MIS2502: Data Analytics
Decision Tree Induction Using R

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

Download both files and save them to the folder where you keep your R files.

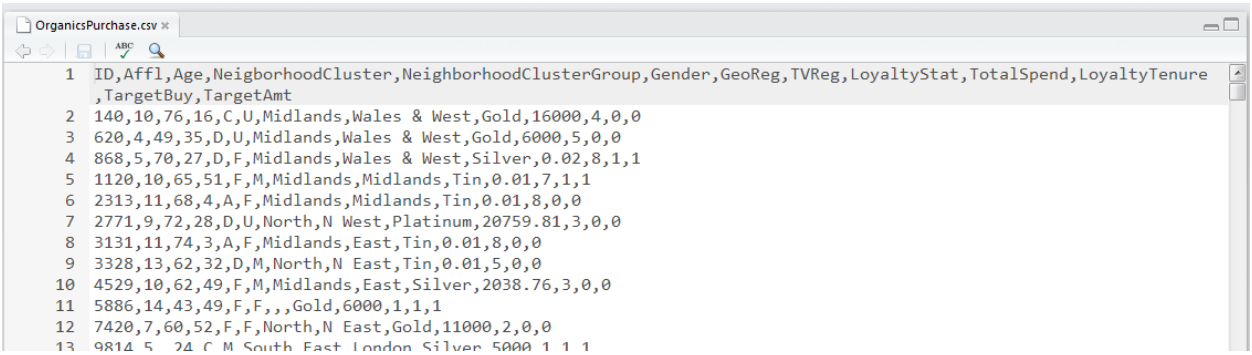
Also make sure you are connected to the Internet when you do this exercise!

Part 1: Look at the Data File

1) Start RStudio.

2) Open the OrganicsPurchase.csv data file. If it warns you that it's a big file, that's ok. Just click "Yes."

3) You'll see something like this:



This is the raw data for our analysis. This is a comma-separated file (CSV). That just means that each data value is separated by a comma.

What if you are starting with an Excel spreadsheet? You can save it as a CSV file by selecting File/Save As… and choosing "CSV (Comma delimited) (.csv)" from the "Save as type" menu.*

You can also open a CSV file directly in Excel. It will figure out what to do with the commas and create a nice spreadsheet for you. You can edit the file in Excel and the save it as a CSV.

Now look at the contents of the file. The first line contains the names of the fields (think of them like columns in a spreadsheet). You can see the first field is called ID, the second field is called Affl, the third 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.

[1] Adapted from SAS Enterprise Miner data set.

A full list of the variables is on the next page:

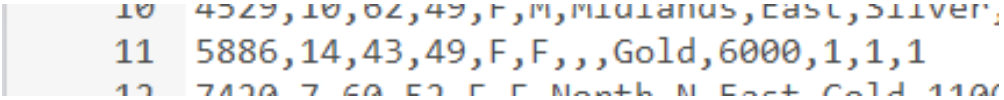
Variable Name	Variable Description
ID	Customer loyalty identification number
Affl	Affluence grade on a scale from 1 to 30 (1 = least affluent, 30 = most affluent)
Age	Age, in years
NeighborhoodCluster	Identifier for residential neighborhood
NeighborhoodClusterGroup	A set of similar neighborhoods (A-F = 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

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.).

TargetBuy is a typical outcome variable because it describes a discrete, dichotomous event (1 = did buy, 0 = did not buy).

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).

4) Now look at line 11 of the file:



You'll see GeoReg and TVReg have no values – we know this because there is nothing in-between the commas (Gender is F (female) and LoyaltyStat is Gold). In order to keep the data in the right columns, missing data still needs to be separated by commas.

Variable Name	Variable Description
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

5) Close the OrganicsPurchase.csv file by selecting File/Close. 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. 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.
   - Lines that start with a "#" symbol should be grey.
   - String values in between quotes should be green.
   - Numeric values should be orange.

   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).

   The code is heavily commented. If you want to understand how the code works line-by-line you can read the comments. For the purposes of this exercise (and this course), we're going to assume that it works and just adjust what we need to in order to perform our analysis.

2) Look at lines 8 through 27. These contain the parameters for the decision tree model. Here's a rundown:

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.
OUTCOME_COL	12	The outcome variable (TargetBuy) is in column 12 of the OrganicsPurchase dataset.

3) Lines 33 through 40 load the three packages necessary for decision tree analysis – rpart, caret, and rpart.plot.

4) Now let's look at the decision tree model. Scroll down to lines 76 through 78:

```
MyTree <- rpart(TargetBuy ~ Affl + Age + NeighborhoodClusterGroup + Gender + GeoReg + TVReg +
                LoyaltyStat + TotalSpend + LoyaltyTenure, data=trainingSet, method="class",
                control=rpart.control(minsplit=MINIMUMSPLIT, cp=COMPLEXITYFACTOR));
```

You can see a few things at work:

- The `rpart()` function is use to classify the data into a decision tree (in this case, called MyTree).
- The formula for a decision tree model is `outcome ~ predictor1 + predictor 2 + etc.`
- `TargetBuy` is the outcome event you're trying to predict (i.e., will they buy organics or not?).
- All nine variables to the right of the ~ are used to predict the outcome. 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) Save the dTree.r file – you've made some changes in the previous section.

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

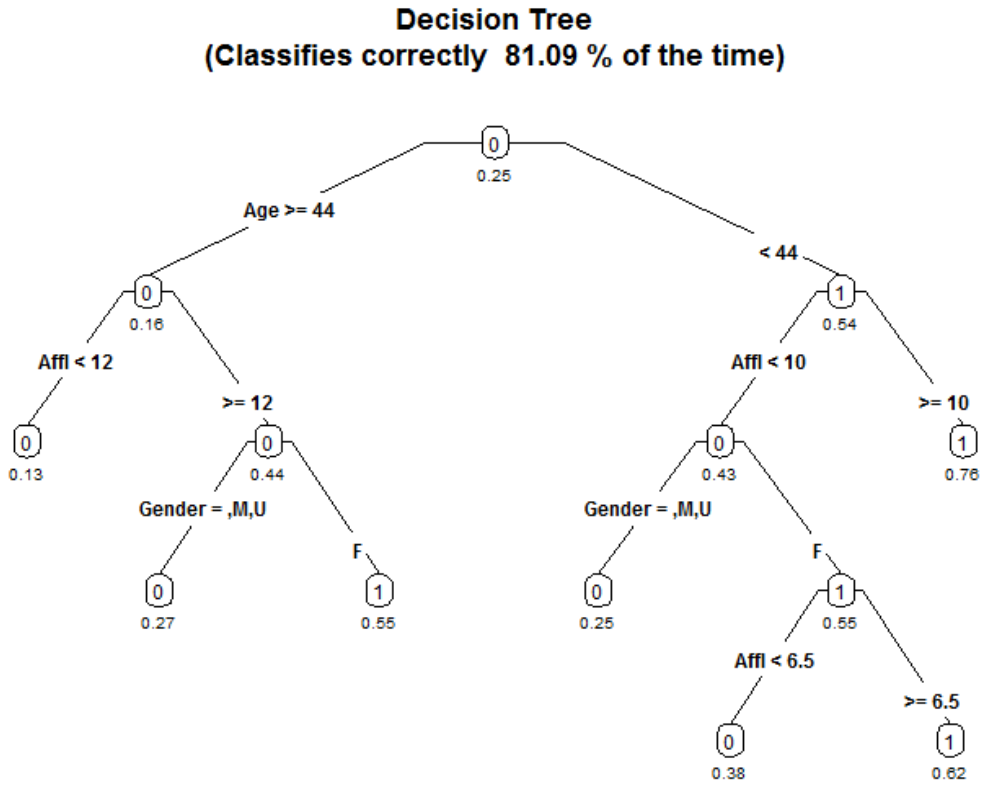
3) Select Code/Run Region/Run All. It could take a few seconds to run since the first time it has to install some extra modules to do the analysis. Be patient!

4) 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(Classifies correctly ",round(predRate,2)*100,"% of the time)"),
+     type=4, extra=6, faclen=0, under= .... [TRUNCATED]

> dev.off();
RStudioGD
        2
>
```

5) And you'll see the decision tree on the right:



The tree will correctly predict whether someone will buy organics 81.09% of the time. The tree has seven leaf nodes (nodes with nothing beneath them). Each of those seven leaf nodes represent a prediction based on a combination of predictor variables.

The number under the node is the probability of a **positive** outcome – in this case, that they will buy organic products. The "0" and "1" labels inside the node simply indicates whether there are more positive or negative outcomes 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.

Age is the best predictor, as it is the first split. It creates the most differentiation 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 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.

Try it:

Based on the tree you've generated, how likely is it that the following customers will buy organic products (the answers are on the last page):

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, aren't in the decision tree. This is because the decision tree algorithm determined they didn't contribute enough beyond Age, Affl, and Gender to meaningfully differentiate between buyers and non-buyers.

To see some proof of this, open the DecisionTreeOutput.txt file in your working directory. If you double-click on the file, it will open in Notepad. Scroll down until you see "Variable Importance":

```
Variable importance
  Age   Affl Gender
   59     30     11
```

This is generated by the summary(MyTree) statement. It shows that only Age, Affl, and Gender were included in the model. The relative importance of each of those is indicated by its number. Notice they all add up to 100. Basically, this means Age (59) is about twice as important as Affl (30) and about 5.3 times as important as Gender (11) in predicting the outcome (buy organic products).

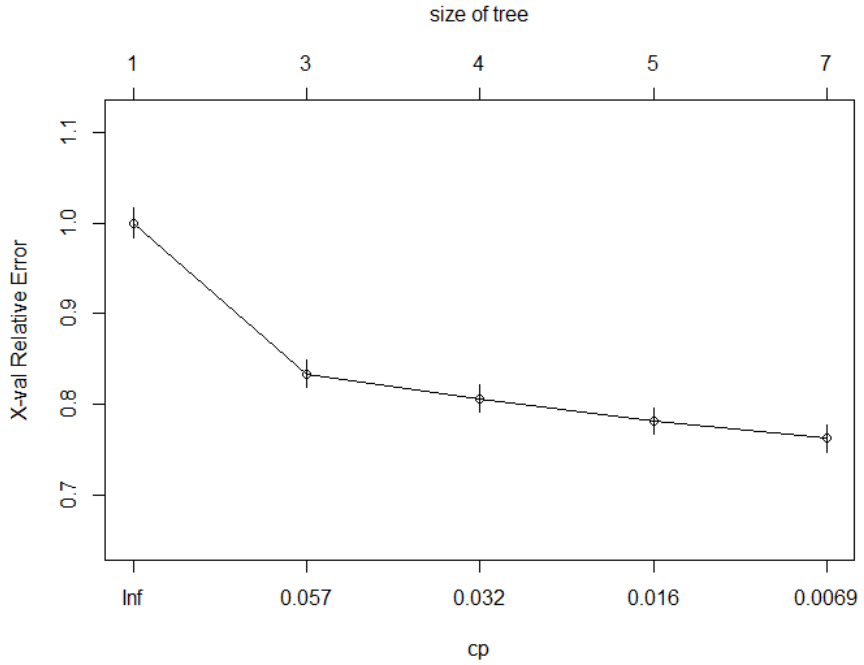
Now look right under those two lines. You'll see the analysis of node 1 (the root node):

```
Node number 1: 11112 observations,    complexity param=0.08371763
  predicted class=0  expected loss=0.2536897  P(node) =1
    class counts:  8293  2819
   probabilities: 0.746 0.254
  left son=2 (8312 obs) right son=3 (2800 obs)
  Primary splits:
      Age        < 44.5     to the right, improve=617.89290, (762 missing)
      Affl       < 12.5     to the left,  improve=347.03130, (550 missing)
      Gender     splits as  LRLL,         improve=251.63400, (0 missing)
      TotalSpend  < 1694.725 to the right, improve= 67.82093, (0 missing)
      LoyaltyStat splits as  LLLR,        improve= 48.55094, (0 missing)
```

There is a big drop off in "improve" after Gender (251.6 for Gender, but 67.8 for Total spend). This means that after Gender, the other variables do much less to improve the model. So they get left out.

If our threshold for "contribution" was lower, then we might see other variables in our tree. This is adjusted using the COMPLEXITYFACTOR variable, which we will do in a few steps.

6) Now click the back arrow ( ) in the plot window 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.

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 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 correct classification rate.

The correct classification rate measures how often the decision tree made the right prediction (i.e,. buy or not buy).

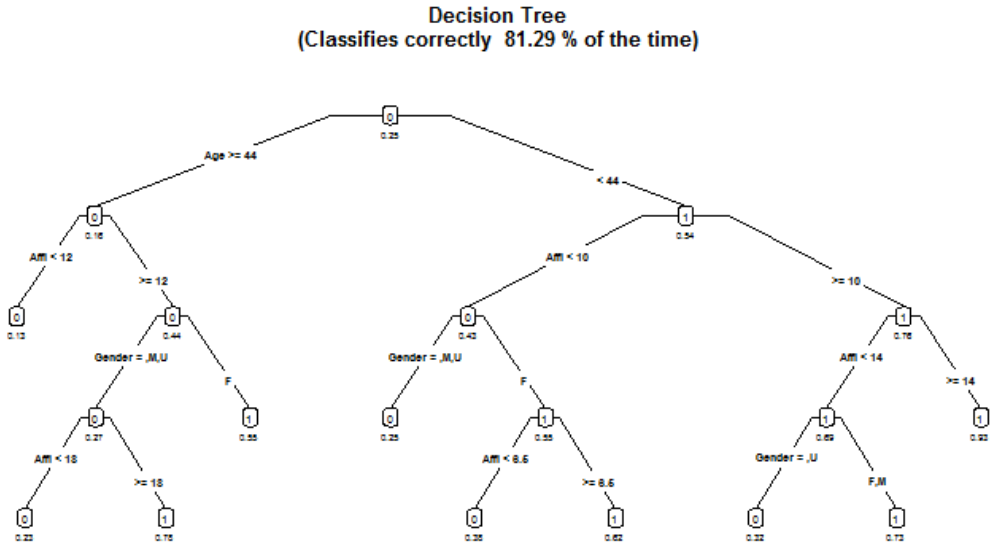
The relative error rate is how much a tree with n-nodes improves the decision over a tree that just puts everyone in the same category – in this case that baseline tree would classify everyone in the data set as buying organics.

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).

7) Now click the "Clear all" button in the plot window (the broom) to clear the plots you've generated.

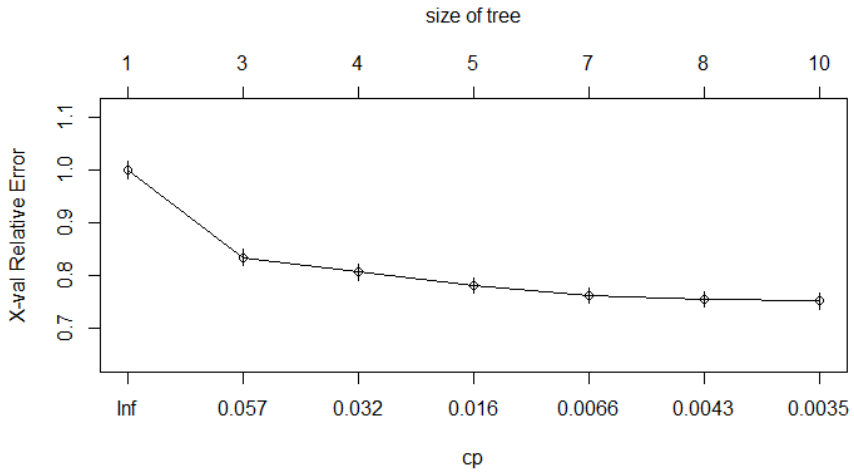
8) Go to line 26 and change COMPLEXITYFACTOR from 0.005 to 0.003. This means a smaller incremental improvement in the tree is necessary to make the decision to add an additional node. In other words, we're willing to put up with a more complex tree if it helps our result.

9) Run the script by selecting Code/Run Region/Run All. You'll now see this decision tree:

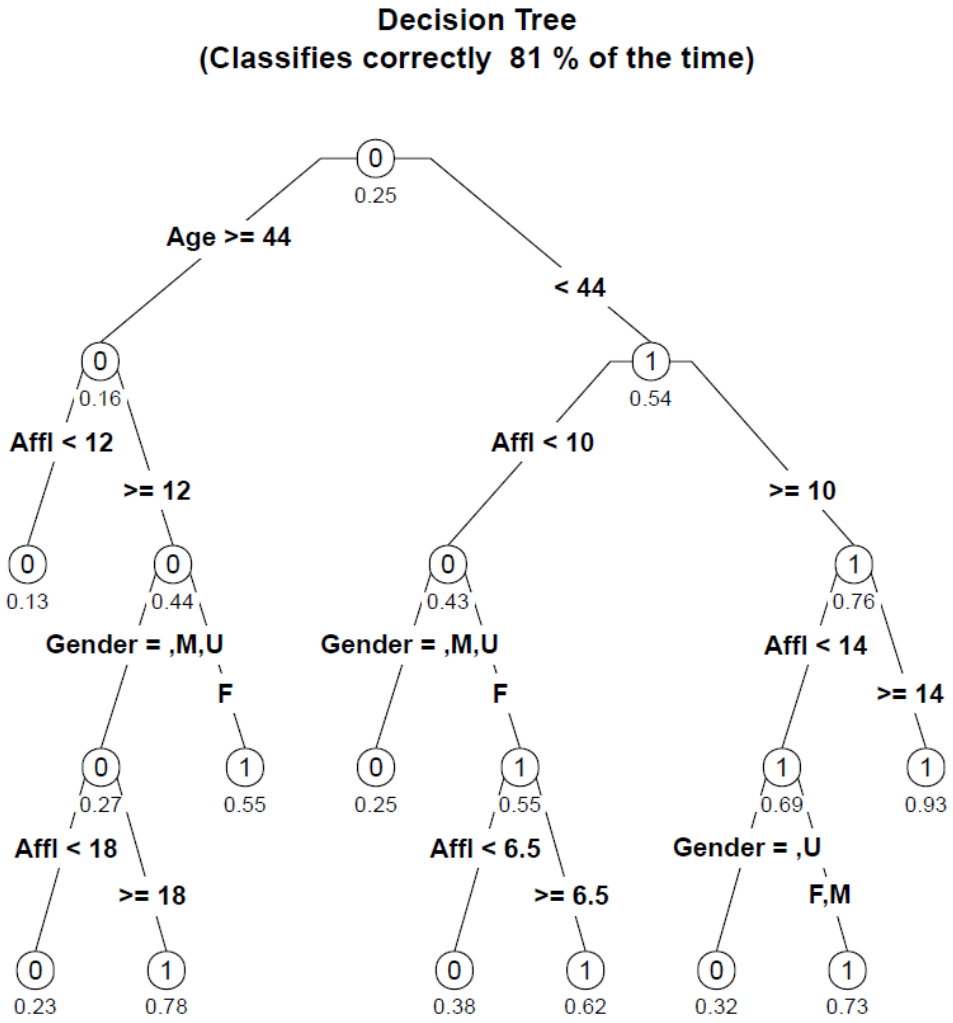


We can clearly see the tree is more complex (it has 10 leaf nodes now instead of 7), and that the correct classification rate has risen from 81.09% to 81.29%.

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



10) It's 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 (the one with your R script in it). You should see a file called TreeOutput.pdf. Open the file and you will see this:



Try it:

Based on this tree, how likely is it that the following customers will buy organic products (the answers are on the last page):

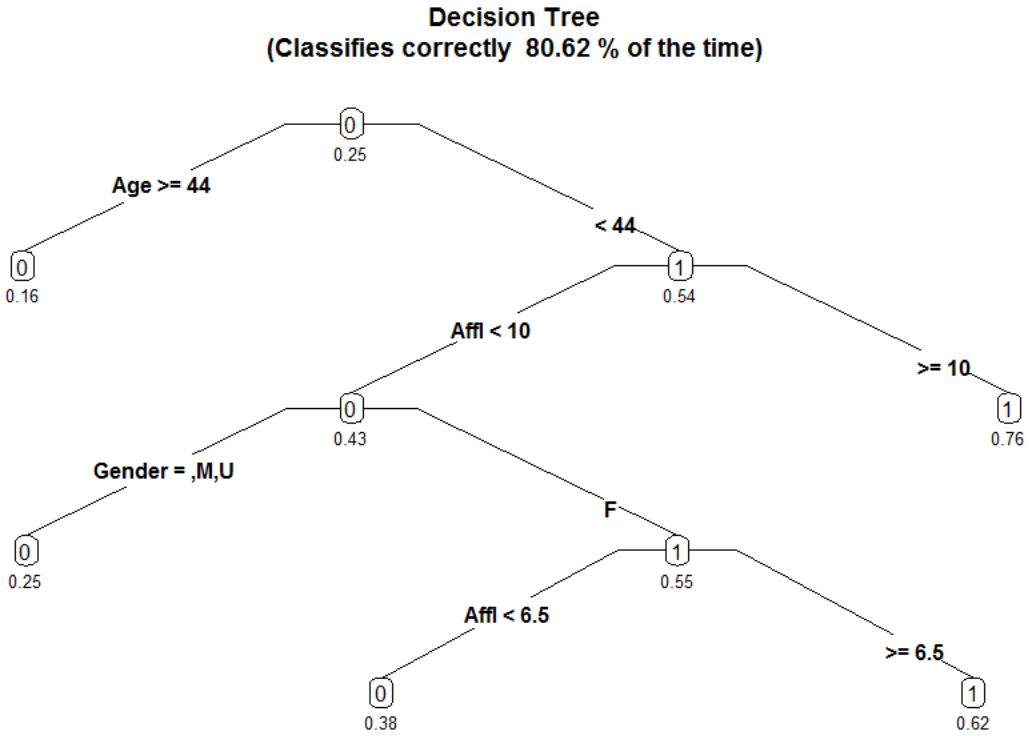
A 25 year-old male with an affluence grade of 4.5?
A 65 year-old female with an affluence grade of 20?

Describe the characteristics of the most likely and least likely groups to buy organic products?

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

11) 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).

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 4: Change the minimum split

1) Change COMPLEXITYFACTOR back to 0.005 (line 26). Re-run the script to verify that we're back to our tree with seven nodes.

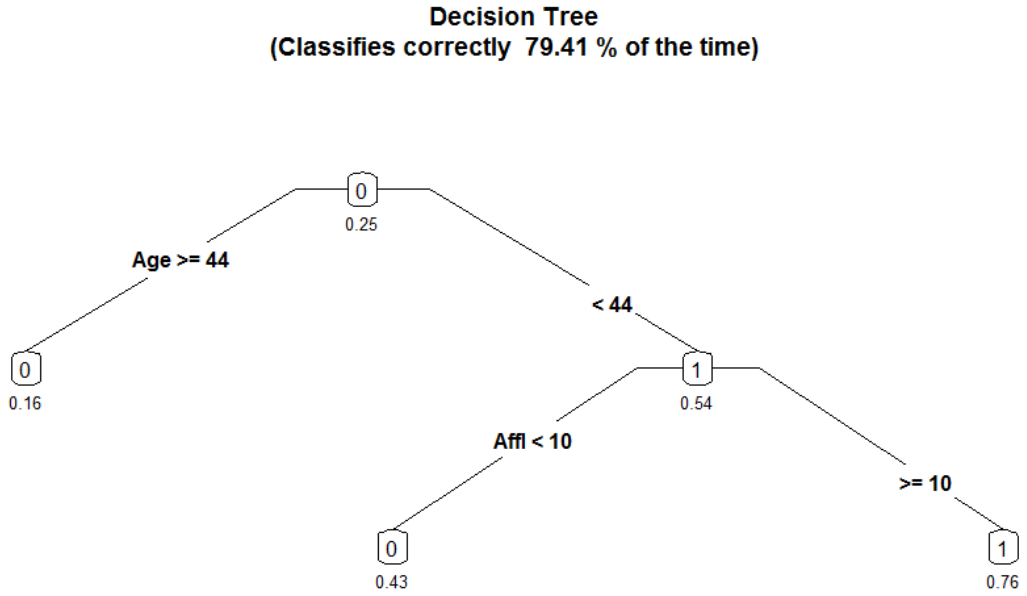
2) Change MINIMUMSPLIT from 50 to 2000. This means that each node has to have at least 2000 observations in it.

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 complexity of the tree. A

larger number prevents the algorithm from creating nodes that only describe very specific demographic groups.

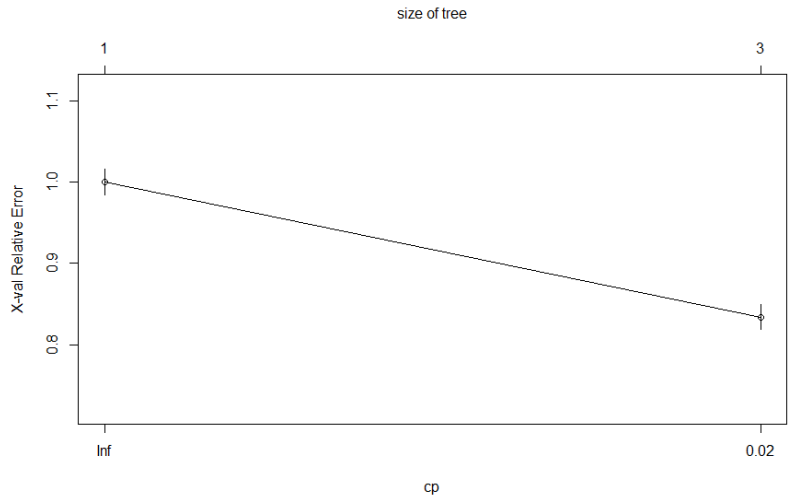
For example, if there were only 100 customers over 50 years old with an affluence grade less than 5 it would not give them their own node. Instead, it would combine them with another (similar) group of people when creating the tree.

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%. Plus, 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 5; 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 error rate was 0.763.

Part 5: 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	Relative Error
1	6	0.005	50	7	81.09%	0.762
2	8	0.003	50	10	81.29%	0.752
3	10	0.01	50	5	80.62%	0.782
4	11	0.005	2000	3	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.

Answer Key

Try it page 6:

25 year-old male with an affluence grade of 4.5	0.25 (25%)
30 year-old male with an affluence grade of 15	0.76 (76%)
30 year-old female with an affluence grade of 15	0.76 (76%)
43 year-old female with an affluence grade of 8	0.62 (62%)

Try it page 9:

25 year-old male with an affluence grade of 4.5	0.25 (25%)
65 year-old female with an affluence grade of 20	0.55 (55%)

Most likely to buy:

- Less than 44 years old with an affluence grade greater than 14 (93% likely to buy organics). Gender doesn't matter.

Least likely to buy:

- 44 years old or older with an affluence grade less than 12 (13% likely to buy organics). Gender doesn't matter.