DECISION TREES

Methodology Training Document (Module 6)

YEAR 2015

DART

EXL
look deeper.

Objectives and Scope

Course Goals

- Introduction to usage of decision tree in line with EXL DA Methodology
- Explain key concepts of decision tree and provide an illustration
- Explain advantages and drawbacks of decision tree in general
- Provide helpful “tricks of the trade”

Beyond the Scope of this Training

- Comprehensive coaching on every feature of any specific decision tree tool like CART, SAS E-Miner
- Backend algorithms of decision tree tools

Self Study Goals

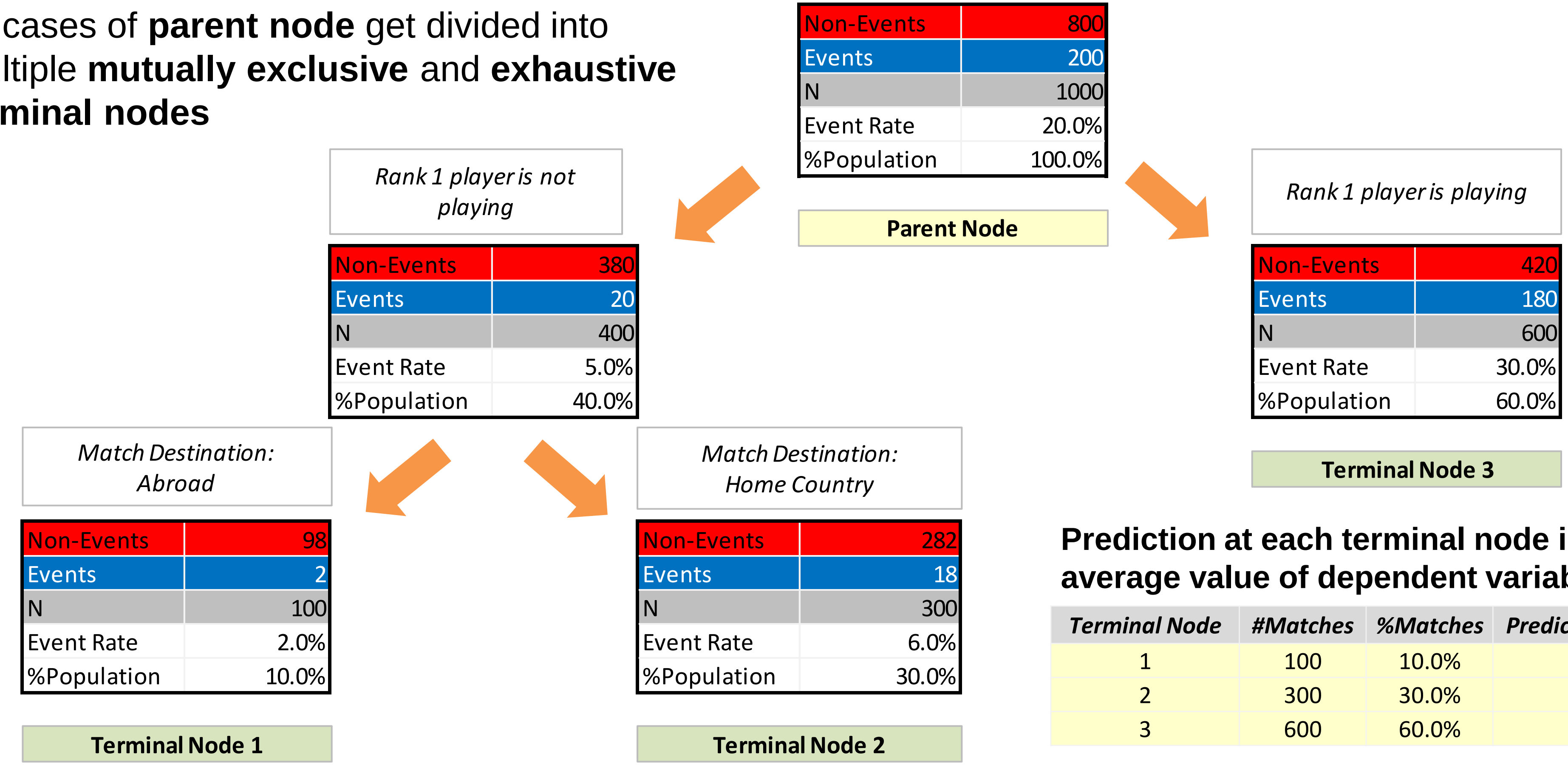
- Exploration of various features of CART by practising on some dummy data (Refer to CART overview in Appendix for self-study)
- Innovations and new features of CART as and when released by Salford Systems
- Discussion on advanced features can be taken up offline

1. Decision Tree: Illustration

Data : 1000 cricket matches

Event : Win

All cases of **parent node** get divided into multiple **mutually exclusive** and **exhaustive terminal nodes**



Parent Node

Non-Events	800
Events	200
N	1000
Event Rate	20.0%
%Population	100.0%

Rank 1 player is not playing

Non-Events	380
Events	20
N	400
Event Rate	5.0%
%Population	40.0%

Rank 1 player is playing

Terminal Node 3

Non-Events	420
Events	180
N	600
Event Rate	30.0%
%Population	60.0%

Match Destination: Abroad

Terminal Node 1

Non-Events	98
Events	2
N	100
Event Rate	2.0%
%Population	10.0%

Match Destination: Home Country

Terminal Node 2

Non-Events	282
Events	18
N	300
Event Rate	6.0%
%Population	30.0%

Prediction at each terminal node is average value of dependent variable

Terminal Node	*#Matches*	*%Matches*	*Prediction*
1	100	10.0%	0.02
2	300	30.0%	0.06
3	600	60.0%	0.30

2. Decision Tree: Key Concepts

- Classification Tree: Target variable is categorical. For example –

Binary	: IND_DEFAULT	– Takes value 1 for payment default, else takes value 0
Ordered Multinomial	: INCOME_GROUP	– Takes three values: “High”, “Medium” and “Low”
Unordered Multinomial	: FAVORITE_COLOR	– Take three values: “Red”, “Blue” and “Green”

- Regression Tree: Target variable is continuous. For example –

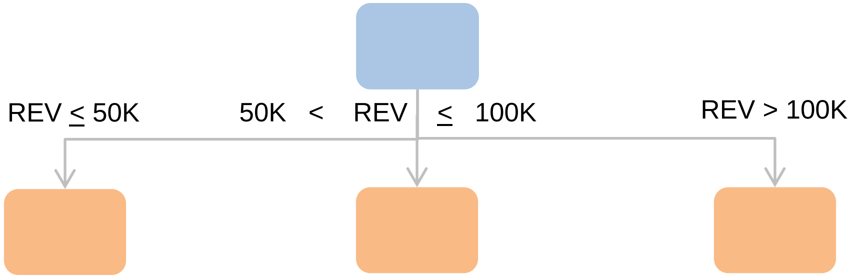
Count	: ATTENDANCE	– #Days of presence in school in month of Jul 2015
Percentage	: PCT_CHNG_MARKS	– %Change in marks in Statistics over last semester
Non-negative & continuous	: REVENUE	– Revenue (in USD) of a US-based electronics store

- Minimum #Cases in Terminal Node: Without any restriction, a decision tree can continue having numerous levels of splits until there is a single record in the terminal node. However, it is important to have significant sizing in terminal nodes because of following reasons:
  - Validation data may not have exact combination of attributes for replicating terminal node generated on training data
  - It leads to overfitting problem - With increase in the depth of the tree, the algorithm becomes more specific to the training data, i.e. It performs really well in the specified scenarios in the training data but not otherwise

  As a thumb rule, terminal node should have at least 1% of total records. If total population size is low, terminal node minimum sizing can be set to even 5%

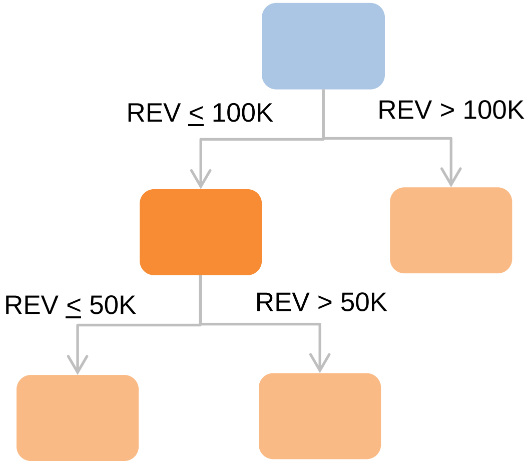
Decision Tree: Key Concepts (Contd.)

- #Branches at every split: A decision tree tool like SAS Enterprise Miner allows for split into more than 2 branches. For Example:
  - Splitter – Revenue: Splits into three branches
    - Revenue ≤ 50K;
    - 50K < Revenue ≤ 100K; and
    - Revenue > 100K.



On the other hand, CART (a decision tree tool by Salford Systems) allows for split into only 2 branches at every level. However, splitter variable can be repeated at next level. Hence, effectively multiple branches are possible.

- Splitter 1 – Revenue: Splits into two branches
  - Revenue ≤ 100K (Node 1); and
  - Revenue > 100K (Node 2)
- Splitter 2 – Revenue: Node 1 can be further split into two branches
  - Revenue ≤ 50K (Node 1 A); and
  - Revenue > 50K (Node 1 B)



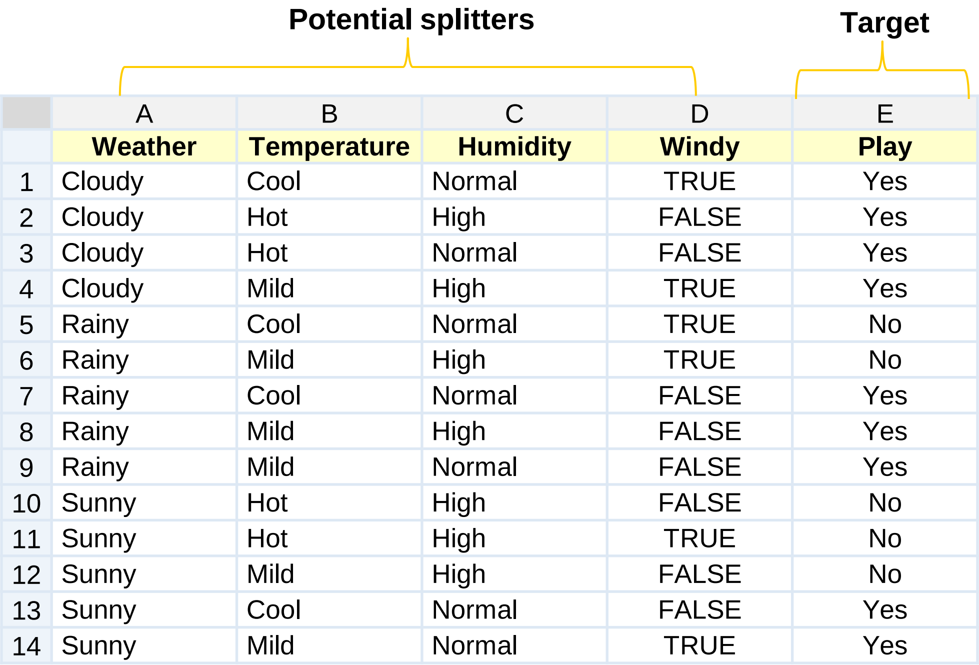
- Entropy: A measure of uncertainty i.e. the amount of unpredictability in a sequence of events

3. Choice of Splitter: An Illustration

There are several Decision tree algorithms*:

- ID3 (Iterative Dichotomiser 3)
- C4.5
- CART (Classification and Regression Tree)
- CHAID (CHi-Squared Automatic Interaction Detector)

Let us consider **a classic example** to understand how decision tree can choose a splitter based on concepts of **entropy** and **information gain**.



Potential splitters

Target

	A	B	C	D	E
	Weather	**Temperature**	**Humidity**	**Windy**	**Play**
1	Cloudy	Cool	Normal	TRUE	Yes
2	Cloudy	Hot	High	FALSE	Yes
3	Cloudy	Hot	Normal	FALSE	Yes
4	Cloudy	Mild	High	TRUE	Yes
5	Rainy	Cool	Normal	TRUE	No
6	Rainy	Mild	High	TRUE	No
7	Rainy	Cool	Normal	FALSE	Yes
8	Rainy	Mild	High	FALSE	Yes
9	Rainy	Mild	Normal	FALSE	Yes
10	Sunny	Hot	High	FALSE	No
11	Sunny	Hot	High	TRUE	No
12	Sunny	Mild	High	FALSE	No
13	Sunny	Cool	Normal	FALSE	Yes
14	Sunny	Mild	Normal	TRUE	Yes

Of 14 records, target variable “Play” takes value “Yes” for 9 cases and value “No” for 5 cases.

$$\text{Initial Entropy} = -\left(\left(\frac{9}{14}\right)\log_2\left(\frac{9}{14}\right)+\left(\frac{5}{14}\right)\log_2\left(\frac{5}{14}\right)\right)$$

$$= \mathbf{0.94}$$

Note: Discussion on algorithms is beyond the scope of this training. However, an illustration based on ID3 algorithm is covered for basic understanding.

Choice of Splitter: An Illustration (Contd.)

Case 1: Variable “Weather” is chosen as a splitter

- CLOUDY : 4 records; 4 Yes => $-\left(\left(\frac{4}{4}\right)\log_2\left(\frac{4}{4}\right)+\left(\frac{0}{4}\right)\log_2\left(\frac{0}{4}\right)\right)$ = 0.00
- RAINY : 5 records; 3 Yes => $-\left(\left(\frac{3}{5}\right)\log_2\left(\frac{3}{5}\right)+\left(\frac{2}{5}\right)\log_2\left(\frac{2}{5}\right)\right)$ = 0.97
- SUNNY : 5 records; 2 Yes => $-\left(\left(\frac{2}{5}\right)\log_2\left(\frac{2}{5}\right)+\left(\frac{3}{5}\right)\log_2\left(\frac{3}{5}\right)\right)$ = 0.97

Expected New Entropy =$\left(\frac{4}{14}\right)0.00+\left(\frac{5}{14}\right)0.97+\left(\frac{5}{14}\right)0.97$ = **0.69**

Information Gain = 0.94 – 0.69 = **0.25**

Case 2: Variable “Temperature” is chosen as a splitter

- COOL : 4 records; 3 Yes => $-\left(\left(\frac{3}{4}\right)\log_2\left(\frac{3}{4}\right)+\left(\frac{1}{4}\right)\log_2\left(\frac{1}{4}\right)\right)$ = 0.81
- HOT : 4 records; 2 Yes => $-\left(\left(\frac{2}{4}\right)\log_2\left(\frac{2}{4}\right)+\left(\frac{2}{4}\right)\log_2\left(\frac{2}{4}\right)\right)$ = 1.00
- MILD : 6 records; 4 Yes => $-\left(\left(\frac{4}{6}\right)\log_2\left(\frac{4}{6}\right)+\left(\frac{2}{6}\right)\log_2\left(\frac{2}{6}\right)\right)$ = 0.92

Expected New Entropy =$\left(\frac{4}{14}\right)0.81+\left(\frac{4}{14}\right)1.00+\left(\frac{6}{14}\right)0.92$ = **0.91**

Information Gain = 0.94 – 0.91 = **0.03**

Choice of Splitter: An Illustration (Contd.)

Case 3: Variable “Humidity” is chosen as a splitter

- NORMAL : 7 records; 6 Yes => $-\left(\left(\frac{6}{7}\right)\log_2\left(\frac{6}{7}\right)+\left(\frac{1}{7}\right)\log_2\left(\frac{1}{7}\right)\right)$ = 0.59
- HIGH : 7 records; 3 Yes => $-\left(\left(\frac{3}{7}\right)\log_2\left(\frac{3}{7}\right)+\left(\frac{4}{7}\right)\log_2\left(\frac{4}{7}\right)\right)$ = 0.99

Expected New Entropy =$\left(\frac{7}{14}\right)0.59+\left(\frac{7}{14}\right)0.99$ = **0.79**

Information Gain = 0.94 – 0.79 = **0.15**

Case 4: Variable “Windy” is chosen as a splitter

- TRUE : 6 records; 3 Yes => $-\left(\left(\frac{3}{6}\right)\log_2\left(\frac{3}{6}\right)+\left(\frac{3}{6}\right)\log_2\left(\frac{3}{6}\right)\right)$ = 1.00
- FALSE : 8 records; 6 Yes => $-\left(\left(\frac{6}{8}\right)\log_2\left(\frac{6}{8}\right)+\left(\frac{2}{8}\right)\log_2\left(\frac{2}{8}\right)\right)$ = 0.81

Expected New Entropy =$\left(\frac{6}{14}\right)1.00+\left(\frac{8}{14}\right)0.81$ = **0.89**

Information Gain = 0.94 – 0.89 = **0.05**

Since information gain is maximum for variable “Weather”, a decision tree based on ID3 algorithm will use it as the first splitter.

4. Decision Tree: Advantages and Drawbacks

Advantages

- Can be used for **creating a new variable** for a segment that has mean target value significantly different from overall mean target value. Such a node variable can be used as input in other modeling techniques
- Provides decision rules (splitters with cut-offs). This helps in **key driver analysis**
- Can be used to **club categories** of similar event rate together for a categorical independent variable
- Can be used for **binning** for a continuous independent variable
- Can be used for **taking segmentation decision** (i.e. identification of segments for building separate models)
- Provides **variable importance list** that can be used for variable selection
- Can be used to do **missing value imputation** (refer to Data Prep module for details)
- Can be used for **prediction** (though there is a disadvantage)

Drawbacks

- In a decision tree model, predicted value for each node is the mean target value of that node. Records in a particular node can't be rank-ordered !
- Since the number of unique values of prediction is very low, metrics like lift are not as meaningful as they are in a regression modeling techniques

Appendix: For Self-Study

I. Introduction to CART
II. CART Basics
III. Setting Up CART Model
IV. CART Output

I. Introduction to CART

CLASSIFICATION AND REGRESSION TREES (CART)

- CART is a robust decision-tree tool for data mining, predictive modeling, and data preprocessing.
- CART automatically searches for important patterns and relationships, uncovering hidden structure even in highly complex data.
- CART trees can be used to generate accurate and reliable predictive models for a broad range of applications from bioinformatics to risk management.
- The most common applications include churn prediction, credit scoring, drug discovery, fraud detection, manufacturing quality control, and wildlife research.

Note: Several hundred detailed applications studies are available at http://www.salford-systems.com

II. CART Basics

Points to Remember:

- Maximum number of cells (rows x columns) allowed in the analysis is limited by license
- CART is case insensitive for variable names; all reports show variables in upper case
- CART supports both character and numeric variable values
- Variable names must not exceed 32 characters
- Variable names must have only letters, numbers or underscores
- Variable names must start with a letter
- When a variable name ends with "$" and/or if the data value is surrounded by quotes on the first record, it is processed as a character variable. In this case, a "$" sign is added to the variable name if needed.

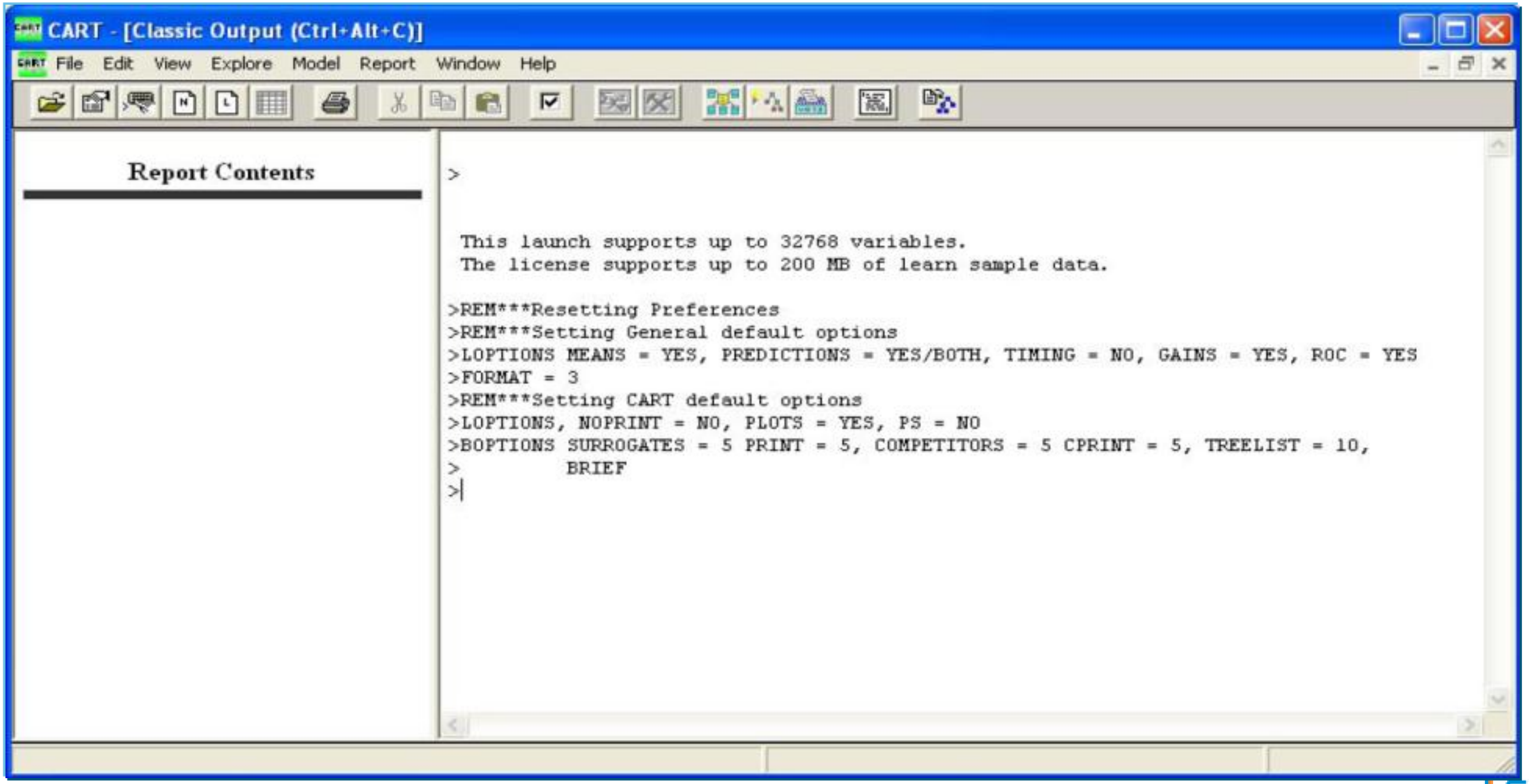
Examples on Acceptable/Unacceptable Variable Names:

VARIABLE NAME	TYPE	COMMENT
AGE	Numeric	OK
AGE_1	Numeric	OK
1PAYMENT	Numeric	Unacceptable; leading character other than letter.
%WEIGHT	Numeric	Unacceptable; leading character other than letter.
SOCIAL_SECURITY_NUMBER_AND_ACCOUNT	Numeric	Unacceptable; too long. Variable name will be truncated to 32 characters.
DEPOSIT&AMOUNT	Numeric	Unacceptable; "&" is not letter, number or underscore. This character will be replaced with an underscore.
NAME	Character	OK. But being a character variable, "$" will automatically be added at the end. This variable name would be displayed as "NAME$".

II. CART Basics

CART Desktop:

This is the first screen that pops up as we double-click on CART program icon.



```
>


 This launch supports up to 32768 variables.
 The license supports up to 200 MB of learn sample data.

>REM***Resetting Preferences
>REM***Setting General default options
>LOPTIONS MEANS = YES, PREDICTIONS = YES/BOTH, TIMING = NO, GAINS = YES, ROC = YES
>FORMAT = 3
>REM***Setting CART default options
>LOPTIONS, NOPRINT = NO, PLOTS = YES, PS = NO
>BOPTIONS SURROGATES = 5 PRINT = 5, COMPETITORS = 5 CPRINT = 5, TREELIST = 10,
>        BRIEF
>|
```

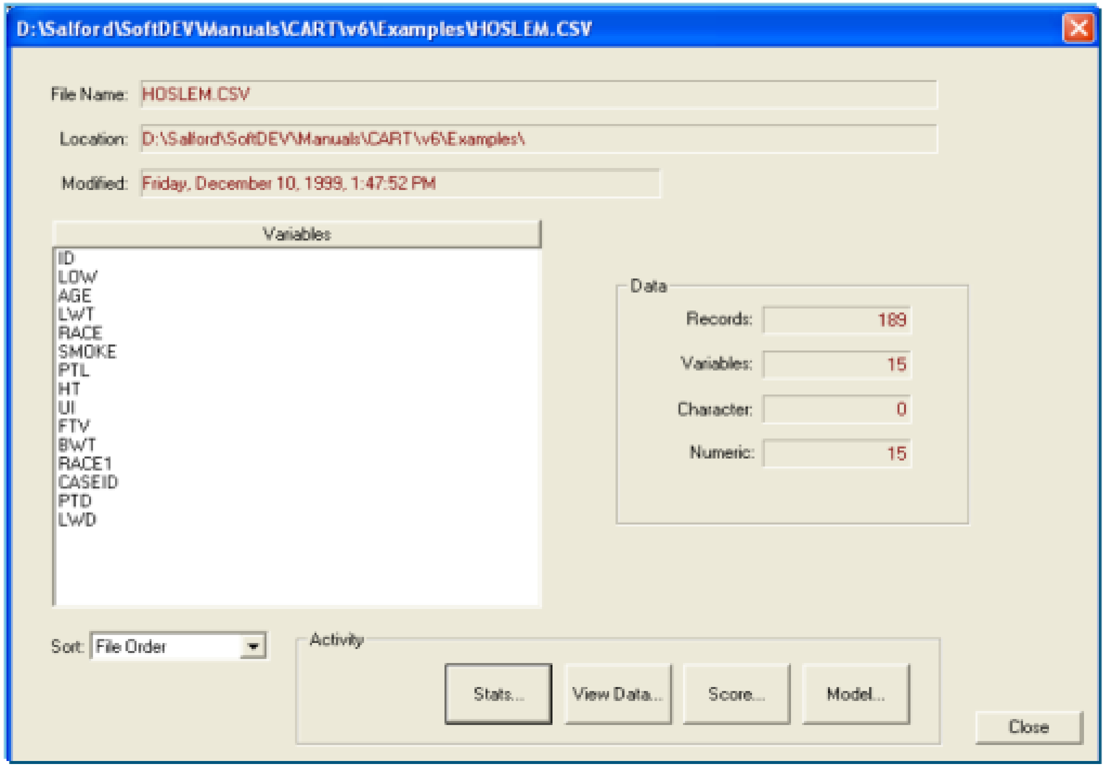
II. CART Basics

Opening a File:

- Select Open → Data File … from the **File** menu (or click on the 📂 toolbar icon)
- Navigate to the file location and select the file to open.

Note: As an alternative to navigation, default input and output directories can be reset; select Options … from the **Edit** menu and select the **Directories** tab

Activity Dialog Box:



D:\Salford\SoftDEV\Manuals\CART\v6\Examples\HOSLEM.CSV

File Name: HOSLEM.CSV

Location: D:\Salford\SoftDEV\Manuals\CART\v6\Examples\

Modified: Friday, December 10, 1999, 1:47:52 PM

Variables

ID
LOW
AGE
LWT
RACE
SMOKE
PTL
HT
UI
FTV
BWT
RACE1
CASEID
PTD
LWD

Data

Records:	189
Variables:	15
Character:	0
Numeric:	15

Sort: File Order

Example here imports a CSV file. The data file, however, could be a SAS dataset.

The file contains 189 records and 15 variables and all are numeric variables.

Using Sort drop-down control, variables can be sorted in either Alphabetical or File Order.

III. Setting Up CART Model

THE MODEL TAB

- Tab heading is displayed in **RED** if tab *requires information from user* before a model can be built.
- **Target Variable Selection**
  - The moment a **TARGET** variable is selected, CART knows which one of all the variables is to be analyzed or predicted. This is *the only required step* in setting up a model. Everything else is optional.
- **Tree Type**
  - **Classification Tree**: It uses a "*categorical*" target variable (e.g.; YES/NO). The purpose of classification is to accurately discriminate between classes.
  - **Regression Tree**: it uses a "*continuous*" variable (such as AGE or INCOME). The purpose of regression is to predict values that are close to a true outcome.
- **Predictor Variable Selection**
  - Candidate predictor (independent) variables are specified by check marks in the Predictor column
- **Case Weights**
  - To select a variable as the case weight, simply put a checkmark against that variable in the **Weight** column.
  - An observation's case weight can be thought of as a repetition factor. It may take on fractional values or whole numbers.
  - A missing, negative or zero case weight causes the observation to be deleted, just as if the target variable were missing.

III. Setting Up CART Model

THE MODEL TAB

- **Auxiliary Variables**
  - Auxiliary variables are variables that are tracked throughout the CART tree but are not necessarily used as predictors.
  - By marking a variable as Auxiliary, it is indicated that basic summary statistics for such a variable may be retrieved later for any node in the CART tree.
- **Setting Focus Class**
  - In classification runs, some of the reports generated by CART (gains, prediction success, color coding etc.) have one target class in focus.

By default, CART will put the first class it finds in the dataset *in focus*.

The binary categorical variable "LOW" (coded 0/1) is the target (or dependent) variable.

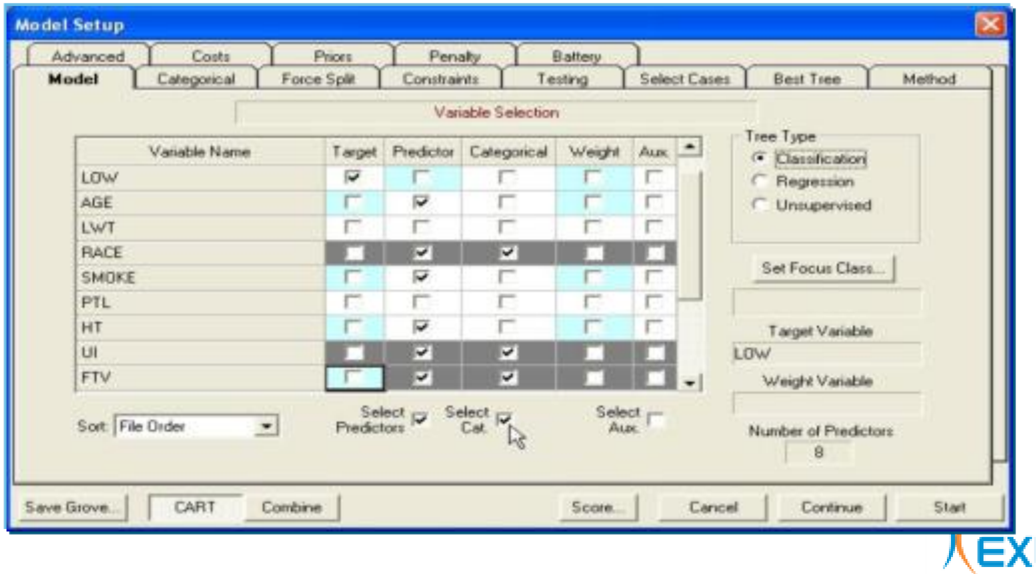
For categorical dependent variable, Tree Type ought to be selected as "Classification".

8 variables have been selected as predictor variables.

Using Sort drop-down control, variables can be sorted in either Alphabetical or File Order.

***Note**: Each of the Model Setup tabs contains a [**Save Grove...**] button in the lower left corner. It helps saving the model for future review, scoring or export.*

A Snapshot of Model Tab



Variable Selection

Variable Name	Target	Predictor	Categorical	Weight	Aux
LOW	☑	□	□	□	□
AGE	□	☑	□	□	□
LWT	□	□	□	□	□
RACE	□	☑	☑	□	□
SMOKE	□	☑	□	□	□
PTL	□	□	□	□	□
HT	□	☑	□	□	□
UI	□	☑	☑	□	□
FTV	□	☑	☑	□	□

Sort: File Order

Select Predictors ☑ Select Cat. ☑ Select Aux. □

Tree Type
● Classification
○ Regression
○ Unsupervised

Set Focus Class...

Target Variable: LOW

Weight Variable:

Number of Predictors: 8

III. Setting Up CART Model

THE CATEGORICAL TAB

- Setting Class Names
  - Select variable. Press [**Set Class Names**] option to enter/edit class names for the variable.
- High Level Categorical (HLC) Predictors
  - Such variables present computational challenge because of exploding number of possible ways to split the data in a node.

#Levels	#Distinct Splits
K	2^{K-1} - 1
2	1
3	3
4	7
. . .	. . .
10	511
. . .	. . .
21	Over a million !!
. . .	. . .

For binary target variable, CART has special shortcuts for HLC predictors that always work. HLC settings, thus, are relevant only if target variable has more than 2 levels.

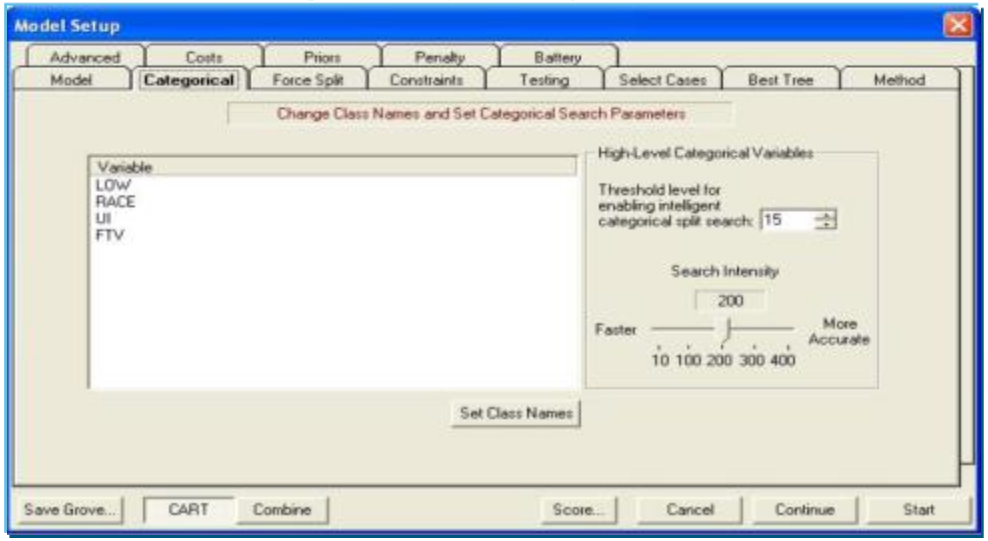
The threshold level of 15 indicates that

▪*for categorical predictors with 15 or fewer levels, CART would search all possible splits and definitely find the overall best partition*

▪*for predictors with more than 15 levels, intelligent shortcuts would be used to do fast "local" searches for very good partitions (though these may not be the absolute overall best).*

As the short-cut method explore only a limited range of possible splits, search intensity can be set within a range of 0-400. The default setting is ***200****.*

A Snapshot of Categorical Tab



Change Class Names and Set Categorical Search Parameters

Variable
LOW
RACE
UI
FTV

High-Level Categorical Variables

Threshold level for enabling intelligent categorical split search: 15

Search Intensity: 200

Searching too aggressively for the best HLC splitter increases the likelihood of over-fitting the model to the training data.

III. Setting Up CART Model

THE TESTING TAB

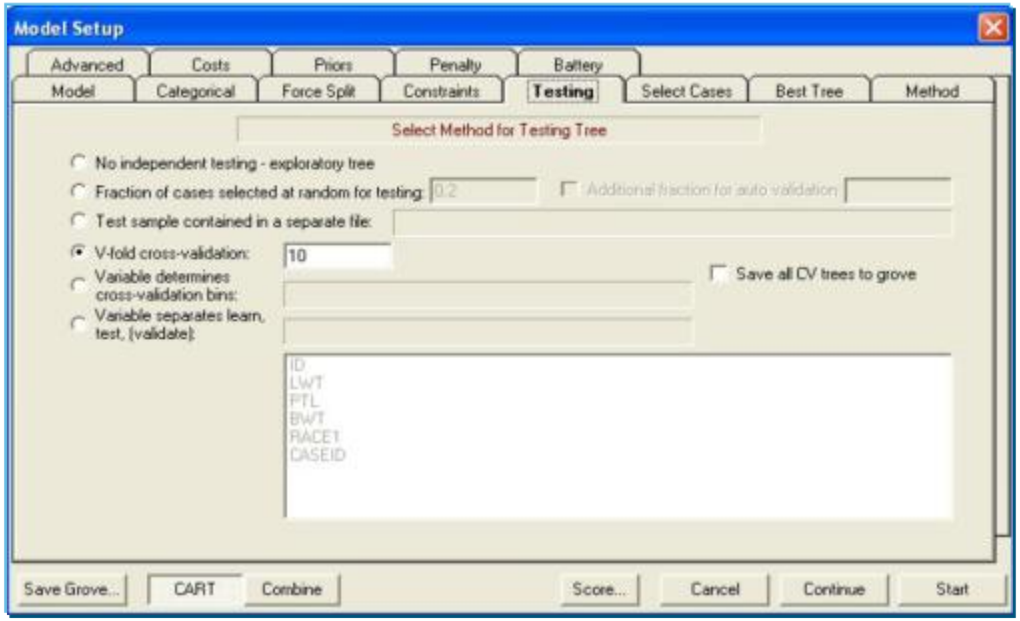
- **No independent testing - exploratory tree**
  - This option *skips the entire testing phase* and simply reports the largest tree grown.
  - Because no test method is specified, CART does not select an “optimal” tree.
  - Bypassing the test phase can be useful when CART is being used to generate a quick cross tabulation of the target against one of the predictors. It is also useful for “supervised binning” or aggregation of variables such as high-level categorical candidates.
- **Fraction of cases selected at random for testing**
  - Use this option to let CART *automatically separate a specified percentage of data for test purposes*. Because no optimal fraction is best for all situations, user may want to experiment.
  - This mechanism does not provide user with a way of tagging the records used for testing.
- **Variable separate learn, test, (validate)**
  - A variable on the data set can be used to flag which records are to be used for learning (training) and which are to be used for testing or validation.
  - Use a *binary (0/1) numeric variable* to define simple *learn/test partitions*.
- **Test sample contained in a separate file**
  - *Two separate files* are assumed — one for *learning* and one for *testing*. The files can be in different database formats and their columns do not need to be in the same order.
  - The train and test files must both contain ALL variables to be used in the modeling process.

III. Setting Up CART Model

THE TESTING TAB

- V-fold cross-validation
  - Cross validation is a ***marvelous way to make the maximal use of training data***, although it is typically used when data sets are small.
  - Even if dataset is large, in case of low event rate, cross-validation may be the only viable testing method.
  - Cross validation allows user to build tree using all the data. The testing phase requires running an additional '**V**' trees (in V-fold CV), each of which is tested on a different **V**% of the data. The results from those '**V**' test runs are combined to create a table of synthesized test results.

A Snapshot of Testing Tab



Select Method for Testing Tree

○ No independent testing - exploratory tree

○ Fraction of cases selected at random for testing: 0.2 □ Additional fraction for auto validation:

○ Test sample contained in a separate file:

● V-fold cross-validation: 10 □ Save all CV trees to grove

○ Variable determines cross-validation bins:

○ Variable separates learn, test, (validate):

Default test setting: *10-fold cross validation*

Note: *Every target class must have at least as many records as the number of folds in the cross validation. Otherwise, the process breaks down, an error message is reported, and a "No Tree Built" situation occurs. This means that if data set contains only nine YES records in a YES/NO problem, CART cannot run more than nine-fold cross validation.*

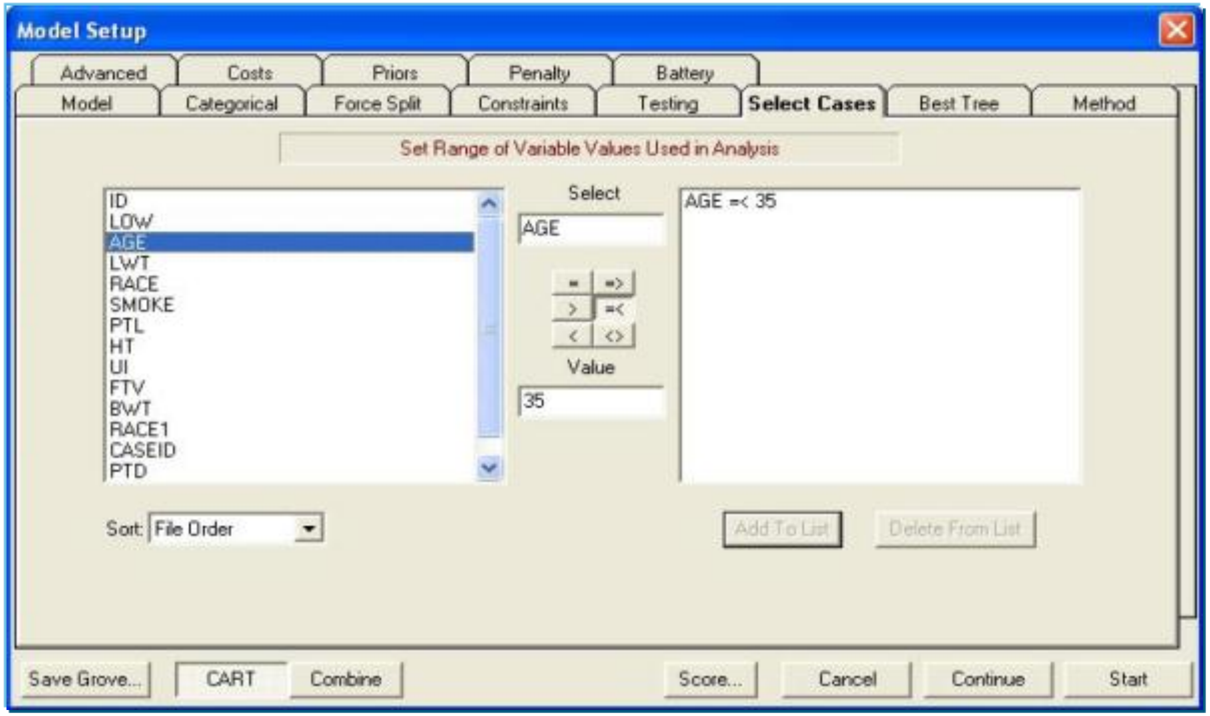
III. Setting Up CART Model

THE SELECT CASES TAB

- **The Model Setup – Select Cases tab** allows the user to specify **up to ten selection criteria** for building a tree based on a subset of cases.

A selection criterion can be specified in terms of ***any variable appearing in the data set***, whether or not that variable is involved in the model.

A Snapshot of Select Cases Tab



Select: AGE

Value: 35

AGE =< 35

Sort: File Order

Steps

1. Double-click a variable in the variable list to add that variable to the **Select** text box.
2. Select one of the predefined logical relations by clicking its radio button.
3. Enter a numerical value in the **Value** text box.
4. Click **[Add to List]** to add the constructed criterion to the right window **and** use **[Delete from List]** to remove.

III. Setting Up CART Model

THE BEST TREE TAB

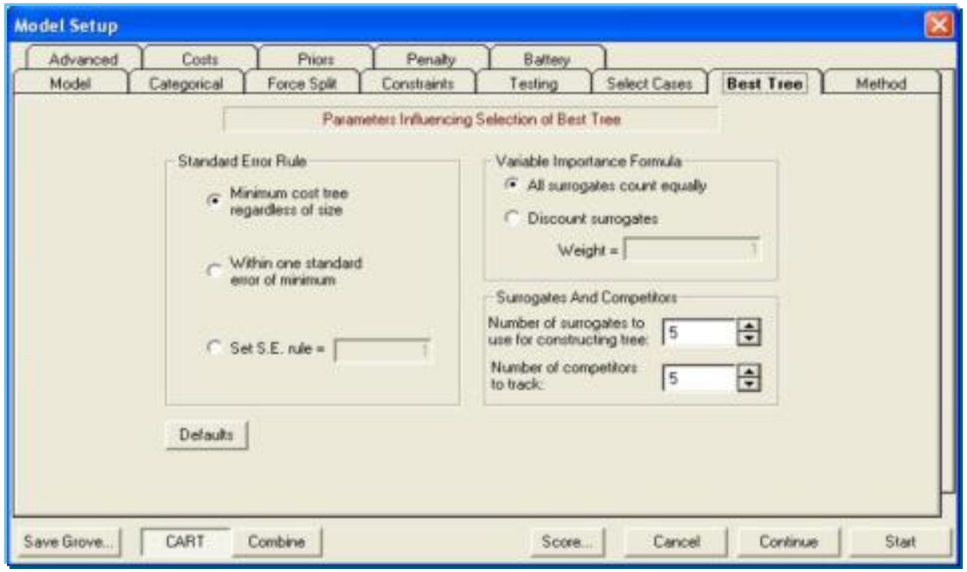
- Default Best Tree settings:
  - *Minimum cost tree regardless of size* (most accurate tree, given the specified testing method)

    Note: Alternatively, the user may wish to trade a more accurate tree for a smaller tree by selecting the smallest tree within one standard error of the minimum cost tree or by setting the standard error parameter equal to any nonnegative value.
  - *Five surrogates* used to construct tree

    Note: Alternatively, the user can increase or decrease the number of surrogates that CART searches for.
  - *All surrogates count equally* to compute variable importance

    Note: Alternatively, the user can fine-tune the variable importance calculation by specifying a weight to be used to discount the surrogates. Click on the Discount surrogates radio button and enter a value between 0 and 1 in the Weight text box.

A Snapshot of Best Tree Tab



Parameters Influencing Selection of Best Tree

Standard Error Rule
● Minimum cost tree regardless of size
○ Within one standard error of minimum
○ Set S.E. rule = 1

Variable Importance Formula
● All surrogates count equally
○ Discount surrogates
Weight = 1

Surrogates And Competitors
Number of surrogates to use for constructing tree: 5
Number of competitors to track: 5

Note: Surrogates refer to *splitters that are similar to the primary splitter and can be used when the primary split variable is missing.*

The Model Setup—Best Tree tab is largely of ***historical interest*** as it dates to a time when CART would produce a single tree in any run.

In today's CART, user has full access to every tree in the pruned tree sequence and can readily select trees of a size different than considered optimal.

III. Setting Up CART Model

THE METHOD TAB

- **Classification Tree Splitting Rules:**
  - ***Gini:*** This default rule ***often works well across a broad range of problems***. Gini has a tendency to generate trees that include some rather small nodes highly concentrated with the class of interest.
  - ***Symmetric Gini:*** This is a special variant of the Gini rule designed specifically to work with a cost matrix. If different costs for different classification errors are not specified, the Gini and the Symmetric Gini are identical.
  - ***Entropy:*** This tends to produce even smaller terminal nodes ("end cut splits") and is usually *less accurate than Gini.*
  - ***Class Probability:*** Probability trees tend to be larger than Gini trees and the predictions made in individual terminal nodes tend to be less reliable, but the details of the data structure that they reveal can be very valuable. When the user is primarily interested in *performance of top few nodes of a tree*, *probability trees are more useful*.
  - ***Twoing:*** The major difference between the Twoing and other splitting rules is that *Twoing tends to produce more balanced splits (in size)*. Twoing has a built-in penalty that makes it avoid unequal splits. A Gini or Entropy tree could easily produce 90/10 splits whereas Twoing will tend to produce 50/50 splits. The differences between the Twoing and other rules become more evident in case of multi-class target variable.
  - ***Ordered Twoing:*** The Ordered Twoing rule is *useful when target levels are ordered classes*. Ordered Twoing can be thought of as developing a model that is somewhere between a classification and a regression. Remember that the other splitting rules would not care at all which levels were grouped together because they ignore the numeric significance of the class label.

III. Setting Up CART Model

THE METHOD TAB

- **Regression Tree Splitting Methods:**
  - *Least Squares:* The objective function is to minimize sum of square of errors.
  - *Least Absolute Deviation:* The objective function is to minimize sum of absolute errors.

  In general, Least Squares Method is preferred, as the error magnitude itself is assigned as 'weight' to the corresponding error term.

- **Favor Even Splits:**
  - By default, the setting is 0, which indicates no bias in favor of even or uneven splits.

  On binary targets when both "Favor even splits" and unit cost matrix is set to 0, Gini, Symmetric Gini, Twoing, and Ordered Twoing will produce near identical results.

- **Linear Combination Splits:**
  - To deal more effectively with linear structure, CART has an option that allows node splits to be made on linear combinations of non-categorical variables. This option is implemented by clicking on the **Use Linear Combinations for Splitting** check box on the Method tab.

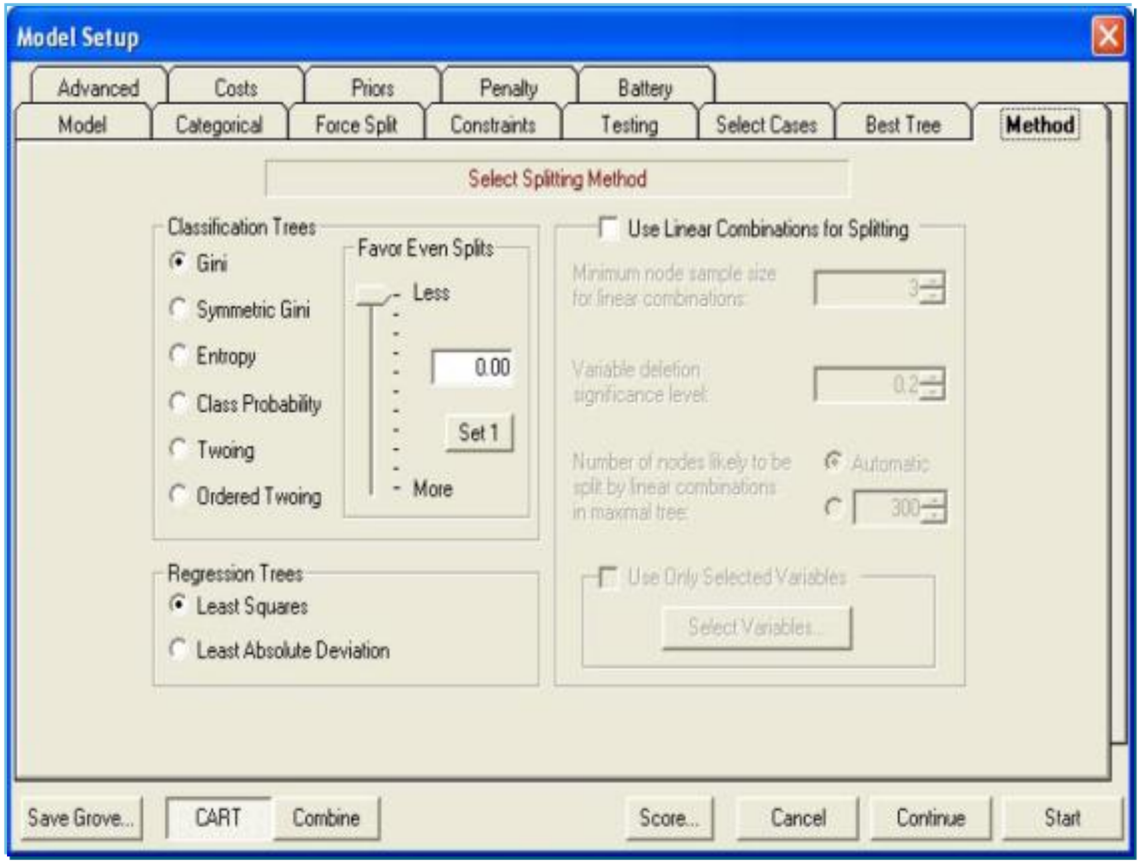
⚠ **CART will terminate the model-building process prematurely if it finds that it needs more linear combination splits than were actually reserved.**

⚠ **Linear combination splits will be automatically turned off for all nodes that have any constant predictors (all values the same for all records). Thus, having a constant predictor in the training data will effectively turn off linear combinations for the entire tree.**

III. Setting Up CART Model

THE METHOD TAB

A Snapshot of Method Tab



Select Splitting Method

Classification Trees

- ● Gini
- ○ Symmetric Gini
- ○ Entropy
- ○ Class Probability
- ○ Twoing
- ○ Ordered Twoing

Favor Even Splits

Less

0.00

Set 1

More

□ Use Linear Combinations for Splitting

Minimum node sample size for linear combinations: 3

Variable deletion significance level: 0.2

Number of nodes likely to be split by linear combinations in maximal tree: ● Automatic ○ 300

□ Use Only Selected Variables

Select Variables...

Regression Trees

- ● Least Squares
- ○ Least Absolute Deviation

For a *two-level* dependent variable that can be predicted with a ***relative error of less than 0.50***, the ***Gini splitting rule*** is typically best.

For a *two-level* dependent variable that can be predicted with a ***relative error of only 0.80 or higher***, ***Power-Modified Twoing*** tends to perform best.

For target variables with *four to nine levels*, ***Twoing*** has a good chance of being the best splitting rule.

For higher-level categorical dependent variables with *10 or more levels*, ***Twoing*** or ***Power-Modified Twoing*** is often considerably more accurate than Gini.

III. Setting Up CART Model

THE ADVANCED TAB

- **Minimum Node Size:**
  - *Parent Node Minimum Cases:* In search of best optimal tree, CART may continue splitting nodes till it runs out of data. However, nodes with negligible size are practically irrelevant. This option allows the user to set a minimum number of records for all parent nodes of the tree.
  - *Terminal Node Minimum Cases: This control specifies the smallest number of observations that may be separated into a child node.*

  As a rule of thumb, parent node's minimum size should be at least three times the minimum size of terminal node.

- **Minimum Complexity:**
  - Setting complexity to a value greater than the default value 'zero' places a penalty on larger trees, and causes CART to stop its tree-growing process before reaching the largest possible tree size a child node.

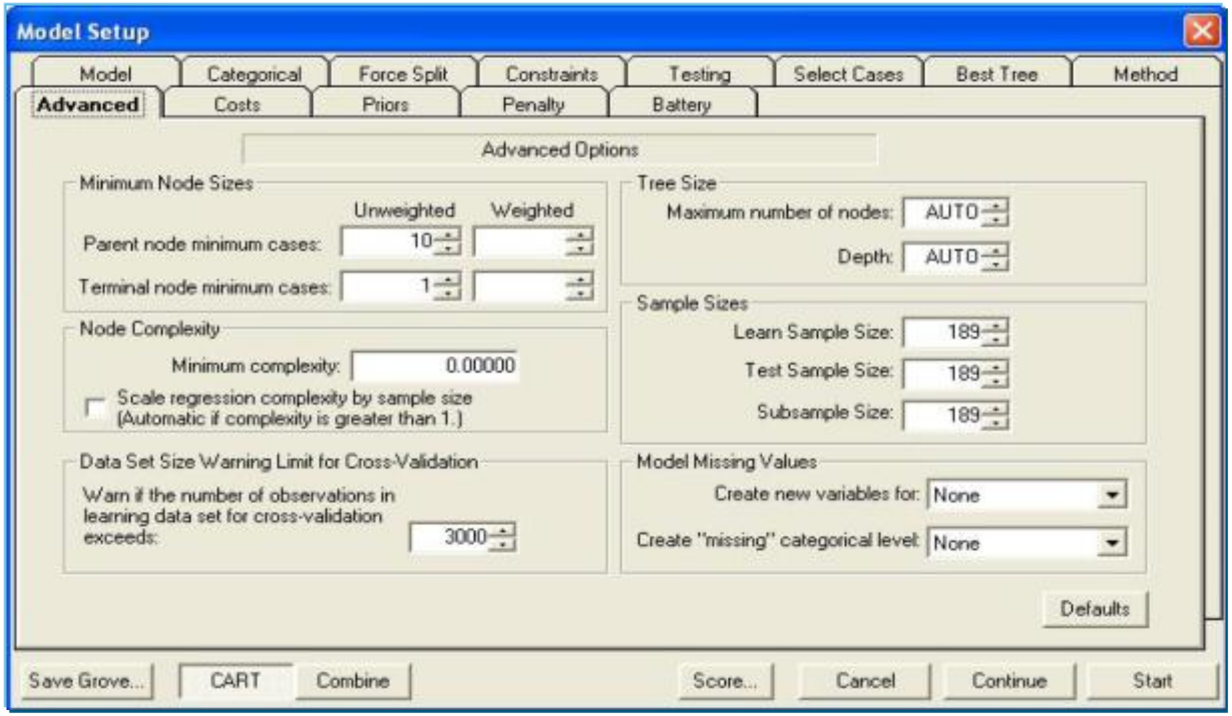
- **Tree Size:**
  - Used to specify 'Maximum number of nodes' (internal plus terminal) and 'Depth' (the root node corresponds to the depth of zero!)

- **Sample Size:**
  - *Learn Sample Size:* This *option* limits CART to processing only the first part of the data available and simply ignoring any data that comes after the allowed records. The control allows for faster processing of the data because the entire data file is never read.
  - *Test Sample Size:* The TEST setting is similar to LEARN.
  - *Subsample Size:* In node sub-sampling, the tree generation process continues to work with the complete data set in all respects except for the split search procedure (which is conducted on a specified size of node sub-sample).

III. Setting Up CART Model

THE ADVANCED TAB

A Snapshot of Advanced Tab



Advanced Options

Minimum Node Sizes

	Unweighted	Weighted
Parent node minimum cases:	10	
Terminal node minimum cases:	1	

Node Complexity

Minimum complexity: 0.00000

□ Scale regression complexity by sample size (Automatic if complexity is greater than 1.)

Data Set Size Warning Limit for Cross-Validation

Warn if the number of observations in learning data set for cross-validation exceeds: 3000

Tree Size

Maximum number of nodes: AUTO

Depth: AUTO

Sample Sizes

Learn Sample Size: 189

Test Sample Size: 189

Subsample Size: 189

Model Missing Values

Create new variables for: None

Create "missing" categorical level: None

By default, CART sets the maximum DEPTH value so large that it will never be reached.

Unlike complexity, the NODES and DEPTH controls may handicap the tree and result in inferior performance.

Usually decision tree users tend to set depth values to small limits such as five or eight. These limits are generally set very low to create the illusion of fast data processing. However, if user wants to be sure to get the best tree, there is a need to allow for somewhat deeper trees.

III. Setting Up CART Model

THE COSTS TAB

TP / TN / FP / FN CONCEPT

⬜ *Correct Classification*

🟧 *Misclassification*

Actual Class	Predicted Class	Prediction Type	Match Result	Classification Type
1	1	POSITIVE	TRUE	TRUE POSITIVE
0	0	NEGATIVE	TRUE	TRUE NEGATIVE
1	0	NEGATIVE	FALSE	FALSE NEGATIVE
0	1	POSITIVE	FALSE	FALSE POSITIVE

⚠ Every mistake is not equally serious or equally costly !!

Classic Example: *Medical Test*

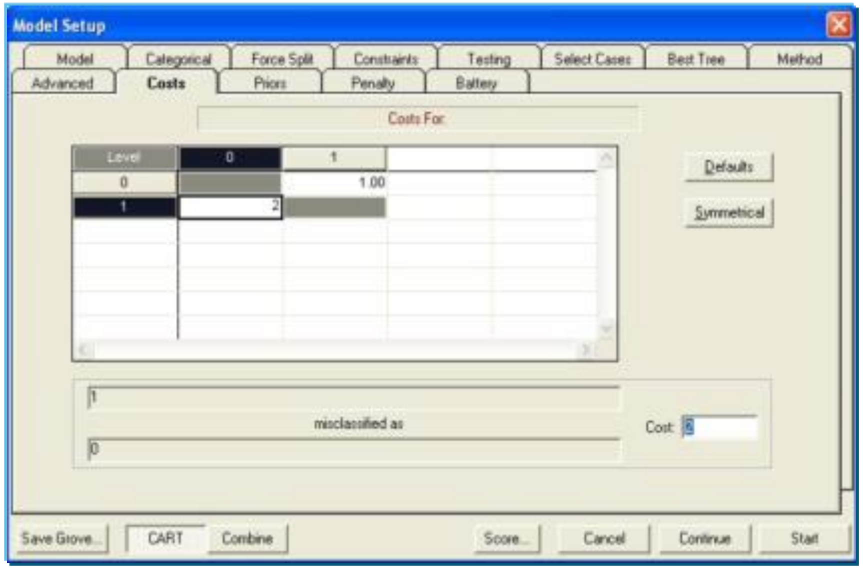
*A **false positive** on a medical test might cause additional more costly tests amounting to several hundreds of dollars. A **false negative** might allow a potentially life-threatening illness to go untreated.*

Note:

***Only cell ratios matter**, that is, the actual value in each cell of the cost matrix is of no consequence—setting costs to 1 and 2 for the binary case is equivalent to setting costs to 10 and 20.*

*CART requires **all costs to be strictly positive** (zero is not allowed). Use small values, such as .001, to effectively impose zero costs in some cells.*

A Snapshot of Costs Tab



Model Setup

Costs For

Level	0	1
0		1.00
1	2	

1

misclassified as

0

Cost: 2

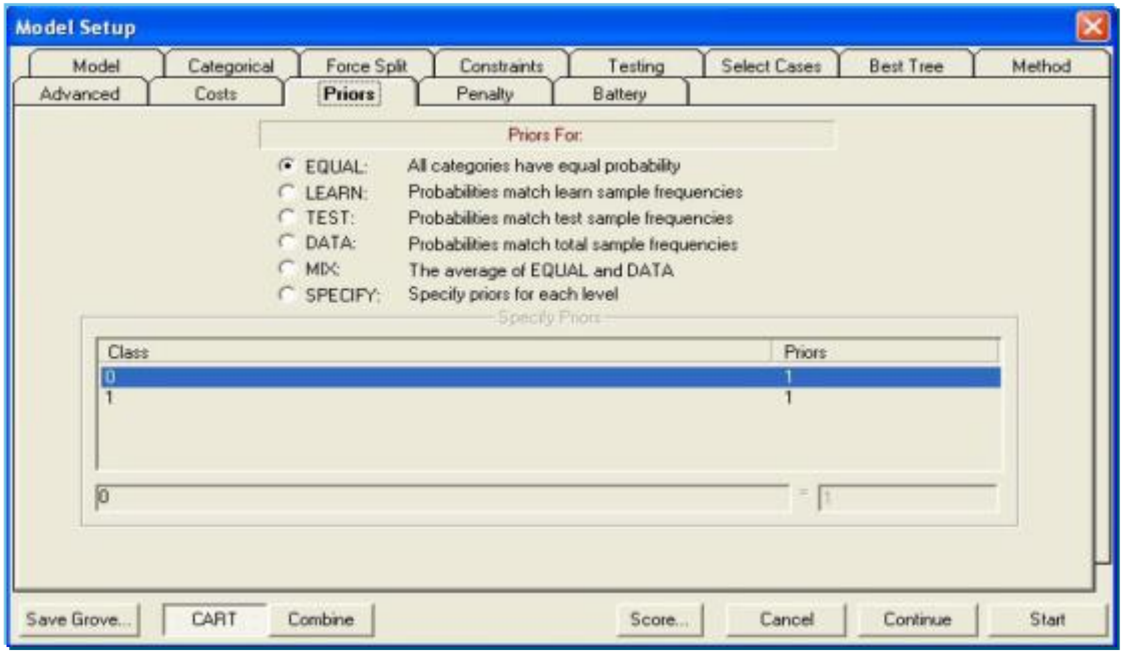
III. Setting Up CART Model

THE PRIORS TAB

Six different priors options are available, as follows:

EQUAL : Equivalent to weighting classes to achieve BALANCE
DATA : Larger classes are allowed to dominate the analysis
MIX : Priors set to the average of the DATA and EQUAL options
LEARN : Class sizes calculated from LEARN sample only
TEST : Class sizes calculated from TEST sample only
SPECIFY : Priors set to user-specified values

A Snapshot of Priors Tab



Priors For:

● EQUAL: All categories have equal probability
○ LEARN: Probabilities match learn sample frequencies
○ TEST: Probabilities match test sample frequencies
○ DATA: Probabilities match total sample frequencies
○ MIX: The average of EQUAL and DATA
○ SPECIFY: Specify priors for each level

Specify Priors

Class	Priors
0	1
1	1

0 = 1

EQUAL PRIORS

It is the default method used for classification trees to deal with unbalanced data (event rate ≠ 0.50) and ***often works supremely well****. Each class is treated as equally important for the purpose of achieving classification accuracy.*

Priors are usually specified as fractions that sum to 1.0. In a two-class problem EQUAL priors would be expressed numerically as 0.50, 0.50, and in a three-class problem they would be expressed as 0.333, 0.333, 0.333.

Other Options

PRIORS DATA (or PRIORS LEARN or PRIORS TEST) makes no adjustments for relative class sizes. Under this setting small classes will have less influence on the CART tree and may even be ignored if they interfere with CART's ability to classify the larger classes accurately. PRIORS DATA is perfectly reasonable when the importance of classification accuracy is proportional to class size.

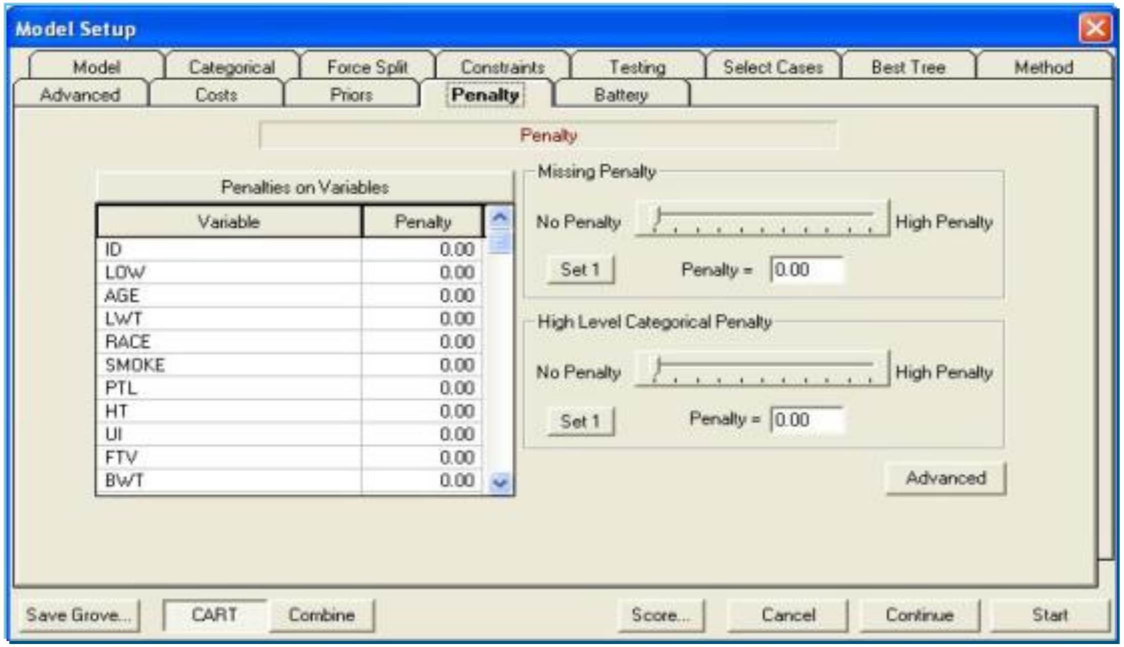
III. Setting Up CART Model

THE PENALTY TAB

Penalties can be imposed on variables to reflect the reluctance to use a variable as a splitter. Of course, the modeler can always exclude a variable; the penalty offers an opportunity to permit a variable into the tree but only under special circumstances.

A penalty will lower a predictor's improvement score, thus making it less likely to be chosen as the primary splitter.

A Snapshot of Penalty Tab



Penalties on Variables

Variable	Penalty
ID	0.00
LOW	0.00
AGE	0.00
LWT	0.00
RACE	0.00
SMOKE	0.00
PTL	0.00
HT	0.00
UI	0.00
FTV	0.00
BWT	0.00

Missing Penalty

No Penalty — High Penalty

Set 1 Penalty = 0.00

High Level Categorical Penalty

No Penalty — High Penalty

Set 1 Penalty = 0.00

THREE CATEGORIES OF PENALTY

- ***Missing Value Penalty**:*

  Predictors are penalized to reflect how frequently they are missing. The penalty is recalculated for every node in the tree.

- ***High Level Categorical Penalty**:*

  Categorical predictors with many levels can distort a tree due to their explosive splitting power. The HLC penalty levels the playing field.

- ***Predictor Specific Penalties**:*

  Each predictor can be assigned a custom penalty. Setting the penalty to one is equivalent to effectively removing that predictor from the predictor list.

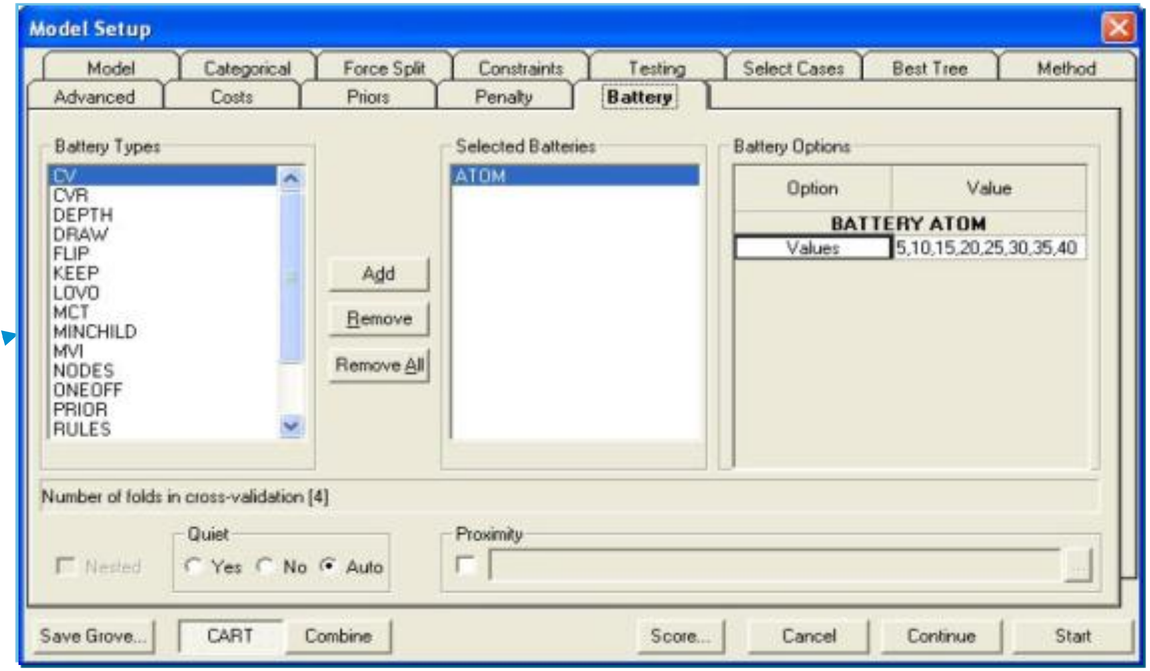
***Note**: Penalties for missing values (for categorical and continuous predictors) and a high number of levels (for categorical predictors only) can range from "No Penalty" to "High Penalty".*

III. Setting Up CART Model

THE BATTERY TAB

The optimal values for many parameters in CART algorithm cannot be determined beforehand and require a trial and error experimental approach. CART batteries are *designed to automate the most frequently occurring modeling situations* requiring multiple collections of CART runs.

A Snapshot of Battery Tab



Selected Batteries: ATOM

Option	Value
BATTERY ATOM	
Values	5,10,15,20,25,30,35,40

Number of folds in cross-validation [4]

☐ Nested

Quiet: ○ Yes ○ No ● Auto

Proximity ☐

There are numerous battery types available for use.

III. Setting Up CART Model

THE BATTERY TAB

Battery Type	Function	Illustrative Valid Values / Actions
ATOM	Varies the atom size (the minimum required parent node size)	5,10,15,20,25,30,35,40
CV	Runs Cross Validation	5,10,20,50
CVR	Runs multiple CV using different random number seeds	20 cycles
DEPTH	Specifies the depth limit of the tree	2,3,4,5,6,7,8,9
DRAW	Runs a series of models where the learn sample is repeatedly drawn (without replacement) from “main” learn sample as specified by the Testing tab; the test sample is not altered	70% Learn, 30% Test: Twenty 50% drawings from the Learn partition
FLIP	Generates two runs with the meaning of learn and test samples flipped	Two runs by default: Original & Flip
KEEP	Randomly selects a specified number of variables from the initial list of predictors and repeats the random selection multiple times.	Sampling 10 predictors, thirty times
LOVO Leave One Variable Out	Generates a sequence of runs where each run omits one of the variables on the predictor list one at a time. Assuming K predictors on the initial keep list, the battery produces K models having K-1 predictors each.	Specify a set of K predictors
MCT	Generates a Monte Carlo test on the significance of model performance	Specify a set of K predictors
MINCHILD	Varies the required minimum terminal node size	5,10,15,20,25,30,35,40
MVI	Addresses missing value handling by running a series of five models: MVI_No_P : use regular predictors, missing value indicators, and no missing value penalties No_MVI_No_P : use regular predictors only (default CART model, no MVIs, no penalties) MVI_only : use missing value indicators only (no regular predictors, no penalties) MVI_P : use regular predictors, missing value indicators, and missing value penalties No_MVI_P : use regular predictors and missing value penalties (no MVIs)	Specify a set of K predictors

III. Setting Up CART Model

THE BATTERY TAB

Battery Type	Function	Illustrative Valid Values / Actions
NODES	Varies the limit on the tree size in nodes according to a user-supplied setting	10,20,30,50,100
ONEOFF	Displays results of using one variable at a time to predict the response **Note:** Better than correlation analysis, tries to identify potential non-linearity as well.	Specify a set of K predictors
PRIOR	Allows priors to be varied within the specified range in user-supplied increments	(0.05, 0.95) to (0.95, 0.05) in increments of 0.05 (19 runs)
RULES	Runs each available splitting method Six Classification Runs for: ▪Gini ▪Symmetric Gini ▪Entropy ▪Class Probability ▪Twoing ▪Ordered Twoing Two Regression Runs for: ▪ Least Squares ▪ Least Absolute Deviation	Specify a set of K predictors
SAMPLE	Investigates the amount of accuracy loss incurred in the course of progressive reduction of the train data size (observation-wise): FIVE runs are produced	Training Data: 100%, 75%, 50%, 25% and 12.5%
SHAVING	Eliminates one or a group of variables based on a specified strategy ▪**BOTTOM** : Remove the least important variables (up to K runs) ▪**TOP** : Remove the most important variables (up to K runs) ▪**ERROR** : Remove the variable with the least contribution based on LOVO battery	Specify a set of K predictors
SUBSAMPLE	Varies the sample size used at each node to determine competitors & surrogates	100, 250, 500, 1000 and 5000
TARGET	Takes each variable from the current predictor list as a target and builds a model to predict this target (classification tree for categorical predictors and regression tree for continuous predictors) using the remaining variables.	Specify a set of K predictors

III. Setting Up CART Model

OTHER FEATURES

- **Unsupervised Learning**
  - *This does not begin with a target variable. Instead the objective is to find groups of similar records in the data. One can think of unsupervised learning as a form of data compression: we search for a moderate number of representative records to summarize or stand in for the original database.*

- **Force Splits:**
  - The Model Setup — Force Split tab allows the user to dictate the splitter to be used in the root node (primary splitter), or in either of the two child nodes of the root. Users wanting to impose some modest structure on a tree frequently desire this control. More specific controls also allow the user to specify the split values for both continuous and categorical variables.

- **Constraints:**
  - By default, all predictors are allowed to be used as primary splitters and as surrogates at all depths and node sizes. The Model Setup — Constraints tab is used to specify at which depths and in which partitions (by size) the predictor, or group of predictors, are not permitted to be used, either as a splitter, a surrogate, or both.

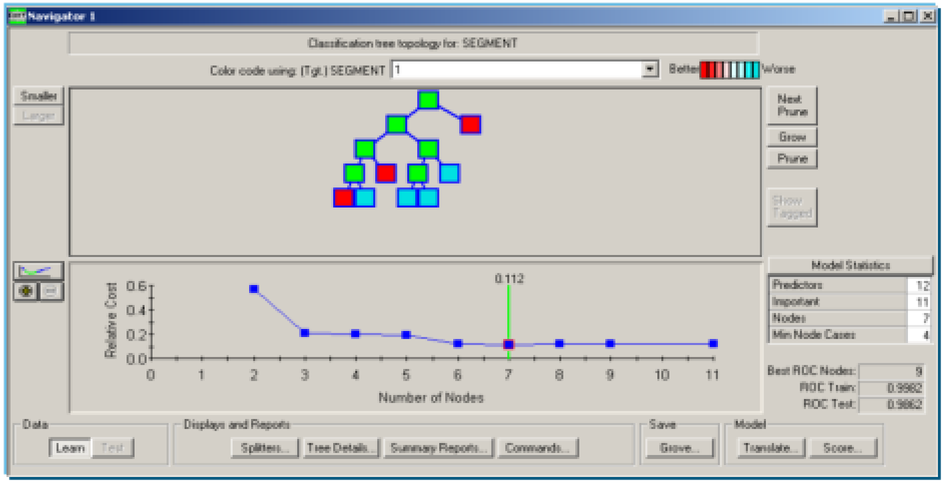
- **Ensemble Models:**
  - CART's Combine dialog allows the user to choose from two methods for combining CART trees into a single predictive model. In both **Bootstrap Aggregation** (**Bagging**) and **Adaptive Resampling and Combining** (**ARCing**), a set of trees is generated by resampling with replacement from the original training data.

Bagging : Each new resample is drawn in an identical way (independent samples)

ARCing : The way a new sample is drawn for the next tree depends on the performance of the prior trees. Cases that get misclassified in previous trees receive an increasing probability of selection in the next sample; while cases getting correctly classified in previous trees receive declining weights.

IV. CART Output

NAVIGATOR



Classification tree topology for: SEGMENT

Color code using: (Tgt.) SEGMENT 1

Model Statistics

Predictors	12
Important	11
Nodes	7
Min Node Cases	4

Best ROC Nodes: 9
ROC Train: 0.9982
ROC Test: 0.9862

__Navigator__ provides an immediate snapshot of the tree's size and depth. By default, the optimal or minimum cost tree is initially displayed.

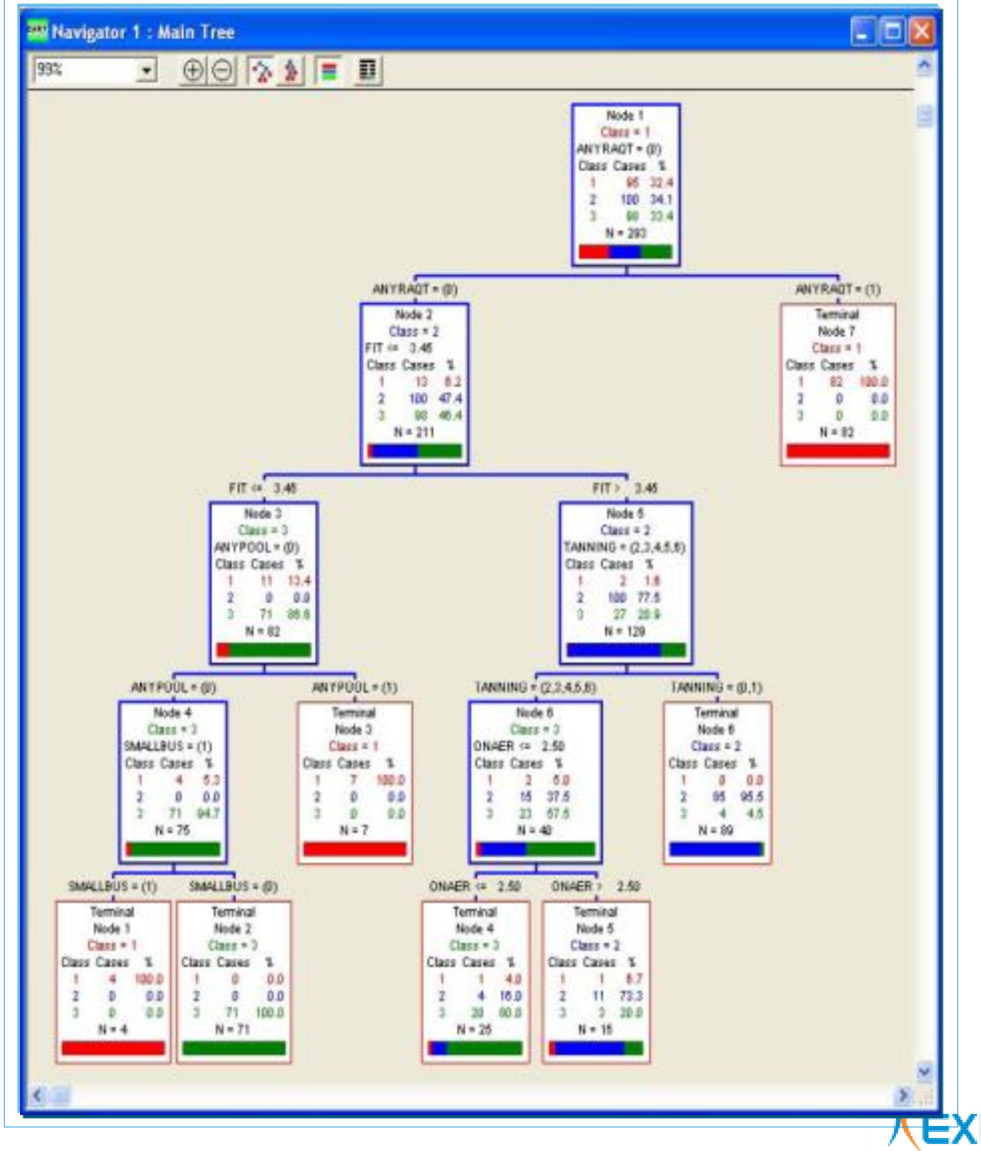
To view the __entire tree__, click on the [__Tree Details...__] button at the bottom of the Navigator.

To get a __SAS code__ for the current CART model, click on the [__Translate...__] button at the bottom of the Navigator.

To __score new dataset__ using current CART Model, click on the [__Score...__] button at the bottom of the Navigator.

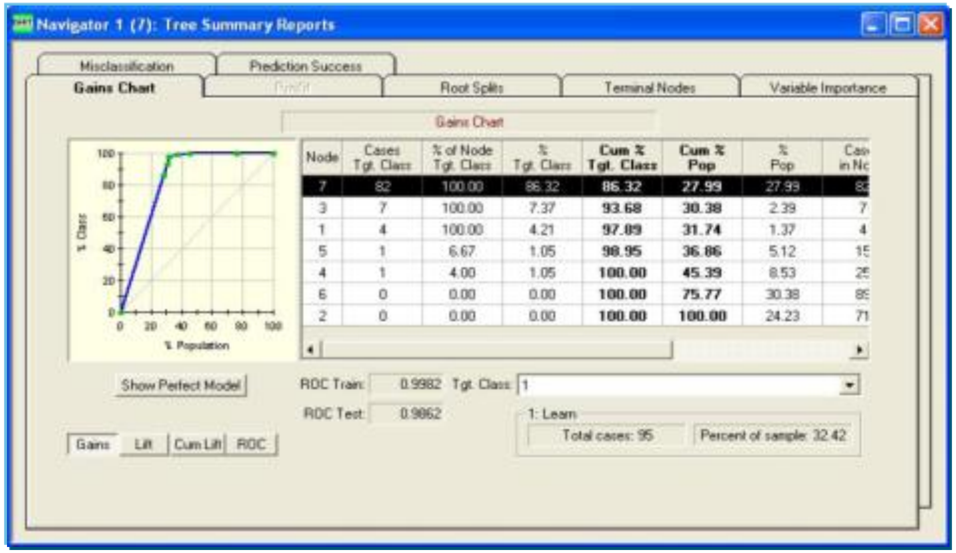
Do remember to save grove for any future reference.

TREE DETAILS



IV. CART Output

GAINS CHART



Node	Cases Tgt. Class	% of Node Tgt. Class	% Tgt. Class	Cum % Tgt. Class	Cum % Pop	% Pop	Cas in Nc
7	82	100.00	86.32	86.32	27.99	27.99	82
3	7	100.00	7.37	93.68	30.38	2.39	7
1	4	100.00	4.21	97.89	31.74	1.37	4
5	1	6.67	1.05	98.95	36.86	5.12	15
4	1	4.00	1.05	100.00	45.39	8.53	25
6	0	0.00	0.00	100.00	75.77	30.38	89
2	0	0.00	0.00	100.00	100.00	24.23	71

ROC Train: 0.9982 Tgt. Class: 1

ROC Test: 0.9862

1: Learn

Total cases: 95 Percent of sample: 32.42

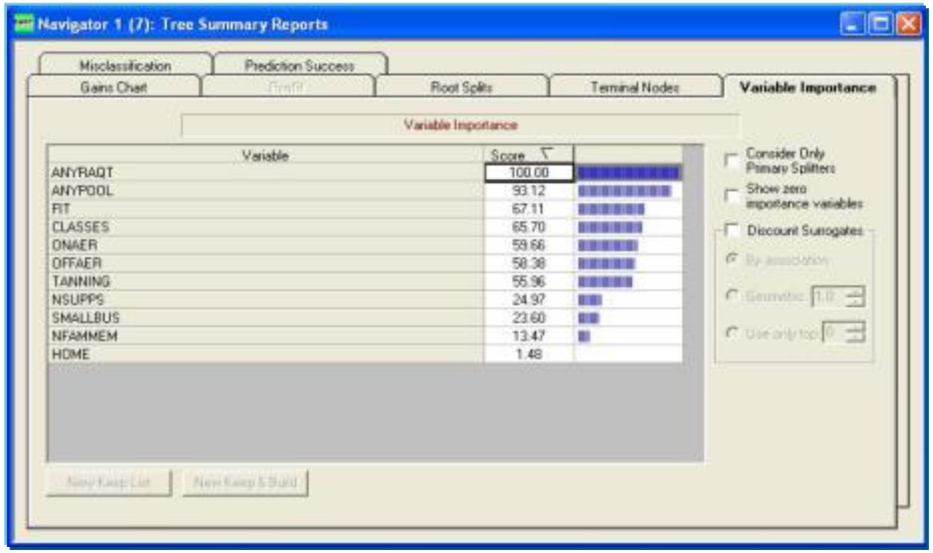
*The overall performance of the current tree is summarized in the Summary Reports dialog tabs. To access the reports, click [**Summary Reports…**] at the bottom of the Navigator window.*

***Gains Chart**: The vertical difference between two lines depicts the **gain from CART model** at each point (over any random model) along the x-axis. The Gains Table can be exported to Excel by a right-mouse click and then choosing [**Export…**] from pop-up menu.*

***Variable Importance**: The scores reflect the contribution each variable makes in classifying or predicting the target variable, with the contribution stemming from both the **variable's role** as a **primary splitter** and as a **surrogate** to other primary splitters.*

***Root Splits**: The report shows the **competing root node splits** in reverse order of improvement.*

VARIABLE IMPORTANCE



Variable	Score
ANYRAQT	100.00
ANYPOOL	93.12
FIT	67.11
CLASSES	65.70
ONAER	59.66
OFFAER	58.38
TANNING	55.96
NSUPPS	24.97
SMALLBUS	23.60
NFAMMEM	13.47
HOME	1.48

□ Consider Only Primary Splitters

□ Show zero importance variables

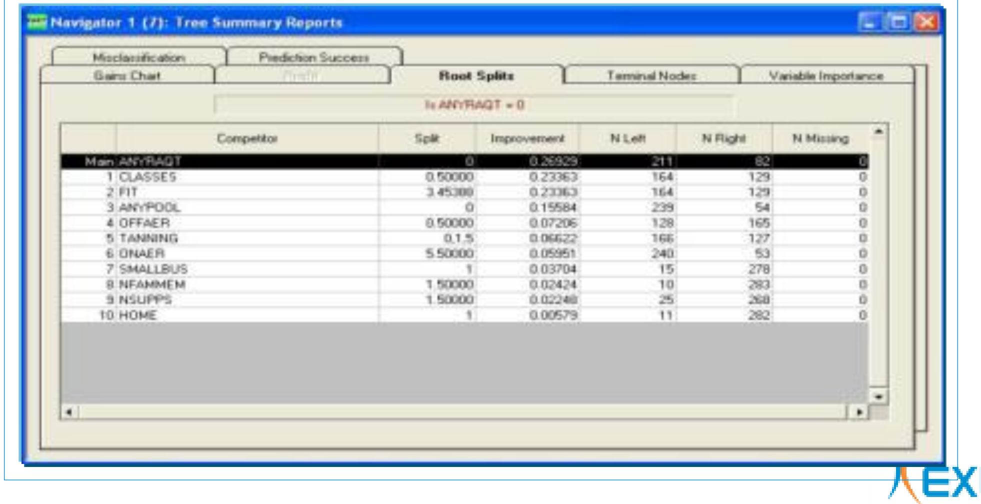
□ Discount Surrogates

● By association

○ Geometric 1.0

○ Use only top 0

ROOT SPLITS

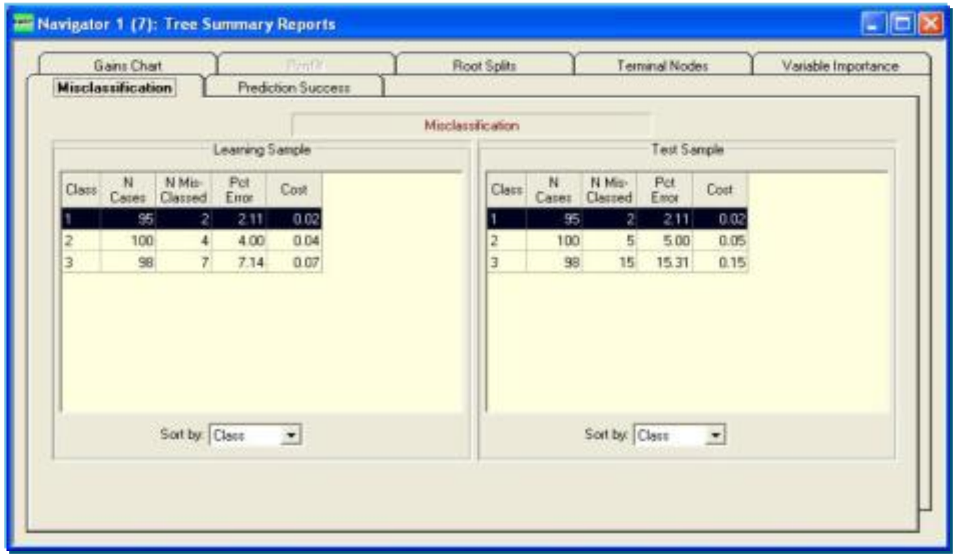


Is ANYRAQT = 0

	Competitor	Split	Improvement	N Left	N Right	N Missing
Main	ANYRAQT	0	0.26929	211	82	0
1	CLASSES	0.50000	0.23363	164	129	0
2	FIT	3.45300	0.23363	164	129	0
3	ANYPOOL	0	0.15584	239	54	0
4	OFFAER	0.50000	0.07206	128	165	0
5	TANNING	0.1.5	0.06622	166	127	0
6	ONAER	5.50000	0.05951	240	53	0
7	SMALLBUS	1	0.03704	15	278	0
8	NFAMMEM	1.50000	0.02424	10	283	0
9	NSUPPS	1.50000	0.02248	25	268	0
10	HOME	1	0.00579	11	282	0

IV. CART Output

MISCLASSIFICATION



Misclassification

Learning Sample

Class	N Cases	N Mis-Classed	Pct Error	Cost
1	95	2	2.11	0.02
2	100	4	4.00	0.04
3	98	7	7.14	0.07

Test Sample

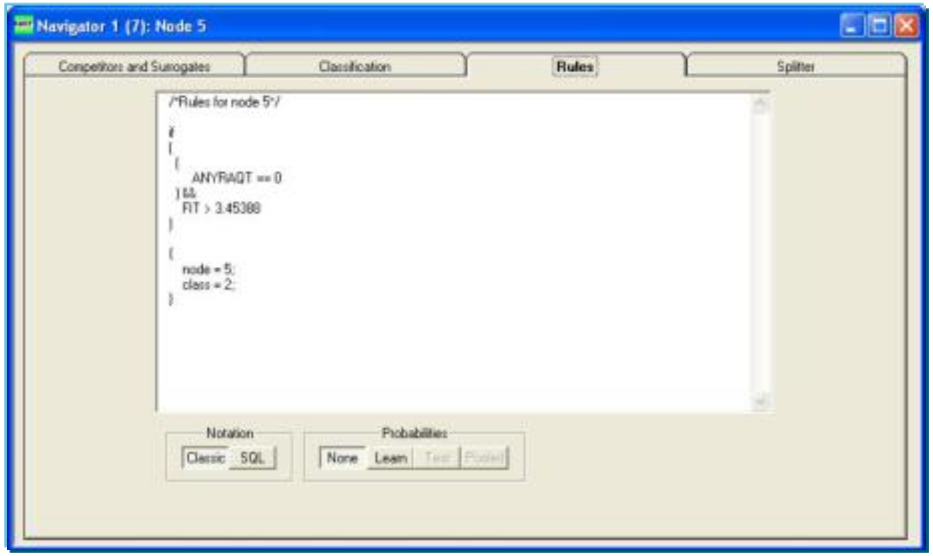
Class	N Cases	N Mis-Classed	Pct Error	Cost
1	95	2	2.11	0.02
2	100	5	5.00	0.05
3	98	15	15.31	0.15

Misclassification: *The report shows how many cases were incorrectly classified in the overall tree for both learn and test (or cross-validated) samples.*

Node Rule: *The Terminal node reports (with the exception of the root node) contain a Rules dialog that displays the rules for the selected node and/or sub-tree.*

Prediction Success: *The Prediction Success table (also known as the confusion matrix) shows whether CART tends to concentrate its misclassifications in specific classes and, if so, where the misclassifications are occurring.*

NODE RULES

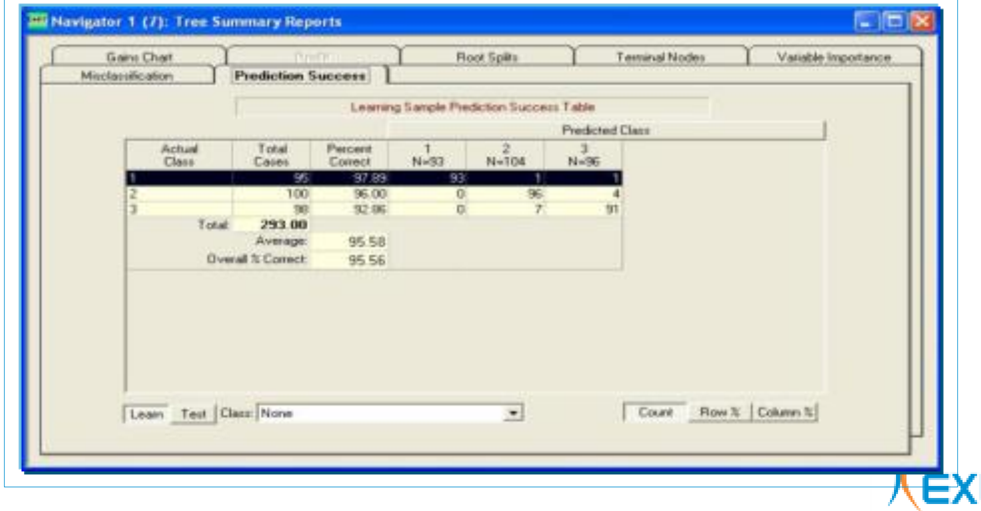


```
/*Rules for node 5*/

if
(
 (
   ANYRAQT == 0
 ) &&
  FIT > 3.45388
)

{
  node = 5;
  class = 2;
}
```

PREDICTION SUCCESS



Learning Sample Prediction Success Table

			Predicted Class		
Actual Class	Total Cases	Percent Correct	1 N=93	2 N=104	3 N=96
1	95	97.89	93	1	1
2	100	96.00	0	96	4
3	98	92.86	0	7	91
Total:	293.00				
Average:		95.58			
Overall % Correct:		95.56			

Thanks

For queries, contact Varun Aggarwal at Varun.Aggarwal@exlservice.com