

Introduction to IBM SPSS Modeler v18

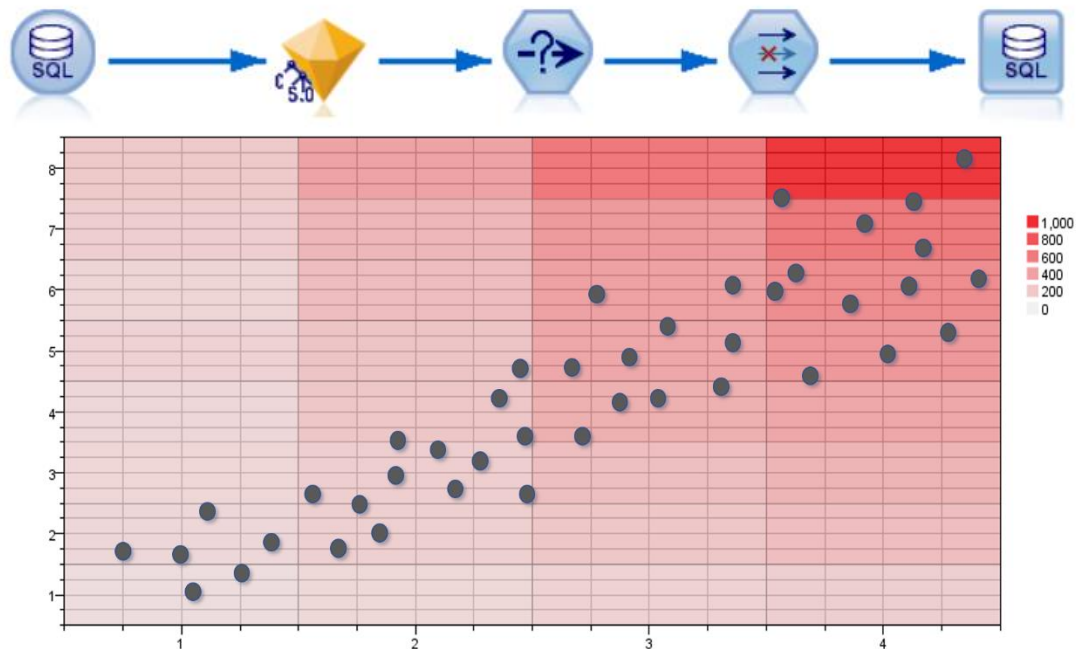


Table of Contents

Section 1: Working with IBM SPSS Modeler	1 - 1
Section 2: Reading Data Sources	2 - 17
Section 3: Data Understanding	3 - 45
Section 4: Restructuring Data	4 - 73
Section 5: Merging Data	5 - 99
Section 6: Changing and Creating Data	6 - 125
Section 7: Exploring Data	7 - 167
Section 8: Building a Predictive Model	8 - 203
Section 9: Assessing and Selecting Predictive Models	9 - 223
Section 10: Deploying Predictive Models	10 - 269
Practice Sessions	303

Section 1:

Working with IBM SPSS Modeler



- How Modeler is used
- CRISP-DM
- A first look at Modeler
- Working with Modeler
- Executing Streams
- A Predictive Analytics Tour

IBM® SPSS® Modeler is an analytical platform that enables organisations and researchers to uncover patterns in data and build predictive models to address key business outcomes. Moreover, aside from a suite of predictive algorithms, SPSS Modeler also contains an extensive array of analytical routines that include data segmentation procedures, association analysis, anomaly detection, feature selection and time series forecasting. These analytical capabilities, coupled with Modeler's rich functionality in the areas of data integration and preparation tasks, enable users build entire end-to-end applications from the reading of raw data files to the deployment of predictions and recommendations back to the business. As such, IBM® SPSS® Modeler is widely regarded and one of the most mature and powerful applications of its kind.

Modeler was originally developed and released under the name 'Clementine' by the UK software company ISL (Integral Solutions Limited). Clementine was one of the first true data mining platforms to incorporate the icon-driven user interface that so many predictive analytics and data mining platforms now utilise. In 1998 ISL was acquired by SPSS Inc, who later rebranded the software as SPSS Modeler. In 2009 SPSS Inc was itself acquired by IBM for \$1.2 billion.

How Modeler is used

Today Modeler is used extensively across many industries to address a very wide variety of business applications. These include:

- Predicting which current customers are likely to cancel their subscriptions so that phone or media companies may proactively intervene to retain their custom
- Identifying key equipment or assets at risk of failure so that utility companies may deploy engineering staff to prevent it occurring
- Using association analysis to uncover which groups of products are often purchased on the same occasion so that retailers can design promotions to drive more sales

- Predicting which patients are at risk of being re-admitted to hospital following discharge so that medical professionals can provide proactive care
- Early identification of fraudulent transactions for hotel companies so that they may take immediate action to mitigate the damage
- Segmenting donor behaviour for charities so that they may interact with their supporter base in a manner that reflects their personal profile

The Premium edition of IBM® SPSS® Modeler includes sophisticated Text Mining functionality. This additional functionality is used to classify and categorise unstructured data from sources such as call centre comments, social media feeds and document collections. Modeler Text Mining is also used to mine and extract data from engineer reports, error messages, insurance claims and medical notes to enhance the accuracy and clarity of predictive models.

CRISP-DM

Despite the media's recent burgeoning interest in all things analytical, among the thousands of articles and discussion pieces devoted to big data analytics, data science, machine learning and artificial intelligence, very few dwell upon the practical aspects of ensuring analytical applications deliver tangible results, preferring instead to focus on the power and pitfalls of algorithms. Nevertheless, experienced data analysts are acutely aware that the success or failure of predictive analytics projects often rest on aspects of the process that have little to do with data modelling per se. In fact, the CRISP-DM methodology describes a process that many expert analysts follow when designing initiatives driven by predictive analytics. CRISP-DM stands for Cross Industry Standard Process for Data Mining. It is described as 'a hierarchical process model' for data mining applications, although in this context the term 'data mining' may be taken to mean 'predictive analytics'.

As can be seen in figure 1.1, CRISP-DM describes the predictive analytics process in terms of six main phases. A full description of the methodology explains each of these phases in greater detail and in fact each one can be broken down hierarchically into a set of tasks (generic tasks, specialized tasks and process instances). For more details, visit the website <http://www.crisp-dm.eu>.

As figure 1.1 illustrates, the process of developing predictive analytics applications is cyclical in nature as each iteration seeks to correct faults and improve the outcomes. Moreover, you may notice that 'Modelling' represents only *one* of the phases in the development of the application.

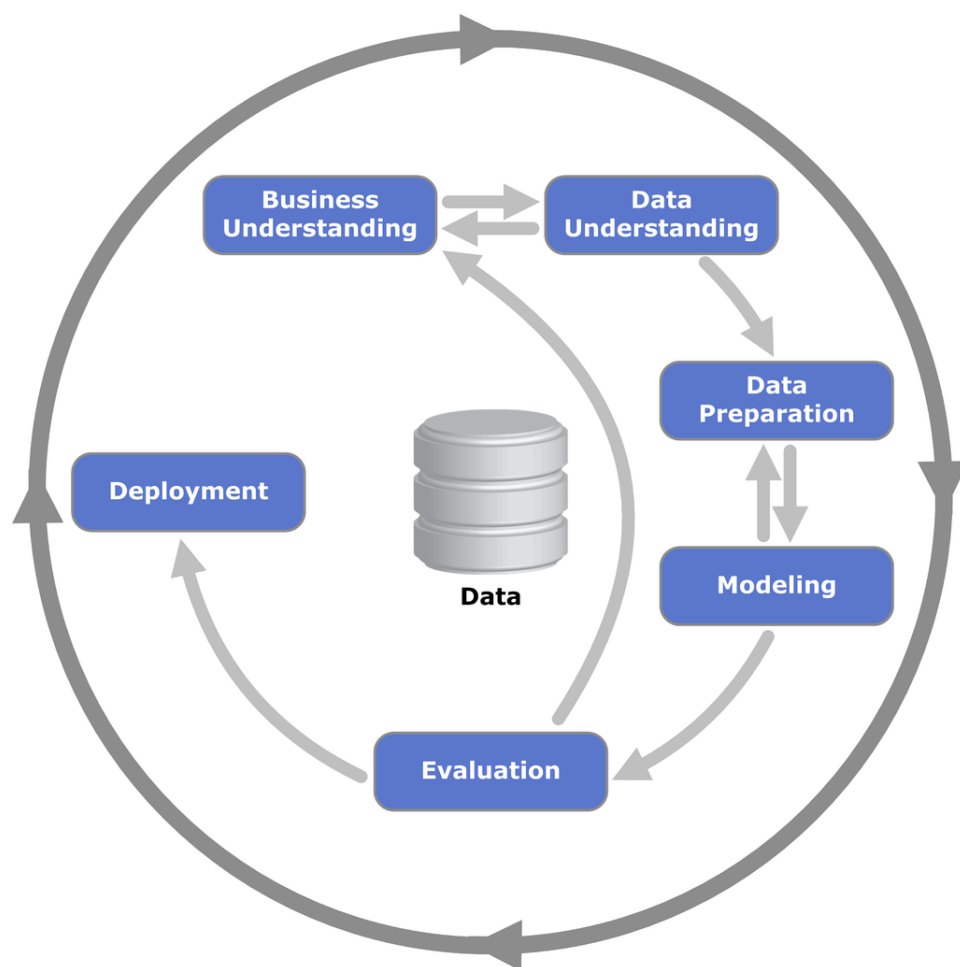


Figure 1.1 The CRISP-DM Process

Business Understanding

The first step in a new Predictive Analytics initiative is the Business Understanding phase. In fact, this is a critical step and one which is all too often glossed over. Spending time scrutinising *why* a particular subject has been chosen as the focus of a project may often yield valuable results. If for example, we are trying to predict which individuals are likely to renew a contract or insurance policy, how accurate should the model be in order to proceed? Among the predictions, how many false positives or false negatives are acceptable? What is the current policy renewal rate? What are the costs and benefits associated with say a 10% decrease or increase in renewals? This phase demands that we determine our success criteria early on. Furthermore, the Business Understanding phase tasks us with determining our project resources as well as the risks and assumptions that underpin the project. In fact, many initiatives are abandoned when subject to this level of scrutiny or are re-focussed on another aspect of the original business objective. Finally, Business Understanding requires us to produce an agreed upon project plan that documents all these elements.

Data Understanding

The Data Understanding phase aims to provide a thorough audit of the project's primary raw material: data. It is likely that the available data are in more than one format. That the quality and quantity of the data will need to be inspected. The nature of the data, whether structured (quantitative) or unstructured (free text/audio/image) will need to be given due consideration. What do the actual fields within the data refer to? How many data are missing? How often are the data sources updated? If we are dealing with data that are related to customers or clients, then are there any pertinent aspects of the customer/client behaviour or demographics that we don't have access to? In short, we are attempting to document the limitations and challenges associated with the project data early in the cycle so that we might better understand our chances of success.

Data Preparation

Many analysts regard the Business Understanding and Data Understanding phases as the most critical to the success of a project. However, many more analysts may agree that the Data Preparation phase is most time-consuming. Data preparation takes up time because it often involves numerous tasks associated with preparing data that *was not originally collected with analysis in mind*. As such, tasks such as data cleaning, merging sources, transforming file structures and deriving new measures are often among the most onerous in the project life-cycle. Often the analyst has a rough idea as to what the final data should look like in order to be 'fit for modelling'. This may require, for example, the transformation of the data so that each row represents an individual person rather than a transaction, or a website visit or a product.

Modelling

As the name implies, the modelling phase involves the development of a model that addresses the project objectives. In fact, many modelling techniques allow analysts to build models automatically and quickly. This means that much of the time in this phase is taken up with trying different modelling techniques and tuning the selected models. Tuning the models may result in more useful results but so will transforming and manipulating the input data, so this phase has its own feedback arrow to the previous Data Preparation phase.

Evaluation

Having developed a (hopefully promising) model or set of models, the analyst must take time to assess its usefulness. There are many ways in which data analysts view the utility of a model, whether in terms of its complexity, clarity, accuracy or stability. In this phase, the analyst must also evaluate the model in terms of the original project objectives as documented in the Business Understanding phase. Does the model meet the performance

criteria as determined at the outset of the initiative? Clearly, this a critical phase of the project and it may be that the evaluation indicates that the project should iterate back to the initial phase of the process and begin again from the start.

Deployment

The Deployment phase sees fruits of all the previous work deployed into the real world. It is here that new data is fed into the model and the generated outcomes, in the form of risk scores, likelihood values, customer segments, classifications, anomaly indices, probabilities or forecasts are used influence decisions and actions. It may be that the initial deployment is done on the basis of a control and test scenario or that only a small proportion of the decisions are affected, but in either case, the results must be carefully monitored. In fact, it makes sense that the Deployment Phase includes the opportunity to create a maintenance plan for the model(s). After all, any predictive analytics model is unlikely to provide useful results indefinitely, so plans must be made to when and how often the model should be updated with more recent information or replaced entirely.

A first look at Modeler

Figure 1.2 shows an overview of the Modeler interface. We can see from the image that there are a number of elements that make up the application. It's worth taking a moment at this stage to familiarise ourselves with the key aspects of the Modeler UI.

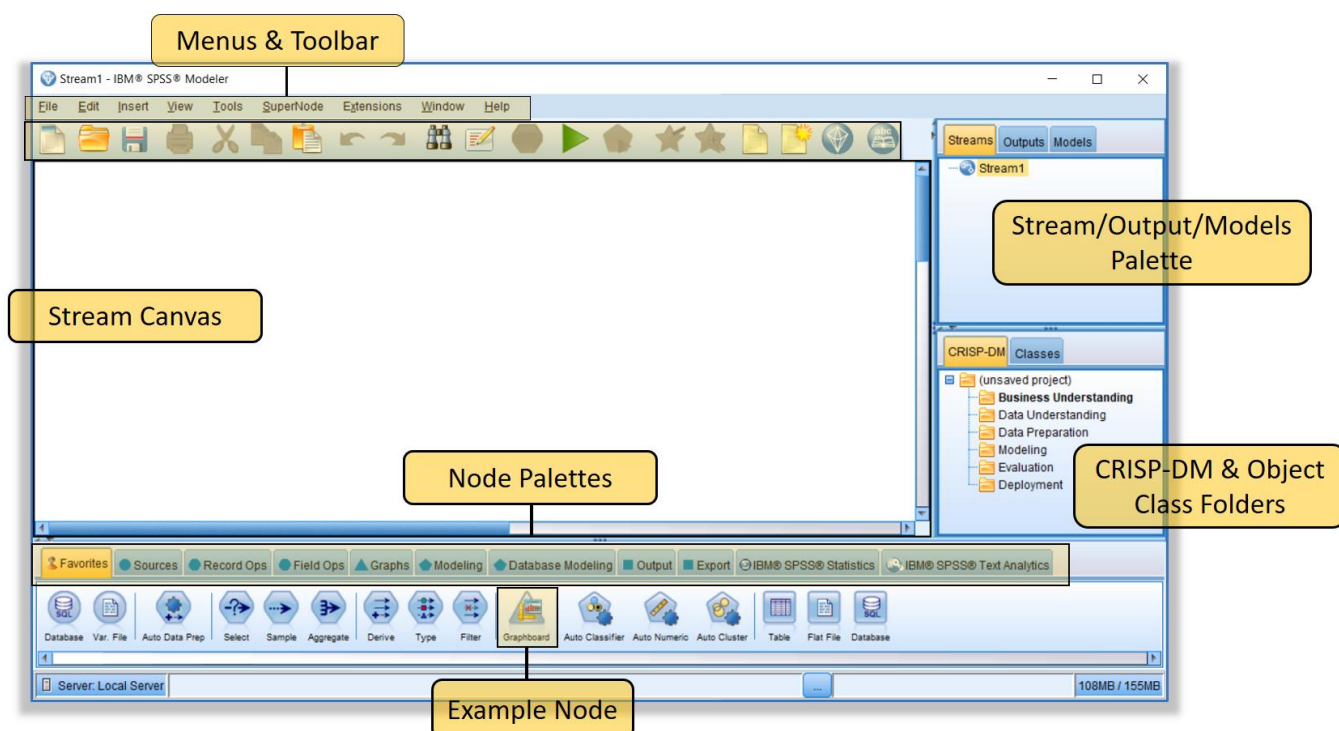


Figure 1.2 The Modeler Interface

The Menus and Toolbar – As with many analytical platforms, SPSS Modeler allows us to request specific procedures from a series of drop-down menus or by using the toolbar to carry out frequently used procedures such as saving or opening Modeler streams, copying and pasting nodes or executing procedures. The menus often allow us an alternative way to build up streams instead of selecting specific nodes from the palette tabs along the bottom of the interface.

The Stream Canvas – This is the main part of the Modeler interface. The stream canvas allows us to build up a series of procedures using nodes. We can think of modeler streams as *visual programs* that allow us to carry out multiple tasks.

The Node Palettes – The palettes organise the different procedures in Modeler (represented by nodes) into their respective functional tabs. Examples include the Source nodes palette which contains a several nodes for reading different kinds of data files, the Record Ops palette which contains a number of data preparation nodes that affect how data rows are represented or the Modeling palette which contains a number of algorithms that are used to generate new models.

The Stream/Output/Models Palette – Here Modeler has created three palettes that allow use to view and switch between three different kinds of files: Modeler streams that are open in the application; any outputs that we have created from running a stream; and any models that we have created as a result of running an algorithm within Modeler.

The CRISP-DM and Object Class Folders – this section of the interface allows us to organise the different documents, streams, models and outputs associated with a project into custom folders. By default, Modeler creates a folder structure according to the CRISP-DM process, but these may be customised to suit each project. This collection of folders and files can be saved as a project file (*.prj) so that users may have easy access to multiple files and outputs even if they are saved in separate locations (see figure 1.3).

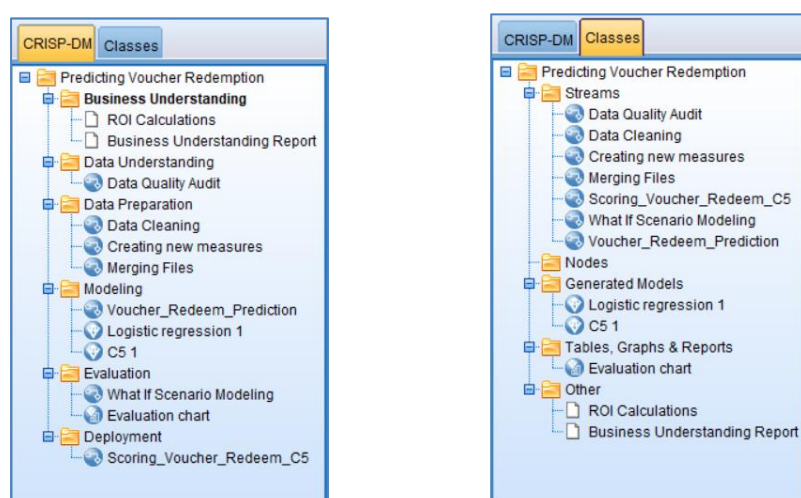


Figure 1.3 The CRISP-DM and Object Class folders

Working with Modeler

As you may have gathered, Modeler works by executing connected nodes that are joined together in a stream. The stream itself is the main file type that most users work with. There are various conventions associated with stream execution but the main one is that the execution begins with data being read from a source node (which appears as a circle) and ends with a node that generates some sort of output. Terminating a stream with a node that doesn't have a way to display its results will generate an error. Figure 1.4 shows the three main node types that create results: an output node (square), a chart node (triangle) and a model generating node (a pentagon). As an illustration, you can see that the figure includes an 'intermediary' select cases node (hexagon). You can't end a stream with this type of node because its job is simply to filter out rows of cases and it needs another node type (such as a chart or data table) to display the result.

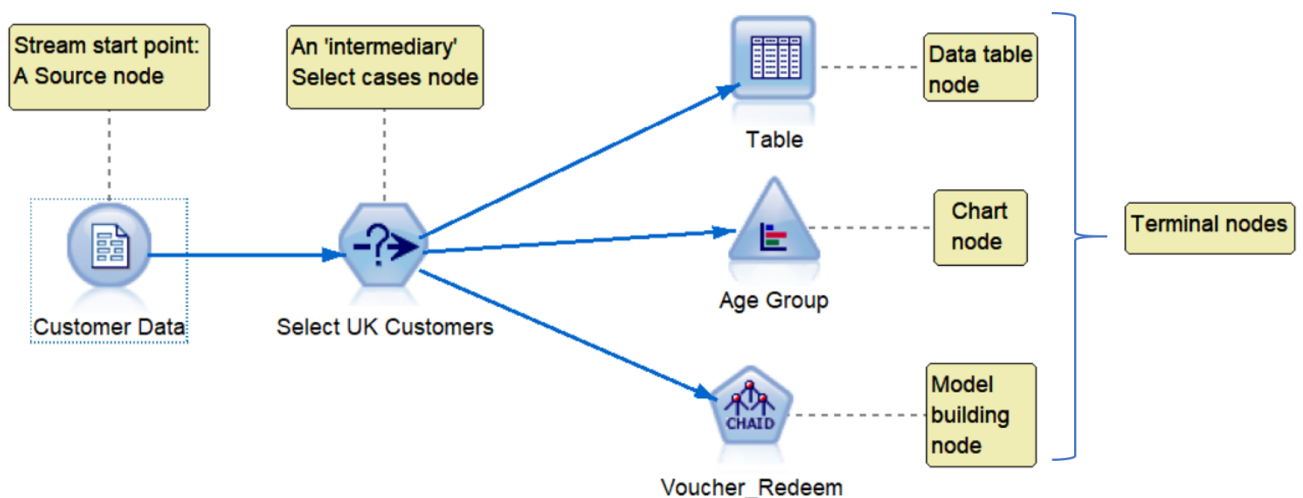


Figure 1.4 Typical Modeler Stream Structure

As we will see, these streams can be saved with the file extension '.str', but Modeler allows individual nodes, outputs and models to also be saved as files as well. Each object type has its own file extension as illustrated in figure 1.5.

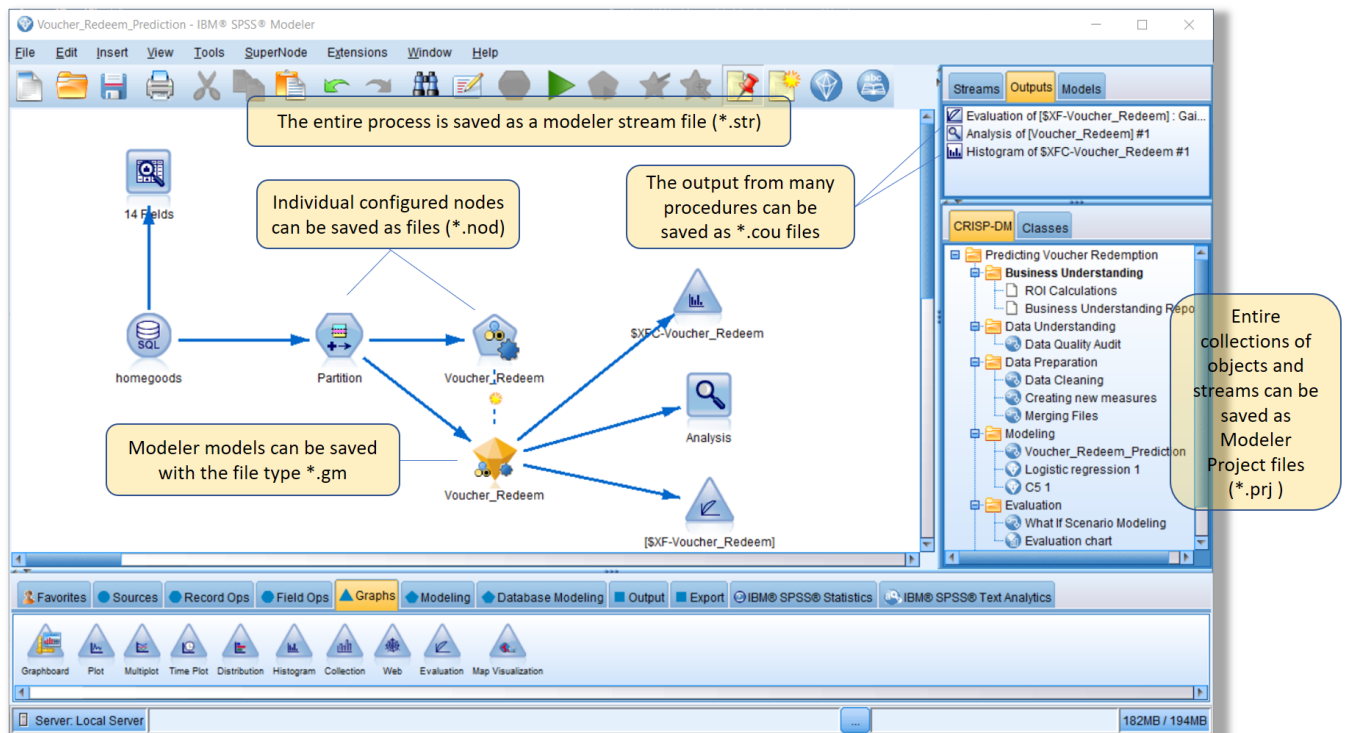


Figure 1.5 File types within the modeler interface

Streams are built up by placing nodes on the stream canvas and defining logical connections between them. The nodes are edited to ensure that they run in an appropriate fashion. The stream is then executed and the stream outputs are generated.

Starting with an empty stream, we can place nodes in one of two ways on the canvas.

- Drag the selected node from the palette below onto the canvas
- Double-click the node and it will automatically appear on the canvas

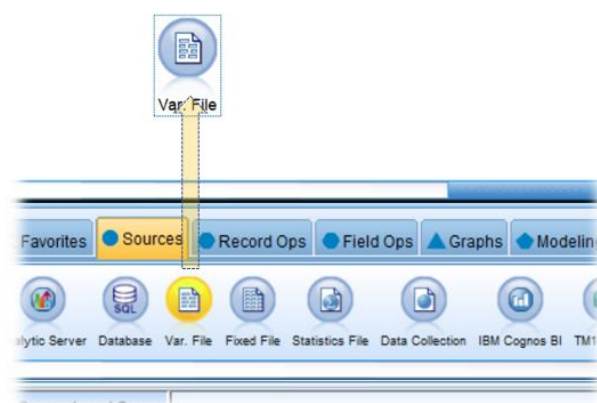


Figure 1.6 Placing nodes on the Modeler Stream Canvas

To delete a node from the canvas, simply highlight it and click delete.

There is also more than one way to define the connections between the nodes. To connect one node to another, the direction must be in the order that the data flows (from source to output). Furthermore, the connection itself must adhere to Modeler's own logical conventions: for example, you can't connect two output-generating nodes to each other.

To define a valid connection, choose one of the following options:

- Hold the *middle* mouse button down and draw the connection between the two nodes
- Click to select the first node in the connection then *double-click* the connecting node in its palette to automatically join the two together
- Select and right-click on the first node and choose 'Connect' from the drop-down menu then click the second node
- Click the first node, press F2 on the keyboard and click the second node

To delete a connection, you may either:

- Hover your mouse pointer over the connection line and right click. From the pop-up menu choose 'Delete Connection'
- Select and right-click on either node and choose 'Disconnect' from the drop-down menu
- Click the either node, press F3 on the keyboard

To insert a node into an existing connection, you have a couple of options: either delete the existing connections between the nodes and connect the new node between the existing ones using one of the methods above, or simply click with the *middle mouse button* on the existing connecting line and drag it over the intermediary node to route the connection through it (this is much quicker). Figure 1.7 illustrates this.

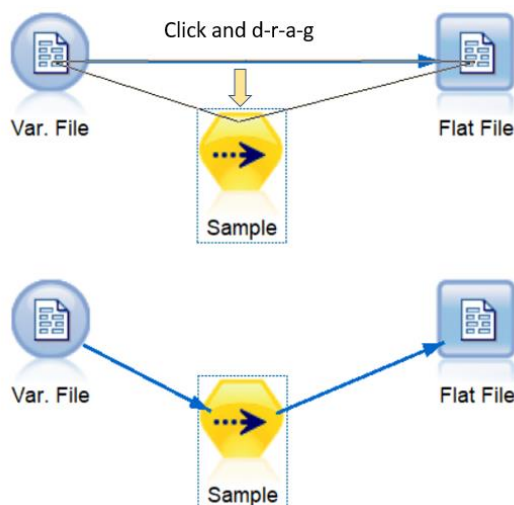


Figure 1.7 Inserting a node into an existing connection

Executing Streams

In order for streams to be executed, the nodes need to be correctly configured. Later in the course we will look at how we can edit the settings in individual nodes to make them work with the data appropriately. Figure 1.8 shows a stream with the nodes already configured to execute correctly.

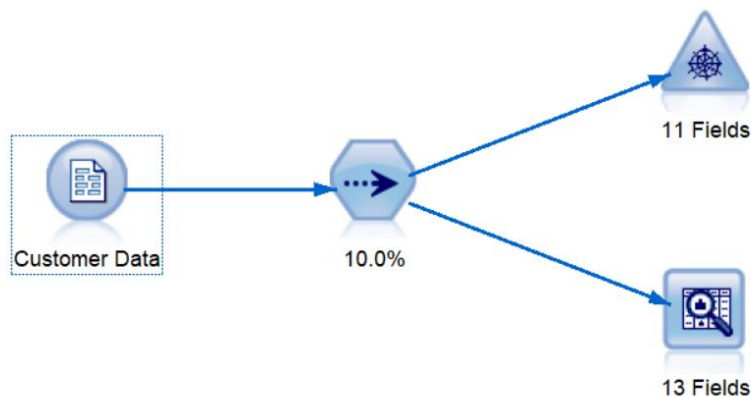


Figure 1.8 Pre-configured stream ready for execution

To execute the entire stream, we could select the source node and right-click to generate a drop-down menu. As figure 1.9 shows, you can select the option 'Run From Here' and all nodes downstream from the source will be executed.

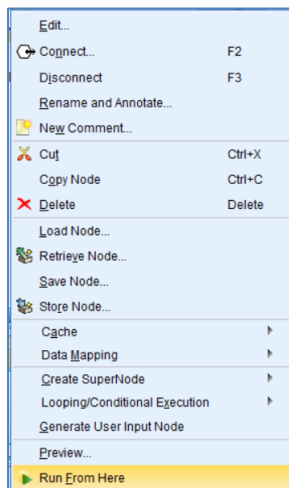


Figure 1.9 Executing a stream from a source node

The same effect is achieved by clicking the green triangle on the toolbar:



As this button simply executes the entire stream.

If however, you wish to only execute one branch of the stream, you can highlight the terminal node of the required branch and click the yellow run selection button:



Figure 1.10 illustrates this process.

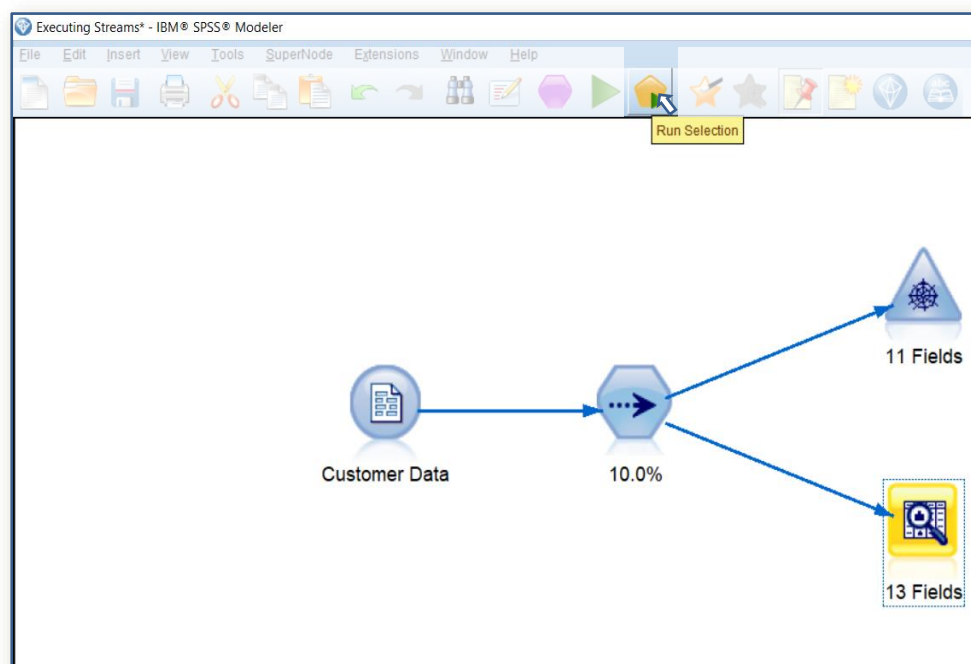


Figure 1.10 Executing one branch of a stream

The same effect can be achieved by highlighting the required terminal node, right-clicking and selecting 'Run' from the drop-down menu.

A Predictive Analytics Tour

Earlier in this section we looked at the CRISP-DM methodology that is so strongly associated with the Modeler platform. CRISP-DM provides us with an overview of the entire process of planning and executing a predictive analytics project. In this last topic, we will turn our attention to how one might build, evaluate and deploy a model within the Modeler itself. To illustrate this, we can build up a detailed example in a step by step fashion using Modeler's 'Insert' menu.

To start the process, we will need to read some data and check its suitability for modelling. Beginning with a blank stream canvas (clicking File > New Stream will create one), we can insert the first part of the example. In this first instance, we will insert part of a stream that reads some sample data for the demonstration.

From the main menu click:

Insert

Stream from File

Browse to 'C:\SV Training\Modeler Intro\Section1'

From within the folder choose the stream:

'PA Tour Part 1.str'

Insert

The first part of the demonstration stream is inserted. It shows two separate data source nodes configured to read two different data file types. The data are then merged together to create a single file. The resultant merged file can then be explored using a special output node (the Data Audit node). If we execute the stream, we can see that the data contains a number of fields of various types. Each field is represented by a chart that is coloured by the two values of a *target* variable called 'Churned Customer'. So we can see that this is a stream where the analyst probably wants to understand what drives customer defection and wishes to predict which customers are likely to defect in future. Figure 1.11 illustrates this.

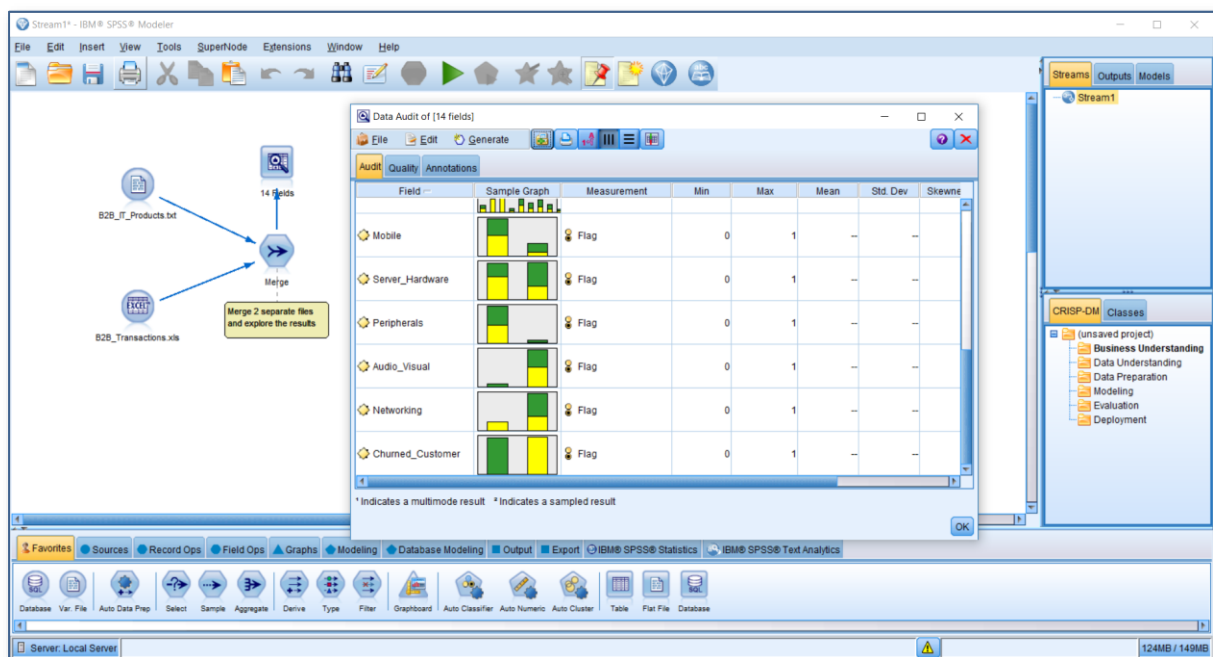


Figure 1.11 PA Tour Part 1 with results from Data Audit node

As we will see later, the Data Audit node is a valuable way to assess the quality and suitability of our data for modelling purposes. Having checked the data with it, we can move to the next stage where we will try to build a predictive model. From the main menu click:

Insert

Stream from File

'PA Tour Part 2.str'

Insert

Now another section of Modeler stream has been inserted. We can join the first part of the stream to this section by connecting the Merge node and the Partition node together. Using the nodes in this second section, we can build a model to predict customer churn. Notice that the stream contains a Partition node that enables us to randomly split the data into two parts. One part of the dataset will consist of rows of customers that may be used to 'train' the model (the 'Train' partition) the second part of the dataset will contain customers that have not been used in the model building process but instead are withheld to check the accuracy of the model. The second node is a type of model-building algorithm known as a decision tree. If we execute this node, the algorithm quickly builds a model represented by a 'golden nugget'. We can browse this model to see what it contains and gain a deeper understanding of how it attempts to predict the outcome. Figure 1.12 illustrates this.

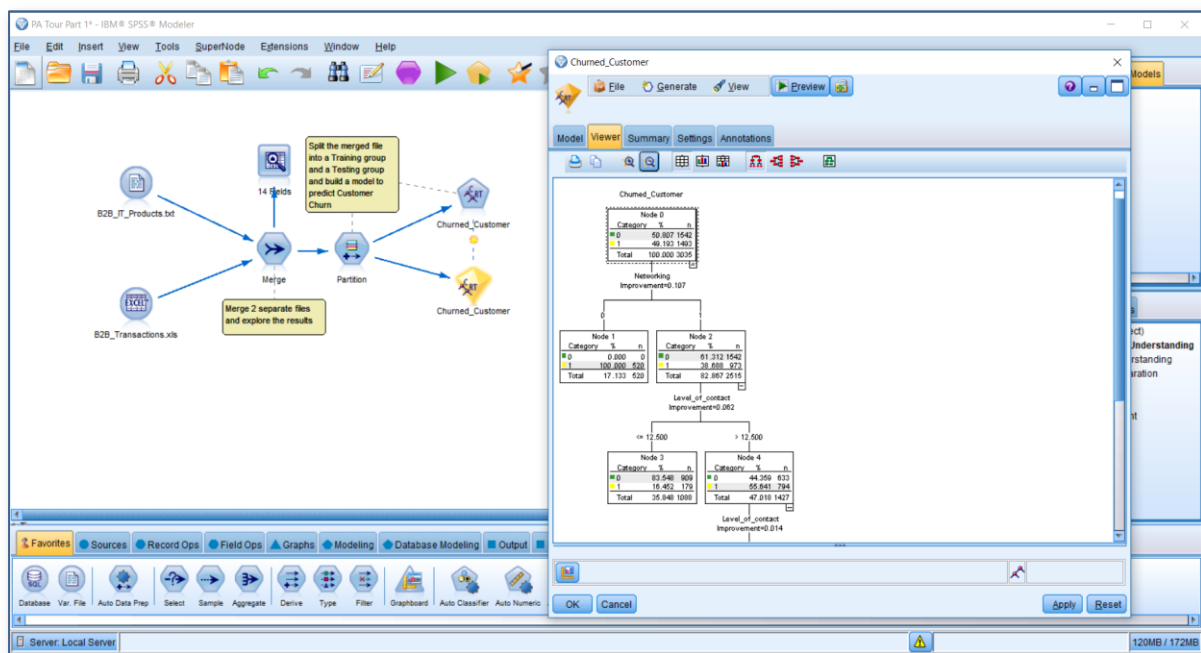


Figure 1.12 PA Tour Part 2 with results from the decision tree model

Having inspected the model, the data analyst at this stage might wish to assess how accurately it manages to predict churn in both the 'Training' and 'Testing' groups as defined by the Partition node earlier.

To continue, from the main menu click:

Insert

Stream from File

'PA Tour Part 3.str'

Insert

A third section of Modeler stream has now been inserted. In this case it contains two nodes that are explicitly designed to assess the performance of predictive models. By connecting the model nugget to these output-generating nodes, we can gain an understanding of how accurate the model is. By executing the Analysis node, we are shown a group of tables that tell us the model was nearly 77% accurate in the Training partition and around 75% accurate in the Testing group. We would conclude from this that the model seems to perform well on test data. The triangular 'Evaluation' chart also shows how the model performs on both partition groups. The chart it creates shows how much better the model is at identifying customers who have churned as compared to a random selection or indeed a perfect prediction. Figure 1.13 illustrates this.

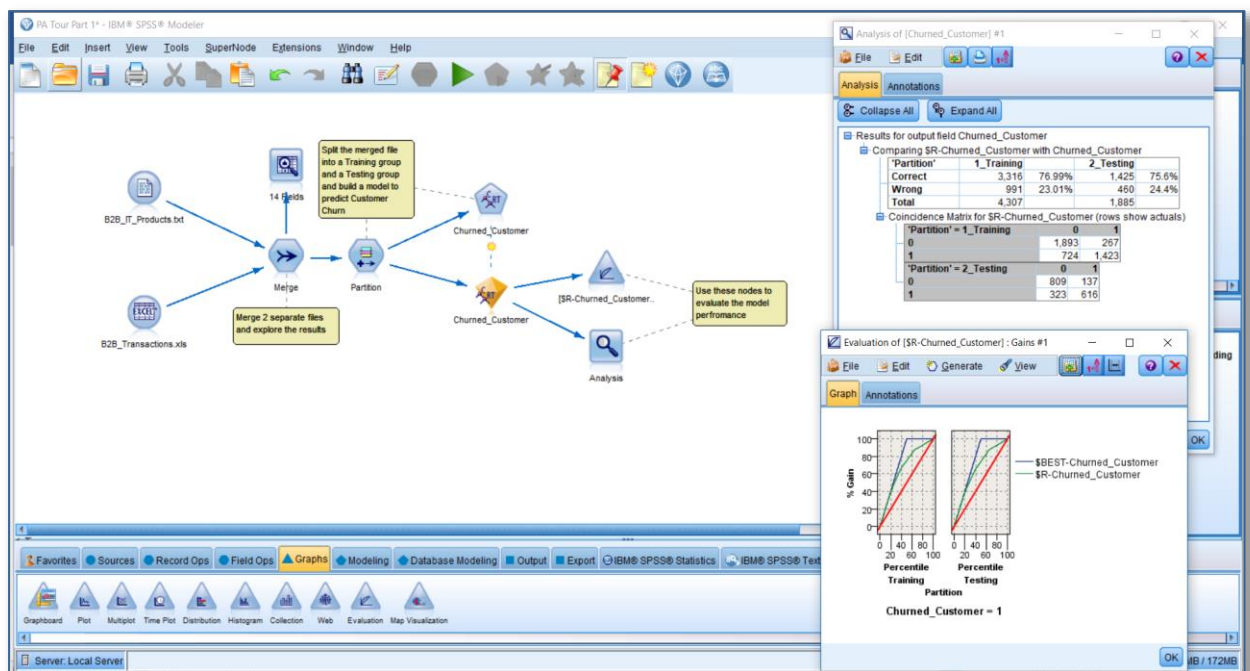


Figure 1.13 PA Tour Part 3 with results from the Analysis and Evaluation nodes

At this stage, the Data Analyst may conclude that the model is sufficiently useful to deploy back to the business. This will require the analyst to apply the model to data about the *current customers* to generate predictions as to which ones are likely to defect in the near future. To show how this is commonly done, from the main menu click:

Insert

Stream from File

'PA Tour Part 4.str'

Insert

The final section of the Modeler stream uses a new data source, this time containing only current customers. To apply the model we created earlier to this data source, we need only copy and paste the nugget and connect it to the new Source node and the downstream Filter node. As soon as we do so, we are able to execute the Histogram node to see the range of 'Churn Likelihood' values that the model generates having 'scored' the new data source. By interacting with the histogram, we can select all those current customers that the model scores as having a churn likelihood greater than 0.6 (60%). This process generates a Select node that filters out customers who scored lower than 0.6 and when connected to the final Table node allows us to view a list of around 1200 customer IDs with their respective likelihood scores, all of whom could be meaningful targets for a proactive retention campaign aimed at keeping their custom. Figure 1.14 illustrates this.

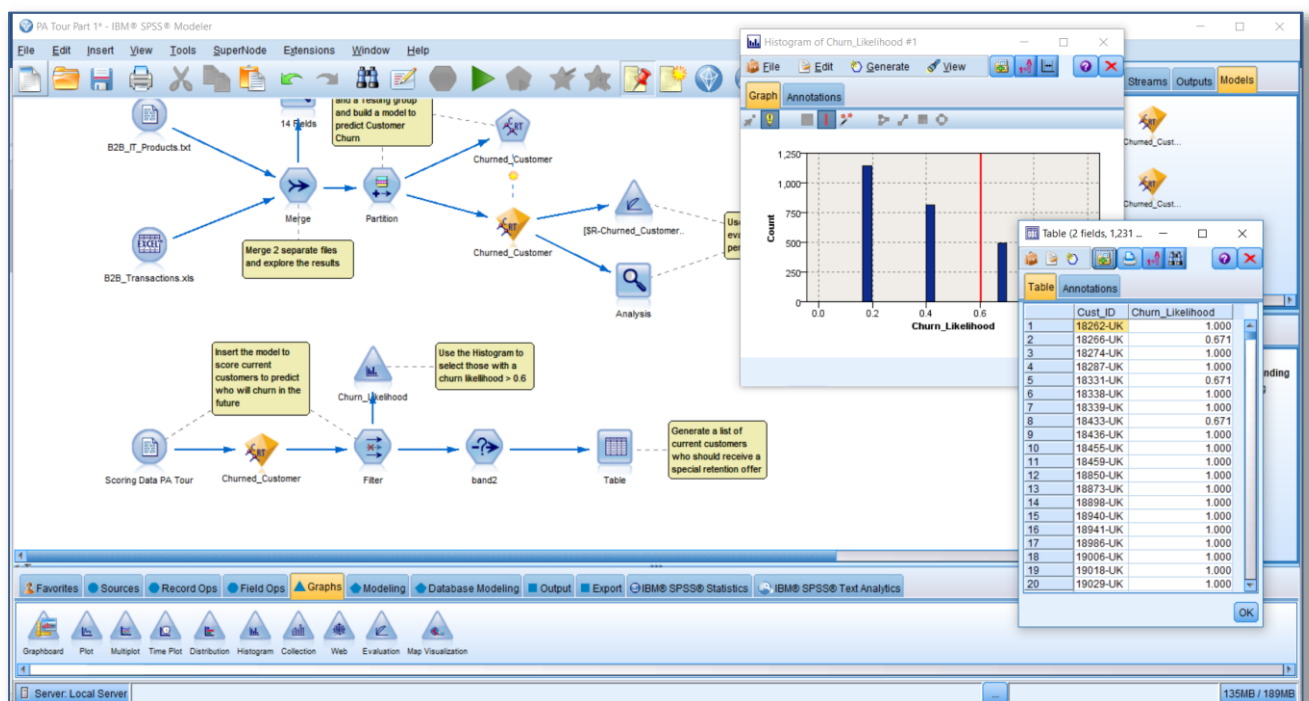


Figure 1.14 PA Tour Part 4 with results from scoring current customers using the earlier churn prediction model

We can see from the demonstration that the process of building, evaluating and deploying a model is very straightforward in Modeler. In fact, Modeler automatically takes care of a lot of the mapping of variables to the appropriate settings in the next node. As we progress through the rest of the course, we will see many other examples where Modeler is able to greatly enhance the productivity of users as they build analytical applications.

Section 2:

Reading Data Sources



- Reading Excel Files
- Reading Text Files
- The Generate Menu
- The Select Node
- Reading Database Tables

One of the more challenging realities of working in Predictive Analytics is that normally we are not working with data sources that have been designed with analysis (let alone advanced analytics) in mind. Not only will the analyst have to contend with the fact that the data sources at their disposal have in fact been specifically configured for purposes as diverse as customer billing, inventory admin, outbound marketing, supply chain management, CRM, workforce administration, asset registration or at best, business intelligence reporting, but furthermore, they may also find that the sources themselves are stored in many different formats such as proprietary databases, text files, spreadsheets or xml. So the first technical hurdle that we often face is figuring out how to correctly read the data. Fortunately, Modeler has strong capabilities with regard to parsing and reading a wide range of data file types.

Reading Excel Files



Importing data from Excel files is usually a relatively straightforward process in Modeler. One thing to be aware of is that the user is required to specify whether the file has the spreadsheet '.xls' or workbook '.xlsx' extension. It's important to understand that when reading files 'into' Modeler we are not importing them into some kind of internal database or converting them to a native file format. Although the process may require the creation of temporary files, generally speaking, the data files themselves remain in their original location.

Starting with a blank canvas, we can navigate to the Sources palette within Modeler and either:

Double-click the Excel node or click and drag it onto the canvas

Having placed the node on the canvas, we can configure it to read an Excel file. To do so:

Right-click on the Excel Source Node and from the pop-up menu choose 'Edit'

Figure 2.1 shows the Excel node with the associated control menu.

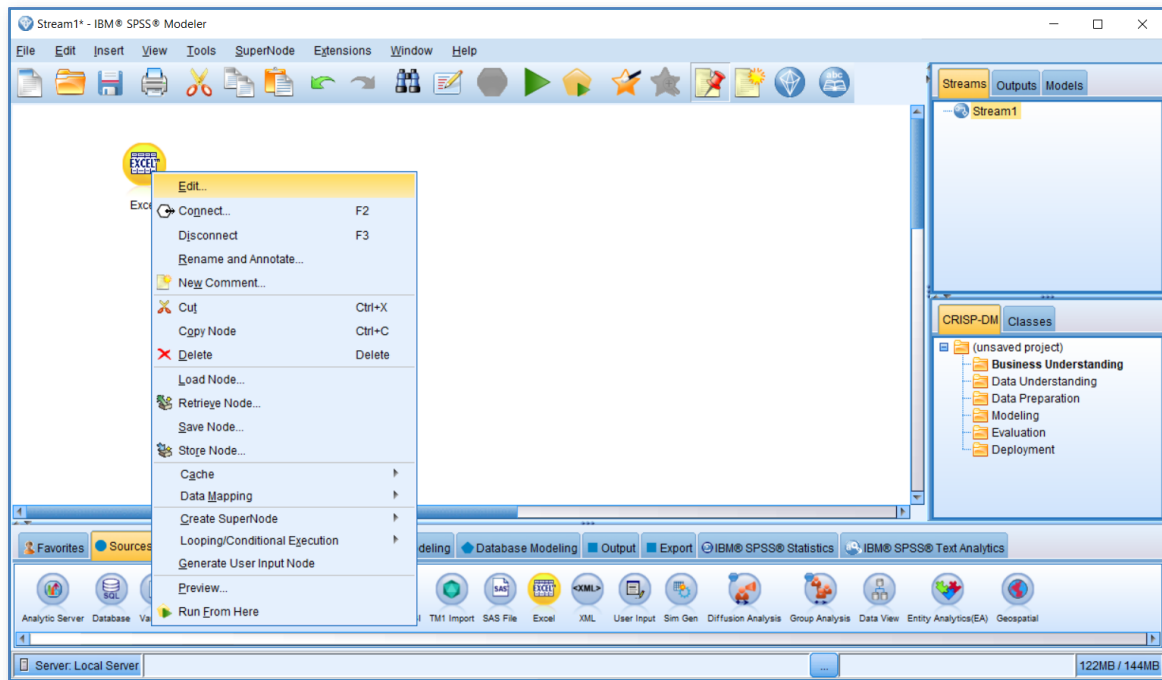


Figure 2.1 Excel Source Node with its associated control menu

By right-clicking and choosing 'Edit' in any Modeler node, we gain access to that particular node's unique configuration dialog. Figure 2.2 shows the associated control dialog for the Excel node.

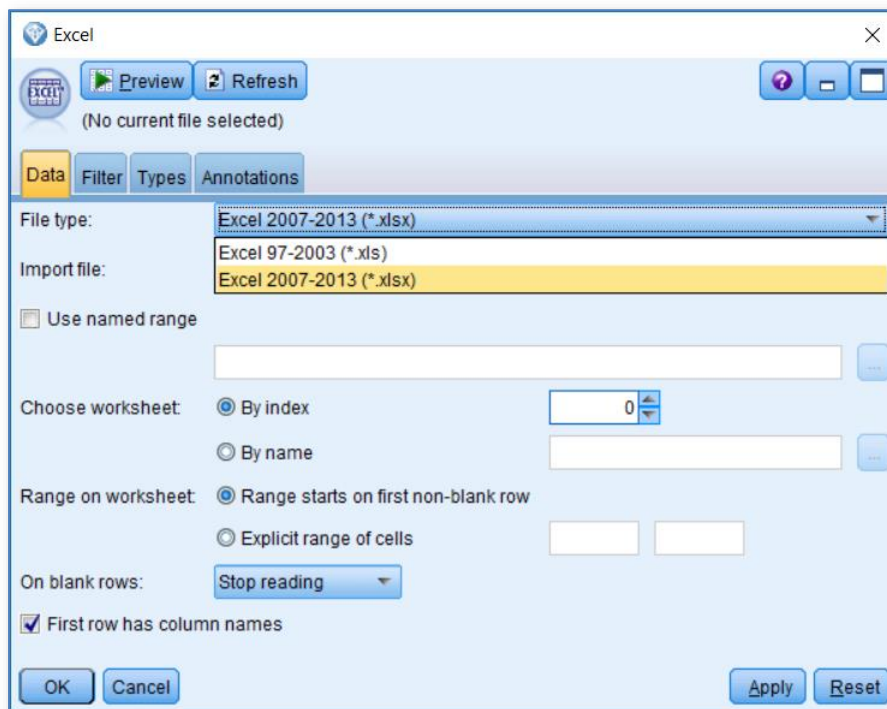


Figure 2.2 Control dialog for Excel Source node

You can see from the dialog that there is a drop-down menu to choose the type of Excel file to be read. In this case we will read an Excel workbook file. From within the dialog:

Ensure the 'Excel 2007-2013 .xlsx' option is selected from the drop down 'File Type' menu

Click the 'Import file:' ellipsis button  and browse to 'C:\SV Training\Modeler Intro\Section2'

Locate the file called 'Customer_Contacts.xlsx' and click 'Open'

Figure 2.3 shows the file selected within the Excel Source node.

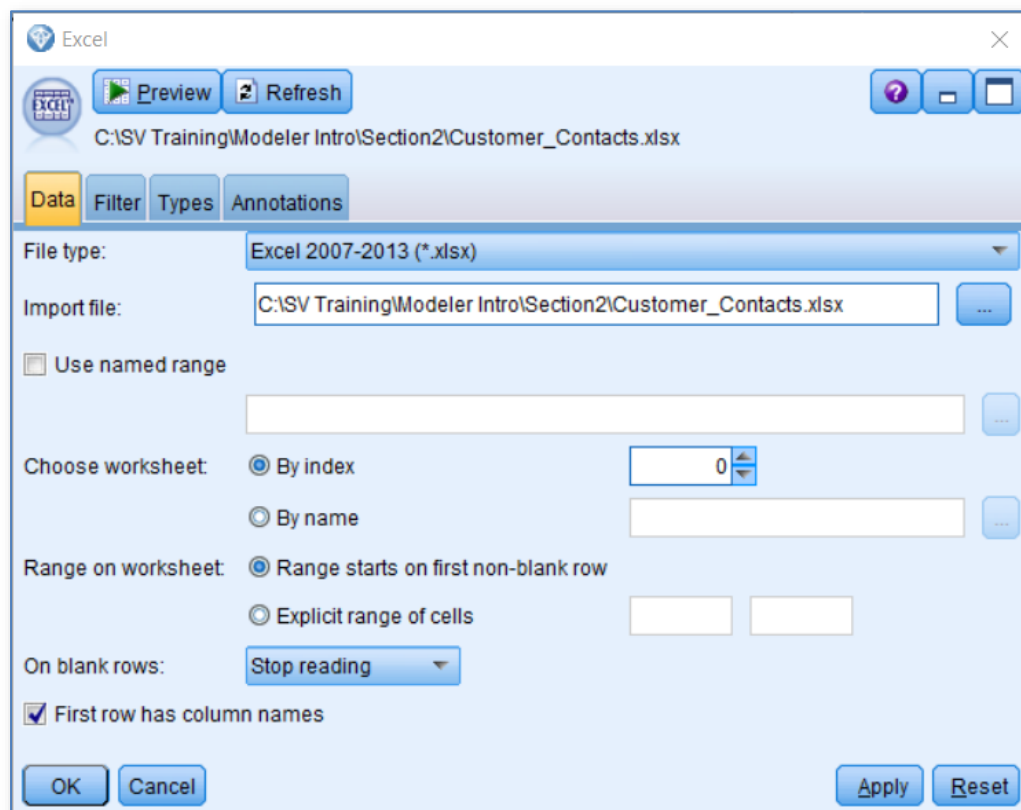


Figure 2.3 Excel workbook file selected in the Excel Source node

At this point we could probably just connect up an appropriate terminal node and read the file. However, it's a good idea to check that Modeler is reading the file correctly. There are a few ways to this. To begin with, let's examine some of the tabs across the top of the dialog. Click the tab marked:

Filter

Figure 2.4 shows the resultant Filter tab displayed.

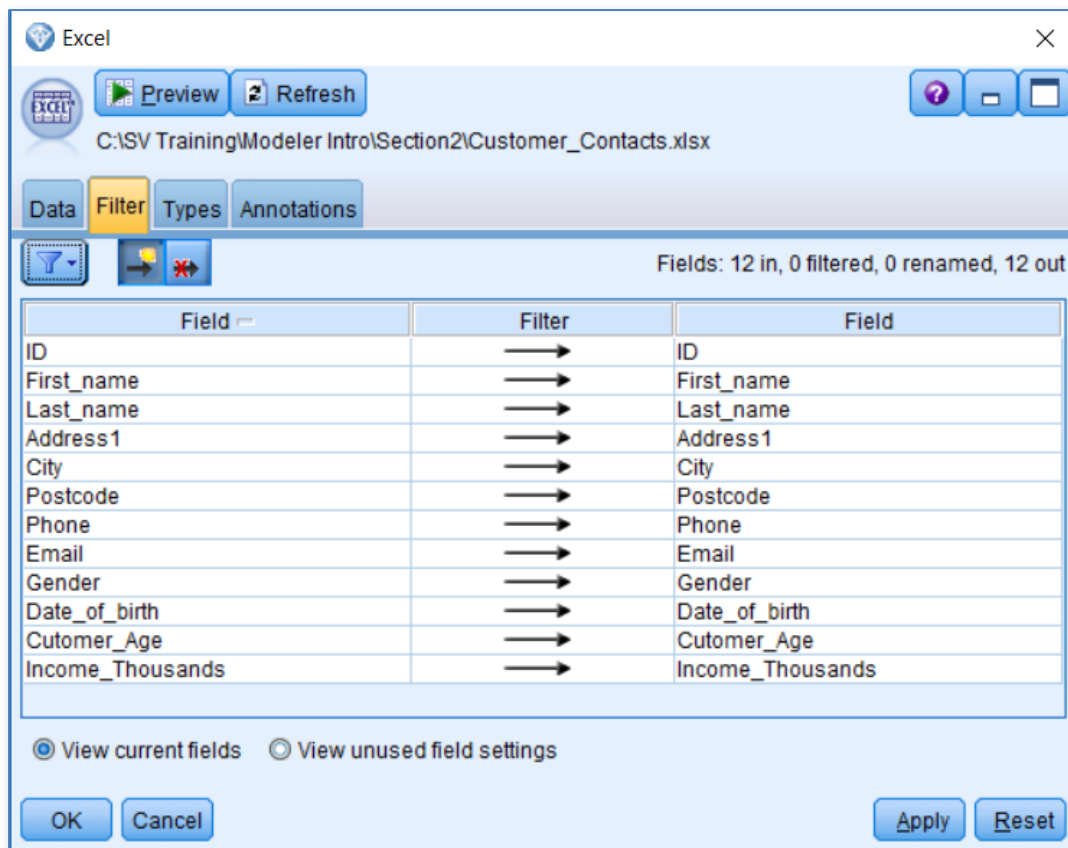


Figure 2.4 Filter tab showing the field names within the Excel node

All of the data Source nodes in Modeler contain a Filter tab. This tab offers the same basic functionality that the Filter node in Modeler does explicitly. It allows users to control which fields are to be read and to edit their names. It's also a useful way to check if a source node is correctly parsing the field names in a data file. In this case it seems to indicate that it is. To continue our examination of the Excel source node, click the tab marked:

Types

Figure 2.5 shows the Types tab within the Excel source node. Again, it's important to recognise that the basic functionality in this tab is available in Modeler as a node in its own right. In fact, the functionality in the Type node (or the Types tab) is crucial to ensuring that Modeler reads and treats data correctly as it acts as a central control hub for the overall management of data fields.

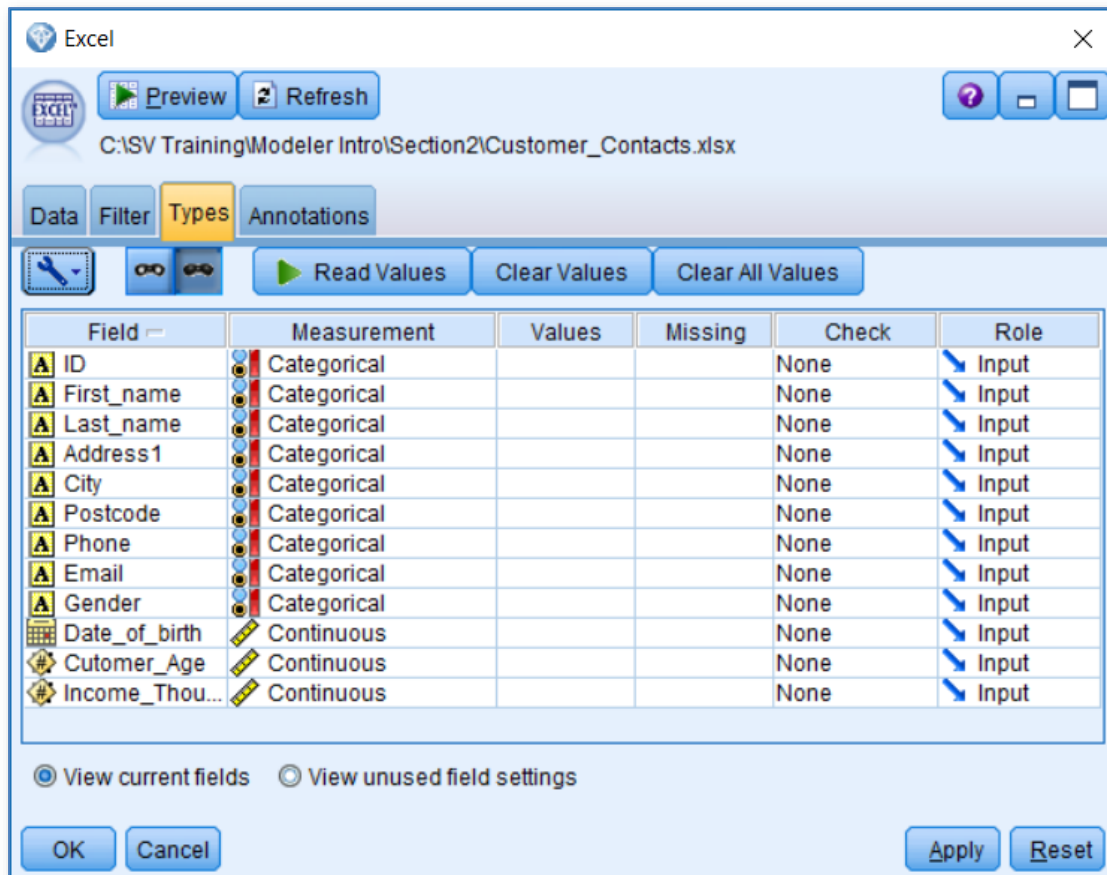


Figure 2.5 The Types Tab in the Excel Source node

Once again, there are a few important things to take note of here. We can see from the icons next to the field names that the Types tab has attempted to identify the nature of the fields in terms of the data they contain. In fact, this node/tab will categorise data in the following way:

	String data (contains non-numeric characters)
	Numerical Integer (whole numbers)
	Numerical Real (contains decimal places)
	Date
	Time Stamp

We can also see that the Types tab has classified the fields in terms of their 'measurement level' and that all the string fields are marked as 'Categorical' whereas the numeric and date fields are marked as 'Continuous'. We will return to this idea of measurement level later in the course. There are other columns in the Types tab that provide key functionality in Modeler and we will return to these later to see how they can help.

Finally, to actually see some of the data in the file we can click the 'Preview' button at the top of the dialog. Click:

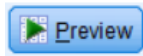


Figure 2.6 shows the data preview in the generated sample Table node.

Preview from Customer_Contacts.xlsx Node (12 fields, 10 records)

	ID	First_name	Last_name	Address1	City	Postcode	Phone	Email	Gender	Date_of_birth	Customer_Age	Income_Thousands
1	0440	Rose	Collins	28 Oak Close	ABWQ City	AB 1307	0 3866	rose_collins@www.upoazpubr.net	F	14/03/1963	52.000	53.000
2	0121	Melissa	Coleman	1 Astra Road	ABWQ City	AB 1433	0 2978	melissa_coleman@www.avlykextl.org	F	03/01/1987	28.000	37.000
3	0469	Gerald	Young	6 Ridge Way	ABWQ City	AB 1505	0 5431		M	03/04/1929	86.000	65.000
4	0669	Brandon	King	89 Alpha Avenue	ABWQ City	AB 1635	0 4553	brandon_king@www.ceqxxocw.com	M	19/06/1993	22.000	29.000
5	0608	Sara	Peterson	14 Kepler Drive	ABWQ City	AB 1682	0 2206	sara_peterson@www.zuqzsbv.com	F	26/11/1960	55.000	36.000
6	0513	Ryan	Edwards	30 North Avenue	ABWQ City	AB 1711	0 3696	ryan_edwards@www.bloazpbqm.org	M	16/10/1989	26.000	25.000
7	0697	Anne	Robinson	45 South Lane	ABWQ City	AB 2271	0 2448	a.robinson@www.qfuxqrsf.org	F	13/06/1979	36.000	21.000
8	0499	Douglas	Russell	94 Oak Avenue	ABWQ City	AB 2368	0 9785		M	12/12/1946	69.000	18.000
9	0105	Marilyn	Martinez	64 Springburn Close	ABWQ City	AB 2460	0 7763	m.martinez@www.ajyhkpj.net	F	26/11/1952	63.000	75.000
10	0168	Joan	Bryant	15 Aadrivark Way	ABWQ City	AB 2592	0 8964	joan_bryant@www.kmyhkbhe.net	F	22/01/1986	29.000	38.000

Figure 2.6 The data Preview Table for the Excel workbook file

By clicking the 'Preview' button, Modeler immediately shows us the first 10 rows of data in the file. If we wish to see more rows of data, we can change the default value associated with this function or connect an actual Table node to the Source node and look at the entire dataset. To do so, click:

OK (to close the preview window)

OK (to close the Source node dialog)

Click the Source node to select it

Click the Output Tab along the bottom of the application

Within the Output palette double-click the Table node

Figure 2.7 shows the Source and Table nodes joined together.

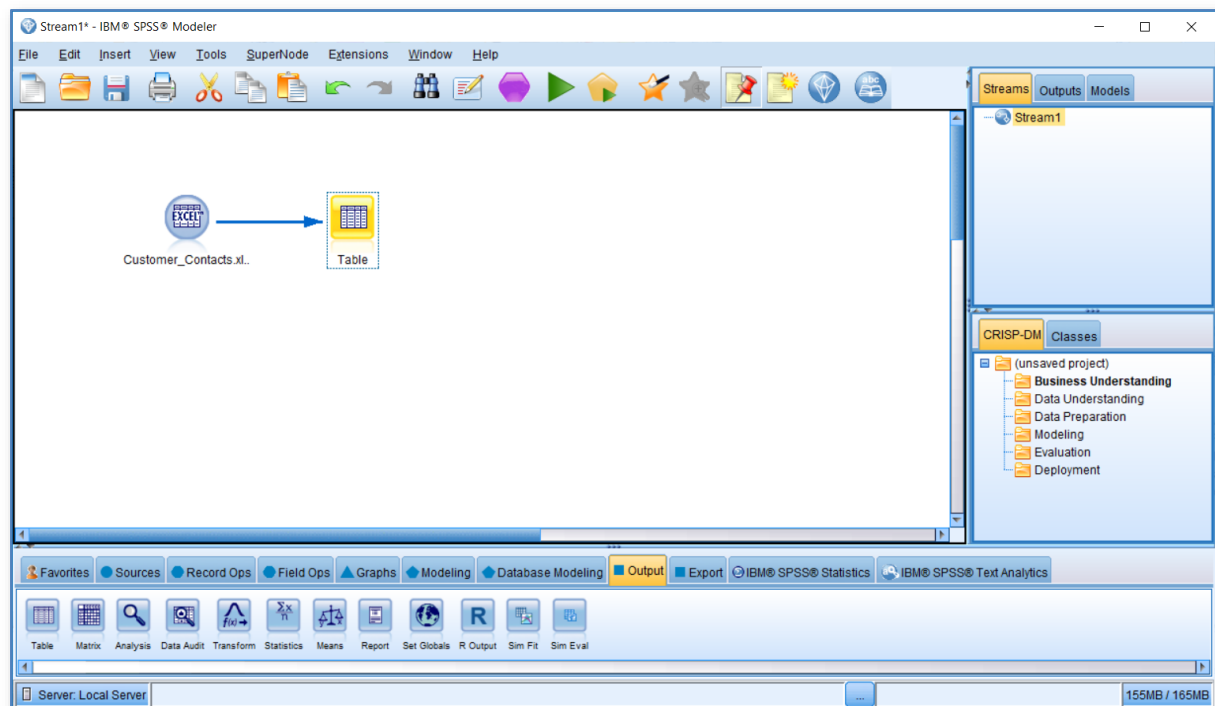


Figure 2.7 Source node reading Excel workbook file joined to a Table node

To read the file and examine the output table simply click the 'run stream' button on the toolbar. Click:



The data file is successfully read and the Table node is now populated with the contents of the Excel workbook 'Customer_Contacts.xlsx' (see figure 2.8). Notice that the title banner on the output window shows the number of fields and records read (12 fields, 850 records). The table node is normally used to simply view the data, although as we will see, there are some useful aspects of this node with regard to finding particular records or quickly generating nodes to help us select specific groups.

Table (12 fields, 850 records) #1

File Edit Generate

Table Annotations

	ID	First_name	Last_name	Address1	City	Postcode	Phone	Email
1	0440	Rose	Collins	28 Oak Close	ABWQ City	AB 1307	0 3866	rose_collins@www.upoazpubr.net
2	0121	Melissa	Coleman	1 Astra Road	ABWQ City	AB 1433	0 2978	melissa_coleman@www.avlykexl.org
3	0469	Gerald	Young	6 Ridge Way	ABWQ City	AB 1505	0 5431	
4	0669	Brandon	King	89 Alpha Avenue	ABWQ City	AB 1635	0 4553	brandon_king@www.ceqxxocw.com
5	0608	Sara	Peterson	14 Kepler Drive	ABWQ City	AB 1682	0 2206	sara_peterson@www.zuqyzsbv.com
6	0513	Ryan	Edwards	30 North Avenue	ABWQ City	AB 1711	0 3696	ryan_edwards@www.bloazpbqm.org
7	0697	Anne	Robinson	45 South Lane	ABWQ City	AB 2271	0 2448	a.robinson@www.qfuxqrsd.org
8	0499	Douglas	Russell	94 Oak Avenue	ABWQ City	AB 2368	0 9785	
9	0105	Marilyn	Martinez	64 Springburn Close	ABWQ City	AB 2460	0 7763	m.martinez@www.ajyhkpuj.net
10	0168	Joan	Bryant	15 Aadrvarik Way	ABWQ City	AB 2592	0 8964	joan_bryant@www.kmyhkbhe.net
11	0659	Beverly	Brooks	21 Kepler Way	ABWQ City	AB 2862	0 9500	
12	0558	Martin	Butler	12 Treehill Close	ABWQ City	AB 2960	0 8769	martin_butler@www.ndsiacmo.net
13	0782	Betty	Williams	25 Humber Close	ABWQ City	AB 2977	0 1011	b.williams@www.hjsiaiqe.net
14	0129	Andrea	Murphy	49 Abbey Lane	ABWQ City	AB 3049	0 5676	a.murphy@www.yyoazpudg.net
15	0304	Angela	Hernandez	93 Treehill Way	ABWQ City	AB 3206	0 9196	
16	0433	Virginia	Martinez	27 Barn Drive	ABWQ City	AB 3501	0 8652	virginia_martinez@www.xwqyzilh.org
17	0055	Anna	Lewis	86 Aadrvarik Road	ABWQ City	AB 3546	0 4355	a.lewis@www.ijoazpkmc.com
18	0275	Phillip	White	40 Astra Drive	ABWQ City	AB 3558	0 9715	p.white@www.eeyhkekv.net
19	0124	Kathy	Patterson	74 Astra Avenue	ABWQ City	AB 3561	0 8291	
20	0158	Earl	Powell	58 South Way	ABWQ City	AB 3804	0 6919	earl_powell@www.qbqyzdwr.net

OK

Figure 2.8 Table node showing contents of the Excel Workbook

Reading Text Files



One of the most common data formats that analysts encounter is the text file. This is because many third-party applications (spreadsheet programs, databases and Business Intelligence tools) can save their contents in one of many text file formats. Text files are simply files that are comprised of unformatted text so they can be viewed by various text editors. However, the files themselves come in many different shapes and sizes. Things to bear in mind when dealing with this type of file format is that some text files...

- Separate the different fields in the data by using a 'delimiter' character such as a comma, a tab or pipe ('|') character – these are called 'Delimited' files.
- Text files can have any extension, though the more common ones are '.txt', '.csv' or '.dat'.
- Instead of using delimiters, some text file formats ensure that the data values for fields always appear in the same column locations (e.g. column 1-5 for 'ID', 6-8 for 'Age'). These are often referred to as 'fixed' files (the Section 2 folder contains an example).

- Sometimes string characters appear with single or double-quotes around them (this is especially true when the file uses commas as a delimiter).
- On rare occasions, the file may contain individual records that require more than one row to store the values, so that each new record starts in every odd-numbered row.

By far the most common text file format is one that uses some sort of delimiter such as a comma. These are often referred to as 'comma separated files'. Because the data format is text, we can view the file prior to reading it in a text editor. Figure 2.9 shows our next example file displayed in MS Notepad.

```

Health_Club_Transactions_Sample.txt - Notepad
File Edit Format View Help
Member_ID|Transxn_ID|Transxn_Date|Club|Amount|Activity_Type
10530|98381640|2012-03-01 07:38:00|Edinburgh|7|Squash
1090|98386434|2012-03-01 09:03:00|Edinburgh|7|Squash
1095|98386866|2012-03-01 09:10:00|Edinburgh|7|Squash
10701|98386854|2012-03-01 09:10:00|Edinburgh|8|Gym Pass
10702|98387208|2012-03-01 09:16:00|Edinburgh|4|Gym Pass
10120|98387844|2012-03-01 09:21:00|Edinburgh|7|Squash
10703|98388018|2012-03-01 09:23:00|Edinburgh|4|Gym Pass
10703|98388180|2012-03-01 09:24:00|Edinburgh|8|Gym Pass
10117|98388666|2012-03-01 09:26:00|Edinburgh|7|Squash
10704|98389080|2012-03-01 09:28:00|Edinburgh|8|Gym Pass
10705|98389302|2012-03-01 09:30:00|Edinburgh|4|Gym Pass
1077|98389512|2012-03-01 09:32:00|Edinburgh|7|Squash
10575|98389638|2012-03-01 09:34:00|London|8|Gym Pass
10575|98389746|2012-03-01 09:35:00|London|8|Gym Pass
1091|98389704|2012-03-01 09:35:00|Edinburgh|7|Squash
10575|98390058|2012-03-01 09:36:00|London|8|Gym Pass
1096|98390130|2012-03-01 09:37:00|Edinburgh|7|Squash
10212|98390088|2012-03-01 09:37:00|Edinburgh|7|Squash
10706|98390202|2012-03-01 09:38:00|Edinburgh|8|Gym Pass
10657|98390226|2012-03-01 09:38:00|Edinburgh|7|Squash
10509|98390244|2012-03-01 09:38:00|Edinburgh|7|Squash
10706|98390268|2012-03-01 09:39:00|Edinburgh|8|Gym Pass
10118|98390304|2012-03-01 09:39:00|Edinburgh|7|Squash
10706|98390292|2012-03-01 09:39:00|Edinburgh|8|Gym Pass
10706|98390430|2012-03-01 09:40:00|Edinburgh|8|Gym Pass
10123|98390544|2012-03-01 09:40:00|Edinburgh|7|Squash
10707|98390598|2012-03-01 09:41:00|Edinburgh|8|Gym Pass
10707|98390562|2012-03-01 09:41:00|Edinburgh|8|Gym Pass
10707|98390670|2012-03-01 09:42:00|Edinburgh|8|Gym Pass
10708|98390808|2012-03-01 09:44:00|Edinburgh|4|Gym Pass
10709|98392482|2012-03-01 10:00:00|London|4|Gym Pass

```

Figure 2.9 Delimited text file displayed in a Windows text editor

As long as the file isn't too large, it's often a good idea to view it in a text editor first. That way, we can identify the delimiter character that has been used. We can see from this example that the field names are displayed across the top of the file in the first row and that each row seems to represent a unique record. We can also see that the delimiter used is the 'pipe' character '|'. This is a fairly common delimiter, especially in files that contain fields with verbatim comments as the comments themselves are unlikely to contain the delimiter character (unlike commas or tabs) so it can be safely used to separate the different fields.

To read this file, we need to add a Modeler data source node specifically designed to read delimited text files: the 'Var. File' node. From the Source node palette double click the node marked:

Var. File

The node is added to the stream containing the Excel source node. Once again, we can edit the source node by right-clicking on it and selecting:

Edit

Within the edited node, browse to 'C:\SV Training\Modeler Intro\Section2'

Locate the file called 'Health_Club_Transactions_Sample.txt' and click 'Open'

Figure 2.10 shows the dialog so far.

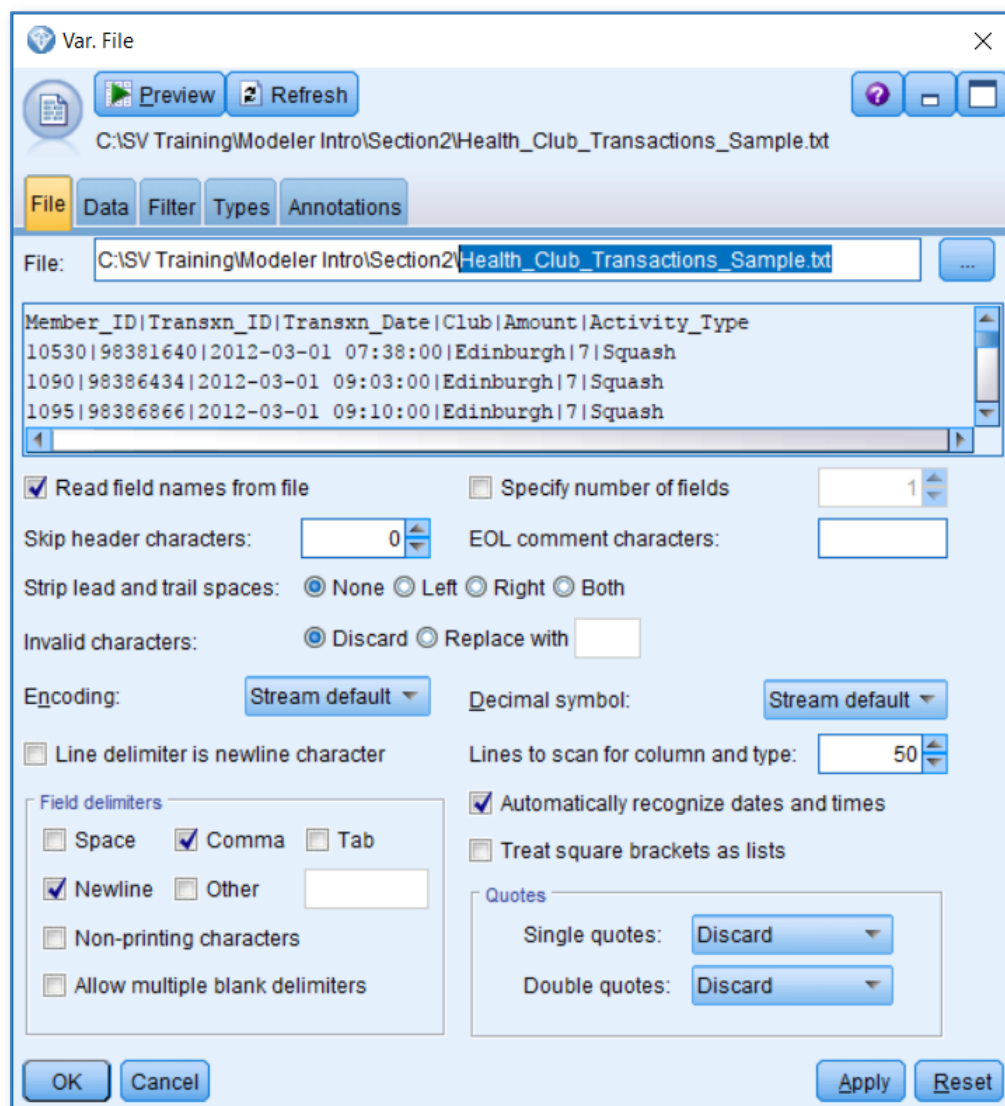


Figure 2.10 Partially complete Var. File dialog for importing a pipe-delimited file

We can immediately see that there are a lot controls and options within this dialog. At this point however, there is one option that is the most important: choosing the correct delimiter. The bottom left-hand of the dialog shows that the default delimiter, a comma character, is currently selected. However, the file preview pane at the top of the dialog clearly shows that the fields are delimited by a pipe character. We must edit this part of the dialog as shown in figure 2.11 to read the file correctly.

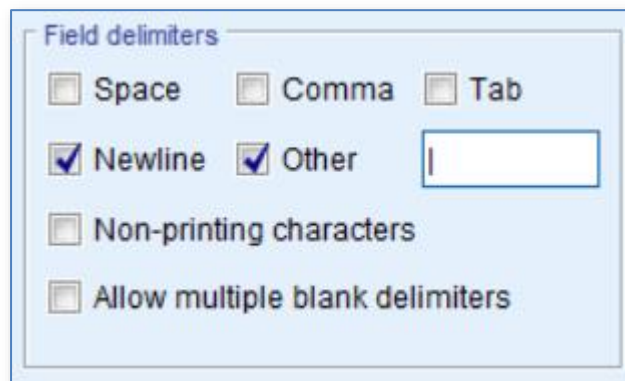


Figure 2.11 Editing the Field delimiter character so that the file is read correctly.

Figure 2.12 provides an annotated guide to some of the other interesting aspects of this dialog.

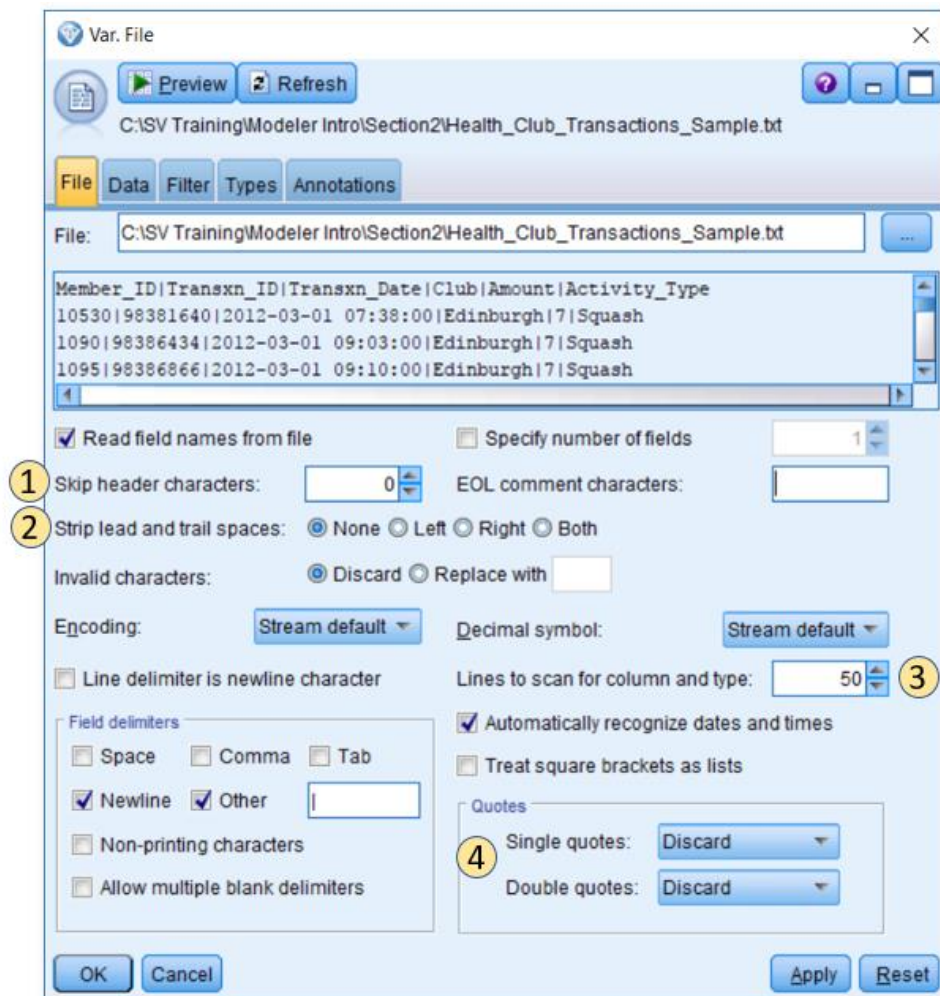


Figure 2.12 Key functionality in the Var. File node

The following numbered features refer to numbered annotations in figure 2.12.

1. **Skip header characters:** sometimes the text file contains a title or comment within the first line or so of the data. This option allows us to instruct Modeler to ignore the first n characters before it begins to read the data itself.
2. **Strip lead and trail spaces:** often values appear within the text file that contain hidden spaces either preceding or immediately following the data value itself. This can cause error messages downstream so we have the option to remove them while reading the data.
3. **Lines to scan for column and type:** because the data consists entirely of values within a text file, there is no field formatting intrinsic to the file itself. Modeler therefore attempts to identify the data type within each column based on the first 50 rows it encounters. If for example, the first 50 rows contained only whole numbers for a column, then Modeler would assume that the field was integer and would read it as such. If however, the 51st row for the same field contained a string character Modeler

would convert it to a null value. Increasing the number of scanned rows therefore allows Modeler to look deeper into the file to correctly identify the field type.

4. **Quotes:** As mentioned earlier, sometimes text data files contain quotation marks around string fields. This option allows us to discard these characters on an individual basis, to look for pairs and discard them or to read them as part of the data.

As with the Excel Source node, the Var. Node contains both a Filter and a Types tab. However, this particular source node contains an extra tab that allows us to overwrite how it reads the data columns. To explore this, click on the tab marked:

Data

Figure 2.13 shows the Data tab within the Var. File source node.

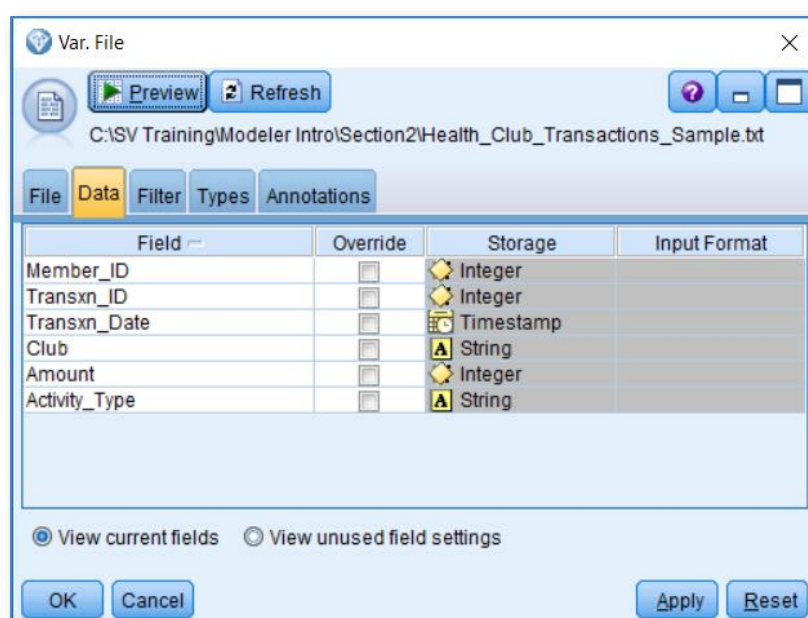


Figure 2.13 Data Tab within the Var. File source node

As we can see, the Data tab allows us to manually control how the data column is being read. By checking the 'Override' column we can choose a different storage type (e.g. integer, string etc.) from the drop-down menu. A useful aspect of this tab is the ability to control how date fields are read. Although by default the Var. File node attempts to automatically detect and read date or datetime fields, there are nevertheless occasions when it fails to do so correctly and instead reads the column as a string. Not only does the Data tab allow the user to override this but it also allows us to choose the exact date format from a list within the 'Input Format' column. Figure 2.14 illustrates this.

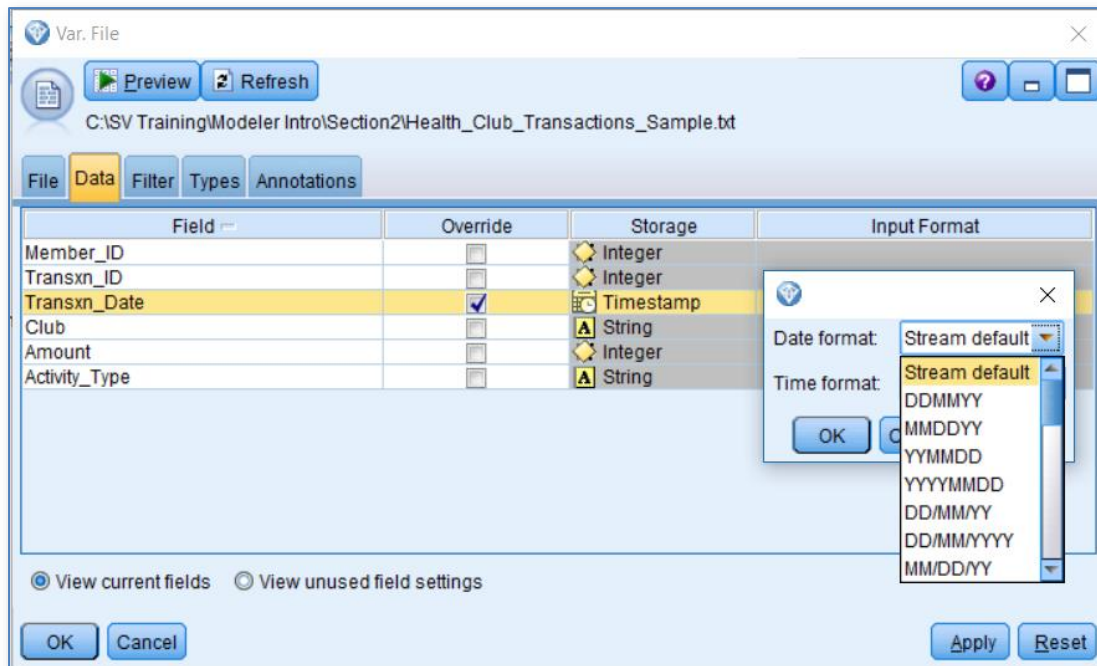


Figure 2.14 The Override and Input format functions within the Data tab

In this case there is no need to override any of the settings within the data tab. In fact, it appears that having specified the correct delimiter character, we can now read the data file. To show the data in an output table:

Right-click on the existing Output Table node in the stream and choose 'Copy Node' from the drop-down menu (or click Ctrl+C)

Right-click on the stream canvas and choose 'Paste' (or click Ctrl+V)

Manually connect it to the Var. File source node

We can now execute this part of the stream by either:

Right-clicking on the newly-pasted terminal node and clicking 'Run' or clicking the 'Run Selection' button 

Figure 2.15 illustrates this and figure 2.16 shows the resulting output.

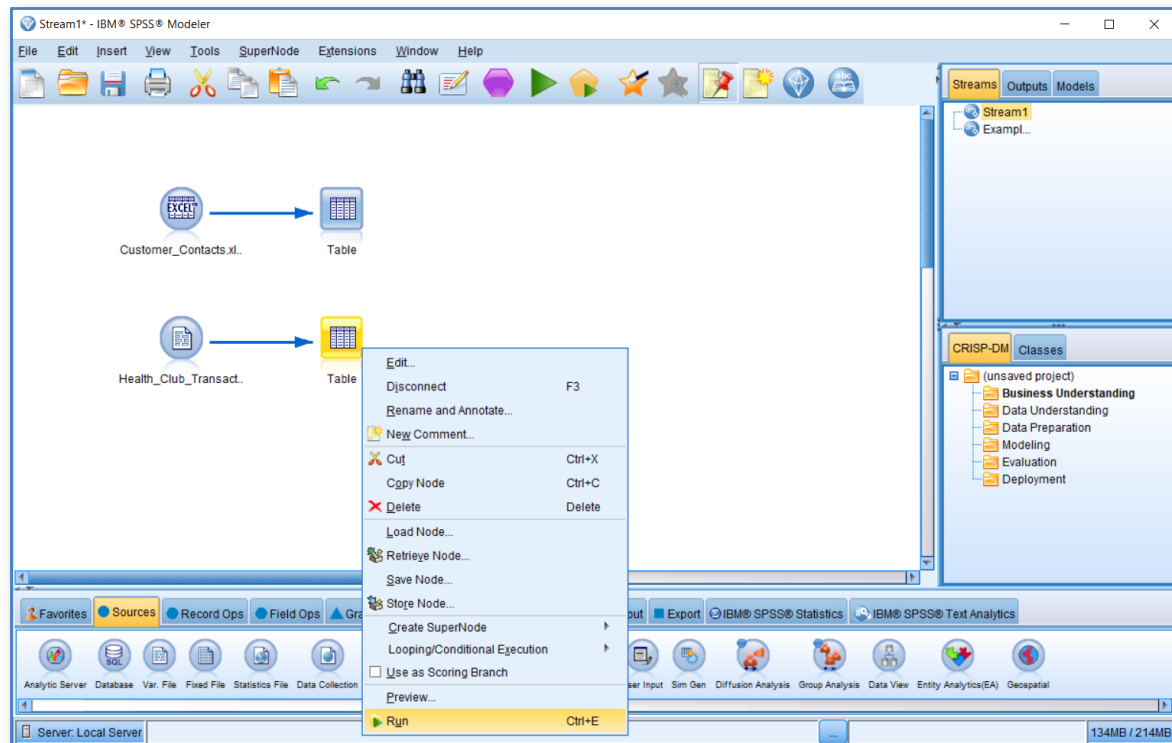


Figure 2.15 Running the new stream branch from the pasted Output Table node

Table (6 fields, 19,827 records)

	Member_ID	Transxn_ID	Transxn_Date	Club	Amount	Activity_Type
1	10530	98381640	01/03/2012 07:38:00	Edinburgh	7	Squash
2	1090	98386434	01/03/2012 09:03:00	Edinburgh	7	Squash
3	1095	98386866	01/03/2012 09:10:00	Edinburgh	7	Squash
4	10701	98386854	01/03/2012 09:10:00	Edinburgh	8	Gym Pass
5	10702	98387208	01/03/2012 09:16:00	Edinburgh	4	Gym Pass
6	10120	98387844	01/03/2012 09:21:00	Edinburgh	7	Squash
7	10703	98388018	01/03/2012 09:23:00	Edinburgh	4	Gym Pass
8	10703	98388180	01/03/2012 09:24:00	Edinburgh	8	Gym Pass
9	10117	98388666	01/03/2012 09:26:00	Edinburgh	7	Squash
10	10704	98389080	01/03/2012 09:28:00	Edinburgh	8	Gym Pass
11	10705	98389302	01/03/2012 09:30:00	Edinburgh	4	Gym Pass
12	1077	98389512	01/03/2012 09:32:00	Edinburgh	7	Squash
13	10575	98389638	01/03/2012 09:34:00	London	8	Gym Pass
14	10575	98389746	01/03/2012 09:35:00	London	8	Gym Pass
15	1091	98389704	01/03/2012 09:35:00	Edinburgh	7	Squash
16	10575	98390058	01/03/2012 09:36:00	London	8	Gym Pass
17	1096	98390130	01/03/2012 09:37:00	Edinburgh	7	Squash
18	10212	98390088	01/03/2012 09:37:00	Edinburgh	7	Squash
19	10706	98390202	01/03/2012 09:38:00	Edinburgh	8	Gym Pass
20	10657	98390226	01/03/2012 09:38:00	Edinburgh	7	Squash
21	10509	98390244	01/03/2012 09:38:00	Edinburgh	7	Squash

Figure 2.16 Contents of the file Health_Club_Transactions_Sample.txt shown in a Table node

The table node shows that although the file itself only contains 6 fields it is comprised of 19,827 records.

The Generate Menu

While the table output is still displayed, we can take the opportunity to point out one of the most useful features within Modeler: the Generate menu. This menu enhances the user's productivity greatly by automatically generating different nodes based on selections within the output. Figure 2.17 shows the options available for the Table Output node under the Generate menu.

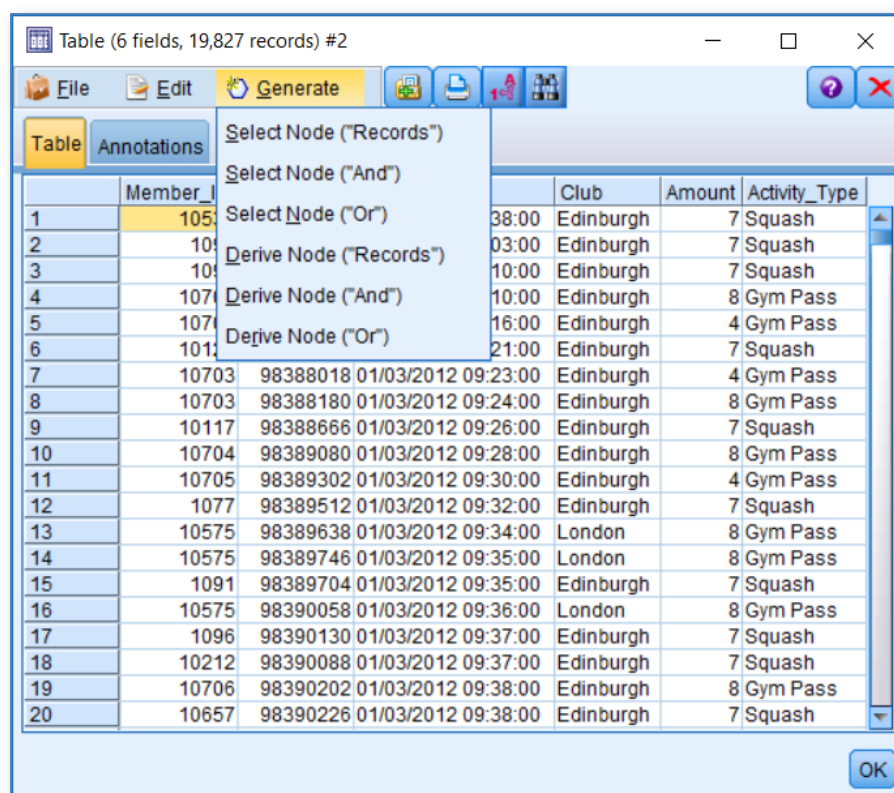


Figure 2.17 The Generate Menu in the Table output node

In this instance, the Generate menu will create two different types of nodes: Select and Derive. The Select node allows us to filter records based on specific criteria, whereas the Derive node creates new fields according to various criteria (we will take a deeper look at the Derive node later in the course). In both of these cases, we could manually retrieve these nodes from their respective palettes and attach them to the stream before finally editing them to run the procedure in a particular way. However, by automatically generating these nodes, the Generate menu can save users a lot of time and effort especially when carrying out common tasks such as filtering data, re-coding categories or creating flag fields.

The Select Node



We can illustrate how useful this function is by using the Generate menu to create a Select node. Let's assume that the user is only interested in club members who play squash in London. To select only these people:

Use ctrl-click to select any two cells in the table that correspond to 'London' and 'Squash'

From the Generate menu choose:

Select Node ("And")

Figure 2.18 illustrates this process.

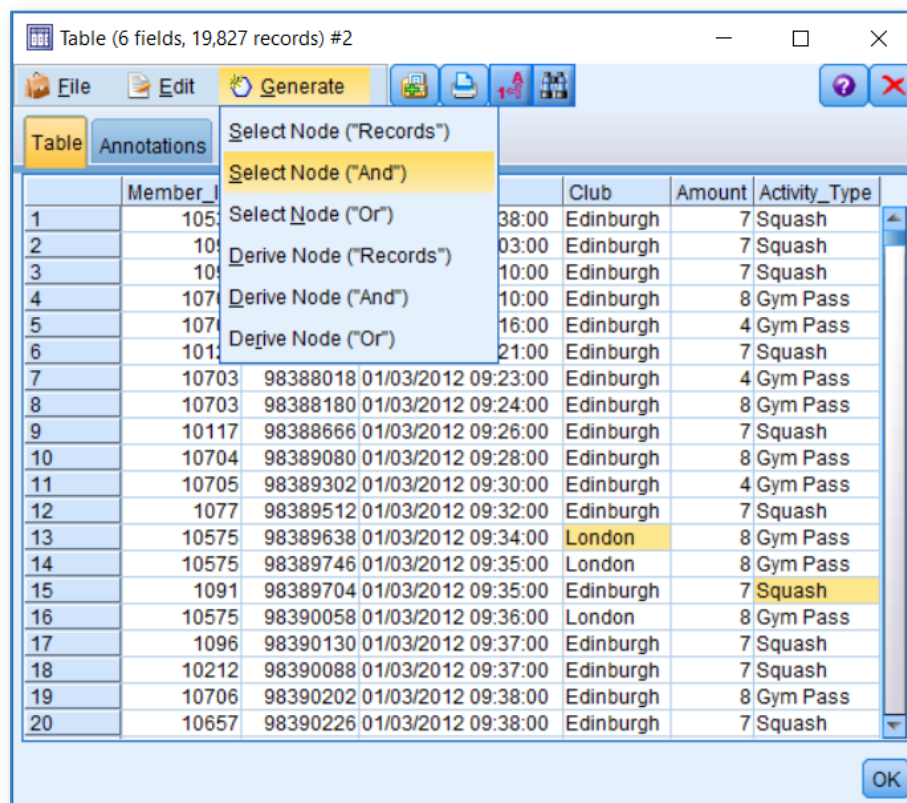


Figure 2.18 Using the Generate menu to create a Select node that selects only London members who play squash

A Select node is now automatically generated and placed on the stream canvas.

Connect the newly-generated Select node to the Health_Club_Transactions_Sample.txt source node.

Connect a new Table Output node (you can copy and paste again) downstream of the Select node

Figure 2.19 shows the completed Select node stream branch.

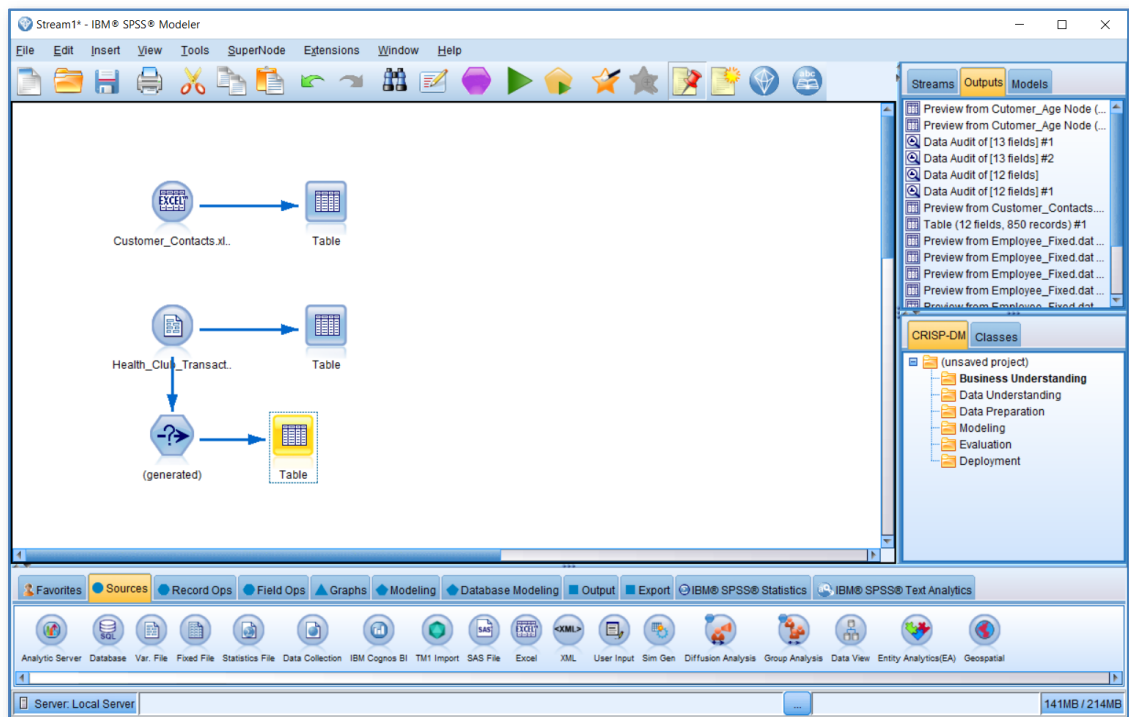


Figure 2.19 Generated Select node connected to the original source node with Table Output node attached

Before we run the actual node, let's take a look inside to see what the Generate menu has done.

Right-click on the generated Select Node and choose 'Edit'

Figure 2.20 shows the contents of the generated node.

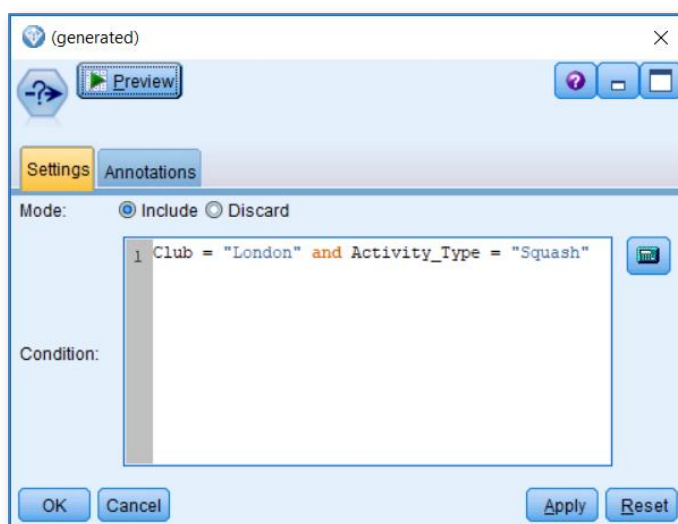


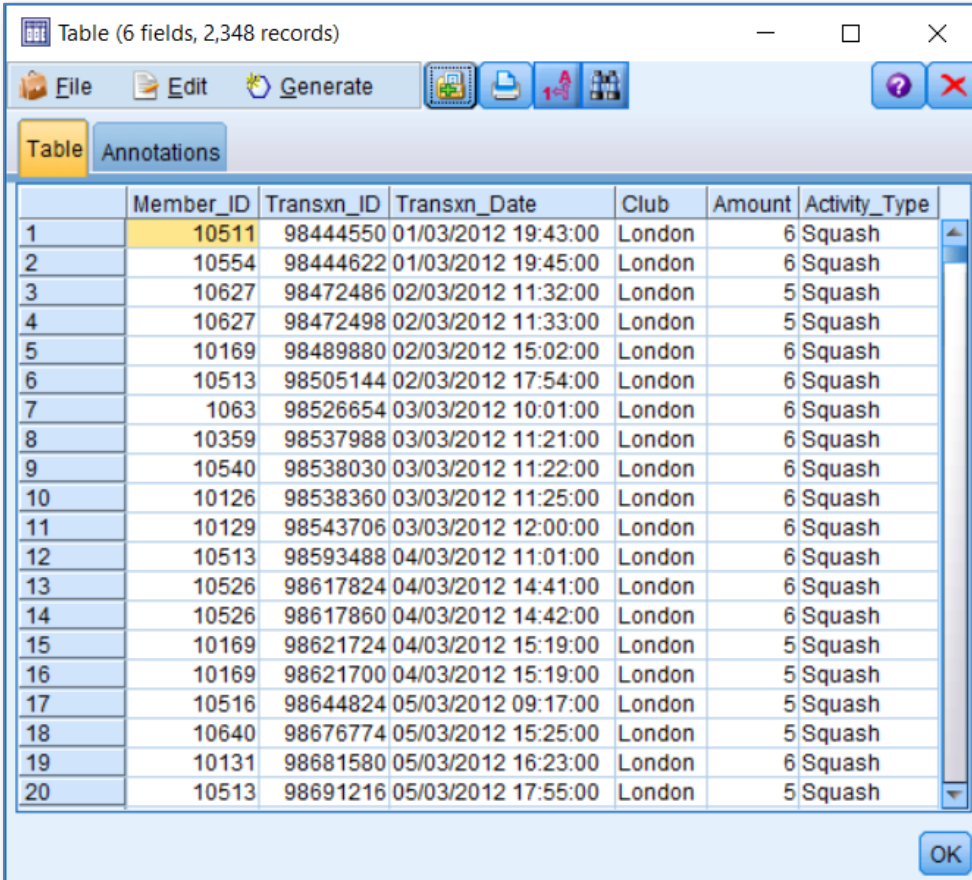
Figure 2.20 Contents of the Generated Select node

The contents of the Select node show that we can switch between 'Include' and 'Discard' mode. By default, the generated node will include those transactions from customers whose health club is in London and where the purchased activity type is squash. To run the node, close the dialog by clicking:

OK

Select and run the associated Table Output node

Figure 2.21 shows that the node selects 2,348 transactions that meet the criteria.



	Member_ID	Transxn_ID	Transxn_Date	Club	Amount	Activity_Type
1	10511	98444550	01/03/2012 19:43:00	London	6	Squash
2	10554	98444622	01/03/2012 19:45:00	London	6	Squash
3	10627	98472486	02/03/2012 11:32:00	London	5	Squash
4	10627	98472498	02/03/2012 11:33:00	London	5	Squash
5	10169	98489880	02/03/2012 15:02:00	London	6	Squash
6	10513	98505144	02/03/2012 17:54:00	London	6	Squash
7	1063	98526654	03/03/2012 10:01:00	London	6	Squash
8	10359	98537988	03/03/2012 11:21:00	London	6	Squash
9	10540	98538030	03/03/2012 11:22:00	London	6	Squash
10	10126	98538360	03/03/2012 11:25:00	London	6	Squash
11	10129	98543706	03/03/2012 12:00:00	London	6	Squash
12	10513	98593488	04/03/2012 11:01:00	London	6	Squash
13	10526	98617824	04/03/2012 14:41:00	London	6	Squash
14	10526	98617860	04/03/2012 14:42:00	London	6	Squash
15	10169	98621724	04/03/2012 15:19:00	London	5	Squash
16	10169	98621700	04/03/2012 15:19:00	London	5	Squash
17	10516	98644824	05/03/2012 09:17:00	London	5	Squash
18	10640	98676774	05/03/2012 15:25:00	London	5	Squash
19	10131	98681580	05/03/2012 16:23:00	London	6	Squash
20	10513	98691216	05/03/2012 17:55:00	London	5	Squash

Figure 2.21 Table Output node showing selected transaction records

We could also use the Generate menu to select records based on 'Or' statements (by clicking "Select node 'Or'") or simply to select individual records that we have clicked on within the Table output ("Select Node 'Records'"). This is a simple but powerful approach to selecting data interactively and as we will see later, the Generate menu has many other uses that make data manipulation tasks easier.

The Database Node



In this final example of reading data sources, we will look at the Database source node. Modeler supports many proprietary database formats including Microsoft SQL Server, Oracle, DB2, Teradata, Greenplum, Amazon Redshift and SAP Hanna. Using the Database node, we can open a pre-configured connection to a database. Normally the ability to define and open this connection requires administrator rights. So often the link to the organisation's data warehouse is configured by the local IT staff. In this example, we will use a MS Access database to illustrate how to use the Database node. Further information providing an overview of how database connections (ODBC links) are created is shown as an appendix at the end of this section. From the Data Source palette:

Find the Database node and place it on the existing stream canvas

Right-click on the Database source node and choose 'Edit'

Figure 2.22 shows the Database source node added to the stream and the control dialog open.

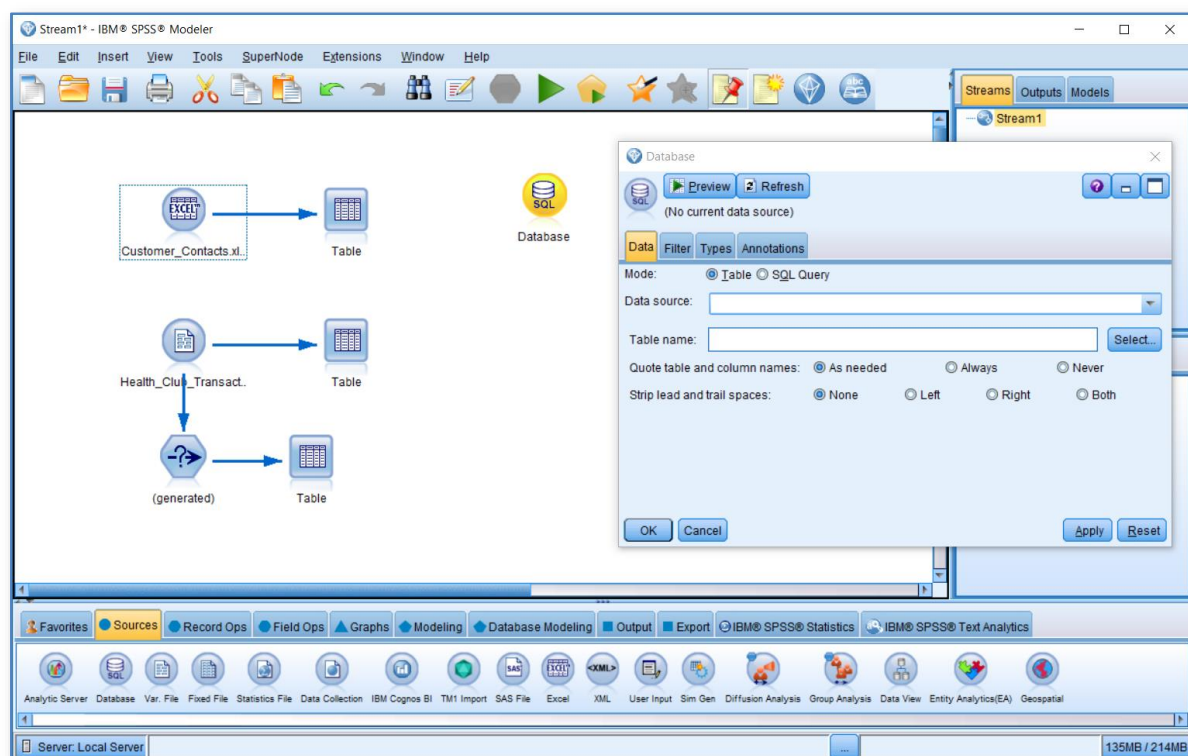


Figure 2.22 The Database Source Node and associate control dialog

We can see from the two radio buttons at the top of the dialog that we can choose an existing connection whilst in 'Table' mode or switch to 'SQL Query' mode and use a SQL instruction to select and retrieve data. Sticking with the default setting here, figure 2.23 shows that clicking on the Data Source drop-down menu allows us to 'Add a new database connection'.

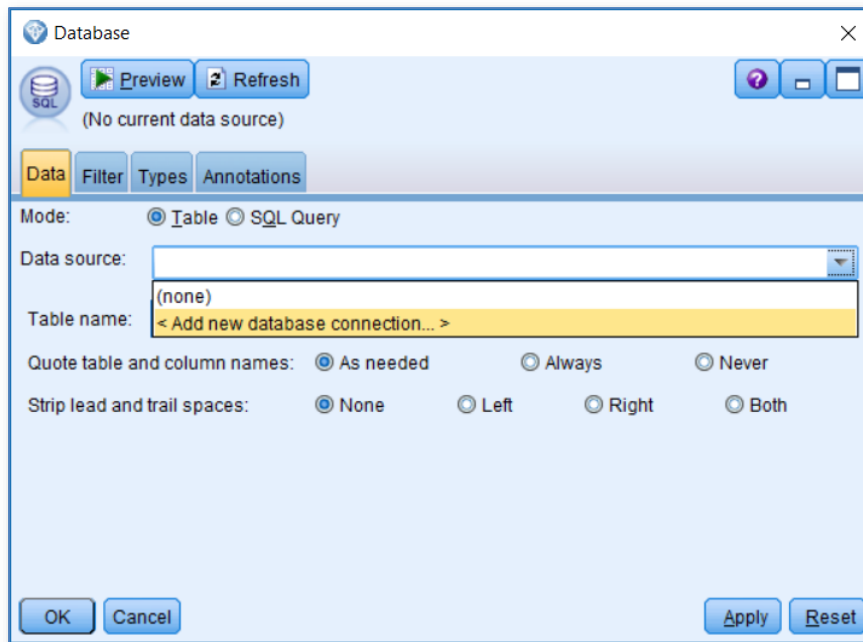


Figure 2.23 Adding a new Database connection in the Database source node

By choosing this option, a second sub-dialog is opened.

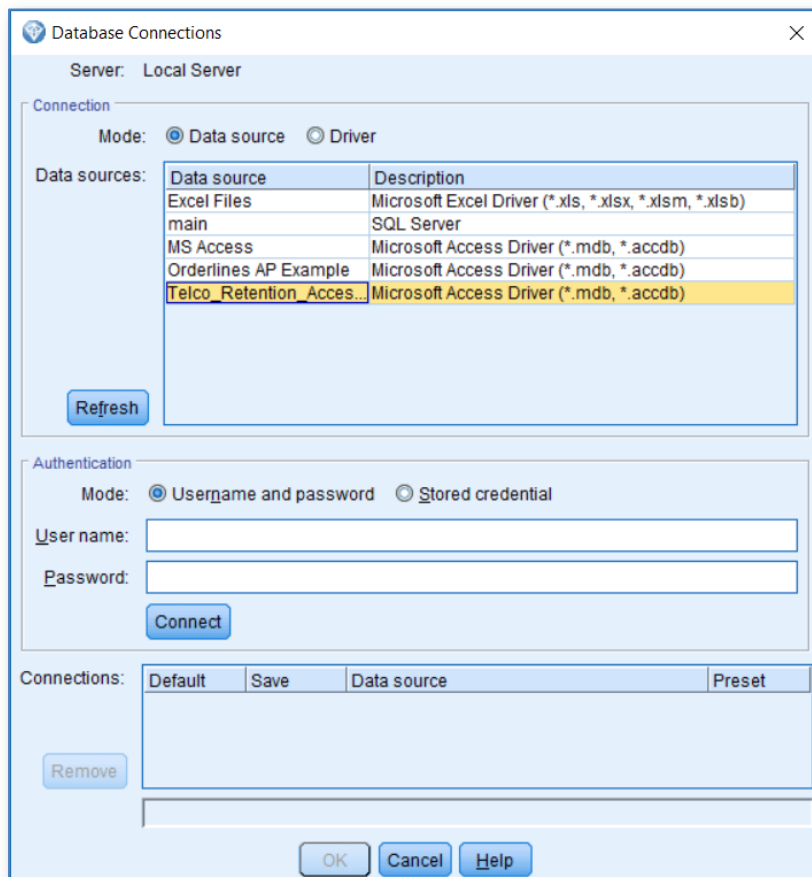


Figure 2.24 Database Connections sub-dialog

We can see from the screenshot, that the PC in the example already has some pre-configured connections to databases. In this case, we will click on the database marked 'Telco_Retention_Access_DB' and then click:

Connect

OK

Having established a connection to the database, we are returned to the main control dialog for the Database source node. Here we can select an individual table to read from the database (most databases are normally comprised of a number of related tables). To do so, click the button marked:

Select

Another sub-dialog is opened. This time it shows the available tables for us to read although in this database there is only one present. Figure 2.25 shows this.

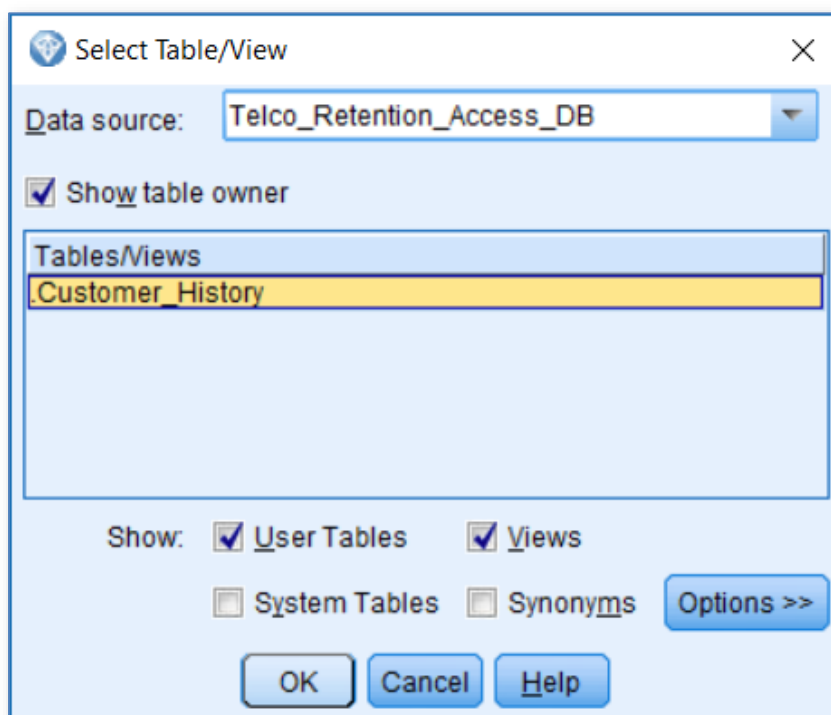


Figure 2.25 Choosing a Database table to read

To return to the main dialog, click:

OK

Figure 2.26 shows the completed dialog.

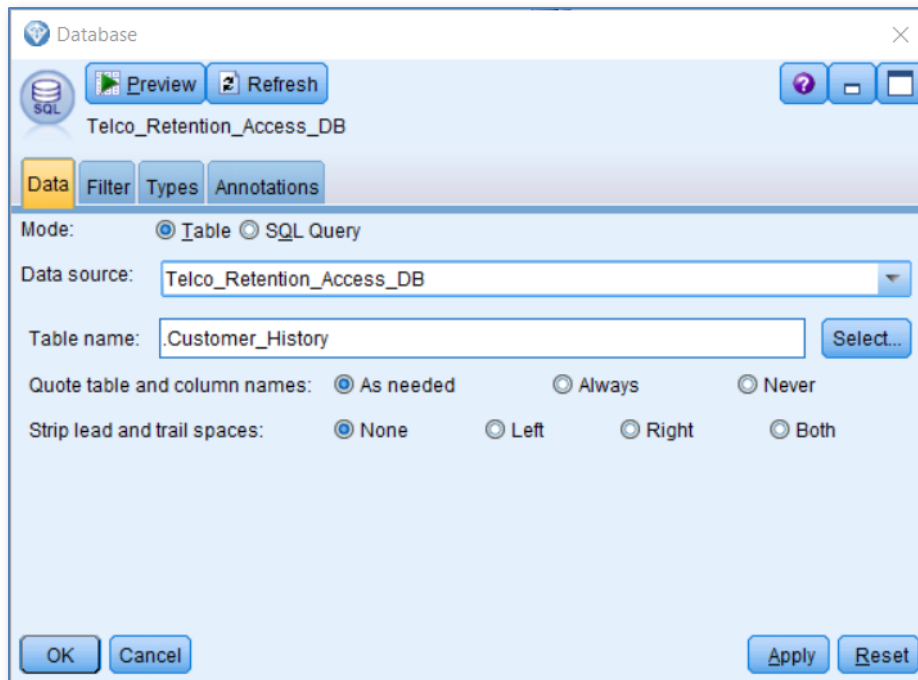


Figure 2.26 Completed dialog for the Database source node

To finish the process, click:

OK

Attach a Table output node to the Database source node and run the stream branch to view the data

Figure 2.27 shows the Access data displayed in a Table output node.

	customerID	gender	Pensioner	Partner	Dependents	Tenure	Landline	Multiline	Broadba...	Internet_Protection	Backup	Insurance	Premium Support	Super_Sports	Box_Office_Movies	Auto_Renew	Et
1	0	Female	0.000	Yes	No	1.000	No	Not Applicable	DSL	No	Yes	No	No	No	No	Month-to-month	Ye
2	1	Male	0.000	No	No	34.000	Yes	No	DSL	Yes	No	Yes	No	No	No	One year	Nc
3	2	Male	0.000	No	No	2.000	Yes	No	DSL	Yes	Yes	No	No	No	No	Month-to-month	Ye
4	3	Male	0.000	No	No	45.000	No	Not Applicable	DSL	Yes	No	Yes	Yes	No	No	One year	Nc
5	4	Female	0.000	No	No	2.000	Yes	No	Fiber optic	No	No	No	No	No	No	Month-to-month	Ye
6	5	Female	0.000	No	No	8.000	Yes	Yes	Fiber optic	No	No	Yes	No	Yes	Yes	Month-to-month	Ye
7	6	Male	0.000	Yes	Yes	22.000	Yes	Yes	Fiber optic	No	Yes	No	No	Yes	No	Month-to-month	Ye
8	7	Female	0.000	No	No	10.000	No	Not Applicable	DSL	Yes	No	No	No	No	No	Month-to-month	Nc
9	8	Female	0.000	Yes	No	28.000	Yes	Yes	Fiber optic	No	No	Yes	Yes	Yes	Yes	Month-to-month	Ye
10	9	Male	0.000	Yes	Yes	62.000	Yes	No	DSL	Yes	Yes	No	No	No	No	One year	Nc
11	10	Male	0.000	Yes	Yes	13.000	Yes	No	DSL	Yes	No	No	No	No	No	Month-to-month	Ye
12	11	Male	0.000	No	No	16.000	Yes	No	No	No internet service	No int...	No intern...	No internet servi...	No internet s...	No internet service	Two year	Nc
13	12	Male	0.000	Yes	No	58.000	Yes	Yes	Fiber optic	No	No	Yes	No	Yes	Yes	One year	Nc
14	13	Male	0.000	No	No	49.000	Yes	Yes	Fiber optic	No	Yes	Yes	No	Yes	Yes	Month-to-month	Ye
15	14	Male	0.000	No	No	25.000	Yes	No	Fiber optic	Yes	No	Yes	Yes	Yes	Yes	Month-to-month	Ye
16	15	Female	0.000	Yes	Yes	69.000	Yes	Yes	Fiber optic	Yes	Yes	Yes	Yes	Yes	Yes	Two year	Nc
17	16	Female	0.000	No	No	52.000	Yes	No	No	No internet service	No int...	No intern...	No internet servi...	No internet s...	No internet service	One year	Nc
18	17	Male	0.000	No	Yes	71.000	Yes	Yes	Fiber optic	Yes	No	Yes	No	Yes	Yes	Two year	Nc
19	18	Female	0.000	Yes	Yes	10.000	Yes	No	DSL	No	No	Yes	Yes	No	No	Month-to-month	Nc
20	19	Female	0.000	No	No	21.000	Yes	No	Fiber optic	No	Yes	Yes	No	No	Yes	Month-to-month	Ye
21	20	Male	1.000	No	No	1.000	No	Not Applicable	DSL	No	No	Yes	No	No	Yes	Month-to-month	Ye
22	21	Male	0.000	Yes	No	12.000	Yes	No	No	No internet service	No int...	No intern...	No internet servi...	No internet s...	No internet service	One year	Nc
23	22	Male	0.000	No	No	1.000	Yes	No	No	No internet service	No int...	No intern...	No internet servi...	No internet s...	No internet service	Month-to-month	Nc
24	23	Female	0.000	Yes	No	58.000	Yes	Yes	DSL	No	Yes	No	Yes	No	No	Two year	Ye
25	24	Male	0.000	Yes	Yes	49.000	Yes	No	DSL	Yes	Yes	No	Yes	No	No	Month-to-month	Nc
26	25	Female	0.000	No	No	30.000	Yes	No	DSL	Yes	Yes	No	No	No	No	Month-to-month	Ye
27	26	Male	0.000	Yes	Yes	47.000	Yes	Yes	Fiber optic	No	Yes	No	No	Yes	Yes	Month-to-month	Ye

Figure 2.27 Data from the Access database 'Telco_Retention_Access_DB'

Save the final stream as 'Data Sources Example.str'.

Appendix: Defining an ODBC link to a database

The acronym ODBC stands for 'Open Database Connectivity'. This refers to an open standard interface for accessing different kinds of databases. In Windows environments, the ODBC Manager is usually accessed from the Control Panel under the heading 'Administrative Tools'. The following section provides an overview of the process for defining an ODBC connection.

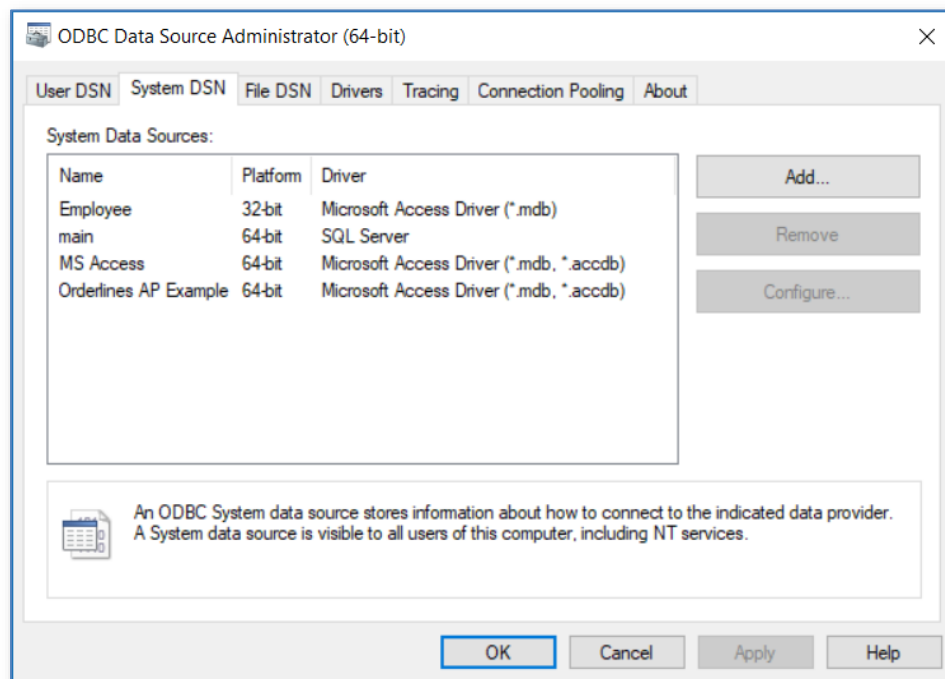


Figure A1 – The main interface for the ODBC Administrator (64 bit)

Figure A1 shows the ODBC Administrator with a few existing database connections already defined within the System DSN tab (DSN simply stands for 'Data Source Name'). To add a new connection, the user clicks the 'Add' button. As figure A2 shows, a sub-dialog is generated, and list of installed database drivers is shown. The IBM® SPSS® Data Access Pack includes a set of drivers that users may download from the IBM site (if not already installed with the Modeler software).

In this example, we are attempting to connect to an Access file, so the Microsoft Access Driver (*.mdb *.acldb) is chosen.

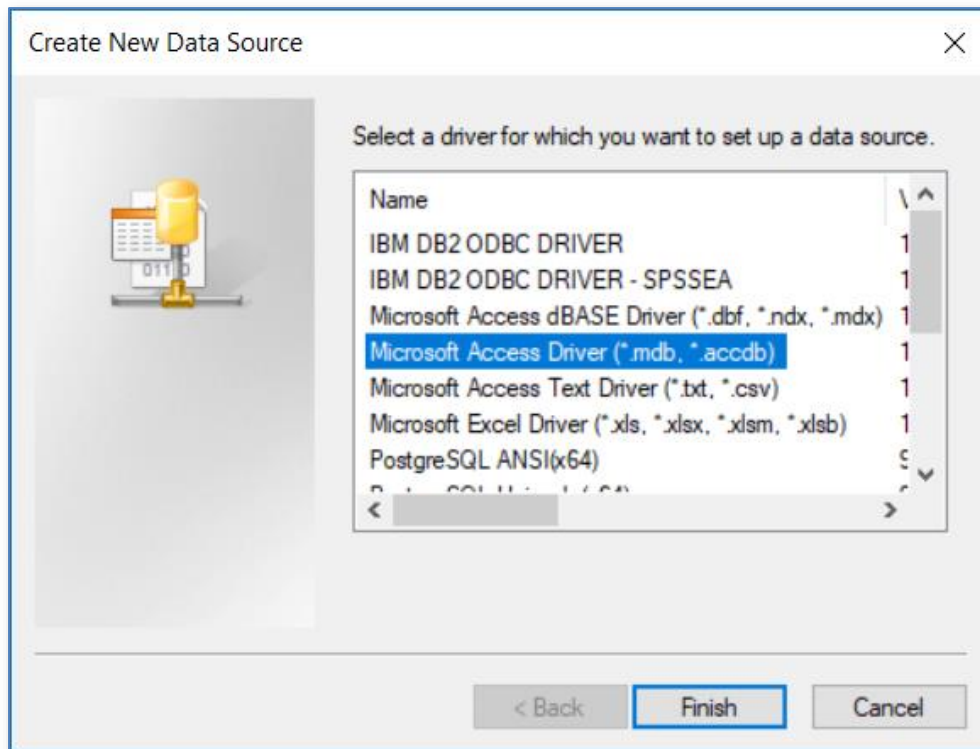


Figure A2 – Database Access drivers for ODBC connections

Clicking 'Finish' takes us to the Setup dialog for Microsoft Access files. Here we can browse to the location of the file in question.

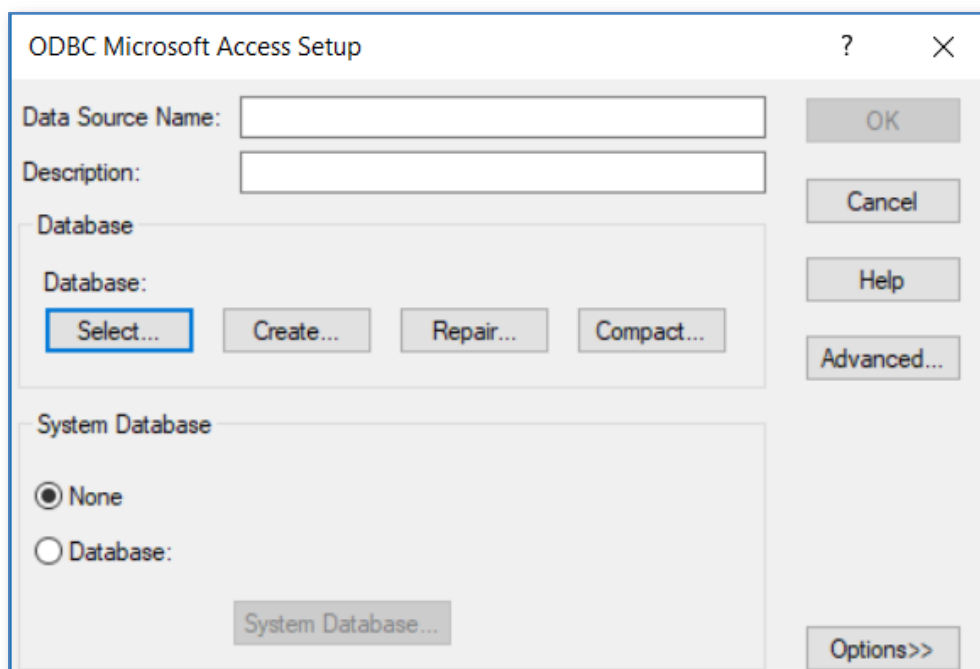


Figure A3 – Setup dialog for Microsoft Access files

Figure A4 shows the Select Database dialog that achieves this.

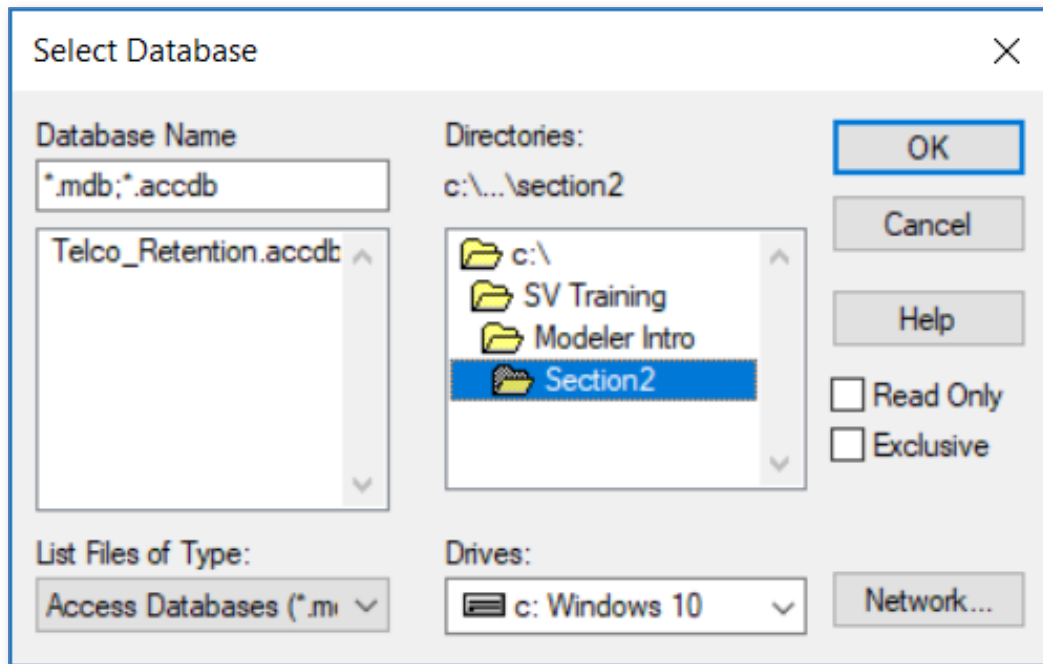


Figure A4 - Select Database dialog

Having selected the database, we can provide a name and description for the DSN.

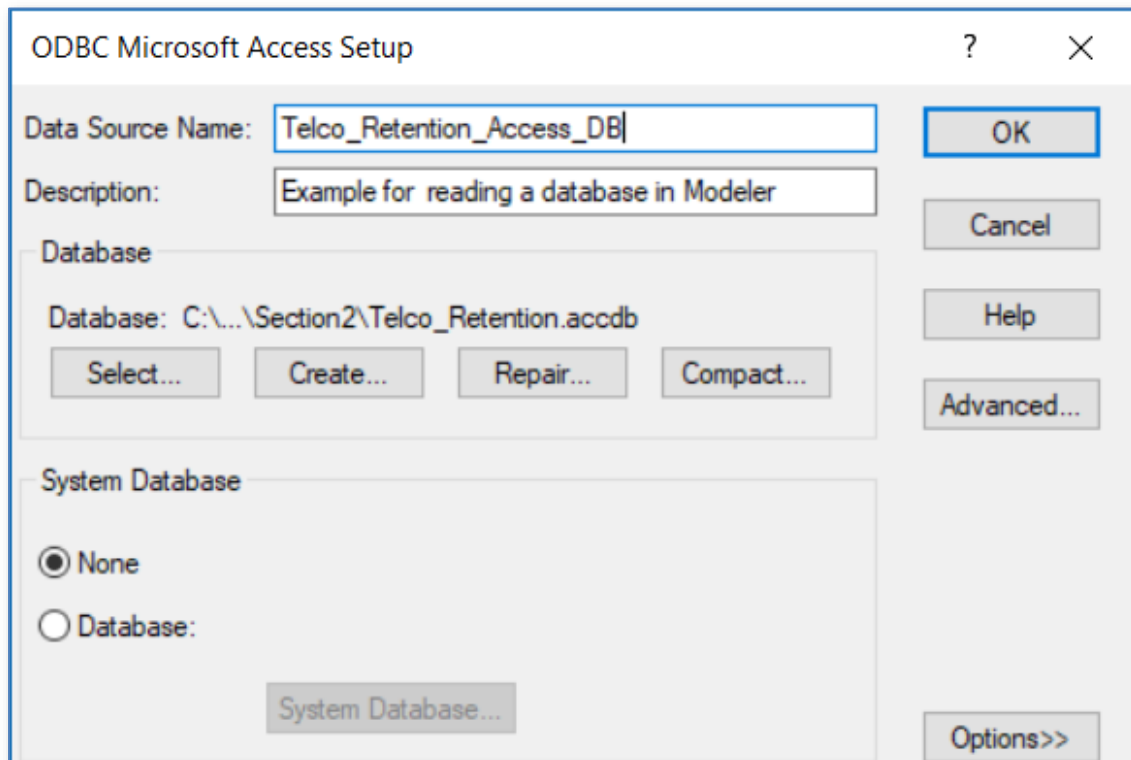


Figure A5 - Providing a name and description for the DSN

Finally, as figure A6 shows, we are returned to the main ODBC Administrator dialog and the connection is defined.

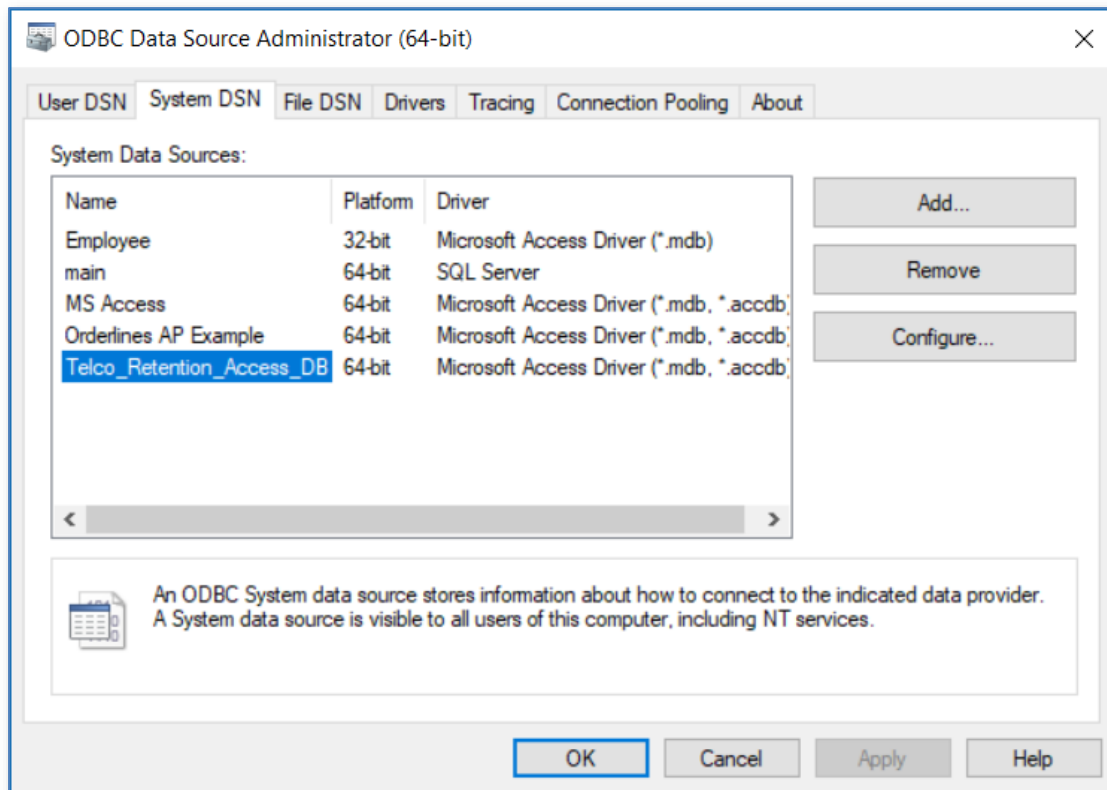


Figure A6 – The main ODBC Administrator dialog with the new connection displayed

Now, when the Database Source node is edited within the Modeler tool, the link to the Microsoft Access file 'Telco_Retention_Access_DB' will be displayed and we will be able to connect to it.

Section 3: Data Understanding



- The Filter Node
- The Type Node
- Levels of Measurement
- The Check Column
- The Values Column
- Missing Values
- The Variable Role
- The Data Audit Node

In classical statistics, it is regarded as good practice to thoroughly explore the data before any deeper analysis is performed. In predictive analytics however, it is essential. This is because not only must the analyst gain an understanding as to what the data fields actually record, as well as the general variability within the data, they very often need to deal with issues of data quality. Errors and inaccuracies in data can occur due to any number reasons: occasional typos when entering values; missing values due to a reliance on voluntary information; fraudulent responses; insufficient staff training leading to misunderstandings and general inconsistencies. For these reasons, the Data Understanding stage of the CRISP-DM process is a crucial step as it is often the first opportunity the analyst has to assess how feasible the project's aims are having seen the data. Modeler offers us many methods to address these kinds of issues, and in this section, we will look at three key nodes to help us to achieve that.

The Filter Node



As we saw in the previous section, the filter tab can be accessed within Modeler's data source nodes. However, the ability to edit field names and to include or exclude them from downstream procedures also appears as a node in its own right. To demonstrate this, from the Section 3 folder:

Open the Modeler stream file 'Data Understanding.str'

Right-click and edit the Filter node

Figure 3.1 shows the resultant control dialog.

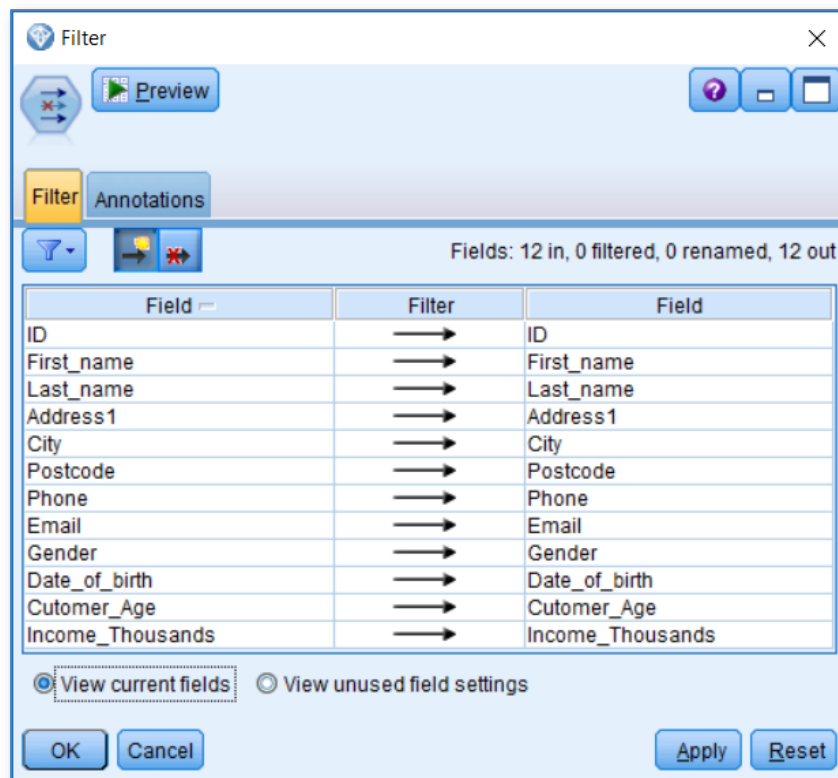


Figure 3.1 The Filter node

The filter node is especially useful when dealing with files that have a great number of irrelevant fields. Using the node, we can instantly exclude them all and choose a smaller selection of fields that we wish to include. Moreover, analysts are often confronted with field names that are unwieldy or which don't actually reflect the what the data records. The filter node also makes it easy for us to edit the field names. In our example, we can edit the filter node to exclude the following fields:

Address1

Postcode

Phone

We can also correct the misspelled field name from:

'Cutomer_Age' to 'Customer_Age'

Figure 3.2 shows the edited Filter node.

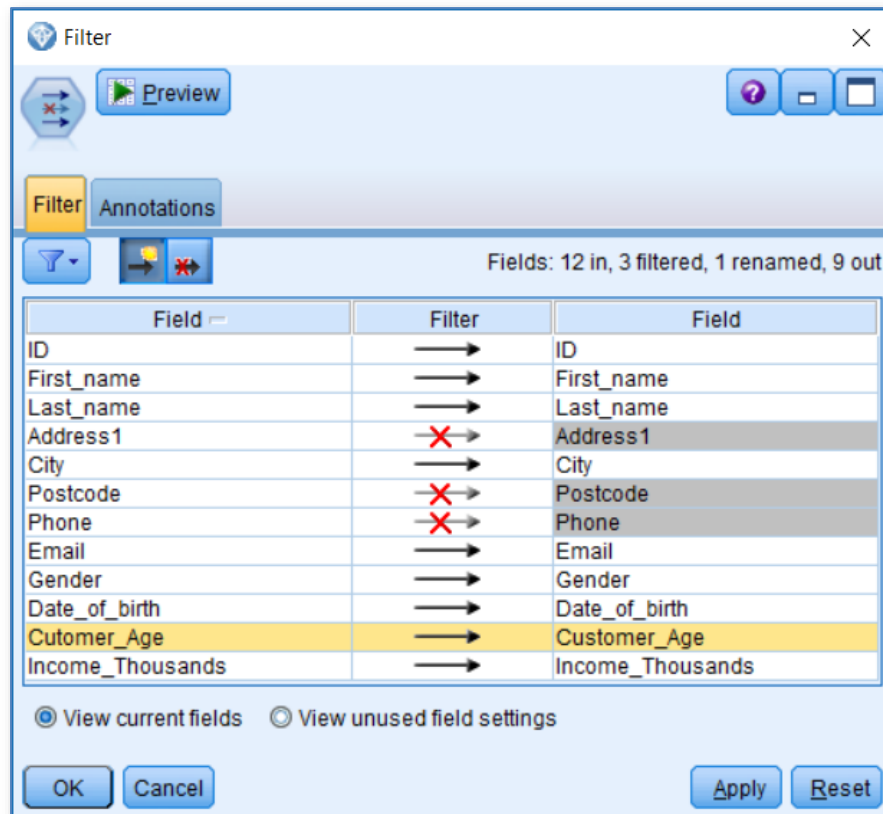


Figure 3.2 The Filter node – excluding fields and editing a field name

To check the effect of the edited node, click the button marked:

Preview

Figure 3.3 shows the results of the preview displaying the first ten records in the file.

	ID	First_name	Last_name	City	Email	Gender	Date_of_birth	Customer_Age
1	0440	Rose	Collins	ABWQ City	rose_collins@www.upoazpubr.net	F	14/03/1963	52.000
2	0121	Melissa	Coleman	ABWQ City	melissa_coleman@www.avlykext.org	F	03/01/1987	28.000
3	0469	Gerald	Young	ABWQ City		M	03/04/1929	86.000
4	0669	Brandon	King	ABWQ City	brandon_king@www.ceqxxocw.com	M	19/06/1993	22.000
5	0608	Sara	Peterson	ABWQ City	sara_peterson@www.zuqyzsbv.com	F	26/11/1960	55.000
6	0513	Ryan	Edwards	ABWQ City	ryan_edwards@www.bloazpbqm.org	M	16/10/1989	26.000
7	0697	Anne	Robinson	ABWQ City	a.robinson@www.qfuxqrsr.org	F	13/06/1979	36.000
8	0499	Douglas	Russell	ABWQ City		M	12/12/1946	69.000
9	0105	Marilyn	Martinez	ABWQ City	m.martinez@www.ajyhkpuij.net	F	26/11/1952	63.000
10	0168	Joan	Bryant	ABWQ City	joan_bryant@www.kmyhkbhe.net	F	22/01/1986	29.000

Figure 3.3 File preview from edited Filter node

Before we exit the Filter node, it is worth drawing attention to the Filter Options button in the top left of the dialog. Click:



Figure 3.4 shows the Filter Options drop-down menu.

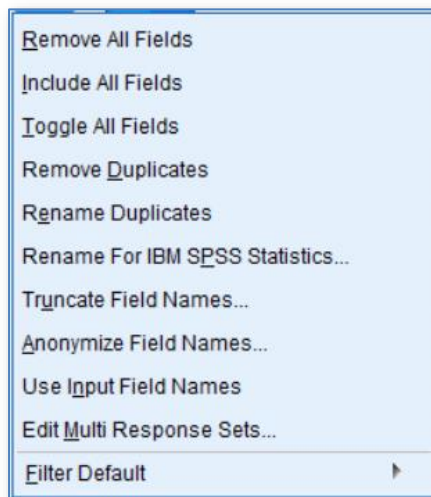


Figure 3.4 The Filter Options menu

The Filter Options menu reveals a range of additional functions such as the ability to truncate field names or anonymize them. Of particular use is the option to 'Rename For IBM SPSS Statistics'. Even if you don't have access to the SPSS Statistics software, this procedure for example, can provide a very quick way to replace all of the spaces in a set of field names with underscores. To continue working with the nodes in the stream click:

OK

The Type Node

The Type node is arguably the single most important node within the Modeler application. Given that Modeler doesn't need to import disparate file formats into its own internal database or even convert them to its own native format, it nevertheless manages to reconcile and consolidate a myriad of different data storage formats via the Type node. But as we shall see, the Type node is extremely functionally rich. With the Type node we can control the roles that fields play in the analytical process, the range of acceptable values, the definition of missing data and the treatment of invalid values. Figure 3.5 shows the contents of the example stream's Type node upon editing it.

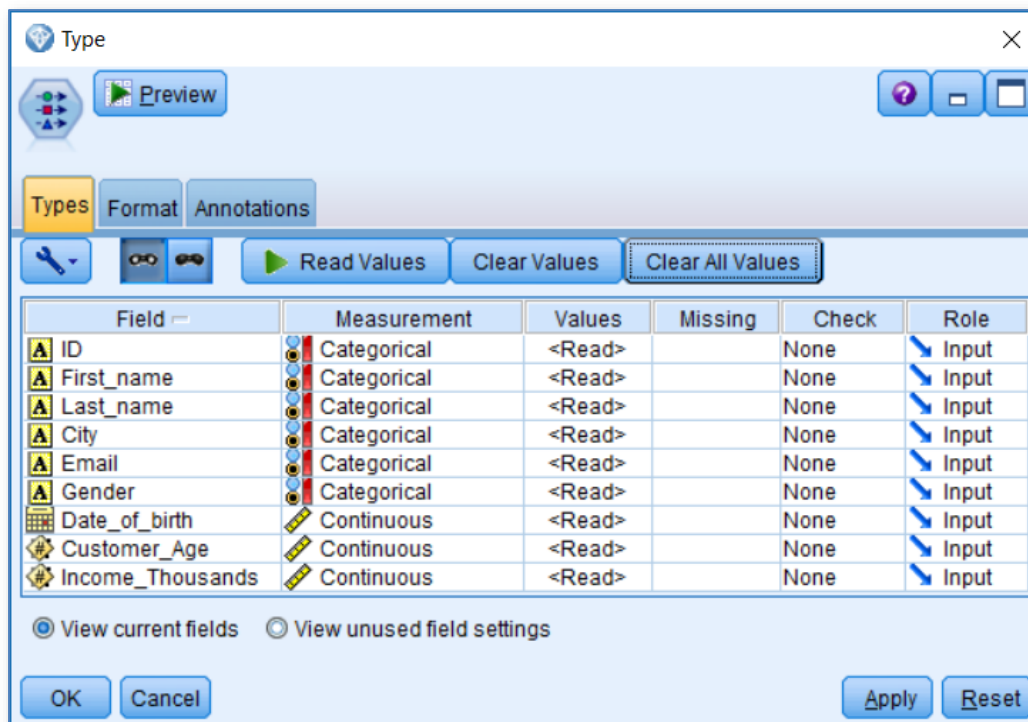



Figure 3.5 Partially instantiated Type node


When learning about the Type node, the first key concept we need to grasp is that of instantiation. Instantiation is when the data are examined to determine additional information about the file contents (i.e. to obtain meta-data) such as how the columns are stored, what types of field are available and what range of values lie within them. When Modeler encounters data where the storage type values are unknown, the data in question are regarded as uninstantiated. Uninstantiated data are set to 'Default' within the Measurement column of the type node and appear like this:

 <Default>

When Modeler has some information about the data fields, such as whether they are string or numeric in nature, the data are regarded as *Partially Instantiated*. In our example, the Var. File source node has already determined whether the data are numeric or string by virtue of the fact that, by default, it scans the first 50 rows of data. For this reason, when the Type node is attached, the data are already partially instantiated. Partial instantiation means that numeric fields are set to 'Continuous' within the Measurement column and appear as:


 Continuous


On the other hand, partial instantiation means that string fields are set to 'Categorical' in the Measurement column and appear as:


 Categorical

Full instantiation means that all of the necessary details about a field are known: including the storage type, the range of values and its 'level of measurement' (in which case the labels nominal, ordinal, flag, or continuous are displayed in the measurement column). However, the 'continuous' measurement (referring to integers and real numbers) is used for both partially instantiated and fully instantiated data fields. For categorical data, full instantiation can mean that the fields appear in the Measurement column as one of the following:

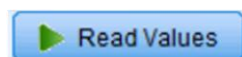
 Nominal

 Ordinal

 Flag

 Typeless

Full instantiation is a requirement for certain downstream procedures such as exporting a file in a proprietary format or a procedure that needs to read the categories within a field. To fully instantiate a Type node, the user only needs to click the following button:



In our example stream, the effect of clicking 'Read Values' within the Type node is shown in figure 3.6.

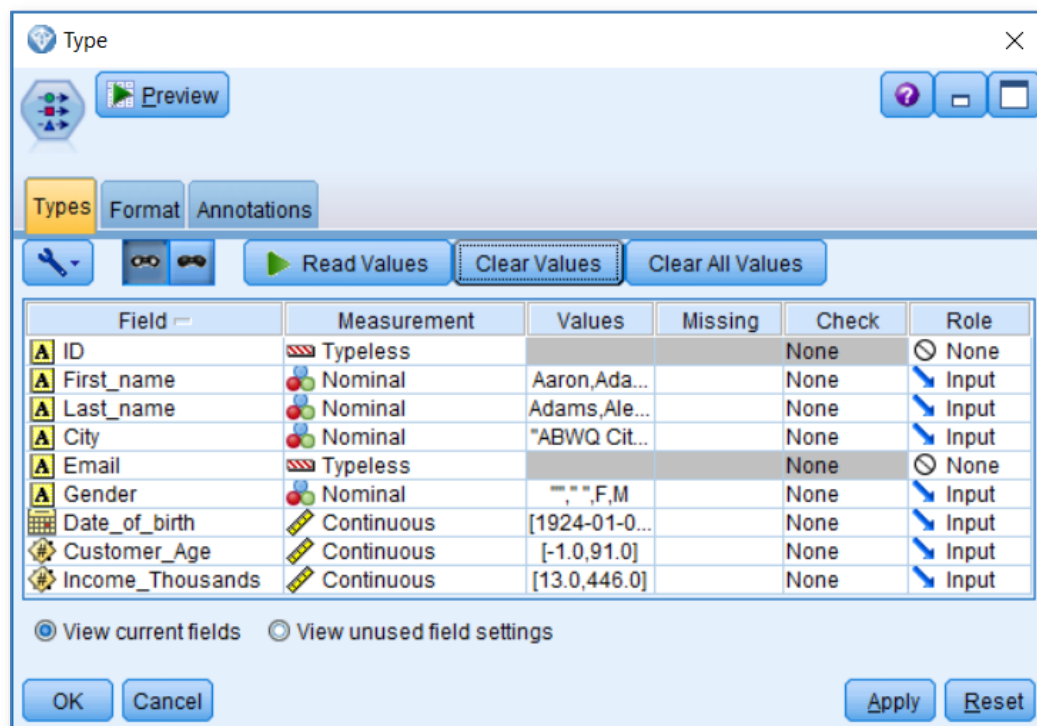


Figure 3.6 The fully Instantiated Type node

Levels of Measurement

We can instantly see the effect of clicking 'Read Values' as within the Type node many of the symbols and labels have changed. Also, we can see the range of data values for some fields in the Values column. We will take a closer look at these values a little later in this section, but for now, let's look at the different measurement levels that have been assigned.

Typeless Typeless

Before the data were fully instantiated, the fields 'ID' and 'Email' were regarded as Categorical in the same way the other string fields in the dataset were. Now, Modeler has decided to set them as *Typeless*. What is going on? Fields are regarded as typeless when they are deemed to be unlikely to be of *analytical value*. After all, we are not likely to use a field like 'Email' or 'Firstname' in an actual predictive model. By default, the application simply decides to set a column as typeless when it is a) categorical and it b) contains more than 250 unique categories (this default can be overridden). Typeless fields are not used in analytical procedures downstream.

What about the other measurement levels for the fully instantiated fields? Up until now we have spent time considering how data are *stored*, but the idea with levels of measurement is that we classify data in terms of their *analytical properties*. At this point we may refer to the data fields using a term that analysts and statisticians often employ: *variables*.

Nominal Nominal

In many ways, nominal data are characterised by their simplicity. They are comprised of values that signify different discrete categories and not much more than that. As such, the categories can be in any order and represented by coded values (such as numbers, letters or both) or by actual string descriptions of the categories. The order of the values is not important because there is no natural ranking between the categories anyway. Analysts tend to summarise nominal variables such as region, ethnicity or product type with tables of frequencies and percentages or graphically using pie or bar charts (generated by the Distribution node in Modeler).

Ordinal Ordinal

Ordinal variables are those comprised of discrete categories with a specific order or ranking. Here the order of the categories is important. In assessing student performance, teachers and lecturers commonly assign letters of the alphabet to represent each person's grade. We all know that an 'A' grade represents a better performance than a 'C' grade. The categories are discrete, but the order is very important. The most common examples of ordinal variables are rating scales such as level of satisfaction or agreement, but ordinal variables could also include

level of service ('First Class', 'Club Class' or 'Economy') or months of the year. The values in ordinal variables are not true numbers because they lack the property of equal distance between the categories. One cannot for example, say that there is the same amount of quantifiable difference between being 'Somewhat Satisfied' and 'Satisfied' as there is between 'Satisfied' and 'Very Satisfied'. There is some debate among analysts as to whether the numeric values representing different ratings or rankings can ever be regarded as true numbers (rather than numerals). Defining a variable as Ordinal as opposed to Nominal makes little difference in Modeler, although there are certain procedures that will take account of a variable's category order if it is defined as Ordinal.

Flag Flag

Flag fields are used when the variable has only two distinct values. The values could be strings such as 'Yes' and 'No' or 'True' and 'False' or indeed they could be numeric values. In which case flag fields can be comprised of any data storage type (string, integer, real or datetime).

Continuous Continuous

As we've already seen, the continuous type can be used for both partially instantiated and fully instantiated fields. In terms of storage, continuous data can be either comprised of integers or real numbers. Analysts tend to use continuous data when calculating and comparing summary statistics like means or standard deviations. In terms of graphical representations, continuous distributions are often described by histograms and relationships between continuous data with scatterplots (generated by the Plot node in Modeler).

Figure 3.7 shows the main properties of the Nominal, Ordinal and Continuous data.

Level of Measurement	Distinct Categories	Rank Order	Numeric scale	Example
Nominal	<input checked="" type="checkbox"/>			Ethnicity
Ordinal	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		Satisfaction
Continuous	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Spend

Figure 3.7 Comparing the properties of Nominal, Ordinal and Continuous data

The Check Column

The Check column within the Type node contains a series of functions that examine the data fields looking for values that are deemed invalid or which fall outside an acceptable range. The functions themselves reflect the various actions that the user can take when Modeler encounters an invalid value. The Check options represent one of the many areas where Modeler can quickly clean up a dataset early on in the CRISP-DM process.

Figure 3.8 shows the different options under the Check column.

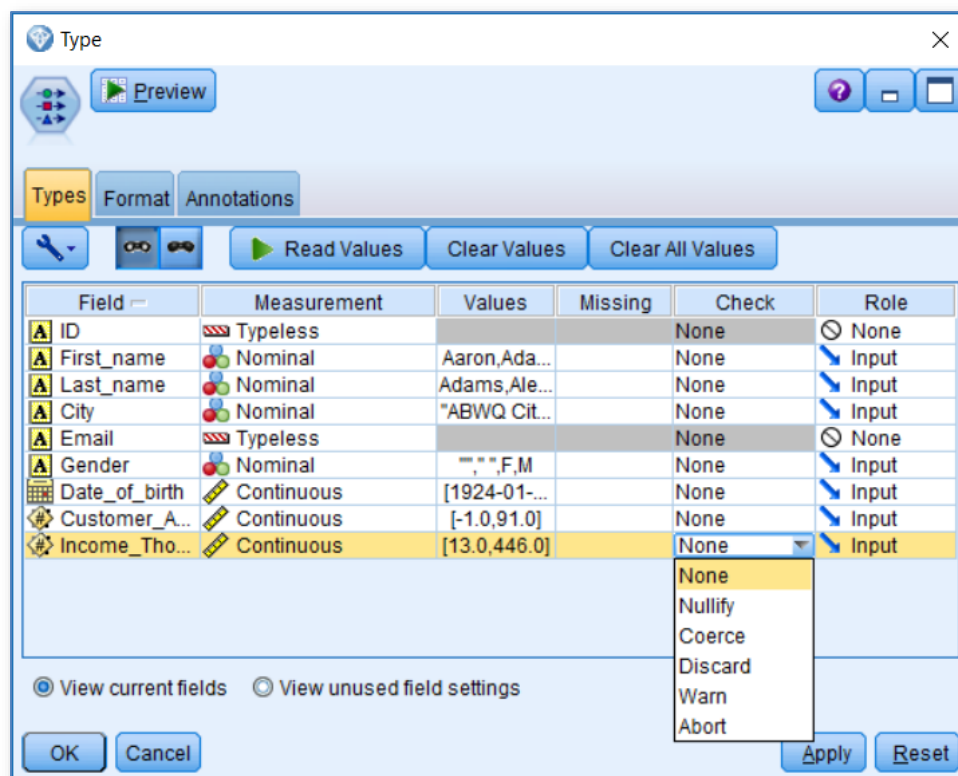


Figure 3.8 Optional settings within the Check column

None: This is the default setting where no values are checked.

Nullify: Any invalid values are changed to the system null (\$null\$).

Coerce: Invalid values will be converted using the following rules:

- For flags, any value other than the true or false value is converted to the false value.
- For sets (nominal or ordinal), any unknown value is converted to the first member of the set's values.
- For continuous data, invalid values that are greater than the highest value of an acceptable range are replaced by the upper limit within that range.

- For continuous data, invalid values that are lower than the lowest value of an acceptable range are replaced by the lower limit within that range.
- Null values in a range are given the midpoint value for that range.

Discard: When invalid values are found, the entire record is discarded.

Warn: A warning report displaying the rule violation is shown in the stream properties dialog box.

Abort: Upon encountering an invalid value, the procedure terminates the stream execution and the error is reported in the stream properties dialog box.

By default, the Check column will treat existing 'Null' values within continuous variables as invalid and will act accordingly. To illustrate this:

For the 'Income_Thousands' field, set the Check column to the 'Warn' option and run the stream

You may notice that a small warning symbol appears on the task bar along the bottom of the main application window. Clicking the warning symbol...



...reveals that Modeler encountered several null values (\$null\$) within the field. Figure 3.9 shows this.

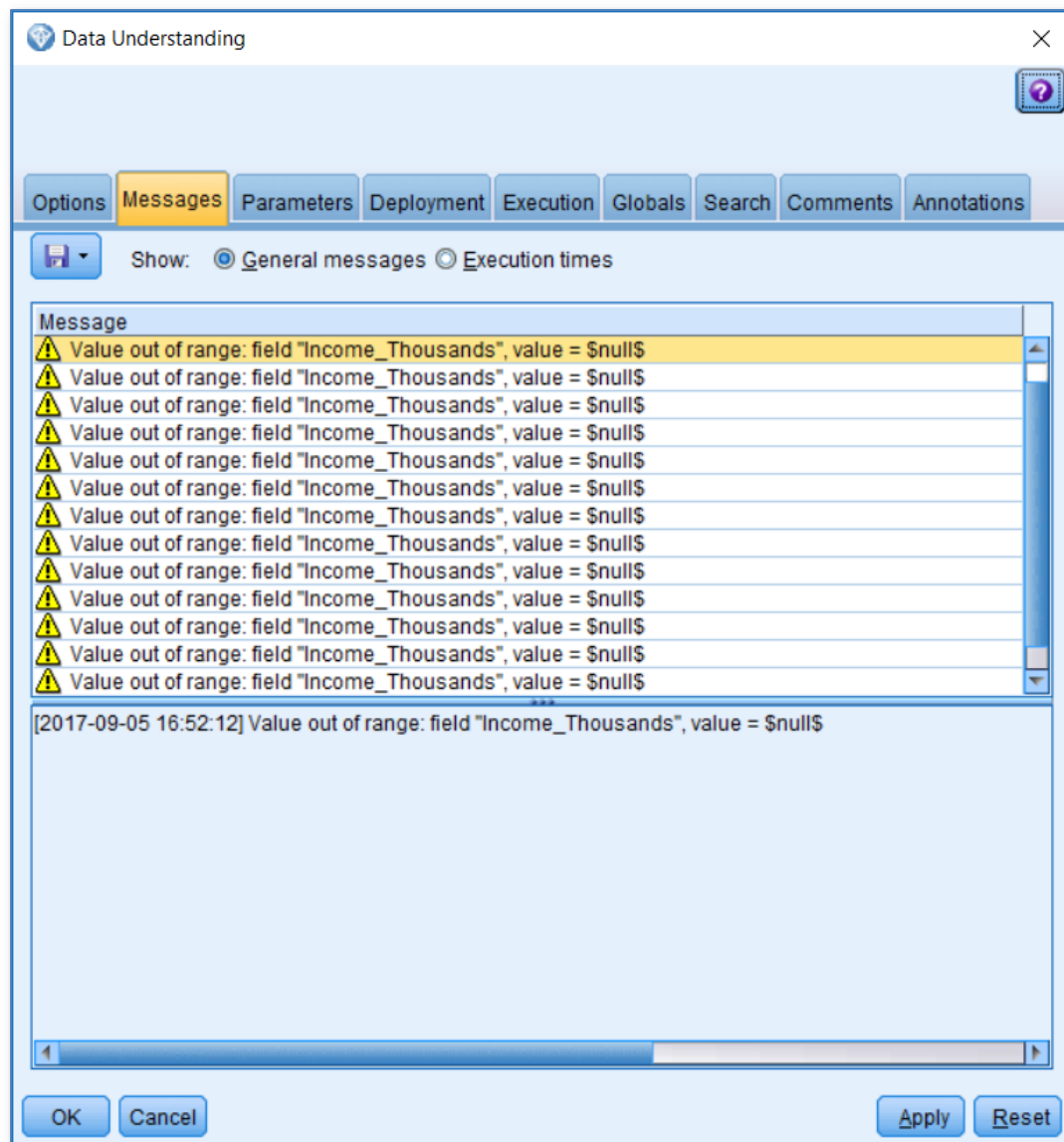


Figure 3.9 Check column generates warnings about invalid Null values in the variable 'Income_Thousands'

To see the effect of changing the action in the Check column:

Edit the type node and for the 'Income_Thousands' field, set the Check column to the 'Discard' option and run the stream

On running the stream, the table node shows that of the original 850 records 837 have been passed downstream. So changing the Check option to discard records with invalid values in this variable has resulted in 13 records being dropped.

Figure 3.10 shows the table node after we changed the Check action to 'Discard'.

Table (9 fields, 837 records) #2

FileEditGenerate

File options

ons

	ID	First_name	Last_name	City	Email	Gender	Date_of_birth	Customer_Age
1	0440	Rose	Collins	ABWQ City	rose_collins@www.upoazpubr.net	F	14/03/1963	52.000
2	0121	Melissa	Coleman	ABWQ City	melissa_coleman@www.avlykexl.org	F	03/01/1987	28.000
3	0469	Gerald	Young	ABWQ City		M	03/04/1929	86.000
4	0669	Brandon	King	ABWQ City	brandon_king@www.ceqxxocw.com	M	19/06/1993	22.000
5	0608	Sara	Peterson	ABWQ City	sara_peterson@www.zuqzsbv.com	F	26/11/1960	55.000
6	0513	Ryan	Edwards	ABWQ City	ryan_edwards@www.bloazpbqm.org	M	16/10/1989	26.000
7	0697	Anne	Robinson	ABWQ City	a.robinson@www.qfuxqrsb.org	F	13/06/1979	36.000
8	0499	Douglas	Russell	ABWQ City		M	12/12/1946	69.000
9	0105	Marilyn	Martinez	ABWQ City	m.martinez@www.ajyhkpuj.net	F	26/11/1952	63.000
10	0168	Joan	Bryant	ABWQ City	joan_bryant@www.kmyhkbhe.net	F	22/01/1986	29.000
11	0659	Beverly	Brooks	ABWQ City		F	26/12/1925	90.000
12	0558	Martin	Butler	ABWQ City	martin_butler@www.ndsiacmo.net	M	22/06/1964	51.000
13	0782	Betty	Williams	ABWQ City	b.williams@www.hjsiaiqe.net	F	12/05/1972	43.000
14	0129	Andrea	Murphy	ABWQ City	a.murphy@www.yyoazpudg.net	F	03/09/1983	32.000
15	0304	Angela	Hernandez	ABWQ City		F	04/04/1944	71.000
16	0433	Virginia	Martinez	ABWQ City	virginia_martinez@www.xwqzilh.org	F	09/11/1990	25.000
17	0055	Anna	Lewis	ABWQ City	a.lewis@www.ijoazpkmc.com	F	18/08/1934	81.000
18	0275	Phillip	White	ABWQ City	p.white@www.eeyhkekv.net	M	18/10/1968	47.000
19	0124	Kathy	Patterson	ABWQ City		F	27/12/1931	84.000
20	0474	Gregory	Anderson	ABWQ City	g.anderson@www.ewuxqtsb.net	M	03/06/1995	20.000

1

OK

Figure 3.10 Table node showing 837 records after the Check option was changed to 'Discard' for the variable 'Income_Thousands'

The Values Column

But what if the user wants to specify a valid range of values for checking? To do this, we need to edit the contents of the 'Values' column. Figure 3.11 shows the drop-down menu that appears when we edit the values for the variable 'Customer_Age'.

Field	Measurement	Values	Missing	Check	Role
ID	Typeless			None	None
First_name	Nominal	Aaron, Ada...		None	Input
Last_name	Nominal	Adams, Al...		None	Input
City	Nominal	"ABWQ Cit...		None	Input
Email	Typeless			None	None
Gender	Nominal	"", "F, M		None	Input
Date_of_birth	Continuous	[1924-01-...		None	Input
Customer_Age	Continuous	<Curre...		None	Input
Income_Thousands	Continuous	<Read>		None	Input

Figure 3.11 The drop-down menu options within the Values column

Before we even get to the specifying a valid range of values, the drop-down menu shows that we have control over instantiation at the level of individual fields. This is because if we alter the nature or values of a field (via a Modeler node such as Derive for instance), the current storage type or range of values for that field may no longer be valid, so we may need to instantiate the individual field again to take account of the changes. Using the Clear Values button, you can clear any changes to the range of values made in this node (non-inherited values) and reread the values from upstream operations. This option is useful for resetting changes that you may have made for specific fields upstream.

The field values options are as follows:

<Read>	This is the standard option. The data is read when the node is executed and the existing range of values is determined.
<Read+>	This scans the data and looks for any new values added to the data. It then appends the values and updates the node.
<Pass>	The data is not scanned.
<Current>	Retains the current range of data values.
Specify...	This opens a separate dialog box, so the user can specify the values, missing values and the variable's measurement level.

Figure 3.12 The various options for determining field values within the 'Values' column of the Type node

You will also notice that near the top of the dialog you have two options controlling the clearing of the existing instantiation within the type node. Remember that you may have more than one Type node in a stream. The Type tab within the source node may already have read the values and instantiated the data. If this was so, and we placed a Type node downstream of it, we would notice that the second node had 'inherited' earlier Type tab's values and that furthermore, the values themselves were greyed out (inactive). If we edit these values in any way, we would immediately activate the controls for the individual edited fields. In other words, Modeler doesn't have to instantiate the entire dataset again (which could be a time-consuming process with a large file). If at the same time, we only wanted to clear the values of the newly edited fields, we could do this by clicking the 'Clear Values' button as this will only affect the edited values in the local Type node. On the other hand, clicking 'Clear All Values' will affect all the fields in the type node (including those with 'inherited' values). To manually edit the values within a given field.

Click the cell within the Values column for the variable 'Customer_Age'.

From the drop-down menu click:

Specify...

Figure 13.13 shows the resultant 'Values' dialog.

Figure 3.13 The Values dialog for the variable 'Customer_Age'

There are two important aspects of this dialog. Firstly, we can edit the Lower and Upper Boundary Limits of the range of acceptable values (in this case Modeler can see values for customer age running from -1 to 91). The second aspect is that we can specify *missing values*. Modeler refers to missing values as 'Blanks', though in fact these could take the form of null values, empty strings or a value (or range of values) that the user specifies.

Figure 3.14 shows this dialog edited so that the valid range of values runs from 0 to 91 (note also that we can set the Check values option here as well). The table output shows that when the 'Discard' option is switched on, we end up with 849 records. In other words, only one record with a value of -1 has been discarded.

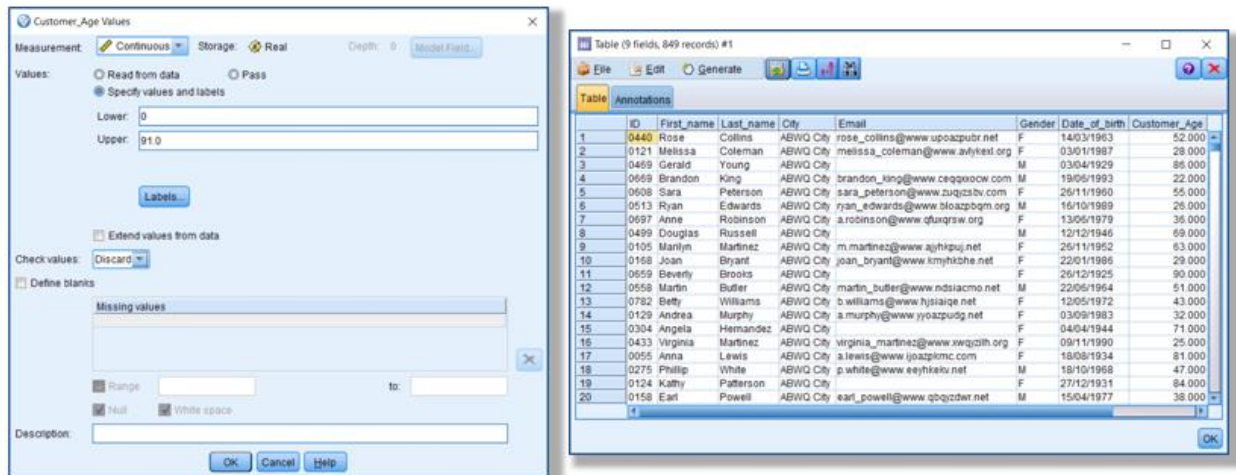


Figure 3.14 One record discarded with the valid range defined as 0 to 91

Figure 3.15 shows the dialog edited again but now the valid range of values runs from 18 to 91. The table output now indicates that 3 records have been discarded (847 have been passed to the output).

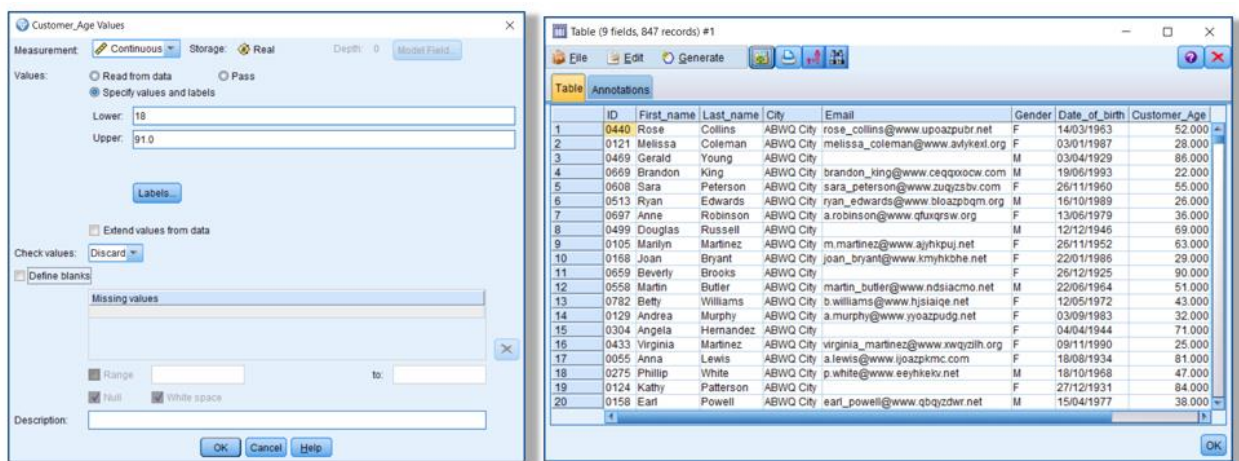


Figure 3.15 Three records discarded with the valid range defined as 18 to 91

Missing Values

Now let's investigate the effect of defining one of these values as 'missing'. Figure 3.16 shows what happens when we edit the dialog and check the box marked 'Define blanks'.

By default, when we switch on 'missing values', Modeler looks for two kinds of missing values:

- System null (\$null\$) values in numeric continuous fields
- Empty string values and white space (strings with no visible characters)

However, Modeler also offers us the opportunity to define our own blank values (effectively *user-missing values*) either as discrete values, a range of values or a combination of both. In this case we have:

Checked the box marked 'Define Blanks'

Entered the value '-1' as a Missing Value

Bear in mind that the valid value range *still* runs from 18 to 91. Perhaps in this example, the user has decided that the value -1 has *been deliberately entered* to indicate that the customer's age is unknown whereas the other values below 18 are just mistakes and should be treated differently. Missing data in Modeler is handled in different ways by different techniques so defining values as *missing or blanks* is a way of flagging this early in the analysis process.

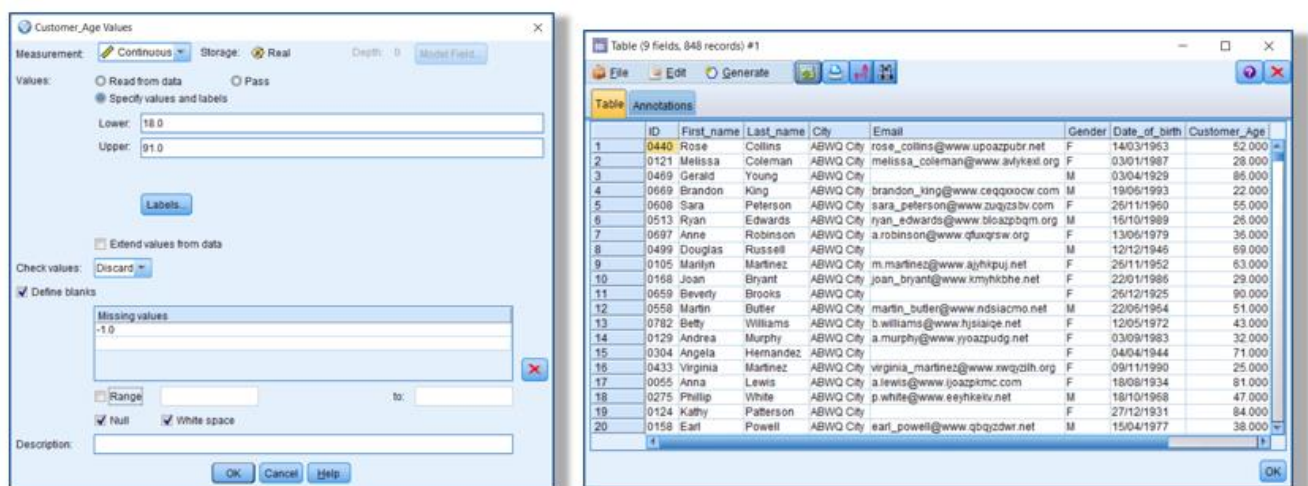


Figure 3.16 Two records discarded with the valid range defined as 18 to 91 and with -1 defined as a missing value

Figure 3.16 shows an interesting result. The case with the now (user-defined) blank value of -1 has not been discarded (it has the value 0768 in the 'ID' column). This is because defined blanks (or user-missing values) take precedence over the actions in the Check column. User-missing values are valid values that are dealt with appropriately within the context of the analytical technique. Moreover, we can switch on the default missing values for any column simply by checking the corresponding cell for the variable under the 'Missing' column of the Type node. Figure 3.17 shows the Type node with the missing values for the field 'Customer_Age' switched on. You may notice that this is indicated by the presence of an asterisk in the column.

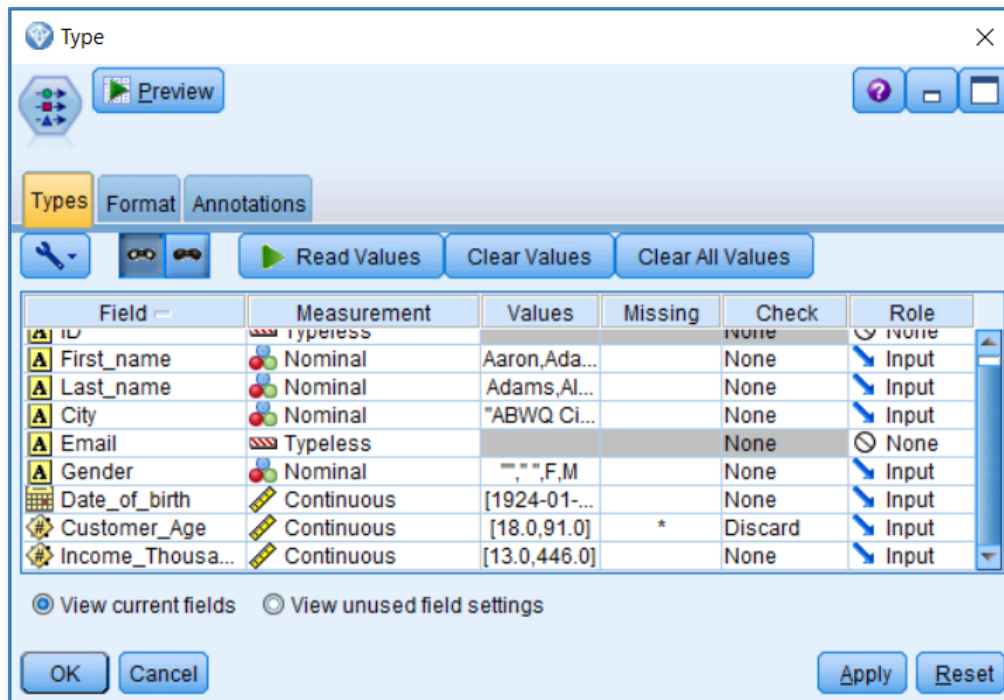


Figure 3.17 Type node with Missing Values defined for the variable 'Customer_Age'

Before continuing further:

Change the discard option under the Check menu for 'Customer_Age' to 'Nullify'

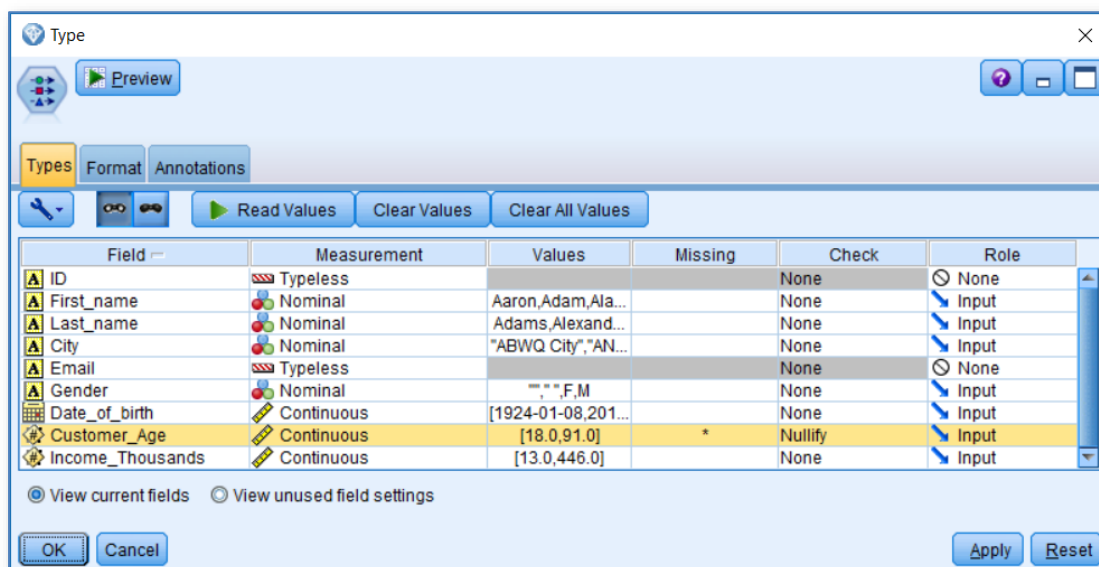


Figure 3.18 Changing the Check option to 'Nullify' in 'Customer_Age'

The Variable Role

The final column within the node is called the 'Role'. This column allows to define how a variable is to be used analytically. For this reason, typeless variables have their role set to None as they have no analytical use. By default, Modeler sets all the other fields to the 'Input' role as it is assumed that they will be used to help direct the analysis or predict an outcome. Figure 3.19 displays the various analytical roles that the data fields may play within the Type node.

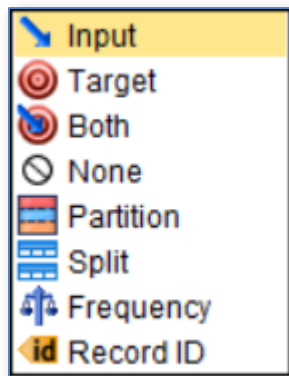


Figure 3.19 Options under the 'Role' column in the Type node

The variable Role options can be summarised as follows.

Input - The field will be used as an input (or predictor) for a predictive modelling algorithm.

Target - The target role sets the field to be the outcome that a model attempts to predict. For example, outcomes such as campaign response, credit worthiness, customer churn or fraud are all commonly defined as the Target field within the Type node.

Both - Certain association model algorithms (such as Apriori) allow fields to serve the role of Target or Input. Setting the role to 'Both' will enable this.

None - The field will be ignored by most analytical procedures.

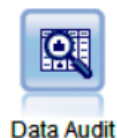
Partition - Partition fields are specially created for the purposes of testing model accuracy. They work by splitting the data into separate samples for training, testing, and (optionally) validation purposes. The model is built using the training partition and then applied to the test (and possibly the validation) sample for see how well it performs. As we will see later, the Partition *node* is used to create fields that do precisely this.

Split - The split role only applies to nominal, ordinal or flag fields. It specifies that a model is to be built for each possible value of the field.

Frequency - The frequency role only applies to numeric fields. The role enables a field value to be used as a frequency weighting factor for the record. The Frequency role is only supported by certain algorithms (C&R Tree, CHAID, QUEST and Linear models).

Record ID – This role indicates that the field will be used as the unique record identifier. It is supported by Linear models though most algorithms ignore it.

The Data Audit Node



The Data Audit node provides more functionality for inspecting and cleaning the data sources. As such, it is a key tool for analysts working through the 'Data Understanding' phase of CRISP-DM. Attaching the Data Audit node (from the Output palette) to the Type node in the demonstration stream (as shown in figure 3.20) and executing it generates output that provides us with a detailed overview of the data.

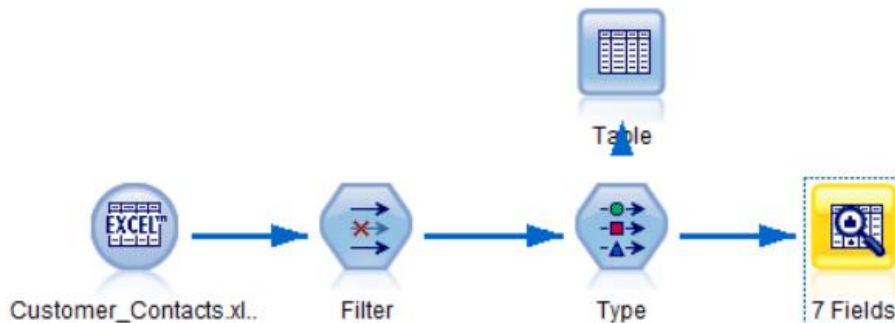


Figure 3.20 The Data Audit node attached to the Type node in the demonstration stream

Figure 3.21 shows the initial results from the Data Audit output.

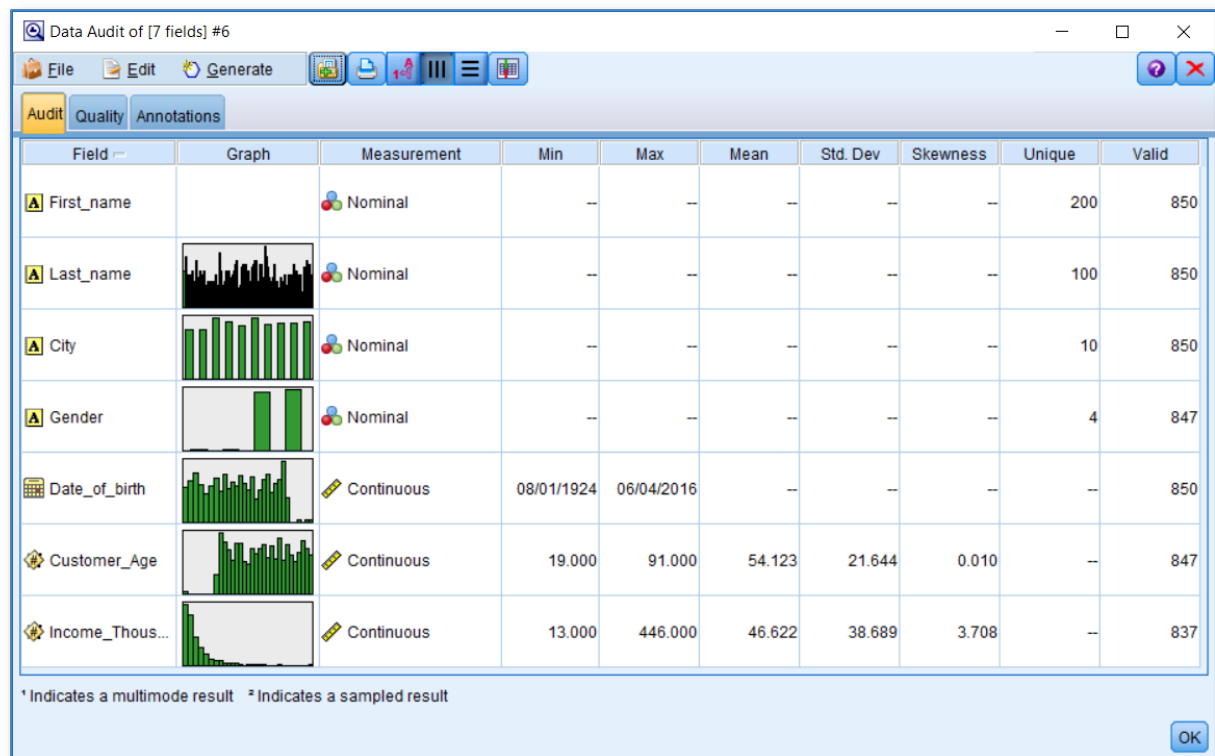


Figure 3.21 Output from the Data Audit node

The output from the data audit node provides an immediate high-level view of the data. Categorical fields are displayed with distribution charts, whilst histograms are used to represent the continuous ones. Summary statistics are calculated and displayed. Again, Modeler takes account of each field's level of measurement, displaying statistical summaries for continuous data such as the mean, minimum, maximum and standard deviations (additional statistical measures may be requested). The final column displays the number of valid cases for each field in the dataset (excluding records with the default missing categories of system nulls and empty strings). These values can be useful when spotting errors in the data especially by paying close attention to the minimum and maximum values or the charts themselves. In the example here, we can see that the variable 'Gender' has four categories. If we double-click the distribution chart corresponding this field, we can see why. Figure 3.22 shows the chart.

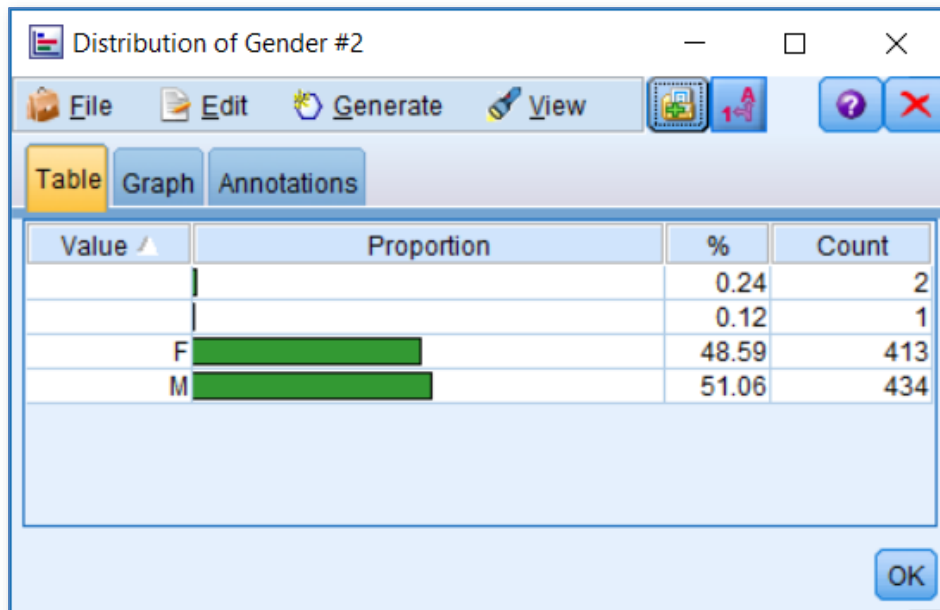


Figure 3.22 Distribution chart of the variable 'Gender'

We can immediately see from the distribution chart that in addition to the values 'F' and 'M', the variable contains two blank categories. Presumably, one of these categories consists of invisible characters (such as spaces) and the other contains no string values at all. You may also notice that in the main Data Audit Output window this field had 347 valid cases. This is because the three apparently blank cases are not counted as 'valid'.

The output from the Data Audit node also provides many quality assessment and data cleaning functions. To access these, from within the output window, click the tab marked:

Quality

Figure 3.23 shows an annotated image of the Data Audit Quality tab.

Field	Measurement	Outliers	Extremes	Action	Impute Missing	Method	% Complete	Valid Records	Null Value	Empty String	White Space	Blank Value
First_name	Nominal	0	0	---	Never	Fixed	100	850	0	0	0	0
Last_name	Nominal	0	0	---	Never	Fixed	100	850	0	0	0	0
City	Nominal	0	0	---	Never	Fixed	100	850	0	0	0	0
Gender	Nominal	0	0	---	Never	Fixed	99.647	847	0	2	3	0
Date_of_birth	Continuous	0	0 None	---	Never	Fixed	100	850	0	0	0	0
Customer_A	Continuous	0	0 None	---	Never	Fixed	99.647	847	2	0	0	3
Income_Tho	Continuous	9	7 None	---	Never	Fixed	98.471	837	13	0	0	0

Figure 3.23 Annotated image of the Data Quality Tab

Field	Measurement	Outliers	Extremes	Action	Impute Missing	Method	% Complete	Valid Records	Null Value	Empty String	White Space	Blank Value
First_name	Nominal	0	0	---	Never	Fixed	100	850	0	0	0	0
Last_name	Nominal	0	0	---	Never	Fixed	100	850	0	0	0	0
City	Nominal	0	0	---	Never	Fixed	100	850	0	0	0	0
Gender	Nominal	0	0	---	Never	Fixed	99.647	847	0	2	3	0
Date_of_birth	Continuous	0	0 None	---	Never	Fixed	100	850	0	0	0	0
Customer_A	Continuous	0	0 None	---	Never	Fixed	99.647	847	2	0	0	3
Income_Tho	Continuous	9	7 None	---	Never	Fixed	98.471	837	13	0	0	0

1. The Quality tab shows the percentage of fields that have complete cases and the overall percentage of records that are complete.

Field	Measurement	Outliers	Extremes	Action	Impute Missing	Method	% Complete	Valid Records	Null Value	Empty String	White Space	Blank Value
First_name	Nominal	0	0	---	Never	Fixed	100	850	0	0	0	0
Last_name	Nominal	0	0	---	Never	Fixed	100	850	0	0	0	0
City	Nominal	9	7	---	Never	Fixed	100	850	0	0	0	0

2. This part of the quality output shows how many cases have 'Outlier' or 'Extreme' values in each field. The thresholds that define whether a value is seen as an outlier or an extreme can be edited within the Data Audit node prior to execution. The default setting is that outlier cases are those which have values that are more than 3 standard deviations above or below the variable's mean. Whereas extremes are those that are more than 5 standard deviations above or below the mean.

Field	Measurement	Outliers	Extremes	Action	Impute Missing	Method	% Complete	Valid Records	Null Value	Empty String	White Space	Blank Value
First_name	Nominal	0	0	---	Never	Fixed	100	850	0	0	0	0
Last_name	Nominal	0	0	---	Never	Fixed	100	850	0	0	0	0
City	Nominal	9	7	---	Never	Fixed	100	850	0	0	0	0

3. We saw in the Type node that we can trigger actions to deal with values that are deemed to be 'invalid'. In the same way, within the Quality tab, we can also trigger actions that deal with outlier/extreme values. Here we might choose to coerce, discard or nullify outliers or extremes. To make this take effect, we need to use the Generate menu to create the relevant nodes to complete the action.

4

Impute Missing	Method
Never	Fixed
Never	Fixed
Never	Fixed
Never	Fixed
Never	Fixed
Never	Fixed
Never	Fixed

4. In this section of the quality tab we are offered greater control as to how we handle missing data. The 'Impute Missing' column allows to specify *how we define missing values* such as null values, blank values or according to a specified condition. The dialog within the 'Impute Missing' column that enables this, will also let us populate the values in the next column marked 'Method'. Within the Method column, we can choose *how* we wish to deal with our defined missing data: the options include a random number, a fixed value, a user-defined expression or even an algorithm to *predict* the missing value. Again, in order to execute these actions, we would need to use the Generate menu to create the relevant nodes.

5

% Complete	Valid Records	Null Value	Empty String	White Space	Blank Value
100	850	0	0	0	0
100	850	0	0	0	0
100	850	0	0	0	0
99.647	847	0	2	3	0
100	850	0	0	0	0
99.647	847	2	0	0	3
98.471	837	13	0	0	0

5. The Last section of output within the Quality tab provides a report on missing data. The percentage complete and the number of valid records is shown for each field. As we can see, the Null Value column shows counts of two and thirteen in the 'Customer_Age' and 'Income_Thousands' fields respectively. Interestingly, for the variable 'Gender', we can see two cases that are marked as 'Empty String' and three cases marked as 'White Space'. When we looked at the distribution chart for Gender earlier, we noticed only three records with apparently empty values so why does it look like there are five records with missing data in total here? The answer is that 'White Space' within the Quality tab refers to *both strings with no values and strings with invisible characters*. The two cases in the Empty String count here are those with *no values*, meaning that there must be one other case with *invisible characters* to make up the White Space count to three. Lastly, we encounter the Blank value column. This counts cases with values that have been *declared as missing* within the Type node. Notice that the field 'Customer_Age' has a Blank Value count of three. Remember that within the Type node we declared Null Values and the value '-1' as *missing*? For that reason, we have total count of three as there are two cases with the value '\$null\$' and one with the value '-1'. The last variable, 'Income_Thousands' has a Null Value count of thirteen but zero in the Blank Value count as Null Values for this field were not declared as missing in the type node.

To demonstrate how we can use the output in the Data Audit node in conjunction with the Generate menu, within the output itself, on the main menu click:

Generate

The drop-down Generate menu shows the basic functionality for dealing with missing values (as shown in figure 3.24). The menu offers us the option to generate either a 'Missing Values Filter Node' that filters out *fields* with an excessive proportion of missing values, or a 'Missing Values Select Node' that will select *records* that contain missing data. From the drop-down menu, click:

Missing Values Select Node

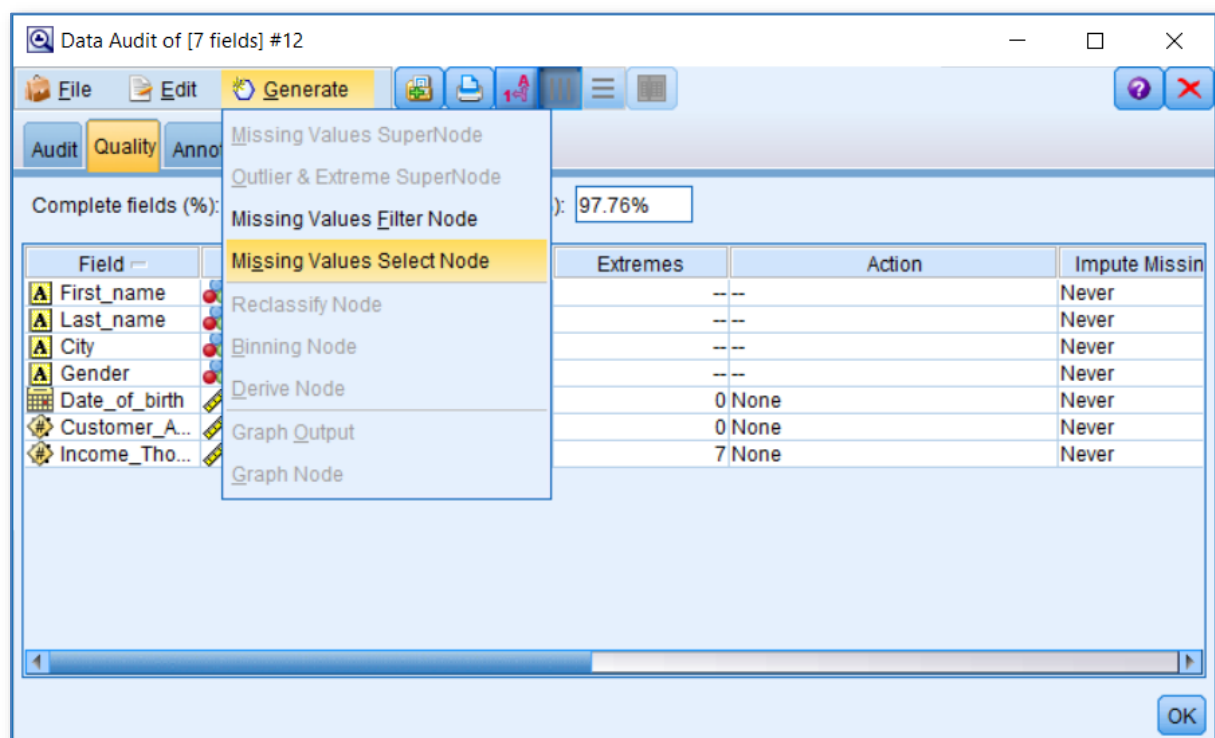


Figure 3.24 Using the Generate menu within the Quality tab of the Data Audit output

Looking at figure 3.25, the 'Generate Select Node' sub-dialog allows the user to specify when a record is selected, and which fields should be included in the selection procedure.

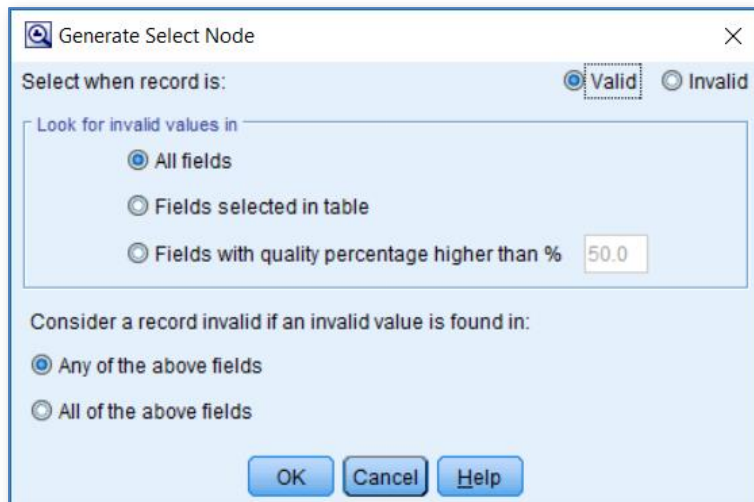


Figure 3.25 Generate Select Node sub-dialog

For this example, we will use the sub-dialog to select only the valid records. In the section marked 'Select when record is', click the radio button marked:

Valid

Now click:

OK

The procedure will now place a new select node on the stream canvas.

Connect the Select node to the Type node and use Copy & Paste to connect another Data Audit node to the end

The stream should look like Figure 3.26.

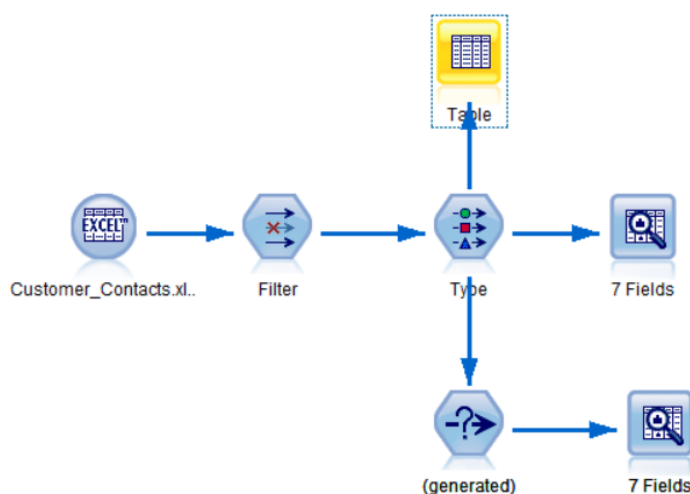


Figure 3.26 The newly Generated Select Node connected to a new Data Audit Node

When we run the new Data Audit node and view the results, we can see within the Quality tab that none of the fields have missing data.

Data Audit of [7 fields] #17

Complete fields (%): 100% Complete records (%): 100%

Field	Measurement	Outliers	Extremes	Action	Impute Missing	Method	% Complete
First_name	Nominal	--	--		Never	Fixed	100
Last_name	Nominal	--	--		Never	Fixed	100
City	Nominal	--	--		Never	Fixed	100
Gender	Nominal	--	--		Never	Fixed	100
Date_of_birth	Continuous	0	0 None		Never	Fixed	100
Customer_Age	Continuous	0	0 None		Never	Fixed	100
Income_Thousands	Continuous	8	7 None		Never	Fixed	100

Figure 3.27 Data Audit output from the newly Generated Select Node showing all fields are 100% complete

Again, purely to demonstrate the functionality within this output:

Click to select the last field 'Income_Thousands'

Click on the corresponding cell under the Action column

From the drop-down menu, select 'Coerce outliers/discard extremes'

Data Audit of [7 fields] #17

Complete fields (%): 100% Complete records (%): 100%

Field	Measurement	Outliers	Extremes	Action	Impute Missing	Method
First_name	Nominal	--	--		Never	Fixed
Last_name	Nominal	--	--		Never	Fixed
City	Nominal	--	--		Never	Fixed
Gender	Nominal	--	--		Never	Fixed
Date_of_birth	Continuous	0	0 None		Never	Fixed
Customer_Age	Continuous	0	0 None		Never	Fixed
Income_Thousands	Continuous	8	7	None Coerce Discard Nullify Coerce outliers / discard extremes Coerce outliers / nullify extremes	Never	Fixed

Figure 3.28 Choosing an action to deal with Outliers and Extremes within the Quality tab

Again, click to select the last field 'Income_Thousands' so that it is highlighted

From the main menu click:

Generate

Outlier and Extreme Super Node

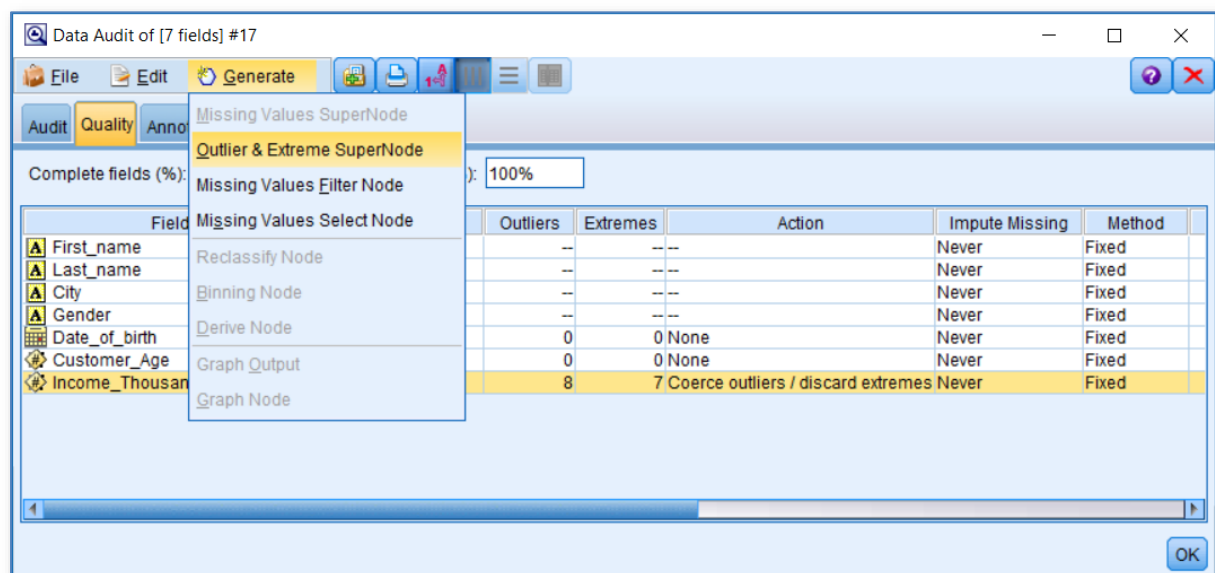


Figure 3.29 Generating a supernode to deal with Outliers and Extremes

A small dialog appears asking which fields the supernode should apply to. Choose:

Selected fields only

OK

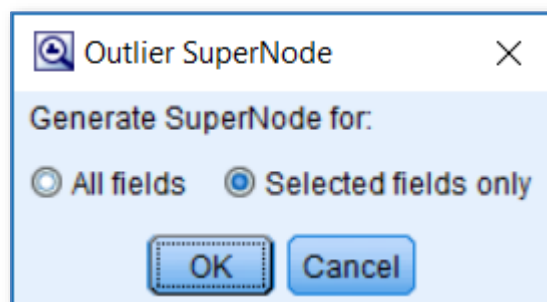


Figure 3.30 Outlier SuperNode sub-dialog

The procedure will have generated a supernode and placed it on the stream canvas. If we add the new supernode and an additional Data Audit node to the stream as shown in figure 3.31 we can view the effect of the coercing outliers and discarding extremes on the data.

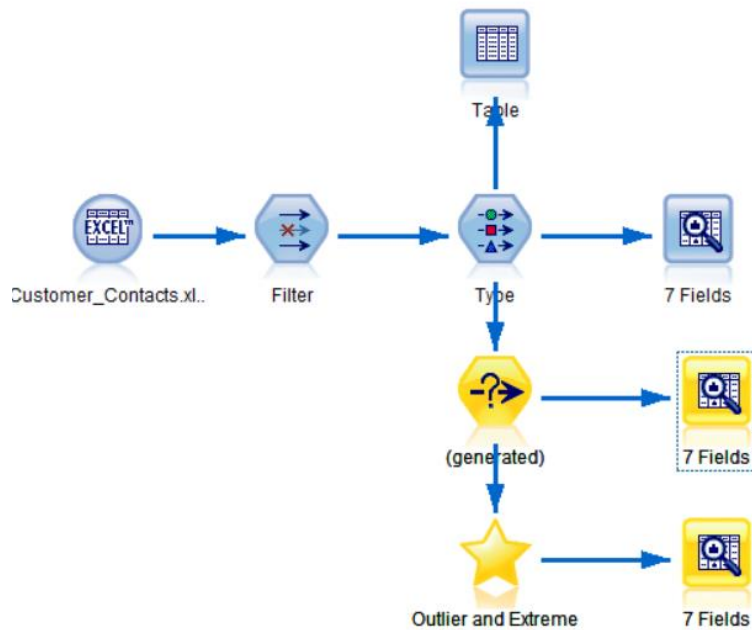


Figure 3.31 The newly Generated Outlier and Extreme SuperNode connected to the previously generated Select node

Figure 3.32 shows the difference in the data before and after the supernode has been added.

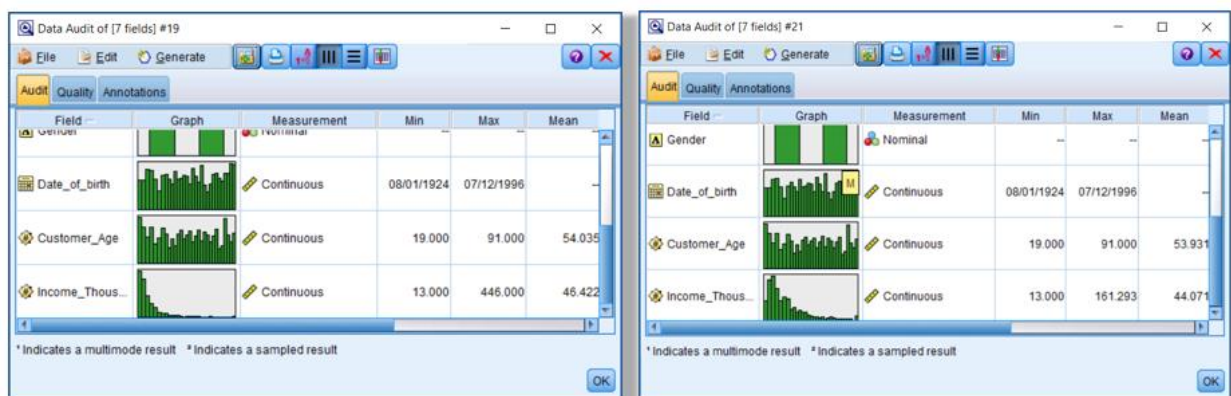


Figure 3.32 The Data Audit node before and after the Outlier/Extreme supernode is included in the stream

Figure 3.32 shows that before the outlier/extreme supernode was added, the maximum value for the variable 'Income_Thousands' was 446.000. After adding the supernode the effect of discarding extreme values and coercing outliers means that the maximum value is now 161.293.

Save the completed stream as:

Data Understanding Complete.str

Section 4:

Restructuring Data



- The Distinct Node
- The Aggregate Node
- The Set to Flag Node
- The Web Node

A key concern for data analysts engaged in predictive analytics projects is defining and creating the 'unit of analysis'. Put simply, this refers to what a row of data needs to represent in order for the analysis to make sense. If, for example, the analytical goal of the project is to predict which customers are likely to redeem a voucher, then the predictive model will probably require data where each row is a *different customer* not a *different transaction*. After all, a person represents a single entity that can make many visits, with several transactions of multiple items. If, on the other hand, the analytical goal is to identify fraudulent attempts to purchase tickets for an event, then the model probably requires sample data where each row is a *transaction*. So whether the analysis outcome is focussed on people, businesses or physical assets, as opposed to events or instances, makes a big difference with regard to how we represent this in a dataset for modelling purposes. For these reasons, when the analytical goal requires data at the *customer* level, if the analyst wants to utilise information such as the customers' transaction history, their interactions with the business and the various products or services they purchase, they need to find a way to summarise the multiple rows data that record these events and represent them as a single record for each person.

The Distinct Node



Before we even begin the process of summarising multiple transaction records as a single row of data, it makes sense for us to check the integrity of the data we have about the customers. For example, are we sure that we have a single row of data for each person? Duplications in data files are a regular headache for most businesses and they can occur for lot of reasons. Often, the customers may register themselves more than once; contact details may be updated causing an extra row of data to be added rather than overwriting the existing one; merging information from different departments or organisations may create duplicates because there isn't an exact match. In any case, Modeler helps us to identify duplicate cases and resolve these issues with the Distinct node. To begin the demonstration, start with a new blank stream canvas and using a Var. File node open the following comma separated file from within the Section 4 folder:

Health_Club_Contacts.csv

Figure 4.1 shows the edited source node and the preview data output for this file.

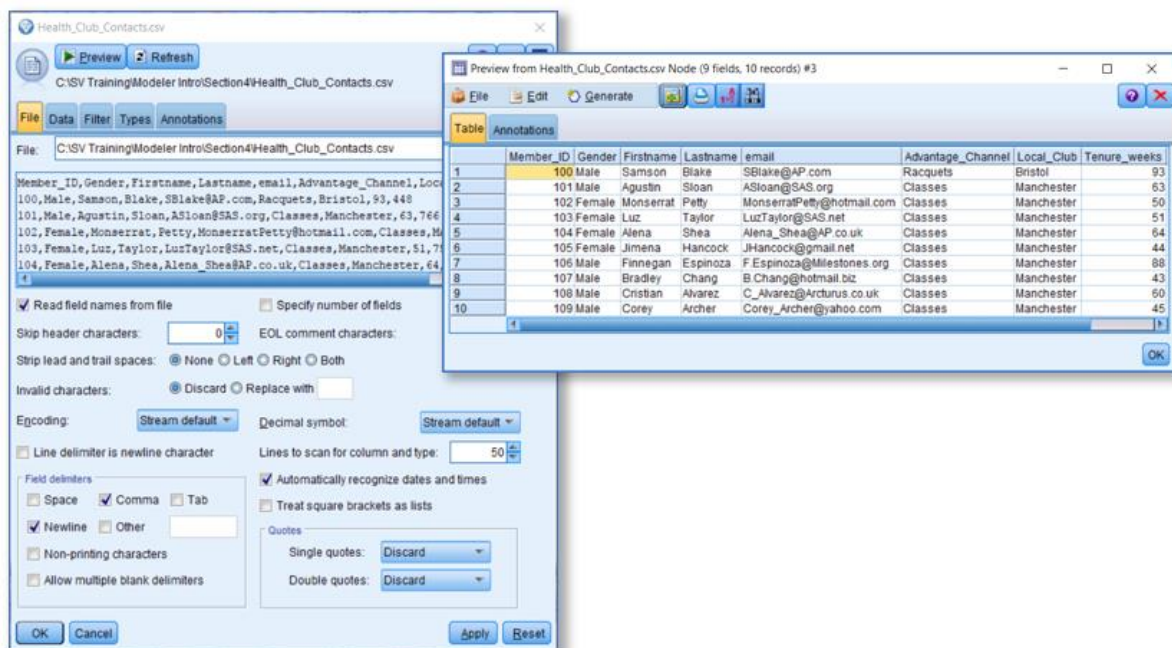


Figure 4.1 Source node containing the file 'Health_Club_Contacts.csv'

If we attach a Table output node to this data source and read the full file, we will see that it is comprised of 3,818 records. By attaching the Distinct node, we should be able to identify if there are any duplicated records. From the Record Ops palette:

Select and attach the Distinct node to the Source node as shown if figure 4.2



Figure 4.2 Distinct Node attached to data source node

Let's take a look at the functionality on offer in this node.

Edit the Distinct Node

Figure 4.3 shows the default setting in the node.

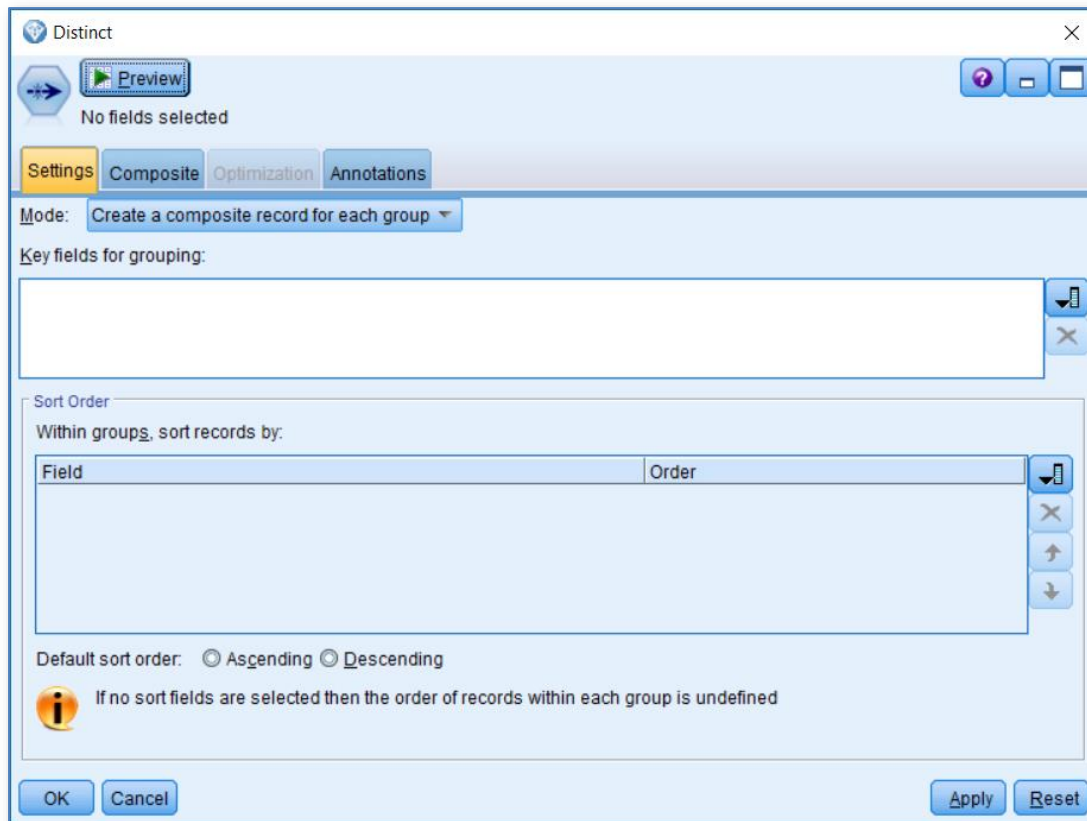


Figure 4.3 The default settings within the Distinct node

The first thing to note about this node is that it has three modes:

- **Create a composite record for each group:** This is the default mode. When it is activated, the tab marked 'Composite' is available for use. This mode allows users to control which values from the duplicate records are to be included in the final selection. Suppose you have two records with the same customer number. Maybe you know that the demographic data that is initially collected for new customers is more reliable than values in the duplicated records. In which case, you could request that the demographic data is taken from the first record for each customer. On the other hand, you might know that every time a customer makes a purchase, the transaction count increases, so you might request that the duplicate record with the largest transaction count for the customer is used as this is likely to be most up to date value for that field.
- **Include only the first record in each group:** This mode drops all duplicate cases. Only the first occurrence of each duplicate record is taken.
- **Discard only the first record in each group:** This mode drops all the unique records and first record of each duplicate. It basically selects the duplicated records. It is useful for checking which records are the actual duplicated ones.

To begin with, we can check if any of the records are *complete duplicates*. That is where every single value has been duplicated. To do so, edit the Distinct node so that:

The mode is set to 'Discard only the first record in each group'

All of the fields are entered into the 'Key fields for grouping' box

Figure 4.4 shows the completed dialog.

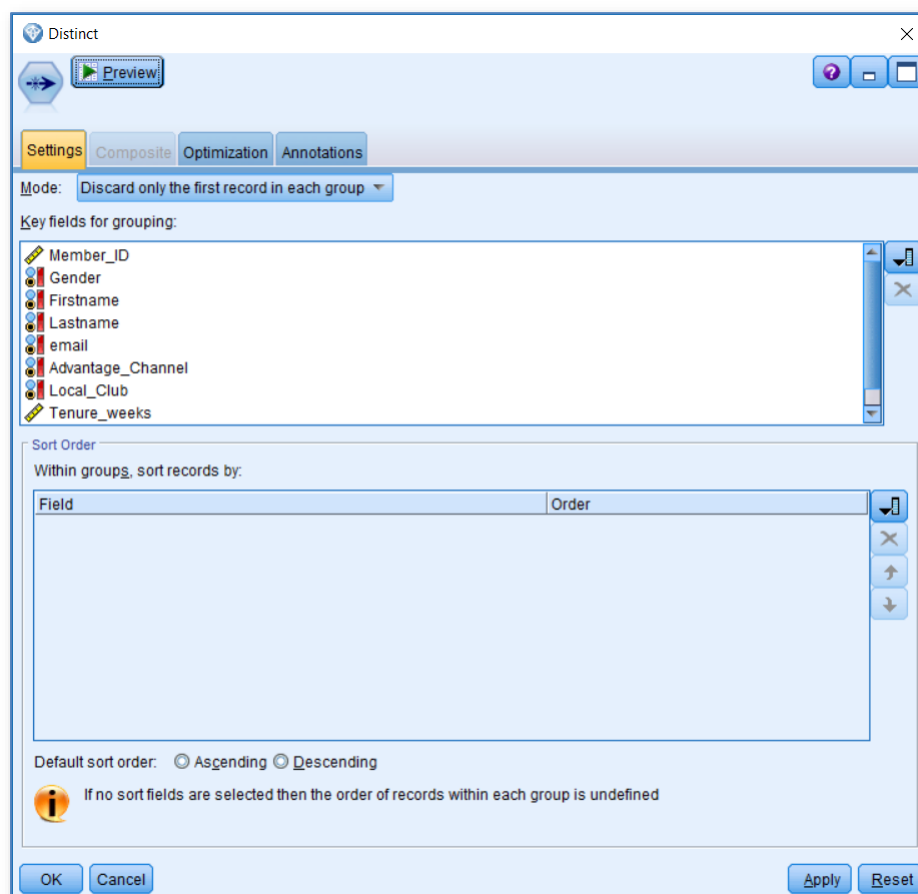


Figure 4.4 Using the Distinct node's 'Discard records' function to see if there are any completely identical records in the dataset

Remember that by using this function we are asking Modeler to select the duplicate records. To check if any meet this criterion, from within the Distinct control dialog, click:

Preview

As we can see in figure 4.5, no records are shown, so we know that there are no cases where *every value* has been duplicated.

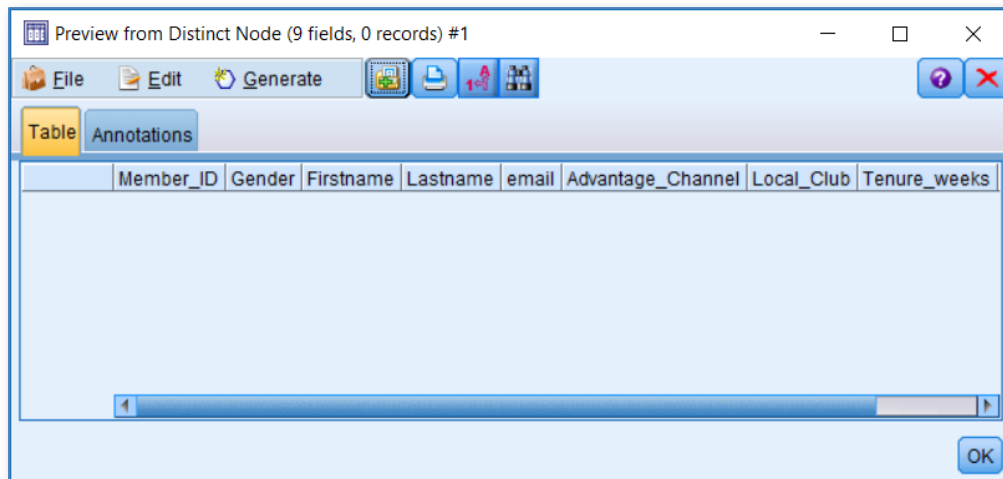


Figure 4.5 Preview showing no completely duplicated records found

By editing the Distinct node, we can now check to see if there are any records that have the same membership ID. To do so,

Remove all the fields from the 'Key fields for grouping' box

Figure 4.6 shows the completed dialog.

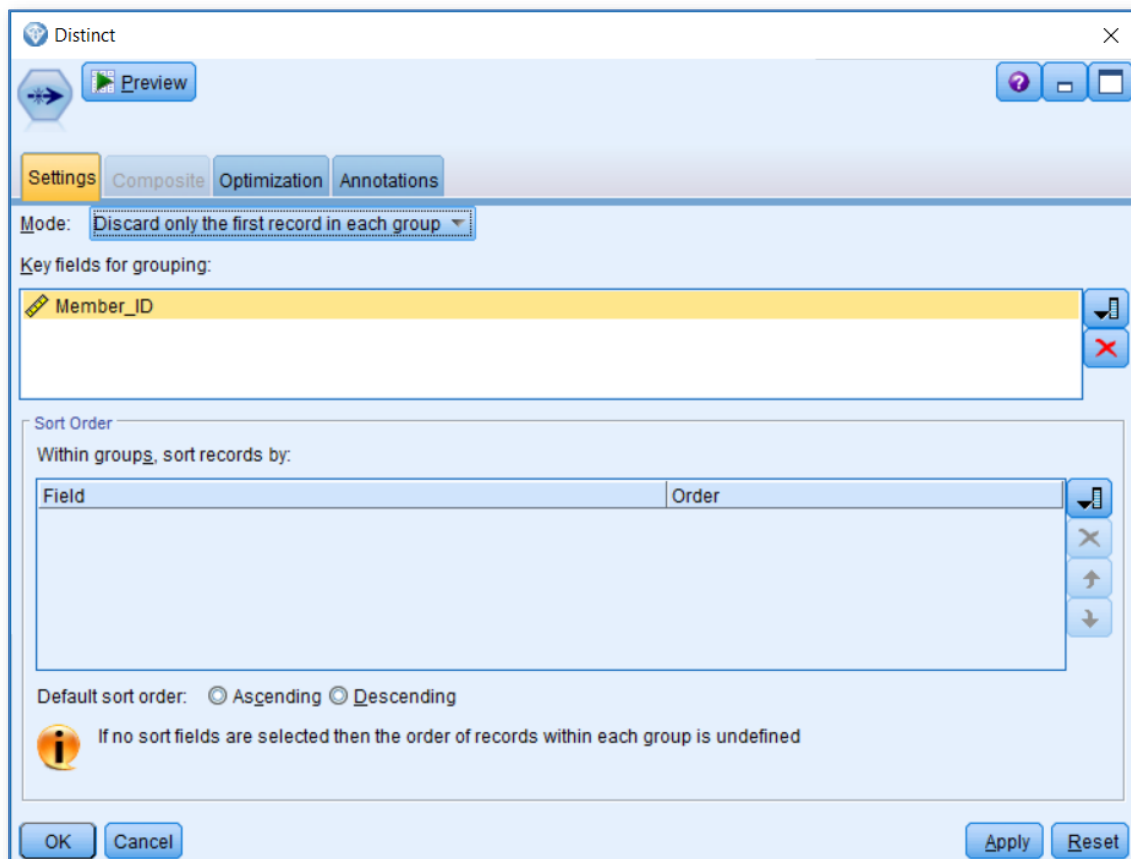


Figure 4.6 Checking for records that have duplicate ID values

Once, again, within the dialog, click:

Preview

As Figure 4.7 shows, the Distinct node has found three records with duplicate member ID values.

	Member_ID	Gender	Firstname	Lastname	email	Advantage_Channel	Local_Club	Tenure_weeks	Total_Spend
1	1091	Male	Lawrence	Patrick	LPatrick@Arcturus.org	Racquets	Edinburgh	156	4102
2	10609	Female	Isla	Norton	Islaisgreat@gmail.com	Workouts	London	148	2034
3	10967	Female	Jacqueline	Pope	JacquelinePope@aol.org	Workouts	London	146	133

Figure 4.7 Using the Distinct node to find records with duplicate ID values

We can now edit the Distinct node to only select the unique records (and drop the duplicate ones). Remember that to do so, we can either select the first record as it occurs in the dataset or we can use the Composite mode to select values from the duplicates according to specific criteria. Because the dataset doesn't contain a date field, we can't tell which of the duplicate records is the most recent. It seems reasonable to assume that the more recent records have larger values for the member's tenure and their total spend. Figure 4.8 shows how we can select records based on this by editing the Distinct node so that:

The mode is switched to 'Create a composite record for each group'

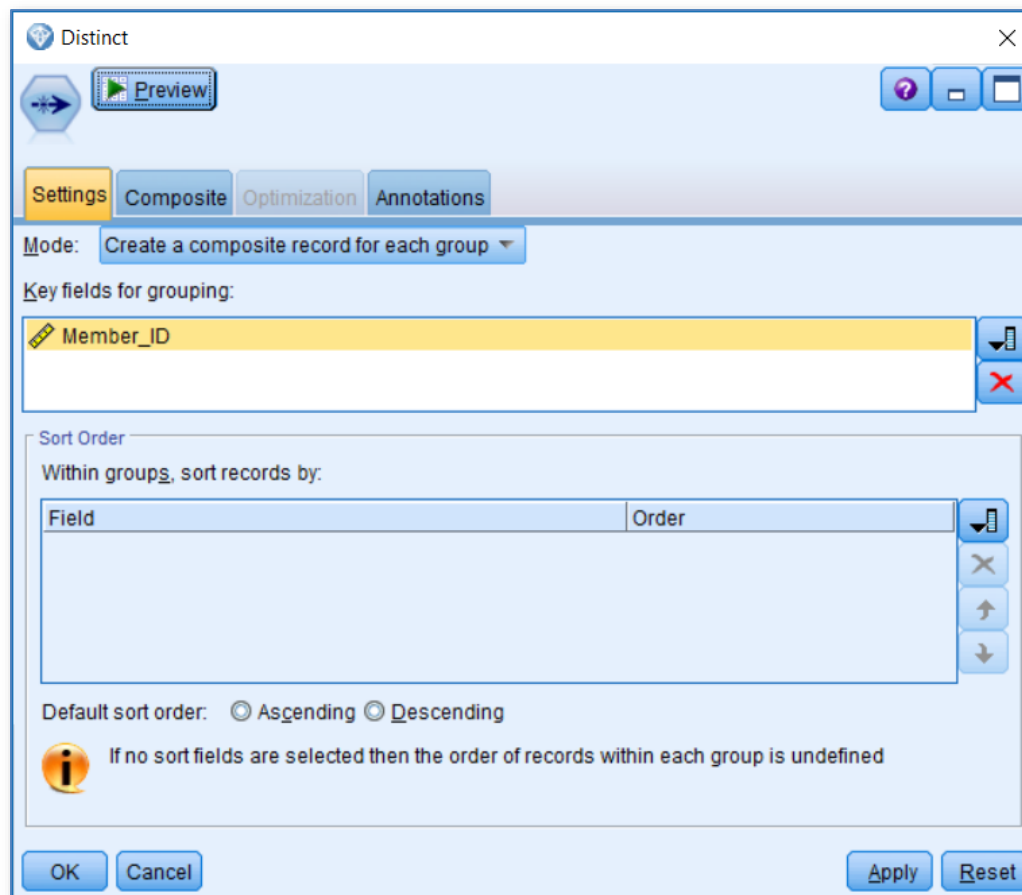


Figure 4.8 Distinct node switched to 'Composite' mode

To specify which values from the duplicate records will be used in the composite case, click the tab marked:

Composite

Edit the tab so that:

The maximum values for 'Tenure_weeks' and 'Total_Spend' are selected

By default, the 'Composite' mode creates a counter field that indicates how many records were used to create the composite. In this case it's not very useful so:

Uncheck the box marked 'Include record count in field'

Figure 4.9 shows the edited tab.

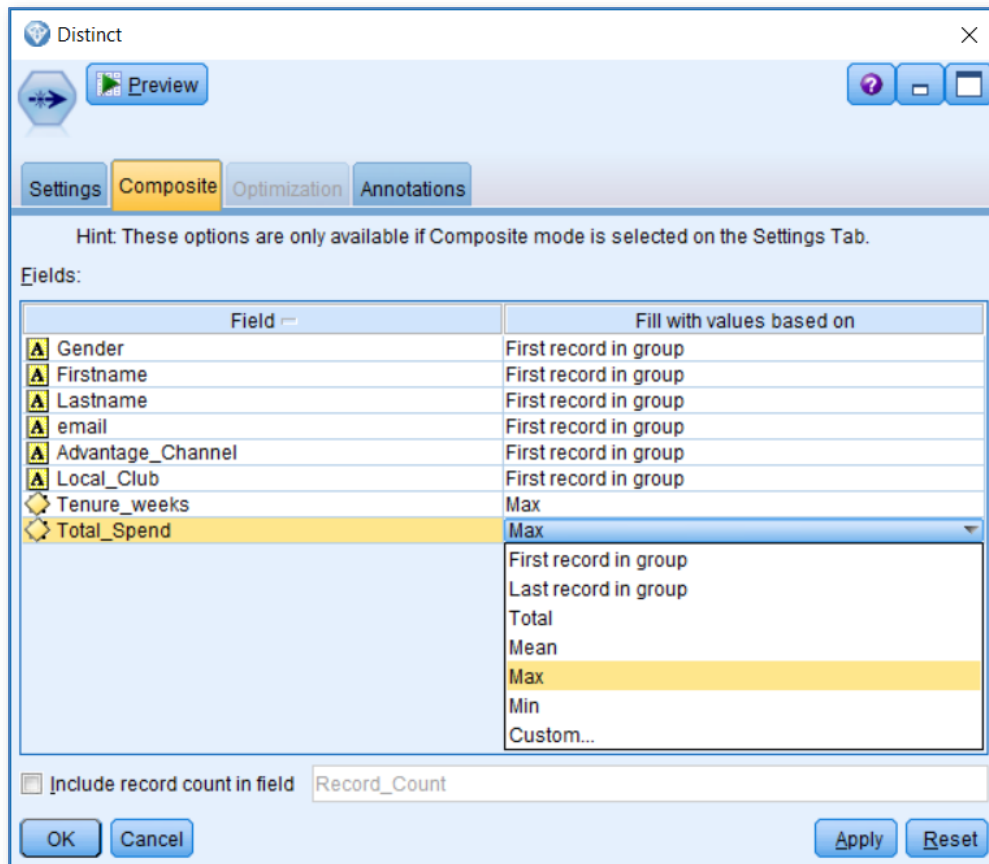


Figure 4.9 Editing the Composite tab in the Distinct node

To run the procedure, click:

Click 'OK' and attach a Table output node to Distinct node.

Figure 4.10 shows the resultant stream with an annotated Table node.



Figure 4.10 Distinct node using the 'Composite' mode to remove three records with duplicate ID values

It is possible that duplicate records still exist in the dataset however. These records might have different Member ID values but the same email address. To check if this is the case:

Copy and paste the Distinct node

Attach the pasted node to existing Distinct node

Edit the mode so that it switches back to 'Discard only the first record in each group'

Swap the field 'Member_ID' for the field 'email' within the 'Key fields for grouping' box

Figure 4.11 shows the resultant dialog.

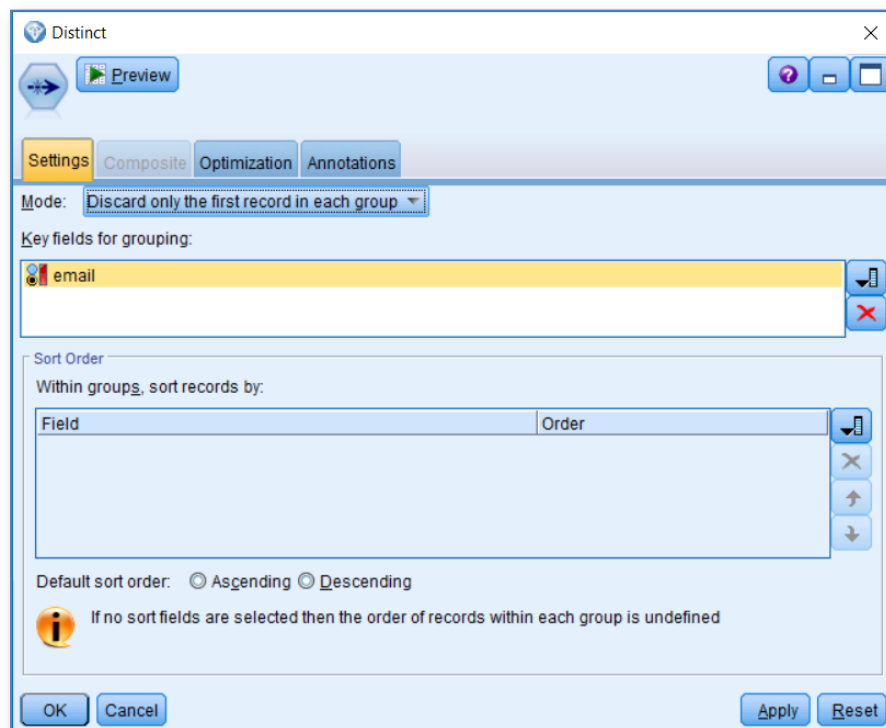


Figure 4.11 Checking for records with duplicate email addresses

To test if any records with duplicate email addresses exist, click:

Preview

Figure 4.12 shows the result.

	Member_ID	Gender	Firstname	Lastna...	email	Advantage_Channel	Local_Club	Tenure_weeks	Total_Spend
1	103859	Male	Bernardo	Clippings	barnyard@SVE.com	Racquets	London	163	2111
2	103877	Male	JM	LeGrand	leGrand@orange.com	Racquets	London	70	511

Figure 4.12 Distinct Node detecting two records with duplicate email addresses

As we can see, there are in fact two records with duplicate email addresses. These records of course will have different membership ID values, so we need to choose which of the duplicate records we wish to keep, or which *values* of the duplicate records we wish to retain to create a unique composite record. In this case, we can use the same criteria as before and simply switch the mode back to:

Create a composite record for each group

The same settings in the Composite tab will be applied here as in the previous example and we can attach a new Table node to the second distinct node and execute it. Figure 4.13 shows the updated stream.

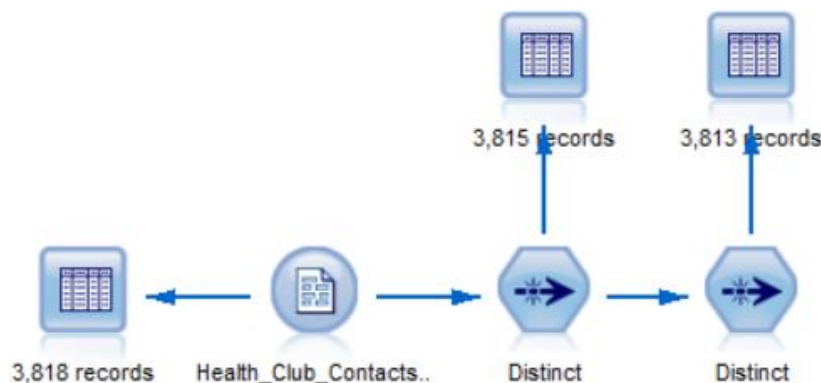


Figure 4.13 Distinct Node detecting and removing three records with duplicate membership ID values and a further two records with duplicate email addresses

The Aggregate Node



Data aggregation is a common data preparation task that many different applications can perform. Aggregation is a way to reduce multiple rows of data so that they are summarised by each value within a grouping field. We could use aggregation to change a file

so that instead of each row representing the individual items that a group of customers purchased over a year when visiting a webstore, they now represent the average amount of money spent by each customer and the total amount of purchases they made. In this scenario, the grouping field would be something like the customer id or email address. Aggregation therefore represents a way to simplify the data file and ensure that we end up with only one row per customer. We should note however that aggregation is best performed on cleaned data as missing values in any of the variables that are to be summarised (e.g. spend) or duplicates in the grouping field itself (e.g. the customer id) can cause problems with our downstream data preparation and analysis procedures.

To demonstrate the aggregation procedure in Modeler, we will import data from containing member transactions. Within the same stream, use a Var. File node to read the following pipe-separated file from within the Section 4 folder:

Health_Club_Transactions_Sample.txt

Transactional files are typical candidates for aggregation as they contain valuable information that we might wish to include in a model. However, they contain multiple rows per customer (and sometimes multiple rows per customer 'basket'). Figure 4.14 shows the edited source node for this data file and a preview of the data rows (note that the fields in the text-based data set are separated by the pipe character ('|')).

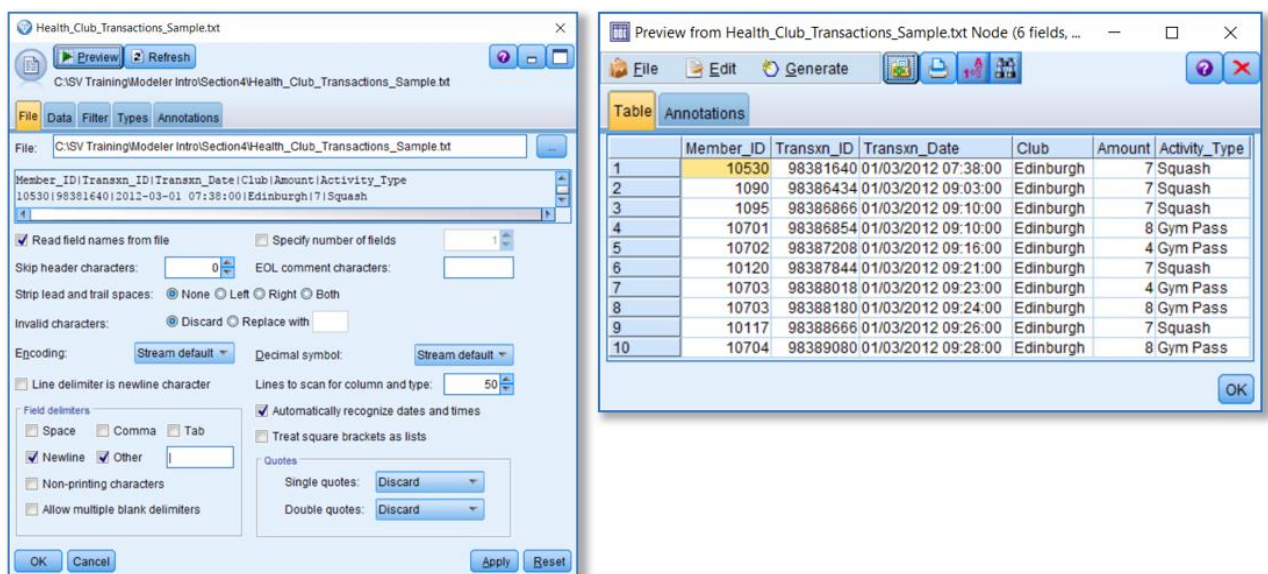


Figure 4.14 Editing the Var. file source node to read the data file 'Health_Club_Transactions_Sample.txt'

As with the Distinct node, the Aggregate node can also be found in the Record Ops palette within Modeler.

Select the source node containing 'Health_Club_Transactions_Sample.txt'

From within the Record Ops palette, double click the Aggregate node

Figure 4.15 shows the Aggregate node joined to the source node containing the new text data file.

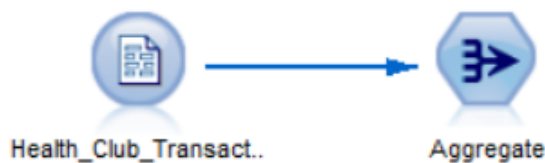


Figure 4.15 The Aggregate node attached to the source node containing 'Health_Club_Transactions_Sample.txt'

Right click on the Aggregate node and from the pop-up menu

Choose 'Edit' to view the aggregation controls

Figure 4.16 shows the main control dialog for the Aggregate node with numbered annotations.

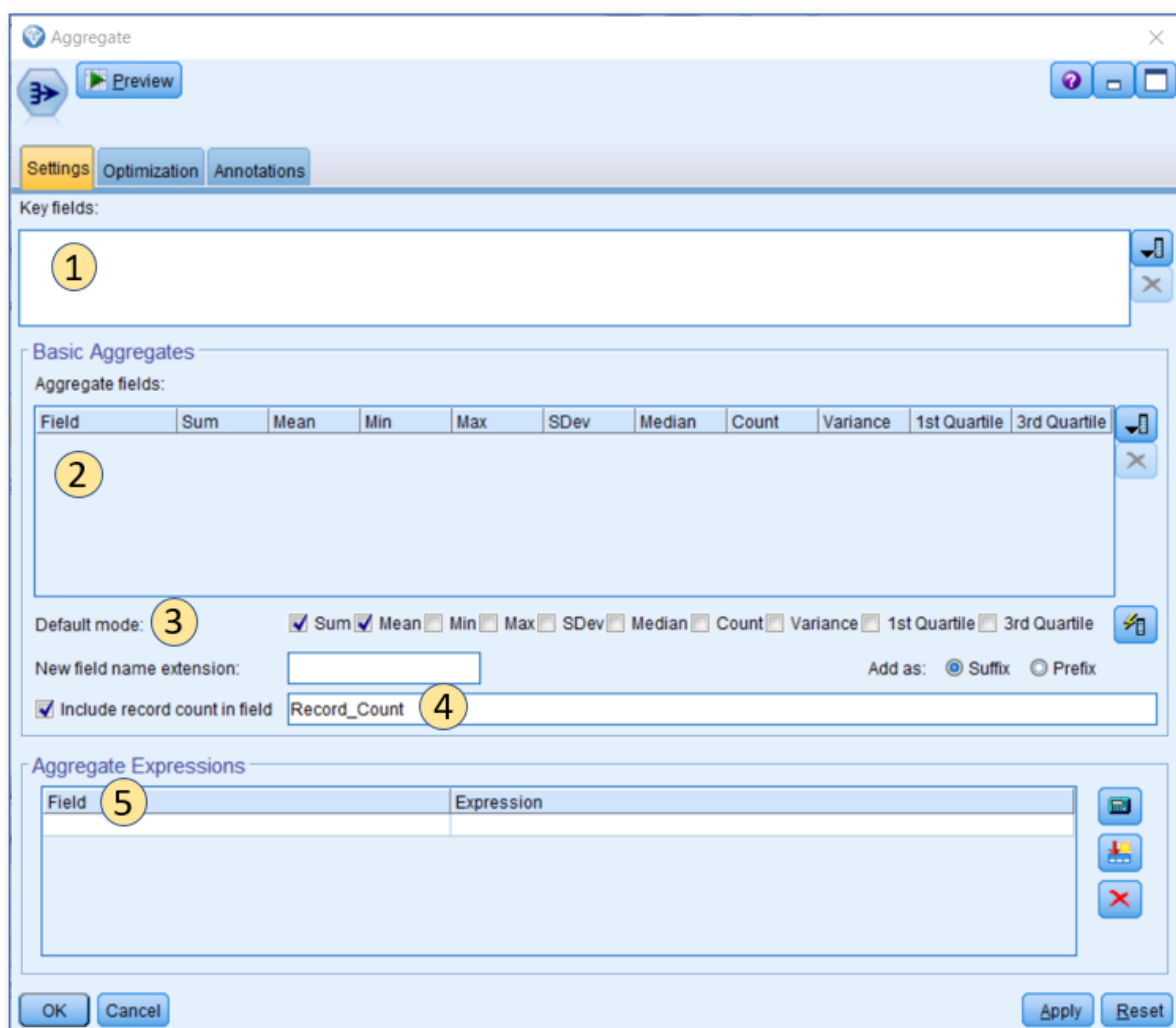


Figure 4.16 The various aggregation options within the Aggregate node

The Aggregation node allows us to summarise data files in a number of ways.

1. **Key Fields** – this is where we choose the grouping field(s) for the procedure. It allows us to specify what we are aggregating by. The key field in an aggregation procedure is normally some sort of identifier such as a customer id, a store number or an item code. It can also represent an event such as a transaction number or a datetime stamp. It's important to understand that we can have more than one key field and that the order in which we specify them creates a hierarchy. We might for example want each aggregated row to be defined by store number and the month that the data was collected. In which case if we had 10 stores with data collected over a one-year period, the final aggregated dataset would be comprised of 120 rows.
2. **Aggregate fields** – this where we specify the fields to be aggregated and the method(s) we choose to summarise them. It allows us to choose how we wish to aggregate the data. We might want to sum all the purchase values of each transaction for each customer. Additionally, we might want to take the average amount spent by each person. If the data set contained a date field, we could use the 'Min' and 'Max' functions to return the values for the date of the first and most recent purchases respectively.
3. **Default Mode** – There are many times when users want to be able to summarise multiple variables using more than one method. Rather than choosing a summary method for each variable individually, we can set the default methods here and simply edit the methods for fields where the defaults don't apply.
4. **Include record count in field** – Obviously the whole point of aggregation is to summarise multiple records by the levels of the key field. However, some ID values may only appear once in the data whereas others may appear multiple times. The 'Include record count in field' function creates an additional field that tells us how many records were aggregated for each value in the key field. If we are aggregating individual transactions for each customer, it effectively tells us how many transactions were aggregated to create the summary row. For this reason, we can edit the name of the field to reflect what is being aggregated.
5. **Aggregate Expressions** - This is a special function within the Aggregate node that enables the user to create their own summary measures when reading from a Database source node. The Aggregate Expression function uses an expression builder dialog to, for example, define measures that relate to relationships between fields such as the average amount of elapsed time between two date fields or the difference between the maximum value of one variable and the minimum value of another.

One of the limitations of aggregation procedures relates to categorical (set) fields. If we have a variable such as product purchased, there is no easy way to summarise the data. We can't for example take the average of the field. We might be able to record the minimum and maximum values, but this will probably not produce meaningful results. Nevertheless, this is a problem we can address using a different method as we shall see later.

In this example, we will aggregate the data by the membership id variable (Member_ID). To do so:

Choose the field 'Member_ID' to be entered into the Key Field box

Next, we need to choose the fields to be summarised. As noted earlier, the best fields for summarising tend to be those with continuous values.

Choose the field 'Amount' to be added to the 'Aggregate fields' section of the dialog

We can see that by adding this field, the default summary measures of 'Sum' and 'Mean' are applied. So this procedure will create an aggregated file with two fields recording the total and average amount spent by each member. However, we can also add a date variable to be summarised.

Choose the field 'Transxn_Date' to be added to the 'Aggregate fields' section of the dialog

Modeler recognises that this field type is a Datetime so the default measures (Sum and Mean) are greyed out. Instead we can ask that the procedure records the first transaction date for each member.

Check the box under the column marked 'Min' for the field 'Transxn_Date'

Finally, we can see that the procedure will create a counter variable recording how many records were aggregated for each member. As these individual records represent the actual transactions we can edit the column name to reflect this.

In the 'Include record count in field' section change the name of the field from 'Record_Count' to 'Transactions'

Figure 4.17 shows the completed dialog for the Aggregate procedure.

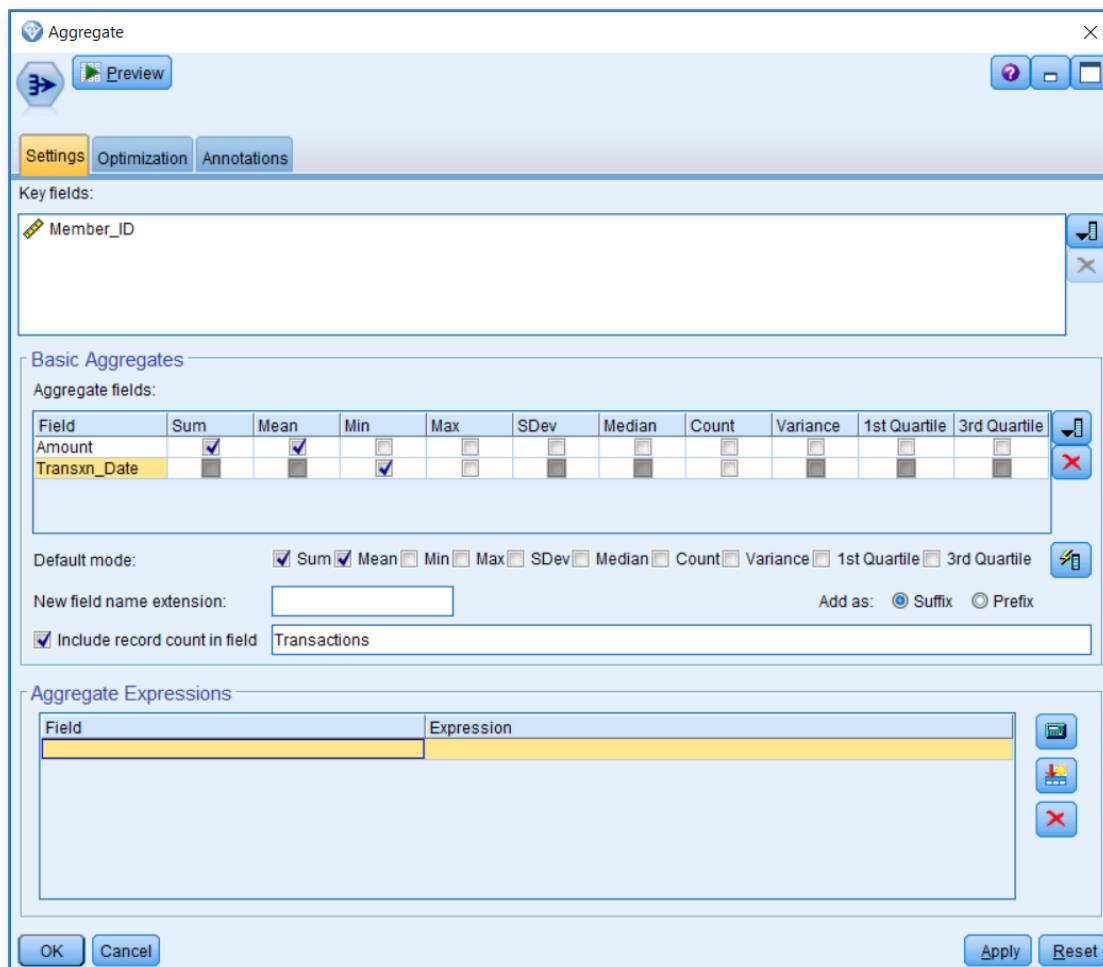


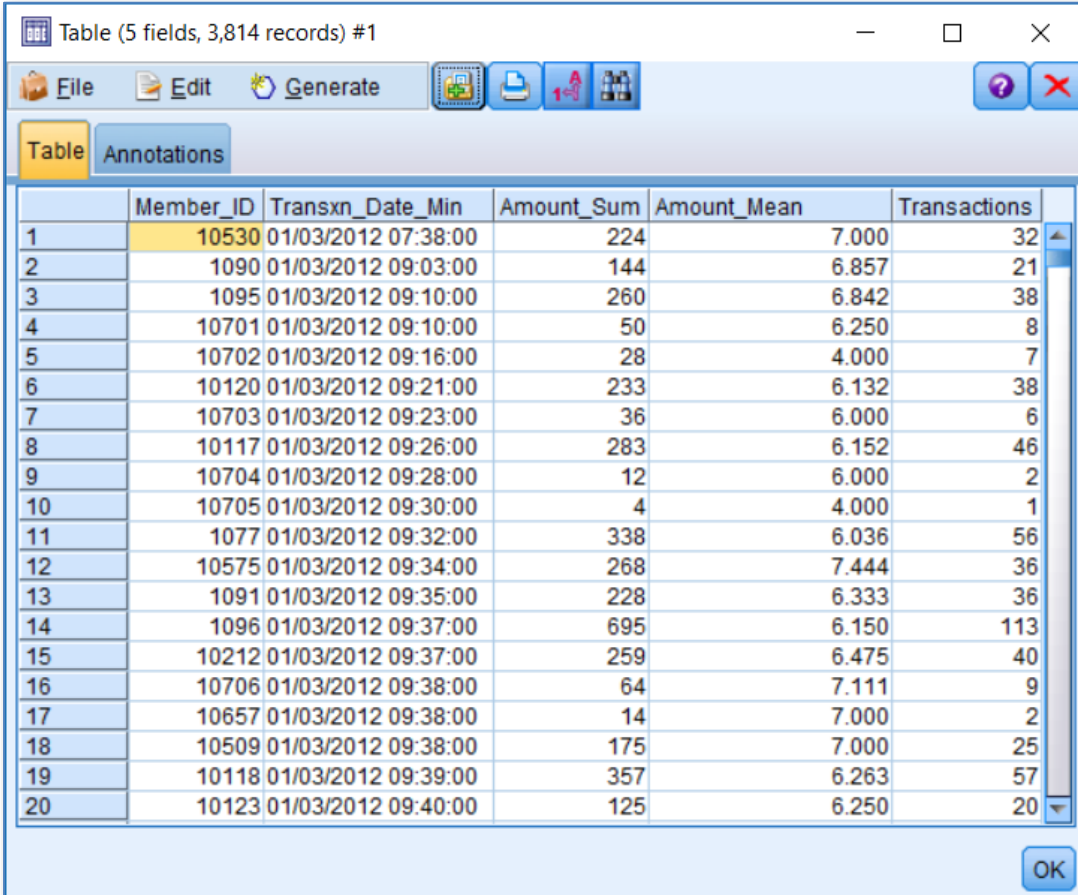
Figure 4.17 Aggregating the data by membership id to create three summary variables

To see the effects of the aggregate procedure:

Click OK and add a Table node to the Aggregate node

Execute the stream with the Aggregate node

Figure 4.18 shows the aggregated data in the table output. We can see in the screenshot that the field 'Transactions' has been added to the last column position showing how many actual transactions each member made (and therefore how many rows were aggregated to the create the summary values shown). The summary measures themselves are indicated by the suffixes '_Sum' and '_Mean' respectively to the 'Amount' variable.



	Member_ID	Transxn_Date_Min	Amount_Sum	Amount_Mean	Transactions
1	10530	01/03/2012 07:38:00	224	7.000	32
2	1090	01/03/2012 09:03:00	144	6.857	21
3	1095	01/03/2012 09:10:00	260	6.842	38
4	10701	01/03/2012 09:10:00	50	6.250	8
5	10702	01/03/2012 09:16:00	28	4.000	7
6	10120	01/03/2012 09:21:00	233	6.132	38
7	10703	01/03/2012 09:23:00	36	6.000	6
8	10117	01/03/2012 09:26:00	283	6.152	46
9	10704	01/03/2012 09:28:00	12	6.000	2
10	10705	01/03/2012 09:30:00	4	4.000	1
11	1077	01/03/2012 09:32:00	338	6.036	56
12	10575	01/03/2012 09:34:00	268	7.444	36
13	1091	01/03/2012 09:35:00	228	6.333	36
14	1096	01/03/2012 09:37:00	695	6.150	113
15	10212	01/03/2012 09:37:00	259	6.475	40
16	10706	01/03/2012 09:38:00	64	7.111	9
17	10657	01/03/2012 09:38:00	14	7.000	2
18	10509	01/03/2012 09:38:00	175	7.000	25
19	10118	01/03/2012 09:39:00	357	6.263	57
20	10123	01/03/2012 09:40:00	125	6.250	20

Figure 4.18 Aggregated data from created using the Aggregate node based on the 'Health_Club_Transactions_Sample.txt' data set

The original dataset was comprised of 19,827 cases. After aggregation, the number of records is reduced to 3,814. The Aggregate node is a powerful and flexible tool for creating summarised data files, but it also allows us to create new datasets where the unit of analysis may be more appropriate for modelling purposes. However, Modeler has another key procedure that helps users to aggregate data containing categorical variables.

The Set to Flag Node



Within Modeler, categorical variables are sometimes referred to as set fields. With this in mind, the 'Set to Flag' node is designed to derive flag fields that represent the individual categories within a set field. This is extremely useful because although we may not be able to use a statistical summary measure to summarise which product categories a customer has purchased from over a period of time, we can at least create a series of flag indicators that tell us whether or not they have made purchases within each category.

The Set to Flag node is found within the Field Ops palette of Modeler. We can use the procedure to create flag fields for each of the activity types that the health club members have signed up for in the transactions data set. To do so,

Select the source node containing 'Health_Club_Transactions_Sample.txt'

From within the Field Ops palette, double click the Set to Flag node

Figure 4.19 shows the node added to the stream and connected to the source node.

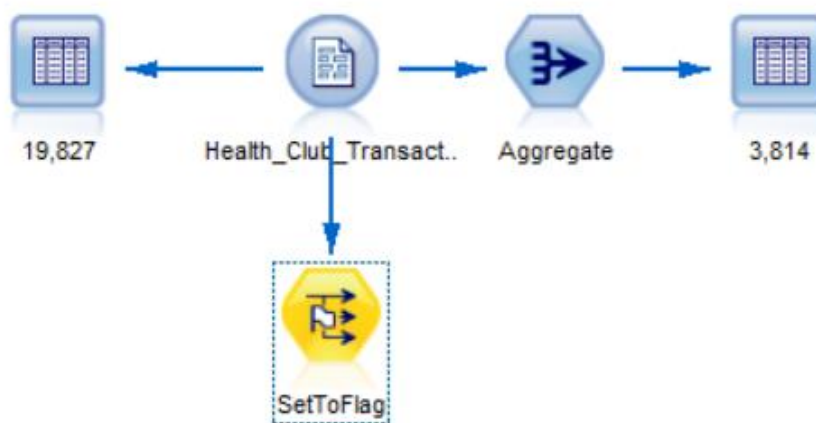


Figure 4.19 The Set to Flag node is attached to the previous source node

Let's take a look at how the procedure works.

Right click on the Set to Flag node and from the pop-up menu choose 'Edit'

Figure 4.20 shows an annotated image of the main control dialog for the Set to Flag node.

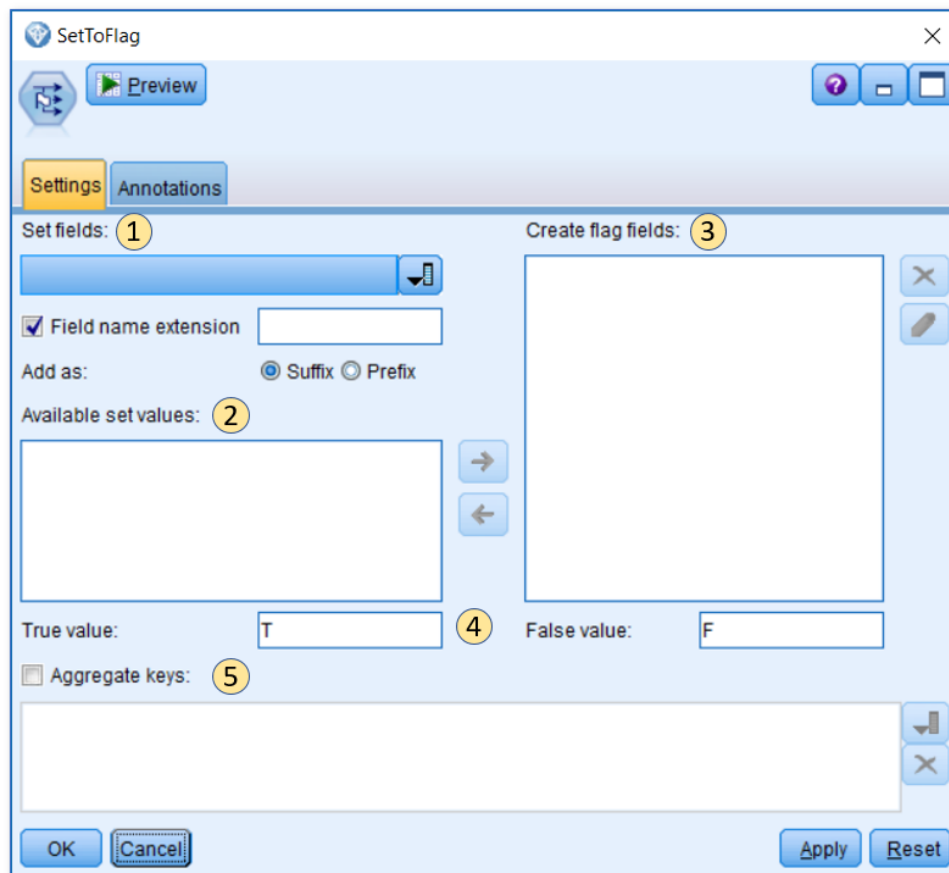


Figure 4.20 The controls for the Set to Flag Node

1. **Set fields** – Here we specify the categorical field(s) that we wish to use to create Flag variables from. Interestingly, this is one of the procedures where the data needs to be fully instantiated in the Type tab/node. If the data are not fully instantiated, no categorical fields are shown in the drop-down menu.
2. **Available set Values** – If the categorical field is fully instantiated, then the node will look at the values of the categories within the type node and list them here. Perhaps some of the values are not relevant for deriving the flag fields, in which case they can be ignored by the rest of the procedure.
3. **Create flag fields** – Here we can view the fields that the selected values within the categorical variable will create.
4. **True Value/False Value** – As you know, Flag fields only consist of two values. Here we can specify the flag value that indicates a particular category occurred within the keyed group or identifier. The default setting uses 'T' to indicate 'True' and 'F' for 'False'. However, some analysts prefer to use other values, such as the numeric codes '1' and '0' as outcome indicators, so the procedure allows us to edit them here.

5. **Aggregate keys** – This is a critical requirement in the procedure if we wish to aggregate the data. Just as we saw in the Aggregate node, specifying a Key field enables the analyst to create one row per value for each level of the grouping or identifier field. This is where one would normally add a variable like 'Customer number' to the procedure. To demonstrate the procedure:

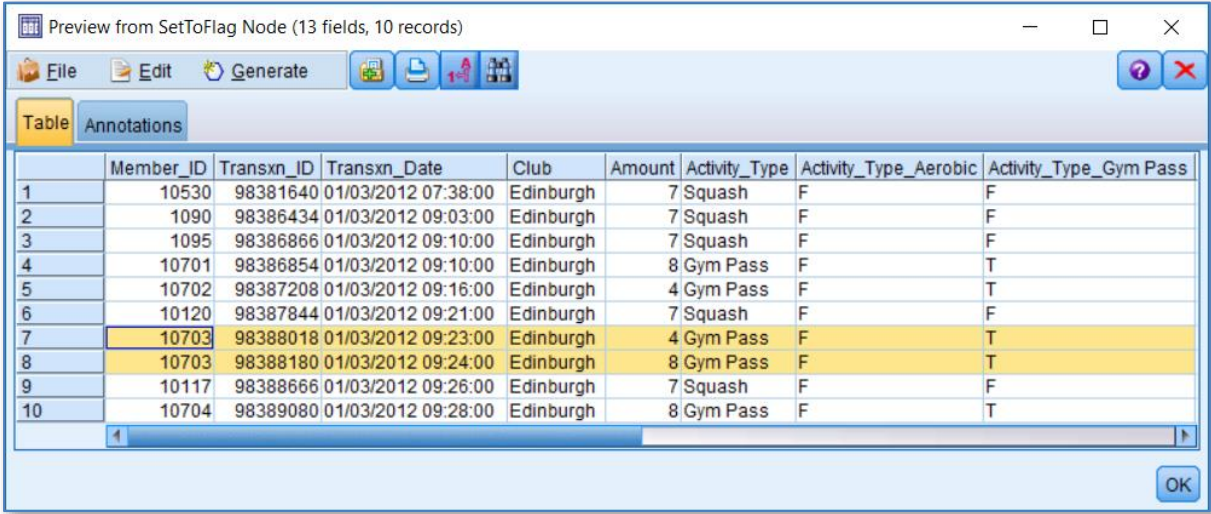
Check the source node make sure that the data is fully instantiated. If it isn't, click 'Read Values' in the Type tab

Return to the edited Set to Flag node and from the within the 'Set fields' drop-down menu choose the field 'Activity_Type'

You may notice that the instant this field is chosen, the 'Available set values' list box is populated with all the categories of the variable 'Activity_Type'.

Highlight and select all the categories within the 'Available set values' list box and click the arrow to send them to the 'Create flag fields' list box

At this point the data remains unaggregated. We can see this by clicking the Preview button. As figure 4.21 shows, although new flag fields have been created, the unaggregated data contains two occurrences of the membership ID '10703'.



Preview from SetToFlag Node (13 fields, 10 records)

	Member_ID	Transxn_ID	Transxn_Date	Club	Amount	Activity_Type	Activity_Type_Aerobic	Activity_Type_Gym Pass
1	10530	98381640	01/03/2012 07:38:00	Edinburgh	7 Squash	F		F
2	1090	98386434	01/03/2012 09:03:00	Edinburgh	7 Squash	F		F
3	1095	98386866	01/03/2012 09:10:00	Edinburgh	7 Squash	F		F
4	10701	98386854	01/03/2012 09:10:00	Edinburgh	8 Gym Pass	F		T
5	10702	98387208	01/03/2012 09:16:00	Edinburgh	4 Gym Pass	F		T
6	10120	98387844	01/03/2012 09:21:00	Edinburgh	7 Squash	F		F
7	10703	98388018	01/03/2012 09:23:00	Edinburgh	4 Gym Pass	F		T
8	10703	98388180	01/03/2012 09:24:00	Edinburgh	8 Gym Pass	F		T
9	10117	98388666	01/03/2012 09:26:00	Edinburgh	7 Squash	F		F
10	10704	98389080	01/03/2012 09:28:00	Edinburgh	8 Gym Pass	F		T

Figure 4.21 Preview of the results of a Set to flag node when no Aggregate key field has been specified

From within the Set to Flag node control dialog,

Check the box marked 'Aggregate keys' and select the variable 'Member_ID' from the drop down variable list on the right-hand side

Figure 4.22 shows the completed Set to flag node.

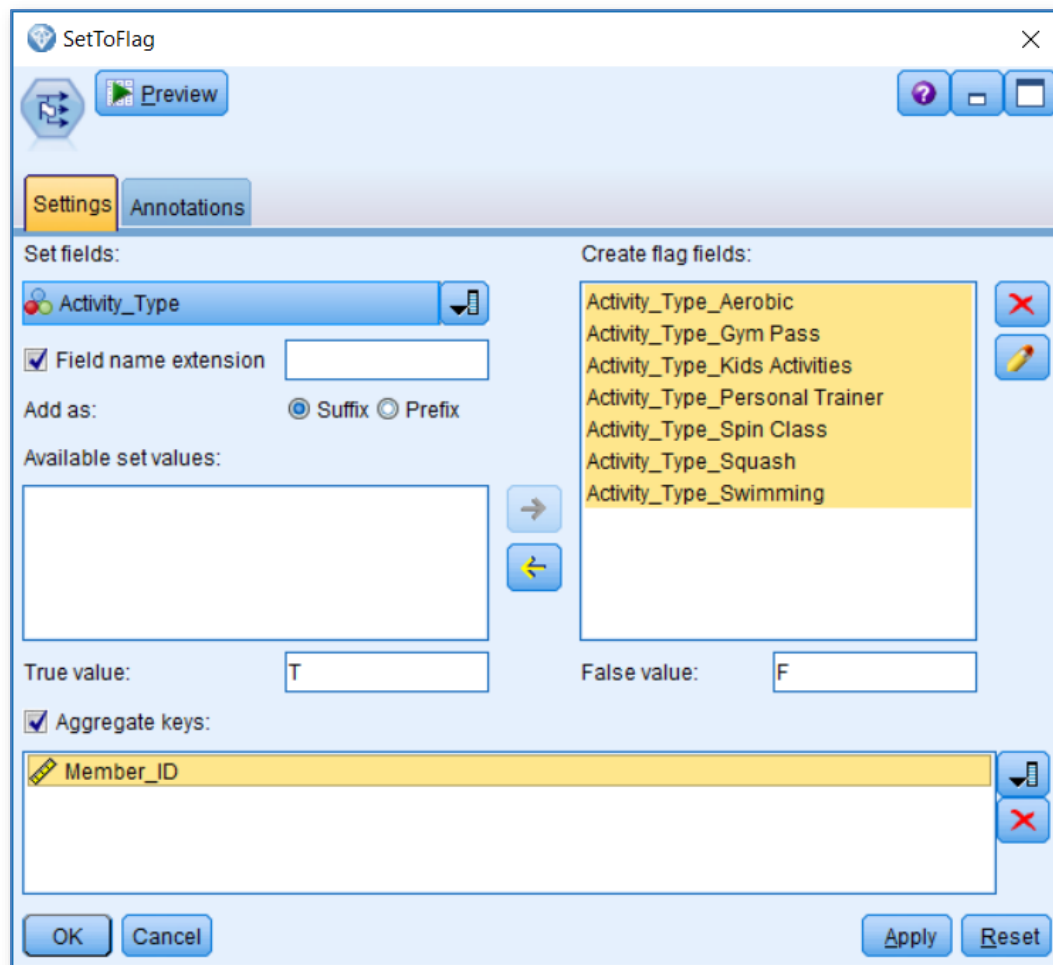


Figure 4.22 The completed Set to flag node

Within the dialog, click 'OK' and attach a table node to the completed Set to Flag node

Figure 4.23 shows the completed stream.

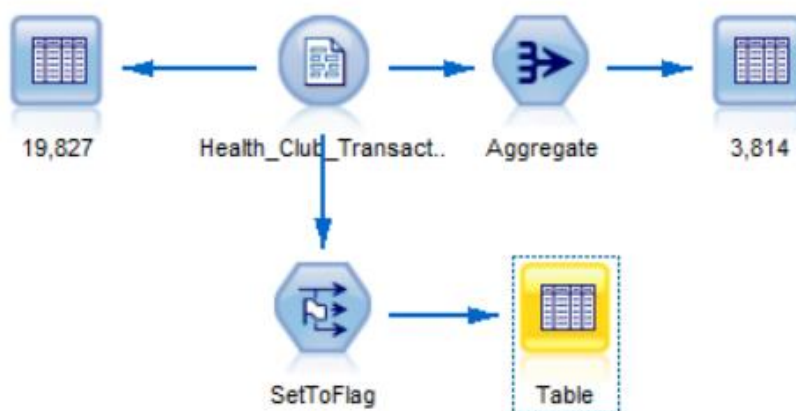


Figure 4.23 The completed stream containing the Set to flag node

To see the effects of the Set to flag procedure:

Run the stream branch containing the Set to Flag node

Figure 4.24 shows the output data from the node.

	Member_ID	Activity_Type_Aerobic	Activity_Type_Gym Pass	Activity_Type_Kids Activities	Activity_Type_Personal Trainer	Activity_Type_Spin Class	Activity_Type_Squash	Activity_Type_Swimming
1	10530	F	F	F	F	F	T	F
2	1090	F	F	F	F	F	T	F
3	1095	F	F	F	F	F	T	F
4	10701	F	T	F	F	F	T	F
5	10702	F	T	F	F	F	F	F
6	10120	F	F	F	F	F	T	F
7	10703	F	T	F	F	F	F	F
8	10117	F	F	F	F	F	T	F
9	10704	F	T	F	F	F	F	F
10	10705	F	T	F	F	F	F	F
11	1077	F	F	F	F	F	T	F
12	10575	F	T	F	F	F	F	F
13	1091	F	F	F	F	F	T	F
14	1096	F	F	F	F	F	T	F
15	10212	F	F	F	F	F	T	F
16	10706	F	T	F	F	F	F	F
17	10657	F	F	F	F	F	T	F
18	10509	F	F	F	F	F	T	F
19	10118	F	F	F	F	F	T	F

Figure 4.24 The completed stream containing the Set to flag node

As we can see from the table node, we are again dealing with an aggregated dataset of 3,814 records. However only the newly-created flag fields are shown. Moreover, each time a 'T' character is displayed within a cell, it indicates that that particular activity type was chosen by the member in the corresponding row.

The Web Node



The Web node performs no functions related to restructuring data but is a very useful way for us to investigate the interrelationships between the flag fields that we created using the Set to Flag node. The Web node can be found within the Graphs palette of Modeler. Figure 4.25 shows the Web node attached to the Set to Flag node.

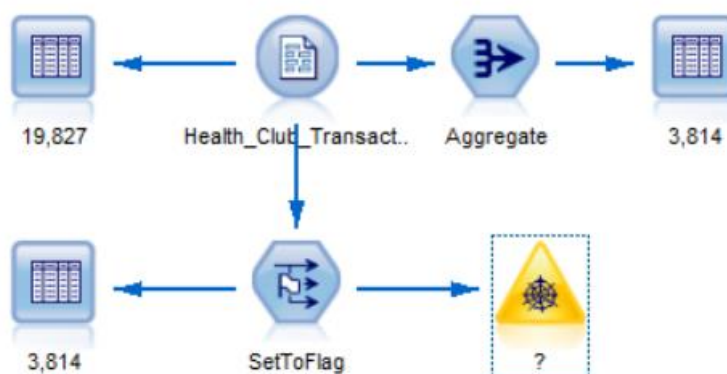


Figure 4.25 The Web node attached to the Set to flag node

If we edit the web node, we are able to select any categorical fields for inclusion in the plot. In this case, the flag fields from the Set to Flag node will suffice. Figure 4.26 shows the edited Web node with the newly created flag fields included in the field plot list. You may note that the box marked 'Show true flags' is checked, this simply tells the node to plot the relationships between the activity types that the members chose (ignoring the relationships between those activities they did not select).

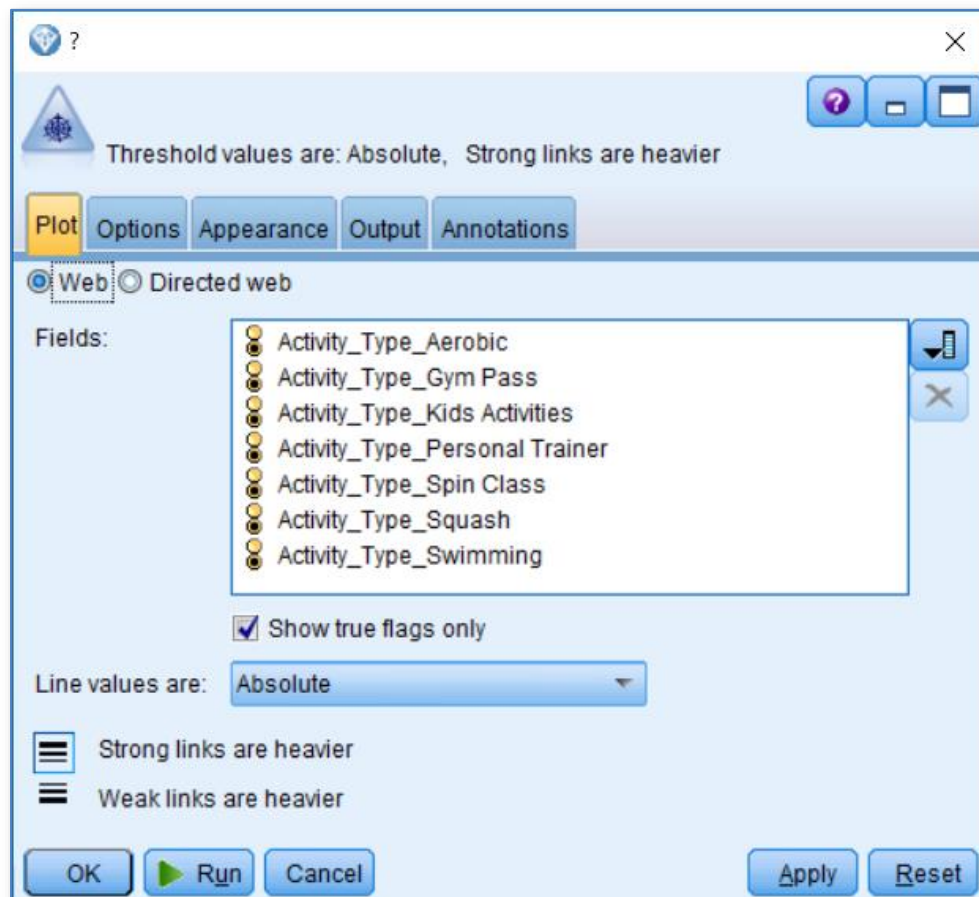


Figure 4.26 The main Web node dialog with the new flag fields added to the field plot list

To execute the plot, click:

Run

Figure 4.27 shows the initial output from the web plot (the chart has been edited to increase the clarity in print). You can see from the plot that each activity is connected to each other. Thicker lines indicate stronger connections between activities – for example, there are 402 records that chose 'Swimming' and 'Personal Trainer' activities but only 29 records that chose both 'Swimming' and 'Aerobic' activities.

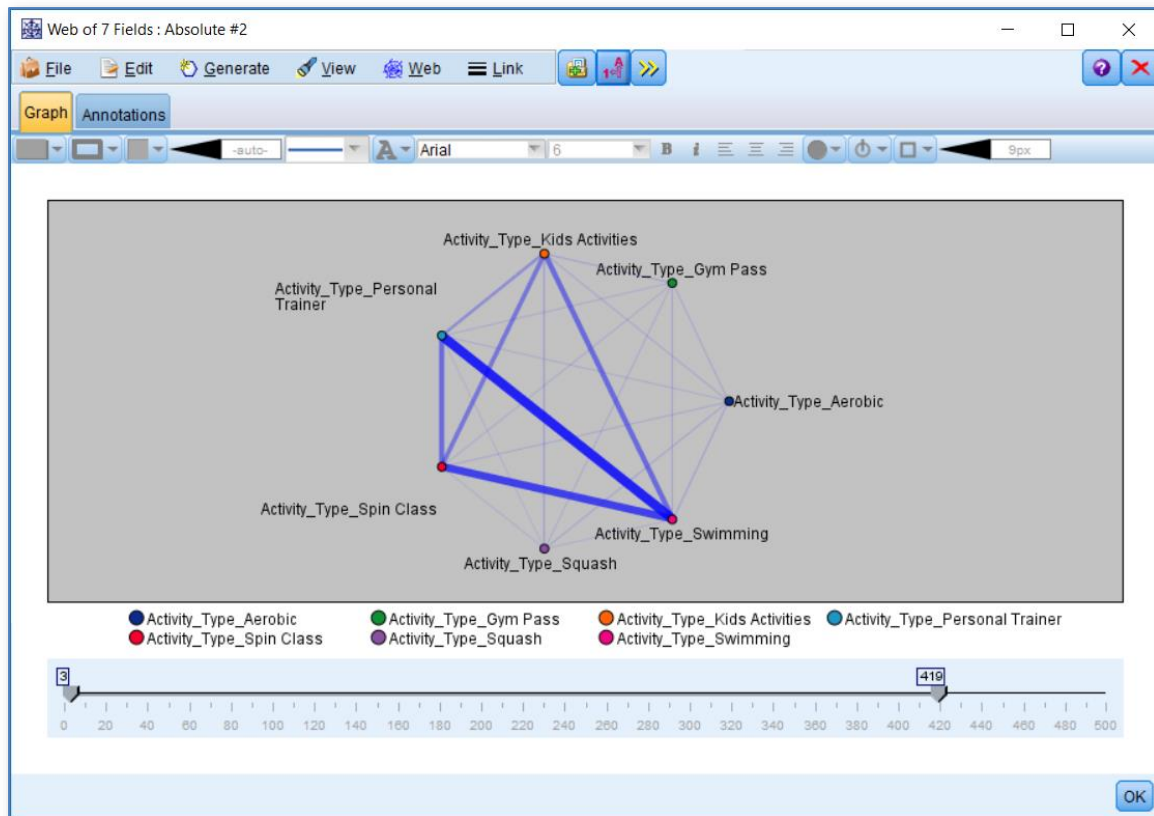


Figure 4.27 The Web plot showing the relationships between the activities – thicker lines represent stronger connections

One of the most compelling aspects of the Web plot is that it allows the user to explore these relationships interactively. To control the thresholds of the line thickness, users can access the web output controls by clicking the following button on the main toolbar:



The plot output is now expanded and displays the number of links between the activity types in a tabular format (see figure 4.28).

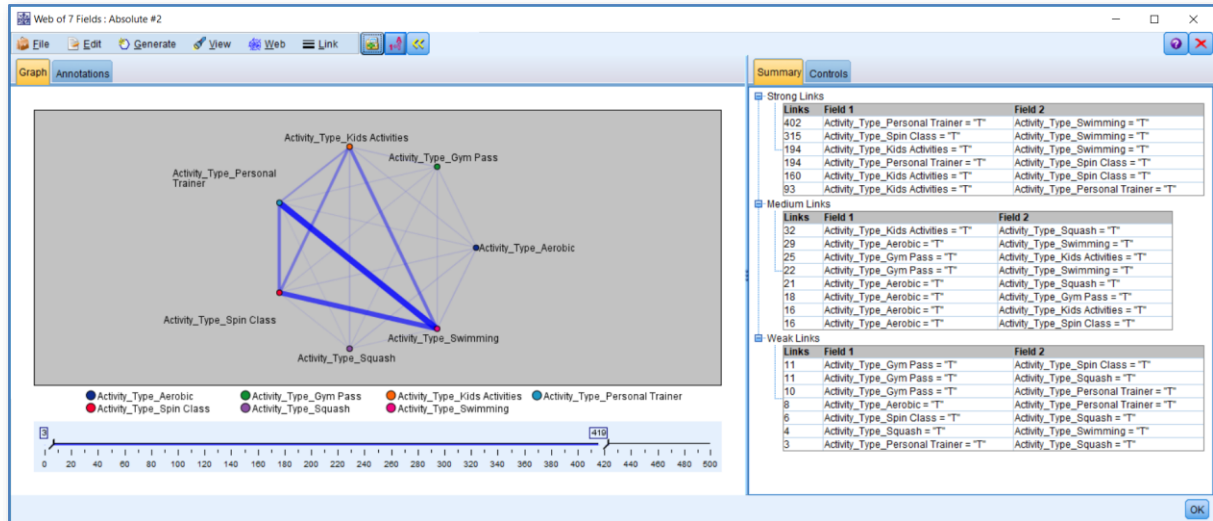


Figure 4.28 Accessing the web plot controls

To gain control over the display of the line thickness, click the tab marked:

Controls

Check the radio button marked:

Size shows strong/normal/weak

As figure 4.29 shows we can now use the slider controls to change the threshold points at which the lines display strong, medium or weak relationships between the activity types.

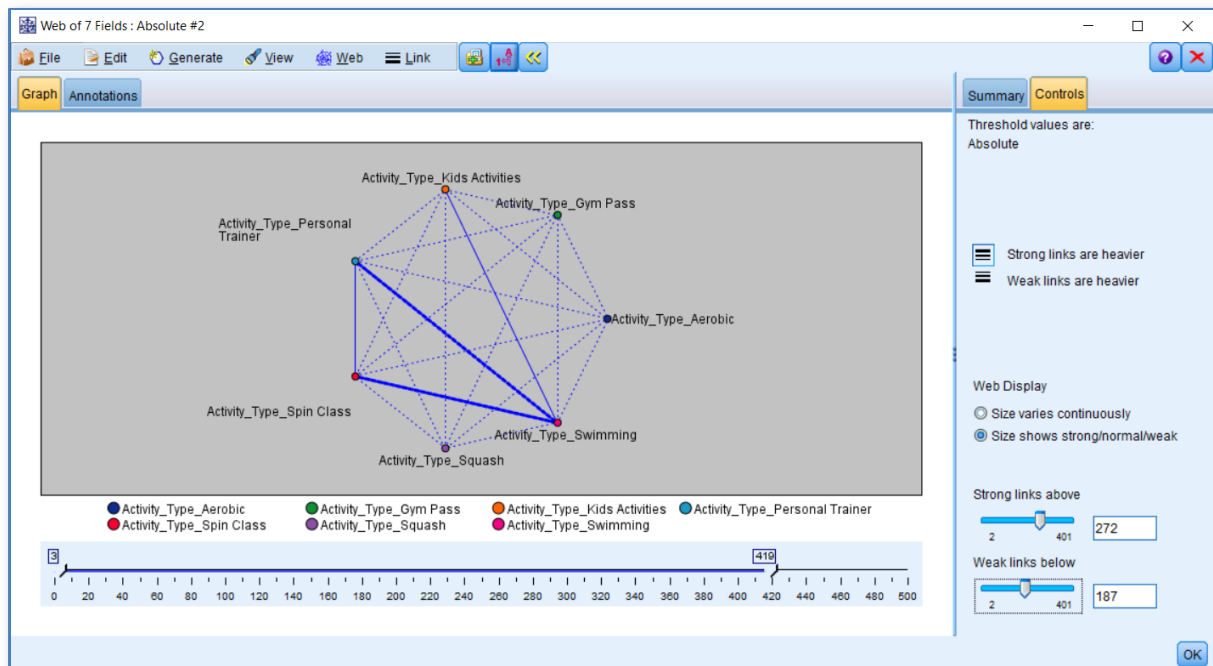


Figure 4.29 Changing the threshold values for line thickness

By setting the 'Strong' relationship threshold at 272, we can immediately see two strong relationships between the activity types 'Swimming' and 'Personal Trainer' and also between 'Swimming' and 'Spin Class'. Furthermore, we can see two medium strength relationships between the activity types 'Personal Trainer' and 'Spin Class' and between 'Swimming' and 'Kids' Activities'. These kinds of plots are very useful for spotting natural 'affinities' between products or services in consumer behaviour and can be used as the basis of a model to identify potential offers that could be made to consumers. In the next section we will continue the data preparation theme by looking at how we can create consolidated datasets using procedures for merging and appending data files.

Section 5: Merging Data



- The Append Node
- The Merge Node
- The Field Reorder Node

As we have seen in earlier sections, Modeler is designed to read multiple data files in multiple formats simultaneously. In the previous section, we looked at methods for restructuring the data in order to create the required *unit of analysis*. In this section, we will look at how we can *consolidate* the cleaned and transformed data. There are two main methods that data management and analysis platforms use to achieve this: merging and appending.

The Append Node

Appending data files is a process of concatenating the data to create a larger file with additional records. Append procedures often become necessary when joining data files that refer to different customer groups, regions or time periods. The Append node joins two or more files together by combining their respective records and matching the data on the basis that:

- The fields in the different files have the same names *or* occupy the same position
- The fields with the same names or position are the same type

Fortunately, the Append node in Modeler makes it easy to append multiple data files and to deal with issues of mismatched columns. To illustrate this, open the following Modeler stream from the Section 5 folder:

Section 5 Start.str

Figure 5.1 shows three source nodes near the top of the stream that have been configured to read the files we will use in the append demonstration. We can see from the source nodes that they represent three separate years of customer transactions (from 2012 to 2014). You may notice that each file is stored differently in terms of format or delimiter.

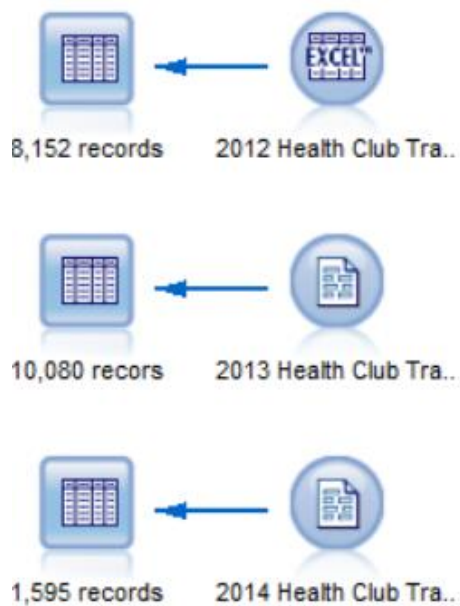


Figure 5.1 Source files representing three separate different years of customer transactions to be appended into a single data file

The Append node is stored in the Record Ops palette. We can add it to the stream and attach each of the source nodes to it.

Double click the Append node within the Record Ops palette

Join each Data Source node to the Append node

Note that Modeler detects the order in which the data sources are joined to the Append node. Figure 5.2 shows the source nodes joined to the added Append node.

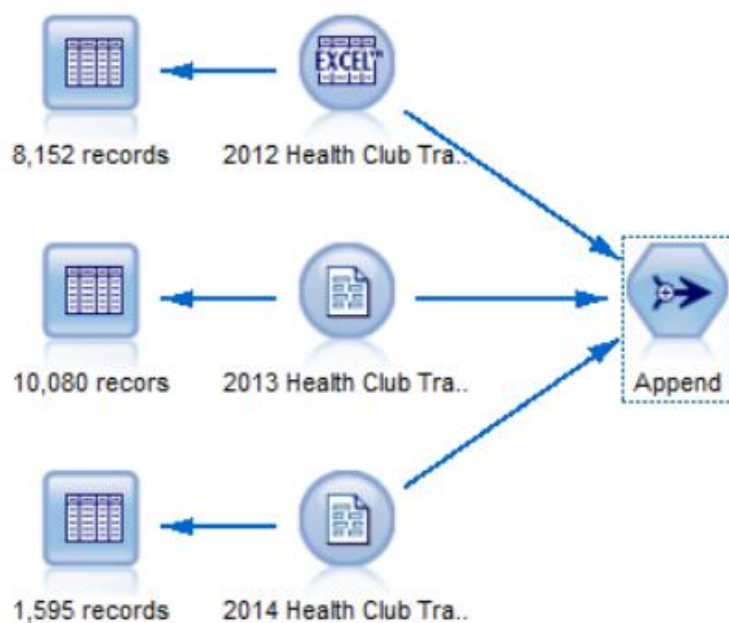


Figure 5.2 The Append node attached to three data source nodes

To see the available functionality within this node.

Edit the Append node

Figure 5.3 shows the default settings in the Append tab with numbered annotations.

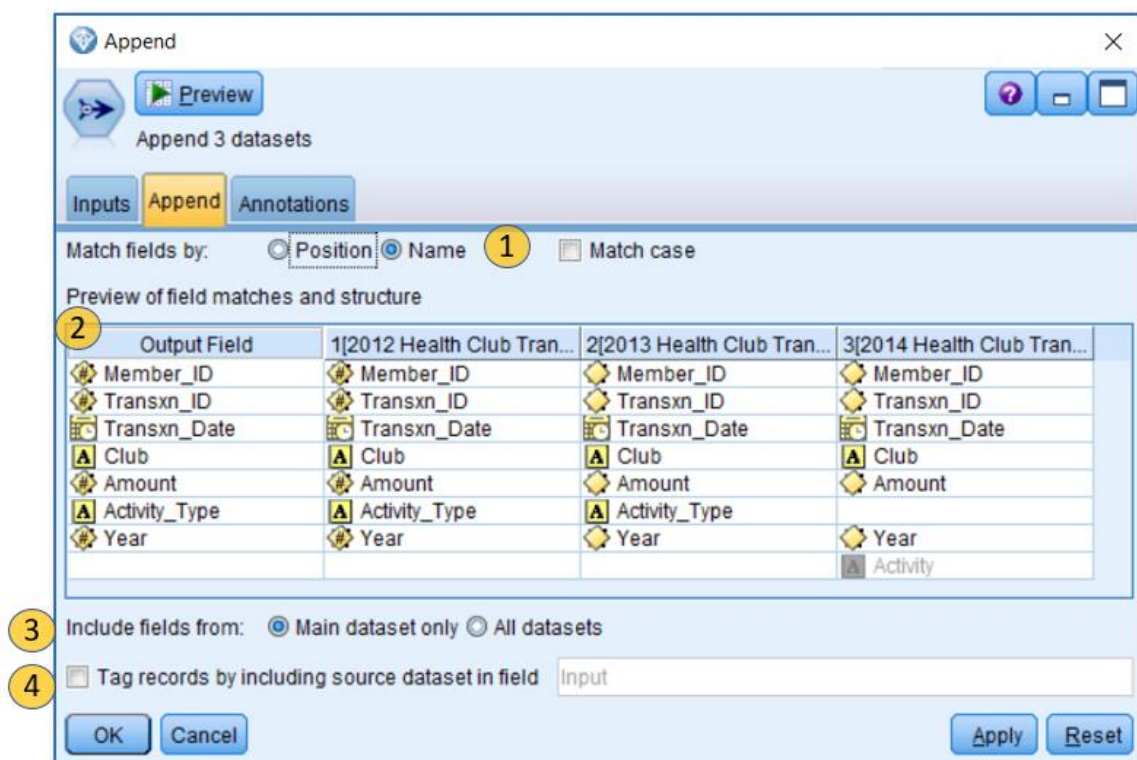


Figure 5.3 The edited Append node with numbered annotations

Initially, it looks like the node is referencing four files as opposed to three. This is because the Append tab within the node shows us information about the fields from the three input files but also shows us the information about the final appended data file the procedure will create.

1. **Match fields by mode:** Within the node we have the option to match the data files based on fields with the same name (and optionally the same character case) or the same position within the files. You can see that the field marked 'Activity' in the last column (from the 2014 file) is greyed out. This does not have the same name as the other two files where the column is called 'Activity_Type'.
2. **Preview of field matches and structure:** The column marked 'Output Field' shows us what fields will be displayed in the appended output file. Although the field 'Activity_Type' will be included, it will not contain values from the 2014 file.
3. **Include fields from Mode:** The 'Main dataset' refers to the first dataset joined to the Append node (this can be edited within the 'Inputs' tab). The main dataset is the

leading data source. The names and order of the fields within this dataset represent the template that the other appended files must adhere to.

4. **Tag records by option:** Finally, the Append tab offers us the opportunity to create a new field (called 'Input' by default) that creates a numbered indicator (in this case running from 1 to 3) showing which source file contributed each row in the appended output file.

As we can see in the example, there is a mismatch between the field names for the variable 'Activity_Type'. Perhaps we can resolve this issue by switching the 'Match fields by' mode. Within the Append tab, next to the 'Match fields by' label click the option marked:

Position

Figure 5.4 shows what happens when we try this option.

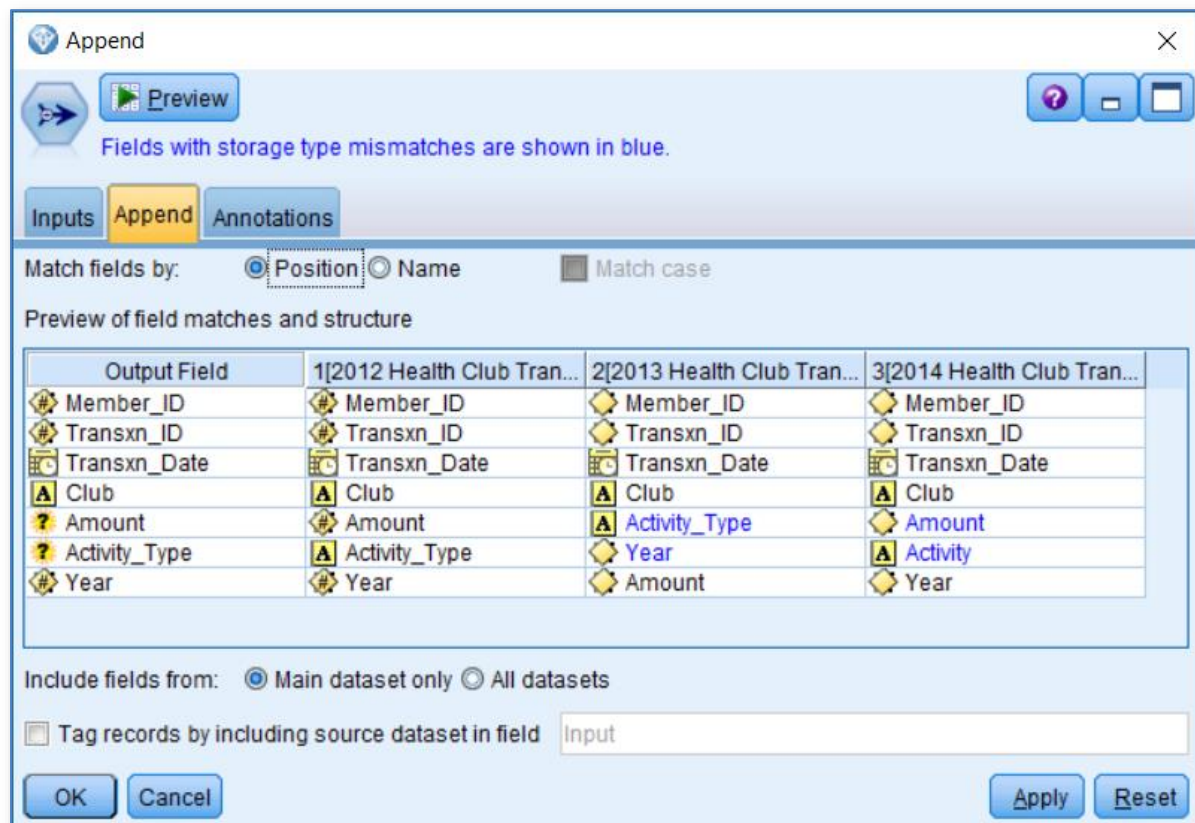


Figure 5.4 Changing the 'Match fields by' mode to 'Position' instead of 'Name'

Unfortunately, changing the field matching mode has not helped to resolve the issue. The reason is that not all the files have the same fields in the same position. We can see that the field 'Amount' in the 2013 data file appears in the last column position (whereas in the other files it is the third last column). Perhaps we could go back to the default setting and try another approach.

Within the dialog, click the button marked:

Reset

In section labelled, 'Include fields from', click the option marked:

All datasets

Figure 5.5 shows the results of changing this option.

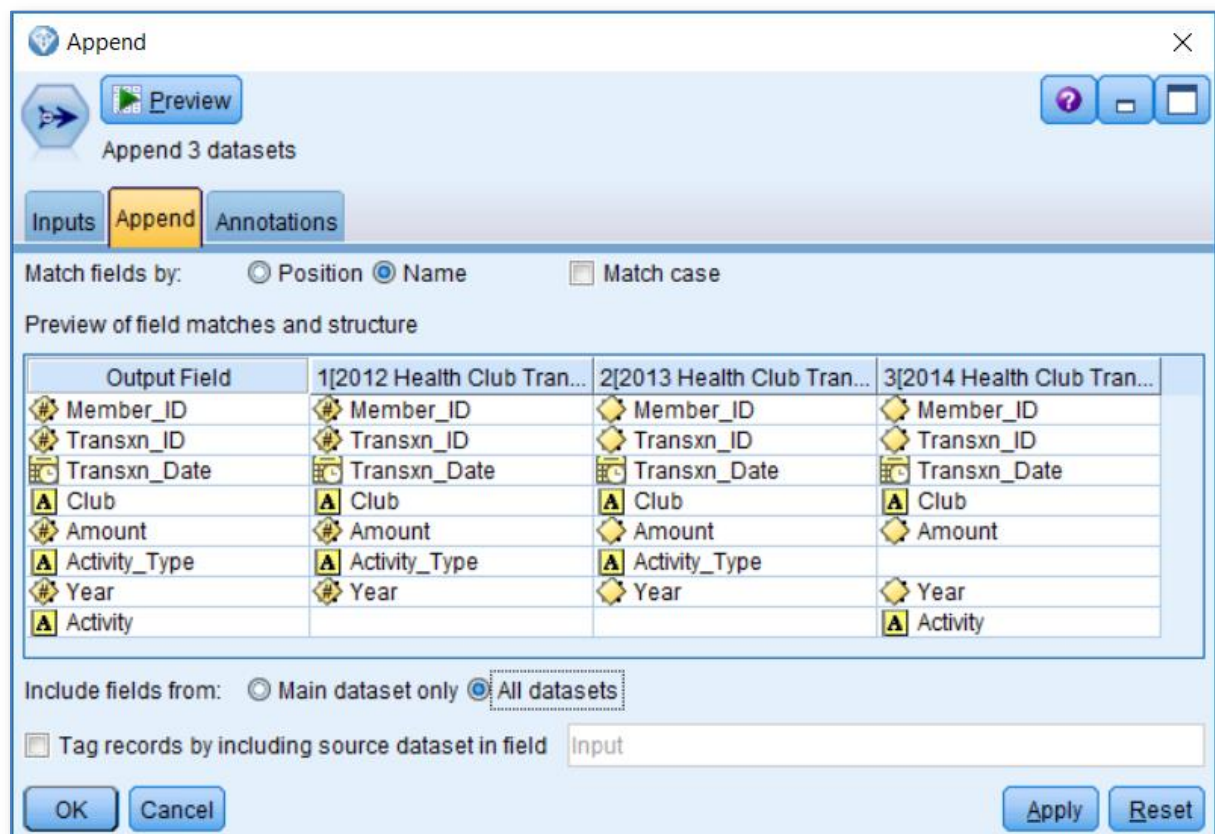


Figure 5.5 'Changing the 'Include fields from' option to 'All datasets'

We can see from the preview that switching off the leading dataset means that the values from the column 'Activity' in the 2014 data file will now be added to the appended output file, but unfortunately, they will appear as a separate column.

The easiest way to resolve this issue is of course to simply rename the column in the source node for the data file. To do so, click:

Reset

OK

Edit the source node for the file '2014 Health Club Transactions.csv' and switch to the Filter tab. Figure 5.6 shows the field name being changed from 'Activity' to 'Activity_Type'.

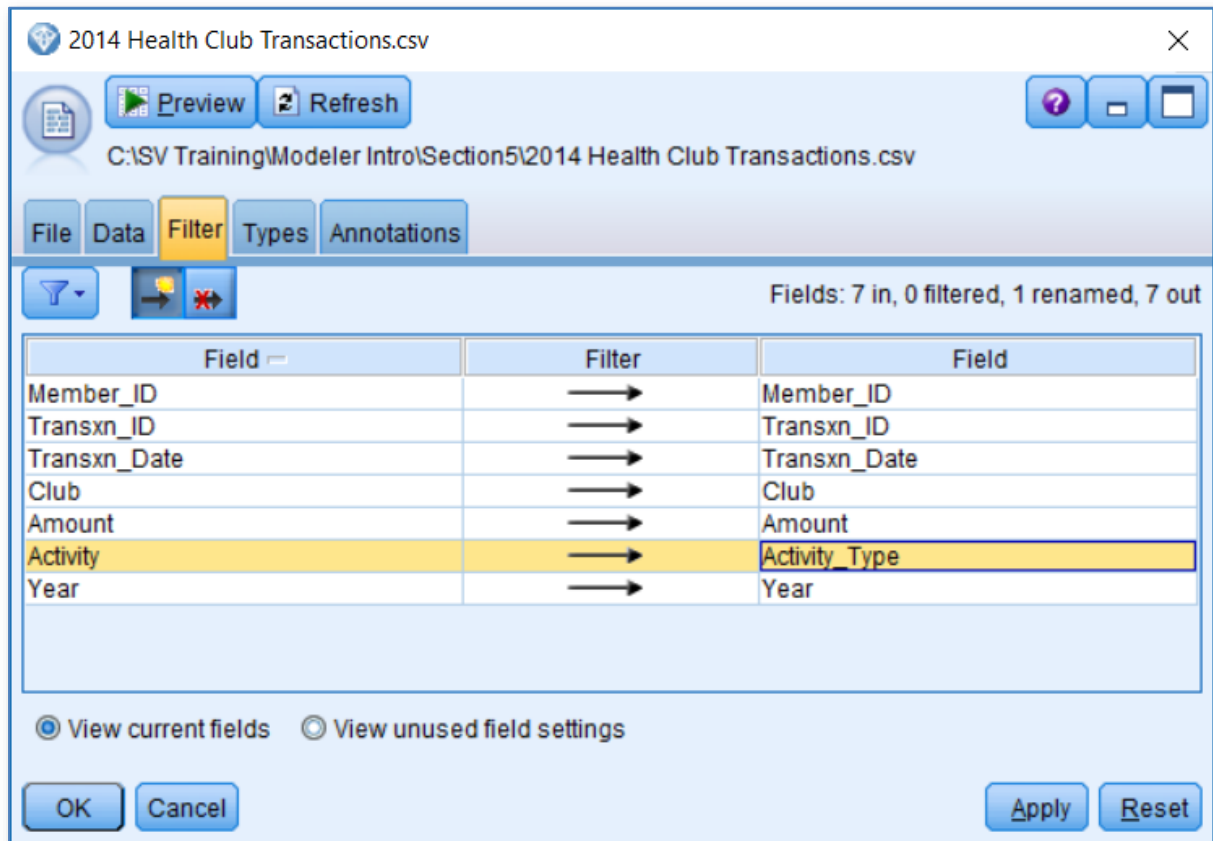


Figure 5.6 Changing the field name from 'Activity' to 'Activity_Type'

Now return to and edit the Append node.

We can immediately see that all the fields are now correctly matched. Within the Append tab, check the box marked:

'Tag records by including source dataset in field'

Edit the field name:

From 'Input' to 'Appended_File_Order'

Figure 5.7 shows the completed dialog.

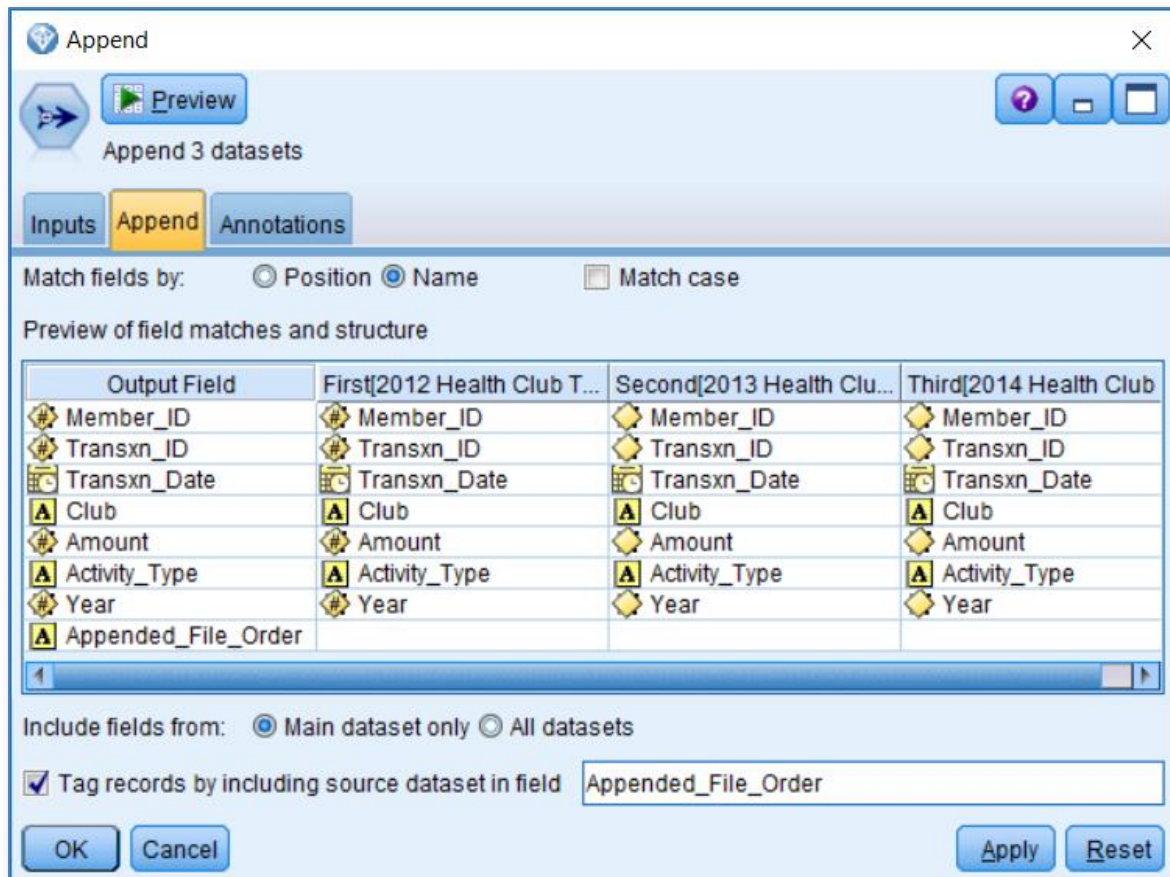


Figure 5.7 Completed Append dialog

Finally, before we run the procedure, click the tab marked:

Inputs

Within the Inputs tab, we find that by clicking on the rows that correspond to each file, we can click the and down arrow buttons in the dialog to change which file represents the 'Main dataset'. We can also click on the numbered labels within the newly created tag column ('Appended_File_Order') and edit them:

Edit the value Tag values from '1', '2' and '3' to 'First', 'Second' and 'Third'

Figure 5.8 shows the edited Inputs tab.

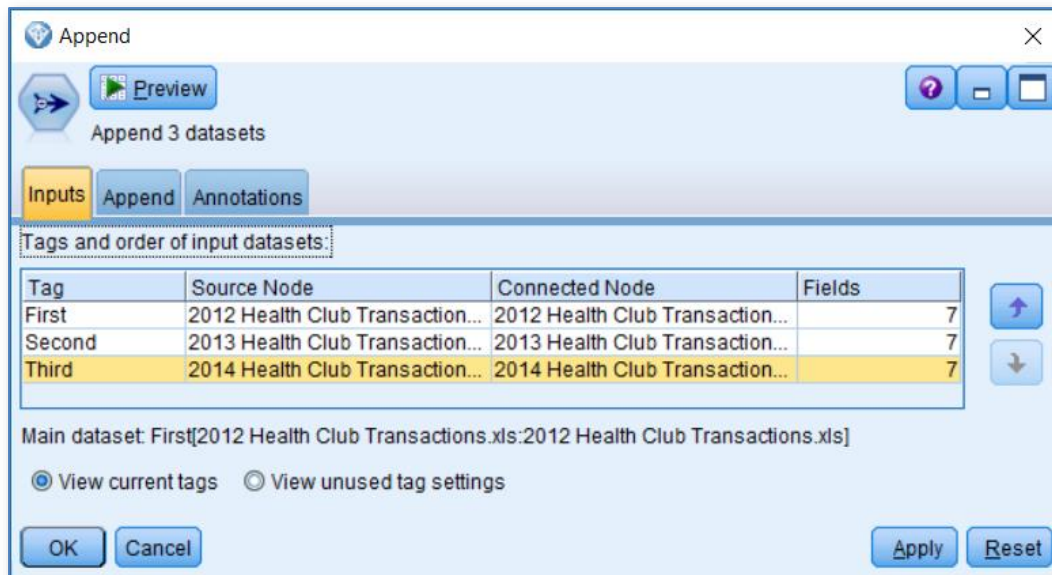


Figure 5.8 Inputs tab illustrating how we can choose the dataset order and edit the Tag values

To execute the procedure:

Attach a table node to the Append node and run that branch of the stream

Figure 5.9 shows the results in the table output.

	Member_ID	Transxn_ID	Transxn_Date	Club	Amount	Activity_Type	Year	Appended_File_Order
1	10530.000	98381640.0...	01/03/2012 07:38:00	Edinburgh	7.000	Squash	2012....	First
2	1090.000	98386434.0...	01/03/2012 09:03:00	Edinburgh	7.000	Squash	2012....	First
3	1095.000	98386866.0...	01/03/2012 09:10:00	Edinburgh	7.000	Squash	2012....	First
4	10701.000	98386854.0...	01/03/2012 09:10:00	Edinburgh	8.000	Gym Pass	2012....	First
5	10702.000	98387208.0...	01/03/2012 09:16:00	Edinburgh	4.000	Gym Pass	2012....	First
6	10120.000	98387844.0...	01/03/2012 09:21:00	Edinburgh	7.000	Squash	2012....	First
7	10703.000	98388018.0...	01/03/2012 09:23:00	Edinburgh	4.000	Gym Pass	2012....	First
8	10703.000	98388180.0...	01/03/2012 09:24:00	Edinburgh	8.000	Gym Pass	2012....	First
9	10117.000	98388666.0...	01/03/2012 09:26:00	Edinburgh	7.000	Squash	2012....	First
10	10704.000	98389080.0...	01/03/2012 09:28:00	Edinburgh	8.000	Gym Pass	2012....	First
11	10705.000	98389302.0...	01/03/2012 09:30:00	Edinburgh	4.000	Gym Pass	2012....	First
12	1077.000	98389512.0...	01/03/2012 09:32:00	Edinburgh	7.000	Squash	2012....	First
13	10575.000	98389638.0...	01/03/2012 09:34:00	London	8.000	Gym Pass	2012....	First
14	10575.000	98389746.0...	01/03/2012 09:35:00	London	8.000	Gym Pass	2012....	First
15	1091.000	98389704.0...	01/03/2012 09:35:00	Edinburgh	7.000	Squash	2012....	First
16	10575.000	98390058.0...	01/03/2012 09:36:00	London	8.000	Gym Pass	2012....	First
17	1096.000	98390130.0...	01/03/2012 09:37:00	Edinburgh	7.000	Squash	2012....	First
18	10212.000	98390088.0...	01/03/2012 09:37:00	Edinburgh	7.000	Squash	2012....	First
19	10706.000	98390202.0...	01/03/2012 09:38:00	Edinburgh	8.000	Gym Pass	2012....	First
20	10657.000	98390226.0...	01/03/2012 09:38:00	Edinburgh	7.000	Squash	2012....	First

Figure 5.9 Appended file comprised of 19,827 records created from three data sources

The Merge Node



The Merge node is normally used to create new datasets by adding fields to an existing data source. With the Append procedure, we were concerned with how the fields in the various files would match up with one another to create the appended output file. With the Merge node, a key consideration is how the *records* will match up between the files to create a consolidated row containing all the required fields. Merging data files can sometimes be a more complex process than appending records to an existing file because there are many ways in which the data files can be joined together. To begin with, we can consider the how the merge procedure determines which records should be joined.

- **Merging by Record Order** – This is a very simple way to merge records as it assumes that the first record in one dataset corresponds to the first record in the second dataset. This continues throughout all the rows of data in both datasets. To make the procedure work correctly, the analyst must ensure that both datasets are sorted in the correct order.
- **Merging with a Key field** – This is much more common scenario than merging data based on the record order. A key field is an identifier such as a customer ID number, a transaction code or an email address. The merging process then uses the identifier to match up the correct records in both files. Often a single unique identifier doesn't exist and so more than one field is specified to create a *compound ID* (such as Date of Birth, First Name and Surname). We should note that using any sort of keyed field to merge datasets creates choices as to the method used to *join* the data (of which more later).
- **Merging by Condition** – This merge method is employed less often than the previous technique. It allows the user to specify a condition that must be met for the merge to happen. For example, you could merge data based on the values in one key field but only if the data are not missing. If however the data *is* missing within the keyed field, you could revert to matching on a different field. The condition itself can be specified directly in the node or created using the Expression Builder.
- **Merge by Ranked Condition** – Like the previous merge type, using a ranked condition allows you to merge data according to a conditional rule but sorts the rows from low to high. For example, you might wish to merge records based on the fact that they coincide within the same geographical region and sort them in order of their proximity to specific location within the area. This is quite a specialist method and is used less often.

When merging records with keyed fields, there are various ways in which datasets can be joined together.

Types of Join

In the following figures, we can see examples of the effects of choosing different methods for joining datasets together within a merge procedure. In each example, we begin with the same two files. Each file contains a single record with an ID value that doesn't match with the other file.

Inner Join

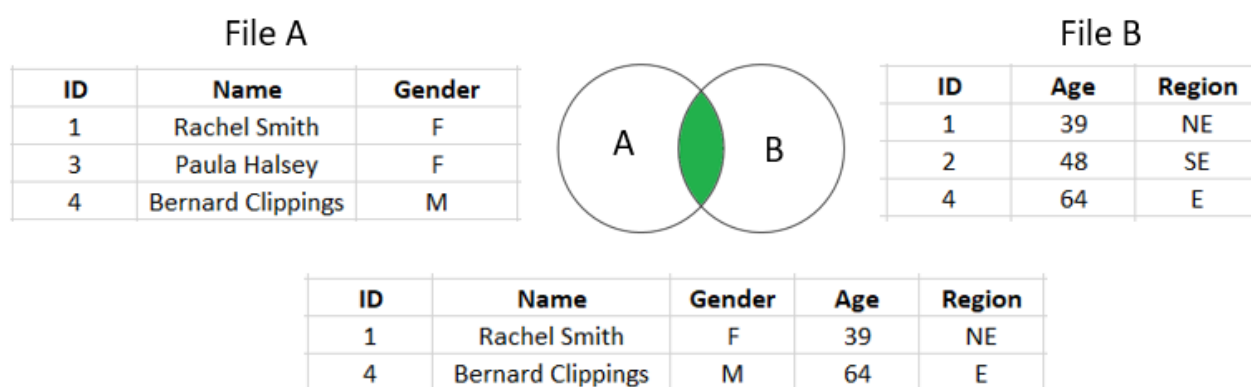


Figure 5.10 Inner Join

Using an Inner Join, the merge procedure creates a file with only two cases. This is because the inner join method only joins those records where a match can be found in both files.

Full Outer Join

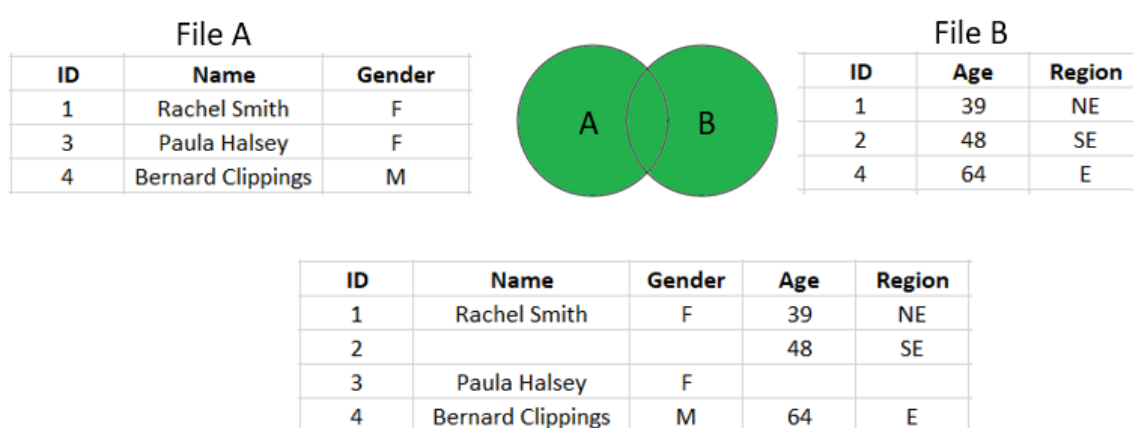


Figure 5.11 Full Outer Join

Figure 5.11 shows the effect of choosing a Full Outer Join method. In this case the merged file contains all the records from both files even if a match cannot be found. As a result, the

merged file contains empty (or missing) cells for those records where the ID value cannot be used to form a pair.

Partial Outer Join

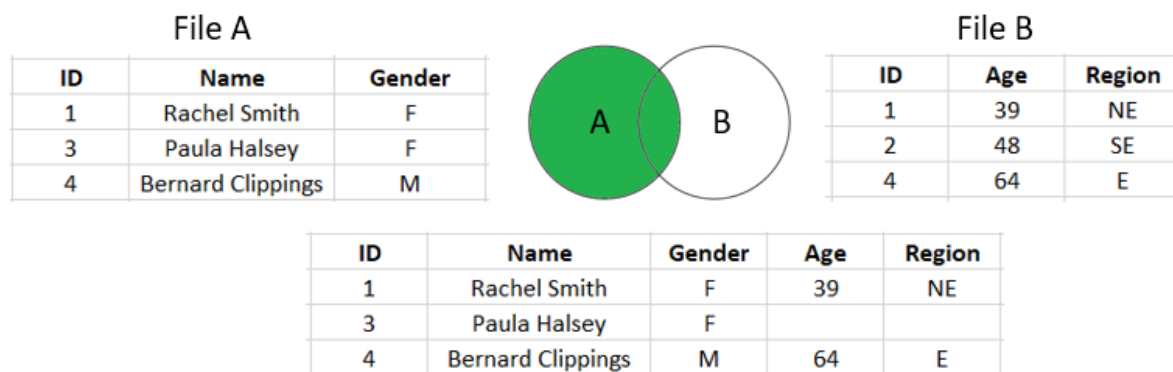


Figure 5.12 Partial Outer Join

Partial outer joins (often called 'left' or 'right' outer joins) create merged data where all the records one table are retained (commonly referred to as the keyed table) and only the *matching* records from the other table(s) are added. This is a commonly used join method when we wish to retain all the records from one 'master' table. For example, a table containing the demographic details of 60K customers may be retained and another table containing their aggregated transaction values may be joined to it. Assuming there are no duplicates, we will still end up with 60K customers even if there were some customers where we could not find any transaction values to match on.

Anti-Join

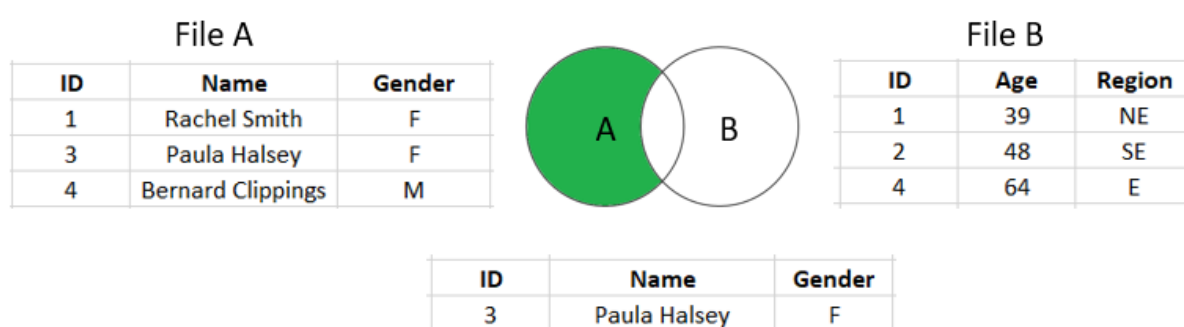


Figure 5.13 Anti-Join

As figure 5.13 shows, an anti-join results in a dataset of records from the first table where no match could be found. It can be a useful diagnostic tool as it isolates the records that can't be matched to the data in the other file(s). It's important to point out that all the examples above are based on data without duplicates. Merging data with records containing duplicate keys may result in output with more records than either file originally contained as each

permutation of the duplicates creates an additional record. This is not always desired or expected so removing duplicates before the merge process often makes a lot of sense.

Before we begin using the Merge node in our demonstration stream, we will need to prepare the appended data so that the unit of analysis is converted to one row per customer. We can do this if we can use the Aggregate and Set to Flag nodes that we saw in the previous chapter. To save time, both of the nodes have already been configured and encapsulated within a supernode which we can add to the stream. To do so, from the main menu within Modeler, click:

Insert

Supernode from file...

Navigate to the folder marked 'Section 5' and select the file

'Aggregate & Set to flag.slb'

Insert

The pre-prepared supernode is now inserted into the stream. To see how it transforms the appended data, connect the node to Append node and add a Table node downstream of the supernode. Figure 5.14 shows the supernode added to the stream with a Table node as the terminal point in the branch.

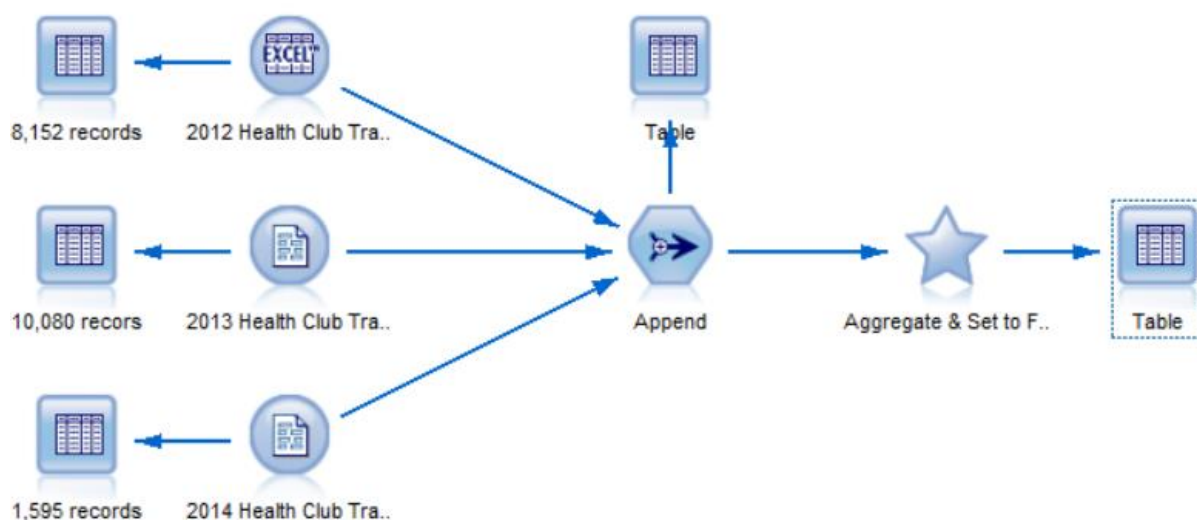


Figure 5.14 An inserted supernode added to the Append procedure to prepare the data for merging

Figure 5.15 shows the table output generated from this modified branch of the stream. We can see that the appended and transformed data is comprised of 3,814 records.

	Member_ID	First_Transn_Date	Last_Transn_Date	Club	Total_Amount	Average_Amount	Most_Recent_Year	Num_of_Transactions	Activity_Type_Aerobic	Activity_Type_Gym Pass	Activity_Type_Kids Activities	Activity_T
1	100.000	21/03/2013 16:52:00	13/01/2014 17:28:00	Bristol	21.000	7.000	2014.000	3	F	F	F	F
2	101.000	14/10/2013 13:09:00	14/10/2013 13:09:00	Manchester	36.000	36.000	2013.000	1	T	F	F	F
3	102.000	12/01/2014 09:55:00	12/01/2014 09:55:00	Manchester	36.000	36.000	2014.000	1	T	F	F	F
4	103.000	09/01/2014 18:35:00	09/01/2014 18:35:00	Manchester	36.000	36.000	2014.000	1	T	F	F	F
5	104.000	08/10/2013 08:48:00	08/10/2013 08:48:00	Manchester	36.000	36.000	2013.000	1	T	F	F	F
6	105.000	01/03/2014 08:34:00	01/03/2014 08:34:00	Manchester	36.000	36.000	2014.000	1	T	F	F	F
7	106.000	23/04/2013 09:44:00	03/03/2014 09:23:00	Manchester	100.000	50.000	2014.000	2	T	F	F	F
8	107.000	06/03/2014 18:51:00	06/03/2014 18:51:00	Manchester	36.000	36.000	2014.000	1	T	F	F	F
9	108.000	09/11/2013 13:04:00	09/11/2013 13:04:00	Manchester	36.000	36.000	2013.000	1	T	F	F	F
10	109.000	20/02/2014 20:18:00	20/02/2014 20:18:00	Manchester	36.000	36.000	2014.000	1	T	F	F	F
11	1010.000	03/03/2014 17:56:00	03/03/2014 17:56:00	Manchester	36.000	36.000	2014.000	1	T	F	F	F
12	1011.000	13/11/2013 21:08:00	13/11/2013 21:08:00	Manchester	36.000	36.000	2013.000	1	T	F	F	F
13	1012.000	01/03/2014 15:11:00	01/03/2014 15:11:00	Manchester	36.000	36.000	2014.000	1	T	F	F	F
14	1013.000	09/01/2014 22:04:00	09/01/2014 22:04:00	Manchester	36.000	36.000	2014.000	1	T	F	F	F
15	1014.000	28/08/2013 18:37:00	28/08/2013 18:37:00	Manchester	36.000	36.000	2013.000	1	T	F	F	F
16	1015.000	02/09/2013 15:43:00	02/09/2013 15:43:00	Manchester	36.000	36.000	2013.000	1	T	F	F	F
17	1016.000	02/09/2013 15:44:00	02/09/2013 15:44:00	Manchester	36.000	36.000	2013.000	1	T	F	F	F
18	1017.000	01/09/2013 10:41:00	01/09/2013 10:41:00	Manchester	36.000	36.000	2013.000	1	T	F	F	F
19	1018.000	09/10/2013 10:39:00	09/10/2013 10:39:00	Manchester	36.000	36.000	2013.000	1	T	F	F	F
20	1019.000	07/10/2013 10:38:00	07/10/2013 10:38:00	Manchester	36.000	36.000	2013.000	1	T	F	F	F

Figure 5.15 Table output from the inserted supernode showing the data converted to one row customer.

Having converted the data via the supernode, we can now merge it with the de-duped data from the file 'Health_Club_Contacts.csv'. From the Record Ops palette:

Place a Merge node on the stream canvas

Connect the Supernode to the Merge node

Connect the last Distinct node to the Merge node

Figure 5.16 shows the two stream branches joined by the Merge node.

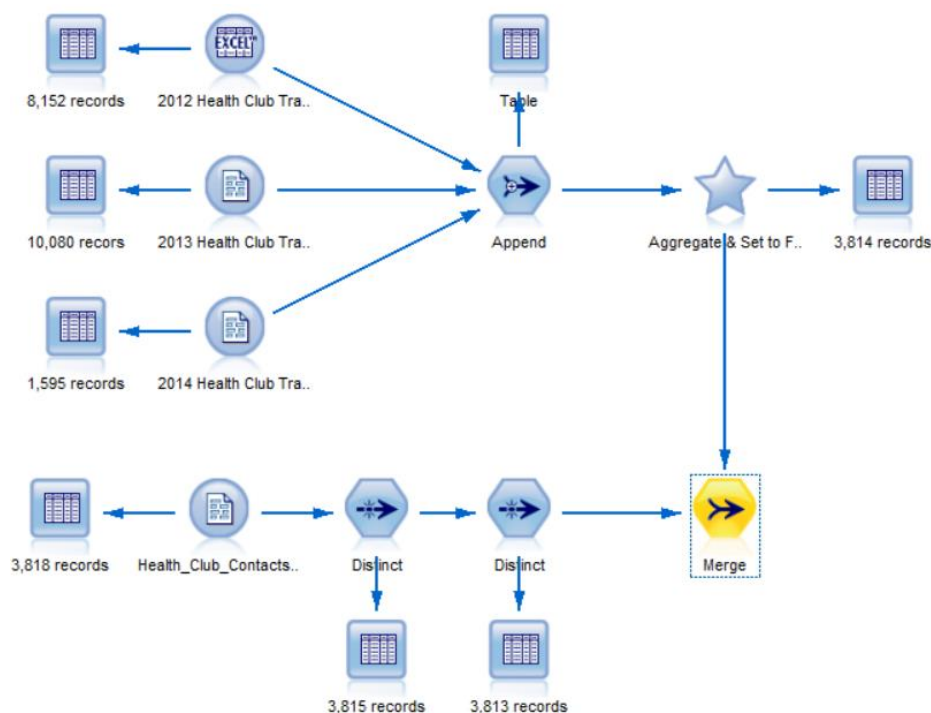


Figure 5.16 Using the Merge node to join transactional data and customer contact information together

We can now configure the Merge node to join the customer contact data to the appended and summarised transactional data.

Right-click on and edit the Merge node

As the resultant dialog shows, the default Merge method is 'Order'. This assumes that both files are sorted in the same order and that there is an exact match in terms of the relative positions of the records in both files (e.g. record 2,925 in first file matches with record 2,925 in the second file and so on). In most situations this is rarely appropriate. Here we need to match by a key field, so within the dialog click on the drop-down menu marked...

Method

...and choose

Keys

We can see that the field 'Member_ID' is chosen as a possible key because it happens that there is a field with that exact name in both files. Select this field and send it to the list box marked 'Keys for merge' by clicking:



Figure 5.17 shows the Merge dialog so far.

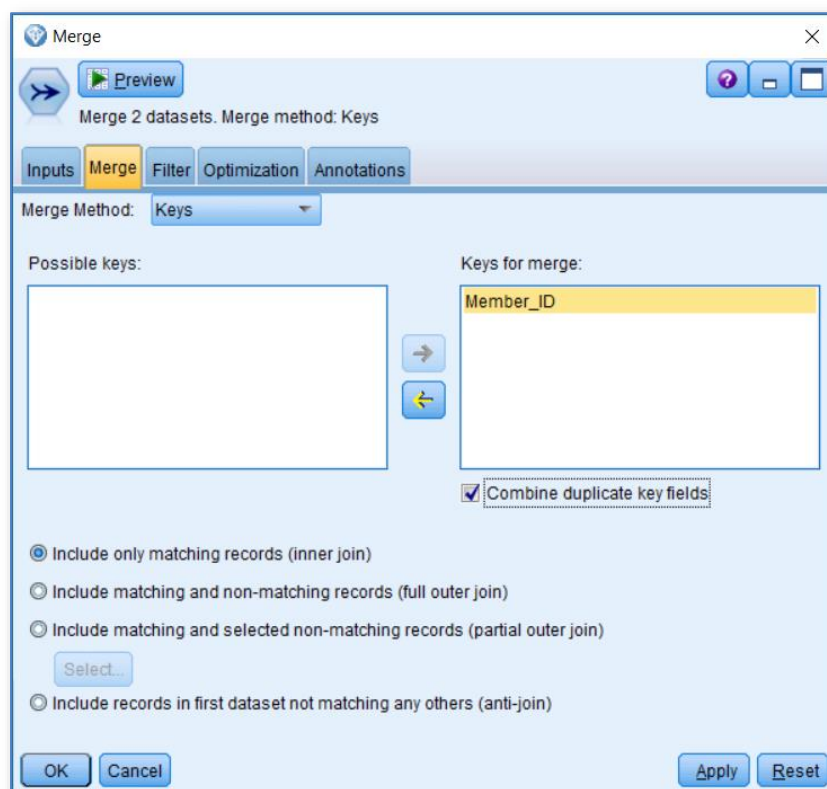


Figure 5.17 The edited Merge node using a key field to define the merge

At this point we can choose a join method to complete the merge. Before we do so, you may recall that when we used the Distinct nodes earlier to remove records with duplicate customers IDs and email addresses, we ended up with 3,813 records. However, after adding the supernode to the appended transactional data we had 3,814 records. Clearly one record cannot find a match in both tables. Remember that we can use the anti-join method here to identify this case. Using an anti-join identifies the records from the *first* table that couldn't be matched to any of the subsequent tables. The data from the supernode made the first connection to the merge node, so it constitutes the first table (although the order of the connections can be altered using the 'Inputs' tab in the edited Merge node). To identify which case from the transactional data can't be matched to the contact data, click the radio button marked:

Include records in the first dataset not matching any others (anti-join)

Now click the button marked:

Preview

Figure 5.18 shows the data preview output that is generated.

The screenshot shows a window titled 'Preview from Merge Node (15 fields, 1 records) #1'. It has a toolbar with 'File', 'Edit', 'Generate', and other icons. Below the toolbar are tabs for 'Table' and 'Annotations'. The 'Table' tab is active, displaying a table with 9 columns: Member_ID, First_Transxn_Date, Last_Transxn_Date, Club, Total_Amount, Average_Amount, Most_Recent_Year, and Num_of. The table contains one row with the following data: Member_ID: 10610.000, First_Transxn_Date: 17/05/2012 10:21:00, Last_Transxn_Date: 17/05/2012 10:21:00, Club: London, Total_Amount: 8.000, Average_Amount: 4.000, Most_Recent_Year: 2012.000, and Num_of: 1. The Member_ID '10610.000' is highlighted in yellow. At the bottom right is an 'OK' button.

	Member_ID	First_Transxn_Date	Last_Transxn_Date	Club	Total_Amount	Average_Amount	Most_Recent_Year	Num_of
1	10610.000	17/05/2012 10:21:00	17/05/2012 10:21:00	London	8.000	4.000	2012.000	

Figure 5.18 Unmatched record identified using anti-join method in the merge procedure

The anti-join method shows that the record with Membership ID '10610' cannot be found in the customer contacts data. At this stage, we can make a decision as to whether we wish to use a join method to create a merged dataset that also contains this record (albeit without any of the contact information) or whether we simply ignore it and perform a join that drops this unmatched record from the output data. As it is only one record, we can afford to create a merged dataset without it. To do this, we could either perform an inner-join or a partial outer join using the contacts data containing 3,813 records as the keyed table: either approach will create identical results.

From within the Merge dialog, select the radio button marked:

Include only matching records (inner join)

Figure 5.19 shows the completed Merge dialog.

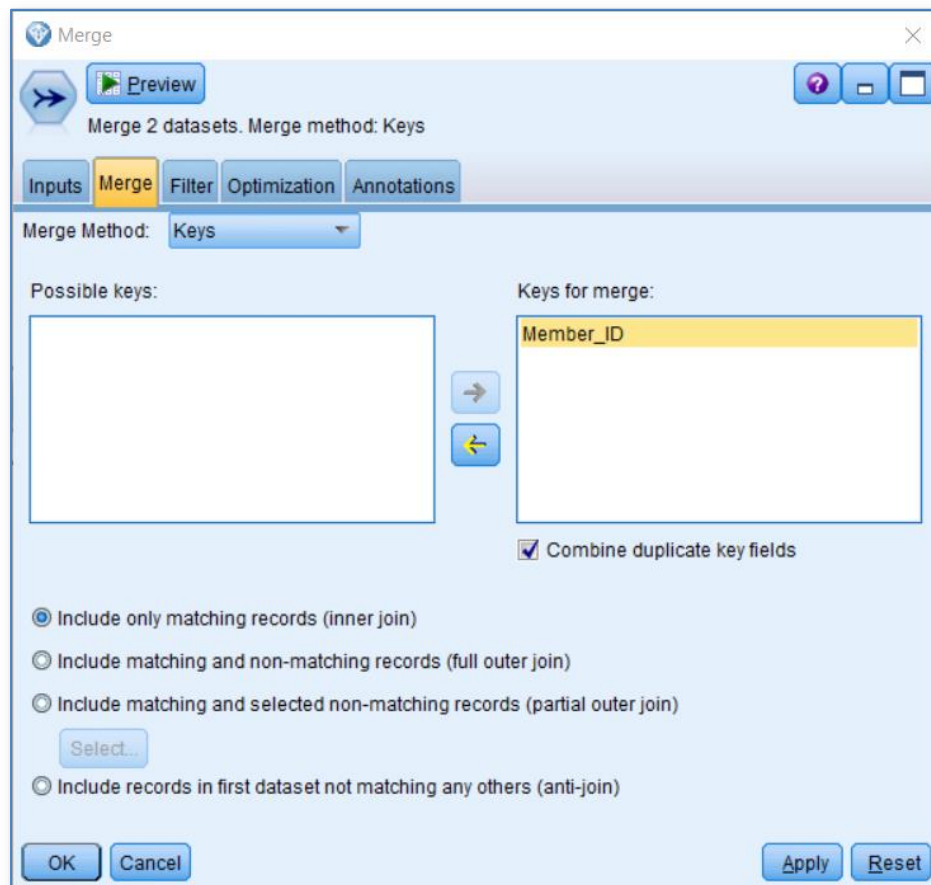


Figure 5.19 Using the Merge node to perform an inner join

To perform the inner join:

Click 'OK' and add a Table node downstream of the Merge node

Figure 5.20 shows a partial view of the merged table output data and figure 5.21 shows the Modeler stream so far.

Table (23 fields, 3,813 records) #2

	Activity_Type_Squash	Activity_Type_Swimming	Gender	Firstname	Lastname	email	Advantage_Ct
1	T	F	Male	Samson	Blake	SBlake@AP.com	Racquets
2	F	F	Male	Agustin	Sloan	ASloan@SAS.org	Classes
3	F	F	Female	Monserrat	Petty	MontserratPetty@hotmail.com	Classes
4	F	F	Female	Luz	Taylor	LuzTaylor@SAS.net	Classes
5	F	F	Female	Alena	Shea	Alena_Shea@AP.co.uk	Classes
6	F	F	Female	Jimena	Hancock	JHancock@gmail.net	Classes
7	F	F	Male	Finnegan	Espinoza	F.Espinoza@Milestones.org	Classes
8	F	F	Male	Bradley	Chang	B.Chang@hotmail.biz	Classes
9	F	F	Male	Cristian	Alvarez	C_Alvarez@Arcturus.co.uk	Classes
10	F	F	Male	Corey	Archer	Corey_Archer@yahoo.com	Classes
11	F	F	Male	Orion	Burch	Orion.Burch@global.co.uk	Classes
12	F	F	Male	Bradley	Houston	B.Houston@Arcturus.com	Classes

Figure 5.20 Table output showing the merged data

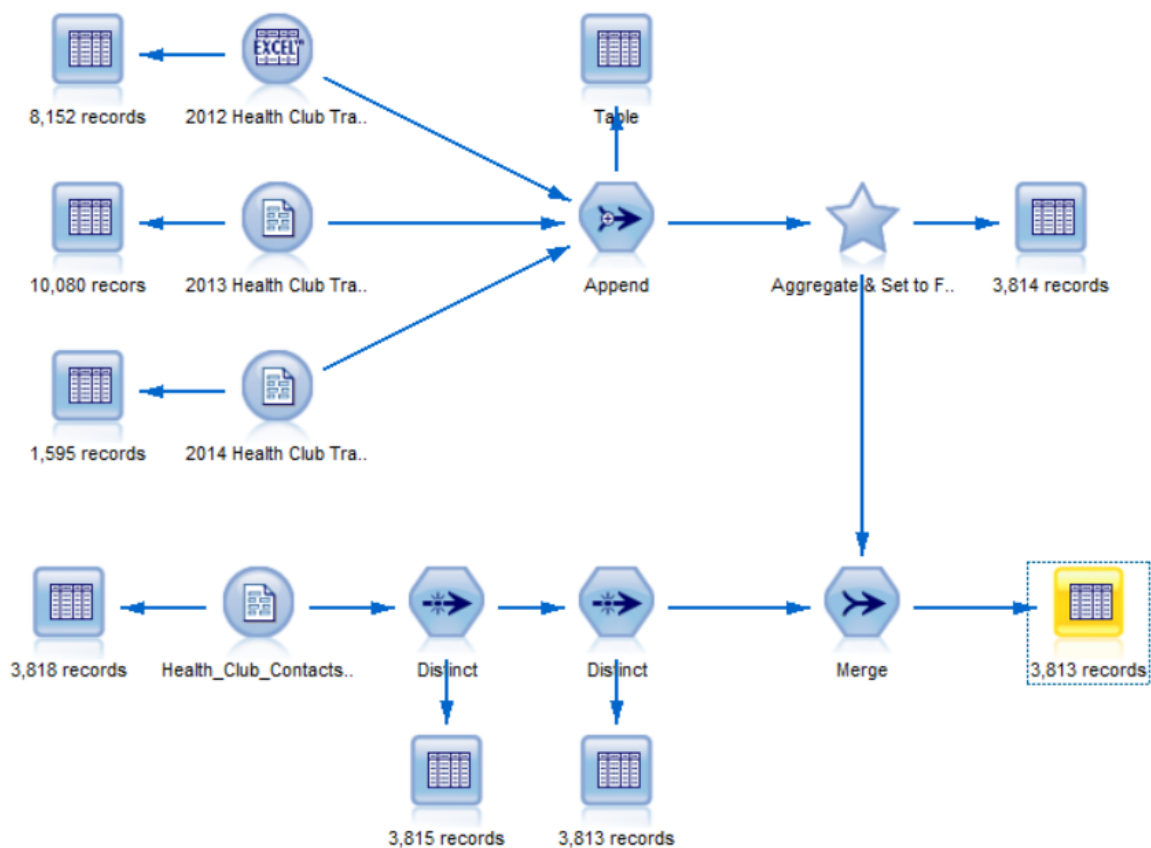
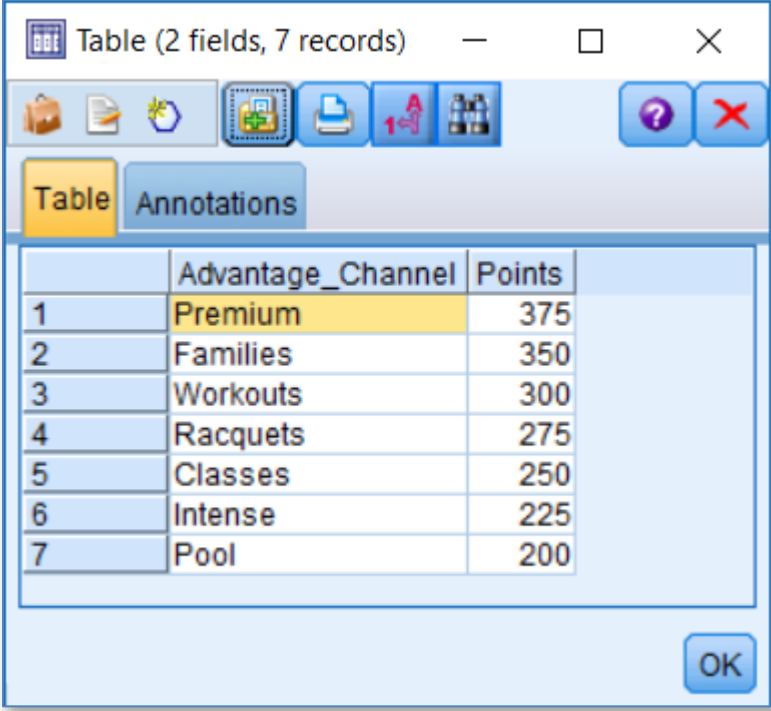


Figure 5.21 The Merge node used to join two data streams

A situation that often occurs, is the requirement to merge data from a reference table that contains additional information about one of the fields in the main dataset. Figure 5.22 shows the contents of a dataset called 'Channel_Loyalty_Points.csv'. This dataset simply contains

extra information about how many loyalty points are associated with each preferred loyalty channel.



	Advantage_Channel	Points
1	Premium	375
2	Families	350
3	Workouts	300
4	Racquets	275
5	Classes	250
6	Intense	225
7	Pool	200

Figure 5.22 Loyalty points reference table

Obviously, if we use the merge node to add this data to existing stream we cannot match by Customer ID as the loyalty points are not associated with individual customers but rather the activities that the club wants to reward loyalty for. As we will be using a different key to perform the merge, we will need to add a separate Merge node. The source node for this small dataset can be found at the bottom of the stream.

Add a new Merge node and connect the Source node 'Channel_Loyalty_Points.csv' to it

Connect the newly added Merge node to existing Merge node

Figure 5.23 shows the stream containing the two merge nodes.

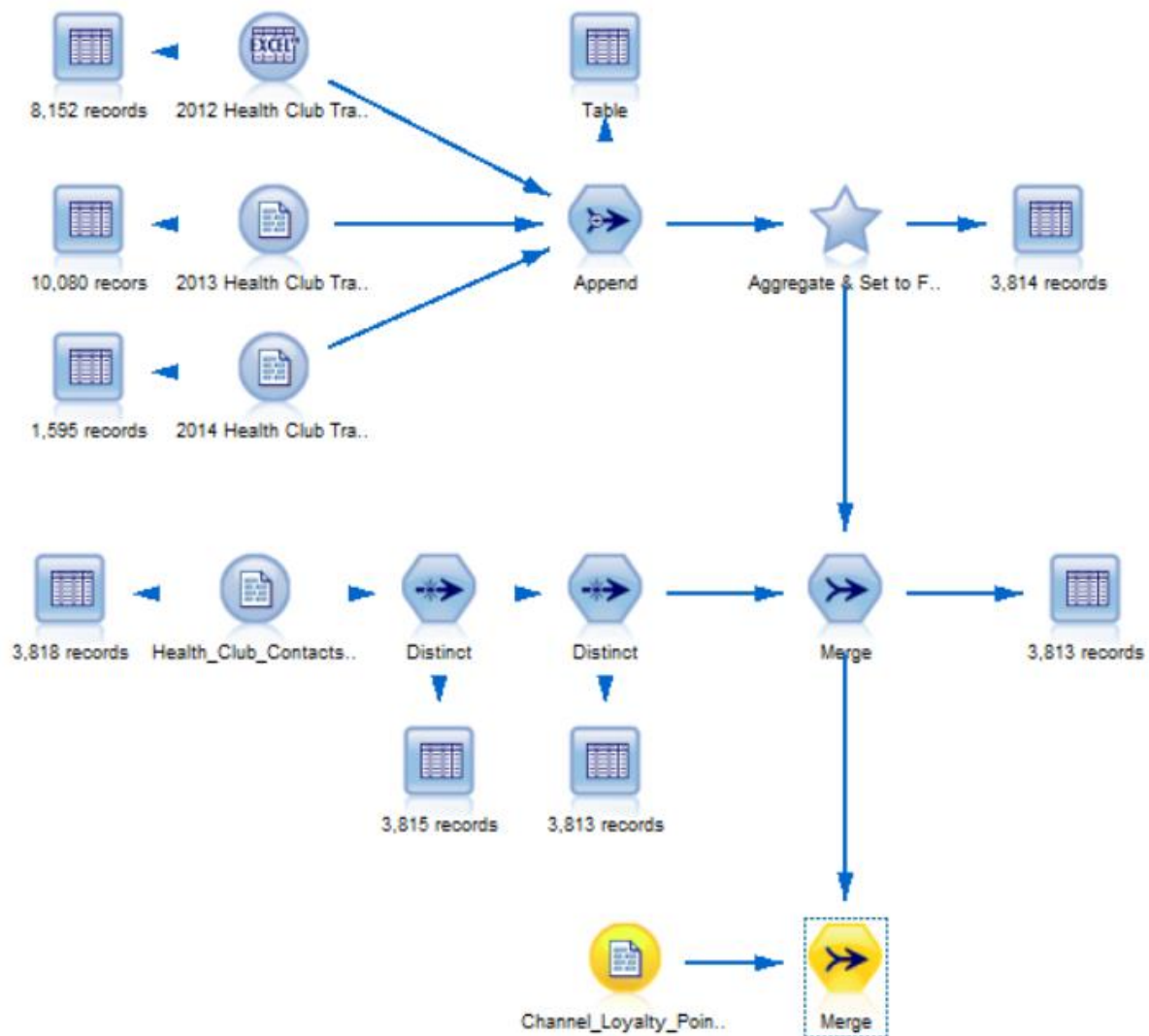


Figure 5.23 Second Merge node added to the stream

To perform the merge:

Right-click on and Edit the new merge node

Again, we will need to perform a merge using a key field, so *within the dialog*, click on the drop-down menu marked...

Method

...and choose

Keys

On this occasion, we can see from the dialog within the edited Merge node, that the only field identified as a possible key is 'Advantage_Channel'.

Select this field and send it to the 'Keys for merge' list box.

Figure 5.24 shows the partially completed dialog.

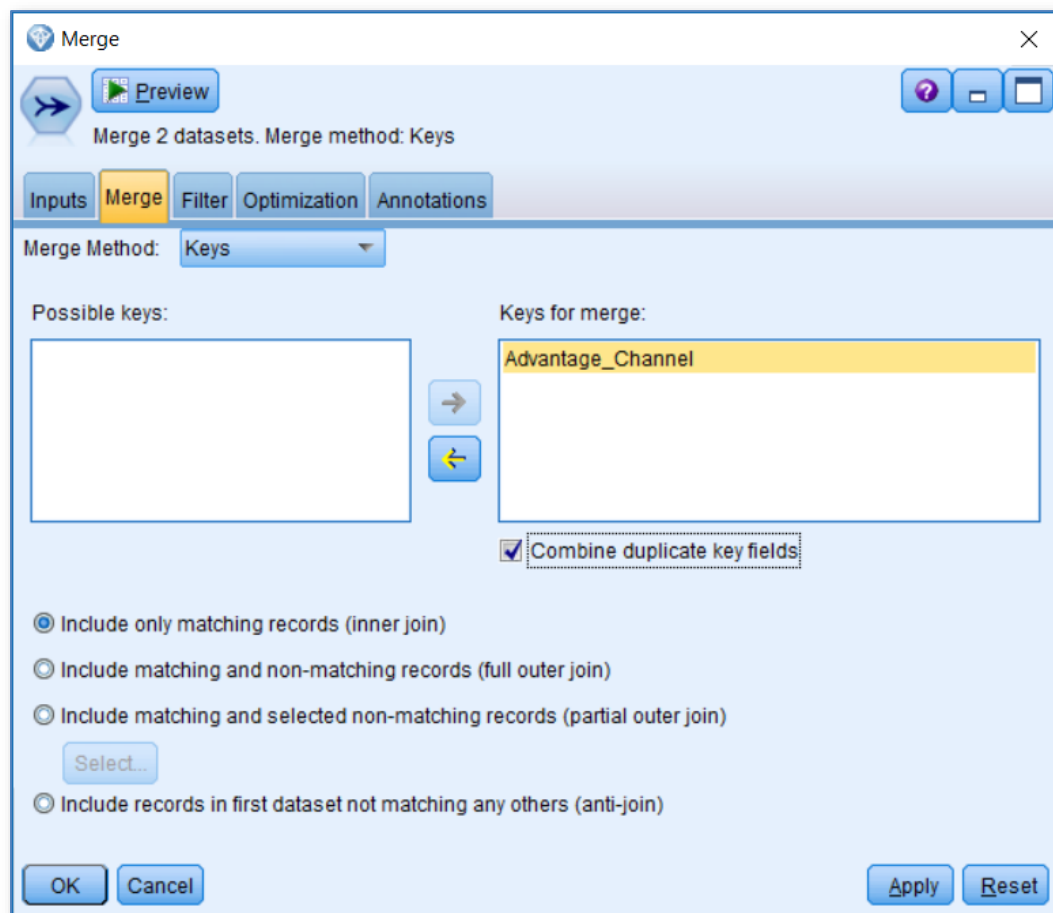


Figure 5.24 Partially completed Merge node for adding data from a reference table

Using the 'Inner join' method will not work here as the reference table only contains 7 records. Instead we must define a partial outer join where the already merged data forms the 'outer join'. To do this, click the radio button marked:

Include matching and selected non-matching records (partial outer join)

Now click:

Select

In the sub-dialog that is generated:

Check the box marked 'Outer join' that corresponds to the already merged data in the source node (and connected node) marked 'Merge'

Figure 5.25 shows the completed outer join sub-dialog and the main merge dialog.

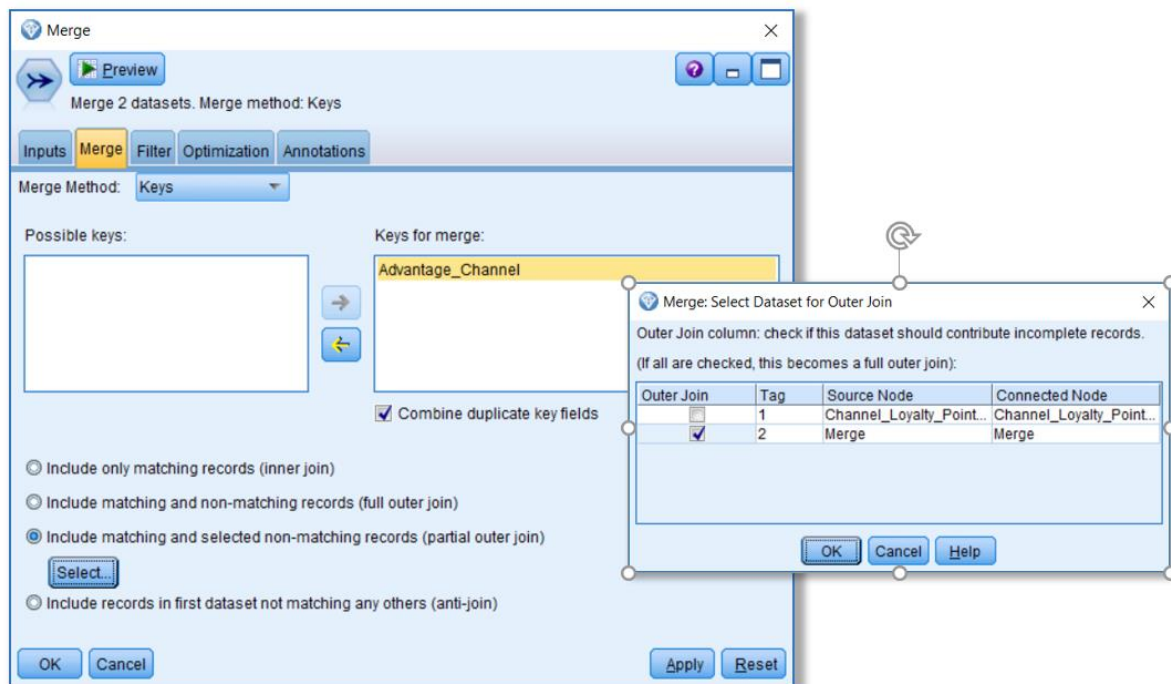


Figure 5.25 Completed Merge node using a Partial Outer Join

To continue, click:

OK

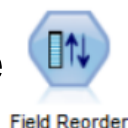
OK

Add a Table node to the new Merge node and run the stream branch. Figure 5.26 shows some of the table output.

Table (24 fields, 3,813 records) #1									
	Advantage_Channel	Points	Member_ID	First_Transxn_Date	Last_Transxn_Date	Club	Total_Amount	Average_Amount	
1	Classes	250	103858.000	16/09/2012 09:46:00	16/09/2012 09:46:00	Lond...	108.000	108.000	
2	Classes	250	103857.000	16/09/2012 09:31:00	16/09/2012 09:31:00	Lond...	108.000	108.000	
3	Classes	250	103856.000	16/09/2012 12:51:00	16/09/2012 12:51:00	Lond...	100.000	100.000	
4	Classes	250	103855.000	11/12/2013 18:51:00	11/12/2013 18:51:00	Bristol	134.000	134.000	
5	Classes	250	103854.000	14/06/2013 09:35:00	14/06/2013 09:35:00	Lond...	46.000	46.000	
6	Classes	250	103853.000	24/03/2013 10:26:00	24/03/2013 10:26:00	Bristol	162.000	162.000	
7	Classes	250	103852.000	17/04/2013 15:00:00	17/04/2013 15:00:00	Lond...	100.000	100.000	
8	Classes	250	103851.000	26/04/2013 18:05:00	26/04/2013 18:05:00	Lond...	46.000	46.000	
9	Classes	250	103850.000	17/01/2013 14:27:00	17/01/2013 14:27:00	Lond...	62.000	62.000	
10	Classes	250	103849.000	15/03/2013 18:56:00	15/03/2013 18:56:00	Bristol	217.000	217.000	
11	Classes	250	103848.000	13/01/2013 11:27:00	13/01/2013 11:27:00	Bristol	77.000	77.000	
12	Classes	250	103847.000	14/01/2013 12:10:00	14/01/2013 12:10:00	Bristol	77.000	77.000	
13	Classes	250	103846.000	30/03/2013 11:22:00	30/03/2013 11:22:00	Lond...	100.000	100.000	
14	Classes	250	103845.000	06/01/2013 10:34:00	06/01/2013 10:34:00	Bristol	85.000	85.000	

Figure 5.26 Table output from merging loyalty data from a reference table to the main merged file

The Field Reorder Node



Now that we have consolidated a number of data files into a single dataset, we can edit the order of the fields within the file itself. As the name implies, the Field Reorder node allows us to do just that. The node is found within the Field Ops palette and we can add it to the most recent Merge node that we placed on the stream canvas.

Double click on the Field Reorder node in the Field Ops palette to add it to the stream canvas

Join the Field Reorder node to the last Merge node

Right-click on the Field Reorder node and edit it

Figure 5.27 shows the default view of the edited Field Reorder node

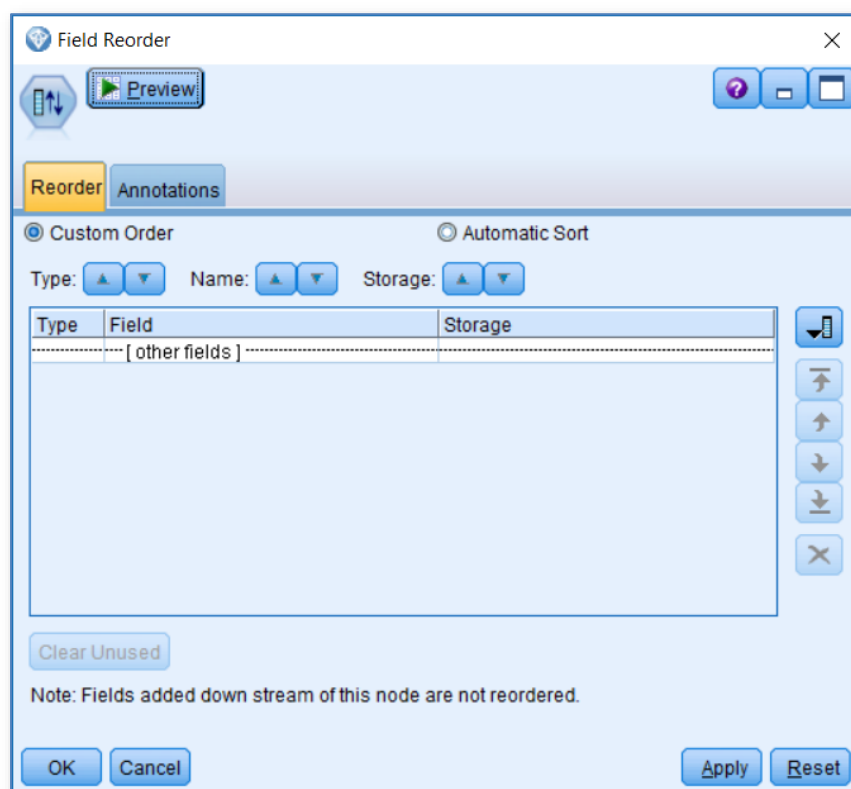


Figure 5.27 The default view of the Field Reorder node

The Reorder tab within the node allows us to either automatically sort the fields in terms of their field names, type or storage or alternatively to define a custom order. To define a custom order, click the field picker button:



Choose the following fields to appear at the start of the file:

Member_ID
First_Transxn_Date
Last_Transxn_Date

Click the field picker button again:



Choose the next group of fields (at the end of the list) to appear after 'Last_Transxn_Date'.

Gender
Firstname
Lastname
email
Local_Club

In the field order list click:

[other fields]

Send it to the bottom of the field order list by clicking:



The completed dialog should look like figure 5.28.

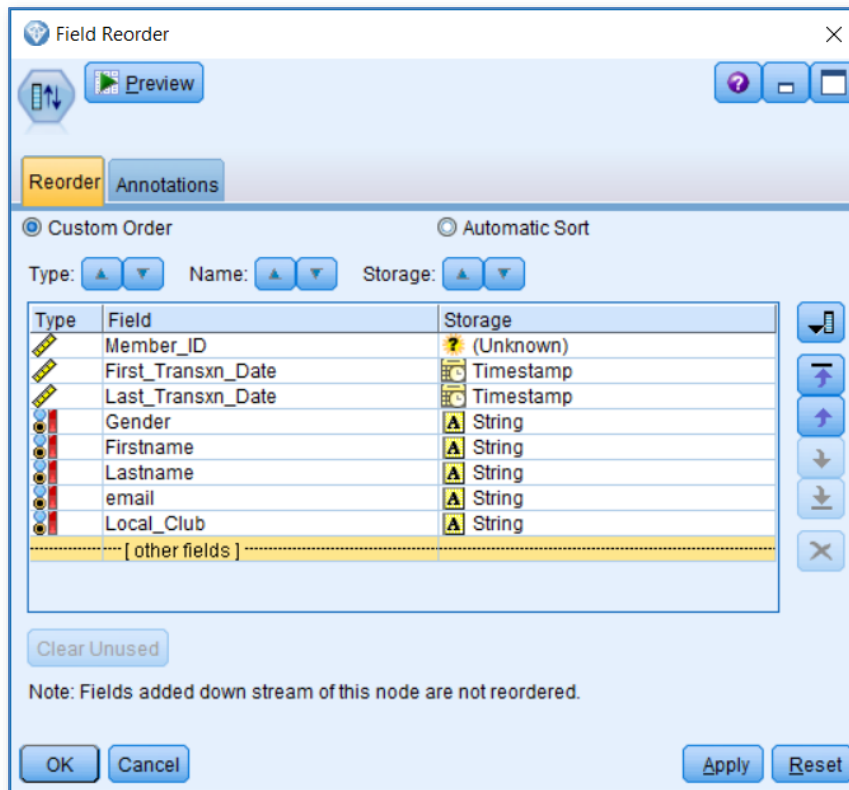


Figure 5.28 Defining the field order within the Field Reorder node

By clicking the data preview button, we can see that the fields have indeed been reordered (figure 5.29).

	Member_ID	First_Transxn_Date	Last_Transxn_Date	Gender	Firstname	Lastname	email	Local_Club
1	103858.000	16/09/2012 09:46:00	16/09/2012 09:46:00	Male	Clinton	Lowe	C_Lowe@global.com	London
2	103857.000	16/09/2012 09:31:00	16/09/2012 09:31:00	Female	Kaleigh	Fletcher	K.Fletcher@global.com	London
3	103856.000	16/09/2012 12:51:00	16/09/2012 12:51:00	Male	Vaughn	Page	Vaughn_Page@BT.biz	London
4	103855.000	11/12/2013 18:51:00	11/12/2013 18:51:00	Female	Savanna	Davies	Savanna_Davies@Smartvision.biz	Bristol
5	103854.000	14/06/2013 09:35:00	14/06/2013 09:35:00	Female	Kenley	Ochoa	Kenley.Ochoa@aol.co.uk	London
6	103853.000	24/03/2013 10:26:00	24/03/2013 10:26:00	Female	Bria	Gill	Bria_Gill@SAS.org	Bristol
7	103852.000	17/04/2013 15:00:00	17/04/2013 15:00:00	Male	Vaughn	Hendricks	V_Hendricks@global.biz	London
8	103851.000	26/04/2013 18:05:00	26/04/2013 18:05:00	Male	Uriel	Kaiser	U.Kaiser@Milestones.net	London
9	103850.000	17/01/2013 14:27:00	17/01/2013 14:27:00	Female	Charity	Noble	C_Noble@Smartvision.net	London
10	103849.000	15/03/2013 18:56:00	15/03/2013 18:56:00	Female	Karlee	Potts	Karlee.Potts@IBM.org	Bristol

Figure 5.29 Data preview showing the effects of the Field Reorder node

Finally, we can export the consolidated and merged data as a complete file. To do so, from the Export palette, choose the following node and attach it to the Field Reorder node:

Flat File

Edit the node so that it is exported as a Tab-delimited file

Name the file 'Health_Club_Merged.txt' and request that it is saved in the Section 5 folder

Figure 5.30 shows the complete dialog.

Run the procedure so that the data is exported

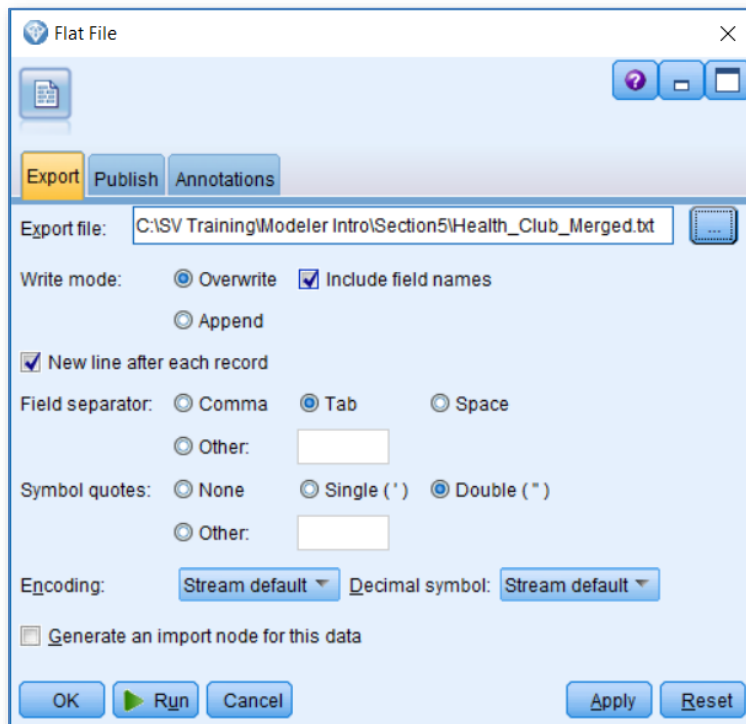


Figure 5.30 Exporting the merged data as a single flat file

Finally, save the completed stream as:

Section 5 complete.str

Figure 5.31 shows the completed stream.

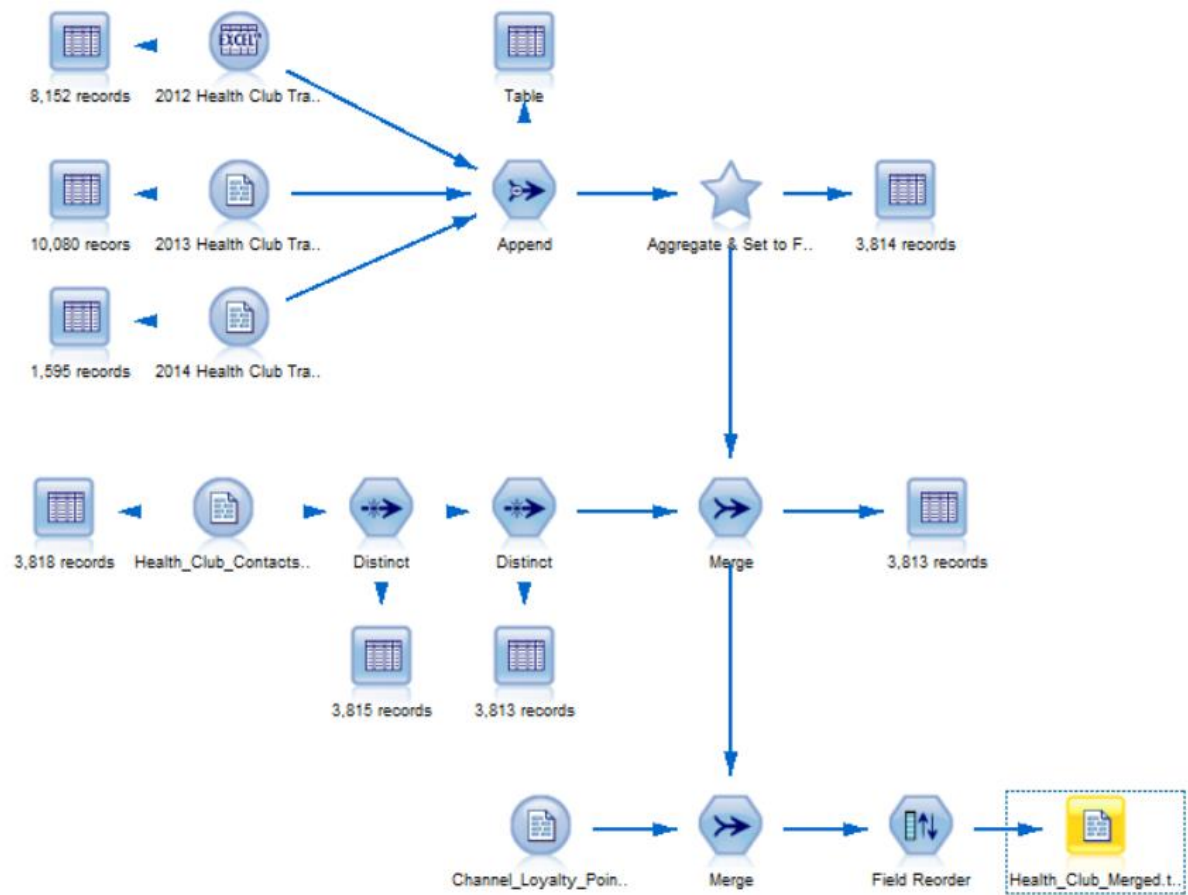


Figure 5.31 The completed Section 5 stream

Section 6: Changing and Creating Data



- The Reclassify Node
- The Filler Node
- The Derive Node

In this section, we will look at procedures that help us to transform and create fields. These procedures allow us to do everything from fixing data inputting errors and simplifying categories to calculating with variables and creating new variables based on formulae or Modeler functions.

The Reclassify Node



We begin this section by taking a look at the Reclassify node. The Reclassify node is used to recode the values within categorical variables. For instance, it can be used to reduce the number of categories in a field by regrouping a list of products into a simpler set of categories or by creating a 'miscellaneous' group for rarely occurring values. As we shall see, it can also be used to help with data errors created by inconsistent coding. Furthermore, the procedure is flexible:

- It can be used to create new fields or overwrite existing fields
- It can be used against single or multiple fields

To introduce the procedure, open the following Modeler stream file:

Section 6 Start.str

The stream contains a new data file ('Telco_Retention_Section_6.txt') which consists of a mixture of categorical and continuous variables. To view the file:

Run the Audit procedure attached to the Source node

Figure 6.1 shows the output from the Data Audit node. It also shows the distribution chart for the variable 'Gender'. We can immediately see that field contains many different values for 'Male' and 'Female'. This kind of issue is precisely the sort of problem that the Reclassify node helps to address.

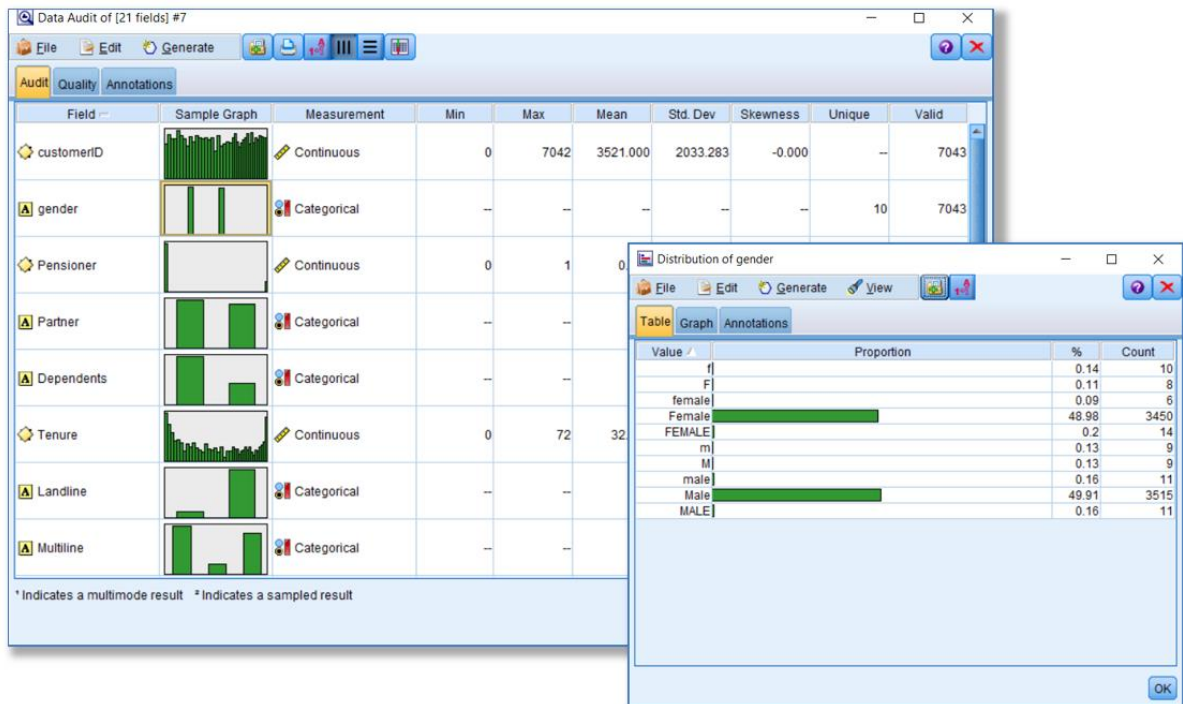


Figure 6.1 Data Audit procedure showing that the field 'Gender' has several inconsistencies

Using the Reclassify node, we can overwrite the existing field with the correct values. To demonstrate this, from the Field Ops palette:

Add the Reclassify Node to the source node in the stream

Figure 6.2 shows the Reclassify Node added to the stream.

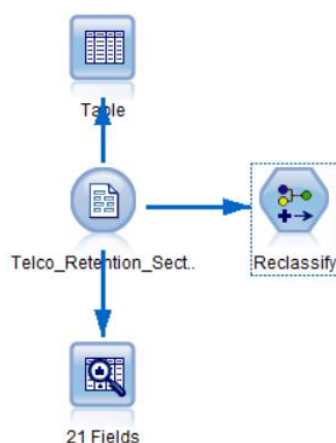


Figure 6.2 The Reclassify Node added to the current stream

Figure 6.3 shows the contents of the of the Reclassify node when it is edited. You may notice from the screenshot, that the drop-down list of the categorical fields under the section marked 'Reclassify field' indicates that none of the fields have been fully instantiated. Just like

the Set to Flag node that we encountered earlier, the Reclassify Node requires that the data are fully instantiated so that it can scan the values within the field(s) to be reclassified. At this point, attempting to reclassify the *partially* instantiated Gender field will result in the error message shown in figure 6.3.

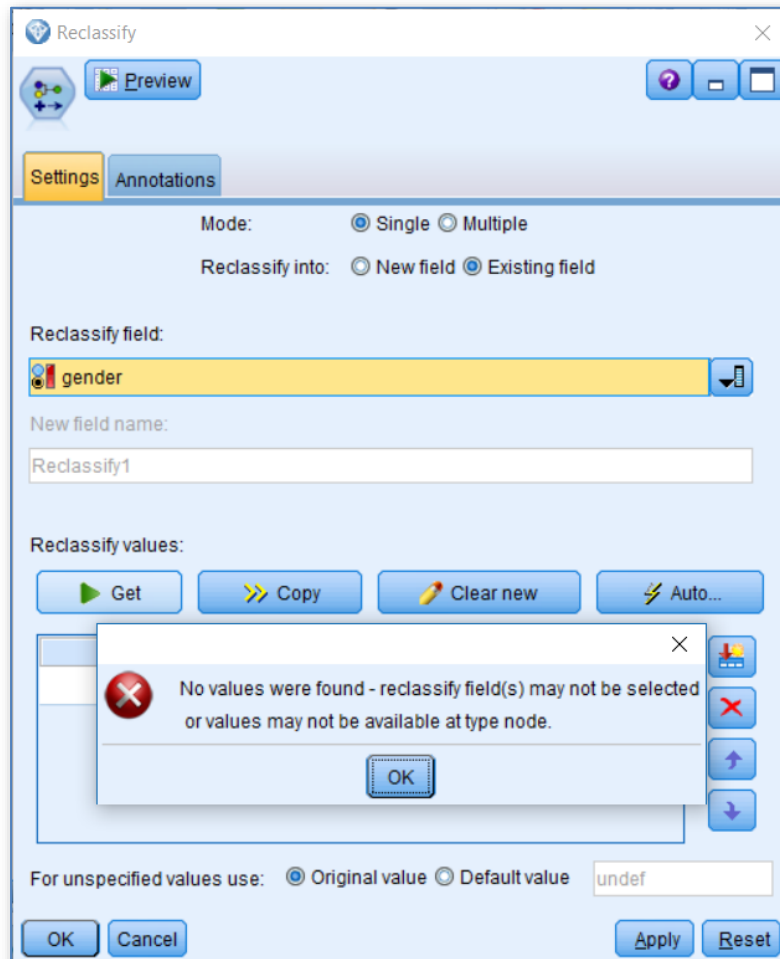


Figure 6.3 Error message generated when using the Reclassify Node with partially instantiated categorical field

We need to instantiate the data in the source node to allow the Reclassify Node to read the values in the Gender field.

Edit the Source node and click the 'Read Values' button within the Type tab to fully instantiate the data

Figure 6.4 shows the Type tab within the Source node before and after full instantiation has occurred.

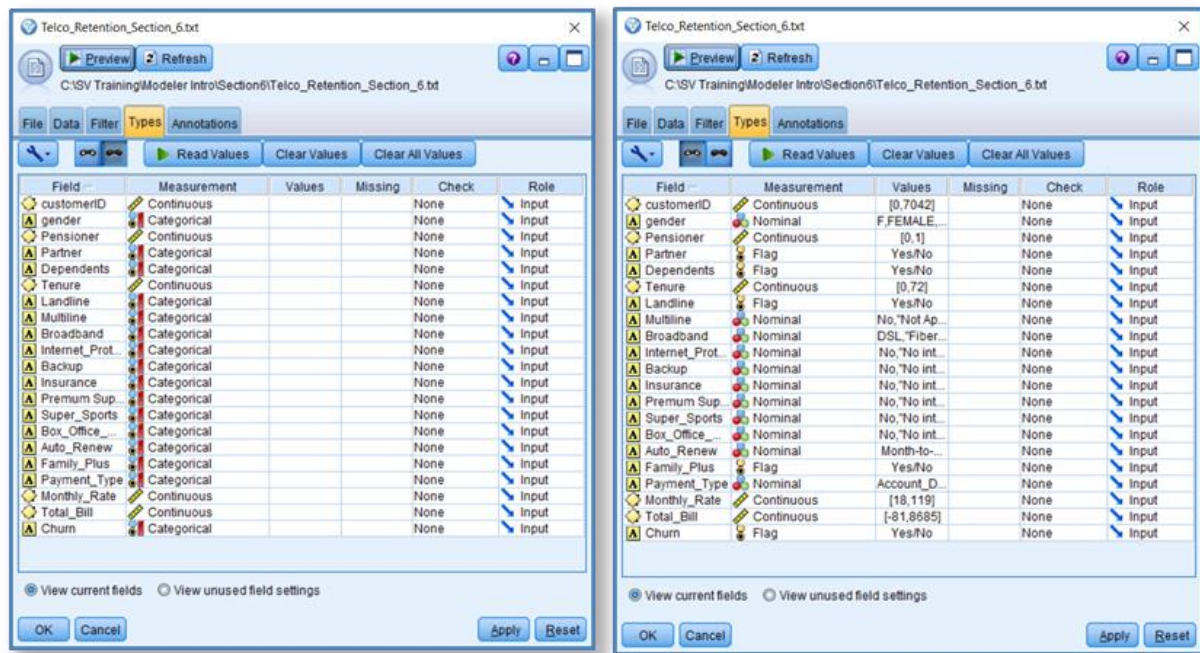


Figure 6.4 The Type tab within the Source node showing a before and after view of partial and full instantiation of the fields within the data file

Returning to the Reclassify node, within the section marked 'Reclassify into', click the radio button:

Existing field

We can now select the field 'Gender' from the drop-down menu and in order to make the procedure retrieve all the values within the field, in the 'Reclassify values' section, click the button marked:

Get

Immediately, the node retrieves and displays all the current values within the Gender field. Figure 6.5 shows the results of this.

Reclassify2

Preview

Settings Annotations

Mode: ☒ Single ☐ Multiple

Reclassify into: ☐ New field ☒ Existing field

Reclassify field:
gender

New field name:
Reclassify2

Reclassify values:

Get Copy Clear new Auto...

Original value	New value
FEMALE	
Female	
M	
MALE	
Male	

For unspecified values use: ☒ Original value ☐ Default value undef

OK Cancel Apply Reset

Figure 6.5 The Reclassify node after the values for the field 'Gender' have been retrieved for reclassifying

If we only wished to reclassify one or two values, we could use the 'Copy' button to copy across all the current values in the variable to the column marked 'New value' and then simply edit the values that needed to be changed. In this case however, we will simply enter the values as we wish them to be displayed in the cells under the 'New value' column. Within the first cell (corresponding to the value 'F') type:

Female

If we now click on the next row down in the 'New value' column, we can pick the previously entered new value from a drop-down list. In other words, the procedure 'remembers' the values that you enter each time into the 'New value' column. From the drop-down list button select:

Female

We can continue doing this of course until we encounter the cells corresponding to the values for male respondents. In which case, we need only edit the blank cell and enter the value:

Male

Figure 6.6 shows this process in action.

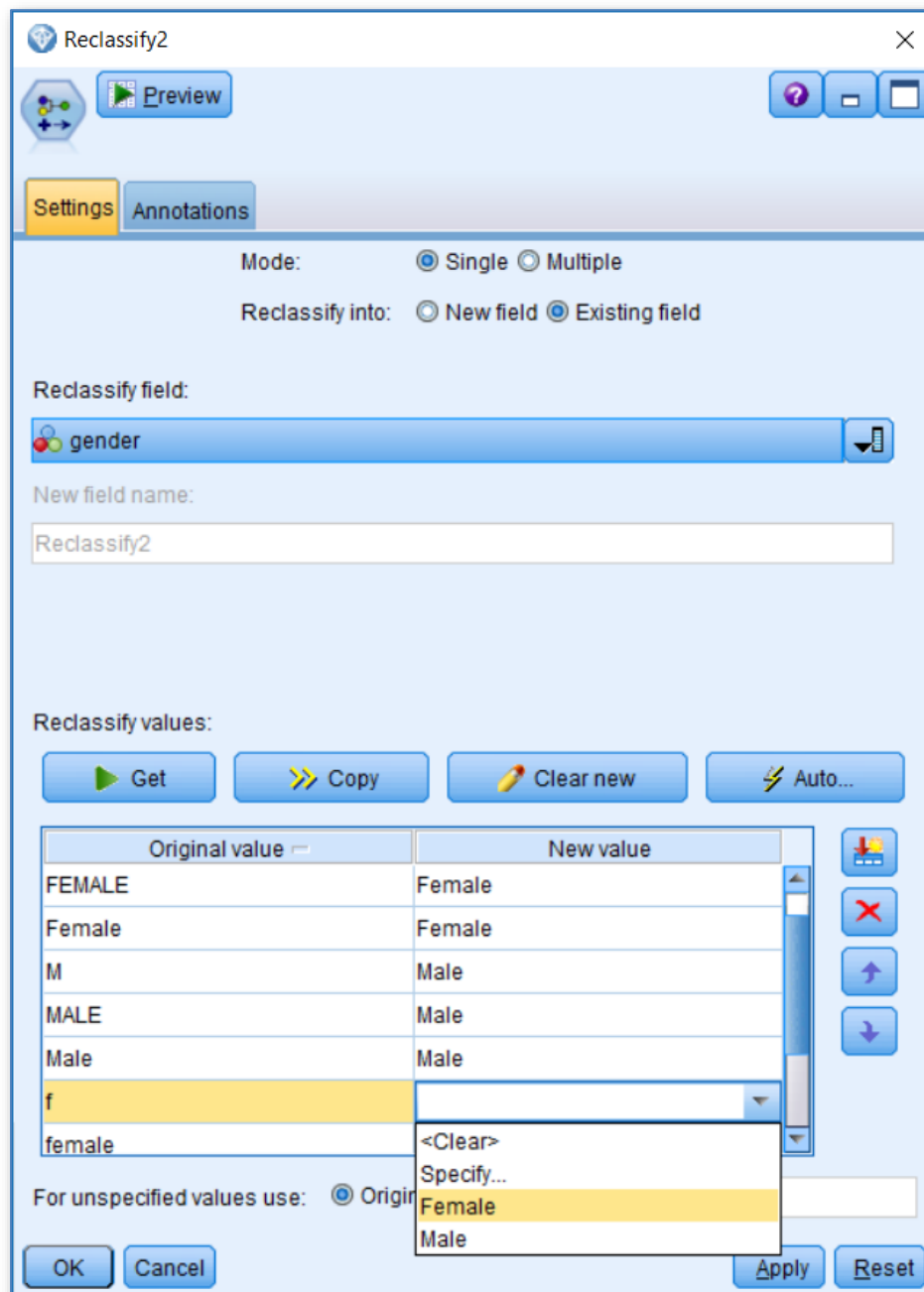


Figure 6.6 Choosing new values within the Reclassify node from a list of previously entered values

Having completed the process of assigning consistent values for the variable Gender, we can attach a Distribution node to the edited Reclassify node and request a chart of the newly reclassified field. Figures 6.7 and 6.8 show the updated stream and the Distribution chart output of the updated Gender field.

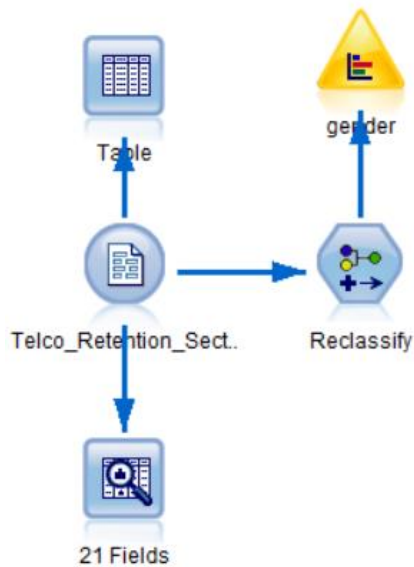


Figure 6.7 Attaching a Distribution node to the updated Gender field

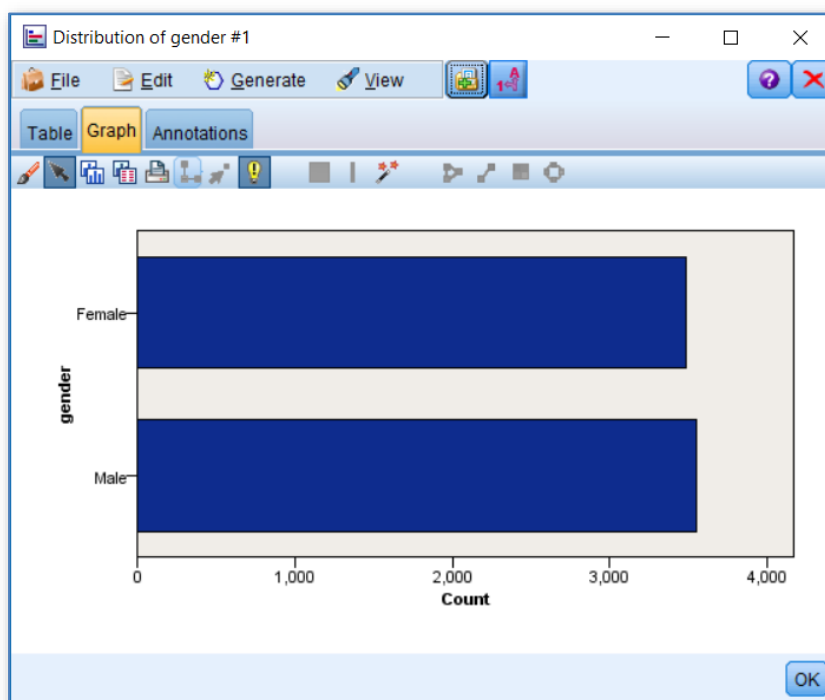


Figure 6.8 The Gender field cleaned by the Reclassify procedure and displayed in a Distribution chart

Another way in which we can reclassify data is by using the Generate menu to create a Reclassify node for us. To illustrate this, we can edit the existing Distribution chart node so that it creates graph of a different variable.

Edit the Distribution chart node replacing the field 'Gender' with the field 'Payment_Type' and run the procedure

Figure 6.9 shows the resultant output.

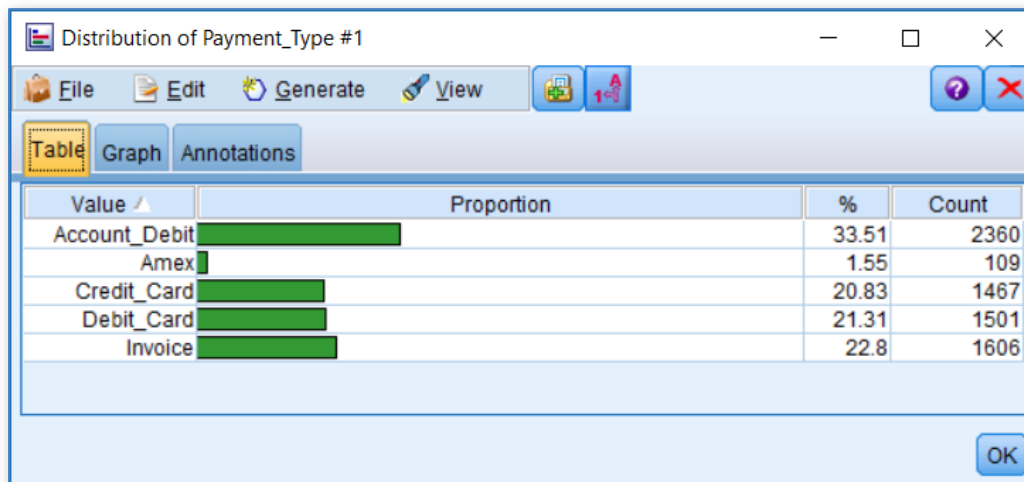


Figure 6.9 Distribution chart of the variable 'Payment_Type'

We can directly interact with the values displayed on the Table tab of the chart output. As shown in figure 6.10:

Click and drag the mouse cursor to highlight the categories 'Amex', 'Credit_Card' and 'Debit_Card'

Right-click on the selection to generate a pop-up menu

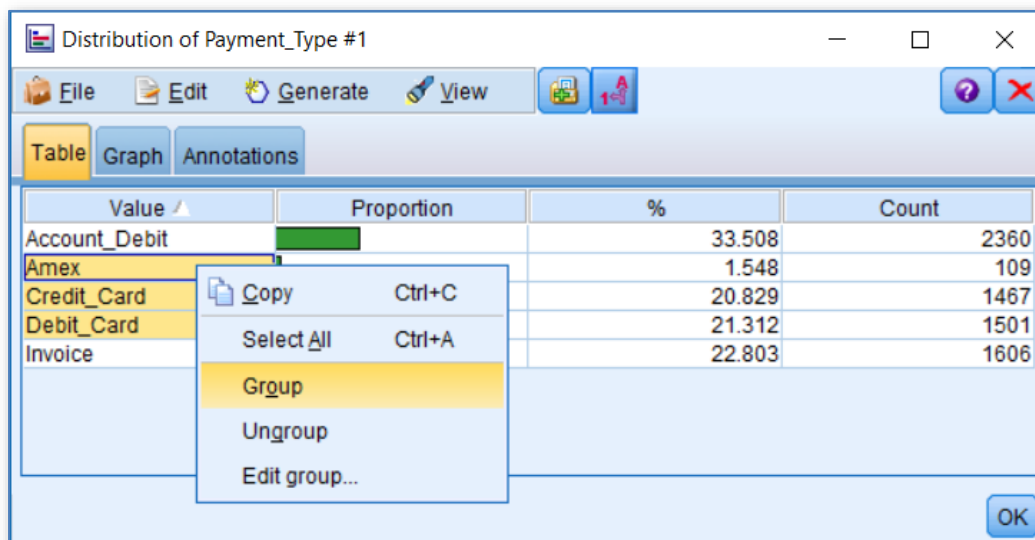


Figure 6.10 Selecting values to create a new grouped category

From the pop-up menu, click:

Group

The three categories are now grouped together to create a new value ('Group1'). To change the name of this set of grouped values, once again:

Right-click on the selection to generate a pop-up menu

Figure 6.11 shows that we can now edit the group name.

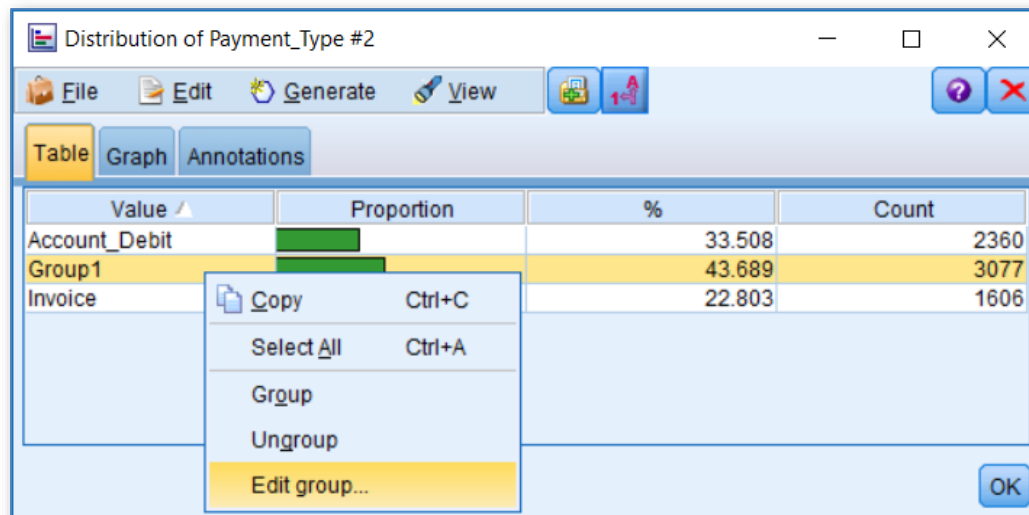


Figure 6.11 Editing a group name within the Distribution Table output

To assign a new name to the grouping, click:

Edit group...

An 'Edit group' sub-dialog is generated. To continue:

Edit the group name so that it changed to 'Card_Payment'

Figure 6.12 illustrates this process.

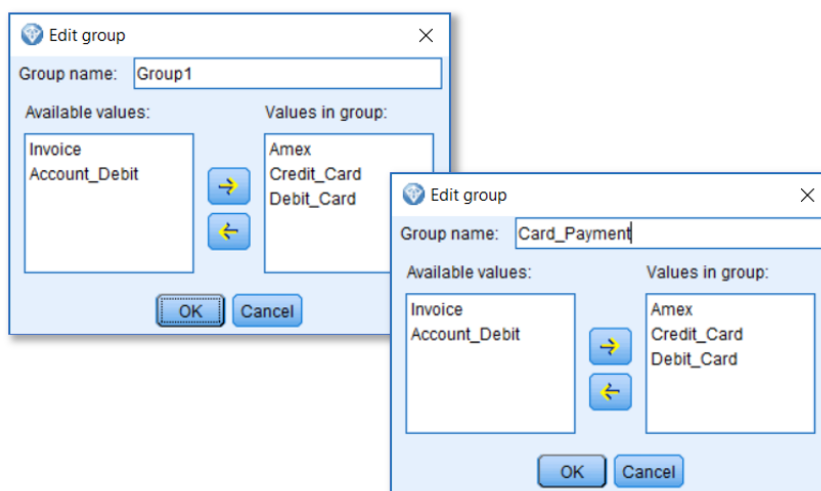


Figure 6.12 Editing the group name

To apply the new group name, click:

OK

We are now returned to the Distribution output view and we can use the 'Generate' menu to create a new Reclassify node based on the grouping we have created in the chart. From the main menu in the Distribution output, click:

Generate

Reclassify node (Groups)

Figure 6.13 shows this process.

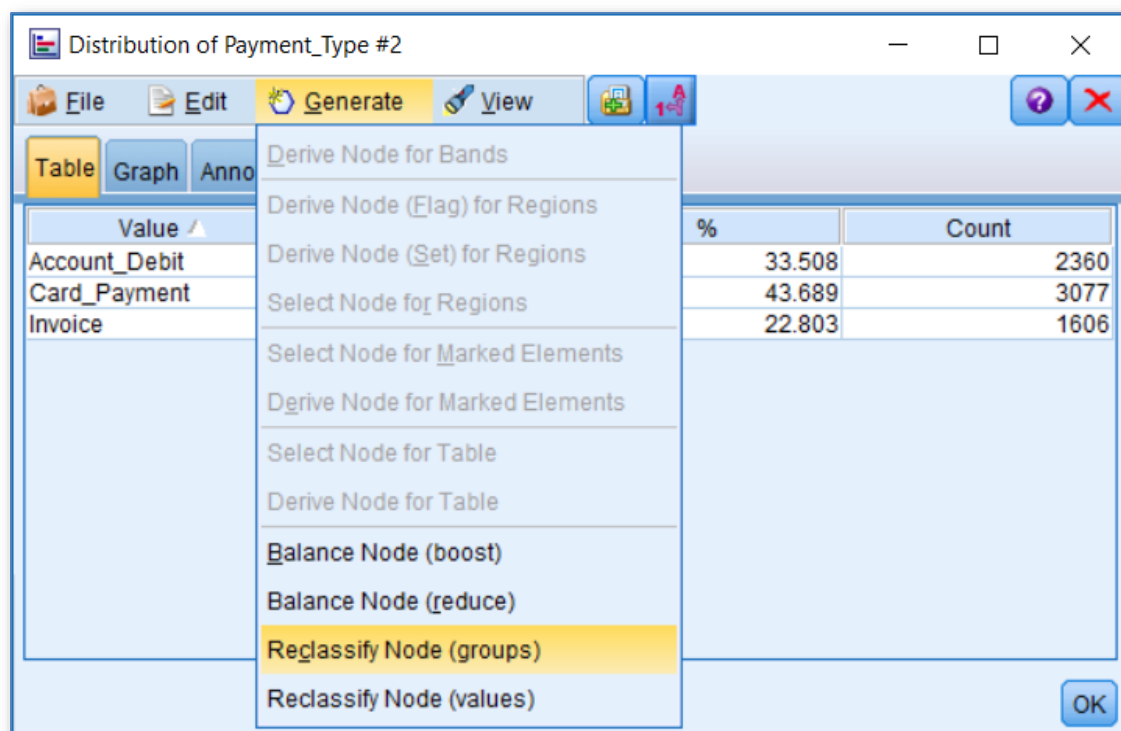


Figure 6.13 Using the Generate menu to create a new Reclassify node

A new Reclassify node is added to the stream canvas (in the top left-hand corner). To apply the grouping back to the data:

Connect this node to the previous Reclassify node and edit it

We can now see the familiar interface of the Reclassify node. We can also see that all three individual card types have been assigned the new value 'Card_Payment'. The remaining payment methods are left untouched by virtue of the radio button near the bottom of the dialog indicating that 'For unspecified values' the procedure will use the 'Original value'.

On this occasion, we can use the Reclassify node to create a new field.

Edit the description in the box headed 'New field name:' so that the new field is called 'Payment_Type_3Grp'

Figure 6.14 shows the edited Reclassify node with the new field name entered.

(generated)

Preview

Settings Annotations

Mode: ☒ Single ☐ Multiple

Reclassify into: ☒ New field ☐ Existing field

Reclassify field:

Payment_Type

New field name:

Payment_Type_3Grp

Reclassify values:

Get Copy Clear new Auto...

Original value	New value
Amex	Card_Payment
Credit_Card	Card_Payment
Debit_Card	Card_Payment

For unspecified values use: ☒ Original value ☐ Default value undef

OK Cancel Apply Reset

Figure 6.14 Using the generated Reclassify node to create a new version of the variable 'Payment_Type'

To apply the procedure, click:

OK

To check that it has worked:

Attach another Distribution node to the end of the stream and request a chart of 'Payment_Type_3Grp'.

Figure 6.15 shows the resulting table and chart.

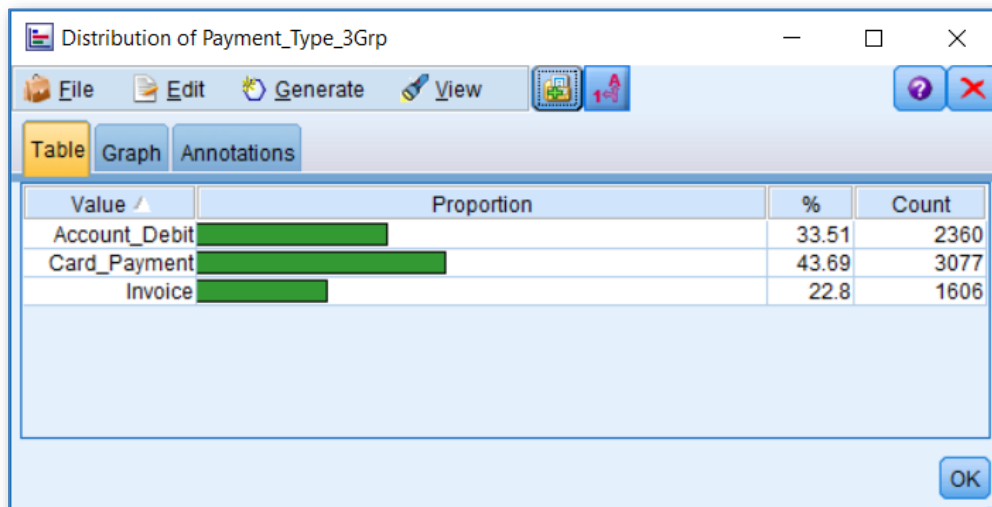
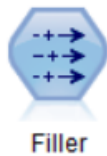


Figure 6.15 The results of creating a new reclassified field via the Distribution node and the Generate menu

The Filler Node



The Filler node is used to replace existing field values or to change the storage type. Unlike the Reclassify procedure, the Filler node cannot be used to create *new* fields. The Filler node is commonly used to replace missing or invalid data values (blanks or nulls or both). The Filler node can replace or alter values on a conditional or unconditional basis. The node can be applied in the following ways.

- It can *only* be used to *overwrite* existing fields
- It can be used against single or multiple fields

The Filler node often makes good use of Modeler's CLEM expression language including some of Modeler's special system variables, such as:

@FIELD	When an expression is to be applied to several fields at once, the @FIELD system variable represents each field in turn
@BLANK	Applies to user-defined missing values
@NULL	Applies to numeric system missing values (\$null\$)

Returning to the Data Audit output that we saw earlier, we can see that one of the fields has unusual values. Figure 6.16 highlights the minimum values for the variable 'Total_Bill'. Because a small number of customers cancelled their subscriptions within the 30-day money-back guarantee period, the subsequent refunds show up as negative values in their bill. We can use the Filler node to replace these values with a number that reflects how much revenue the company really got from them (zero).

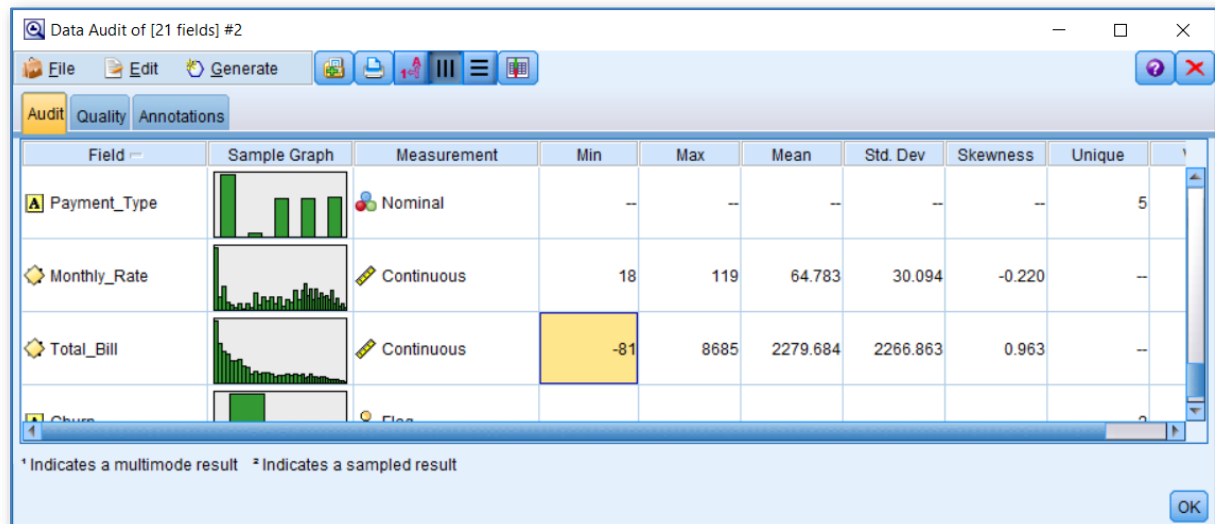


Figure 6.16 The Data Audit output showing negative values for the variable 'Total_Bill'

From the Fields Ops palette, locate the Filler node:

Place the node on the stream canvas and connect it to the last Reclassify node

Figure 6.17 shows this.

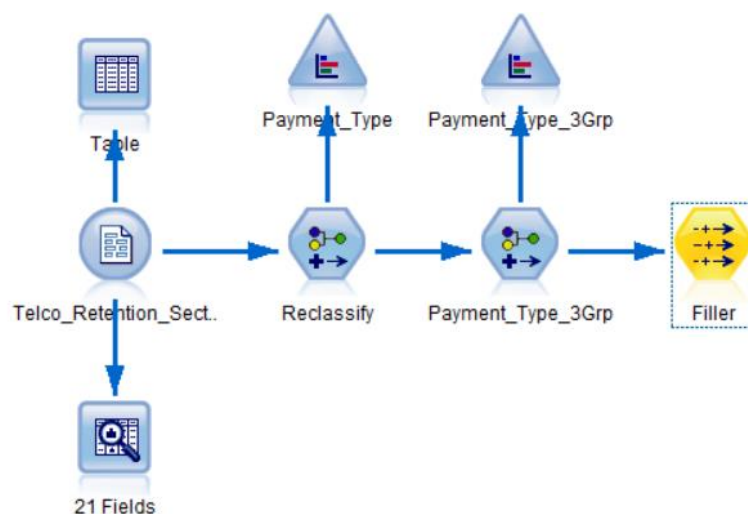


Figure 6.17 The Filler node added to the existing stream

Edit the connected Filler node as shown in Figure 6.18.

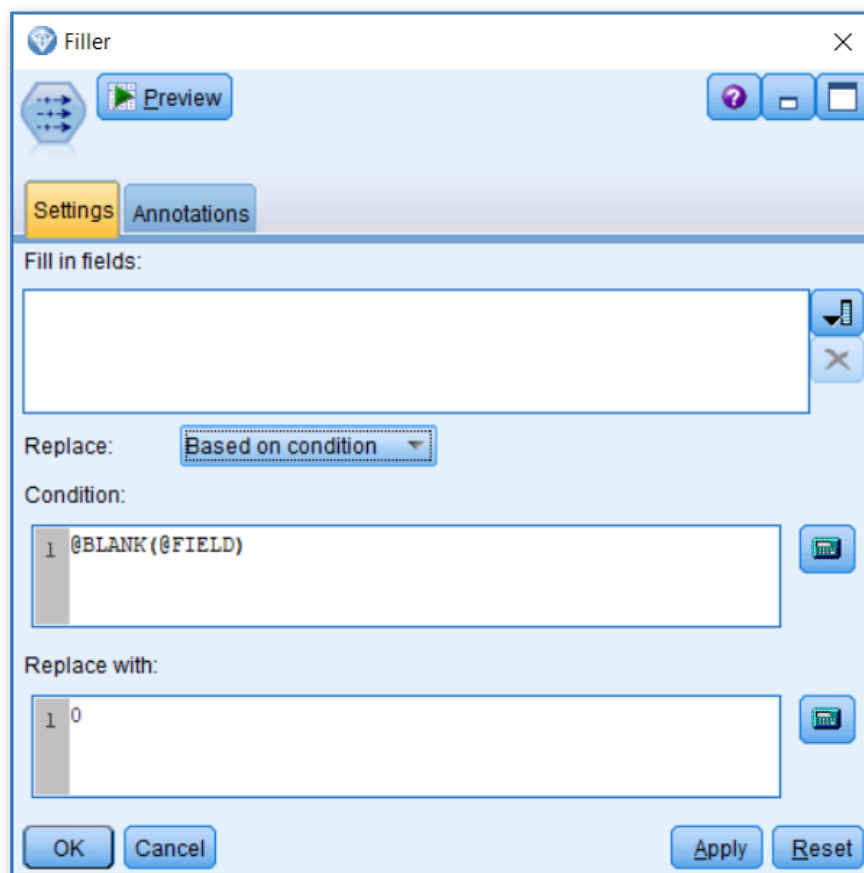


Figure 6.18 The default Filler node settings

The default settings for the Filler node assume that the user intends to replace defined missing values (note the @BLANK system variable in the 'Condition' box) for one or more fields (note the @FIELD system variable that the condition is to be applied to). We could use these settings if we simply returned to the Type tab within the Source node and defined all the negative values in the field Total_Bill as 'missing'. But perhaps it's easier simply to reference these values in a simple expression within the Filler node itself. Filler has four 'Replace' modes:

1. **Based on a condition** – This is the default and assumes you will create an expression to trigger the replacement
2. **Always** – This 'greys out' the condition box as it will replace the values unconditionally
3. **Blank values** - This 'greys out' the condition box and targets user-defined blank values. It doesn't allow you to edit the condition.
4. **Null values** - This 'greys out' the condition box and targets numeric system-missing values (\$null\$). It doesn't allow you to edit the condition.

5. **Blank and null values** - This 'greys out' the condition box and targets both user-defined blank values and numeric system-missing values (\$null\$). It doesn't allow you to edit the condition

At this stage, we can directly edit the existing expression in the 'Condition' box, however we can use the opportunity to view the Modeler expression builder. To view the expression builder, notice the 'calculator' icon next to the Condition box and click:



Figure 6.19 shows an annotated view of the Expression Builder window within the Filler node.

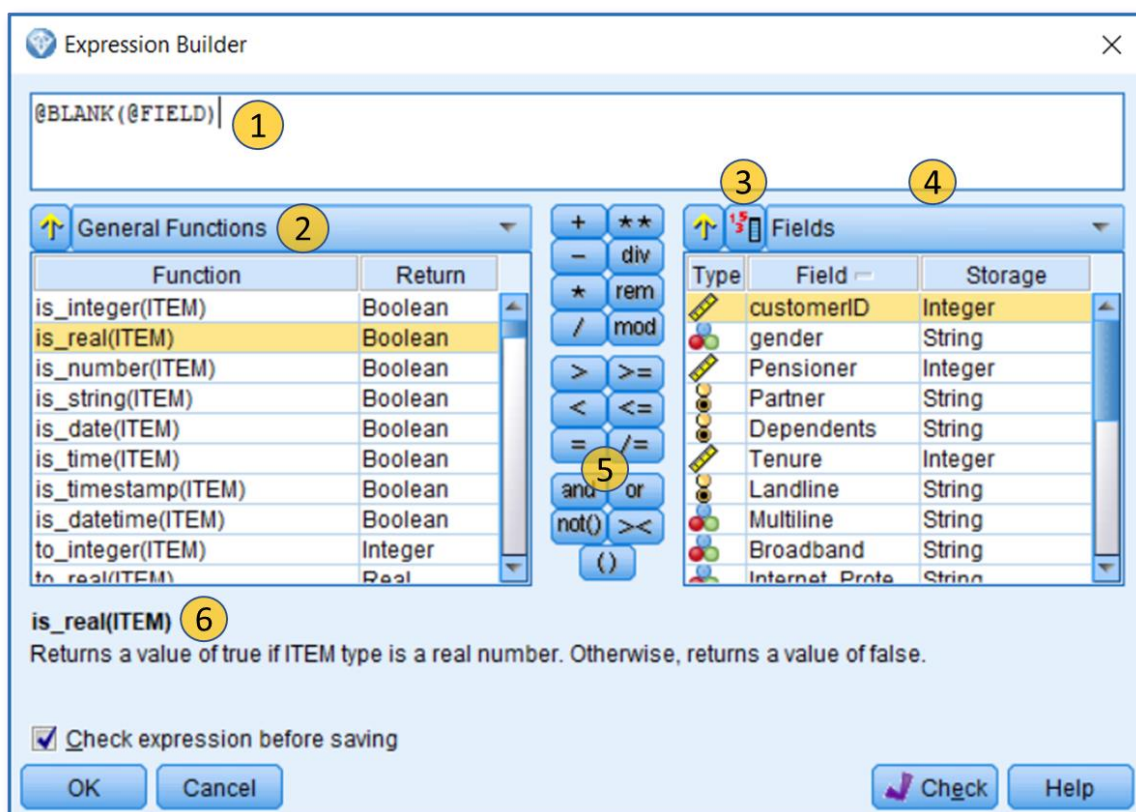


Figure 6.19 The annotated Expression Builder

1. **The Expression Editor** - The current default expression which can be edited, erased or replaced within this editing box.
2. **Function Library** - A drop-down list of Modeler's CLEM expression language functions. These are extensive in number and application. Typical functions include routines for dealing with string fields, dates, random numbers, mathematical and statistical routines and field type conversion functions.

3. **Select from field values** - A useful feature of the Expression Builder is the ability to view the minimum and maximum values within continuous fields or to choose individual values within categorical fields. The data must be fully instantiated for this to work.
4. **Drop-down selector** – Primarily used to select the existing fields, this drop-down selector can also be used to filter the list to include advanced elements such as parameters, global summary measures and multi-response sets.
5. **CLEM Operators** – This is the interface to the CLEM language operators (see the table below)
6. **Function Definition** – Provides a short description as to the how the currently selected function is applied.

+	Plus (add two numbers)	**	Raise x to the power of
-	Minus (subtract two numbers)	div	Return the quotient of dividing x by y
*	Multiply two numbers	rem	Return the remainder of dividing x by y
/	Divide two numbers	mod	Return the result of x modulo y
>	Greater than	>=	Greater than equal to
<	Less than	<=	Less than or equal to
=	Equal to	/=	Not equal to
and	And (expression 1 and expression 2 is true)	><	Concatenate
or	Or (either expression 1 or expression 2 is true)	()	Enclose selection in parentheses
Not()	Expression is not true		

Figure 6.20 CLEM Expression Language operators

To use the Filler node to replace the negative values in the field 'Total_Bill', we need only edit the existing condition within the expression builder. To do so, edit the expression so it reads:

Total_Bill < 0

Figure 6.21 shows the edited expression in the Expression Builder.

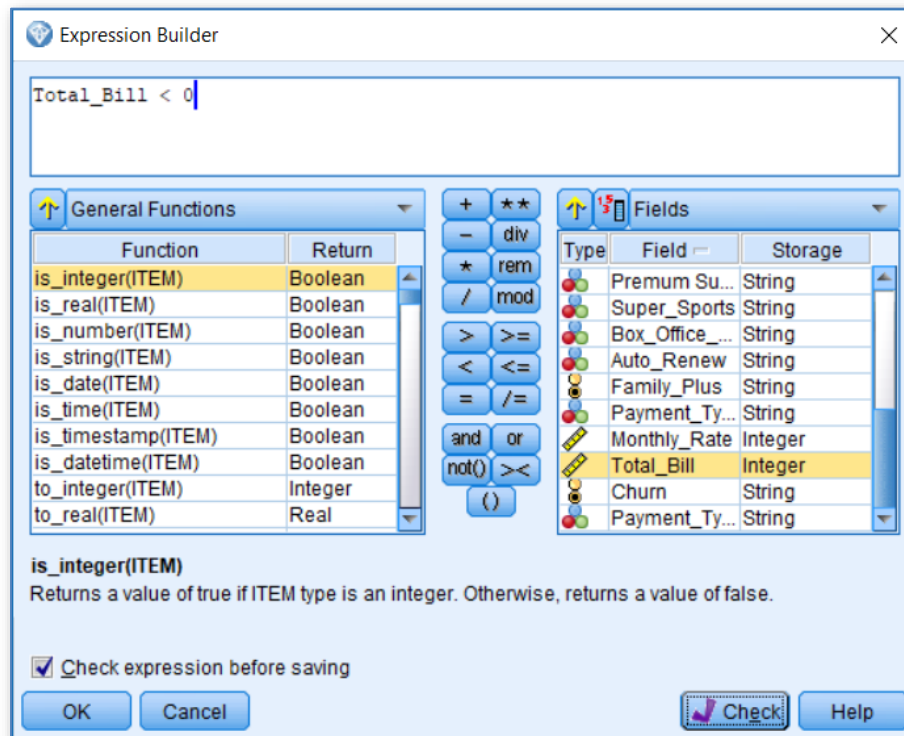
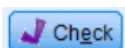


Figure 6.21 Edited condition in the Expression Builder

To check if the expression syntax makes sense, click:



The text in the expression box changes from red to black to indicate that the syntax is correct. To continue, click:

OK

We now are returned to the main Filler node dialog. It's worth pointing out that we could have used the condition '@FIELD < 0' in the Expression Builder. This would have been very useful if we needed to use this rule to replace values in *multiple* fields. We now need only to specify that we are filling in the values in the field 'Total_Bill'. Next to the box at the top, headed 'Fill in fields', click the field picker button:



From the field list, choose the field:

'Total_Bill'

The completed dialog should look like figure 6.22.

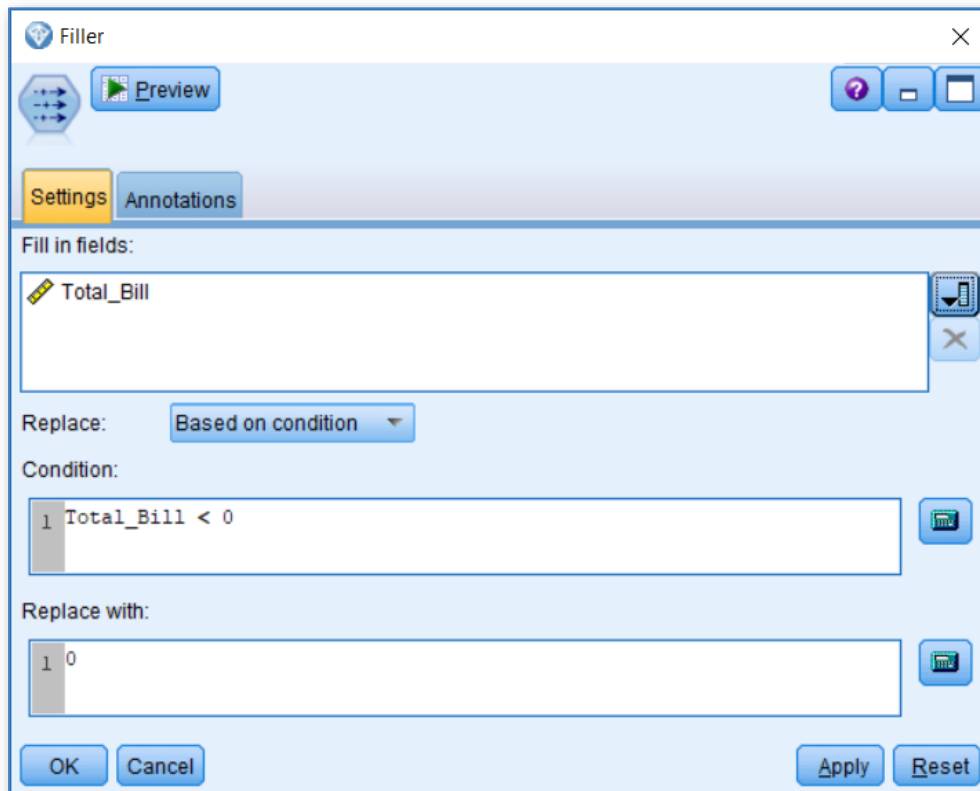


Figure 6.22 The completed Filler node - edited to replace negative values in the field 'Total_Bill' with a zero

To check if the procedure works, click:

OK

Copy and paste the existing Data Audit node and attach it to the Filler node

Figure 6.23 shows the updated stream and figure 6.24 shows the output from the second Data Audit node. You can see that the minimum value for the field 'Total_Bill' is now zero.

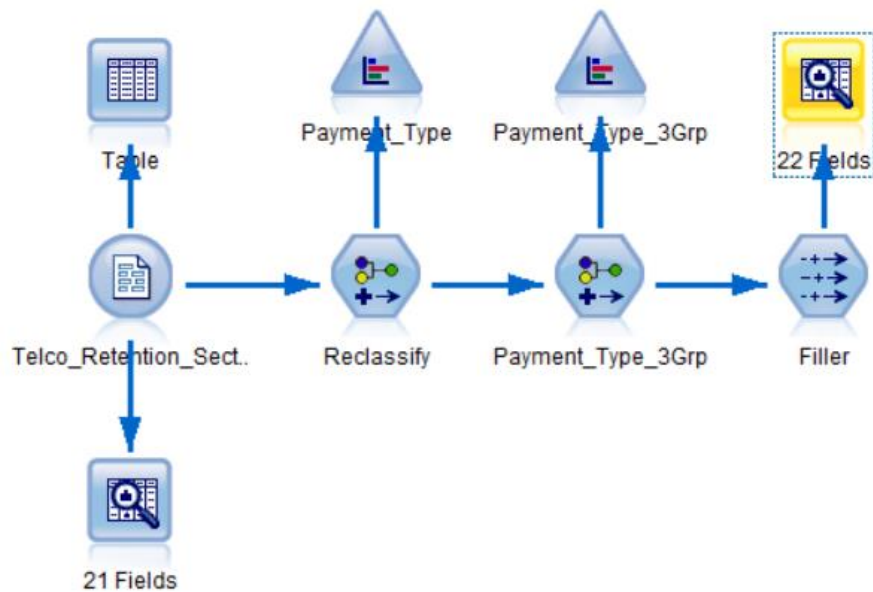


Figure 6.23 The updated stream with the Filler node and a second Data Audit node attached

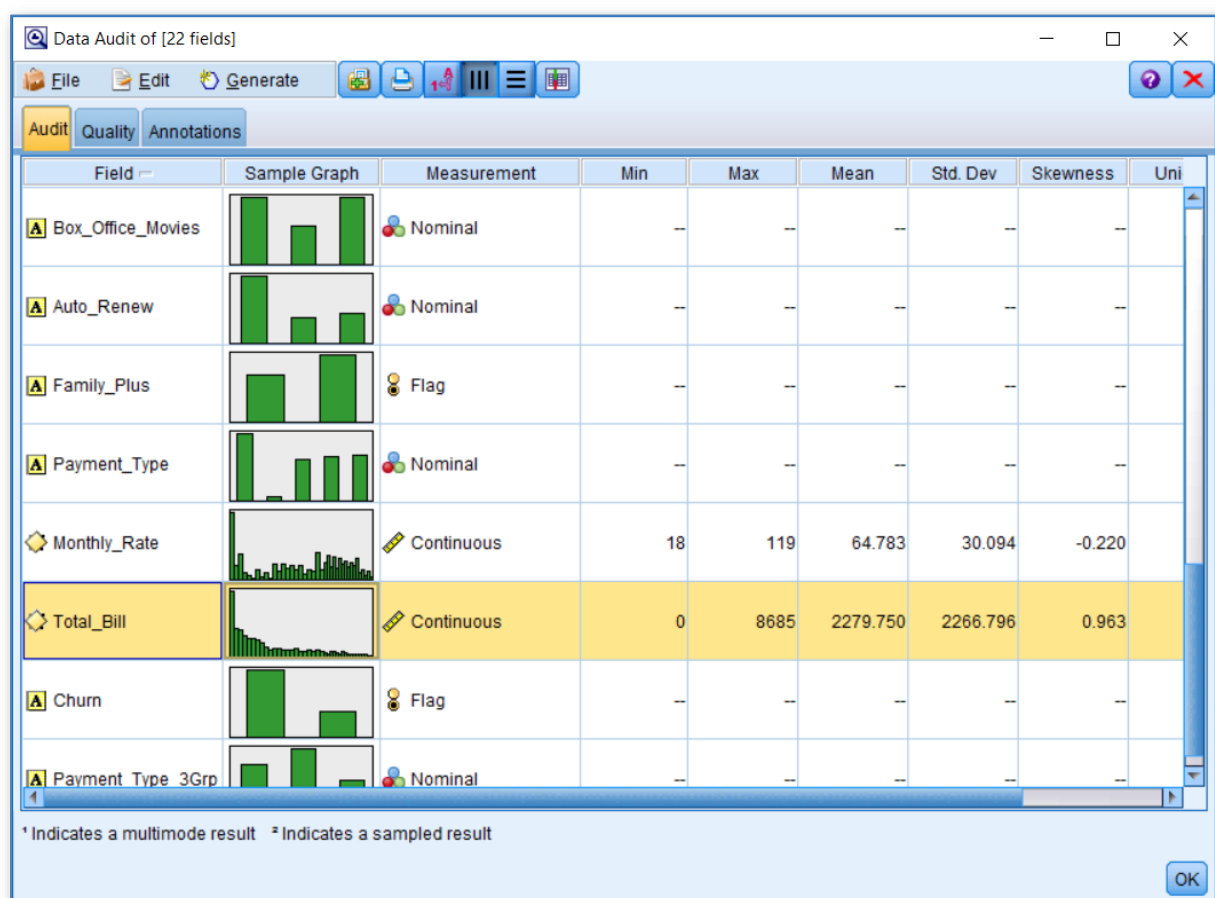


Figure 6.24 The Data Audit output showing that the minimum value for the variable 'Total_Bill' is now zero

The Derive Node



One of the most versatile ways to create new fields within Modeler is by using the Derive node. The Derive node is a general-purpose node used for an extremely broad range of tasks. By employing the built-in CLEM expression functions, the Derive node can be used for the following tasks and many more:

- Using String search and replace functions to clean postcode data that contain inconsistencies
- Creating an ascending count of transactions for each customer
- Calculating elapsed time between date fields
- Applying complex formula to numeric data
- Identifying entities that are geographically close to one another

The node can be applied in the following ways.

- It can *only* be used to create *new* fields, *not* to overwrite existing ones
- It can be used against single or multiple fields

Part of the reason that the Derive node is so versatile is that it has six different modes of operation for creating new fields. Figures 6.25a to 6.25c show screenshots of the different Derive modes.

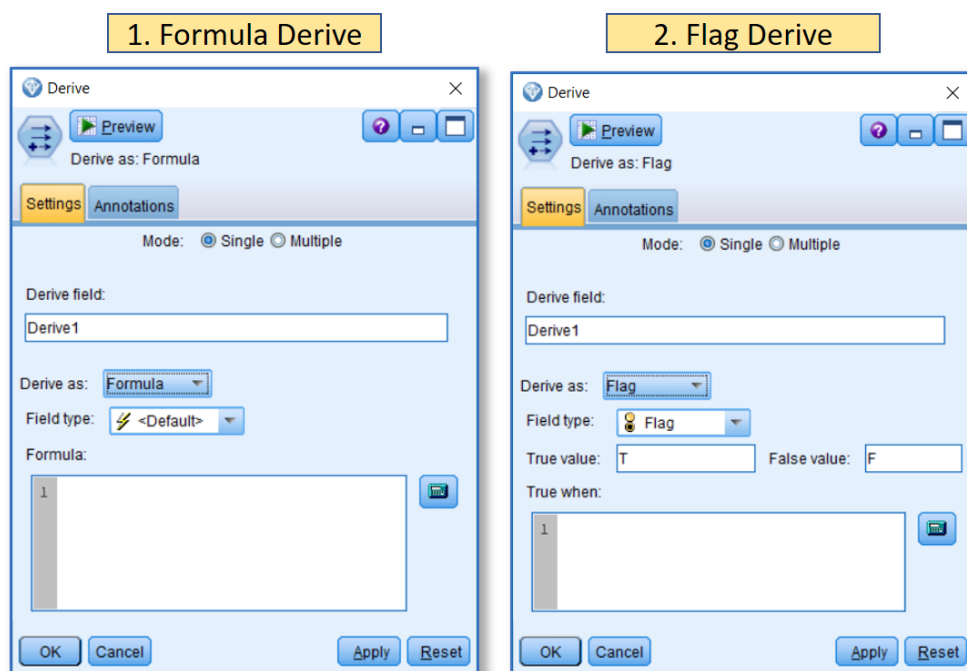


Figure 6.25a Formula and Flag Derive modes

1. **Formula Derive mode** – This is a simple mode where a formula or function is applied to the data without any conditions.
2. **Flag Derive mode** – This is also a simple Boolean mode where a binary (flag) outcome is derived based on whether or not a statement or condition is true.

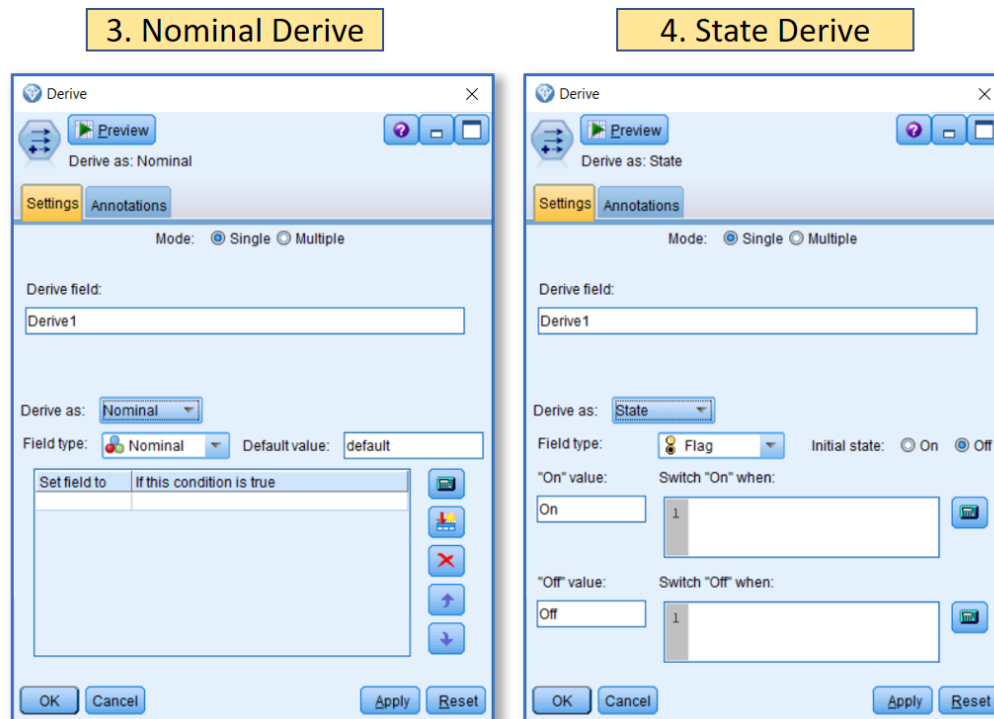


Figure 6.25b Nominal and State Derive modes

3. **Nominal Derive mode** – This mode is used to derive Nominal (or Ordinal) fields based on a series of conditions. It can be used in the same way as the Reclassify node or it can be used to generate a series of specific values based on more complex formula. A common application of the Nominal Derive mode is to create categories (such as 'Low', 'Medium' and 'High') based on value ranges.
4. **State Derive mode** – This is a special mode within the Derive procedure that enables a binary outcome to be generated based on two separate conditions or formulae. It is useful when flagging issues or marking interesting records where the data that has been collected over a continuous period. An example might be creating an alert when a mortgage payment is missed. In which case, the second specified condition could be used to ensure that the alert status is not 'switched off' until three consecutive payments are successfully made.

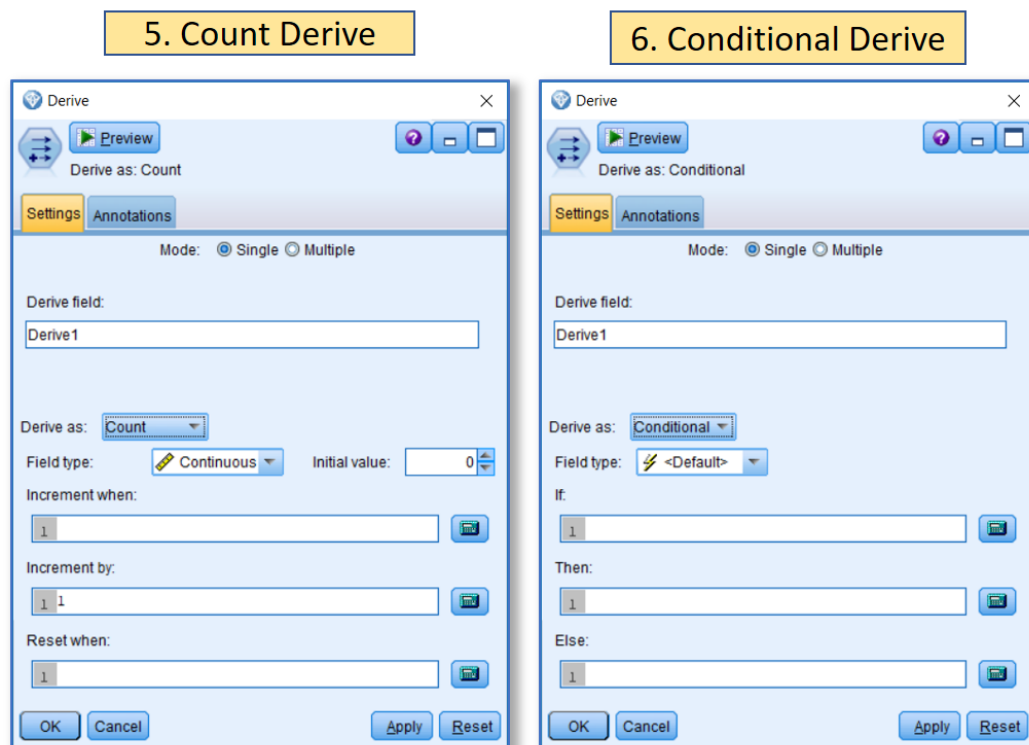


Figure 6.25c Count and Conditional Derive modes

5. **Count Derive mode** – This is another special mode that, in this case, is used to create an incremental count. It is designed to be used with data collected over time. For example, it may be used to count how many times an account is overdrawn or to incrementally record how many errors have been generated by a remote system or asset.
6. **Conditional Derive mode** – This is a commonly-used mode where a formula is applied but only if it meets a certain condition. The mode is comprised of three separate conditions in the form of 'If', 'Then' and 'Else'. The values of 'Then' or 'Else' could be based on constants or formulae.

Let's take a look at how we can use the Derive mode to apply a simple formula to the dataset in order to calculate the average monthly bill for each customer. The Derive node can be found within the Field Ops palette. To attach it to the stream:

Click the already edited Filler node to select it in the stream

Double-click the Derive node within the Field Ops palette to automatically attach it to the Filler node

Figure 6.26 shows the Derive node added to the existing stream.

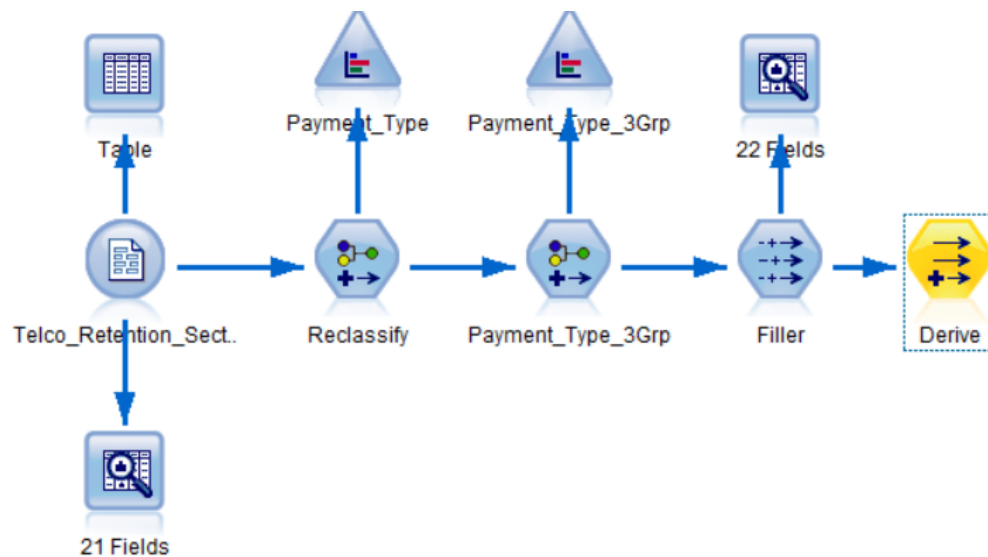


Figure 6.26 The Derive Node added to the stream and attached to the Filler node

Edit the Derive node to view the default mode

Change the default name for the field to be derived from 'Derive1' to 'Average_Monthly_Bill'

In the same dialog, change the Field type from Default to Continuous

Figure 6.27 shows the first part of this process

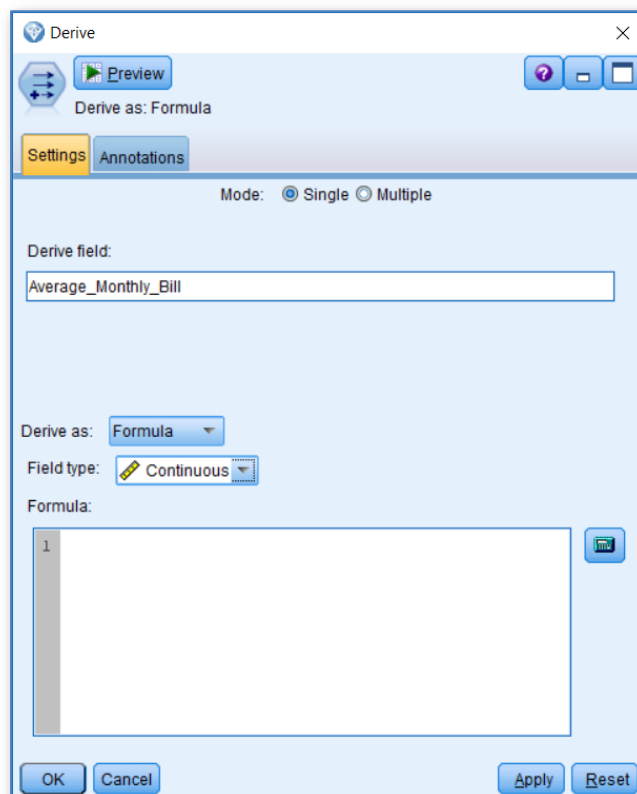


Figure 6.27 The edited Derive node in 'formula' mode - a field name has been specified and the field type has been set as continuous

We can now enter the formula to derive this new field. We could in fact enter this directly into the Formula box or use the Expression Builder. To remind ourselves what the fields are called in the file, click:



In the expression box, create the following formula:

Total_Bill/Tenure

As the field 'Tenure' is measured in months, we can assume that this will return a value that equates to the average monthly bill. Figure 6.28 shows the formula entered in the Expression Builder.

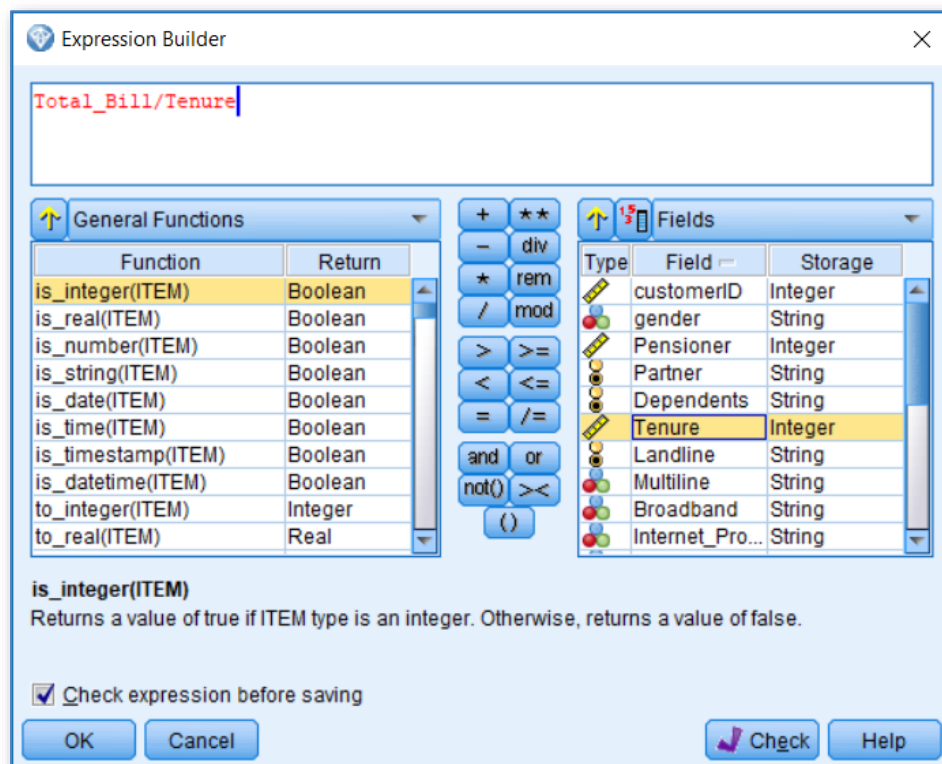


Figure 6.28 The Derive node formula entered into the expression builder

To complete the process, click:

OK

OK

To view a summary of the newly derived field:

Copy and paste the last Data Audit node and attach it to the Derive node

Figure 6.29 shows the updated stream and figure 6.30 shows the output from the third Data Audit node output. You can see that the summary statistics for the field 'Average_Monthly_Bill' appear in the last row of the output window.

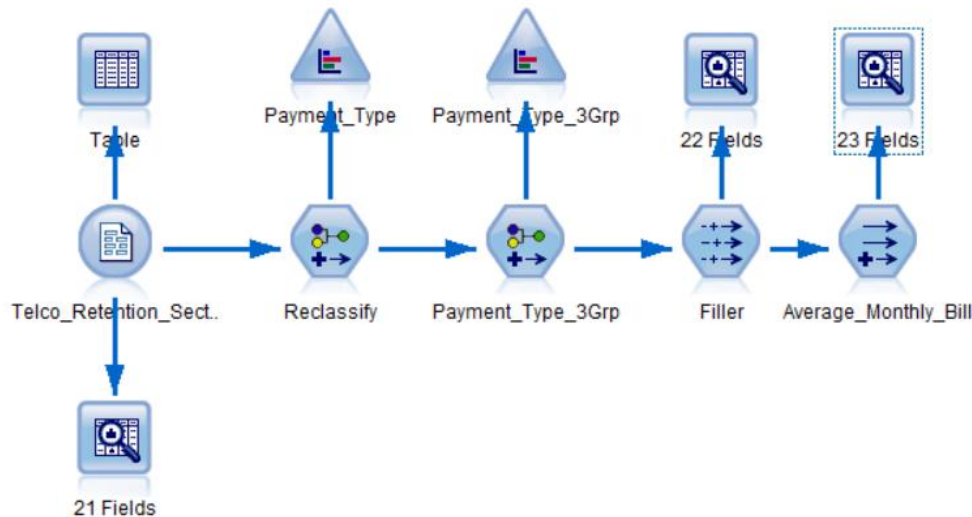


Figure 6.29 The updated stream with the Derive node attached

Data Audit of [23 fields]									
Field	Sample Graph	Measurement	Min	Max	Mean	Std. Dev	Skewness	Unique	Valid
Auto_Renew		Nominal	--	--	--	--	--	3	7043
Family_Plus		Flag	--	--	--	--	--	2	7043
Payment_Type		Nominal	--	--	--	--	--	5	7043
Monthly_Rate		Continuous	18	119	64.783	30.094	-0.220	--	7043
Total_Bill		Continuous	0	8685	2279.750	2266.796	0.963	--	7043
Churn		Flag	--	--	--	--	--	2	7043
Payment_Type_3Grp		Nominal	--	--	--	--	--	3	7043
Average_Monthly_Bill		Continuous	0.000	121.500	64.799	30.193	-0.212	--	7032

* Indicates a multimode result * Indicates a sampled result

Figure 6.30 The Data Audit output showing summary values for the new field 'Average_Monthly_Bill'

In our next example, we will switch to the 'Flag' mode and create a new field that identifies the most loyal customers in the data file.

To add another Derive node:

Click on the last Derive node to select it

From the Field Ops palette:

Double-click on the Derive node to automatically add another to the stream

Edit the newly added Derive node

Within the Derive node dialog:

Edit the text in the derive field box so that the new field is called 'Very_Loyal'

Change the 'Derive as' mode from 'Formula' to 'Flag'

Specify the following condition in the 'True When' condition box:

Tenure >= 60

Figure 6.31 shows the completed Flag Derive node.

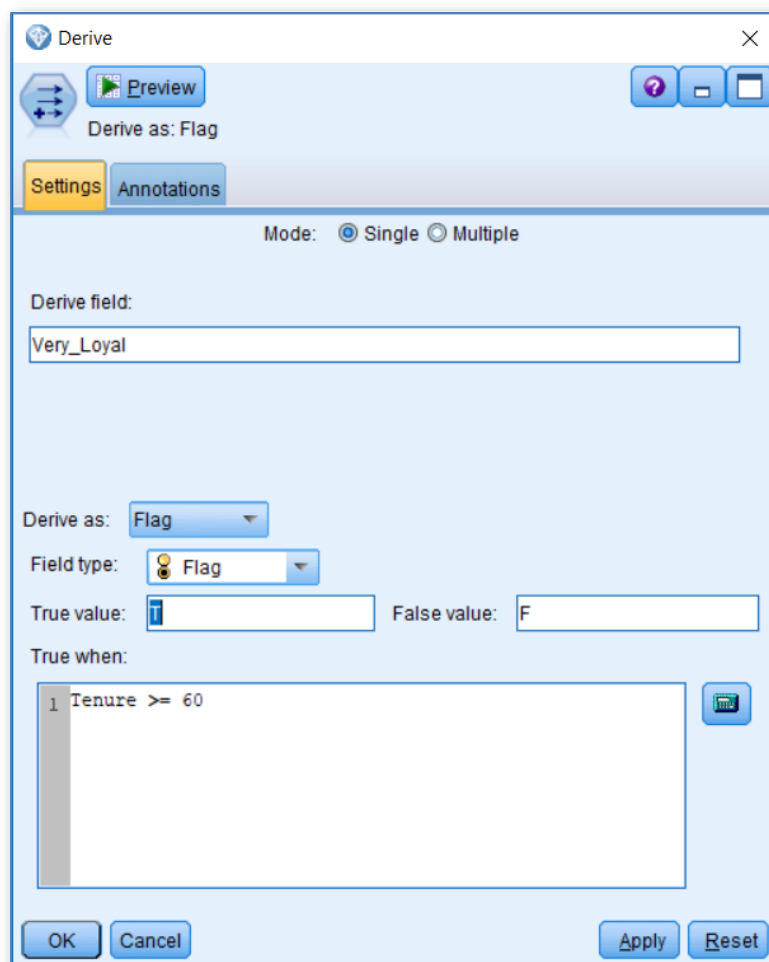


Figure 6.31 The completed Flag Derive node

To see the frequency of values within this new field. Click:

OK

From the Graphs palette:

Select and attach a Distribution node to the new Derive node

Edit the Distribution node and choose 'Very_Loyal' as the field

Run the Distribution node to generate the required output

Figure 6.32 shows the updated stream and figure 6.33 shows the output from the Distribution node.

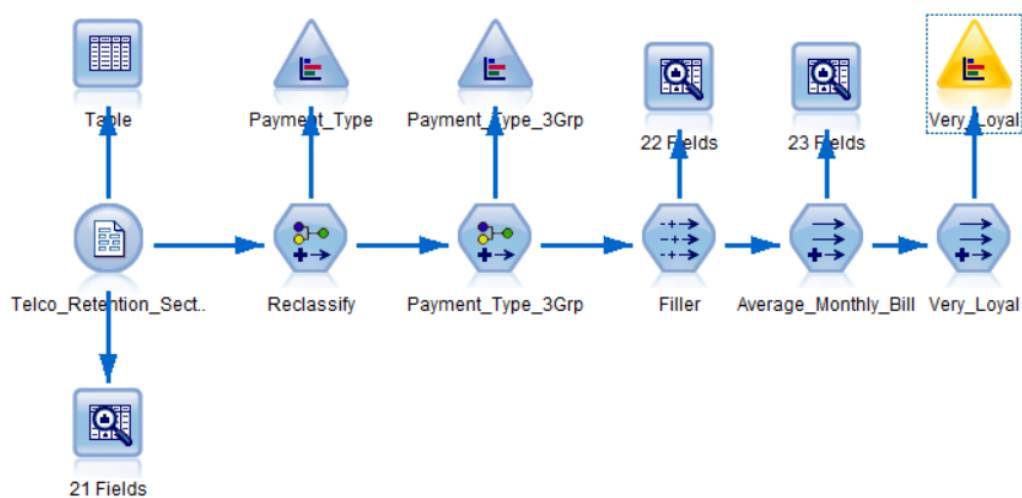


Figure 6.32 The updated stream with a Flag Derive and a Distribution node added

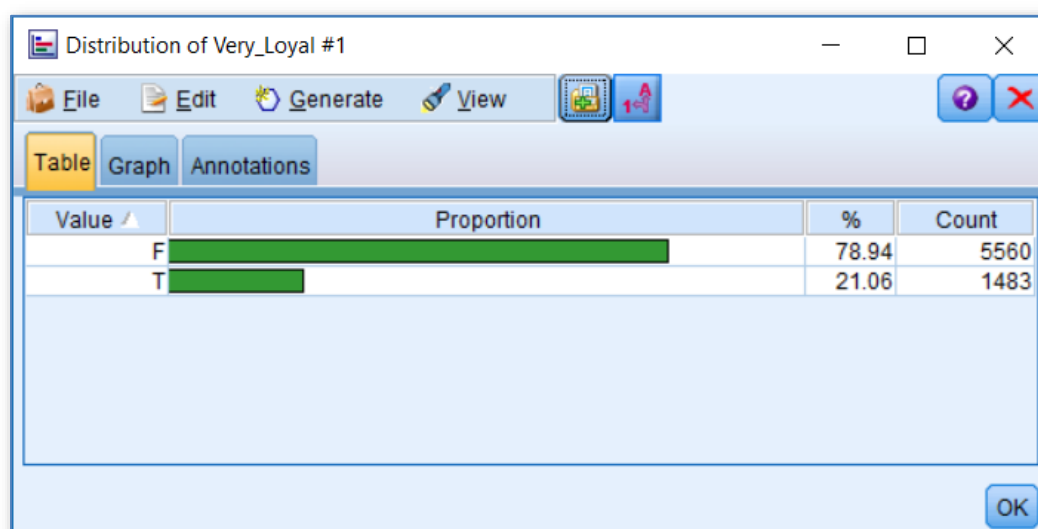


Figure 6.33 The Distribution node output showing the relative frequencies of the two values in the new flag field 'Very_Loyal'

To further illustrate the effects of choosing different derive modes, we will look at another example, this time deriving a field based on a conditional expression. From the Field ops palette:

Add another Derive node to the stream and connect it to the previous one

Edit the Derive node and specify the field name as 'Cashback'

Change the 'Derive as' mode to 'Conditional' and then set the Field type as Continuous

Figure 6.34 shows the Derive node at this stage.

The screenshot shows a configuration window for a 'Derive' node named 'Cashback'. The 'Derive as' dropdown is set to 'Conditional', and the 'Field type' dropdown is set to 'Continuous'. The 'Mode' is set to 'Single'. There are three input fields for 'If:', 'Then:', and 'Else:', each containing the value '1'. The 'Derive field' is named 'Cashback'. Buttons for 'OK', 'Cancel', 'Apply', and 'Reset' are at the bottom.

Figure 6.34 Creating a new field using a Conditional Derive

In this example, we can assume that the organisation wishes to reward its most loyal customers with a one-time cash rebate. The aim is to create a new field that calculates how much money each customer will receive. If the customer falls into the 'Very Loyal' group, then they get a rebate equal to 30% of their average monthly bill. The remaining customers get zero. To do this, we must enter values into each of the three condition boxes in the Derive dialog ('If', 'Then' and 'Else').

We begin by opening the Expression Builder next to the first condition box marked 'If'. Click:



Within the variable list box, find and click on the field marked 'Very_Loyal'.

Send it to the expression editor by clicking:



Enter an equals sign so the expression reads:

Very_Loyal =

We can of course just edit this expression manually, but if by chance you have forgotten the value that identifies the customer as 'Very Loyal', click the value selection button:



Figure 6.35 shows the Expression Builder at this stage.

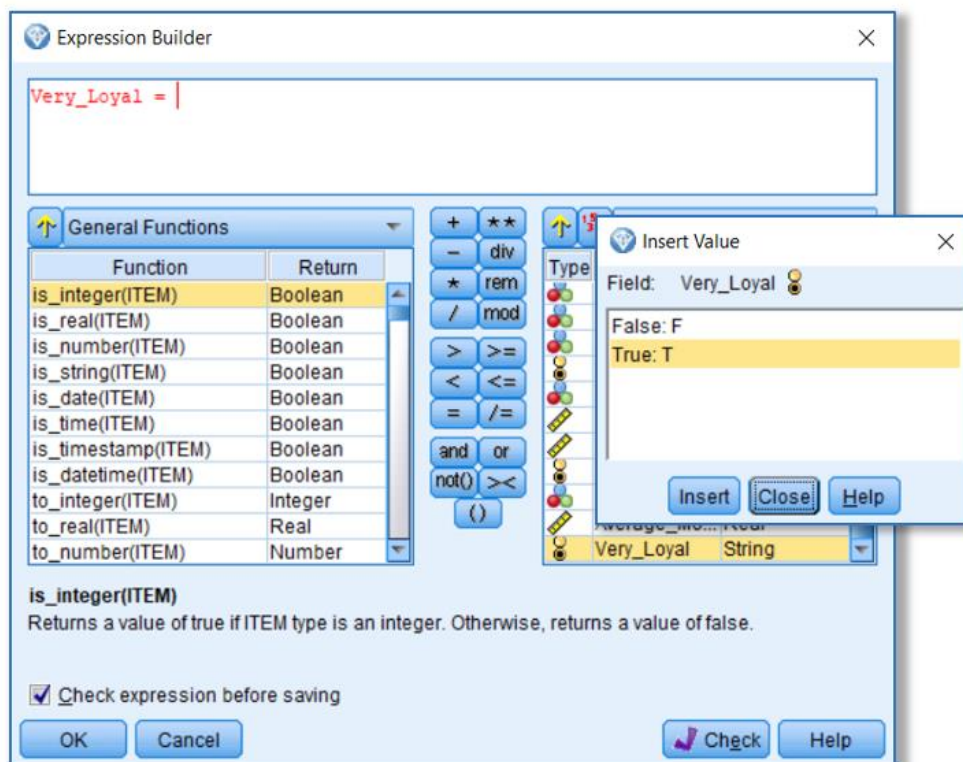


Figure 6.35 Using the Value Selector in the Expression Builder window

From the 'Insert Value' window we can see the two values for flag field 'Very_Loyal'. This is a very useful function especially when making multiple selections from within a categorical (or 'set') field. To select the 'True' group, click:

True: T

The Expression builder now looks like figure 6.36. Note that by using the Value Selector button, Modeler automatically places quotation marks around the character string value for true.

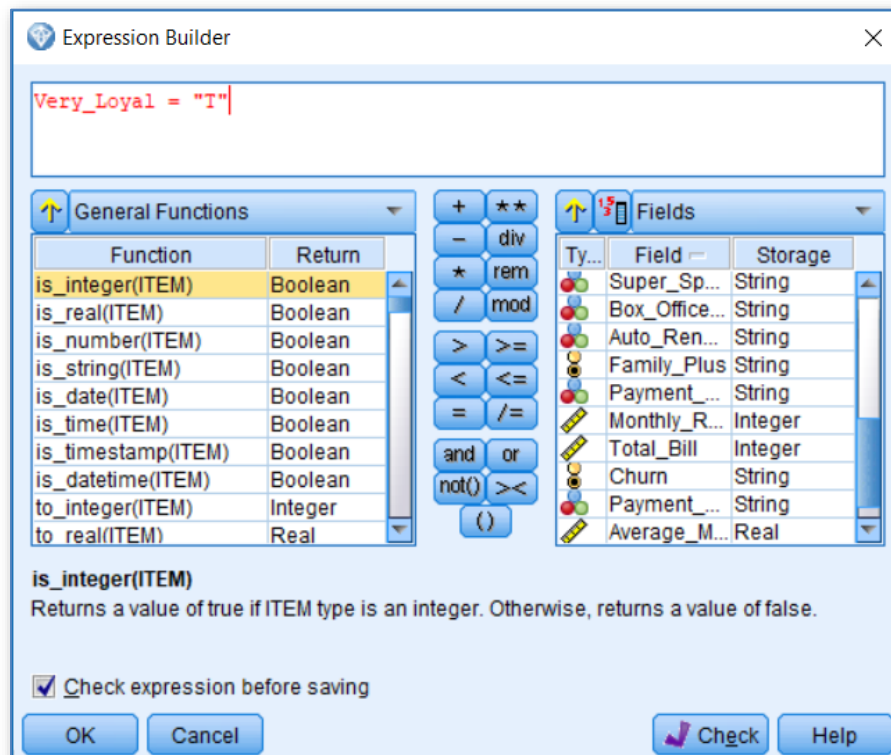


Figure 6.36 The completed condition is the Expression Builder selecting loyal customers

Return to the main dialog by clicking:

OK

We now need to enter expressions into the remaining boxes ('Then' and 'Else'). In the box marked 'Then', either directly or through the Expression Builder, enter the following expression that calculates 30% of each customer's average monthly bill:

Average_Monthly_Bill * 0.30

In the final box marked 'Else', simply enter the value zero:

0

The completed dialog is shown in figure 6.37.

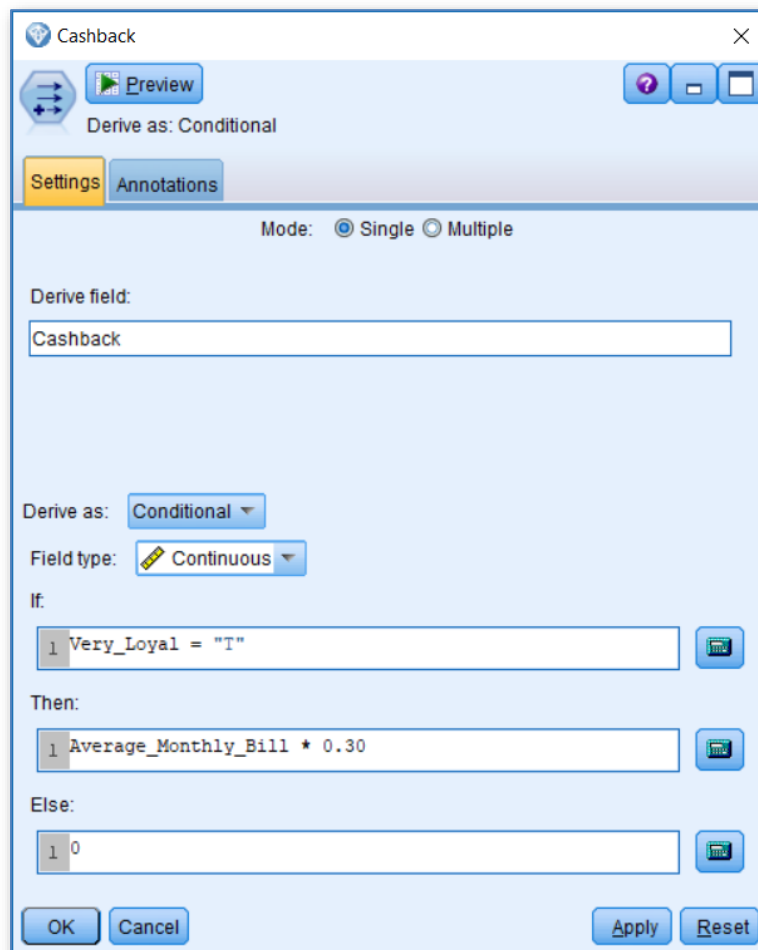


Figure 6.37 The completed Conditional Derive dialog

To view this new distribution, click:

OK

From the Output palette:

Double-click the Table node

The Table node should automatically attach itself to the new conditional Derive.

Run the Table node

Scroll across to the last column showing the 'Cashback' field and scroll down to the last few records in the output table

Figure 6.38 shows the updated stream and figure 6.39 shows the last records in the output table for the field 'Cashback'.

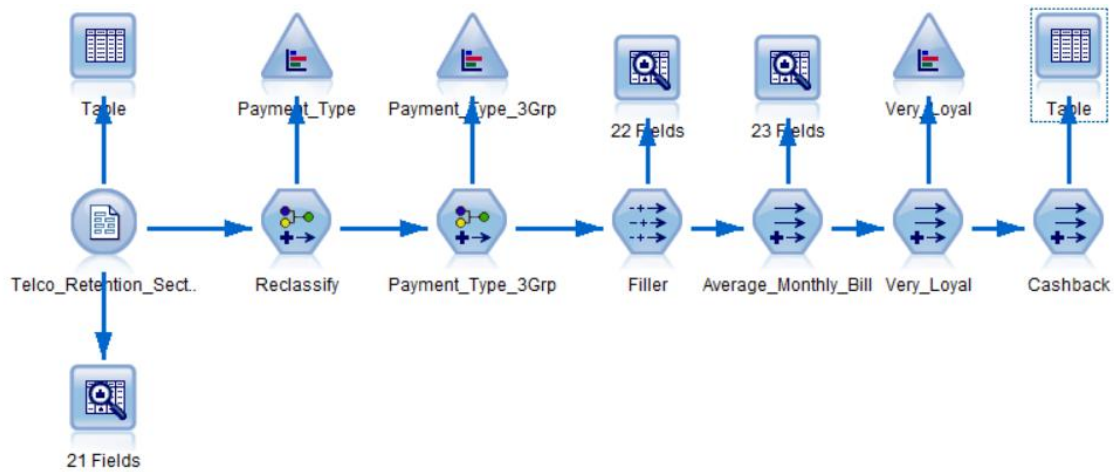


Figure 6.38 The updated stream with the Conditional Derive node added

	ate	Total_Bill	Churn	Payment_Type_3Grp	Average_Monthly_Bill	Very_Loyal	Cashback
7024	54	3784	No	Card_Payment	52.556	T	15.767
7025	110	7759	No	Card_Payment	107.764	T	32.329
7026	106	7565	No	Card_Payment	105.069	T	31.521
7027	26	1937	No	Card_Payment	26.903	T	8.071
7028	26	1864	No	Card_Payment	25.889	T	7.767
7029	115	8350	No	Account_Debit	115.972	T	34.792
7030	118	8670	No	Card_Payment	120.417	T	36.125
7031	20	1567	No	Invoice	21.764	T	6.529
7032	108	7767	No	Card_Payment	107.875	T	32.362
7033	70	5102	No	Card_Payment	70.861	T	21.258
7034	26	1872	No	Card_Payment	26.000	T	7.800
7035	25	1787	No	Card_Payment	24.819	T	7.446
7036	56	3880	No	Card_Payment	53.889	T	16.167
7037	98	6841	No	Account_Debit	95.014	T	28.504
7038	84	6052	No	Card_Payment	84.056	T	25.217
7039	74	5237	No	Account_Debit	72.736	T	21.821
7040	63	4686	No	Card_Payment	65.083	T	19.525
7041	105	7544	No	Account_Debit	104.778	T	31.433
7042	21	1419	No	Card_Payment	19.708	T	5.912
7043	103	7363	No	Card_Payment	102.264	T	30.679

Figure 6.39 The table node showing calculated rebate values for loyal customers

In our final example of working with the Derive node, we will see how to create a banded variable from a Nominal Derive node. However, instead of placing another Derive node on the canvas stream, let's look at how we can utilise the 'Generate' menu to do this for us.

Click the last Conditional Derive node to select it

From the graphs palette:

Double click the Histogram node to automatically join it to the last Derive node ('Cashback')

Edit the histogram node and choose the field 'Total_Bill' to be displayed

Run the histogram

Figure 6.40 shows the resulting Histogram output displaying the distribution of values in the field 'Total_Bill'.

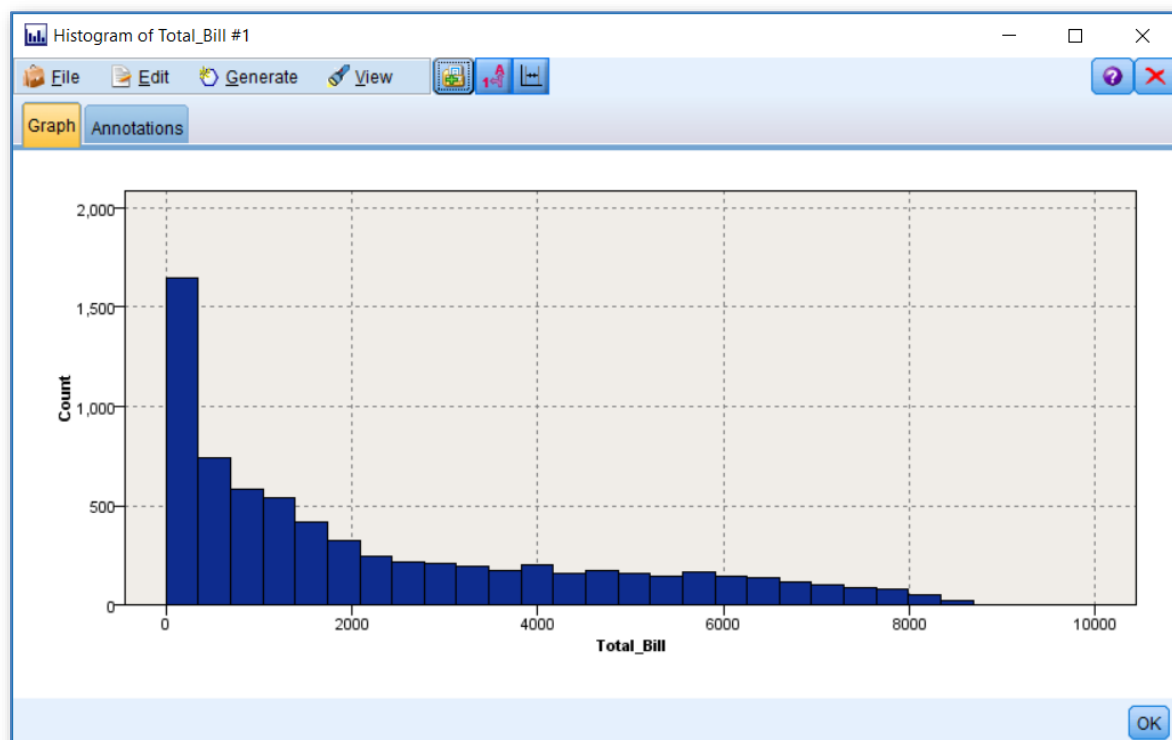


Figure 6.40 Histogram output displaying the distribution of values in the field 'Total_Bill'

Having generated a histogram of the field from which we wish to create banded groups, we can interact with the graph itself to begin this process. From the menu bar in the graph output, click the 'Split graph into bands' button:



Immediately a 'Split into equal bands' toolbar is generated. The default setting is to split the distribution into 3 equal bands.

Change the 'Split into value' to 4

Now click:



The histogram is split using three lines to create four groups. Figure 6.41 shows this.

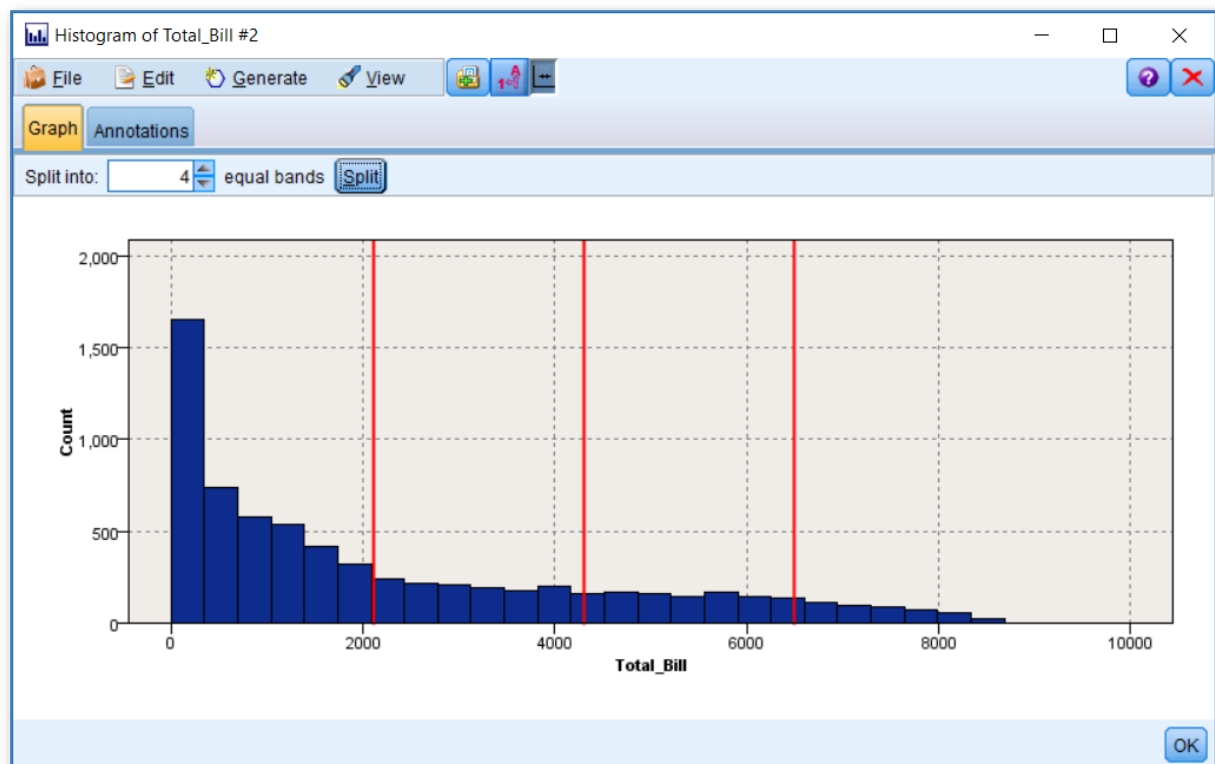


Figure 6.41 Histogram output split into four equal groups

You may notice from the split histogram, that 'Four equal groups' does not mean four groups of equal size but rather that the range of values is split equally. As a result, the four groups are very different in terms of the number of cases in each group. If you needed to split a continuous variable so that the groups contained an *equal* number of cases, procedures such as the Binning node can do this.

We can use this approach as way to get started with creating a banded field. From the menu within the Histogram output, click:

Generate

Derive node for bands

Once again, the Generate menu places a new node on the stream canvas.

Close the Histogram output

Attach the new Derive node to the last Derive node ('Cashback')

Figure 6.42 shows the updated stream

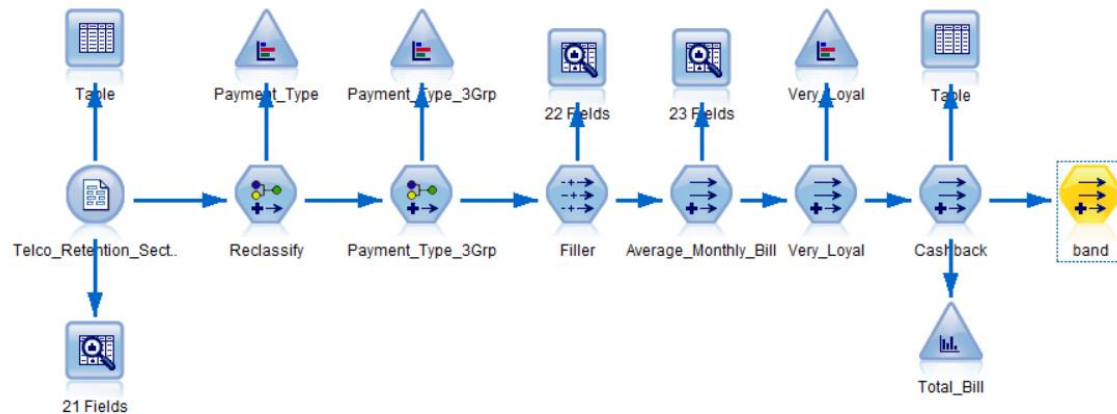


Figure 6.42 The updated stream with the Nominal Derive node added

We can use the newly generated Nominal Derive node as the basis for our banded variable. To do this:

Edit the Nominal Derive node

Figure 6.43 shows the edited Nominal Derive node.

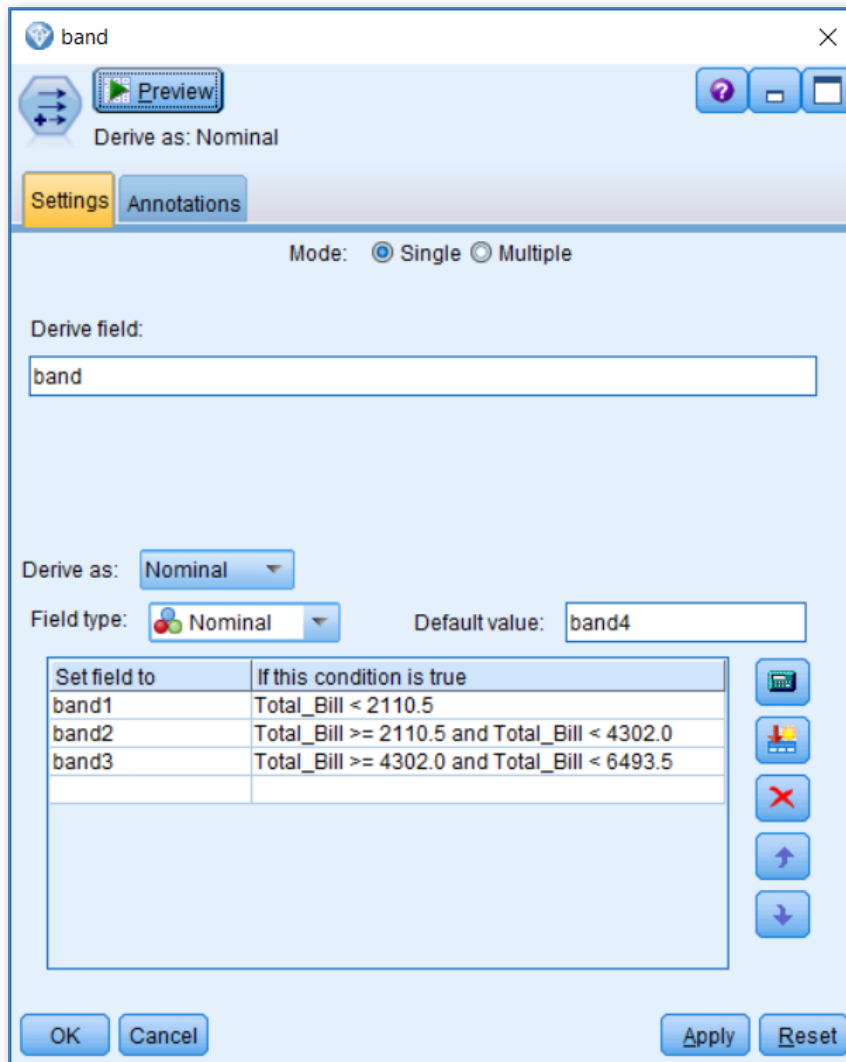


Figure 6.43 Nominal Derive node, generated from a Histogram to create four bands in the variable 'Total_Bill'

We can now edit the existing values within the dialog so that they create more meaningful banded groups.

Change the Derive field name to 'Customer_Tier'

Change the field type to Ordinal

In the column marked 'Set field to':

Change the description from 'band1' to 'a. Premium'

In the adjacent cell under the column marked 'If this condition is true':

Change the expression to 'Total_Bill >= 5000'

Returning to the column marked 'Set field to', in the second row:

Change the description from 'band2' to 'b. Select'

In the adjacent cell under the column marked 'If this condition is true':

Total_Bill >= 2000 and Total_Bill <5000

Returning to the column marked 'Set field to', in the third row:

Change the description from 'band3' to 'c. Standard'

In the adjacent cell under the column marked 'If this condition is true':

Total_Bill >0 and Total_Bill <2000

Finally, in the box marked Default value:

Change the description from 'band4' to 'd. Refunded'

Figure 6.44 shows this completed dialog.

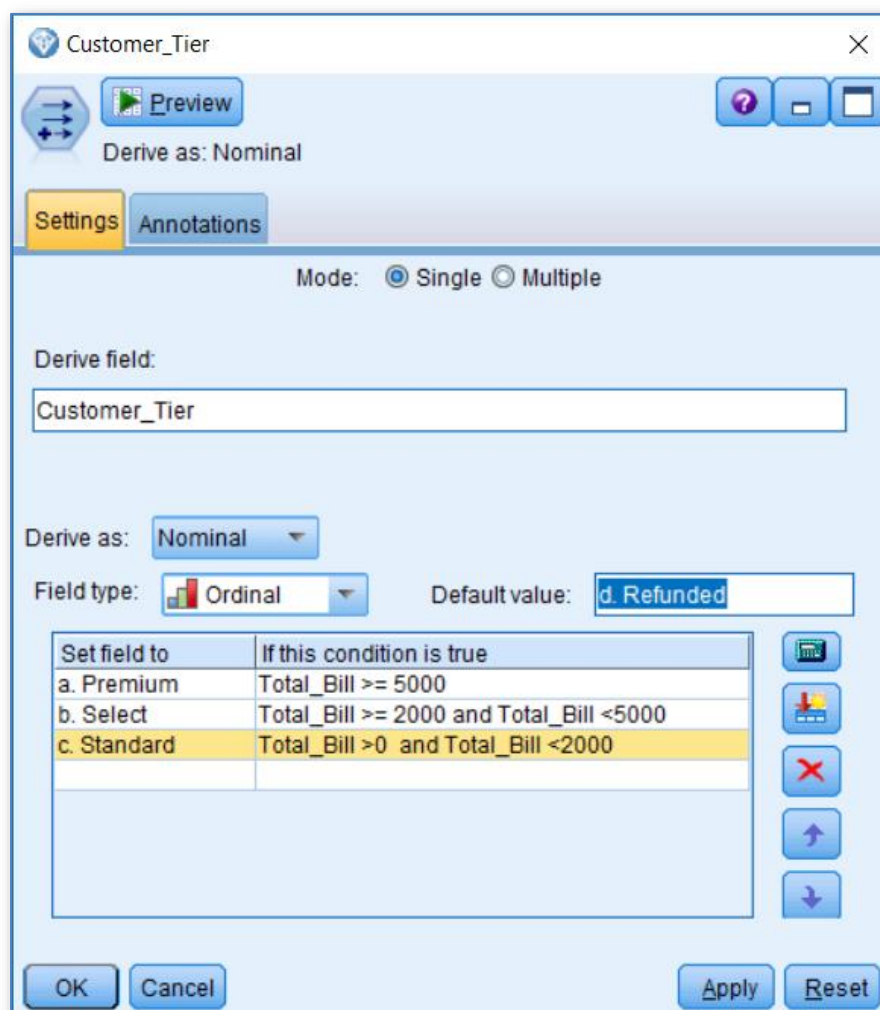


Figure 6.44 Completed Nominal Derive node creating the banded field 'Customer_Tier'

To complete the process, click:

OK

To check the that the Nominal Derive node has correctly specified the ranges, from the graphs palette:

Double-click the Distribution node to automatically attach it to the last Nominal Derive node ('Customer_Tier')

Edit the Distribution Node and choose the field 'Customer_Tier' to be displayed

Run the completed Distribution node

We can see from the output generated, that there were 12 cases that fell into the default group of 'd. Refunded'. These were cases where the customer's total bill was equal to zero. It's good practice to ensure that each category is prefixed with a letter or number. This is to preserve the desired order of the groups (otherwise Modeler simply displays them in numeric or alphabetical order).

Figure 6.45 shows the generated Distribution table output.

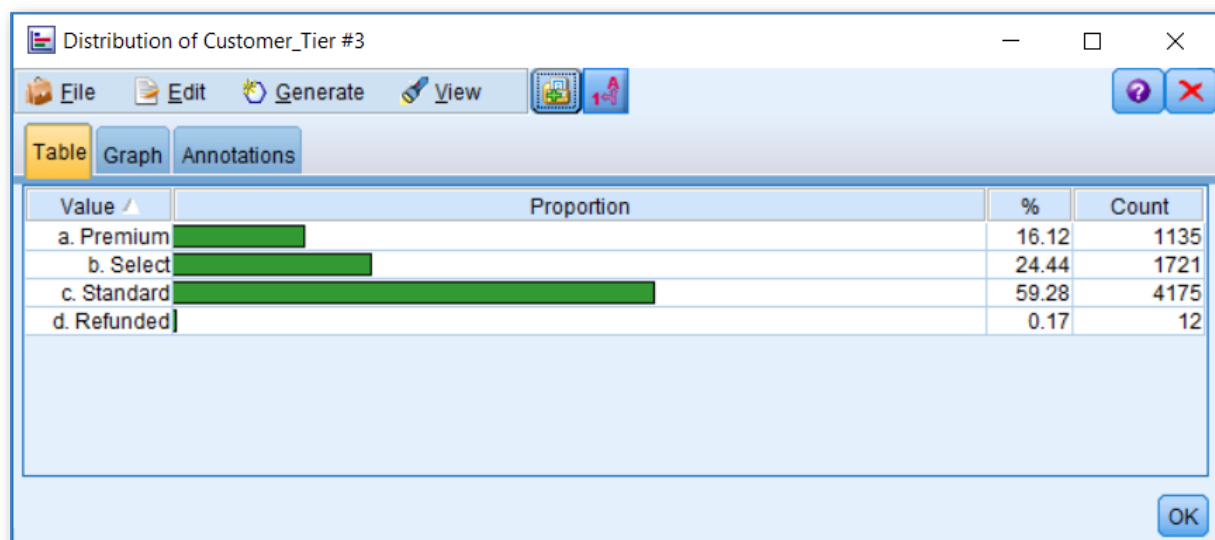


Figure 6.45 Distribution chart of the newly derived field 'Customer_Tier'

Figure 6.46 shows the updated stream with all the completed nodes within it.

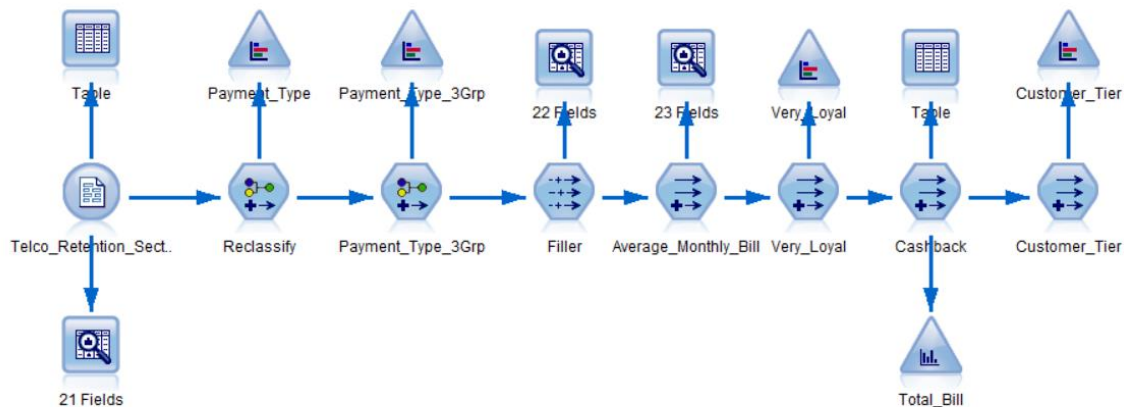


Figure 6.46 Completed stream demonstrating the use of the Reclassify, Filler and Derive nodes

The stream is now quite crowded with nodes that change existing fields and create new ones. We can easily encapsulate these nodes within one Supernode. Before we do, from the Output palette,

Attach a final Data Audit node to last Drive Node in the stream ('Customer_Tier')

Now create a Supernode:

Drag a selection rectangle around all the nodes downstream of the original Data Source node but not including the last Data Audit node

Right-click on one of the highlighted nodes so that a pop-up menu is generated

Figure 6.47 shows this process in action.

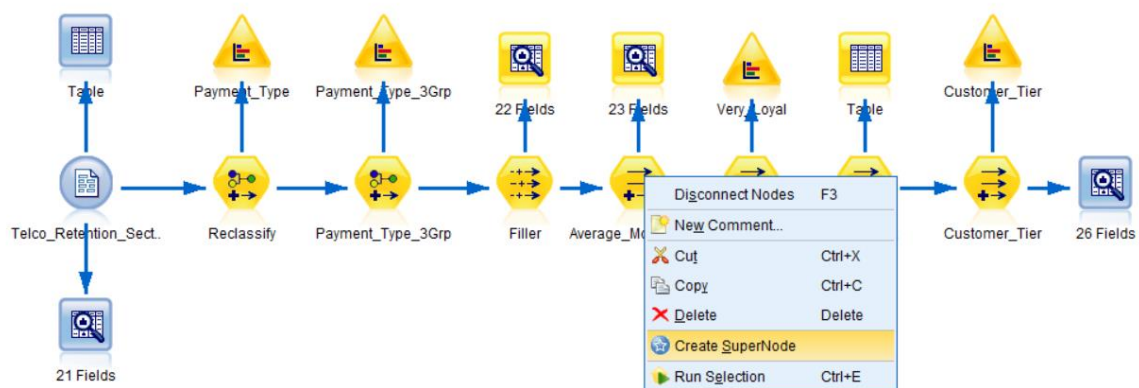


Figure 6.47 Selecting several nodes to create a Supernode

From the pop-up menu, click:

Create Supernode

Edit the Supernode and click on the Annotations Tab

Give the Supernode the custom name 'Data Preparation'

Figure 6.48 shows the resultant Supernode and the Annotations tab within it.

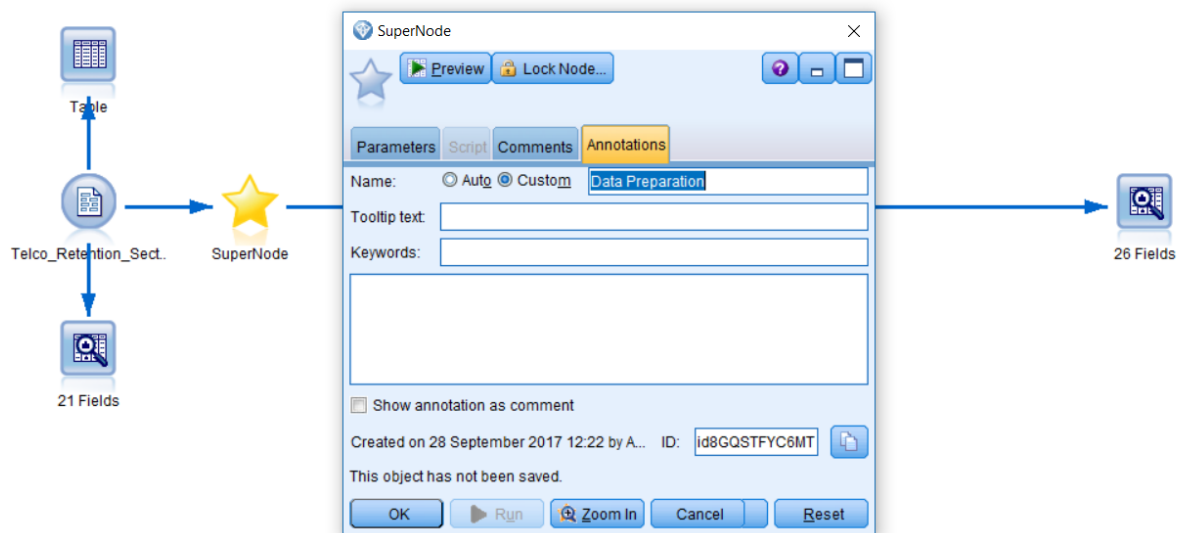


Figure 6.48 Annotating the newly created Supernode

Now click:

OK

Reposition the Data Audit node so it's closer to the Supernode

Run the Data Audit node

Figure 6.49 shows the Data Audit output.

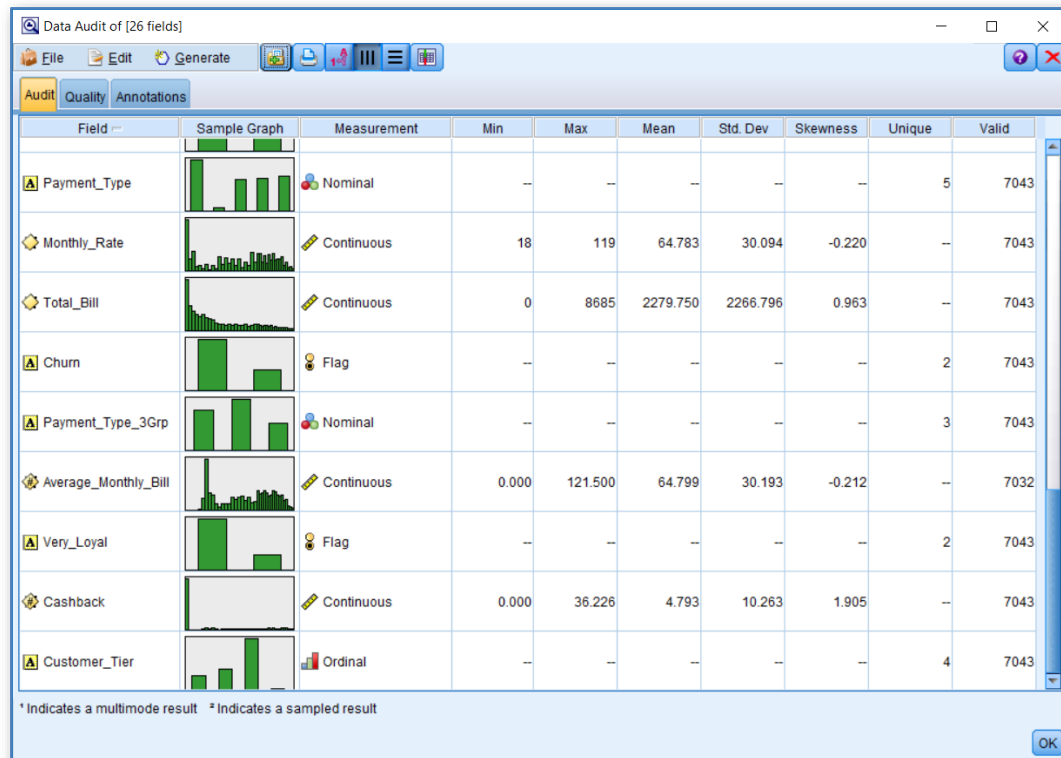


Figure 6.49 Data Audit output showing summary statistics for updated and new fields created using the Reclassify, Filler and Derive nodes

Using a combination of the Reclassify, Filler and Derive nodes we have now transformed three existing fields and created four new ones.

Section 7: Exploring Data

- The Matrix Node
- The Distribution Node
- The Histogram Node
- The Means Node
- The Plot Node
- The Statistics Node
- The Graphboard Node



In this section, we will look at several procedures that enable us to explore simple relationships in data. Data exploration is a key aspect of predictive analytics. By exploring the data, we can query our consolidated file to check that the relationships within it make sense as well as gain a deeper understanding as to how the variables are interconnected. Within SPSS Modeler, there are several ways we can do this using a combination of charts and output nodes.

The Matrix Node



The Matrix node allows us to generate crosstabs within Modeler. Crosstabs (or contingency tables) are a simple and effective way to analyse the relationships between categorical fields. However, as we shall see, within Modeler we can also incorporate continuous fields into the Matrix node's output. To begin the section, open the following stream from the Section 7 folder:

Section 7 Start.str

Figure 7.1 shows this stream. Note that it reads an updated version of the Telco file.



Figure 7.1 The SPSS Modeler stream 'Section 7 Start.str'

We can start the process of exploring the data by attaching a Matrix node to the stream. From the Output tab:

Select and attach a Matrix node to the Data Source node

Figure 7.2 shows the attached Matrix node.



Figure 7.2 Matrix node attached to the source data file

Right-click the Matrix node to edit it

Figure 7.3 shows the default Settings tab within the edited Matrix node.

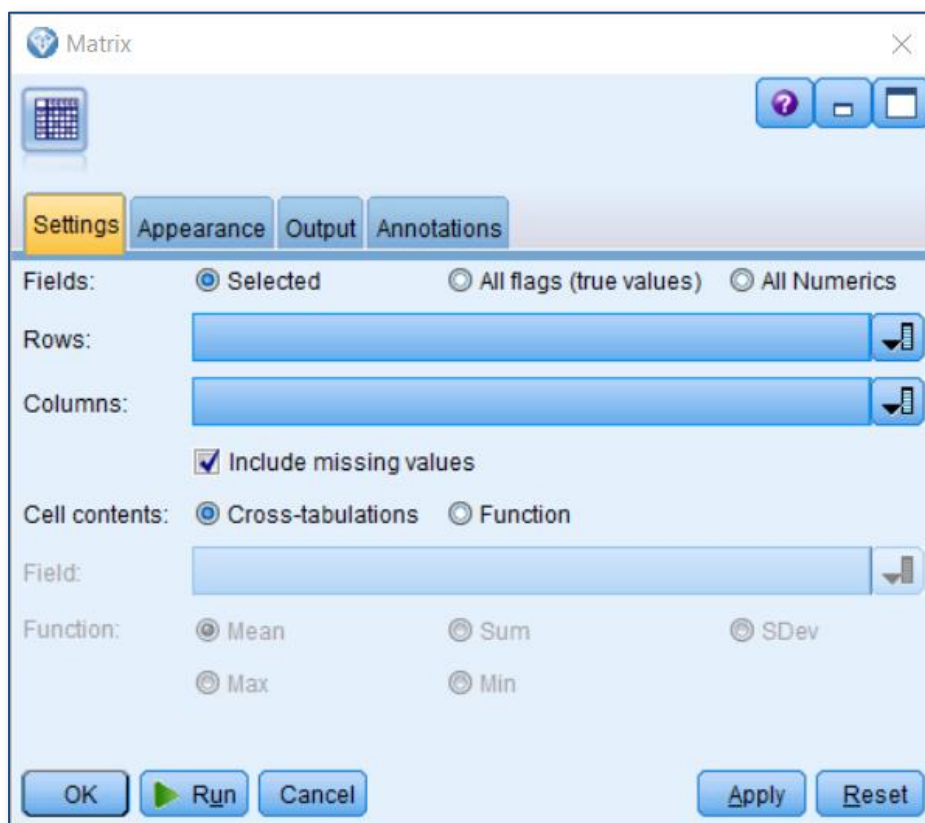


Figure 7.3 The default Settings tab within the edited Matrix node

A simple crosstab is comprised of a single categorical field in each row and column dimension of a table. To illustrate, click the drop-down field selector marked:

Rows

From the field list, choose the field:

Partner

Now click the drop-down field selector marked:

Columns

From the field list, choose the field:

Customer_Tier

Figure 7.4 shows the completed dialog.

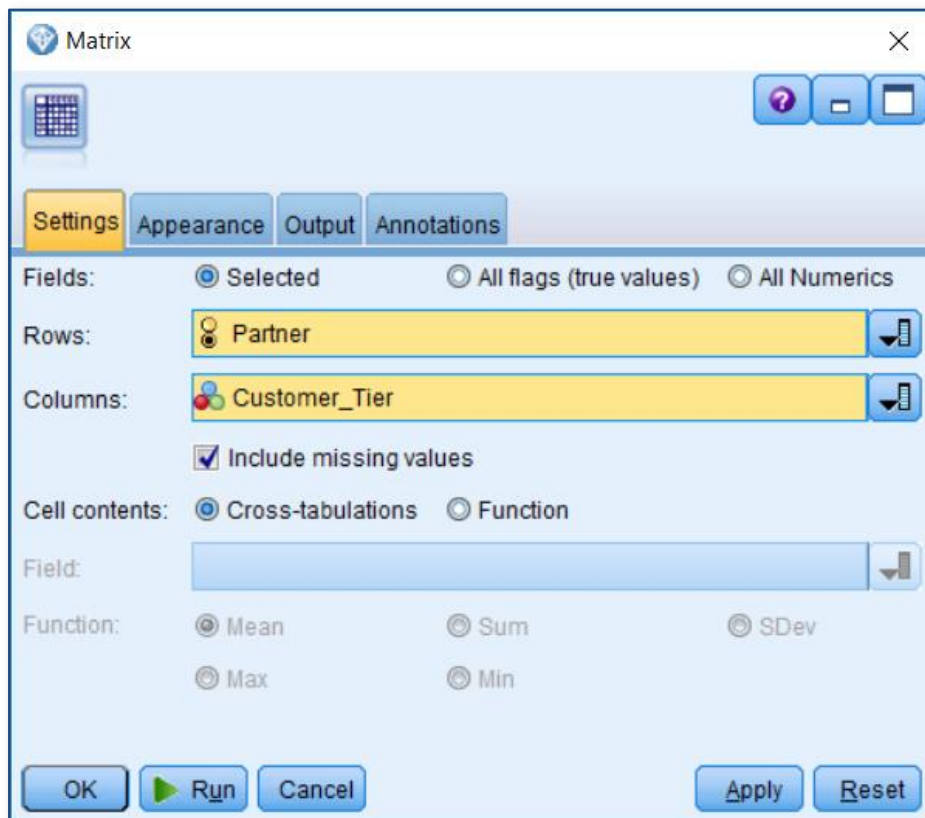


Figure 7.4 The completed Settings tab within the Matrix node

Click:

Run

Figure 7.5 shows the resultant output.

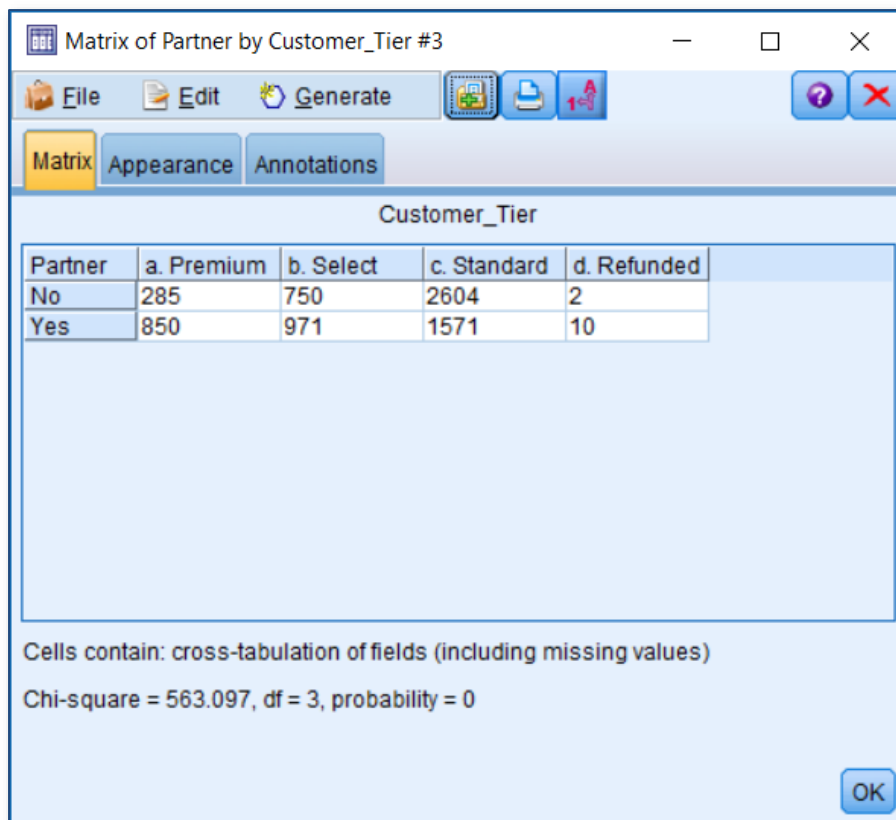


Figure 7.5 The Crosstab output from the Matrix node

As we can see in Figure 7.5, by default the Matrix output shows the frequency count of values in each customer tier broken down by whether or not the person is living with a partner. Moreover, the crosstabulation includes a probability value shown at the bottom of the output screen. The probability value is derived from a Chi-Square statistical test. Chi-Square is used to test the null hypothesis that the two variables are unrelated. In this case, the probability that this null hypothesis is true, is very small (the output shows this is equal to zero but in reality, it's just a very small number). We would conclude that whether or not the customer is living with a partner is probably not randomly related to their customer tier group.

You may also note that the last column in the crosstab shows a few cases where the customers fell into the category 'd. Refunded'. In fact, within the Type tab of the Data Source node, this category has been marked as 'missing' and within the Matrix node we can choose whether missing values are displayed or not. To illustrate:

Close the Matrix output and edit the Matrix node again

Within the setting tab:

Uncheck the box next to 'Include missing values'

Figure 7.6 shows the updated Settings tab within the edited Matrix node.

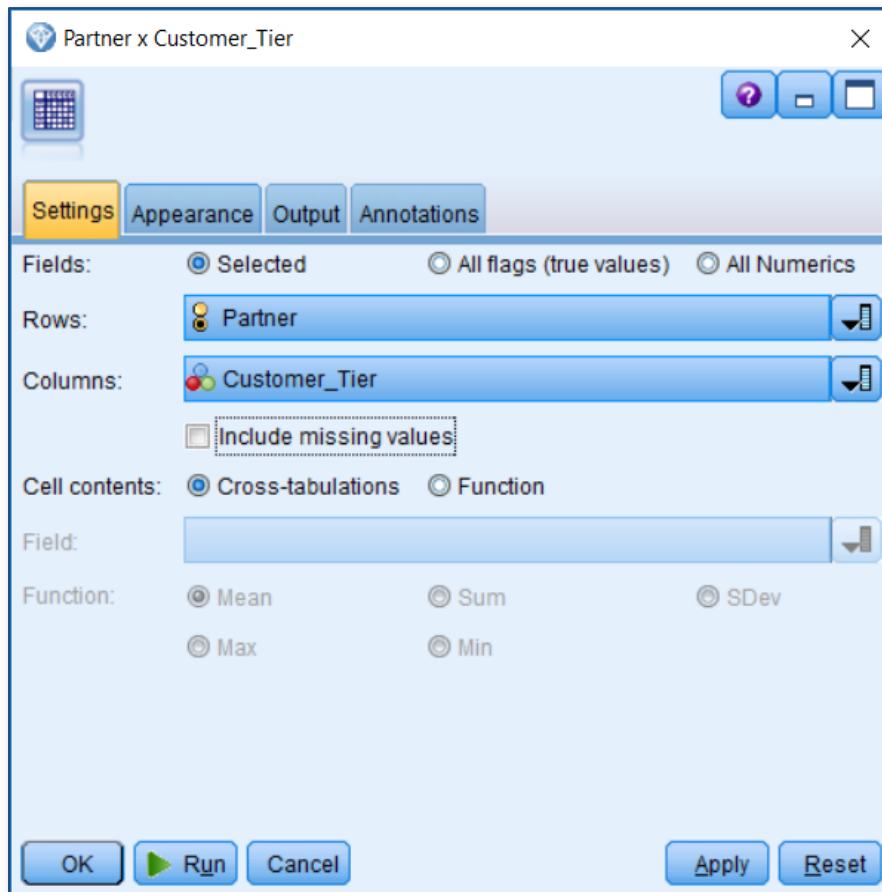


Figure 7.6 The Settings tab with 'Include missing values' deselected

Click:

Run

Figure 7.7 shows new output from the procedure.

Matrix of Partner by Customer_Tier #4

File Edit Generate

Matrix Appearance Annotations

Customer_Tier

Partner	a. Premium	b. Select	c. Standard
No	285	750	2604
Yes	850	971	1571

Cells contain: cross-tabulation of fields

Chi-square = 557.236, df = 2, probability = 0

OK

Figure 7.7 Output from the Matrix node with Missing Values suppressed

We can see from the results, that the category 'd. Refunded' has now been dropped from the table. However, it is still difficult to compare the groups as the total number of cases in each category are different. Ideally, we would like to examine what percentage of people who live with or without a partner fall into the various customer tier categories. We can easily do this by requesting that percentages are displayed in the cells. In fact, the output from the Matrix node is one of the few places in Modeler where we can request that the displayed values are updated without having to re-run the node. To do so, while the Matrix output is still displayed, click the tab marked:

Appearance

Within the Appearance tab, check the boxes marked:

Percentage of row

Include row and column totals

Figure 7.8 shows the completed Appearance tab.

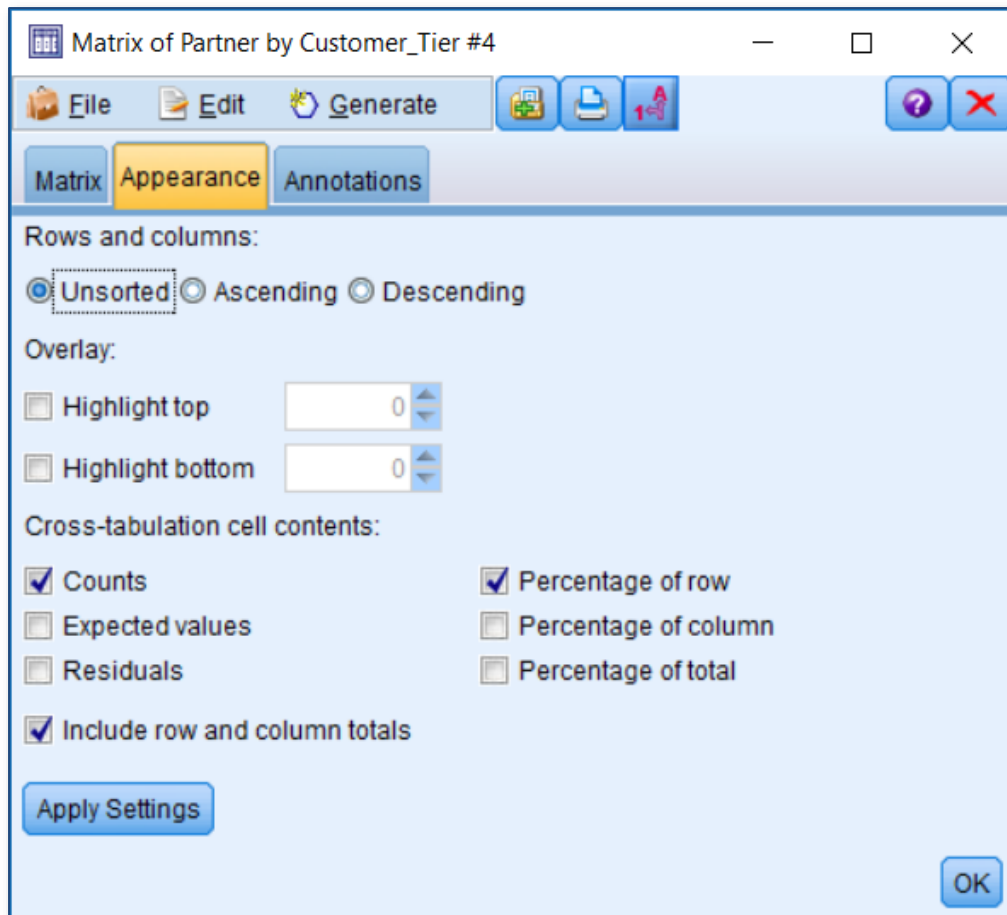


Figure 7.8 Requesting additional values for the Matrix Output

To see the effect of adding these values, click:

Apply Settings

Return to the output by clicking the tab marked:

Matrix

Figure 7.9 shows the updated crosstab with the Row percentages and Totals included.

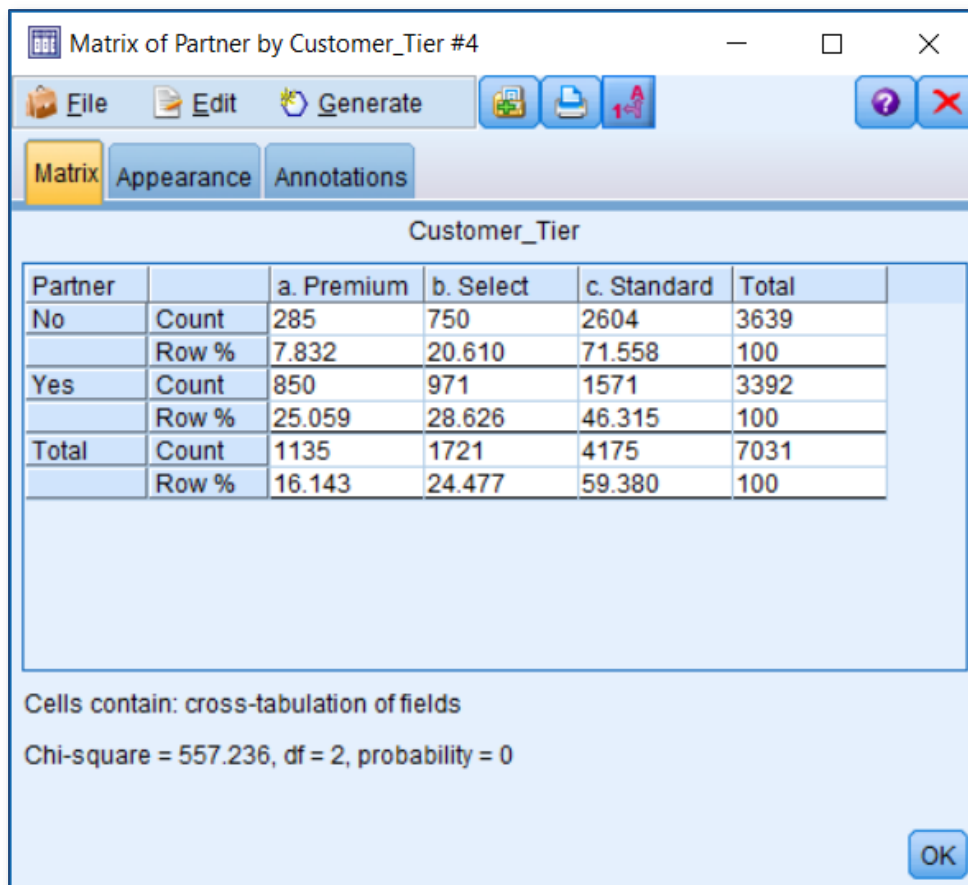


Figure 7.9 Matrix output with Count, Row percentages and Totals displayed

Now the Matrix output has row percentages, it's much easier to compare the groups. We can see that those customers who indicated that they lived with a partner are much more likely to fall into the 'a. Premium' or the 'b. Select' customer tiers than those that do not. In fact, 71% of those who do not live with a partner fall into the 'c. Standard' customer tier compared to 46% of those who do.

Before we leave the Matrix node, let's look at another example where we can use the procedure to compare groups in terms of a summary measure derived from a third continuous field. From the output palette:

Add another Matrix node to the existing source node.

Figure 7.10 illustrates this.

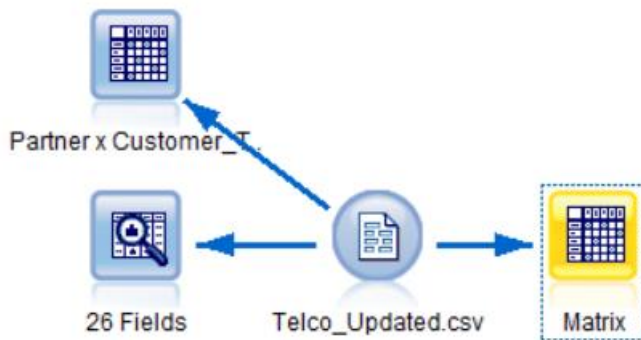


Figure 7.10 Adding a second Matrix node to the stream

Having attached the second Matrix node:

Right-click and edit it

From within the Settings tab:

Choose the variable `Payment_Type` for the Rows dimension

Choose the variable `Auto_Renew` for the Columns dimension

At this point, the node will generate a Crosstabulation of the two categorical fields 'Payment_Type' and 'Auto_Renew' with frequency counts displayed in the cells. We can however, add a continuous field and request that a summary measure is displayed in the cells instead. To do so, click the radio button marked:

Function

Within the activated drop-down field menu

Choose the variable `Tenure`

Figure 7.11 shows the completed dialog at this stage.

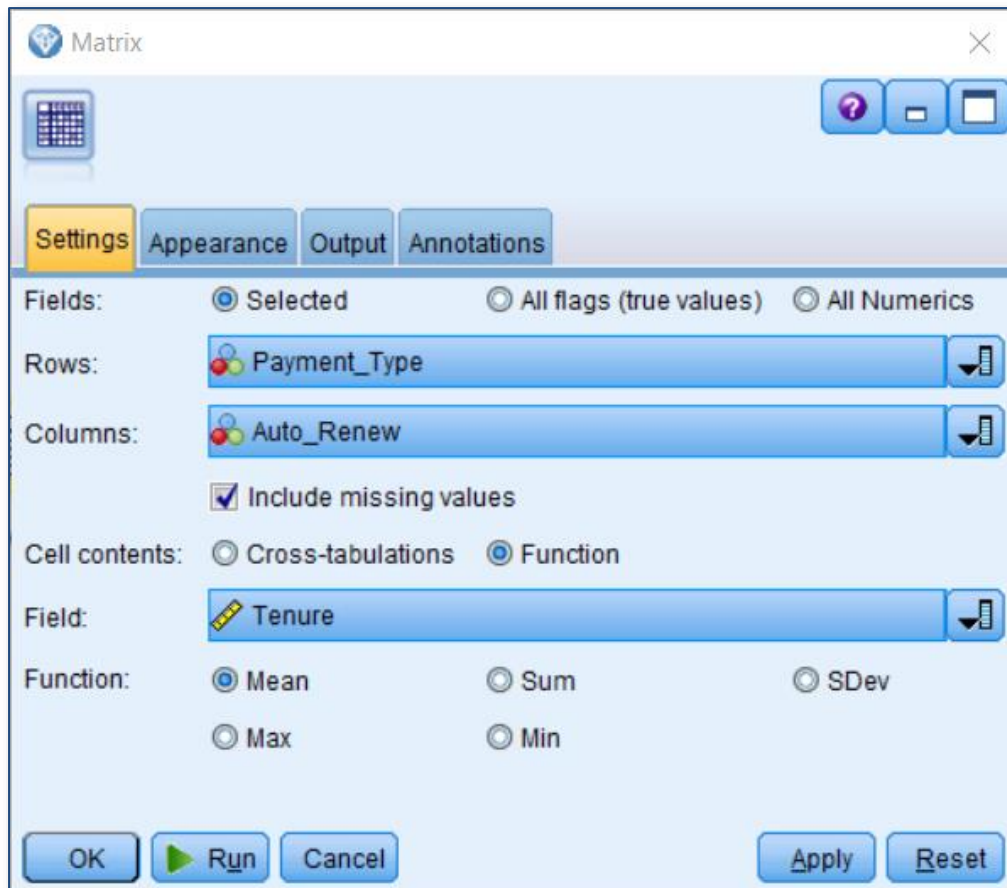


Figure 7.11 The Settings tab with the variable Tenure added as a summary field

To see the resultant output, click:

Run

As figure 7.12 shows, the Matrix output displays how the average customer tenure varies by their payment method and the renewal length of their current contract.

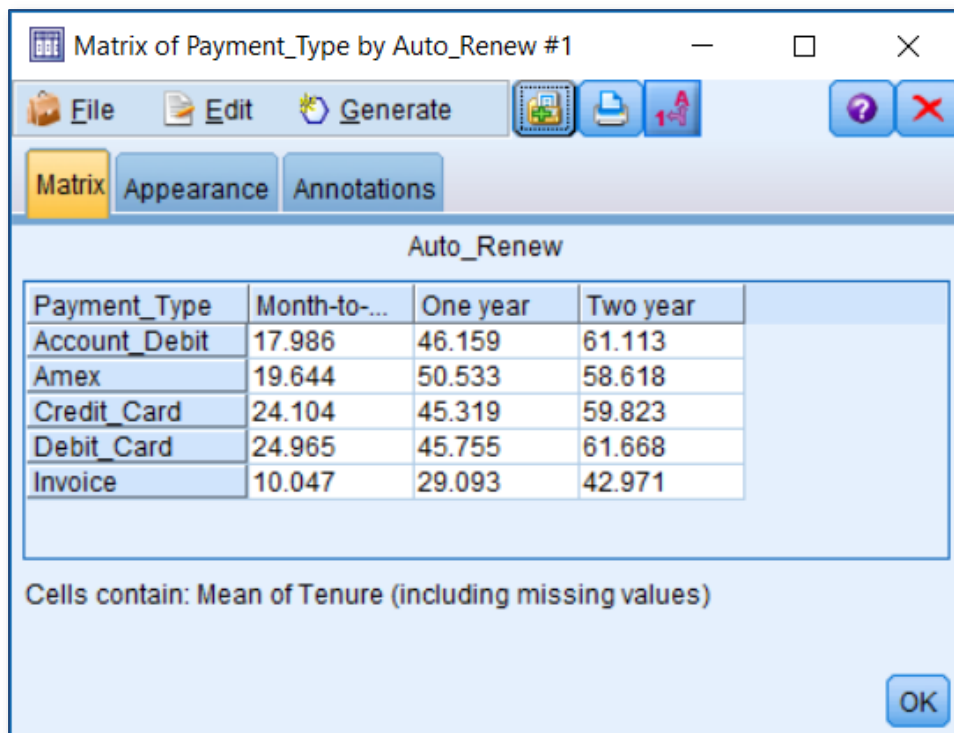


Figure 7.12 Matrix output where the cells contain the mean values for the variable Tenure

The Distribution Node



Distribution

We have already briefly seen the Distribution node being used to describe categorical fields in earlier sections. In particular, we have seen many examples of the default bar charts that it produces displayed within the output from the Data Audit node. In this example we will take a quick look at how we can use the Distribution node to illustrate the kinds of interactions between categorical fields that we might observe within a Matrix node. From the Graphs palette:

Select and join a Distribution node to the existing Data Source node in the stream

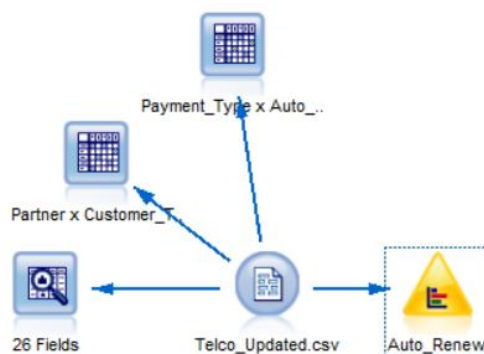


Figure 7.13 Adding a Distribution Node to the stream

Double-click on the newly added node to edit it

Within the Plot tab of the edited node:

Choose the variable 'Auto_Renew' as the field to plot

In the Overlay section of the same tab:

Choose the variable 'Partner' as the colour field

Figure 7.14 shows the completed tab.

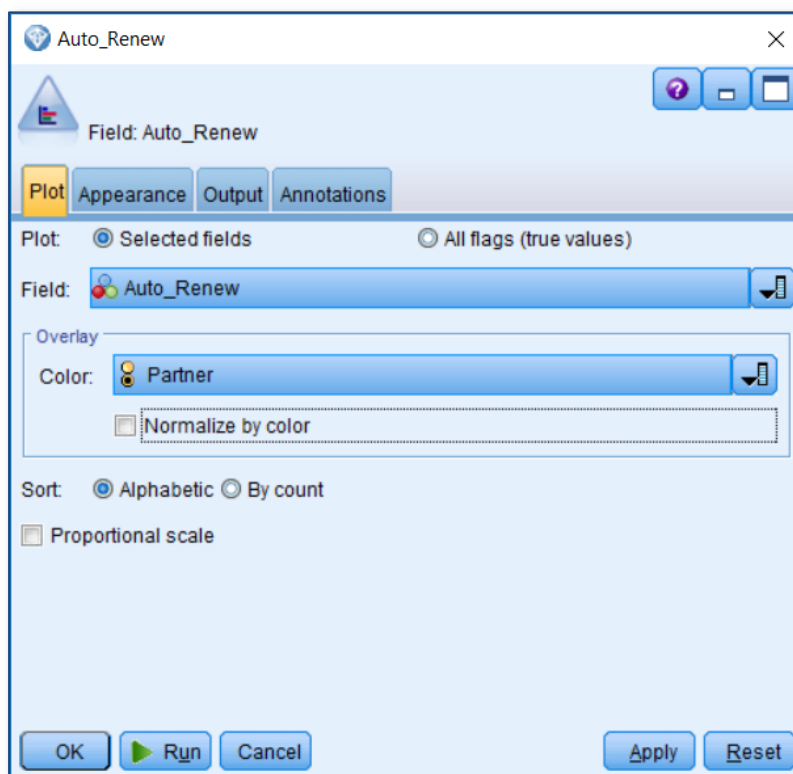


Figure 7.14 Completed Plot tab for a stacked Distribution chart

Now click:

Run

Figure 7.15 shows the generated output.

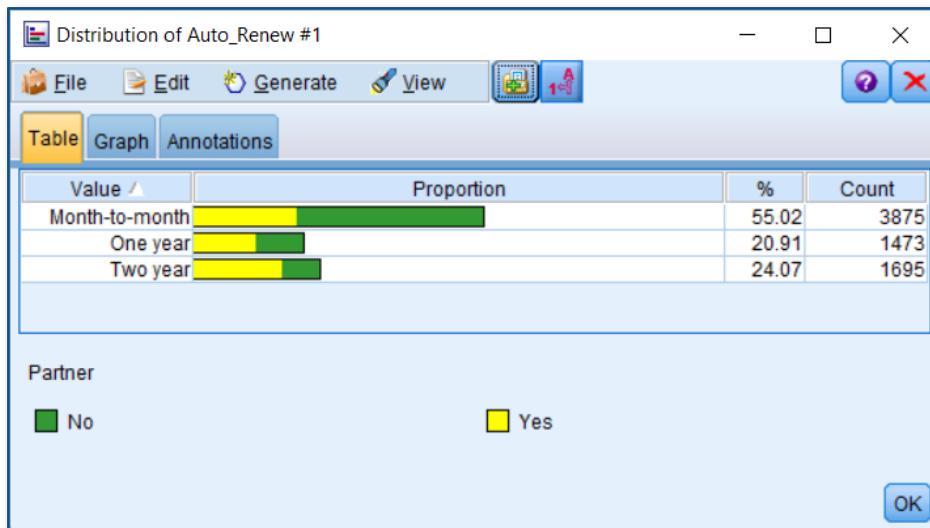


Figure 7.15 Table Tab Output from the Distribution Node

The initial output that the Distribution node displays is the Table tab. This shows both graphical and tabular results. We can use this tab to sort the categories (clicking on the header columns) in alphabetical or numerical order. We can also interact with the output and use the Generate menu to create new nodes to regroup the categories, create flag fields or select cases. We can also view the output purely as a chart by clicking the tab marked:

Graph

Figure 7.16 shows the Graph tab output.

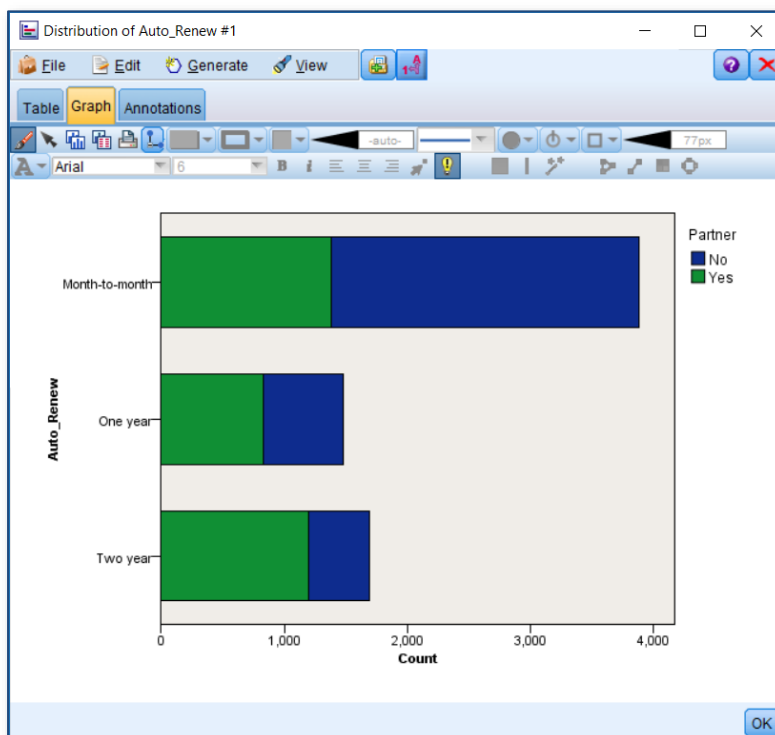


Figure 7.16 Graph Tab Output from the Distribution Node

Initially, the Graph tab is in 'Explore' mode. This enables the user to view the data values within the chart and to activate different chart areas in order to use the functionality in the Generate menu. However, the Graph tab also allows us full access to the editing controls that affect the chart appearance. Here we can exclude categories, customise their order, change the scale, change fonts, colours and backgrounds. To access these controls, from the main menu within the tab, click:

View

Edit Mode

Note: you may need to check the boxes in the drop-down menu to unlock the edit controls for various aspects of the chart.

Figure 7.17 shows the chart with all the edit controls activated.

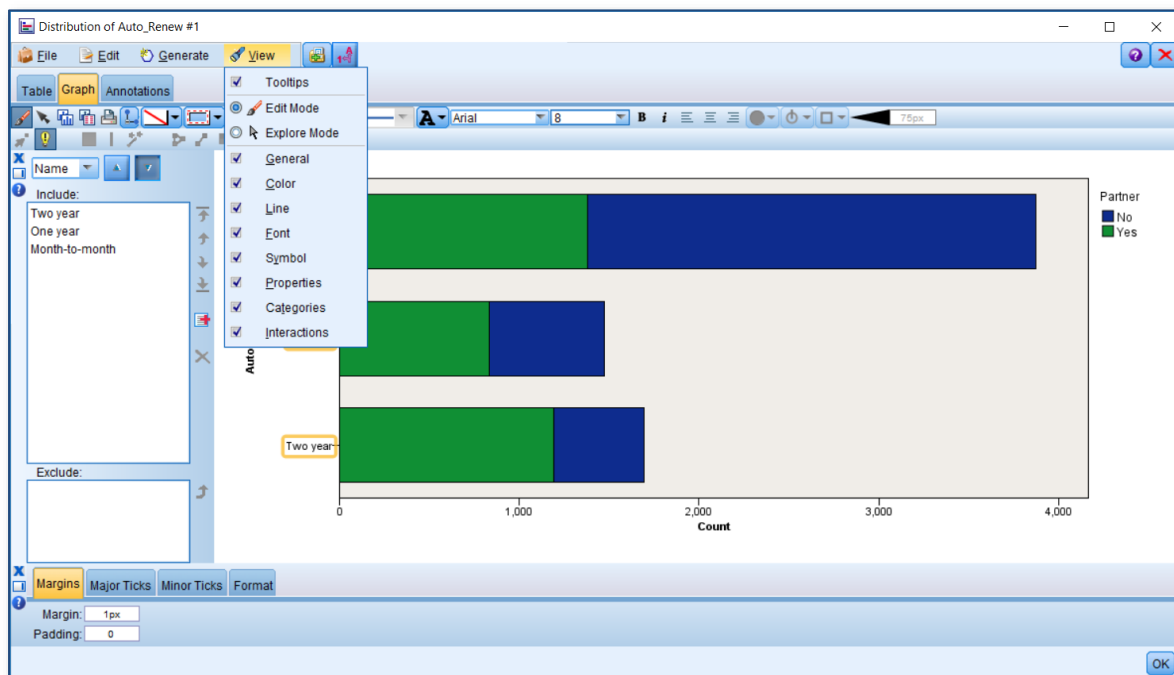


Figure 7.17 Graph Tab Output in Edit Mode

Unfortunately, it is often difficult to compare categories within stacked Distribution charts if we are dealing with groups of unequal size and comparing frequency counts as opposed to proportions. This is the equivalent of running a Matrix node (crosstab) and requesting only counts. To more easily compare the groups within the chart, we will need to run it again with a slightly different setting. To close the output, click:

OK

To re-run the chart, once again:

Double-click the Distribution node

Within the Plot tab, check the box marked:

Normalize by color

Figure 7.18 shows the edited Distribution node.

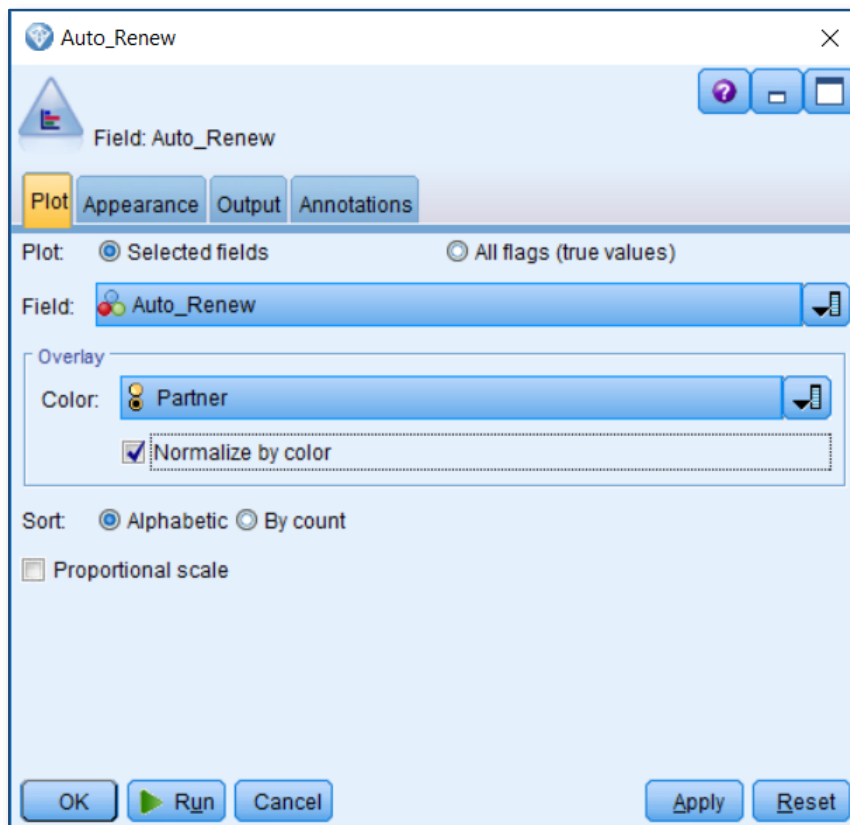


Figure 7.18 The Plot tab with the 'Normalize by color' option selected

To generate the chart, click:

Run

Figure 7.19 shows the resultant output.

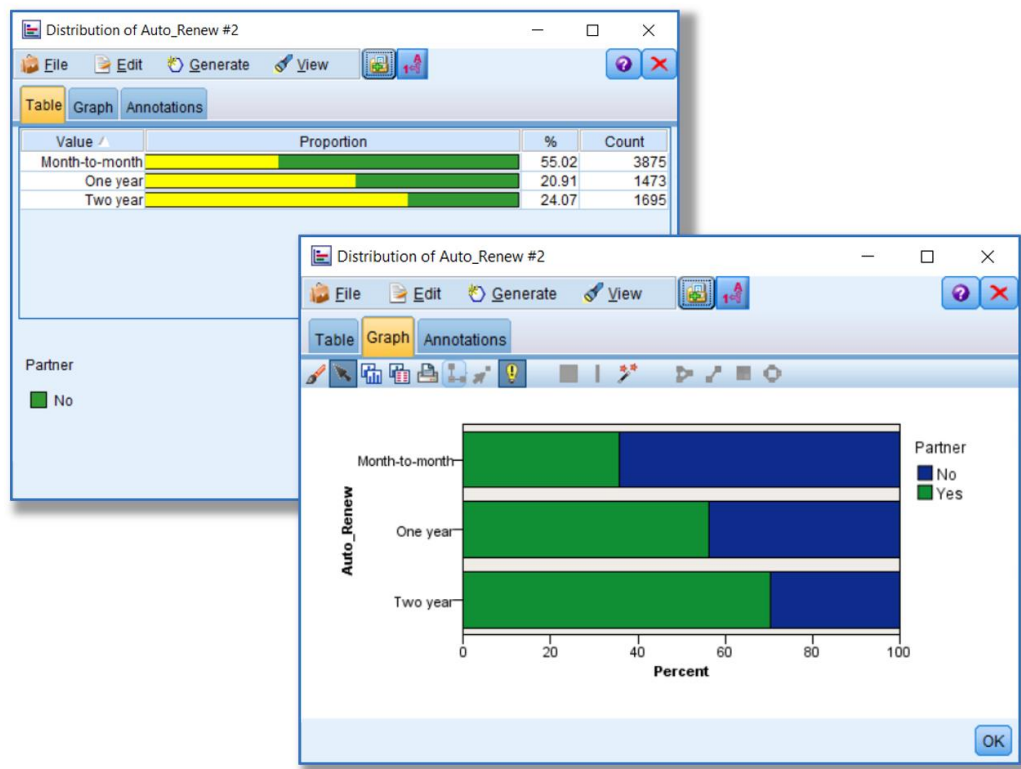


Figure 7.19 The Table and Graph tabs in the Distribution node output showing the effect of the 'Normalize by color' option

We can see that by switching on the 'Normalize by color' option, we can more easily compare the interaction between the variables in the chart. For example, we can see that the people on a 'Month-to-month' contract are much less likely to be living with a partner than those on a 'Two year' contract.

The Histogram Node



Histograms are a simple but effective way to view continuous data as a graphical distribution. By default, the Histogram node in Modeler creates 25 equal-ranged 'bins' of the continuous field to be graphed. To see the standard Histogram output, from the Charts palette:

Select and attach a Histogram node to the Data Source node

Figure 7.20 shows the Histogram node attached to the source node.

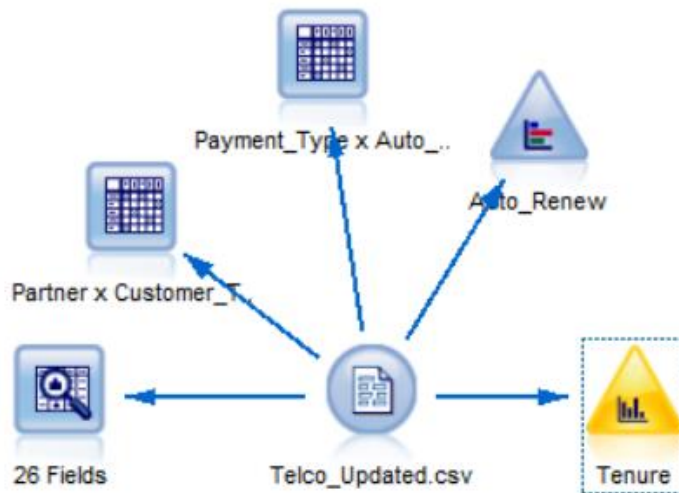


Figure 7.20 The Histogram node attached to the Data Source node in the current stream

To choose a field for charting in the Histogram:

Double-click the Histogram node

Within the Plot tab of the resulting dialog:

Choose the variable *Tenure* from the drop-down Field list

Figure 7.21 shows the completed procedure for a basic Histogram.

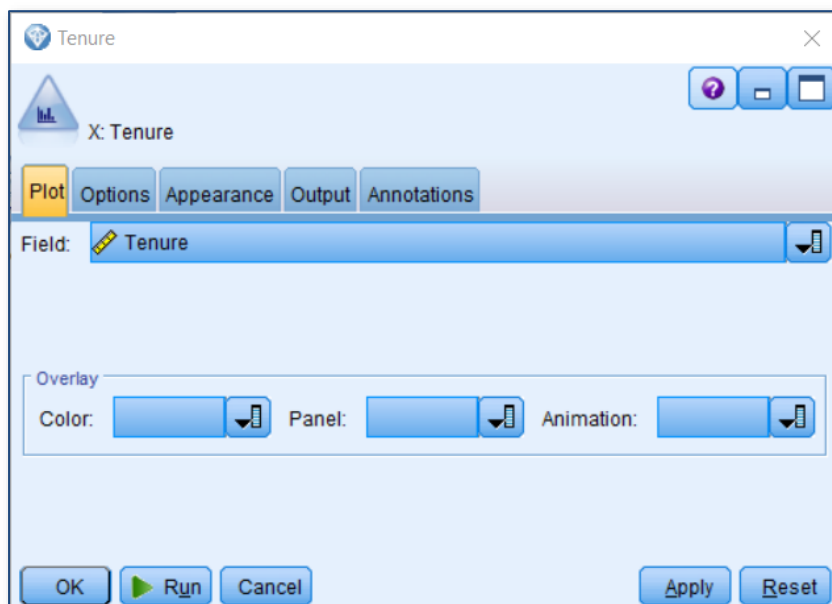


Figure 7.21 Requesting a histogram of Tenure

Click:

Run

Figure 7.22 shows the Histogram output.

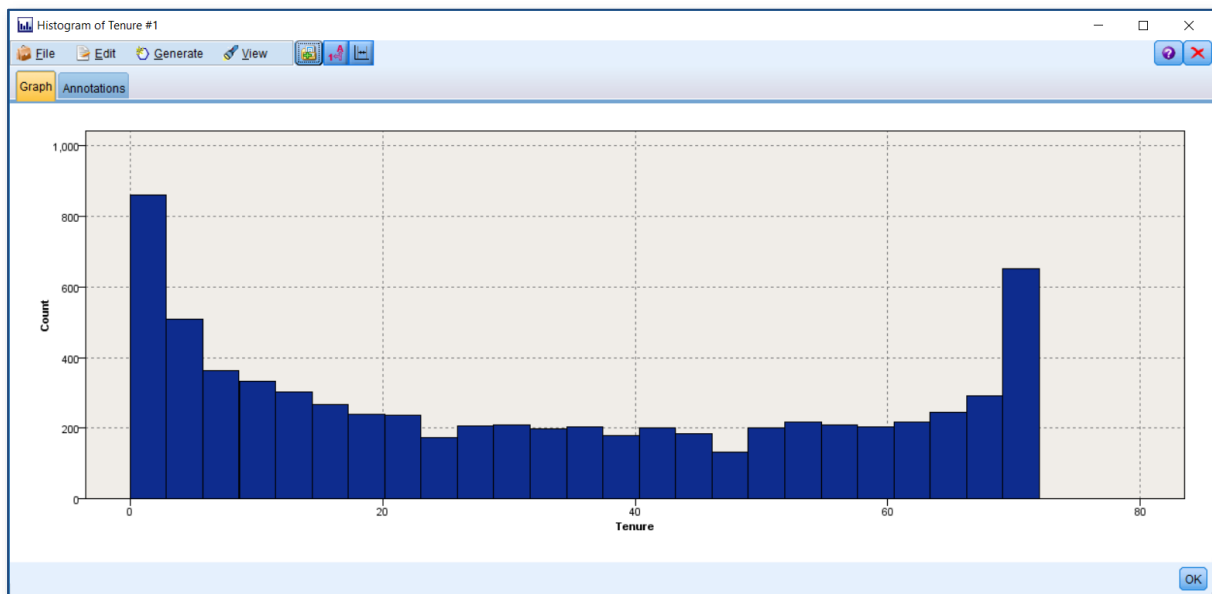


Figure 7.22 Histogram of Tenure

The histogram is comprised of an arbitrary number of individual bins (25). The range that each bin covers isn't important as the point of this chart type is to describe the shape of the distribution. However, the analyst can alter the number of bins to increase or decrease the level of detail in the chart. In this case, the histogram simply shows that there are two peaks around new customers (i.e. those with a small tenure value) and the most loyal customers (those with a high tenure value). Let's re-run the histogram but alter the number of bins and introduce a second variable.

Close the output and double-click the histogram node to edit it again

From the drop-down menu marked 'Color':

Choose the variable Churn

Now click the tab marked:

Options

In the Options tab:

Increase the value in the 'No. of bins' box to 35

Figure 7.23 shows the completed Histogram Plot and Options tabs.

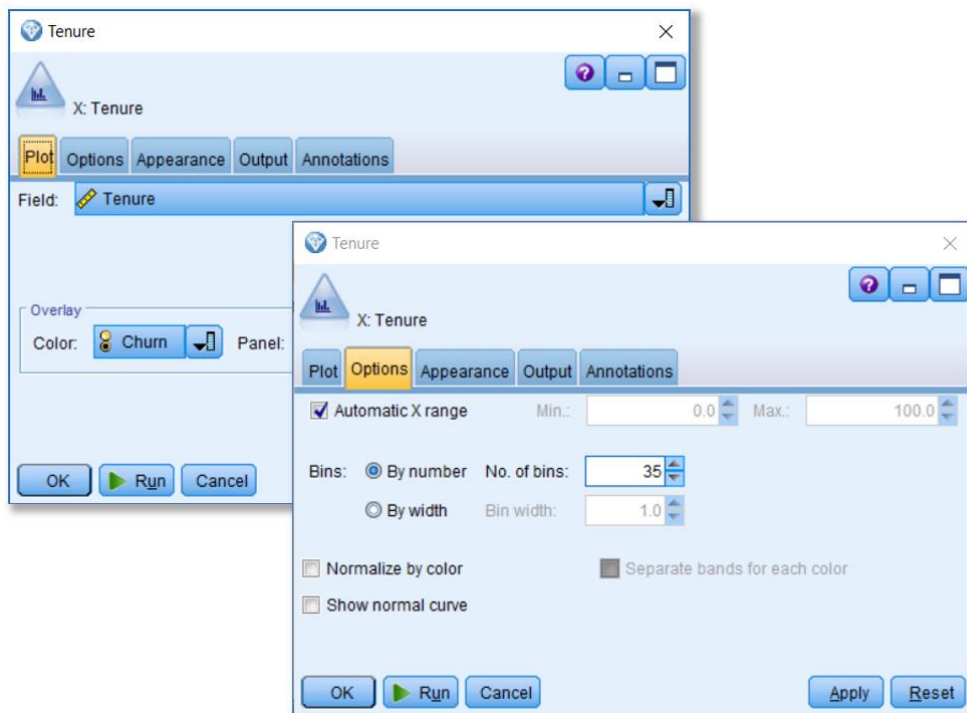


Figure 7.23 Editing the Histogram to increase the number of bins and include the field Churn

To see the effect of these changes, click:

Run

Figure 7.24 shows the new histogram output.

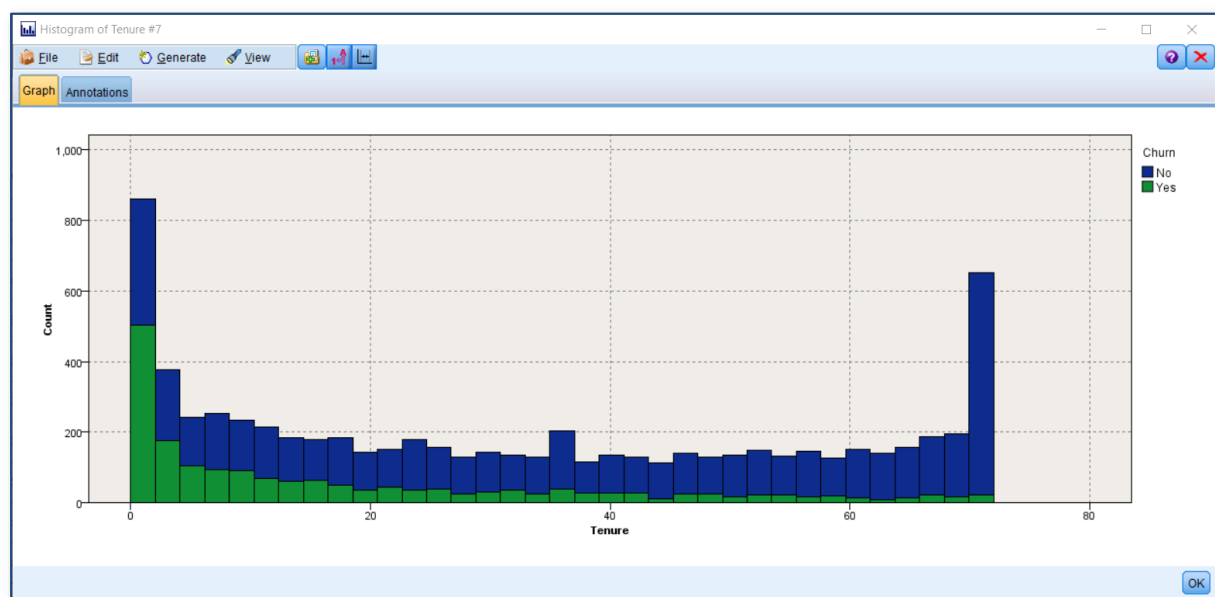


Figure 7.24 Histogram of Tenure by Churn with 35 bins

Now we can see a little more detail in the distribution of values in the chart. We can also detect that the proportion of churners seems to decrease as the tenure increases. Just as we did with the Distribution chart earlier, we can normalise this view to better see how the proportion of churn changes over the tenure of the customer groups. Once again:

Close the output and double-click the histogram node to edit it

Select the tab marked:

Options

In the Options tab:

Check the box marked 'Normalize by color'

Figure 7.25 shows the completed Histogram Plot and Options tabs.

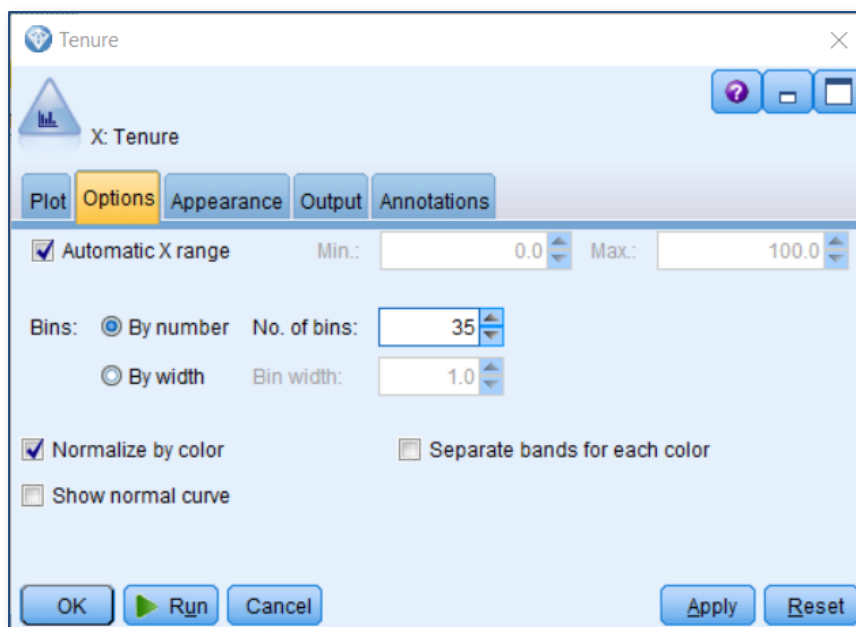


Figure 7.25 Requesting a histogram with normalised colour groupings

To see how this affects the output, click:

Run

Figure 7.26 shows the new histogram output.

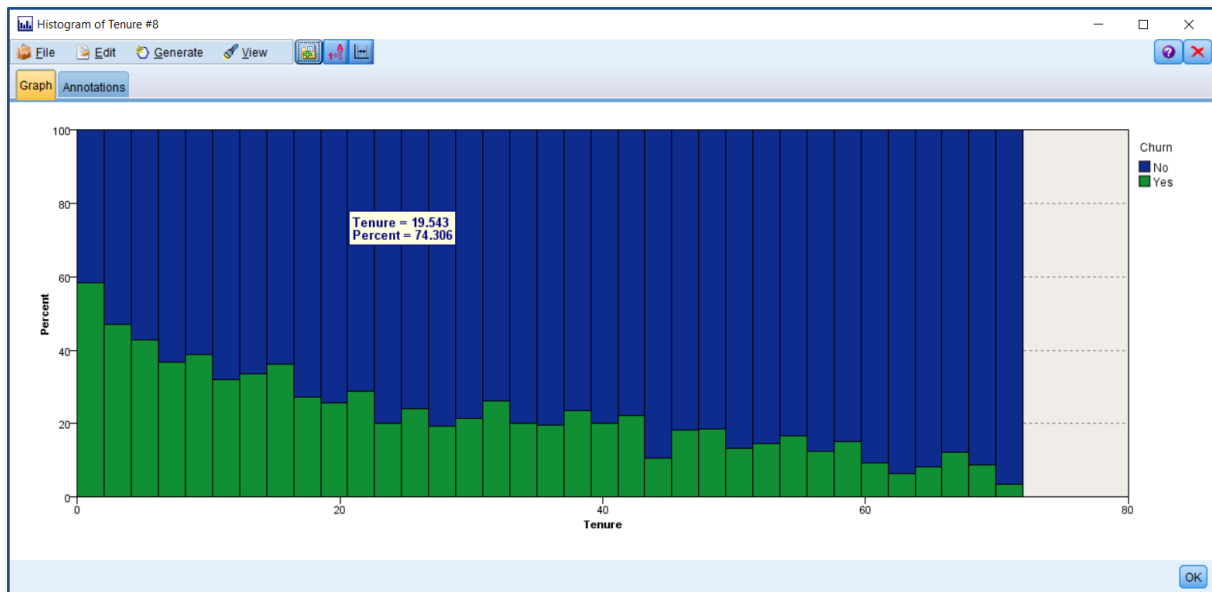


Figure 7.26 Histogram of Tenure by Churn normalised by colour

We can see exactly now that the proportional decrease in the churn rates occurs fairly smoothly throughout the distribution. We must bear in mind that when viewing charts like this, the actual counts of cases may vary by quite a bit throughout the distribution so it's a good idea to view the chart without normalising the data first. These kinds of charts are especially useful when trying to detect the point or threshold at which a change in a continuous value has particular effect such as the proportion of people accepting an offer based on discount value or the number of faults that occur across a certain temperature range.

The Means Node



The Means node allows us to compare the arithmetic averages of different groups in a simple tabular output format. To illustrate, from the Output palette:

Select and attach a Means node to the Data Source node

Figure 7.27 shows the Means node attached to the source node in the existing stream.

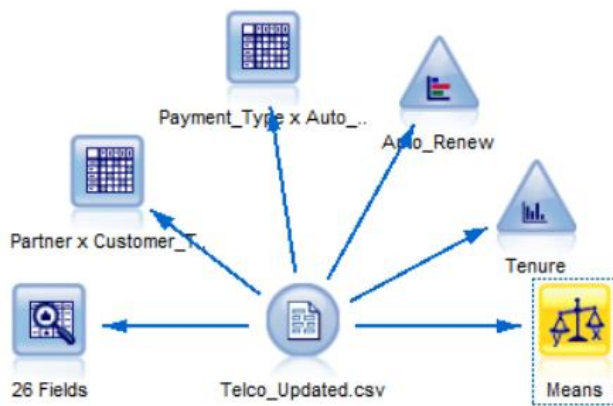


Figure 7.27 The Means node attached to the Data Source node in the current stream

To show how the procedure works:

Double-click the Means node

The resulting dialog allows us to specify a categorical variable to define the groups we wish to compare and one or more continuous ‘test’ fields from which the mean values are calculated. Within the Settings tab of the edited Means node, from the drop-down Grouping field list:

Choose the field `Payment_Type`

In the area marked ‘Test field(s)’, click the field selection button and:

Choose the field `Monthly_Rate`

Figure 7.28 shows the completed tab.

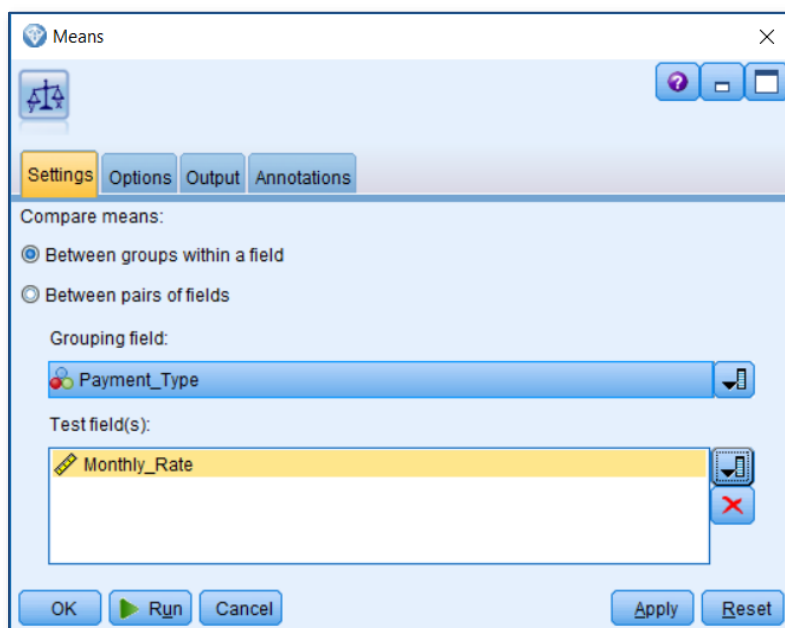


Figure 7.28 The completed Setting tab within the Means node

To generate the Means output, click:

Run

Figure 7.29 shows the Means output.

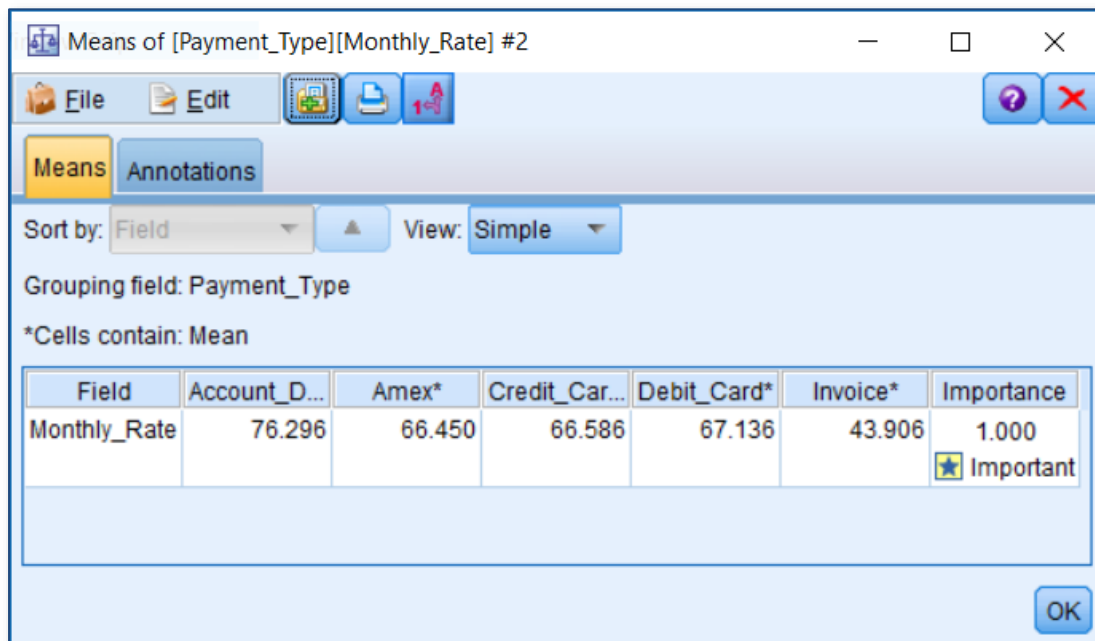


Figure 7.29 The Means output showing the average Monthly rate values for the five different payment types

We can see that the output simply shows the five mean values of the variable 'Monthly_Rate' for each of the different payment types. It also indicates that at least one of these differences is 'Important'. This refers to an F-test that the procedure runs to test for statistical significance. If the probability of the F-test is below 0.05 (the 5% level) then the output is marked as 'Important'. You may notice that the 'view' button in the output indicates that the output is shown in 'Simple' mode.

Click the 'View' button to switch the output to 'Advanced mode'.

Figure 7.29 shows the 'Advanced' output from the Means node.

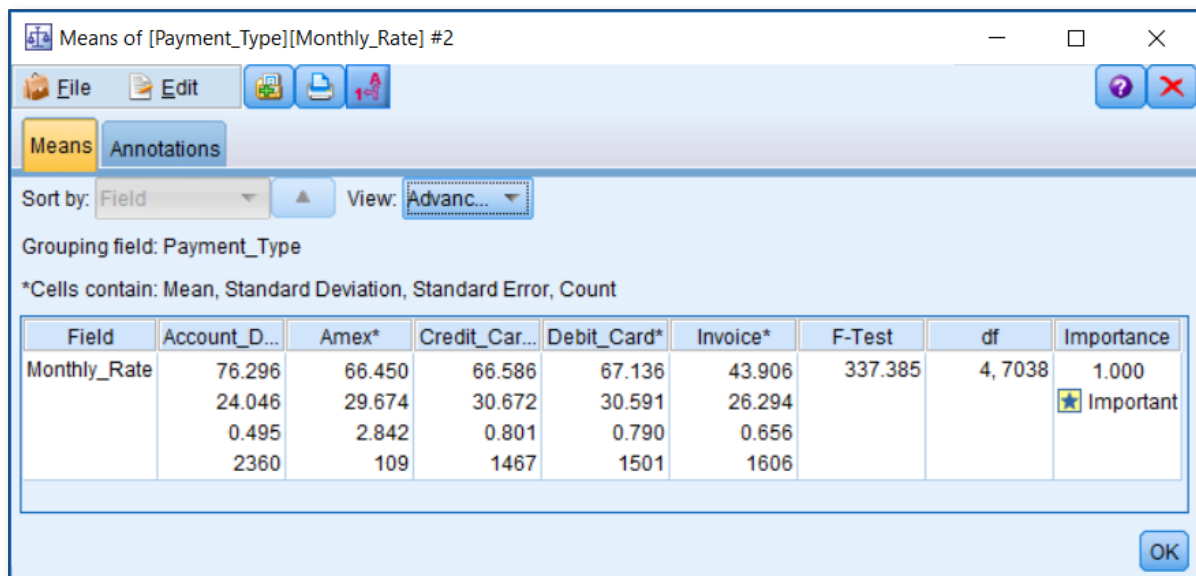


Figure 7.30 The Means output showing the 'Advanced' view

The Advanced output shows not just the mean values but the mean, standard deviation, standard error and frequency count, in that order, from top to bottom within each group column. It also shows details of the F-test used to compare the group means and to assess whether any of the differences between can be regarded as 'statistically significant'.

The Means procedure also allows users to calculate linear correlations between specific pairs of variables. To see how this is done:

Close the output and double-click on the Means node again

Choose the radio button marked 'Between pairs of fields'

From the drop-down variable menu entitled 'Field one' choose:

Tenure

From the drop-down variable menu entitled 'Field two' choose:

Total_Bill

Now click:

Add

Figure 7.31 shows the completed dialog at this stage.

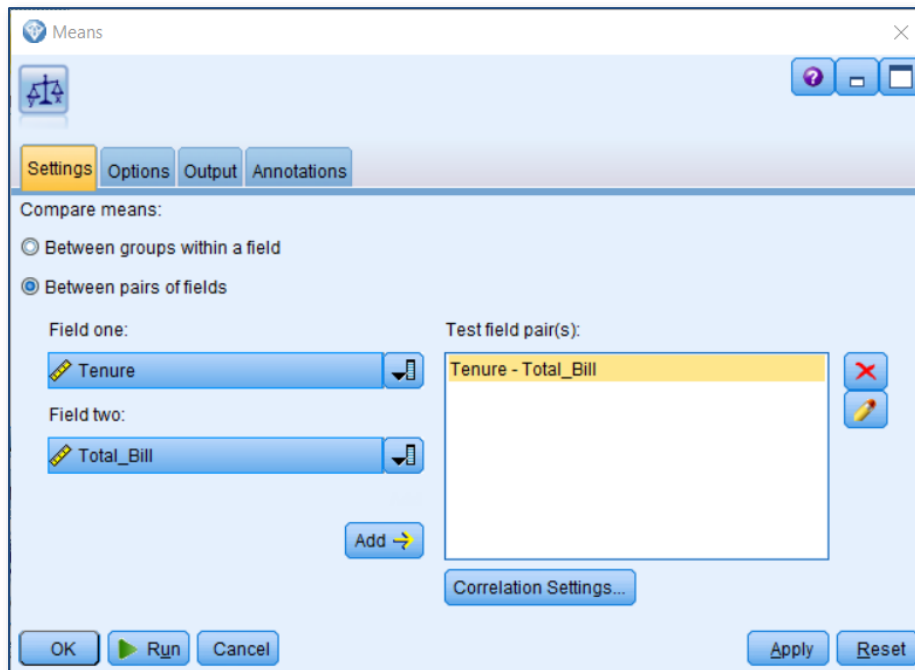


Figure 7.31 Partially completed dialog for comparing pairs of continuous fields in the Means procedure

Before we run the procedure, we can request that the correlation value reveals the actual linear correlation coefficient. To do so, click:

Correlation Settings

In the pop-up sub-dialog, choose the radio button:

Define correlation strength by absolute value

Figure 7.31 shows the completed sub-dialog.

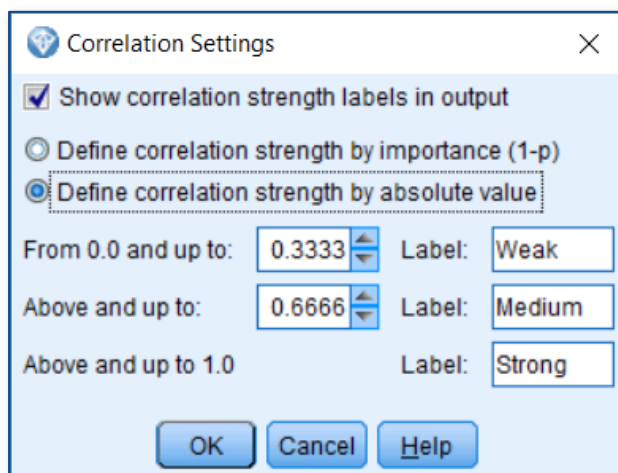


Figure 7.32 Sub-dialog within the Means procedure for requesting that the absolute values for correlations are displayed

You can see from the sub-dialog that the user can set custom threshold values for the correlation coefficient. In fact, linear correlation values run from -1 to +1. The closer the value is to either of those extremes, the stronger is the apparent linear relationship between the two fields. Conversely, values close to zero show weaker (or even random) relationships.

To view the output from the procedure, click:

OK

Run

Figure 7.33 shows the resulting output.

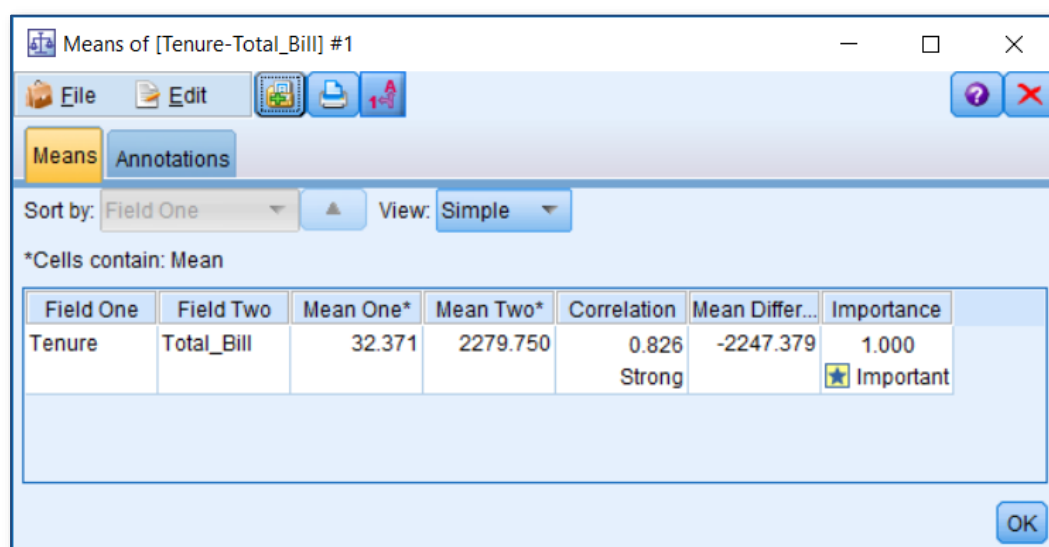


Figure 7.33 Means output for a pair of continuous fields showing absolute correlation value

We can see that the output now shows the two mean values for each of the continuous fields. However, it also shows the linear correlation of 0.826 which is marked as 'Strong' and reflects a positive linear relationship between the fields.

The Plot Node



Plot

One way to graph the relationships between continuous fields is to generate a scatterplot. In Modeler, the Plot node allows us to do this. To demonstrate this, from the Graphs palette:

Choose a Plot node and attach it to the Data source node in the existing stream

Figure 7.34 shows the updated stream.

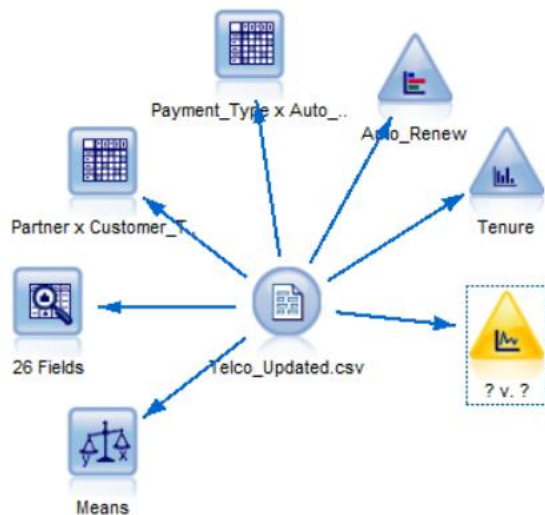


Figure 7.34 The updated stream with the Plot node attached

Now we can illustrate the linear relationship we observed using the Means node in the last example.

Double-click the Plot node

In the drop-down field menus:

Assign Tenure to the X-axis

Assign Total_Bill to the Y-axis

In the drop-down field chooser titled 'Color'

Choose the field Customer_Tier

Figure 7.35 shows the completed dialog.

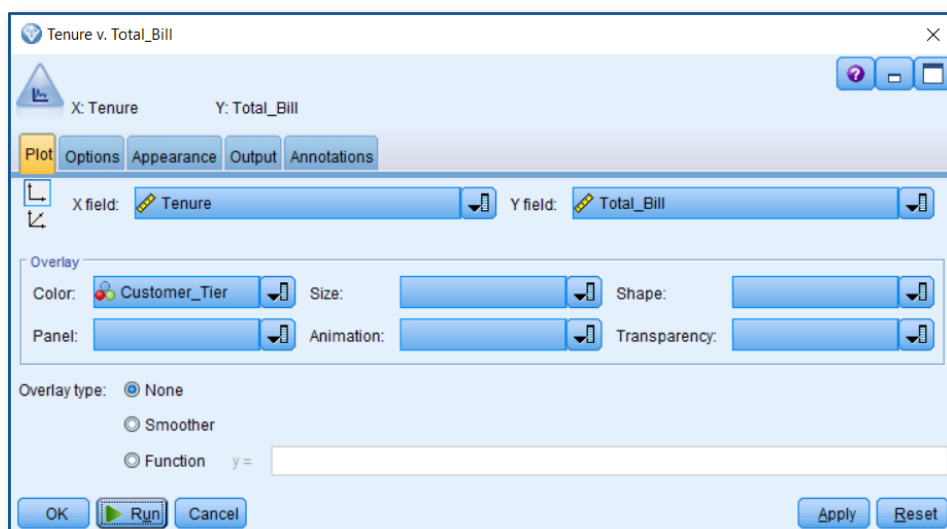


Figure 7.35 Completed dialog for creating a scatterplot of Tenure by Total_Bill coloured by Customer_Tier

Now run the procedure

Figure 7.36 shows the output.

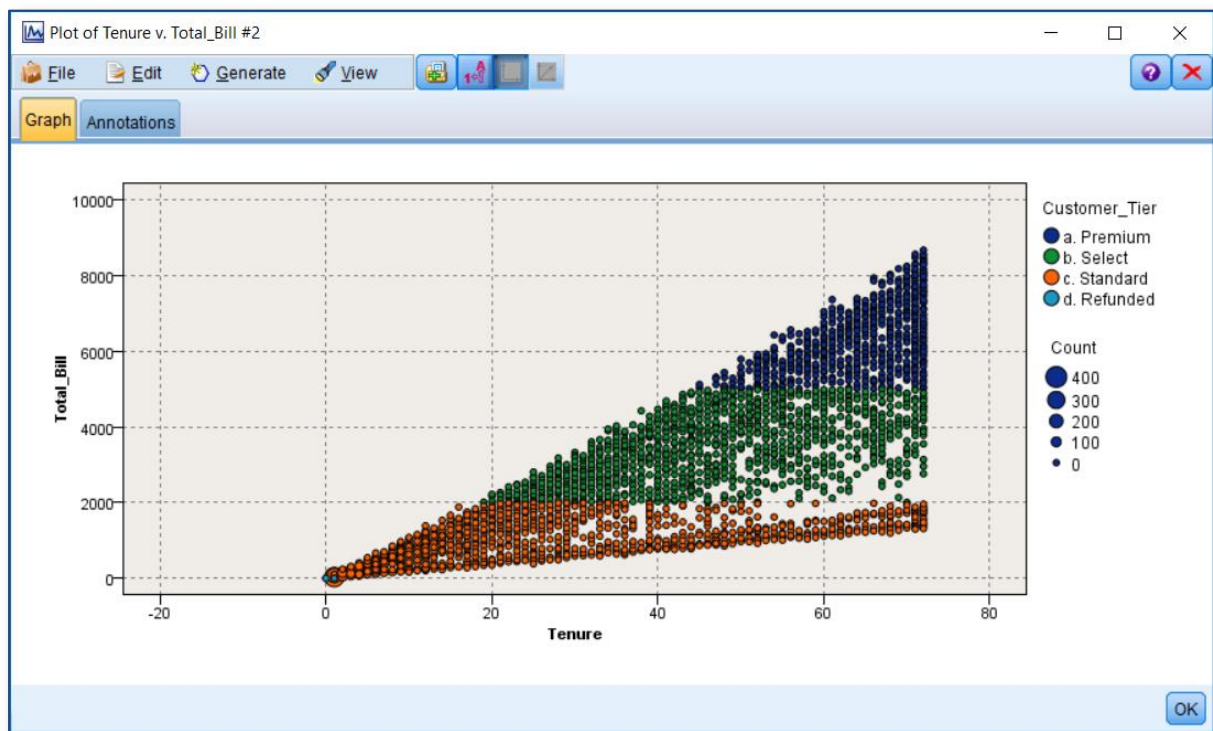


Figure 7.36 Output from the Plot procedure showing relationship between Tenure by Total_Bill coloured by Customer_Tier

The scatterplot displays the strong linear relationship that was detected in the earlier correlation. The points themselves are binned so that larger points represent more data records. The customer tier variable clearly delineates the different strata in the scatterplot itself. In fact, the Plot node allows us to vary the size of the points according to values in another field and to create separate (panelled) scatterplots according to the levels of a specified grouping variable.

The Statistics Node



The Statistics node offers a simple way to view summary statistics for many fields at once. As such, its functionality overlaps somewhat with the Data Audit node. Moreover, like the Means node it allows us to view correlations between variables. To investigate the Statistics node, from the Output palette:

Choose a Statistics node and attach it to the Data source node in the existing stream

Figure 7.37 shows the updated stream.

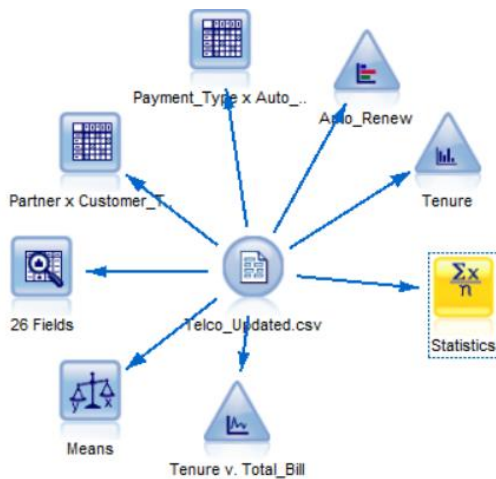


Figure 7.37 The updated stream with the Statistics node attached

Double-click the Statistics node

Using the Field selection button add the following variables to the Examine box:

Tenure

Monthly_Rate

Total_Bill

Using the second Field selection button (bottom) add the following variables to the Correlate box:

Average_Monthly_Bill

Figure 7.38 shows the Statistics dialog at this stage.

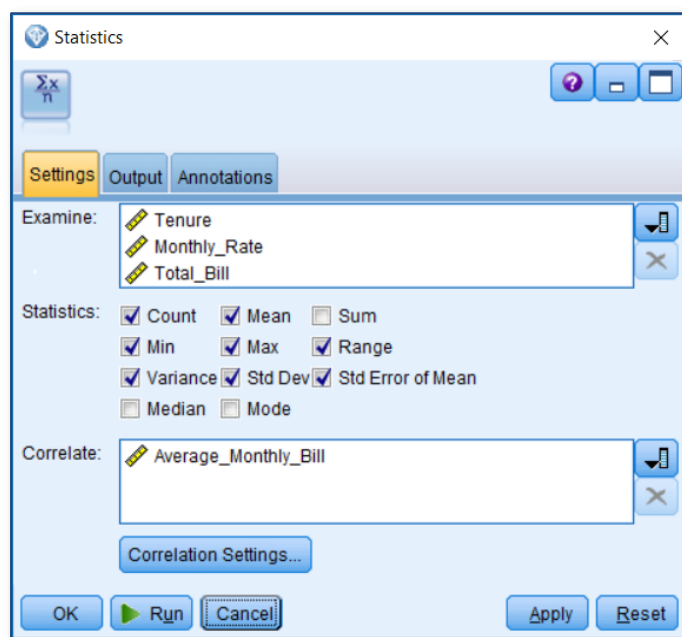


Figure 7.38 The Statistics Node dialog

Just as with the Means procedure, we can request that the correlation value reveals the actual linear correlation coefficient. To do so, within the Statistics node dialog click:

Correlation Settings

In the pop-up sub-dialog, choose the radio button:

Define correlation strength by absolute value

Figure 7.39 shows the completed sub-dialog.

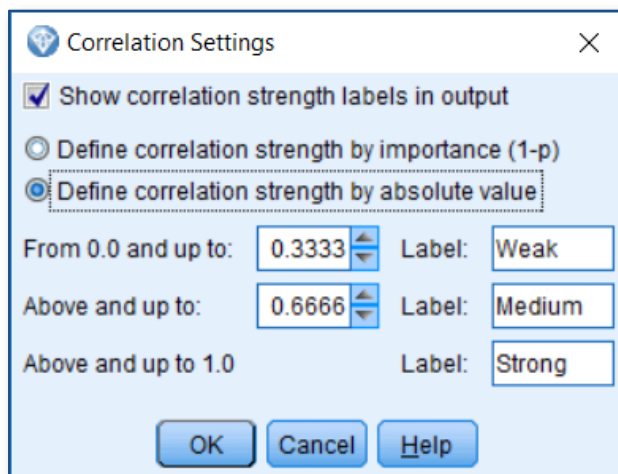


Figure 7.39 Sub-dialog within the Statistics procedure for requesting that the absolute values for correlations are displayed

Click:

OK

Run

The output from the Statistics procedure is shown in figure 7.40.

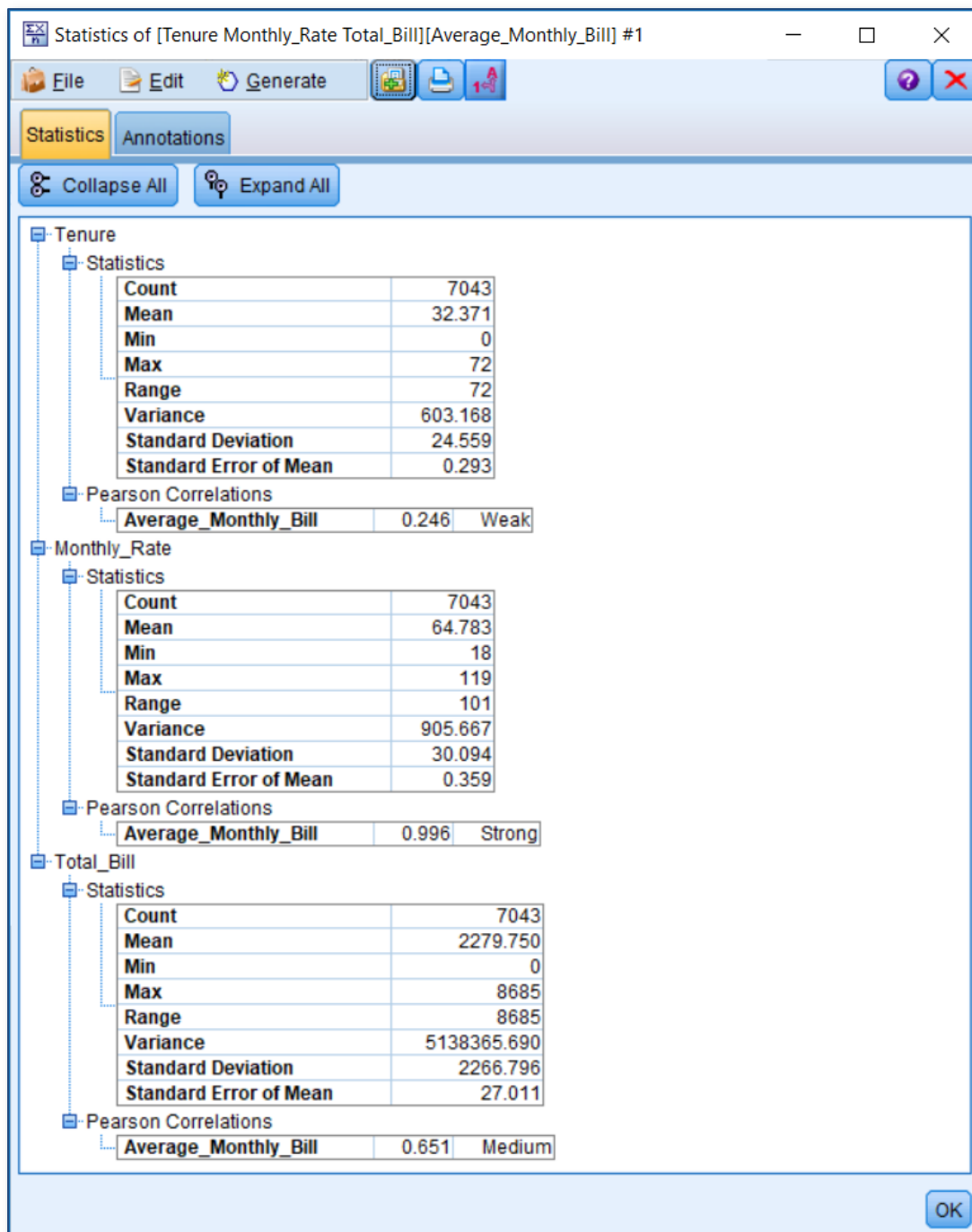
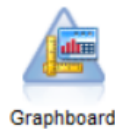


Figure 7.40 Output from the Statistics procedure

We can see from the output that summary statistics are displayed for each of the selected fields in turn. Furthermore, each block of summary statistics displays the correlation value for the linear relationship between the field `Average_Monthly_Bill` and the selected variable. Although the Statistics node generates output that can also be calculated by other procedures, it worth pointing out that the Generate menu within the output will create a Filter node allowing the user to filter out fields with weak correlations against a key variable. This can be useful for identifying those fields that are strong predictors of a particular continuous target.

The Graphboard Node



As a final example of procedures to help us explore data, we will look at the Graphboard node. This is powerful graphical procedure that will 'suggest' chart types to the user based on the combination of field types that they select. To see this, from the Graphs palette:

Choose the Graphboard node and attach it to the Data source node in the existing stream

Figure 7.41 shows the updated stream.

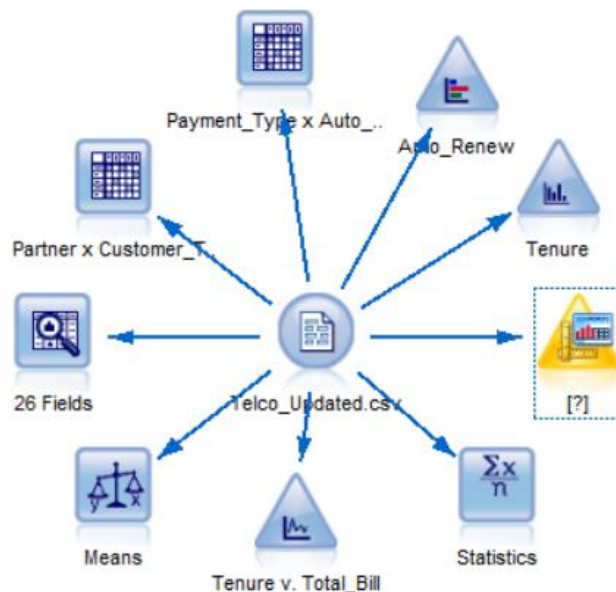


Figure 7.41 The updated stream with the Graphboard node attached

Double-click the Graphboard node

One of the first things we notice when choosing fields within the Graphboard dialog is that different chart types appear based on the selection of fields that we choose (ctrl-click is used to make multiple selections). Figure 7.42 illustrates this. The screenshot shows a range of different chart types that are suggested based on the different field types that have been clicked in the source variable list on the left-hand side.

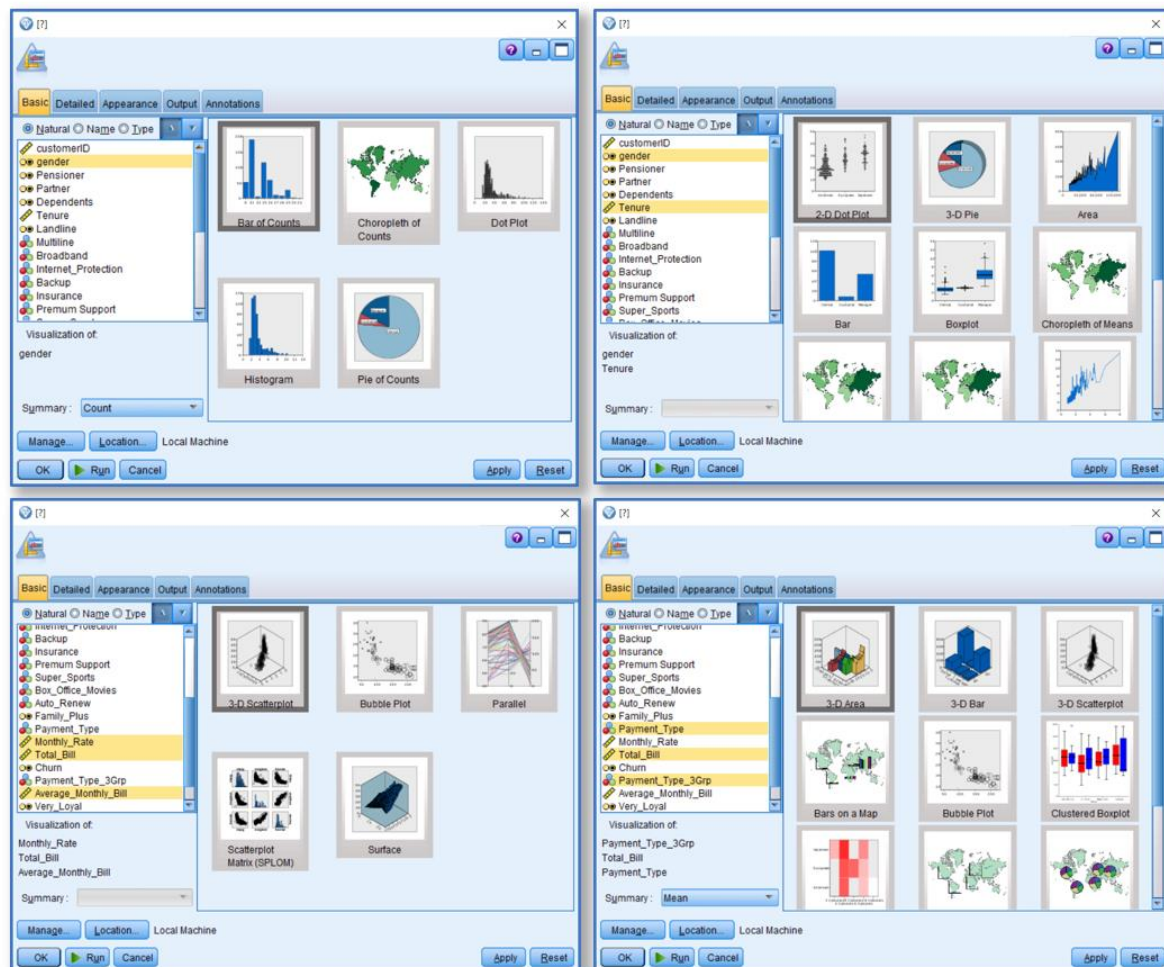


Figure 7.42 The Graphboard dialog automatically suggesting different chart types based on the combination of selected variables

Using control-click, select the following fields:

Box_Office_Movies

Payment_Type

Total_Bill

Because we have selected two categorical fields and a continuous field, the Graphboard node immediately offers a range of chart types that might be appropriate for that combination of variables. Within the portfolio of chart types select:

Heat Map

Figure 7.43 shows this selection being made in the Graphboard dialog.

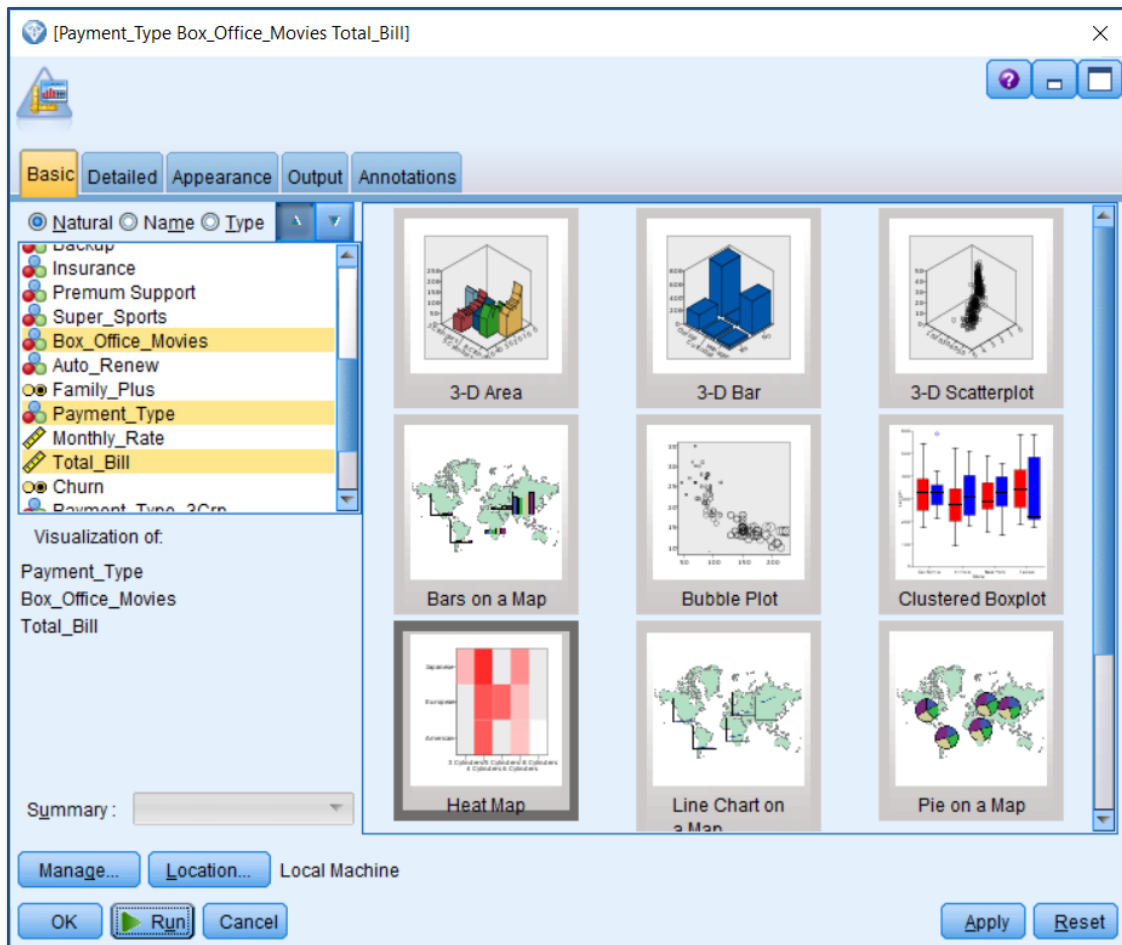


Figure 7.43 Selecting a Heat Map in the Graphboard dialog

As a chart type, heat maps are especially useful when we are looking at the relationship between two categorical fields (with several categories) in terms of a third summary measure. Examples of this include using heat maps to see how average profit varies by region and customer segment or to compare the mean household income values of different property types by age group. In this example, the heat map will show the mean values for total bill for each combination of the categories within `Payment_Type` and `Box_office_Movies`. To generate the chart, click:

Run

Figure 7.44 shows the generated heat map.

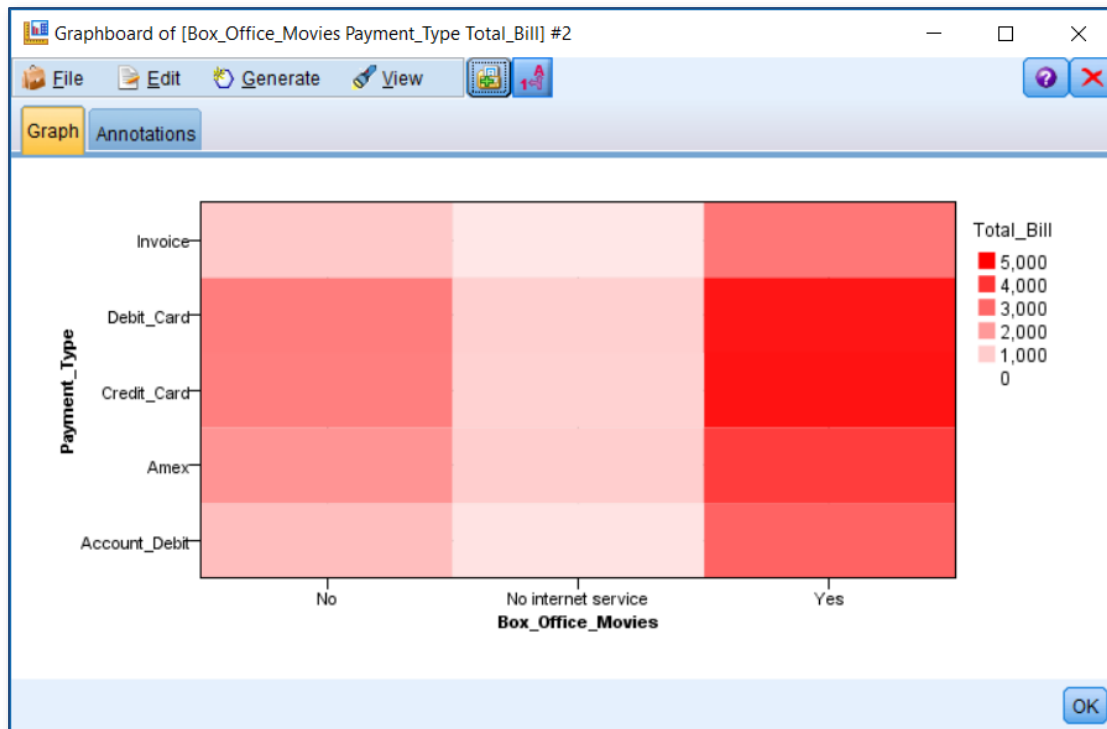


Figure 7.44 A heat map showing the mean values for `Total_Bill` by each category of `Box_Office_Movies` and `Payment_Type`

The heatmap makes it very easy to compare these groups quickly as it reveals the ‘hot-spots’ where the average total bill value is highest. As we saw, earlier there are many other chart types within the Graphboard node and most analysts will find it worthwhile to familiarise themselves with these various options.

Section 8:

Building a Predictive Model



- Model Types
- Classification Models
- Association Models
- Segmentation Models
- The CHAID Node
- Model Nuggets

As we move through the CRISP-DM process we are of course working towards the goal of creating a deployable model that will enable better decision making within our application area. In this section, we can begin the process of learning about building and interpreting predictive models. Before we do so, let's turn our attention towards the main kinds of modelling techniques that SPSS Modeler offers us.

Model Types

There are many model-building algorithms within SPSS Modeler. Indeed, most of these algorithms contain sub-methods and optional settings that increase the number of possible modelling methods by an order of magnitude. Generally, the modelling techniques fall into the following groups:

- Classification
- Association
- Segmentation

Classification Models

Classification models use the values of input fields to predict the values of specific target fields. The target fields themselves can be categorical or continuous in nature. Not surprisingly for a predictive analytics workbench, these predictive modelling techniques are the most numerous type of algorithm in SPSS Modeler. Classification models include rule-based techniques and decision trees such as CHAID, C&R Tree, Random Trees and C5 algorithms. They also include classical statistical techniques from the regression family such as linear and logistic regression as well as a number of machine learning algorithms such as neural networks, support vector machines and bayesian networks.

Association Models

Association models construct rule sets to discover relationships between multiple categories in a set of fields. Association models differ from the classification techniques discussed earlier in that the fields can act as *both inputs and targets*. With this kind of model therefore, we are more interested in the interrelationships between the variables rather than the outcomes recorded in a single field. Association models are useful when we wish to understand the co-occurrence of events. As such, they can be used to detect which products are purchased together in a single transaction (basket analysis) or over a period of time (affinity analysis). Moreover, we can use association models to generate rules that estimate how likely it is that a specific product will be purchased given the presence of other products. In the same sense, we can use these techniques to identify component parts that need to be replaced based upon patterns of failure, symptoms that tend to co-occur based upon medical conditions or patterns of behaviour that co-occur in events links to fraud and crime. The Apriori and Carma association algorithms in SPSS Modeler are used to address precisely these kinds of problems. An extension of this capability is the Sequence Detection algorithm which tries to uncover sequential patterns in time-structured data.

Segmentation Models

Segmentation models do not rely on target or output fields. They are designed to create segments, or clusters of records that are similar to one another. Because these techniques are not predictive in nature, they only require input fields. Segmentation models are useful when attempting to characterise a customer database in terms of demographics and shopping behaviour to drive deeper customer insight and provide more compelling products or services. Segmentation (or “clustering” models) can also be used to group similar stores or branches together in terms of their sales patterns to more effectively resource them or to uncover different modes of behaviour, for example, when trying to detect fraud. Such models are evaluated not in terms of their accuracy but rather in terms of their ability to capture interesting groupings in the data that might otherwise have gone undetected. The three main segmentation (cluster) techniques within SPSS modeler are the Two-Step, K-Means and Kohonen network algorithms. A related procedure is the Anomaly Detection node, which rather than trying to create homogeneous groups, identifies unusual cases, or outliers, that do not conform to patterns of “normal” data. Such an approach is specifically tuned to detect unusual events or behaviours and can be useful when identifying sub-groups of cases that represent clandestine behaviour in people, anomalous activity in asset performance or records where a key outcome is difficult to detect using standard predictive models.

The CHAID Node



To demonstrate our first example of building a predictive model we will use a well-documented decision tree algorithm called as CHAID. The CHAID acronym stands for Chi-square Automatic Interaction Detection. It is a technique that produces a highly visual model in the form of a decision tree. The tree itself is comprised of a hierarchy of fields selected for inclusion in the model because each one is deemed to have a statistically significant relationship with the target field (as measured by the Chi-square statistic). To see how the CHAID node works, from the Section 8 folder open the following stream.

Section 8 Start.str

Figure 8.1 shows this stream. Note that it reads an updated version of the Telco file.



Figure 8.1 The SPSS Modeler stream 'Section 8 Start.str'

You will notice that this is the same dataset that we used in the previous chapter. The file contains a field ('Churn') that indicates whether or not the customer has cancelled their contract with the telecommunications company. We can start the process of building a model to predict churn by attaching a CHAID node to the source node. From the modelling palette:

Select and attach a CHAID node to the Data Source node

Figure 8.2 shows the attached Matrix node.



Figure 8.2 CHAID node attached to the source data file

You may notice that the CHAID node is immediately renamed 'Churn'. This is because the churn field is set as the target within the Type tab in the source node. In fact, care should be taken to inspect the field types so that no redundant variables (e.g. email address, transaction number etc.) are included, as well as fields that contain information that could only have been available if the outcome in the target field was known (and so are self-fulfilling). At this stage, we could simply execute the CHAID node and, assuming there were no issues with the data, it

would automatically generate a model for us. Before we do so however, we will have a look at the various options for model-building in the algorithm itself.

Right-click the CHAID node to edit it

Figure 8.3 shows the default Settings tab within the edited CHAID node.

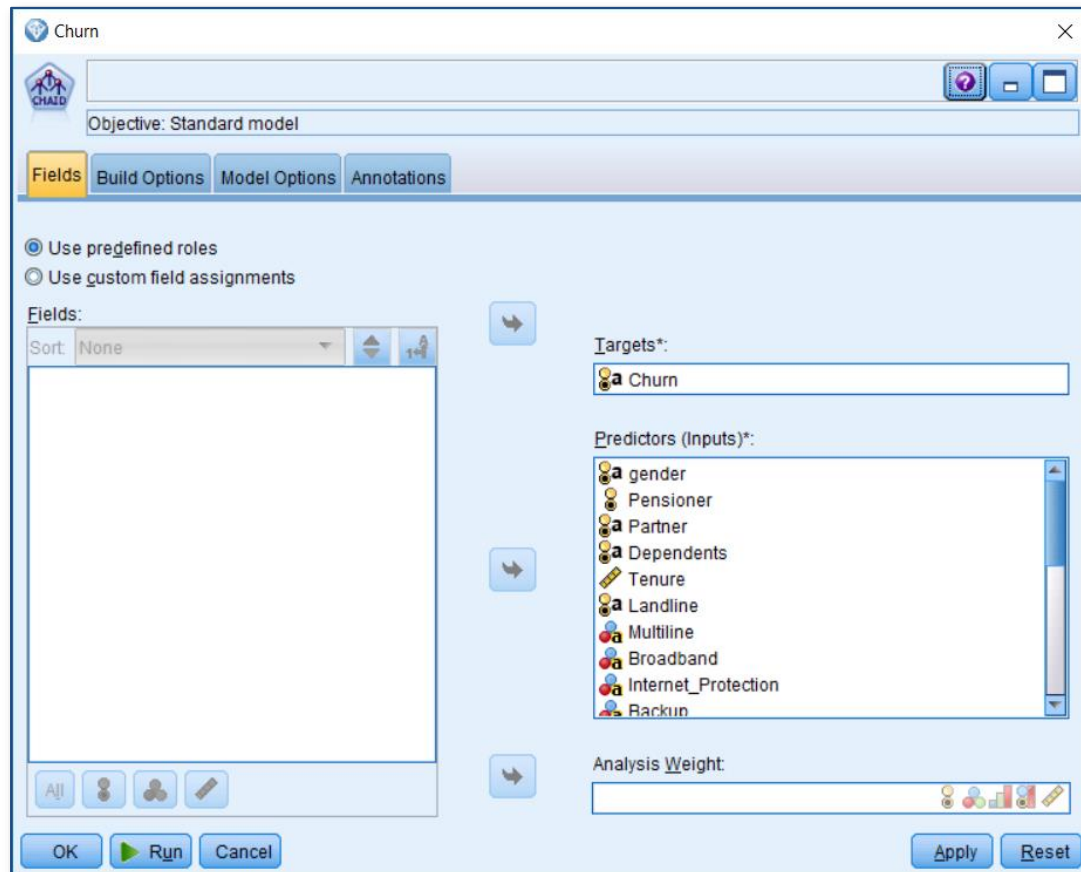


Figure 8.3 The default Fields tab within the edited CHAID node

You can see from the Fields tab that the variables have been assigned to the target and input roles in the CHAID node. This is because of how their roles in the Type tab of the source node have been set. Note also that you can click the radio button marked 'Use custom field assignments' to overwrite this default setting and reassign them as you wish. To continue, click the tab marked:

Build Options

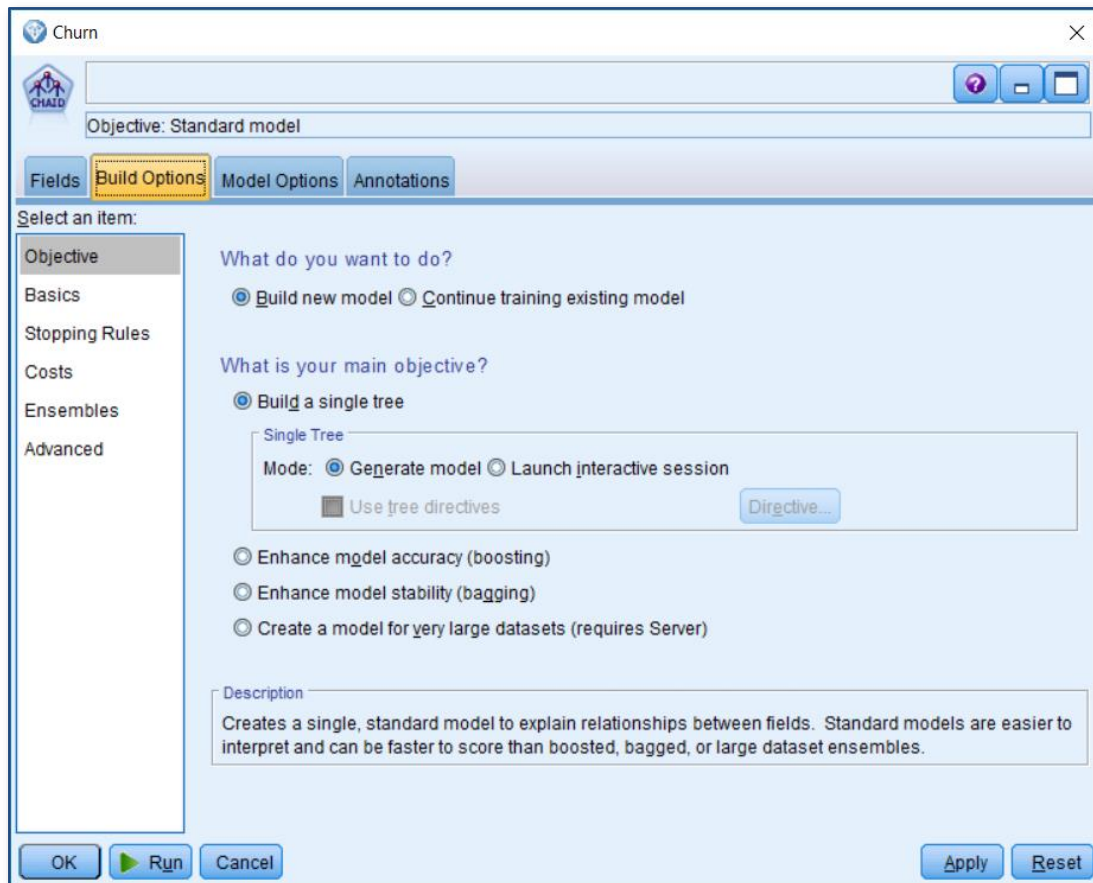


Figure 8.4 The Build Options Tab showing the model-building objectives in the CHAID node

The Build Options tab contains many controls concerning the nature of the model that we will generate with the CHAID procedure. At this stage, we are going to request that we build a model interactively, so we can view the how the model is created at our own pace. To request this option, click the radio button marked:

Launch interactive session

To start the model building process, click:

Run

Figure 8.5 shows the initial model output displayed in an interactive CHAID tree window.

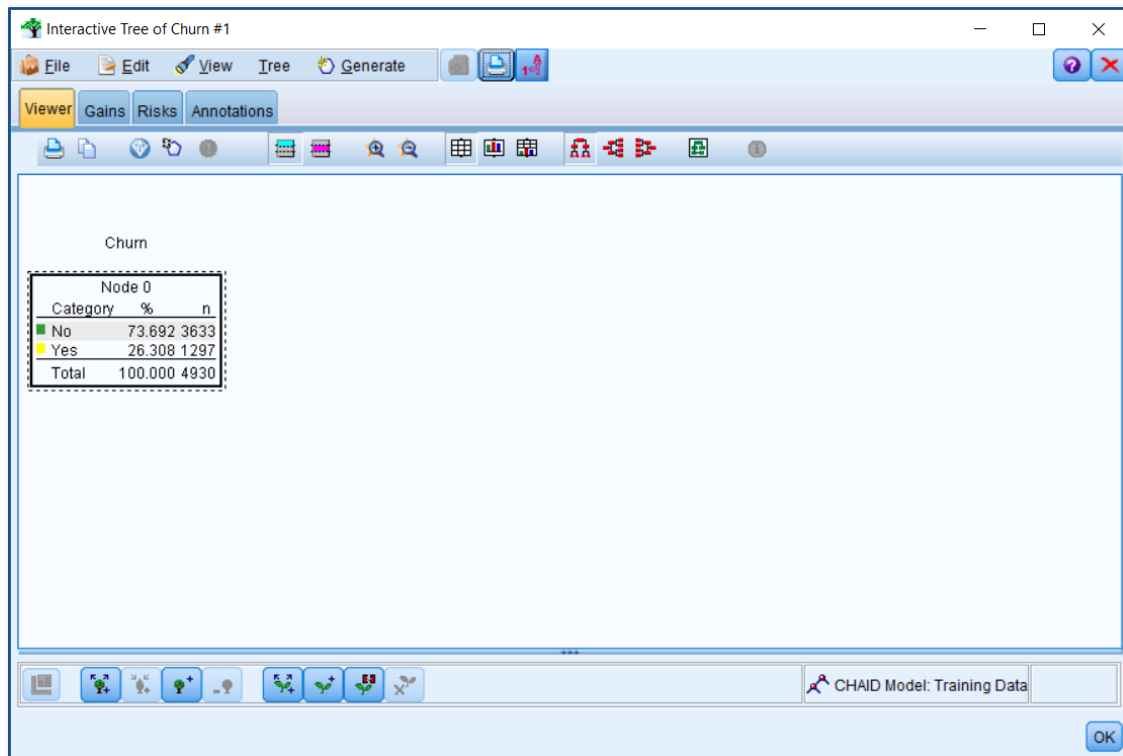


Figure 8.5 The Interactive CHAID tree (note: random seed = 701499504)

The initial node within the interactive CHAID tree shows a random sample representing approximately 70% of the data. This sample called the “tree growing set”. Around 30% of the data is held back from the ‘model training’ process and can be used to verify that the model provides reliable results. This first node shows the frequency and percentage breakdown of the two groups within the target field. Of the 4,930 people in the sample, around 74% have not yet churned. The aim of the model is find the best variables that discriminate between current customers and those who have defected. To do so, the CHAID model tests each and every independent variable against the target field to see which one should be entered first. This is the equivalent of running a crosstab of each field against the target. To use a continuous field, the CHAID algorithm breaks the variable into binned deciles and so effectively converts it to a categorical field so that the comparison can be made. At this stage in the interactive model, the CHAID tree has already calculated which fields should be entered based upon the probability values derived from a series of chi-square tests. We can view the results of these tests by clicking the custom split button:



Figure 8.6 shows the Define Split dialog. This dialog not only allows us to view the results of the Chi-square significance tests, but also to view and edit how the algorithm has defined a variable ‘split’. The ‘split’ refers to where the algorithm has re-grouped the categories in a field to simplify the variable whilst preserving its predictive power. For example, if the model finds that it can group together a number of locations within a variable representing region without exceeding a specified probability level, it will do so.

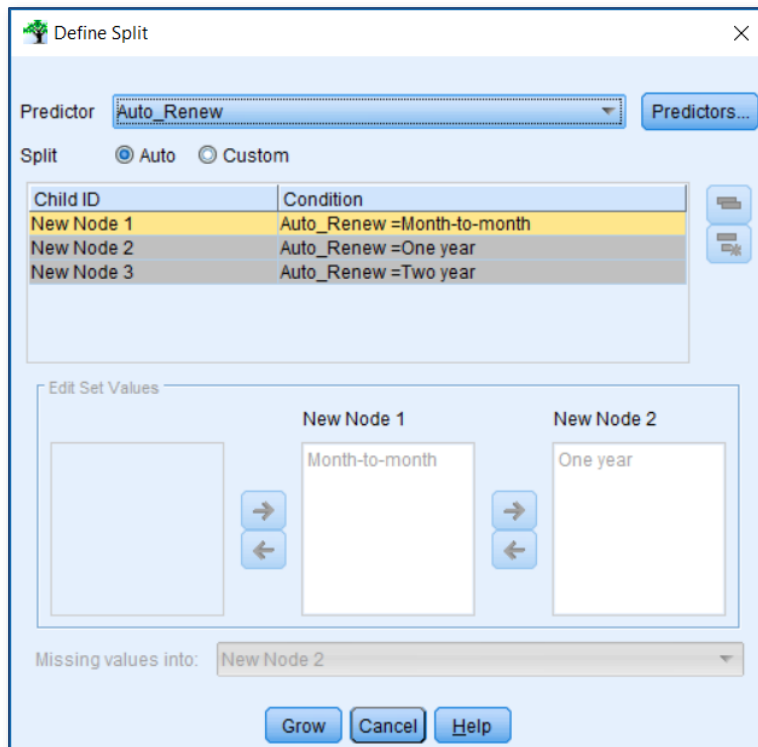


Figure 8.6 The Define Split Dialog

To view the probability values associated with each test of an independent variable against the target field, click:

Predictors...

Figure 8.7 shows the Select Predictor dialog.

Predictor	Nodes	Statistic	DF	Adj. Prob.
Auto_Renew	3	Chi-square=819.405	2	0.000
Tenure	7	Chi-square=686.801	6	0.000
Internet_Protection	3	Chi-square=577.610	2	0.000
Premium_Support	3	Chi-square=562.719	2	0.000
Broadband	3	Chi-square=505.293	2	0.000
Payment_Type_3Grp	2	Chi-square=443.770	1	0.000
Payment_Type	3	Chi-square=447.868	2	0.000
Backup	3	Chi-square=401.367	2	0.000
Insurance	3	Chi-square=373.610	2	0.000
Monthly_Rate	4	Chi-square=298.270	3	0.000
Total_Bill	7	Chi-square=306.566	6	0.000
Cashback	3	Chi-square=275.453	2	0.000
Very_Loyal	2	Chi-square=260.469	1	0.000
Super_Sports	3	Chi-square=253.077	2	0.000
Box_Office_Movies	3	Chi-square=252.582	2	0.000
Average_Monthly_Bill	5	Chi-square=272.665	4	0.000
Family_Plus	2	Chi-square=193.341	1	0.000
Dependents	2	Chi-square=145.072	1	0.000
Customer_Tier	3	Chi-square=131.587	2	0.000
Partner	2	Chi-square=115.476	1	0.000
Pensioner	2	Chi-square=114.548	1	0.000
Multiline	2	Chi-square=11.271	1	0.002
Landline	2	Chi-square=2.260	1	0.133
gender	2	Chi-square=0.373	1	0.541

Figure 8.7 The Select Predictor Dialog

We can see that according to the Select Predictor dialog, the variable with the largest Chi-square value (and more importantly the smallest probability) is 'Auto_Renew'. This is the field that using the default settings, the model would select as the first predictor variable to be entered. To see this, click:

Cancel

Grow

Figure 8.8 shows the tree model grown one level with the Auto-Renew field used to define the first split.

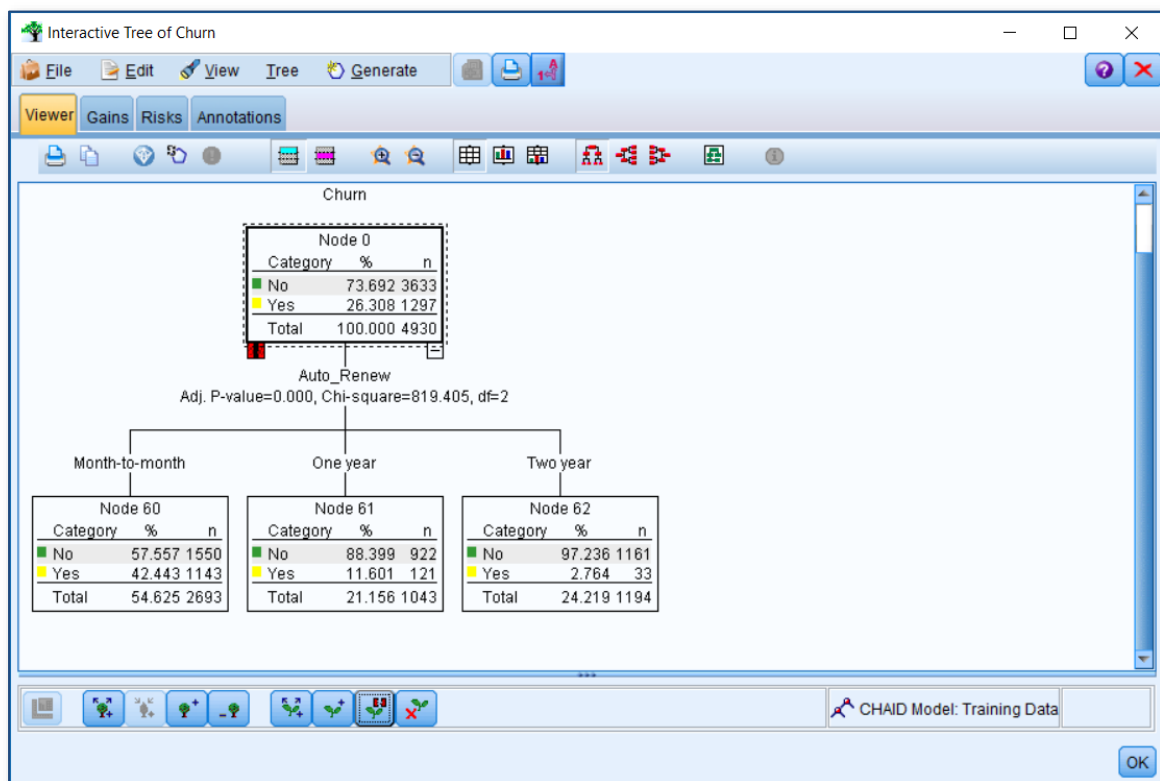


Figure 8.8 The CHAID tree model grown with one predictor field

Using the auto renew field, the original sample of 4,930 records has been split into three groups. We can see that those customers on a month-to-month renewal plan have the highest churn rate (42.4%), followed by those on a one-year plan (11.6%) whereas those on a two-year plan have the lowest churn rate (2.7%). This tells us that how customers choose to renew their contract is the single biggest predictor of churn according to the CHAID algorithm. If we grow the tree a further level deep, we may well find that more than one field is chosen to define the next three branches. After all, at this point the model has effectively created three separate populations of customer and we shouldn't expect that the same field will be selected as the next best predictor of churn at each of these three points. To grow the tree another level, click:



Figure 8.9 shows the tree model grown another level.

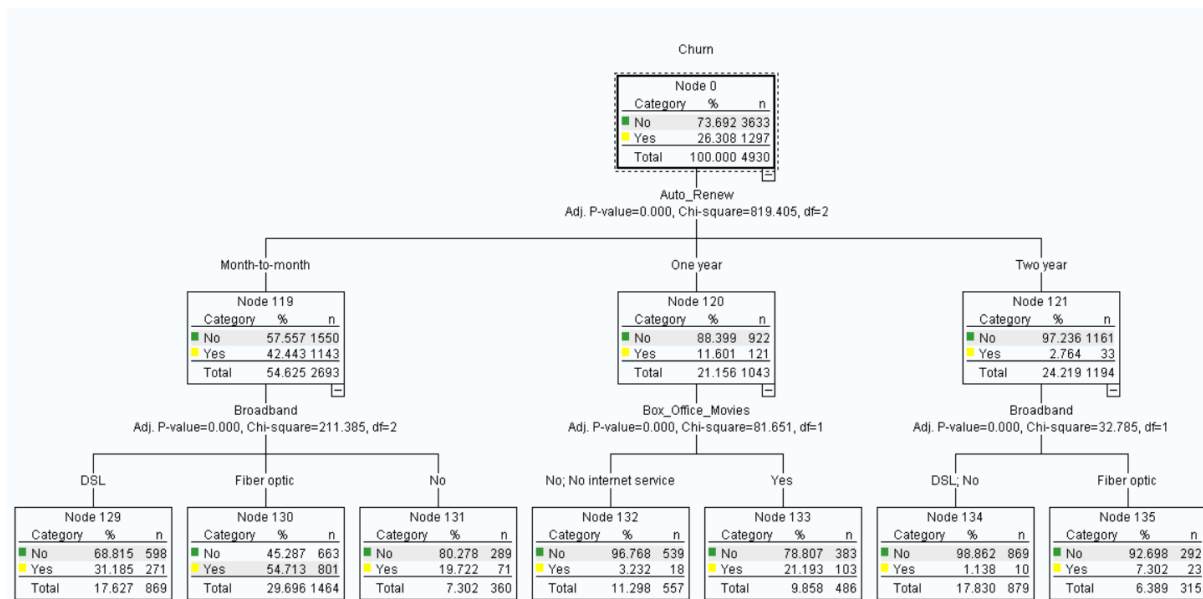


Figure 8.9 The CHAID tree model grown to two levels deep

By growing the tree another level, we can see that two more variables are chosen. Firstly, the variable that records the type of broadband service the customers receive has been chosen to split the records in the “Month-to-month” branch. We can see that the churn rate within this branch varies from 54.7% for fibre optic users to 19.7% for those with no broadband package. Interestingly, the same variable is chosen to split those customers on a two-year contract, except that here, those with no broadband package have been merged with those using a DSL package to form a single group. This has happened because the algorithm has found that the churn rates for these two groups are not statistically different from one another. For both merged groups the churn rate has dropped to around 1%. Finally, CHAID has chosen a completely different variable to split those customers on a one-year contract. Again, it has merged two groups: this time within the variable that records whether or not the customer has opted to include ‘Box Office Movies’ in their package. For those that take this option the churn rate jumps to 21%, otherwise the rate drops to 3%. It is precisely this ability of decision tree algorithms like CHAID to find subtle patterns in the way variables interact with each other against a target field, that makes it such a powerful tool for driving insight whilst predicting outcomes.

As mentioned earlier, the tree that we are viewing is based on a random 70% subset of the data. This is used mainly so that we can validate the model when tested against another sample (the proportion of the split value can be altered in the ‘Advanced’ settings within the model options). In fact, one of the most powerful aspects of the interactive mode within CHAID is the ability to allow us to switch back and forth between the ‘training’ sample (the ‘Tree growing set’) and the withheld ‘test’ group that represents the remaining 30% of the data. This test group is labelled the ‘Overfit Prevention Set’. ‘Overfitted’ models refer to predictive models which are too specific and sensitive to the subtleties of the sample data and

so perform badly when applied to other datasets (or in the real world). To view how the tree looks when applied to this sub-sample, on the toolbar click:



Figure 8.10 shows the view of the tree when applied to the overfit prevention set.

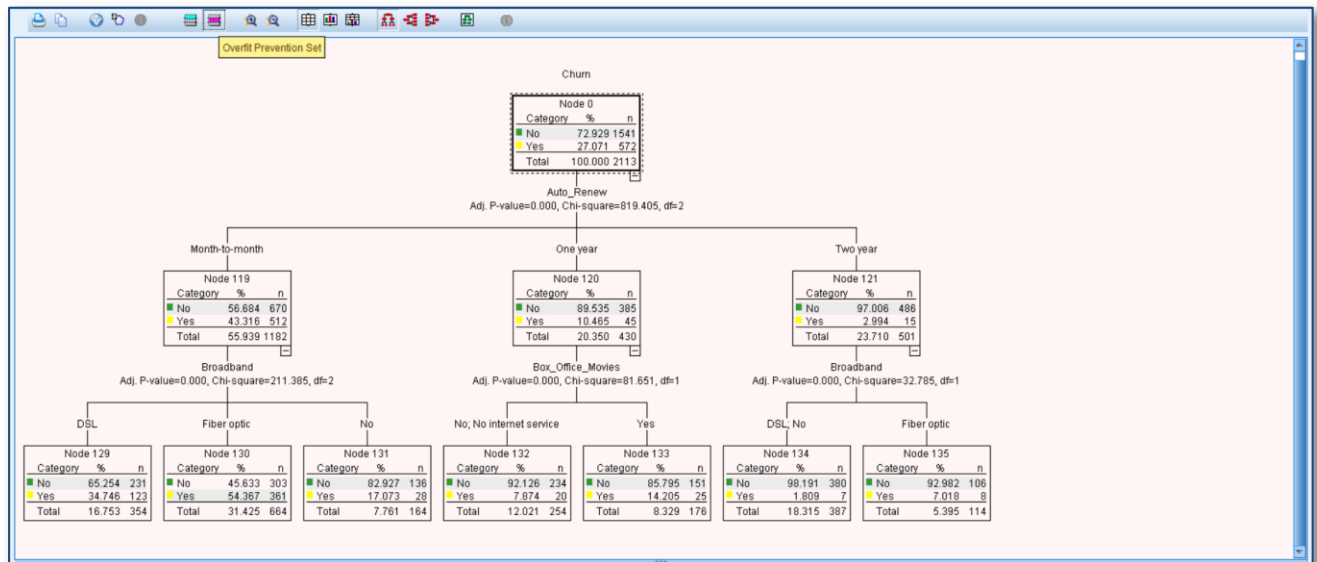


Figure 8.10 The current CHAID tree model applied to the 'overfit prevention set'

In most cases we can see that the numbers in the tree change only slightly (maybe by 1 or 2 percent). However, the proportion of churners in the "Box office movies" branch has changed more significantly for those in the 'No' or 'No Internet Service' group. Here the value has changed from 3% in the tree growing set to 7% in the overfit prevention set. Meanwhile for those who do have the "Box office movies" option, the value has changed from 21% in the tree growing set to 14% in the overfit prevention set. This simply reminds us that predictive models have different margins of error and that when applied to other data samples, we should expect to see some variation in the generated outcomes. There are many different approaches that analysts can employ to test model performance just as there are many strategies they may pursue to enhance the accuracy and stability of predictive models. At this stage however, we can return to building the rest of the decision tree to view the final model. To do so, click the toolbar button to switch back to view the tree growing set:



To grow the rest of the tree, click:



When the tree has completed growing, we may find it useful to click the tree map button to gain a better overall view of the final model: On the toolbar, click:



Figure 8.11 shows the fully grown CHAID decision tree and tree map.

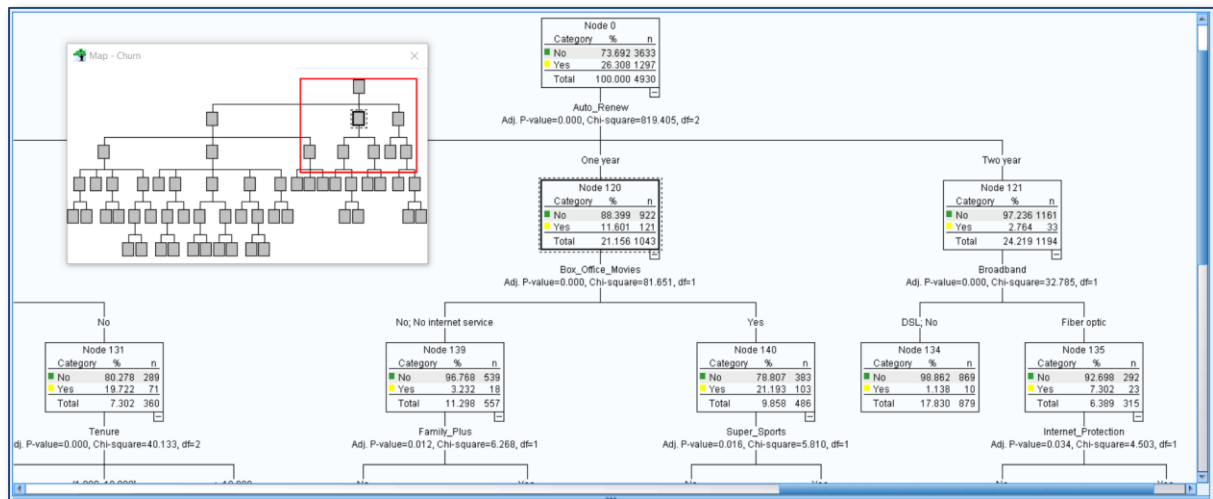


Figure 8.11 The fully grown CHAID tree with a tree map window to provide orientation

Now we can see that the entire model constitutes a much more complex set of variables and interactions. The tree itself has grown to a maximum of 5 levels deep. In fact, the algorithm continues to grow the tree from the initial root node until it encounters a 'stopping rule'. Stopping rules include the maximum allowable tree depth, which by default is set at 5; the minimum percentage (or frequency) of records in a 'parent node' (i.e. any node that generates another level) which is set at 2%; and the minimum percentage (or frequency) of records in a 'child node' (any node that doesn't generate another level) which is set at 1%. When any of these conditions cannot be met the tree stops growing. Furthermore, the significance threshold (probability value) that allows the splitting or merging of variables in the tree can also be set (the default probability value is 0.05). All of these values can be edited within the CHAID node and they enable analysts to control how complex or 'conservative' the final model might be.

Having inspected the fully unfolded model, we can close the model output and return to the initial CHAID node. At this point we should note that if we wish to close the interactive tree output and return to it later we can do so, and it will remain available to us for the rest of the session. To do this, from within the Interactive Tree output window, click:

File

Close

The interactive tree will remain open in the background for us. It's worth noting however, that doing this with many complex interactive trees may lead to the depletion of system memory resources as several sessions will be concurrently maintained. If you wish to fully close the interactive tree and free up these resources, within the Interactive Tree window, click the 'close and delete output' button:



A message dialog appears simply telling us that any changes we have made within the interactive tree will be lost should we continue.

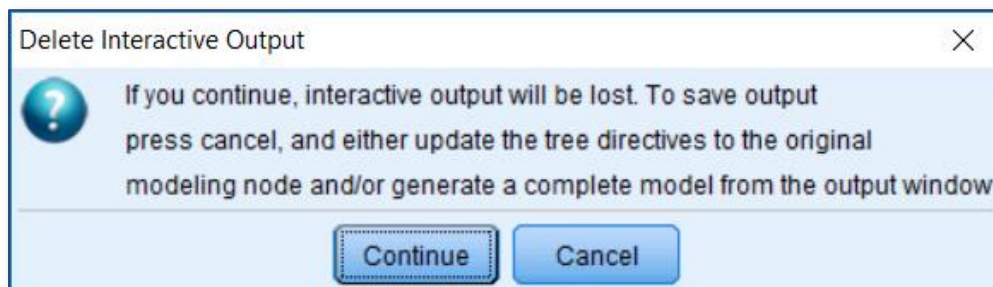


Figure 8.12 Delete Interactive Output message dialog

Click:

Continue

Now we can now see what happens when we build a CHAID tree using the default (non-interactive) method.

Right-click on the CHAID node and edit it

In the resulting dialog, within the Build Options tab (under the Objective item in the left-hand option list) click the radio button marked:

Generate model

Figure 8.13 shows this completed dialog.

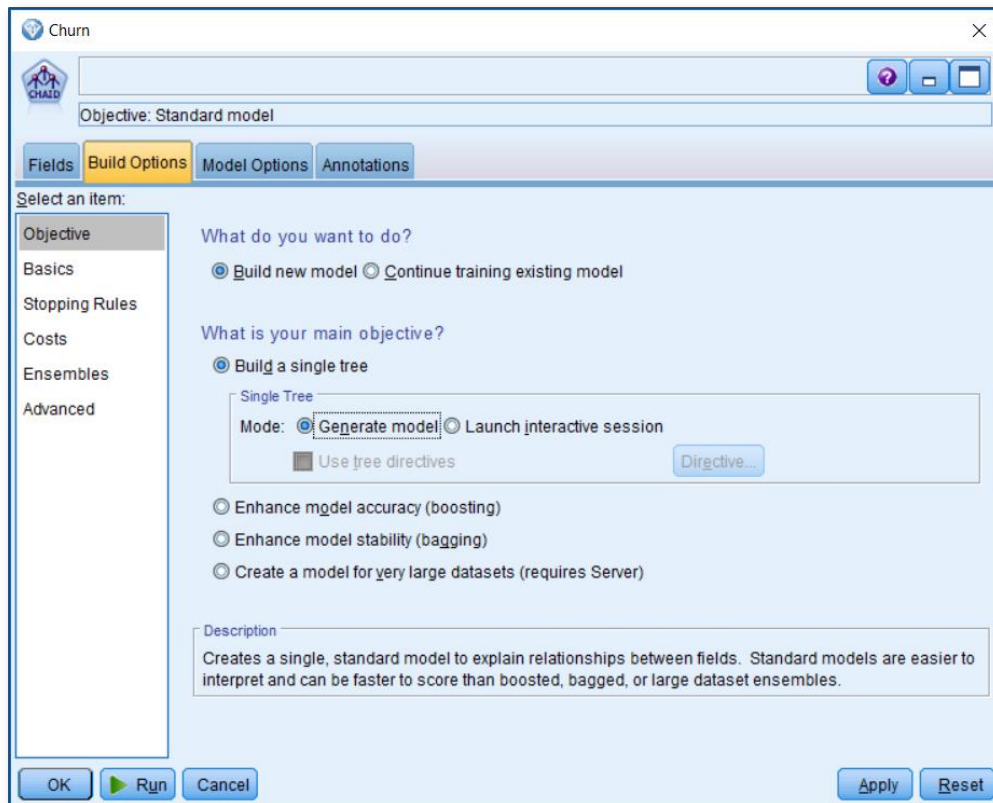


Figure 8.13 Switching from the 'Launch Interactive Session' mode to the default 'Generate Model' mode

To create the CHAID model automatically, click:

Run

The effect of doing this is that the CHAID node immediately creates a model nugget using the default settings of the algorithm. We should note that unlike the interactive mode, the default 'Generate Model' mode uses *all the data* in the sample. It does not create a model based on a random subset of the data. Figure 8.14 shows the Modeler stream with the new CHAID model nugget added.

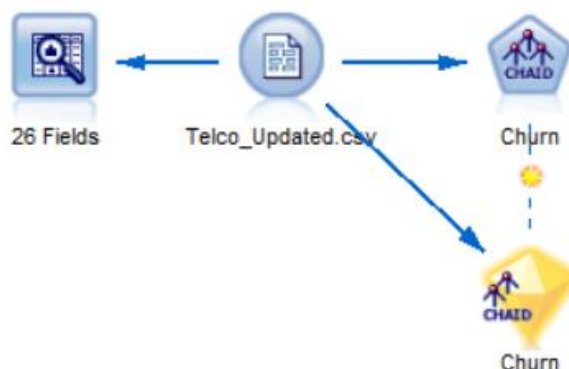


Figure 8.14 A newly generated CHAID model nugget

Model Nuggets



Model nuggets are generated by modelling algorithms. They encapsulate the actual models which in turn may generate outcomes based on statistical coefficients, decision rules or complex (and hidden) code. Some of these modelling nuggets can be saved as PMML objects. PMML, or 'Predictive Modelling Mark-up Language', is a model coding standard which acts as a common language and so allows models to be shared between third party applications. Furthermore, users with *server licences* for SPSS Modeler may find that by utilising the *SQL Pushback* facility within Modeler, model nuggets such as CHAID can be automatically converted to SQL to allow *in-database scoring* of data. Most model nuggets allow us to browse the contents of the generated model and view important details about its construction. To see this:

Right-click on the CHAID model nugget and edit it

Figure 8.15 shows the contents of the generated model nugget.

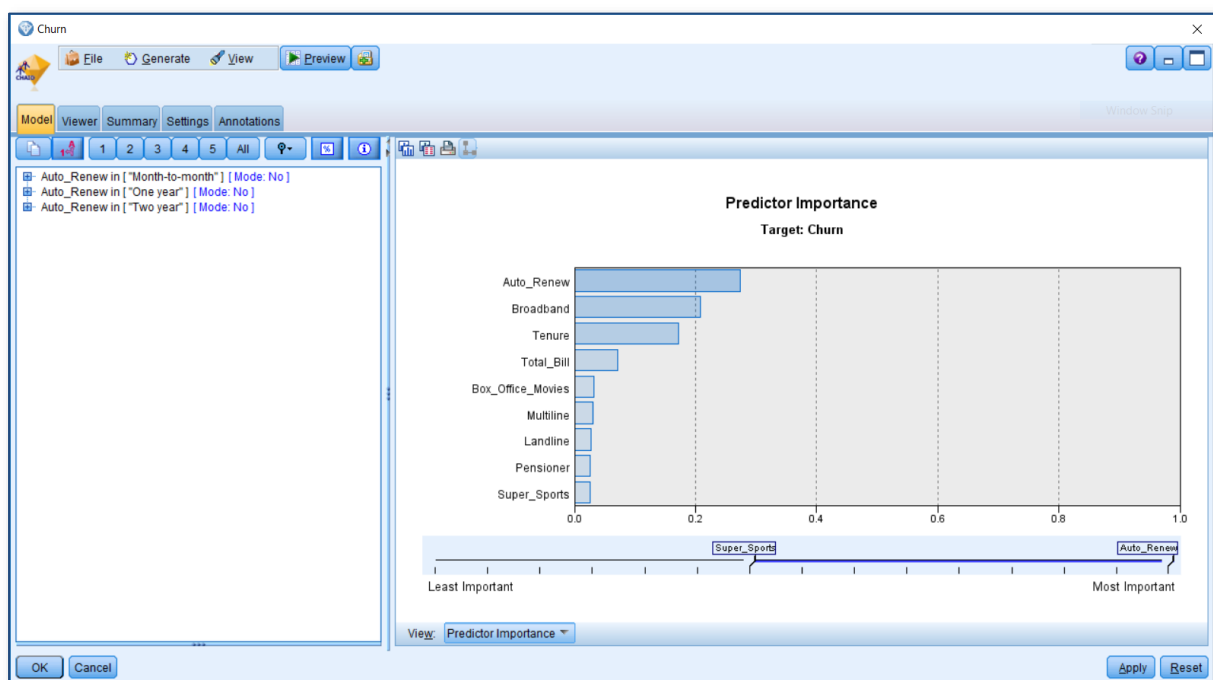


Figure 8.15 The Model Rules and Predictor Importance panes within the CHAID nugget

The first thing than we notice when we browse the model contents is the Predictor Importance chart. This is generated independently of the model itself and is really an analysis of the how important each field is in the final model. It may *not* correspond to the order in which the variables themselves are entered into a decision tree. Instead it attempts to measure how much the model error is reduced by including the field in question. Furthermore, the values that correspond to each variable are relative and standardised so that they all sum to 1.0. On the left hand-side of the model tab we can see the first level of the

decision tree expressed as a ruleset. To unfold each branch of the ruleset, click the plus sign button:



By the clicking “show/hide coincidence figures” button on the toolbar we can see the accuracy of each rule. Click:



Figure 8.16 shows the unfolded ruleset of the first level of the decision tree.

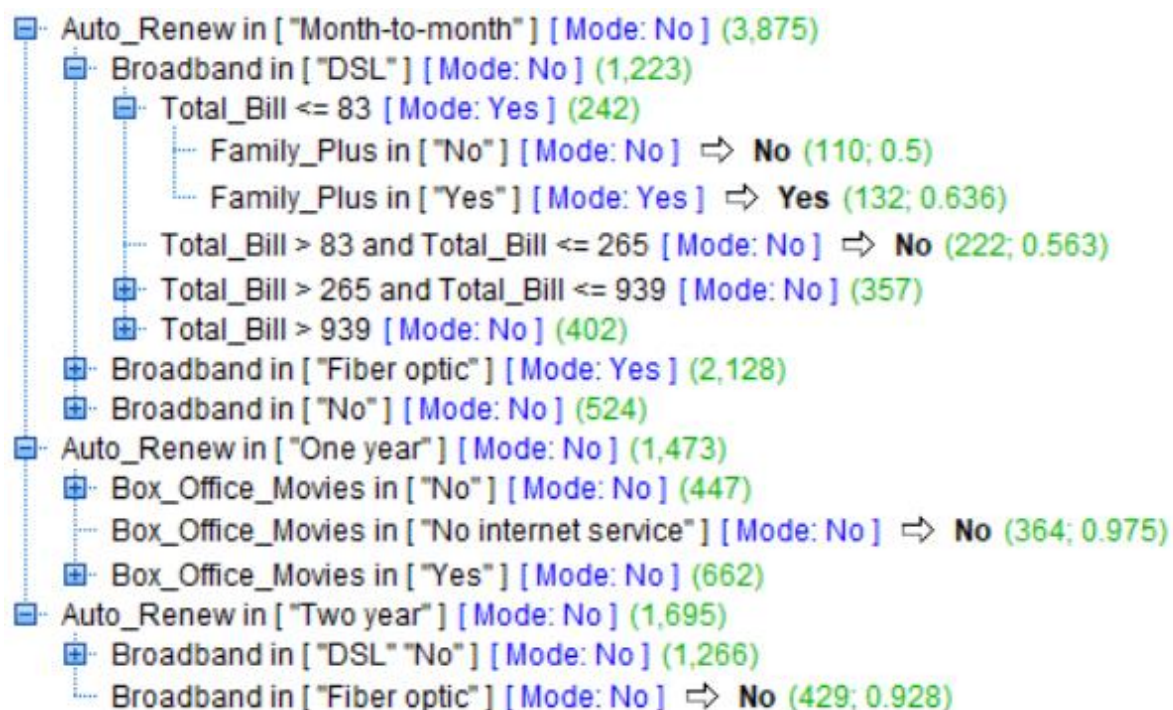


Figure 8.16 The model rules unfolded to show the ruleset of the first level of the CHAID decision tree model

Clicking the corresponding “Show level” buttons on the toolbar across the top of the ruleset means we can unfold the rules for the subsequent levels in the tree. Figure 8.17 shows the rules unfolded down to the third level of the decision tree.

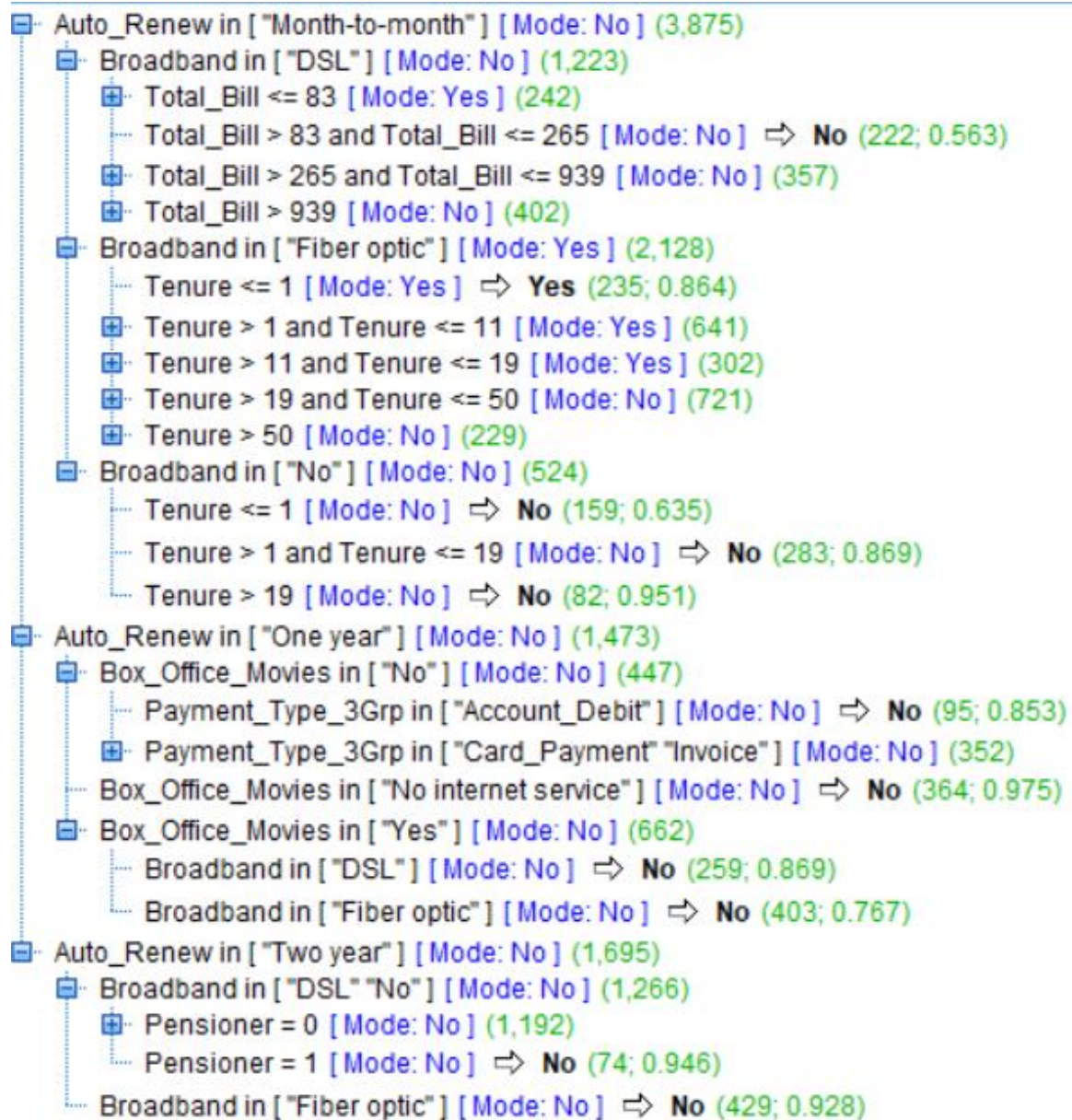


Figure 8.17 The model rules unfolded to show the ruleset of the third level of the CHAID decision tree model

Rulesets like this are a description of the decision tree simply expressed as logical rules. The key to making sense of them is to remember that they are *hierarchical* rules just as the tree is a hierarchy of interactions. Some modelling techniques actually abandon the hierarchical decision tree approach altogether and simply generate models that are based on several groups of rules that can overlap with one another.

By clicking the Viewer tab, we see the decision tree itself (figure 8.18). This is a similar view to the interactive viewer we saw earlier but note the absence of the toolbar buttons that switch between the 'Tree growing set' and the 'Overfit Prevention Set'. This is of course because in non-interactive mode, a single tree is generated using all the available data.

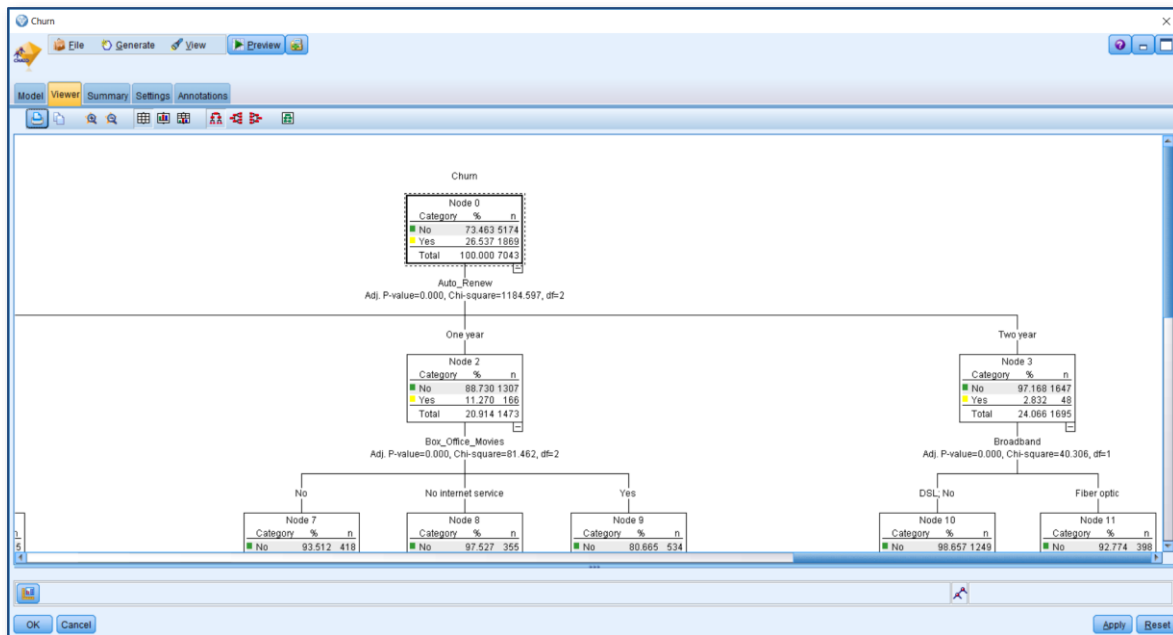


Figure 8.18 The Viewer Tab showing a graphical representation of the CHAID Tree

The Summary tab (figure 8.19) provides us with detailed information regarding the variable inputs used and the chosen algorithm options to generate the model itself.

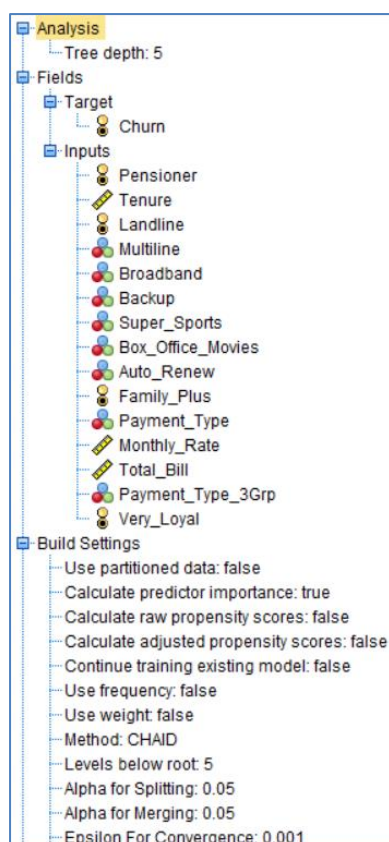


Figure 8.19 Information in the Summary tab showing the inputs and options selected to build the model

The Settings tab shows the options related to the data that *the model generates*. By default, predictive models for categorical targets within SPSS Modeler will create new fields indicating which of the target categories each record belongs to. Furthermore, they produce a score running from 0 to 1 indicating the level of *confidence* associated with the prediction. Thus, the model may predict an individual person is likely to churn with a confidence of 0.77 (77%) or alternatively, predict that a person is likely to remain a current customer with a confidence of 0.81 (81%). In traditional statistical models, this score is normally shown as a single probability of the case falling into one of the two outcomes. Normally the category with the ‘higher’ value is chosen as the target reference for the probability score. SPSS Modeler does this as well but refers to the probability score as a ‘Propensity score’. If we assume that the ‘Yes’ group in the Churn variable is the target reference category, then using the same example as previous, the propensity score would still generate a value of 0.77 for the person it predicted to churn but a score of 0.19 for the person it predicted to remain a customer (because $1 - 0.81 = 0.19$). This simply indicates that there is a 19% chance that they would churn. To request propensity scores to be generated alongside confidence scores, check the box marked:

Calculate raw propensity scores

Figure 8.20 shows the completed Setting tab within the dialog.

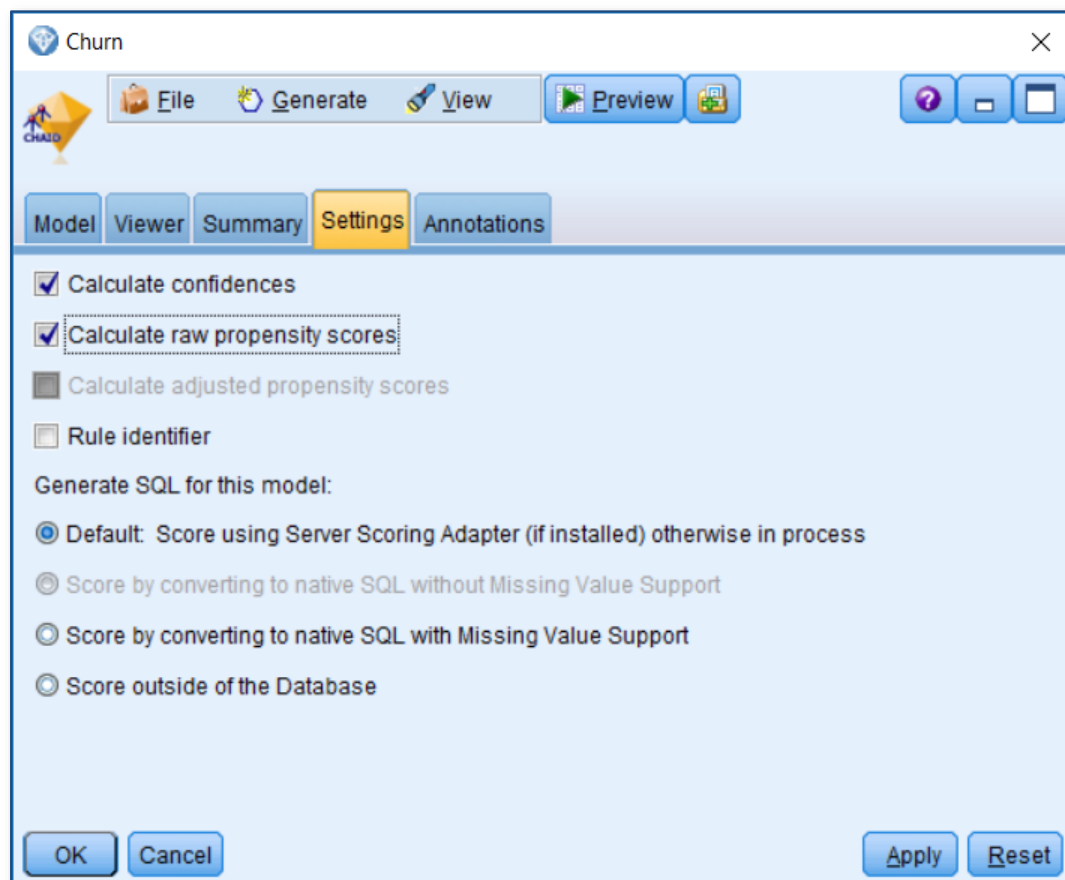


Figure 8.20 The Settings Tab showing the scoring options for the CHAID model

To view a preview of the scores themselves, click:

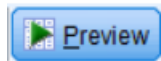


Figure 8.21 shows the predictions that the model nugget generates in the form of additional fields added to the end of the dataset.

	Type_3Grp	Average_Monthly_Bill	Very_Loyal	Cashback	Customer_Tier	\$R-Churn	\$RC-Churn	\$RRP-Churn
1	Current	\$null\$ F		0.000	d. Refunded	No	0.933	0.067
2		\$null\$ F		0.000	d. Refunded	No	0.992	0.008
3		\$null\$ F		0.000	d. Refunded	No	1.000	0.000
4		\$null\$ F		0.000	d. Refunded	No	0.992	0.008
5	Current	\$null\$ F		0.000	d. Refunded	No	0.933	0.067
6		\$null\$ F		0.000	d. Refunded	No	0.992	0.008
7		\$null\$ F		0.000	d. Refunded	No	0.992	0.008
8		\$null\$ F		0.000	d. Refunded	No	0.992	0.008
9		\$null\$ F		0.000	d. Refunded	No	0.975	0.025
10		\$null\$ F		0.000	d. Refunded	No	1.000	0.000

Figure 8.21 Model predictions added to the file

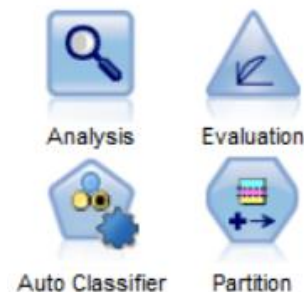
The output table shows that the model prefixes each of the new fields with the characters '\$R'. The '\$' symbol simply denotes that this is a field created by a model, whereas the 'R' character indicates that it is a CHAID model (the character 'C' had already been assigned to a modelling technique when the CHAID algorithm was added to Modeler). The field \$R-Churn represents the actual predictions themselves. Here the first 10 rows are all predicted to be in the 'No' group (i.e. to remain current customers). The field \$RC-Churn shows how *confident* the model is of the prediction (based upon where the record would appear in a CHAID tree). The field \$RRP-Churn represents the *propensity* score. Here the target reference is the 'Yes' group (i.e. the customers who churn). Because the confidence values for are so high, and all of the first ten records are predicted to remain customers, the corresponding propensity scores are all very low.

By comparing these predictions to the actual outcomes, we are therefore able to measure the accuracy of the predictive model. As well shall see in the next section, SPSS Modeler contains two dedicated nodes that directly analyse the new fields generated by model nuggets in order to assess the model performance.

Section 9:

Assessing and Selecting Predictive Models

- Assessing Model Performance
- The Analysis Node
- Success Criteria Scenarios
- The Evaluation Node
- The Partition Node
- The Auto Classifier Node



As we have already seen, the CRISP-DM process explicitly contains a step for assessing the results of the modelling stage: Evaluation. It's important to understand that what might be regarded as an excellent model in one set of circumstances might also be regarded as completely inadequate in another. Consider the issue of false positives and false negatives. Models always generate some errors for any outcome that we try to predict or estimate. In fact, in most cases models that claim close to 100% accuracy are usually badly flawed. Nevertheless, the goal of the modelling process is of course to try to minimise this error. But even when trying to predict a two-category outcome, we can't always assume the level of accuracy in both outcomes is the same. In other words, just because the model is 85% accurate *overall*, doesn't mean that it is 85% accurate in predicting *both* outcomes. It may well predict every record to have the *same value* (e.g. no response) and if 85% of the data happens to have that outcome, then the overall accuracy is 85%. Moreover, with certain predicted outcomes, we may be more tolerant of error. If a model is trained to detect a fatal illness it is more likely that the analyst will aim for one that has as few *false negatives* as possible, because in such circumstances, a false negative would mean that the model *fails* to detect the disease when the patient *actually has it*. With such serious consequences we tend to err on the side of caution, so this situation could be regarded as more dangerous than predicting that the patient probably has the disease when in fact they don't (a false positive).

Unfortunately, it is not an uncommon situation for analysts to spend considerable time preparing and transforming data before finally building predictive models only to find that they don't know how useful or valuable the results are. For this reason, the Evaluation phase of CRISP-DM compels us to assess the resultant models *in terms of the success criteria defined during the Business Understanding stage*.

Assessing Model Performance

By following CRISP-DM, we should have documented the model success criteria early in the process. In this section, we can explore some example success criteria that might be applied to the sample datasets we have been working with thus far. For now though, let's consider

what additional pragmatic factors are associated choosing one predictive model over another.

Accuracy

It goes without saying that predictive models have to be sufficiently accurate at predicting an outcome to be regarded as useful. But what we mean by 'sufficiently accurate' depends on the context that the model is to be used in. If we wish to predict an outcome that only occurs 1% of the time, then technically speaking, any model with an accuracy of greater than 1% at predicting that particular outcome is an improvement. For this reason, it is essential that analysts establish a *baseline* against which to judge the model. As we have seen already, there are *costs* associated with *false positives* and *false negatives* and we must be aware of these when assessing the model accuracy. Moreover, within statistics and predictive analytics, there are several specific metrics that can be used to measure accuracy and 'model fit' such as overall accuracy, lift values, area under the curve, R-square etc. So care must be taken to choose a criterion that helps us select a model that is both accurate and useful.

Interpretability

There are many different techniques and algorithms that can be employed to generate a predictive model. Some approaches (especially machine learning algorithms) yield '*black box*' models. These are models that cannot be directly interpreted in the same way that certain statistical or rule-based models can. They are often comprised of large numbers of hidden rules, variable weights or data transformations that the analyst cannot inspect or make sense of. In some fields model accuracy is deemed to be more important than interpretability, so these algorithms are regularly employed. In other disciplines however (such as credit scoring, epidemiology or social research), being able to understand *how* the model generates predictions is of paramount importance. In an ideal world, most analysts would prefer to generate models that are highly accurate *and* easily interpretable.

Stability

Analytical models are based on samples of data collected under certain circumstances and within specific time-frames. We should not be surprised to find that a model fails to generate accurate results when applied to a range of values or circumstances that are very different from the ones it was developed under. For example, model accuracy may decay over time as changes in fashion, demographics, competitor behaviour or market offerings begin to look very different from the time period that the model was developed in. Also, if the model is based on an unrepresentative sample, we can find that it generates inaccurate predictions or even wild estimations when encountering an unfamiliar case. Even certain relatively novel combinations of demographic factors such as ethnicity, gender, age and region can mean that the model is unable to accurately predict the outcome of interest. Stable models however,

are able to generate reliable predictions and estimates with a wide range of data combinations over a useful period of time before they need to be updated or 'refreshed' with new data.

Coherence

As mentioned earlier, many analysts only work with models that can be directly *interpreted*. One of the reasons for doing this to make sure that the model *makes sense*. It is not unusual to discover that a model utilises counter-intuitive rules or non-sensical relationships to estimate a value. Examples include price rises increasing the likelihood to purchase, missing values for variables such as age generating higher estimates of revenue or low satisfaction scores reducing likelihood to churn. In many situations, there may be a sensible explanation for these contradictory relationships. However, more often than not, what is really driving the relationship is a *hidden* variable that explains what's going on. Perhaps price rises increase the likelihood to purchase because they are related to higher demand in the market. Therefore, demand is the driving factor and price is merely a function of it. Missing data for variables such as age may indicate that the person registered for a product or service through a different channel (e.g. *in person* as opposed to via the website) and that in reality the channel is the key predictor. Even lower satisfaction scores could simply indicate that a person *cares* more about a service or has previously complained and subsequently received a discount so lowering their likelihood to defect. Coherent models are valued not only because of the obvious insights they deliver, but because they provide reassurance that the model is not based on a combination of spurious relationships.

Simplicity

Most predictive models are *multivariate* in nature. They are developed from a combination of interrelationships between variables or *model terms*. Many of them can be likened to a house of cards where each layer is precariously added to previous tier. Complexity therefore not only leads to issues with interpretability but also stability. For these reasons alone, simplicity is a key criterion when selecting a predictive model. For example, an analyst might well reject a model with an overall accuracy of 85% but based on 18 variables in favour of one with an accuracy of 82% but only based on 8 variables. This is because including the terms from an additional 10 variables to gain a mere 3% of accuracy may be regarded as poor trade-off.

Performance

A final consideration is the computational performance of the model. Various modelling algorithms utilise resources very differently from one another. Some might not work well with categorical data and require data transformations to perform effectively. Some algorithms

might take a very long time to build a final model or require significant memory allocation and processing power. These requirements can mean that it takes much longer to refine and uncover a final satisfactory model. In a similar vein, when the model is deployed to generate predictions, it may require unacceptable resources in terms of computing power and time to 'score' new cases. In fact, there have been documented examples of predictive models winning predictive analytics competitions by achieving the best score based on a specific accuracy measure only to be deemed unusable in the real world due to their computational resource requirements.

The Analysis Node



The Analysis node is specifically designed to evaluate the performance of predictive models. To see how it works, from the Section 9 folder open the following stream.

Section 9 start.str

Figure 9.1 shows the stream.

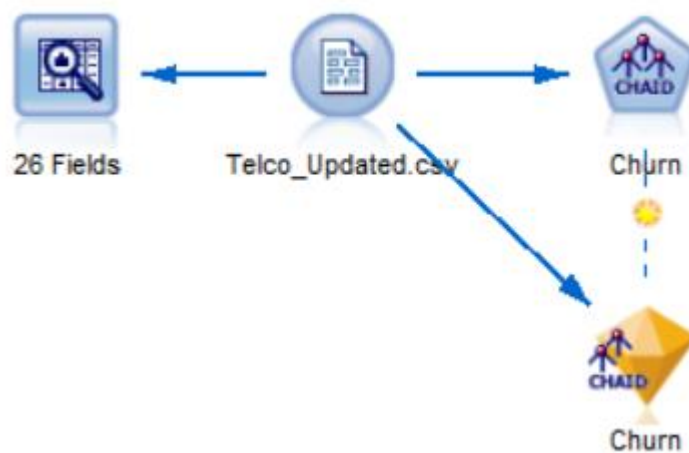


Figure 9.1 The SPSS Modeler stream 'Section 9 Start.str'

The stream contains a basic CHAID model generated in the previous section. To view the overall accuracy of the model, highlight the model nugget and from the Output tab:

Double click the Analysis node

Figure 9.2 shows the Analysis node attached to the CHAID nugget.

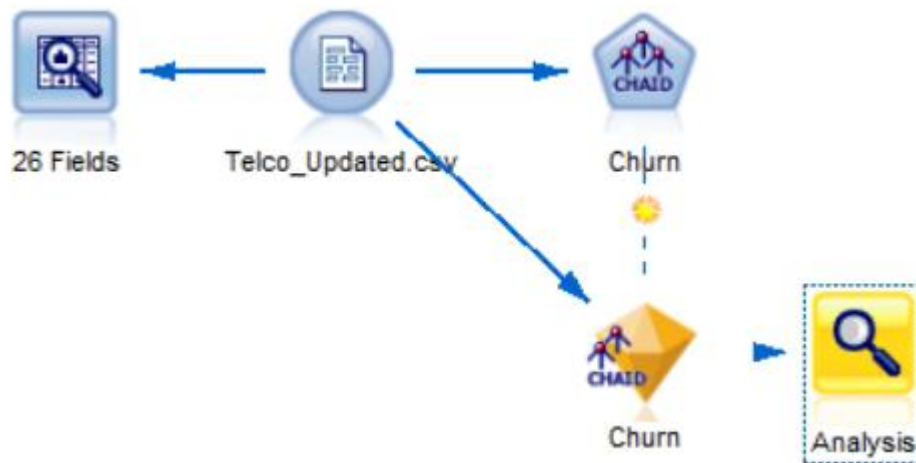


Figure 9.2 The Analysis node attached to the CHAID model nugget

Right-click the Analysis node and select Run

Figure 9.3 shows the Analysis node output.

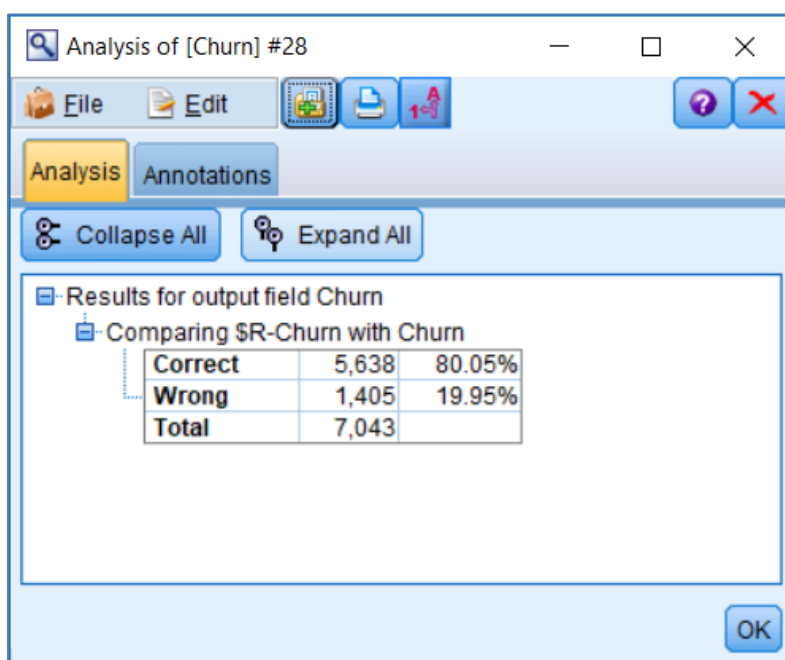


Figure 9.3 The default Analysis node output showing the CHAID model performance

As we can see from the default Analysis node output, the CHAID model has correctly predicted the outcome in 80% of the cases. The node itself simply calculates how often the two values in the target field (Churn) coincide with the predicted outcomes in the \$R-Churn field that the model nugget generates. However, we must remember that 73% of the data is comprised of customers who have not churned. This represents our baseline against which to compare the model performance. In which case, if the model itself simply predicted every case to be a current customer, the analysis node would indicate that the model was accurate

73% of the time. To see how well the model does at predicting the individual outcomes ('Yes'/'No') for whether or not the customer has churned, we must re-run the Analysis node and request additional output. To do so:

Click 'OK' to close the existing Analysis node output

Right-click on the attached Analysis node and edit it

As figure 9.4 shows, there are multiple options available for evaluating the model performance. In this case however, we need only request that the node includes a simple crosstab to compare the classification accuracy of the predicted values against the actual outcomes. To do so, check the box marked:

Coincidence matrices (for symbolic targets)

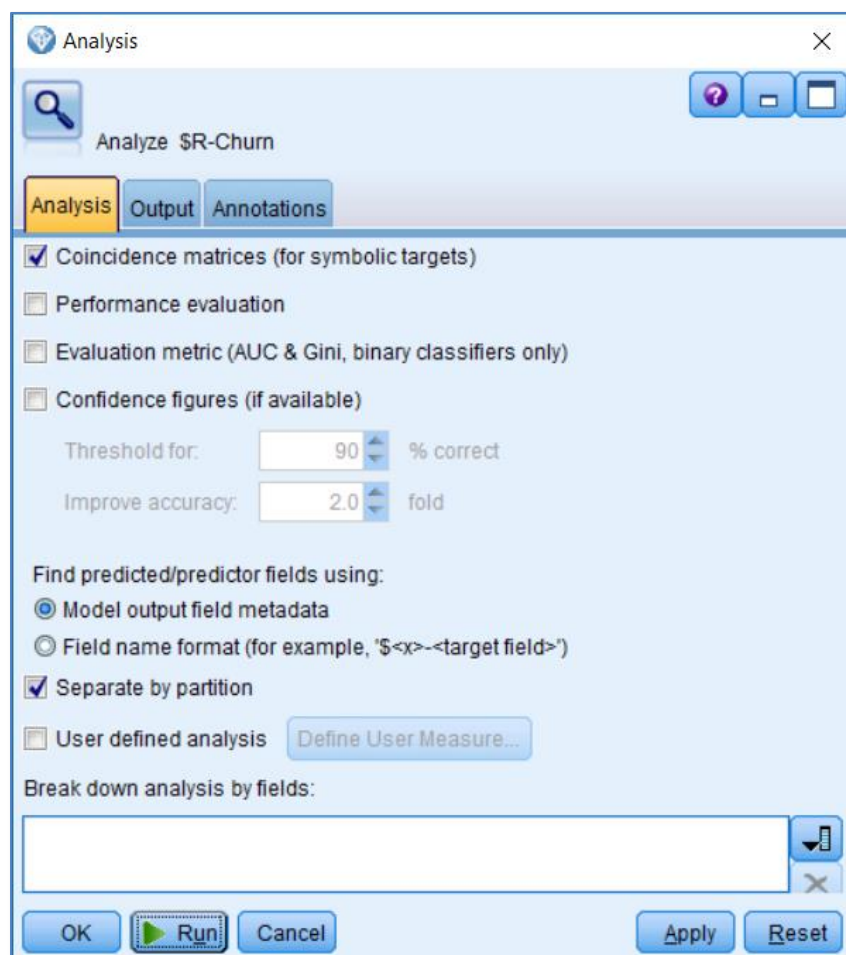


Figure 9.4 Requesting a coincidence matrix (crosstab) to see how well the model predicts the individual outcomes

Now click:

Run

Figure 9.5 shows the updated Analysis node output.

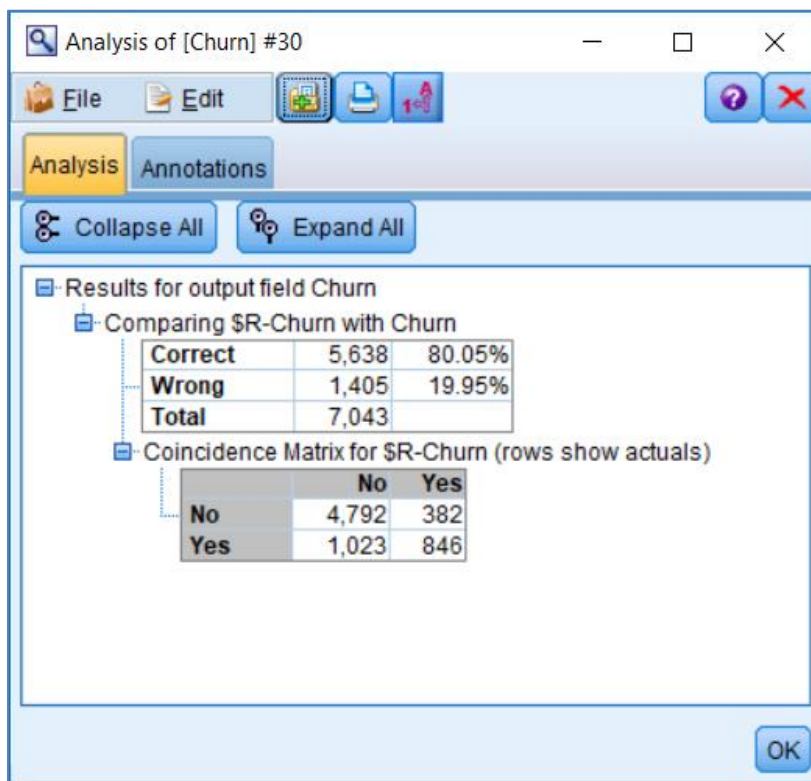


Figure 9.5 The Analysis node output showing a coincidence matrix (crosstab) of the predicted outcomes against the actuals

This time the output from the Analysis node tells us how many of the cases that the model expected to churn had in fact done so (as well as how many of those it predicted to be current customers remained with the telco). The output itself directs us to view the categories in the rows as the *actual* values with the *predicted* outcomes shown in the column dimension. Therefore, we can see that 4,792 people were correctly predicted to remain current customers. However, the model erroneously predicted 382 of the current customers to have churned when in fact they had not (the false positives). Worse still, it predicted 1,023 of the customers who had in fact churned to remain current customers (the false negatives). On the other hand, it correctly identified 846 of the customers who churned. How are the cases being assigned to these predicted groups? Remember that the model generates a probability (or confidence) value for each case. Whichever outcome group ('Yes' or 'No') has the highest probability value, is assigned as the predicted outcome. So even though the baseline probability for a customer churning is around 27%, even if the model finds that a person has a 49% chance of being a churner, they are *still* predicted to be a current customer because the probability that they aren't a churner is slightly higher (51%). At this stage, we need to decide what is more important, accurately predicting current customers at the expense of accurately predicting churners, accurately predicting churners at the expense of accurately predicting current customers or are both groups equally important?

To illustrate this further, from the Section 9 folder open the following stream.

Section 9 Choose best model.str

Figure 9.6 shows the stream.

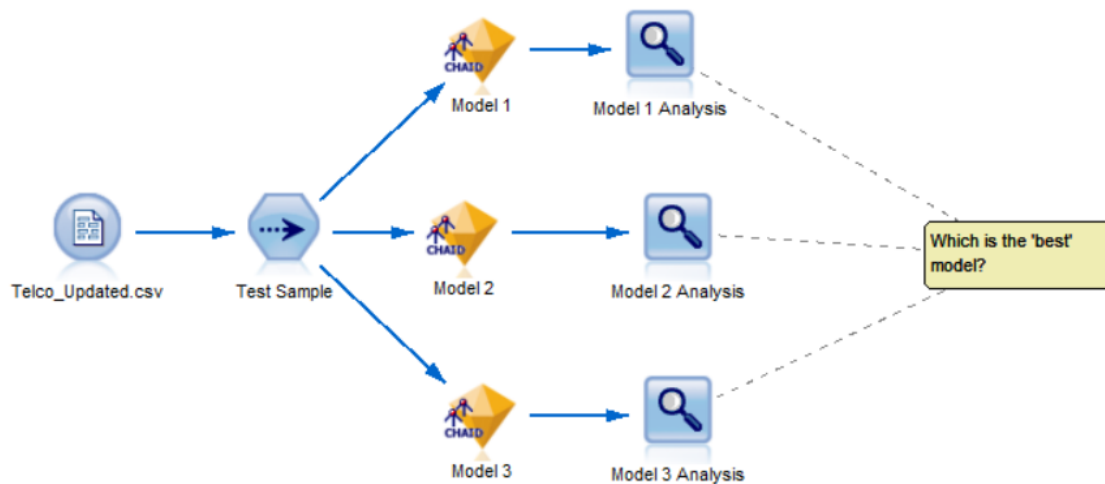


Figure 9.6 The Modeler stream 'Section 9 Choose best model.str'

The stream shows three alternative CHAID models built on the same random sub-sample of data but using different parameter settings and applied back to the same test sample. Figures 9.7a to 9.7c show the output from the three Analysis nodes attached to each respective model.

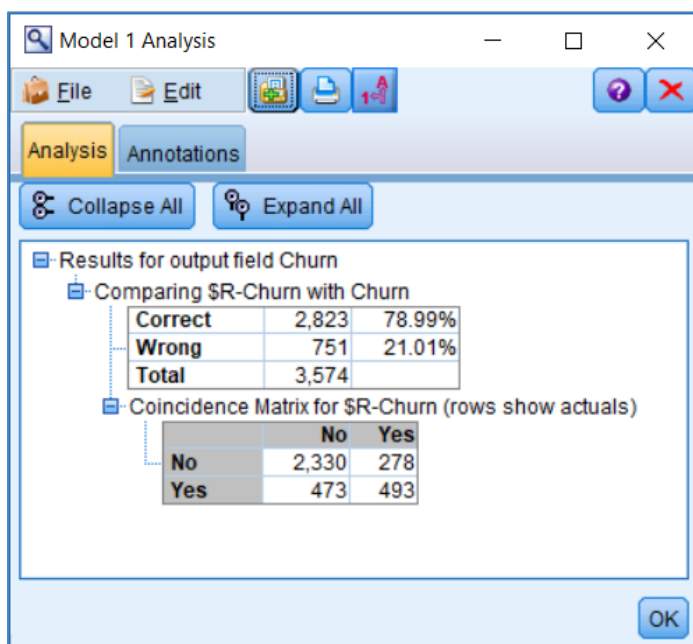


Figure 9.7a Analysis node output with coincidence matrix for Model 1

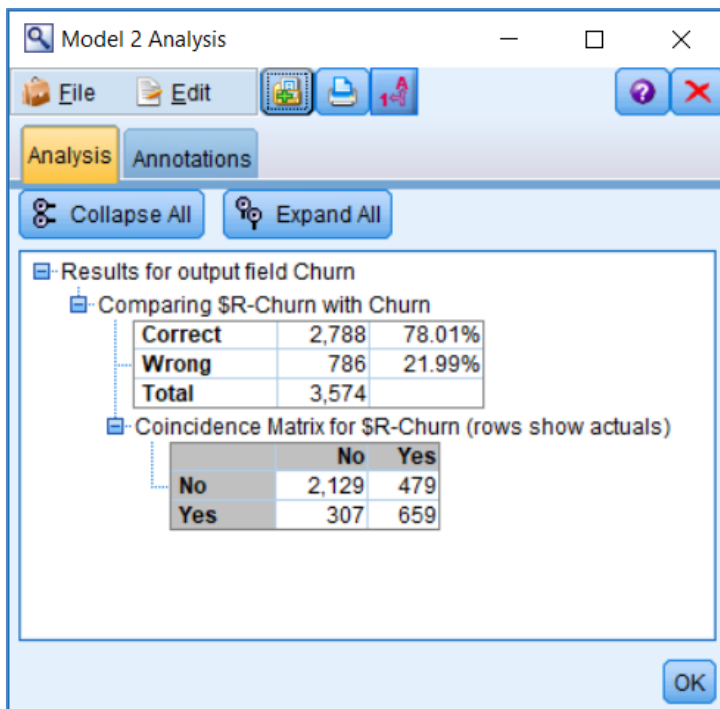


Figure 9.7b Analysis node output with coincidence matrix for Model 2

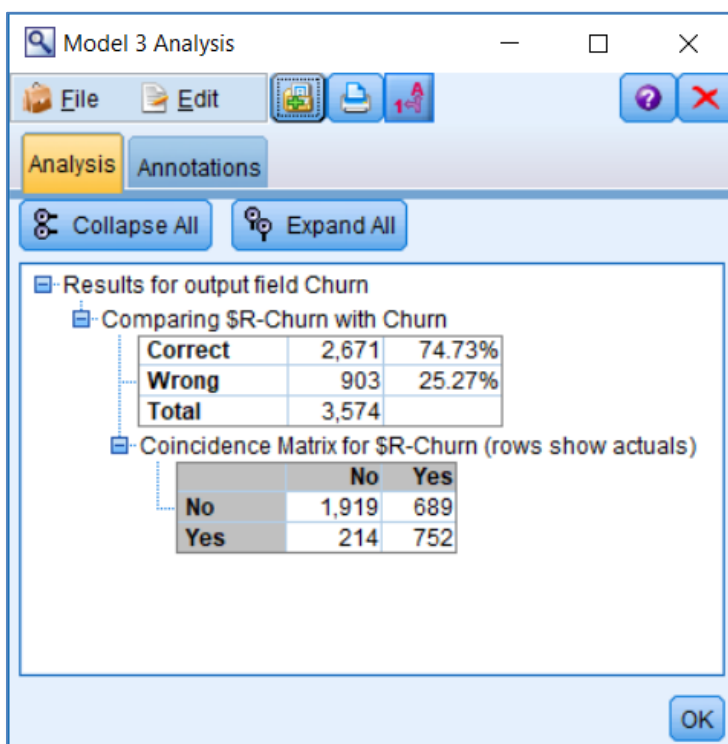


Figure 9.7c Analysis node output with coincidence matrix for Model 3

Although the overall model accuracy varies slightly with each Analysis node output. Looking at each of model's performance evaluations, we can see that a model with increased accuracy in correctly identifying the customers who have churned, tends to have a *decreased* accuracy in correctly identifying the current customers and vice-versa. The question as to which is the

best model relates back to our Business Understanding and the costs associated with each outcome. What are the costs associated with erroneously predicting someone to have a high risk of defecting compared to the cost of mistaking someone who in reality is likely to cancel their contract, for one who will remain a customer? Before we consider some example success criteria, let's take a look at how we can use the Analysis node to compare the performance of multiple models. From the Section 9 folder, open the following stream:

Section 9 Two Model Comparison.str

Figure 9.8 shows the stream.

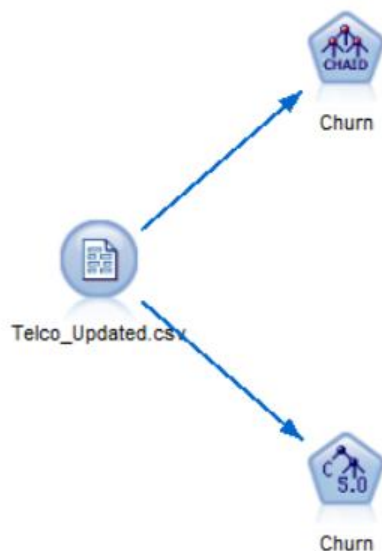


Figure 9.8 The Modeler stream 'Section 9 Two Model Comparison.str'

The stream shows two decision tree modelling nodes that have been configured to predict the same target field. To generate both models, from the main toolbar click the stream run button:



Both model nuggets are generated. To compare their respective performance with the same Analysis node, connect the two models in the same stream branch as shown in figure 9.9.

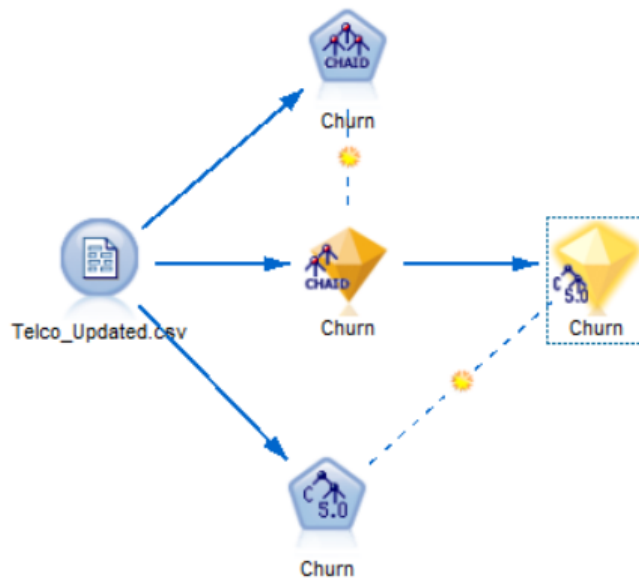


Figure 9.9 Connecting two model nuggets in the same stream branch

Having connected the two model nuggets, we can attach the analysis node to compare their performance. From the Output palette,

Attach an Analysis node to the last model nugget

Right-click on the analysis node and request ‘Coincidence matrices (for symbolic targets)’

Figure 9.10 shows the completed stream.

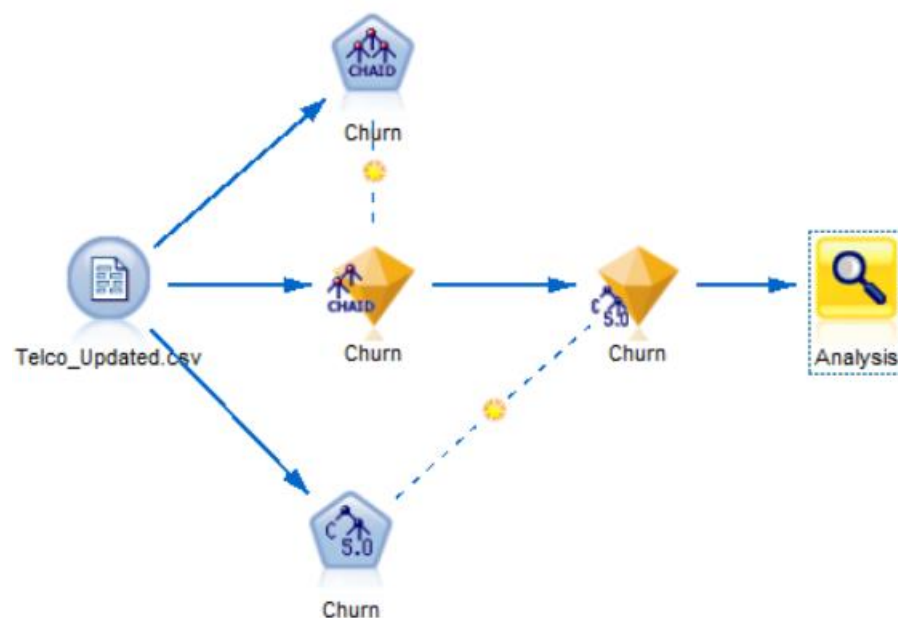


Figure 9.10 Analysis node attached to stream branch containing two model nuggets

Now:

Right-click and on the Analysis node and select run

Figure 9.11 shows the Analysis node output.

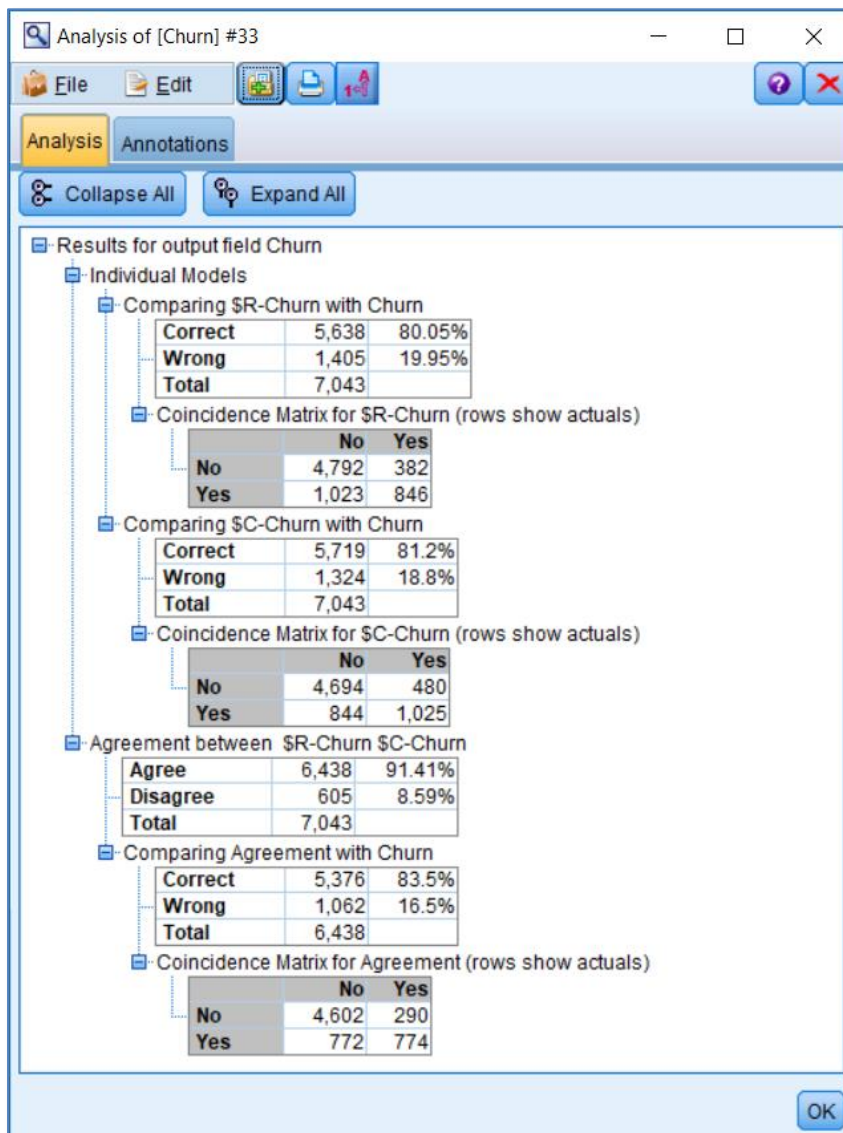


Figure 9.11 Analysis node showing performance comparison for two predictive models

The first part of the output shows the classification accuracy for the CHAID model (as denoted by the \$R-Churn variable). The second section shows the classification accuracy for the C5 model (as denoted by the \$C-Churn variable). The overall accuracy for the C5 model is slightly higher than the CHAID model (81.2% vs 80.05% respectively). Note that the C5 model is better at accurately predicting customers who churned compared to the CHAID model but worse at predicting the current customers. The third section of output looks at the degree of agreement between the two models: showing that both models gave the same predictions in 91.41% of cases. This represents 6,438 out of a total 7,043 records. Lastly, the output shows

that in the 6,438 records where both models agreed, the predictive accuracy was 83.5%. The final table shows a crosstab (or coincidence matrix) of these predictions against the actual outcomes.

Success Criteria Scenarios

So what success criteria might the decision makers within the telco organisation have established to choose a final model? In all of the following examples, the organisation's aims rest on its ability to proactively identify customers with a high risk of churning.

Scenario 1:

- *Currently around 100K customers cancel their contracts each month.*
- *Previous tests have shown that sending offers to a random group of 100K current customers approaching contract end dates each month, reduces the churn amount by 7K customers (7%).*
- *We would like to send 50K offers to customers with a high risk of churning with view to reducing churn by 14K (14%).*
- *The model should therefore identify the 50K current customers approaching their contract end date who have the highest likelihood of churning with a view to retaining at least 14K of them.*

Scenario 2:

- *Currently we have a monthly outbound email campaign that targets 400K customers approaching their contract end.*
- *Tests indicate that this retains around 15K customers who would otherwise have churned.*
- *However, the total cost of the offers is very expensive as we suspect many customers with a low likelihood of churning also redeem them.*
- *We would like to reduce the outbound mailing to 100K customers and still retain the 15K customers who would otherwise have churned.*

Scenario 3:

- *Currently we have a monthly outbound email campaign that targets 60K key customers approaching their contract end.*
- *The campaign offers the next month's standard call costs for free if they extend their contract by two months.*
- *However, the campaign makes a net loss of \$130K each month as many customers churn after the additional two months anyway.*
- *We would like to make a net profit of \$70K per month by targeting only those with a low likelihood of churning after accepting the offer.*

The first scenario directs the analyst towards finding the 50K customers with the highest likelihood of churning. The success criterion here is that of these 50K customers, the subsequent offer should enable the company to retain at least 14K customers. We should bear in mind the models we have built thus far have been focused on predicting which customers will churn, not which of those customers will respond to the retention offer. However, it's reasonable to assume that if an offer to a random selection of 100K customers succeeds in retaining 7% of churners, then the same offer to a highly targeted group should do much better.

In the second scenario, the aim is to reduce the number of offers being made whilst still retaining 15K customers. This is more about reducing the cost of the campaign as measured by the number of customers contacted with an offer (400K). Here the criterion is that the model should be four times better than a random approach as the organisation only wants to send the offer to 100K people without losing any additional customers.

The third scenario directly focuses on the profitability of the campaign. Specifically, the profits associated with offering free services to customers who do decide to churn anyway. This of course requires a data sample where high risk customers have already been identified (perhaps through an earlier churn model) and have then been made a retention offer. The aim here is to only make the retention offer to those customers that are likely to remain long enough with the organisation that the offer costs can be recouped, and a net profit realised. As we shall see in the next section, it's possible to make this selection by entering a few estimated costs and revenue values to identify the selection of customers where the business has the best chance of maximising the profit from a campaign.

The Evaluation Node



We have already seen how the analysis node will allow us to assess the accuracy of a classification model using crosstabs (or coincidence matrices). The Evaluation node is stored in the Modeler Graphs palette however and therefore allows us to visualise the model performance via a portfolio of charts. To see an example of this we can return to the currently open stream 'Section 9 start.str' as shown in figure 9.12.

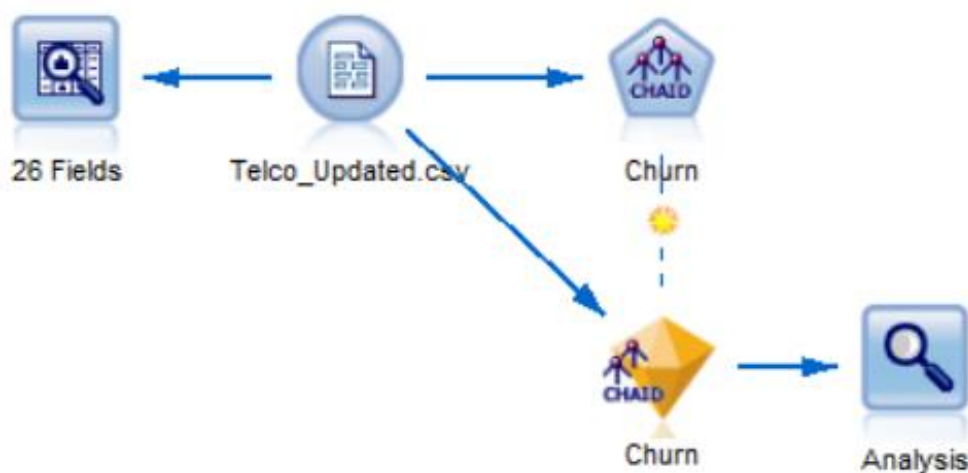


Figure 9.12 The currently open Modeler stream 'Section 9 start.str'

To attach an Evaluation node:

Click on the CHAID nugget in the stream to select it

From the Graphs palette within SPSS modeler:

Double click the Evaluation node

The Evaluation node will automatically attach itself to the model nugget. Figure 9.13 shows the updated stream.

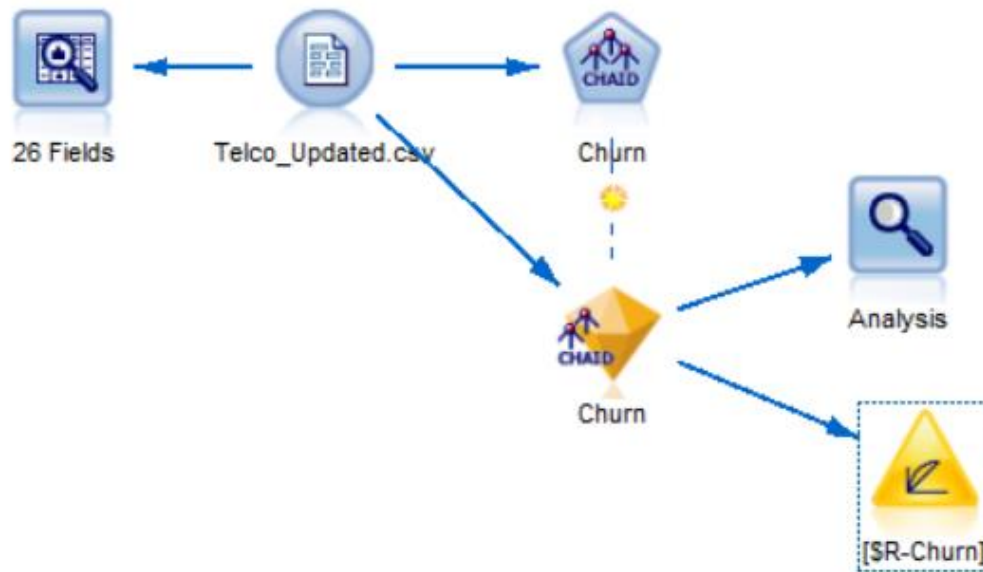


Figure 9.13 Evaluation node added to the current stream

Before running the node itself, it's useful to make one alteration to the default settings.

Right-click on the Evaluation node and edit it

Within the control dialog for the Evaluation node, check the box marked:

Include Best Line

You may notice at this stage that the default chart type in the Evaluation node is set to 'Gains'. Gains charts are a common way to show how well a classification model performs. Figure 9.14 shows the edited Evaluation node.

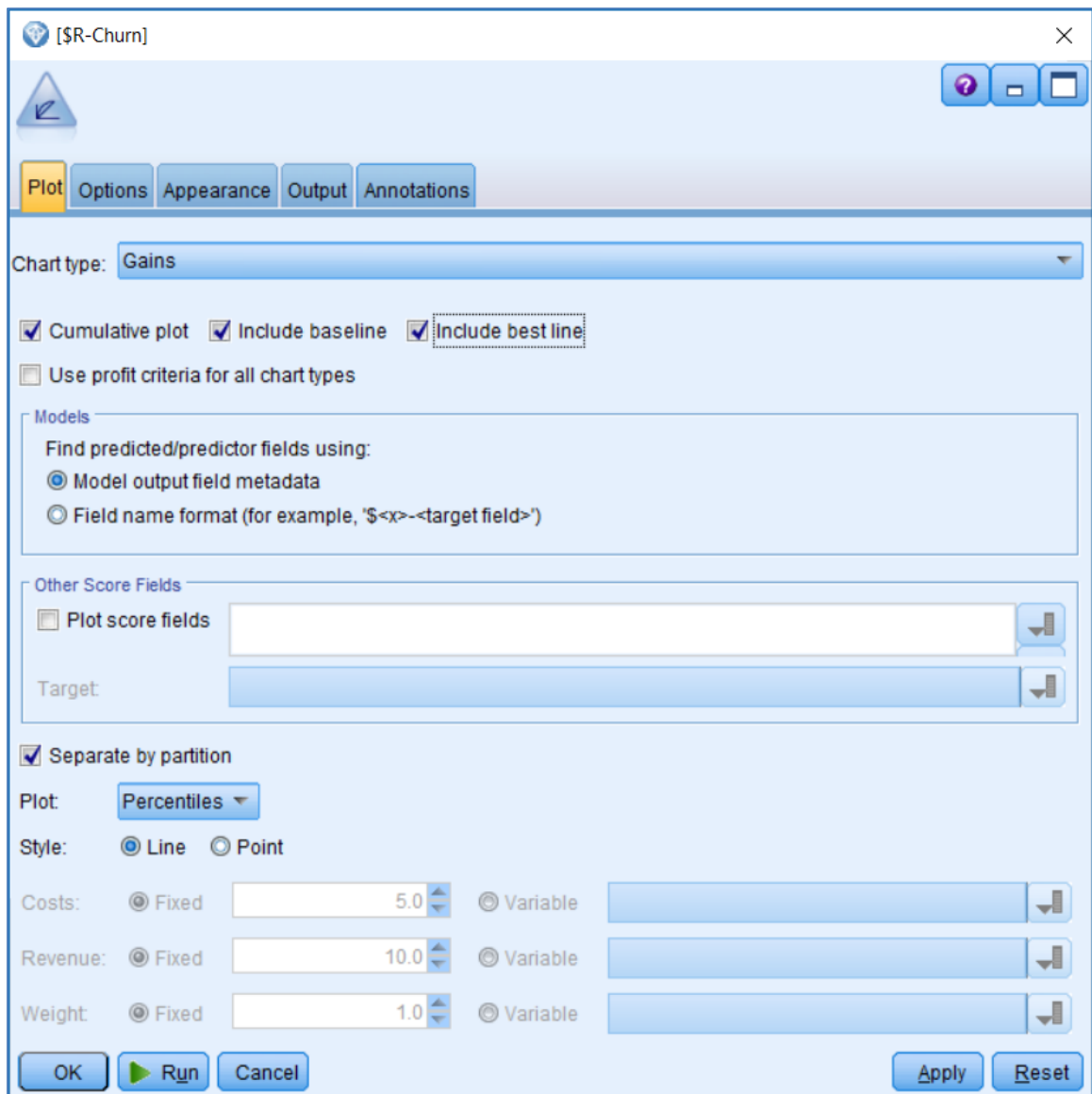


Figure 9.14 The edited Evaluation node with the 'Include Best Line' option selected

To execute the Evaluation node and view the Gains chart, click:

Run

Figure 9.15 shows the resultant Gains chart.

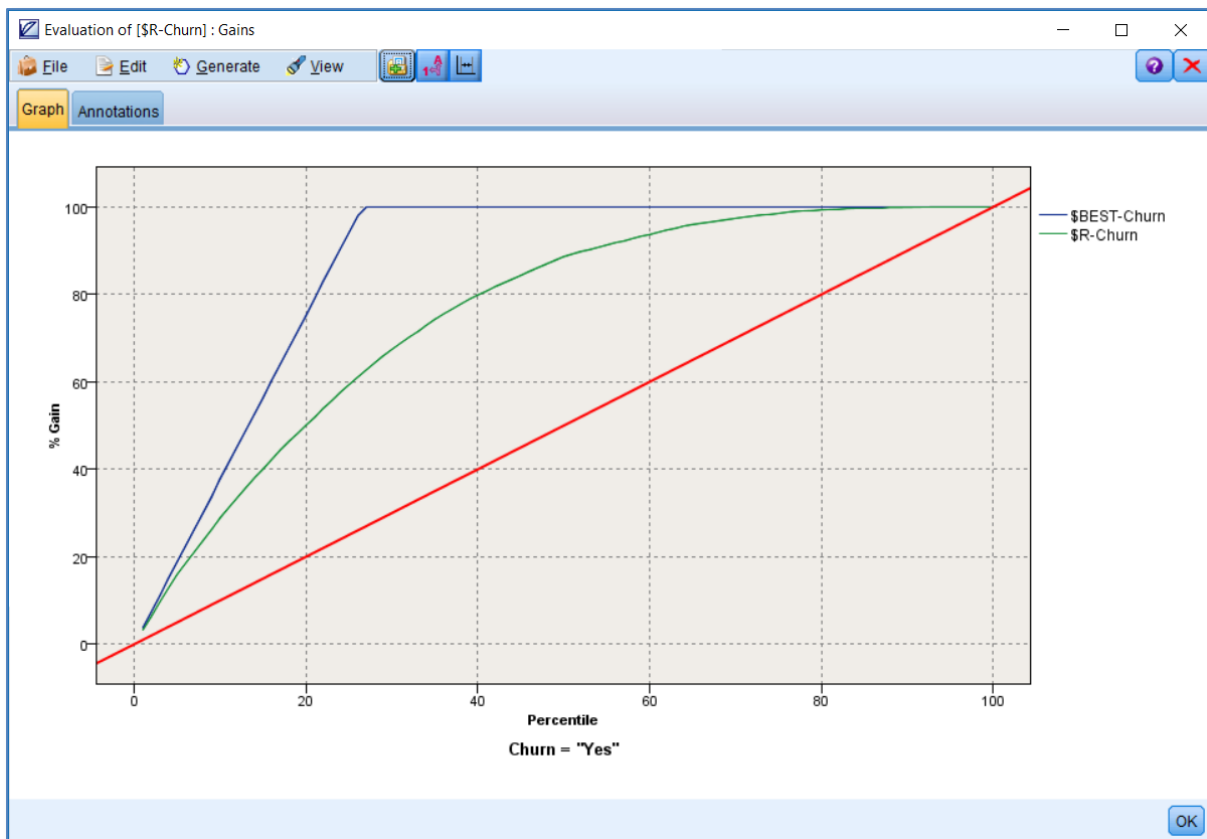


Figure 9.15 The Gains chart as generated by the Evaluation node with the 'Best Line' option displayed

The purpose of the Gains chart is to show the proportion of records in a target group that we can 'gain' by using the model. Perhaps the easiest way to make sense of it is to note that the chart itself only can focus on one group within the target field at a time. Here the default group of interest is those customers who have churned (as evidenced by the label on the horizontal axis indicating that 'Churn= "Yes"'). The diagonal red line within the chart simply tells us what proportion of records within this group we might expect to find if we were to randomly sample the data. The line therefore simply indicates that for example, using a random approach, we could only expect to find (or 'gain') 20% of the customers in the Churn group from 20% of the data, or indeed 50% of the churners by sampling 50% of the file. The top line (or 'Best line') however, represents what a perfect predictive model would like. Here we could gain 100% of the churners from 27% of the data. This is simply because within the sample dataset, 27% of the records belong to those customers who have churned. Having established what a random model and a perfect model would look like, the middle line shows us how many customers we might expect to find using the predictive model itself. Or alternatively, how much better the model is than random and how much worse it is than perfect. By moving the cursor along any of these lines, a pop-up label appears telling us what proportion of churners (the 'gain') we might detect as we increase our sample size. In this case, the model indicates that we could expect to find about 63% of the people who churned from 27% of the data. This is extremely useful, because if we wished to reduce the number of customers that we intend to contact to reduce churn, the gains chart will tell us what

proportion of the churners the model could detect. In this case, by using the chart, we could select the 50% of customers with the highest estimated risk and still expect to capture 88% of the customers who churn (see figure 9.16).

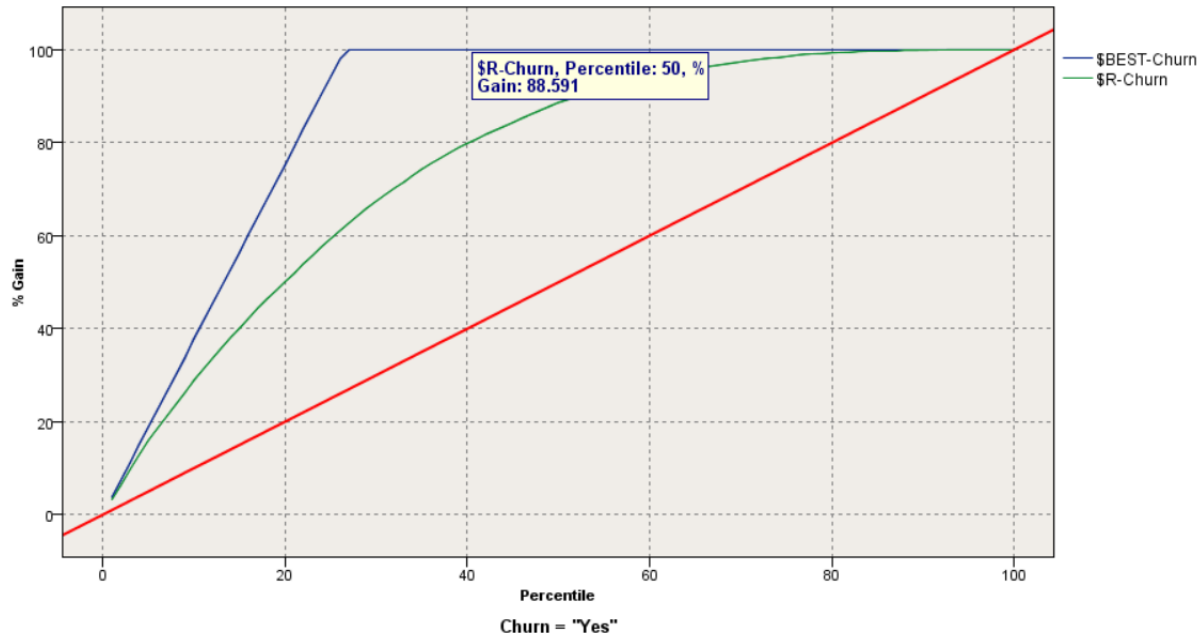


Figure 9.16 Using the predictive model to select the highest risk 50% in the sample we would expect to detect around 88% of the customers who churn

We can also use the Evaluation node to compare multiple models. To show this, within the currently open stream, 'Section 9 start.str':

Right-click on the Evaluation node and select 'Copy Node' from the pop-up menu

Return to the previously opened stream 'Section 9 Two Model Comparison' and paste the Evaluation node into the stream

Attach the pasted Evaluation node to the second model nugget

Figure 9.17 shows the updated stream.

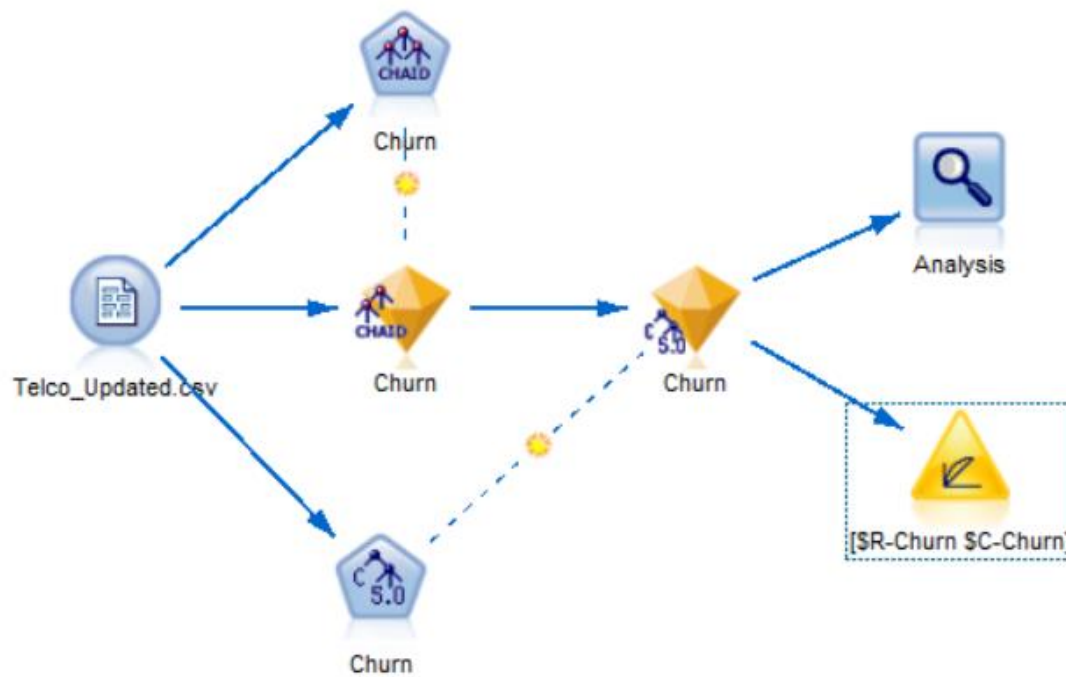


Figure 9.17 Attaching an Evaluation node downstream of two model nuggets

Right-click and run the node

Figure 9.18 shows the resultant chart

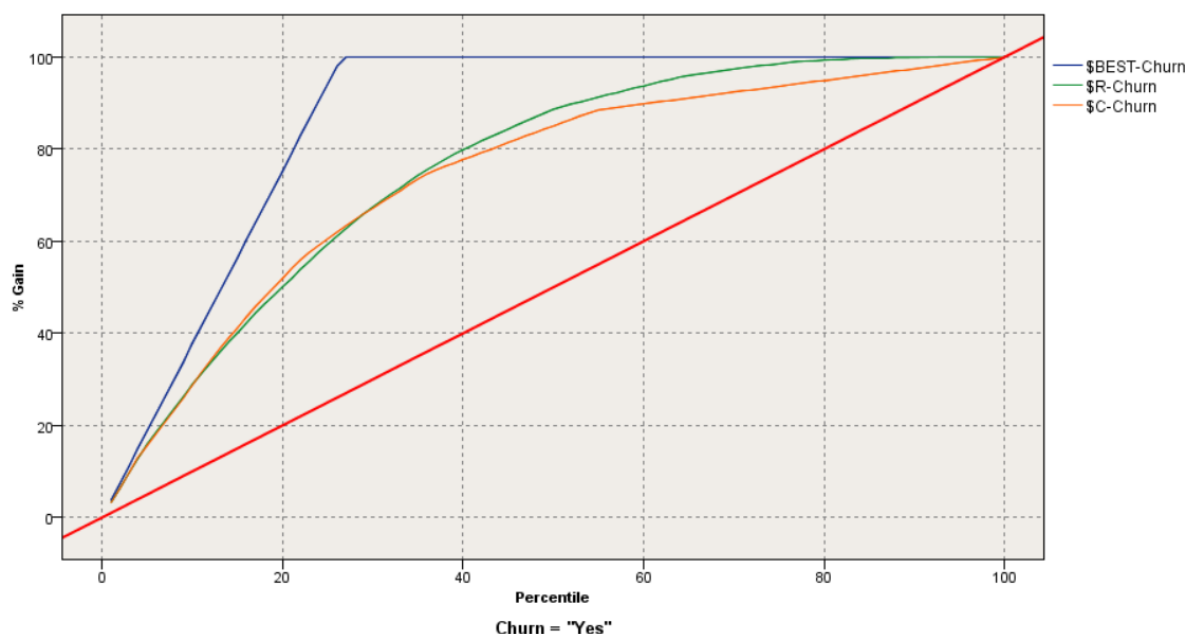


Figure 9.18 Gains chart generated by an Evaluation node displaying the performance of two classification models simultaneously

We can see from the Gains chart that the two models exhibit subtle differences in their classification accuracy. If the goal of the analyst was to select the top 20% of the file

containing customers with the highest risk of churning, the C5 model (\$C-Churn) has a slightly better performance than the CHAID model (\$R-Churn). If however the analyst wanted to select the 40% of cases with the highest risk, the CHAID model seems to be slightly better than the C5.

To show how this can be done, within the chart itself, from the main menu, click:

View

Interactions

Figure 9.19 illustrates this process.

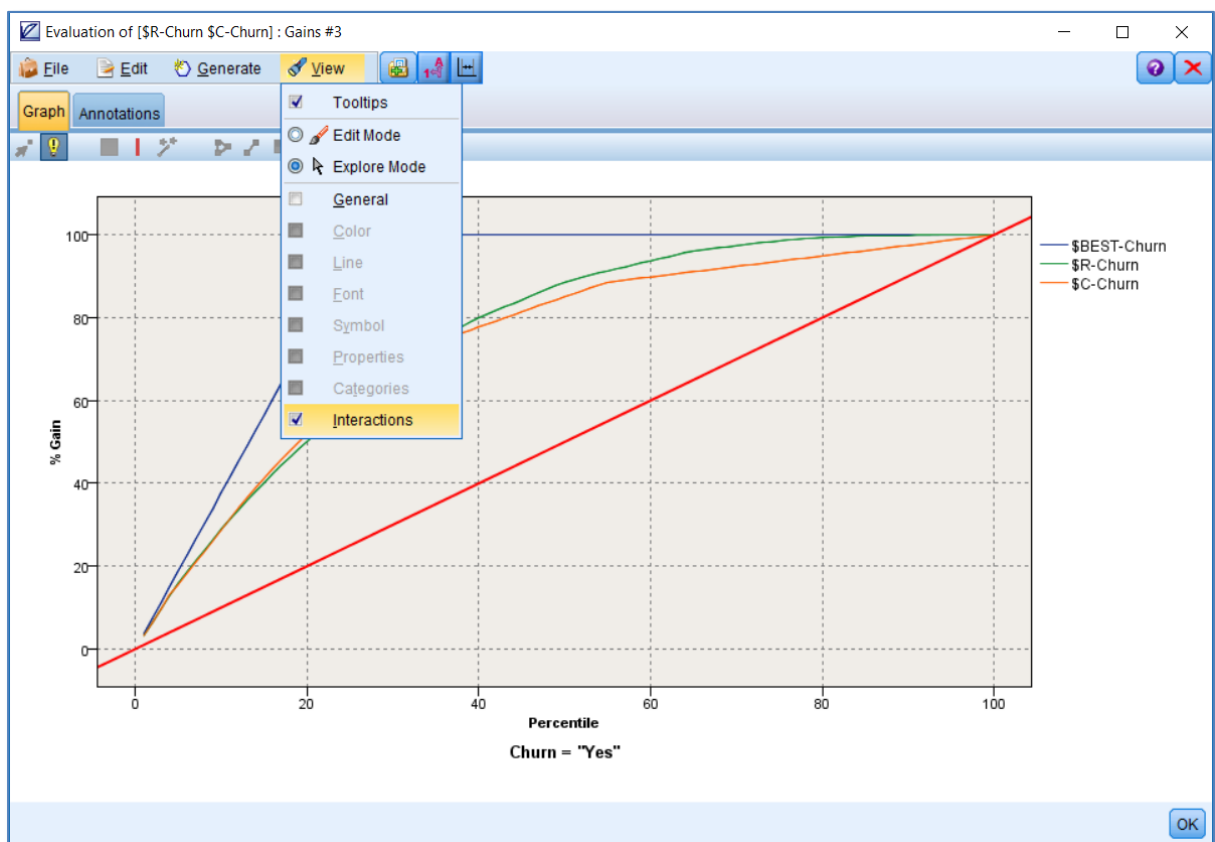


Figure 9.19 Switching on Interactive mode in a Gains chart.

On the chart toolbar click the band selection tool:



Using the tool:

Move the cursor to around the 40th percentile on the horizontal axis and click

Figure 9.20 illustrates this.

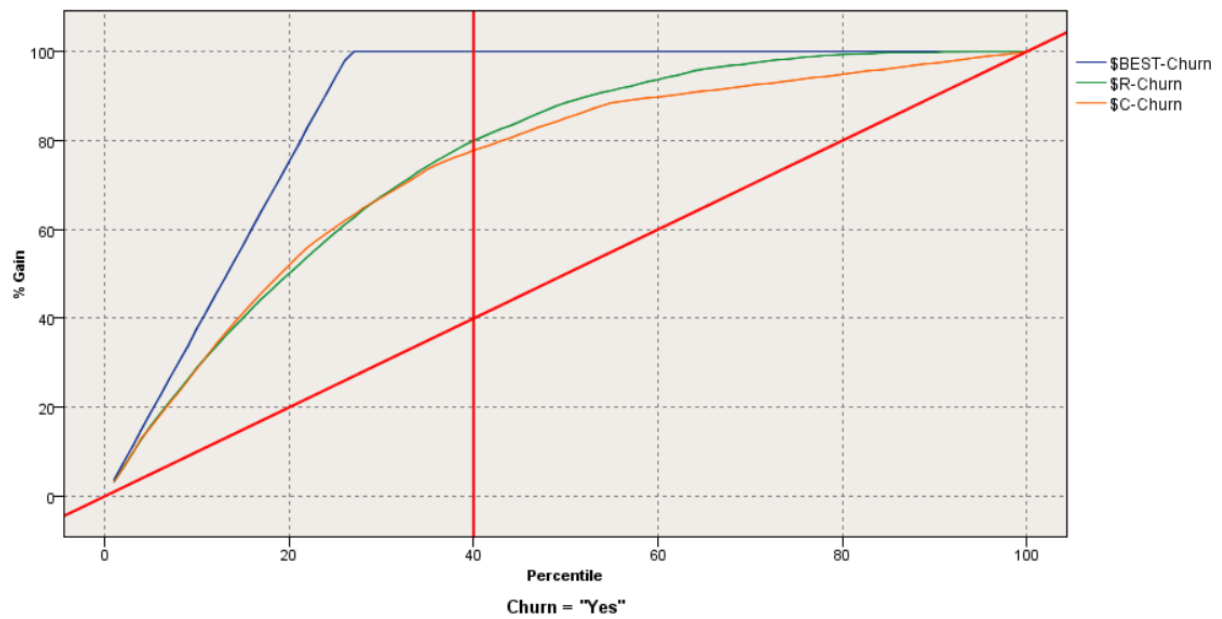


Figure 9.20 Using the Band Selection tool to interact with the Gains chart

Now:

Right-click on the area to the left of the red selection line

From the pop-up menu choose:

Generate Select Node for Band

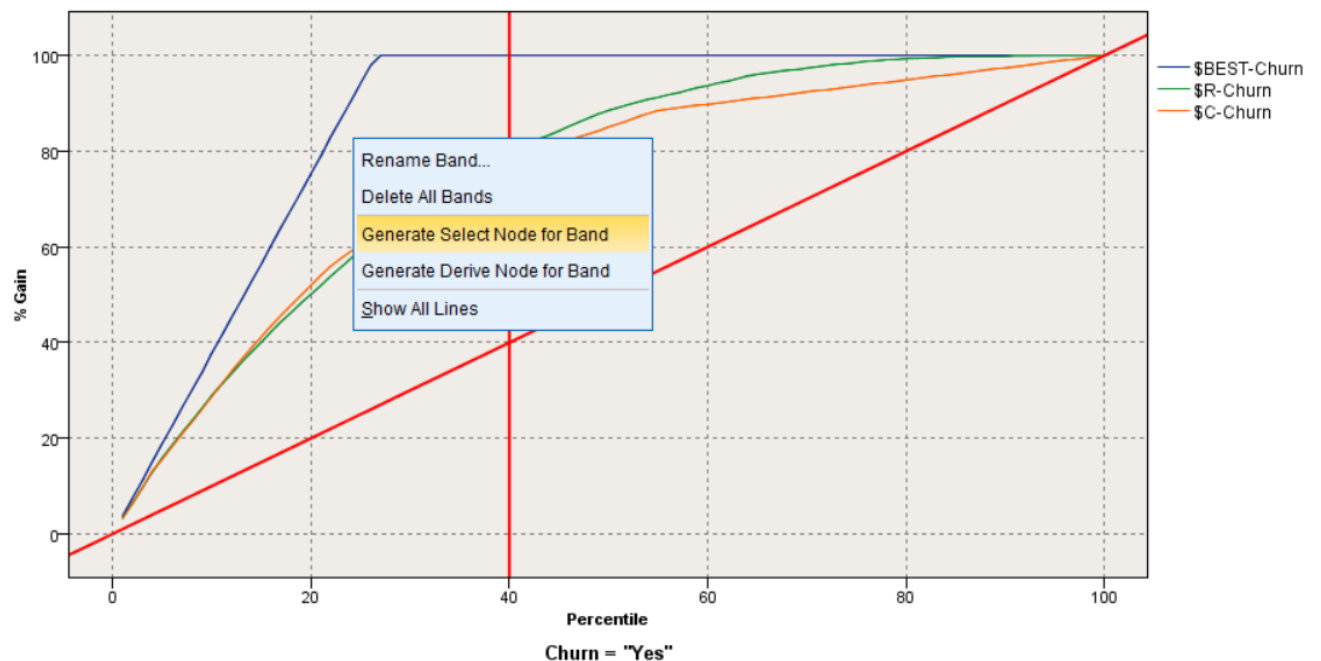


Figure 9.21 Generating a Select node for the highest risk 40% of data

A pop-up menu appears asking you to choose which model the selection should refer to (see figure 9.22).

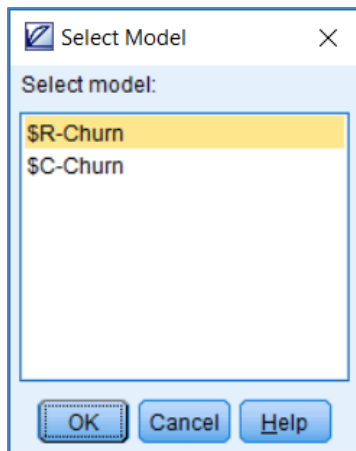


Figure 9.22 Select model pop-up menu

From the pop-up menu choose the variable containing the CHAID model's predictions:

\$R-Churn

OK

A new Select node (labelled 'Band 1') is added to the stream. This node will use the CHAID model scores to select the 40% of records that contain 80% of the customers who churned in the sample data. We can attach the node downstream of the CHAID nugget as shown in figure 9.23 and display the records in a Table node as shown in figure 9.24.

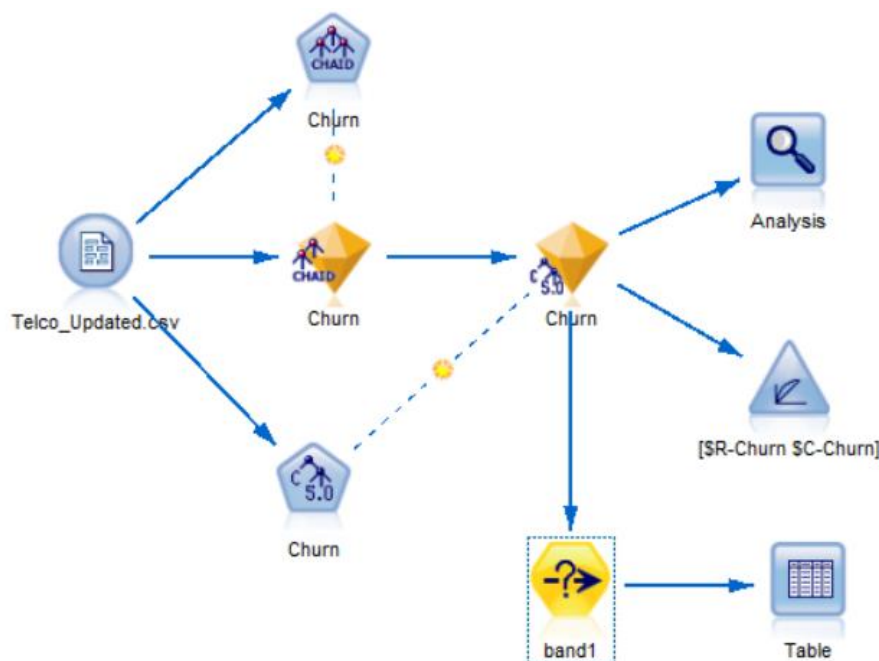


Figure 9.23 The generated Select node that selects the 40% of customers containing 80% of the churners

	back	Customer_Tier	\$R-Churn	\$RC-Churn	\$C-Churn	\$CC-Churn	
1	0.000	d. Refunded	Yes	0.636	Yes	0.549	
2	0.000	c. Standard	Yes	0.636	Yes	0.683	
3	0.000	c. Standard	No	0.635	No	0.811	
4	0.000	c. Standard	No	0.500	Yes	0.549	
5	0.000	c. Standard	No	0.635	No	0.811	
6	0.000	c. Standard	No	0.500	Yes	0.683	
7	0.000	c. Standard	No	0.500	No	0.627	
8	0.000	c. Standard	Yes	0.864	Yes	0.864	
9	0.000	c. Standard	Yes	0.864	Yes	0.864	
10	0.000	c. Standard	No	0.635	No	0.811	
11	0.000	c. Standard	No	0.635	No	0.811	
12	0.000	c. Standard	No	0.635	No	0.811	
13	0.000	c. Standard	Yes	0.636	Yes	0.549	
14	0.000	c. Standard	Yes	0.864	Yes	0.864	
15	0.000	c. Standard	Yes	0.636	Yes	0.549	
16	0.000	c. Standard	No	0.500	Yes	0.549	
17	0.000	c. Standard	No	0.500	Yes	0.549	
18	0.000	c. Standard	Yes	0.636	Yes	0.549	
19	0.000	c. Standard	No	0.500	Yes	0.549	
20	0.000	c. Standard	No	0.635	No	0.811	

Figure 9.24 Table containing the 40% of customers comprising the 80% of churners

The Partition Node



A fundamental problem with the creation of predictive models is the uncertainty as to how the model will perform when applied in the real world. There are number of techniques that analysts employ to simulate how a model might behave when applied to new data and one of the most popular is the use of training and test samples. Training samples are simply extracts of data (usually randomly chosen) that are used to develop the model. As we have already seen, Modeler's predictive algorithms are able to automatically build models using default settings against representative historical data where the outcome of interest is known. Depending on the nature of the algorithm, this model 'training' process can be achieved using statistical techniques or machine learning methods to select, transform and incorporate variables into a final model. The model itself might be expressed as a mathematical formula, a set of rules or a composite of hidden transformations, but the overall goal is to predict the outcome with the highest degree of accuracy. It's essential therefore that the sample training dataset is sufficiently large and unbiased so that the resultant model can then be applied in a

real-world context to generate accurate estimates. Nevertheless, every data sample is unique in its own way, and as such, there is always a danger that the training process results in a model which is overly specific to the idiosyncrasies of the training data. In analytics, this is known as 'overfitting'. An overfitted model will tend to perform poorly when applied to a different sample of data from the one it was trained on. To get around this, many researchers retain a sub-sample of the main dataset for testing purposes. If the analyst can test the model to see if it performs well on both the training and the test samples, then they will have greater confidence that it will perform well when deployed against data where the outcome to be predicted is not yet known. The Partition node within SPSS Modeler allows us to do just this.

To see how we can use the partition node to compare model performance on separate training and test samples:

Return to the stream 'Section 9 start.str'

Delete the existing CHAID model nugget from the stream

From the Field Ops palette:

Select and add the Partition node to the stream

Drag the connection between the Data Source node and the CHAID node so that passes through the Partition node

Figure 9.25 shows the edited stream at this stage.

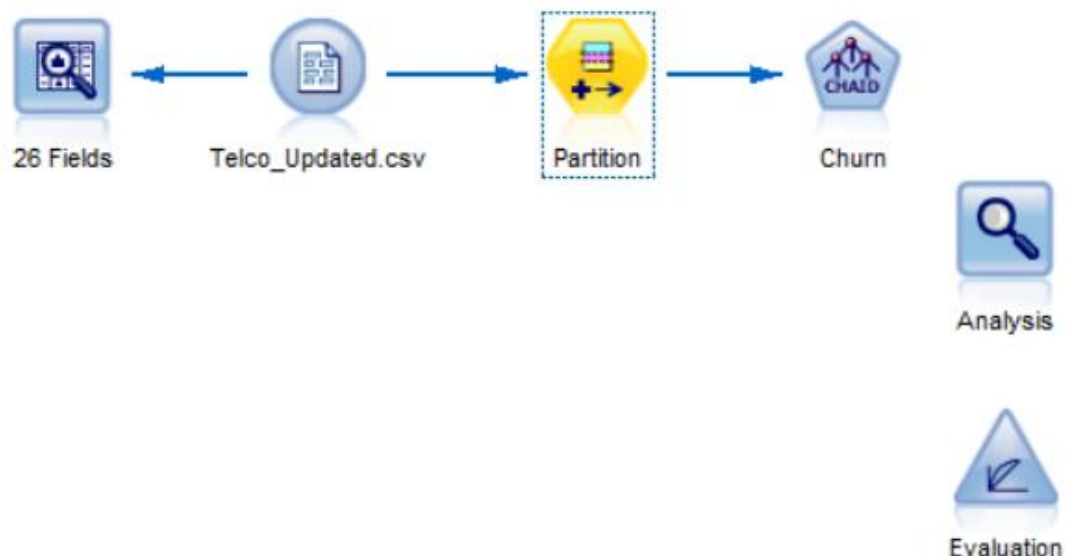


Figure 9.25 Adding a Partition node to the existing stream file 'Section 9 start.str'

Before running the CHAID node, we can take a look at how the Partition node is configured. To do so:

Right-click on the Partition node and edit it

Figure 9.26 shows the edited Partition node.

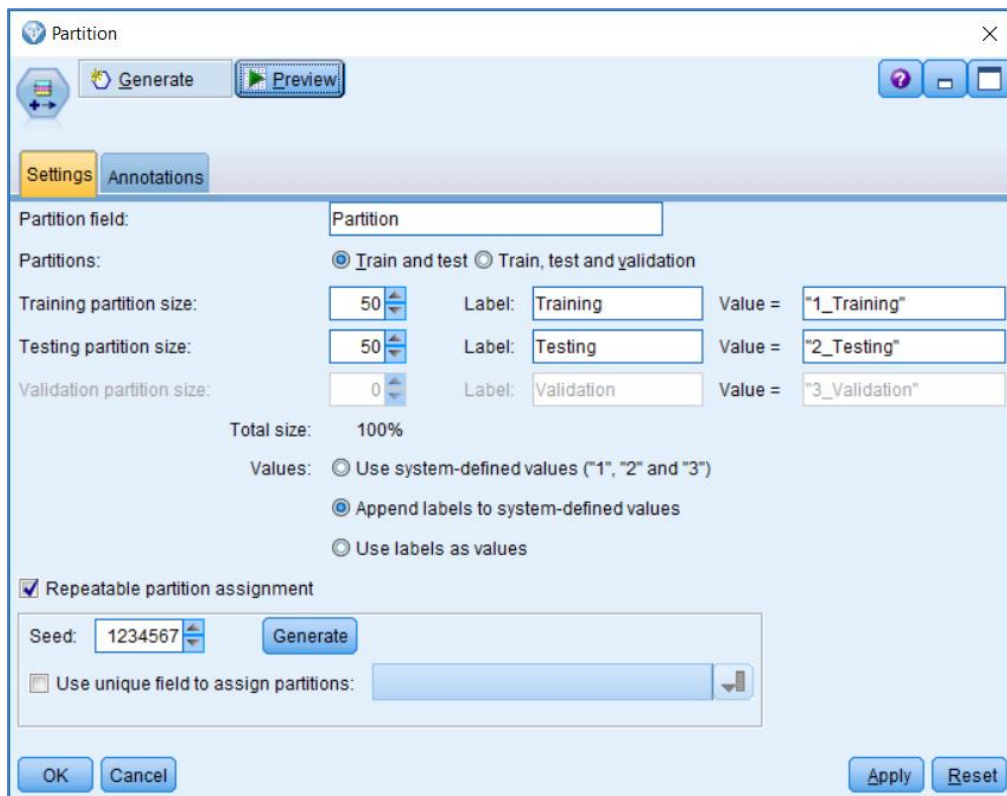


Figure 9.26 The Partition node dialog showing its default settings

By editing the Partition node, we are able to define what proportion of the sample data will be assigned as the 'Training' group and the 'Testing' group. Depending on the size of the main dataset, analysts often assign a smaller proportion for testing purposes than the training set. Note that you can also define a third group here: a Validation sample. Some analysts like to create this additional random partition when they are trying to select the best performing model from a number of candidates and they suspect that a particular model might be performing well on the Testing sample as the result of chance. You can also see that the node allows the analyst to define or generate a random seed number. Using the same seed number allows us compare one model to the next as the same cases will be randomly assigned to the training and testing samples. To illustrate how we might use the Partition node:

Change the Training partition size to 70

Change the Testing partition size to 30

Change the Random Seed number to 7654321

Figure 9.27 shows the edited Partition node control dialog.

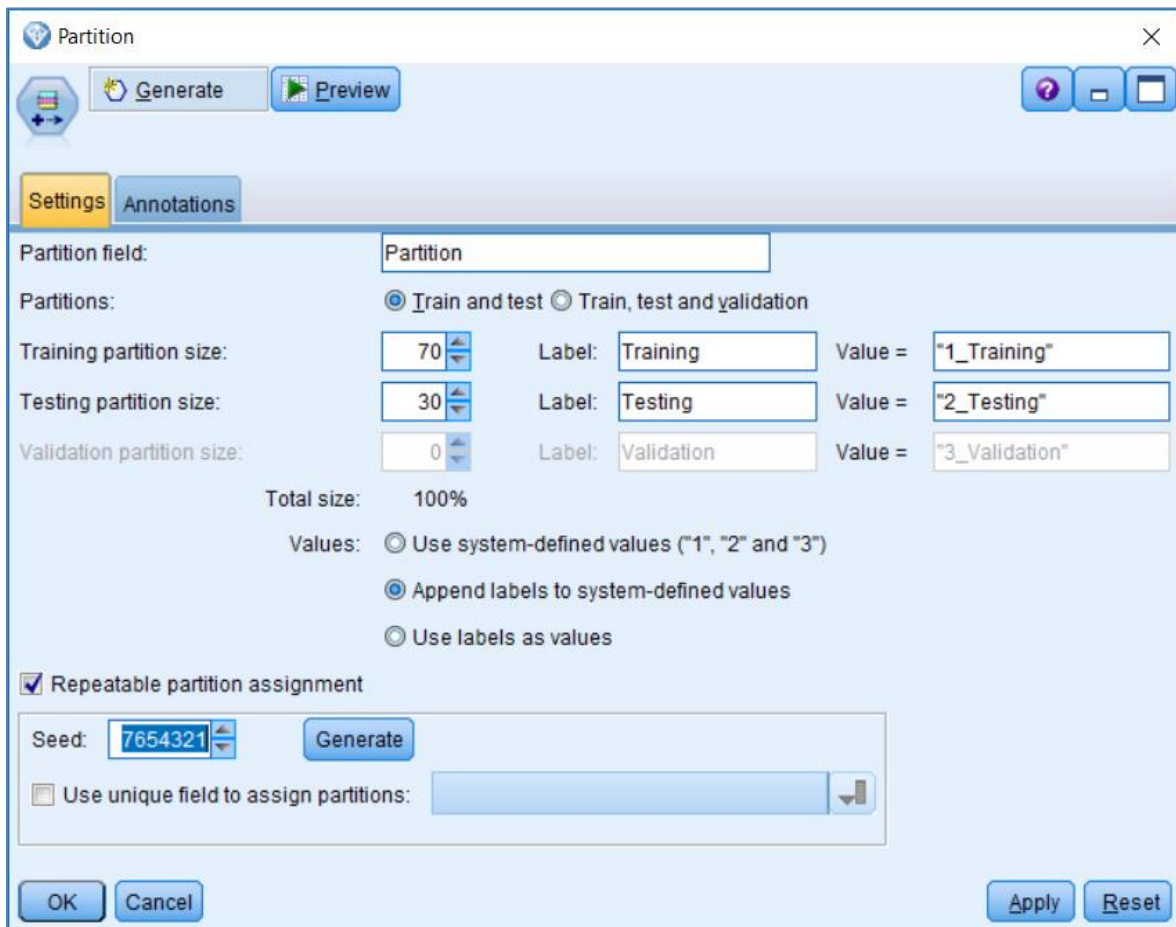


Figure 9.27 The edited Partition node dialog

Click OK and run the CHAID node

The CHAID node once again generates a CHAID model nugget.

Attach the CHAID model nugget to the Analysis node and the Evaluation node

Figure 9.28 shows the updated stream.

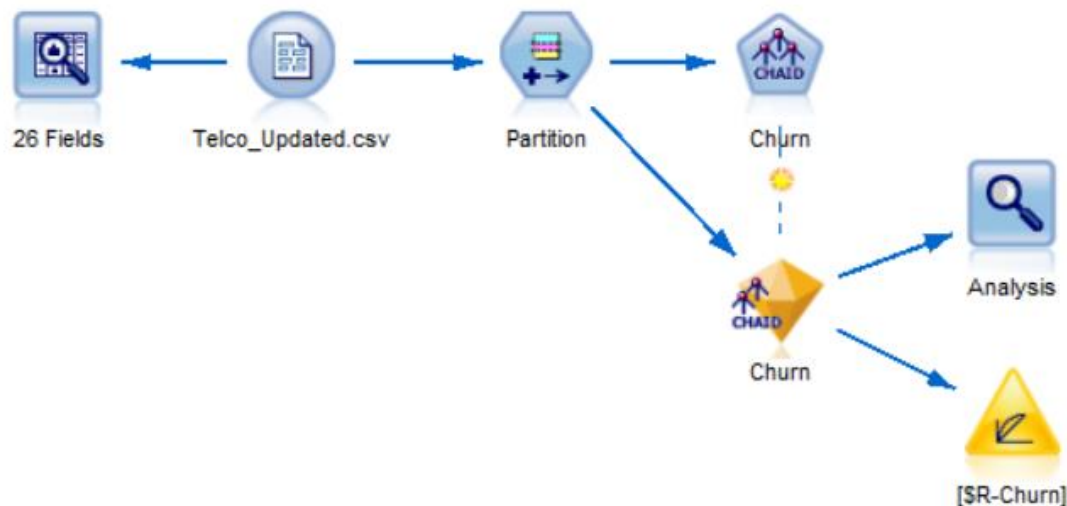


Figure 9.28 The newly generated CHAID nugget added to the Analysis node and Evaluation node

To see the effect of building a model downstream of a Partition node:

Run the Analysis node

Figure 9.29 shows the Analysis node output.

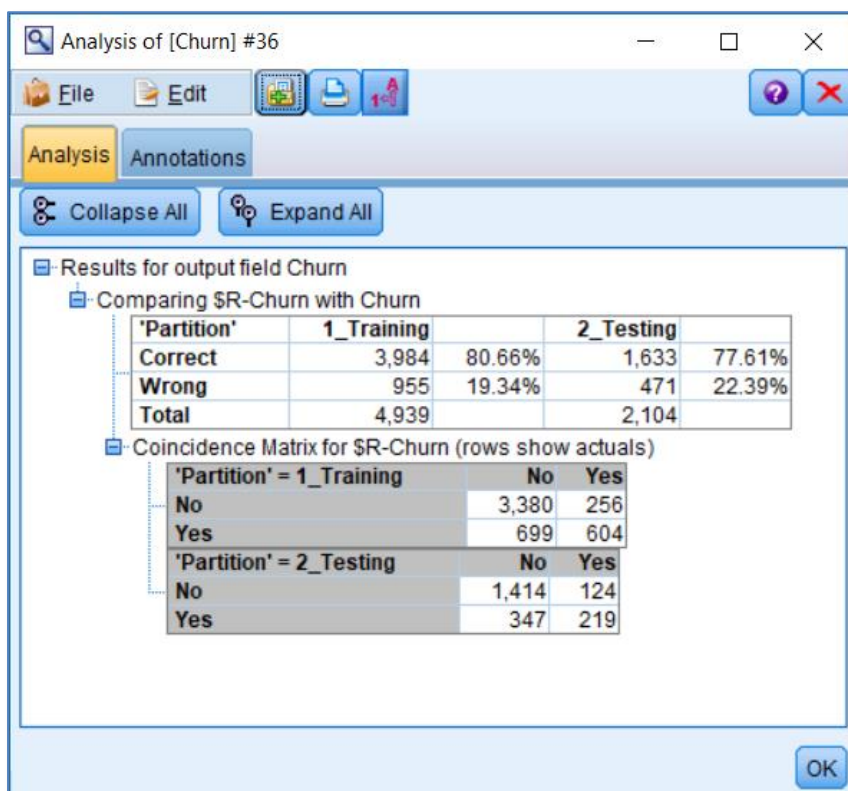


Figure 9.29 The Analysis node showing the relative performance of a partitioned model

As we can see from the results, not surprisingly, the model appears more accurate when applied to the Training sample (80.66% correct) than the Testing sample (77.61% correct). However, the difference is quite small so on this evidence we might conclude that the model shows little sign of overfitting. At this stage we could try to improve the model or test to see if it continues to give similar results by re-building it with different random splits (by generating new random seed numbers). For now though, we can:

Return to the stream and run the Evaluation node

Figure 9.30 shows the output from the Evaluation node.

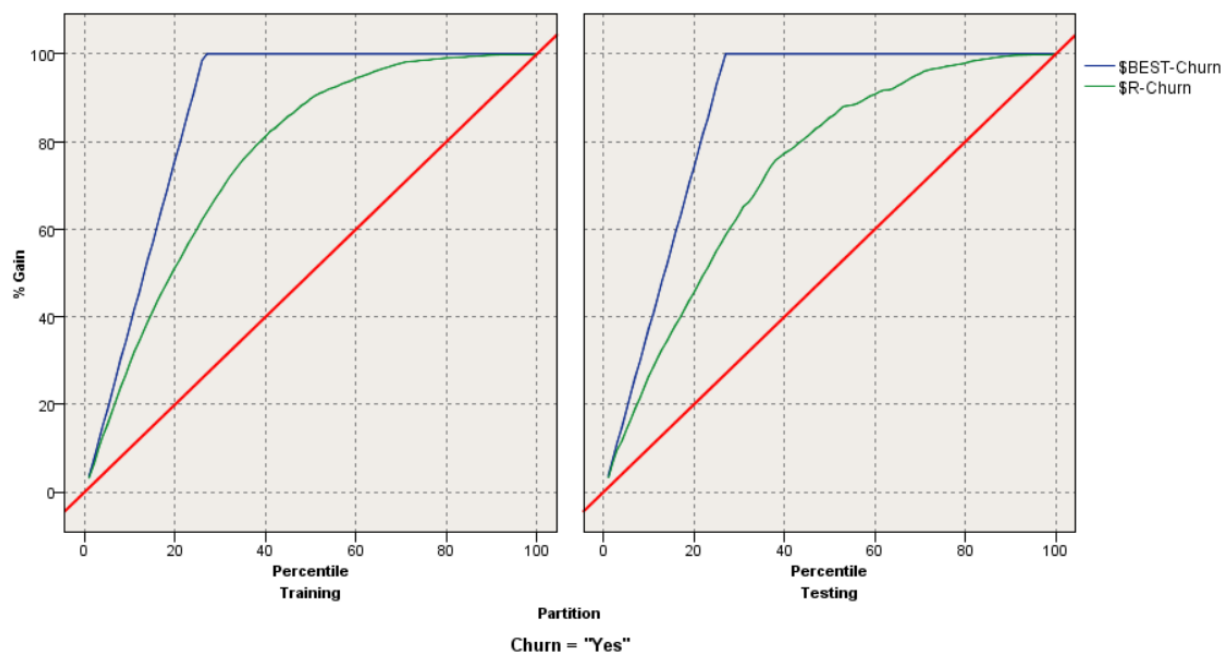


Figure 9.30 The output from the Evaluation node displaying the relative performance of a partitioned model

As we can see, the Evaluation node has now created two Gains charts for the testing sample and the training sample respectively. The charts show a similar story to Analysis node in that the model performs slightly more poorly on the Testing sample. The partition node has created this split in the data by creating a special partition field in the data. To view the field:

From the Output palette, add a Table node to the CHAID model nugget

Figure 9.31 shows the Table node added to the stream.

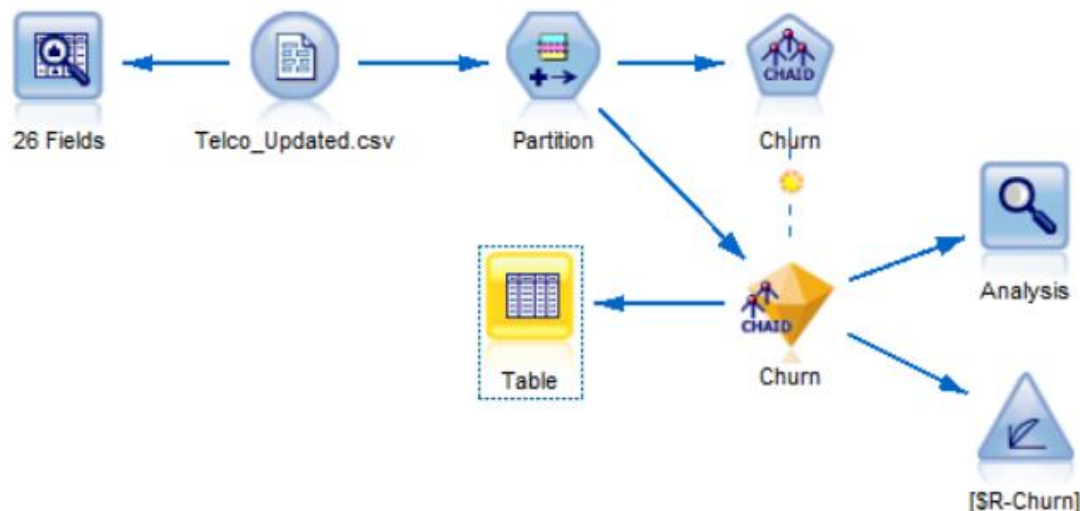


Figure 9.31 Table node added to the stream to illustrate the Partition field

Run the Table node

Figure 9.32 shows the output from the Table node and the generated Partition field.

	Payment_Type_3Grp	Average_Monthly_Bill	Very_Loyal	Cashback	Customer_Tier	Partition	\$SR-Churn	\$SRC-Churn
2527	ard_Payment	95.188 F		0.000	c. Standard	1_Training	Yes	0.613
2528	ccount_Debit	82.000 F		0.000	c. Standard	1_Training	Yes	0.613
2529	ard_Payment	66.688 F		0.000	c. Standard	1_Training	No	0.892
2530	voice	78.062 F		0.000	c. Standard	2_Testing	Yes	0.613
2531	ard_Payment	78.812 F		0.000	c. Standard	1_Training	Yes	0.633
2532	voice	18.875 F		0.000	c. Standard	2_Testing	No	0.994
2533	ard_Payment	85.938 F		0.000	c. Standard	1_Training	Yes	0.613
2534	ard_Payment	79.250 F		0.000	c. Standard	1_Training	Yes	0.613
2535	ard_Payment	69.688 F		0.000	c. Standard	1_Training	No	0.983
2536	ccount_Debit	99.250 F		0.000	c. Standard	2_Testing	Yes	0.613
2537	ard_Payment	90.125 F		0.000	c. Standard	1_Training	Yes	0.613
2538	ard_Payment	73.000 F		0.000	c. Standard	1_Training	No	0.534
2539	ccount_Debit	96.250 F		0.000	c. Standard	1_Training	Yes	0.613
2540	ard_Payment	20.625 F		0.000	c. Standard	2_Testing	No	0.908
2541	ard_Payment	82.562 F		0.000	c. Standard	2_Testing	Yes	0.613
2542	ccount_Debit	92.688 F		0.000	c. Standard	1_Training	Yes	0.613
2543	voice	17.250 F		0.000	c. Standard	1_Training	No	0.908
2544	ard_Payment	18.438 F		0.000	c. Standard	1_Training	No	0.908
2545	ard_Payment	98.375 F		0.000	c. Standard	1_Training	Yes	0.613
2546	voice	17.250 F		0.000	c. Standard	1_Training	No	0.908

Figure 9.32 Output from the Table node showing the random values of the Partition field

As we can see, the Partition node simply creates a field called 'Partition' and randomly assigns the values '1_Training' or '2_Testing', whilst honouring the requested proportions to the records in the dataset. The relevant nodes within Modeler detect the presence of this field and take account of it when building or evaluating models downstream of it.

The Auto Classifier Node



Within Modeler there are a large number of algorithms that can be used to predict classification outcomes such as customer churn. The Auto Classifier node can help to narrow down which technique (or combination of techniques) will yield the best model. Assuming that the modelling fields have been correctly configured in the stream Type node (or Type tab in the Data Source node) and that the target has been correctly typed as a flag or nominal field, the Auto Classifier node can automatically execute several model-building algorithms at once. The goal of this procedure is usually to retain the resulting best models according to a pre-specified criterion such as overall accuracy, lift or profit. To see this procedure in action, from the Section 9 folder open the Modeler stream:

Section 9 Auto Classifier.str

Figure 9.33 shows the stream.



Figure 9.33 The Modeler stream 'Auto Classifier.str'

From the Modelling palette:

Select and attach an Auto Classifier node to the Partition node in the stream

(If you can't see the Auto Classifier node, make sure the node filter tab marked 'All' is highlighted on the left side of the palette).

Figure 9.34 shows the updated stream.

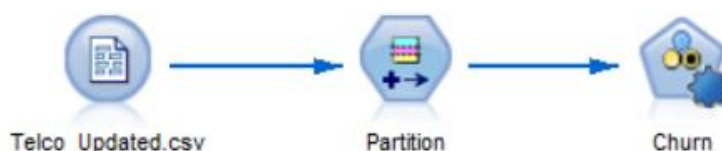


Figure 9.34 The Modeler stream 'Auto Classifier.str' with the Auto Classifier node added

To see the options associated with this node:

Double-click the Auto Classifier node to edit it

Figure 9.35 shows the edited Auto-Classifer node.

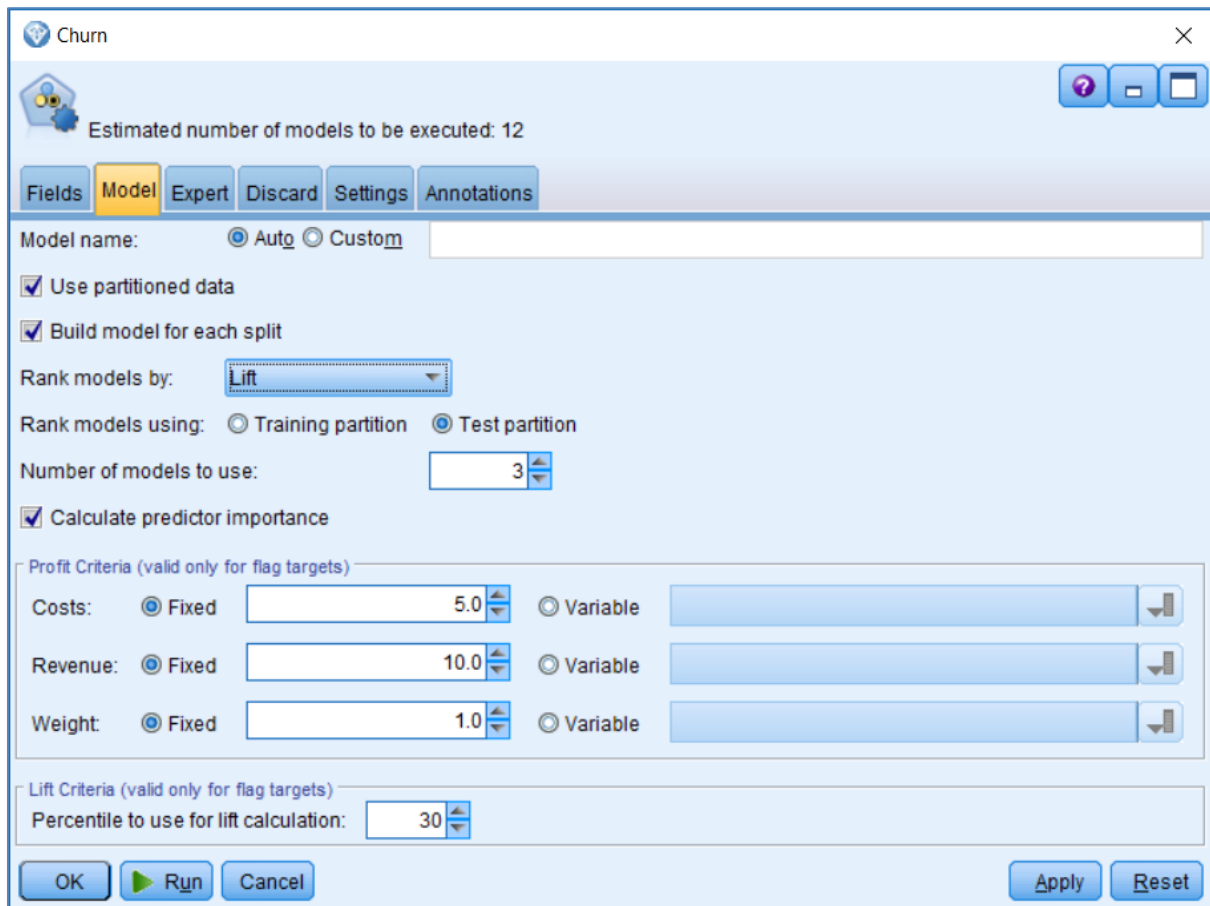


Figure 9.35 The edited Auto Classifier node showing the options within the Model tab

Within the edited node, some of the more interesting options in the Model tab include how the procedure ranks the Models it builds. Here we can see the default option is to rank models by 'Lift'. In fact, the options for ranking models include:

- **Overall Accuracy** – This is simply the overall (or average) accuracy the model exhibits when considering all the groups in the target field. This measure is more useful when the group sizes are close to equal.
- **Area Under the Curve** – The Area Under the Curve (or AUC) measure looks at the false positive vs true positive rate. Recall that the Gains chart displays a diagonal line representing a random classifier. The diagonal line simply indicates that using a random model, you would expect to find 50% of the responders (or churners) from randomly sampling 50% of the data. The AUC score for a model that followed the random line would therefore be 0.5 (scores lower than 0.5 would indicate the model was worse than random). A score for a perfect classification model would be 1. As you

might expect, most models are somewhere between these two values with higher values associated with higher accuracy in predicting the outcome of interest in the target group (usually the category denoted with values such as 'Yes' or 'T' or '1').

- **Profit** – As the dialog shows, it's possible to include costs and benefit parameters when the target field is binary (flag). This is useful if we can associate numerical values with the outcomes such as when trying to select a model that will drive the most profit for a marketing campaign. Alternatively, it could be used to select a model that minimises the financial loss associated with outcomes such as asset failures. Note that the user can enter specific parameter values or identify variables that record the revenue or costs associated with the outcomes.
- **Lift** – The lift value is a measure of how much better the model does at classifying the data (i.e. predicting the outcome) compared to a random model (or the baseline proportion). Here the proportion of churners is about 26%. So, a random model predicting every case to be a churning would be right about 26% of the time. This would generate a lift value equal to 1.0. If however the model was better than random, and was able to predict the proportion of churners with 52% accuracy it would be twice as accurate as the random approach and would generate a lift value of 2.0. The higher the lift value, the better than random the model is. Here the Lift measure is based on top 30% of data where the model is most confident.
- **Number of Fields** – The last method to rank the models is simply by the number of fields. Models based on fewer fields are regarded as more desirable as they are likely to be simpler and more efficient. In Statistics, such models are sometimes described as having greater 'parsimony'.

Two other things to note from this tab are that firstly, by default, the procedure creates a nugget based on the best three models and that this value can be altered. Secondly, models are ranked based upon their performance on the Testing group as identified by the Partition node.

Before we run the procedure, let's examine the range of algorithms the Auto Classifier uses by default. Click the tab marked:

Expert

Figure 9.36 shows the Expert tab in the edited Auto Classifier node.

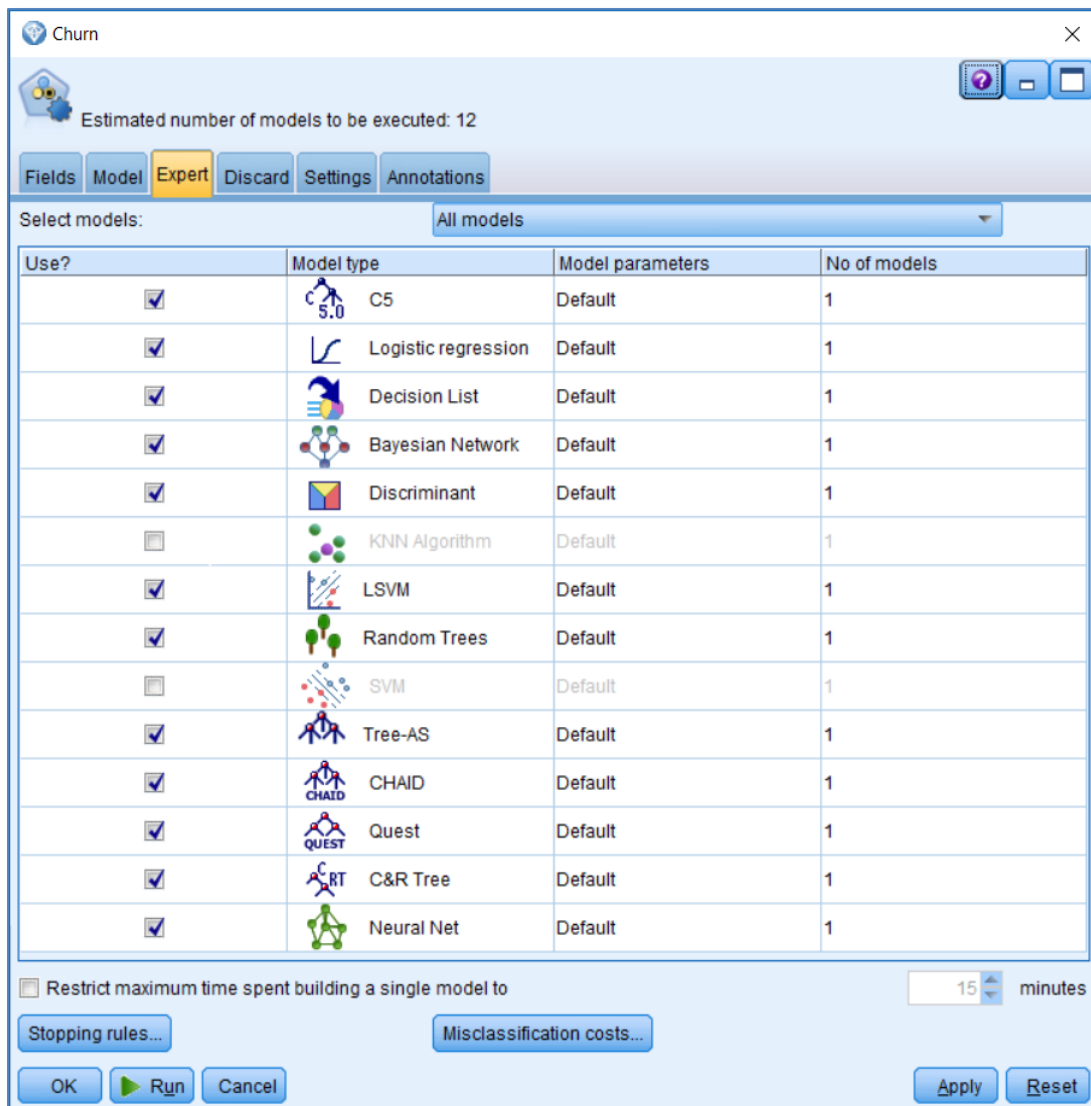


Figure 9.36 The edited Auto Classifier node showing the options within the Expert tab

Not only can we decide to include or drop individual algorithms from the Auto Classifier process, but we can also request that multiple models are built within the same techniques by editing an algorithm and specifying that additional parameters or settings are tried. To illustrate this, within the Model Parameters column:

Click the cell marked Default next to the C5 model type

Click:

Specify

Figure 9.37 shows the resulting sub-dialog.

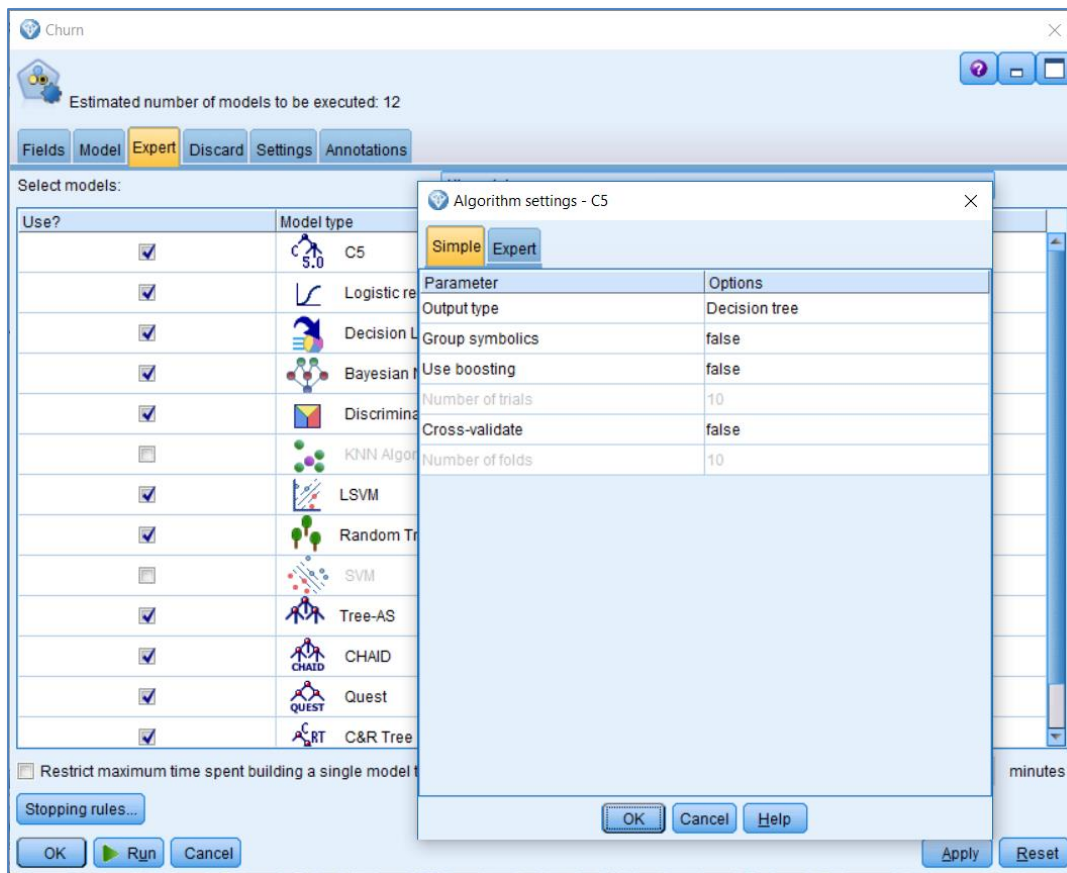


Figure 9.37 The Model Parameters sub-dialog for the C5 model type

Edit the cell next to the Output type parameter

From the drop-down menu request that the procedure also includes a C5 Ruleset by clicking:

Both

Figure 9.38 shows this:

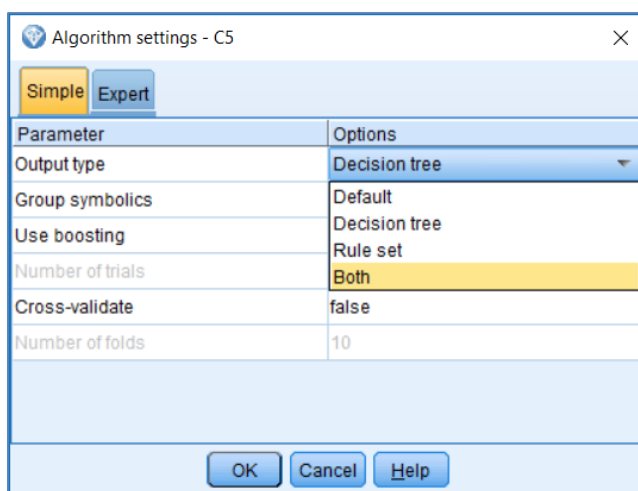


Figure 9.38 Requesting that a Ruleset model is built as a decision tree within the C5 model type

Click:

OK

We are returned to the main Expert tab within the Auto classifier. The cell corresponding to the C5 model type under the column header 'No. of models' now indicates that two C5 models will be built. We can also de-select some of the algorithms. To illustrate, uncheck the boxes in the column marked 'Use?' next to the following algorithms:

LSVM

Random Trees

Tree-AS

Figure 9.39 shows the updated Expert tab.

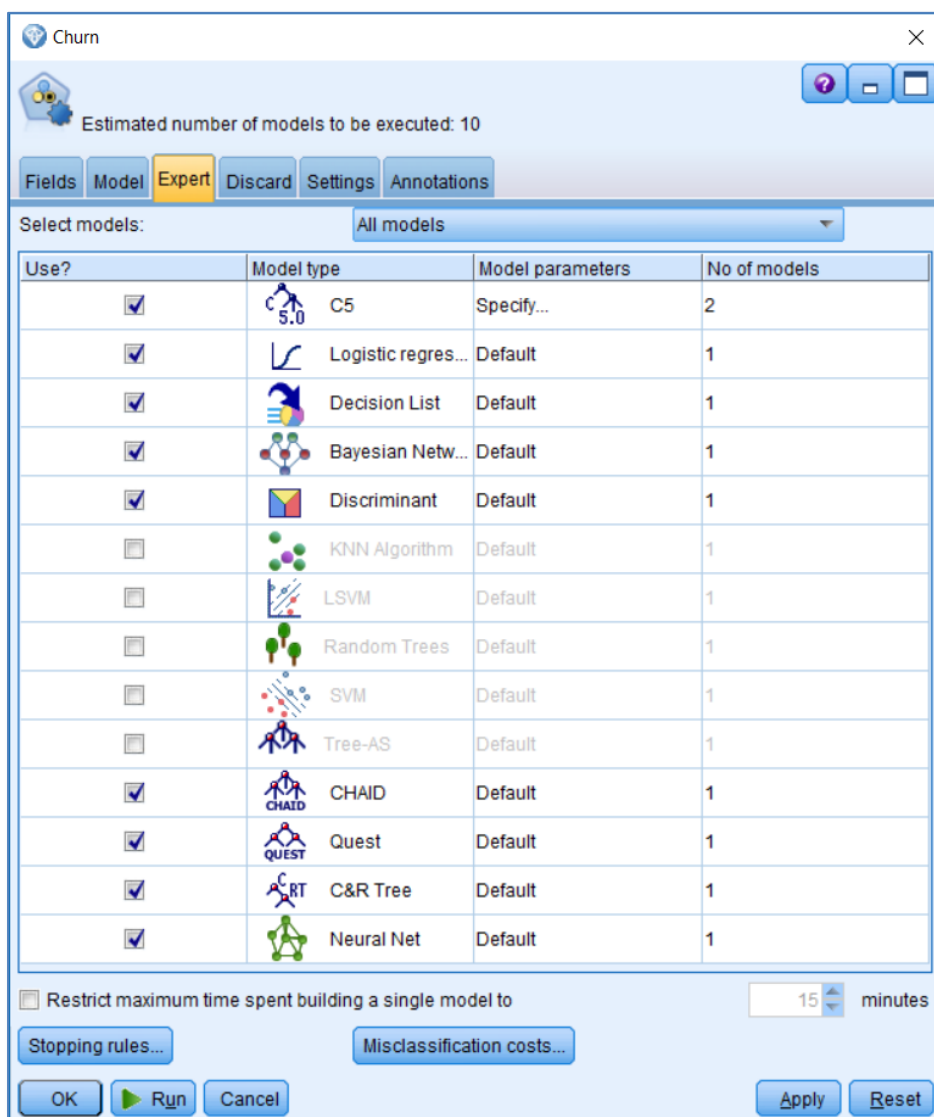


Figure 9.39 The updated Expert tab within the edited Auto Classifier node

Now if we run the procedure, Modeler will attempt to build 10 separate models retaining the 3 completed models with the highest lift value when applied to the Test partition group. Click:

Run

A progress window appears showing how many of the models are to be built and any which fail to complete or which are discarded.

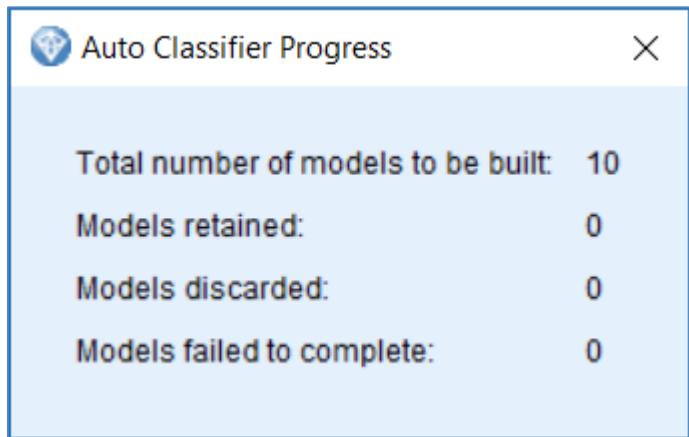


Figure 9.40 Auto Classifier Progress window

The Auto Classifier model is nugget is created as shown in figure 9.40.

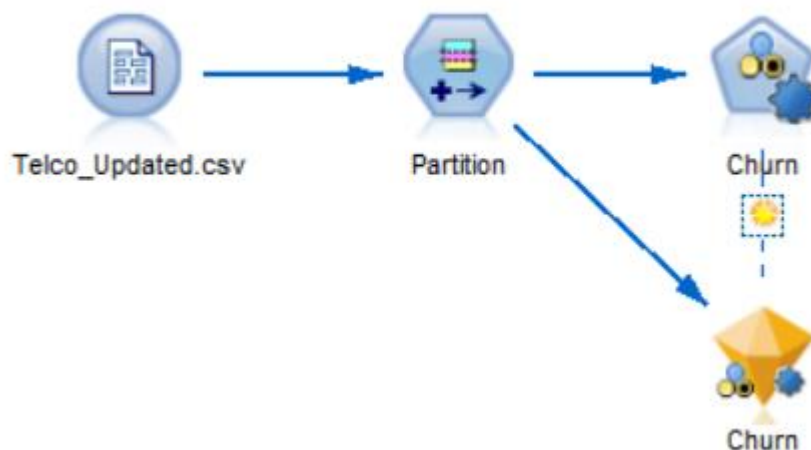


Figure 9.41 Auto Classifier model nugget added to the stream

To view the its contents:

Double-click the Auto Classifier model nugget

Figure 9.42 shows the contents of the nugget.

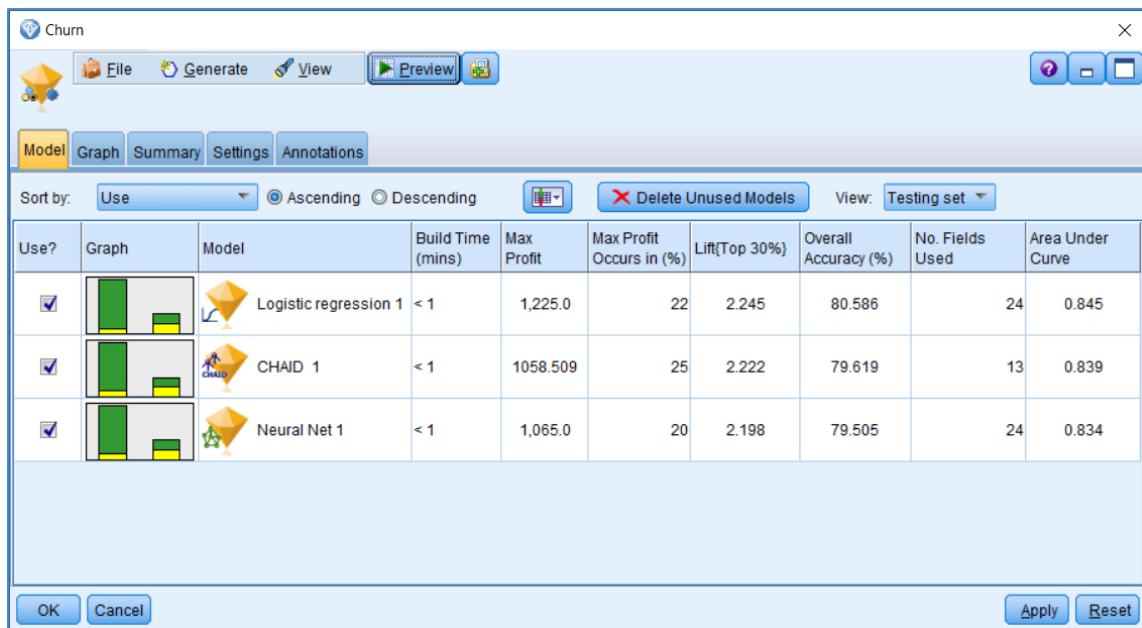


Figure 9.42 The results of the Auto Classifier procedure based on the Testing set

In our example, the procedure has selected three model types, Logistic Regression, CHAID and Neural Network (different results may occur if using a different build version of Modeler). This initial output screen contains a wealth of information that we may summarise here:

- A series of check boxes under the column marked 'Use?' indicating whether or not the model is to be retained or dropped.
- A clustered bar chart that shows the misclassification rate for each model. This can be double-clicked so the user can get a more detailed view.
- A model nugget for the individual model type that the user can double-click and browse to see more details.
- A series of performance measures showing, for example, the model build time, Lift value, overall accuracy, number of fields used and AUC values. In this case, the Logistic Regression model has the highest lift value, so it is shown at the top of the list.
- The option to click on a drop-down view box and switch between the statistics for the models' performance on the Testing set or the Training set

To see if the model performed equally well on the Training set, click the drop-down menu next to the view label and choose:

Training Set

Figure 9.43 shows the resulting model performance output,

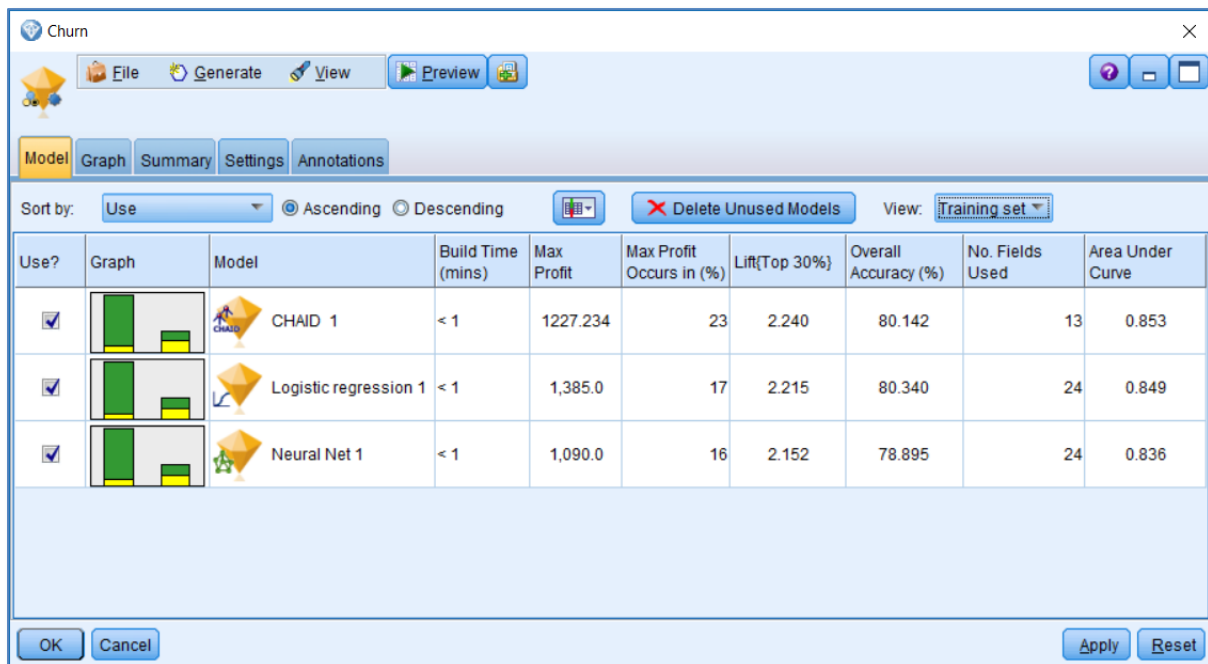


Figure 9.43 The results of the Auto Classifier procedure based on the Training set

We can see that in actual fact, the CHAID model performed better than the Logistic Regression model in terms of the lift value when applied to the Training set. However, the Logistic Regression model gave superior results on the Testing set. To view the contents of any model nugget we need only double-click it. As an example:

Double-click the Logistic Regression nugget

Click the Advanced tab

As figure 9.44 shows, Logistic Regression models are displayed as a series of statistical coefficients and as such are quite different from the rule or tree-based models such as C5 and CHAID.

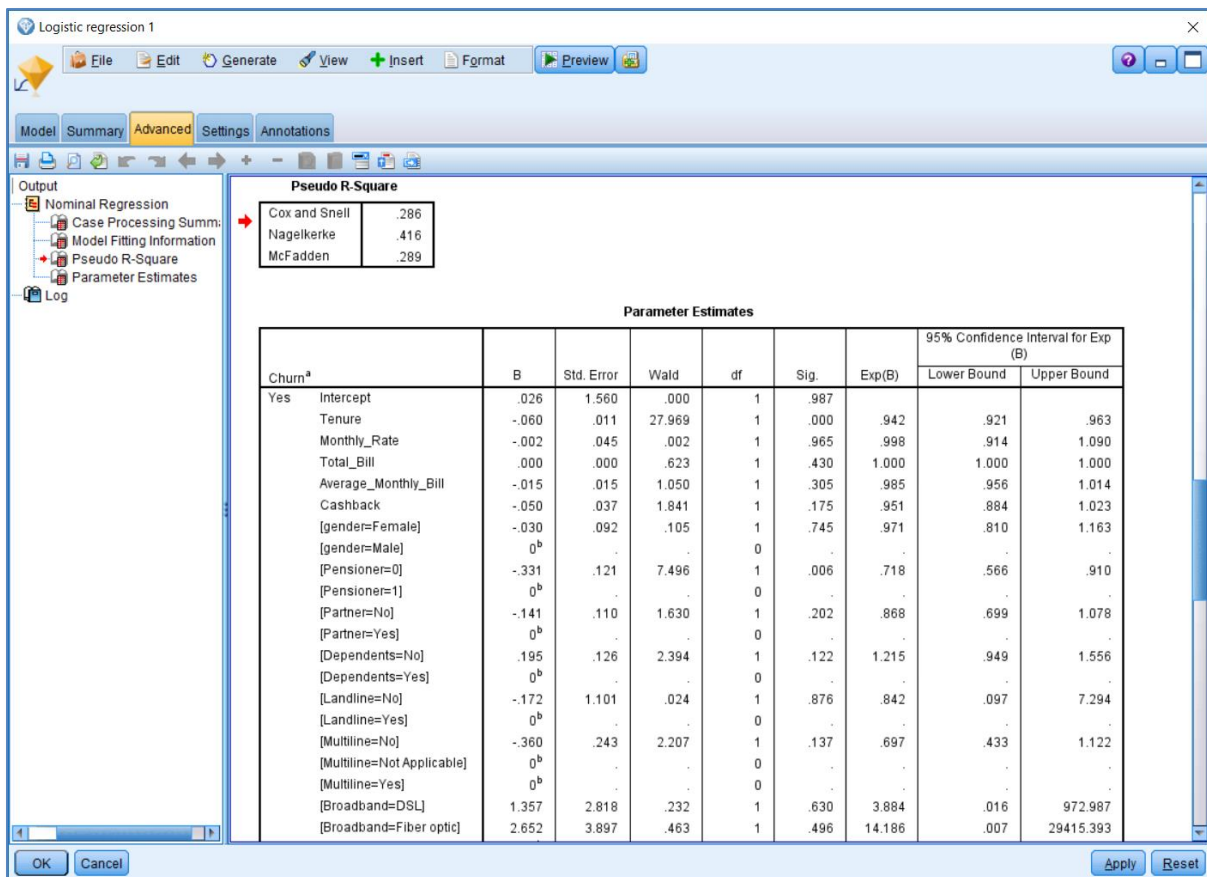


Figure 9.44 Logistic Regression output from the Logistic model in the Auto Classifier node

To return to the Auto Classifier output, click:

Cancel

As well as clicking on the individual clustered bar charts next to each generated model, we can view the performance of the auto-classifier nugget using all three models combined in a single clustered bar chart. To do so, click the tab marked:

Graph

Figure 9.45 shows the output.

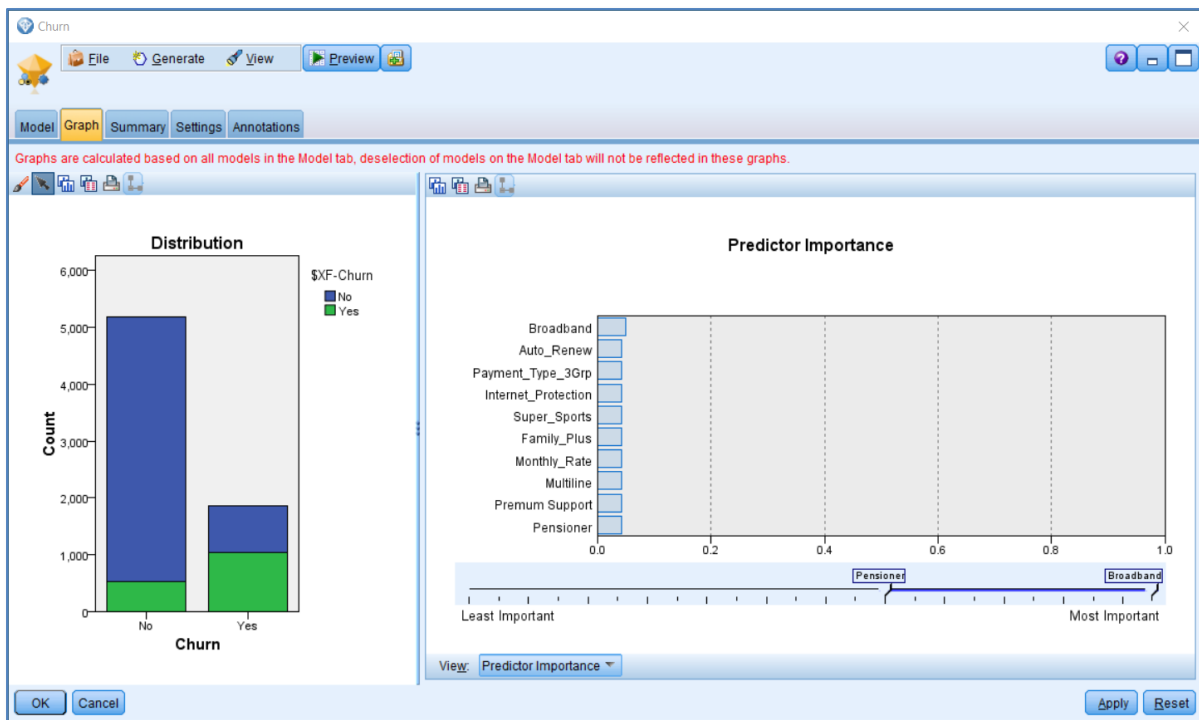


Figure 9.45 The Graph output from the Auto Classifier model nugget

The graph tab shows the overall classification chart for the combined models. A combination of models such as this are often referred to as an Ensemble model. The bars show the actual status of the customers whilst the colours show the predicted outcomes. You can see that the combined models do a better job of predicting the customers who have not defected compared to those who have churned. You can also see that the predicted outcome variable the Auto Classifier produces is labelled '\$XF-Churn'. The '\$X' indicates that this is an ensemble model. The predictor importance chart shows which variables are most important in the three models that the procedure has chosen. A note at the top of the screen reminds us that the charts are based on all the models in the nugget and will not be affected or updated by changing whether an individual model is selected or not in the Model tab. We can of course evaluate the ensemble model performance by attaching an Analysis or Evaluation node to the nugget. From the output palette:

Choose an Analysis node and attach it to the Auto Classifier nugget

Edit the node so that the 'Coincidence Matrix' is included

Now choose an Evaluation node and attach it to the Auto Classifier nugget

Edit the node so that the 'Best Line' is included

Figure 9.46 shows the updated stream.

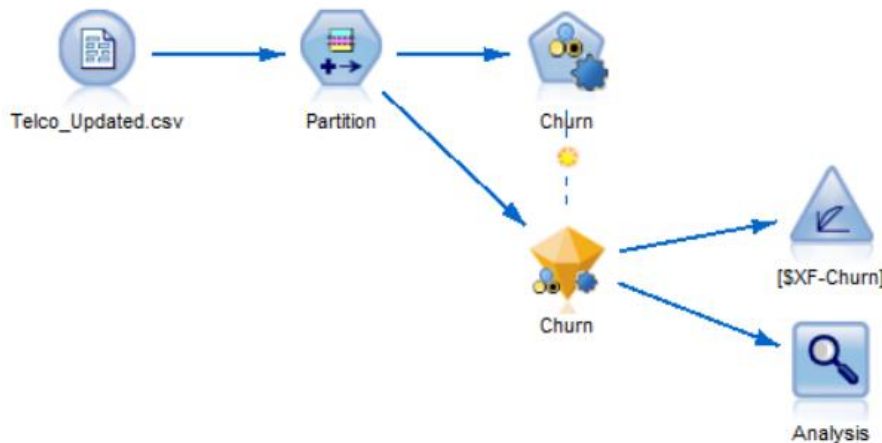


Figure 9.46 The Auto Classifier ensemble model with an Evaluation and Analysis node attached

Run the Evaluation node

Figure 9.47 shows the resulting output.

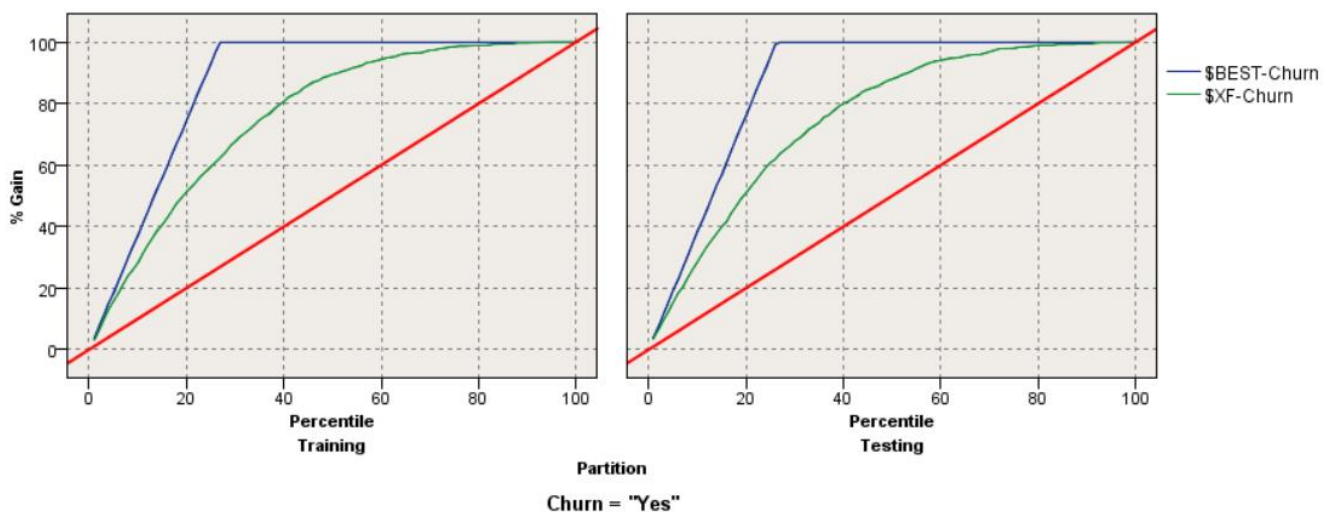


Figure 9.47 Gains charts from the Evaluation node showing the performance of the combined (ensemble) models in the Auto Classifier nugget

By default, Modeler combines the scores from the different models using 'Confidence-Weighted Voting'. This method enables each model to 'vote' as to whether or not they predict the customer to churn. With confidence-weighted voting, each vote is weighted based on the confidence value for that prediction. In a situation where one model predicts a customer to be a churner and the other two models predict them to be remain a current customer, the model that predicts 'churn' will win if its confidence value is greater than other the two predictions combined. Here the Gains chart only shows a single curve representing the ensemble model. In this case, it indicates a good performance for predicting churn compared to the random classifier model line (the diagonal). To continue:

Run the Analysis node

Figure 9.47 shows the Analysis node output.

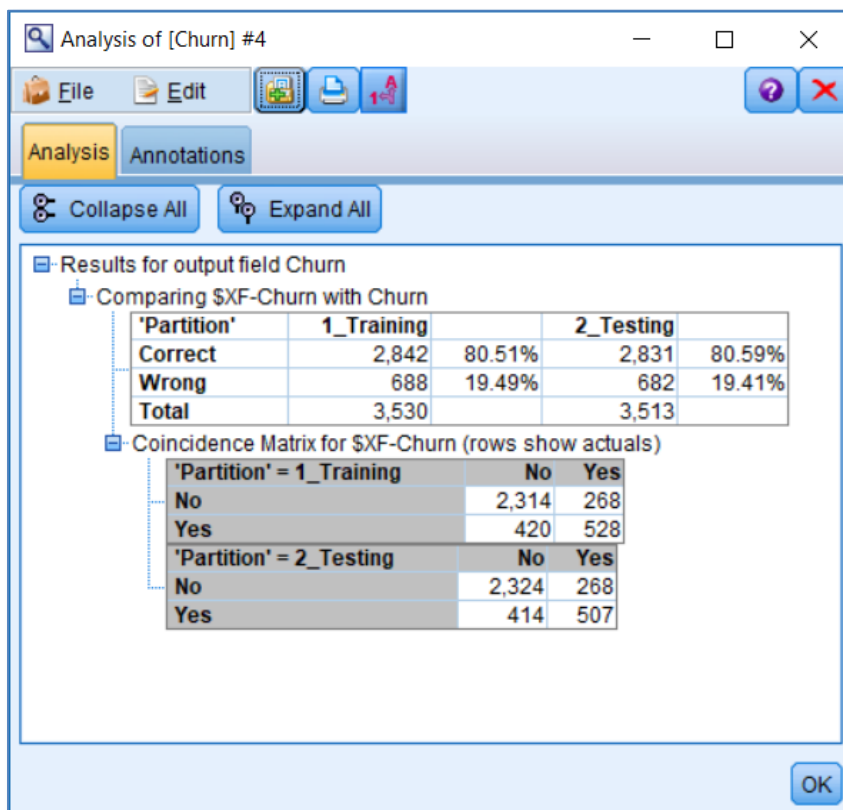


Figure 9.48 Analysis node output showing the overall accuracy for the Auto Classifier ensemble model and coincidence matrices

The ensemble model performs well, with the majority of the churners in the testing set correctly classified. This time we can actually affect the model Evaluation and Analysis node results by selecting or deselecting models within the Auto Classifier nugget. To illustrate:

Double-click the Auto Classifier nugget

Uncheck the last model type in the 'Use?' column (the Neural Network)

Figure 9.49 shows this window after the model is de-selected

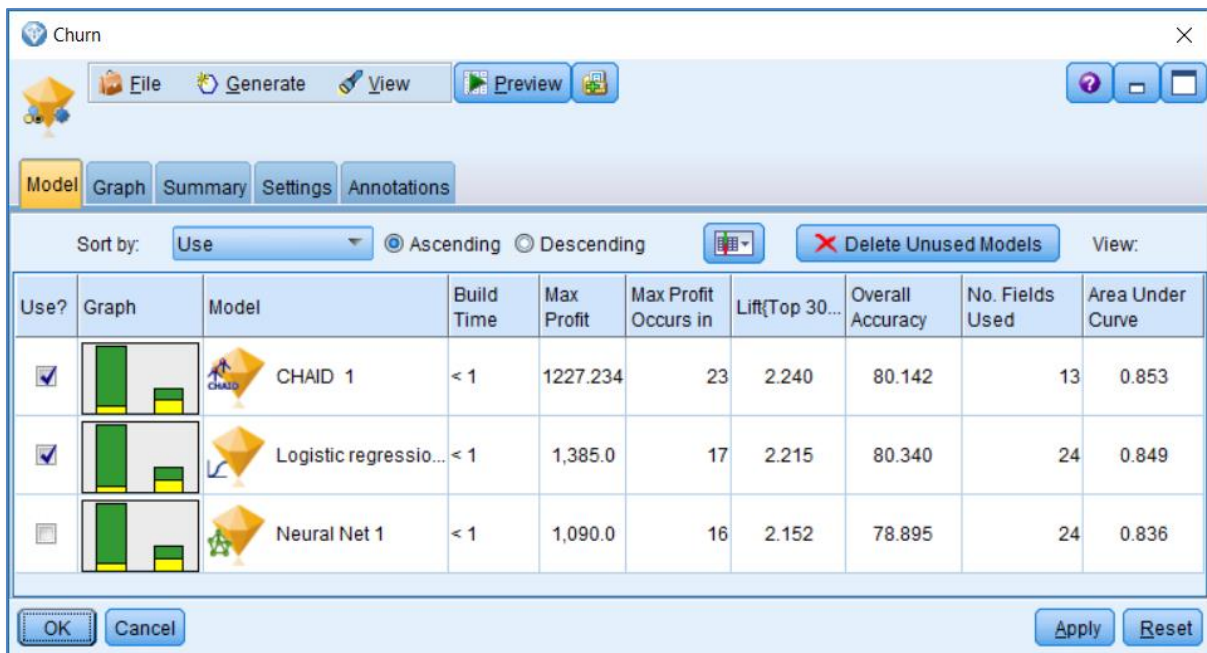


Figure 9.49 De-Selecting the Neural Network model type in the Auto Classifier nugget

Click OK and re-run the Analysis node

Figure 9.50 shows the results.

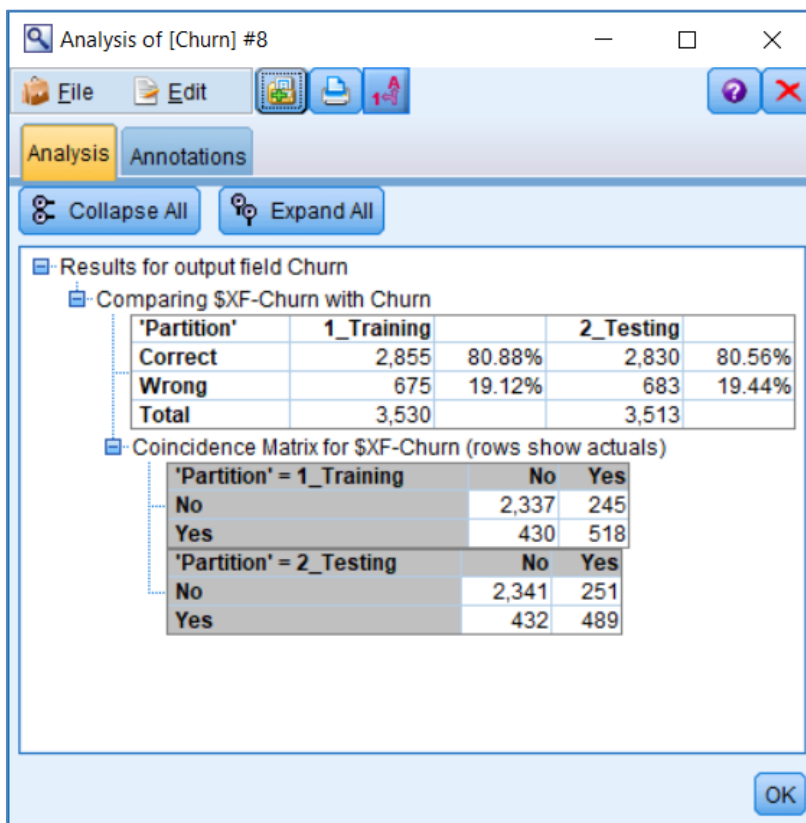
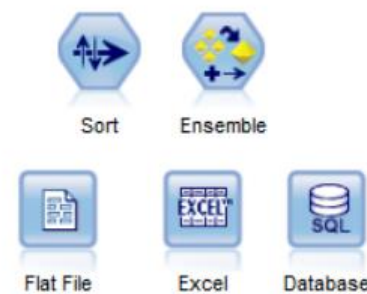


Figure 9.50 The Analysis node results based on the Logistic Regression and CHAID models in the Auto Classifier nugget

The results are quite interesting as they illustrate one of the reasons why ensemble models can be powerful. Even though the Neural Network model was the weakest classification model, by removing it, the ensemble model is less accurate at predicting the outcomes in the testing set. Perhaps the model was more accurate than the other two model types in a particular sub-set of cases and because of this the confidence-weighted voting was enough to ensure it made a positive contribution to the accuracy of the ensemble model.

In the next section we will turn our attention to the Deployment phase of the CRISP-DM process by looking at how we can use these models to score data about current customers.

Section 10: Deploying Models



- Scoring Data with a Model Nugget
- The Sort Node
- The Flat File Export Node
- The Ensemble Node
- The Excel Export Node
- Scoring Data based on Profit
- The Database Export Node

The final phase of the CRISP-DM process is 'Deployment'. This is the point at which the predictive analytics application is used to drive better decision making in the real world. The deployment itself could take the form of something as simple as a report or presentation outlining the insights uncovered by the process. However, creating reports is not really the point of CRISP-DM. Rather, the Deployment stage implies that the outputs from the application will be acted upon: the values that a predictive model generates will be used to actively target opportunities or mitigate threats; the recommendations from an association model will be used to present customers with new offers or employees with suggested actions; anomaly detection models will alert security systems to unusual activity; segmentation models will drive more customised content to websites. Furthermore, often the Deployment phase is the point at which the CRISP-DM process starts to encroach on the activities of other roles within the organisation. Because deployment can take the form of regular updates of a customer database or website, the IT department may need to be involved in overseeing this process. Alternatively, the project may be focussed on identifying customers for inclusion in a marketing campaign, so key personnel and processes within the marketing department will be affected. Whether the project deployment takes the form of generating real time recommendations in a call centre environment or creating maintenance schedules to inspect 'at risk' assets, it is of critical importance that it meets the original objectives as outlined in the Business Understanding phase. Once again, by the time the analysts reach this phase they should already have a detailed plan as to *how* the deployment will be executed, *who* will be impacted and *what* constitutes success.

Scoring Data with a Model Nugget

The process of applying a predictive model to a dataset to generate predictions is referred to as 'Scoring' the data. In the next example, we will use a model nugget to score data about current customers so that we can create a file containing customer IDs whose likelihood to churn is above a certain specific probability threshold. To illustrate this, from the Section 10 folder open the stream:

Section 10 Propensity Selection.str

Figure 10.1 shows the upper section of the stream.

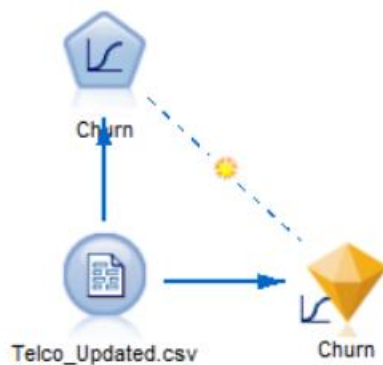


Figure 10.1 The upper section of the stream 'Section 10 Propensity Selection.str'

The stream contains a model nugget generated by the Logistic Regression node. To view the new data fields that the Logistic Regression model nugget generates:

Double-click the Logistic Regression model nugget to edit it

Click the Preview button to preview the data

Scroll to the last fields in the Preview table

Figure 10.2 shows the newly created prediction variables ('\$L-Churn' and '\$LP-Churn') in the preview table.

Preview from Churn Node (28 fields, 10 records) #3

	e_Monthly_Bill	Very_Loyal	Cashback	Customer_Tier	\$L-Churn	\$LP-Churn
1	103.712	T	31.114	a. Premium	No	0.953
2	76.750	F	0.000	c. Standard	Yes	0.719
3	31.455	F	0.000	c. Standard	No	0.574
4	102.264	T	30.679	a. Premium	No	0.874
5	82.958	F	0.000	c. Standard	No	0.910
6	19.708	T	5.912	c. Standard	No	0.998
7	61.917	F	0.000	c. Standard	No	0.728
8	78.684	F	0.000	c. Standard	No	0.506
9	102.776	T	30.833	a. Premium	No	0.831
10	69.079	F	0.000	b. Select	No	0.700

OK

Figure 10.2 The variables '\$L-Churn' and '\$LP-Churn' in the Preview table

Remember that by default, predictive models generate scores that reflect the *confidence* of the prediction. In figure 10.2 we can see that the first prediction indicates that the customer belongs to the 'No' group in the Churn variable (i.e. they are a current customer). The confidence score for this prediction is 0.953. Indicating that according the model, there is a 95% chance that the person falls into this group. The next case indicates that the model predicts this customer to have churned with a confidence value of 0.719 (i.e. a 71.9% likelihood). The reason this is important, is that the confidence value only helps us to select the records where the model is most or least *confident* in its predictions *not* where the cases themselves *are more or less likely* to be churners. To select those with a higher than average likelihood to churn, we must request *Propensity* scores. To do so:

Click OK to close the Preview table

In the edited Logistic Regression model nugget, click the 'Settings' tab

Check the box marked 'Calculate raw propensity scores'

Figure 10.3 shows the updated Settings tab in the Logistic model nugget.

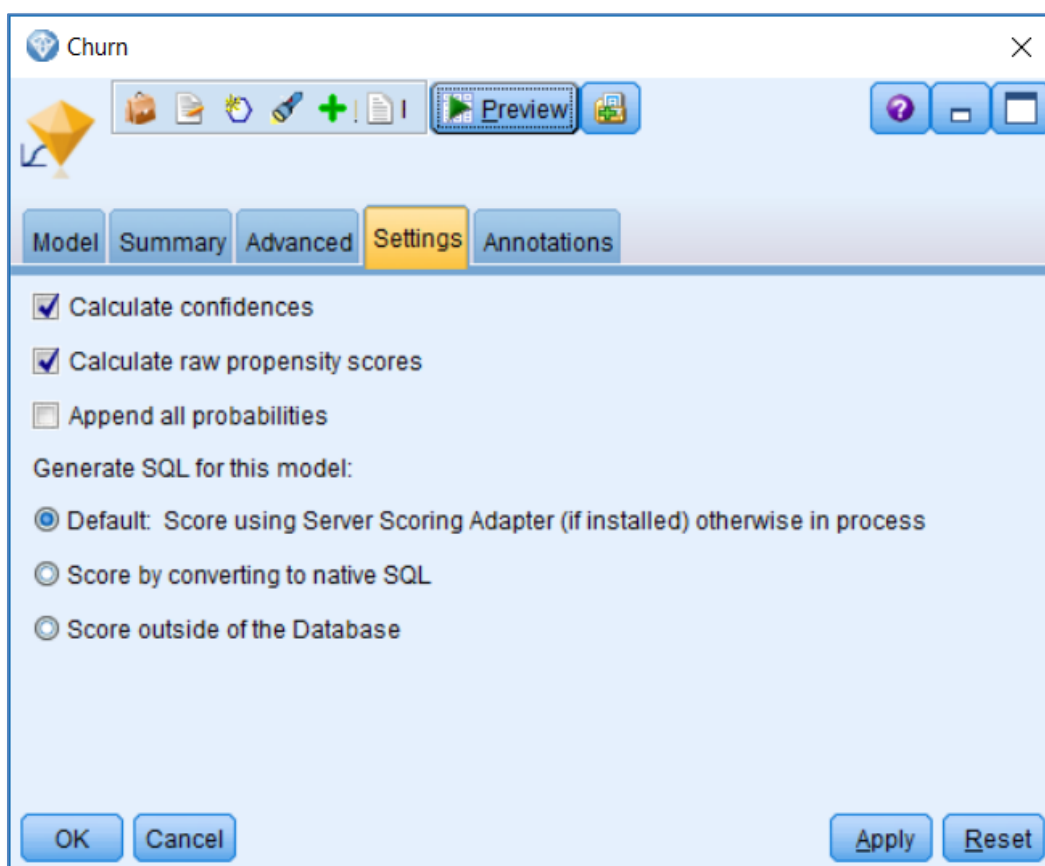


Figure 10.3 Requesting Propensity scores in the Settings tab of the edited Logistic Regression model nugget

Propensity values are akin to the *Probability scores* that traditional statistical models create. The convention is that the scores show the likelihood of being in the key outcome category of

a dichotomous (flag) target variable. In Modeler, this tends to be the 'higher' value numerically or alphabetically speaking. However, values such as '1', 'T' or 'Yes' are automatically assumed to be the category of interest. So here the Propensity value shows, for each customer, the probability that they have churned. To preview these scores, within the currently open dialog, once again click:

Preview

Scroll to the last few fields in the Preview table

Figure 10.4 shows the updated Preview table with the additional Propensity scores in the variable '\$LRP-Churn'

	Cashback	Customer_Tier	\$L-Churn	\$LP-Churn	\$LRP-Churn
1	31.114	a. Premium	No	0.953	0.047
2	0.000	c. Standard	Yes	0.719	0.719
3	0.000	c. Standard	No	0.574	0.426
4	30.679	a. Premium	No	0.874	0.126
5	0.000	c. Standard	No	0.910	0.090
6	5.912	c. Standard	No	0.998	0.002
7	0.000	c. Standard	No	0.728	0.272
8	0.000	c. Standard	No	0.506	0.494
9	30.833	a. Premium	No	0.831	0.169
10	0.000	b. Select	No	0.700	0.300

Figure 10.4 The Confidence and Propensity scores in the variables '\$LP-Churn' and '\$LRP-Churn' respectively, as displayed in the Preview table

You can now see that the first record is predicted to be a current customer, as denoted by the 'No' value under the variable header '\$L-Churn'. Furthermore, as the variable '\$LP-Churn' shows, the confidence of this prediction is 0.953 (95.3%). So this first case has only a 4.7% likelihood of belonging to the churned category. Close the currently open windows by clicking:

OK

OK

We can now use the generated Propensity value to help choose current customers with a higher than average likelihood of churning so that the Telco can take proactive measures to retain them. From the Graphs tab:

Select the Histogram node and attach it to the existing Logistic Regression model

Figure 10.5 shows the updated stream.

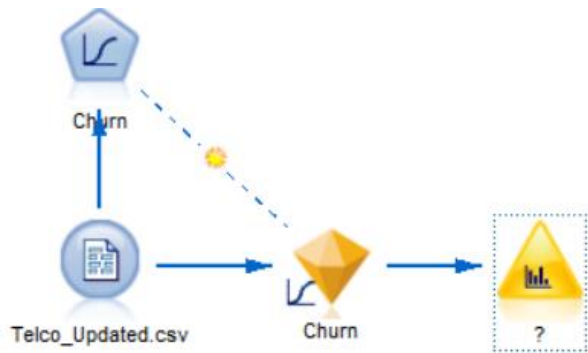


Figure 10.5 Attaching a Histogram node to the Logistic Regression nugget

Double-click the Histogram nugget to edit it

Using the drop-down field selection menu next to the label 'Field', choose:

\$LRP-Churn

Using the drop-down field selection menu next to the label 'Color', choose

Churn

Figure 10.6 shows the completed Histogram dialog.

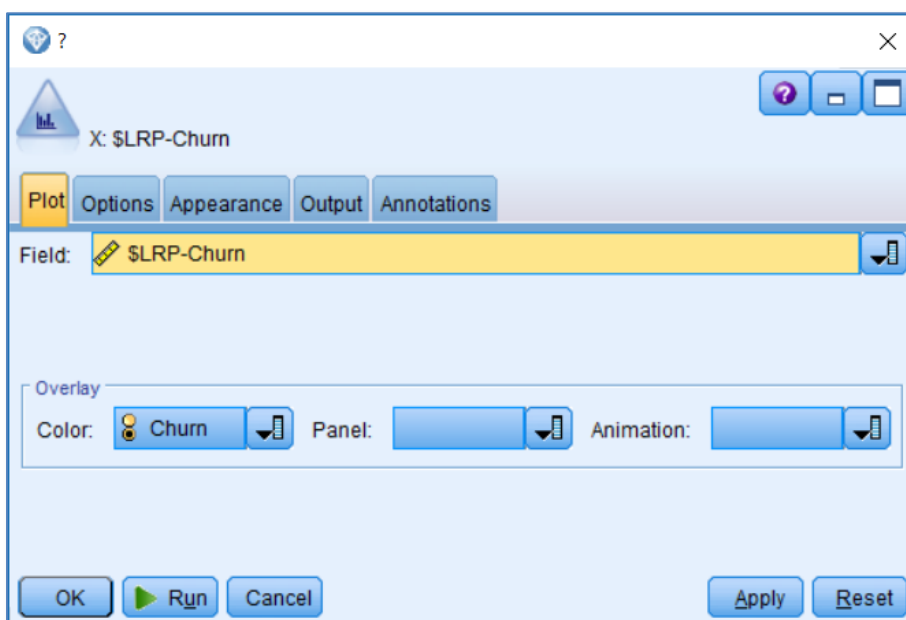


Figure 10.6 Requesting a histogram of *Propensity to Churn* coloured by the actual outcome field: Churn

The histogram will enable us to visualise the model's estimation of propensity to churn as a *distribution*. So not only will we be able to see how evenly (or unevenly) the model estimates

the risk of churn across the sample dataset. But by adding the Churn field as a colour category, we will be able to see what proportion of people actually *did* in fact churn when the model indicated there was a low likelihood and what proportion remained current customers when the model estimated that they would have a high likelihood of defecting. To generate the histogram, click:

Run

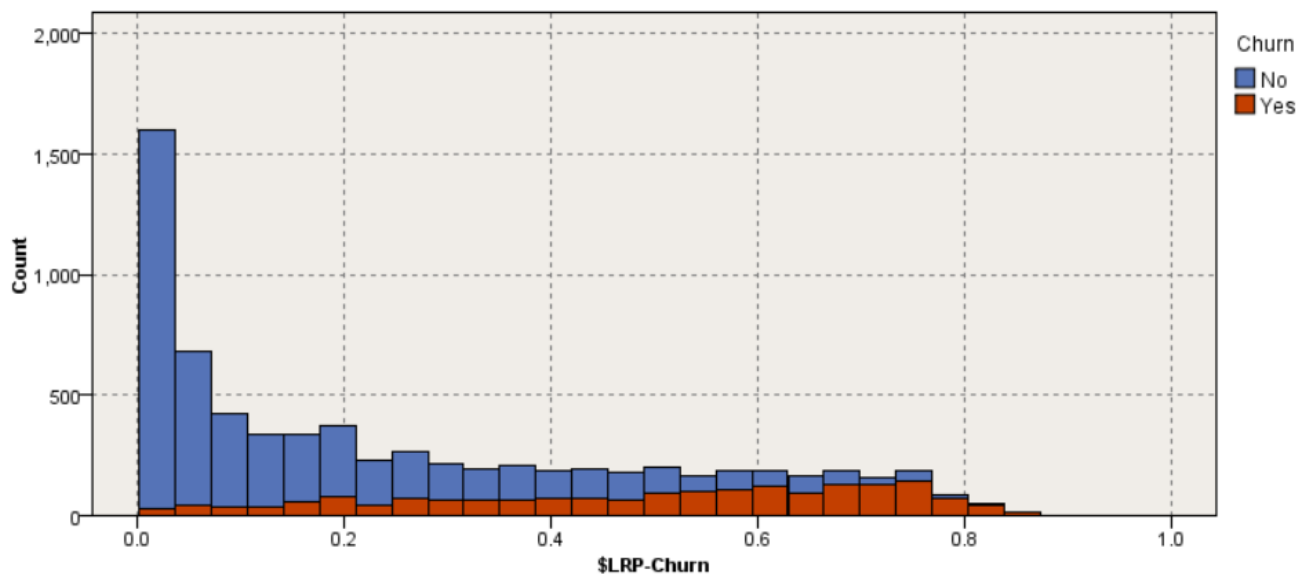


Figure 10.7 Histogram of Churn propensity coloured by the actual outcome field: Churn

The histogram shows that the model estimates most of the customers to have a low likelihood of churn. None of the customers appear to have a churn propensity greater than about 0.85. As we would expect, the higher the propensity to churn, the larger the proportion of those in the 'Yes' group of the churn outcome variable appear in the histogram bins. Nevertheless, a small proportion of *actual* churners are evaluated by the model to be low risk even in the bottom 10% of the propensity distribution. To view the different proportions more effectively:

Copy and paste the existing Histogram node and attach the copy to the Logistic Regression model nugget

Figure 10.8 illustrates this.

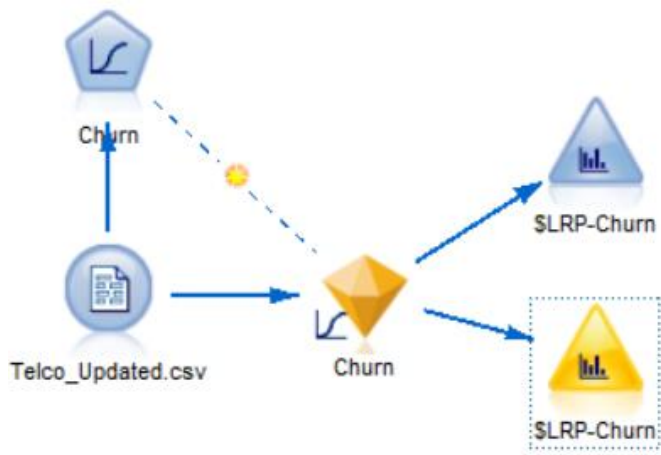


Figure 10.8 The copied and pasted Histogram node attached to the Logistic Regression model nugget

Double-click the new Histogram node to edit it

Click the Options Tab within the edited Histogram node

Check the box marked 'Normalize by color'

Figure 10.9 shows the edited Options tab.

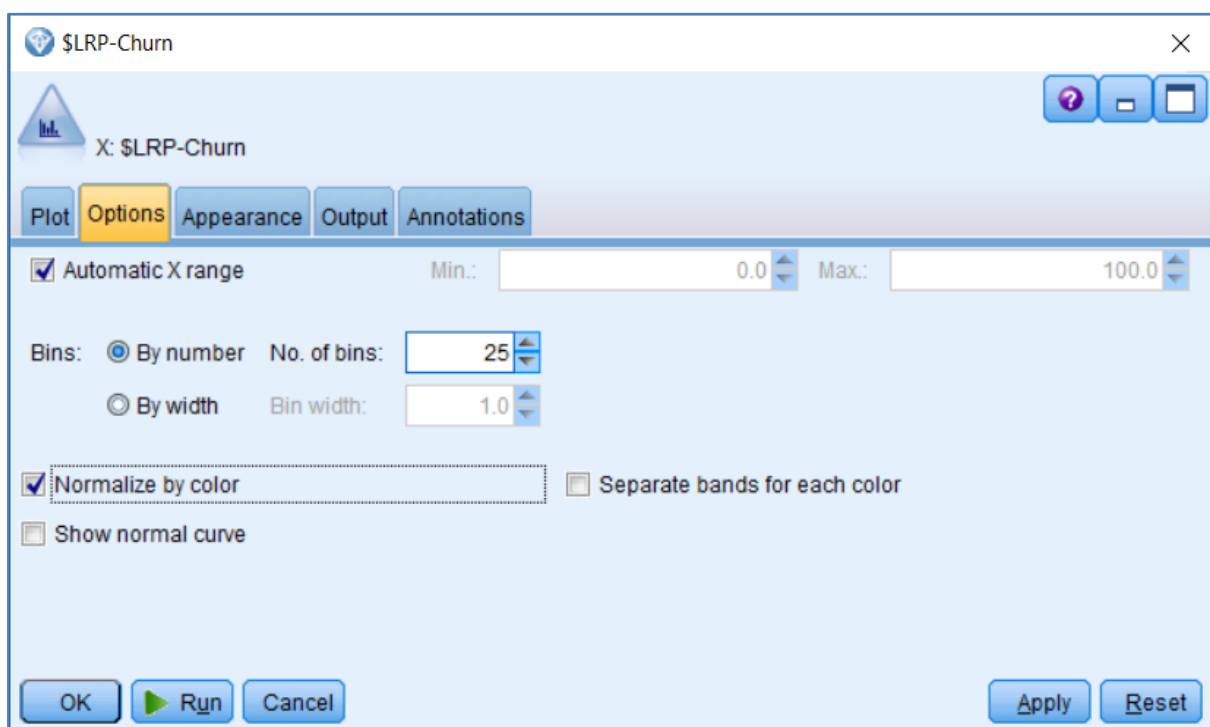


Figure 10.9 Requesting a second histogram – this time normalised by the levels of the colour field

Click:

Run

Figure 10.10 shows the normalised histogram.

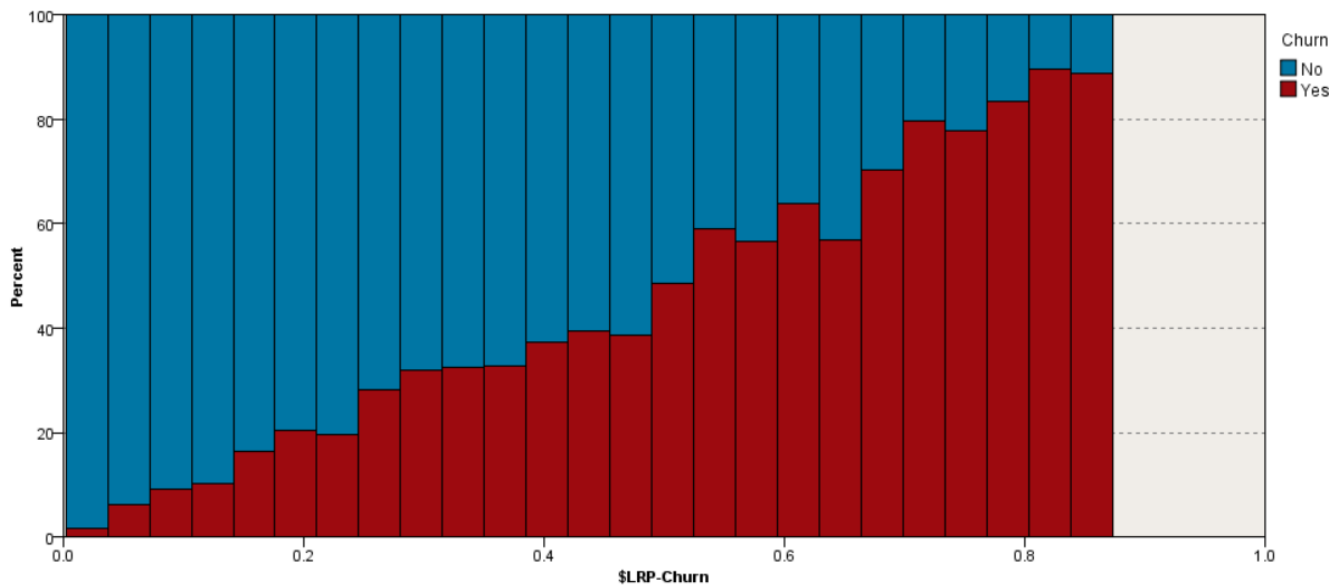


Figure 10.10 'Normalized by color' - Histogram of Churn propensity by the actual outcome field: Churn

The normalised histogram now shows a more or less smooth continuum as the risk of churn rises proportionately with respect to the actual outcome. In this example, we can imagine that the work done in the Business Understanding phase has determined that the goal of the application should be to identify those current customers with a 40 percent or greater risk of churning. Because we have already requested the Propensity score, we can use the histogram to make this selection. To do so, within the Histogram output, from the main menu click:

View

Interactions

Figure 10.11 shows this process in action.

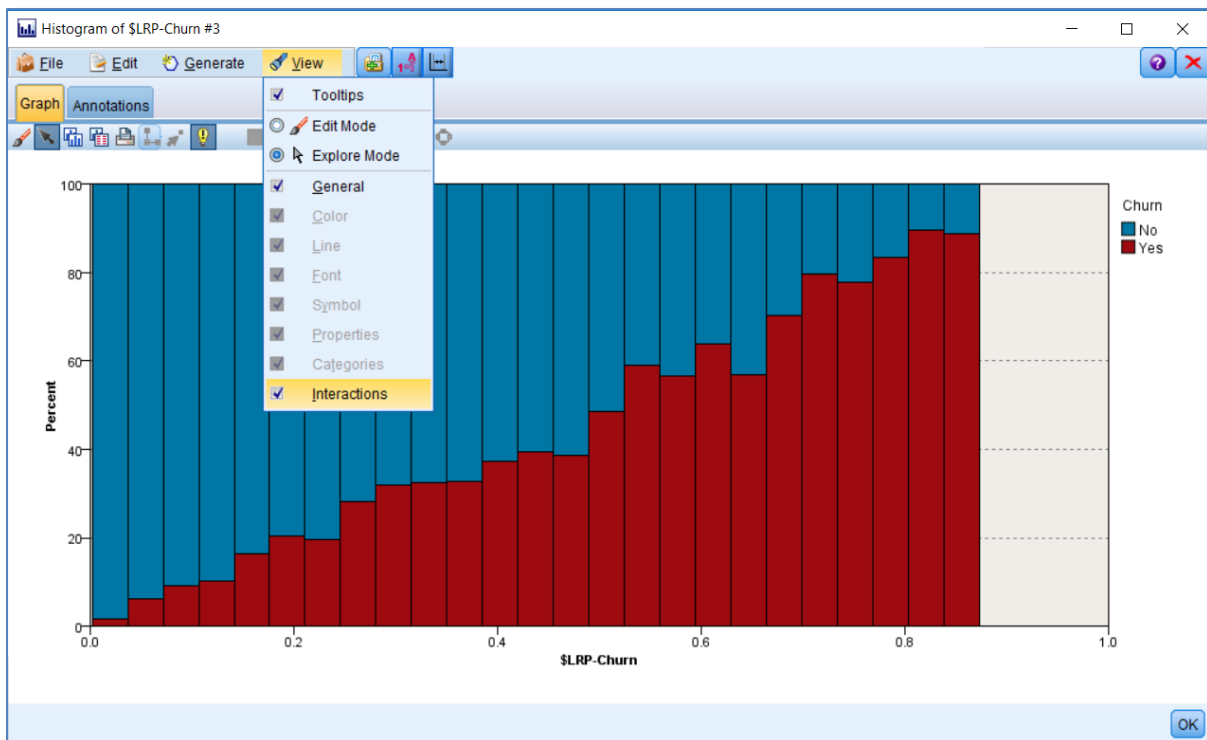


Figure 10.11 Switching to 'Interaction mode' within the Histogram output

As before, by using the Band Selection tool, we can define the threshold at which the propensity risk is above 40 percent (See figure 10.12).

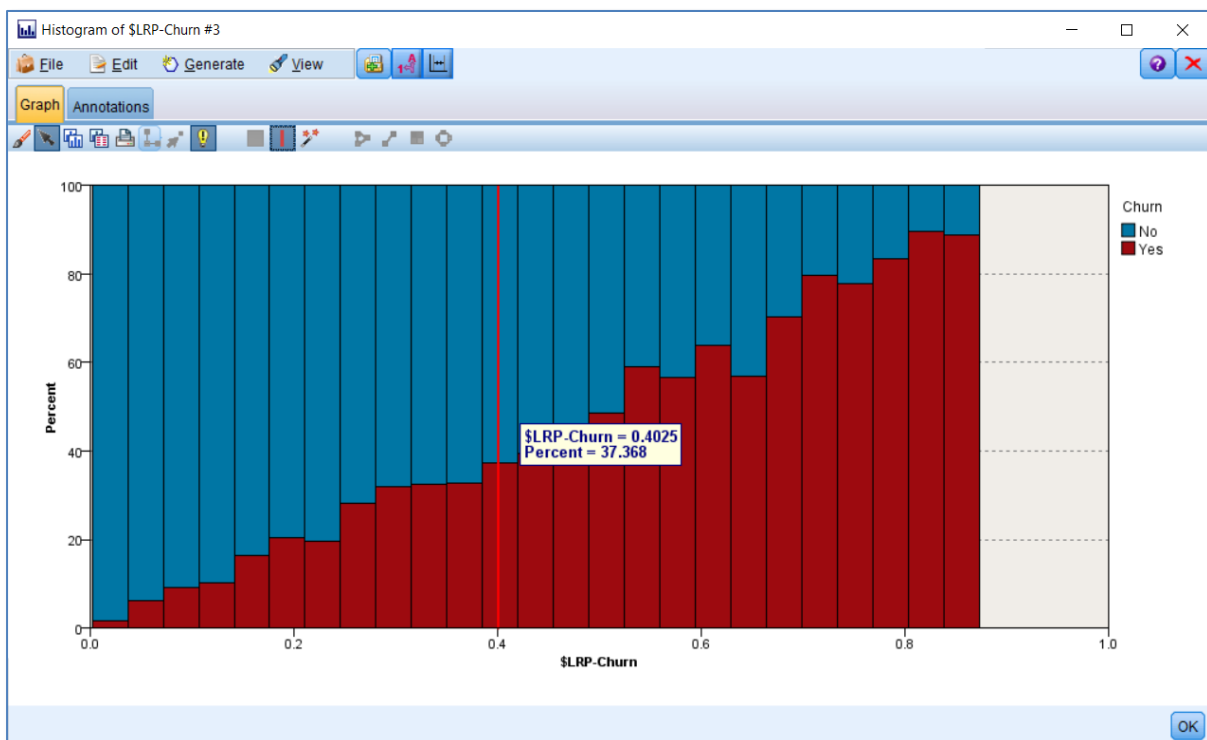


Figure 10.12 Slicing the Histogram at 0.40 Propensity value

Now we can select those cases that lie above this threshold. To do so:

Right-click in the area to the right of the selection line

From the pop-up menu choose:

Generate Select Node for Band

Figure 10.13 shows this process.

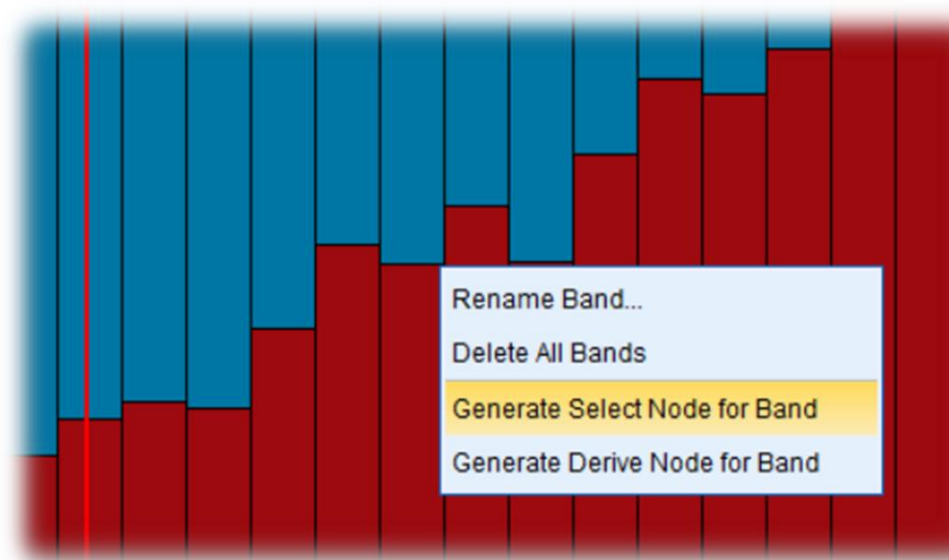


Figure 10.13 Generating a Select node within the Histogram of Propensity values

A Select node is now added to the stream canvas.

Attach the newly generated text node to the Logistic Regression Model nugget

Figure 10.14 illustrates this.

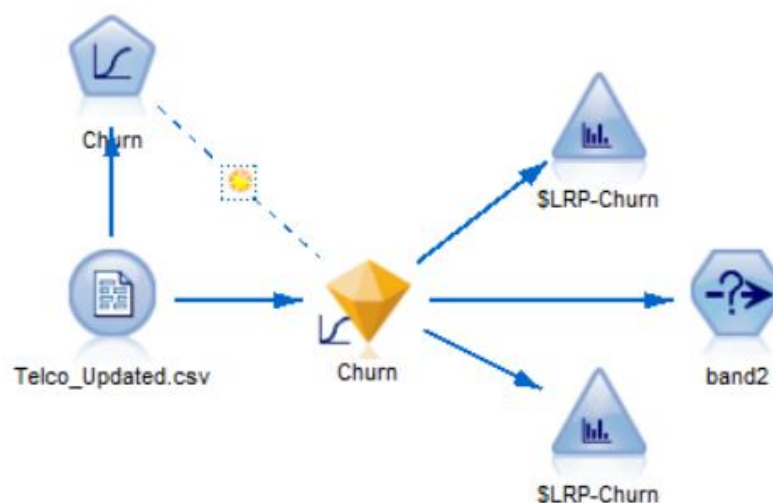


Figure 10.14 Attaching the generated Select node to the model nugget

We can see the edited contents of the Select node in figure 10.15.

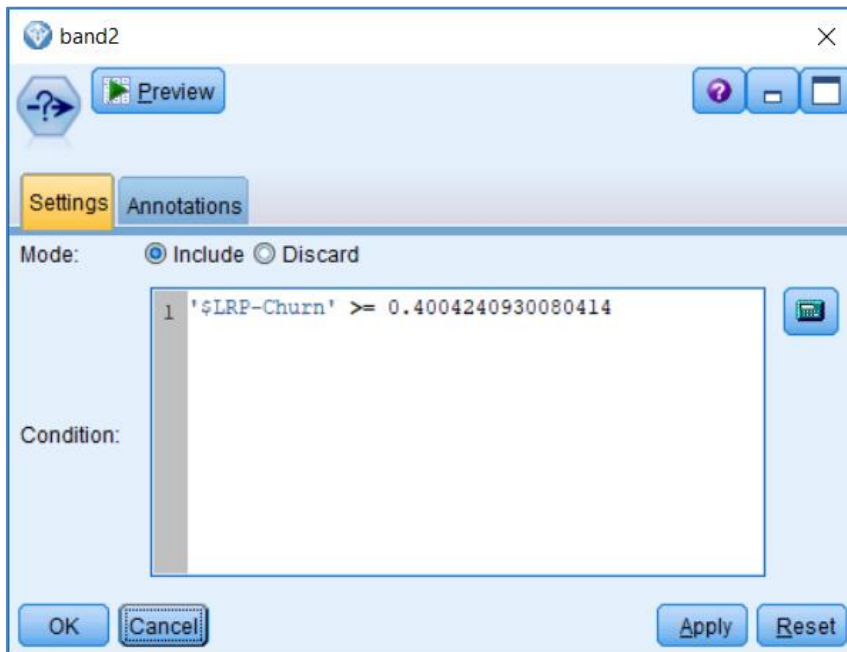


Figure 10.15 The edited contents of the newly generated Select node

Using the rule within the Select node, only 2,084 of the original 7,043 records had propensity scores of around 0.4 or above (about 30%). Having, selected the records with higher than average propensity We can now copy and paste the selection and the model to the scoring data. To do so:

Highlight the model nugget and the select node

Right click and select 'Copy' from the pop-up menu

Figure 10.16 shows this in action.

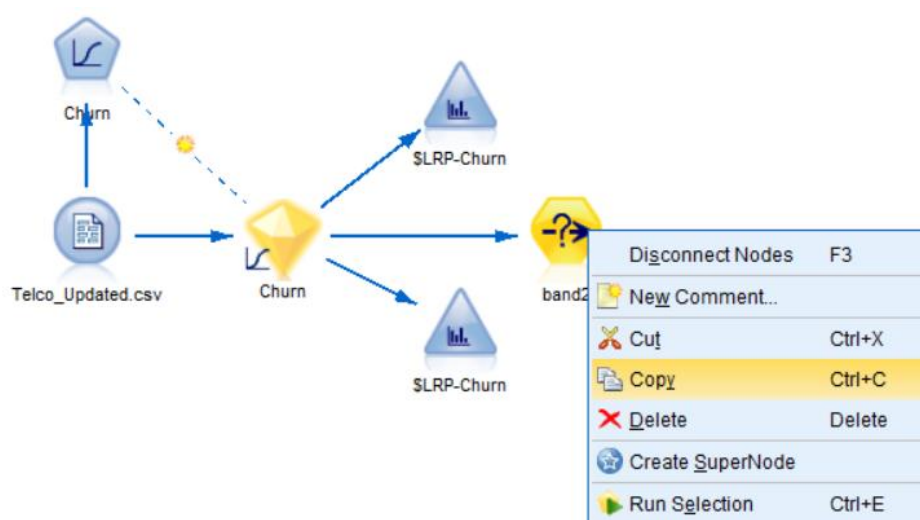


Figure 10.16 Copying the model nugget and select node

Now we can paste the nodes next to the scoring data source.

Scroll down the stream canvas until the Source node for the file 'Scoring Data Simple.csv' is visible

Right-click on the stream canvas and choose 'Paste' from the pop-up menu'

Connect the pasted model nugget to the Data Source node

Right-click on the dashed model link line for the model and choose 'Remove Link'

Figure 10.17 illustrates this.

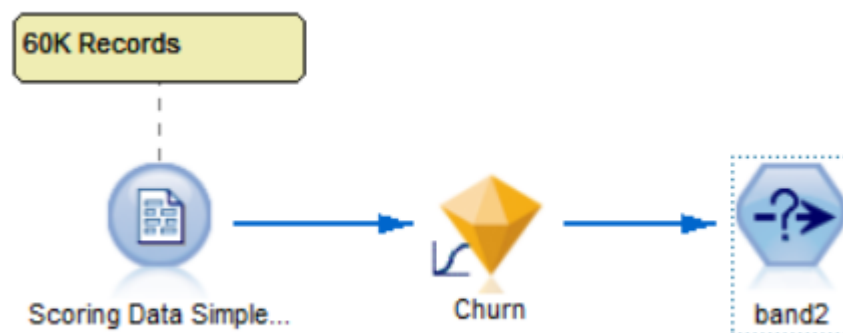


Figure 10.17 Connecting the pasted model nugget and select node to the Data Source node containing the scoring data

As the comment label indicates, the scoring data contains 60,000 records of current customers. Of course, this file will *not* contain a field indicating if the customer has churned because all the records are all still current customers. Once the model has scored the data, the only variables that are needed for any further selection are the customer ID and propensity score. To that end:

Connect a Filter node to the pasted Select node

Edit the Filter node and remove all the fields except for 'customerID' and '\$LRP-Churn'

Rename '\$LRP-Churn' to 'Churn_Likelihood'

Figure 10.18 illustrates this.

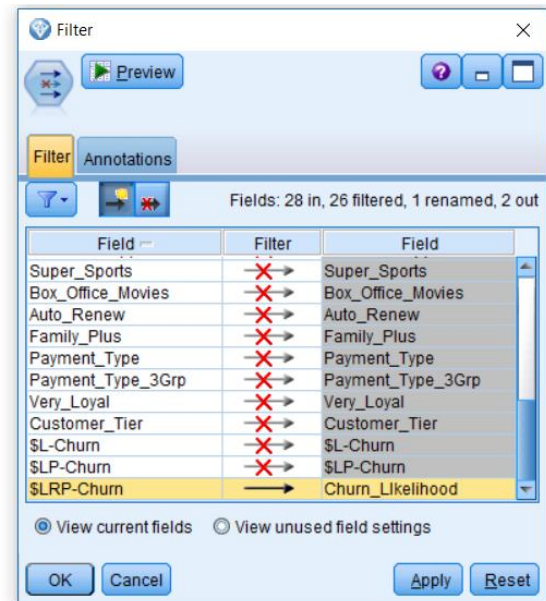


Figure 10.18 Removing unnecessary fields and renaming '\$LRP-Churn' to 'Churn_Likelihood'

To continue, click:

OK

The Sort Node



We can also sort the propensity scores in the 'Churn Likelihood' variable. The Sort node is a simple procedure that allows us to do that. To sort the data, from the Record Ops palette:

Select and attach a Sort node to the Filter node.

Double-click the Sort node to edit it

Click the Field picker button and choose the variable 'Churn_Likelihood'

Change the sort direction from 'Ascending' to 'Descending'

Figure 10.19 shows the completed Sort dialog.

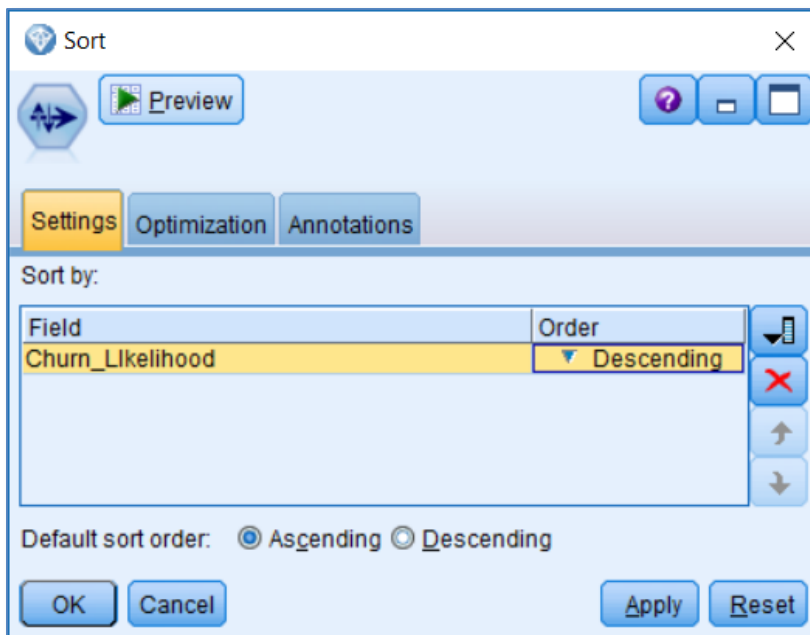


Figure 10.19 Sorting the scored data in descending order by likelihood to churn

Figure 10.20 shows a preview of the sorted dataset.

	customerID	Churn_Likelihood
1	6949.000	0.995
2	3551.000	0.993
3	944.000	0.992
4	2797.000	0.992
5	4630.000	0.991
6	5513.000	0.990
7	2330.000	0.990
8	21.000	0.990
9	3706.000	0.990
10	3748.000	0.990

Figure 10.20 Preview of the scored data sorted in descending order by likelihood to churn

Once again, to continue click:

OK

Figure 10.21 shows the updated scoring branch of the stream at this stage.

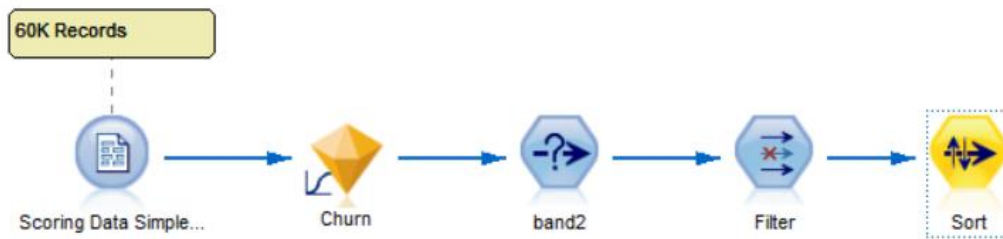


Figure 10.21 The updated scoring branch of the current stream

The Flat File Export Node



Flat File

By now we should be very familiar with reading text-based data via nodes such as the Var. File node. The Flat File output node can therefore be thought of as the mirror equivalent of the Var. File node as it *writes out* data in a delimited text file rather than reading it. We can use this node to generate a tab-delimited file containing the customer IDs and churn likelihood values of the customers with a higher than average propensity to defect. From the Export palette:

Double-click the Flat File node

Edit the Flat File Node

Specify the that exported file should be named 'High Propensity Churners.txt'

You may notice that the dialog contains various options as to the field delimiter in the file. The default character to separate the fields is a comma. For this example, we will create a tab separated file. In the Field Separator area, choose the radio button marked:

Tab

Figure 10.22 shows the completed Flat File export dialog.

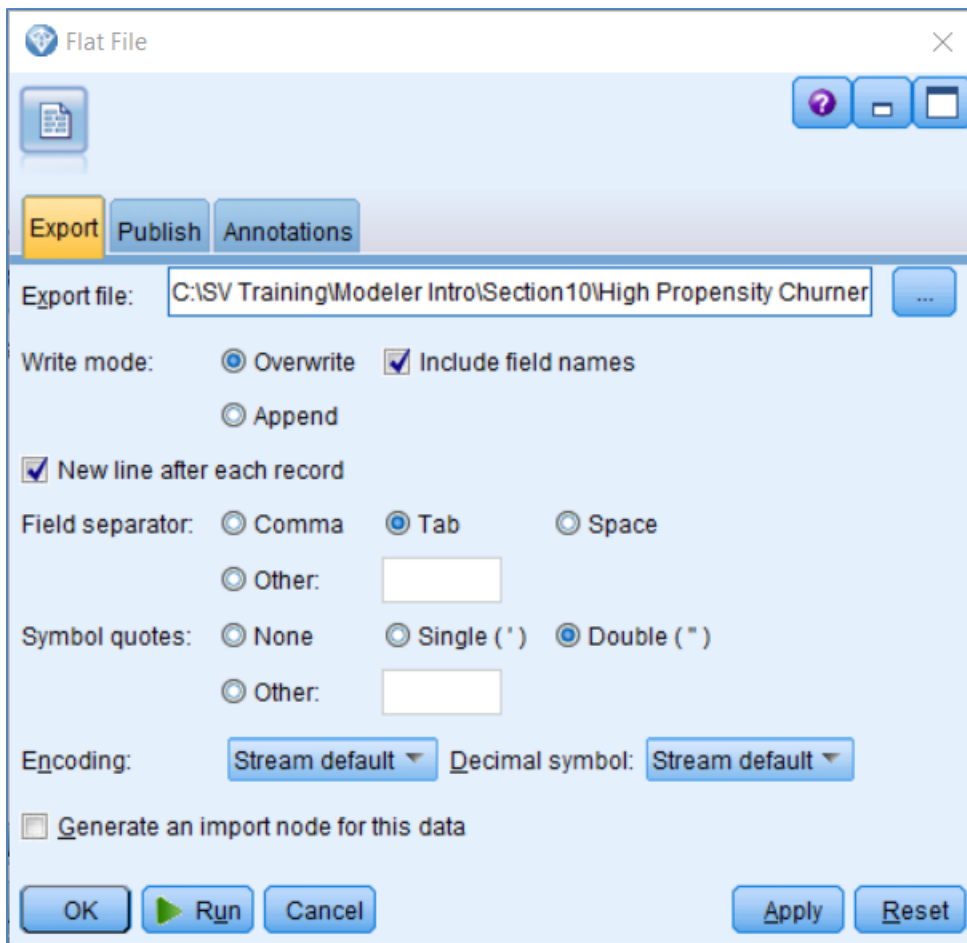
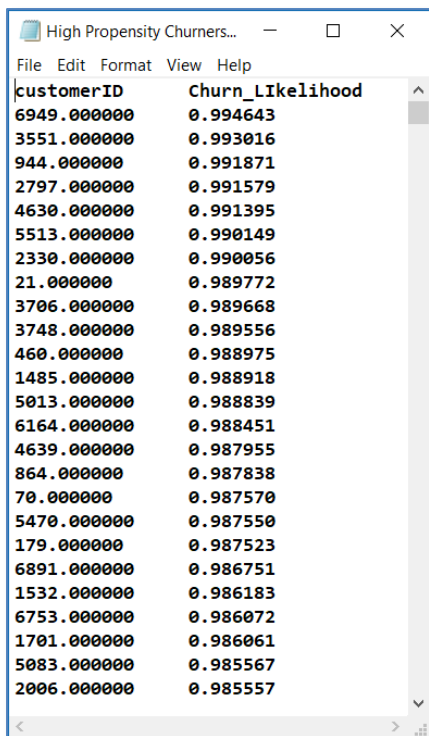


Figure 10.22 The Flat File export node edited to create a tab-separated text file.

To export the scored data, click:

Run

Figure 10.23 shows the exported text file as it appears in MS Notepad.



customerID	Churn_Likelihood
6949.000000	0.994643
3551.000000	0.993016
944.000000	0.991871
2797.000000	0.991579
4630.000000	0.991395
5513.000000	0.990149
2330.000000	0.990056
21.000000	0.989772
3706.000000	0.989668
3748.000000	0.989556
460.000000	0.988975
1485.000000	0.988918
5013.000000	0.988839
6164.000000	0.988451
4639.000000	0.987955
864.000000	0.987838
70.000000	0.987570
5470.000000	0.987550
179.000000	0.987523
6891.000000	0.986751
1532.000000	0.986183
6753.000000	0.986072
1701.000000	0.986061
5083.000000	0.985567
2006.000000	0.985557

Figure 10.23 The exported file 'High Propensity Churners.txt'

The Ensemble Node



Earlier, we introduced the idea of ensemble models by using the Auto Classifier node. In fact, we can combine individual models that utilise the same target fields and data sources to create our own merged model scores as if we were using an ensemble model. To illustrate this, from the Section 10 folder, open the stream:

Section 10 choose best 50 pct.str

Figure 10.24 shows the upper part of this stream.

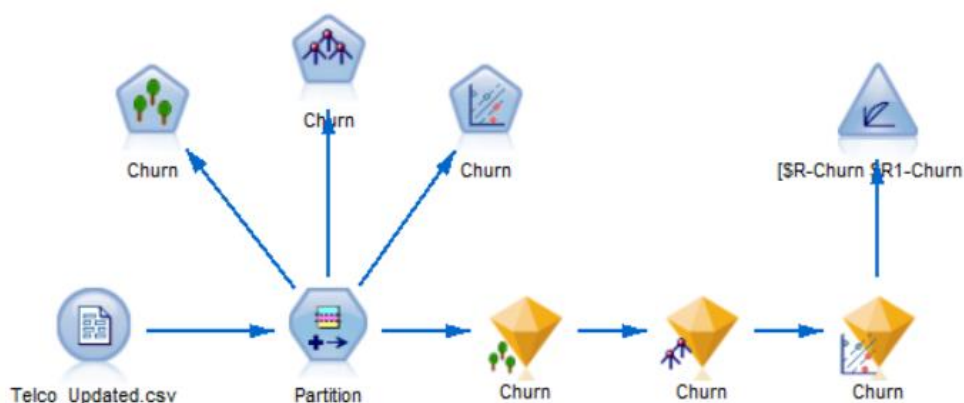


Figure 10.24 Part of the stream 'Section 10 choose best 50 pct.str'

Figure 10 consists of two side-by-side line graphs. Both graphs have a y-axis labeled '% Gain' ranging from 0 to 100 and an x-axis labeled 'Percentile' ranging from 0 to 100. A red diagonal line represents random performance. The left graph is labeled 'Percentile Training' on the x-axis, and the right graph is labeled 'Percentile Testing' on the x-axis. Both graphs show four curves: \$BEST-Churn (dark blue), \$R-Churn (green), \$R1-Churn (orange), and \$L-Churn (light blue). In both graphs, \$BEST-Churn shows the highest performance, reaching 100% gain at a lower percentile than the others. \$R-Churn and \$R1-Churn perform similarly, while \$L-Churn shows slightly lower performance in the testing set.

The evaluation node shows that all three models show similar classification accuracy when predicting those cases that fall into in the churner category (the \$R1-Churn appears to perform best of all). In this example, the application goal, as determined during the Business Understanding phase, is to choose the 50% of customers at the highest level of risk of churning with the aim of identifying at least 14 thousand churners among them. Rather than choosing one model to select this group, we can use the Ensemble node to combine them into a single model score. To do so, from the Field Ops palette:

Copy and paste the existing Evaluation node so that it is attached to the Ensemble node

Deploying Models

Figure 10.27 shows the effect of the Ensemble node when we run the second Evaluation node.

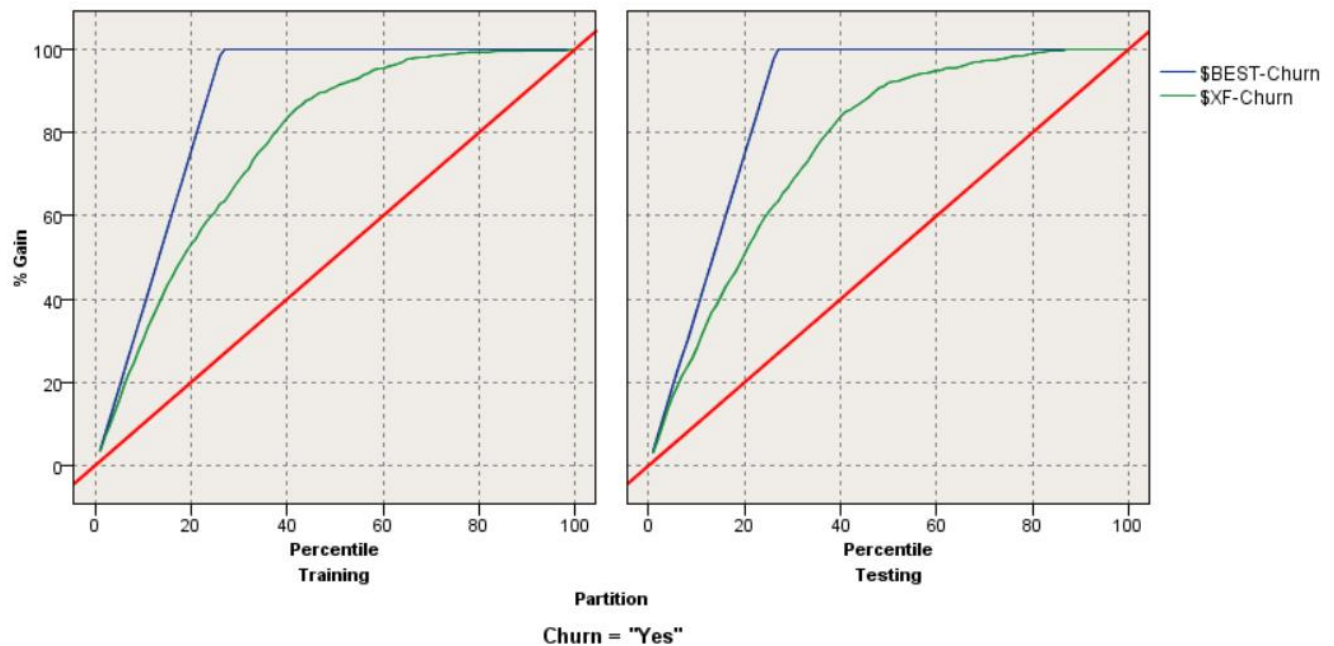


Figure 10.27 Evaluation node showing model scores combined via an Ensemble node to create the field \$XF-Churn

As figure 10.28 shows, by switching on Interactive mode in the chart, we can use the band selection tool to choose the top 50% of cases with the highest risk.

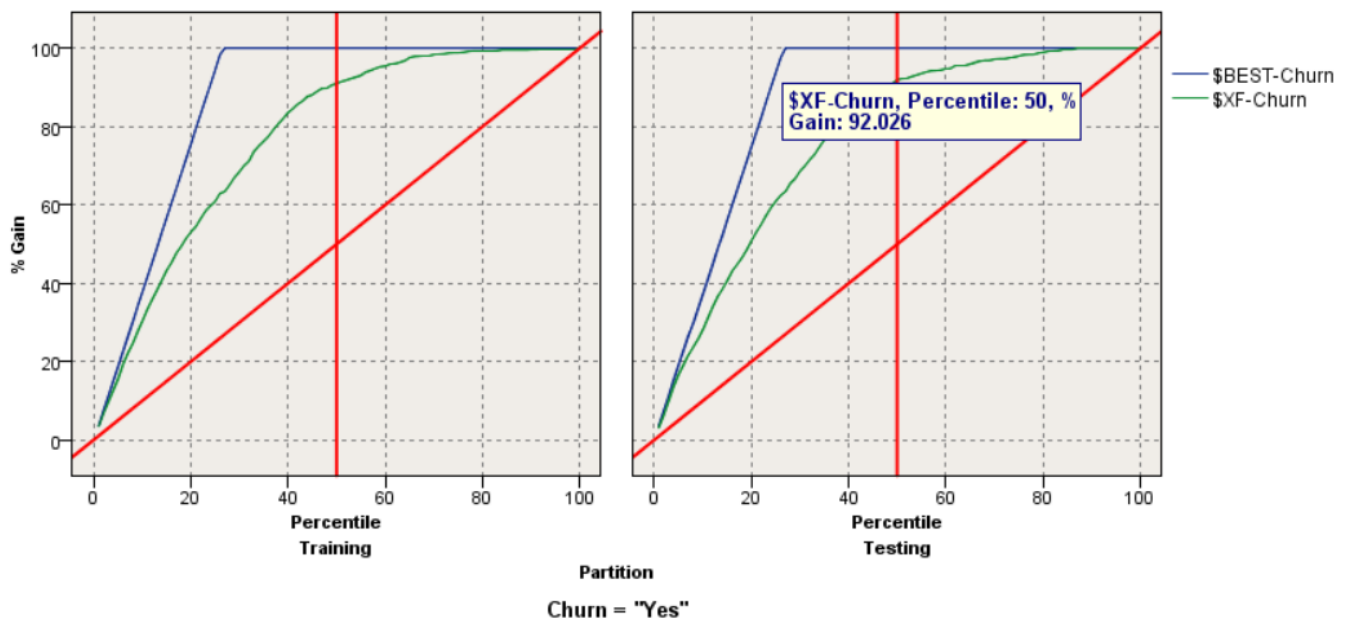


Figure 10.28 Using the band selection tool to choose the 50% of cases of with highest risk of churning.

Figure 10.29 shows how we can generate a Select node by right-clicking on the left-hand side of the selection line.

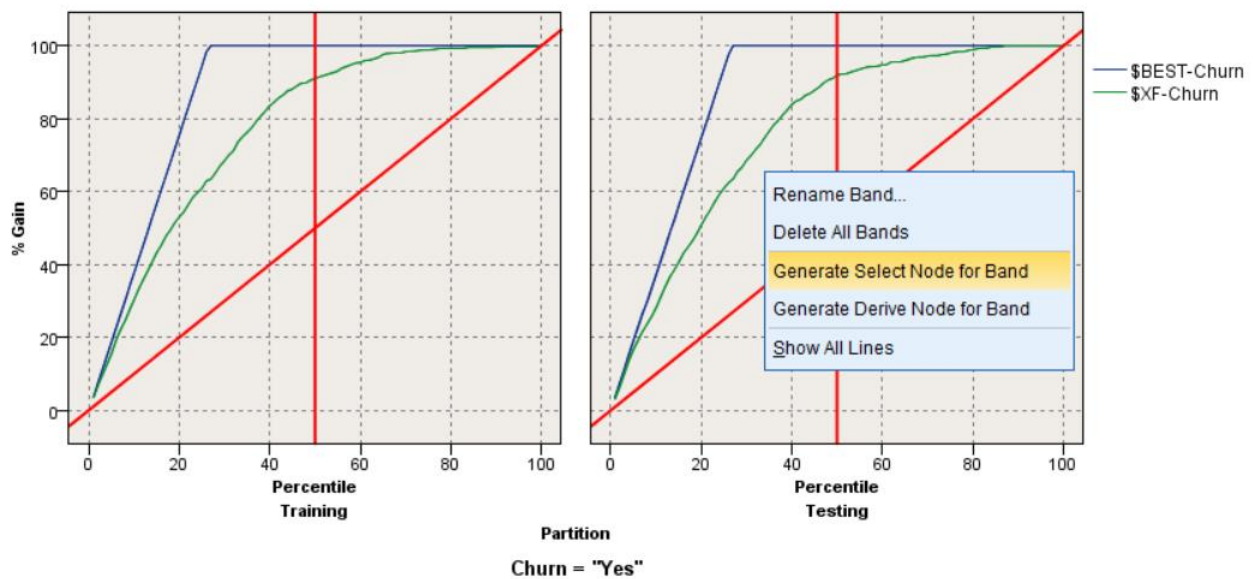


Figure 10.29 Generating a Select node to select the 50% of cases with the highest risk of churning

Just as in the previous example, we can copy and paste the models and select node to score a dataset containing only current customers to test the success criteria as identified during the Business Understanding phase. In this case, at the same time:

Select and copy the three model nuggets and the Ensemble node

Scroll down the stream canvas and paste the model nuggets and Ensemble node next to the Data source node containing the file 'Scoring for Top 50 pct.txt'

Attach the pasted nodes to the Data source node

Do the same for the Select node so that the stream looks like Figure 10.30 below



Figure 10.30 Scoring data with an Ensemble of models and a generated Select node

As the comment attached to the source node indicates, the scoring data is comprised of 100 thousand records and we wish to detect at least 14 thousand customers who will churn in the top 50%. To see if this is the case, we can attach a distribution node to the Select node and create a chart of the Ensemble node's combined predicted outcome field: 'XF-Churn'. Figure 10.31 shows this chart.

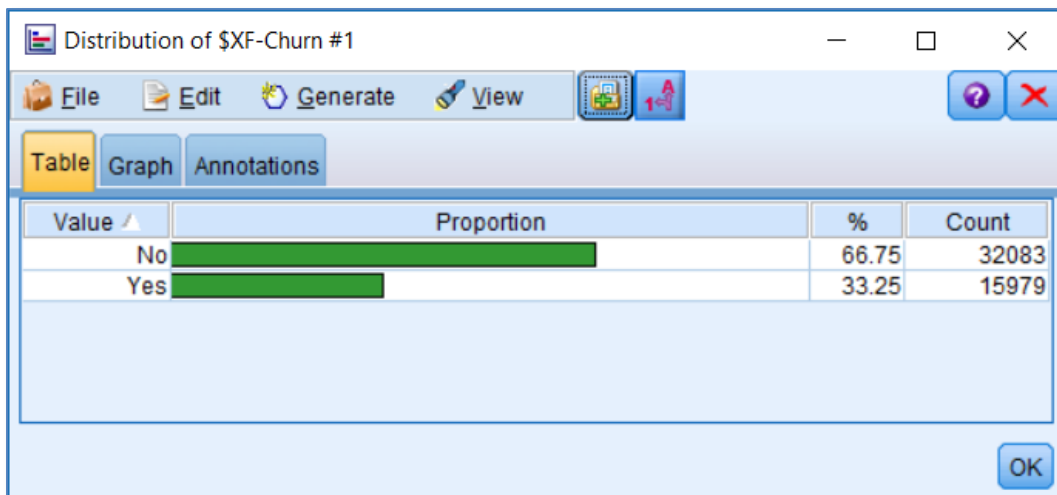


Figure 10.31 Predicted outcomes from three models combined by an Ensemble node

The chart shows that in fact just under 16 thousand customers are predicted to churn from a selection of around 48 thousand records. Taken at face value, the model has met the success criteria as set out in the original Business Understanding. It is now up to the company to take the appropriate proactive measures to persuade as many of these people as possible to remain customers.

Once again, we can attach a Filter node to the end of the stream and drop any unnecessary fields. In this case, the selection was based on the Churn *confidence* not the *propensity* value (so a score of 0.39 could indicate a 39% confidence that the person will *remain* a customer). If we wanted to convert it to a score indicating likelihood to defect, we could of course use a Derive or Filler node here to generate the propensity value by simply subtracting all the confidence scores for those predicted to fall into the 'No' category from a value of 1. However, we can assume that all that is required, is we provide the Customer Retention team with the ID values of those who fall into the highest 50% risk group. The added Filter node shown in figure 10.32 simply drops all the fields except for 'customerID'.

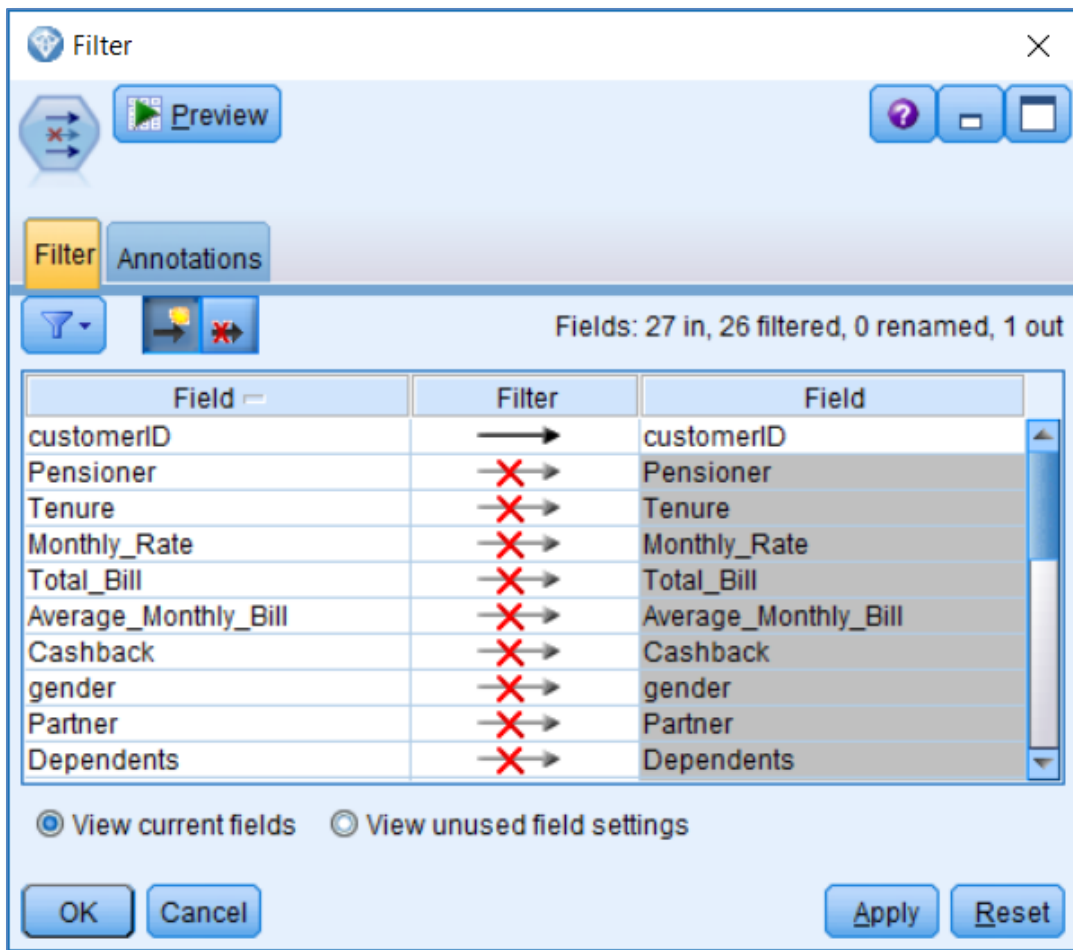


Figure 10.32 Filtering out all fields except for 'customerID'

Figure 10.33 shows the updated stream at this stage.



Figure 10.33 The updated scoring stream with a Filter node added

Finally, because we intend to export the data into a third-party file format (Excel) we must also attach a Type node and fully instantiate the customer ID field. Figure 10.34 shows the updated stream.



Figure 10.34 The updated scoring stream with a Type node added

Figure 10.35 shows the fully instantiated customer id field in the Type node.

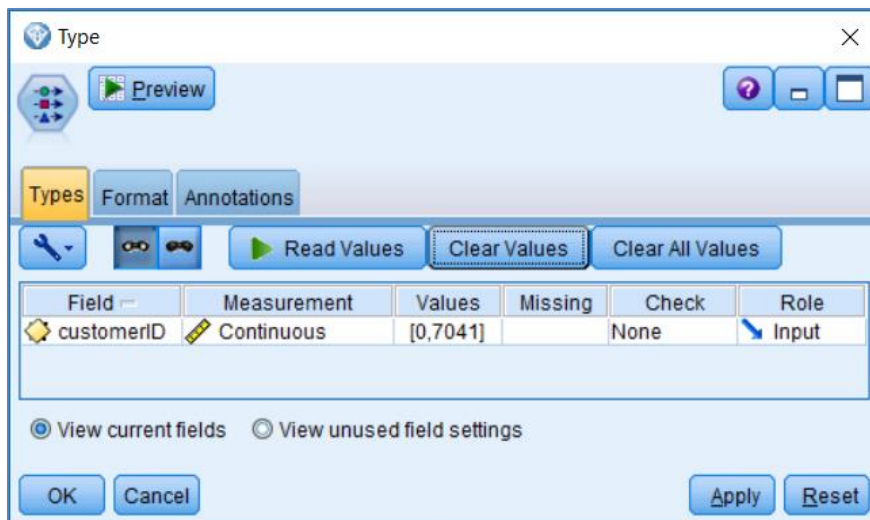


Figure 10.35 The fully instantiated field 'CustomerID' in the Type node

The scoring branch of the stream is now quite long. We can of course now simplify the stream's appearance by using a Supernode to encapsulate the model nuggets and nodes from the Field Ops palette. Figure 10.36 shows this with an annotation attached to the resultant Supernode.



Figure 10.36 The updated scoring stream simplified with a Supernode

The Excel Export Node



Finally, we can export the scored data in this example to a single file. In this example, from the Export palette:

Choose an Excel file export node and attach it to the Supernode

Figure 10.37 shows the updated stream.



Figure 10.37 Exporting the records with high risk customers as an Excel file

Double-click the Excel export node

Edit the file destination box so that the node creates a file called 'High_Risk_Customers.xlsx' in the Section 10 folder

Check the box marked 'Launch Excel™'

Figure 10.38 shows the edited Excel export node.

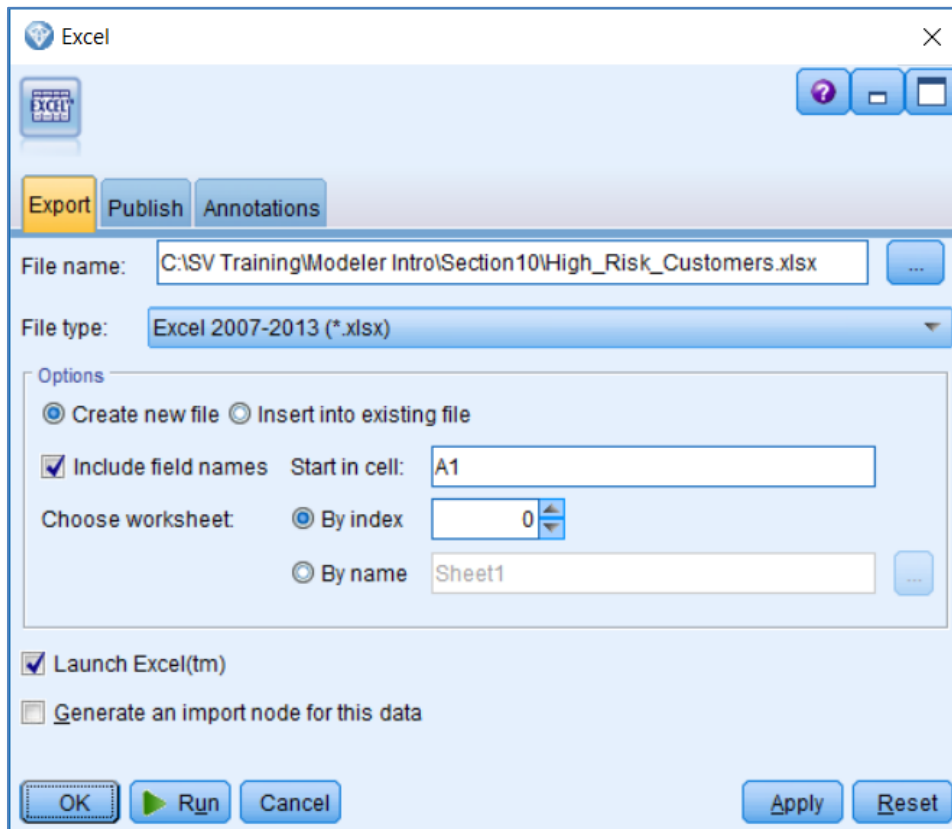


Figure 10.38 The Excel export node configured to generate the file 'High_Risk_Customers.xlsx'

Run the Excel export node

The customer data is now scored, and an Excel file of high risk customers has been created. Figure 10.39 shows the file in the Excel application that the node launches.

	customerID
1	customerID
2	6523
3	2667
4	2871
5	3254
6	667
7	1257
8	5687
9	2215
10	5669
11	4141
12	831
13	6857
14	3575
15	5553
16	4690
17	637
18	4211

Figure 10.38 The customer IDs with a high risk of defecting displayed in the launched Excel application

Scoring Data based on Profit

In our final example we will return to using the Evaluation node in combination with a model to achieve project goals as set out in the Business Understanding phase. In this situation however, we are not attempting to select the top X percent of high-risk cases or those above a certain threshold in the model propensity value, rather we are focussed on trying to find the selection of records that are likely to drive the *greatest profit* in a campaign. To begin this process, from the Section 10 folder, open the stream:

Section 10 Choose most profitable model.str

Figure 10.39 shows the upper part of the stream.

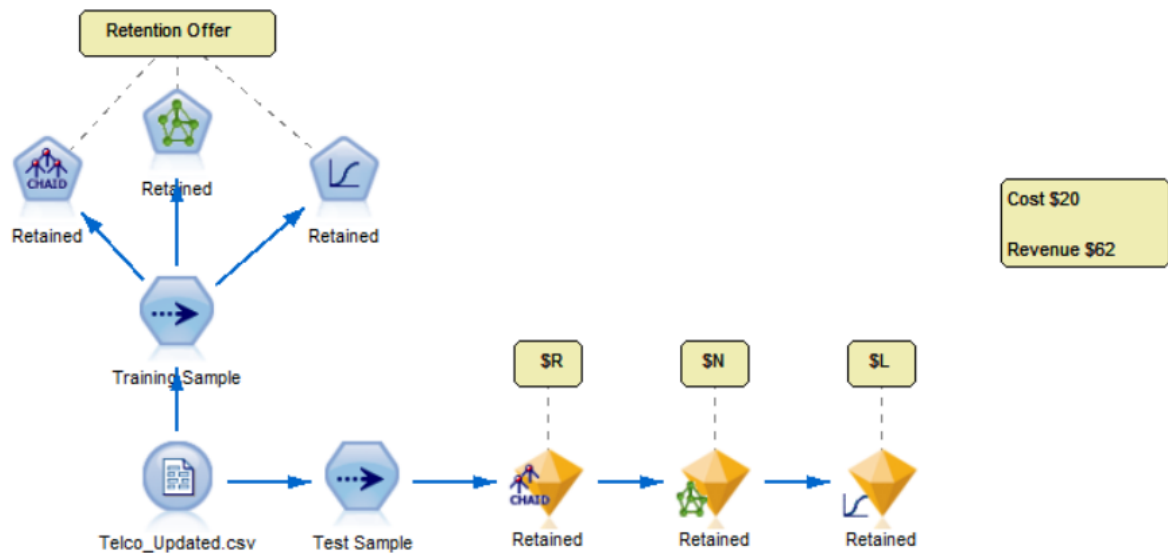


Figure 10.39 The stream file 'Section 10 Choose most profitable model.str'

As we can see, the stream contains three separate model nuggets that are predicting the categories in the target field 'Retained'. In this example, we can imagine that the Business Understanding phase has determined that the goal of the application should be to make customer retention campaigns *profitable*. The context here is that costly retention offers are being made to customers with a high risk of churning. The offers take the form of extending their contracts by two months but with free telephony charges. Many of the those who take the offer however churn after the two months anyway leading to a net loss of \$140K in each campaign. The telco would like to ensure that each campaign makes a net profit of \$70K. From the Output palette:

Choose the Evaluation node and attach it to the last model nugget in the upper part of the stream.

Figure 10.40 shows the updated stream at this stage.

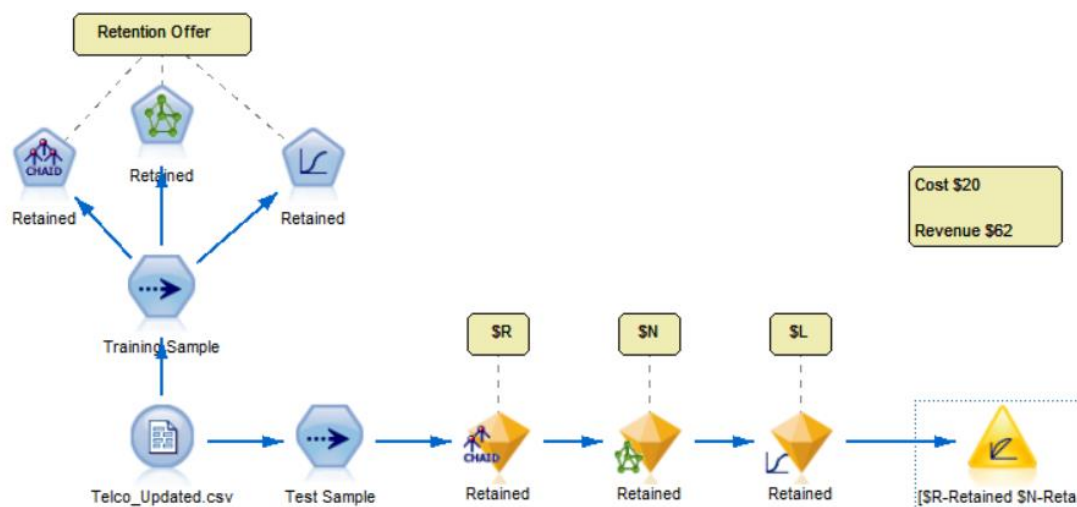


Figure 10.40 The updated stream with an Evaluation node attached

Now:

Double-click to edit the Evaluation node

Click the 'Include best line' option

In the 'Chart Type' drop-down menu:

Change the chart type from 'Gains' to 'Profit'

In the box next to 'Cost':

Edit the fixed value so that it is 20

In the box next to 'Revenue':

Edit the fixed value so that it is 62

Figure 10.41 shows the edited Evaluation node dialog at this stage.

The screenshot shows the 'Evaluation' dialog box for a 'Profit' chart. The 'Chart type' is set to 'Profit'. The 'Options' tab is active, showing checkboxes for 'Cumulative plot' (checked), 'Include baseline' (checked), and 'Include best line' (checked). The 'Use profit criteria for all chart types' checkbox is unchecked. Under 'Models', 'Find predicted/predictor fields using:' has 'Model output field metadata' selected. Under 'Other Score Fields', 'Plot score fields' is empty. The 'Target' field is empty. The 'Separate by partition' checkbox is checked. The 'Plot' is set to 'Percentiles'. The 'Style' is set to 'Line'. The 'Costs' section has 'Fixed' selected with a value of 20.0. The 'Revenue' section has 'Fixed' selected with a value of 62.0. The 'Weight' section has 'Fixed' selected with a value of 1.0. The 'Run' button is highlighted in green.

Figure 10.41 The edited Evaluation node: configure for a Profit chart

The resulting Profit chart has been configured so that a model 'hit' (i.e. where the variable 'Retain' is equal to 'Yes') generates \$62 whereas those who are not retained cost the telco \$20. To see the profit chart, click:

Run

Figure 10.42 shows the resulting output.

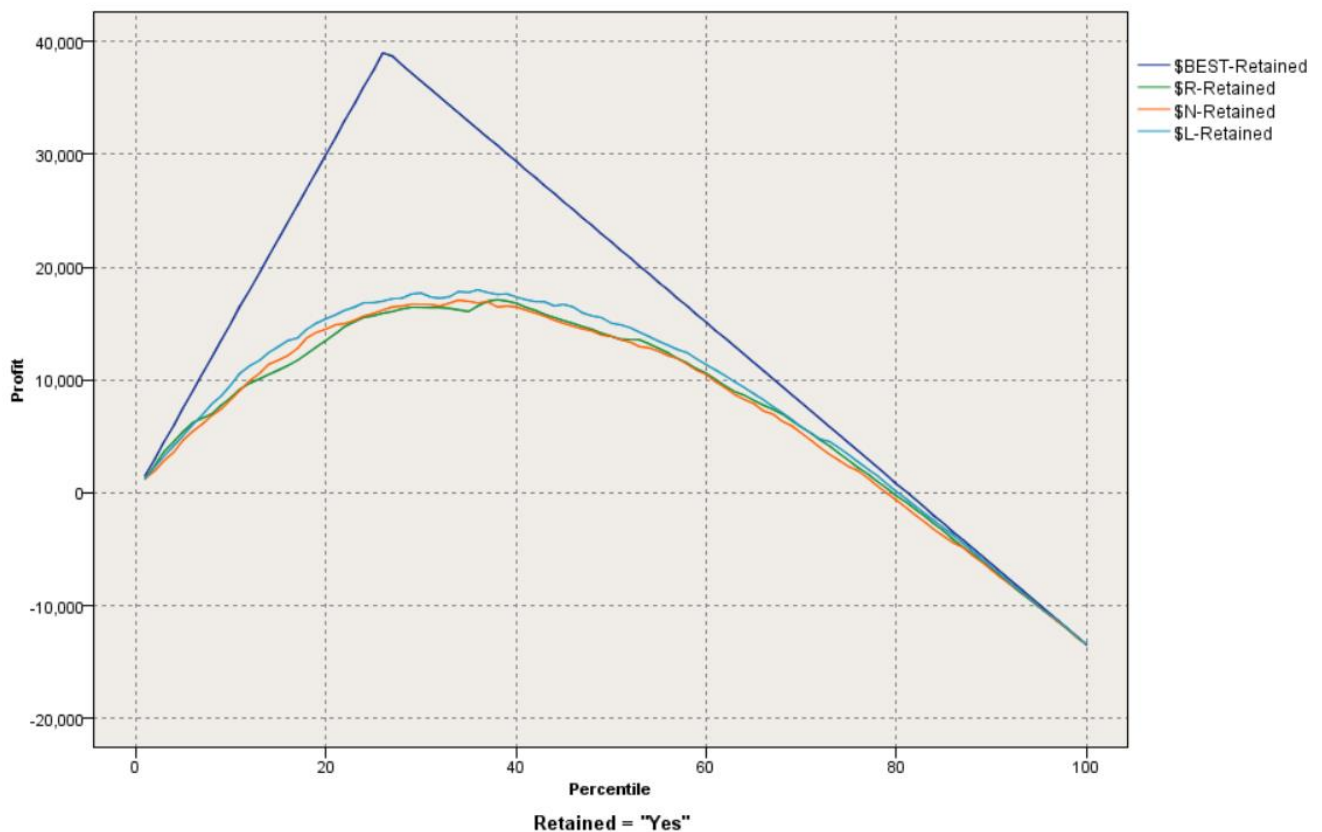


Figure 10.42 Profit chart showing performance of three models and 'Best line'

The chart looks very different from the Gains charts that we have seen before. The easiest way to interpret it is to first look at the apex of the 'Best line' (near the 26th percentile on the horizontal axis). This indicates that if a model was able to identify all those customers that would be retained by the offer, with no churners, then the campaign would generate just under \$40K profit (assuming the campaign size was equal to the sample size in this example). The lowest point of the line (near the 100th percentile) indicates that if we were to make the offer to everyone, irrespective of their likelihood to be retained, then this sample would generate a loss of around \$13.5K. The apex of the three models however, is around the 36th percentile. If we hover the cursor over this point we can see that the Logistic Regression model (\$L-Retained) estimates a net profit of just under \$18K. Figure 10.43 illustrates this.

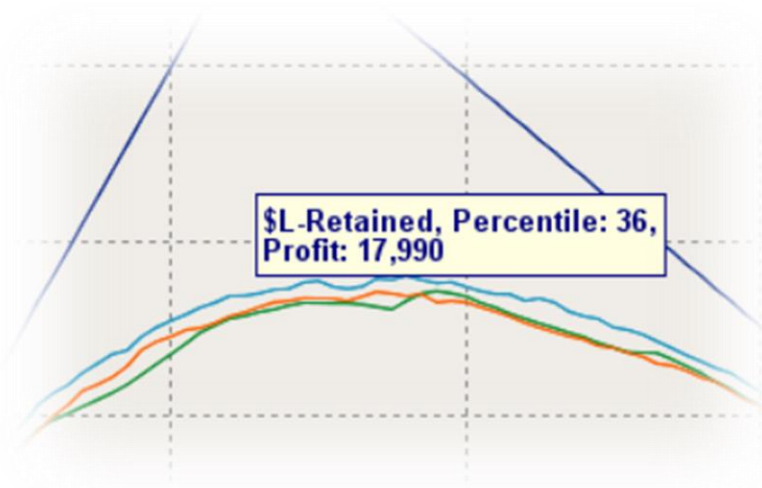


Figure 10.43 The Logistic Regression model estimates a profit of \$17,990 at the 36th percentile of the Profit chart.

The Profit chart indicates that if we choose the 36% of cases that the Logistic Regression model indicates have the highest likelihood of being retained (or the lowest likelihood of defecting) then the estimated profit from a campaign of this size would be \$17,990.

Once again, we need only switch to interactive mode within the chart and use the band selection tool to generate a Select node that will retrieve the optimal percentage of records. According to the Logistic Regression model's predictions, this selection should generate the maximum profit. Figure 10.44 illustrates this.

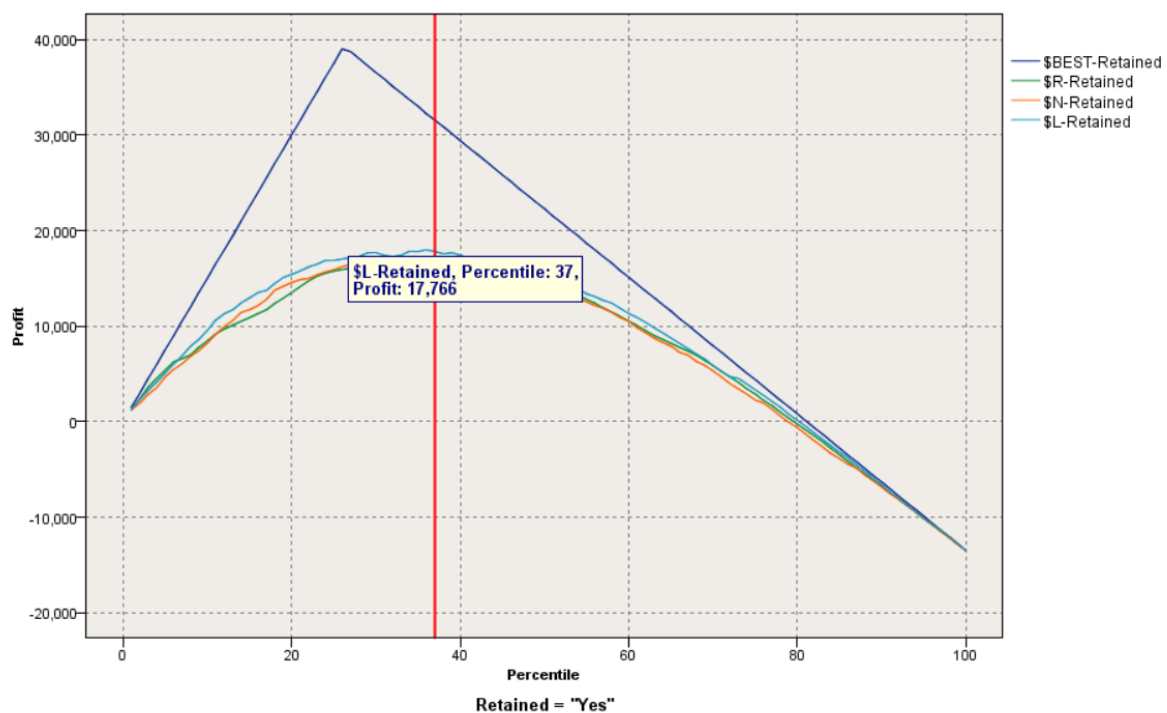


Figure 10.44 Using the Band Selection tool in Interactive mode to select the optimal selection of records to maximise profit

Again, to generate the select node:

Right-click on the area to the left of the selection line

From the pop-up menu chose:

Generate Select node for Band

Figure 10.45 shows the resultant pop-up menu.

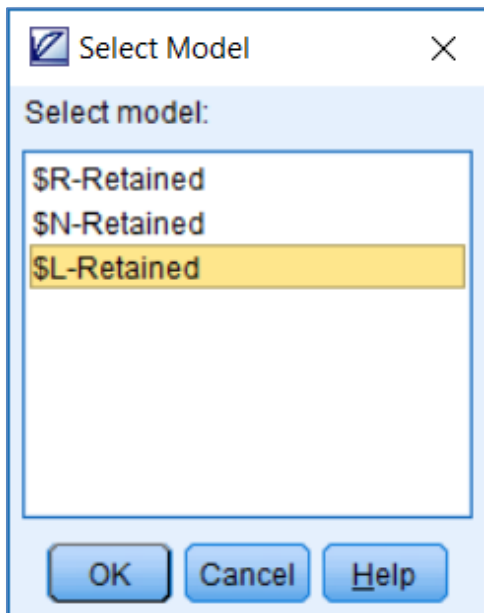


Figure 10.45 Choosing a Model to generate a Select Node

From the pop-up menu, choose the Logistic Regression model denoted as:

\$L-Retained

OK

Once again, a Select node is created and placed on the stream canvas. In this example, we need only copy the Select node and the Logistic Regression model nugget to the area where the campaign data is to be scored.

Copy the Select node and the Logistic Regression nugget

Scroll down the stream canvas and paste them next to the source node for the file 'Scoring Data for Profit selection.txt'

Connect the pasted Select node to the Source node

Connect the model nugget to the Select node

Figure 10.46 shows this updated stream branch.

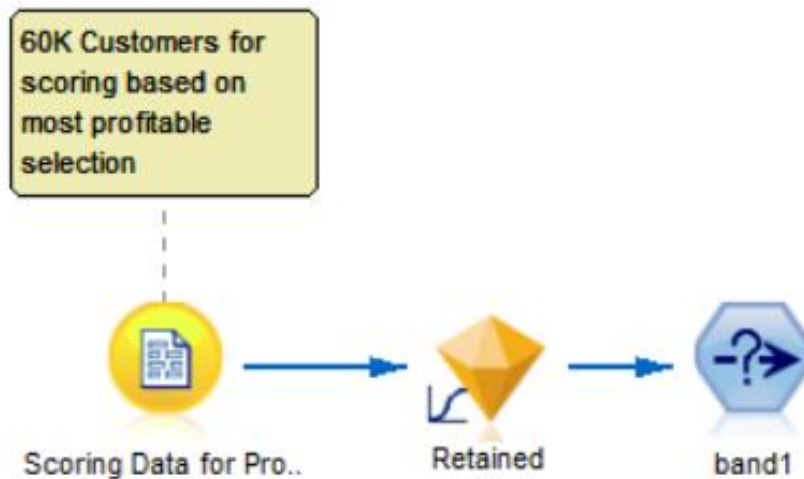


Figure 10.46 Connecting the model nugget and Select node to a data source for scoring

As before, we can add a Filter node to remove unwanted fields and rename variables.

Attach a Filter node to the pasted Select node

Edit the Filter node and remove all fields except:

customerID

\$L-Retained

\$LP-Retained

Rename '\$L-Retained' to 'Prediction'

Rename '\$LP-Retained' to 'Confidence'

Figure 10.47 shows the completed Filter node.

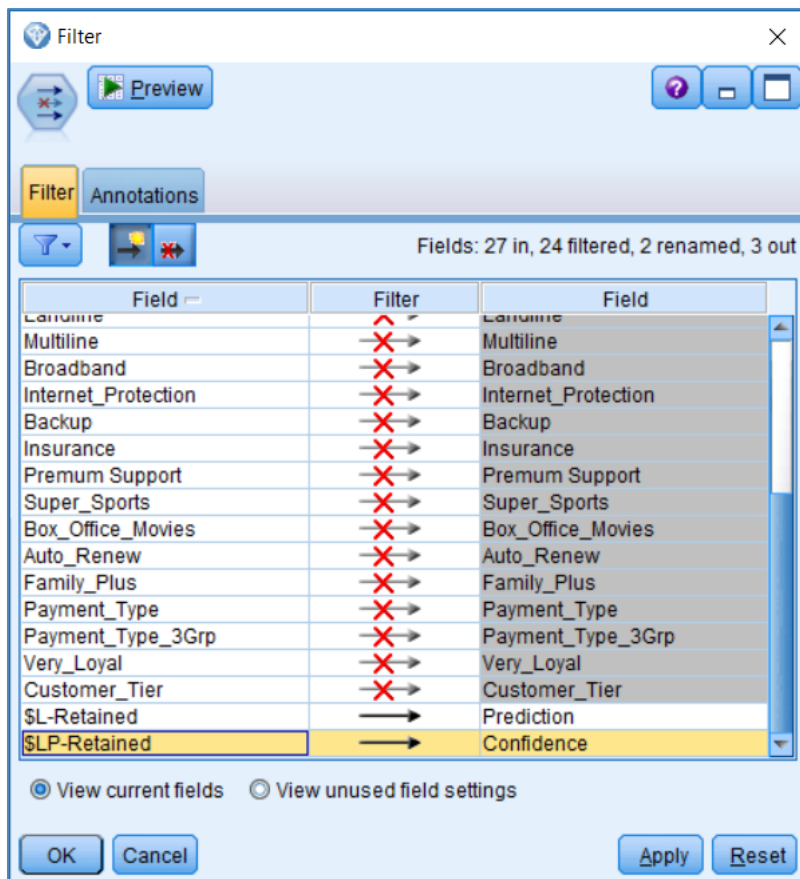


Figure 10.47 Filtering and Renaming fields prior to export

The Database Export Node



In our last example of exporting data, we will use the Database Export node. This node is equivalent of the Database Source node except that of course it *writes* rather than reads data. From the Export palette:

Choose and attach a Database Export node to the Filter node

Double-click the Database Export node to edit it

In the Data Source drop down choose:

Telco_Retention_Access_DB

Specify the table name as:

Retention_For_Profit

Figure 10.48 shows the completed dialog.

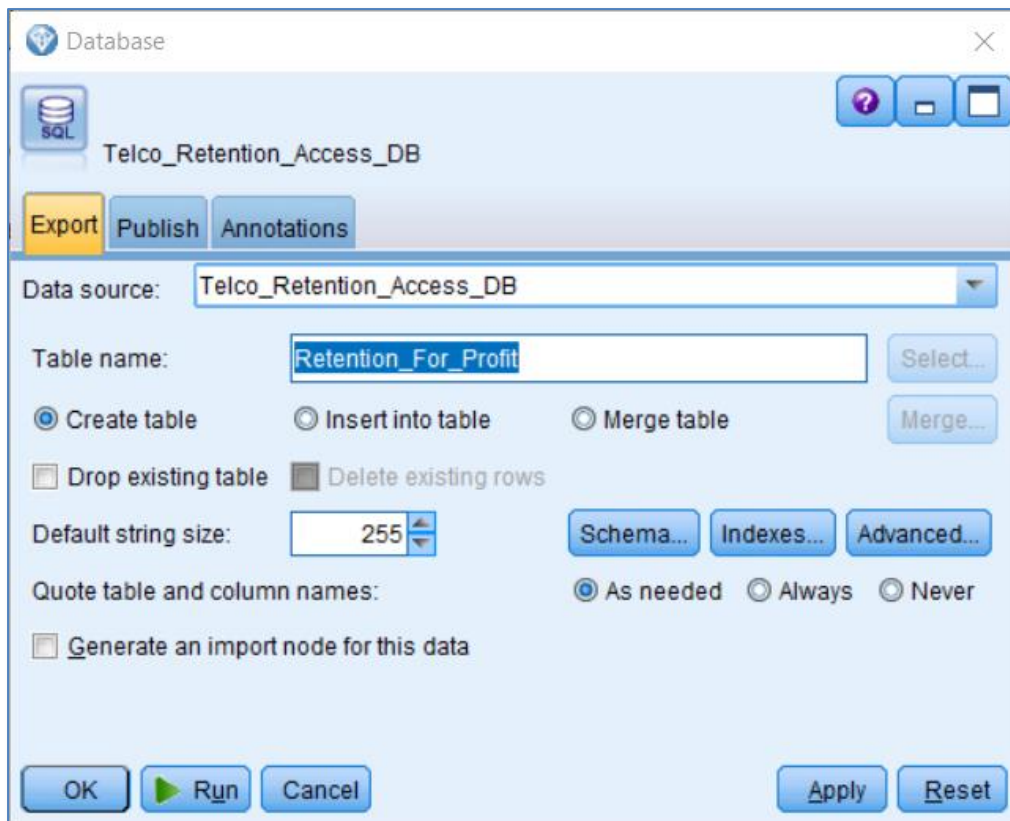


Figure 10.48 The Database Export node configured to send scored data to a database

Click:

Run

The data is now scored with the Logistic Regression. In turn, the Select node selects those cases prior to export that are likely to generate the greatest overall profit in the retention campaign.



PRACTICE SESSIONS

In this part of the course you will have a chance to explore the topics that each section has covered and to try out some of the procedures that we've seen during the demonstrations.

You should find all the files needed to complete each practice session within this folder:

C:\SV Training\Modeler Intro\Practice

Each session will consist of a set of tasks that you are asked to complete. If you need a little help to complete a task, then try looking at the hint section printed at the end of each session.

Section 1 – Practice Session

This first practice section focusses on CRISP-DM and getting used to working with the Modeler interface.

Task 1

1. Which phase of CRISP-DM is typically the most time-consuming?
-

2. Which of the following topics are not one of the explicit CRISP-DM Phases?

- **Business Understanding**
- **Data Exploration**
- **Data Understanding**
- **Data Simulation**
- **Prediction**
- **Evaluation**
- **Deployment**

3. In which phase of CRISP-DM do we check to see if the model has met the business objectives?
-

Contd...

Task 2

Start the Modeler application and from the directory **C:\SV Training\Modeler Intro\Practice** open the following stream file:

Section 1 Practice.str

1. Connect the Data Source node to the Table node and run that part of the stream. How many fields and records are there in the file? Close the output.
2. Connect the Data Source node to the Data Audit node and run that part of the stream. Which field is marked as 'Ordinal'? Close the output.
3. Build a model using the Model building node.
 - a. Inspect the contents of the model nugget.
 - b. Click the Preview button within the model viewer window and look at the last fields in the previewed dataset. What do they refer to?
 - c. Close the model output window.
 - d. Delete the mode nugget from the stream.
4. Insert/re-route the Select node between the Data Source node and the Model building node. Run the model building node again. Is the new model different from the previous one?
5. Look in the Outputs tab and the Models tab on the right-hand side. Do they contain any objects?

Section 1 – Practice Hints

This first practice section focusses on CRISP-DM and getting used to working with the Modeler interface.

Task 1

1. Which phase of CRISP-DM is typically the most time-consuming? Look at page 1.4
-

2. Which of the following topics are not one of the explicit CRISP-DM Phases? Look on page 1.3

- **Business Understanding**
- **Data Exploration**
- **Data Understanding**
- **Data Simulation**
- **Prediction**
- **Evaluation**
- **Deployment**

3. In which phase of CRISP-DM do we check to see if the model has met the business objectives? Look at page 1.4
-

Contd...

Task 2

Start the Modeler application and from the directory **C:\SV Training\Modeler Intro\Practice** open the following stream file:

Section 1 Practice.str Click: File> Open Stream and navigate to the Practice folder

1. Connect the Data Source node to the Table node and run that part of the stream. How many fields and records are there in the file? Right-click on the connected table node and click 'Run'. Look at the title bar of the resultant output window. Close the output.
2. Connect the Data Source node to the Data Audit node and run that part of the stream. Which field is marked as 'Ordinal'? Age Group. Close the output.
3. Build a model using the Model building node.
 - a. Inspect the contents of the model nugget. Double Click the nugget.
 - b. Click the Preview button within the model viewer window and look at the last fields in the previewed dataset. What do they refer to? They are the model's predictions.
 - c. Close the model output window.
 - d. Delete the mode nugget from the stream. Just click it and press the delete key.
4. Insert/re-route the Select node between the Data Source node and the Model building node. Try clicking on the existing connection and using the middle mouse button. Run the model building node again. Is the new model different from the previous one? Yes, it has a different number of cases and uses a different combination of variables.
5. Look in the Outputs tab and the Models tab on the right-hand side. Do they contain any objects? The Model tab should contain the model nugget. The Output tab may contain the outputs from the Table node and the Data Audit node. It depends on how you closed the output – 'Close' will keep a copy of the Output in the Outputs tab but 'Close and Delete' will not.

Section 2 – Practice Session

This second practice section focusses on reading data files into Modeler. Imagine you have been sent three data files to begin a predictive analytics project. The files are stored in:

C:\SV Training\Modeler Intro\Practice

Feel free to navigate to the folder using Windows Explorer and examine any of the data files in this practice session independently of Modeler.

Task 1

1. Start with a clean stream canvas.
2. From the Source tab, choose the appropriate Data Source node to read this file:

Goldscreenz_Customers.xlsx

3. From the Output tab connect a Table node to view the data. How many fields and records does it contain?

Task 2

1. Within the same stream. From the Source tab, choose the appropriate Data Source node to read this file:

Goldscreenz_Products.csv

2. Again, from the Output tab connect a Table node to view the data. How many fields and records does it contain?

Contd...

Task 3

1. Within the same stream. From the Source tab, choose the appropriate Data Source node to read this file:

Goldscreenz_Usage.dat

2. Again, from the Output tab connect a Table node to view the data. How many fields and records does it contain?

Task 4

1. Returning to the file Goldscreenz_Customers.xlsx, using either the Preview button in the Source node or the output from the Table node, find a way to use the Generate menu to select *only customers where the value for the field 'Cancelled' equals 'T'*.
2. How many customers have cancelled their contracts in the file?

Task 5

1. Save the stream as **'Section 2 Practice Reading Data.str'**.

Section 2 – Practice Hints

This second practice section focusses on reading data files into Modeler. Imagine you have been sent three data files to begin a predictive analytics project. The files are stored in:

C:\SV Training\Modeler Intro\Practice

Feel free to navigate to the folder using Windows Explorer and examine any of the data files in this practice session independently of Modeler.

Task 1

1. *Start with a clean stream canvas. File > New Stream*
2. *From the Source tab, choose the appropriate Data Source node to read this file: The Excel Source node*

Goldscreenz_Customers.xlsx

3. *From the Output tab connect a Table node to view the data. How many fields and records does it contain? 7 fields 1724 records*

Task 2

1. *Within the same stream. From the Source tab, choose the appropriate Data Source node to read this file: The Var. File node.*

Goldscreenz_Products.csv

2. *Again, from the Output tab connect a Table node to view the data. How many fields and records does it contain? 5 fields, 1725 records*

Contd...

Task 3

1. Within the same stream. From the Source tab, choose the appropriate Data Source node to read this file: The Var. File node. Use a pipe '|' character as the delimiter.

Goldscreenz_Usage.dat

2. Again, from the Output tab connect a Table node to view the data. How many fields and records does it contain? 7 fields, 44,858 records.

Task 4

1. Returning to the file Goldscreenz_Customers.xlsx, using either the Preview button in the Source node or the output from the Table node, find a way to use the Generate menu to select only customers where the value for the field 'Cancelled' equals 'T'. Click on a cell containing a 'T' character. Then click Generate > Select Node ("And") or Select Node ("Or"). Use the generated Select node to filter the data rows.
2. How many customers have cancelled their contracts in the file? 862 customers.

Task 5

1. Save the stream as '**Section 2 Practice Reading Data.str**'. You know how to save a file, right?

Section 3 – Practice Session

This third practice section focusses on Data Understanding. You can try out techniques to help gain an understanding of the quality of the data you will be working with.

Task 1

1. Open the stream **Section 3 Practice.str**
2. From the Field Ops tab, select and join a Filter node to the data Source node reading the file **Goldscreenz_Customers.xlsx**.
3. Use the Filter node to:
 - a. Rename the field '**Customer_id**' to '**Cust_ID**'
 - b. Remove the field 'Email'
4. From the Field Ops tab, select and join a Type node to the previous Filter node.
5. Use the Type node to fully instantiate the data. Does the full instantiation reveal any issues with any of the fields?
6. From the Output tab, attach a Data Audit node to the previous Type node and run it. Do any of the fields contain invalid records?

Task 2

1. Attach a Type node to the data Source node reading the file **Goldscreen_Products.csv**.
2. Fully instantiate the data. Do any of the fields appear to have unusual or erroneous values?
3. Attach a Data Audit node to the previous Type node and run it. Do any of the fields contain invalid values?

Contd...

4. Return to the Type node for this data file and change the Check column setting for the **Revenue** field to '**Coerce**'. Re-run the Data Audit node for this branch of the stream. Has anything changed? Inspect the Quality tab within the Audit node. Are any of the data missing?
5. Return again to the Type node for this data file and 'switch on' missing values for the **Revenue** field. Re-run the Data Audit node. What has changed?

Task 3

1. Attach a Type node to the data Source node reading the file **Goldscreenz_Usage.dat**.
2. Fully instantiate the data. Are the levels of measurement correct for all the fields? Edit this property if you think it is appropriate to do so for any of them.
3. Attach a Data Audit node to this Type node and run it. Are there any fields with invalid values? Are there any outliers or extremes in the data?
4. Return to the type node and define '-1' as a missing value for the fields **Premium_Content_Hours** and **Standard_Content_Hours**.
5. Re-run the Data Audit node. How many cases contain missing values for these two fields?
6. Within the Quality tab of the Audit node, use the Generate menu to identify any cases that contain the missing data in any of the fields. How many cases does the procedure select?
7. Close the stream without saving it.

Section 3 – Practice Hints

This third practice section focusses on Data Understanding. You can try out techniques to help gain an understanding of the quality of the data you will be working with.

Task 1

1. Open the stream **Section 3 Practice.str**
2. From the Field Ops tab, select and join a Filter node to the data Source node reading the file **Goldscreenz_Customers.xlsx**.
3. Use the Filter node to: both of these tasks are pretty easy
 - a. Rename the field '**Customer_id**' to '**Cust_ID**'
 - b. Remove the field '**Email**'
4. From the Field Ops tab, select and join a Type node to the previous Filter node.
5. Use the Type node to fully instantiate the data. Does the full instantiation reveal any issues with any of the fields? Look at the field '**Gender**'.
6. From the Output tab, attach a Data Audit node to the previous Type node and run it. Do any of the fields contain invalid records? No, they are all valid.

Task 2

1. Attach a Type node to the data Source node reading the file **Goldscreen_Products.csv**.
2. Fully instantiate the data. Do any of the fields appear to have unusual or erroneous values? The field '**Revenue**' has minus numbers and the field '**International**' seems to have inconsistencies.
3. Attach a Data Audit node to the previous Type node and run it. Do any of the fields contain invalid values? Look at the field '**Revenue**'.

Contd...

4. Return to the Type node for this data file and change the Check column setting for the **Revenue** field to '**Coerce**'. Re-run the Data Audit node for this branch of the stream. Has anything changed? Inspect the Quality tab within the Audit node. Are any of the data missing? It all suddenly looks complete. All the invalid values have been coerced to valid. Basically, any null values have been replaced with the mid-point value of the field 'Revenue'. Furthermore, Modeler assumes any negative values are also legitimate unless told otherwise.
5. Return again to the Type node for this data file and 'switch on' missing values for the **Revenue** field. This requires you to click in the adjacent cell under the 'Missing' column and select 'On'. Remember that, by default, Modeler will now treat Nulls and empty strings as missing. Re-run the Data Audit node. What has changed? The null values are no longer coerced. Also, they are being counted as 'missing'.

Task 3

1. Attach a Type node to the data Source node reading the file **Goldscreenz_Usage.dat**.
2. Fully instantiate the data. Are the levels of measurement correct for all the fields? Edit this property if you think it is appropriate to do so for any of them. Arrears Flag?
3. Attach a Data Audit node to this Type node and run it. Are there any fields with invalid values? Yes. There are 88 cases containing empty strings in the field 'Main_Genre' and there is a single record with a null value in the field 'Premium_Content_Hours'. Are there any outliers or extremes in the data? Yes. Several.
4. Return to the type node and define '-1' as a missing value for the fields **Premium_Content_Hours** and **Standard_Content_Hours**.
5. Re-run the Data Audit node. How many cases contain missing values for these two fields? 5 cases for Premium Content and 4 cases for Standard Content. Remember that the field with 5 cases containing missing values includes a null which in this situation has automatically been treated as missing as well.
6. Within the Quality tab of the Audit node, use the Generate menu to identify any cases that contain the missing data in any of the fields. How many cases does the procedure select? It finds 89 records in total. This procedure selects any case that contains: a null value, a user missing value (such as -1) or a white space (empty string).
7. Close the stream without saving it

Section 4 – Practice Session

The fourth practice section focusses on restructuring the data. You can try out the Distinct, Aggregate and Set to Flag nodes as well as the Web plot node.

Task 1

1. Open the stream **Section 4 Practice.str**
2. From the Record Ops palette, attach a Distinct node to the data source node reading the file **Goldscreenz_Customers.xlsx**.
3. Use the correct setting in the Distinct Node to identify the records that contain duplicate Customer IDs.
4. Edit the Distinct node procedure so that you can create a composite record for each group. Using this mode, create a composite record that includes:
 - a. The earliest registration date
 - b. The most recent transaction date
 - c. The maximum value for Tenure
 - d. Last record for Cancelled
5. Attach a Table node to the completed Distinct node and run it. Check to ensure that the number of records in the file has reduced by the correct amount.
6. Use the Distinct node to check if there are any duplicate customer IDs in the file **Goldscreen_Products.csv**. Are there?

Task 2

1. From the Record Ops palette, attach an Aggregate node to the data source node reading the file **Goldscreenz_Usage.dat**.

2. Use the Aggregate node to create a summary data file that for each customer ID records the following information:
 - a. The most recent billing date
 - b. The maximum value for the arrears flag
 - c. The total number of and average value of service failures
 - d. The total number of and average value of premium content hours
 - e. The total number of and average value of standard content hours
 - f. Create a field that records the number of records that were used to create each summary row – call this field 'Billed_Months'
3. Add a Table node to the completed Aggregate node and run it. How many records and fields are shown in the summarised file?

Task 3

1. From the Field Ops palette, attach a Set to Flag node to the data source node reading the file **Goldscreenz_Usage.dat**.
2. Use the Set to Flag node to create a series of flag fields for each category of the **Main_Genre**. Remember that there should only be one record for each customer ID.
3. From the Graphs palette attach a Web Plot node to the Set to Flag node. Edit the Web plot node so that:
 - a. It plots all of the genre flag nodes except the flag field with no genre specified.
 - b. Ensure that only the 'True' values are plotted
 - c. Click the 'Appearance' tab and switch off the legend display
 - d. Experiment with the resultant plot so that you only see the strongest associations between movie genres
4. When you have finished, close the stream without saving it.

Section 4 – Practice Hints

The fourth practice section focusses on restructuring the data. You can try out the Distinct, Aggregate and Set to Flag nodes as well as the Web plot node.

Task 1

1. Open the stream **Section 4 Practice.str**
2. From the Record Ops palette, attach a Distinct node to the data source node reading the file **Goldscreenz_Customers.xlsx**.
3. Use the correct setting in the Distinct Node to identify the records that contain duplicate Customer IDs. You need only specify 'Customer_id' in the 'Key fields for grouping' box. You don't even have to change the mode to 'Include only the first record in each group'.
4. Edit the Distinct node procedure so that you can create a composite record for each group. Using this mode, create a composite record that includes: Click on the Composite tab, choose each variable in turn and just choose the correct summary from the drop-down next the field name. You can ignore the variable Gender.
 - a. The earliest registration date
 - b. The most recent transaction date
 - c. The maximum value for Tenure
 - d. Last record for Cancelled
5. Attach a Table node to the completed Distinct node and run it. Check to ensure that the number of records in the file has reduced by the correct amount. Look at the title bar.
6. Use the Distinct node to check if there are any duplicate customer IDs in the file **Goldscreen_Products.csv**. Are there? No there aren't.

Task 2

1. From the Record Ops palette, attach an Aggregate node to the data source node reading the file **Goldscreenz_Usage.dat**.
2. Use the Aggregate node to create a summary data file that for each customer ID records the following information: The key field is CUST_ID.
 - a. The most recent billing date – choose max
 - b. The maximum value for the arrears flag -choose max
 - c. The total number of and average value of service failures – sum and mean
 - d. The total number of and average value of premium content hours -sum and mean
 - e. The total number of and average value of standard content hours sum and mean
 - f. Create a field that records the number of records that were used to create each summary row – call this field 'Billed_Months' – rename 'Record_Count' to 'Billed_Months'.
3. Add a Table node to the completed Aggregate node and run it. How many records and fields are shown in the summarised file? 1,724 records and 10 fields

Task 3

1. From the Field Ops palette, attach a Set to Flag node to the data source node reading the file **Goldscreenz_Usage.dat**.
2. Use the Set to Flag node to create a series of flag fields for each category of the **Main_Genre**. Remember that there should only be one record for each customer ID. So specify CUST_ID as the Aggregate Key.
3. From the Graphs palette attach a Web Plot node to the Set to Flag node. Edit the Web plot node so that:
 - a. It plots all of the genre flag nodes except the flag field with no genre specified.
 - b.

- c. Ensure that only the 'True' values are plotted - Check the box marked 'Show true flags only'*
 - d. Click the 'Appearance' tab and switch off the legend display – uncheck the 'Show legend box'*
 - e. Experiment with the resultant plot so that you only see the strongest associations between movie genres – click the '>>' button, click the Control tab, click the radio button 'Size shows strong/normal/weak'.*
- 4. When you have finished, close the stream without saving it.*

Section 5 – Practice Session

Practice Section 5 looks at combining data sources. You can try out the Append, Merge and Field Reorder nodes in this practice session.

Task 1

1. Open the stream **Section 5 Practice.str**
2. From the Record Ops palette select an Append node and connect it to the two Distinct nodes connected to the Data Source nodes reading the files **2015 Customers.xlsx** and **Post 2015 Customers.xlsx** respectively.
3. Edit the Append node and make any changes necessary to the stream so that:
 - a. The two files are correctly appended to one another.
 - b. Switch off the 'Tag records by....' function.
 - c. Attach a Table node to the completed Append node. How many records and fields are there in the consolidated file?

Task 2

1. From the Record Ops palette select a Merge node and connect it between the Data Source node reading the file **Goldscreen_Products.csv** and the completed Append node.
2. Edit the Merge node. You should aim to complete the merge the data so that all records are matched to the customer data (in the consolidated file from the Append node). After the merge is complete, there should be no records that are unmatched to the customer data.
3. Attach a Table node and check to see if the number of merged records looks correct.

Task 3

1. Use a Merge node to correctly merge the records from the Aggregate node and the Set to Flag node from the file **Goldscreenz_Usage.dat**.
2. Attach a Table node and check to ensure this has worked.

Task 4

1. Now join the two Merge nodes to each other.
2. If there is a problem, take whichever necessary steps you feel are appropriate to resolve this and place a table node after all the data files have been merged to check that it has worked.

Task 5

1. From the Field Ops palette place attach a Field Reorder node to the last merge step and ensure that the following fields occur at the start of the file:
Cust_ID
Registration_Date
Last_Transaction_Date
2. Ensure that the last field is **Cancelled**.
3. Place a final Table node at the end of the stream data and view the merged data.

Section 5 – Practice Hints

Practice Section 5 looks at combining data sources. You can try out the Append, Merge and Field Reorder nodes in this practice session.

Task 1

1. Open the stream **Section 5 Practice.str**
2. From the Record Ops palette select an Append node and connect it to the two Distinct nodes connected to the Data Source nodes reading the files **2015 Customers.xlsx** and **Post 2015 Customers.xlsx** respectively.
3. Edit the Append node and make any changes necessary to the stream so that: You will need to ensure that the files have fields with the same name. The simplest way to resolve this is to go to the Filter tab in the source node for the file Post 2015 Customers.xlsx and rename 'Tenure_Mnths' to 'Tenure_months'.
 - a. The two files are correctly appended to one another.
 - b. Switch off the 'Tag records by....' function.
 - c. Attach a Table node to the completed Append node. How many records and fields are there in the consolidated file?

Task 2

1. From the Record Ops palette select a Merge node and connect it between the Data Source node reading the file **Goldscreen_Products.csv** and the completed Append node.
2. Edit the Merge node. You should aim to complete the merge the data so that all records are matched to the customer data (in the consolidated file from the Append node). After the merge is complete, there should be no records that are unmatched to the customer data. An inner join will work.
3. Attach a Table node and check to see if the number of merged records looks correct.

Task 3

1. Use a Merge node to correctly merge the records from the Aggregate node and the Set to Flag node from the file **Goldscreenz_Usage.dat**. An inner join will work here.
2. Attach a Table node and check to ensure this has worked.

Task 4

1. Now join the two Merge nodes to each other.
2. If there is a problem, take whichever necessary steps you feel are appropriate to resolve this and place a table node after all the data files have been merged to check that it has worked. – You can make this work by adding a filter node after the most recent Merge node and changing the field name 'CUST_ID' to 'Cust_ID'

Task 5

1. From the Field Ops palette place attach a Field Reorder node to the last merge step and ensure that the following fields occur at the start of the file: Just experiment with the arrangement of the fields in the node.

Cust_ID
Registration_Date
Last_Transaction_Date

2. Ensure that the last field is **Cancelled**.
3. Place a final Table node at the end of the stream data and view the merged data.

Section 6 – Practice Session

Practice Section 6 looks at creating and changing fields. In this session you will use the Reclassify, Filler and Derive nodes.

Task 1

1. Open the stream **Section 6 Practice.str**
2. Investigate the field 'Gender' in the file.
3. From the Field Ops palette, use the Reclassify node to overwrite the existing 'Gender' variable ensuring that it's values are consistently represented. Make sure the field only has two categories: 'Male' or 'Female'.
4. Check that it has worked using a Distribution node from the Graphs palette.

Task 2

1. Use a Filler node to overwrite the field 'Revenue'. Make sure that any negative values are changed to zero.
2. Attach another Data Audit node to check that it has worked.

Task 3

1. Use a Derive node to create a field called 'Failure_Rate'. The field should be calculated by dividing 'Service_Failures_Sum' by 'Tenure_Months'.
2. Examine the results by creating a Histogram of the newly created field.

Task 4

1. Use a Derive node to create a Flag field called 'High_Failure', where those cases with a failure rate of 4 or higher are marked as 'T'.
2. What percentage of customers experienced a high failure rate according to this field?

Task 5

1. Use a Derive node to create a field called 'Refund'. Ensure that if a customer has experienced a high failure rate, then the refund value should be equal to 15% of their Revenue amount. Otherwise the field Refund equals zero.
2. What is the maximum refund amount?

Task 6

1. Use a Derive node to create a field that groups the values of the variable 'Standard_Content_Hours_Mean' according to the following criteria:
 - a. The new variable should be called 'Standard_Content_Band'
 - b. Values up to *and including* 6 hours are coded as "1. Low"
 - c. Values above 6 hours but less than 12 are coded as "2. Standard"
 - d. Values of 12 *and over* are coded as "3. High"
 - e. The variable type should be set to 'Ordinal'
2. What percentage of cases are coded as "2. Standard"?

Task 7

1. Attach a Table to the end of the stream and run it.
2. Try to encapsulate as many of the newly added nodes in a single supernode leaving out the Data Source node and the final Table node.
3. Edit the supernode annotation tab so that is labelled "Calculations".
4. Right-click on the supernode and save it as an object in the Practice folder with the file name **Calculations.slb**.

Section 6 – Practice Hints

Practice Section 6 looks at creating and changing fields. In this session you will use the Reclassify, Filler and Derive nodes.

Task 1

1. Open the stream **Section 6 Practice.str**
2. Investigate the field 'Gender' in the file. Run the Data Audit node and look at the field.
3. From the Field Ops palette, use the Reclassify node to overwrite the existing 'Gender' variable ensuring that it's values are consistently represented. Click the radio button 'Existing field'. Make sure the field only has two categories: 'Male' or 'Female'.
4. Check that it has worked using a Distribution node from the Graphs palette.

Task 2

1. Use a Filler node to overwrite the field 'Revenue'. Make sure that any negative values are changed to zero. The condition `@FIELD < 0` will work.
2. Attach another Data Audit node to check that it has worked.

Task 3

1. Use a Derive node to create a field called 'Failure_Rate'. The field should be calculated by dividing 'Service_Failures_Sum' by 'Tenure_Months'. Use the default 'Formula derive' mode.
2. Examine the results by creating a Histogram of the newly created field.

Task 4

1. Use a Derive node to create a Flag field called 'High_Failure', where those cases with a failure rate of 4 or higher are marked as 'T'. Choose 'Flag' from the 'Derive as' drop-down button.

2. What percentage of customers experienced a high failure rate according to this field? 8.94%

Task 5

1. Use a Derive node to create a field called 'Refund'. Ensure that if a customer has experienced a high failure rate, then the refund value should be equal to 15% of their Revenue amount. Otherwise the field Refund equals zero. Choose 'Conditional' from the 'Derive as' drop-down button. IF High_Failure = 'T'. THEN Revenue * 0.15. ELSE 0.
2. What is the maximum refund amount? 83.85

Task 6

1. Use a Derive node to create a field that groups the values of the variable 'Standard_Content_Hours_Mean' according to the following criteria: Choose 'Nominal' from the 'Derive as' drop-down button
 - a. The new variable should be called 'Standard_Content_Band'
 - b. Values up to and including 6 hours are coded as "1. Low"
 - c. Values above 6 hours but less than 12 are coded as "2. Standard"
 - d. Values of 12 and over are coded as "3. High"
 - e. The variable type should be set to 'Ordinal'
2. What percentage of cases are coded as "2. Standard"? 51.39%

Task 7

1. Attach a Table to the end of the stream and run it.
2. Try to encapsulate as many of the newly added nodes in a single supernode leaving out the Data Source node and the final Table node. Just draw a box around the nodes, right-click and choose 'Create SuperNode'.
3. Edit the supernode annotation tab so that is labelled "Calculations".
4. Right-click on the supernode and save it as an object in the Practice folder with the file name **Calculations.slb**.

Section 7 – Practice Session

Practice Section 7 examines the various nodes that enable users to explore relationships between fields in Modeler.

Task 1

1. Open the stream **Section 7 Practice.str**
2. From the Output palette, choose a Matrix node to investigate the relationship between the field 'Cancelled' and the field 'Standard_Content_Band'.
3. Use the Appearance tab within the Matrix node to add percentages to the output so that you can tell *what percentage* of people in '1. Low' group of the field 'Standard_Content_Band' have Cancelled their contracts.
4. Looking at the Matrix Output's Chi-Square test, would this relationship be regarded as 'Statistically significant'?

Task 2

1. From the Graphs palette, choose a Distribution node and create a bar chart of the variable 'Box_Office'. Colour the bars by the variable 'High_Failure'. Can you tell from the chart if those customers with the Box Office option have experienced higher failure rates than those without?
2. Re-run the Distribution node but this time request the option 'Normalize by color'. Are the groups easier to compare?

Task 3

1. From the Graphs palette, choose a Histogram node and create a chart showing the distribution of the variable 'Revenue'.
2. Re-run the histogram but increase the number of bins to 50.
3. Re-run the histogram but colour the bins by the variable 'Cancelled'.

4. Find a way to 'normalise' the histogram. Which customers are more likely to cancel their contracts?

Task 4

1. From the Output palette, attach a Means node to the source node. Using the Means procedure, find out if those customers whose main genre is Documentaries have a higher average tenure.
2. Now find out if those customers in the International group have higher average tenures than those that don't.

Task 5

1. From the Graphs palette, choose the Plot node and chart the relationship between the variables 'Standard_Content_Hours_Sum' and 'Billed-Months'.
2. From the Output palette, choose the Statistics node and find a way to examine the correlation between these two fields. Remember to click the button 'Correlation Settings' and choose the option 'Define correlation strength by absolute value'. What is the absolute correlation value?

Task 6

1. From the Graphs palette, choose the Graphboard node and create a heat map using the fields 'Standard_Content_Band', 'Cancelled' and 'Tenure_Months'. Which group has the highest average tenure?

Section 7 – Practice Hints

Practice Section 7 examines the various nodes that enable users to explore relationships between fields in Modeler.

Task 1

1. Open the stream **Section 7 Practice.str**
2. From the Output palette, choose a Matrix node to investigate the relationship between the field 'Cancelled' and the field 'Standard_Content_Band'.
3. Use the Appearance tab within the Matrix node to add percentages to the output so that you can tell what percentage of people in '1. Low' group of the field 'Standard_Content_Band' have Cancelled their contracts.
4. Looking at the Matrix Output's Chi-Square test, would this relationship be regarded as 'Statistically significant'? Yes, the probability is less than 0.05.

Task 2

1. From the Graphs palette, choose a Distribution node and create a bar chart of the variable 'Box_Office'. Colour the bars by the variable 'High_Failure'. Can you tell from the chart if those customers with the Box Office option have experienced higher failure rates than those without? It's hard to tell unless the chart is normalised.
2. Re-run the Distribution node but this time request the option 'Normalize by color'. Are the groups easier to compare? Those with the Box Office option are more likely to fall into the 'High failure' group.

Task 3

1. From the Graphs palette, choose a Histogram node and create a chart showing the distribution of the variable 'Revenue'.
2. Re-run the histogram but increase the number of bins to 50. Click on the Options tab.
3. Re-run the histogram but colour the bins by the variable 'Cancelled'.

4. Find a way to 'normalise' the histogram. Which customers are more likely to cancel their contracts? Click the Options tab and check the box marked 'Normalise by color'.

Task 4

1. From the Output palette, attach a Means node to the source node. Using the Means procedure, find out if those customers whose main genre is Documentaries have a higher average tenure. Make 'Main_Genre_Documentary' the grouping field.
2. Now find out if those customers in the International group have higher average tenures than those that don't. Make 'International' the grouping field.

Task 5

1. From the Graphs palette, choose the Plot node and chart the relationship between the variables 'Standard_Content_Hours_Sum' and 'Billed-Months'.
2. From the Output palette, choose the Statistics node and find a way to examine the correlation between these two fields. Remember to click the button 'Correlation Settings' and choose the option 'Define correlation strength by absolute value'. What is the absolute correlation value? Put one field in the 'Examine' box and the other field in the 'Correlate' box. Click 'Correlation Settings' and change the option to 'Define correlation strength by absolute value'

Task 6

1. From the Graphs palette, choose the Graphboard node and create a heat map using the fields 'Standard_Content_Band', 'Cancelled' and 'Tenure_Months'. Simply use control click to choose each of the fields and then choose the Heatmap from the chart suggestions. Which group has the highest average tenure? Those in the 'Standard' group of the 'Standard_Content_Band' variable who have also cancelled their contracts.

Section 8 – Practice Session

In this practice section you will have the chance to build a predictive model using the interactive function within the CHAID node.

Task 1

1. Open the stream **Section 8 Practice.str**
2. Double click the source node and change the role for the field 'Cancelled' to 'Target'.
3. Run the Data Audit node and inspect the output first.

Task 2

1. From the Modelling palette (Classification section), find the CHAID node and attach it to the source node.
2. Edit the CHAID node. Within the 'Build Options' tab, choose the section marked 'Objective' (on the left-hand side) and click the radio button 'Launch interactive session'.
3. On the left-hand side of the dialog click the section marked 'Advanced'. To generate a new random seed, click the button marked 'Generate'.
4. Click the Run button to start the interactive model-building session.

Task 3

1. Within the Interactive Tree viewer, on the toolbar, click the buttons to switch between the Tree Growing set and the Overfit Prevention set (these buttons have blue and magenta stripes respectively). Note the different sample sizes in both sets of data.
2. Switch to the Tree Growing set and grow the tree one level (hover your mouse over the buttons along the bottom of the viewer to find the appropriate one).
3. Make a note of the first variable that the decision tree has chosen. Now prune the tree back to the original root node by clicking the 'Remove Branch' button.

4. Now click the button on the bottom of the view that enables you to 'Grow Branch with Custom split'. In the 'Define Split' dialog, click the 'Predictors' button.
5. Choose the second best different predictor field.
6. Back in the 'Define Split' dialog, click the radio button marked 'Custom'. Can you create your own custom split? Try selecting more than one row in the split table and clicking the 'Group value(s)' button. Click 'Grow'.
7. Experiment with growing a single branch of the tree. Switch back and forth between the Tree Growing set and the Overfit Prevention set to see if the percentages in the nodes are similar in both trees.
8. Now grow the entire tree and click the Tree Map button to navigate around the fully-grown tree.
9. Close and delete the Interactive tree (check the Outputs tab on the right-hand side to make sure it isn't still open').

Task 4

1. Return to the CHAID node and change the Objective mode to 'Generate model' (i.e. switch off interactive mode). Click 'Run' to build a model nugget.
2. Double-click the model nugget and inspect its contents.
3. Within the 'Settings' tab, check the box marked, 'Calculate raw propensity scores'.
4. Click the 'Preview' button and look at the last fields in the Preview table to see the variables that the Model generates. Make sure you are clear as to the difference between the Confidence values and the Raw Propensity values.
5. Click 'OK' to close the outputs and close the stream without saving.

Section 8 – Practice Hints

In this practice section you will have the chance to build a predictive model using the interactive function within the CHAID node.

Task 1

1. *Open the stream **Section 8 Practice.str***
2. *Double click the source node and change the role for the field 'Cancelled' to 'Target'.*
3. *Run the Data Audit node and inspect the output first.*

Task 2

1. *From the Modelling palette (Classification section), find the CHAID node and attach it to the source node.*
2. *Edit the CHAID node. Within the 'Build Options' tab, choose the section marked 'Objective' (on the left-hand side) and click the radio button 'Launch interactive session'.*
3. *On the left-hand side of the dialog click the section marked 'Advanced'. To generate a new random seed, click the button marked 'Generate'. This should generate a new random number so everyone gets a slightly different model.*
4. *Click the Run button to start the interactive model-building session. Just run the node.*

Task 3

1. *Within the Interactive Tree viewer, on the toolbar, click the buttons to switch between the Tree Growing set and the Overfit Prevention set (these buttons have blue and magenta stripes respectively). Note the different sample sizes in both sets of data.*
2. *Switch to the Tree Growing set and grow the tree one level (hover your mouse over the buttons along the bottom of the viewer to find the appropriate one). Pop-up labels appear telling you what each button does.*

3. *Make a note of the first variable that the decision tree has chosen. Now prune the tree back to the original root node by clicking the 'Remove Branch' button. Again, just look at the buttons on the bottom and find the right one.*
4. *Now click the button on the bottom of the view that enables you to 'Grow Branch with Custom split'. In the 'Define Split' dialog, click the 'Predictors' button. You should see a table with all the fields showing the best predictors in descending order.*
5. *Choose the second best different predictor field.*
6. *Back in the 'Define Split' dialog, click the radio button marked 'Custom'. Can you create your own custom split? Try selecting more than one row in the split table and clicking the 'Group value(s)' button. Click 'Grow'. These buttons are on the right-hand side of the dialog.*
7. *Experiment with growing a single branch of the tree. Switch back and forth between the Tree Growing set and the Overfit Prevention set to see if the percentages in the nodes are similar in both trees. Just choose a few nodes and see if the differences are big or small.*
8. *Now grow the entire tree and click the Tree Map button to navigate around the fully-grown tree. Look on the toolbar to find the Tree Map button.*
9. *Close and delete the Interactive tree (check the Outputs tab on the right-hand side to make sure it isn't still open'). Click the red 'X' in the right-hand corner and then click 'Continue'.*

Task 4

1. *Return to the CHAID node and change the Objective mode to 'Generate model' (i.e. switch off interactive mode). Click 'Run' to build a model nugget.*
2. *Double-click the model nugget and inspect its contents.*
3. *Within the 'Settings' tab, check the box marked, 'Calculate raw propensity scores'.*
4. *Click the 'Preview' button and look at the last fields in the Preview table to see the variables that the Model generates. Make sure you are clear as to the difference between the Confidence values and the Raw Propensity values. Look at page 8.220.*
5. *Click 'OK' to close the outputs and close the stream without saving.*

Section 9 – Practice Session

In this practice section you will have the chance to build more predictive models and explore methods for evaluating their performance.

Task 1

1. Open the stream **Section 9 Practice.str**
2. From the Modelling palette, choose either a CHAID, C5.0 or a C&R Tree node and build a model with it.
3. From the Output palette, choose an Analysis node and attach it to the model nugget. Run the Analysis node and note how accurate the model is overall. Does this provide enough detail to assess the model performance?
4. Edit the Analysis node and check the box marked 'Coincidence matrices (for symbolic targets)'. Re-run the analysis node. What does the additional detail show?

Task 2

1. Delete the model nugget from the previous task.
2. From the Field Ops palette, insert a Partition node between the data source node and the model-building node.
3. Edit the partition node and change the Training partition size value to '70' and the Testing partition size value to '30'. Click the button marked 'Generate' to create a new random seed.
4. Run the model-building node again. Connect the resultant model nugget to the Analysis node and re-run the stream. What has changed in the Analysis node output? Does the model perform as well on the Testing partition as the Training partition?

Task 3

1. From the Graphs palette, select an Evaluation node and connect it to the model nugget. Run this branch of the stream and inspect the output. Note that the output creates two charts for the Training and Testing groups respectively. Which

outcome category in the target field does the chart refer to? How much better than random is the model at finding cases that fall into this category?

2. Close the output and edit the Evaluation node. Check the box marked 'Include best line'. Re-run the Evaluation node. What has been changed?

Task 4

1. Delete both the model building node and the model nugget from the previous task.
2. From the Modelling palette, find the Auto Classifier node (use the selection tab on the left-hand side to show all models) and attach it to the Partition node. Run the Auto Classifier node.
3. Inspect the resultant model nugget. Which algorithms has the Auto Classifier chosen? Use the appropriate button in the viewer to switch between the Testing and Training view of the results. Do any of the models change positions in the table?
4. Connect the model nugget to the Analysis and Evaluation nodes. Re-run these nodes and compare the model performance to earlier. Has it improved? Which model(s) in the Auto Classifier nugget is generating the predictions?

Section 9 – Practice Hints

In this practice section you will have the chance to build more predictive models and explore methods for evaluating their performance.

Task 1

1. Open the stream **Section 9 Practice.str**
2. From the Modelling palette, choose either a CHAID, C5.0 or a C&R Tree node and build a model with it. They are all Decision Trees and should perform similarly.
3. From the Output palette, choose an Analysis node and attach it to the model nugget. Run the Analysis node and note how accurate the model is overall. Does this provide enough detail to assess the model performance? Not usually. It only provides information on the overall model performance not the false positive or false negatives.
4. Edit the Analysis node and check the box marked 'Coincidence matrices (for symbolic targets)'. Re-run the analysis node. What does the additional detail show? It shows a crosstab that enables us to see how well the model classifies the individual categories in the target field.

Task 2

1. Delete the model nugget from the previous task.
2. From the Field Ops palette, insert a Partition node between the data source node and the model-building node.
3. Edit the partition node and change the Training partition size value to '70' and the Testing partition size value to '30'. Click the button marked 'Generate' to create a new random seed.
4. Run the model-building node again. Connect the resultant model nugget to the Analysis node and re-run the stream. What has changed in the Analysis node output? It now shows two sets of tables for the Training and Testing partitions. Does the model perform as well on the Testing partition as the Training partition? Look at the performance values.

Task 3

1. From the Graphs palette, select an Evaluation node and connect it to the model nugget. Run this branch of the stream and inspect the output. Note that the output creates two charts for the Training and Testing groups respectively. Which outcome category in the target field does the chart refer to? Those who cancelled their contracts. How much better than random is the model at finding cases that fall into this category? Should be significantly better.
2. Close the output and edit the Evaluation node. Check the box marked 'Include best line'. Re-run the Evaluation node. What has been changed? An additional line has been added showing what a 'perfect' model would look like.

Task 4

1. Delete both the model building node and the model nugget from the previous task.
2. From the Modelling palette, find the Auto Classifier node (use the selection tab on the left-hand side to show all models) and attach it to the Partition node. Run the Auto Classifier node.
3. Inspect the resultant model nugget. Which algorithms has the Auto Classifier chosen? Just look at their names under the column header 'Model'. Use the appropriate button in the viewer to switch between the Testing and Training view of the results. Click the button marked 'Testing set'. Do any of the models change positions in the table? They might do if they performed differently on the Training set than the Testing set.
4. Connect the model nugget to the Analysis and Evaluation nodes. Re-run these nodes and compare the model performance to earlier. Has it improved? Probably has improved. Which model(s) in the Auto Classifier nugget is generating the predictions? All of them.

Section 10 – Practice Session

In this practice section you will explore the procedures for scoring data with a predictive model.

Task 1

1. Open the stream **Section 10 Practice.str**.
2. The stream contains a Logistic Regression model connected to an Evaluation node. Run the Evaluation node and inspect the output.
3. Within the output, under the View menu, switch on the Interactive mode. From the chart toolbar activate the Band Selection Tool. Use the tool to select the best 50% of cases that the Logistic model finds in the Gains chart for the Training sample – try to select as close to the 50th percentile as possible.
4. Now right-click in the appropriate area of the chart and generate a Select node for this group.
5. Copy both the generated Select node and the Model nugget and paste them next to the scoring data source that reads the file **Goldscreenz_Scoring.tab**. Connect the model nugget and the Select node to the scoring data source node.
6. Connect a Filter node downstream of the Select node. Use the Filter node to remove all the fields except for the Customer ID field and the field containing the model Propensity score (\$LRP-Cancelled').
7. Use the Filter node to rename **\$LRP-Cancelled** to **Churn-Likelihood**.
8. From the Record Ops palette, add a Sort node to the previous Filter node. Sort the records in *descending* order by the values of **Churn-Likelihood**.
9. Add a Table node to check the results.
10. Finally, from the Export palette, add a Flat file node and export the data as a comma separated file called '**Retention_Campaign_Scores.csv**'.

Section 10 – Practice Hints

In this practice section you will explore the procedures for scoring data with a predictive model.

Task 1

1. Open the stream **Section 10 Practice.str**.
2. The stream contains a Logistic Regression model connected to an Evaluation node. Run the Evaluation node and inspect the output.
3. Within the output, under the View menu, switch on the Interactive mode. Click View > Interactions. From the chart toolbar activate the Band Selection Tool. It looks like a red vertical line. Use the tool to select the best 50% of cases that the Logistic model finds in the Gains chart for the Training sample – try to select as close to the 50th percentile as possible. Just hover the tool over the model line in the Training half of the chart – keep moving it until the percentile value reaches 50 and then click.
4. Now right-click in the appropriate area of the chart and generate a Select node for this group. Right-click on the left-hand side of the band selection and choose ‘Generate Select node for band’ from the pop-up menu.
5. Copy both the generated Select node and the Model nugget and paste them next to the scoring data source that reads the file **Goldscreenz_Scoring.tab**. Connect the model nugget and the Select node to the scoring data source node.
6. Connect a Filter node downstream of the Select node. Use the Filter node to remove all the fields except for the Customer ID field and the field containing the model Propensity score (\$LRP-Cancelled’).
7. Use the Filter node to rename **\$LRP-Cancelled** to **Churn-Likelihood**.
8. From the Record Ops palette, add a Sort node to the previous Filter node. Sort the records in descending order by the values of **Churn-Likelihood**.
9. Add a Table node to check the results. Run the Table node, do the results look right?
10. Finally, from the Export palette, add a Flat file node and export the data as a comma separated file called **‘Retention_Campaign_Scores.csv’**.

