Final Project - IDSC 4444

Henrique Lispector

May 5, 2016

Overview and Motivation

As mankind currently dives into the world of big data, data mining and machine learning algorithms become extremely useful in making predictions and finding hidden trends in data. The fields where these algorithms can be applied is vast, as long as there is data available to be analyzed. In this project, the focus will be on analyzing a dataset about a cardiac disease, and making predictions with the classification decision tree algorithm. The motivation for this choice of topic is because heart disease is the No. 1 cause of death in the United States, killing nearly 787,000 people alone in 2011, according to The Heart Foundation (refer to "References" section). Therefore, the more research is conducted in that area, the more scientists and doctors will be able to understand heart diseases and, hopefully, prevent them in our society.

The dataset comes from Duke University Cardiovascular Disease Databank, with the title "Duke Cardiac Catheterization Coronary Artery Disease Diagnostic Dataset". It consists of 3504 patients and 6 variables, and it was gathered in the year of 2002. It can be found in the following website from the Biostatistics Department at Vanderbilt University: http://biostat.mc.vanderbilt.edu/wiki/pub/Main/DataSets/acath.xls.zip. A more detailed description of the dataset is available here: http://biostat.mc.vanderbilt.edu/wiki/pub/Main/DataSets/acath.html; and here: http://biostat.mc.vanderbilt.edu/wiki/pub/Main/DataSets/Cacath.html.

Dataset Description

The name of the dataset is "acath", and it has the following variables:

sex: 0 = male, 1 = female;
age: age of patients in years;
cad.dur: Duration of Symptoms of Coronary Artery Disease in a non-specified integer time unit;
choleste: Cholesterol of patient in mg %;
sigdz: Significant Coronary Disease by Cardiac Catheterization. 1 = TRUE, 0 = FALSE;
tvdlm: Three Vessel or Left Main Disease by Cardiac Catheterization. 1 = TRUE, 0 = FALSE;

The response variable of this study is "sigdz", as, ultimately, we are trying to predict if a patient has significant coronary disease by cardiac catheterization or not. The classification decision tree algorithm is appropriate for this type of dataset since our dependent variable is binary.

Loading the data

```r
acath <- read.csv("~/Desktop/acath.csv")
```

Data Pre-Processing

First, we will look at the first six rows of the data to see how the dataset looks like:

```
head(acath)
```

```
##   sex age cad.dur choleste sigdz tvdlm
## 1   0  73     132      268     1     1
## 2   0  68      85      120     1     1
## 3   0  54      45       NA     1     0
## 4   1  58      86      245     0     0
## 5   1  56       7      269     0     0
## 6   0  64       0       NA     1     0
```

We can see right of the bat that there are NA's in the dataset, which can cause problems to our analysis. For that reason, we will delete the rows with NA's from the dataset with the following command, and check the first six rows again to confirm that the NA's were removed:

```
acath <- na.omit(acath)
head(acath)
```

```
##    sex age cad.dur choleste sigdz tvdlm
## 1    0  73     132      268     1     1
## 2    0  68      85      120     1     1
## 4    1  58      86      245     0     0
## 5    1  56       7      269     0     0
## 8    0  41      15      247     1     0
## 12   0  35      44      257     0     0
```

With the NA's removed, we move on to verifying if our variables are coded with the correct data types:

```
lapply(acath, class)
```

```
## $sex
## [1] "integer"
##
## $age
## [1] "integer"
##
## $cad.dur
## [1] "integer"
##
## $choleste
## [1] "integer"
##
## $sigdz
## [1] "integer"
##
## $tvdlm
## [1] "integer"
```

As we can see above, all variables are coded as integers. However, some variables are categorical, and should be classified as such, as done in the code below. In addition, we check again the data types of the variables to see if the changes were made correctly:

```
acath <- with(acath, data.frame(data.frame(sex=factor(sex), age=age, cad.dur=cad.dur,
                                           choleste=choleste,
sigdz = factor(sigdz), tvdlm = factor(tvdlm))))

lapply(acath, class)
```

```
## $sex
## [1] "factor"
##
## $age
## [1] "integer"
##
## $cad.dur
## [1] "integer"
##
## $choleste
## [1] "integer"
##
## $sigdz
## [1] "factor"
##
## $tvdlm
## [1] "factor"
```

With all variables coded with the correct data type, we can advance to our next phase of the analysis, the exploration of the data.

Exploratory Data Analysis

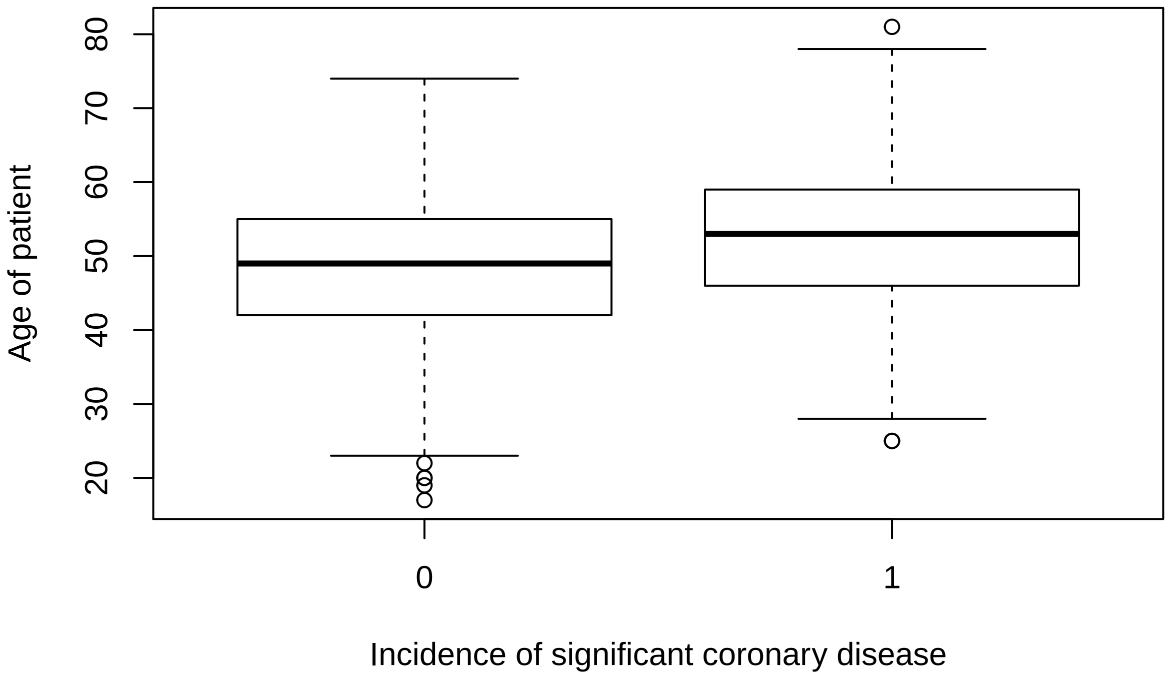
In order to get the big picture of the dataset, we can use the summary function below:

```
summary(acath)
```

```
##  sex           age           cad.dur          choleste     sigdz
##  0:1569   Min.   :17.00   Min.   :  0.00   Min.   : 29.0   0: 768
##  1: 689   1st Qu.:45.00   1st Qu.:  6.00   1st Qu.:196.0   1:1490
##           Median :51.00   Median : 19.00   Median :224.5
##           Mean   :50.82   Mean   : 41.91   Mean   :229.9
##           3rd Qu.:57.00   3rd Qu.: 58.00   3rd Qu.:259.0
##           Max.   :81.00   Max.   :416.00   Max.   :576.0
##  tvdlm
##  0:1535
##  1: 723
##
##
##
##
```

For a more visual appealing exploration, we can graph box plots between each numerical variable and the response variable:

```
boxplot(age~sigdz, data=acath, ylab="Age of patient",
        xlab="Incidence of significant coronary disease")
```



In the box plot above, we can visualize that the age median of patients with coronary disease is slightly higher than the patients without it. To query the age means of each of the two response groups, we can use the following function:

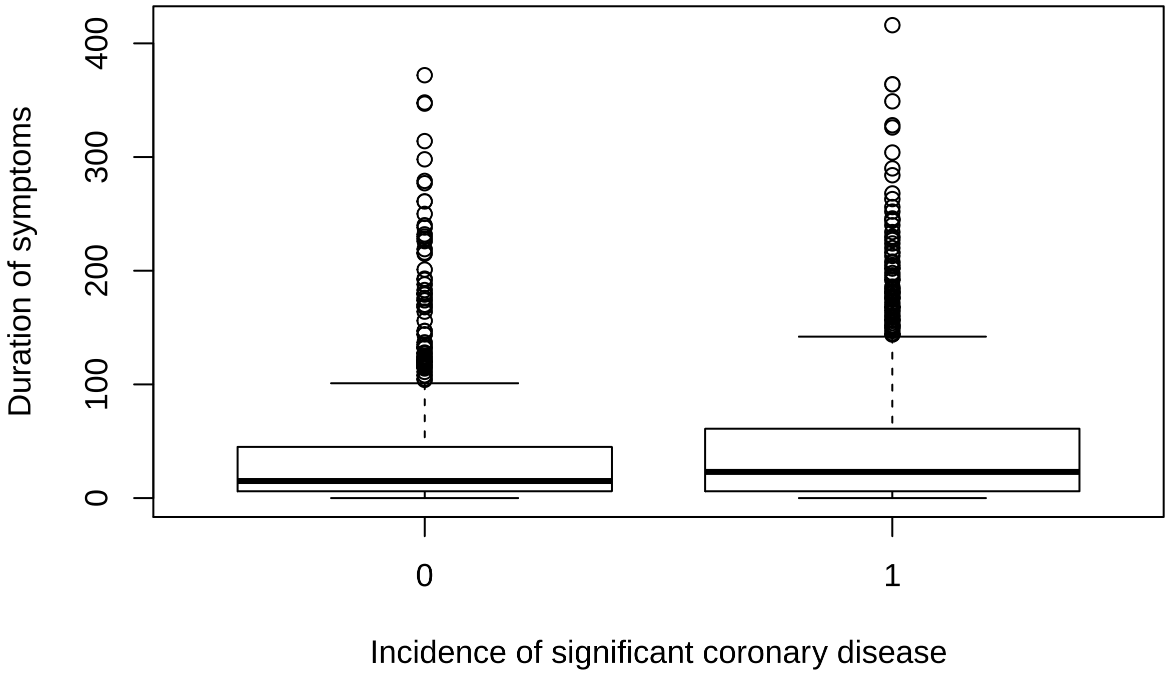
```
with(acath, tapply(age, sigdz, mean))
```

```
##        0        1
## 47.97786 52.29128
```

The output above tells us that the average age of patients with coronary disease is around 52 years old, while ~48 years old are the patients without the disease, confirming what we roughly see in the boxplot.

Now we will graph a box plot and use the tapply function with the “cad.dur”" variable:

```
boxplot(cad.dur~sigdz, data=acath, ylab="Duration of symptoms",
        xlab="Incidence of significant coronary disease")
```



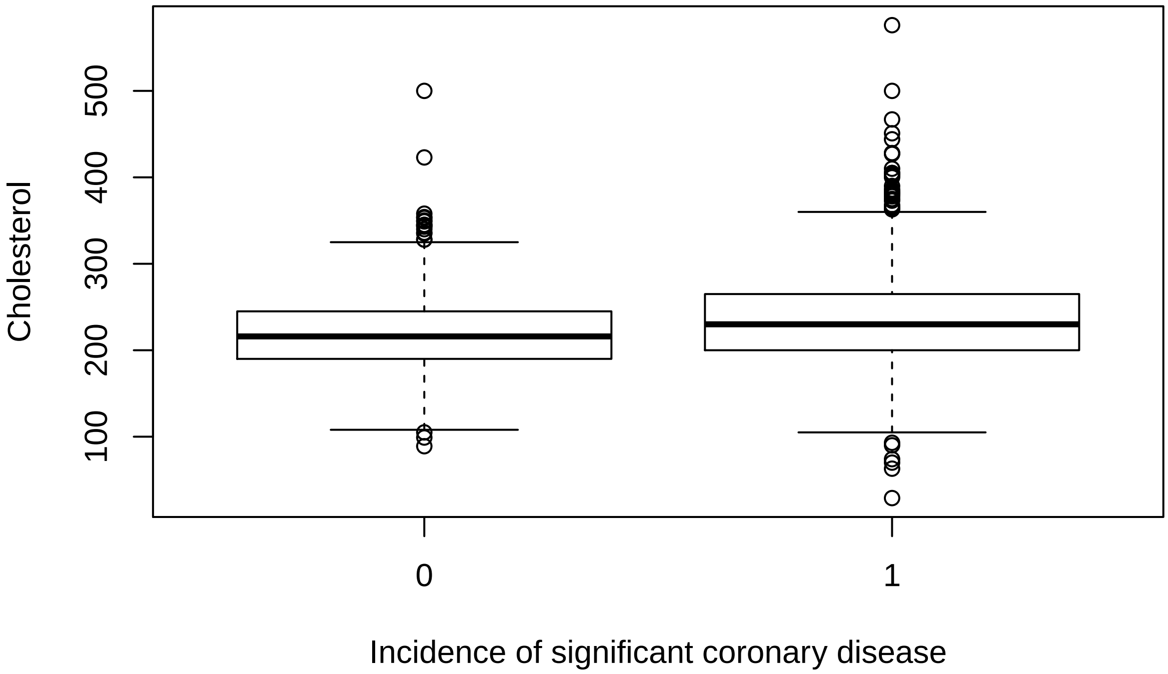
```r
with(acath, tapply(cad.dur, sigdz, mean))
```

```
##        0        1
## 37.92839 43.96107
```

The information above shows us that patients with coronary disease have longer symptom durations than patients without coronary disease.

Similarly, we can do the same with the cholesterol independent variable:

```r
boxplot(choleste~sigdz, data=acath, ylab="Cholesterol",
        xlab="Incidence of significant coronary disease")
```



```
with(acath, tapply(choleste, sigdz, mean))
```

```
##        0        1
## 220.1146 234.9866
```

As the information above shows us, patients with the incidence of significant coronary disease have higher cholesterol than patients without the disease.

So far we analyzed all three numeric independet variables. Now we will explore the categorical variables "sex" and "tvdlm" with the tables below:

```
with(acath, table(sigdz,sex))
```

```
##      sex
## sigdz    0    1
##     0  350  418
##     1 1219  271
```

The table above shows us that roughly 77% of the males analyzed in this study have coronary disease, while only about 39% of females had coronary disease. This may suggest that "sex" might be one of the first splits in the decision tree algorithm.

Now "tvdlm":

```
with(acath, table(sigdz,tvdlm))
```

```
##      tvdlm
## sigdz   0   1
##     0 768   0
##     1 767 723
```

The table above tells us something very interesting. All the patients that had the three vessel or left main disease by cardiac catheterization also had incidence of significant coronary disease. Since we already know that this variable will be the first split and that it will significantly bias our predictions, this means that this variable will not add much information to our classification tree, so we will eliminate it from our dataset.

```
acath <- acath[,-6]
head(acath)
```

```
##   sex age cad.dur choleste sigdz
## 1   0  73     132      268     1
## 2   0  68      85      120     1
## 3   1  58      86      245     0
## 4   1  56       7      269     0
## 5   0  41      15      247     1
## 6   0  35      44      257     0
```

Partitioning the data

Before we apply the classification tree algorithm and start making predictions, it is wise to partition our data into two subsets: a training data subset, and a test data subset. We will apply the algorithm to the training data, test it on the test data, and check the performance of predictions. The "set.seed" function will assign a random constant value to the runs, so the analysis is simplified. In addition, this training data will contain a random sample of 70% from the "acath" dataset, while the other 30% will be assigned to the test data subset.

```
set.seed(5)
index_training <- sample(1:nrow(acath), round(0.7*nrow(acath)))
training_data <- acath[index_training,]
test_data <- acath[-index_training,]
```

Fitting the Model

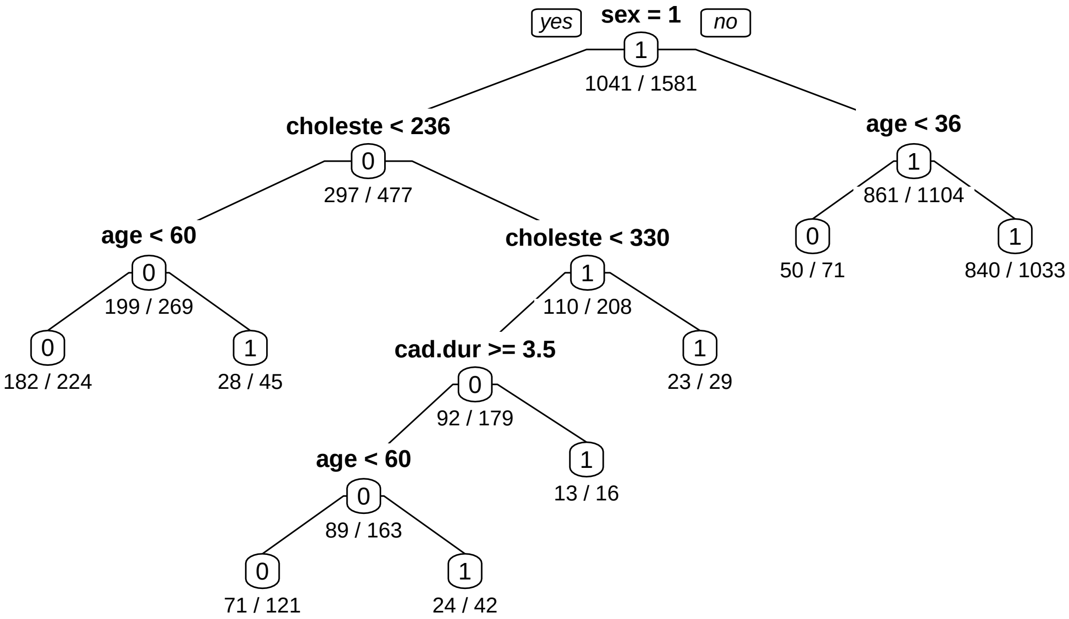
First, we will load the packages required for classification trees:

```
library(rpart)
library(rpart.plot)
```

Now, we will create our first tree:

```
tree_1 <- rpart(sigdz ~ sex + age + cad.dur + choleste, data = training_data, method = "class",
                control = rpart.control(minsplit=0))
rpart.plot(tree_1,type = 1,extra=2,under=TRUE, main = "Classification Tree for Coronary Disease")
```

Classification Tree for Coronary Disease



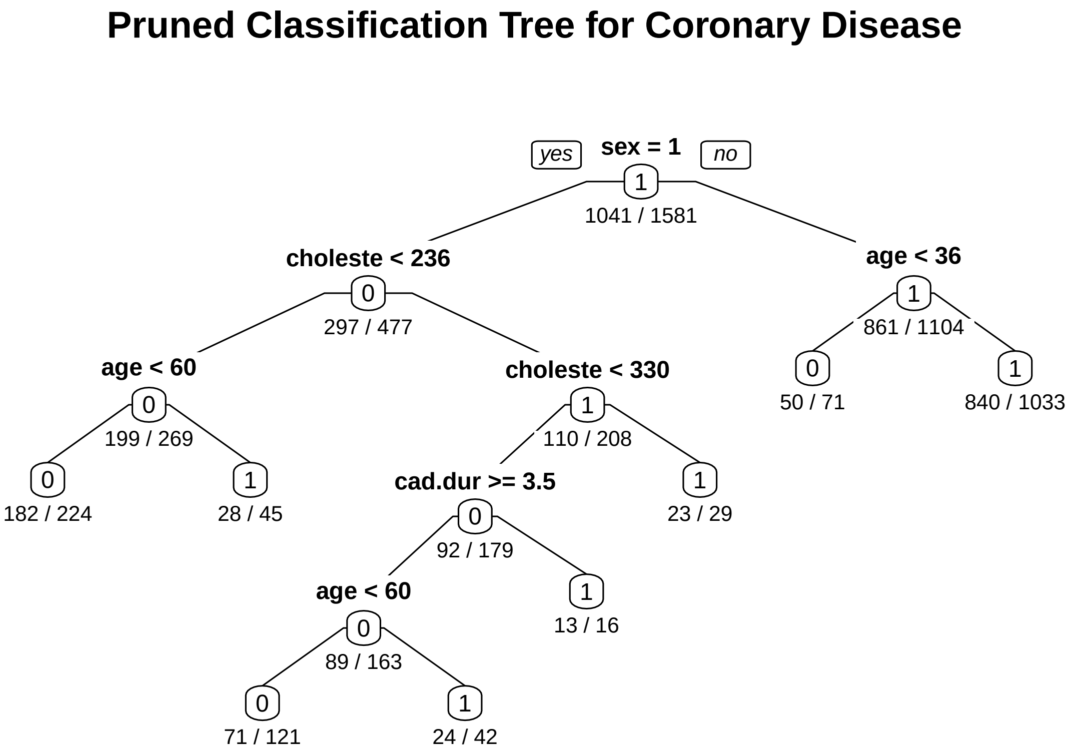
As no pre-prunning was made, let us try to post prune this tree. The following function provides the best prunnings based on Complexity Parameter (cp) value of the tree and cross-validated error:

```
printcp(tree_1)
```

```
##
## Classification tree:
## rpart(formula = sigdz ~ sex + age + cad.dur + choleste, data = training_data,
##     method = "class", control = rpart.control(minsplit = 0))
##
## Variables actually used in tree construction:
## [1] age      cad.dur  choleste sex
##
## Root node error: 540/1581 = 0.34156
##
## n= 1581
##
##         CP nsplit rel error  xerror     xstd
## 1 0.216667      0   1.00000 1.00000 0.034919
## 2 0.053704      1   0.78333 0.78333 0.032596
## 3 0.022222      2   0.72963 0.75185 0.032168
## 4 0.020370      3   0.70741 0.75741 0.032245
## 5 0.013889      4   0.68704 0.72222 0.031742
## 6 0.011111      6   0.65926 0.71481 0.031631
## 7 0.010000      7   0.64815 0.69630 0.031349
```

From the list of “cp” values above as well as from the graph, we select the “cp” that has the least cross-validated error associated with it, and use it to prune the tree. The value of “cp”" should be the smallest, so that the cross-validated error rate is minimum.

```
pruned_tree_1 <- prune(tree_1, cp = 0.01)
rpart.plot(pruned_tree_1,type = 1,extra=2,under=TRUE,
           main = "Pruned Classification Tree for Coronary Disease")
```



The prunned tree is actually identical to the non-prunned, which means if I try a pre-prunning process with different minsplit values or if I try different “cp”" values, the accuracy of my predictions should not improve significantly. So we move on with the non-prunned tree.

Prediction and Validation

Now we are finally ready to make our predictions.

```
test_data$pred <- predict(tree_1,test_data, type="class")
table(test_data$sigdz,test_data$pred,dnn = c("Actual","Predicted"))
```

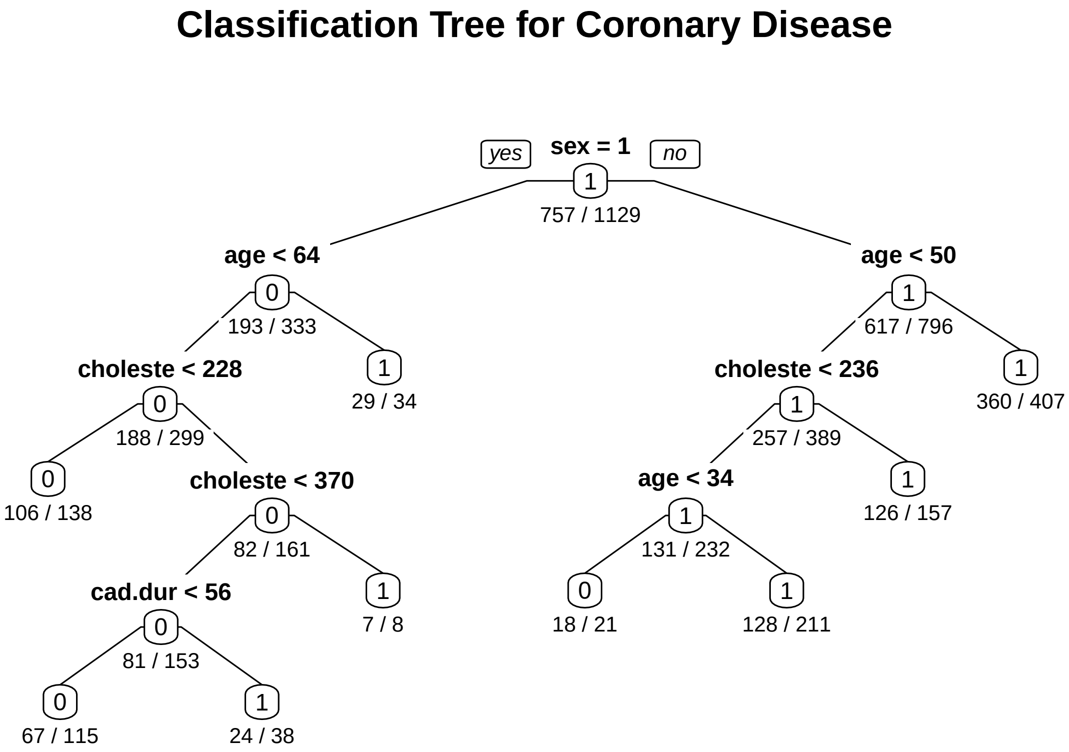
```
##       Predicted
## Actual   0   1
##      0 109 119
##      1  81 368
```

Accuracy: *0.7046*
Recall for positive class (sensitivity): *0.8196*
Recall for negative class (specificity): *0.4781*
Precision of prediction for positive class: *0.7556*
Precision of prediction for negative class: *0.5737*

We care here more about the false negatives than the false positives. The reason is because if a person who does not have disease is classified as a person with the disease, there is still a chance the diagnosis will still be reversed in further medical tests. However, if a person that has the disease is wrongly diagnosed as if they did not have the disease, it is a very grave mistake, as there is a very high chance the person may never find out they actually have the disease.

The accuracy is not great, but it is not bad. The sensitivity factor is good. Let us try to do the analysis with a different sample size for the training and test subsets of the data to see if we can improve those metrics. This time, we will assign 50% of the dataset to the training subset and the other 50% to the test subset.

```r
set.seed(27)
index_training_2 <- sample(1:nrow(acath), round(0.5*nrow(acath)))
training_data_2 <- acath[index_training_2,]
test_data_2 <- acath[-index_training_2,]

tree_2 <- rpart(sigdz ~ sex + age + cad.dur + choleste, data = training_data_2,
                method = "class", control = rpart.control(minsplit=0))
rpart.plot(tree_2,type = 1,extra=2,under=TRUE, main = "Classification Tree for Coronary Disease")
```

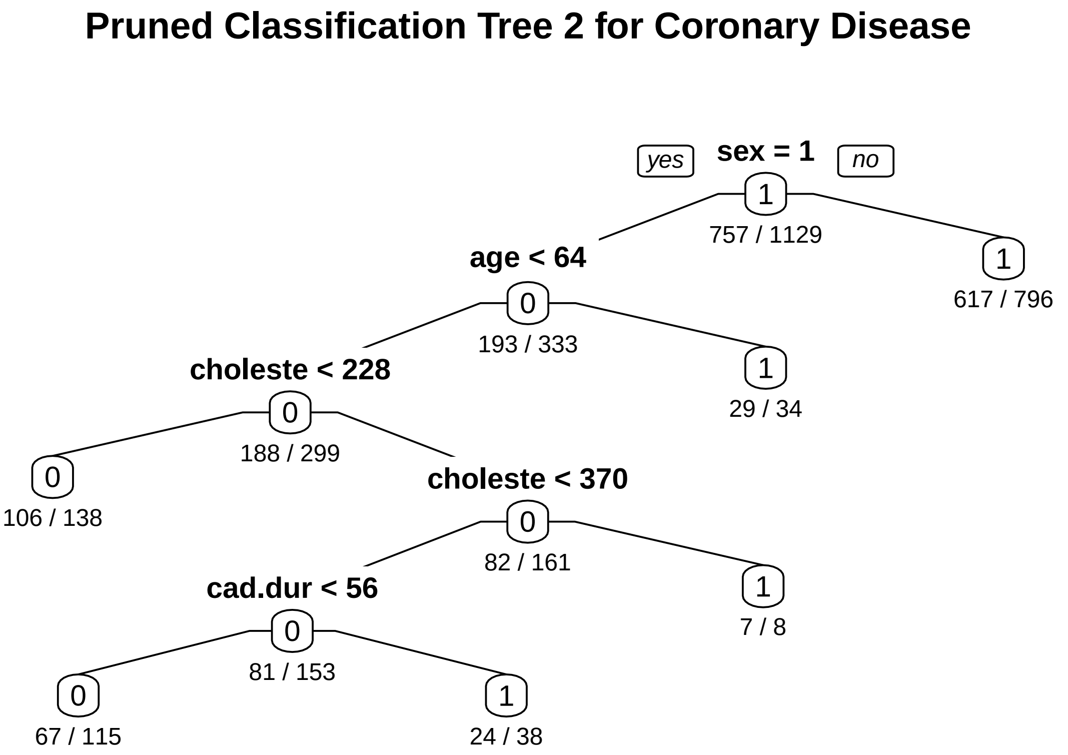


```r
printcp(tree_2)
```

```
##
## Classification tree:
## rpart(formula = sigdz ~ sex + age + cad.dur + choleste, data = training_data_2,
##     method = "class", control = rpart.control(minsplit = 0))
##
## Variables actually used in tree construction:
## [1] age      cad.dur  choleste sex
##
## Root node error: 372/1129 = 0.3295
##
```

```
## n= 1129
##
##          CP nsplit rel error  xerror     xstd
## 1 0.142473      0   1.00000 1.00000 0.042455
## 2 0.064516      1   0.85753 0.85753 0.040668
## 3 0.014337      2   0.79301 0.80376 0.039855
## 4 0.013441      5   0.75000 0.84140 0.040432
## 5 0.010000      8   0.70968 0.83333 0.040312
```

```
pruned_tree_2 <- prune(tree_2, cp = 0.0143)
rpart.plot(pruned_tree_2,type = 1,extra=2,under=TRUE,
           main = "Pruned Classification Tree 2 for Coronary Disease")
```



In this case, we see a different tree 2 after it was optimally post pruned. So we go with the pruned tree 2 and move on to make our predictions for this second model:

```
test_data_2$pred <- predict(pruned_tree_2,test_data_2, type="class")
table(test_data_2$sigdz,test_data_2$pred,dnn = c("Actual","Predicted"))
```

```
##       Predicted
## Actual   0   1
##      0 189 207
##      1  90 643
```

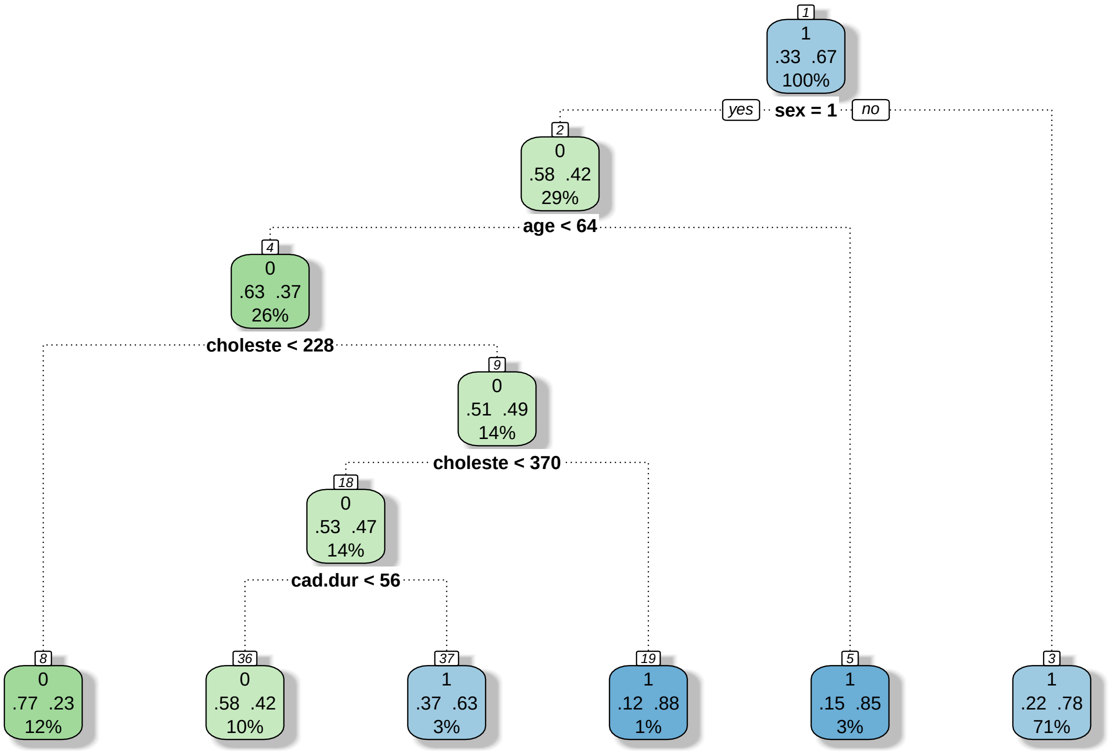
Accuracy: *0.7369*
Recall for positive class (sensitivity): *0.8772*
Recall for negative class (specificity): *0.4772*
Precision of prediction for positive class: *0.7565*
Precision of prediction for negative class: *0.6774*

We observe an increase in accuracy by around 3 percentage points and an increase in sensitivity by about 6 percentage points in the second model compared to the first model, so that is our chosen model for our conclusion. To add more information to our chosen tree, we can also graph the tree with the following code in the following fashion:

```
library(rattle)
```

```
## Warning: Failed to load RGtk2 dynamic library, attempting to install it.

## Please install GTK+ from http://r.research.att.com/libs/GTK_2.24.17-X11.pkg

## If the package still does not load, please ensure that GTK+ is installed and that it is on your PATH

## IN ANY CASE, RESTART R BEFORE TRYING TO LOAD THE PACKAGE AGAIN

## Rattle: A free graphical interface for data mining with R.
## Version 4.1.0 Copyright (c) 2006-2015 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
fancyRpartPlot(pruned_tree_2)
```



Rattle 2016−May−06 17:13:52 HenriqueLispector

Conclusions and Suggested Courses of Action

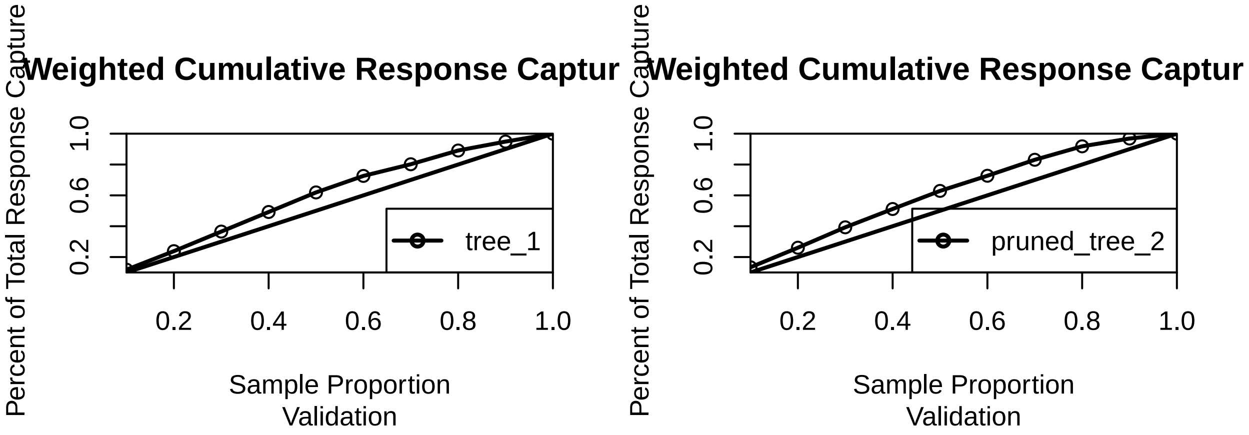
After developing two classification trees with two different training and test subsets from the same dataset, we conclude that the model based on the second tree (pruned) and second training and test subsets makes more accurate predictions than the model based on the first tree and first training and test subsets. The lift charts below show that the slope of the second tree is also a bit steeper than the slope of the first tree, indicating that the second tree finds correct results more quickly than the first tree, and certainly more quickly than if we were selecting patients at random to see if they have coronary disease or not.

```
library(BCA)
par(mfrow=c(2,2))
lift.chart(c("tree_1"), data=test_data,targLevel="1",trueResp=1490/2258, type="cumulative",
           sub="Validation")
```

```
## [1] 0.6632201
```

```
lift.chart(c("pruned_tree_2"), data=test_data_2, targLevel="1",trueResp=1490/2258, type="cumulative",
           sub="Validation")
```

```
## [1] 0.6492471
```



The interpretation of the second tree, is as follows:

- Out of the 2,258 patients in the dataset, only 1,129 were considered in tree two as the training subset.
- Out of 796 patients that were male, about 78% had incidence of significant coronary disease.
- Out of 34 patients who were females and 64 years old or older, 85% had incidence of significant coronary disease.
- Out of the 8 patients who were females, younger than 64 years old, and had cholesterol at 370mg% or higher, 87.5% had incidence of significant coronary disease.
- Out of the 38 patients who were females, younger than 64 years old, had cholesterol between 228mg% and 370mg%, and symptom duration of 56 units of time or higher, 63% had incidence of significant coronary disease.
- Out of the 115 patients who were females, younger than 64 years old, had cholesterol between 228mg% and 370mg%, and symptom duration of lower than 56 units of time, 42% had incidence of significant coronary disease.
- Out of the 138 patients that were females, younger than 64 years old, and had cholesterol of less than 228mg%, only 23% had incidence of significant coronary disease.

Overall, the model can predict with about 74% of accuracy if a patient has incidence of significant coronary disease or not. Based on the characteristics above, the possible courses of action could be to contact a doctor or a medical student, and show them the results of this analysis, so they can improve the accuracy of their diagnosis. In addition, they can use the conclusions of this study to develop targeted medications for types of patients with high probability of having coronary disease.

References

Heart disease statistics source:

- http://www.theheartfoundation.org/heart-disease-facts/heart-disease-statistics/

Optimal "cp" value and "rattle" package information source:

- http://www.edureka.co/blog/implementation-of-decision-tree/