**In-Class Activity #12: Decision Trees in R**

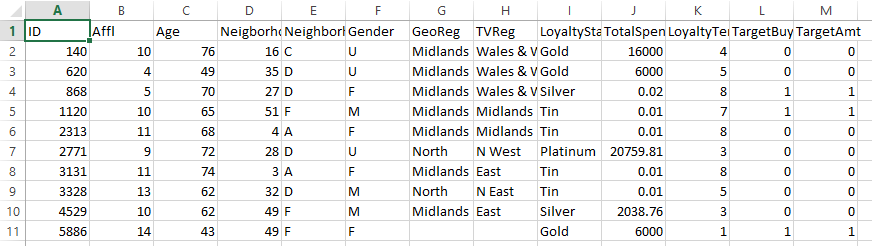
|  |
| --- |
| **Submission Instructions**  **What to submit:** a single word/pdf file with answers for **Questions 1, 2 and 3**.  You’ll need two files to do this exercise: **dTree.r** (the R script file) and **OrganicsPurchase.csv** (the data file[[1]](#footnote-1)). Both of those files can be found 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 ID for the first customer is 140, the Affl for the first customer is 10, the 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 |

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.

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 36 through 40 load the two packages necessary for decision tree analysis – **rpart** and **rpart.plot**.

|  |
| --- |
| **Experiencing Problem with Package Installation?**  If you use your own computer, there is a small chance that you may have problems downloading the packages. If so, try a different computer.  **Computer labs**  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 65 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:

MyTree <- rpart(TargetBuy ~ Affl + Age + NeighborhoodClusterGroup + Gender + GeoReg + TVReg +

LoyaltyStat + TotalSpend + LoyaltyTenure, data=trainingSet, method="class",

control=rpart.control(minsplit=MINIMUMSPLIT, cp=COMPLEXITYFACTOR))

The **rpart()** function is use to classify the data into a decision tree

* The results of the rpart() function will be stored in a variable called MyTree.
* The formula for a decision tree model is outcome ~ predictor1 + predictor2 + etc.
* TargetBuy is the outcome 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.**

|  |
| --- |
| **Spreading code across multiple lines**  As you can see, you can span across multiple lines. when spreading code over multiple lines, you have to make sure that R knows something is coming, either by:   * leaving a bracket open, or * ending the line with an operator   For example, when you enter the following command in RStudio’s console:  MyTree <- rpart(TargetBuy ~ Affl + Age + NeighborhoodClusterGroup + Gender + GeoReg + TVReg +  LoyaltyStat + TotalSpend + LoyaltyTenure, data=trainingSet, method="class",  control=rpart.control(minsplit=MINIMUMSPLIT, cp=COMPLEXITYFACTOR))  R considers the first line of the command as “not finished” and will continue to read the next time, because the first line ends with an operator “+”. |

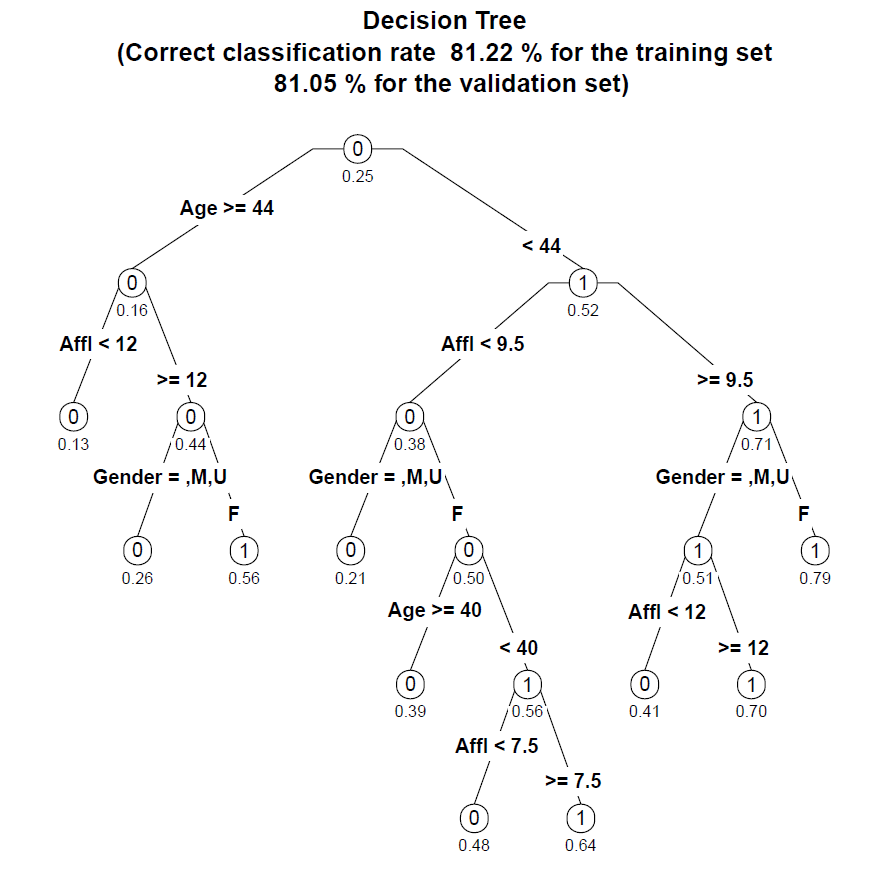
# Part 3: Execute the dTree.r Script

1. Set working directory. Select Session/Set Working Directory/To Source File Location to change the working directory to the location of your R script.
2. Run the script. 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!)

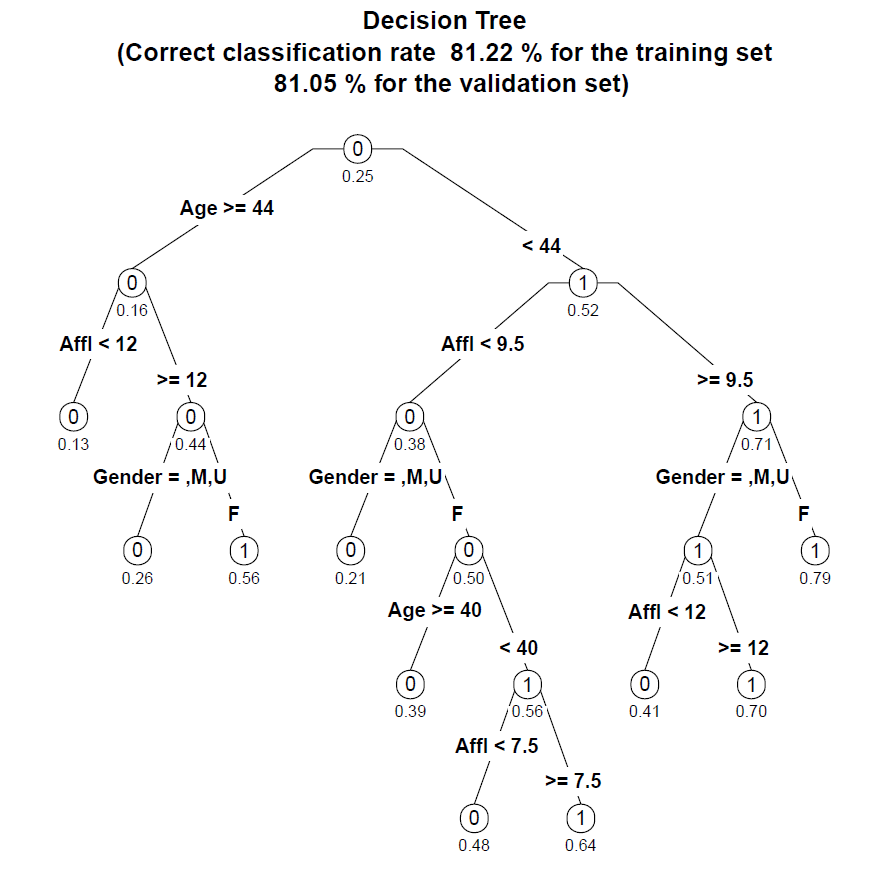
You’ll see a lot of action in the Console window at the bottom left side of the screen, ending with this:  
  
> dev.off()

RStudioGD

2

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

(It may be difficult to read the decision tree plot – the text is quite small. Open the folder of 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 file and go back to RStudio!)



**(Tree #1)**

# Part 4: Read the Decision Tree

The tree has **ten** *leaf nodes* (nodes with nothing beneath them). Each of those ten leaf nodes represents a prediction based on a combination of predictor variables.

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

The tree correctly predicts whether someone will buy organics 81.22% of the time for the training set, and 81.05% of the time for the validation set.

**Probabilities of a positive outcome (the numbers/values under the nodes)**

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

**Predicted outcome (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 can read the left branch of the tree like this:**

*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.*

*Males 44 years or older with an affluence grade of 12 or more will buy organic products only 26% of the time.*

*Females who are 44 years or older and have an affluence grade of 12 or more will buy organic products 56% of the time.*

Note: Variables can appear twice within a branch of the tree if it further differentiates between outcomes -- Note that Affluence 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.

**Confusion Matrix**

If you open the *DecisionTreeOutput.txt* file, you can find the detailed confusion matrices for both the training set and the validation set (see below). Based on the confusion matrices, you can also compute the correct classification rates on your own.

###### Confusion Matrix for the training set ######

> table(Predicted=predTraining,Observed=trainingSet[, OUTCOME\_COL] )

Observed

Predicted 0 1

0 7859 1575

1 512 1166

> cat("\n###### Confusion Matrix for the validation set ######\n")

###### Confusion Matrix for the validation set ######

> table(Predicted=predValidation,Observed=validationSet[, OUTCOME\_COL] )

Observed

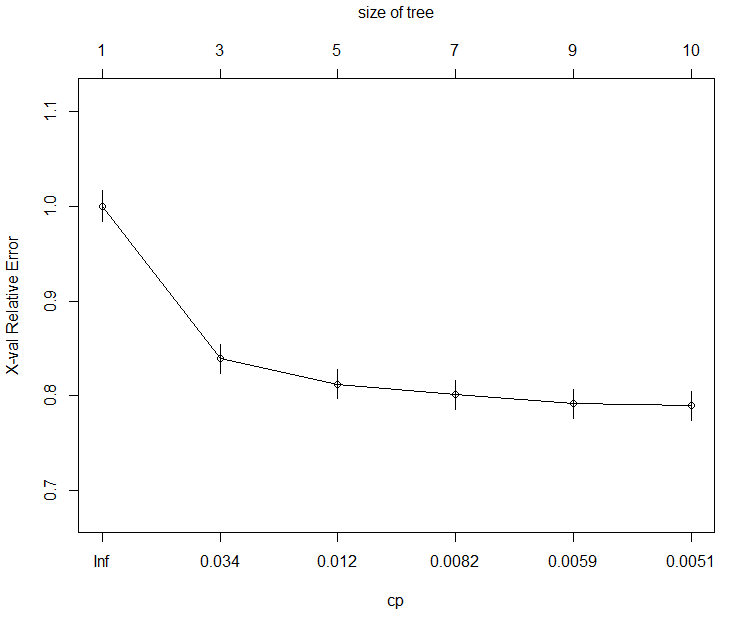
Predicted 0 1

0 7825 1583

1 522 1181

# Part 5: Tree Size and Relative Error

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. (When there is only 1 node, the X-val relative error is scaled to 1.0).

All this is saying is that the tree gets a lot more accurate (the X-val relative error 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 10 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 *relative to* 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.

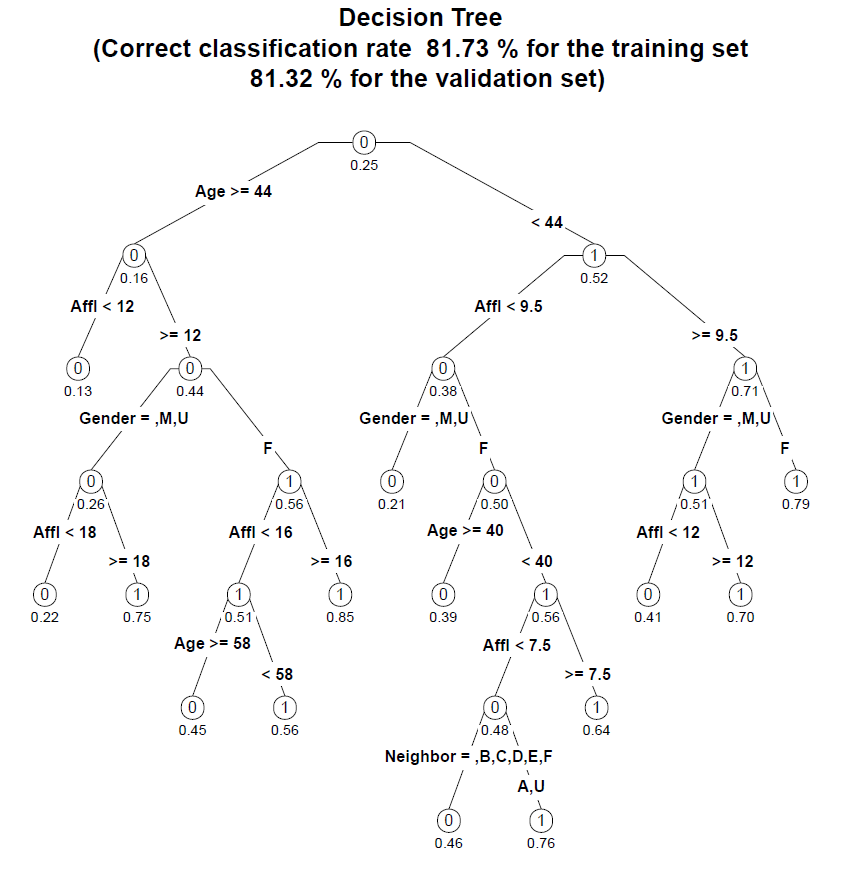
That’s why the first tree on this plot has one node (i.e., everyone buys) and a relative error rate of 1.0 (it’s the baseline).

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

# Part 6: Change the Complexity Factor

## What if you use a smaller COMPLEXITYFACTOR?

1. *Go to line 27 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.
2. Run the script by selecting Code/Run Region/Run All. You’ll now see this decision tree.



**(Tree #2)**

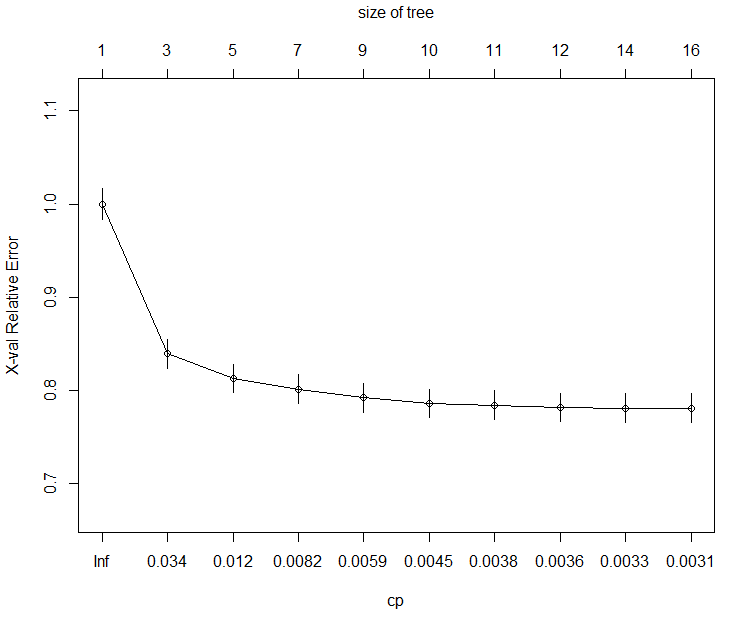
**Question 2:**

Based on this tree, 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 14 leaf nodes now instead of 10), and that the correct classification rate has risen from 81.22% to 81.73% for the validation set.

That is, **smaller COMPLEXITYFACTOR → more complex tree & higher accuracy.**

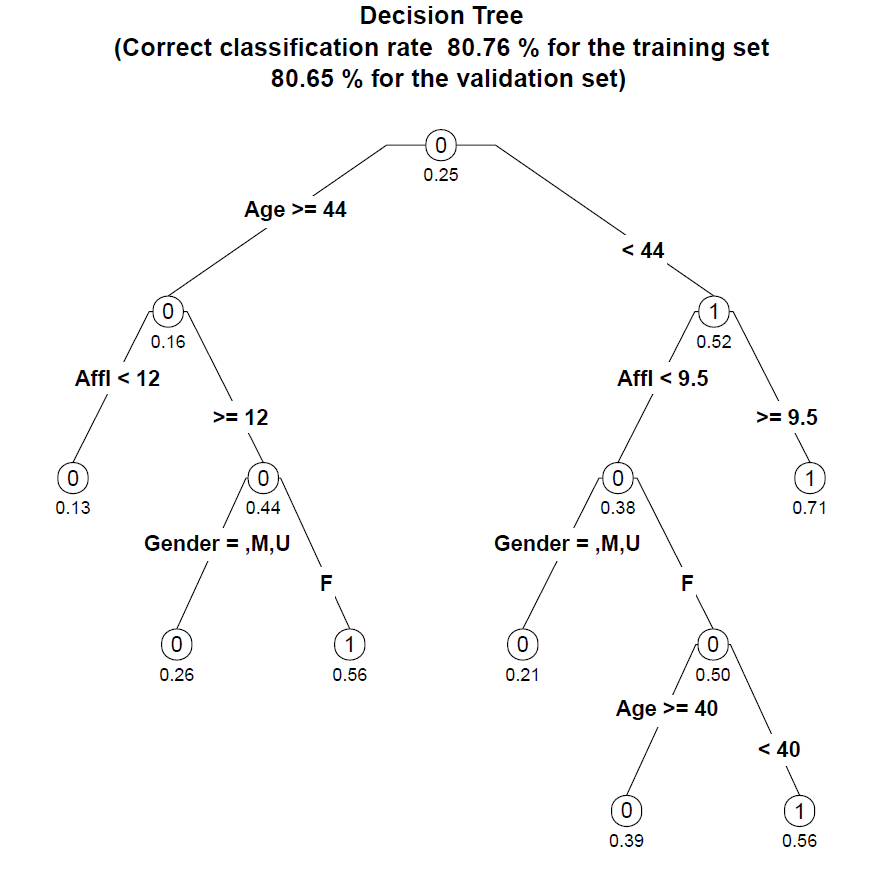
We can also see view complexity parameter plot by clicking the back arrow () :



Compared with the previous complexity parameter plot, this plot is has the same cp values for sizes from 1 to 10 nodes. For example, for the 3-nodes case, the cp value is 0.034 in both plots. However, this plot has more complex trees (with 14 leaf nodes). The reason is that, with smaller COMPLEXITYFACTOR, a smaller reduction in error is required to add more nodes.

## Now, what if you use a larger COMPLEXITYFACTOR?

1. *Go to line 27 and change COMPLEXITYFACTOR from 0.003 to 0.01.* 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 (i.e. larger reduction in error).
2. Run the script by selecting Code/Run Region/Run All. You’ll now see this decision tree.



**(Tree #3)**

This new tree has 7 leaf nodes, as opposed to our first tree with 10 nodes and our second tree with 14 nodes. Increasing the complexity factor threshold makes the tree simpler, and also a little less accurate (80.76% versus 81.22% and 81.73% of the previous trees for the validation set).

That is, **larger COMPLEXITYFACTOR → less complex tree & lower accuracy.**

# Part 7: Which tree do we use?

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tree #** | **On page** | **COMPLEXITY FACTOR** | **# of Nodes** | **Correct Classification Rate**  **(Training Set)** | **Correct Classification Rate**  **(Validation Set)** |
| 1 | 4 | 0.005 | 10 | 81.22% | 81.05% |
| 2 | 7 | 0.003 | 14 | 81.73% (best) | 81.32% (best) |
| 3 | 9 | 0.01 | 7 | 80.76% (worst) | 80.65% (worst) |

* We can see that Tree #2 has the highest correct classification rate (best).
* We can also see that Tree #3 has the lowest correct classification rate (worst).
* Trees #1 is somewhere in-between.

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 third) 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 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** can alter the decision tree.

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