Clustering and Association Models with IBM SPSS Modeler
Student Guide

Course Code: 0A042

ERC 1.0

Authorized IBM Training

Clustering and Association Models with IBM SPSS Modeler

0A042

Published October 2010

TABLE OF CONTENTS

Lesson 1: Introduction to Association and Cluster Modeling Techniques in PASW Modeler

Objectives

- Give a brief introduction to the Association and Clustering modeling techniques available in PASW® Modeler
- Discuss when to use a particular technique and on what type of data

1.1 *Introduction*

PASW Modeler includes three main classes of modeling techniques: Predictive, Clustering and Associations. The purpose of Predictive Models is to predict the value of an outcome variable. Some examples of predictive models are neural nets, decision trees, and regression analysis. For those of you who are interested in these types of model, there is a separate course entitled Predictive Modeling with PASW Modeler. The goal of this course is to introduce you to the two other types of modeling techniques that are available in PASW Modeler.

Unlike Predictive Models, Association and Clustering models do not involve prediction but instead search for groupings or associations in the data. The aim of Clustering techniques is to try to segment the data into groups of cases/records that have similar patterns of input fields.. This type of analysis is often employed in market segmentation studies whose aim it is to find distinct types of customers so they can be marketed to more effectively. For example, segments may consist of people who have similar buying habits when they go to the grocery store, or who visit the same types of restaurants.

Association techniques are analogous to Cluster analysis, but while Cluster analysis typically clusters cases (often people), Association analysis clusters fields (actually, values of fields). These techniques are often referred to as Market Basket Analysis because they were developed to analyze consumer-shopping patterns. These methods can also be used to find associations between: medical procedures prescribed for patients; insurance claims; bank transactions; or telcom services used by customers. While these techniques can be used to predict a specific outcome (and so have some affinity with predictive modeling), the usual first step is to declare the fields within the data to act as both inputs and outputs. Once the normally many associations are found, it is possible to develop rules that focus on a specific outcome. These rules could be used to define several conditions that would lead to the purchase of a particular item. For example, if a store wished to market a particular product more effectively, such as alcohol, these rules would identify patterns between the purchase of alcohol and other types of foods. As with the segments from cluster analysis, what a business or organization does with the rules depends on the business question being investigated.

While Association analysis can be used to detect patterns in product purchase, its rules cannot be used to predict the order in which the products are bought. Typically, association rules are developed without taking time into account. Related to Association analysis is Sequence analysis, which will look for association rules over time (i.e., sequences). These could be stages in processing customer service problems, web pages visited during a visit to a website that led to a product inquiry, or the order of procedures administered to a hospital patient.

In the following sections we will briefly introduce you to the different Clustering and Association Modeling techniques that are covered in this course.

1.1 *Clustering*

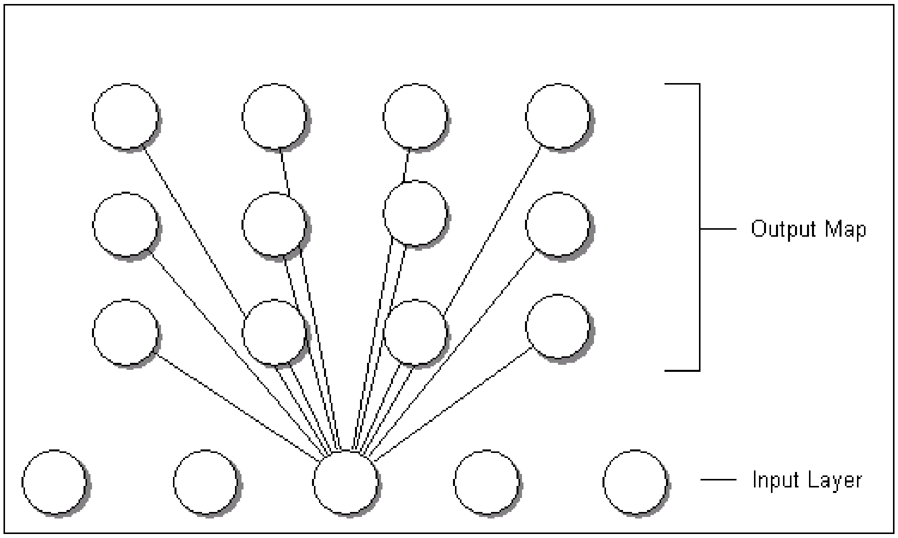
Clustering methods help discover groups of data records with similar values or patterns. These techniques are used in marketing (customer segmentation) and other business applications (records that fall into clusters with very few members may contain errors or be instances of fraud). Clustering is sometimes performed prior to predictive modeling. In such instances, the customer groups might be modeled individually (taking the approach that each cluster is unique) or the cluster group might be an additional input to the model. PASW Modeler offers three clustering methods: Kohonen networks, K-Means clustering, and TwoStep clustering.

Kohonen Networks

A Kohonen network is a type of neural network that performs unsupervised learning; that is, it has no output or outcome to predict. Such networks are used to cluster or segment data based on the patterns of input fields. Kohonen networks make the basic assumption that clusters are formed from patterns that share similar features and will therefore group similar patterns together.

Kohonen networks are usually one- or two-dimensional grids or arrays of artificial neurons. Each neuron is connected to each of the inputs (input fields), and weights are placed on each of these connections. The weights for a neuron represent a profile for that cluster on the fields used in the analysis. There is no actual output layer in Kohonen networks, although the Kohonen map containing the neurons can be thought of as an output. The figure below shows a simple representation of an output grid or Kohonen map.

Figure 1.1 Basic Representation of a Kohonen Network



Note that the connections from the input neuron layer are shown for only one neuron.

When a record is presented to the grid, its pattern of inputs is compared with those of the artificial neurons within the grid. The artificial neuron with the pattern most like that of the input “wins” the input. This causes the weights of the artificial neuron to change to make it appear even more like the input pattern. The Kohonen network also slightly adjusts the weights of those artificial neurons surrounding the one with the pattern that wins the input.

This has the effect of moving the most similar neuron and the surrounding nodes, to a lesser degree, to the position of the record in the input data space. The result, after the data have passed through the network a number of times, will be a map containing clusters of records corresponding to different types of patterns within the data. The user must decide on the appropriate number of clusters, and there will usually be some clusters with only a few records.

K-Means Clustering

K-means clustering is a relatively quick method for exploring clusters in data. The user sets the number of clusters (k) to be created, and the procedure selects k well-spaced data records as starting clusters. Each data record is then assigned to the nearest of the k clusters. The cluster centers (means on the fields used in the clustering) are updated to accommodate the new members. Additional data passes are made as needed; as the cluster centers shift, a data record may need to be moved to its now nearest cluster.

Since the user must set the number of clusters, this procedure is typically run several times, assessing the results (mean profiles, number of records in each cluster, cluster separation) for different numbers of clusters (values of k).

TwoStep Clustering

Unlike the two previous cluster methods, two-step clustering will automatically select the number of clusters. The user specifies a range (Minimum (*Min*) and Maximum (*Max*) for the number of clusters. In the first step, all records are classified into pre-clusters. These pre-clusters are designed to be well separated. In the second step, a hierarchical agglomerative cluster method (meaning that once records are joined together to create clusters, they are never split apart) is used to successively combine the pre-clusters. This produces a set of cluster solutions containing from *Max* clusters down to *Min* clusters. A criterion (likelihood-based) is then used to decide which of these solutions is best.

The two-step clustering method thus has the advantage of automatically selecting the number of clusters (within the range specified) and does not require enormous machine resources (since only the *Max* clusters, not all records, are used in the second step).

1.2 *Association Rules*

Association rule methods search for things (events, purchases, attributes) that typically occur together in the data. Association rule algorithms automatically find the patterns in data that you could manually find using visualization techniques such as the web node, or multi-way tables, but can do so much quicker and can explore more complex patterns.

The rules found associate a particular outcome category (called a conclusion) with a set of conditions. The outcome fields may vary from rule to rule and as a result the user does not often focus on one particular output field. In fact, the advantage of these algorithms over rule induction is that associations can exist between any of the fields. One disadvantage to rule associations is that they attempt to find patterns in what is potentially a very large search space, and can be slow in running. A second disadvantage is that often many rules are found and the user must manually examine the rules for ones of interest.

The three algorithms provided by PASW Modeler to generate association rules are called Apriori, Carma and Sequence. The algorithms begin by generating a set of extremely simple rules. These rules are then specialized by adding more refined conditions to them (making the rules more complex) and the most interesting rules are stored.

PASW Modeler allows certain restrictions on the algorithms to help speed up the process such as limiting the number of possible conditions within a rule. The result is a set of rules that can be viewed but cannot be used directly for predicting.

1.3 *Sequence Detection*

Sequence detection methods search for sequential patterns in time-structured data. Their focus on time-ordered sequences, rather than general association, is what distinguishes them from the association rule methods. In such analyses there may be interest in identifying common sequence patterns or in finding sequential patterns that often lead to a particular conclusion (for example, a purchase on a web-site, or a failure of a piece of equipment).

Application areas of sequence detection include retail shopping, web log analysis, and process improvement (for example, finding common sequences in the steps taken to resolve problems with electronic devices).

Sequence detection is applied to categorical fields and if numeric fields are input, their values will be treated as categories. That is, a field that takes the values from 1 to 100 would be treated as having one hundred categories.

The Sequence node in PASW Modeler uses the CARMA algorithm, which makes only two passes through the data. It can also generate nodes that make predictions based on specific sequences.

1.4 *Which Technique, When?*

If you want to find groups of individuals that behave similarly on a number of fields in the data, then any of the clustering methods is appropriate. Association rules are not going to directly give you the ability to predict, but are extremely useful as a tool for understanding the various patterns among fields in the data. If there is interest in sequential patterns in data, then sequence detection methods are the techniques of choice and some of them can be used to generate predictions.

But if you want to go further and decide which particular prediction technique will work better, then unfortunately the answer is that it depends on the particular data you are working on. In fact, more accurately, it depends on the particular fields you want to predict and how they are related to the various inputs. There are suggested guidelines as to when one technique may work better than another, and we will mention these in the following lessons, but these are only suggestions and not rules. They will be broken on many occasions!

Summary

In this lesson you have been introduced to a number of the machine learning and statistical modeling capabilities of PASW Modeler. You should now be able to explain the different types of analyses you can perform and the different algorithms that can help you achieve your desired outcome.

Lesson 2: Techniques for Clustering

Objectives

- Review the three clustering algorithms in PASW Modeler
- Provide comparisons among the three techniques
- Provide advice on when and how to use each

Data

In this lesson we use the dataset *churn.txt*, containing information on 1,477 customers of a telecommunication firm who has at some time purchased a mobile phone. The customers fall into one of three groups: current customers, involuntary leavers, and voluntary leavers. The file contains information on the customer account including length of time spent on local, long distance and international calls, the type of billing scheme, and a variety of basic demographics, such as age and gender. The data are typical of what is often referred to as a churn example (hence the file name).

2.1 *Introduction*

Cluster analysis is an exploratory data analysis technique designed to reveal natural groupings within a dataset. The basic criterion used for this is distance, in the sense that cases (records) close together should fall into the same cluster, while cases far apart should be in different clusters. Ideally, the cases within a cluster would be relatively homogeneous, but different from those contained in the other clusters.

Cluster analysis is not always (usually) an end in itself, and can be one step in a data mining project. Moreover, there is usually not a unique or definitive solution in a cluster analysis. The number of clusters created is a user-controlled setting in both the K-Means and Kohonen nodes, and typically several iterations of the K-Means procedure will be run until a satisfactory cluster solution is found. However, the TwoStep cluster node will select an optimal cluster solution within a range that you specify. To determine whether a cluster solution is useful requires exploration of the cluster profiles and how clusters differ on important fields, both those used to create the clusters and others.

There are many different clustering methods, but in the area of data mining only a few are in wide use. This is because the many hierarchical clustering methods require that distances between every pair of records be stored and updated in memory, which places a substantial demand on memory and resources for large files. Instead, clustering is performed using K-means or the Kohonen net, which is an unsupervised neural network. PASW Modeler also has a third clustering algorithm (TwoStep) that requires only a single pass through the data and is efficient in memory usage.

We begin with some general advice on clustering solutions, then move on to examine each clustering method.

2.2 *What to Look For When Clustering*

There are some standard principles that can be applied to any clustering solution. We mention the most critical in this section.

Number of Records per Cluster

The goal of cluster analysis is to create groupings of cases that reflect natural groupings in the data. But to be useful, clusters should not be too small in size and contain only a few cases. While it is certainly possible for a few outlying cases to form their own cluster, 5 or 10 cases in a dataset of 1,000 records are too tiny a cluster to be practically useful in the great majority of circumstances.

There is no hard and fast rule concerning the minimum number of cases in a cluster, but 5-10% of the file size is a reasonable limit to use upon first examination. If clusters are found that are too small, you have several options:

- Rerun the analysis requesting a solution with fewer clusters.
- Combine the small clusters with others that are nearby (in distance), although this is most easy to do with K-means where distances between clusters can be readily output.
- Drop the cluster from the final solution. This is reasonable when the percentage of cases in a small cluster is truly a tiny fraction of the file, or the cases are uninteresting for other reasons.
- Use the TwoStep node has an option to exclude outliers, which will exclude records in very small clusters from the cluster solution.

Note that points 3 and 4 suggest that the final clustering solution will not apply to all records (also known as incomplete coverage). This strategy is fairly common in data mining, where the overall goal is not necessarily to model all the records but instead to find the most promising relationships.

Number of Clusters

The intent of cluster analysis is to reduce the dimensionality of the data so that a relatively small number of clusters can represent hundreds or thousands of cases and several variables. However, if a data file is large, with tens of thousands of records, and many variables are used in clustering, then even a clustering solution with 100 clusters will be a significant reduction in dimensionality.

Nonetheless, a solution with so many clusters is rarely helpful or practically useful. This is because, as we discuss in the next section, cluster solutions cannot be accepted on face value but must instead be validated to determine their adequacy. Trying to understand the characteristics of many clusters is time-consuming. Thus we recommend limiting the number of clusters to at most a dozen or so, unless you have strong reasons to believe otherwise.
Given that the number of clusters is arbitrary (the exception to this is the TwoStep algorithm) it is standard practice to try solutions with different numbers of clusters, examining each in turn to see which is most useful. Note that the TwoStep cluster algorithm will select an optimal number of clusters within a range that you specify and thus will provide guidance as to the number of clusters.

Validation

A clustering solution should not be accepted until it has been validated. There are at least three approaches to validation.

First, differences between the clusters should be investigated, using both the fields included in the clustering and then other important fields. If there is an outcome field that you hope to predict it should typically not be included among the fields used to create the clusters, but you can observe how it varies across the clusters. In general you examine a profile of each cluster so that the characteristics of each cluster can be described both numerically and in words. With this information in hand, you can then determine whether or not the clusters are truly different in ways that you find substantively important (because the clustering solution does not guarantee this will be true).

Often in modeling we use training and test datasets, and although this is less commonly done in clustering, there is no reason why this technique can’t be applied. A test dataset can be created (perhaps the same one to be used later in modeling), and a second cluster analysis done, attempting to replicate the cluster solution found in the training data. If the same basic clusters are found in the test data (the clusters will never be identical), then you have more confidence that the clustering solution will apply to future data.

Finally, you can try several clustering algorithms on the same data. Thus, if you begin with K-means, you can also try using a Kohonen network or the TwoStep algorithm. However, given that these are very different approaches to clustering, there is no necessary reason to expect that the cluster solutions found will be similar. It may be more effective to use various techniques on the training data to find the best clustering solution there, rather than using one technique to validate another's clustering solution. This is, like so many aspects of cluster analysis, a matter of judgment.

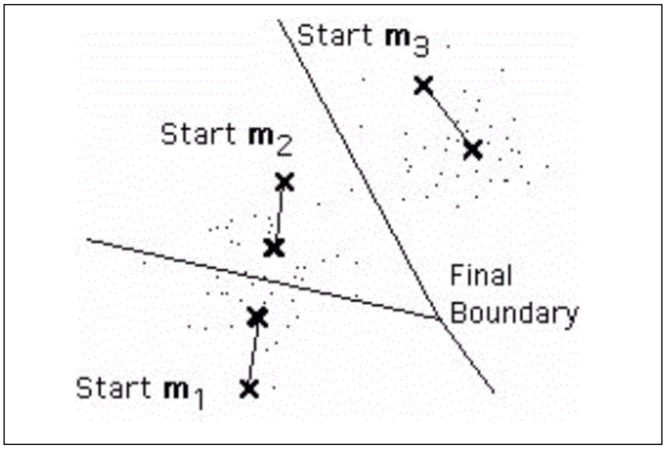
2.3 *K-Means Clustering*

The most popular nonhierarchical clustering method is the K-means algorithm. The “K” in the name is derived from the fact that for a specific execution the analyst chooses the number of clusters (K) to be fit. The “means” portion of the name refers to the fact that the mean of observations (on the selected fields) in a cluster represent a cluster. Thus, this technique is based on geometric notions of similarity, which makes it much faster than the Kohonen network and gives the analyst a measure of distance from each record to the cluster center, and from each cluster to the others. Although there are many possible methods to calculate distance, K-means in PASW Modeler uses Euclidean distance.

As noted above in the section on Number of Clusters, you typically request solutions with a different number of clusters and then evaluate the results to find the most useful.
Based on simulation studies, K-means clustering is effective when the starting cluster centers are well spaced (and it is much less effective when this is not true). Given this, the implementation of K-means in PASW Modeler uses the “maximin” method to first select cluster centers. The position of the first cluster center is simply the first record in the data file. The remaining centers (up to K) are also created from actual data records, but by searching for positions in n-dimensional space that are as far as possible from any other cluster center previously created. This means that the initial cluster centers will cover as large a data range as possible. (As a further step at validation of K-Means models, the data file is often randomly sorted; then another clustering solution with new cluster centers is rerun to see whether it is comparable.)

In a second stage, the Euclidean distance is calculated between each record and every cluster center. Records are assigned to whichever cluster center is the smallest squared Euclidean distance away. After all the cases have been assigned to a cluster group, the location of each cluster center is then recalculated to be the average of all the cases within the cluster. This may have the effect of moving the cluster center quite dramatically. This process is repeated, with all cases reassigned to their respective nearest cluster center once again, and cluster centers recalculated. This process is reiterated until one of the stopping criteria is reached, by default either a change in means or a number of iterations.
Figure 2.1 provides a graphic illustration of the process in a small dataset with only two input fields. For a three-cluster solution, three widely spaced records are chosen in this two-dimensional space. Then distances are calculated and the cluster centers recalculated accordingly, which causes them to move through the two-dimensional space to their final positions. The solid lines then depict the boundaries between clusters.

Figure 2.1 K-Means Clustering Technique Illustrated



2.4 *The K-Means Node*

The K-means node is used to create a K-means clustering model and can be found in the Modeling palette within PASW Modeler. Once trained the result will be a K-Means generated model node, which will be held in the Models manager. This node represents the final cluster solution and can be browsed.

As with the Kohonen network node, all data must be fully instantiated before modeling takes place and all fields that are to be used in the clustering process must have their role set to INPUT (or be selected on the Fields tab of the K-Means dialog). Fields to be excluded in the clustering require their role to be set to NONE. There are no output fields in a cluster analysis.

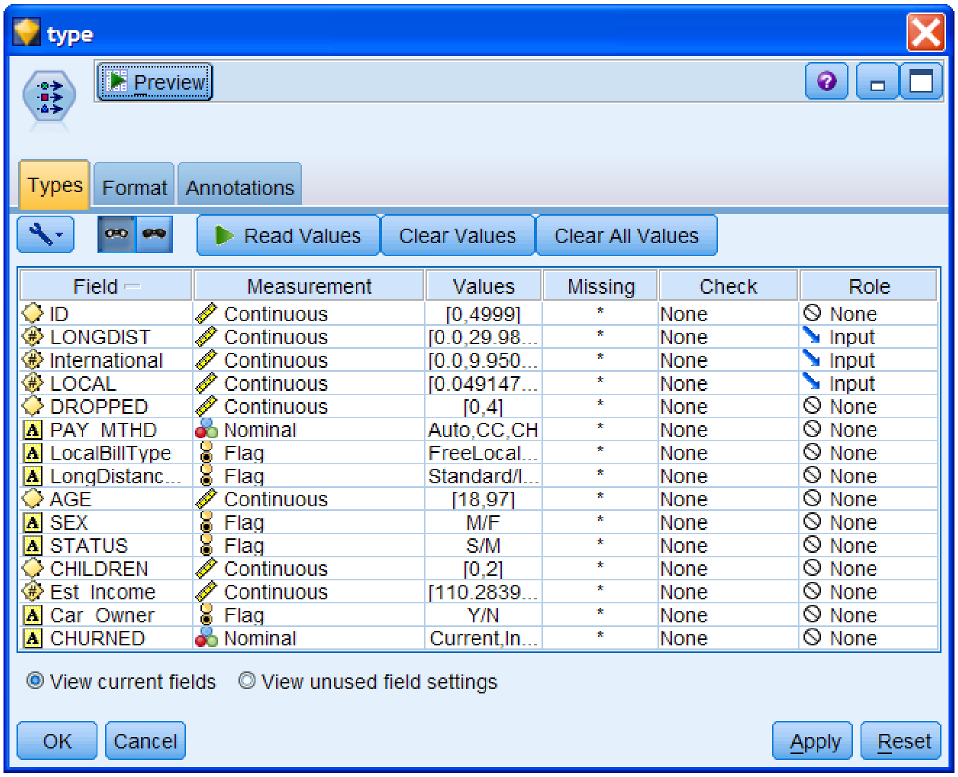
In this lesson we will use the data file *churn.txt*. We will attempt to find natural segments, or clusters, of customers to see whether they can be targeted for different promotions and to explore if cluster differences relate to customer status. Here we will cluster the entire file rather than maintaining separate training and validation samples (although this could be done for validation purposes).

We will use a pre-existing stream file to begin our analysis.

Open the stream file **Kmeans.str** in the c:\Train\ModelerClusAssoc directory
Run the **Table** node
Examine the data and close the **Table** window
Edit the **Type** node

As shown in Figure 2.2, only three fields have role INPUT, corresponding to the amount of time spent on long distance, international, and local telephone calls, in minutes. There are no missing (blank) data in this file, but we will discuss how K-means handles such data later.

Figure 2.2 Type Node Setup for K-Means Clustering



Field	Measurement	Values	Missing	Check	Role
ID	Continuous	[0,4999]	*	None	None
LONGDIST	Continuous	[0.0,29.98...	*	None	Input
International	Continuous	[0.0,9.950...	*	None	Input
LOCAL	Continuous	[0.049147...	*	None	Input
DROPPED	Continuous	[0,4]	*	None	None
PAY MTHD	Nominal	Auto,CC,CH	*	None	None
LocalBillType	Flag	FreeLocal...	*	None	None
LongDistanc...	Flag	Standard/I...	*	None	None
AGE	Continuous	[18,97]	*	None	None
SEX	Flag	M/F	*	None	None
STATUS	Flag	S/M	*	None	None
CHILDREN	Continuous	[0,2]	*	None	None
Est Income	Continuous	[110.2839...	*	None	None
Car Owner	Flag	Y/N	*	None	None
CHURNED	Nominal	Current,In...	*	None	None

● View current fields ○ View unused field settings

Selection of Fields for Clustering

In this example we will be using fields that have the same scale, i.e., minutes of time. This certainly isn't required for a clustering solution, but the selection of the input fields is an important decision. Unfortunately, very little advice is given about this in most references on clustering, where is it is often assumed that the fields to be included will be obvious.

In typical data mining applications, however, dozens if not hundreds of fields are available for clustering, so the decision is anything but obvious. So, for example, in the churn data demographic fields are included, such as age, sex, income, and number of children. Also included are nominal fields and flags recording payment methods, number of dropped calls on the mobile phone, and bill type for local and long distance. All of these fields could be included in a clustering solution, but then the investigation of the clusters would be more time-consuming and complicated.

The selection of fields should hinge upon the natural structure we expect in the data, the goal(s) of the analysis, and the general criterion that the clusters should be as simple as possible. In most instances, demographic fields are not used for clustering but instead to validate and explore a clustering solution. Demographic fields are available in most data files and thus could be used for clustering, but usually they are of secondary interest compared to more specific measures unique to a particular problem.

Fields that measure essentially the same concept and thus are highly related should not all be used. If three fields that are highly correlated (e.g., a correlation coefficient above about .80 in absolute value) are included, this will effectively weight the value of this field three times, which is not usually desirable (note the Statistics node in PASW Modeler can produce correlation coefficients). We don't have this problem in the current data.

The churn data concern mobile phone usage, so it makes sense that we should use the seven fields that measure something about mobile phone usage for clustering. It would be perfectly reasonable to use all these fields, but we should first ponder what natural clusters of customers might exist, and what fields these would represent. Clusters may well be related to the type and amount of calls made with the mobile phone, such that some customers make only local calls, others make relatively more long distance or international calls, and some customers make many calls of all types, and so forth. It is possible that there are also clusters by bill type, but this seems less likely, as does using dropped calls to define clusters (we might expect this should simply relate to the total number of minutes of usage). And we don't want to use the field *CHURNED* because that's what we wish to (eventually) predict.

So based upon this reasoning and acknowledging there are other possibilities, we use only the three fields recording phone usage.

K-Means Node Settings

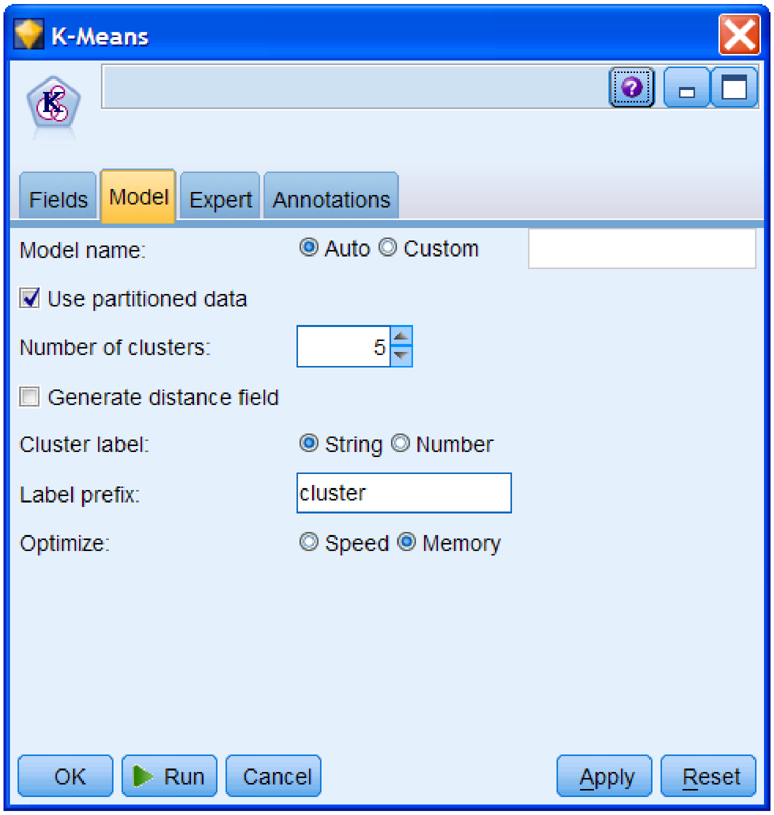
We next review the various options and settings in the K-Means node.

Close the Type window
Select a **K-Means** node, place it on the Stream canvas, and attach it to the **Type** node
Edit the **K-Means** node

The default Number of Clusters is set to 5, which means that the K-means model is going to return a 5-cluster solution. This number can be increased or decreased accordingly per the discussion above. We don't have any good sense for the number of clusters in the churn data, so we will retain the default value.

The *Generate distance field* option, if checked, will create a field in the model node called *$KMD-K-Means*, which shows the distance between a particular record and its respective cluster center. Note, that the generated model node will always create a field named *$KM-K-Means*, recording the cluster membership for each record.

Figure 2.3 K-Means Dialog



Model name: ● Auto ○ Custom

☑ Use partitioned data

Number of clusters: 5

□ Generate distance field

Cluster label: ● String ○ Number

Label prefix: cluster

Optimize: ○ Speed ● Memory

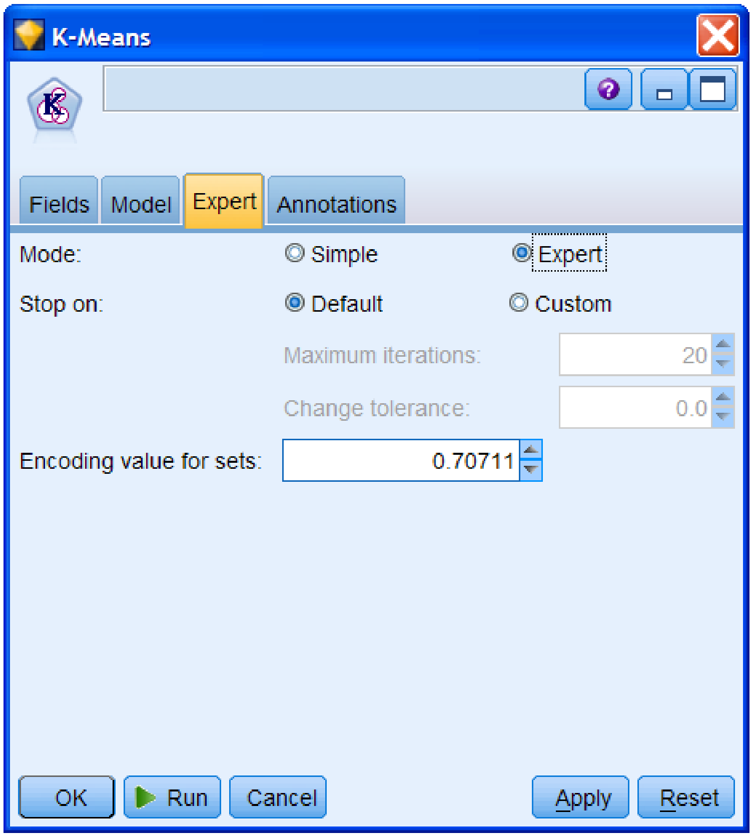
The *Cluster label* option allows you to decide whether the data values in the cluster membership field are to be coded as strings or numbers. The type of coding to choose depends on how you plan to use the field in analyses.

By default the algorithm is optimized for memory usage, but you can check the *Speed* option button to instruct K-Means to perform all its work in memory, thus increasing speed (assuming enough memory is available).

Click the **Expert** tab
Click the **Expert** button

The clustering algorithm stops iteratively recalculating the cluster centers once the stopping criteria have been reached. The *Stop On:* group allows you to specify the stopping criteria. Iterations can be terminated once the sum of all change in movement of cluster centers (*Change tolerance*) is less than a specified threshold (by default .000001) or alternatively when the algorithm has performed a specified number of Iterations (by default 20). The default combines both the change and iterations so that training stops when either the change threshold or iteration number is reached. The defaults are usually satisfactory, but if not you might first try increasing the number of iterations. You must be sure the model has converged to a solution when examining the K-Means model output.

Figure 2.4 K-Means Expert Options



Mode: ○ Simple ● Expert

Stop on: ● Default ○ Custom

Maximum iterations: 20

Change tolerance: 0.0

Encoding value for sets: 0.70711

Calculating Distances in K-Means

To understand the *Encoding value for sets* option available in K-Means we must consider how data are coded.

As mentioned, the Euclidean distance is used by K-Means to calculate distances. This can cause a problem when fields with widely differing ranges and standard deviations are used. For example, if we use the three fields recording phone usage plus the number of dropped calls in a clustering solution, we would likely find that the phone usage variables dominate the solution. This is because the dropped call field varies from only 0 to 4, while the number of minutes of local calls can range from small values to more than 100. Distances calculated directly from these values will clearly be influenced most strongly by the usage fields.

In order to avoid this, the data for K-Means are standardized before clustering begins. Moreover, since the algorithm must have numeric data to calculate distances, categorical data must be transformed into a numeric equivalent. Fields of continuous measurement are transformed to a scale of 0 to 1 using the following formula:

$$\text{New value} = \frac{(\text{Value - Lower bound})}{\text{Range}}$$

Flag fields are coded such that the false value=0 and the true value=1. For categorical fields, each category will have a new temporary input field assigned to it, coded as either 0 or 1 (a dummy variable). Thus, a categorical field with three values will have three new inputs. For example, suppose you have a field that records payment method, where there are 3 types of payment—cash, check or credit card. If a customer pays by credit card this will be represented as (0 0 1) in the three inputs; alternatively, payment by check would be represented by (0 1 0). This form of coding requires one temporary input field for each distinct value of a categorical field.

However, if the new inputs are coded with a value of 1, they will tend to dominate the cluster solution compared to numeric fields. Accordingly, the value of .70711 is used instead (the square root of 1/2).

The *Encoding value for sets* value can be set between .0001 and 1.0, inclusive. Values below .70711 will decrease the importance of categorical and flag fields, while values above that will do the reverse.

Flag fields are not encoded in this manner, as doing so may distort the distances between records. They are given values of 0 and 1.

If categorical fields are included in a clustering solution, we recommend leaving the encoding value at its default setting unless you have a good reason to change the influence of these fields.

We will now run K-Means.

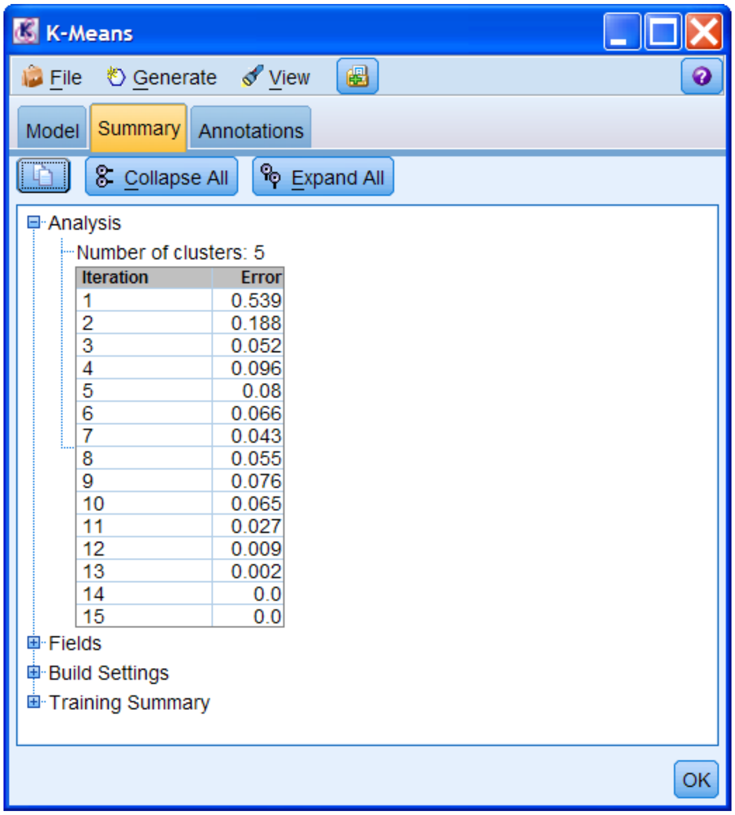
Run the **K-Means** node

Browsing the K-Means Results

The first step in studying a cluster solution is to browse the generated model to look at the cluster characteristics.

After the model has run, **Browse** (right-click, then click Browse) the **K-Means model** node in the Models manager
Click the **Summary** tab

Figure 2.5 Model Summary from K-Means Clustering



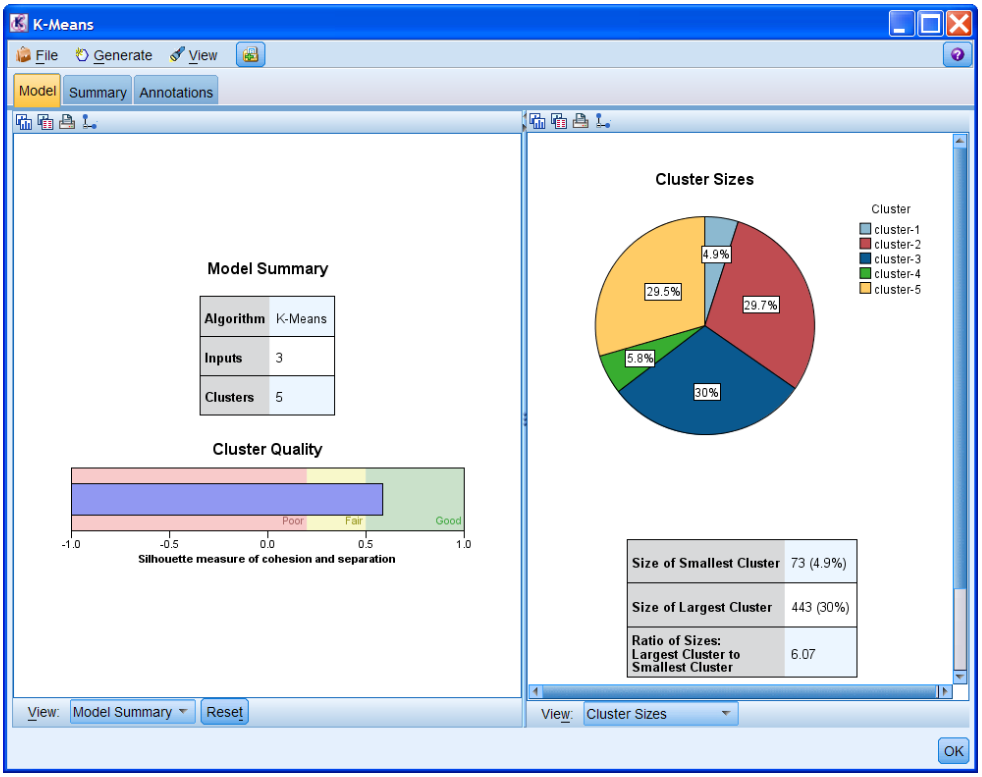
Number of clusters: 5

Iteration	Error
1	0.539
2	0.188
3	0.052
4	0.096
5	0.08
6	0.066
7	0.043
8	0.055
9	0.076
10	0.065
11	0.027
12	0.009
13	0.002
14	0.0
15	0.0

We see that five clusters were created, as requested. Fifteen iterations were required before the error, or change in means, from one step to the next decreased below .000001. So far so good!

Click **Model** tab

Figure 2.6 Model Information from K-Means Analysis



Model Summary

Algorithm	K-Means
Inputs	3
Clusters	5

Size of Smallest Cluster	73 (4.9%)
Size of Largest Cluster	443 (30%)
Ratio of Sizes: Largest Cluster to Smallest Cluster	6.07

The Model tab for cluster models shows a graphical display of summary statistics and distributions for fields between clusters; this is known as the Cluster Viewer. The model information is shown in two panels, the main view on the left and the linked, or auxiliary, view on the right, displaying the summarized and detailed information respectively.

In the left (main) panel we see our five clusters are induced by three input variables, as set in the Type node, and the cluster quality falls in the “Good” range.

In the right panel we see a pie chart that illustrates the relative size of each cluster. The smallest cluster has 73 records, or about 5% of the file, which is verging on being too small to be useful. The largest cluster has 443 records. The ratio of sizes of the largest cluster to the smallest cluster is 6.07.

2.5 *Exploring the Cluster Profiles*

More options are provided to explore the cluster profiles in a visual way.

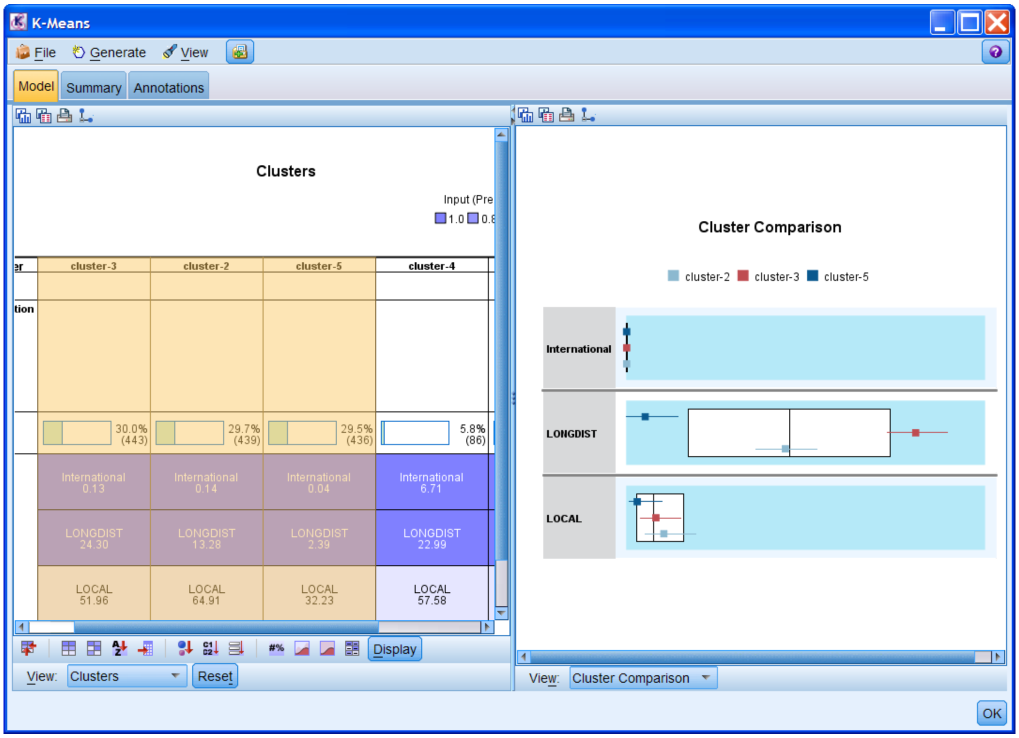
Click **Clusters** on the View drop-down list (in the left panel)

By default the overall importance and cluster centers of each input (feature) in each cluster are displayed. For numeric input fields (features), cluster centers are means, and for categorical inputs (no categorical inputs were included in our analysis), cluster centers are the percentages of records in each cluster relative to the whole file.

Click **Cluster Comparison** on the View drop-down list (in the right panel)

Click **clusters (columns in the grid)** in the main panel while holding Ctrl to select cluster-2, cluster-3, cluster-5.
Use either Ctrl-click or Shift-click to select or deselect more than one cluster for comparison.

Figure 2.7 K-Means Cluster Profiles in Cluster Viewer



Clusters

er	cluster-3	cluster-2	cluster-5	cluster-4
tion				
	30.0% (443)	29.7% (439)	29.5% (436)	5.8% (86)
	International 0.13	International 0.14	International 0.04	International 6.71
	LONGDIST 24.30	LONGDIST 13.28	LONGDIST 2.39	LONGDIST 22.99
	LOCAL 51.96	LOCAL 64.91	LOCAL 32.23	LOCAL 57.58

The Cluster Comparison view consists of a grid-style layout, with features in the rows and selected clusters in the columns. This view helps you to better understand the factors that make up the clusters; it also enables you to see differences between clusters not only as compared with the overall data, but with each other.

We can now easily compare the clusters. We see that three of the clusters (2, 3, and 5) have the majority of cases. Cluster 5 includes customers who use their phone very little, as they have the smallest mean minutes of usage for all three input fields. Customers in cluster 3 use more long distance minutes, while those in cluster 2 are similar but use fewer long distance minutes and more local minutes.

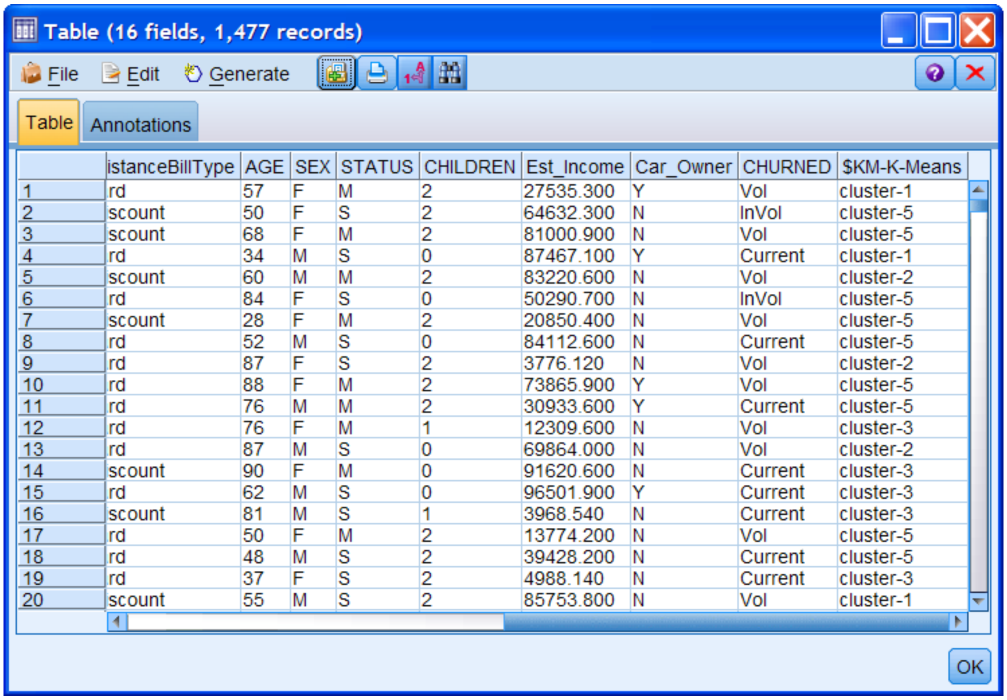
It is too early to tell whether this cluster solution is useful, as it needs to be examined in terms of the goals of the data-mining project as well as business and organizational knowledge about mobile phone customers.

As noted previously, the K-Means model creates a new field, recording the cluster membership (two if the *Generate distance field* option is checked). Let's view it.

Close the K-Means generated model window
Place another **Table** node in the Stream canvas to the right of the **K-Means generated model** node
Connect the **K-Means generated model** node to the **Table** node
Run the **Table** node
Scroll to the **right** in the Table window

Each record has a value for the new field $KM-K-Means, which records cluster membership.

Figure 2.8 Table Showing Cluster Field Created by K-Means Model Node



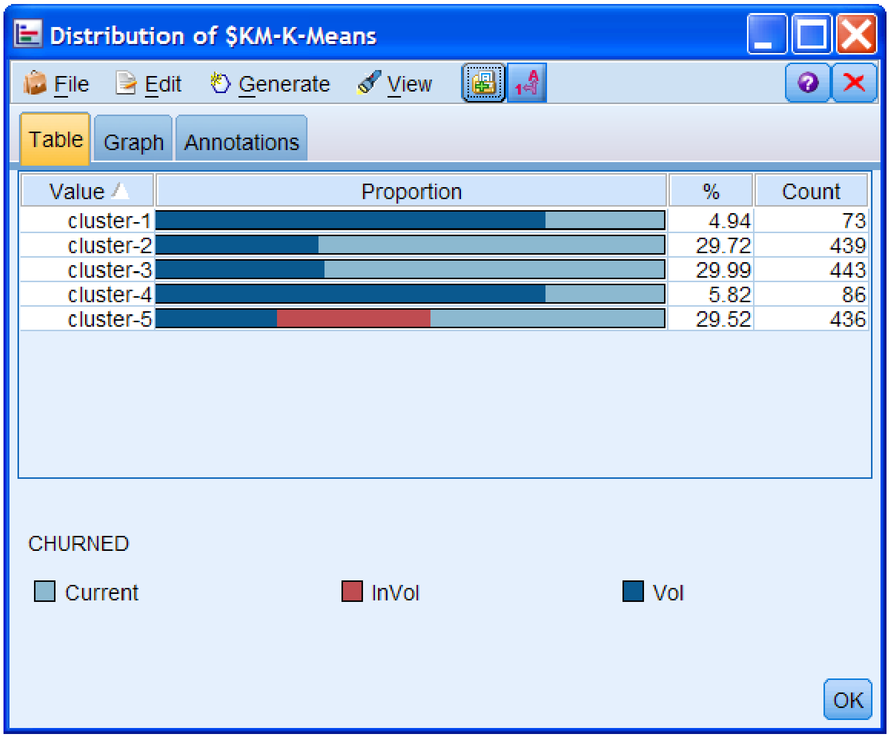
	istanceBillType	AGE	SEX	STATUS	CHILDREN	Est_Income	Car_Owner	CHURNED	$KM-K-Means
1	rd	57	F	M	2	27535.300	Y	Vol	cluster-1
2	scount	50	F	S	2	64632.300	N	InVol	cluster-5
3	scount	68	F	M	2	81000.900	N	Vol	cluster-5
4	rd	34	M	S	0	87467.100	Y	Current	cluster-1
5	scount	60	M	M	2	83220.600	N	Vol	cluster-2
6	rd	84	F	S	0	50290.700	N	InVol	cluster-5
7	scount	28	F	M	2	20850.400	N	Vol	cluster-5
8	rd	52	M	S	0	84112.600	N	Current	cluster-5
9	rd	87	F	S	2	3776.120	N	Vol	cluster-2
10	rd	88	F	M	2	73865.900	Y	Vol	cluster-5
11	rd	76	M	M	2	30933.600	Y	Current	cluster-5
12	rd	76	F	M	1	12309.600	N	Vol	cluster-3
13	rd	87	M	S	0	69864.000	N	Vol	cluster-2
14	scount	90	F	M	0	91620.600	N	Current	cluster-3
15	rd	62	M	S	0	96501.900	Y	Current	cluster-3
16	scount	81	M	S	1	3968.540	N	Current	cluster-3
17	rd	50	F	M	2	13774.200	N	Vol	cluster-5
18	rd	48	M	S	2	39428.200	N	Current	cluster-5
19	rd	37	F	S	2	4988.140	N	Current	cluster-3
20	scount	55	M	S	2	85753.800	N	Vol	cluster-1

We won’t spend much more time studying the cluster solution here, but let’s request a bit more output to illustrate how to explore a cluster model. First, it will be useful to examine the relationship between cluster membership and the field *CHURNED*.

Close the **Table** window
Place a **Distribution** node in the Stream canvas near the **K-Means** generated model node and connect them
Edit the **Distribution** node (not shown)
Select **$KM-K-Means** in the **Field** box and **CHURNED** in the **Overlay Color** field box
Select the **Normalize by color** checkbox
Click **Run**

We see that, interestingly, all involuntary churners—those customers dropped by the company—are in cluster 5. This cluster was composed of customers with generally low usage. That all of them fall into one cluster may be a tangible indicator that this cluster solution is useful. Thus, we may find that predicting which customers will need to be dropped (because of late bill payment or other problems) will be more accurate by including cluster membership in a model.

Figure 2.9 Distribution of CHURNED by Cluster Membership



Value	Proportion	%	Count
cluster-1		4.94	73
cluster-2		29.72	439
cluster-3		29.99	443
cluster-4		5.82	86
cluster-5		29.52	436

CHURNED

□ Current 🟥 InVol ■ Vol

Voluntary churners, often the most critical customer group, tend to concentrate in clusters 1 and 4, which are also the smallest clusters. This may not be satisfactory and thus may require, despite the argument above, a different cluster solution. Involuntary churners are exclusively associated with Cluster 5, the set of customers who use the phone very little and therefore are apparently dropped by the company.

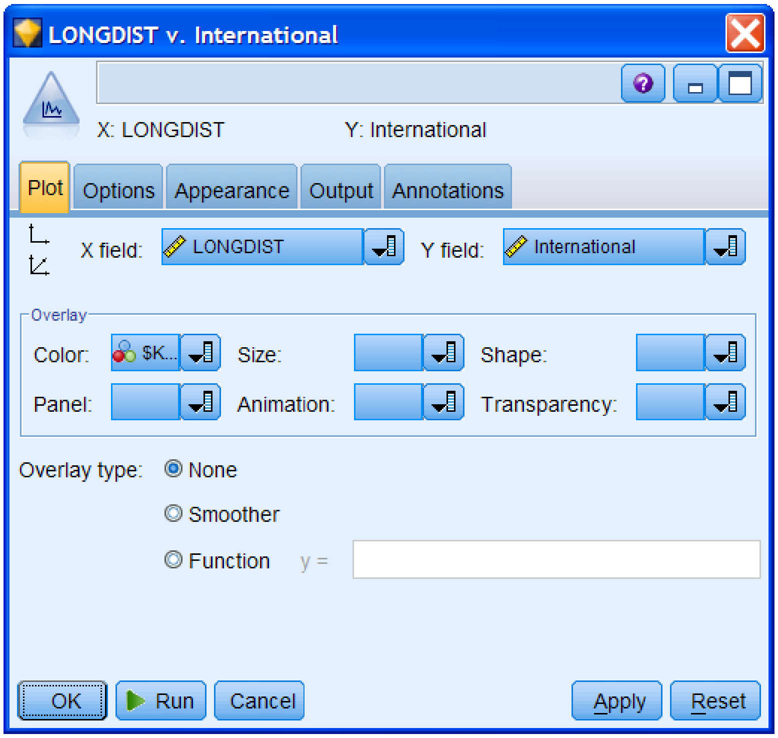
You would normally continue using a Distribution node or the Data Audit node with the cluster membership field as an overlay to examine how other fields relate to the cluster membership. To complete our exploration, we will plot the values of two of the phone usage fields by cluster membership.

When there are only a few fields used in clustering, and they are continuous measurement, a scatterplot can be useful to visualize the clustering solution.

Close the **Distribution graph** window
Place a **Plot** node in the Stream canvas near the **K-Means** model node and connect them
Edit the **Plot** node
Select **LONGDIST** as the **X field**, **International** as the **Y field**, and **$KM-K-Means** as the **Overlay Color** field

Although we choose to represent the cluster membership field (*$KM-K-Means*) as a color field (one plot for different colors representing the different clusters), we could have specified it as an overlay panel (one plot for each cluster), size, or shape field. We could include the Local field by choosing a 3-D plot.

Figure 2.10 Plot Node Dialog



LONGDIST v. International

X: LONGDIST Y: International

X field: LONGDIST Y field: International

Overlay

Color: $K... Size: Shape:

Panel: Animation: Transparency:

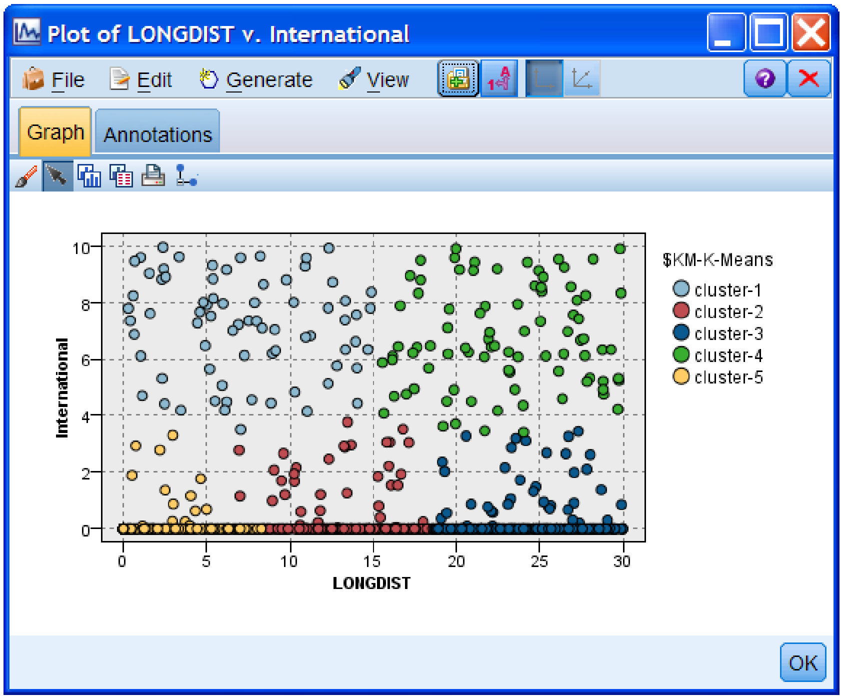
Overlay type: ● None ○ Smoother ○ Function y =

Run the **Plot** node

Because there are only three fields used to create the clusters, and because long distance and international usage are so important in the cluster solution, the plot graphically displays the cluster memberships with little overlap. There is little overlap among the groups. In other words, from this plot it appears that only the *LONGDIST* and *International* fields are necessary to create an equivalent cluster solution. We see graphically that Cluster 5 is composed of customers who have relatively low usage on both fields, while cluster 4 contains customers who have high usage on both types of calls. If you refer back to Figure 2.6, you can see why cluster 1's distance, or proximity, was greatest to cluster 3.
We can also observe how tightly, or loosely, the clusters are grouped, i.e., how restricted their range is in the two dimensions. Clusters 2 and 5 are especially tightly grouped (in the bottom left half of the plot), as compared to clusters 1 and 4. Note how the data are spread throughout the two-dimensional space, with no one cluster truly well-separated from any other. Remember that we got a five-cluster solution because that is what we requested, not because the data naturally have five separate groupings. This is why other cluster solutions should be requested when running K-Means.

A scatterplot also allows you to find those cases that almost fall into another cluster (outliers) and investigate their characteristics. In general, this type of plot is extremely useful, but again, you need numeric variables as inputs and, most probably, either only a small number of input fields or a few that are dominant in the clustering solution.

Figure 2.11 Plot of Long Distance and International Usage With Cluster Membership



Missing Data with K-Means

In the K-means node, blanks are handled by substituting "neutral" values for the missing ones. For continuous and flag fields with missing values (blanks and nulls), the missing value is replaced with 0.5 (for continuous fields, this is done after the original values have been transformed into a 0-1 range). Continuous field values below the lower bound (in the last Type node) will be set to the lower bound, and values above the upper bound will be set to the upper bound value. For categorical fields, the derived indicator field values are all set to 0.0. Since this is done automatically, you should check your data using the Data Audit node and decide whether fields or records with much missing data should be removed (filtered, selected) prior to analysis, or given substitute values.

Before turning to clustering with a Kohonen network, let's save the current stream for use in the exercises.

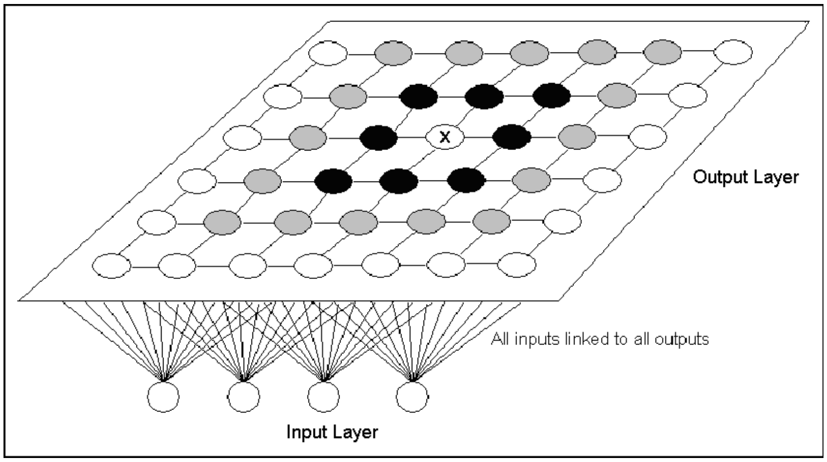
Close the **Plot** window
Click on **File…Save Stream As**
Save the stream with the name **Lesson2_Kmeans**

2.6 *Clustering with a Kohonen Network*

Kohonen networks were first developed in the early 1980s (yes, by a fellow named Kohonen). They are a type of neural network that is based upon the idea of a self-organized map, a form of unsupervised learning. This means that there is no target information that the algorithm is attempting to predict and is thus using to continually adjust its output. Consequently, these networks are ideal for clustering where there is also no output field.

Kohonen networks make the basic assumption that clusters are formed from patterns that share similar features, and so they will group similar patterns (records) together. The networks consist of either a one or, more commonly, a two-dimensional grid of neurons. Each neuron is connected to each of the inputs, and as with all neural networks, weights (importance) are placed on each of these connections. Each neuron is also connected to the surrounding neurons. The figure below shows the layout of a Kohonen network.

Figure 2.12 Representation of a Kohonen Network



The network trains by presenting records, one by one, to the grid. A record’s characteristics are compared with those of all the neurons in the grid, which are initially given random weights. The neuron with the pattern most like that of the input “wins” that particular record. This causes the weights of the artificial neuron to be adjusted to be more similar to that of the record it has just won. Therefore, if another record with similar characteristics is shown to the network, the same node will have a better chance of winning that record as well. The network also slightly adjusts the weights of the surrounding neurons so that they, too, should also attract records of a similar input profile. In the figure, different shading of the nodes represents this.

The result, after the data have passed through the network a number of times, will be a map containing clusters of records corresponding to different types of patterns in the data. And those patterns that are similar should be closer together in the map than patterns that are dissimilar, just as we saw in the scatterplot above in Figure 2.11 (although you should not think of a Kohonen network as a type of scatterplot).

The Kohonen algorithm is a two-part procedure. There is an initial phase of large-scale changes and then a second phase of more fine-tuning with smaller changes in the weights. Hence, when controlling how a Kohonen network is to be trained it is necessary to specify learning rates for both phase 1 and phase 2 (as we discuss below).

A network stops training either when a certain number of cycles have been executed or when the change in weights from one cycle to the next becomes very small.

If the records in the data file are ordered in a particular manner it is possible that the same node could win the first n records, hence changing not only the weights of its own node but of those surrounding it as well. This is not desirable because it would lead to a state where one node has so much influence that it attracts all incoming records and dominates the clustering process. In order to combat this, the Kohonen network has a dynamically changing neighborhood influence, so that a node’s influence on surrounding nodes is decayed over a number of cycles, thus reducing its dominance.

Given that training of the network involves much iteration, each with weight adjustments, Kohonen networks typically take longer to train than K-means or TwoStep clustering solutions. Still, Kohonen networks provide a different and potentially valuable view of groupings in the data. It isn't correct to claim that one of these clustering algorithms is generally better than another. Analysts do try several methods on the same data.

As a reminder, the discussion earlier in the lesson about the number of clusters, selecting fields, and validation apply equally to clusters created with a Kohonen network.

2.7 *The Kohonen Node*

The Kohonen node creates a Kohonen network in PASW Modeler and places a generated Kohonen net node in the Models manager. We will use an existing stream to investigate the expert options for a Kohonen network, continuing with the churn dataset.

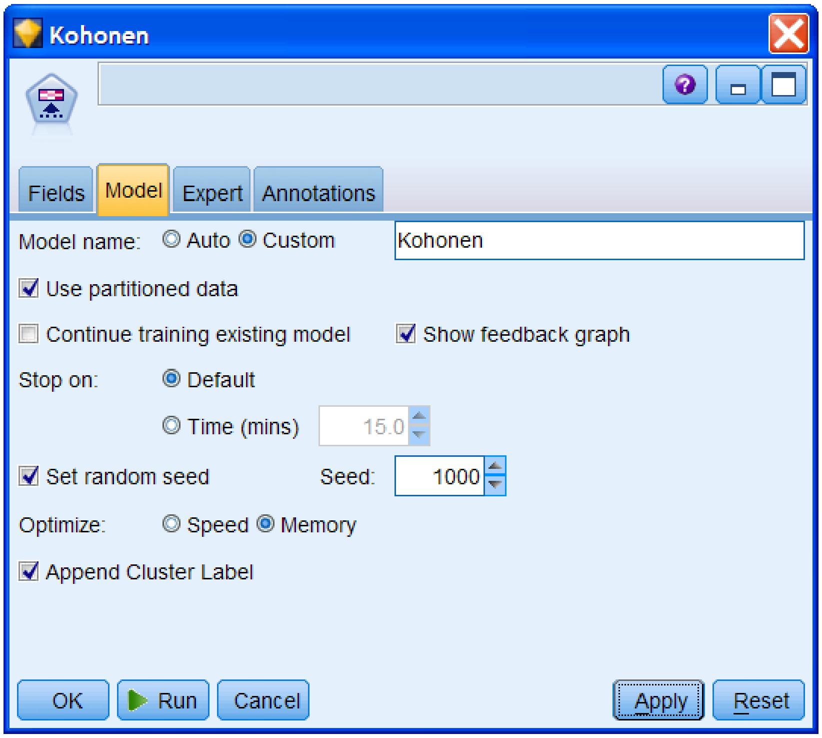
Close the K-Means stream
Open the stream file **Kohonen.str**
Run the **Table** node
Close the **Table** window
Edit the **Kohonen** node

As with the K-Means node, the Kohonen node requires fully instantiated field types. We will use the same three input variables that record phone usage so we can directly compare the clustering solution here to that from K-Means.

When using a Kohonen node you will probably use the expert options because the default number of clusters is set rather large. Before we delve into the various options, note that the random seed can be set, just as in the other neural-network modeling node (Neural Net) in PASW Modeler. This is because there is a random component to any neural network due to the need to initialize the weights to some small, nonzero values. The weights are initialized using random numbers generated from the random seed. Thus, to replicate an analysis, you can set an identical random seed. Usually, though, you let the seed vary to see whether the same basic cluster solution is discovered. We can try setting the random seed so everyone gets the same cluster solution.

Click the **Set random seed** checkbox and set the seed value to **1000**

Figure 2.13 Kohonen Network Dialog (Model Tab)



Model name: ○ Auto ● Custom Kohonen

☑ Use partitioned data

☐ Continue training existing model ☑ Show feedback graph

Stop on: ● Default

○ Time (mins) 15.0

☑ Set random seed Seed: 1000

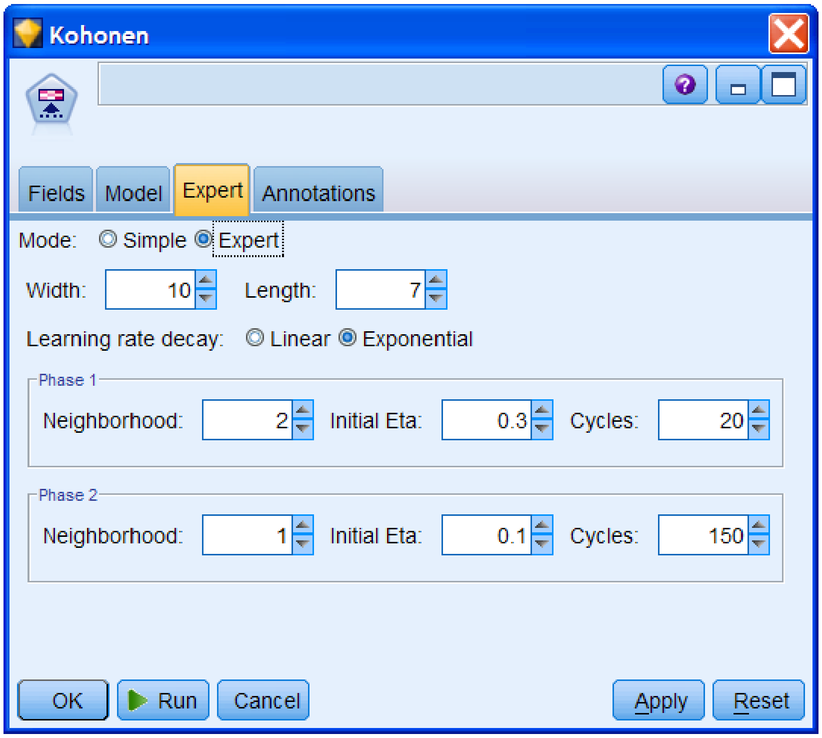
Optimize: ○ Speed ● Memory

☑ Append Cluster Label

Next we explore the expert options.

Click **Expert tab**, then click **Expert** Mode option button

Figure 2.14 Kohonen Network Dialog (Expert Tab)



Mode: ○ Simple ● Expert

Width: 10 Length: 7

Learning rate decay: ○ Linear ● Exponential

Phase 1

Neighborhood: 2 Initial Eta: 0.3 Cycles: 20

Phase 2

Neighborhood: 1 Initial Eta: 0.1 Cycles: 150

Number of Clusters

The *Width* and *Length* settings allow you to specify the dimensions of the Kohonen output grid. In Expert mode, the default setting for this is 10 by 7, which will effectively create 70 nodes in the output grid (although not all may be used; that is, not all may contain records). This means that the final model will yield a solution with many clusters, and given our advice above about the suitable number of clusters to create, this is far too many to deal with effectively. (Note that in Simple mode, the grid size chosen for this problem is considerably smaller.) Consequently, the number of nodes in the output grid should be reduced to a more manageable number. The geometry of the grid can make a bit of difference in the characteristics of clusters created, but usually has much less of an effect than the input fields themselves and the total number of neurons, so whether you request a 4 by 3 or a 3 by 4 grid to get 12 clusters shouldn't make much difference.

The width can vary between 2 and 99, while the length can vary between 1 and 99. To try to reproduce the five-cluster solution from before, we modify the width and length. Here we request only five output nodes in an attempt to match the K-means analysis. In practice, an analyst would almost certainly try a larger output grid (say a 4 by 4 or larger), expecting that some nodes would capture few records.

Change the **Width** to **5**
Change the **Length** to **1** (not shown)

Learning Rates

As with the neural network methods described earlier in the course, the Kohonen network has some of the same learning parameters, including eta and *Learning Rate Decay* (eta decay). They have default values, as depicted in Figure 2.14. Thus *Initial Eta-Phase 1*, which refers to the learning rate, begins at a value of 0.3, the same as in the Neural Net node. The eta value decays exponentially by default. The decay can be set to linear, which reduces the decay less quickly than an exponential decay.

Note that the options are separated into two phases, and that there is an eta in each phase. The first phase of training is the coarser mapping in which larger changes occur in the weights. The learning rate, or change in weights in this phase, starts at the value of *Initial Eta* (*Phase 1*) and decreases to the value of low eta, which is the lesser of the *Initial Eta* values in the two phases. The training proceeds for the number of iterations requested with *Cycles* (*Phase 1*).

Each neuron, as it wins a record, influences the weights of its neighbors. The number of nodes around the "winning" node that are updated with changes is specified in *Neighborhood* (*Phase 1*). The neighbors are defined symmetrically around the winning node, as depicted in Figure 2.12. The number of neighbors updated is decreased to the value of *Neighborhood* (*Phase 2*) +1. By default, this value is equivalent to that for *Neighborhood* (*Phase 1*), so no decrease occurs in phase 1.

In the second phase of training, finer adjustments are made to the weights, and the learning rate here is controlled with *the Phase 2* settings.

The learning rate begins at *Initial Eta* (*Phase 2*), which should, of course, be smaller than *Initial Eta* (*Phase 1*), and decreases to 0. Training proceeds for the number of iterations specified by the value of *Cycles* (*Phase 2*), which is typically larger than *Cycles* (*Phase 1*) to allow the algorithm to converge even when making small updates to the weights.

Neighborhood (*Phase 2*) defines the boundary of nodes that are updated in this second phase, again symmetrically around the winning node. By default this includes the nodes immediately surrounding the winning node, as the default value of *Neighborhood* (*Phase 2*) is 1. The number of neighbors is decreased to the value of 1 as training proceeds, which means that no decrease occurs with this default setting. This decrease in the number of neighbors updated from phase 1 to phase 2 reflects the change in influence of a node on its neighbors over time to reduce the node's dominance, as explained earlier.

The *Initial Eta* values can be between 0 and 1, and the *Neighborhood* values between 1 and 10.

We provide some general advice for these settings.

- In small grids, the value of *Neighborhood* (*Phase 1*) shouldn't be set higher than the default of *Neighborhood* (*Phase 2*); otherwise the whole grid will be affected.
- Don't change the *Cycles* settings unless you get an odd-looking solution; they should be large enough for almost all circumstances, except for unusual data or a large number of clusters.
- The *Initial Eta* settings are the most likely place to begin to modify a network, and you would normally set *Initial Eta* (*Phase 1*) a bit higher, and perhaps *Initial Eta* (*Phase 2*) as well.
- If you expect to find one dominant cluster, leave the *Learning decay rate* as *Exponential*. If not, you can try using a linear decay.

We'll now run the Kohonen network.

Click **Run**

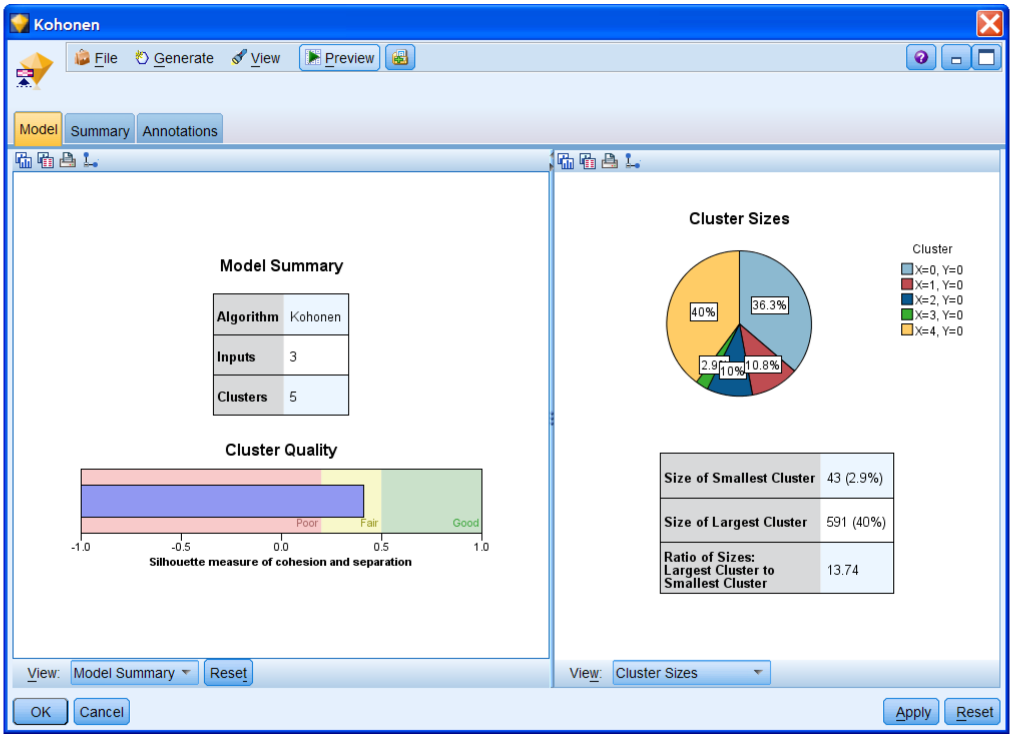
Once the Kohonen node has finished training, the feedback graph disappears (or never appeared because this data file is relatively small) and a Kohonen generated model node appears in the Models manager. Browsing this node yields profile information on the clusters. A single cluster membership field and the distances between clusters are not produced, since these are not part of the Kohonen network algorithm (a cluster membership field can be created, as we will see shortly).

Exploring the Cluster Solution

To explore the clustering solution, we'll begin with the cluster viewer, and then create a cluster membership field and do some additional analysis.

Edit the **Kohonen generated model** node on the Stream Canvas

Figure 2.15 Kohonen Cluster Profiles in Cluster Viewer



Model Summary

Algorithm	Kohonen
Inputs	3
Clusters	5

Size of Smallest Cluster	43 (2.9%)
Size of Largest Cluster	591 (40%)
Ratio of Sizes: Largest Cluster to Smallest Cluster	13.74

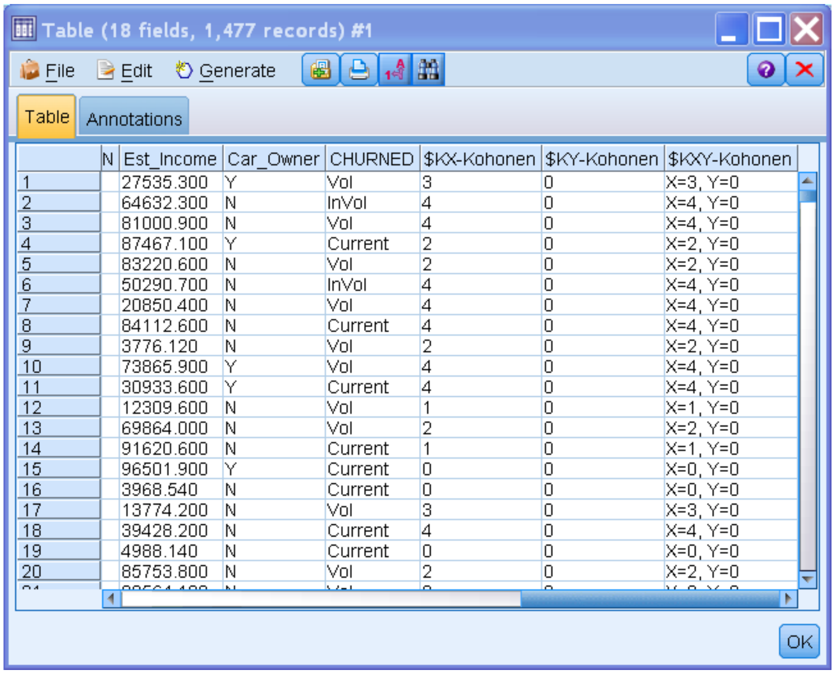
We can see that one of the clusters is very small (X=3, Y=0) and two are very large (X=4, Y=0; X=0, Y=0). (The X and Y values refer to coordinates in the grid used for training, with the X and Y values beginning at 0.) The two largest clusters, accounting for about three quarters of the sample, are larger in size than the three largest clusters in the K-means solution (see Figure 2.7). It remains to be seen whether there is a correspondence between the two solutions.

Does the Kohonen network create clusters similar to those created by K-Means? The first cluster (X=0, Y=0) is comparable to cluster 5 in the K-Means solution, with low usage of the mobile phone for any type of call (and a large number of cases; refer to Figure 2.7). The fifth cluster (X=4, Y=0) is similar to cluster 3 in the K-Means solution. Also, the second cluster (X=1, Y=0) seems very similar to cluster 2 in the Kohonen solution (although much smaller in size). The remaining clusters (X=2, Y=0) and (X=3, Y=0) don’t have exact counterparts in the K-Means clusters. Which cluster solution is better for our purposes has yet to be determined.

Note

If the generated Kohonen model is added to the stream, it will create three variables; *$KX-Kohonen* being the X-coordinate in the grid, *$KY-Kohonen* being the Y-coordinate in the grid, and *$KXY-Kohonen* being the concatenation of the values from the *$KX-Kohonen* and *$KY-Kohonen* coordinate fields. This field records the cluster membership. The figure below shows an example of the fields that will be derived using the generated Kohonen model.

Figure 2.16 Calculating Kohonen Membership Field from Coordinate Fields



	N	Est_Income	Car_Owner	CHURNED	$KX-Kohonen	$KY-Kohonen	$KXY-Kohonen
1		27535.300	Y	Vol	3	0	X=3, Y=0
2		64632.300	N	InVol	4	0	X=4, Y=0
3		81000.900	N	Vol	4	0	X=4, Y=0
4		87467.100	Y	Current	2	0	X=2, Y=0
5		83220.600	N	Vol	2	0	X=2, Y=0
6		50290.700	N	InVol	4	0	X=4, Y=0
7		20850.400	N	Vol	4	0	X=4, Y=0
8		84112.600	N	Current	4	0	X=4, Y=0
9		3776.120	N	Vol	2	0	X=2, Y=0
10		73865.900	Y	Vol	4	0	X=4, Y=0
11		30933.600	Y	Current	4	0	X=4, Y=0
12		12309.600	N	Vol	1	0	X=1, Y=0
13		69864.000	N	Vol	2	0	X=2, Y=0
14		91620.600	N	Current	1	0	X=1, Y=0
15		96501.900	Y	Current	0	0	X=0, Y=0
16		3968.540	N	Current	0	0	X=0, Y=0
17		13774.200	N	Vol	3	0	X=3, Y=0
18		39428.200	N	Current	4	0	X=4, Y=0
19		4988.140	N	Current	0	0	X=0, Y=0
20		85753.800	N	Vol	2	0	X=2, Y=0

Since our Kohonen network is based on a 5 by 1 output layer, we really need only work with $KX-Kohonen. However, the typical output layer has two or more rows and columns, and the field $KXY-Kohonen is appropriate for this more general situation.

Rather than continuing with our Kohonen analysis, we leave that for the exercises. As with the K-Means models, much work remains to be done here, including creating models with a different number of clusters.

As a final bit of advice, it is important to keep in mind that clustering solutions are somewhat arbitrary, and the K-means and Kohonen clustering algorithms will, in general, always create the number of clusters you request. This means that a clustering model should not be accepted without carefully studying the results and, critically for Kohonen and K-means, examining models with different numbers of clusters.

For later use in the exercises, let's save the current stream.

> Close the Cluster Viewer window
> Click on **File…Save Stream As**
> Save the stream with the name **Lesson2_Kohonen**

Missing Data with Kohonen Networks

In the Kohonen node, blanks are handled by substituting "neutral" values for the missing ones. For continuous and flag fields with missing values (blanks and nulls), the missing value is replaced with 0.5 (for continuous fields, this is done after the original values have been transformed into a 0-1 range). Continuous field values below the lower bound (in the last Type node) will be set to the lower bound and values above the upper bound will be set to the upper bound value. For categorical fields, the derived indicator field values are all set to 0.0. This is the same missing data handling as we found in the K-Means node. Since this is done automatically, you should check your data using the Data

Audit node, and decide whether fields or records with much missing data should be removed (filtered, selected) prior to analysis, or given substitute values.

2.8 *TwoStep Clustering*

The TwoStep algorithm is efficient when clustering large data files and uses a statistical criterion to determine the final number of clusters. Thus you may not need to run the analysis several times, as you generally would with the K-Means node and possibly with the Kohonen node. However, even then, it is important to examine the TwoStep solution carefully, and if you are not satisfied, request other solutions with a slightly different number of clusters.

TwoStep performs the cluster analysis with only a single pass through the data. This contrasts to the other cluster algorithms that usually need multiple (K-Means) to many (Kohonen) passes through the data.

The first step of the TwoStep clustering involves a sequential clustering approach. As each record is read, based on a distance criterion, the algorithm decides whether it should be merged with an existing cluster or used to start a new cluster. If the number of clusters becomes too large (the maximum number is set by you), then the distance criterion is increased and clusters whose distance is now less than the modified distance criterion are merged. In this way, the records are clustered into a preliminary set of clusters in a single pass of the data.

Note

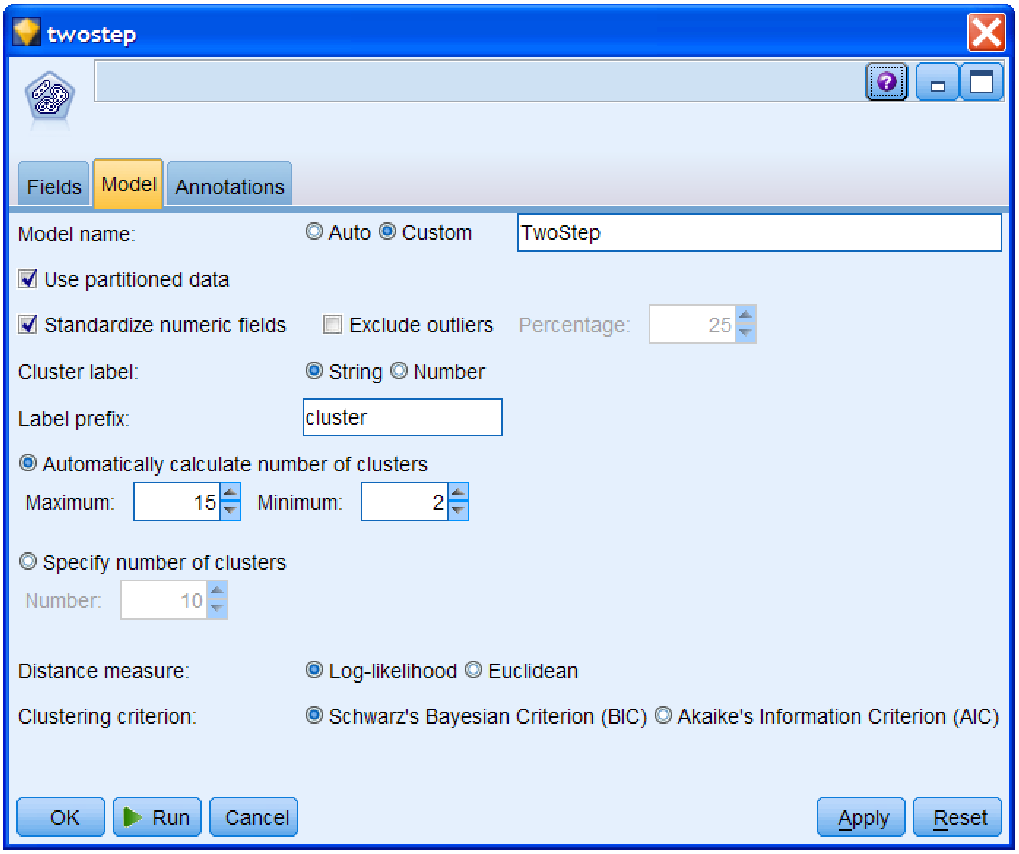
For those interested in the technical details, a modified CF-tree is used to create and store the preliminary clusters (see Zang, T. , Ramakrishnon, R., and Livny, M., 1996. BIRCH: An Efficient Data Clustering Methods for Very Large Databases. *Proceeding of the ACM SIGMOD Conference on Management of Data*, pp. 103-114, Montreal, Canada). At the end of the first step, a limited number of preliminary clusters (continuous fields are represented by their means; categorical fields are represented by their counts in each category) are stored.

The second step takes the preliminary clusters as input and performs a hierarchical cluster analysis (agglomerative). This produces a range of cluster solutions. By default, each is evaluated using a statistical criterion (log likelihood function) that includes a penalty (Bayesian Information Criterion) function based on the number of clusters to prevent solutions with many clusters that are not truly distinct. A similar problem occurs in other forms of model building (decision trees with a large number of branches, statistical models with many parameters, neural networks with many nodes in the hidden layer(s)): as the number of clusters increases, the clusters better match the data, with the best fit being a solution in which each cluster contains a single record. The clusters created in the second step are examined on the basis of this criterion and also the change in criterion (when the number of clusters increases or decreases by one), and the best solution is reported (for more detail, see the *PASW Modeler Algorithms Guide*) It should be noted that the likelihood function assumes that numeric predictors follow normal distributions and the categorical predictors follow multinomial distributions; the former assumption is not likely to hold in many data mining situations. It also assumes that the input fields are independent of each other (no relations among input fields, which is often unlikely) and that the records are independent. Also, since the first step uses a sequential clustering method, different record orderings of the same dataset may not produce the same solution. Still, the combination of speed, memory efficiency, and a criterion for selecting the number of clusters yields a cluster method with attractive properties for data mining.

Close the current stream (click **File…Close Stream**)
Open the stream file **TwoStep.str** from the c:\Train\ModelerClusAssoc directory

Run the **Table** node connected to the Type node
Close the **Table** window
Edit the **TwoStep** node

Figure 2.17 TwoStep Cluster Dialog



Model name: ○ Auto ● Custom TwoStep

☑ Use partitioned data

☑ Standardize numeric fields ☐ Exclude outliers Percentage: 25

Cluster label: ● String ○ Number

Label prefix: cluster

● Automatically calculate number of clusters

Maximum: 15 Minimum: 2

○ Specify number of clusters

Number: 10

Distance measure: ● Log-likelihood ○ Euclidean

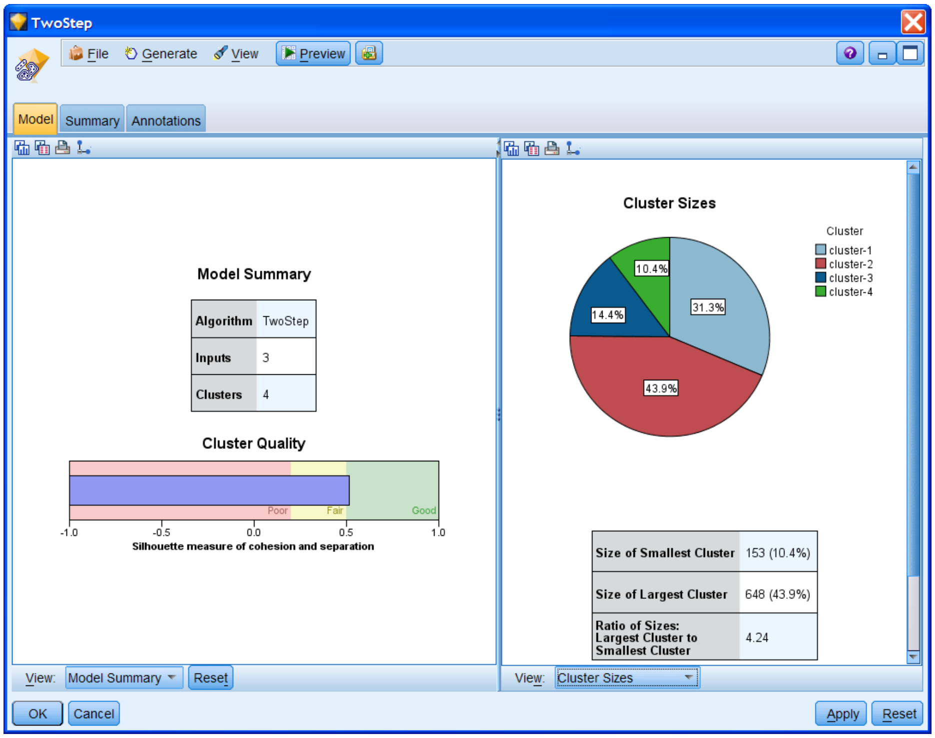
Clustering criterion: ● Schwarz's Bayesian Criterion (BIC) ○ Akaike's Information Criterion (AIC)

Notice that the TwoStep dialog has no Expert tab. By default, numeric fields are standardized before the clustering is performed. This prevents numeric fields with large variances from dominating the solution, but can be unchecked if you want fields with greater volatility to have greater influence in the solution. The *Exclude outliers* option will exclude during the first step of cluster formation those clusters with a relatively small number of records. As in the K-Means node, the field containing the cluster membership can be a string or a numeric field.

By default, the TwoStep dialog is set to have the algorithm automatically calculate the optimal number of clusters. If you wish to fit a specific number of clusters (perhaps to compare to a solution found by one of the other cluster methods, or to explore solutions with more or fewer clusters), simply click the *Specify number of clusters* option button and enter the number of clusters. The *Maximum number of clusters* setting determines the number of clusters created in the first step. In the second step, solutions from the maximum to the minimum number of clusters are considered. In practice, you might consider increasing the maximum number, especially, if under the default settings, the number of clusters found is near the maximum.

Click the **Run** button
Right-click the **TwoStep generated model** node on the Stream Canvas, and then click **Edit**

Figure 2.18 TwoStep Cluster Profiles in Cluster Viewer



Model Summary

Algorithm	TwoStep
Inputs	3
Clusters	4

Size of Smallest Cluster	153 (10.4%)
Size of Largest Cluster	648 (43.9%)
Ratio of Sizes: Largest Cluster to Smallest Cluster	4.24

The TwoStep algorithm selected four clusters, not five. There are no small clusters, and two of the clusters (1 and 2) are large. Cluster 1 is similar to cluster 00 found by Kohonen (Figure 2.15), Cluster 2 is somewhat akin to cluster 40, and Cluster 4 is similar to Kohonen cluster 20. Even then, these matches are far from perfect, as evidenced by the different number of cases in these pairs of clusters. Cluster 3 is striking in the very high local usage, which is more than double the highest cluster value found in the other solutions. (You can see the cluster profiles by clicking **Clusters** on the View drop-down list in the left panel as in the Figure 2.7.)

Note

One point concerning the TwoStep algorithm is that the assignment of records to clusters in the first step is not independent of the record order. Thus, as with the K-means and Kohonen procedures, different orderings of the data file will not necessarily produce an identical solution, and you may wish to randomly sort the data and rerun the algorithm.

Missing Data with TwoStep

The TwoStep cluster node does not support blanks (missing data). Records containing defined blanks, nulls, or missing values of any kind are excluded from model building. Thus, if you wish to include these by substituting valid values for missing data, you will need to do this before clustering with TwoStep.

We'll save the stream for use in the exercises.

Click on **File…Save Stream As**
Save the stream with the name **lesson2_TwoStep**

Summary Exercises

This set of exercises is written around the following data file: *churn.txt*. The following text gives details of the file.

churn.txt contains information from a telecommunications company. The data are comprised of customers who at some point have purchased a mobile phone. The primary interest of the company is to understand which customers will remain with the organization or leave for another company.

ChurnTrain.txt and **ChurnValidate.txt** were created by splitting the records from churn.txt into two files, one used during training and one for model validation.

The files contain the following fields:

ID	Customer reference number
LONGDIST	Time spent on long distance calls per month
International	Time spent on international calls per month
LOCAL	Time spent on local calls per month
DROPPED	Number of dropped calls
PAY_MTHD	Payment method of the monthly telephone bill
LocalBillType	Tariff for locally based calls
LongDistanceBillType	Tariff for long distance calls
AGE	Age
SEX	Gender
STATUS	Marital status
CHILDREN	Number of Children
Est_Income	Estimated income
Car_Owner	Car owner
CHURNED	(3 categories) Current – Still with company Vol – Leavers who the company wants to keep Invol – Leavers who the company doesn’t want

In these exercises you can continue to use the three streams we saved in the lesson.

1. In this exercise use the K-Means stream file we saved (*lesson2_kmeans.str* or the backup copy, *backup_kmeans.str*). The goal is to fully explore a cluster solution with n clusters (where n will vary from 3 to 6). Working with your instructor, have each person or group pick the number of clusters to request from the K-Means node. Then, after obtaining a K-Means model, do the same analysis we did in the lesson (if there are enough people in the class, the group with the 5-cluster solution can simply review the results).

2. In this next step, extend the analysis by seeing how the clusters are related to the demographic variables, using whatever techniques you think are useful. With this information and that from the first exercise, summarize the results so you can describe the cluster membership and characteristics succinctly to the group when the exercises are done. Are you satisfied with the cluster solution you found? Why or why not?

3. Now add the fields *LocalBillType* and *LongDistanceBillType* and rerun the K-Means clustering. Then repeat the analysis of steps 1 and 2. Are you any more, or less, satisfied with the cluster solution. Again, why or why not?

4. We turn next to Kohonen network clustering. Use the stream file we saved (*lesson2_kohonen.str* or the backup copy, *backup_kohonen.str*) in the lesson. Here everyone can work on the same problem, and we will continue with the 5-cluster solution, instead varying some of the other parameters that control the learning process. Do the following:
   a. Change the random seed to some other number and rerun the cluster solution;
   b. Set the random seed back to 1000, then double *Initial Eta* (*Phase 1*) and rerun the cluster solution;
   c. Now decrease *Initial Eta* (*Phase 1*) to .15 and rerun the cluster solution;
   d. Set *Initial Eta* (*Phase 1*) back to .30 and change the Learning rate decay to *Linear* and rerun the cluster solution.
   e. Each time you get a new model look at the cluster characteristics in the Cluster Viewer window of the generated model node. Also look at cluster membership with *CHURNED* as an overlay. For which model settings did the cluster characteristics change more? Would these results change how satisfied you are with the original 5-cluster solution?

5. Reset everything back to the default settings, but leave the random seed at 1000. Now try a 4-cluster solution by changing the width to 4 and the length to 1. Investigate the characteristics of this solution. Now try a 4-cluster solution with the width set to 2 and length set to 2. What type of solution did you obtain now? Can you suggest reasons for the difference?

6. *For those with extra time.* Reset everything back to the default settings, but leave the random seed at 1000. Now try a larger output layer by changing the width to 4 and length to 4. How many of the clusters contain more than 10% of the records? Examine the larger clusters and compare them to the 5-cluster solution found in the lesson.

7. In this next exercise use the TwoStep stream file we saved (*chapte24_TwoStep.str* or the backup copy, *backup_TwoStep.str*). The goal is to fully explore a cluster solution with n clusters (where n will vary from 3 to 6). Working with your instructor, have each person or group pick the number of clusters to request from the TwoStep node. Then, after obtaining a TwoStep model, do the same analysis we did in the lesson (examine the cluster sizes and profiles).

8. Add the fields LocalBillType and LongDistanceBillType and rerun the TwoStep clustering (with TwoStep automatically calculating the number of clusters). Examine the profiles and compare the cluster solution to that in the original analysis (based on three inputs). Are you any more, or less, satisfied with the cluster solution. Again, why or why not?

9. Rerun the TwoStep analysis, this time selecting the option to *Exclude outliers*. How much influence does this have on the solution?

10. Using the TwoStep procedure, increase the maximum number of clusters to 50 (using the original three inputs). Does this have a substantial impact on the solution and its interpretation? Repeat with the maximum number of clusters set to 100.
11. Continuing with TwoStep, set the maximum number of clusters back to 15 and uncheck the Standardize option. How would you characterize the change in the solution?

Lesson 3: Association Rules

Objectives

- Introduce three methods of generating association rules
- Use the Apriori node to build a set of association rules
- Interpret the results

Data

In this lesson we perform a market basket analysis of a dataset containing shopping information, *Shopping.txt*. The file contains fields that indicate whether or not a customer, during a single visit, purchased a particular product. Thus each record represents a store visit in which at least one product was purchased. The file also contains basic demographics, such as gender and age group.

3.1 *Introduction*

When people buy cigarettes do they tend to buy chocolate or beer? If people have high cholesterol do they also tend to have high blood pressure? If people buy car insurance do they also buy house insurance?

Answers to such questions can form the basis of brand positioning, advertising and even direct marketing. But how do we find whether associations such as these exist, and how can we begin to search for them when our databases have tens of thousands of records and many fields?

Association detection algorithms provide rules describing which values of fields typically occur together. They can therefore be used as an approach to this area of data understanding.

PASW Modeler contains three different algorithms that perform association detection: Apriori, Carma and Sequence.

Apriori has a slightly more efficient approach to association detection but has the limitation that it only accepts categorical fields as inputs. It also contains options that provide choice in the criterion measure used to guide rule generation.

Carma, in contrast to Apriori, offers build settings for rule support (support for both antecedent and consequent, to be defined below) rather than antecedent support. Carma also allows rules with multiple consequents, or outcomes.

Sequence is based on the Carma association rules algorithm, which uses an efficient two-pass method for finding sequences.

More detailed discussion of these algorithms is provided in the PASW *Modeler Algorithms Guide*.

All three procedures produce model nodes in the Models palette.

Association rules are presented in the format:

	Consequent	**Antecedent(s)**
Rule 1	**…**	**…**

Rule 2	…	…
…		
Rule R	…	…

For example:

Consequent	**Antecedents**
Newspapers	Fuel Chocolate

The rule states that individuals who buy fuel and chocolate are likely to also buy newspapers.

When PASW Modeler produces a set of association rules it also gives measures indicating the frequency and strength of the association for each rule. These measures are referred to as rule support and rule confidence and are given in the format:

Consequent	**Antecedent**	**Instances**	**Support**	**Confidence**	**Rule Support**	**Lift**

Instances is the number of records in the dataset that match the antecedents.
Support is the percentage of records that match the antecedents.
Confidence is the percentage of all records matching the antecedents that also match the consequent.
Rule support is the percentage of records that match the entire rule (both the antecedents and consequent).
Lift is the expected return using a model or rule. In this context it is the ratio of the rule confidence to the overall percentage occurrence of the consequent in the data.

Therefore the full format of a rule will appear as:

Consequent	**Antecedent**	**Instances**	**Support**	**Confidence**	**Rule Support**	**Lift**
Newspapers	Fuel Chocolate	2051	15.0	71.0	10.6	2.00

In this example, 15% of the customers (2051 individuals) bought fuel and chocolate. Of these, 71% also bought newspapers. 10.6% of the customers bought fuel, chocolate, and newspapers. The lift value of 2.00 indicates that those who purchase fuel and chocolate are twice as likely to buy newspapers as is the overall sample (71.0% versus 35.5%—not shown in the table).

3.2 *The Apriori Node*

Since our data fields are categorical, we will demonstrate the Apriori rule association detection algorithm.

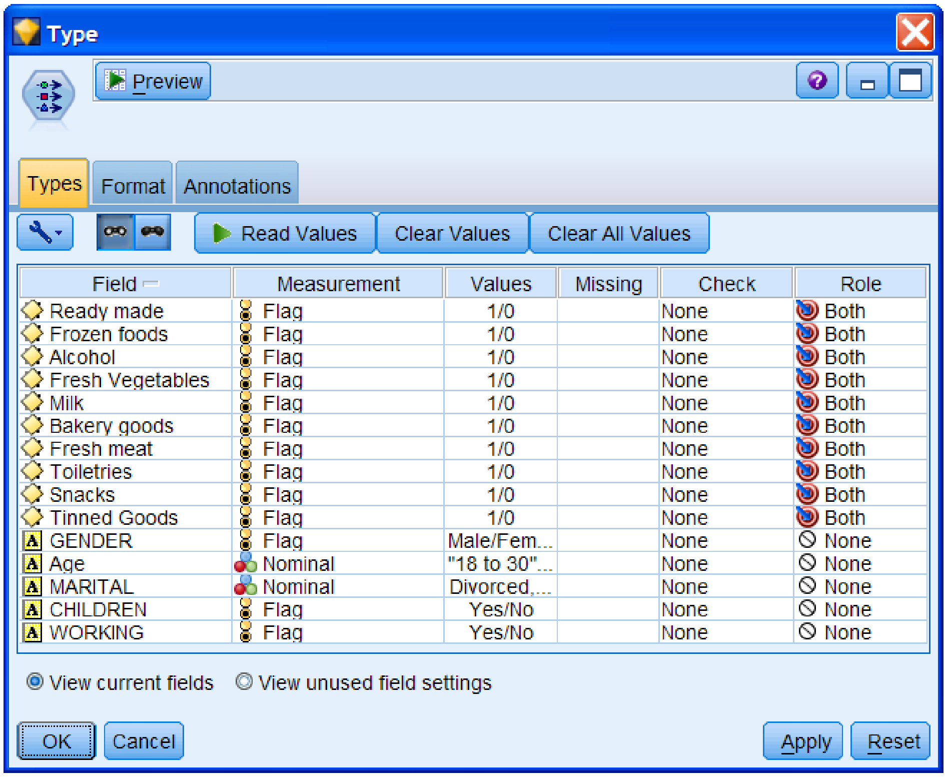
As with all the modeling nodes, Type declaration (via a source or Type node) must precede the Apriori node in the stream. Also, all field types must be fully instantiated. The Apriori, Carma and Sequence are the only modeling nodes that recognize the role setting of BOTH, since a field may appear as an antecedent and a consequent.

We begin by opening a previously prepared PASW Modeler stream file that contains a source node that reads the Shopping.txt data file, along with Type and Table nodes.

Click **File…Open Stream**, move to the **c:\Train\ModelerClusAssoc** directory and double-click **Shoppingdef.str**
Double-click the **Type** node
Change the **Role** setting for all **product fields (Ready made** to **Tinned goods)** to **Both** (select all product fields, right-click and choose **Set Role…Both** from the context menu)
Change the **Role** setting for the **demographic fields** to **None** (select the fields, right-click and choose **Set Role…None** from the context menu)

Notice that the shopping fields are set to flag measurement even though they are coded (0,1). As in the Kohonen example earlier, we set their measurement to Categorical and then read the data. Otherwise, they would have measurement continuous, which isn't appropriate.

Figure 3.1 Type Node for Rule Association Modeling (Apriori)



Field	Measurement	Values	Missing	Check	Role
Ready made	Flag	1/0		None	Both
Frozen foods	Flag	1/0		None	Both
Alcohol	Flag	1/0		None	Both
Fresh Vegetables	Flag	1/0		None	Both
Milk	Flag	1/0		None	Both
Bakery goods	Flag	1/0		None	Both
Fresh meat	Flag	1/0		None	Both
Toiletries	Flag	1/0		None	Both
Snacks	Flag	1/0		None	Both
Tinned Goods	Flag	1/0		None	Both
GENDER	Flag	Male/Fem...		None	None
Age	Nominal	"18 to 30"...		None	None
MARITAL	Nominal	Divorced,...		None	None
CHILDREN	Flag	Yes/No		None	None
WORKING	Flag	Yes/No		None	None

● View current fields ○ View unused field settings

We are now ready to find a set of association rules using Apriori.

Click **OK** to close the **Type** node
Place the **Apriori** node from the Modeling palette to the right of the **Type** node
Connect the **Type** node to the **Apriori** node
Double-click the **Apriori** node named **10 fields**

The default name of the resulting unrefined model is the number of fields. A new name can be entered in the Custom Model name text box.

The *Use partitioned data* option splits the data into separate Training, Testing (and Validation) samples. Because we will not be using the Partition node in this stream this option will be ignored even though it is checked.

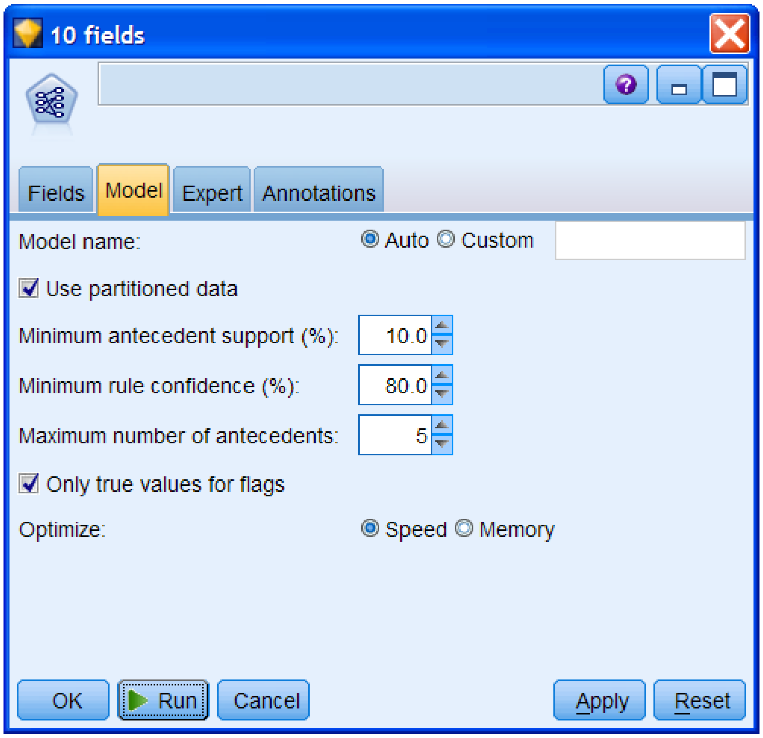
Apriori will produce rules that, by default, have a minimum support of 10% of the sample, and a minimum confidence of 80%. These values can be changed using the spin controls. In practice there is some trial and error involved in finding useful values (too high and there are no rules generated; too low and there are many rules generated). Examining distribution plots on the fields to be analyzed provides base rates for the fields, which can provide some guidance in setting the minimum support value.

The maximum number of antecedents in a rule is set using the *Maximum number of antecedents* option. Increasing this value tends to increase processing time.

By default, rules will only contain the true value of fields that are defined as flags. This can be changed by deselecting the *Only true values for flags* check box. Since we are interested in rules describing what individuals purchase, rather than what they don't purchase, we will retain this setting.

The algorithm can be optimized in terms of its *Speed* or its *Memory* usage.

Figure 3.2 Apriori Model Node Dialog



10 fields

Model name: ● Auto ○ Custom

☑ Use partitioned data

Minimum antecedent support (%): 10.0

Minimum rule confidence (%): 80.0

Maximum number of antecedents: 5

☑ Only true values for flags

Optimize: ● Speed ○ Memory

Apriori searches for rules and discards rules that are not of interest based on the specified criteria. As in the other modeling tools, Expert options give the user greater control over the search. The reader is referred to the *PASW Modeler User's Guide* and Lesson 4 in this training manual for more information on expert settings. In this example we will start with the default settings. We may find that the support and confidence values need to be adjusted.

Click **Run**

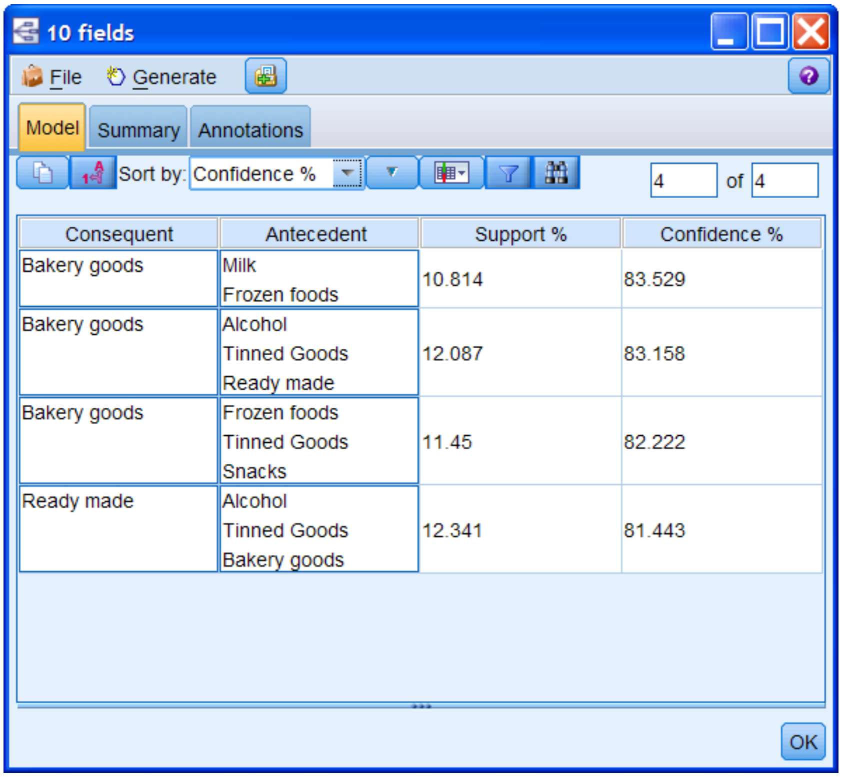
An Apriori generated model node will appear in the Models palette.

Right-click the **Apriori node** in the Models palette of the Manager window, then click **Browse**

The Apriori algorithm has found only four association rules. By default, in addition to the consequent and antecedent, only the *Support %* and *Confidence %* are displayed. The rules are initially sorted by *Confidence %*. The first rule tells us that on 10.81% of the records (shopping visits), milk and frozen foods were purchased. Of this group, 83.53% also bought bakery goods.

Rules can be sorted by *Support %*, by *Confidence %*, alphabetically by *Consequent*, or by *Rule Support %* and *Lift* if displayed. Simply select the order using the *Sort by* drop-down list and the *Sort by* button ▼ controls the direction of the sort.

Figure 3.3 Browsing the Rules Generated by Apriori



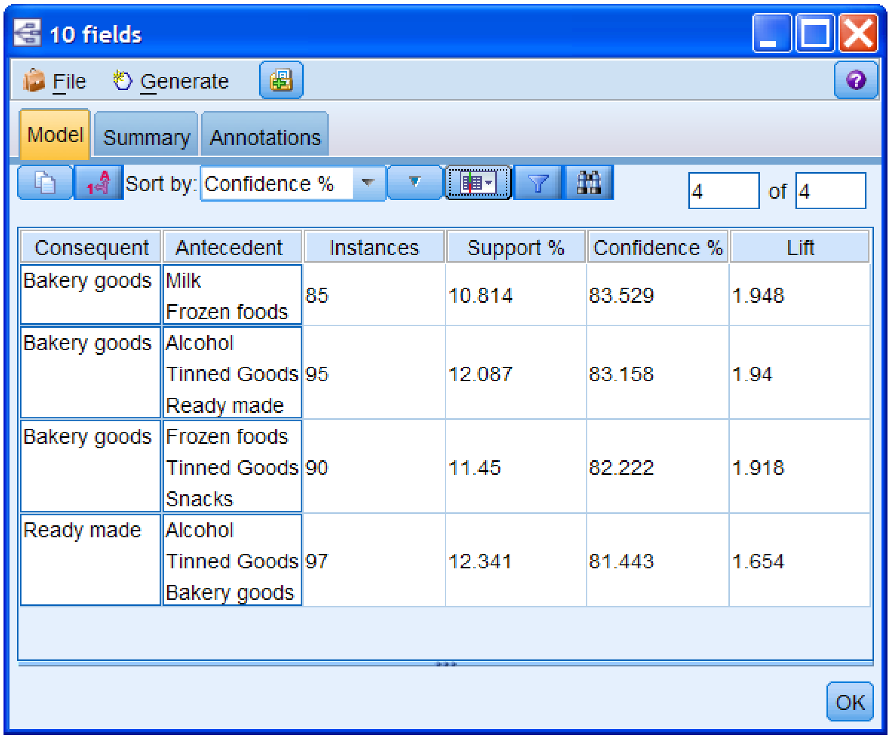
Consequent	Antecedent	Support %	Confidence %
Bakery goods	Milk Frozen foods	10.814	83.529
Bakery goods	Alcohol Tinned Goods Ready made	12.087	83.158
Bakery goods	Frozen foods Tinned Goods Snacks	11.45	82.222
Ready made	Alcohol Tinned Goods Bakery goods	12.341	81.443

To see the other available measures:

Click the **Show/hide criteria menu** button ▦
Click **Instances**
Click the **Show/hide criteria menu** button again
Click **Lift**

We now see that milk and frozen foods were purchased on 85 shopping trips. There are 786 records in the file, so this is (85/786)*100, or 10.81% of the total number of shopping trips (*Support %*). The *Lift* value is the expected return resulting from using the model or rule. Here it is the ratio of the confidence to the base rate of the consequent. For example, bakery goods were purchased on 42.88% of the trips overall (we can use a Distribution node to determine this), but were purchased 83.5% of the time when milk and frozen food were purchased. The lift from using this rule is 83.5/42.88 or 1.948, and means that the chances of buying bakery goods almost double when milk and frozen foods are purchased.

Figure 3.4 Adding Instances and Lift to Displayed Rule Statistics



Consequent	Antecedent	Instances	Support %	Confidence %	Lift
Bakery goods	Milk Frozen foods	85	10.814	83.529	1.948
Bakery goods	Alcohol Tinned Goods Ready made	95	12.087	83.158	1.94
Bakery goods	Frozen foods Tinned Goods Snacks	90	11.45	82.222	1.918
Ready made	Alcohol Tinned Goods Bakery goods	97	12.341	81.443	1.654

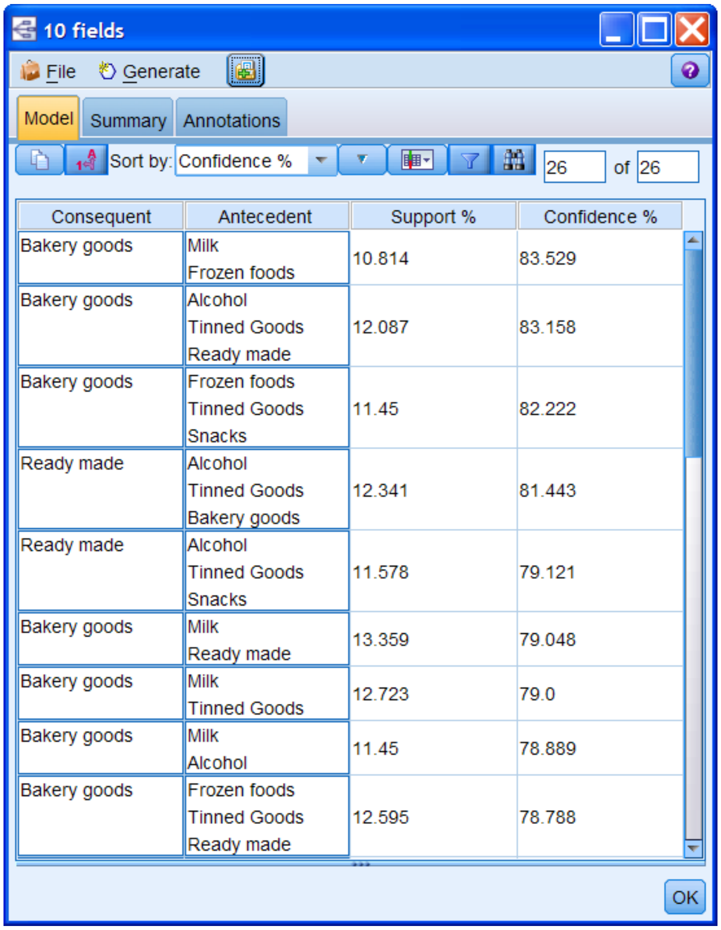
Association rules can be exported as HTML, text, or PMML using the *File…Export* menu. This menu also has a print option.

The *Generate* menu allows you to generate a selection node for records that conform to a rule; just select the rule and choose *Select Node* from the *Generate* menu. A complete rule set for a specified consequent can also be generated.

We will now investigate whether, by dropping the confidence of the rules to 75%, we can obtain a larger number of associations.

Click **File…Close** to close the Apriori Association Rules Browser window
Double-click the **Apriori** modeling node
Enter **75** in the **Minimum rule confidence:** text box (type or use the spin control)
Click **Run**
Right-click the **Apriori node** in the Models palette of the Manager window, then click **Browse**

Figure 3.5 Association Rules Generated by Apriori with Lower Minimum Confidence



Consequent	Antecedent	Support %	Confidence %
Bakery goods	Milk Frozen foods	10.814	83.529
Bakery goods	Alcohol Tinned Goods Ready made	12.087	83.158
Bakery goods	Frozen foods Tinned Goods Snacks	11.45	82.222
Ready made	Alcohol Tinned Goods Bakery goods	12.341	81.443
Ready made	Alcohol Tinned Goods Snacks	11.578	79.121
Bakery goods	Milk Ready made	13.359	79.048
Bakery goods	Milk Tinned Goods	12.723	79.0
Bakery goods	Milk Alcohol	11.45	78.889
Bakery goods	Frozen foods Tinned Goods Ready made	12.595	78.788

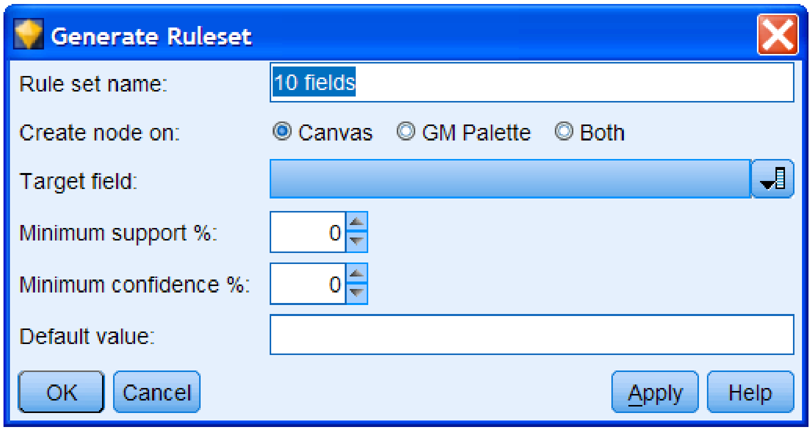
This is a far richer set of associations, and 26 rules are produced in all. The first three rules (ranked by confidence, all over 82%) involve the same consequent, bakery goods, with support of approximately 11-12%. The challenge is now to examine the rules for those that might be useful in the context of your business question or goal.

3.3 *Using the Associations*

A rule set, similar to that of the rule induction models, can be created for a selected consequent by selecting the Rule Set option under the Generate menu of the Association Rules browser window. This will generate a new model which, when placed in a data stream, will create a field indicating whether a rule in the rule set applies to the record, along with its confidence value. Note that more than a single rule may apply to a record and, by default, the confidence value is based on the first rule whose conditions match the record. We will create a rule set for the *Alcohol* field.

Click **Generate…Rule Set**

Figure 3.6 Generate Ruleset Dialog



Generate Ruleset

Rule set name: 10 fields

Create node on: ● Canvas ○ GM Palette ○ Both

Target field:

Minimum support %: 0

Minimum confidence %: 0

Default value:

A model node, labeled by the name in the *Rule set name* text box, can be created on the Stream Canvas (*Canvas*), the Models palette (*GM Palette*), or *Both*.

A new association rule set node for the *Target field* will be created. This node will contain all rules with the specified *Target field* as the conclusion. When the data stream is passed through the generated Rule Set node, it will create a new field in the data recording whether a record has met the conditions for one or more of the rules in the rule set.

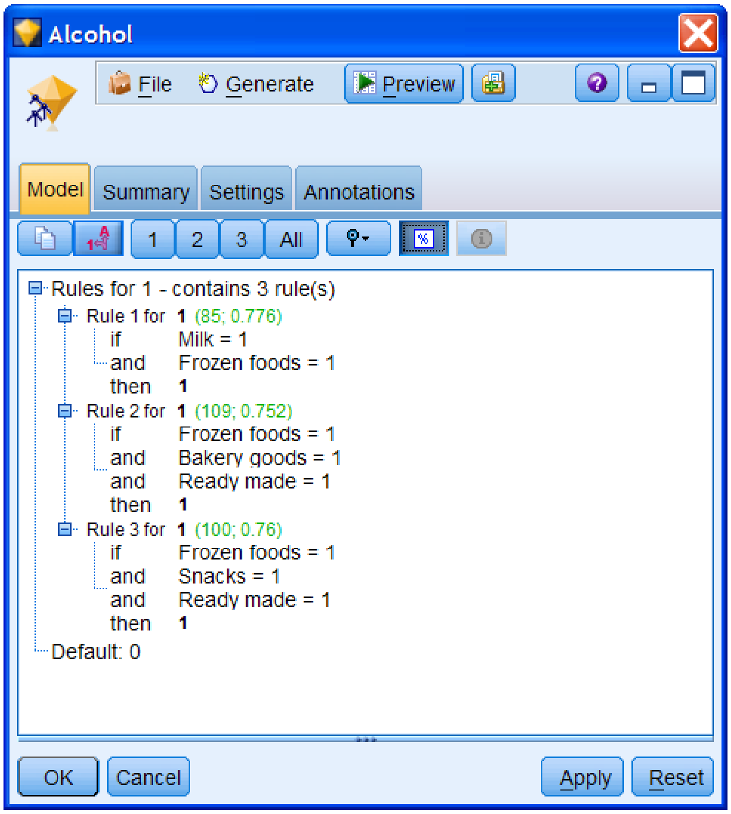
You may specify a *Default value* for the rule set (value if no rule applies to a record), as well as *Minimum support* and *Minimum confidence* for the rules.

Type **Alcohol** in the **Rule set name** text box
Select **Alcohol** in the **Target Field** list
Type **0** in the **Default value**: text box (not shown)
Click **OK**
Click **File…Close** to close the Rule browser window

A generated Apriori Rule Set node named *Alcohol* will appear in the upper left corner of the Stream Canvas.

Double-click the generated **Apriori Rule Set** node named **Alcohol** in the Stream Canvas
Click the **All** button
Click the **Show or hide instance and confidence figures** button

Figure 3.7 Fully Unfolded Rule Set Generated from Apriori



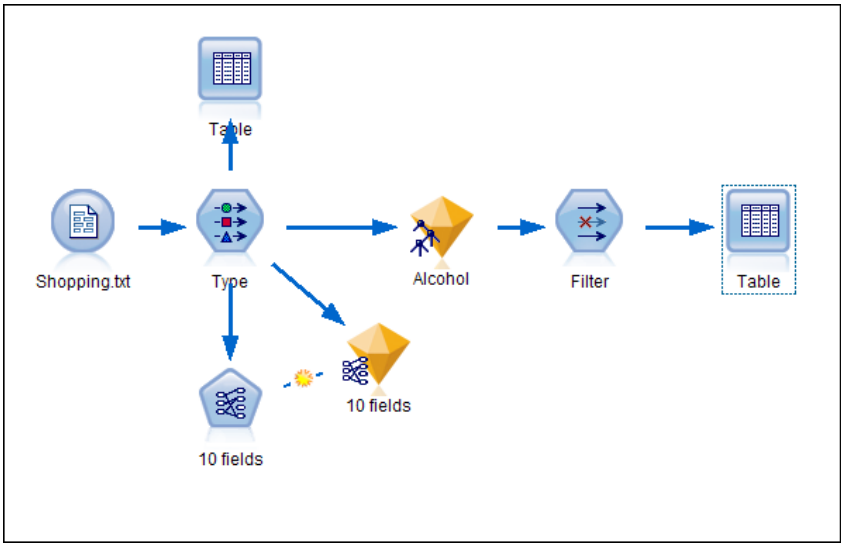
```
Rules for 1 - contains 3 rule(s)
   Rule 1 for  1  (85; 0.776)
      if       Milk = 1
      and      Frozen foods = 1
      then     1
   Rule 2 for  1  (109; 0.752)
      if       Frozen foods = 1
      and      Bakery goods = 1
      and      Ready made = 1
      then     1
   Rule 3 for  1  (100; 0.76)
      if       Frozen foods = 1
      and      Snacks = 1
      and      Ready made = 1
      then     1
Default: 0
```

In this example the rule set contains three rules whose consequent is buying Alcohol. The first associates the purchase of milk and frozen foods. The second associates frozen foods, bakery goods and ready-made meals. The third associates frozen foods, snacks and ready made-meals meals.

We can now pass the data through this node and generate a field indicating whether or not a record conforms to the conditions of any of the three rules.

Click **OK** to close the Rule Set Browser window
Drag the **Apriori Rule Set** node named **Alcohol** to the right of the **Type** node
Connect the **Type** node to the **Apriori Rule Set** node named **Alcohol**
Place a **Filter** node from the Field Ops palette to the right of the **Apriori Rule Set** node named **Alcohol**, and connect the **Apriori Rule Set** node to it
Edit the **Filter** node and **deselect** the fields that are not used in the rules (**Fresh Vegetables**, **Fresh Meat**, **Toiletries**, **Tinned Goods** and the **demographic fields**), and then click **OK**
Place a **Table** node from the Output palette to the right of the **Filter** node
Connect the **Filter** node to the new **Table** node

Figure 3.8 Stream with Generated Rule Set



Right-click the new **Table** node, then click **Run**

Figure 3.9 Fields Created by the Generated Apriori Rule Set Node

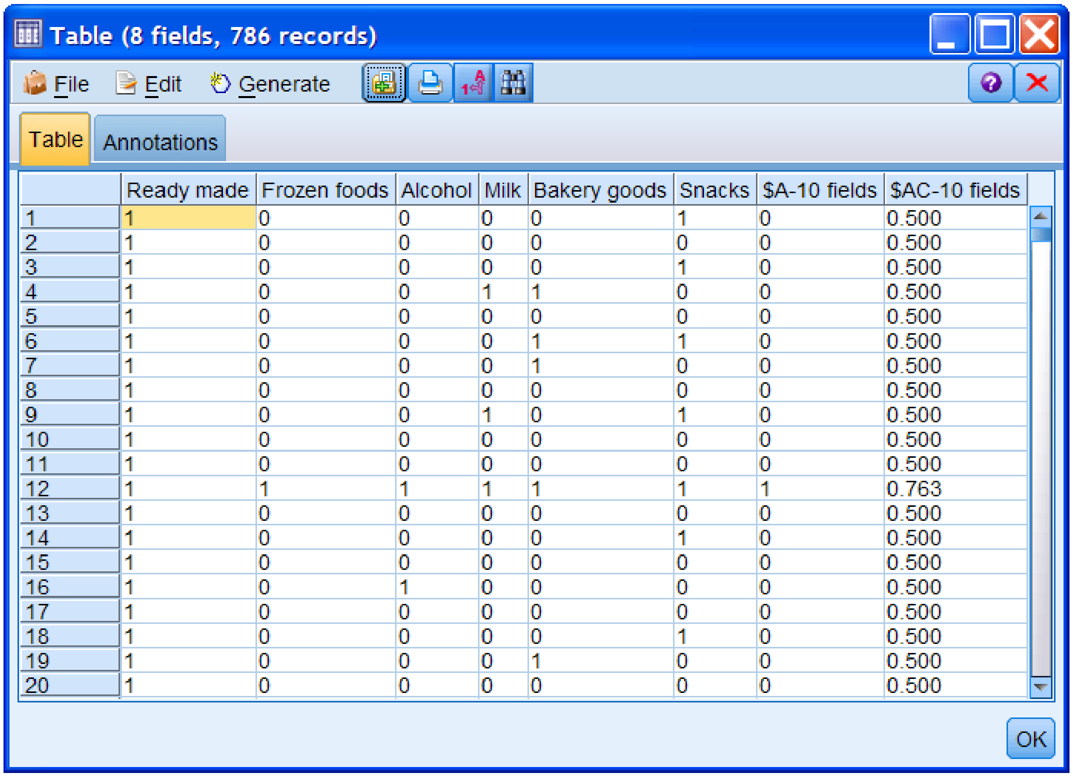


Table (8 fields, 786 records)

	Ready made	Frozen foods	Alcohol	Milk	Bakery goods	Snacks	$A-10 fields	$AC-10 fields
1	1	0	0	0	0	1	0	0.500
2	1	0	0	0	0	0	0	0.500
3	1	0	0	0	0	1	0	0.500
4	1	0	0	1	1	0	0	0.500
5	1	0	0	0	0	0	0	0.500
6	1	0	0	0	1	1	0	0.500
7	1	0	0	0	1	0	0	0.500
8	1	0	0	0	0	0	0	0.500
9	1	0	0	1	0	1	0	0.500
10	1	0	0	0	0	0	0	0.500
11	1	0	0	0	0	0	0	0.500
12	1	1	1	1	1	1	1	0.763
13	1	0	0	0	0	0	0	0.500
14	1	0	0	0	0	1	0	0.500
15	1	0	0	0	0	0	0	0.500
16	1	0	1	0	0	0	0	0.500
17	1	0	0	0	0	0	0	0.500
18	1	0	0	0	0	1	0	0.500
19	1	0	0	0	1	0	0	0.500
20	1	0	0	0	0	0	0	0.500

Two new fields appear in the data table. The first, *$A-Alcohol*, is 0 unless one of the three rules in the rule set applies to the record, in which case it has a value of 1. The second field, *$AC-Alcohol*, represents the confidence figure for the rules decision. Notice that when the conditions of the rules do not apply to a record, its confidence value is .5.

We see that Rule 3 applied for record 12, since the confidence value is .76 (see Figure 3.9. The use of a rule set allows you to identify which customers are associated with a particular rule. From here various types of other analysis can be done (e.g., are those customers who match this rule more likely to be male or female, younger or older, etc.)

Extension: Exporting Model Rules

If you wish to pass model values (predictions, confidence values, cluster groups) to other programs, you can easily do so using the Flat File, Statistics Export, Database, Excel, or SAS® Export nodes in the Output palette. In addition, the PASW Modeler Solution Publisher contains options to export scores to databases.

Summary

In this lesson you have been introduced to association rule detection within PASW Modeler.

You should now be able to:

- Create a set of association rules using Apriori
- View the resulting rules by browsing the model
- Explain the meaning of rule confidence, support, rule support and lift
- Sort the rules based on different criteria
- Create a rule set and use this to identify those records whose conditions are related to a selected conclusion

Summary Exercises

In this session we will attempt to run a market basket type analysis using a dataset containing information on persons attending SPSS® courses. The data file, *UKTraining.txt*, contains information regarding the type of courses more than 2000 individuals have attended. The following text gives details of the file.

UKTraining.txt contains information on which courses each UK customer took, the course location, whether or not they had a subscription, how much money they spent, and which sector they are classified in. The file contains 2136 records and the following fields:

ID
Sector
Number of Courses
Privilege card
Answertree
Introduction to CHAID
Data Entry
Intermediate Techniques
Introduction to SPSS® and Statistics
Perceptual Mapping
Neural Connections
Introduction to Statistics
Time Series
Total Spend
Venue
Mini subscription or 5 day deal
Subscription
Building Predictive Models
Classification and Clustering
FastTrack
Introduction to SPSS®
Mapinto
Market Segmentation
Scripting
Tables
ANOVA models

1. Open the stream saved in the previous lesson, UKTraining.*str*.

2. Edit the Var. File node and select the Types tab.

3. Change all of the course title fields to role BOTH, so that they appear in rules as either an antecedent or consequent, apart from the following:

   Introduction to SPSS
   Introduction to SPSS and Statistics
   FastTrack

   which must all be set to INPUT. (We are interested in which introductory courses lead to others, so these fields will act as conditions only.)

4. Connect an Apriori node to the Type node and edit the Apriori node.

5. Set the Minimum rule support to 1% and the Minimum rule confidence to 50%. Click the *Only true values for flags* check box (so it is checked).

6. Run this section of the stream

7. Browse the resulting model in the Generated Models palette and sort the rules by support x confidence.
8. Do the rules make sense?

Lesson 4: Advanced Association Rules

Objectives

- Review the three types of Association model nodes in PASW Modeler
- Consider when each should be used and how they can be applied to association data
- Run examples and review the output for comparison purposes

Data

In this lesson we continue to use the dataset containing *Shopping.txt* containing shopping information. The file contains fields that indicate whether or not a customer, during a single visit, purchased a particular product. Thus each record represents a store visit in which at least one product was purchased. The file also contains basic demographics, such as gender and age group.

4.1 *Introduction*

Generalized rule induction, association rule discovery, affinity analysis and market basket analysis are terms used to describe a particular type of pattern detection algorithm that is distinct from decision trees. These methods generate rules that are independent of other rules and are not restricted to a single output or dependent field. The rules discovered by these methods reveal which values of two or more fields typically occur together (such as buying bread, wine, and cheese on one shopping trip to the grocery store). Negative relationships can also be discovered, e.g., not buying bread is associated with buying soda.

As we learned in the previous lesson, PASW Modeler includes three different modeling nodes to perform association rule discovery, Apriori, Carma and Sequence. We will examine each algorithm in turn and run examples of each, modifying the default settings to understand the behavior of the algorithms under the different options. Sequence detection, a related technique that focuses on discovering rules in data that are time structured, is reviewed in Lessson 5.

4.2 *Association Rules*

Apriori, begin by generating simple rules (those involving two items) and testing them against the data. The most interesting of these rules—those that meet the specified minimum criteria for support and confidence, plus an evaluation measure—are stored. Next, the rules are expanded by adding an additional condition from a third field (this process is called specialization) and are again validated against the data. The most interesting of these rules are stored, and specialization continues until no rules meet the minimum criteria, or the maximum number of antecedents is attained.

In contrast, Carma (continuous association rule mining algorithm) uses only two data passes. In the first, it identifies frequent itemsets in the data, where an itemset is a group of items that tend to co-occur. Then in a second stage, the exact frequencies for the candidate itemsets are computed, and those that meet the support and confidence criteria are retained.

As a reminder, association rules are statements of the form:

Conclusion ← Condition(1) & Condition(2) & … & Condition(n)

Where the conditions are known as the *antecedents* of the rule. Association rules are generally evaluated by three criteria, support, rule support, and confidence.

Support is the percentage of records in the dataset for which the conditions (antecedents) hold. It thus indicates how general the conditions to the rule are.

Rule support is the percentage of records in the dataset for which both the antecedents and the consequents hold. It thus indicates how general the rule is—to what percentage of the data it applies.

Confidence is the proportion of those records meeting the conditions (antecedents) that also meet the conclusion. It thus indicates how likely the conclusion, given that the conditions are met.

Although association rules have some advantages over tree-based methods, the search space for independent rules is exponential in the number of attributes; hence association rule algorithms are computationally expensive. For this reason, when using large datasets, it is recommended that a representative sample be used initially to help determine appropriate values for support and confidence. This is less true for the Carma algorithm because it requires only two data passes.

4.3 *Apriori*

The Apriori algorithm works with only categorical data—fields coded as flags or nominal. To further minimize computation time, Apriori does some clever indexing and minimizes passes through the complete dataset to generate the association rules. (see "Fast Discovery of Association Rules" by Agrawal, et al in Fayyad, Piatetsky-Shapiro, Smyth, and Uthurusamy, *Advances in Knowledge Discovery and Databases*, 1996 for further information on the exact procedures).

Three selection criteria control the Apriori algorithm: minimum support, minimum rule confidence, and maximum number of antecedents. The defaults for these values (10%, 80%, and 5, respectively) are unlikely to be appropriate for any particular dataset. Typically the support is decreased in the beginning of analysis to generate more potential rules, although this depends in part on personal preference and the base rates of the items. Note that if minimum support or minimum rule confidence is set too high for a particular dataset, no rules will be generated.

4.4 *Carma*

The Carma algorithm only uses categorical fields of measurement flag, when reading standard tabular data (one record per case). The fields can be of any role, and Carma models are comparable to Apriori models with all fields set to Both. You can constrain which items appear only as antecedents or consequents by filtering the model after it is built.

Carma also uses the three parameters of rule support, rule confidence, and maximum rule size (total number of antecedents and consequents) to control rule building. These values are 20%, 20%, and 10, respectively, by default, and will invariably be modified by the user. Usually the rule support is decreased as a first step.

4.5 *Apriori Expert Options*

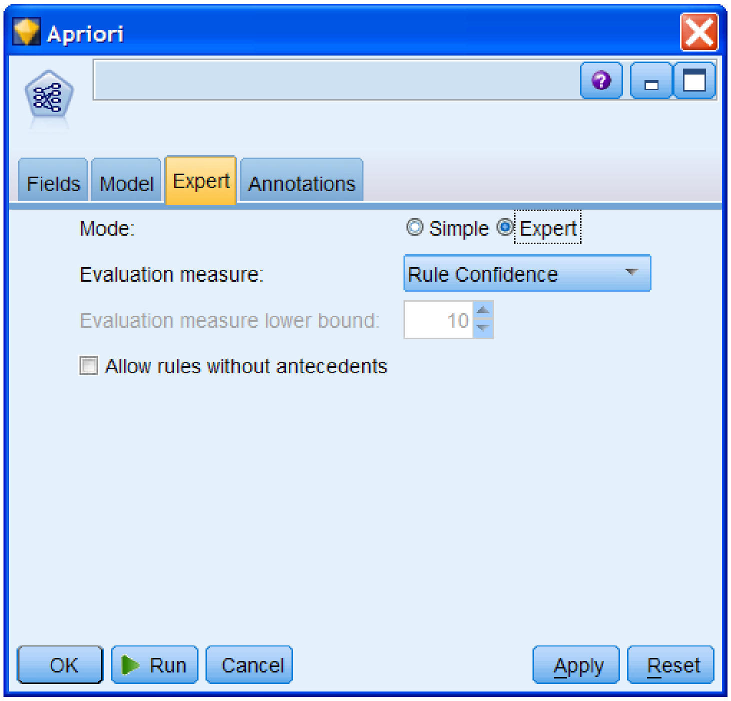
We'll now consider the available options under the expert setting in Apriori. To see these in PASW Modeler we need to place an Apriori node on the Stream pane.

> Place an **Apriori** node in the Stream canvas
> Double-click the **Apriori** node to edit it
> Click on the **Expert** tab
> Click the **Expert** option button

When the *Expert Mode* option button is selected, the *Evaluation Measure* drop-down list becomes active (as will the *Evaluation measure lower bound* spin control if an evaluation measure other than *Rule Confidence* is chosen).

Apriori uses the antecedent support and confidence of a rule, as explained previously, to determine which rules to retain. This default method is the *Rule Confidence* evaluation measure, based on rule confidence. However, this rule-selection measure is not necessarily the best for all circumstances. Essentially, if rules are selected using only confidence, or correct predictions, then the algorithm will only find accurate rules. This sounds like a tautology, but accurate rules are not necessarily the most interesting and useful rules.

Figure 4.1 Expert Options for the Apriori Node



Apriori

Mode: ○ Simple ● Expert

Evaluation measure: Rule Confidence

Evaluation measure lower bound: 10

□ Allow rules without antecedents

For example, imagine that we are attempting to derive a rule set from a medical database with information on many types of conditions. For rules with only one antecedent, we may well find that the following rule has the highest level of confidence:

Pregnant → Female (if pregnant then female)

Obviously, this rule would have 100% confidence (we hope), but it is hardly an interesting rule. But the rule confidence evaluation measure would discover it because its confidence exceeds a threshold value.

This is an extreme example, but let’s look at a typical retail case to further motivate our discussion of the various evaluation measures. Imagine that in a supermarket 40% of all customers buy cheese on a shopping trip. Suppose then that a simple rule is found stating that “if fruit, then cheese,” with a confidence of 43% (we would have to change the default *Minimum rule confidence* setting to find it). In other words, 43% of customers who buy fruit also buy cheese.
Is this an important rule? Clearly, the answer is no, since buying fruit does not have a significant influence on whether someone buys cheese, as the percentages are almost the same. But if we set a

confidence limit of 40%, say, we would find this relatively trivial rule with the rule confidence evaluation measure.

Another common situation is where the conclusion occurs less often when conditions are added. Assume that the confidence of the rule "if fruit, then cheese" is not 43%, but 15%. With a confidence limit of 40% this rule would not be selected, but it may be very important to know about this relationship for a retailer! Together with fruit, cheese is bought less frequently than it is bought at all. Perhaps fruit is some kind of substitute for cheese on dessert trays (when dining in the Continental style).

In any case, a rule with low confidence can be interesting and is equivalent to negation, since "fruit, then cheese" with 20% confidence is equivalent to "fruit, no cheese" with 80% confidence. It is possible to tell Apriori to use both true and false values for flag fields, and negative relationships will then be found, but typically so many will be found that this setting is not always an effective means to discover associations.

The bottom line is that potentially interesting rules are often those that have large differences in their confidence from the baseline value of confidence for the conclusion itself. Adding an item to the antecedent is only informative if it significantly changes the confidence of the rule; otherwise, the simpler rule suffices. The lift statistic, which is the ratio of the rule confidence to the base rate of the conclusion alone and appears in model output, is sensitive to large increases in confidence.

Apriori provides four evaluation measures that attempt to find interesting rules by looking for differences in confidence as just explained. Each compares the current confidence of a rule to that for the rule with empty antecedent, i.e., with only the conclusion. The confidence of an empty rule is simply the relative frequency of the conclusion, and it is called the *prior confidence* in all that follows. The confidence of a rule with one or more antecedents is called the *posterior confidence*. The lift measure, mentioned earlier, is the ratio of the posterior confidence to the prior confidence. The prior confidence of an item can be found with the *Allow rules without antecedents* check box, which allows rules that only include the consequent, i.e., one item.

Confidence Difference

This is the simplest measure and is based on the absolute difference between the posterior and prior confidence. The *Evaluation measure lower bound* (by default set to 10%) sets the minimum difference between the two confidences. Thus in equation form, this translates to the following condition for rule selection:

$$|\,\text{Posterior confidence} - \text{Prior confidence}\,| > \text{Confidence difference lower bound}$$

Thus a rule whose conclusion is nearly always true will only be selected if its prior confidence is low. Moreover, given that the absolute value of the difference is being used, negative rules can be found by this measure.

In the fruit and cheese example, if the *Evaluation measure lower bound* were set to 50%, and the prior confidence of cheese at 40%, the rule "if fruit, then cheese" would have to attain better than 90% confidence to be selected. The selection of rules is symmetrical about the prior confidence value. So if the Evaluation measure lower bound were set to, say 20%, cheese rules would be found that had confidence above 60% or below 20%.

Confidence Ratio

Another relatively simple way to compare the two confidences is to compute their ratio and subtract it from 1. Since this method uses a ratio, unlike the confidence difference measure, it is more sensitive to ratios in the lower confidence regions. (The ratio between 30% (prior) and 40% (posterior) is greater than that between 70% (prior) and 80% (posterior), even though the absolute difference is identical).

Rules are selected when the following relationship holds:

$$1 - \frac{\text{min(Posterior confidence, Prior confidence)}}{\text{Max(Posterior confidence, Prior confidence)}} > \text{Confidence ratio lower bound}$$

Thus if posterior confidence > prior confidence, a rule is selected when:

$$\text{Posterior confidence} > \frac{\text{Prior confidence}}{(1 - \text{EMLB})}$$

And if posterior confidence < prior confidence, a rule is selected when:

$$\frac{\text{Posterior confidence}}{(1 - \text{EMLB})} < \text{Prior confidence}$$

where EMLB is the evaluation measure lower bound.

This option can find rarities and negative effects, and it can be more finely tuned than the confidence difference. It will find both positive and negative rules across a range of prior confidences. In essence, if the prior confidence is low, then only a small change in the posterior confidence is required to select a rule, even for negative values. An example will be helpful.

Let’s say that the prior confidence of buying cheese is 20%. We set the *Evaluation measure lower bound* to 40%. In addition, suppose that when customers buy fruit, they then buy cheese 35% of the time.

Since the posterior confidence (35%) is greater than the prior confidence (20%), we need to determine if Posterior confidence > $\frac{\text{Prior confidence}}{(1 - \text{EMLB})}$, or is .35 > $\frac{.2}{(1 - .4)}$? Since .35 is greater than .33 (or .2/.6), the rule would be selected. If, instead, the prior confidence were 60%, and the posterior confidence 90% for the same two products, the right-hand side becomes $\frac{.6}{(1 - .4)}$ or 1, which exceeds .9, so the rule would not be selected, even though the difference (.9 – .6 = .3) is twice as large (.35 – .2 = .15) as before.

Clearly, the confidence ratio measure is biased toward selecting rules with low prior confidences, or toward negative rules with high prior confidences.
Note that PASW Modeler reports confidence as a percentage, not a proportion, and, in order to be consistent, multiplies by 100 the confidence ratio value as calculated in the formula above.

Information Difference

This evaluation measure is more complicated than the previous two and is based on the information gain measure used in the C5.0 algorithm. The information gain measure is calculated in bits of information. The idea behind the algorithm is as follows: Let's say that we have a certain probability for cheese and no cheese (say 40% and 60%). This distribution has a certain entropy, which reaches its maximum when the distribution has an equal number of cases in each category (i.e., entropy is 1 for a flag field when the categories are 50/50). After we get the information about the items in the antecedent of a rule, we have a different probability distribution for when people buy cheese with fruit, with another entropy value. The change in entropy between the two conditions is then used for rule selection after normalization, so that it ranges from 0 to 1.

The Evaluation measure lower bound is given in 100ths of a bit, or equivalently, the percentage of one bit that entropy changes when the antecedents are added. So an Evaluation measure lower bound of 50% means that we are looking for rules that increase our information by .50 bit.

The information difference measure takes into account the support of a rule, so that under this measure, for the same prior and posterior confidences, a rule will be valued more highly if it applies to a larger number of cases. It thus tends to eliminate the rarely relevant rules sometimes found by other algorithms. However, since it is difficult to translate information gain in bits to percentage differences, using this evaluation measure typically requires a bit of experimentation to get a useful setting.

Normalized Chi-square

This measure uses the well-known chi-square test from classical statistics to detect dependencies between variables. We can illustrate it for the simple situation of one antecedent with the table below, which illustrates the four possible combinations of values for fruit and cheese.

	Not Buy Fruit	Buy Fruit
Not Buy Cheese	A	B
Buy Cheese	C	D

We use the column and row totals, and the table total, to determine the expected number of cases in each cell. Thus, the expected value for cell D, the one of interest where customers buy fruit and cheese, is:

$$\frac{(B+D) \text{ x } (C+D)}{(A+B+C+D)}$$

The value of chi-square is normalized to remove the effect of the number of cases (records). With this normalization, the chi-square measure can assume values between 0 (no relationship) and 1 (perfect relationship). The Evaluation measure lower bound is thus the strength of the dependence, or chi-square value.

As with the information difference to prior measure, the normalized chi-square measure is not intuitively related to the differences between the prior and posterior confidences, so some experimentation is required to get useful settings. In general, this measure is more sensitive to support, all other things being equal.

Support and Confidence with Evaluation Measures

For all these evaluation measures, it is important to note that the *Minimum rule confidence* and *Minimum antecedent support* thresholds are still applied. Thus interesting rules may be eliminated even though they satisfy the specific evaluation measure. To have *only* the evaluation measure determine which rules are found, set the support and confidence thresholds to zero. In addition, if you want to find negative rules for flags, the *Only true values for flags* checkbox must be deselected.

Data Format

Apriori accepts the data in two formats. Tabular data has items represented by separate flag fields, and each record represents a complete set of associated items. Transactional data contains two fields: one for an ID and one for content. Each record represents a single item and having the same ID links associated items. Suppose a customer purchased five software products from a set containing ten product possibilities. In Tabular format, she would be represented by one data record containing ten fields, one for each product, with each field coded true or false. In Transactional format, she would be represented by five records, each containing an ID field (coded with her ID value) and a content field (containing a product code).

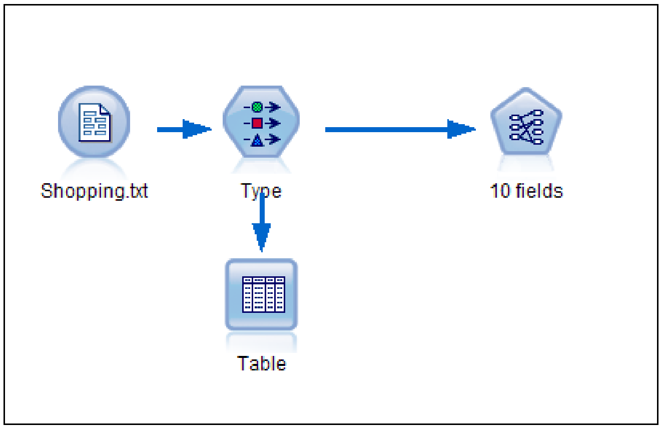
An Apriori Example

Now that we've reviewed the expert options, we will run some examples of the Apriori algorithm, in preparation for additional examples to be run during the exercises. The first step is to load a preexisting stream into PASW Modeler.

Delete the **Apriori** node in the Stream canvas
Open the stream file **Apriori.str** in the c:\Train\ModelerClusAssoc directory
Run the **Table** node to insure that the field types are fully instantiated and view the data
Close the **Table** window

The data file is *Shopping.txt*, which we used in Lesson 3. It contains demographic information on shoppers plus 10 product fields, coded as either 1 or 0 to indicate whether a product in that category was purchased or not.

Figure 4.2 Apriori Stream

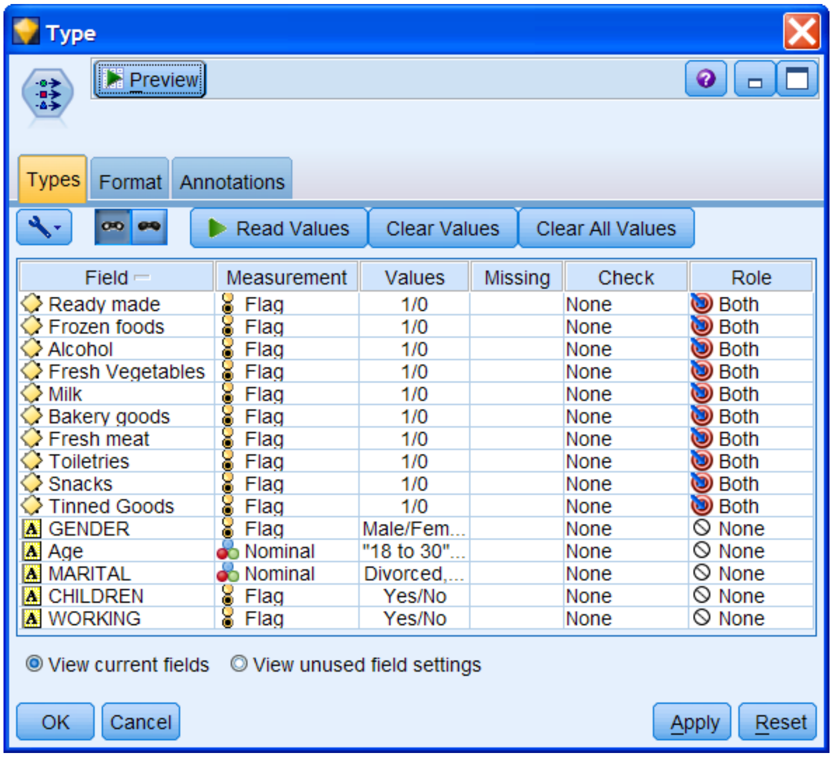


We should check the field definitions in the Type node or the source node.

Double-click the **Type** node

As shown below, all the product fields are correctly set to flags, with role BOTH. The demographics have role set to NONE because they won't be used in these examples.

Figure 4.3 Field Types Set for Association Rule Analysis



Field	Measurement	Values	Missing	Check	Role
Ready made	Flag	1/0		None	Both
Frozen foods	Flag	1/0		None	Both
Alcohol	Flag	1/0		None	Both
Fresh Vegetables	Flag	1/0		None	Both
Milk	Flag	1/0		None	Both
Bakery goods	Flag	1/0		None	Both
Fresh meat	Flag	1/0		None	Both
Toiletries	Flag	1/0		None	Both
Snacks	Flag	1/0		None	Both
Tinned Goods	Flag	1/0		None	Both
GENDER	Flag	Male/Fem...		None	None
Age	Nominal	"18 to 30"...		None	None
MARITAL	Nominal	Divorced,...		None	None
CHILDREN	Flag	Yes/No		None	None
WORKING	Flag	Yes/No		None	None

● View current fields ○ View unused field settings

Close the **Type** node window

To get a baseline for comparison, the first task is to run Apriori without changing any expert options.

Double-click the **Apriori** node
Change the **Minimum Rule Confidence** to **75%** (not shown)
Click the **Run** button

After the generated model appears in the Models manager

Right-click the **Apriori generated model** node
Click **Browse** from the Context menu
Click **Show/Hide Criteria** button ▦, then click **Show all**

There are 26 rules produced, with up to four antecedents, sorted from high to lower confidence. Interestingly, there are no simple rules with only one antecedent. A majority of the rules are for the conclusion of bakery goods or ready-made products, and all the rules, of course, have a posterior confidence (*Confidence*) greater than 75%.

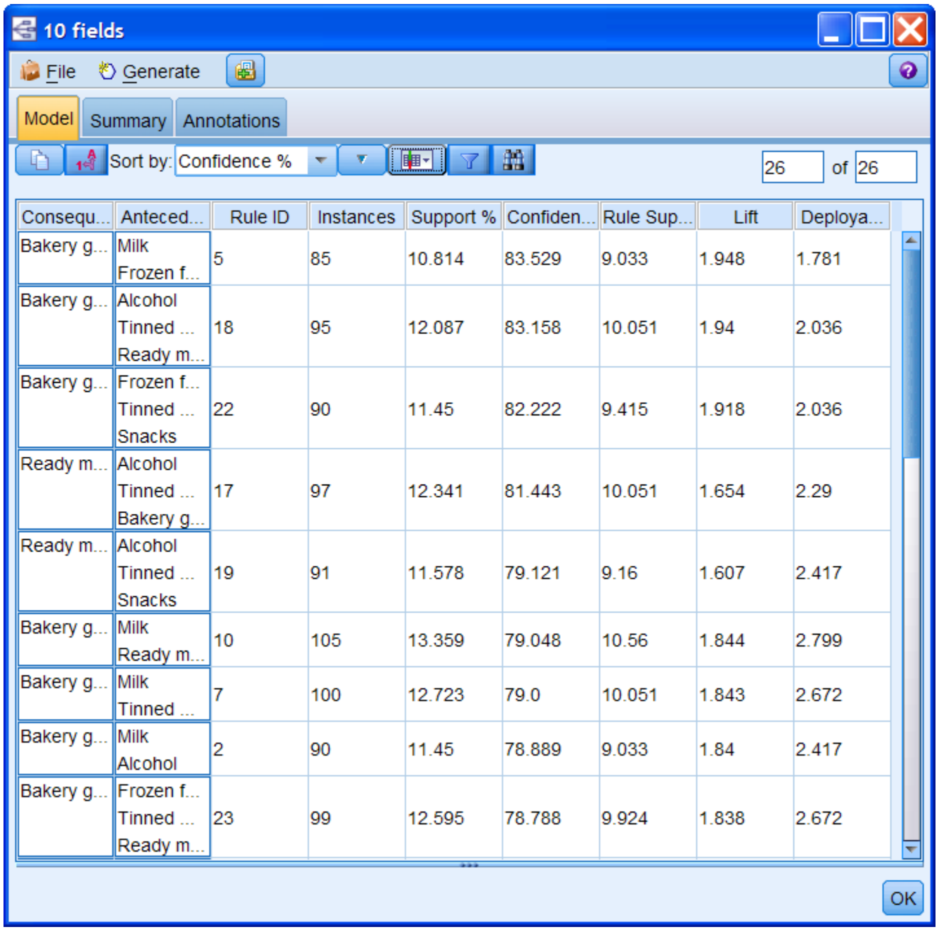
In addition to the measures we discussed above, we note that:

Instances is the number of records to which the antecedents apply, the number upon which Support % is based.

Lift displays the *ratio* of confidence for the rule to the prior probability of having the consequent. Larger numbers are better.

Deployability is a measure of what percentage of the data satisfies the conditions of the antecedent but does not satisfy the consequent. The deployability statistic is defined as (Antecedent Support in # of Records – Rule Support in # of Records) / Total Number of Records. Smaller numbers are generally better.

Figure 4.4 Apriori Rules Using Rule Confidence Measure



Sort by: Confidence %

26 of 26

Consequ...	Anteced...	Rule ID	Instances	Support %	Confiden...	Rule Sup...	Lift	Deploya...
Bakery g...	Milk Frozen f...	5	85	10.814	83.529	9.033	1.948	1.781
Bakery g...	Alcohol Tinned ... Ready m...	18	95	12.087	83.158	10.051	1.94	2.036
Bakery g...	Frozen f... Tinned ... Snacks	22	90	11.45	82.222	9.415	1.918	2.036
Ready m...	Alcohol Tinned ... Bakery g...	17	97	12.341	81.443	10.051	1.654	2.29
Ready m...	Alcohol Tinned ... Snacks	19	91	11.578	79.121	9.16	1.607	2.417
Bakery g...	Milk Ready m...	10	105	13.359	79.048	10.56	1.844	2.799
Bakery g...	Milk Tinned ...	7	100	12.723	79.0	10.051	1.843	2.672
Bakery g...	Milk Alcohol	2	90	11.45	78.889	9.033	1.84	2.417
Bakery g...	Frozen f... Tinned ... Ready m...	23	99	12.595	78.788	9.924	1.838	2.672

Now we're ready to try some of the expert options for Apriori. We'll begin with the confidence difference, the simplest expert evaluation measure. Since the measure is based on the prior confidence, we should have some idea of the prior confidence or relative frequency of the different products. We could get this information from the Data Audit or Distribution nodes, or by checking the *Allow rules without antecedents* check box, but to save time, the prior confidences are displayed in the table below, in descending order.

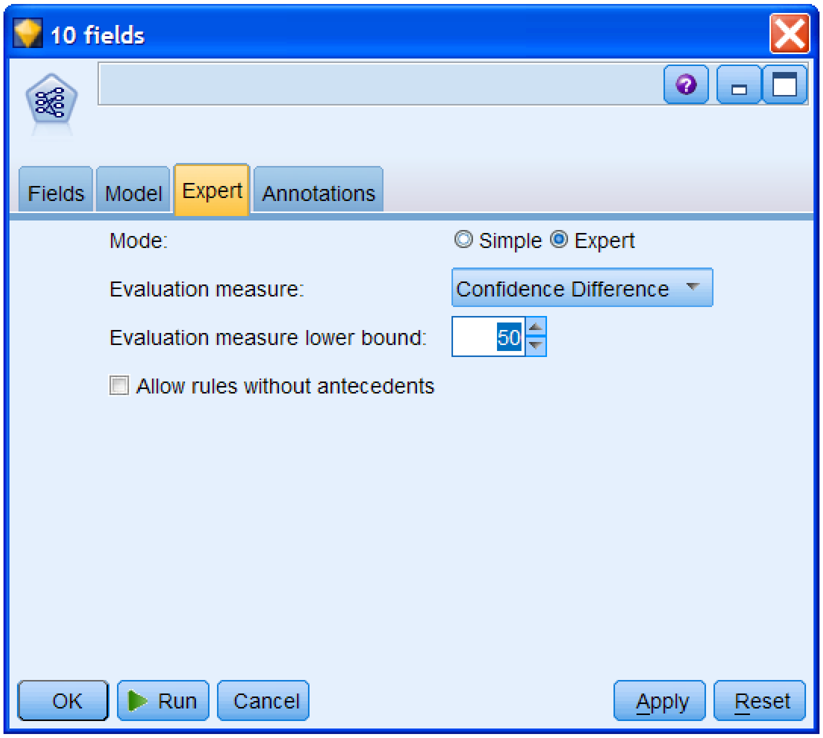
Product	Prior Confidence
Ready-made	49.2
Snacks	47.5
Tinned goods	45.5
Bakery goods	42.9
Frozen foods	40.2
Alcohol	39.4
Milk	18.8
Toiletries	10.0
Fresh Vegetables	8.2
Fresh Meat	2.9

Seeing these values also helps to clarify why the association rules discovered in the first model do not contain any toiletries, fresh vegetables, or fresh meat items. Their support values are too low to be included. This example should make evident why examining baseline data like this is necessary to fully understand a set of association rules. For example, we might reduce the minimum rule support to try to incorporate these products in the rules (this is best done using business knowledge as well).

With this knowledge in hand, we can turn to the expert settings for the Apriori node.

Close the **Association Rules** browser window
Double-click the **Apriori** node
Click the **Expert** tab
Click the **Expert** Mode option button
Select the **Confidence difference** on the **Evaluation measure** drop-down list
Set the **Evaluation measure lower bound** value to **50**

Figure 4.5 Apriori Dialog with Expert Options



Mode: ○ Simple ● Expert

Evaluation measure: Confidence Difference

Evaluation measure lower bound: 50

□ Allow rules without antecedents

Since we want to see the effect of this measure only, we'll need to change the rule support and rule confidence settings to 0 or its minimum.

Click **Model** tab
Change **Minimum antecedent support** to **0**
Change **Minimum rule confidence** to 5.0 (the minimum) (not shown)

The key decision in this example is the value of the *Evaluation measure lower bound*. Recall that for this evaluation measure, the lower bound refers to the absolute value of the difference between the prior and posterior confidences. Referring to the table of prior confidences, we have two distinct groups, with several products between about 40% and 50%, and then another set below 20%. Since the highest confidence was about 83, we might suspect that an Evaluation measure lower bound of 50 will probably find essentially no rules for the products that are purchased more often. It might find rules for the other group of products, but it would be more likely to find negative rules (except that the default setting is still to use only true values for flags).

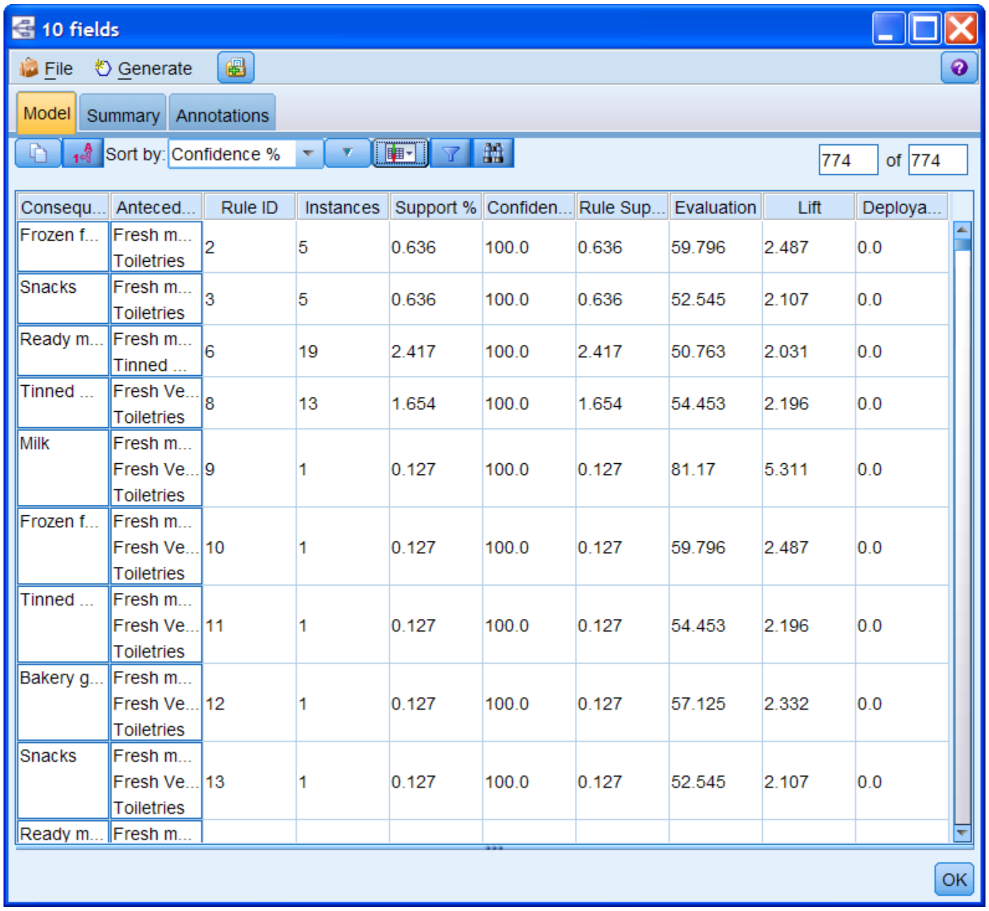
Despite this reasoning, let's run a model with the value of 50% for the lower bound. Remember that with the support set to 0%, there is a countervailing tendency to include many rules.

Click the **Run** button
After the model has run, right-click the **Apriori generated model** in the Models manager
Click **Browse** from the Context menu
Click **Show/Hide Criteria** button, and then click **Show all**

We have found many more than 26 association rules (774). And the rules, for the most part, have very high levels of confidence but very low levels of support. To see what has happened, consider the first rule, with antecedents of fresh meat and toiletries and a conclusion of frozen foods. Five shoppers bought both of the antecedents, and of these 5, 100% bought frozen foods (the posterior confidence of 100). The prior confidence of ready-made goods is 40.2, so the absolute difference between the two (100 – 40.2 or 59.8) is greater than 50, the lower bound for the confidence difference measure. The *Evaluation* column displays this difference, and all values in it will be greater than 50 (Why?). Of course, 5 shoppers constitute only about 0.6% of the file, and this rule may or may not be interesting from a business perspective. The Lift value is the ratio of the confidence (posterior) to the prior confidence (100/40.2 or 2.487).

We did manage to include in the rules those products that were excluded before, but there is no overlap between the rules generated in these two models. And realistically, the first set of rules based on the rule confidence may be more useful because of support. As mentioned in our earlier discussion, tweaking the settings for the expert evaluation measures is necessary to find interesting and useful rules.

Figure 4.6 Apriori Rules Using Confidence Difference Evaluation Measure



Sort by: Confidence %

774 of 774

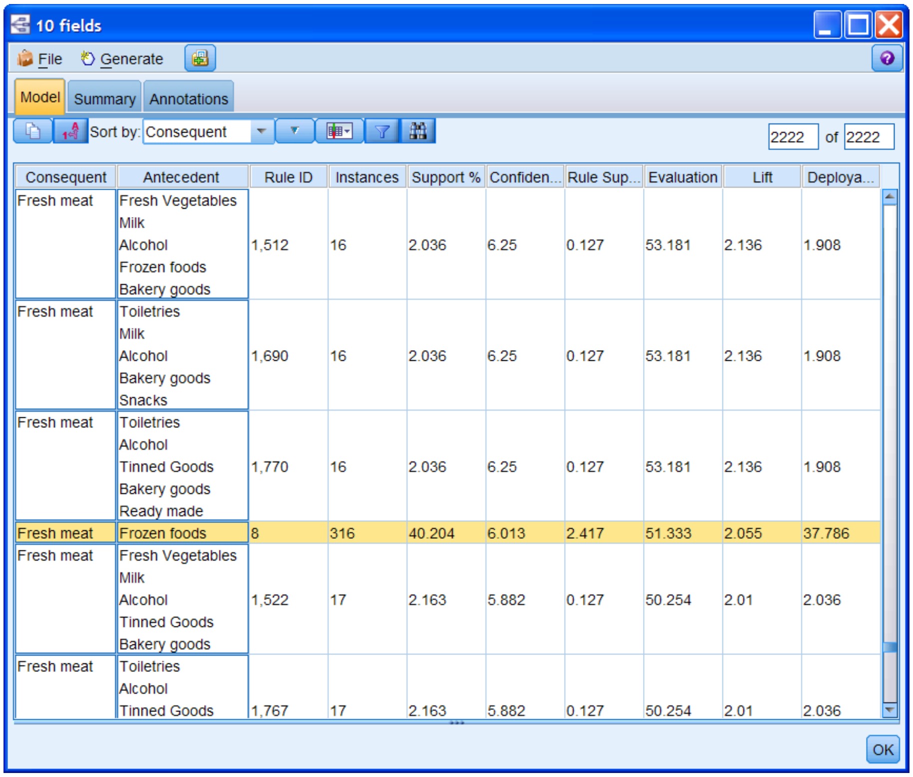
Consequ...	Anteced...	Rule ID	Instances	Support %	Confiden...	Rule Sup...	Evaluation	Lift	Deploya...
Frozen f...	Fresh m... Toiletries	2	5	0.636	100.0	0.636	59.796	2.487	0.0
Snacks	Fresh m... Toiletries	3	5	0.636	100.0	0.636	52.545	2.107	0.0
Ready m...	Fresh m... Tinned ...	6	19	2.417	100.0	2.417	50.763	2.031	0.0
Tinned ...	Fresh Ve... Toiletries	8	13	1.654	100.0	1.654	54.453	2.196	0.0
Milk	Fresh m... Fresh Ve... Toiletries	9	1	0.127	100.0	0.127	81.17	5.311	0.0
Frozen f...	Fresh m... Fresh Ve... Toiletries	10	1	0.127	100.0	0.127	59.796	2.487	0.0
Tinned ...	Fresh m... Fresh Ve... Toiletries	11	1	0.127	100.0	0.127	54.453	2.196	0.0
Bakery g...	Fresh m... Fresh Ve... Toiletries	12	1	0.127	100.0	0.127	57.125	2.332	0.0
Snacks	Fresh m... Fresh Ve... Toiletries	13	1	0.127	100.0	0.127	52.545	2.107	0.0
Ready m...	Fresh m...								

You'll get a chance to try different settings in the exercises. We'll conclude our discussion of the Apriori expert settings by creating a model with another evaluation measure.

Close the **Association Rules** browser window
Double-click on the **Apriori** modeling node to edit it
Click the **Expert** tab
Select **Confidence Ratio** on the **Evaluation measure** drop-down list (not shown)
Leave the **Evaluation measure lower bound** at **50**

Click the **Run** button
After the model has run, right-click the **Apriori generated model** node in the Models manager
Click **Browse** from the Context menu
Click **Show/Hide Criteria** button , and then click **Show all**
Click **Consequent** on the Sort by drop-down list
Scroll down about 90% of the way through the list of rules until the rule with **Fresh Meat** as the consequent and **Frozen Foods** as the antecedent is visible (near the bottom of the rules for Fresh Meat)

Figure 4.7 Apriori Rules Using Confidence Ratio



Consequent	Antecedent	Rule ID	Instances	Support %	Confiden...	Rule Sup...	Evaluation	Lift	Deploya...
Fresh meat	Fresh Vegetables Milk Alcohol Frozen foods Bakery goods	1,512	16	2.036	6.25	0.127	53.181	2.136	1.908
Fresh meat	Toiletries Milk Alcohol Bakery goods Snacks	1,690	16	2.036	6.25	0.127	53.181	2.136	1.908
Fresh meat	Toiletries Alcohol Tinned Goods Bakery goods Ready made	1,770	16	2.036	6.25	0.127	53.181	2.136	1.908
Fresh meat	Frozen foods	8	316	40.204	6.013	2.417	51.333	2.055	37.786
Fresh meat	Fresh Vegetables Milk Alcohol Tinned Goods Bakery goods	1,522	17	2.163	5.882	0.127	50.254	2.01	2.036
Fresh meat	Toiletries Alcohol Tinned Goods	1,767	17	2.163	5.882	0.127	50.254	2.01	2.036

This model also has a large number of rules! (2222), but some are of a different sort than the previous model. For example, the Fresh Meat ← Frozen Foods rule (highlighted in Figure 4.7) has support of 40.2%, quite large, but confidence of only about 6% (those who buy frozen foods also buy fresh meat on about 6% of all trips). It was selected because the prior confidence is 2.9% for fresh meat, and (1 - .029 / .06)*100 is about 51 (see the Evaluation value), which is greater than the evaluation measure lower bound of 50. This is an illustration of how the confidence ratio measure can find rules with low prior confidences that are interesting. This rule has a large support value because so many shoppers buy frozen foods, and it isolates an association where the likelihood of buying fresh meat doubles from the baseline value. The relationship could be useful from the perspective of grocery store management. How small could the posterior confidence have been for this association to still be selected?

This evaluation measure can also find rules similar to that found by the absolute confidence difference, as evidenced by the rule Frozen Foods <= Fresh Meat (not shown). It applies to only 23 shoppers (all those who buy meat), and states that these people also buy frozen foods in 82.6% of all trips.

As we noted above, you can only interpret the rules and decide whether they are important by referring to the prior confidences for the items. We haven't yet allowed Apriori to use false values for flag variables, but you can try that in the exercises. This completes our exploration of the Apriori algorithms. We turn next to the Carma node.

4.6 *Carma Expert Options*

The Carma node works only with fields of storage type string (discussed further below). When using tabular data, all fields in the model must be of measurement flag. The role of a field is irrelevant, and Carma doesn't recognize Input, Target, or Both field roles (fields with role None and Typeless are ignored). All fields used will be treated as if their role is set to Both. Carma, unlike Apriori, allows rules with multiple consequents, which can be a distinct advantage, but can also create more complicated rules.

As noted above, Carma is a very efficient algorithm for generating association rules. As a consequence, it uses a very different approach than Apriori. On a first pass through the data, Carma identifies frequent sets of items and creates a lattice of parent and child itemsets. For example, if X = {milk, cheese, bread} is an itemset, then Y = {milk, cheese} is a parent of X, and Z = {milk, cheese, bread, sugar} is a child of X.

During the first pass, to help conserve on memory, it periodically prunes the lattice of itemsets, removing small itemsets with lower support. The user can control this parameter.

On a second pass of the data, Carma computes the exact frequencies for the candidate itemsets, and then generates rules from the remaining itemsets. The algorithm employed tends to eliminate redundant rules (see Aggarwal, C. C., and P. S. Yu. 1998. "Online generation of association rules," in *Proceedings of the 14th International Conference on Data Engineering,* Los Alamitos, CA: IEEE Computer Society Press, 402–411).

The most direct method to control the creation of rules is through changing the support and confidence values.

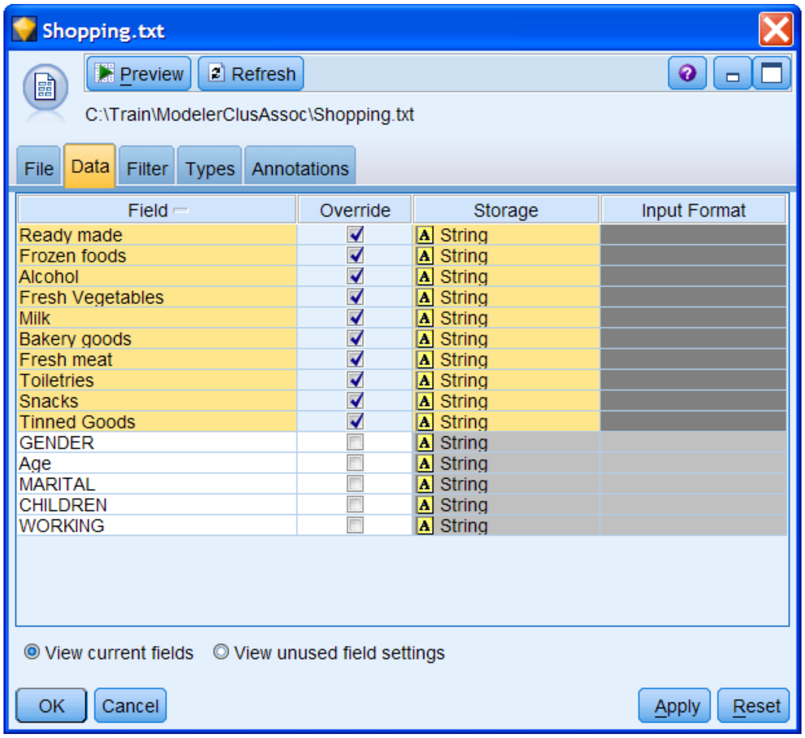
Carma Node Options and Example

We will add a Carma node to the stream. Since the product fields are numeric (coded 0 and 1), we also need to change them to storage type string.

Close the **Association rules** browser window
Add a **Carma** node to the Stream canvas below the shopping.txt node
Edit the **Shopping.txt** node
Click on the **Data** tab
Select all **10 product fields**
Right-click on any selected field and select **Override…true** from the context menu
Right-click again and select **Set Storage…String**

Changing the storage of a field to string (even when that field contains numeric data), allows us to use Carma to generate association rules. However, the Type node in the stream recognizes the previous storage setting for these fields. It will be easier to simply add a new Type node to the stream and fully instantiate the data in this second Type node.

Figure 4.8 Storage Changed to String for Product Fields



C:\Train\ModelerClusAssoc\Shopping.txt

Field	Override	Storage	Input Format
Ready made	☑	String	
Frozen foods	☑	String	
Alcohol	☑	String	
Fresh Vegetables	☑	String	
Milk	☑	String	
Bakery goods	☑	String	
Fresh meat	☑	String	
Toiletries	☑	String	
Snacks	☑	String	
Tinned Goods	☑	String	
GENDER	☐	String	
Age	☐	String	
MARITAL	☐	String	
CHILDREN	☐	String	
WORKING	☐	String	

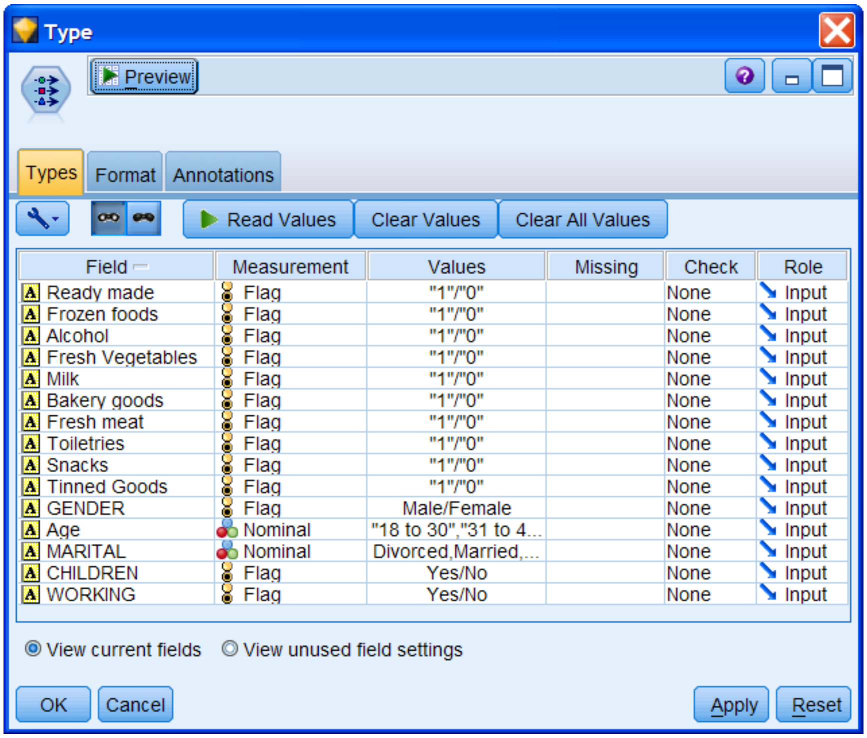
● View current fields ○ View unused field settings

Click **OK**
Add a **Type** node to the stream between the data source and the Carma node
Connect the **data source** to the **Type** node, and the **Type** node to the **Carma** node
Add a **Table** node to the stream and connect it to the new **Type** node
Run the **Table** node
Close the Table window
Edit the **Type** node

Notice how the data have not changed, but now the values of 1 and 0 for the product fields are in quotes in the Values column, indicating that they are now stored as strings by PASW Modeler.

We could change the role of the product fields, or more to the point, set the role of the other fields to None, but we will instead specify the exact fields to be used in the Carma node itself.

Figure 4.9 Type Node with Product Fields as Strings



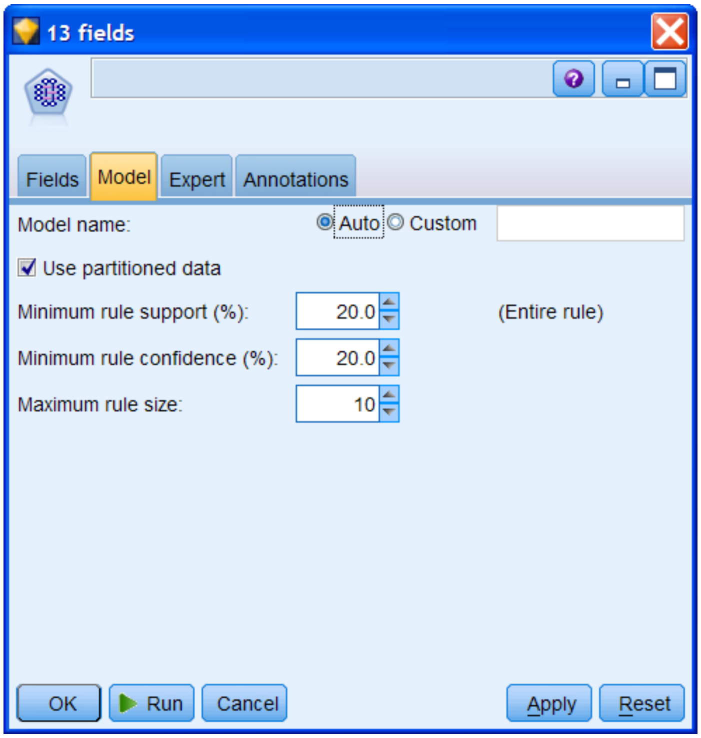
Field	Measurement	Values	Missing	Check	Role
Ready made	Flag	"1"/"0"		None	Input
Frozen foods	Flag	"1"/"0"		None	Input
Alcohol	Flag	"1"/"0"		None	Input
Fresh Vegetables	Flag	"1"/"0"		None	Input
Milk	Flag	"1"/"0"		None	Input
Bakery goods	Flag	"1"/"0"		None	Input
Fresh meat	Flag	"1"/"0"		None	Input
Toiletries	Flag	"1"/"0"		None	Input
Snacks	Flag	"1"/"0"		None	Input
Tinned Goods	Flag	"1"/"0"		None	Input
GENDER	Flag	Male/Female		None	Input
Age	Nominal	"18 to 30","31 to 4...		None	Input
MARITAL	Nominal	Divorced,Married,...		None	Input
CHILDREN	Flag	Yes/No		None	Input
WORKING	Flag	Yes/No		None	Input

● View current fields ○ View unused field settings

Click **OK**
Edit the **Carma** node
Click the **Field** tab, and then click **Use custom settings**
Select **all ten product fields** as Inputs (not shown)
Click the **Model** tab

Unlike Apriori, Carma allows the user to control the *Minimum rule support %*, which applies to both the antecedents and the consequent. The default value is 20%. The *Minimum rule confidence* is set lower than either of the other two association models, also at 20%. The maximum rule size is set to 10, which refers to the maximum number of itemsets (a consequent with two items is one itemset, for example). There is no setting for the maximum number of rules, which is essentially unlimited.

Figure 4.10 Carma Model Options



Model name: ● Auto ○ Custom

☑ Use partitioned data

Minimum rule support (%): 20.0 (Entire rule)

Minimum rule confidence (%): 20.0

Maximum rule size: 10

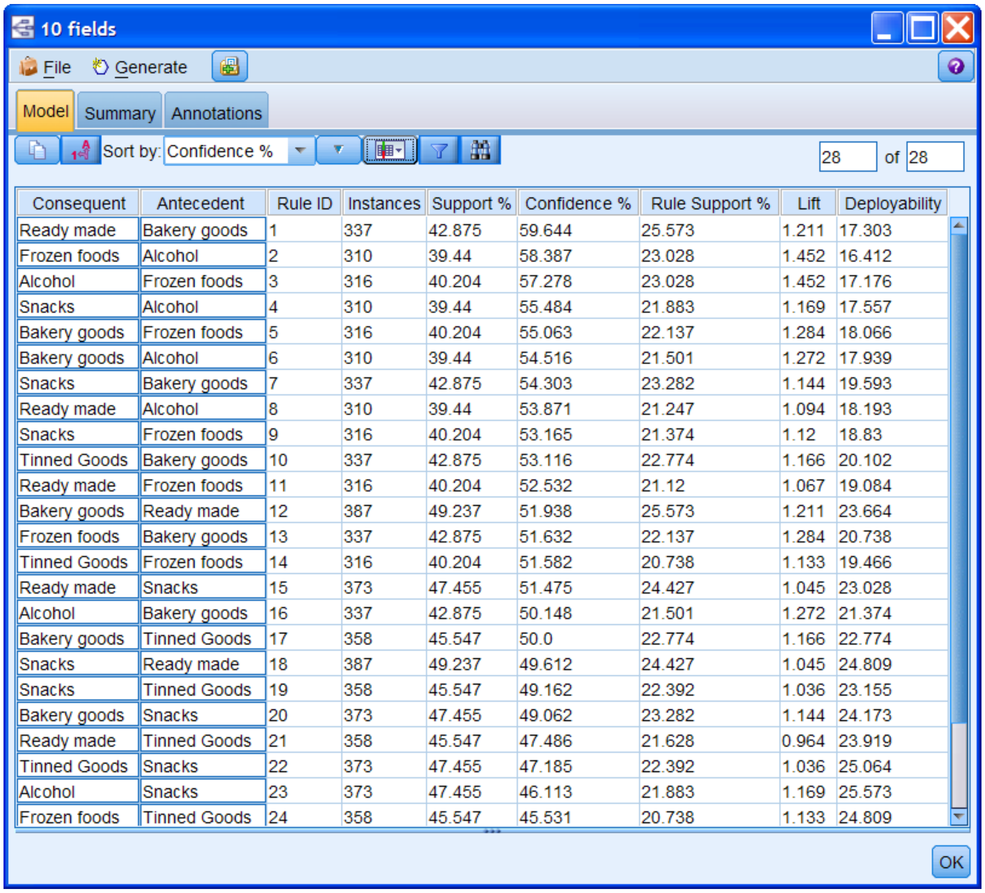
We will review the expert options, but first let's run Carma with the default settings.

Click the **Run** button
Browse the **Carma generated model** node in the Models Manager window
Click the **Show/Hide Criteria** button, then click **Show all**

Twenty-eight rules were generated, compared to 26 for Apriori. The rules, though, are quite different than those generated by default by Apriori, as they are much simpler, with only one antecedent in every case. The reason is the fairly severe requirement that the minimum rule support be at least 20%, which means that a rule must apply to at least one-fifth of the file. As a result, the highest confidence value is 59.64 (the rule support is 25.57% for this rule).

As always, the input parameters of a model have a tremendous influence on the types of rules that are generated.

Figure 4.11 Carma Association Rules with Default Settings



Sort by: Confidence %

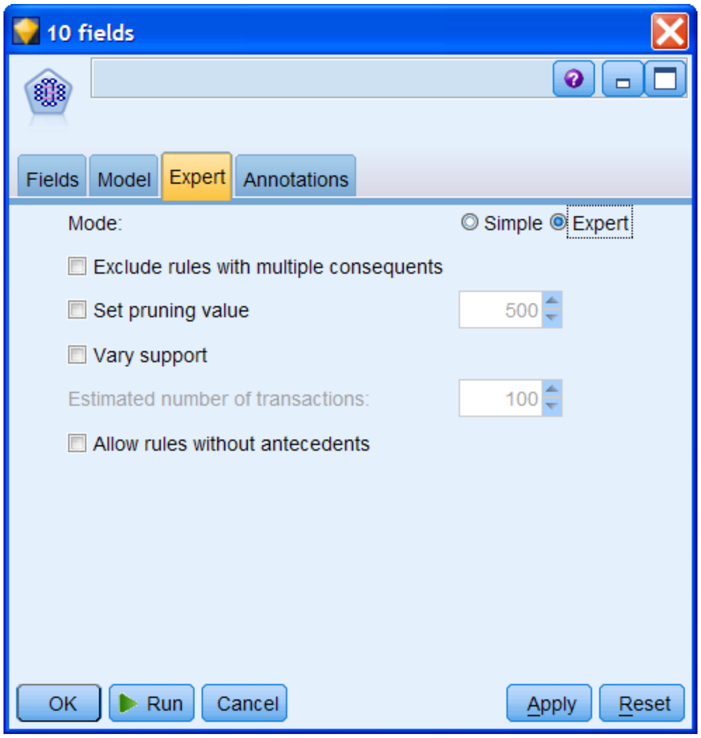
28 of 28

Consequent	Antecedent	Rule ID	Instances	Support %	Confidence %	Rule Support %	Lift	Deployability
Ready made	Bakery goods	1	337	42.875	59.644	25.573	1.211	17.303
Frozen foods	Alcohol	2	310	39.44	58.387	23.028	1.452	16.412
Alcohol	Frozen foods	3	316	40.204	57.278	23.028	1.452	17.176
Snacks	Alcohol	4	310	39.44	55.484	21.883	1.169	17.557
Bakery goods	Frozen foods	5	316	40.204	55.063	22.137	1.284	18.066
Bakery goods	Alcohol	6	310	39.44	54.516	21.501	1.272	17.939
Snacks	Bakery goods	7	337	42.875	54.303	23.282	1.144	19.593
Ready made	Alcohol	8	310	39.44	53.871	21.247	1.094	18.193
Snacks	Frozen foods	9	316	40.204	53.165	21.374	1.12	18.83
Tinned Goods	Bakery goods	10	337	42.875	53.116	22.774	1.166	20.102
Ready made	Frozen foods	11	316	40.204	52.532	21.12	1.067	19.084
Bakery goods	Ready made	12	387	49.237	51.938	25.573	1.211	23.664
Frozen foods	Bakery goods	13	337	42.875	51.632	22.137	1.284	20.738
Tinned Goods	Frozen foods	14	316	40.204	51.582	20.738	1.133	19.466
Ready made	Snacks	15	373	47.455	51.475	24.427	1.045	23.028
Alcohol	Bakery goods	16	337	42.875	50.148	21.501	1.272	21.374
Bakery goods	Tinned Goods	17	358	45.547	50.0	22.774	1.166	22.774
Snacks	Ready made	18	387	49.237	49.612	24.427	1.045	24.809
Snacks	Tinned Goods	19	358	45.547	49.162	22.392	1.036	23.155
Bakery goods	Snacks	20	373	47.455	49.062	23.282	1.144	24.173
Ready made	Tinned Goods	21	358	45.547	47.486	21.628	0.964	23.919
Tinned Goods	Snacks	22	373	47.455	47.185	22.392	1.036	25.064
Alcohol	Snacks	23	373	47.455	46.113	21.883	1.169	25.573
Frozen foods	Tinned Goods	24	358	45.547	45.531	20.738	1.133	24.809

Next we briefly examine the Expert settings for Carma.

Close the **Association Rules** browser window
Double-click the **Carma** node
Click the **Expert** tab
Click the **Expert** option button

Figure 4.12 Carma Node Expert Options



Mode: ○ Simple ● Expert

□ Exclude rules with multiple consequents

□ Set pruning value 500

□ Vary support

Estimated number of transactions: 100

□ Allow rules without antecedents

If you prefer not to create rules with multiple consequents, *Exclude rules with multiple consequents* can be selected. By default, Carma will not create rules without antecedents, as was discussed for the other association models, but that setting can be modified.

To conserve memory, the Carma algorithm periodically removes, or prunes, infrequent itemsets. The pruning value is set by default to 500, which is the number of transactions processed. Enter a smaller value to decrease the memory requirements (but potentially increase the training time), or enter a larger value to do the reverse. This setting should have little effect on the actual rules generated.

The option to *Vary support* will, if selected, increase the efficiency of Carma by literally changing the target support value as transactions are processed. The support values start higher than the user-supplied value and are reduced down to that setting.

Because Carma allows the user to specify the minimum rule support, not the antecedent support, we won't attempt to rerun Carma with similar settings to Apriori (although you can certainly compare the algorithms in the exercises for this lesson).

4.7 *Choosing a Method and Expert Options*

PASW Modeler provides three models to discover a set of association rules. Two, Apriori, Carma, were discussed in this lesson. Two natural questions are then "When should I use each of these models?" For which situations is one model more suitable?" Second, several settings can be changed for each model, more if the expert options are used. Can any advice be given about these?

It might seem that any algorithm will find the same rules in a given data file, but this isn't exactly true. Given how many rules are possible in a data file with many fields and thousands of records, and

given that each model takes a different approach to sift through the possible rule candidates, these two methods are not guaranteed to find exactly the same rules in a file.
To help clarify these issues, the following table summarizes the available settings for each method.

Table 4.1 Comparison of Association Modeling Nodes

Setting	Apriori	Carma
Method of Rule Selection	Five types available	Frequent Itemsets
Respects Field Role	Yes	No
Number of rules	No maximum	No maximum
Antecedent Support	0 to 100%	N/A
Rule Support	N/A	0 to 100%
Rule Confidence	0 to 100%	0 to 100%
Maximum Number of Antecedents	32	N/A
Maximum Rule Size	N/A	10 (itemsets)
Use only True values for flags	Optional	Mandatory

First, of course, Apriori can handle numeric fields as antecedents in rules, so clearly it is the preferred method for datasets with numeric fields. Carma has the limitation that the data be of measurement flag (and storage type string) with tabular data, so clearly that can affect which method you choose. But, if the data are changed to transactional format, Carma can use fields of nominal measurement type.

Although Carma can be more efficient than Apriori, unless datasets are very large, speed is usually not a major issue when generating association rules sets, and so is probably not a determining factor.

If you wish to have a choice of rule selection method, clearly Apriori will be preferred because it gives the user more control over the method of rule generation.

If you prefer to focus on how general a rule is, including both antecedents and consequent(s), then Carma allows direct control of this setting. If instead you want to focus on the frequency of the antecedents, you may decide to use either Apriori. If you wish to focus on only certain consequents or antecedents, Apriori can provide more control because they respect the role of fields in the Type node. Rules have to be filtered for specific items with Carma.

Clearly, if creating rules for false values of flag fields is interesting, you will prefer Apriori (although you can reclassify the data values so that true and false values are switched, then use Carma).

Beyond these general guidelines, to determine how to set support and confidence, you need to consider whether you:

- Wish to highlight rare combinations
- Are interested in only rules with high confidence and/or support
- Are interested in rules with only a few antecedents, or many
- Wish to find rules with a large absolute, or relative, difference between the prior and posterior confidence

Based on your answers to these questions, you can pick a method and some initial settings for the various parameters. But you will invariably have to experiment and tweak the settings to find an optimal set of rules. And you will probably want to use at least two of these models and compare results.

4.8 *Missing Data with Association Rules*

There are no missing data for the product fields in the shopping file. However, missing or blank data are a fact of life in most circumstances. Normally, there are three methods for handling missing data in PASW Modeler when developing models.

1. Include the missing data in the model directly
2. Estimate the missing data values, then include them in the model
3. Delete the missing data (records) in the stream before constructing a model.

Some modeling techniques in PASW Modeler are designed to accommodate missing data. For example C&R Tree uses a substitute predictor to advance a record with a missing value for a predictor down the decision tree. Alternatively, missing a value for income may be predictive of an outcome, and the missing value could be included as an additional category in a decision tree analysis. Other techniques, such as neural nets or regression, are not designed to handle missing data directly, and the missing data either needs to be eliminated or modified somehow before modeling begins (which is done automatically by Neural Net).

Blanks are ignored by the Association rule algorithms. The algorithms will handle records containing blanks for input fields, but such a record will not be considered to match any rule containing one or more of the fields for which it has blank values.

Most of the time when using Apriori, we will either not define data as blanks, or we will define data as blanks and then remove the records with blank values. Either one of these approaches will lead to no confusion in the association rules created. But if, instead, blanks are defined and then fed into either algorithm, the rules and statistics calculated might not be what we expect.

Summary Exercises

This set of exercises is written around the following data file: *shopping.txt*. The following text gives details of the file.

shopping.txt contains information on items purchased by customers of a supermarket on single shopping trips. There are 10 different product classifications along with some demographic information about the customer. The primary interest of the supermarket was to understand which customers products are bought by different groups of people. The file contains the following fields:

Ready made	Did the customer buy Ready made meals
Frozen Food	Did the customer buy Frozen food
Alcohol	Did the customer buy Alcohol
Fresh Vegetables	Did the customer buy Fresh vegetables
Milk	Did the customer buy Milk
Bakery goods	Did the customer buy Bakery goods
Fresh meat	Did the customer buy Fresh meat
Toiletries	Did the customer buy Toiletries
Snacks	Did the customer buy Snacks
Tinned Goods	Did the customer buy Tinned goods
Age	Age
MARITAL	Marital status
CHILDREN	Number of Children
WORKING	Currently employed?

In these exercises you can continue to use the stream called *Apriori.str* that was used in this lesson.

1. The first step is to build a set of association rules with the two evaluation measures available under expert options in Apriori that we didn’t use in the lesson examples. Run each one with the current settings for support, confidence, and the evaluation measure lower bound. Is a generated model created? If not, change the lower bound settings until a model is created. What value of the lower bound was required?

2. Compare the rules generated by the information difference and chi-square methods to the other association rules from the first two methods in Apriori. How would you describe the differences in the rules found by these two methods?

3. Now try creating association rules using the false values for flag variables for each evaluation measure, including rule confidence. Be sure to set the evaluation measure lower bound at an appropriate value for each method. How does adding false flag values change the rules generated? To reduce the number of rules generated, pick one algorithm and vary the settings for rule support, confidence, and the lower bound, to find a useful and reasonably sized set of rules.

4. To see how association models can incorporate different sets of variables, add the demographic variables to the model. First, make a copy of the Type node, disconnect the original one, and then connect the copy to the data source and Apriori node. Now change the role of all the fields in the Type node to INPUT, then change the role of Alcohol to TARGET. Change the measurement of *GENDER*, *CHILDREN*, and *WORKING* to Nominal so that both of their values will be used in the rules. Use a Distribution node to examine the

data for each of the demographic fields. Then create association rules with the evaluation measure of your choice and settings, but be sure to use only true values for flags. What settings were required to include demographics in the rules? What associations did you find between buying alcohol and the demographic fields? As you interpret the data, keep in mind the prior confidence for alcohol.

5. Now try the Carma node with this data. Remember that you will need to change the storage of the product fields to string as we did in the lesson. Use all the flag variables in the file and run a model. Examine the rules. Then rerun the model and check Vary support under Expert options. What difference did this make to the rules you found? Now change the support and/or confidence levels by reducing them by 5%. What difference did this make? Did you find the same type of associations between alcohol, the other products, and the demographics as above?

6. *For those with extra time*. Filter only the rules with alcohol as the consequent and create a ruleset.

7. To better understand how income was incorporated into the rules, investigate the relationship between income and beer with an appropriate technique. What did you find? Does this help explain the rules generated?

Lesson 5: Sequence Detection

Objectives

- Introduce sequence detection methods
- Use the Sequence node to find a set of common sequences
- Interpret the sequence rules and add sequence predictions to the stream

Data

In this lesson we search for common sequences of diagnostic/repair steps needed to resolve problems with a telecom's service. When a customer reports a problem with telecom service, different tests and operations are performed to resolve the problem. In some cases only a single test is needed, but sometimes many steps are required. The company is interested in identifying common sequences needed to resolve service problems, discovering any repeating patterns (repetition of steps), and identifying sequences that were related to failure to resolve the problem. Data codes are masked and values are simulated based on a customer analysis. The data are stored in the tab-separated file *Telrepair.txt*. The file contains three fields: an ID number corresponding to the problem report, a sequence code for each step taken during problem resolution, and a code representing the diagnostic/repair activity in the step. Code 90 represents the reporting of the original problem (all reports should begin with code 90, but not all do). Codes 210 and 299 are termination codes. Code 210 indicates the problem was resolved, while code 299 indicates it was not successfully resolved. Codes between 100 and 195 represent different diagnostic/repair activities. The file contains information on 750 service problems

5.1 *Introduction*

What are the most common sequences of web-clicks when visiting the web site of an online retailer? Does a pattern of retail purchases predict the future purchase of an item? If a manufactured part, insurance claim, or sales order must go through a number of steps to completion, what are the most common sequences, and do any of these involve repeating steps? These questions involve looking for patterns in data that are time ordered. In Lessons 3 and 4 we discussed general association rules; here the event sequence is formally taken into account.

The Sequence node in PASW Modeler performs sequence detection analysis. In this lesson we will use the Sequence node to explore common sequences of diagnostic/repair steps taken when attempting to solve telecom service problems.

Results from a Sequence node analysis will be of the form:

	Antecedent(s)	**Consequent**
Rule 1	**…**	**…**
Rule 2	**…**	**…**
…		
Rule R	**…**	**…**

This is a similar format to the association nodes, although for Sequence there is an ordering to the antecedents and consequent. A natural way of thinking about a sequence rule is shown below:

Antecedent1 > Antecedent2 > … > AntecedentN => Consequent

For example:

Base and Regression Models > Advanced Models => PASW Modeler

Individuals who buy Base and Regression Models and later buy Advanced Models, are likely to later buy PASW Modeler.

The "and" indicates that the two items are members of an item set. Thus, "Base and Regression Models" means that both items were purchased at the same time, while Base is antecedent1 and Regression Models is antecedent2 implies that the purchase of SPSS Base preceded the purchase of Regression Models.

When the Sequence node produces a set of sequences, it provides evaluation measures similar to those we reviewed when we discussed association rules. The measures are called *instances*, *support %*, *rule support %* and *confidence %*. The measures are based not on the number of records in a file but on the number of unique IDs (in the telecom file, the number of distinct problem reports).

The definition of these measures is not quite the same as for association models. For association models, instances are the number of records that match just the antecedents, not the full rule. For Sequence rules, instances refers to the number of IDs for which the full sequence is true.

Support refers to the proportion of IDs to which the antecedents apply— that is, the number of unique cases for which the antecedents appear in the proper order. Confidence refers to that proportion of the cases that contain the antecedent sequence that also contain the consequent. Rule support refers to the percentage of cases for which the antecedents *and* the consequent appear in the correct order.

Thus an example of the full format of a Sequence rule would be as shown below.

Antecedents	Consequent	Instances	Support%	Confidence%	Rule Support%
Base & Regression Models Advanced Models	PASW Modeler	48	15.0	64.0	9.6

Assume there are 500 customers (unique customer IDs) in this dataset. There are 48 customers who purchased SPSS Base and Regression Models at the same time, and then purchased Advanced Models at a later time, and then later purchased PASW Modeler. Thus, the complete rule is true for 48 customers (Instances). These 48 are 9.6% of the total number of customers (Rule Support %). 15.0% of the customers (Support %) had a sequence with the antecedents (Base & Regression Models, followed by Advanced Models). The number of these customers is .15*500 = 75. Of those 75 customers, 64% later bought PASW Modeler (Confidence %), which is equal to 48 customers (Instances), closing the circle.

5.2 *Data Organization for Sequence Detection*

The Sequence node can analyze data in either of two formats. In the Tabular data format, each item is represented by a flag field coded to record the presence or absence of the item. In transactional data format, item values are stored in one or more content fields, usually defined as measurement set fields.

Consider the software transaction example above, in which a customer first purchased SPSS Base and Regression Models, then later purchased Advanced Models, and then purchased PASW Modeler.

It could appear in tabular data format as follows:

Customer	Date	Base	Regression	Adv Models	PASW Modeler	…	Decision Time
101	Feb 2, 2001	T	T	F	F	…	F
101	May 1, 2002	F	F	T	F	…	F
101	Dec 31, 2002	F	F	F	T	…	F

The same sequence in transactional data format would be:

Customer	Date	Purchase
101	Feb 2, 2001	Base
101	Feb 2, 2001	Regression
101	May 1, 2002	Adv Models
101	Dec 31, 2002	PASW Modeler

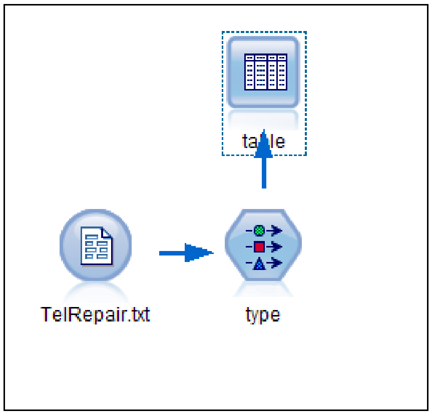
In tabular format, it is clear that SPSS Base and Regression Models were purchased together (they are treated as an item set). In transactional format, the same items would be treated as an item set if the date field were specified as a time field within the Sequence node.

5.3 *The Sequence Node*

We will load a previously defined stream to access the data and then add the Sequence node.

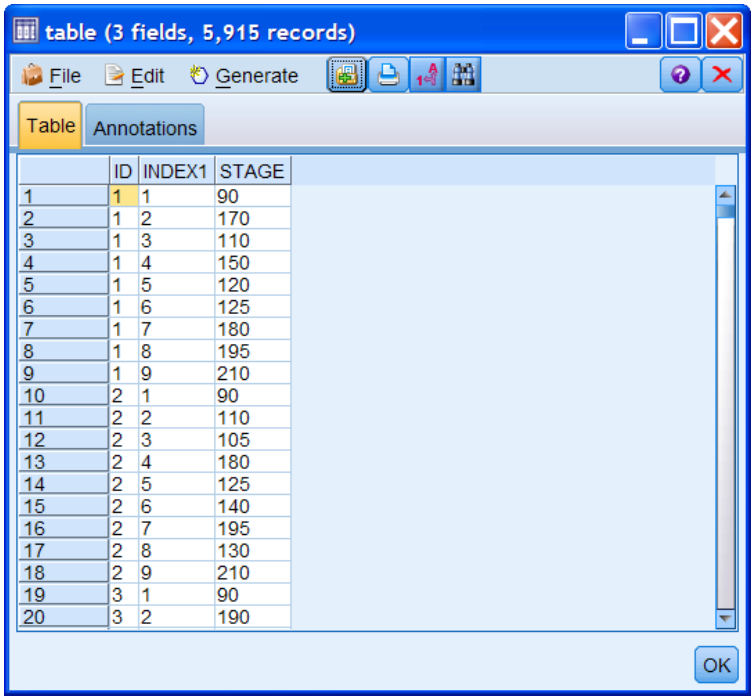
Click **File…Open Stream**, move to the **c:\Train\ModelerClusAssoc** directory and double-click on **Telrepairdef.str**

Figure 5.1 Telrepairdef Stream



Right-click the **Table** node, then click **Run**

Figure 5.2 Telecom Repair Sequence Data



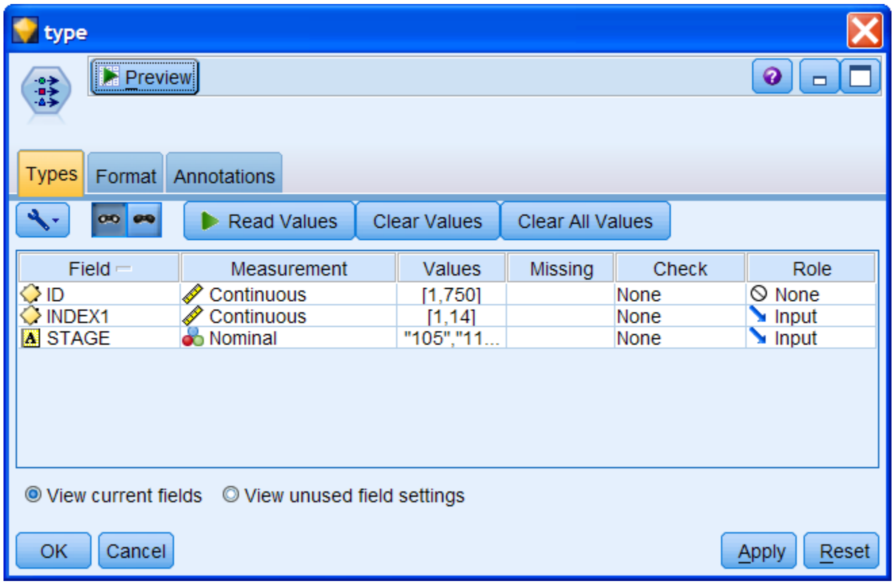
	ID	INDEX1	STAGE
1	1	1	90
2	1	2	170
3	1	3	110
4	1	4	150
5	1	5	120
6	1	6	125
7	1	7	180
8	1	8	195
9	1	9	210
10	2	1	90
11	2	2	110
12	2	3	105
13	2	4	180
14	2	5	125
15	2	6	140
16	2	7	195
17	2	8	130
18	2	9	210
19	3	1	90
20	3	2	190

Each service problem is identified by a unique *ID* value. The field *INDEX1* records the sequence in which the diagnostic/repair steps were performed and the *STAGE* field contains the actual diagnostic/repair codes. All repair sequences should begin with code 90, and a successful repair has 210 as the final code (299 is used if the problem was not successfully resolved).

The data file is presorted by *ID* and by *INDEX1* within *ID*. The Sequence node has an option to sort the data prior to analysis or the Sort node (located in the Record Ops palette) could be used.

Close the **Table** window
Double-click the **Type** node

Figure 5.3 Type Node for Sequence Detection Modeling



Field	Measurement	Values	Missing	Check	Role
ID	Continuous	[1,750]		None	None
INDEX1	Continuous	[1,14]		None	Input
STAGE	Nominal	"105","11...		None	Input

● View current fields ○ View unused field settings

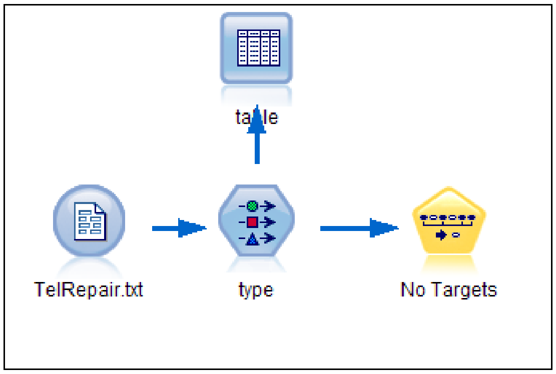
Even though numeric codes are used for the diagnostic/repair values in *STAGE*, this field is declared as measurement nominal. This was done to emphasize that the *STAGE* values represent categories: different diagnostic/testing steps by the sequence detection algorithm. (In order to accomplish this, the storage for *STAGE* was overridden and set to string in the Var. File node.) The analysis could be run with *STAGE* having continuous measurement and if there were a large number of distinct values in the *STAGE* field, then declaring it as continuous measurement would make the stream more efficient. Even if the content field were continuous, the sequence algorithm would treat values as categorical; that is, 90 and 95 would be treated as two categories, not as similar numeric values.

In this analysis a single field, *STAGE*, contains the content to be analyzed. The field(s) containing the content can be of role *Input*, *Target*, or *Both*. If there are multiple content fields, they all must be the same measurement type.

The field (here *ID*) that identifies the unit of analysis can be either numeric or categorical measurement types and can have any role—here it is set to *None*.

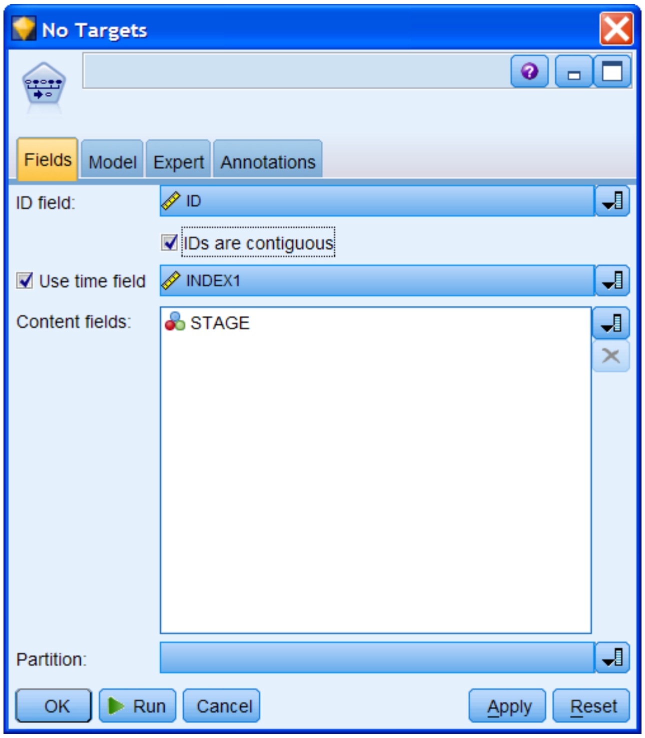
Close the **Type** node
Place the **Sequence** node from the Modeling palette to the right of the **Type** node
Connect the **Type** node to the **Sequence** node

Figure 5.4 Sequence Node Added to Stream



Double-click the **Sequence** node
Select **ID** in the **ID field** box
Click **Use time field** check box (so it is checked)
Select **Index1** in the **Use time field** box
Select **Stage** in the **Content fields** box
Click the **IDs are contiguous** checkbox

Figure 5.5 Sequence Node Dialog



ID field: ID
☑ IDs are contiguous
☑ Use time field INDEX1
Content fields: STAGE
Partition:

By default, the model node is named after the *ID* field and you can change this in the Annotations tab of the Sequence dialog. The *ID* field defines the basic unit of analysis for the Sequence node. In our example, the analysis unit is the service problem and each problem has a unique value for the field named *ID*.

A time field is not required and if no time field is specified, the data are assumed to be time ordered for a given ID. We specify *INDEX1* is the time field for this analysis, although since the data records are ordered by *INDEX1* for each ID, this is not necessary. Under expert options (Expert tab), you have additional controls based on the time field (for example, an event occurring more than a specified interval since another event can be considered to begin a new sequence).

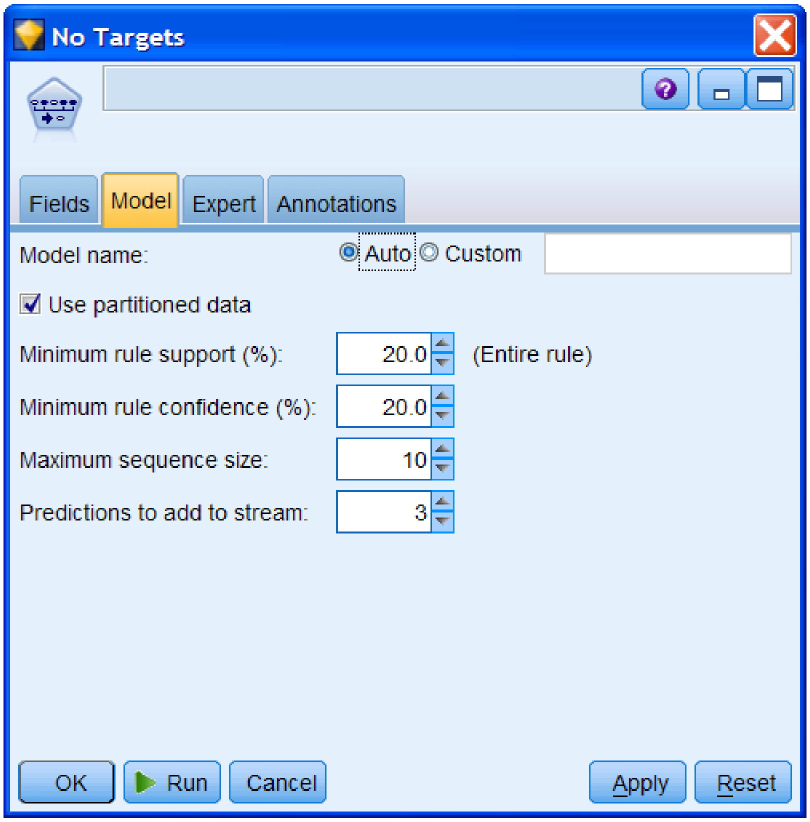
The *Content fields* contain the values that constitute the sequences. In our example, the content is stored in a single field, but multiple fields can be analyzed.

If the data records are already sorted so that all records for an ID are contiguous, you can check the *IDs are contiguous* check box, in which case the Sequence node will not resort the data, saving resources.

Model options provide greater control over various aspects of the analysis. We illustrate these options by examining the Support and Confidence controls.

Click **Model** tab

Figure 5.6 Sequence Node Dialog: Model Tab



No Targets

Model name: ● Auto ○ Custom

☑ Use partitioned data

Minimum rule support (%): 20.0 (Entire rule)

Minimum rule confidence (%): 20.0

Maximum sequence size: 10

Predictions to add to stream: 3

We see that the Model options allow us to set the *Minimum rule support* (default 20%) and *Minimum Rule Confidence* (20%) values for sequences. It is useful to be aware of these values, since as with association rules, depending on the number and distribution of data values, the default settings might produce too many or too few sequences.

Expert options are discussed in the *PASW Modeler User's Guide* and the following lesson of this manual.

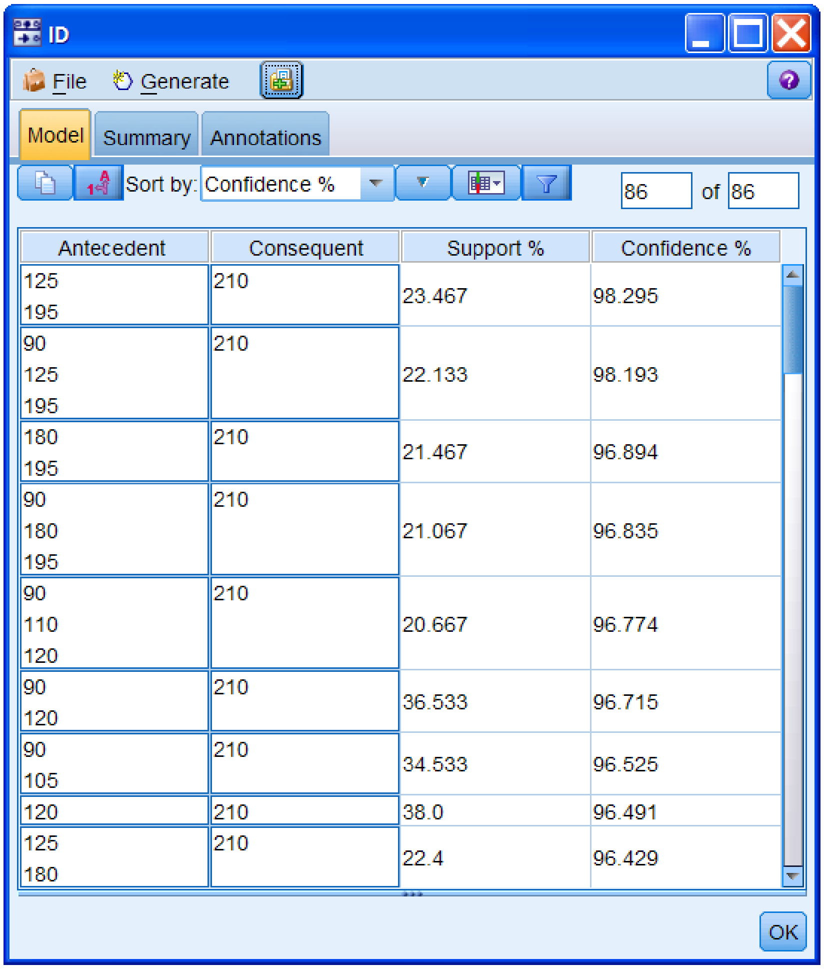
Click **Run**

5.4 *Exploring Sequences*

When the sequence detection analysis is complete, a generated sequence ruleset node will appear in the Models tab of the Manager.

Right-click on the **generated Sequence ruleset** node in the Models tab of the Manager, and then click **Browse** on the Context menu

Figure 5.7 Sequence Rule Sets



Sort by: Confidence %   86 of 86

Antecedent	Consequent	Support %	Confidence %
125 195	210	23.467	98.295
90 125 195	210	22.133	98.193
180 195	210	21.467	96.894
90 180 195	210	21.067	96.835
90 110 120	210	20.667	96.774
90 120	210	36.533	96.715
90 105	210	34.533	96.525
120	210	38.0	96.491
125 180	210	22.4	96.429

By default, in addition to the consequent and antecedents, only the *Support %* and *Confidence %* are displayed.

The Sequence node found 86 rules, which are presented in descending order by rule confidence. The second rule “90 > 125 > 195 => 210” is a sequence that begins with code 90 (all actual repair/diagnostic sequences should start with 90), followed sometime later by code 125, then later by code 195, and then later by code 210 (successful resolution). The antecedent sequence occurs in 22.1% (*Support %*) of all IDs (there are 750 IDs in the data). Of the cases containing the sequence “90 > 125 > 195”, code 210 occurred sometime later in 98.2% of these instances (*Confidence %*).

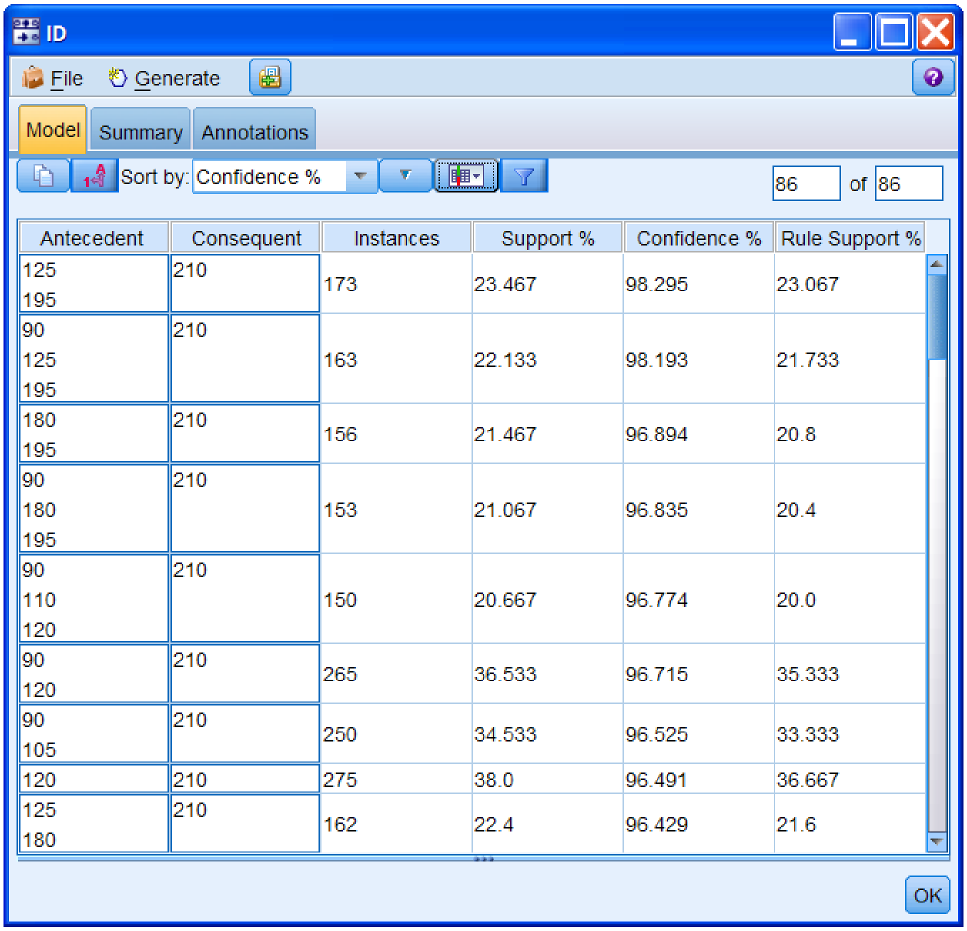
Notice that codes 90 and 210 appear frequently in the rules. This is because almost all service problem sequences begin with 90 and end with 210. Someone with domain knowledge of this area could now examine the sequences to determine if there is anything interesting or unexpected, such as a sequence that should not occur given the nature of the diagnostic tests/repairs, or a repeating sequence.

To see the other available measures:

Click the **Show/hide criteria menu** button
Click **Show all**

We now see that the second rule—antecedents and consequent—applies to 163 IDs, which is 21.7% (163/750) of the total file. Thus almost one fourth of all service problems in the data showed this pattern.

Figure 5.8 Adding Instances and Rule Support to Displayed Rule Statistics

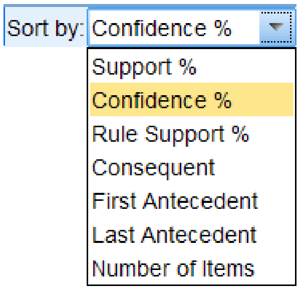


Antecedent	Consequent	Instances	Support %	Confidence %	Rule Support %
125 195	210	173	23.467	98.295	23.067
90 125 195	210	163	22.133	98.193	21.733
180 195	210	156	21.467	96.894	20.8
90 180 195	210	153	21.067	96.835	20.4
90 110 120	210	150	20.667	96.774	20.0
90 120	210	265	36.533	96.715	35.333
90 105	210	250	34.533	96.525	33.333
120	210	275	38.0	96.491	36.667
125 180	210	162	22.4	96.429	21.6

The sequence rule sets are ordered by confidence value (descending order). To view the most common sequences, we simply sort by Rule support percentage.

Click the **Sort by** drop-down list

Figure 5.9 Sort Options for Sequence Rule Sets

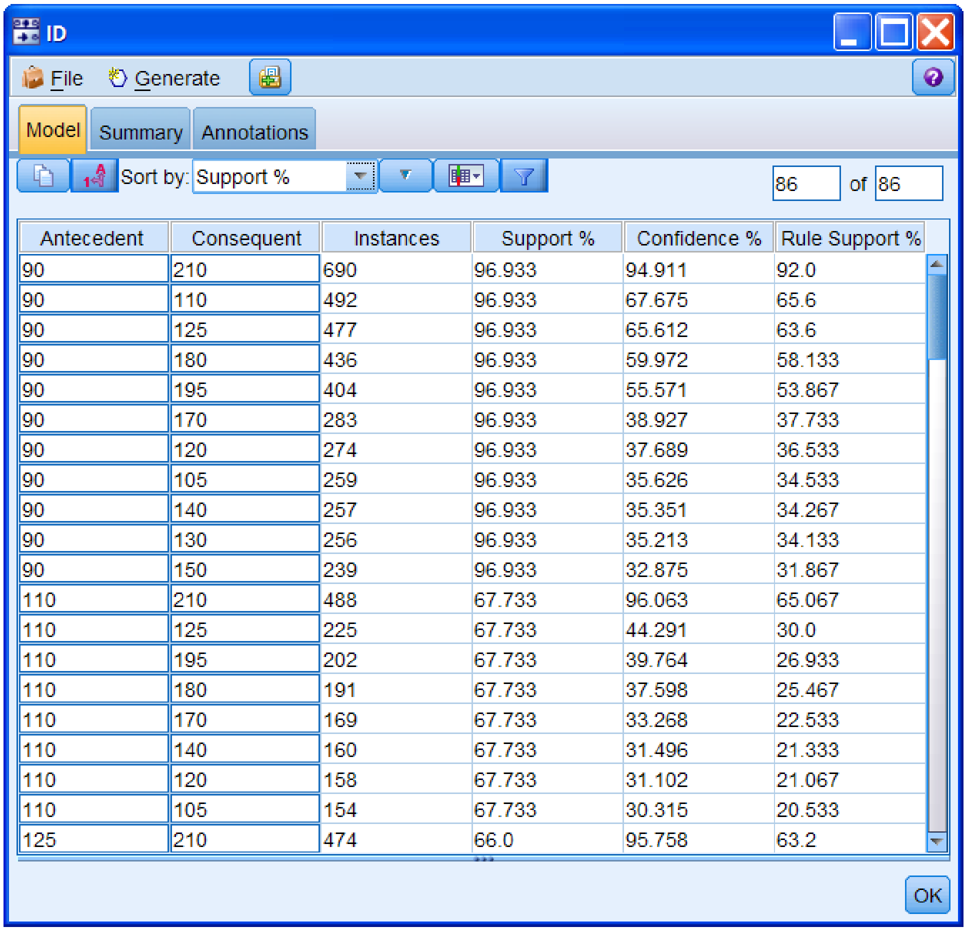


Sort by: Confidence %

The sequence rules can be sorted in a number of ways. For those interested in sequences beginning or ending with a particular event (for example, clicking on a specific web-page), the sorts by first antecedent or consequent would be of interest. Notice that the sorts can be done in ascending or descending order.

Click **Support %** on the **Sort by** drop-down list

Figure 5.10 Sequence Rule Sets Sorted by Support



Sort by: Support %

86 of 86

Antecedent	Consequent	Instances	Support %	Confidence %	Rule Support %
90	210	690	96.933	94.911	92.0
90	110	492	96.933	67.675	65.6
90	125	477	96.933	65.612	63.6
90	180	436	96.933	59.972	58.133
90	195	404	96.933	55.571	53.867
90	170	283	96.933	38.927	37.733
90	120	274	96.933	37.689	36.533
90	105	259	96.933	35.626	34.533
90	140	257	96.933	35.351	34.267
90	130	256	96.933	35.213	34.133
90	150	239	96.933	32.875	31.867
110	210	488	67.733	96.063	65.067
110	125	225	67.733	44.291	30.0
110	195	202	67.733	39.764	26.933
110	180	191	67.733	37.598	25.467
110	170	169	67.733	33.268	22.533
110	140	160	67.733	31.496	21.333
110	120	158	67.733	31.102	21.067
110	105	154	67.733	30.315	20.533
125	210	474	66.0	95.758	63.2

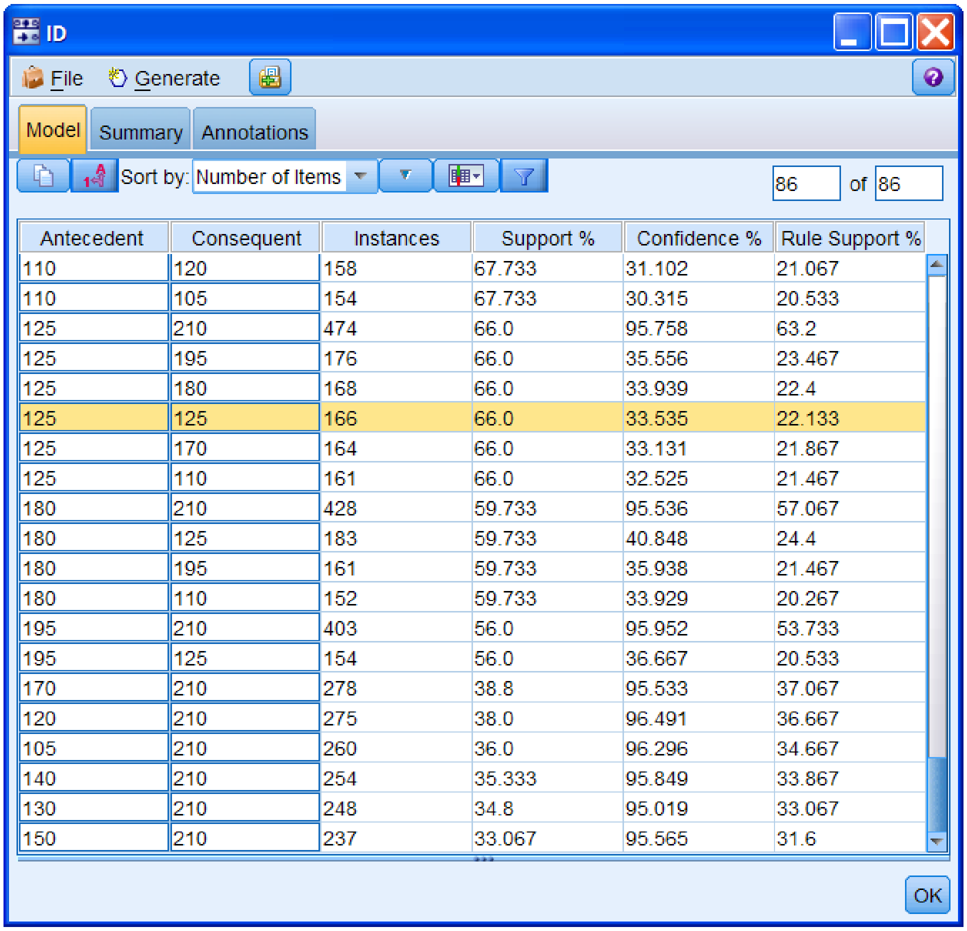
Code 110 appears in two of the three most frequent rules. The sequence 90 followed by 210 occurs in about 92% of the service problems, which we would expect in a very high proportion of the sequences. Code 299, which indicates the problem was not resolved, has not appeared. This is because it is relatively infrequent (fortunately so, for the business and customers). If we were interested in sequences containing 299, we would have to lower the support to below 5%, which is the base rate for code 299. The exercises for this lesson perform some exploration of sequences in which the problem was not resolved.

A domain expert would be interested in the most frequent sequences, which describe the typical path a service problem follows. If some stages were more expensive or time consuming, they would attract particular attention. We will view the results in one other order.

Click **Number of Items** on the **Sort by** drop-down list
Scroll down to the **bottom** of the list (showing two-item sequences)

The sequences are now sorted by the number of items. About ten lines from the bottom we find the sequence 125 => 125, which occurs in 22% of the IDs. This would be of interest, because, ideally, a diagnostic/repair stage should not be repeated. Someone familiar with the diagnostic/repair process would look into why this stage is repeating (erroneous test results at that stage, records not being forwarded properly, etc.) and modify the process to reduce it. Other repeating sequences may be present, but do not meet the minimum support and confidence criteria.

Figure 5.11 Sequence Rule Sets Sorted by Number of Items in Sequence



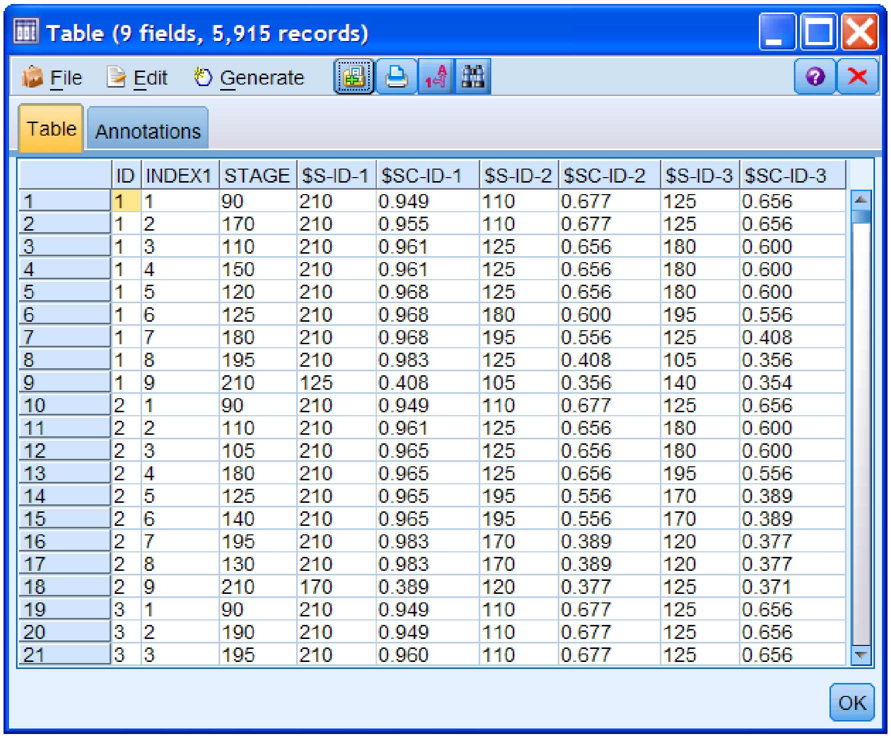
Antecedent	Consequent	Instances	Support %	Confidence %	Rule Support %
110	120	158	67.733	31.102	21.067
110	105	154	67.733	30.315	20.533
125	210	474	66.0	95.758	63.2
125	195	176	66.0	35.556	23.467
125	180	168	66.0	33.939	22.4
125	125	166	66.0	33.535	22.133
125	170	164	66.0	33.131	21.867
125	110	161	66.0	32.525	21.467
180	210	428	59.733	95.536	57.067
180	125	183	59.733	40.848	24.4
180	195	161	59.733	35.938	21.467
180	110	152	59.733	33.929	20.267
195	210	403	56.0	95.952	53.733
195	125	154	56.0	36.667	20.533
170	210	278	38.8	95.533	37.067
120	210	275	38.0	96.491	36.667
105	210	260	36.0	96.296	34.667
140	210	254	35.333	95.849	33.867
130	210	248	34.8	95.019	33.067
150	210	237	33.067	95.565	31.6

5.5 *Model Predictions*

Next we view the model predictions.

Click **OK** to close the **Sequence Ruleset** browser window
Place a **Table** node from the Output palette to the right of the **Sequence generated model** node on the Stream Canvas
Connect the **Sequence generated model** node to the **Table** node
Run the **Table** node attached to the Sequence generated model node

Figure 5.12 Top Three Sequence Predictions



	ID	INDEX1	STAGE	$S-ID-1	$SC-ID-1	$S-ID-2	$SC-ID-2	$S-ID-3	$SC-ID-3
1	1	1	90	210	0.949	110	0.677	125	0.656
2	1	2	170	210	0.955	110	0.677	125	0.656
3	1	3	110	210	0.961	125	0.656	180	0.600
4	1	4	150	210	0.961	125	0.656	180	0.600
5	1	5	120	210	0.968	125	0.656	180	0.600
6	1	6	125	210	0.968	180	0.600	195	0.556
7	1	7	180	210	0.968	195	0.556	125	0.408
8	1	8	195	210	0.983	125	0.408	105	0.356
9	1	9	210	125	0.408	105	0.356	140	0.354
10	2	1	90	210	0.949	110	0.677	125	0.656
11	2	2	110	210	0.961	125	0.656	180	0.600
12	2	3	105	210	0.965	125	0.656	180	0.600
13	2	4	180	210	0.965	125	0.656	195	0.556
14	2	5	125	210	0.965	195	0.556	170	0.389
15	2	6	140	210	0.965	195	0.556	170	0.389
16	2	7	195	210	0.983	170	0.389	120	0.377
17	2	8	130	210	0.983	170	0.389	120	0.377
18	2	9	210	170	0.389	120	0.377	125	0.371
19	3	1	90	210	0.949	110	0.677	125	0.656
20	3	2	190	210	0.949	110	0.677	125	0.656
21	3	3	195	210	0.960	110	0.677	125	0.656

By default, the Sequence generated model node contains three prediction fields (prefixed with "$S-"), containing the three most confident predictions of codes that will appear later in the sequence, predicted from the sequence observed to that point. The confidence value for each prediction is stored in a field prefixed with "$SC-".

The sequence value in the first record is stage 90 (for ID=1, Index1=1), which is the problem report. The most likely stage to occur later, given that stage 90 has occurred, is stage 210 with confidence .949. (Note: this rule can be seen in Figure 5.10.) Since most sequences end with stage 210, the second and third most confident predictions are, in some sense, more interesting for this analysis. Thus, the next most likely stage to occur later, given that stage 90 has occurred, is stage 110 with confidence .677. And the third most likely is stage 125. In this way, the three most confident future predictions, based on the observed sequence, are generated.

Examining the predictions for ID 1, notice that the most likely item to occur later can change as the observed sequence changes. This makes sense, since as more information becomes available about a sequence, additional rules can apply.

Extensions

Thus far we have explored the sequence rules and sequence predictions. The Generate menu in the rule browser window can be used to create supernodes (star-shaped nodes that encapsulate other nodes), which when added to the stream, can detect sequences, count sequences, and generate predictions. By default, this and the predictions from the Sequence generated rule node are made for the three most confident rules in the rule set, but this number can be increased in the Model tab of the Sequence node.

Summary

In this lesson you have been introduced to sequence detection within PASW Modeler.

You should now be able to:

- Create a set of sequence rules using the Sequence node
- View the resulting sequences by browsing the generated model node
- Explain the meaning of confidence and support of the sequences
- Produce predictions using the Sequence generated model node

Summary Exercises

This set of exercises is written around the following data file: *FailTelRepair.txt*.

In this session we will explore sequences associated with a specific event: failure to resolve a telecom service problem. We will use a subset of records from the dataset used in Lesson 5: sequences with termination code 299 (meaning that the problem was not resolved).

1. Read the tab-delimited text file named *FailTelRepair.txt* into PASW Modeler (you can use a Var. File node or modify the *Telrepairdef.str* stream file).

2. Click the Types tab, then click the Read Values button, and then click OK

3. Add a Table node to the stream and run it.

4. Add a Sequence node to the stream. Specify *ID* as the ID field, *Index1* as the time field, and *Stage* as the content field. Run the sequence detection analysis in Simple mode.

5. Browse the generated rule set. What are the most confident and the most common sequences? Try different sorts of the generated rules to see if they provide additional insight.

6. Compare these results to those reported in Lesson 5. Do you find any stage codes related that frequently appear in sequences involving code 299 (failure to resolve problem) that do not frequently appear in sequences involving code 210 (problem solved)? These might provide some insight into which stages lead to a repair failure.

Lesson 6: Advanced Sequence Detection

Objectives

- Discuss sequence analysis in depth in PASW Modeler
- Review the expert options for the Sequence node
- Provide an example of searching for sequences in web log data

Data

The data consist of a subset of a simulated web log used with PASW Modeler Web Mining. The file, named weblog.txt, contains 3000 records, each representing a line from a web log. The data have undergone preprocessing and cleaning, and include only three fields (*VisitID*, *TimeStamp*, and *Page*). *VisitID* is an identifier assigned to each distinct session (created by earlier processing of web logs). *TimeStamp* is an integer value created by converting the original date and time fields into seconds. Thus, while the actual *TimeStamp* values are not of interest, they are used to order the data and create the time interval (in seconds) between two ordered (by *TimeStamp* within *VisitID*) records from the same visit. The *Page* field is a description of the web page viewed. The file has been presorted by VisitID (major sort) and *TimeStamp* (minor sort).

6.1 *Introduction*

Sequence detection involves searching for patterns in data that are time ordered. In Lesson 5 we introduced sequence detection. In this lesson we explore the Sequence node so that you are aware of the various options when doing a sequence analysis.

Within PASW Modeler, Sequence, Apriori, and Carma present results in a similar format. As a reminder, it is important to note that for Sequence the antecedents and consequent are time-ordered, while they need not be for the association models.

6.2 *Sequence Node*

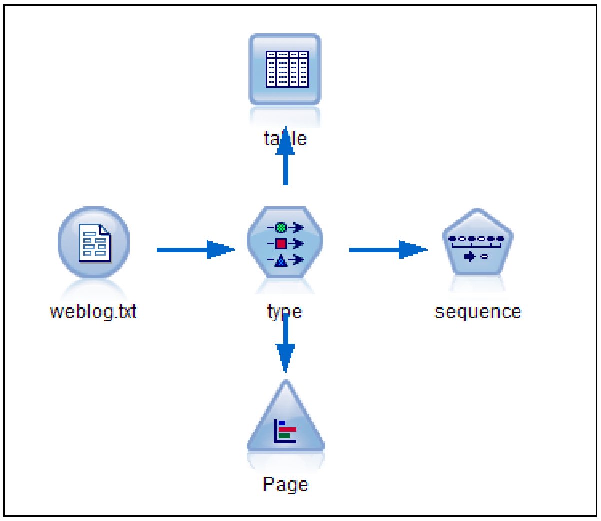
The Sequence node permits the members of an item set to be spread across multiple records (transaction format) or to be spread across multiple fields (tabular format) in a single record. There is an option to permit separation or construction of item sets based on the time elapsed between two records. You can set minimum support value, confidence value, and the sequence lengths of interest (maximum sequence length). Sequence uses the CARMA algorithm, which produces a solution in two data passes. In addition, the Sequence node permits you to add model predictions to the stream.

6.3 *Sequence Node Expert Options*

We will review the expert options available within the Sequence node by working with a prepared stream. The options will be discussed and the exercise section provides an opportunity to try the variations and observe the results.

Click **File…Open Stream** and move to the c:\Train\ModelerClusAssoc directory
Double-click on **WebSequence.str**

Figure 6.1 WebSequence.str Stream with Sequence Node



The *WebSequence.str* stream reads web log data from a text file. The Distribution node displays the frequency of viewing different web pages. The Sequence node performs a sequence detection analysis.

Run the **Table** node

Figure 6.2 Records from Web Log File

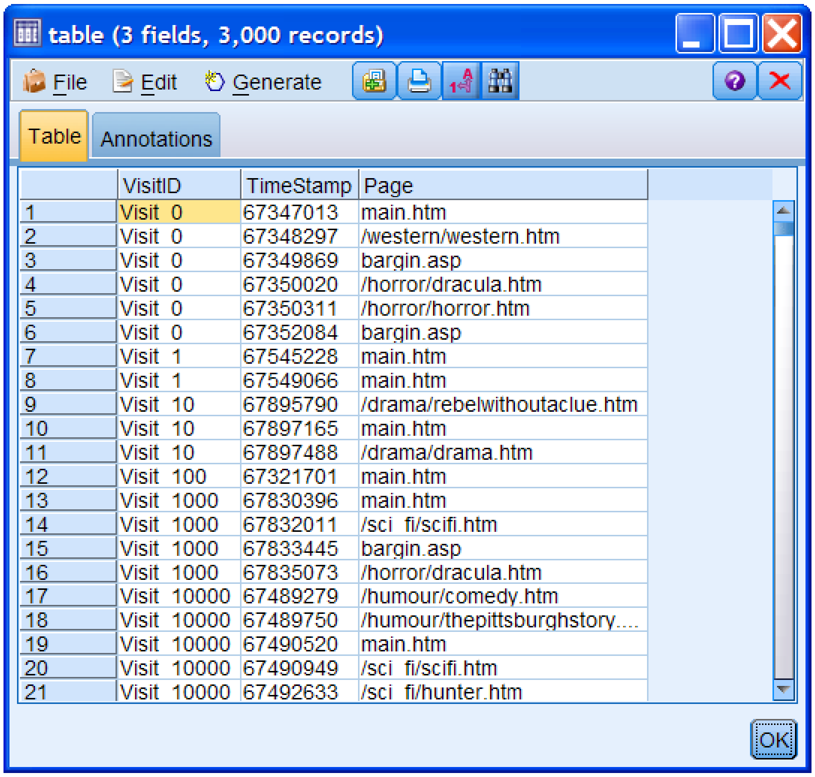


table (3 fields, 3,000 records)

	VisitID	TimeStamp	Page
1	Visit 0	67347013	main.htm
2	Visit 0	67348297	/western/western.htm
3	Visit 0	67349869	bargin.asp
4	Visit 0	67350020	/horror/dracula.htm
5	Visit 0	67350311	/horror/horror.htm
6	Visit 0	67352084	bargin.asp
7	Visit 1	67545228	main.htm
8	Visit 1	67549066	main.htm
9	Visit 10	67895790	/drama/rebelwithoutaclue.htm
10	Visit 10	67897165	main.htm
11	Visit 10	67897488	/drama/drama.htm
12	Visit 100	67321701	main.htm
13	Visit 1000	67830396	main.htm
14	Visit 1000	67832011	/sci_fi/scifi.htm
15	Visit 1000	67833445	bargin.asp
16	Visit 1000	67835073	/horror/dracula.htm
17	Visit 10000	67489279	/humour/comedy.htm
18	Visit 10000	67489750	/humour/thepittsburghstory....
19	Visit 10000	67490520	main.htm
20	Visit 10000	67490949	/sci_fi/scifi.htm
21	Visit 10000	67492633	/sci_fi/hunter.htm

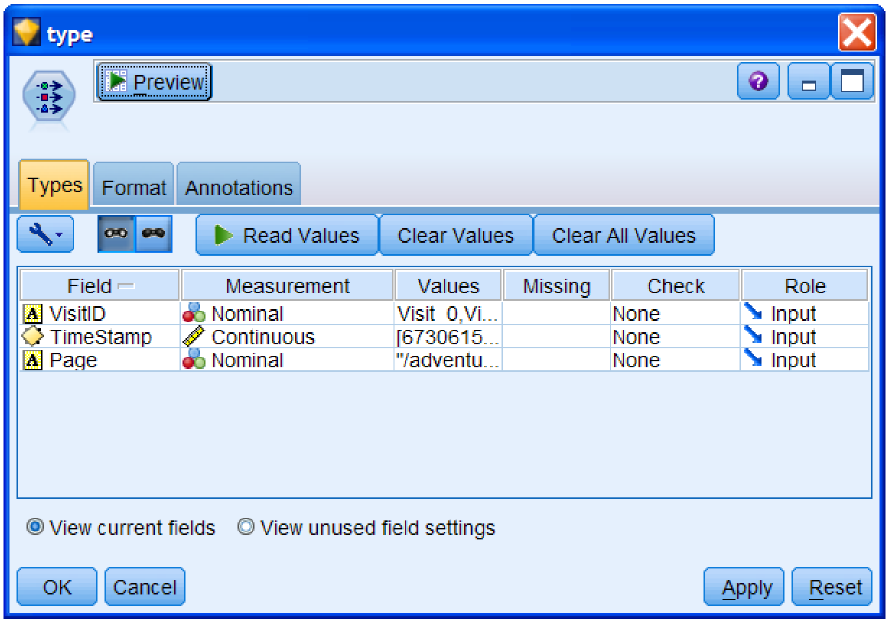
There are 3000 records in this data extract (although there are only 912 distinct values of *VisitID*, so each visitor viewed about three pages, on average). The *VisitID* field is coded as a string (Nominal measurement). When analyzing large files, there is an advantage to coding ID fields, such as *VisitID*, as numeric fields (Continuous measurement) since each value need not be stored as part of the field definition and reduces the stream file size.

The Time field for the Sequence node must be numeric. The *TimeStamp* field was created from an original time stamp field by calculating the number of seconds from January 1, 1999 (the date baseline value - set within the Stream Properties Options dialog) to the second at which the page was clicked. Thus while the values are large and not directly interpretable, they provide a way to order the data by time and calculate time elapsed between page clicks.

The *Page* field records the page that was viewed. There are many pages and most receive a tiny fraction of the clicks (to see this, run the Distribution node).

Close the **Table** window
Double-click the **Type** node

Figure 6.3 Type Node for Sequence Detection



Field	Measurement	Values	Missing	Check	Role
VisitID	Nominal	Visit_0,Vi...		None	Input
TimeStamp	Continuous	[6730615...		None	Input
Page	Nominal	"/adventu...		None	Input

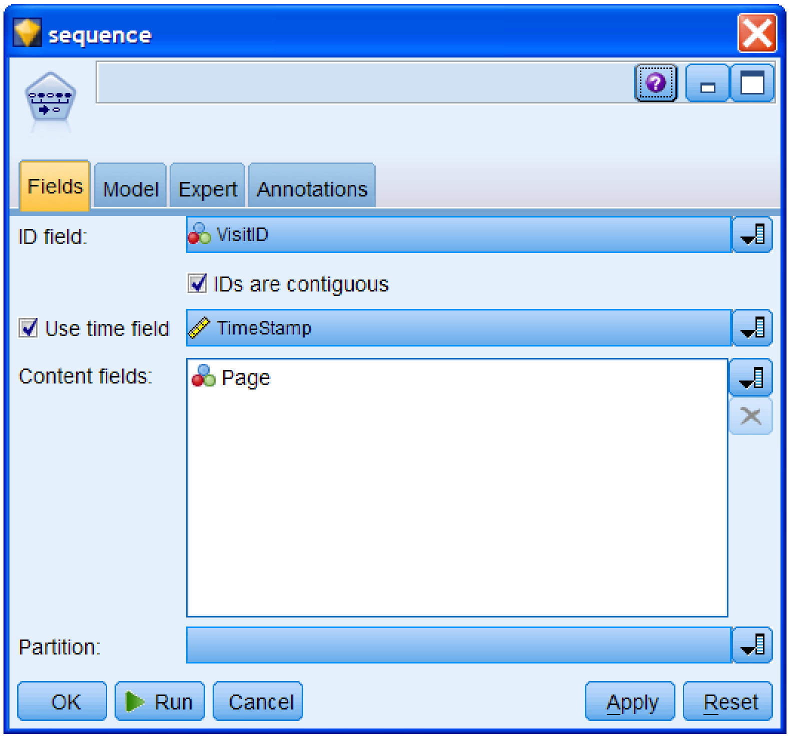
● View current fields ○ View unused field settings

The *Page* field, which is the basis of the sequence analysis, has its role set to Input. The field(s) containing the content can be of role *Input*, *Target*, or *Both*. If there are multiple content fields, they all must be the same measurement type. The field (here *VisitID*) that identifies the unit of analysis can be either continuous or categorical measurement types and can have any role—here it is set to *Input*.

The field *TimeStamp* is Continuous (numeric) measurement, and in general, the time field must be numeric, date, time, or timestamp.

Close the **Type** node dialog
Double-click on the **Sequence** node

Figure 6.4 Sequence Node Dialog (Fields Tab)



ID field: VisitID

☑ IDs are contiguous

☑ Use time field: TimeStamp

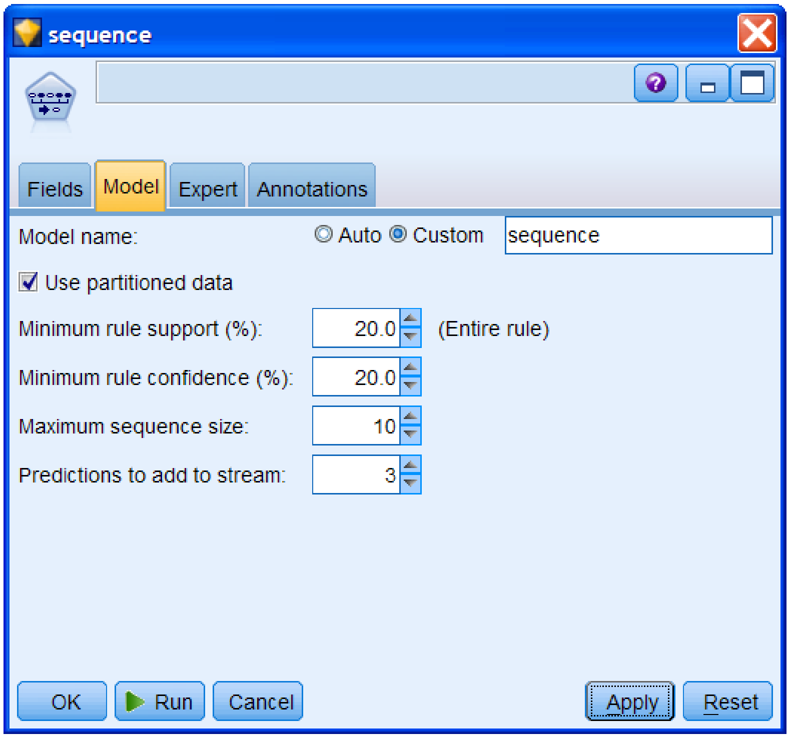
Content fields: Page

Partition:

The *ID field* identifies the unit of analysis. In this case it would be (as near as could be determined) a single visit to the website. The Time field is optional, but can be used to order the data and, when used, supports an expert option involving the maximum time duration of sequences. You can specify a single content field, as we did here (*Page*), which is appropriate for data in transactional format. Multiple content fields are specified when the data are in tabular format; that is, multiple values are attached to a single time point and form an item set within a record. The data file was presorted by *VisitID* for efficiency, and to prevent Sequence from resorting the data, the *Ids are contiguous* box is checked.

Click the **Model** tab

Figure 6.5 Sequence Node: Model Tab



Model name: ○ Auto ● Custom sequence

☑ Use partitioned data

Minimum rule support (%): 20.0 (Entire rule)

Minimum rule confidence (%): 20.0

Maximum sequence size: 10

Predictions to add to stream: 3

The default *Minimum rule support* (percentage of IDs that contain both the antecedents and consequent) and *Minimum rule confidence* values are 20%. These values, especially support, are usually modified giving consideration to the base rates of the items. The Distribution node can provide information about this. For example, since there are many web pages reported in the web log, when minimum support is set to 20%, the only rule found is “main.htm=>main.htm” which is not all that interesting.

You might first run the analysis with the default *Minimum rule support* and *Minimum rule confidence* values, and then adjust the settings based on these results.

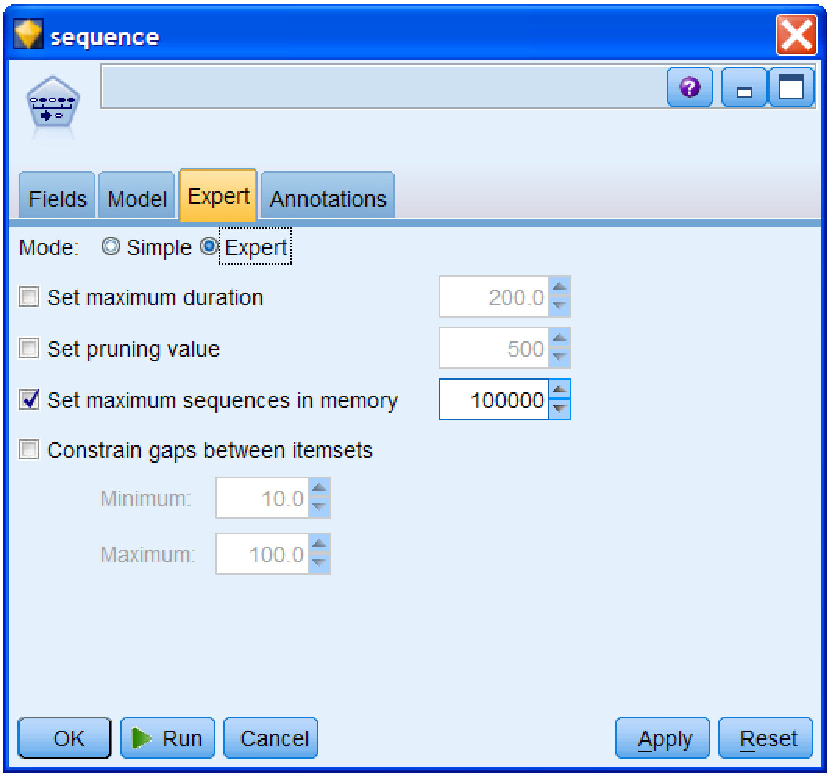
The *Maximum sequence size* option controls the maximum number of distinct item sets in a sequence. This setting would be changed if you wish to explore very long sequences or only short sequences (reducing the *Maximum sequence size* would speed up processing).

Executing the Sequence node will produce a generated Sequence ruleset node containing the sequences discovered. This node can be browsed to examine the detected sequences and added to a data stream in order to generate predictions. The *Predictions to add to stream* option controls how many of the top prediction rules will be used to generate prediction fields. By default, predictions will be created for the top three sequence rules and this number can be increased.

Click the **Expert tab**, and then click the **Expert** option button

The *Set maximum duration* option concerns the time span or duration within which sequences will be examined. When the option is checked, the time duration between the first and last item sets in a sequence must be less than or equal to the maximum duration value. For example, if a Time field is used when analyzing banking transaction patterns, you may only be interested in sequences that complete within a one-day time period. If no Time field is specified, then the maximum duration is expressed in terms of the number of records.

Figure 6.6 Sequence Node: Expert Tab



Mode: ○ Simple ● Expert

☐ Set maximum duration 200.0

☐ Set pruning value 500

☑ Set maximum sequences in memory 100000

☐ Constrain gaps between itemsets

Minimum: 10.0

Maximum: 100.0

The *Set pruning value* option controls the frequency at which the CARMA algorithm (on which Sequence is based) prunes infrequently occurring sequences as it reads the data. If this option is enabled, increasing the pruning value will increase speed and possibly increase memory usage, while decreasing the pruning value will decrease memory usage and potentially decrease speed.

The control to restrict memory usage during model building allows you to limit the total number of sequences stored in memory during model building. The *maximum sequences in memory* value does not apply to sequence length, but rather to the number of candidate sequences that are stored and tracked, and this number is typically much larger than the reported sequences. Decreasing this value will result in less memory used when the Sequence node is executed.

The *Constrain gaps between item sets* option determines whether item sets separated either by very short, or very long, time gaps will be considered part of a sequence. Thus item sets with time gaps (difference in Time field) less than the *Minimum* gap value will not be considered as part of a sequence. This might be desirable if a very small time gap indicates an error, a double entry, or the refresh of a web page. Similarly, if you specify a *Maximum* gap value, then item sets separated by a large time interval (greater than the Maximum gap) will not be considered as part of a sequence.

Whether it makes sense to specify gap values depends very much on the domain area in which you are working. For example, if you are exploring common sequences in automobile repairs while cars are under warranty, is there a time gap beyond which you would consider a new repair to be part of a new sequence, rather than part of an earlier sequence?

6.4 *Sequence Results*

Next we run a sequence detection analysis and examine the generated Sequence Ruleset node.

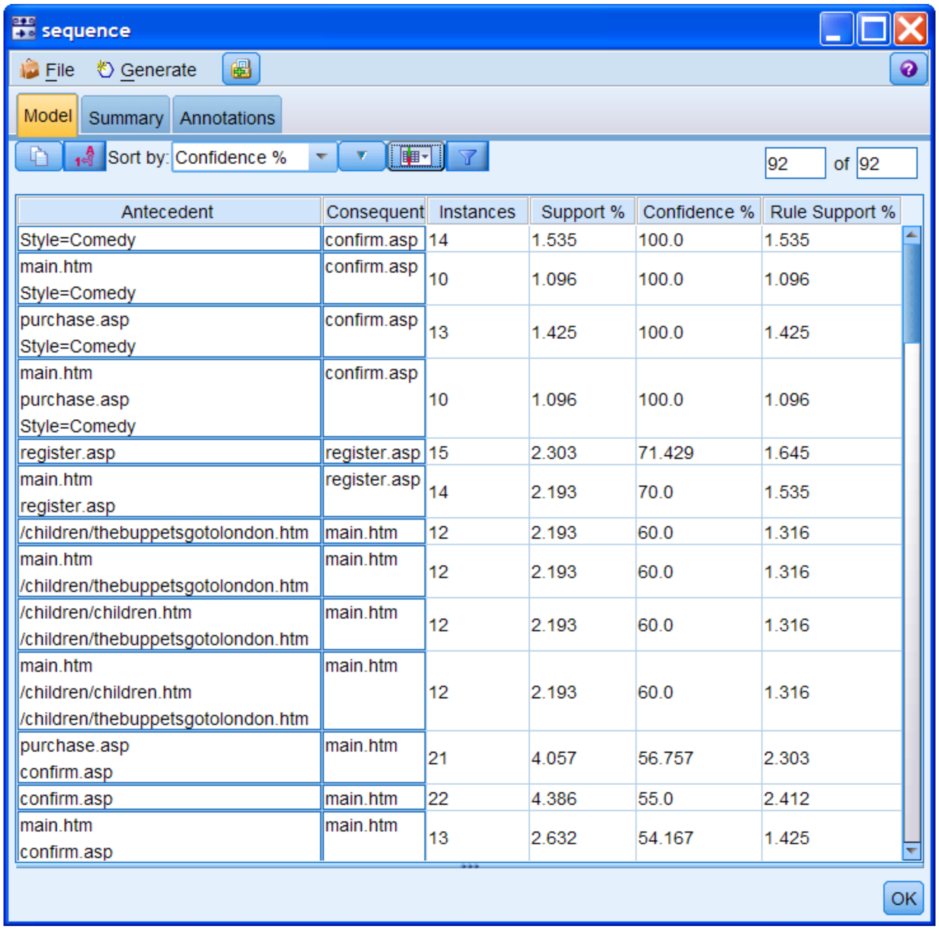
Click the **Simple Mode** option button
Click the **Model** tab
Set the **Minimum rule support** value to **1** (not shown)

We reduce the *Minimum rule support* value because most web pages appear in the web log with very low frequencies and the default *Minimum rule support* and *Minimum rule confidence* values produce but a single rule (this is shown in the exercises).

Click **Run**
Browse the **generated Sequence Ruleset** node (named **sequence**) in the Models Manager window
Click **Show/Hide Criteria** button and request **Show all** from the menu
If necessary, **increase** the **column widths** to better read the item values

Ninety-two sequence rules were found and they are sorted by confidence value (the *Sort by* drop-down list can produce other sort orders). The sequence "Style=Comedy => confirm.asp" appeared for 14 (*Instances)*, or 1.5%, of the IDs (*Rule Support %*, or 14/912 * 100) and all occurrences of "Style=Comedy" were followed by "confirm.asp," as the rule confidence is 100.000. Note that the support values shown on this screen all tend to be less than 3%, which is the reason we lowered the *Minimum rule support* value.

Figure 6.7 Generated Sequence Ruleset Node Browse Window



Sort by: Confidence %

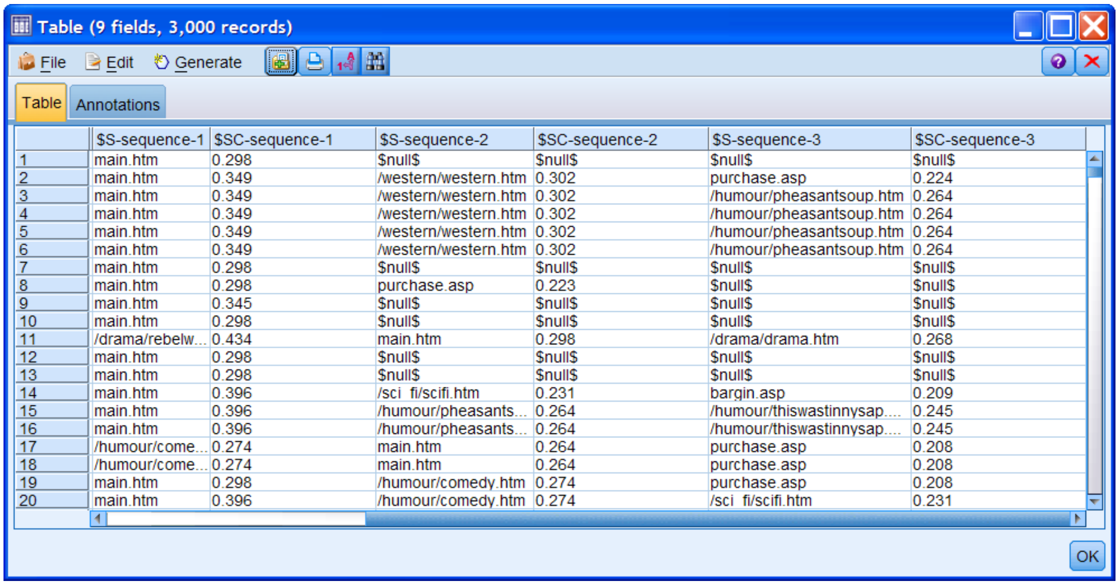
92 of 92

Antecedent	Consequent	Instances	Support %	Confidence %	Rule Support %
Style=Comedy	confirm.asp	14	1.535	100.0	1.535
main.htm Style=Comedy	confirm.asp	10	1.096	100.0	1.096
purchase.asp Style=Comedy	confirm.asp	13	1.425	100.0	1.425
main.htm purchase.asp Style=Comedy	confirm.asp	10	1.096	100.0	1.096
register.asp	register.asp	15	2.303	71.429	1.645
main.htm register.asp	register.asp	14	2.193	70.0	1.535
/children/thebuppetsgotolondon.htm	main.htm	12	2.193	60.0	1.316
main.htm /children/thebuppetsgotolondon.htm	main.htm	12	2.193	60.0	1.316
/children/children.htm /children/thebuppetsgotolondon.htm	main.htm	12	2.193	60.0	1.316
main.htm /children/children.htm /children/thebuppetsgotolondon.htm	main.htm	12	2.193	60.0	1.316
purchase.asp confirm.asp	main.htm	21	4.057	56.757	2.303
confirm.asp	main.htm	22	4.386	55.0	2.412
main.htm confirm.asp	main.htm	13	2.632	54.167	1.425

When a rule is 100% accurate, the support percentage is equal to the rule support percentage. Otherwise, they will differ, and support will be higher than rule support, as is true for the fifth rule. The relationship between these three measures is Support*(Confidence/100) = Rule support, so for the fifth rule, 2.303*(71.42/100) = 1.645.

Close the **Sequence Ruleset** browser window
Connect the **generated Sequence Ruleset** node on the Stream Canvas to the **Table** node
Run the **Table** node
Scroll to the **right** of the Table window

Figure 6.8 Fields Created by the Generated Sequence Ruleset Node



	$S-sequence-1	$SC-sequence-1	$S-sequence-2	$SC-sequence-2	$S-sequence-3	$SC-sequence-3
1	main.htm	0.298	$null$	$null$	$null$	$null$
2	main.htm	0.349	/western/western.htm	0.302	purchase.asp	0.224
3	main.htm	0.349	/western/western.htm	0.302	/humour/pheasantsoup.htm	0.264
4	main.htm	0.349	/western/western.htm	0.302	/humour/pheasantsoup.htm	0.264
5	main.htm	0.349	/western/western.htm	0.302	/humour/pheasantsoup.htm	0.264
6	main.htm	0.349	/western/western.htm	0.302	/humour/pheasantsoup.htm	0.264
7	main.htm	0.298	$null$	$null$	$null$	$null$
8	main.htm	0.298	purchase.asp	0.223	$null$	$null$
9	main.htm	0.345	$null$	$null$	$null$	$null$
10	main.htm	0.298	$null$	$null$	$null$	$null$
11	/drama/rebelw...	0.434	main.htm	0.298	/drama/drama.htm	0.268
12	main.htm	0.298	$null$	$null$	$null$	$null$
13	main.htm	0.298	$null$	$null$	$null$	$null$
14	main.htm	0.396	/sci fi/scifi.htm	0.231	bargin.asp	0.209
15	main.htm	0.396	/humour/pheasants...	0.264	/humour/thiswastinnysap....	0.245
16	main.htm	0.396	/humour/pheasants...	0.264	/humour/thiswastinnysap....	0.245
17	/humour/come...	0.274	main.htm	0.264	purchase.asp	0.208
18	/humour/come...	0.274	main.htm	0.264	purchase.asp	0.208
19	main.htm	0.298	/humour/comedy.htm	0.274	purchase.asp	0.208
20	main.htm	0.396	/humour/comedy.htm	0.274	/sci fi/scifi.htm	0.231

The first six records belong to a single web site visit. The fields with names beginning “$S-” contain predictions based on the generated ruleset. The integer portions of these field names (1, 2 and 3 here) reflect the confidence order of the predictions. Thus “$S-sequence-1” field is the page prediction with the highest confidence based on the portion of the sequence examined to this point and “$S-sequence-2” is the page prediction with the second highest confidence. The corresponding confidence values are stored in the fields beginning with “$SC-”.

The actual page value for the first record is “main.htm.” The consequent with the greatest confidence value predicted from this antecedent is also “main.htm” and the confidence value for this rule is .298 (this rule can be found by browsing the generated Sequence Ruleset node). The current ruleset has no other rules for the sequence (“main.htm” to this point), so the second and third most confident predictions are $null$.

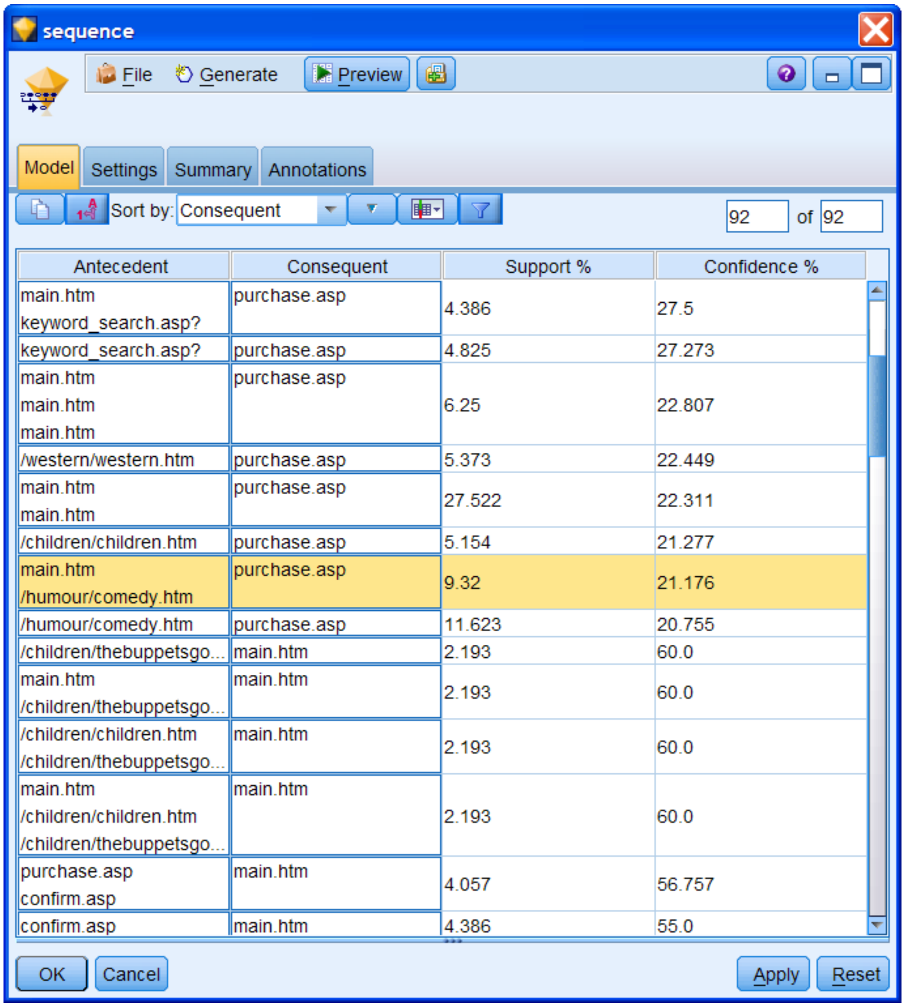
The second record contains the page “/western/western.htm.” The most confident prediction remains the same, but notice that the confidence value changes (to .349). This is because a different rule with a higher confidence value can now be applied (“main.htm > /western/western.htm => main.htm” with confidence .349). In addition, we find a second (“/western/western.htm”) and third (“purchase.asp”) predictions, both with lower confidence (.302 and .224, respectively), because multiple rules now apply to this sequence. It may seem odd that the prediction can change across records within the same ID, but not if you consider that each record adds information about the sequence and this can lead to a different rule applying, which in turn can lead to a different prediction and/or confidence.

Extensions

The Generate menu within the generated Sequence Ruleset browser window can create supernodes that can be added to this or another data stream. Supernodes can be generated that act as: (1) Sequence detectors that indicate whether a target sequence is found within an ID, (2) Sequence counters that count the number of times the target sequence appears for each ID, and (3) Sequence predictors that add predictions to the data stream based on the target sequences. We will illustrate this feature by adding a sequence detector to the stream (the *PASW Modeler User’s Guide* and help system describes the behavior of these supernodes in greater detail.)

Close the Table window
Edit the **generated Sequence ruleset node** (named **sequence**) in the Stream canvas
Select **Consequent** in the **Sort by** drop-down list
Select the rule "**main.htm > /humour/comedy.htm => purchase.asp**" (it is the 13th rule in the list)

Figure 6.9 Rule Selected in Sequence Ruleset



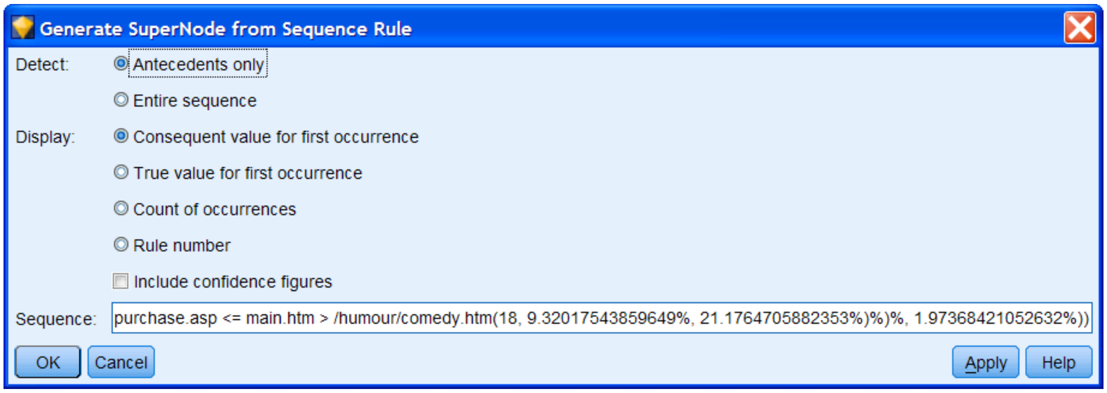
Sort by: Consequent

92 of 92

Antecedent	Consequent	Support %	Confidence %
main.htm keyword_search.asp?	purchase.asp	4.386	27.5
keyword_search.asp?	purchase.asp	4.825	27.273
main.htm main.htm main.htm	purchase.asp	6.25	22.807
/western/western.htm	purchase.asp	5.373	22.449
main.htm main.htm	purchase.asp	27.522	22.311
/children/children.htm	purchase.asp	5.154	21.277
main.htm /humour/comedy.htm	purchase.asp	9.32	21.176
/humour/comedy.htm	purchase.asp	11.623	20.755
/children/thebuppetsgo...	main.htm	2.193	60.0
main.htm /children/thebuppetsgo...	main.htm	2.193	60.0
/children/children.htm /children/thebuppetsgo...	main.htm	2.193	60.0
main.htm /children/children.htm /children/thebuppetsgo...	main.htm	2.193	60.0
purchase.asp confirm.asp	main.htm	4.057	56.757
confirm.asp	main.htm	4.386	55.0

Click **Generate…Rule Supernode**

Figure 6.10 Generate Supernode from Sequence Rule Dialog



Generate SuperNode from Sequence Rule

Detect: ● Antecedents only
○ Entire sequence

Display: ● Consequent value for first occurrence
○ True value for first occurrence
○ Count of occurrences
○ Rule number

□ Include confidence figures

Sequence: purchase.asp <= main.htm > /humour/comedy.htm(18, 9.32017543859649%, 21.1764705882353%)%)%, 1.97368421052632%))

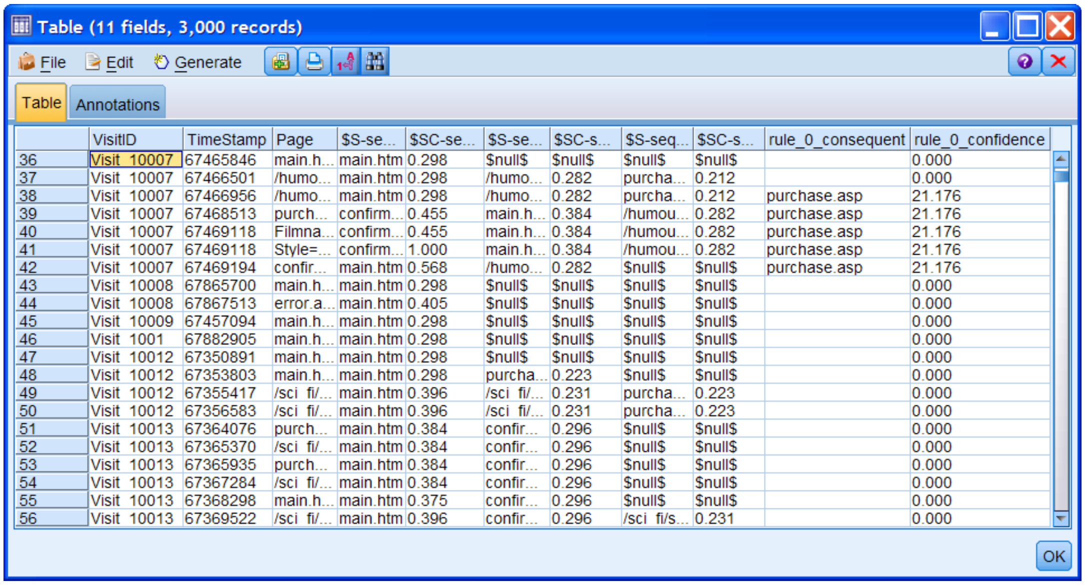
The selected rule appears in the Sequence box as a reference.

The *Detect* option gives you a choice as to whether you wish to search only for antecedents to the selected rule or both the antecedents and consequent (*Entire Sequence*). The *Display* group controls the values that will be added to the field created by this dialog. These values can represent: the rule consequent which will appear the first time the antecedents are found for an ID; the true value (T) for the first occurrence of the sequence for an ID and false (F) otherwise; a count of the number of times the sequence occurs within a given ID; the rule number (based on the number of supernodes previously generated by this generated Sequence Ruleset node). If you wish several of the *Display* choices, you can later return to the Generate Supernode from the Sequence Rule dialog and produce another supernode at each visit. Also, note that the rule's confidence value can be added as an additional field.

The predictions and confidence values produced by the generated Sequence Rule node are valuable in that they identify the most confident predictions based on the entire rule set. In contrast, we focus here on a single rule and can add different summary measures to the stream based on it. For this reason if you want to predict the item most likely to occur later in the sequence you would use the generated Sequence Ruleset node. If you want to locate a specific sequence in a new dataset, then the supernode generated from the Sequence Rule dialog is needed.

Click **Include confidence figures** check box
Click **OK**
Close the Generated sequence ruleset node
Drag the **generated supernode** (named **rule_0**) to the right of the generated Sequence Rule node (**sequence**) and connect them
Connect **rule_0 supernode** to a **Table** node and run the **Table**
Scroll to the **right** in the Table to view the last few columns
Scroll down until you find a value in the **rule_0_consequent** field

Figure 6.11 Fields Added by Sequence Rule Supernode



	VisitID	TimeStamp	Page	$S-se...	$SC-se...	$S-se...	$SC-s...	$S-seq...	$SC-s...	rule_0_consequent	rule_0_confidence
36	Visit 10007	67465846	main.h...	main.htm	0.298	$null$	$null$	$null$	$null$		0.000
37	Visit 10007	67466501	/humo...	main.htm	0.298	/humo...	0.282	purcha...	0.212		0.000
38	Visit 10007	67466956	/humo...	main.htm	0.298	/humo...	0.282	purcha...	0.212	purchase.asp	21.176
39	Visit 10007	67468513	purch...	confirm...	0.455	main.h...	0.384	/humou...	0.282	purchase.asp	21.176
40	Visit 10007	67469118	Filmna...	confirm...	0.455	main.h...	0.384	/humou...	0.282	purchase.asp	21.176
41	Visit 10007	67469118	Style=...	confirm...	1.000	main.h...	0.384	/humou...	0.282	purchase.asp	21.176
42	Visit 10007	67469194	confir...	main.htm	0.568	/humo...	0.282	$null$	$null$	purchase.asp	21.176
43	Visit 10008	67865700	main.h...	main.htm	0.298	$null$	$null$	$null$	$null$		0.000
44	Visit 10008	67867513	error.a...	main.htm	0.405	$null$	$null$	$null$	$null$		0.000
45	Visit 10009	67457094	main.h...	main.htm	0.298	$null$	$null$	$null$	$null$		0.000
46	Visit 1001	67882905	main.h...	main.htm	0.298	$null$	$null$	$null$	$null$		0.000
47	Visit 10012	67350891	main.h...	main.htm	0.298	$null$	$null$	$null$	$null$		0.000
48	Visit 10012	67353803	main.h...	main.htm	0.298	purcha...	0.223	$null$	$null$		0.000
49	Visit 10012	67355417	/sci fi/...	main.htm	0.396	/sci fi/...	0.231	purcha...	0.223		0.000
50	Visit 10012	67356583	/sci fi/...	main.htm	0.396	/sci fi/...	0.231	purcha...	0.223		0.000
51	Visit 10013	67364076	purch...	main.htm	0.384	confir...	0.296	$null$	$null$		0.000
52	Visit 10013	67365370	/sci fi/...	main.htm	0.384	confir...	0.296	$null$	$null$		0.000
53	Visit 10013	67365935	purch...	main.htm	0.384	confir...	0.296	$null$	$null$		0.000
54	Visit 10013	67367284	/sci fi/...	main.htm	0.384	confir...	0.296	$null$	$null$		0.000
55	Visit 10013	67368298	main.h...	main.htm	0.375	confir...	0.296	$null$	$null$		0.000
56	Visit 10013	67369522	/sci fi/...	main.htm	0.396	confir...	0.296	/sci fi/s...	0.231		0.000

In Visit_10007 (*VisitID* field) the target antecedents (main.htm > /humour/comedy.htm) appear in the first two records (look at the Page column). This triggers the rule, so the consequent appears in the rule_0_consequent field in all the remaining records for this ID value. Also, the rule confidence value (see Figure 6.9) appears in the rule_0_confidence field. In this way, specific sequences can be identified in the current or a new dataset. Although here the antecedents occurred contiguously (first and second record for the ID), this is not necessary in order to trigger the rule application.

Because PASW Modeler expressions to identify sequences can be complex, the code to accomplish this is stored in a supernode. It has the side benefit that you can examine its logic.

Missing Data with Sequence Models

The Sequence node ignores blanks. The algorithm will handle records containing blanks for input fields, but such a record will not be considered to match any rule containing one or more of the fields for which it has blank values.

Summary Exercises

This set of exercises is written around the following data file: weblog.*txt*. The following text gives details of the file.

weblog.txt contains a simulated web log that has already undergone preprocessing. Interest is in finding commonly occurring page sequences during visits to a web site. The file contains 3000 records and the following fields:

VISITID	Identifies (roughly) individual web site visits
TIMESTAMP	Numeric timestamp in seconds from a baseline date
PAGE	Web page viewed

1. Rerun the Sequence node from the stream file in the lesson, using the default values for *Minimum rule support* (20%) and *Minimum rule confidence* (20%). Compare the sequences reported to those in our earlier analysis. Explain the difference. Experiment by changing these values and examine the differences in the rule sets.

2. Rerun the Sequence node after changing the *Predictions to add to stream* value to 5 (see Model tab). Verify that additional prediction and confidence fields are created when you add the generated Sequence Ruleset node to the stream.

3. *For those with extra time:* Continue the last exercise by modifying the Minimum and Maximum gap values. Do you observe changes in the generated rule sets when you make large changes to these values? Characterize, if you can, the nature of these changes.

4. *For those with more extra time*: Browse one of the generated Sequence Rule nodes built earlier. Select a rule and generate a rule supernode. Attach the supernode to the stream and examine the fields it adds. Zoom in on the supernode and try to understand its logic and functioning.

Additional Readings

Aldenderfer, M.S. and R. K. Blashfield. 1984. *Cluster Analysis*. Thousand Oaks. CA: Sage.

Berry, Michael J. and G. Linoff. 2004. *Data Mining Techniques for Marketing, Sales, and Customer Support*. New York: Wiley.

Berry, Michael J. and G. Linoff. 2001. *Mastering Data Mining: The Art and Science of Customer Relationship Management*. New York: Wiley.

Berry, Michael J. and G. Linoff. 2002. *Mining the Web: Transforming Customer Data into Customer Value*. New York: Wiley.

Everitt, Brian S., S. Landau and M. Leese. 2001. *Cluster Analysis* (4th ed.). London: Edward Arnold.

Han, Jiawei and M. Kamber. 2005. *Data Mining: Concepts and Techniques* (2nd ed.). San Francisco: Morgan Kauffman.

Mannila, Heikki, P. Smyth, and D. J. Hand. 2001. *Principles of Data Mining*. Cambridge, MA: MIT.

Mena, Jesus. 1998. *Data Mining Your Website*. Boston, MA: Digital Press.

Mena, Jesus. 2001. *Webmining for Profit: E-Business Optimization*. Burlington, MA: Butterworth-Heinemann.

Mena, Jesus. 2003. *Investigative Data Mining for Security and Criminal Detection*. Burlington, MA: Butterworth-Heinemann.

SPSS Inc. 2009. *Modeler 13.0 Algorithms Guide*. Chicago: SPSS Inc.

Westphal, Christopher and T. Blaxton. 1998. *Data Mining Solutions*. New York: John Wiley.