Customer Prioritizing and Segmentation

People with similar attributes tend to exhibit similar behaviours in various ways. This fact is particularly useful in customer relationship management, marketing, and risk management. It can be used in prioritizing customer services and risk management. For example, customers with higher risk of churning can be identified using predictive analytics and prevention measures can be executed based on customer priority.

In this paper, the following predictive methods for customer prioritization are described;

- Customer Hotspot-Segment Profiling
- Customer Scoring (Neural Network)
- (Supervised) Customer Segmentation (Decision Tree)
- (Unsupervised) Customer Segmentation (Neural Clustering)

The first method is to identify ***profiles*** of customer segments with highest/lowest probability. It will provide quick information which customer segments are in need for priority actions.

The second method is to compute customer ***scores*** using neural network predictive modeling. Customer scores then are used to rank and prioritize customers.

The last two methods are based on segmentation. The third uses decision tree. Using supervised learning techniques, ***segmentation tree*** is developed. Segments are then ordered based on higher probability.

The fourth is based on unsupervised learning techniques. It uses neural clustering, aka, Self-Organizing Maps (SOM). It partitions and clusters customers based on ***similarity of attribute values***. Note that this method does not use target variables in computing similarities. Segmentation may not produce useful results.

Rosella Software
www.roselladb.com

Data Preparation

Historical data is prepared in a relational database system as a ***database table***. A lightweight database system, such as MySQL, PostgreSQL, MS SQL Server, MS Access, etc., is recommended. This is done by extracting and copying data from operational database systems. Historical data must be cleaned before analysis (***data cleaning***). Erroneous and synonymous data should be corrected. Outliers should be replaced with normal values.

Each dataset is required to have a ***unique record identifier field*** whose values are unique. For example, 1, 2, 5, 10, 11, …. It is recommended to use "RID" as the name (if not illegal). This field is required for updating records and plotting charts. In addition, this record identifier field should be ***indexed*** or be a ***primary key*** for performance reasons.

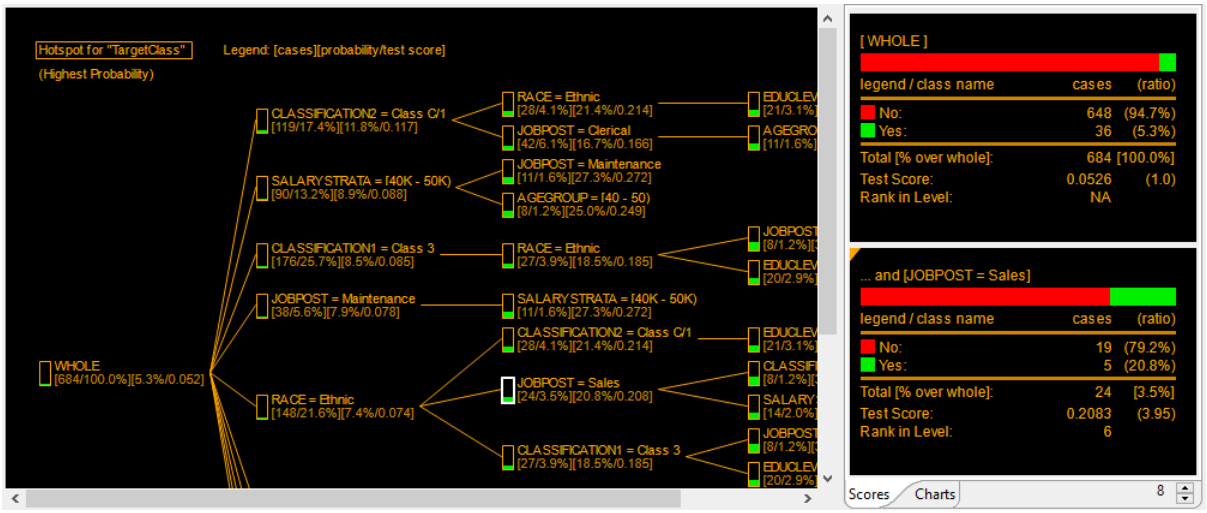
Normally a predictive model is developed using a ***target variable*** (aka ***dependent variable***) and a number of ***independent variables***. The dependent variable is the prediction target. For modeling purposes, target is coded with the following two variables. Note that we will use the following variable names in this paper. You may use different names such as RiskFlag, RiskClass, etc.

- **TargetFlag** : 0 and 1. If yes, then 1. No is 0. An integer data type is recommended as it can reduce dataset size when imported into CMSR.
- **TargetClass** : "Yes" or "No". (Note that you can use other values such as "Risk", "Safe", etc.) But these values will be used as references in this paper.

In insurance, "Yes" means insurance claims. In credit loans, it refers default and/or delinquency. For customer retention, it will mean customer churn. For marketing, it means customers responded positively. And so on.

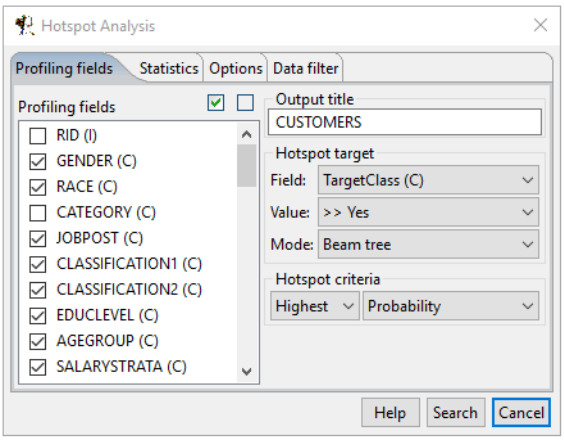
[Method 1] Customer Hotspot-Segment Profiling

CMSR Hotspot drill-down analysis can identify profiles of high-interest customer segments in terms of “TargetClass” ratios. It can identify segments with highest/lowest ratios of “Yes” or “No” of the “TargetClass” variable. The following figure shows a hotspot drill down chart.



The left chart is a beam search tree, drawn from left to right. From the root “WHOLE”, the next level shows a number of profiling variable-value conditions. They are displayed from highest (or lowest) probability to lower (or higher) probabilities. The number of entries in this level is limited by entering a number at creation. Note that nodes do not partition data. Data can be used in multiple nodes. At the next level, each node is further extended with another condition (or predicate) a number of times. To see details of a node, click the node with mouse. The right frames will show details of selected nodes.

To produce this hotspot drill-down charts, select a dataset. Then select “Hotspot drill-down analysis” from “Analytics” menu. Then the following dialog window will appear;



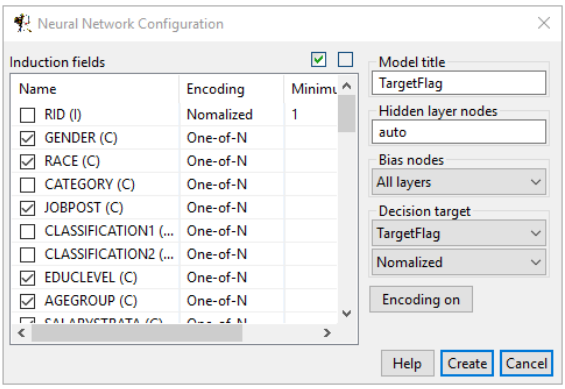
From the dialog window, select (or tick) all independent variables for “Profiling fields”. Select “TargetClass” for “Field:” in “Hotspot target”. Choose “>> Yes” for “Value:”. Then select “Highest Probability” for “Hotspot criteria”. Then press the “Search” button.

[Method 2] Customer Scoring (Neural Network)

Customer scores are numerical ratings of customers. Customer rating scores can be computed as values normally ranging between 0 and 1. (But can be a bit smaller than 0 or a bit larger than 1.) The variable "TargetFlag" can be used as target of predictive models. Neural network is a perfect technique for this. This section will describe customer scoring technique based on neural network. (Note that extended version of this customer scoring method is fully described in the paper "Modeling Guides for Neural Network.pdf". You are recommended to read the file.)

(1) Neural Network Configuration

Neural network modeling starts with network configuration. To configure a neural network model, open a dataset. If the dataset is already opened, select the dataset by clicking the tab of the dataset. Then select "New neural network" from the "Modeling" menu. Or simply press the neural network button from the toolbar. Then following dialog window will appear;



From "Induction fields", select a set of independent variables by ticking the boxes in front. Note that independent variable selection techniques are fully described in the section "2. Independent Variables: Relevancy Analysis and Feature Extraction" of the file "Modeling Guides for Neural Network.pdf".

If needed, change "Encoding", "Minimum" and "Maximum" values. To change this information, press the "Encoding on" button and click a variable. Then change the variable options.

For numerical variables, pay attention to "Minimum" and "Maximum" values to cover unseen future values. They should not be too small or too large. If needed, adjust them.

Then select "Decision target" as "TargetFlag".

Specify "Hidden layer nodes". This specifies a number of hidden layer nodes, e.g., 35, etc. When "auto" is specified, the number will be automatically calculated as the 60% of total input nodes. When it is empty, internal hidden layer will be omitted. This creates "regression" neural network.

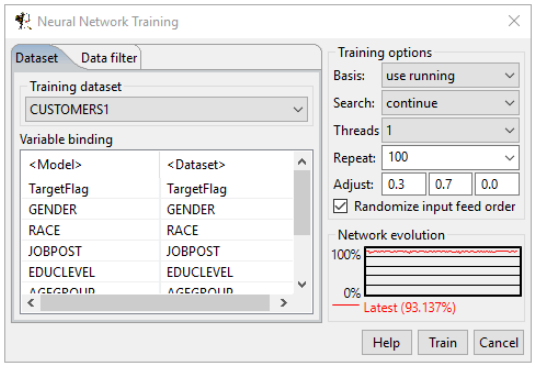
It is noted that too many hidden layer nodes can lead to overfitting. On the other hand, too few internal nodes will result in poor accuracy. So a proper internal node size is recommended. A proper size can be found through testing multiple models with different sizes. Try "auto". Then try higher and lower numbers of internal nodes.

Finally, enter "Model title" and press the "Create" button.

(2) Neural Network Training

Neural network training is repetitive process. You can train neural network many times. CMSR allows you to train network many different times, even using different datasets.

To train a neural network model, open a dataset and the model. Select the model to the front (of the main frame). Press the neural network button of the toolbar or select “Train neural network” from the “Modeling” menu. Then the following dialog window will appear;



If needed, select “Training dataset” and “Variable binding”. If you use the same dataset used in creating the model, CMSR will automatically bind variables correctly. If you use different dataset with different variable names, you will need to select “Variable binding” manually.

The “Network evolution” shows recent accuracy ratio history and the “Latest” accuracy value (“Latest (93.137%)” in the figure). Reading this, you can determine whether network is fully trained.

The most important training parameter is “Error correction ratio”. This is the first value of “Adjust:” in the above dialog window. Start training with “0.5” as the “Error correction ratio”. If the “Latest” accuracy value fluctuates or does not change significantly, then move to next correction ratio “0.3” until the latest accuracy value fluctuates or does not change significantly again. Then move to “0.1”, “0.05”, and finally “0.01”. When completed, network training is done.

Training can automatically repeat a number of times for each training run. The “Repeat:” option specifies this. If you choose small numbers, you may have to repeat many training runs. If you choose too large numbers, training runs may take long time. A suitable number can be “1,000,000 / (records in training dataset)” or “5,000,000 / (records in training dataset)”. If the dataset is larger than that, you may choose 5 or 10.

Neural network models maintain two versions: the ***best*** and the ***running*** versions. The best version is automatically searched. If you want to reset the best version, select an option from “Search:”.

Make sure to save the model after network training. To save, press the “Save model as” button. It is the first toolbar button.

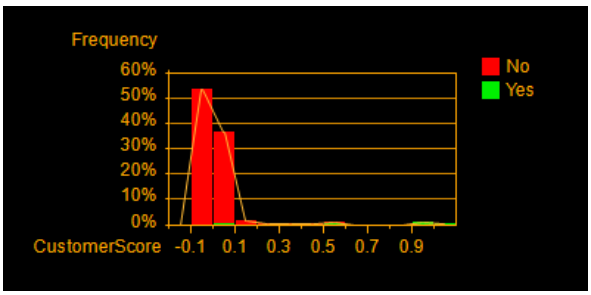
(3) Database Update and Model Fitness Testing

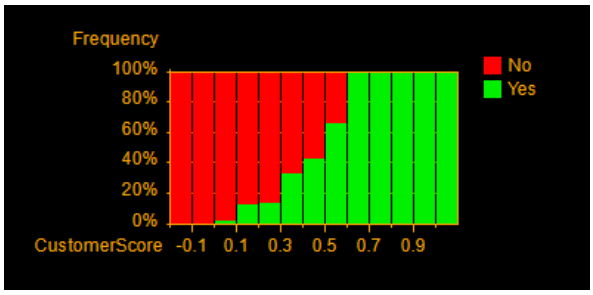
Once a neural network model is fully trained, the next step is to apply model to database table and analyse model characteristics. This is called model ***fitness testing***. To test a model, the following preparation is needed.

(1) Create a database table column, say, "CustomerScore" on the dataset table. A float data type such as "float" or "double" is required for this score column.
(2) Open the neural network model. Select a database and login. Press the "Apply to database" button and update the database table field "CustomerScore" with model's output values.

This will store model's output values which are customer scores. These values can be used to prioritize customer services.

Now it's time to analyse how the model's output values are distributed. The best way to view model desirability is to use ***histogram*** charts. The following charts are histograms showing predicted customer scores and its distribution of "Yes" and "No".





The left figure shows distribution of dataset samples over predicted "CustomerScore" values. Most samples scored less than 0.1. This is because most customers are of "0" (="No") "TargetFlag" value. The right figure shows relative "Yes" proportions by intervals. Red parts show "Yes" proportions of intervals. The higher the predicted score is, the higher the "Yes" (=red) proportion is. This is a very good attribute of good predictive models.

This chart was produced as follows. Select the database and login. From the "Database" menu, select "Histograms". From "Database table", select the training dataset database table. Select "CustomerScore" as "Value:". And select "TargetClass" as "Category:". Then the left figure chart will show up. To get the right figure chart, select "Categoric proportion" for the "Percent" option.

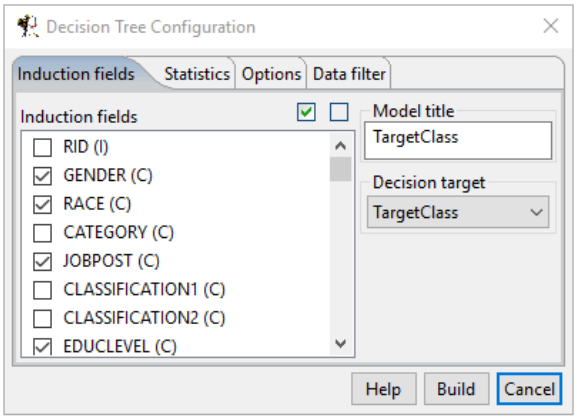
More model fitness testing techniques are described in the section "9. Model Fitness Testing" of the file "Modeling Guides for Neural Network.pdf". You are recommended to read it.

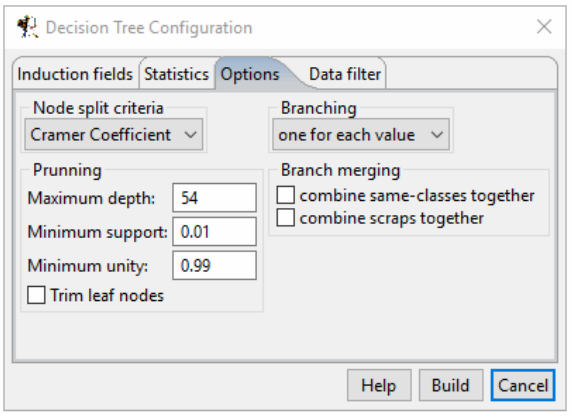
[Method 3] Supervised Customer Segmentation (Decision Tree)

Decision tree is normally used for classification tasks. But it can be used as a segmentation tool as well. Decision tree requires the "TargetClass" variable for supervised learning. In this section, how decision tree can be used as a segmentation tool is described.

(1) Segmentation Decision Tree

To create a decision tree for segmentation, open a dataset. Then select "New decision tree model" from the "Modeling" menu or simply press the decision tree toolbar button. Then following dialog window will appear.

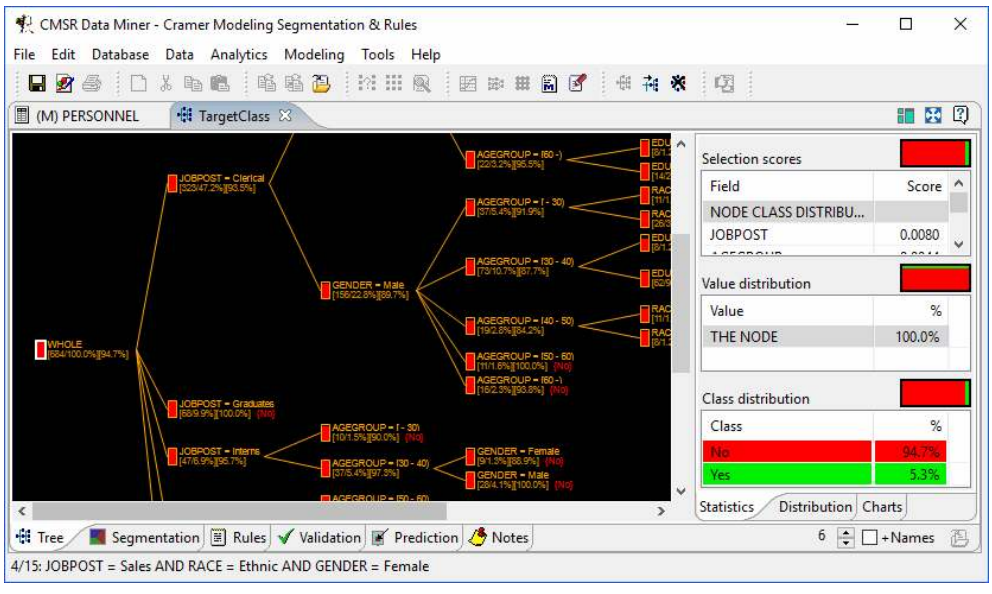




From the "Induction fields", select a set of variables to be used as segmentation variables. For "Decision Target", select the "TargetClass" variable.

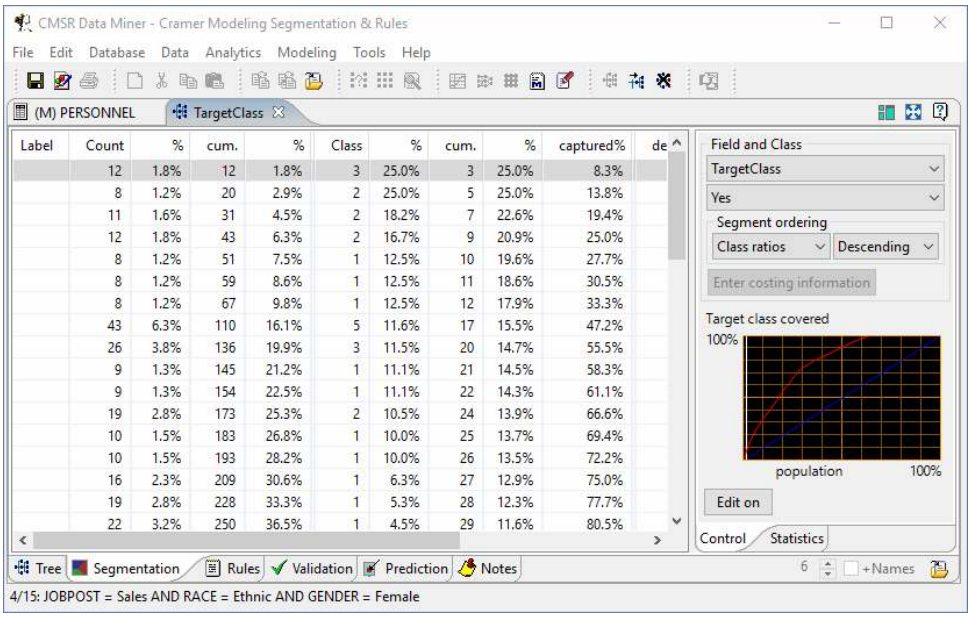
If the "Yes" value of "TargetClass" is severely skewed, decision tree will be empty, consisting of only the root node. Or result in only few segments. This normally happens if the ratio of "Yes" is below, say, 30% or over 70%. In this case, still decision tree for segmentation purposes can be produced by disabling the "Trim leaf nodes" of the "Options" tab (at the right figure). This will prevent tree pruning.

Finally, press the "Build" button. Then the following decision tree will appear.



(2) Segmentation and Gains Analysis

The next step is to prioritize segments produced by decision tree. This is done by sorting segments on ratios of "Yes" of the "TargetClass" variable descendingly. That is, segments with higher "Yes" ratio first. To do this, click "Segmentation tab". Then the following figure will appear.



To rearrange segments with higher "Yes" ratio first, select "Field and Class" as "TargetClass" and "Yes". Then segments will rearrange based on "Yes" ratios as shown in the above figure.

"Target class covered" shows the gains chart. A rapidly rising red line is a good indication of good segmentation. This means that customers with the "Yes" value are concentrated on earlier segments.

You may wish to give segments label names. Label names can be used to update customer database table. Press the "Edit on" button. Edit the "Label" column values which are the first column of the table. When finished, press the "Edit off" button.

Make sure to save the model for later use. To save, press the "Save model as" button. It is the first toolbar button.

(3) Updating Customer Database Table with Segment Labels

Once a model is ready, the next step is to update customer database table with segment labels. To update customer database table, follow these steps;

(1) Create a database table column, say, "CustomerSegment" on the customer dataset table. A character string data type, such as VARCHAR(20), is required for this segment label column.
(2) Open the decision tree model. Select a database and login. Press the "Apply to database" button. Select a schema and table. Select "CustomerSegment" for "Store field:" in the "Prediction save" box. Select "Segment labels" for "Values to save:". Select other fields. Finally, press the "Apply" button.

This will store model's segment labels to customer database table.

[Method 4] Unsupervised Customer Segmentation (Neural Clustering)

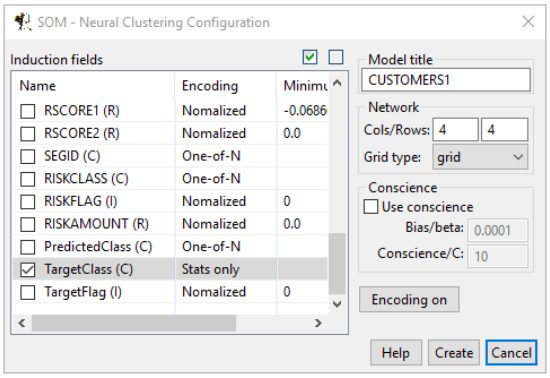
Neural clustering, aka SOM (Self-Organizing Maps), can organize customers with similar attributes together, into m x n cells. Neural clustering learns customer similarity automatically without guidance of target variables, thus unsupervised learning. In other words, the "TargetClass" variable will not be used as a clustering variable. But it will be used for statistical purposes only.

Due to the nature of unsupervised learning, clustering may not produce useful result. Analysing "TargetClass" distribution of cluster cells can be used to determine usefulness of clustering.

In this section, neural clustering based customer segmentation will be described.

(1) Neural Clustering Network Configuration

To create a neural clustering model, open a dataset. Then select "New neural clustering (SOM)" from the "Modeling" menu or simply press the neural clustering toolbar button. Then following dialog window will appear.



Select a set of "Induction fields" which are used to compute customer similarity. In addition, select the "TargetClass" variable, as shown in the above figure. "Encoding" of the "TargetClass" variable should be "Stats only" as it should not be used in computing similarity. To change "Encoding", press the "Encoding on" button and change. When finished, press the button again.

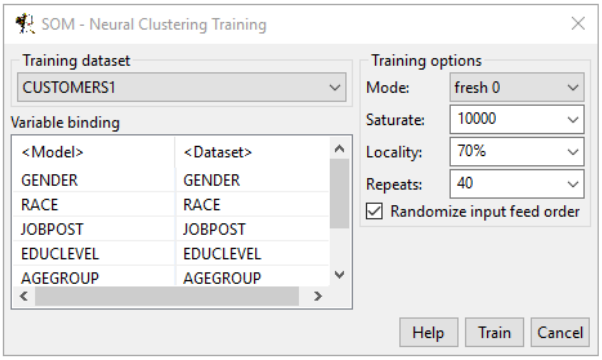
Enter "Cols/Rows" of the "Network" box. This specifies model grid size. Numbers should be between 1 and 100. One dimensional array can be specified by entering 1 for either for columns or for rows.

Press the "Create" button and proceed with network training.

(2) Neural Clustering Network Training

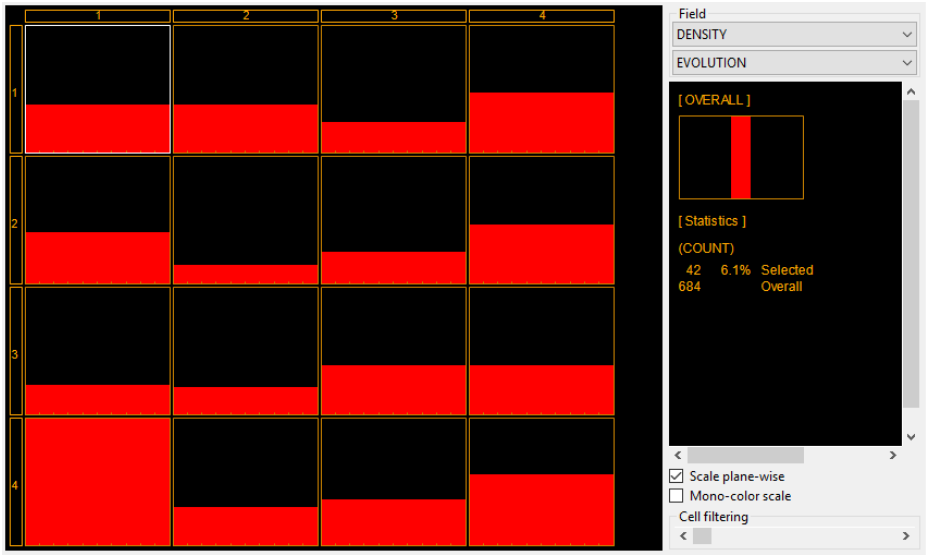
Network training is repetitive process. You can train network many times. CMSR allows you to train network many different times.

To train a neural network model, open a dataset and the neural clustering model. Select the model to the front (of the main frame). Press the neural clustering button of the toolbar or select “Train neural clustering (SOM)” from the “Modeling” menu. Then following dialog window will appear.



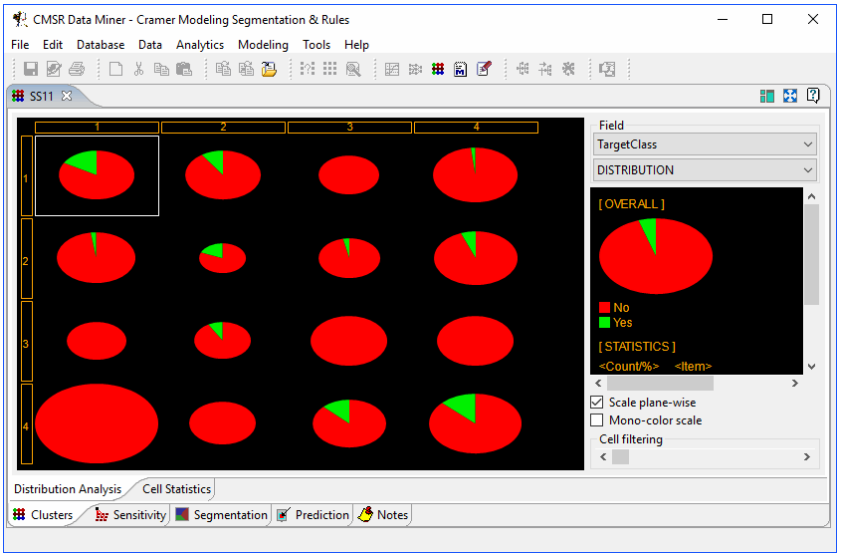
If you open the dataset used in creation of this model, it will automatically perform “Variable binding”. Otherwise select “Training dataset” and setup “Variable binding” manually.

Enter or select “Repeats” in the “Training options” box and press the “Train” button. Then network training will start. After each repeat, the following network evolution chart will be updated. Each cell shows network evolution chart for the most recent 10 repeats. It indicates relative size of numbers of customers of cells. When charts are flat, network didn’t change during recent 10 repeats. It indicates network training is completed. Perform some extra repeats.



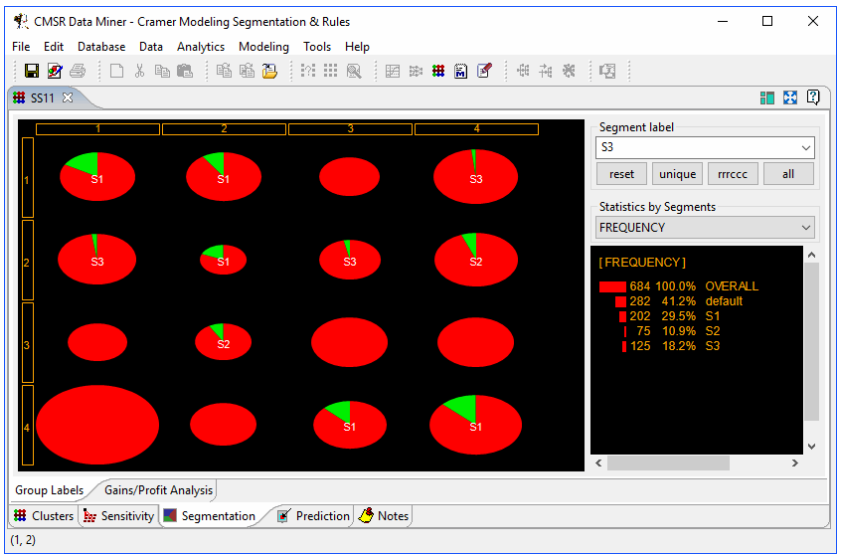
(3) Cell Performance Analysis and Grouping

Once network training is completed, the next step is to analyse cells whether some cells have higher rates of “Yes” customers (of “TargetClass”). To see distribution of “Yes” customers, select “TargetClass” for the “Field” box, as shown in the following figure.



Each cell shows a pie chart for the “TargetClass” variable. Green slices are “Yes” portions. Some cells have high “Yes” proportions. Some have lower proportions. Others don’t. This is exactly what we are looking for. If there is no this pattern in cells, then you may go back to the previous step “(2) Neural Clustering Network Training” and train network further. Still if there is no pattern, then try different models with different combinations of “Induction fields”. If this also does not produce patterns, clustering may work for your data!

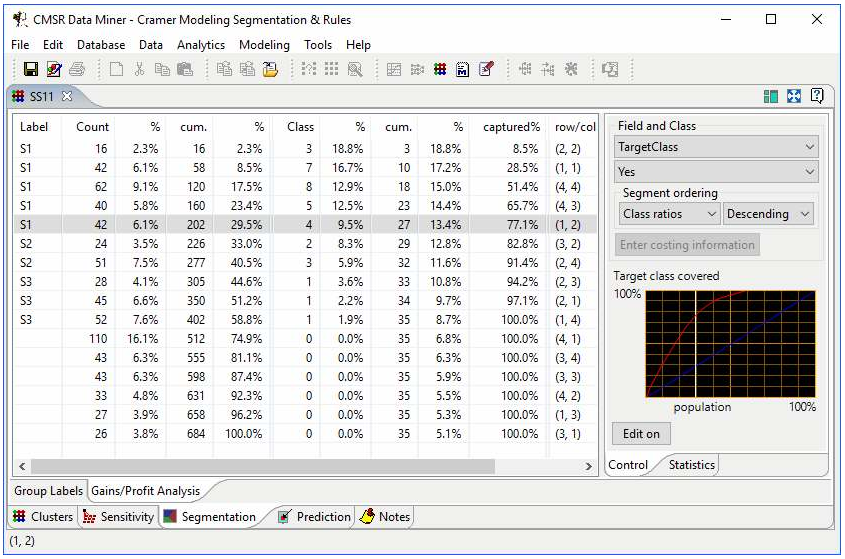
Based on the proportion sizes, cells can be grouped together to form customer segments. To do this, click (=select) the “Segmentation” tab. Then select “Group labels”. The screen will change as follows;



This example shows “S1”, “S2”, “S3” and empty “” segment names. To name cells, enter a name at “Segment label”. Then click cells that will have the name using the mouse. Cell names will change to the “Segment label” values.

(4) Gain/Lift Analysis

Next is gains/lift analysis. This is done by sorting cells on ratios of "Yes" of the "TargetClass" descendingly. That is, cells with higher "Yes" ratio first. To do this, click the "Segmentation" tab. Then click the "Gains/Profit Analysis" tab again. Then the screen will change as follows;



To rearrange segments with higher "Yes" ratio first, select "Field and Class" as "TargetClass" and "Yes". Then segments will rearrange based on "Yes" ratios as shown in the above figure.

"Target class covered" shows the gains chart. A rapidly rising red line is a good indication of good segmentation. This means that customers with the "Yes" value are concentrated on earlier segments (=cells).

You may wish to change cell label names. Press the "Edit on" button. Edit the "Label" column values which are the first column of the table. When finished, press the "Edit off" button.

Make sure to save the model for later use. To save, press the "Save model as" button. It is the first toolbar button.

(5) Updating Customer Database Table with Segment Labels

Once segmentation is ready, the next step is to update customer database table with segment labels. To update customer database table, follow these steps;

(1) Create a database table column, say, "CustomerSegment" on the customer dataset table. A character string data type, such as VARCHAR(20), is required for this segment label column.
(2) Open the neural clustering model. Click the "Prediction" tab. Make sure that "Field:" of "Prediction Value" is "SEGMENT/GROUP LABELS".
(3) Select a database and login. Press the "Apply to database" button. Select a schema and table. Select "CustomerSegment" for "Store field:" in the "Prediction save" box. Select other fields. Finally, press the "Apply" button.

This will store model's segment labels to customer database table.