Jiahui Wu #2670210

Lab 4

Designing and Building prediction models for bike buyer data with a classifier.

Part1. Data preprocess and Split training and testing data

Part2. Decision tree using C50

Part3. Decision tree using CART

Part4. Comparison between C50 and CART

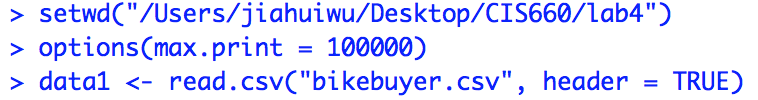
Part5. Feature selection for extra credit

Part 1. Preprocess data

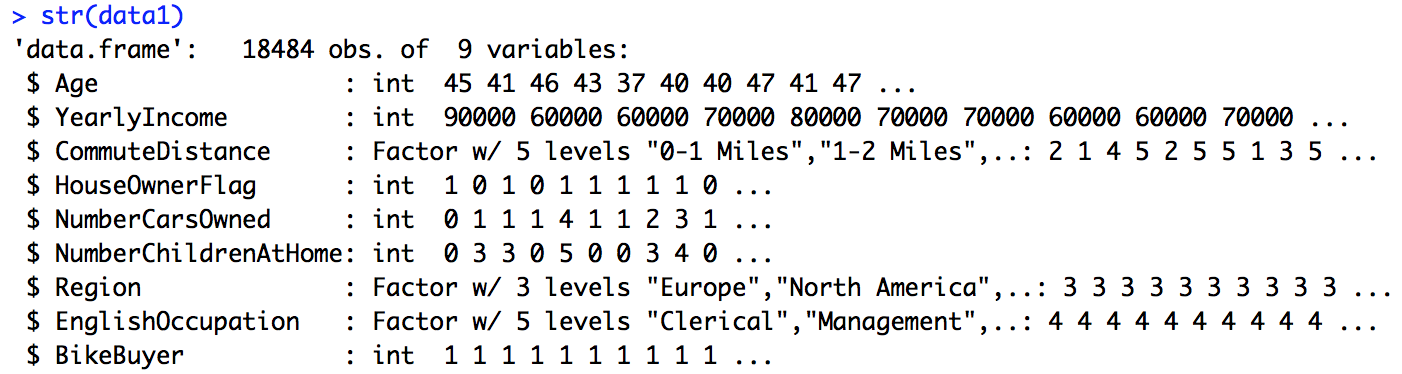
1-1. Data normalization and discretization

I did this step because the data from lab 1 was missing an important attribute: the class attribute BikeBuyer, so I retrieved data using with this attribute along with other 8 attributes and started processing.

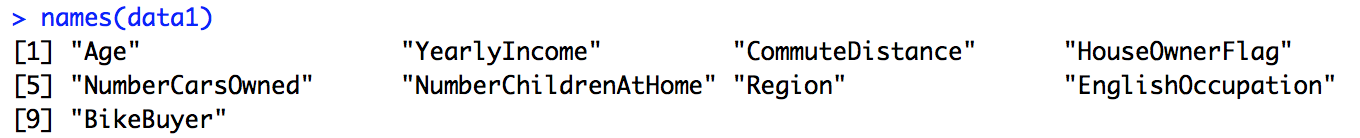
1-1-1. In R, set the working directory, set max print and read the .csv file.



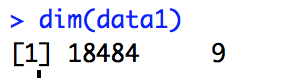
The structure of data1 is as following:



Attributes of data1

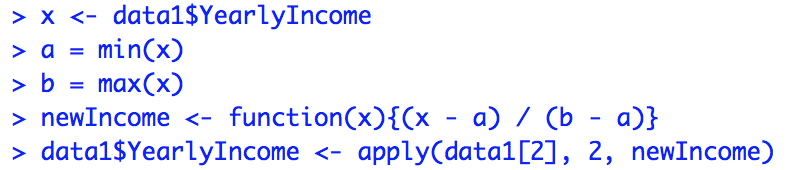


dimension of data1, there are 9 columns and 18484 rows



The following data preprocessing steps are from lab1.

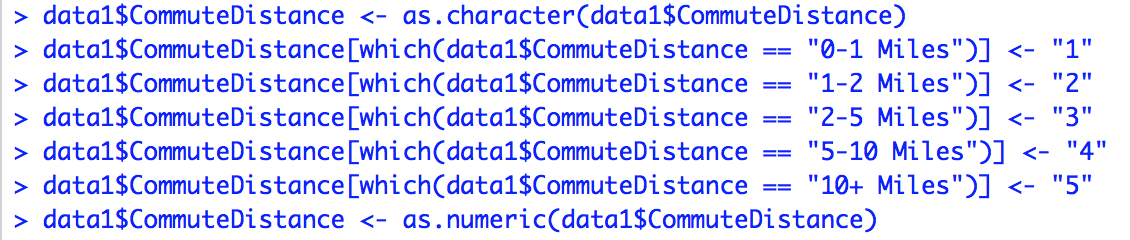
1-1-2. Using the R code in lab1 to normalize yearly income to range 0-1



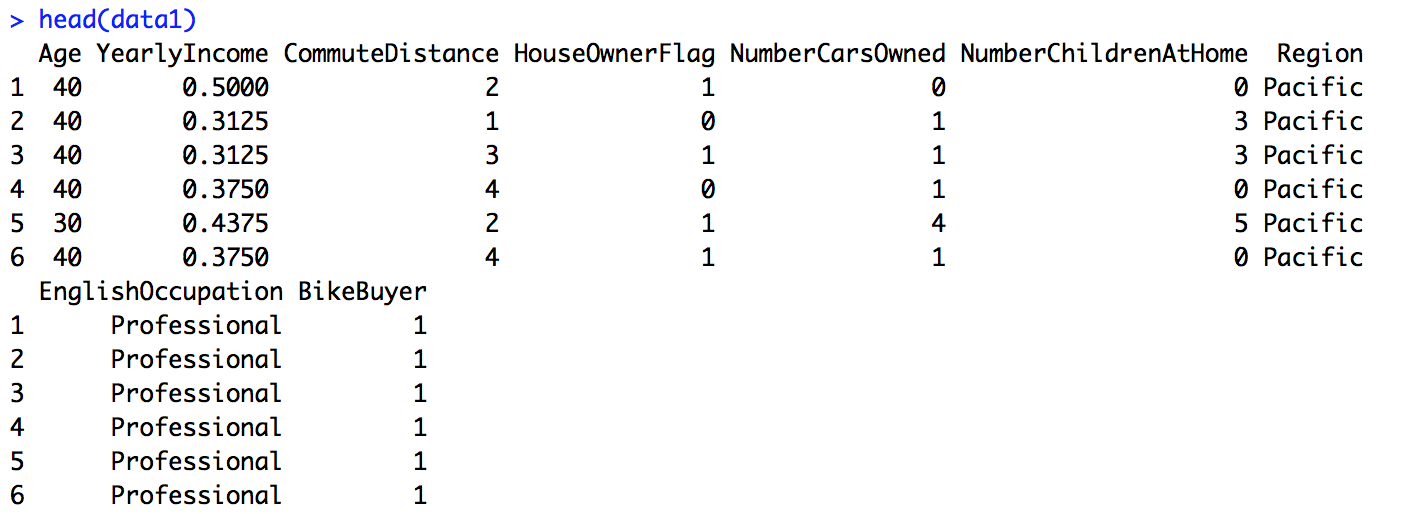
1-1-3. Discretization age, cut the age into different groups.



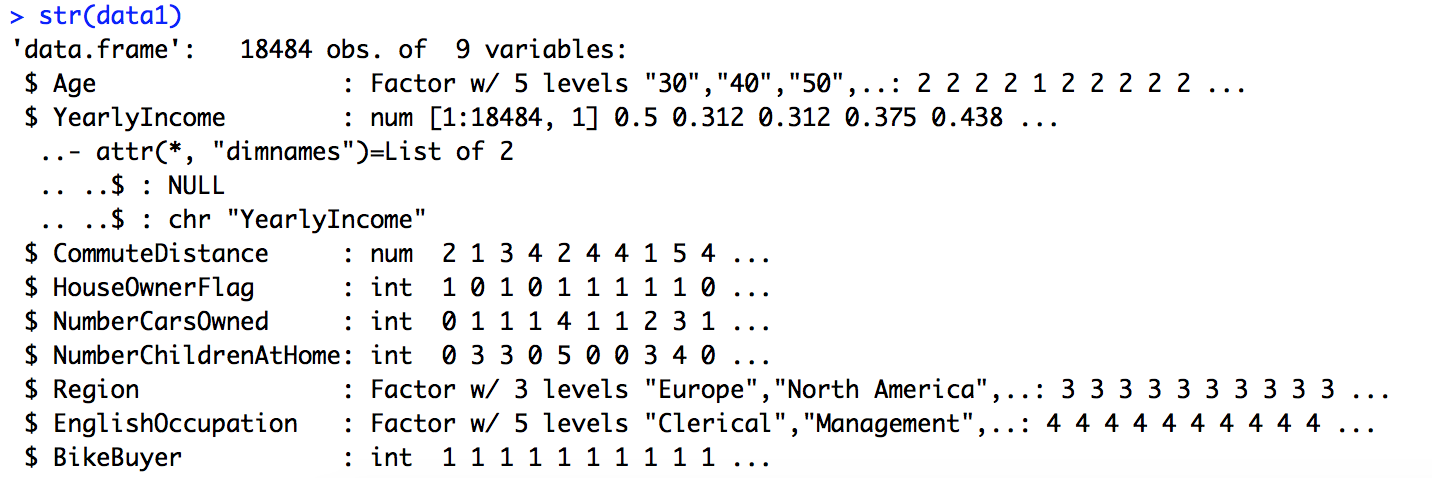
1-1-4. Replace ordinal attribute commuteDistance with numbers 1-5.



The first 6 row of data1 after normalization and disretization.

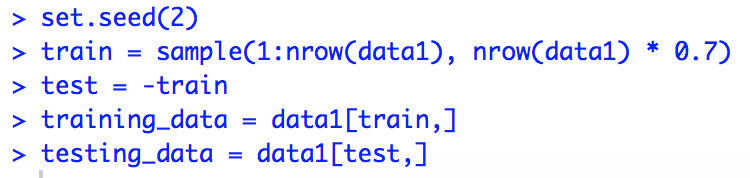


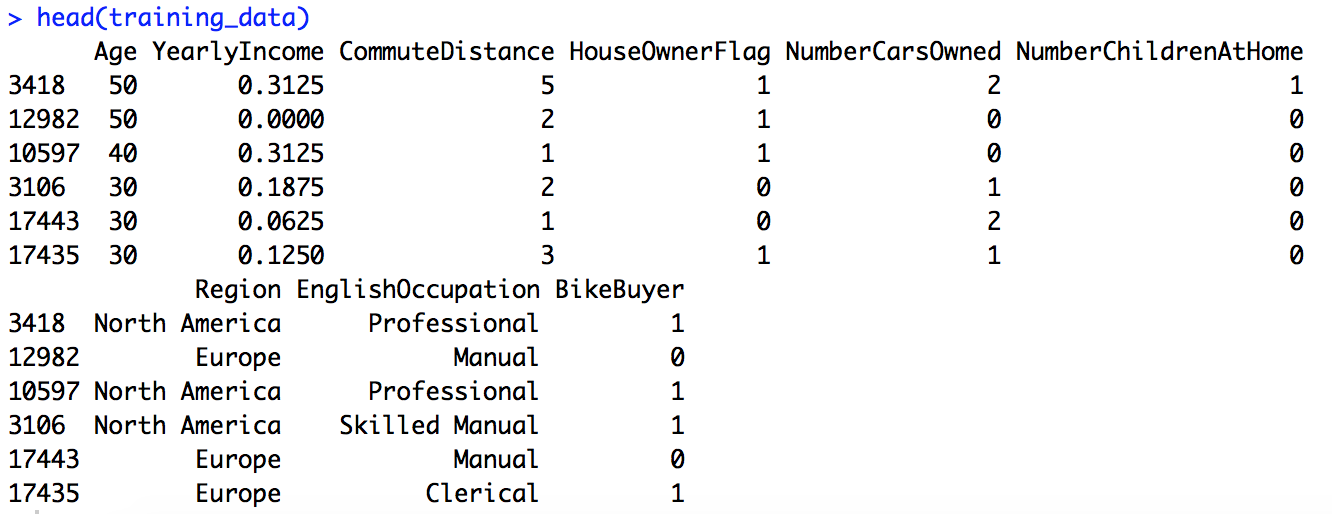
The structure of data1 now becomes the following:

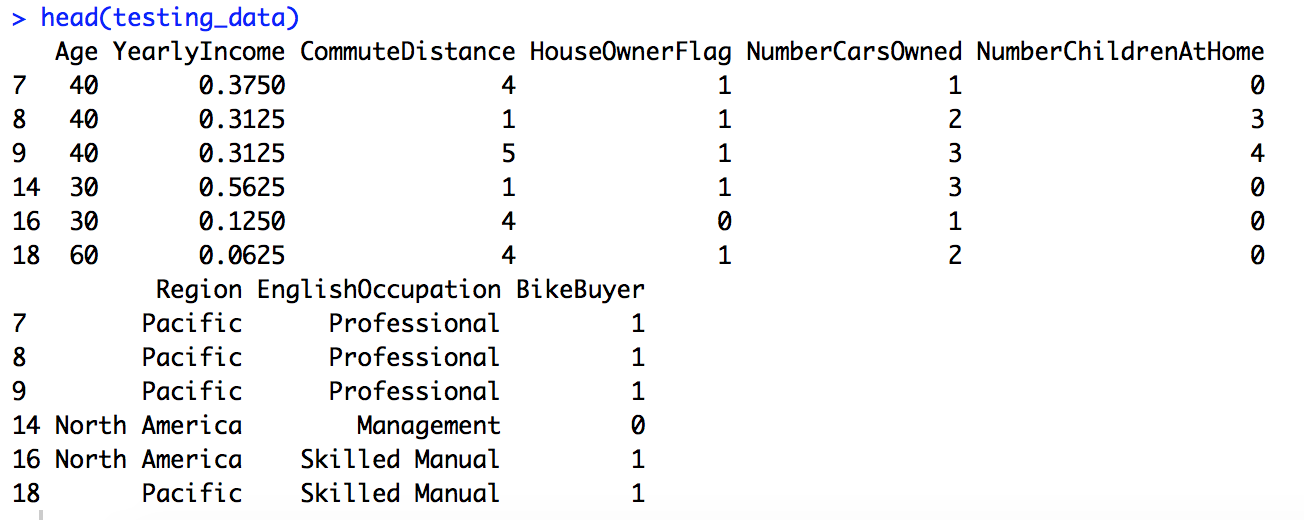


1-2. Split data into testing and training sets

To keep consistency, I use set.seed(2) to do sampling, I chose 70% of records as training data set and 30% records as testing dataset.

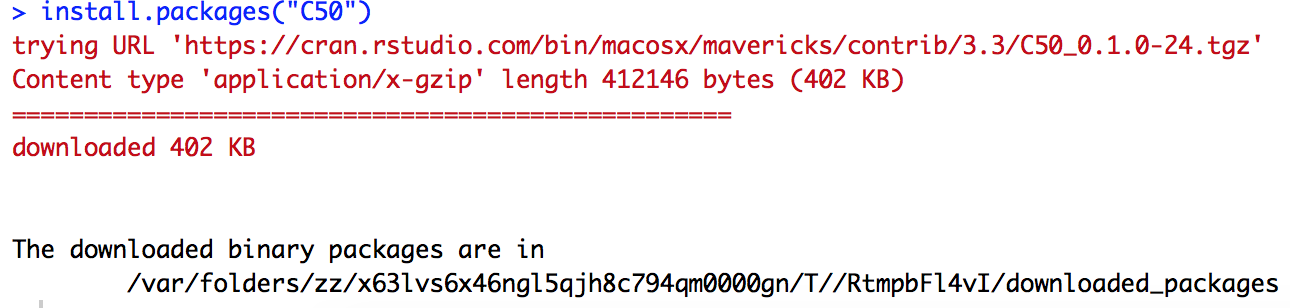






Part 2. Decision tree using C5.0

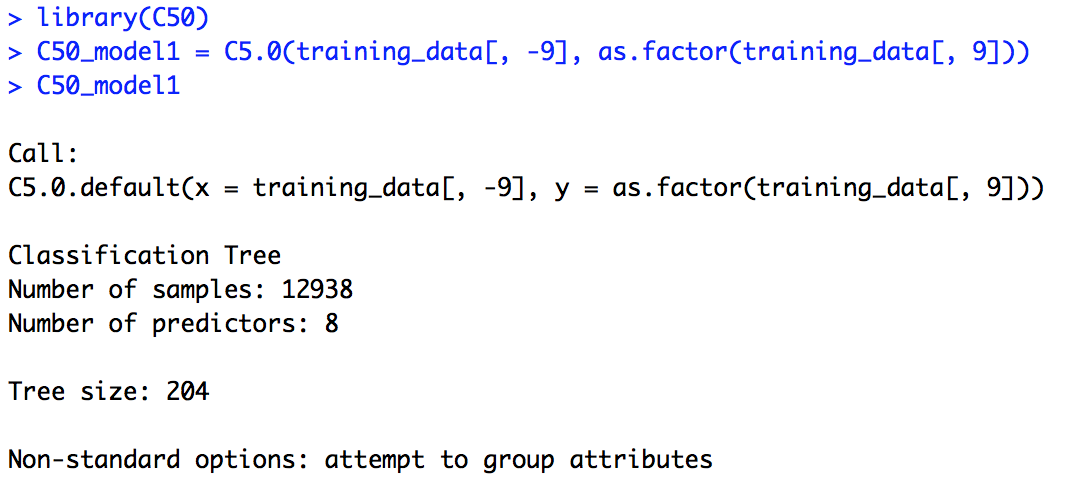
2-1. install package C50



2-2. Use parameter default settings to build decision tree.

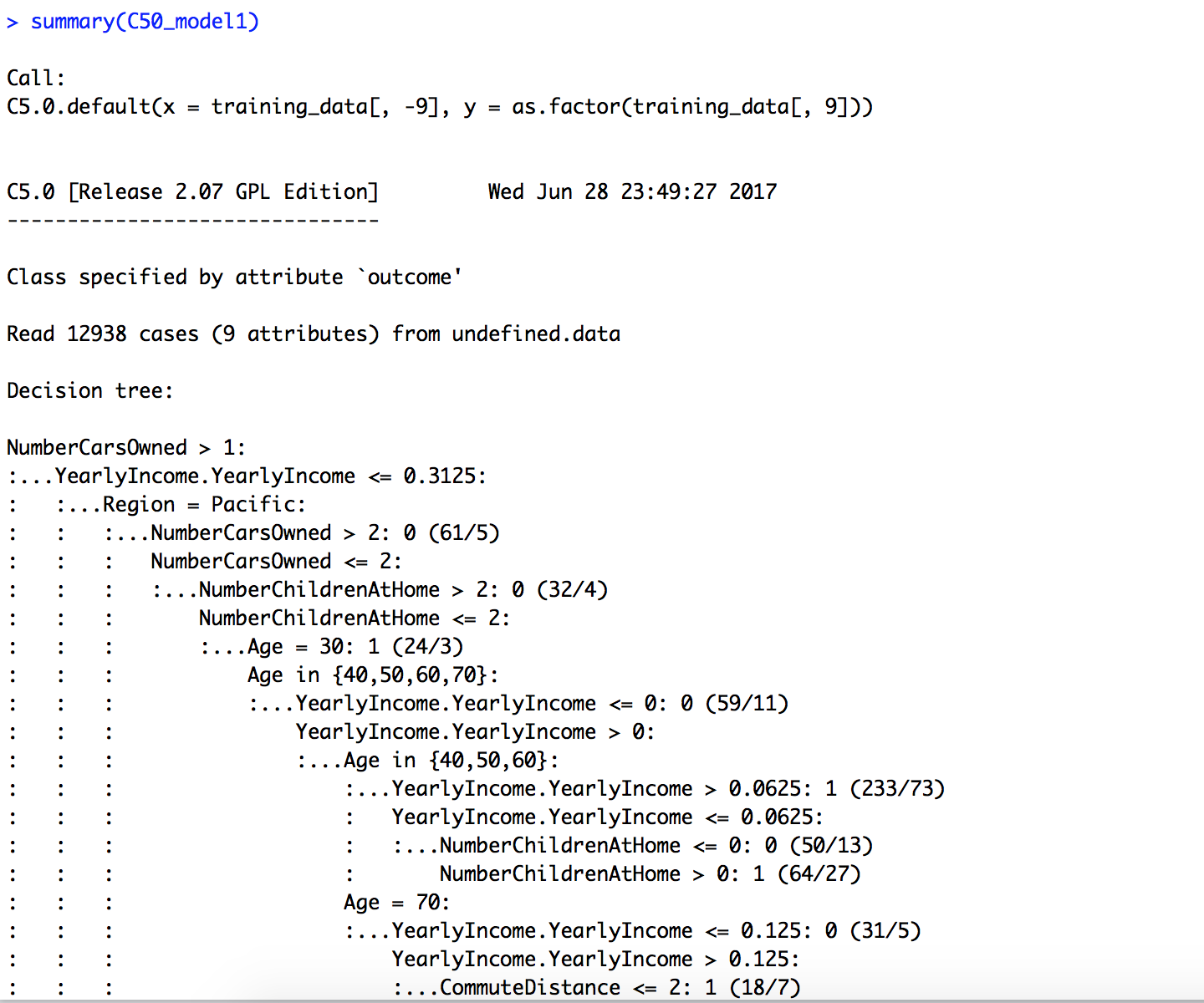
2-2-1. The first parameter in function C5.0() is training\_data[, -9], I eliminated the class attribute BikeBuyer, which is the 9th column. The first one is the training data potential predictors. The second parameter in function C5.0() is the target.

By type the model name we can see the sample volume, prediction number and tree size, etc. In my case, the sample size is 12938, which is 70% of the records, predictors are 8, and tree size is 204 (number of leaf nodes or terminal nodes).

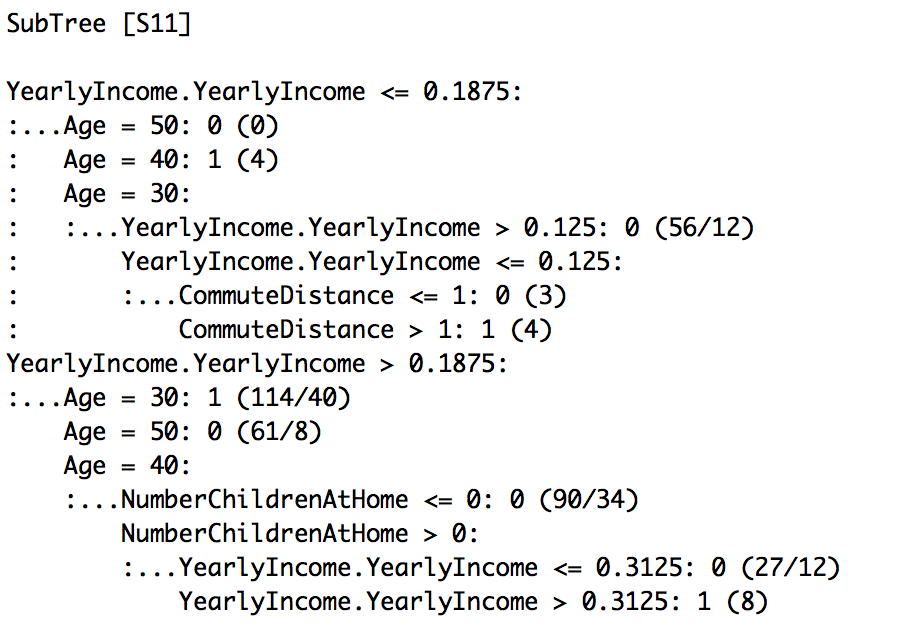


Using summary command, we can see a lot of more info.

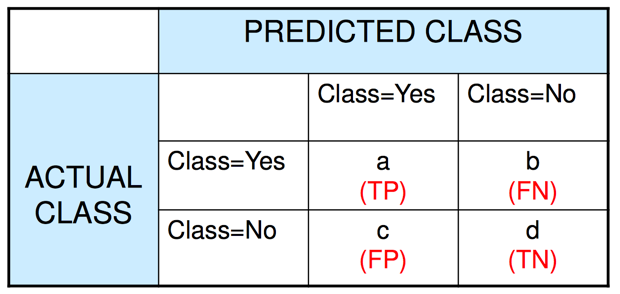
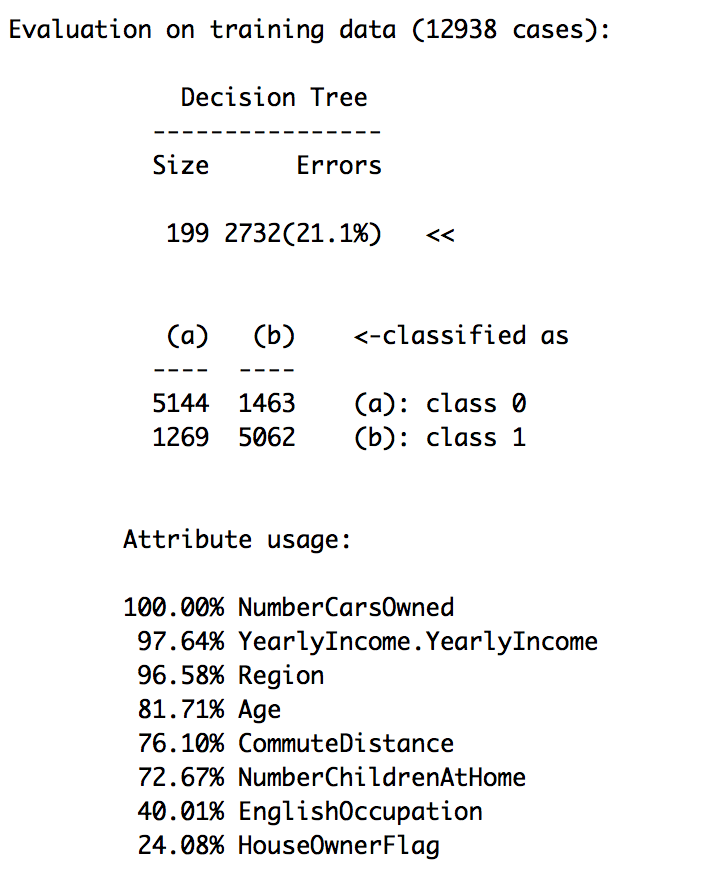
The following screenshot shows partial decision tree:



::::::::::::::::::::::::: The body of the decision tree is omitted.

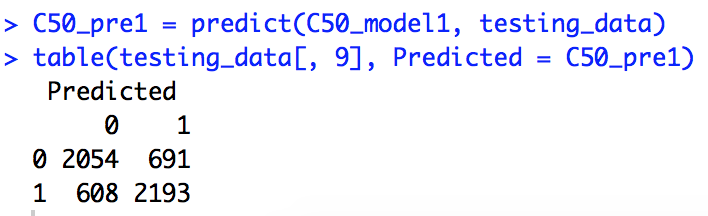


The following info tells me that the error of the classification is 21.1%, the confusion matrix shows that 5144 are TP, 5062 are TN, 1463 are FN and 1269 are FP. 2732, so we got errors = (1463 + 1269) / 12938 = 21.1 %. 12938 is the size of training set.



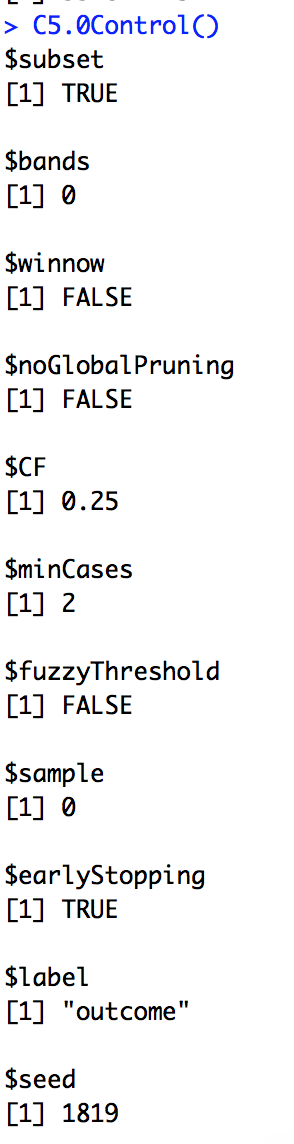
2-2-2. Next we want to see how well the model is doing on the testing data set.

I use predict() to do test the model and made a confusion matrix table. The actual bike buying data of test data was put on the horizontal axis of the table. Accuracy = (2054+2193) / 5546 = 76.58 %. 5546 is the size of testing data set.



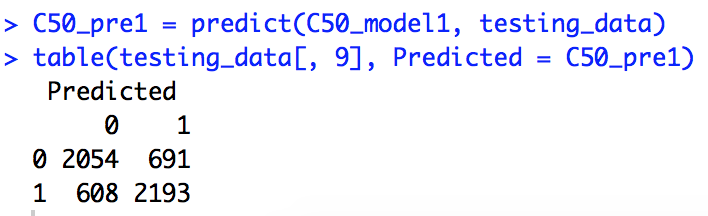
2-3. Change parameter settings of C50 and see how the accuracy changes.

C5.0Control() display all default parameters. I tried to change the parameters and see if accuracy improved.

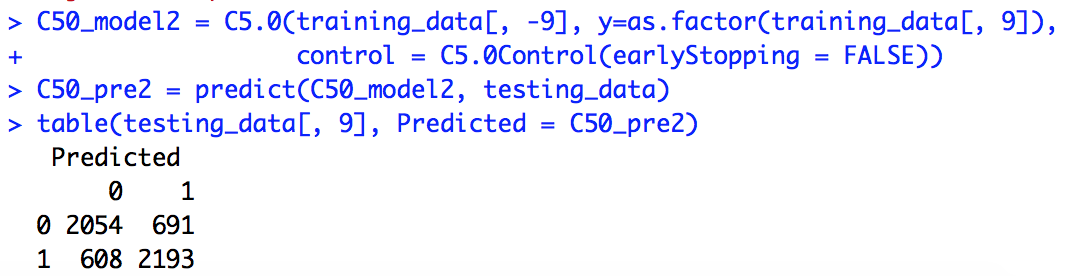


change parameters. The parameter setting principle is that, first change 1 parameter, if it improves the accuracy, keep the change, move on to next; otherwise, keep it default. From the following confusion table, no change of default setting improves accuracy. The default parameter settings give the best performance.

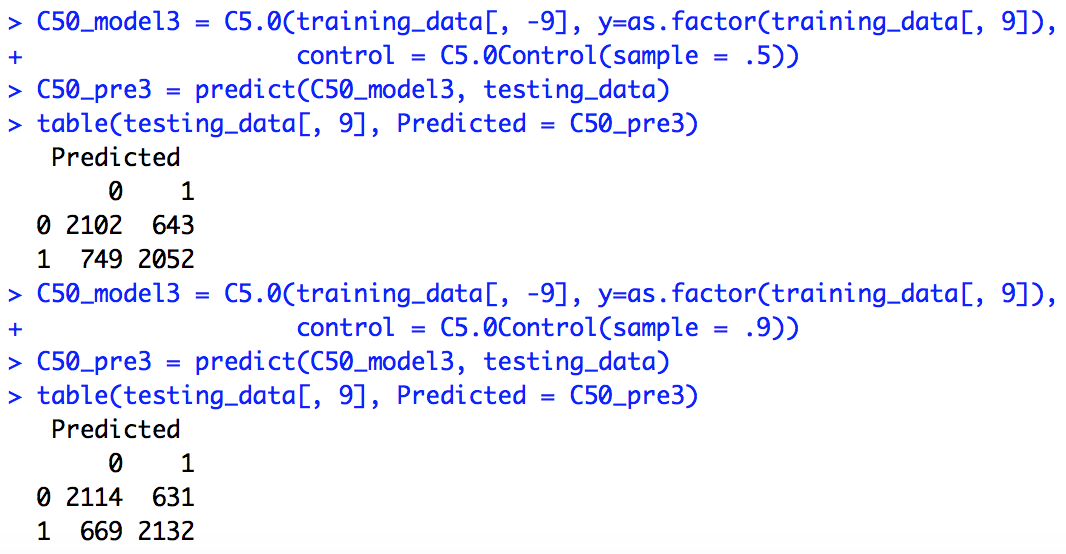
1)default



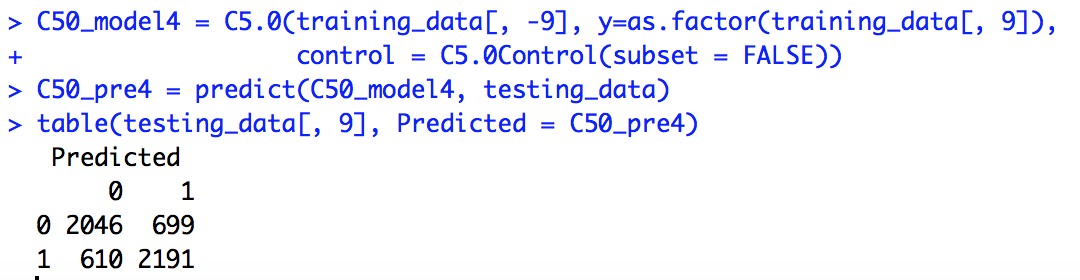
2)



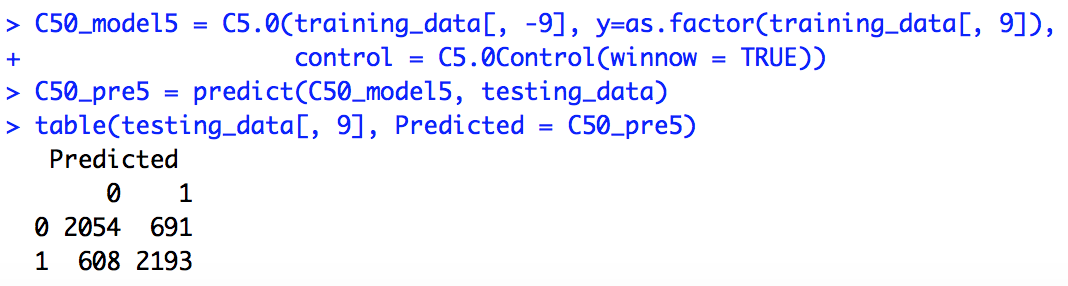
3)



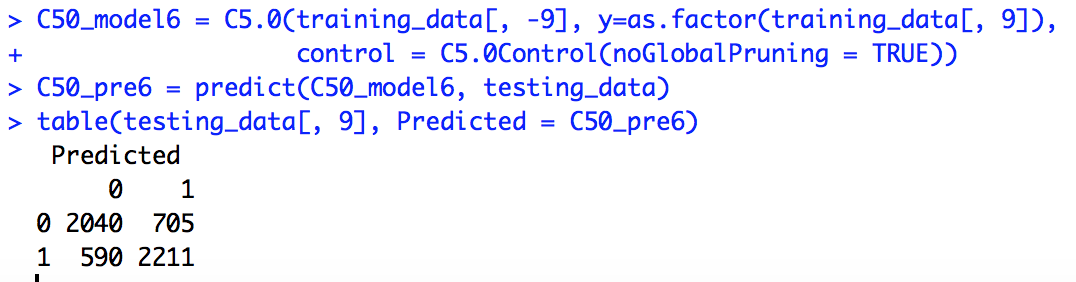
4)



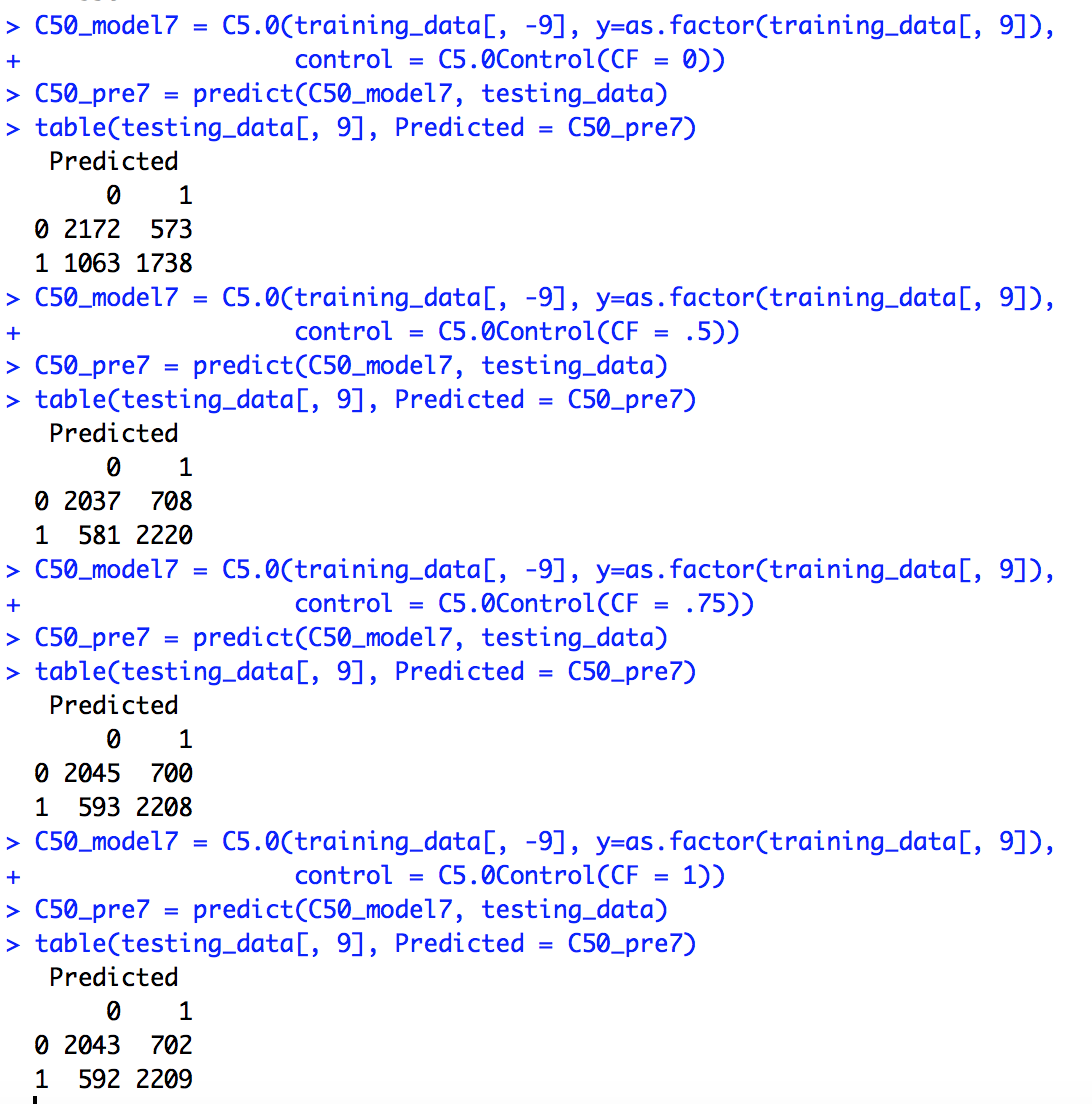
5)



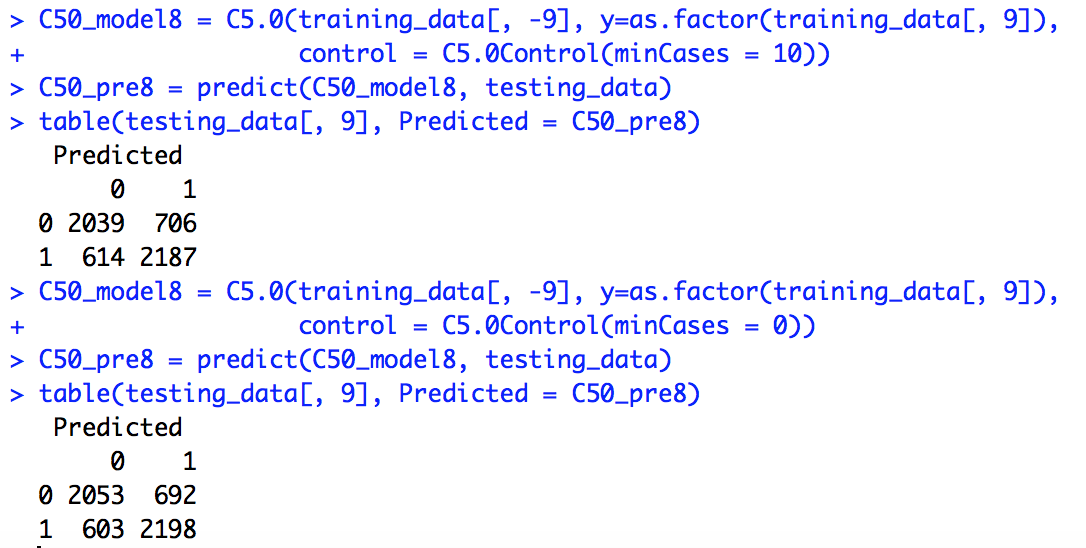
6)



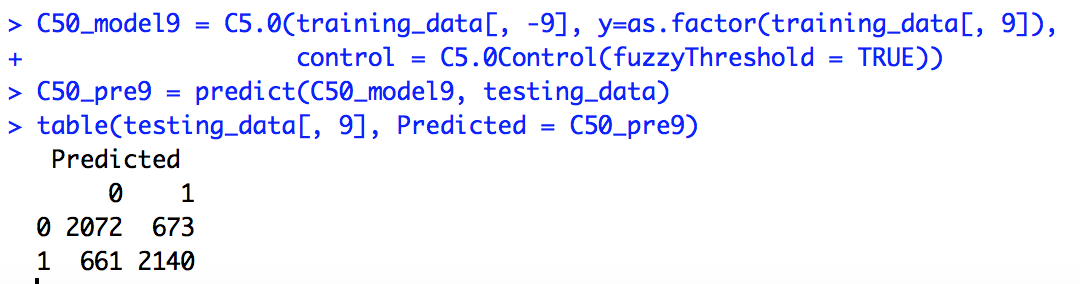
7)

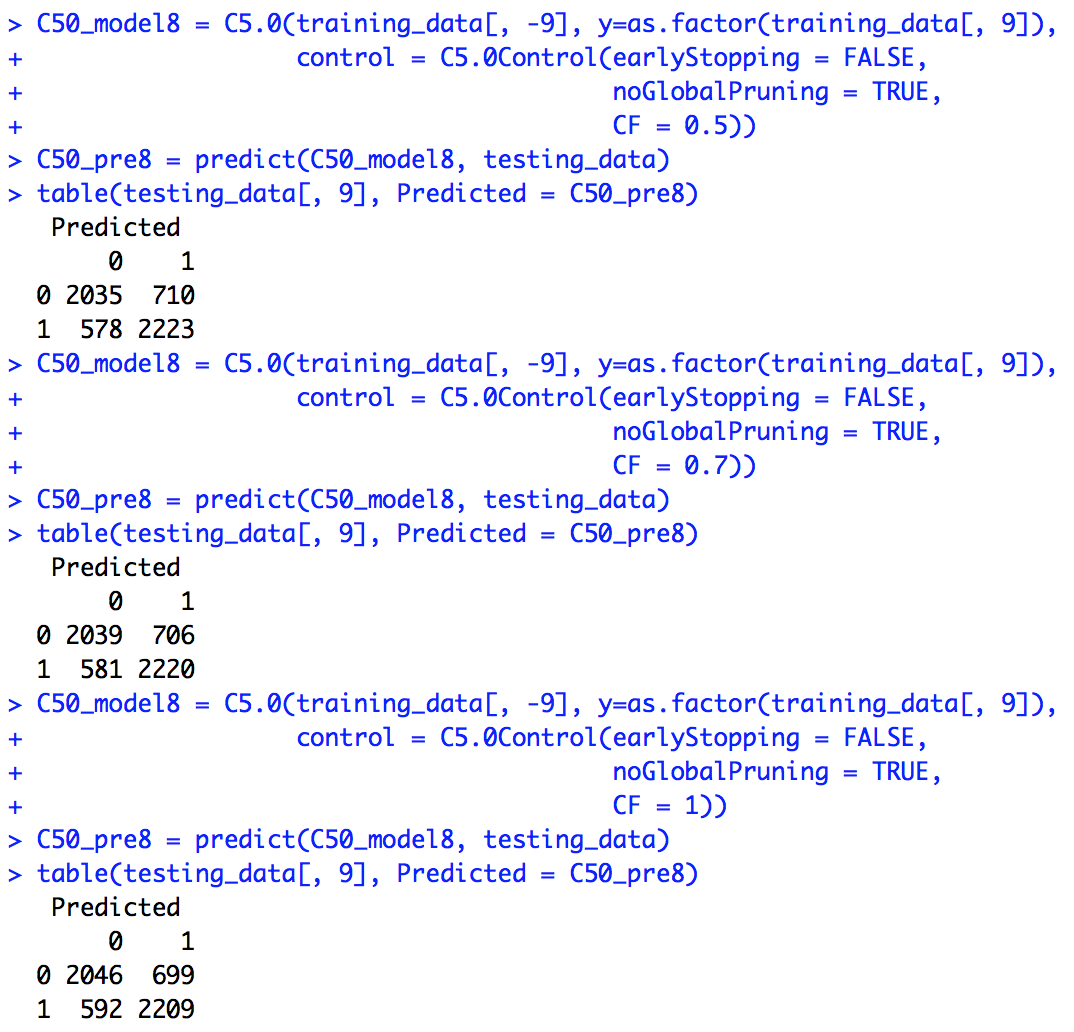


8)

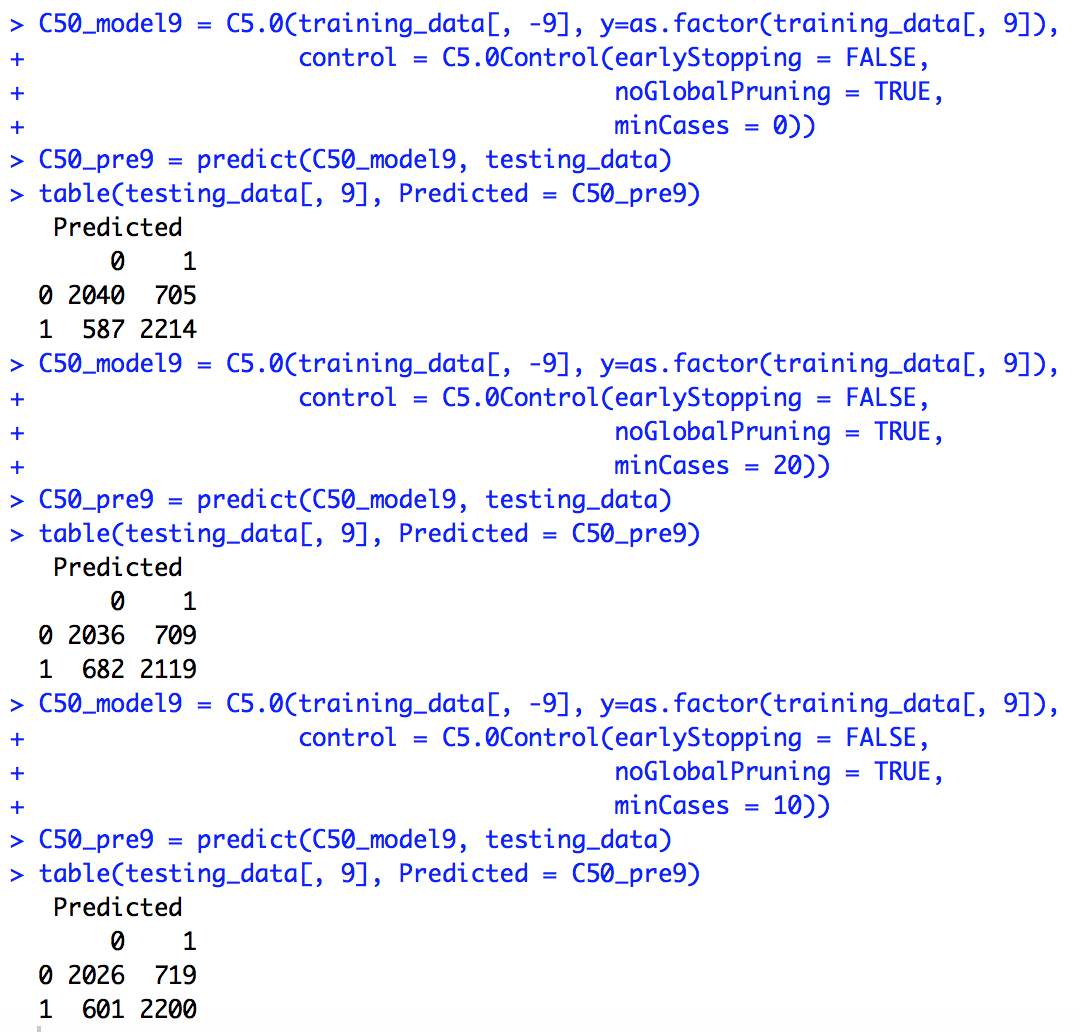


9)

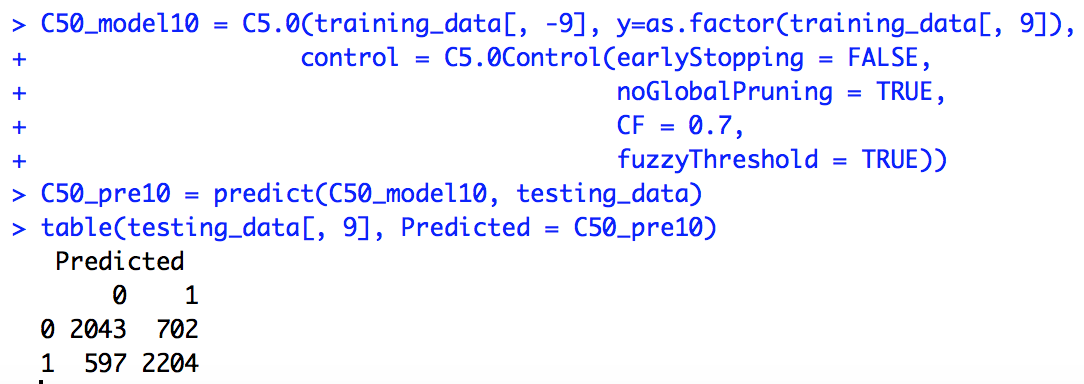




9)

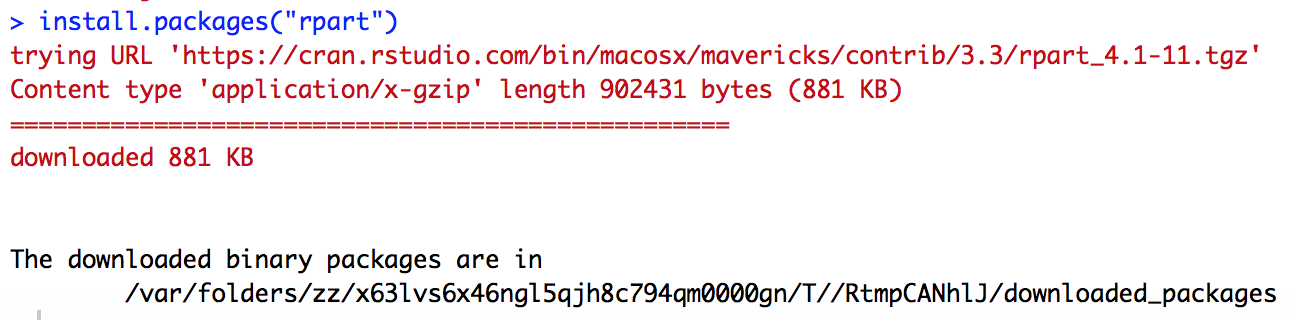


10)



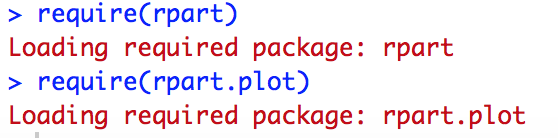
Part 3. Using rpart to do classification

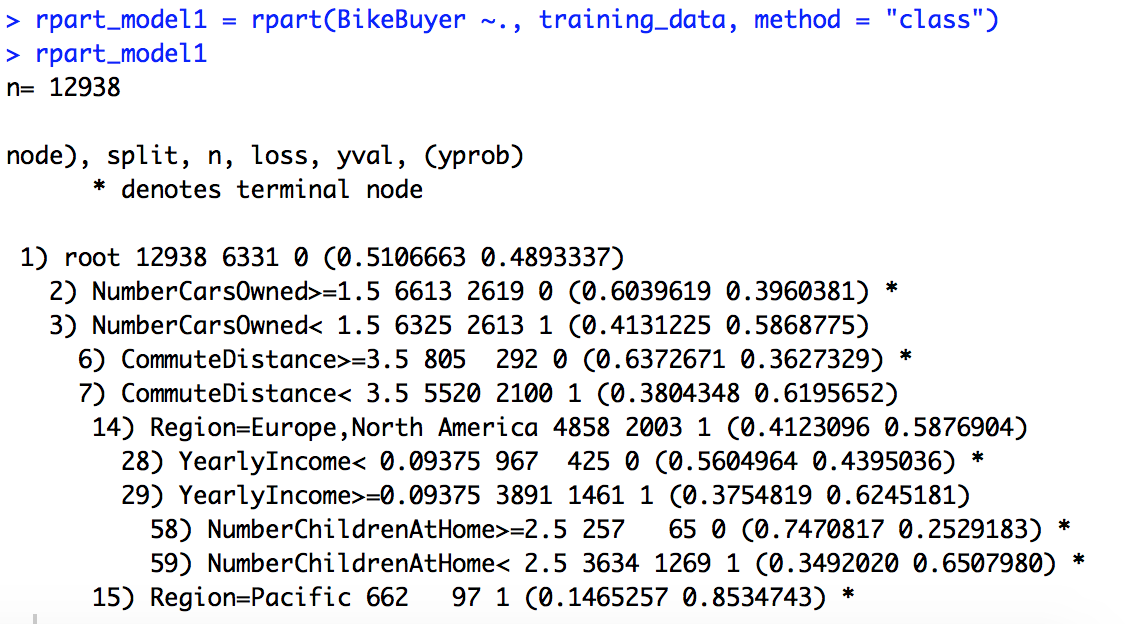
3-1. install packages “rpart” and “rpart.plot”.





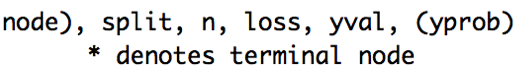
3-2. The target used in the rpart() function is BikeBuyer , predictors are all the attributes, we use . to represent. Dataset is training data set. Our target is categorical features so we want to create a classification tree.





To elaborate the model, we could see the legend. The follow shows

Node number, which feature it was split on, how many obervations, how many misclassifications, the prediction if stop there. Take the following as example, the node number is 2), splite on NumberCarsOwned>=1.5, observation is 6613, misclassification is 2619, prediction is 0 (not bikebuyer) if stop in this node. And this is a leaf node.

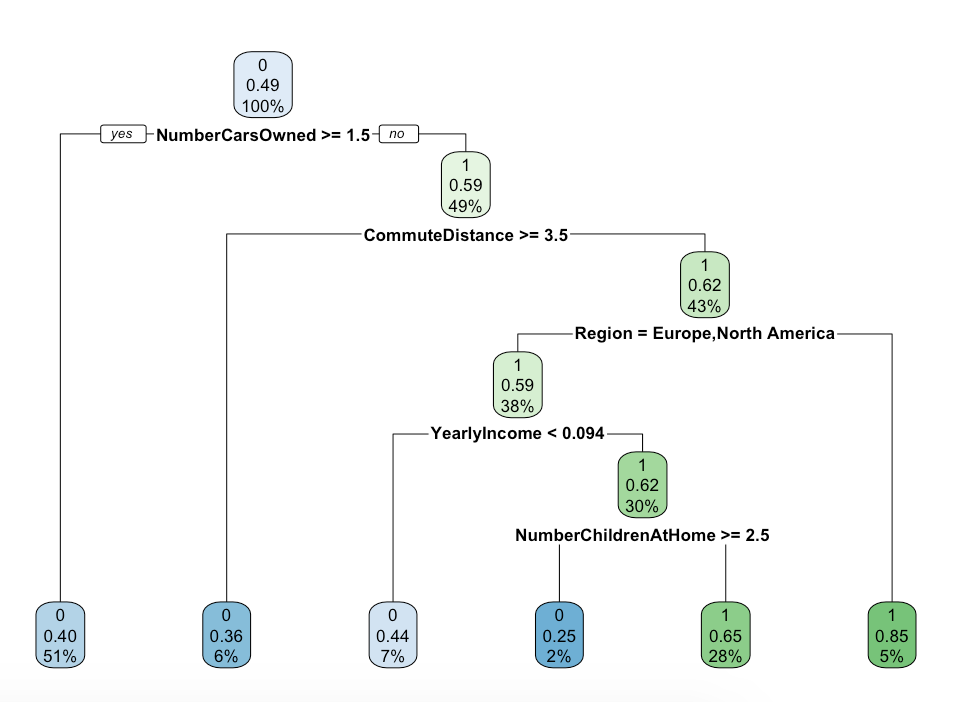


../../pic/Screen%20Shot%202017-06-29%20at%2011.18.20%20AM.png

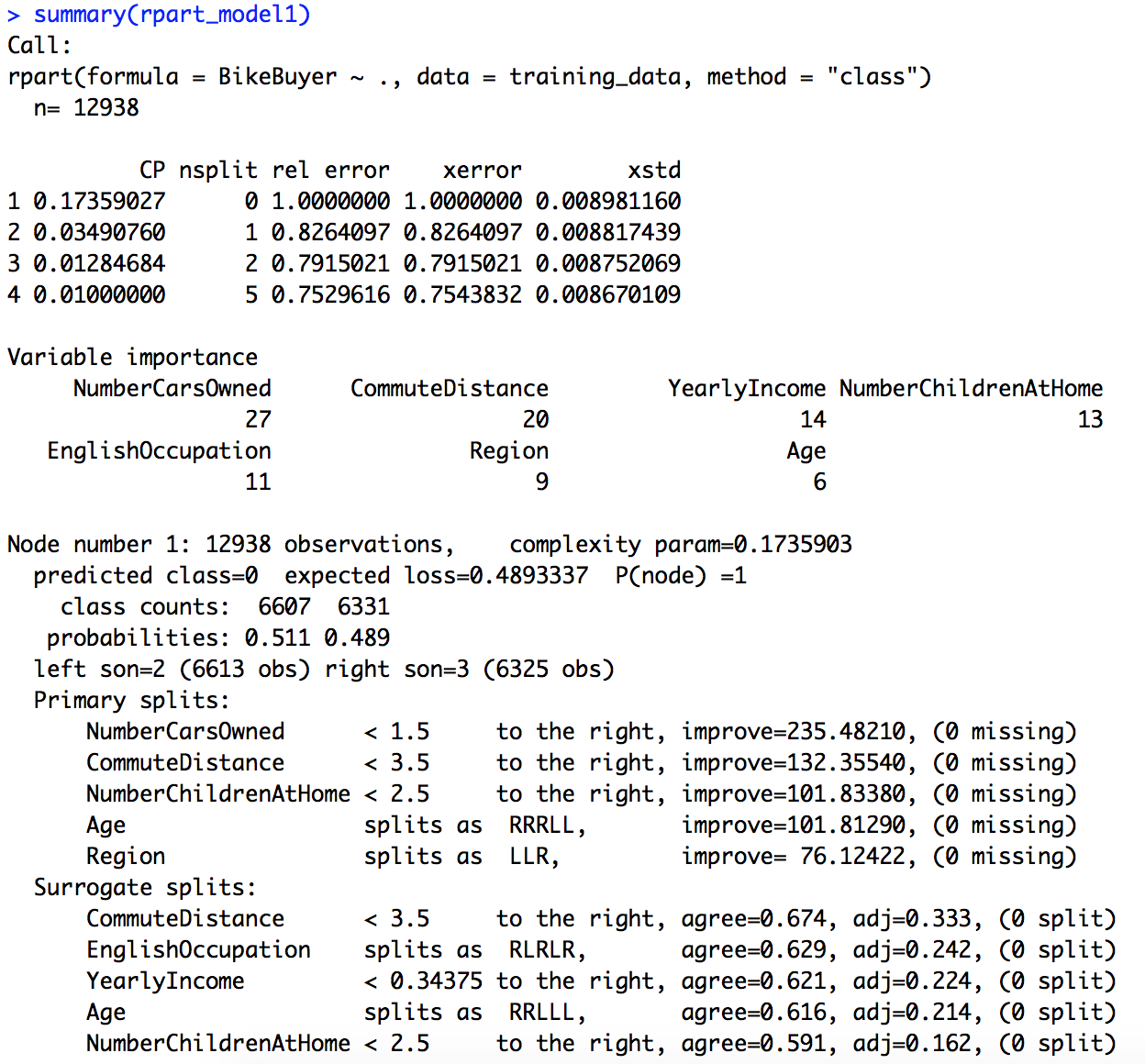
../../pic/Screen%20Shot%202017-06-29%20at%2011.07.28%20AM.png

3-3. To better visualize the tree, I use rplot to plot the model.

../../pic/Screen%20Shot%202017-06-29%20at%2011.07.28%20AM.png



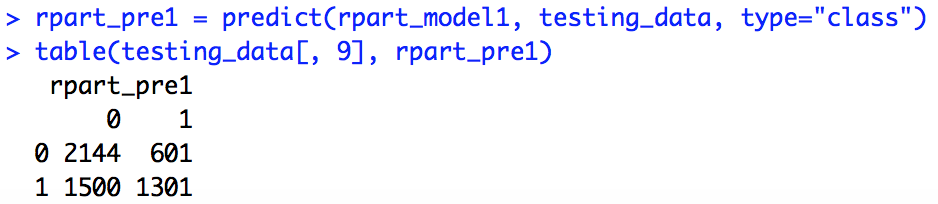
3-4 Summary



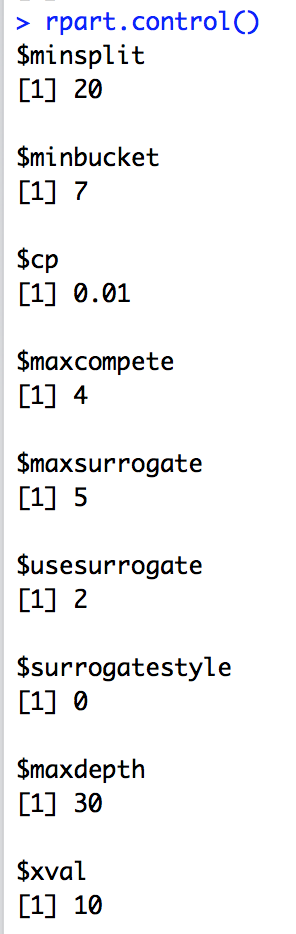
:::::::::::::::omitted

3-5. Test using testing data

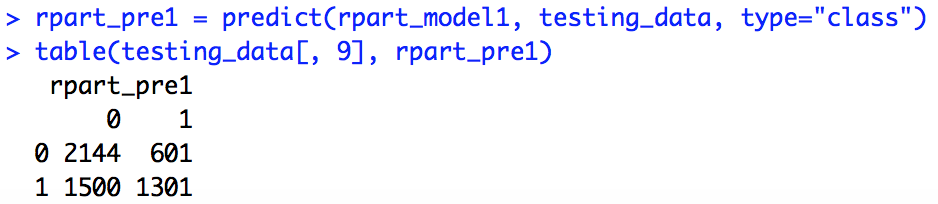
accuracy = (2144+1301)/5546 = 62.12%



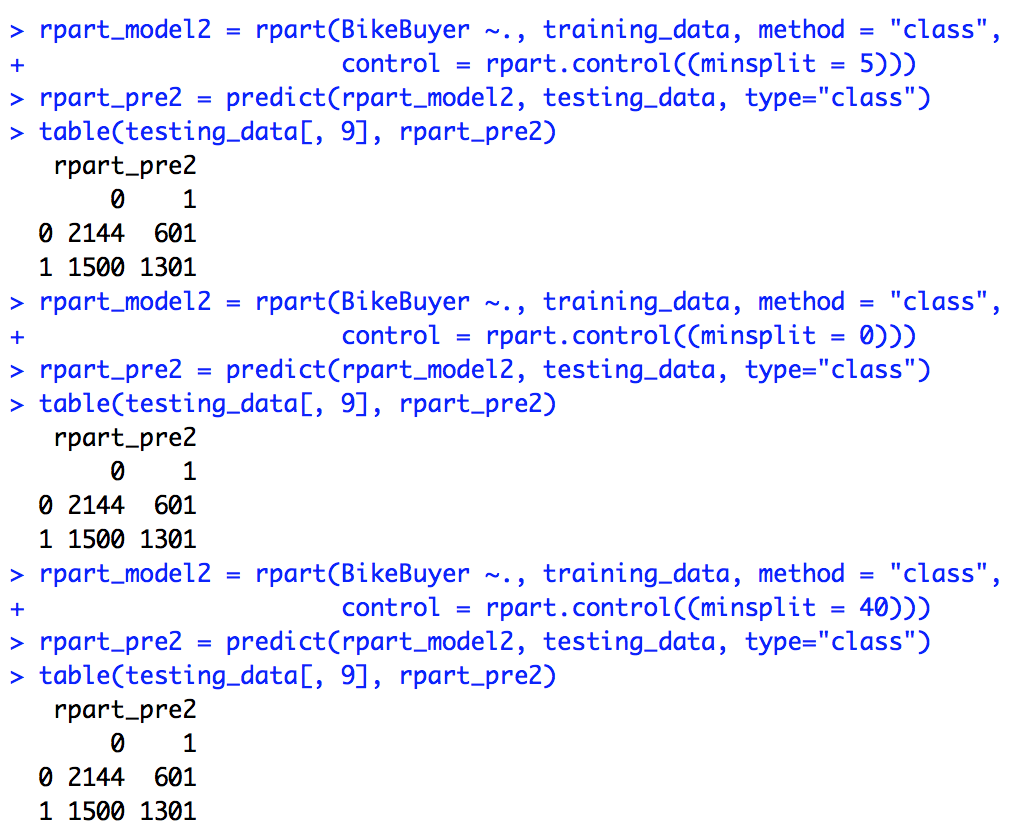
3-6. change parameters



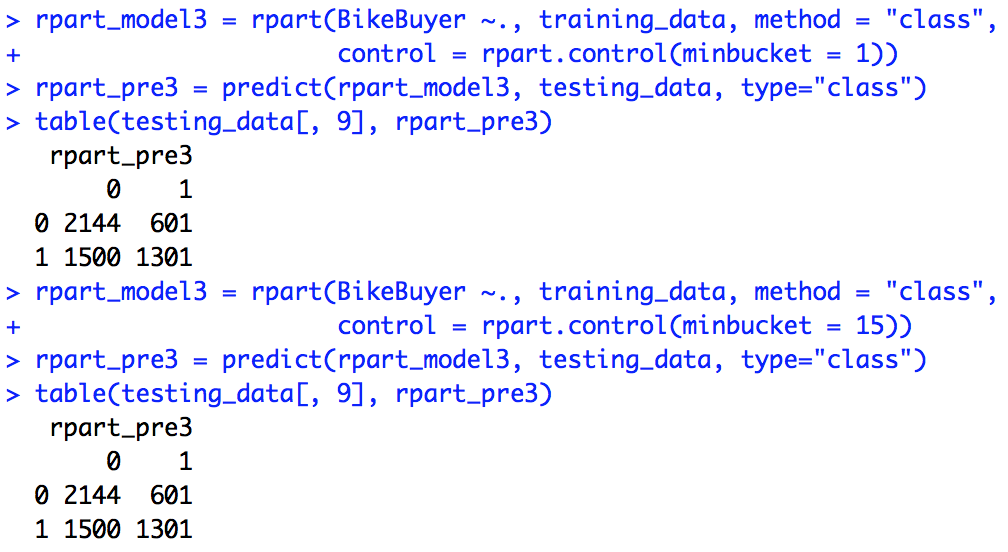
1)default



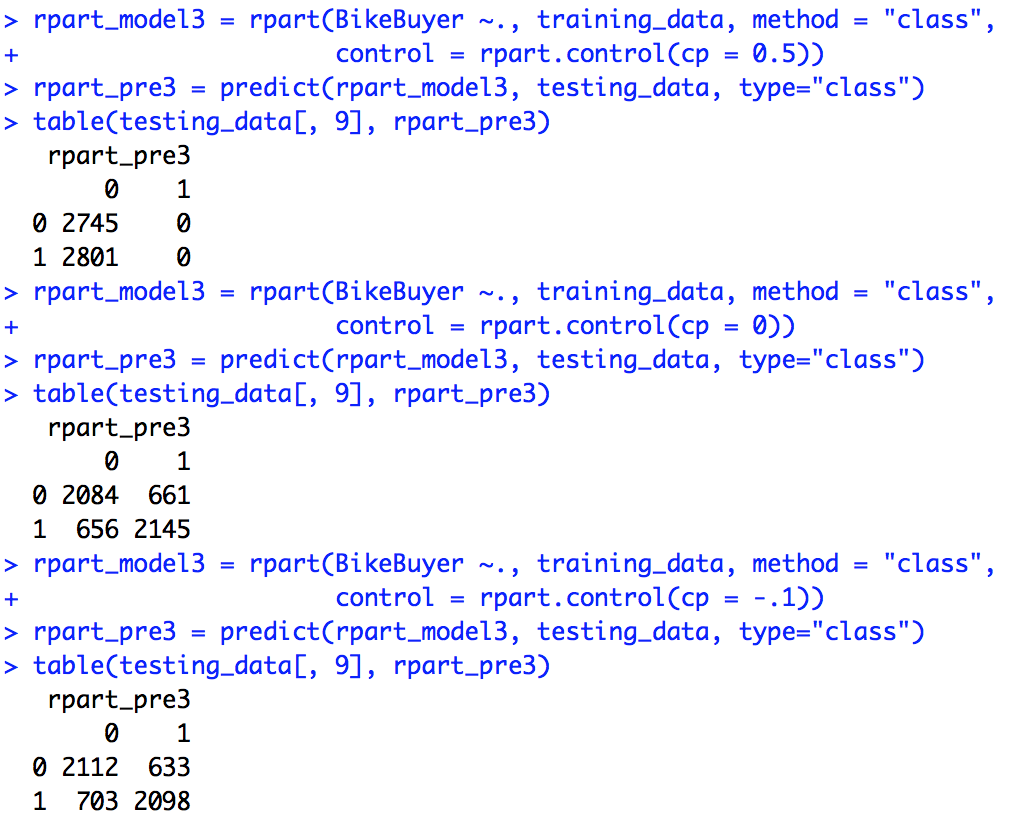
2) minsplit, the changes do nothing to accuracy.



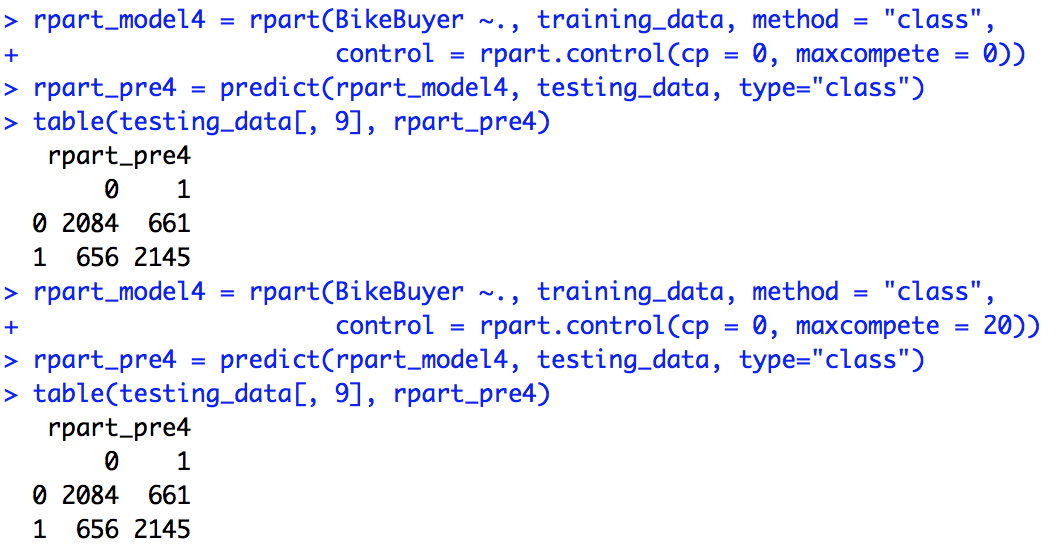
3)minbucket, does not help.



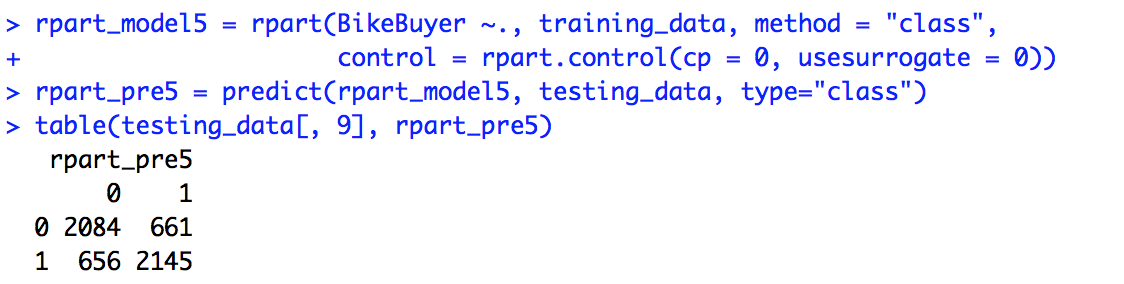
4) cp value, accuracy was improve a lot when cp was changed from 0.5 to 0. When cp = 0, accuracy = (2084+2145)/5546 = 76.25%, great improvement comparing to default one which is 62.12%. So I keep setting cp = 0 in the following test.



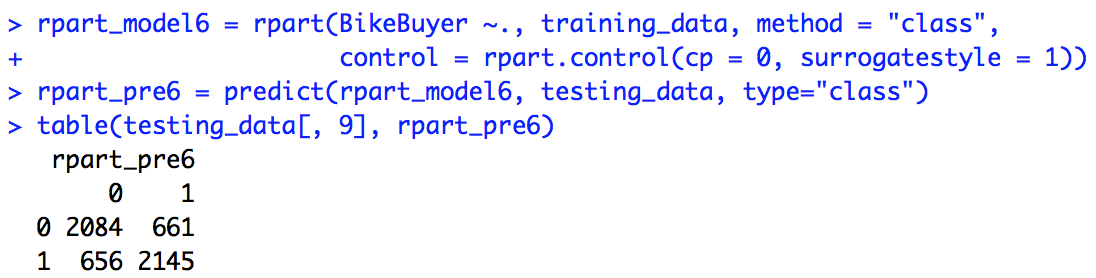
5)maxcompete, no help

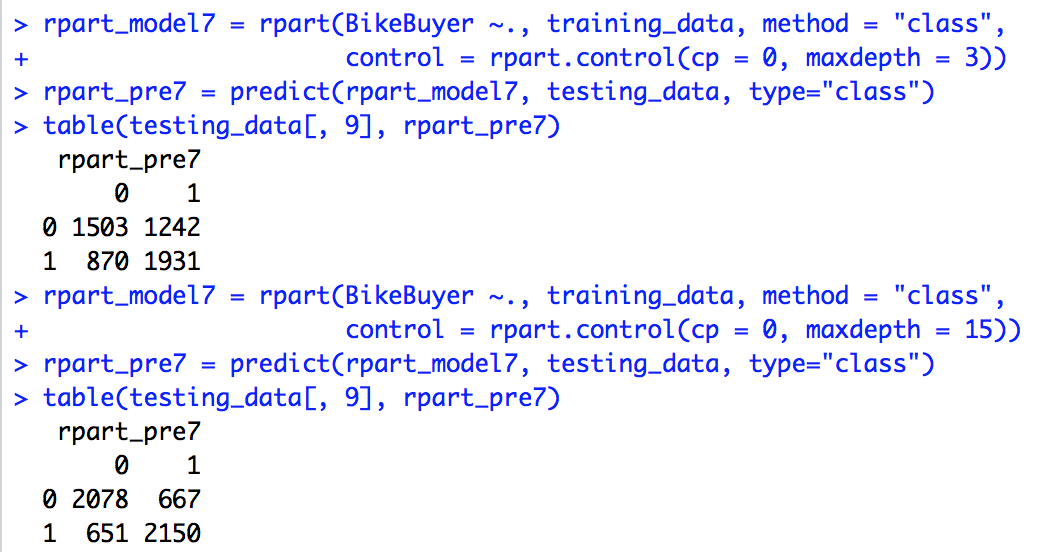


6)usesurrogate, no help

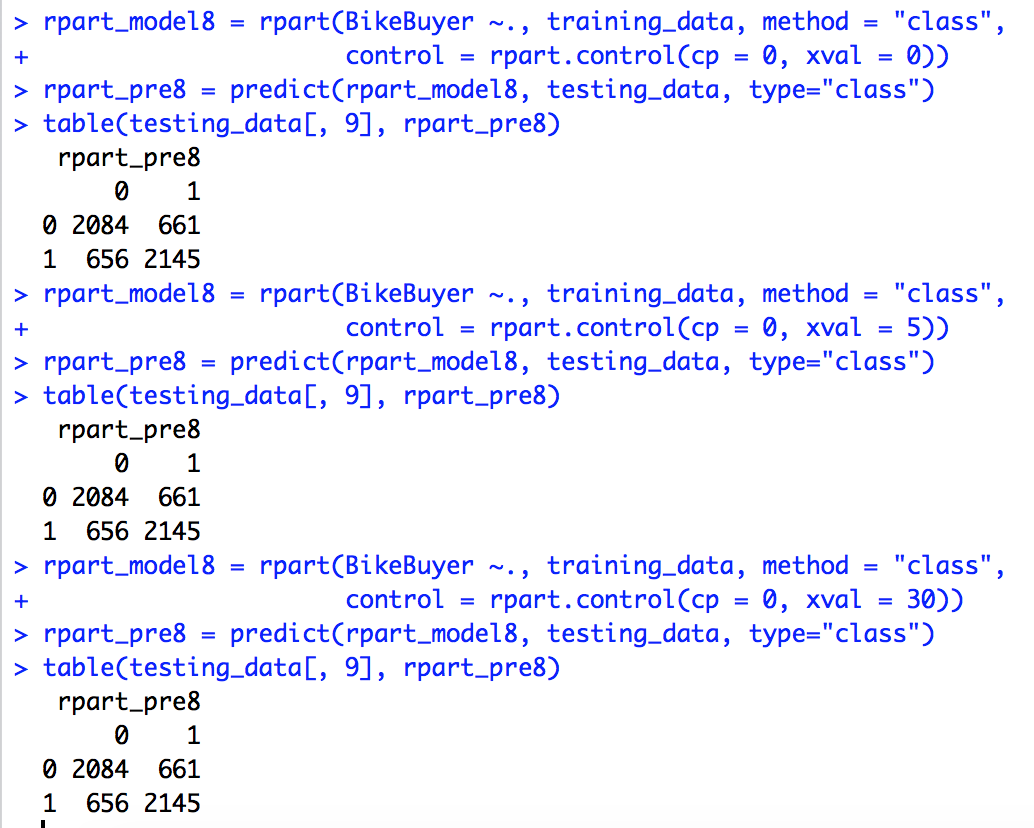


7)surrogatestype, no change



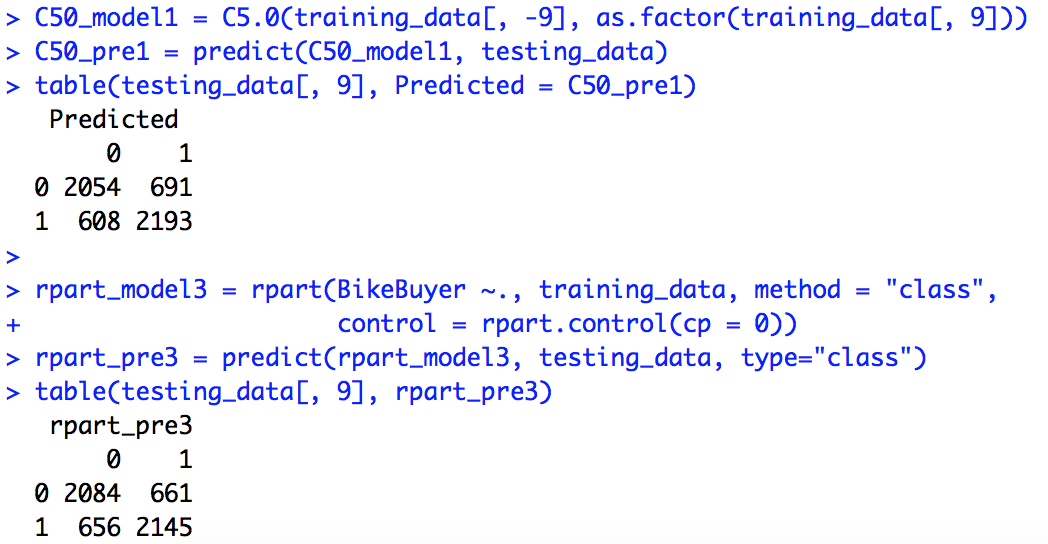
8)maxdepth, no change

9) xval value, no change.



Part 4. Comparison between two classifiers C50 and CART

In this lab, I did C50 modeling va C5.0 and CART modeling via rpart. After changing parameters, the default setting of C5.0 gave me best result on my dataset. For CART modeling, the complexity parameter equaling 0 gives best result.

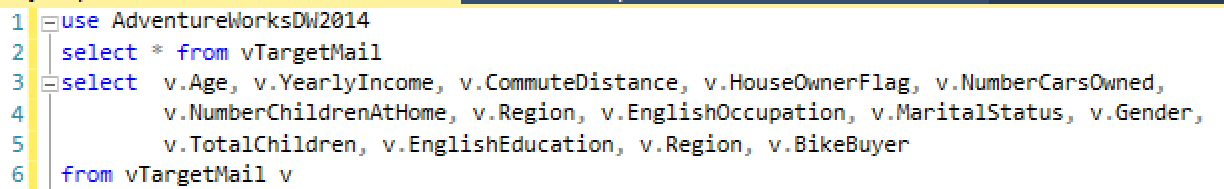


AccuracyC50 = 76.58 %, AccuracyCART(cp = 0.0) = 76.25 %, AccuracyCART(default) = 61.12 %.

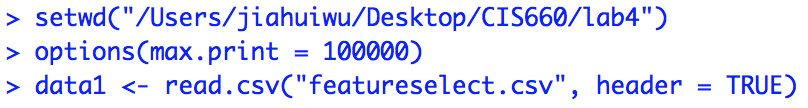
In default parameter setting, C50 has better performance. After setting parameters, the two classifiers have similar performances. Overall, classifier C50 is a nit better than CART for my dataset.

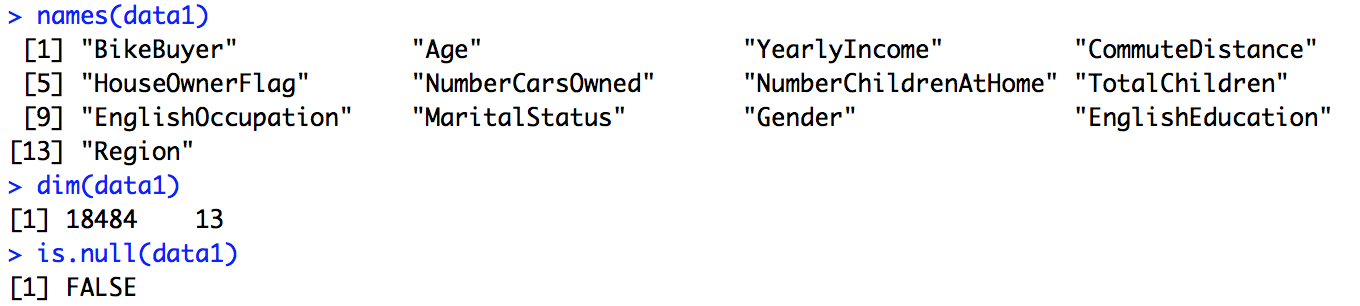
**Part 5. Feature Selection for Extra Credit**

5-1. I selected 13 features from the database, and save the file to .csv.



5-2. In R, I took a look of the new data



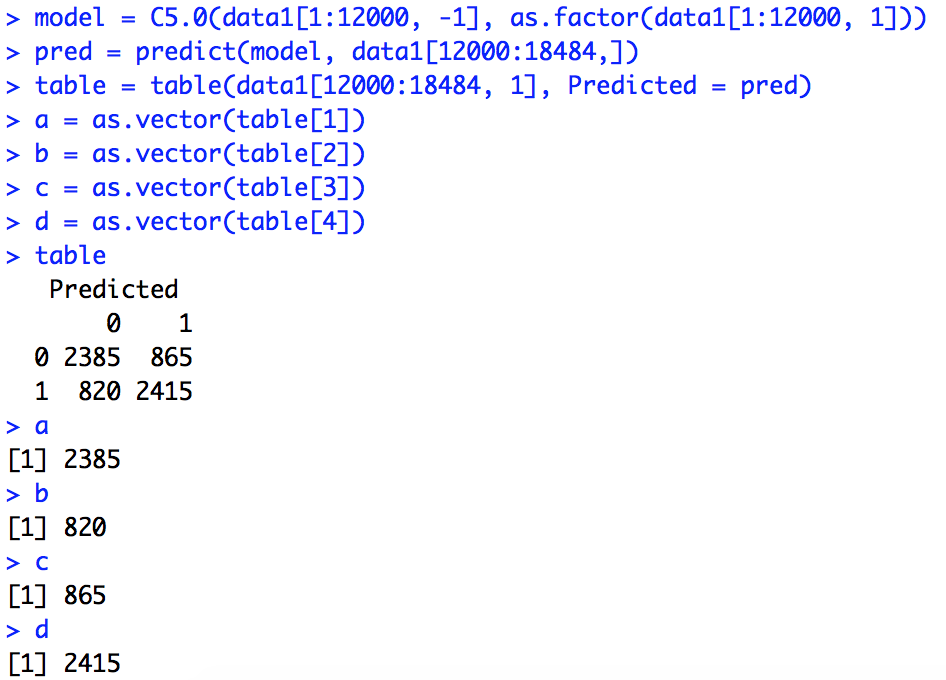


5-3. To find the best feature number, I use a for loop to iterate the column from 13 to 3. The first column is the class feature named “BikeBuyer”, I started with 13 features, and eliminate one at a time to calculate accuracy. I define two variables maxAccuracy and bestFeature, and initialized them to 0 and 13, respectively. Since the C50 needs at least two features to build the model, I can only loop from columns 1-13 to columns 1:3 (the first column is class attribute).

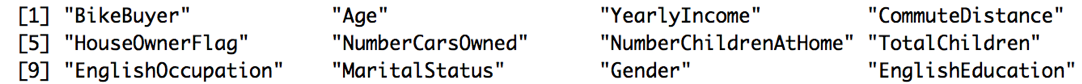
5-3-1. The following print shows the attribute selections in each iteration, we can see that the features starts from “BikeBuyer” to “Region” which are 13 in total, each time eliminate the last feature, and in the end the remaining features are “BikeBuyers”, “Age”, “YearlyIncome”.

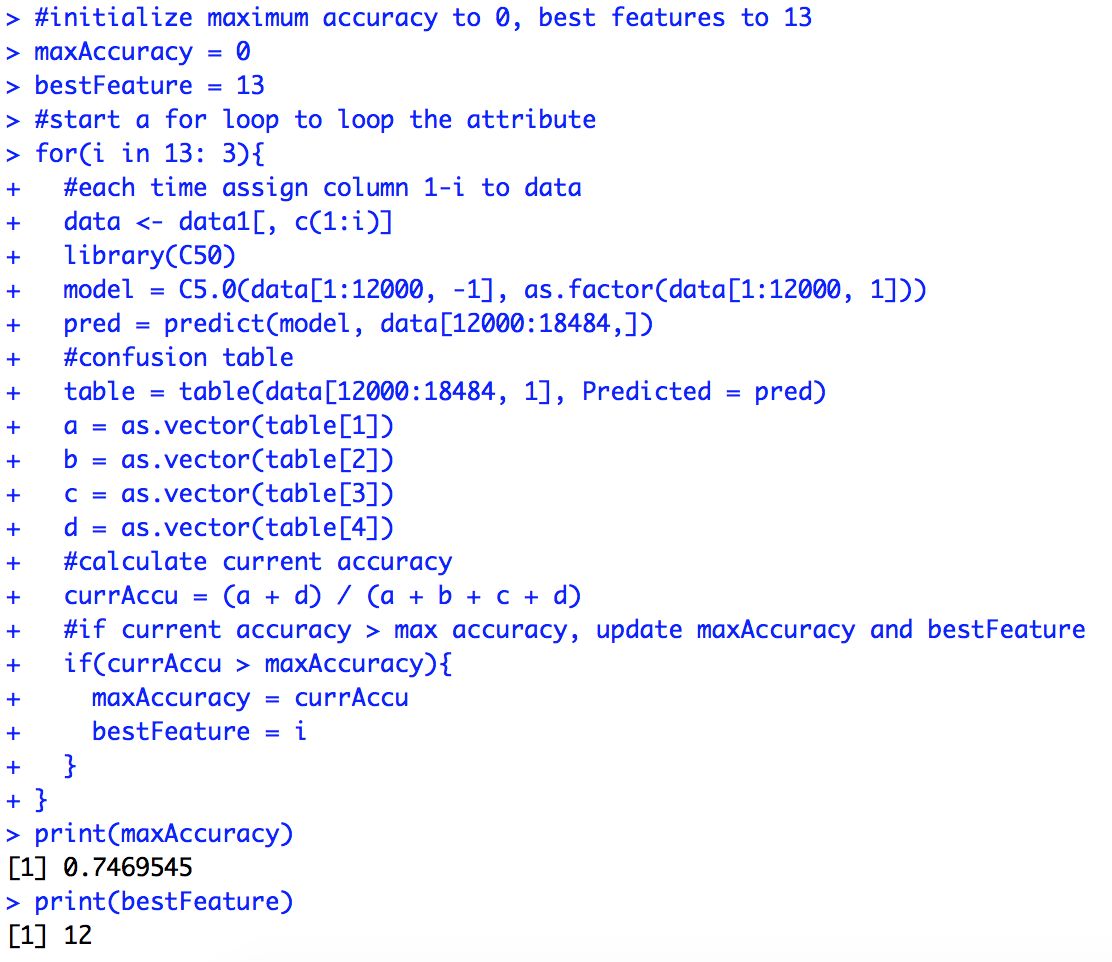


5-3-2. Calculate the accuracy. I use first 12000 as training and 12000-18484 as testing set. I extracted each value from the confusion table to calculate accuracy.



5-4. The code in R. I use C50 in this code. After running the code, the max accuracy is 74.69% and the best feature number is 12, which are the following. The accuracy is different from that in part 3 is because I used different training and testing dataset here.





One drawback of using this algorithm is that it did not take the order of features into consideration. There are much more combinations of features to calculate to achieve the best accuracy. In above code we only consider the simplest condition.