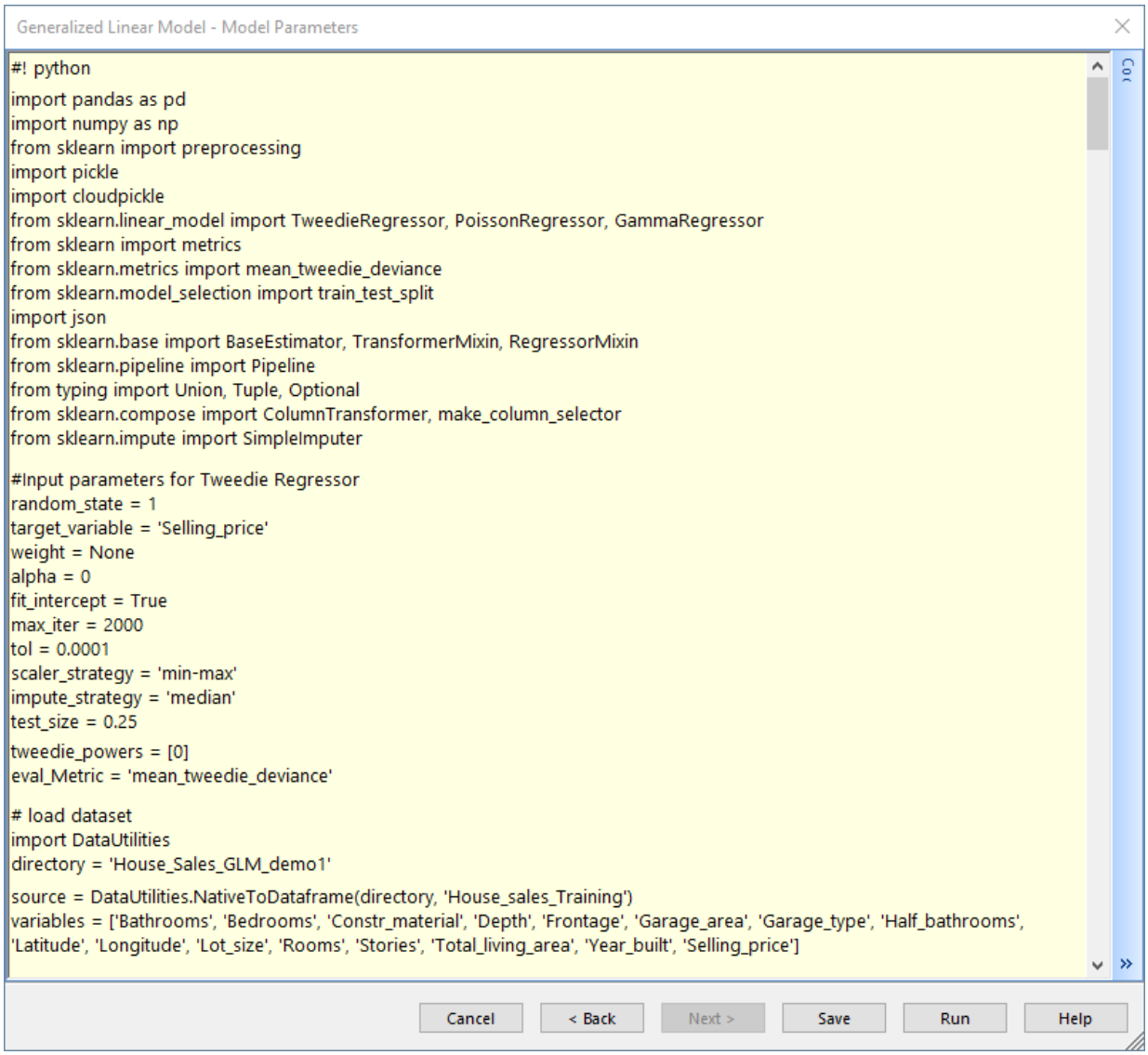


To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog.



This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

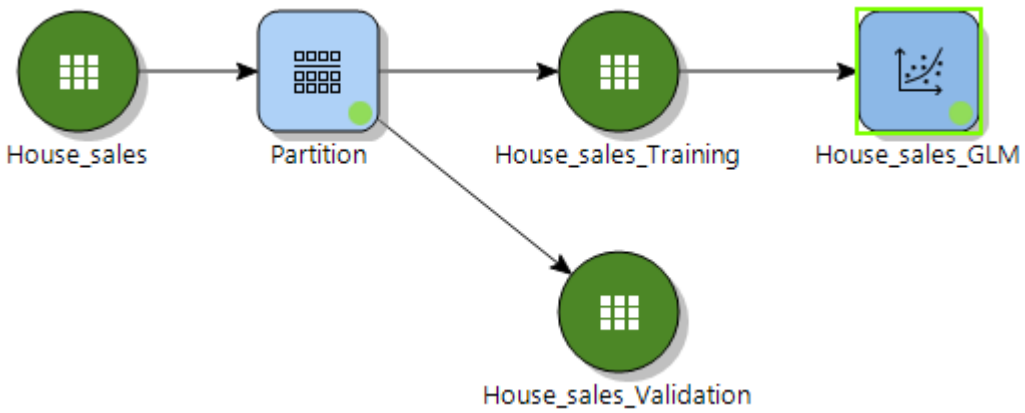
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately. The input dataset is converted to a Pandas dataframe, and the linear model functions of the `sklearn` library are applied. The training stops when the total number of iterations reaches the specified limit or when the metric threshold in the stopping criterion is reached.

After running the node, double-click the model node to view the results.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The output dataset can be connected to a Model Analyzer node to analyze the model performance.

Double-click on the Sklearn GLM node to view the training results report. It displays the information on the dependent variable and independent variables, and the values of the evaluation metrics selected in the wizard.

Model Overview

Model Type: Tweedie Regressor

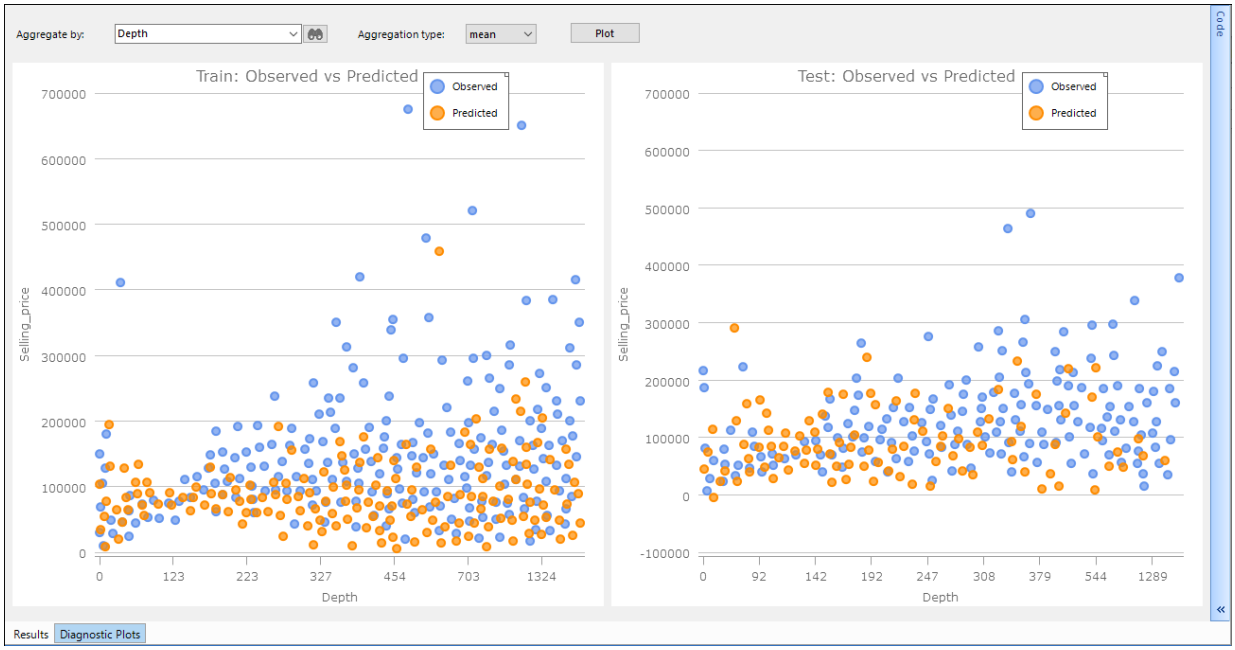
Model Parameters	
Dependent Variable:	Selling_price
Independent Variable(s):	Bathrooms Bedrooms Constr_material Depth Frontage Garage_area Garage_type Half_bathrooms Latitude Longitude Lot_size Rooms Stories Total_living_area Year_built

Training Results	Train	Test
Deviance:	1	1
Tweedie Deviation Regression Loss:	N/A	N/A
MAE:	18573.8969192997	18729.4528098179
MSE:	911021508.6766	984171657.829598
RMSE:	30183.1328505939	31371.5102892672

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The **Diagnostic Plots** tab shows the observed and predicted values of the dependent variables aggregated by a selected independent variable on the training and test samples.

- Select the desired independent variable to aggregate by.
- Select the aggregation type (*mean* or *sum*).
- Select the weight, if applicable.
- Click the **Plot** button to build the plots.



After training, the model can be validated on test data or used for scoring new data. See also the Sklearn GLM Validation and Sklearn GLM Scoring topics.

See also: Model Analyzer

Sklearn GLM Validation

Validation of Sklearn GLM models is performed using the **Model Validation** node in the **Evaluate** palette.

For general information about validation, see the topic About Validation.

Prerequisites for Sklearn GLM model validation: Sklearn GLM node and an input dataset to validate.

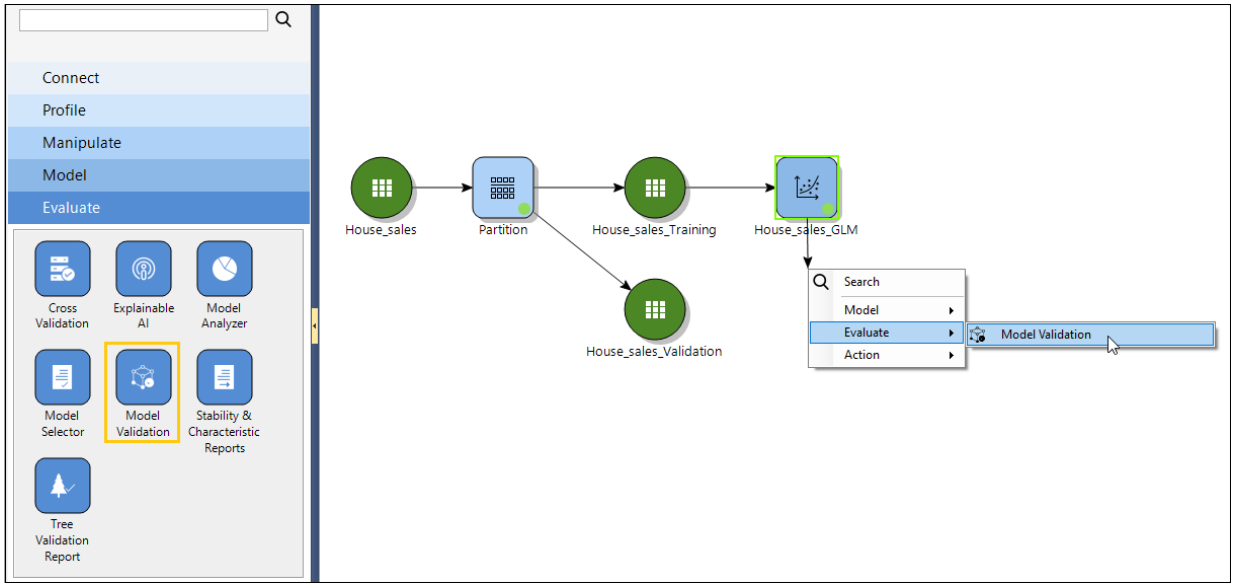
To add a Model Validation operation to your workflow:

- Drag the **Model Validation** node from the **Evaluate** palette onto the Workflow canvas and connect the inputs: an **Sklearn GLM** node and a dataset node that contains the data to validate.

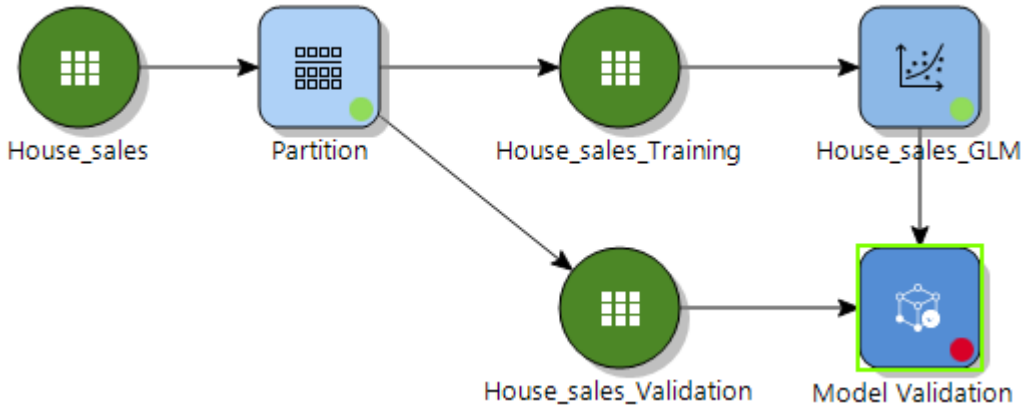
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

OR

- Right-click anywhere on the workflow canvas and select the context menu command **Evaluate --> Model Validation**. This will create a new Model Validation node. Connect the inputs: an **Sklearn GLM** model node and a dataset node that contains the data to validate.



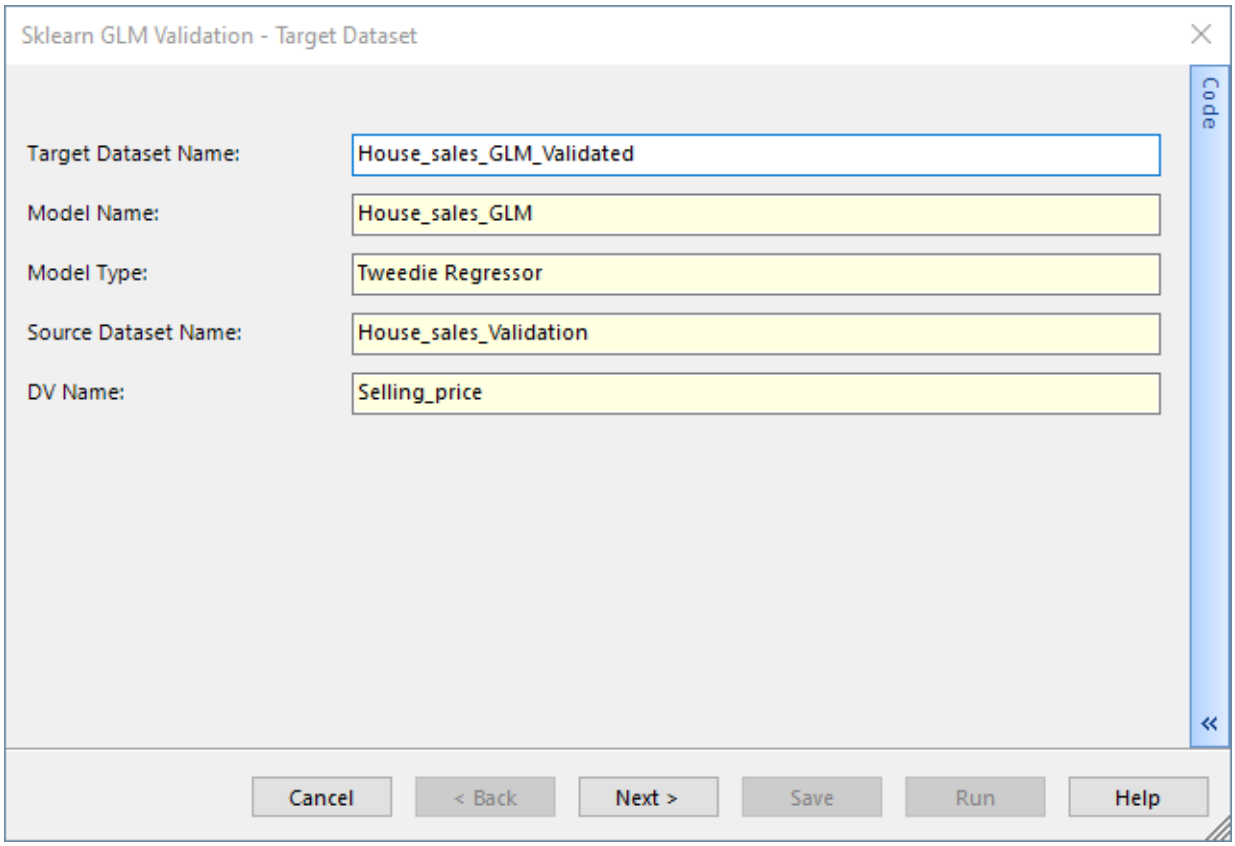
Double-click the **Model Validation** node or right-click on it and select **Modify**.



The Sklearn GLM Validation wizard opens. It displays the names of the input model and dataset, the regressor type, and the name of the dependent variable.

In the *Target Dataset Name* field, specify the name for the output dataset that will contain the validation information.

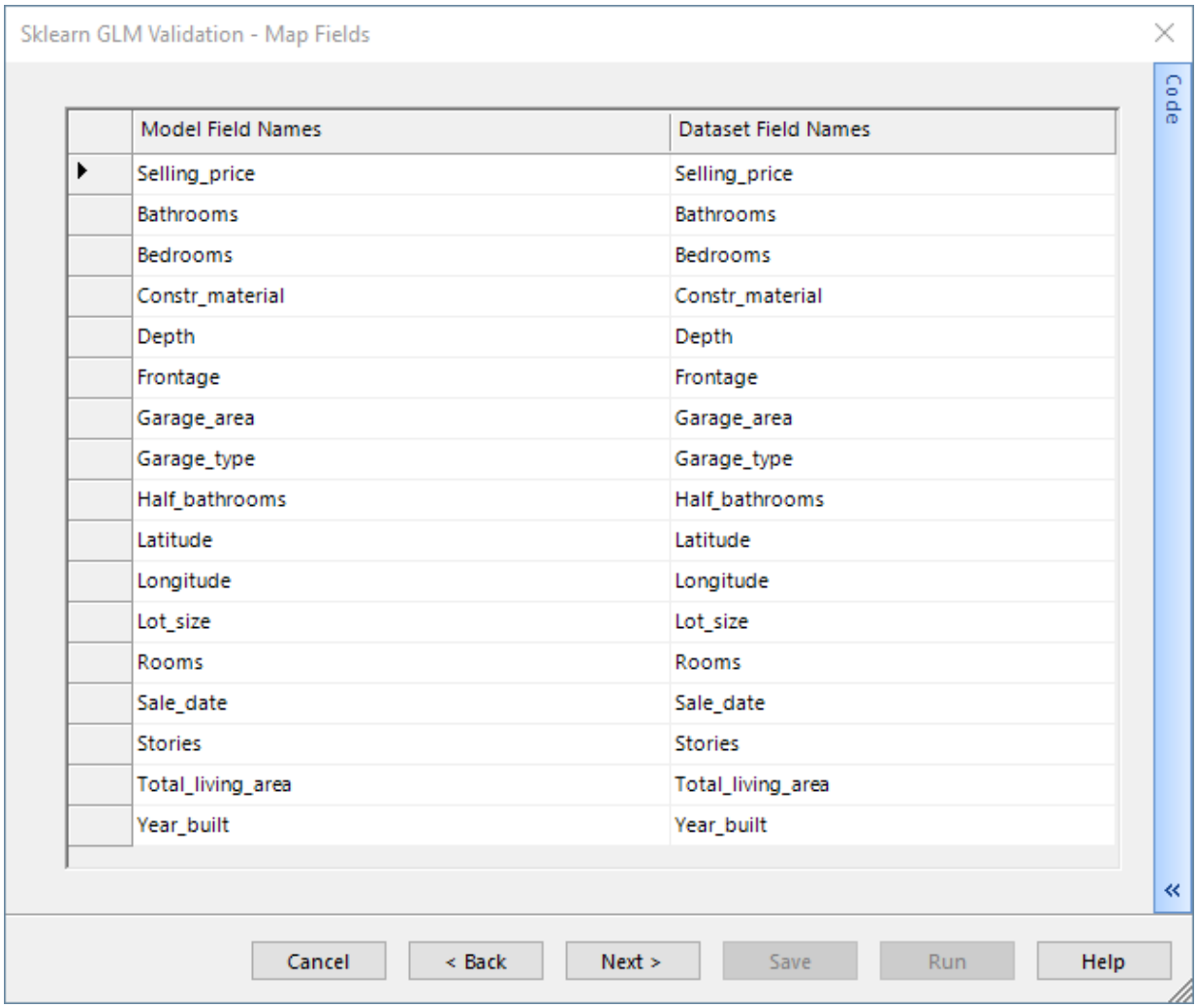
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

The next step provides the option of mapping the field names in the model to the ones in the validation input dataset. In some situations, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map fields by clicking in the right column (Dataset Field Names) and selecting the right fields to match the model variables in the left column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

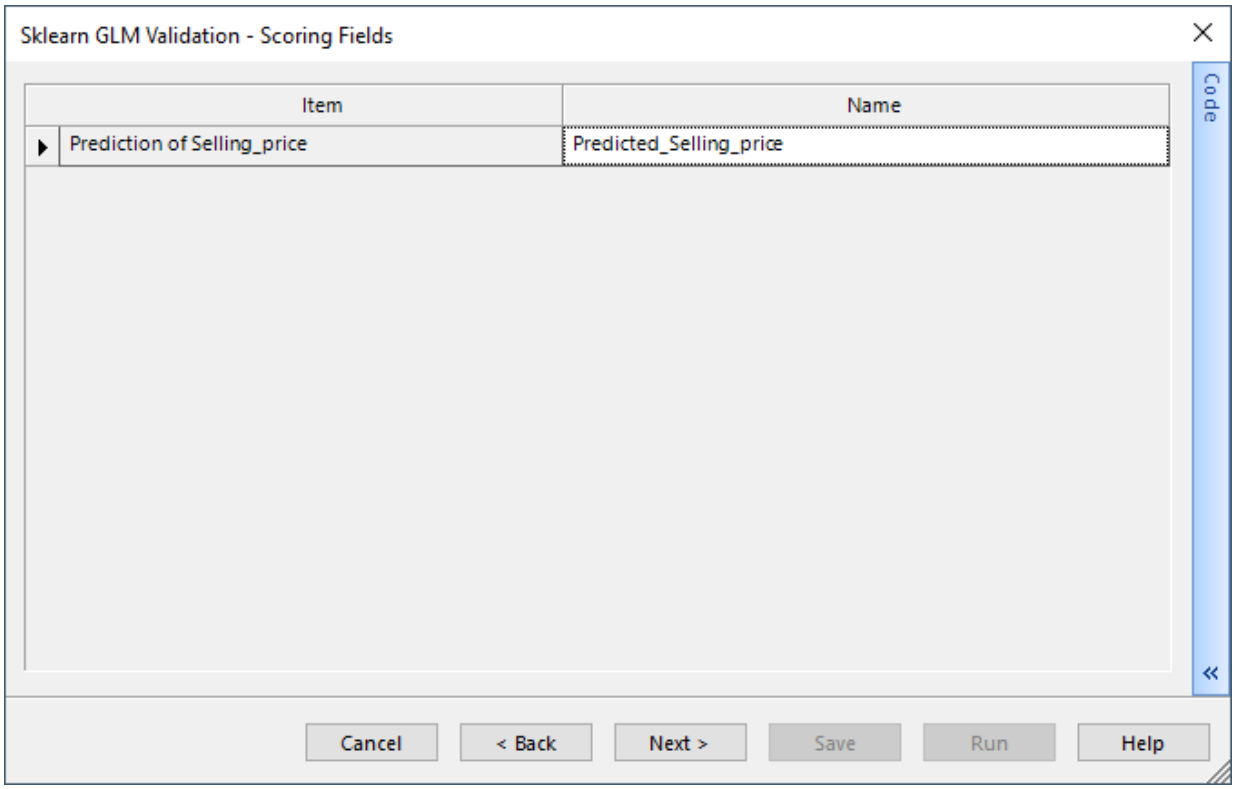


Note: When validating GLM models that use derived variables (also called calculated columns) with expressions, the input dataset must include these variables. This is different from the case of native Altair models with derived variables, which can be validated on a dataset with only *original* variables that the derived ones depend on, while the derived ones are not required.

Click **Next**.

In the **Scoring Fields** page, review the scoring fields to be produced by the model and modify their names if necessary.

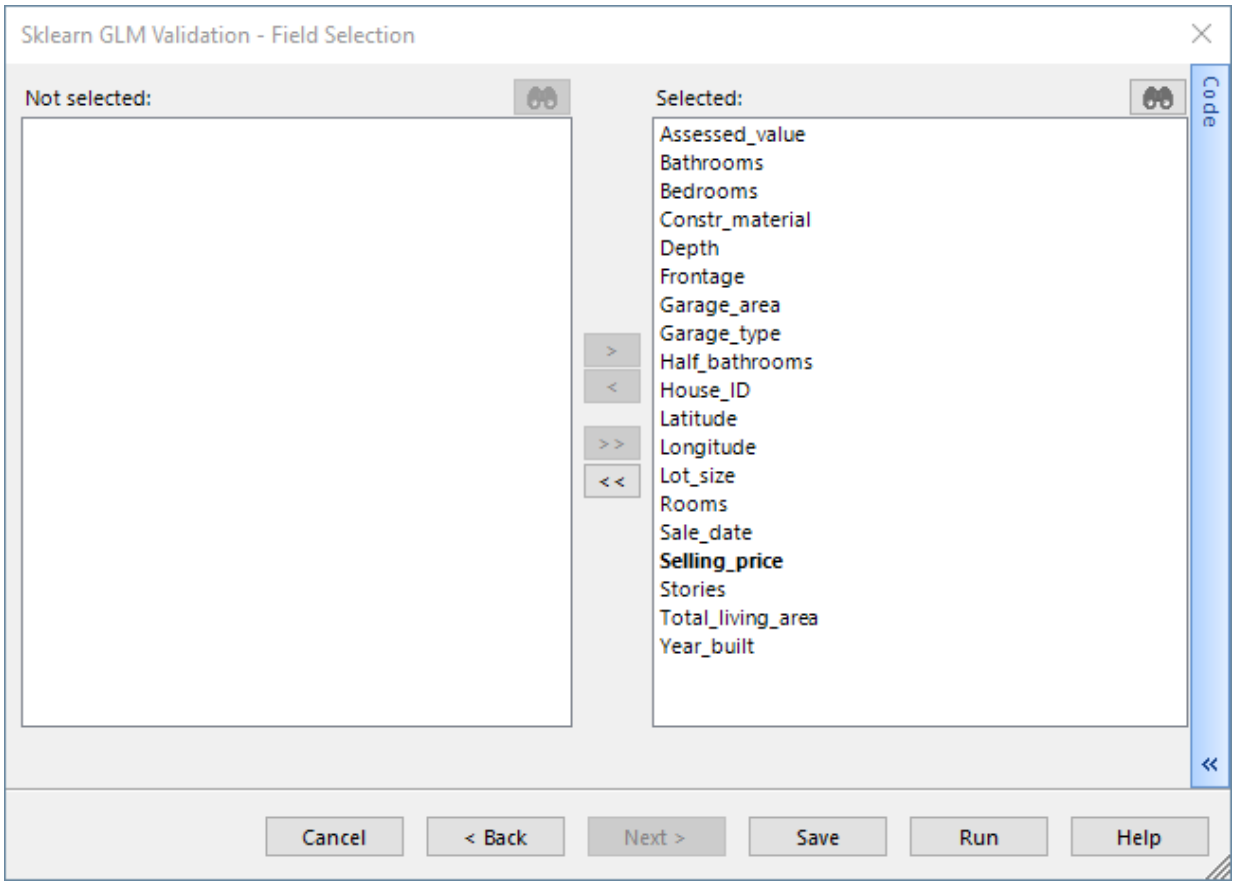
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

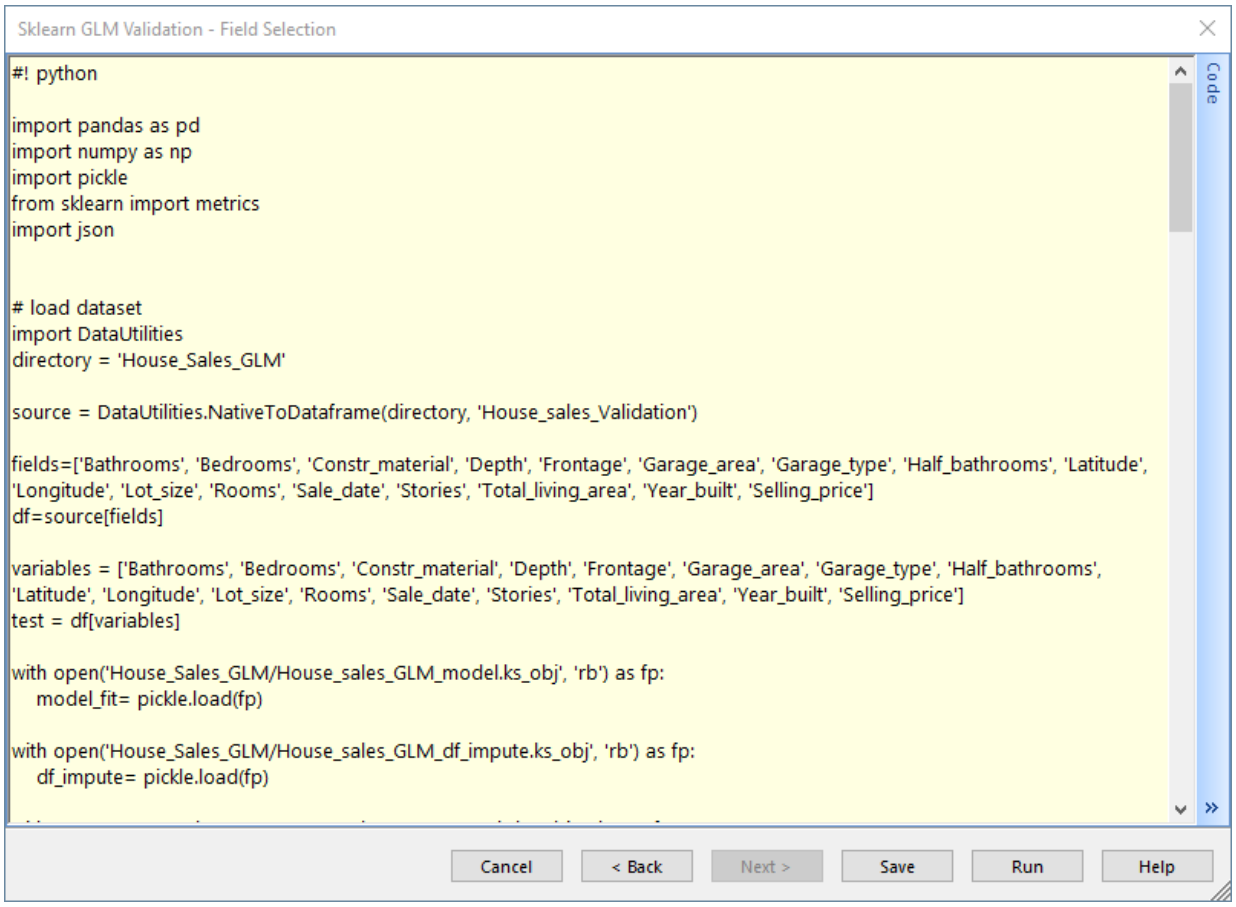
In the **Field Selection** page, choose the fields from the input dataset that you would like to include in the validation output in addition to the new score / probability fields. The fields required by the model are highlighted in bold font and cannot be excluded. By default, all fields in the validation input dataset are selected to be included in the output.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

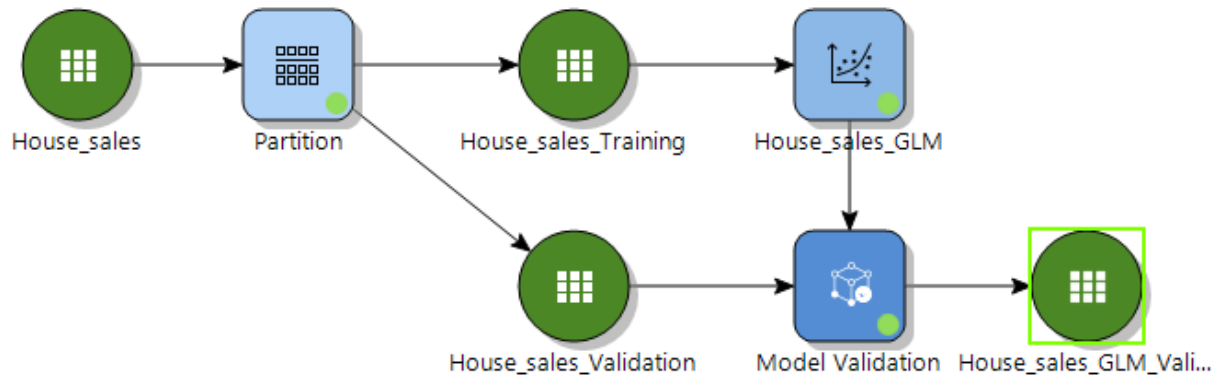


To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

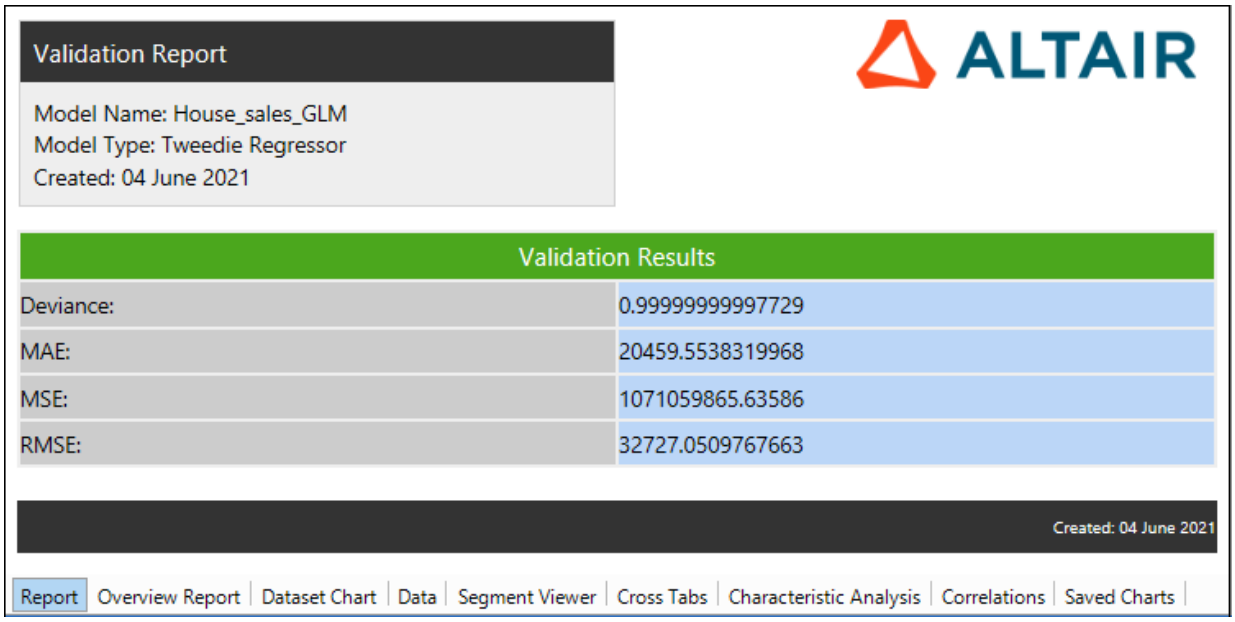


Click **Save** to save the settings, or click **Run** to save the settings and run the validation process immediately. The model is validated in a Python container session, and then the dataframe with generated probabilities / scores is converted back from Pandas to Altair KDD format. The output dataset node is created in the workflow. Double-click the output dataset to view the results.



The first tab of the output dataset is the Validation Report. It displays the values of the model evaluation metrics.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also:

Generalized Linear Model

Scoring a GLM Model

A Generalized Linear Model is scored using the **Scoring** node in the Action palette. For general information about scoring, see the topic About Scoring.

Prerequisites for Scoring: an **Sklearn GLM** model node and an input dataset to score.

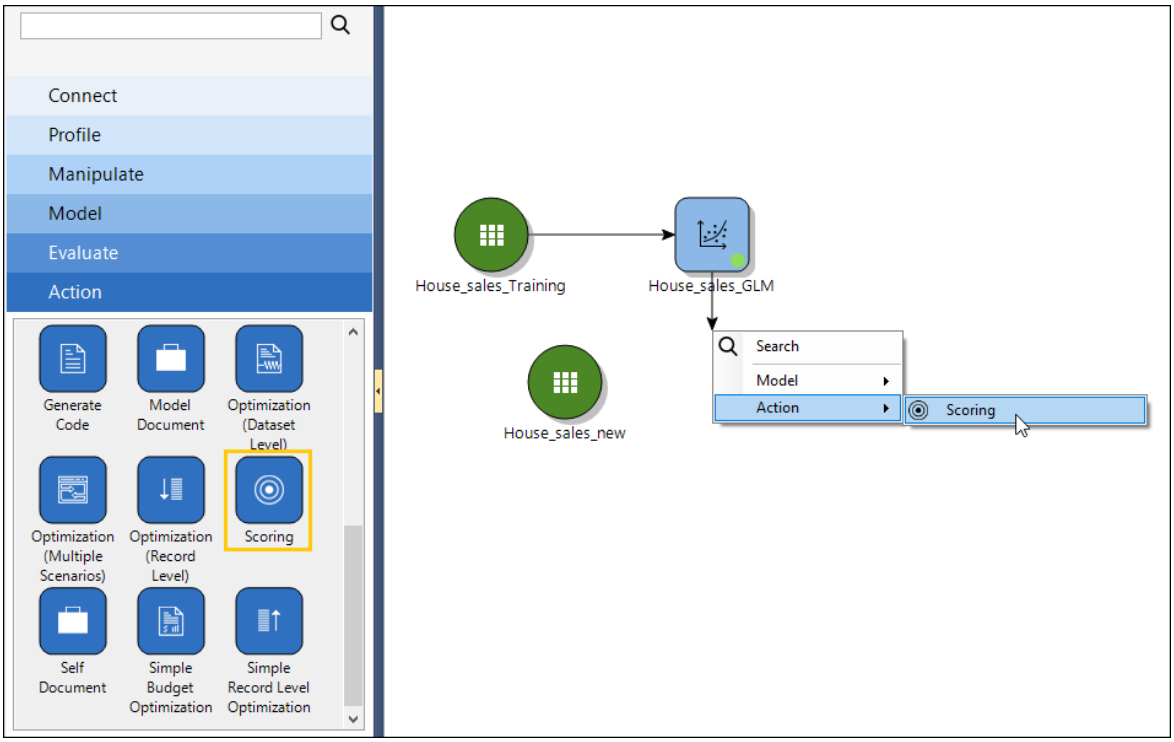
To add an Scoring operation to your workflow:

- Drag the **Scoring** node from the **Action** palette onto the Workflow canvas and connect the inputs: an **Sklearn GLM** model node and a dataset node that contains the data to score.

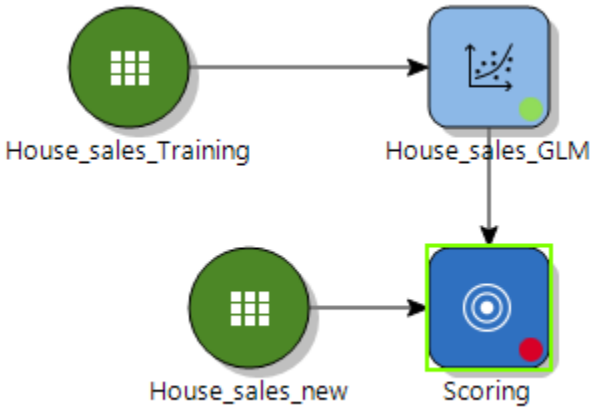
  OR

- Right-click anywhere on the workflow canvas and select the context menu command **Action --> Scoring**. This will create a new Scoring node. Connect the inputs: an **Sklearn GLM** node and a dataset node that contains the data to score.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



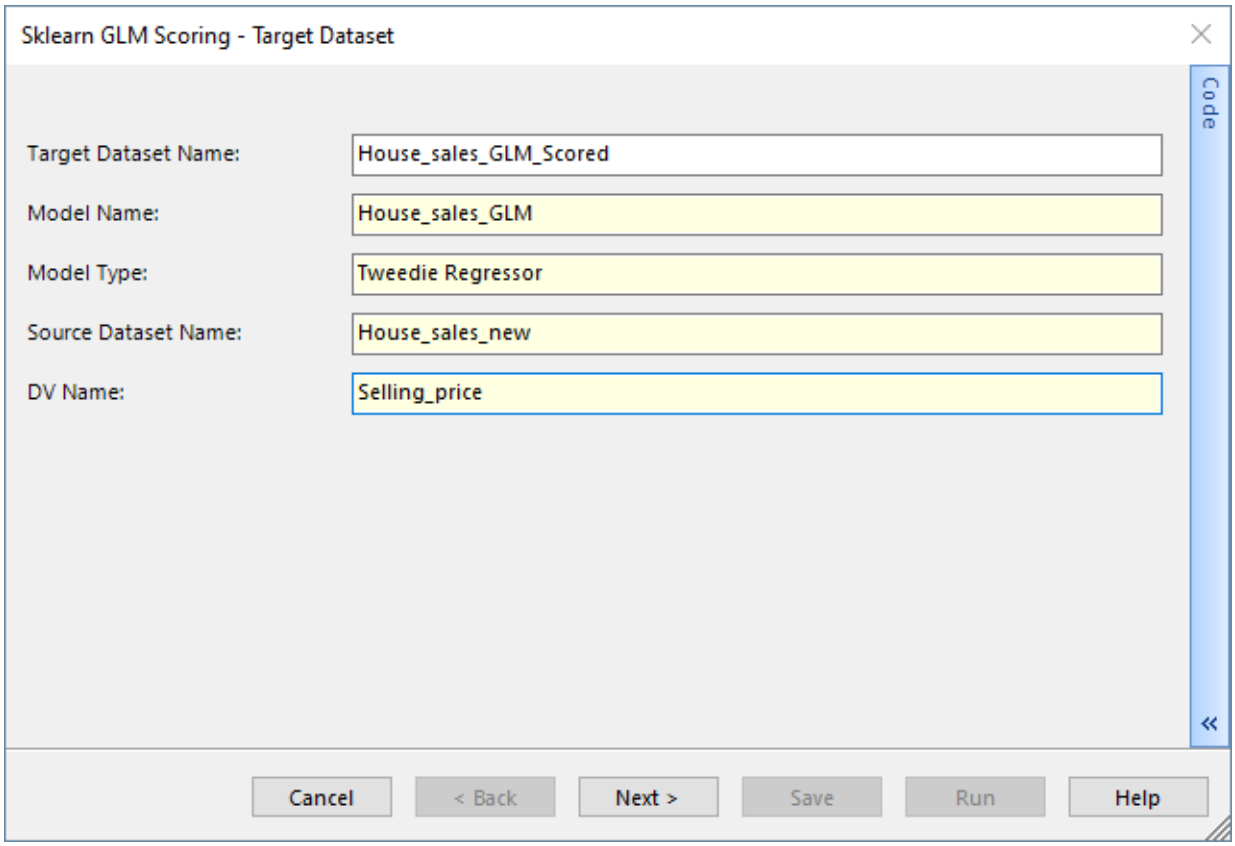
Double-click the **Scoring** node or right-click on it and select **Modify**.



The Scoring wizard opens. It displays the names of the input model and dataset, the regressor type, and the name of the dependent variable.

In the *Target Dataset Name* field, specify the name for the output dataset that will contain the scores.

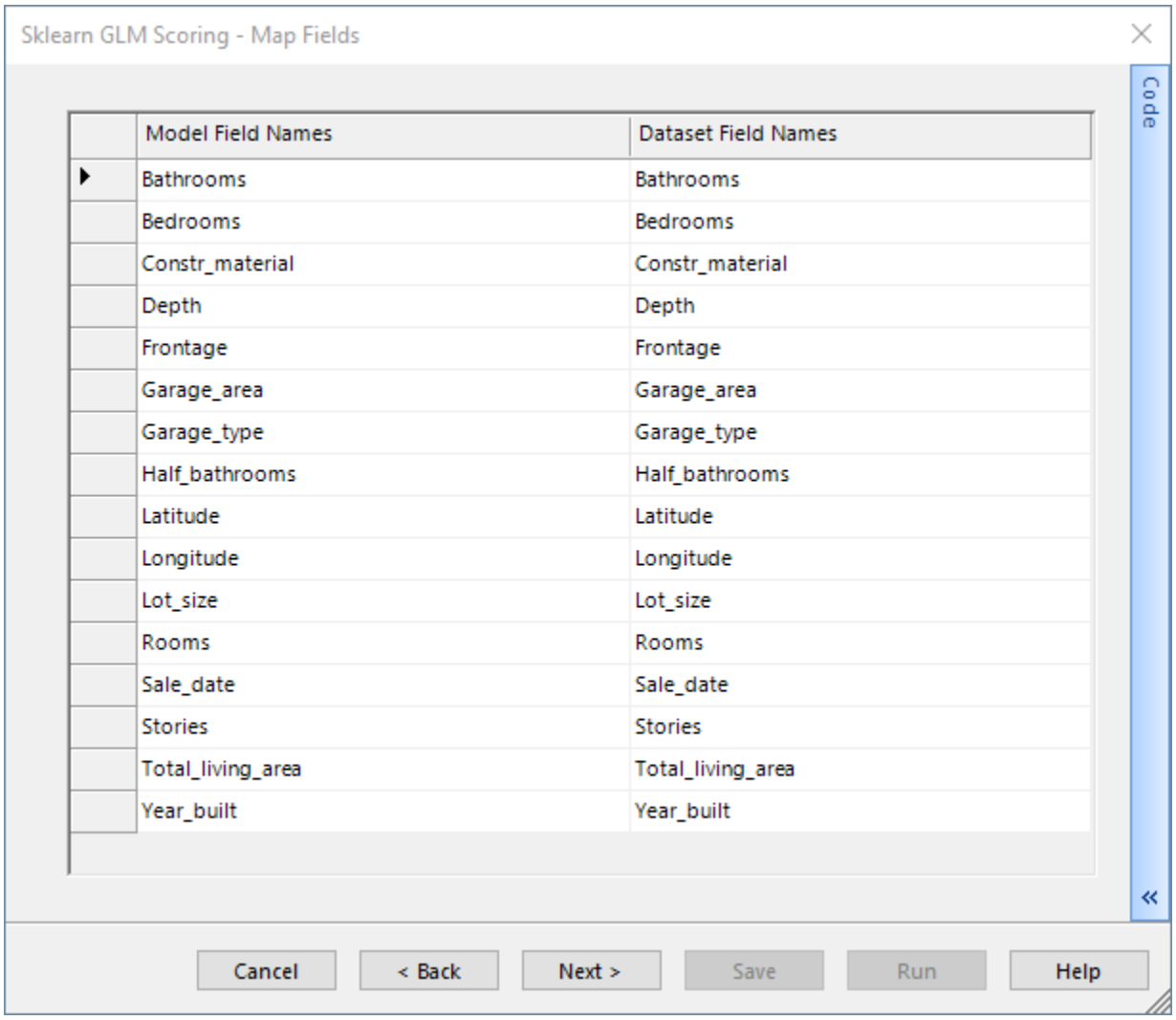
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

The next step provides the option of mapping the field names in the model to the ones in the scoring input dataset. In some situations, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map fields by clicking in the right column (Dataset Field Names) and selecting the right fields to match the model variables in the left column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

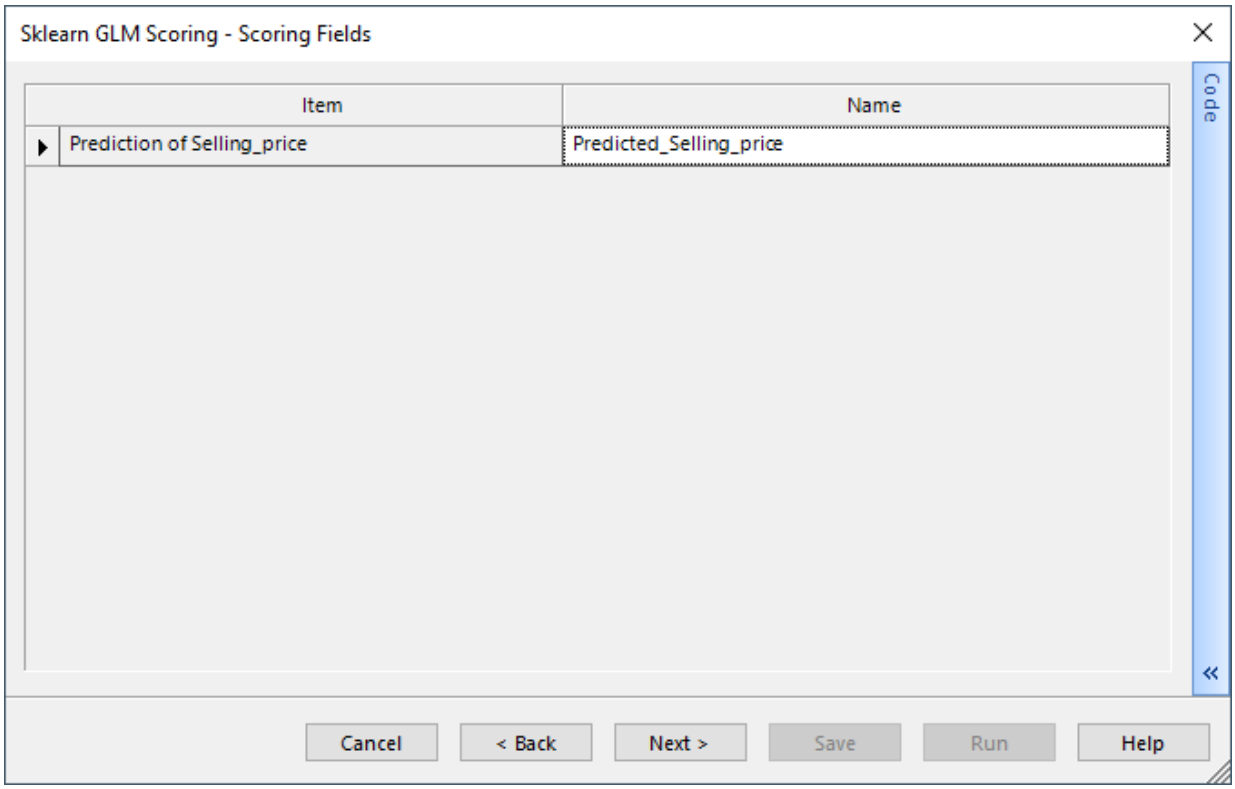


Note: When scoring GLM models that use derived variables (also called calculated columns) with expressions, the input dataset must include these variables. This is different from the case of native Altair models with derived variables, which can be scored on a dataset with only *original* variables that the derived ones depend on, while the derived ones are not required.

Click **Next**.

In the **Scoring Fields** page, review the scoring fields to be produced by the model and modify their names if necessary.

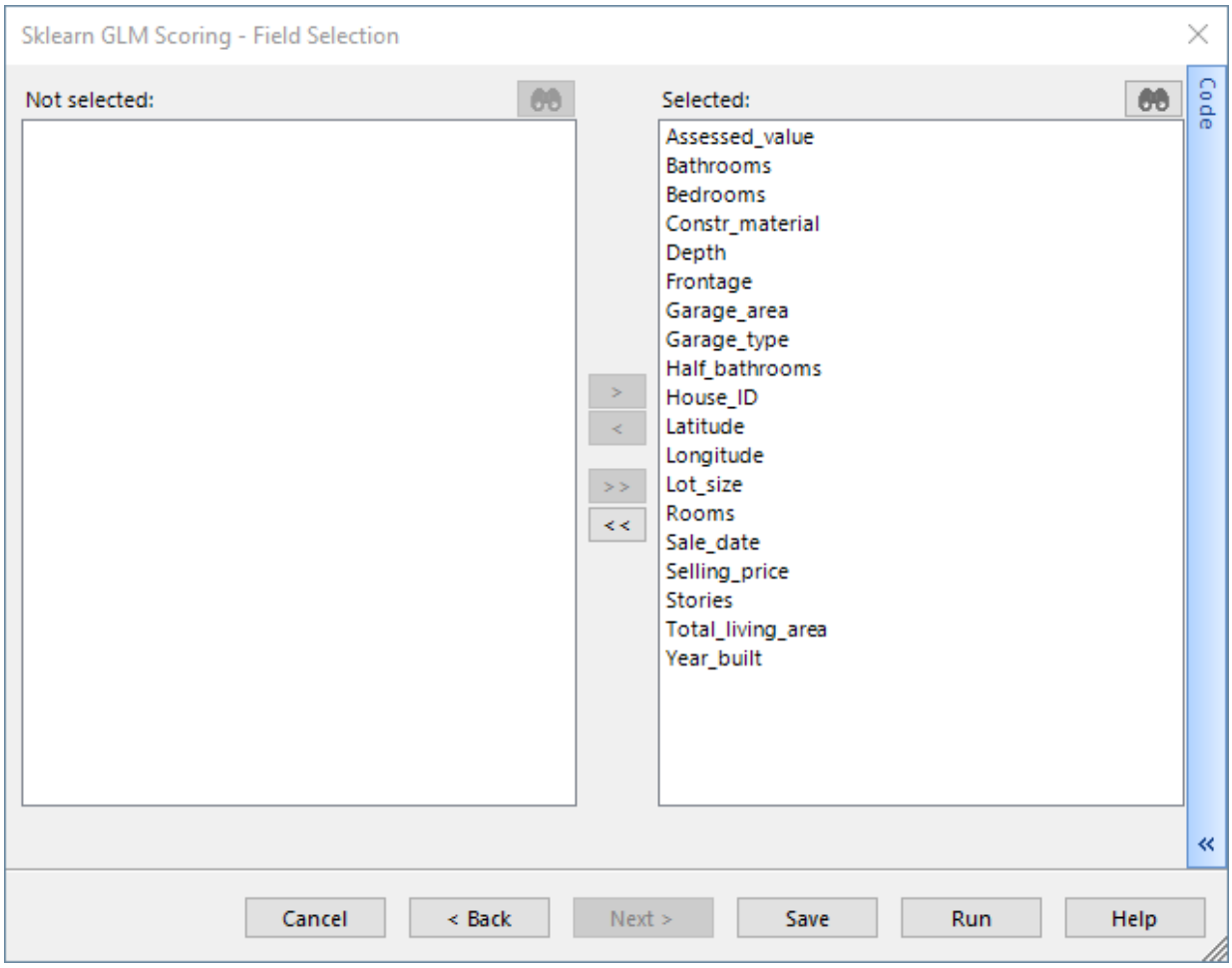
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

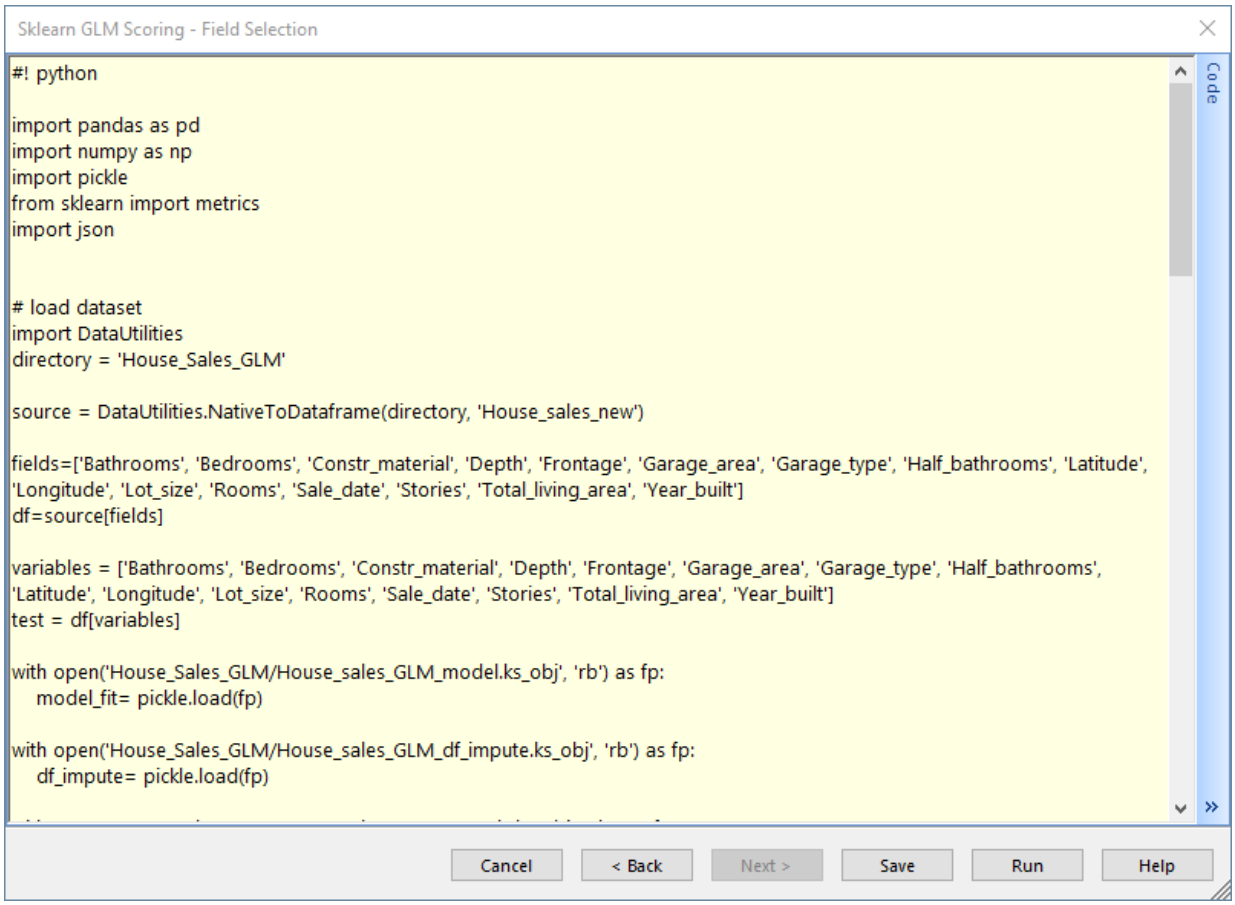
In the **Field Selection** page, choose the fields from the input dataset that you would like to include in the scoring output in addition to the new score / probability fields. By default, all fields in the scoring input dataset are selected to be included in the output.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

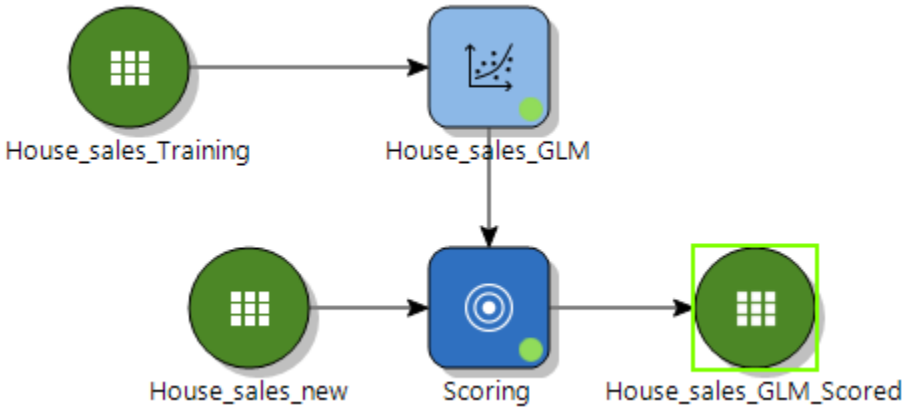


To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

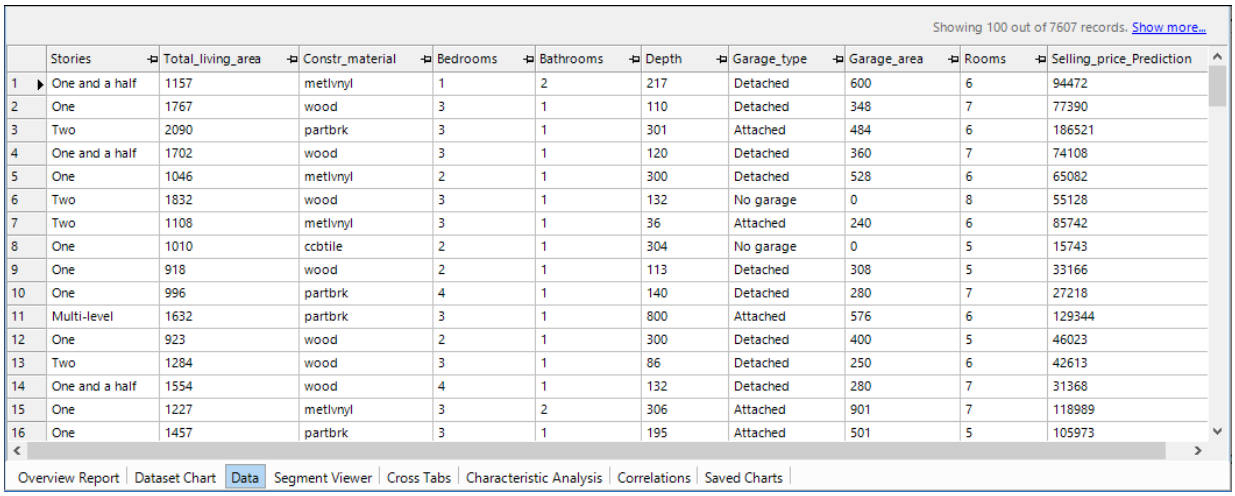


Click **Save** to save the settings, or click **Run** to save the settings and run the scoring process immediately. The model is scored in a Python container session, and then the dataframe with generated predictions (scores) is converted back from Pandas to Altair KDD format. The output dataset node is created in the workflow. Double-click the output dataset to view the results.



Click the Data tab of the output dataset to see the model predictions in the last column.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Showing 100 out of 7607 records. Show more...

	Stories	Total_living_area	Constr_material	Bedrooms	Bathrooms	Depth	Garage_type	Garage_area	Rooms	Selling_price_Prediction
1	One and a half	1157	metlvnyl	1	2	217	Detached	600	6	94472
2	One	1767	wood	3	1	110	Detached	348	7	77390
3	Two	2090	partbrk	3	1	301	Attached	484	6	186521
4	One and a half	1702	wood	3	1	120	Detached	360	7	74108
5	One	1046	metlvnyl	2	1	300	Detached	528	6	65082
6	Two	1832	wood	3	1	132	No garage	0	8	55128
7	Two	1108	metlvnyl	3	1	36	Attached	240	6	85742
8	One	1010	ccbtile	2	1	304	No garage	0	5	15743
9	One	918	wood	2	1	113	Detached	308	5	33166
10	One	996	partbrk	4	1	140	Detached	280	7	27218
11	Multi-level	1632	partbrk	3	1	800	Attached	576	6	129344
12	One	923	wood	2	1	300	Detached	400	5	46023
13	Two	1284	wood	3	1	86	Detached	250	6	42613
14	One and a half	1554	wood	4	1	132	Detached	280	7	31368
15	One	1227	metlvnyl	3	2	306	Attached	901	7	118989
16	One	1457	partbrk	3	1	195	Attached	501	5	105973

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

See also:

- Generalized Linear Model
- Sklearn GLM Validation

Novelty & Outlier Detector

Overview

The **Sklearn Novelty & Outlier Detector** node in the **Model** palette of Knowledge Studio helps determine whether a set of observations belong to the same distribution as the other observations in a dataset or should be classified as outliers. This is useful for data cleansing to prepare data for modeling. An observation is considered to be an outlier if its distance (in terms of a specific metric) from the other observations in a dataset exceeds a certain threshold. Outlier detection estimators try to fit the regions where the data in the training dataset is the most concentrated, ignoring the deviant observations.

When the training data is assumed to be not polluted by outliers, the purpose is to detect whether a **new** observation is an outlier. In this context, an outlier is also called a ***novelty***.

The **Sklearn Novelty & Outlier Detector** node requires a single input dataset and produces one or two output datasets (depending on the method used) with columns containing anomaly metrics that determine the outlier status. In the cases where two outputs are produced, he first output dataset contains all original records, while the second one contains only non-outliers.

This feature works through integration with Python libraries. It uses functions of **`scikit-learn`** - a popular Python library for machine learning. For details, see the scikit-learn documentation for Isolation Forest, Local Outlier Factor, and One Class SVM algorithms (Copyright © 2007 - 2021, scikit-learn developers, BSD License). The node wizard provides a convenient graphical interface to specify the options and parameters for the required `sklearn` functions. The wizard automatically translates the inputs into Python code, which can be previewed before running the node. The code then runs in a container with the necessary Python packages and their dependencies. The input dataset is converted from the Altair KDD format to pandas dataframes in the process. The resulting dataframe is converted back to produce the output Altair dataset.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

After training, the model can be applied to the observations in a new dataset using the Sklearn Novelty & Outlier Scoring node.

Inserting a Novelty & Outlier Detector node

An **Sklearn Novelty & Outlier Detector** model requires a single dataset as input and can be added to the workflow in one of the following ways:

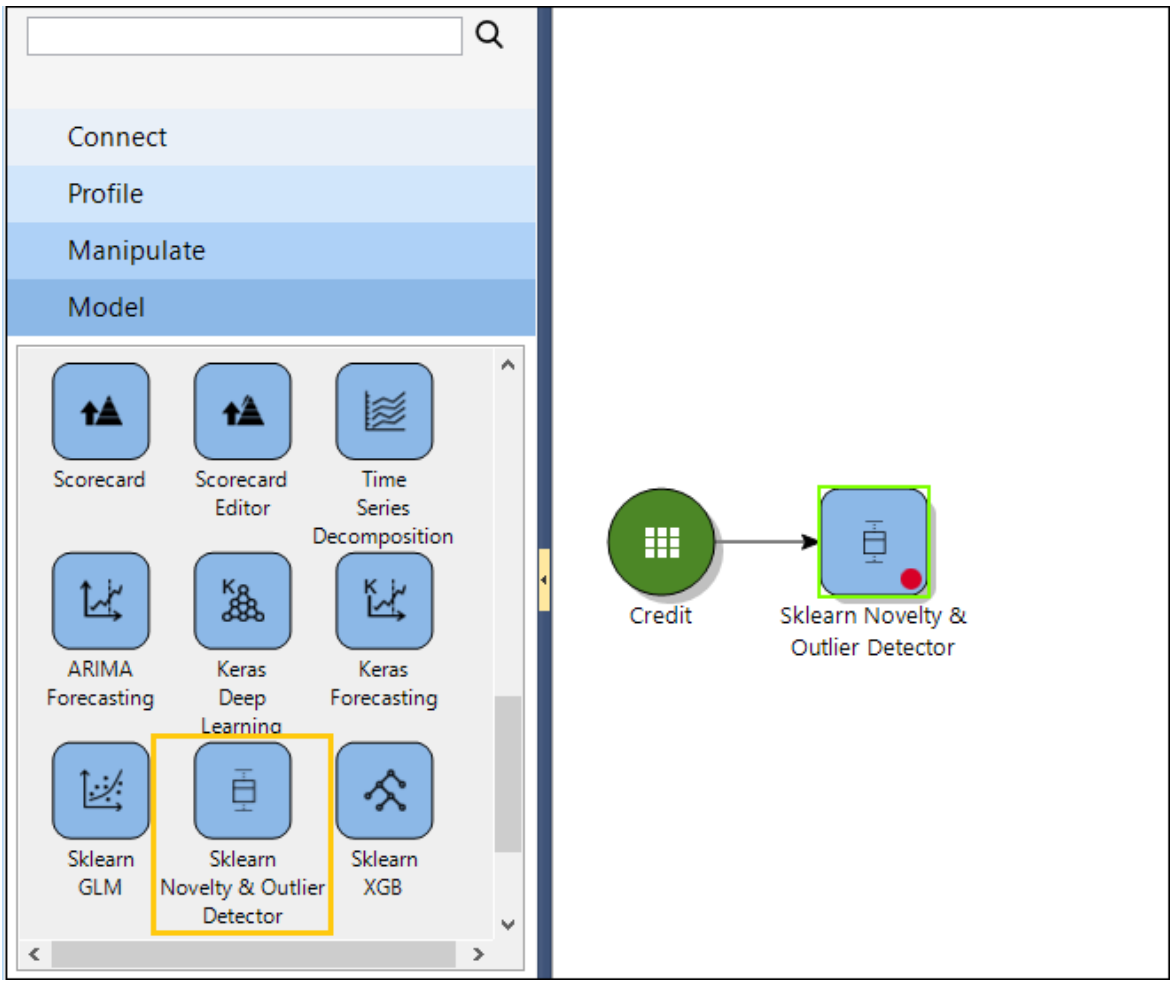
- Drag a connection arrow from the input dataset node to any area on the Workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Model --> Sklearn Novelty & Outlier Detector**. A new **Sklearn Novelty & Outlier Detector** node will be created, and the wizard will open.

OR

- Right-click on the Workflow canvas and select **Model --> Sklearn Novelty & Outlier Detector** from the context menu. Connect the training dataset.

OR

- Open the **Model** palette and drag the **Sklearn Novelty & Outlier Detector** node onto the workflow canvas. Connect the input dataset to the new node.



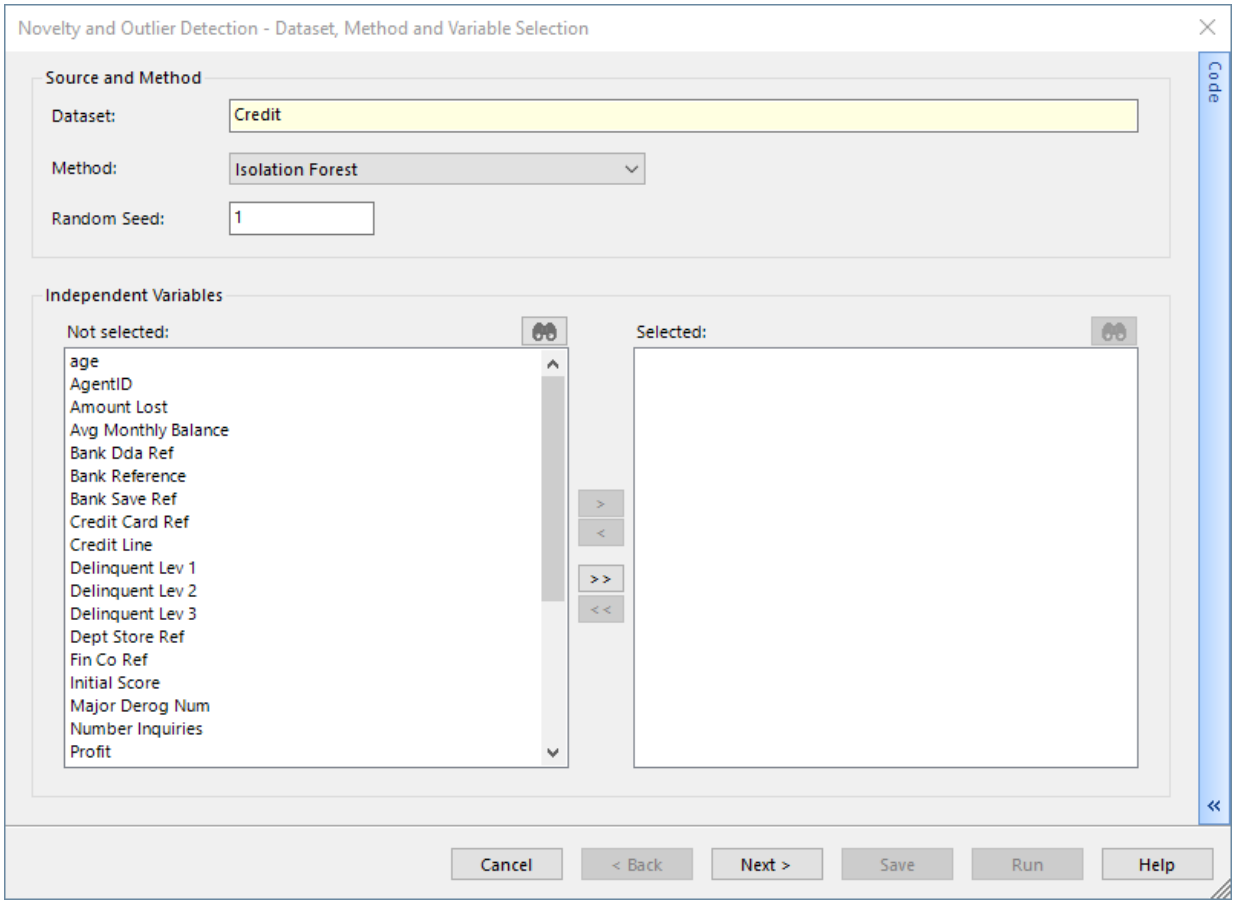
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If the **Sklearn Novelty & Outlier Detector** node is already on the canvas, double-click it or right-click on it and select **Modify** from the context menu to open the wizard.

In the first page of the wizard:

- Specify the method for novelty & outlier detection. The available methods are *Isolation Forest*, *Local Outlier Factor*, and *One Class SVM*.
- Specify the Random Seed value for the repeatability of results, if necessary.
- Select the desired independent variables to analyze for outliers.

Tip: Advanced field finding and multi-selection capabilities are also available via the Field Finder dialog, which can be opened by clicking the binoculars button.



Click **Next**.

The options in the next page depend on the chosen method.

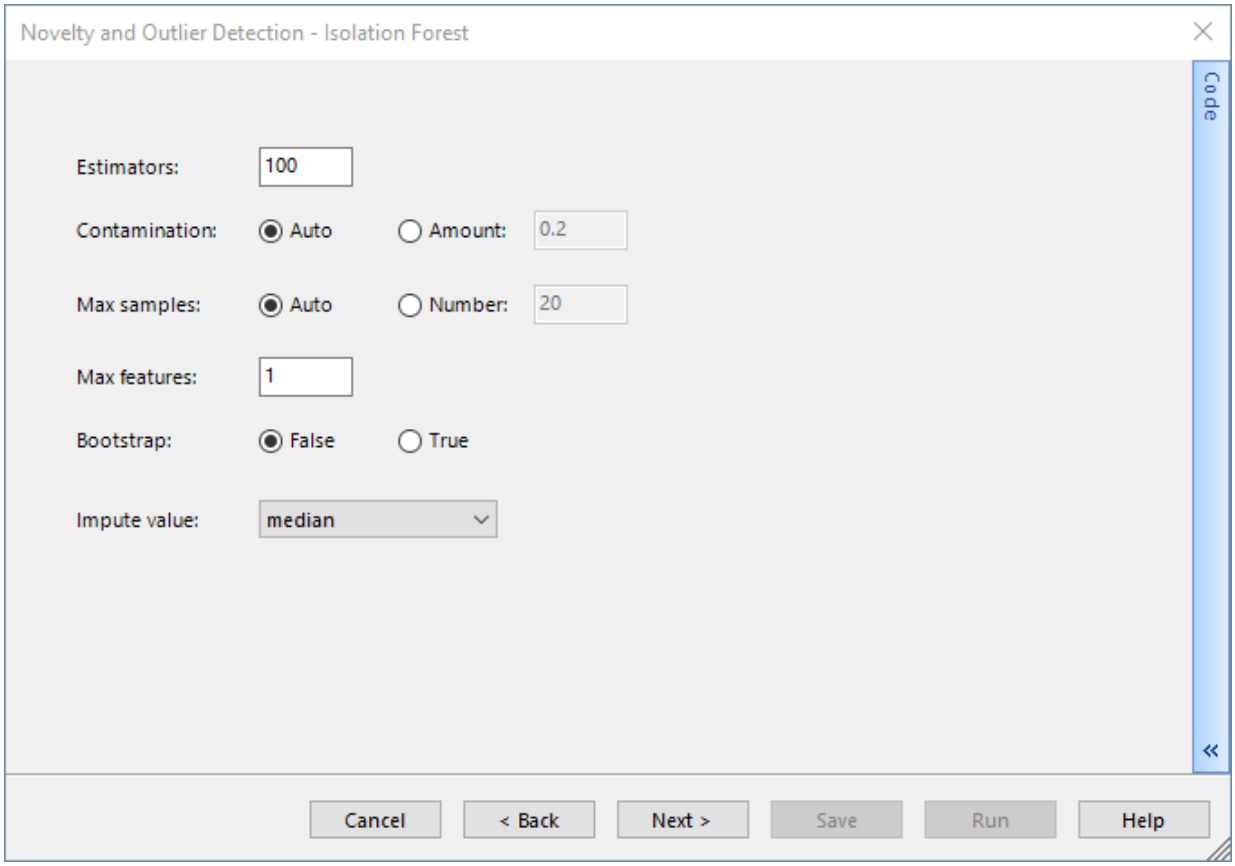
For the ***Isolation Forest*** method, specify the following options and parameters. For details, see the description of the Isolation Forest Algorithm parameters in the Scikit-learn documentation.

- Specify the number of *estimators* in the ensemble. The default is 100.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Specify the amount of *contamination* (the proportion of outliers) in the dataset: choose Auto (automatic) or specify the desired amount.
- Select the *maximum number of samples*: choose Automatic (Auto) or specify the desired number.

- Specify the *maximum number of features* to draw to train each base estimator. The default is 1.
- Set the *Bootstrap* option to True if you need sampling with replacement, otherwise set it to False.
- Select the *imputation strategy* for preprocessing. The available imputation options are: *median*, *mean*, and *mode*. The default value is *median*.

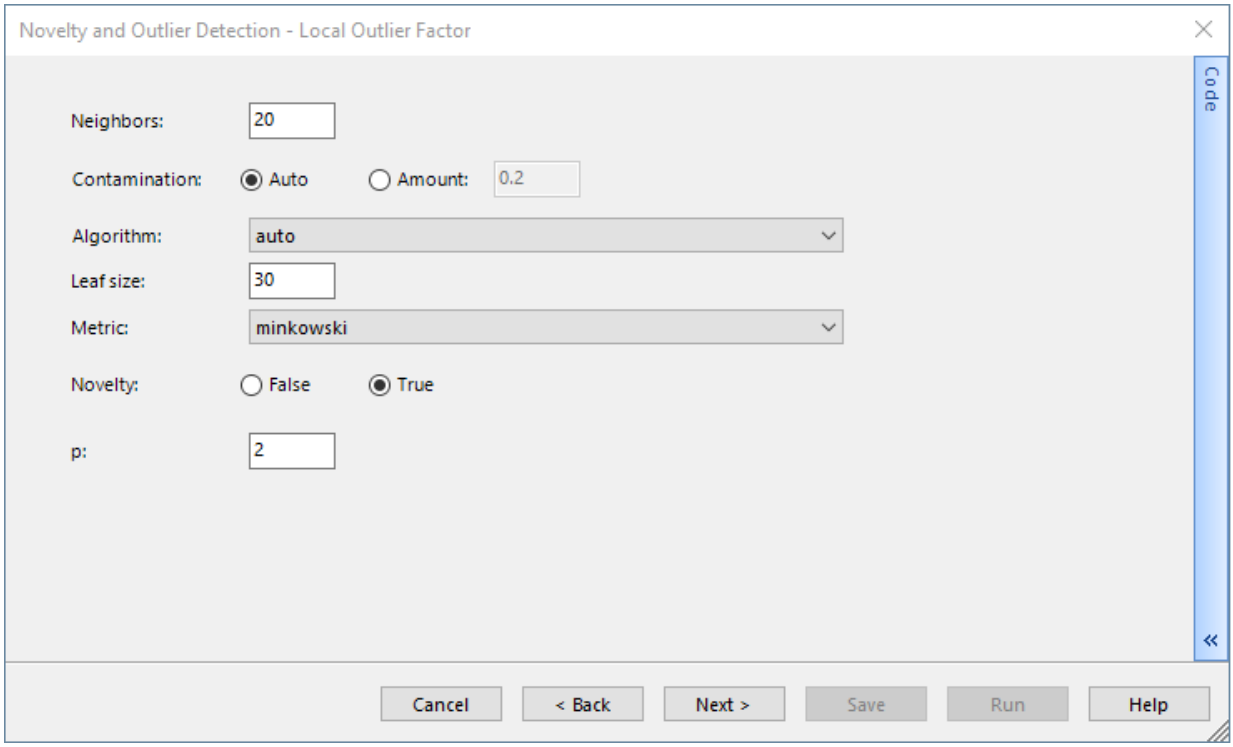


For the ***Local Outlier Factor*** method, specify the following options and parameters. For details, see the description of the Local Outlier Factor Algorithm parameters in the Scikit-learn documentation.

- Specify the number of *neighbors* for samples. The default is 20.
- Specify the amount of *contamination* (the proportion of outliers) in the dataset: choose Auto (automatic) or specify the desired amount.
- Choose the *algorithm* used to compute the nearest neighbors. The available options are *ball_tree*, *kd_tree*, *brute* (brute-force search), and *auto.* The default is *auto.*
- Specify the *leaf size* passed to ball_tree or kd_tree. The default is 30.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

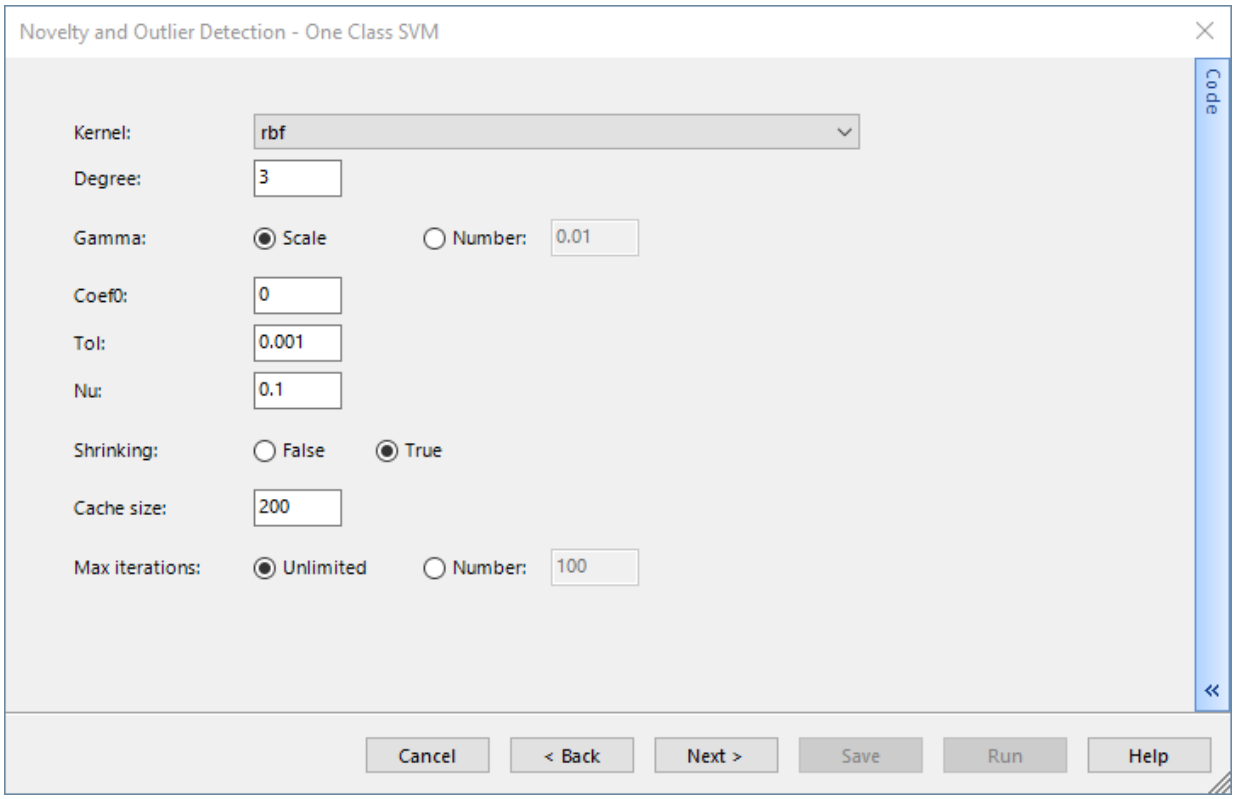
- Choose the m*etric* used to compute the distance. The available metrics are: *minkowski*, *euclidean*, and *manhattan.* The default is *minkowski.*
- Set the *Novelty* option to True if you want to use the method for novelty detection. In this case it is assumed that the training data is not polluted by outliers, and you are interested in detecting whether a new observation is an outlier..
- Specify the value of the parameter *p* for the Minkowski metric from sklearn.metrics.pairwise.pairwise_distances. The default is 2.



For the ***One Class SVM*** method, specify the following options and parameters. For details, see the description of the One Class SVM algorithm parameters in the Scikit-learn documentation.

- Select the *kernel type*. The available options are *rbf*, *linear*, *poly*, *sigmoid*, and *precomputed*. The default is *rbf*.
    - Specify the *degree* of the polynomial kernel function. The default is 3.
- Specify the *Gamma* coefficient for rbf, poly or sigmoid kernel types. 'Scale' is selected by default.
- Specify *Coef0* - the Independent term in the kernel function (for ‘poly’ or ‘sigmoid’ kernel). The default is 0.
- Specify the value of tolerance *tol* for the stopping criterion. The default is 0.001.
- Specify the value of *Nu* - the upper bound on the fraction of training errors and a lower bound of the fraction of support vectors. The default is 0.1
- For the *shrinking* option, select True to use the shrinking heuristic. Otherwise, select False.
- Specify the kernel *cache size* (in MB). The default is 200 MB.
- Specify the *maximum number of iterations* within the solver. The default is *Unlimited*.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

- Select the original fields to be included in the output dataset(s).
- Specify which output datasets are to be created. The output depends on the detection method.

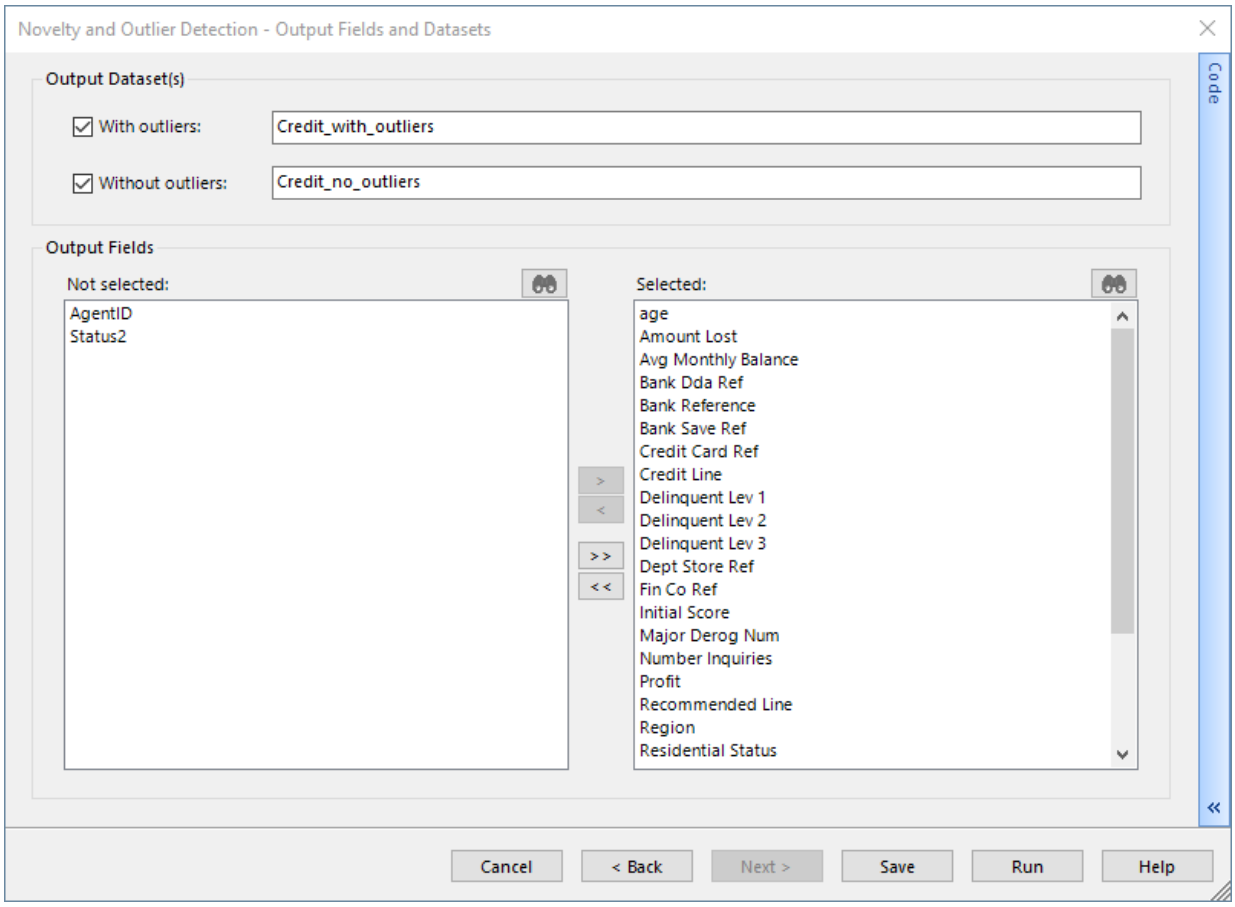
    - For the Isolation Forest method, two output datasets are available:
        - The dataset with outliers contains all records of the original dataset, the ones classified as outliers being flagged according to their computed *Anomaly_Score* and *Oposite_Anomaly_Score,* shown in two additional fields.
            - The *Anomaly_Score* column contains the computed anomaly score values for each record, followed by the *Opposite_Anomaly_Score* column. The *Anomaly_Flag* column displays -1 for outliers and 1 for non-outliers.
        - The second dataset contains only non-outlier records. It includes the *Anomaly_Score* and *Oposite_Anomaly_Score* columns.

    - For the *One Class SVM* method, two output datasets are available:
        - The dataset with outliers contains all records of the original dataset and three additional columns.
            - The *Anomaly_Flag* column displays -1 for outliers and 1 for non-outliers.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

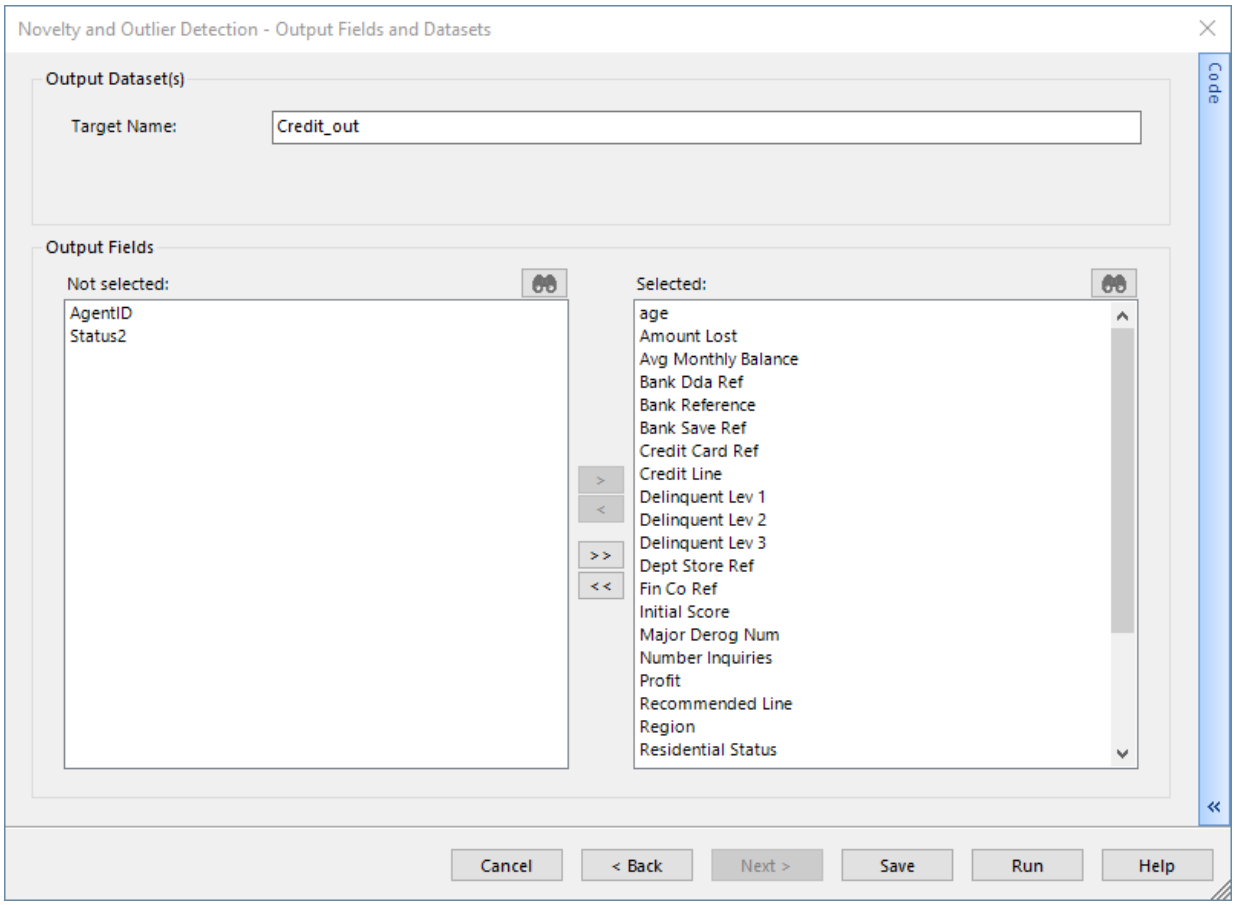
- The *Separating_hyperplane_distance* contains the computed score values for each record.
- The *Scoring_Function* column.

- The second dataset contains only non-outlier records. It includes two additional columns: *Separating_hyperplane_distance* and *Scoring_Function*.

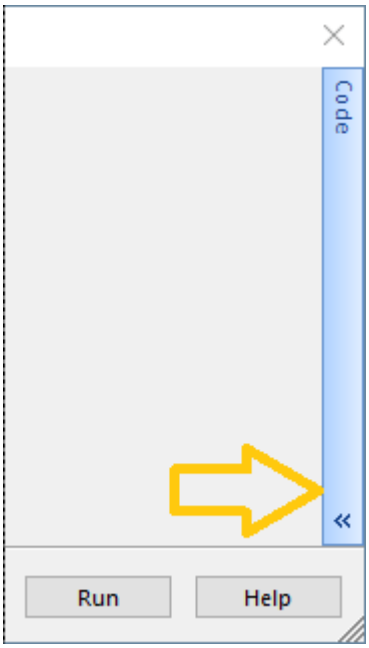


- For the Local Outlier Factor method, there is a single output dataset.
  - If the *Novelty* option was set to True, the output dataset will have three new columns: the *Anomaly_Score* column contains the computed anomaly score values for each record, followed by the *Opposite_Anomaly_Score* column. The *Anomaly_Flag* column displays -1 for anomaly and 1 for not an anomaly.
  - If the *Novelty* option was set to False, the output dataset has one new column: *Negative Outlier Factor*.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

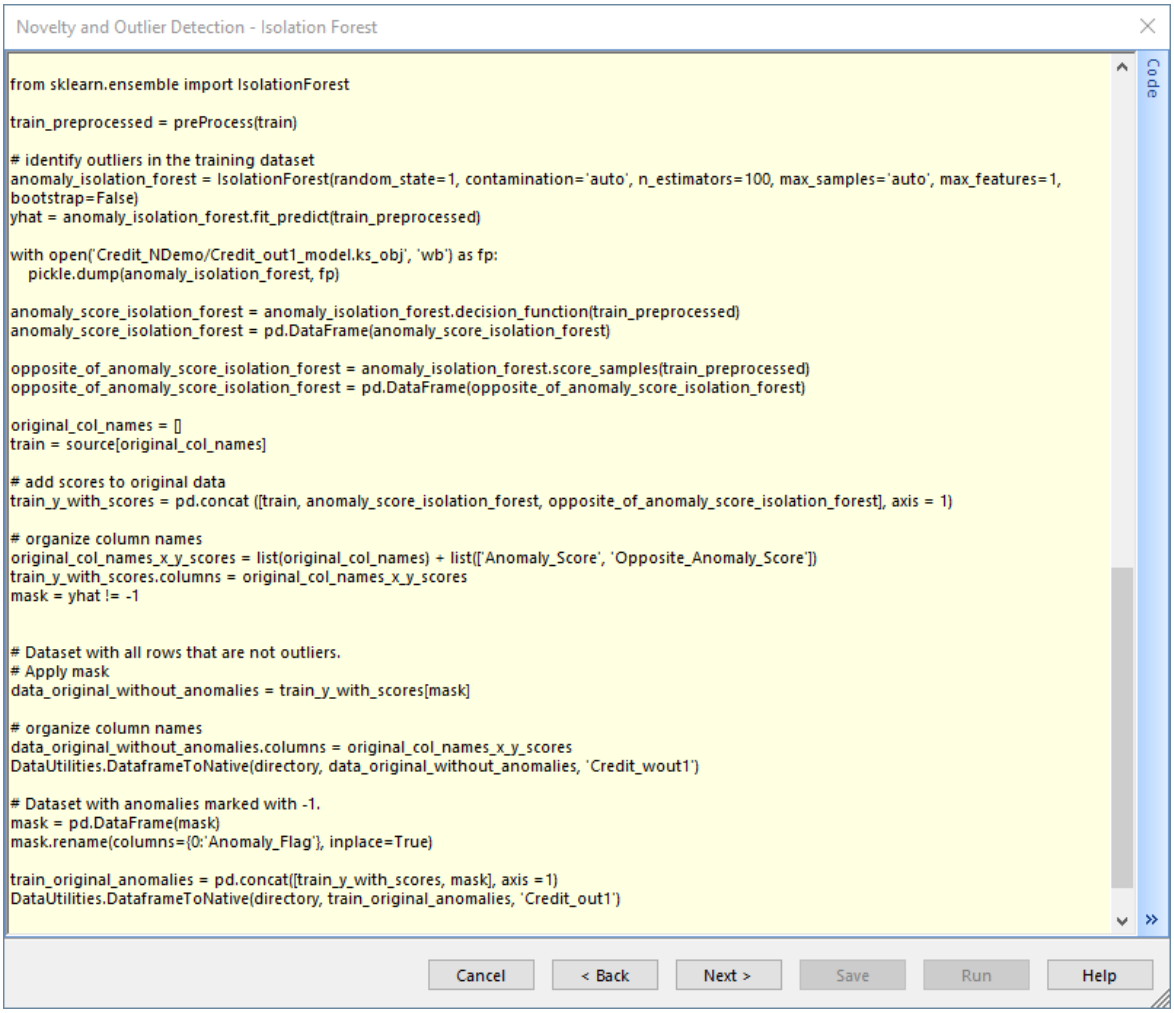


To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

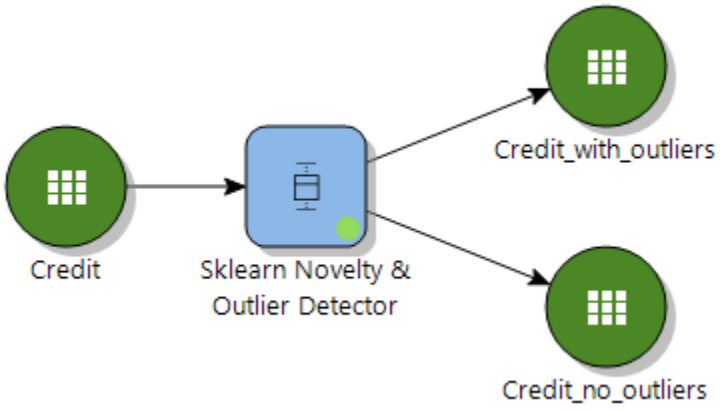
This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key. Click on the blue Code bar again to close the code preview panel.



Click **Save** to save the settingsor click Run to save the settings and run the process immediately. The code generated from your inputs runs in a Python container session. The input dataset is converted to a Pandas dataframe, and the required functions of the `sklearn` library are applied. The output data is converted from Pandas to the Altair KDD format and is shown as the output dataset node at the end.

Double-click the output dataset node(s) to view the results.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click the Data tab to view the new columns added.

Example 1: Output dataset with outliers (Isolation Forest method). The *Anomaly_Flag* field is the outlier indicator based on the values of *Anomaly_Score* and *Opposite_Anomaly_Score*.

	Total Service	Worst Rating Des	Year at Address	Year Last Check	Anomaly_Score	Opposite_Anomaly_Score	Anomaly_Flag
1	1	3 satisfactory ratings	10	1	0.000867930621737345	-0.499132069378263	☑
2	2	(null)	(null)	(null)	0.0313802700330305	-0.468619729966969	☑
3	3	1 satisfactory rating	23	1	0.0132539822796954	-0.486746017720305	☑
4	6	(null)	(null)	(null)	0.0255671819827542	-0.474432818017246	☑
5	0	(null)	(null)	(null)	0.0273426656560493	-0.472657334343951	☑
6	111	5 or more satisfactory ratings	(null)	0	-0.0243173989998243	-0.524317398999824	☐
7	284	(null)	10	(null)	0.00673247165971619	-0.493267528340284	☑
8	0	(null)	(null)	(null)	0.0292755414470232	-0.470724458552977	☑
9	47	File but no rating	1	0	-0.00098775065024248	-0.500987750650242	☐
10	38	File but no rating	3	98	-0.0157806793598609	-0.515780679359861	☐
11	0	2 satisfactory ratings	(null)	1	0.00701388289900789	-0.492986117100992	☑
12	115	2 satisfactory ratings	1	0	0.0154828208525022	-0.484517179147498	☑
13	157	No File	(null)	99	0.00136647916434529	-0.498633520835655	☑
14	0	(null)	(null)	(null)	0.0274657975381126	-0.472534202461887	☑
15	0	(null)	(null)	(null)	0.0246972324736272	-0.475302767526373	☑

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

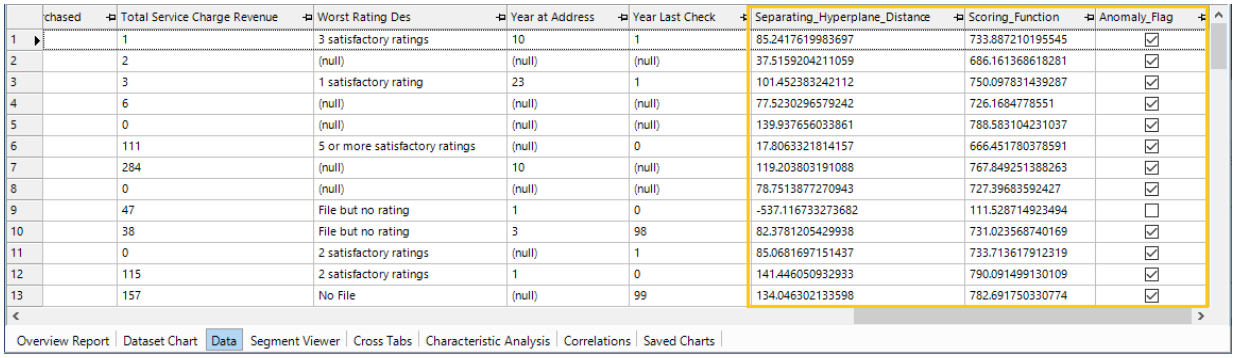
Example 2: Output dataset without outliers (Isolation Forest method). The records classified as outliers are removed. The fields *Anomaly_Score* and *Opposite_Anomaly_Score* show the anomaly metrics for each record.

	Total Service Charge Revenue	Worst Rating Des	Year at Address	Year Last Check	Anomaly_Score	Opposite_Anomaly_Score
1	1	3 satisfactory ratings	10	1	0.000867930621737345	-0.499132069378263
2	2	(null)	(null)	(null)	0.0313802700330305	-0.468619729966969
3	3	1 satisfactory rating	23	1	0.0132539822796954	-0.486746017720305
4	6	(null)	(null)	(null)	0.0255671819827542	-0.474432818017246
5	0	(null)	(null)	(null)	0.0273426656560493	-0.472657334343951
6	284	(null)	10	(null)	0.00673247165971619	-0.493267528340284
7	0	(null)	(null)	(null)	0.0292755414470232	-0.470724458552977
8	0	2 satisfactory ratings	(null)	1	0.00701388289900789	-0.492986117100992
9	115	2 satisfactory ratings	1	0	0.0154828208525022	-0.484517179147498
10	157	No File	(null)	99	0.00136647916434529	-0.498633520835655
11	0	(null)	(null)	(null)	0.0274657975381126	-0.472534202461887
12	0	(null)	(null)	(null)	0.0246972324736272	-0.475302767526373
13	0	(null)	(null)	(null)	0.026059262103508	-0.473940737896492
14	0	(null)	(null)	(null)	0.034842179753759	-0.465157820246241
15	0	(null)	(null)	(null)	0.0306606035987099	-0.46933939640129

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Example 3: Output dataset with outliers (One Class SVM method).



	chased	Total Service Charge Revenue	Worst Rating Des	Year at Address	Year Last Check	Separating_Hyperplane_Distance	Scoring_Function	Anomaly_Flag
1		1	3 satisfactory ratings	10	1	85.2417619983697	733.887210195545	☑
2		2	(null)	(null)	(null)	37.5159204211059	686.161368618281	☑
3		3	1 satisfactory rating	23	1	101.452383242112	750.097831439287	☑
4		6	(null)	(null)	(null)	77.5230296579242	726.1684778551	☑
5		0	(null)	(null)	(null)	139.937656033861	788.583104231037	☑
6		111	5 or more satisfactory ratings	(null)	0	17.8063321814157	666.451780378591	☑
7		284	(null)	10	(null)	119.203803191088	767.849251388263	☑
8		0	(null)	(null)	(null)	78.7513877270943	727.39683592427	☑
9		47	File but no rating	1	0	-537.116733273682	111.528714923494	☐
10		38	File but no rating	3	98	82.3781205429938	731.023568740169	☑
11		0	2 satisfactory ratings	(null)	1	85.0681697151437	733.713617912319	☑
12		115	2 satisfactory ratings	1	0	141.446050932933	790.091499130109	☑
13		157	No File	(null)	99	134.046302133598	782.691750330774	☑

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

The Isolation Forest and One Class SVM methods can be used to clean the input dataset from anomalies. The output dataset without outliers / anomalies can be used as a clean dataset for modeling.

The output of the Local Outlier Factor method is a single dataset that includes all records of the original data. It is assumed that all records are not outliers (clean dataset). Use the Sklearn Novelty & Outlier Scoring node to check a new dataset for anomalies.

See also: Novelty & Outlier Scoring.

Novelty & Outlier Scoring

Scoring a Novelty & Outlier Detector model is similar to scoring other models, but it produces anomaly estimation metrics and outlier indicators instead of probabilities. A Novelty & Outlier Detector model is scored using the **Scoring** node in the Action palette. For general information about scoring, see the topic About Scoring.

Prerequisites for Novelty & Outlier model scoring: a **Novelty & Outlier Detector** node and an input dataset with new data that has no outlier or anomaly information.

> ***Note***: For the Local Outlier Factor method, scoring is not applicable when the *Novelty* option is set to False.

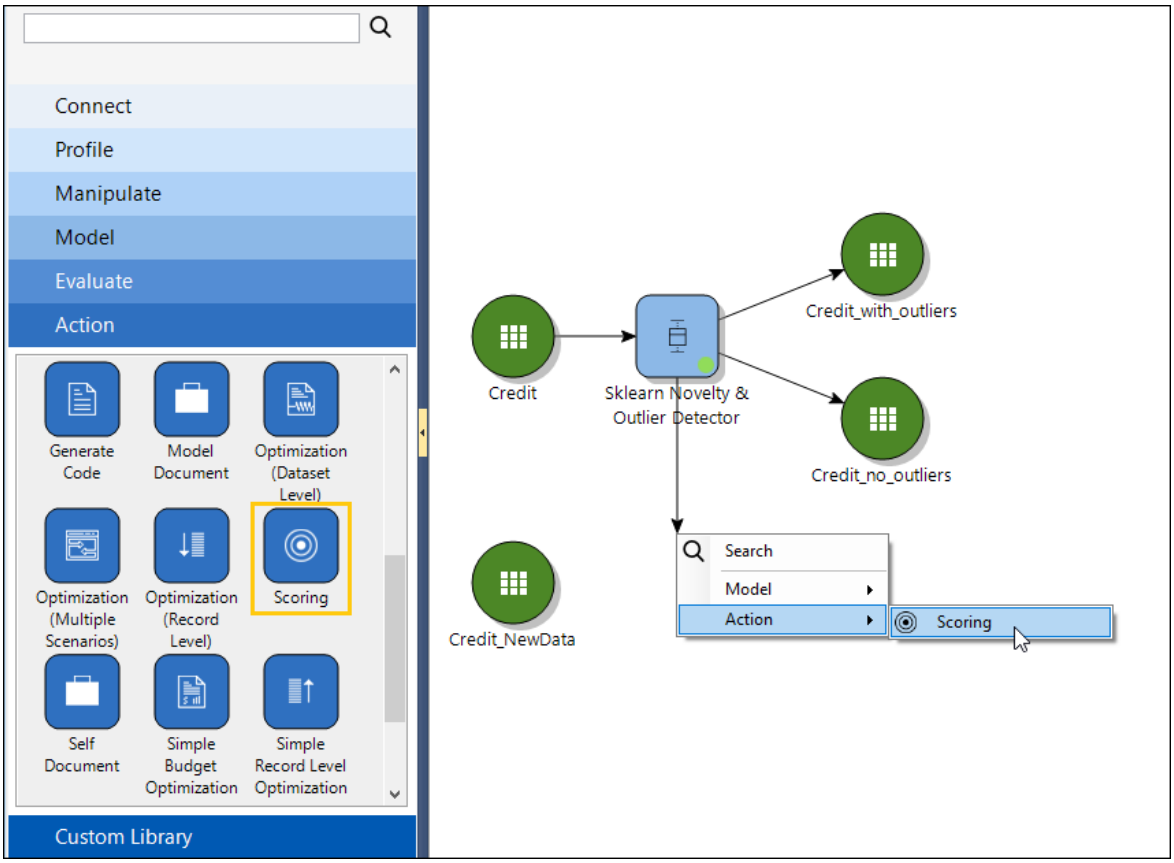
To add a Scoring operation to your workflow:

- Drag the **Scoring** node from the **Action** palette onto the Workflow canvas and connect the inputs: **a Novelty & Outlier Detector** node and a dataset node that contains the new data to score.

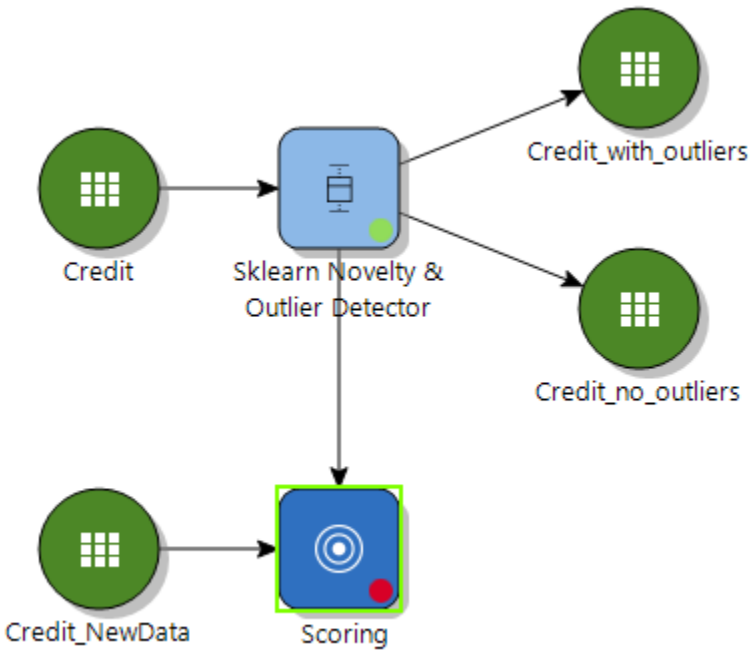
  OR

- Right-click anywhere on the workflow canvas and select the context menu command **Action --> Scoring**. This will create a new Scoring node. Connect the inputs a **Novelty & Outlier Detector** node and a dataset node that contains the data to score.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



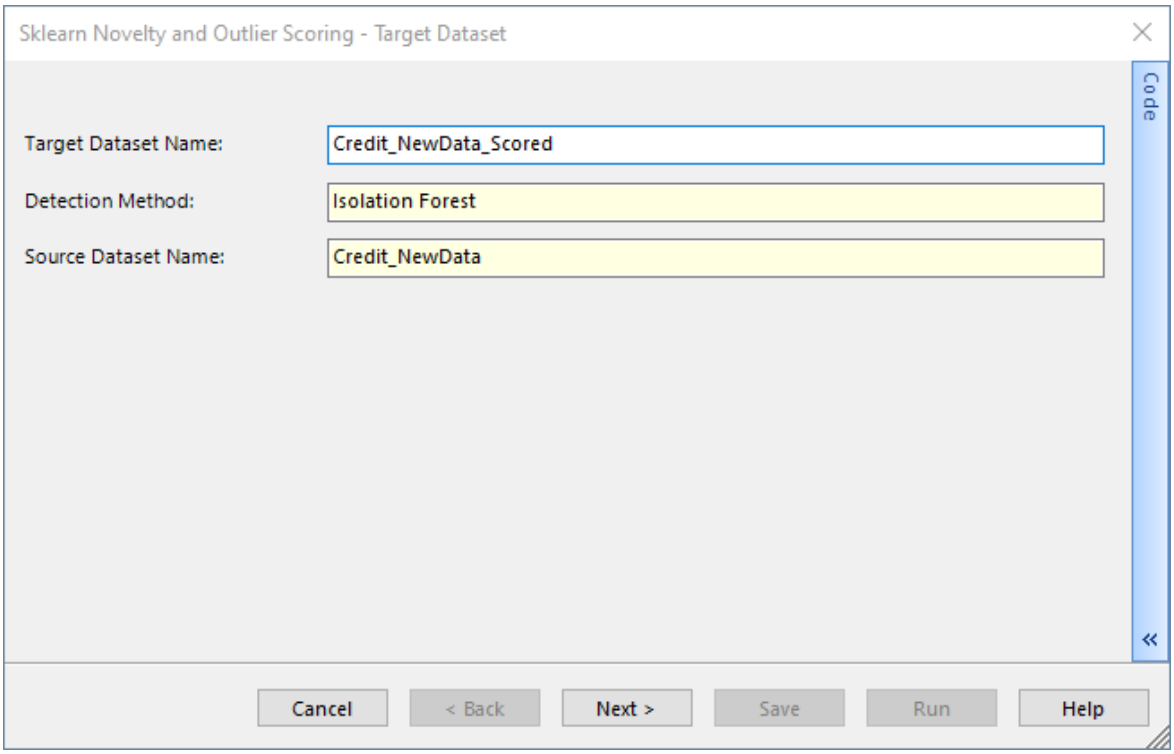
Double-click the **Scoring** node or right-click on it and select **Modify**.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The Scoring wizard opens. It displays the name of the input dataset and the method used in the input Novelty & Outlier Detection model node.

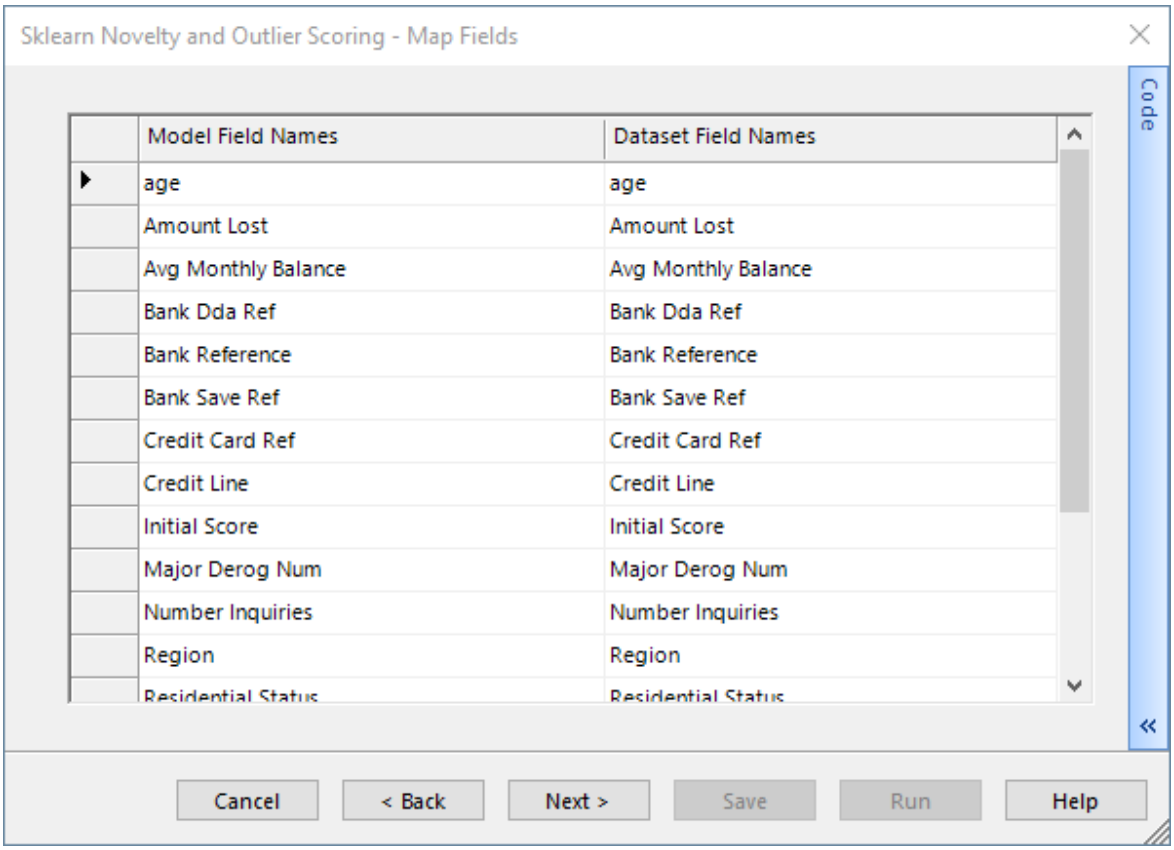
In the *Target Dataset Name* field, specify the name for the output dataset that will contain the scores.



Click **Next**.

The next step provides the option of mapping the field names in the model to the ones in the validation input dataset. In some situations, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map fields by clicking in the right column (Dataset Field Names) and selecting the right fields to match the model variables in the left column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

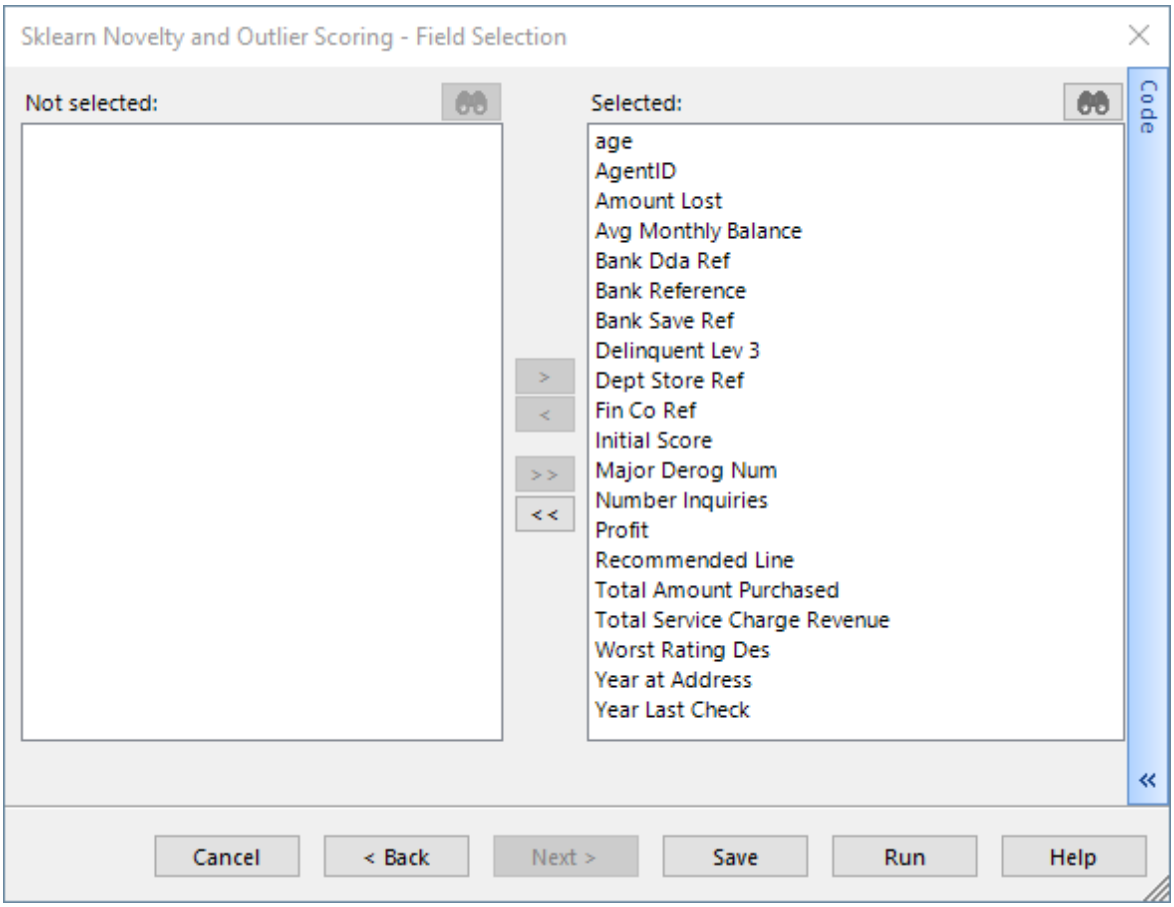


Note: When scoring Novelty & Outlier Detector models that use derived variables (also called calculated columns) with expressions, the input dataset must include these variables. This is different from the case of native Altair models with derived variables, which can be scored on a dataset with only *original* variables that the derived ones depend on, while the derived ones are not required.

Click **Next**.

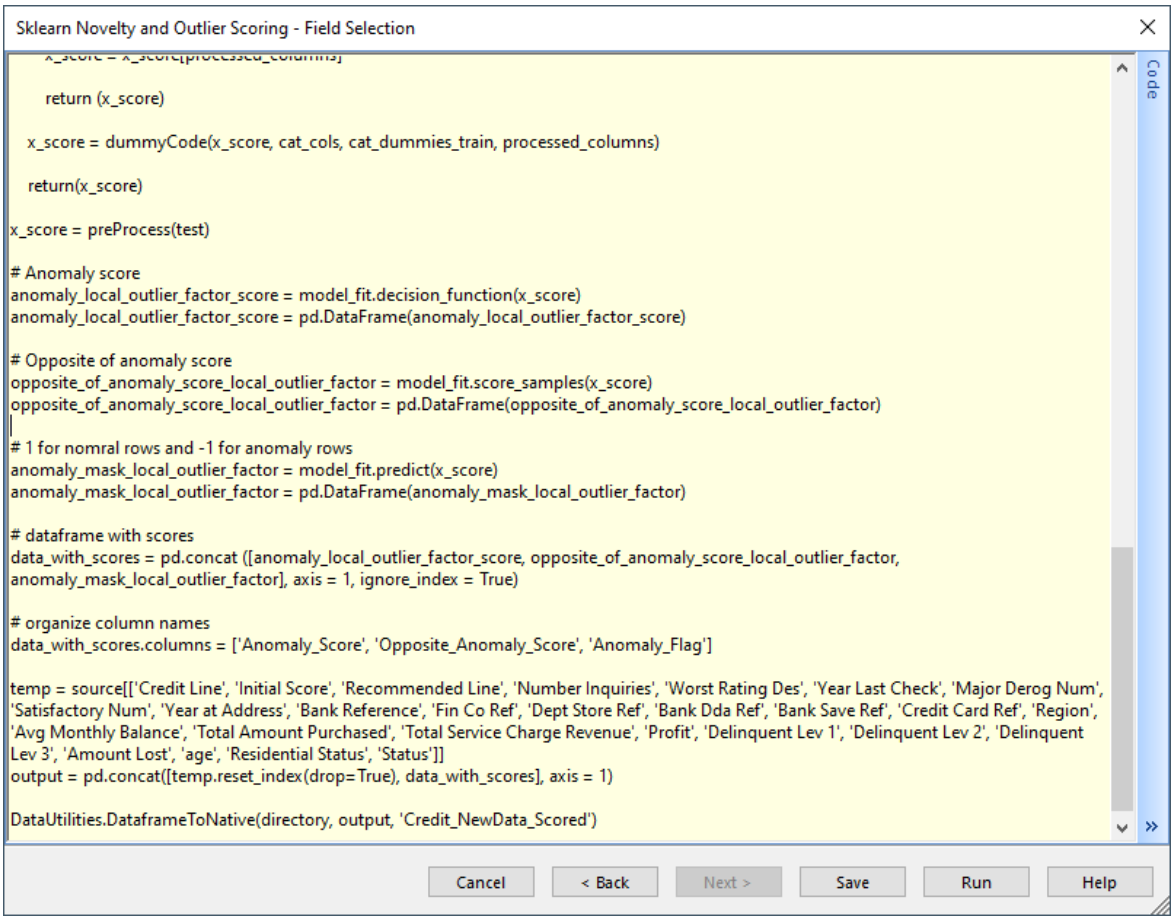
In the **Field Selection** page, choose the fields from the input dataset that you would like to include in the scoring output in addition to the new anomaly score fields. By default, all fields in the scoring input dataset are selected to be included in the output.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



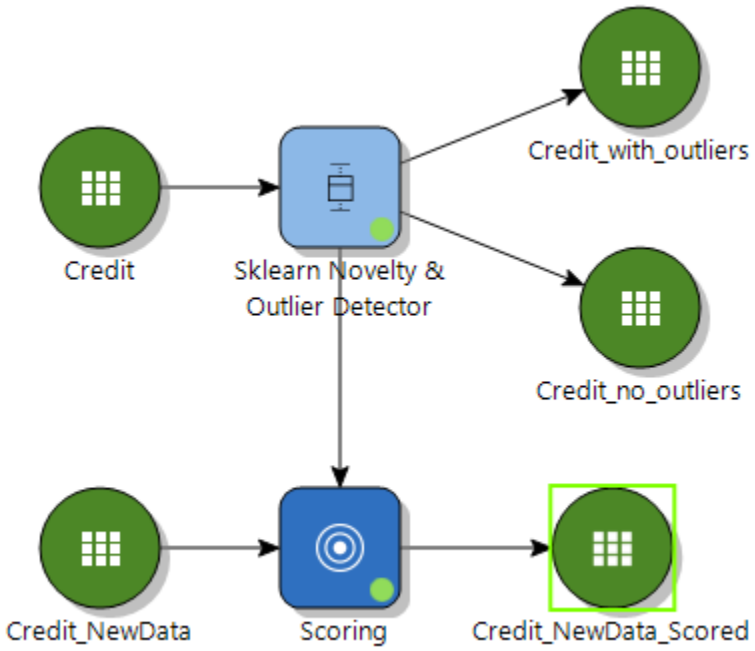
To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the scoring process immediately. The model is scored in a Python container session, and then the dataframe with generated scores is converted back from Pandas to Altair KDD format. The output dataset node is created in the workflow. Double-click the output dataset to view the results.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click the Data tab of the output dataset and scroll to the rightmost columns. In this example from scoring a model that uses the Isolation Forest method, the scored dataset has the *Anomaly_Flag* column showing the outlier indicator based on the values of *Anomaly_Score* and *Opposite_Anomaly_Score*.

	Delinquent	Amount Lost	age	Residential Status	Status	Anomaly_Score	Opposite_Anomaly_Score	Anomaly_Flag
1	0	0	(null)	(null)	OK	0.0306769841368138	-0.469323015863186	1
2	3	557.67	22	Rent	Bad Debt	-0.0157806793598609	-0.515780679359861	-1
3	0	0	37	Own Home	OK	0.00701388289900789	-0.492986117100992	1
4	0	0	25	Other	OK	0.0154828208525022	-0.484517179147498	1
5	0	0	81	(null)	OK	0.0239839169398195	-0.47601608306018	1
6	0	0	(null)	(null)	OK	0.0341440893542848	-0.465855910645715	1
7	0	0	(null)	(null)	OK	0.0299562376556643	-0.470043762344336	1
8	0	0	62	(null)	OK	0.0245103050179628	-0.475489694982037	1
9	1	0	67	(null)	OK	0.0175292749437517	-0.482470725056248	1
10	0	0	30	Other	OK	0.0200072406322655	-0.479992759367735	1
11	0	0	20	Own Home	OK	0.0108821254141328	-0.489117874585867	1
12	0	0	61	Own Home	OK	0.00165246899136623	-0.498347531008634	1
13	0	0	35	Rent	OK	0.0118635398366399	-0.48813646016336	1
14	0	0	21	(null)	OK	0.0256281662644814	-0.474371833735519	1
15	0	0	59	Own Home	OK	0.000304559304685592	-0.499695440695314	1

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

See also: Novelty & Outlier Detector

Sklearn XGBoost Model

Overview

The **Sklearn XGB** node in the **Model** palette of Knowledge Studio allows you to build models using the Python implementation of XGBoost (Extreme Gradient Boost) algorithm. It uses gradient

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

boosting functions for classification and regression in **`scikit-learn`** - a popular Python library for machine learning. For details, see the scikit-learn documentation for Gradient Boosting Classifier and Gradient Boosting Regressor (Copyright © 2007 - 2021, scikit-learn developers, BSD License).

The node wizard provides a convenient graphical interface to specify the options and parameters for the required functions in the `scikit-learn (sklearn)` library. The wizard automatically translates the inputs into Python code, which can be previewed before running the node. The code then runs in a built-in Python environment, loading all required packages and their dependencies. The input dataset is converted from the Altair KDD format to pandas dataframes in the process.

After training, the model can be validated using the Model Validation node and scored on new data using the Scoring node.

Building an Sklearn XGB Model

An **Sklearn XGB** model requires a single dataset as input. An **Sklearn XGB** node can be added to the workflow in one of the following ways:

- Drag a connection arrow from the input dataset node to any area on the Workflow canvas background. From the context menu that opens upon releasing the mouse button, select **Model --> Sklearn XGB** as in the example screenshot below. A new **Sklearn XGB** node will be created, and the wizard will open.

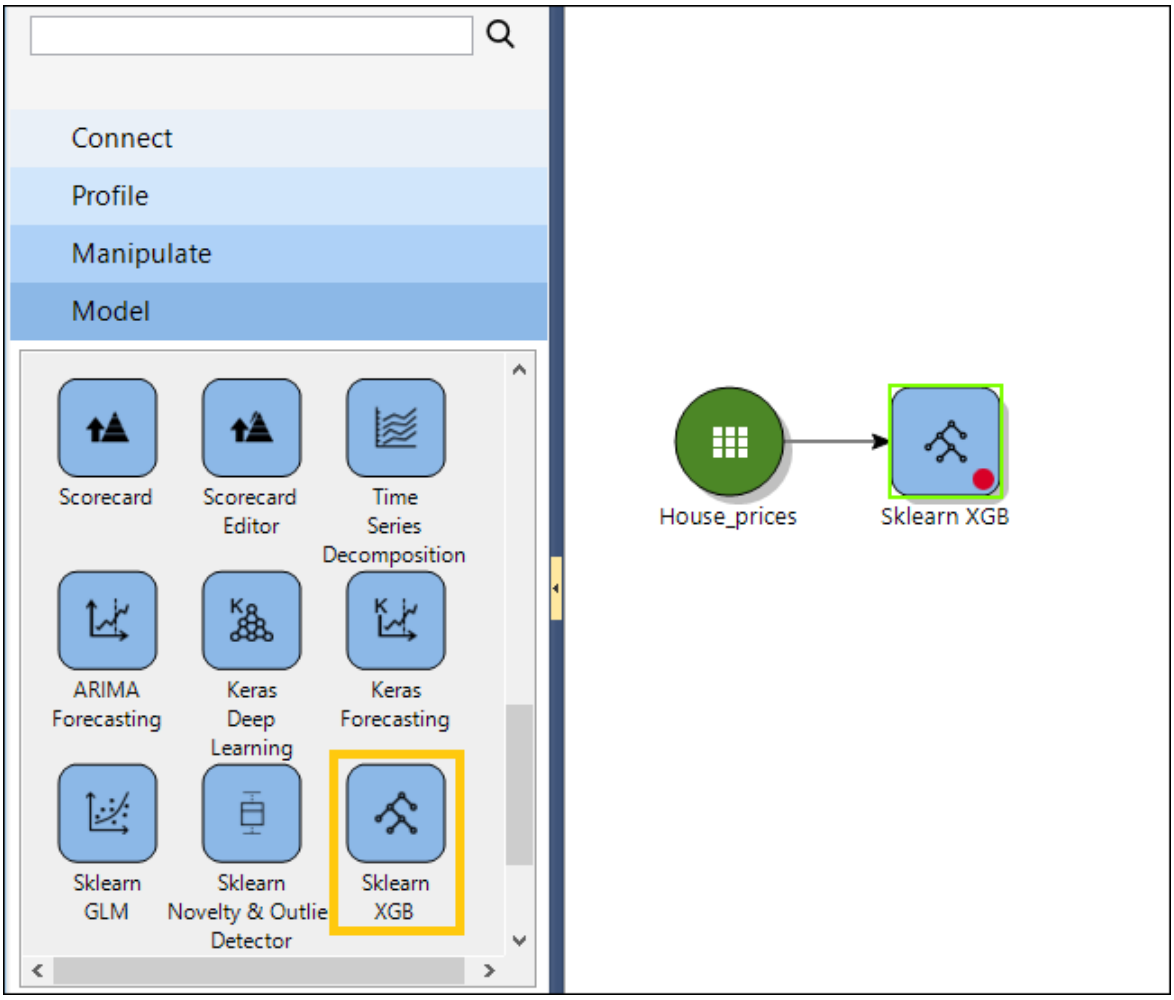
  OR

- Right-click on the Workflow canvas and select **Model --> Sklearn XGB** from the context menu. Connect the input dataset. Optionally, connect another predictive model node as a base model to use as a template.

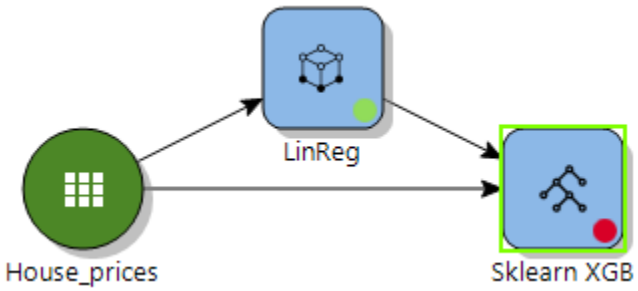
  OR

- Open the **Model** palette and drag the **Sklearn XGB** node onto the workflow canvas. Connect the input dataset. Optionally, connect another predictive model node as a base model to use as a template.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



In the following example, an existing Linear Regression model is used as a base model for the new Sklearn XGB model.



Double-click the Sklearn XGB node or right-click on it and select **Modify** to open the wizard.

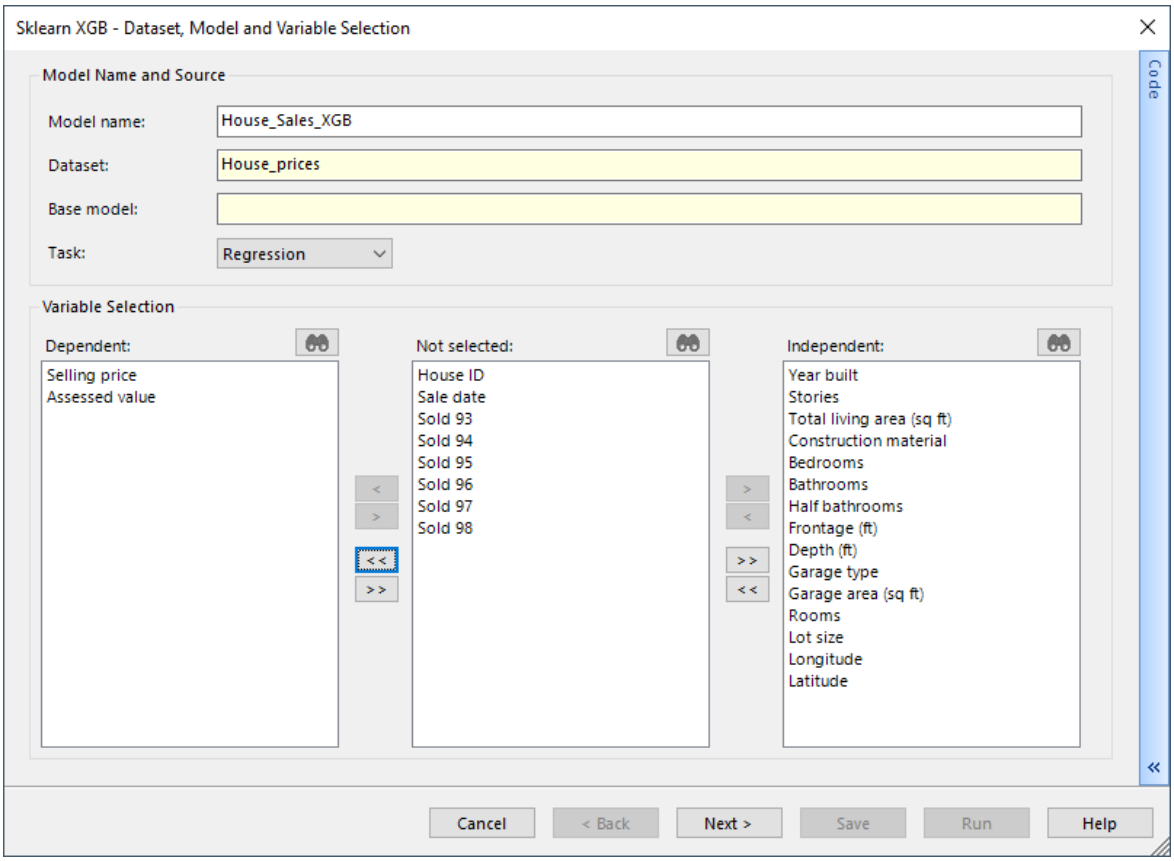
In the first page of the wizard:

- Specify the model name.
- Specify the Task type: *Regression* or *Classification*

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Select at least one dependent variable. If multiple dependent variables are selected, all of them must be either continuous or binary. Multiple binary dependent variables must be of the same data type.
- Select the independent variables.
    - To select or deselect variables, use the arrow buttons or double-click on the variable names. Advanced field finding and multi-selection capabilities are also available via the Field Finder dialog, which can be opened by clicking the binoculars button .

Note: If a Base model is used, the dependent variable and independent variables are automatically inherited from the Base model.



Click **Next**.

In the Preprocessing Parameters page, specify the following parameters for imputation and scaling. See the scikit-learn documentation for Gradient Boosting Classifier and Gradient Boosting Regressor for the detailed description of the function parameters.

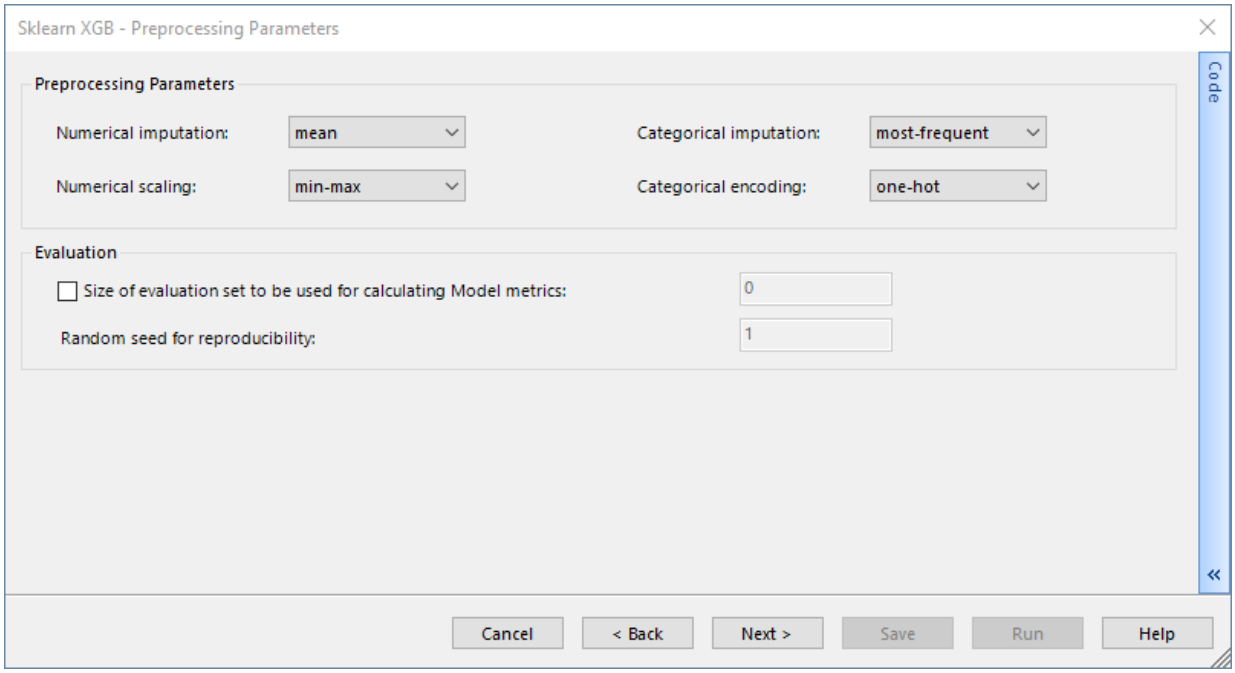
- Specify the *imputation* method for numerical variables. Available methods are: *mean, median, most-frequent, zero, min, max*. The default is *mean*.
- Specify the *scaling* method for numerical variables. Available methods are: *min-max* and *standard*. The default limit is *min-max*.

For categorical variables (classification problems), the imputation method is set to *most-frequent*, and the encoding method is set to *one-hot* encoder.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Imputation is applied to missing and NaN (undefined) values.

- Specify the size of the evaluation sample to be used for calculating model performance metrics during training. Select the checkbox and enter a number between 0 and 1 to indicate the sample size with respect to the training dataset.
    - Note: For Classification tasks, when splitting the dataset for model evaluation, each value of the dependent variable must occur in both partitions. If the dependent variable contains a value that occurs only once in the entire dataset, the following error may occur: "*ValueError: The least populated class in y has only 1 member, which is too few. The minimum number of groups for any class cannot be less than 2*." In such cases, disable evaluation by deselecting the checkbox in the Preprocessing Parameters page of the wizard.



Click **Next**.

In the Model Parameters page, specify the following model parameters. See the scikit-learn documentation for Gradient Boosting Classifier and Gradient Boosting Regressor for the detailed description of the function parameters.

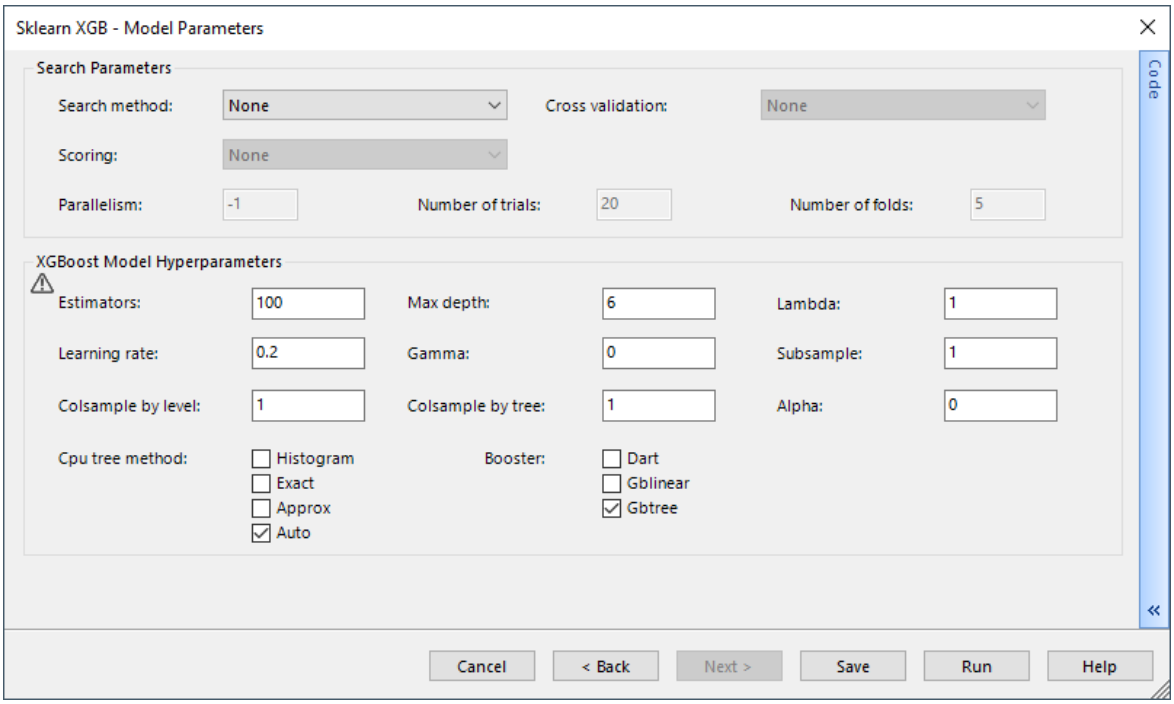
- Specify the **Search Parameters**:
    - The *Search method* (None, *GridSearchCV,* or *OptunaSearchCV*). The default is None.
    - The *Cross validation* option (None, *KFold*, or *StratifiedKFold*). The default is None. Not applicable if Search method is None.
    - The *Scoring* option
        - For Regression tasks, the Scoring options are: None, *Neg mean absolute error*, *Neg mean squared error*, or *Neg mean squared log error*. The default is None.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

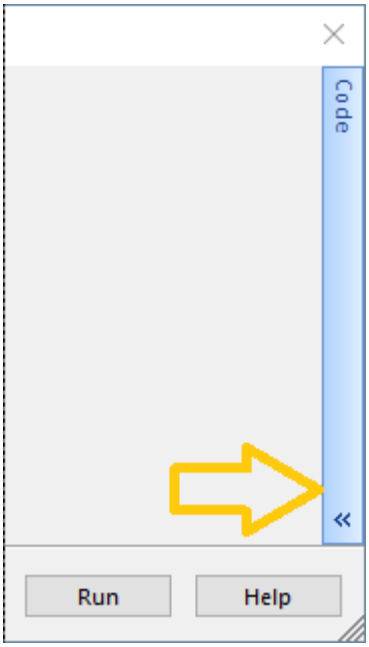
        - For Classification tasks, the Scoring options are: None, *Accuracy*, *F1 macro, F1 micro, Precision macro, Precision micro, Recall macro,* or *Recall micro*. The default is None.
    - The *Parallelism* parameter. The default is -1.
    - The *Number of trials.* The default is 20.
    - The *Number of folds*. The default is 5.
- Specify the **XGBoost Model Hyperparameters**:
    - The number of *estimators* in the ensemble. The default is 100.
    - The *maximum depth* of the tree estimators in the ensemble. The default limit is 6 levels.
    - The *learning rate*. The default is 0.2.
    - The *Subsample* size. Specify a positive number between 0 and 1. The default is 1.
    - The *alpha*, *lambda*, and *gamma* parameter values (the defaults are 0, 1, and 0, respectively)
    - *Colsample by level* and *Colsample by tree* (the default is 1 for both)
    - *Cpu tree method* (one or more, depending on the selected *Search method*)
    - *Booster* (one or more, depending on the selected Search method)

    When specifying the hyperparameter values:
        - If the *Search method* is not specified (*None* is selected), you can specify a single value per parameter.
        - If the selected Search method is *GridSearchCV*, you can specify
            - -- a single value per parameter (static during the search).
            - -- three slash-separated values for interval creation (start, end, step). For example, **100/120/5** will create the interval values to be tested [100, 110, 115] (last value not included).
            - -- multiple *Cpu tree methods* and *Booster* choices.
        - If the selected Search method is *OptunaSearchCV*, you can specify
            - -- a single value per parameter (static during the search)
            - -- two comma-separated values for interval search (lower bound, upper bound).
            - -- multiple *Cpu tree methods* and *Booster* choices.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

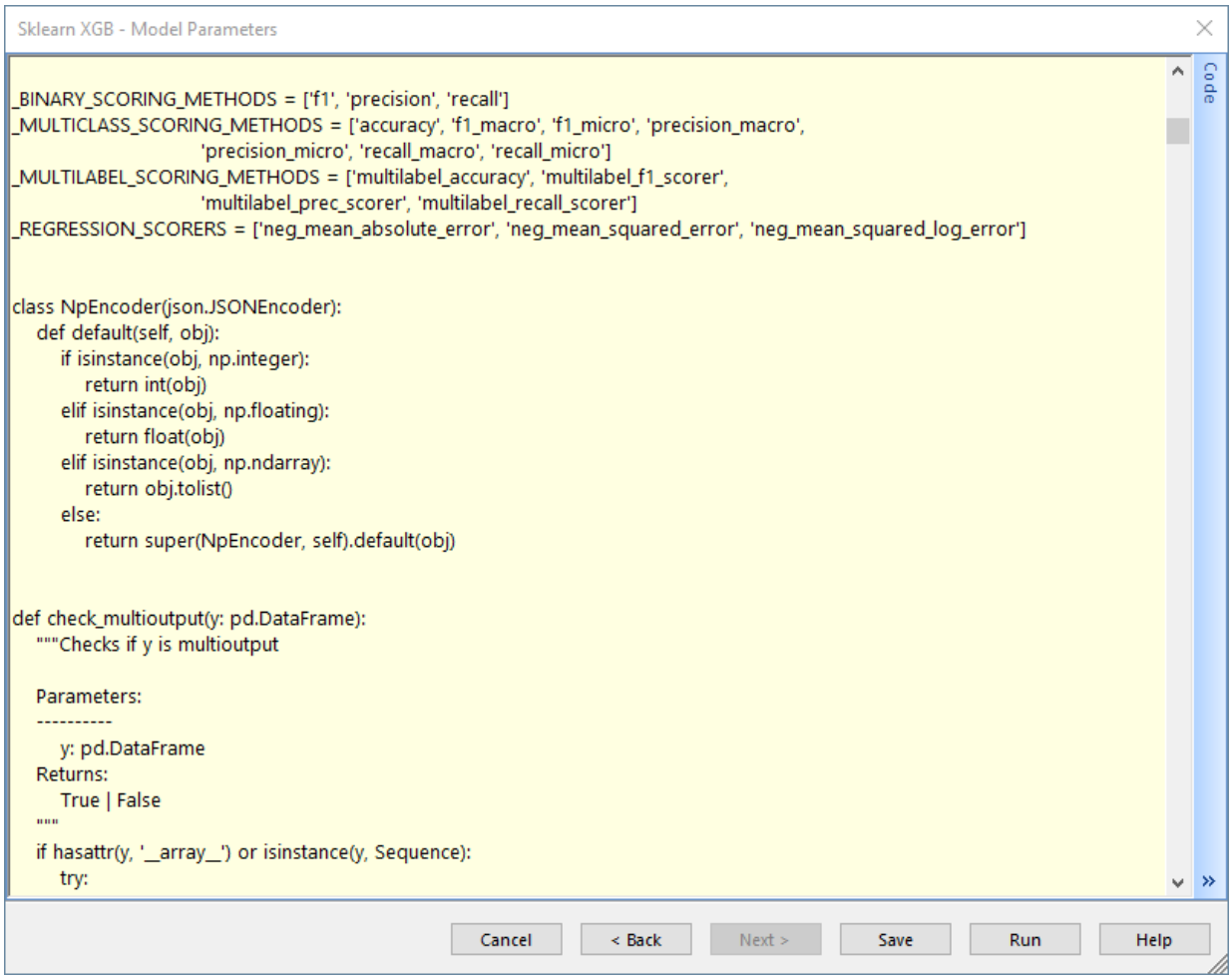


To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog.



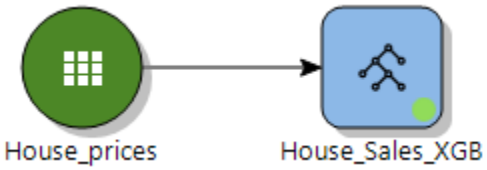
This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



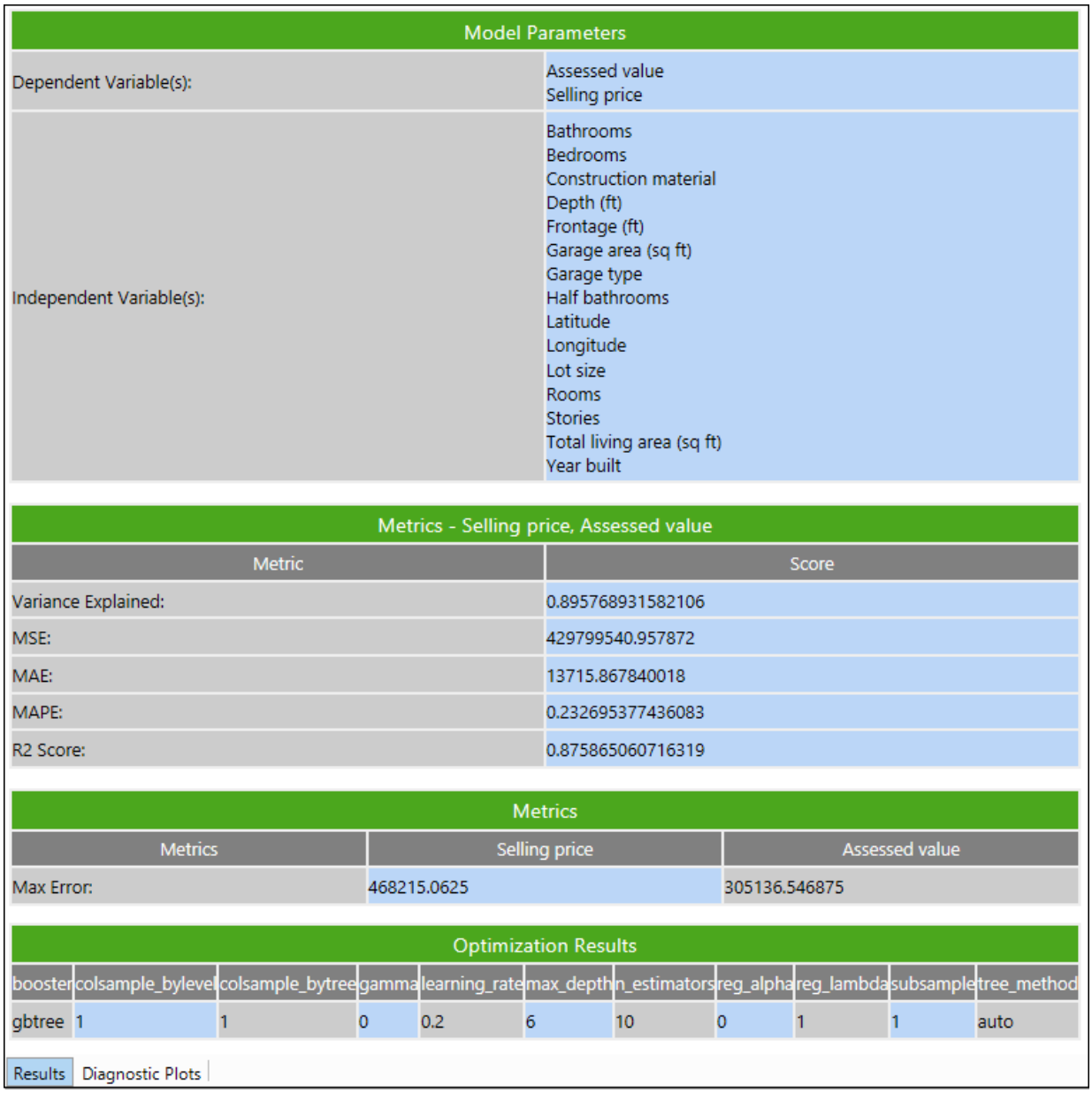
Click **Save** to save the settings or click **Run** to save the settings and run the process immediately. The input dataset is converted to a Pandas dataframe, and the required functions of `sklearn` and other libraries are applied. Then the Gradient Boosting Classifier or Regressor functions with specified parameters are applied.

After running the node, double-click the model node to view the results.



The model training report displays the list of the dependent variable(s) and independent variables, and the model accuracy and performance metrics.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Model Parameters	
Dependent Variable(s):	Assessed value Selling price
Independent Variable(s):	Bathrooms Bedrooms Construction material Depth (ft) Frontage (ft) Garage area (sq ft) Garage type Half bathrooms Latitude Longitude Lot size Rooms Stories Total living area (sq ft) Year built

Metrics - Selling price, Assessed value	
Metric	Score
Variance Explained:	0.895768931582106
MSE:	429799540.957872
MAE:	13715.867840018
MAPE:	0.232695377436083
R2 Score:	0.875865060716319

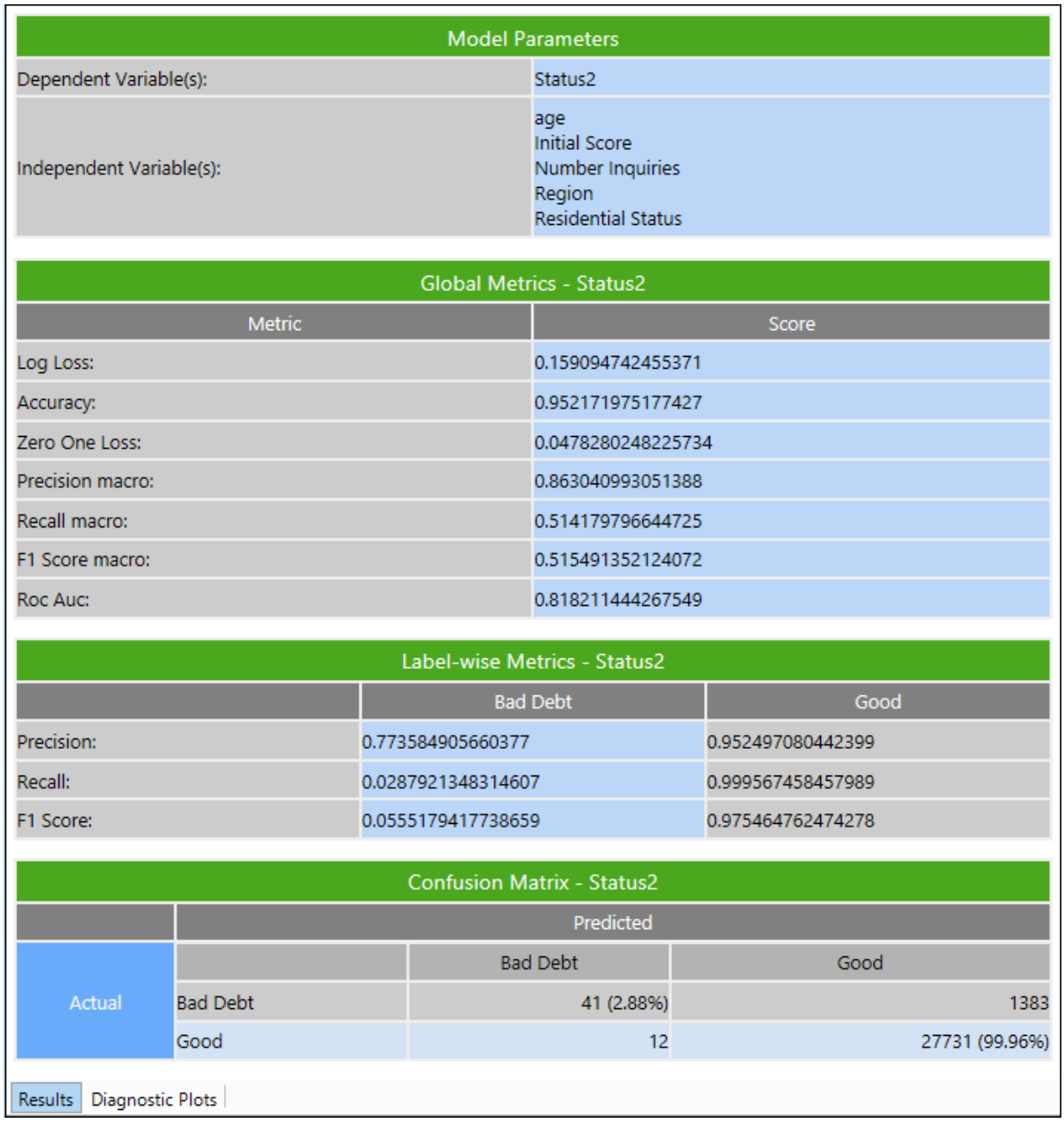
Metrics		
Metrics	Selling price	Assessed value
Max Error:	468215.0625	305136.546875

Optimization Results										
booster	colsample_bylevel	colsample_bytree	gamma	learning_rate	max_depth	n_estimators	reg_alpha	reg_lambda	subsample	tree_method
gbtree	1	1	0	0.2	6	10	0	1	1	auto

Results | Diagnostic Plots

In the case of a categorical dependent variable, the Results report shows Confusion Matrices for the model performance on the training and test partitions. Here is an example of the Results report for an XGB model with a single binary dependent variable:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Model Parameters	
Dependent Variable(s):	Status2
Independent Variable(s):	age Initial Score Number Inquiries Region Residential Status

Global Metrics - Status2	
Metric	Score
Log Loss:	0.159094742455371
Accuracy:	0.952171975177427
Zero One Loss:	0.0478280248225734
Precision macro:	0.863040993051388
Recall macro:	0.514179796644725
F1 Score macro:	0.515491352124072
Roc Auc:	0.818211444267549

Label-wise Metrics - Status2		
	Bad Debt	Good
Precision:	0.773584905660377	0.952497080442399
Recall:	0.0287921348314607	0.999567458457989
F1 Score:	0.0555179417738659	0.975464762474278

Confusion Matrix - Status2			
	Predicted		
		Bad Debt	Good
Actual	Bad Debt	41 (2.88%)	1383
	Good	12	27731 (99.96%)

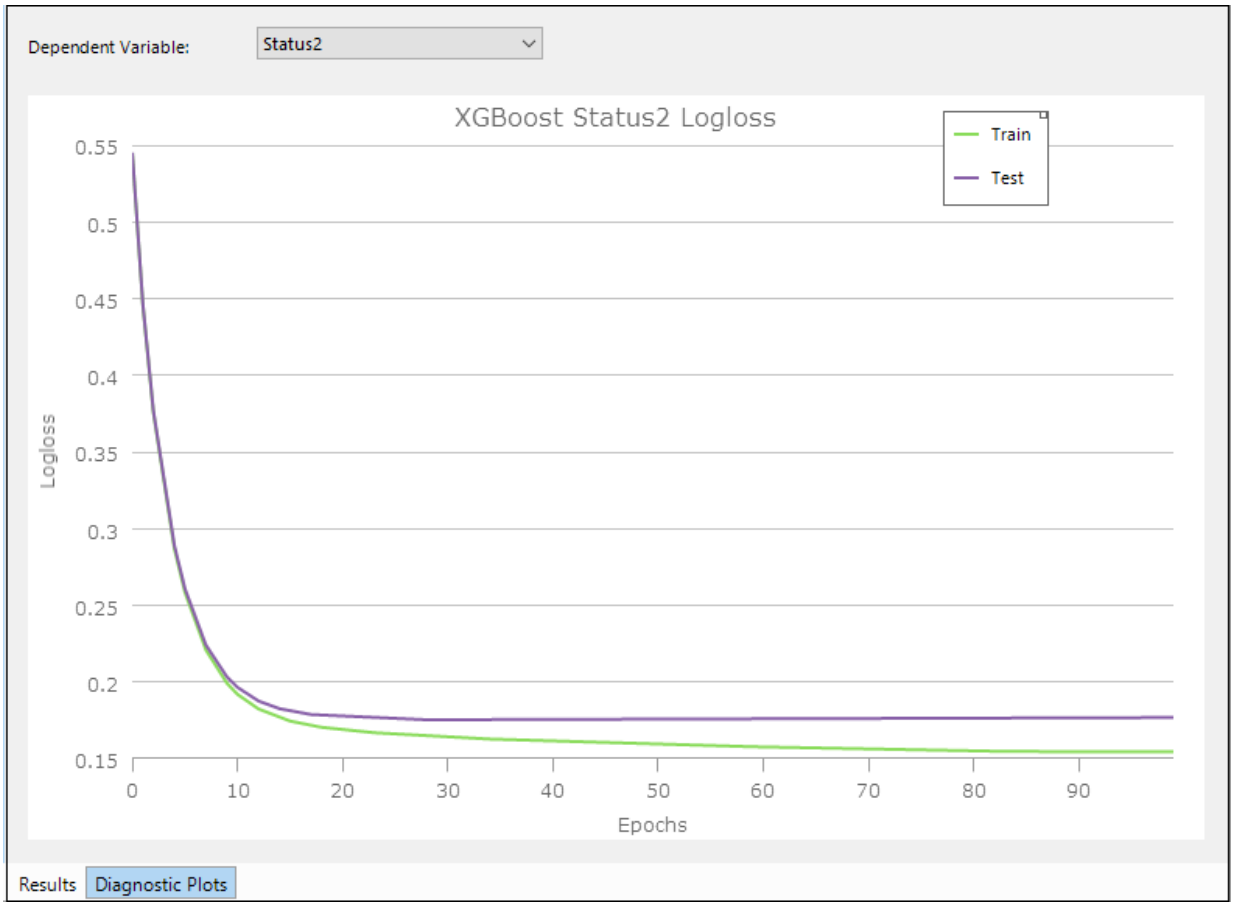
Results | Diagnostic Plots

The **Diagnostic Plots** tab shows diagnostic plots that depend on the type of the dependent variable. If the model has multiple dependent variables, select the desired variable from the drop-down list at the top of the Plots view.

For a discrete dependent variable, it shows the *Logloss* plot for both Training and Testing partitions for all epochs. The X axis shows the number of epochs, while the Y axis shows the Classification error rate on the left chart and Logloss on the right chart.

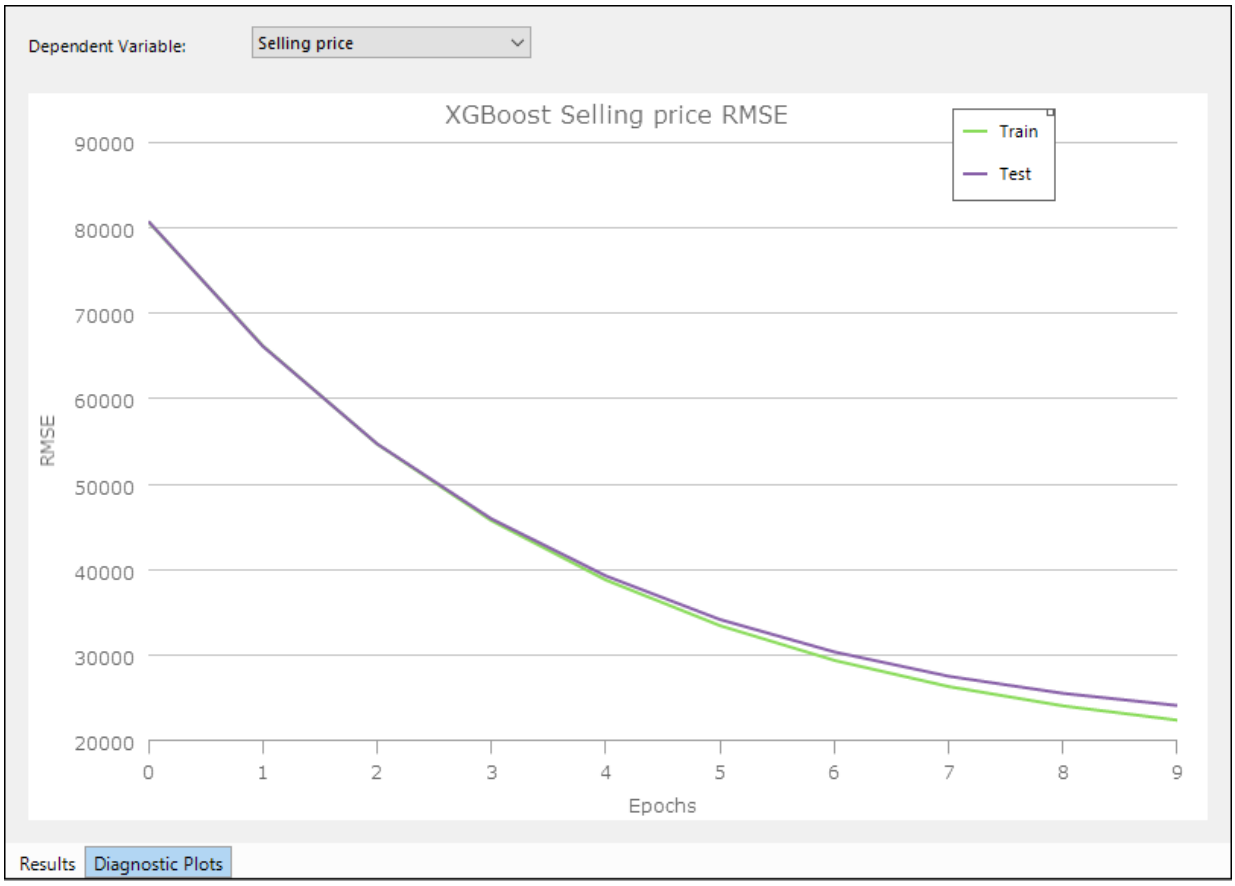
> *Note*: Diagnostic plots and confusion matrix are not supported in the case of multiple categorical dependent variables.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



For a continuous dependent variable, the **Diagnostic Plots** tab shows the Root Mean Square Error (RMSE) on the Training and Test partitions. The X axis shows the epochs.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



After training, the model can be used for scoring new data using the Scoring node. If the model is scored on the training dataset or another dataset with known outcomes of the dependent variable, the resulting dataset can be used to evaluate the model performance with Model Analyzer.

Scoring an XGBoost Model

An Sklearn XGBoost Model is scored using the **Scoring** node in the Action palette. For general information about scoring, see the topic About Scoring.

Prerequisites for Scoring: an **Sklearn XGB** model node and an input dataset to score.

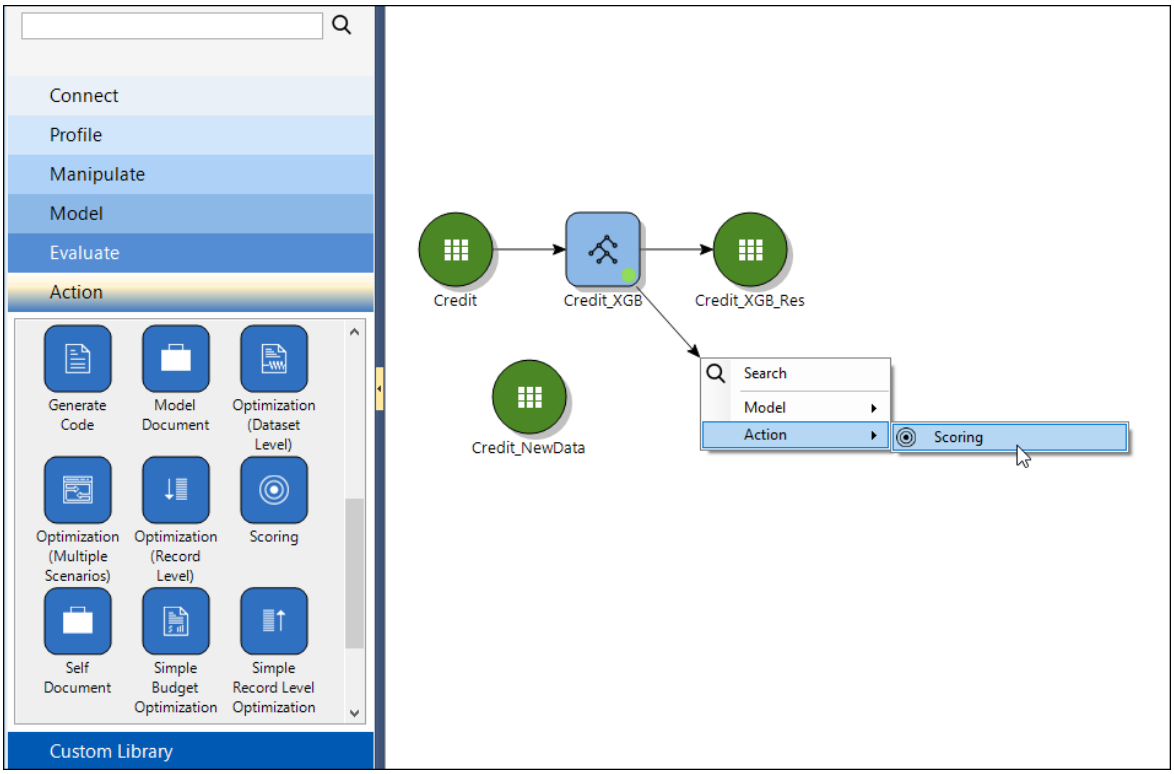
To add a Scoring operation to your workflow:

- Drag the **Scoring** node from the **Action** palette onto the Workflow canvas and connect the inputs: an **Sklearn XGB** model node and a dataset node that contains the data to score.

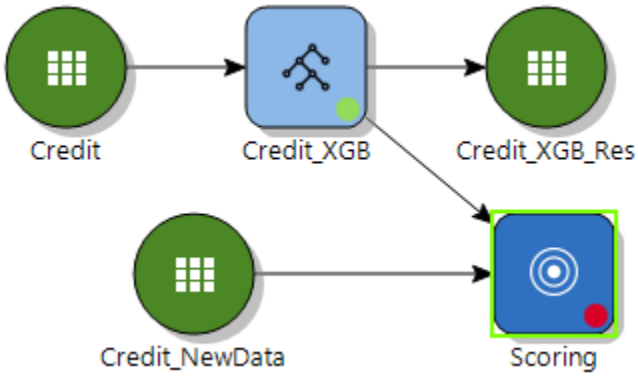
  OR

- Right-click anywhere on the workflow canvas and select the context menu command **Action --> Scoring**. This will create a new Scoring node. Connect the inputs: an **Sklearn XGB** node and a dataset node that contains the data to score.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



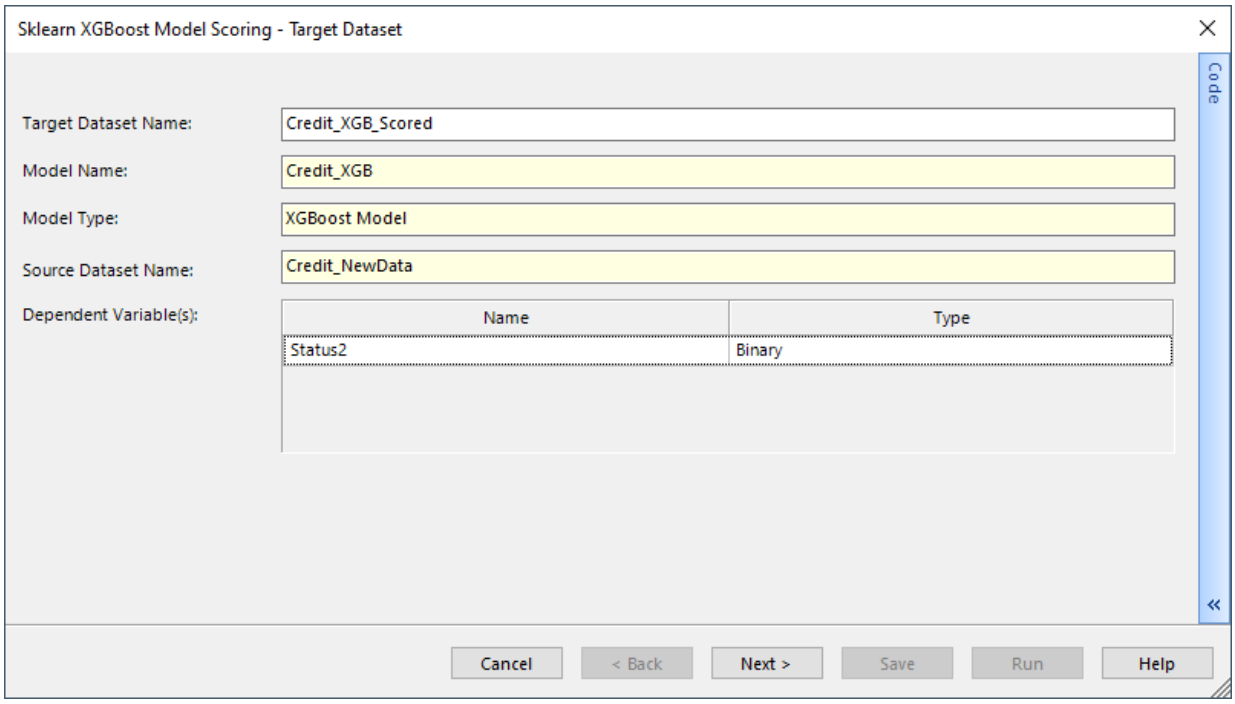
Double-click the **Scoring** node or right-click on it and select **Modify**.



The Scoring wizard opens. It displays the names of the input model and dataset, the regressor type, and the name of the dependent variable.

In the *Target Dataset Name* field, specify the name for the output dataset that will contain the scores.

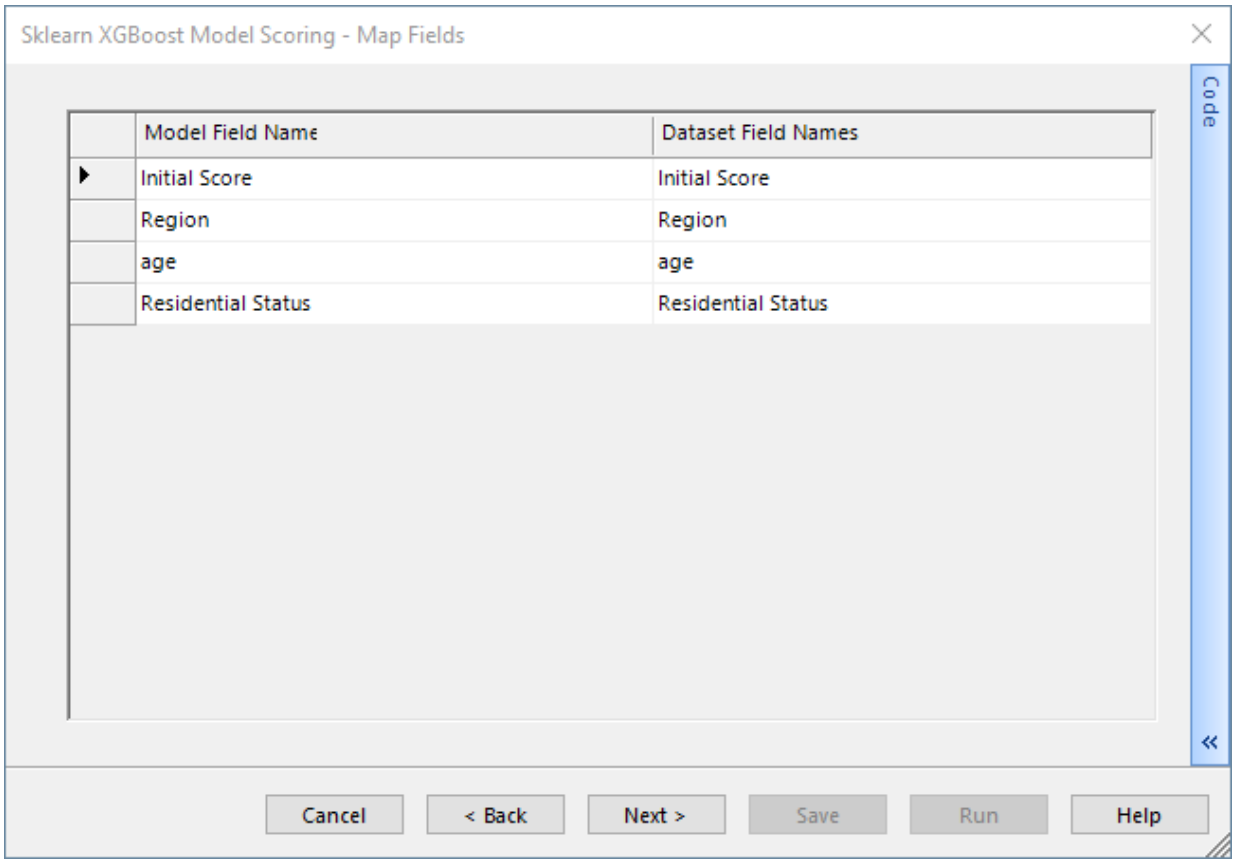
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

The next step provides the option of mapping the field names in the model to the ones in the validation input dataset. In some situations, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map fields by clicking in the right column (Dataset Field Names) and selecting the right fields to match the model variables in the left column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

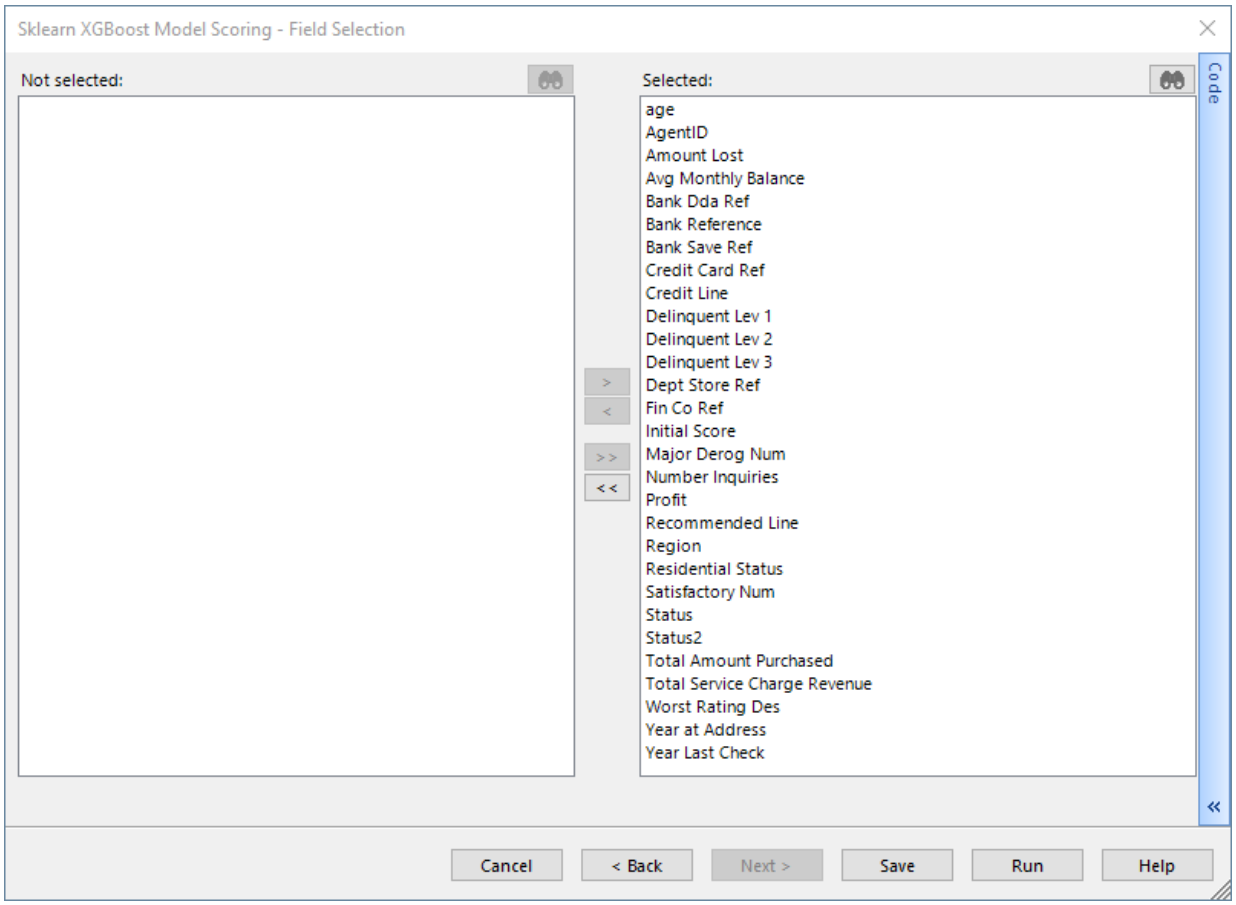


Note: When scoring XGB models that use derived variables (also called calculated columns) with expressions, the input dataset must include these variables. This is different from the case of native Altair models with derived variables, which can be scored on a dataset with only *original* variables that the derived ones depend on, while the derived ones are not required.

Click **Next**.

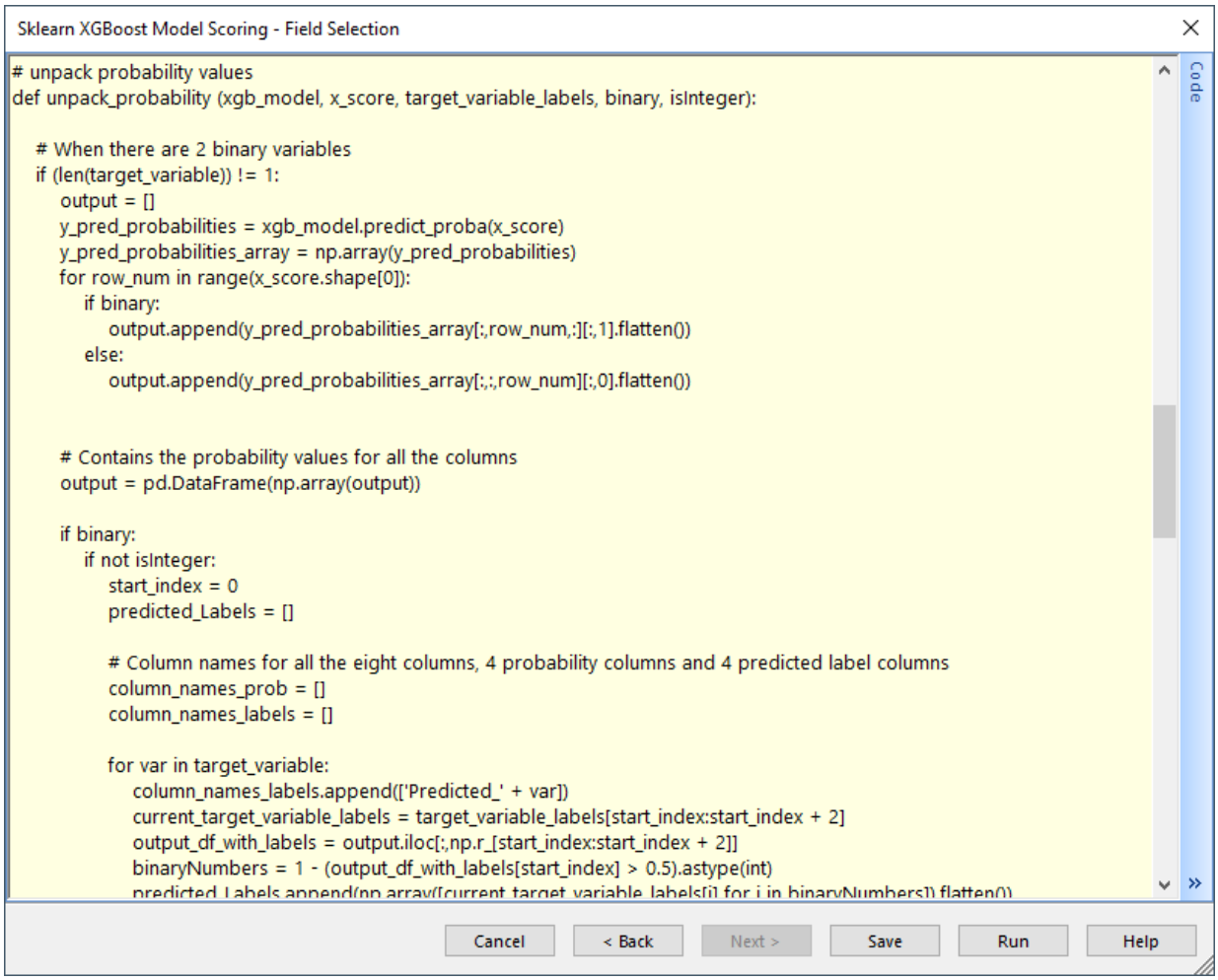
In the **Field Selection** page, choose the fields from the input dataset that you would like to include in the scoring output in addition to the new score fields. By default, all fields in the scoring input dataset are selected to be included in the output.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

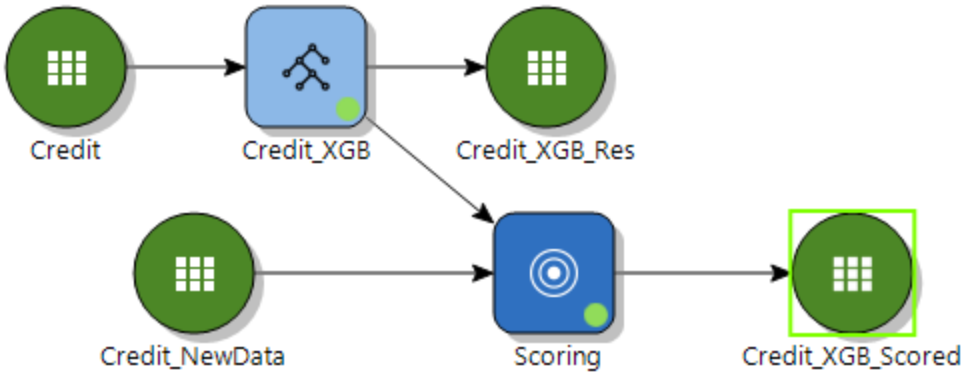


To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

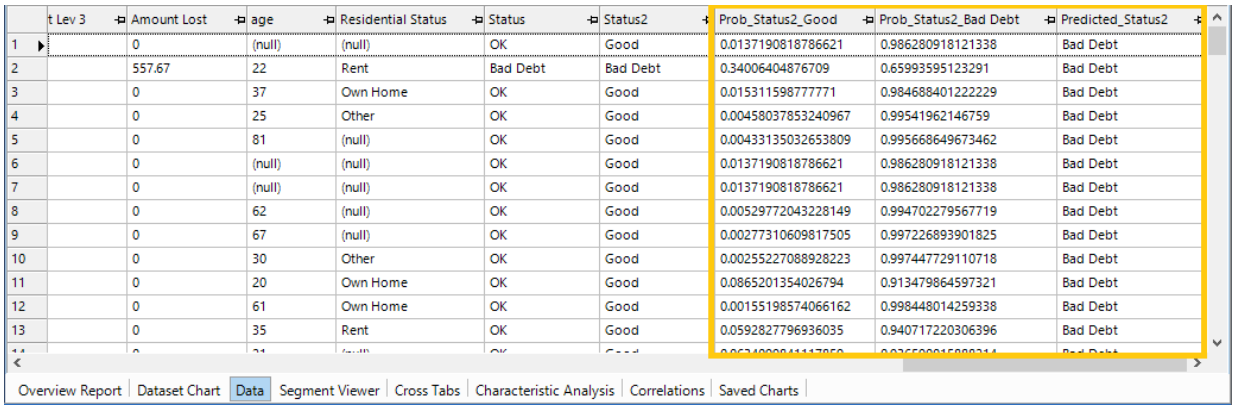


Click **Save** to save the settings, or click **Run** to save the settings and run the scoring process immediately. The model is scored in a Python container session, and then the dataframe with generated predictions (scores) is converted back from Pandas to Altair KDD format. The output dataset node is created in the workflow. Double-click the output dataset to view the results.



Click the Data tab of the output dataset to see the probabilities of the dependent variable(s) and model predictions.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



	t Lev 3	Amount Lost	age	Residential Status	Status	Status2	Prob_Status2_Good	Prob_Status2_Bad Debt	Predicted_Status2
1		0	(null)	(null)	OK	Good	0.0137190818786621	0.986280918121338	Bad Debt
2		557.67	22	Rent	Bad Debt	Bad Debt	0.34006404876709	0.65993595123291	Bad Debt
3		0	37	Own Home	OK	Good	0.015311598777771	0.984688401222229	Bad Debt
4		0	25	Other	OK	Good	0.00458037853240967	0.99541962146759	Bad Debt
5		0	81	(null)	OK	Good	0.00433135032653809	0.995668649673462	Bad Debt
6		0	(null)	(null)	OK	Good	0.0137190818786621	0.986280918121338	Bad Debt
7		0	(null)	(null)	OK	Good	0.0137190818786621	0.986280918121338	Bad Debt
8		0	62	(null)	OK	Good	0.00529772043228149	0.994702279567719	Bad Debt
9		0	67	(null)	OK	Good	0.00277310609817505	0.997226893901825	Bad Debt
10		0	30	Other	OK	Good	0.00255227088928223	0.997447729110718	Bad Debt
11		0	20	Own Home	OK	Good	0.0865201354026794	0.913479864597321	Bad Debt
12		0	61	Own Home	OK	Good	0.00155198574066162	0.998448014259338	Bad Debt
13		0	35	Rent	OK	Good	0.0592827796936035	0.940717220306396	Bad Debt

Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

See also: Sklearn XGBoost Model

Sklearn XGB Validation

Validation of an Sklearn XGBoost model is performed using the **Model Validation** node available the **Evaluate** palette. For general information about validation, see the topic About Validation.

Prerequisites for Sklearn XGB model validation: An Sklearn XGB model node and a dataset with the dependent variable and independent variables required by the model.

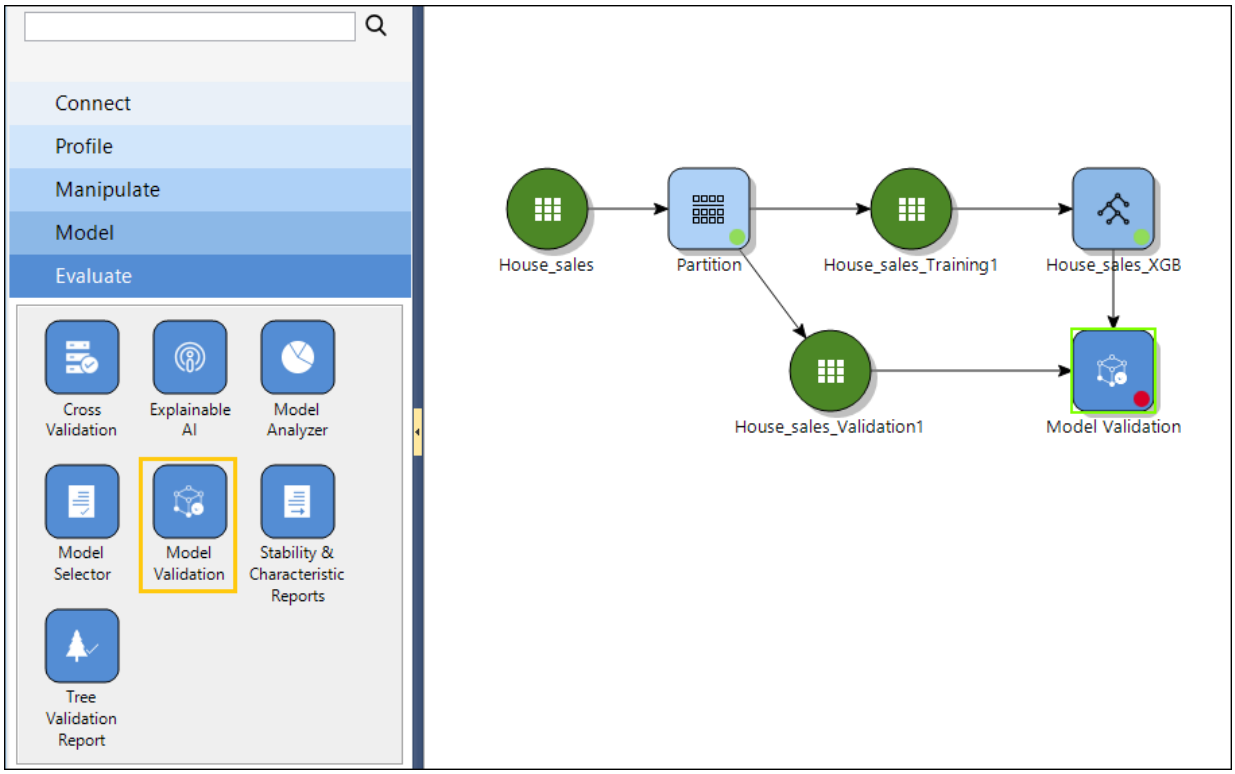
To add a Model Validation operation to your workflow:

- Drag the **Model Validation** node from the **Evaluate** palette onto the Workflow canvas and connect the inputs: an **Sklearn XGB** node and a dataset node that contains the data to validate.

  OR

- Right-click anywhere on the workflow canvas and select the context menu command **Evaluate --> Model Validation**. This will create a new Model Validation node. Connect the inputs: an **Sklearn XGB** model node and a dataset node that contains the data to validate.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

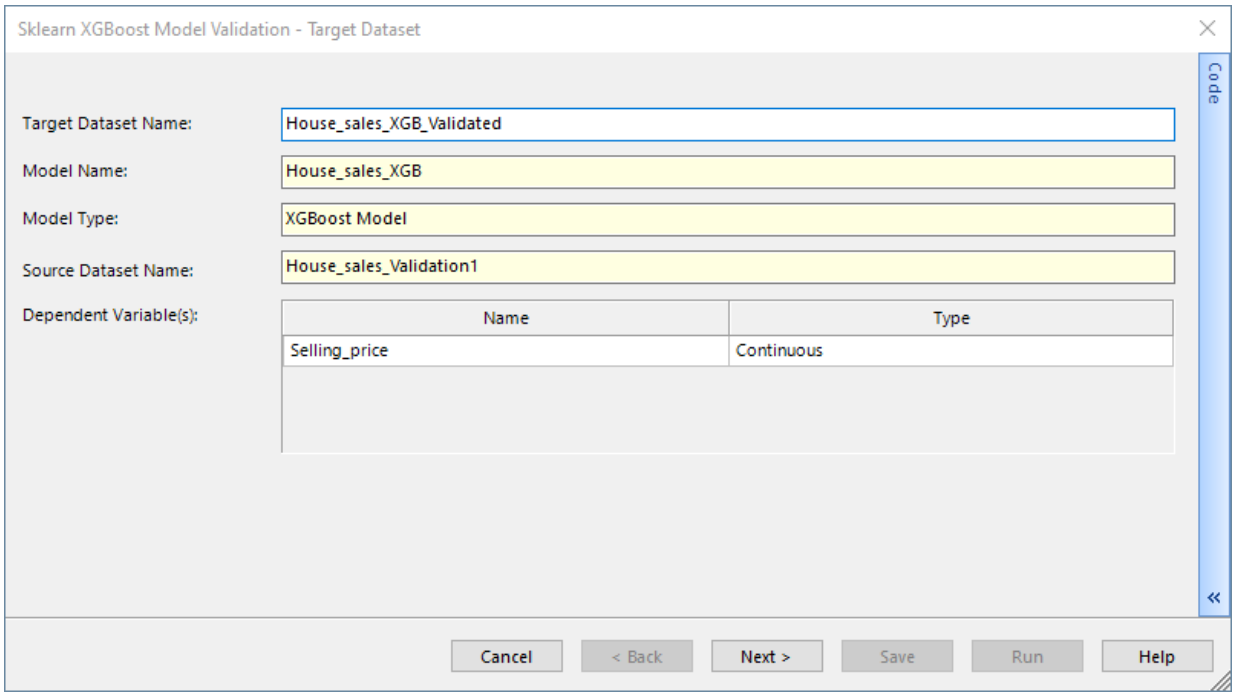


Double-click the **Model Validation** node or right-click on it and select **Modify** from the context menu.

The Sklearn XGB Validation wizard opens. It displays the names of the input model and dataset, the name and type of the dependent variable (*Continuous* or *Discrete*).

In the *Target Dataset Name* field, specify the name for the output dataset that will contain the validation data.

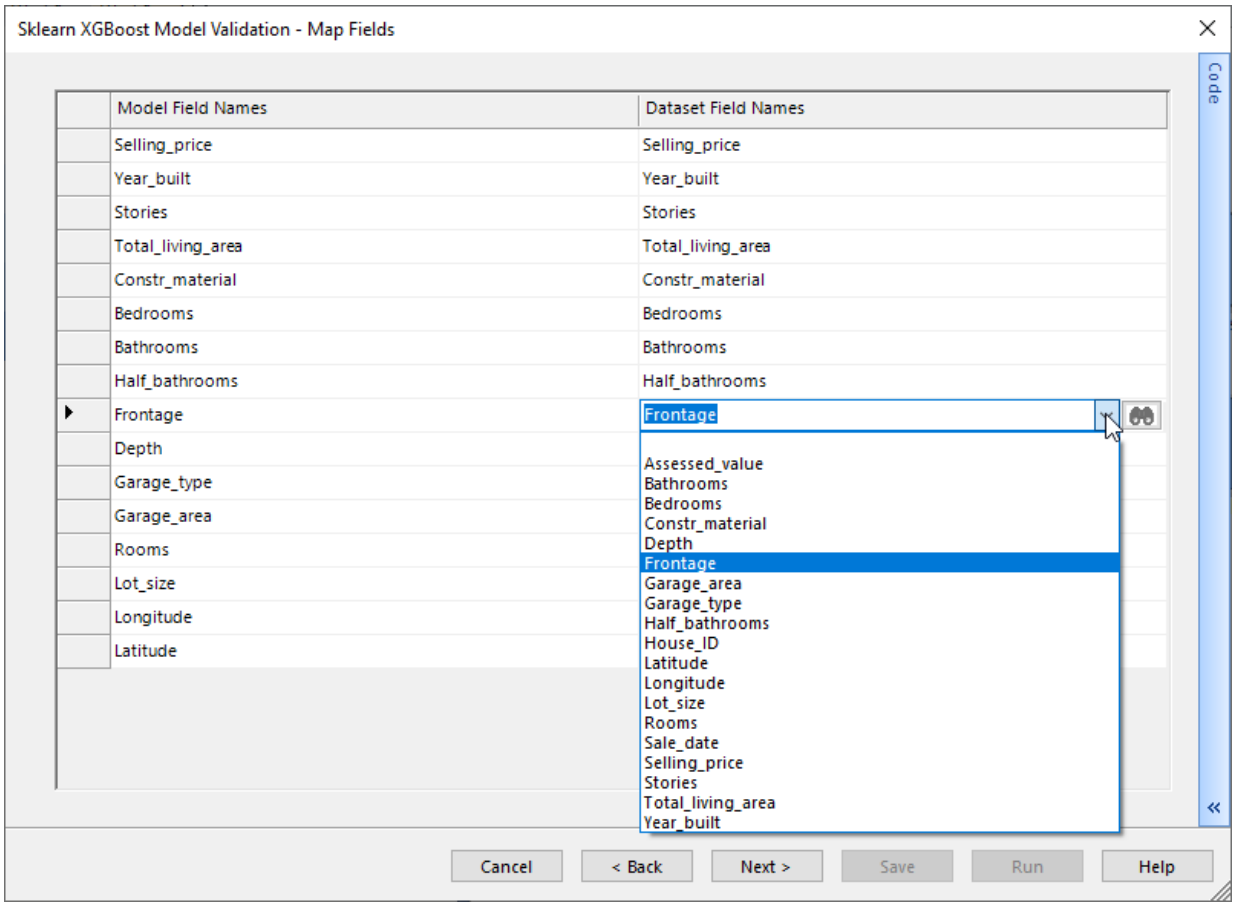
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

The next step provides the option of mapping the field names in the model to the ones in the validation input dataset. In some cases, the names of model variables may not match the names of the corresponding fields in the input dataset, so you will need to manually map fields by clicking in the right column (Dataset Field Names) and selecting the right fields to match the model variables in the left column:

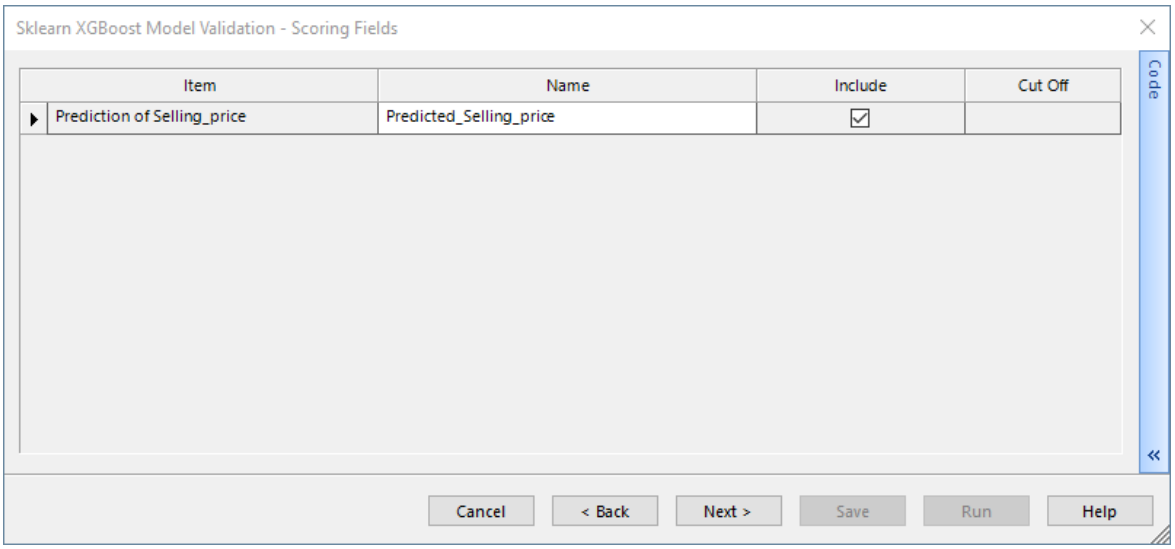
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Next**.

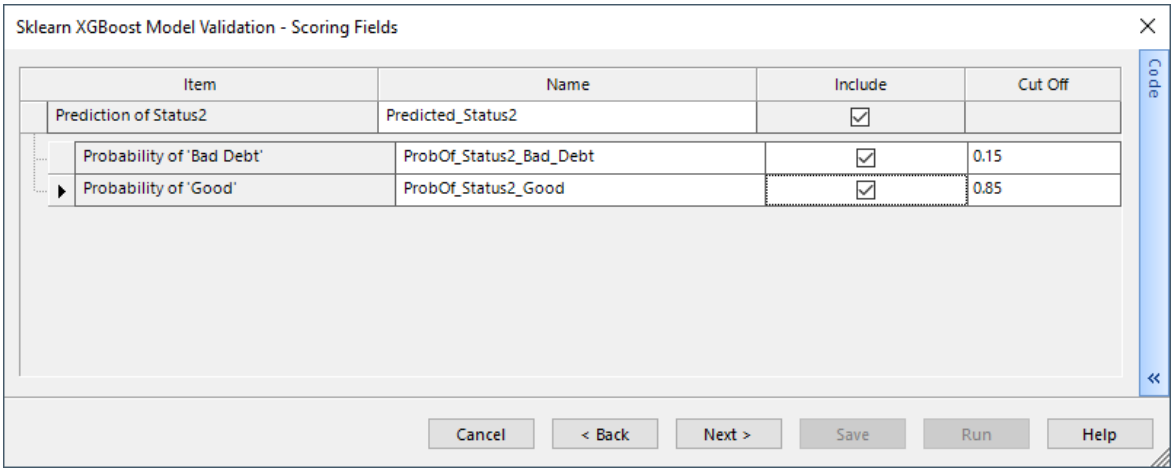
In the **Scoring Fields** page, review the scoring fields to be produced by the model and modify their names if necessary..

Example 1: Scoring fields for a model with a continuous dependent variable:



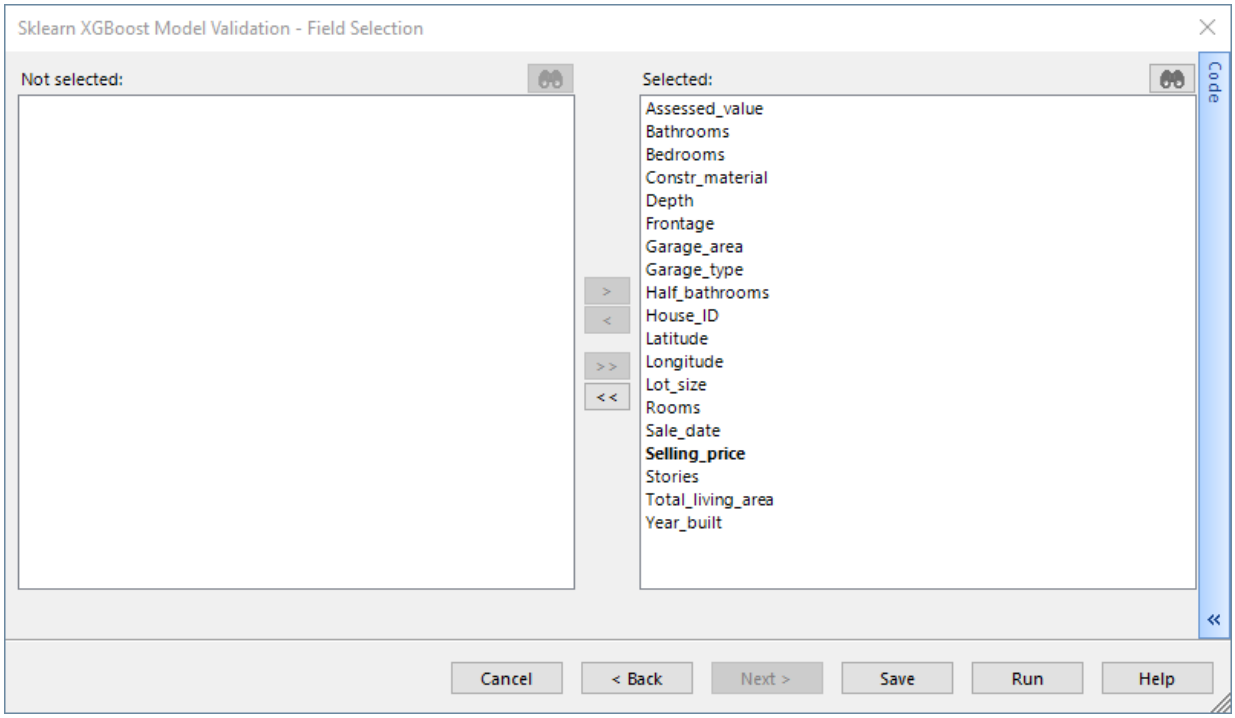
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Example 2: Scoring fields for a model with a discrete (categorical) dependent variable. If the dependent variable is binary, specify the cut-off value for the probability of the target outcome to classify the predictions.



Click **Next**.

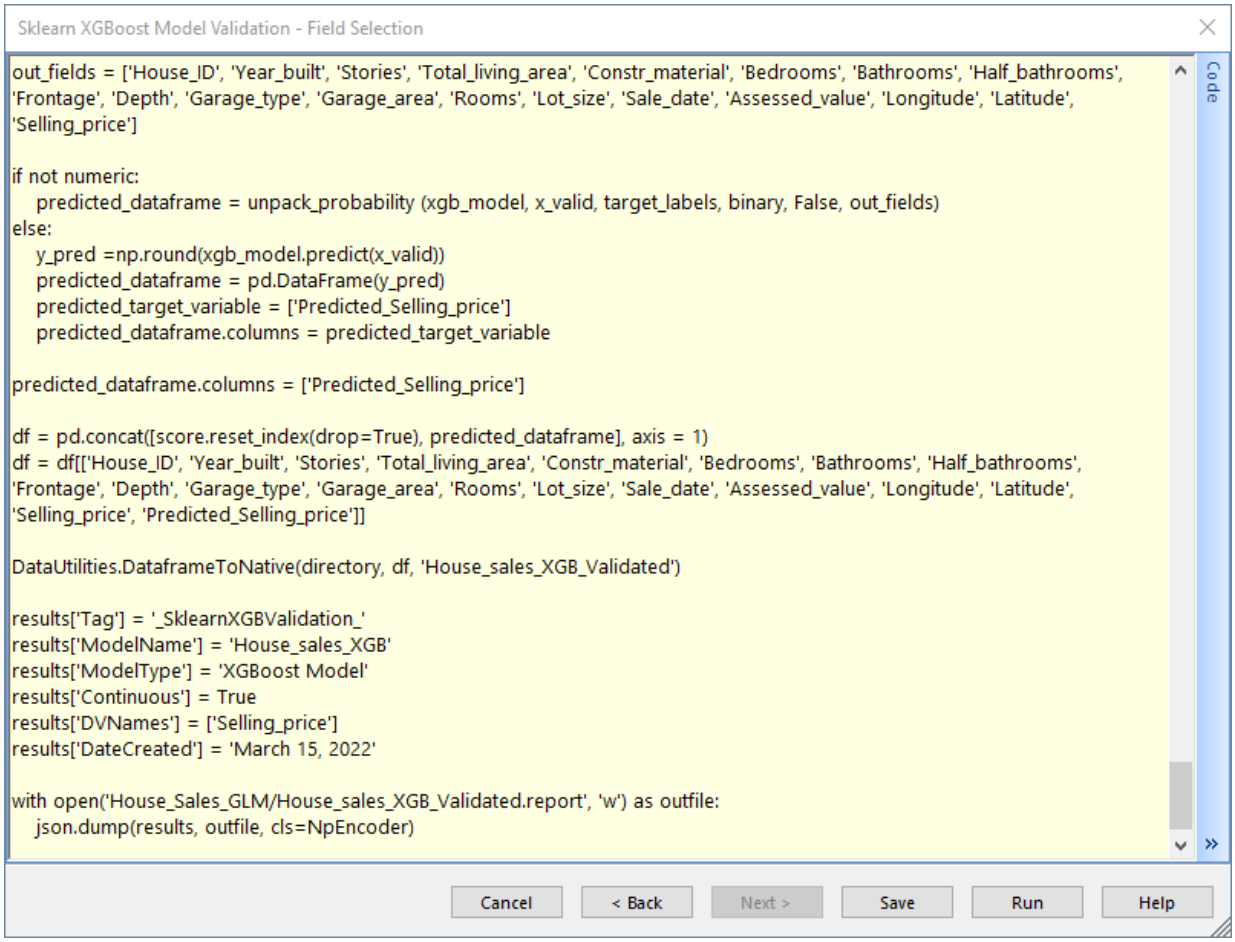
In the **Field Selection** page, choose the fields from the input dataset to include in the validation output in addition to the new scoring fields. The fields required by the model are highlighted in bold font and cannot be excluded. By default, all fields in the validation input dataset are selected to be included in the output.



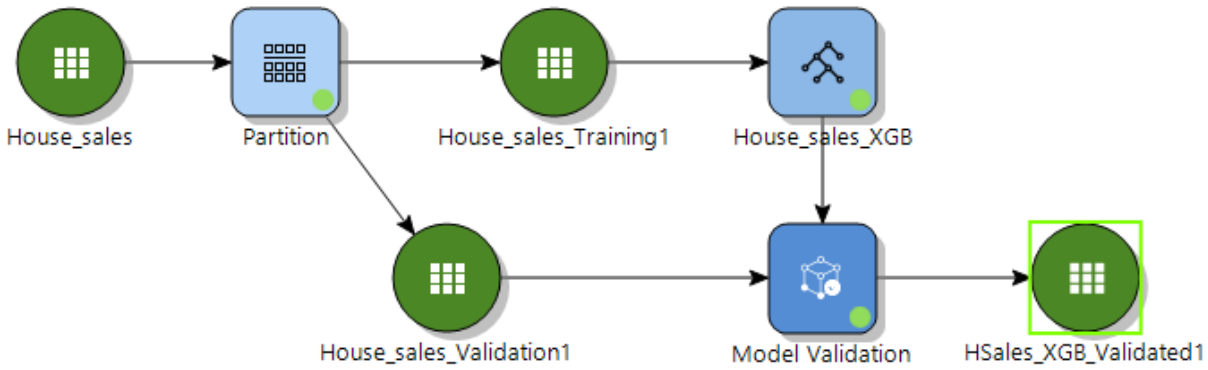
To preview the Python code generated for your current inputs at any point, click anywhere on the **Code** bar - the blue vertical bar on the right side of the dialog. This opens the Code Preview panel. When viewing the code, you can increase the font size by scrolling the mouse wheel while holding

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

down the CTRL key on the keyboard. Click on the blue Code bar again to close the code preview panel.



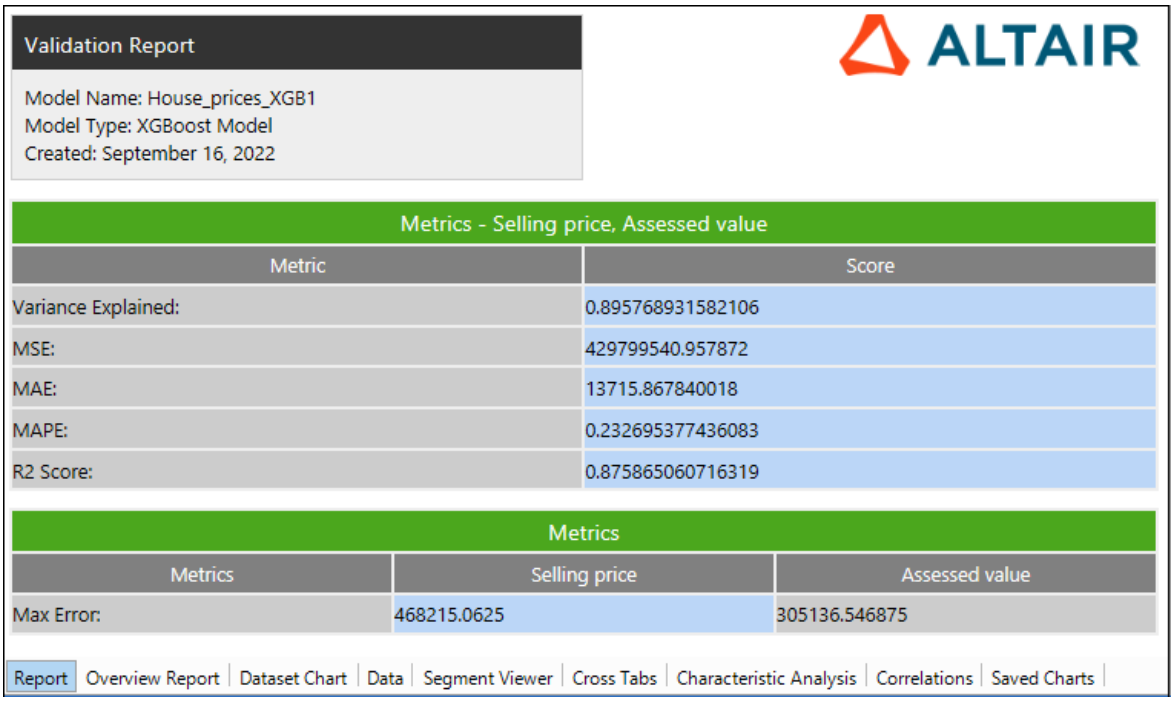
Click **Save** to save the settings, or click **Run** to save the settings and run the validation process immediately. The model is validated in a built-in Python environment, then the dataframe with generated probabilities / scores is converted back from Pandas to Altair KDD format. The output dataset node is created in the workflow. Double-click the output dataset to view the results.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The first tab of the output dataset is the Validation Report. It displays the values of the model evaluation metrics.

Example 1: Validation Report for a regression model with two continuous dependent variables.



Validation Report

Model Name: House_prices_XGB1
Model Type: XGBoost Model
Created: September 16, 2022

ALTAIR

Metrics - Selling price, Assessed value

Metric	Score
Variance Explained:	0.895768931582106
MSE:	429799540.957872
MAE:	13715.867840018
MAPE:	0.232695377436083
R2 Score:	0.875865060716319

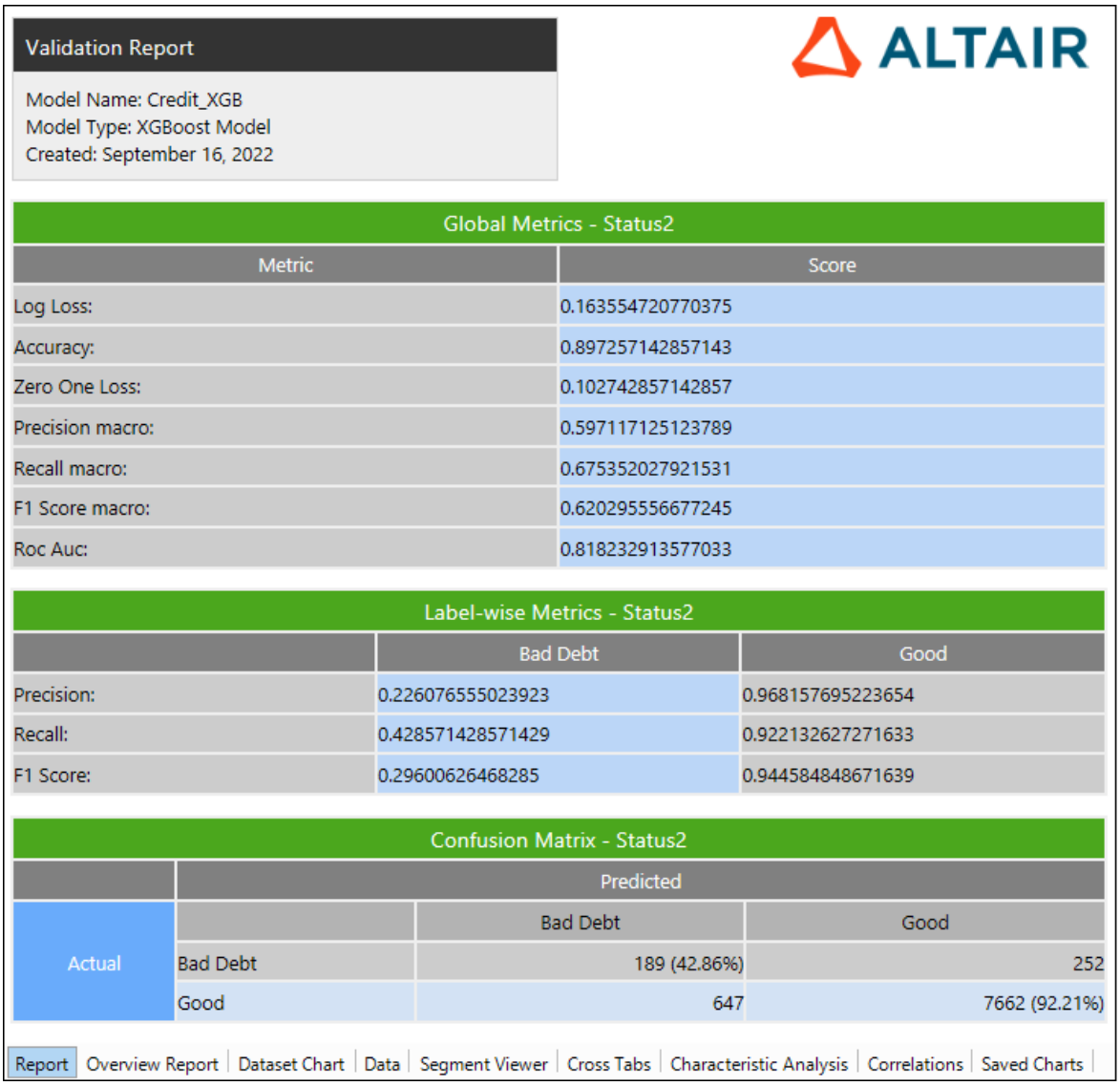
Metrics

Metrics	Selling price	Assessed value
Max Error:	468215.0625	305136.546875

Report | Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

Example 2: Validation Report for a classification model with a single discrete dependent variable.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Validation Report

Model Name: Credit_XGB
Model Type: XGBoost Model
Created: September 16, 2022

ALTAIR

Global Metrics - Status2

Metric	Score
Log Loss:	0.163554720770375
Accuracy:	0.897257142857143
Zero One Loss:	0.102742857142857
Precision macro:	0.597117125123789
Recall macro:	0.675352027921531
F1 Score macro:	0.620295556677245
Roc Auc:	0.818232913577033

Label-wise Metrics - Status2

	Bad Debt	Good
Precision:	0.226076555023923	0.968157695223654
Recall:	0.428571428571429	0.922132627271633
F1 Score:	0.29600626468285	0.944584848671639

Confusion Matrix - Status2

		Predicted	
		Bad Debt	Good
Actual	Bad Debt	189 (42.86%)	252
	Good	647	7662 (92.21%)

Report | Overview Report | Dataset Chart | Data | Segment Viewer | Cross Tabs | Characteristic Analysis | Correlations | Saved Charts

The output dataset can be connected to a Model Analyzer node to further analyze the model performance.

See also: Sklearn XGBoost Model, Model Analyzer.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Integration with Tableau and Qlik

Integration with Tableau

Knowledge Seeker and Knowledge Studio support integration with Tableau® - business intelligence software for interactive data visualization. Supported Tableau versions are 2019.1 and higher.

A standard feature in Knowledge Seeker and Knowledge Studio, integration with Tableau Server and Tableau Desktop allows users to create and publish workbooks for Tableau dashboards and share them throughout the organization from within the application workflow. Workbooks from Tableau Server dashboards can be downloaded to Knowledge Seeker and Knowledge Studio and incorporated into workflows.

These features enrich the reporting capabilities and provide greater flexibility and freedom for analysts to combine advanced predictive analytics with a popular business intelligence tool widely recognized for its outstanding data visualization and intuitive interface.

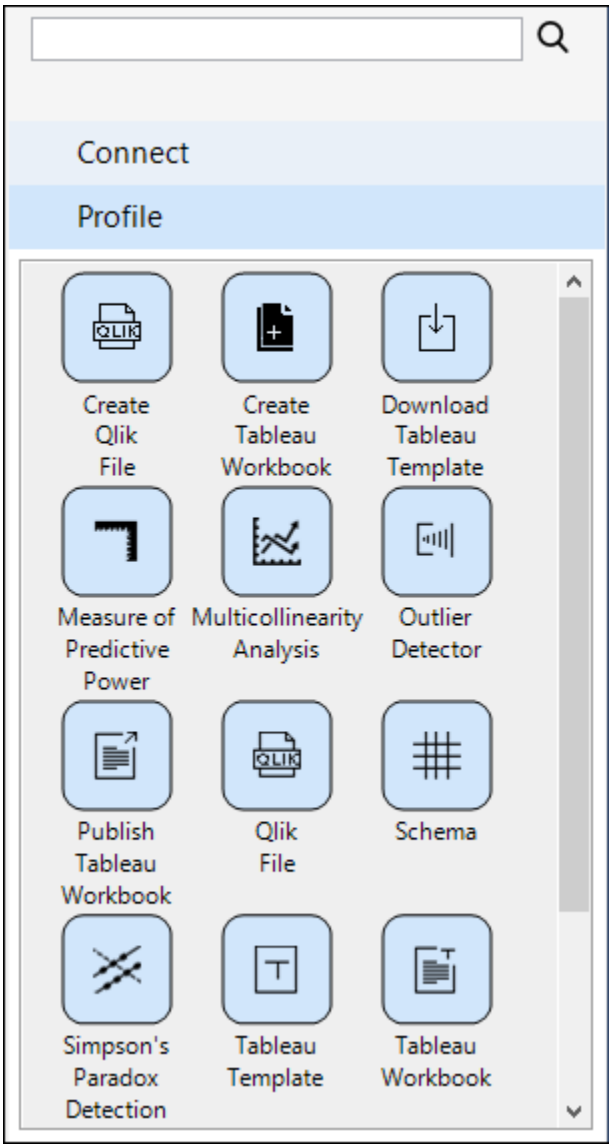
If you are not a user of Tableau software and would like to know more about its features, please visit https://www.tableau.com/.

The following capabilities are provided for Tableau users:

- Include Tableau workbook files and workbook templates in workflows
- Create new Tableau workbooks from datasets
- Import Tableau workbooks and templates directly from a Tableau Server in workflows
- Publish data sources and Tableau workbooks from a Knowledge Seeker or Knowledge Studio project to a Tableau Server

The Tableau integration features are available in the **Profile** palette. There are five functional nodes available: Tableau Template, Tableau Workbook, Download Tableau Template, Create Tableau Workbook, and Publish Tableau Workbook. These functions are described in the corresponding Help topics.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Proceed to Creating a Tableau Workbook

See also:

- Adding Tableau Workbooks and Templates to Workflow
- Downloading Tableau Templates
- Publishing a Tableau Workbook

Creating a Tableau Workbook

Tableau Workbooks can be created from any dataset node. A Tableau Template can optionally be used in this process to define columns for the workbook.

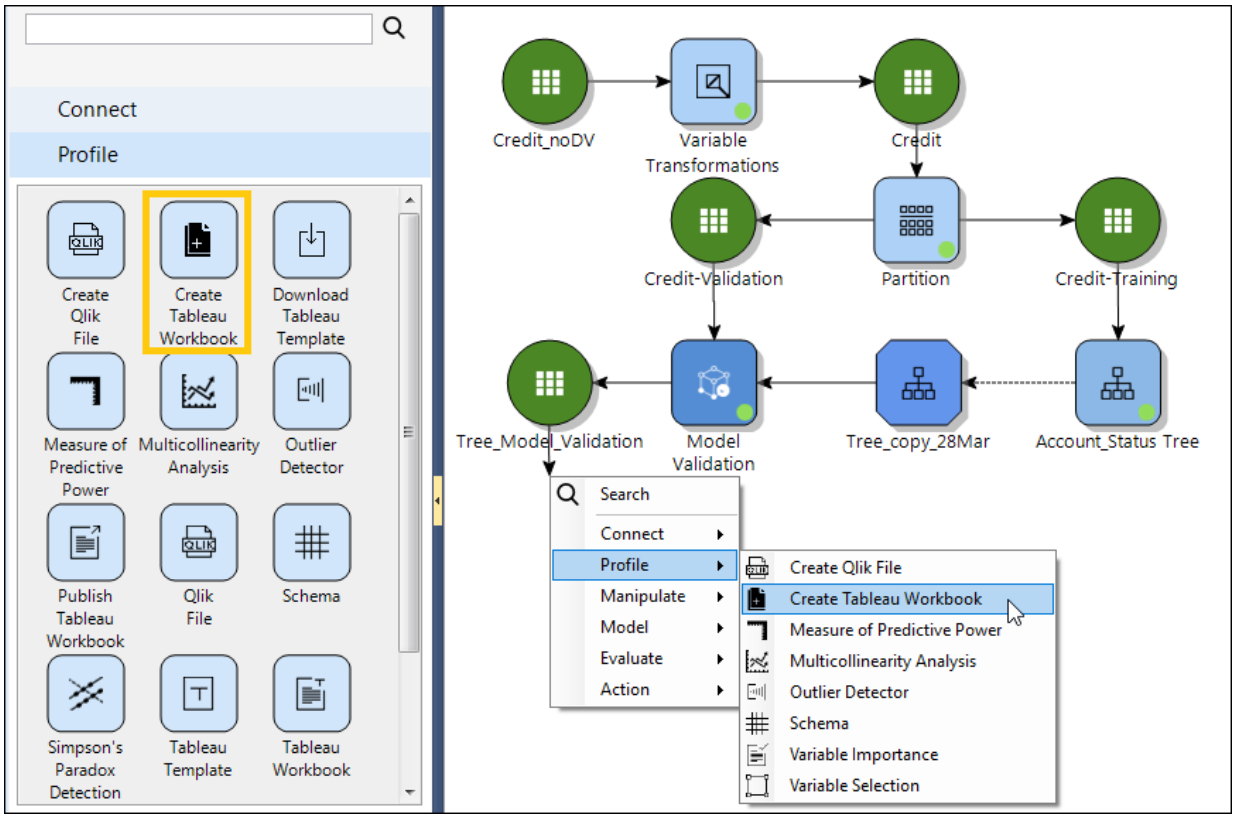
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To create a Tableau Workbook from a dataset:

- Drag a connection arrow from the dataset node to any area on the Workflow canvas background. This brings up a context menu that allows you to select the desired operation. From the context menu, select **Profile --> Create Tableau Workbook**, as shown in the example below.

OR:

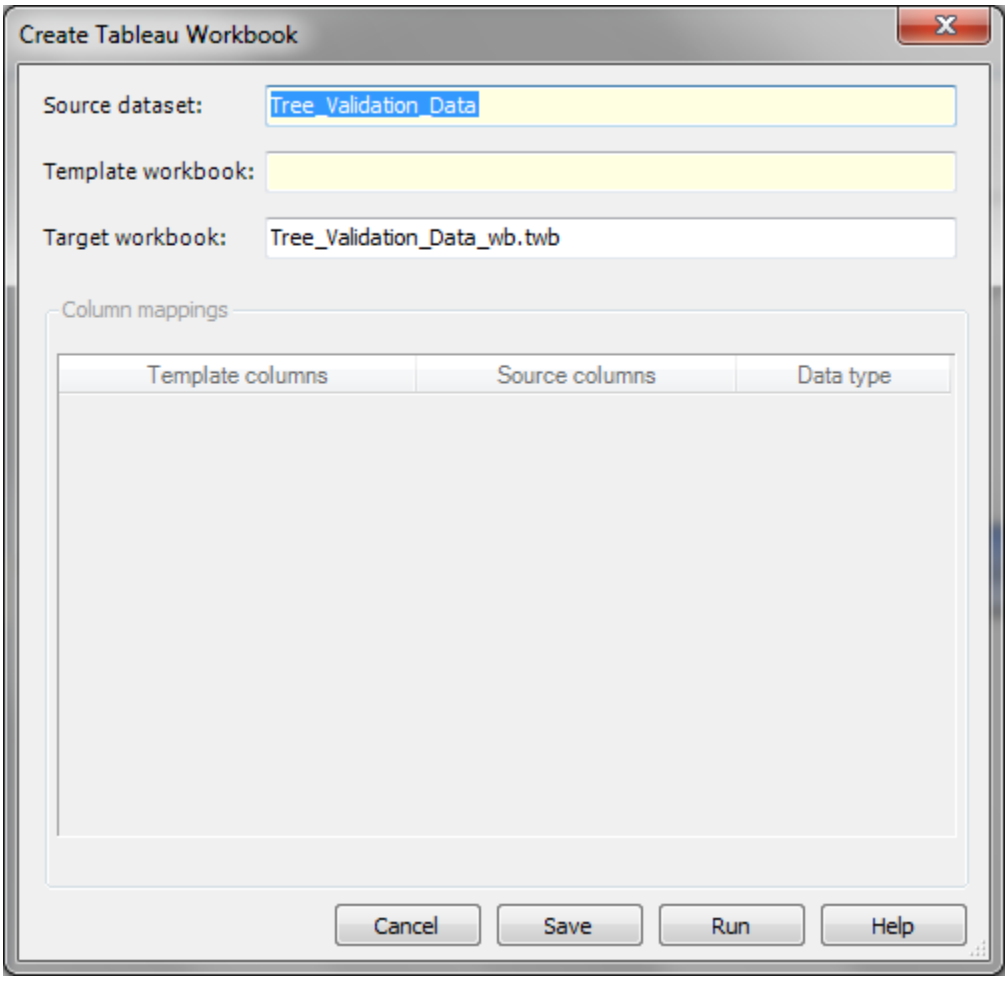
- Drag the **Create Tableau Workbook** node from the **Profile** palette onto the canvas and connect the input dataset to this node.



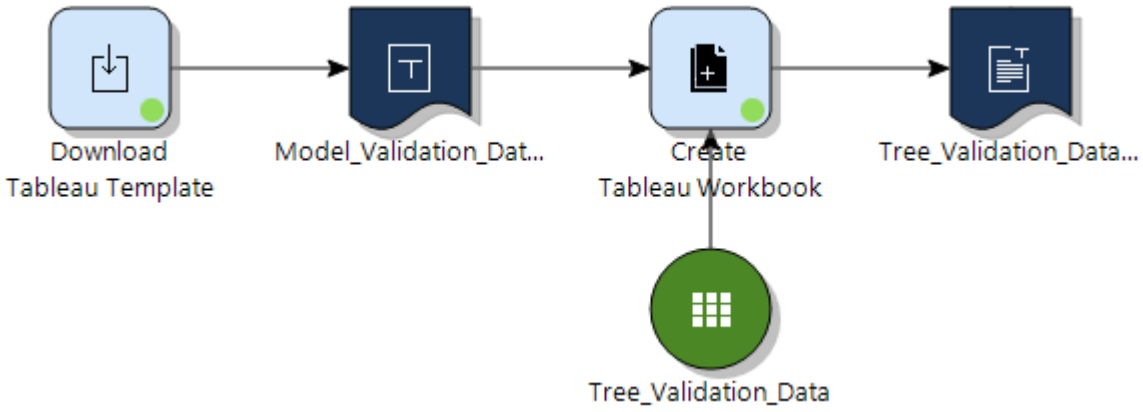
Double-click the **Create Tableau Workbook** node or right-click on it and select **Modify** to open the wizard.

In the wizard, specify the name for the output Tableau workbook.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

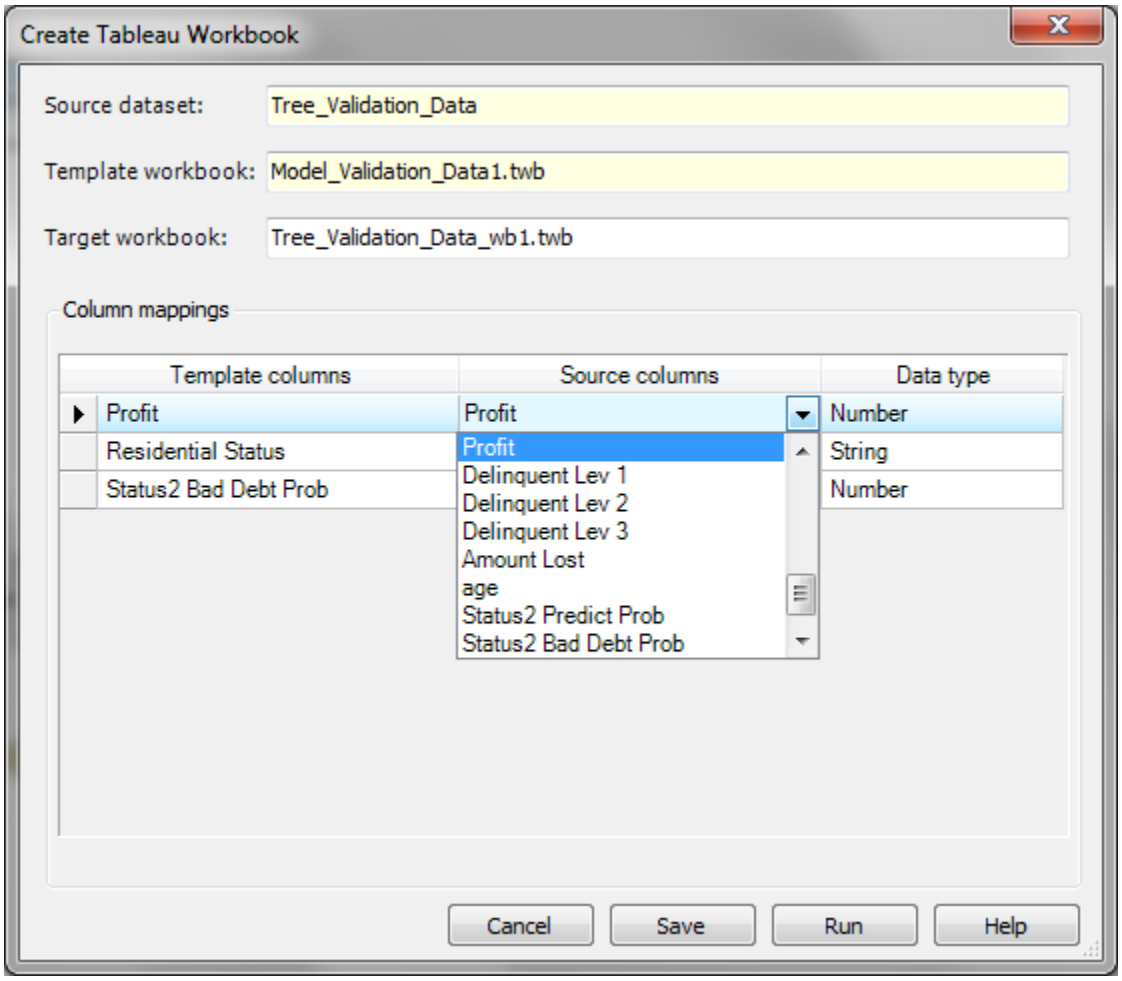


You can also use an existing Tableau Template node as an input, as shown in the example below. In that case the resulting workbook will use the columns in the template.

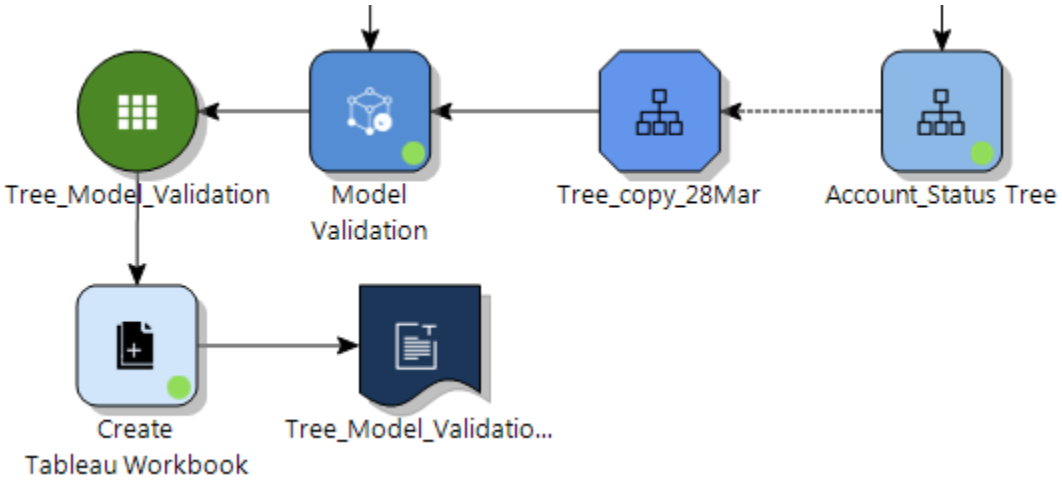


If you connected a Tableau Template to the Create Tableau Workbook node, map the template columns to the source dataset columns if necessary. To select a column, click on the desired column name in *Source columns:*

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



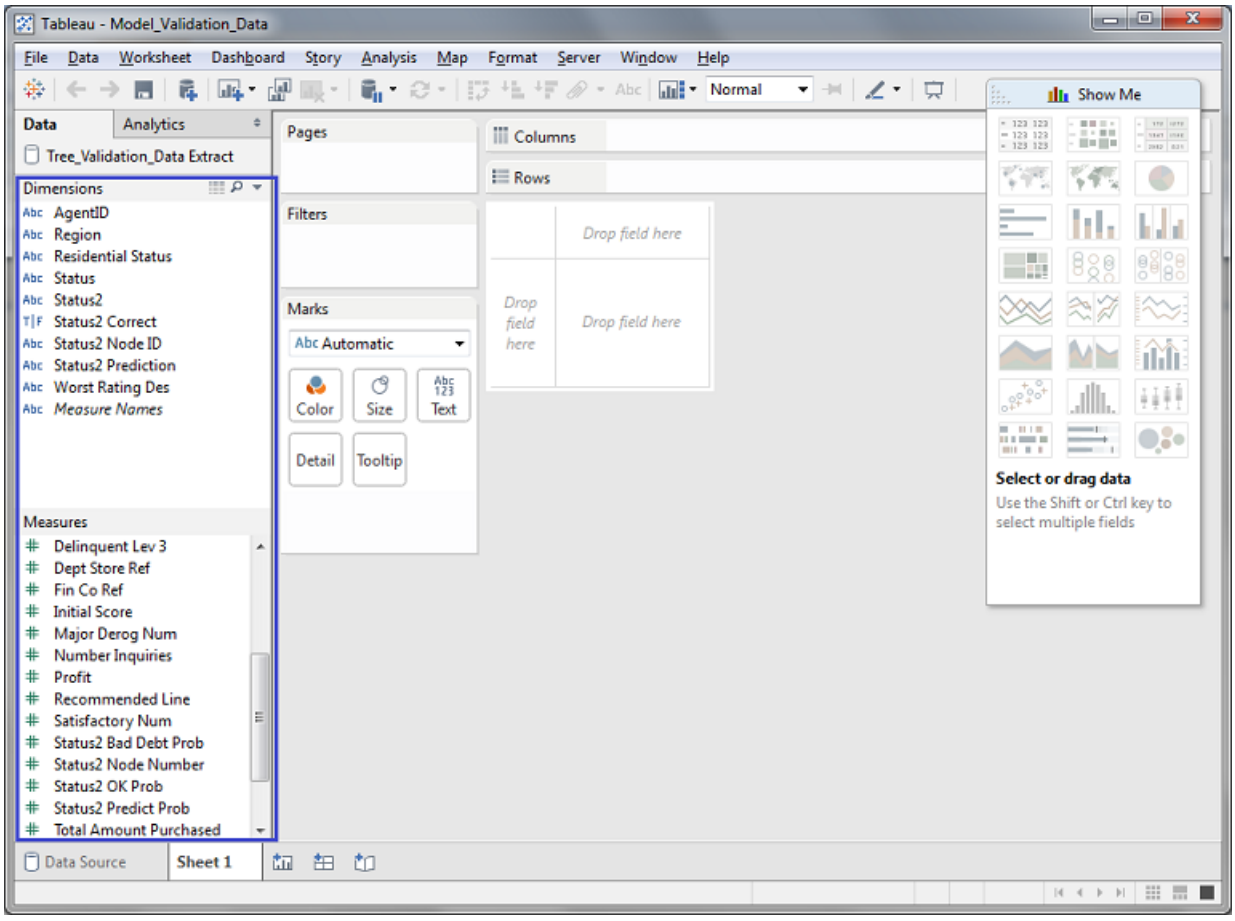
Click **Save** to save the settings and close the node, or click **Run** to save the settings and run the process immediately. The output Tableau Workbook node is created the workflow.



Double-click the output Tableau Workbook node to open it in Tableau Desktop. The variables of the input dataset are displayed in the left panels. Here you can build the desired charts and graphs

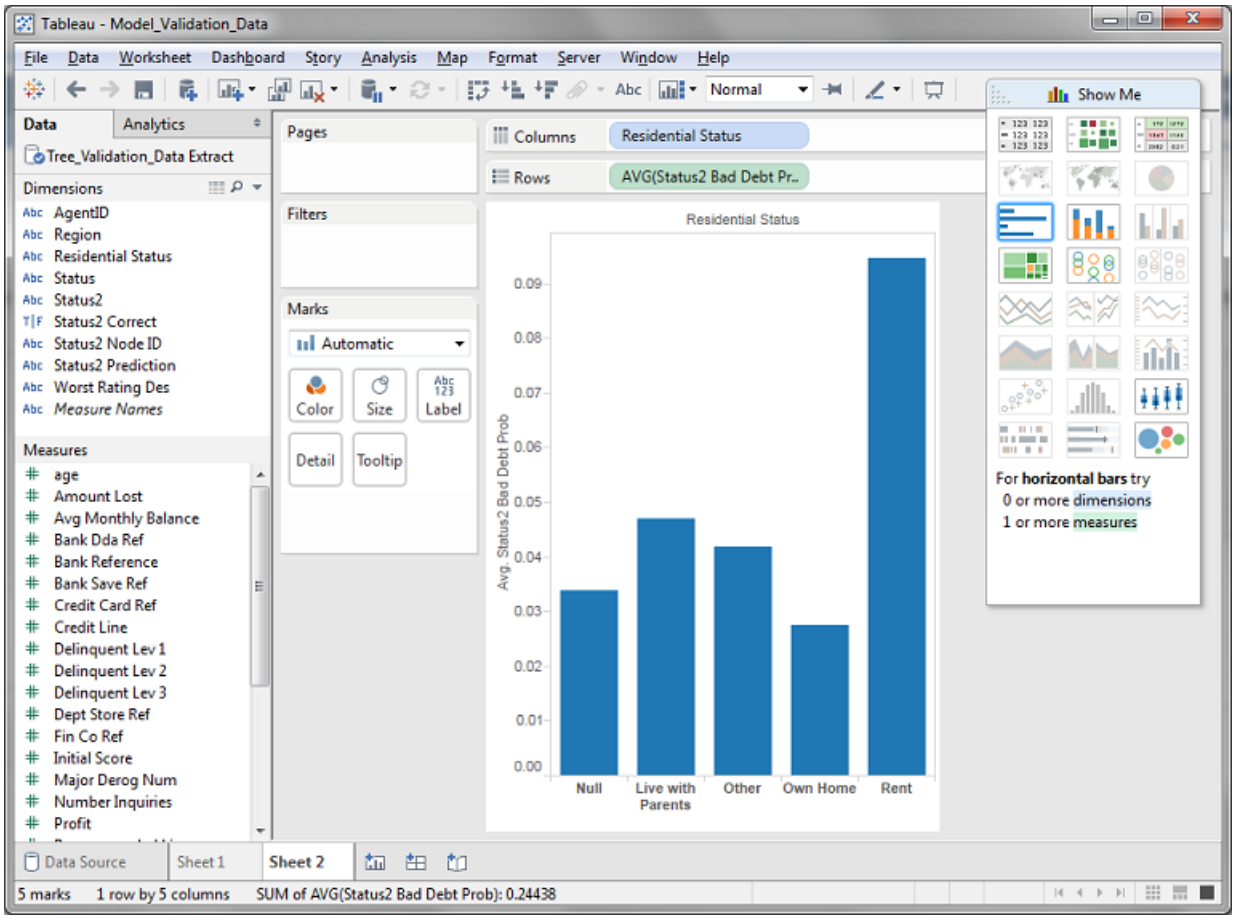
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

and use all the diverse visualization features of Tableau. Refer to the Tableau documentation for details on working with Tableau Workbooks in Tableau Desktop.

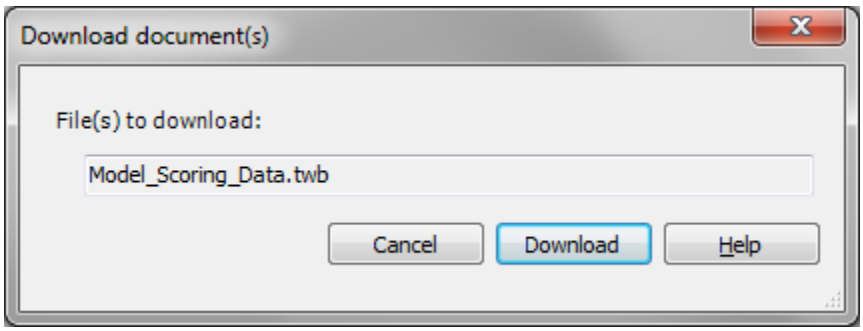


In the following example, a simple chart is created using two variables (Average *Bad Debt Probability* depending on *Residential Status*).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If you double-click on a Tableau Workbook node in the client/server mode, you are prompted to download the workbook to the local drive, and then it will automatically open with Tableau Desktop or other Tableau software on your PC.



See also:

Integration with Tableau: Overview

Adding Tableau Workbooks and Templates to Workflow

Downloading Tableau Templates

Publishing a Tableau Workbook

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Adding Tableau Workbooks and Templates to Workflow

Using Link nodes, you can add Tableau Workbook files or Template files to the Workflow.

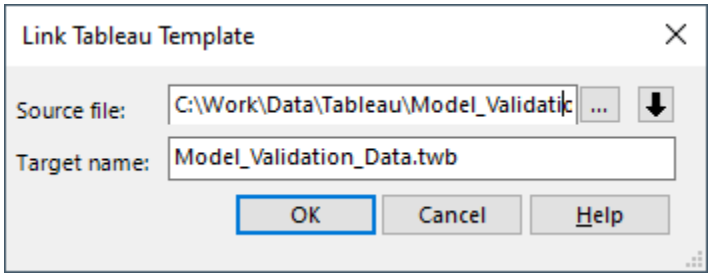
To add an existing Tableau Workbook or Template file to your Workflow:

- Right-click anywhere on the Workflow canvas background. This brings up a context menu with applicable operations. From the context menu, select **Profile --> Tableau Workbook** (or **Tableau Template**), as shown in the example below. A new Tableau Workbook or Template node will be created and the link dialog will open.

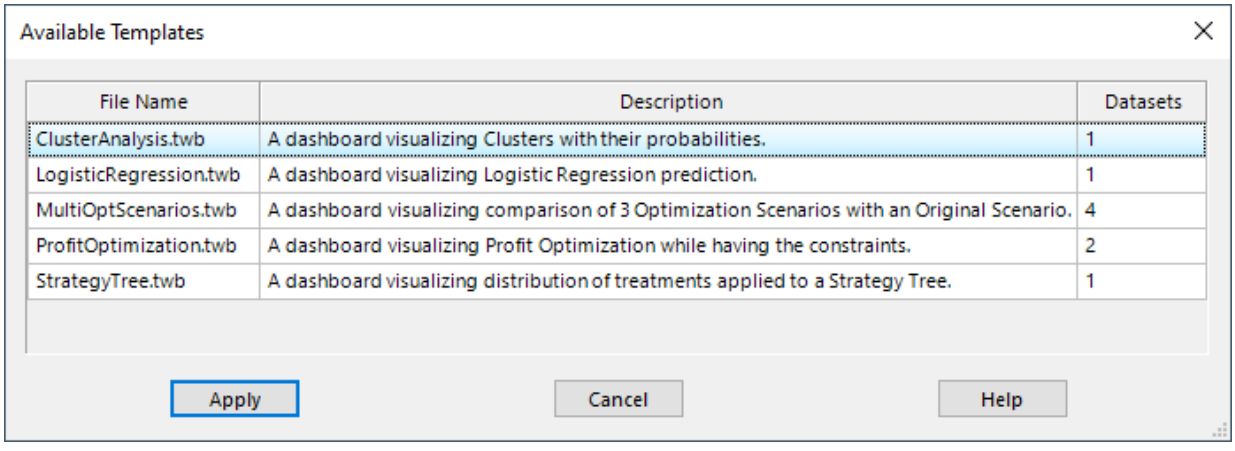
OR:

- Drag the **Tableau Workbook** or **Tableau Template** node from the **Profile** palette onto the canvas.

Double-click the new node or right-click on it and select **Modify** to open the link dialog. Browse for the desired Tableau Workbook file or Template file or type the file path in the **Source File** field.



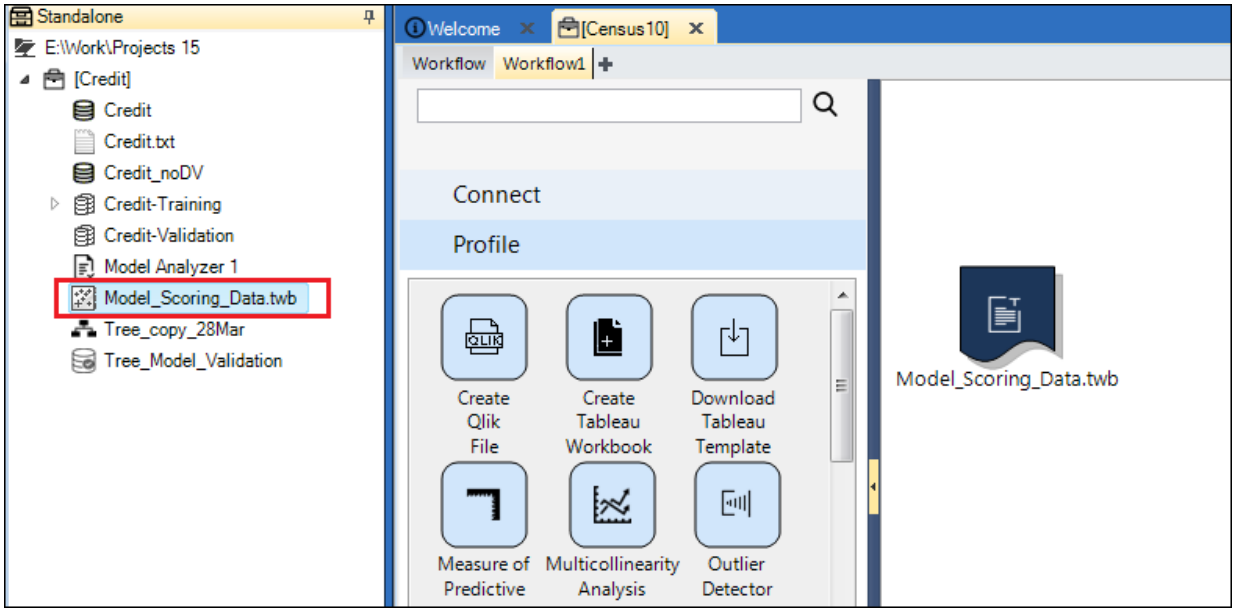
Alternatively, click the downward arrow on the right if you want to add one of the available example templates provided with your Altair application. This brings up the Available Templates dialog, where you can select the desired template and click **Apply**.



Click **OK** in the Link Tableau Template dialog. The new Tableau Workbook or Template node is now linked to the actual file. The linking operation copies the source workbook or template file to the current project. If it was not already in the project, the linked workbook or template will now be displayed in the Project Explorer. You can open it by double-clicking the node in the workflow or

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

the corresponding object in the Project Explorer. The workbook or template will be opened in Tableau Desktop (if it is installed on your PC).



Once you have the Tableau Workflow or Template node on the canvas, you can use it in the workflow. To see the list of operations defined for these node types, drag a connection from the node to the Workflow canvas background and select the desired command from the context menu.

See also:

Integration with Tableau: Overview

Creating a Tableau Workbook

Publishing a Tableau Workbook

Downloading Tableau Templates

Publishing a Tableau Workbook

A Tableau Workbook can be uploaded to a Tableau Server. Supported Tableau Server versions are 2019.1 and higher.

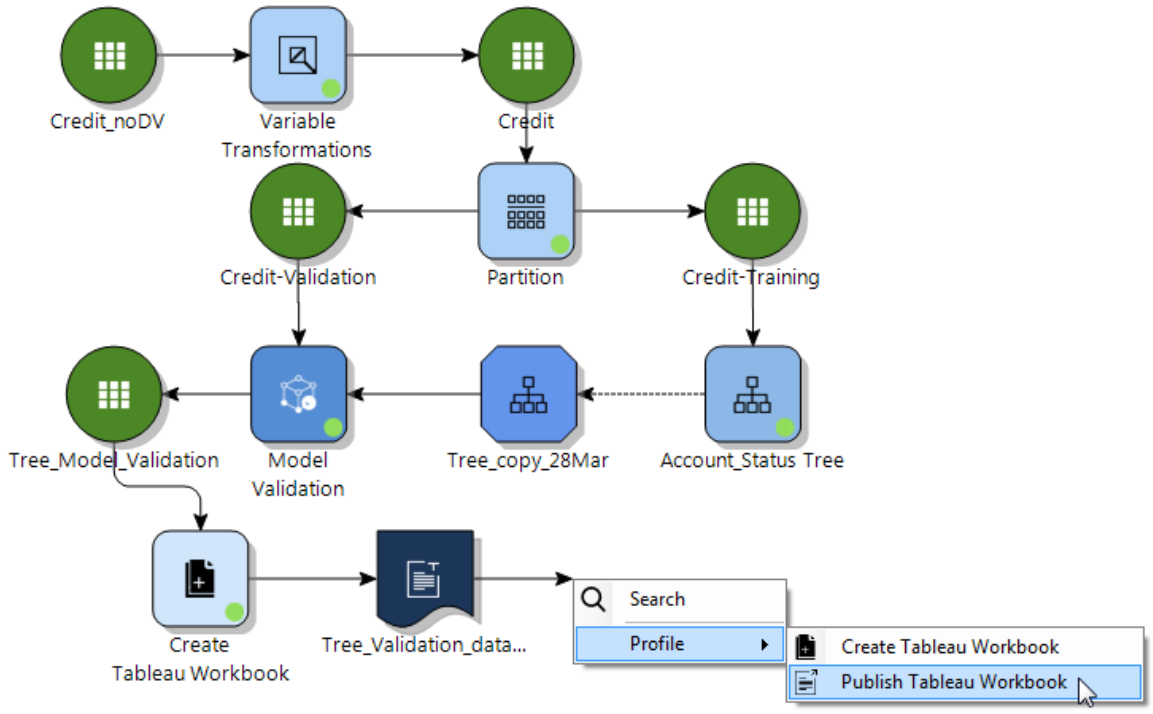
To publish a Tableau Workbook to a Tableau Server:

- Drag a connection arrow from the **Tableau Workbook** node to any area on the Workflow canvas background. This brings up a context menu with applicable operations. From the context menu, select **Profile --> Publish Tableau Workbook**, as shown in the example below. A new Publish Tableau Workbook node will be created.

OR:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

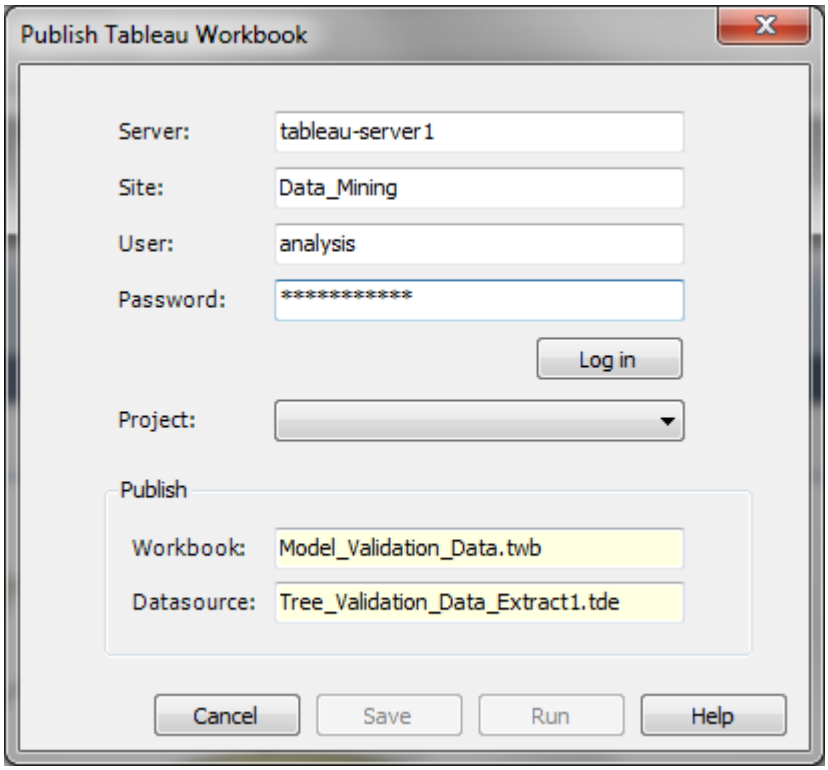
- Drag the **Publish Tableau Workbook** node from the **Profile** palette onto the canvas and connect the input **Tableau Workbook** node to it.



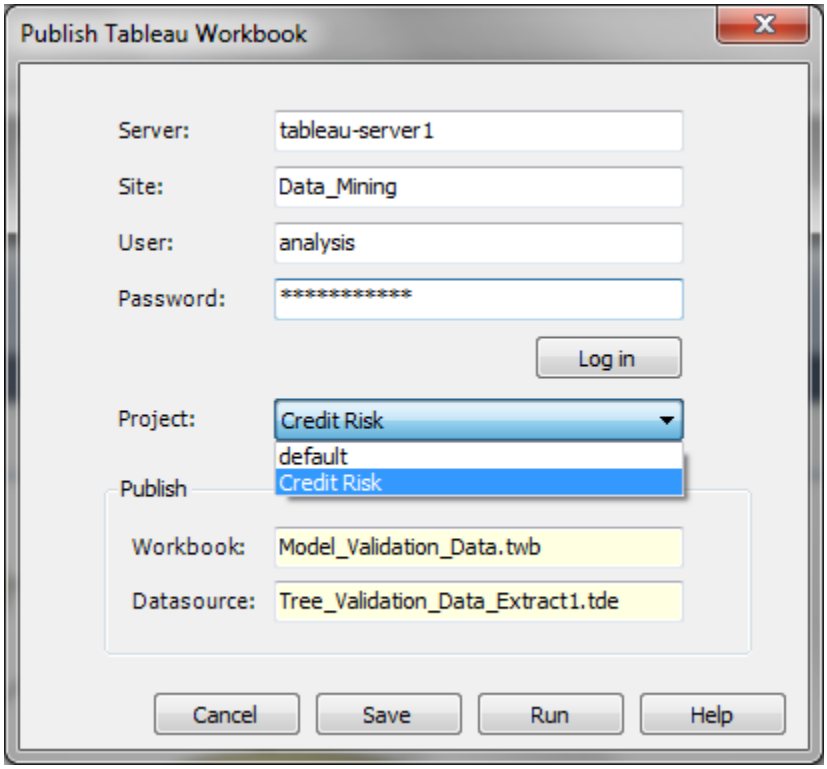
Double-click the **Publish Tableau Workbook** node or right-click on it and select **Modify** to open the wizard.

- Specify the name of the destination Tableau Server. The “http://” prefix can be omitted.
- Specify the Site that contains the destination project for the workbook.
- Specify your Tableau Server user name and password.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Once you have specified the right credentials, either click the **Log In** button or click on the **Project** drop-down list to choose the destination project for the workbook.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.

After you have published a workbook, you can log in to the Tableau Server by double-clicking the **Publish Tableau Workbook** node to check if the workbook and the data source extract appear in the desired project. Refer to the Tableau documentation and your organization guidelines for details on working with the Tableau Server.

See also:

- Integration with Tableau: Overview
- Creating a Tableau Workbook
- Downloading Tableau Templates
- Adding Tableau Workbooks and Templates to Workflow

Downloading Tableau Templates

Tableau Workbook Templates can be downloaded from Tableau Server. Supported Tableau Server versions are 2019.1 and higher.

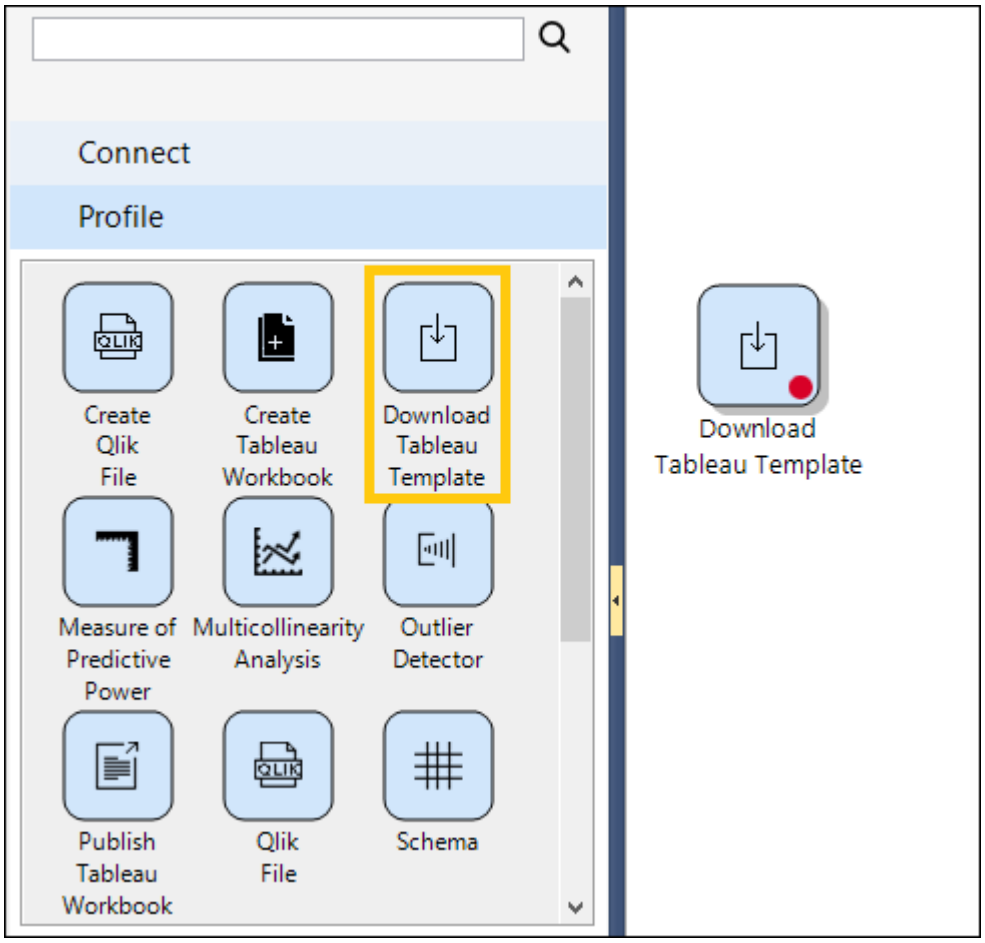
To download a Tableau Workbook Template from a Tableau Server:

- Right-click anywhere on the Workflow canvas background. This brings up a context menu with applicable operations. From the context menu, select **Profile --> Download Tableau Template**, as shown in the example below.

  OR:

- Drag the **Download Tableau Template** node from the **Profile** palette onto the canvas.

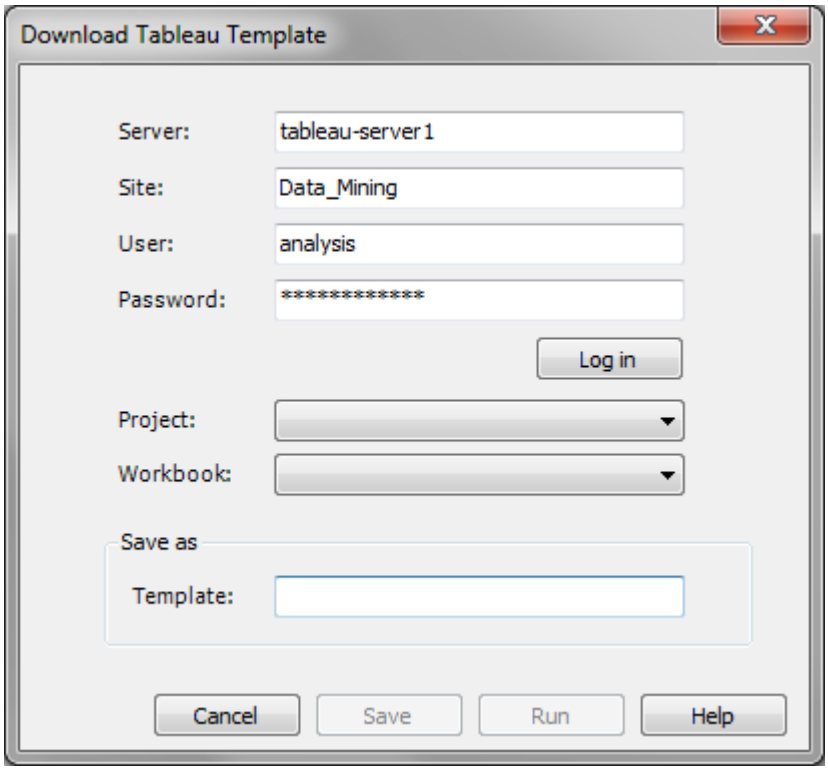
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If you used the context menu, a new node called **Download Tableau Template** will be crated and the download dialog will open. Otherwise, double-click the new node or right-click on it and select **Modify** to open the download dialog.

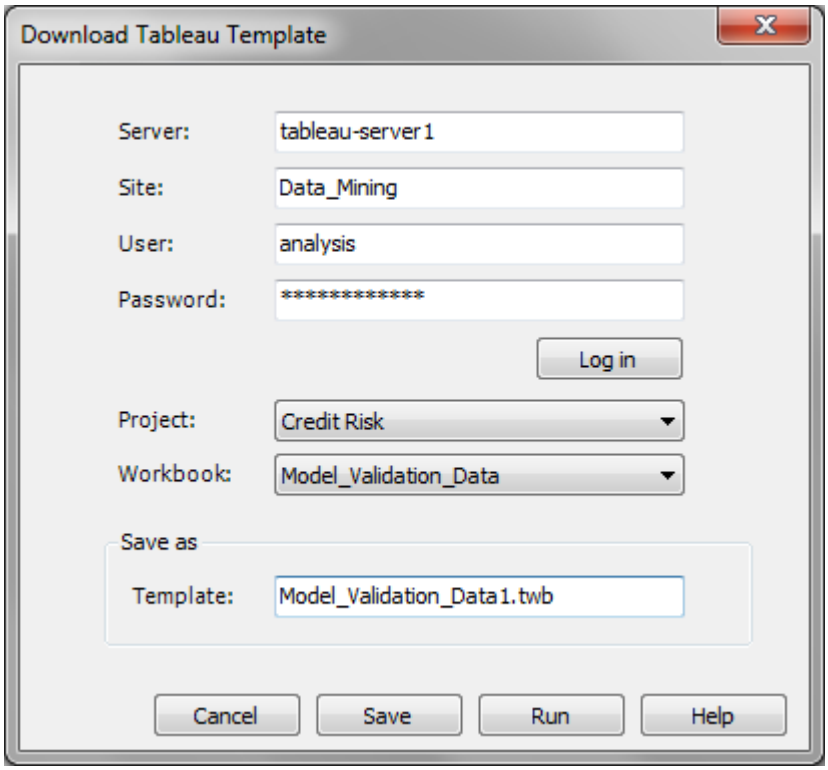
- Specify the name of the source Tableau Server. The “http://” prefix can be omitted.
- Specify the Site within the Tableau Server that contains the project with the desired template.
- Specify the Tableau Server user name and password.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Once you have specified the right credentials, either click the **Log In** button or click on the **Project** drop-down list to choose the destination project for the workbook. This will automatically populate the Project and Workbook lists, if the login is successful. Then select the project and the workbook whose template you would like to download. If necessary, change the name of the resulting template as it will appear in the Workflow and the Project Explorer.

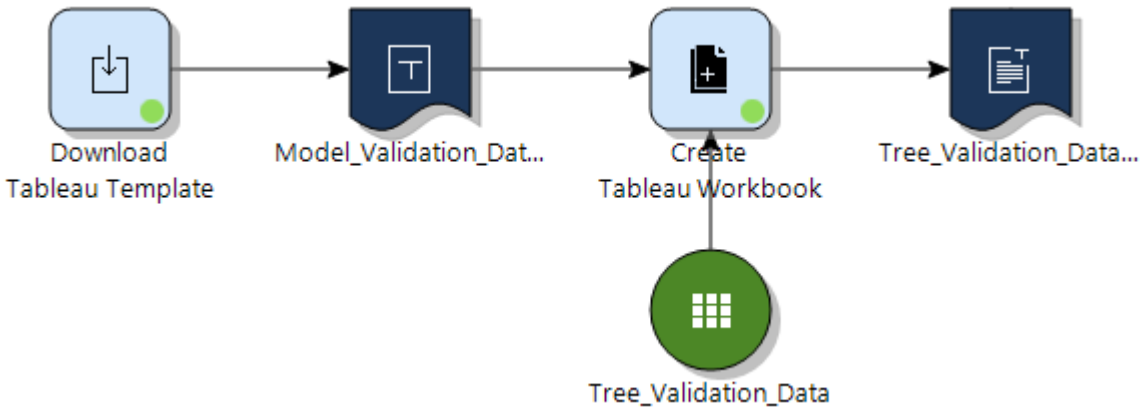
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.

A new template cannot be opened as an independent object. It is used to create other objects.

After you have run the node and downloaded a workbook template, you can use this template and a dataset in the workflow to create a new Tableau workbook, as shown below:



Refer to the Tableau documentation and your organization guidelines for details on working with the Tableau Server.

See also:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Integration with Tableau: Overview
Creating a Tableau Workbook
Publishing a Tableau Workbook
Adding Tableau Workbooks and Templates to Workflow

Integration with Qlik

Knowledge Studio and Knowledge Seeker include integration with Qlik® - business intelligence software for creating flexible, interactive visualizations to aid in decision-making. Integration with Qlik is supported in the desktop mode and in the client/server mode for Windows and Linux. It is not supported in the client/server mode for AIX.

A standard feature in Knowledge Seeker and Knowledge Studio, integration with Qlik allows users to create Qlik files within the workflows and embed existing Qlik files into the workflows. These features enrich the reporting capabilities and provide greater flexibility and freedom for analysts to combine advanced predictive analytics with a popular business intelligence tool known for its simple drag-and-drop interfaces for data exploration and smart visualizations automatically adapting to the user-defined parameters.

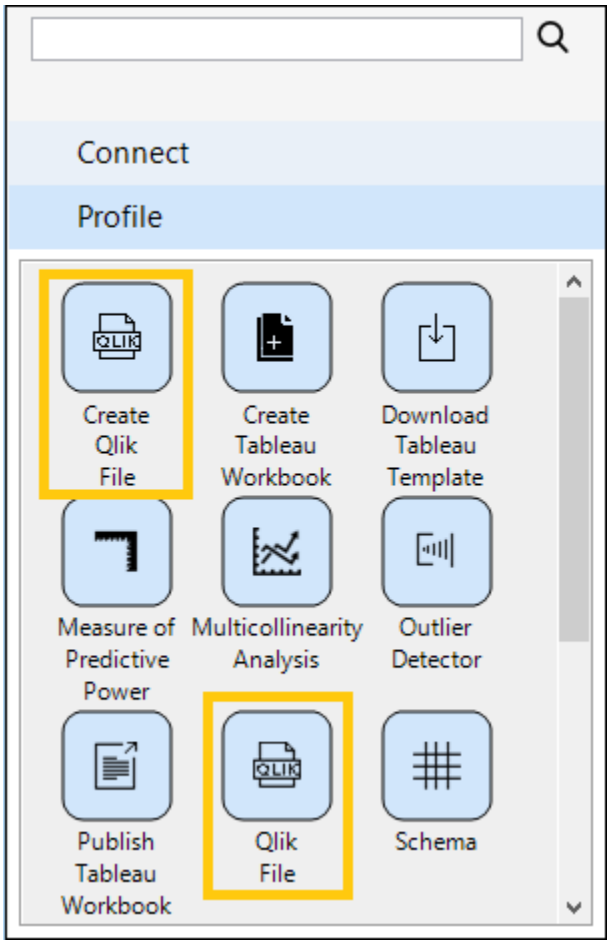
QlikView® or any other Qlik product that allows viewing Qlik table files (*.qvx) is a prerequisite. If you are not a user of Qlik software and would like to know more about its features, please visit https://www.qlik.com/us/products.

Knowledge Seeker and Knowledge Studio provide the following capabilities for Qlik users:

- Include existing Qlik files in workflows
- Create new Qlik files (*.qvx) from datasets and embed them in workflows.

The Qlik integration features are available in the **Profile** palette. There are two functional nodes available: Creating a Qlik File, and Qlik File. These functions are described in the corresponding Help topics.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Proceed to Creating a Qlik File

See also: Adding Qlik Files to Workflow

Creating a Qlik File

Qlik Files can be created from any dataset node.

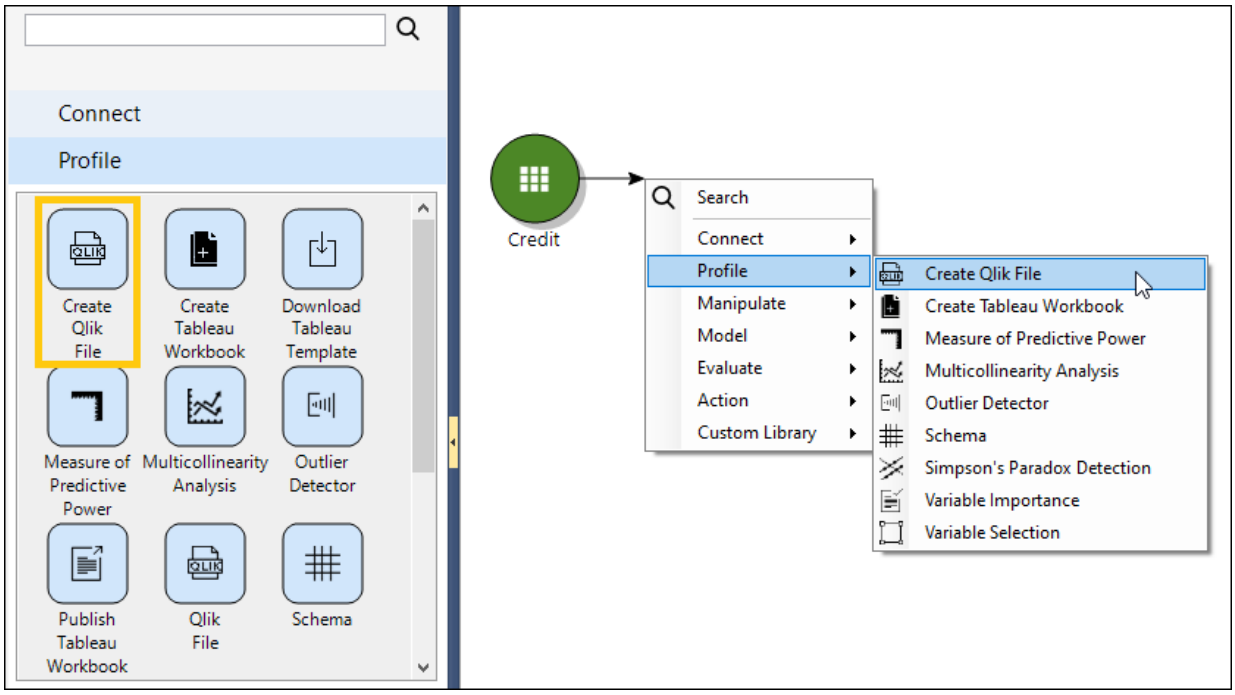
To create a Qlik File from a dataset:

- Drag a connection arrow from the dataset node to any area on the Workflow canvas background. This brings up a context menu that allows you to select the desired operation. From the context menu, select **Profile --> Create Qlik File**, as shown in the example below.

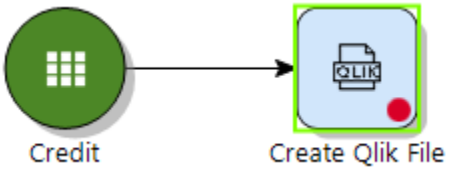
OR:

- Drag the **Create Qlik File** node from the **Profile** palette onto the canvas and connect the input dataset to this node.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

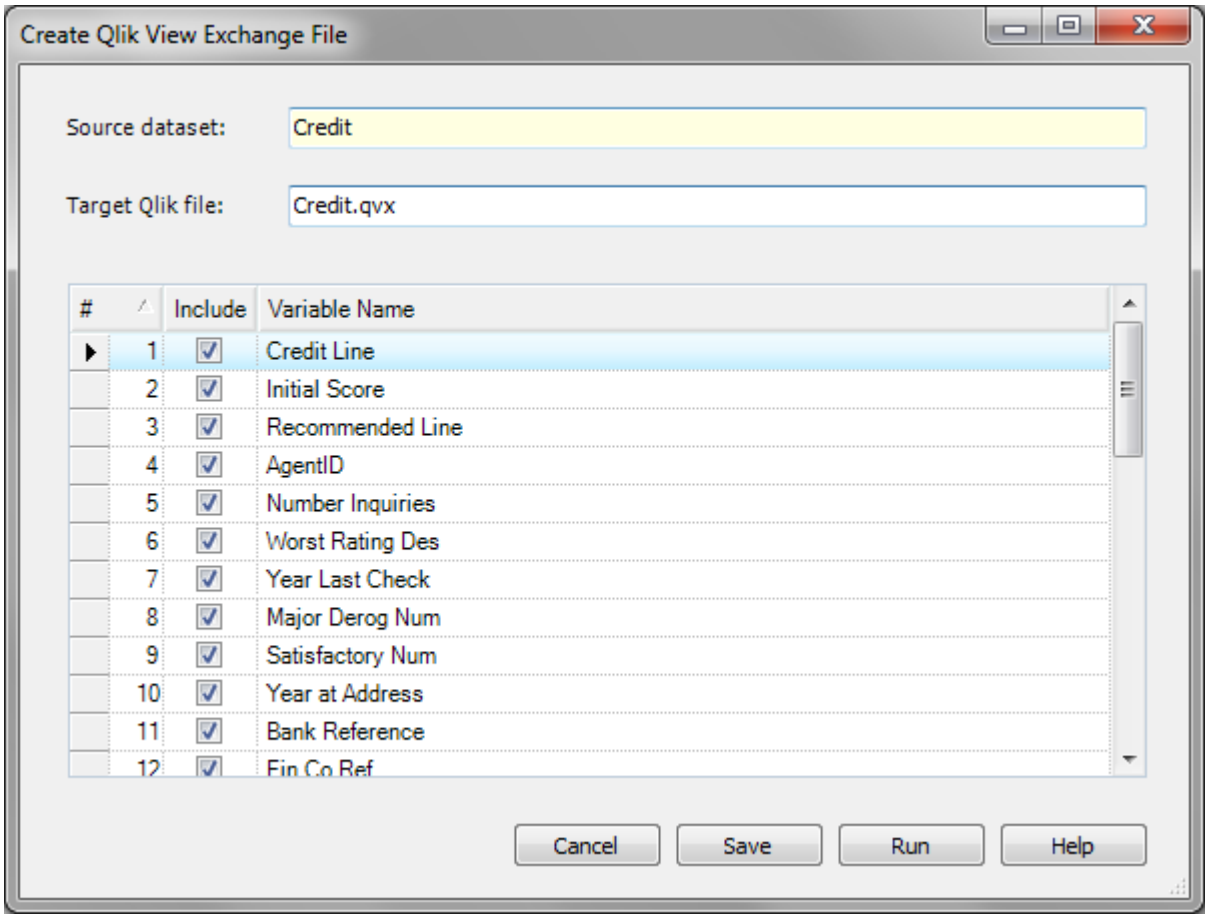


Double-click the **Create Qlik File** node or right-click on it and select **Modify** to open the wizard.

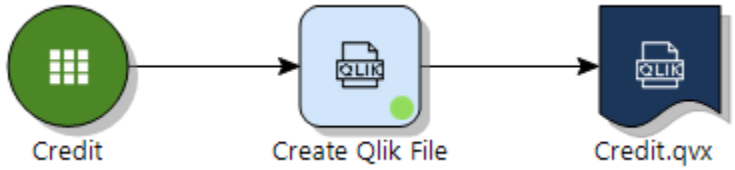


In the wizard, specify the name for the output Qlik File and select the variables to include in it. Note that according to the Qlik naming conventions, the file name must contain only English letters, digits, or underscores, and not start with a digit.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings and close the node, or click **Run** to save the settings and run the process immediately. The output Qlik File node is created the workflow.

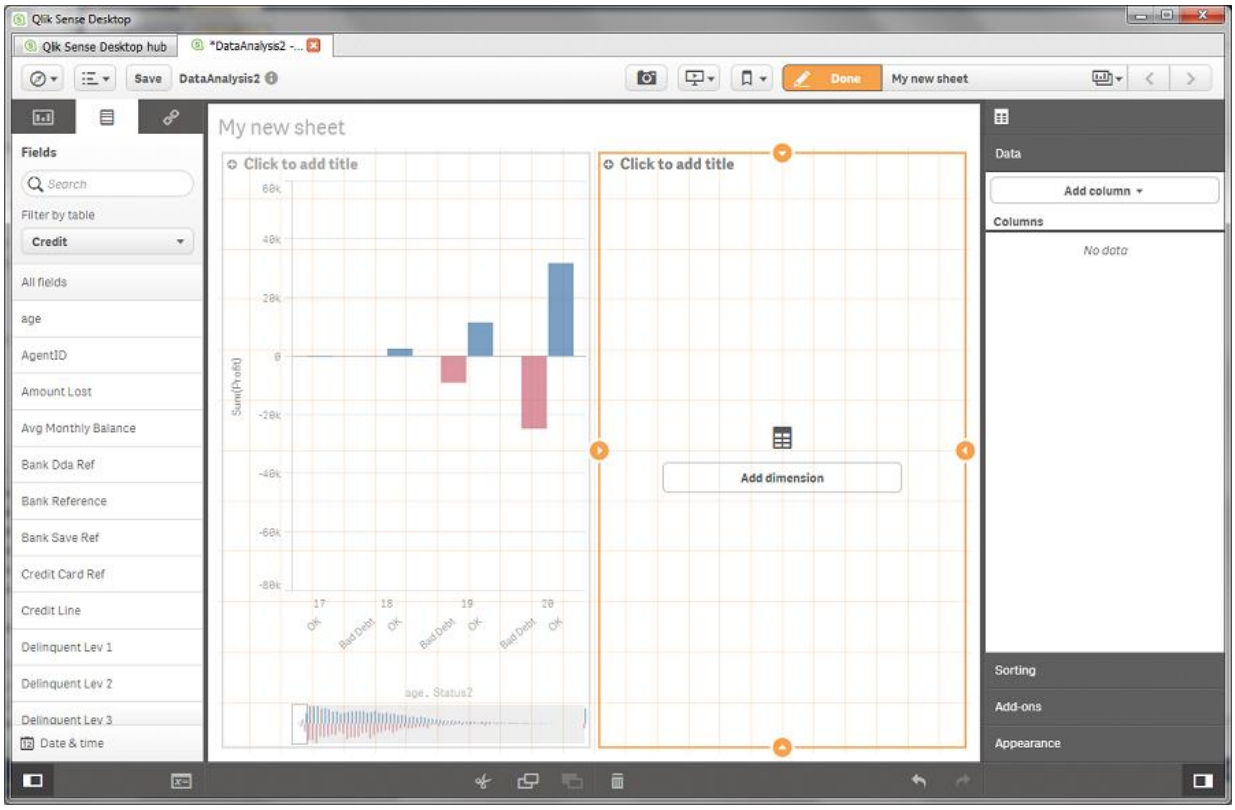


Double-click the output Qlik File node to open it in QlikView® or another Qlik application that can open *.qvx files.

*Note: If Qlik is not installed or *.qvx is not associated with Qlik on your PC, double-clicking on this node will open a standard Windows prompt asking you to choose the desired program to open this file.*

After loading the data, the variables and data of the file are displayed. Build the desired charts and graphs and use the available visualization features of Qlik. Refer to the Qlik documentation for details on working with Qlik files in Qlik applications.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



See also:

[Integration with Qlik: Overview]

[Adding Qlik Files to Workflow]

Adding Qlik Files to Workflow

Using Link nodes, you can add Qlik files to your project workflows.

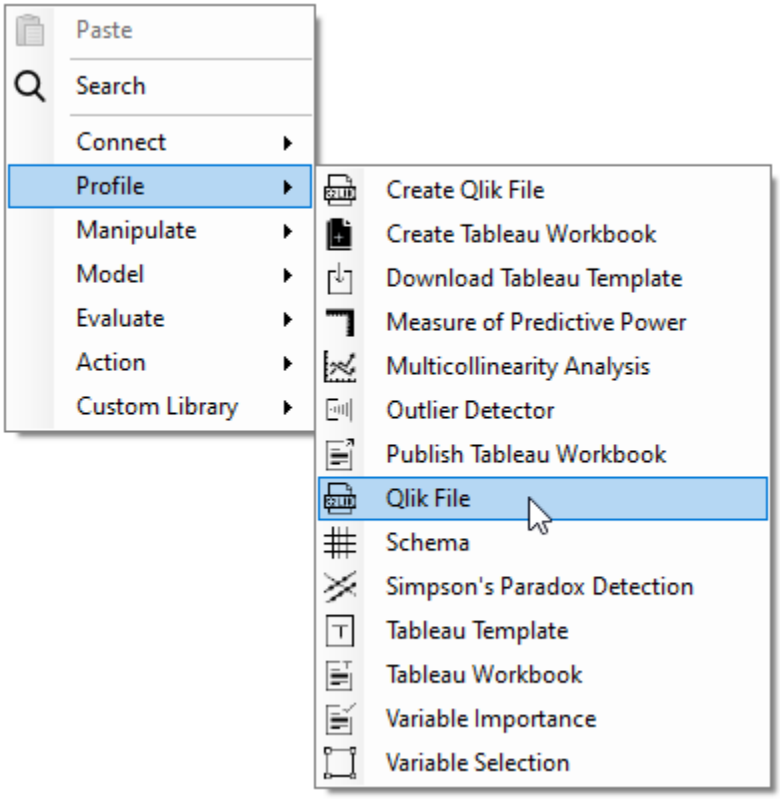
To add an existing Qlik file to your Workflow:

- Right-click anywhere on the Workflow canvas background. From the context menu, select **Profile --> Qlik File**, as shown in the example below. A new Qlik File node will be created and the link dialog will open.

  OR:

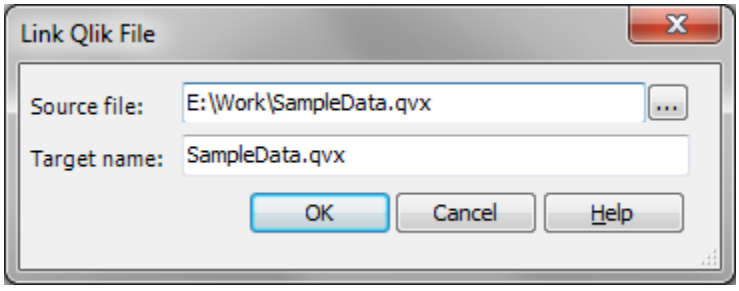
- Drag the **Qlik File** node from the **Profile** palette onto the canvas.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.





Double-click the new node or right-click on it and select **Modify** to open the link dialog. Browse for the desired Qlik file or type the path to it in the Source File field.



Click OK. The new Qlik File node is now linked to the actual file. The linking operation copies the source file to the current project. If it was not already in the project, the linked file will now be displayed in the Project Explorer. You can open it by double-clicking the node in the workflow or the corresponding object in the Project Explorer. The file will be opened in the Qlik application on your PC.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

*Note: If Qlik is not installed or *.qvx is not associated with Qlik on your PC, double-clicking on this node will open a standard Windows prompt asking you to choose the desired program to open this file.*

See also:

Integration with Qlik: Overview

Creating a Qlik File

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Selecting and Reusing Variables with Variable Lists

Variable List

Variable List (often used in combination with the **Field Finder**) is a convenient tool that saves time and gives you more control in the variable selection process in many areas of the application, including data import, selecting independent variables for your models, etc.

Using variable lists, you can quickly select variables in datasets, dataframes, models, etc., and then use the saved variable list in other objects and processes. Variable Lists can be exported and imported, which also makes it possible to share them between several projects.

Variable Lists also provide a way to quickly repeat tasks that involve variable selection and can save you a lot of time and effort especially in the case of datasets with a lot of variables (hundreds or even thousands). For example, after importing and analyzing a dataset, you can save in a variable list a subset of variables selected for modelling. Then this list can be used in the future for quick selection of this subset when creating or modifying models, importing and exporting data, etc.

Variable Lists also allow you to do advanced variable selection based on various statistics available in the object you are working with. For example, based on the Overview Report of a dataset, you can quickly select variables depending on the number or percentage of missing values, their mean values, etc. If you are using a Decision Tree for variable reduction, you can rank all your variables by performing a Find Split and then quickly generate a variable list that would pick only variables with high Information Value (see Mathematical Formulations: Decision Tree).

- To insert a Variable List, select the menu command **Insert | Variable List**
- To save a copy of an existing Variable List in the same project, open it and select the menu command **File | Save Variable List As...**.

The interface of the Variable List dialog depends on its source object (the object that is open when you are inserting the list).

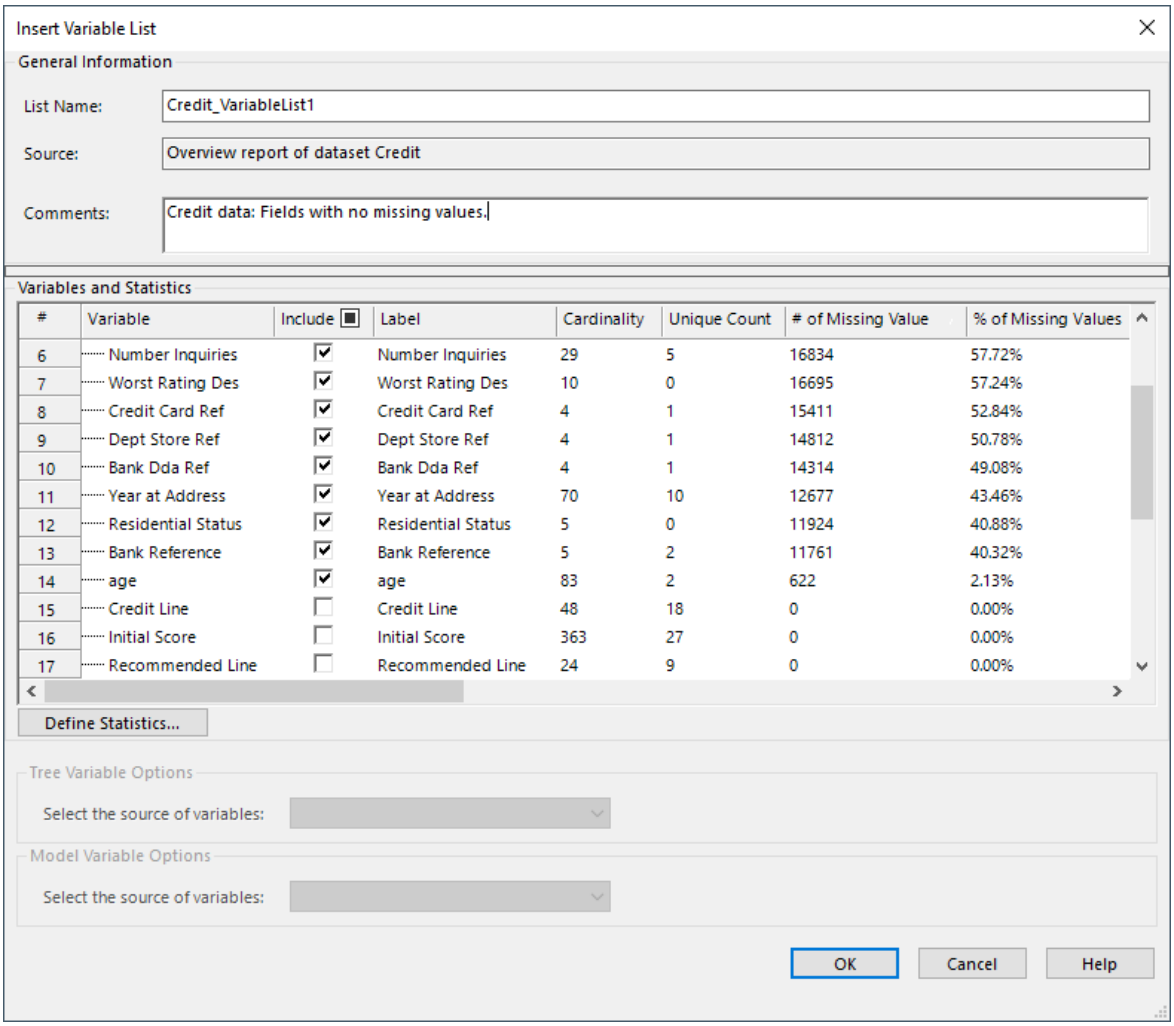
A Variable List can be exported to an XML file or a plain text file. See Exporting Variable Lists for details.

1. Variable List based on the Overview Report of a Dataset or a Dataframe

An example of a variable list based on Overview Report is shown below. It includes univariate statistics shown and calculated in the Overview Report at the time the list is inserted.

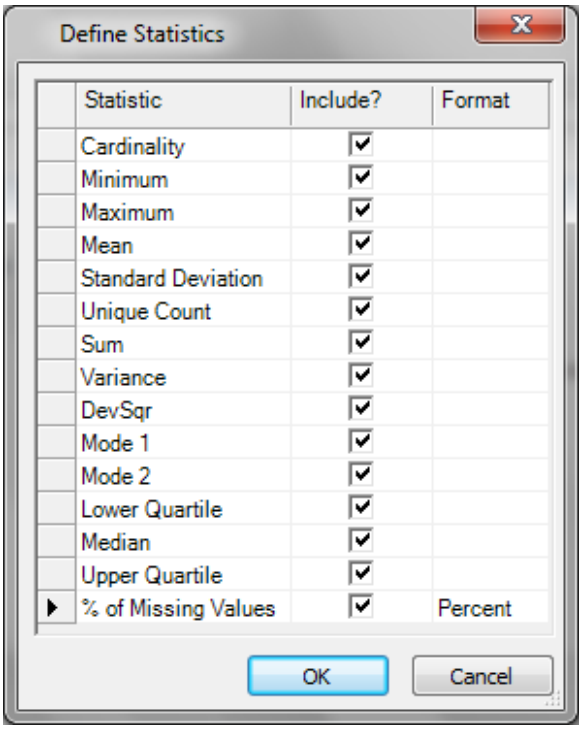
> ***Note:*** Not all of the univariate statistics are shown in the Overview Report by default. If you need more statistics, such as Sum, Variance, etc., add them to Overview Report using **Tools | Options** and calculate them using **Calculate All** *before inserting the variable list*.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The **Define Statistics** button allows you to show or hide columns in the Variable List.

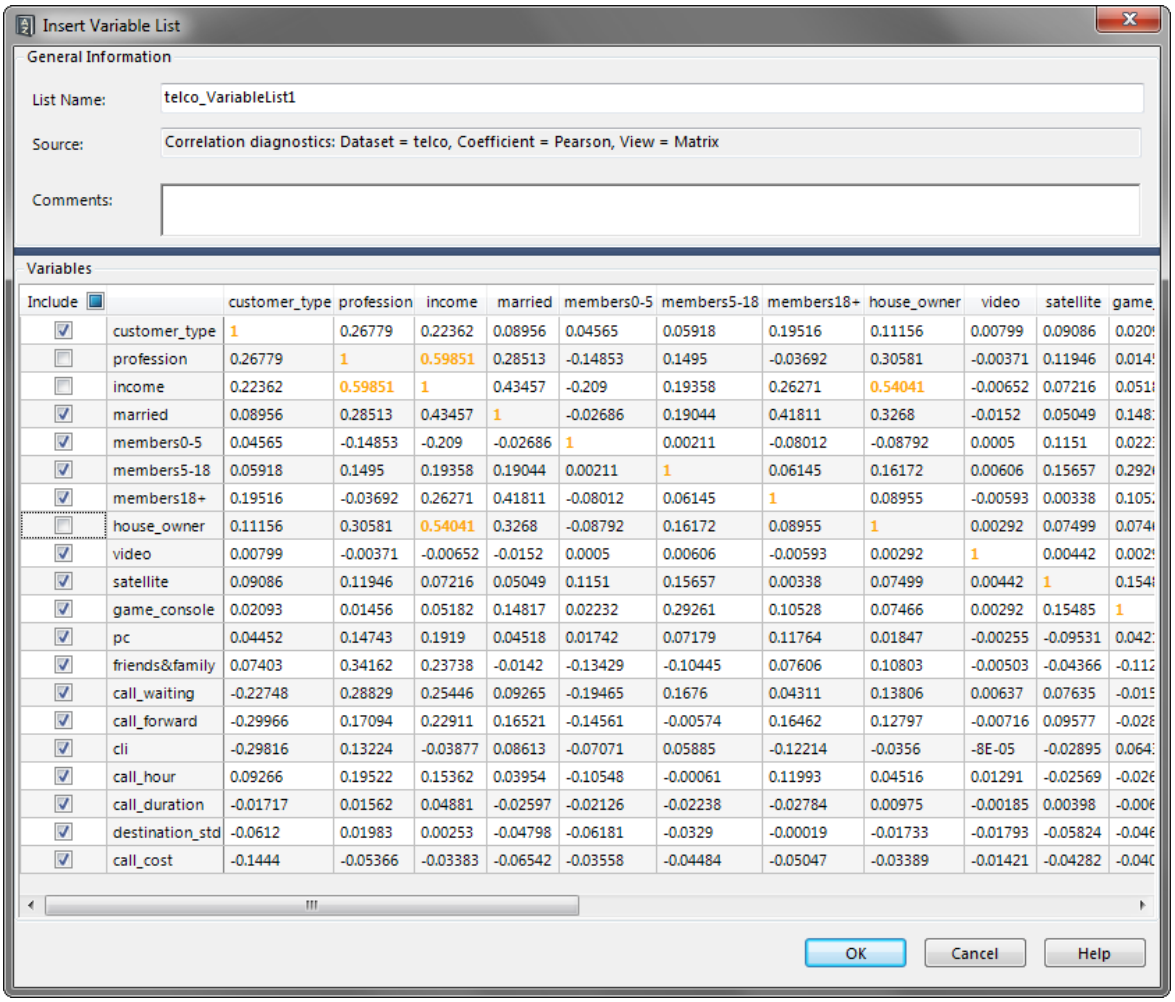
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



2. Variable List based on the Correlations View

An example of a variable list based on the Correlations View of a dataset is shown below. In addition the list of variables, it also shows the correlation matrix and the highlighting that was applied at the time of inserting the list.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

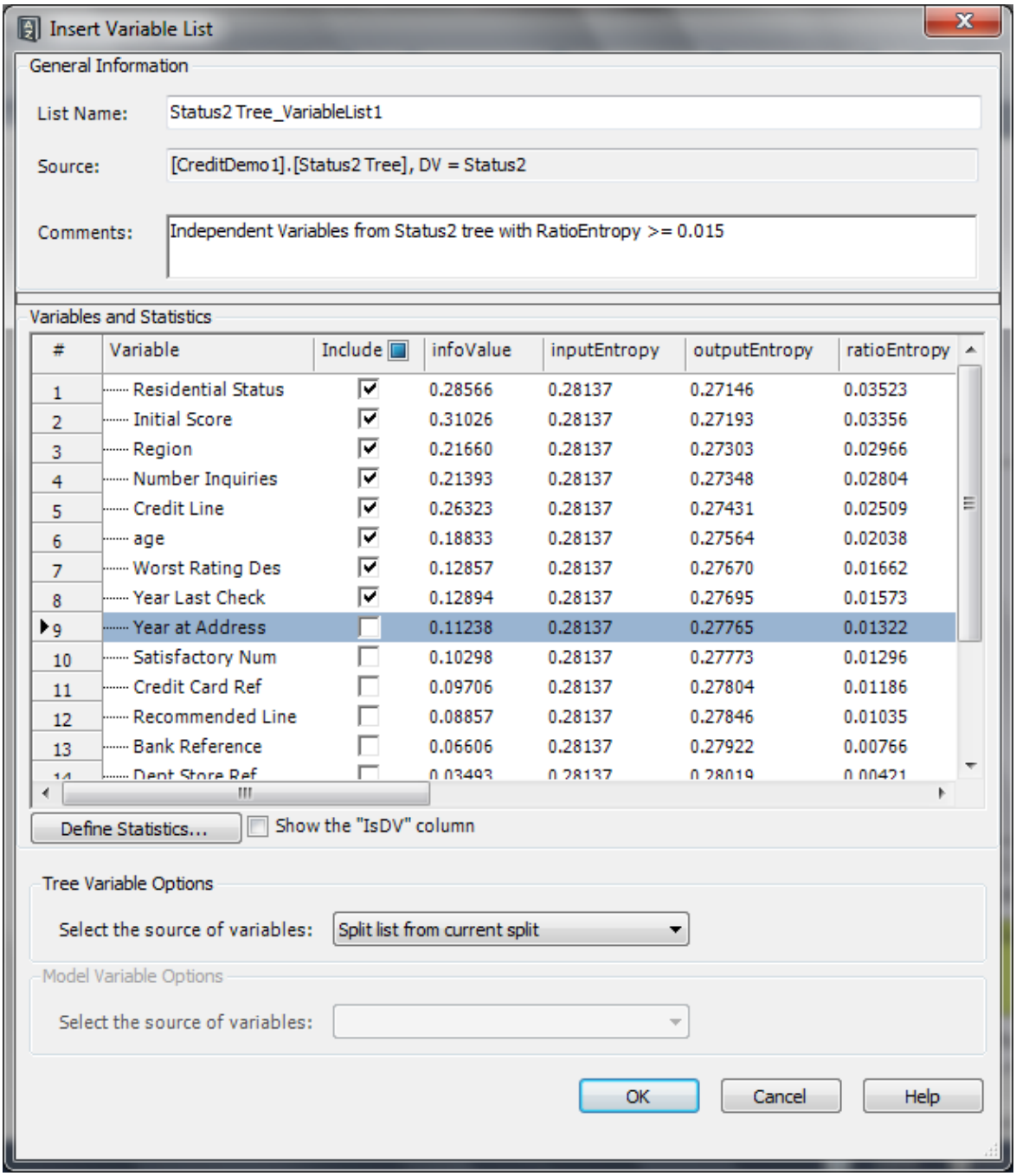


Include		customer_type	profession	income	married	members0-5	members5-18	members18+	house_owner	video	satellite	game
☑	customer_type	1	0.26779	0.22362	0.08956	0.04565	0.05918	0.19516	0.11156	0.00799	0.09086	0.020
☐	profession	0.26779	1	0.59851	0.28513	-0.14853	0.1495	-0.03692	0.30581	-0.00371	0.11946	0.014
☐	income	0.22362	0.59851	1	0.43457	-0.209	0.19358	0.26271	0.54041	-0.00652	0.07216	0.051
☑	married	0.08956	0.28513	0.43457	1	-0.02686	0.19044	0.41811	0.3268	-0.0152	0.05049	0.148
☑	members0-5	0.04565	-0.14853	-0.209	-0.02686	1	0.00211	-0.08012	-0.08792	0.0005	0.1151	0.022
☑	members5-18	0.05918	0.1495	0.19358	0.19044	0.00211	1	0.06145	0.16172	0.00606	0.15657	0.292
☑	members18+	0.19516	-0.03692	0.26271	0.41811	-0.08012	0.06145	1	0.08955	-0.00593	0.00338	0.105
☐	house_owner	0.11156	0.30581	0.54041	0.3268	-0.08792	0.16172	0.08955	1	0.00292	0.07499	0.074
☑	video	0.00799	-0.00371	-0.00652	-0.0152	0.0005	0.00606	-0.00593	0.00292	1	0.00442	0.002
☑	satellite	0.09086	0.11946	0.07216	0.05049	0.1151	0.15657	0.00338	0.07499	0.00442	1	0.154
☑	game_console	0.02093	0.01456	0.05182	0.14817	0.02232	0.29261	0.10528	0.07466	0.00292	0.15485	1
☑	pc	0.04452	0.14743	0.1919	0.04518	0.01742	0.07179	0.11764	0.01847	-0.00255	-0.09531	0.042
☑	friends&family	0.07403	0.34162	0.23738	-0.0142	-0.13429	-0.10445	0.07606	0.10803	-0.00503	-0.04366	-0.112
☑	call_waiting	-0.22748	0.28829	0.25446	0.09265	-0.19465	0.1676	0.04311	0.13806	0.00637	0.07635	-0.015
☑	call_forward	-0.29966	0.17094	0.22911	0.16521	-0.14561	-0.00574	0.16462	0.12797	-0.00716	0.09577	-0.028
☑	cli	-0.29816	0.13224	-0.03877	0.08613	-0.07071	0.05885	-0.12214	-0.0356	-8E-05	-0.02895	0.064
☑	call_hour	0.09266	0.19522	0.15362	0.03954	-0.10548	-0.00061	0.11993	0.04516	0.01291	-0.02569	-0.026
☑	call_duration	-0.01717	0.01562	0.04881	-0.02597	-0.02126	-0.02238	-0.02784	0.00975	-0.00185	0.00398	-0.006
☑	destination_std	-0.0612	0.01983	0.00253	-0.04798	-0.06181	-0.0329	-0.00019	-0.01733	-0.01793	-0.05824	-0.046
☑	call_cost	-0.1444	-0.05366	-0.03383	-0.06542	-0.03558	-0.04484	-0.05047	-0.03389	-0.01421	-0.04282	-0.040

3. Variable List based on a Decision Tree or a Strategy Tree

An example of a variable list based on a Decision Tree is shown below. The IsDV column (if shown) indicates whether a variable is the dependent variable of the tree. You can hide this column by unchecking the box "**Show the IsDV column**".

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Tree Variable Options

Split list from current split	If this option is selected, the Variable List includes the DV and the variables of all splits ***at the currently selected node***. It is the same as the list of variables in the Split Report tab or the list displayed by the Go To Split command for the currently selected node. It will also

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

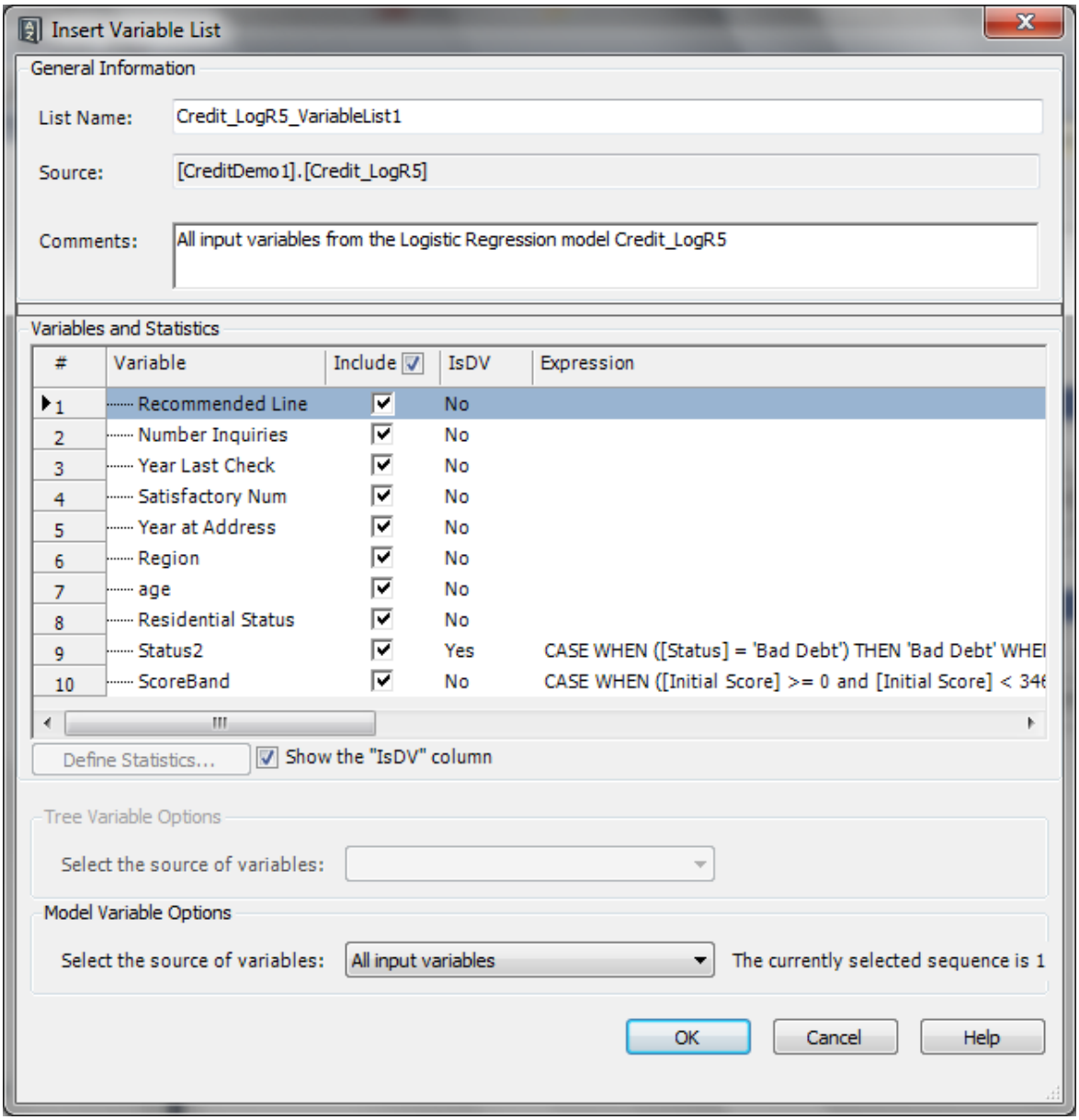
	show the split parameters for each split variable (note that split parameters depend on the measure).
Variables used in visible tree	If this option is selected, the Variable List includes the DV and the split variables of ***all visible splits in the tree***. It does **not** include the variables of the splits that are not shown (but accessible via the Go To Split command). Split parameters are not shown in this case.
Split list from all splits	Variable List includes all variables from all splits in the tree. It does not show split parameters in this case.

4. Variable List based on a Predictive Model or a Cluster Analysis Model

This is an example of a variable list based on a Predictive Model or a Cluster Analysis Model.

In this case, the source is a Logistic Regression model with stepwise variable selection. In the Model Variable Options drop-down list, you can select a particular sequence as a source of variables. By default, all input variables of the model are selected.

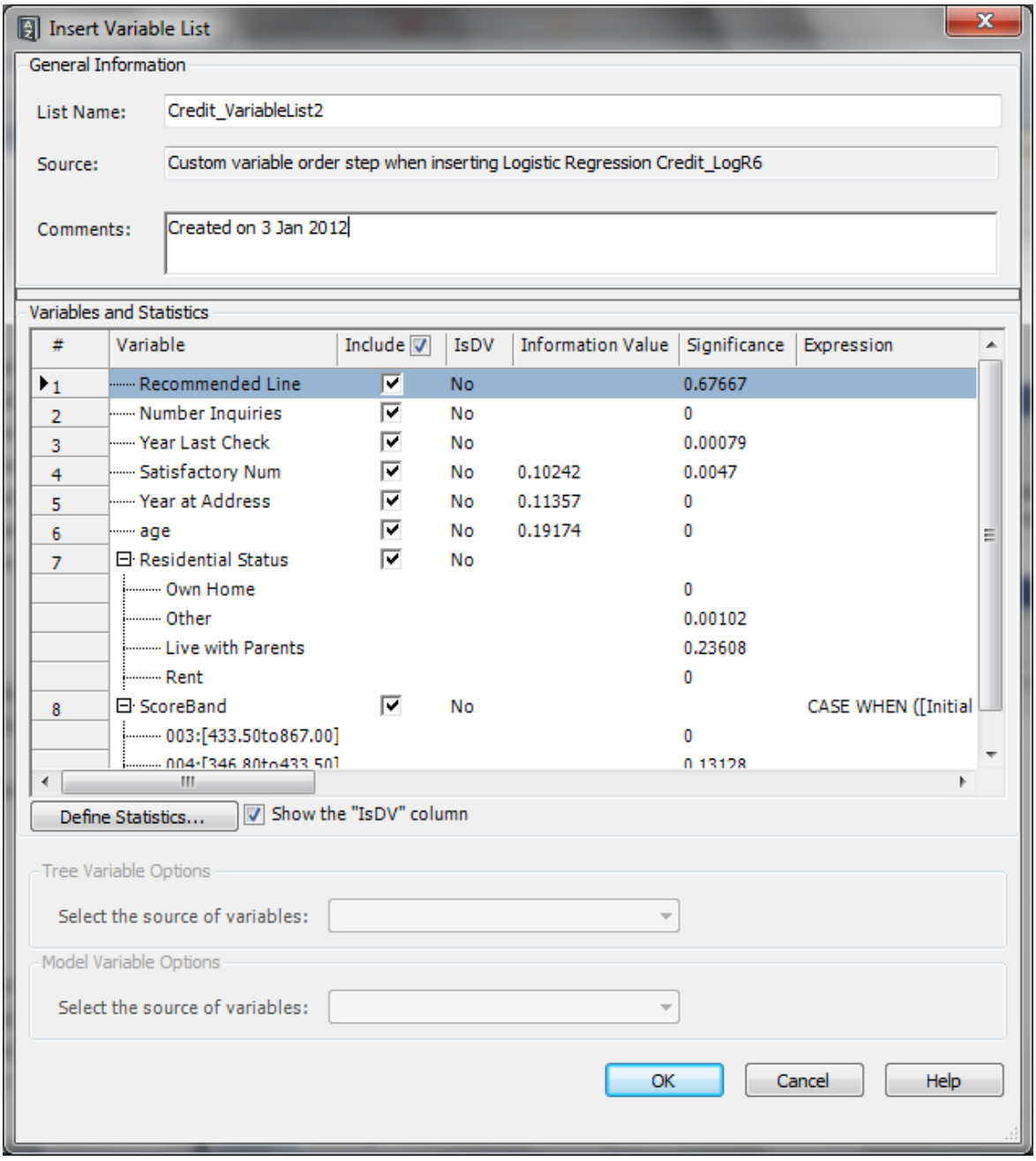
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



5. Variable List based on a Custom Variable Sequence in Predictive Model Wizard

In the Variable Selection Methods of the Predictive Model Wizard, there is an option to define a variable sequence of your choice. If you use this option and define a custom sequence, you can save it in a Variable List by clicking the Save Variable List button in the wizard page.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

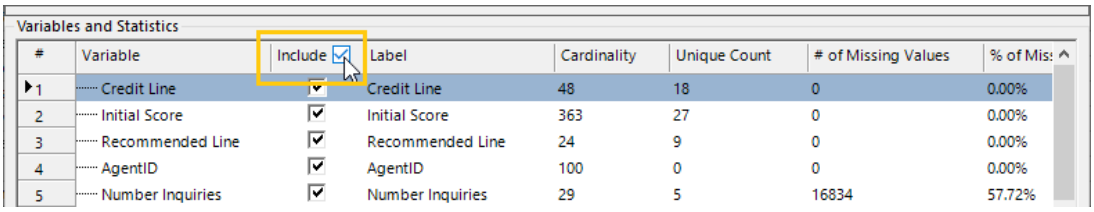


You can save a copy of an existing variable list using the command **File | Save Variable List As...**. The list can be modified before saving, if necessary.

Usability Features

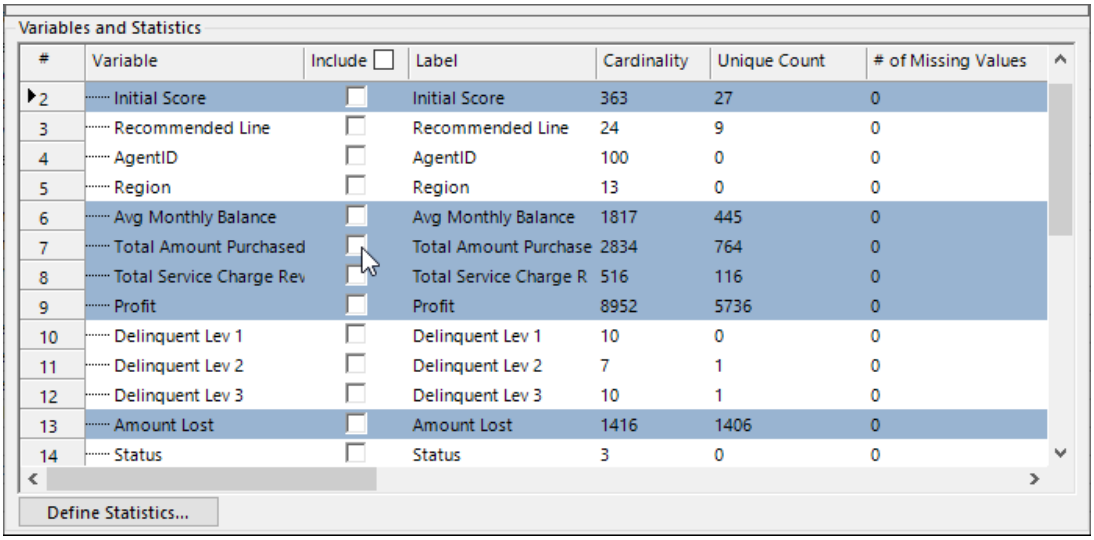
- All columns in the Variable List grid are sortable. To sort on a column, click on its header.
- To select or deselect all variables at once, select the checkbox in the header of the ***Include*** column:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



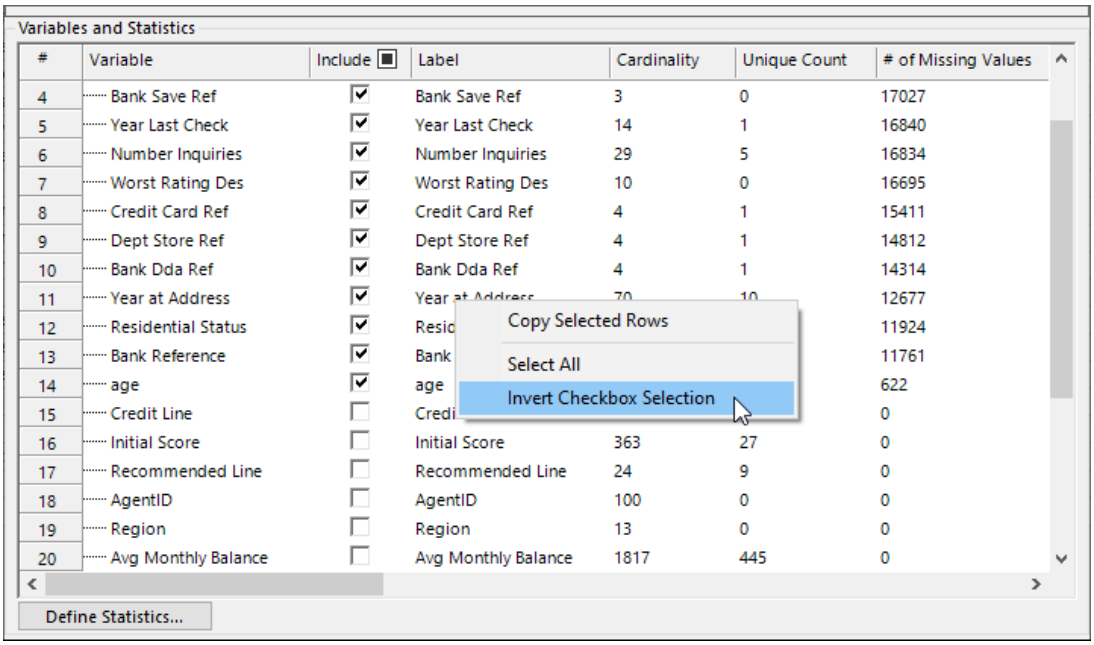
Alternatively, highlight all rows by press CTRL+A on your keyboard or right-clicking anywhere in the grid and choosing **Select All** from the context menu. Then click in any checkbox in the *Include* column.

- To select multiple variables, first, highlight the required rows by clicking in the index column (the leftmost column) and dragging the mouse down. Alternatively, click in the index column while holding down the SHIFT or CTRL key on your keyboard. Then select the checkbox in the *Include* column of for any of the highlighted rows. This will select all highlighted variables.



- To invert the current variable selection, right-click anywhere in the grid and choose the **Invert Checkbox Selection** command from the context menu.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Alternatively, highlight all rows by press CTRL+A on your keyboard or right-clicking anywhere in the grid and choosing **Select All** from the context menu. Then click in any checkbox in the *Include* column.

See also:

Field Finder

Exporting Variable Lists

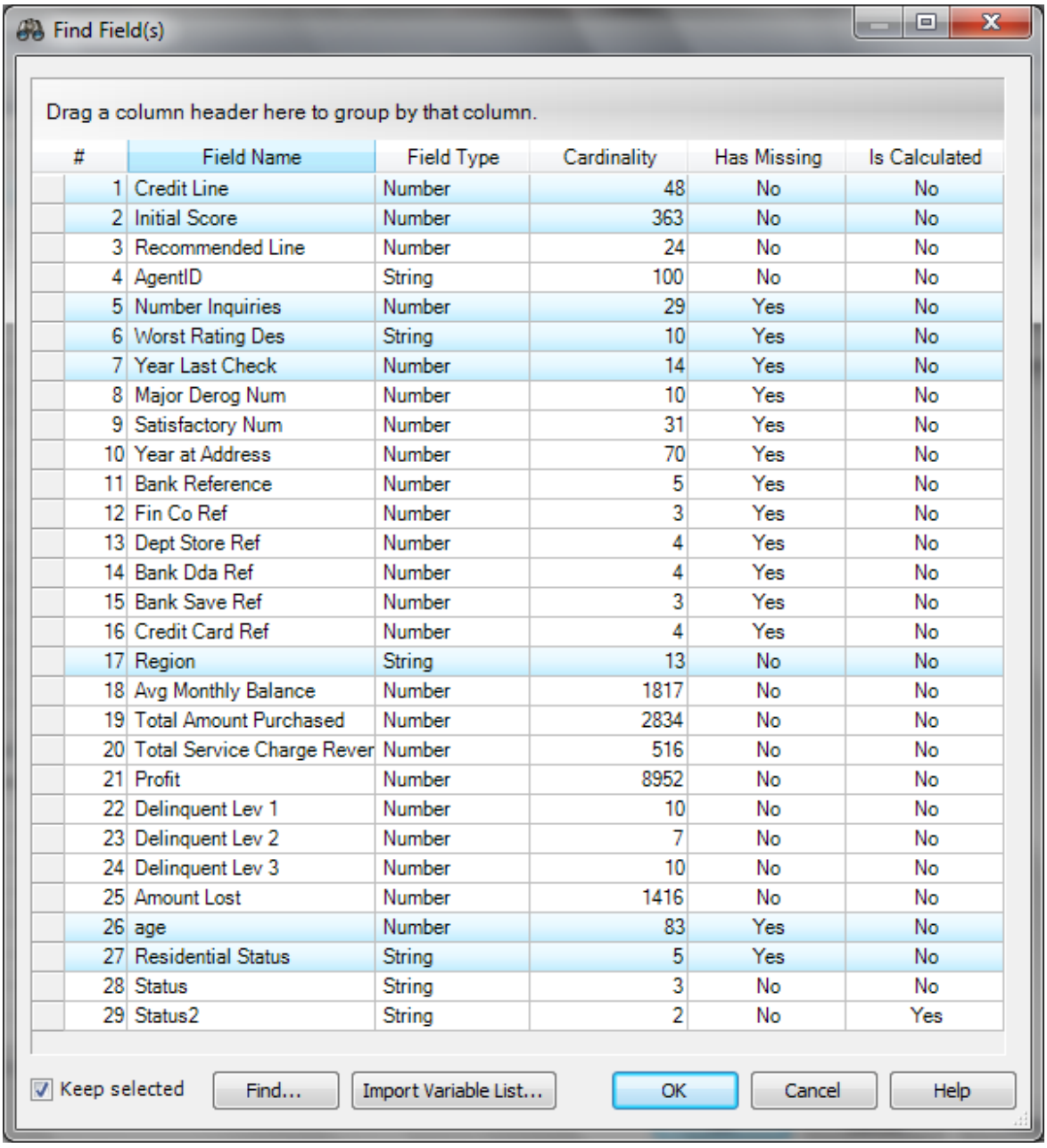
Field Finder

Field Finder provides an easy way to find and select the fields of interest in any area of the application, for example, when importing a dataset, selecting variables for correlation analysis, predictive model, segment viewer charts, etc. It is especially helpful when there are a lot of variables in your dataset (hundreds or even thousands).

Field Finder is launched by the binoculars button , which can be found in any view or dialog that involves selecting or highlighting a field (variable).

The field attributes displayed in the Field Finder are: data type, cardinality, missing values indicator (yes/no), and the indicator of whether a field is calculated (i.e., has an associated expression in Dataset Editor).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

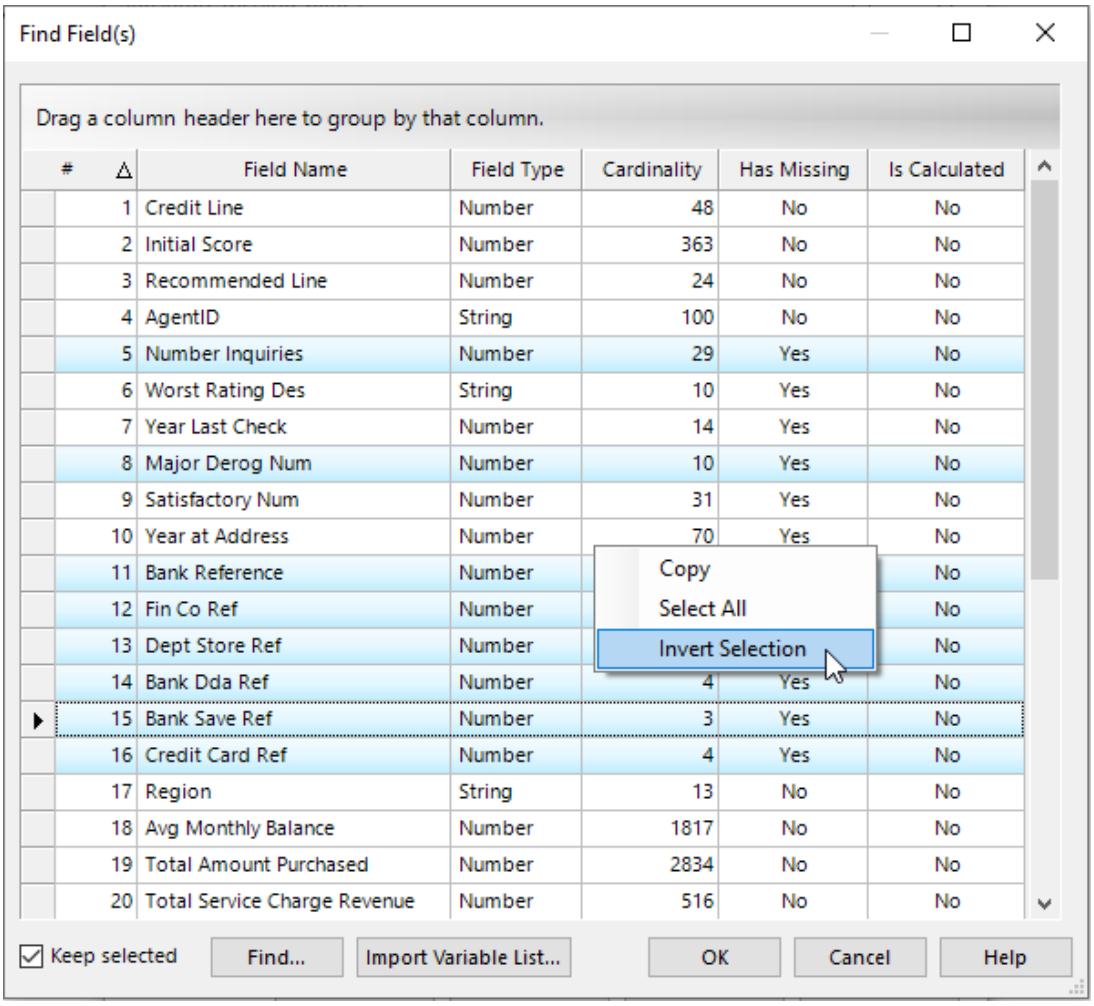


#	Field Name	Field Type	Cardinality	Has Missing	Is Calculated
1	Credit Line	Number	48	No	No
2	Initial Score	Number	363	No	No
3	Recommended Line	Number	24	No	No
4	AgentID	String	100	No	No
5	Number Inquiries	Number	29	Yes	No
6	Worst Rating Des	String	10	Yes	No
7	Year Last Check	Number	14	Yes	No
8	Major Derog Num	Number	10	Yes	No
9	Satisfactory Num	Number	31	Yes	No
10	Year at Address	Number	70	Yes	No
11	Bank Reference	Number	5	Yes	No
12	Fin Co Ref	Number	3	Yes	No
13	Dept Store Ref	Number	4	Yes	No
14	Bank Dda Ref	Number	4	Yes	No
15	Bank Save Ref	Number	3	Yes	No
16	Credit Card Ref	Number	4	Yes	No
17	Region	String	13	No	No
18	Avg Monthly Balance	Number	1817	No	No
19	Total Amount Purchased	Number	2834	No	No
20	Total Service Charge Rever	Number	516	No	No
21	Profit	Number	8952	No	No
22	Delinquent Lev 1	Number	10	No	No
23	Delinquent Lev 2	Number	7	No	No
24	Delinquent Lev 3	Number	10	No	No
25	Amount Lost	Number	1416	No	No
26	age	Number	83	Yes	No
27	Residential Status	String	5	Yes	No
28	Status	String	3	No	No
29	Status2	String	2	No	Yes

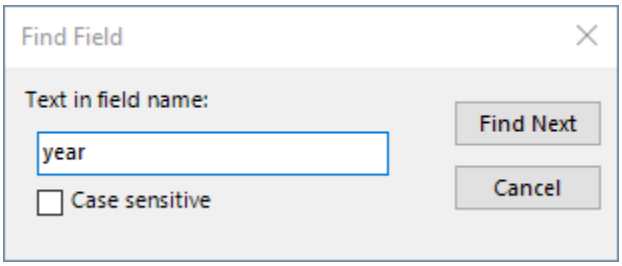
Usability Features

- All columns in the Field Finder grid are sortable. To sort on a column, click on its header.
- To select or deselect all fields at once, press CTRL+A on your keyboard or right-click anywhere in the grid and choose **Select All** from the context menu.
- To select multiple fields at once, click on any row and drag the mouse down. Alternatively, click on the rows while holding down the SHIFT or CTRL key on your keyboard. .
- To invert the current selection, right-click anywhere in the grid and choose the **Invert Selection** command from the context menu, as shown below:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- If the "**Keep Selected**" box is checked, then a single mouse-click adds fields to your selection without cancelling the existing selection. This option may be especially useful if the number of fields is very large.
- Click the **Find** button at the bottom to search for a field (or fields) whose name contains a particular substring.



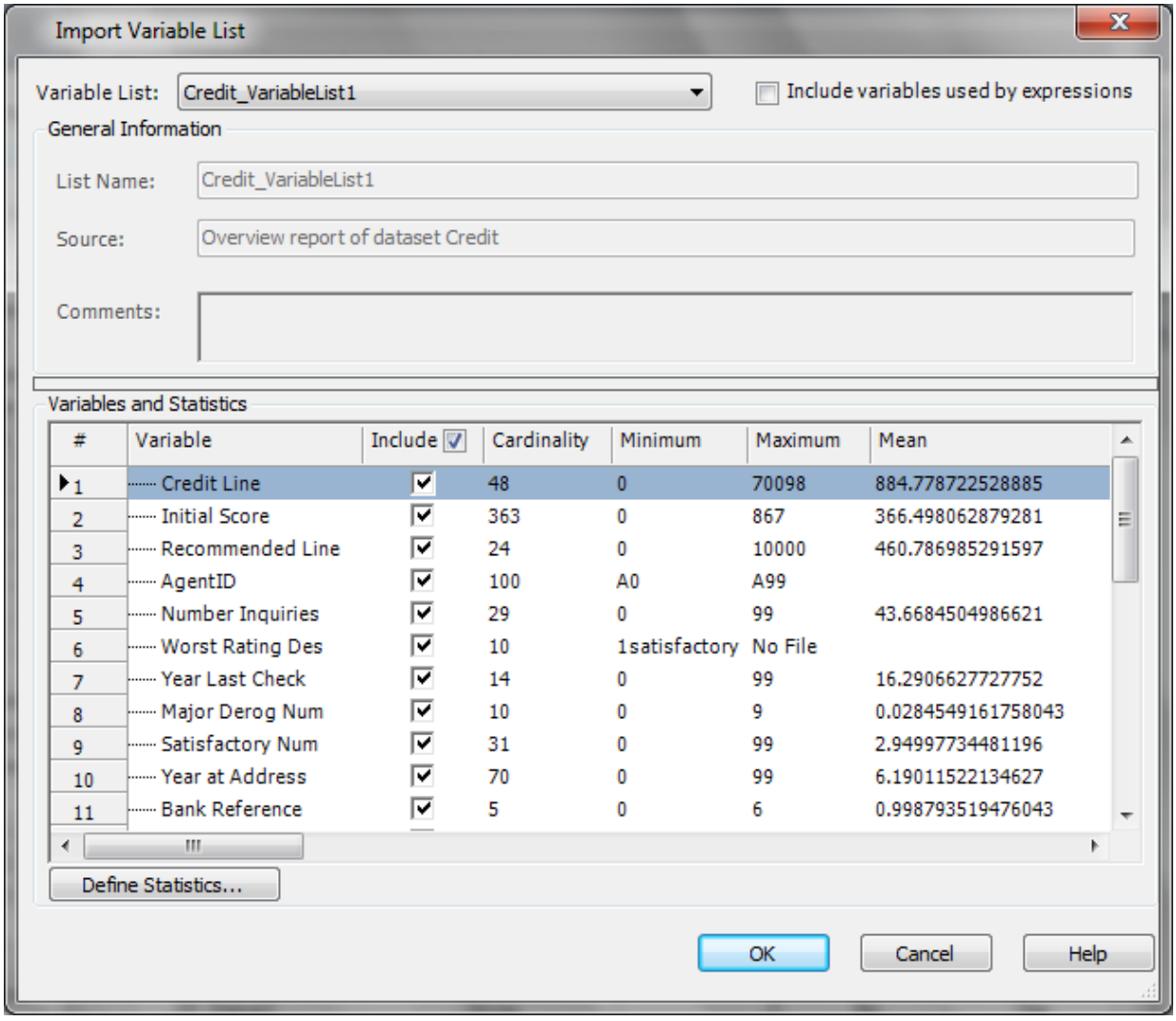
- Click the **Import Variable List** button at the bottom to apply the variable selection from a previously saved Variable List.

Importing Variable Lists

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Field Finder can use Variable Lists for advanced field selection to apply the selection of fields from any Variable List saved previously. (The Variable List option is not applicable in the case of Spark dataframes). For example, suppose that you saved the list of all variables of your dataset that have no missing values. If at some point you need the same selection of variables for your correlation analysis or principal component analysis, you can use the Field Finder in the respective wizards and apply the selection by importing your variable list.

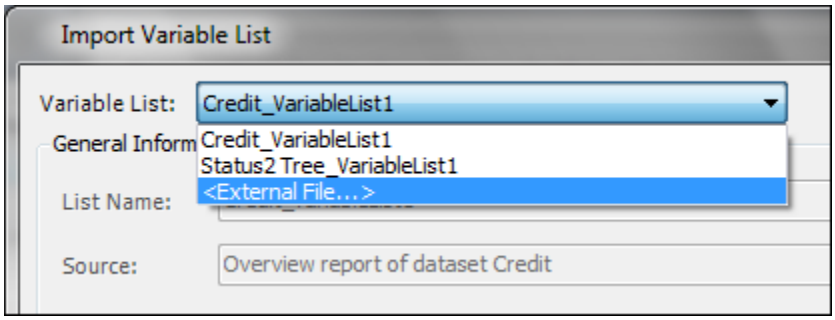
Using the **Import Variable List** button, you can import any variable list inserted previously and apply its selection to the list in the Field Finder. You can import a variable list from any project or even an external file.



If the variable list you are importing contains calculated columns, then you can choose to include all variables occurring in the expressions for these calculated columns. For example, in the figure above, the *ScoreBand* variable is a calculated field that depends on the *Initial Score* variable. If the checkbox "Include variables used by expressions" is checked, then *Initial Score* will be imported together with *ScoreBand*.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If you wish to import a variable list from another project or elsewhere, use the External File option in the drop-down list:



Then browse for the desired variable list file. (Variable lists are represented by files with ***.kvl** extension in project directories).

If you are importing a variable list from an XML file (see Exporting Variable Lists), change the File Type from *.kvl to *.xml and browse for the desired file.

See also:

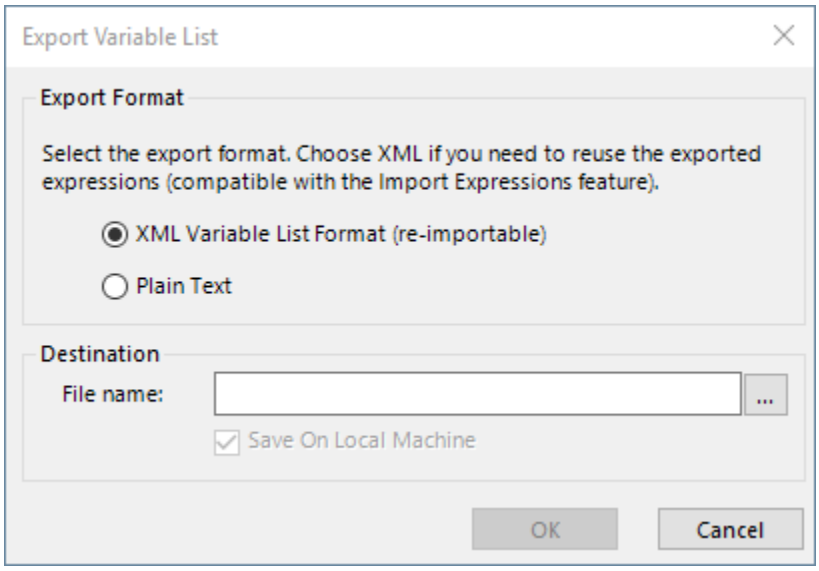
- Variable List
- Exporting Variable Lists

Exporting Variable Lists

A Variable List can be exported to an XML file for future use in Field Finder. Although in Field Finder you can import variable lists *directly* from the current project or other projects, you may want to save a variable list in an external file to store it in case your project may get deleted or even for the purpose of sharing it with other users.

To export a variable list, open it and select the command **File | Export ...** from the main menu.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



The default format for the output file is **XML**. Variable List files can have extensions KVL, KVLC, and XML files (*.kvl,*.kvlc, *xml).

The resulting variable list file can be imported into Field Finder whenever necessary in order to apply the same field selection. See the Field Finder topic for details on importing variable lists.

Choosing the **Plain Text** format will produce a text file with the variable list in a tabular form. A variable list file in the plain text format cannot be reimported.

If you are working in the client/server mode, you can save the exported file either on the local machine or on the server.

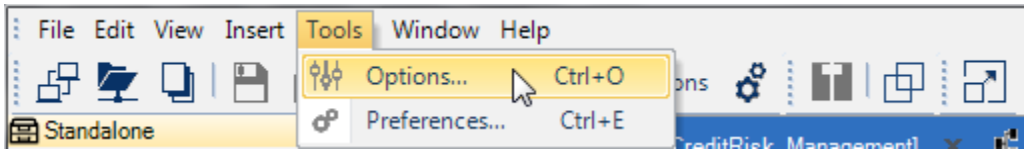
See also:

- Variable List
- Field Finder

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Options and Preferences

Various attributes, parameters and display settings for all types of objects and views can be changed using the **Tools | Options** and **Tools | Preferences** menu commands. The **Options** command is only applicable to the object or tab that is currently in active view, while the **Preferences** command is always enabled and is applied globally.



Purpose and Scope of Options and Preferences

- Changes made through **Tools | Options** are applied locally - only to the object that is currently active. They take effect immediately. Aside from the **Tools** menu, Options can be accessed by clicking the Options toolbar button 

or, in some cases, using the right-click context menu (for instance, in the Decision Tree view).

- The settings and properties that can be changed through **Tools | Preferences** are global preferences determining general display settings or the properties of objects (models, datasets, etc.) to be created. If you change any preferences for a certain type of objects (for example, decision trees), this will apply to all objects of this type created in the future in any project. The changes in Preferences do not affect existing datasets and models. Aside from the Tools menu, Preferences can be accessed by clicking the toolbar button  .

  The Preferences dialog allows you to save your current preferences or apply previously saved preferences. For example, this can be useful if you would like to keep your current preferences when upgrading to a new version. See the topic Importing and Exporting Preferences for details.

Categories of Options and Preferences

For the description of each category of options and preferences, click on the appropriate link:

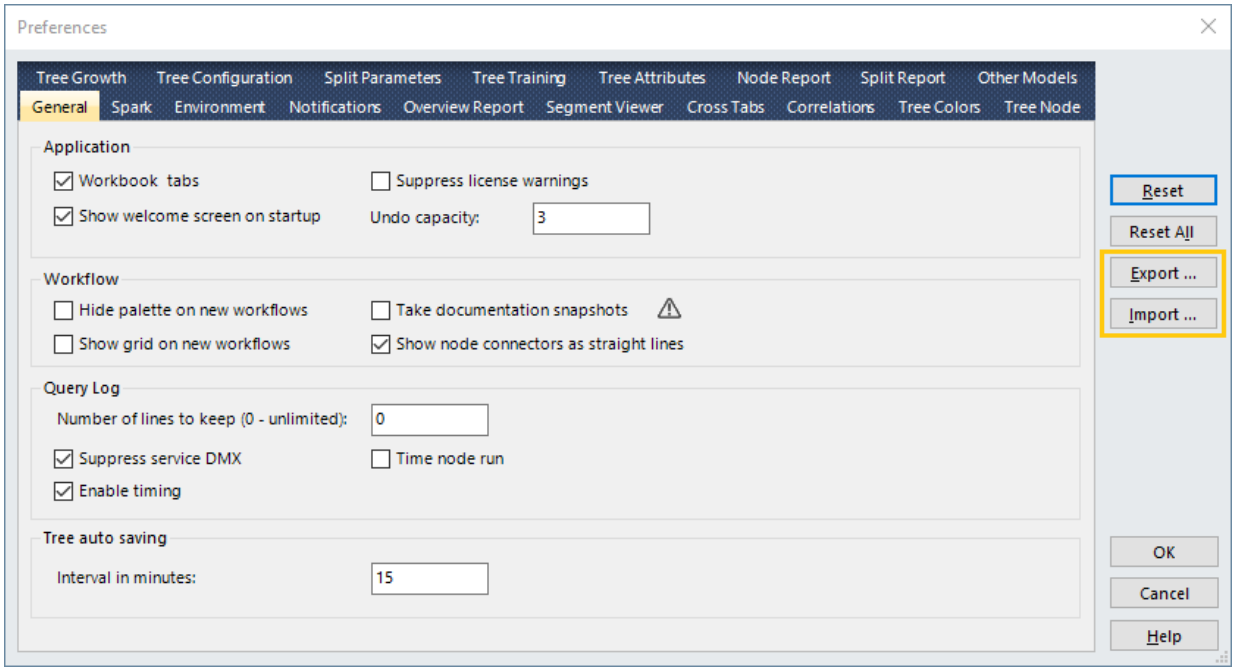
- General
- Overview Report
- Segment Viewer
- Cross Tabs
- Correlations
- Tree Training
- Tree Growth
- Tree Configuration
- Tree Node Settings
- Tree Colors

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Split Parameters
- Node Report
- Tree Attributes
- Other Model Attributes (for models other than Decision Trees and Strategy Trees)
- Split Report
- Environment
- Notifications

The **Preferences** dialog accessible from the **Tools** menu allows you to set application preferences. See the topic Options and Preferences topic for the detailed description of application settings.

You can save your current preferences or apply previously saved preferences. Use the **Export** and **Import** buttons in the Preferences dialog to export preferences to a file or import them from a file.



- Use the **Export** button to save your current preferences for the following purposes:
  - to keep the preferences of the current version and apply them when upgrading to a new version
  - to keep several different preference profiles that can be loaded in different business contexts
  - as a backup in case you need to reinstall the application and reapply the preferences
  - to share your preferences with co-workers

  Click the **Export** button, browse the file system for the desired destination folder, and specify the file name. Click **Save**. The current settings from all tabs of the Preferences dialog will be saved in a re-importable XML file. In the client/server mode, the file is saved on the file system of the client machine.

- Use the **Import** button to import preferences from an existing preferences file:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

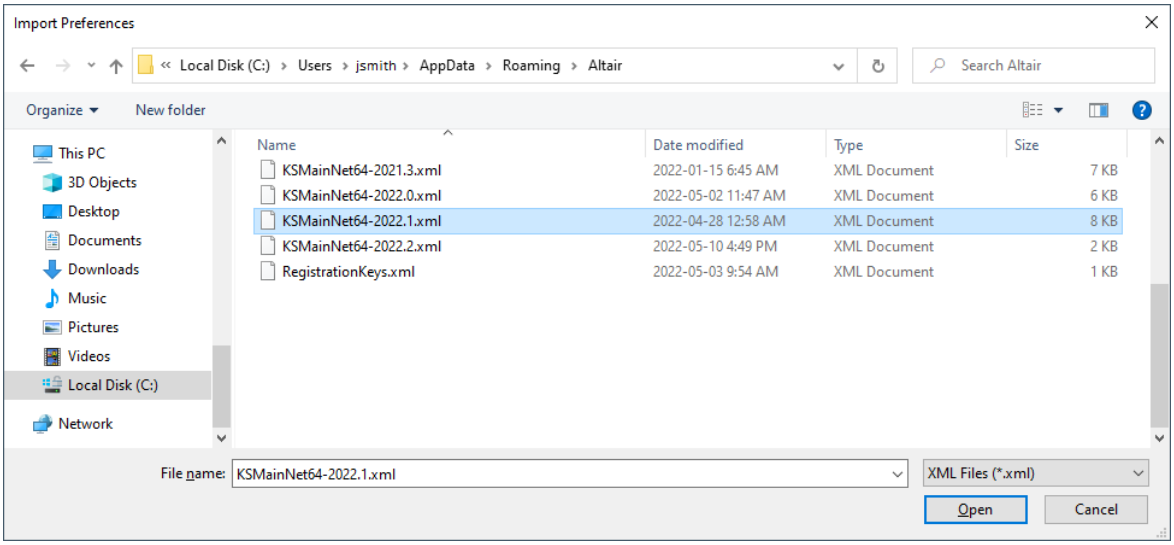
- to apply the preferences of the previous version after upgrading to a new version
- to load one of the several previously saved preference profiles
- to restore your preferences from a backup file after reinstalling the application
- to apply a preferences file shared by a co-worker

Click the **Import** button and browse for the XML file that contains the desired preferences. Select the file and click **Open**. In the client/server mode, the file system of the client machine is browsed. The current settings from all tabs of the Preferences dialog will be saved in a re-importable XML file. The current settings in the tabs of the Preferences dialog are overwritten by those in the imported file. Preferences not specified in the imported file remain unaffected.

Applying Preferences When Upgrading to a New Version

The preferences of a previous version can be applied even without explicitly saving the preferences file before the upgrade.

To apply the preferences of the previous version after upgrading to a new version, import the application preferences file of the previous version `KSMainNet64-<version>.xml`. It is stored in the Altair application data folder of the current user profile: `%APPDATA%\Altair` (or `%APPDATA%\Angoss` for versions older than 10.8.2). For example, for version 2021.1 it is `C:\Users\<user_id>\AppData\Roaming\Altair\KSMainNet64-2021.1.xml`. Here is an example of the Import Preferences browsing dialog when importing a 2021.1 preferences file:



The exported preferences file can also be specified in the command line as part of the silent install for the new version of the application. The installer will then apply your preferences to the new version. Specify the preferences XML file as an argument in the silent install command line.

Example:

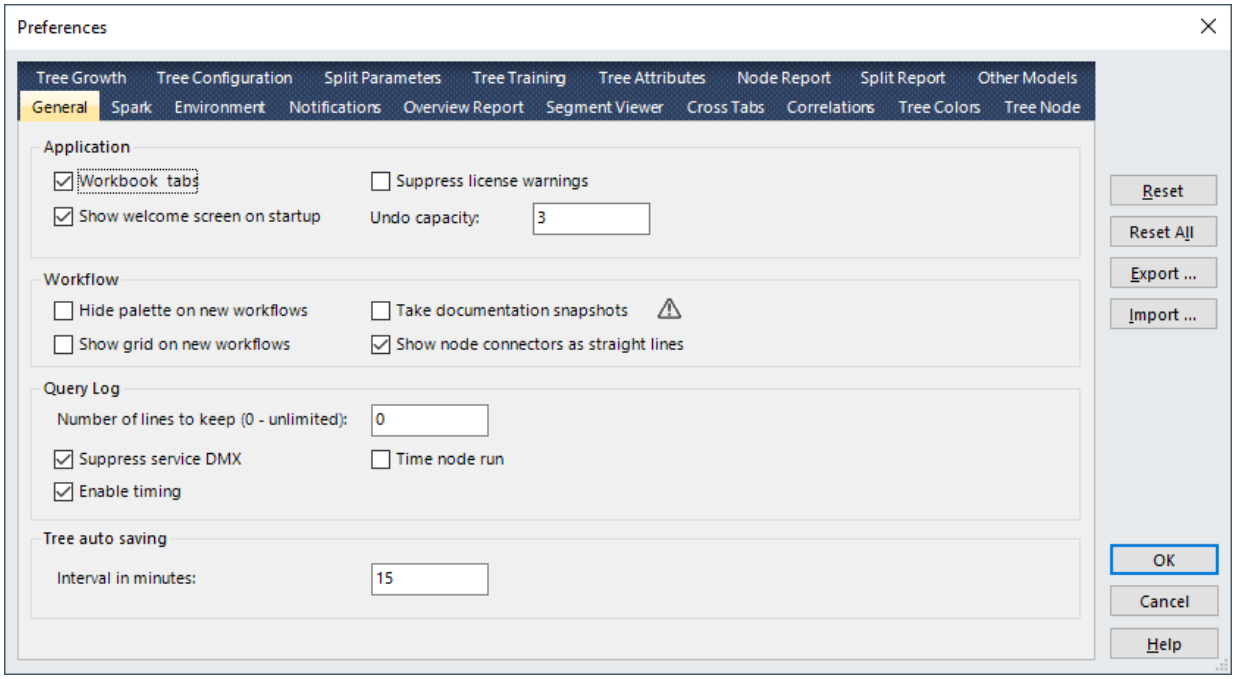
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
msiexec /i "KS-Workstation-2022.3-64bit.msi" /l*v "log.log"
PREFERENCES="C:\Users\jsmith\AppData\Roaming\Altair\KSMainNet6
4-2021.1.xml" /quiet
```

As a result, all applicable parameters in the specified XML file that correspond to the preference values will be set as the default preference settings in **Tools | Preferences** of the installed application.

For details about the silent install, see the Deployment Guide available on the software download page.

General application preferences



Workbook tabs	Shows a separate tab for every object that is currently open. This makes it easier to navigate between the currently open objects.
Show welcome screen on startup	Shows the Welcome screen every time the application is started.
Suppress license warnings	License warnings may appear for various reasons, for example, if the license is about to expire. Selecting this option will suppress all license warnings.
Undo capacity	The maximum number of actions that can be rolled back using the Undo command in the toolbar. This applies to Decision Trees and Strategy Trees.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Workflow preferences

Hide palette on new workflows	If this option is selected, the Palette panel is automatically docked when a new workflow is created.
Show grid on new workflows	Shows the grid by default for any new workflow created in the project.
Take documentation snapshots	Snapshots are taken automatically when opening any view of any object. The snapshots are used by the Self-Documentation and Model Documentation nodes. **Note**: Since this feature can slow down the performance, it is not recommended to turn this option on unless you plan to document your workflow using the Documentation nodes.
Show node connectors as straight lines	If this option is selected, the node connectors on the workflow are shown as straight lines. This may be useful especially in busy workflows with many nodes.

Query Log preferences

Query Log provides the history of your data mining activities in the form of SQL-like commands (DMX commands). It is updated in real time as you perform your actions in the project.

For more information about the Query Log, see the topic Query Log in the Logging Capabilities section.

Number of lines to keep	Specifies the number of the most recent DMX commands to preserve in the Query Log (the default is 0, which stands for "unlimited", i.e., all commands processed during the current session will be kept in the log).
Suppress service DMX	Suppresses the auxiliary DMX commands that are not directly related to the data mining tasks being performed (for example, commands that update the project hierarchy, or commands that create temporary objects).
Enable timing	Displays the starting time and duration of each DMX command.
Time node run	Measure the time it took for each node to run and include these times in the log. Example: # =====> Node run started at 2018-12-03 2:24 AM # 'Decision Tree: Census_DecisionTree' started run at 2018-12-03 2:24:21 AM

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

```
# 'Decision Tree: Census_DecisionTree' finished
run at 2018-12-03  2:24:21 AM in 0.62 sec (0.616 sec)
# =====> Node run finished in 1.12 sec (1.116 sec)
```

Tree Auto-Saving preferences

Using this option, you can set the preferred time interval for auto-saving trees. The default interval is 15 minutes.

Cross Tabs

The settings for the Cross Tabs view of the Dataset are specified in this tab.



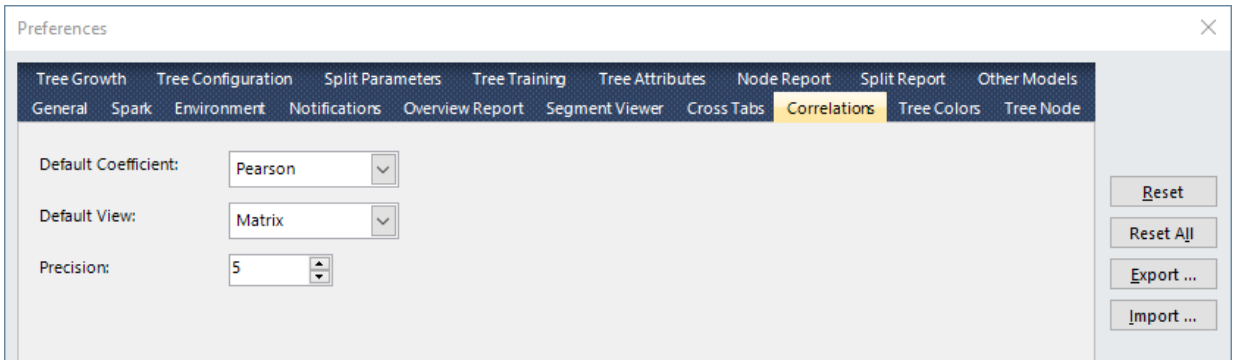
Number of bins	Maximum number of bins in which to distribute the data. The default value is 10.
Equal Height	When this is checked, the data is distributed so that each bin contains approximately the same number of observations.
Equal Width	When this is checked, the width of the intervals on the x-axis will be the same for all bins (default).

See also: Cross Tabs Chart

Correlations

Correlation and covariance coefficients are displayed in the Correlations view in the dataset. The general settings for the Correlations view are specified in this tab.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Default Coefficient	The type of correlation coefficient calculated and displayed by default in the Correlations view when you open it. The available options are: *All types, Pearson, Spearman's Rho, Kendall's Tau-b,* and *Hoeffding*. Selecting a single type instead of *All types* is recommended, since the calculation of all types may take a long time for large datasets with many numeric variables. The calculation of Pearson and Spearman's Rho coefficients take less time than other types. Pearson coefficient is selected by default.
Default View	Determines the type of the view (*Matrix*, *Pairs*, or *Chart*) displayed by default in the Correlations view when you open it. For the description of each view, see the Correlations Tab topic.
Precision	The default precision of the values of the correlation coefficients displayed in the Correlations view. 5 digits after the decimal point is the default.

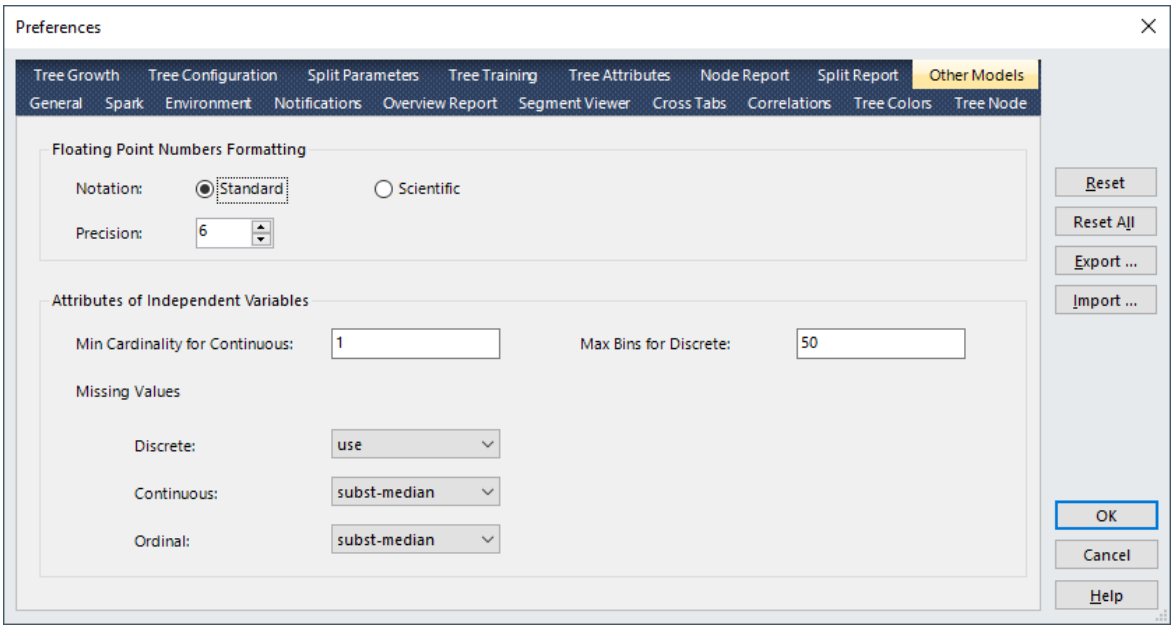
See also: Correlations Tab.

Model Attributes

The settings in the **Other Models** tab of the Preferences dialog determine the following properties:

1. Attributes of independent variables of models other than decision trees and strategy trees. (The tree attribute defaults are defined in the Tree Attributes tab of Preferences)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



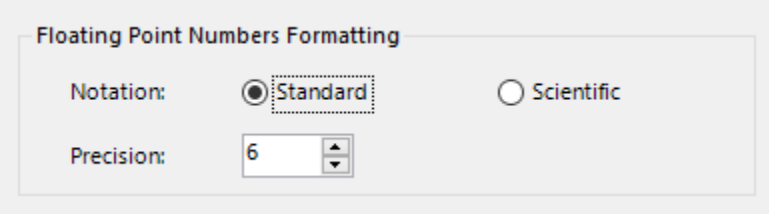
Min Cardinality for Continuous	Numeric independent variables with cardinality higher than this value are considered to be continuous. The default threshold is **1** (all numeric variables are continuous by default). If you set this threshold to **x**, then all numeric variables with cardinality >= **x** will have Usage = continuous, and all numeric variables with cardinality less than **x** will have Usage = ordinal (i.e., ordered discrete).
Max Bins for Discrete	The maximum number of categories for discrete variables. The default is 50. If the specified maximum number of categories (bins) **N** is larger or equal to the cardinality of the variable, then each of its values is represented as a separate category. If **N** is less than the cardinality, the following mapping is applied: the first N-1 unique values are represented as separate categories and the rest are mapped into one category denoted by triple number sign ("###") or "OTHER".
Missing Values	The way missing values are treated for different types of independent variables (discrete, continuous, and ordinal). By default, the missing value in a discrete variable is considered as a valid category ("use"). The available options are "use", "ignore", and "substitute mode". Missing values in continuous and ordinal variables are substituted by the median value by default. The available options for continuous are "ignore" (drop records with missing values) and "substitute median / mean / minimum / maximum / mode / zero". The available options for continuous are "use", "ignore" (drop records with missing values), and "substitute median / minimum / maximum / mode".

2. The format and precision for the numbers displayed in the following reports:

- the Parameters and Attributes tab of trees and models
- the Results tab of Predictive Models (including the Sequence Statistics views that show model parameters and detailed model output)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- the Results tab of Cluster Analysis Models



The default precision for numbers is 6 digits after the decimal point.

In some cases, for example, if a model parameter is displayed as 0.000000, it's actual value may be too small for the default display precision, so you may want to use this dialog to increase the precision or switch to the Scientific notation.

Node Report

The Node Report view in the Decision Tree provides you with information about the terminal nodes in your tree.

Click the **Node Report** tab in the **Preferences** dialog to set the following Node Report preferences:

- Font in the Node Report grid. Click the browse button to choose the font.
- Option: *Display rule with a column per IV.* If this option is selected, each independent variable occurring in at least one node rule is represented by a column in Node Report. For any given rule (row), if a condition for the variable occurs in the rule, this condition (in the form of a value, a set of values, or an interval) is displayed in the column representing this variable. See the Node Report. topic for details.



Overview Report

The Overview Report view in datasets displays the dataset structure and univariate statistics such as the mean, maximum, and minimum values, the number of missing values, etc. See the topic Overview Report for detailed information about the available statistics.

To customize the report for the current dataset, select the **Options** command from the **Tools** menu or click the **Options** button in the toolbar.

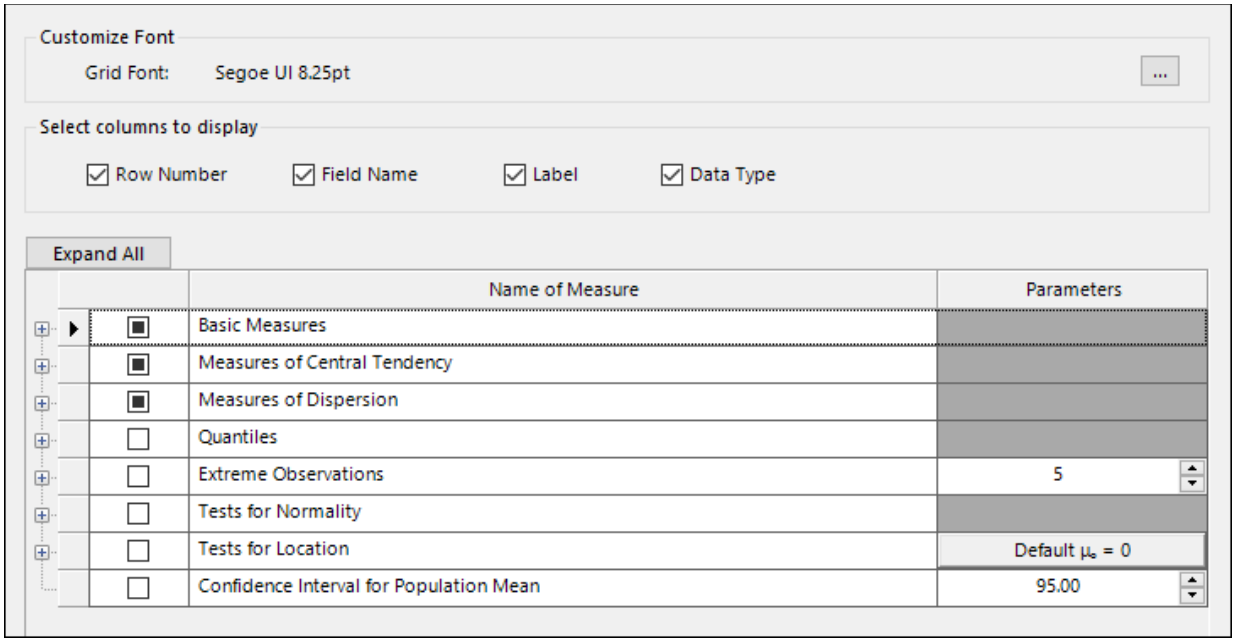
To set the columns to be displayed by default in the Overview Report for all *new* datasets, select **Tools | Preferences** instead of **Options** and choose the Overview Report tab. Changes in Preferences do not affect the datasets already created.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Select the statistics to be displayed in Overview Report. To hide unnecessary statistics, deselect the corresponding boxes. To change the default font for Overview Report, click the button to the right of the Grid Font name and select the desired font.

There are 8 groups of statistics with the total of 40 different statistics available. You can expand any group by clicking on the plus sign on the left, or expand all groups by clicking **Expand All**. The application default includes cardinality, unique count, the number and percentage of unique values, minimum, maximum, mean, and standard deviation.

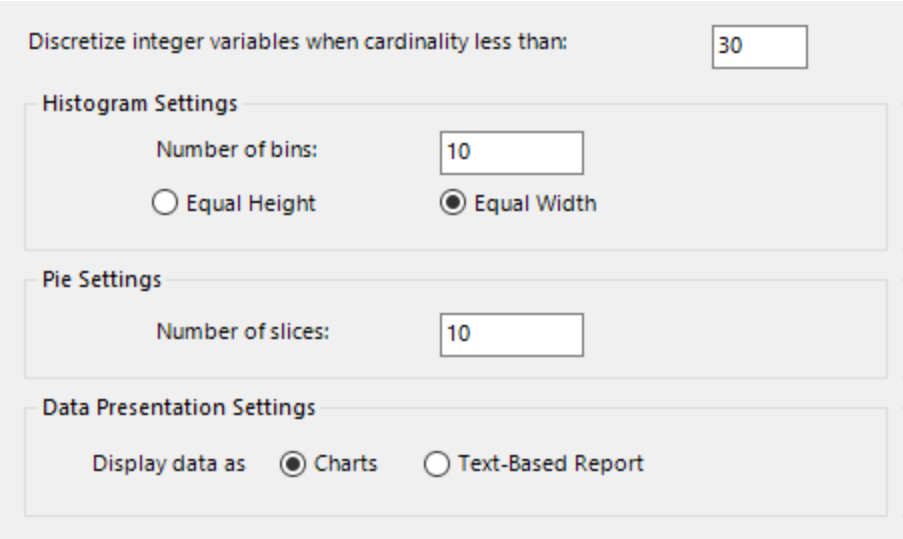
For the detailed description of all available statistics, see the Overview Report topic.



Segment Viewer

The Segment Viewer tab in the Dataset displays the distributions of the independent variables under each category of the selected segmentation variable. The general settings for the dataset Segment Viewer are specified in this tab.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Discretize integer variables when cardinality less than	The distribution of integer variables whose cardinality is less than or equal to this number will be displayed in the form of bar charts (histograms) by default. Otherwise, it will be displayed as pie charts. The default is 30.
Number of bins	Number of bins in which to distribute the data (for histograms).
Equal Height	When this is checked, the data is distributed so that each bin contains approximately the same number of observations.
Equal Width	When this is checked, the width of the intervals on the x-axis will be the same for all bins.
Number of slices	Specifies the maximum number of categories to display on the pie chart.

See also: Segment Viewer.

Split Report

The **Split Report** view of the Decision Tree displays the parameters of all splits at the currently selected node of the decision tree, provided that their significance (i.e., the value of the *Info* parameter) exceeds the Significance Threshold specified in the Split Report preferences or options. The default value of Significance Threshold is zero. Adjust the value of this parameter as desired if you want to filter out the splits for the least significant variables in Split Report.

For more details on the informativeness of splits and the meaning of the *Info* parameter, see Section 1.2.1 and the subsequent sections of Mathematical Formulations: Decision Tree.

See also: Split Report.

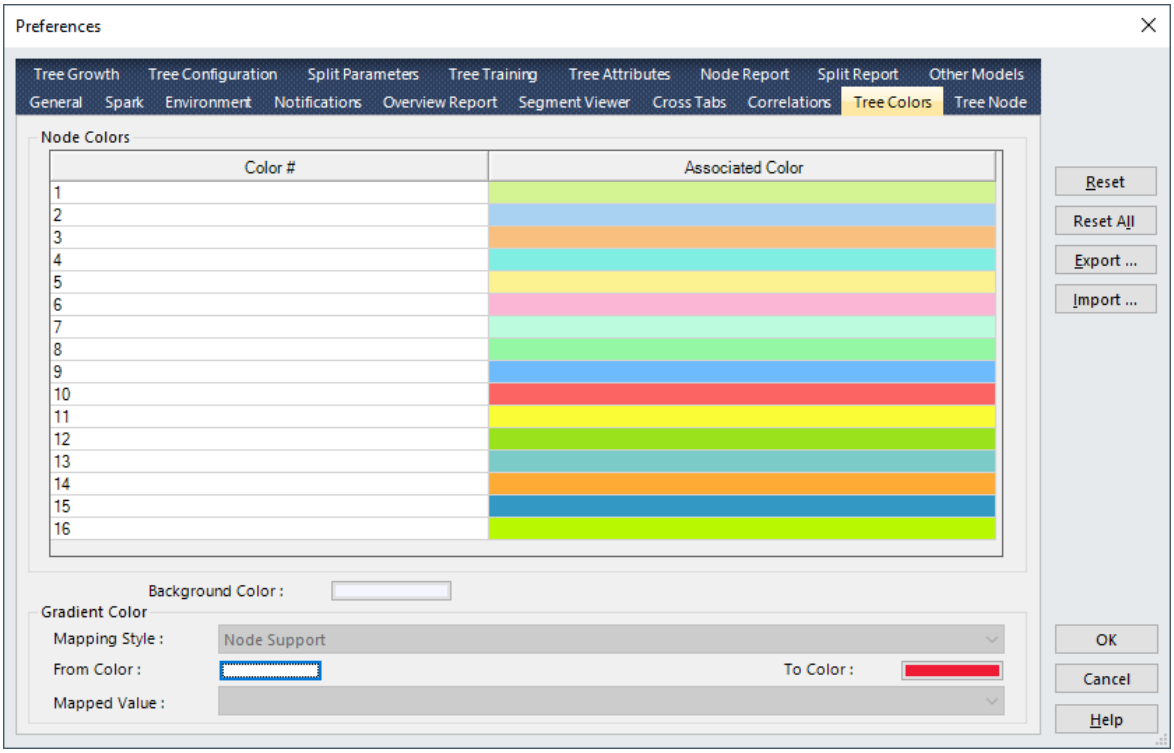
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Tree Colors

Color options and preferences depend on the tree type. The first subsection of this topic deals with Decision Tree color options and preferences. Strategy Tree colors are described in Subsection 2.

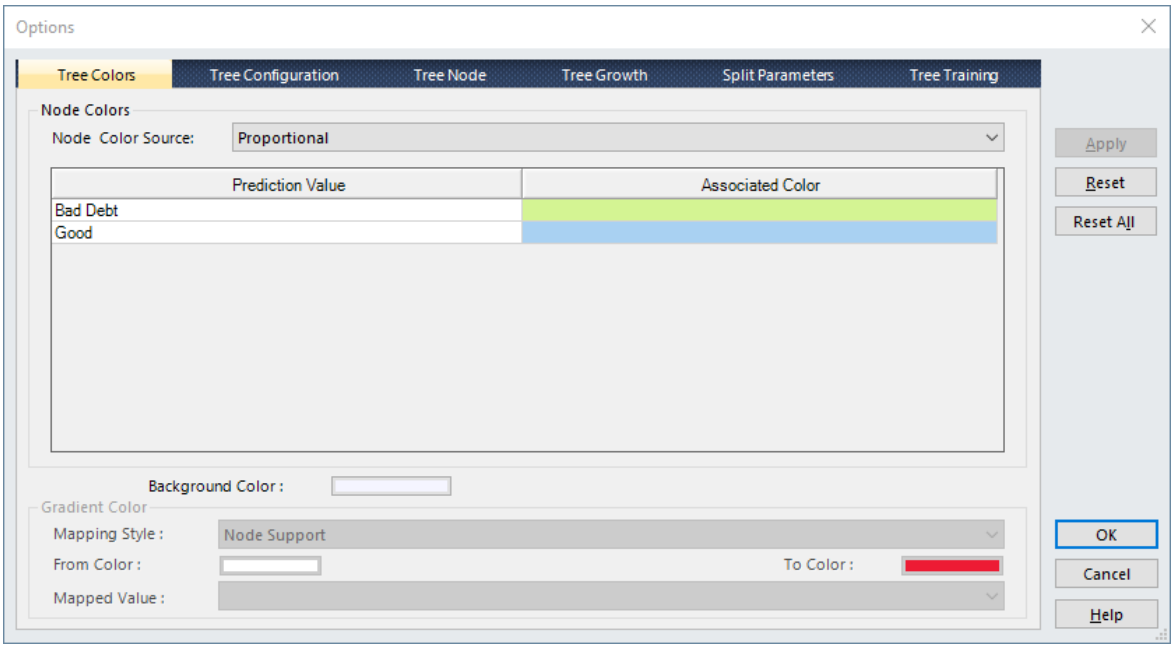
1. Decision Tree Colors

The default color settings for the Decision Tree and its background are specified in the **Tree Colors** tab of the Options dialog (menu: **Tools | Options**) and Preferences dialog (menu: **Tools | Preferences**). These settings affect the Tree tab of the Decision Tree object.



- For trees with **discrete** dependent variable (DV), the Node Colors list in this dialog box assigns a color to each value of the dependent variable. The values are listed in alphabetical order. This setting applies if the Node Color Source is set to *Proportional* in **Tools | Options | Tree Colors.**

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- The default color settings for trees with **continuous** DV are defined by the Gradient Color options (Mapping Style and From/To Color).
  - Mapping Style determines which parameter is the Gradient Color is based on. The options are: *Node Support* (the total number of records in a node), *Average* (the Average value of the DV in a node), and *StdDev* (the Standard Deviation of the DV in a node).
  - From/To Color: determines the color boundaries on the spectrum. The tree nodes will be assigned the colors within these color boundaries. For example, if the Mapping Style is *Average*, the color chosen as the From Color will be assigned to the node with the minimum value of Average, and the color chosen as the To Color will be assigned to the node with the maximum value of Average.

Changing the colors:

To assign a custom color to a DV category, click in the corresponding row of the Associated Color column. To specify a background color for the tree, click in the Background Color field. To specify the color boundaries for the Gradient Color style, click in the From Color or To Color fields.

The color palette will appear. Select the desired color and click OK to apply the change.

2. Strategy Tree Color Options

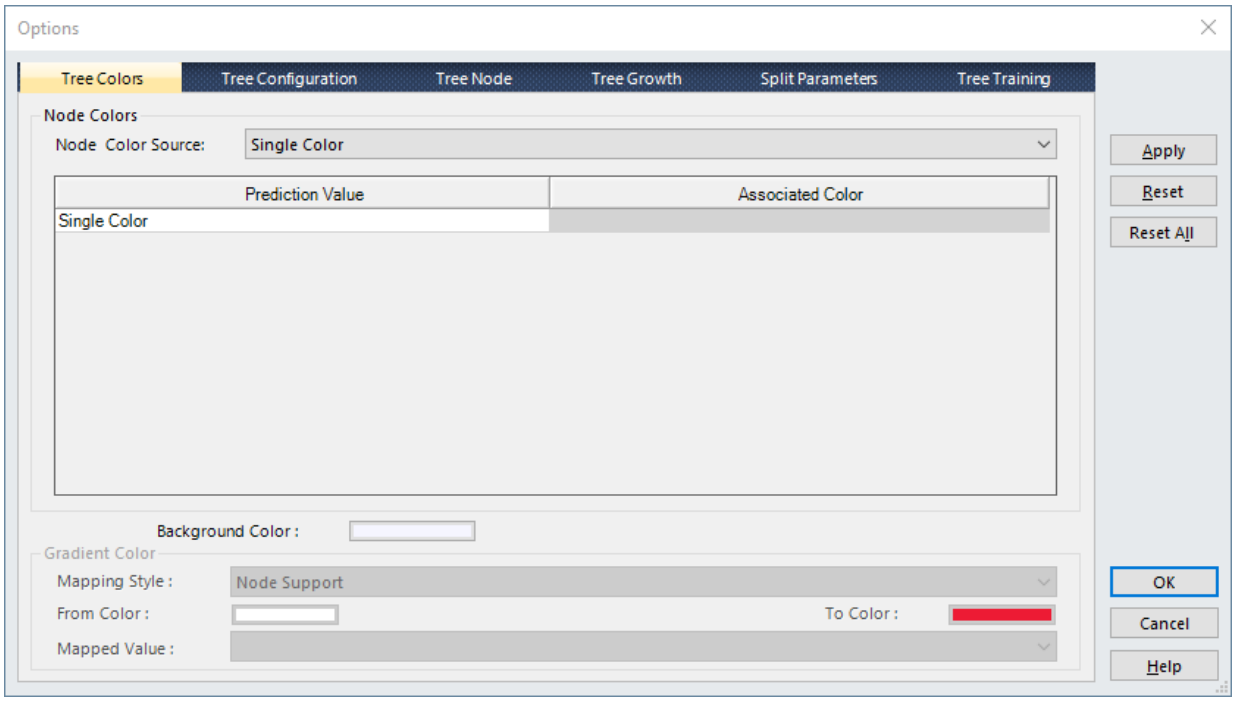
There are five Strategy Tree color schemes, depending on the selected *Node Color Source*.

The *Proportional Calculations* and *Gradient* schemes are based on the values of existing node calculations. The *Expression* color scheme allows you to enter a custom expression to assign node colors based on a specific condition. The *Treatments* scheme allows you to color nodes based on the *treatments* assigned to them.

a) Single Color

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

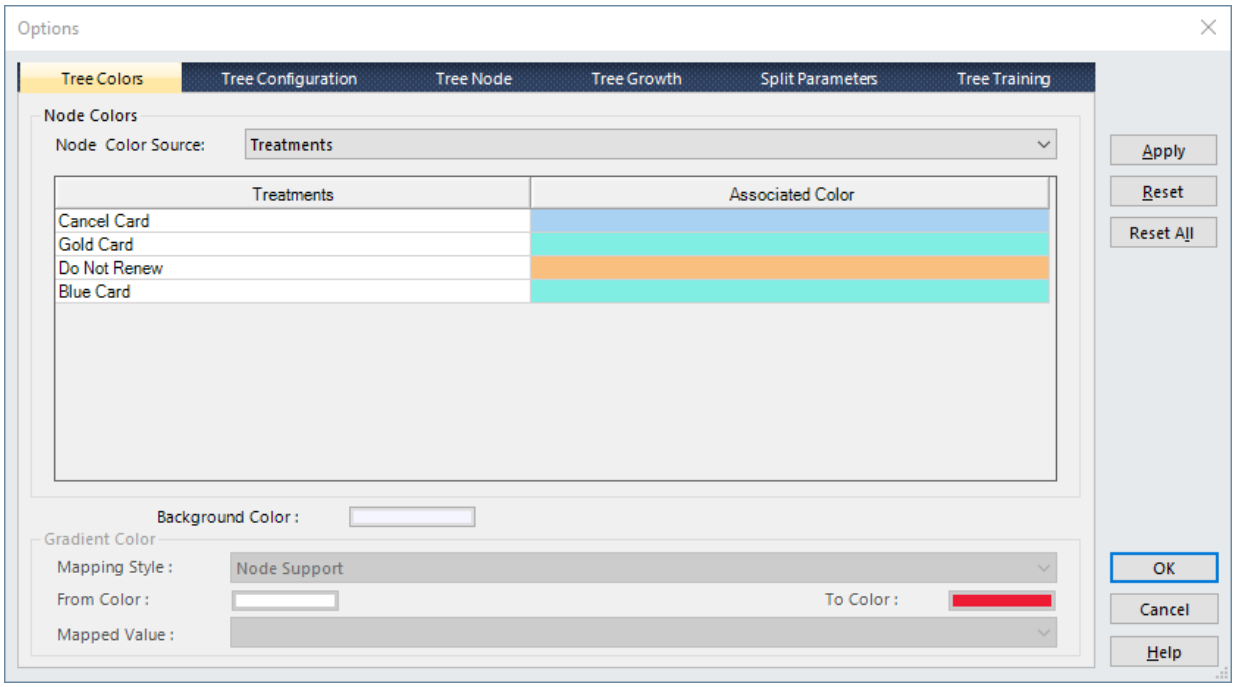
The first, and the simplest scheme is ***Single Color***, where all nodes have the same color. The default is gray. You can change it by clicking in the Associated Color column.



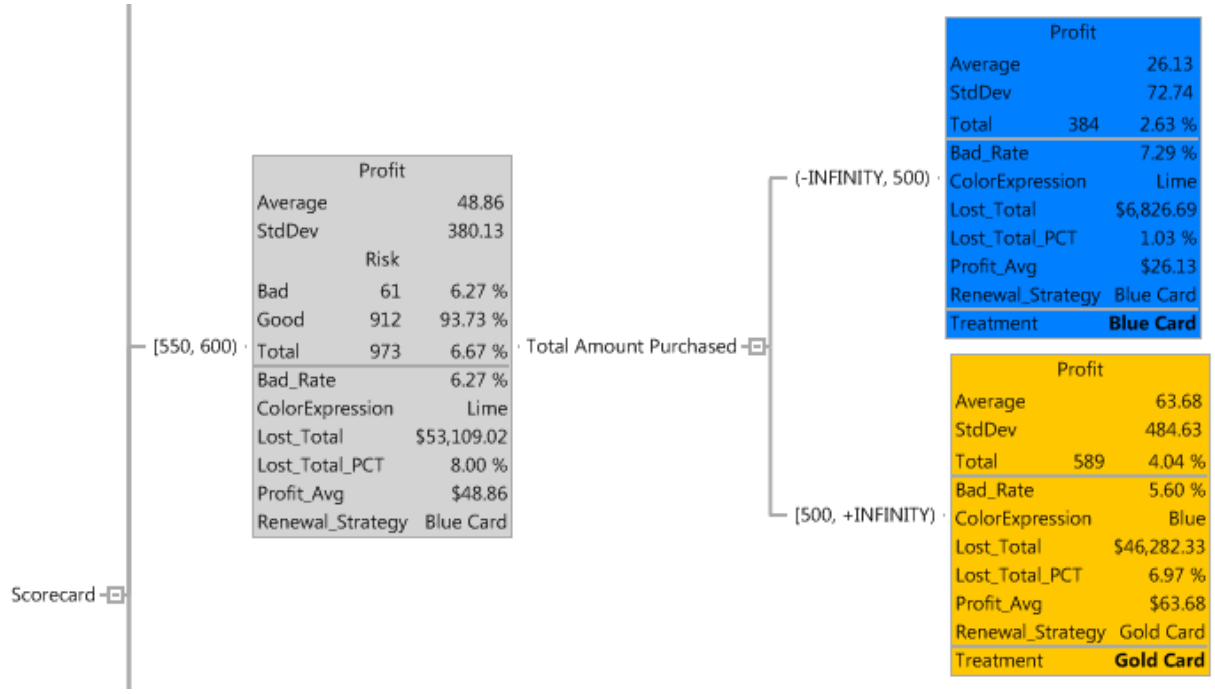
b) Treatments

The **Treatments** option is available in the Node Color Source list if there is at least one treatment defined in the Strategy Tree. If you select Treatments as Node Color Source, all available treatments are listed in the left column and the associated colors are shown in the right column. Click in the Associated Colors column if you want to change the default colors for the treatments. In the example below, there are three treatments in a Strategy Tree defining a renewal strategy for credit card accounts: 'Cancel Card', 'Blue Card' offer, and 'Gold Card' offer.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



A fragment of the Strategy Tree with this color scheme is shown below.



c) Gradient Calculations

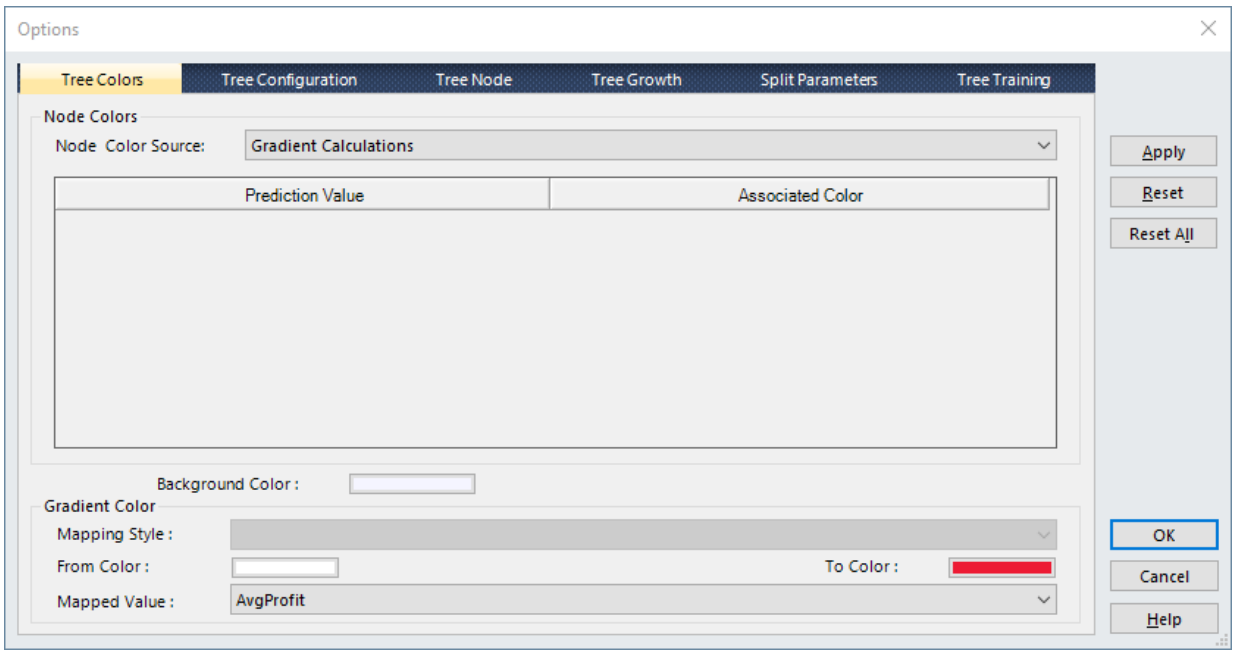
The **Gradient Calculations** color scheme uses gradations of color to express the numerical value of a node calculation. The color range (*From Color*, *To Color*) corresponds to the range of values

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

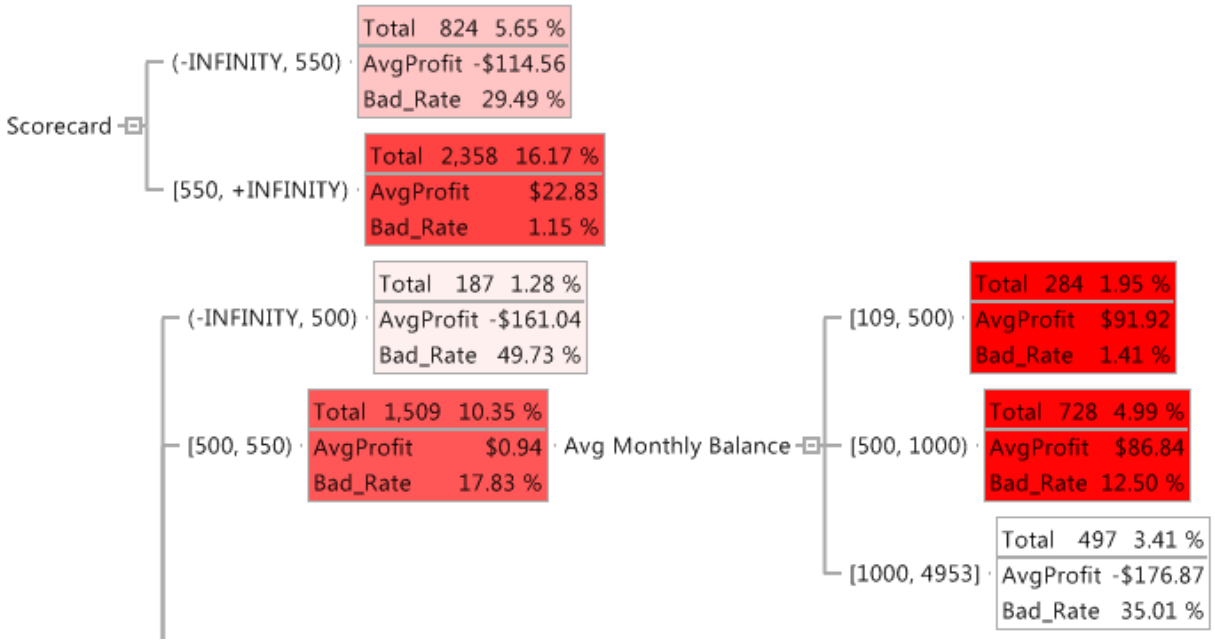
of the calculation. See the Options dialog below. The higher the calculation value at the node, the closer its color is to the color on the right-hand side (*To Color*). The lower the calculation value at the node, the closer its color is to the color on the left-hand side (*From Color*). The node calculation can be selected from the ***Mapped Value*** list.

The default color range is from white to red, so, by default, the calculation value is expressed with the gradation of red. Click on the From and To Color fields if you want to change the default color range.

In the example below, the calculation called *AvgProfit* (average Profit) is selected as Mapped Value. A fragment of a Strategy Tree with this color scheme for AvgProfit is shown.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.



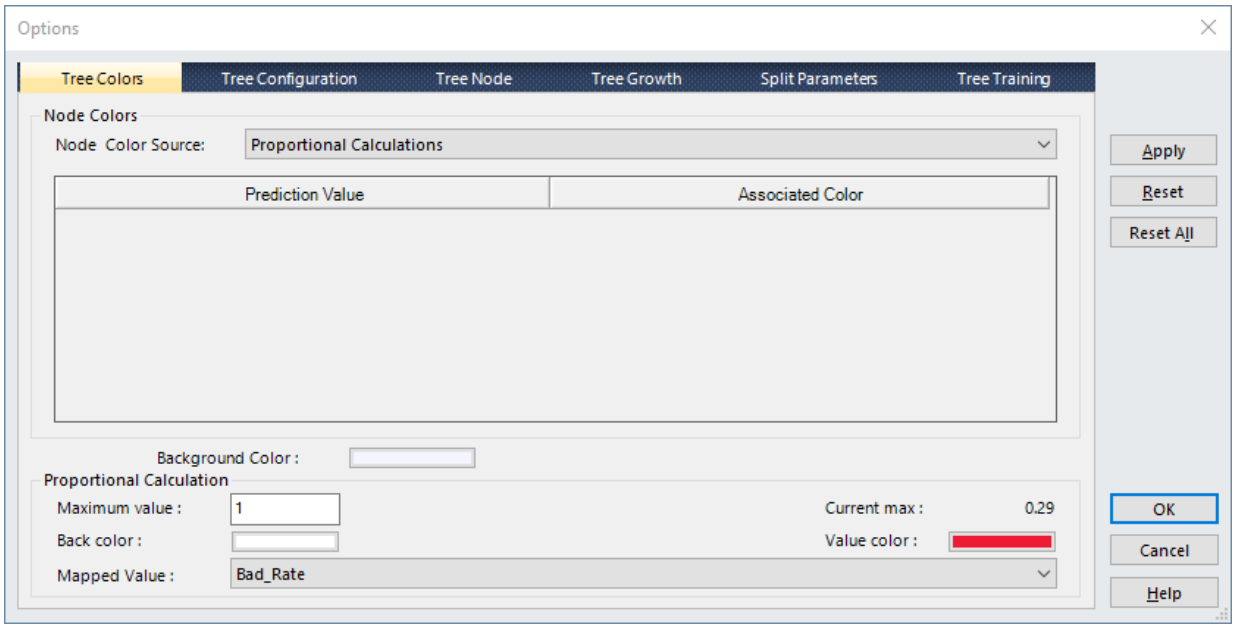
The higher the value of AvgProfit, the more red the node appears.

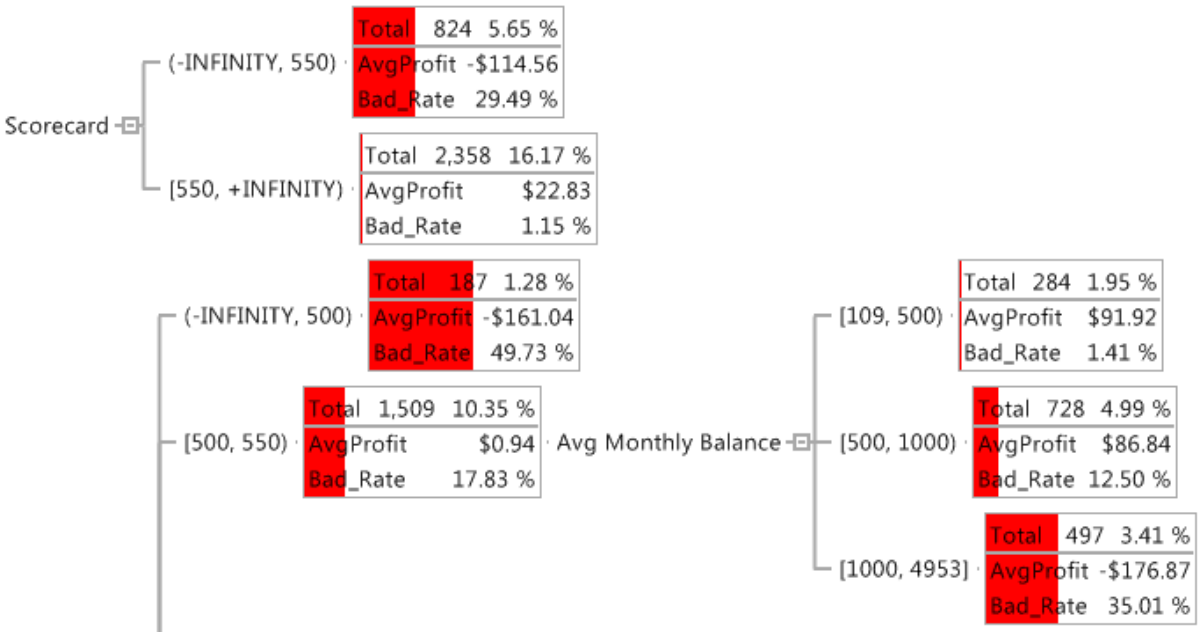
d) Proportional Calculations

In the **Proportional Calculations** color scheme, portions of nodes are colored to express the numerical value of a calculation. This color scheme is suitable mainly for positive-valued calculations, especially discrete distributions such as percentage of a specific value.

Select the desired node calculation from the ***Mapped Value*** list. The higher the calculation value at the node, the larger is its portion colored in *Value Color,* which is red by default. The rest of the node has color selected as *Back Color,* which is white by default. See the Options dialog below. As a result, nodes at which the calculation value equals the *Maximum value* specified in the Options dialog are entirely colored in *Value Color.* Nodes with zero value of the calculation are entirely colored in *Back Color*.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



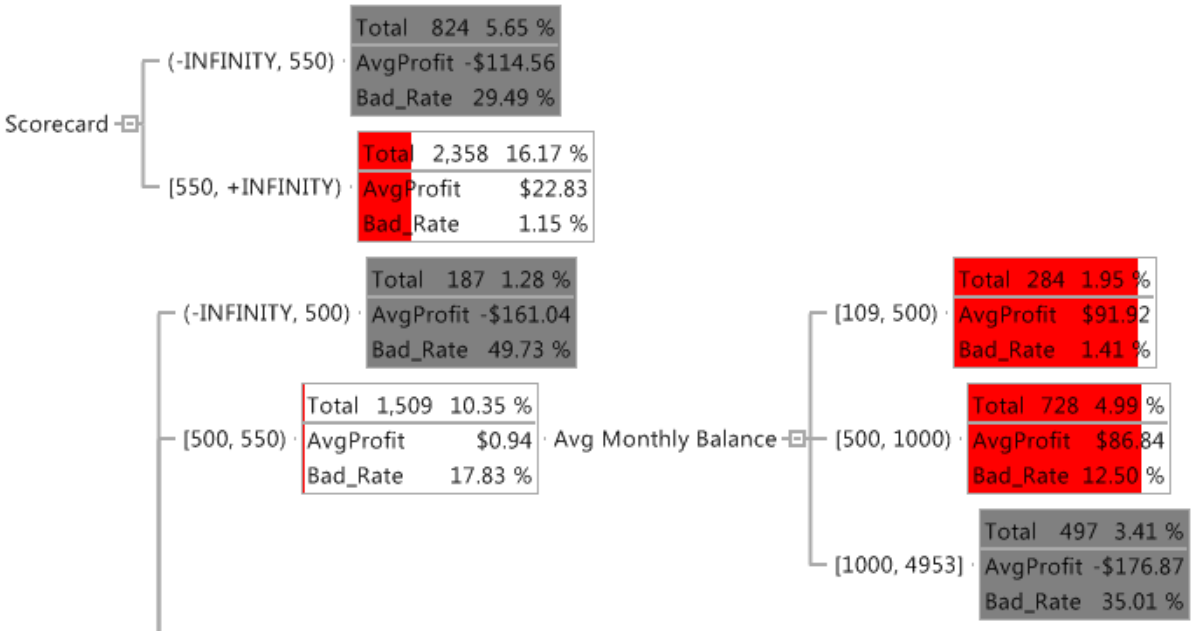


In the example above, approximately one third of the bottom right child node is colored in red because *Bad_Rate*=35.01% (0.3501), and 0.3501 is approximately one third of *Maximum value* = *1.00* specified in the Options dialog.

Note that you can set the *Maximum value* to be as high or low as desired. Therefore, at some tree nodes the calculation value may be higher than the *Maximum value*. In this case the *Current max* shown in the Options dialog exceeds *Maximum value.* Nodes at which the value of the calculation is either negative or exceeds the *Maximum value* are entirely grey.

For example, if we choose *AvgProfit* as Mapped Value and set the *Maximum value* to 100, then the same tree fragment will look as follows:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



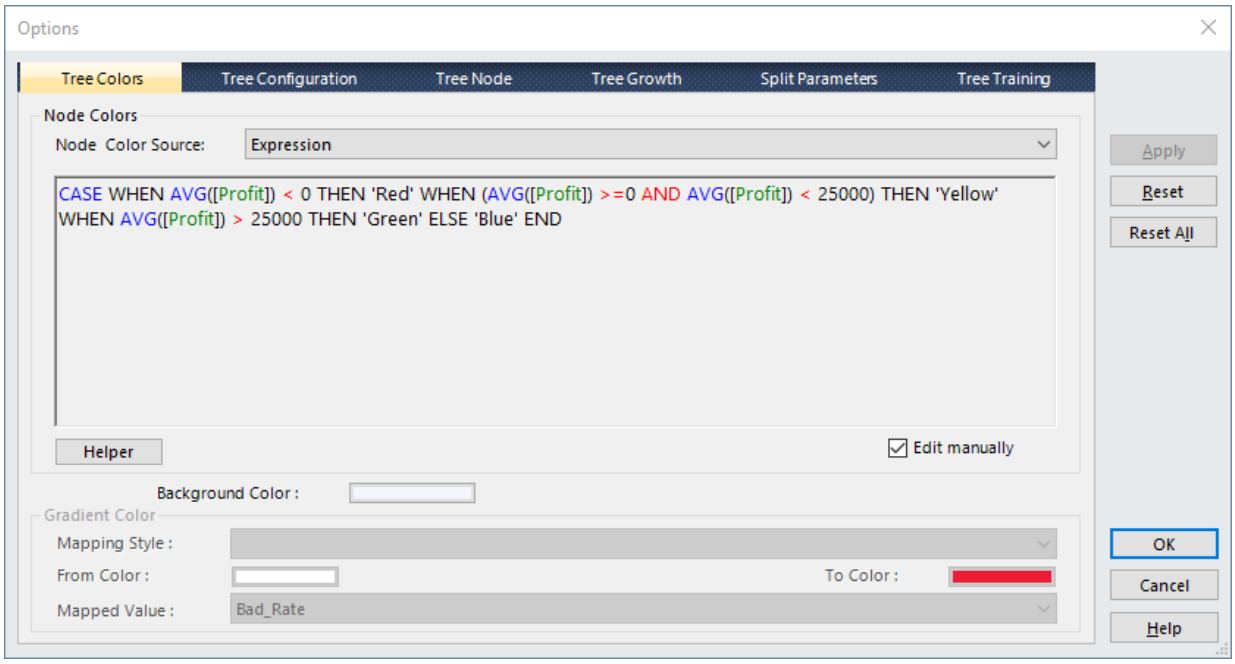
AvgProfit = 91.92 at the top-right node, so approximately 9/10 of it this node is colored in red because 100 is the *Maximum value* specified in the Options dialog. The bottom-right child node is grey because the value of *AvgProfit* is negative (-176.87). Any child node with *AvgProfit* > 100 would also be grey.

Unlike the predefined *Maximum value,* ***Current max*** changes as you grow the tree. *Maximum value* is a fixed, user-defined reference point and the threshold beyond which the nodes will be colored in grey.

e) Expression

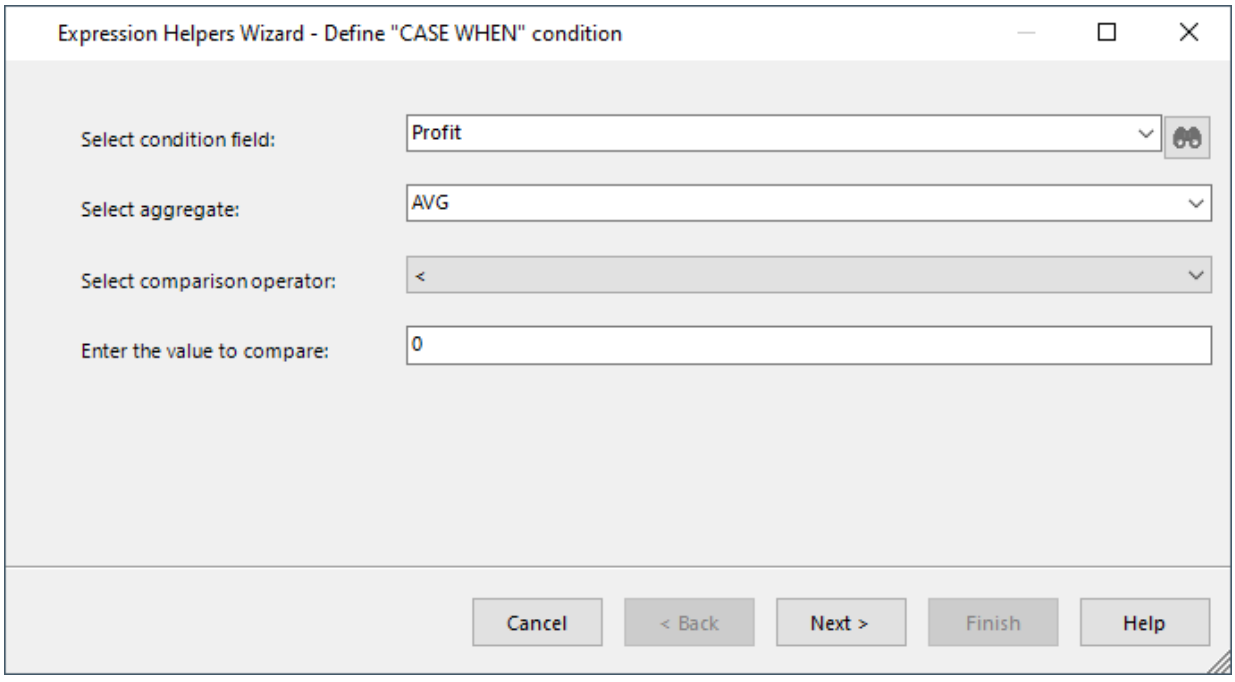
This option allows you to specify an arbitrary expression defining node colors. For example, this way you can assign different colors to nodes depending on the interval that the AVG([Profit]) calculation belongs to. See an example below.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



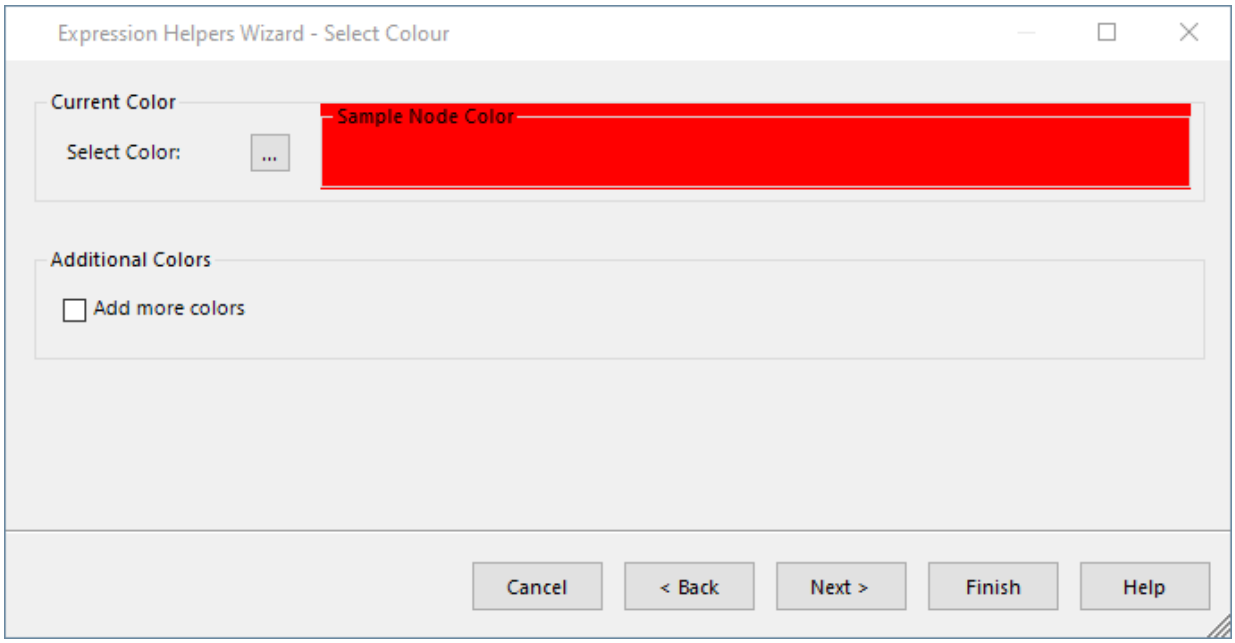
To enter an expression manually, check the *Edit manually* box.

Alternatively, you can click the Helper button to run the wizard that lets you specify conditions and the corresponding colors. Specify the CASE WHEN condition. In the example below, the resulting condition will be CASE WHEN AVG([Profit]) < 0



Click Next and specify the color to assign to nodes with AVG([Profit]) < 0.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



At this stage, the resulting expression will be CASE WHEN AVG([Profit]) < 0 THEN 'Red'.

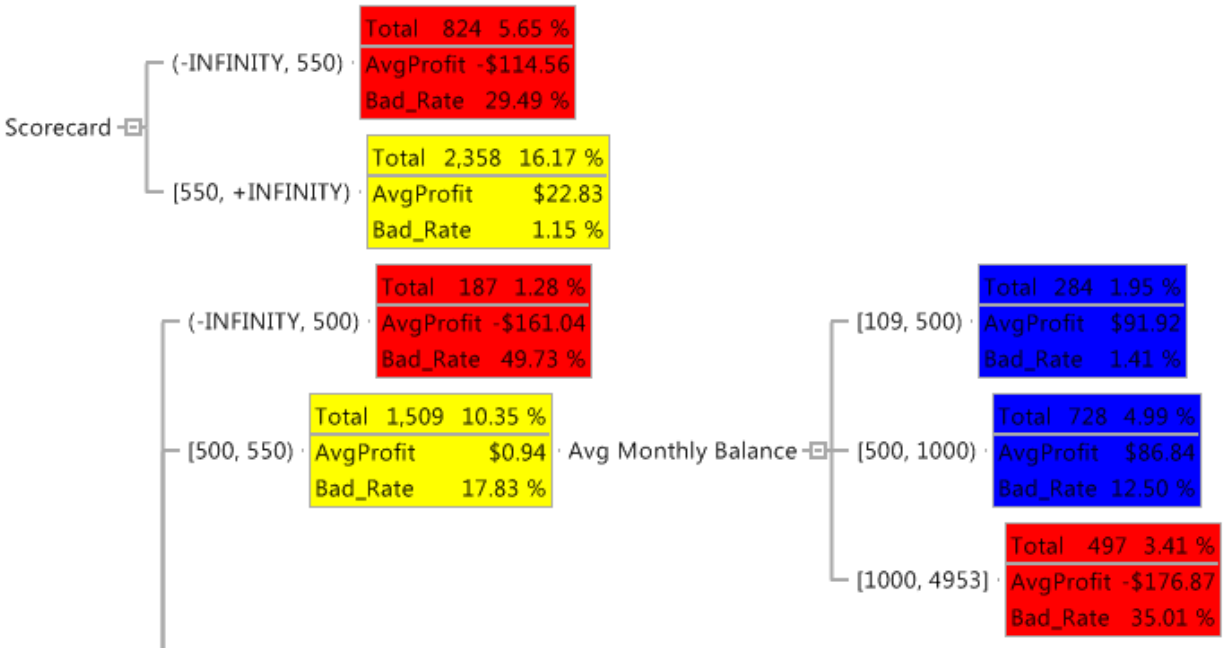
Select the *Add more colors* option if you need to add more conditions, for example specify a different color for AVG([Profit]) > 100.

In the above example, the resulting expression is obtained in four steps of the wizard using the *Add more colors* option:

CASE WHEN AVG([Profit]) < 0 THEN 'Red' WHEN AVG([Profit]) < 25 THEN 'Yellow' WHEN AVG([Profit]) < 50 THEN 'Lime' ELSE 'Blue' END

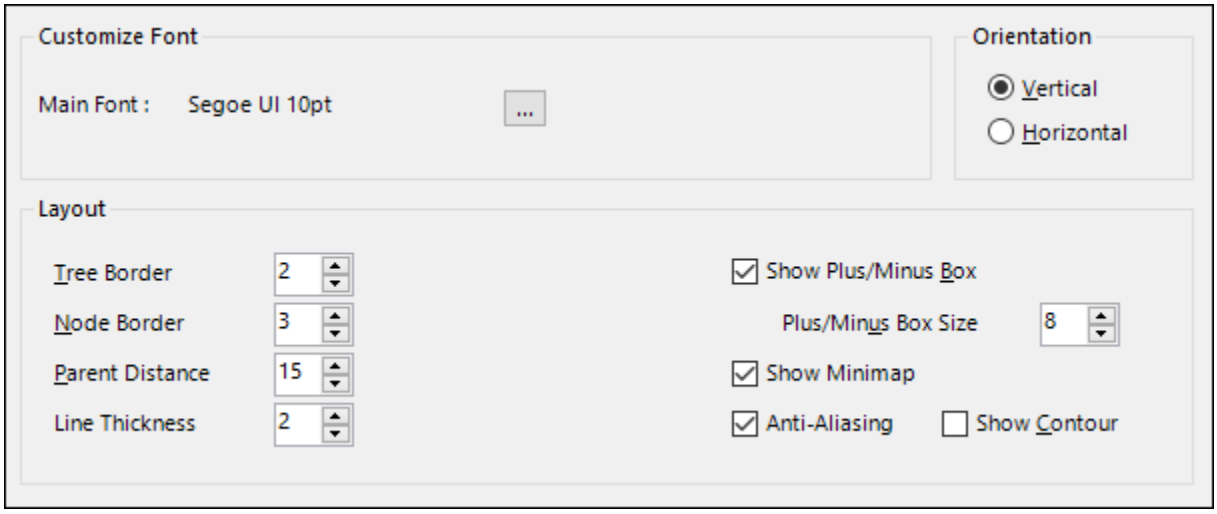
A fragment of the resulting tree is shown below:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Tree Configuration

The settings in the Tree Configuration tab of the Preferences dialog define the default font, orientation, and layout of the tree.



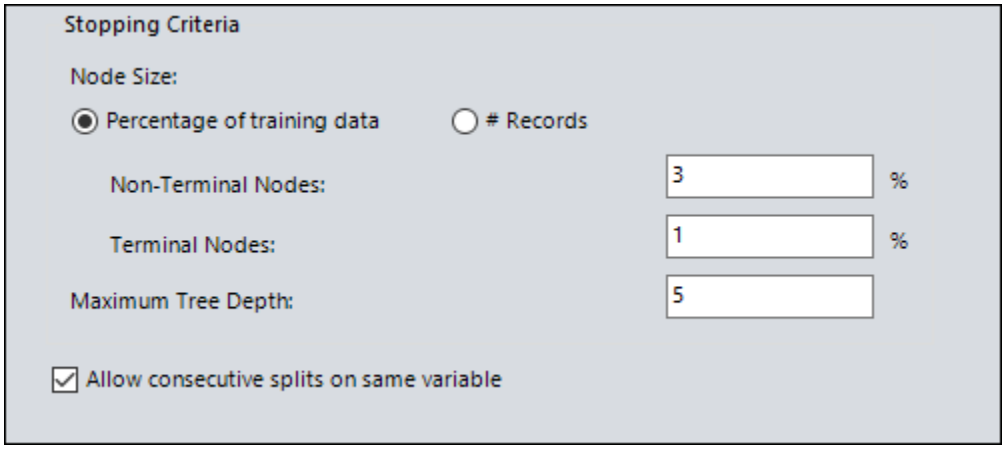
Main Font	The font used in the decision tree nodes and marginal rules. The default is Tahoma, 10pt.
Vertical	The tree is displayed vertically. This is the default orientation for Decision Trees.
Horizontal	The tree is displayed horizontally. This is the default orientation for Strategy Trees.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Show Plus/Minus Box	Show or hide the plus/minus box beneath each node that has a split, which serves as a shortcut to the **Expand / Collapse** command under the **View** menu. If the plus is displayed, the tree branch under the node is collapsed. Clicking on it will expand the branch and the plus will turn into minus. Conversely, clicking on the minus sign collapses the branch. The plus/minus box is not shown by default. The size of the plus/minus box can be customized by adjusting the value in the Plus/Minus Box Size field. The default size is 8.
Tree Border	Determines the distance from the root node to the top of the Tree view window. The default is 2.
Node Border	Determines the distance between the adjacent tree nodes of the same level. The default is 3.
Parent Distance	Determines the vertical distance from the child nodes to their parent node. The default is 10.
Line Thickness	Determines the thickness of the lines connecting child nodes with their parent node. The default is 2.
Anti-Aliasing	Smooths the lines connecting child nodes with their parent node. This option is turned on by default.
Show Contour	Show or hide the contour around currently selected node and the whole tree branch beneath this node. Contours are not shown by default.

Tree Growth

The settings in the **Tree Growth** tab of the Preferences dialog define several constraints that control the tree growth performed with the **Find Split** or **Automatic Grow** operations: minimum node size, maximum tree depth, and allowing consecutive splits on the same variable.

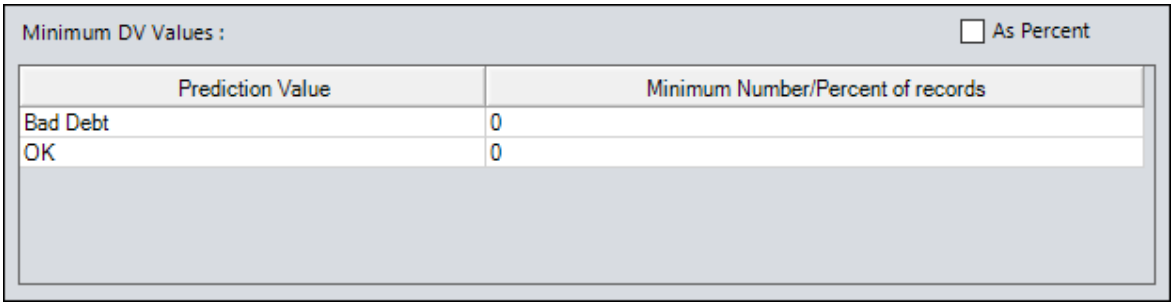


Node Size Limits in Percentage or	If the **Percentage** option is selected, the node size limits must be specified in percentage of total. This option is selected by default and is recommended,

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Number of Records	since it is logical to set these limits relative to the size of the input dataset. Select the option **# Records** if you want to specify the node size limits in absolute numbers of records.
Minimum Node Size for Non-Terminal Nodes	The minimum number of records for a node to contain before the **Find Split** or **Automatic Grow** operations are allowed to split it.
Minimum Node Size for Terminal Nodes	The minimum number of records allowed for terminal nodes.
Maximum Tree Depth for Automatic Grow	The maximum number of levels allowed for the **Automatic Grow** command to grow the decision tree. The number of levels includes the root node, so the depth of a tree with a single split is 2. Enter the value of zero (0) for unlimited depth (in that case, the depth is limited only by the minimum node size constraints defined in this dialog).
Allow consecutive splits on same variable	If this option is selected, splits on the same variable are allowed on two consecutive levels of the tree, on the same branch. This option is selected by default.

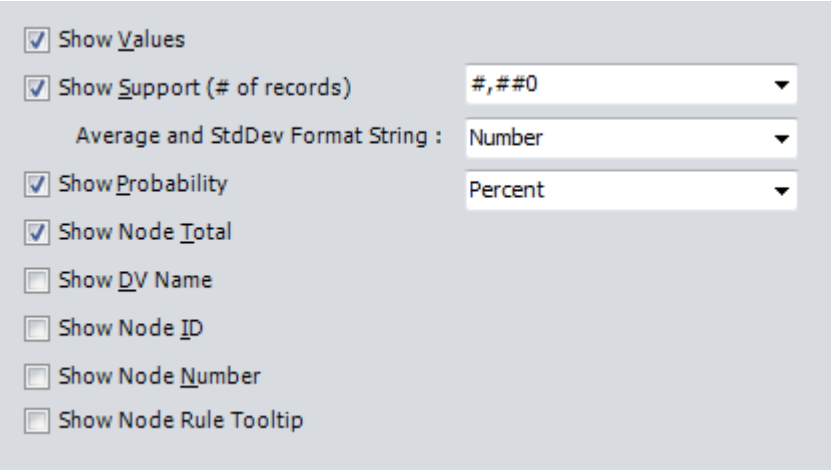
In each individual tree, the Tree Growth tab under **Tools | Options** also allows you to set the minimum limits for the number or percentage of the dependent variable values in the tree nodes. This applies to the **Find Split** or **Automatic Grow** operations. Enter the minimum numbers allowed for each DV value, if required. If you want to set the limits in terms of percentage of total, select the option "**As percent**". See the topic Automatic Grow for details.



Tree Node Settings

The settings in the Tree Node Settings tab of the Preferences dialog determine the information to be displayed in the tree nodes by default.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

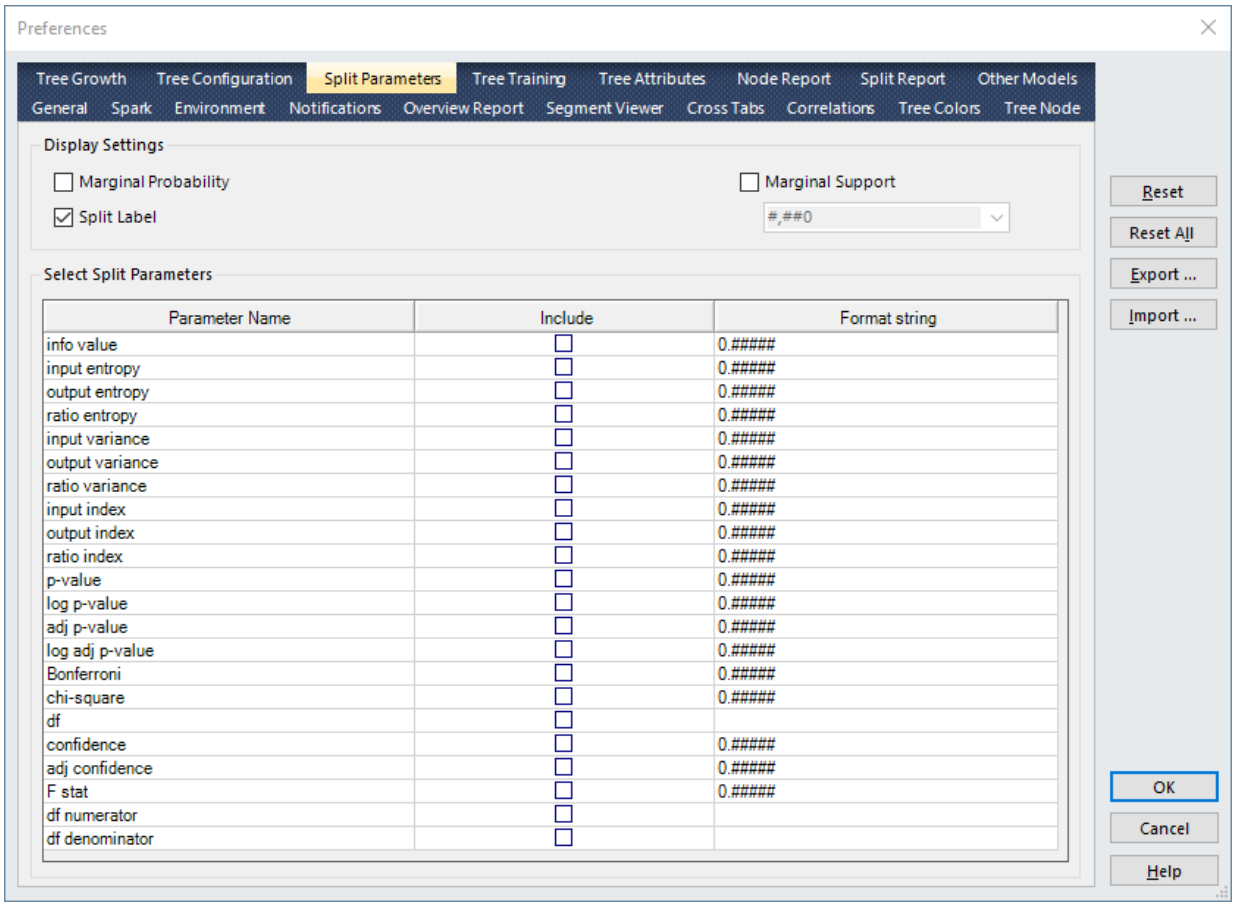


Show values	Shows the values of the dependent variable (for a discrete DV) or Average and StdDev (for a continuous DV) in the node.
Show Support (# of records)	Shows the total number of records in a node and the number of records for each DV value (for a discrete DV) or the values of Average and StdDev (for a continuous DV).
Show Probability	Shows the percentage of the number or records for each DV value (for a discrete DV) and the percentage of total number of records (if the Show Node Total box is checked).
Show Node Total	Shows the total number of records and percentage of total in each node.
Show Title	Shows the name of the DV in each node as a node title.
Show Node ID	Shows the node ID in each node.

Tree Split Parameters

The settings in the **Split Parameters** tab of the **Preferences** dialog determine the information to be displayed for each split in the tree. In the **Options** dialog, the choice of split parameters is limited to those applicable to the currently selected Measure of the active tree.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Marginal Probability Displays the probability of records being assigned to each node (in percent). For any node, the sum of the marginal probability values for all its child nodes must add up to 100%.

Split Label For each split in the tree, displays the name of the independent variable that defines the split rule.

Marginal Support If the split variable is discrete: For each value of the split variable, displays the number of all records in the node that contain this value.

If the split variable is continuous: Displays the number of records for each value range of the split variable in the marginal rules.

The Split Parameters list The list of all split parameters for all available measures. Each measure has a different set of split parameters. For any given tree, only the split parameters applicable to the measure of the current tree can be displayed.

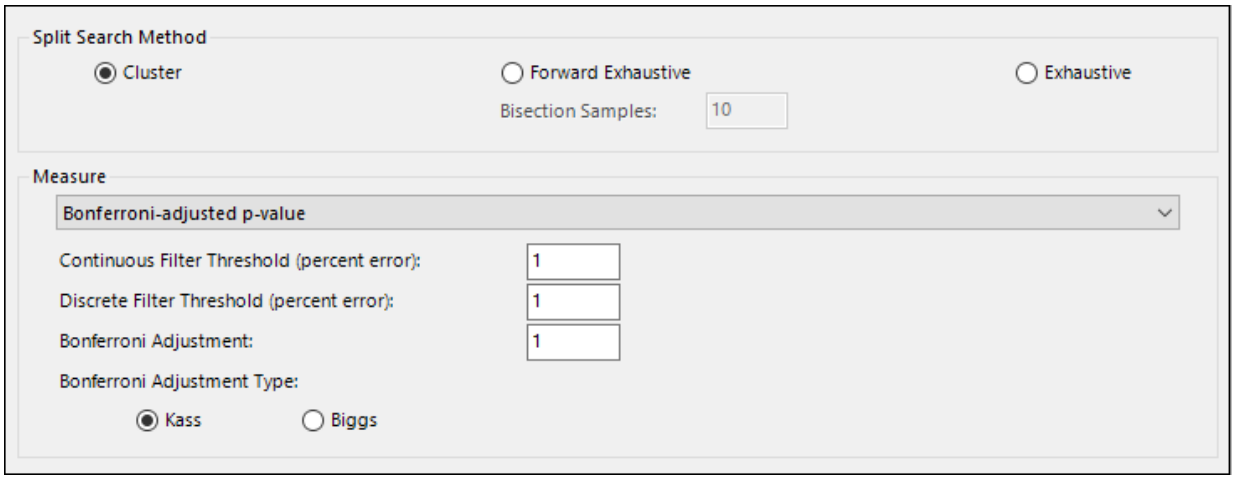
You can specify the display format for each split parameter in the Format String column of this dialog. In the above example, the confidence level (C) will be displayed with four and P-value with two digits after the decimal point, and chi2 will be displayed in scientific notation. For more information on format strings, see the topic Specifying Format in the Expression Editor section.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

For the explanation of parameters for each measure, see the topic Supported Measures in the section "Decision Trees --> Inserting a Decision Tree". Also see the Decision Tree section in Mathematical Formulations.

Tree Training

The settings in the *Tree Training* tab of the Preferences dialog determine the main parameters of the tree growth algorithm of the **Find Split** and **Automatic Grow** operations: **split search method**, **measure**, and the relevant thresholds and adjustments. They apply not only to the Decision Trees, but also to Ensemble Trees (Random Forest, Bagging, and Boosting) and the tree algorithm in Outlier Detector.



Cluster The *Cluster* method finds groups that maximize similarity within the groups and dissimilarity between the groups. Merging stops when either there is only one group left or there is no pair of groups that is "similar" enough with respect to the "merging threshold". The Cluster method is faster and more efficient than Exhaustive, but may potentially stop at slightly less optimal solutions. This method is the application default. See the Find Split topic for more technical details.

Exhaustive The *Exhaustive* method performs an exhaustive search to find groups that maximize statistical significance. Merging stops when there is only one group left. While merging, the sequence of merges is recorded in a special history buffer, noting the "best", global, solution found to date. When the merging process is finished, a re-initialization to the start point is performed, and the sequence of merges recorded in the aforementioned "history buffer" is performed. This process stops as soon as the "best" history entry (i.e., the noted best solution) is encountered. Splits derived from the Exhaustive method display the most significant relationships. See the Find Split topic for more technical details.

Forward Exhaustive This split search method is optimized for ordered variables. It begins with no split points, and adds them iteratively. It combines binary split search applied to the bins of the current (k-th) iteration and ternary splits on the bins obtained at the previous iteration (k-1). In addition, at each iteration it uses a discrete mesh of potential split points to increase the chances of achieving the global minimum p-value,

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

rather than stopping at a local minimum. The **Bisection Samples** parameter determines the number of nodes in this mesh. The default is 10. The admissible values are nonnegative integers. Zero means no sampling (i.e., a complete sweep is performed). The higher the value, the more likely the method is to find the optimal global solution – at the expense of performance. These optimizations affect only ordered variables – the variables whose Usage is 'ordinal' or 'continuous'. For splits on nominal variables, the regular Exhaustive method is used.

Measure

Six measures are available: Entropy Variance; Gini Variance; Unadjusted p-value; Unadjusted log p-value; Bonferroni-adjusted p-value, Bonferroni-adjusted log p-value. For the description of the measures, see Supported Measures in the section "Decision Trees -> Inserting a Decision Tree". Also see the Decision Tree section in Mathematical Formulations.

Continuous Filter Threshold (percent error)

Discrete Filter Threshold (percent error)

This option is applicable only for the Bonferroni-adjusted measures.

When analyzing data using the adjusted or unadjusted P-value measure, you run the risk that your data may display results obtained through chance, which can be potentially misleading. By adjusting the filter threshold value, you can lower the possibility that chance will play a part in your analysis - you determine how rigorously your data will be analyzed.

Continuous and discrete filter thresholds are for trees with continuous and discrete dependent variable, respectively. These thresholds are based on the confidence level (C) of a split, which is described in the section “Mathematical Formulations" (Decision Trees). The ***Info*** parameter in Split Report (same as Rank in the Go To Split dialog) is the value of C expressed in percentage. For example, if C=0.95, then Info = Rank = 95%.

Note: If you want the parameter C to be displayed at every split of the current tree (if the tree uses a P-value measure), select **Tools | Options** from the main menu, go to the Tree Split Parameters tab, and select the checkbox for C.

By specifying the filter threshold, you define the minimum value of Info (or Rank) that is of interest to you, so that the Find Split and Automatic Grow operations would not show the splits on independent variables whose rank is lower than 100% minus the specified error. For example, if you are interested only in the independent variables whose Rank is higher than 98%, specify 2 as the filter threshold (percent error). In this case, variables with Rank less than 98% (= 100% - 2%) will not appear in the Go To Split dialog and no statistics will be displayed for them in Split Report..

Bonferroni Adjustment

This option is applicable only for the Bonferroni-adjusted measures.

To minimize the discovery of chance relationships, Bonferroni adjustments let you adjust the level of significance used when testing the validity of a relationship.

This type of adjustment compensates for amount of redundant variables in your dataset. Normally, if a concept is captured by more than one variable, it has a disproportionate opportunity to enter the

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

analysis by chance. Use the Bonferroni adjustment to compensate for this possibility.

Bonferroni adjustments are quite conservative. Increasing the size of the adjustment increases the level of conservatism. It is not meaningful to set this value greater than the total number of variables in your dataset.

Bonferroni Adjustment Type (Kass / Biggs)

Two options are available for the Bonferroni adjustment type: **Kass** and **Biggs** (named after the authors of these methods, G.V. Kass and D. Biggs). The Kass type is the default. These options influence the number of child nodes in splits, and their effect depends on the type (nominal, ordinal, or continuous) and the cardinality of the split variable. For example, according to G. Ritschard, the Kass adjustment type tends to penalize high-cardinality predictors more than low-cardinality ones, and nominal predictors more than ordinal or continuous predictors. For details, see the articles by these authors listed in the References section of Mathematical Formulations.

Thus, for instance, if you increase the *Default number of bins* (the initial number of bins for the Find Split operation) in Attrubute Editor before growing the tree with Find Split or Automatic Grow, the Kass adjustment type may be more preferable as the Find Split algorithm will be less likely to pick variables with lower relevance to the target. The Biggs type may be preferable in other contexts. For guidance, see the articles cited above.

Tree Attributes

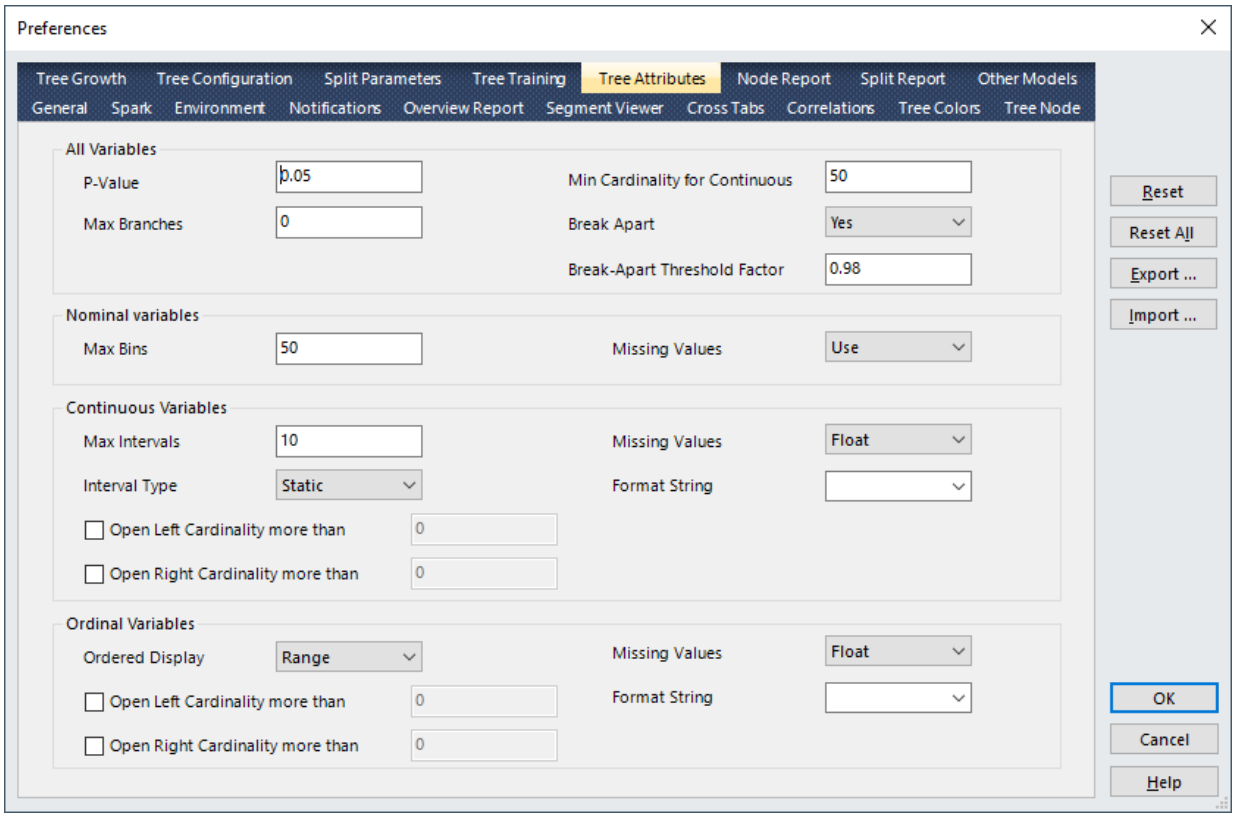
Global preferences for Tree Attributes can be set using the **Tools | Preferences** menu.

The settings in the **Tree Attributes** tab of the **Preferences** dialog determine the defaults in the **Attribute Editor** for Decision Trees and Strategy Trees. These are the default attributes for any new decision tree or strategy tree to be created.

(The Attribute Editor for Decision Trees and Strategy Trees is available under the Tools menu and also in the context menu of any tree when you click anywhere on the tree background. For detailed description of the Attribute Editor options, see the topic Using the Attribute Editor.)

NOTE: These settings do not apply to the dependent variable. They apply to independent variables only.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



P-Value The p-value threshold to merge child nodes during the Find Split procedure in trees. The default is 0.05. Increasing this value will make the splits more "bushy", i.e., values of any given the split variable will be less likely to be grouped.

Max Branches The Max Branches value determines the maximum number of branches (leaves) produced by Find Split on the given variable. See the topic Using the Attribute Editor for details. The default is 0 (which means "no fixed limit", aside from the upper limit set in the column Default # Bins, if applicable).

Min Cardinality for Continuous Numeric independent variables with cardinality higher than this value are considered to be continuous. The default is 50. If you set this threshold to **x**, then all numeric variables with cardinality >= **x** will have Usage = continuous, and all numeric variables with cardinality less than **x** will have Usage = ordinal (i.e., ordered discrete).

Max Bins (nominal variables) The maximum number of categories (bins) to display for nominal variables. The default is 50. If the maximum number of bins N is larger or equal to the cardinality of the variable, then each of its values is represented as a separate category. If N is less than the cardinality, the following mapping is applied: the first N-1 unique values are represented as separate categories and the rest are mapped into one category denoted by triple number sign ("###"). Note that this threshold also affects the scoring behaviour. See the topic Attribute Editor: Default Number of Bins for details.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Max Intervals (continuous variables)	The default for the initial number of intervals for continuous independent variables. (To form splits in a decision tree, continuous values are initially divided into discrete intervals where each interval has approximately the same number of observations. These discrete intervals are then treated as ordered and combined accordingly. The number specified here determines how many **initial** intervals are formed for this continuous variable prior to the ordered grouping. The resulting number of nodes (intervals) may be less than or equal to this initial value.) See the topic Using the Attribute Editor for more details.
Open Left Cardinality more than...	Applies to continuous variables or ordinal variables displayed in ranges. If this option is selected, the leftmost interval of any variable with cardinality higher than the specified value will be open from the left.
Open Right Cardinality more than...	Applies to continuous variables or ordinal variables displayed in ranges. If this option is selected, the rightmost interval of any variable with cardinality higher than the specified value will be open from the right.
Format String	This determines the default display format or precision for numeric independent variables. For the description of format strings that can be specified, see the topic Specifying Format in the Expression Editor section.

The Missing Values default is *Float* for both Ordinal and Continuous variables.

The default Ordered Display option for Ordinal variables is *Range.*

Environment

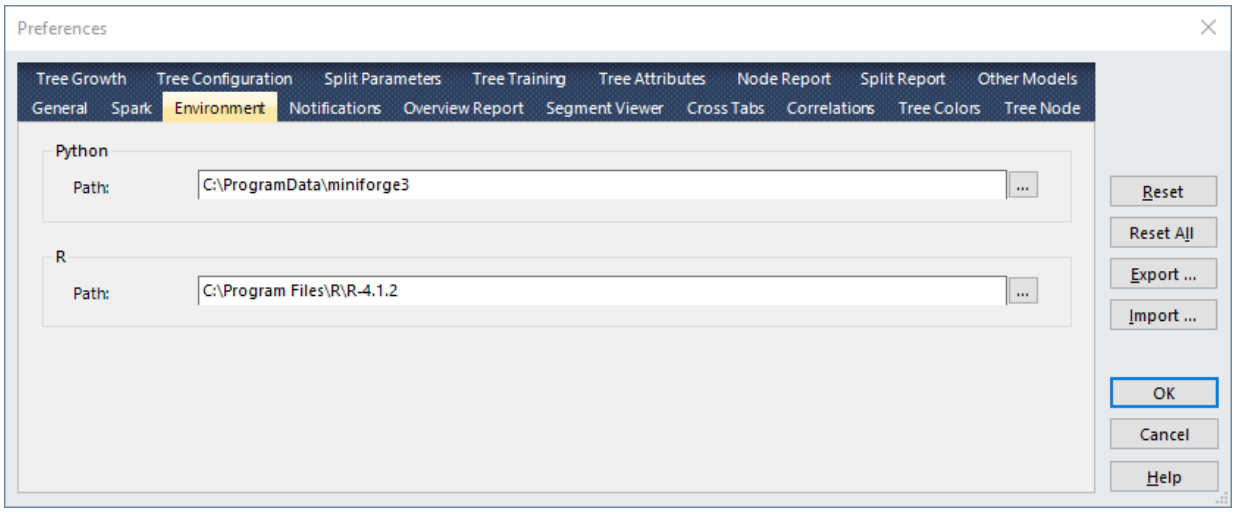
Environment Preferences determine the default location of R and Python executables and libraries for R and Python integration features - R Code and Python Code nodes.

***Note**: These settings are applicable in the Standalone configuration only. In the Client/Server configuration, R and Python environment used by KS Server is defined by the configuration on the KS Server host machine. Please contact your server administrator if you need to change R or Python environment used by KS Server.*

- The R path specified here is the location of the R **bin** folder where `R.exe` resides, as well as the location of the R **library** folder.
- The Python path specified here is the location of `python.exe` and the Python packages.

These paths determine the default value of the **Path to Executable** in the R Code and Python Code node wizards.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



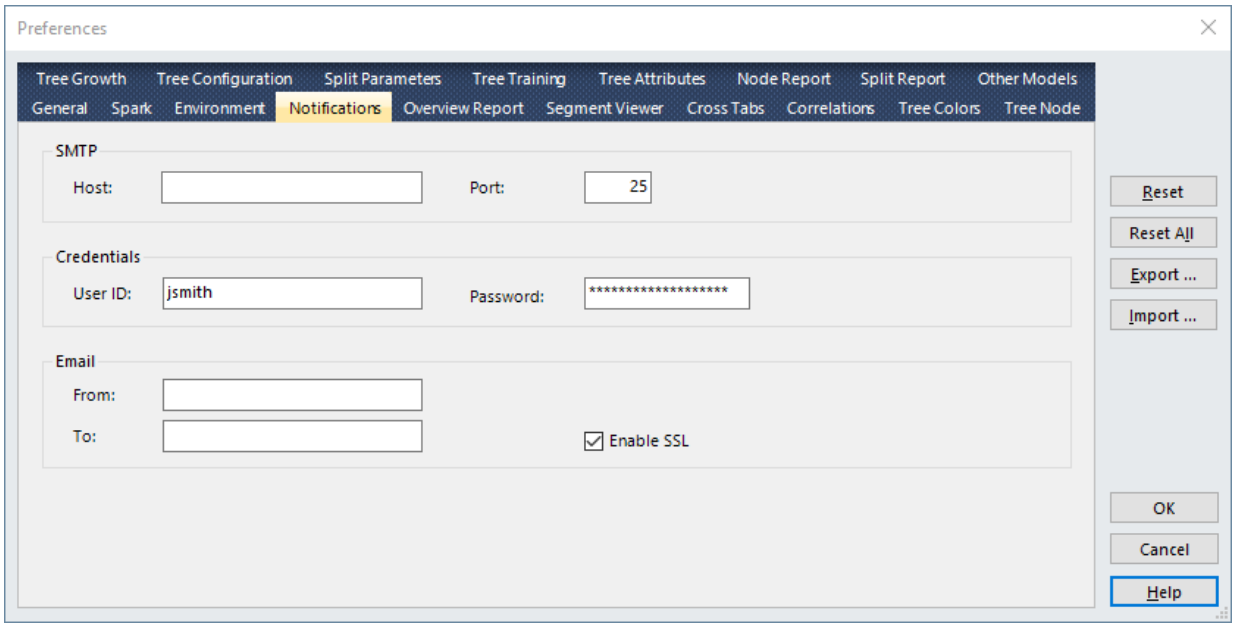
Initially, these paths are identified at the installation time by the KS Workstation installer. Change these paths in the Environment tab of Preferences whenever you change your R or Python environment (for example, after upgrading R or Python on your machine) or if you need to use a different version of R or Python when more than one version is installed.

Notifications

To configure the Email notifications for scheduled tasks, use the **Notifications** tab in the **Preferences** dialog accessible from the **Tools** menu. The notifications can be set for tasks scheduled in the Multiple Scenario Optimization node (if InsightOPTIMIZER features are enabled) and all Spark-integrated nodes (in Knowledge Studio for Apache Spark).

Specify the SMTP host name or IP address, SMTP port, User ID and password (optional), and the *From/To* Email addresses. The option **Enable SSL** is selected by default, but you may need to disable it (depending on the security settings in your Email client). If you are not getting notifications after configuring these settings, deselect the Enable SSL option and try again.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Auto-Documentation, Logging, and Command Line Tools

Desktop Document and Report Links

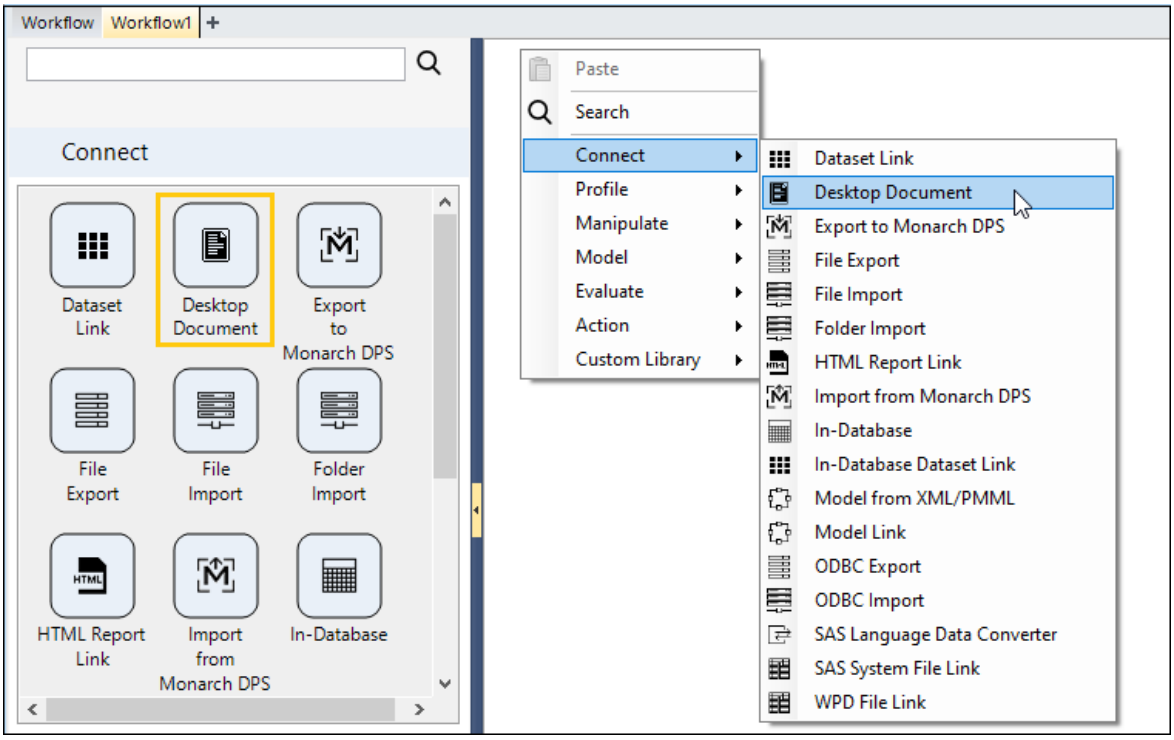
If you would like to add project documentation in an arbitrary format (PDF, Microsoft Word, Excel, plain text, etc.) and embed your document within the project workflow, you can do that using the **Desktop Document** node.

To add any document to the workflow of the current project:

- Right-click anywhere on the Workflow canvas background. From the context menu, select **Connect --> Desktop Document** as shown in the example below. A new **Desktop Document** node will be created and the Link Document dialog will open.

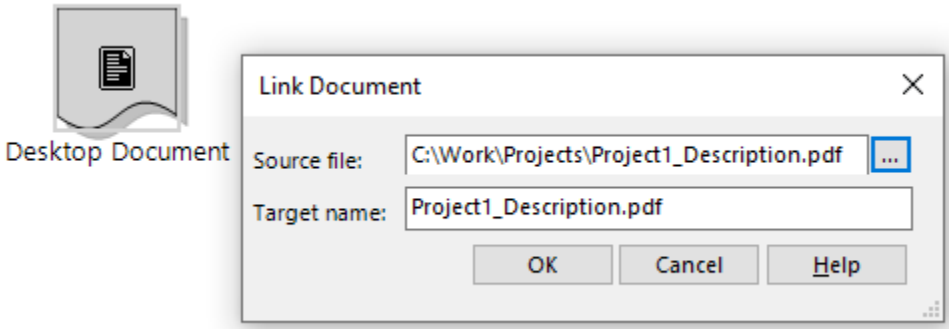
  OR:

- Drag the **Desktop Document** node from the **Connect** palette onto the canvas. Then double-click the node or right-click on it and select **Link** from the context menu.



- Browse for the desired file and specify the file path in the *Source file* field.
- Specify the name for the document node on the workflow canvas.

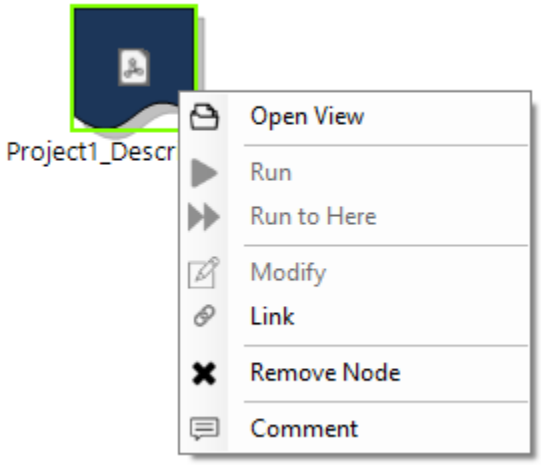
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- Click **OK**. The document file will be copied to the current project folder, shown in the Project Explorer, and linked as a new Desktop Document node on the Workflow canvas.



Double-click the document node to open it or right-click on it and select **Open View**. The document will be opened by the program currently associated with its file extension. If Windows cannot find a program associated with this file extension, it will display a standard dialog prompting you to select an application for opening this file type.



You can also link a different document file to this node at any time by selecting the **Link** command from the context menu of the node. In this case the previously linked document file will still remain in the project folder, but the node will now be linked to the new file, which will also be copied to the project folder.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

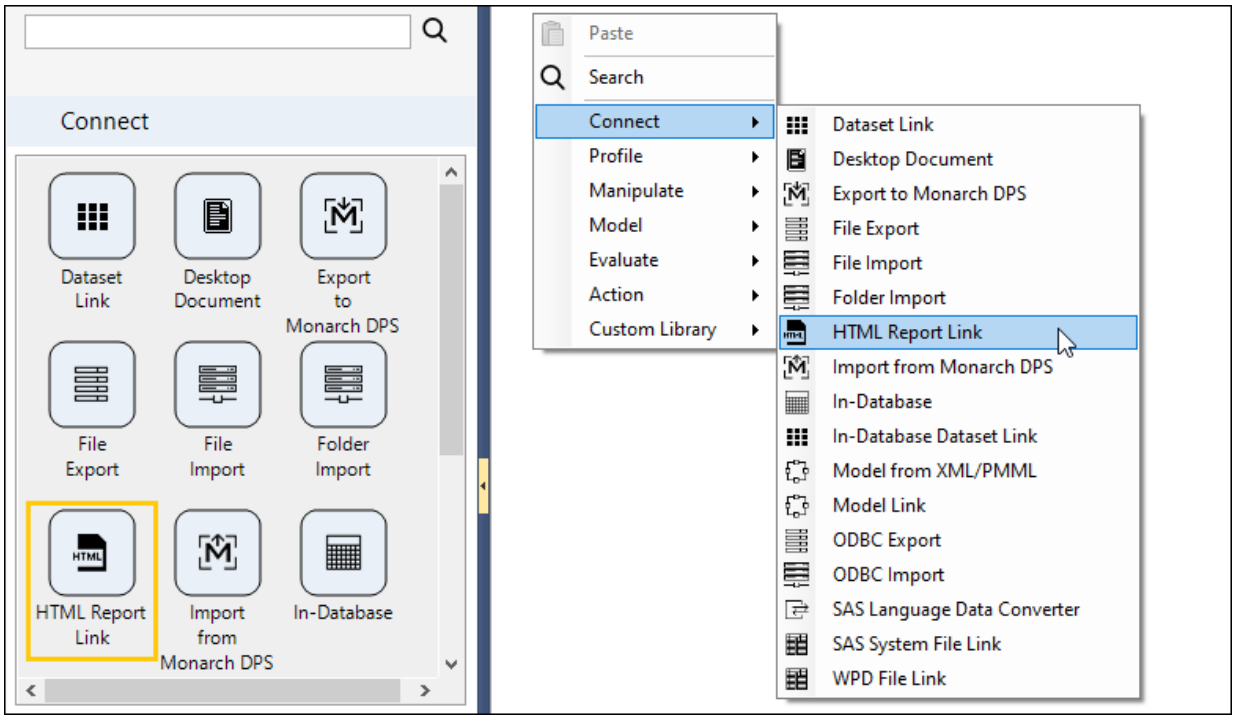
If you would like to add project documentation in the HTML format and embed an HTML file within the project workflow, you can do that using the **HTML Report Link** node.

To add an HTML report to the workflow of the current project:

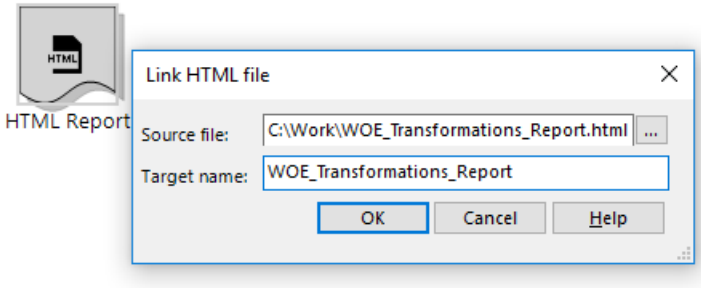
- Right-click anywhere on the Workflow canvas background. From the context menu, select **Connect --> HTML Report Link** as shown in the example below. A new **HTML Report Link** node will be created and the Link HTML File dialog will open.

  OR:

- Drag the **HTML Report Link** node from the **Connect** palette onto the canvas. Then double-click the node or right-click on it and select **Link** from the context menu.



- Browse for the source HTML file and specify the file path in the *Source file* field.
- Specify the name for the report node on the workflow canvas.

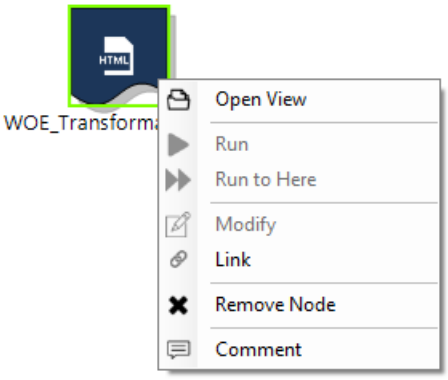


- Click **OK**. The source HTML file will be copied to the current project folder, shown in the Project Explorer, and linked as a new HTML Report node on the Workflow canvas.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Double-click the report node to open the report or right-click on it and select **Open View**. You can also link a different HTML file to this node by selecting the **Link** command. In this case the previously linked HTML file will still remain in the project folder, but the node on the workflow will now be linked to the new HTML file, which will also be copied to the project folder.



Self Documentation

The **Self Documentation** node from the **Action** palette provides an audit trail of the analytic process. It automatically documents all actions you have taken along the workflow path leading to the node it is attached to. The result of its execution is a Microsoft Word document containing snapshots of all views you have used (opened and modified) in all nodes in this workflow path. This helps you keep a record of your activities within the project: specific data profiling steps, developed decision trees, model training results, model evaluation charts, etc.

Before running the **Self Documentation** node, you can review the saved list of all activities performed in the workflow path and select the ones to be included in the final report (Word document). The charts and tables you worked with at the selected steps will be included in the document. The images for the views that have never been visited or initialized in the workflow are not included.

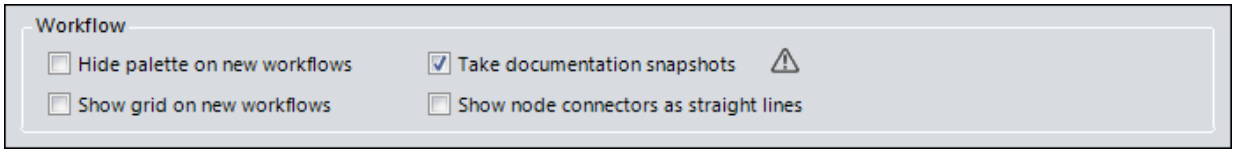
You can attach the **Self Documentation** node not only at the end of a workflow, but also anywhere in the middle of it. The path being documented is the one leading up to the input node of the Self Documentation node.

Note: The *Run to Here* command on the Self Documentation node context menu is never enabled. The node can only be run separately. If you run the workflow leading to a Self Documentation node that has been run previously, any existing snapshots in the report will be erased.

To add a self-documentation operation to the workflow:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

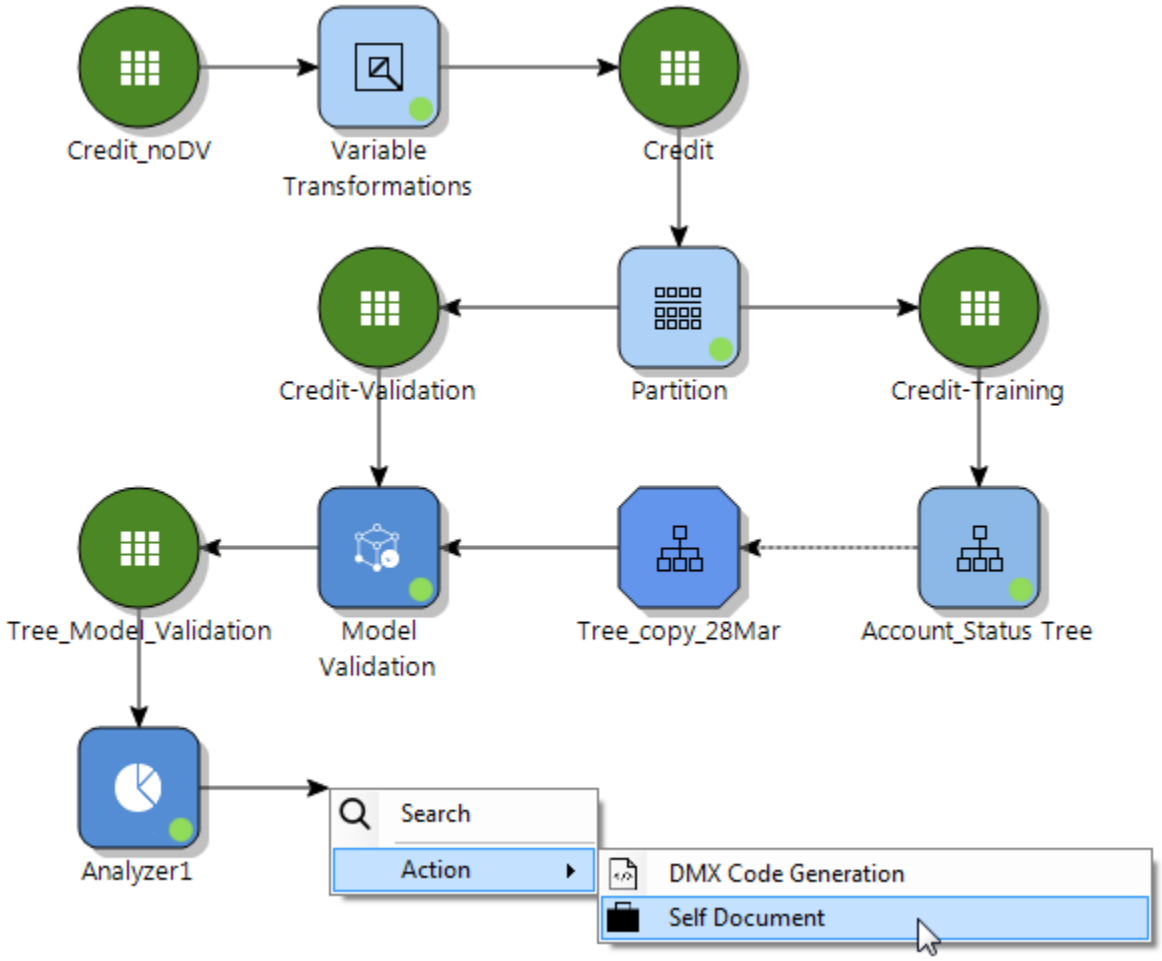
Select the **Preferences** command from the **Tools** menu, go to the General tab and make sure the option "**Take Documentation Snapshots**" is enabled. Note that this option affects the application performance, so it is recommended that you disable it when it is not needed.



- From the final node of the workflow path that needs to be auto-documented, drag a connection arrow to any area on the canvas background. Releasing the mouse button brings up a context menu. Select **Action --> Self Documentation** as in the example screenshot below. A new **Self Documentation** process node will be created.

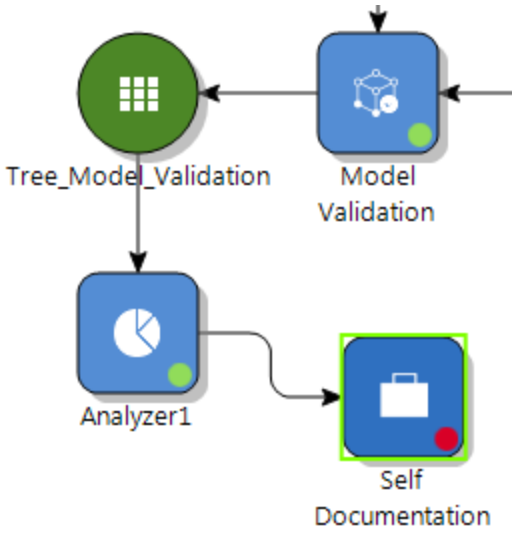
OR:

- Drag the **Self Documentation** node from the **Action** palette onto the canvas. From the final node of the workflow path that needs to be auto-documented, drag a connection arrow to the **Self Documentation** node.

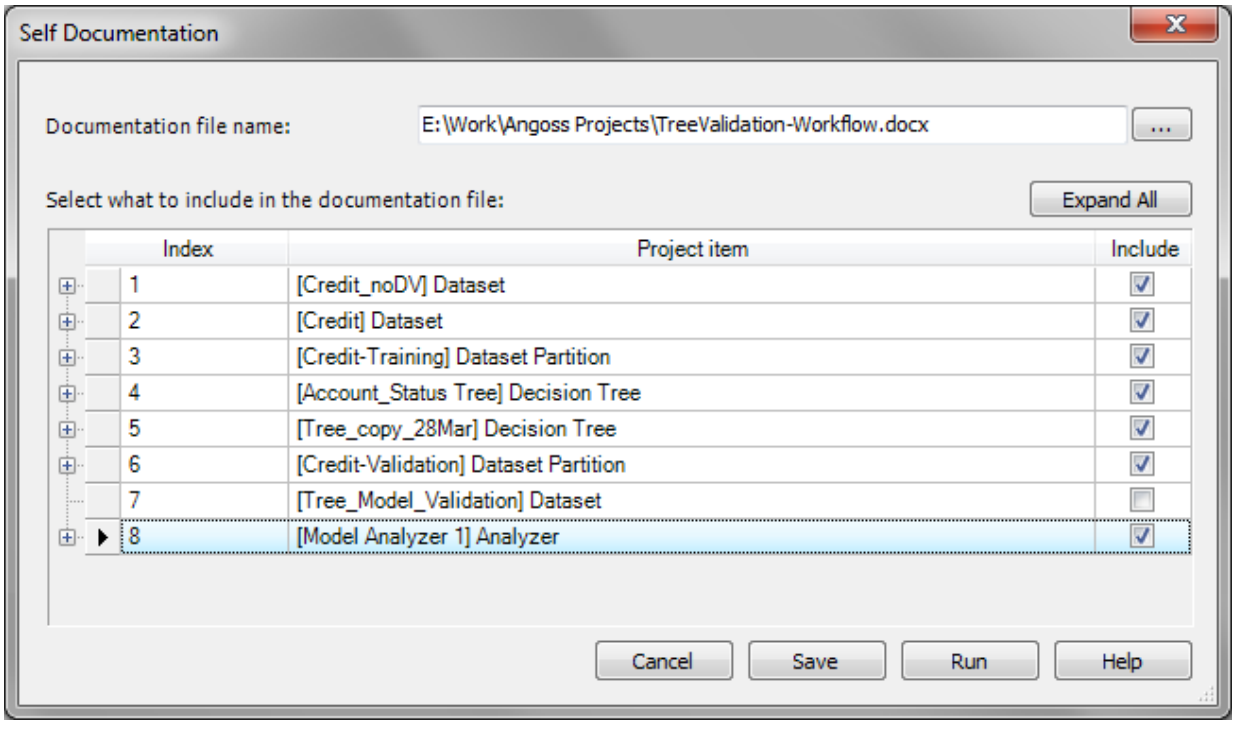


Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If the wizard does not open automatically, double-click the **Self Documentation** node or right-click on it and select **Modify** to open the wizard.

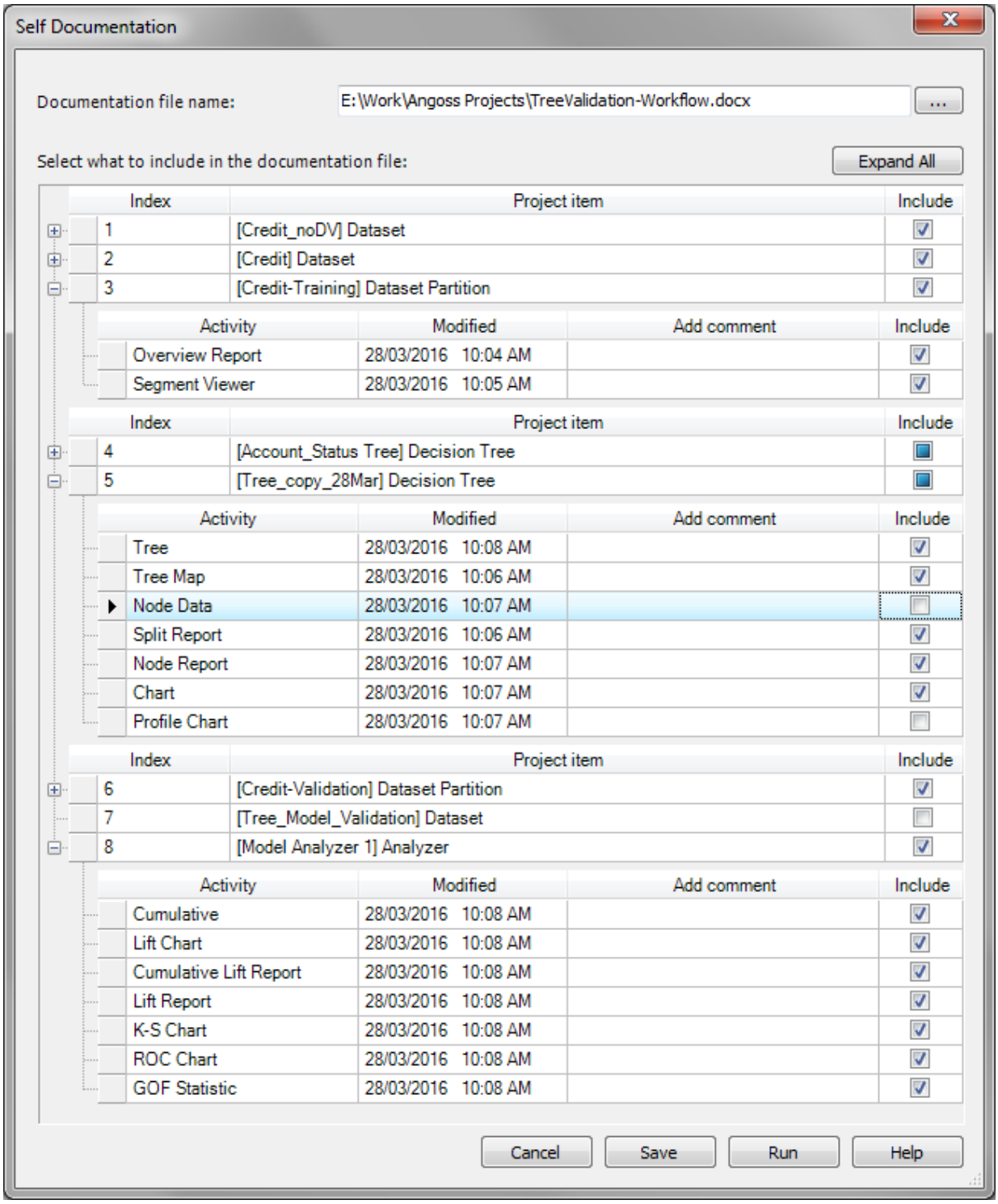


In the wizard, specify the name for the Microsoft Word document that will contain the report and select the views whose snapshots need to be included in the report. Only the images for the views that have been initialized or modified in the workflow can be included. If you are unable to Include an object in the list, this means it has not been viewed or no statistics have been calculated in it.



To select particular views to include under each object, either click on the plus sign next to the required object or click the **Expand All** button to show all available views for all objects.

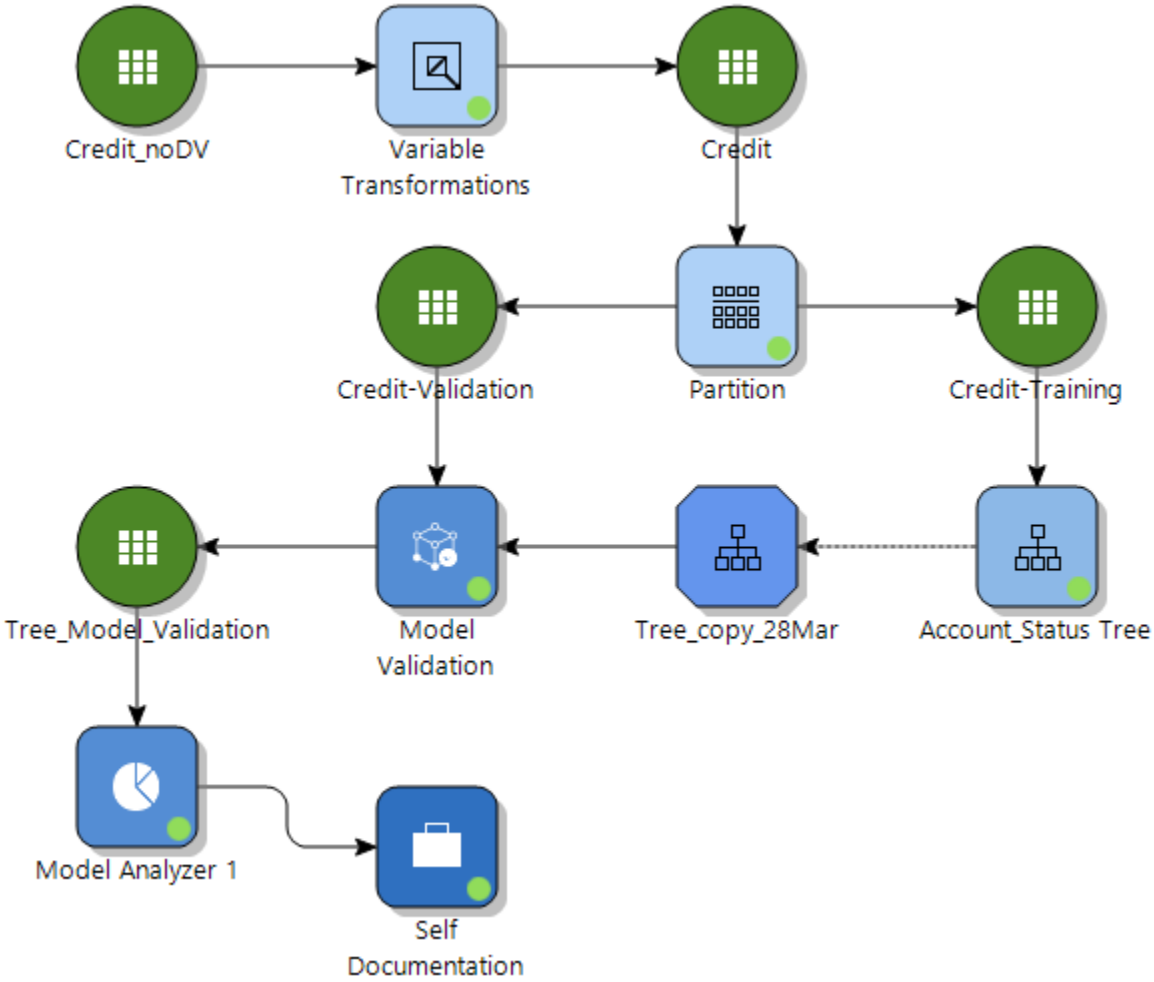
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



When finished making your selection, click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.

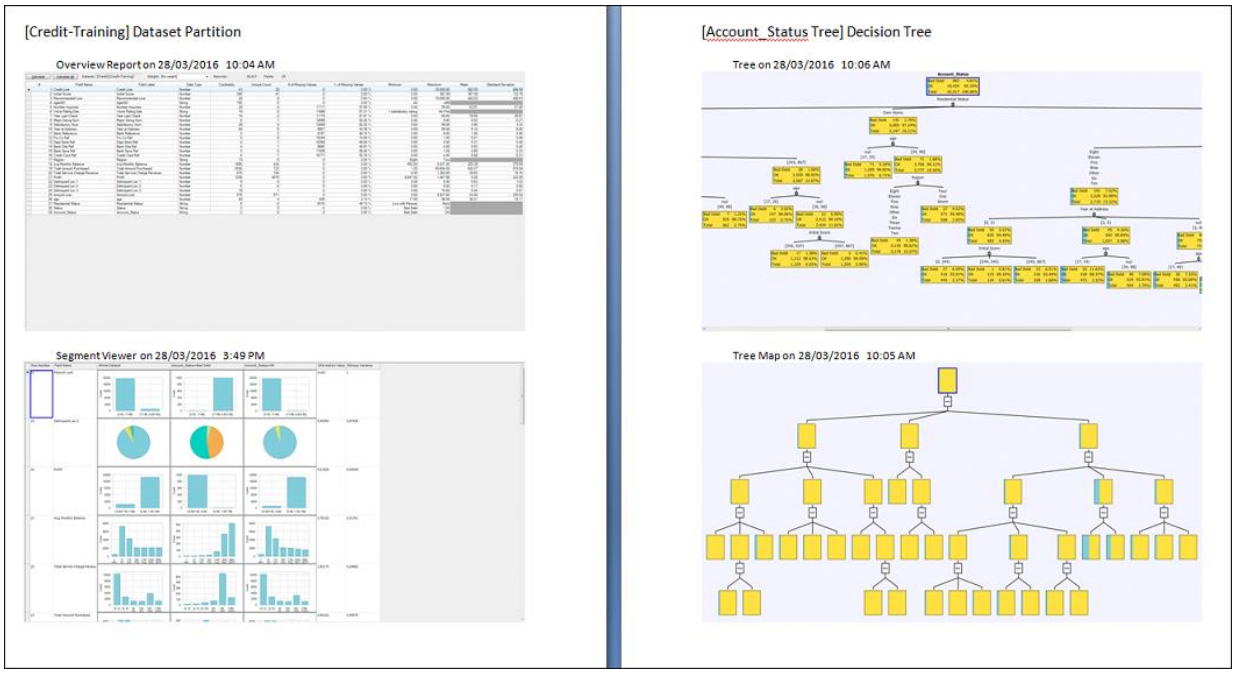
Double-click the Self Documentation node after running it to view the resulting report.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



This will open the Microsoft Word document that contains the report. Here are some example pages from a report produced by a Self Documentation node:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If necessary, edit or save it as PDF or in another format using Microsoft Word tools.

See also: Model Documentation

Model Documentation

The **Model Documentation** node from the **Action** palette provides a report of all actions you have taken when developing and viewing a model. The result of its execution is a Microsoft Word document containing snapshots of all model views you have used (opened and/or modified). For example, for a Decision Tree it may include the views such as Tree, Spit Report, Node Report, etc.

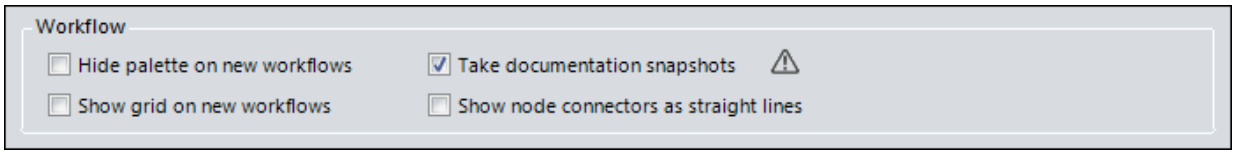
Before running the **Model Documentation** node, you can review the saved list of all views in the model path and select the ones to be included in the final report (Word document). The charts and tables you worked with in the selected views will be included in the document. The images for the views that have never been visited or initialized in the model are not included.

Note: The *Run to Here* command on the Model Documentation node context menu is never enabled. The node can only be run separately. If you run the workflow leading to a Model Documentation node that has been run previously, any existing snapshots in the report will be erased.

To add a Model Documentation operation to the workflow:

Select the **Preferences** command from the **Tools** menu, go to the General tab and make sure the option "**Take Documentation Snapshots**" is enabled. Note that this option affects the application performance, so it is recommended that you disable it when it is not needed.

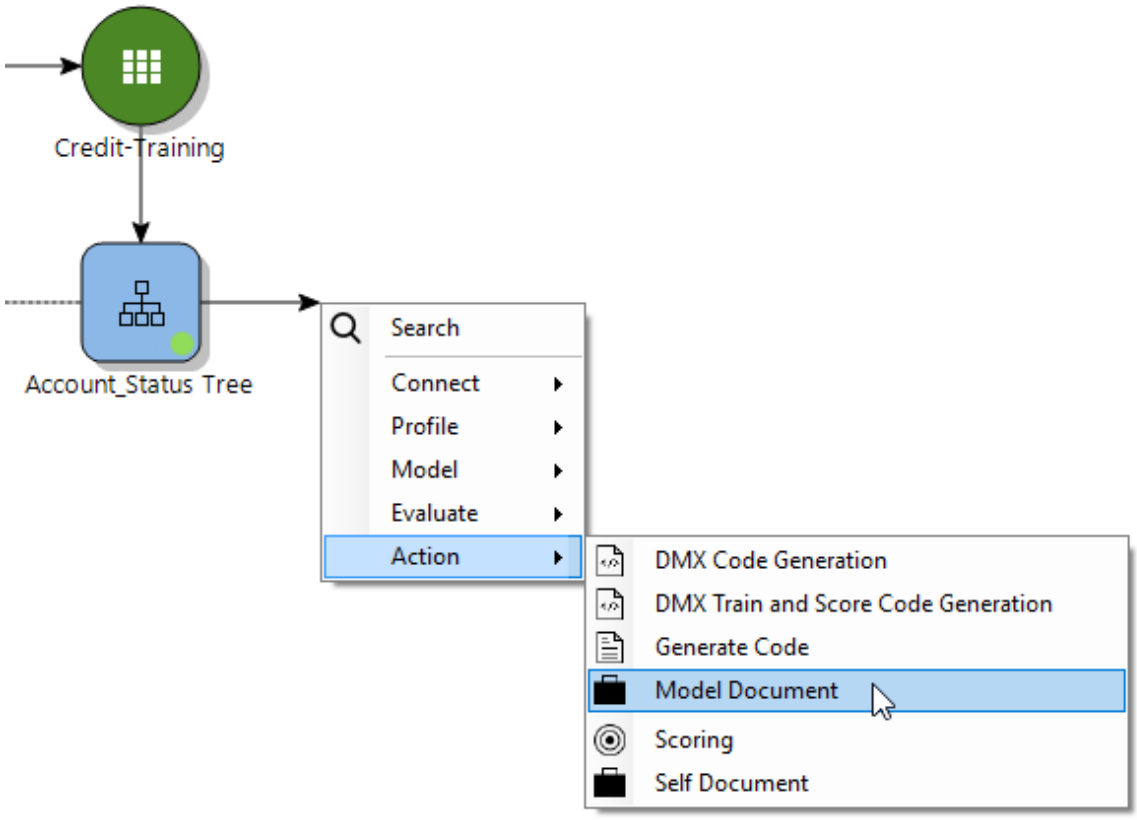
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



- Drag a connection arrow from the model node that needs to be auto-documented to any area on the canvas background. Releasing the mouse button brings up the context menu. Select **Action --> Model Documentation** as in the example screenshot below. A new **Model Documentation** process node will be created.

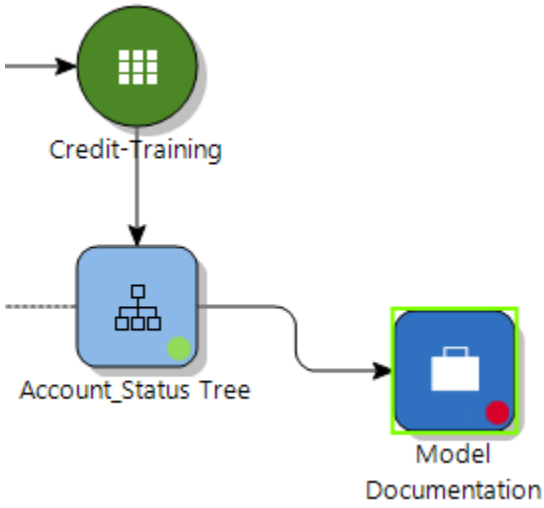
  OR:

- Drag the **Model Documentation** node from the **Action** palette onto the canvas. Connect the desired model node to the **Model Documentation** node.



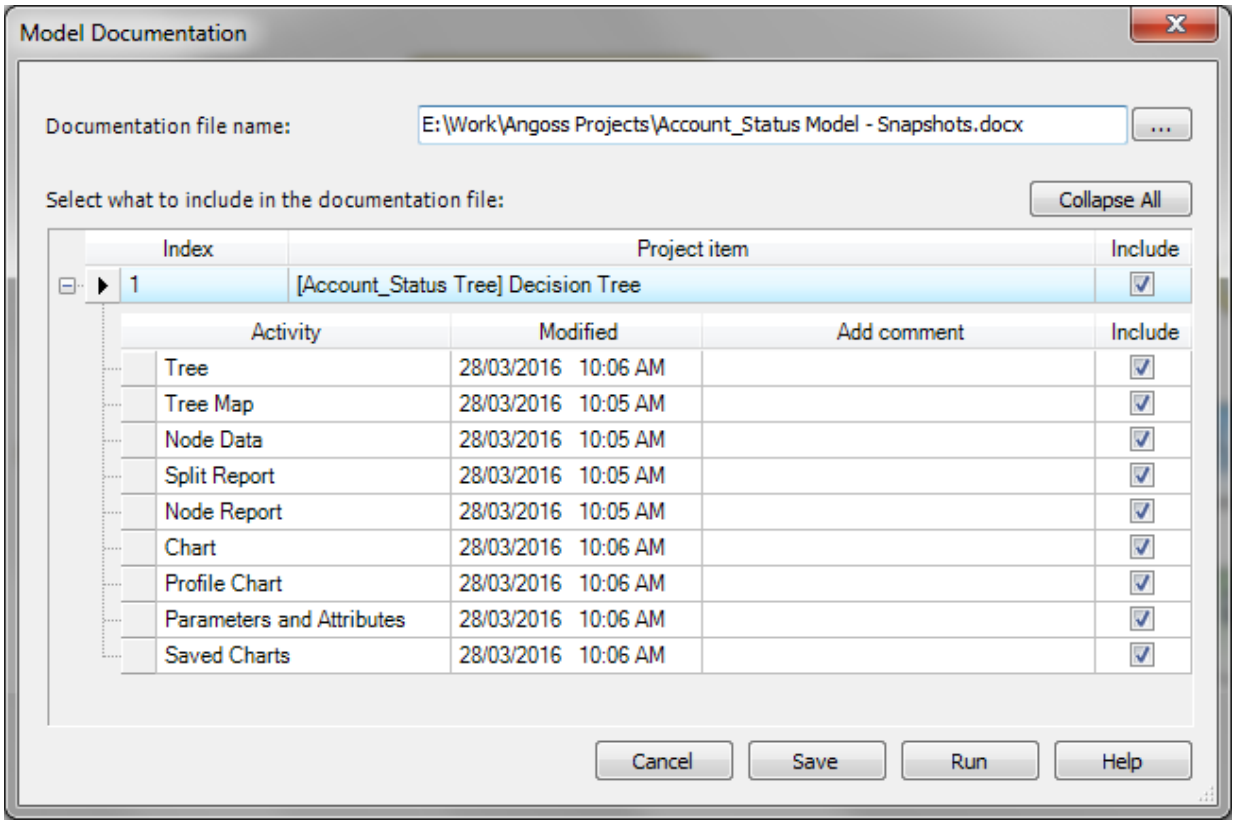
If the Model Documentation node is already on the canvas, right-click on it and select **Modify** to open the wizard.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



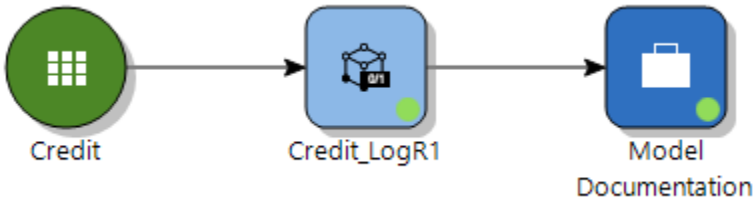
In the wizard, specify the name for the Microsoft Word document that will contain the report and select the views whose snapshots need to be included in the report. Only the images for the views that have been initialized or modified in the model object can be included. If you are unable to Include a view in the list, this means it has not been viewed or calculated.

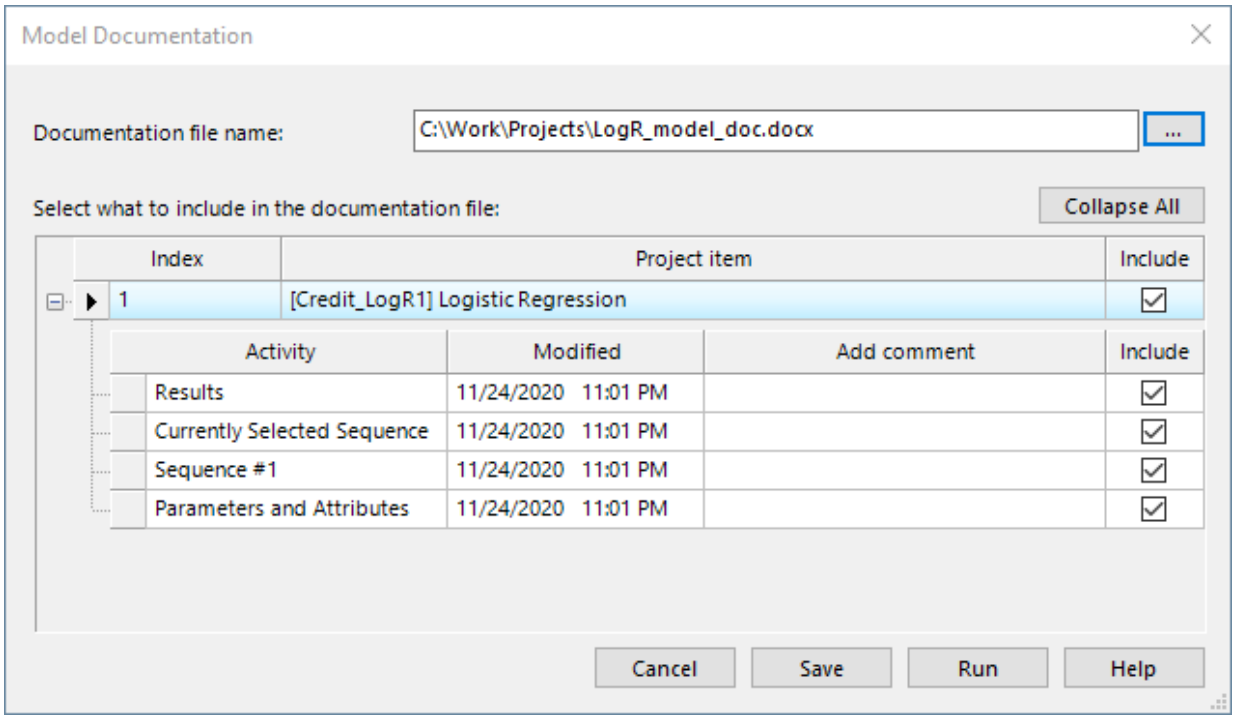
Example 1: **Model Documentation for a Decision Tree model**. In this example, all views of the Account_Status Tree have been viewed, and all of them are included in the document.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Example 2: **Model Documentation for a Logistic Regression model**. In this example, all views of the logistic regression model Credit_LogR1 have been viewed, including the detailed *Sequence statistics* pages in its Results tab, and all of them are included in the document.

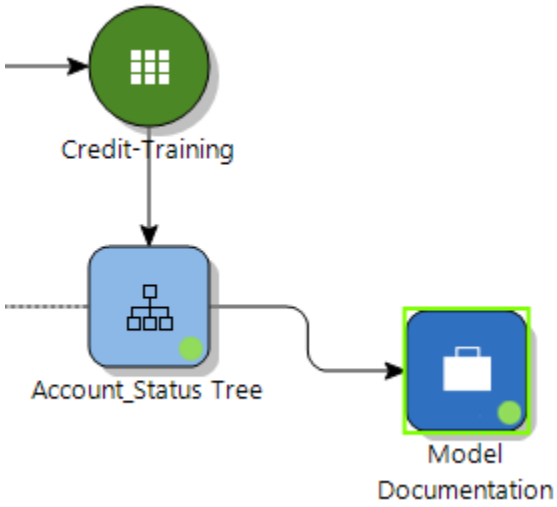




When finished making your selection, click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.

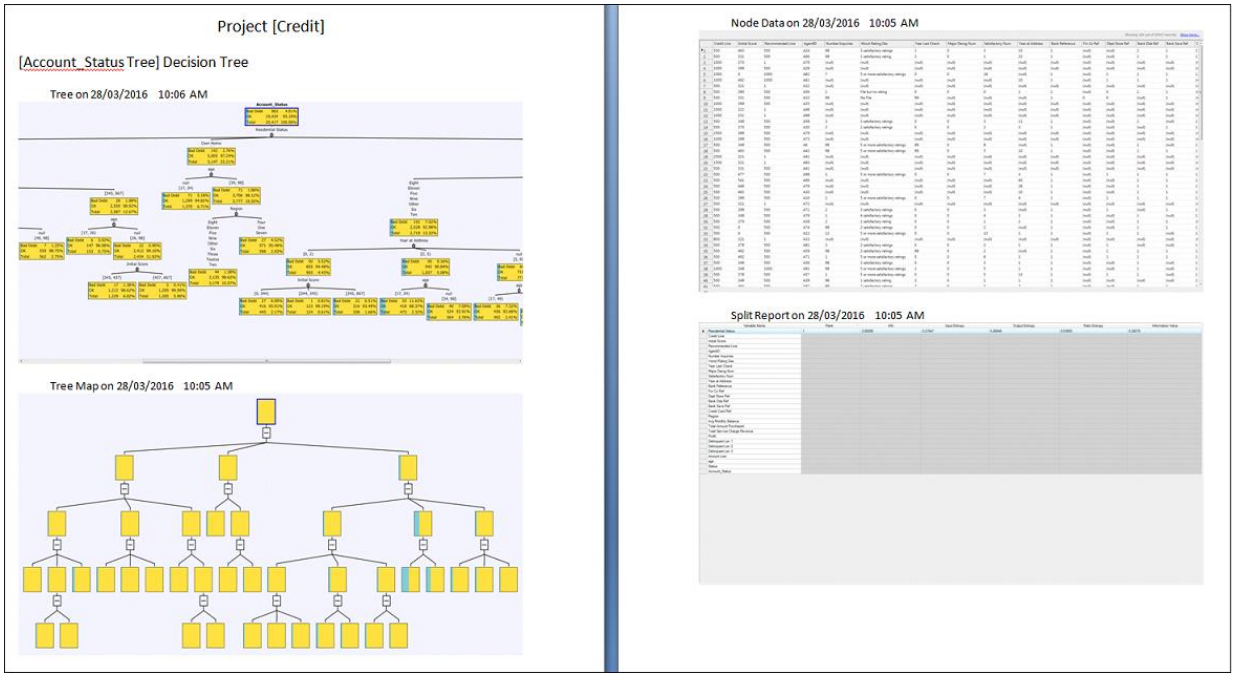
Double-click the **Model Documentation** node after running it to view the resulting report.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



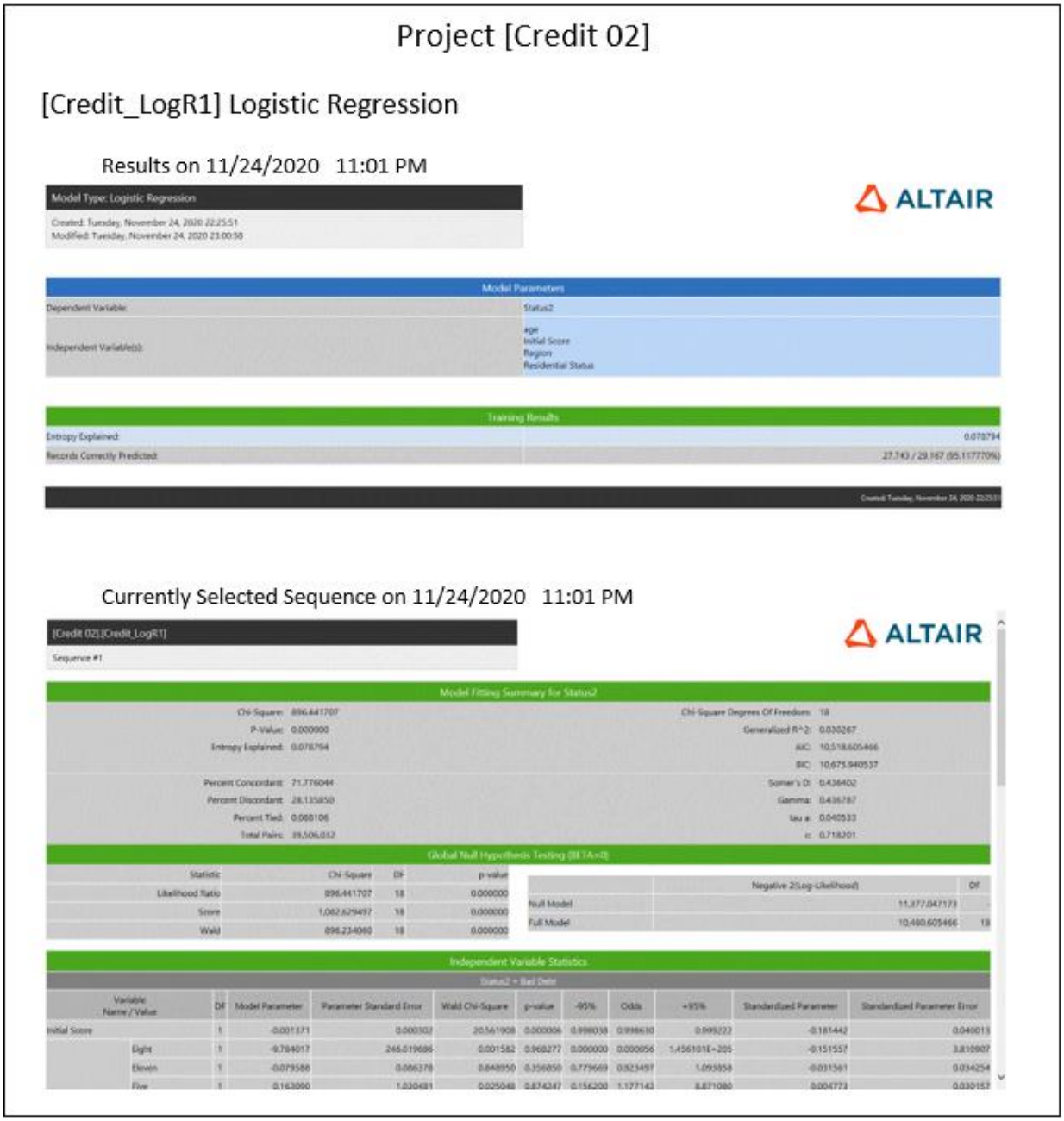
This will open the Microsoft Word document that contains the report. All images in the document are resizeable. If the font in the screenshots is too small to read, simply resize the images to enlarge them until the font is legible.

Here are some example pages from a report produced by a Model Documentation node for a Decision Tree:



An example page from a report produced by a Model Documentation node for a Logistic Regression model:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



If necessary, edit or save it as PDF or in another format using Microsoft Word tools.

See also: Self Documentation

Activity Log and Code Logs

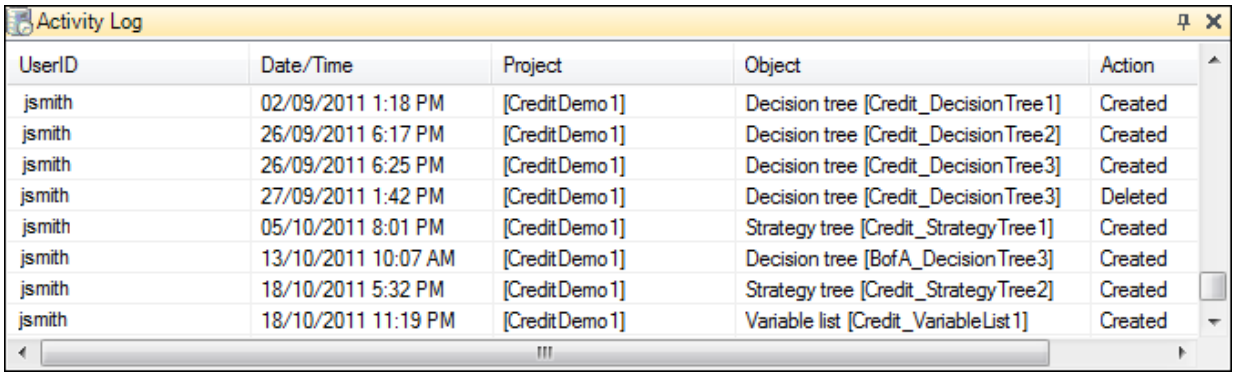
Activity Log is a history of your activities which is stored with your projects. It provides a permanent record of major events in the current project (such as creating trees, predictive models, importing

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

data, scoring or validation). It is a convenient way of tracking the changes you made in the project over time.

To view the Activity Log, select **View | Activity Log** from the application menu.

The Activity Log grid opens in the lower part of the program window. For any event, it displays the author's user ID, date and time, name and type of object created, and type of action.



UserID	Date/Time	Project	Object	Action
jsmith	02/09/2011 1:18 PM	[CreditDemo1]	Decision tree [Credit_DecisionTree1]	Created
jsmith	26/09/2011 6:17 PM	[CreditDemo1]	Decision tree [Credit_DecisionTree2]	Created
jsmith	26/09/2011 6:25 PM	[CreditDemo1]	Decision tree [Credit_DecisionTree3]	Created
jsmith	27/09/2011 1:42 PM	[CreditDemo1]	Decision tree [Credit_DecisionTree3]	Deleted
jsmith	05/10/2011 8:01 PM	[CreditDemo1]	Strategy tree [Credit_StrategyTree1]	Created
jsmith	13/10/2011 10:07 AM	[CreditDemo1]	Decision tree [BofA_DecisionTree3]	Created
jsmith	18/10/2011 5:32 PM	[CreditDemo1]	Strategy tree [Credit_StrategyTree2]	Created
jsmith	18/10/2011 11:19 PM	[CreditDemo1]	Variable list [Credit_VariableList1]	Created

Activity Log Window Properties

The Activity Log window is dockable. You can detach it and relocate within the application window by clicking its title bar and dragging to the desired place. To close the Activity Log, click the 'x' button in its top right corner or select **View | Activity Log** again.

You can make the Activity Log auto-hide when it is not active by right-clicking on its title bar and selecting Auto Hide from the context menu.

Query Log

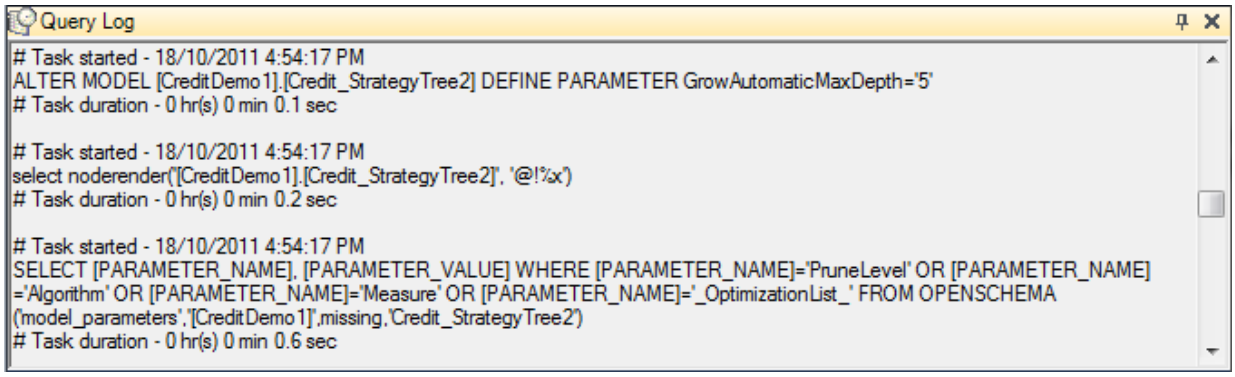
Query Log is a tool for advanced users that provides a record of your actions represented in the form of SQL-like commands sent to the Altair Data Mining Engine from the user interface when your data mining tasks are being performed.

The commands displayed in the Query Log are expressed in the ***Data Mining Extensions query language (DMX)***, which is an extension of SQL for Data Mining.

To open the Query Log, select **View | Query Log** from the menu.

In the following example, the second command in the Query Log represents the Automatic Grow operation performed on the tree Credit_P1_Decision Tree1 based on the CreditRisk dataset.

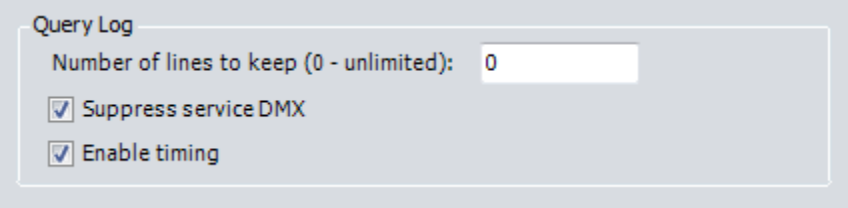
Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Query Log Display Options

You can choose the level of detail represented in the Query Log, specify how many recent commands to keep in the log, and choose to display the starting time for each command and the time it takes to execute.

To define the display options for the commands in Query Log, select **Tools | Preferences** from the menu and select the ***General*** tab.



Number of lines to keep	Specifies the number of the most recent DMX commands to preserve in the Query Log (the default is 0, which stands for "unlimited", i.e., all commands processed during the current session will be kept in the log).
Suppress service DMX	Suppresses the auxiliary DMX commands that are not directly related to the data mining tasks being performed (for example, commands that update the project hierarchy, or commands that create temporary objects).
Enable timing	Displays the starting time and duration of each DMX command.

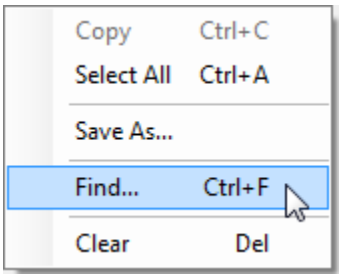
Query Log Window Properties

The Query Log window is dockable. You can detach it and relocate within the application window by clicking its title bar and dragging to the desired place. To close the Query Log, click the 'x' button in its top right corner or select **View | Query Log** again.

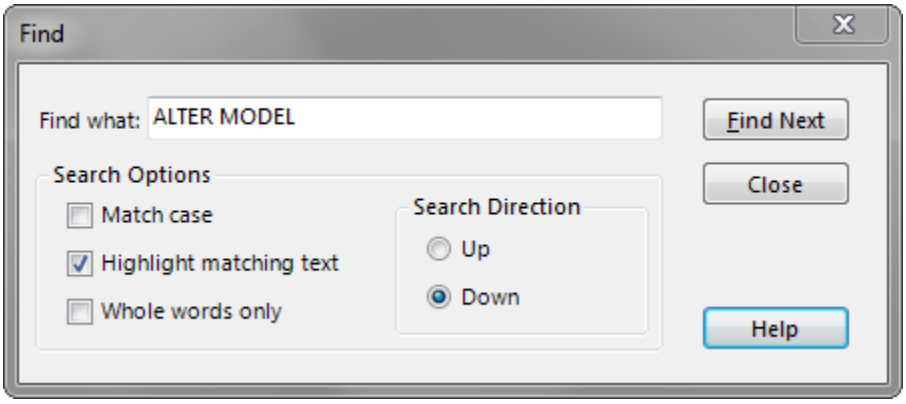
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

You can make the Query Log auto-hide when it is not active by right-clicking on its title bar and selecting Auto Hide from the context menu.

To find a particular entry in the Query Log, press CTRL F or right-click in the Query Log frame and select **Find**.



This brings up the Find dialog, where you can perform advanced search for a string.



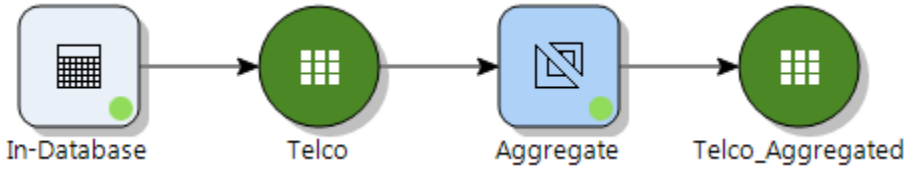
ODBC/Code Log

ODBC/Code Log is a tool for advanced users that provides a record of your actions when you run In-Database Analytics tasks.

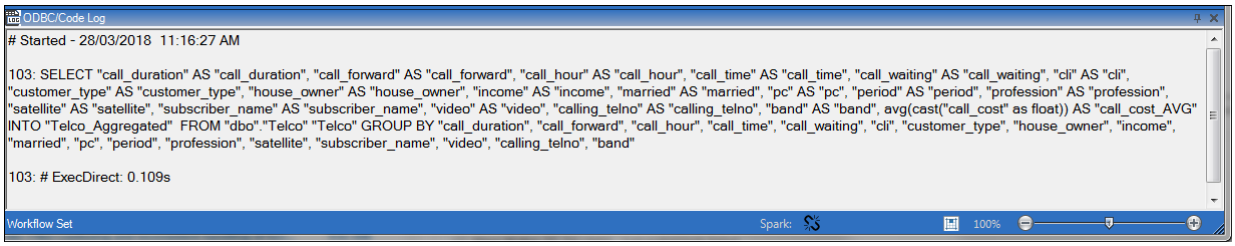
ODBC Log displays the commands translated from the DMX code of the Altair Data Mining Engine into the SQL dialect of the target database.

To open the ODBC Log, select **View | ODBC/Code Log** from the menu.

In the following example, the second command in the log represents the aggregation query that was run in the database.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

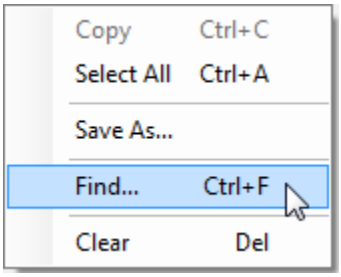


ODBC/Code Log Window Properties

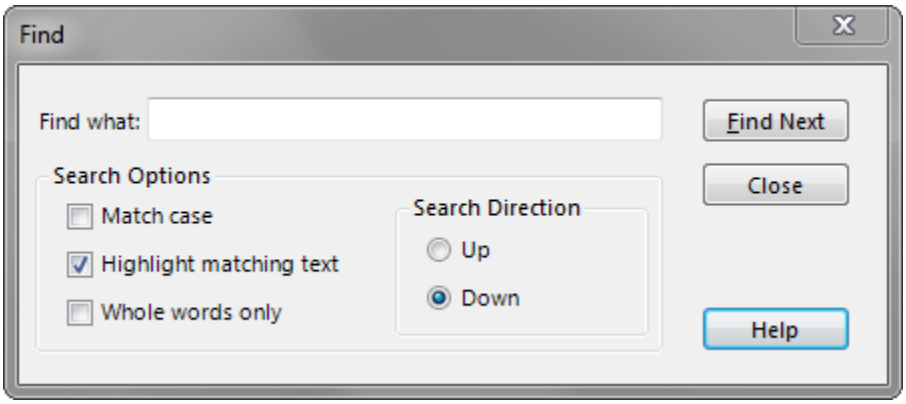
The ODBC/Code Log window is dockable. You can detach it and relocate within the application window by clicking its title bar and dragging to the desired place. To close the ODBC/Code Log, click the 'x' button in its top right corner or select **View | ODBC/Code Log** again.

You can make the ODBC/Code Log frame auto-hide when it is not active by right-clicking on its title bar and selecting Auto Hide from the context menu.

To find a particular entry in the log, press CTRL F or right-click in the ODBC/Code Log frame and select **Find**.



This brings up the Find dialog, where you can perform advanced search for a string.



KSQuery Command Line Interface

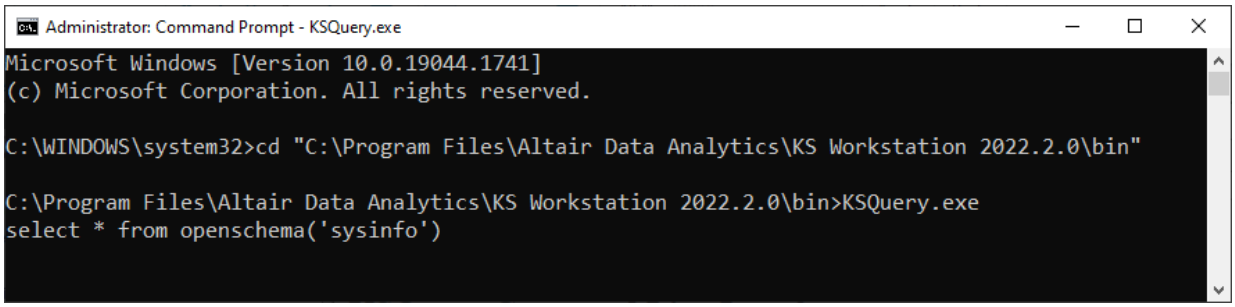
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

ksquery is a command line executable that allows to you issue data mining commands in the DMX language (SQL with Data Mining Extensions) directly in the Altair Data Mining Engine from the Windows command line or Linux/Unix shell.

On Windows, KSQuery.exe is located in the *bin* subdirectory of the application program folder (for example, the default location in version 2022.3 is C:\Program Files\Altair Data Analytics\KS Workstation 2022.3.0\bin). On Linux, ksquery is located in the *lib* subdirectory of the KS Server home directory (the default is /opt/Altair-Data-Analytics/KS-Server-2022.3/lib).

Ksquery allows you to test the Data Mining Engine without the use of the graphical user interface to diagnose technical issues, get more detailed, or even automate some tasks that for any reason you cannot or would not want to automate via the graphical user interface of the application. Use the command `ksquery -help` for syntax help.

To start ksquery on Windows, run the Command Prompt, change to the *bin* subdirectory of the Altair Data Analytics program folder, type *ksquery,* and press Enter. The ksquery command prompt opens. The ksquery prompt is an empty line with a cursor. To exit the ksquery prompt, type `:exit` (colon followed by "exit"). Alternatively, press CTRL C on the keyboard, or close the Command Prompt window.



Some examples:

To get the information on the current licenses, type

```
select * from openschema ('licenses')
```

To get the detailed system information, type

```
select * from openschema ('sysinfo')
```

You can specify statements directly after the ksquery command using the switch *--exec* and enclosing your statement in double quotes. For example, the following command lists all objects in the project Project_CreditScoring:

```
ksquery --exec "SELECT * FROM [Project_CreditScoring]"
```

You can also copy a sequence of commands in a query file and execute all commands in the file as follows:

```
ksquery --read queries.txt
```

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Using ksquery, you can execute any statements recorded in the Query Log. Copy any command from the Query Log and paste it into the ksquery command prompt. They will be executed, provided that the Working Directory is set properly. To set the working directory in ksquery, use the --init switch with the Mining Location option:

```
ksquery --init "Mining Location=<path to the working directory>"
```

This example shows how to explicitly specify the Working Directory and the license file name within the **--init** argument:

```
ksquery --init "Mining Location=C:\Projects;BaseLicenseFileName=ADB3F91C.lic" --read query_file.sql
```

To get syntax help for KSQuery, use the command

```
KSQuery.exe --help
```

General Usage:

```
ksquery [--xml] [--provider ppp] [--init iii]
                [--read fff] [--exec qqq] [--echo] [--help]
                [{fff|qqq} ...]
```

`--xml` forces output to be an XML document

`--dme` enables DME protocol

`ppp` the name of the provider (the default is KDSProvider64-2022.3.dll on Windows and KDSProvider2022.3.0.so on Linux)

`iii` the connect string for the provider. The connect string must be enclosed in double quotes and is comprised of several parts separated by a semicolon.

Example connect string: `"Mining Location=[.];LicenseAutoLock=[true/false]"`

`fff` the name of the file containing the queries to execute

`qqq` the query to execute

`--echo` turns on query echo

`--help` produces the syntax help

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Example use of KSQuery with connect string:

```
ksquery --init "Mining Location=[.];LicenseAutoLock=[true/false]" --read [myfile]
```


KSQuery and License Location

KSQuery looks for the license location specified in the file `RegistrationKeys.xml`, which is in the same folder as KSQuery. KSQuery reads the path specified in the line `<License directory="XXXXXXX"></License>`, then loads the license(s) found there.

The file `RegistrationKeys.xml` is created and updated only once - when KS Workstation is installed. If you change the license location via the License Manager dialog, this does not affect RegistrationKeys.xml. If the license is then moved to the new location, KSQuery will be unaware of this unless you manually update the <License directory> entry in `RegistrationKeys.xml`. Also, make sure the license directory specified there has Read permissions for the user who runs KSQuery.

For more details on the DMX and the ksquery syntax, please contact Technical Support.

Version Compatibility

If more than one version of Knowledge Studio or Knowledge Seeker are installed side by side on the same Windows machine, running KSQuery.exe of the older version may return an error: "*The requested operation is not licensed*". In that case the name of the license file for the older version has to be explicitly specified in the KSQuery command line within the **--init** argument. Here is an example:

```
ksquery.exe --init "Mining Location=C:\Projects;BaseLicenseFileName=license_file1.lic" --read query_file.sql
```

To find out the name of your older license file, select License Manager from the Help menu and scroll to the right in the License Manager window until you see the FILE column.

DMX Code Generation

DMX is the query language of the Altair Data Mining Engine. More precisely: **SQL DMX** (***SQL with Data Mining Extensions***) - an extension of the SQL query language for Data Mining. Commands in this language are sent to the Altair Data Mining Engine from the user interface when your data mining tasks are being performed. You can view this commands in the **Query Log**, available under the View menu.

The **DMX Code Generation** node from the **Action** palette generates DMX code for operations in your workflow. The result of its execution is a file that contains the Altair Data Mining Engine queries used for all operations in the workflow up to the point of the node connected to the DMX Code

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Generation node. The **DMX Code Generation** node can be attached not only at the end of a workflow, but also anywhere in the middle of it. The workflow path being automatically scripted is the path leading up to the Self Documentation node.

This may be useful for advanced users if they need to run the command line tool (**ksquery**) for batch-mode processing. The generated DMX code file can be passed to the command-line executable **ksquery** as an input. The result would be a re-run of the scripted workflow up to the point of the code generation. Command-line scripts can be run as scheduled tasks if necessary.

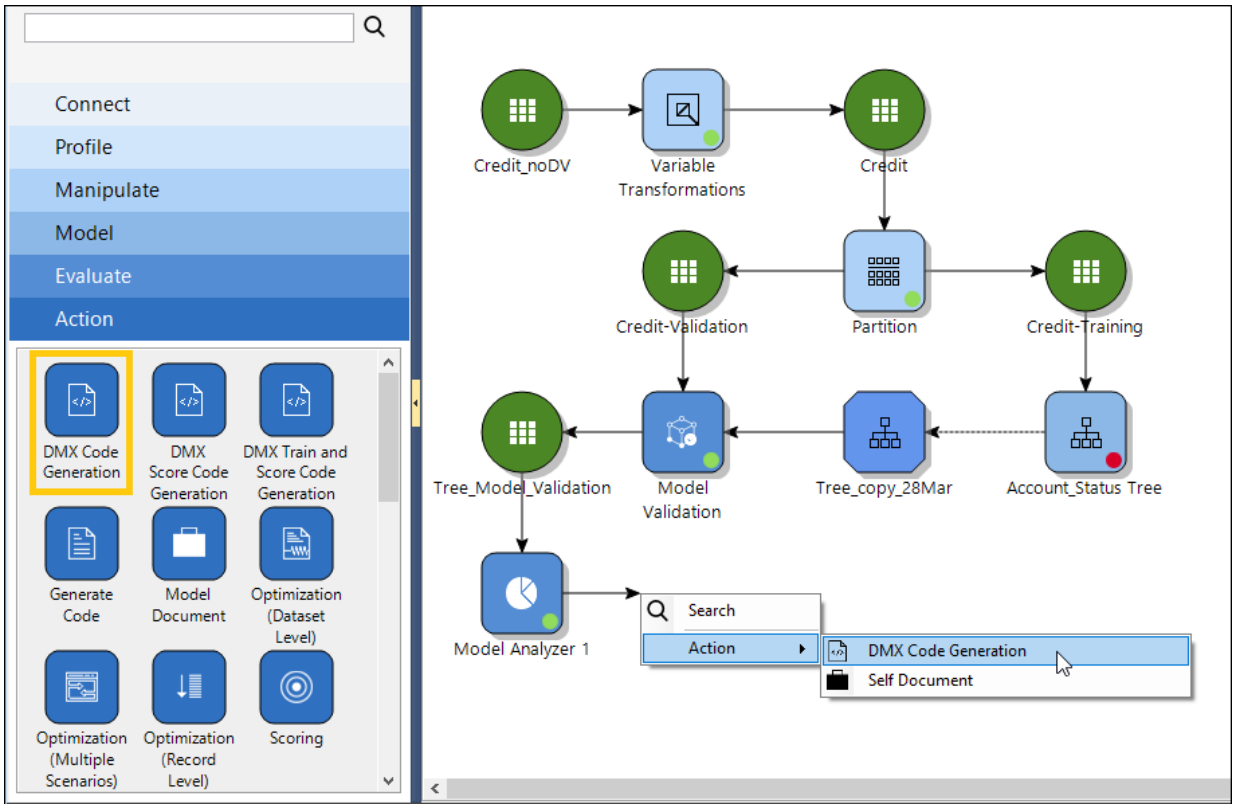
Note: Only the **Run to Here** command is available for the DMX Code Generation node. When you execute it, the whole workflow path leading to it will be run.

To add a DMX Code Generation operation to the workflow:

- From the final node of the workflow path that needs to be scripted, drag a connection arrow to any area on the canvas background. Releasing the mouse button brings up a context menu. Select **Action --> DMX Code Generation** as in the example screenshot below. A new **DMX Code Generation** node will be created.

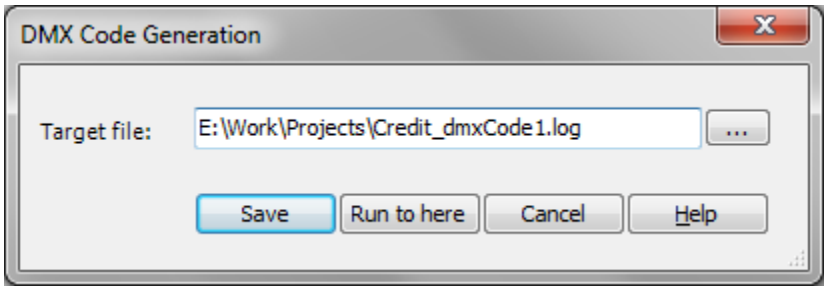
  OR:

- Drag the **DMX Code Generation** node from the **Action** palette onto the canvas. From the final node of the workflow path that needs to be scripted, drag a connection arrow to the **DMX Code Generation** node.

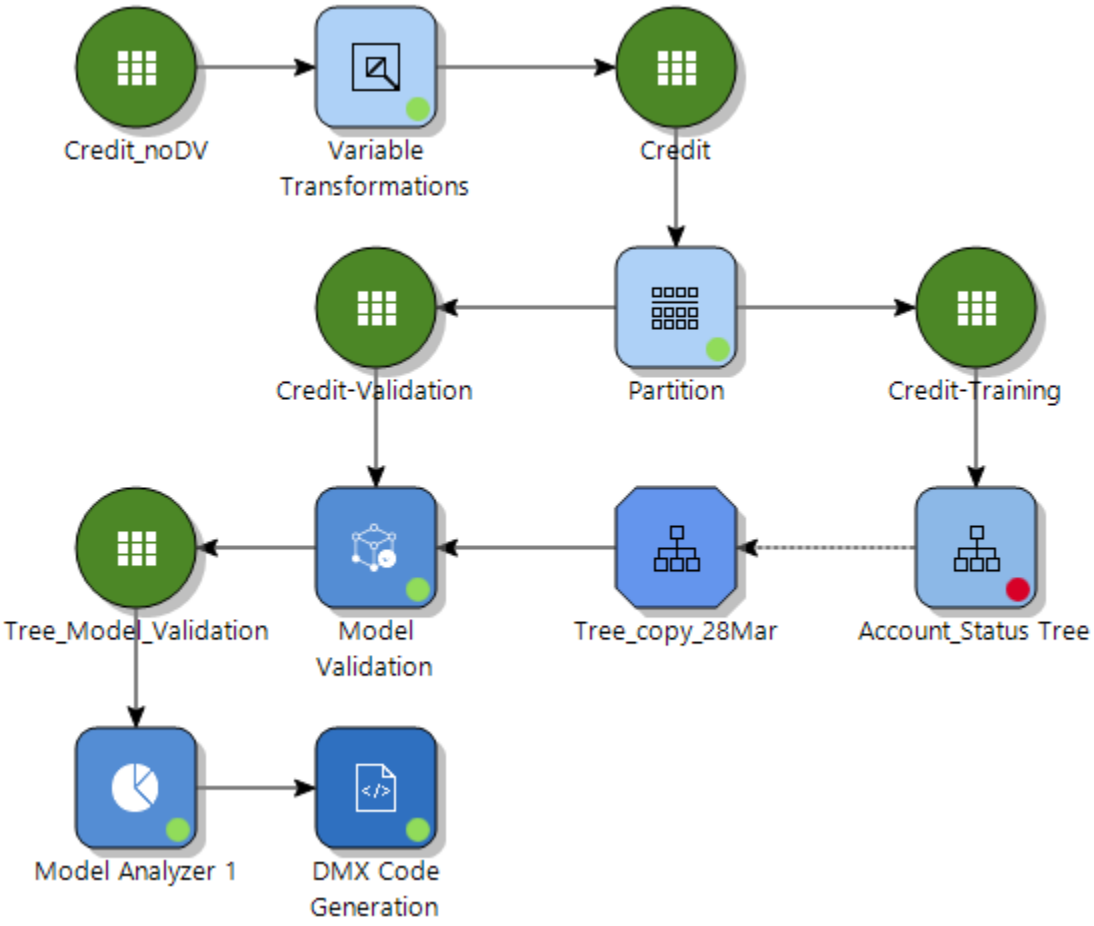


Specify the name for the file that will contain the generated DMX code. The default extension is *.log.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

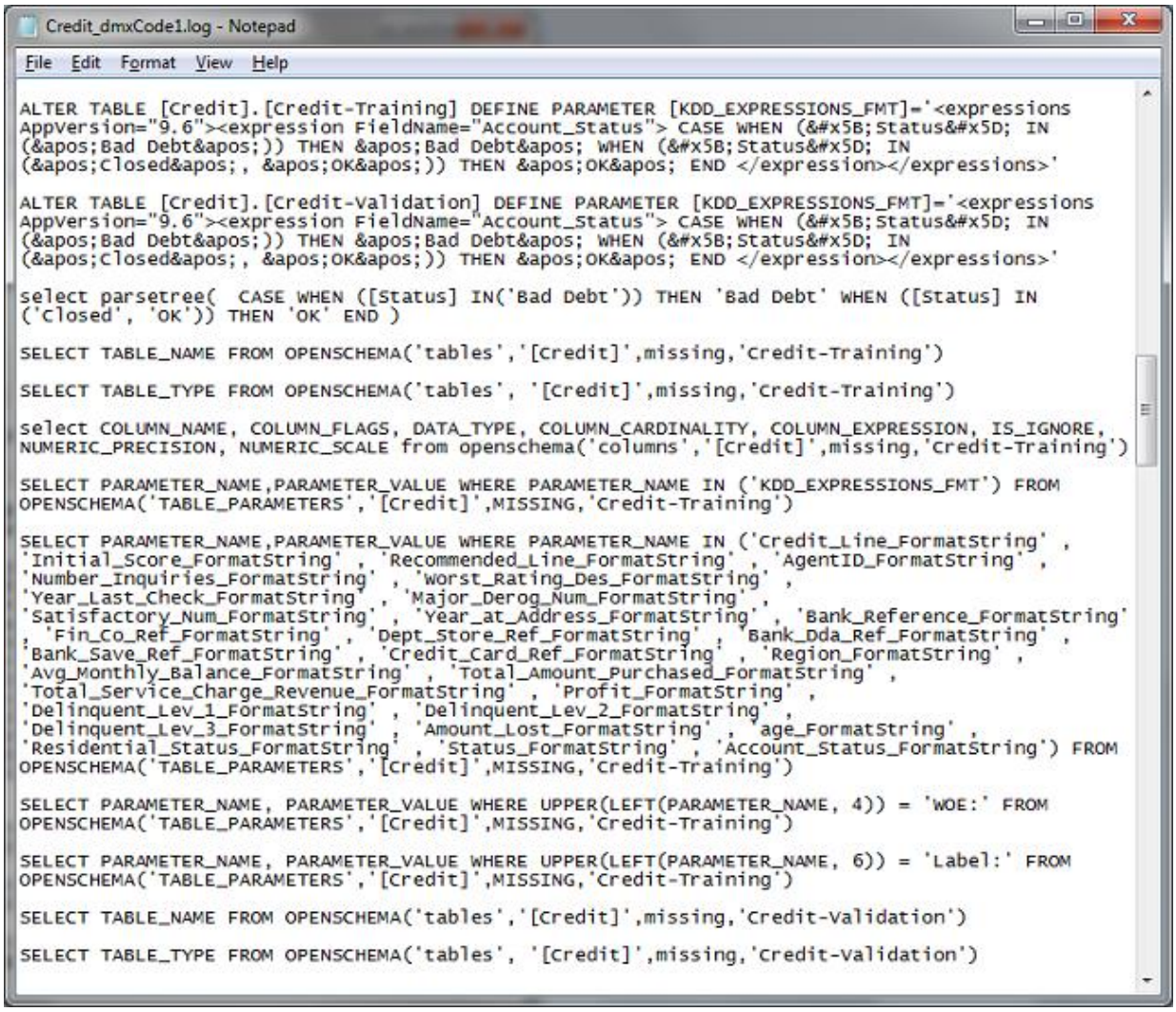


Click **Save** to save the settings, or click **Run to here** to save the settings and run the process immediately.



Double-click the DMX Code Generation node after running it to view the resulting report. It opens in your default editor for log files.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



For instructions on running command line tools with DMX query files, see the topic Command Line Tools.

See also:

- DMX Score Code Generation
- DMX Train & Score Code Generation
- Query Log
- Command Line Tools

The **DMX Score Code Generation** node in the **Action** palette applies the function of **DMX Code Generation** specifically for scripting **Scoring** operations. In the workflow, it can only be connected to a Scoring node. The result of its execution is a file that contains the DMX code for the Scoring node it is attached to.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

For general information on the Altair Data Mining Engine query language and scripting with the use of operations in the **Action** palette, see the topics Command Line Tools and **DMX Code Generation**.

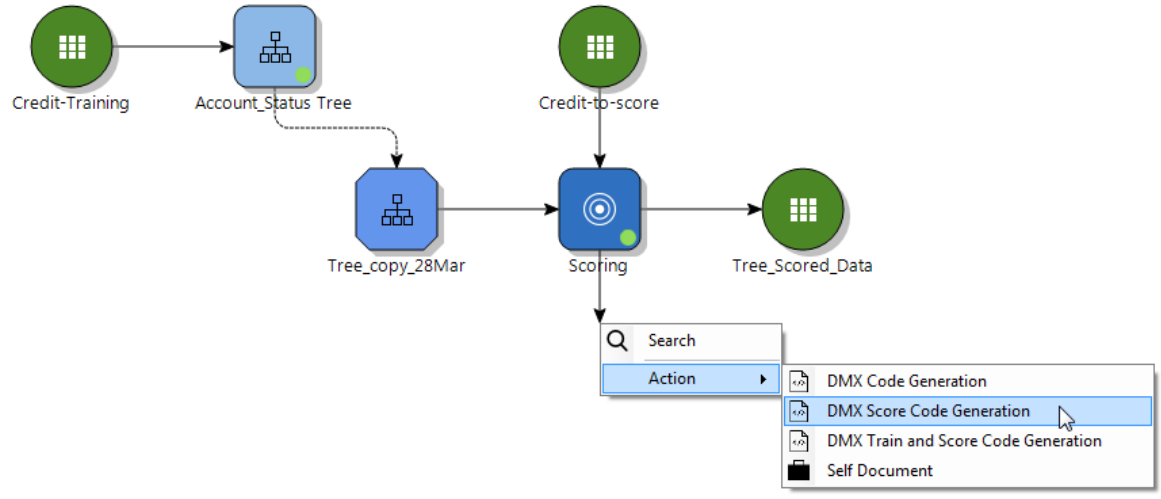
DMX Score Code Generation may be useful for advanced users interested in running batch scoring in the command-line mode. The generated DMX code file can be passed to the command line executable **ksquery** as an input. Command-line scripts can be run as scheduled tasks if necessary.

To add a DMX Score Code Generation operation to the workflow:

- From the Scoring node that needs to be scripted, drag a connection arrow to any area on the canvas background. Releasing the mouse button brings up a context menu. Select **Action --> DMX Score Code Generation** as in the example screenshot below. A new **DMX Score Code Generation** node will be created.

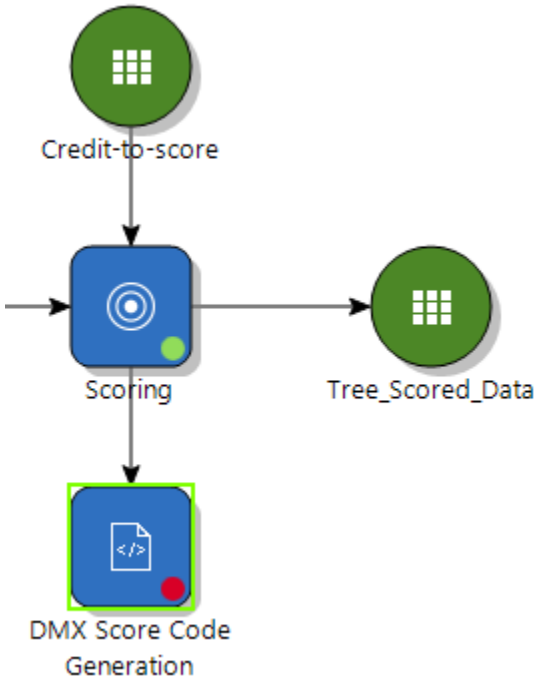
OR:

- Drag the **DMX Score Code Generation** node from the **Action** palette onto the canvas. From the Scoring node that needs to be scripted, drag a connection arrow to the **DMX Score Code Generation** node.

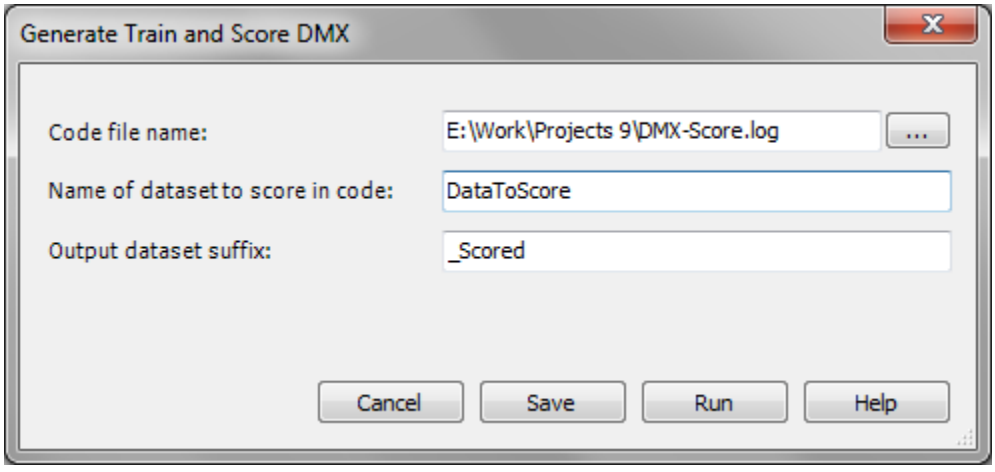


If the node wizard does not open automatically, double-click the new DMX Score Code Generation node or right-click on it and select **Modify**.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

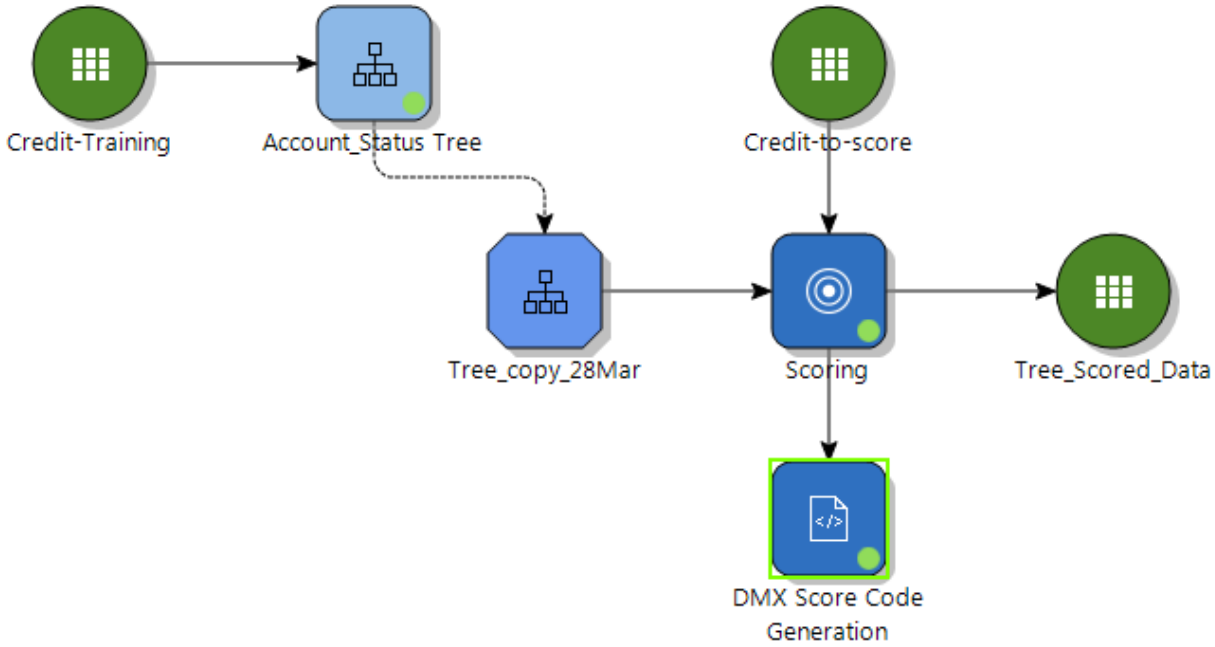


- Specify the name for the log file that will contain the generated DMX code. The default extension is *.log.
- In the code to be generated, the name of the input dataset of the Scoring node in your workflow will be replaced with the name you specify here. The default is "*DataToScore*", and the output suffix is "*_Scored*", so the code will score the dataset "*DataToScore*" and produce "*DataToScore_Scored*" unless you change the defaults.



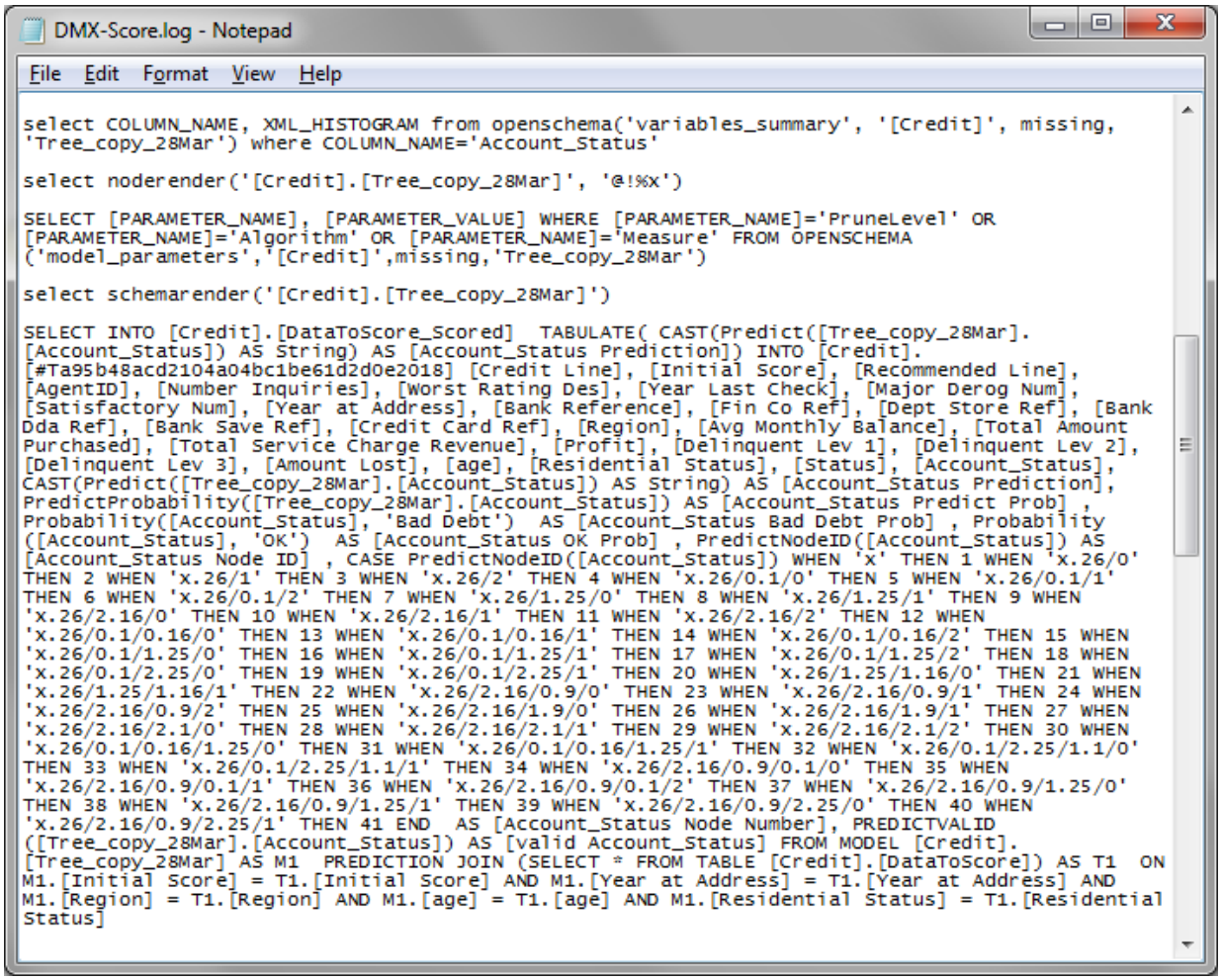
Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Double-click the DMX Score Code Generation node after running it to view the resulting report. It opens in your default editor for log files.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Note: Only the **Run** command is available for the DMX Score Code Generation node. *Run to Here* is not available on the context menu for this node.

For instructions on running command line tools with DMX query files, see the topic Command Line Tools.

See also:

- DMX Code Generation
- DMX Train & Score Code Generation
- Query Log
- Command Line Tools

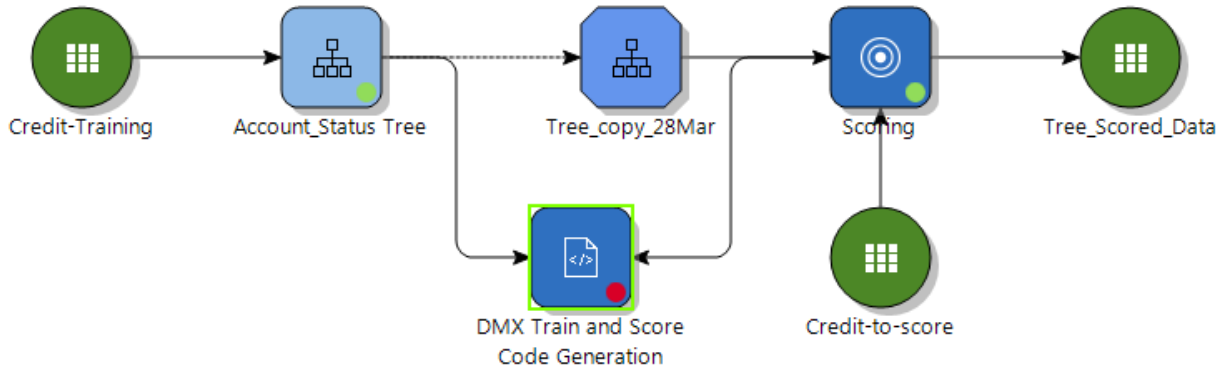
The **DMX Train & Score Code Generation** node in the **Action** palette provides the function of **DMX Code Generation** specifically for scripting the combination of **Training and Scoring** operations. When used in the workflow, it admits two inputs: a Model node and a Scoring node. The result of its execution is a file that contains the DMX code for the model training followed by its scoring.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

For general information on the Altair Data Mining Engine query language and scripting with the use of operations in the **Action** palette, see the topics Command Line Tools and **DMX Code Generation**.

DMX Train & Score Code Generation may be useful for advanced users interested in running regular model training (refresh) and scoring operations in the batch mode. The generated DMX code file can be passed to the command-line executable **ksquery** as an input. Command-line scripts can be run as scheduled tasks using Windows Task Scheduler if necessary.

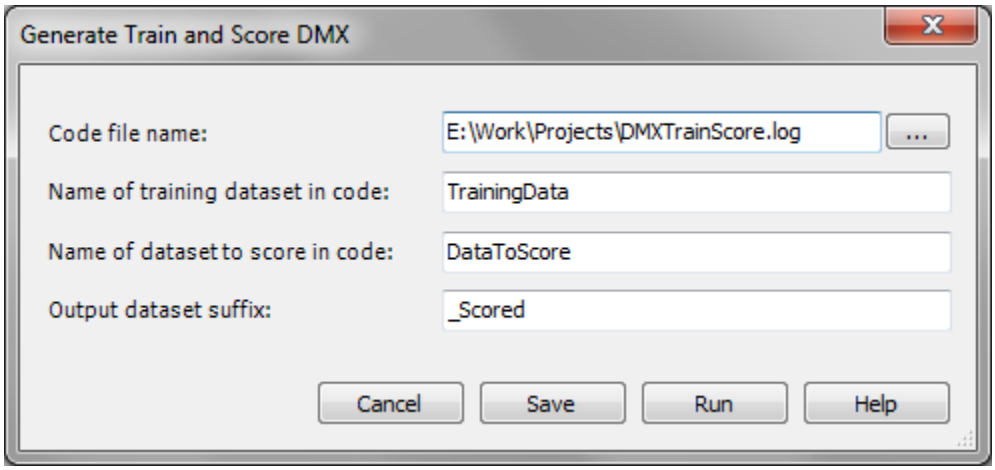
To generate DMX code for training and scoring a model in the workflow, drag the **DMX Train & Score Code Generation** node from the **Action** palette onto the canvas. Connect both the desired model node and the scoring node to the same the DMX Train & Score Code Generation node.



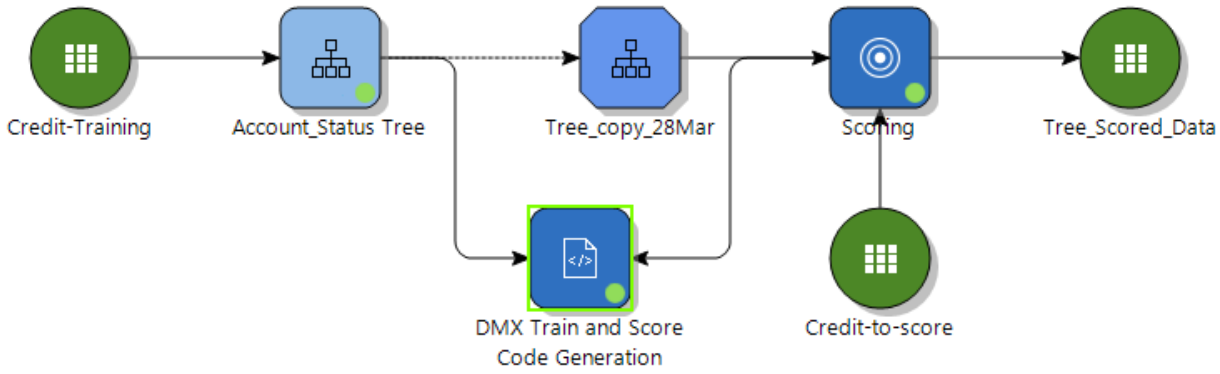
Double-click the new DMX Train & Score Code Generation node or right-click on it and select **Modify**.

- Specify the name for the log file that will contain the generated DMX code. The default extension is *.log.
- In the code to be generated, the names of the actual inputs for model training and scoring in your workflow will be replaced with the names you specify in the wizard. The default name for the training dataset in the code is "*TrainingData*", for the scoring input - "*DataToScore*", and the output name will be the latter plus the suffix "*_Scored*". The code will train the model on "*TrainingData*", score it on "*DataToScore*" and produce "*DataToScore_Scored*" unless you change the default names.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



Click **Save** to save the settings, or click **Run** to save the settings and run the process immediately.



Double-click the DMX Train & Score Code node after running it to view the resulting report. It opens in your default editor for log files.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

DMXTrainScore.log - Notepad

File Edit Format View Help

```
OPENSCHEMA('TABLE_PARAMETERS','[Credit]',MISSING,'TrainingData')

CREATE MINING MODEL [Credit].[Account_Status Tree] ([Credit Line] SIGNIFICANCE 0.05 MAX
BRANCHES 0 BREAK true DISPLAY Range SET null AS float FORMAT '' ORDERED, [Initial Score]
SIGNIFICANCE 0.05 MAX BRANCHES 0 BREAK true DISCRETIZED (THRESHOLDS, 10) SET null AS float
FORMAT '' DISPLAY Range CONTINUOUS, [Recommended Line] SIGNIFICANCE 0.05 MAX BRANCHES 0 BREAK
true DISPLAY Range SET null AS float FORMAT '' ORDERED, [AgentID] SIGNIFICANCE 0.05 MAX
BRANCHES 0 BREAK true DISCRETIZED (THRESHOLDS, 50) SET null AS null DISPLAY Present, [Number
Inquiries] SIGNIFICANCE 0.05 MAX BRANCHES 0 BREAK true DISPLAY Range SET null AS float FORMAT
'' ORDERED, [Worst Rating Des] SIGNIFICANCE 0.05 MAX BRANCHES 0 BREAK true DISCRETIZED
(THRESHOLDS, 10) SET null AS null DISPLAY Present, [Year Last Check] SIGNIFICANCE 0.05 MAX
BRANCHES 0 BREAK true DISPLAY Range SET null AS float FORMAT '' ORDERED, [Major Derog Num]
SIGNIFICANCE 0.05 MAX BRANCHES 0 BREAK true DISPLAY Range SET null AS float FORMAT '' ORDERED,
[Satisfactory Num] SIGNIFICANCE 0.05 MAX BRANCHES 0 BREAK true DISPLAY Range SET null AS float
FORMAT '' ORDERED, [Year at Address] SIGNIFICANCE 0.05 MAX BRANCHES 0 BREAK true DISCRETIZED
(THRESHOLDS, 10) SET null AS float FORMAT '' DISPLAY Range CONTINUOUS, [Bank Reference]
SIGNIFICANCE 0.05 MAX BRANCHES 0 BREAK true DISPLAY Range SET null AS float FORMAT '' ORDERED,
[Fin Co Ref] SIGNIFICANCE 0.05 MAX BRANCHES 0 BREAK true DISPLAY Range SET null AS float FORMAT
'' ORDERED, [Dept Store Ref] SIGNIFICANCE 0.05 MAX BRANCHES 0 BREAK true DISPLAY Range SET null
AS float FORMAT '' ORDERED, [Bank Dda Ref] SIGNIFICANCE 0.05 MAX BRANCHES 0 BREAK true DISPLAY
Range SET null AS float FORMAT '' ORDERED, [Bank Save Ref] SIGNIFICANCE 0.05 MAX BRANCHES 0
BREAK true DISPLAY Range SET null AS float FORMAT '' ORDERED, [Credit Card Ref] SIGNIFICANCE
0.05 MAX BRANCHES 0 BREAK true DISPLAY Range SET null AS float FORMAT '' ORDERED, [Region]
SIGNIFICANCE 0.05 MAX BRANCHES 0 BREAK true DISCRETIZED (THRESHOLDS, 13) SET null AS null
DISPLAY Present, [Avg Monthly Balance] SIGNIFICANCE 0.05 MAX BRANCHES 0 BREAK true DISCRETIZED
(THRESHOLDS, 10) SET null AS float FORMAT '' DISPLAY Range CONTINUOUS, [Total Amount Purchased]
SIGNIFICANCE 0.05 MAX BRANCHES 0 BREAK true DISCRETIZED (THRESHOLDS, 10) SET null AS float
FORMAT '' DISPLAY Range CONTINUOUS, [Total Service Charge Revenue] SIGNIFICANCE 0.05 MAX
BRANCHES 0 BREAK true DISCRETIZED (THRESHOLDS, 10) SET null AS float FORMAT '' DISPLAY Range
CONTINUOUS, [Profit] SIGNIFICANCE 0.05 MAX BRANCHES 0 BREAK true DISCRETIZED (THRESHOLDS, 10)
SET null AS float FORMAT '' DISPLAY Range CONTINUOUS, [Delinquent Lev 1] SIGNIFICANCE 0.05 MAX
BRANCHES 0 BREAK true DISPLAY Range SET null AS float FORMAT '' ORDERED, [Delinquent Lev 2]
SIGNIFICANCE 0.05 MAX BRANCHES 0 BREAK true DISPLAY Range SET null AS float FORMAT '' ORDERED,
[Delinquent Lev 3] SIGNIFICANCE 0.05 MAX BRANCHES 0 BREAK true DISPLAY Range SET null AS float
FORMAT '' ORDERED, [Amount Lost] SIGNIFICANCE 0.05 MAX BRANCHES 0 BREAK true DISCRETIZED
(THRESHOLDS, 10) SET null AS float FORMAT '' DISPLAY Range CONTINUOUS, [age] SIGNIFICANCE 0.05
MAX BRANCHES 0 BREAK true DISCRETIZED (THRESHOLDS, 10) SET null AS float FORMAT '' DISPLAY
Range CONTINUOUS, [Residential Status] SIGNIFICANCE 0.05 MAX BRANCHES 0 BREAK true DISCRETIZED
(THRESHOLDS, 5) SET null AS null DISPLAY Present, [Status] SIGNIFICANCE 0.05 MAX BRANCHES 0
BREAK true DISCRETIZED (THRESHOLDS, 3) SET null AS null DISPLAY Present, [Account_Status] IS (
CASE WHEN ([Status] IN('Bad Debt')) THEN 'Bad Debt' WHEN ([Status] IN('Closed', 'OK')) THEN
'OK' END ) PREDICT) Using  [Tree]
(GrowAutomaticMaxDepth=1,MultiTree=false,Algorithm='KnowledgeSEEKER',Measure='EntropyVariance',
ClusterMethod='0',DiscreteFilterThreshold='0.01',ContinuousFilterThreshold='0.01',UserBonferron
iAdjustment='1',GrowAutomaticStopSize='300',MinWeight='100',GrowAutomaticMaxDepth='1',AllowCons
ecutiveSplits='True',DataSource='[Credit].[TrainingData]') FROM [Credit].[TrainingData]

INSERT INTO [Credit].[Account_Status Tree] FROM (SELECT [Credit Line], [Initial Score],
[Recommended Line], [AgentID], [Number Inquiries], [Worst Rating Des], [Year Last Check],
[Major Derog Num], [Satisfactory Num], [Year at Address], [Bank Reference], [Fin Co Ref], [Dept
Store Ref], [Bank Dda Ref], [Bank Save Ref], [Credit Card Ref], [Region], [Avg Monthly
Balance], [Total Amount Purchased], [Total Service Charge Revenue], [Profit], [Delinquent Lev
1], [Delinquent Lev 2], [Delinquent Lev 3], [Amount Lost], [age], [Residential Status],
[Status], [Account_Status] FROM TABLE [Credit].[TrainingData])
```

Note: Only the **Run** command is available for the DMX Train & Score Code Generation node. *Run to Here* is not available on the context menu for this node.

For instructions on running command line tools with DMX query files, see the topic Command Line Tools.

See also:

- DMX Code Generation
- DMX Score Code Generation
- Query Log
- Command Line Tools

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

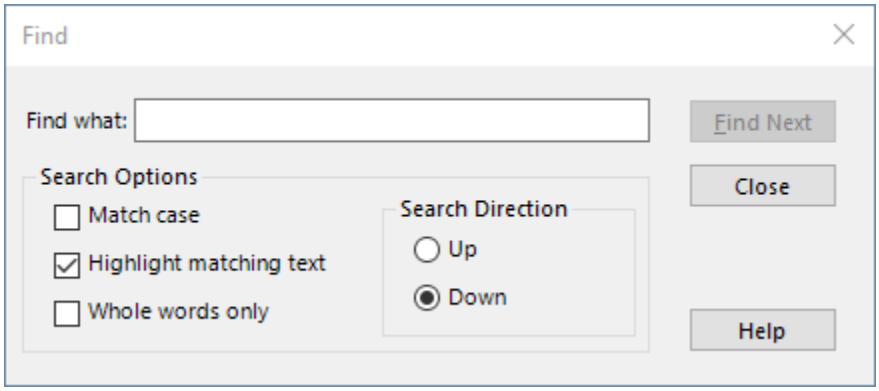
Find and Replace Text

To find or replace text in various views and frames, use the common keyboard shortcuts **CTRL+F** and **CTRL+H**, respectively.

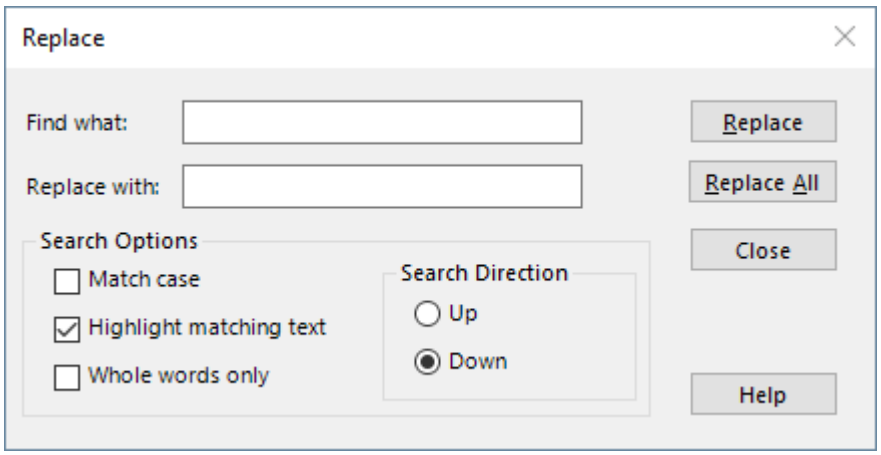
These shortcuts work, for example, in the following dialogs and frames:

- Query Log
- ODBC / Code Log
- Expression Editor in Variable Transformations and Strategy Tree Calculations
- Expression Editor in Optimization node wizards

Pressing **CTRL+F** invokes the Find dialog. Enter the text to find and select the desired search options. Then click *Find Next*.



Pressing **CTRL+H** invokes the Replace dialog. It is only applicable to editable frames such as Expression Editor. Enter the text to find, the text to replace it with, and select the desired search options. Then click *Replace* or *Replace All*.



See also: Field Finder

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Printing

Printing a Tree

To print a tree, select **File | Print** from the menu or press ***Ctrl P*** on your keyboard when the tree is open.

The **File | Print Preview** command provides advanced page scaling options that are especially useful when printing large trees.

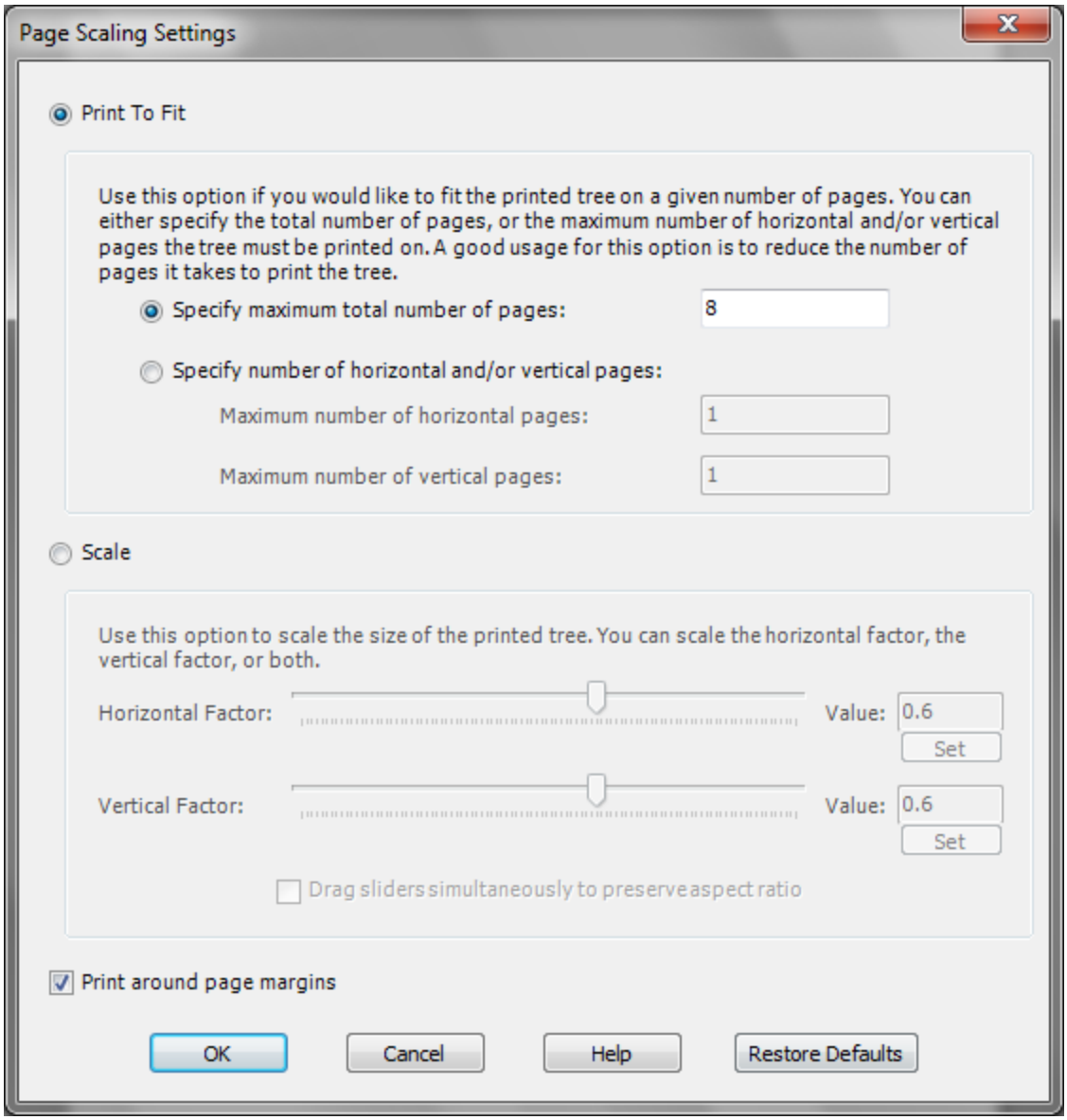
The tree printing algorithm is designed to preserve the integrity of tree nodes, i.e., nodes are never split across adjacent pages. A tree that spans over two or more pages in the Print Preview window may not look exactly as it does in the original Tree view, since the length of its branches may be adjusted for printing purposes to prevent the splitting of nodes across adjacent pages.

The last but one page in Tree Preview shows the correct layout of the printed pages - the way you should arrange them to get the full image of the tree.

Open the tree you want to print and select **File | Print Preview** from the menu.

In the Print Preview window, select **Page Scaling...** from the menu.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.



In the Page Scaling Settings page, you can specify

- the maximum number of pages
- the maximum number of vertical and horizontal pages
- the horizontal and vertical scale factors for the tree
- whether to use page margins

Page Setup (settings for paper size, orientation, and margins) can be accessed either directly by selecting **File | Page Setup...** from the main menu, or from the Print Preview menu.

To print a tree to a PDF file, see the topic Printing to PDF.

Printing Charts, Grids, and Reports

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

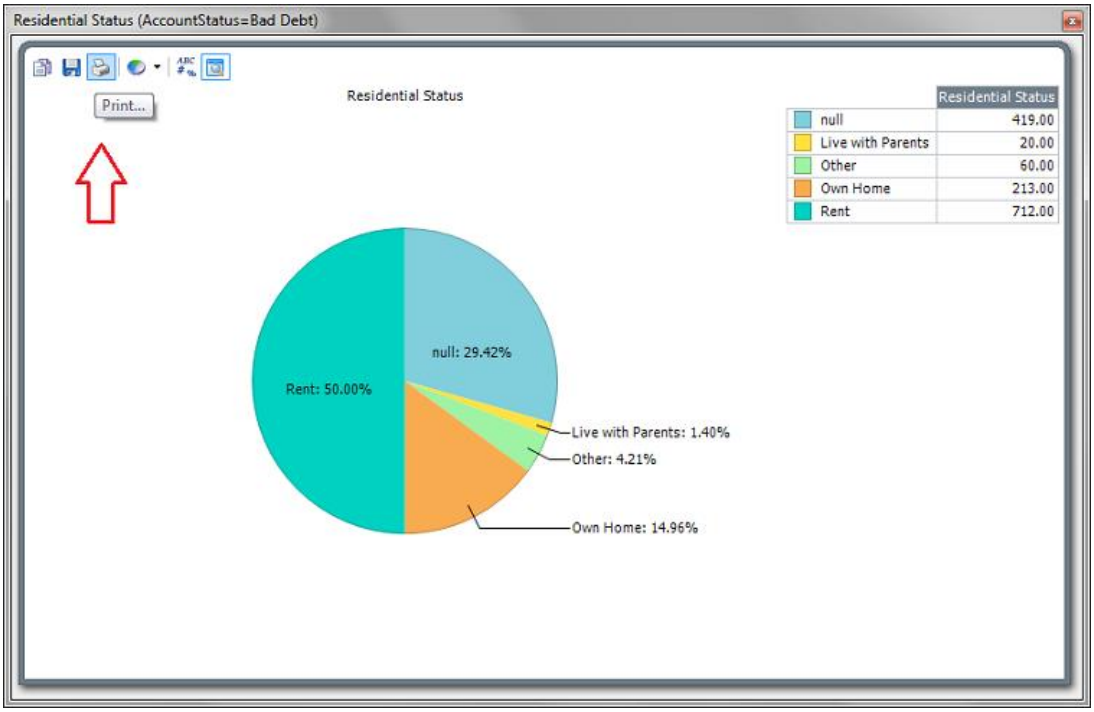
Any chart, grid, or report in the application can be printed using the menu command **File | Print** or pressing **Ctrl P** on your keyboard.

Use the **File | Print Preview** command to preview the object to be printed. As usual, Page Setup (settings for paper size, orientation, margins, etc.) can be accessed either directly by selecting **File | Page Setup...** from the main menu, or from the Print Preview menu.

- **Charts:**

  Whether it is a single chart view (such as Dataset Chart) or a multi-chart view (such as Segment Viewer), the active view will be printed exactly as it is displayed.

  An individual chart in a multi-chart view such as Segment Viewer can be printed by double-clicking the chart and then clicking the Print icon in the chart window:



  If the chart toolbar is not displayed, right-click anywhere on the background (not on the graph itself) and select the Toolbar option.

  To print the underlying data of a chart, select the menu command **Chart | Copy To Clipboard | As Text**. Then open Microsoft Excel or another MS Office application, paste the chart data and print from there.

- **Reports**

  Reports in the HTML format such as Model Overview for Regression models, Parameters and Attributes, Validation and Scoring Results are printed as any other HTML pages.

- **Grids**

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Any grid can be printed either directly from the application or copied (**Edit | Copy**) and pasted to Microsoft Excel or another MS Office application and printed from there.

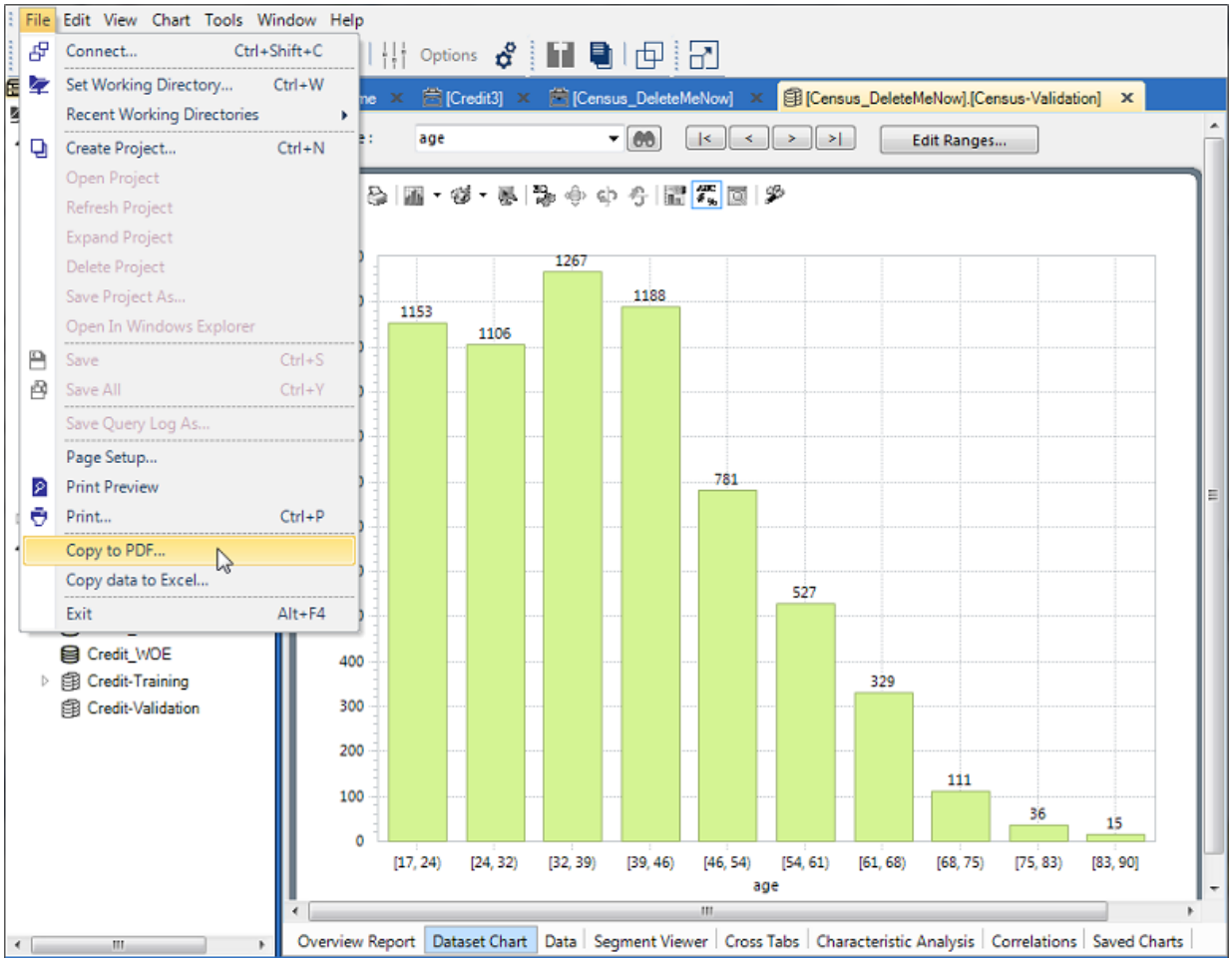
To print a grid, chart, or report to a PDF file, see the topic Printing to PDF.

Printing to PDF

Any view, including decision or strategy trees, charts, tables and reports can be saved in the PDF format (printed to PDF) using the menu command **File | Copy To PDF**.

Note that the advanced scaling options of the Print Preview feature for decision trees and strategy trees are not applicable when using the Copy to PDF function.

Open the view that you want to save to a PDF file and select **File | Copy To PDF** from the main menu. You will be prompted to specify the output file name and location. Enter the file name and click **Save**.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Troubleshooting the KS PDF Printer Issues

If the **Copy to PDF** command is not available under the File menu, KS PDF Printer either failed to install or was uninstalled from your computer. To reinstall KS PDF Printer, do the following:

- open the Windows command line prompt as administrator (on your Windows desktop, select **Start --> All Programs --> Accessories**, right-click the Command Prompt item in the Accessories list, and select "*Run as administrator*".
- change to the directory C:\Program Files\Altair Data Analytics\KS Workstation 2022.3.0\bin
- run **AngossPDFInstall.cmd**

Version compatibility

When using KS Workstation 2022.3 side by side with an older version, users may encounter the error "*Printer not activated, error code - 41*" when using the **Copy To PDF** command under the **File** menu. The workaround is to reinstall both versions of the KS PDF Printer as follows:

Run the Command Prompt as administrator and execute the following commands in this order (version 2020.3 is used as an example of an older version):

```
cd "C:\Program Files\Altair Data Analytics\KS Workstation 2022.3.0\bin"

Install -s -u "KS PDF Printer"

Install -s "KS PDF Printer"

cd "C:\Program Files\Altair Knowledge Works\KS Workstation 2020.3.0\bin"

Install -s -u "KS PDF Printer"

Install -s "KS PDF Printer"
```

See also:

Printing Charts, Grids, and Reports

Printing a Tree

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Mathematical Formulations

Decision Tree

1.1 Notation

Y - dependent variable (DV);

N - number of records at the node.

For a given independent variable (IV):

Category - a group of values that a variable can take (ranging from a single value to all possible values);

r - number of disjoint categories;

v_i - i-th category;

n_i - number of records at the node with IV in v_i .

1.1.1 Numeric DV

$y_{i1},\dots,y_{in_i}$ - values of DV for records with IV in v_i ;

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

$$y_i = \sum_{j=1}^{n} y_{ij} \;;$$

$$\bar{y}_i = \frac{y_i}{n_i} \;;$$

$$y = \sum_{i=1}^{r} y_i = \sum_{i=1}^{r} \sum_{j=1}^{n_i} y_{ij} \;;$$

$$\bar{y} = \frac{y}{N};$$

$$STD = \sum_{i=1}^{r} \sum_{j=1}^{n} (y_{ij} - \bar{y})^2 \;;$$

$$SSTR = \sum_{i=1}^{r} n_i (\bar{y}_i - \bar{y})^2 \;;$$

$$SSE = \sum_{i=1}^{r} \sum_{j=1}^{n_i} (y_{ij} - \bar{y}_i)^2 \;;$$

$$MSTR = \frac{SSTR}{r-1} \;;$$

$$MSE = \frac{SSE}{N-r} \;;$$

$$F^* = \frac{MSTR}{MSE} \;;$$

$$PV = F(F^*; r-1, N-r) \;.$$

If *i* and *k* are two categories of the IV then:

$$MSTR_{\{i,k\}} = n_i \left(\bar{y}_i - \frac{y_i + y_k}{n_i + n_k} \right)^2 + n_k \left(\bar{y}_k - \frac{y_i + y_k}{n_i + n_k} \right)^2 \;;$$

$$MSE_{\{i,k\}} = \frac{\sum_{j=1}^{n_i} (y_{ij} - \bar{y}_i)^2 + \sum_{j=1}^{n_k} (y_{kj} - \bar{y}_k)^2}{n_i + n_k - 2} \;;$$

$$F^*_{\{i,k\}} = \frac{MSTR_{\{i,k\}}}{MSE_{\{i,k\}}} \;.$$

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

1.1.2 Categorical DV

d - number of distinct values that DV can take;

w_j - jth value of DV;

q_{ij} - number of records with DV= w_j and IV in v_i ;

m_j - number of records with DV= w_j ;

$$\bar{q}_{ij} = \frac{n_i}{N} m_j .$$

We have:

$$m_j = \sum_{i=1}^{r} q_{ij}, \quad n_i = \sum_{j=1}^{d} q_{ij}, \quad N = \sum_{j=1}^{d} m_j = \sum_{i=1}^{r} n_i = \sum_{i=1}^{r}\sum_{j=1}^{d} q_{ij} ;$$

$$X^2 = \sum_{i=1}^{r}\sum_{j=1}^{d} \frac{(q_{ij} - \bar{q}_{ij})^2}{\bar{q}_{ij}} ;$$

$$X^2_{\{i,k\}} = \sum_{j=1}^{d} \frac{(q_{ij} - \bar{q}_{ij})^2}{\bar{q}_{ij}} + \sum_{j=1}^{d} \frac{(q_{kj} - \bar{q}_{kj})^2}{\bar{q}_{kj}} ;$$

$$PV = \chi^2(X^2; df) \text{ where } df = (d-1)(r-1) ;$$

$$E_i = -\frac{1}{\ln d} \sum_{j=1}^{d} \frac{q_{ij}}{n_i} \ln \frac{q_{ij}}{n_i} ;$$

$$E = -\frac{1}{\ln d} \sum_{j=1}^{d} \frac{m_j}{N} \ln \frac{m_j}{N} ;$$

$$G = 1 - \frac{\sum_{j=1}^{d} m_j^2}{N^2} ;$$

$$G_i = 1 - \frac{\sum_{j=1}^{d} q_{ij}^2}{n_i^2} .$$

1.2 Algorithms

Treatment of a Numeric Independent Variable

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

To use a numeric IV in a decision tree it must be made categorical. The range of values of the IV is subdivided into a user-controlled number of intervals (set by default to 10) so that the number of records for each interval is the same. Each interval is then considered a category.

1.2. Knowledge Seeker

Finding a Split for a given IV

The split is found by iteratively grouping categories of the IV into new categories. An eligible pair of categories is grouped into a single category if the two are “similar” with respect to the DV. For an unordered IV all pairs are eligible. For an ordered IV only adjacent pairs are. There are two grouping methods: exhaustive and cluster.

Exhaustive Method

This method starts with each value of the IV being a separate category. The following occurs at each iteration:

• For every eligible pair of categories (*i,k*) its similarity statistic is computed. For a numeric DV this statistic is $F^{*}_{(i,k)}$. For a categorical DV it is $X^{2}_{(i,k)}$;

• The pair with the highest similarity statistic is merged into a single category thus decreasing the total number of categories *r* by 1. The new set of categories is the next candidate split and its p-value is computed. For a numeric DV the p-value is $F(F^{*};r-1,N-r)$. For a categorical DV it is $\chi^{2}(X^{2};df)$.

These two steps are repeated until *r* = 1. Of all candidate splits, the one with the highest *Info* (see below) is chosen. If *Info* is above the user-controlled threshold, the split is presented for that IV. Otherwise no significant splits for the IV can be found.

Cluster Method

This method also starts with each value of the IV being a separate category. The following occurs at each iteration:

• For every eligible pair of categories (*i,k*) its similarity statistic is computed. For a numeric DV this statistic is $F^{*}_{(i,k)}$. For a categorical DV it is $X^{2}_{(i,k)}$;

• The pair with the highest similarity statistic is merged into a single category thus decreasing the total number of categories *r* by 1. The new set of categories is the next candidate split and its p-value is computed. For a numeric DV the p-value is $F(F^{*};r-1,N-r)$. For a categorical DV it is $\chi^{2}(X^{2};df)$.

These two steps are repeated until the p-value of the pair with the highest similarity statistic found in step 1 exceeds the user-controlled threshold specified in the *P-Value* column of the Tree Attribute

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Editor (accessible from the Tools menu), where it is set by default to 0.05, or until *r* = 1. The above p-value is equal to $F(F^{*}_{\{i,k\}};1,N-2)$ for a numeric DV and $\chi^2(X^2_{\{i,k\}};d-1)$ for a categorical DV. Of all candidate splits, the one with the highest *Info* is chosen. After this split is found the algorithm attempts to improve the significance by taking single DV values from categories to which they were assigned and grouping them with other categories. If improvement is achieved, the split that produced it becomes the candidate. This re-splitting process returns a possibly different split from the one that merging produced. If its *Info* is above the user-controlled threshold the split is presented for that IV. Otherwise no significant splits for the IV can be found.

Note: The exact technical details of re-splitting are available on request.

Measuring Informativeness of splits obtained for each IV

In the first part of the algorithm at most one split has been found for each IV. Informativeness of these splits is computed according to the split measure in effect. Split measures are described below. A split is informative if it passes the user-controlled threshold, set in the Measure tab of the Tree Configuration dialog - Tools menu - or in Step 3 of the Insert Tree wizard in the Maximum Attr. Info and Minimum Attr. Info fields. The user can choose any informative split. The most informative one is chosen by default.

Growth Control Parameters

The tree growth is also constrained by a number of user-controlled parameters found in the *Tree Growth* tab of the **Options** and **Preferences** dialogs (see Options or Preferences under the **Tools** menu).

The following parameters affect the Find Split and Automatic Grow operations:

Minimum Node Creation Size - Nodes smaller than the specified size will not be created. More precisely: If, at the end of a splitting algorithm, a node in the found split contains fewer records than this number, that node is merged with an adjacent node (for an ordered IV) or with the most statistically similar node (for an unordered IV);

Auto-Grow Stop Size (min. records) - If a node contains fewer records than specified by this parameter, that node will not be split;

Auto-Grow Maximum Tree Depth - If the path from the root node to a given node contains as many nodes as specified by this parameter, that node will not be split. When this parameter is equal to 0 (as it is by default) it means that tree depth is not restricted.

Minimum DV Values - These parameters apply to trees with discrete dependent variables only. They define the minimum threshold values for the frequencies or percentages (if the As Percent option is selected) of the DV values in nodes to be created. Nodes with frequencies (or percentages) of DV values below these thresholds will not be created.

1.3 Measures for Splitting a Node

Split Statistics

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

When evaluating the significance of a split, Knowledge Studio calculates a number of statistics (apart from record counts and proportions) that can be viewed in the Split Report tab of a decision tree or displayed directly in the tree via settings in the Node Detail - Edit menu. Which statistics are reported is measure-dependent except *Info* which is given for every measure and DV type (it can only be viewed in the Split Report).

Below we provide the list of statistics with mathematical expressions for each measure and DV type.

1.3.1 Statistical Measures

A split is a partition of the node dataset into subsets according to the value of the independent variable. According to a statistical measure the best split is the one that generates subsets which are least likely to come from the same distribution. The likelihood is measured by standard statistical tests.

A split is presented if it passes a predetermined significance level. This level can be chosen in two ways:

1. With an *unadjusted* p-value measure the desired significance level is directly applied to the best split .

2. The *adjusted* measure takes into account the fact that a number of different splits is evaluated for every IV (see *Algorithms* above) and only one of them is required to be significant. This split has to pass a more stringent significance test than if only a single split was attempted. The choice of that stringent test is made within the Bonferroni framework by adjusting the p-value of the split. Exact technical details of this adjustment are beyond the current scope of this document and are available on request.

Unadjusted p-value

1. Numeric DV

- $Info = 100 \times C$;
- $df1 = N - r$;
- $df2 = r - 1$;
- $F = F^* = \frac{MSTR}{MSE}$;
- $P = PV = F(F^*; df2, df1)$;
- $C = 1 - P$.

2. Categorical DV

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- $Info = 100 \times C$;
- $df = (d-1)(r-1)$;
- $Chi2 = X^2 = \sum_{i=1}^{r} \sum_{j=1}^{d} \frac{(q_{ij} - \overline{q}_{ij})^2}{\overline{q}_{ij}}$;
- $P = PV = \chi^2(X^2; df)$;
- $C = 1 - P$.

Adjusted p-value

1. Numeric DV

- $Info = 100 \times C$;
- $df1 = N - r$;
- $df2 = r - 1$;
- $F = F^* = \frac{MSTR}{MSE}$;
- $uP = PV = F(F^*; df2, df1)$;
- $uC = 1 - uP$;
- $P = B \times uP$;
- $C = 1 - P$;
- $B = B_a B_m$;

- B_m is the additional adjustment factor that the user can specify in the Measure tab of the Tree Configuration dialog when adjusted measure is chosen. By default it is set to 1;

- B_a is the Bonferroni adjustment factor determined automatically by the algorithm (see above).

Note: B_a depends on the number of distinct values of IV(*r*), the number of categories formed by the algorithm (Cluster or Exhaustive), the algorithm itself, and the grouping type of IV. B_a is greater for unordered than for ordered IV, for Cluster than for Exhaustive and it increases with the cardinality of IV.

2. Categorical DV

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- $Info = 100 \times C$;
- $df = (d-1)(r-1)$;
- $Chi2 = X^2 = \sum_{i=1}^{r} \sum_{j=1}^{d} \frac{(q_{ij} - \overline{q}_{ij})^2}{\overline{q}_{ij}}$;
- $uP = PV = \chi^2(X^2; df)$;
- $uC = 1 - uP$;
- $P = B \times uP$;
- $C = 1 - P$;
- $B = B_a B_m$;

• B_m is the additional adjustment factor that the user can specify in the Measure tab of the Tree Configuration dialog when adjusted measure is chosen. By default it is set to 1;

• B_a is the Bonferroni adjustment factor determined automatically by the algorithm (see above).

Note: B_a depends on the number of distinct values of IV(*r*), the number of categories formed by the algorithm (Cluster or Exhaustive), the algorithm itself, and the grouping type of IV. B_a is greater for unordered than for ordered IV, for Cluster than for Exhaustive and it increases with the cardinality of IV.

1.3.2 Non-statistical Measures

Entropy Variance

This quantity tells how much information about a DV value is contained in a random draw from the node dataset. If DV values were spread evenly within that set then, by drawing a record at random, one would gain no information because every DV value would be equally likely. If, on the contrary, all records had the same DV value, a random draw would provide perfect information about it. High entropy means low information and vice versa. Entropy E achieves a maximum of 1 for a uniform (evenly spread) distribution and a minimum of 0 for a degenerate (peaked) distribution.

1. Numeric DV

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- $Info = 100 \times RatioVariance$;
- $InputVariance = STD$;
- $OutputVariance = SSE$;
- $RatioVariance = 1 - \frac{OutputVariance}{InputVariance}$.

2. Categorical DV

- $Info = 100 \times RatioEntropy$;
- $InputEntropy = E$;
- $OutputEntropy = \sum_{i=1}^{r} \frac{n_i}{N} E_i$;
- $RatioEntropy = 1 - \frac{OutputEntropy}{InputEntropy}$.

Gini Variance

Gini variance is analogous to entropy variance in that it also measures how evenly (or unevenly) DV values are spread within a dataset.

1. Numeric DV

- $Info = 100 \times RatioVariance$;
- $InputVariance = STD$;
- $OutputVariance = SSE$;
- $RatioVariance = 1 - \frac{OutputVariance}{InputVariance}$.

2. Categorical DV

- $Info = 100 \times RatioIndex$;
- $InputIndex = G$;
- $OutputIndex = \sum_{i=1}^{r} \frac{n_i}{N} G_i$;
- $RatioIndex = 1 - \frac{OutputIndex}{InputIndex}$.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

1.4 Thresholds for Informativeness of Splits

In the description of algorithms above it was mentioned that only informative splits (i.e. those with *Info* passing the threshold) are presented. This threshold is set differently for different measure types.

1.4.1 Statistical Measures

For statistical measures the *Info* threshold is derived from the significance level that the user sets in the Measure tab of the Tree Configuration dialog - Tools menu. Splits for which P is below that significance level (and thus *Info* is above the derived threshold) are presented. The default value of the significance level is 0.05.

1.4.2 Non-statistical Measures

For non-statistical measures the user can set maximum and minimum thresholds for *Info*. This can be done in Step 3 of the Insert Tree wizard in the Maximum Attr. Info and Minimum Attr. Info fields. Splits with *Info* value between these two numbers are highlighted even though all splits with positive *Info* are presented.

1.5 Information Value in Trees with Binary DV

Given two probability distributions P and Q, the information value is defined to be

$$IV(P,Q) = D_{KL}(P \| Q) + D_{KL}(Q \| P)$$

where D_{KL} is the *Kullback-Leibler (KL) divergence from P to Q*, given by

$$D_{KL}(P \| Q) = \int_{-\infty}^{+\infty} p(x) \log \frac{p(x)}{q(x)} dx$$

or the equivalent sum, should P and Q be discrete.

Both the information value and the KL divergence are measures of the difference between a “true” probability distribution P and an arbitrary distribution Q. However, D_{KL} is not symmetric, that is,

$$D_{KL}(P \| Q) \neq D_{KL}(Q \| P)$$

so it is important to keep track of what P and Q actually are. In contrast, the Information Value is symmetric:

$$IV(P,Q) = IV(Q,P)$$

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

When the information value is reduced to discrete form, we have (see Siddiqi, N., Credit Risk Scorecards: Developing and Implementing Intelligent Credit Scoring, John Wiley & Sons, 2005, p. 81.):

$$IV = D_{KL}(P \parallel Q) + D_{KL}(Q \parallel P) = \sum_i (P_i - Q_i)\log(\frac{P_i}{Q_i})$$

This measure of the distance between P and Q is available for display in decision trees with binary dependent variable (one DV category becomes P and the other becomes Q).

Note that the information value is undefined if any P_i or Q_i are zero (in this case it is displayed as "null" in the application user interface).

1.6 Tree Pruning

Tree Pruning allows one to simplify a given tree model by reducing the number of nodes on the tree while maintaining the lowest misclassification rate possible.

1.6.1 Error-Complexity Pruning

This algorithm only applies to trees with a discrete DV.

The Error-Complexity Pruning algorithm follows these steps:

1. Build a complete tree.

Calculate α for each subtree (except the first) and select the sub-tree with the smallest value of α for pruning. Repeat this process until there are no sub-trees left, to yield a series of increasingly pruned trees.

$$\alpha = \frac{R(t) - R(T_t)}{N_T - 1}$$

where *t* is a node, *T* is a sub-tree, N_T is the number of nodes for sub-tree *T*, $R(T_t)$ is error cost if a sub-tree is kept, and $R(t)$ is the error cost if the sub-tree becomes a leaf through pruning,

$R(T_t) = \sum R(i)$ for *i* = sub-tree leaves,

and

$$R(i) = r(i)p(i)$$

r (*i*) = Number of wrongly classified records in node *i* / Total population of node *i*,

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

p (*i*) = Total population of node *i* / Total population of the tree;

$$R(t) = r(t)p(t)$$

2. Run a test dataset through the sub-tree with the smallest α, to get the misclassification rate. Repeat this process until you get the tree with the smallest misclassification rate.

1.6.2 Critical Value Pruning

The method estimates the important or strength of a node from calculations done in the tree creation stage.

1. Calculate the critical values for each node. Which critical value is used depends on the algorithm of decision tree.

2. Rank the critical values. The larger the critical value selected, the greater the degree of pruning and the smaller the resulting tree.

3. Validate the tree when it has been pruned, and then calculate the misclassification.

4. Repeat steps 2 and 3 steps until the misclassification rate stops decreasing.

1.6.3 Reduced-Error Pruning

This algorithm only applies to trees with a discrete DV.

This pruning method produces a series of pruned trees by using the test data directly. It ends with the smallest minimum-error tree on the test data.

1. Start with a complete tree and run the test data through it.

2. Count the error cost if a sub-tree is kept ($R(T_t)$) and the error cost if the sub-tree becomes a leaf through pruning ($R(t)$), where *t* is a node, and *T* is a sub-tree,

$R(T_t) = \sum R(i)$ for *i* = sub-tree leaves, and

$$R(i) = r(i)p(i)$$

r (*i*) = Number of wrongly classified records of node *i* / Total population of node *i*,

p (*i*) = Total population of node *i* / Total population of the tree;

$$R(t) = r(t)p(t)$$

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

3. Use the difference between the error costs (if positive) as a measure of the gain from pruning the sub-tree, i.e.,

A Measure = $R(T_t) - R(t)$;

4. Choose the one with largest difference as the sub-tree to prune;

5. Continue this process until further pruning would increase the misclassification rate of whole tree, *R*, *R* = Misclassified Rows / Total Rows.

Predictive Models: Regression

1.1 Notation

The following notation is common to all predictive models. The dataset to which it applies, depending on context, may be the training dataset or the validation dataset.

N - number of observations (records) in the dataset;

$y_1,\ldots,y_N$ - actual DV values in the dataset;

$\hat{y}_1,\ldots,\hat{y}_N$ - predicted DV values;

p - number of independent variables in the model (excluding intercept for regressions).

Numeric DV

$$\bar{y} = \frac{y_1 + \ldots + y_N}{N};$$

$$\bar{\hat{y}} = \frac{\hat{y}_1 + \ldots + \hat{y}_N}{N};$$

$$SSTO = \sum_{i=1}^{N} (y_i - \bar{y})^2;$$

$$SSR = \sum_{i=1}^{N} (\hat{y}_i - \bar{y})^2;$$

$$SSE = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2;$$

$$MDEV = \frac{1}{N} \sum_{i=1}^{N} | y_i - \hat{y}_i |$$

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Categorical DV

d - number of different values that the DV can take;

$v_1,\ldots,v_d$ - possible values of the DV;

π_j^i - predicted probability that the DV is equal to for the *i* th record;

m_i - value index of y_i , i.e., $m_i = m$ if $y_i = v_m$;

$L = -\sum_{i=1}^{N} \ln \pi_{m_i}^i$ - log-likelihood function;

$f_1,\ldots,f_d$ - frequencies of DV values;

$\hat{f}_1,\ldots,\hat{f}_d$ - predicted frequencies of DV values;

$$E = -\frac{1}{\ln d}\sum_{j=1}^{d}\frac{f_j}{N}\ln\frac{f_j}{N};$$

$$\hat{E} = -\frac{1}{\ln d}\sum_{i=1}^{d}\frac{\hat{f}_i}{N}\ln\frac{f_i}{N}.$$

1.1.1 Common Output

After the parameters for a predictive model have been specified, its training progress and results can be seen in three views. Output there depends only on the type of DV and not on that of the model (except regressions where conventional output is provided, see below). Training of any predictive model is performed by minimizing the appropriate objective function using records in the training set.

Results View

There are four numeric outputs in this view.

1. **Sample Size** is the number of records in the training set.

2. **Training Results:**

- **Variance/Entropy Explained.** For a numeric DV it is $1-\frac{SSE}{SSTO}$ or $\frac{SSR}{SSTO}$, for a categorical DV it is $1-\frac{\hat{E}}{E}$;
- **Records correctly predicted.** Self-explanatory;

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- **Percentage correctly predicted.** Self-explanatory.

Training View

The output of analytical interest in this view is green and blue lines that show the training progress. The green line, scaled at the left, shows the value of the objective function divided by the number of records, calculated for the training subset. The blue line shows the value of **Variance/Entropy Explained**, calculated for the validation subset.

Scoring View

Output in the scoring view becomes available after the model has been validated using the dataset specifically reserved for this purpose. This dataset must have the same structure as the training dataset and the values of the DV must be known for that set. After the validation is finished, the table in the bottom part of the view is filled with output. For a categorical DV the top part of the view displays a confusion matrix.

In the bottom table:

- Training Variance/Entropy Explained is the same number as in the Results View;
- # Records refers to the validation set;
- Invalid Records. A record can be found invalid for a variety of reasons. One is the presence of missing values, especially in the DV field. Another example is the difference in the field structure between the training and validation datasets;
- Correct Records is the number of correct predictions on the validation set;
- Scoring Variance/Entropy Explained is Variance/Entropy Explained calculated for the validation set;
- Mean Deviation is only calculated for a numeric DV and is equal to *MDEV* above.

1.1.2 Regression Models

Linear or logistic regression is used to predict the value of a DV using values of a set of numeric or categorical independent variables (IV).

Inclusion of a categorical variable as IV in a regression is organized as follows. Consider a categorical variable with *n* distinct values. These values are ordered lexicographically by their names. The first value in this ordering is chosen as reference. For every other value a design variable is created. Thus, *n*-1 design variables are added to the regression. The default choice of the reference value can be changed in Model Attribute Editor.

Introducing the following notation:

N - number of observations included in the regression;

p - number of variables in the regression (excluding intercept);

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

$$X = \begin{bmatrix} x_{11} & \dots & x_{1p} & 1 \\ x_{21} & \dots & x_{2p} & 1 \\ . & . & & . \\ . & . & & . \\ . & . & & . \\ x_{N1} & \dots & x_{Np} & 1 \end{bmatrix}$$ - matrix of IV values (design matrix; in models without intercept, the last column is omitted);

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ . \\ y_N \end{bmatrix}$$ - column vector of DV values;

$$b = \begin{bmatrix} b_0 \\ b_1 \\ . \\ . \\ . \\ b_p \end{bmatrix}$$ - column vector of estimated parameters (in models without intercept, b_0 is omitted);

$$J = \begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & 1 & \dots & 1 \\ . & . & & . \\ . & . & & . \\ . & . & & . \\ 1 & 1 & \dots & 1 \end{bmatrix}$$ - *N* x *N* matrix of 1s.

Linear Regression

In a linear regression, the DV must be numeric. Knowledge Studio estimates linear regression by ordinary least squares. Thus, the vector of parameter estimates is equal to

$$b = (X'X)^{-1} X'Y$$

It is computed using singular value decomposition of *X*.

Regression output can be found in the **Results** view of the predictive model when one of the sequences is chosen in the Output to view menu. The output has the following format:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Sequence Statistics

Dependent Variable: *DV name*

Variance Explained: $\frac{SSR}{SSTO}$

Adjusted Variance Explained: $1-\frac{MSE}{MSO}$

F-Ratio: $\frac{MSR}{MSE}$

F-Ratio Degrees of Freedom 1 / 2: *RDF* / *EDF*

P-Value: $F\left(\frac{MSR}{MSE};RDF,EDF\right)$

Analysis of Variance

Source	Sum-of-Squares	DF	Mean-Square
REGRESSION	*SSR*	*RDF*	*SSR / RDF*
ERROR	*SSE*	*EDF*	*SSE / EDF*
TOTAL	*SSTO*	*TDF*	*SSTO / TDF*

Independent Variable Statistics

The following columns are displayed:

Variable Name/Value: Values for categorical variables

Model Parameters: b_j

Parameter Standard Error: $SE(b_j)=\left(MSE\cdot(XX)^{-1}_{jj}\right)^{1/2}$

Wald Statistics: $\left[\frac{b_j}{SE(b_j)}\right]^2$

Significance: $\chi^2\left(\left[\frac{b_j}{SE(b_j)}\right]^2,1\right)$

In the above:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

TDF = *N* - number of observations in the regression;

RDF = *p* - number of variables in the regression excluding the intercept;

EDF = *TDF* - *RDF* ;

$$SSR = b'X'Y - \frac{1}{N}Y'JY;$$

$$SSTO = Y'Y - \frac{1}{N}Y'JY = y_1^2 + \ldots + y_N^2 - \frac{1}{N}(y_1 + \ldots + y_N)^2$$

;

$$SSE = SSTO - SSR = (Y - Xb)'(Y - Xb)$$

After the set of variables to consider for regression is chosen, there is an option for performing variable selection. Regressions that are run during variable selection are called sequences and their output is available for viewing (see above). Six methods are available:

1. Forward Selection starts with the base model (which includes the intercept and all variables chosen to be forced into the model by the user) and adds variables in the order that has been set in Step 4 of the Predictive Model Wizard (page "Must Include Selection") starting from the first.

2. Backward Selection starts with the full model (all variables from the set) and deletes variables in the order determined in Step 4 of the Predictive Model Wizard (page "Must Include Selection") starting from the last.

3. Stepwise Selection starts with the base model and, at every step, adds the variable with the highest contribution to the explanatory power of the model chosen in the previous step in terms of the partial *F* test, if this contribution is higher than the predetermined level chosen simultaneously with the variable selection method. This is called ***Significance Limits - Entry*** and by default is set to 0.05. After addition every variable added before is checked and the variable with the smallest contribution is dropped if its Wald statistic is below the predetermined level (***Significance Limits - Exit*** with a default value of 0.05). The selection process also stops if a variable is cycled (i.e., included and excluded) three times.

6. R-Squared Selection simply runs regressions on all subsets of the chosen set of variables.

Note: For discrete independent variables, the dummy variables that derive from one independent variable are considered as a unit for entry or exit.

Logistic Regression

For logistic regression the DV must be categorical. Knowledge Studio estimates binary or polytomous logistic regression. Parameters are estimated by the maximum likelihood method.

Let *Y* take values in $\{v_0,\ldots,v_d\}$ with v_0 being the reference value. Denote:

$\pi_j(x_i) = P(Y = v_j \mid x_i)$, $j = 0,\ldots,d$;

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

$$g_j(x_i) = \ln \frac{\pi_j(x_i)}{\pi_0(x_i)} = b_{j0} + b_{j1}x_{i1} + \ldots + b_{jp}x_{ip}$$, $j = 1,\ldots,d$;

$$x_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ . \\ . \\ . \\ x_{ip} \end{bmatrix}$$

Thus,

$$\pi_0(x_i) = \frac{1}{1 + \sum_{k=1}^{p} e^{g_k(x_i)}} \qquad \pi_j(x_i) = \frac{e^{g_j(x_i)}}{1 + \sum_{k=1}^{p} e^{g_k(x_i)}}$$ for $j = 1,\ldots,d$

Let m_i be the value index of y_i , i.e., $m_i = m$ if $y_i = v_m$. Then the log-likelihood function is

$$L(b) = \sum_{i=1}^{N} \ln \pi_{m_i}(x_i) = \sum_{i=1}^{N} \{g_{m_i}(x_i) - \ln[1 + \sum_{k=1}^{p} e^{g_k(x_i)}]\}$$

Logistic regression belongs to a class of non-linear models that require an iterative process to train. During training, the model parameters are determined as best estimates that minimize the error the model would generate. The green line of training process in Knowledge Studio represents the values of Mean Square of Error (*MSE*) between the estimate $L(\hat{b})$ and true $L(b)$, i.e.,

$$MSE = [L(\hat{b}) - L(b)]^2$$.

If a validation dataset is applied during the training, a blue line in the Knowleldge Studio training process will represent the values of Mean Square of Error of the validation data. The *MSE* lines are also applied in Neural Network training process.

The output of a logistic regression has the following format:

Sequence Statistics

Dependent Variable: *DV name*

Entropy Explained: $1 - \frac{L(b)}{L(b_0)}$

Chi-Square: $G = -2[L(b_0) - L(b)]$

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Chi-Square Degrees of Freedom: *p*

P-Value: $\chi^2(G, p)$

Model Fitting Summary

Source	Negative 2 Log-Likelihood	DF
Intercept only	$-2L(b_0)$	-
Full model	$-2L(b)$	*p*

Independent Variable Statistics

The following parameters are displayed:

Dependent variable value

Independent Variable Name/Value. Values are listed for categorical variables.

Model Parameters: b_i

Parameter Standard Error: $SE(b_j) = \Sigma(b)_{jj}$

Wald Statistics: $\left[\frac{b_j}{SE(b_j)}\right]^2$

Significance: $\chi^2\left(\left[\frac{b_j}{SE(b_j)}\right]^2, 1\right)$

Values are listed for categorical variables

In the above:

$L(b_0)$ is the log-likelihood for the regression with intercept only;

$\Sigma(b) = -\left[\frac{\partial^2 L}{\partial b^2}\right]^{-1}$, where $\frac{\partial^2 L}{\partial b^2}$ is the estimated Hessian of the log-likelihood;

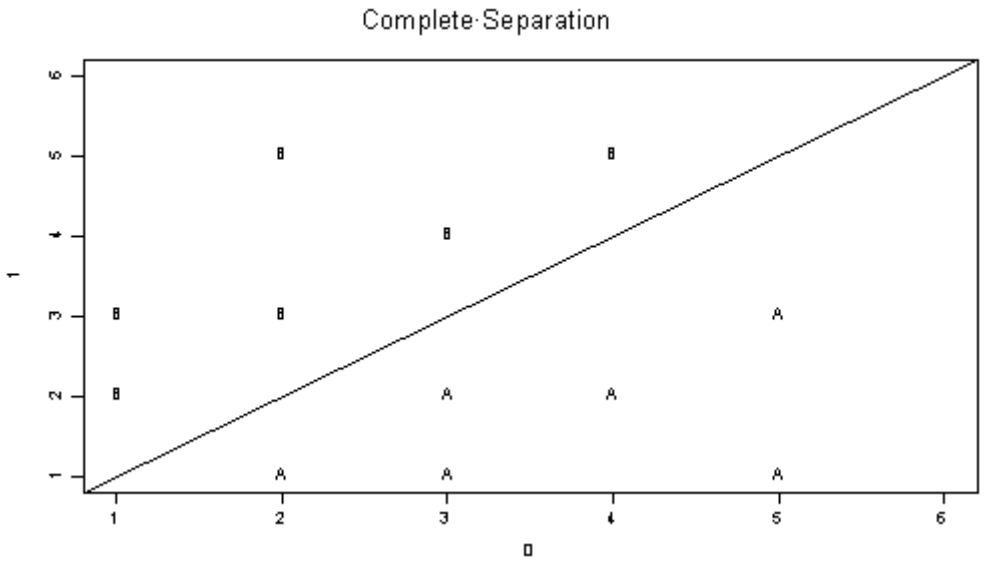
When the DV takes more than two values, the Independent Variable Statistics table is given for every value except the reference one which is by default the first in alphabetical order.

Variable selection methods are identical to those for linear regression except that the likelihood ratio test is used for inclusion and exclusion of variables.

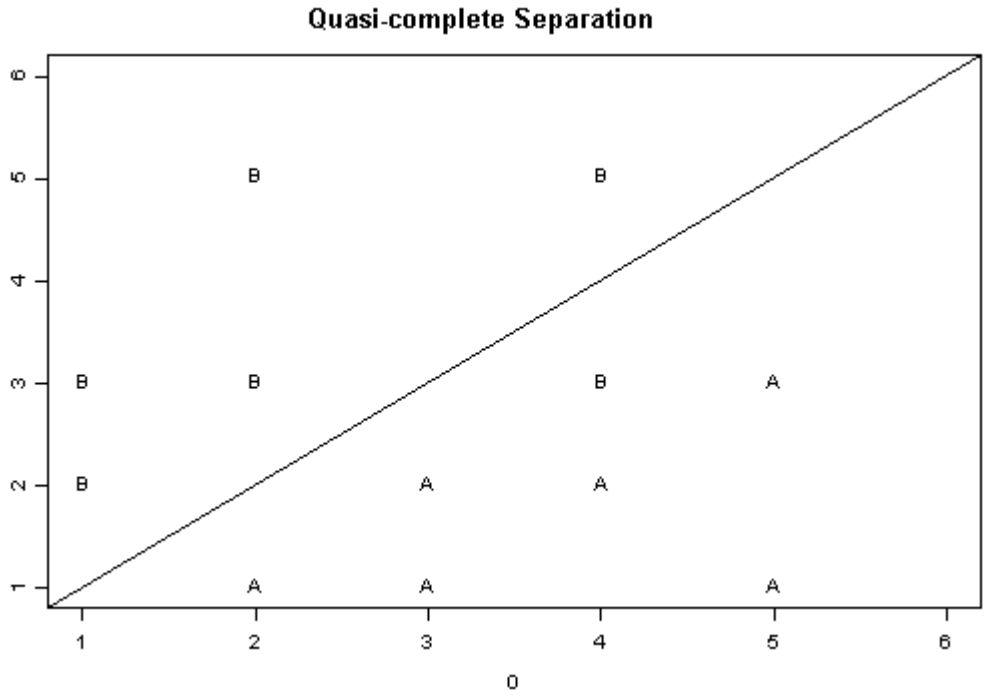
Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The Maximum Likelihood estimate for a logistic model does not always have a finite solution. In order to have a finite maximum likelihood estimate, the data points cannot be linearly separable, in other words, the data points must have some overlap. The types of data configuration without or with few overlapping points are called linearly separable. There are two types of Separable cases:

Complete Separation: When there are no overlap values at all, it is called Complete Separation, as demonstrated in the graph below.



Quasi-complete Separation: When overlap is at a single or a few tied values the configuration is termed quasi-complete separation, as demonstrated in the graph below.



Altair Engineering, Inc. Copyright © 2022. All rights reserved.

A condition in which the independent variables in a regression model are themselves highly correlated is called multicollinearity. Multicollinearity often leads to models in which the coefficients are poorly estimated and, while a fitted model may be good for predictive purposes, it can be difficult to interpret the relative effects of the various independent variables. Knowledge Studio applies Singular Value Decomposition (SVD) to detect the multicollinearity in linear and logistic regression.

SVD methods are based on the following theorem of linear algebra: Any *M* x *N* matrix *A* whose number of rows *M* is greater than or equal to its number of columns *N* can be written as the product of an *M* x *N* column-orthogonal matrix *U*, an *N* x *N* diagonal matrix *W* with positive or zero elements (the singular values), and the transpose of an *N* x *N* orthogonal matrix *V*. The various shapes of these matrices can be expressed as follows:

$$\begin{bmatrix} & A & \end{bmatrix} = \begin{bmatrix} & U & \end{bmatrix} \cdot \begin{bmatrix} w_1 & & \\ & \Lambda & \\ & & w_N \end{bmatrix} \cdot \begin{bmatrix} & V^T & \end{bmatrix}$$

The Singular Value Threshold for Collinear IV Removal (in the Predictive Model wizard step 4 of Knowledge Studio) relates to singular values, w_i . The number of w_i which are less or equal to the value of Singular Value Threshold for Collinear Removal is the number of independent variables that are going to be removed, and the independent variables causing collinearity will be identified by matrix *V*.

For General Linear Model (here refer to Linear and Logistic regression), which is defined as

$$y(x) = \sum_{k=1}^{M} a_k X_k(x)$$

where $X_1(x), \ldots\ldots\ldots\ldots, X_M(x)$ are arbitrary fixed functions of *x*, called the basis functions. The least square error function of General Linear Model equals

$$\sum_{i=1}^{N} \left[\frac{y_i - \sum_{k=1}^{M} a_k X_k(x_i)}{\sigma_i} \right]^2$$

Let *A* be an *N* x *M* matrix:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

$$A=\begin{bmatrix} \frac{X_1(x_1)}{\sigma_1} & \frac{X_2(x_1)}{\sigma_1} & \Lambda & \frac{X_M(x_1)}{\sigma_1} \\ \frac{X_1(x_2)}{\sigma_2} & \frac{X_2(x_2)}{\sigma_2} & \Lambda & \frac{X_M(x_2)}{\sigma_2} \\ \mathrm{M} & \mathrm{M} & & \mathrm{M} \\ \frac{X_1(x_N)}{\sigma_N} & \frac{X_2(x_N)}{\sigma_N} & \Lambda & \frac{X_M(x_N)}{\sigma_N} \end{bmatrix}$$

Vectors *b* and *a*:

$$b=\begin{bmatrix} \frac{y_1}{\sigma_1} \\ \frac{y_2}{\sigma_2} \\ \mathrm{M} \\ \frac{y_N}{\sigma_N} \end{bmatrix}$$

$$a=\begin{bmatrix} a_1 & a_2 & \Lambda & a_M \end{bmatrix}$$

Then the Least Square Error function can be expressed as:

$$\left|A\cdot a-b\right|^2$$

Apply SVD on matrix *A*, and then the variance in the estimate of a parameter a_j is given by

$$\sigma^2(a_j)=\sum_{i=1}^{M}\left(\frac{V_{ji}}{w_i}\right)^2$$

Since the independent variables, which cause collinearity, will lead the divergence of their estimates of parameters, i.e., the large variances of the parameter estimates. Based on the large values of matrix *V*, the independent variables causing collinearity will be identified.

Parameter Optimization Algorithm

BFGS, Newton-Raphson, Levenberg-Marquardt, Classic Back-Propagate, and Conjugate Gradient parameter optimization algorithms are applied in the non-linear regression training process of Knowledge Studio, such as Logistic Regression and Neural Network. However, BFGS is the only algorithm applied in time series models. Descriptions of these algorithms and their mathematic formulas are presented as follows.

Sum of Square Error Function

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

The generalized discriminant function with *M* predefined non-linear functions $\phi_j(x)$ is presented as:

$$y(x) = \sum_{j=0}^{M} w_j \phi_j(x)$$

,

and then Sum of Square Error Function is given:

$$E(w) = \frac{1}{2}\{y(x,w) - t)\}^2$$

,

where $y(x,w)$ represents the output as a function of the input vector *x* and weight vector *w* ; *t* is the target value for output. $E(w)$ is a differentiable function of *w*.

Broyden-Fletcher-Goldfarb-Shanno (BFGS)

From the Newton formula, we get that the weight vector at steps τ and $\tau + 1$ are related to the corresponding gradients by:

$$w^{(\tau+1)} - w^{(\tau)} = -H^{-1}(g^{(\tau+1)} - g^{(\tau)})$$

,

where the Hessian matrix *H* is defined by

$$(H)_{ij} \equiv \frac{\partial E}{\partial w_i \partial w_j}|_{\hat{w}}$$

,

$\hat{w}$ is the estimate for weight w , *g* is the gradient between points w and w^*, w^* is the point which minimizes the error function, and *g* is defined as

$$g = \nabla E = H(w - w^*)$$

.

To avoid the possibility that Newton step may take the search outside the range of validity of the quadratic approximation, the weight vector is updated using:

$$w^{(\tau+1)} - w^{(\tau)} = \alpha^{(\tau)} G^{(\tau)} g^{(\tau)}$$

,

where $\alpha^{(\tau)}$ is chosen to minimize $E(\alpha) = E(w^{(\tau)} + \alpha G^{(\tau)} g^{(\tau)})$, and $G^{(\tau+1)}$, which represents increasingly accurate approximations to the inverse Hessian H^{-1}, and in the BFGS process, is given by:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

$$G^{(s+1)} = G^{(s)} + \frac{PP^T}{P^T V} - \frac{(G^{(s)}V)V^T G^{(s)}}{V^T G^{(s)} V} + (V^T G^{(s)} V)\mu\mu^T,$$

P, *V*, and μ vectors are defined as

$$P = w^{(s+1)} - w^{(s)},$$

$$V = g^{(s+1)} - g^{(s)},$$

$$\mu = \frac{P}{P^T V} - \frac{G^{(s)}V}{V^T G^{(s)} V}.$$

.

Newton-Raphson

The Newton-Raphson method finds the slope (the tangent line) of the function at the current point and uses the zero of the tangent line as the next reference point. The process is repeated until the root is found.

$$w_{new} = w_{old} - H^{-1}\nabla E,$$

where w is the weights, ∇E is a vector with elements ∇E^n (∇E^n is the error for the *n*th pattern), and *H* is the matrix of partial derivatives (Hessian) of these errors with respect to the weights.

The Newton-Raphson method is much more efficient than some other methods for solving a system of nonlinear equations. The method requires the calculation of the derivative of a function at the reference point, which is not always easy. Furthermore, the tangent line may shoot wildly and might occasionally be trapped in a loop. The problem can be resolved by adding to the Hessian a positive-definite symmetric matrix which comprises the unit matrix *I* times a constant factor λ, which is called Levenberg-Marquardt algorithm.

Levenberg-Marquardt

The Levenberg-Marquardt algorithm is designed specifically to minimize the sum-of-squares error function, using a formula that (partly) assumes that the underlying function modeled by the network is linear. The algorithm can make very rapid progress. Levenberg-Marquardt therefore compromises between the linear model and a gradient-descent approach. A move is only accepted if it improves the error.

Its restrictions are list as follows:

Levenberg-Marquardt can only be used on networks with a single output unit.

Levenberg-Marquardt is not good for use on networks of any great size (more than a few hundred weights).

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Levenberg-Marquardt is only defined for the sum squared error functions. It is usually therefore only appropriate for regression networks.

Levenberg-Marquardt uses the update formula

$$w_{new} = w_{old} - (H + \lambda I)^{-1} \nabla E ,$$

where w is the weights, ∇E is a vector with elements ∇E^n (∇E^n is the error for the *n*th pattern), *I* is the unit matrix, and *H* is the matrix of partial derivatives of these errors with respect to the weights. λ is a parameter that governs the step size, and an appropriate value of λ will be given during the minimization process.

Classic Back-Propagation.

The Back-Propagation is applicable only in networks with only one modifiable connection layer. It evaluates the derivatives of the error with respect to the weights, and these derivatives can then be used to find weight values that minimize the error function.

Back-propagation learning proceeds in the following way: The input pattern determines the activations of the input nodes, and the error values of the output layer are being computed. Then the error values for all hidden layer nodes are computed by output layer nodes. At the end, the error values of the hidden layer are used to compute new weights for the output layer. The weights' changes are done using the following formula:

The back-propagate process's four steps for computing the derivatives of the error E^n with respect to the weights are:

1. Apply an input vector X^n to the network and forward propagate through the network using

$$a_j = \sum w_{ij} z_i \quad \text{and} \quad z_j = g(a_j) ,$$

where z_i is the activation of a unit, or input, which sends a connection to unit *j*, and w_{ij} is the weight associated with the connection. *g* is a non-linear activation function, to find the activations of all the hidden and output units.

2. Compute the error δ_k, which are the errors of units *k* sending a connection to unit *j* by using

$$\delta_k \equiv \frac{\partial E^n}{\partial a_k} ,$$

where E^n is a differentiable function of the network output variables.

3. Back-Propagate the δ's using

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

$$\delta_j = g'(a_j)\sum_k w_{kj}\delta_k$$

to obtain δ_j for each hidden unit in the network.

4. Use

$$\frac{\partial E^n}{\partial w_{ji}} = \delta_j z_i$$

to compute the required derivatives.

Conjugate Gradient

The Conjugate Gradient is used to decide a search direction, $d^{(\tau)}$, at each iteration stage of linear search, and the linear search uses following formula to update the weight vector

$$w^{(\tau+1)} = w^{(\tau)} + \lambda^{(\tau)} d^{(\tau)},$$

where τ is the τ's step of iteration the parameter $\lambda^{(\tau)}$ is chosen to minimize the error function along that direction:

$$E(\lambda) = E(w^{(\tau)} + \lambda d^{(\tau)}).$$

If the next direction of conjugate gradient, $d^{(\tau+1)}$, is defined as

$$(d^{(\tau+1)})^T H d^{(\tau)} = 0,$$

where *H* is the Hessian matrix, whose elements are the partial derivatives of the error with respect to the weights.

1.2 Activation Functions

The following is a description of the activation functions available when creating predictive models.

1.2.1 Logistic Function

$$F(x) = 1/(1+\exp(-x)),$$

1.2.2 Inverse Cumulative Normal

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

$$F(x) = \Phi(x) = (2\pi)^{-1/2} \int_{-\infty}^{x} \exp(-z^2/2) dz$$

,

1.2.3 Complementary Log-Log

$$F(x) = 1 - \exp(-\exp(-x))$$

,

1.2.4 Log-Log

$$F(x) = \exp(-\exp(-x))$$

.

Two types of predictive models will use link functions, neural network and logistic regression. In Knowledge Studio, the application of activation functions for binary, multi-categorical, and continuous dependent variable varies among the different predictive models. So there are five possible situations:

1. Logistic Regression with Binary Dependent Variable:4 types of activation functions available.

2. Logistic Regression with Multi-Categorical Dependent Variable. The softmax function, given by the following formula, is applied to the selected activation function.

$$F(x_i) = \frac{\exp(x_i)}{\sum_{j=1}^{c} \exp(x_j)}$$

,

where $i = 1, \Lambda, c$, and c is the number of categories.

3. Neural Network with Binary Dependent VariableOnly logistic (Sigmoid) link function is applied on the hidden layer, but the output layer will use the link function that is chosen.

4. Neural Network with Multi-Categorical Dependent VariableLogistic (Sigmoid) link function is applied on the hidden layer, and the output layer uses the Softmax link function.

5. Neural Network with Continuous Dependent VariableBoth hidden layer and output layer use logistic (Sigmoid) link function.

1.3 Cross Entropy, Entropy of Predict, Entropy of Actual, and K-L Divergence

1.3.1 Cross-entropy

Cross-entropy for discrete distribution

Cross-entropy for two classes is defined as

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

$$E = -\sum_n \left\{ t^n \ln y^n + (1-t^n)\ln(1-y^n) \right\}$$

,

where *t* = 1 if the input vector belongs to class C_1 and *t* = 0 if it belongs to class C_2.

Cross-entropy for multiple classes is defined as

$$E = -\sum_n \sum_{k=1}^{c} t_k^n \ln y_k^n$$

,

consider a network with one output y_k for each class, and $t_k^n = \delta_{ki}$, where δ_{ki} denotes the Kronecker delta, that is, $\delta_{ki} = 1$ if *k* = *I* and $\delta_{ki} = 0$ otherwise.

In fact, the Cross-entropy for two classes is a special case of Cross-entropy for multiple classes.

1.3.2 Entropy of Predict and Entropy of Actual

For discrete distribution, the Entropy is defined as:

$$S = -\sum_i p_i \ln p_i$$

,

where $p_i = N_i / N$, N_i is the sample size of class *i*, and *N* is total sample size. The Entropy of Predict and Entropy of Actual are calculated from the different samples using the same Entropy formula.

For continuous distribution we defined the entropy to be:

$$S = -\int p(X) \ln p(X) dX$$

,

where $X \equiv (x_1, \Lambda\ , x_d)^T$.

1.3.3 K-L Divergence

This statistic is known as a measure of the distance between density function $p(x)$, and its estimate $\tilde{p}(x)$, which is defined as

$$L = -\int p(x) \ln \frac{\tilde{p}(x)}{p(x)} dx$$

.

1.4 Covariance and Correlation

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Let *X* and *Y* be two random variables, and let , $E(X) = \mu_x$, $E(Y) = \mu_y$, $Var(X) = \sigma_x^2$, and $Var(Y) = \sigma_y^2$. The covariance of *X* and *Y*, which is denoted by $Cov(X,Y)$, is defined as follows:

$$Cov(X,Y) = E[(X - \mu_x)(Y - \mu_y)]$$

.

If $0 < \sigma_x^2 < \infty$ and $0 < \sigma_y^2 < \infty$, then the correlation of *X* and *Y*, which is denoted by $\rho(X,Y)$, is defined as follows:

$$\rho(X,Y) = \frac{Cov(X,Y)}{\sigma_x \sigma_y}$$

1.5 Model Reasons

In a number of situations, it is useful to "explain" why a particular model is making a particular prediction. This is encapsulated with the PREDICTREASON() function, accessible during data/model joins. The statement

SELECT …, PREDICTREASON([m].[pv], class) AS [reason] … FROM [m] PREDICTION JOIN (…)

returns a string in a column named '**reason**' that is meant to rationalize the model's behavior for its prediction of 'class', given it's input.

Predict reasons can be included or excluded from the model scoring results in the Scoring Fields page of the Scoring Wizard and Validation Wizard.

In general, there is no single cause for a prediction, and usually it's a number of correlated factors that are to blame (or celebrate). Presently, the reasons returned by PREDICTREASON() are limited to being a comma-separated list of variable names that contributed the most to the prediction.

Predict Reasons are only available for linear and logistic regression models.

1.5.1 Logistic Regression

Given some input vector x, the log-odds for predicting the probability of some class is

$$\log \frac{P_{class}}{1 - P_{class}} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n$$

where β_i are the model parameters associated with the class, and $x_1 \; x_2 \ldots x_n$ is a row of data. The contribution of x_k to the prediction is its product with the corresponding parameter, $\beta_k x_k$.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Whether this is a big or small contribution can be determined by comparing it to the observed mean contribution of the training data. The cases of a continuous predictor and a discrete predictor are similar, but distinct.

Continuous predictor

A continuous predictor is associated with exactly one model parameter. If we denote the observed mean of the above product as U_k, then we have

$$U_k = \frac{1}{N}\sum_{i=1}^{N} \beta_k X_{ik} = \beta_k \overline{X_k}$$

where X denotes the training data. The "reasons" for a continuous x_k are then defined to be the difference

$$R_k = \beta_k (x_k - \overline{X_k})$$

Discrete predictor

A discrete variable is associated with a number of model parameters – the so-called "dummy variables". The single product term in the continuous case is then generalized to a sum of *w* terms

$$\sum_{i=0}^{w-1} \beta_{k+i} x_{k+i}$$

The reasons follow suit

$$R_k = \sum_{i=0}^{w-1} \beta_{k+i} (x_{k+i} - \overline{X_{k+i}})$$

Listing

For all input variables, the values of R_k are gathered in descending order. The top Q variables that have positive values of R_k are returned by PREDICTREASON(). The default value of Q is 3. The value of Q can be modified in the third page of the Scoring Wizard or Validation Wizard. Note that the value of 0 means "all reasons". If you don't want to include predict reasons in your scoring or validation results, just uncheck them in the same wizard page.

1.5.2 Linear Regression

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

A linear model estimates some continuous value Y as

$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n$$

"Reasons" are extracted in the same way as for logistic regression (described above). However, the class name argument to PREDICTREASON() is not a particular value of Y, but client interest in whether or not Y is increasing or decreasing. The string "H" indicates interest in an increasing Y, and "L" for a decrease.

1.6 Variance Inflation Factors

The Variance Inflation Factors (VIF) show the extent to which collinear variables are affecting the model. The model results of each sequence display the VIF. To view them, right-click the predictive model node after it has been trained and select Open View from the context menu. Then choose the desired sequence in the "Output to View" drop-down list in the Results tab of your Predictive Model. Scroll down to the bottom to see the Variance Inflation Factors. The higher the VIF values are, the more unstable the model estimate is.

The VIF for the ith independent variable is defined as

$$VIF_i = \frac{1}{1 - R_i^2}$$

where R_i^2 is the unadjusted R^2 from the linear regression of the ith variable against all other variables. When the ith variable is not linearly dependent on the rest of the variables, its R^2 should be small, ideally zero, then VIF would be exactly 1.0, indicating no linear dependence. However, when there is "some" collinearity between the variable and the rest of the independent variables, its equivalent R^2 will have some value and VIF will be larger than unity. Therefore, from the multi-collinearity point of view, "healthy" sets of independent variables have VIF's close to 1.0. Usually the degree of multicollinearity is considered to be high if VIF is greater than 5.

Note that the Altair Data Mining Engine does not directly evaluate the VIF formula above because of inefficiency in the case of more than one variable. Instead, it computes them all "in parallel" with the following steps:

1. Compute the correlation matrix for all the independent variables in the model.

2. Invert this matrix.

At this point, the VIF for the j'th variable is the j'th diagonal element of this matrix.

Deep Learning (Multi-layer Neural Network)

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Deep Learning models based on Multi-layer Neural Networks is one of most popular models with a nonparametric algorithm. It can be viewed as an interconnected network made up of nodes as simple computational units. The nodes are arranged into one or more layers. The first layer is input layer and the final layer is output layer. Other layers between are called hidden layers. The output of a node in hidden layer is served as an input to the next layer. Each hidden node outputs the value from calculating a sigmoid function to a weighted sum function of its inputs.

Between each pair of adjoining layers, there are the weights used by this node to weigh the inputs to summing as output for this node and input for the node in next layer. The weights are free parameter to be trained by the data.

Before neural network training begins, the original data is normalized as follows.

Normalization of attributes

With selected attributes for both output and input variables, the original data is rescaled to new dimensionless attributes.

- If an attribute is continuous, the data matrix is standardized to convert original attribute scores to new attributes with mean of 0 and standard deviation of 1.
- If an attribute is ordered, the treatment of rescaling is the same as that for a continuous variable.
- If an attribute is qualitative, it is replaced by binary indicator variables.

Model Algorithm

Knowledge Studio has three activation functions to choose as a transfer function for output *y* of each node.

$Y = \frac{1-e^{-net}}{1+e^{-net}}$: Hyperbolic tangent function;

$Y = net$: Linear function;

$Y = \frac{1}{1+e^{-net}}$: Sigmoid function;

Where $net = w_0 + \sum_{i=1}^{N} w_i * x_i$

$i = 1, \ldots, N$: number of nodes from an input layer;

W_0 : bias of the node;

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

W_i : weight for the connection from *i*-th node of input layer;

X_i : input from the *i*-th node of input layer.

Back-propagation is one of many error-minimizing functions which tunes these weights to generate the desired smooth nonlinear mapping between input and output variables.

Compute squared error cost function:

$$E = \sum_{i=1}^{N} \left\{ \frac{1}{2} \sum_{j=1}^{J} (d_{ij} - y_{ij})^2 \right\};$$

i = 1,..., *N* : number of nodes from an input layer;

j = 1,..., *J* : number of nodes at an output layer;

d_{ij}: desired value of output j given by input node i;

y_{ij}: actual output value from the model.

Back-propagation often uses an optimization algorithm other than just a simple gradient descent. Such an algorithm tunes weights by means of gradient descent plus momentum. In Knowledge Studio, conjugate descent procedure is chosen as a form of gradient descent plus momentum, in which the parameters η and α are determined automatically at each iteration.

With a single hidden layer, assume *K*-1 is the input layer, *K* is the hidden layer and *K*+1 is the output layer. Suppose the following algorithm is a sigmoid function.

Compute sigmoid function for node *j* of layer *K*:

$$y_{jk} = \frac{1}{1 + e^{-net_{jk}}}$$

where $net_{jk} = w_{0j} + \sum_{i=1}^{N_{K-1}} w_{ij} * x_i$;

$i = 1,\ldots, N_{K-1}$ N_{K-1} nodes of layer *K*-1;

$j = 1,\ldots, N_K$ N_K nodes of layer *K*;

$l = 1,\ldots, N_{K+1}$ N_{K+1} nodes of layer *K*+1;

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

w_{0j}: Bias;

w_{ij} : Weight between node *i* of layer *K*-1 layer and node *j* of layer *K*;

x_i: Input from node *I* of layer *K*-1;

At output layer *K*+1:

for estimating purpose, use linear transfer function

$$y_{jk+1} = w_{0l} + \sum_{j=1}^{N_k} w_{jl} * x_j$$;

for classification purpose, use sigmoid non-linear function

$$y_{jk+1} = \frac{1}{1+e^{-net_{jk+1}}}$$;

Where $$net_{jk+1} = w_{0k+1} + \sum_{j=1}^{N_k} w_{jk+1} * x_j$$;

Compute average root-mean square error:

$$E_l = \frac{1}{2}\sum_{j=1}^{N_k} (d_{jl} - y_{jl})^2$$;

d_{jl}: desired value of output l given by input node *j* ;

y_{jl} : actual value of output.

Overall error function:

$$E = \sum_{l=1}^{N_{k+}} (E_l)$$.

The back-propagation procedure to train a neural net to minimize error function E_l at output layer *K*+1: pick learning rate η and momentum α by conjugate method and then adjust weight w_{ij} between node *j* and input node *i*.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

$$\Delta w_{ij} = -\eta \frac{\partial E_i}{\partial w_{ij}} + \alpha(\Delta w_{ij}^{'} - \Delta w_{ij}^{*})$$;

$\frac{\partial E_i}{\partial w_{ij}}$: partial derivative of error with respect to w_{ij} ;

η : step size of steepest gradient descent. It is the learning rate affecting how much weights are change on each update;

α : momentum term in order to have minimization “smoothed” over successive descents. I.e., to have fraction of previous steps to be included in current step.

Compute $\frac{\partial E_i}{\partial w_{ij}}$ from output layer down to input layer, using chain rule

$$\frac{\partial E_i}{\partial w_{ij}} = \frac{\partial net_{jk}}{\partial w_{ij}} * \frac{\partial y_{jk}}{\partial net_{jk}} * \frac{\partial E_i}{\partial y_{jk}}$$;

where $\frac{\partial net_{jk}}{\partial w_{ij}} = x_i$;

$\frac{\partial y_{jk}}{\partial net_{jk}} = y_{jk}(1 - y_{jk})$ for sigmoid function $Y = \frac{1}{1+e^{-net}}$;

If the linear function $Y = net$ is chosen, then

$$\frac{\partial y_{jk}}{\partial net_{jk}} = 1$$;

If the hyperbolic tangent function $Y = \frac{1-e^{-net}}{1+e^{-net}}$ is chosen, then

$$\frac{\partial y_{jk}}{\partial net_{jk}} = \frac{1}{2}(1 + y_{jk})(1 - y_{jk})$$;

$\frac{\partial E_i}{\partial y_{jk}} = d_{ji} - y_{ji}$ for output layer *K*+1;

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

$$\frac{\partial E_j}{\partial y_{ik-1}} = \sum_{j=1}^{N_k} \frac{\partial net_{jk}}{\partial y_{ik-1}} * \frac{\partial y_{jk}}{\partial net_{jk}} * \frac{\partial E_j}{\partial y_{jk}} = \sum_{j=1}^{N_k} w_{ij} * y_{jk}(1 - y_{jk}) * \frac{\partial E_j}{\partial y_{jk}} ;$$

for hidden layer *K*.

Then add up the weight changes for all sample inputs, and change weights.

Each cycle of back-propagation training is called an epoch. Once number of epochs or specific value for error reached, computer will stop training, weights generated from final epoch will be the final weights for neural network model.

Cluster Analysis Models

General Information

Cluster analysis is a technique for grouping individuals or objects into clusters so that the objects in the same cluster are more like each other than they are like the objects in other clusters. The resulting object clusters should then exhibit internal (within-cluster) homogeneity and high external (between-cluster) heterogeneity with respect to some predetermined selecting criteria.

Two clustering algorithms are presented: K-Means and Expectation-Maximization (EM). Both are nonhierarchical clustering procedures which don't involve tree-like construction processing. The K-Means similarity algorithm is based on distance. EM is based on the probabilistic theorem.

Execution of two preparation steps (before clustering) is common to both K-means and EM. The first step is to normalize the attributes so that values are on an even plain. The second step is to select the initial cluster seeds.

Normalization of Attributes

For selected attributes, original data is rescaled to new dimensionless attributes.

- If an attribute is a continuous variable, standardize data matrix to convert original attributes scores to new attributes with mean of 0 and standard deviation of 1.
- If an attribute is an ordered variable, the treatment of rescaling is the same as continuous variable.
- If attribute is a qualitative variable, it is replaced by binary indicator variables.

Selecting Initial Cluster Seeds

There are several options provided:

1. Randomly select *K* objects as initial cluster seeds simultaneously.

2. Use the following refinement algorithm to produce refined initial cluster seeds by an iterative algorithm to converge to a 'better' local minimum.

- Select J random sub-samples of full dataset, i=1,…,J;
- Randomly select K objects as initial cluster seeds for each sub-sample;
- Use K-means to generate cluster solutions for each sub-sample;

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- If there is an empty cluster at termination of K-means, re-assign initial seeds which is farthest from empty cluster and then re-cluster sub-sample through K-means;
- Now we got J sets of cluster solutions CM_i, i=1,…,J and form CM_i as a new dataset CM;
- Initialize cluster seeds from CM and use K-means to cluster CM_i again and produce a cluster solution FM_i;
- FM_i have minimal distortion over dataset CM and become a set of initial cluster seeds.

3. Use the KKZ initialization method (Katsavounidis, Kuo, and Zhang).

The KKZ method was originally described in I. Katsavounidis, C. Kuo, and Z. Zhang, "A new initialization technique for generalized Lloyd iteration". *IEEE Signal Processing Letters*, **1**(10):144–146, 1994.

The KKZ method belongs to the category of distance optimization methods. It utilizes the sorted pairwise distances for initialization. It is briefly described as follows.

Let x_j be the data vectors (*j = 1,...,N*).

- The first cluster seed is initialized using the input with the maximal norm, i.e., $c_1 = x_{J1} \equiv argmax\{||x_j||\}$.
- For *i* = 2,...,K, each c_i is initialized as follows: for each input sample x_j, calculate its distance to the closest seed cluster $d_j = min\{||x_j - c_k||\}$ (over all existing c_k), and set $c_i = x_{Ji} \equiv argmax\{d_j\}$.

The KKZ method does not require a threshold during seed vector search. KKZ could be computationally intensive due to the large number of distance calculations. However, since generation of new seed clusters does not affect the existing ones, the computational complexity of KKZ can be comparable to one K-Means iteration using a buffering mechanism that stores the distances between each input sample and the existing seed clusters.

KKZ method is deterministic, in the sense that the Cluster Analysis training results are repeatable if you specify the same parameters.

For comparative description of methods for initialization of clusters, see, for example:

He J., M. Lan, C.-L. Tan, S.-Y. Sung and H.-B. Low, "Initialization of clusters refinement algorithms: a review and comparative study", *International Joint Conference on Neural Networks*, IJCNN 2004, 25-29 July 2004, Budapest, Hungary.

M. Al-Daoud and S. Roberts. "New methods for the initialisation of clusters." *Technical Report* 94.34, School of Computer Studies, University of Leeds, 1994.

K-Means Algorithm Procedure

- Assign remaining objects to cluster seed each object is nearest.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

- Calculate average Euclidean distance coefficients:

$$E_{jk} = \sqrt{\sum_{i=1}^{I} (X_{ij} - M_{ik})^2}$$

i=1, …, I attributes;

j=1, …, N objects;

k=1, …, K cluster seeds;

X_{ij}: value of data for the *i* th attribute and *j* th object;

M_{ik}: cluster seed value for the *i* th attribute and *k*th cluster.

Assign object *j* to

$$\min_{k=1}^{K} \{E_{jk}\}$$

Cluster seed is recomputed to be the mean of all objects within cluster.

Calculate new cluster seeds

$$M_{ik} = \frac{1}{N_k} \sum_{j=1}^{N_k} x_{ij}$$

Reassign each object to its nearest new cluster seed.

- If there is no further change in the grouping of objects after a number of iterations, then the iterative process terminates.
- Calculate sum-of-squares clustering function:

$$E^2 = \sum_{k=1}^{K} \sum_{j=1}^{N_k} E_{jk}^2$$

- Stop iteration when the value of E^2 will not decrease.
- K final clusters are composed of objects surrounding K cluster seeds from the final iteration.

Expectation-Maximization Algorithm

EM estimates the probability density function (PDF) for each class, and then performs classification based on Bayes' rule:

$$P(j \mid x^n) = \frac{P(x^n \mid j) * P(j)}{P(x^n)}$$

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Posterior probability for class *j*: .

Where $P(x^n \mid j)$ is PDF for class *j*.

EM will estimate $P(x^n \mid j)$ as a weighted average of multiple Gaussians.

$$P(x^n \mid j) = \frac{1}{N_j} \sum_{l=1}^{N_j} G_l$$ is the estimate of PDF for class *j*.

$$G_l = \frac{1}{\sqrt[n]{2\pi}\sqrt{|\Sigma_l|}} e^{-\frac{1}{2}(x^n-\mu_l)^T \Sigma_l^{-1}(x^n-\mu_l)}$$

μ_l : mean of Gaussian.

Σ_l : covariance matrix.

$P(x^n)$ is overall PDF.

$$P(x^n) = \sum_{j=1}^{K} P(x^n \mid j) * P(j)$$,

and $P(j)$ is prior probability for class *j*.

$$P(j) = \frac{N_j}{N}$$.

Calculate likelihood function:

$$L_j = \prod_{n=1}^{N} P(x^n \mid j)$$.

Find a new set of parameter values to maximize the likelihood in an iterative fashion.

$$\mu_j^{new} = \frac{\sum_{n=1}^{N} P^{old}(j \mid x^n) * x^n}{\sum_{n=1}^{N} P^{old}(j \mid x^n)}$$.

$$(\sigma_j^{new})^2 = \frac{1}{d} \frac{\sum_{n=1}^{N} P^{old}(j \mid x^n) * (x^n - \mu_j^{new})^2}{\sum_{n=1}^{N} P^{old}(j \mid x^n)}$$.

$$P^{new}(j) = \frac{1}{N} \sum_{n=1}^{N} P^{old}(j \mid x^n)$$.

Then treat the ‘new’ parameter as ‘old’, and search for a new parameter again to maximize the likelihood L_j.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Repeat this iterative process until parameter values converge when the likelihood is maximized.

K cluster is formed with centroids of final K seeds of parameter values.

Assign each record to the cluster which has the maximum likelihood score.

Calculate the likelihood of that record for each distribution.

$$L_j(x^n) = P(x^n \mid j) * P(j)$$

Clustering Probability

The probability of each point, x^n, to the *j* th center seed is calculated by the formula

$$P(j \mid x^n) = \frac{P(x^n \mid j) * P(j)}{P(x^n)},$$

and the definition has been stated in the section "Expectation Maximization Algorithm".

Pseudo-F statistic

The Pseudo-F statistic, also known as the Calinski-Harabasz index, was first proposed in Calinski and Harabasz (1974) as a criterion for the best number of clusters:

$$\text{Pseudo F} = \frac{(S - P_K)/(K-1)}{P_K/(N-K)}$$

where *N* is the total number of observations, *K* is the number of clusters, *S* is the total sum of squares, and *PK* is the within-group sum of squares.

In the original notation (see Calinski and Harabasz (1974); Milligan and Cooper (1985))

$$CH(K) = \frac{B(K)/(K-1)}{W(K)/(N-K)}$$

where

$$B(K) = \sum_{k=1}^{K} n_k (u(k) - u)(u(k) - u)^T$$

u(*k*) is the centre of the *k*-th cluster,

u is the mean of entire dataset,

N is the total number of observations,

n_k is the number of observations in the *k*-th cluster

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

$$W(K) = \sum_{k=1}^{K} \sum_{i=1}^{n_k} (x(i) - u(k))(x(i) - u(k))^T$$

(The inner sum is taken over the observations in each cluster.)

B(*k*) is the "between the clusters" measure and *W*(*k*) is the "within the clusters" measure.

The Goodness of Clusters Metric

The goodness of clusters metric is simply a normalized Calinski-Harabasz pseudo F statistic (see above). It is equal to Pseudo F divided by (*N* - *K*), where *N* is the total number of observations and *K* is the number of clusters.

R-squared

R^2 is the proportion of variance accounted for by the clusters.

$$R^2 = 1 - \frac{\text{trace}(W)}{\text{trace}(T)}$$

where *T* is the total-sample sum-of-squares and cross products matrix ($T = X^T X$), and

$$W = (X - Z\overline{X})^T (X - Z\overline{X})$$

Z is the cluster indicator matrix (z_{ik}=1 if the *i* th observation belongs to the *k* th cluster), and $\overline{X}$ is the *K* by *I* matrix of cluster means (Sarle, 1983).

Cubic Clustering Criterion (CCC)

The Cubic Clustering Criterion (CCC) is a measure of the deviation of the clusters from the distribution expected for data sampled from a uniform distribution.

$$CCC = \ln\left[\frac{1 - E(R^2)}{1 - R^2}\right] \frac{\sqrt{\frac{Np^*}{2}}}{(0.001 + E(R^2))^{1.2}}$$

where *N* is the total number of observations, R^2 is the observed R-squared, $E(R^2)$ is the expected R-squared, p^* is the dimensionality of the between-cluster variation. The factor on the right is an empirically derived variance-stabilizing transformation (see Sarle, 1983). The larger the values of the *CCC,* the better the solution (compared to a uniform distribution). If the clustering variables are highly correlated, the value of the *CCC* may not reflect the quality of the solution.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Clustering Relevance

In each cluster, there are two distributions formed for each variable: one for “in cluster” and another for "not in cluster”.

These distributions are then compared with the Kulback-Liebler (K-L) distance, for which there are continuous and discrete versions with comparable meaning. Because we only have the mean and variance for the continuous variables, we assume these are normal distributions for the purpose of computing the KL distance.

The closer the distance (i.e., the smaller the K-L number), the less the variable has to do with the cluster.

To assess the degree of relevance of a variable to each cluster, the **relevance** parameter is provided in the Segment Viewer tab of Cluster Analysis (select **Tools | Options** and choose "**Show relevance**").

The value of **relevance** is the K-L divergence (also called *information divergence*, or *relative entropy*).

The Cluster Options dialog (opened via the **Tools | Options** menu command). The higher the value of Relevance, the more different the probability distributions are, which is the desired result.

See the References section for the relevant literature.

Principal Component Analysis

1. Introduction

The analysis of sets of data containing a large number of variables can fall victim to the so-called *curse of dimensionality*: as the number of variables increases, the sample size grows exponentially. As an example, if a sample of 10 equally spaced points on some one-dimensional interval was in some sense sufficient to model an effect, an equivalent sample of equally spaced points in only five dimensions would require 105 points – 10000 times larger. For tables with tens or hundreds or even thousands of columns, it is unlikely – with even trillions of rows of data – that the entire feature space will be sampled well enough to justify models with a corresponding number of variables. Furthermore, depending on the model, the analysis itself can be computationally intractable with respect to time and memory, and the internal structure of any models produced may be difficult to interpret. In these situations, culling the set of variables – *dimensionality reduction* – is a practical necessity.

Principal components analysis (PCA) is a pre-processing technique that constructs a new set of uncorrelated variables as linear combinations of the old variables, measures their variances, and allows the analyst to tune the selection of variables by quantifying the inevitable reduction in information prior to final modeling.

2. Transformation

Covariance

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Given an N x M matrix of data:

$$X = \begin{bmatrix} x_{11} & x_{12} & . & x_{1M} \\ x_{21} & x_{22} & . & x_{2M} \\ . & . & . & . \\ . & . & . & . \\ x_{N1} & x_{N2} & . & x_{NM} \end{bmatrix}$$

The sample covariance matrix for *X* is:

$$\Sigma = \begin{bmatrix} \sigma_{11}^2 & \sigma_{12}^2 & . & \sigma_{1M}^2 \\ \sigma_{21}^2 & \sigma_{22}^2 & . & \sigma_{2M}^2 \\ . & . & . & . \\ \sigma_{M1}^2 & \sigma_{M2}^2 & . & \sigma_{MM}^2 \end{bmatrix}$$

where

$$\mathrm{cov}(x_i, x_j) = \sigma_{ij}^2 = \frac{1}{N-1}\sum_{k=1}^{N}(x_{ki} - \mu_i)(x_{kj} - \mu_j)$$

is the covariance between variable x_i and x_j, and the mean of x_i is

$$\mu_i = \frac{1}{N}\sum_{k=1}^{N} x_{ki}$$

The PCA transformation decomposes the total variance in the data *X*

$$\sigma^2 = Tr(\Sigma) = \sum_{i=1}^{M} \sigma_{ii}^2$$

into *M* uncorrelated components by extracting the eigenvalues and eigenvectors of the covariance matrix. That is, an *M* x *M* rotation matrix *V* is found, such that:

$$V^{-1}\Sigma V = D$$

where *D* is a diagonal matrix of the form

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

$$D = \begin{bmatrix} \lambda_1 & 0 & . & 0 \\ 0 & \lambda_2 & . & 0 \\ . & . & . & . \\ 0 & 0 & . & \lambda_M \end{bmatrix}$$

$$\lambda_1 \geq \lambda_2 \geq \mathrm{K} \geq \lambda_M$$

The diagonal elements of *D* - its eigenvalues - are the variances of a new variable defined by projecting the original data along the corresponding eigenvector, which is just a column of the *V* matrix:

$$x_i' = \sum_{k=1}^{M} V_{ki} x_i$$

Because these new variables are uncorrelated, that is to say

$$\mathrm{cov}(x_i', x_j') = 0$$

when we select a set of new variables

$$x_1', x_2', \Lambda\ , x_P', \quad P \leq M$$

the variance captured by this selection of new variables can be immediately compared to the total observed variance by

$$\frac{selected\,(P)}{total} = \frac{\sum_{i=1}^{P} \lambda_i}{\sum_{i=1}^{M} \sigma_{ii}^2}$$

Though in practice, the analyst usually presents a target ratio, and the smallest value of *P* that satisfies

$$\frac{selected\,(P)}{total} \geq desired$$

is found.

Correlation

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

If the ratio

$$\frac{\max \sigma_{ii}^2}{\min \sigma_{jj}^2}$$

is large, a PCA can be biased towards selecting fewer variables, even if the small-scale variables contain more information. For example, suppose, given some table, a PCA identifies 4 important components. If you augment this table with a column containing a random number with a very large variance – enough to drive the scale ratio to 1000 or even larger - re-running the PCA will likely produce only 1 component, that component being dominated by the random data.

In this situation, it is preferable to scale the data so all columns have the same variance. Or, equivalently, to run the PCA described above against the correlation matrix, not the covariance matrix. That is, replace the covariance matrix with

$$R = \begin{bmatrix} \rho_{11} & \rho_{12} & \cdot & \rho_{1M} \\ \rho_{21} & \rho_{22} & \cdot & \rho_{2M} \\ \cdot & \cdot & \cdot & \cdot \\ \rho_{M1} & \rho_{M2} & \cdot & \rho_{MM} \end{bmatrix}$$

where each element of the matrix is the pair-wise sample correlation coefficient

$$\rho_{ij} = \frac{\sigma_{ij}^2}{\sqrt{\sigma_{ii}^2}\sqrt{\sigma_{jj}^2}}$$

When using the correlation matrix as an input to the PCA, component selection is slightly modified. Since the denominator in the preceding formula has divided out the observed variance, the selected variance ratio no longer has the meaning it did (described above). Instead, when subjecting a correlation matrix to PCA, the number of components is generally selected by the Kaiser criteria: choose P so that all of the eigenvalues are larger than 1.

3. Score Equations

After the components have been selected, the entire table is *scored* against them, producing the new variables. In the standardized form, this scoring process can be written as

$$x' = WS(x - \mu) = \mathrm{cov}(Kx)^{-1} KS(x - \mu)$$

where *x* is the original row of data (but here represented by a *column* vector), μ is the mean of data. The matrix S depends on the source of information for the PCA. If a covariance matrix was analyzed, then S is an identity matrix of dimension *M*:

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

For a correlation matrix, S is a diagonal matrix constructed from the square root of the diagonal elements of the covariance matrix Σ.

That is,

$$S = \begin{bmatrix} \sqrt{\sigma_{11}^2} & 0 & . & 0 \\ 0 & \sqrt{\sigma_{22}^2} & . & 0 \\ . & . & . & . \\ 0 & 0 & . & \sqrt{\sigma_{MM}^2} \end{bmatrix}$$

The matrix *K* – called the *rotated loading matrix* – is further decomposed into

$$K = (L\sqrt{\lambda}R)'$$

Here, the *M* x *P* matrix *L* is formed from the eigenvectors selected from the PCA, and the matrix λ is diagonal, consisting of the corresponding eigenvalues. The product $L\sqrt{\lambda}$ is called the *unrotated loading matrix* and is (optionally) subjected to a VARIMAX rotation, with Kaiser normalization. This produces the rotation matrix *R*. If VARIMAX is not applied to the loading matrix, *R* will be an identity matrix of dimension *P*, the number of selected components.

References

Al-Daoud, M. and Roberts, S. (1994). New methods for the initialisation of clusters. *Technical Report* 94.34, School of Computer Studies, University of Leeds.

Breiman, L., Friedman, J., Stone, C. J., and Olshen R. A. (1984). *Classification and Regression Trees*. Chapman & Hall/CRC.

Biggs, D., de Ville, B., and Suen, E. (1991). A method of choosing multiway partitions for classification and decision trees. *Journal of Applied Statistics*, **18**, 49-62.

Calinski, T. and Harabasz, J. (1974). A dendrite method for cluster analysis. *Communications in Statistics*, **3**, 1-27.

Dunn, O. J. (1961). Multiple comparisons among means. *J. Amer. Statist. Assoc*., **56**, 54-64.

Efron, B. (1983). Estimating the error rate of a prediction rule. *J. Amer. Statist. Assoc*., **78**, 316-333.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Feller, W. (1968). *An Introduction to Probability Theory and its Applications*. J. Wiley & Sons, New York.

Hawkins, D. M. and Kass, G. V. (1982). Automatic interaction detection. In: *Topics in Applied Multivariate Analysis,* Cambridge University Press.

He, J., Lan, M., Tan, C.-L., Sung, S.-Y, and Low, H.-B. (2004). Initialization of clusters refinement algorithms: a review and comparative study, *International Joint Conference on Neural Networks*, IJCNN 2004, 25-29 July 2004, Budapest, Hungary.

Kass, G. V. (1980). An exploratory technique for investigating large quantities of categorical data. *Applied Statistics*, **29**, 119-127.

Kass, G. V. (1976). Significance testing in automatic interaction detection. *Applied Statistics*, **24**(2), 178-189.

Katsavounidis, I., Kuo, C., and Zhang, Z. (1994). A new initialization technique for generalized Lloyd iteration. *IEEE Signal Processing Letters*, **1**(10), 144–146.

Lundberg, S. M. and Lee, S. (2017). A Unified Approach to Interpreting Model Predictions. *Advances in Neural Information Processing Systems 30* (*NIPS 2017*).

Lattin, J. M., Carroll, J. D., and Green, P. E. (2003). *Analyzing Multivariate Data*. Brooks/Cole.

Michie, D. (1991). Methodologies from machine learning in data analysis and software. *The Computer Journal*, **34**(6), 559-565.

Miller, R. G. (1981). *Simultaneous Statistical Inference.* Springer, New York.

Milligan, G. and Cooper, M. (1985). An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, **50**(2), 159-179.

Quinlan, R. (1979). Discovering rules by induction from large collections of examples. In: *Expert Systems in the Micro-Electronic Age*, 168-201, Edinburgh.

Refaat, M. (2011). *Credit Risk Scorecards: Development and Implementation Using SAS*. Mamdouh Refaat, ISBN 978-1-4475-1119-9.

Ritschard, G. (2013). CHAID and Earlier Supervised Tree Methods. In J.J. McArdle & G. Ritschard (eds), *Contemporary Issues in Exploratory Data Mining in Behavioral Sciences*, Routeledge, New York, 48-74.

Sarle, W. S. (1983). *Cubic Clustering Criterion*. SAS Technical Report A-108, SAS Institute Inc., Cary, NC.

Sonquist, J. A., Baker, E. and Morgan, J. (1973). *Searching For Structure*. Institute for Social Research, University of Michigan, Ann Arbor, MI.

Siddiqi, N., (2005). *Credit Risk Scorecards: Developing and Implementing Intelligent Credit Scoring*, John Wiley & Sons.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Shapiro, S. S. and Wilk, M. B. (1965). An analysis of variance test for normality (complete samples). *Biometrika* **52** (3-4), 591–611. JSTOR 2333709. MR 205384.

de Ville, B. (1990). Applying statistical knowledge to database analysis and knowledge base construction. In: *Proceedings of the 6th IEEE Conference on Artificial Intelligence Applications*, IEEE Computer Society, Washington.

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Index

A

accuracy 640

Aggregating data 187

amalgamation paradox 497

Append 191

ARIMA 1037, 1190, 1393, 1404

Association rules 973

Attribute editor

  attribute editor dialog 625, 914

  break-apart 625, 914

  cardinality 625, 914

  grouping type 633

  include 625, 914

  intervals and bins 634

  missing values 625, 914

  ordered display 625, 914

  significance 625, 914

  usage 625, 914

  weights 631

Auto Feature Engineering 252, 1376

Automated Machine Learning 833

Automatic Grow 548

AutoML 833

  Copy to Another Workflow 852

C

Charts

  profile chart 604

Class imbalance 392

Client-server mode 54

Cluster analysis

  about 954

  inserting a cluster model 955

Code generation 1207

  C 1207

  English 1207

  Generic pseudocode 1207

  Java 1207

  PMML 1207

  Python 1207

  R 1207

  SAS Language 1207

  SPSS 1207

  SQL 1207

  XML 1207

Connection 54

constraint 894

Conversion 15

Converter 1285

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Copy
data 624
screen 625
splits 617
tree 621
Correlations 446
binning 447
confidence interval 447
matrix 447
multicollinearity 489
pairs 447
Covariance 446
Cross tabs 433
Cumulative lift chart 1122
about 1121, 1122
calculating the cumulative lift 1122
Custom Library 1339
D
Data export
Monarch 178
ODBC export 170
Data Preparation
aggregation 187
Auto Feature Engineering 252, 1376
Filter 201
Filtering Join 211
imbalanced classes 392
lag 246
pivot 242
rank 246
sort 246
Data profiling 414
multicollinearity analysis 489
Dataset
about 414
append 191
Autocorrelation tab 512
characteristic analysis 442
chart 424
correlations 446
covariance 446
cross tabs 433
data 427
editor 261
filter
interactive dockable view 459
join 214
overview report 417
segment viewer 428
time series tab 505
Dataset Editor 261
Decision tree

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

about 521
Automatic Grow 548
inserting 524
measures 535
Deduplication 198
Deployment
SmartWorks Analytics 1239
E
Edit split 562
Email options 1538
Ensemble Trees 648, 654, 660
Bagging 648
Boosting 660
Random Forest 654
Environment 1537
Erase splits 561
Excel
multi-sheet workbook 112, 148
Explainable AI 1101
Decision Tree 1101
ICE Plot 1101
LIME Plot 1101
Partial Dependence Plot 1101
SHAP Plot 1101
Expression editor
function syntax 282
helpers (dataset) 327
binning 330
date and time join 333
date and time split 338
logarithm transform 355
outliers clipper 350
power transform 355
substitute missing values 343
text join 356
text split 361
weight 351
helpers (strategy tree) 879
Expressions
export 387
import 385
syntax 282
F
Factor Analysis 1004
Loadings 1007
Scores 1008
Feature Engineering 833
Field Finder 1502
Filter 201
Filtering Join 211
Find split 542
Folder import 126

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

Force split 550

Forecasting 1027, 1352

ARIMA forecasting 1037, 1393

Keras forecasting 1027, 1352

Format

Reports 1514

G

Generalized Linear Model 686, 1385, 1408

GLM 686, 1385, 1408

Global variables 1258

Go to split 547

GPU 733, 1027, 1344, 1352

H

Hyperparameter Tuning 833

I

ICE plot 1101

Ignore variable 560

Imbalanced classes 392

Import 91

folder 126

from Monarch DPS 134

importing through ODBC 138

in-database analytics 161

Infinity 275

Information Value 1577

Integration

R & Python environment 1537

SmartWorks Analytics 1239

Intervals Type 625, 914

J

Join 214

Jupyter notebook 1321

K

Keras 733, 1027, 1344, 1352

Deep Learning 733, 1344

forecasting 1027, 1352

K-S chart 1129

about 1129

interpreting the K-S chart 1130

K-S Statistic 1129

L

Lagged variables 246

License

acquire by email 50

trial license 25

lift report 1128

LIME 1101

M

Market Basket Analysis 973

Mathematical Formulations

Information Value 1577

Measure 535

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

misclassification rate 640

Missing values 388

substitution 388

Model

AutoML 833

cluster analysis 955

constrained regression 708

deep learning 723

linear regression 673

logistic regression 694

market basket analysis 973

model instance 518

neural network 723

Partial Least Squares Regression 715

PLS regression 715

regularization 711

stacking 809

upload to SmartWorks Analytics 1239

Model analyzer

about 1118

about the K-S chart 1129

goodness of fit 1134

inserting a model analyzer 1119

profit curve 1138

roc curve 1131

Model Selector 805, 1139

Monarch 178

Multicollinearity 489

N

Next Split 546

Node

data 594

dataset 638

report 598

Node calculation

about 872

expression editor 874

insert 873

Node dataset

about 638

inserting 638

notifications 1538

Novelty and Outlier Detector 822, 1430

O

Optimization 894, 944

dataset-level 1258

global variables 1258

scenario 1262

solution constraint 894

Ordered Display 625, 914

Outliers 822, 1430

Overview Report 417, 423, 1516

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

P

Partial Dependence Plot 1101

Partition

about 229

insert 229

Paste

splits 617

PDP plot 1101

Predictive model

about 666

AutoML 833

cluster analysis 954

comparison 805, 1139

generalized linear model 686, 1385, 1408

GLM 686, 1385, 1408

model instance 518

Sklearn XGBoost 776, 1446

stacking 809

XGB 776, 1446

Preferences

environment 1537

export 1509

import 1509

R and Python environment 1537

Previous Split 546

Principal Component Analysis

about 990

calculations from PCA 996, 1009

Components 995

eigenvectors 994

Scree Plot 994

Profile chart 604

Project Documentation

desktop documents 181, 1540

Projects

about 59

creating a project 62

documents 79

Python 1321

Custom Library 1339

Deep Learning 733, 1344

environment 1537

Jupyter notebook 1321

Keras forecasting 1027, 1352

Keras scoring 741, 1164, 1363

R

R

environment 1537

Ranking 246

Reject inference 754

hard cutoff 754

parcelling 754

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

relative error 640

resubstitution 640

reversal paradox 497

S

SAS Language

Code node 1288

Data converter 1285

SAS System File Link 100, 1280

WPD File Link 98, 1282

SAS Language Code 1288

Score

ARIMA scoring 1190, 1404

Keras scoring 741, 1164, 1363

model 1144

Novelty & Outlier 1158, 1440

Sklearn GLM scoring 1177, 1423

strategy 935

XGB model 1184, 1457

SHAP plot 1101

Shapley Additive Explanations 1101

Simpson's Paradox 497

Sklearn GLM 686, 1385, 1408

SmartWorks Analytics 1239

Sorting 246

Splits

about 522

copy 617

edit 562

erase 561

find 542

force 550

go to 547

ignore 560

next 546

paste 617

previous 546

report 596

Standalone mode 54

Strategy tree

about 855

inserting 857

score 935

treatment 896

validate 925

validation tree 931

Substitute missing values 388

T

Technical information

copyright 19

system requirements 16

Technical support 18

Time Series 223

Altair Engineering, Inc. Copyright © 2022. All rights reserved.

ARIMA 1037, 1393
Time Series Decomposition 1017
forecast plots 1023
Treatment 896
Tree views
node data 594
node report 598
profile chart 604
split report 596
tree map 593
tree view 578
Trees
strategy trees 855
U
Usage 625, 914
V
Validate
GLM model 1070, 1416
Keras Deep Learning model 748, 1063, 1369
model 1049
Sklearn GLM model 1070, 1416
strategy 925
XGBoost model 1077, 1463
Validation
Cross Validation 1087
Keras Deep Learning 748, 1063, 1369
XGBoost model 1077, 1463
Variable list 1493
export 1506
import 1502
W
Weight 415, 625, 670, 914, 965
frequency 631
sample 631
Workflow 62
working directory 56, 59
WPD File 98, 1282
WPS
Data converter 1285
X
XAI 1101
Y
Yule–Simpson effect 497

Altair Engineering, Inc. Copyright © 2022. All rights reserved.