**Creating Decision Trees.**

**Objectives:**

The aim of the lab is to use SAS Enterprise Miner:

* To build a decision tree model on the HMEQ (Home Equity) dataset and to examine and interpret the results.

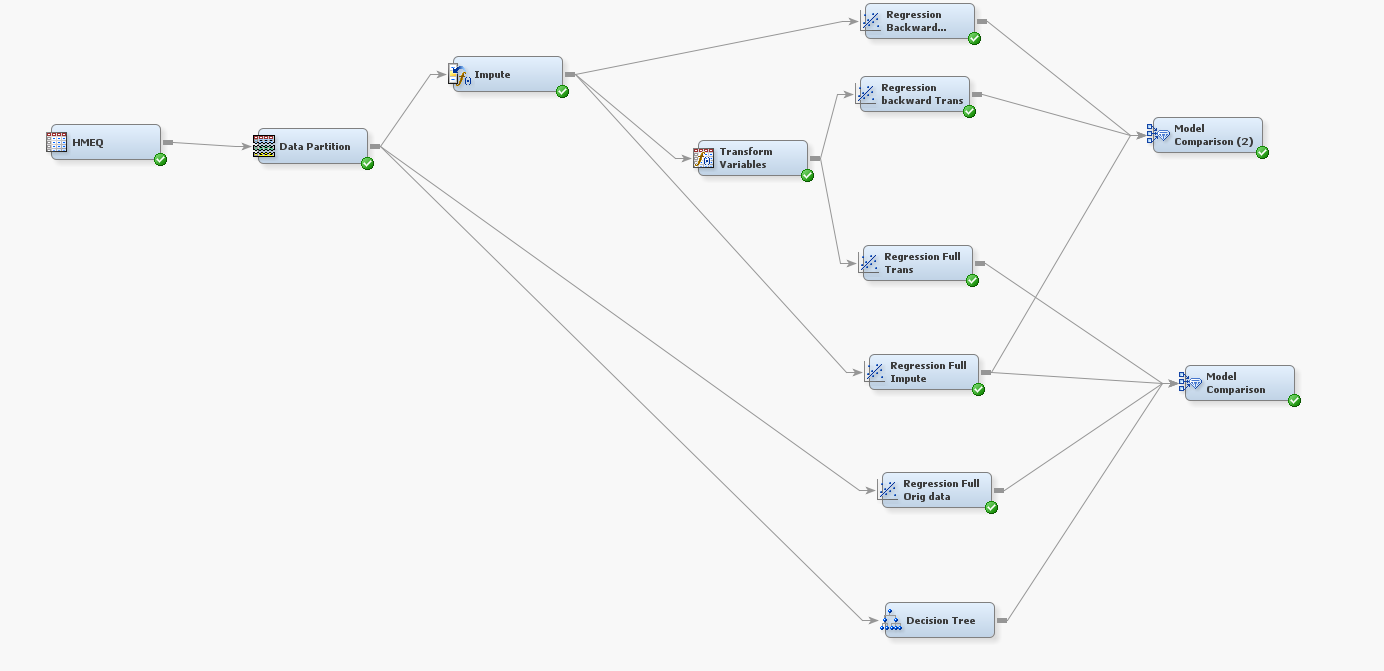
**By the end of this lab you will:**

* have explored the types of decision tree models available in SAS Enterprise Miner, built a decision tree model, examined and interpreted the model results such as the decision tree, the tree ring, the plot of the training and validation data set.
* Explored the effect of different settings including number of branch points, minimum number of observations in a leaf, and different purity metrics.
* Generated a manual tree and compared it to the trees generated automatically

**Task 1: Fitting and evaluating a Default Decision Tree**

Building on Lab 5

In SAS Enterprise Miner, open the HMEQ process flow diagram (**HMEQ.dmd**) that you built in the previous Lesson.



* **Add the Tree node** after the Data Partition node.
  1. Connect the ***Tree*** node after ***the Data Partition node.***



2. Obtain the decision tree model by right-clicking the ***Tree node*** and selecting **Run**. Just use the defaults.

Recall (from lecture): A decision tree handles missing values so a Data Replacement node is not required – hence the reason the Tree node is connected directly to the Data Partition node.

* Go to the **View** **menu ->** **select Tree** to obtain the decision tree. Run the node and examine the results

**Qu1:** Interpret (i.e. draw out the main information from) this decision tree.

What is the critical path ? This is the rule which leads to the leaf with the most observations

Which nodes have the target variable =1 ? Also know as the target path

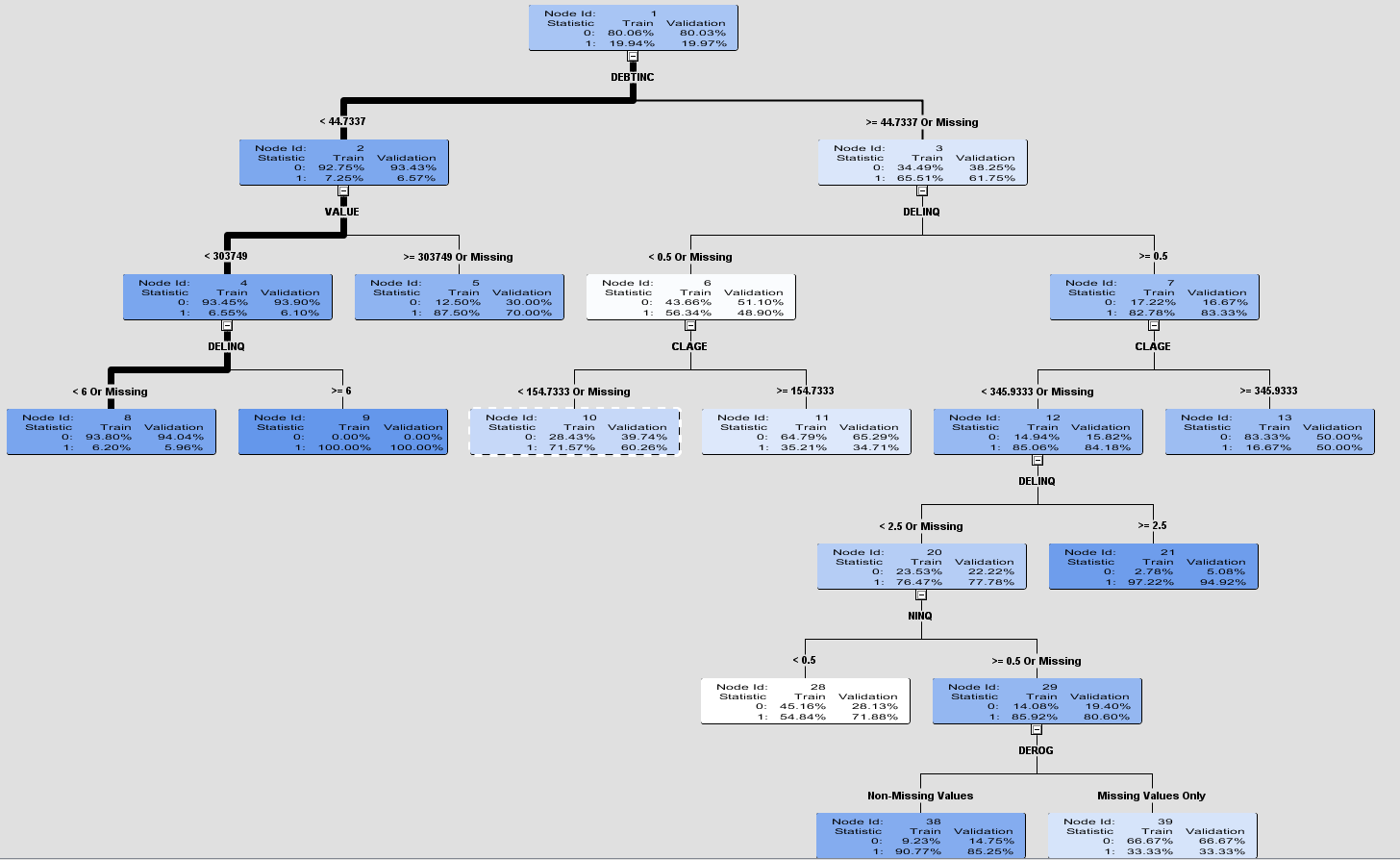
How reliable are the rules at the leaf, is there good correspondence between the validation and partition data ? Are the percentages for the target classes the same if so it indicates the rule is reliable.

Do you get the feeling that the tree is overfitted ? Does the validation data perform better than the training data ? As the tree becomes more complex does model performance plateau.

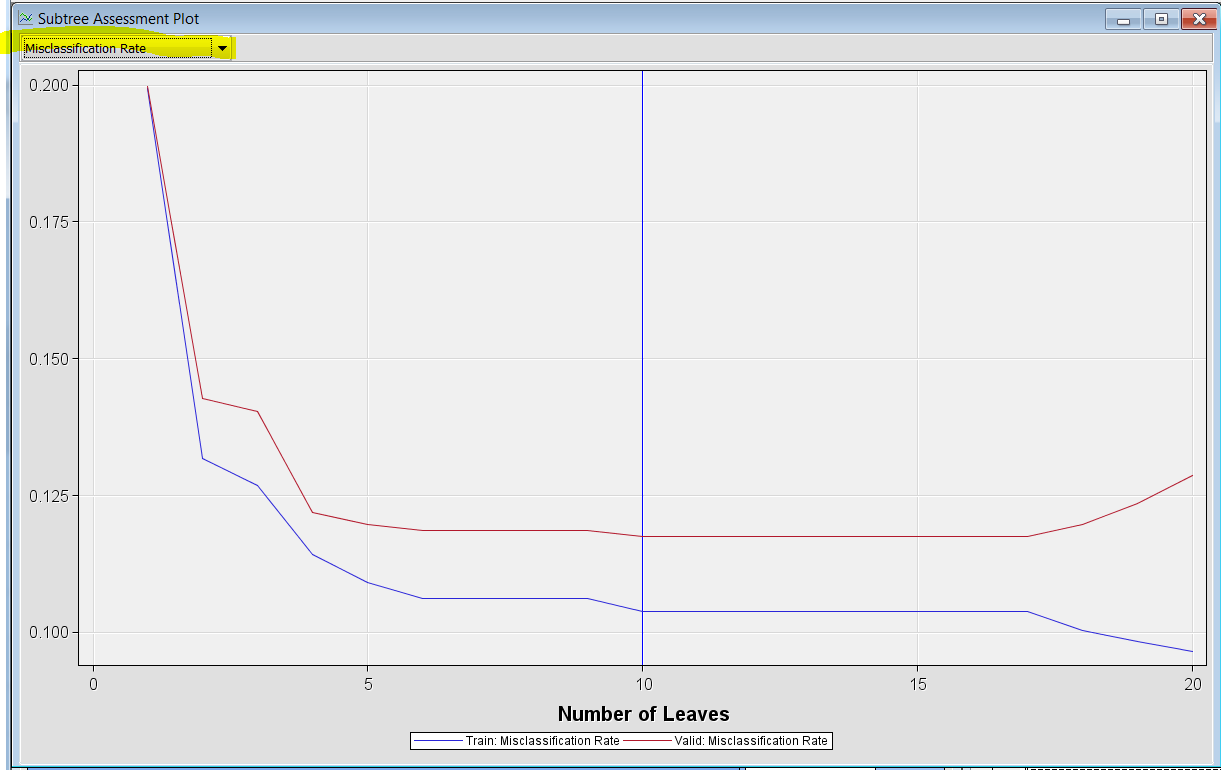
Write out the most important rules ? Critical path and the target path.

Is there any correspondence with the regression model ? Are the same independent variables used ?

* **Close the Tree window** .



View > Model > subtree assessment plot, select misclassification rate from the drop down box in the upper right hand corner.

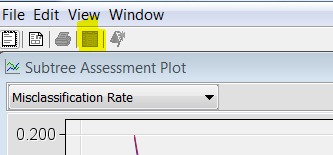


From the plot (in the lower-right corner) you can see that a tree with 20 leaves was originally grown based on the training data set and pruned back to a tree with 10 leaves based on the validation data set.

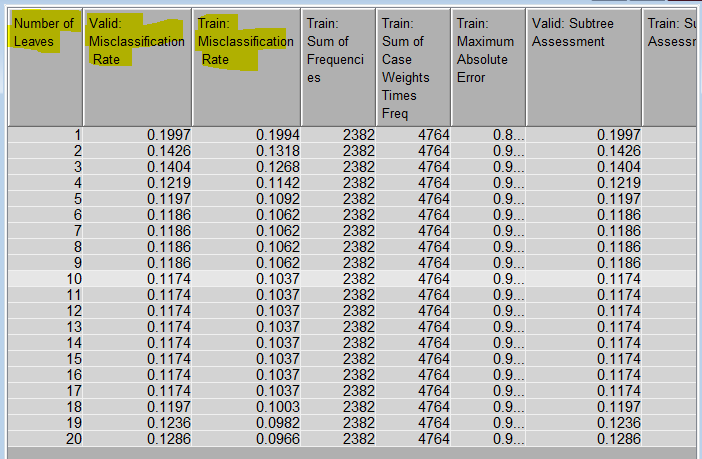
**Qu2:** Interpret the plot ? At what point is the data overfitted ?

(Note the Validation data is above the Training data ie is more erroneous).

Click on the table icon



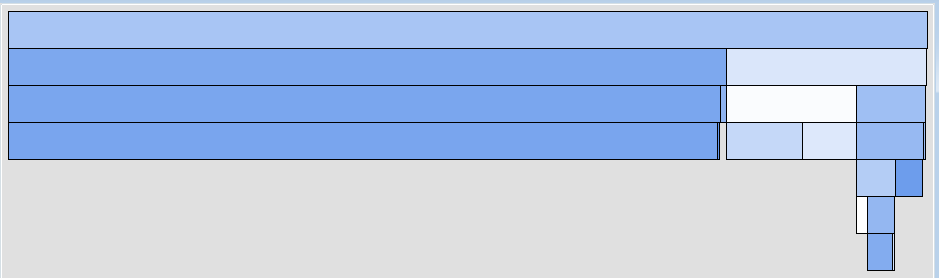
The following table is produced



Move the columns around by dragging them with a mouse to display number of leafs, valid misclassification rate and train misclassification rate next to each other.

The table in the lower-left corner shows that the 10-leaf model has a misclassification rate of 11.74% on the validation data set.

**Next > view > model >the Tree Map**

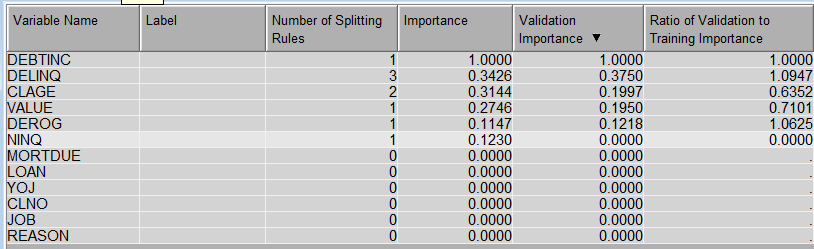


**Qu3:** Interpret the tree map – relate it to the tree diagram, the deeper the colour the greater the leaf purity, the greater the area the more observations in the leaf. If you are not sure compare the tree map to the decision tree, they describe the same thing.

View the importance variable table, interpret the output ? The bigger the value the more discriminating of the two target classes that variable is.

You can also determine the variables that were important in growing the tree,

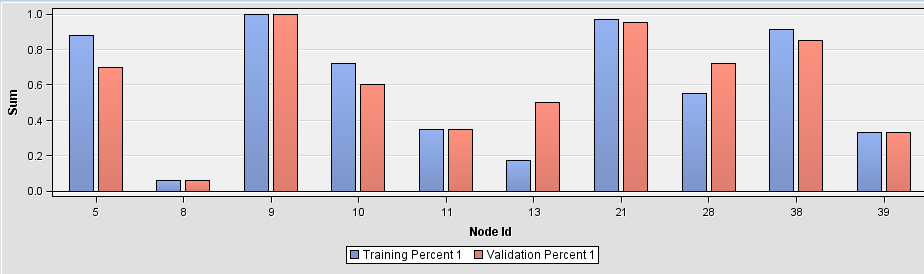
**By selecting the variable importance table, select model from the view menu, variable importance table.**

The Variable importance table displays the relative importance of variables used in growing the tree. It also can be used to export new variable roles.

**Qu4:** Which variable was the most important in growing the tree? Which variable was the 2nd most important? Which variable was the 3rd most important?

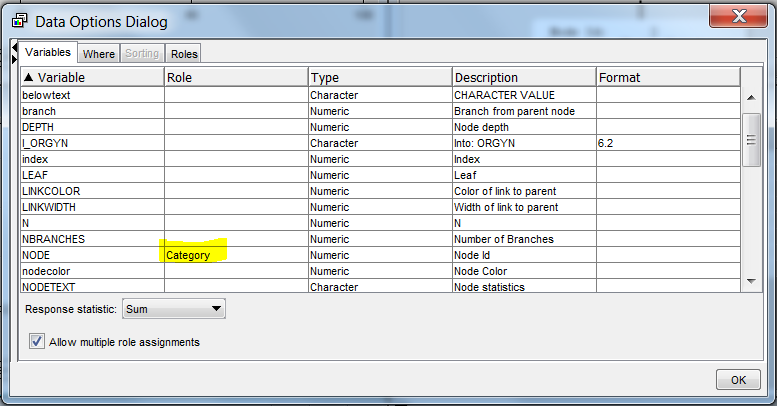
* **Close the Results window and save the changes when prompted.**

There are two further plots the leaf statistics plot and the variable histogram plot.

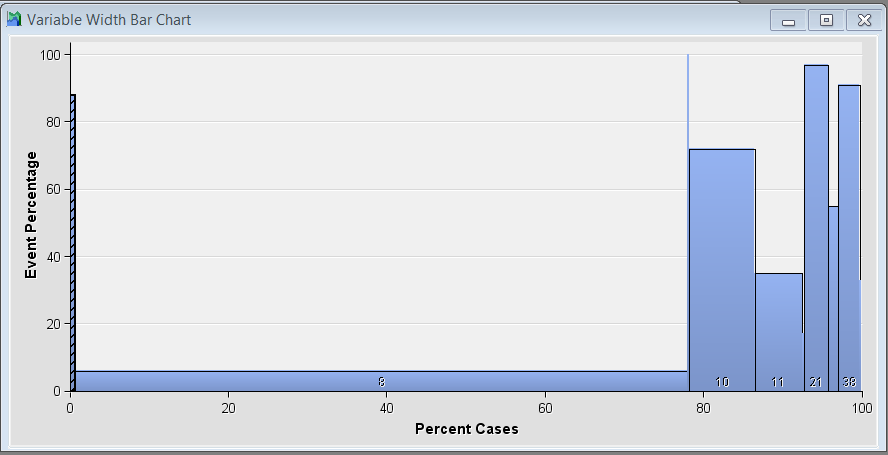


The leaf statistics plot provides you with information as to the reliability of a node. Change the settings to display the node id right click on the graph and select data options from the hierarchical menu.

**Qu 5**. Which rules are the most unreliable ? If they are reliable you would expect to get the same performance between the model sets.



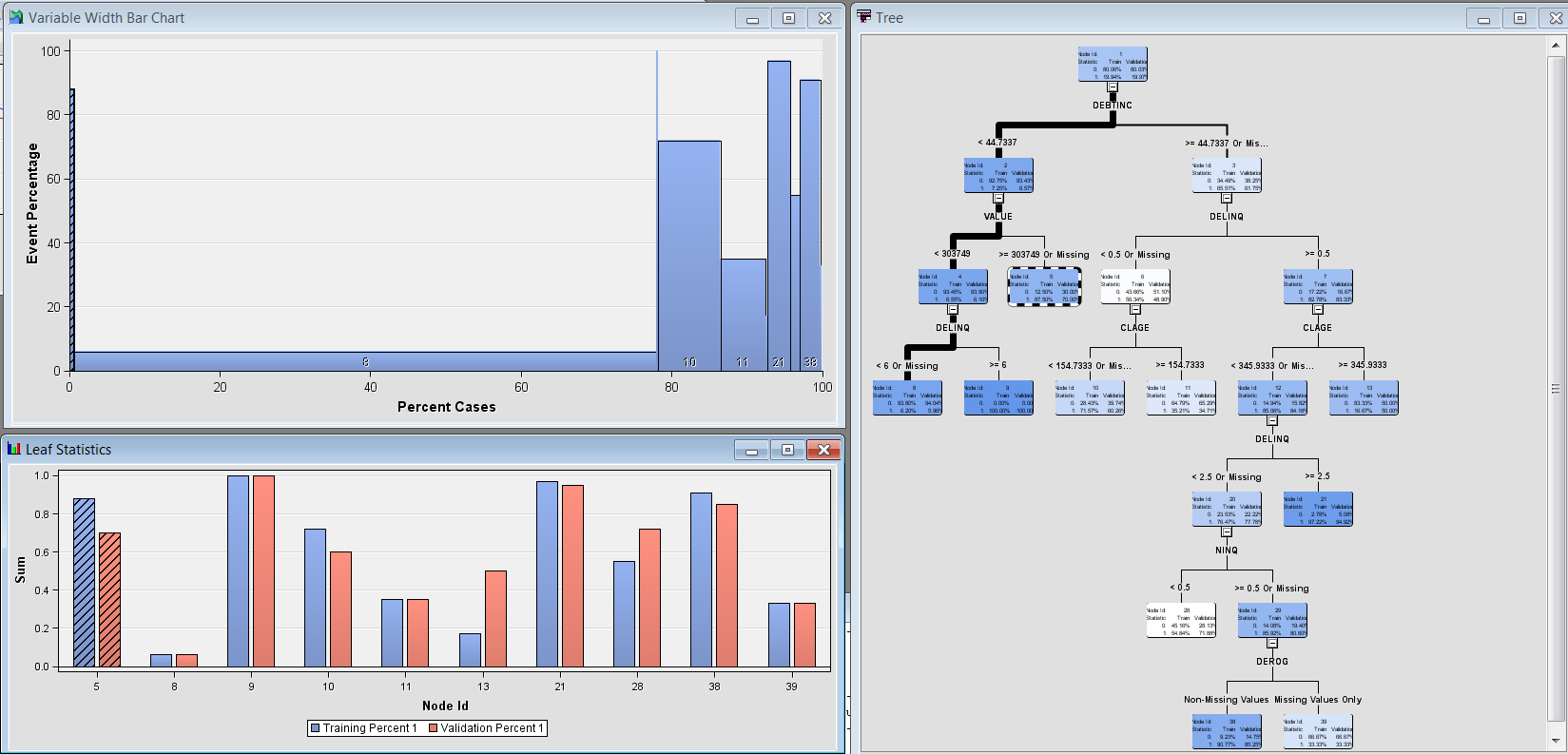
View the variable histogram plot



Repeat the above process to display node id.

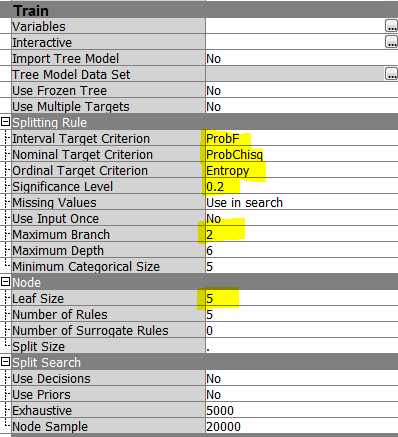
The variable histogram plot shows you which leaves have the highest response, % target =1 > 50%, the width of the histogram relates to the number of observations. The greater the area the more important the leaf. Related these two diagrams to the decision tree produced earlier.

Notice SAS uses a brushing feature to interconnect diagrams



**Do you have any new observations** ?

Close the results window and click on the tree node to display the default properties for the splitting diagram.



Many of the options for building a decision tree are controlled in the properties dialog box.

* The splitting criteria available depend on the measurement level of the target variable.

For binary or nominal target variables, the default splitting criterion is the chi-square test with a significance level of 0.2. (Alternatively, you could choose to use entropy reduction or Gini reduction as the splitting criterion.

For an ordinal target variable, only entropy reduction or Gini reduction are available.

For an interval target variable, you have a choice of two splitting criteria: the default F test or variance reduction.

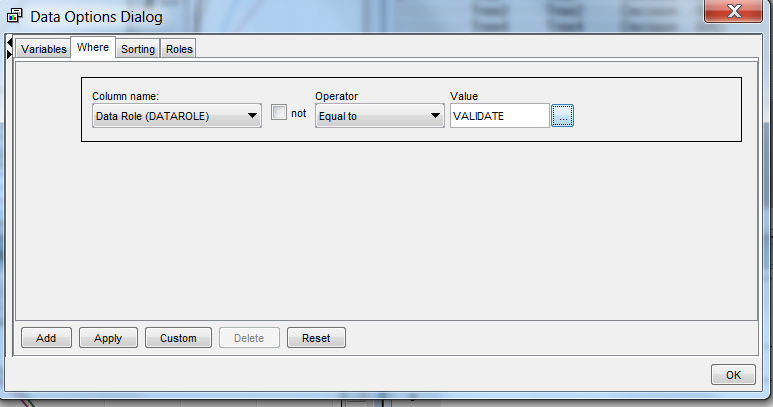
* The other options available in the Basic tab affect the growth and size of the tree. By default, only binary splits are permitted, the maximum depth of the tree is 6 levels, and the minimum number of observations in a leaf is the larger of (5 or the total number of observations divided by 1000).
* However, there is also a setting for the required number of observations in a node in order to split the node. The default is the total number of observations available in the training data set divided by 100.

Consult the appendix for defaults for CHAID algorithm or see online reference for details.

We are now going to assess the decision trees performance.

* **Connect the Tree node to the Model comparison node ->** **run. Select Yes** to view the results when prompted.

Display the lift chart, by default you will get a lift chart for training, validation and test. You can filter by clicking on the chart right click and select data options and move to the where tab.



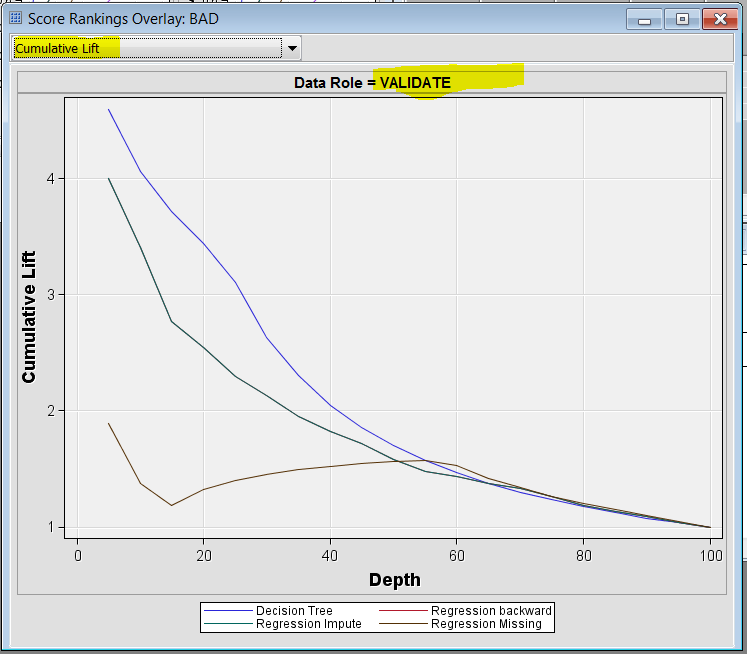
Filter to just view Validation data, right click data options > where select the options above select >add data role = validation, then select apply

*Only view the test data if you want to assess model performance on unseen data. If you want to compare models use validation data*

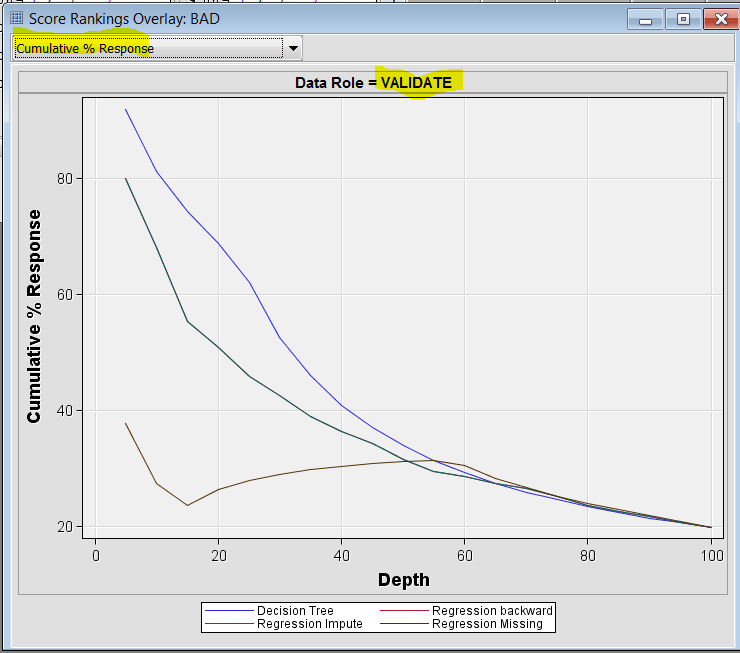
Normally we would examine for validation data for assessing model performance

The top drop down box on the left allows you to view four chart variations describing the same data

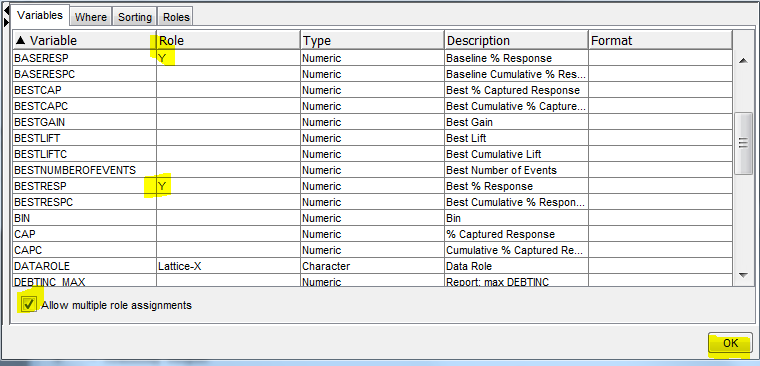
**Cumulative Lift**



**Cumulative Response**



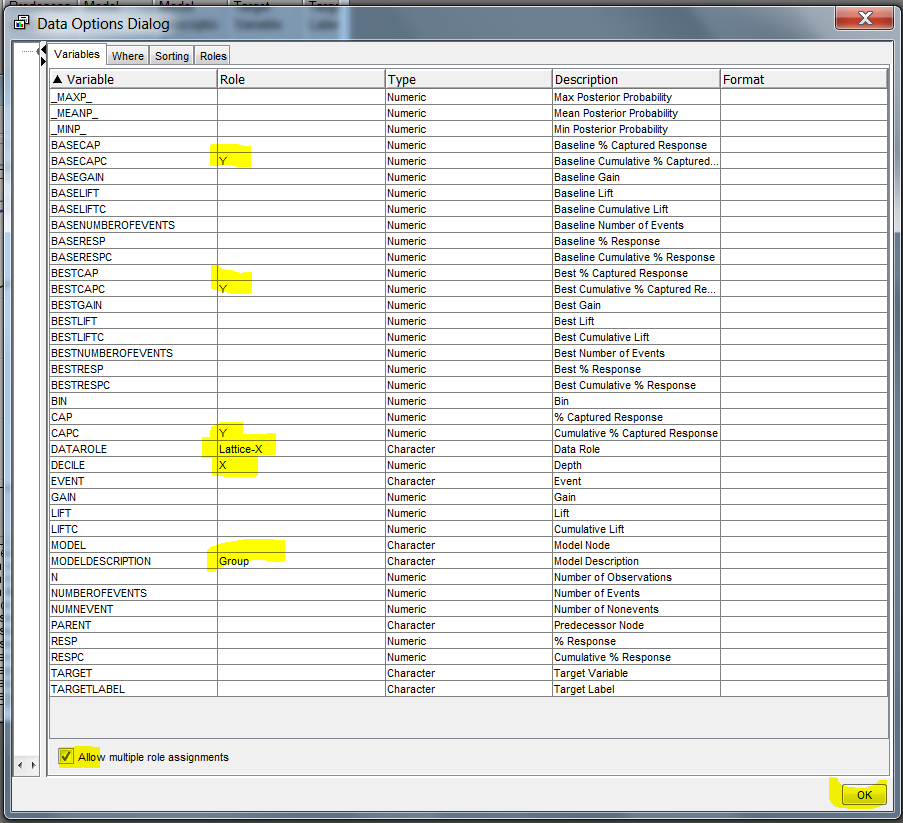
**Non-Cumulative %Response Lift Chart**

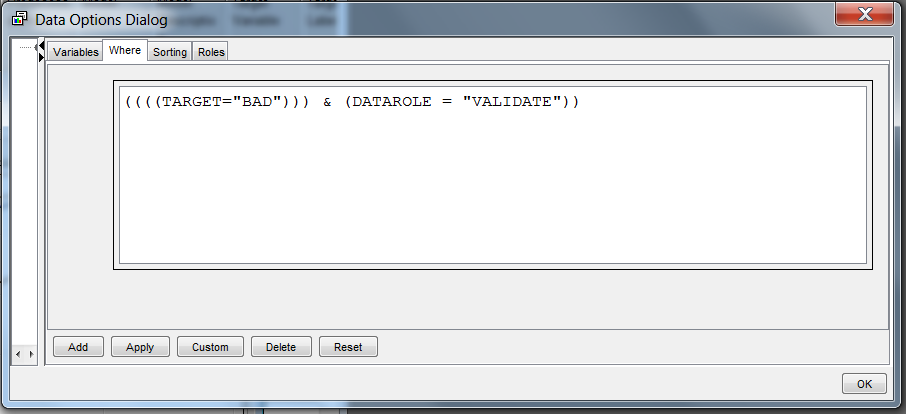




**Cumulative Captured Response LiftChart**

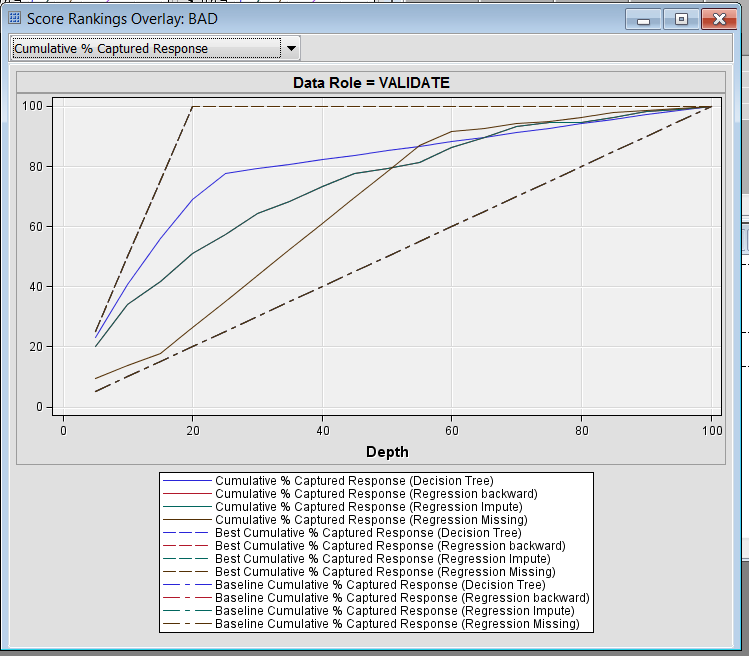
Right click on the chart select the following data roles





Referring to the various lift charts,

**Qu6:** Is the tree model useful?Comment.



**Qu7:** Interpret the Cumulative %Response Lift Chart, the Non-Cumulative %Response Lift Chart and the %Captured Response Chart for the decision tree model.

**Qu8:** So for the HMEQ data, which data mining model is better?Explain.

When you have finished viewing the lift charts

* **close the various windows**

**Task 3: Making adjustments to the default tree algorithm.**

Adjustments may be made to the default tree algorithm causing the tree to grow differently - these changes do *not necessarily* improve the classification performance of the tree, but they may improve its interpretability.

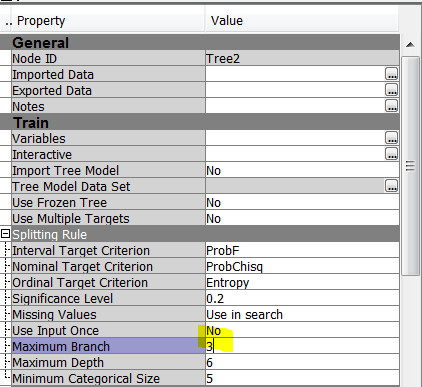
By default, the Tree node splits a node into 2 nodes (called binary splits).

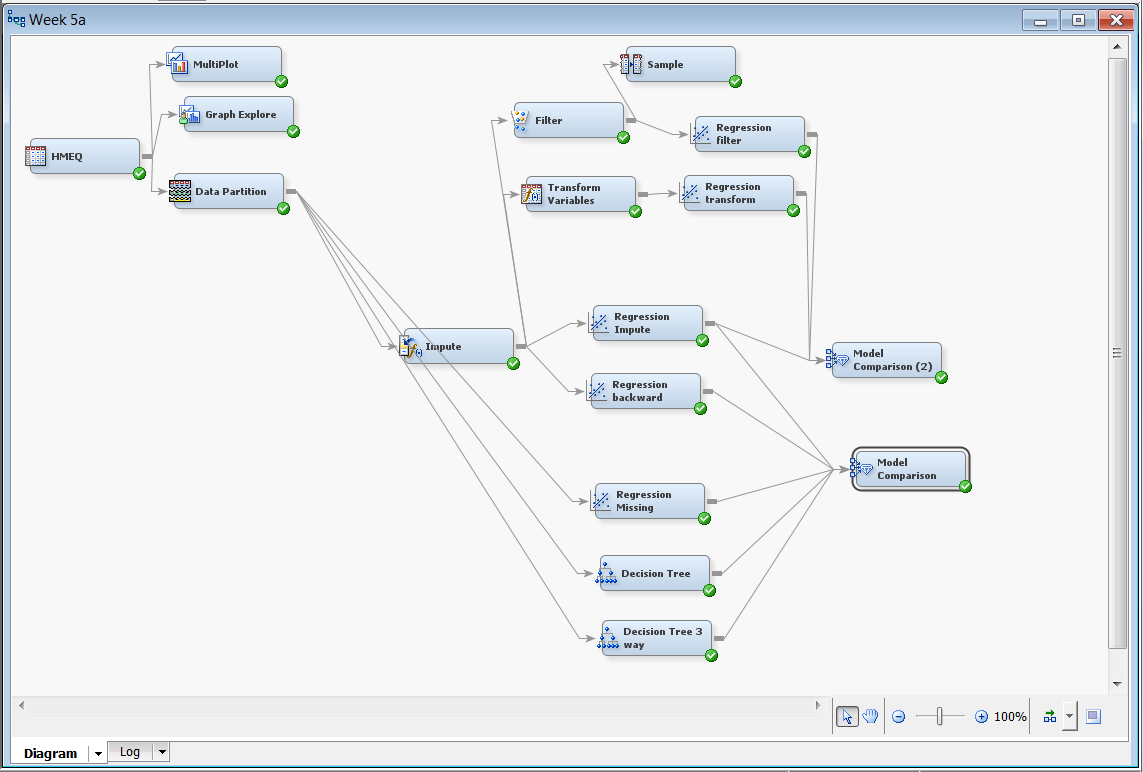
In theory, trees using multi-way splits are no more flexible or powerful than trees using binary splits. However let’s consider if a competing tree that allows up to 3-way splits increases the tree’s interpretability.

In the process flow diagram,

* **Add another Tree node** in the diagram workspace and **change the label to Tree3way. Right click select rename.**

**Select the properties dialog box for the node. Change maximum branch from 3 to 2.**





* **Obtain the various lift charts comparing the 3-way split Decision Tree with the 2-way split Decision Tree.**

**Qu9:** a) What are the main differences in the results with the 3-way tree (in comparison to the 2-way tree)?

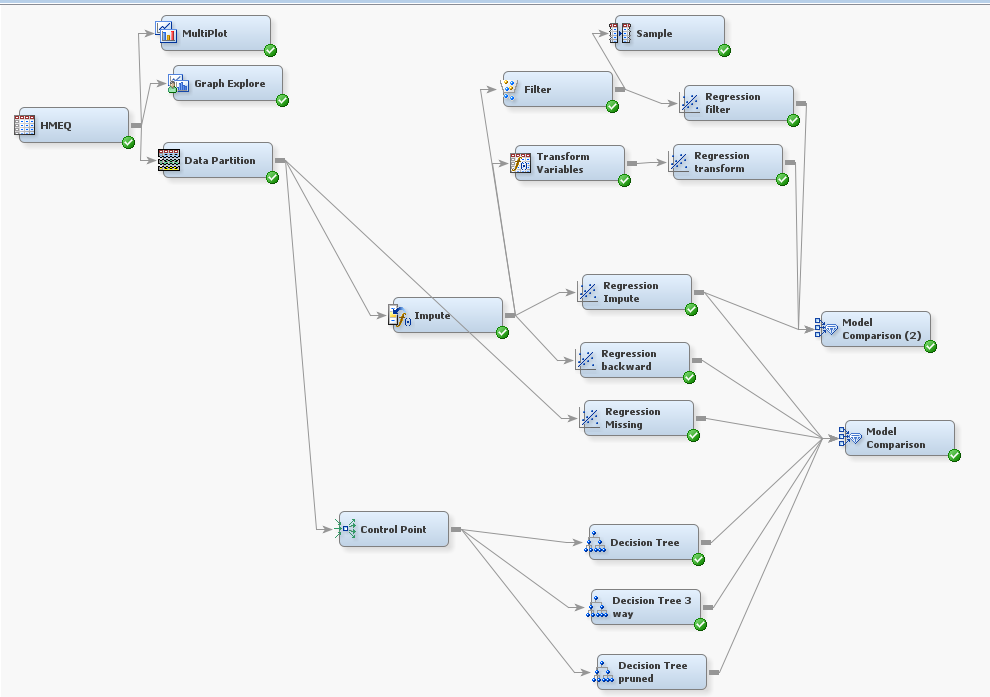
b) Is the 3-way tree more interpretable than the 2-way tree? Comment.

c) By interpreting the lift charts, which decision tree model is the best: the 2-way split decision tree or the 3-way split decision tree?

**Task 4: Limiting Tree Growth**

Various stopping or stunting rules (also known as prepruning) can be used to limit the growth of a decision tree. For example, it may be deemed beneficial not to split a node with fewer than 50 cases and require that each node have at least 25 cases.

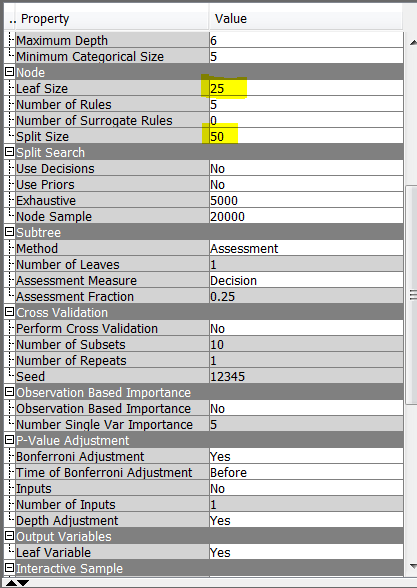
Modify the Tree3way node and employ these stunting rules to keep the tree from generating so many small terminal nodes:



Note a control node is added to the workflow diagram to tidy up the diagram. The control point does nothing.

Add another decision tree node, label it prued

* **Open the 3wayTree node and select the properties dialog box**



* **Type 25 for the minimum number of observations in a leaf and then press the Enter key.**
* **Type 50 for the number of observations required for a split search and then press the Enter key.**

NoteThe Decision Tree node requires that:

(Observations required for a split search) ≥ 2\*(Minimum number of observations

in a leaf).

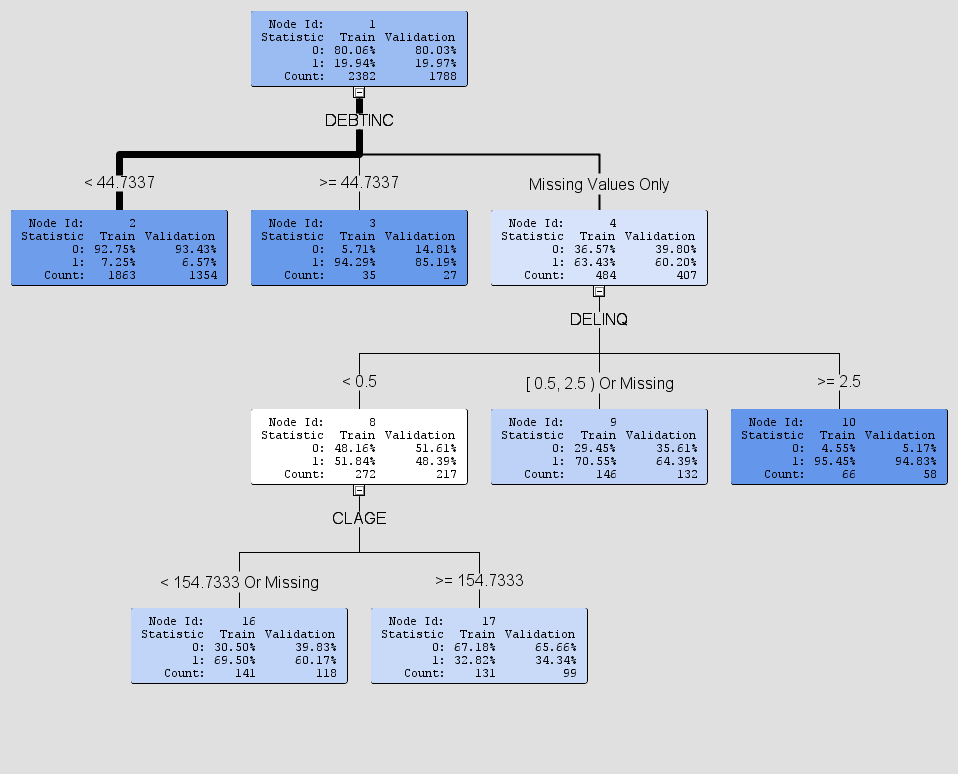
In this example, the observations required for a split search must be greater than

2\*25=50. A node with fewer than 50 observations cannot be split into two nodes

with each having at least 25 observations.

If you specify numbers that violate this requirement, you will not be able to close

the window.

* **Rerun the Tree node and view the results as before.**
* 

**Qu10:** Compare the ‘stunted’ 3-way tree to the ‘unstunted’ 3-way tree obtained in Task3. Examine the model performance, is there a big penalty for having a much more simpler tree, more parsimonious.

**Generating a Manual Decision Trees.**

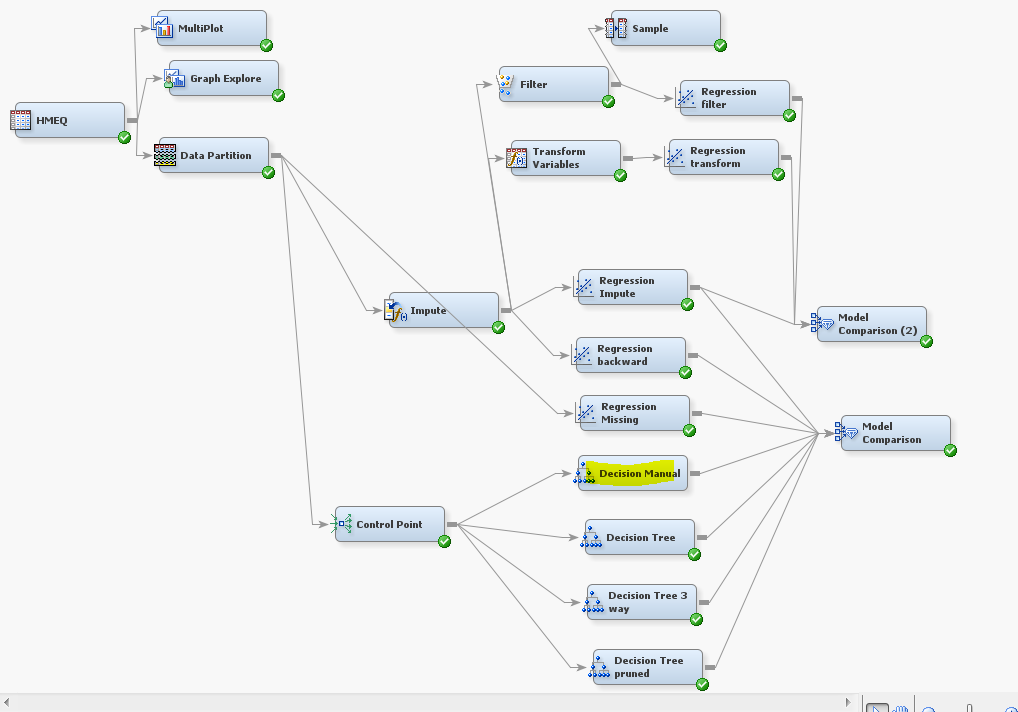
**Objectives:**

The aim of this lab is:

To interpret the results of decision tree model in more detail by building a tree by hand you will be able to appreciate the tree growing and pruning process and be better able to asses if the “automatic trees” are overfitted.

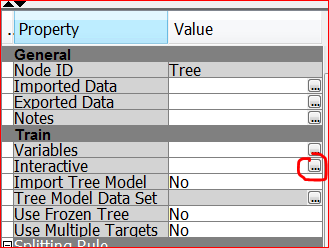
**Task:** Exploring results in the Decision Tree model

1. In Enterprise Miner, obtain your HMEQ process flow diagram. Run the first decision tree model, however the decision tree is very simplistic, so we will use the interactive decision tree tool to make a slightly more complex tree, this a manual tool for creating tree.

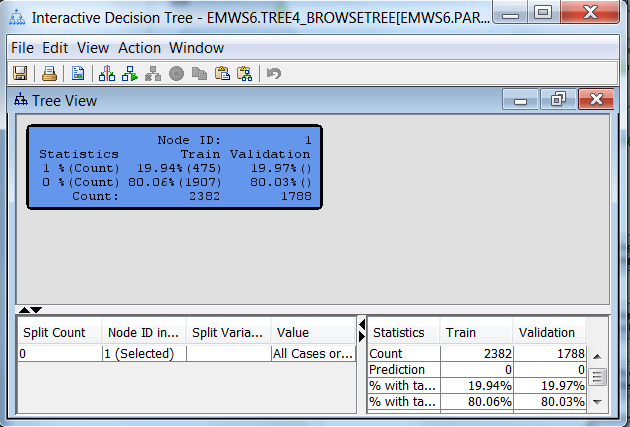


Add a new decision tree node to the workflow diagram from the model ribbon bar, second icon from the left. Rename it to decision tree manual.

Select the ellipse and a new gui component opens up that allows you to create a tree by hand.

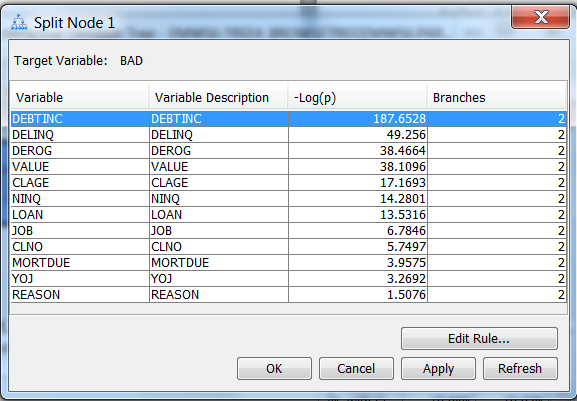


The following window opens.

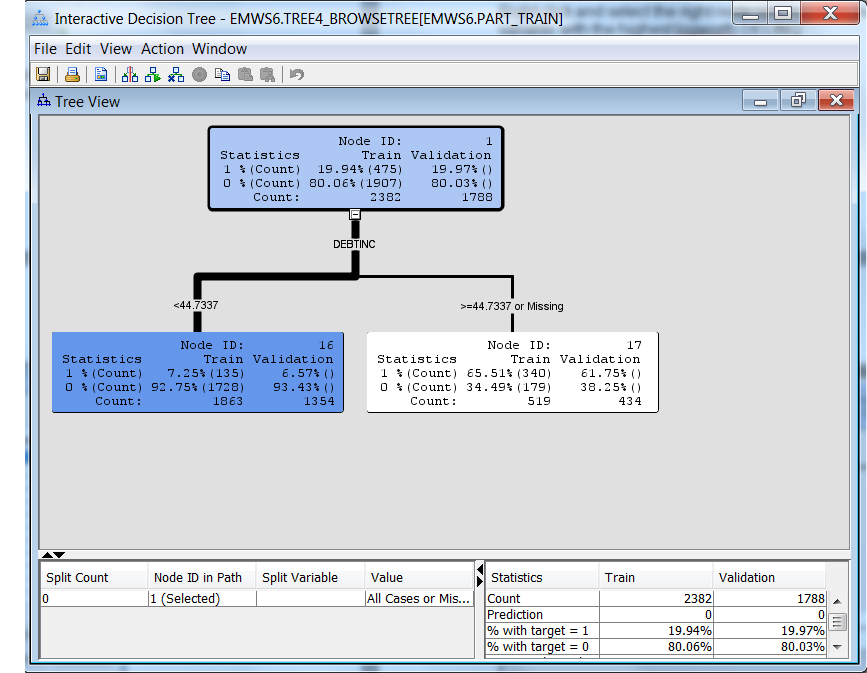


Note the blue box represents the root node, which contains all the data. The blue colour indicates a high leaf purity because 80% of the data are good payers!

Right click and select the right node split to create two more leaves, select the variable with the highest logworth DELINQ



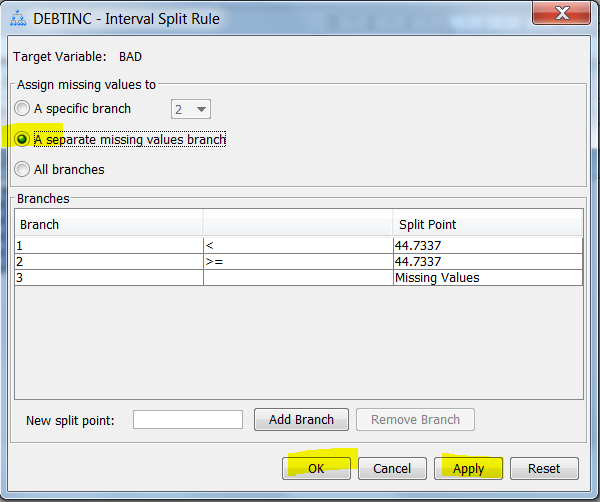
Logworth is a measure of leaf purity the higher the number the more effective the leaf split. Select OK

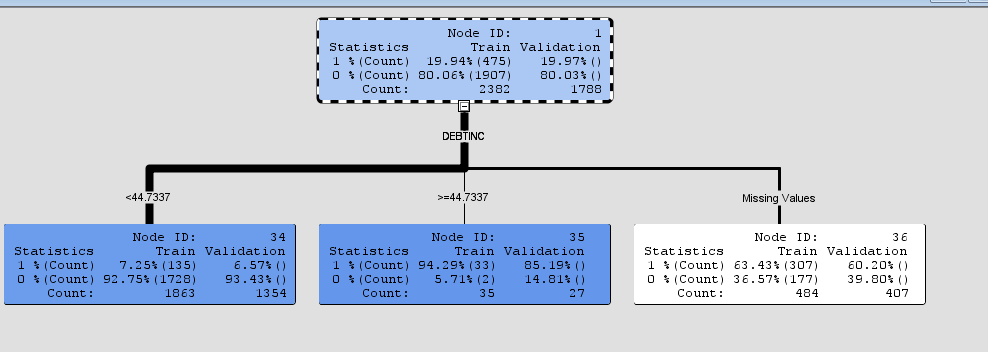


Notice this split does a very good job, of separating out good payers on the critical path.

We can repeat this exercise again with a slightly different setting.

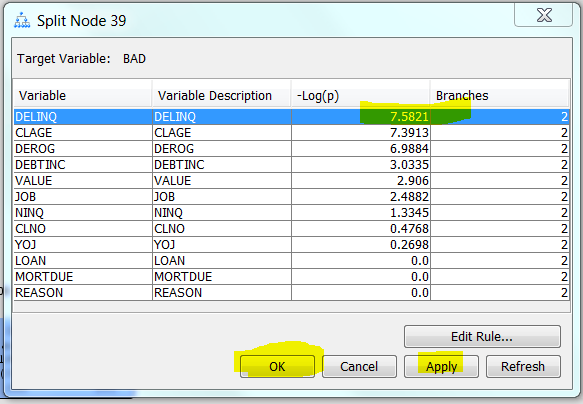
Right click on the rood node, and select prune node, this will get you back to the beginning again. This time split the node on DEBTINC, but also create a separate split for a class measurement level missing ! To do this you will have to edit the rule.

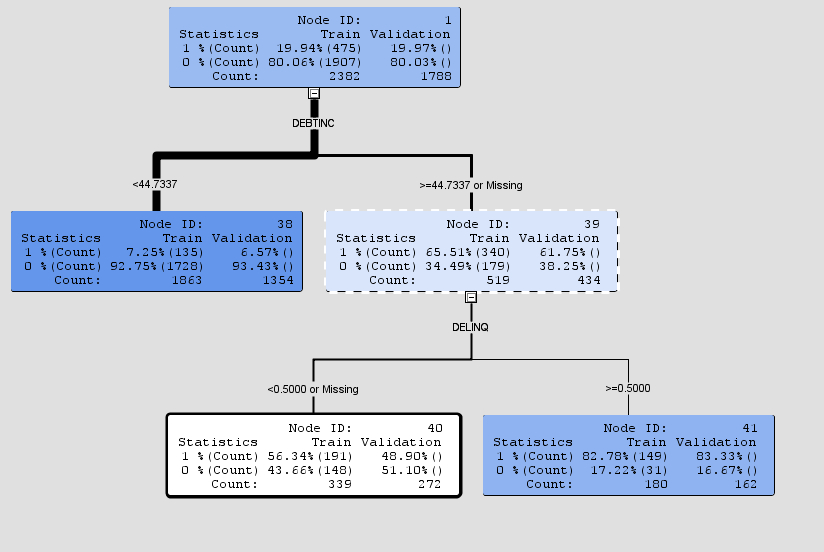




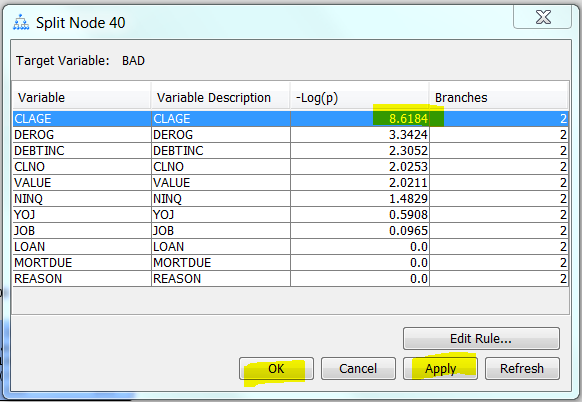
What do you notice about the distribution of target levels for the two new nodes !

When you have noted your observation, even though the leave purity is high there is few observations in the target path. Continue splitting the target path, the white node. This time split on the next highest logworth, DELINQ

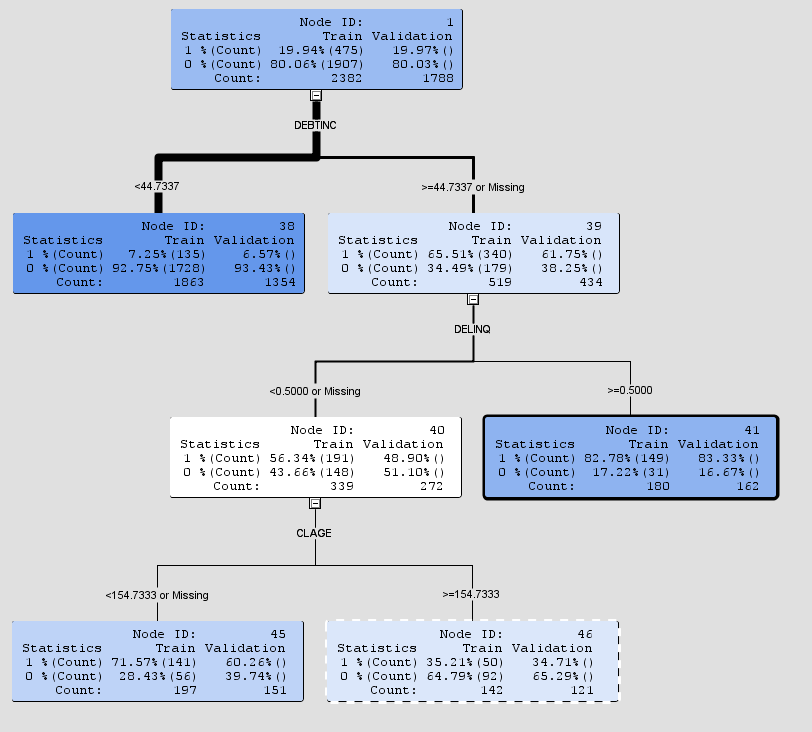




Note this split does a better job on the target path with approx. 83% of 180 observations for that leaf having a target value =1, bad payers. Note this is a big enrichment over the root node nearly 4:1 in the other direction. Note the other node is white so there is still work to do. Split node 40



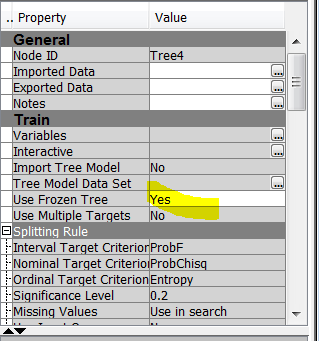
Select CLAGE which has a logworth of 8.6.



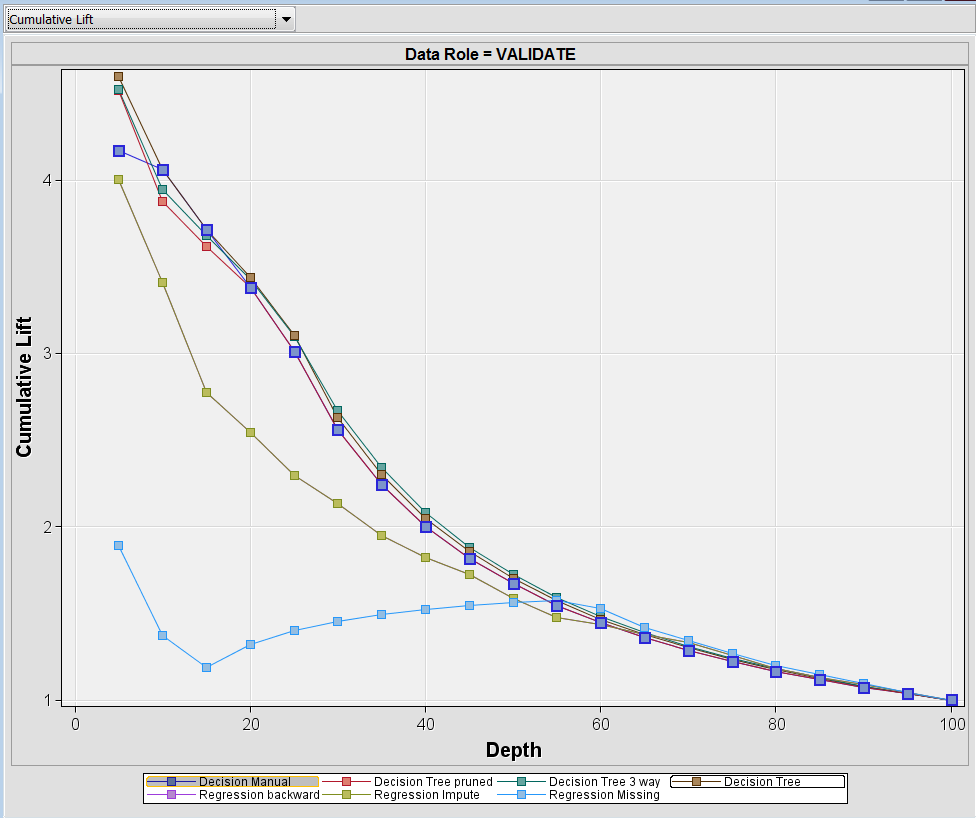
It is debateable whether splitting the remaining nodes improves the model. Feel free to explore. You can keeping splitting and pruning at will. You may recreate the original tree which was derived from 20 leaves ! When to stop is a matter of judgement. By producing a model by hand we have generated a much simpler, more parsimonious model 4 leaves.

If you examine the subtree assessment plot this might help you decide if the tree should be split further this is a matter of judgement. You will notice that the misclassification rate plateaus around 4/5 splits up to 10. Notice the simpler model’s performance is comparable with the trees that were developed by the automatic stopping rules.

Once you have decided on your final tree. Close the window then you need to freeze it. By selecting the appropriate property in the tree node. **If you don’t do this a tree is generated automatically each time you hit the run button and you loose your good work and will have to start again.**



Connect the manual tree up to the comparison node to generate a comparison lift chart.



Which tree is the best ?

Note that the manual tree is comparable with the pruned tree. The performance of a tree with four leaves is comparable to ten leaves. The same variables are important in defining the target path.

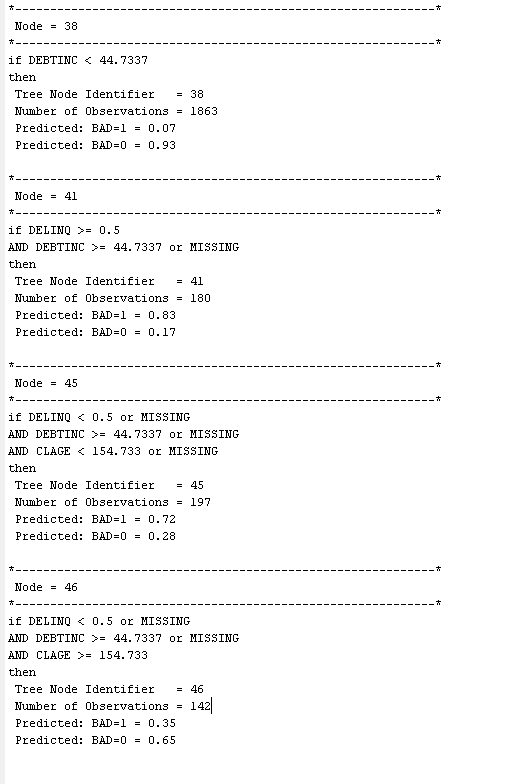
Examine the variable width plot and leaf statistics in the results of the manual decision tree node. Have another look at the subtree assessment plot for the default, the first decision tree you generated with 10 leaves.

From the menu item you can display the node rules for your decision tree.

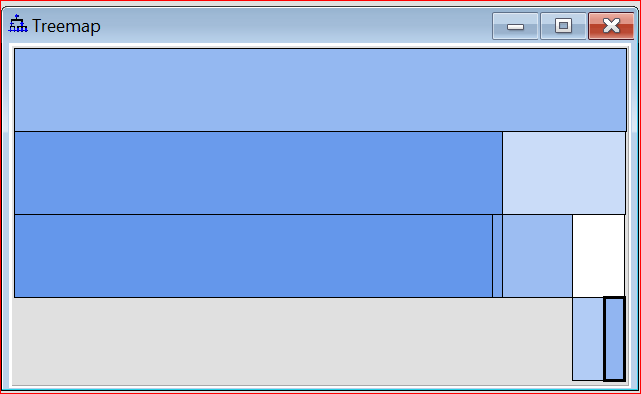
Identify the critical path, leaf with the most observations. Identify the target paths for this simple tree. Rules reproduced below

**Qu1: Interpret (i.e. draw out the main information from) this decision tree. Think about:**

1. **The critical path (where the majority of customers are).**
2. **Where the customers are who default on their repayments (i.e. the target event BAD=1 of the data mining task)**



1. View the treemap

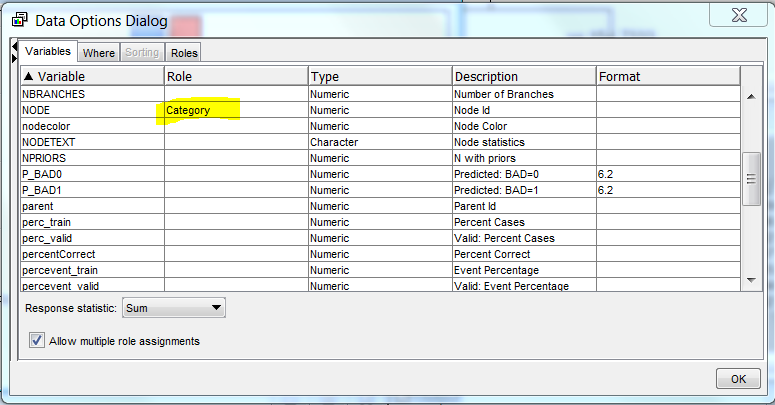


a)

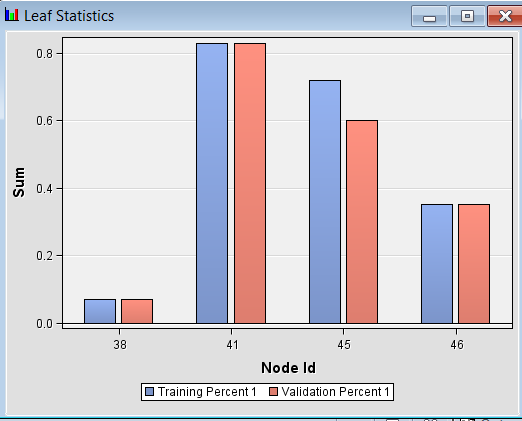
**Qu2: Interpret this tree map.**

**Examine the leaf statistics**

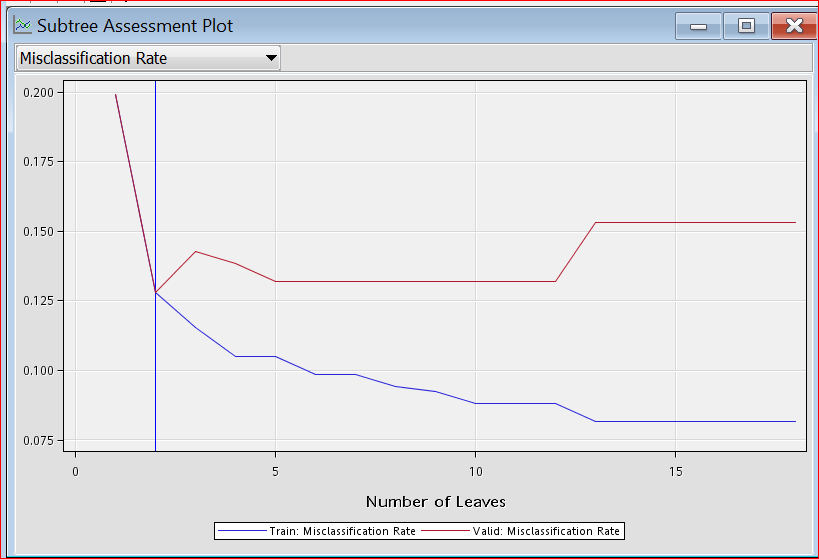
Right click and change the node to category



What can you say about the consistency of the classification between the training and validation results for each leaf?

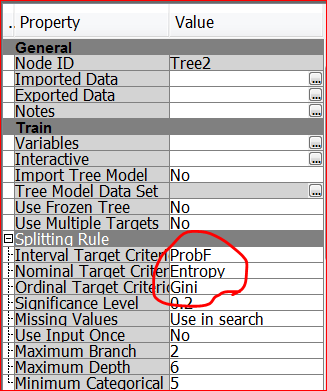


1. Close the window and return to the results of the decision node plot. View subtree assessment plot, select the misclassification rate drop down box.



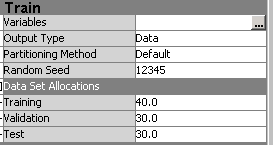
**Qu3: Interpret this plot.** (Note the Validation plot is above the Training plot).

**Extra, go back and change the Decision tree setting, for instance make the splitting rules all Gini, change the minimum number of observations in a leaf to 25, play around the maximum depth of tree from 4 to 20. How stable (consistency between validation and test set) are bushy trees. Interpret the new decision tree is it better than the Interactive tree?**



**Note if you wish to make adjustments to your workflow, create two decision tree nodes label one interactive the second default. For the interactive tree select frozen the tree to make it immune to changes. Assess the subtree plot, tree, diagnostic chart, and cumulative % response chart.**

**Finally go to the partition node and remove 12345 from random seed, run the tree several times ring the changes. If you see any differences what do you conclude about the sets of data the partition node creates. If you remove a number for the random seed property data is shuffled from a different starting point each time.**



**Hint: If you produce an unstable tree, poor correspondence between validation and training performance try a different random seed. Another approach is to increase the size of the Training set, or remove the validation set altogether.**

### Appendix

## Comparison with Other Tree Methods

### CHAID

The following discussion provides a brief description of the CHAID (chi-square automatic interaction detection) algorithm for building decision trees. This includes how the CHAID algorithm differs from the **Decision Tree** node and how it can be approximated.

For CHAID, the inputs are either nominal or ordinal. Many software applications accept interval inputs and automatically group the values into ranges before growing the tree.

The splitting criteria is based on p-values from the F-distribution (interval targets) or Chi-square distribution (nominal targets). The p-values are adjusted to accommodate multiple testing.

A missing value can be treated as a separate value. For nominal inputs, a missing value constitutes a new category. For ordinal inputs, a missing value is free of any order restrictions.

The search for a split on an input proceeds stepwise. Initially, a branch is allocated for each value of the input. Branches are alternately merged and re-split as is warranted by the p-values. The original CHAID algorithm by Kass stops when no merge or re-splitting operation creates an adequate p-value. The final split is adopted. A common alternative, sometimes called the exhaustive method, continues merging to a binary split. It then adopts the split with the most favorable p-value among all splits the algorithm considered.

After a split is adopted for an input, its p-value is adjusted. The input with the best adjusted p-value is selected as the splitting variable. If the adjusted p-value is smaller than a threshold that you specified, then the node is split.

Tree construction ends when all the adjusted p-values of the splitting variables in the unsplit nodes are above the user-specified threshold.

The CHAID algorithm differs from the **Decision Tree** node in a number of ways:

* The **Decision Tree** node seeks the split that minimizes the adjusted p-value. The original KASS algorithm does not.
* The CHAID exhaustive method is similar to the **Decision Tree** node heuristic method, except that the exhaustive method "re-splits" and the **Decision Tree** node "re-assigns."
* CHAID software discretizes interval inputs. The **Decision Tree** node sometimes consolidates observations into groups.
* The **Decision Tree** node searches on a within-node sample, unlike CHAID.

### How to Approximate the CHAID Method with the Decision Tree Node

To approximate CHAID, the interval inputs should first be discretized into a few dozen values.  
  
You should then set the following **Decision Tree** node properties:

* For nominal targets, do the following:
  + Set the **Nominal Criterion** property to **PROBCHISQ**.
* For interval targets, do the following:
  + Set the **Interval Criterion** property to **PROBF**.
* For either nominal or interval targets, do the following:
  + To avoid automatic pruning, set the **Method** property to **Largest**.
  + Set the Chi-square or F test **Significance Level** to 0.05 (or whatever significance level seems appropriate).
  + Set the **Maximum Branch** property to the maximum number of categorical values in an input.
  + Set the **Number of Surrogate Rules** property to 0.
  + To force a heuristic search, set the **Exhaustive** property to 0.
  + Set the **Leaf Size** to 1.
  + Set the **Split Size** property to 2.
  + Set the **Bonferroni Adjustment** property to **Yes**, and the **Time of Kass Adjustment** property to **After**.

The CHAID method does not apply to ordinal targets. The methodology for ordinal targets included in CHAID software is fundamentally different, and originated a decade after the CHAID work.