IBM SPSS Modeler 16 Applications Guide

Note

Before using this information and the product it supports, read the information in "Notices" on page 335.

Product Information

This edition applies to version 16, release 0, modification 0 of IBM(r) SPSS(r) Modeler and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Chapter 1. About IBM SPSS Modeler

IBM® SPSS® Modeler is a set of data mining tools that enable you to quickly develop predictive models using business expertise and deploy them into business operations to improve decision making. Designed around the industry-standard CRISP-DM model, IBM SPSS Modeler supports the entire data mining process, from data to better business results.

IBM SPSS Modeler offers a variety of modeling methods taken from machine learning, artificial intelligence, and statistics. The methods available on the Modeling palette allow you to derive new information from your data and to develop predictive models. Each method has certain strengths and is best suited for particular types of problems.

SPSS Modeler can be purchased as a standalone product, or used as a client in combination with SPSS Modeler Server. A number of additional options are also available, as summarized in the following sections. For more information, see http://www.ibm.com/software/analytics/spss/products/modeler/.

IBM SPSS Modeler Products

The IBM SPSS Modeler family of products and associated software comprises the following.

- IBM SPSS Modeler
- IBM SPSS Modeler Server
- IBM SPSS Modeler Administration Console
- IBM SPSS Modeler Batch
- IBM SPSS Modeler Solution Publisher
- IBM SPSS Modeler Server adapters for IBM SPSS Collaboration and Deployment Services

IBM SPSS Modeler

SPSS Modeler is a functionally complete version of the product that you install and run on your personal computer. You can run SPSS Modeler in local mode as a standalone product, or use it in distributed mode along with IBM SPSS Modeler Server for improved performance on large data sets.

With SPSS Modeler, you can build accurate predictive models quickly and intuitively, without programming. Using the unique visual interface, you can easily visualize the data mining process. With the support of the advanced analytics embedded in the product, you can discover previously hidden patterns and trends in your data. You can model outcomes and understand the factors that influence them, enabling you to take advantage of business opportunities and mitigate risks.

SPSS Modeler is available in two editions: SPSS Modeler Professional and SPSS Modeler Premium. See the topic “IBM SPSS Modeler Editions” on page 2 for more information.

IBM SPSS Modeler Server

SPSS Modeler uses a client/server architecture to distribute requests for resource-intensive operations to powerful server software, resulting in faster performance on larger data sets.

SPSS Modeler Server is a separately-licensed product that runs continually in distributed analysis mode on a server host in conjunction with one or more IBM SPSS Modeler installations. In this way, SPSS Modeler Server provides superior performance on large data sets because memory-intensive operations can be done on the server without downloading data to the client computer. IBM SPSS Modeler Server also provides support for SQL optimization and in-database modeling capabilities, delivering further benefits in performance and automation.

IBM SPSS Modeler Administration Console

The Modeler Administration Console is a graphical application for managing many of the SPSS Modeler Server configuration options, which are also configurable by means of an options file. The application provides a console user interface to monitor and configure your SPSS Modeler Server installations, and is available free-of-charge to current SPSS Modeler Server customers. The application can be installed only on Windows computers; however, it can administer a server installed on any supported platform.

IBM SPSS Modeler Batch

While data mining is usually an interactive process, it is also possible to run SPSS Modeler from a command line, without the need for the graphical user interface. For example, you might have long-running or repetitive tasks that you want to perform with no user intervention. SPSS Modeler Batch is a special version of the product that provides support for the complete analytical capabilities of SPSS Modeler without access to the regular user interface. SPSS Modeler Server is required to use SPSS Modeler Batch.

IBM SPSS Modeler Solution Publisher

SPSS Modeler Solution Publisher is a tool that enables you to create a packaged version of an SPSS Modeler stream that can be run by an external runtime engine or embedded in an external application. In this way, you can publish and deploy complete SPSS Modeler streams for use in environments that do not have SPSS Modeler installed. SPSS Modeler Solution Publisher is distributed as part of the IBM SPSS Collaboration and Deployment Services - Scoring service, for which a separate license is required. With this license, you receive SPSS Modeler Solution Publisher Runtime, which enables you to execute the published streams.

IBM SPSS Modeler Server Adapters for IBM SPSS Collaboration and Deployment Services

A number of adapters for IBM SPSS Collaboration and Deployment Services are available that enable SPSS Modeler and SPSS Modeler Server to interact with an IBM SPSS Collaboration and Deployment Services repository. In this way, an SPSS Modeler stream deployed to the repository can be shared by multiple users, or accessed from the thin-client application IBM SPSS Modeler Advantage. You install the adapter on the system that hosts the repository.

IBM SPSS Modeler Editions

SPSS Modeler is available in the following editions.

SPSS Modeler Professional

SPSS Modeler Professional provides all the tools you need to work with most types of structured data, such as behaviors and interactions tracked in CRM systems, demographics, purchasing behavior and sales data.

SPSS Modeler Premium

SPSS Modeler Premium is a separately-licensed product that extends SPSS Modeler Professional to work with specialized data such as that used for entity analytics or social networking, and with unstructured text data. SPSS Modeler Premium comprises the following components.

IBM SPSS Modeler Entity Analytics adds an extra dimension to IBM SPSS Modeler predictive analytics. Whereas predictive analytics attempts to predict future behavior from past data, entity analytics focuses on improving the coherence and consistency of current data by resolving identity conflicts within the records themselves. An identity can be that of an individual, an organization, an object, or any other entity for which ambiguity might exist. Identity resolution can be vital in a number of fields, including customer relationship management, fraud detection, anti-money laundering, and national and international security.

IBM SPSS Modeler Social Network Analysis transforms information about relationships into fields that characterize the social behavior of individuals and groups. Using data describing the relationships underlying social networks, IBM SPSS Modeler Social Network Analysis identifies social leaders who influence the behavior of others in the network. In addition, you can determine which people are most affected by other network participants. By combining these results with other measures, you can create comprehensive profiles of individuals on which to base your predictive models. Models that include this social information will perform better than models that do not.

IBM SPSS Modeler Text Analytics uses advanced linguistic technologies and Natural Language Processing (NLP) to rapidly process a large variety of unstructured text data, extract and organize the key concepts, and group these concepts into categories. Extracted concepts and categories can be combined with existing structured data, such as demographics, and applied to modeling using the full suite of IBM SPSS Modeler data mining tools to yield better and more focused decisions.

IBM SPSS Modeler Documentation

Documentation in online help format is available from the Help menu of SPSS Modeler. This includes documentation for SPSS Modeler, SPSS Modeler Server, and SPSS Modeler Solution Publisher, as well as the Applications Guide and other supporting materials.

Complete documentation for each product (including installation instructions) is available in PDF format under the *\Documentation* folder on each product DVD. Installation documents can also be downloaded from the web at http://www-01.ibm.com/support/docview.wss?uid=swg27038316.

Documentation in both formats is also available from the SPSS Modeler Information Center at http://publib.boulder.ibm.com/infocenter/spssmodl/v16r0m0/.

SPSS Modeler Professional Documentation

The SPSS Modeler Professional documentation suite (excluding installation instructions) is as follows.

- **IBM SPSS Modeler User's Guide.** General introduction to using SPSS Modeler, including how to build data streams, handle missing values, build CLEM expressions, work with projects and reports, and package streams for deployment to IBM SPSS Collaboration and Deployment Services, Predictive Applications, or IBM SPSS Modeler Advantage.
- **IBM SPSS Modeler Source, Process, and Output Nodes.** Descriptions of all the nodes used to read, process, and output data in different formats. Effectively this means all nodes other than modeling nodes.
- **IBM SPSS Modeler Modeling Nodes.** Descriptions of all the nodes used to create data mining models. IBM SPSS Modeler offers a variety of modeling methods taken from machine learning, artificial intelligence, and statistics.
- **IBM SPSS Modeler Algorithms Guide.** Descriptions of the mathematical foundations of the modeling methods used in IBM SPSS Modeler. This guide is available in PDF format only.
- **IBM SPSS Modeler Applications Guide.** The examples in this guide provide brief, targeted introductions to specific modeling methods and techniques. An online version of this guide is also available from the Help menu. See the topic “Application Examples” on page 4 for more information.
- **IBM SPSS Modeler Python Scripting and Automation.** Information on automating the system through Python scripting, including the properties that can be used to manipulate nodes and streams.
- **IBM SPSS Modeler Deployment Guide.** Information on running IBM SPSS Modeler streams and scenarios as steps in processing jobs under IBM SPSS Collaboration and Deployment Services Deployment Manager.

- **IBM SPSS Modeler CLEF Developer's Guide.** CLEF provides the ability to integrate third-party programs such as data processing routines or modeling algorithms as nodes in IBM SPSS Modeler.
- **IBM SPSS Modeler In-Database Mining Guide.** Information on how to use the power of your database to improve performance and extend the range of analytical capabilities through third-party algorithms.
- **IBM SPSS Modeler Server Administration and Performance Guide.** Information on how to configure and administer IBM SPSS Modeler Server.
- **IBM SPSS Modeler Administration Console User Guide.** Information on installing and using the console user interface for monitoring and configuring IBM SPSS Modeler Server. The console is implemented as a plug-in to the Deployment Manager application.
- **IBM SPSS Modeler CRISP-DM Guide.** Step-by-step guide to using the CRISP-DM methodology for data mining with SPSS Modeler.
- **IBM SPSS Modeler Batch User's Guide.** Complete guide to using IBM SPSS Modeler in batch mode, including details of batch mode execution and command-line arguments. This guide is available in PDF format only.

SPSS Modeler Premium Documentation

The SPSS Modeler Premium documentation suite (excluding installation instructions) is as follows.

- **IBM SPSS Modeler Entity Analytics User Guide.** Information on using entity analytics with SPSS Modeler, covering repository installation and configuration, entity analytics nodes, and administrative tasks.
- **IBM SPSS Modeler Social Network Analysis User Guide.** A guide to performing social network analysis with SPSS Modeler, including group analysis and diffusion analysis.
- **SPSS Modeler Text Analytics User's Guide.** Information on using text analytics with SPSS Modeler, covering the text mining nodes, interactive workbench, templates, and other resources.

Application Examples

While the data mining tools in SPSS Modeler can help solve a wide variety of business and organizational problems, the application examples provide brief, targeted introductions to specific modeling methods and techniques. The data sets used here are much smaller than the enormous data stores managed by some data miners, but the concepts and methods involved should be scalable to real-world applications.

You can access the examples by clicking **Application Examples** on the Help menu in SPSS Modeler. The data files and sample streams are installed in the *Demos* folder under the product installation directory. See the topic "Demos Folder" for more information.

Database modeling examples. See the examples in the *IBM SPSS Modeler In-Database Mining Guide*.

Scripting examples. See the examples in the *IBM SPSS Modeler Scripting and Automation Guide*.

Demos Folder

The data files and sample streams used with the application examples are installed in the *Demos* folder under the product installation directory. This folder can also be accessed from the IBM SPSS Modeler program group on the Windows Start menu, or by clicking *Demos* on the list of recent directories in the File Open dialog box.

Chapter 2. IBM SPSS Modeler Overview

Getting Started

As a data mining application, IBM SPSS Modeler offers a strategic approach to finding useful relationships in large data sets. In contrast to more traditional statistical methods, you do not necessarily need to know what you are looking for when you start. You can explore your data, fitting different models and investigating different relationships, until you find useful information.

Starting IBM SPSS Modeler

To start the application, click:

Start > **[All] Programs** > **IBM SPSS Modeler 16** > **IBM SPSS Modeler 16**

The main window is displayed after a few seconds.

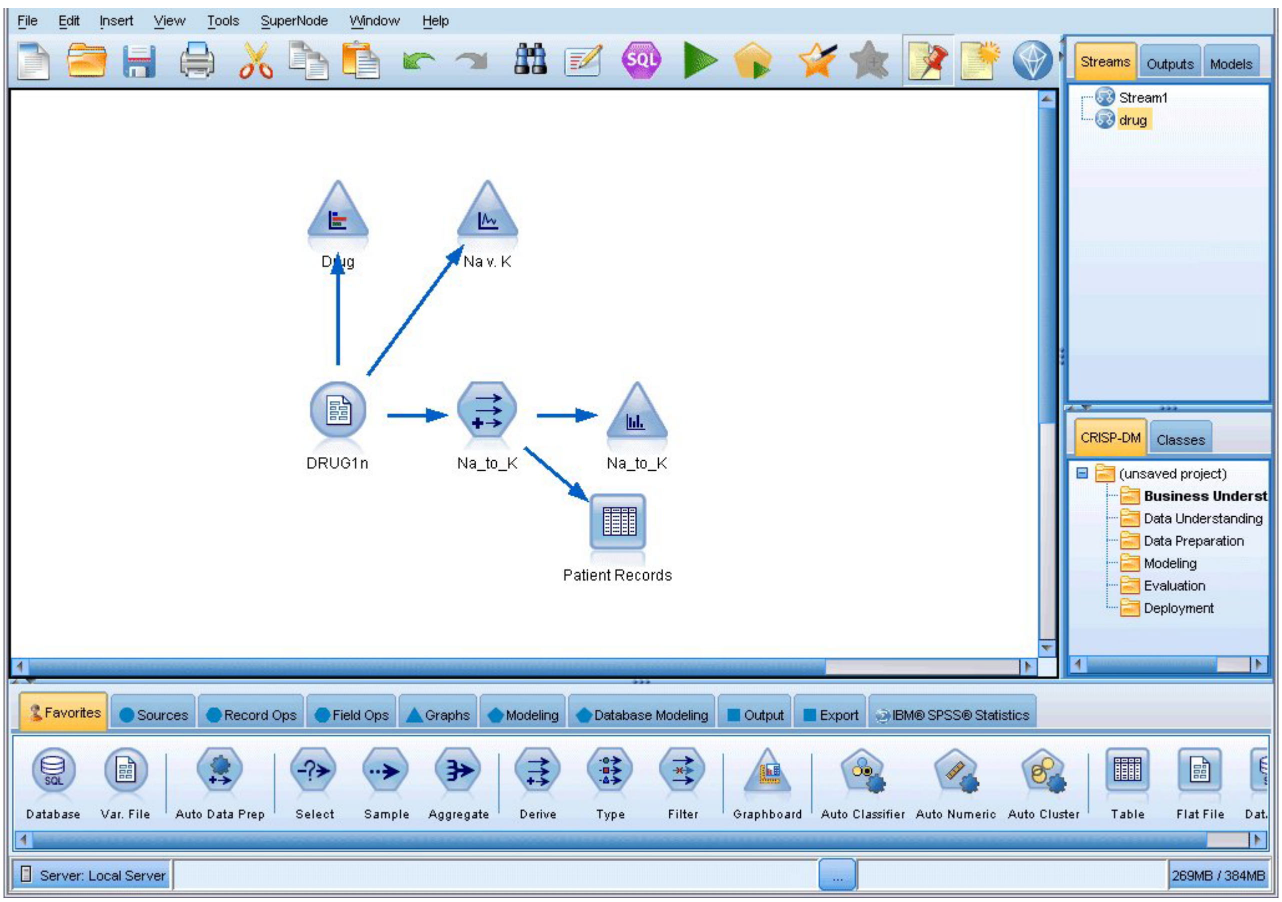


Figure 1. IBM SPSS Modeler main application window

Launching from the Command Line

You can use the command line of your operating system to launch IBM SPSS Modeler as follows:

1. On a computer where IBM SPSS Modeler is installed, open a DOS, or command-prompt, window.

2. To launch the IBM SPSS Modeler interface in interactive mode, type the `modelerclient` command followed by the required arguments; for example:

```
modelerclient -stream report.str -execute
```

The available arguments (flags) allow you to connect to a server, load streams, run scripts, or specify other parameters as needed.

Connecting to IBM SPSS Modeler Server

IBM SPSS Modeler can be run as a standalone application, or as a client connected to IBM SPSS Modeler Server directly or to an IBM SPSS Modeler Server or server cluster through the Coordinator of Processes plug-in from IBM SPSS Collaboration and Deployment Services. The current connection status is displayed at the bottom left of the IBM SPSS Modeler window.

Whenever you want to connect to a server, you can manually enter the server name to which you want to connect or select a name that you have previously defined. However, if you have IBM SPSS Collaboration and Deployment Services, you can search through a list of servers or server clusters from the Server Login dialog box. The ability to browse through the Statistics services running on a network is made available through the Coordinator of Processes.

To Connect to a Server

1. On the Tools menu, click **Server Login**. The Server Login dialog box opens. Alternatively, double-click the connection status area of the IBM SPSS Modeler window.
2. Using the dialog box, specify options to connect to the local server computer or select a connection from the table.
   - Click **Add** or **Edit** to add or edit a connection. See the topic "Adding and Editing the IBM SPSS Modeler Server Connection" on page 7 for more information.
   - Click **Search** to access a server or server cluster in the Coordinator of Processes. See the topic "Searching for Servers in IBM SPSS Collaboration and Deployment Services" on page 7 for more information.

   Server table. This table contains the set of defined server connections. The table displays the default connection, server name, description, and port number. You can manually add a new connection, as well as select or search for an existing connection. To set a particular server as the default connection, select the check box in the Default column in the table for the connection.

   Default data path. Specify a path used for data on the server computer. Click the ellipsis button **(...)** to browse to the required location.

   Set Credentials. Leave this box unchecked to enable the **single sign-on** feature, which attempts to log you in to the server using your local computer username and password details. If single sign-on is not possible, or if you check this box to disable single sign-on (for example, to log in to an administrator account), the following fields are enabled for you to enter your credentials.

   User ID. Enter the user name with which to log on to the server.

   Password. Enter the password associated with the specified user name.

   Domain. Specify the domain used to log on to the server. A domain name is required only when the server computer is in a different Windows domain than the client computer.
3. Click **OK** to complete the connection.

To Disconnect from a Server

1. On the Tools menu, click **Server Login**. The Server Login dialog box opens. Alternatively, double-click the connection status area of the IBM SPSS Modeler window.
2. In the dialog box, select the Local Server and click **OK**.

Adding and Editing the IBM SPSS Modeler Server Connection

You can manually edit or add a server connection in the Server Login dialog box. By clicking Add, you can access an empty Add/Edit Server dialog box in which you can enter server connection details. By selecting an existing connection and clicking Edit in the Server Login dialog box, the Add/Edit Server dialog box opens with the details for that connection so that you can make any changes.

Note: You cannot edit a server connection that was added from IBM SPSS Collaboration and Deployment Services, since the name, port, and other details are defined in IBM SPSS Collaboration and Deployment Services. Best practice dictates that the same ports should be used to communicate with both IBM SPSS Collaboration and Deployment Services and SPSS Modeler Client. These can be set as `max_server_port` and `min_server_port` in the options.cfg file.

To Add Server Connections

1. On the Tools menu, click **Server Login**. The Server Login dialog box opens.
2. In this dialog box, click **Add**. The Server Login Add/Edit Server dialog box opens.
3. Enter the server connection details and click **OK** to save the connection and return to the Server Login dialog box.

- **Server.** Specify an available server or select one from the list. The server computer can be identified by an alphanumeric name (for example, *myserver*) or an IP address assigned to the server computer (for example, 202.123.456.78).
- **Port.** Give the port number on which the server is listening. If the default does not work, ask your system administrator for the correct port number.
- **Description.** Enter an optional description for this server connection.
- **Ensure secure connection (use SSL).** Specifies whether an SSL (**Secure Sockets Layer**) connection should be used. SSL is a commonly used protocol for securing data sent over a network. To use this feature, SSL must be enabled on the server hosting IBM SPSS Modeler Server. If necessary, contact your local administrator for details.

To Edit Server Connections

1. On the Tools menu, click **Server Login**. The Server Login dialog box opens.
2. In this dialog box, select the connection you want to edit and then click **Edit**. The Server Login Add/Edit Server dialog box opens.
3. Change the server connection details and click **OK** to save the changes and return to the Server Login dialog box.

Searching for Servers in IBM SPSS Collaboration and Deployment Services

Instead of entering a server connection manually, you can select a server or server cluster available on the network through the Coordinator of Processes, available in IBM SPSS Collaboration and Deployment Services. A server cluster is a group of servers from which the Coordinator of Processes determines the server best suited to respond to a processing request.

Although you can manually add servers in the Server Login dialog box, searching for available servers lets you connect to servers without requiring that you know the correct server name and port number. This information is automatically provided. However, you still need the correct logon information, such as username, domain, and password.

Note: If you do not have access to the Coordinator of Processes capability, you can still manually enter the server name to which you want to connect or select a name that you have previously defined. See the topic “Adding and Editing the IBM SPSS Modeler Server Connection” for more information.

To search for servers and clusters

1. On the Tools menu, click **Server Login**. The Server Login dialog box opens.

2. In this dialog box, click **Search** to open the Search for Servers dialog box. If you are not logged on to IBM SPSS Collaboration and Deployment Services when you attempt to browse the Coordinator of Processes, you will be prompted to do so.
3. Select the server or server cluster from the list.
4. Click **OK** to close the dialog box and add this connection to the table in the Server Login dialog box.

Changing the Temp Directory

Some operations performed by IBM SPSS Modeler Server may require temporary files to be created. By default, IBM SPSS Modeler uses the system temporary directory to create temp files. You can alter the location of the temporary directory using the following steps.
1. Create a new directory called *spss* and subdirectory called *servertemp*.
2. Edit *options.cfg*, located in the */config* directory of your IBM SPSS Modeler installation directory. Edit the `temp_directory` parameter in this file to read: `temp_directory, "C:/spss/servertemp"`.
3. After doing this, you must restart the IBM SPSS Modeler Server service. You can do this by clicking the **Services** tab on your Windows Control Panel. Just stop the service and then start it to activate the changes you made. Restarting the machine will also restart the service.

All temp files will now be written to this new directory.

Note: The most common error when you are attempting to do this is to use the wrong type of slashes; forward slashes are used.

Starting Multiple IBM SPSS Modeler Sessions

If you need to launch more than one IBM SPSS Modeler session at a time, you must make some changes to your IBM SPSS Modeler and Windows settings. For example, you may need to do this if you have two separate server licenses and want to run two streams against two different servers from the same client machine.

To enable multiple IBM SPSS Modeler sessions:
1. Click:
   Start > **[All] Programs** > **IBM SPSS Modeler 16**
2. On the IBM SPSS Modeler 16 shortcut (the one with the icon), right-click and select **Properties**.
3. In the **Target** text box, add `-noshare` to the end of the string.
4. In Windows Explorer, select:
   Tools > **Folder Options...**
5. On the File Types tab, select the IBM SPSS Modeler Stream option and click **Advanced**.
6. In the Edit File Type dialog box, select Open with IBM SPSS Modeler and click **Edit**.
7. In the **Application used to perform action** text box, add `-noshare` before the **-stream** argument.

IBM SPSS Modeler Interface at a Glance

At each point in the data mining process, IBM SPSS Modeler's easy-to-use interface invites your specific business expertise. Modeling algorithms, such as prediction, classification, segmentation, and association detection, ensure powerful and accurate models. Model results can easily be deployed and read into databases, IBM SPSS Statistics, and a wide variety of other applications.

Working with IBM SPSS Modeler is a three-step process of working with data.
- First, you read data into IBM SPSS Modeler.
- Next, you run the data through a series of manipulations.
- Finally, you send the data to a destination.

This sequence of operations is known as a **data stream** because the data flows record by record from the source through each manipulation and, finally, to the destination--either a model or type of data output.

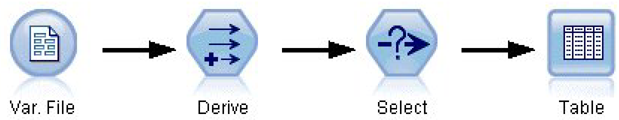


Figure 2. A simple stream

IBM SPSS Modeler Stream Canvas

The stream canvas is the largest area of the IBM SPSS Modeler window and is where you will build and manipulate data streams.

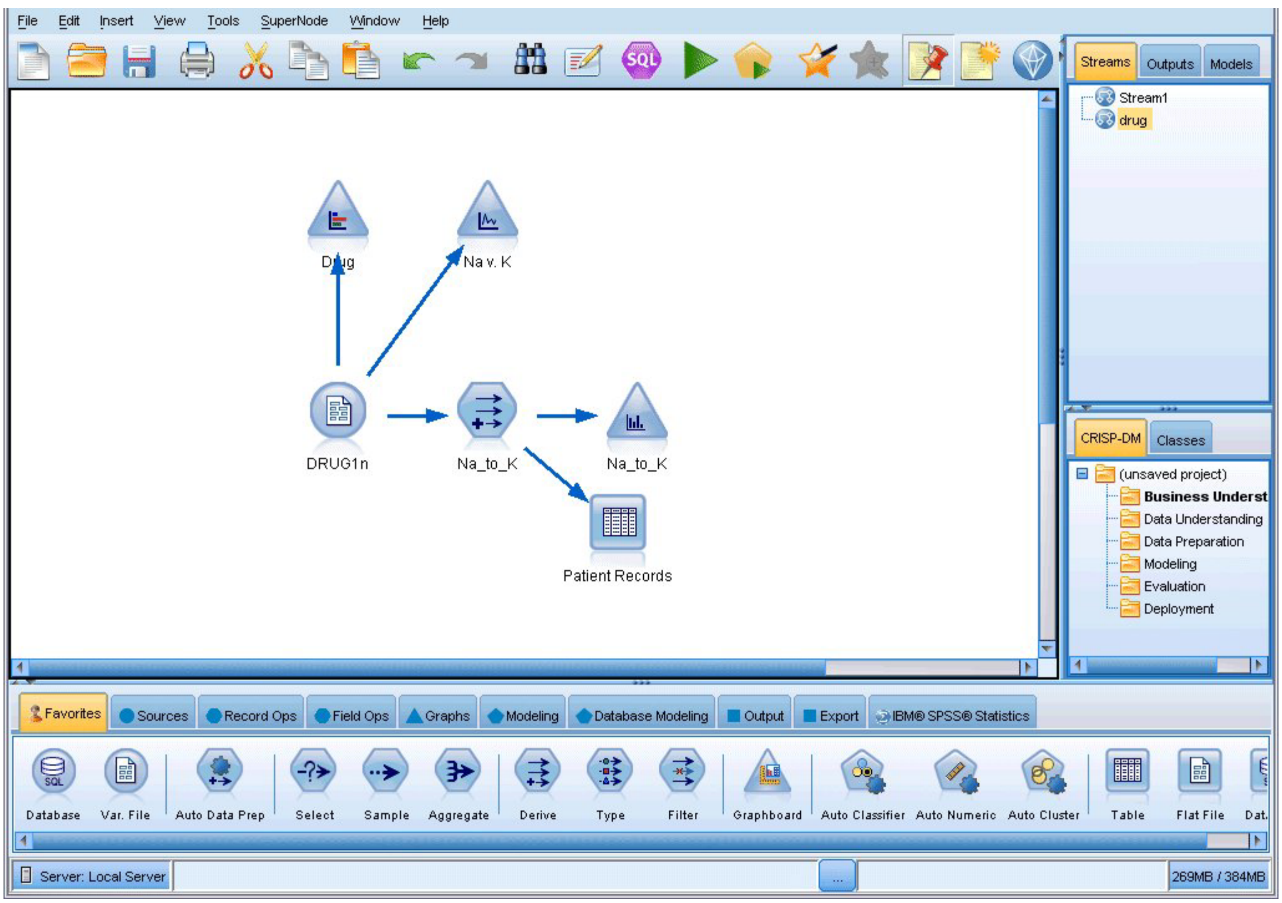


Figure 3. IBM SPSS Modeler workspace (default view)

Streams are created by drawing diagrams of data operations relevant to your business on the main canvas in the interface. Each operation is represented by an icon or **node**, and the nodes are linked together in a **stream** representing the flow of data through each operation.

You can work with multiple streams at one time in IBM SPSS Modeler, either in the same stream canvas or by opening a new stream canvas. During a session, streams are stored in the Streams manager, at the upper right of the IBM SPSS Modeler window.

Nodes Palette

Most of the data and modeling tools in IBM SPSS Modeler reside in the **Nodes Palette**, across the bottom of the window below the stream canvas.

For example, the Record Ops palette tab contains nodes that you can use to perform operations on the data **records**, such as selecting, merging, and appending.

To add nodes to the canvas, double-click icons from the Nodes Palette or drag and drop them onto the canvas. You then connect them to create a **stream**, representing the flow of data.

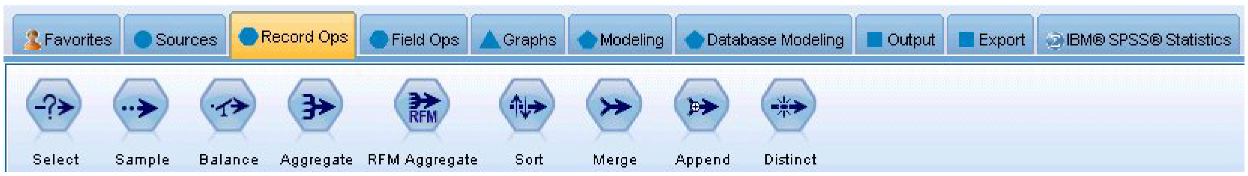


Figure 4. Record Ops tab on the nodes palette

Each palette tab contains a collection of related nodes used for different phases of stream operations, such as:

- **Sources.** Nodes bring data into IBM SPSS Modeler.
- **Record Ops.** Nodes perform operations on data **records**, such as selecting, merging, and appending.
- **Field Ops.** Nodes perform operations on data **fields**, such as filtering, deriving new fields, and determining the measurement level for given fields.
- **Graphs.** Nodes graphically display data before and after modeling. Graphs include plots, histograms, web nodes, and evaluation charts.
- **Modeling.** Nodes use the modeling algorithms available in IBM SPSS Modeler, such as neural nets, decision trees, clustering algorithms, and data sequencing.
- **Database Modeling.** Nodes use the modeling algorithms available in Microsoft SQL Server, IBM DB2, and Oracle and Netezza databases.
- **Output.** Nodes produce a variety of output for data, charts, and model results that can be viewed in IBM SPSS Modeler.
- **Export.** Nodes produce a variety of output that can be viewed in external applications, such as IBM SPSS Data Collection or Excel.
- **IBM SPSS Statistics.** Nodes import data from, or export data to, IBM SPSS Statistics, as well as running IBM SPSS Statistics procedures.

As you become more familiar with IBM SPSS Modeler, you can customize the palette contents for your own use.

Located below the Nodes Palette, a report pane provides feedback on the progress of various operations, such as when data is being read into the data stream. Also located below the Nodes Palette, a status pane provides information on what the application is currently doing, as well as indications of when user feedback is required.

IBM SPSS Modeler Managers

At the top right of the window is the managers pane. This has three tabs, which are used to manage streams, output and models.

You can use the Streams tab to open, rename, save, and delete the streams created in a session.

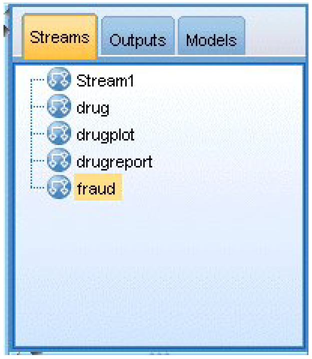


Figure 5. Streams tab

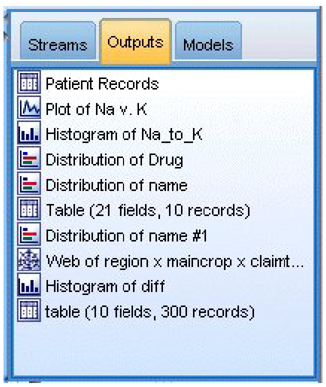


Figure 6. Outputs tab

The Outputs tab contains a variety of files, such as graphs and tables, produced by stream operations in IBM SPSS Modeler. You can display, save, rename, and close the tables, graphs, and reports listed on this tab.

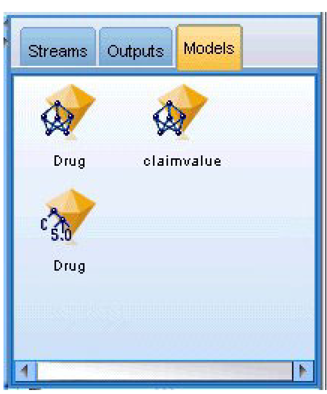


Figure 7. Models tab containing model nuggets

The Models tab is the most powerful of the manager tabs. This tab contains all model **nuggets**, which contain the models generated in IBM SPSS Modeler, for the current session. These models can be browsed directly from the Models tab or added to the stream in the canvas.

IBM SPSS Modeler Projects

On the lower right side of the window is the project pane, used to create and manage data mining **projects** (groups of files related to a data mining task). There are two ways to view projects you create in IBM SPSS Modeler—in the Classes view and the CRISP-DM view.

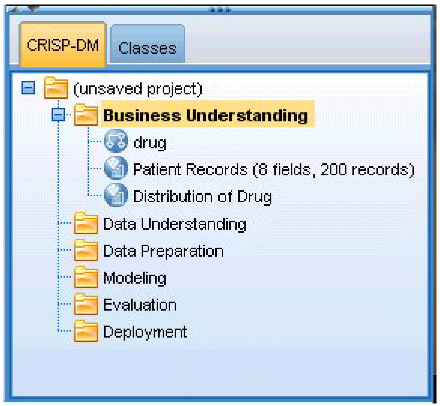


Figure 8. CRISP-DM view

The CRISP-DM tab provides a way to organize projects according to the Cross-Industry Standard Process for Data Mining, an industry-proven, nonproprietary methodology. For both experienced and first-time data miners, using the CRISP-DM tool will help you to better organize and communicate your efforts.

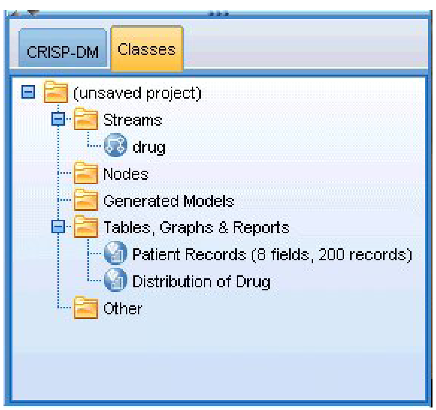


Figure 9. Classes view

The Classes tab provides a way to organize your work in IBM SPSS Modeler categorically—by the types of objects you create. This view is useful when taking inventory of data, streams, and models.

IBM SPSS Modeler Toolbar

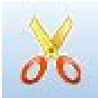
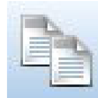
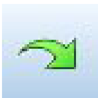
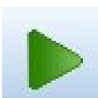
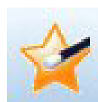
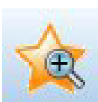
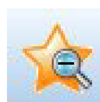
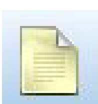
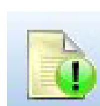
At the top of the IBM SPSS Modeler window, you will find a toolbar of icons that provides a number of useful functions. Following are the toolbar buttons and their functions.

 Create new stream

 Open stream

 Save stream

 Print current stream

	Cut & move to clipboard		Copy to clipboard
	Paste selection		Undo last action
	Redo		Search for nodes
	Edit stream properties		Preview SQL generation
	Run current stream		Run stream selection
	Stop stream (Active only while stream is running)		Add SuperNode
	Zoom in (SuperNodes only)		Zoom out (SuperNodes only)
	No markup in stream		Insert comment
	Hide stream markup (if any)		Show hidden stream markup
	Open stream in IBM SPSS Modeler Advantage		

Stream markup consists of stream comments, model links, and scoring branch indications.

Model links are described in the *IBM SPSS Modeling Nodes* guide.

Customizing the Toolbar

You can change various aspects of the toolbar, such as:
- Whether it is displayed
- Whether the icons have tooltips available
- Whether it uses large or small icons

To turn the toolbar display on and off:
1. On the main menu, click:
   View > **Toolbar** > **Display**

To change the tooltip or icon size settings:

1. On the main menu, click:

   View > **Toolbar** > **Customize**

Click **Show ToolTips** or **Large Buttons** as required.

Customizing the IBM SPSS Modeler Window

Using the dividers between various portions of the IBM SPSS Modeler interface, you can resize or close tools to meet your preferences. For example, if you are working with a large stream, you can use the small arrows located on each divider to close the nodes palette, managers pane, and project pane. This maximizes the stream canvas, providing enough work space for large or multiple streams.

Alternatively, on the View menu, click **Nodes Palette**, **Managers**, or **Project** to turn the display of these items on or off.

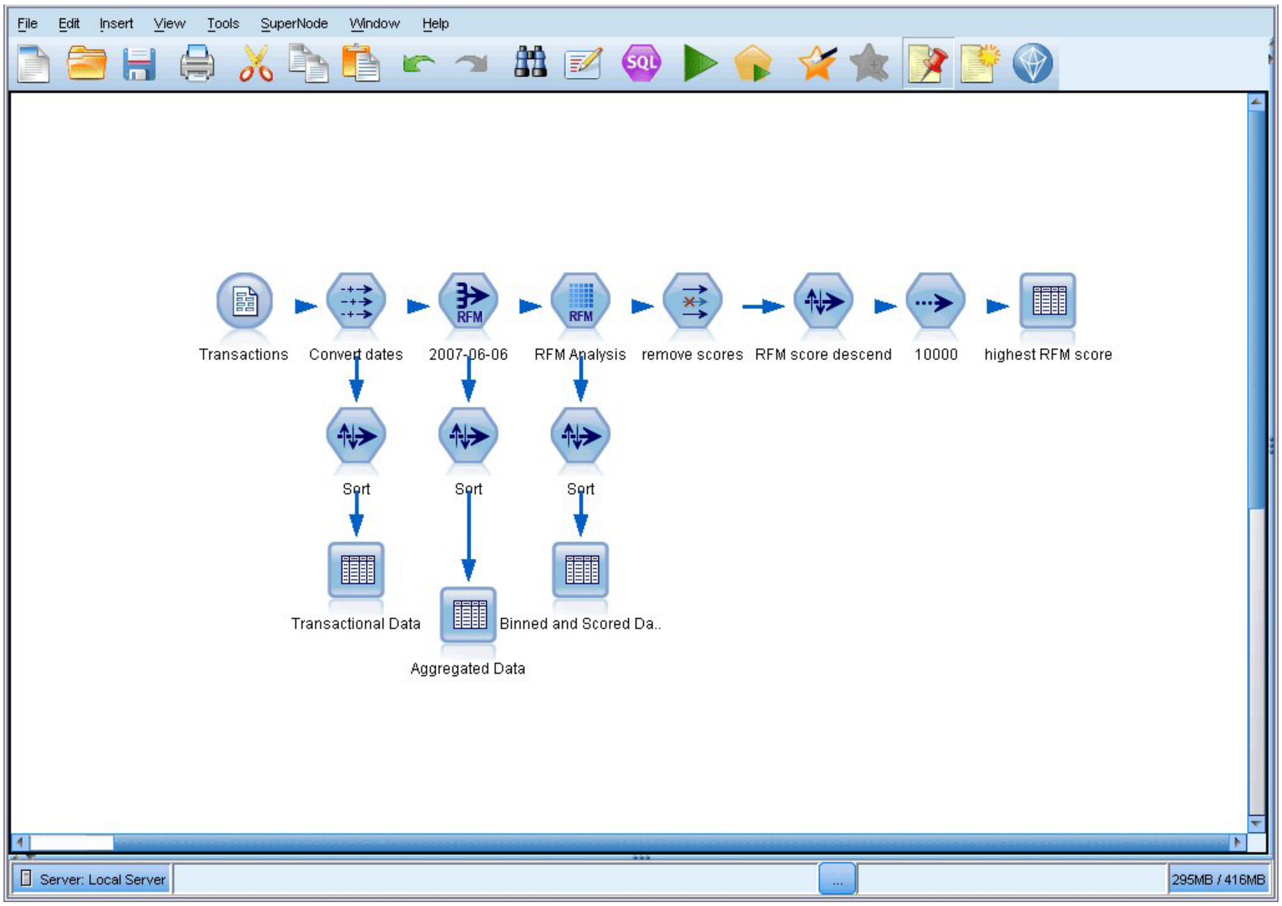


Figure 10. Maximized stream canvas

As an alternative to closing the nodes palette, and the managers and project panes, you can use the stream canvas as a scrollable page by moving vertically and horizontally with the scrollbars at the side and bottom of the IBM SPSS Modeler window.

You can also control the display of screen markup, which consists of stream comments, model links, and scoring branch indications. To turn this display on or off, click:

View > **Stream Markup**

Changing the icon size for a stream

You can change the size of the stream icons in the following ways.

- Through a stream property setting
- Through a pop-up menu in the stream
- Using the keyboard

You can scale the entire stream view to one of a number of sizes between 8% and 200% of the standard icon size.

To scale the entire stream (stream properties method)

1. From the main menu, choose
   Tools > **Stream Properties** > **Options** > **Layout.**
2. Choose the size you want from the Icon Size menu.
3. Click **Apply** to see the result.
4. Click **OK** to save the change.

To scale the entire stream (menu method)

1. Right-click the stream background on the canvas.
2. Choose **Icon Size** and select the size you want.

To scale the entire stream (keyboard method)

1. Press Ctrl + [-] on the main keyboard to zoom out to the next smaller size.
2. Press Ctrl + Shift + [+] on the main keyboard to zoom in to the next larger size.

This feature is particularly useful for gaining an overall view of a complex stream. You can also use it to minimize the number of pages needed to print a stream.

Using the Mouse in IBM SPSS Modeler

The most common uses of the mouse in IBM SPSS Modeler include the following:

- **Single-click.** Use either the right or left mouse button to select options from menus, open pop-up menus, and access various other standard controls and options. Click and hold the button to move and drag nodes.
- **Double-click.** Double-click using the left mouse button to place nodes on the stream canvas and edit existing nodes.
- **Middle-click.** Click the middle mouse button and drag the cursor to connect nodes on the stream canvas. Double-click the middle mouse button to disconnect a node. If you do not have a three-button mouse, you can simulate this feature by pressing the Alt key while clicking and dragging the mouse.

Using Shortcut Keys

Many visual programming operations in IBM SPSS Modeler have shortcut keys associated with them. For example, you can delete a node by clicking the node and pressing the Delete key on your keyboard. Likewise, you can quickly save a stream by pressing the S key while holding down the Ctrl key. Control commands like this one are indicated by a combination of Ctrl and another key--for example, Ctrl+S.

There are a number of shortcut keys used in standard Windows operations, such as Ctrl+X to cut. These shortcuts are supported in IBM SPSS Modeler along with the following application-specific shortcuts.

Note: In some cases, old shortcut keys used in IBM SPSS Modeler conflict with standard Windows shortcut keys. These old shortcuts are supported with the addition of the Alt key. For example, Ctrl+Alt+C can be used to toggle the cache on and off.

Table 1. Supported shortcut keys

Shortcut Key	Function
Ctrl+A	Select all
Ctrl+X	Cut
Ctrl+N	New stream
Ctrl+O	Open stream
Ctrl+P	Print
Ctrl+C	Copy
Ctrl+V	Paste
Ctrl+Z	Undo
Ctrl+Q	Select all nodes downstream of the selected node
Ctrl+W	Deselect all downstream nodes (toggles with Ctrl+Q)
Ctrl+E	Run from selected node
Ctrl+S	Save current stream
Alt+Arrow keys	Move selected nodes on the stream canvas in the direction of the arrow used
Shift+F10	Open the pop-up menu for the selected node

Table 2. Supported shortcuts for old hot keys

Shortcut Key	Function
Ctrl+Alt+D	Duplicate node
Ctrl+Alt+L	Load node
Ctrl+Alt+R	Rename node
Ctrl+Alt+U	Create User Input node
Ctrl+Alt+C	Toggle cache on/off
Ctrl+Alt+F	Flush cache
Ctrl+Alt+X	Expand SuperNode
Ctrl+Alt+Z	Zoom in/zoom out
Delete	Delete node or connection

Printing

The following objects can be printed in IBM SPSS Modeler:
- Stream diagrams
- Graphs
- Tables
- Reports (from the Report node and Project Reports)
- Scripts (from the stream properties, Standalone Script, or SuperNode script dialog boxes)
- Models (Model browsers, dialog box tabs with current focus, tree viewers)
- Annotations (using the Annotations tab for output)

To print an object:
- To print without previewing, click the Print button on the toolbar.
- To set up the page before printing, select **Page Setup** from the File menu.
- To preview before printing, select **Print Preview** from the File menu.

- To view the standard print dialog box with options for selecting printers, and specifying appearance options, select **Print** from the File menu.

Automating IBM SPSS Modeler

Since advanced data mining can be a complex and sometimes lengthy process, IBM SPSS Modeler includes several types of coding and automation support.

- **Control Language for Expression Manipulation** (CLEM) is a language for analyzing and manipulating the data that flows along IBM SPSS Modeler streams. Data miners use CLEM extensively in stream operations to perform tasks as simple as deriving profit from cost and revenue data or as complex as transforming web log data into a set of fields and records with usable information.
- **Scripting** is a powerful tool for automating processes in the user interface. Scripts can perform the same kinds of actions that users perform with a mouse or a keyboard. You can also specify output and manipulate generated models.

Chapter 3. Introduction to Modeling

A model is a set of rules, formulas, or equations that can be used to predict an outcome based on a set of input fields or variables. For example, a financial institution might use a model to predict whether loan applicants are likely to be good or bad risks, based on information that is already known about past applicants.

The ability to predict an outcome is the central goal of predictive analytics, and understanding the modeling process is the key to using IBM SPSS Modeler.

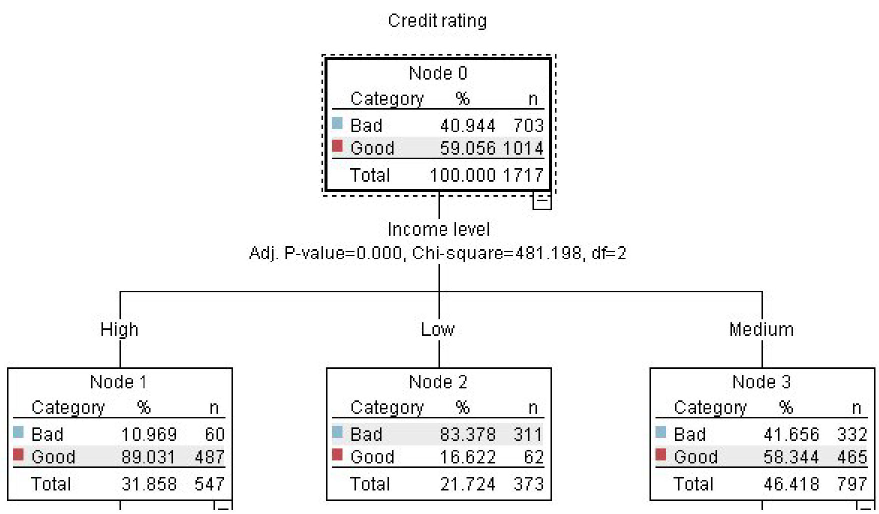


Figure 11. A simple decision tree model

This example uses a **decision tree** model, which classifies records (and predicts a response) using a series of decision rules, for example:

```
IF income = Medium
AND cards <5
THEN -> 'Good'
```

While this example uses a CHAID (Chi-squared Automatic Interaction Detection) model, it is intended as a general introduction, and most of the concepts apply broadly to other modeling types in IBM SPSS Modeler.

To understand any model, you first need to understand the data that go into it. The data in this example contain information about the customers of a bank. The following fields are used:

Field name	Description
Credit_rating	Credit rating: 0=Bad, 1=Good, 9=missing values
Age	Age in years
Income	Income level: 1=Low, 2=Medium, 3=High
Credit_cards	Number of credit cards held: 1=Less than five, 2=Five or more
Education	Level of education: 1=High school, 2=College
Car_loans	Number of car loans taken out: 1=None or one, 2=More than two

The bank maintains a database of historical information on customers who have taken out loans with the bank, including whether or not they repaid the loans (Credit rating = Good) or defaulted (Credit rating = Bad). Using this existing data, the bank wants to build a model that will enable them to predict how likely future loan applicants are to default on the loan.

Using a decision tree model, you can analyze the characteristics of the two groups of customers and predict the likelihood of loan defaults.

This example uses the stream named *modelingintro.str*, available in the *Demos* folder under the *streams* subfolder. The data file is *tree_credit.sav*. See the topic "Demos Folder" on page 4 for more information.

Let's take a look at the stream.
1. Choose the following from the main menu:

   File > **Open Stream**
2. Click the gold nugget icon on the toolbar of the Open dialog box and choose the Demos folder.
3. Double-click the *streams* folder.
4. Double-click the file named *modelingintro.str*.

Building the Stream

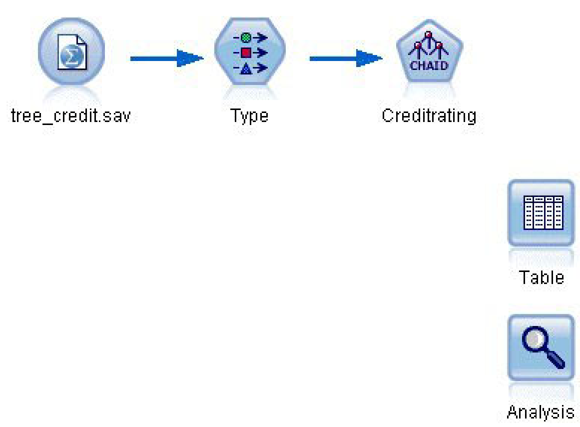


Figure 12. Modeling stream

To build a stream that will create a model, we need at least three elements:
- A source node that reads in data from some external source, in this case an IBM SPSS Statistics data file.
- A source or Type node that specifies field properties, such as measurement level (the type of data that the field contains), and the role of each field as a target or input in modeling.
- A modeling node that generates a model nugget when the stream is run.

In this example, we're using a CHAID modeling node. CHAID, or Chi-squared Automatic Interaction Detection, is a classification method that builds decision trees by using a particular type of statistics known as chi-square statistics to work out the best places to make the splits in the decision tree.

If measurement levels are specified in the source node, the separate Type node can be eliminated. Functionally, the result is the same.

This stream also has Table and Analysis nodes that will be used to view the scoring results after the model nugget has been created and added to the stream.

The Statistics File source node reads data in IBM SPSS Statistics format from the *tree_credit.sav* data file, which is installed in the *Demos* folder. (A special variable named *$CLEO_DEMOS* is used to reference this folder under the current IBM SPSS Modeler installation. This ensures the path will be valid regardless of the current installation folder or version.)

Figure 13. Reading data with a Statistics File source node

The Type node specifies the **measurement level** for each field. The measurement level is a category that indicates the type of data in the field. Our source data file uses three different measurement levels.

A **Continuous** field (such as the *Age* field) contains continuous numeric values, while a **Nominal** field (such as the *Credit rating* field) has two or more distinct values, for example *Bad*, *Good*, or *No credit history*. An **Ordinal** field (such as the *Income level* field) describes data with multiple distinct values that have an inherent order—in this case *Low*, *Medium* and *High*.

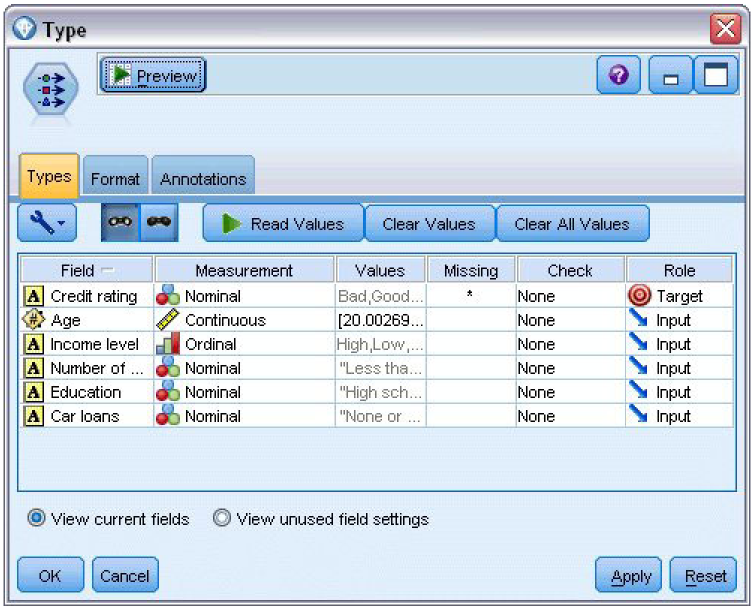


Figure 14. Setting the target and input fields with the Type node

For each field, the Type node also specifies a **role**, to indicate the part that each field plays in modeling. The role is set to *Target* for the field *Credit rating*, which is the field that indicates whether or not a given customer defaulted on the loan. This is the **target**, or the field for which we want to predict the value.

Role is set to *Input* for the other fields. Input fields are sometimes known as **predictors**, or fields whose values are used by the modeling algorithm to predict the value of the target field.

The CHAID modeling node generates the model.

On the Fields tab in the modeling node, the option **Use predefined roles** is selected, which means the target and inputs will be used as specified in the Type node. We could change the field roles at this point, but for this example we'll use them as they are.

1. Click the Build Options tab.

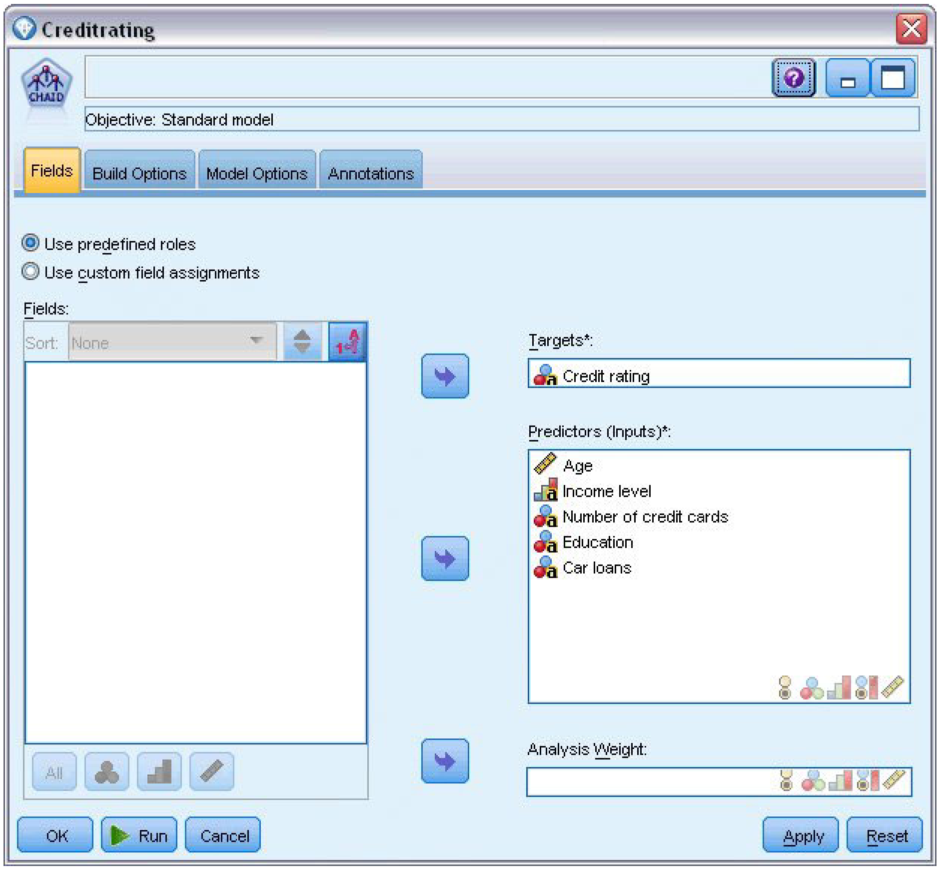


Figure 15. CHAID modeling node, Fields tab

Here there are several options where we could specify the kind of model we want to build.

We want a brand-new model, so we'll use the default option **Build new model**.

We also just want a single, standard decision tree model without any enhancements, so we'll also leave the default objective option **Build a single tree**.

While we can optionally launch an interactive modeling session that allows us to fine-tune the model, this example simply generates a model using the default mode setting **Generate model**.

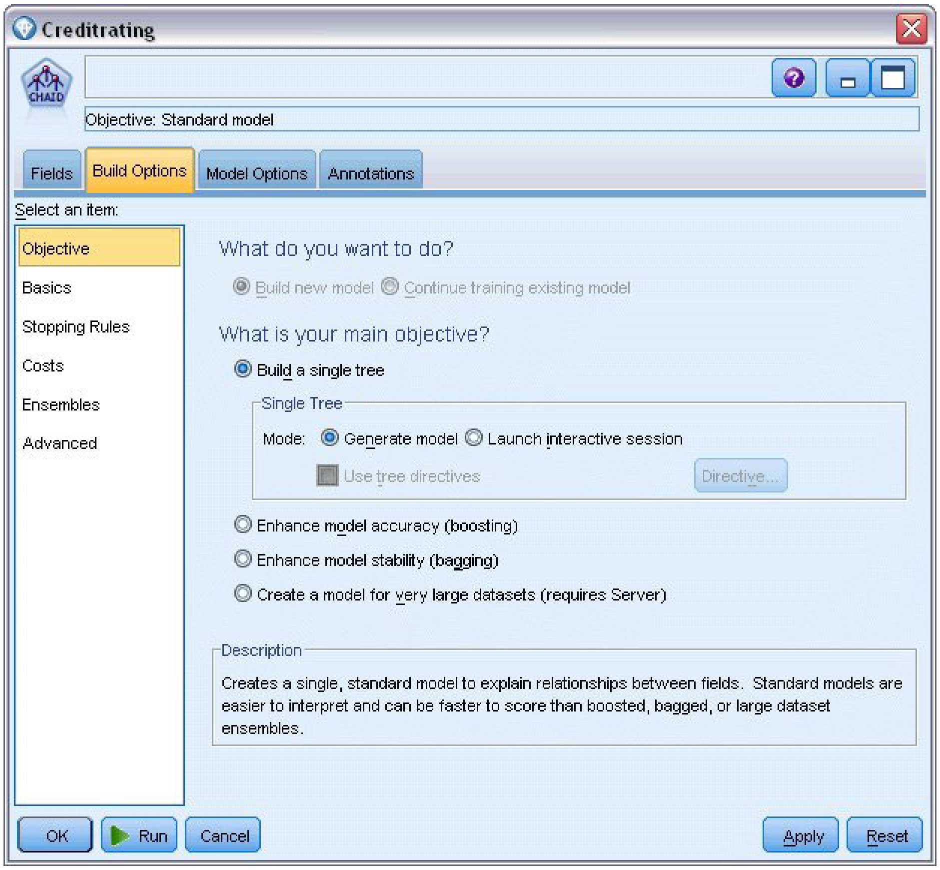


Figure 16. CHAID modeling node, Build Options tab

For this example, we want to keep the tree fairly simple, so we'll limit the tree growth by raising the minimum number of cases for parent and child nodes.

2. On the Build Options tab, select **Stopping Rules** from the navigator pane on the left.
3. Select the **Use absolute value** option.
4. Set **Minimum records in parent branch** to 400.
5. Set **Minimum records in child branch** to 200.

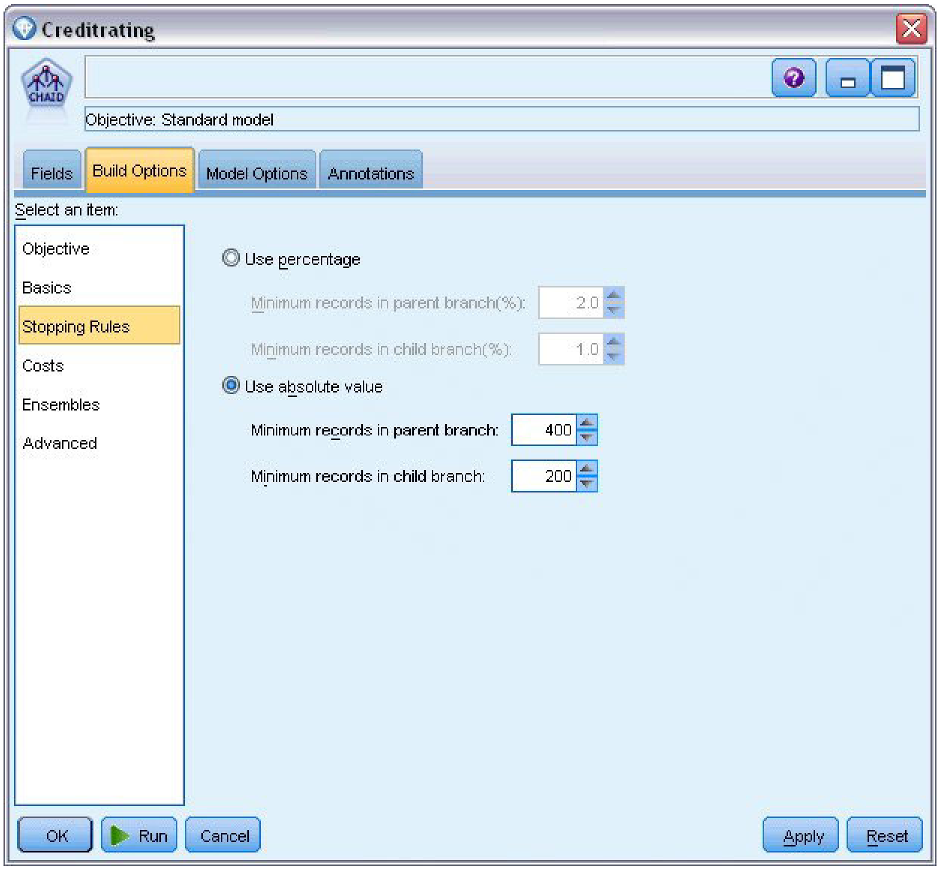


Figure 17. Setting the stopping criteria for decision tree building

We can use all the other default options for this example, so click **Run** to create the model. (Alternatively, right-click on the node and choose **Run** from the context menu, or select the node and choose **Run** from the Tools menu.)

Browsing the Model

When execution completes, the model nugget is added to the Models palette in the upper right corner of the application window, and is also placed on the stream canvas with a link to the modeling node from which it was created. To view the model details, right-click on the model nugget and choose **Browse** (on the models palette) or **Edit** (on the canvas).

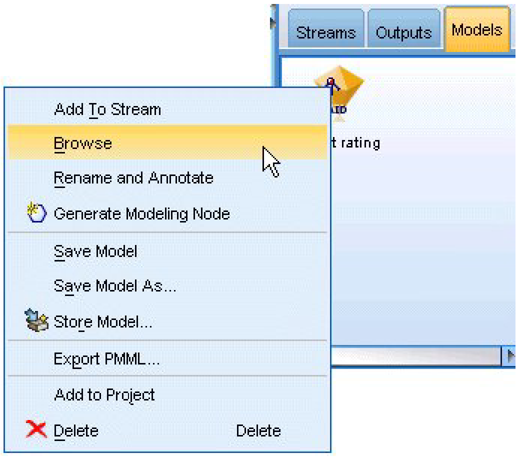


Figure 18. Models palette

In the case of the CHAID nugget, the Model tab displays the details in the form of a rule set--essentially a series of rules that can be used to assign individual records to child nodes based on the values of different input fields.

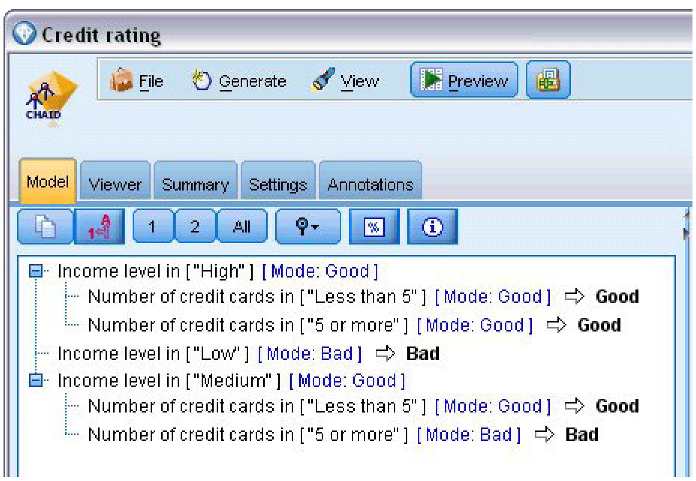


Figure 19. CHAID model nugget, rule set

For each decision tree terminal node--meaning those tree nodes that are not split further--a prediction of *Good* or *Bad* is returned. In each case the prediction is determined by the **mode**, or most common response, for records that fall within that node.

To the right of the rule set, the Model tab displays the Predictor Importance chart, which shows the relative importance of each predictor in estimating the model. From this we can see that *Income level* is easily the most significant in this case, and that the only other significant factor is *Number of credit cards*.

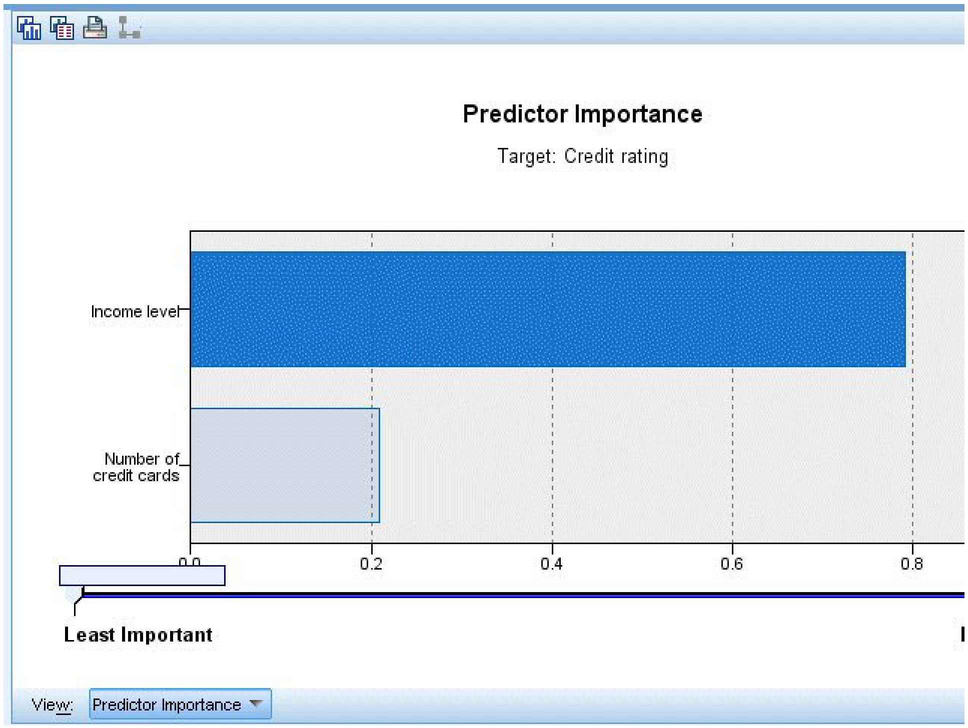


Figure 20. Predictor Importance chart

The Viewer tab in the model nugget displays the same model in the form of a tree, with a node at each decision point. Use the Zoom controls on the toolbar to zoom in on a specific node or zoom out to see the more of the tree.

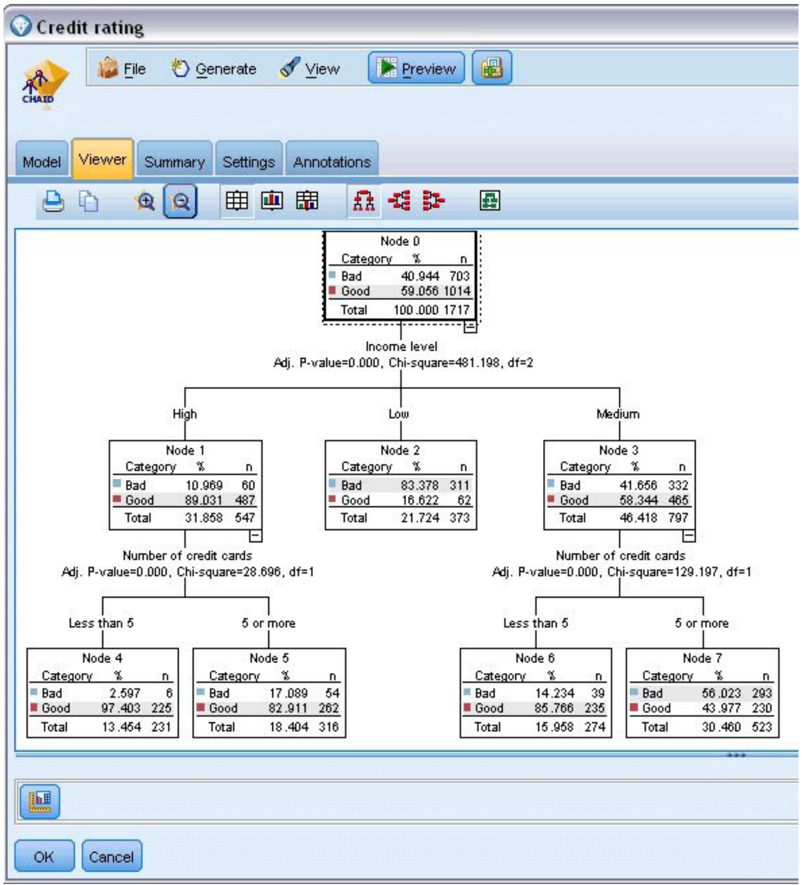


Figure 21. Viewer tab in the model nugget, with zoom out selected

Looking at the upper part of the tree, the first node (Node 0) gives us a summary for all the records in the data set. Just over 40% of the cases in the data set are classified as a bad risk. This is quite a high proportion, so let's see if the tree can give us any clues as to what factors might be responsible.

We can see that the first split is by *Income level*. Records where the income level is in the *Low* category are assigned to Node 2, and it's no surprise to see that this category contains the highest percentage of loan defaulters. Clearly lending to customers in this category carries a high risk.

However, 16% of the customers in this category actually *didn't* default, so the prediction won't always be correct. No model can feasibly predict every response, but a good model should allow us to predict the *most likely* response for each record based on the available data.

In the same way, if we look at the high income customers (Node 1), we see that the vast majority (89%) are a good risk. But more than 1 in 10 of these customers has also defaulted. Can we refine our lending criteria to minimize the risk here?

Notice how the model has divided these customers into two sub-categories (Nodes 4 and 5), based on the number of credit cards held. For high-income customers, if we lend only to those with fewer than 5 credit cards, we can increase our success rate from 89% to 97%--an even more satisfactory outcome.

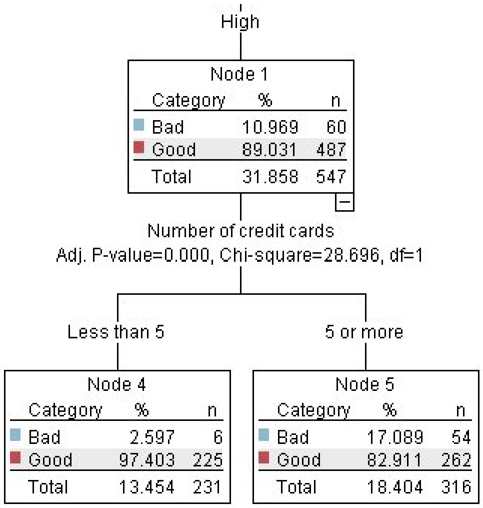


Figure 22. Tree view of high-income customers

But what about those customers in the Medium income category (Node 3)? They're much more evenly divided between Good and Bad ratings.

Again, the sub-categories (Nodes 6 and 7 in this case) can help us. This time, lending only to those medium-income customers with fewer than 5 credit cards increases the percentage of Good ratings from 58% to 85%, a significant improvement.

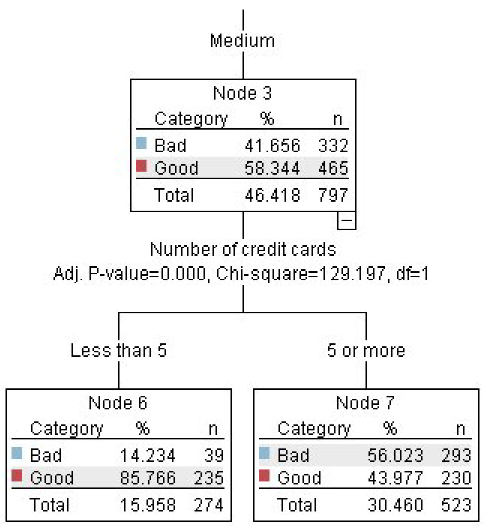


Figure 23. Tree view of medium-income customers

So, we've learnt that every record that is input to this model will be assigned to a specific node, and assigned a prediction of *Good* or *Bad* based on the most common response for that node.

This process of assigning predictions to individual records is known as **scoring**. By scoring the same records used to estimate the model, we can evaluate how accurately it performs on the training data—the data for which we know the outcome. Let's look at how to do this.

Evaluating the Model

We've been browsing the model to understand how scoring works. But to evaluate *how accurately* it works, we need to score some records and compare the responses predicted by the model to the actual results. We're going to score the same records that were used to estimate the model, allowing us to compare the observed and predicted responses.

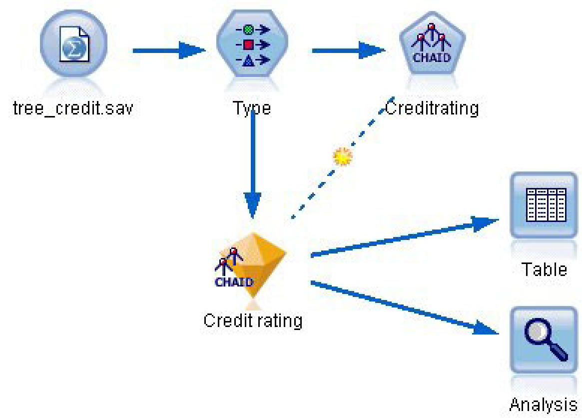
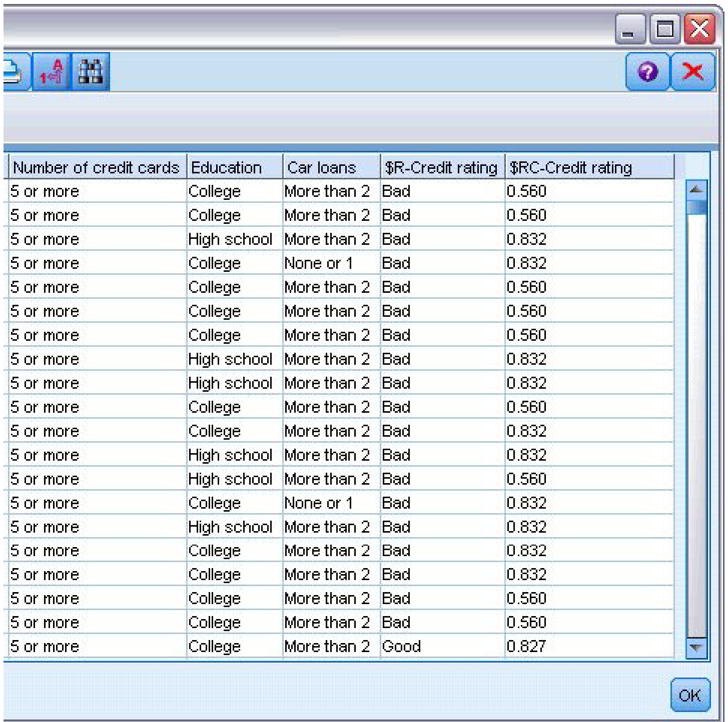


Figure 24. Attaching the model nugget to output nodes for model evaluation

1. To see the scores or predictions, attach the Table node to the model nugget, double-click the Table node and click **Run**.

   The table displays the predicted scores in a field named *$R-Credit rating*, which was created by the model. We can compare these values to the original *Credit rating* field that contains the actual responses.

   By convention, the names of the fields generated during scoring are based on the target field, but with a standard prefix such as *$R-* for predictions or *$RC-* for confidence values. Different models types use different sets of prefixes. A **confidence value** is the model's own estimation, on a scale from 0.0 to 1.0, of how accurate each predicted value is.



Number of credit cards	Education	Car loans	$R-Credit rating	$RC-Credit rating
5 or more	College	More than 2	Bad	0.560
5 or more	College	More than 2	Bad	0.560
5 or more	High school	More than 2	Bad	0.832
5 or more	College	None or 1	Bad	0.832
5 or more	College	More than 2	Bad	0.560
5 or more	College	More than 2	Bad	0.560
5 or more	College	More than 2	Bad	0.560
5 or more	High school	More than 2	Bad	0.832
5 or more	High school	More than 2	Bad	0.832
5 or more	College	More than 2	Bad	0.560
5 or more	College	More than 2	Bad	0.832
5 or more	High school	More than 2	Bad	0.832
5 or more	High school	More than 2	Bad	0.560
5 or more	College	None or 1	Bad	0.832
5 or more	High school	More than 2	Bad	0.832
5 or more	College	More than 2	Bad	0.832
5 or more	College	More than 2	Bad	0.832
5 or more	College	More than 2	Bad	0.560
5 or more	College	More than 2	Bad	0.560
5 or more	College	More than 2	Good	0.827

Figure 25. Table showing generated scores and confidence values

As expected, the predicted value matches the actual responses for many records but not all. The reason for this is that each CHAID terminal node has a mix of responses. The prediction matches the *most common* one, but will be wrong for all the others in that node. (Recall the 16% minority of low-income customers who did not default.)

To avoid this, we could continue splitting the tree into smaller and smaller branches, until every node was 100% pure—all *Good* or *Bad* with no mixed responses. But such a model would be extremely complicated and would probably not generalize well to other datasets.

To find out exactly how many predictions are correct, we could read through the table and tally the number of records where the value of the predicted field *$R-Credit rating* matches the value of *Credit rating*. Fortunately, there's a much easier way--we can use an Analysis node, which does this automatically.

2. Connect the model nugget to the Analysis node.
3. Double-click the Analysis node and click **Run**.

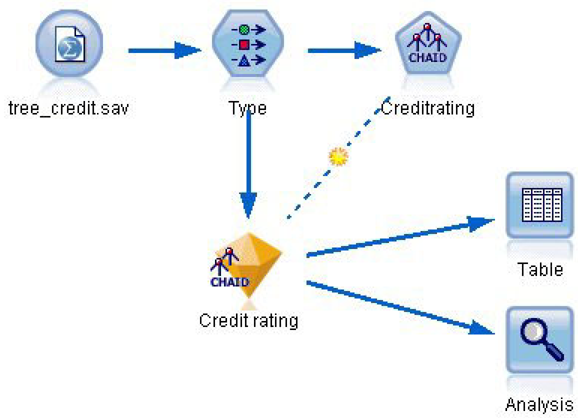


Figure 26. Attaching an Analysis node

The analysis shows that for 1899 out of 2464 records--over 77%--the value predicted by the model matched the actual response.

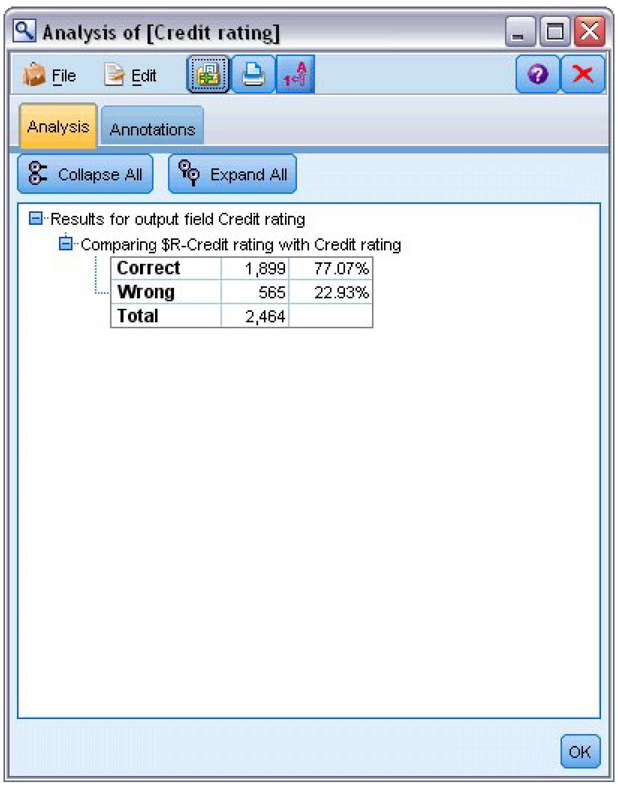


Figure 27. Analysis results comparing observed and predicted responses

This result is limited by the fact that the records being scored are the same ones used to estimate the model. In a real situation, you could use a Partition node to split the data into separate samples for training and evaluation.

By using one sample partition to generate the model and another sample to test it, you can get a much better indication of how well it will generalize to other datasets.

The Analysis node allows us to test the model against records for which we already know the actual result. The next stage illustrates how we can use the model to score records for which we don't know the outcome. For example, this might include people who are not currently customers of the bank, but who are prospective targets for a promotional mailing.

Scoring Records

Earlier, we scored the same records used to estimate the model in order to evaluate how accurate the model was. Now we're going to see how to score a different set of records from the ones used to create the model. This is the goal of modeling with a target field: Study records for which you know the outcome, to identify patterns that will allow you to predict outcomes you don't yet know.

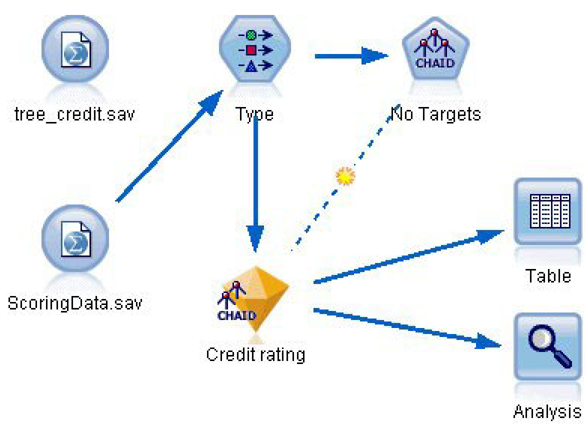


Figure 28. Attaching new data for scoring

You could update the Statistics File source node to point to a different data file, or you could add a new source node that reads in the data you want to score. Either way, the new dataset must contain the same input fields used by the model (*Age*, *Income level*, *Education* and so on) but not the target field *Credit rating*.

Alternatively, you could add the model nugget to any stream that includes the expected input fields. Whether read from a file or a database, the source type doesn't matter as long as the field names and types match those used by the model.

You could also save the model nugget as a separate file, export the model in PMML format for use with other applications that support this format, or store the model in an IBM SPSS Collaboration and Deployment Services repository, which offers enterprise-wide deployment, scoring, and management of models.

Regardless of the infrastructure used, the model itself works in the same way.

Summary

This example demonstrates the basic steps for creating, evaluating, and scoring a model.

- The modeling node estimates the model by studying records for which the outcome is known, and creates a model nugget. This is sometimes referred to as training the model.
- The model nugget can be added to any stream with the expected fields to score records. By scoring the records for which you already know the outcome (such as existing customers), you can evaluate how well it performs.
- Once you are satisfied that the model performs acceptably well, you can score new data (such as prospective customers) to predict how they will respond.

- The data used to train or estimate the model may be referred to as the analytical or historical data; the scoring data may also be referred to as the operational data.

Chapter 4. Automated Modeling for a Flag Target

Modeling Customer Response (Auto Classifier)

The Auto Classifier node enables you to automatically create and compare a number of different models for either flag (such as whether or not a given customer is likely to default on a loan or respond to a particular offer) or nominal (set) targets. In this example we'll search for a flag (yes or no) outcome. Within a relatively simple stream, the node generates and ranks a set of candidate models, chooses the ones that perform the best, and combines them into a single aggregated (Ensembled) model. This approach combines the ease of automation with the benefits of combining multiple models, which often yield more accurate predictions than can be gained from any one model.

This example is based on a fictional company that wants to achieve more profitable results by matching the right offer to each customer.

This approach stresses the benefits of automation. For a similar example that uses a continuous (numeric range) target, see .

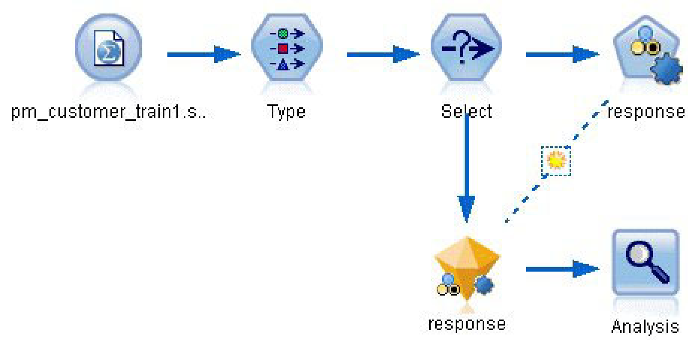


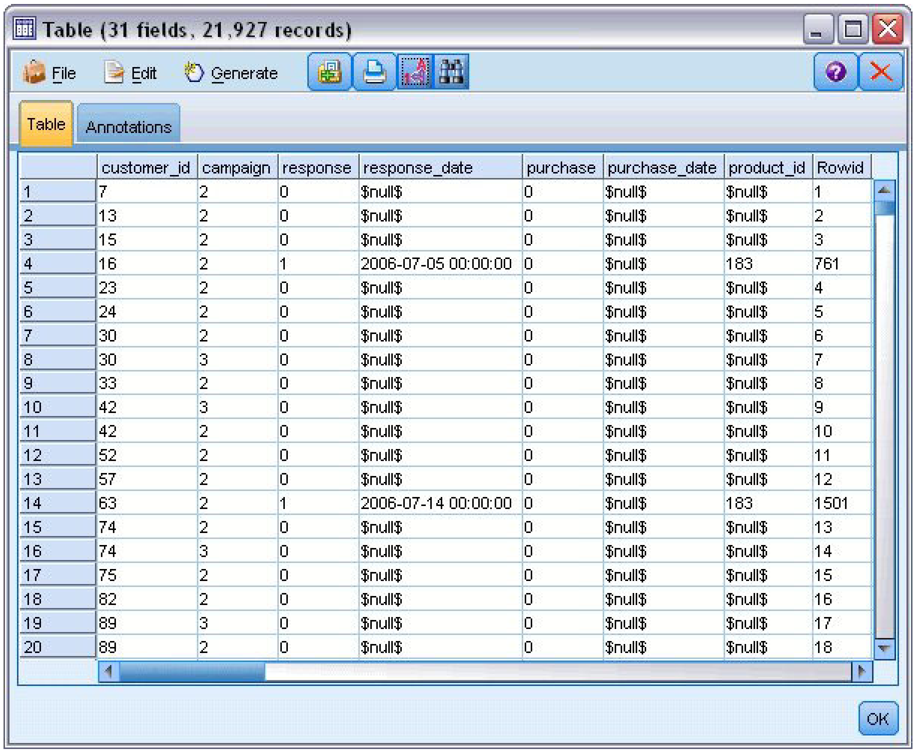
Figure 29. Auto Classifier sample stream

This example uses the stream *pm_binaryclassifier.str*, installed in the Demo folder under *streams*. The data file used is *pm_customer_train1.sav*. See the topic "Historical Data" for more information.

Historical Data

The file *pm_customer_train1.sav* has historical data tracking the offers made to specific customers in past campaigns, as indicated by the value of the *campaign* field. The largest number of records fall under the *Premium account* campaign.

The values of the *campaign* field are actually coded as integers in the data (for example 2 = *Premium account*). Later, you'll define labels for these values that you can use to give more meaningful output.



	customer_id	campaign	response	response_date	purchase	purchase_date	product_id	Rowid
1	7	2	0	$null$	0	$null$	$null$	1
2	13	2	0	$null$	0	$null$	$null$	2
3	15	2	0	$null$	0	$null$	$null$	3
4	16	2	1	2006-07-05 00:00:00	0	$null$	183	761
5	23	2	0	$null$	0	$null$	$null$	4
6	24	2	0	$null$	0	$null$	$null$	5
7	30	2	0	$null$	0	$null$	$null$	6
8	30	3	0	$null$	0	$null$	$null$	7
9	33	2	0	$null$	0	$null$	$null$	8
10	42	3	0	$null$	0	$null$	$null$	9
11	42	2	0	$null$	0	$null$	$null$	10
12	52	2	0	$null$	0	$null$	$null$	11
13	57	2	0	$null$	0	$null$	$null$	12
14	63	2	1	2006-07-14 00:00:00	0	$null$	183	1501
15	74	2	0	$null$	0	$null$	$null$	13
16	74	3	0	$null$	0	$null$	$null$	14
17	75	2	0	$null$	0	$null$	$null$	15
18	82	2	0	$null$	0	$null$	$null$	16
19	89	3	0	$null$	0	$null$	$null$	17
20	89	2	0	$null$	0	$null$	$null$	18

Figure 30. Data about previous promotions

The file also includes a *response* field that indicates whether the offer was accepted (0 = *no*, and 1 = *yes*). This will be the **target field**, or value, that you want to predict. A number of fields containing demographic and financial information about each customer are also included. These can be used to build or "train" a model that predicts response rates for individuals or groups based on characteristics such as income, age, or number of transactions per month.

Building the Stream

1. Add a Statistics File source node pointing to *pm_customer_train1.sav*, located in the *Demos* folder of your IBM SPSS Modeler installation. (You can specify `$CLEO_DEMOS/` in the file path as a shortcut to reference this folder. Note that a forward slash—rather than a backslash— must be used in the path, as shown.)

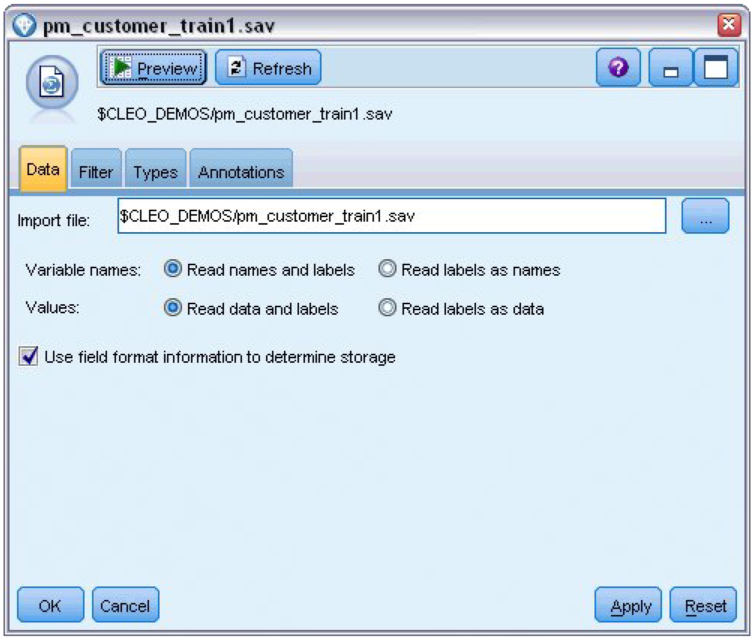


Figure 31. Reading in the data

2. Add a Type node, and select *response* as the target field (Role = **Target**). Set the Measurement for this field to **Flag**.

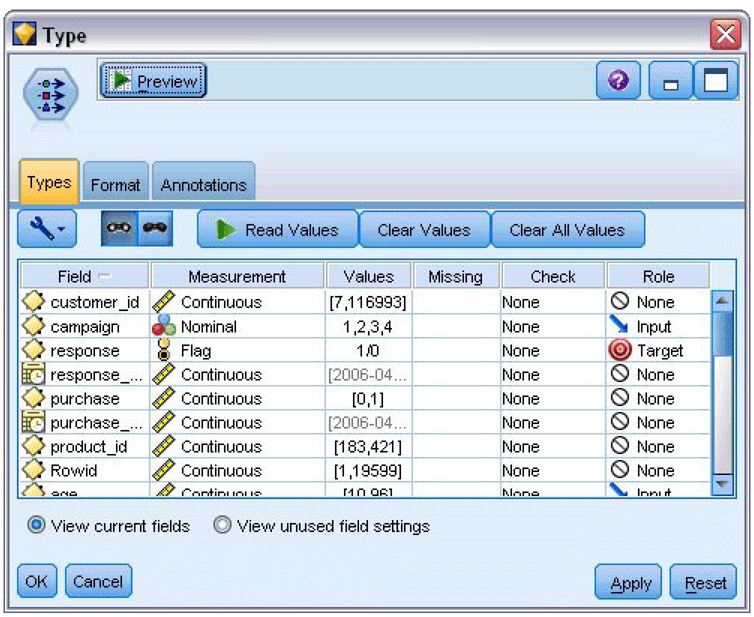


Figure 32. Setting the measurement level and role

3. Set the role to **None** for the following fields: *customer_id*, *campaign*, *response_date*, *purchase*, *purchase_date*, *product_id*, *Rowid*, and *X_random*. These fields will be ignored when you are building the model.
4. Click the **Read Values** button in the Type node to make sure that values are instantiated.

   As we saw earlier, our source data includes information about four different campaigns, each targeted to a different type of customer account. These campaigns are coded as integers in the data, so to make it easier to remember which account type each integer represents, let's define labels for each one.

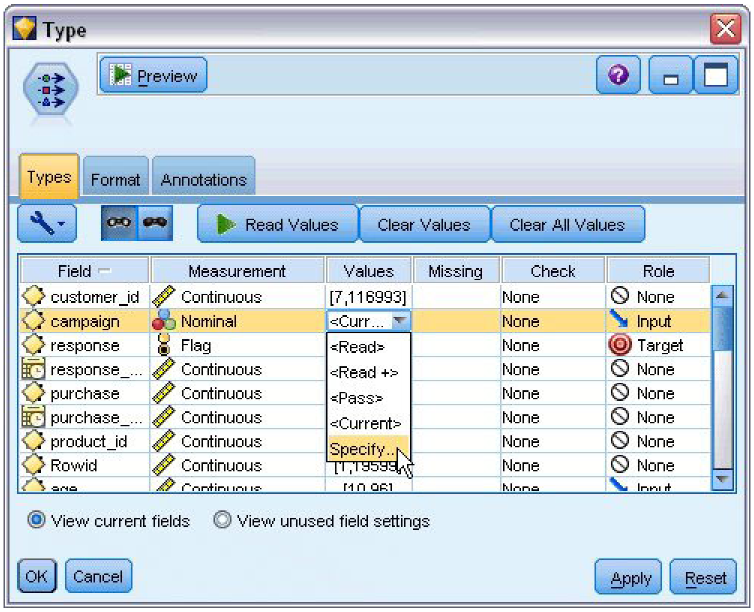


Figure 33. Choosing to specify values for a field

5. On the row for the **campaign** field, click the entry in the **Values** column.
6. Choose **Specify** from the drop-down list.

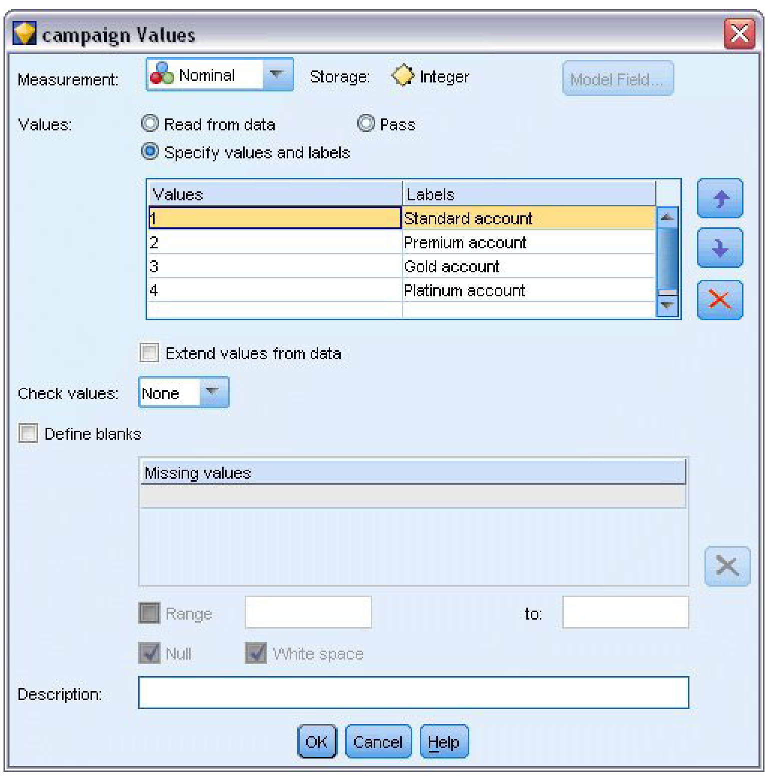


Figure 34. Defining labels for the field values

7. In the **Labels** column, type the labels as shown for each of the four values of the **campaign** field.

8. Click **OK**.

   Now you can display the labels in output windows instead of the integers.

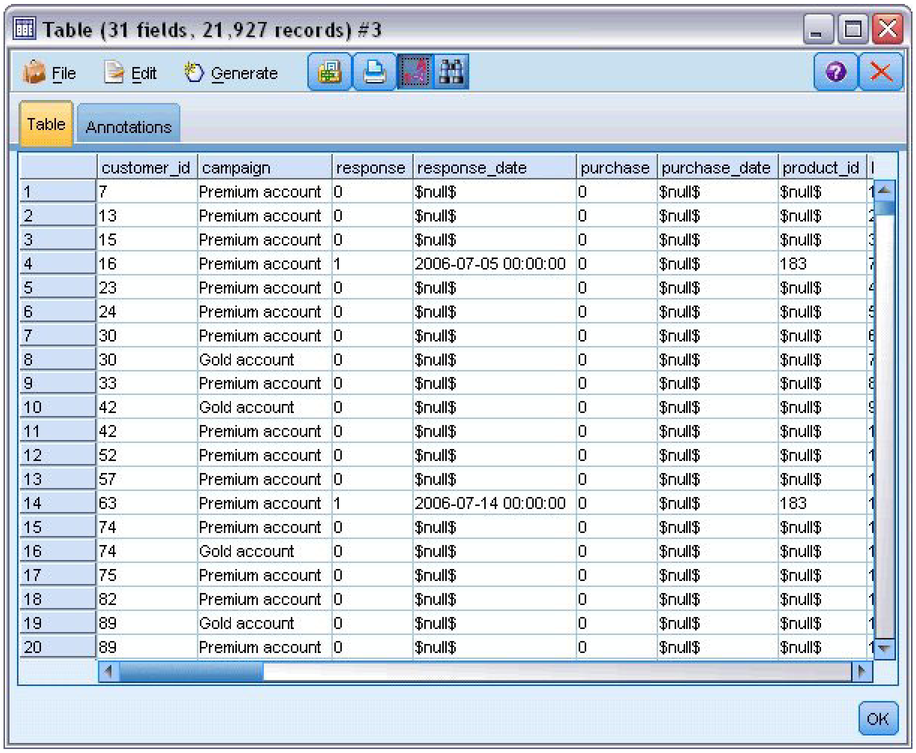


Figure 35. Displaying the field value labels

9. Attach a Table node to the Type node.
10. Open the Table node and click **Run**.
11. On the output window, click the **Display field and value labels** toolbar button to display the labels.
12. Click **OK** to close the output window.

Although the data includes information about four different campaigns, you will focus the analysis on one campaign at a time. Since the largest number of records fall under the Premium account campaign (coded *campaign=2* in the data), you can use a Select node to include only these records in the stream.

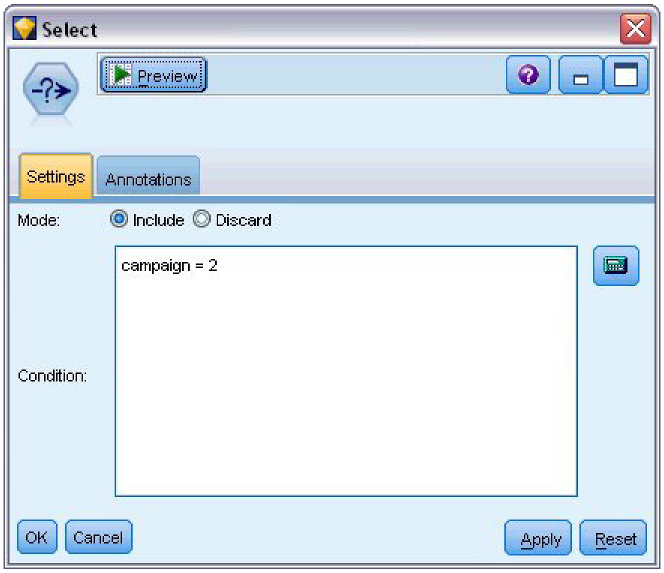


Figure 36. Selecting records for a single campaign

Generating and Comparing Models

1. Attach an Auto Classifier node, and select **Overall Accuracy** as the metric used to rank models.
2. Set the **Number of models to use** to 3. This means that the three best models will be built when you execute the node.

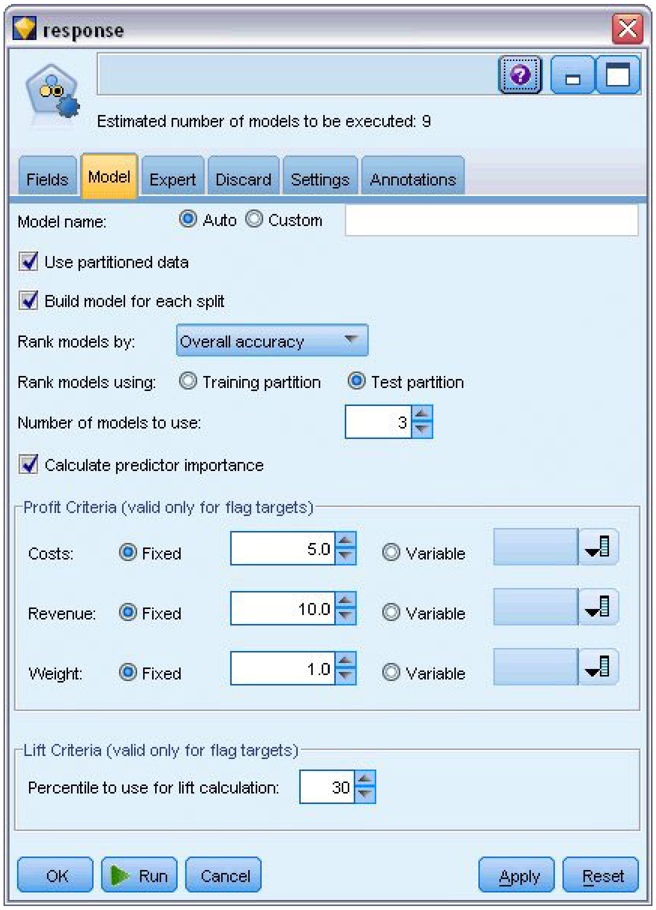


Figure 37. Auto Classifier node Model tab

On the Expert tab you can choose from up to 11 different model algorithms.

3. Deselect the **Discriminant** and **SVM** model types. (These models take longer to train on these data, so deselecting them will speed up the example. If you don't mind waiting, feel free to leave them selected.)

   Because you set **Number of models to use** to 3 on the Model tab, the node will calculate the accuracy of the remaining nine algorithms and build a single model nugget containing the three most accurate.

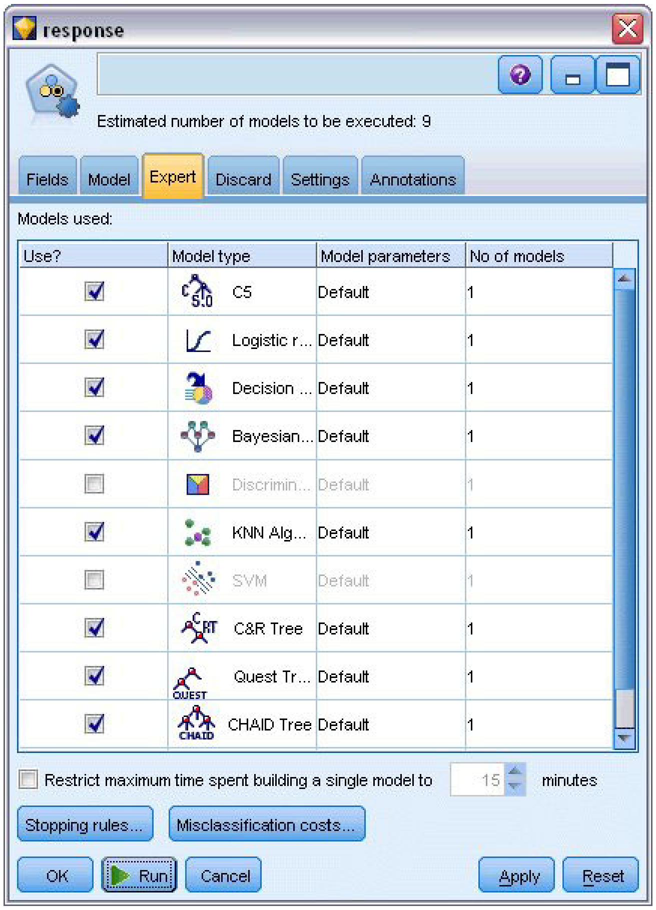


Figure 38. Auto Classifier node Expert tab

4. On the Settings tab, for the ensemble method, select **Confidence-weighted voting**. This determines how a single aggregated score is produced for each record.

   With simple voting, if two out of three models predict *yes*, then *yes* wins by a vote of 2 to 1. In the case of confidence-weighted voting, the votes are weighted based on the confidence value for each prediction. Thus, if one model predicts *no* with a higher confidence than the two *yes* predictions combined, then *no* wins.

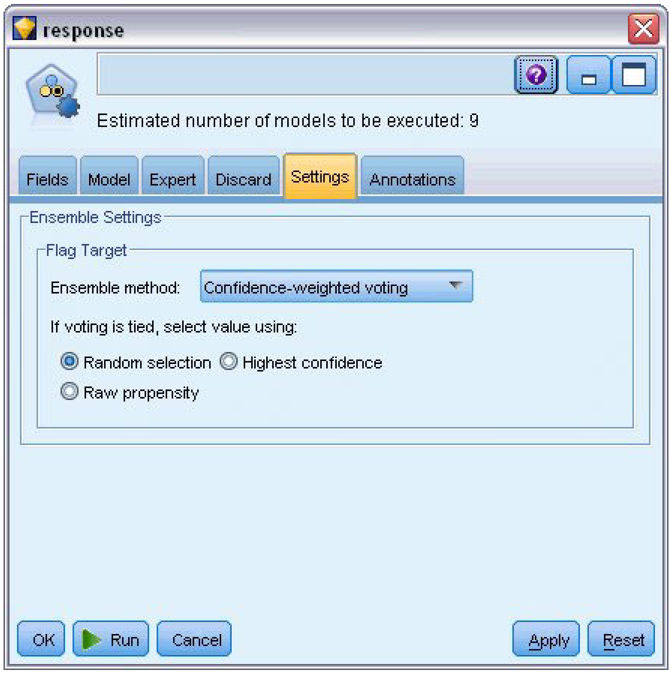


Figure 39. Auto Classifier node: Settings tab

5. Click **Run**.

After a few minutes, the generated model nugget is built and placed on the canvas, and on the Models palette in the upper right corner of the window. You can browse the model nugget, or save or deploy it in a number of other ways.

Open the model nugget; it lists details about each of the models created during the run. (In a real situation, in which hundreds of models may be created on a large dataset, this could take many hours.) See Figure 29 on page 35.

If you want to explore any of the individual models further, you can double-click on a model nugget icon in the **Model** column to drill down and browse the individual model results; from there you can generate modeling nodes, model nuggets, or evaluation charts. In the **Graph** column, you can double-click on a thumbnail to generate a full-sized graph.

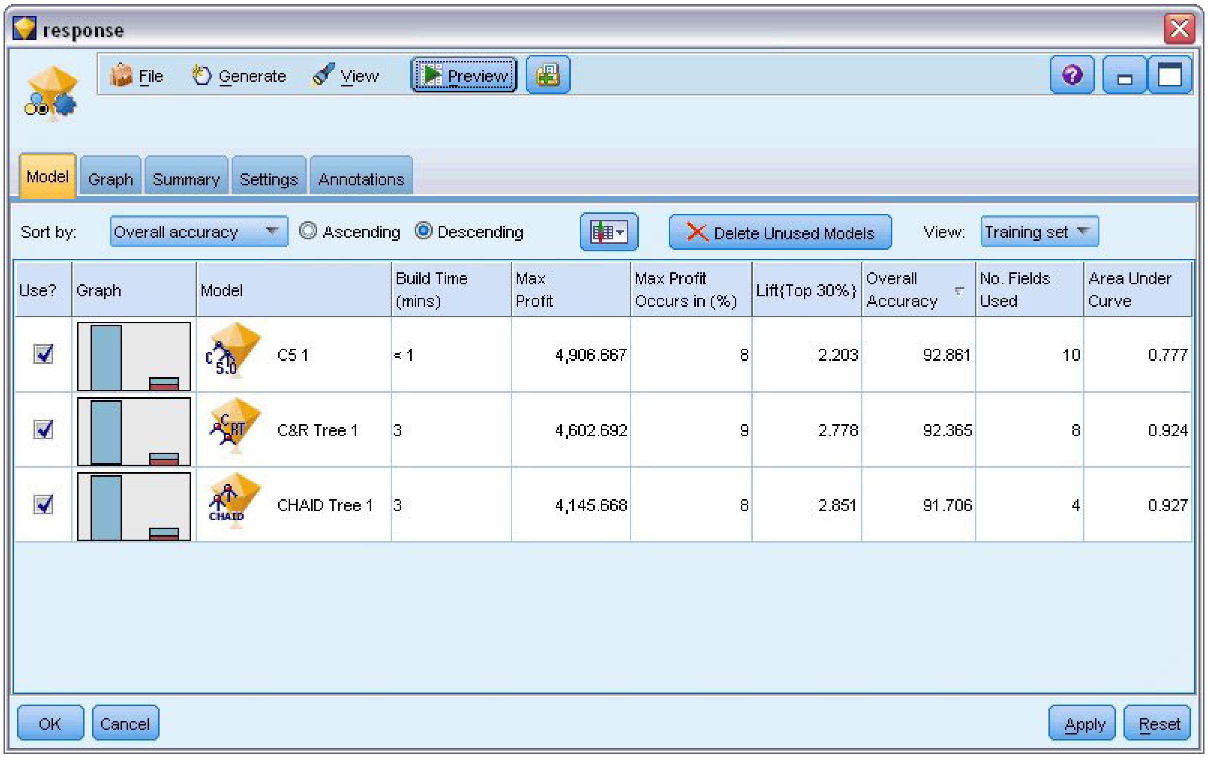


Figure 40. Auto Classifier results

By default, models are sorted based on overall accuracy, because this was the measure you selected on the Auto Classifier node Model tab. The C51 model ranks best by this measure, but the C&R Tree and CHAID models are nearly as accurate.

You can sort on a different column by clicking the header for that column, or you can choose the desired measure from the **Sort by** drop-down list on the toolbar.

Based on these results, you decide to use all three of these most accurate models. By combining predictions from multiple models, limitations in individual models may be avoided, resulting in a higher overall accuracy.

In the **Use?** column, select the C51, C&R Tree, and CHAID models.

Attach an Analysis node (Output palette) after the model nugget. Right-click on the Analysis node and choose **Run** to run the stream.

The aggregated score generated by the ensembled model is shown in a field named *$XF-response*. When measured against the training data, the predicted value matches the actual response (as recorded in the original *response* field) with an overall accuracy of 92.82%.

While not quite as accurate as the best of the three individual models in this case (92.86% for C51), the difference is too small to be meaningful. In general terms, an ensembled model will typically be more likely to perform well when applied to datasets other than the training data.

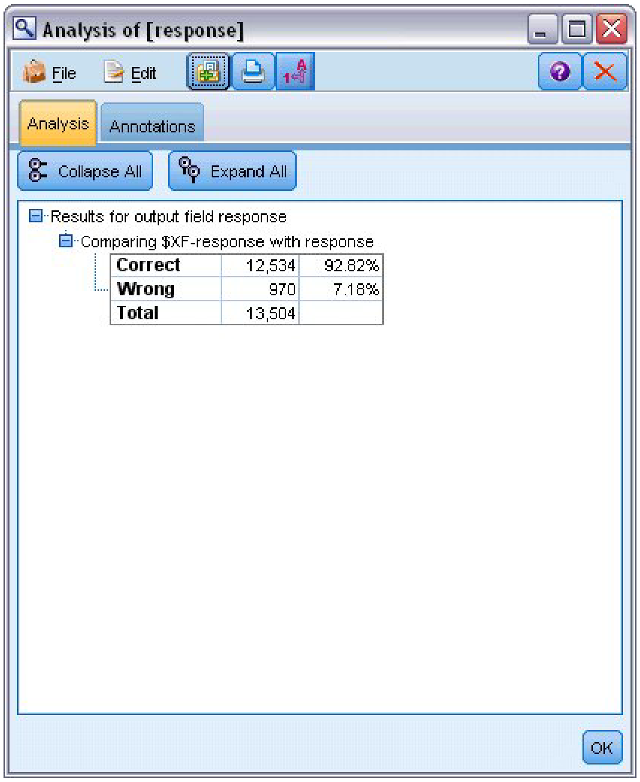


Figure 41. Analysis of the three ensembled models

Summary

To sum up, you used the Auto Classifier node to compare a number of different models, used the three most accurate models and added them to the stream within an ensembled Auto Classifier model nugget.

- Based on overall accuracy, the C51, C&R Tree, and CHAID models performed best on the training data.
- The ensembled model performed nearly as well as the best of the individual models and may perform better when applied to other datasets. If your goal is to automate the process as much as possible, this approach allows you to obtain a robust model under most circumstances without having to dig deeply into the specifics of any one model.

Chapter 5. Automated Modeling for a Continuous Target

Property Values (Auto Numeric)

The Auto Numeric node enables you to automatically create and compare different models for continuous (numeric range) outcomes, such as predicting the taxable value of a property. With a single node, you can estimate and compare a set of candidate models and generate a subset of models for further analysis. The node works in the same manner as the Auto Classifier node, but for continuous rather than flag or nominal targets.

The node combines the best of the candidate models into a single aggregated (Ensembled) model nugget. This approach combines the ease of automation with the benefits of combining multiple models, which often yield more accurate predictions than can be gained from any one model.

This example focuses on a fictional municipality responsible for adjusting and assessing real estate taxes. To do this more accurately, they will build a model that predicts property values based on building type, neighborhood, size, and other known factors.

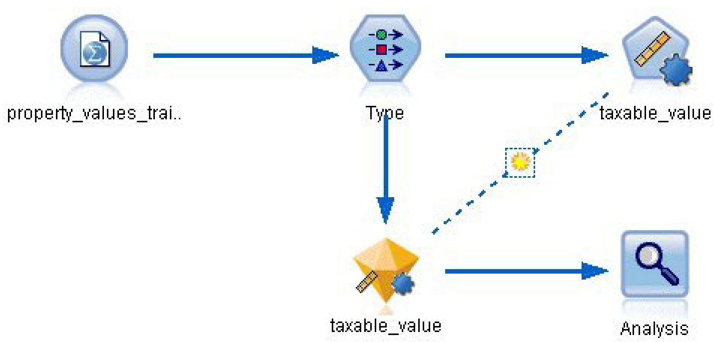


Figure 42. Auto Numeric sample stream

This example uses the stream *property_values_numericpredictor.str*, installed in the Demos folder under *streams*. The data file used is *property_values_train.sav*. See the topic “Demos Folder” on page 4 for more information.

Training Data

The data file includes a field named *taxable_value*, which is the **target field**, or value, that you want to predict. The other fields contain information such as neighborhood, building type, and interior volume and may be used as predictors.

Field name	Label
property_id	Property ID
neighborhood	Area within the city
building_type	Type of building
year_built	Year built
volume_interior	Volume of interior
volume_other	Volume of garage and extra buildings
lot_size	Lot size

Field name	Label
taxable_value	Taxable value

A scoring data file named *property_values_score.sav* is also included in the Demos folder. It contains the same fields but without the *taxable_value* field. After training models using a dataset where the taxable value is known, you can score records where this value is not yet known.

Building the Stream

1. Add a Statistics File source node pointing to *property_values_train.sav*, located in the *Demos* folder of your IBM SPSS Modeler installation. (You can specify `$CLEO_DEMOS/` in the file path as a shortcut to reference this folder. Note that a forward slash—rather than a backslash—must be used in the path, as shown.)

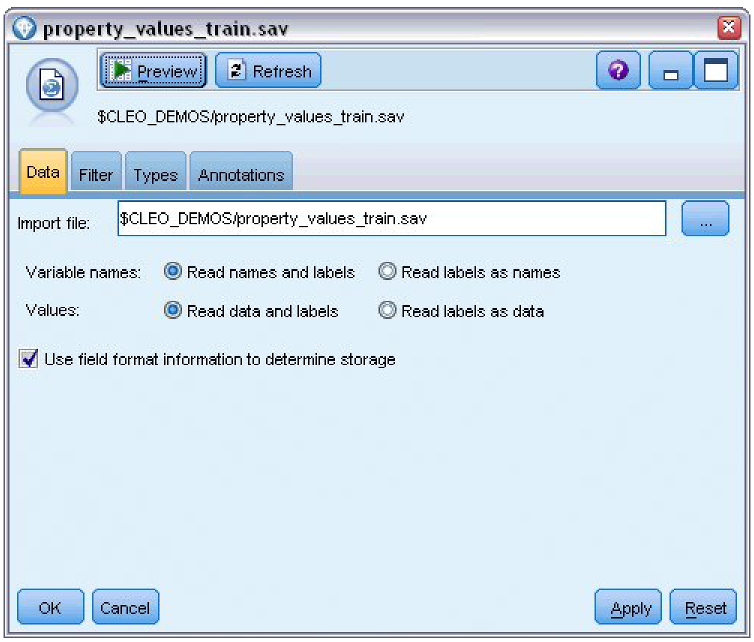


Figure 43. Reading in the data

2. Add a Type node, and select *taxable_value* as the target field (Role = **Target**). Role should be set to **Input** for all other fields, indicating that they will be used as predictors.

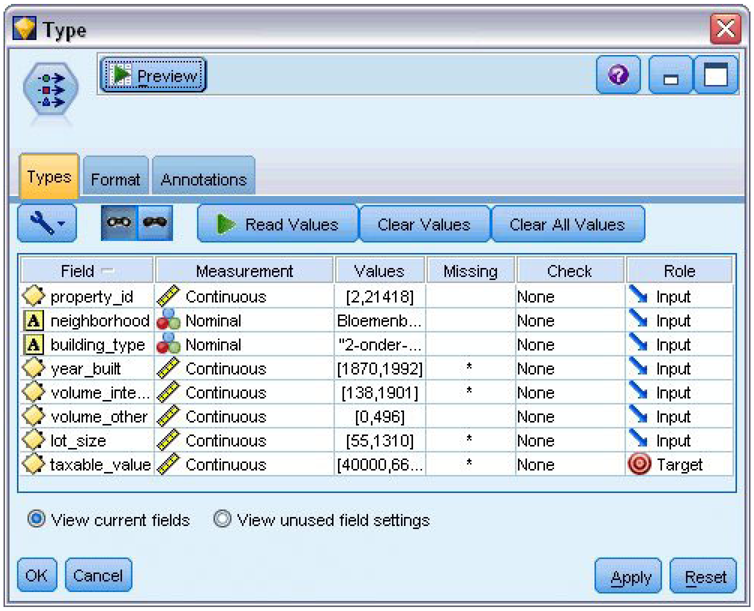


Figure 44. Setting the target field

3. Attach an Auto Numeric node, and select **Correlation** as the metric used to rank models.
4. Set the **Number of models to use** to 3. This means that the three best models will be built when you execute the node.

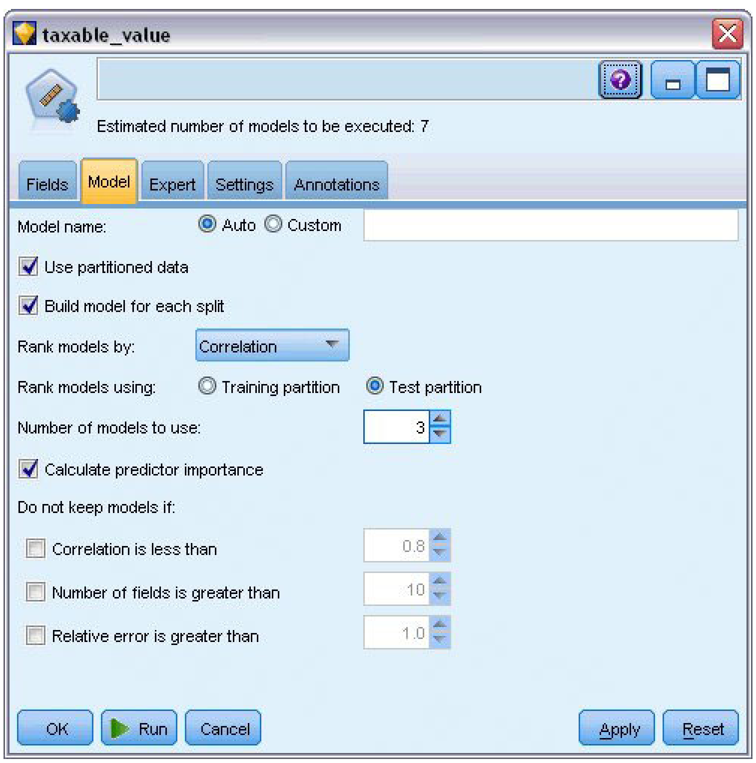


Figure 45. Auto Numeric node Model tab

5. On the Expert tab, leave the default settings in place; the node will estimate a single model for each algorithm, for a total of seven models. (Alternatively, you can modify these settings to compare multiple variants for each model type.)

Because you set **Number of models to use** to 3 on the Model tab, the node will calculate the accuracy of the seven algorithms and build a single model nugget containing the three most accurate.

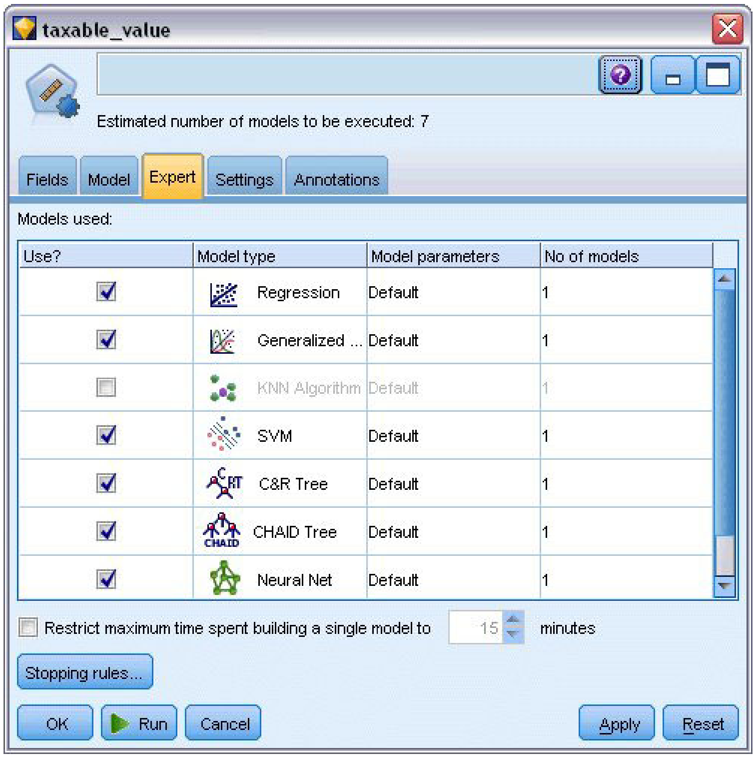


Figure 46. Auto Numeric node Expert tab

6. On the Settings tab, leave the default settings in place. Since this is a continuous target, the ensemble score is generated by averaging the scores for the individual models.

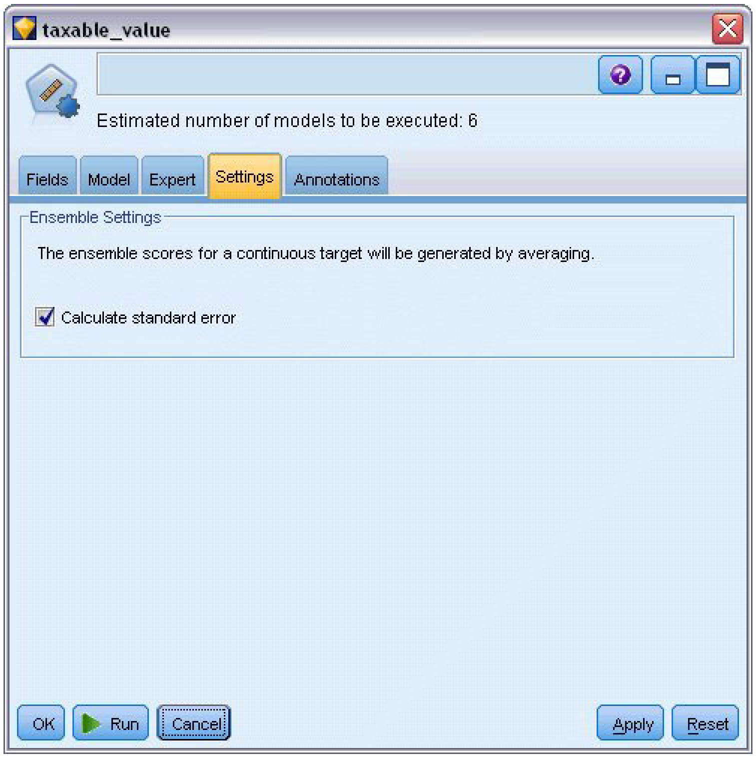


Figure 47. Auto Numeric node Settings tab

Comparing the Models

1. Click the Run button.

The model nugget is built and placed on the canvas, and also on the Models palette in the upper right corner of the window. You can browse the nugget, or save or deploy it in a number of other ways.

Open the model nugget; it lists details about each of the models created during the run. (In a real situation, in which hundreds of models are estimated on a large dataset, this could take many hours.) See Figure 42 on page 47.

If you want to explore any of the individual models further, you can double-click on a model nugget icon in the **Model** column to drill down and browse the individual model results; from there you can generate modeling nodes, model nuggets, or evaluation charts.

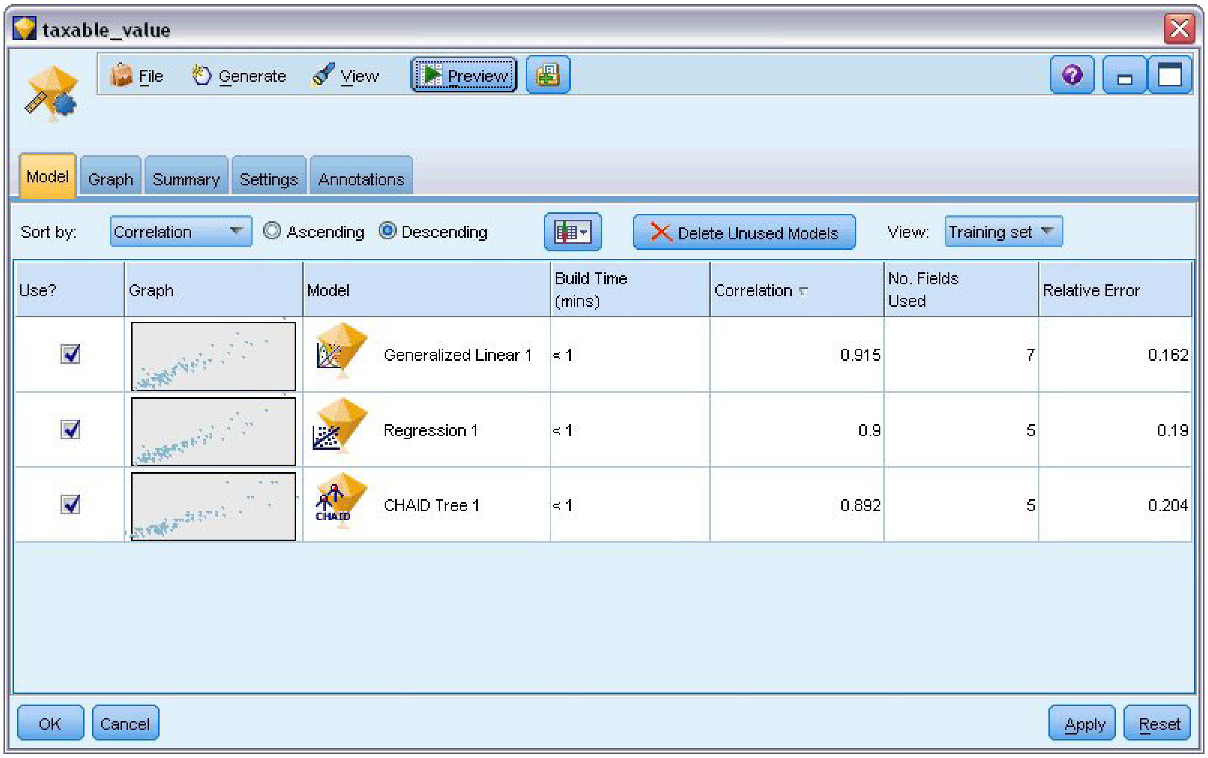


Figure 48. Auto Numeric results

By default, models are sorted by correlation because this was the measure you selected in the Auto Numeric node. For purposes of ranking, the absolute value of the correlation is used, with values closer to 1 indicating a stronger relationship. The Generalized Linear model ranks best on this measure, but several others are nearly as accurate. The Generalized Linear model also has the lowest relative error.

You can sort on a different column by clicking the header for that column, or you can choose the desired measure from the **Sort by** list on the toolbar.

Each graph displays a plot of observed values against predicted values for the model, providing a quick visual indication of the correlation between them. For a good model, points should cluster along the diagonal, which is true for all the models in this example.

In the **Graph** column, you can double-click on a thumbnail to generate a full-sized graph.

Based on these results, you decide to use all three of these most accurate models. By combining predictions from multiple models, limitations in individual models may be avoided, resulting in a higher overall accuracy.

In the **Use?** column, ensure that all three models are selected.

Attach an Analysis node (Output palette) after the model nugget. Right-click on the Analysis node and choose **Run** to run the stream.

The averaged score generated by the ensembled model is added in a field named *$XR-taxable_value*, with a correlation of 0.922, which is higher than those of the three individual models. The ensemble scores also show a low mean absolute error and may perform better than any of the individual models when applied to other datasets.

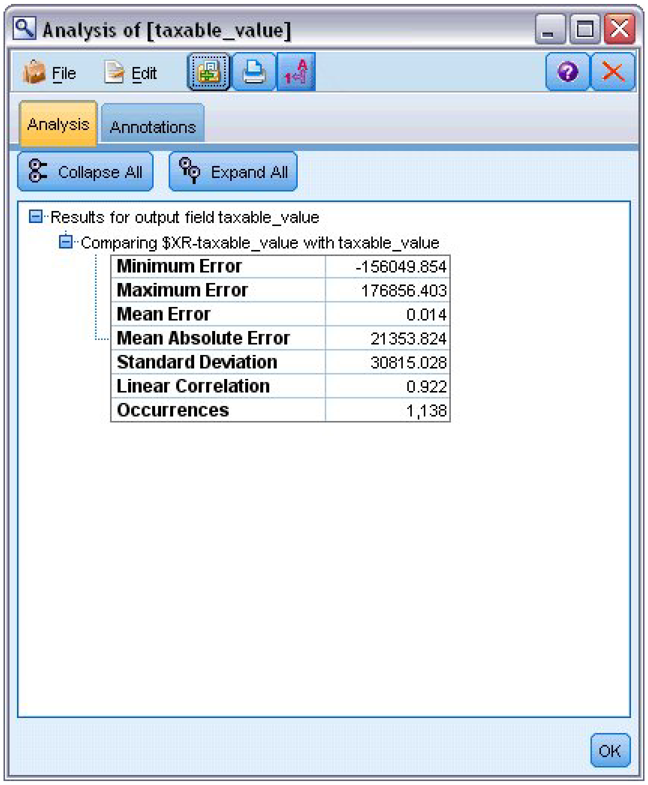


Figure 49. Auto Numeric sample stream

Summary

To sum up, you used the Auto Numeric node to compare a number of different models, selected the three most accurate models and added them to the stream within an ensembled Auto Numeric model nugget.

- Based on overall accuracy, the Generalized Linear, Regression, and CHAID models performed best on the training data.
- The ensembled model showed performance that was better than two of the individual models and may perform better when applied to other datasets. If your goal is to automate the process as much as possible, this approach allows you to obtain a robust model under most circumstances without having to dig deeply into the specifics of any one model.

Chapter 6. Automated Data Preparation (ADP)

Preparing data for analysis is one of the most important steps in any data-mining project—and traditionally, one of the most time consuming. The Automated Data Preparation (ADP) node handles the task for you, analyzing your data and identifying fixes, screening out fields that are problematic or not likely to be useful, deriving new attributes when appropriate, and improving performance through intelligent screening techniques. You can use the node in fully automated fashion, allowing the node to choose and apply fixes, or you can preview the changes before they are made and accept or reject them as desired.

Using the ADP node enables you to make your data ready for data mining quickly and easily, without needing to have prior knowledge of the statistical concepts involved. If you run the node with the default settings, models will tend to build and score more quickly.

This example uses the stream named *ADP_basic_demo.str*, which references the data file named *telco.sav* to demonstrate the increased accuracy that may be found by using the default ADP node settings when building models. These files are available from the *Demos* directory of any IBM SPSS Modeler installation. This can be accessed from the IBM SPSS Modeler program group on the Windows Start menu. The *ADP_basic_demo.str* file is in the *streams* directory.

Building the Stream

1. To build the stream, add a Statistics File source node pointing to *telco.sav* located in the *Demos* directory of your IBM SPSS Modeler installation.

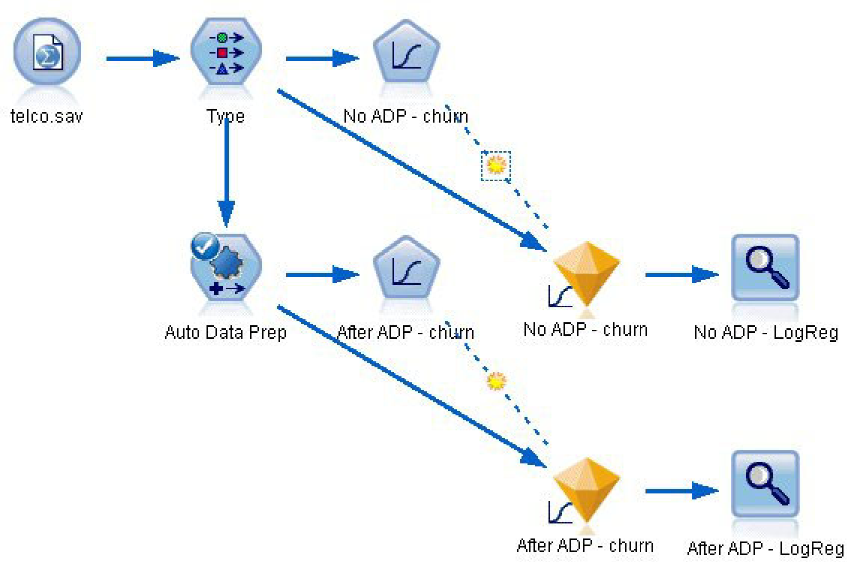


Figure 50. Building the stream

2. Attach a Type node to the source node, set the measurement level for the *churn* field to **Flag**, and set the role to **Target**. All other fields should have their role set to **Input**.

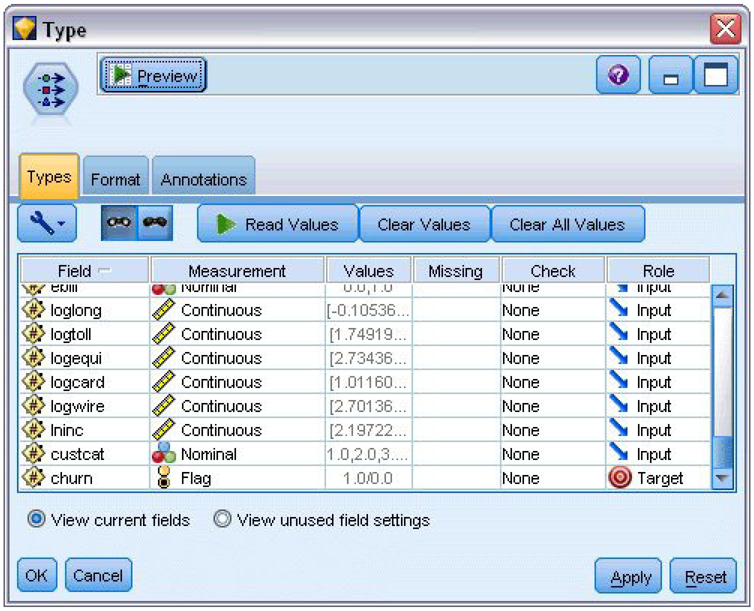


Figure 51. Selecting the target

3. Attach a Logistic node to the Type node.
4. In the Logistic node, click the Model tab and select the **Binomial** procedure. In the *Model name* field, select **Custom** and enter `No ADP - churn`.

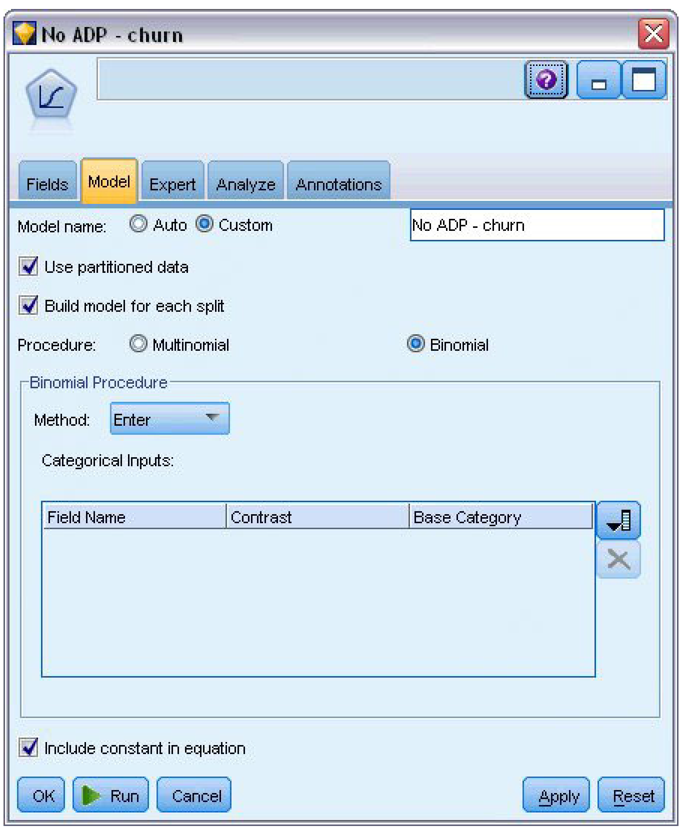


Figure 52. Choosing model options

5. Attach an ADP node to the Type node. On the Objectives tab, leave the default settings in place to analyze and prepare your data by balancing both speed and accuracy.
6. At the top of the Objectives tab, click **Analyze Data** to analyze and process your data.

   Other options on the ADP node enable you to specify that you want to concentrate more on accuracy, more on the speed of processing, or to fine tune many of the data preparation processing steps.

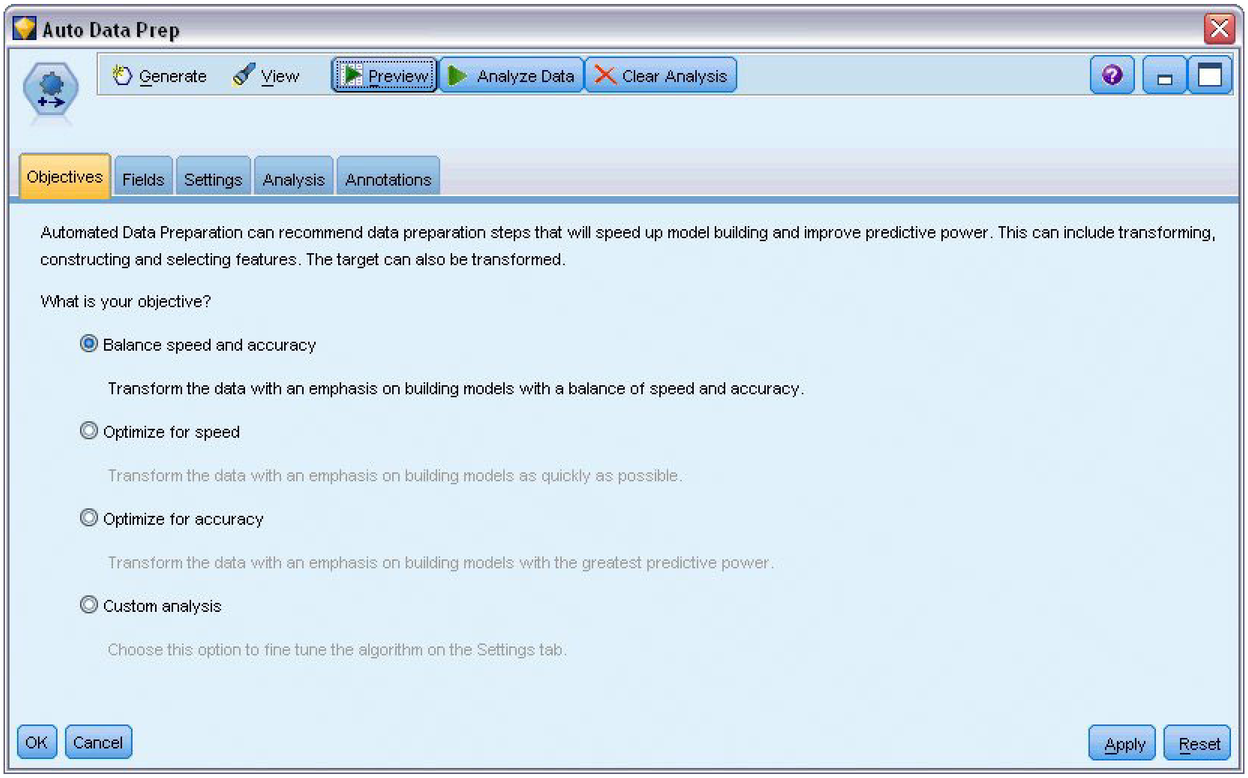


Figure 53. ADP default objectives

The results of the data processing are displayed on the Analysis tab. The **Field Processing Summary** shows that of the 41 data features brought in to the ADP node, 19 have been transformed to aid processing, and 3 have been discarded as unused.

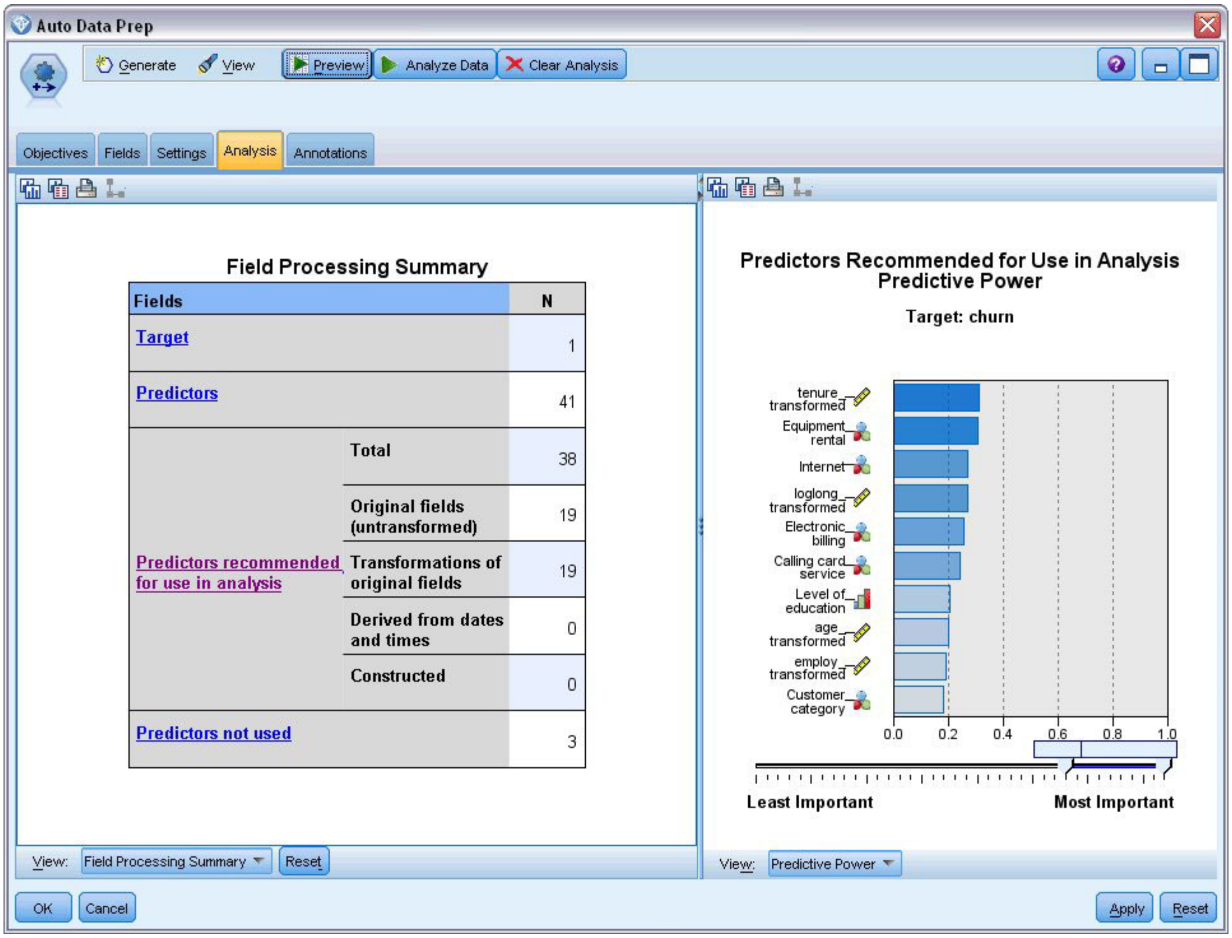


Figure 54. Summary of data processing

7. Attach a Logistic node to the ADP node.
8. In the Logistic node, click the Model tab and select the **Binomial** procedure. In the *Modeling name* field, select **Custom** and enter `After ADP - churn`.

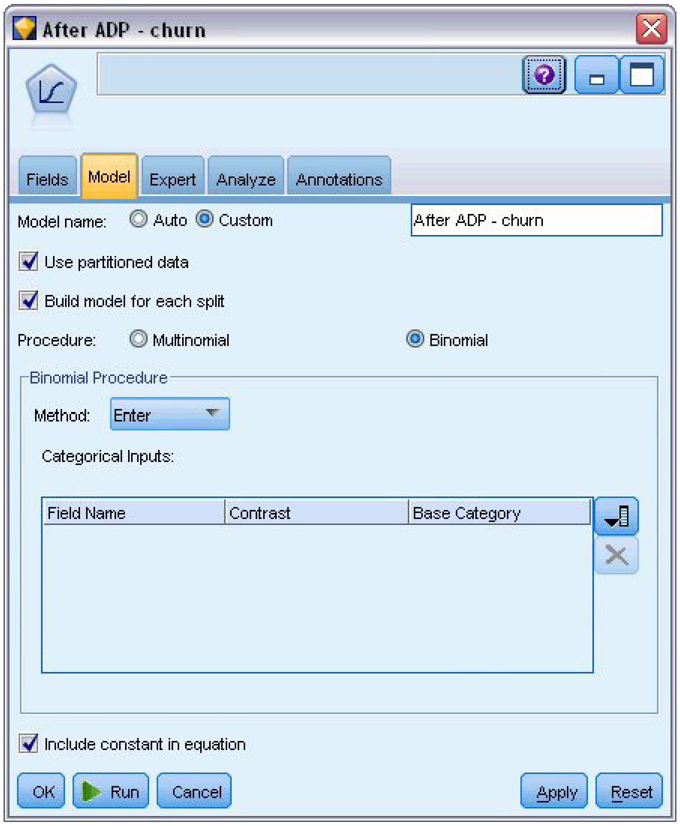


Figure 55. Choosing model options

Comparing Model Accuracy

1. Run both Logistic nodes to create the model nuggets, which are added to the stream and to the Models palette in the upper-right corner.

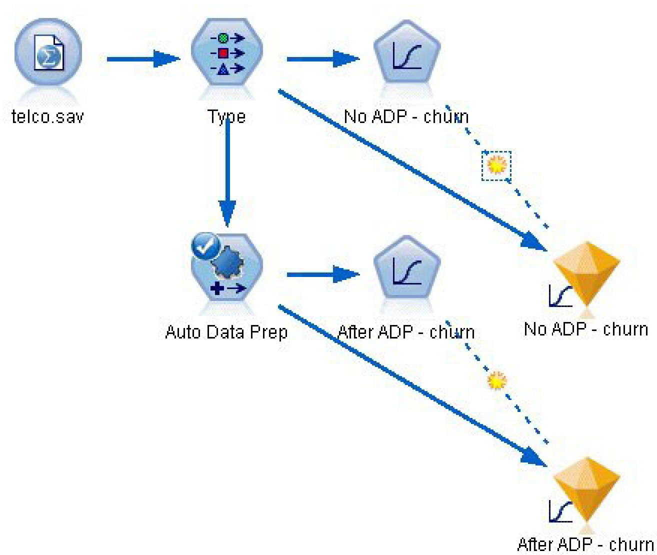


Figure 56. Attaching the model nuggets

2. Attach Analysis nodes to the model nuggets and run the Analysis nodes using their default settings.

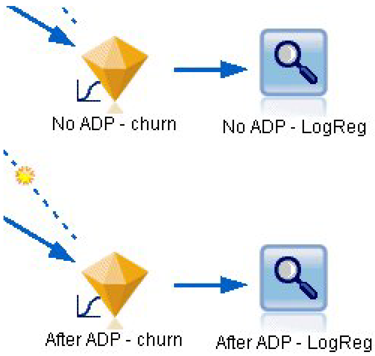


Figure 57. Attaching the Analysis nodes

The Analysis of the non ADP-derived model shows that just running the data through the Logistic Regression node with its default settings gives a model with low accuracy - just 10.6%.

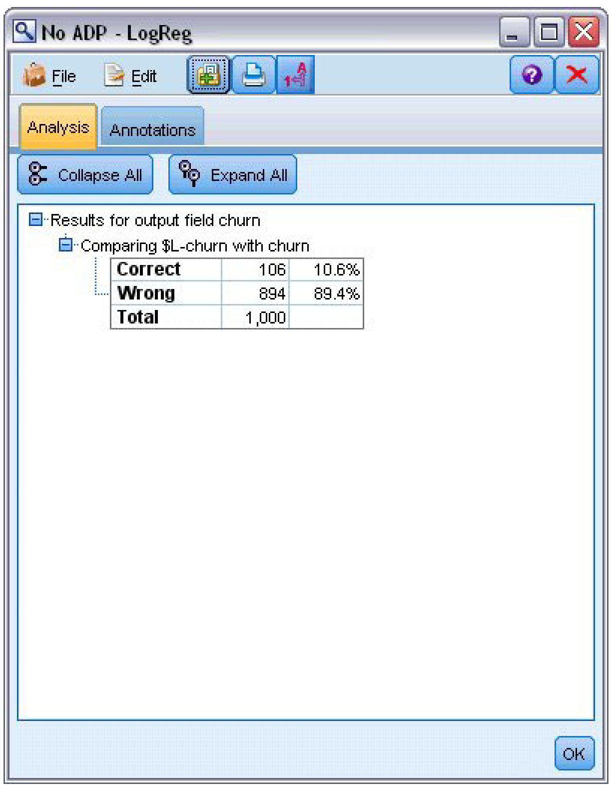


Figure 58. Non ADP-derived model results

The Analysis of the ADP-derived model shows that running the data through the default ADP settings, you have built a much more accurate model that is 78.8% correct.

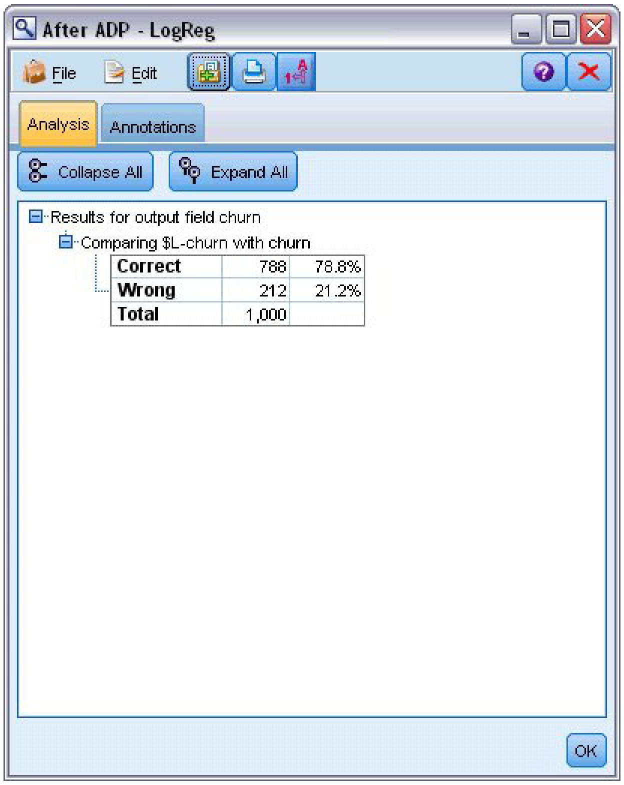


Figure 59. ADP-derived model results

In summary, by just running the ADP node to fine tune the processing of your data, you were able to build a more accurate model with little direct data manipulation.

Obviously, if you are interested in proving or disproving a certain theory, or want to build specific models, you may find it beneficial to work directly with the model settings; however, for those with a reduced amount of time, or with a large amount of data to prepare, the ADP node may give you an advantage.

Explanations of the mathematical foundations of the modeling methods used in IBM SPSS Modeler are listed in the *IBM SPSS Modeler Algorithms Guide*, available from the *\Documentation* directory of the installation disk.

Note that the results in this example are based on the training data only. To assess how well models generalize to other data in the real world, you would use a Partition node to hold out a subset of records for purposes of testing and validation.

Chapter 7. Preparing Data for Analysis (Data Audit)

The Data Audit node provides a comprehensive first look at the data you bring into IBM SPSS Modeler. Often used during the initial data exploration, the data audit report shows summary statistics as well as histograms and distribution graphs for each data field, and it allows you to specify treatments for missing values, outliers, and extreme values.

This example uses the stream named *telco_dataaudit.str*, which references the data file named *telco.sav*. These files are available from the *Demos* directory of any IBM SPSS Modeler installation. This can be accessed from the IBM SPSS Modeler program group on the Windows Start menu. The *telco_dataaudit.str* file is in the *streams* directory.

Building the Stream

1. To build the stream, add a Statistics File source node pointing to *telco.sav* located in the *Demos* directory of your IBM SPSS Modeler installation.

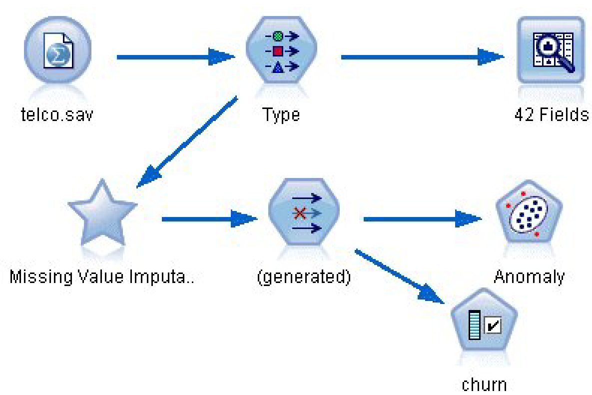


Figure 60. Building the stream

2. Add a Type node to define fields, and specify *churn* as the target field (Role = **Target**). Role should be set to **Input** for all of the other fields so that this is the only target.

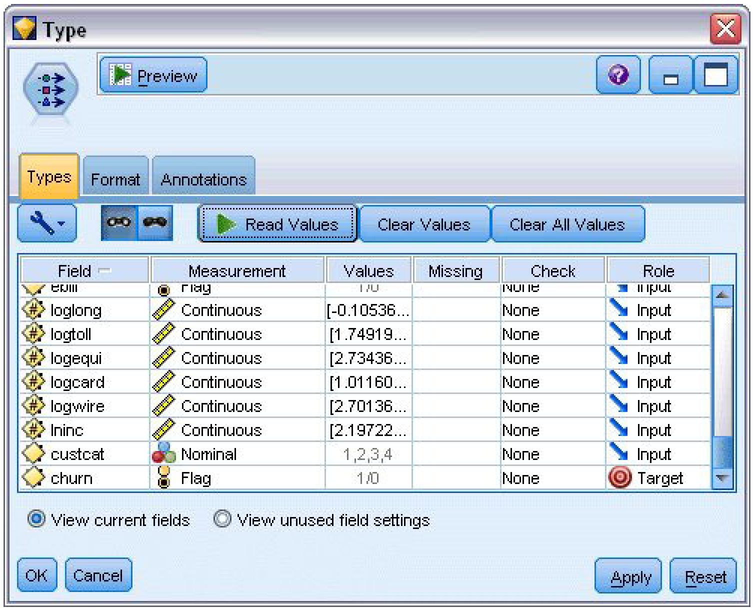


Figure 61. Setting the target

3. Confirm that field measurement levels are defined correctly. For example, most fields with values 0 and 1 can be regarded as flags, but certain fields, such as gender, are more accurately viewed as a nominal field with two values.

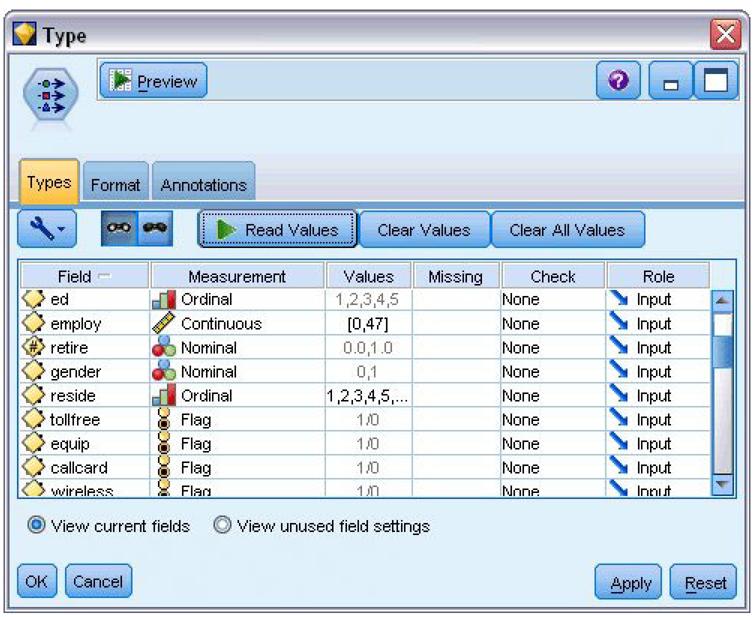


Figure 62. Setting measurement levels

Tip: To change properties for multiple fields with similar values (such as 0/1), click the *Values* column header to sort fields by that column, and use the Shift key to select all of the fields you want to change. You can then right-click on the selection to change the measurement level or other attributes for all selected fields.

4. Attach a Data Audit node to the stream. On the Settings tab, leave the default settings in place to include all fields in the report. Since *churn* is the only target field defined in the Type node, it will automatically be used as an overlay.

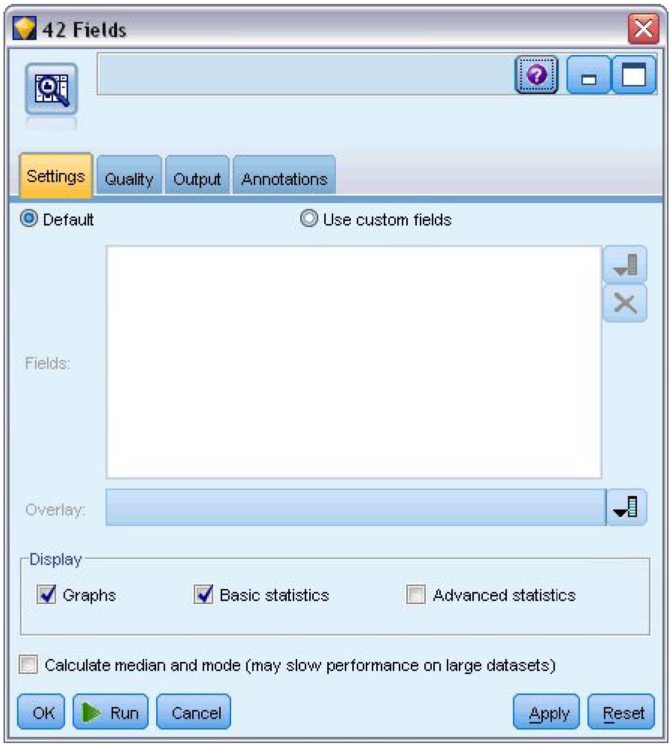


Figure 63. Data Audit node, Settings tab

On the Quality tab, leave the default settings for detecting missing values, outliers, and extreme values in place, and click **Run**.

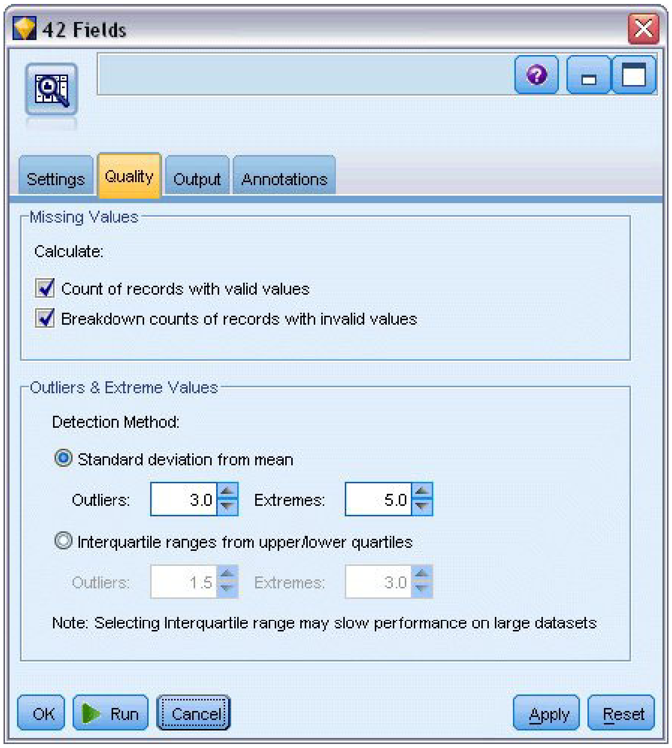


Figure 64. Data Audit node, Quality tab

Browsing Statistics and Charts

The Data Audit browser is displayed, with thumbnail graphs and descriptive statistics for each field.

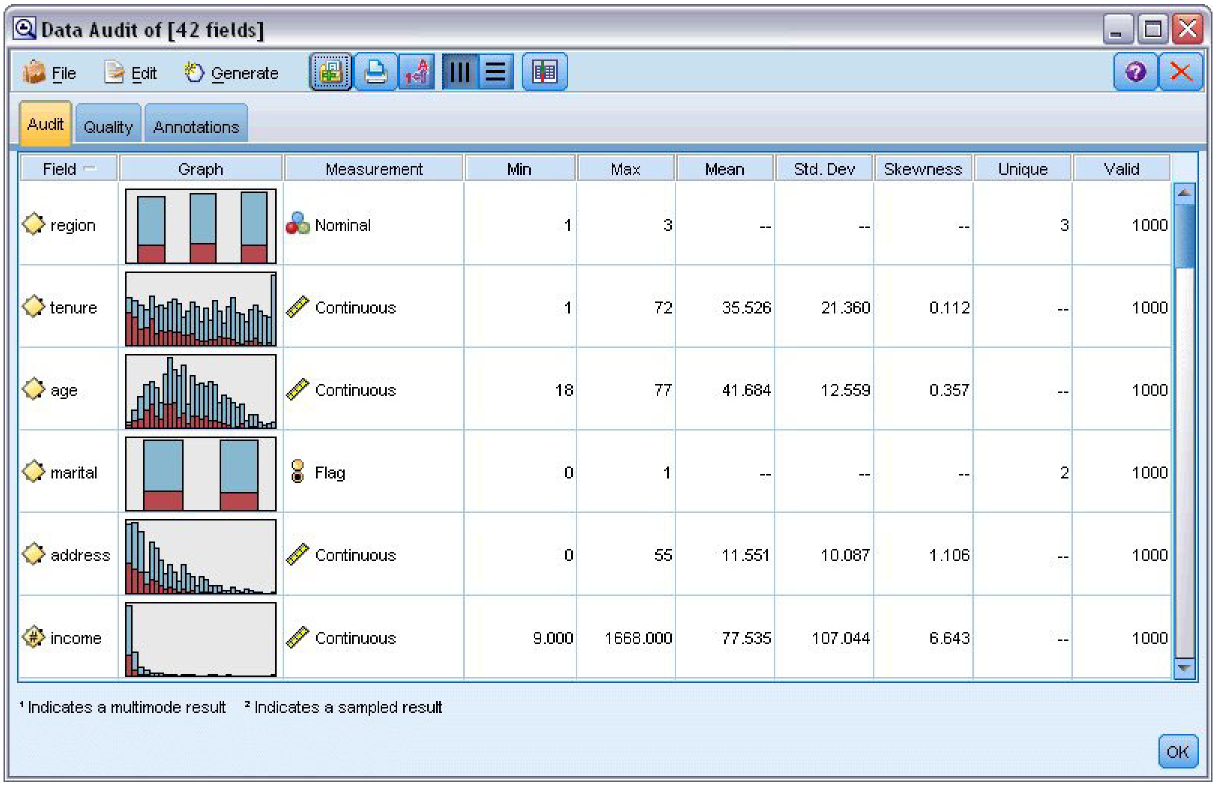


Figure 65. Data Audit browser

Use the toolbar to display field and value labels, and to toggle the alignment of charts from horizontal to vertical (for categorical fields only).

1. You can also use the toolbar or Edit menu to choose the statistics to display.

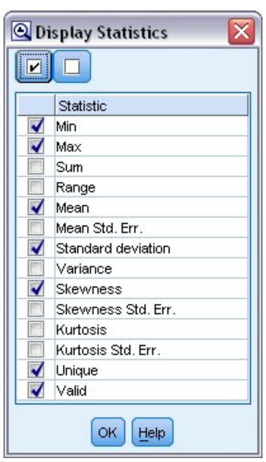


Figure 66. Display Statistics

Double-click on any thumbnail graph in the audit report to view a full-sized version of that chart. Because *churn* is the only target field in the stream, it is automatically used as an overlay. You can toggle the display of field and value labels using the graph window toolbar, or click the Edit mode button to further customize the chart.

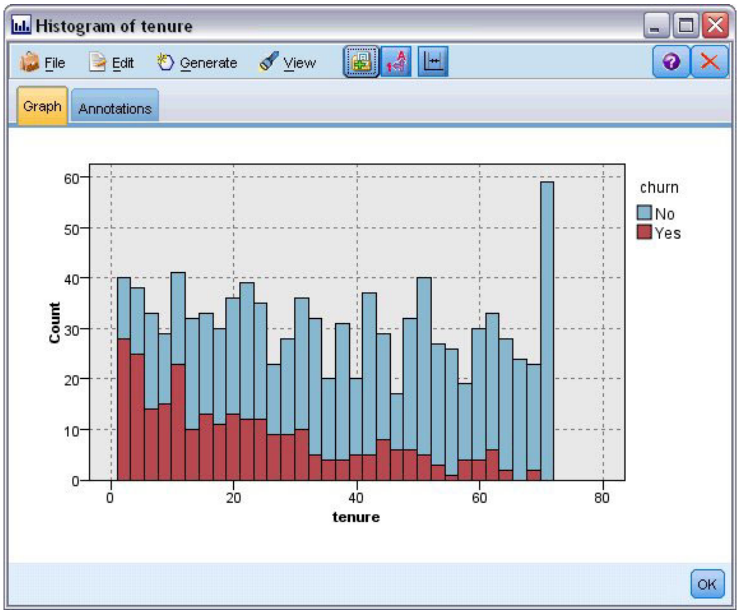


Figure 67. Histogram of tenure

Alternatively, you can select one or more thumbnails and generate a Graph node for each. The generated nodes are placed on the stream canvas and can be added to the stream to re-create that particular graph.

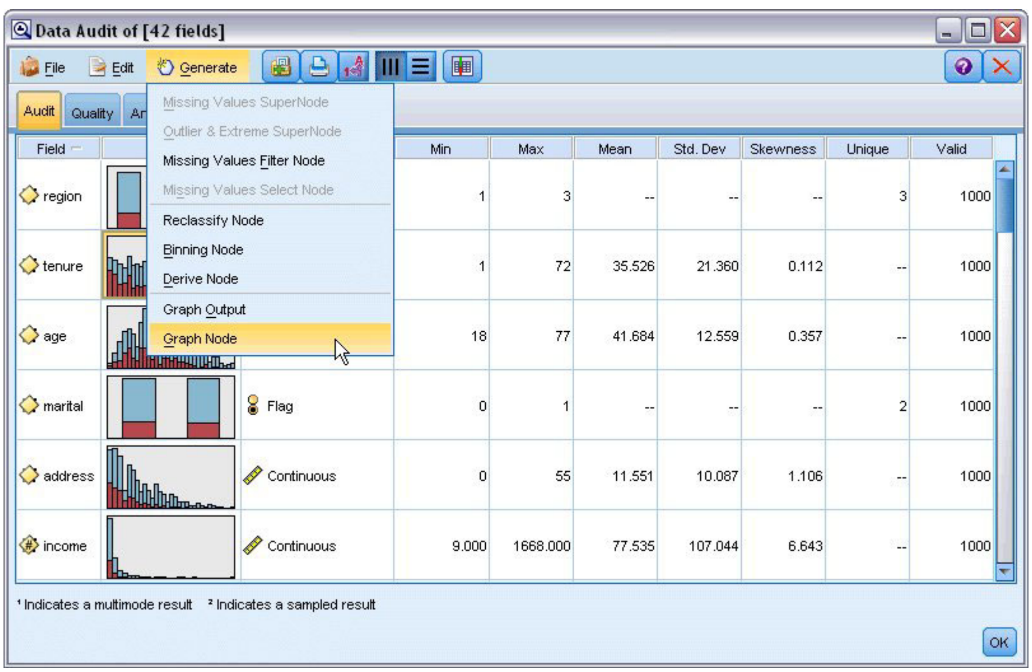


Figure 68. Generating a Graph node

Handling Outliers and Missing Values

The Quality tab in the audit report displays information about outliers, extremes, and missing values.

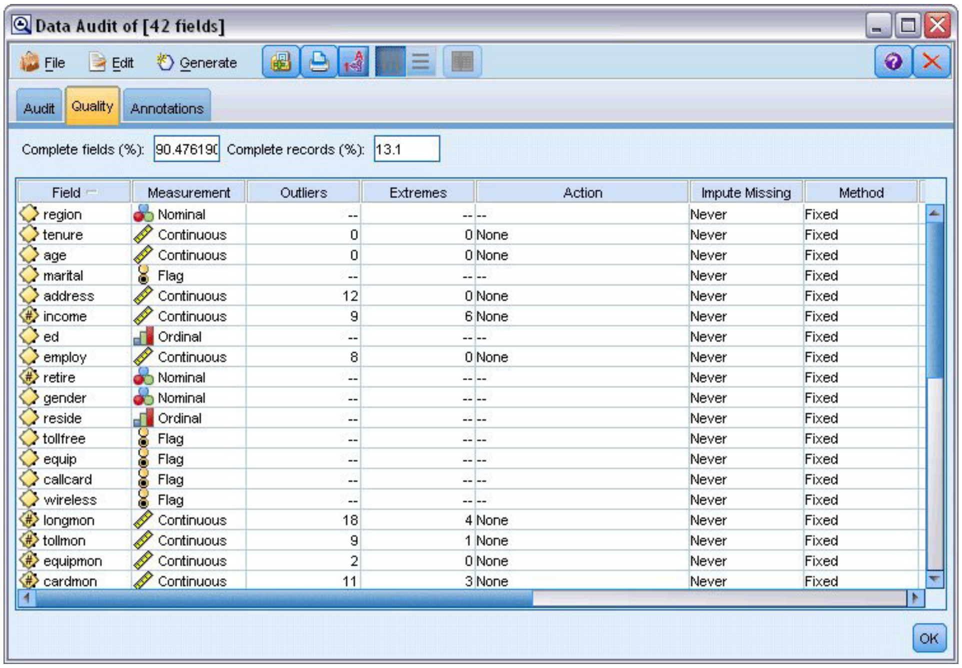


Figure 69. Data Audit browser, Quality tab

You can also specify methods for handling these values and generate SuperNodes to automatically apply the transformations. For example you can select one or more fields and choose to impute or replace missing values for these fields using a number of methods, including the C&RT algorithm.

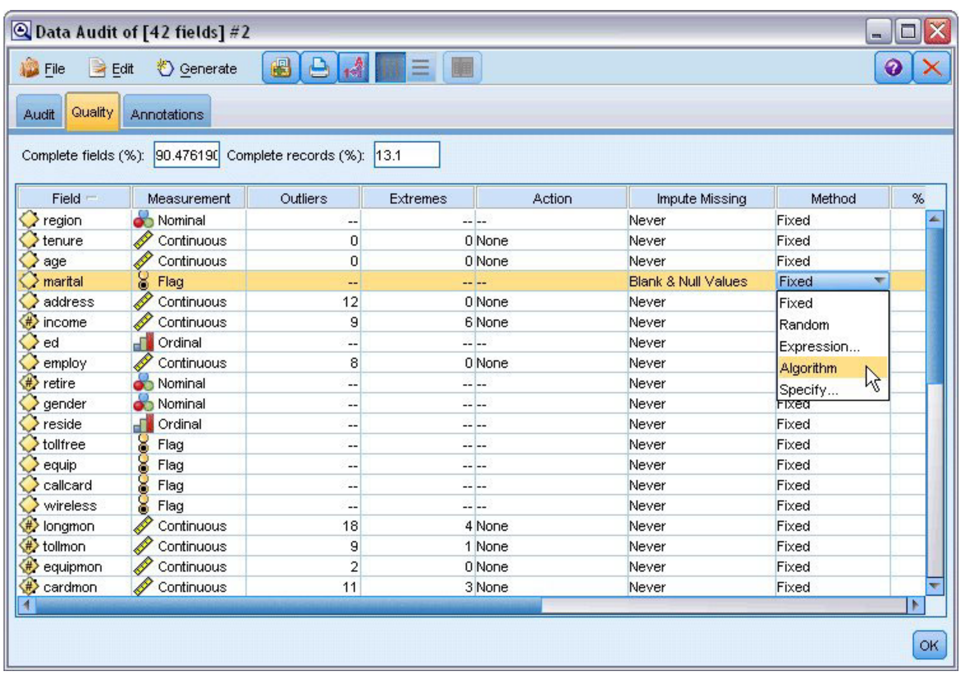


Figure 70. Choosing an impute method

After specifying an impute method for one or more fields, to generate a Missing Values SuperNode, from the menus choose:

Generate > Missing Values SuperNode

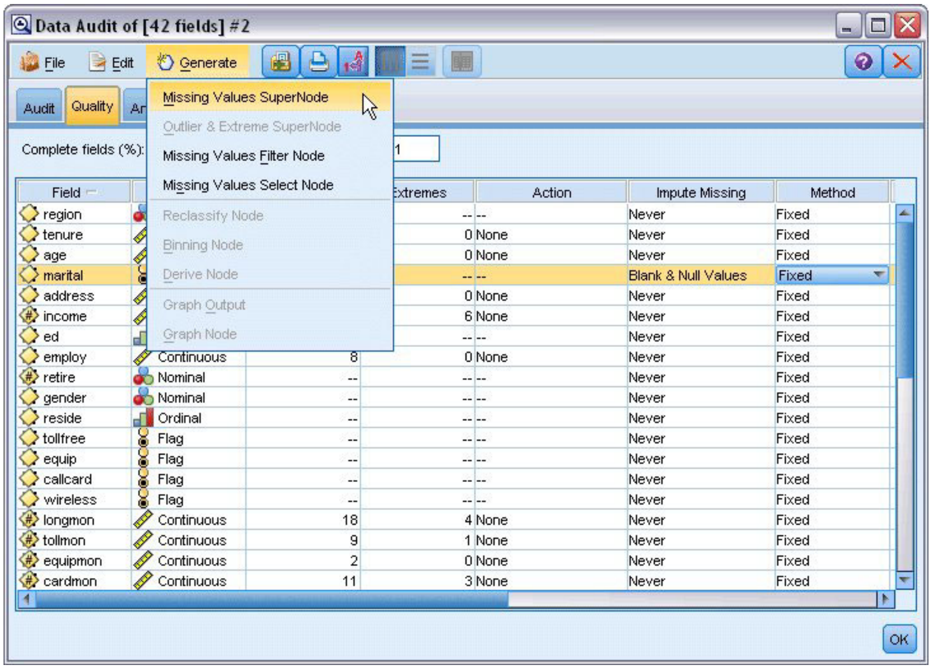


Figure 71. Generating the SuperNode

The generated SuperNode is added to the stream canvas, where you can attach it to the stream to apply the transformations.

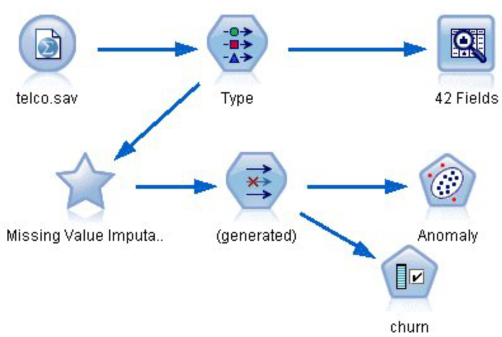


Figure 72. Stream with Missing Values SuperNode

The SuperNode actually contains a series of nodes that perform the requested transformations. To understand how it works, you can edit the SuperNode and click **Zoom In**.

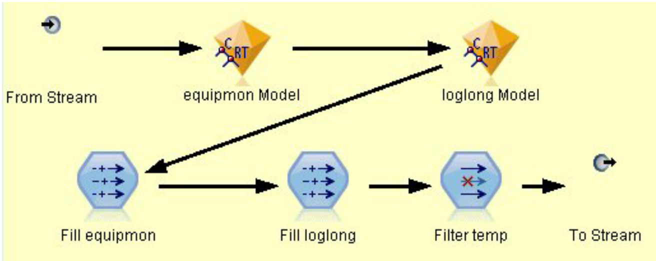


Figure 73. Zooming in on the SuperNode

For each field imputed using the algorithm method, for example, there will be a separate C&RT model, along with a Filler node that replaces blanks and nulls with the value predicted by the model. You can add, edit, or remove specific nodes within the SuperNode to further customize the behavior.

Alternatively, you can generate a Select or Filter node to remove fields or records with missing values. For example, you can filter any fields with a quality percentage below a specified threshold.

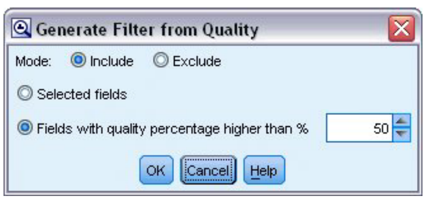


Figure 74. Generating a Filter node

Outliers and extreme values can be handled in a similar manner. Specify the action you want to take for each field—either coerce, discard, or nullify—and generate a SuperNode to apply the transformations.

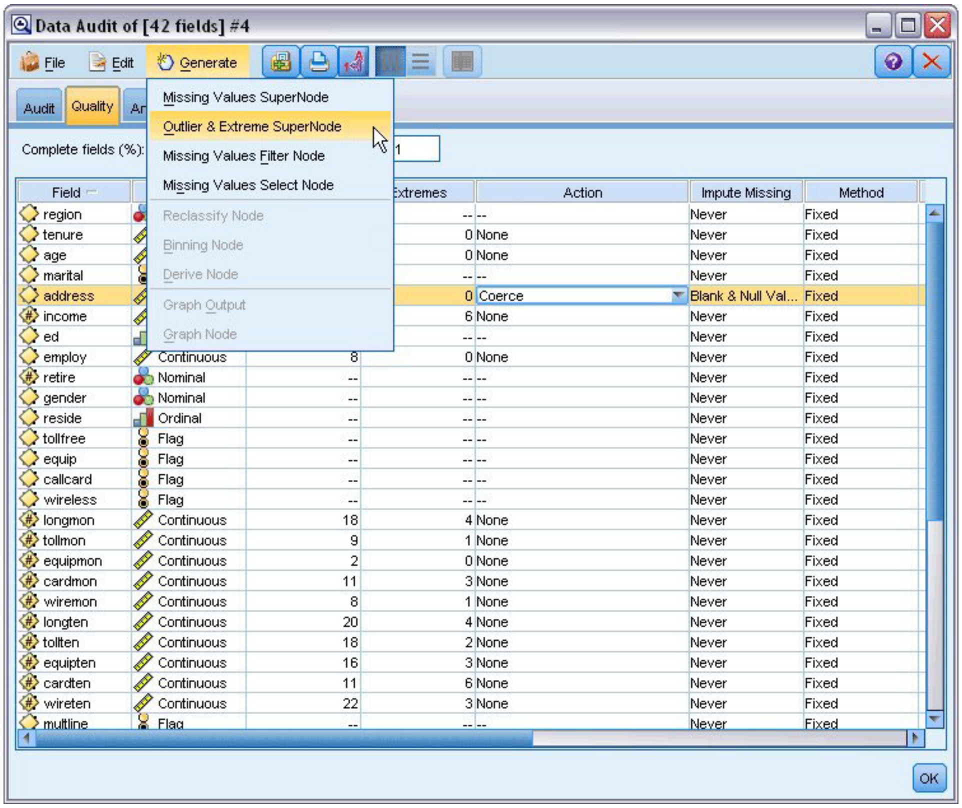


Figure 75. Generating a Filter node

After completing the audit and adding the generated nodes to the stream, you can proceed with your analysis. Optionally, you may want to further screen your data using Anomaly Detection, Feature Selection, or a number of other methods.

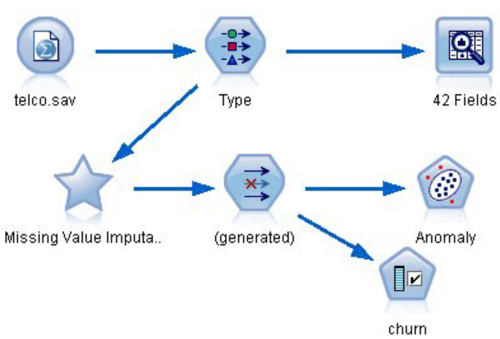


Figure 76. Stream with Missing Values SuperNode

Chapter 8. Drug Treatments (Exploratory Graphs/C5.0)

For this section, imagine that you are a medical researcher compiling data for a study. You have collected data about a set of patients, all of whom suffered from the same illness. During their course of treatment, each patient responded to one of five medications. Part of your job is to use data mining to find out which drug might be appropriate for a future patient with the same illness.

This example uses the stream named *druglearn.str*, which references the data file named *DRUG1n*. These files are available from the *Demos* directory of any IBM SPSS Modeler installation. This can be accessed from the IBM SPSS Modeler program group on the Windows Start menu. The *druglearn.str* file is in the *streams* directory.

The data fields used in the demo are:

Data field	Description
Age	(Number)
Sex	*M* or *F*
BP	Blood pressure: *HIGH*, *NORMAL*, or *LOW*
Cholesterol	Blood cholesterol: *NORMAL* or *HIGH*
Na	Blood sodium concentration
K	Blood potassium concentration
Drug	Prescription drug to which a patient responded

Reading in Text Data

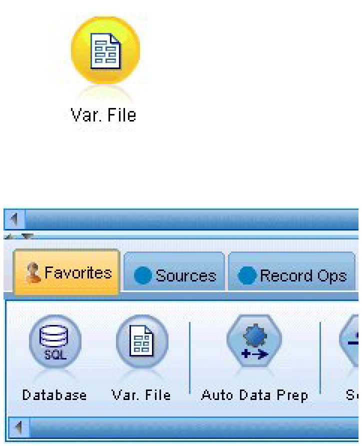


Figure 77. Adding a Variable File node

You can read in delimited text data using a **Variable File node**. You can add a Variable File node from the palettes--either click the **Sources** tab to find the node or use the **Favorites** tab, which includes this node by default. Next, double-click the newly placed node to open its dialog box.

Click the button just to the right of the File box marked with an ellipsis (...) to browse to the directory in which IBM SPSS Modeler is installed on your system. Open the *Demos* directory and select the file called *DRUG1n*.

Ensuring that **Read field names from file** is selected, notice the fields and values that have just been loaded into the dialog box.

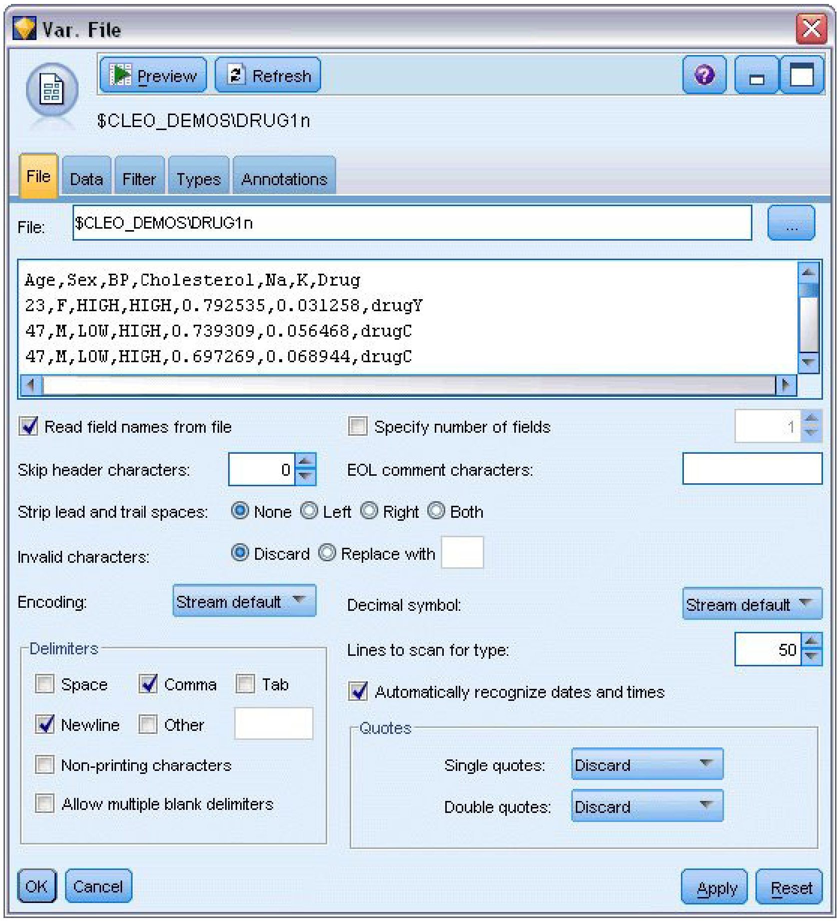


Figure 78. Variable File dialog box

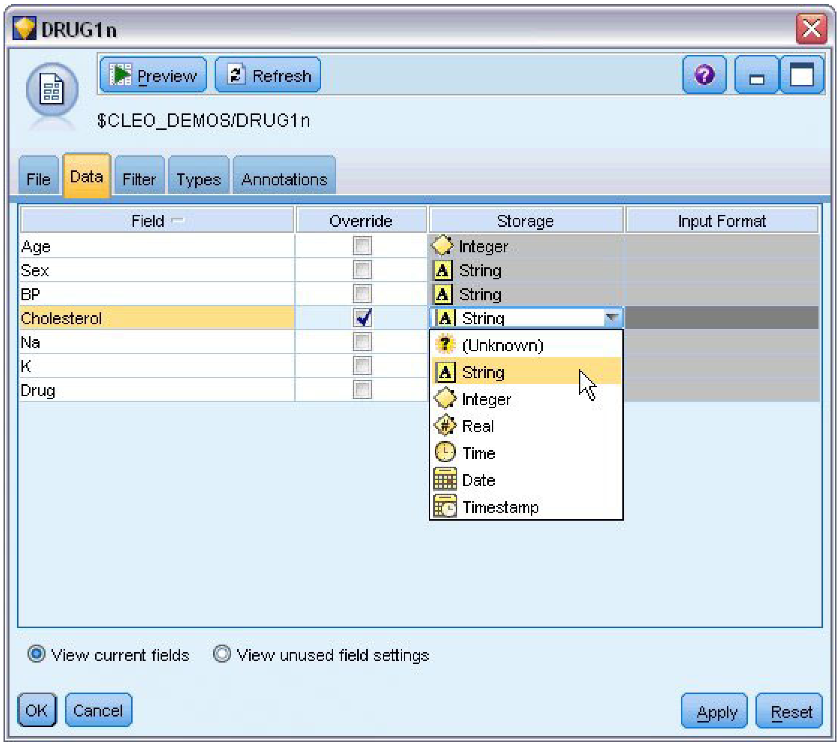


Figure 79. Changing the storage type for a field

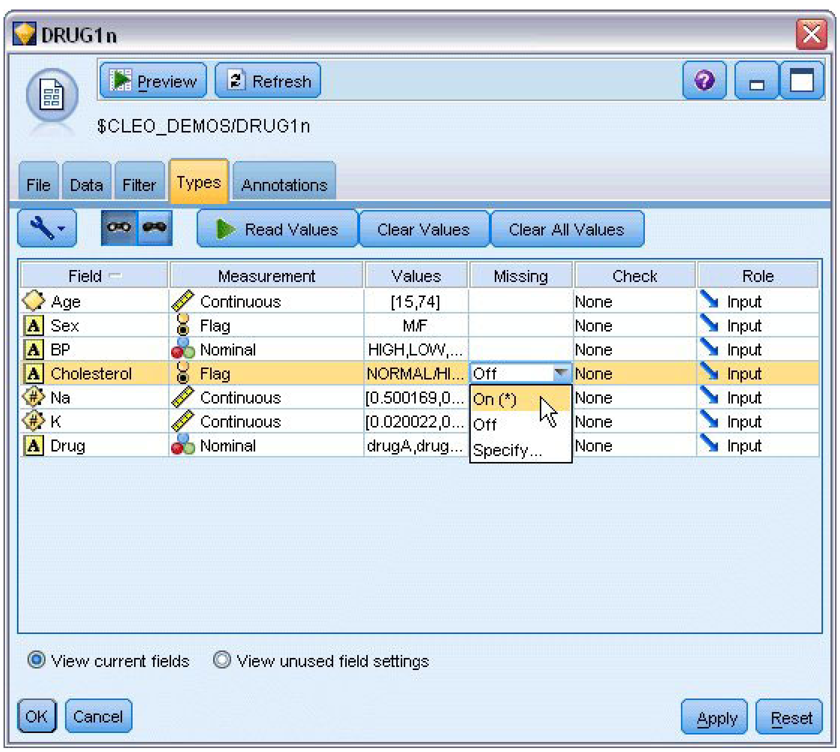


Figure 80. Selecting Value options on the Types tab

Click the **Data** tab to override and change **Storage** for a field. Note that storage is different from **Measurement**, that is, the measurement level (or usage type) of the data field. The **Types** tab helps you learn more about the type of fields in your data. You can also choose **Read Values** to view the actual values for each field based on the selections that you make from the *Values* column. This process is known as **instantiation**.

Adding a Table

Now that you have loaded the data file, you may want to glance at the values for some of the records. One way to do this is by building a stream that includes a Table node. To place a Table node in the stream, either double-click the icon in the palette or drag and drop it on to the canvas.

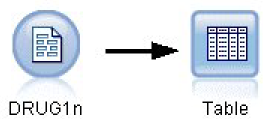
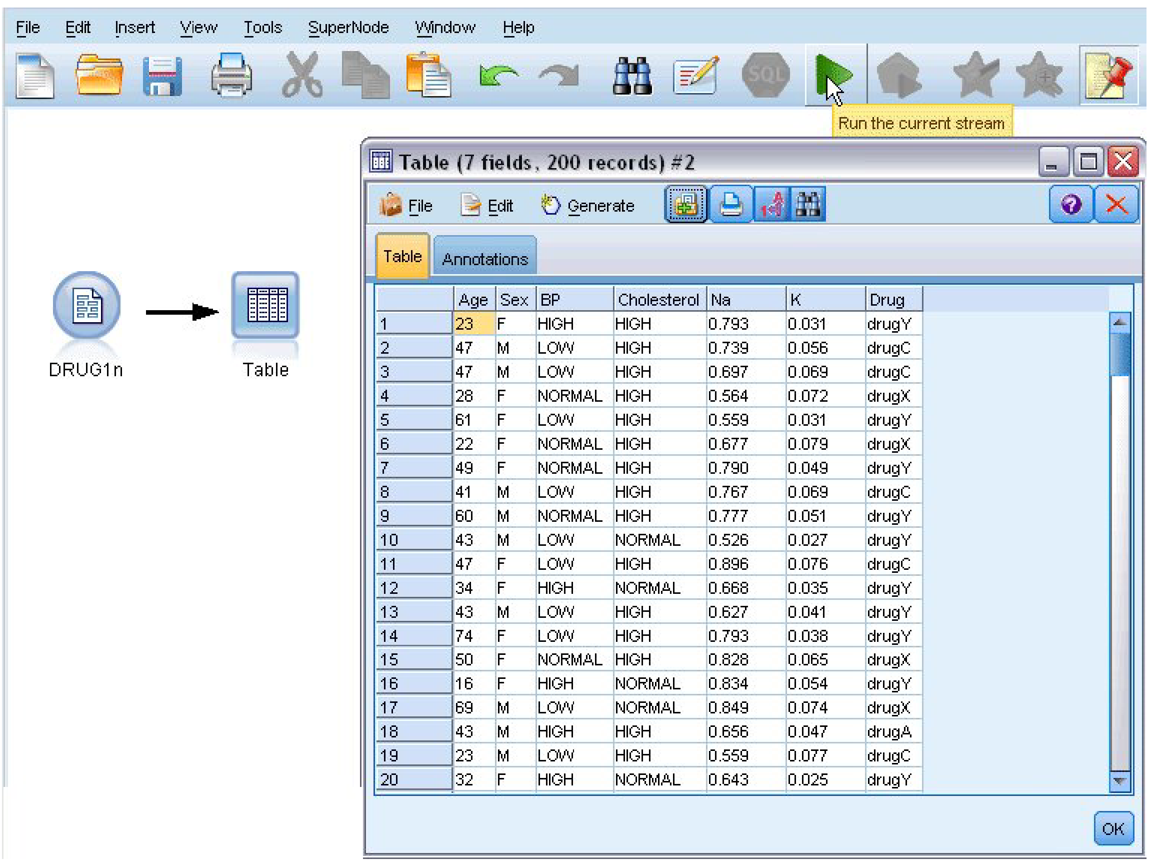


Figure 81. Table node connected to the data source



	Age	Sex	BP	Cholesterol	Na	K	Drug
1	23	F	HIGH	HIGH	0.793	0.031	drugY
2	47	M	LOW	HIGH	0.739	0.056	drugC
3	47	M	LOW	HIGH	0.697	0.069	drugC
4	28	F	NORMAL	HIGH	0.564	0.072	drugX
5	61	F	LOW	HIGH	0.559	0.031	drugY
6	22	F	NORMAL	HIGH	0.677	0.079	drugX
7	49	F	NORMAL	HIGH	0.790	0.049	drugY
8	41	M	LOW	HIGH	0.767	0.069	drugC
9	60	M	NORMAL	HIGH	0.777	0.051	drugY
10	43	M	LOW	NORMAL	0.526	0.027	drugY
11	47	F	LOW	HIGH	0.896	0.076	drugC
12	34	F	HIGH	NORMAL	0.668	0.035	drugY
13	43	M	LOW	HIGH	0.627	0.041	drugY
14	74	F	LOW	HIGH	0.793	0.038	drugY
15	50	F	NORMAL	HIGH	0.828	0.065	drugX
16	16	F	HIGH	NORMAL	0.834	0.054	drugY
17	69	M	LOW	NORMAL	0.849	0.074	drugX
18	43	M	HIGH	HIGH	0.656	0.047	drugA
19	23	M	LOW	HIGH	0.559	0.077	drugC
20	32	F	HIGH	NORMAL	0.643	0.025	drugY

Figure 82. Running a stream from the toolbar

Double-clicking a node from the palette will automatically connect it to the selected node in the stream canvas. Alternatively, if the nodes are not already connected, you can use your middle mouse button to connect the Source node to the Table node. To simulate a middle mouse button, hold down the Alt key while using the mouse. To view the table, click the green arrow button on the toolbar to run the stream, or right-click the Table node and choose **Run**.

Creating a Distribution Graph

During data mining, it is often useful to explore the data by creating visual summaries. IBM SPSS Modeler offers several different types of graphs to choose from, depending on the kind of data that you want to summarize. For example, to find out what proportion of the patients responded to each drug, use a Distribution node.

Add a Distribution node to the stream and connect it to the Source node, then double-click the node to edit options for display.

Select *Drug* as the target field whose distribution you want to show. Then, click **Run** from the dialog box.

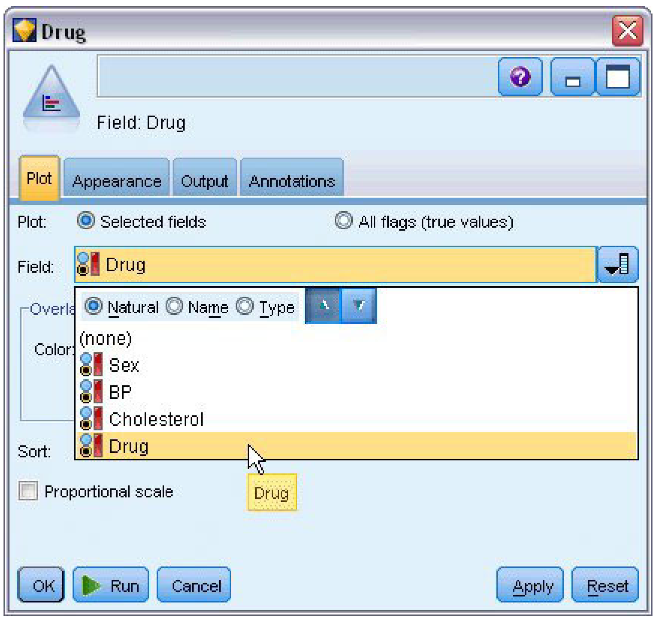


Figure 83. Selecting drug as the target field

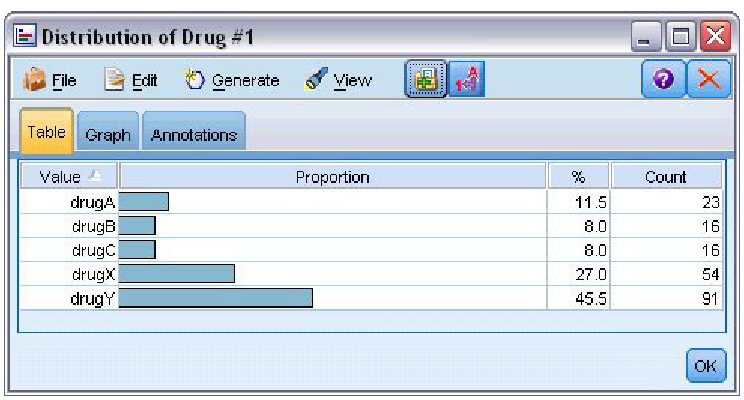


Figure 84. Distribution of response to drug type

The resulting graph helps you see the "shape" of the data. It shows that patients responded to drug *Y* most often and to drugs *B* and *C* least often.

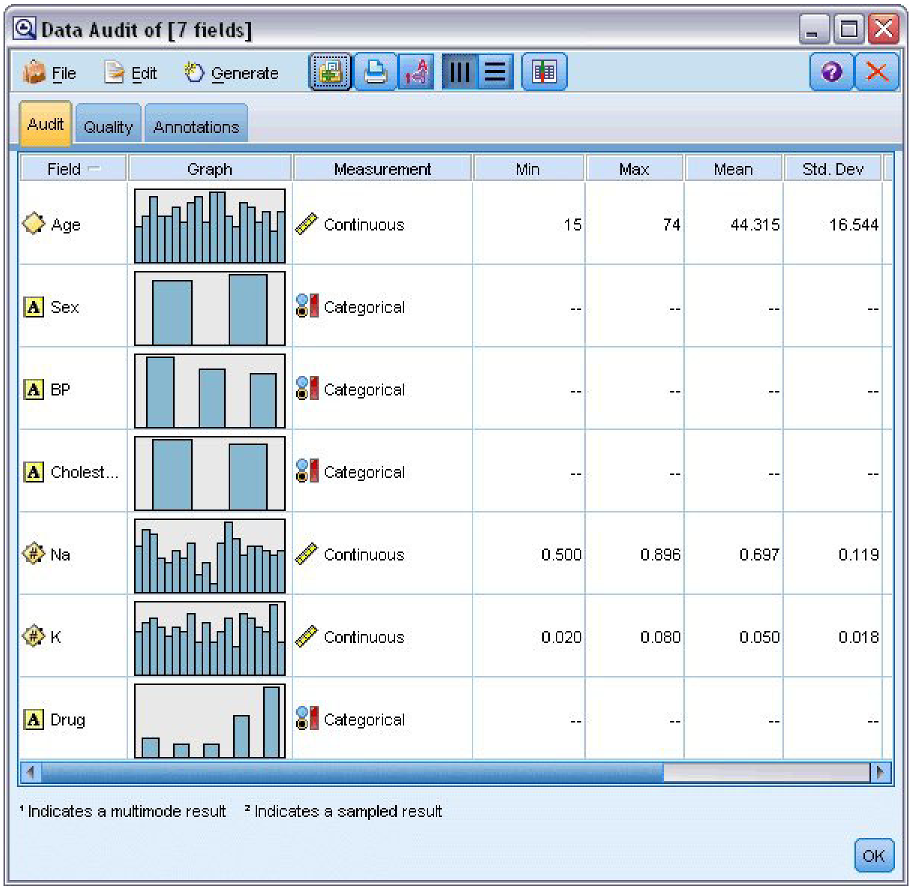


Figure 85. Results of a data audit

Alternatively, you can attach and execute a Data Audit node for a quick glance at distributions and histograms for all fields at once. The Data Audit node is available on the Output tab.

Creating a Scatterplot

Now let's take a look at what factors might influence *Drug*, the target variable. As a researcher, you know that the concentrations of sodium and potassium in the blood are important factors. Since these are both numeric values, you can create a scatterplot of sodium versus potassium, using the drug categories as a color overlay.

Place a Plot node in the workspace and connect it to the Source node, and double-click to edit the node.

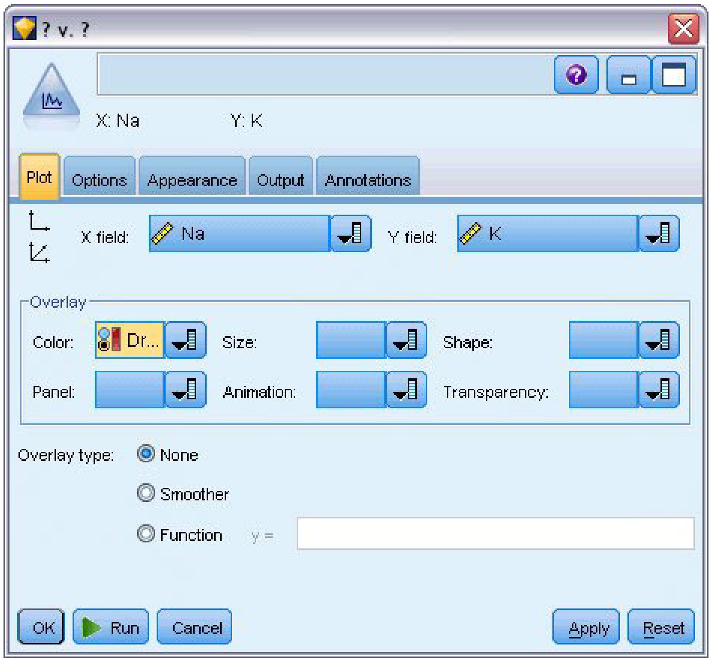


Figure 86. Creating a scatterplot

On the Plot tab, select *Na* as the X field, *K* as the Y field, and *Drug* as the overlay field. Then, click **Run**.

The plot clearly shows a threshold above which the correct drug is always drug *Y* and below which the correct drug is never drug *Y*. This threshold is a ratio--the ratio of sodium (*Na*) to potassium (*K*).

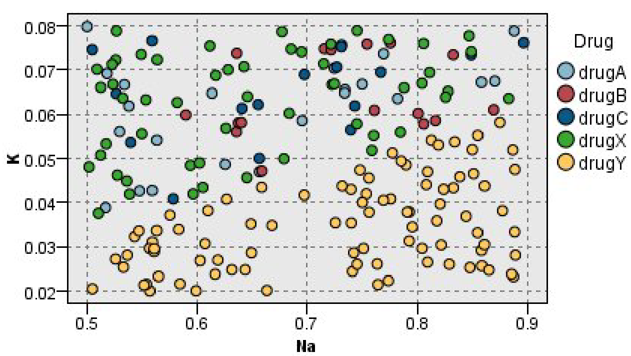


Figure 87. Scatterplot of drug distribution

Creating a Web Graph

Since many of the data fields are categorical, you can also try plotting a web graph, which maps associations between different categories. Start by connecting a Web node to the Source node in your workspace. In the Web node dialog box, select *BP* (for blood pressure) and *Drug*. Then, click **Run**.

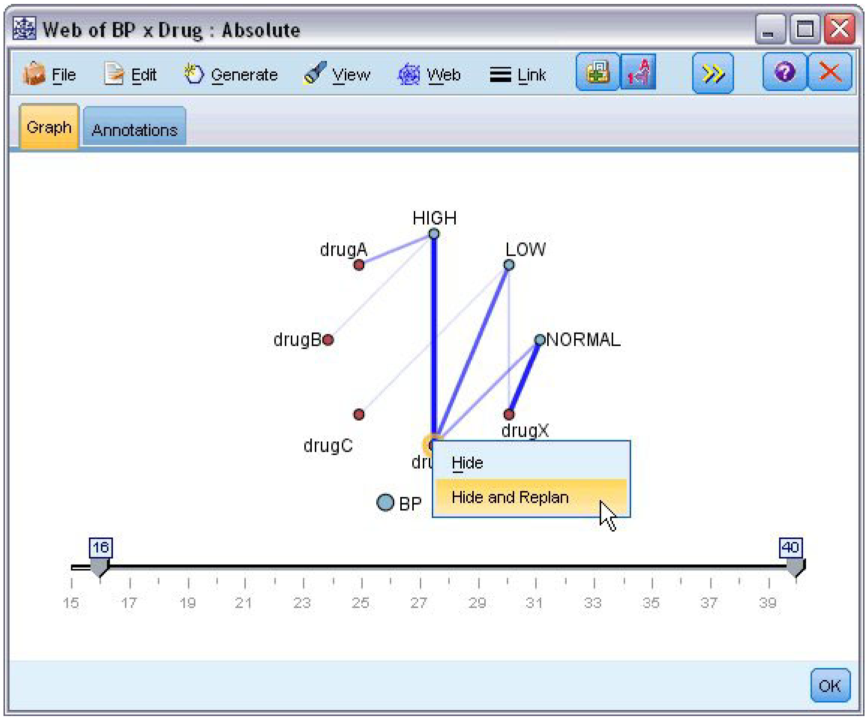


Figure 88. Web graph of drugs vs. blood pressure

From the plot, it appears that drug *Y* is associated with all three levels of blood pressure. This is no surprise--you have already determined the situation in which drug *Y* is best. To focus on the other drugs, you can hide drug *Y*. On the **View** menu, choose **Edit Mode**, then right-click over the drug *Y* point and choose **Hide and Replan**.

In the simplified plot, drug *Y* and all of its links are hidden. Now, you can clearly see that only drugs *A* and *B* are associated with high blood pressure. Only drugs *C* and *X* are associated with low blood pressure. And normal blood pressure is associated only with drug *X*. At this point, though, you still don't know how to choose between drugs *A* and *B* or between drugs *C* and *X*, for a given patient. This is where modeling can help.

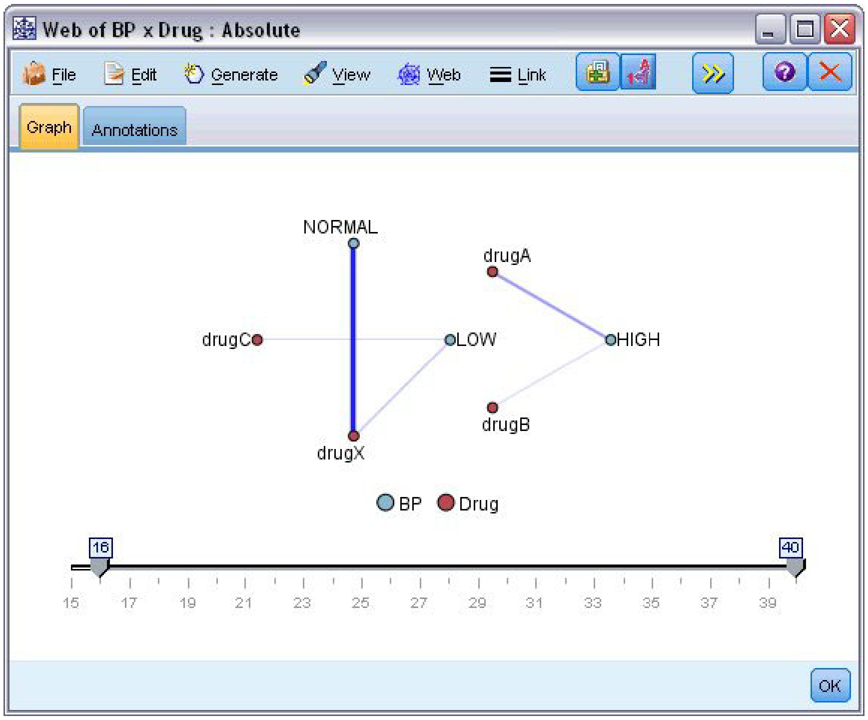


Figure 89. Web graph with drug Y and its links hidden

Deriving a New Field

Since the ratio of sodium to potassium seems to predict when to use drug *Y*, you can derive a field that contains the value of this ratio for each record. This field might be useful later when you build a model to predict when to use each of the five drugs. To simplify the stream layout, start by deleting all the nodes except the DRUG1n source node. Attach a Derive node (Field Ops tab) to DRUG1n, then double-click the Derive node to edit it.

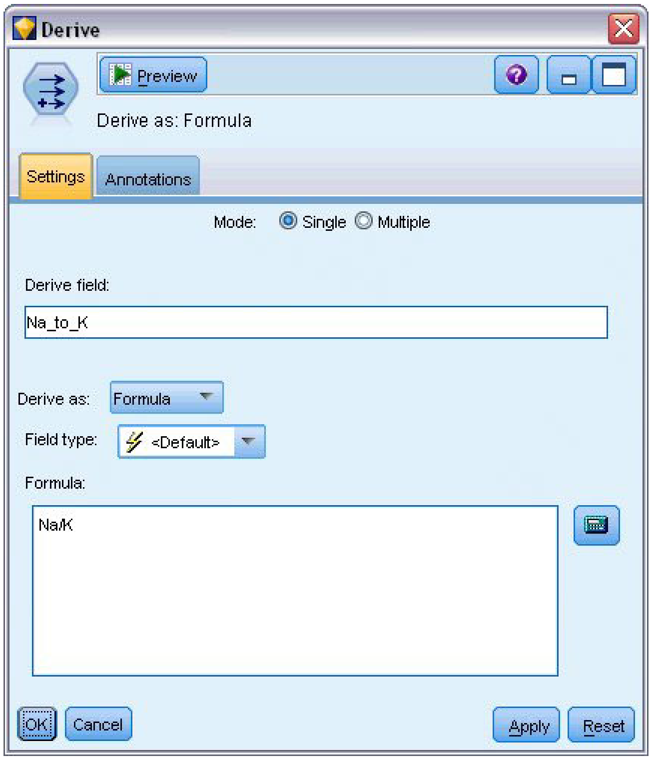


Figure 90. Editing the Derive node

Name the new field *Na_to_K*. Since you obtain the new field by dividing the sodium value by the potassium value, enter `Na/K` for the formula. You can also create a formula by clicking the icon just to the right of the field. This opens the Expression Builder, a way to interactively create expressions using built-in lists of functions, operands, and fields and their values.

You can check the distribution of your new field by attaching a Histogram node to the Derive node. In the Histogram node dialog box, specify *Na_to_K* as the field to be plotted and *Drug* as the overlay field.

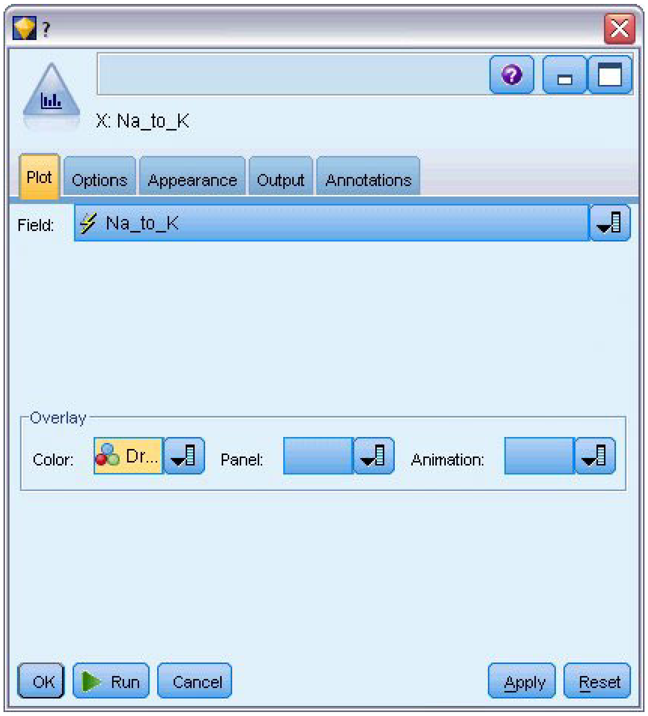


Figure 91. Editing the Histogram node

When you run the stream, you get the graph shown here. Based on the display, you can conclude that when the *Na_to_K* value is about 15 or above, drug *Y* is the drug of choice.

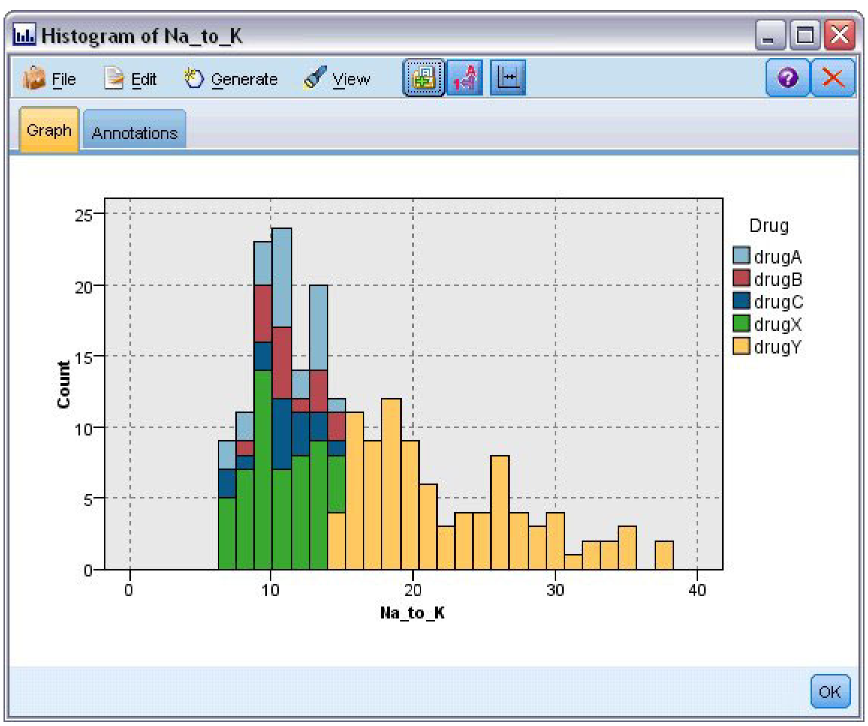


Figure 92. Histogram display

Building a Model

By exploring and manipulating the data, you have been able to form some hypotheses. The ratio of sodium to potassium in the blood seems to affect the choice of drug, as does blood pressure. But you cannot fully explain all of the relationships yet. This is where modeling will likely provide some answers. In this case, you will use try to fit the data using a rule-building model, C5.0.

Since you are using a derived field, *Na_to_K*, you can filter out the original fields, *Na* and *K*, so that they are not used twice in the modeling algorithm. You can do this using a Filter node.

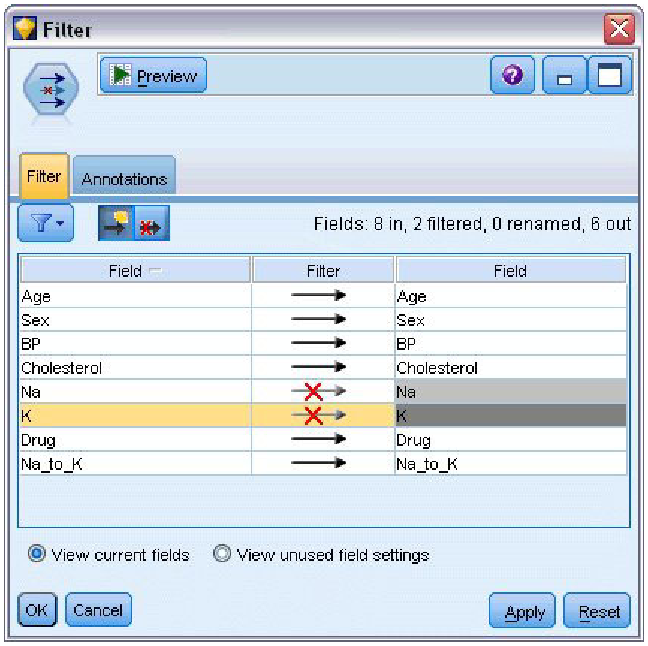


Figure 93. Editing the Filter node

On the Filter tab, click the arrows next to *Na* and *K*. Red Xs appear over the arrows to indicate that the fields are now filtered out.

Next, attach a Type node connected to the Filter node. The Type node allows you to indicate the types of fields that you are using and how they are used to predict the outcomes.

On the Types tab, set the role for the *Drug* field to **Target**, indicating that *Drug* is the field you want to predict. Leave the role for the other fields set to **Input** so they will be used as predictors.

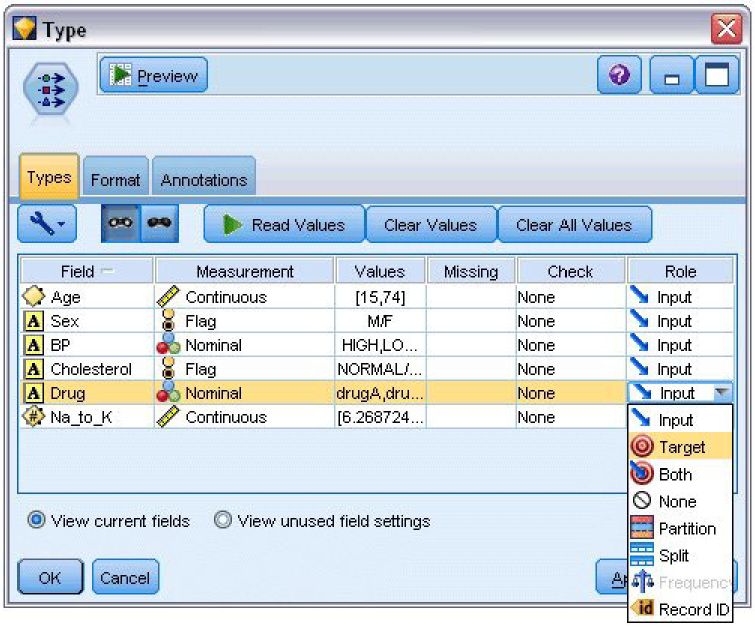


Figure 94. Editing the Type node

To estimate the model, place a C5.0 node in the workspace and attach it to the end of the stream as shown. Then click the green **Run** toolbar button to run the stream.

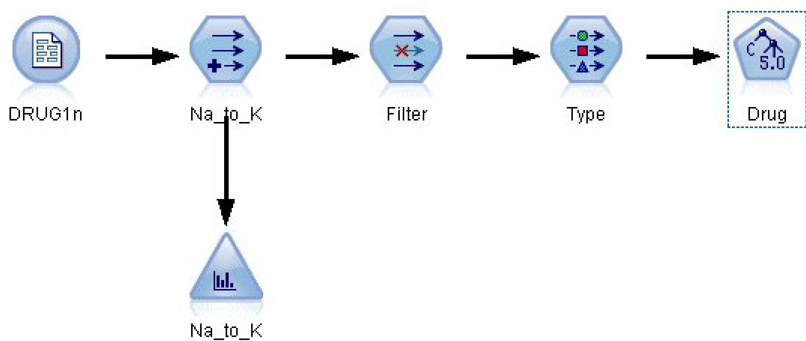


Figure 95. Adding a C5.0 node

Browsing the Model

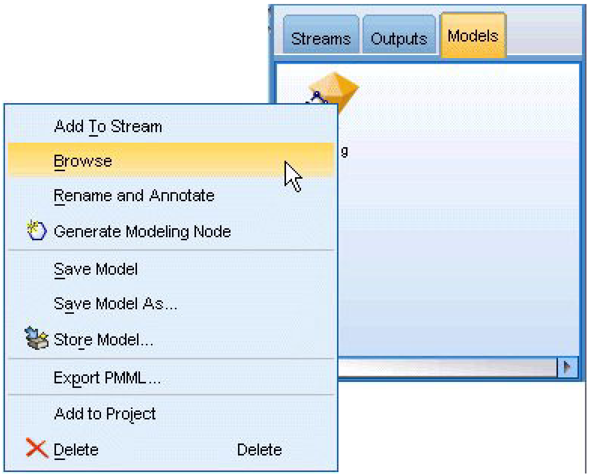


Figure 96. Browsing the model

When the C5.0 node is executed, the model nugget is added to the stream, and also to the Models palette in the upper-right corner of the window. To browse the model, right-click either of the icons and choose **Edit** or **Browse** from the context menu.

The Rule browser displays the set of rules generated by the C5.0 node in a decision tree format. Initially, the tree is collapsed. To expand it, click the **All** button to show all levels.

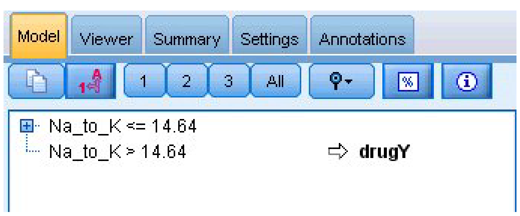


Figure 97. Rule browser

Now you can see the missing pieces of the puzzle. For people with an *Na*-to-*K* ratio less than 14.64 and high blood pressure, age determines the choice of drug. For people with low blood pressure, cholesterol level seems to be the best predictor.

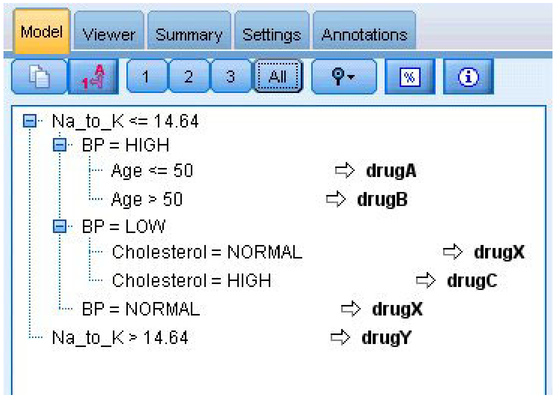


Figure 98. Rule browser fully expanded

The same decision tree can be viewed in a more sophisticated graphical format by clicking the **Viewer** tab. Here, you can see more easily the number of cases for each blood pressure category, as well as the percentage of cases.

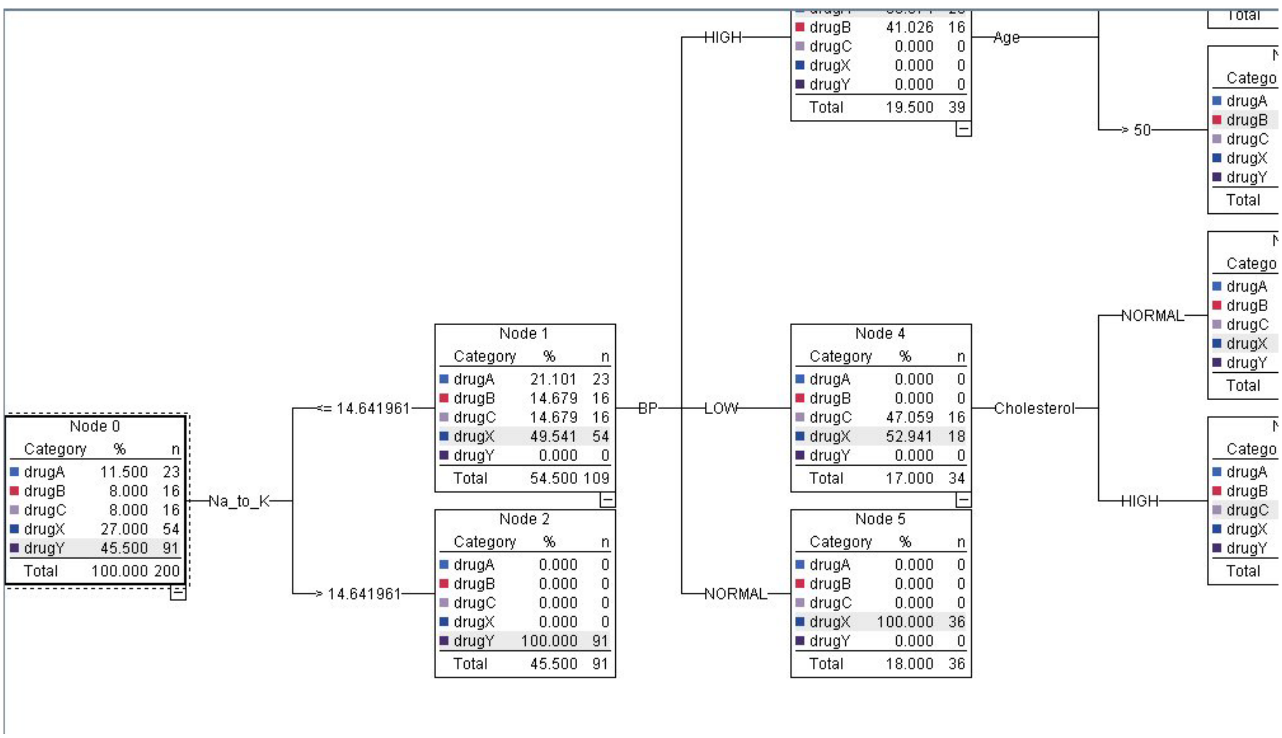


Figure 99. Decision tree in graphical format

Using an Analysis Node

You can assess the accuracy of the model using an analysis node. Attach an Analysis node (from the Output node palette) to the model nugget, open the Analysis node and click **Run**.

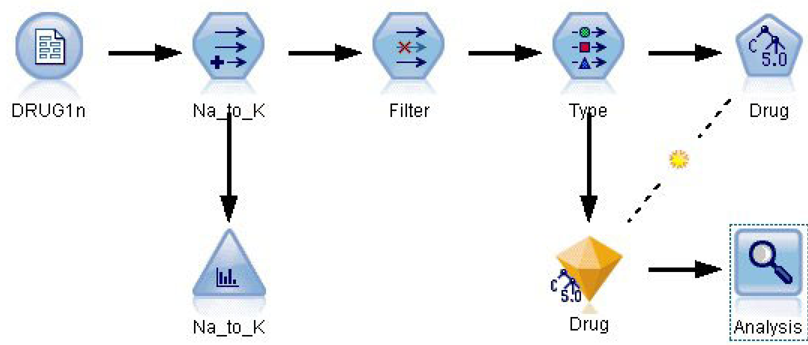


Figure 100. Adding an Analysis node

The Analysis node output shows that with this artificial dataset, the model correctly predicted the choice of drug for every record in the dataset. With a real dataset you are unlikely to see 100% accuracy, but you can use the Analysis node to help determine whether the model is acceptably accurate for your particular application.

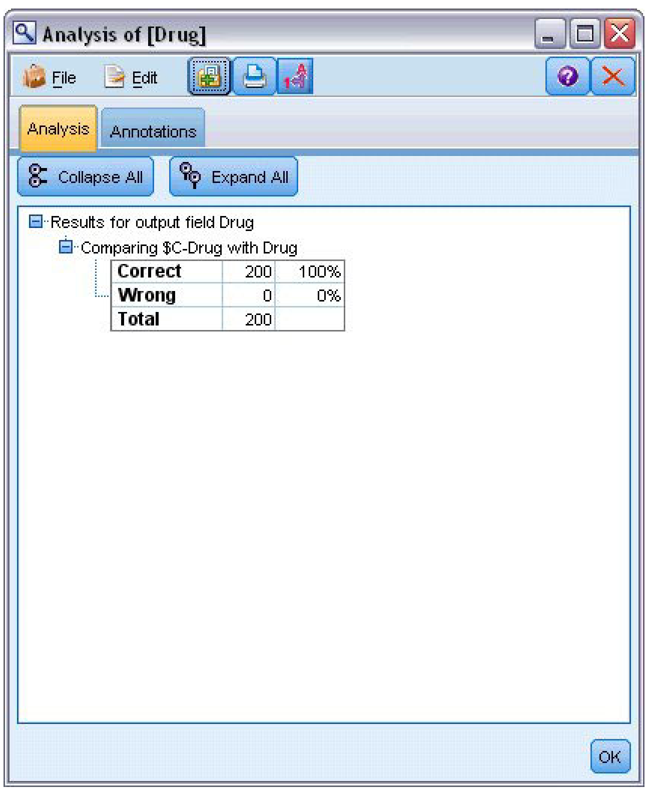


Figure 101. Analysis node output

Chapter 9. Screening Predictors (Feature Selection)

The Feature Selection node helps you to identify the fields that are most important in predicting a certain outcome. From a set of hundreds or even thousands of predictors, the Feature Selection node screens, ranks, and selects the predictors that may be most important. Ultimately, you may end up with a quicker, more efficient model—one that uses fewer predictors, executes more quickly, and may be easier to understand.

The data used in this example represent a data warehouse for a hypothetical telephone company and contain information about responses to a special promotion by 5,000 of the company's customers. The data include a large number of fields containing customers' age, employment, income, and telephone usage statistics. Three "target" fields show whether or not the customer responded to each of three offers. The company wants to use this data to help predict which customers are most likely to respond to similar offers in the future.

This example uses the stream named *featureselection.str*, which references the data file named *customer_dbase.sav*. These files are available from the *Demos* directory of any IBM SPSS Modeler installation. This can be accessed from the IBM SPSS Modeler program group on the Windows Start menu. The *featureselection.str* file is in the *streams* directory.

This example focuses on only one of the offers as a target. It uses the CHAID tree-building node to develop a model to describe which customers are most likely to respond to the promotion. It contrasts two approaches:

- Without feature selection. All predictor fields in the dataset are used as inputs to the CHAID tree.
- With feature selection. The Feature Selection node is used to select the top 10 predictors. These are then input into the CHAID tree.

By comparing the two resulting tree models, we can see how feature selection produces effective results.

Building the Stream

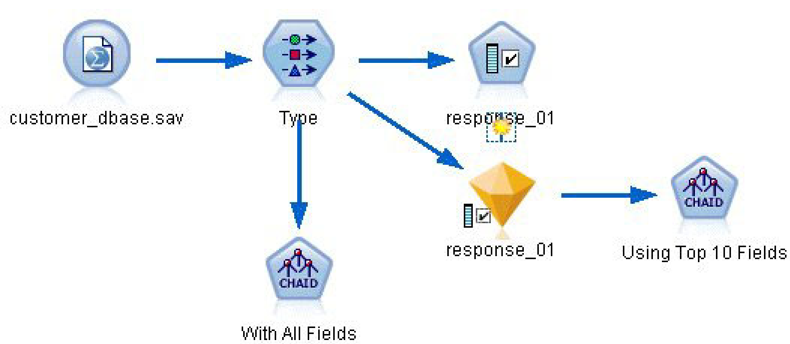


Figure 102. Feature Selection example stream

1. Place a Statistics File source node onto a blank stream canvas. Point this node to the example data file *customer_dbase.sav*, available in the *Demos* directory under your IBM SPSS Modeler installation. (Alternatively, open the example stream file *featureselection.str* in the *streams* directory.)
2. Add a Type node. On the Types tab, scroll down to the bottom and change the role for *response_01* to *Target*. Change the role to *None* for the other response fields (*response_02* and *response_03*) as well as for the customer ID (*custid*) at the top of the list. Leave the role set to *Input* for all other fields, and

click the **Read Values** button, then click **OK**.

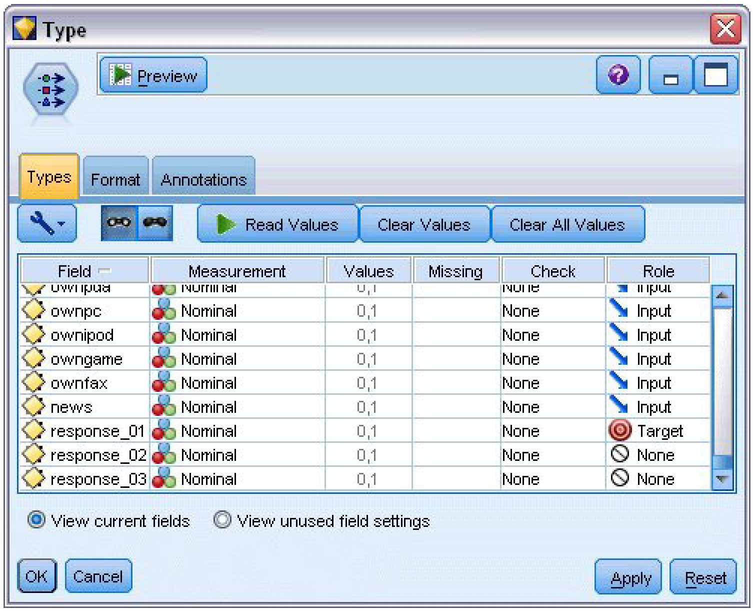


Figure 103. Adding a Type node

3. Add a Feature Selection modeling node to the stream. On this node, you can specify the rules and criteria for screening, or disqualifying, fields.
4. Run the stream to create the Feature Selection model nugget.
5. Right-click the model nugget on the stream or in the Models palette and choose **Edit** or **Browse** to look at the results.

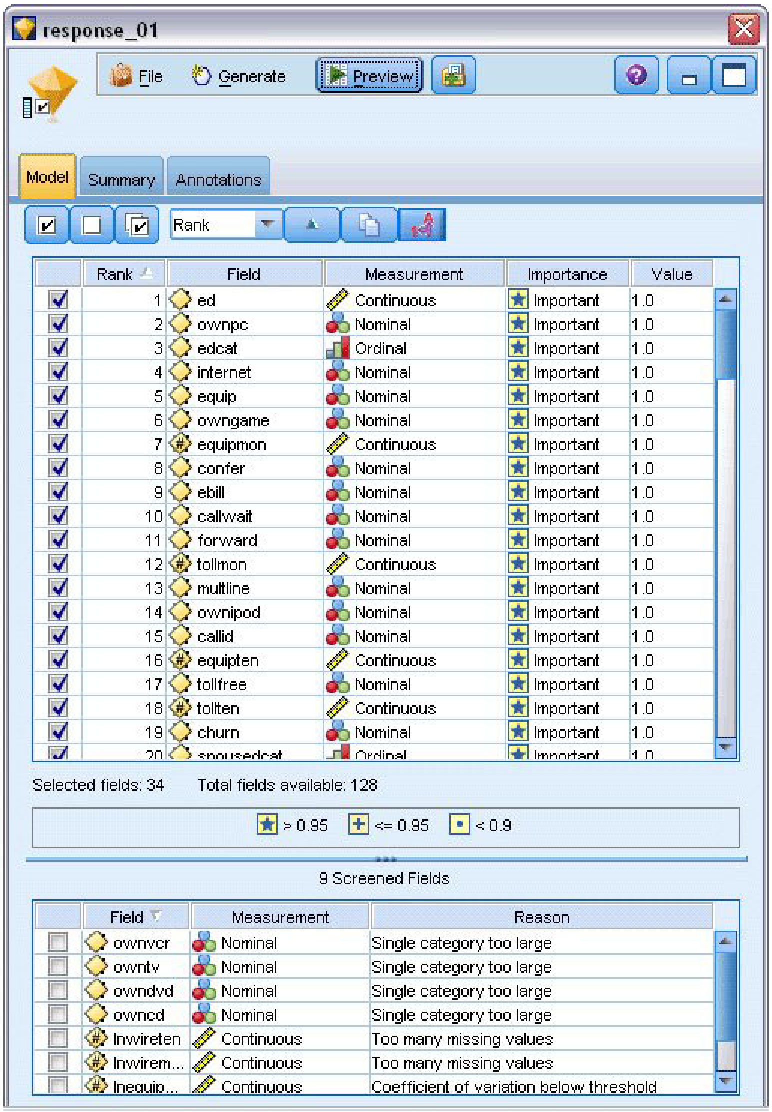


Figure 104. Model tab in Feature Selection model nugget

The top panel shows the fields found to be useful in the prediction. These are ranked based on importance. The bottom panel shows which fields were screened from the analysis and why. By examining the fields in the top panel, you can decide which ones to use in subsequent modeling sessions.

6. Now we can select the fields to use downstream. Although 34 fields were originally identified as important, we want to reduce the set of predictors even further.
7. Select only the top 10 predictors using the check marks in the first column to deselect the unwanted predictors. (Click the check mark in row 11, hold down the Shift key and click the check mark in row 34.) Close the model nugget.
8. To compare results without feature selection, you must add two CHAID modeling nodes to the stream: one that uses feature selection and one that does not.
9. Connect one CHAID node to the Type node, and the other one to the Feature Selection model nugget.
10. Open each CHAID node, select the Build Options tab and ensure that the options **Build new model**, **Build a single tree** and **Launch interactive session** are selected in the Objectives pane.

On the Basics pane, make sure that **Maximum Tree Depth** is set to 5.

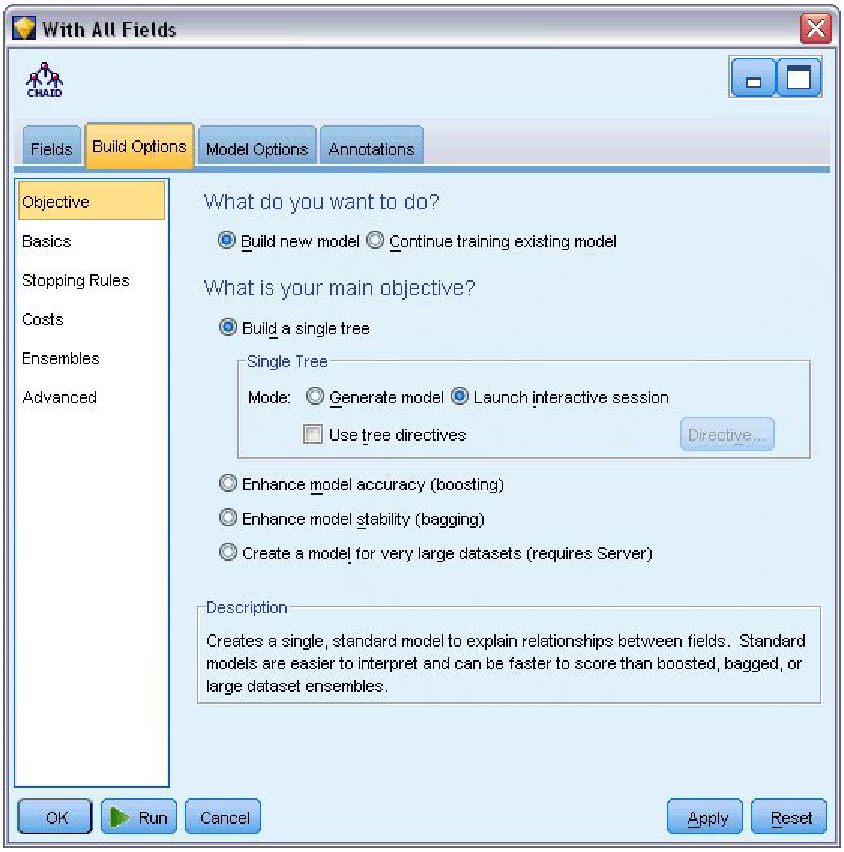


Figure 105. Objectives settings for CHAID modeling node for all predictor fields

Building the Models

1. Execute the CHAID node that uses all of the predictors in the dataset (the one connected to the Type node). As it runs, notice how long it takes to execute. The results window displays a table.
2. From the menus, choose **Tree > Grow Tree** to grow and display the expanded tree.

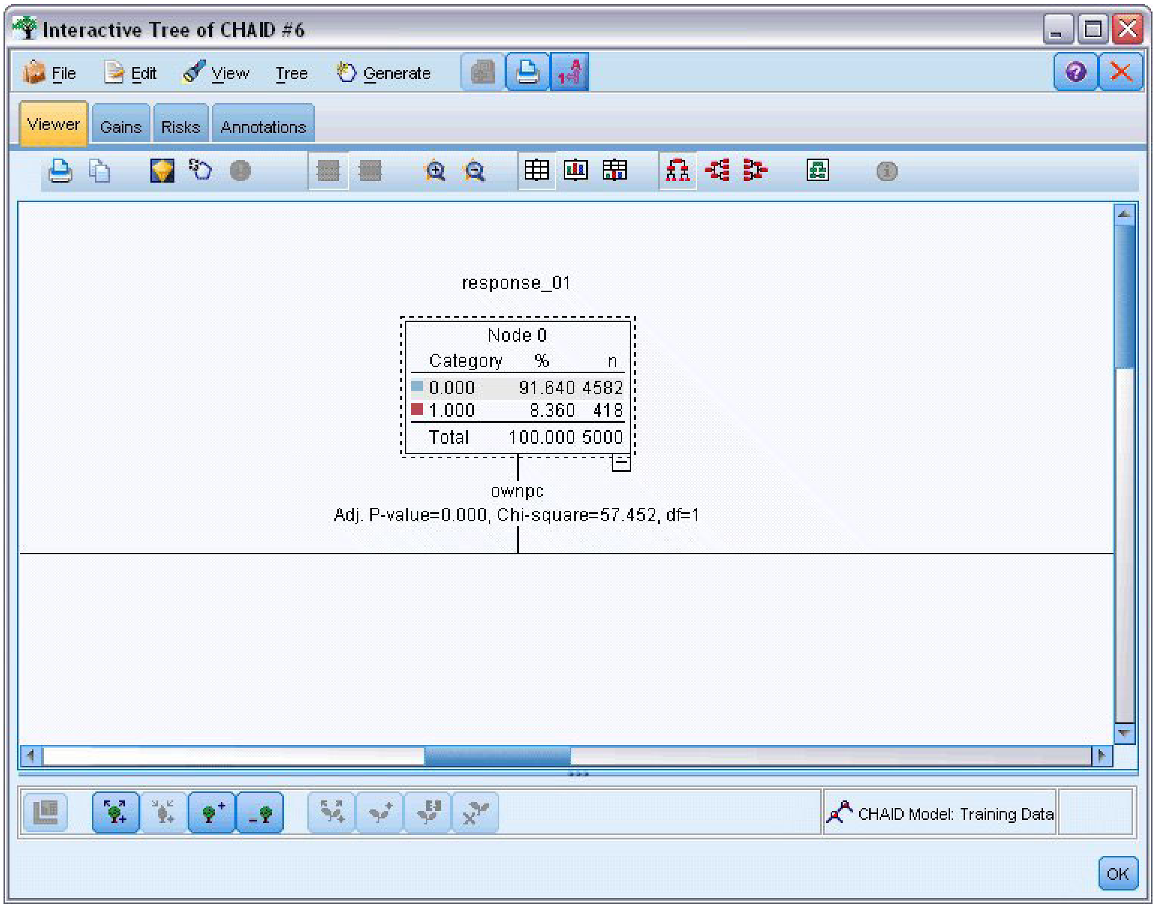


Figure 106. Growing the tree in the Tree Builder

3. Now do the same for the other CHAID node, which uses only 10 predictors. Again, grow the tree when the Tree Builder opens.

The second model should have executed faster than the first one. Because this dataset is fairly small, the difference in execution times is probably a few seconds; but for larger real-world datasets, the difference may be very noticeable—minutes or even hours. Using feature selection may speed up your processing times dramatically.

The second tree also contains fewer tree nodes than the first. It is easier to comprehend. But before you decide to use it, you need to find out whether it is effective and how it compares to the model that uses all predictors.

Comparing the Results

To compare the two results, we need a measure of effectiveness. For this, we will use the Gains tab in the Tree Builder. We will look at **lift**, which measures how much more likely the records in a node are to fall under the target category when compared to all records in the dataset. For example, a lift value of 148% indicates that records in the node are 1.48 times more likely to fall under the target category than all records in the dataset. Lift is indicated in the *Index* column on the Gains tab.

1. In the Tree Builder for the full set of predictors, click the Gains tab. Change the target category to 1.0. Change the display to quartiles by first clicking the Quantiles toolbar button. Then select **Quartile** from the drop-down list to the right of this button.
2. Repeat this procedure in the Tree Builder for the set of 10 predictors so that you have two similar Gains tables to compare, as shown in the following figures.

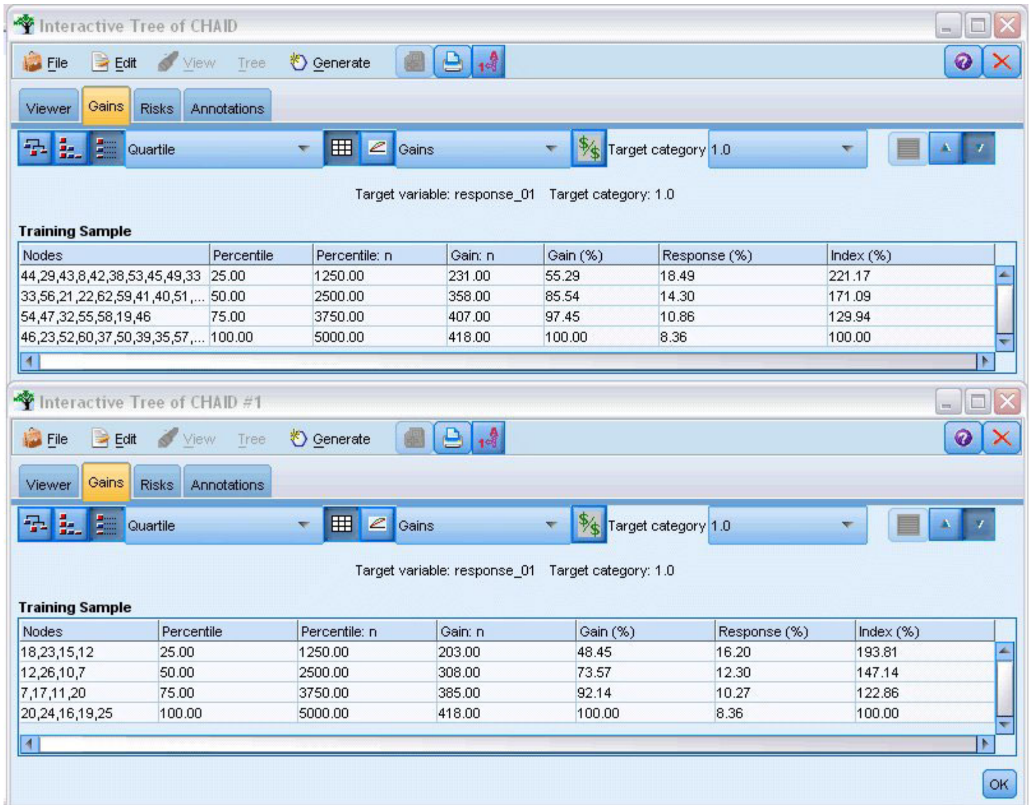


Figure 107. Gains charts for the two CHAID models

Each Gains table groups the terminal nodes for its tree into quartiles. To compare the effectiveness of the two models, look at the lift (*Index* value) for the top quartile in each table.

When all predictors are included, the model shows a lift of 221%. That is, cases with the characteristics in these nodes are 2.2 times more likely to respond to the target promotion. To see what those characteristics are, click to select the top row. Then switch to the Viewer tab, where the corresponding nodes are now outlined in black. Follow the tree down to each highlighted terminal node to see how the predictors were split. The top quartile alone includes 10 nodes. When translated into real-world scoring models, 10 different customer profiles can be difficult to manage.

With only the top 10 predictors (as identified by feature selection) included, the lift is nearly 194%. Although this model is not quite as good as the model that uses all predictors, it is certainly useful. Here, the top quartile includes only four nodes, so it is simpler. Therefore, we can determine that the feature selection model is preferable to the one with all predictors.

Summary

Let's review the advantages of feature selection. Using fewer predictors is less expensive. It means that you have less data to collect, process, and feed into your models. Computing time is improved. In this example, even with the extra feature selection step, model building was noticeably faster with the smaller set of predictors. With a larger real-world dataset, the time savings should be greatly amplified.

Using fewer predictors results in simpler scoring. As the example shows, you might identify only four profiles of customers who are likely to respond to the promotion. Note that with larger numbers of predictors, you run the risk of overfitting your model. The simpler model may generalize better to other datasets (although you would need to test this to be sure).

You could have used a tree-building algorithm to do the feature selection work, allowing the tree to identify the most important predictors for you. In fact, the CHAID algorithm is often used for this purpose, and it is even possible to grow the tree level-by-level to control its depth and complexity. However, the Feature Selection node is faster and easier to use. It ranks all of the predictors in one fast step, allowing you to identify the most important fields quickly. It also allows you to vary the number of predictors to include. You could easily run this example again using the top 15 or 20 predictors instead of 10, comparing the results to determine the optimal model.

Chapter 10. Reducing Input Data String Length (Reclassify Node)

Reducing Input Data String Length (Reclassify)

For binomial logistic regression, and auto classifier models that include a binomial logistic regression model, string fields are limited to a maximum of eight characters. Where strings are more than eight characters, they can be recoded using a Reclassify node.

This example uses the stream named *reclassify_strings.str*, which references the data file named *drug_long_name*. These files are available from the *Demos* directory of any IBM SPSS Modeler installation. This can be accessed from the IBM SPSS Modeler program group on the Windows Start menu. The *reclassify_strings.str* file is in the *streams* directory.

This example focuses on a small part of a stream to show the sort of errors that may be generated with overlong strings and explains how to use the Reclassify node to change the string details to an acceptable length. Although the example uses a binomial Logistic Regression node, it is equally applicable when using the Auto Classifier node to generate a binomial Logistic Regression model.

Reclassifying the Data

1. Using a Variable File source node, connect to the dataset *drug_long_name* in the *Demos* folder.

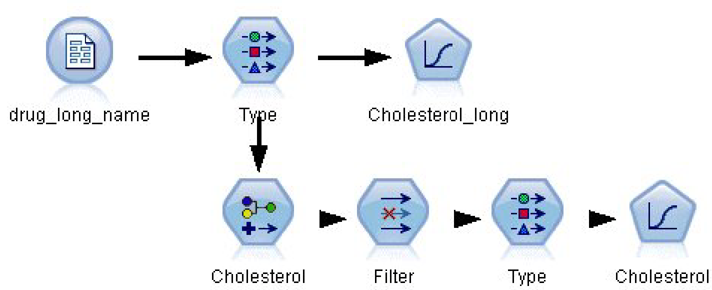


Figure 108. Sample stream showing string reclassification for binomial logistic regression

2. Add a Type node to the Source node and select **Cholesterol_long** as the target.
3. Add a Logistic Regression node to the Type node.
4. In the Logistic Regression node, click the Model tab and select the **Binomial** procedure.

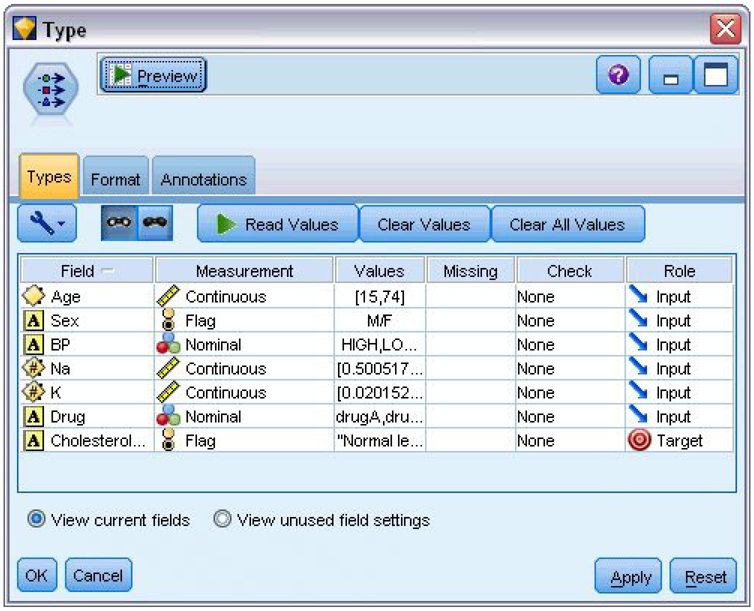


Figure 109. Long string details in the "Cholesterol_long" field

5. When you execute the Logistic Regression node in *reclassify_strings.str*, an error message is displayed warning that the **Cholesterol_long** string values are too long.

   If you encounter this type of error message, follow the procedure explained in the rest of this example to modify your data.

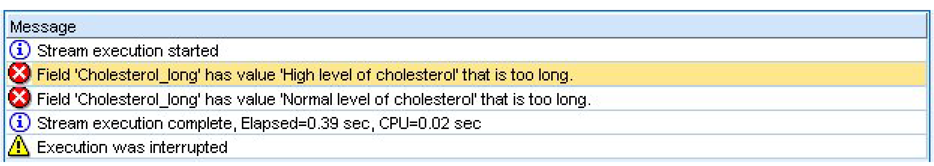


Figure 110. Error message displayed when executing the binomial logistic regression node

6. Add a Reclassify node to the Type node.
7. In the Reclassify field, select **Cholesterol_long**.
8. Type **Cholesterol** as the new field name.
9. Click the **Get** button to add the **Cholesterol_long** values to the original value column.
10. In the new value column, type **High** next to the original value of **High level of cholesterol** and **Normal** next to the original value of **Normal level of cholesterol**.

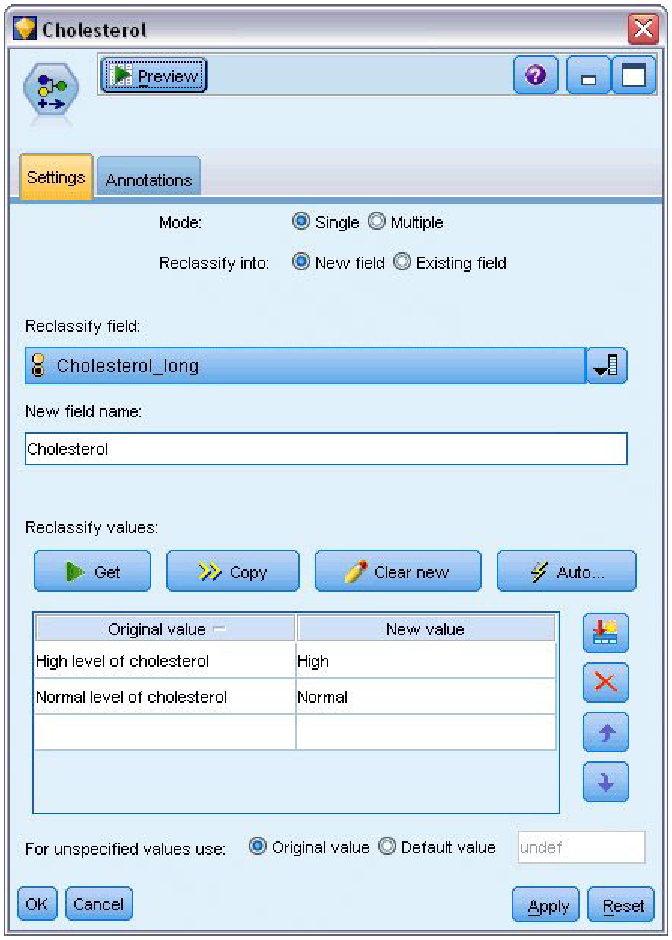


Figure 111. Reclassifying the long strings

11. Add a Filter node to the Reclassify node.
12. In the Filter column, click to remove **Cholesterol_long**.

Figure 112. Filtering the "Cholesterol_long" field from the data

13. Add a Type node to the Filter node and select **Cholesterol** as the target.

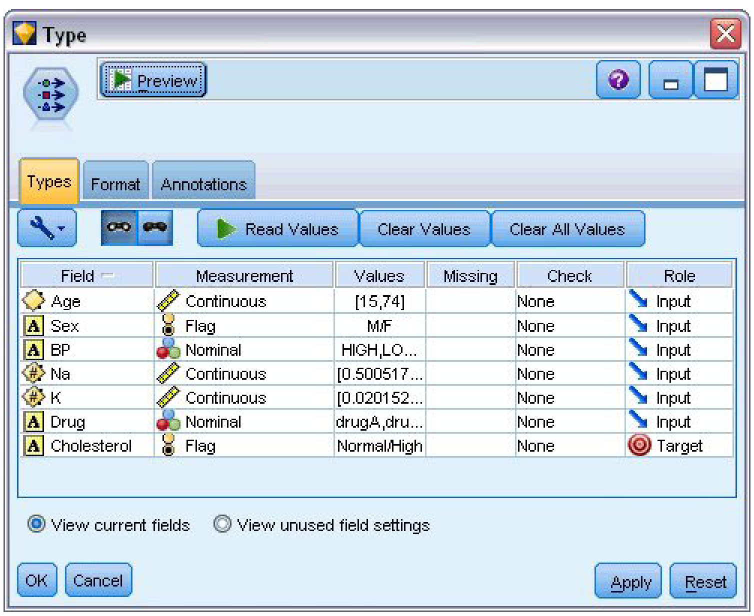


Figure 113. Short string details in the "Cholesterol" field

14. Add a Logistic Node to the Type node.
15. In the Logistic node, click the Model tab and select the **Binomial** procedure.
16. You can now execute the Binomial Logistic node and generate a model without displaying an error message.

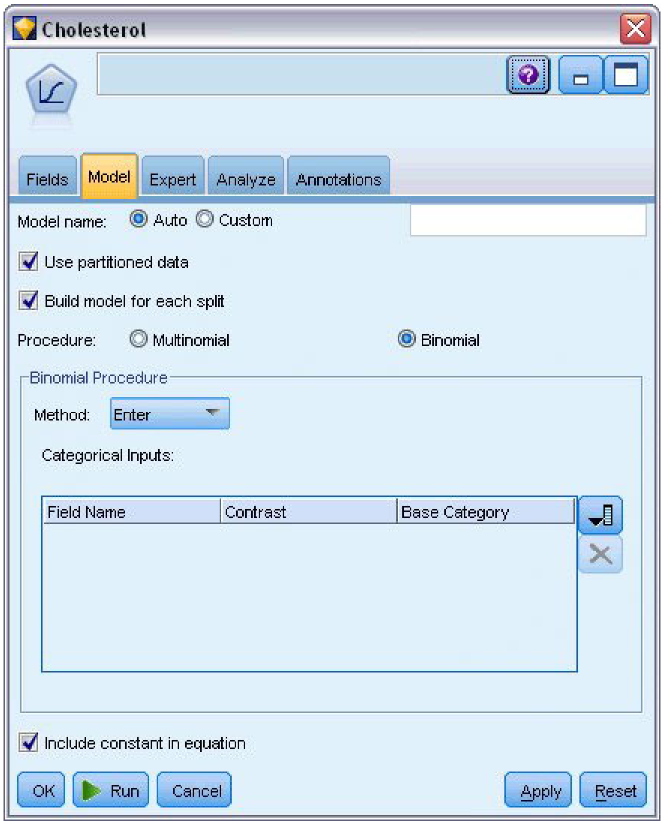


Figure 114. Choosing Binomial as the procedure

This example only shows part of a stream. If you require further information about the types of streams in which you may need to reclassify long strings, the following examples are available:

- Auto Classifier node. See the topic "Modeling Customer Response (Auto Classifier)" on page 35 for more information.
- Binomial Logistic Regression node. See the topic Chapter 13, "Telecommunications Churn (Binomial Logistic Regression)," on page 137 for more information.

More information on how to use IBM SPSS Modeler, such as a user's guide, node reference, and algorithms guide, are available from the *\Documentation* directory of the installation disk.

Chapter 11. Modeling Customer Response (Decision List)

The Decision List algorithm generates rules that indicate a higher or lower likelihood of a given binary (yes or no) outcome. Decision List models are widely used in customer relationship management, such as call center or marketing applications.

This example is based on a fictional company that wants to achieve more profitable results in future marketing campaigns by matching the right offer to each customer. Specifically, the example uses a Decision List model to identify the characteristics of customers who are most likely to respond favorably, based on previous promotions, and to generate a mailing list based on the results.

Decision List models are particularly well suited to interactive modeling, allowing you to adjust parameters in the model and immediately see the results. For a different approach that allows you to automatically create a number of different models and rank the results, the Auto Classifier node can be used instead.

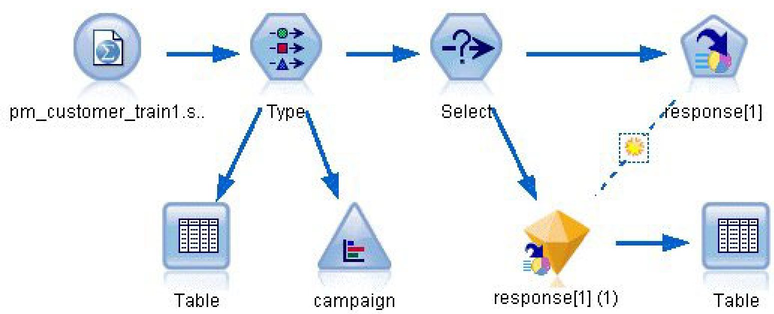
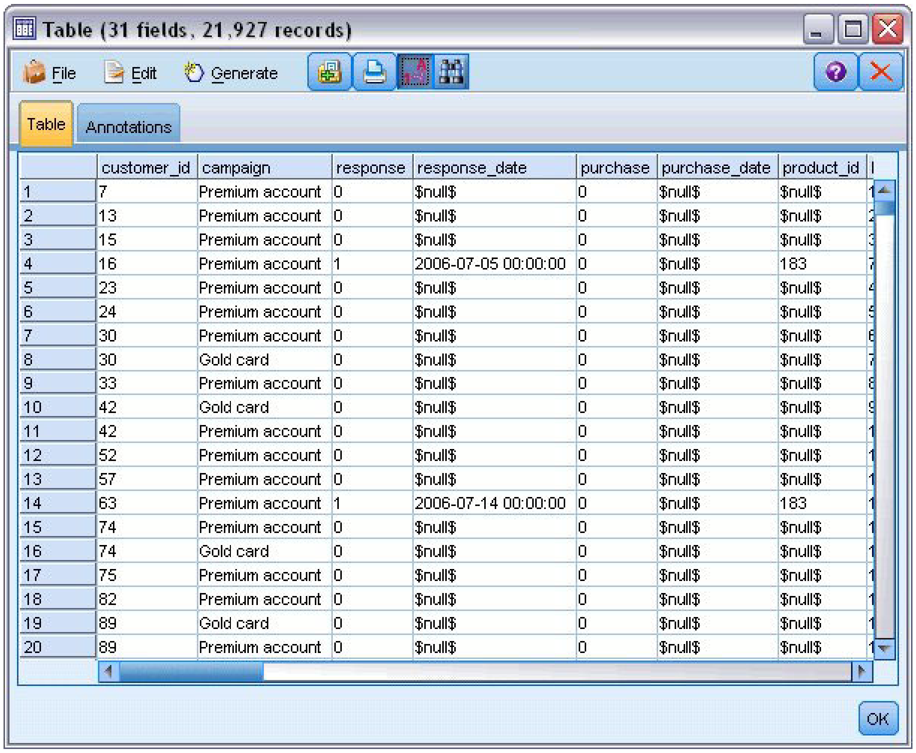


Figure 115. Decision List sample stream

This example uses the stream *pm_decisionlist.str*, which references the data file *pm_customer_train1.sav*. These files are available from the *Demos* directory of any IBM SPSS Modeler installation. This can be accessed from the IBM SPSS Modeler program group on the Windows Start menu. The *pm_decisionlist.str* file is in the *streams* directory.

Historical Data

The file *pm_customer_train1.sav* has historical data tracking the offers made to specific customers in past campaigns, as indicated by the value of the *campaign* field. The largest number of records fall under the *Premium account* campaign.



	customer_id	campaign	response	response_date	purchase	purchase_date	product_id	I
1	7	Premium account	0	$null$	0	$null$	$null$	1
2	13	Premium account	0	$null$	0	$null$	$null$	2
3	15	Premium account	0	$null$	0	$null$	$null$	3
4	16	Premium account	1	2006-07-05 00:00:00	0	$null$	183	7
5	23	Premium account	0	$null$	0	$null$	$null$	4
6	24	Premium account	0	$null$	0	$null$	$null$	5
7	30	Premium account	0	$null$	0	$null$	$null$	6
8	30	Gold card	0	$null$	0	$null$	$null$	7
9	33	Premium account	0	$null$	0	$null$	$null$	8
10	42	Gold card	0	$null$	0	$null$	$null$	9
11	42	Premium account	0	$null$	0	$null$	$null$	1
12	52	Premium account	0	$null$	0	$null$	$null$	1
13	57	Premium account	0	$null$	0	$null$	$null$	1
14	63	Premium account	1	2006-07-14 00:00:00	0	$null$	183	1
15	74	Premium account	0	$null$	0	$null$	$null$	1
16	74	Gold card	0	$null$	0	$null$	$null$	1
17	75	Premium account	0	$null$	0	$null$	$null$	1
18	82	Premium account	0	$null$	0	$null$	$null$	1
19	89	Gold card	0	$null$	0	$null$	$null$	1
20	89	Premium account	0	$null$	0	$null$	$null$	1

Figure 116. Data about previous promotions

The values of the *campaign* field are actually coded as integers in the data, with labels defined in the Type node (for example, *2* = *Premium account*). You can toggle display of value labels in the table using the toolbar.

The file also includes a number of fields containing demographic and financial information about each customer that can be used to build or "train" a model that predicts response rates for different groups based on specific characteristics.

Building the Stream

1. Add a Statistics File node pointing to *pm_customer_train1.sav*, located in the *Demos* folder of your IBM SPSS Modeler installation. (You can specify `$CLEO_DEMOS/` in the file path as a shortcut to reference this folder.)

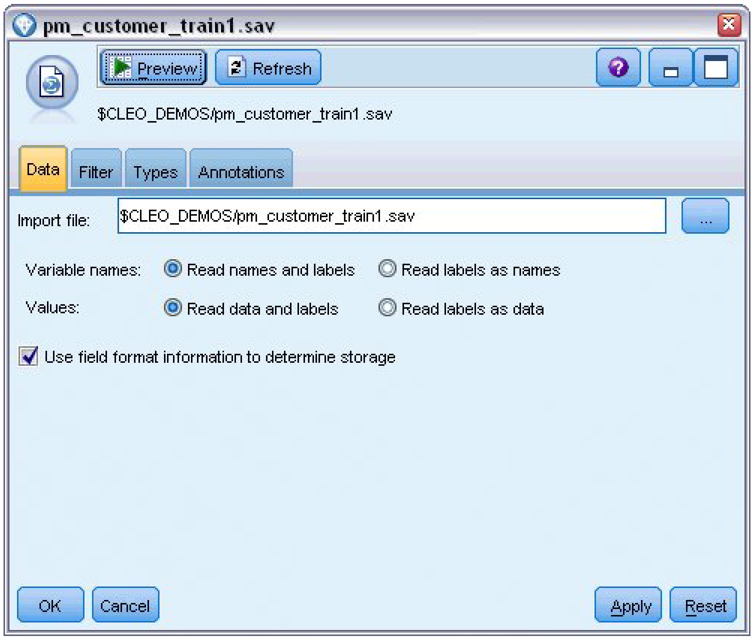


Figure 117. Reading in the data

2. Add a Type node, and select *response* as the target field (Role = **Target**). Set the measurement level for this field to **Flag**.

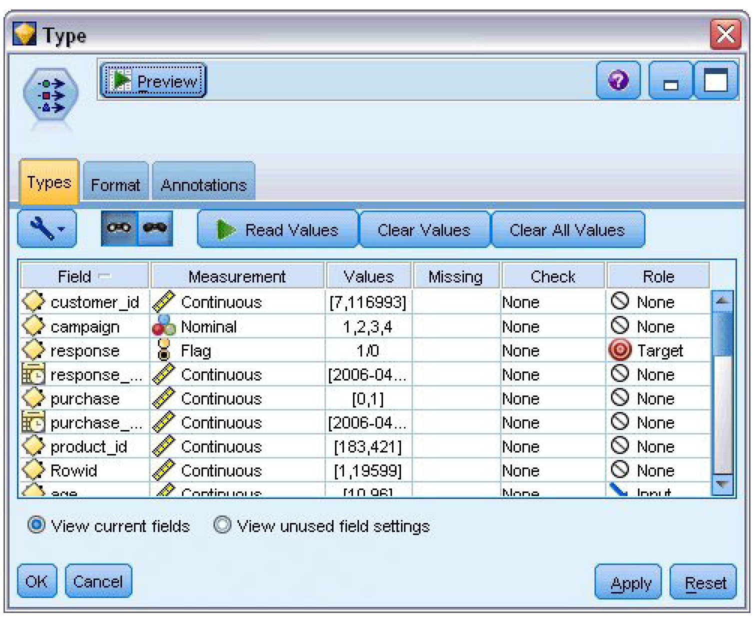


Figure 118. Setting the measurement level and role

3. Set the role to **None** for the following fields: *customer_id*, *campaign*, *response_date*, *purchase*, *purchase_date*, *product_id*, *Rowid*, and *X_random*. These fields all have uses in the data but will not be used in building the actual model.
4. Click the **Read Values** button in the Type node to make sure that values are instantiated.

Although the data includes information about four different campaigns, you will focus the analysis on one campaign at a time. Since the largest number of records fall under the Premium campaign (coded *campaign* = 2 in the data), you can use a Select node to include only these records in the stream.

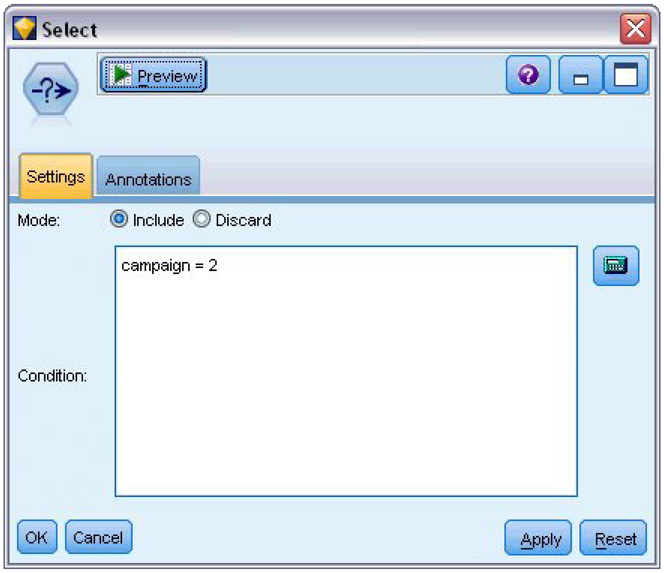


Figure 119. Selecting records for a single campaign

Creating the Model

1. Attach a Decision List node to the stream. On the Model tab, set the **Target value** to 1 to indicate the outcome you want to search for. In this case, you are looking for customers who responded *Yes* to a previous offer.

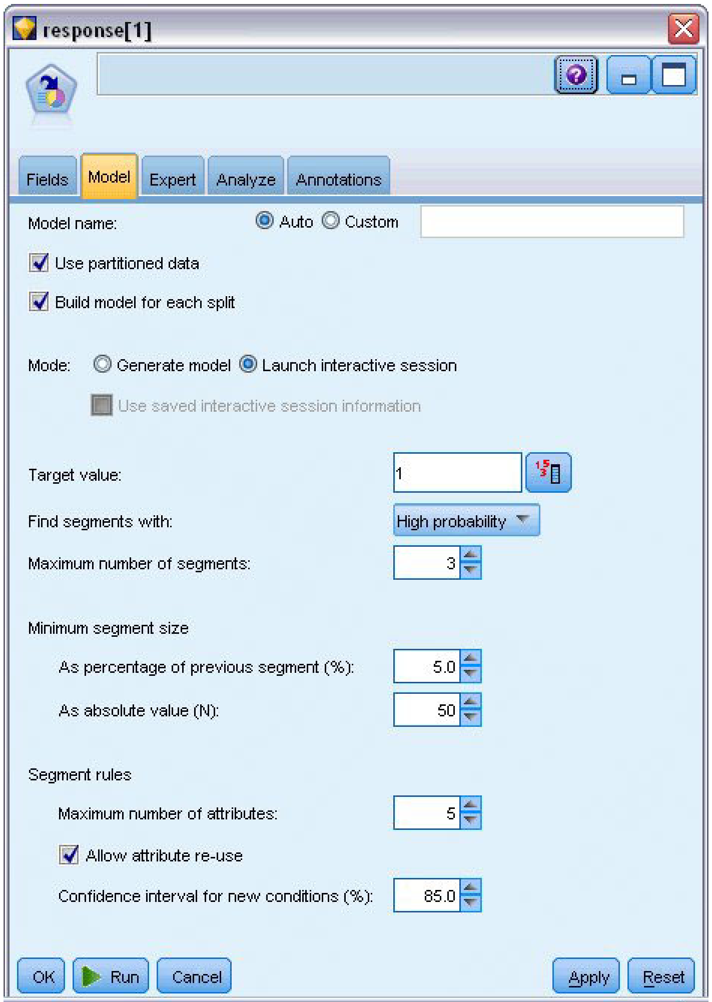


Figure 120. Decision List node, Model tab

2. Select **Launch interactive session**.
3. To keep the model simple for purposes of this example, set the maximum number of segments to 3.
4. Change the confidence interval for new conditions to 85%.
5. On the Expert tab, set the **Mode** to **Expert**.

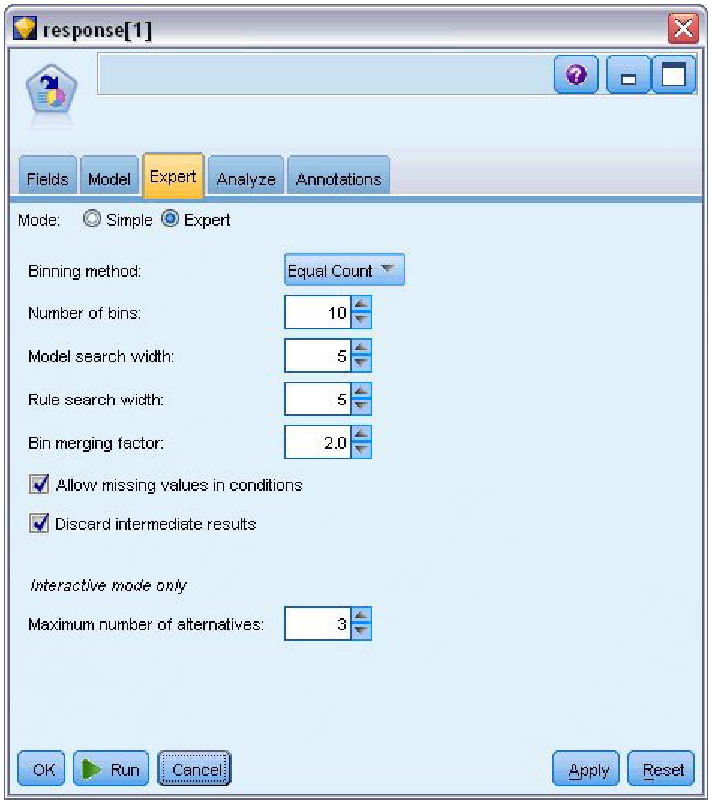


Figure 121. Decision List node, Expert tab

6. Increase the **Maximum number of alternatives** to 3. This option works in conjunction with the **Launch interactive session** setting that you selected on the Model tab.
7. Click **Run** to display the Interactive List viewer.

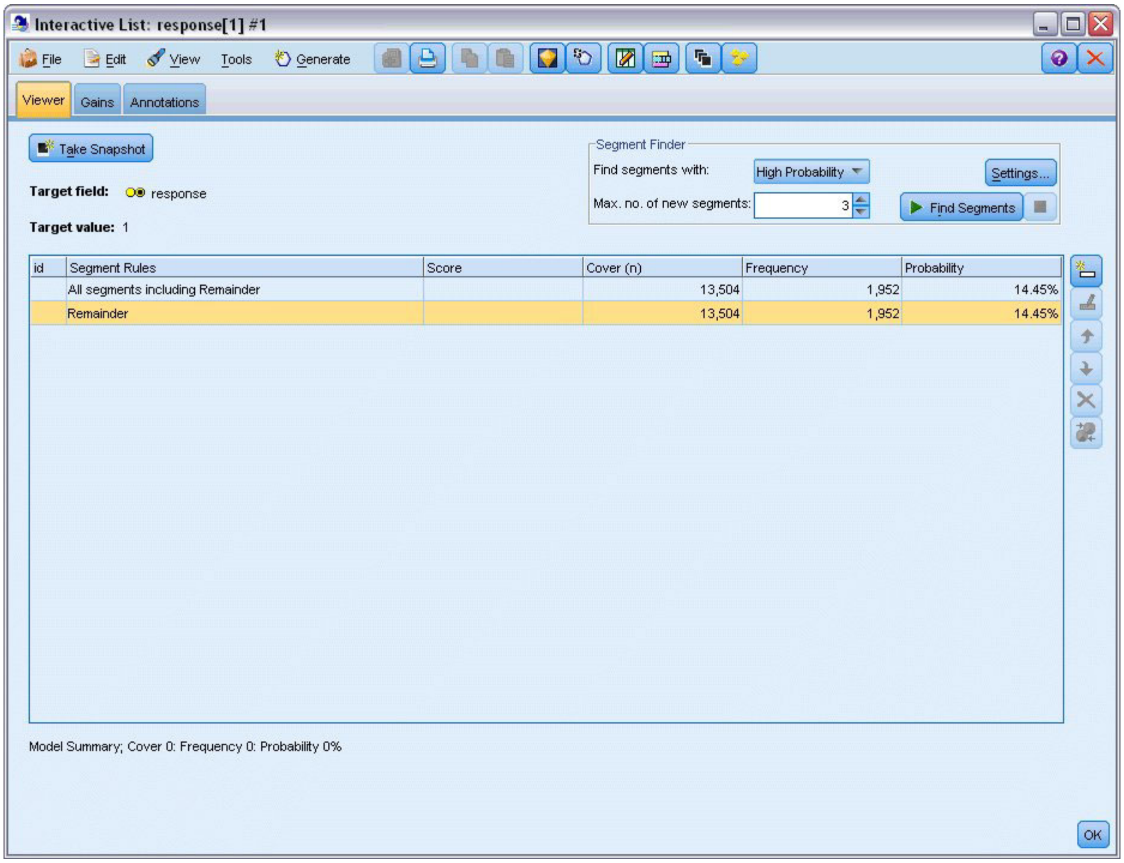


Figure 122. Interactive List viewer

Since no segments have yet been defined, all records fall under the remainder. Out of 13,504 records in the sample, 1,952 said *Yes*, for an overall hit rate of 14.45%. You want to improve on this rate by identifying segments of customers more (or less) likely to give a favorable response.

8. In the Interactive List viewer, from the menus choose:

   Tools > **Find Segments**

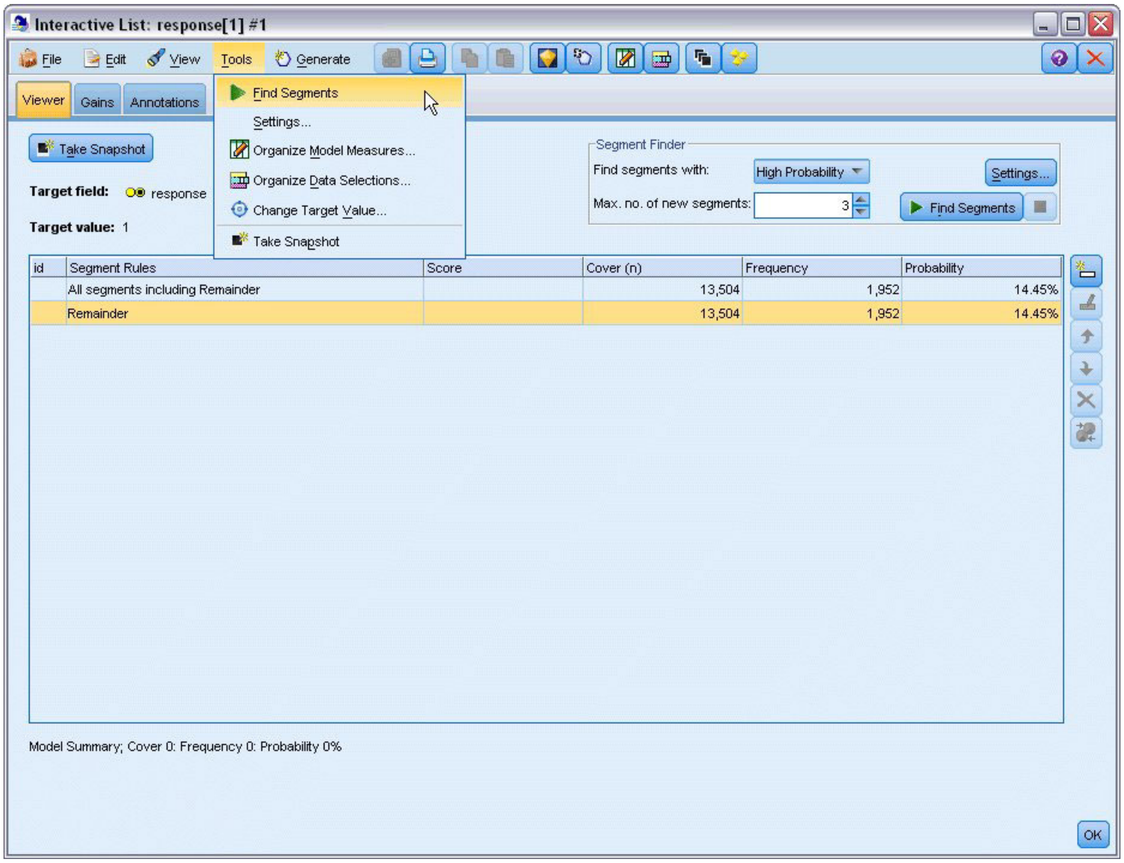


Figure 123. Interactive List viewer

This runs the default mining task based on the settings you specified in the Decision List node. The completed task returns three alternative models, which are listed in the Alternatives tab of the Model Albums dialog box.

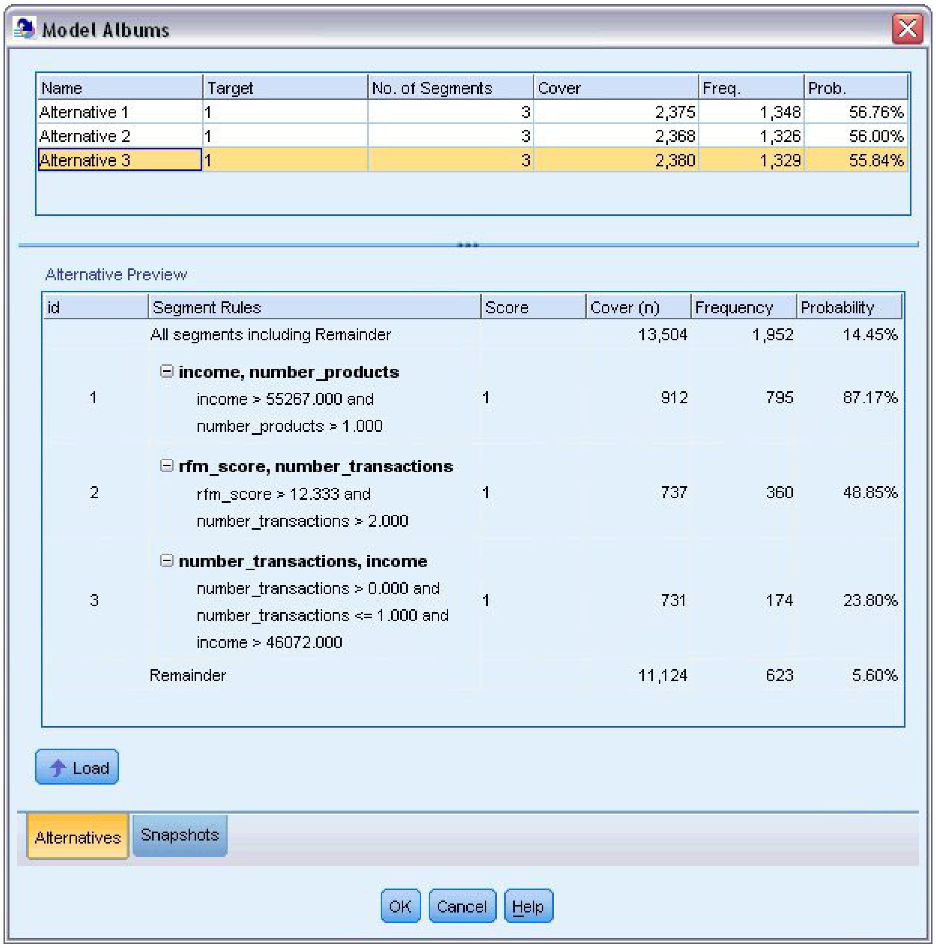


Figure 124. Available alternative models

9. Select the first alternative from the list; its details are shown in the Alternative Preview panel.

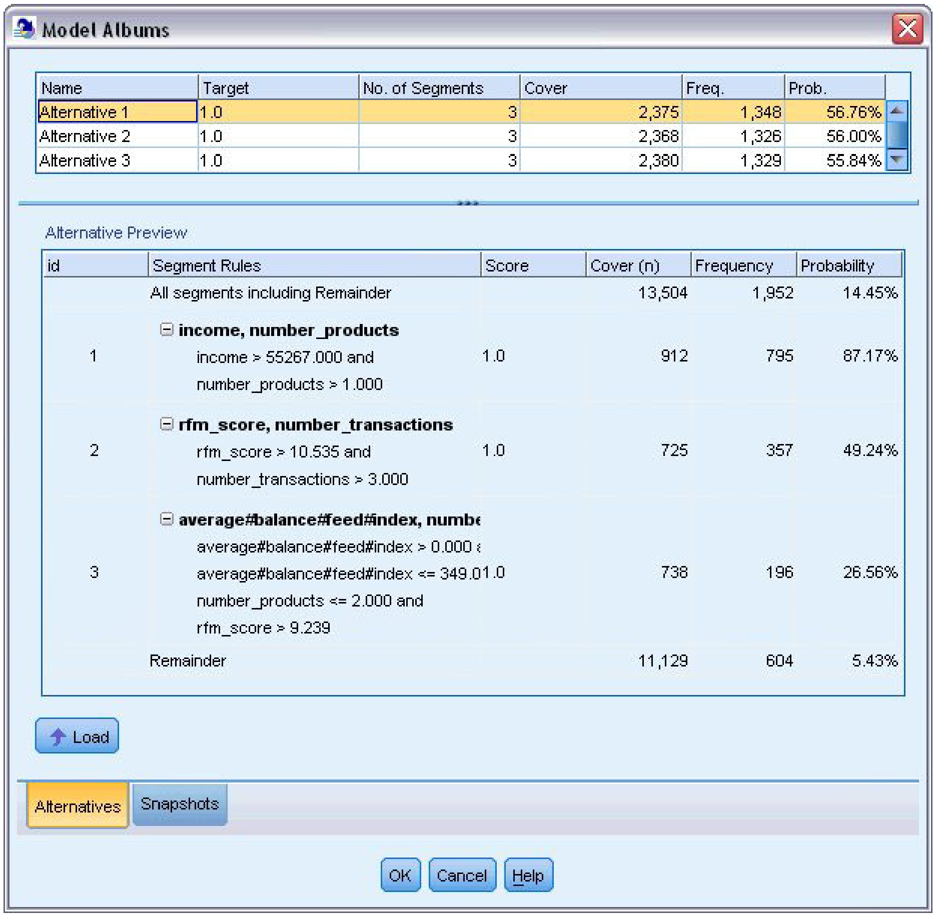


Figure 125. Alternative model selected

The Alternative Preview panel allows you to quickly browse any number of alternatives without changing the working model, making it easy to experiment with different approaches.

Note: To get a better look at the model, you may want to maximize the Alternative Preview panel within the dialog, as shown here. You can do this by dragging the panel border.

Using rules based on predictors, such as income, number of transactions per month, and RFM score, the model identifies segments with response rates that are higher than those for the sample overall. When the segments are combined, this model suggests that you could improve your hit rate to 56.76%. However, the model covers only a small portion of the overall sample, leaving over 11,000 records—with several hundred hits among them—to fall under the remainder. You want a model that will capture more of these hits while still excluding the low-performing segments.

10. To try a different modeling approach, from the menus choose:

    Tools > **Settings**

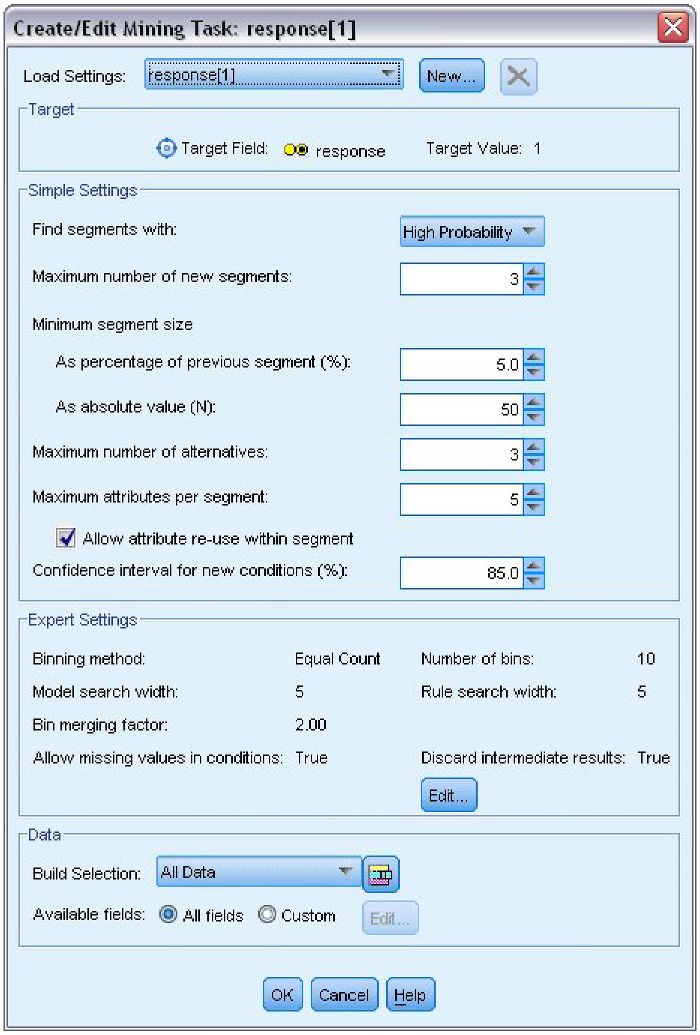


Figure 126. Create/Edit Mining Task dialog box

11. Click the **New** button (upper right corner) to create a second mining task, and specify *Down Search* as the task name in the New Settings dialog box.

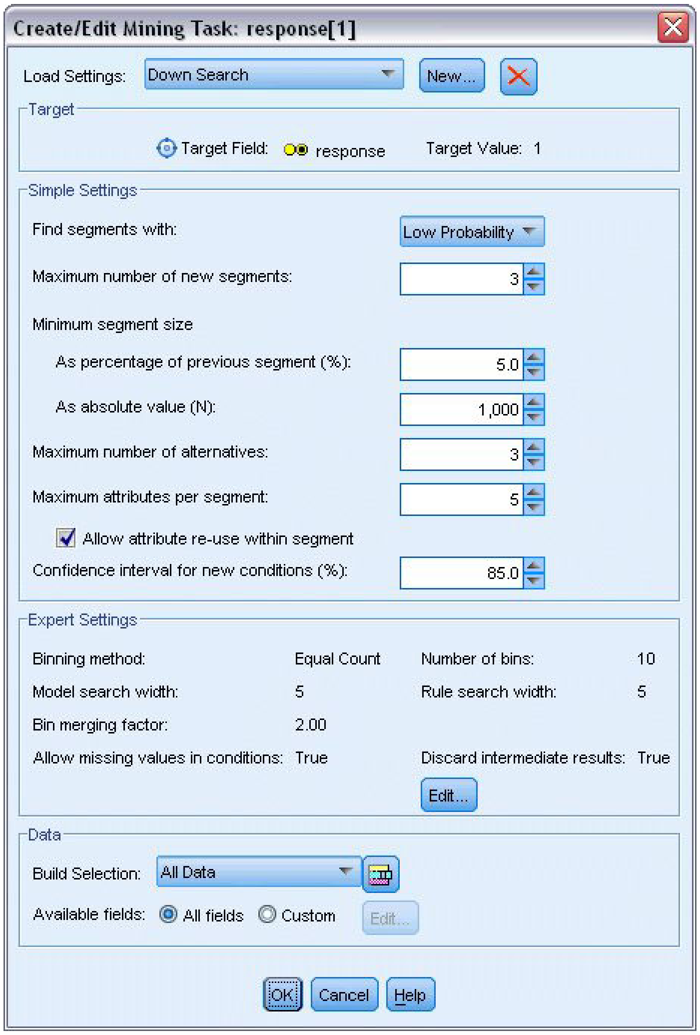


Figure 127. Create/Edit Mining Task dialog box

12. Change the search direction to **Low probability** for the task. This will cause the algorithm to search for segments with the *lowest* response rates rather than the highest.
13. Increase the minimum segment size to 1,000. Click **OK** to return to the Interactive List viewer.
14. In Interactive List viewer, make sure that the *Segment Finder* panel is displaying the new task details and click **Find Segments**.

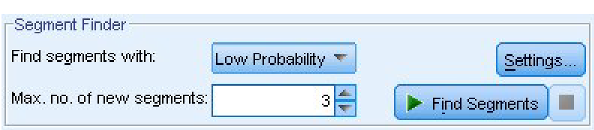


Figure 128. Find segments in new mining task

The task returns a new set of alternatives, which are displayed in the Alternatives tab of the Model Albums dialog box and can be previewed in the same manner as previous results.

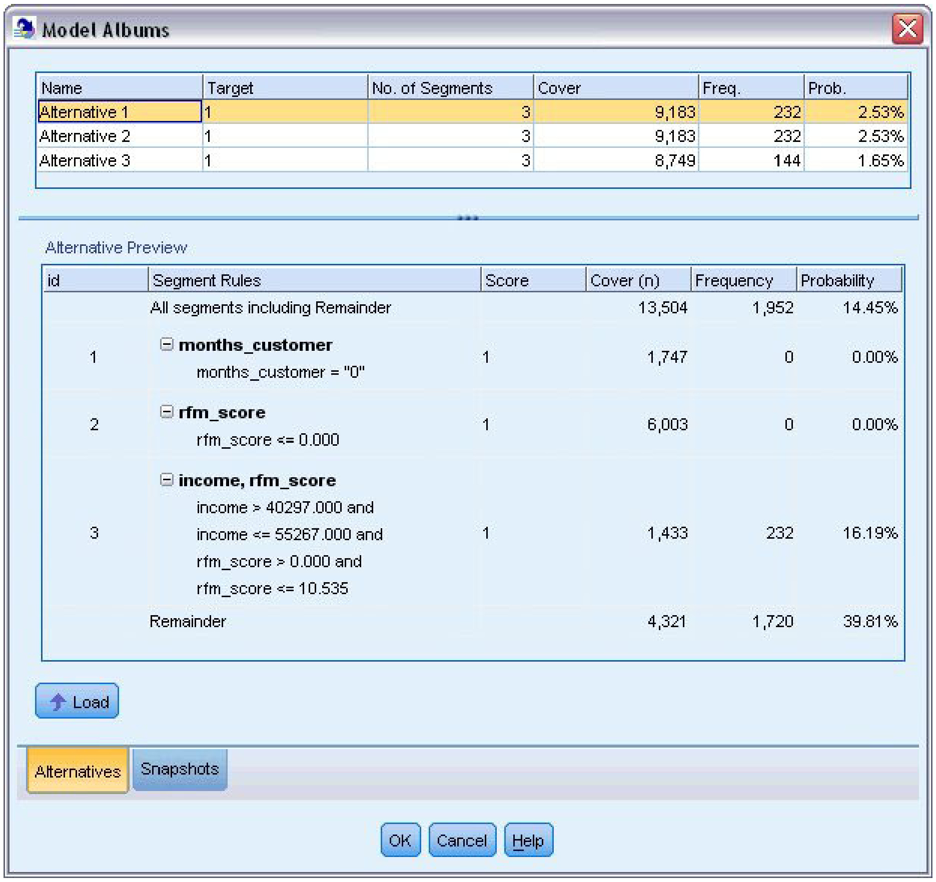


Figure 129. Down Search model results

This time each model identifies segments with low response probabilities rather than high. Looking at the first alternative, simply excluding these segments will increase the hit rate for the remainder to 39.81%. This is lower than the model you looked at earlier but with higher coverage (meaning more total hits).

By combining the two approaches—using a Low Probability search to weed out uninteresting records, followed by a High Probability search—you may be able to improve this result.

15. Click **Load** to make this (the first Down Search alternative) the working model and click **OK** to close the Model Albums dialog box.

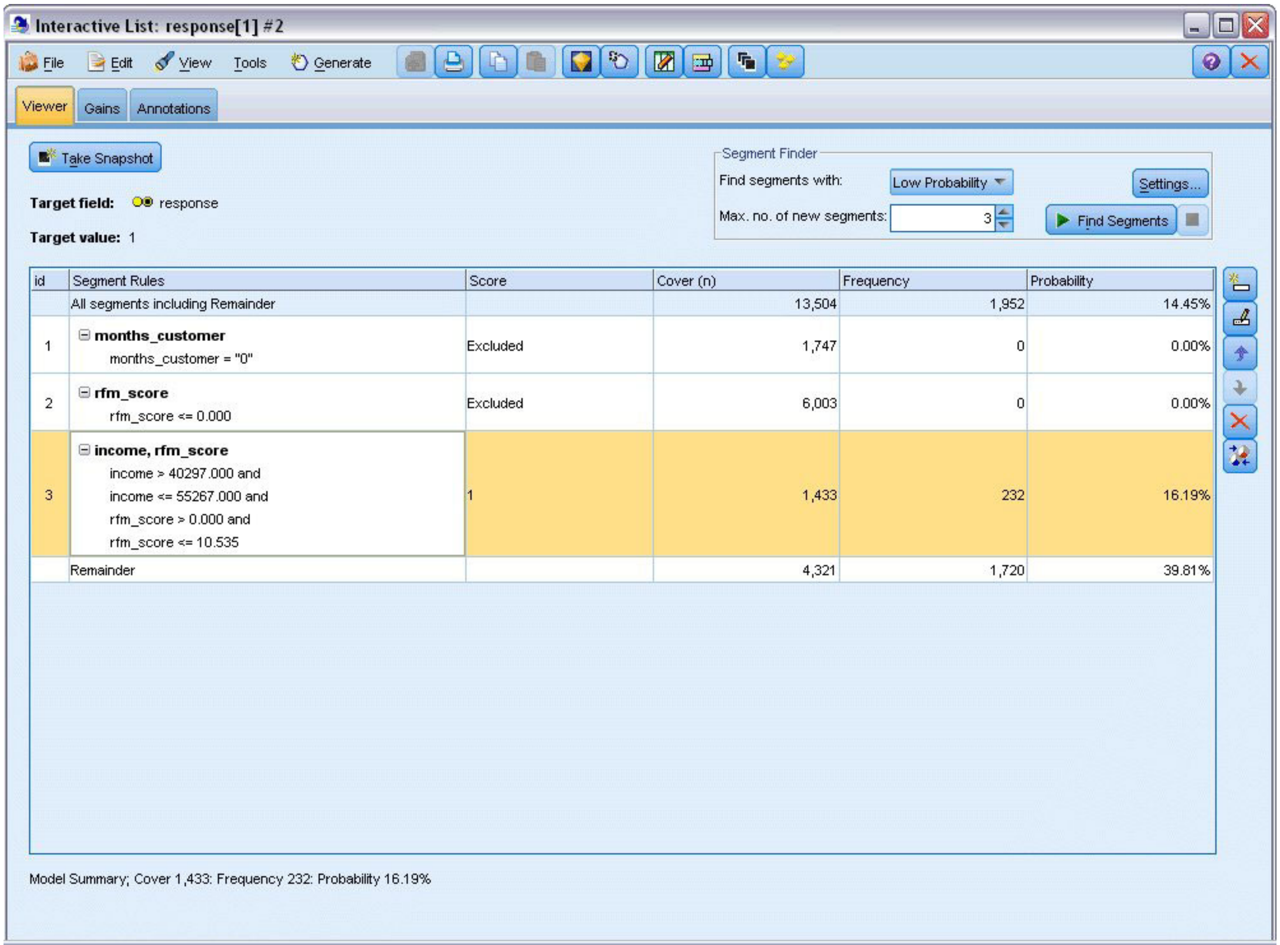


Figure 130. Excluding a segment

16. Right-click on each of the first two segments and select **Exclude Segment**. Together, these segments capture almost 8,000 records with zero hits between them, so it makes sense to exclude them from future offers. (Excluded segments will be scored as null to indicate this.)
17. Right-click on the third segment and select **Delete Segment**. At 16.19%, the hit rate for this segment is not that different than the baseline rate of 14.45%, so it doesn't add enough information to justify keeping it in place.

    Note: Deleting a segment is not the same as excluding it. Excluding a segment simply changes how it is scored, while deleting it removes it from the model entirely.

    Having excluded the lowest-performing segments, you can now search for high-performing segments in the remainder.
18. Click on the remainder row in the table to select it, so that the next mining task will apply to the remainder only.

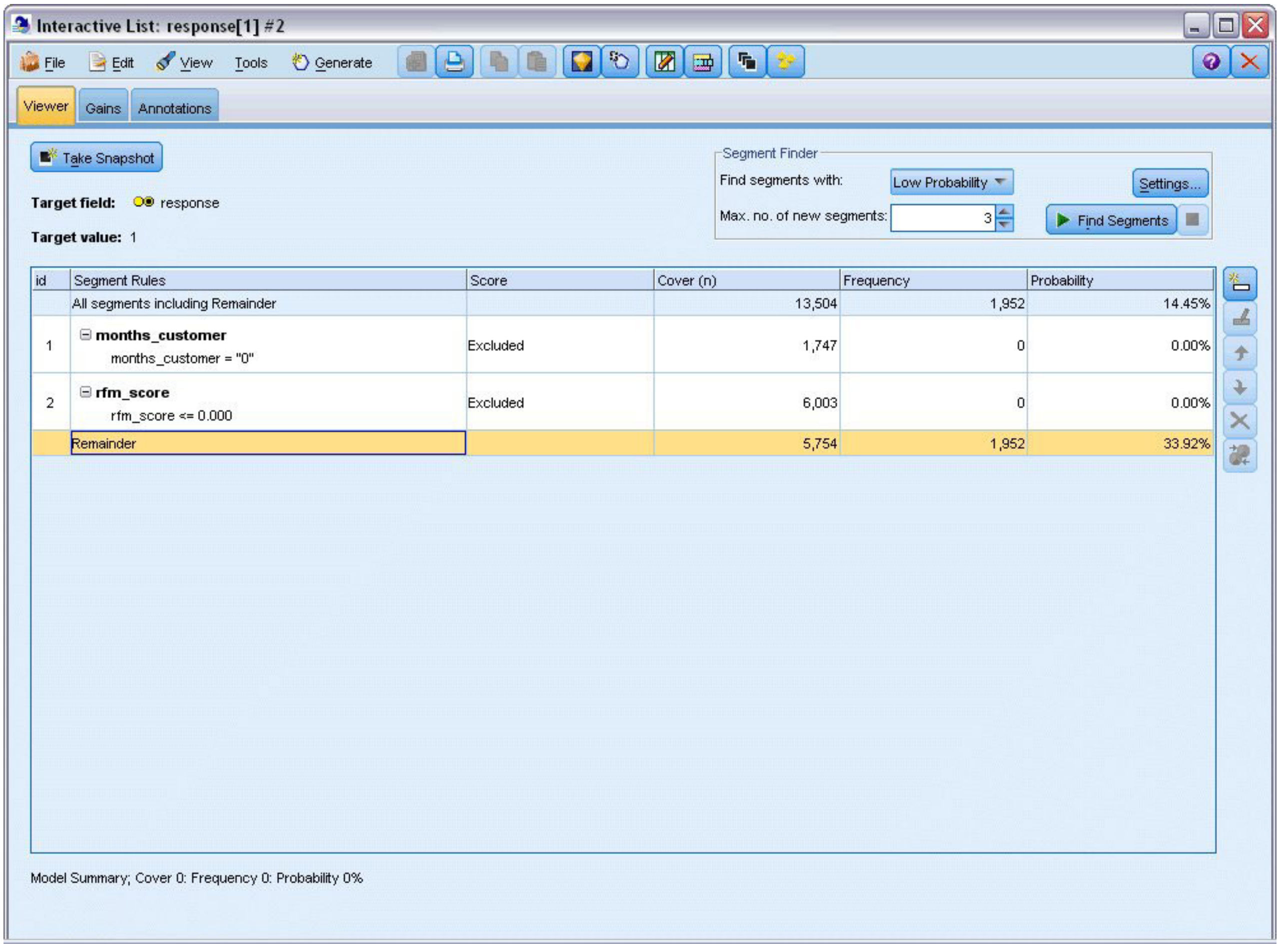


Figure 131. Selecting a segment

19. With the remainder selected, click **Settings** to reopen the Create/Edit Mining Task dialog box.
20. At the top, in **Load Settings**, select the default mining task: **response[1]**.
21. Edit the **Simple Settings** to increase the number of new segments to 5 and the minimum segment size to 500.
22. Click **OK** to return to the Interactive List viewer.

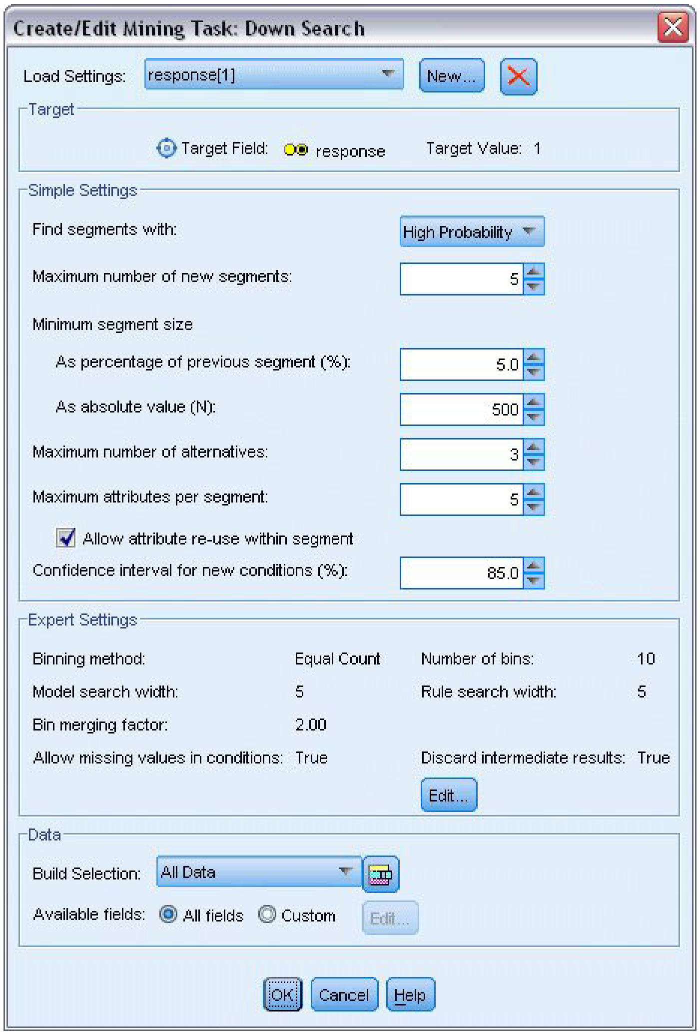
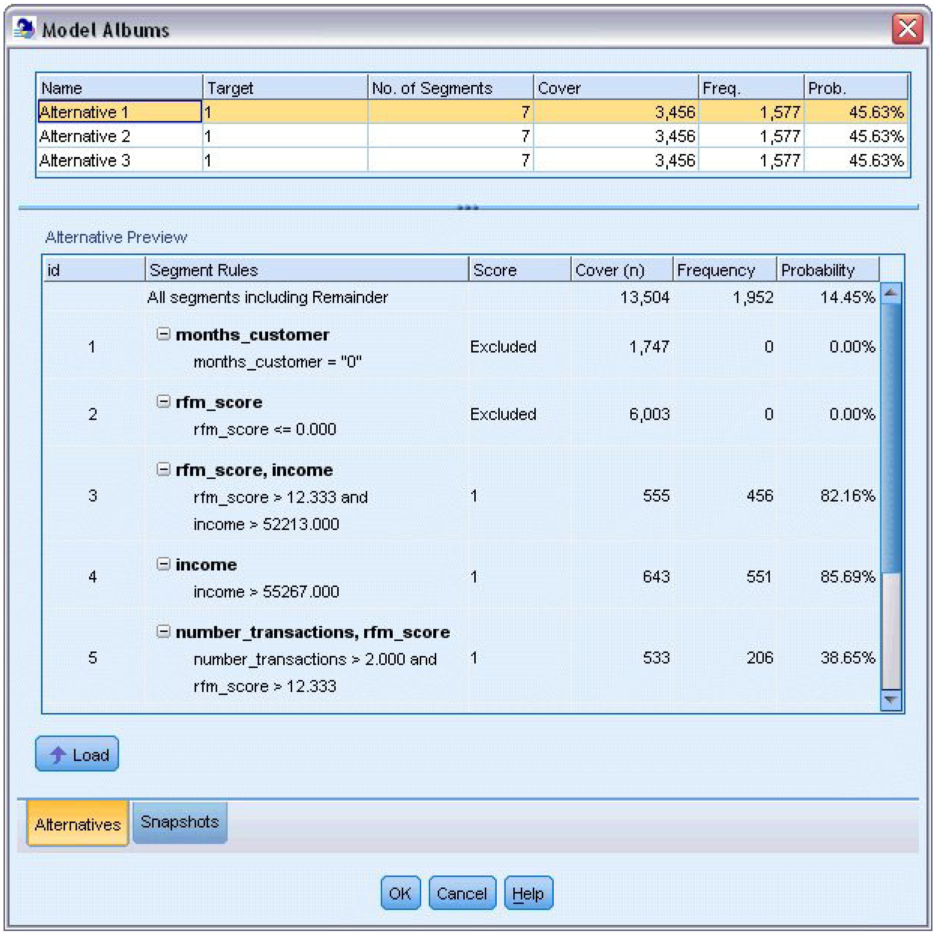


Figure 132. Selecting the default mining task

23. Click **Find Segments**.

    This displays yet another set of alternative models. By feeding the results of one mining task into another, these latest models contain a mix of high- and low-performing segments. Segments with low response rates are excluded, which means that they will be scored as null, while included segments will be scored as 1. The overall statistics reflect these exclusions, with the first alternative model showing a hit rate of 45.63%, with higher coverage (1,577 hits out of 3,456 records) than any of the previous models.



Name	Target	No. of Segments	Cover	Freq.	Prob.
Alternative 1	1	7	3,456	1,577	45.63%
Alternative 2	1	7	3,456	1,577	45.63%
Alternative 3	1	7	3,456	1,577	45.63%

Alternative Preview

id	Segment Rules	Score	Cover (n)	Frequency	Probability
	All segments including Remainder		13,504	1,952	14.45%
1	**months_customer** months_customer = "0"	Excluded	1,747	0	0.00%
2	**rfm_score** rfm_score <= 0.000	Excluded	6,003	0	0.00%
3	**rfm_score, income** rfm_score > 12.333 and income > 52213.000	1	555	456	82.16%
4	**income** income > 55267.000	1	643	551	85.69%
5	**number_transactions, rfm_score** number_transactions > 2.000 and rfm_score > 12.333	1	533	206	38.65%

Figure 133. Alternatives for combined model

24. Preview the first alternative and then click **Load** to make it the working model.

Calculating Custom Measures Using Excel

1. To gain a bit more insight as to how the model performs in practical terms, choose **Organize Model Measures** from the Tools menu.

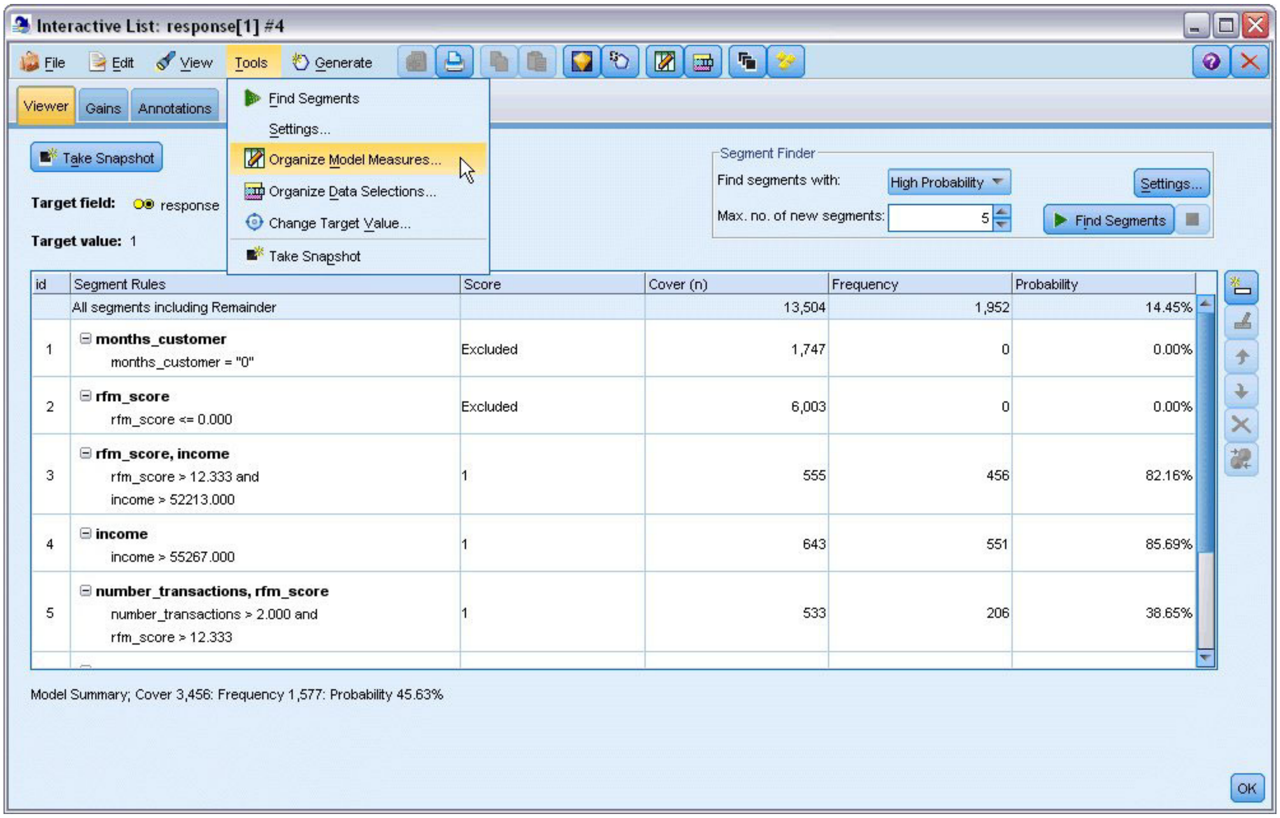


Figure 134. Organizing model measures

The Organize Model Measures dialog box allows you to choose the measures (or columns) to show in the Interactive List viewer. You can also specify whether measures are computed against all records or a selected subset, and you can choose to display a pie chart rather than a number, where applicable.

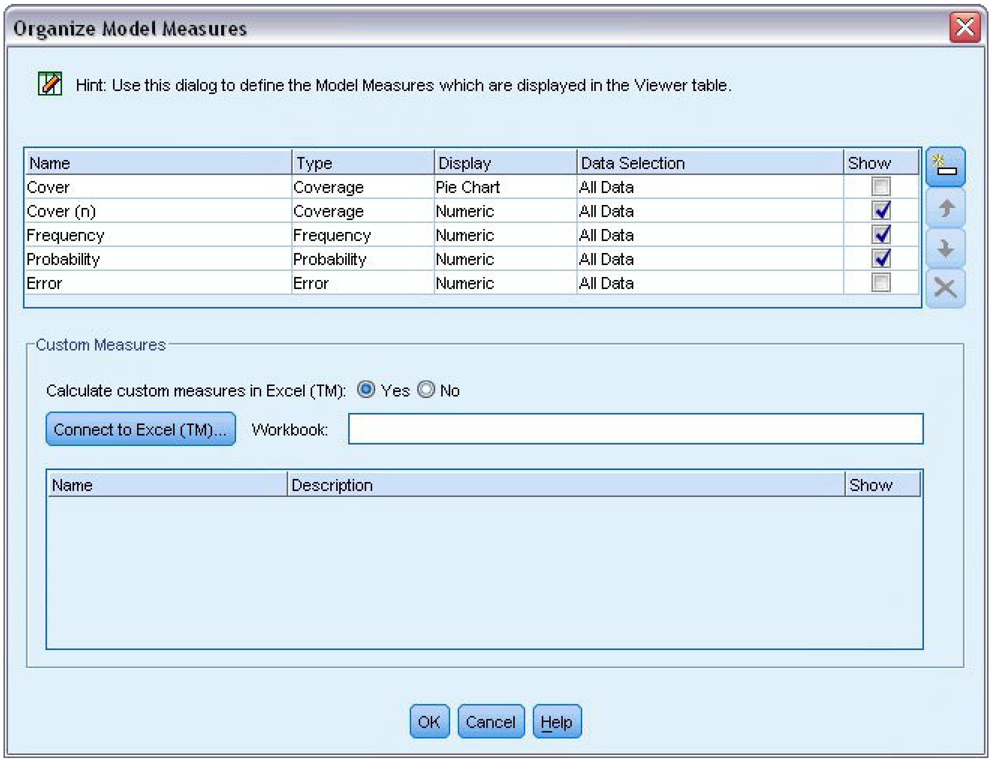


Figure 135. Organize Model Measures dialog box

In addition, if you have Microsoft Excel installed, you can link to an Excel template that will calculate custom measures and add them to the interactive display.

2. In the Organize Model Measures dialog box, set **Calculate custom measures in Excel (TM)** to **Yes**.
3. Click **Connect to Excel (TM)**
4. Select the *template_profit.xlt* workbook, located under *streams* in the *Demos* folder of your IBM SPSS Modeler installation, and click **Open** to launch the spreadsheet.

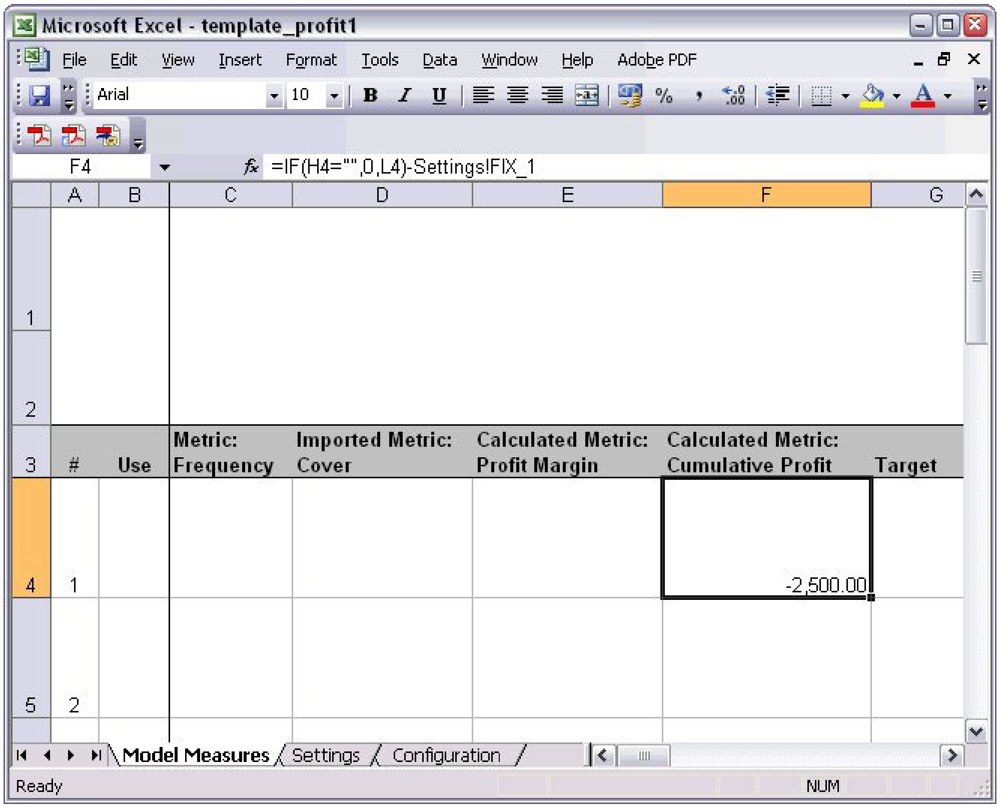


Figure 136. Excel Model Measures worksheet

The Excel template contains three worksheets:

- **Model Measures** displays model measures imported from the model and calculates custom measures for export back to the model.
- **Settings** contains parameters to be used in calculating custom measures.
- **Configuration** defines the measures to be imported from and exported to the model.

The metrics exported back to the model are:

- **Profit Margin.** Net revenue from the segment
- **Cumulative Profit.** Total profit from campaign

As defined by the following formulas:

```
Profit Margin = Frequency * Revenue per respondent - Cover * Variable cost
```

```
Cumulative Profit = Total Profit Margin - Fixed cost
```

Note that Frequency and Cover are imported from the model.

The cost and revenue parameters are specified by the user on the Settings worksheet.

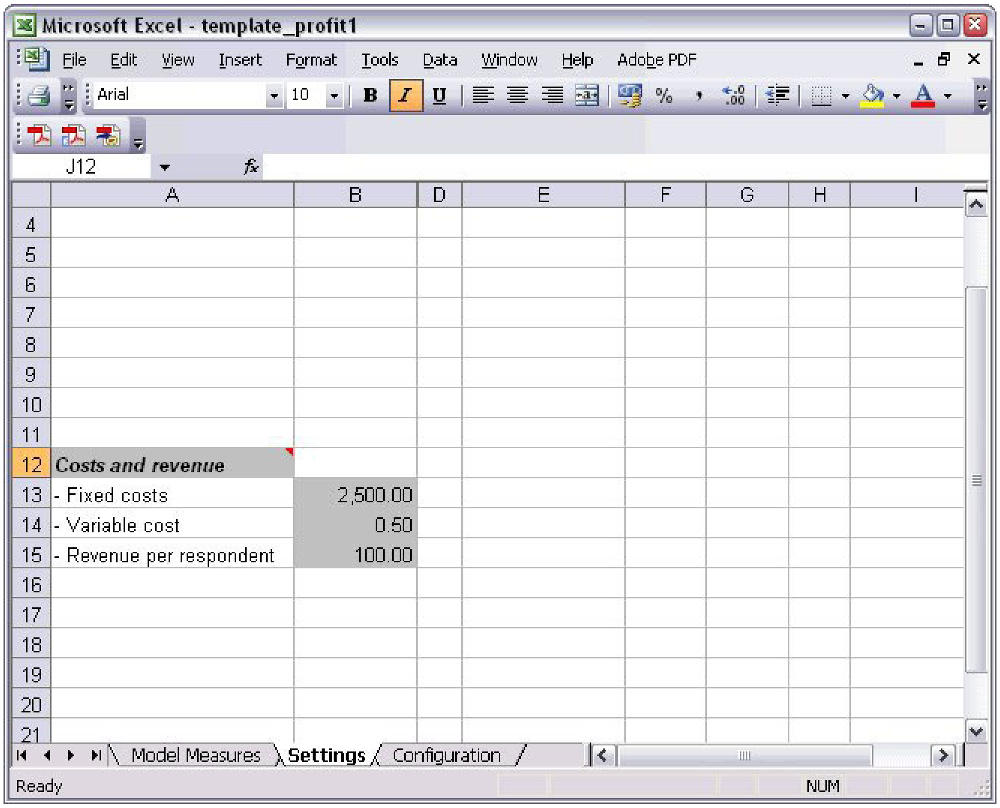


Figure 137. Excel Settings worksheet

Fixed cost is the setup cost for the campaign, such as design and planning.

Variable cost is the cost of extending the offer to each customer, such as envelopes and stamps.

Revenue per respondent is the net revenue from a customer who responds to the offer.

5. To complete the link back to the model, use the Windows taskbar (or press Alt+Tab) to navigate back to the Interactive List viewer.

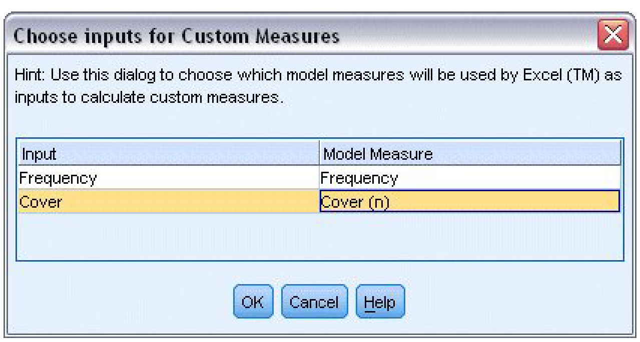


Figure 138. Choosing inputs for custom measures

The Choose Inputs for Custom Measures dialog box is displayed, allowing you to map inputs from the model to specific parameters defined in the template. The left column lists the available measures, and the right column maps these to spreadsheet parameters as defined in the Configuration worksheet.

6. In the **Model Measures** column, select **Frequency** and **Cover (n)** against the respective inputs and click **OK**.

In this case, the parameter names in the template—Frequency and Cover (n)—happen to match the inputs, but different names could also be used.

7. Click **OK** in the Organize Model Measures dialog box to update the Interactive List viewer.

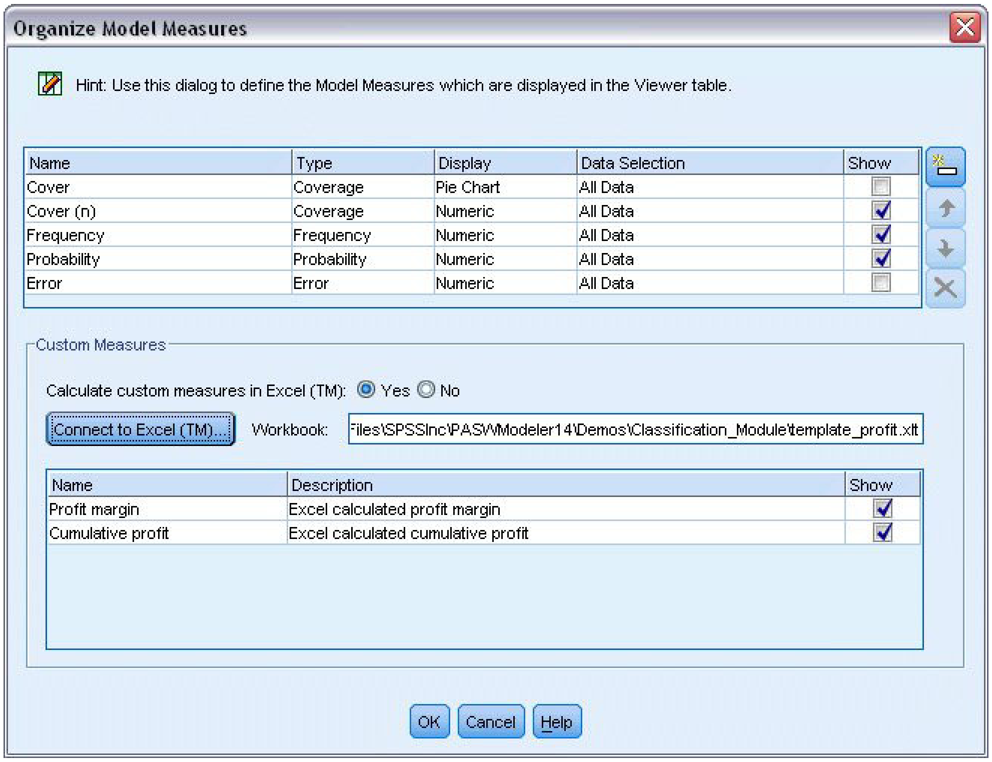


Figure 139. Organize Model Measures dialog box showing custom measures from Excel

The new measures are now added as new columns in the window and will be recalculated each time the model is updated.

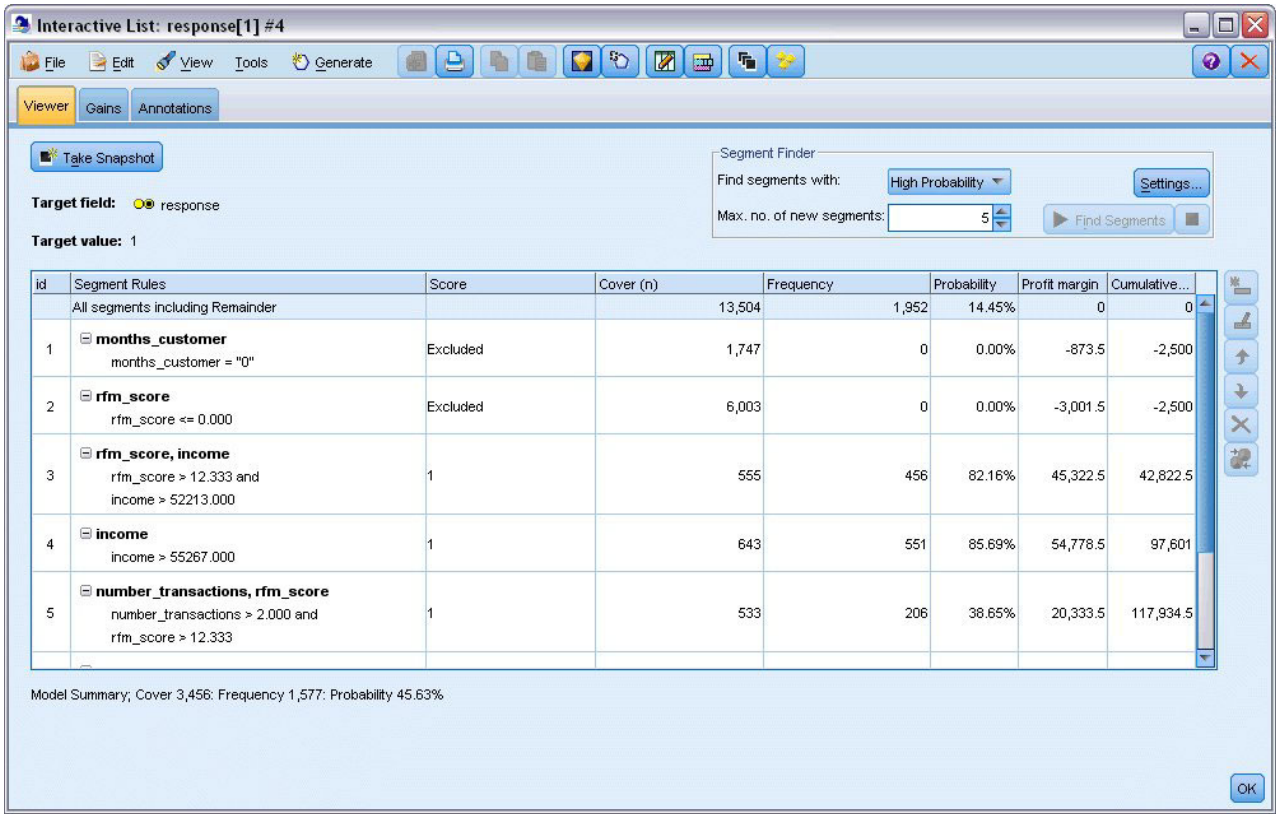


Figure 140. Custom measures from Excel displayed in the Interactive List viewer

By editing the Excel template, any number of custom measures can be created.

Modifying the Excel template

Although IBM SPSS Modeler is supplied with a default Excel template to use with the Interactive List viewer, you may want to change the settings or add your own. For example, the costs in the template may be incorrect for your organization and need amending.

Note: If you do modify an existing template, or create you own, remember to save the file with an Excel 2003 *.xlt* suffix.

To modify the default template with new cost and revenue details and update the Interactive List viewer with the new figures:

1. In the Interactive List viewer, choose **Organize Model Measures** from the Tools menu.
2. In the Organize Model Measures dialog box, click **Connect to Excel™.**
3. Select the *template_profit.xlt* workbook, and click **Open** to launch the spreadsheet.
4. Select the Settings worksheet.
5. Edit the **Fixed costs** to be 3,250.00, and the **Revenue per respondent** to be 150.00.

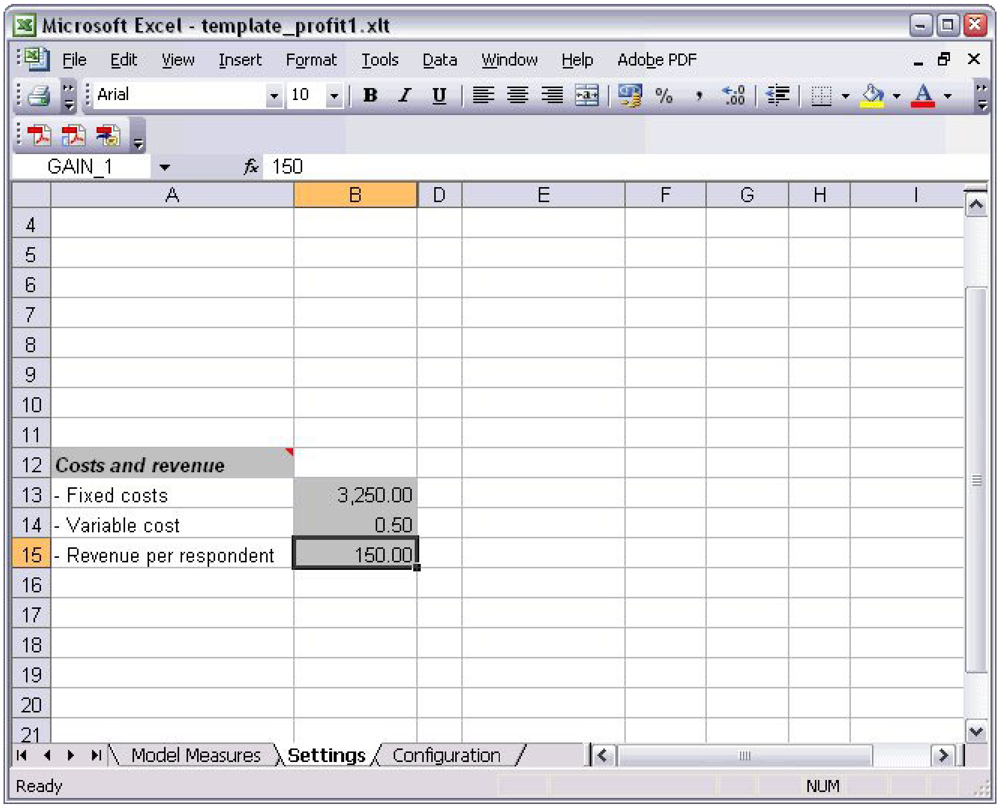


Figure 141. Modified values on Excel Settings worksheet

6. Save the modified template with a unique, relevant filename. Ensure it has an Excel 2003 *.xlt* extension.

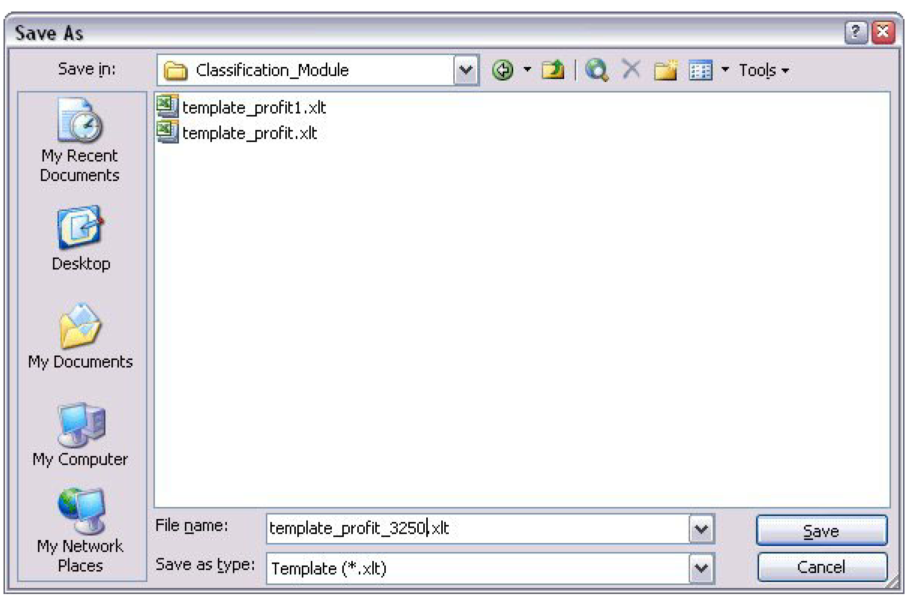


Figure 142. Saving modified Excel template

7. Use the Windows taskbar (or press Alt+Tab) to navigate back to the Interactive List viewer.

   In the Choose Inputs for Custom Measures dialog box, select the measures you want to display and click **OK**.

8. In the Organize Model Measures dialog box, click **OK** to update the Interactive List viewer.

Obviously, this example has only shown one simple way of modifying the Excel template; you can make further changes that pull data from, and pass data to, the Interactive List viewer, or work within Excel to produce other output, such as graphs.

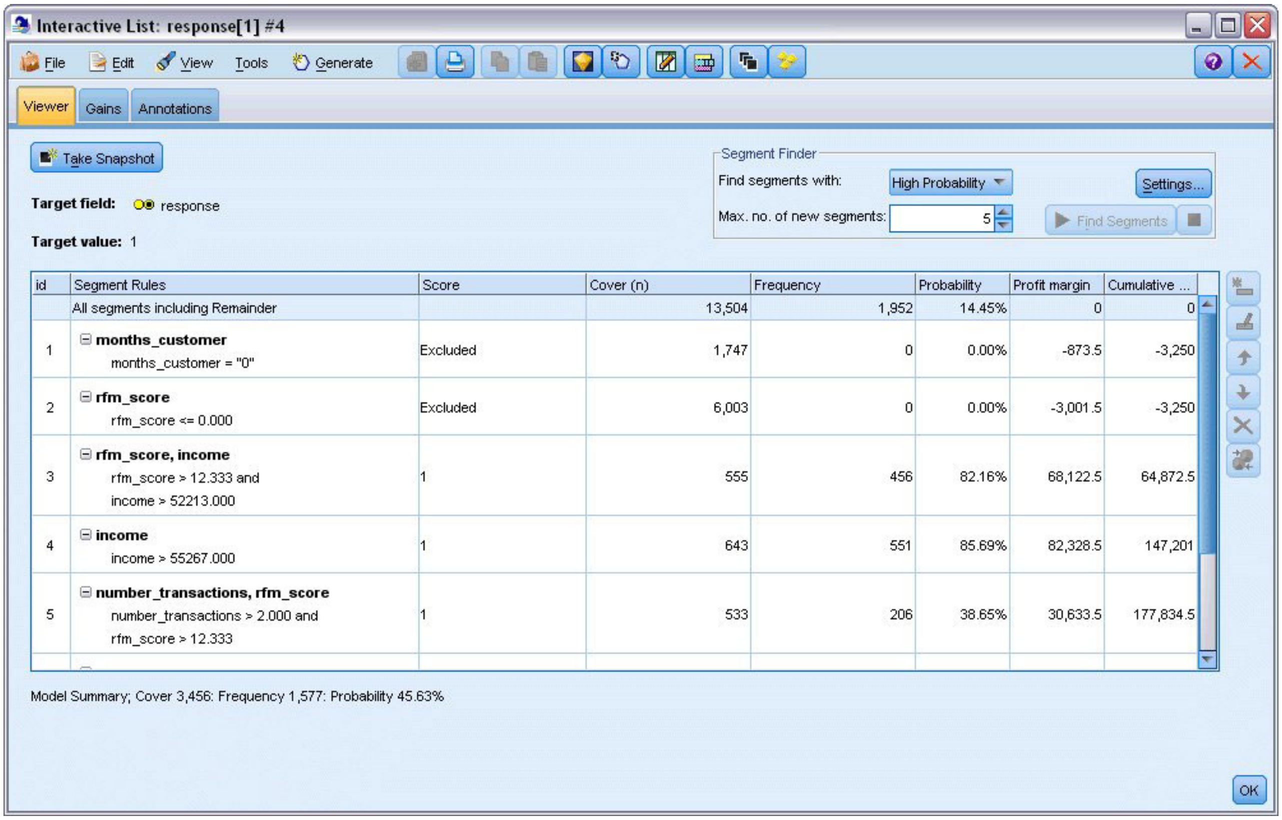


Figure 143. Modified custom measures from Excel displayed in the Interactive List viewer

Saving the Results

To save a model for later use during your interactive session, you can take a snapshot of the model, which will be listed on the Snapshots tab. You can return to any saved snapshot at any time during the interactive session.

Continuing in this manner, you can experiment with additional mining tasks to search for additional segments. You can also edit existing segments, insert custom segments based on your own business rules, create data selections to optimize the model for specific groups, and customize the model in a number of other ways. Finally, you can explicitly include or exclude each segment as appropriate to specify how each will be scored.

When you are satisfied with your results, you can use the Generate menu to generate a model that can be added to streams or deployed for purposes of scoring.

Alternatively, to save the current state of your interactive session for another day, choose **Update Modeling Node** from the File menu. This will update the Decision List modeling node with the current settings, including mining tasks, model snapshots, data selections, and custom measures. The next time you run the stream, just make sure that **Use saved session information** is selected in the Decision List modeling node to restore the session to its current state.

Chapter 12. Classifying Telecommunications Customers (Multinomial Logistic Regression)

Logistic regression is a statistical technique for classifying records based on values of input fields. It is analogous to linear regression but takes a categorical target field instead of a numeric one.

For example, suppose a telecommunications provider has segmented its customer base by service usage patterns, categorizing the customers into four groups. If demographic data can be used to predict group membership, you can customize offers for individual prospective customers.

This example uses the stream named *telco_custcat.str*, which references the data file named *telco.sav*. These files are available from the *Demos* directory of any IBM SPSS Modeler installation. This can be accessed from the IBM SPSS Modeler program group on the Windows Start menu. The *telco_custcat.str* file is in the *streams* directory.

The example focuses on using demographic data to predict usage patterns. The target field *custcat* has four possible values that correspond to the four customer groups, as follows:

Value	Label
1	Basic Service
2	E-Service
3	Plus Service
4	Total Service

Because the target has multiple categories, a multinomial model is used. In the case of a target with two distinct categories, such as yes/no, true/false, or churn/don't churn, a binomial model could be created instead. See the topic Chapter 13, "Telecommunications Churn (Binomial Logistic Regression)," on page 137 for more information.

Building the Stream

1. Add a Statistics File source node pointing to *telco.sav* in the *Demos* folder.

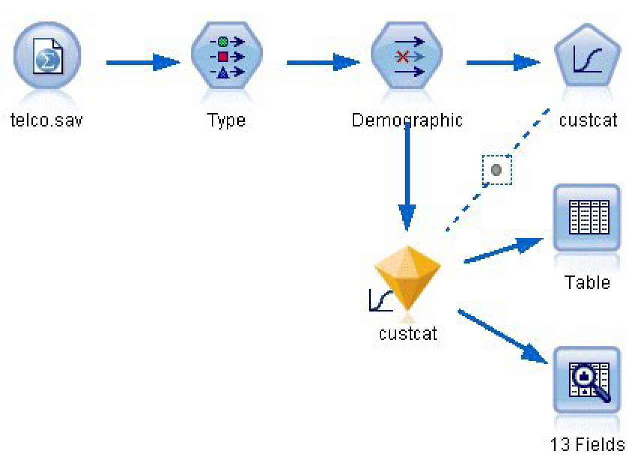


Figure 144. Sample stream to classify customers using multinomial logistic regression

a. Add a Type node and click **Read Values**, making sure that all measurement levels are set correctly. For example, most fields with values 0 and 1 can be regarded as flags.

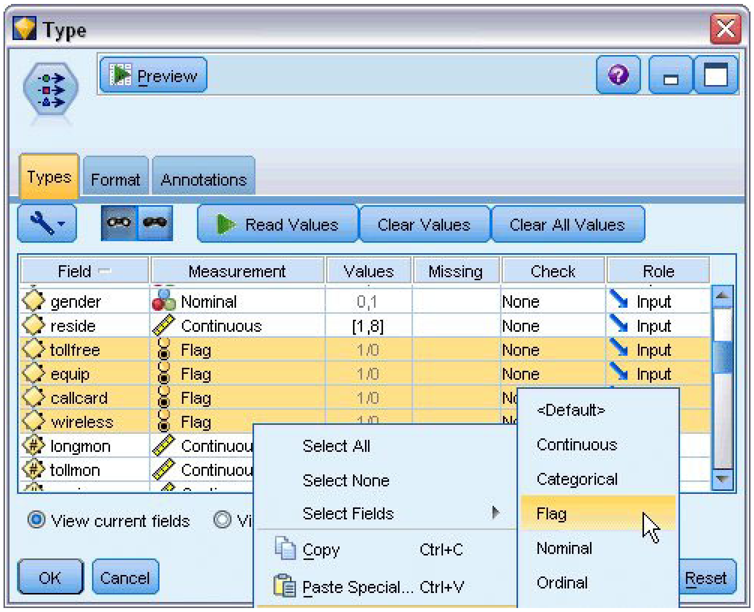


Figure 145. Setting the measurement level for multiple fields

Tip: To change properties for multiple fields with similar values (such as 0/1), click the *Values* column header to sort fields by value, and then hold down the shift key while using the mouse or arrow keys to select all the fields you want to change. You can then right-click on the selection to change the measurement level or other attributes of the selected fields.

Notice that *gender* is more correctly considered as a field with a set of two values, instead of a flag, so leave its Measurement value as **Nominal**.

b. Set the role for the *custcat* field to **Target**. All other fields should have their role set to **Input**.

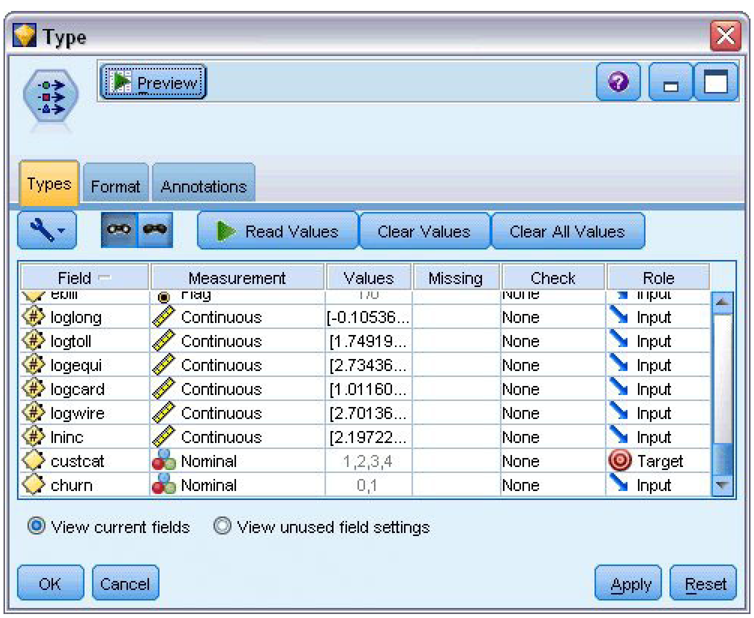


Figure 146. Setting field role

Since this example focuses on demographics, use a Filter node to include only the relevant fields (*region*, *age*, *marital*, *address*, *income*, *ed*, *employ*, *retire*, *gender*, *reside*, and *custcat*). Other fields can be excluded for the purpose of this analysis.

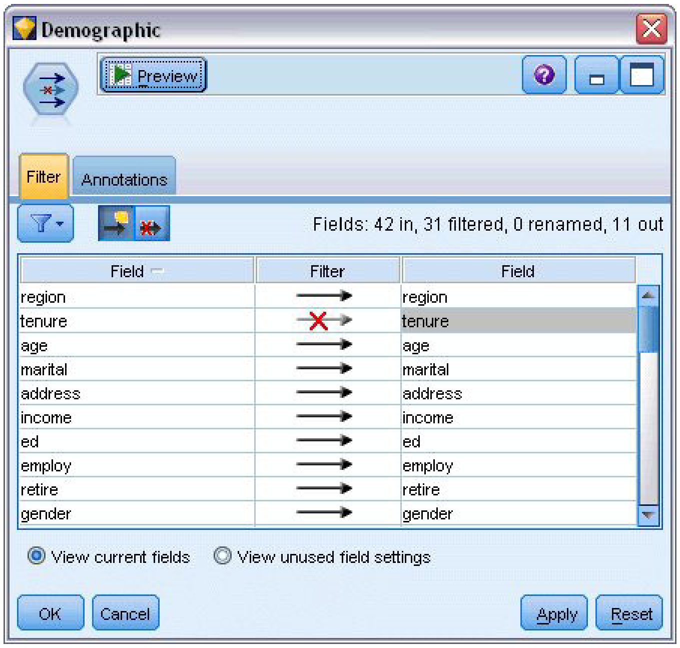


Figure 147. Filtering on demographic fields

(Alternatively, you could change the role to **None** for these fields rather than exclude them, or select the fields you want to use in the modeling node.)

2. In the Logistic node, click the **Model** tab and select the **Stepwise** method. Select **Multinomial**, **Main Effects**, and **Include constant in equation** as well.

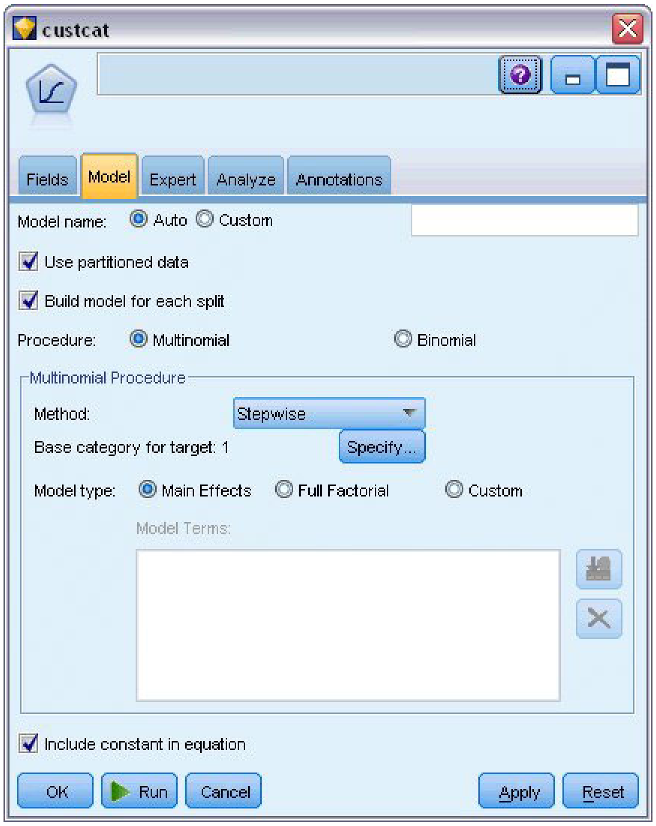


Figure 148. Choosing model options

Leave the Base category for target as 1. The model will compare other customers to those who subscribe to the Basic Service.

3. On the Expert tab, select the **Expert** mode, select **Output**, and, in the Advanced Output dialog box, select **Classification table**.

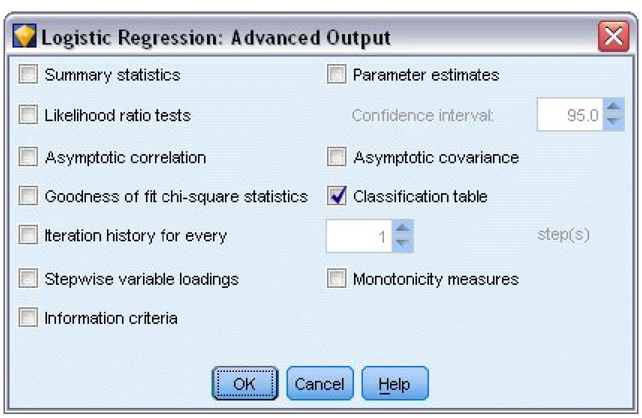


Figure 149. Choosing output options

Browsing the Model

1. Execute the node to generate the model, which is added to the Models palette in the upper-right corner. To view its details, right-click on the generated model node and choose **Browse**.

The model tab displays the equations used to assign records to each category of the target field. There are four possible categories, one of which is the base category for which no equation details are shown. Details are shown for the remaining three equations, where category 3 represents Plus Service, and so on.

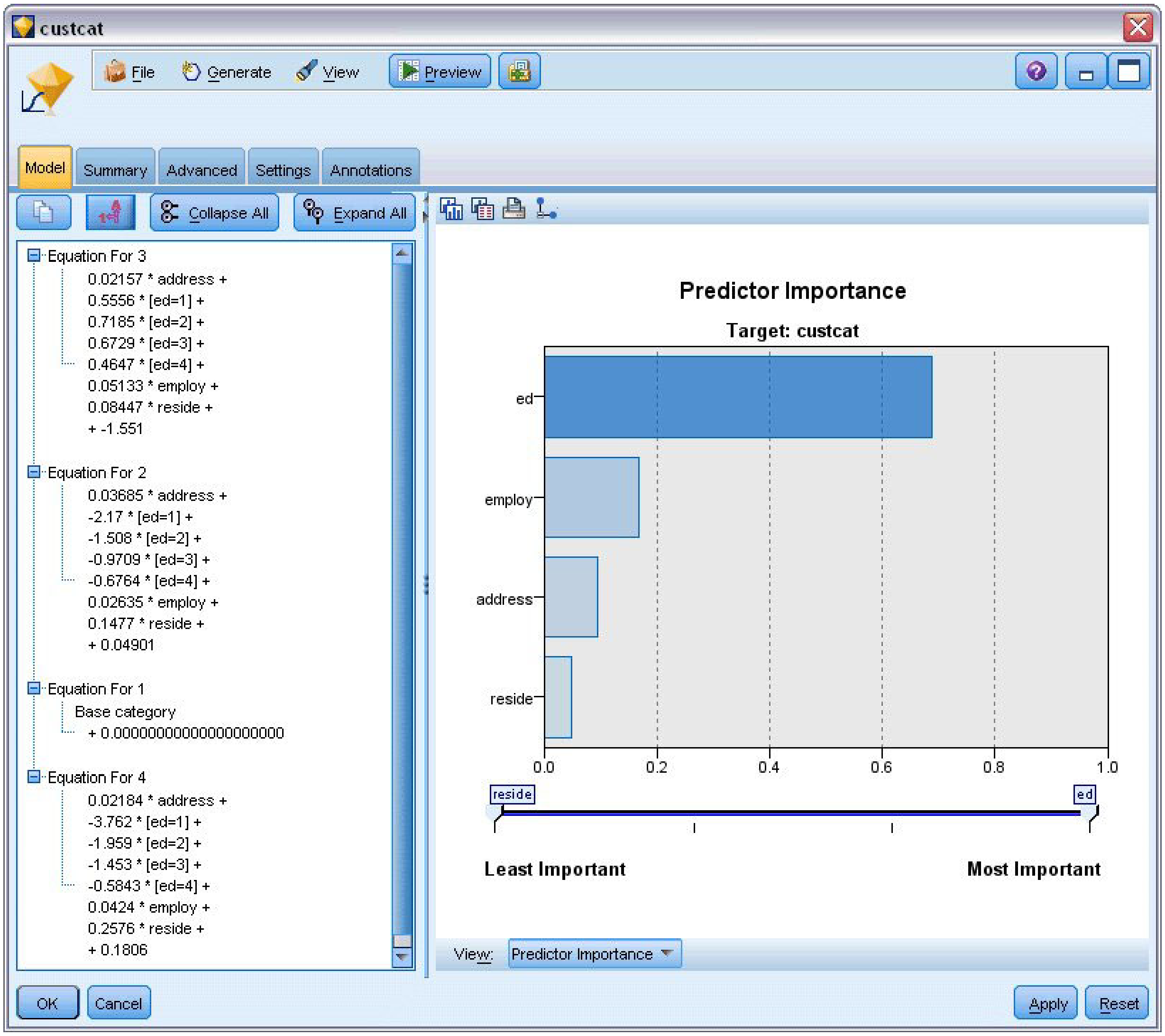


Figure 150. Browsing the model results

The Summary tab shows (among other things) the target and inputs (predictor fields) used by the model. Note that these are the fields that were actually chosen based on the Stepwise method, not the complete list submitted for consideration.

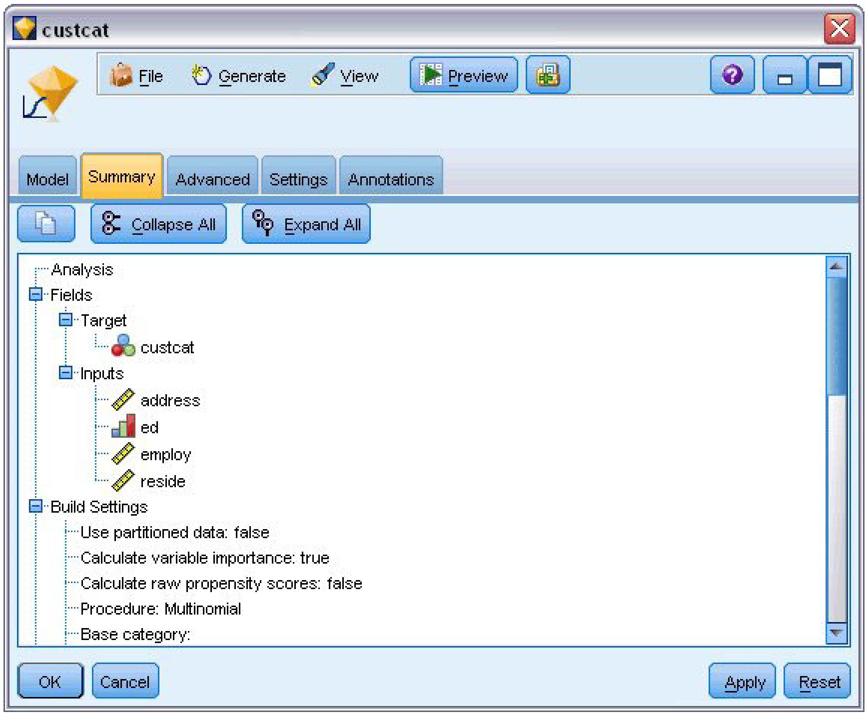


Figure 151. Model summary showing target and input fields

The items shown on the Advanced tab depend on the options selected on the Advanced Output dialog box in the modeling node.

One item that is always shown is the Case Processing Summary, which shows the percentage of records that falls into each category of the target field. This gives you a null model to use as a basis for comparison.

Without building a model that used predictors, your best guess would be to assign all customers to the most common group, which is the one for Plus service.

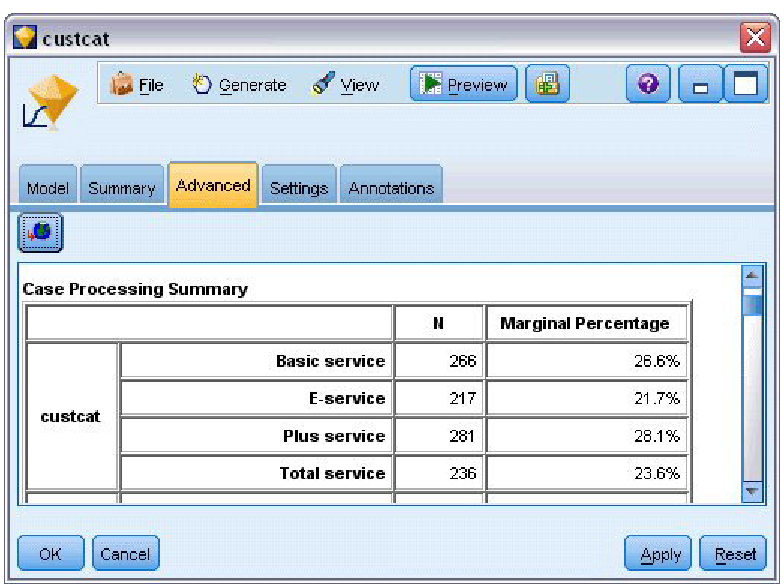
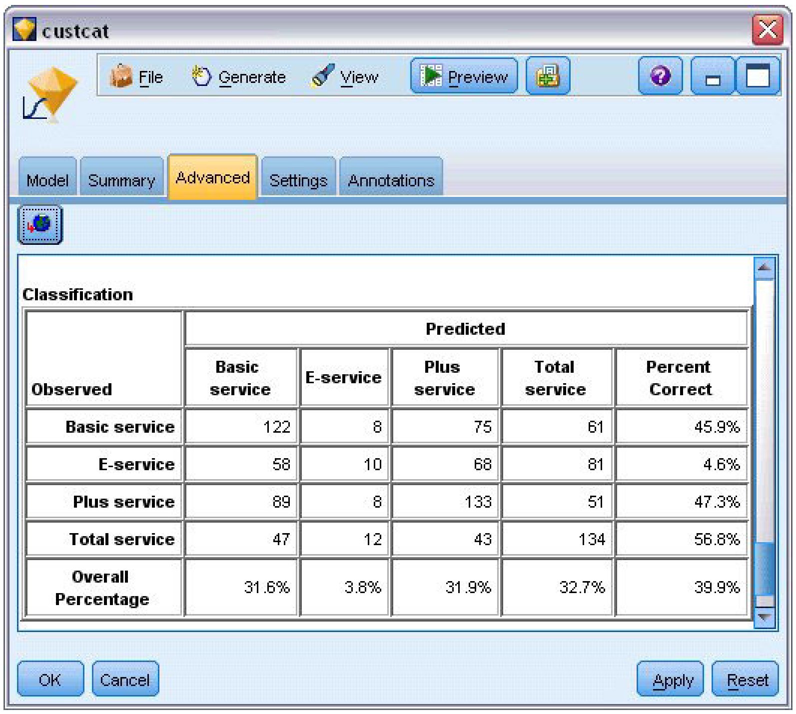


Figure 152. Case processing summary

Based on the training data, if you assigned all customers to the null model, you would be correct 281/1000 = 28.1% of the time. The Advanced tab contains further information that enables you to examine the model's predictions. You can then compare the predictions with the null model's results to see how well the model works with your data.

At the bottom of the Advanced tab, the Classification table shows the results for your model, which is correct 39.9% of the time.

In particular, your model excels at identifying Total Service customers (category 4) but does a very poor job of identifying E-service customers (category 2). If you want better accuracy for customers in category 2, you may need to find another predictor to identify them.



Observed	Predicted: Basic service	E-service	Plus service	Total service	Percent Correct
Basic service	122	8	75	61	45.9%
E-service	58	10	68	81	4.6%
Plus service	89	8	133	51	47.3%
Total service	47	12	43	134	56.8%
Overall Percentage	31.6%	3.8%	31.9%	32.7%	39.9%

Figure 153. Classification table

Depending on what you want to predict, the model may be perfectly adequate for your needs. For example, if you are not concerned with identifying customers in category 2, the model may be accurate enough for you. This may be the case where the E-service is a loss-leader that brings in little profit.

If, for example, your highest return on investment comes from customers who fall into category 3 or 4, the model may give you the information you need.

To assess how well the model actually fits the data, a number of diagnostics are available in the Advanced Output dialog box when you are building the model. Explanations of the mathematical foundations of the modeling methods used in IBM SPSS Modeler are listed in the *IBM SPSS Modeler Algorithms Guide*, available from the *Documentation* directory of the installation disk.

Note also that these results are based on the training data only. To assess how well the model generalizes to other data in the real world, you can use a Partition node to hold out a subset of records for purposes of testing and validation.

Chapter 13. Telecommunications Churn (Binomial Logistic Regression)

Logistic regression is a statistical technique for classifying records based on values of input fields. It is analogous to linear regression but takes a categorical target field instead of a numeric one.

This example uses the stream named *telco_churn.str*, which references the data file named *telco.sav*. These files are available from the *Demos* directory of any IBM SPSS Modeler installation. This can be accessed from the IBM SPSS Modeler program group on the Windows Start menu. The *telco_churn.str* file is in the *streams* directory.

For example, suppose a telecommunications provider is concerned about the number of customers it is losing to competitors. If service usage data can be used to predict which customers are liable to transfer to another provider, offers can be customized to retain as many customers as possible.

This example focuses on using usage data to predict customer loss (churn). Because the target has two distinct categories, a binomial model is used. In the case of a target with multiple categories, a multinomial model could be created instead. See the topic Chapter 12, "Classifying Telecommunications Customers (Multinomial Logistic Regression)," on page 129 for more information.

Building the Stream

1. Add a Statistics File source node pointing to *telco.sav* in the *Demos* folder.

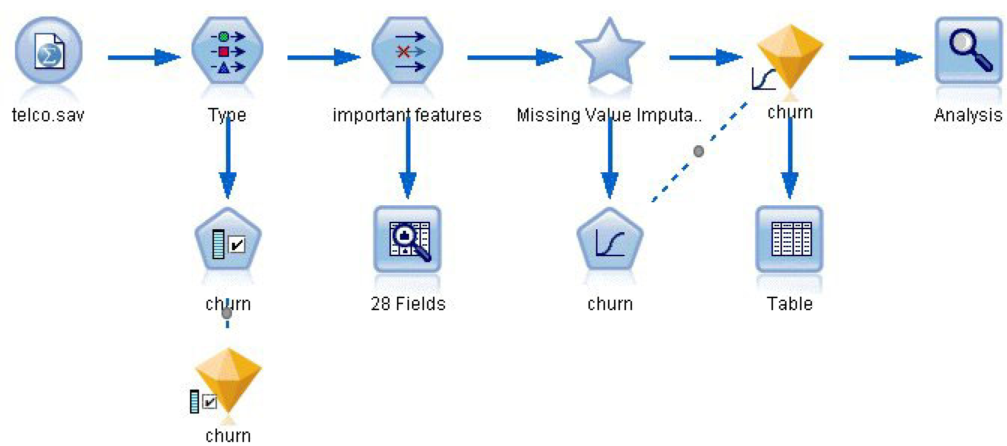


Figure 154. Sample stream to classify customers using binomial logistic regression

2. Add a Type node to define fields, making sure that all measurement levels are set correctly. For example, most fields with values 0 and 1 can be regarded as flags, but certain fields, such as gender, are more accurately viewed as a nominal field with two values.

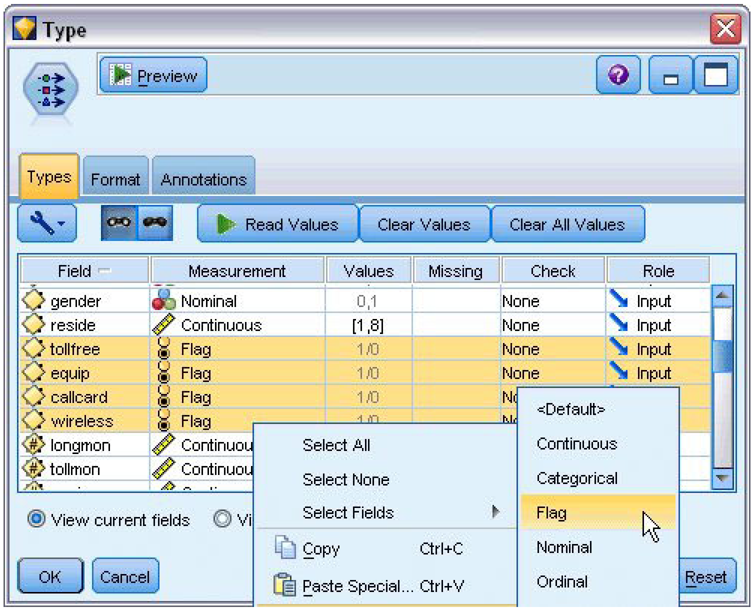


Figure 155. Setting the measurement level for multiple fields

Tip: To change properties for multiple fields with similar values (such as 0/1), click the *Values* column header to sort fields by value, and then hold down the Shift key while using the mouse or arrow keys to select all of the fields that you want to change. You can then right-click on the selection to change the measurement level or other attributes of the selected fields.

3. Set the measurement level for the *churn* field to **Flag**, and set the role to **Target**. All other fields should have their role set to **Input**.

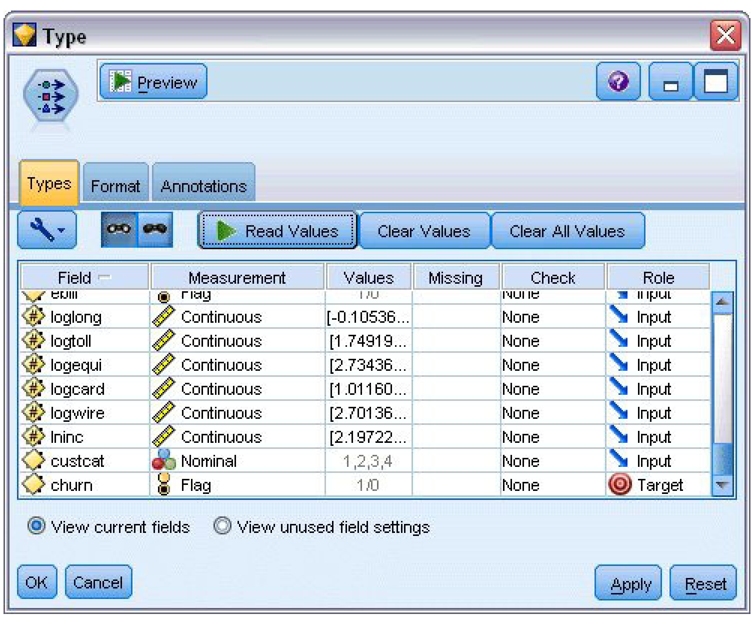
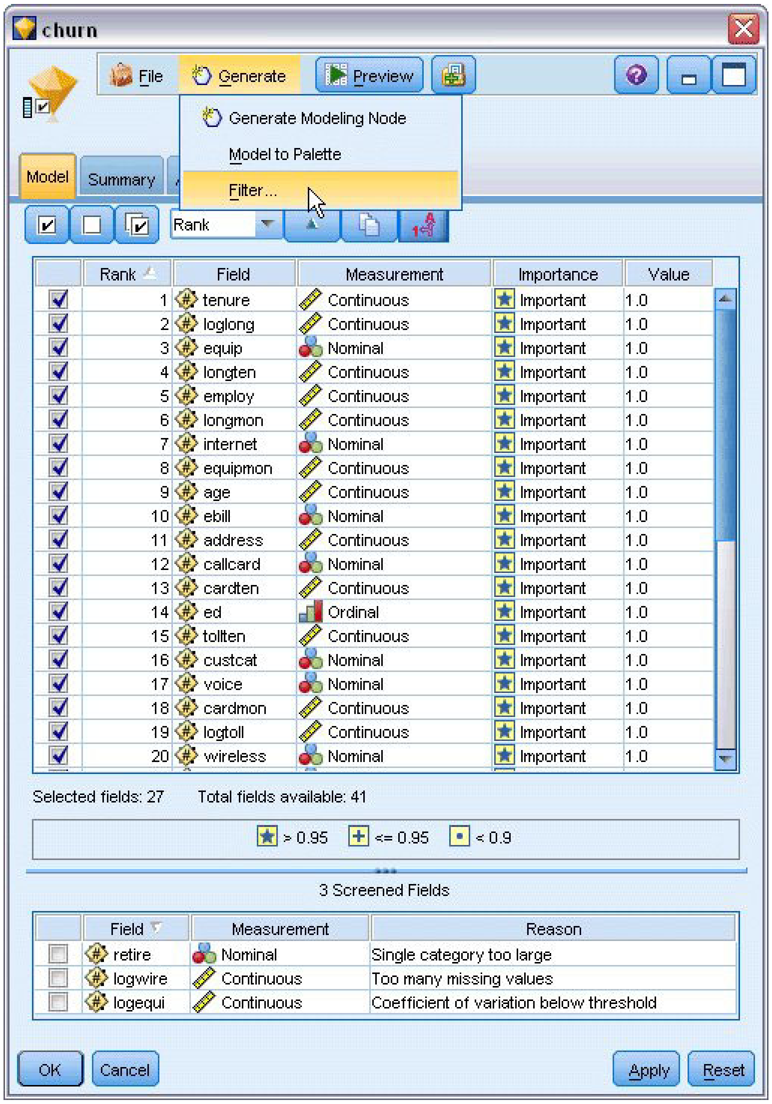


Figure 156. Setting the measurement level and role for the churn field

4. Add a Feature Selection modeling node to the Type node.

   Using a Feature Selection node enables you to remove predictors or data that do not add any useful information with respect to the predictor/target relationship.
5. Run the stream.

6. Open the resulting model nugget, and from the **Generate** menu, choose **Filter** to create a Filter node.



Rank	Field	Measurement	Importance	Value
1	tenure	Continuous	Important	1.0
2	loglong	Continuous	Important	1.0
3	equip	Nominal	Important	1.0
4	longten	Continuous	Important	1.0
5	employ	Continuous	Important	1.0
6	longmon	Continuous	Important	1.0
7	internet	Nominal	Important	1.0
8	equipmon	Continuous	Important	1.0
9	age	Continuous	Important	1.0
10	ebill	Nominal	Important	1.0
11	address	Continuous	Important	1.0
12	callcard	Nominal	Important	1.0
13	cardten	Continuous	Important	1.0
14	ed	Ordinal	Important	1.0
15	tollten	Continuous	Important	1.0
16	custcat	Nominal	Important	1.0
17	voice	Nominal	Important	1.0
18	cardmon	Continuous	Important	1.0
19	logtoll	Continuous	Important	1.0
20	wireless	Nominal	Important	1.0

Field	Measurement	Reason
retire	Nominal	Single category too large
logwire	Continuous	Too many missing values
logequi	Continuous	Coefficient of variation below threshold

Figure 157. Generating a Filter node from a Feature Selection node

Not all of the data in the *telco.sav* file will be useful in predicting churn. You can use the filter to only select data considered to be important for use as a predictor.

7. In the Generate Filter dialog box, select **All fields marked: Important** and click **OK**.
8. Attach the generated Filter node to the Type node.

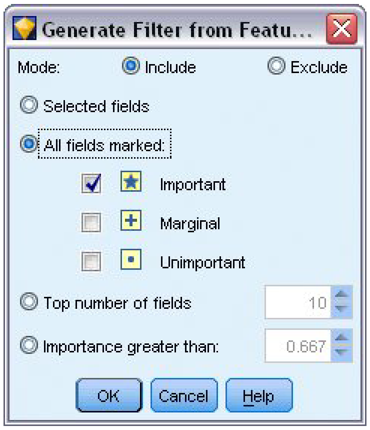


Figure 158. Selecting important fields

9. Attach a Data Audit node to the generated Filter node.

   Open the Data Audit node and click **Run**.

10. On the Quality tab of the Data Audit browser, click the *% Complete* column to sort the column by ascending numerical order. This lets you identify any fields with large amounts of missing data; in this case the only field you need to amend is *logtoll*, which is less than 50% complete.
11. In the *Impute Missing* column for *logtoll*, click **Specify**.

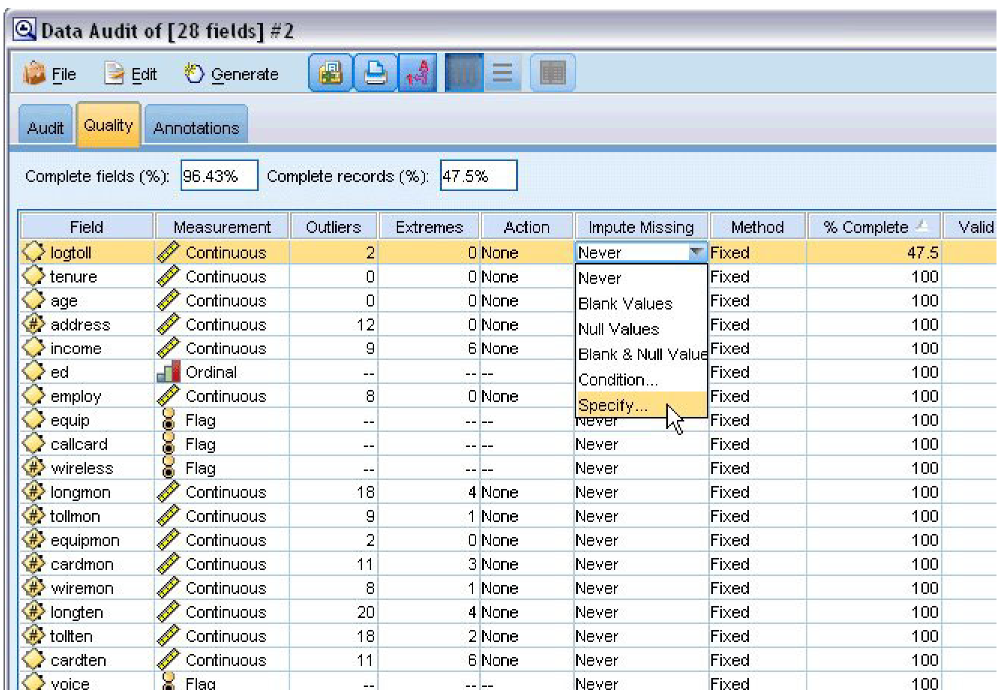


Figure 159. Imputing missing values for logtoll

12. For **Impute when**, select **Blank and Null values**. For **Fixed As**, select **Mean** and click **OK**.

    Selecting **Mean** ensures that the imputed values do not adversely affect the mean of all values in the overall data.

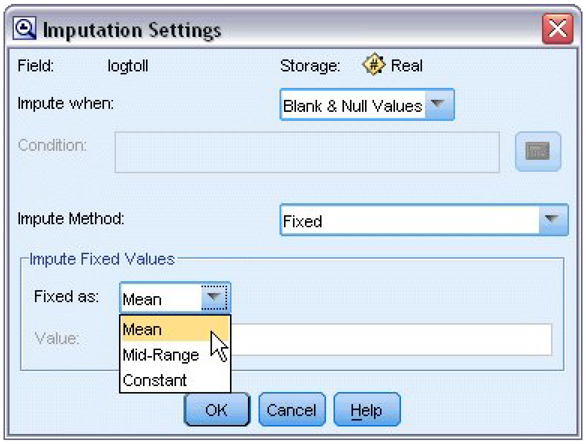


Figure 160. Selecting imputation settings

13. On the Data Audit browser Quality tab, generate the Missing Values SuperNode. To do this, from the menus choose:

    Generate > **Missing Values SuperNode**

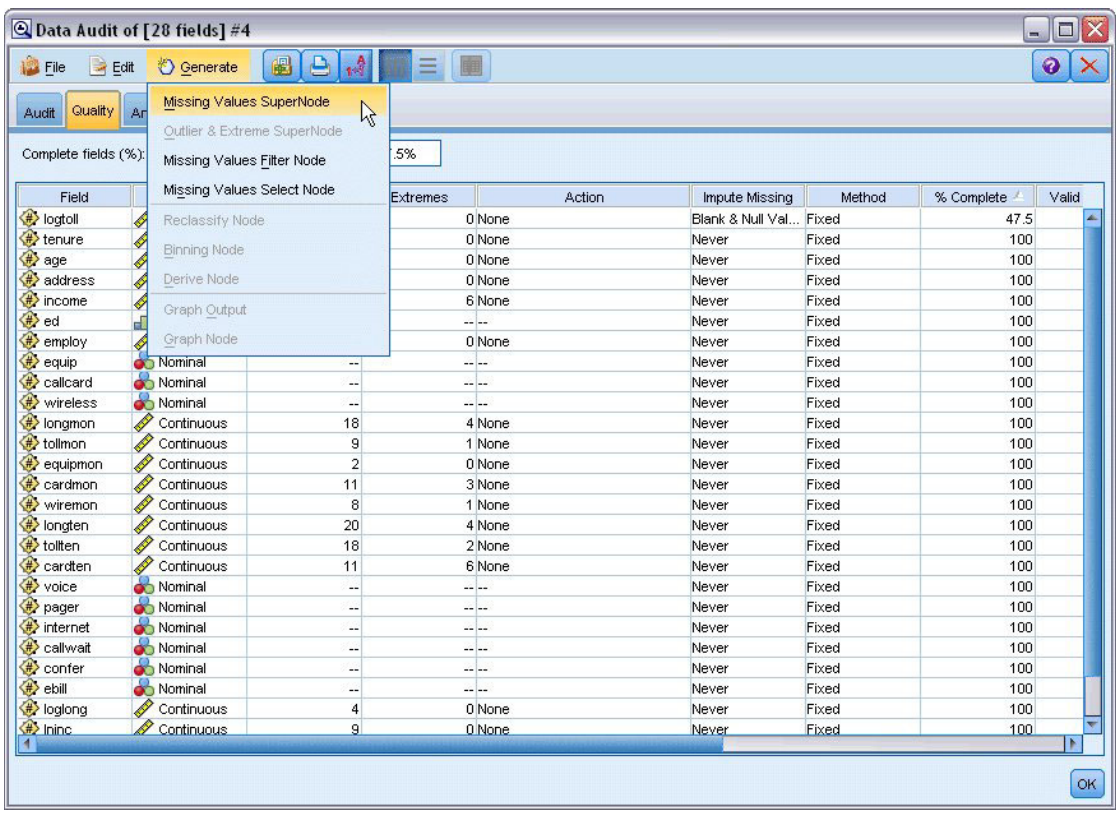


Figure 161. Generating a missing values SuperNode

In the Missing Values SuperNode dialog box, increase the **Sample Size** to 50% and click **OK**.

The SuperNode is displayed on the stream canvas, with the title: *Missing Value Imputation*.

14. Attach the SuperNode to the Filter node.

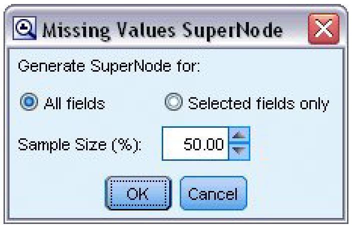


Figure 162. Specifying sample size

15. Add a Logistic node to the SuperNode.
16. In the Logistic node, click the Model tab and select the **Binomial** procedure. In the *Binomial Procedure* area, select the **Forwards** method.

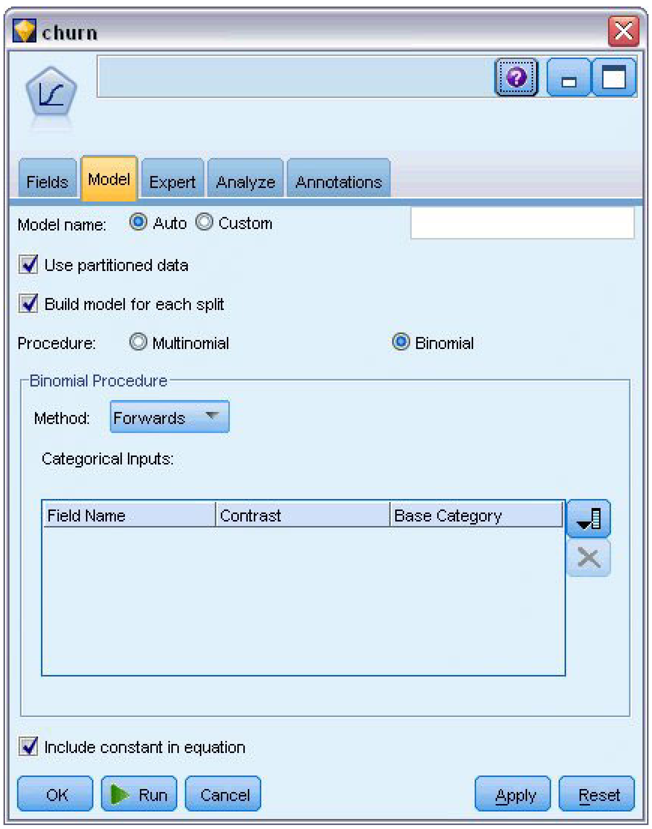


Figure 163. Choosing model options

17. On the Expert tab, select the **Expert** mode and then click **Output**. The Advanced Output dialog box is displayed.
18. In the Advanced Output dialog, select **At each step** as the *Display* type. Select **Iteration history** and **Parameter estimates** and click **OK**.

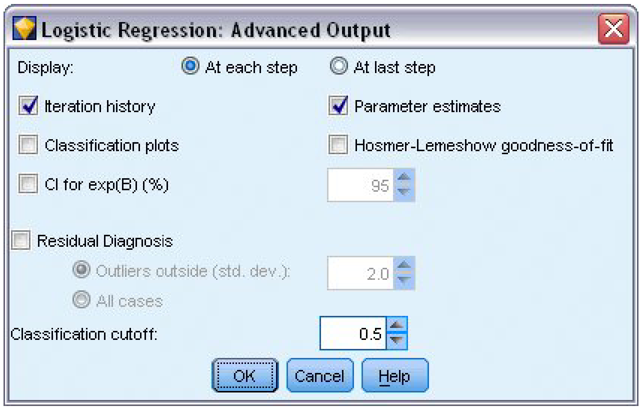


Figure 164. Choosing output options

Browsing the Model

1. On the Logistic node, click **Run** to create the model.

   The model nugget is added to the stream canvas, and also to the Models palette in the upper-right corner. To view its details, right-click on the model nugget and select **Edit** or **Browse**.

   The Summary tab shows (among other things) the target and inputs (predictor fields) used by the model. Note that these are the fields that were actually chosen based on the Forwards method, not the complete list submitted for consideration.

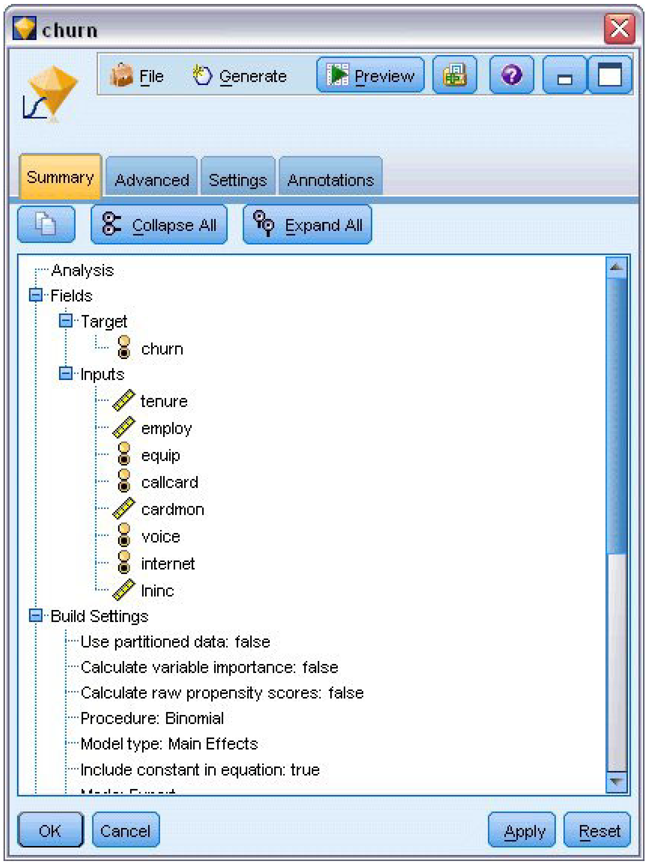


Figure 165. Model summary showing target and input fields

The items shown on the Advanced tab depend on the options selected on the Advanced Output dialog box in the Logistic node. One item that is always shown is the Case Processing Summary, which shows the number and percentage of records included in the analysis. In addition, it lists the number of missing cases (if any) where one or more of the input fields are unavailable and any cases that were not selected.

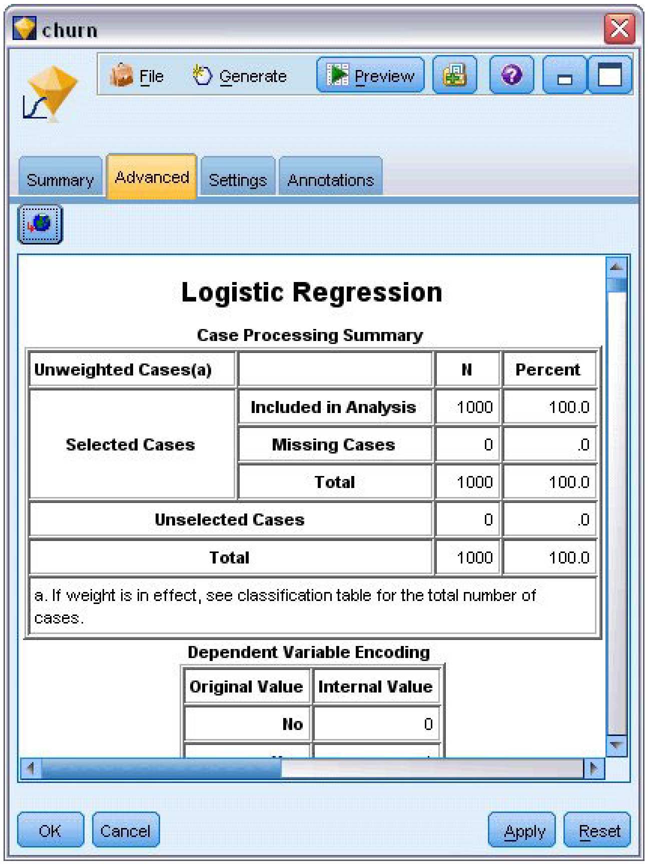


Figure 166. Case processing summary

2. Scroll down from the Case Processing Summary to display the Classification Table under Block 0: Beginning Block.

   The Forward Stepwise method starts with a null model - that is, a model with no predictors - that can be used as a basis for comparison with the final built model. The null model, by convention, predicts everything as a 0, so the null model is 72.6% accurate simply because the 726 customers who didn't churn are predicted correctly. However, the customers who did churn aren't predicted correctly at all.

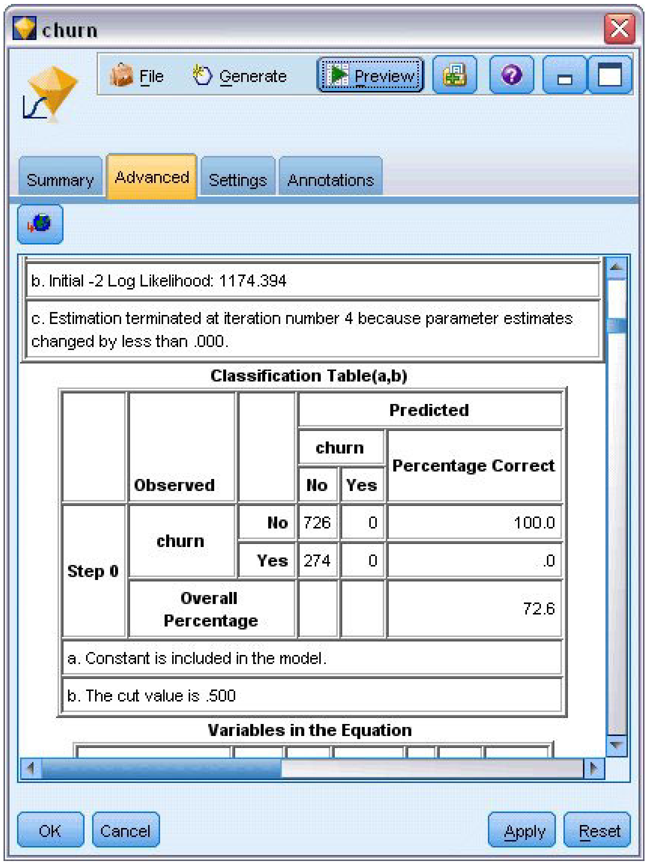


Figure 167. Starting classification table- Block 0

3. Now scroll down to display the Classification Table under Block 1: Method = Forward Stepwise.

   This Classification Table shows the results for your model as a predictor is added in at each of the steps. Already, in the first step - after just one predictor has been used - the model has increased the accuracy of the churn prediction from 0.0% to 29.9%

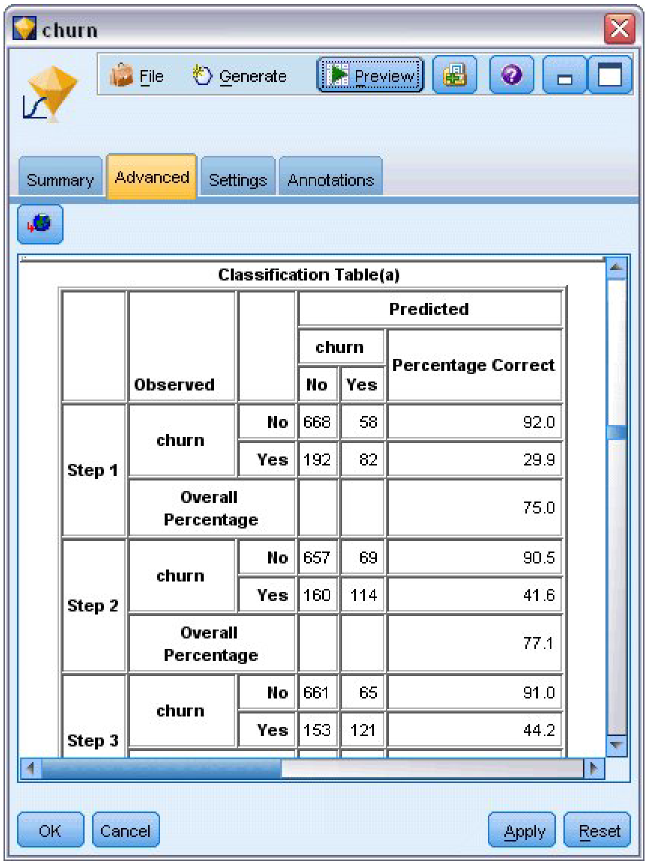


Figure 168. Classification table - Block 1

4. Scroll down to the bottom of this Classification Table.

The Classification Table shows that the last step is step 8. At this stage the algorithm has decided that it no longer needs to add any further predictors into the model. Although the accuracy of the non-churning customers has decreased a little to 91.2%, the accuracy of the prediction for those who did churn has risen from the original 0% to 47.1%. This is a significant improvement over the original null model that used no predictors.

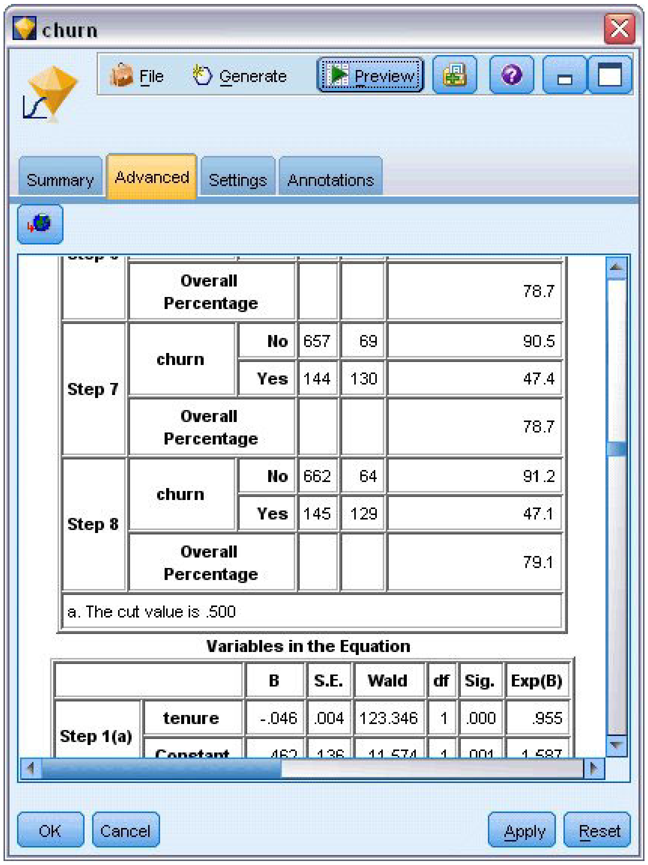


Figure 169. Classification table - Block 1

For a customer who wants to reduce churn, being able to reduce it by nearly half would be a major step in protecting their income streams.

Note: This example also shows how taking the Overall Percentage as a guide to a model's accuracy may, in some cases, be misleading. The original null model was 72.6% accurate overall, whereas the final predicted model has an overall accuracy of 79.1%; however, as we have seen, the accuracy of the actual individual category predictions were vastly different.

To assess how well the model actually fits the data, a number of diagnostics are available in the Advanced Output dialog box when you are building the model. Explanations of the mathematical foundations of the modeling methods used in IBM SPSS Modeler are listed in the *IBM SPSS Modeler Algorithms Guide*, available from the *Documentation* directory of the installation disk.

Note also that these results are based on the training data only. To assess how well the model generalizes to other data in the real world, you would use a Partition node to hold out a subset of records for purposes of testing and validation.

Chapter 14. Forecasting Bandwidth Utilization (Time Series)

Forecasting with the Time Series Node

An analyst for a national broadband provider is required to produce forecasts of user subscriptions in order to predict utilization of bandwidth. Forecasts are needed for each of the local markets that make up the national subscriber base. You will use time series modeling to produce forecasts for the next three months for a number of local markets. A second example shows how you can convert source data if it is not in the correct format for input to the Time Series node.

These examples use the stream named *broadband_create_models.str*, which references the data file named *broadband_1.sav*. These files are available from the *Demos* folder of any IBM SPSS Modeler installation. This can be accessed from the IBM SPSS Modeler program group on the Windows Start menu. The *broadband_create_models.str* file is in the *streams* folder.

The last example demonstrates how to apply the saved models to an updated dataset in order to extend the forecasts by another three months.

In IBM SPSS Modeler, you can produce multiple time series models in a single operation. The source file you'll be using has time series data for 85 different markets, although for the sake of simplicity you will only model five of these markets, plus the total for all markets.

The *broadband_1.sav* data file has monthly usage data for each of 85 local markets. For the purposes of this example, only the first five series will be used; a separate model will be created for each of these five series, plus a total.

The file also includes a date field that indicates the month and year for each record. This field will be used in a Time Intervals node to label records. The date field reads into IBM SPSS Modeler as a string, but in order to use the field in IBM SPSS Modeler you will convert the storage type to numeric Date format using a Filler node.

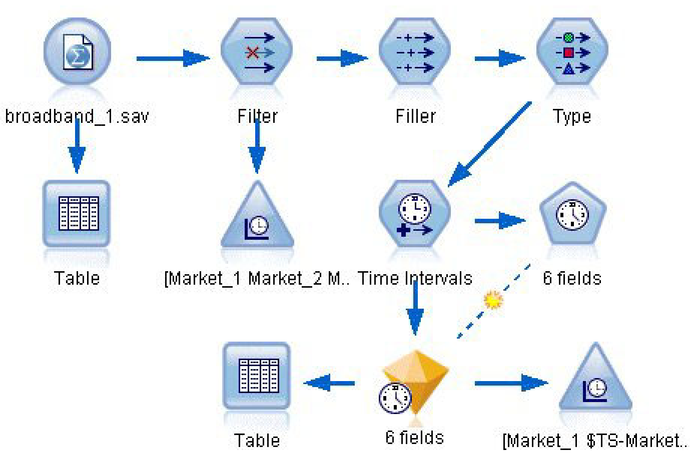
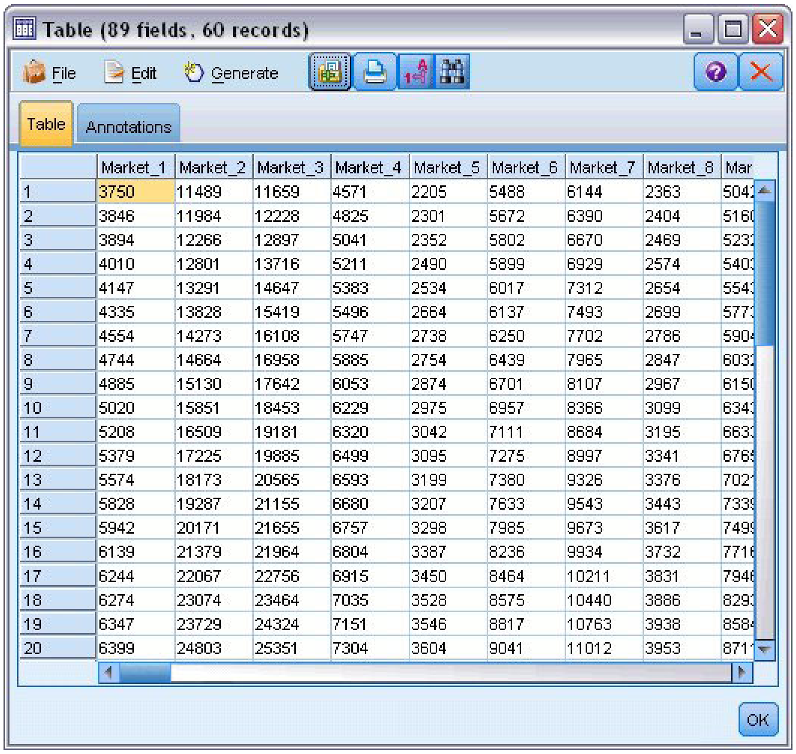


Figure 170. Sample stream to show Time Series modeling

The Time Series node requires that each series be in a separate column, with a row for each interval. IBM SPSS Modeler provides methods for transforming data to match this format if necessary.



	Market_1	Market_2	Market_3	Market_4	Market_5	Market_6	Market_7	Market_8	Mar
1	3750	11489	11659	4571	2205	5488	6144	2363	504
2	3846	11984	12228	4825	2301	5672	6390	2404	516
3	3894	12266	12897	5041	2352	5802	6670	2469	523
4	4010	12801	13716	5211	2490	5899	6929	2574	540
5	4147	13291	14647	5383	2534	6017	7312	2654	554
6	4335	13828	15419	5496	2664	6137	7493	2699	577
7	4554	14273	16108	5747	2738	6250	7702	2786	590
8	4744	14664	16958	5885	2754	6439	7965	2847	603
9	4885	15130	17642	6053	2874	6701	8107	2967	615
10	5020	15851	18453	6229	2975	6957	8366	3099	634
11	5208	16509	19181	6320	3042	7111	8684	3195	663
12	5379	17225	19885	6499	3095	7275	8997	3341	676
13	5574	18173	20565	6593	3199	7380	9326	3376	702
14	5828	19287	21155	6680	3207	7633	9543	3443	733
15	5942	20171	21655	6757	3298	7985	9673	3617	749
16	6139	21379	21964	6804	3387	8236	9934	3732	771
17	6244	22067	22756	6915	3450	8464	10211	3831	794
18	6274	23074	23464	7035	3528	8575	10440	3886	829
19	6347	23729	24324	7151	3546	8817	10763	3938	858
20	6399	24803	25351	7304	3604	9041	11012	3953	871

Figure 171. Monthly subscription data for broadband local markets

Creating the Stream

1. Create a new stream and add a Statistics File source node pointing to *broadband_1.sav*.
2. Use a Filter node to filter out the *Market_6* to *Market_85* fields and the *MONTH_* and *YEAR_* fields to simplify the model.

Tip: To select multiple adjacent fields in a single operation, click the *Market_6* field, hold down the left mouse button and drag the mouse down to the *Market_85* field. Selected fields are highlighted in blue. To add the other fields, hold down the Ctrl key and click the *MONTH_* and *YEAR_* fields.

Figure 172. Simplifying the model

Examining the Data

It is always a good idea to have a feel for the nature of your data before building a model. Do the data exhibit seasonal variations? Although the Expert Modeler can automatically find the best seasonal or nonseasonal model for each series, you can often obtain faster results by limiting the search to nonseasonal models when seasonality is not present in your data. Without examining the data for each of the local markets, we can get a rough picture of the presence or absence of seasonality by plotting the total number of subscribers over all five markets.

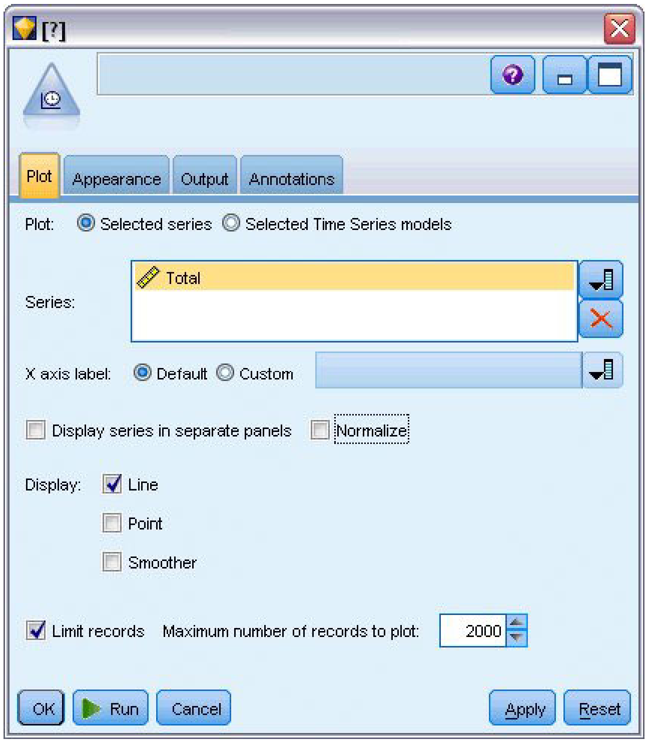


Figure 173. Plotting the total number of subscribers

1. From the Graphs palette, attach a Time Plot node to the Filter node.
2. Add the *Total* field to the Series list.
3. Deselect the **Display series in separate panels** and **Normalize** check boxes.
4. Click **Run**.

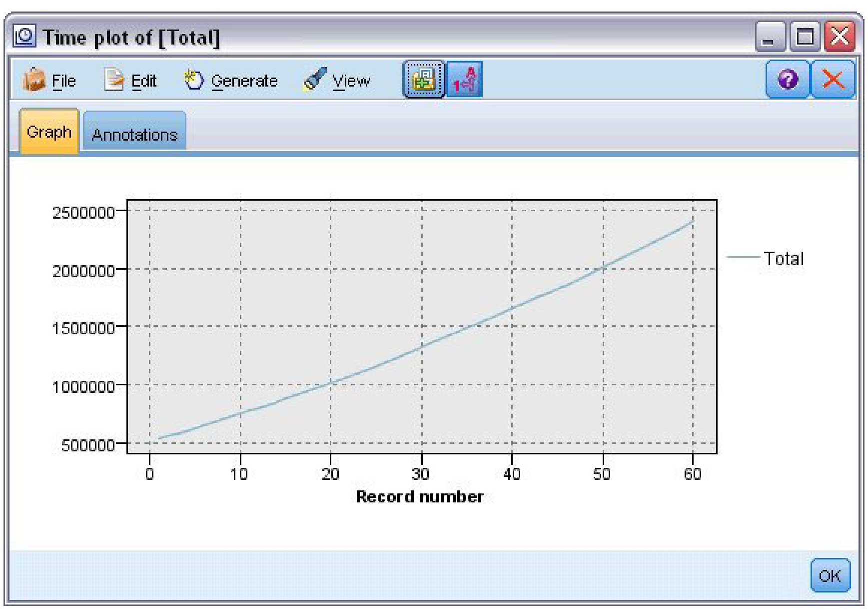


Figure 174. Time plot of Total field

The series exhibits a very smooth upward trend with no hint of seasonal variations. There might be individual series with seasonality, but it appears that seasonality is not a prominent feature of the data in general.

Of course you should inspect each of the series before ruling out seasonal models. You can then separate out series exhibiting seasonality and model them separately.

IBM SPSS Modeler makes it easy to plot multiple series together.

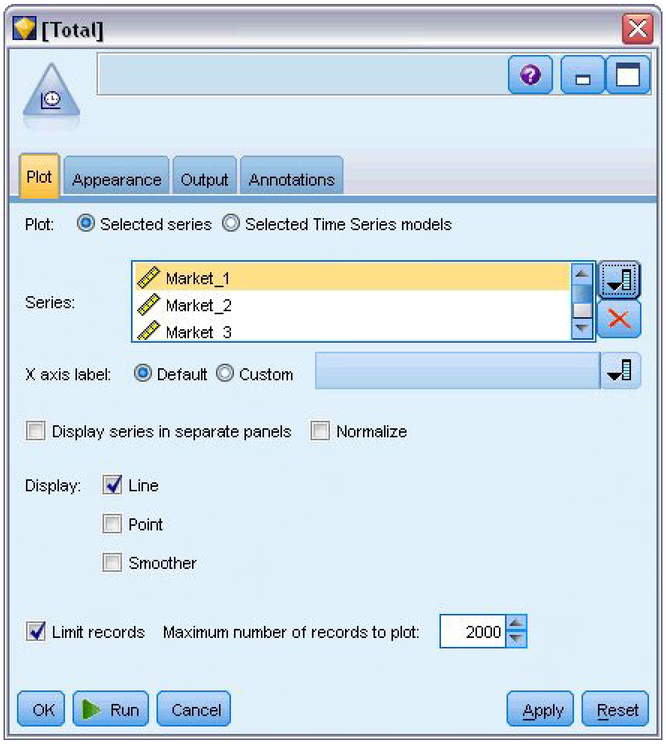


Figure 175. Plotting multiple time series

5. Reopen the Time Plot node.
6. Remove the *Total* field from the Series list (select it and click the red X button).
7. Add the *Market_1* through *Market_5* fields to the list.
8. Click **Run**.

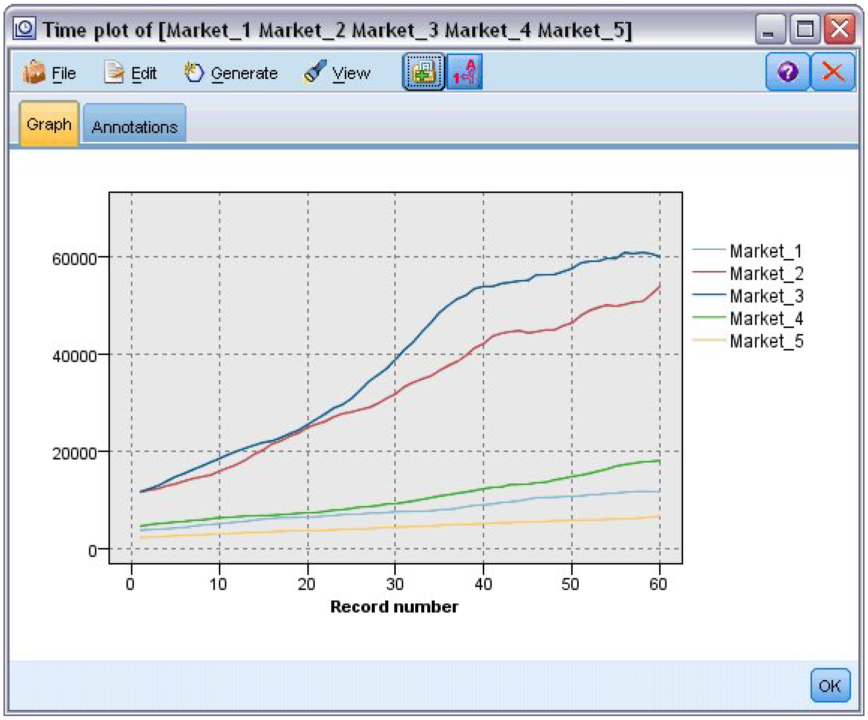


Figure 176. Time plot of multiple fields

Inspection of each of the markets reveals a steady upward trend in each case. Although some markets are a little more erratic than others, there is no evidence of seasonality to be seen.

Defining the Dates

Now you need to change the storage type of the *DATE_* field to Date format.

1. Attach a Filler node to the Filter node.
2. Open the Filler node and click the field selector button.
3. Select **DATE_** to add it to **Fill in fields**.
4. Set the **Replace** condition to **Always**.
5. Set the value of **Replace with** to **to_date(DATE_)**.

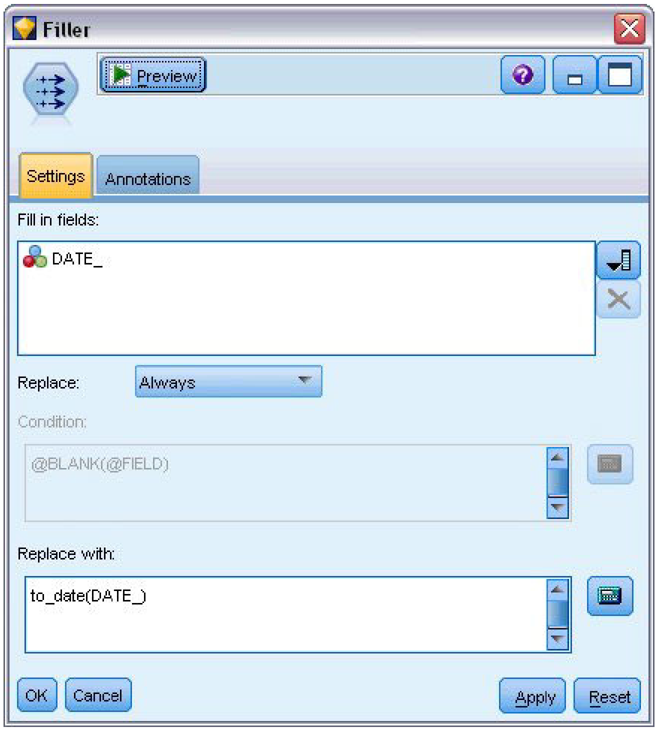


Figure 177. Setting the date storage type

Change the default date format to match the format of the Date field. This is necessary for the conversion of the Date field to work as expected.

6. On the menu, choose **Tools** > **Stream Properties** > **Options** to display the Stream Options dialog box.
7. Set the default **Date format** to **MON YYYY** .

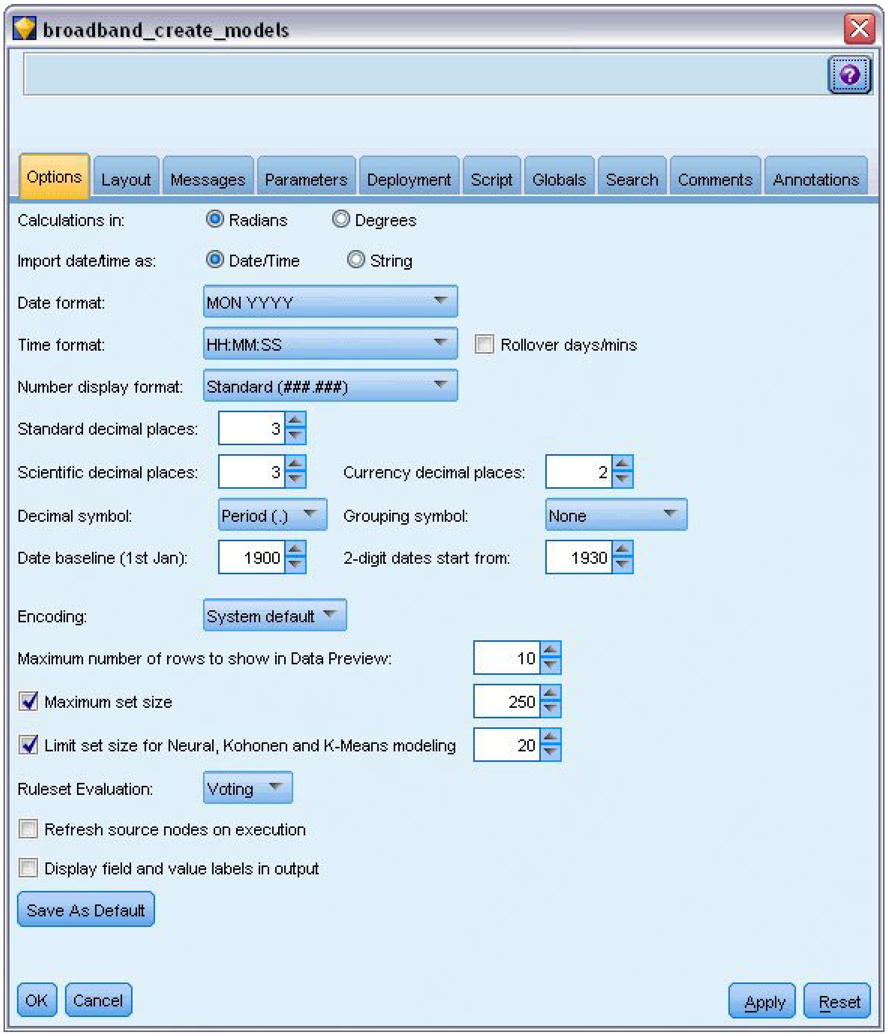


Figure 178. Setting the date format

Defining the Targets

1. Add a Type node and set the role to **None** for the *DATE_* field. Set the role to **Target** for all others (the *Market_n* fields plus the *Total* field).
2. Click the **Read Values** button to populate the Values column.

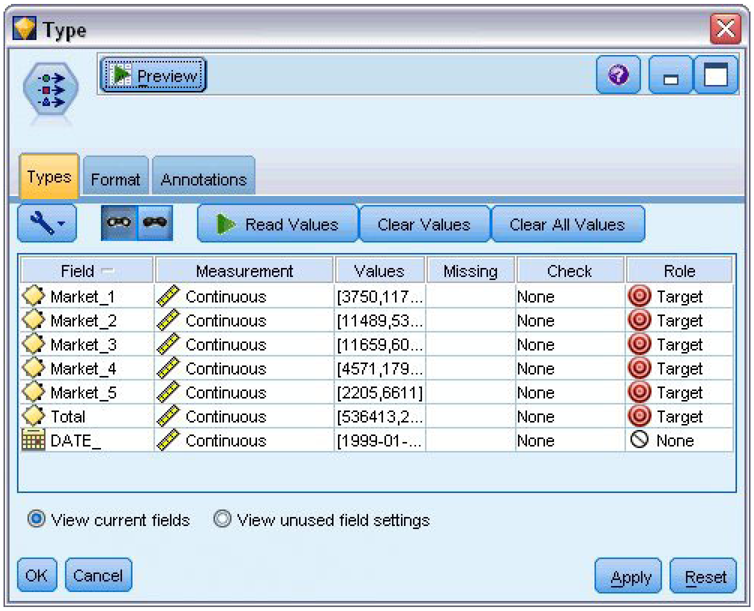


Figure 179. Setting the role for multiple fields

Setting the Time Intervals

1. Add a Time Intervals node (from the Field Operations palette).
2. On the Intervals tab, select **Months** as the time interval.
3. Select the **Build from data** option.
4. Select **DATE_** as the build field.

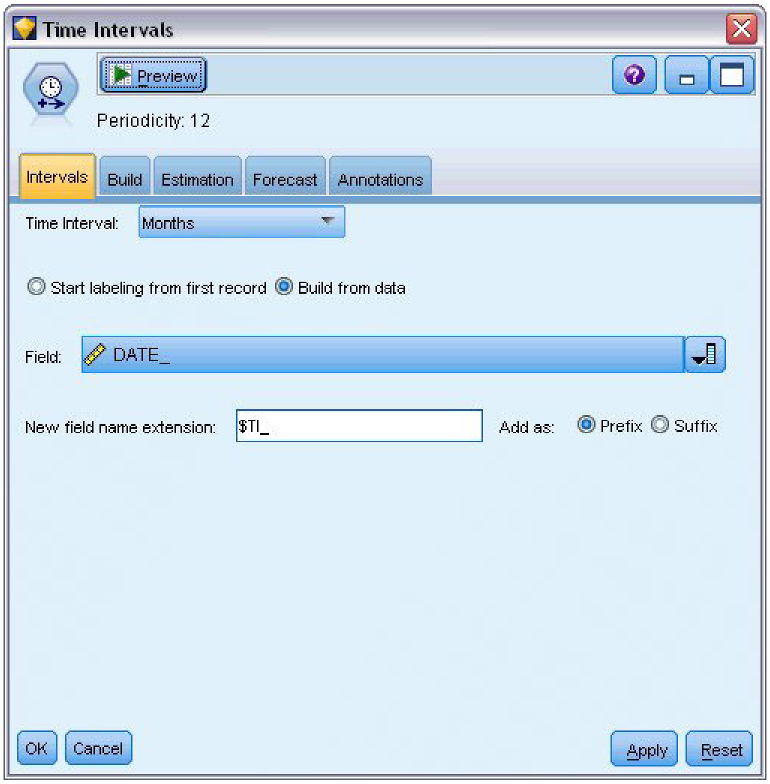


Figure 180. Setting the time interval

5. On the Forecast tab, select the **Extend records into the future** check box.
6. Set the value to **3**.
7. Click **OK**.

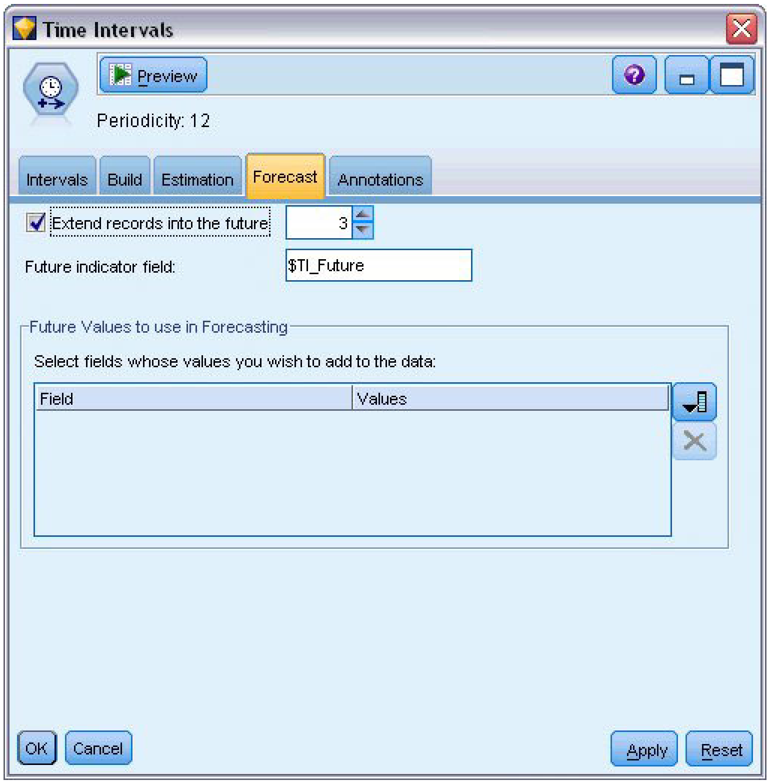


Figure 181. Setting the forecast period

Creating the Model

1. From the Modeling palette, add a Time Series node to the stream and attach it to the Time Intervals node.
2. Click **Run** on the Time Series node using all default settings. Doing so enables the Expert Modeler to decide the most appropriate model to use for each time series.

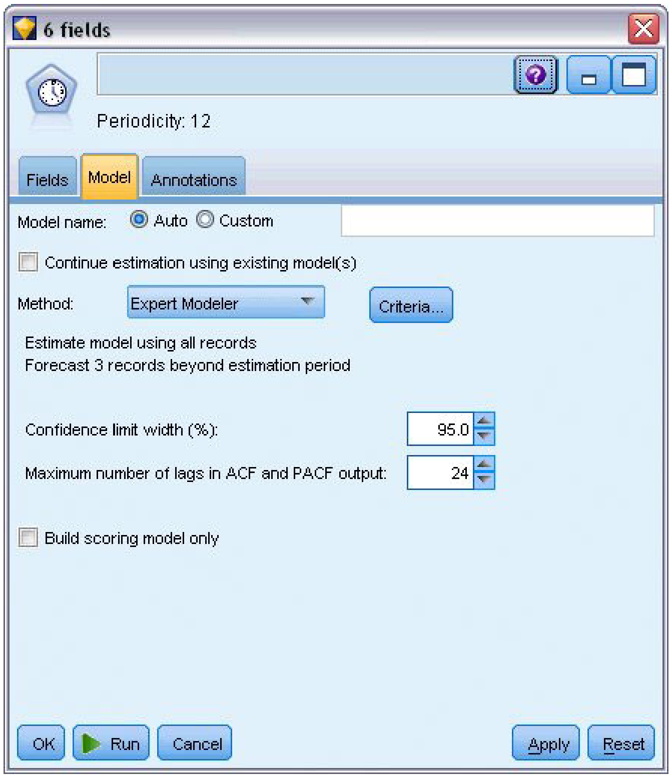


Figure 182. Choosing the Expert Modeler for Time Series

3. Attach the Time Series model nugget to the Time Intervals node.
4. Attach a Table node to the Time Series model and click **Run**.

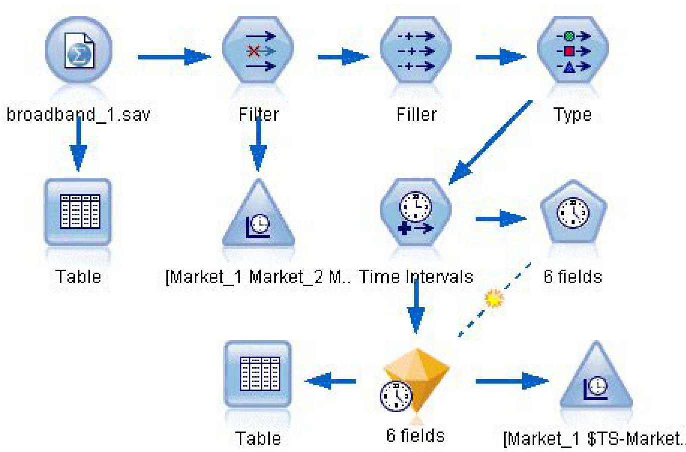


Figure 183. Sample stream to show Time Series modeling

There are now three new rows (61 through 63) appended to the original data. These are the rows for the forecast period, in this case January to March 2004.

Several new columns are also present now--a number of *$TI_* columns added by the Time Intervals node and the *$TS-* columns added by the Time Series node. The columns indicate the following for each row (i.e., each interval in the time series data):

Column	Description
$TI_TimeIndex	The time interval index value for this row.
$TI_TimeLabel	The time interval label for this row.
$TI_Year	The year and month indicators for the generated data in this row.
$TI_Month	
$TI_Count	The number of records involved in determining the new data for this row.
$TI_Future	Indicates whether this row contains forecast data.
$TS-*colname*	The generated model data for each column of the original data.
$TSLCI-*colname*	The lower confidence interval value for each column of the generated model data.
$TSUCI-*colname*	The upper confidence interval value for each column of the generated model data.
$TS-Total	The total of the $TS-*colname* values for this row.
$TSLCI-Total	The total of the $TSLCI-*colname* values for this row.
$TSUCI-Total	The total of the $TSUCI-*colname* values for this row.

The most significant columns for the forecast operation are the *$TS-Market_n*, *$TSLCI-Market_n*, and *$TSUCI-Market_n* columns. In particular, these columns in rows 61 through 63 contain the user subscription forecast data and confidence intervals for each of the local markets.

Examining the Model

1. Double-click the Time Series model nugget to display data about the models generated for each of the markets.

   Note how the Expert Modeler has chosen to generate a different type of model for Market 5 from the type it has generated for the other markets.

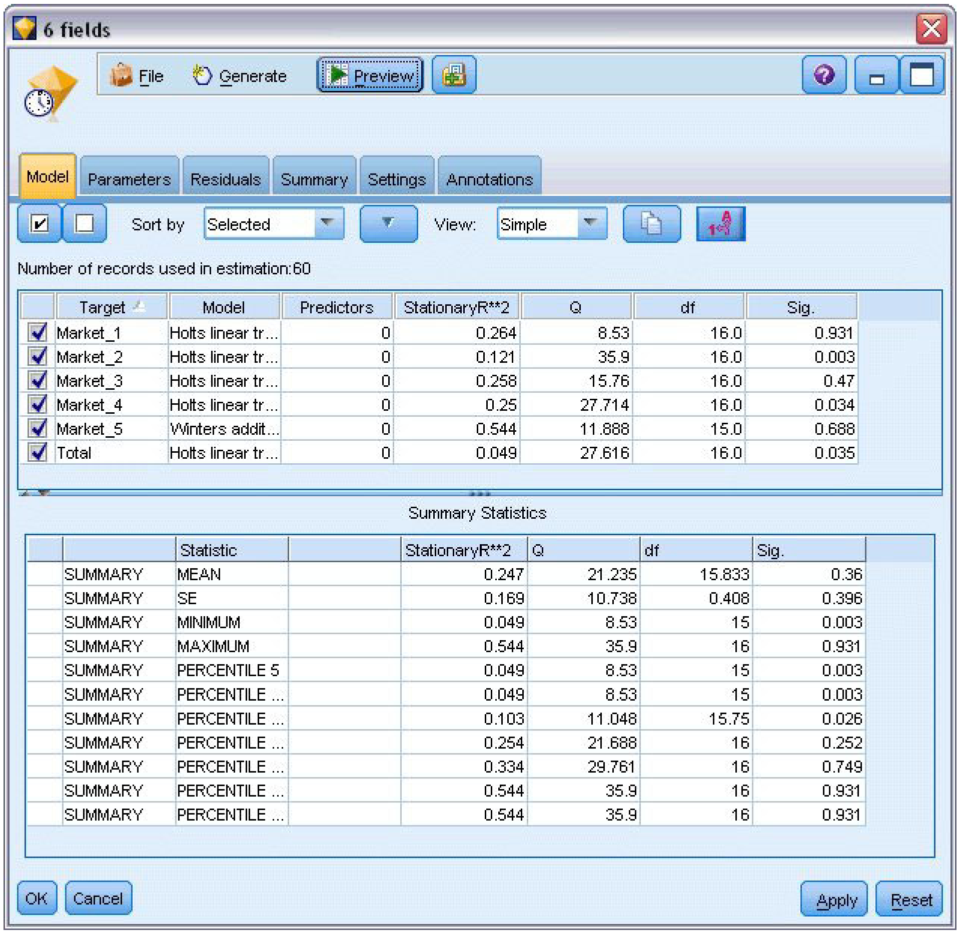


Figure 184. Time Series models generated for the markets

The Predictors column shows how many fields were used as predictors for each target—in this case, none.

The remaining columns in this view show various goodness-of-fit measures for each model. The **StationaryR**2** column shows the Stationary *R*-squared value. This statistic provides an estimate of the proportion of the total variation in the series that is explained by the model. The higher the value (to a maximum of 1.0), the better the fit of the model.

The **Q**, **df**, and **Sig.** columns relate to the Ljung-Box statistic, a test of the randomness of the residual errors in the model--the more random the errors, the better the model is likely to be. **Q** is the Ljung-Box statistic itself, while **df** (degrees of freedom) indicates the number of model parameters that are free to vary when estimating a particular target.

The **Sig.** column gives the significance value of the Ljung-Box statistic, providing another indication of whether the model is correctly specified. A significance value less than 0.05 indicates that the residual errors are not random, implying that there is structure in the observed series that is not accounted for by the model.

Taking both the Stationary *R*-squared and Significance values into account, the models that the Expert Modeler has chosen for *Market_1*, *Market_3*, and *Market_5* are quite acceptable. The **Sig.** values for *Market_2* and *Market_4* are both less than 0.05, indicating that some experimentation with better-fitting models for these markets might be necessary.

The summary values in the lower part of the display provide information on the distribution of the statistics across all models. For example, the mean Stationary *R*-squared value across all the models is 0.247, while the minimum such value is 0.049 (that of the *Total* model) and the maximum is 0.544 (the value for *Market_5*).

SE denotes the standard error across all the models for each statistic. For example, the standard error for Stationary *R*-squared across all models is 0.169.

The summary section also includes percentile values that provide information on the distribution of the statistics across models. For each percentile, that percentage of models have a value of the fit statistic below the stated value.

Thus for example, only 25% of the models have a Stationary *R*-squared value that is less than 0.121.

2. Click the View drop-down list and select **Advanced**.

   The display shows a number of additional goodness-of-fit measures. **R**2** is the *R*-squared value, an estimation of the total variation in the time series that can be explained by the model. As the maximum value for this statistic is 1.0, our models are fine in this respect.

	MAPE	MAE	MaxAPE	MaxAE	Norm. BIC	Q	df	Sig.
47	0.94	73.869	2.147	224.517	9.15	8.53	16.0	0.931
76	0.94	314.721	1.867	927.949	12.059	35.9	16.0	0.003
33	0.776	306.877	1.918	1,030.105	12.1	15.76	16.0	0.47
38	0.78	79.49	1.942	233.544	9.329	27.714	16.0	0.034
32	0.936	39.963	2.481	137.633	8.114	11.888	15.0	0.688
74	0.094	1,326.071	0.299	7,062.662	15.243	27.616	16.0	0.035

Summary Statistics

MAPE	MAE	MaxAPE	MaxAE	Norm. BIC	Q	df	Sig.
0.744	356.832	1.776	1,602.735	10.999	21.235	15.833	0.36
0.328	490.119	0.758	2,702.397	2.641	10.738	0.408	0.396
0.094	39.963	0.299	137.633	8.114	8.53	15	0.003
0.94	1,326.071	2.481	7,062.662	15.243	35.9	16	0.931
0.094	39.963	0.299	137.633	8.114	8.53	15	0.003
0.094	39.963	0.299	137.633	8.114	8.53	15	0.003
0.605	65.393	1.475	202.796	8.891	11.048	15.75	0.026
0.858	193.183	1.93	580.747	10.694	21.688	16	0.252
0.94	567.559	2.231	2,538.245	12.886	29.761	16	0.749
0.94	1,326.071	2.481	7,062.662	15.243	35.9	16	0.931
0.94	1,326.071	2.481	7,062.662	15.243	35.9	16	0.931

Figure 185. Time Series models advanced display

RMSE is the root mean square error, a measure of how much the actual values of a series differ from the values predicted by the model, and is expressed in the same units as those used for the series itself. As this is a measurement of an error, we want this value to be as low as possible. At first sight it appears that the models for *Market_2* and *Market_3*, while still acceptable according to the statistics we have seen so far, are less successful than those for the other three markets.

These additional goodness-of-fit measure include the mean absolute percentage errors (**MAPE**) and its maximum value (**MaxAPE**). Absolute percentage error is a measure of how much a target series varies from its model-predicted level, expressed as a percentage value. By examining the mean and maximum across all models, you can get an indication of the uncertainty in your predictions.

The MAPE value shows that all models display a mean uncertainty of less than 1%, which is very low. The MaxAPE value displays the maximum absolute percentage error and is useful for imagining a worst-case scenario for your forecasts. It shows that the largest percentage error for each of the models falls in the range of roughly 1.8 to 2.5%, again a very low set of figures.

The **MAE** (mean absolute error) value shows the mean of the absolute values of the forecast errors. Like the RMSE value, this is expressed in the same units as those used for the series itself. **MaxAE** shows the largest forecast error in the same units and indicates worst-case scenario for the forecasts.

Interesting though these absolute values are, it is the values of the percentage errors (MAPE and MaxAPE) that are more useful in this case, as the target series represent subscriber numbers for markets of varying sizes.

Do the MAPE and MaxAPE values represent an acceptable amount of uncertainty with the models? They are certainly very low. This is a situation in which business sense comes into play, because acceptable risk will change from problem to problem. We'll assume that the goodness-of-fit statistics fall within acceptable bounds and go on to look at the residual errors.

Examining the values of the autocorrelation function (ACF) and partial autocorrelation function (PACF) for the model residuals provides more quantitative insight into the models than simply viewing goodness-of-fit statistics.

A well-specified time series model will capture all of the nonrandom variation, including seasonality, trend, and cyclic and other factors that are important. If this is the case, any error should not be correlated with itself (autocorrelated) over time. A significant structure in either of the autocorrelation functions would imply that the underlying model is incomplete.

3. Click the Residuals tab to display the values of the autocorrelation function (ACF) and partial autocorrelation function (PACF) for the residual errors in the model for the first of the local markets.

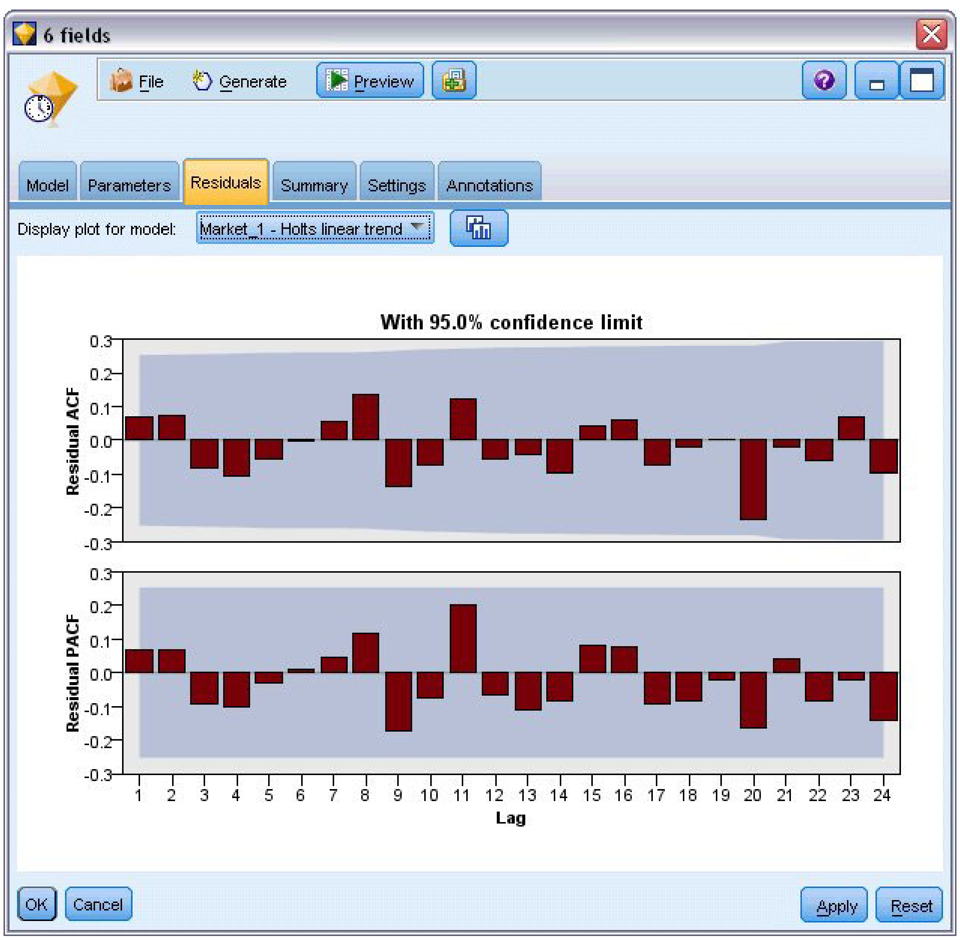


Figure 186. ACF and PACF values for the markets

In these plots, the original values of the error variable have been lagged by up to 24 time periods and compared with the original value to see if there is any correlation over time. For the model to be acceptable, none of the bars in the upper (ACF) plot should extend outside the shaded area, in either a positive (up) or negative (down) direction.

Should this occur, you would need to check the lower (PACF) plot to see whether the structure is confirmed there. The PACF plot looks at correlations after controlling for the series values at the intervening time points.

The values for *Market_1* are all within the shaded area, so we can continue and check the values for the other markets.

4. Click the **Display plot for model** drop-down list to display these values for the other markets and the totals.

   The values for *Market_2* and *Market_4* give a little cause for concern, confirming what we suspected earlier from their **Sig.** values. We'll need to experiment with some different models for those markets at some point to see if we can get a better fit, but for the rest of this example, we'll concentrate on what else we can learn from the *Market_1* model.
5. From the Graphs palette, attach a Time Plot node to the Time Series model nugget.
6. On the Plot tab, uncheck the **Display series in separate panels** check box.
7. At the **Series** list, click the field selector button, select the *Market_1* and *$TS-Market_1* fields, and click **OK** to add them to the list.
8. Click **Run** to display a line graph of the actual and forecast data for the first of the local markets.

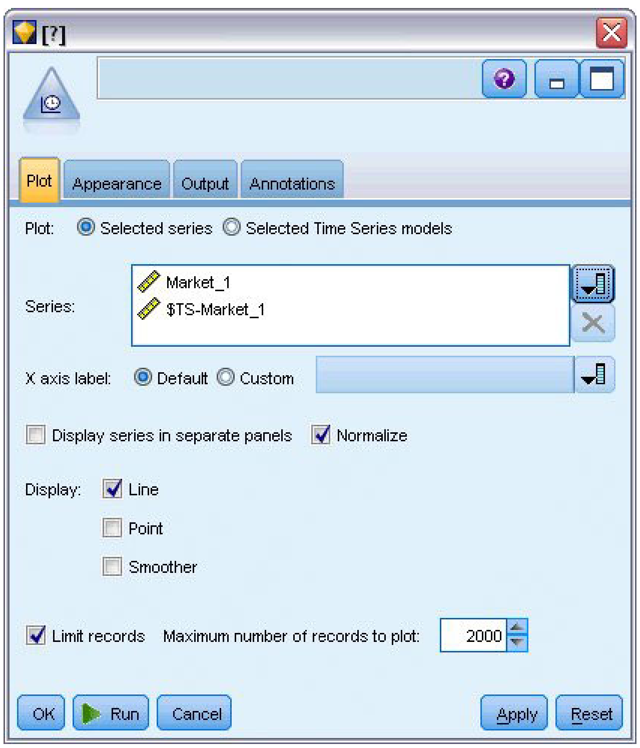


Figure 187. Selecting the fields to plot

Notice how the forecast (*$TS-Market_1*) line extends past the end of the actual data. You now have a forecast of expected demand for the next three months in this market.

The lines for actual and forecast data over the entire time series are very close together on the graph, indicating that this is a reliable model for this particular time series.

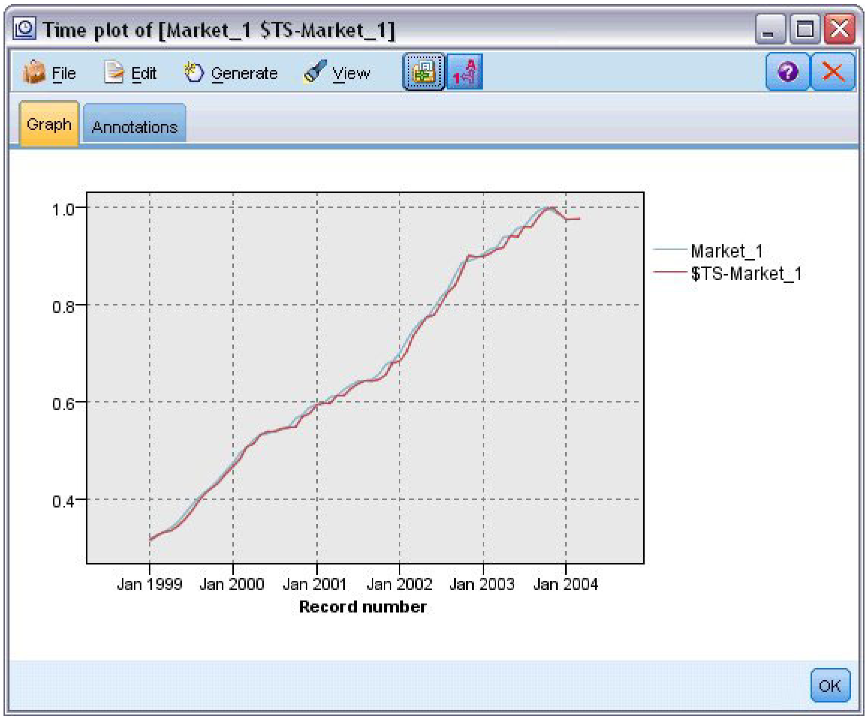


Figure 188. Time Plot of actual and forecast data for Market_1

Save the model in a file for use in a future example:

9. Click **OK** to close the current graph.
10. Open the Time Series model nugget.
11. Choose **File > Save Node** and specify the file location.
12. Click **Save**.

    You have a reliable model for this particular market, but what margin of error does the forecast have? You can get an indication of this by examining the confidence interval.
13. Double-click the last Time Plot node in the stream (the one labeled **Market_1 $TS-Market_1**) to open its dialog box again.
14. Click the field selector button and add the *$TSLCI-Market_1* and *$TSUCI-Market_1* fields to the **Series** list.
15. Click **Run**.

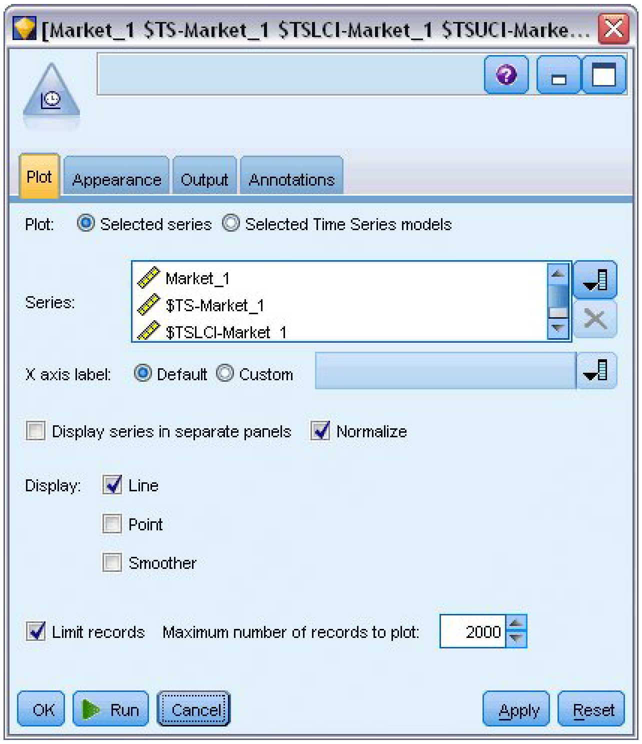


Figure 189. Adding more fields to plot

Now you have the same graph as before, but with the upper (*$TSUCI*) and lower (*$TSLCI*) limits of the confidence interval added.

Notice how the boundaries of the confidence interval diverge over the forecast period, indicating increasing uncertainty as you forecast further into the future.

However, as each time period goes by, you will have another (in this case) month's worth of actual usage data on which to base your forecast. You can read the new data into the stream and reapply your model now that you know it is reliable. See the topic "Reapplying a Time Series Model" on page 168 for more information.

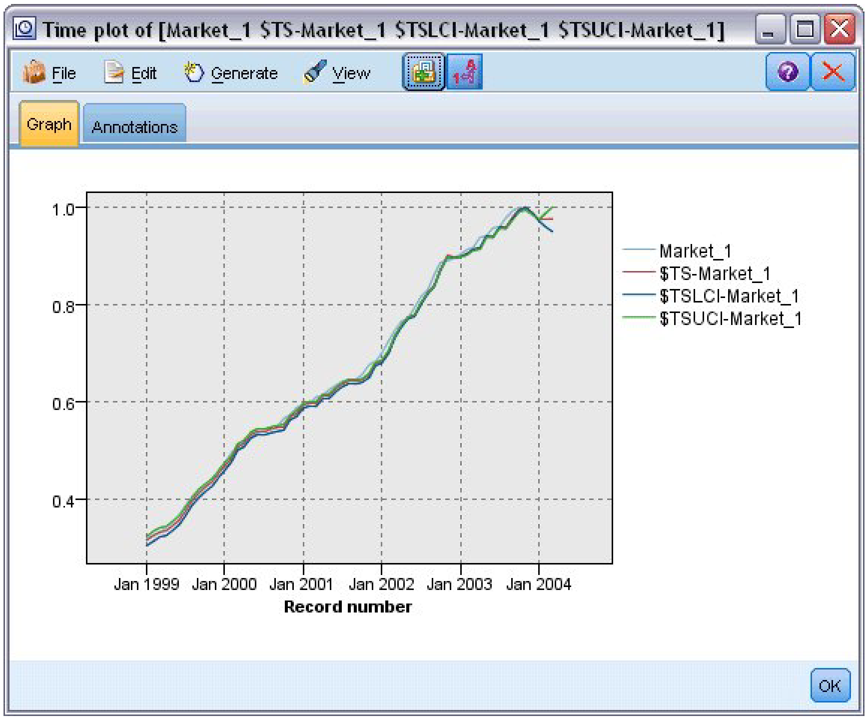


Figure 190. Time Plot with confidence interval added

Summary

You have learned how to use the Expert Modeler to produce forecasts for multiple time series, and you have saved the resulting models to an external file.

In the next example, you will see how to transform nonstandard time series data into a format suitable for input to a Time Series node.

Reapplying a Time Series Model

This example applies the time series models from the first time series example but can also be used independently. See the topic "Forecasting with the Time Series Node" on page 149 for more information.

As in the original scenario, an analyst for a national broadband provider is required to produce monthly forecasts of user subscriptions for each of a number of local markets, in order to predict bandwidth requirements. You have already used the Expert Modeler to create models and to forecast three months into the future.

Your data warehouse has now been updated with the actual data for the original forecast period, so you would like to use that data to extend the forecast horizon by another three months.

This example uses the stream named *broadband_apply_models.str*, which references the data file named *broadband_2.sav*. These files are available from the *Demos* folder of any IBM SPSS Modeler installation. This can be accessed from the IBM SPSS Modeler program group on the Windows Start menu. The *broadband_apply_models.str* file is in the *streams* folder.

Retrieving the Stream

In this example, you'll be recreating a Time Series node from the Time Series model saved in the first example. Don't worry if you don't have a model saved—we've provided one in the *Demos* folder.

1. Open the stream *broadband_apply_models.str* from the *streams* folder under *Demos*.

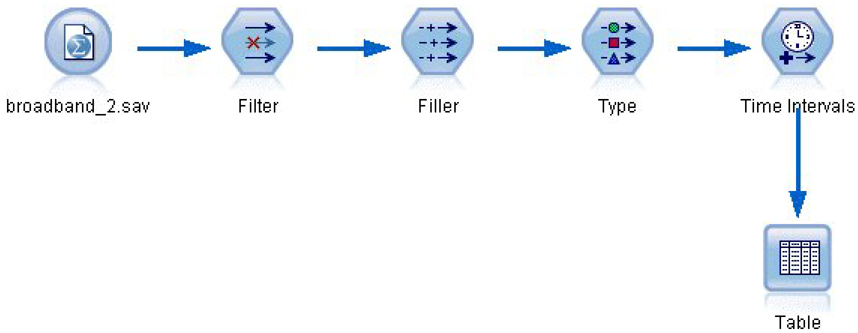
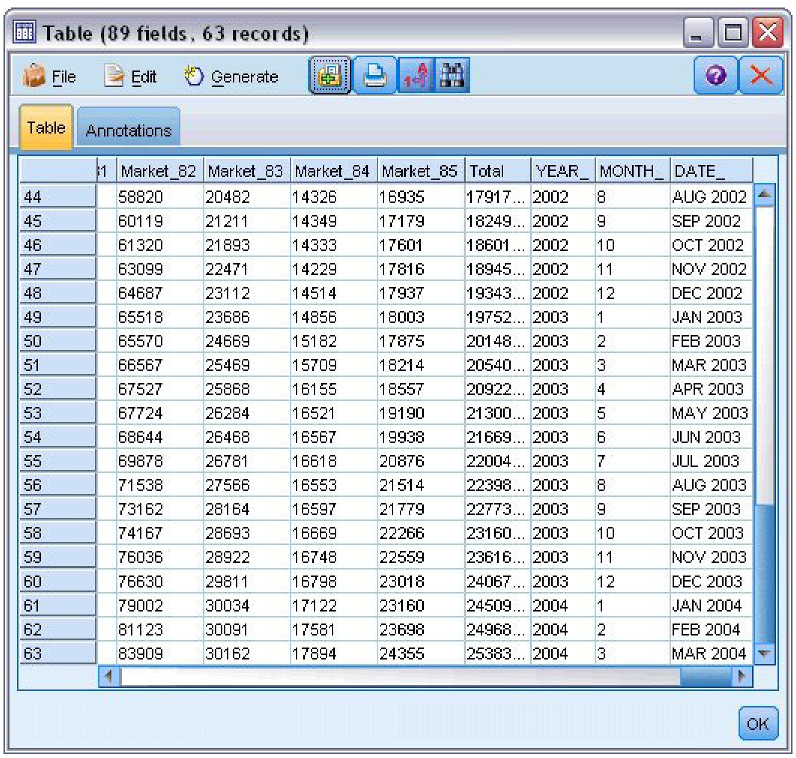


Figure 191. Opening the stream



	Market_82	Market_83	Market_84	Market_85	Total	YEAR_	MONTH_	DATE_
44	58820	20482	14326	16935	17917...	2002	8	AUG 2002
45	60119	21211	14349	17179	18249...	2002	9	SEP 2002
46	61320	21893	14333	17601	18601...	2002	10	OCT 2002
47	63099	22471	14229	17816	18945...	2002	11	NOV 2002
48	64687	23112	14514	17937	19343...	2002	12	DEC 2002
49	65518	23686	14856	18003	19752...	2003	1	JAN 2003
50	65570	24669	15182	17875	20148...	2003	2	FEB 2003
51	66567	25469	15709	18214	20540...	2003	3	MAR 2003
52	67527	25868	16155	18557	20922...	2003	4	APR 2003
53	67724	26284	16521	19190	21300...	2003	5	MAY 2003
54	68644	26468	16567	19938	21669...	2003	6	JUN 2003
55	69878	26781	16618	20876	22004...	2003	7	JUL 2003
56	71538	27566	16553	21514	22398...	2003	8	AUG 2003
57	73162	28164	16597	21779	22773...	2003	9	SEP 2003
58	74167	28693	16669	22266	23160...	2003	10	OCT 2003
59	76036	28922	16748	22559	23616...	2003	11	NOV 2003
60	76630	29811	16798	23018	24067...	2003	12	DEC 2003
61	79002	30034	17122	23160	24509...	2004	1	JAN 2004
62	81123	30091	17581	23698	24968...	2004	2	FEB 2004
63	83909	30162	17894	24355	25383...	2004	3	MAR 2004

Figure 192. Updated sales data

The updated monthly data is collected in *broadband_2.sav*.

2. Attach a Table node to the IBM SPSS Statistics File source node, open the Table node and click **Run**.

   Note: The data file has been updated with the actual sales data for January through March 2004, in rows 61 to 63.

3. Open the Time Intervals node on the stream.
4. Click the **Forecast** tab.
5. Ensure that **Extend records into the future** is set to **3**.

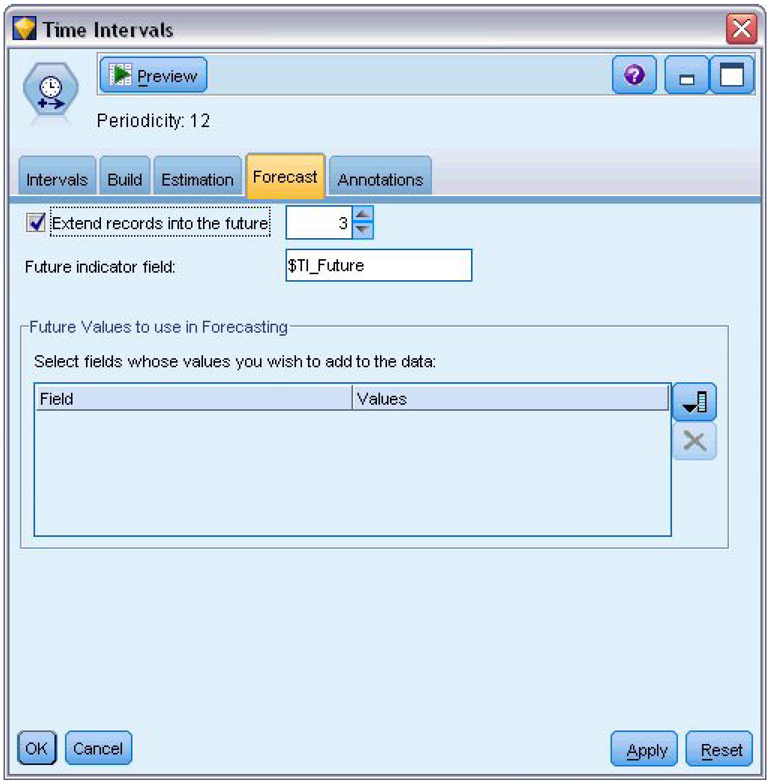


Figure 193. Checking the setting of the forecast period

Retrieving the Saved Model

1. On the IBM SPSS Modeler menu, choose **Insert** > **Node From File** and select the *TSmodel.nod* file from the *Demos* folder (or use the Time Series model you saved in the first time series example).

This file contains the time series models from the previous example. The insert operation places the corresponding Time Series model nugget on the canvas.

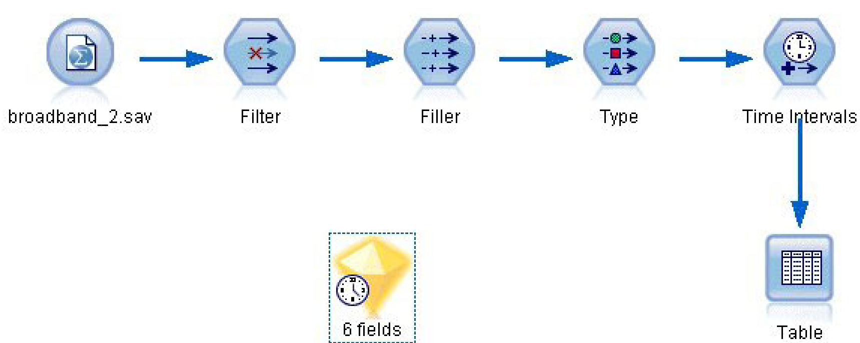


Figure 194. Adding the model nugget

Generating a Modeling Node

1. Open the Time Series model nugget and choose **Generate** > **Generate Modeling Node**.

This places a Time Series modeling node on the canvas.

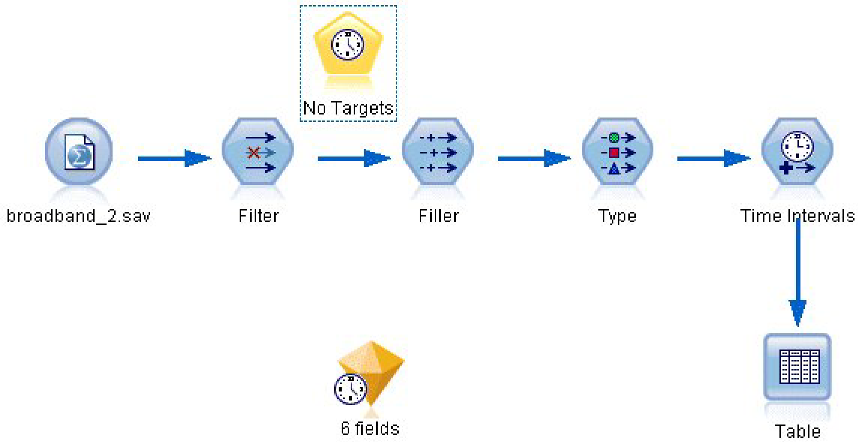


Figure 195. Generating a modeling node from the model nugget

Generating a New Model

1. Close the Time Series model nugget and delete it from the canvas.

   The old model was built on 60 rows of data. You need to generate a new model based on the updated sales data (63 rows).

2. Attach the newly generated Time Series build node to the stream.

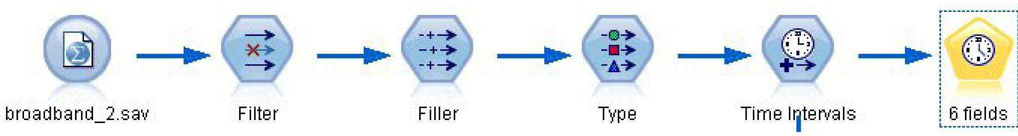


Figure 196. Attaching the modeling node to the stream

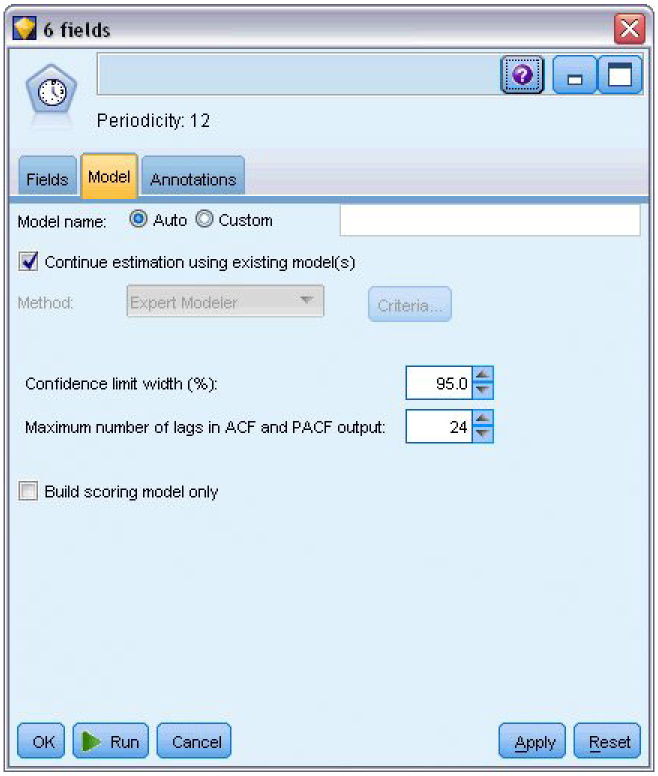
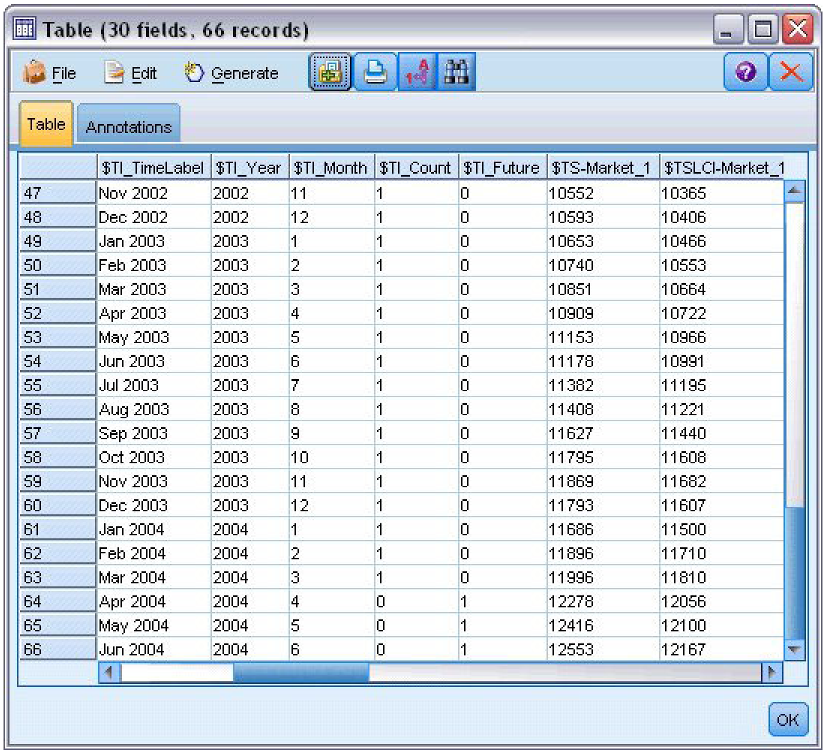


Figure 197. Reusing stored settings for the time series model

3. Open the Time Series node.
4. On the **Model** tab, ensure that **Continue estimation using existing models** is checked.
5. Click **Run** to place a new model nugget on the canvas and in the Models palette.

Examining the New Model



	$TI_TimeLabel	$TI_Year	$TI_Month	$TI_Count	$TI_Future	$TS-Market_1	$TSLCI-Market_1
47	Nov 2002	2002	11	1	0	10552	10365
48	Dec 2002	2002	12	1	0	10593	10406
49	Jan 2003	2003	1	1	0	10653	10466
50	Feb 2003	2003	2	1	0	10740	10553
51	Mar 2003	2003	3	1	0	10851	10664
52	Apr 2003	2003	4	1	0	10909	10722
53	May 2003	2003	5	1	0	11153	10966
54	Jun 2003	2003	6	1	0	11178	10991
55	Jul 2003	2003	7	1	0	11382	11195
56	Aug 2003	2003	8	1	0	11408	11221
57	Sep 2003	2003	9	1	0	11627	11440
58	Oct 2003	2003	10	1	0	11795	11608
59	Nov 2003	2003	11	1	0	11869	11682
60	Dec 2003	2003	12	1	0	11793	11607
61	Jan 2004	2004	1	1	0	11686	11500
62	Feb 2004	2004	2	1	0	11896	11710
63	Mar 2004	2004	3	1	0	11996	11810
64	Apr 2004	2004	4	0	1	12278	12056
65	May 2004	2004	5	0	1	12416	12100
66	Jun 2004	2004	6	0	1	12553	12167

Figure 198. Table showing new forecast

1. Attach a Table node to the new Time Series model nugget on the canvas.
2. Open the Table node and click **Run**.

   The new model still forecasts three months ahead because you're reusing the stored settings. However, this time it forecasts April through June because the estimation period (specified on the Time Intervals node) now ends in March instead of January.

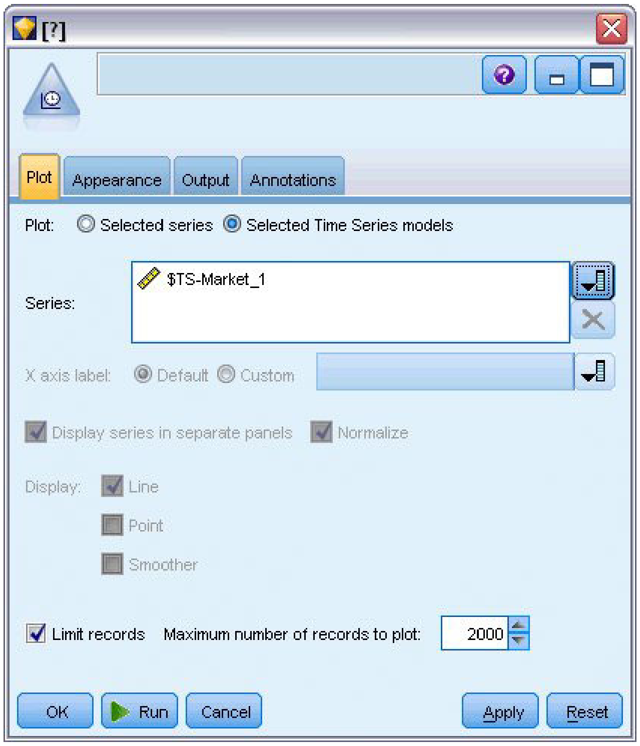


Figure 199. Specifying fields to plot

3. Attach a Time Plot graph node to the Time Series model nugget.

   This time we'll use the time plot display designed especially for time series models.
4. On the Plot tab, choose the **Selected Time Series models** option.
5. At the **Series** list, click the field selector button, select the *$TS-Market_*1 field and click **OK** to add it to the list.
6. Click **Run**.

Now you have a graph that shows the actual sales for *Market_1* up to March 2004, together with the forecast (Predicted) sales and the confidence interval (indicated by the blue shaded area) up to June 2004.

As in the first example, the forecast values follow the actual data closely throughout the time period, indicating once again that you have a good model.

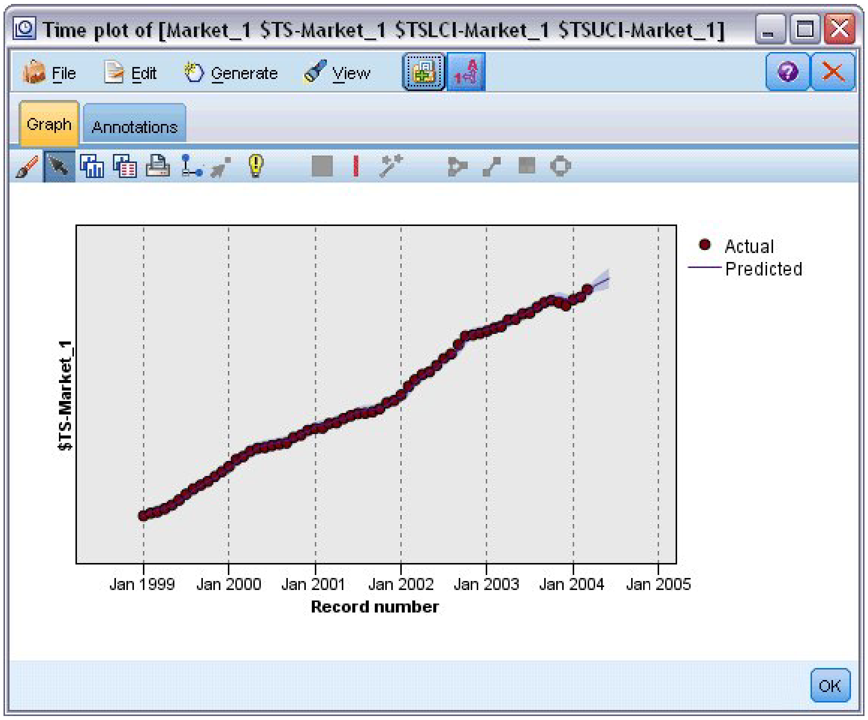


Figure 200. Forecast extended to June

Summary

You have learned how to apply saved models to extend your previous forecasts when more current data becomes available, and you have done this without rebuilding your models. Of course, if there is reason to think that a model has changed, you should rebuild it.

Chapter 15. Forecasting Catalog Sales (Time Series)

A catalog company is interested in forecasting monthly sales of its men's clothing line, based on their sales data for the last 10 years.

This example uses the stream named *catalog_forecast.str*, which references the data file named *catalog_seasfac.sav*. These files are available from the *Demos* directory of any IBM SPSS Modeler installation. This can be accessed from the IBM SPSS Modeler program group on the Windows Start menu. The *catalog_forecast.str* file is in the *streams* directory.

We've seen in an earlier example how you can let the Expert Modeler decide which is the most appropriate model for your time series. Now it's time to take a closer look at the two methods that are available when choosing a model yourself--exponential smoothing and ARIMA.

To help you decide on an appropriate model, it's a good idea to plot the time series first. Visual inspection of a time series can often be a powerful guide in helping you choose. In particular, you need to ask yourself:

- Does the series have an overall trend? If so, does the trend appear constant or does it appear to be dying out with time?
- Does the series show seasonality? If so, do the seasonal fluctuations seem to grow with time or do they appear constant over successive periods?

Creating the Stream

1. Create a new stream and add a Statistics File source node pointing to *catalog_seasfac.sav*.

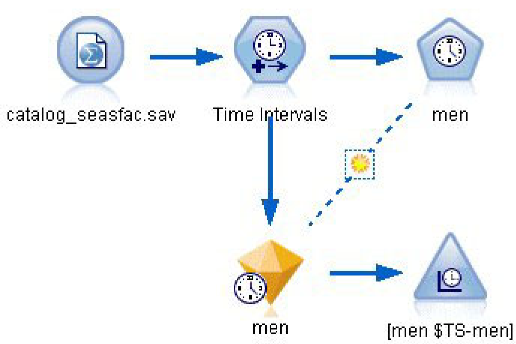


Figure 201. Forecasting catalog sales

Figure 202. Specifying the target field

2. Open the IBM SPSS Statistics File source node and select the Types tab.
3. Click **Read Values**, then **OK**.
4. Click the *Role* column for the *men* field and set the role to **Target**.
5. Set the role for all the other fields to **None**, and click **OK**.

Figure 203. Setting the time interval

6. Attach a Time Intervals node to the IBM SPSS Statistics File source node.
7. Open the Time Intervals node and set **Time Interval** to **Months**.
8. Select **Build from data**.
9. Set **Field** to **date**, and click **OK**.

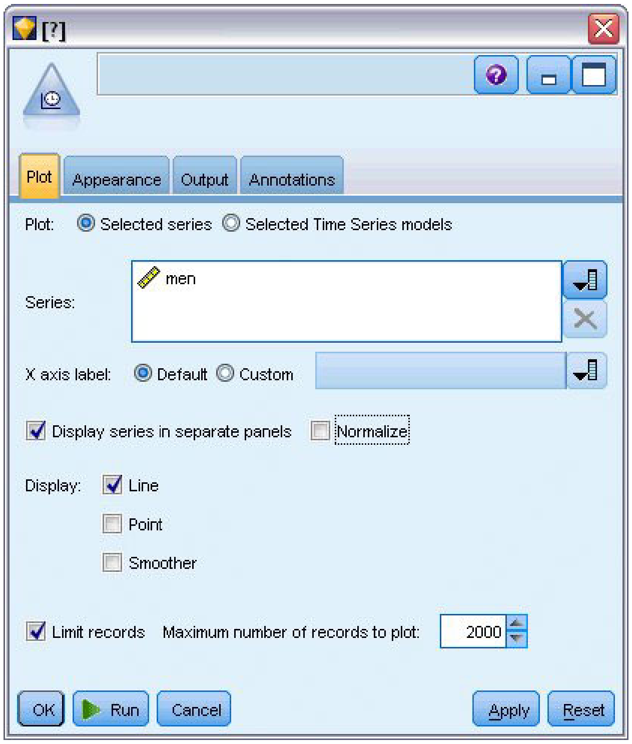


Figure 204. Plotting the time series

10. Attach a Time Plot node to the Time Intervals node.
11. On the Plot tab, add **men** to the Series list.
12. Deselect the **Normalize** check box.
13. Click **Run**.

Examining the Data

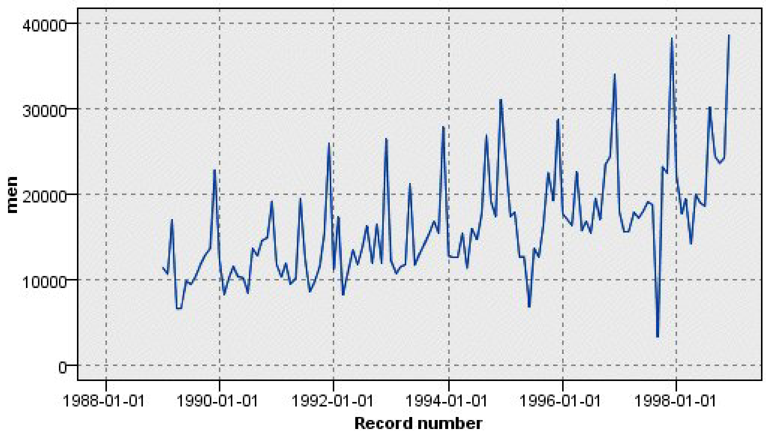


Figure 205. Actual sales of men's clothing

The series shows a general upward trend; that is, the series values tend to increase over time. The upward trend is seemingly constant, which indicates a linear trend.

The series also has a distinct seasonal pattern with annual highs in December, as indicated by the vertical lines on the graph. The seasonal variations appear to grow with the upward series trend, which suggests multiplicative rather than additive seasonality.

1. Click **OK** to close the plot.

Now that you've identified the characteristics of the series, you're ready to try modeling it. The exponential smoothing method is useful for forecasting series that exhibit trend, seasonality, or both. As we've seen, your data exhibit both characteristics.

Exponential Smoothing

Building a best-fit exponential smoothing model involves determining the model type—whether the model needs to include trend, seasonality, or both—and then obtaining the best-fit parameters for the chosen model.

The plot of men's clothing sales over time suggested a model with both a linear trend component and a multiplicative seasonality component. This implies a Winters model. First, however, we will explore a simple model (no trend and no seasonality) and then a Holt model (incorporates linear trend but no seasonality). This will give you practice in identifying when a model is not a good fit to the data, an essential skill in successful model building.

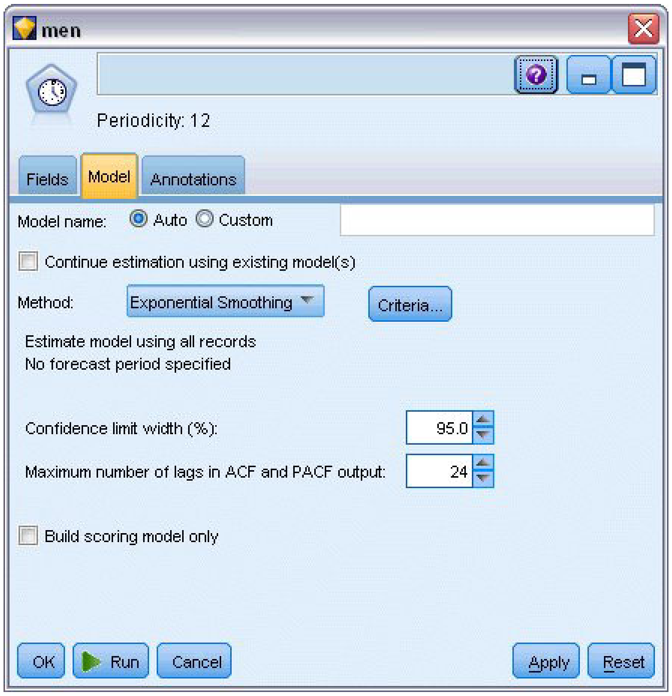


Figure 206. Specifying exponential smoothing

We'll start with a simple exponential smoothing model.

1. Attach a Time Series node to the Time Intervals node.
2. On the **Model** tab, set **Method** to **Exponential Smoothing**.
3. Click **Run** to create the model nugget.

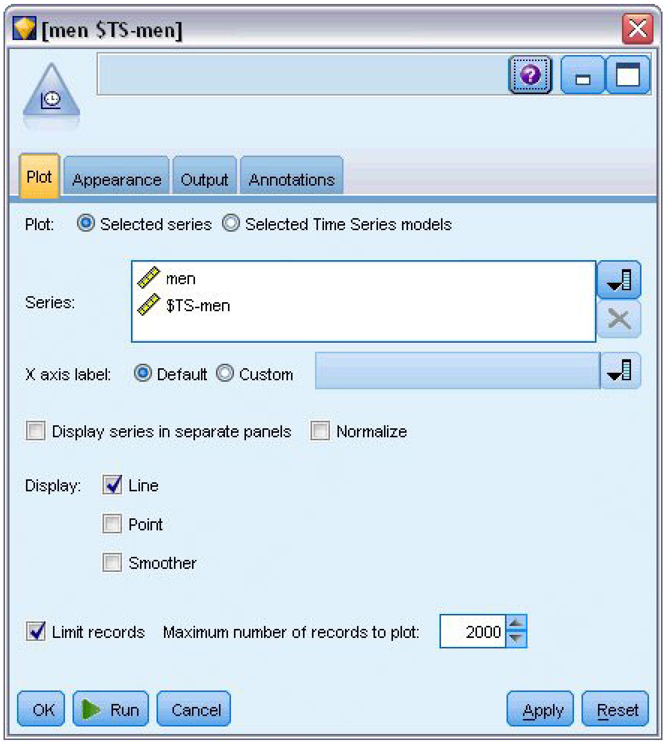


Figure 207. Plotting the Time Series model

4. Attach a Time Plot node to the model nugget.
5. On the **Plot** tab, add *men* and *$TS-men* to the **Series** list.
6. Deselect the **Display series in separate panels** and **Normalize** check boxes.
7. Click **Run**.

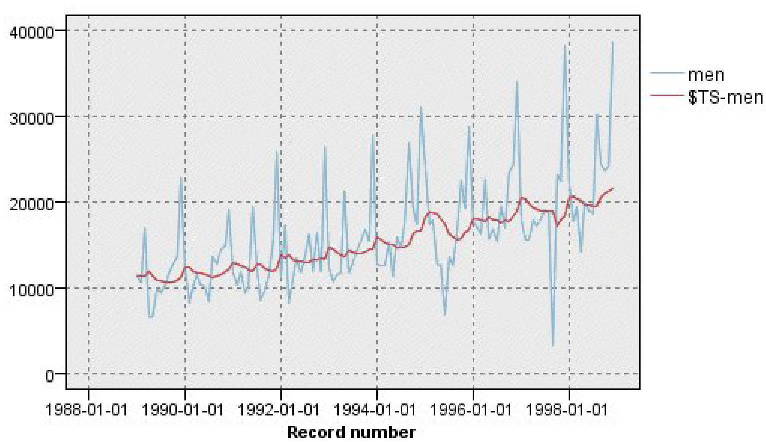


Figure 208. Simple exponential smoothing model

The **men** plot represents the actual data, while **$TS-men** denotes the time series model.

Although the simple model does, in fact, exhibit a gradual (and rather ponderous) upward trend, it takes no account of seasonality. You can safely reject this model.

8. Click **OK** to close the time plot window.

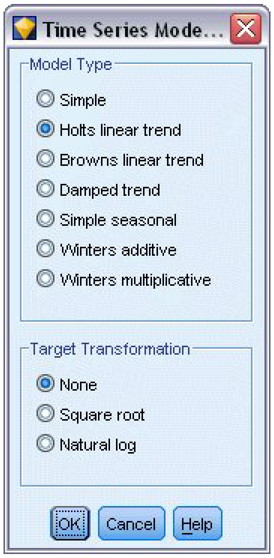


Figure 209. Selecting Holt's model

Let's try Holt's linear model. This should at least model the trend better than the simple model, although it too is unlikely to capture the seasonality.

9. Reopen the Time Series node.
10. On the **Model** tab, with **Exponential Smoothing** still selected as the method, click **Criteria**.
11. On the Exponential Smoothing Criteria dialog box, choose **Holts linear trend**.
12. Click **OK** to close the dialog box.
13. Click **Run** to re-create the model nugget.
14. Re-open the Time Plot node and click **Run**.

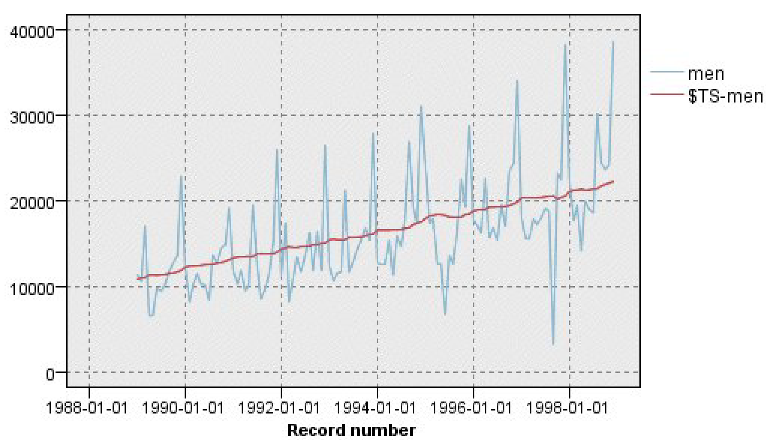


Figure 210. Holt's linear trend model

Holt's model displays a smoother upward trend than the simple model but it still takes no account of the seasonality, so you can discard this one too.

15. Close the time plot window.

You may recall that the initial plot of men's clothing sales over time suggested a model incorporating a linear trend and multiplicative seasonality. A more suitable candidate, therefore, might be Winters' model.

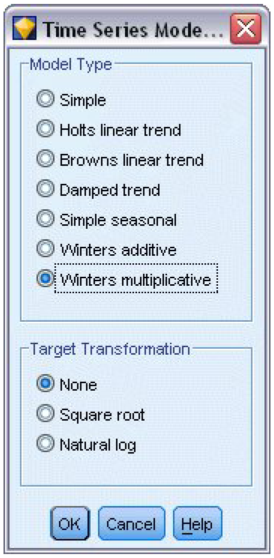


Figure 211. Selecting Winters' model

16. Reopen the Time Series node.
17. On the **Model** tab, with **Exponential Smoothing** still selected as the method, click **Criteria**.
18. On the Exponential Smoothing Criteria dialog box, choose **Winters multiplicative**.
19. Click **OK** to close the dialog box.
20. Click **Run** to re-create the model nugget.
21. Open the Time Plot node and click **Run**.

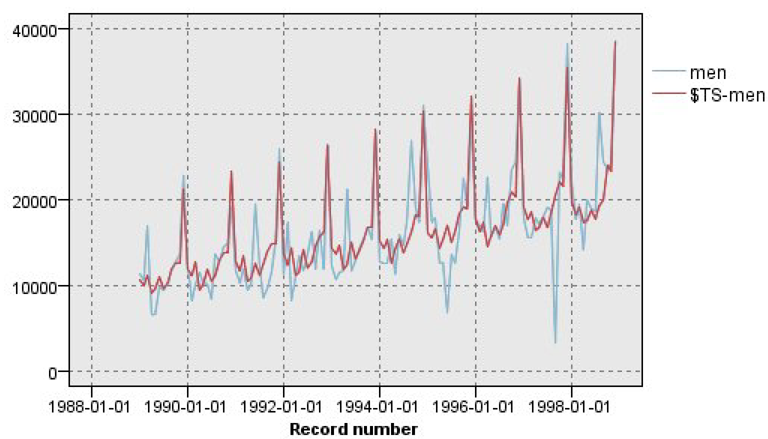


Figure 212. Winters' multiplicative model

This looks better--the model reflects both the trend and the seasonality of the data.

The dataset covers a period of 10 years and includes 10 seasonal peaks occurring in December of each year. The 10 peaks present in the predicted results match up well with the 10 annual peaks in the real data.

However, the results also underscore the limitations of the Exponential Smoothing procedure. Looking at both the upward and downward spikes, there is significant structure that is not accounted for.

If you are primarily interested in modeling a long-term trend with seasonal variation, then exponential smoothing may be a good choice. To model a more complex structure such as this one, we need to consider using the ARIMA procedure.

ARIMA

The ARIMA procedure allows you to create an autoregressive integrated moving-average (ARIMA) model suitable for finely tuned modeling of time series. ARIMA models provide more sophisticated methods for modeling trend and seasonal components than do exponential smoothing models, and they allow the added benefit of including predictor variables in the model.

Continuing the example of the catalog company that wants to develop a forecasting model, we have seen how the company has collected data on monthly sales of men's clothing along with several series that might be used to explain some of the variation in sales. Possible predictors include the number of catalogs mailed and the number of pages in the catalog, the number of phone lines open for ordering, the amount spent on print advertising, and the number of customer service representatives.

Are any of these predictors useful for forecasting? Is a model with predictors really better than one without? Using the ARIMA procedure, we can create a forecasting model with predictors, and see if there is a significant difference in predictive ability over the exponential smoothing model with no predictors.

The ARIMA method enables you to fine-tune the model by specifying orders of autoregression, differencing, and moving average, as well as seasonal counterparts to these components. Determining the best values for these components manually can be a time-consuming process involving a good deal of trial and error, so for this example, we'll let the Expert Modeler choose an ARIMA model for us.

We'll try to build a better model by treating some of the other variables in the dataset as predictor variables. The ones that seem most useful to include as predictors are the number of catalogs mailed (*mail*), the number of pages in the catalog (*page*), the number of phone lines open for ordering (*phone*), the amount spent on print advertising *(print),* and the number of customer service representatives *(service)*.

Figure 213. Setting the predictor fields

1. Open the IBM SPSS Statistics File source node.
2. On the Types tab, set the *Role* for *mail*, *page*, *phone*, *print*, and *service* to **Input**.

3. Ensure that the role for **men** is set to **Target** and that all the remaining fields are set to **None**.
4. Click **OK**.

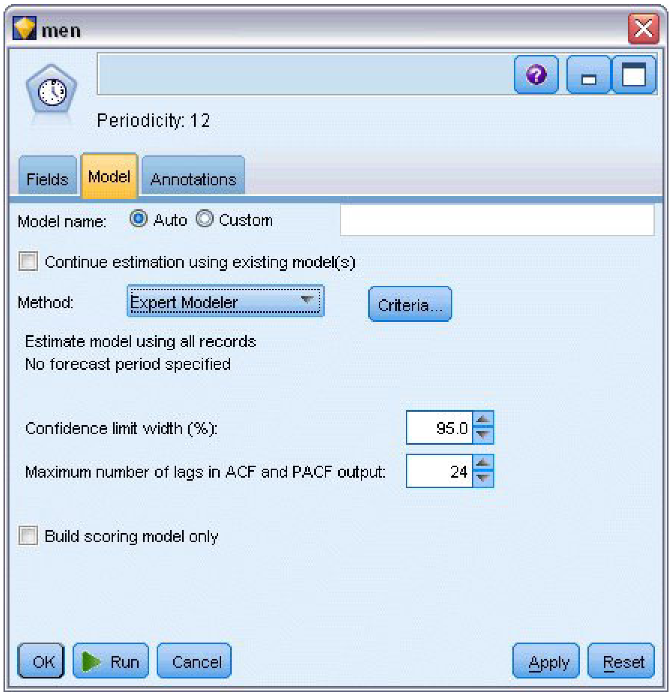


Figure 214. Choosing the Expert Modeler

5. Open the Time Series node.
6. On the Model tab, set **Method** to **Expert Modeler** and click **Criteria**.

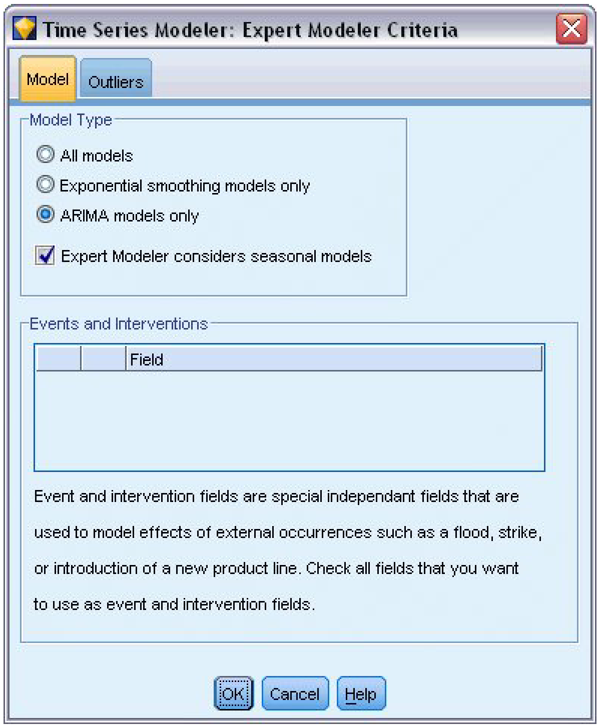


Figure 215. Choosing only ARIMA models

7. On the Expert Modeler Criteria dialog box, choose the **ARIMA models only** option and ensure that **Expert Modeler considers seasonal models** is checked.
8. Click **OK** to close the dialog box.
9. Click **Run** on the Model tab to re-create the model nugget.

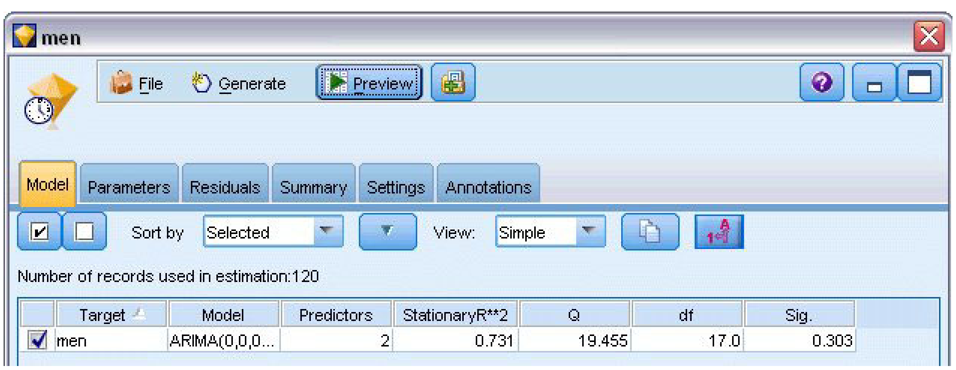


Figure 216. Expert Modeler chooses two predictors

10. Open the model nugget.

    Notice how the Expert Modeler has chosen only two of the five specified predictors as being significant to the model.
11. Click **OK** to close the model nugget.
12. Open the Time Plot node and click **Run**.

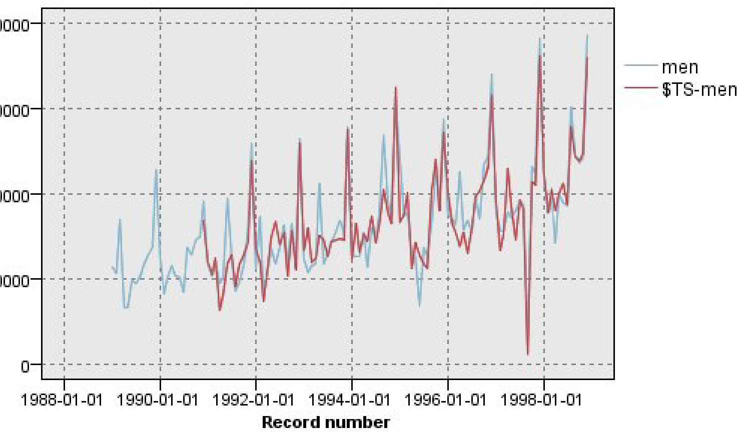


Figure 217. ARIMA model with predictors specified

This model improves on the previous one by capturing the large downward spike as well, making it the best fit so far.

We could try refining the model even further, but any improvements from this point on are likely to be minimal. We've established that the ARIMA model with predictors is preferable, so let's use the model we have just built. For the purposes of this example, we'll forecast sales for the coming year.

13. Click **OK** to close the time plot window.
14. Open the Time Intervals node and select the *Forecast* tab.
15. Select the *Extend records into the future* checkbox and set its value to 12.

The use of predictors when forecasting requires you to specify estimated values for those fields in the forecast period, so that the modeler can more accurately forecast the target field.

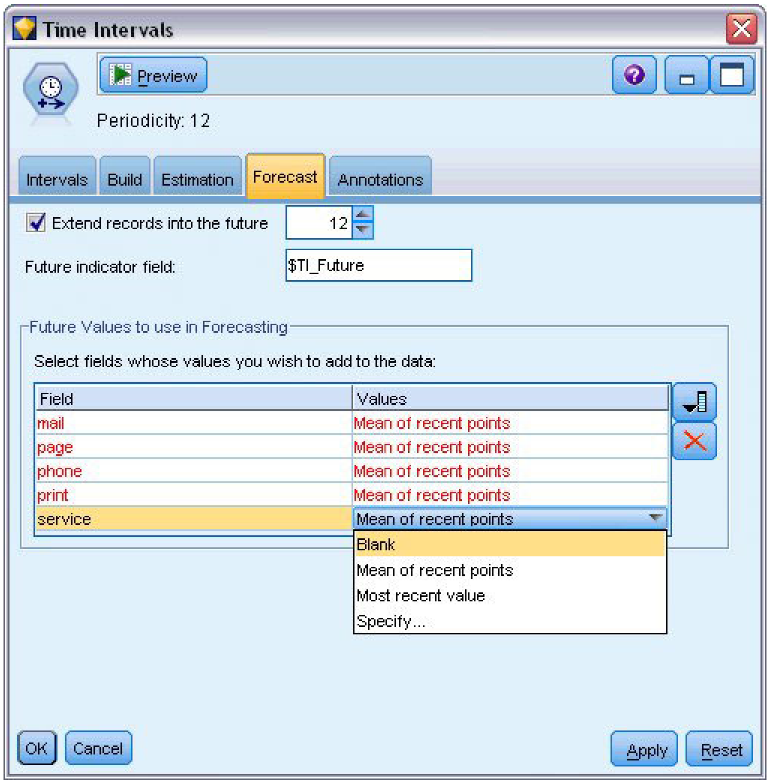


Figure 218. Specifying future values for predictor fields

16. In the **Future Values to use in Forecasting** group, click the field selector button to the right of the Values column.
17. On the Select Fields dialog box, select **mail** through **service** and click **OK**.

    In the real world, you would specify the future values manually at this point, since these five predictors all relate to items that are under your control. For the purposes of this example, we'll use one of the predefined functions, to save having to specify 12 values for each predictor. (When you're more familiar with this example, you might want to try experimenting with different future values to see what effect they have on the model.)
18. For each field in turn, click the **Values** field to display the list of possible values and choose **Mean of recent points**. This option calculates the mean of the last three data points for this field and uses that as the estimated value in each case.
19. Click **OK**.
20. Open the Time Series node and click **Run** to re-create the model nugget.
21. Open the Time Plot node and click **Run**.

The forecast for 1999 looks good--as expected, there's a return to normal sales levels following the December peak, and a steady upward trend in the second half of the year, with sales in general significantly above those for the previous year.

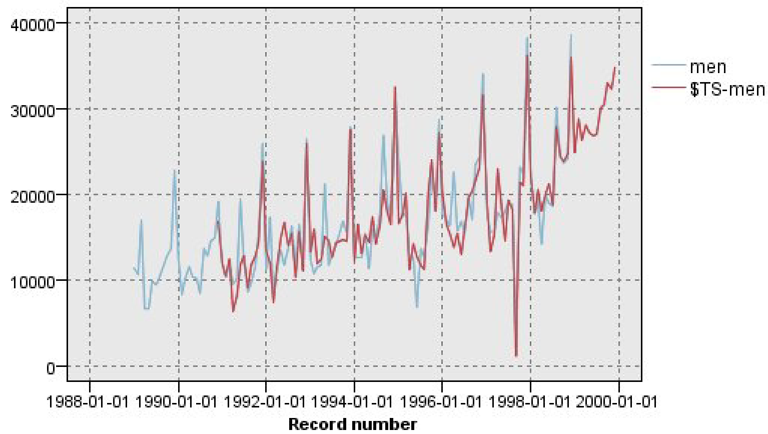


Figure 219. Sales forecast with predictors specified

Summary

You have successfully modeled a complex time series, incorporating not only an upward trend but also seasonal and other variations. You have also seen how, through trial and error, you can get closer and closer to an accurate model, which you have then used to forecast future sales.

In practice, you would need to reapply the model as your actual sales data are updated--for example, every month or every quarter--and produce updated forecasts. See the topic "Reapplying a Time Series Model" on page 168 for more information.

Chapter 16. Making Offers to Customers (Self-Learning)

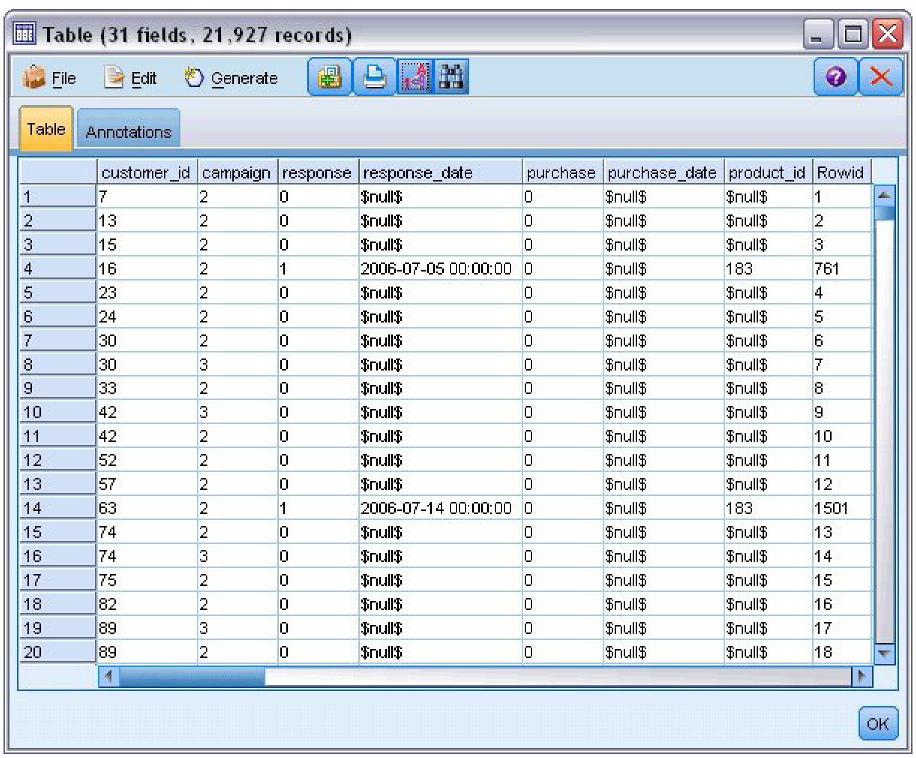
The Self-Learning Response Model (SLRM) node generates and enables the updating of a model that allows you to predict which offers are most appropriate for customers and the probability of the offers being accepted. These sorts of models are most beneficial in customer relationship management, such as marketing applications or call centers.

This example is based on a fictional banking company. The marketing department wants to achieve more profitable results in future campaigns by matching the right offer of financial services to each customer. Specifically, the example uses a Self-Learning Response Model to identify the characteristics of customers who are most likely to respond favorably based on previous offers and responses and to promote the best current offer based on the results.

This example uses the stream *pm_selflearn.str*, which references the data files *pm_customer_train1.sav*, *pm_customer_train2.sav*, and *pm_customer_train3.sav*. These files are available from the *Demos* folder of any IBM SPSS Modeler installation. This can be accessed from the IBM SPSS Modeler program group on the Windows Start menu. The *pm_selflearn.str* file is in the *streams* folder.

Existing Data

The company has historical data tracking the offers made to customers in past campaigns, along with the responses to those offers. These data also include demographic and financial information that can be used to predict response rates for different customers.



	customer_id	campaign	response	response_date	purchase	purchase_date	product_id	Rowid
1	7	2	0	$null$	0	$null$	$null$	1
2	13	2	0	$null$	0	$null$	$null$	2
3	15	2	0	$null$	0	$null$	$null$	3
4	16	2	1	2006-07-05 00:00:00	0	$null$	183	761
5	23	2	0	$null$	0	$null$	$null$	4
6	24	2	0	$null$	0	$null$	$null$	5
7	30	2	0	$null$	0	$null$	$null$	6
8	30	3	0	$null$	0	$null$	$null$	7
9	33	2	0	$null$	0	$null$	$null$	8
10	42	3	0	$null$	0	$null$	$null$	9
11	42	2	0	$null$	0	$null$	$null$	10
12	52	2	0	$null$	0	$null$	$null$	11
13	57	2	0	$null$	0	$null$	$null$	12
14	63	2	1	2006-07-14 00:00:00	0	$null$	183	1501
15	74	2	0	$null$	0	$null$	$null$	13
16	74	3	0	$null$	0	$null$	$null$	14
17	75	2	0	$null$	0	$null$	$null$	15
18	82	2	0	$null$	0	$null$	$null$	16
19	89	3	0	$null$	0	$null$	$null$	17
20	89	2	0	$null$	0	$null$	$null$	18

Figure 220. Responses to previous offers

Building the Stream

1. Add a Statistics File source node pointing to *pm_customer_train1.sav*, located in the *Demos* folder of your IBM SPSS Modeler installation.

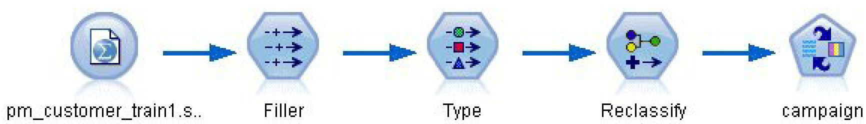


Figure 221. SLRM sample stream

2. Add a Filler node and select `campaign` as the Fill in field.
3. Select a Replace type of **Always**.
4. In the Replace with text box, enter `to_string(campaign)` and click **OK**.

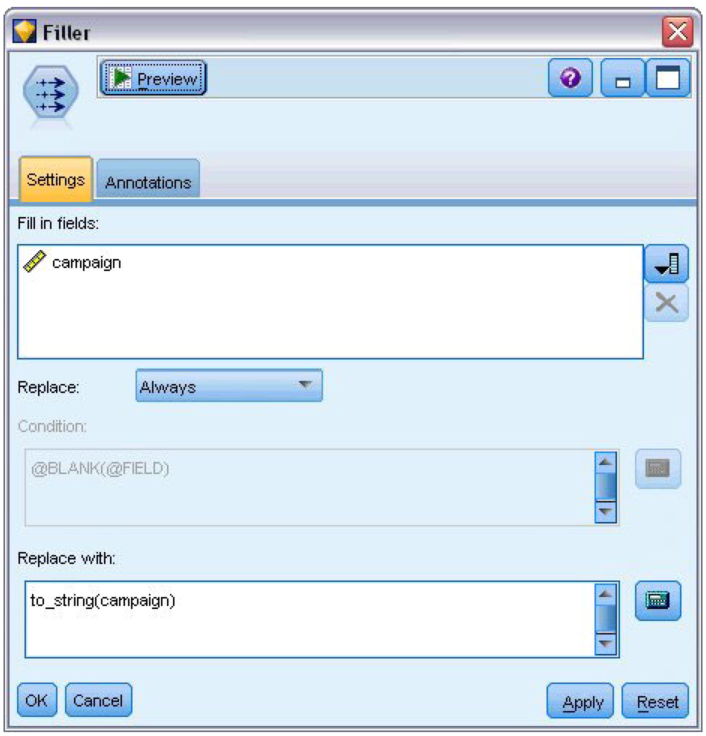


Figure 222. Derive a campaign field

5. Add a Type node, and set the *Role* to **None** for the *customer_id*, *response_date*, *purchase_date*, *product_id*, *Rowid*, and *X_random* fields.

Figure 223. Changing the Type node settings

6. Set the *Role* to **Target** for the *campaign* and *response* fields. These are the fields on which you want to base your predictions.

   Set the **Measurement** to **Flag** for the *response* field.
7. Click **Read Values**, then **OK**.

   Because the campaign field data show as a list of numbers (1, 2, 3, and 4), you can reclassify the fields to have more meaningful titles.
8. Add a Reclassify node to the Type node.
9. In the **Reclassify into** field, select **Existing field**.
10. In the **Reclassify field** list, select **campaign**.
11. Click the **Get** button; the campaign values are added to the *Original value* column.
12. In the *New value* column, enter the following campaign names in the first four rows:
    - **Mortgage**
    - **Car loan**
    - **Savings**
    - **Pension**
13. Click **OK**.

Figure 224. Reclassify the campaign names

14. Attach an SLRM modeling node to the Reclassify node. On the Fields tab, select **campaign** for the Target field, and **response** for the Target response field.

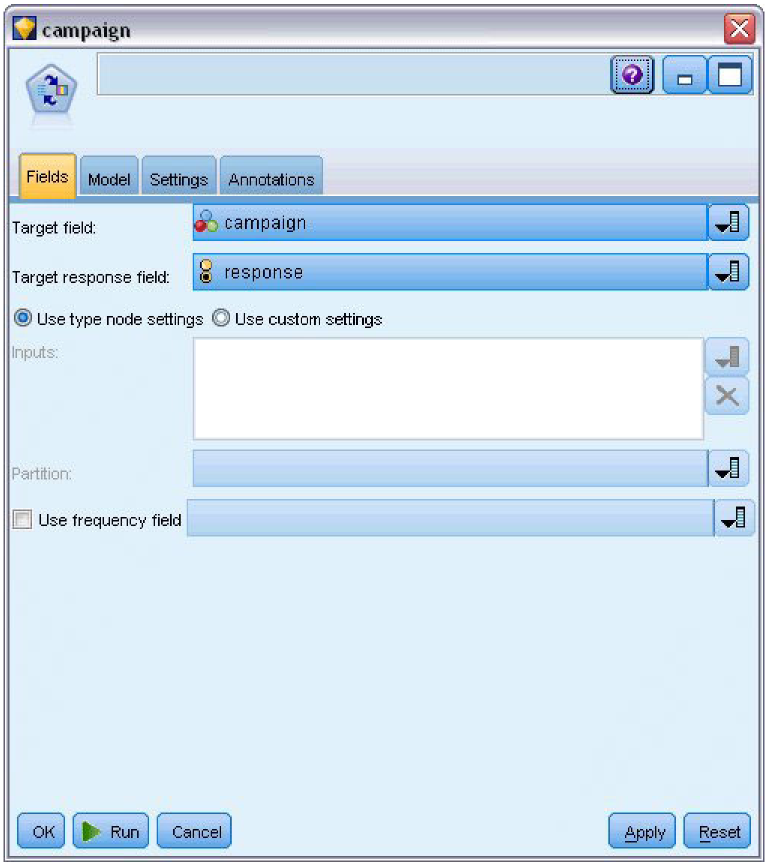


Figure 225. Select the target and target response

15. On the Settings tab, in the Maximum number of predictions per record field, reduce the number to 2. This means that for each customer, there will be two offers identified that have the highest probability of being accepted.
16. Ensure that **Take account of model reliability** is selected, and click **Run**.

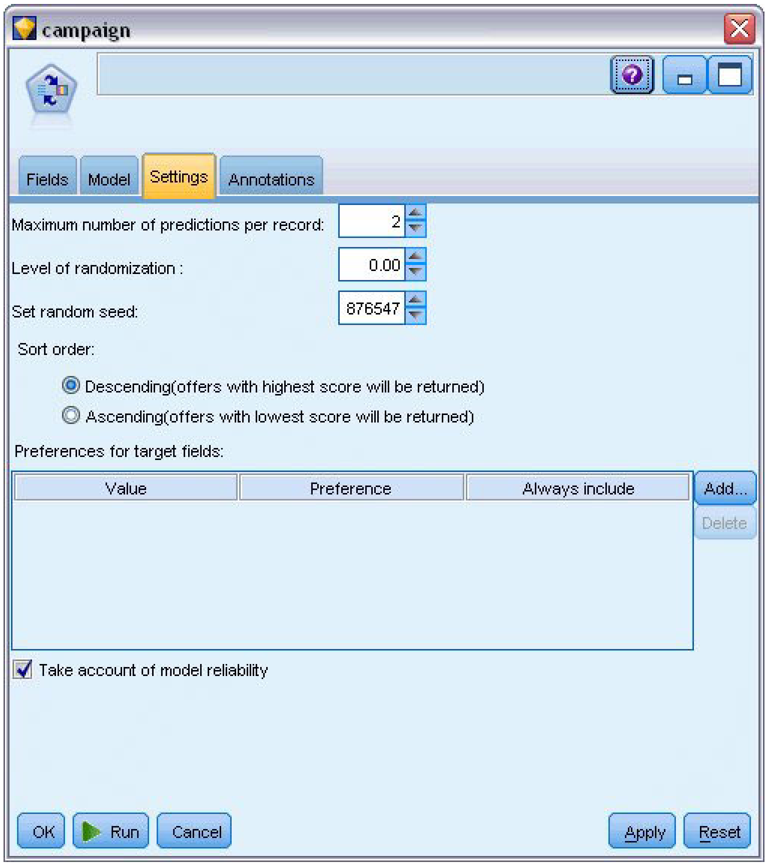


Figure 226. SLRM node settings

Browsing the Model

1. Open the model nugget. The Model tab initially shows the estimated the accuracy of the predictions for each offer and the relative importance of each predictor in estimating the model.

   To display the correlation of each predictor with the target variable, choose **Association with Response** from the **View** list in the right-hand pane.
2. To switch between each of the four offers for which there are predictions, select the required offer from the **View** list in the left-hand pane.

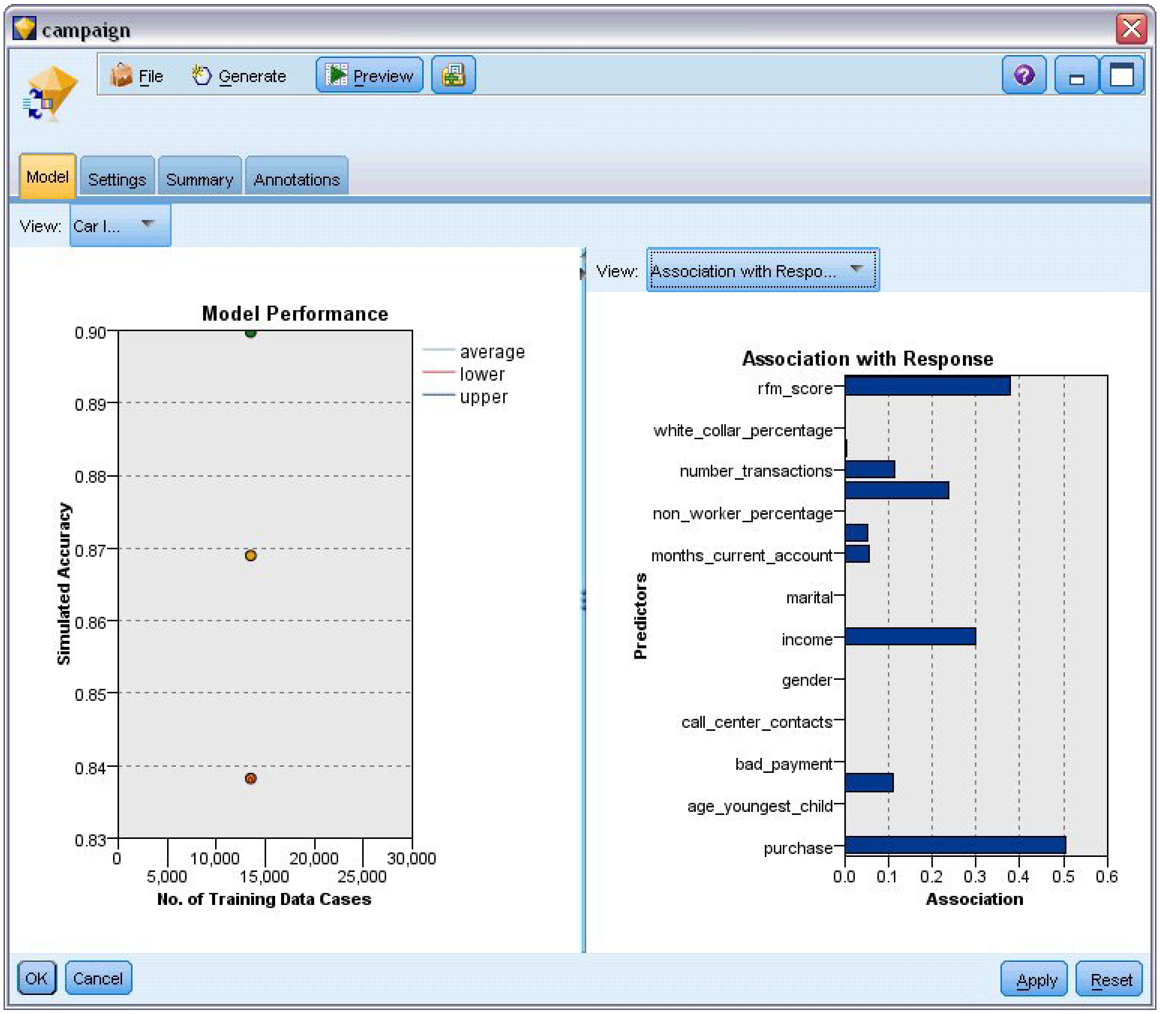


Figure 227. SLRM model nugget

3. Close the model nugget window.
4. On the stream canvas, disconnect the IBM SPSS Statistics File source node pointing to *pm_customer_train1.sav*.
5. Add a Statistics File source node pointing to *pm_customer_train2.sav*, located in the *Demos* folder of your IBM SPSS Modeler installation, and connect it to the Filler node.

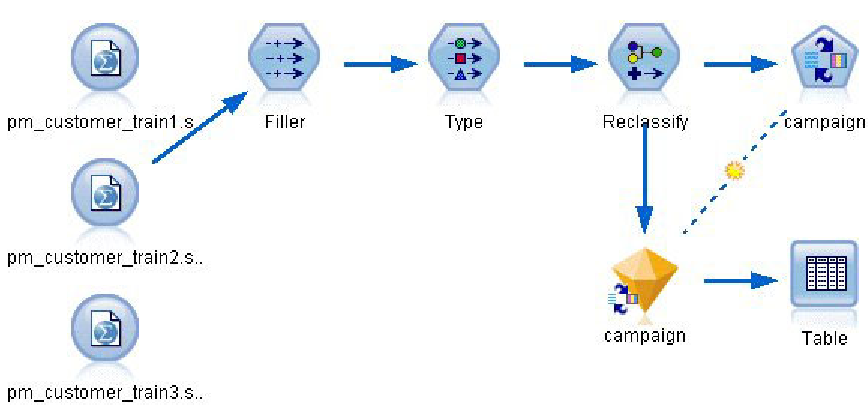


Figure 228. Attaching second data source to SLRM stream

6. On the Model tab of the SLRM node, select **Continue training existing model**.

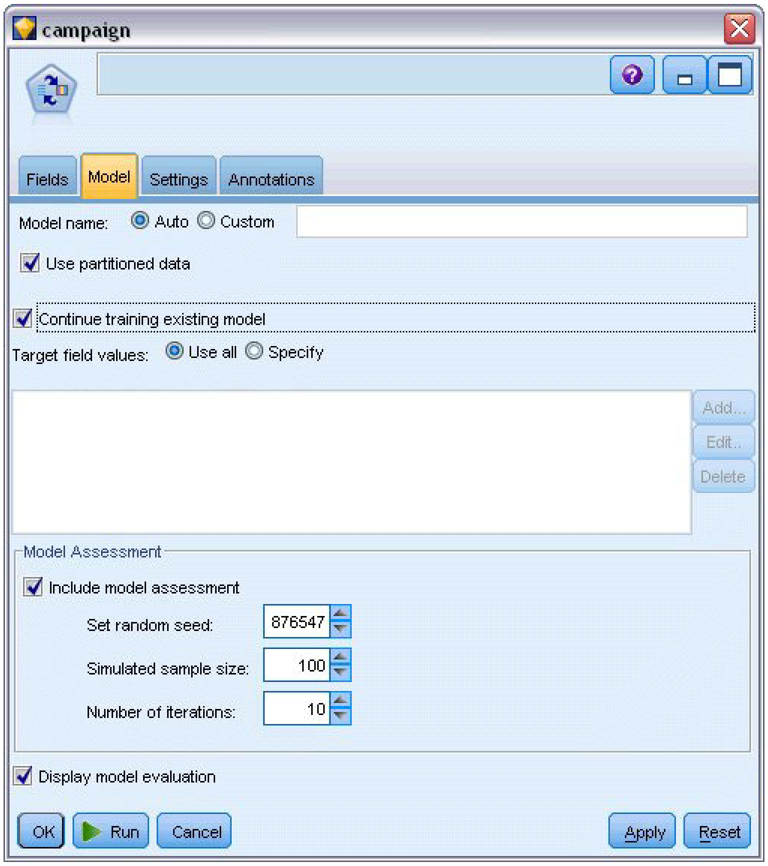


Figure 229. Continue training model

7. Click **Run** to re-create the model nugget. To view its details, double-click the nugget on the canvas.

   The Model tab now shows the revised estimates of the accuracy of the predictions for each offer.

8. Add a Statistics File source node pointing to *pm_customer_train3.sav*, located in the *Demos* folder of your IBM SPSS Modeler installation, and connect it to the Filler node.

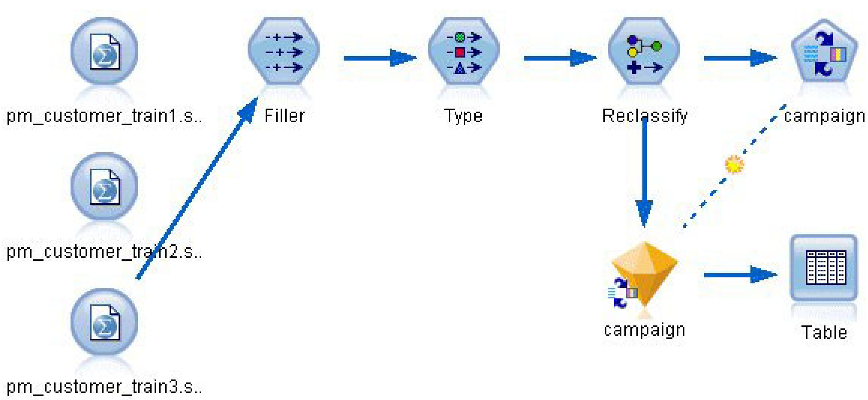


Figure 230. Attaching third data source to SLRM stream

9. Click **Run** to re-create the model nugget once more. To view its details, double-click the nugget on the canvas.

10. The Model tab now shows the final estimated accuracy of the predictions for each offer.

As you can see, the average accuracy fell slightly (from 86.9% to 85.4%) as you added the additional data sources; however, this fluctuation is a minimal amount and may be attributed to slight anomalies within the available data.

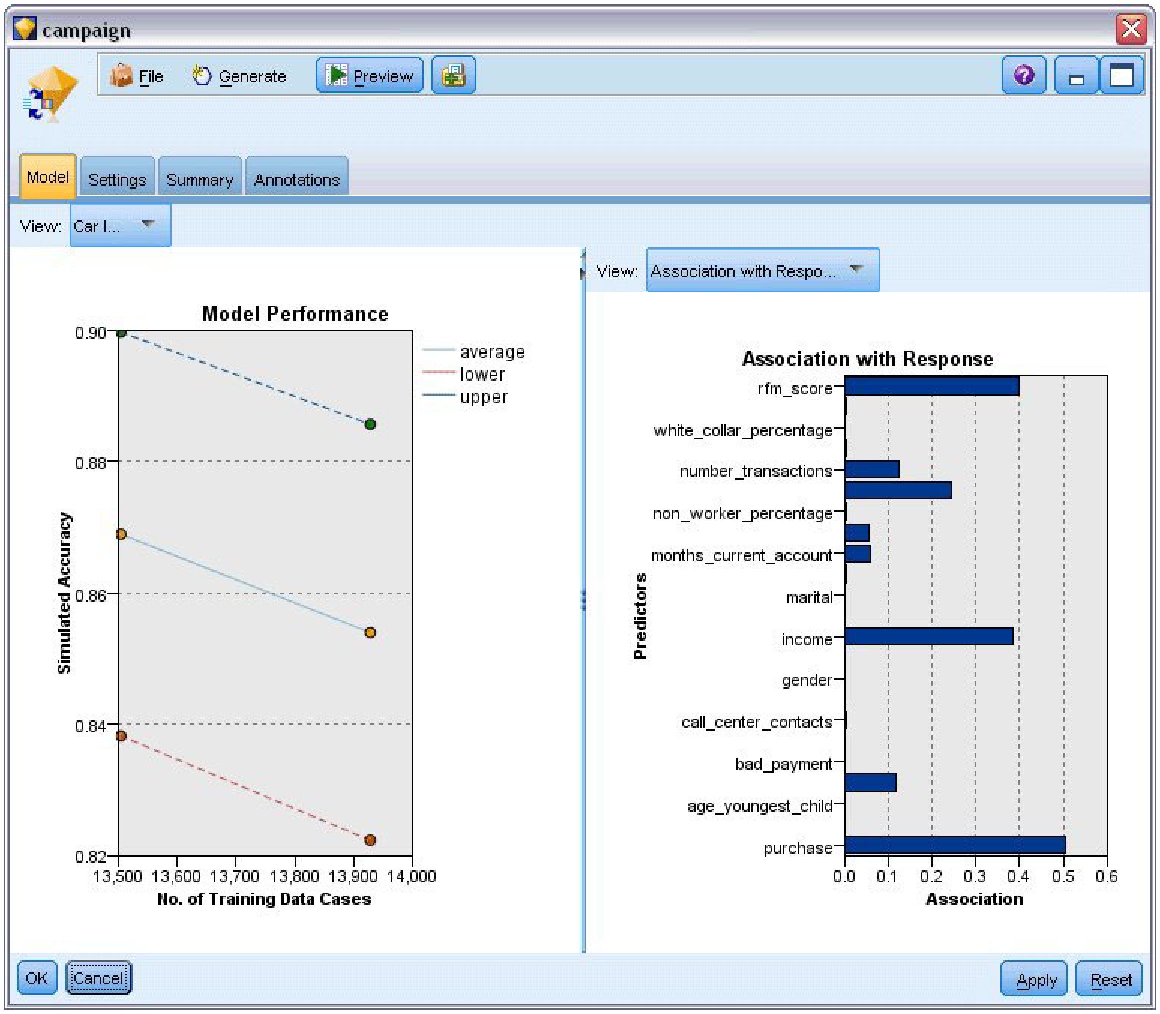
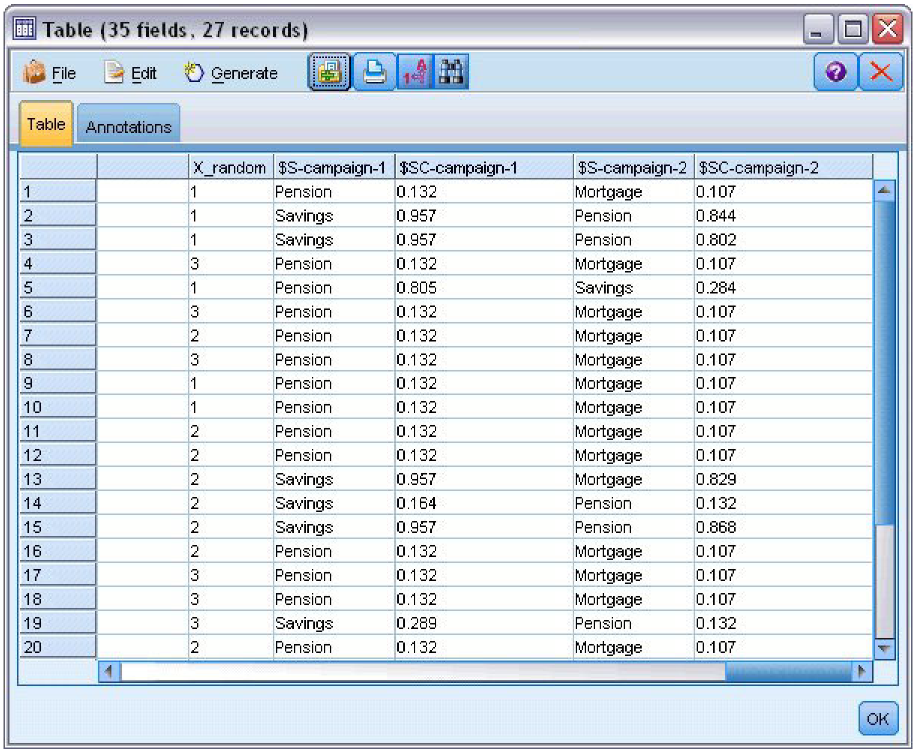


Figure 231. Updated SLRM model nugget

11. Attach a Table node to the last (third) generated model and execute the Table node.
12. Scroll across to the right of the table. The predictions show which offers a customer is most likely to accept and the confidence that they will accept, depending on each customer's details.

For example, in the first line of the table shown, there is only a 13.2% confidence rating (denoted by the value 0.132 in the *$SC-campaign-1* column)) that a customer who previously took out a car loan will accept a pension if offered one . However, the second and third lines show two more customers who also took out a car loan; in their cases, there is a 95.7% confidence that they, and other customers with similar histories, would open a savings account if offered one, and over 80% confidence that they would accept a pension.



		X_random	$S-campaign-1	$SC-campaign-1	$S-campaign-2	$SC-campaign-2
1		1	Pension	0.132	Mortgage	0.107
2		1	Savings	0.957	Pension	0.844
3		1	Savings	0.957	Pension	0.802
4		3	Pension	0.132	Mortgage	0.107
5		1	Pension	0.805	Savings	0.284
6		3	Pension	0.132	Mortgage	0.107
7		2	Pension	0.132	Mortgage	0.107
8		3	Pension	0.132	Mortgage	0.107
9		1	Pension	0.132	Mortgage	0.107
10		1	Pension	0.132	Mortgage	0.107
11		2	Pension	0.132	Mortgage	0.107
12		2	Pension	0.132	Mortgage	0.107
13		2	Savings	0.957	Mortgage	0.829
14		2	Savings	0.164	Pension	0.132
15		2	Savings	0.957	Pension	0.868
16		2	Pension	0.132	Mortgage	0.107
17		3	Pension	0.132	Mortgage	0.107
18		3	Pension	0.132	Mortgage	0.107
19		3	Savings	0.289	Pension	0.132
20		2	Pension	0.132	Mortgage	0.107

Figure 232. Model output - predicted offers and confidences

Explanations of the mathematical foundations of the modeling methods used in IBM SPSS Modeler are listed in the *IBM SPSS Modeler Algorithms Guide*, available from the *Documentation* directory of the product DVD.

Note also that these results are based on the training data only. To assess how well the model generalizes to other data in the real world, you would use a Partition node to hold out a subset of records for purposes of testing and validation. .

Chapter 17. Predicting Loan Defaulters (Bayesian Network)

Bayesian networks enable you to build a probability model by combining observed and recorded evidence with "common-sense" real-world knowledge to establish the likelihood of occurrences by using seemingly unlinked attributes.

This example uses the stream named *bayes_bankloan.str*, which references the data file named *bankloan.sav*. These files are available from the *Demos* directory of any IBM SPSS Modeler installation and can be accessed from the IBM SPSS Modeler program group on the Windows Start menu. The *bayes_bankloan.str* file is in the *streams* directory.

For example, suppose a bank is concerned about the potential for loans not to be repaid. If previous loan default data can be used to predict which potential customers are liable to have problems repaying loans, these "bad risk" customers can either be declined a loan or offered alternative products.

This example focuses on using existing loan default data to predict potential future defaulters, and looks at three different Bayesian network model types to establish which is better at predicting in this situation.

Building the Stream

1. Add a Statistics File source node pointing to *bankloan.sav* in the *Demos* folder.

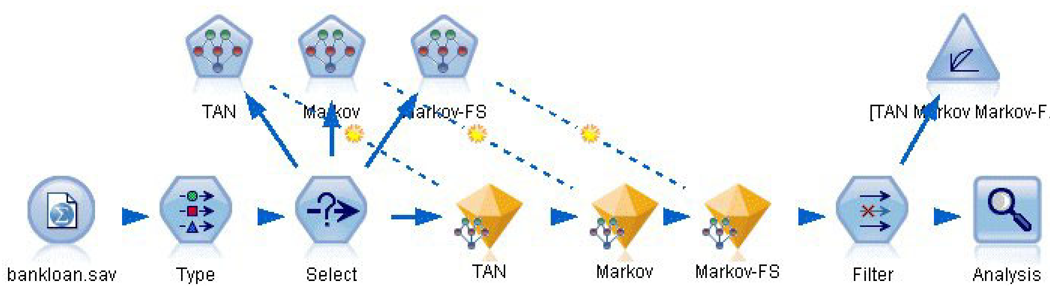


Figure 233. Bayesian Network sample stream

2. Add a Type node to the source node and set the role of the **default** field to **Target**. All other fields should have their role set to **Input**.
3. Click the **Read Values** button to populate the *Values* column.

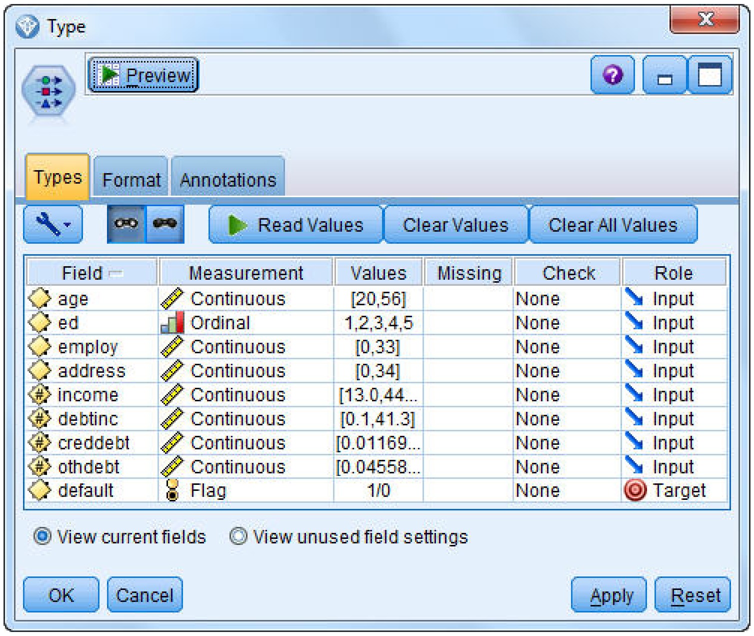


Figure 234. Selecting the target field

Cases where the target has a null value are of no use when building the model. You can exclude those cases to prevent them from being used in model evaluation.

4. Add a Select node to the Type node.
5. For Mode, select **Discard**.
6. In the Condition box, enter **default = '$null$'**.

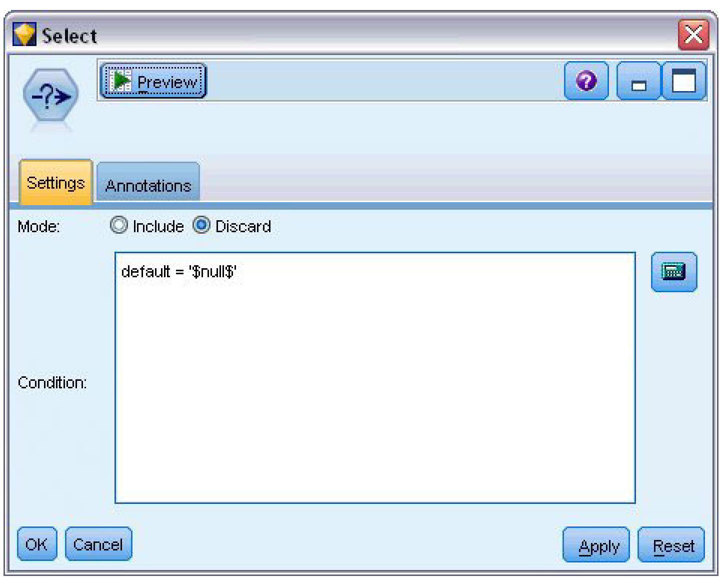


Figure 235. Discarding null targets

Because you can build several different types of Bayesian networks, it is worth comparing several to see which model provides the best predictions. The first one to create is a Tree Augmented Naïve Bayes (TAN) model.

7. Attach a Bayesian Network node to the Select node.
8. On the Model tab, for Model name, select **Custom** and enter `TAN` in the text box.

9. For Structure type, select **TAN** and click **OK**.

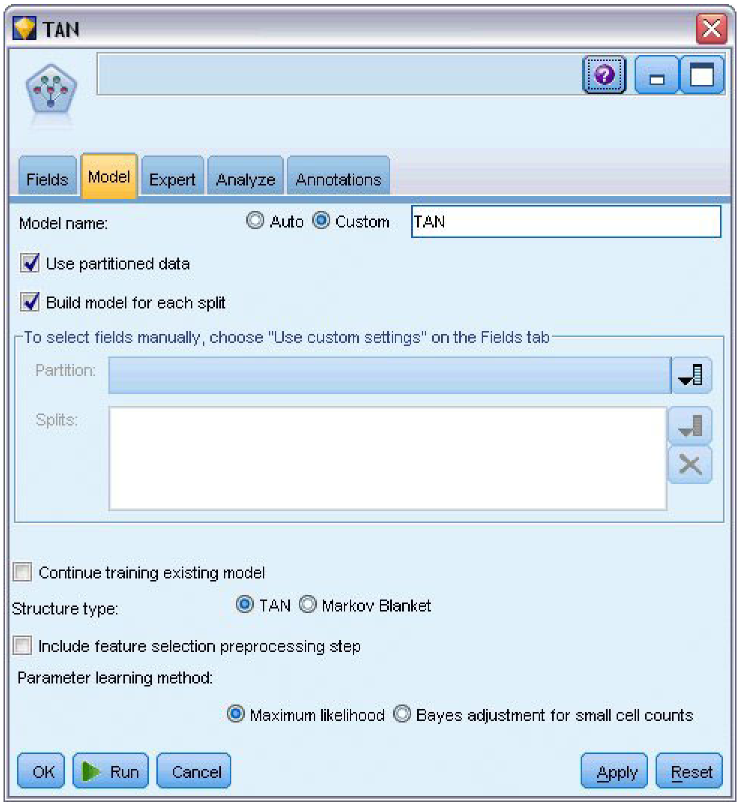


Figure 236. Creating a Tree Augmented Naïve Bayes model

The second model type to build has a Markov Blanket structure.

10. Attach a second Bayesian Network node to the Select node.
11. On the Model tab, for Model name, select **Custom** and enter `Markov` in the text box.
12. For Structure type, select **Markov Blanket** and click **OK**.

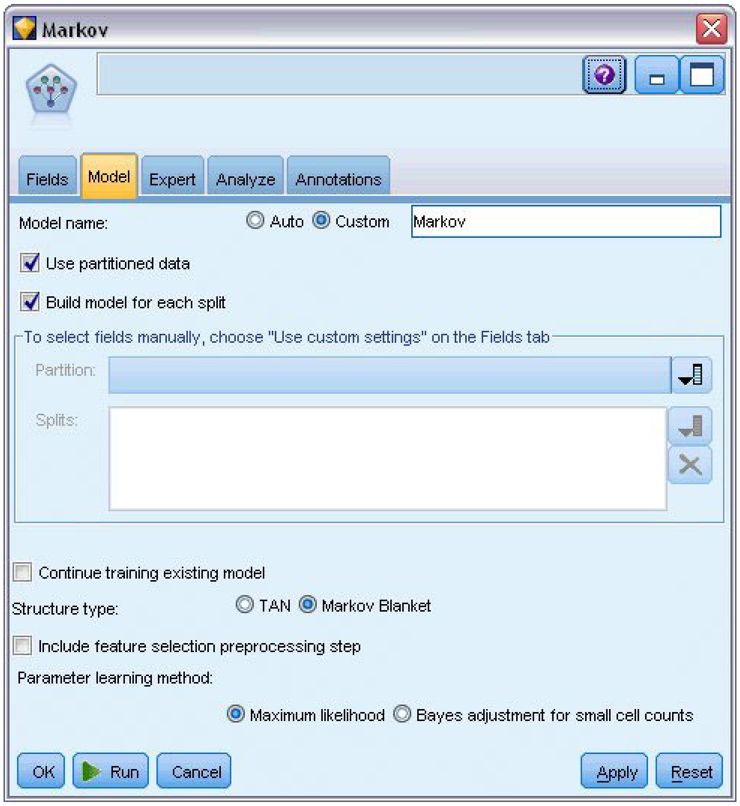


Figure 237. Creating a Markov Blanket model

The third model type to build has a Markov Blanket structure and also uses feature selection preprocessing to select the inputs that are significantly related to the target variable.

13. Attach a third Bayesian Network node to the Select node.
14. On the Model tab, for Model name, select **Custom** and enter `Markov-FS` in the text box.
15. For Structure type, select **Markov Blanket**.
16. Select **Include feature selection preprocessing step** and click **OK**.

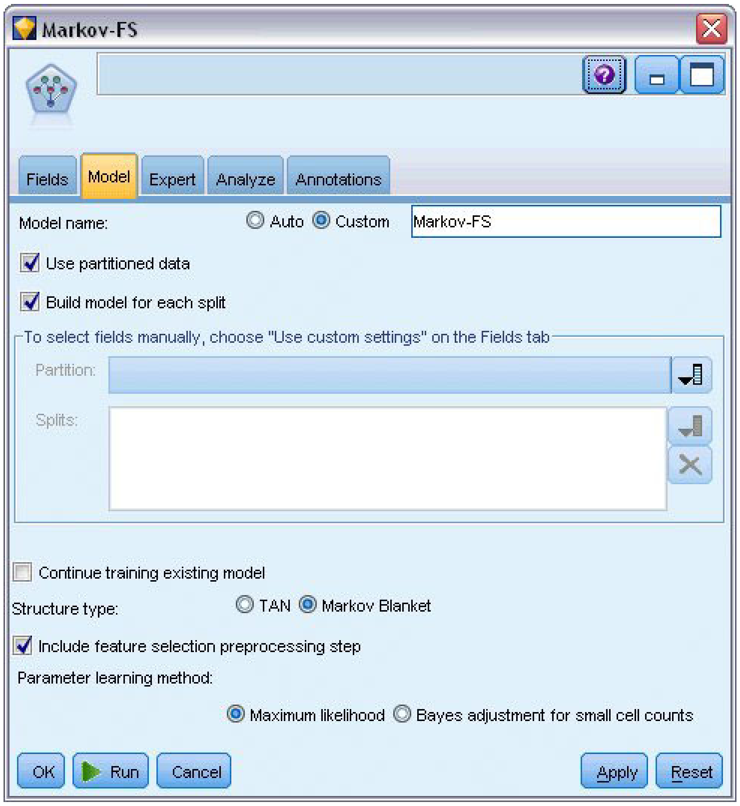


Figure 238. Creating a Markov Blanket model with Feature Selection preprocessing

Browsing the Model

1. Run the stream to create the model nuggets, which are added to the stream and to the Models palette in the upper-right corner. To view their details, double-click on any of the model nuggets in the stream.

   The model nugget Model tab is split into two panes. The left pane contains a network graph of nodes that displays the relationship between the target and its most important predictors, as well as the relationship between the predictors.

   The right pane shows either *Predictor Importance*, which indicates the relative importance of each predictor in estimating the model, or *Conditional Probabilities*, which contains the conditional probability value for each node value and each combination of values in its parent nodes.

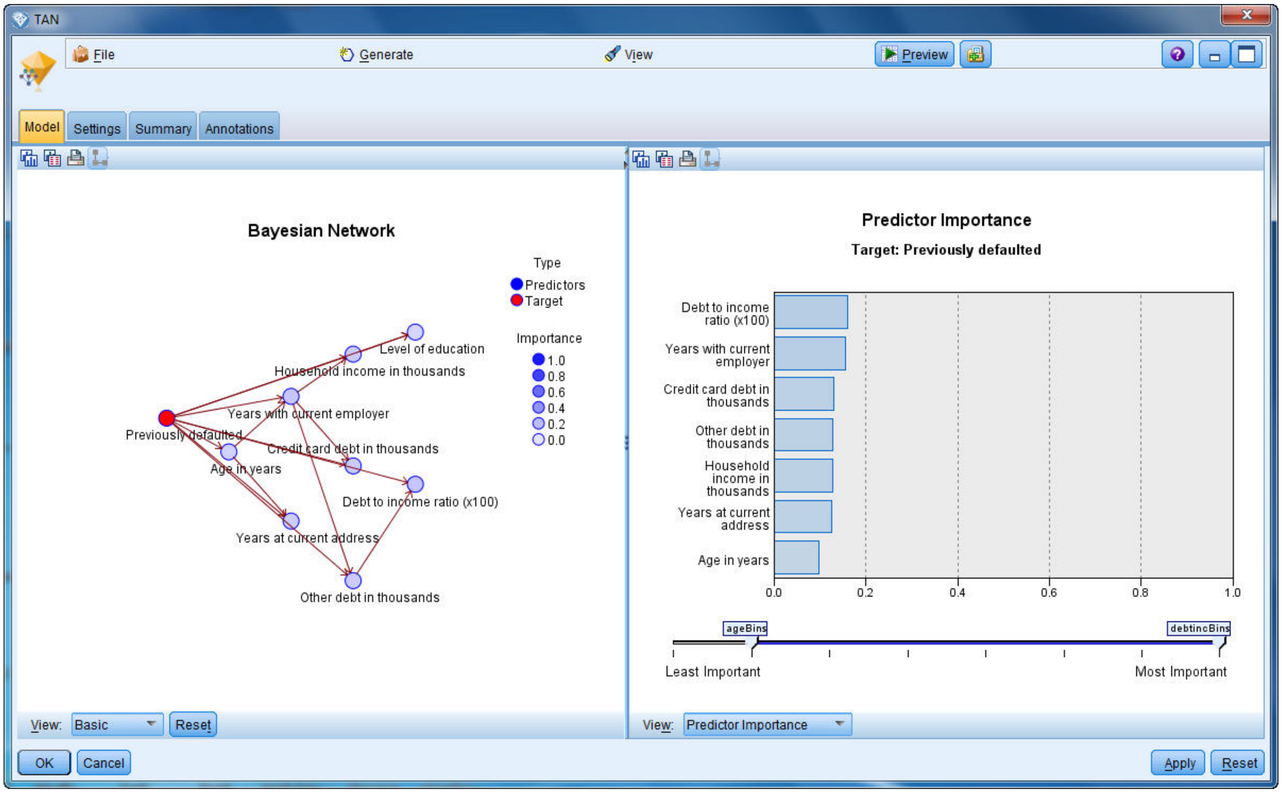


Figure 239. Viewing a Tree Augmented Naïve Bayes model

2. Connect the TAN model nugget to the Markov nugget (choose **Replace** on the warning dialog).
3. Connect the Markov nugget to the Markov-FS nugget (choose **Replace** on the warning dialog).
4. Align the three nuggets with the Select node for ease of viewing.

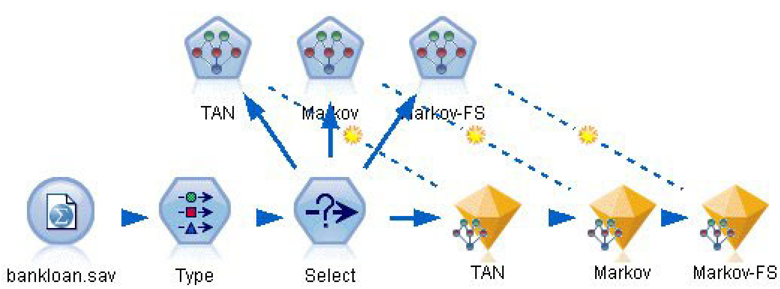


Figure 240. Aligning the nuggets in the stream

5. To rename the model outputs for clarity on the Evaluation graph that you'll be creating, attach a Filter node to the Markov-FS model nugget.
6. In the right *Field* column, rename $B-default as TAN, $B1-default as Markov, and $B2-default as Markov-FS.

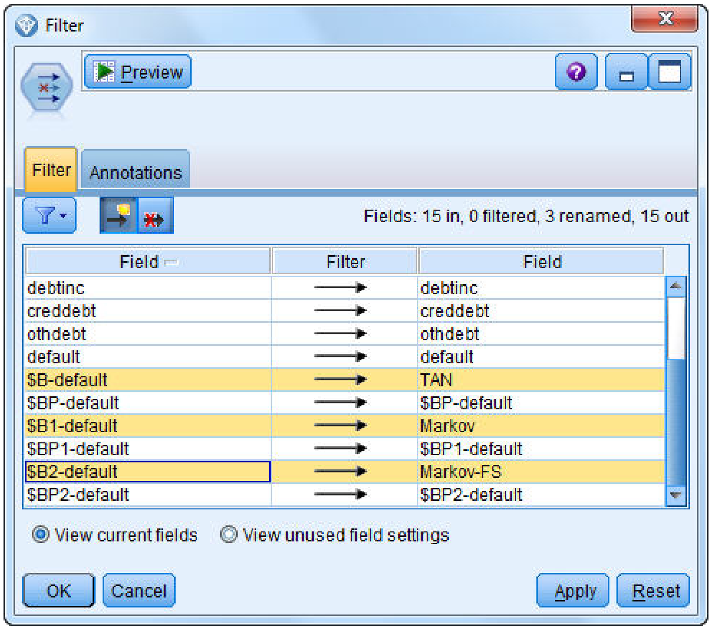


Figure 241. Rename model field names

To compare the models' predicted accuracy, you can build a gains chart.

7. Attach an Evaluation graph node to the Filter node and execute the graph node using its default settings.

   The graph shows that each model type produces similar results; however, the Markov model is slightly better.

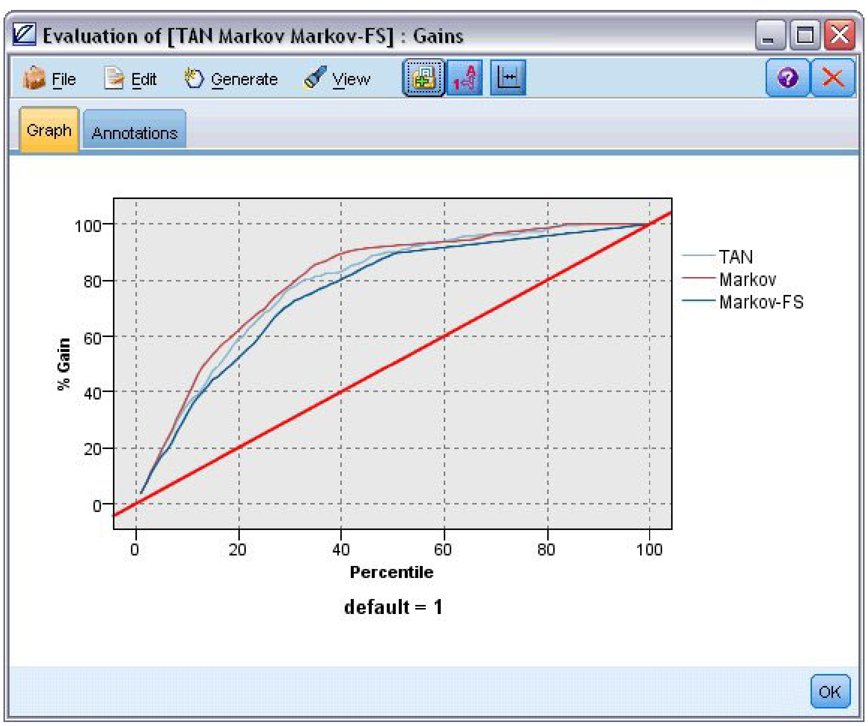


Figure 242. Evaluating model accuracy

To check how well each model predicts, you could use an Analysis node instead of the Evaluation graph. This shows the accuracy in terms of percentage for both correct and incorrect predictions.

8. Attach an Analysis node to the Filter node and execute the Analysis node using its default settings.

As with the Evaluation graph, this shows that the Markov model is slightly better at predicting correctly; however, the Markov-FS model is only a few percentage points behind the Markov model. This may mean it would be better to use the Markov-FS model since it uses fewer inputs to calculate its results, thereby saving on data collection and entry time and processing time.

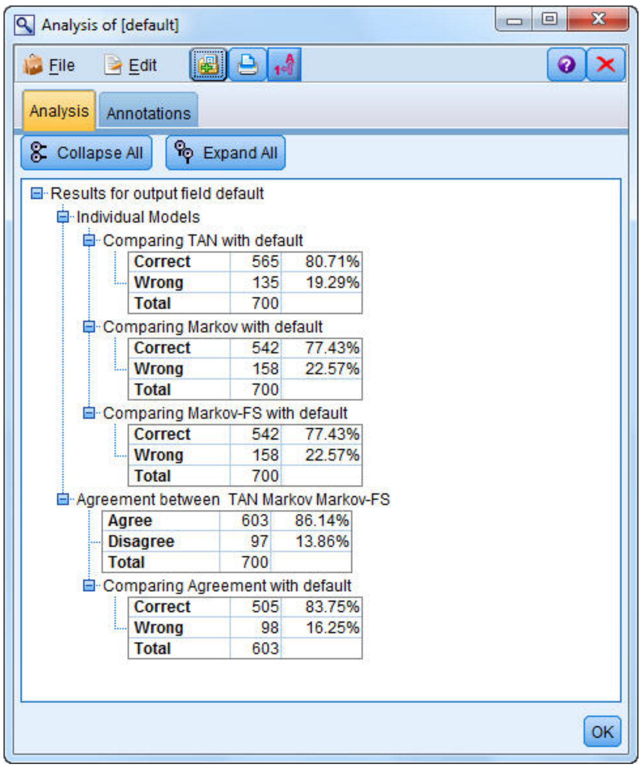


Figure 243. Analyzing model accuracy

Explanations of the mathematical foundations of the modeling methods used in IBM SPSS Modeler are listed in the *IBM SPSS Modeler Algorithms Guide*, available from the *Documentation* directory of the installation disk.

Note also that these results are based on the training data only. To assess how well the model generalizes to other data in the real world, you would use a Partition node to hold out a subset of records for purposes of testing and validation.

Chapter 18. Retraining a Model on a Monthly Basis (Bayesian Network)

Bayesian networks enable you to build a probability model by combining observed and recorded evidence with "common-sense" real-world knowledge to establish the likelihood of occurrences by using seemingly unlinked attributes.

This example uses the stream named *bayes_churn_retrain.str*, which references the data files named *telco_Jan.sav* and *telco_Feb.sav*. These files are available from the *Demos* directory of any IBM SPSS Modeler installation and can be accessed from the IBM SPSS Modeler program group on the Windows Start menu. The *bayes_churn_retrain.str* file is in the *streams* directory.

For example, suppose that a telecommunications provider is concerned about the number of customers it is losing to competitors (churn). If historic customer data can be used to predict which customers are more likely to churn in the future, these customers can be targeted with incentives or other offers to discourage them from transferring to another service provider.

This example focuses on using an existing month's churn data to predict which customers may be likely to churn in the future and then adding the following month's data to refine and retrain the model.

Building the Stream

1. Add a Statistics File source node pointing to *telco_Jan.sav* in the *Demos* folder.

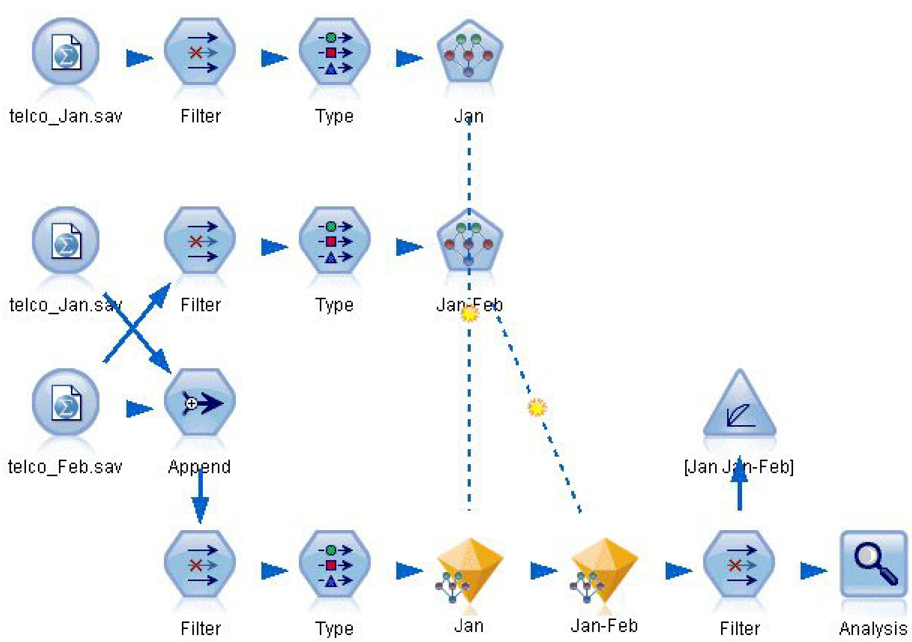


Figure 244. Bayesian Network sample stream

   Previous analysis has shown you that several data fields are of little importance when predicting churn. These fields can be filtered from your data set to increase the speed of processing when you are building and scoring models.
2. Add a Filter node to the Source node.
3. Exclude all fields except *address*, *age*, *churn*, *custcat*, *ed*, *employ*, *gender*, *marital*, *reside*, *retire*, and *tenure*.

4. Click **OK**.

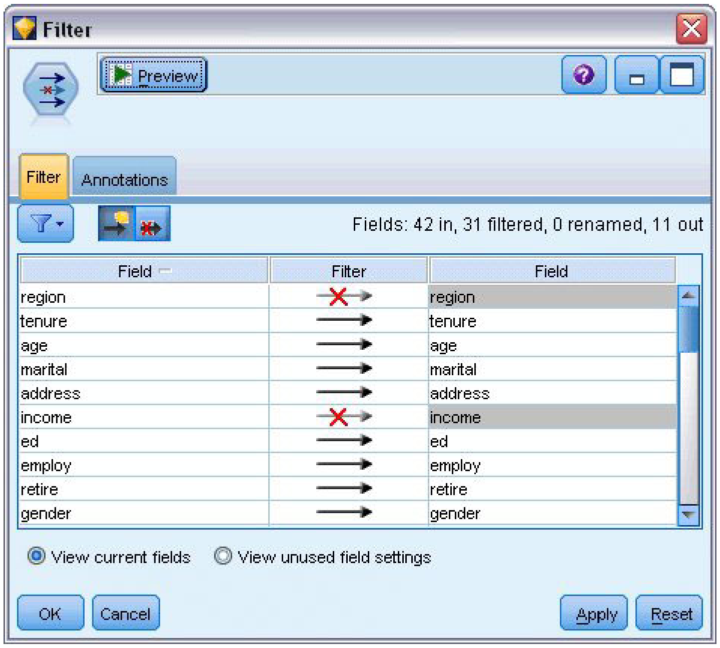


Figure 245. Filtering unnecessary fields

5. Add a Type node to the Filter node.
6. Open the Type node and click the **Read Values** button to populate the *Values* column.
7. In order that the Evaluation node can assess which value is true and which is false, set the measurement level for the *churn* field to **Flag**, and set its role to **Target**. Click **OK**.

Figure 246. Selecting the target field

You can build several different types of Bayesian networks; however, for this example you are going to build a Tree Augmented Naïve Bayes (TAN) model. This creates a large network and ensures that you have included all possible links between data variables, thereby building a robust initial model.

8. Attach a Bayesian Network node to the Type node.
9. On the Model tab, for Model name, select **Custom** and enter `Jan` in the text box.
10. For Parameter learning method, select **Bayes adjustment for small cell counts**.
11. Click **Run**. The model nugget is added to the stream, and also to the Models palette in the upper-right corner.

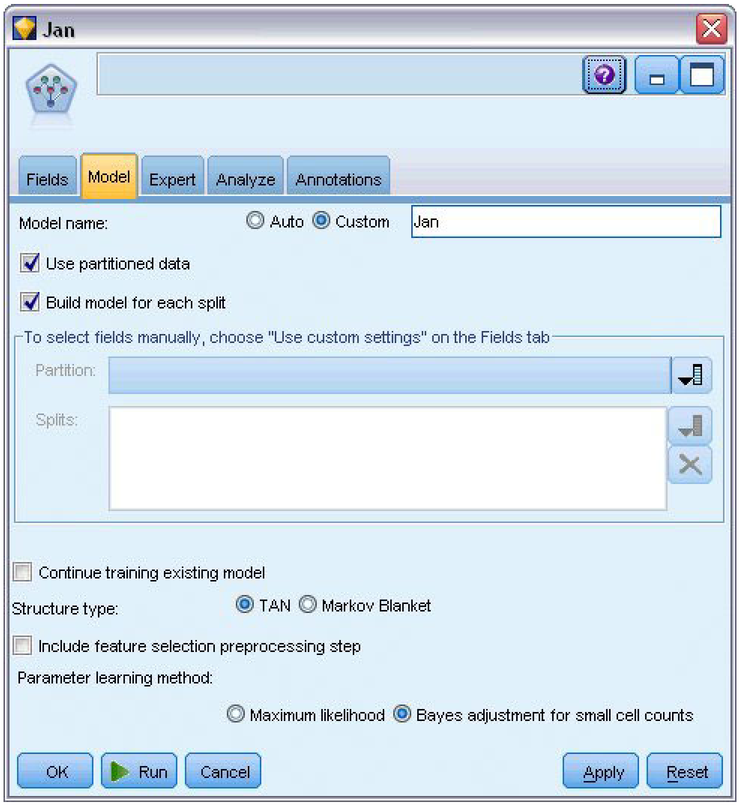


Figure 247. Creating a Tree Augmented Naïve Bayes model

12. Add a Statistics File source node pointing to *telco_Feb.sav* in the *Demos* folder.
13. Attach this new source node to the Filter node (on the warning dialog, choose **Replace** to replace the connection to the previous source node).

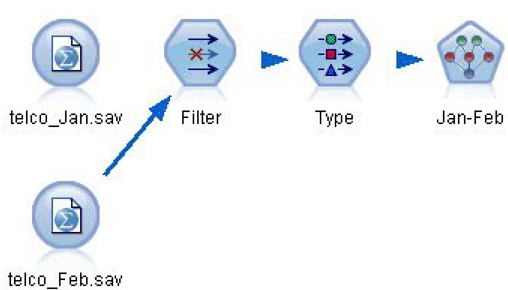


Figure 248. Adding the second month's data

14. On the Model tab of the Bayesian Network node, for Model name, select **Custom** and enter `Jan-Feb` in the text box.
15. Select **Continue training existing model**.

16. Click **Run**. The model nugget overwrites the existing one in the stream, but is also added to the Models palette in the upper-right corner.

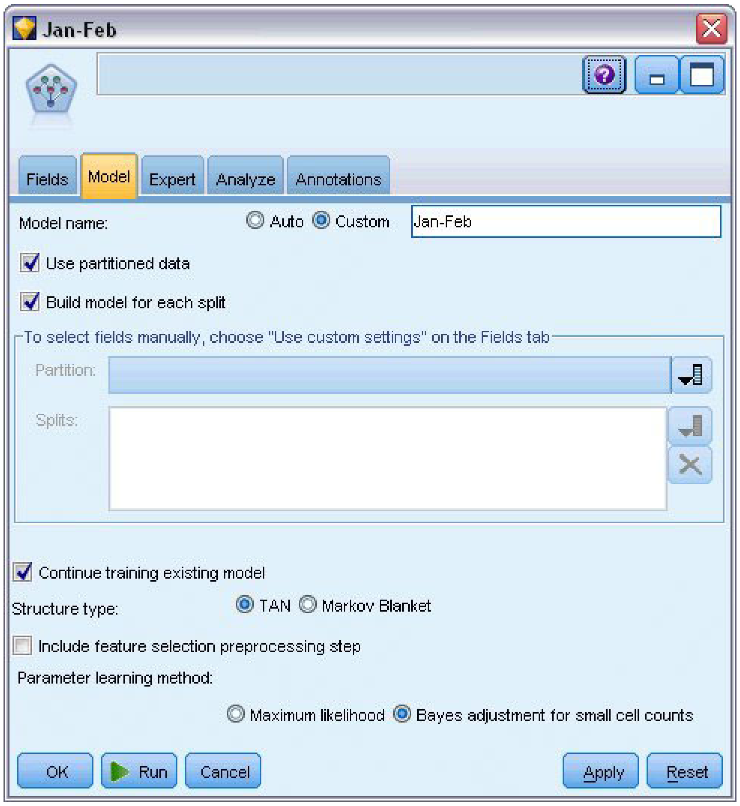


Figure 249. Retraining the model

Evaluating the Model

To compare the models, you must combine the two datasets.

1. Add an Append node and attach both the *telco_Jan.sav* and *telco_Feb.sav* source nodes to it.

Figure 250. Append the two data sources

2. Copy the Filter and Type nodes from earlier in the stream and paste them onto the stream canvas.
3. Attach the Append node to the newly copied Filter node.

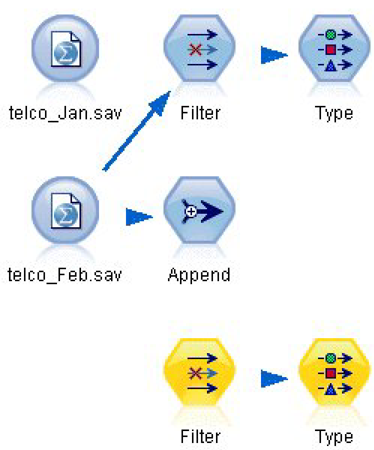


Figure 251. Pasting the copied nodes into the stream

The nuggets for the two Bayesian Network models are located in the Models palette in the upper-right corner.

4. Double-click the Jan model nugget to bring it into the stream, and attach it to the newly copied Type node.
5. Attach the Jan-Feb model nugget already in the stream to the Jan model nugget.
6. Open the Jan model nugget.

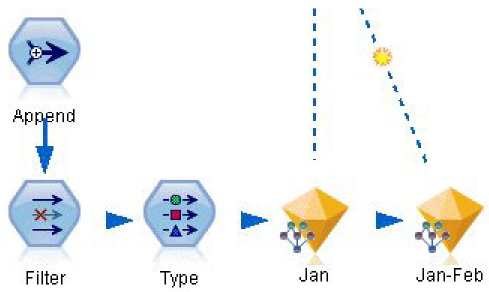


Figure 252. Adding the nuggets to the stream

The Bayesian Network model nugget Model tab is split into two columns. The left column contains a network graph of nodes that displays the relationship between the target and its most important predictors, as well as the relationship between the predictors.

The right column shows either *Predictor Importance*, which indicates the relative importance of each predictor in estimating the model, or *Conditional Probabilities*, which contains the conditional probability value for each node value and each combination of values in its parent nodes.

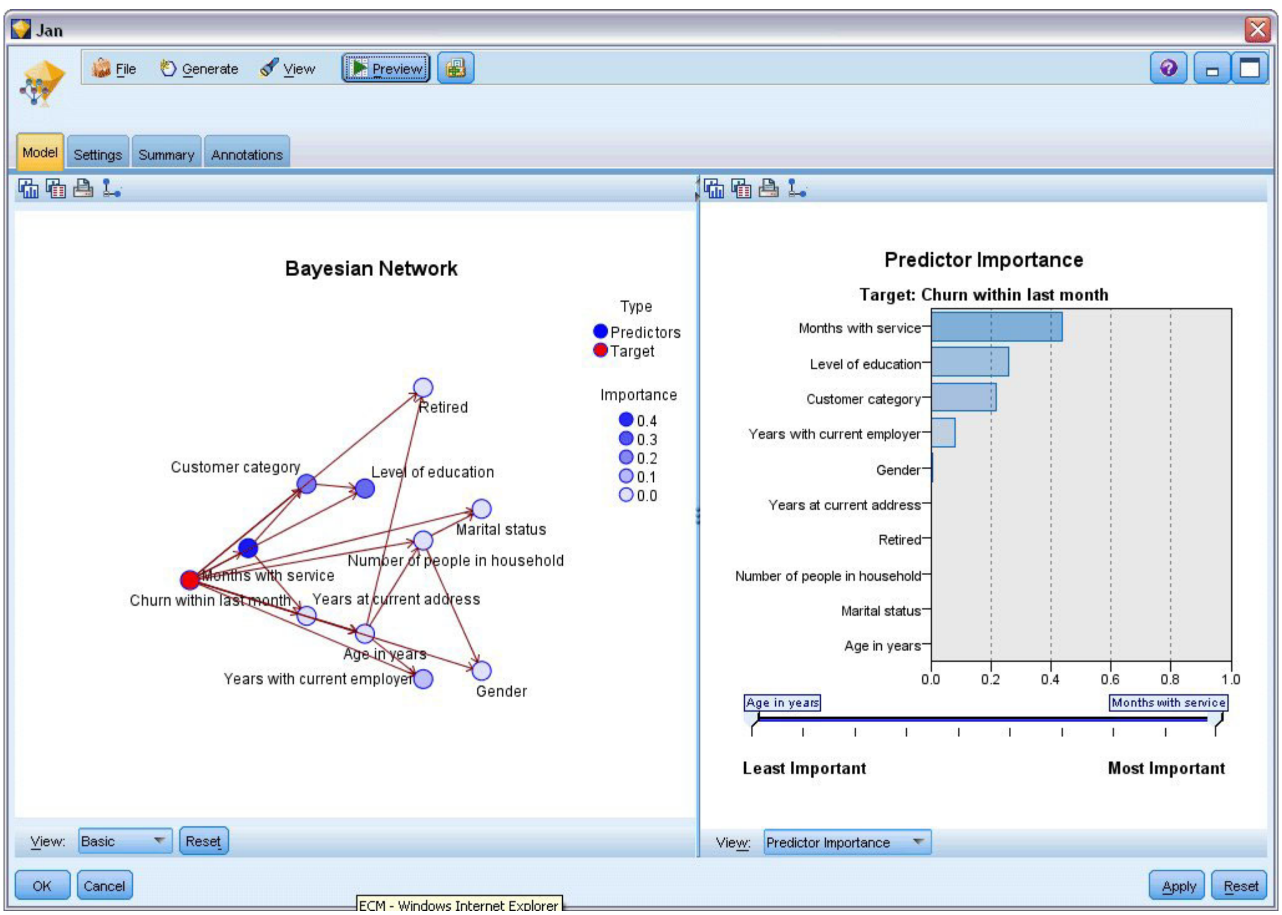


Figure 253. Bayesian Network model showing predictor importance

To display the conditional probabilities for any node, click on the node in the left column. The right column is updated to show the required details.

The conditional probabilities are shown for each bin that the data values have been divided into relative to the node's parent and sibling nodes.

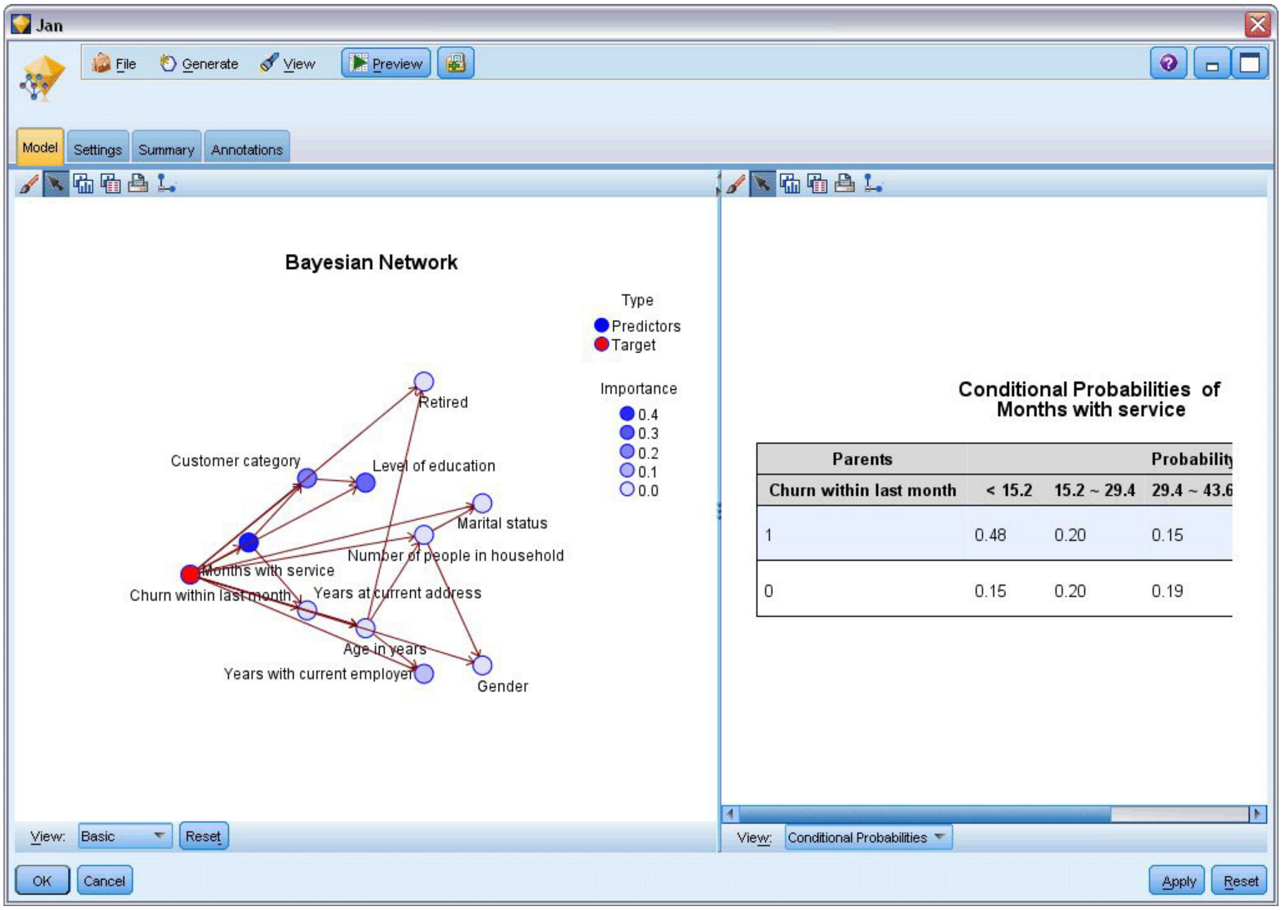


Figure 254. Bayesian Network model showing conditional probabilities

7. To rename the model outputs for clarity, attach a Filter node to the Jan-Feb model nugget.
8. In the right *Field* column, rename $B-churn as Jan and $B1-churn as Jan-Feb.

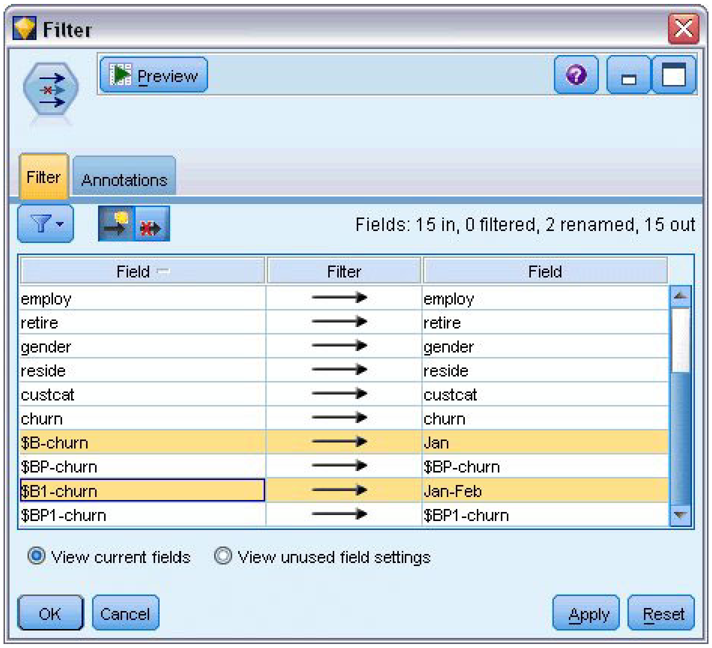


Figure 255. Rename model field names

To check how well each model predicts churn, use an Analysis node; this shows the accuracy in terms of percentage for both correct and incorrect predictions.

9. Attach an Analysis node to the Filter node.
10. Open the Analysis node and click **Run**.

    This shows that both models have a similar degree of accuracy when predicting churn.

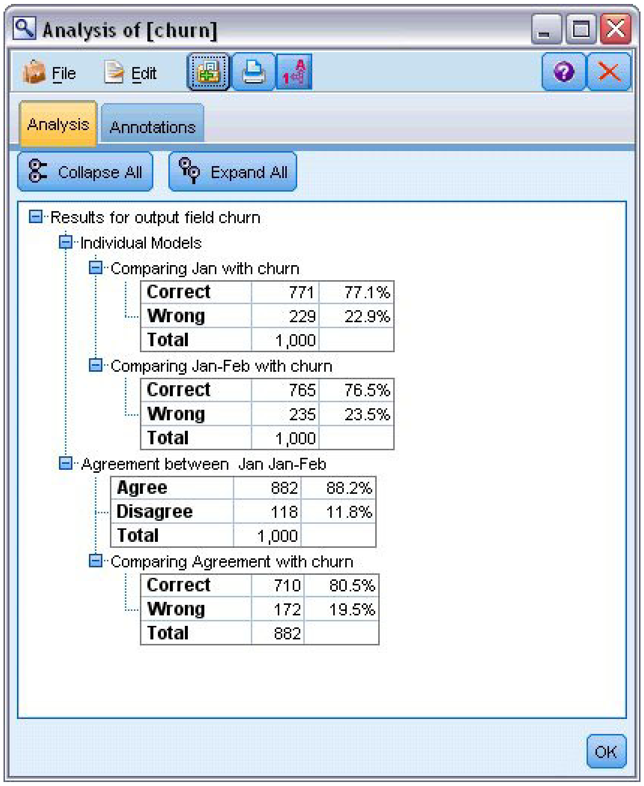


Figure 256. Analyzing model accuracy

As an alternative to the Analysis node, you can use an Evaluation graph to compare the models' predicted accuracy by building a gains chart.

11. Attach an Evaluation graph node to the Filter node.

and execute the graph node using its default settings.

As with the Analysis node, the graph shows that each model type produces similar results; however, the retrained model using both months' data is slightly better because it has a higher level of confidence in its predictions.

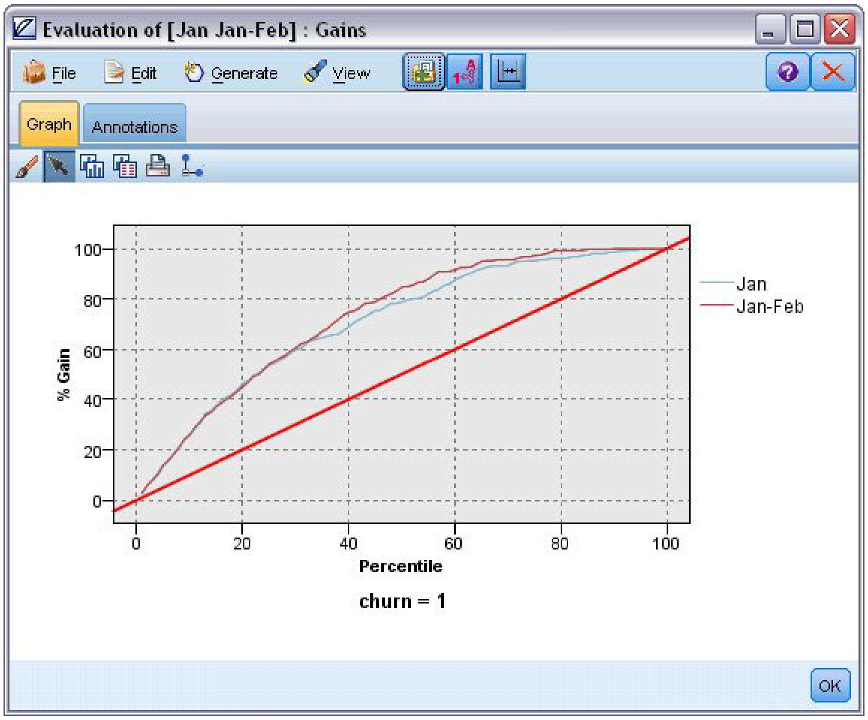


Figure 257. Evaluating model accuracy

Explanations of the mathematical foundations of the modeling methods used in IBM SPSS Modeler are listed in the *IBM SPSS Modeler Algorithms Guide*, available from the *\Documentation* directory of the installation disk.

Note also that these results are based on the training data only. To assess how well the model generalizes to other data in the real world, you would use a Partition node to hold out a subset of records for purposes of testing and validation.

Chapter 19. Retail Sales Promotion (Neural Net/C&RT)

This example deals with data that describes retail product lines and the effects of promotion on sales. (This data is fictitious.) Your goal in this example is to predict the effects of future sales promotions. Similar to the condition monitoring example, the data mining process consists of the exploration, data preparation, training, and test phases.

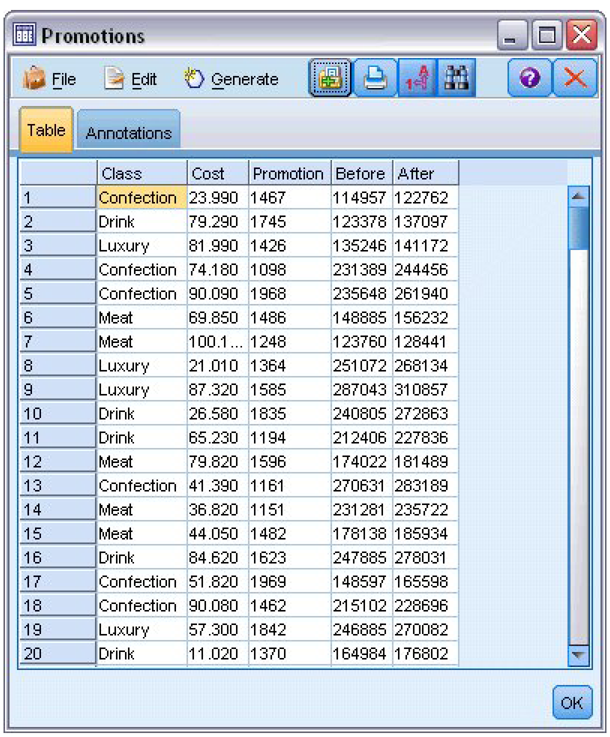
This example uses the streams named *goodsplot.str* and *goodslearn.str*, which reference the data files named *GOODS1n* and *GOODS2n*. These files are available from the *Demos* directory of any IBM SPSS Modeler installation. This can be accessed from the IBM SPSS Modeler program group on the Windows Start menu. The stream *goodsplot.str* is in the *streams* folder, while the *goodslearn.str* file is in the *streams* directory.

Examining the Data

Each record contains:

- *Class*. Product type.
- *Cost*. Unit price.
- *Promotion*. Index of amount spent on a particular promotion.
- *Before*. Revenue before promotion.
- *After*. Revenue after promotion.

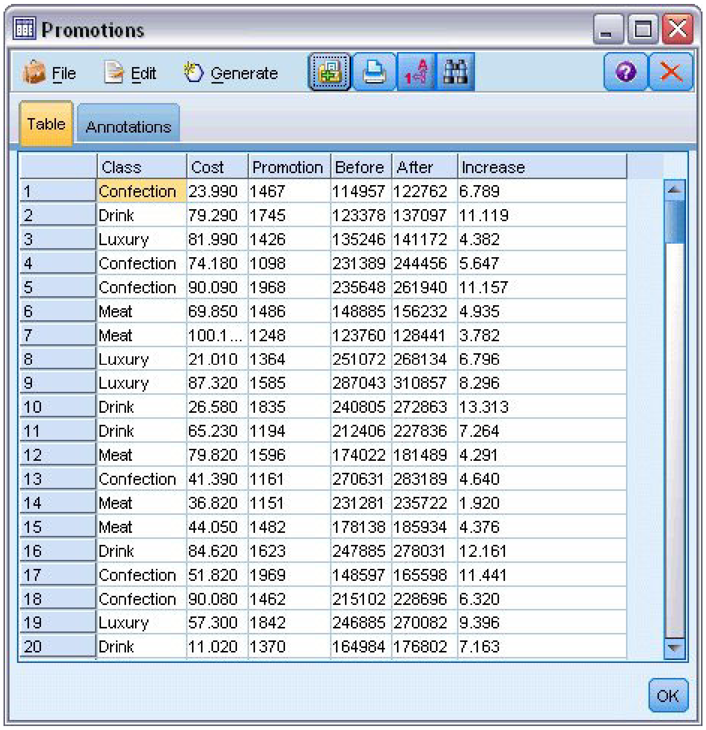
The stream *goodsplot.str* contains a simple stream to display the data in a table. The two revenue fields (*Before* and *After*) are expressed in absolute terms; however, it seems likely that the increase in revenue after the promotion (and presumably as a result of it) would be a more useful figure.



	Class	Cost	Promotion	Before	After
1	Confection	23.990	1467	114957	122762
2	Drink	79.290	1745	123378	137097
3	Luxury	81.990	1426	135246	141172
4	Confection	74.180	1098	231389	244456
5	Confection	90.090	1968	235648	261940
6	Meat	69.850	1486	148885	156232
7	Meat	100.1...	1248	123760	128441
8	Luxury	21.010	1364	251072	268134
9	Luxury	87.320	1585	287043	310857
10	Drink	26.580	1835	240805	272863
11	Drink	65.230	1194	212406	227836
12	Meat	79.820	1596	174022	181489
13	Confection	41.390	1161	270631	283189
14	Meat	36.820	1151	231281	235722
15	Meat	44.050	1482	178138	185934
16	Drink	84.620	1623	247885	278031
17	Confection	51.820	1969	148597	165598
18	Confection	90.080	1462	215102	228696
19	Luxury	57.300	1842	246885	270082
20	Drink	11.020	1370	164984	176802

Figure 258. Effects of promotion on product sales

goodsplot.str also contains a node to derive this value, expressed as a percentage of the revenue before the promotion, in a field called *Increase* and displays a table showing this field.



	Class	Cost	Promotion	Before	After	Increase
1	Confection	23.990	1467	114957	122762	6.789
2	Drink	79.290	1745	123378	137097	11.119
3	Luxury	81.990	1426	135246	141172	4.382
4	Confection	74.180	1098	231389	244456	5.647
5	Confection	90.090	1968	235648	261940	11.157
6	Meat	69.850	1486	148885	156232	4.935
7	Meat	100.1...	1248	123760	128441	3.782
8	Luxury	21.010	1364	251072	268134	6.796
9	Luxury	87.320	1585	287043	310857	8.296
10	Drink	26.580	1835	240805	272863	13.313
11	Drink	65.230	1194	212406	227836	7.264
12	Meat	79.820	1596	174022	181489	4.291
13	Confection	41.390	1161	270631	283189	4.640
14	Meat	36.820	1151	231281	235722	1.920
15	Meat	44.050	1482	178138	185934	4.376
16	Drink	84.620	1623	247885	278031	12.161
17	Confection	51.820	1969	148597	165598	11.441
18	Confection	90.080	1462	215102	228696	6.320
19	Luxury	57.300	1842	246885	270082	9.396
20	Drink	11.020	1370	164984	176802	7.163

Figure 259. Increase in revenue after promotion

In addition, the stream displays a histogram of the increase and a scatterplot of the increase against the promotion costs expended, overlaid with the category of product involved.

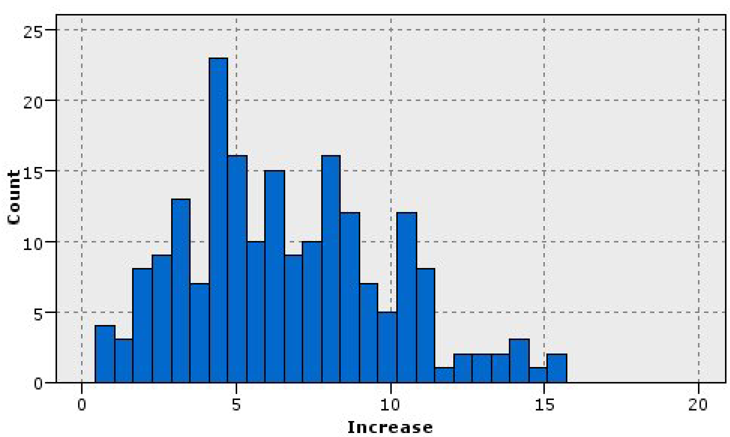


Figure 260. Histogram of increase in revenue

The scatterplot shows that for each class of product, an almost linear relationship exists between the increase in revenue and the cost of promotion. Therefore, it seems likely that a decision tree or neural network could predict, with reasonable accuracy, the increase in revenue from the other available fields.

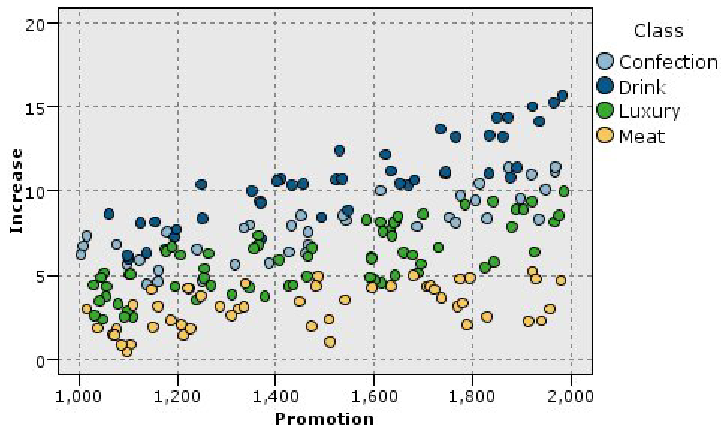


Figure 261. Revenue increase versus promotional expenditure

Learning and Testing

The stream *goodslearn.str* trains a neural network and a decision tree to make this prediction of revenue increase.

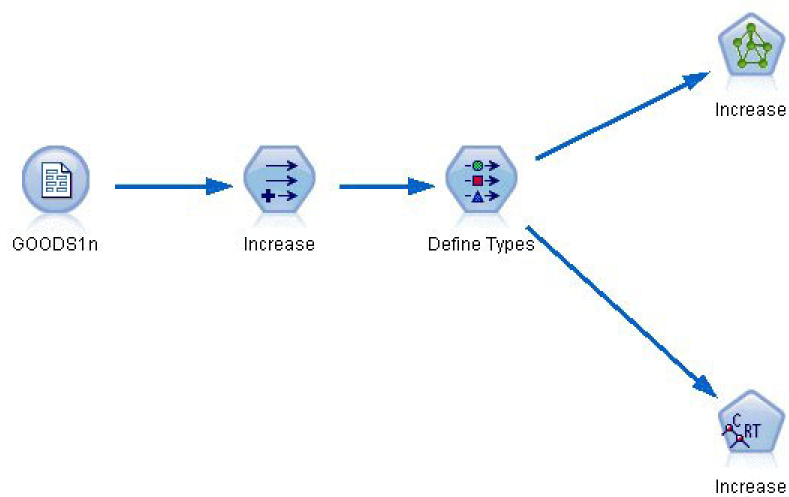


Figure 262. Modeling stream goodslearn.str

Once you have executed the model nodes and generated the actual models, you can test the results of the learning process. You do this by connecting the decision tree and network in series between the Type node and a new Analysis node, changing the input (data) file to *GOODS2n*, and executing the Analysis node. From the output of this node, in particular from the linear correlation between the predicted increase and the correct answer, you will find that the trained systems predict the increase in revenue with a high degree of success.

Further exploration could focus on the cases where the trained systems make relatively large errors; these could be identified by plotting the predicted increase in revenue against the actual increase. Outliers on this graph could be selected using IBM SPSS Modeler's interactive graphics, and from their properties, it might be possible to tune the data description or learning process to improve accuracy.

Chapter 20. Condition Monitoring (Neural Net/C5.0)

This example concerns monitoring status information from a machine and the problem of recognizing and predicting fault states. The data is created from a fictitious simulation and consists of a number of concatenated series measured over time. Each record is a snapshot report on the machine in terms of the following:

- *Time*. An integer.
- *Power*. An integer.
- *Temperature*. An integer.
- *Pressure*. 0 if normal, 1 for a momentary pressure warning.
- *Uptime*. Time since last serviced.
- *Status*. Normally 0, changes to error code on error (101, 202, or 303).
- *Outcome*. The error code that appears in this time series, or 0 if no error occurs. (These codes are available only with the benefit of hindsight.)

This example uses the streams named *condplot.str* and *condlearn.str*, which reference the data files named *COND1n* and *COND2n*. These files are available from the *Demos* directory of any IBM SPSS Modeler installation. This can be accessed from the IBM SPSS Modeler program group on the Windows Start menu. The *condplot.str* and *condlearn.str* files are in the *streams* directory.

For each time series, there is a series of records from a period of normal operation followed by a period leading to the fault, as shown in the following table:

Time	Power	Temperature	Pressure	Uptime	Status	Outcome
0	1059	259	0	404	0	0
1	1059	259	0	404	0	0
			...			
51	1059	259	0	404	0	0
52	1059	259	0	404	0	0
53	1007	259	0	404	0	303
54	998	259	0	404	0	303
			...			
89	839	259	0	404	0	303
90	834	259	0	404	303	303
0	965	251	0	209	0	0
1	965	251	0	209	0	0
			...			
51	965	251	0	209	0	0
52	965	251	0	209	0	0
53	938	251	0	209	0	101
54	936	251	0	209	0	101
			...			
208	644	251	0	209	0	101
209	640	251	0	209	101	101

The following process is common to most data mining projects:

- Examine the data to determine which attributes may be relevant to the prediction or recognition of the states of interest.
- Retain those attributes (if already present), or derive and add them to the data, if necessary.
- Use the resultant data to train rules and neural nets.
- Test the trained systems using independent test data.

Examining the Data

The file *condplot.str* illustrates the first part of the process. It contains a stream that plots a number of graphs. If the time series of temperature or power contains visible patterns, you could differentiate between impending error conditions or possibly predict their occurrence. For both temperature and power, the stream below plots the time series associated with the three different error codes on separate graphs, yielding six graphs. Select nodes separate the data associated with the different error codes.

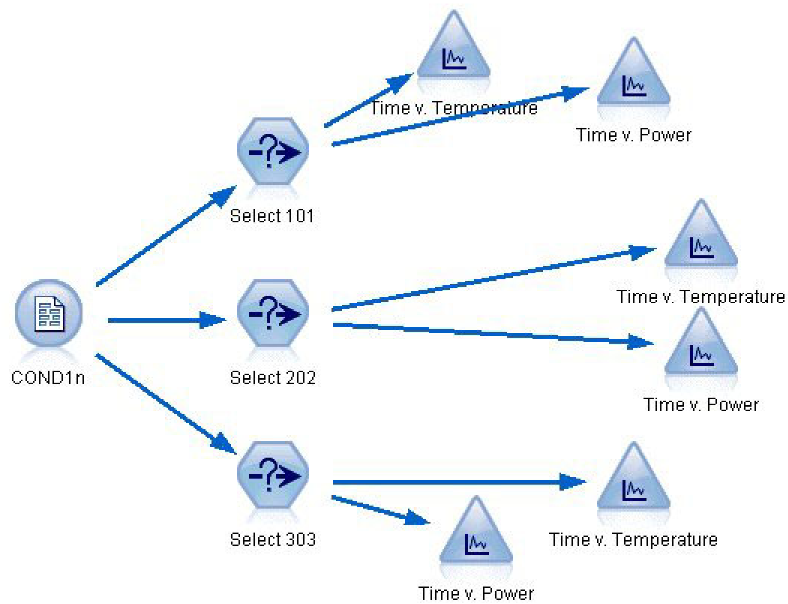


Figure 263. Condplot stream

The results of this stream are shown in this figure.

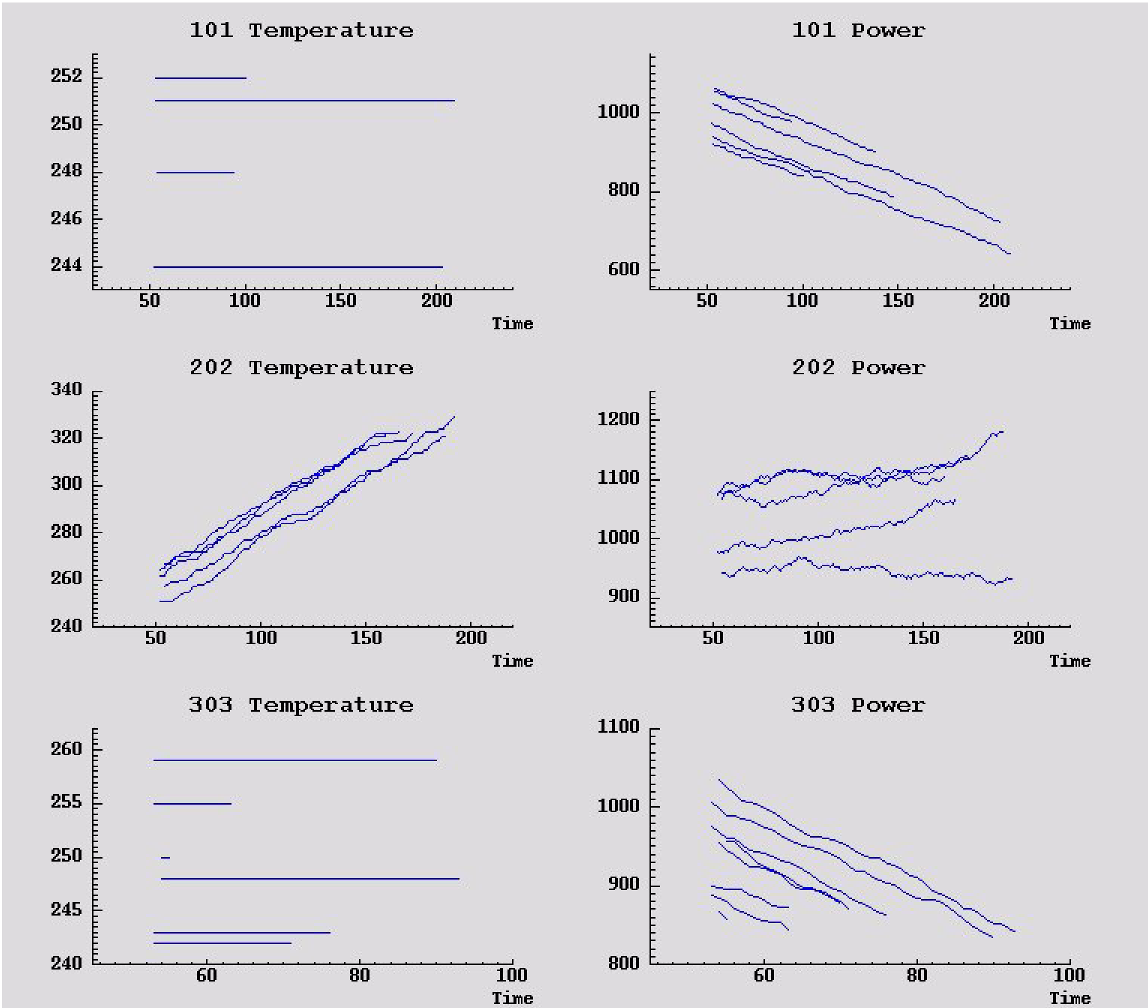


Figure 264. Temperature and power over time

The graphs clearly display patterns distinguishing 202 errors from 101 and 303 errors. The 202 errors show rising temperature and fluctuating power over time; the other errors do not. However, patterns distinguishing 101 from 303 errors are less clear. Both errors show even temperature and a drop in power, but the drop in power seems steeper for 303 errors.

Based on these graphs, it appears that the presence and rate of change for both temperature and power, as well as the presence and degree of fluctuation, are relevant to predicting and distinguishing faults. These attributes should therefore be added to the data before applying the learning systems.

Data Preparation

Based on the results of exploring the data, the stream *condlearn.str* derives the relevant data and learns to predict faults.

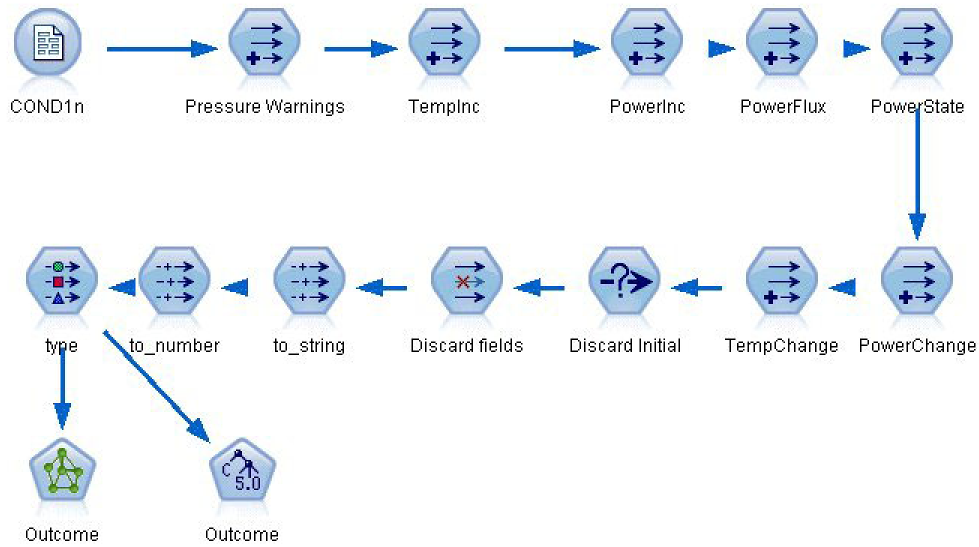


Figure 265. Condlearn stream

The stream uses a number of Derive nodes to prepare the data for modeling.

- **Variable File node**. Reads data file *COND1n*.
- **Derive Pressure Warnings**. Counts the number of momentary pressure warnings. Reset when time returns to 0.
- **Derive TempInc**. Calculates momentary rate of temperature change using `@DIFF1`.
- **Derive PowerInc**. Calculates momentary rate of power change using `@DIFF1`.
- **Derive PowerFlux**. A flag, true if power varied in opposite directions in the last record and this one; that is, for a power peak or trough.
- **Derive PowerState**. A state that starts as *Stable* and switches to *Fluctuating* when two successive power fluxes are detected. Switches back to *Stable* only when there hasn't been a power flux for five time intervals or when *Time* is reset.
- **PowerChange**. Average of *PowerInc* over the last five time intervals.
- **TempChange**. Average of *TempInc* over the last five time intervals.
- **Discard Initial (select)**. Discards the first record of each time series to avoid large (incorrect) jumps in *Power* and *Temperature* at boundaries.
- **Discard fields**. Cuts records down to *Uptime*, *Status*, *Outcome*, *Pressure Warnings*, *PowerState*, *PowerChange*, and *TempChange*.
- **Type**. Defines the role of *Outcome* as **Target** (the field to predict). In addition, defines the measurement level of *Outcome* as **Nominal**, *Pressure Warnings* as **Continuous**, and *PowerState* as **Flag**.

Learning

Running the stream in *condlearn.str* trains the C5.0 rule and neural network (net). The network may take some time to train, but training can be interrupted early to save a net that produces reasonable results. Once the learning is complete, the Models tab at the upper right of the managers window flashes to alert you that two new nuggets were created: one represents the neural net and one represents the rule.

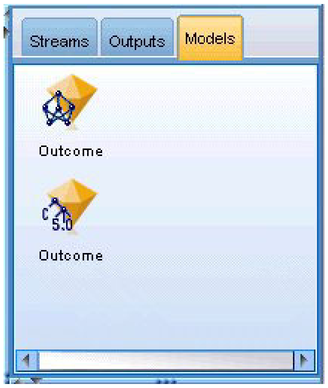


Figure 266. Models manager with model nuggets

The model nuggets are also added to the existing stream, enabling us to test the system or export the results of the model. In this example, we will test the results of the model.

Testing

The model nuggets are added to the stream, both of them connected to the Type node.

1. Reposition the nuggets as shown, so that the Type node connects to the neural net nugget, which connects to the C5.0 nugget.
2. Attach an Analysis node to the C5.0 nugget.
3. Edit the original source node to read the file *COND2n* (instead of *COND1n*), as *COND2n* contains unseen test data.

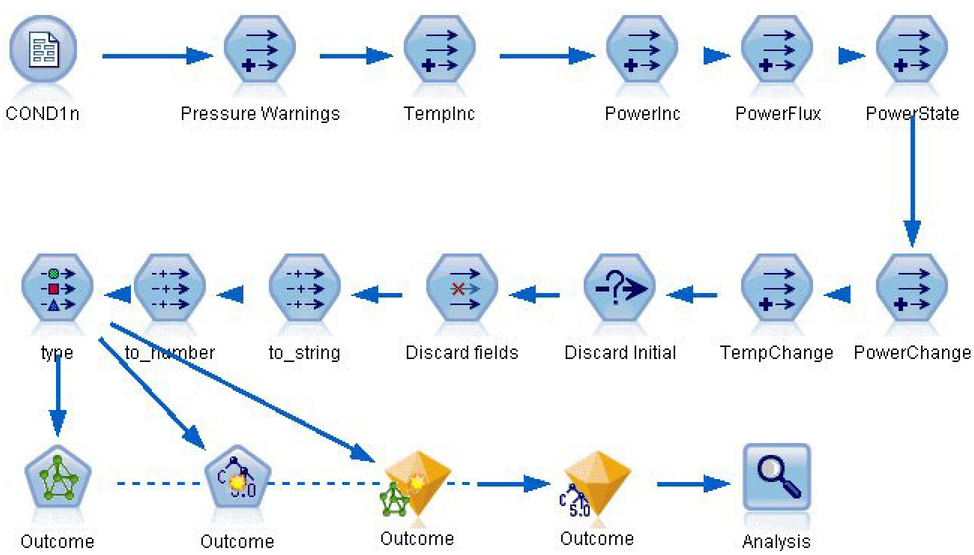


Figure 267. Testing the trained network

4. Open the Analysis node and click Run.

Doing so yields figures reflecting the accuracy of the trained network and rule.

Chapter 21. Classifying Telecommunications Customers (Discriminant Analysis)

Discriminant analysis is a statistical technique for classifying records based on values of input fields. It is analogous to linear regression but takes a categorical target field instead of a numeric one.

For example, suppose a telecommunications provider has segmented its customer base by service usage patterns, categorizing the customers into four groups. If demographic data can be used to predict group membership, you can customize offers for individual prospective customers.

This example uses the stream named *telco_custcat_discriminant.str*, which references the data file named *telco.sav*. These files are available from the *Demos* directory of any IBM SPSS Modeler installation. This can be accessed from the IBM SPSS Modeler program group on the Windows Start menu. The *telco_custcat_discriminant.str* file is in the *streams* directory.

The example focuses on using demographic data to predict usage patterns. The target field *custcat* has four possible values which correspond to the four customer groups, as follows:

Value	Label
1	Basic Service
2	E-Service
3	Plus Service
4	Total Service

Creating the Stream

1. First, set the stream properties to show variable and value labels in the output. From the menus, choose:

   File > **Stream Properties...** > **Options** > **General**
2. Make sure that **Display field and value labels in output** is selected and click **OK**.

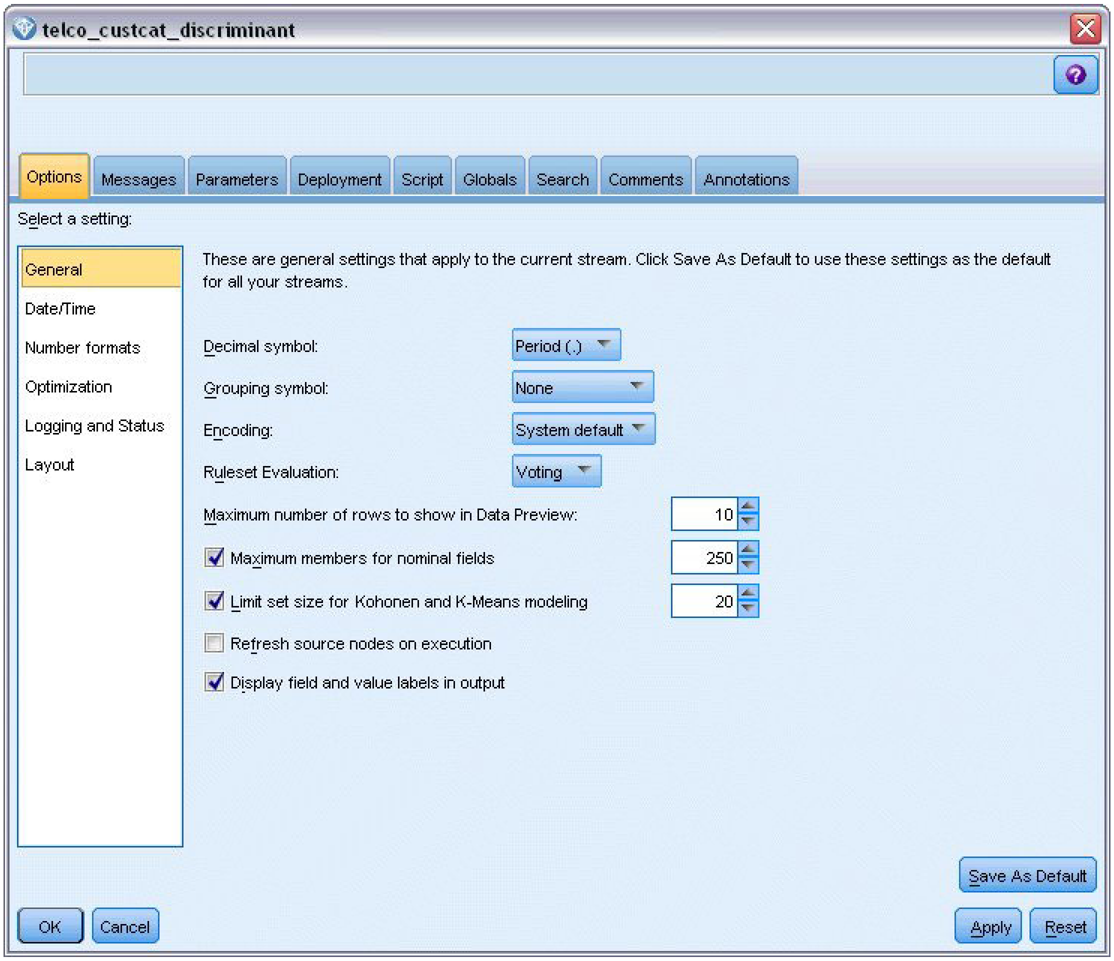


Figure 268. Stream properties

3. Add a Statistics File source node pointing to *telco.sav* in the *Demos* folder.

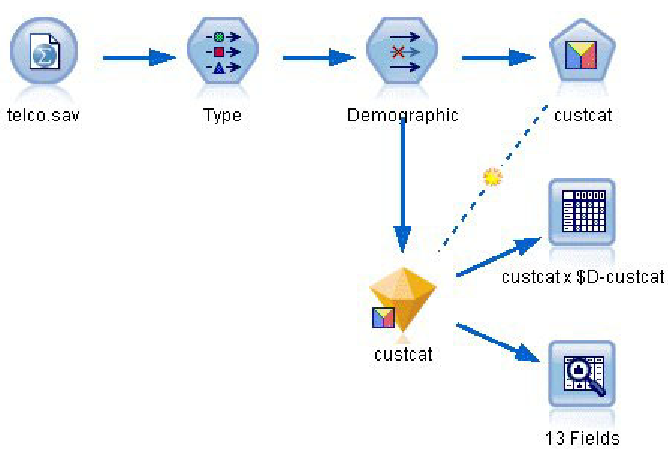


Figure 269. Sample stream to classify customers using discriminant analysis

a. Add a Type node and click **Read Values**, making sure that all measurement levels are set correctly. For example, most fields with values 0 and 1 can be regarded as flags.

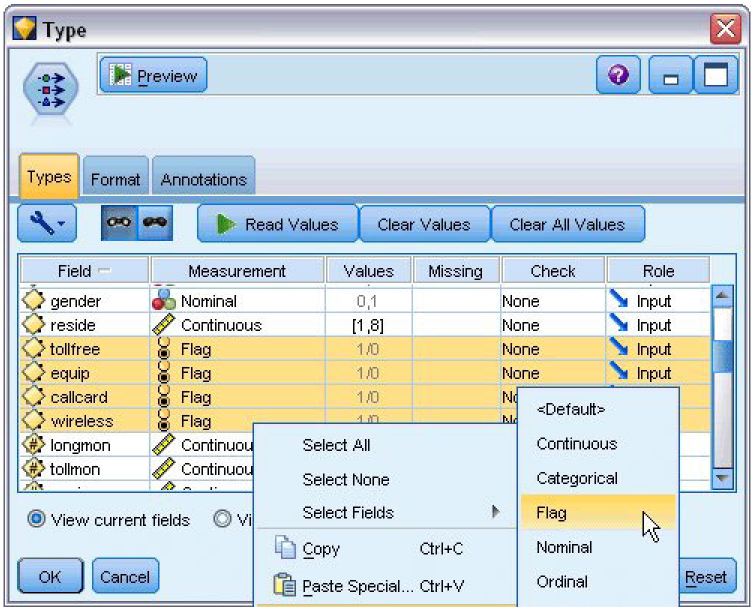


Figure 270. Setting the measurement level for multiple fields

Tip: To change properties for multiple fields with similar values (such as 0/1), click the *Values* column header to sort fields by value, and then hold down the shift key while using the mouse or arrow keys to select all the fields you want to change. You can then right-click on the selection to change the measurement level or other attributes of the selected fields.

Notice that *gender* is more correctly considered as a field with a set of two values, instead of a flag, so leave its Measurement value as **Nominal**.

b. Set the role for the *custcat* field to **Target**. All other fields should have their role set to **Input**.

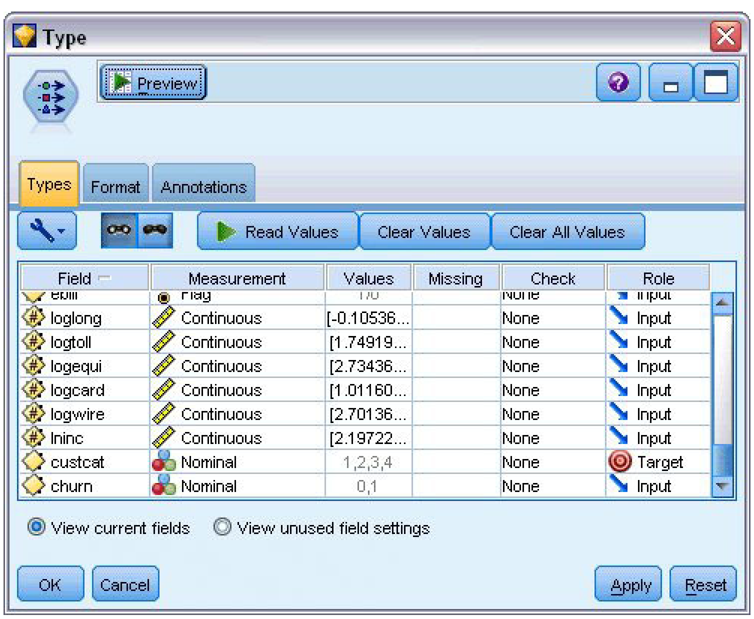


Figure 271. Setting field role

Since this example focuses on demographics, use a Filter node to include only the relevant fields (*region*, *age*, *marital*, *address*, *income*, *ed*, *employ*, *retire*, *gender*, *reside*, and *custcat*). Other fields can be excluded for the purpose of this analysis.

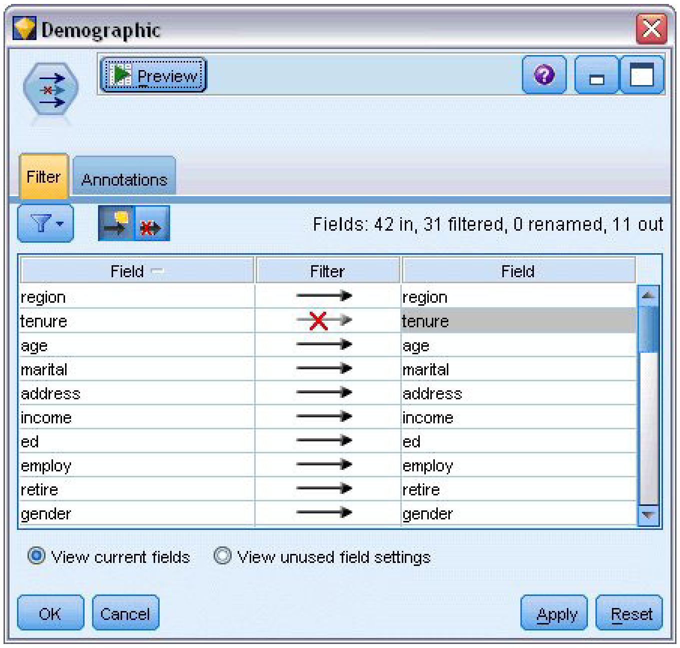


Figure 272. Filtering on demographic fields

(Alternatively, you could change the role to **None** for these fields rather than exclude them, or select the fields you want to use in the modeling node.)

4. In the Discriminant node, click the Model tab and select the **Stepwise** method.

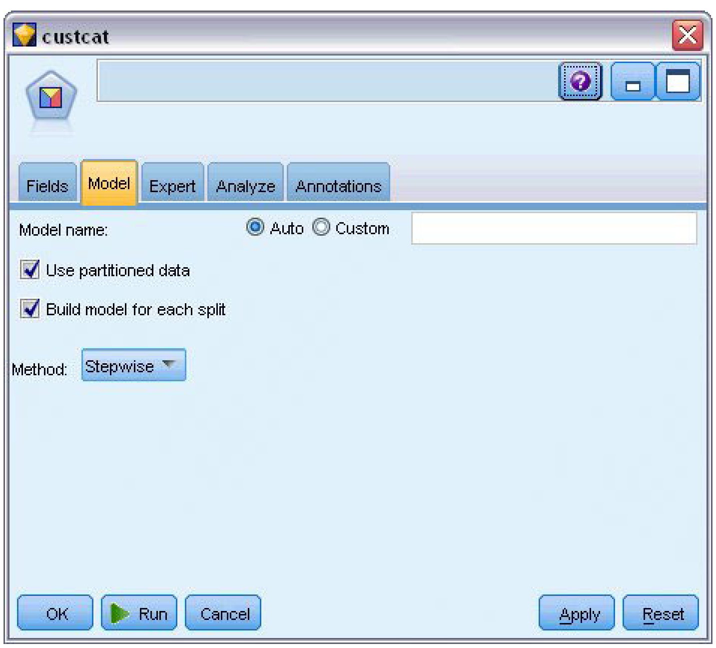


Figure 273. Choosing model options

5. On the Expert tab, set the mode to **Expert** and click **Output**.
6. Select **Summary table**, **Territorial map**, and **Summary of Steps** in the Advanced Output dialog box, then click **OK**.

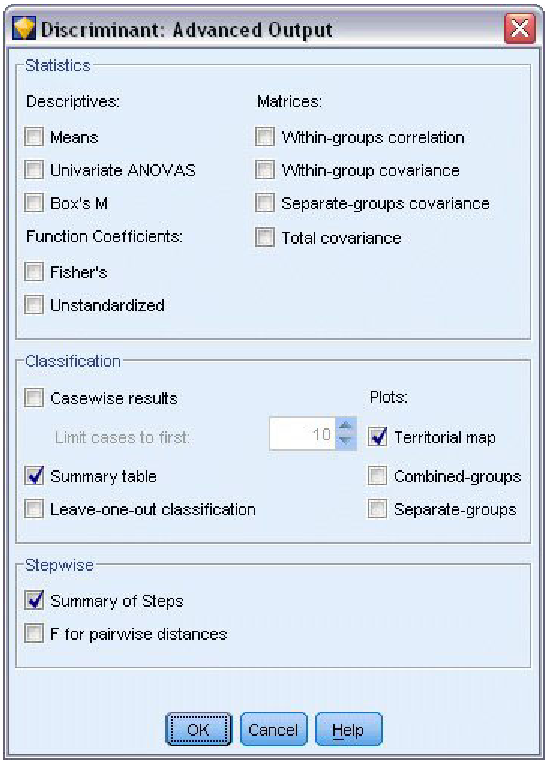


Figure 274. Choosing output options

Examining the Model

1. Click **Run** to create the model, which is added to the stream and to the Models palette in the upper-right corner. To view its details, double-click on the model nugget in the stream.

   The Summary tab shows (among other things) the target and the complete list of inputs (predictor fields) submitted for consideration.

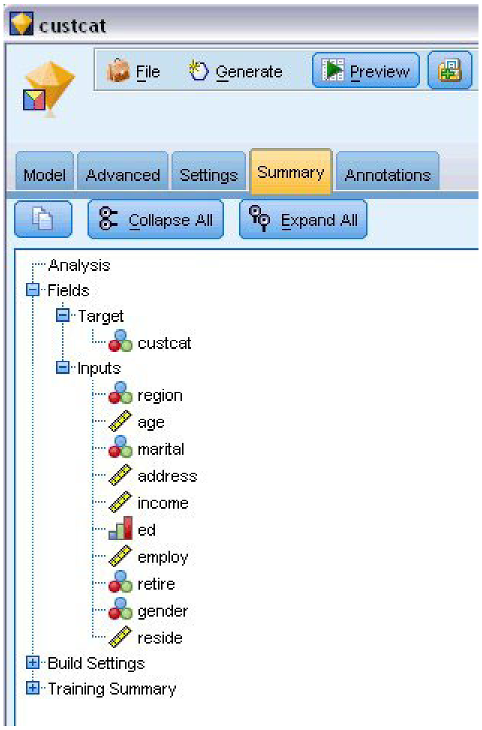


Figure 275. Model summary showing target and input fields

For details of the discriminant analysis results:

2. Click the Advanced tab.
3. Click the "Launch in external browser" button (just below the Model tab) to view the results in your Web browser.

Analyzing Output of Using Discriminant Analysis to Classify Telecommunications Customers

Stepwise Discriminant Analysis

Step		Tolerance	Min. Tolerance	F to Enter	Wilks' Lambda
0	Age in years	1.000	1.000	7.521	.978
	Marital status	1.000	1.000	3.500	.990
	Years at current address	1.000	1.000	8.433	.975
	Household income in thousands	1.000	1.000	6.689	.980
	Level of education	1.000	1.000	61.454	.844
	Retired	1.000	1.000	3.005	.991
	Years with current employer	1.000	1.000	16.976	.951
	Gender	1.000	1.000	.373	.999
	Number of people in household	1.000	1.000	3.976	.988

Figure 276. Variables not in the analysis, step 0

When you have a lot of predictors, the stepwise method can be useful by automatically selecting the "best" variables to use in the model. The stepwise method starts with a model that doesn't include any of the predictors. At each step, the predictor with the largest *F to Enter* value that exceeds the entry criteria (by default, 3.84) is added to the model.

Step		Tolerance	Min. Tolerance	F to Enter	Wilks' Lambda
3	Age in years	.535	.535	.252	.795
	Marital status	.605	.593	1.507	.792
	Years at current address	.776	.771	3.514	.787
	Household income in thousands	.688	.657	.687	.794
	Retired	.917	.880	.353	.795
	Gender	.997	.931	.395	.795

Figure 277. Variables not in the analysis, step 3

The variables left out of the analysis at the last step all have *F to Enter* values smaller than 3.84, so no more are added.

Step		Tolerance	F to Remove	Wilks' Lambda
1	Level of education	1.000	61.454	
2	Level of education	.953	59.108	.951
	Years with current employer	.953	14.933	.844
3	Level of education	.951	60.046	.940
	Years with current employer	.934	15.824	.834
	Number of people in household	.979	4.841	.807

Figure 278. Variables in the analysis

This table displays statistics for the variables that are in the analysis at each step. *Tolerance* is the proportion of a variable's variance not accounted for by other independent variables in the equation. A variable with very low tolerance contributes little information to a model and can cause computational problems.

F to Remove values are useful for describing what happens if a variable is removed from the current model (given that the other variables remain). *F to Remove* for the entering variable is the same as *F to Enter* at the previous step (shown in the Variables Not in the Analysis table).

A Note of Caution Concerning Stepwise Methods

Stepwise methods are convenient, but have their limitations. Be aware that because stepwise methods select models based solely upon statistical merit, it may choose predictors that have no *practical significance*. If you have some experience with the data and have expectations about which predictors are important, you should use that knowledge and eschew stepwise methods. If, however, you have many predictors and no idea where to start, running a stepwise analysis and adjusting the selected model is better than no model at all.

Checking Model Fit

Function	Eigenvalue	% of Variance	Cumulative %	Canonical Correlation
1	.198	80.2	80.2	.407
2	.048	19.4	99.6	.214
3	.001	.4	100.0	.031

Figure 279. Eigenvalues

Nearly all of the variance explained by the model is due to the first two discriminant functions. Three functions are fit automatically, but due to its minuscule eigenvalue, you can fairly safely ignore the third.

Test of Function(s)	Wilks' Lambda	Chi-square	df	Sig.
1 through 3	.796	227.345	9	.000
2 through 3	.953	47.486	4	.000
3	.999	.929	1	.335

Figure 280. Wilks' lambda

Wilks' lambda agrees that only the first two functions are useful. For each set of functions, this tests the hypothesis that the means of the functions listed are equal across groups. The test of function 3 has a significance value greater than 0.10, so this function contributes little to the model.

Structure Matrix

	Function		
	1	2	3
Level of education	.966*	-.090	-.244
Years with current employer	-.182	.964*	-.193
Age in years [a]	-.162	.598*	-.285
Household income in thousands [a]	.109	.514*	-.190
Years at current address [a]	-.151	.394*	-.214
Retired [a]	-.108	.230*	-.137
Gender [a]	.008	.054*	.009
Number of people in household	.232	.097	.968*
Marital status [a]	.132	.134	.600*

Pooled within-groups correlations between discriminating variables and standardized canonical discriminant functions
Variables ordered by absolute size of correlation within function.

*. Largest absolute correlation between each variable and any discriminant function

a. This variable not used in the analysis.

Figure 281. Structure matrix

When there is more than one discriminant function, an asterisk(*) marks each variable's largest absolute correlation with one of the canonical functions. Within each function, these marked variables are then ordered by the size of the correlation.

- *Level of education* is most strongly correlated with the first function, and it is the only variable most strongly correlated with this function.
- *Years with current employer*, *Age in years*, *Household income in thousands*, *Years at current address*, *Retired*, and *Gender* are most strongly correlated with the second function, although *Gender* and *Retired* are more weakly correlated than the others. The other variables mark this function as a "stability" function.
- *Number of people in household* and *Marital status* are most strongly correlated with the third discriminant function, but this is a useless function, so these are nearly useless predictors.

Territorial Map

```
Canonical Discriminant
Function 2
       -3.0      -2.0      -1.0        .0       1.0       2.0       3.0
          +---------+---------+---------+---------+---------+---------+
         |                                 34                           |
     1.0 +          +         +         +  34     +         +           +
         |                                324                           |
         |                               3224                           |
         |                               32 24                          |
         |                         *    32  24                          |
         |                             32   24                          |
      .0 +          +         +    333332  *24 *  +         +           +
         |                  3333333111112    24                         |
         |           33333331111111  *  12   24                         |
         |    33333331111111             12  24                         |
         |33331111111                     12 24                         |
         |1111                             12 24                        |
    -1.0 +          +         +         +  1224   +         +           +
         |                                   124                        |
         |                                    14                        |
          +---------+---------+---------+---------+---------+---------+
       -3.0      -2.0      -1.0        .0       1.0       2.0       3.0
                       Canonical Discriminant Function 1
```

Figure 282. Territorial map

The territorial map helps you to study the relationships between the groups and the discriminant functions. Combined with the structure matrix results, it gives a graphical interpretation of the relationship between predictors and groups. The first function, shown on the horizontal axis, separates group 4 (*Total service* customers) from the others. Since *Level of education* is strongly positively correlated with the first function, this suggests that your *Total service* customers are, in general, the most highly educated. The second function separates groups 1 and 3 (*Basic service* and *Plus service* customers). *Plus service* customers tend to have been working longer and are older than *Basic service* customers. *E-service* customers are not separated well from the others, although the map suggests that they tend to be well educated with a moderate amount of work experience.

In general, the closeness of the group centroids, marked with asterisks (*), to the territorial lines suggests that the separation between all groups is not very strong.

Only the first two discriminant functions are plotted, but since the third function was found to be rather insignificant, the territorial map offers a comprehensive view of the discriminant model.

Classification Results

			Predicted Group Membership				
		Customer category	Basic service	E-service	Plus service	Total service	Total
Original	Count	Basic service	125	11	61	69	266
		E-service	49	15	58	95	217
		Plus service	102	14	112	53	281
		Total service	40	16	37	143	236
	%	Basic service	47.0	4.1	22.9	25.9	100.0
		E-service	22.6	6.9	26.7	43.8	100.0
		Plus service	36.3	5.0	39.9	18.9	100.0
		Total service	16.9	6.8	15.7	60.6	100.0

a. 39.5% of original grouped cases correctly classified.

Figure 283. Classification results

From Wilks' lambda, you know that your model is doing better than guessing, but you need to turn to the classification results to determine how much better. Given the observed data, the "null" model (that is, one without predictors) would classify all customers into the modal group, *Plus service*. Thus, the null model would be correct 281/1000 = 28.1% of the time. Your model gets 11.4% more or 39.5% of the customers. In particular, your model excels at identifying *Total service* customers. However, it does an exceptionally poor job of classifying *E-service* customers. You may need to find another predictor in order to separate these customers.

Summary

You have created a discriminant model that classifies customers into one of four predefined "service usage" groups, based on demographic information from each customer. Using the structure matrix and territorial map, you identified which variables are most useful for segmenting your customer base. Lastly, the classification results show that the model does poorly at classifying *E-service* customers. More research is required to determine another predictor variable that better classifies these customers, but depending on what you are looking to predict, the model may be perfectly adequate for your needs. For example, if you are not concerned with identifying *E-service* customers the model may be accurate enough for you. This may be the case where the E-service is a loss-leader which brings in little profit. If, for example, your highest return on investment comes from *Plus service* or *Total service* customers, the model may give you the information you need.

Also note that these results are based on the training data only. To assess how well the model generalizes to other data, you can use a Partition node to hold out a subset of records for purposes of testing and validation.

Explanations of the mathematical foundations of the modeling methods used in IBM SPSS Modeler are listed in the IBM SPSS Modeler Algorithms Guide. This is available from the *Documentation* directory of the installation disk.

Chapter 22. Analyzing Interval-Censored Survival Data (Generalized Linear Models)

When analyzing survival data with interval censoring—that is, when the exact time of the event of interest is not known but is known only to have occurred within a given interval—then applying the Cox model to the hazards of events in intervals results in a complementary log-log regression model.

Partial information from a study designed to compare the efficacy of two therapies for preventing the recurrence of ulcers is collected in *ulcer_recurrence.sav*. This dataset has been presented and analyzed elsewhere [1]. Using generalized linear models, you can replicate the results for the complementary log-log regression models.

This example uses the stream named *ulcer_genlin.str*, which references the data file *ulcer_recurrence.sav*. The data file is in the *Demos* folder and the stream file is in the *streams* subfolder.

Creating the Stream

1. Add a Statistics File source node pointing to *ulcer_recurrence.sav* in the *Demos* folder.

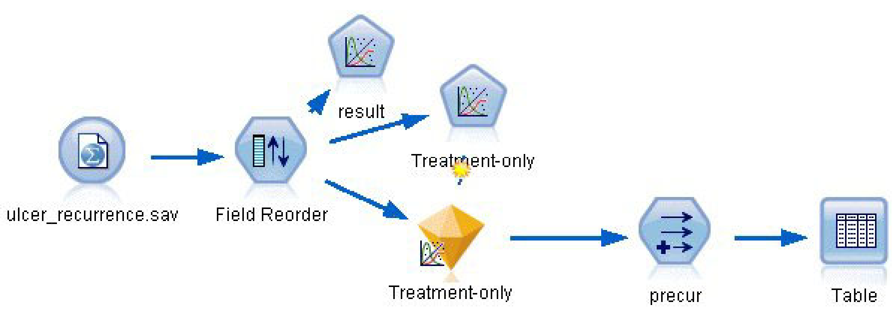


Figure 284. Sample stream to predict ulcer recurrence

2. On the Filter tab of the source node, filter out *id* and *time*.

1. Collett, D. 2003. *Modelling survival data in medical research*, 2 ed. Boca Raton: Chapman & Hall/CRC.

Figure 285. Filter unwanted fields

3. On the Types tab of the source node, set the role for the *result* field to **Target** and set its measurement level to **Flag**. A result of 1 indicates that the ulcer has recurred. All other fields should have their role set to **Input**.
4. Click **Read Values** to instantiate the data.

Figure 286. Setting field role

5. Add a Field Reorder node and specify *duration*, *treatment*, and *age* as the order of inputs. This determines the order in which fields are entered in the model and will help you try to replicate Collett's results.

Figure 287. Reordering fields so they are entered into the model as desired

6. Attach a GenLin node to the source node; on the GenLin node, click the **Model** tab.
7. Select **First (Lowest)** as the reference category for the target. This indicates that the second category is the event of interest, and its effect on the model is in the interpretation of parameter estimates. A continuous predictor with a positive coefficient indicates increased probability of recurrence with increasing values of the predictor; categories of a nominal predictor with larger coefficients indicate increased probability of recurrence with respect to other categories of the set.

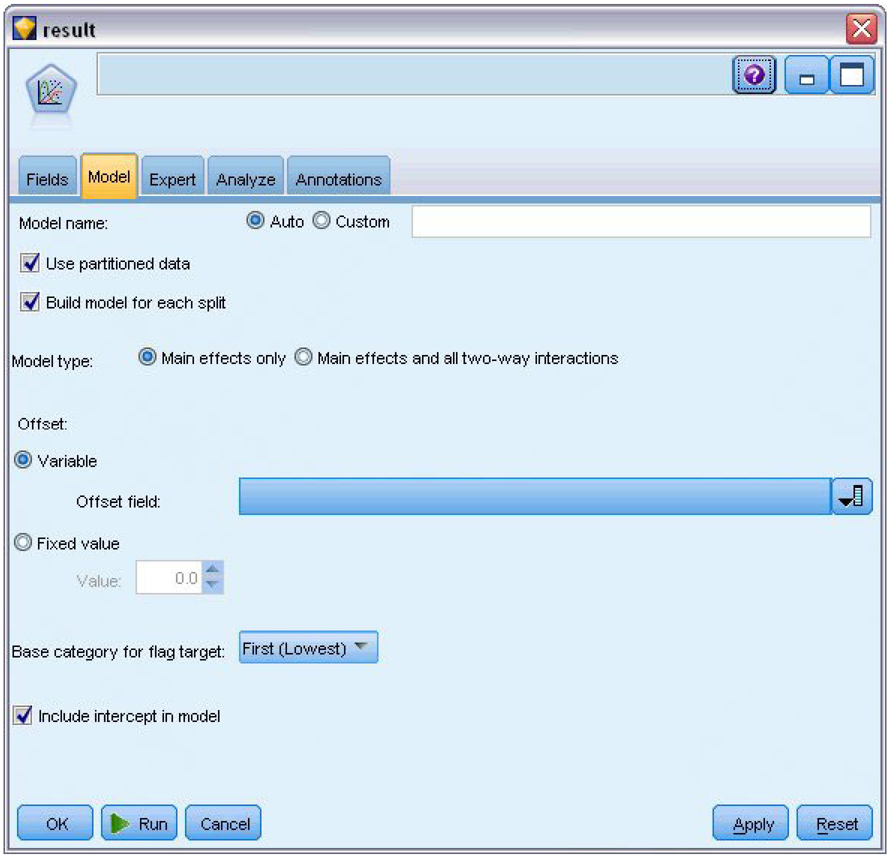


Figure 288. Choosing model options

8. Click the **Expert** tab and select **Expert** to activate the expert modeling options.
9. Select **Binomial** as the distribution and **Complementary log-log** as the link function.
10. Select **Fixed value** as the method for estimating the scale parameter and leave the default value of 1.0.
11. Select **Descending** as the category order for factors. This indicates that the first category of each factor will be its reference category; the effect of this selection on the model is in the interpretation of parameter estimates.

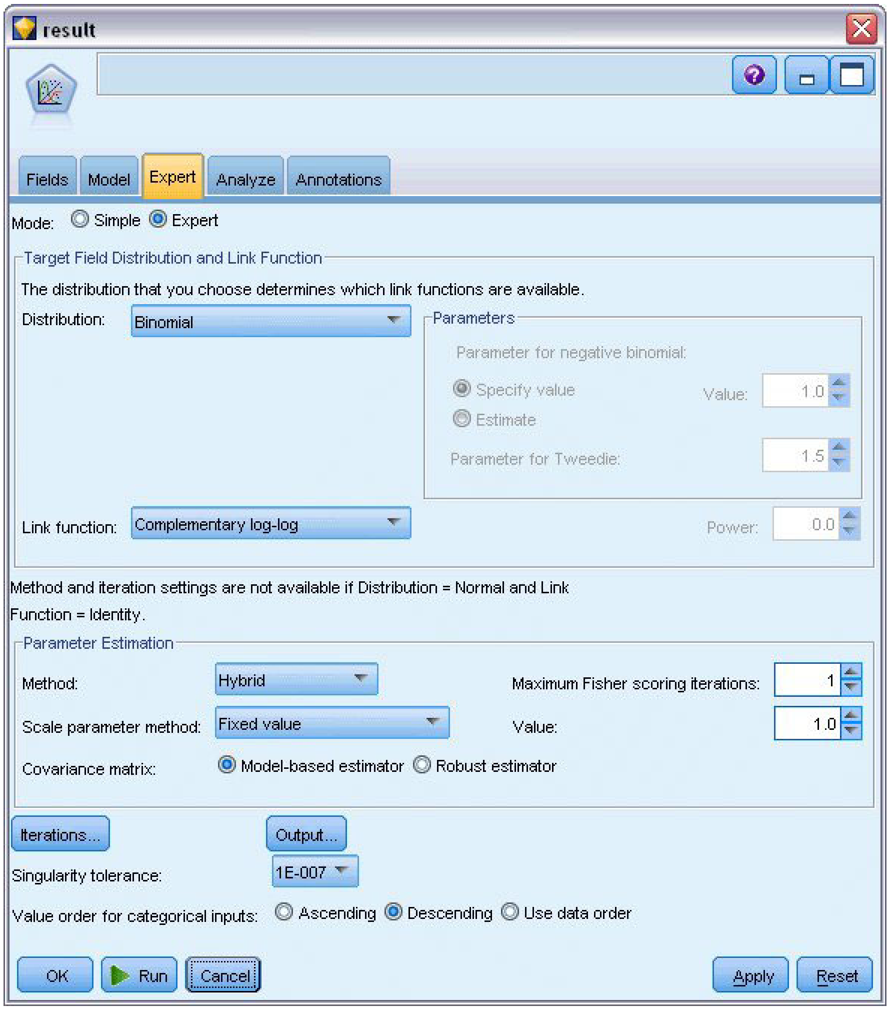


Figure 289. Choosing expert options

12. Run the stream to create the model nugget, which is added to the stream canvas, and also to the Models palette in the upper right corner. To view the model details, right-click the nugget and choose **Edit** or **Browse**.

Tests of Model Effects

Source	Type III		
	Wald Chi-Square	df	Sig.
(Intercept)	.536	1	.464
duration	.003	1	.958
treatment	.382	1	.537
age	.358	1	.550

Dependent Variable: Result
Model: (Intercept), duration, treatment, age

Figure 290. Tests of model effects for main-effects model

None of the model effects is statistically significant; however, any observable differences in the treatment effects are of clinical interest, so we will fit a reduced model with just the treatment as a model term.

Fitting the Treatment-Only Model

1. On the Fields tab of the GenLin node, click **Use custom settings**.
2. Select *result* as the target.
3. Select *treatment* as the sole input.

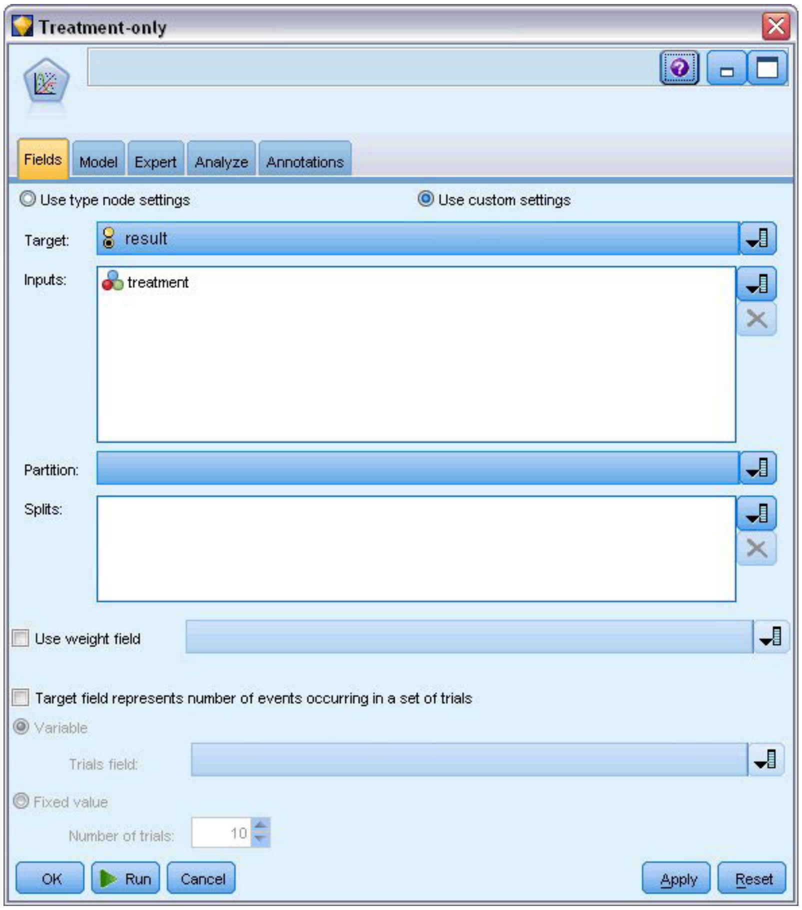


Figure 291. Choosing field options

4. Run the stream and open the resulting model nugget.

On the model nugget, select the **Advanced** tab and scroll to the bottom.

Parameter Estimates

Parameter	B	Std. Error	95% Wald Confidence Interval		Hypothesis Test		
			Lower	Upper	Wald Chi-Square	df	Sig.
(Intercept)	-1.442	.5012	-2.425	-.460	8.282	1	.004
[treatment=1]	.378	.6288	-.855	1.610	.361	1	.548
[treatment=0]	0[a]	.	.	.	.	.	.
(Scale)	1[b]						

Dependent Variable: Result
Model: (Intercept), treatment

a. Set to zero because this parameter is redundant.

b. Fixed at the displayed value.

Figure 292. Parameter estimates for treatment-only model

The treatment effect (the difference of the linear predictor between the two treatment levels; that is, the coefficient for *[treatment=1]*) is still not statistically significant, but only suggestive that treatment *A* *[treatment=0]* may be better than *B* *[treatment=1]* because the parameter estimate for treatment *B* is larger than that for *A*, and is thus associated with an increased probability of recurrence in the first 12 months. The linear predictor, (intercept + treatment effect) is an estimate of $\log(-\log(1-P(recur_{12,t}))$, where $P(recur_{12,\ t})$ is the probability of recurrence at 12 months for treatment t(=*A* or *B*). These predicted probabilities are generated for each observation in the dataset.

Predicted Recurrence and Survival Probabilities

Figure 293. Derive node settings options

1. For each patient, the model scores the predicted result and the probability of that predicted result. In order to see the predicted recurrence probabilities, copy the generated model to the palette and attach a Derive node.
2. In the Settings tab, type `precur` as the derive field.
3. Choose to derive it as **Conditional**.
4. Click the calculator button to open the Expression Builder for the **If** condition.

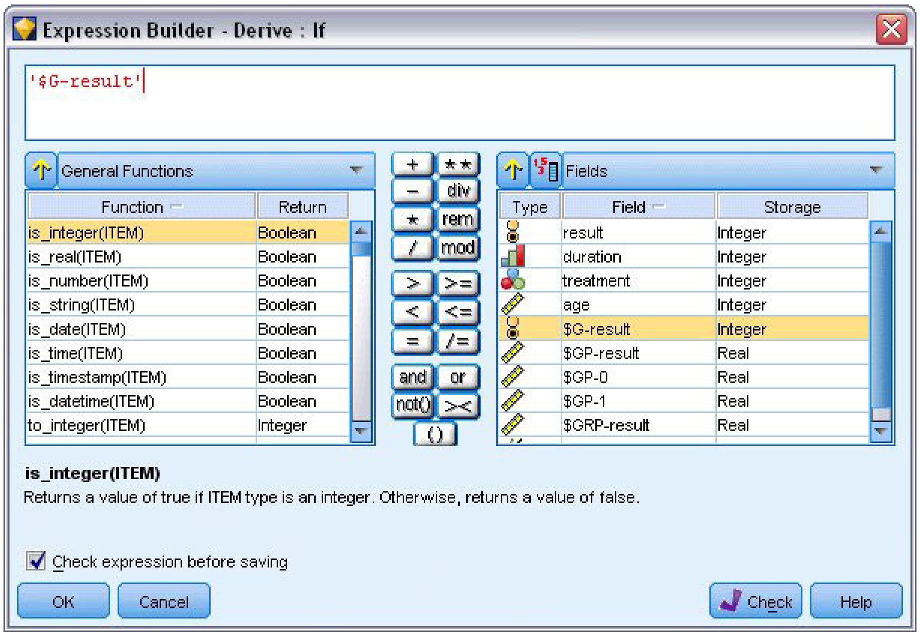


Figure 294. Derive node: Expression Builder for If condition

5. Insert the *$G-result* field into the expression.
6. Click **OK**.

   The derive field *precur* will take the value of the **Then** expression when *$G-result* equals 1 and the value of the **Else** expression when it is 0.

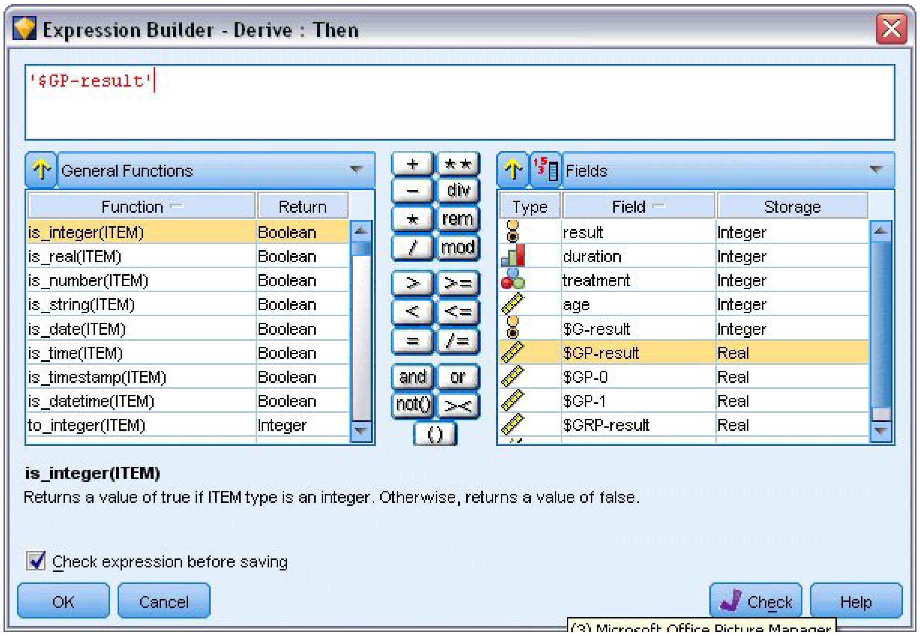


Figure 295. Derive node: Expression Builder for Then expression

7. Click the calculator button to open the Expression Builder for the **Then** expression.
8. Insert the *$GP-result* field into the expression.
9. Click **OK**.

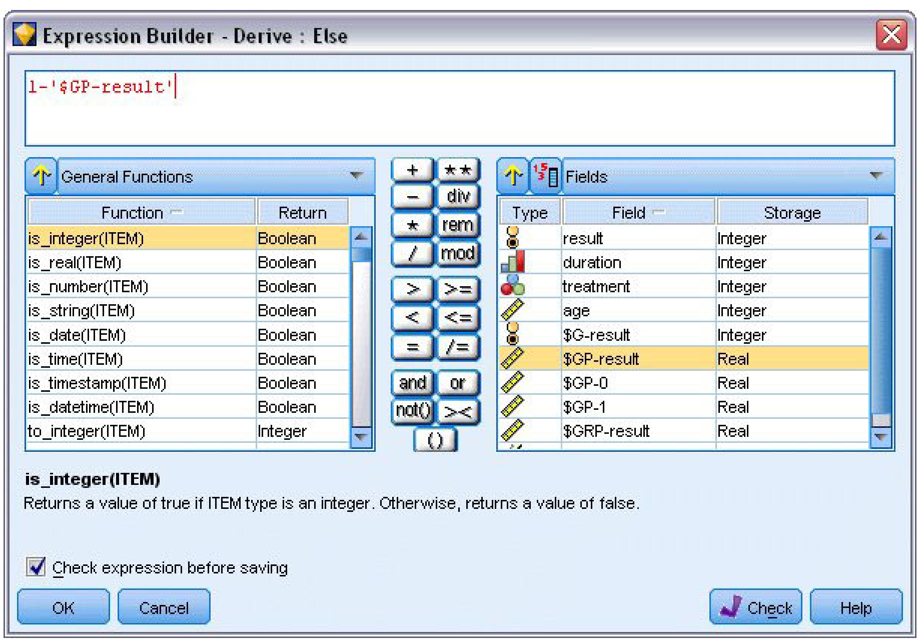
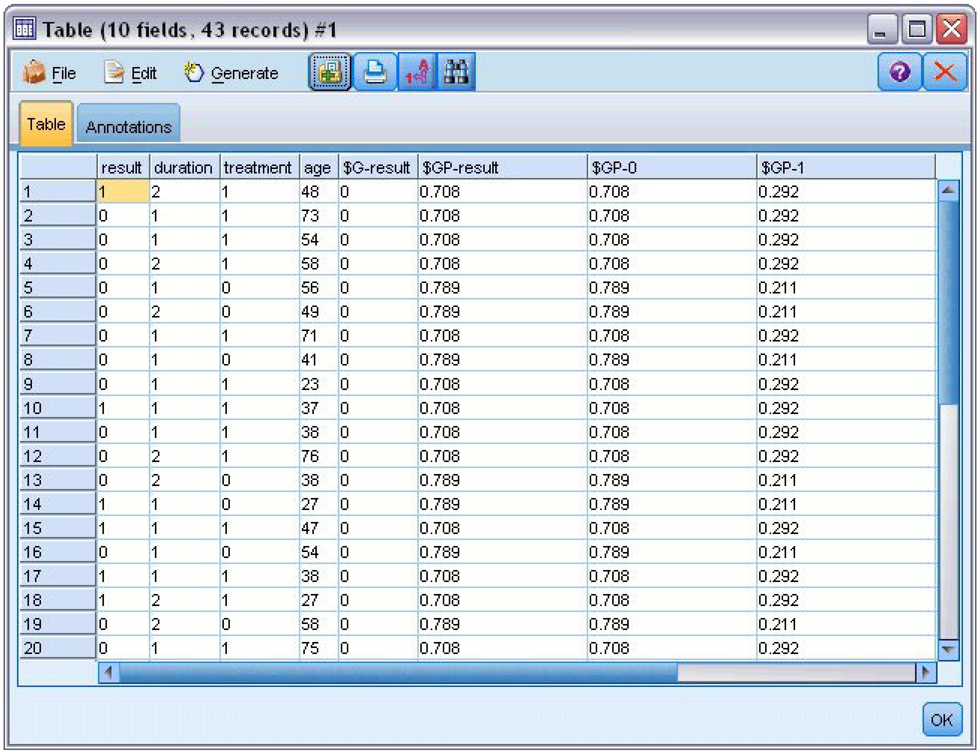


Figure 296. Derive node: Expression Builder for Else expression

10. Click the calculator button to open the Expression Builder for the **Else** expression.
11. Type 1- in the expression and then insert the *$GP-result* field into the expression.
12. Click **OK**.

Figure 297. Derive node settings options

13. Attach a table node to the Derive node and execute it.



	result	duration	treatment	age	$G-result	$GP-result	$GP-0	$GP-1
1	1	2	1	48	0	0.708	0.708	0.292
2	0	1	1	73	0	0.708	0.708	0.292
3	0	1	1	54	0	0.708	0.708	0.292
4	0	2	1	58	0	0.708	0.708	0.292
5	0	1	0	56	0	0.789	0.789	0.211
6	0	2	0	49	0	0.789	0.789	0.211
7	0	1	1	71	0	0.708	0.708	0.292
8	0	1	0	41	0	0.789	0.789	0.211
9	0	1	1	23	0	0.708	0.708	0.292
10	1	1	1	37	0	0.708	0.708	0.292
11	0	1	1	38	0	0.708	0.708	0.292
12	0	2	1	76	0	0.708	0.708	0.292
13	0	2	0	38	0	0.789	0.789	0.211
14	1	1	0	27	0	0.789	0.789	0.211
15	1	1	1	47	0	0.708	0.708	0.292
16	0	1	0	54	0	0.789	0.789	0.211
17	1	1	1	38	0	0.708	0.708	0.292
18	1	2	1	27	0	0.708	0.708	0.292
19	0	2	0	58	0	0.789	0.789	0.211
20	0	1	1	75	0	0.708	0.708	0.292

Figure 298. Predicted probabilities

There is an estimated 0.211 probability that patients assigned to treatment *A* will experience a recurrence in the first 12 months; 0.292 for treatment *B*. Note that 1−P($recur_{12,\ t}$) is the survivor probability at 12 months, which may be of more interest to survival analysts.

Modeling the Recurrence Probability by Period

A problem with the model as it stands is that it ignores the information gathered at the first examination; that is, that many patients did not experience a recurrence in the first six months. A "better" model would model a binary response that records whether or not the event occurred during each interval. Fitting this model requires a reconstruction of the original dataset, which can be found in *ulcer_recurrence_recoded.sav*. This file contains two additional variables:

- *Period*, which records whether the case corresponds to the first examination period or the second.
- *Result by period*, which records whether there was a recurrence for the given patient during the given period.

Each original case (patient) contributes one case per interval in which it remains in the risk set. Thus, for example, patient 1 contributes two cases; one for the first examination period in which no recurrence occurred, and one for the second examination period, in which a recurrence was recorded. Patient 10, on the other hand, contributes a single case because a recurrence was recorded in the first period. Patients 16, 28, and 34 dropped out of the study after six months, and thus contribute only a single case to the new dataset.

1. Add a Statistics File source node pointing to *ulcer_recurrence_recoded.sav* in the *Demos* folder.

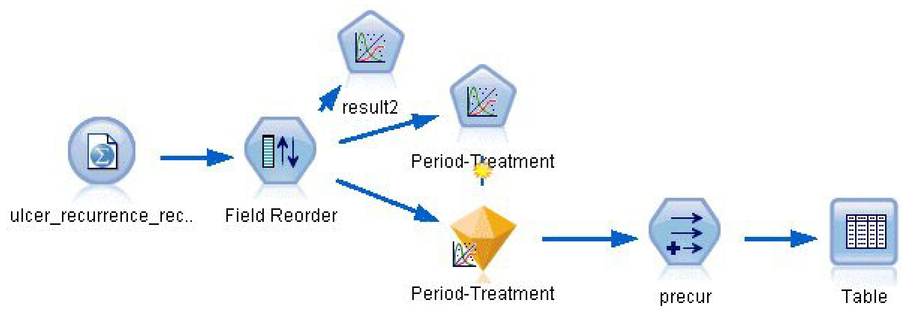


Figure 299. Sample stream to predict ulcer recurrence

2. On the Filter tab of the source node, filter out *id*, *time*, and *result*.

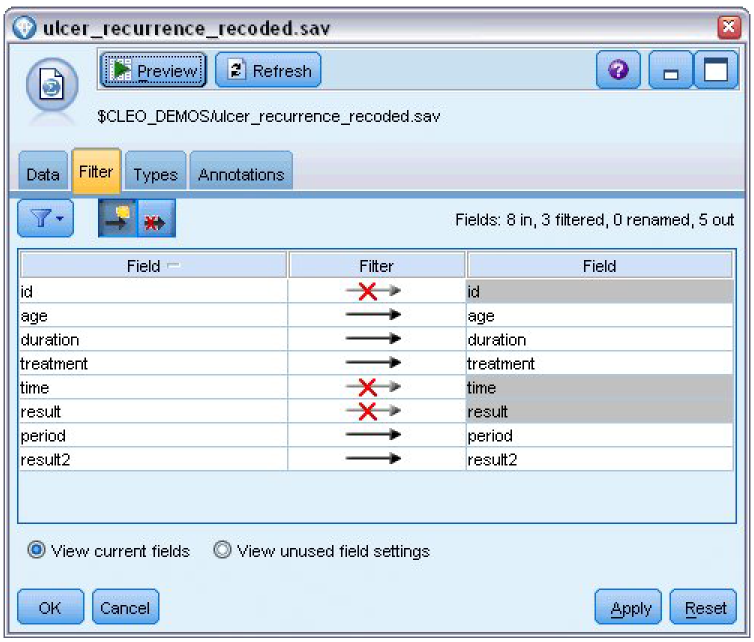


Figure 300. Filter unwanted fields

3. On the Types tab of the source node, set the role for the *result2* field to **Target** and set its measurement level to **Flag**. All other fields should have their role set to **Input**.

Figure 301. Setting field role

4. Add a Field Reorder node and specify *period*, *duration*, *treatment*, and *age* as the order of inputs. Making *period* the first input (and not including the intercept term in the model) will allow you to fit a full set of dummy variables to capture the period effects.

Figure 302. Reordering fields so they are entered into the model as desired

5. On the GenLin node, click the **Model** tab.

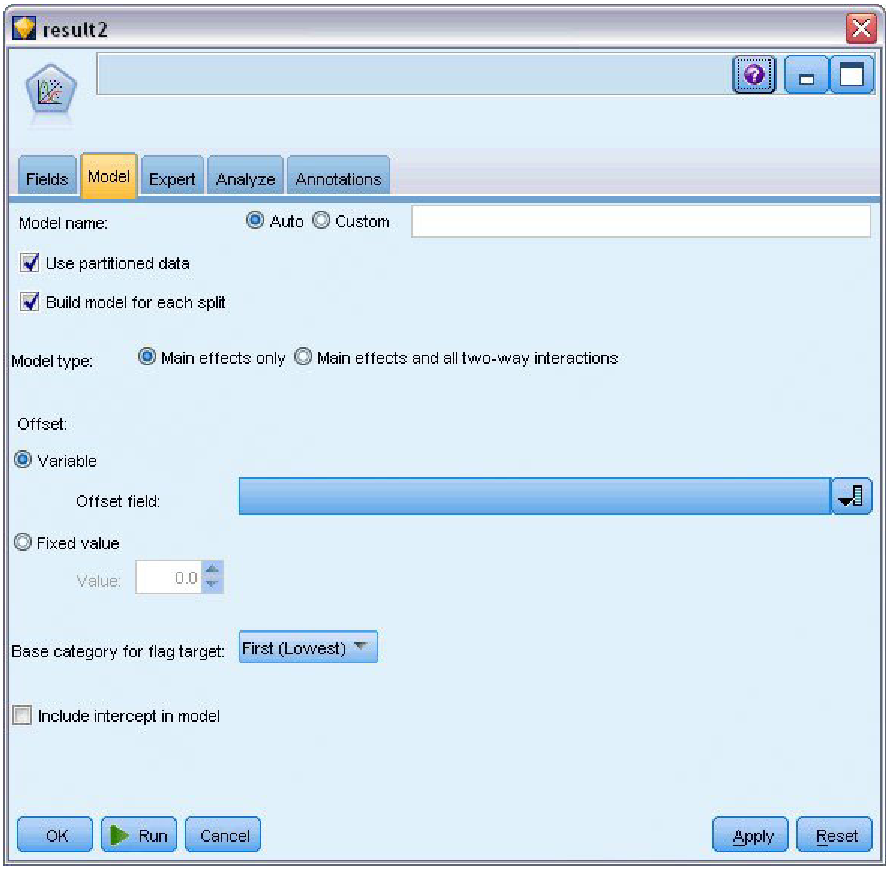


Figure 303. Choosing model options

6. Select **First (Lowest)** as the reference category for the target. This indicates that the second category is the event of interest, and its effect on the model is in the interpretation of parameter estimates.
7. Deselect **Include intercept in model**.
8. Click the **Expert** tab and select **Expert** to activate the expert modeling options.

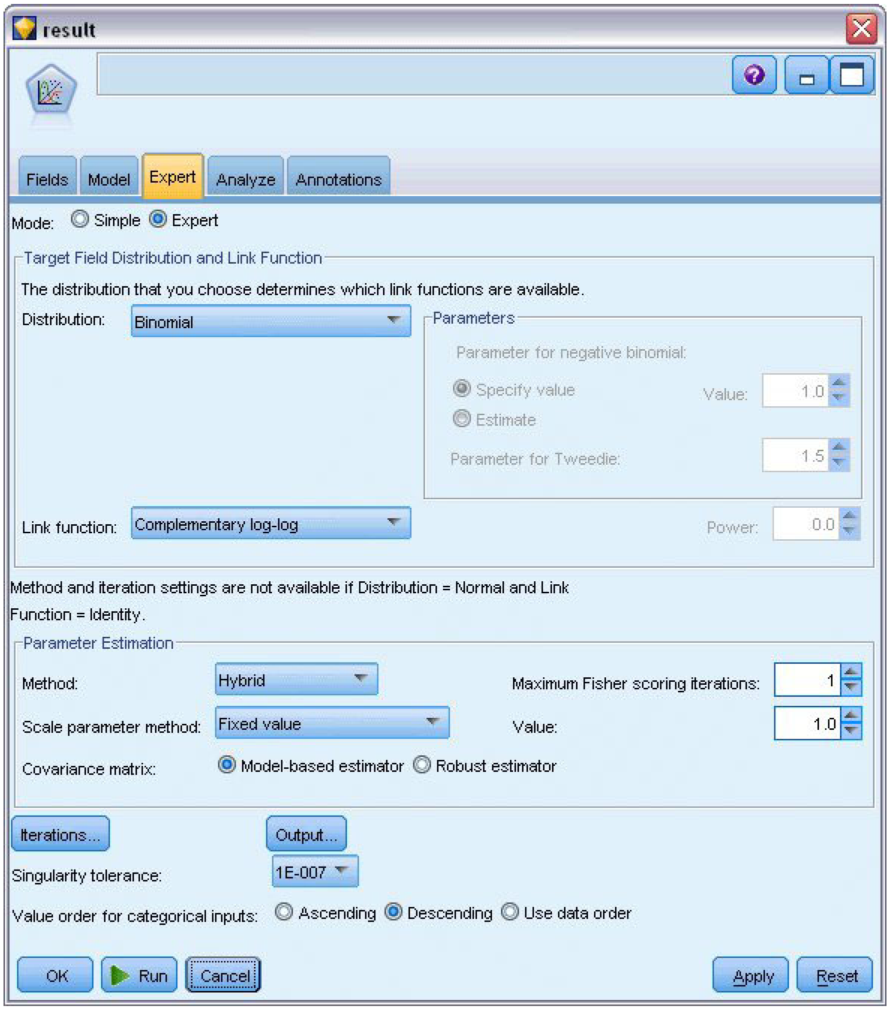


Figure 304. Choosing expert options

9. Select **Binomial** as the distribution and **Complementary log-log** as the link function.
10. Select **Fixed value** as the method for estimating the scale parameter and leave the default value of 1.0.
11. Select **Descending** as the category order for factors. This indicates that the first category of each factor will be its reference category; the effect of this selection on the model is in the interpretation of parameter estimates.
12. Run the stream to create the model nugget, which is added to the stream canvas, and also to the Models palette in the upper right corner. To view the model details, right-click the nugget and choose **Edit** or **Browse**.

Tests of Model Effects

Source	Type III		
	Wald Chi-Square	df	Sig.
period	.464	1	.496
duration	.000	1	.988
treatment	.117	1	.732
age	.314	1	.575

Dependent Variable: Result by period
Model: period, duration, treatment, age

Figure 305. Tests of model effects for main-effects model

None of the model effects is statistically significant; however, any observable differences in the period and treatment effects are of clinical interest, so we will fit a reduced model with just those model terms.

Fitting the Reduced Model

1. On the Fields tab of the GenLin node, click **Use custom settings**.
2. Select *result2* as the target.
3. Select *period* and *treatment* as the inputs.

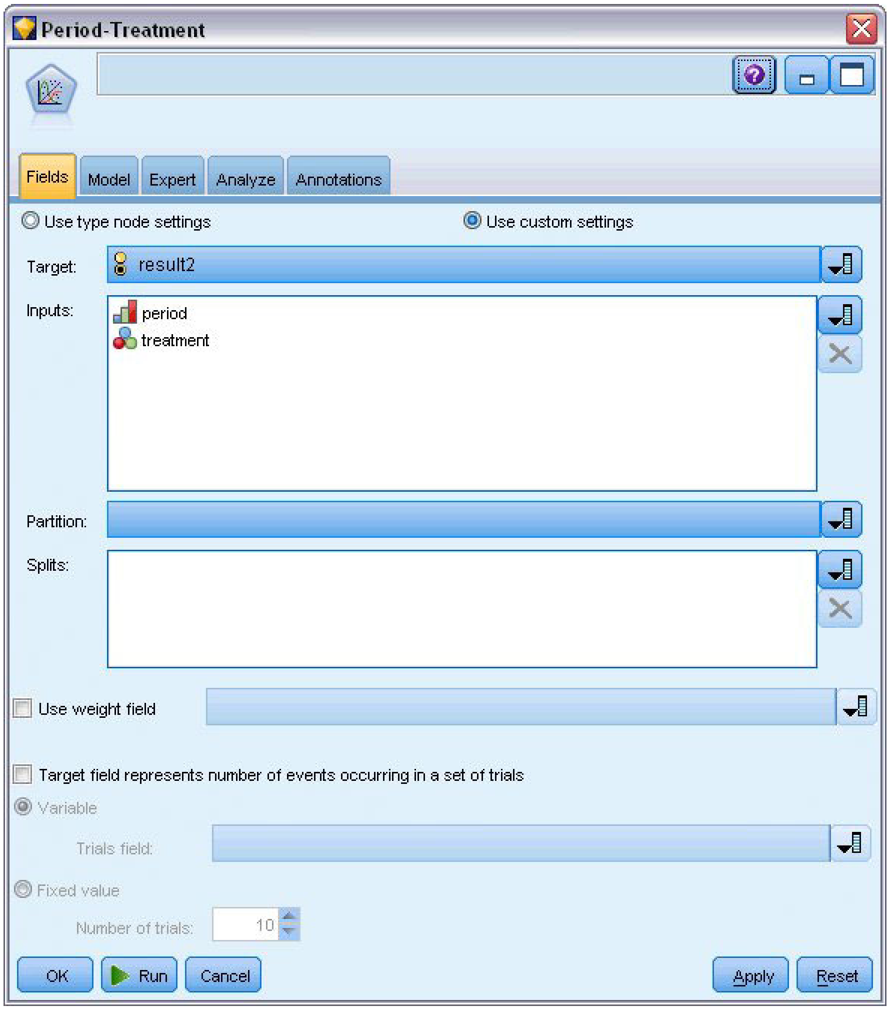


Figure 306. Choosing field options

4. Execute the node and browse the generated model, and then copy the generated model to the palette, attach a table node, and execute it.

Parameter Estimates

Parameter	B	Std. Error	95% Wald Confidence Interval Lower	95% Wald Confidence Interval Upper	Hypothesis Test Wald Chi-Square	Hypothesis Test df	Hypothesis Test Sig.
[period=2]	-1.794	.5792	-2.929	-.659	9.597	1	.002
[period=1]	-2.206	.5912	-3.365	-1.047	13.926	1	.000
[treatment=1]	.195	.6279	-1.035	1.426	.097	1	.756
[treatment=0]	0[a]	.	.	.	.	.	.
(Scale)	1[b]						

Dependent Variable: Result by period
Model: period, treatment

a. Set to zero because this parameter is redundant.

b. Fixed at the displayed value.

Figure 307. Parameter estimates for treatment-only model

The treatment effect is still not statistically significant but only suggestive that treatment *A* may be better than *B* because the parameter estimate for treatment *B* is associated with an increased probability of recurrence in the first 12 months. The period values are statistically significantly different from 0, but this is because of the fact that an intercept term is not fit. The period effect (the difference between the values of the linear predictor for *[period=1]* and *[period=2]*) is not statistically significant, as can be seen in the tests of model effects. The linear predictor (period effect + treatment effect) is an estimate of $\log(-\log(1-P(recur_{p,\ t}))$, where $P(recur_{p,\ t})$ is the probability of recurrence at the period p(=1 or 2, representing six months or 12 months) given treatment t(=*A* or *B*). These predicted probabilities are generated for each observation in the dataset.

Predicted Recurrence and Survival Probabilities

Figure 308. Derive node settings options

1. For each patient, the model scores the predicted result and the probability of that predicted result. In order to see the predicted recurrence probabilities, copy the generated model to the palette and attach a Derive node.
2. In the Settings tab, type `precur` as the derive field.
3. Choose to derive it as **Conditional**.
4. Click the calculator button to open the Expression Builder for the **If** condition.

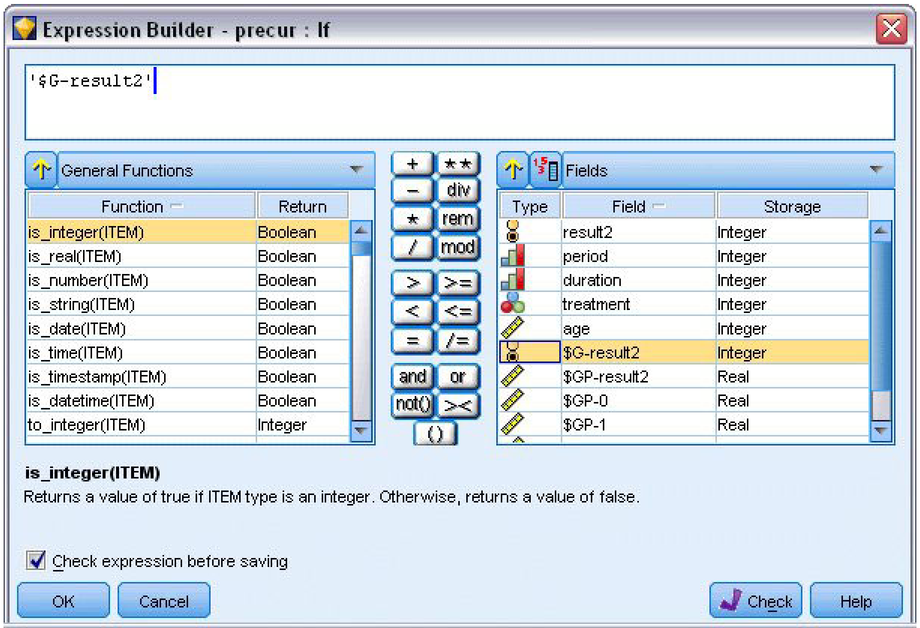


Figure 309. Derive node: Expression Builder for If condition

5. Insert the *$G-result2* field into the expression.
6. Click **OK**.

   The derive field *precur* will take the value of the **Then** expression when *$G-result2* equals 1 and the value of the **Else** expression when it is 0.

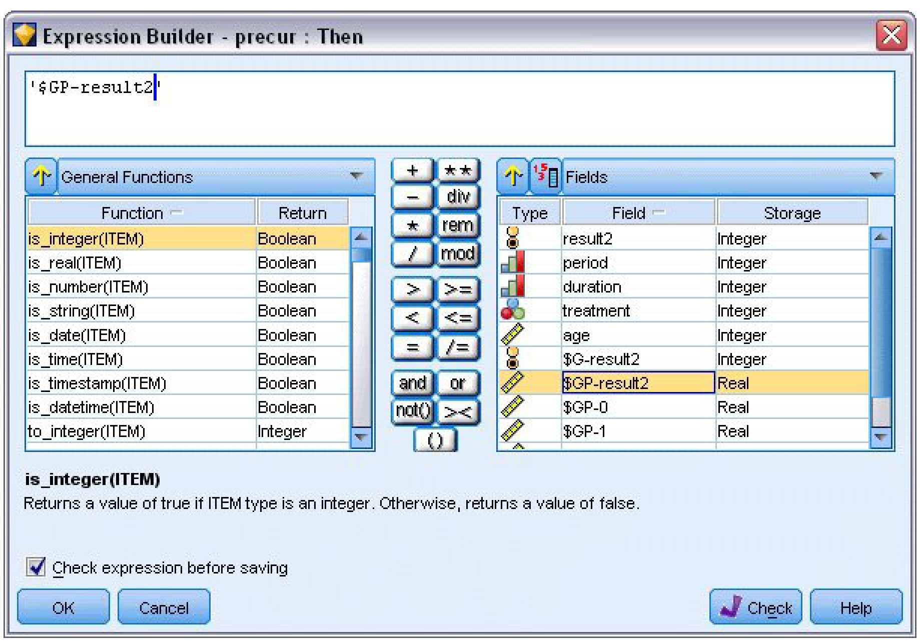


Figure 310. Derive node: Expression Builder for Then expression

7. Click the calculator button to open the Expression Builder for the **Then** expression.
8. Insert the *$GP-result2* field into the expression.
9. Click **OK**.

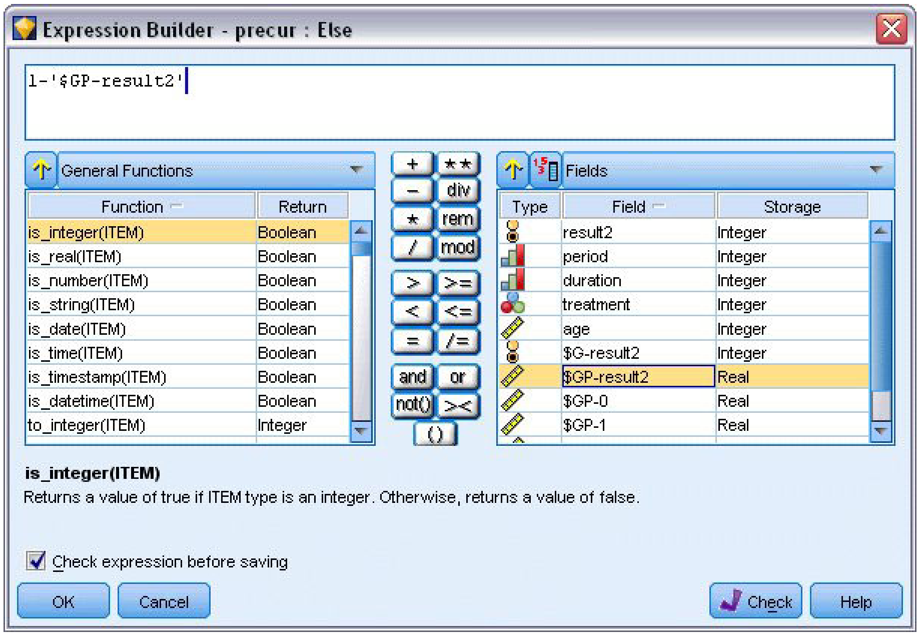
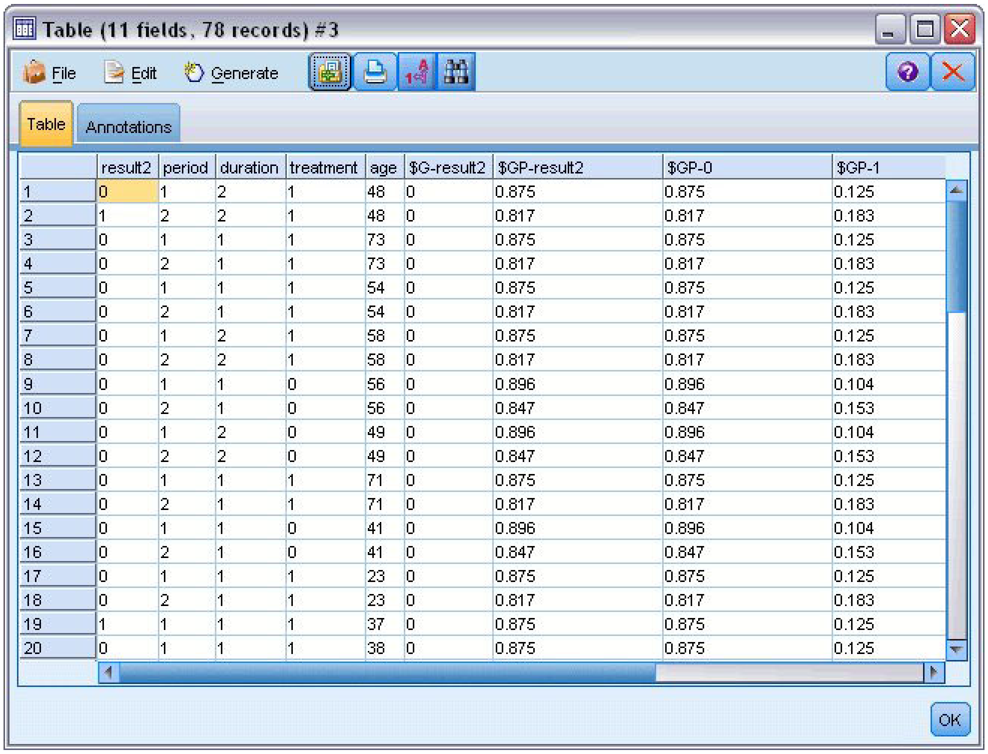


Figure 311. Derive node: Expression Builder for Else expression

10. Click the calculator button to open the Expression Builder for the **Else** expression.
11. Type 1- in the expression and then insert the *$GP-result2* field into the expression.
12. Click **OK**.

Figure 312. Derive node settings options

13. Attach a table node to the Derive node and execute it.



	result2	period	duration	treatment	age	$G-result2	$GP-result2	$GP-0	$GP-1
1	0	1	2	1	48	0	0.875	0.875	0.125
2	1	2	2	1	48	0	0.817	0.817	0.183
3	0	1	1	1	73	0	0.875	0.875	0.125
4	0	2	1	1	73	0	0.817	0.817	0.183
5	0	1	1	1	54	0	0.875	0.875	0.125
6	0	2	1	1	54	0	0.817	0.817	0.183
7	0	1	2	1	58	0	0.875	0.875	0.125
8	0	2	2	1	58	0	0.817	0.817	0.183
9	0	1	1	0	56	0	0.896	0.896	0.104
10	0	2	1	0	56	0	0.847	0.847	0.153
11	0	1	2	0	49	0	0.896	0.896	0.104
12	0	2	2	0	49	0	0.847	0.847	0.153
13	0	1	1	1	71	0	0.875	0.875	0.125
14	0	2	1	1	71	0	0.817	0.817	0.183
15	0	1	1	0	41	0	0.896	0.896	0.104
16	0	2	1	0	41	0	0.847	0.847	0.153
17	0	1	1	1	23	0	0.875	0.875	0.125
18	0	2	1	1	23	0	0.817	0.817	0.183
19	1	1	1	1	37	0	0.875	0.875	0.125
20	0	1	1	1	38	0	0.875	0.875	0.125

Figure 313. Predicted probabilities

Table 3. Estimated recurrence probabilities

Treatment	**6 months**	**12 months**
A	0.104	0.153
B	0.125	0.183

From the estimated recurrence probabilities, the survival probability through 12 months can be estimated as $1-(P(recur_{1,\ t}) + P(recur_{2,\ t})\times(1-P(recur_{1,\ t})))$; thus, for each treatment:

A: 1 − (0.104 + 0.153*0.896) = 0.759

B: 1 − (0.125 + 0.183*0.875) = 0.715

which again shows nonstatistically significant support for *A* as the better treatment.

Summary

Using Generalized Linear Models, you have fit a series of complementary log-log regression models for interval-censored survival data. While there is some support for choosing treatment *A*, achieving a statistically significant result may require a larger study. However, there are some further avenues to explore with the existing data.

- It may be worthwhile to refit the model with interaction effects, particularly between *Period* and *Treatment group*.

Explanations of the mathematical foundations of the modeling methods used in IBM SPSS Modeler are listed in the *IBM SPSS Modeler Algorithms Guide*.

Related Procedures

The Generalized Linear Models procedure is a powerful tool for fitting a variety of models.

- The Generalized Estimating Equations procedure extends the generalized linear model to allow repeated measurements.
- The Linear Mixed Models procedure allows you to fit models for scale dependent variables with a random component and/or repeated measurements.

Recommended Readings

See the following texts for more information on generalized linear models:

Cameron, A. C., and P. K. Trivedi. 1998. *Regression Analysis of Count Data*. Cambridge: Cambridge University Press. Dobson, A. J. 2002. *An Introduction to Generalized Linear Models*, 2 ed. Boca Raton, FL: Chapman & Hall/CRC. Hardin, J. W., and J. M. Hilbe. 2003. *Generalized Linear Models and Extension*. Station, TX: Stata Press. McCullagh, P., and J. A. Nelder. 1989. *Generalized Linear Models*, 2nd ed. London: Chapman & Hall.

Chapter 23. Using Poisson Regression to Analyze Ship Damage Rates (Generalized Linear Models)

A generalized linear model can be used to fit a Poisson regression for the analysis of count data. For example, a dataset presented and analyzed elsewhere [2] concerns damage to cargo ships caused by waves. The incident counts can be modeled as occurring at a Poisson rate given the values of the predictors, and the resulting model can help you determine which ship types are most prone to damage.

This example uses the stream *ships_genlin.str*, which references the data file *ships.sav*. The data file is in the *Demos* folder and the stream file is in the *streams* subfolder.

Modeling the raw cell counts can be misleading in this situation because the *Aggregate months of service* varies by ship type. Variables like this that measure the amount of "exposure" to risk are handled within the generalized linear model as offset variables. Moreover, a Poisson regression assumes that the log of the dependent variable is linear in the predictors. Thus, to use generalized linear models to fit a Poisson regression to the accident rates, you need to use *Logarithm of aggregate months of service*.

Fitting an "Overdispersed" Poisson Regression

1. Add a Statistics File source node pointing to *ships.sav* in the *Demos* folder.

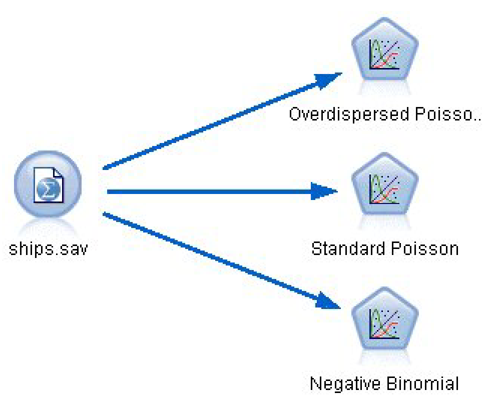


Figure 314. Sample stream to analyze damage rates

2. On the Filter tab of the source node, exclude the field *months_service*. The log-transformed values of this variable are contained in *log_months_service*, which will be used in the analysis.

2. McCullagh, P., and J. A. Nelder. 1989. *Generalized Linear Models*, 2nd ed. London: Chapman & Hall.

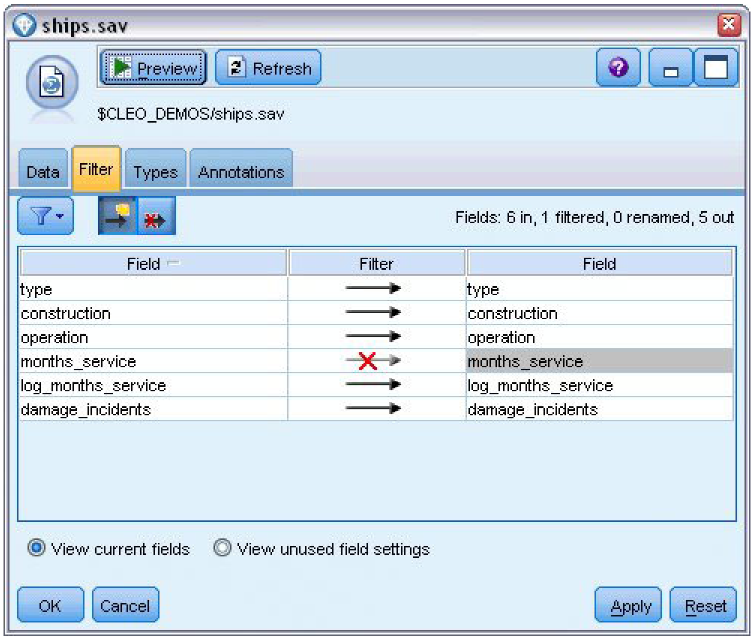


Figure 315. Filtering an unneeded field

(Alternatively, you could change the role to **None** for this field on the Types tab rather than exclude it, or select the fields you want to use in the modeling node.)

3. On the Types tab of the source node, set the role for the *damage_incidents* field to **Target**. All other fields should have their role set to **Input**.
4. Click **Read Values** to instantiate the data.

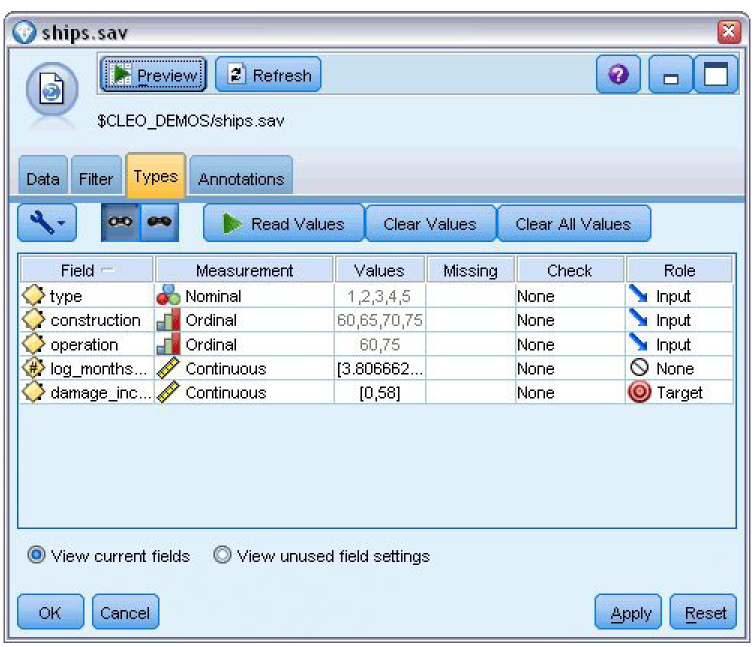


Figure 316. Setting field role

5. Attach a Genlin node to the source node; on the Genlin node, click the **Model** tab.
6. Select *log_months_service* as the offset variable.

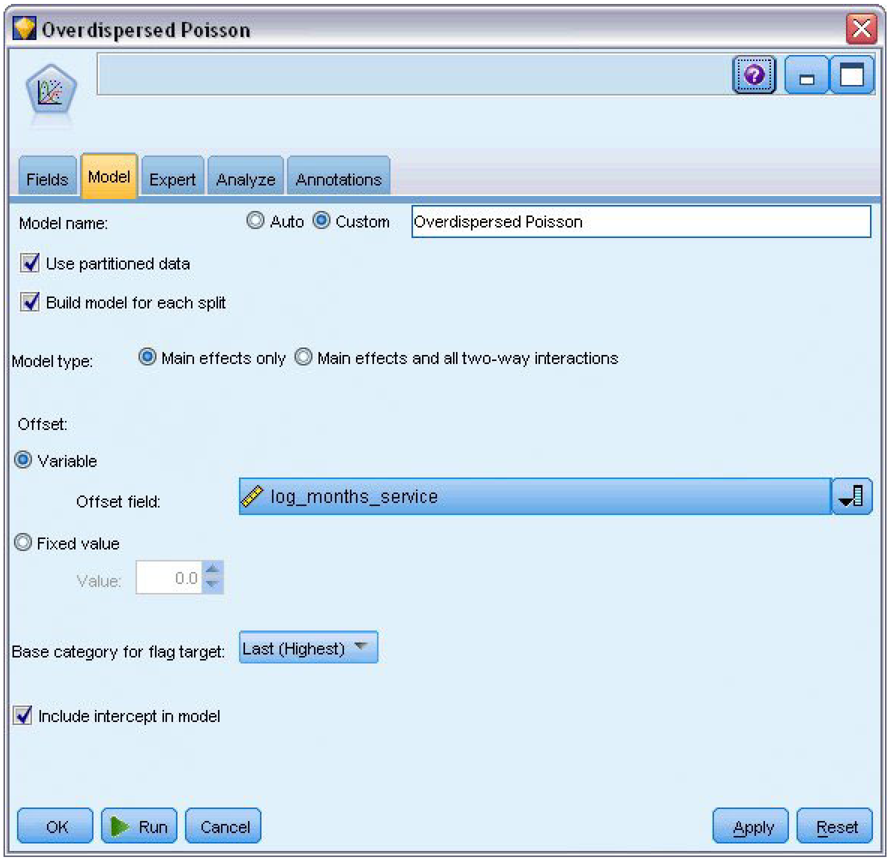


Figure 317. Choosing model options

7. Click the **Expert** tab and select **Expert** to activate the expert modeling options.

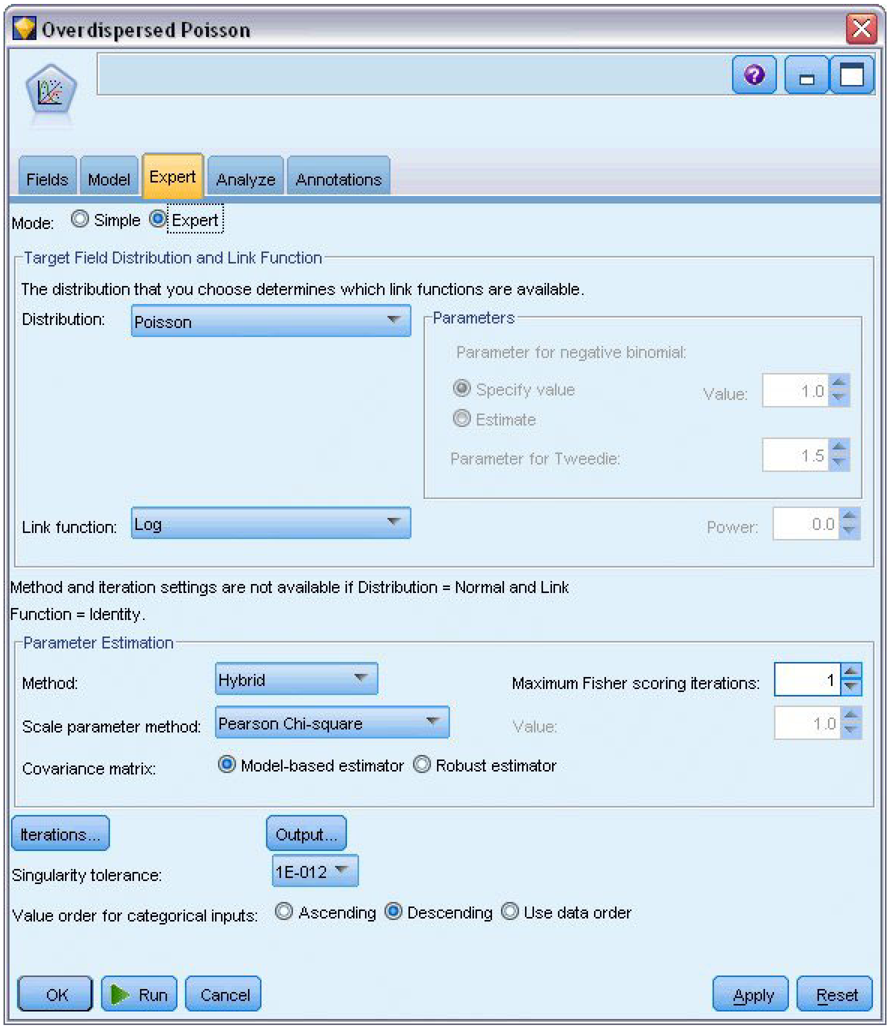


Figure 318. Choosing expert options

8. Select **Poisson** as the distribution for the response and **Log** as the link function.
9. Select **Pearson Chi-Square** as the method for estimating the scale parameter. The scale parameter is usually assumed to be 1 in a Poisson regression, but McCullagh and Nelder use the Pearson chi-square estimate to obtain more conservative variance estimates and significance levels.
10. Select **Descending** as the category order for factors. This indicates that the first category of each factor will be its reference category; the effect of this selection on the model is in the interpretation of parameter estimates.
11. Click **Run** to create the model nugget, which is added to the stream canvas, and also to the Models palette in the upper right corner. To view the model details, right-click the nugget and choose **Edit** or **Browse**, then click the **Advanced** tab.

Goodness-of-Fit Statistics

	Value	df	Value/df
Deviance	38.695	25	1.548
Scaled Deviance	22.883	25	
Pearson Chi-Square	42.275	25	1.691
Scaled Pearson Chi-Square	25.000	25	
Log Likelihood[a]	-68.281		
Akaike's Information Criterion (AIC)	154.562		
Finite Sample Corrected AIC (AICC)	162.062		
Bayesian Information Criterion (BIC)	168.299		
Consistent AIC (CAIC)	177.299		

Dependent Variable: Number of damage incidents
Model: (Intercept), type, construction, operation, offset = log_months_service

a. The full log likelihood function is displayed and used in computing information criteria.

b. Information criteria are in small-is-better form.

Figure 319. Goodness-of-fit statistics

The goodness-of-fit statistics table provides measures that are useful for comparing competing models. Additionally, the *Value/df* for the Deviance and Pearson Chi-Square statistics gives corresponding estimates for the scale parameter. These values should be near 1.0 for a Poisson regression; the fact that they are greater than 1.0 indicates that fitting the overdispersed model may be reasonable.

Omnibus Test

Likelihood Ratio Chi-Square	df	Sig.
107.633	8	.000

Dependent Variable: Number of damage incidents
Model: (Intercept), type, construction, operation, offset = log_months_service

a. Compares the fitted model against the intercept-only model.

Figure 320. Omnibus test

The omnibus test is a likelihood-ratio chi-square test of the current model versus the null (in this case, intercept) model. The significance value of less than 0.05 indicates that the current model outperforms the null model.

Tests of Model Effects

Source	Type III		
	Wald Chi-Square	df	Sig.
(Intercept)	2138.657	1	.000
type	15.415	4	.004
construction	17.242	3	.001
operation	6.249	1	.012

Dependent Variable: Number of damage incidents
Model: (Intercept), type, construction, operation, offset = log_months_service

Figure 321. Tests of model effects

Each term in the model is tested for whether it has any effect. Terms with significance values less than 0.05 have some discernible effect. Each of the main-effects terms contributes to the model.

Parameter Estimates

Parameter	B	Std. Error	95% Wald Confidence Interval		Hypothesis Test		
			Lower	Upper	Wald Chi-Square	df	Sig.
(Intercept)	-6.406	.2828	-6.960	-5.852	513.238	1	.000
[type=5]	.326	.3067	-.276	.927	1.127	1	.288
[type=4]	-.076	.3779	-.817	.665	.040	1	.841
[type=3]	-.687	.4279	-1.526	.151	2.581	1	.108
[type=2]	-.543	.2309	-.996	-.091	5.536	1	.019
[type=1]	0[a]	.	.	.	.	.	.
[construction=75]	.453	.3032	-.141	1.048	2.236	1	.135
[construction=70]	.818	.2208	.386	1.251	13.743	1	.000
[construction=65]	.697	.1946	.316	1.079	12.835	1	.000
[construction=60]	0[a]	.	.	.	.	.	.
[operation=75]	.384	.1538	.083	.686	6.249	1	.012
[operation=60]	0[a]	.	.	.	.	.	.
(Scale)	1.691[b]						

Dependent Variable: Number of damage incidents
Model: (Intercept), type, construction, operation, offset = log_months_service

a. Set to zero because this parameter is redundant.

b. Computed based on the Pearson chi-square.

Figure 322. Parameter estimates

The parameter estimates table summarizes the effect of each predictor. While interpretation of the coefficients in this model is difficult because of the nature of the link function, the signs of the coefficients for covariates and relative values of the coefficients for factor levels can give important insights into the effects of the predictors in the model.

- For covariates, positive (negative) coefficients indicate positive (inverse) relationships between predictors and outcome. An increasing value of a covariate with a positive coefficient corresponds to an increasing rate of damage incidents.
- For factors, a factor level with a greater coefficient indicates greater incidence of damage. The sign of a coefficient for a factor level is dependent upon that factor level's effect relative to the reference category.

You can make the following interpretations based on the parameter estimates:

- Ship type *B [type=2]* has a statistically significantly (*p* value of 0.019) lower damage rate (estimated coefficient of –0.543) than type *A [type=1]*, the reference category. Type *C [type=3]* actually has an estimated parameter lower than *B*, but the variability in *C*'s estimate clouds the effect. See the estimated marginal means for all relations between factor levels.

- Ships constructed between 1965–69 *[construction=65]* and 1970–74 *[construction=70]* have statistically significantly (*p* values <0.001) higher damage rates (estimated coefficients of 0.697 and 0.818, respectively) than those built between 1960–64 *[construction=60]*, the reference category. See the estimated marginal means for all relations between factor levels.
- Ships in operation between 1975–79 *[operation=75]* have statistically significantly (*p* value of 0.012) higher damage rates (estimated coefficient of 0.384) than those in operation between 1960–1974 *[operation=60]*.

Fitting Alternative Models

One problem with the "overdispersed" Poisson regression is that there is no formal way to test it versus the "standard" Poisson regression. However, one suggested formal test to determine whether there is overdispersion is to perform a likelihood ratio test between a "standard" Poisson regression and a negative binomial regression with all other settings equal. If there is no overdispersion in the Poisson regression, then the statistic −2×(log-likelihood for Poisson model − log-likelihood for negative binomial model) should have a mixture distribution with half its probability mass at 0 and the rest in a chi-square distribution with 1 degree of freedom.

1. Select **Fixed value** as the method for estimating the scale parameter. By default, this value is 1.

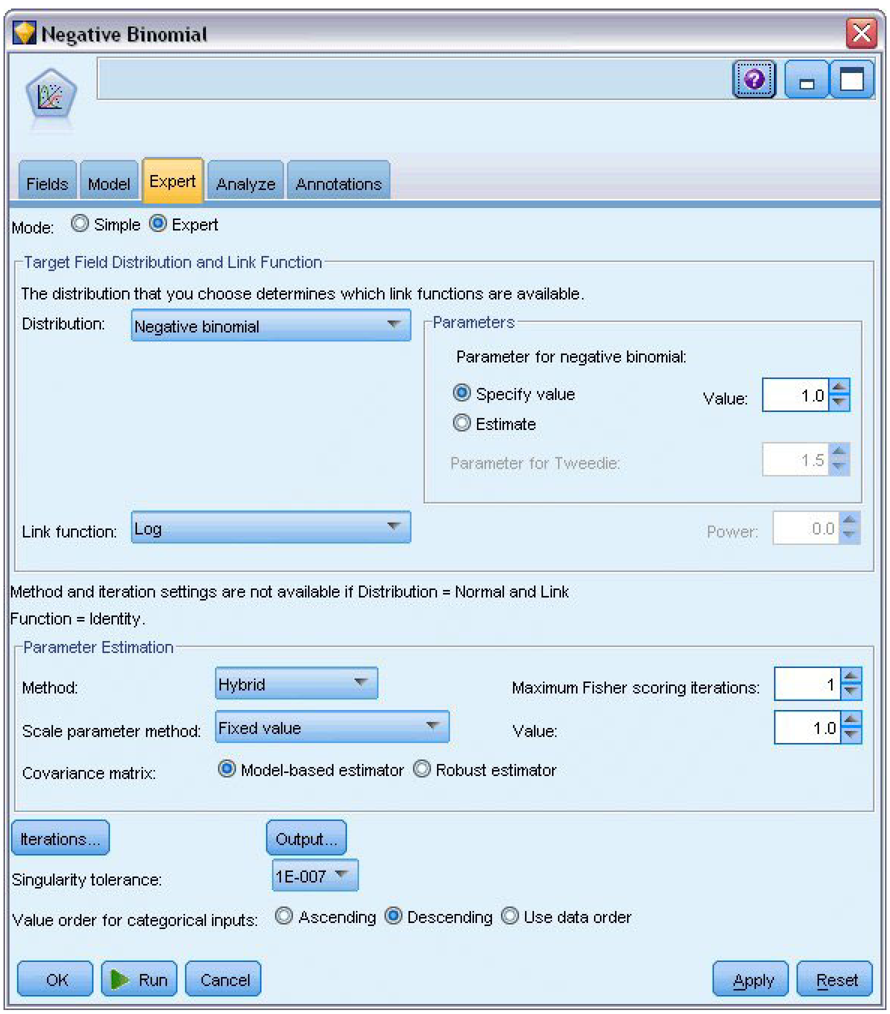


Figure 323. Expert tab

2. To fit the negative binomial regression, copy and paste the Genlin node, attach it to the source node, open the new node and click the **Expert** tab.
3. Select **Negative binomial** as the distribution. Leave the default value of 1 for the ancillary parameter.
4. Run the stream and browse the Advanced tab on the newly-created model nuggets.

Goodness-of-Fit Statistics

	Value	df	Value/df
Deviance	38.695	25	1.548
Scaled Deviance	38.695	25	
Pearson Chi-Square	42.275	25	1.691
Scaled Pearson Chi-Square	42.275	25	
Log Likelihood[a]	-68.281		
Akaike's Information Criterion (AIC)	154.562		
Finite Sample Corrected AIC (AICC)	162.062		
Bayesian Information Criterion (BIC)	168.299		
Consistent AIC (CAIC)	177.299		

Dependent Variable: Number of damage incidents
Model: (Intercept), type, construction, operation, offset = log_months_service

a. The full log likelihood function is displayed and used in computing information criteria.

b. Information criteria are in small-is-better form.

Figure 324. Goodness-of-fit statistics for standard Poisson regression

The log-likelihood reported for the standard Poisson regression is –68.281. Compare this to the negative binomial model.

	Value	df	Value/df
Deviance	11.145	25	.446
Scaled Deviance	11.145	25	
Pearson Chi-Square	8.815	25	.353
Scaled Pearson Chi-Square	8.815	25	
Log Likelihood[a]	-83.725		
Akaike's Information Criterion (AIC)	185.450		
Finite Sample Corrected AIC (AICC)	192.950		
Bayesian Information Criterion (BIC)	199.187		
Consistent AIC (CAIC)	208.187		

Dependent Variable: Number of damage incidents
Model: (Intercept), type, construction, operation, offset = log_months_service

a. The full log likelihood function is displayed and used in computing information criteria.

b. Information criteria are in small-is-better form.

Figure 325. Goodness-of-fit statistics for negative binomial regression

The log-likelihood reported for the negative binomial regression is –83.725. This is actually *smaller* than the log-likelihood for the Poisson regression, which indicates (without the need for a likelihood ratio test) that this negative binomial regression does not offer an improvement over the Poisson regression.

However, the chosen value of 1 for the ancillary parameter of the negative binomial distribution may not be optimal for this dataset. Another way you could test for overdispersion is to fit a negative binomial model with ancillary parameter equal to 0 and request the Lagrange multiplier test on the Output dialog of the Expert tab. If the test is not significant, overdispersion should not be a problem for this dataset.

Summary

Using Generalized Linear Models, you have fit three different models for count data. The negative binomial regression was shown not to offer any improvement over the Poisson regression. The overdispersed Poisson regression seems to offer a reasonable alternative to the standard Poisson model, but there is not a formal test for choosing between them.

Explanations of the mathematical foundations of the modeling methods used in IBM SPSS Modeler are listed in the *IBM SPSS Modeler Algorithms Guide*.

Related Procedures

The Generalized Linear Models procedure is a powerful tool for fitting a variety of models.
- The Generalized Estimating Equations procedure extends the generalized linear model to allow repeated measurements.
- The Linear Mixed Models procedure allows you to fit models for scale dependent variables with a random component and/or repeated measurements.

Recommended Readings

See the following texts for more information on generalized linear models:

Cameron, A. C., and P. K. Trivedi. 1998. *Regression Analysis of Count Data*. Cambridge: Cambridge University Press. Dobson, A. J. 2002. *An Introduction to Generalized Linear Models*, 2 ed. Boca Raton, FL: Chapman & Hall/CRC. Hardin, J. W., and J. M. Hilbe. 2003. *Generalized Linear Models and Extension*. Station, TX: Stata Press. McCullagh, P., and J. A. Nelder. 1989. *Generalized Linear Models*, 2nd ed. London: Chapman & Hall.

Chapter 24. Fitting a Gamma Regression to Car Insurance Claims (Generalized Linear Models)

A generalized linear model can be used to fit a Gamma regression for the analysis of positive range data. For example, a dataset presented and analyzed elsewhere [3] concerns damage claims for cars. The average claim amount can be modeled as having a gamma distribution, using an inverse link function to relate the mean of the dependent variable to a linear combination of the predictors. In order to account for the varying number of claims used to compute the average claim amounts, you specify *Number of claims* as the scaling weight.

This example uses the stream named *car-insurance_genlin.str*, which references the data file named *car_insurance_claims.sav*. The data file is in the *Demos* folder and the stream file is in the *streams* subfolder.

Creating the Stream

1. Add a Statistics File source node pointing to *car_insurance_claims.sav* in the *Demos* folder.

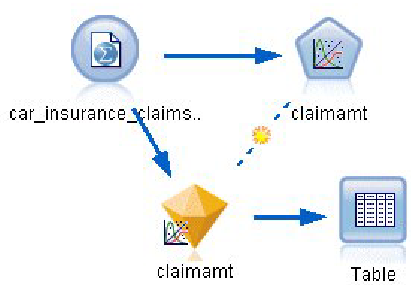


Figure 326. Sample stream to predict car insurance claims

2. On the Types tab of the source node, set the role for the *claimamt* field to **Target**. All other fields should have their role set to **Input**.
3. Click **Read Values** to instantiate the data.

3. McCullagh, P., and J. A. Nelder. 1989. *Generalized Linear Models*, 2nd ed. London: Chapman & Hall.

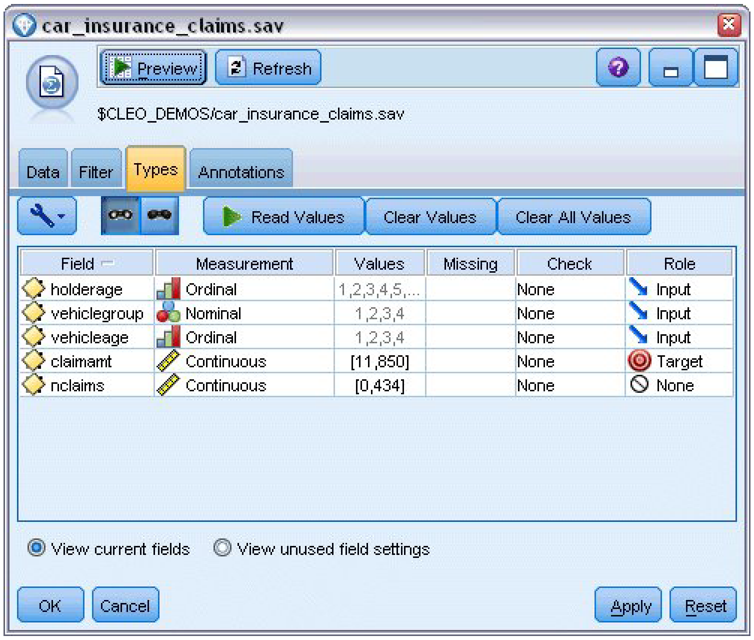


Figure 327. Setting field role

4. Attach a Genlin node to the source node; in the Genlin node, click the Fields tab.
5. Select *nclaims* as the scale weight field.

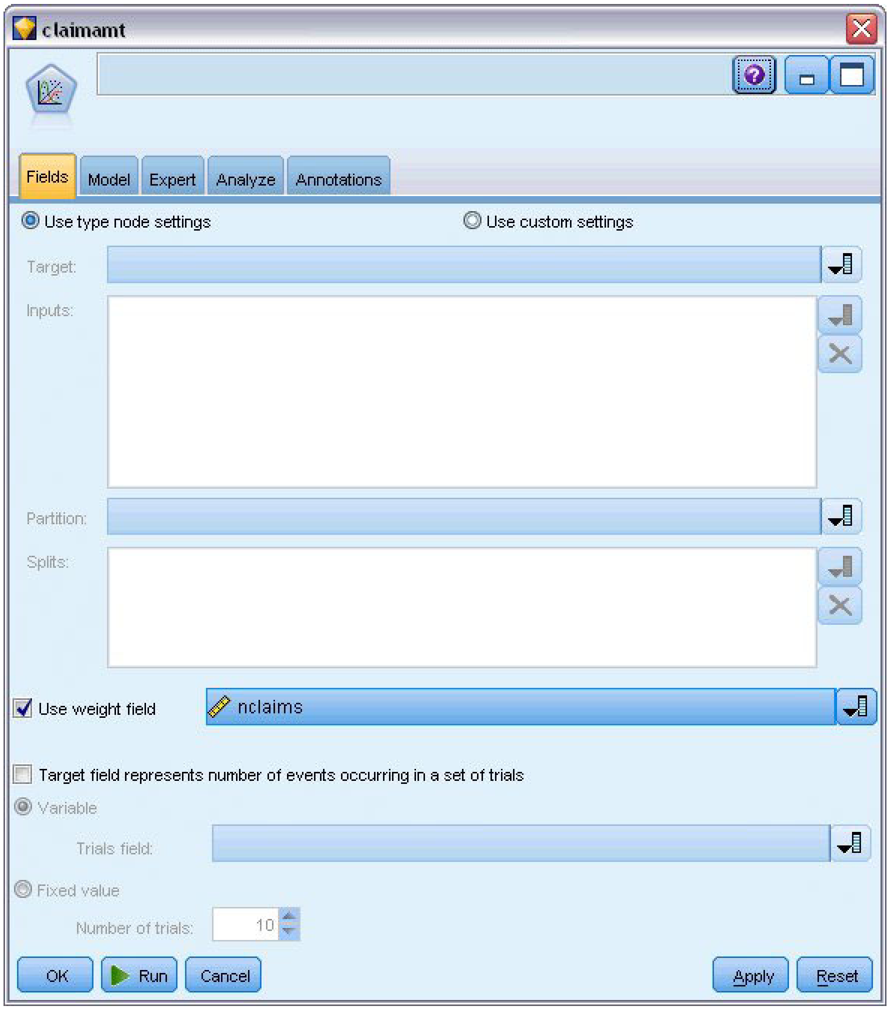


Figure 328. Choosing field options

6. Click the Expert tab and select **Expert** to activate the expert modeling options.

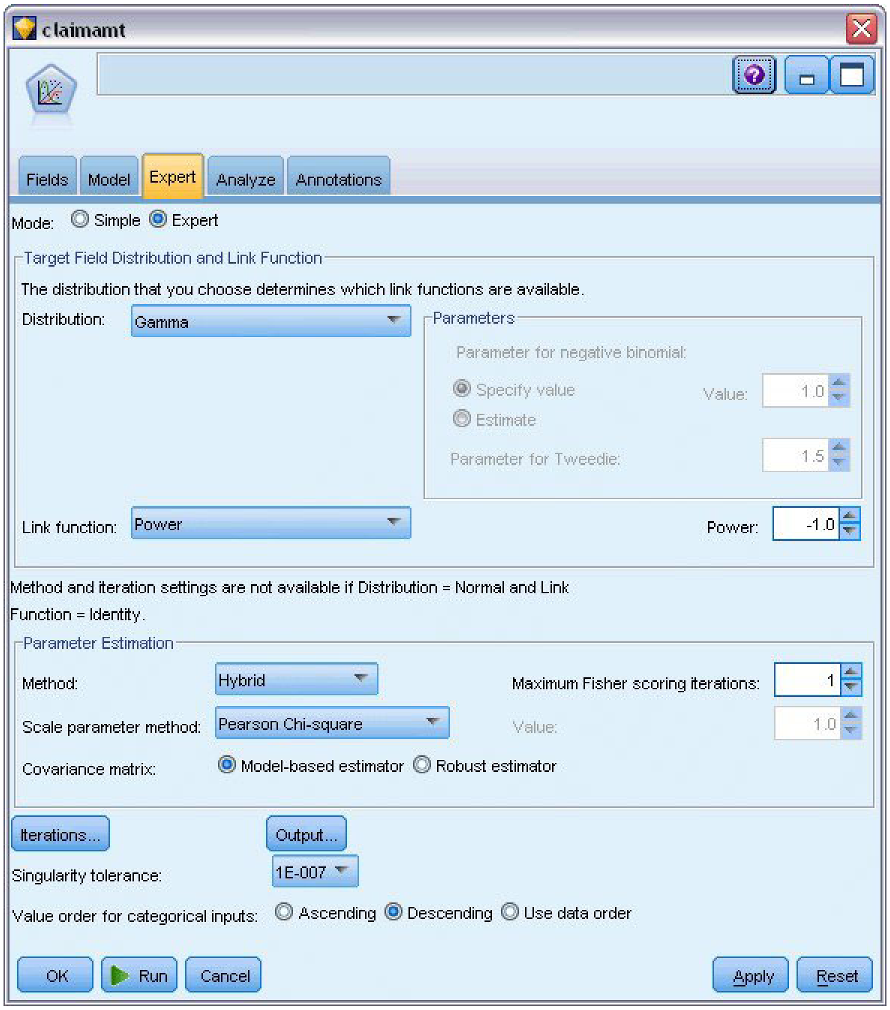


Figure 329. Choosing expert options

7. Select **Gamma** as the response distribution.
8. Select **Power** as the link function and type `-1.0` as the exponent of the power function. This is an inverse link.
9. Select **Pearson chi-square** as the method for estimating the scale parameter. This is the method used by McCullagh and Nelder, so we follow it here in order to replicate their results.
10. Select **Descending** as the category order for factors. This indicates that the first category of each factor will be its reference category; the effect of this selection on the model is in the interpretation of parameter estimates.
11. Click **Run** to create the model nugget, which is added to the stream canvas, and also to the Models palette in the upper-right corner. To view the model details, right-click the model nugget and choose **Edit** or **Browse**, then select the Advanced tab.

Parameter Estimates

Parameter	B	Std. Error	95% Wald Confidence Interval Lower	Upper	Hypothesis Test Wald Chi-Square	df	Sig.
(Intercept)	.003411	.000418	.002591	.004230	66.593	1	.000
[holderage=8]	.000920	.000416	.000105	.001735	4.898	1	.027
[holderage=7]	.000916	.000408	.000117	.001716	5.046	1	.025
[holderage=6]	.000969	.000405	.000176	.001763	5.740	1	.017
[holderage=5]	.001370	.000419	.000548	.002192	10.682	1	.001
[holderage=4]	.000462	.000411	-.000342	.001267	1.268	1	.260
[holderage=3]	.000350	.000412	-.000458	.001158	.720	1	.396
[holderage=2]	.000101	.000436	-.000754	.000956	.054	1	.816
[holderage=1]	.000000[a]	.	.	.	.	.	.
[vehiclegroup=4]	-.001421	.000181	-.001775	-.001067	61.883	1	.000
[vehiclegroup=3]	-.000614	.000170	-.000947	-.000281	13.039	1	.000
[vehiclegroup=2]	.000038	.000169	-.000293	.000368	.050	1	.823
[vehiclegroup=1]	.000000[a]	.	.	.	.	.	.
[vehicleage=4]	.004154	.000442	.003287	.005021	88.175	1	.000
[vehicleage=3]	.001651	.000227	.001207	.002096	53.013	1	.000
[vehicleage=2]	.000366	.000101	.000169	.000564	13.191	1	.000
[vehicleage=1]	.000000[a]	.	.	.	.	.	.
(Scale)	1.209[b]			.001	.0004	.000	.002

Dependent Variable: Average cost of claims
Model: (Intercept), holderage, vehiclegroup, vehicleage

a. Set to zero because this parameter is redundant.

b. Computed based on the Pearson chi-square.

Figure 330. Parameter estimates

The omnibus test and tests of model effects (not shown) indicate that the model outperforms the null model and that each of the main effects terms contribute to the model. The parameter estimates table shows the same values obtained by McCullagh and Nelder for the factor levels and the scale parameter.

Summary

Using Generalized Linear Models, you have fit a gamma regression to the claims data. Note that while the canonical link function for the gamma distribution was used in this model, a log link will also give reasonable results. In general, it is difficult to impossible to directly compare models with different link functions; however, the log link is a special case of the power link where the exponent is 0, thus you can compare the deviances of a model with a log link and a model with a power link to determine which gives the better fit (see, for example, section 11.3 of McCullagh and Nelder).

Explanations of the mathematical foundations of the modeling methods used in IBM SPSS Modeler are listed in the *IBM SPSS Modeler Algorithms Guide*.

Related Procedures

The Generalized Linear Models procedure is a powerful tool for fitting a variety of models.

- The Generalized Estimating Equations procedure extends the generalized linear model to allow repeated measurements.
- The Linear Mixed Models procedure allows you to fit models for scale dependent variables with a random component and/or repeated measurements.

Recommended Readings

See the following texts for more information on generalized linear models:

Cameron, A. C., and P. K. Trivedi. 1998. *Regression Analysis of Count Data*. Cambridge: Cambridge University Press. Dobson, A. J. 2002. *An Introduction to Generalized Linear Models*, 2 ed. Boca Raton, FL: Chapman & Hall/CRC. Hardin, J. W., and J. M. Hilbe. 2003. *Generalized Linear Models and Extension*. Station, TX: Stata Press. McCullagh, P., and J. A. Nelder. 1989. *Generalized Linear Models*, 2nd ed. London: Chapman & Hall.

Chapter 25. Classifying Cell Samples (SVM)

Support Vector Machine (SVM) is a classification and regression technique that is particularly suitable for wide datasets. A wide dataset is one with a large number of predictors, such as might be encountered in the field of bioinformatics (the application of information technology to biochemical and biological data).

A medical researcher has obtained a dataset containing characteristics of a number of human cell samples extracted from patients who were believed to be at risk of developing cancer. Analysis of the original data showed that many of the characteristics differed significantly between benign and malignant samples. The researcher wants to develop an SVM model that can use the values of these cell characteristics in samples from other patients to give an early indication of whether their samples might be benign or malignant.

This example uses the stream named *svm_cancer.str*, available in the *Demos* folder under the *streams* subfolder. The data file is *cell_samples.data*. See the topic “Demos Folder” on page 4 for more information.

The example is based on a dataset that is publicly available from the UCI Machine Learning Repository . The dataset consists of several hundred human cell sample records, each of which contains the values of a set of cell characteristics. The fields in each record are:

Field name	**Description**
ID	Patient identifier
Clump	Clump thickness
UnifSize	Uniformity of cell size
UnifShape	Uniformity of cell shape
MargAdh	Marginal adhesion
SingEpiSize	Single epithelial cell size
BareNuc	Bare nuclei
BlandChrom	Bland chromatin
NormNucl	Normal nucleoli
Mit	Mitoses
Class	Benign or malignant

For the purposes of this example, we're using a dataset that has a relatively small number of predictors in each record.

Creating the Stream

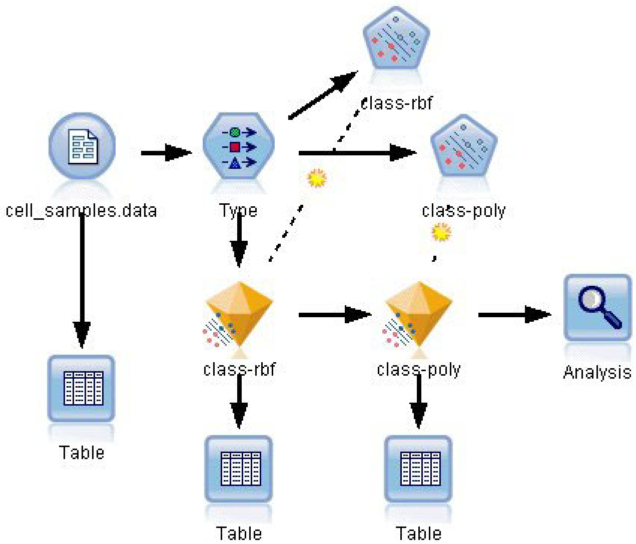
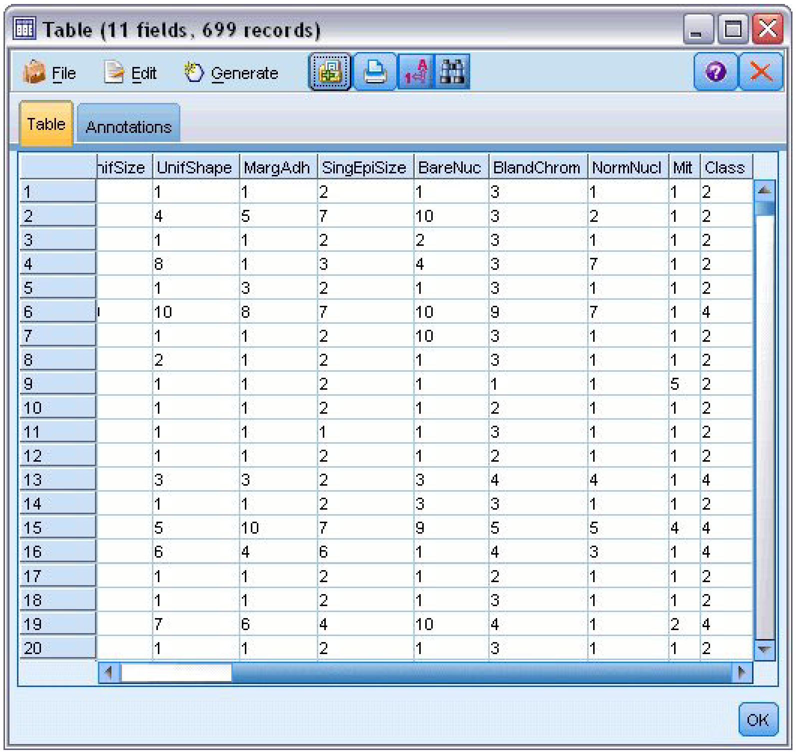


Figure 331. Sample stream to show SVM modeling

1. Create a new stream and add a Var File source node pointing to *cell_samples.data* in the *Demos* folder of your IBM SPSS Modeler installation.

   Let's take a look at the data in the source file.
2. Add a Table node to the stream.
3. Attach the Table node to the Var File node and run the stream.

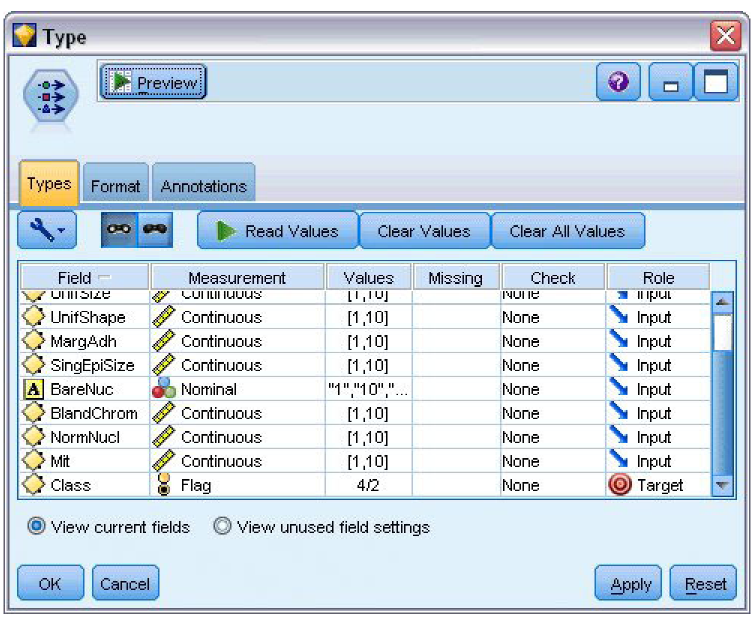


	hifSize	UnifShape	MargAdh	SingEpiSize	BareNuc	BlandChrom	NormNucl	Mit	Class
1		1	1	2	1	3	1	1	2
2		4	5	7	10	3	2	1	2
3		1	1	2	2	3	1	1	2
4		8	1	3	4	3	7	1	2
5		1	3	2	1	3	1	1	2
6		10	8	7	10	9	7	1	4
7		1	1	2	10	3	1	1	2
8		2	1	2	1	3	1	1	2
9		1	1	2	1	1	1	5	2
10		1	1	2	1	2	1	1	2
11		1	1	1	1	3	1	1	2
12		1	1	2	1	2	1	1	2
13		3	3	2	3	4	4	1	4
14		1	1	2	3	3	1	1	2
15		5	10	7	9	5	5	4	4
16		6	4	6	1	4	3	1	4
17		1	1	2	1	2	1	1	2
18		1	1	2	1	3	1	1	2
19		7	6	4	10	4	1	2	4
20		1	1	2	1	3	1	1	2

Figure 332. Source data for SVM

The *ID* field contains the patient identifiers. The characteristics of the cell samples from each patient are contained in fields *Clump* to *Mit*. The values are graded from 1 to 10, with 1 being the closest to benign.

The *Class* field contains the diagnosis, as confirmed by separate medical procedures, as to whether the samples are benign (value = 2) or malignant (value = 4).



Field	Measurement	Values	Missing	Check	Role
UnifSize	Continuous	[1,10]		None	Input
UnifShape	Continuous	[1,10]		None	Input
MargAdh	Continuous	[1,10]		None	Input
SingEpiSize	Continuous	[1,10]		None	Input
BareNuc	Nominal	"1","10","...		None	Input
BlandChrom	Continuous	[1,10]		None	Input
NormNucl	Continuous	[1,10]		None	Input
Mit	Continuous	[1,10]		None	Input
Class	Flag	4/2		None	Target

Figure 333. Type node settings

4. Add a Type node and attach it to the Var File node.

5. Open the Type node.

   We want the model to predict the value of *Class* (that is, benign (=2) or malignant (=4)). As this field can have one of only two possible values, we need to change its measurement level to reflect this.
6. In the **Measurement** column for the *Class* field (the last one in the list), click the value **Continuous** and change it to **Flag**.
7. Click **Read Values**.
8. In the **Role** column, set the role for *ID* (the patient identifier) to **None**, as this will not be used either as a predictor or a target for the model.
9. Set the role for the target, *Class*, to **Target** and leave the role of all the other fields (the predictors) as **Input**.
10. Click **OK**.

    The SVM node offers a choice of kernel functions for performing its processing. As there's no easy way of knowing which function performs best with any given dataset, we'll choose different functions in turn and compare the results. Let's start with the default, RBF (Radial Basis Function).

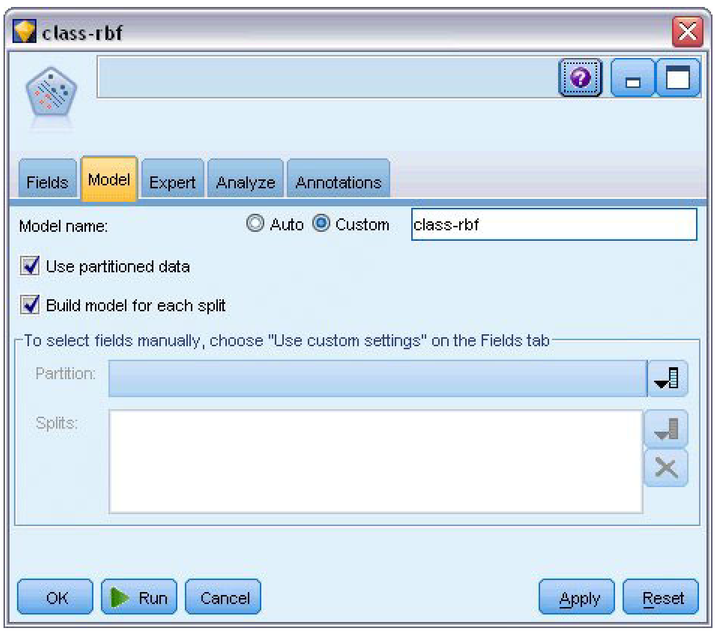


Figure 334. Model tab settings

11. From the Modeling palette, attach an SVM node to the Type node.
12. Open the SVM node. On the **Model** tab, click the **Custom** option for **Model name** and type *class-rbf* in the adjacent text field.

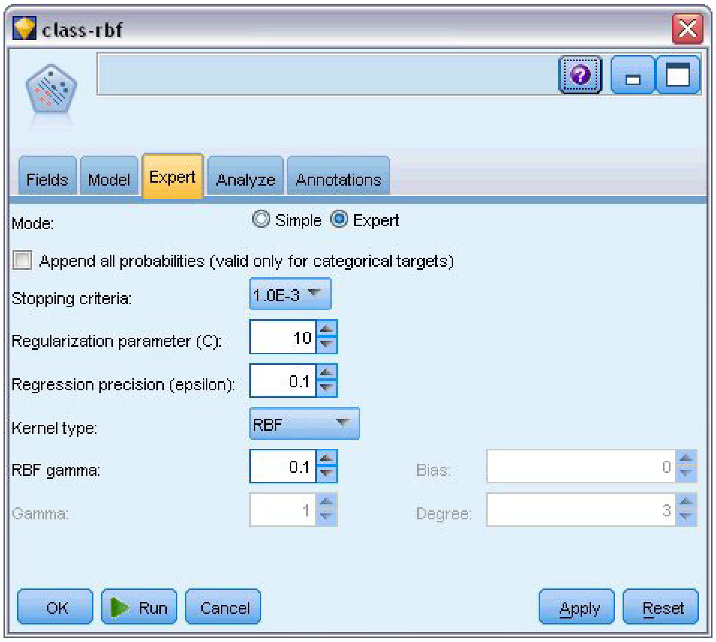


Figure 335. Default Expert tab settings

13. On the **Expert** tab, set the **Mode** to **Expert** for readability but leave all the default options as they are. Note that **Kernel type** is set to **RBF** by default. All the options are greyed out in Simple mode.

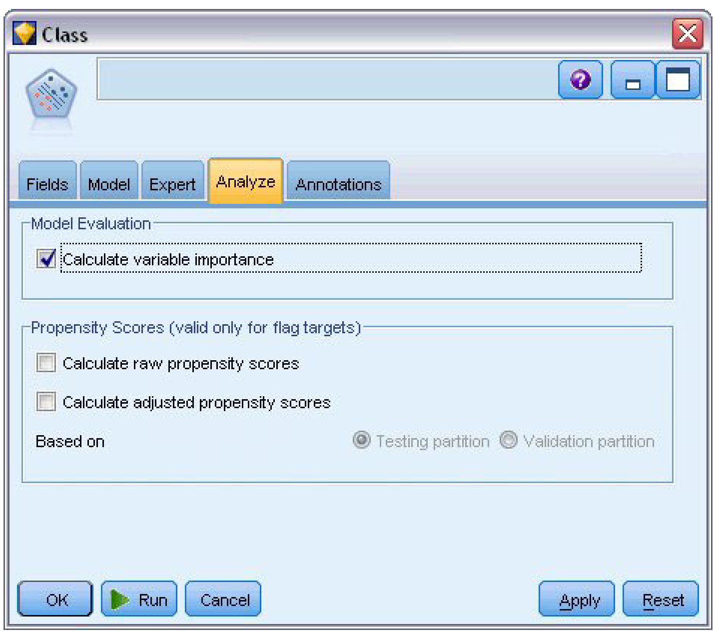


Figure 336. Analyze tab settings

14. On the **Analyze** tab, select the **Calculate variable importance** check box.
15. Click **Run**. The model nugget is placed in the stream, and in the Models palette at the top right of the screen.
16. Double-click the model nugget in the stream.

Examining the Data

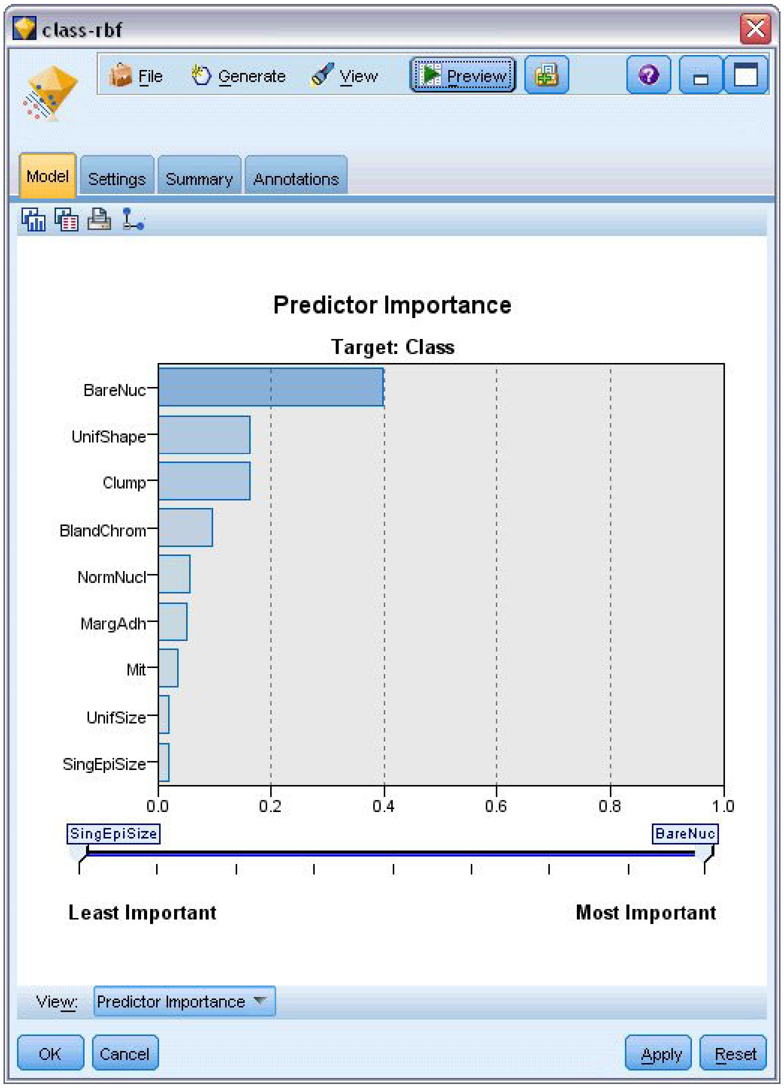


Figure 337. Predictor Importance graph

On the Model tab, the Predictor Importance graph shows the relative effect of the various fields on the prediction. This shows us that *BareNuc* has easily the greatest effect, while *UnifShape* and *Clump* are also quite significant.

1. Click **OK**.
2. Attach a Table node to the *class-rbf* model nugget.
3. Open the Table node and click **Run**.

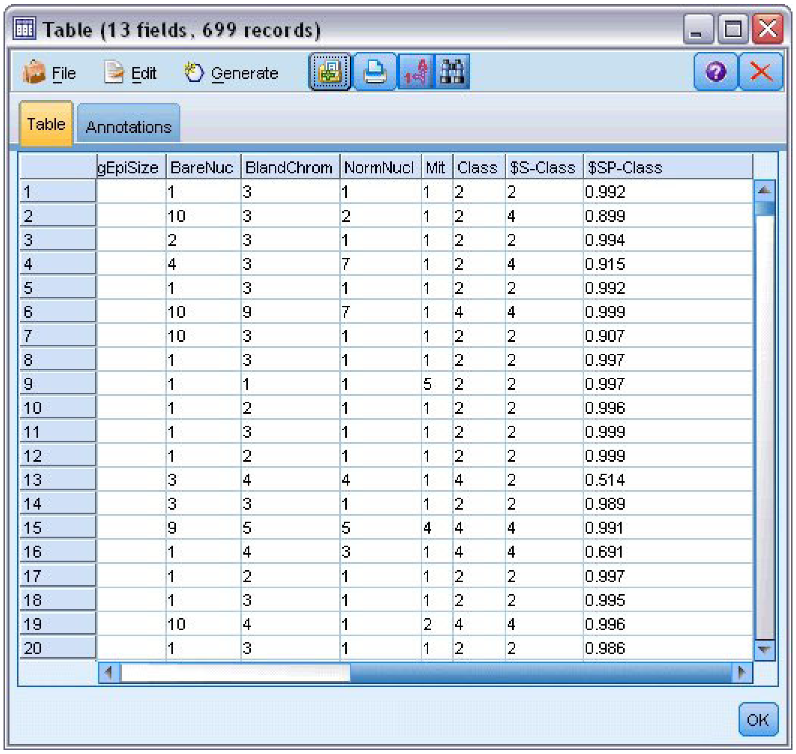


Table (13 fields, 699 records)

	gEpiSize	BareNuc	BlandChrom	NormNucl	Mit	Class	$S-Class	$SP-Class
1		1	3	1	1	2	2	0.992
2		10	3	2	1	2	4	0.899
3		2	3	1	1	2	2	0.994
4		4	3	7	1	2	4	0.915
5		1	3	1	1	2	2	0.992
6		10	9	7	1	4	4	0.999
7		10	3	1	1	2	2	0.907
8		1	3	1	1	2	2	0.997
9		1	1	1	5	2	2	0.997
10		1	2	1	1	2	2	0.996
11		1	3	1	1	2	2	0.999
12		1	2	1	1	2	2	0.999
13		3	4	4	1	4	2	0.514
14		3	3	1	1	2	2	0.989
15		9	5	5	4	4	4	0.991
16		1	4	3	1	4	4	0.691
17		1	2	1	1	2	2	0.997
18		1	3	1	1	2	2	0.995
19		10	4	1	2	4	4	0.996
20		1	3	1	1	2	2	0.986

Figure 338. Fields added for prediction and confidence value

4. The model has created two extra fields. Scroll the table output to the right to see them:

New field name	Description
$S-Class	Value for *Class* predicted by the model.
$SP-Class	Propensity score for this prediction (the likelihood of this prediction being true, a value from 0.0 to 1.0).

Just by looking at the table, we can see that the propensity scores (in the *$SP-Class* column) for most of the records are reasonably high.

However, there are some significant exceptions; for example, the record for patient 1041801 at line 13, where the value of 0.514 is unacceptably low. Also, comparing *Class* with *$S-Class*, it's clear that this model has made a number of incorrect predictions, even where the propensity score was relatively high (for example, lines 2 and 4).

Let's see if we can do better by choosing a different function type.

Trying a Different Function

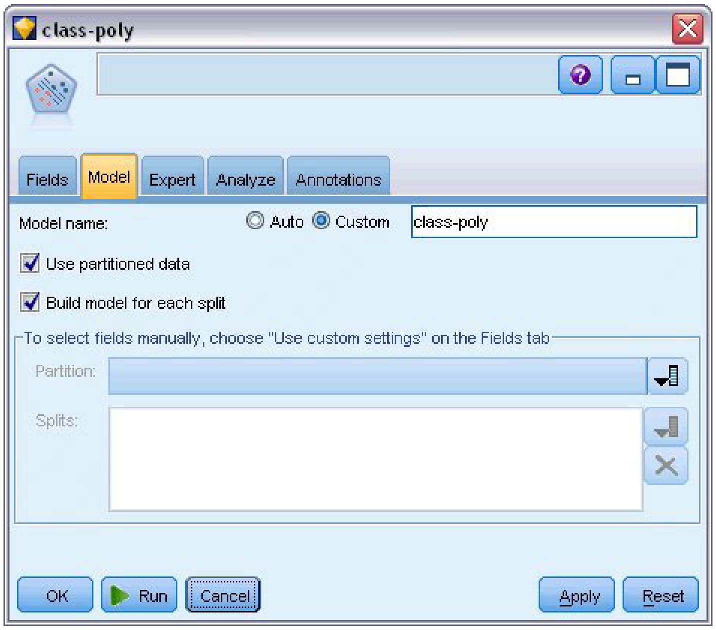


Figure 339. Setting a new name for the model

1. Close the Table output window.
2. Attach a second SVM modeling node to the Type node.
3. Open the new SVM node.
4. On the **Model** tab, choose Custom and type *class-poly* as the model name.

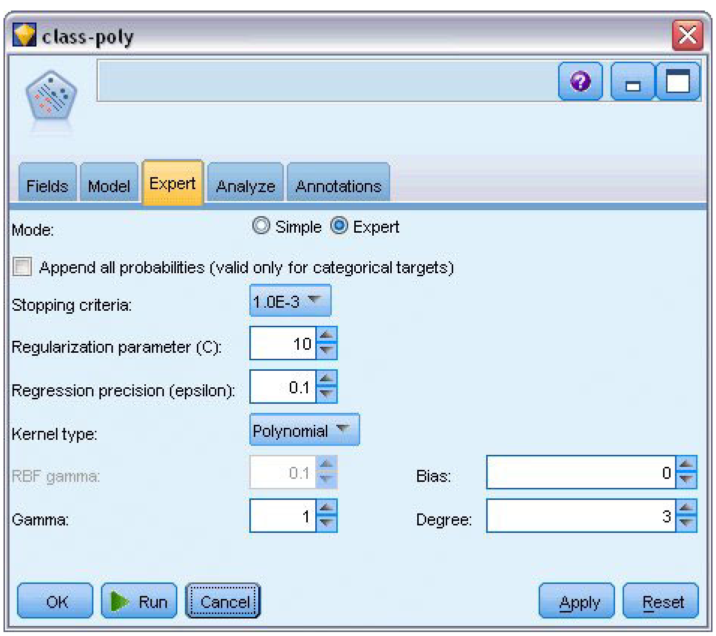
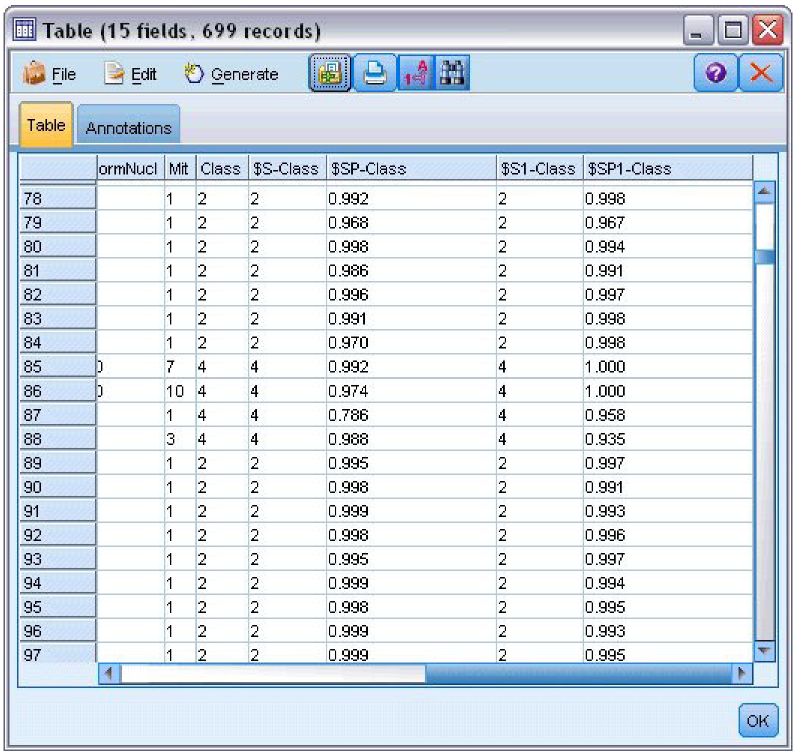


Figure 340. Expert tab settings for Polynomial

5. On the **Expert** tab, set **Mode** to **Expert**.

6. Set **Kernel type** to **Polynomial** and click **Run**. The *class-poly* model nugget is added to the stream, and also to the Models palette at the top right of the screen.
7. Connect the *class-rbf* model nugget to the *class-poly* model nugget (choose **Replace** at the warning dialog).
8. Attach a Table node to the *class-poly* nugget.
9. Open the Table node and click **Run**.

Comparing the Results



	ormNucl	Mit	Class	$S-Class	$SP-Class	$S1-Class	$SP1-Class
78		1	2	2	0.992	2	0.998
79		1	2	2	0.968	2	0.967
80		1	2	2	0.998	2	0.994
81		1	2	2	0.986	2	0.991
82		1	2	2	0.996	2	0.997
83		1	2	2	0.991	2	0.998
84		1	2	2	0.970	2	0.998
85	[illegible]	7	4	4	0.992	4	1.000
86	[illegible]	10	4	4	0.974	4	1.000
87		1	4	4	0.786	4	0.958
88		3	4	4	0.988	4	0.935
89		1	2	2	0.995	2	0.997
90		1	2	2	0.998	2	0.991
91		1	2	2	0.999	2	0.993
92		1	2	2	0.998	2	0.996
93		1	2	2	0.995	2	0.997
94		1	2	2	0.999	2	0.994
95		1	2	2	0.998	2	0.995
96		1	2	2	0.999	2	0.993
97		1	2	2	0.999	2	0.995

Figure 341. Fields added for Polynomial function

1. Scroll the table output to the right to see the newly added fields.

   The generated fields for the Polynomial function type are named *$S1-Class* and *$SP1-Class*.

   The results for Polynomial look much better. Many of the propensity scores are 0.995 or better, which is very encouraging.
2. To confirm the improvement in the model, attach an Analysis node to the *class-poly* model nugget.

Open the Analysis node and click **Run**.

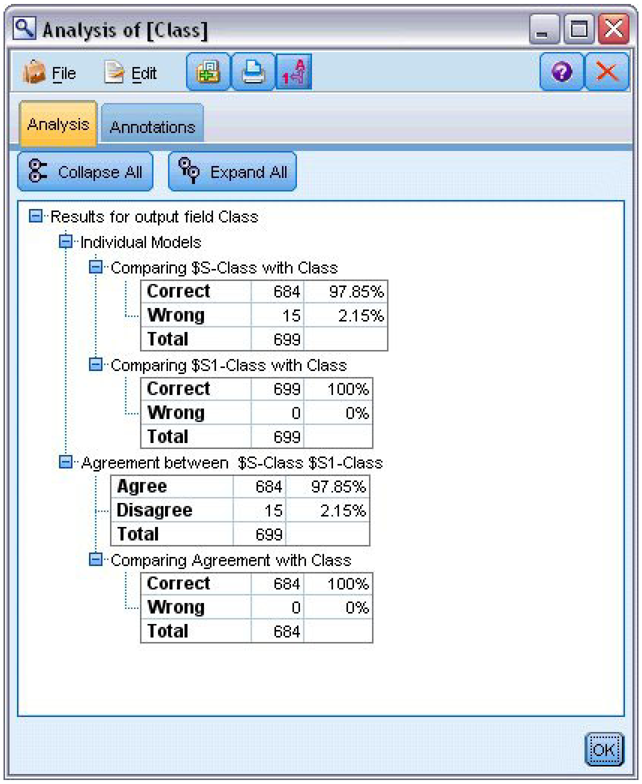


Figure 342. Analysis node

This technique with the Analysis node enables you to compare two or more model nuggets of the same type. The output from the Analysis node shows that the RBF function correctly predicts 97.85% of the cases, which is still quite good. However, the output shows that the Polynomial function has correctly predicted the diagnosis in every single case. In practice you are unlikely to see 100% accuracy, but you can use the Analysis node to help determine whether the model is acceptably accurate for your particular application.

In fact, neither of the other function types (Sigmoid and Linear) performs as well as Polynomial on this particular dataset. However, with a different dataset, the results could easily be different, so it's always worth trying the full range of options.

Summary

You have used different types of SVM kernel functions to predict a classification from a number of attributes. You have seen how different kernels give different results for the same dataset and how you can measure the improvement of one model over another.

Chapter 26. Using Cox Regression to Model Customer Time to Churn

As part of its efforts to reduce customer churn, a telecommunications company is interested in modeling the "time to churn" in order to determine the factors that are associated with customers who are quick to switch to another service. To this end, a random sample of customers is selected and their time spent as customers, whether they are still active customers, and various other fields are pulled from the database.

This example uses the stream *telco_coxreg.str*, which references the data file *telco.sav*. The data file is in the *Demos* folder and the stream file is in the *streams* subfolder. See the topic “Demos Folder” on page 4 for more information.

Building a Suitable Model

1. Add a Statistics File source node pointing to *telco.sav* in the *Demos* folder.

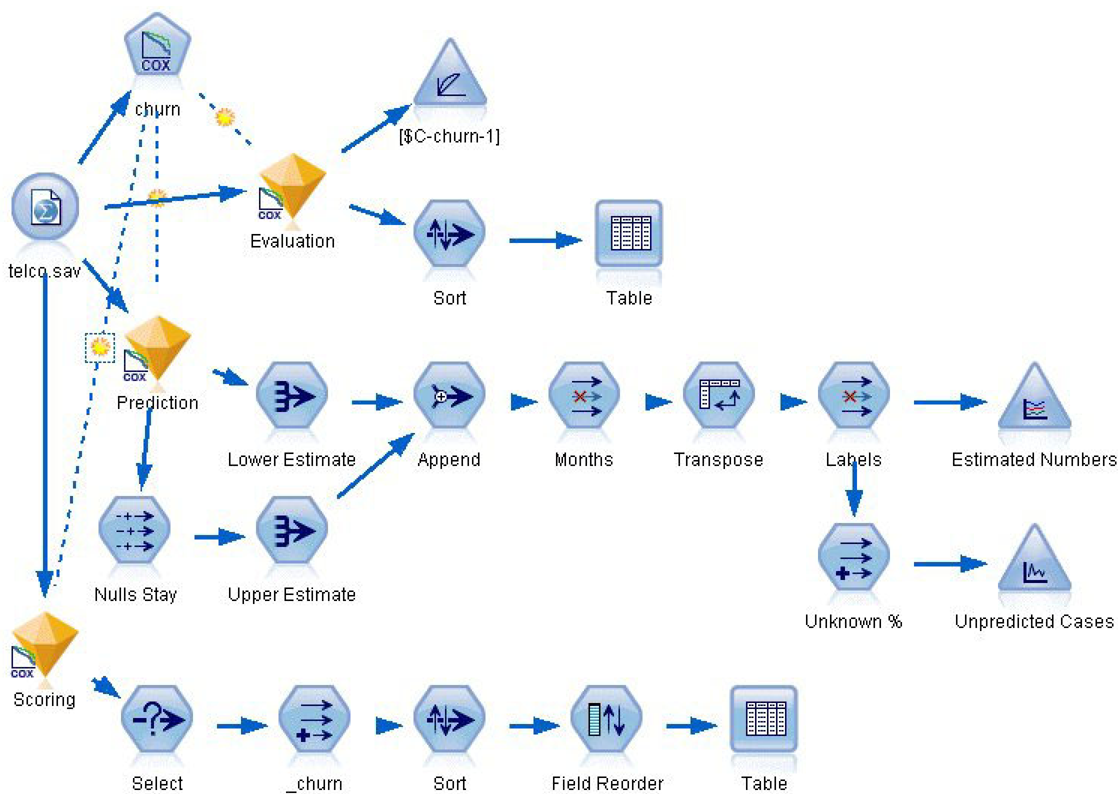


Figure 343. Sample stream to analyze time to churn

2. On the Filter tab of the source node, exclude the fields *region*, *income*, *longten* through *wireten*, and *loglong* through *logwire*.

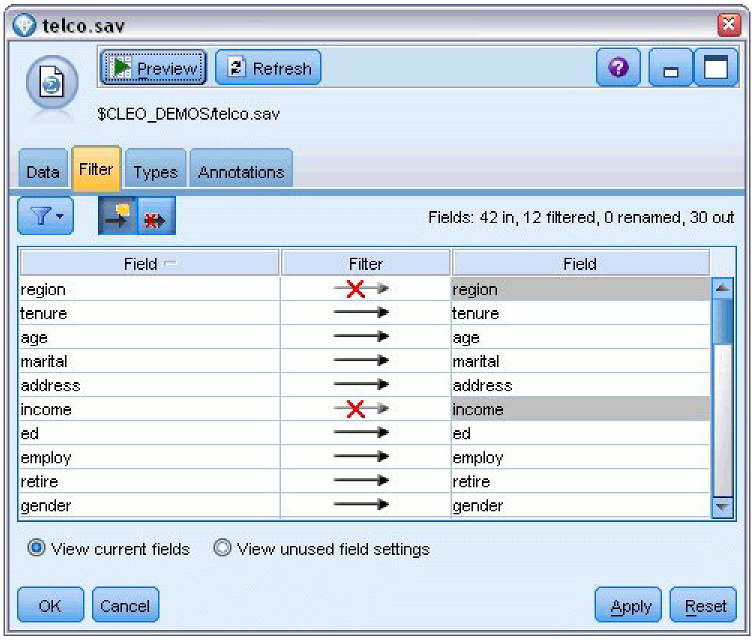


Figure 344. Filtering unneeded fields

(Alternatively, you could change the role to **None** for these fields on the Types tab rather than exclude it, or select the fields you want to use in the modeling node.)

3. On the Types tab of the source node, set the role for the *churn* field to **Target** and set its measurement level to **Flag**. All other fields should have their role set to **Input**.
4. Click **Read Values** to instantiate the data.

Figure 345. Setting field role

5. Attach a Cox node to the source node; in the **Fields** tab, select *tenure* as the survival time variable.

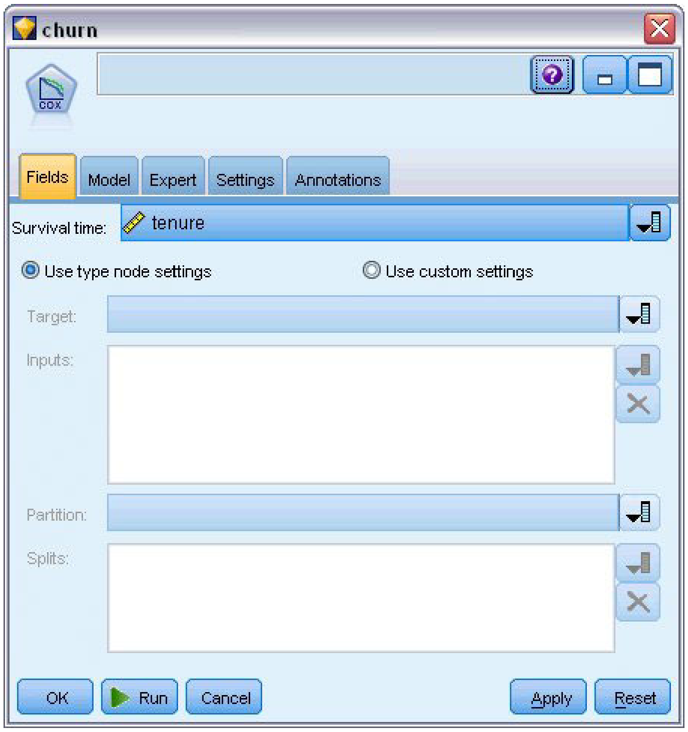


Figure 346. Choosing field options

6. Click the **Model** tab.
7. Select **Stepwise** as the variable selection method.

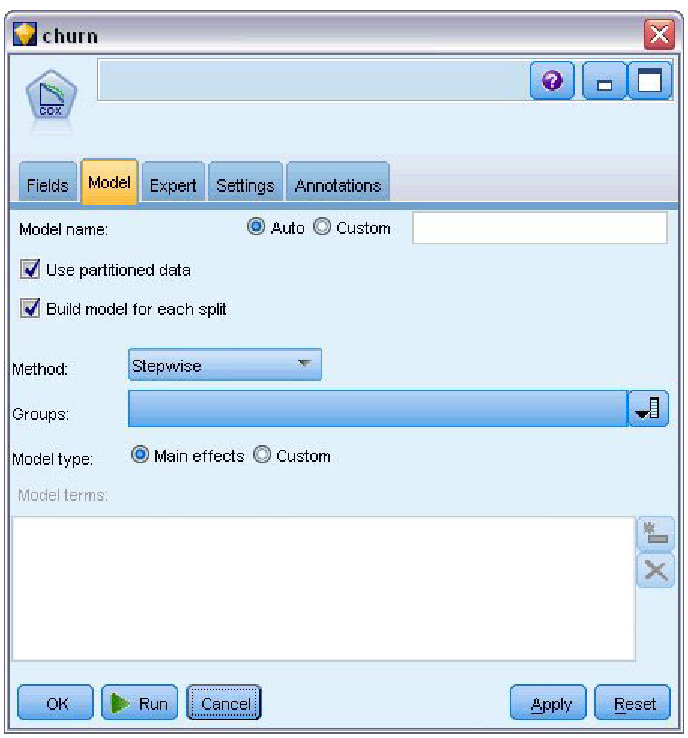


Figure 347. Choosing model options

8. Click the **Expert** tab and select **Expert** to activate the expert modeling options.

9. Click **Output**.

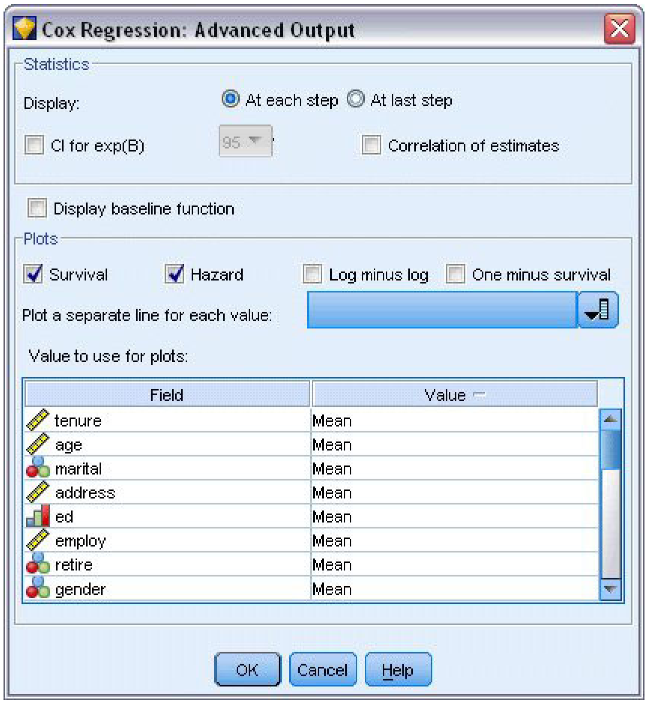


Figure 348. Choosing advanced output options

10. Select **Survival** and **Hazard** as plots to produce, then click **OK**.
11. Click **Run** to create the model nugget, which is added to the stream, and to the Models palette in the upper right corner. To view its details, double-click the nugget on the stream. First, look at the Advanced output tab.

Censored Cases

		N	Percent
Cases available in analysis	Event[a]	274	27.4%
	Censored	726	72.6%
	Total	1000	100.0%
Cases dropped	Cases with missing values	0	.0%
	Cases with negative time	0	.0%
	Censored cases before the earliest event in a stratum	0	.0%
	Total	0	.0%
Total		1000	100.0%

a. Dependent Variable: Months with service

Figure 349. Case processing summary

The status variable identifies whether the event has occurred for a given case. If the event has not occurred, the case is said to be censored. Censored cases are not used in the computation of the regression coefficients but are used to compute the baseline hazard. The case processing summary shows that 726 cases are censored. These are customers who have not churned.

Categorical Variable Codings

		Frequency	(1)[b]	(2)	(3)	(4)
marital[a]	0=Unmarried	505	1			
	1=Married	495	0			
ed[a]	1=Did not complete high school	204	1	0	0	0
	2=High school degree	287	0	1	0	0
	3=Some college	209	0	0	1	0
	4=College degree	234	0	0	0	1
	5=Post-undergraduate degree	66	0	0	0	0
retire[a]	.00=No	953	1			
	1.00=Yes	47	0			
gender[a]	0=Male	483	1			
	1=Female	517	0			
tollfree[a]	0=No	526	1			
	1=Yes	474	0			
equip[a]	0=No	614	1			
	1=Yes	386	0			
callcard[a]	0=No	322	1			
	1=Yes	678	0			
wireless[a]	0=No	704	1			
	1=Yes	296	0			
multline[a]	0=No	525	1			
	1=Yes	475	0			
voice[a]	0=No	696	1			
	1=Yes	304	0			
pager[a]	0=No	739	1			
	1=Yes	261	0			
internet[a]	0=No	632	1			
	1=Yes	368	0			
callid[a]	0=No	519	1			
	1=Yes	481	0			
callwait[a]	0=No	515	1			
	1=Yes	485	0			
forward[a]	0=No	507	1			
	1=Yes	493	0			
confer[a]	0=No	498	1			
	1=Yes	502	0			
ebill[a]	0=No	629	1			
	1=Yes	371	0			
custcat[a]	1=Basic service	266	1	0	0	
	2=E-service	217	0	1	0	
	3=Plus service	281	0	0	1	
	4=Total service	236	0	0	0	

Figure 350. Categorical variable codings

The categorical variable codings are a useful reference for interpreting the regression coefficients for categorical covariates, particularly dichotomous variables. By default, the reference category is the "last" category. Thus, for example, even though *Married* customers have variable values of 1 in the data file, they are coded as 0 for the purposes of the regression.

Variable Selection

Step	-2 Log Likelihood	Overall (score) Chi-square			Change From Previous Step Chi-square			Change From Previous Block Chi-square		
		Chi-square	df	Sig.	Chi-square	df	Sig.	Chi-square	df	Sig.
1[a]	3392.536	162.303	1	.000	133.828	1	.000	133.828	1	.000
2[b]	3087.314	249.392	2	.000	305.222	1	.000	439.050	2	.000
3[c]	3027.085	328.426	3	.000	60.229	1	.000	499.279	3	.000
4[d]	2990.790	347.197	4	.000	36.294	1	.000	535.574	4	.000
5[e]	2973.790	362.673	5	.000	17.000	1	.000	552.574	5	.000
6[f]	2958.796	376.140	6	.000	14.994	1	.000	567.568	6	.000
7[g]	2945.503	384.717	7	.000	13.293	1	.000	580.861	7	.000
8[h]	2936.993	417.341	8	.000	8.510	1	.004	589.371	8	.000
9[i]	2926.000	423.911	9	.000	10.994	1	.001	600.364	9	.000
10[j]	2917.551	428.078	10	.000	8.449	1	.004	608.813	10	.000
11[k]	2913.308	436.837	11	.000	4.243	1	.039	613.056	11	.000
12[l]	2908.078	440.158	12	.000	5.230	1	.022	618.286	12	.000

a. Variable(s) Entered at Step Number 1: callcard
b. Variable(s) Entered at Step Number 2: longmon
c. Variable(s) Entered at Step Number 3: equip
d. Variable(s) Entered at Step Number 4: employ
e. Variable(s) Entered at Step Number 5: multline
f. Variable(s) Entered at Step Number 6: voice
g. Variable(s) Entered at Step Number 7: address
h. Variable(s) Entered at Step Number 8: equipmon
i. Variable(s) Entered at Step Number 9: ebill
j. Variable(s) Entered at Step Number 10: callid
k. Variable(s) Entered at Step Number 11: internet
l. Variable(s) Entered at Step Number 12: reside
m. Beginning Block Number 0, initial Log Likelihood function: -2 Log likelihood: 3526.364
n. Beginning Block Number 1. Method = Forward Stepwise (Likelihood Ratio)

Figure 351. Omnibus tests

The model-building process employs a forward stepwise algorithm. The omnibus tests are measures of how well the model performs. The chi-square change from previous step is the difference between the −2 log-likelihood of the model at the previous step and the current step. If the step was to add a variable, the inclusion makes sense if the significance of the change is less than 0.05. If the step was to remove a variable, the exclusion makes sense if the significance of the change is greater than 0.10. In twelve steps, twelve variables are added to the model.

		B	SE	Wald	df	Sig.	Exp(B)
Step 12	address	-.035	.009	14.543	1	.000	.966
	employ	-.051	.010	25.767	1	.000	.950
	reside	-.103	.046	5.037	1	.025	.902
	equip	-1.948	.381	26.180	1	.000	.143
	callcard	.777	.151	26.451	1	.000	2.175
	longmon	-.233	.022	115.619	1	.000	.792
	equipmon	-.042	.011	15.377	1	.000	.959
	multline	.612	.145	17.854	1	.000	1.844
	voice	-.501	.157	10.197	1	.001	.606
	internet	-.362	.160	5.114	1	.024	.697
	callid	-.464	.148	9.790	1	.002	.629
	ebill	-.399	.156	6.557	1	.010	.671

Figure 352. Variables in the equation (step 12 only)

The final model includes *address*, *employ*, *reside*, *equip*, *callcard*, *longmon*, *equipmon*, *multline*, *voice*, *internet*, *callid*, and *ebill*. To understand the effects of individual predictors, look at Exp(B), which can be interpreted as the predicted change in the hazard for a unit increase in the predictor.

- The value of Exp(B) for *address* means that the churn hazard is reduced by 100%−(100%×0.966)=3.4% for each year a customer has lived at the same address. The churn hazard for a customer who has lived at the same address for five years is reduced by $100\%-(100\%\times0.966^5)=15.88\%$.

- The value of Exp(B) for *callcard* means that the churn hazard for a customer who does not subscribe to the calling card service is 2.175 times that of a customer with the service. Recall from the categorical variable codings that *No* = 1 for the regression.
- The value of Exp(B) for *internet* means that the churn hazard for a customer who does not subscribe to the internet service is 0.697 times that of a customer with the service. This is somewhat worrisome because it suggests that customers with the service are leaving the company faster than customers without the service.

		Score	df	Sig.
Step 12	age	.122	1	.726
	marital	.648	1	.421
	income	1.476	1	.224
	ed	6.328	4	.176
	ed(1)	.007	1	.934
	ed(2)	.203	1	.652
	ed(3)	.835	1	.361
	ed(4)	5.773	1	.016
	retire	.013	1	.908
	gender	.214	1	.644
	tollfree	3.243	1	.072
	wireless	.668	1	.414
	tollmon	.000	1	.987
	cardmon	3.163	1	.075
	wiremon	1.084	1	.298
	pager	1.808	1	.179
	callwait	.266	1	.606
	forward	2.201	1	.138
	confer	2.568	1	.109
	custcat	.864	3	.834
	custcat(1)	.466	1	.495
	custcat(2)	.450	1	.502
	custcat(3)	.019	1	.889

Figure 353. Variables not in the model (step 12 only)

Variables left out of the model all have score statistics with significance values greater than 0.05. However, the significance values for *tollfree* and *cardmon*, while not less than 0.05, are fairly close. They may be interesting to pursue in further studies.

Covariate Means

	Mean
age	41.684
marital	.505
address	11.551
income	77.535
ed(1)	.204
ed(2)	.287
ed(3)	.209
ed(4)	.234
employ	10.987
retire	.953
gender	.483
reside	2.331
tollfree	.526
equip	.614
callcard	.322
wireless	.704
longmon	11.723
tollmon	13.274
equipmon	14.220
cardmon	13.781
wiremon	11.584
multline	.525
voice	.696
pager	.739
internet	.632
callid	.519
callwait	.515
forward	.507
confer	.498
ebill	.629
custcat(1)	.266
custcat(2)	.217
custcat(3)	.281

Figure 354. Covariate means

This table displays the average value of each predictor variable. This table is a useful reference when looking at the survival plots, which are constructed for the mean values. Note, however, that the "average" customer doesn't actually exist when you look at the means of indicator variables for categorical predictors. Even with all scale predictors, you are unlikely to find a customer whose covariate values are all close to the mean. If you want to see the survival curve for a particular case, you can change the covariate values at which the survival curve is plotted in the Plots dialog box. If you want to see the survival curve for a particular case, you can change the covariate values at which the survival curve is plotted in the Plots group of the Advanced Output dialog.

Survival Curve

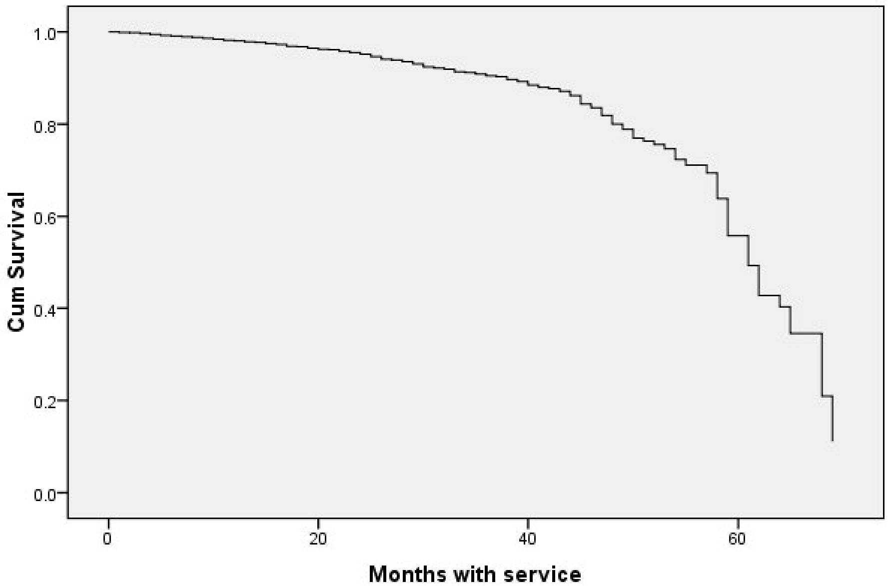


Figure 355. Survival curve for "average" customer

The basic survival curve is a visual display of the model-predicted time to churn for the "average" customer. The horizontal axis shows the time to event. The vertical axis shows the probability of survival. Thus, any point on the survival curve shows the probability that the "average" customer will remain a customer past that time. Past 55 months, the survival curve becomes less smooth. There are fewer customers who have been with the company for that long, so there is less information available, and thus the curve is blocky.

Hazard Curve

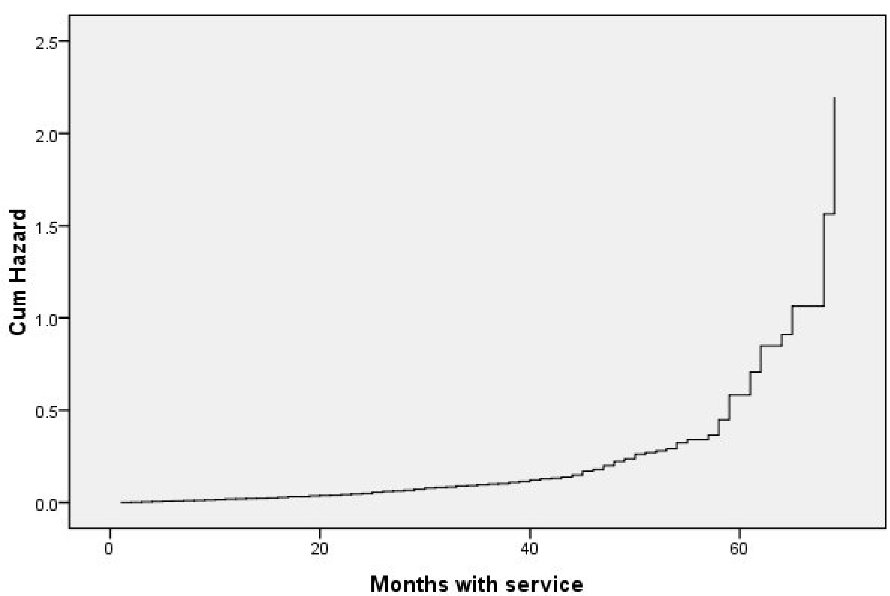


Figure 356. Hazard curve for "average" customer

The basic hazard curve is a visual display of the cumulative model-predicted potential to churn for the "average" customer. The horizontal axis shows the time to event. The vertical axis shows the cumulative hazard, equal to the negative log of the survival probability. Past 55 months, the hazard curve, like the survival curve, becomes less smooth, for the same reason.

Evaluation

The stepwise selection methods guarantee that your model will have only "statistically significant" predictors, but it does not guarantee that the model is actually good at predicting the target. To do this, you need to analyze scored records.

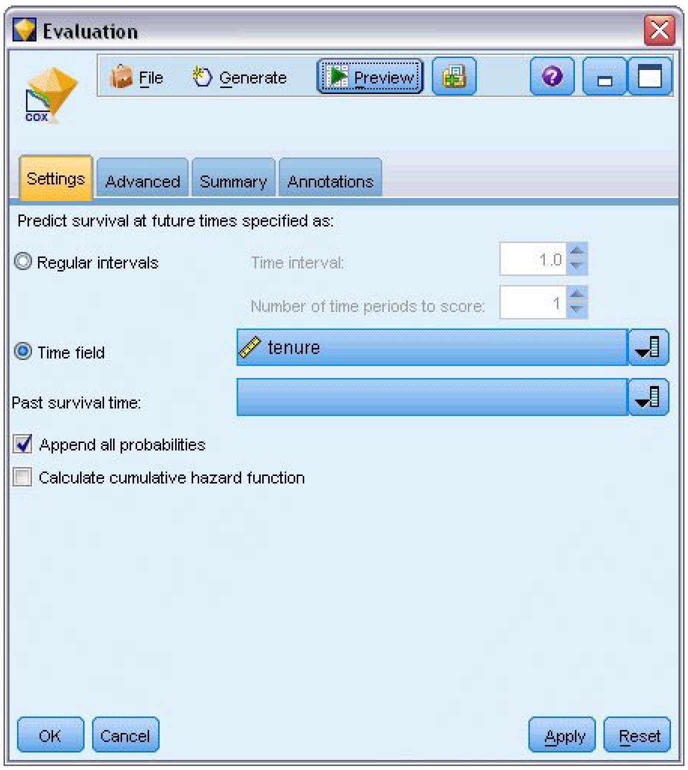


Figure 357. Cox nugget: Settings tab

1. Place the model nugget on the canvas and attach it to the source node, open the nugget and click the Settings tab.
2. Select **Time field** and specify *tenure*. Each record will be scored at its length of tenure.
3. Select **Append all probabilities**.

   This creates scores using 0.5 as the cutoff for whether a customer churns; if their propensity to churn is greater than 0.5, they are scored as a churner. There is nothing magical about this number, and a different cutoff may yield more desirable results. For one way to think about choosing a cutoff, use an Evaluation node.

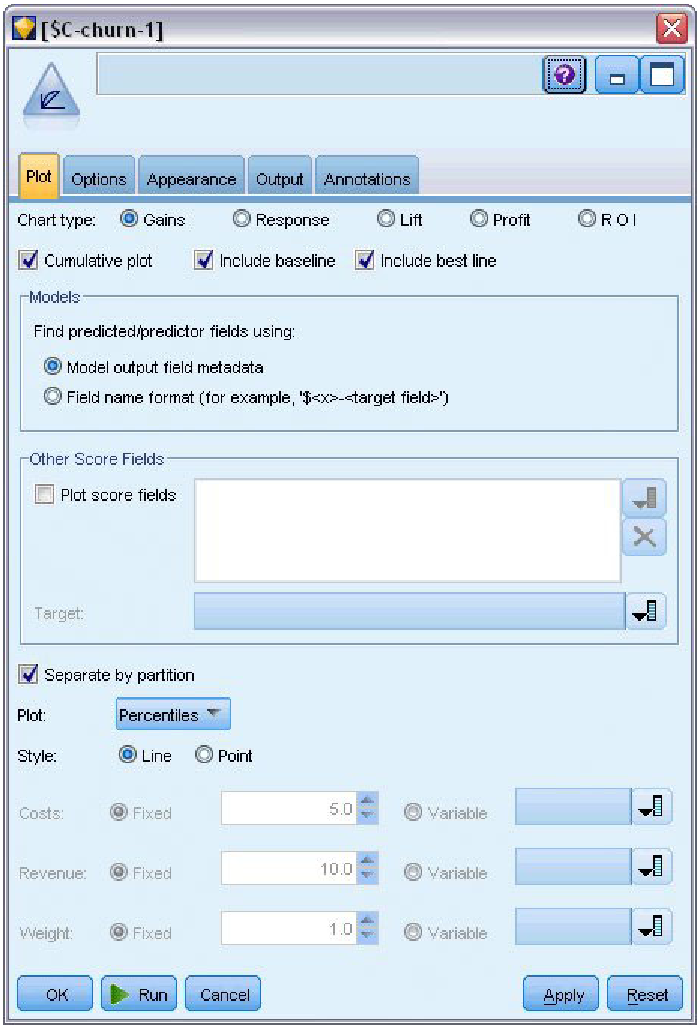


Figure 358. Evaluation node: Plot tab

4. Attach an Evaluation node to the model nugget; on the Plot tab, select **Include best line**.
5. Click the **Options** tab.

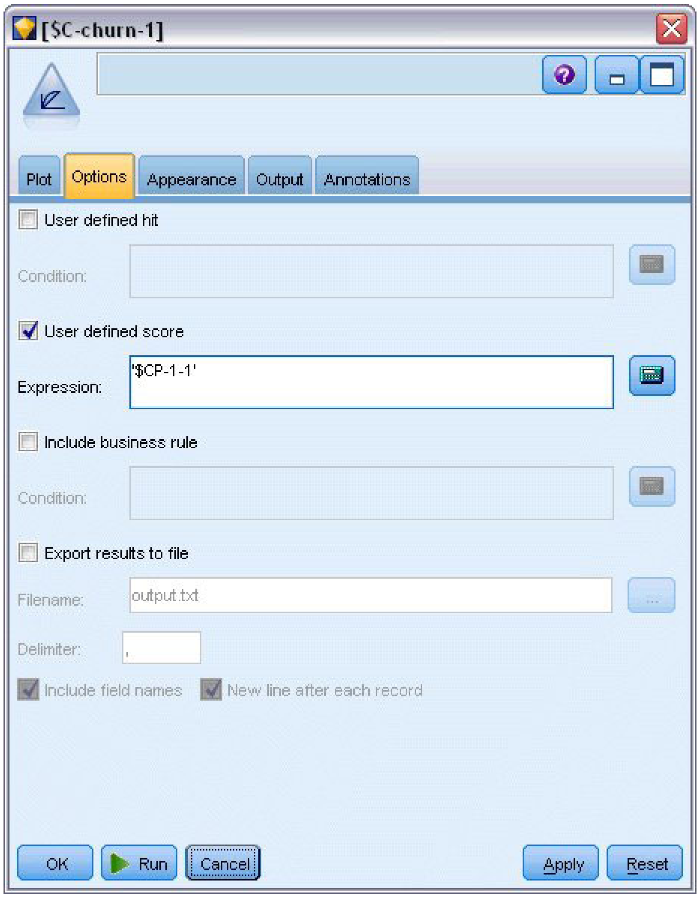


Figure 359. Evaluation node: Options tab

6. Select **User defined score** and type `'$CP-1-1'` as the expression. This is a model-generated field that corresponds to the propensity to churn.
7. Click **Run**.

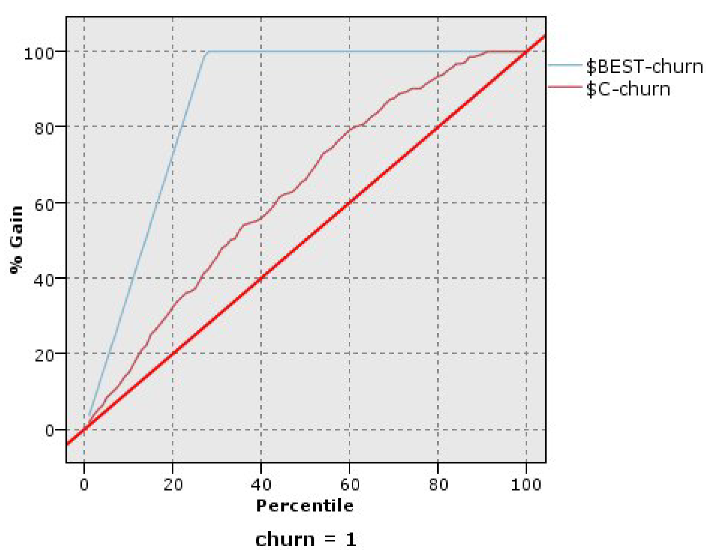


Figure 360. Gains chart

The cumulative gains chart shows the percentage of the overall number of cases in a given category "gained" by targeting a percentage of the total number of cases. For example, one point on the curve is at (10%, 15%), meaning that if you score a dataset with the model and sort all of the cases by predicted propensity to churn, you would expect the top 10% to contain approximately 15% of all of the cases that actually take the category *1* (churners). Likewise, the top 60% contains approximately 79.2% of the churners. If you select 100% of the scored dataset, you obtain all of the churners in the dataset.

The diagonal line is the "baseline" curve; if you select 20% of the records from the scored dataset at random, you would expect to "gain" approximately 20% of all of the records that actually take the category *1*. The farther above the baseline a curve lies, the greater the gain. The "best" line shows the curve for a "perfect" model that assigns a higher churn propensity score to every churner than every non-churner. You can use the cumulative gains chart to help choose a classification cutoff by choosing a percentage that corresponds to a desirable gain, and then mapping that percentage to the appropriate cutoff value.

What constitutes a "desirable" gain depends on the cost of Type I and Type II errors. That is, what is the cost of classifying a churner as a non-churner (Type I)? What is the cost of classifying a non-churner as a churner (Type II)? If customer retention is the primary concern, then you want to lower your Type I error; on the cumulative gains chart, this might correspond to increased customer care for customers in the top 60% of predicted propensity of *1*, which captures 79.2% of the possible churners but costs time and resources that could be spent acquiring new customers. If lowering the cost of maintaining your current customer base is the priority, then you want to lower your Type II error. On the chart, this might correspond to increased customer care for the top 20%, which captures 32.5% of the churners. Usually, both are important concerns, so you have to choose a decision rule for classifying customers that gives the best mix of sensitivity and specificity.

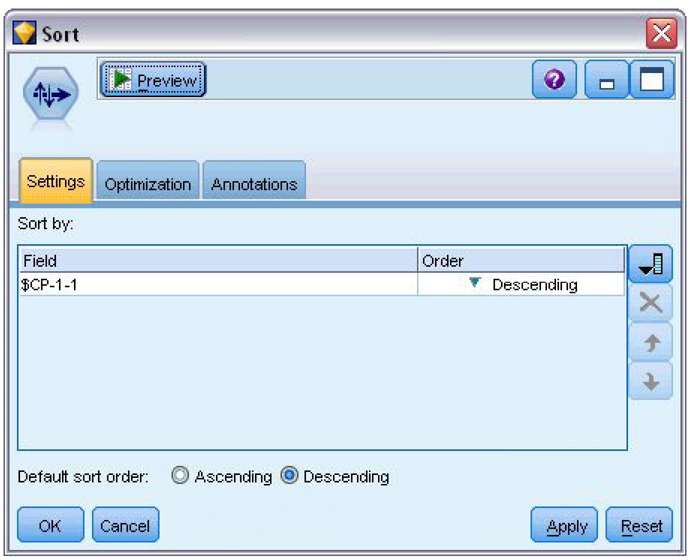


Figure 361. Sort node: Settings tab

8. Say that you have decided that 45.6% is a desirable gain, which corresponds to taking the top 30% of records. To find an appropriate classification cutoff, attach a Sort node to the model nugget.
9. On the Settings tab, choose to sort by *$CP-1-1* in descending order and click **OK**.

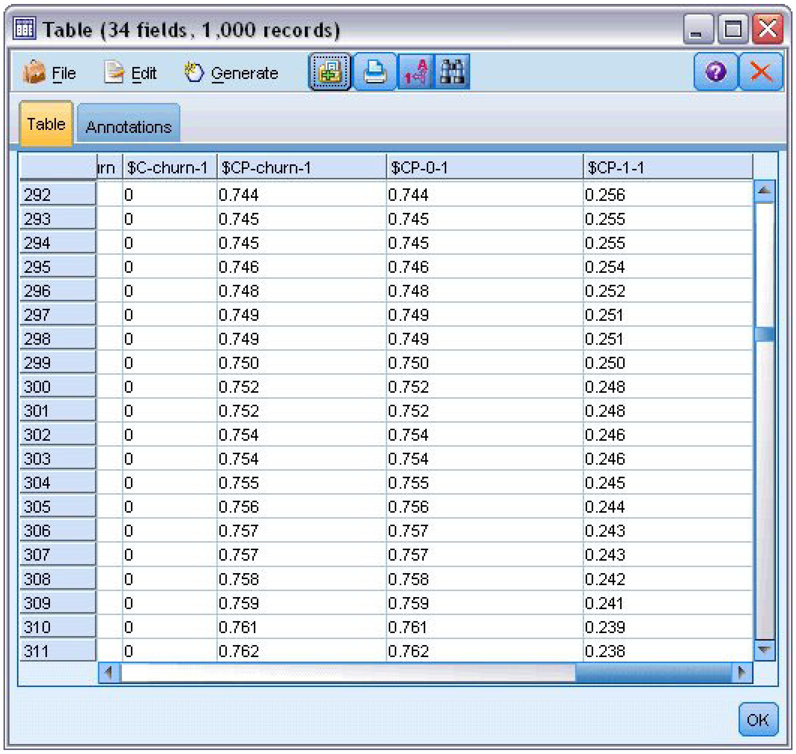


Table (34 fields, 1,000 records)

	rn	$C-churn-1	$CP-churn-1	$CP-0-1	$CP-1-1
292		0	0.744	0.744	0.256
293		0	0.745	0.745	0.255
294		0	0.745	0.745	0.255
295		0	0.746	0.746	0.254
296		0	0.748	0.748	0.252
297		0	0.749	0.749	0.251
298		0	0.749	0.749	0.251
299		0	0.750	0.750	0.250
300		0	0.752	0.752	0.248
301		0	0.752	0.752	0.248
302		0	0.754	0.754	0.246
303		0	0.754	0.754	0.246
304		0	0.755	0.755	0.245
305		0	0.756	0.756	0.244
306		0	0.757	0.757	0.243
307		0	0.757	0.757	0.243
308		0	0.758	0.758	0.242
309		0	0.759	0.759	0.241
310		0	0.761	0.761	0.239
311		0	0.762	0.762	0.238

Figure 362. Table

10. Attach a Table node to the Sort node.
11. Open the Table node and click **Run**.

Scrolling down the output, you see that the value of *$CP-1-1* is 0.248 for the 300th record. Using 0.248 as a classification cutoff should result in approximately 30% of the customers scored as churners, capturing approximately 45% of the actual total churners.

Tracking the Expected Number of Customers Retained

Once satisfied with a model, you want to track the expected number of customers in the dataset that are retained over the next two years. The null values, which are customers whose total tenure (future time + *tenure*) falls beyond the range of survival times in the data used to train the model, present an interesting challenge. One way to deal with them is to create two sets of predictions, one in which null values are assumed to have churned, and another in which they are assumed to have been retained. In this way you can establish upper and lower bounds on the expected number of customers retained.

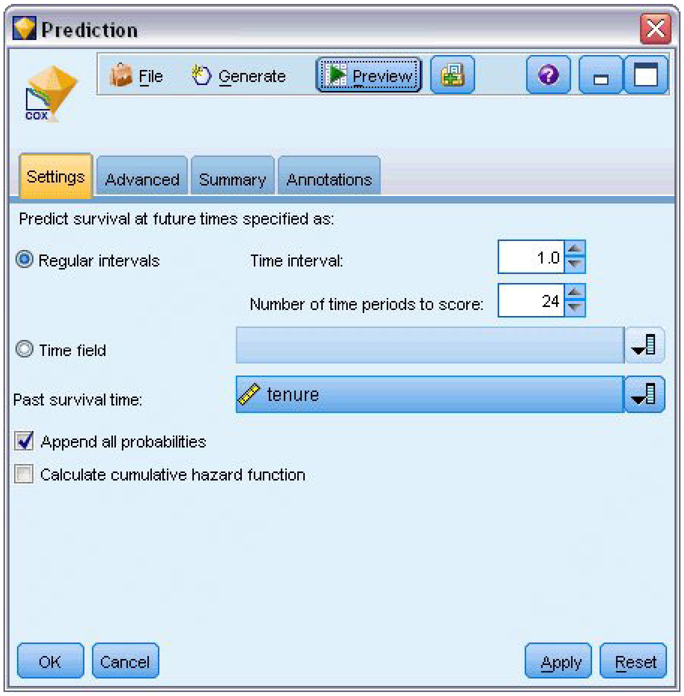


Figure 363. Cox nugget: Settings tab

1. Double-click the model nugget in the Models palette (or copy and paste the nugget on the stream canvas) and attach the new nugget to the Source node.
2. Open the nugget to the Settings tab.
3. Make sure **Regular Intervals** is selected, and specify `1.0` as the time interval and `24` as the number of periods to score. This specifies that each record will be scored for each of the following 24 months.
4. Select *tenure* as the field to specify the past survival time. The scoring algorithm will take into account the length of each customer's time as a customer of the company.
5. Select **Append all probabilities**.

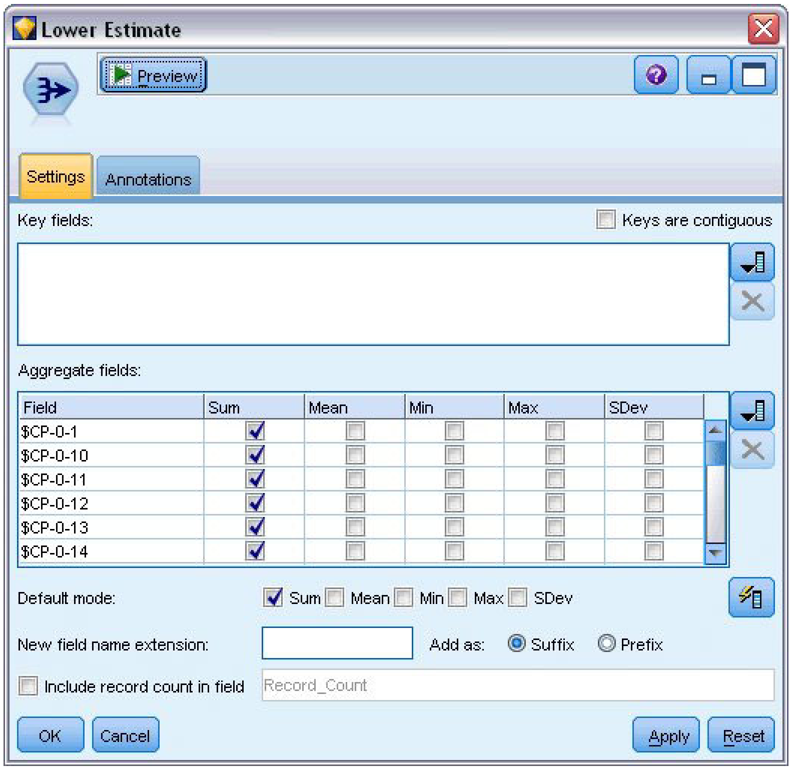


Figure 364. Aggregate node: Settings tab

6. Attach an Aggregate node to the model nugget; on the Settings tab, deselect **Mean** as a default mode.
7. Select *$CP-0-1* through *$CP-0-24*, the fields of form *$CP-0-n*, as the fields to aggregate. This is easiest if, on the Select Fields dialog, you sort the fields by Name (that is, alphabetical order).
8. Deselect **Include record count in field**.
9. Click **OK**. This node creates the "lower bound" predictions.

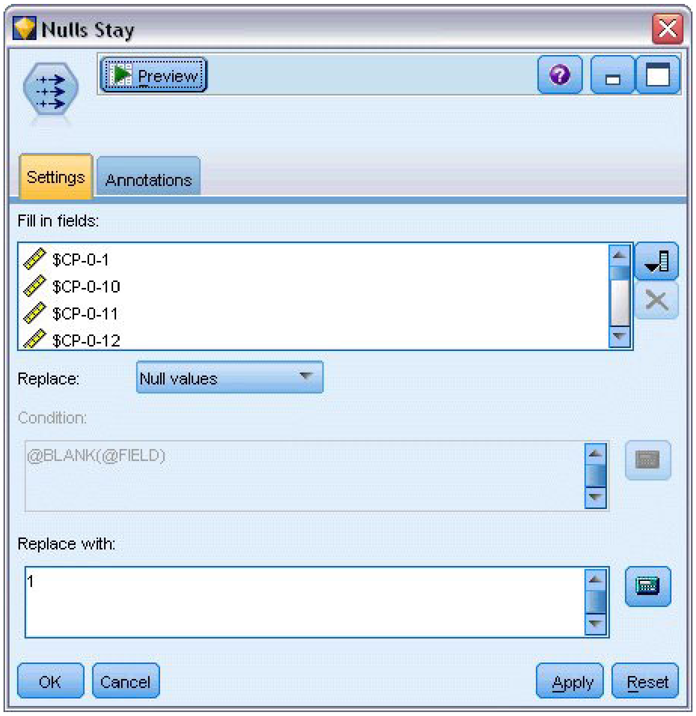


Figure 365. Filler node: Settings tab

10. Attach a Filler node to the Coxreg nugget to which we just attached the Aggregate node; on the Settings tab, select *$CP-0-1* through *$CP-0-24*, the fields of form *$CP-0-n*, as the fields to fill in. This is easiest if, on the Select Fields dialog, you sort the fields by Name (that is, alphabetical order).
11. Choose to replace **Null values** with the value 1.
12. Click **OK**.

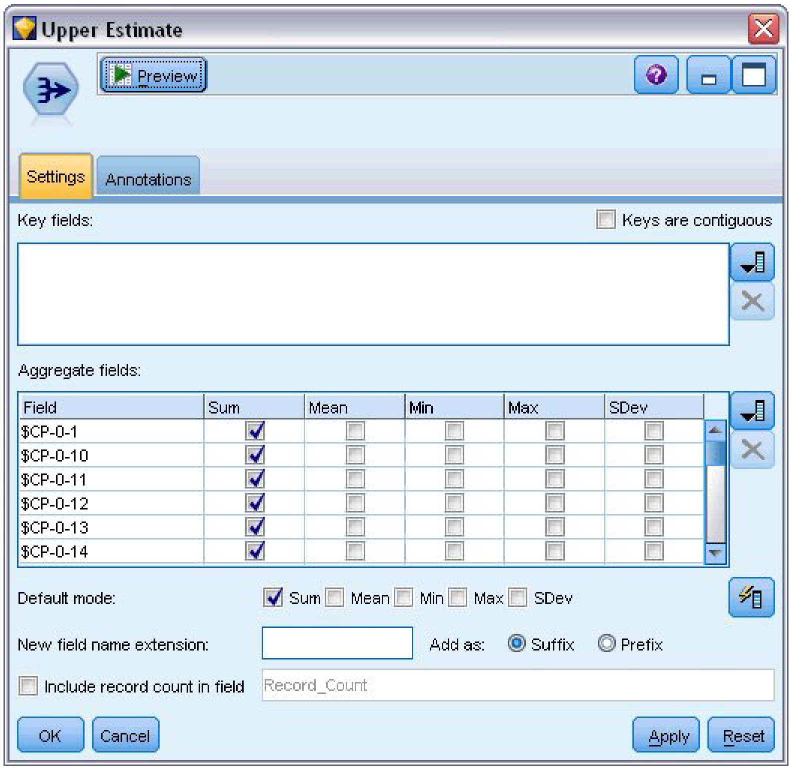


Figure 366. Aggregate node: Settings tab

13. Attach an Aggregate node to the Filler node; on the Settings tab, deselect **Mean** as a default mode.
14. Select *$CP-0-1* through *$CP-0-24*, the fields of form *$CP-0-n*, as the fields to aggregate. This is easiest if, on the Select Fields dialog, you sort the fields by Name (that is, alphabetical order).
15. Deselect **Include record count in field**.
16. Click **OK**. This node creates the "upper bound" predictions.

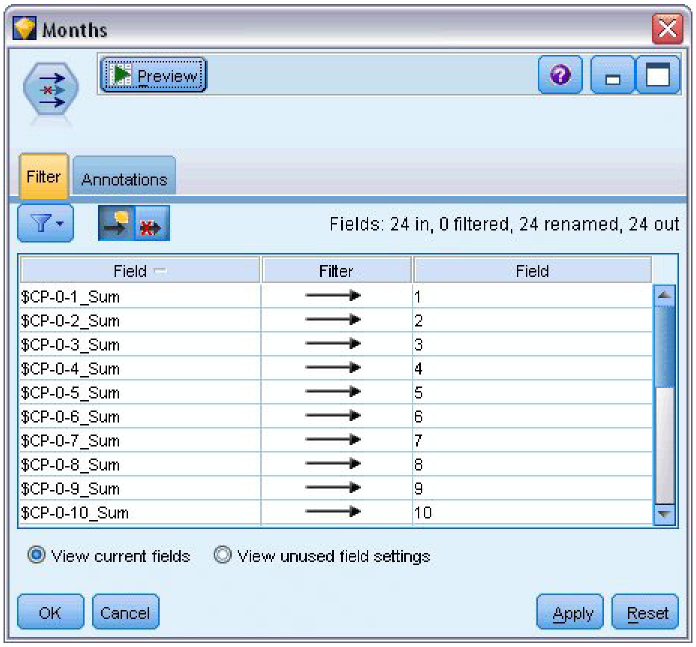


Figure 367. Filter node: Settings tab

17. Attach an Append node to the two Aggregate nodes, then attach a Filter node to the Append node.
18. On the Settings tab of the Filter node, rename the fields to *1* through *24*. Through the use of a Transpose node, these field names will become values for the *x*-axis in charts downstream.

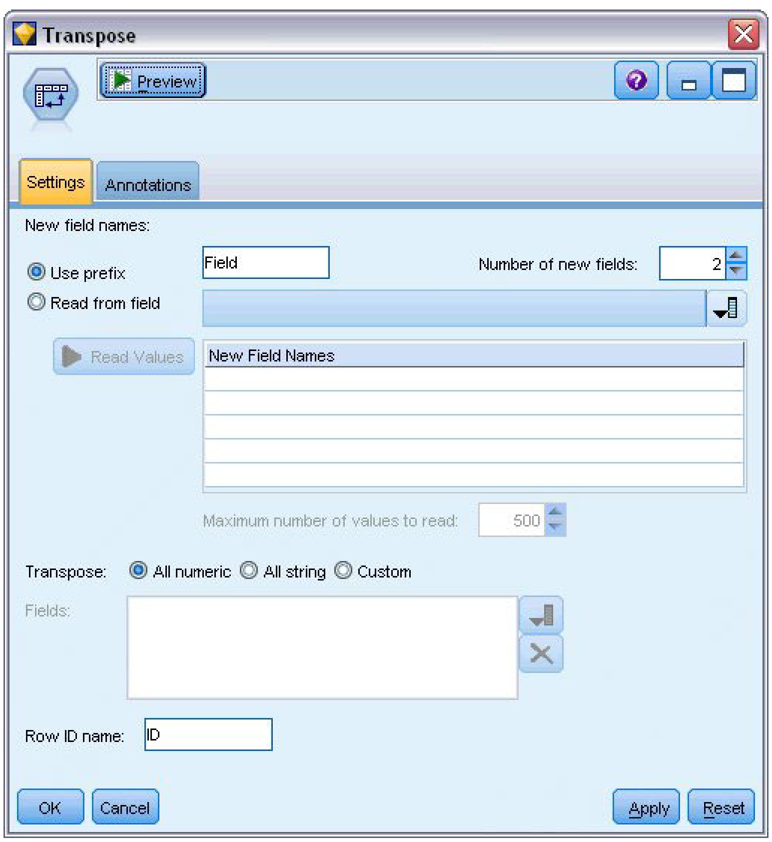


Figure 368. Transpose node: Settings tab

19. Attach a Transpose node to the Filter node.
20. Type 2 as the number of new fields.

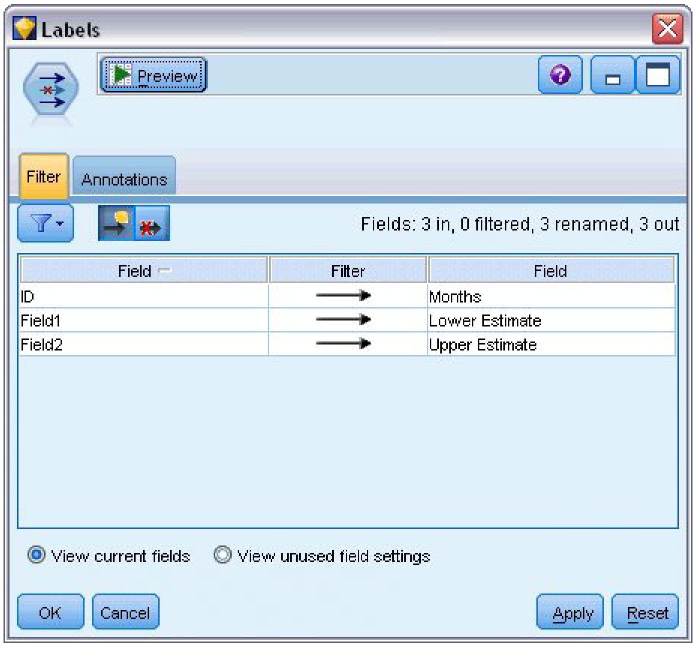


Figure 369. Filter node: Filter tab

21. Attach a Filter node to the Transpose node.
22. On the Settings tab of the Filter node, rename *ID* to *Months*, *Field1* to *Lower Estimate*, and *Field2* to *Upper Estimate*.

Figure 370. Multiplot node: Plot tab

23. Attach a Multiplot node to the Filter node.
24. On the Plot tab, *Months* as the X field, *Lower Estimate* and *Upper Estimate* as the Y fields.

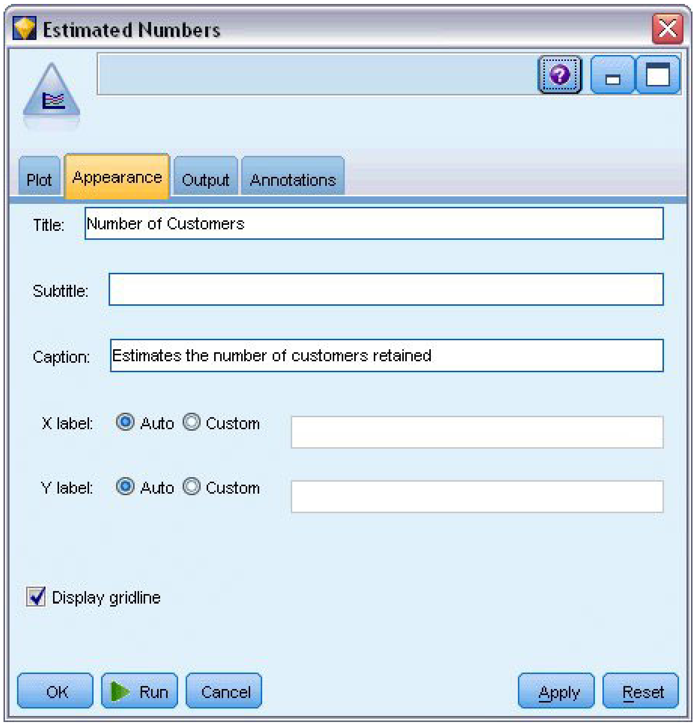


Figure 371. Multiplot node: Appearance tab

25. Click the Appearance tab.
26. Type `Number of Customers` as the title.
27. Type `Estimates the number of customers retained` as the caption.
28. Click **Run**.

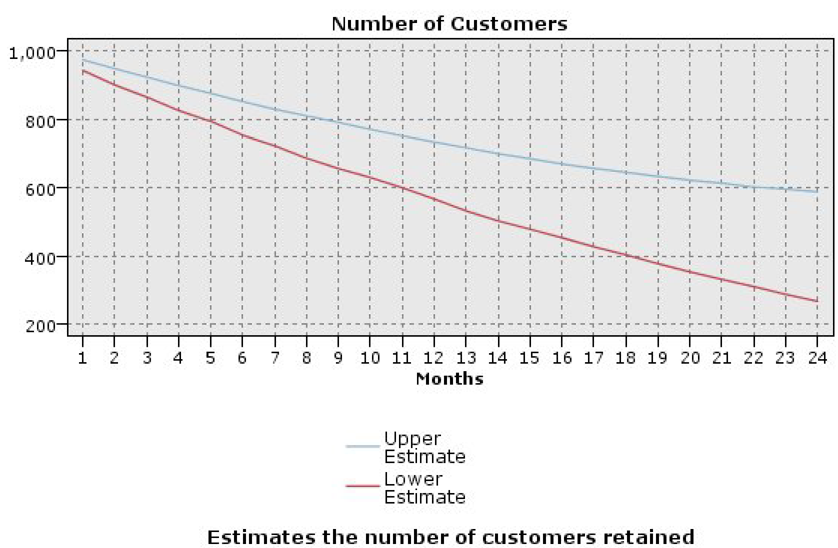


Figure 372. Multiplot estimating the number of customers retained

The upper and lower bounds on the estimated number of customers retained are plotted. The difference between the two lines is the number of customers scored as null, and therefore whose status is highly uncertain. Over time, the number of these customers increases. After 12 months, you can expect to retain between 601 and 735 of the original customers in the dataset; after 24 months, between 288 and 597.

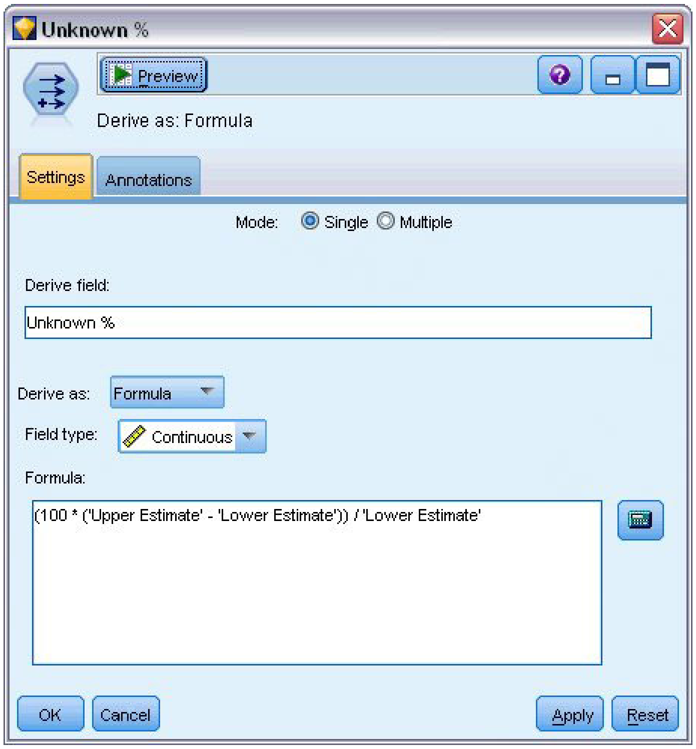


Figure 373. Derive node: Settings tab

29. To get another look at how uncertain the estimates of the number of customers retained are, attach a Derive node to the Filter node.
30. On the Settings tab of the Derive node, type *Unknown %* as the derive field.
31. Select **Continuous** as the field type.
32. Type `(100 * ('Upper Estimate' - 'Lower Estimate')) / 'Lower Estimate'` as the formula. *Unknown %* is the number of customers "in doubt" as a percentage of the lower estimate.
33. Click **OK**.

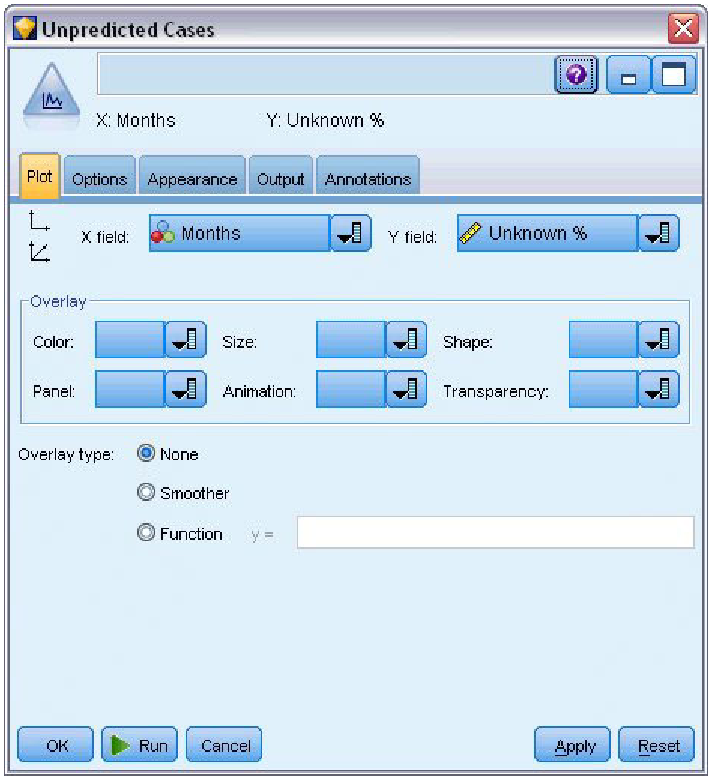


Figure 374. Plot node: Plot tab

34. Attach a Plot node to the Derive node.
35. On the Plot tab of the Plot node, select *Months* as the X field and *Unknown %* as the Y field.
36. Click the **Appearance** tab.

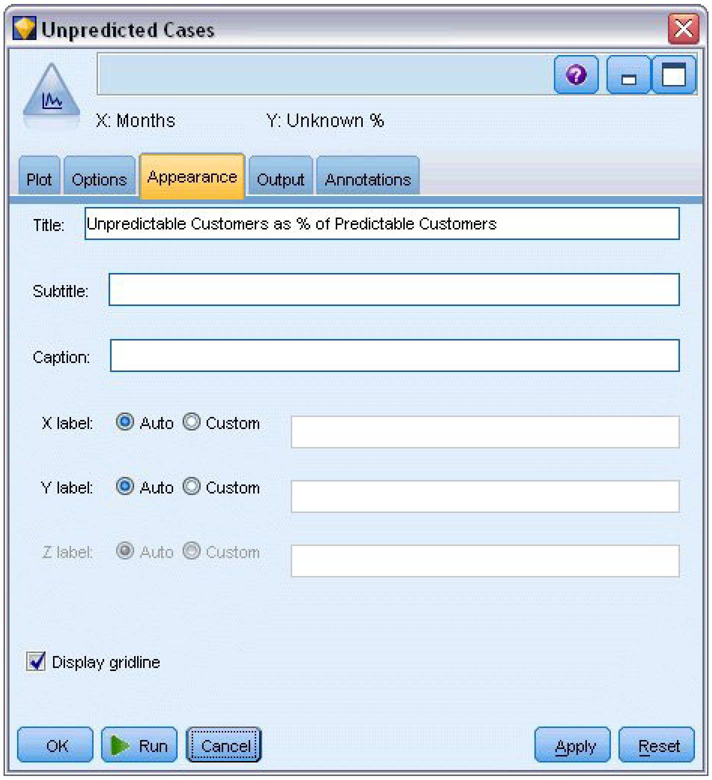


Figure 375. Plot node: Appearance tab

37. Type `Unpredictable Customers as % of Predictable Customers` as the title.
38. Execute the node.

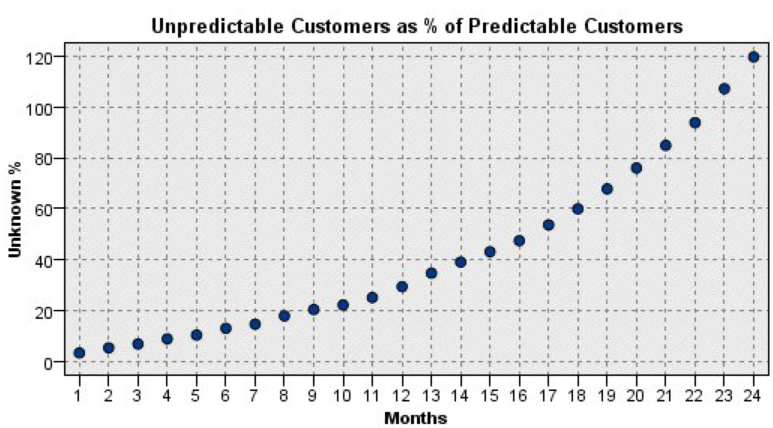


Figure 376. Plot of unpredictable customers

Through the first year, the percentage of unpredictable customers increases at a fairly linear rate, but the rate of increase explodes during the second year until, by month 23, the number of customers with null values outnumber the expected number of customers retained.

Scoring

Once satisfied with a model, you want to score customers to identify the individuals most likely to churn within the next year, by quarter.

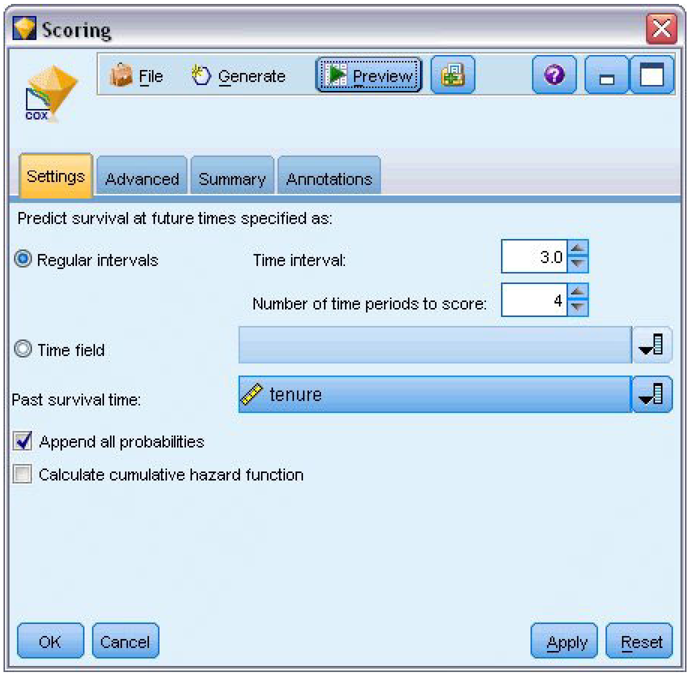


Figure 377. Coxreg nugget: Settings tab

1. Attach a third model nugget to the Source node and open the model nugget.
2. Make sure **Regular Intervals** is selected, and specify `3.0` as the time interval and `4` as the number of periods to score. This specifies that each record will be scored for the following four quarters.
3. Select *tenure* as the field to specify the past survival time. The scoring algorithm will take into account the length of each customer's time as a customer of the company.
4. Select **Append all probabilities**. These extra fields will make it easier to sort the records for viewing in a table.

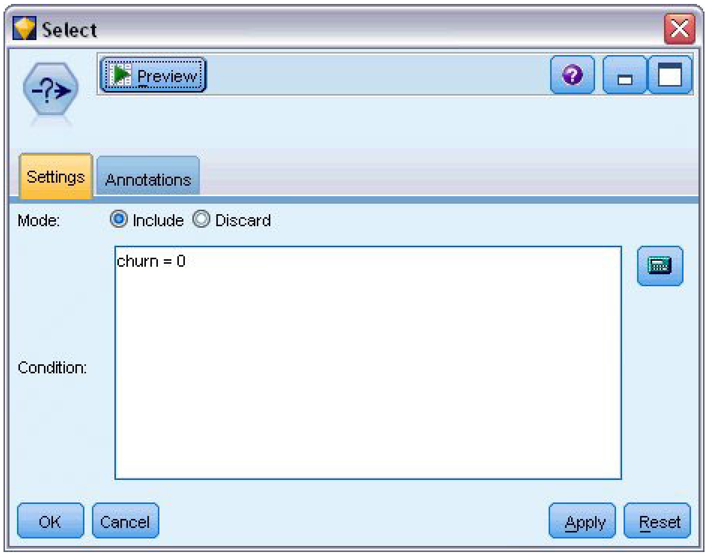


Figure 378. Select node: Settings tab

5. Attach a Select node to the model nugget; on the Settings tab, type `churn=0` as the condition. This removes customers who have already churned from the results table.

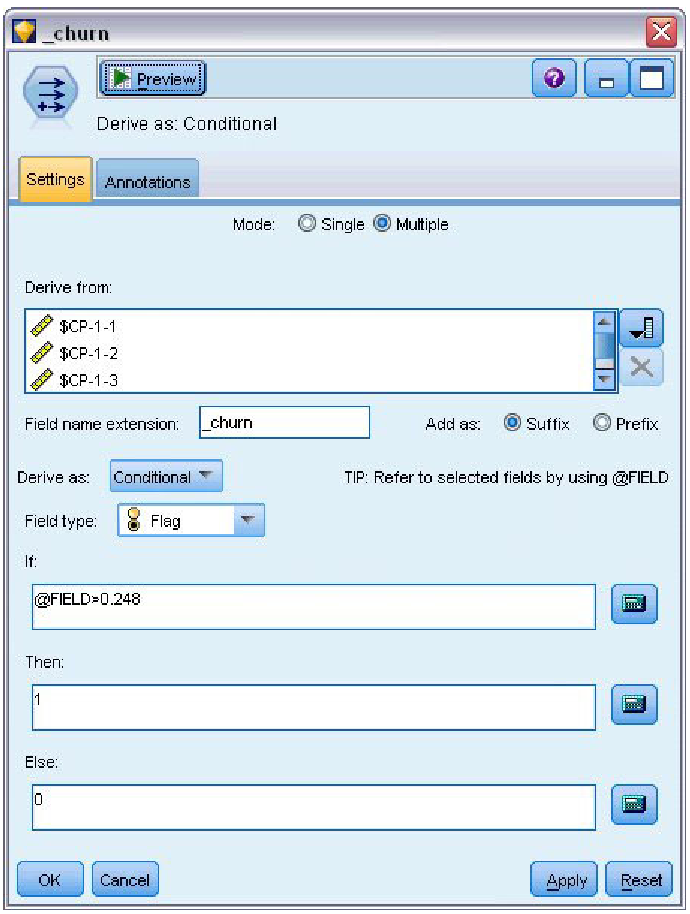


Figure 379. Derive node: Settings tab

6. Attach a Derive node to the Select node; on the Settings tab, select **Multiple** as the mode.

7. Choose to derive from *$CP-1-1* through *$CP-1-4*, the fields of form *$CP-1-n*, and type `_churn` as the suffix to add. This is easiest if, on the Select Fields dialog, you sort the fields by Name (that is, alphabetical order).
8. Choose to derive the field as a **Conditional**.
9. Select **Flag** as the measurement level.
10. Type `@FIELD>0.248` as the **If** condition. Recall that this was the classification cutoff identified during Evaluation.
11. Type 1 as the **Then** expression.
12. Type `0` as the **Else** expression.
13. Click **OK**.

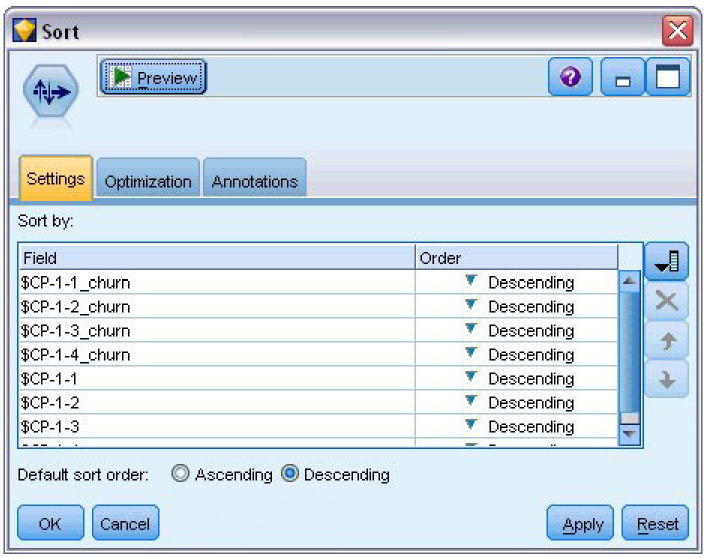
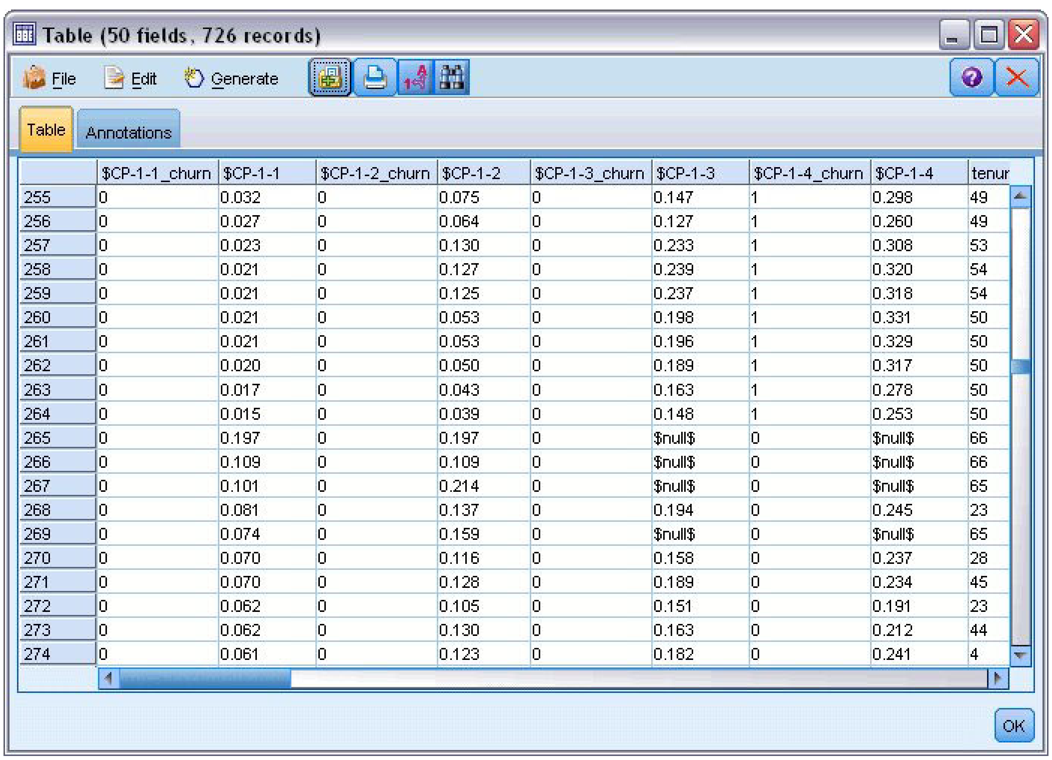


Figure 380. Sort node: Settings tab

14. Attach a Sort node to the Derive node; on the Settings tab, choose to sort by *$CP-1-1_churn* through *$CP-1-4-churn* and then *$CP-1-1* through *$CP-1-4*, all in descending order. Customers who are predicted to churn will appear at the top.

Figure 381. Field Reorder node: Reorder tab

15. Attach a Field Reorder node to the Sort node; on the Reorder tab, choose to place *$CP-1-1_churn* through *$CP-1-4* in front of the other fields. This simply makes the results table easier to read, and so is optional. You will need to use the buttons to move the fields into the position shown in the figure.

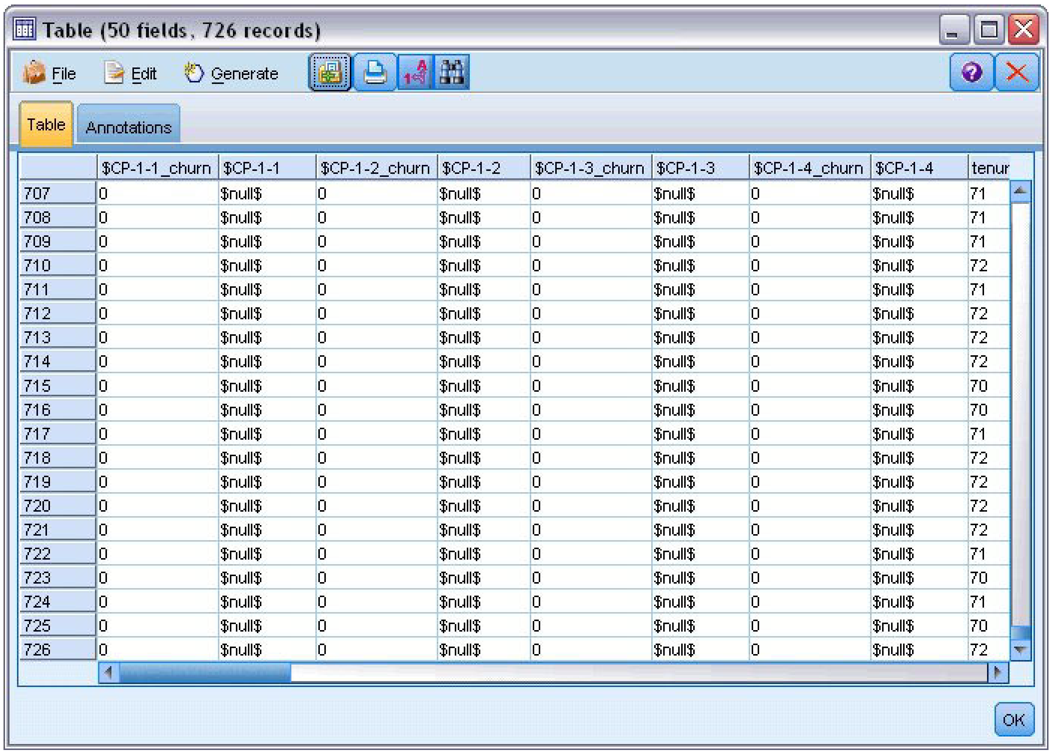


	$CP-1-1_churn	$CP-1-1	$CP-1-2_churn	$CP-1-2	$CP-1-3_churn	$CP-1-3	$CP-1-4_churn	$CP-1-4	tenur
255	0	0.032	0	0.075	0	0.147	1	0.298	49
256	0	0.027	0	0.064	0	0.127	1	0.260	49
257	0	0.023	0	0.130	0	0.233	1	0.308	53
258	0	0.021	0	0.127	0	0.239	1	0.320	54
259	0	0.021	0	0.125	0	0.237	1	0.318	54
260	0	0.021	0	0.053	0	0.198	1	0.331	50
261	0	0.021	0	0.053	0	0.196	1	0.329	50
262	0	0.020	0	0.050	0	0.189	1	0.317	50
263	0	0.017	0	0.043	0	0.163	1	0.278	50
264	0	0.015	0	0.039	0	0.148	1	0.253	50
265	0	0.197	0	0.197	0	$null$	0	$null$	66
266	0	0.109	0	0.109	0	$null$	0	$null$	66
267	0	0.101	0	0.214	0	$null$	0	$null$	65
268	0	0.081	0	0.137	0	0.194	0	0.245	23
269	0	0.074	0	0.159	0	$null$	0	$null$	65
270	0	0.070	0	0.116	0	0.158	0	0.237	28
271	0	0.070	0	0.128	0	0.189	0	0.234	45
272	0	0.062	0	0.105	0	0.151	0	0.191	23
273	0	0.062	0	0.130	0	0.163	0	0.212	44
274	0	0.061	0	0.123	0	0.182	0	0.241	4

Figure 382. Table showing customer scores

16. Attach a Table node to the Field Reorder node and execute it.

264 customers are expected to churn by the end of the year, 184 by the end of the third quarter, 103 by the second, and 31 in the first. Note that given two customers, the one with a higher propensity to churn in the first quarter does not necessarily have a higher propensity to churn in later quarters; for example, see records 256 and 260. This is likely due to the shape of the hazard function for the months following the customer's current tenure; for example, customers who joined because of a promotion might be more likely to switch early on than customers who joined because of a personal recommendation, but if they do not then they may actually be more loyal for their remaining tenure. You may want to re-sort the customers to obtain different views of the customers most likely to churn.



	$CP-1-1_churn	$CP-1-1	$CP-1-2_churn	$CP-1-2	$CP-1-3_churn	$CP-1-3	$CP-1-4_churn	$CP-1-4	tenur
707	0	$null$	0	$null$	0	$null$	0	$null$	71
708	0	$null$	0	$null$	0	$null$	0	$null$	71
709	0	$null$	0	$null$	0	$null$	0	$null$	71
710	0	$null$	0	$null$	0	$null$	0	$null$	72
711	0	$null$	0	$null$	0	$null$	0	$null$	71
712	0	$null$	0	$null$	0	$null$	0	$null$	72
713	0	$null$	0	$null$	0	$null$	0	$null$	72
714	0	$null$	0	$null$	0	$null$	0	$null$	72
715	0	$null$	0	$null$	0	$null$	0	$null$	70
716	0	$null$	0	$null$	0	$null$	0	$null$	70
717	0	$null$	0	$null$	0	$null$	0	$null$	71
718	0	$null$	0	$null$	0	$null$	0	$null$	72
719	0	$null$	0	$null$	0	$null$	0	$null$	72
720	0	$null$	0	$null$	0	$null$	0	$null$	72
721	0	$null$	0	$null$	0	$null$	0	$null$	72
722	0	$null$	0	$null$	0	$null$	0	$null$	71
723	0	$null$	0	$null$	0	$null$	0	$null$	70
724	0	$null$	0	$null$	0	$null$	0	$null$	71
725	0	$null$	0	$null$	0	$null$	0	$null$	70
726	0	$null$	0	$null$	0	$null$	0	$null$	72

Figure 383. Table showing customers with null values

At the bottom of the table are customers with predicted null values. These are customers whose total tenure (future time + *tenure*) falls beyond the range of survival times in the data used to train the model.

Summary

Using Cox regression, you have found an acceptable model for the time to churn, plotted the expected number of customers retained over the next two years, and identified the individual customers most likely to churn in the next year. Note that while this is an acceptable model, it may not be the best model. Ideally you should at least compare this model, obtained using the Forward stepwise method, with one created using the Backward stepwise method.

Explanations of the mathematical foundations of the modeling methods used in IBM SPSS Modeler are listed in the *IBM SPSS Modeler Algorithms Guide*.

Chapter 27. Market Basket Analysis (Rule Induction/C5.0)

This example deals with fictitious data describing the contents of supermarket baskets (that is, collections of items bought together) plus the associated personal data of the purchaser, which might be acquired through a loyalty card scheme. The goal is to discover groups of customers who buy similar products and can be characterized demographically, such as by age, income, and so on.

This example illustrates two phases of data mining:

- Association rule modeling and a web display revealing links between items purchased
- C5.0 rule induction profiling the purchasers of identified product groups

Note: This application does not make direct use of predictive modeling, so there is no accuracy measurement for the resulting models and no associated training/test distinction in the data mining process.

This example uses the stream named *baskrule*, which references the data file named *BASKETS1n*. These files are available from the *Demos* directory of any IBM SPSS Modeler installation. This can be accessed from the IBM SPSS Modeler program group on the Windows Start menu. The *baskrule* file is in the *streams* directory.

Accessing the Data

Using a Variable File node, connect to the dataset *BASKETS1n*, selecting to read field names from the file. Connect a Type node to the data source, and then connect the node to a Table node. Set the measurement level of the field *cardid* to *Typeless* (because each loyalty card ID occurs only once in the dataset and can therefore be of no use in modeling). Select *Nominal* as the measurement level for the field *sex* (this is to ensure that the Apriori modeling algorithm will not treat *sex* as a flag).

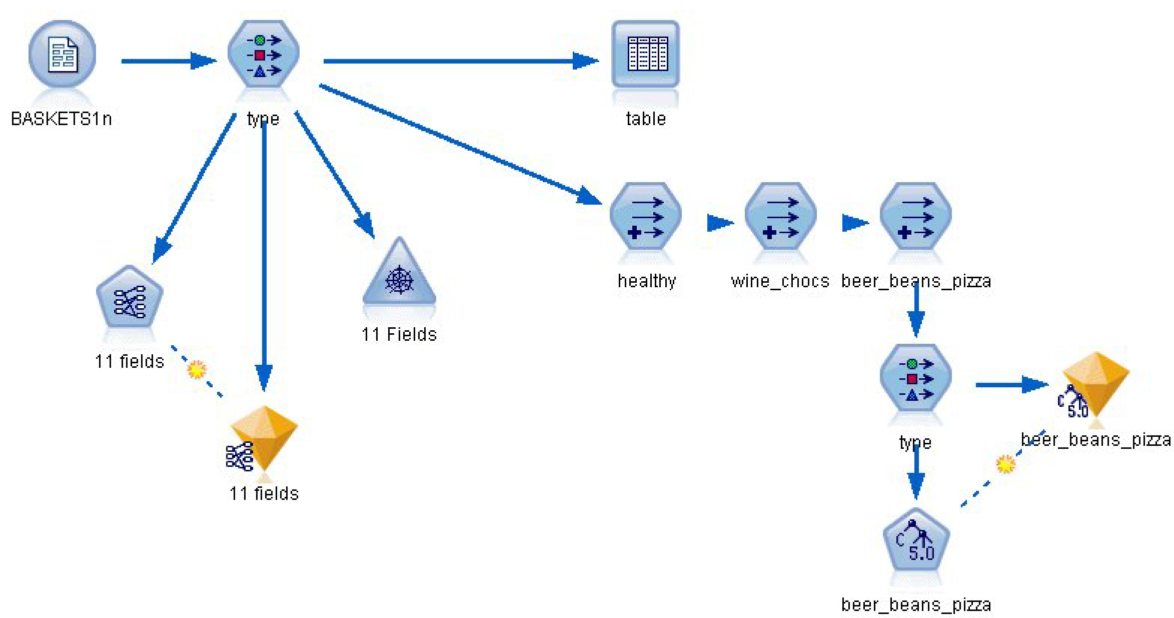


Figure 384. baskrule stream

Now run the stream to instantiate the Type node and display the table. The dataset contains 18 fields, with each record representing a basket.

The 18 fields are presented in the following headings.

Basket summary:

- *cardid*. Loyalty card identifier for customer purchasing this basket.
- *value*. Total purchase price of basket.
- *pmethod*. Method of payment for basket.

Personal details of cardholder:

- *sex*
- *homeown*. Whether or not cardholder is a homeowner.
- *income*
- *age*

Basket contents—flags for presence of product categories:

- *fruitveg*
- *freshmeat*
- *dairy*
- *cannedveg*
- *cannedmeat*
- *frozenmeal*
- *beer*
- *wine*
- *softdrink*
- *fish*
- *confectionery*

Discovering Affinities in Basket Contents

First, you need to acquire an overall picture of affinities (associations) in the basket contents using Apriori to produce association rules. Select the fields to be used in this modeling process by editing the Type node and setting the role of all of the product categories to *Both* and all other roles to *None*. (*Both* means that the field can be either an input or an output of the resultant model.)

Note: You can set options for multiple fields using Shift-click to select the fields before specifying an option from the columns.

Figure 385. Selecting fields for modeling

Once you have specified fields for modeling, attach an Apriori node to the Type node, edit it, select the option **Only true values for flags**, and click run on the Apriori node. The result, a model on the Models tab at the upper right of the managers window, contains association rules that you can view by using the context menu and selecting **Browse**.

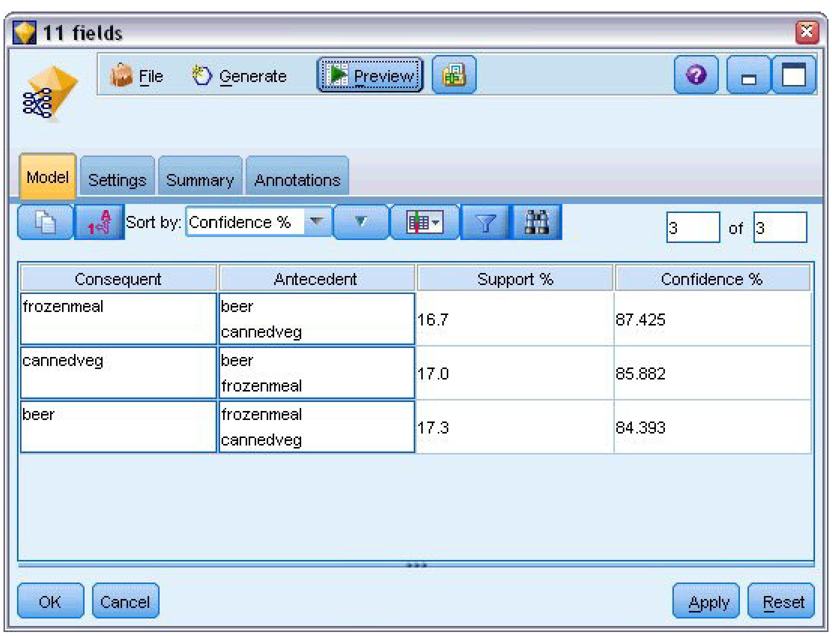


Figure 386. Association rules

These rules show a variety of associations between frozen meals, canned vegetables, and beer. The presence of two-way association rules, such as:

```
frozenmeal -> beer
beer -> frozenmeal
```

suggests that a web display (which shows only two-way associations) might highlight some of the patterns in this data.

Attach a Web node to the Type node, edit the Web node, select all of the basket contents fields, select **Show true flags only**, and click run on the Web node.

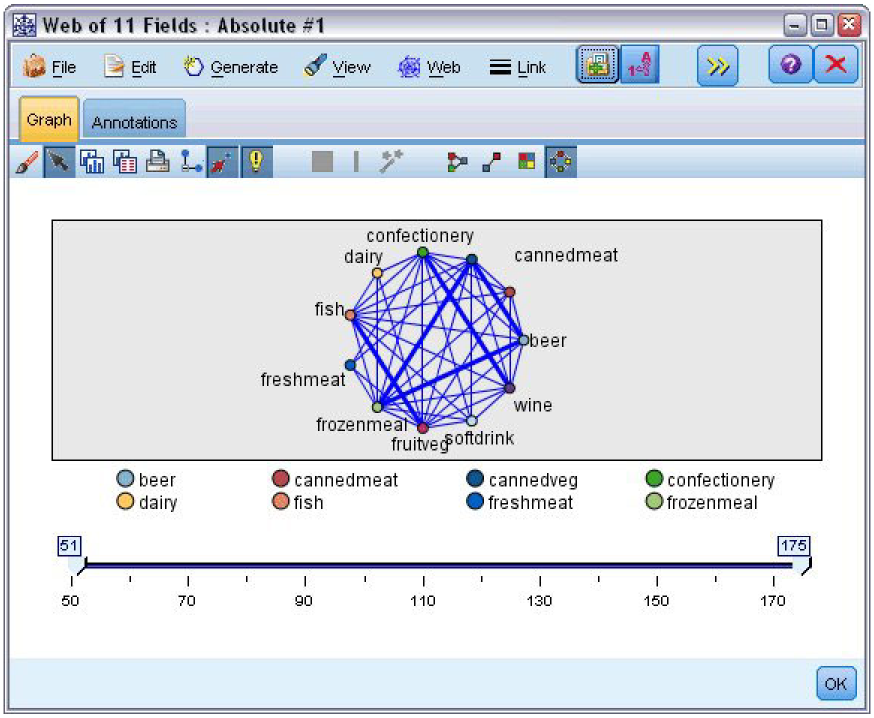


Figure 387. Web display of product associations

Because most combinations of product categories occur in several baskets, the strong links on this web are too numerous to show the groups of customers suggested by the model.

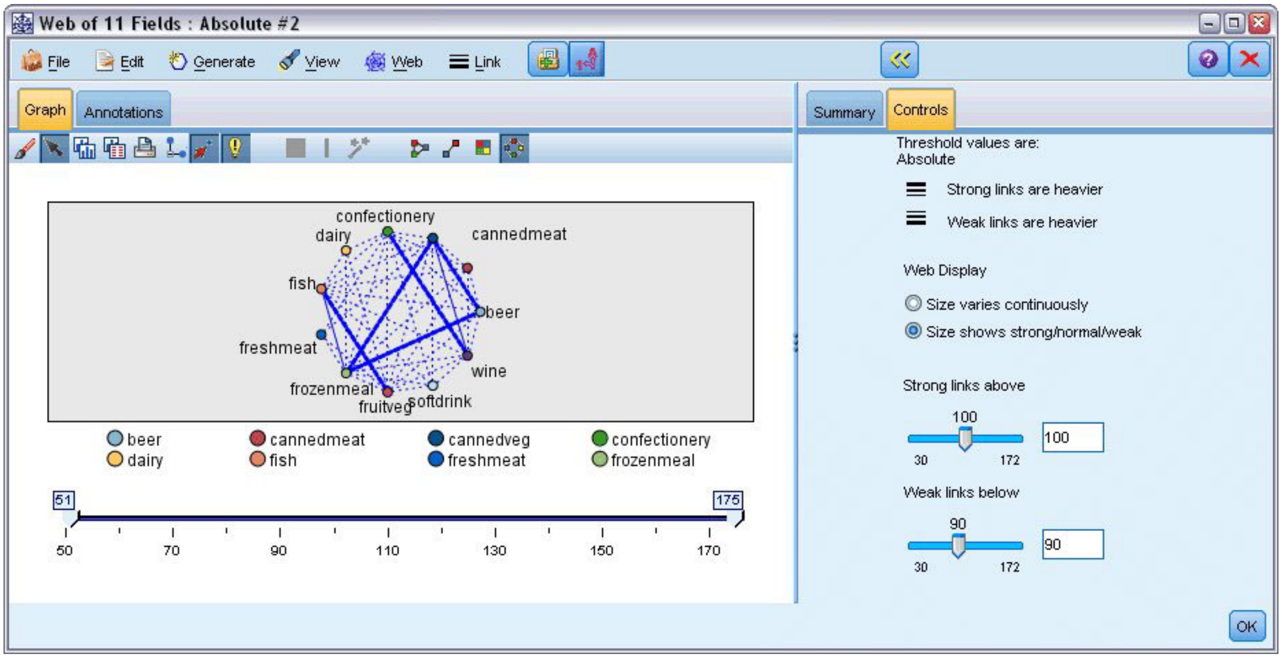


Figure 388. Restricted web display

1. To specify weak and strong connections, click the yellow double arrow button on the toolbar. This expands the dialog box showing the web output summary and controls.
2. Select **Size shows strong/normal/weak**.
3. Set weak links below 90.
4. Set strong links above 100.

In the resulting display, three groups of customers stand out:
- Those who buy fish and fruits and vegetables, who might be called "healthy eaters"
- Those who buy wine and confectionery
- Those who buy beer, frozen meals, and canned vegetables ("beer, beans, and pizza")

Profiling the Customer Groups

You have now identified three groups of customers based on the types of products they buy, but you would also like to know who these customers are--that is, their demographic profile. This can be achieved by tagging each customer with a flag for each of these groups and using rule induction (C5.0) to build rule-based profiles of these flags.

First, you must derive a flag for each group. This can be automatically generated using the web display that you just created. Using the right mouse button, click the link between *fruitveg* and *fish* to highlight it, then right-click and select **Generate Derive Node For Link.**

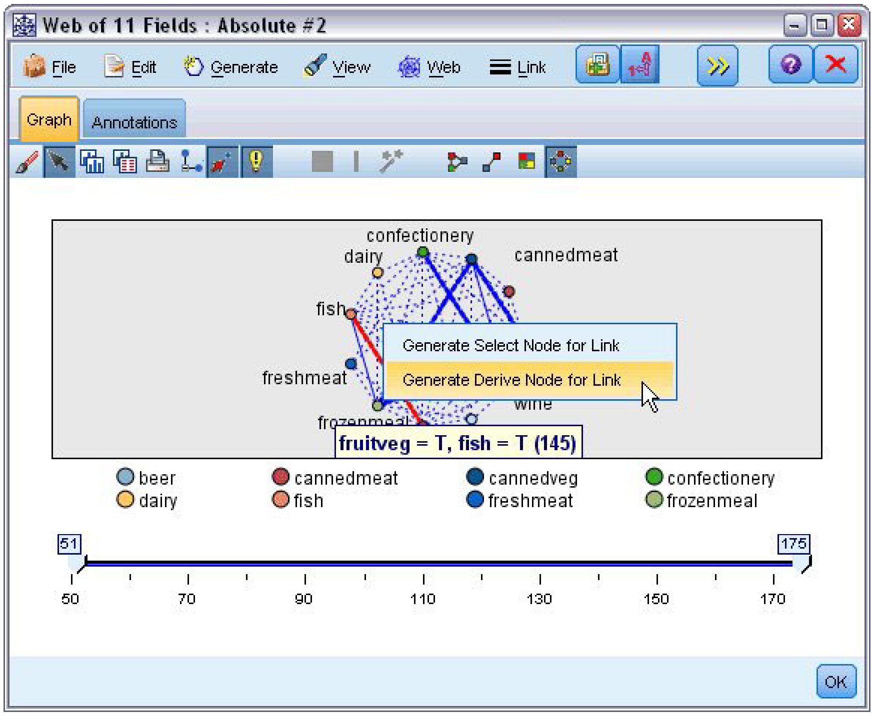


Figure 389. Deriving a flag for each customer group

Edit the resulting Derive node to change the Derive field name to *healthy*. Repeat the exercise with the link from *wine* to *confectionery*, naming the resultant Derive field *wine_chocs*.

For the third group (involving three links), first make sure that no links are selected. Then select all three links in the *cannedveg*, *beer*, and *frozenmeal* triangle by holding down the shift key while you click the left mouse button. (Be sure you are in Interactive mode rather than Edit mode.) Then from the web display menus choose:

Generate > **Derive Node ("And")**

Change the name of the resultant Derive field to *beer_beans_pizza*.

To profile these customer groups, connect the existing Type node to these three Derive nodes in series, and then attach another Type node. In the new Type node, set the role of all fields to *None*, except for *value*, *pmethod*, *sex*, *homeown*, *income*, and *age*, which should be set to *Input*, and the relevant customer group (for example, *beer_beans_pizza*), which should be set to *Target*. Attach a C5.0 node, set the Output type to **Rule set**, and click run on the node. The resultant model (for *beer_beans_pizza*) contains a clear demographic profile for this customer group:

```
Rule 1 for T:
if sex = M
and income <= 16,900
then T
```

The same method can be applied to the other customer group flags by selecting them as the output in the second Type node. A wider range of alternative profiles can be generated by using Apriori instead of C5.0 in this context; Apriori can also be used to profile all of the customer group flags simultaneously because it is not restricted to a single output field.

Summary

This example reveals how IBM SPSS Modeler can be used to discover affinities, or links, in a database, both by modeling (using Apriori) and by visualization (using a web display). These links correspond to groupings of cases in the data, and these groups can be investigated in detail and profiled by modeling (using C5.0 rule sets).

In the retail domain, such customer groupings might, for example, be used to target special offers to improve the response rates to direct mailings or to customize the range of products stocked by a branch to match the demands of its demographic base.

Chapter 28. Assessing New Vehicle Offerings (KNN)

Nearest Neighbor Analysis is a method for classifying cases based on their similarity to other cases. In machine learning, it was developed as a way to recognize patterns of data without requiring an exact match to any stored patterns, or cases. Similar cases are near each other and dissimilar cases are distant from each other. Thus, the distance between two cases is a measure of their dissimilarity.

Cases that are near each other are said to be “neighbors.” When a new case (holdout) is presented, its distance from each of the cases in the model is computed. The classifications of the most similar cases – the nearest neighbors – are tallied and the new case is placed into the category that contains the greatest number of nearest neighbors.

You can specify the number of nearest neighbors to examine; this value is called *k*. The pictures show how a new case would be classified using two different values of *k*. When *k* = 5, the new case is placed in category *1* because a majority of the nearest neighbors belong to category *1*. However, when *k* = 9, the new case is placed in category *0* because a majority of the nearest neighbors belong to category *0*.

Nearest neighbor analysis can also be used to compute values for a continuous target. In this situation, the average or median target value of the nearest neighbors is used to obtain the predicted value for the new case.

An automobile manufacturer has developed prototypes for two new vehicles, a car and a truck. Before introducing the new models into its range, the manufacturer wants to determine which existing vehicles on the market are most like the prototypes--that is, which vehicles are their "nearest neighbors", and therefore which models they will be competing against.

The manufacturer has collected data about the existing models under a number of categories, and has added the details of its prototypes. The categories under which the models are to be compared include price in thousands (*price*), engine size (*engine_s*), horsepower (*horsepow*), wheelbase (*wheelbas*), width (*width*), length (*length*), curb weight (*curb_wgt*), fuel capacity (*fuel_cap*) and fuel efficiency (*mpg*).

This example uses the stream named *car_sales_knn.str*, available in the *Demos* folder under the *streams* subfolder. The data file is *car_sales_knn_mod.sav*. See the topic “Demos Folder” on page 4 for more information.

Creating the Stream

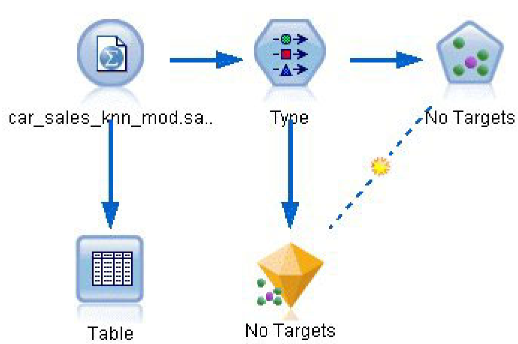
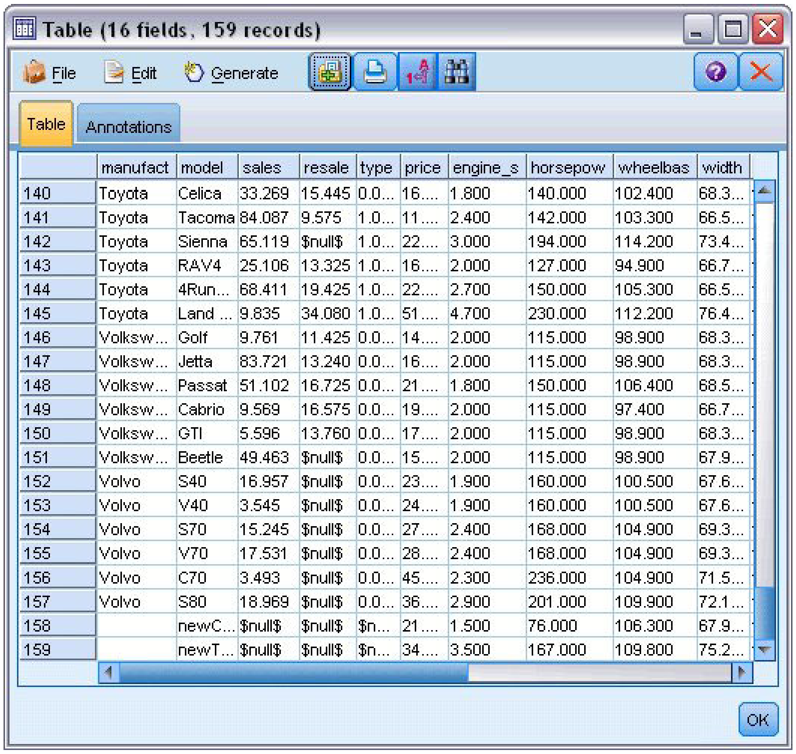


Figure 390. Sample stream for KNN modeling

Create a new stream and add a Statistics File source node pointing to *car_sales_knn_mod.sav* in the *Demos* folder of your IBM SPSS Modeler installation.

First, let's see what data the manufacturer has collected.

1. Attach a Table node to the Statistics File source node.
2. Open the Table node and click **Run**.



	manufact	model	sales	resale	type	price	engine_s	horsepow	wheelbas	width
140	Toyota	Celica	33.269	15.445	0.0...	16....	1.800	140.000	102.400	68.3...
141	Toyota	Tacoma	84.087	9.575	1.0...	11....	2.400	142.000	103.300	66.5...
142	Toyota	Sienna	65.119	$null$	1.0...	22....	3.000	194.000	114.200	73.4...
143	Toyota	RAV4	25.106	13.325	1.0...	16....	2.000	127.000	94.900	66.7...
144	Toyota	4Run...	68.411	19.425	1.0...	22....	2.700	150.000	105.300	66.5...
145	Toyota	Land ...	9.835	34.080	1.0...	51....	4.700	230.000	112.200	76.4...
146	Volksw...	Golf	9.761	11.425	0.0...	14....	2.000	115.000	98.900	68.3...
147	Volksw...	Jetta	83.721	13.240	0.0...	16....	2.000	115.000	98.900	68.3...
148	Volksw...	Passat	51.102	16.725	0.0...	21....	1.800	150.000	106.400	68.5...
149	Volksw...	Cabrio	9.569	16.575	0.0...	19....	2.000	115.000	97.400	66.7...
150	Volksw...	GTI	5.596	13.760	0.0...	17....	2.000	115.000	98.900	68.3...
151	Volksw...	Beetle	49.463	$null$	0.0...	15....	2.000	115.000	98.900	67.9...
152	Volvo	S40	16.957	$null$	0.0...	23....	1.900	160.000	100.500	67.6...
153	Volvo	V40	3.545	$null$	0.0...	24....	1.900	160.000	100.500	67.6...
154	Volvo	S70	15.245	$null$	0.0...	27....	2.400	168.000	104.900	69.3...
155	Volvo	V70	17.531	$null$	0.0...	28....	2.400	168.000	104.900	69.3...
156	Volvo	C70	3.493	$null$	0.0...	45....	2.300	236.000	104.900	71.5...
157	Volvo	S80	18.969	$null$	0.0...	36....	2.900	201.000	109.900	72.1...
158		newC...	$null$	$null$	$n...	21....	1.500	76.000	106.300	67.9...
159		newT...	$null$	$null$	$n...	34....	3.500	167.000	109.800	75.2...

Figure 391. Source data for cars and trucks

The details for the two prototypes, named *newCar* and *newTruck*, have been added at the end of the file.

We can see from the source data that the manufacturer is using the classification of "truck" (value of 1 in the *type* column) rather loosely to mean any non-automobile type of vehicle.

The last column, *partition*, is necessary in order that the two prototypes can be designated as holdouts when we come to identify their nearest neighbors. In this way, their data will not influence the calculations, as it is the rest of the market that we want to consider. Setting the *partition* value of the two holdout records to 1, while all the other records have a 0 in this field, enables us to use this field later when we come to set the focal records--the records for which we want to calculate the nearest neighbors.

Leave the table output window open for now, as we'll be referring to it later.

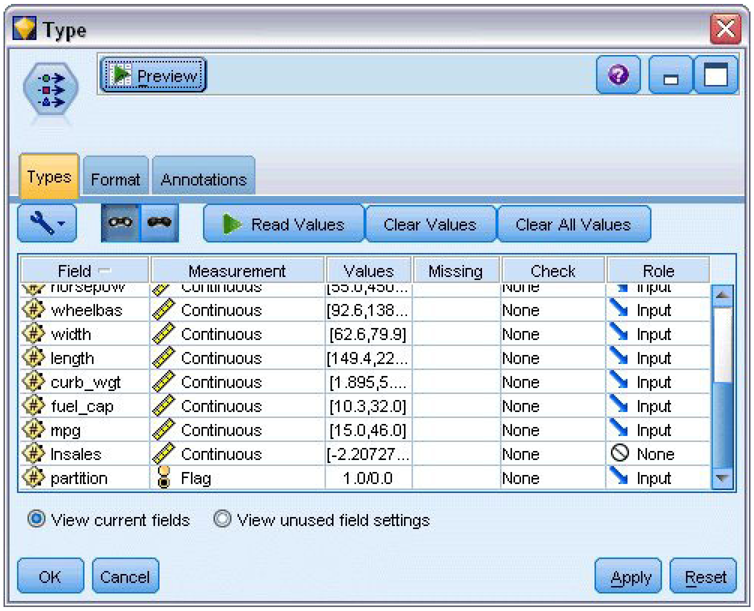


Figure 392. Type node settings

3. Add a Type node to the stream.
4. Attach the Type node to the Statistics File source node.
5. Open the Type node.

   We want to make the comparison only on the fields *price* through *mpg*, so we'll leave the role for all these fields set to **Input**.
6. Set the role for all the other fields (*manufact* through *type*, plus *lnsales*) to **None**.
7. Set the measurement level for the last field, *partition*, to **Flag**. Make sure that its role is set to **Input**.
8. Click **Read Values** to read the data values into the stream.
9. Click **OK**.

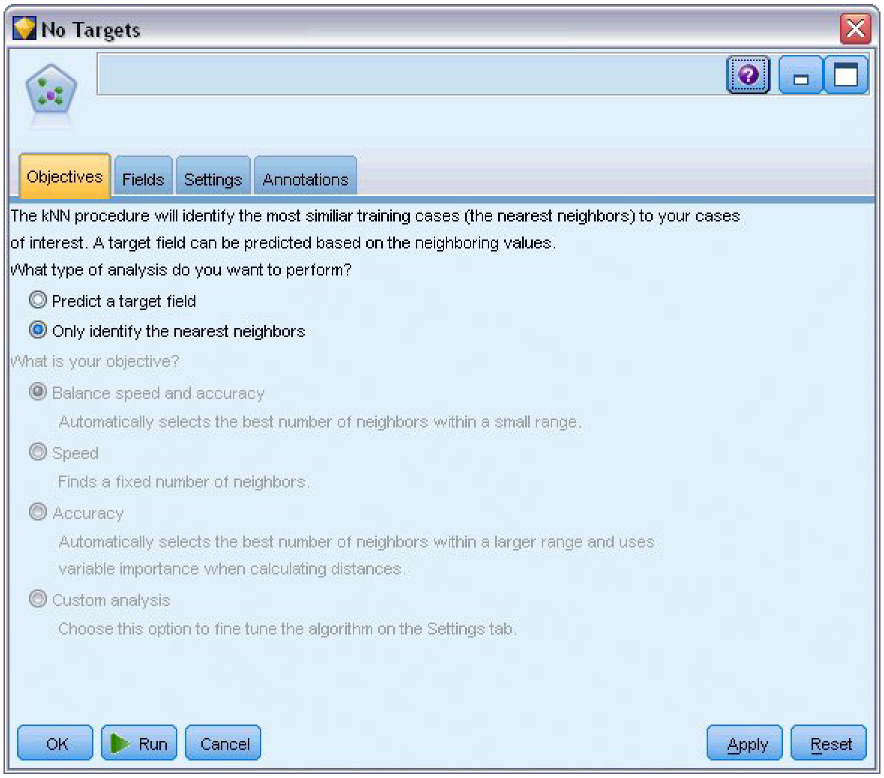


Figure 393. Choosing to identify the nearest neighbors

10. Attach a KNN node to the Type node.
11. Open the KNN node.

    We're not going to be predicting a target field this time, because we just want to find the nearest neighbors for our two prototypes.
12. On the **Objectives** tab, choose **Only identify the nearest neighbors**.
13. Click the **Settings** tab.

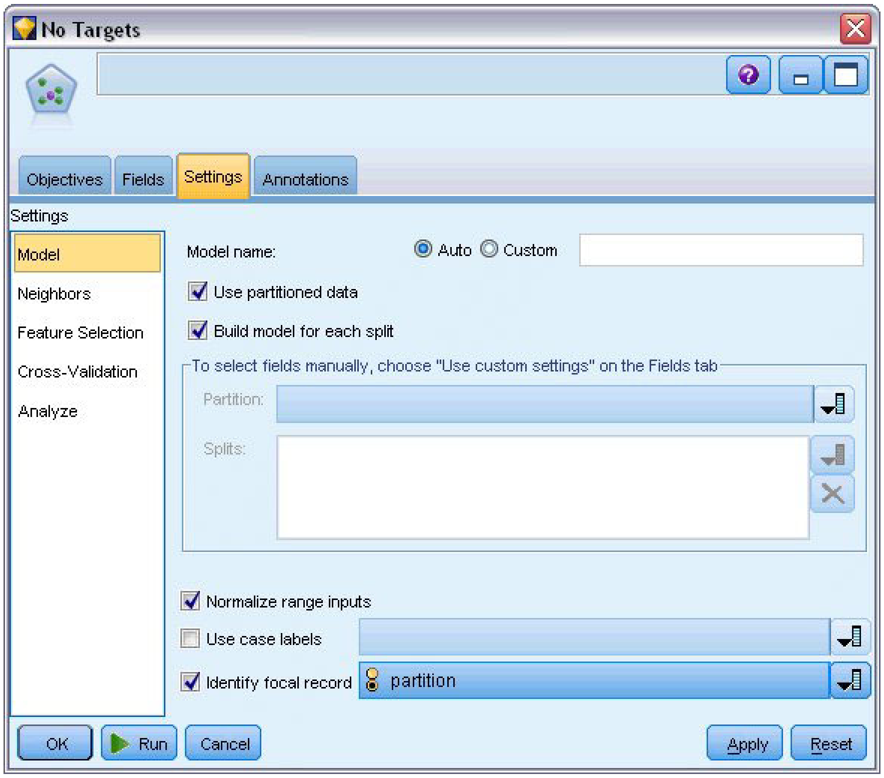


Figure 394. Using the partition field to identify the focal records

Now we can use the *partition* field to identify the focal records--the records for which we want to identify the nearest neighbors. By using a flag field, we ensure that records where the value of this field is set to 1 become our focal records.

As we've seen, the only records that have a value of 1 in this field are *newCar* and *newTruck*, so these will be our focal records.

14. On the **Model** panel of the **Settings** tab, select the **Identify focal record** check box.
15. From the drop-down list for this field, choose **partition**.
16. Click the **Run** button.

Examining the Output

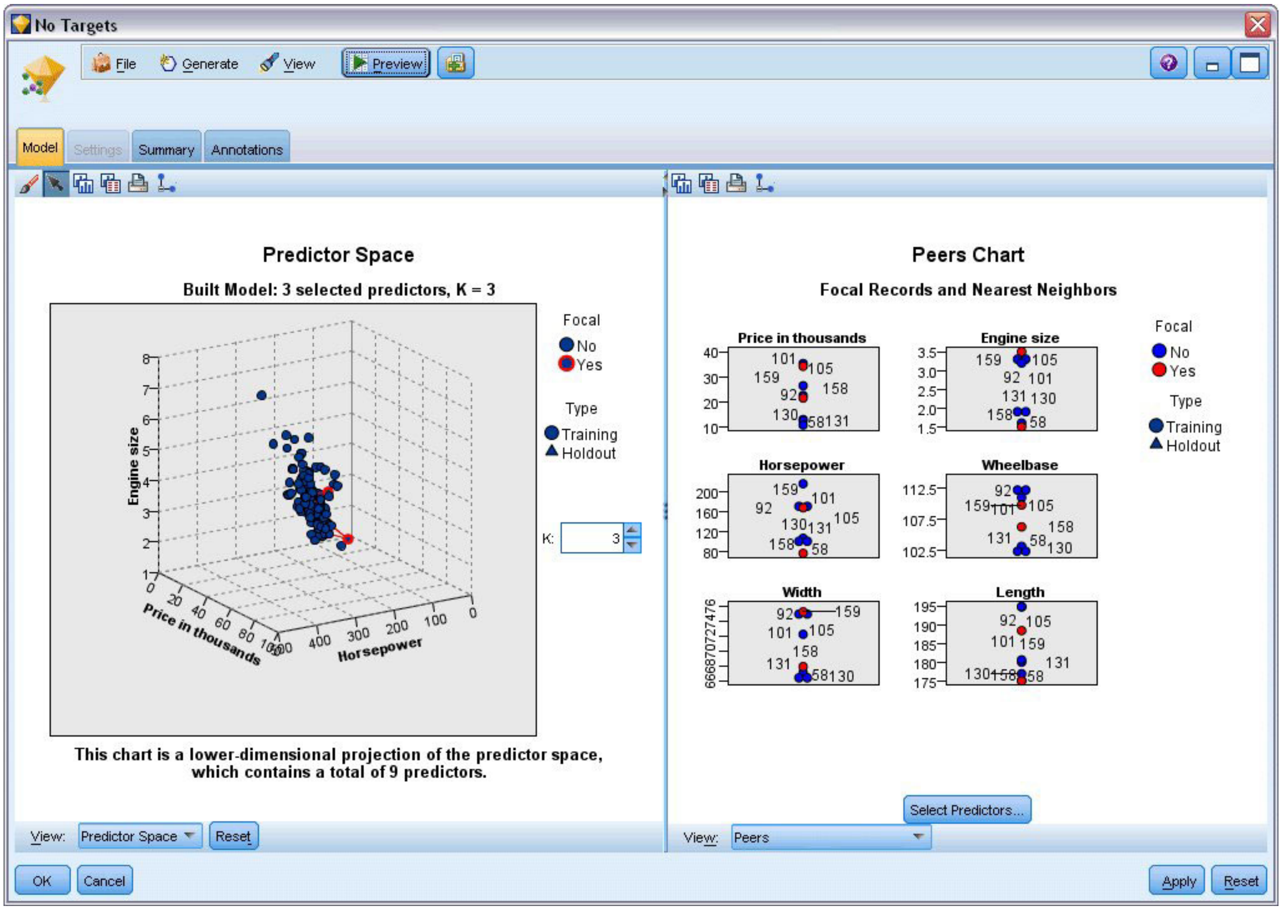


Figure 395. The Model Viewer window

A model nugget has been created on the stream canvas and in the Models palette. Open either of the nuggets to see the Model Viewer display, which has a two-panel window:

- The first panel displays an overview of the model called the main view. The main view for the Nearest Neighbor model is known as the **predictor space**.
- The second panel displays one of two types of views:

  An auxiliary model view shows more information about the model, but is not focused on the model itself.

  A linked view is a view that shows details about one feature of the model when you drill down on part of the main view.

Predictor Space

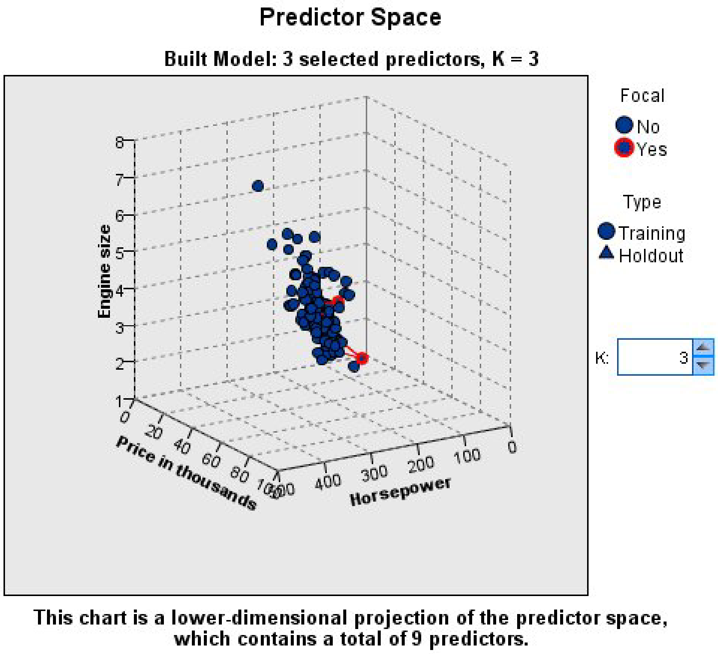


Figure 396. Predictor space chart

The predictor space chart is an interactive 3-D graph that plots data points for three features (actually the first three input fields of the source data), representing price, engine size and horsepower.

Our two focal records are highlighted in red, with lines connecting them to their *k* nearest neighbors.

By clicking and dragging the chart, you can rotate it to get a better view of the distribution of points in the predictor space. Click the **Reset** button to return it to the default view.

Peers Chart

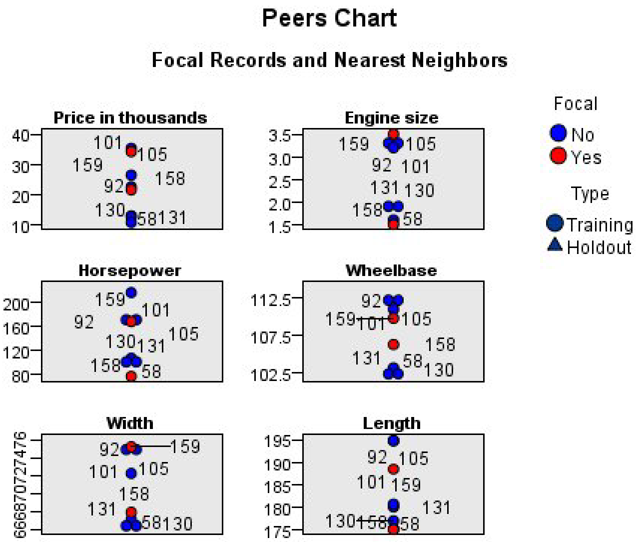


Figure 397. Peers chart

The default auxiliary view is the peers chart, which highlights the two focal records selected in the predictor space and their *k* nearest neighbors on each of six features--the first six input fields of the source data.

The vehicles are represented by their record numbers in the source data. This is where we need the output from the Table node to help identify them.

If the Table node output is still available:

1. Click the **Outputs** tab of the manager pane at the top right of the main IBM SPSS Modeler window.
2. Double-click the entry **Table (16 fields, 159 records)**.

   If the table output is no longer available:
3. On the main IBM SPSS Modeler window, open the Table node.
4. Click **Run**.

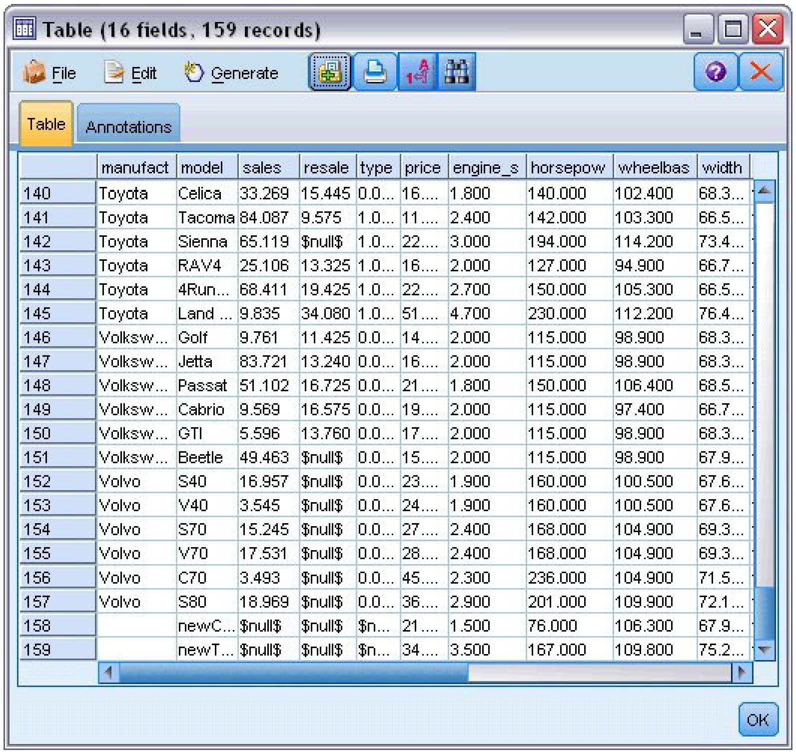


Table (16 fields, 159 records)

File Edit Generate

Table Annotations

	manufact	model	sales	resale	type	price	engine_s	horsepow	wheelbas	width
140	Toyota	Celica	33.269	15.445	0.0...	16....	1.800	140.000	102.400	68.3...
141	Toyota	Tacoma	84.087	9.575	1.0...	11....	2.400	142.000	103.300	66.5...
142	Toyota	Sienna	65.119	$null$	1.0...	22....	3.000	194.000	114.200	73.4...
143	Toyota	RAV4	25.106	13.325	1.0...	16....	2.000	127.000	94.900	66.7...
144	Toyota	4Run...	68.411	19.425	1.0...	22....	2.700	150.000	105.300	66.5...
145	Toyota	Land ...	9.835	34.080	1.0...	51....	4.700	230.000	112.200	76.4...
146	Volksw...	Golf	9.761	11.425	0.0...	14....	2.000	115.000	98.900	68.3...
147	Volksw...	Jetta	83.721	13.240	0.0...	16....	2.000	115.000	98.900	68.3...
148	Volksw...	Passat	51.102	16.725	0.0...	21....	1.800	150.000	106.400	68.5...
149	Volksw...	Cabrio	9.569	16.575	0.0...	19....	2.000	115.000	97.400	66.7...
150	Volksw...	GTI	5.596	13.760	0.0...	17....	2.000	115.000	98.900	68.3...
151	Volksw...	Beetle	49.463	$null$	0.0...	15....	2.000	115.000	98.900	67.9...
152	Volvo	S40	16.957	$null$	0.0...	23....	1.900	160.000	100.500	67.6...
153	Volvo	V40	3.545	$null$	0.0...	24....	1.900	160.000	100.500	67.6...
154	Volvo	S70	15.245	$null$	0.0...	27....	2.400	168.000	104.900	69.3...
155	Volvo	V70	17.531	$null$	0.0...	28....	2.400	168.000	104.900	69.3...
156	Volvo	C70	3.493	$null$	0.0...	45....	2.300	236.000	104.900	71.5...
157	Volvo	S80	18.969	$null$	0.0...	36....	2.900	201.000	109.900	72.1...
158		newC...	$null$	$null$	$n...	21....	1.500	76.000	106.300	67.9...
159		newT...	$null$	$null$	$n...	34....	3.500	167.000	109.800	75.2...

OK

Figure 398. Identifying records by record number

Scrolling down to the bottom of the table, we can see that *newCar* and *newTruck* are the last two records in the data, numbers 158 and 159 respectively.

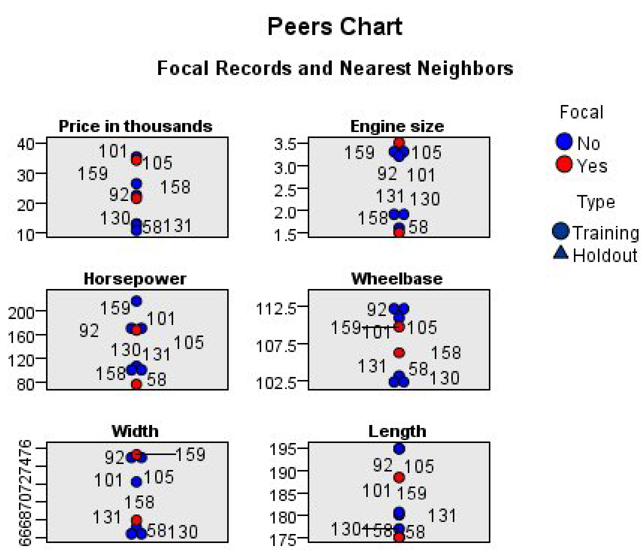


Figure 399. Comparing features on the peers chart

From this we can see on the peers chart, for example, that *newTruck* (159) has a bigger engine size than any of its nearest neighbors, while *newCar* (158) has a smaller engine than any of *its* nearest neighbors.

For each of the six features, you can move the mouse over the individual dots to see the actual value of each feature for that particular case.

But which vehicles are the nearest neighbors for *newCar* and *newTruck*?

The peers chart is a little bit crowded, so let's change to a simpler view.

5. Click the **View** drop-down list at the bottom of the peers chart (the entry that currently says **Peers**).
6. Select **Neighbor and Distance Table**.

Neighbor and Distance Table

k Nearest Neighbors and Distances

Displayed for Initial Focal Records

Focal Record	Nearest Neighbors			Nearest Distar	
	1	2	3	1	2
158	131	130	58	0.979	0.990
159	105	92	101	0.580	0.634

Figure 400. Neighbor and distance table

That's better. Now we can see the three models to which each of our two prototypes are closest in the market.

For *newCar* (focal record 158) they are the Saturn SC (131), the Saturn SL (130), and the Honda Civic (58).

No great surprises there--all three are medium-size saloon cars, so *newCar* should fit in well, particularly with its excellent fuel efficiency.

For *newTruck* (focal record 159), the nearest neighbors are the Nissan Quest (105), the Mercury Villager (92), and the Mercedes M-Class (101).

As we saw earlier, these are not necessarily trucks in the traditional sense, but simply vehicles that are classed as not being automobiles. Looking at the Table node output for its nearest neighbors, we can see that *newTruck* is relatively expensive, as well as being one of the heaviest of its type. However, fuel efficiency is again better than its closest rivals, so this should count in its favor.

Summary

We've seen how you can use nearest-neighbor analysis to compare a wide-ranging set of features in cases from a particular data set. We've also calculated, for two very different holdout records, the cases that most closely resemble those holdouts.

Notices

This information was developed for products and services offered worldwide.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Software Group
ATTN: Licensing
200 W. Madison St.
Chicago, IL; 60606
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Other product and service names might be trademarks of IBM or other companies.

Index

M

N

O

P

R

S

T

U

V

W

Z

Printed in USA