**In-Class Activity #11: Decision Trees in R (Due Friday , Apr 14, 9:00am )**

**What to submit:** a single word/pdf file with answers for Questions 1, 2 and 3.

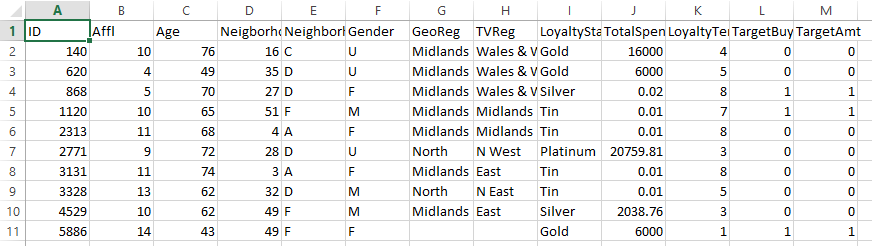
You’ll need two files to do this exercise: **dTree.r** (the R script file) and **OrganicsPurchase.csv** (the data file[[1]](#footnote-1)). Both of those files can be found on the course site. The data file contains 22,223 customer records with demographic information and whether the customer bought organic products.

**Before you start:**

* Download both files and save them to the same folder where you keep your R files.
* Make sure that the file names are the same as “**dTree.r”** and **“OrganicsPurchase.csv”**
* Make sure that you have internet access (in order to install additional packages from the web)

**Part 1: Look at the Data File**

1. Open the **OrganicsPurchase.csv** data file in Excel. If it warns you that the file format and extension don’t match and that detects that it is a SYLK file, that’s ok. Just click “Yes” and then “OK.”
2. You’ll see something like this:

  
This is the raw data for our analysis. You can see the first variable (field) is called ID, the second variable (field) is called Affl, the third variable (field) is called Age, and so on.  
  
The remaining lines of the file contain the data for each customer. So the value of ID for the first customer is 140, the value of Affl for the first customer is 10, the value of Age for the first customer is 76, and so on.

This is a full list of the variables:

|  |  |
| --- | --- |
| **Variable Name** | **Variable Description** |
| **ID** | Customer loyalty identification number |
| **Affl** | Affluence grade on a scale from 1 to 30 (1 = least affluent, 30 = most affluent) |
| **Age** | Age, in years |
| **NeighborhoodCluster** | Identifier for residential neighborhood |
| **NeighborhoodClusterGroup** | A set of similar neighborhoods  (A-F = neighborhood type, U=unknown) |
| **Gender** | M = male, F = female, U = unknown |
| **GeoReg** | Geographic region |
| **TVReg** | Television region |
| **LoyaltyStat** | Loyalty status: Tin, Silver, Gold, or Platinum |
| **TotalSpend** | Total amount spent (in British pounds) |
| **LoyaltyTenure** | Time as loyalty card member (in months) |
| **TargetBuy** | Organics purchased? 1 = Yes, 0 = No |
| **TargetAmt** | Number of organic products purchased |

Categorical outcome Variable: **TargetBuy** (1 = did buy, 0 = did not buy).

We will use this data set to predict whether people will buy organic products (**TargetBuy**) based on any combination of the remaining variables (i.e., Affl, Age, Gender, etc.).

Some variables, like ID, are irrelevant to the analysis. Other variables, like TargetAmt, are also not useful because they don’t give additional insight into the outcome (obviously if TargetAmt is greater than 0 then TargetBuy will be 1).

1. Now look at line 11 of the file:  
     
   

You’ll see GeoReg and TVReg have no values.

|  |  |
| --- | --- |
| **Variable Name** | **Value** |
| **ID** | 5886 |
| **Affl** | 14 |
| **Age** | 43 |
| **NeighborhoodCluster** | 49 |
| **NeighborhoodClusterGroup** | F |
| **Gender** | F |
| **GeoReg** | *missing* |
| **TVReg** | *missing* |
| **LoyaltyStat** | Gold |
| **TotalSpend** | 6000 |
| **LoyaltyTenure** | 1 |
| **TargetBuy** | 1 |
| **TargetAmt** | 1 |

1. Close the OrganicsPurchase.csv file. If it asks you to save the file, choose “**Don’t Save**”.

**Part 2: Explore the dTree.r Script**

1. Open the **dTree.r** file in RStudio. This contains the R script that performs the decision tree analysis.

Make sure the code is “colorized” – meaning that the text color is changing depending on its use.

If all the text is black-and-white then there is a problem with your script. Usually it means that your file has a .txt extension (i.e., dTree.r.txt).

1. Look at lines 8 through 28. These contain the parameters for the decision tree model. Here’s a rundown:

|  |  |  |
| --- | --- | --- |
| **Variable Name in R** | **Value** | **Description** |
| INPUT\_FILENAME | OrganicsPurchase.csv | The data is contained in OrganicsPurchase.csv |
| OUTPUT\_FILENAME | DecisionTreeOutput.txt | The text output of the analysis |
| PLOT\_FILENAME | TreeOutput.pdf | The decision tree plot is output to TreeOutput.pdf |
| TRAINING\_PART | 0.50 | 50% of the data will be used to train the model. |
| MINIMUMSPLIT | 50 | Each node must have at least 50 observations. |
| COMPLEXITYFACTOR | 0.005 | Error must be reduced by at least 0.005 for the tree to add an additional split. (Note: A smaller COMPLEXITYFACTOR often results in a more complex tree.) |
| OUTCOME\_COL | 12 | The outcome variable (TargetBuy) is in column 12 of the OrganicsPurchase dataset. |

1. Lines 34 through 41 load the three packages necessary for decision tree analysis – **rpart**, **caret**, and **rpart.plot**.

|  |
| --- |
| **Experiencing Problem with Package Installation?**  If you use your own computer (especially a Mac), there is a small chance that you may have problems downloading the “caret” package. If so, try a different computer.  **Computer labs**  The computer labs at Fox do have R/RStudio and seem to have no issue with the caret package. Here is the list of computer labs and hours within Fox: <http://www.fox.temple.edu/technology/it/resources/computer-labs/>. The computers in the Tech Center may not have R/RStudio installed. |

1. Look at lines 55 through 71. These lines partition the dataset into two subsets, a training set and a validation set.

You do not need to worry about these lines much. Just need to know that:

* The training set will be used to create the decision tree model;
* And the validation set will be used to evaluate the classification accuracy of the model.

1. Now let’s look at the decision tree model. Scroll down to lines 82 through 84:   
     
      
   You can see a few things at work:

* The **rpart()** function is use to classify the data into a decision tree (the results will be stored in a variable called **MyTree**).
* The formula for a decision tree model is **outcome ~ predictor1 + predictor2 + etc**.
* **TargetBuy** is the outcome event you’re trying to predict (1 = did buy, 0 = did not buy).
* All nine variables to the right of the ~ are predictor variables. Not all of those will be important enough to be included in the decision tree.
* **Our MINIMUMSPLIT and COMPLEXITYFACTOR parameters from above are used here.**

**Part 3: Execute the dTree.r Script**

1. Select Sesson/Set Working Directory/To Source File Location to change the working directory to the location of your R script.
2. Select Code/Run Region/Run All. It could take a few seconds to run since the first time it has to install some extra packages to do the analysis. Be patient!

You’ll see a lot of action in the Console window at the bottom left side of the screen, ending with this:  
  
> prp(prunedTree, main=paste("Decision Tree\n(Correct classification rate ",

+ round(predRateTraining,4)\*100,

+ .... [TRUNCATED]

> dev.off();

RStudioGD

2

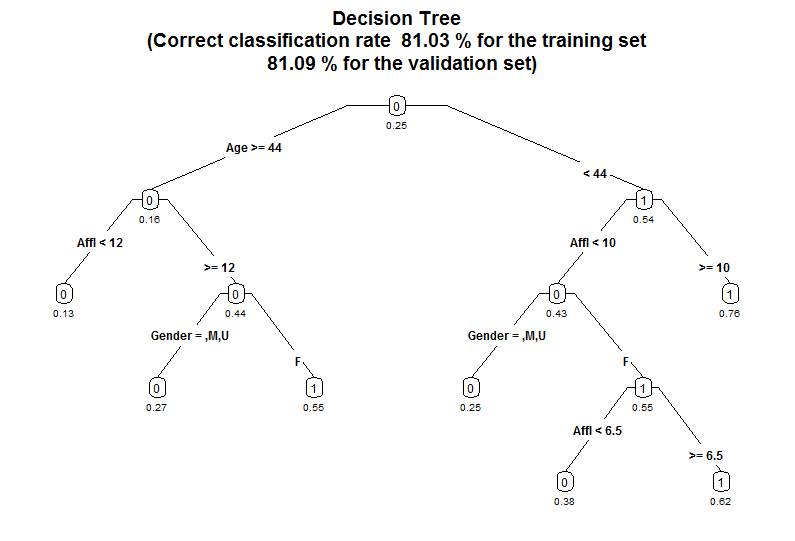
In some cases, you may get a few warning messages like the following. This is fine. You can ignore them.

Warning messages:

1: package caret?was built under R version 3.2.4

2: package ggplot2?was built under R version 3.2.3

1. And you’ll see the decision tree on the right:



**Part 4: Read the Decision Tree**

**Correct classification rate** (see the subtitle of the plot)

The tree correctly predicts whether someone will buy organics 81.03% of the time for the training set, and 81.09% of the time for the validation set. The tree has seven leaf nodes (nodes with nothing beneath them). Each of those seven leaf nodes represents a prediction based on a combination of predictor variables.

**The numbers/values under the nodes**

-- represent the probabilities of a **positive** outcome (=1) – in this case, that they will buy organic products (i.e. TargetBuy = 1).

**The “0” and “1” labels inside the nodes**

-- indicate whether there are more outcomes equal to 1 or 0 in that node – note that all the “0” nodes have a probability less than 0.5, and all the “1” nodes have a probability greater than 0.5.

**Predictor variables**

Age is the best predictor, as it is the first split. It creates the most differentiation (separation) between buyers and non-buyers.

Check the branch on the left: for those customers 44 years old or older, Affluence is the next best predictor, then gender.

You read the left branch of the tree like this:

*Females who are 44 years or older and have an affluence grade of 12 or more will buy organic products 55% of the time.*

*Males 44 years or older with an affluence grade of 12 or more will buy organic products only 27% of the time.*

*If you’re 44 years or older and have an affluence grade of less than 12 then you’ll buy organic products 13% of the time, regardless of your gender.*

Variables can appear twice within a branch of the tree if it further differentiates between outcomes.

Note that Affl appears twice in the right branch.

**Question 1:**

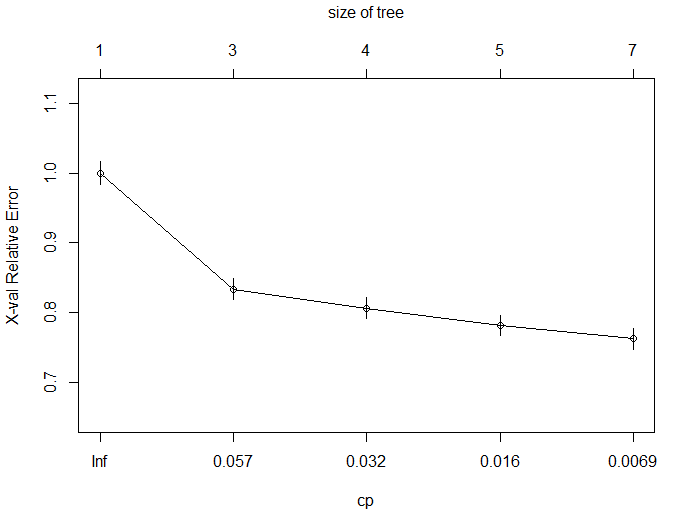
Based on the tree you’ve generated, how likely is it that the following customers will buy organic products:

A 25 year-old male with an affluence grade of 4.5?  
A 30 year-old male with an affluence grade of 15?  
A 30 year-old female with an affluence grade of 15?

A 43 year-old female with an affluence grade of 8?

**Where are the rest of the predictor variables?**

Notice that other predictors like TVReg, LoyaltyStat, and NeighborhoodClusterGroup we have included in the rpart() function, aren’t in the decision tree. This is because the decision tree algorithm determined that they didn’t contribute enough beyond Age, Affl, and Gender to meaningfully differentiate between buyers and non-buyers.

1. Now click the back arrow () in the Plots window (the bottom-right section of the screen) and you’ll see the following:  
   

This is a plot of the **complexity parameter** table generated by the decision tree analysis. It plots the size of the tree against the relative error rate of the tree.

* 1. cp (complexity parameter): incremental reduction in error when adding an additional node
  2. X-val relative error: relative error rate against a baseline tree with 1 node.

All this is saying is that the tree gets a lot more accurate (the error rate goes down) when it has 3 nodes and instead of 1 (a tree with one node would just guess everyone buys organics regardless of other data). That reduction in error rate continues as the tree has more nodes.   
  
The tree stopped at 7 nodes because after that cp (i.e. the incremental reduction in error) no longer is greater than the COMPLEXITYFACTOR threshold. So the decision tree function stops at this point because adding nodes (and making the tree more complex) is no longer worth it.

The “relative error rate” is different from the “error rate” that we typically calculate.

The “error rate” that we typically calculate measures how often the decision tree made the wrong prediction (i.e,. buy or not buy).

The “relative error rate” is how much the a tree with n-nodes improves the decision over a tree with 1-node that just puts everyone in the same category – in this case that baseline tree would classify everyone in the data set as buying organics.

That’s why the first tree on this plot has one node (i.e., everyone buys) and a relative error rate of 1.0 (it’s the baseline).

1. Now click the “Clear all” button in the plot window (the broom) to clear the plots you’ve generated.

**Part 5: Change the Complexity Factor**

1. **Smaller COMPLEXITYFACTOR → more complex tree:**

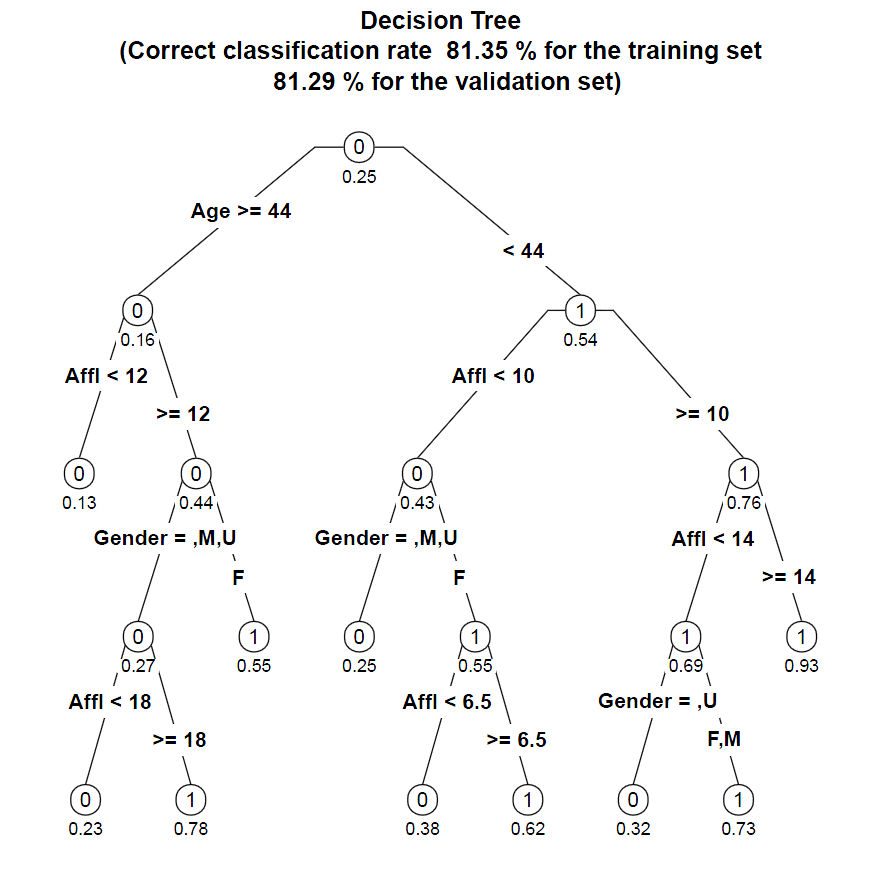
**Go to line 26 and change COMPLEXITYFACTOR from 0.005 to 0.003.** This means a smaller incremental improvement in the tree is necessary to make the decision to add an additional node.

**In other words, we’re willing to put up with a more complex tree if it helps our result.**

1. Run the script by selecting Code/Run Region/Run All. You’ll now see this decision tree.

(It may be difficult to read the decision tree plot – the text is quite small, so let’s use the PDF output from the script. Open the folder that corresponds to your working directory. You should see a file called **TreeOutput.pdf**. Open the file and you will see the tree plot as well.

**When you are done, close the PDF plot of the tree and go back to RStudio!**)

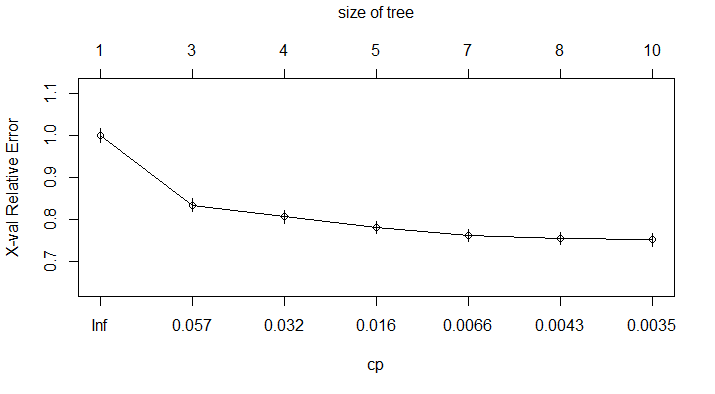


**Question 2:**

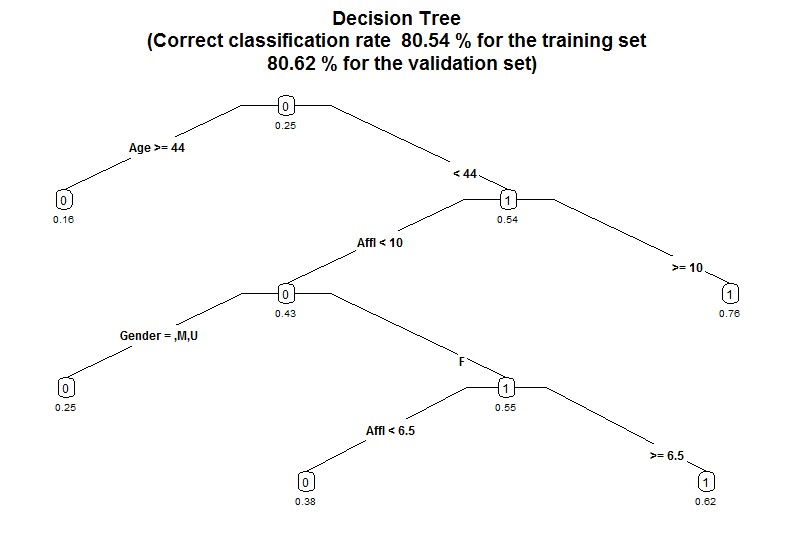
Based on this tree, how likely is it that the following customers will buy organic products:

A 25 year-old male with an affluence grade of 4.5?  
A 65 year-old female with an affluence grade of 20?  
  
Describe the characteristics of the most likely and least likely groups to buy organic products?

We can clearly see the tree is more complex (it has 10 nodes now instead of 7), and that the correct classification rate has risen from 81.09% to 81.29% for the validation set.   
We can also see view complexity parameter plot by clicking the back arrow () :



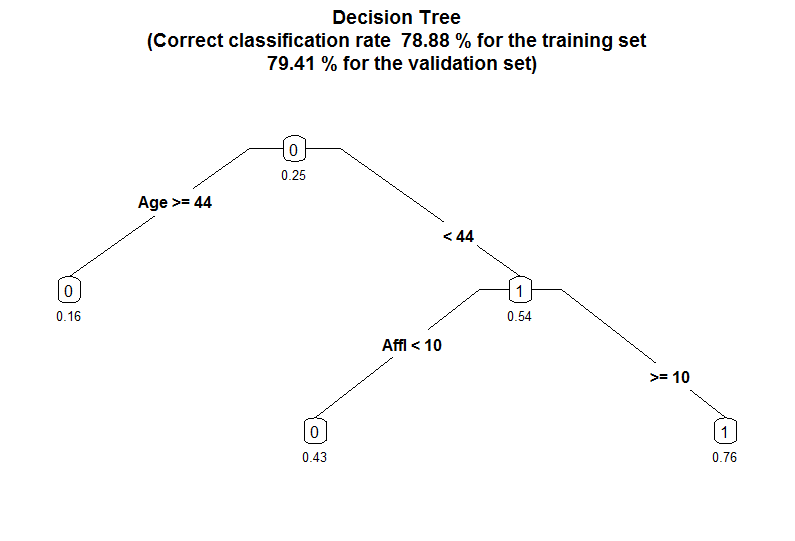
1. **Larger COMPLEXITYFACTOR → less complex tree:**

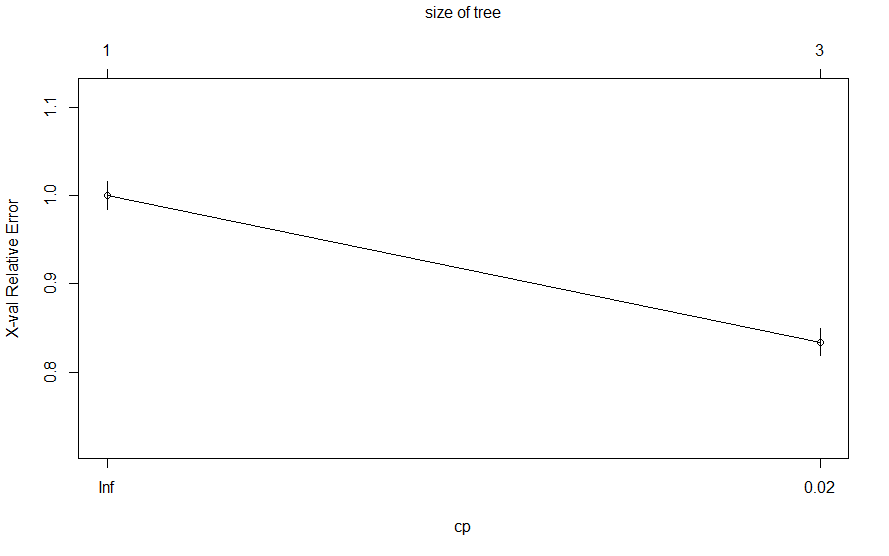
We can also see what happens when we make the COMPLEXITYFACTOR larger. That means that the tree will be less complex (less nodes) because adding nodes have to make an even larger improvement in the overall predictive power of the model.  
  
**Go to line 26 and change COMPLEXITYFACTOR from 0.003 to 0.01.** Re-run the script and it will generate this decision tree:  


This new tree has five nodes, as opposed to our first tree with seven nodes and our second tree with 10 nodes. Increasing the complexity factor threshold makes the tree simpler, and also a little less accurate (80.62% versus 81.09% and 81.29% of the previous trees for the validation set).  
  
We can adjust the complexity factor to find the best tree for our analysis, balancing complexity (too many nodes make the results difficult to read and interpret) and accuracy (generally, more nodes create more accurate predictions and increase the correct classification rate).

We’ll talk a little more about this at the end of the exercise.

**Part 6: Change the minimum split**

1. **Change COMPLEXITYFACTOR back to 0.005 (line 27).** Re-run the script to verify that we’re back to our tree with seven nodes.
2. **Change MINIMUMSPLIT from 50 to 2000 (line 26).** This means that each node has to have at least 2000 observations in it.   
     
   In this case, this means that each node has to describe at least 2000 customers (remember there are over 22,000 customers in the sample). It is another way to control the size of the tree. A larger number prevents the algorithm from adding nodes that only describe very specific demographic groups.  
     
   For example, if there were only 100 customers over 50 years old with an affluence grade less than 5 it would not give them their own node. Instead, it would combine them with another (similar) group of people when creating the tree.
3. Re-run the script. You’ll see the leaf nodes have been reduced from 7 to 3:  
     
   The classification accuracy has fallen from 81.09% to 79.41% for the validation set. And, gender is no longer included in the final tree. This implies that you can make almost as good predictions by using Age and Affluence Level, even if you don’t know the customer’s gender.  
     
   So let’s compare our 25 year-old male with an affluence grade of 4.5. In the original seven node tree on page 4, the likelihood of this customer buying organics was assessed at 25%. In this version of the tree, the likelihood is 43%. We know this tree is less accurate from the correct classification rate, and we can further verify this by clicking on the back arrow () to view the complexity plot:



We see that the error with the three-node tree is around 0.85. We can view the exact error rate of the new tree by going to the Console window at the bottom left of the screen and typing the command   
  
printcp(MyTree)

This displays the complexity statistics for the current tree:  
  
 CP nsplit rel error xerror xstd

1 0.083718 0 1.00000 1.00000 0.016271

2 0.005000 2 0.83256 0.83398 0.015273

The relative error for this tree is 0.833 (look at the xerror column).

We can compare that to the 7 node tree back on page 6; we can tell just by looking at the plot that the error rate is below 0.80. And if you re-run that original tree with a MINIMUMSPLIT of 50 and use printcp(MyTree), you can verify the relative error rate was 0.763.

**Part 7: Which tree do we use?**

We’ve generated four decision trees in this exercise, all based on the same data:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Tree #** | **On  page** | **COMPLEXITY FACTOR** | **MINIMUM SPLIT** | **# of  Nodes** | **Correct Classification Rate**  **(Training Set)** | **Correct Classification Rate**  **(Validation Set)** | **Relative Error Rate** |
| 1 | 4 | 0.005 | 50 | 7 | 81.03% | 81.09% | 0.762 |
| 2 | 7 | 0.003 | 50 | 10 | 81.35% | 81.29% | 0.752 |
| 3 | 8 | 0.01 | 50 | 5 | 80.54% | 80.62% | 0.782 |
| 4 | 9 | 0.005 | 2000 | 3 | 78.88% | 79.41% | 0.834 |

* We can see that Tree #2 has the lowest error and the highest correct classification rate.
* We can also see that Tree #4 has the highest error and the lowest correct classification rate.
* Trees #1 and #3 are somewhere in-between, with #1 doing better than #3.

So which tree is the best? It depends on what your goals are. The difference between the first two trees is relatively small (compared to the others) but there is also not a lot of difference in complexity between those two trees.

If you’re trying to choose a simple tree that is “good enough,” then you’d most likely select Tree #1. If it is important to maximize decision accuracy, Tree #2 is worth the additional complexity.

Note that even comparing the best and the worst trees, there does seem to be that much difference –about 1.88% between the best and the worst trees. However, consider scale: If a grocery chain has 500,000 customers per year, the ability to improve your decision accuracy by 2% means you can identify 10,000 potential buyers of organic products that your competitors may have missed.  
  
You can also choose to target those with the highest propensity to buy, or try to persuade those customers with the lowest propensity to buy.

**Question 3:**

Based on the analyses we have done so far, use your own words to summarize how complexity factor and minimum split can alter the decision tree.

1. Adapted from SAS Enterprise Miner data set. [↑](#footnote-ref-1)