Prediction of Cardiovascular Disease by using different Machine Learning Techniques

MATH 6359 - Statistical Application and Multivariate Statistics

Group 8 - Elizabeth Garcia, Hussein Gaziza, Sara Nafaryeh, Tony Nguyen, Thomas Su

Due Date: May 4th, 2020

Abstract:

We all are aware of how machine learning has revolutionized our world in recent years and has made a variety of complex tasks much easier to perform. The recent breakthroughs in implementing Deep learning techniques has shown that superior algorithms and complex architectures can impart human-like abilities to machines for specific tasks. Several machine learning algorithms have been increasingly utilized for cardiovascular disease prediction. We aim to assess and summarize the overall predictive ability of ML algorithms in cardiovascular diseases. We also hope to use hypothesis testing to determine if cholesterol and high blood pressure contribute to coronary heart disease.

Table of Contents

Research Questions:

- Which total cholesterol level is associated with high risk for heart disease?
- What level of blood pressure is associated with higher risk for heart disease?
- Which Machine Learning Techniques accurately predicts the leading features causing heart disease?

Introduction

In this report, we will start by exploring our data through analysis such as correlation matrix, heat map and graphs such as histograms, bar plots and box plots to visually explain the data. Then we will move onto our first 2 research questions using hypothesis testing to test if there is a correlation between cholesterol levels or blood pressure level and heart diseases.

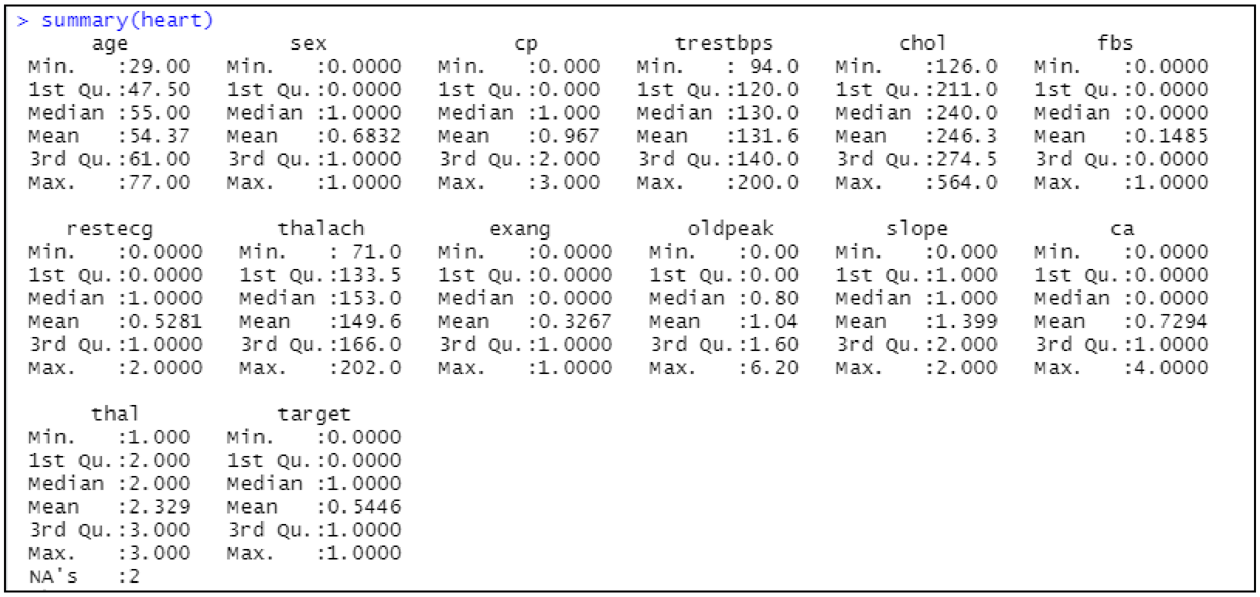
After that, we will use machine learning techniques to test features that can predict cardiovascular diseases to help answer our 3rd research question. We will assess how accurate the predictability of these Machine Learning techniques are in relation to cardiovascular diseases. In Particular, we will be looking at Decision trees where we will use tree-like structures. Decision trees will test out a feature and make a decision after testing it. In addition, we will look at Random Forest which creates more than 1 decision tree. This will create multiple decision trees which will allow us to test out more features and make stronger decisions. We will also look at a Principal component analysis (PCA) to reduce the dimensionality of the data set and transform a large set of variables into a smaller one, which will make the data easier to look at. Lastly, we will explore model building using K-mean clustering and K-Nearest Neighbor (KNN) algorithm to select features with highest accuracy that depict correlation to heart disease.

Data Summary:

In this project, we will use data collected from Kaggle and can be accessed through the following link: https://www.kaggle.com/ronitf/heart-disease-uci

The dataset has 303 observations and 14 attributes (13 attributes and 1 target column). Note that the variables that are within parenthesis (--) are Categorical - Nominal , while those without are Numerical - Discrete.

1. **Age** is the age of the candidate.
2. **(Sex)** has numeric values. 1 is male and 0 is female.
3. **(cp)** Chest Pain pain has values between 0-3. The types of angina that are described in the research paper. The higher the number, the lesser are the odds of heart attack. Value 1: typical angina, Value 2: atypical angina, Value 3: non-anginal pain.
4. **trestbps** Resting blood pressure is normal pressure with no exercise.
5. **chol** Cholesterol means the blockage for blood supply in the blood vessels.
6. **(fbs)** Fasting blood sugar > 120 mg/dl (1 = true; 0 = false) blood sugar taken after a long gap between a meal and the test. Typically, it's taken before any meal in the morning.
7. **(restecg)** Rest ECG results means ECG values taken while a person is on rest which means no exercise and normal functioning of heart is happening.
8. **thalach** The Maximum Heart Rate achieved.
9. **(exang)** Exercise induced angina (1 = yes; 0 = no) is chest pain while exercising or doing any physical activity.
10. **oldpeak** ST Depression is the difference between the value of ECG at rest and after exercise.
11. **(slope)** The slope of the peak exercise ST segment. Value 1: upsloping, Value 2: flat, Value 3: downsloping.
12. **(ca)** The number of major blood vessels (0-3) supplying blood to the heart is blocked.
13. **(thal)** The Types of thalassemia (0 = NA, 1 = normal; 2 = fixed defect; 3 = reversible defect).
14. **(target)** (predicted attribute): diagnosis of heart disease (angiographic disease status): Value 0: < 50% diameter narrowing, Value 1: > 50% diameter narrowing.



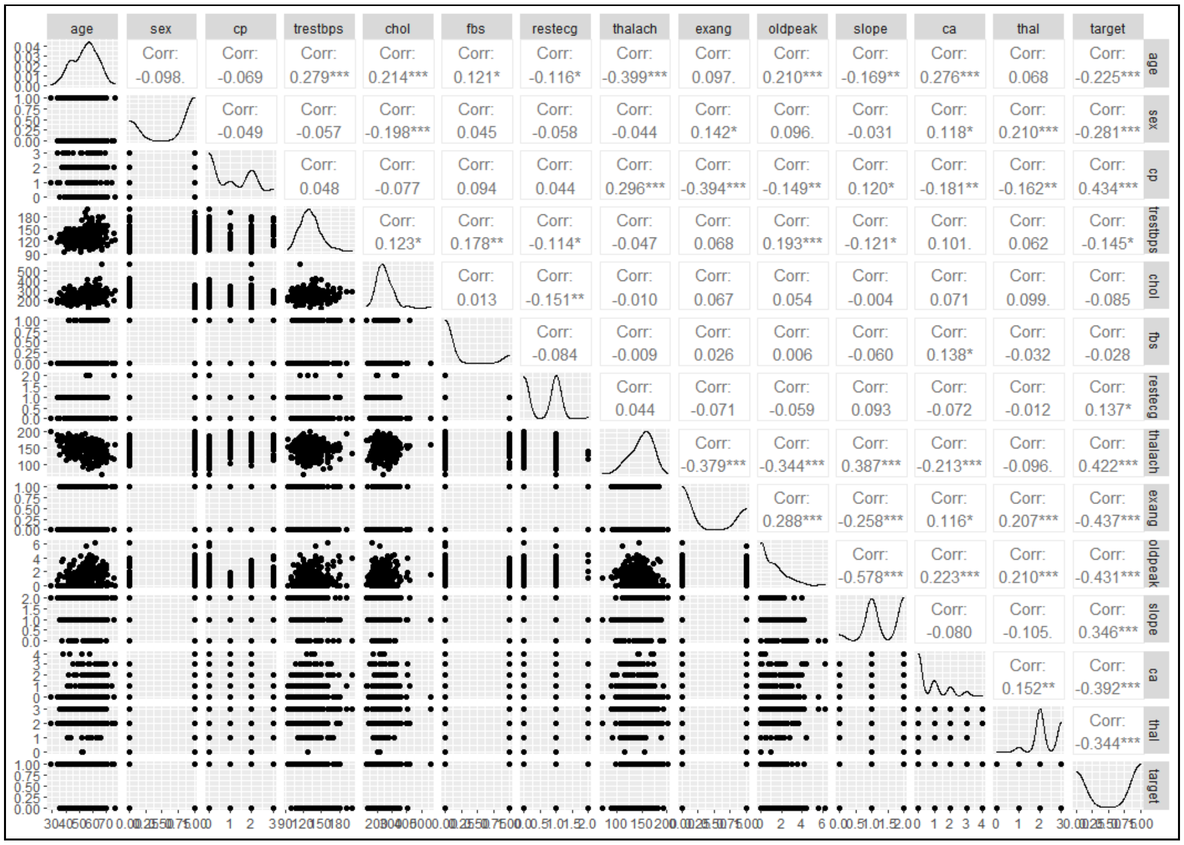
```
> summary(heart)
      age             sex               cp            trestbps          chol            fbs        
 Min.   :29.00   Min.   :0.0000   Min.   :0.000   Min.   : 94.0   Min.   :126.0   Min.   :0.0000  
 1st Qu.:47.50   1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:120.0   1st Qu.:211.0   1st Qu.:0.0000  
 Median :55.00   Median :1.0000   Median :1.000   Median :130.0   Median :240.0   Median :0.0000  
 Mean   :54.37   Mean   :0.6832   Mean   :0.967   Mean   :131.6   Mean   :246.3   Mean   :0.1485  
 3rd Qu.:61.00   3rd Qu.:1.0000   3rd Qu.:2.000   3rd Qu.:140.0   3rd Qu.:274.5   3rd Qu.:0.0000  
 Max.   :77.00   Max.   :1.0000   Max.   :3.000   Max.   :200.0   Max.   :564.0   Max.   :1.0000  
                                                                                                  
    restecg          thalach          exang           oldpeak         slope             ca        
 Min.   :0.0000   Min.   : 71.0   Min.   :0.0000   Min.   :0.00   Min.   :0.000   Min.   :0.0000  
 1st Qu.:0.0000   1st Qu.:133.5   1st Qu.:0.0000   1st Qu.:0.00   1st Qu.:1.000   1st Qu.:0.0000  
 Median :1.0000   Median :153.0   Median :0.0000   Median :0.80   Median :1.000   Median :0.0000  
 Mean   :0.5281   Mean   :149.6   Mean   :0.3267   Mean   :1.04   Mean   :1.399   Mean   :0.7294  
 3rd Qu.:1.0000   3rd Qu.:166.0   3rd Qu.:1.0000   3rd Qu.:1.60   3rd Qu.:2.000   3rd Qu.:1.0000  
 Max.   :2.0000   Max.   :202.0   Max.   :1.0000   Max.   :6.20   Max.   :2.000   Max.   :4.0000  
                                                                                                  
      thal           target      
 Min.   :1.000   Min.   :0.0000  
 1st Qu.:2.000   1st Qu.:0.0000  
 Median :2.000   Median :1.0000  
 Mean   :2.329   Mean   :0.5446  
 3rd Qu.:3.000   3rd Qu.:1.0000  
 Max.   :3.000   Max.   :1.0000  
 NA's   :2                       
```

Data Exploration

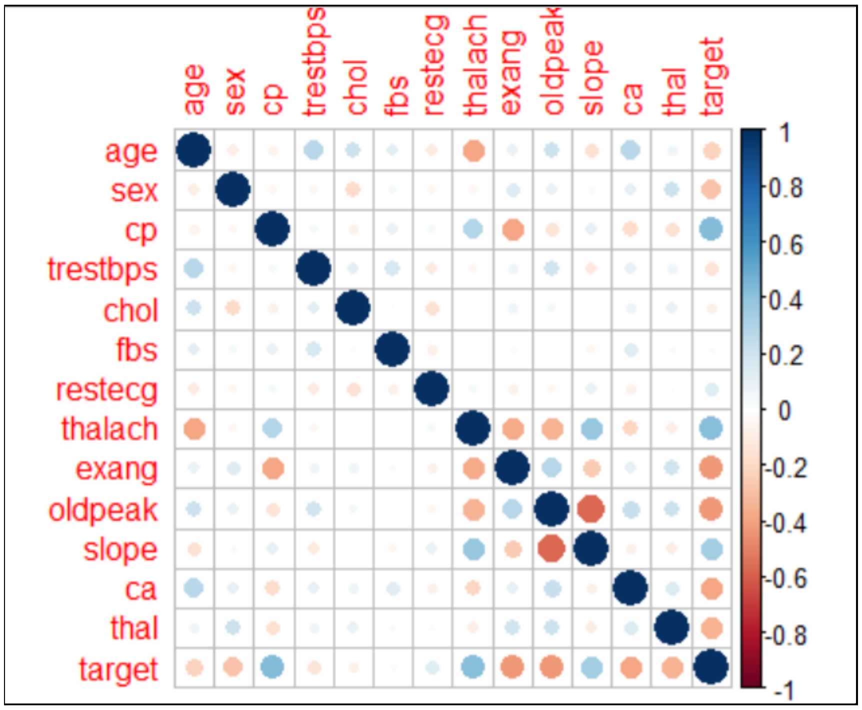
Correlation Matrix

A correlation matrix is a table showing correlation coefficients between variables. Each cell in the table shows the correlation between two variables. A correlation matrix is used to summarize data, as an input into a more advanced analysis, and as a diagnostic for advanced analyses.

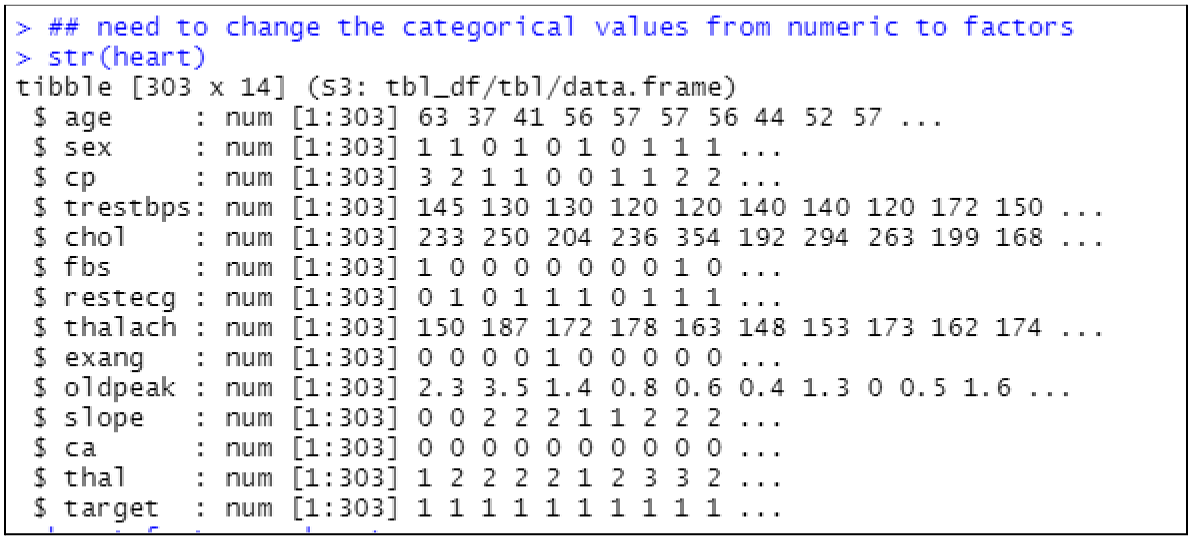
Using the ggpairs() function in the R package ggplot2, we are able to display a scatter plot matrix, where our correlations are shown in the top right triangle, and numeric variables are drawn on the bottom left half of the triangle. Through a quick overview, there doesn't appear to be a strong linear relationship between the variables. The highest value appears to be slope and oldpeak at -0.578.



We can also use the function corrplot() in the R package corrplot to generate a heatmap. By doing so we can view the data in a visually colorful way and identify possible multicollinearity. The color in the plot varies based on the sign/ direction of the correlation: blue for positive and red for negative signs. While the size of the circle demonstrated the strength of the correlation: the larger the circle, the closer it is to 1 (or -1). Once again we can see that the highest relationship value between two variables appears to be slope and oldpeak at -0.578. This could be due to the fact that out of the 13 variables we have 8 of them are categorical.

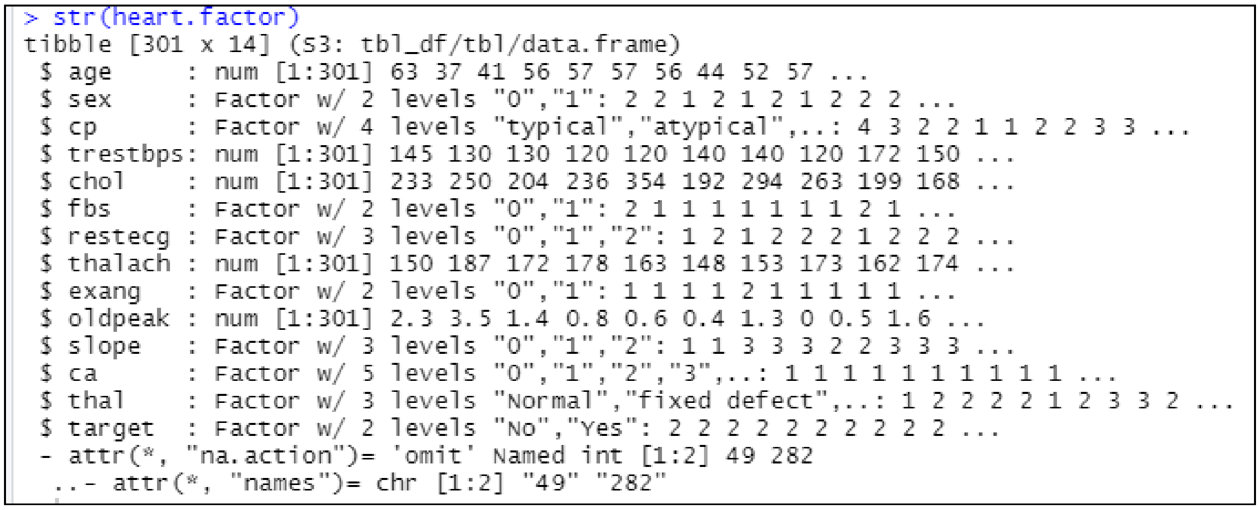


Before moving onto our research question, we use the functions summary() and str() to see that the heart dataset has all variables in numeric form. We need to change a few of the variables into factors using as.factor()



```
> ## need to change the categorical values from numeric to factors
> str(heart)
tibble [303 x 14] (S3: tbl_df/tbl/data.frame)
 $ age     : num [1:303] 63 37 41 56 57 57 56 44 52 57 ...
 $ sex     : num [1:303] 1 1 0 1 0 1 0 1 1 1 ...
 $ cp      : num [1:303] 3 2 1 1 0 0 1 1 2 2 ...
 $ trestbps: num [1:303] 145 130 130 120 120 140 140 120 172 150 ...
 $ chol    : num [1:303] 233 250 204 236 354 192 294 263 199 168 ...
 $ fbs     : num [1:303] 1 0 0 0 0 0 0 0 1 0 ...
 $ restecg : num [1:303] 0 1 0 1 1 1 0 1 1 1 ...
 $ thalach : num [1:303] 150 187 172 178 163 148 153 173 162 174 ...
 $ exang   : num [1:303] 0 0 0 0 1 0 0 0 0 0 ...
 $ oldpeak : num [1:303] 2.3 3.5 1.4 0.8 0.6 0.4 1.3 0 0.5 1.6 ...
 $ slope   : num [1:303] 0 0 2 2 2 1 1 2 2 2 ...
 $ ca      : num [1:303] 0 0 0 0 0 0 0 0 0 0 ...
 $ thal    : num [1:303] 1 2 2 2 2 1 2 3 3 2 ...
 $ target  : num [1:303] 1 1 1 1 1 1 1 1 1 1 ...
```

Now we are ready to test our classification model.

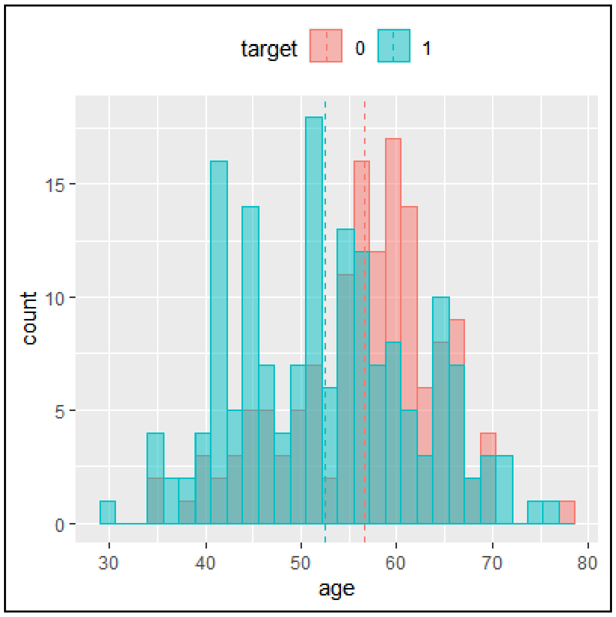


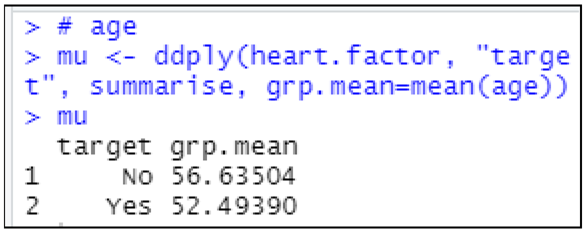
```
> str(heart.factor)
tibble [301 x 14] (S3: tbl_df/tbl/data.frame)
 $ age     : num [1:301] 63 37 41 56 57 57 56 44 52 57 ...
 $ sex     : Factor w/ 2 levels "0","1": 2 2 1 2 1 2 1 2 2 2 ...
 $ cp      : Factor w/ 4 levels "typical","atypical",..: 4 3 2 2 1 1 2 2 3 3 ...
 $ trestbps: num [1:301] 145 130 130 120 120 140 140 120 172 150 ...
 $ chol    : num [1:301] 233 250 204 236 354 192 294 263 199 168 ...
 $ fbs     : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 1 2 1 ...
 $ restecg : Factor w/ 3 levels "0","1","2": 1 2 1 2 2 2 1 2 2 2 ...
 $ thalach : num [1:301] 150 187 172 178 163 148 153 173 162 174 ...
 $ exang   : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 1 1 1 1 ...
 $ oldpeak : num [1:301] 2.3 3.5 1.4 0.8 0.6 0.4 1.3 0 0.5 1.6 ...
 $ slope   : Factor w/ 3 levels "0","1","2": 1 1 3 3 3 2 2 3 3 3 ...
 $ ca      : Factor w/ 5 levels "0","1","2","3",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ thal    : Factor w/ 3 levels "Normal","fixed defect",..: 1 2 2 2 2 1 2 3 3 2 ...
 $ target  : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 2 2 2 2 2 ...
 - attr(*, "na.action")= 'omit' Named int [1:2] 49 282
  ..- attr(*, "names")= chr [1:2] "49" "282"
```

Histograms

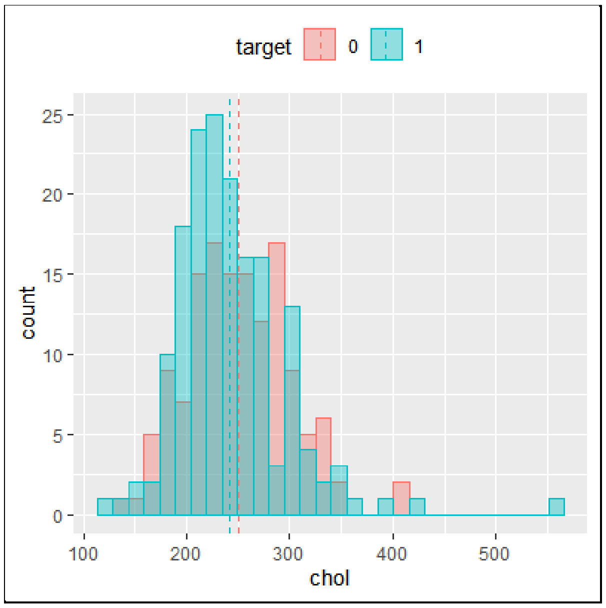
Using histograms we can see in detail the distribution of some of the features, this shows us where most of those who do and don't have heart disease lie near. The distribution of each graph shows where they fall under each count. There is also a presence of the mean. The mean shows us where the average of our target occurrences lie. We have a mean for those who do not have heart disease and a mean for those who do in each feature. The bar with pink "0" shows us data for those that do not have heart disease while the data in blue "1" show us those who do have heart disease.

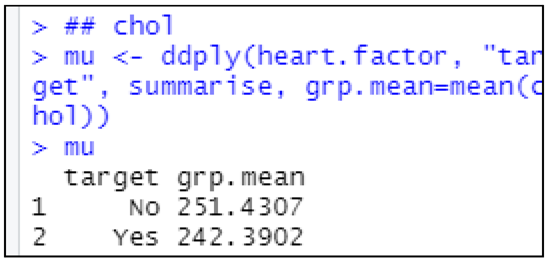
For the age group we see the histogram is close to normal distribution for those that do have heart disease. There are outliers such as those that range in 40 years or so. In contrast for those who do not have heart disease the histogram is skewed left. We can see that the mean for both is very similar. Those that do not have heart disease have a mean of 56.6, while those who do have a heart disease have a mean of 52.5. We can assume from this data that age is not a large influencer for heart disease; however, younger cases do have a higher chance of having heart disease compared to older cases.





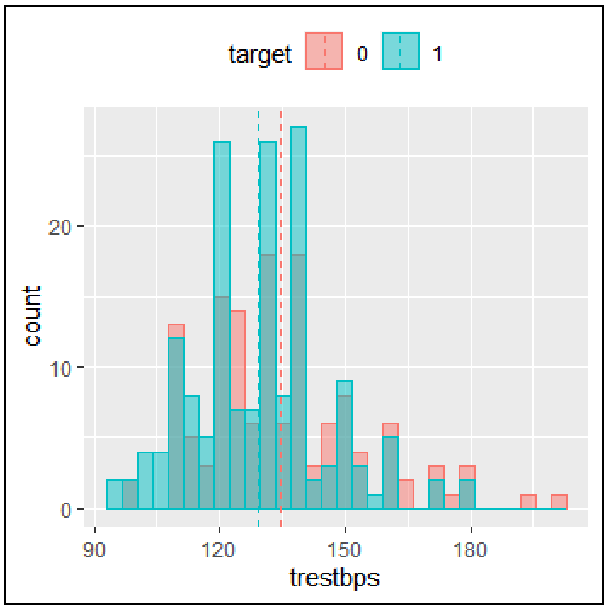
```
> # age
> mu <- ddply(heart.factor, "targe
t", summarise, grp.mean=mean(age))
> mu
  target grp.mean
1     No 56.63504
2    Yes 52.49390
```

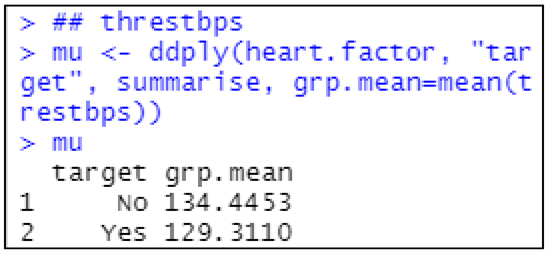




```
> ## chol
> mu <- ddply(heart.factor, "tar
get", summarise, grp.mean=mean(c
hol))
> mu
  target grp.mean
1     No 251.4307
2    Yes 242.3902
```

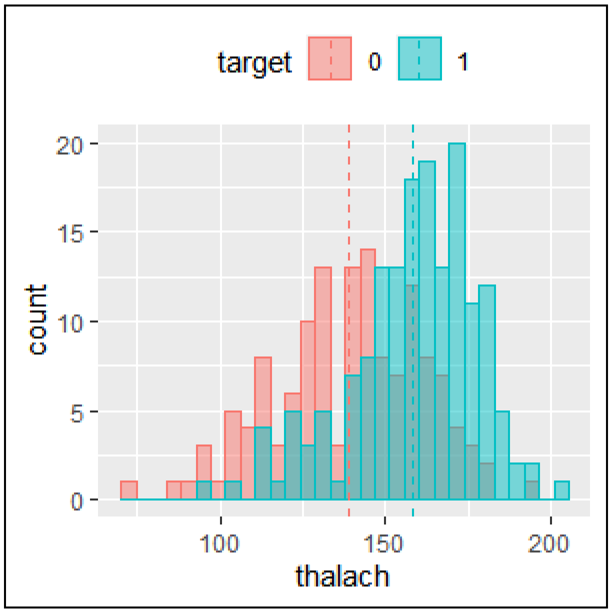
Cholesterol levels can play a factor on whether you get heart diseases or not. In our data we see that the cholesterol feature for both those that have and don't have heart diseases is skewed to the right. There are more occurrences for those that do have heart disease in the range of 240-245mg/dL than those who do not have a heart disease. The mean for those that do not have a heart disease is 251.4 and the mean for those that do is 242.4. They both have a slightly close mean. Our normal cholesterol rate is between 120- 200 mg. Based on this data the averages with and without heart diseases that have a cholesterol level that is 200mg and higher, therefore suggesting that chol might not be a strong influencer for heart disease.

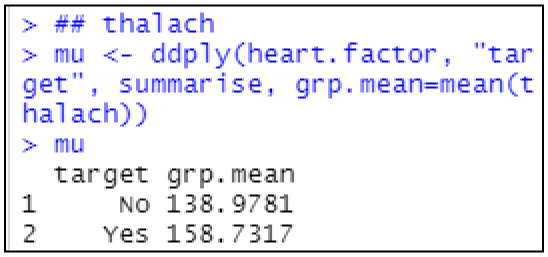




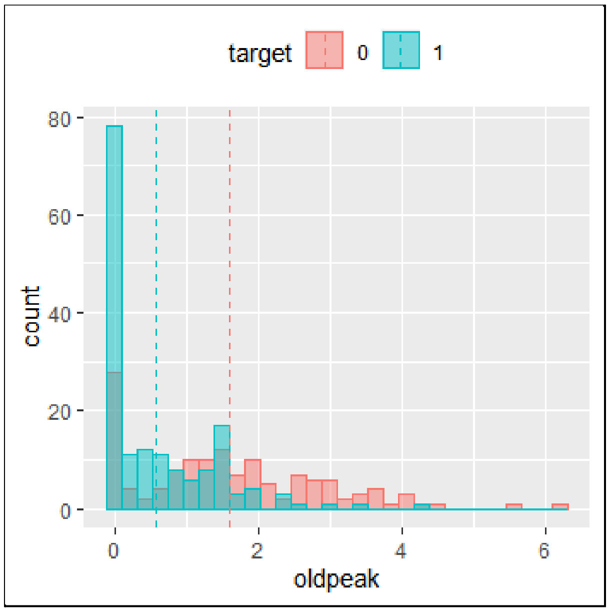
```
> ## threstbps
> mu <- ddply(heart.factor, "tar
get", summarise, grp.mean=mean(t
restbps))
> mu
  target grp.mean
1     No 134.4453
2    Yes 129.3110
```

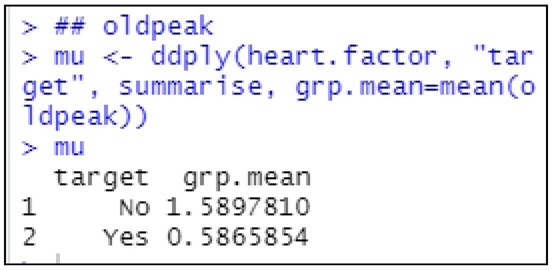
In addition, for this histogram we see the feature resting blood pressure. There seems to be slight right skewness and more occurrences of those that do have a heart disease around the range of 120-140 resting blood pressure. In contrast for those that do not have a heart disease there are less occurrences for those in the same ranges (lower counts of pink). The mean for those who have heart disease is 129.3 /80 mm Hg while the heart disease for those that do not is 134.4 mm Hg. Normal resting blood pressure is defined at 120 mm Hg and based on this data set, on average, both those who have and do not have heart disease show high resting blood pressure. We can even observe a few high resting blood levels at over 180 mm Hg(pink) outliers that are not heart disease cases. Therefore, suggesting that there is not sufficient evidence that trestbps is or is not a strong influencer for heart disease.




```
> ## thalach
> mu <- ddply(heart.factor, "tar
get", summarise, grp.mean=mean(t
halach))
> mu
  target grp.mean
1     No 138.9781
2    Yes 158.7317
```

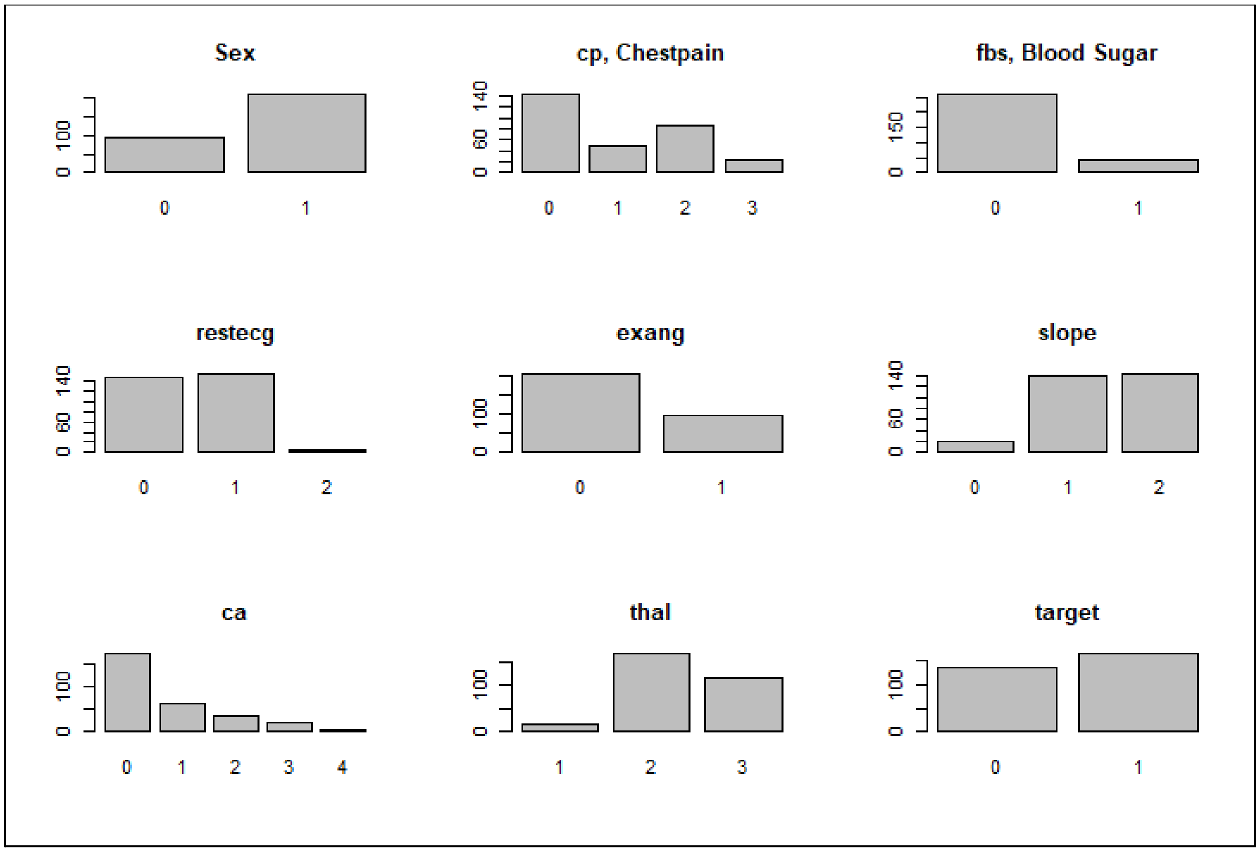
Thalach is the maximum heart rate achieved. Based on this data we notice how for those who do not have a heart disease there is a more normal distribution than those who do. In fact for those who do, the data is skewed to the left. The mean for those that do not have heart disease is 138.98 bpm while the mean for those that do is 158.7bpm. This clearly illustrates how the higher your maximum heart rate the more likely you are to have a heart disease.




```
> ## oldpeak
> mu <- ddply(heart.factor, "tar
get", summarise, grp.mean=mean(o
ldpeak))
> mu
  target  grp.mean
1     No 1.5897810
2    Yes 0.5865854
```

For the variable, oldpeak, we see how for those who do have heart disease the data is skewed to the right, while those who do not have a distribution close to being normal. The mean for those that do not have heart disease is 1.6 while the mean for those that do have a heart disease is .6. The difference in means is big and with this we can assume that it could have an influence on whether you get heart disease or not.

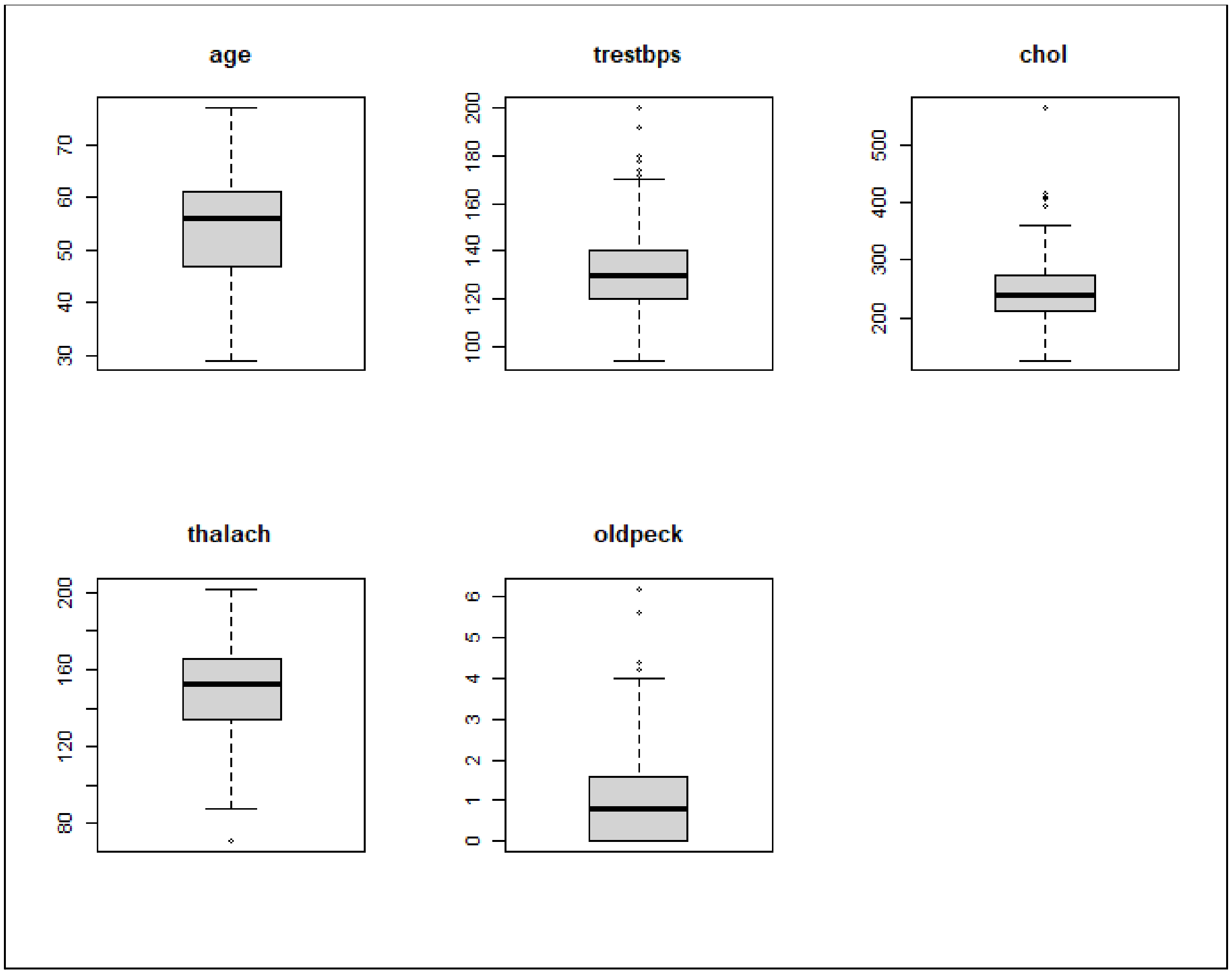
Bar plots for the other 9 variables:



The bar graphs show the occurrences between each feature. We observe that there appears to be twice the amount of Females in this data set than there are Male and could have an influence in the results of cases that have heart disease. We also notice how there are 4 different chest pains but the first type has more of an effect when it comes to heart disease. In addition, we see how for fasting blood sugar there are more occurences for it being less than 120 mg/dl, for exang, exercise induced angina, there are many more occurrences for it not happening. In addition,the most occurrences for the slope is upsloping and flat value, the number of blood vessels blocked are 0, the most occurrences for types of thalassemia (Thal) is fixed defect, and the most occurrences for diagnosis for heart disease is >50% diameter narrowing . All of this plays a role in heart disease.

Box plots:

Box plot gives us a good image of the first quartile, third quartile, median, minimum value, maximum value and outliers. This shows us how tightly grouped the data is and gives us a good idea of where the features lie when it comes to heart disease. We can see that for age, there are no outliers. While in resting blood pressure, cholesterol level,thalach(maximum heart rate achieved), and ST depression (oldpeck) have a few outliers, showing how it will not always follow the statistics for these features. This means that there will be people who will or will not have heart disease and do not follow the data that is correlated with heart disease.

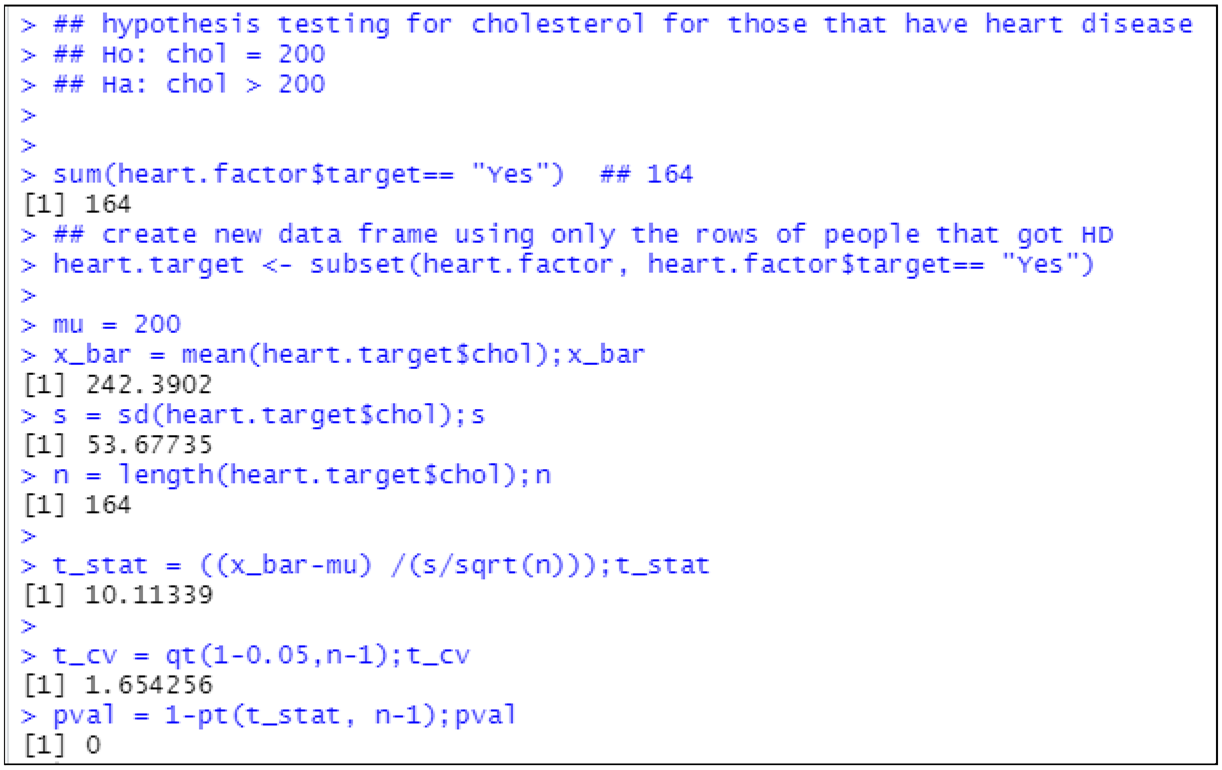


Hypothesis Testing

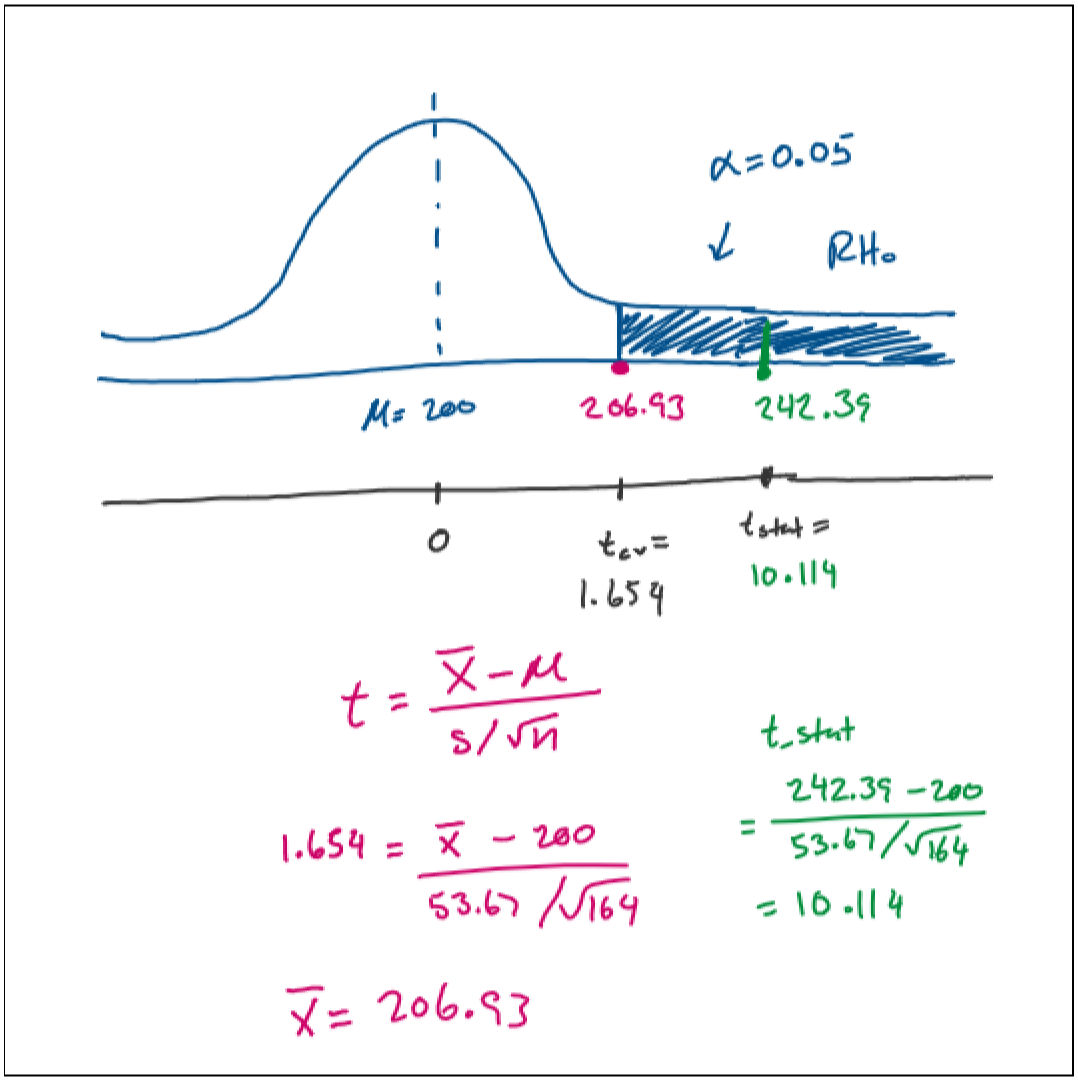
- Which total cholesterol level is associated with high risk for heart disease?

According to Cleveland Clinic, a normal total cholesterol level is between 125 to 200mg/dL for adults over the age of 20. Assuming this data is using the total cholesterol level, and our data contains ages from 29 to 77, we will set up our hypothesis to test whether high cholesterol has an impact on Heart Disease.

We set the Null hypothesis Ho: mu = 200 and the Alternative Ha: mu > 200 for the cases that had a target = “Yes” for Heart Disease. We find that out of the 301 cases, 164 have a target = “Yes” for Heart Disease; therefore a new data frame is created for these cases and tested using the t-test statistic since the population σ is unknown and we are testing a sample.



```
> ## hypothesis testing for cholesterol for those that have heart disease
> ## Ho: chol = 200
> ## Ha: chol > 200
>
>
> sum(heart.factor$target== "Yes")  ## 164
[1] 164
> ## create new data frame using only the rows of people that got HD
> heart.target <- subset(heart.factor, heart.factor$target== "Yes")
>
> mu = 200
> x_bar = mean(heart.target$chol);x_bar
[1] 242.3902
> s = sd(heart.target$chol);s
[1] 53.67735
> n = length(heart.target$chol);n
[1] 164
>
> t_stat = ((x_bar-mu) /(s/sqrt(n)));t_stat
[1] 10.11339
>
> t_cv = qt(1-0.05,n-1);t_cv
[1] 1.654256
> pval = 1-pt(t_stat, n-1);pval
[1] 0
```

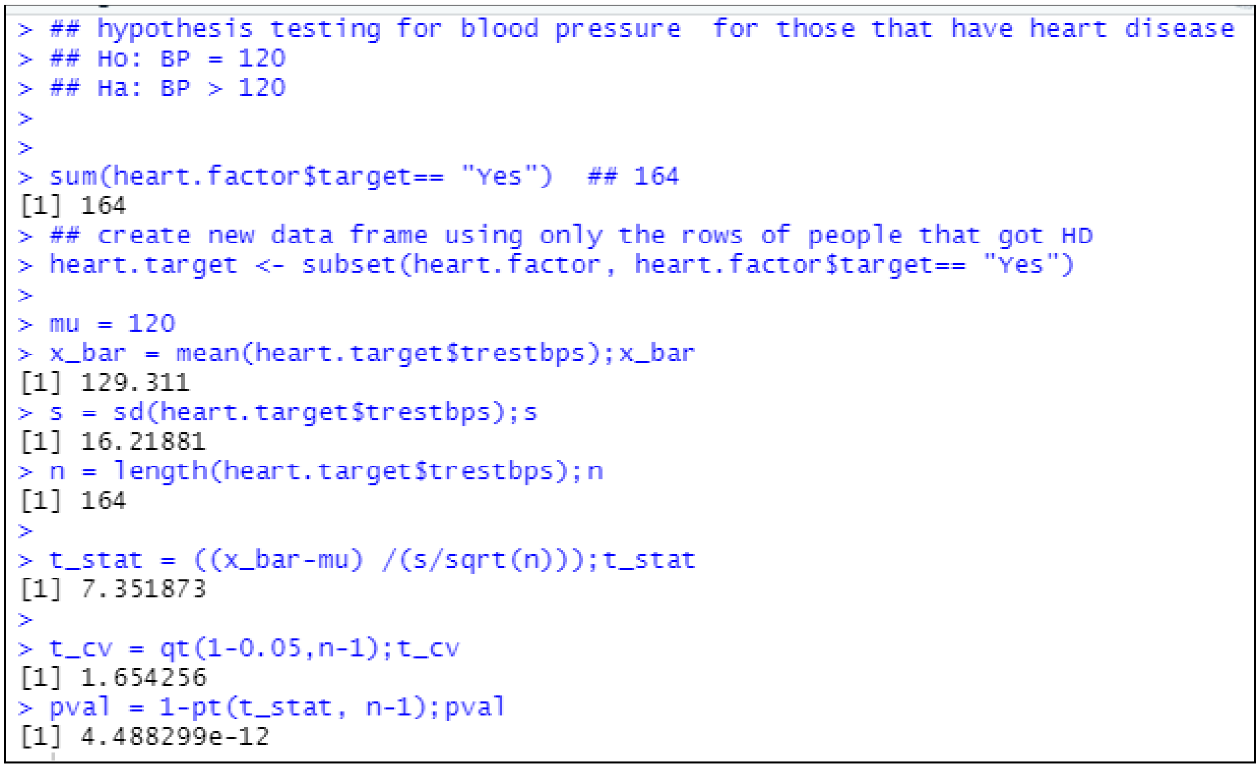

$\alpha = 0.05$
RH_0
$\mu = 200$
206.93
242.39
0
$t_{cv} = 1.654$
$t_{stat} = 10.114$
$t = \frac{\bar{x} - \mu}{s/\sqrt{n}}$
$1.654 = \frac{\bar{x} - 200}{53.67/\sqrt{164}}$
$\bar{x} = 206.93$
$t\_stat = \frac{242.39 - 200}{53.67/\sqrt{164}} = 10.114$

We find that the **t_stat = 10.11 > t_cv = 1.65** and **p-value = 0 < alpha** ∝ **= 0.05** Therefore we can conclude that there is sufficient evidence, at significance level ∝ = **0.05**; We can reject the null hypothesis and conclude that the true mean of Cholesterol is higher than 200mg/dL.

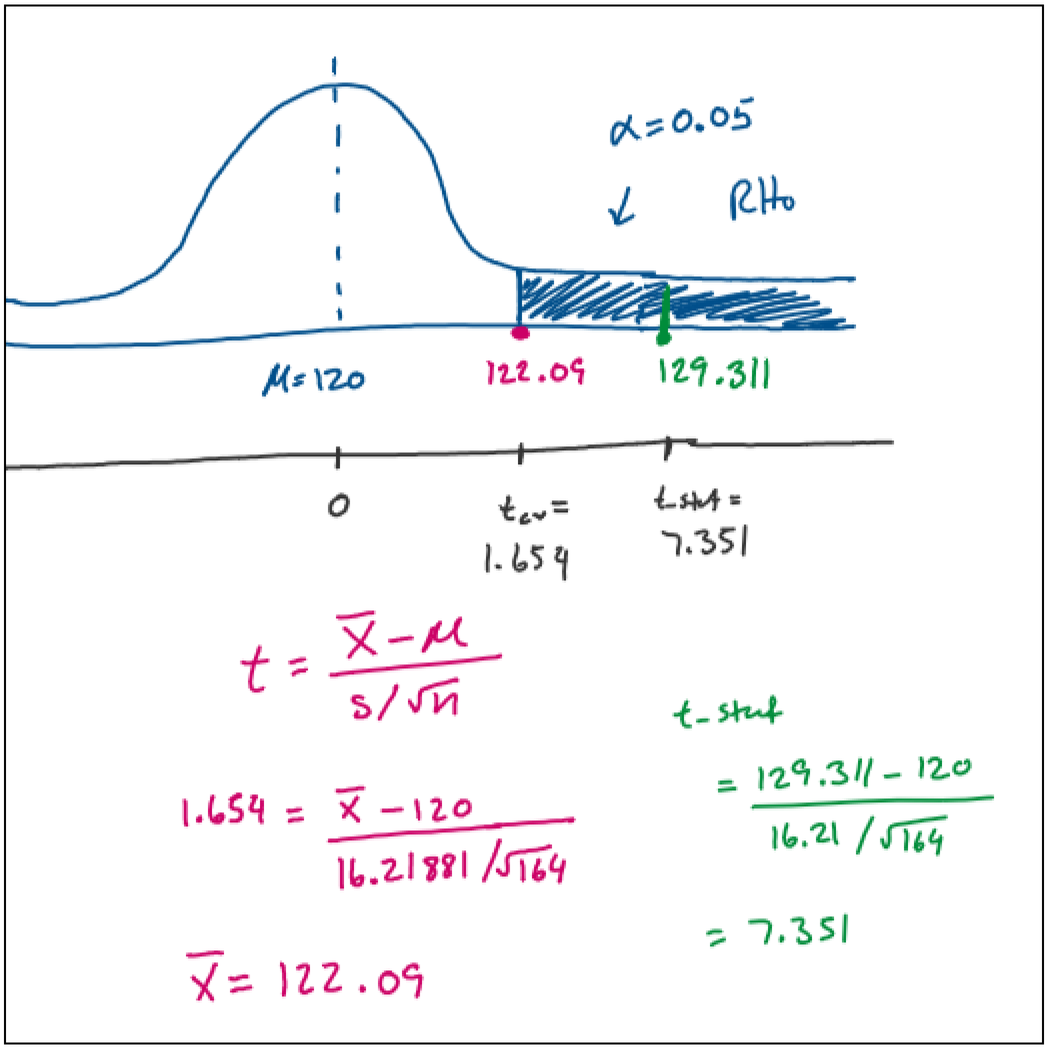
- What level of blood pressure is associated with higher risk for heart disease?

  According to the American Heart Association, a normal Systolic/Diastolic Blood Pressure level is set at 120/80 mg Hg. Assuming this data is using the Systolic level, we will set up our hypothesis to test whether high Blood Pressure has an impact on Heart Disease.

We set the Null hypothesis Ho: mu = 120 and the Alternative Ha: mu > 120 for the cases that had a target = "Yes" for Heart Disease. Using the data frame from earlier, we have 164 cases where target = "Yes" for Heart Disease and calculate our hypothesis using the t-test statistic since the population σ is unknown and we are testing a sample.



```
> ## hypothesis testing for blood pressure  for those that have heart disease
> ## Ho: BP = 120
> ## Ha: BP > 120
>
>
> sum(heart.factor$target== "Yes")  ## 164
[1] 164
> ## create new data frame using only the rows of people that got HD
> heart.target <- subset(heart.factor, heart.factor$target== "Yes")
>
> mu = 120
> x_bar = mean(heart.target$trestbps);x_bar
[1] 129.311
> s = sd(heart.target$trestbps);s
[1] 16.21881
> n = length(heart.target$trestbps);n
[1] 164
>
> t_stat = ((x_bar-mu) /(s/sqrt(n)));t_stat
[1] 7.351873
>
> t_cv = qt(1-0.05,n-1);t_cv
[1] 1.654256
> pval = 1-pt(t_stat, n-1);pval
[1] 4.488299e-12
```


$\alpha = 0.05$
RHo
$\mu = 120$
122.09
129.311
0
$t_{cv} =$ 1.654
$t_{stat} =$ 7.351
$t = \frac{\bar{X} - \mu}{s/\sqrt{n}}$
$1.654 = \frac{\bar{X} - 120}{16.21881/\sqrt{164}}$
$\bar{X} = 122.09$
t_stat
$= \frac{129.311 - 120}{16.21/\sqrt{164}}$
$= 7.351$

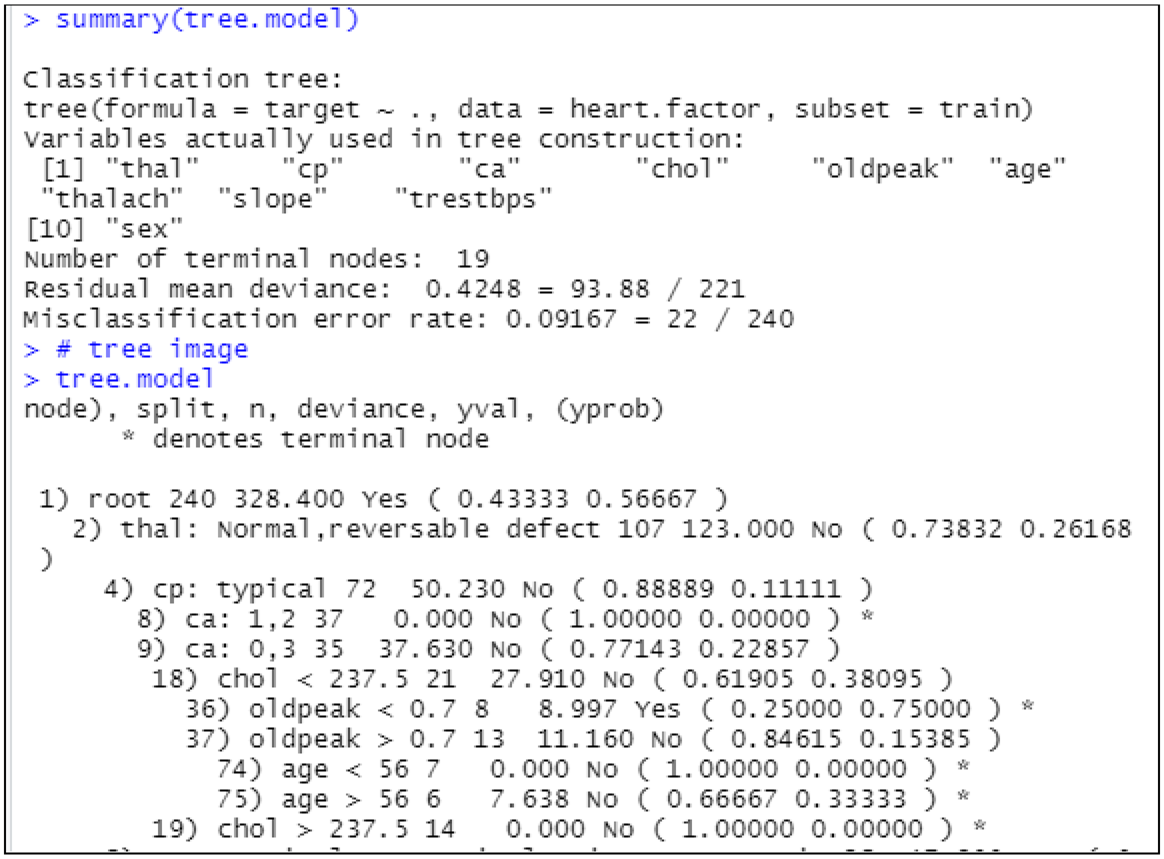
We find that the **t_stat = 7.351 > t_cv = 1.65** and **p-value = 4.488e-12 < alpha** ∝ **= 0.05** Therefore we can conclude that there is sufficient evidence at significance level ∝ = **0.05**; We can reject the null hypothesis and conclude that the true mean of Blood Pressure is higher than 120 mg.

Machine Learning Techniques (Classification)

Decision Tree

A decision tree is a flowchart-like structure in which each internal node represents a test on a feature, each leaf node represents a class label (decision taken after computing all features) and branches represent conjunctions of features that lead to those class labels. The paths from root to leaf represent classification rules. Decision trees are a non-parametric supervised learning method used for both classification and regression tasks and are most widely used and practical methods for supervised learning as well as predictive modelling approaches used in statistics, data mining and machine learning. They are constructed via an algorithmic approach that identifies ways to split a data set based on different conditions. In this report we want to build a tree model that can evaluate which of the 13 variables has the most influence on the response variable, heart disease, as well as find the best performance of this classification tree and observe whether pruning the tree will help with its performance.

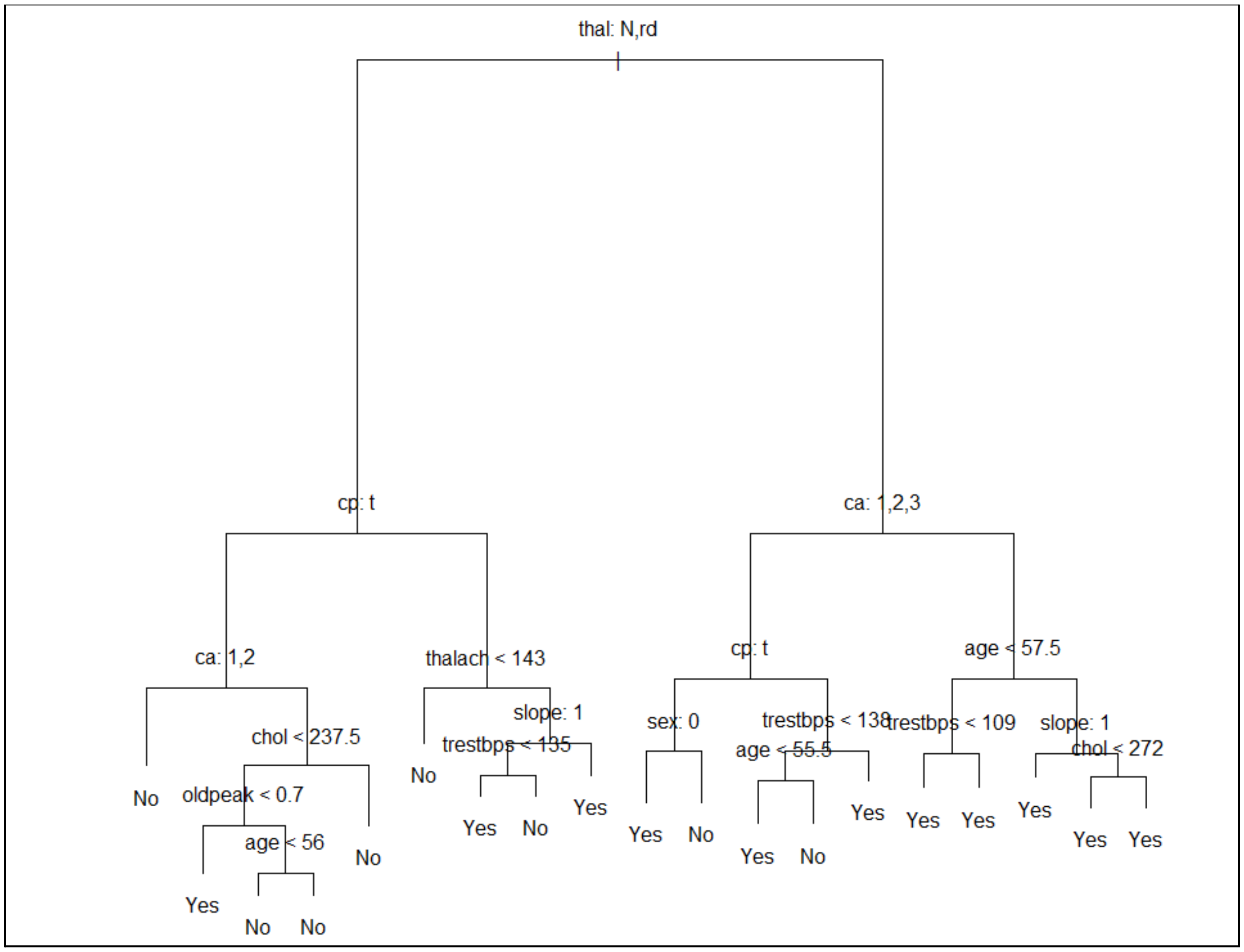
Before we get started we need to organize our data by omitting missing row values and converting some variables into factors such as our ***target***: "Yes" == yes the case has heart disease, and "No" == no the case does not have heart disease. We result with 301 cases along with 13 variables and our 1 response variable. Using the function ***sample(),*** we split 80/20 train/test for the data and use the function ***tree()*** to fit a classification in order to predict our ***target***. Below we observe the summary of this tree model at set.seed(111).



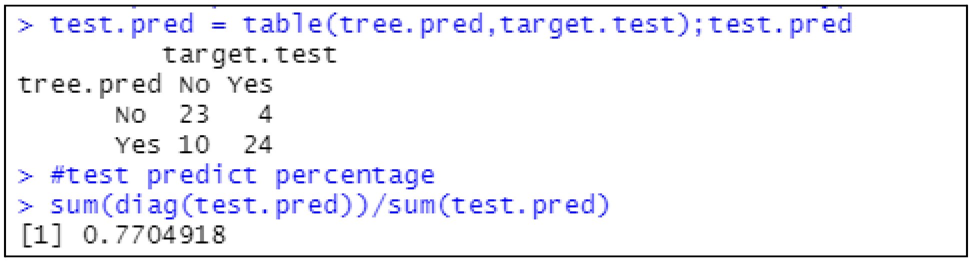
```
> summary(tree.model)

Classification tree:
tree(formula = target ~ ., data = heart.factor, subset = train)
Variables actually used in tree construction:
 [1] "thal"     "cp"       "ca"       "chol"     "oldpeak"  "age"
 "thalach"  "slope"    "trestbps"
[10] "sex"
Number of terminal nodes:  19
Residual mean deviance:  0.4248 = 93.88 / 221
Misclassification error rate: 0.09167 = 22 / 240
> # tree image
> tree.model
node), split, n, deviance, yval, (yprob)
      * denotes terminal node

 1) root 240 328.400 Yes ( 0.43333 0.56667 )
   2) thal: Normal,reversable defect 107 123.000 No ( 0.73832 0.26168
 )
     4) cp: typical 72  50.230 No ( 0.88889 0.11111 )
       8) ca: 1,2 37   0.000 No ( 1.00000 0.00000 ) *
       9) ca: 0,3 35  37.630 No ( 0.77143 0.22857 )
        18) chol < 237.5 21  27.910 No ( 0.61905 0.38095 )
          36) oldpeak < 0.7 8   8.997 Yes ( 0.25000 0.75000 ) *
          37) oldpeak > 0.7 13  11.160 No ( 0.84615 0.15385 )
            74) age < 56 7   0.000 No ( 1.00000 0.00000 ) *
            75) age > 56 6   7.638 No ( 0.66667 0.33333 ) *
        19) chol > 237.5 14   0.000 No ( 1.00000 0.00000 ) *
```

Residual mean deviance indicates whether the tree model provides a good fit to the training data, for our model we have a mean deviance of 42.48%, which is a bit low, with a reasonable training classification error rate at 9.16%.

We can observe that only 10 of our total 13 variables are used as internal nodes in the model and that a total of 19 (branches) terminal nodes are used. We can see that the nodes used in the tree model are indicated with "*", 5 of which are shown in the screenshot above: "ca", "oldpeak", "age", "age", and "chol". Notice age is repeated and that is the case for a few variables. The more it appears in a slip, indicates the importance of that variable. In this model, "age" and "trestbps" are shown to have 3 splits. In this tree.model, "thal" is the most important indicator for ***target*** since it is the first branch and only indicates whether the blood disorder is normal or reversible defect (very bad to have). Therefore, these 5 variables (“thal”, “ca”, “oldpeak”, “age”, and “chol”) can be big influencers when it comes to heart disease.

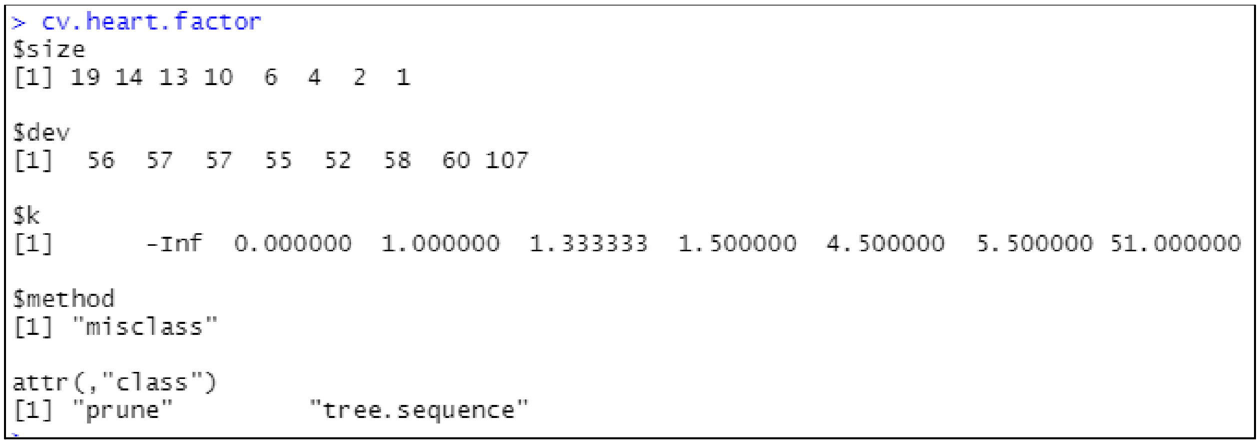


Using the ***predict()*** function we calculate the actual class prediction (***target*** values we predict to be “Yes” or “No” and where actually “Yes” or “No”) of our test set to be 77.04%. This is a fair performance, however, it can be improved using cost complexity pruning.



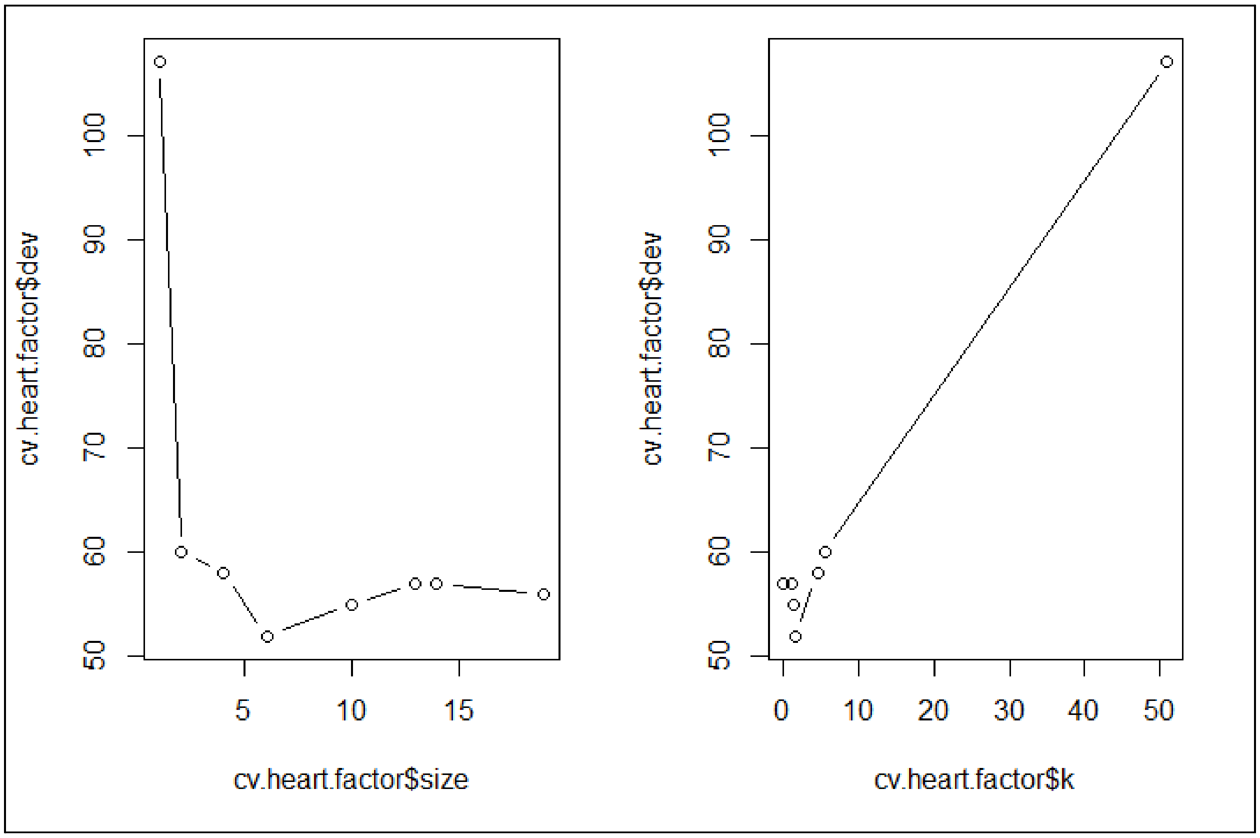
```
> test.pred = table(tree.pred,target.test);test.pred
         target.test
tree.pred No Yes
      No  23   4
      Yes 10  24
> #test predict percentage
> sum(diag(test.pred))/sum(test.pred)
[1] 0.7704918
```

Using the function ***cv.tree()*** we prune the classification and find that a tree of 8 terminal nodes gives the best results in the lowest cross validation error rate and we can see that in the table and graphs below.

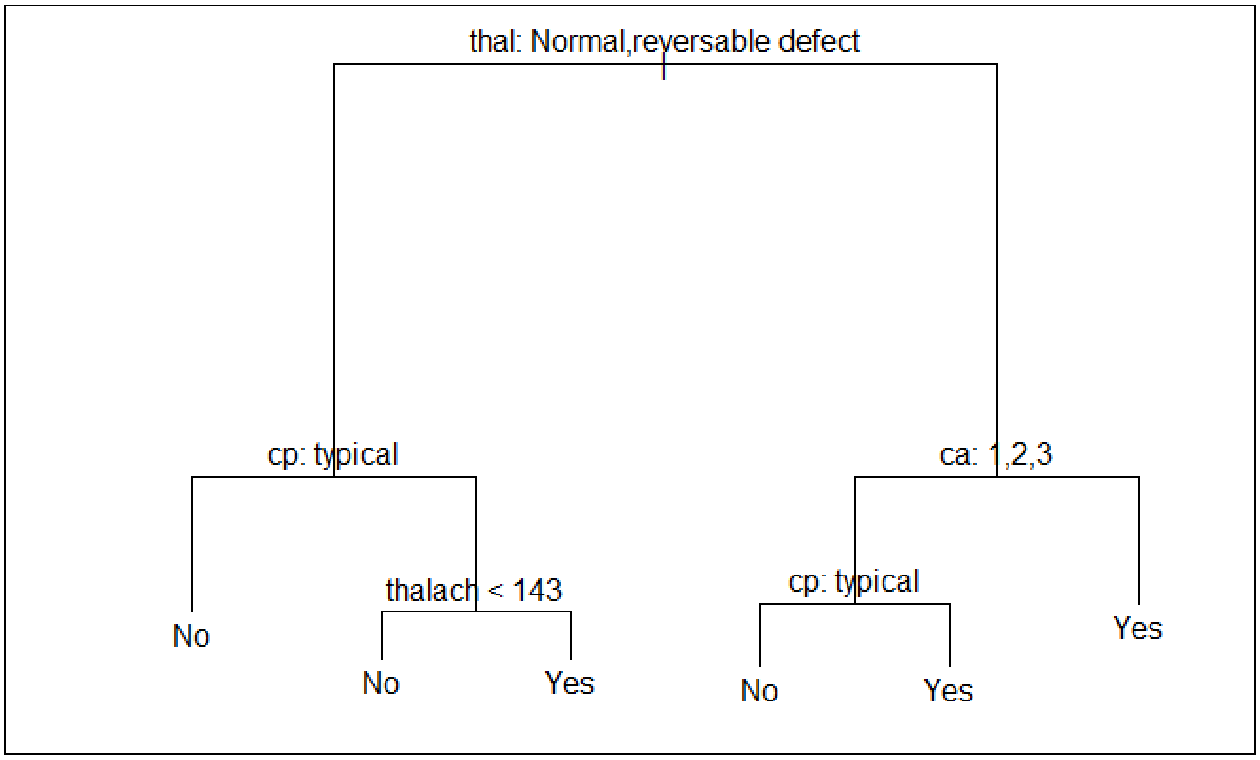


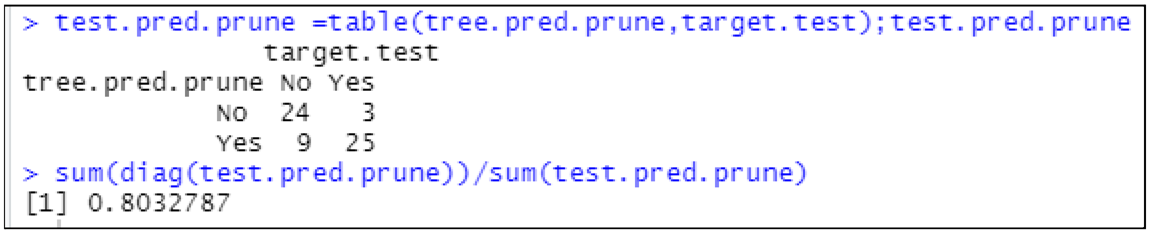
```
> cv.heart.factor
$size
[1] 19 14 13 10  6  4  2  1

$dev
[1]  56  57  57  55  52  58  60 107

$k
[1]      -Inf  0.000000  1.000000  1.333333  1.500000  4.500000  5.500000 51.000000

$method
[1] "misclass"

attr(,"class")
[1] "prune"         "tree.sequence"
```

The tree size as shown on the left for 8 nodes along with complexity k shown on the right indicates the minimum values to be the **best K = 6** of this model.



Applying this information we can prune the tree down to obtain the 8 nodes tree model as follows. Once again, “thal” appears to be the most important indicator since it is the first tree split followed by “cp” to repeat 2 splits. Overall we can infer that these 4 variables can be big influencers when it comes to the response variable heart disease. Using the ***predict()*** function once again we calculate the actual class prediction to see if the performance of the test set has improved and it has since it has increased to 80.32% therefore we are headed in the right direction.





```
> test.pred.prune =table(tree.pred.prune,target.test);test.pred.prune
               target.test
tree.pred.prune No Yes
            No  24   3
            Yes  9  25
> sum(diag(test.pred.prune))/sum(test.pred.prune)
[1] 0.8032787
```

Since this is a classification problem, we use a confusion matrix to evaluate the performance of our model. The values on the diagonal correspond to true positives and true negatives (correct predictions) whereas the others correspond to false positives and false negatives. In this case, from the results obtained above, we have the percentage of true negatives (correct prediction of class 0 as ‘No’) is 88. 89% and true positives (correctly prediction of class 1 as ‘Yes’) is 73.53% for the test set.

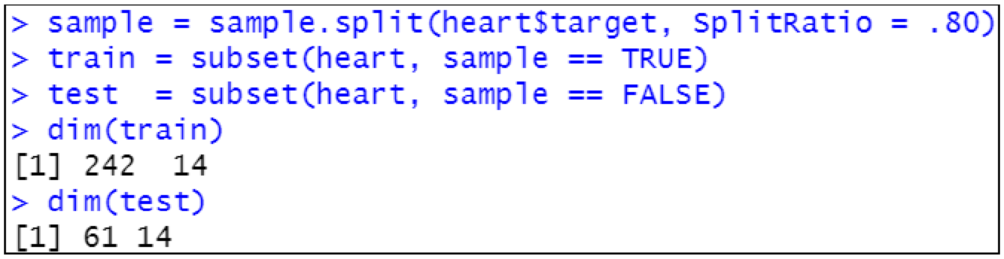
All in all, the accuracy of the improved Decision Tree model is 80.33% measured by the test set in our case, which outperforms 3.29% if compared to the original Decision Tree model (accuracy 77.04%). Therefore, we choose the improved Decision Tree model as our final model for this section.

Random Forest

Random forest is a flexible, easy to use machine learning algorithm that produces, even without hyper-parameter tuning, a great result most of the time. It is also one of the most used algorithms, because of its simplicity and diversity. Random forest is a supervised learning algorithm. The "forest" it builds is an ensemble of decision trees, usually trained with the “bagging” method. The general idea of the bagging method is that a combination of learning models increases the overall result.

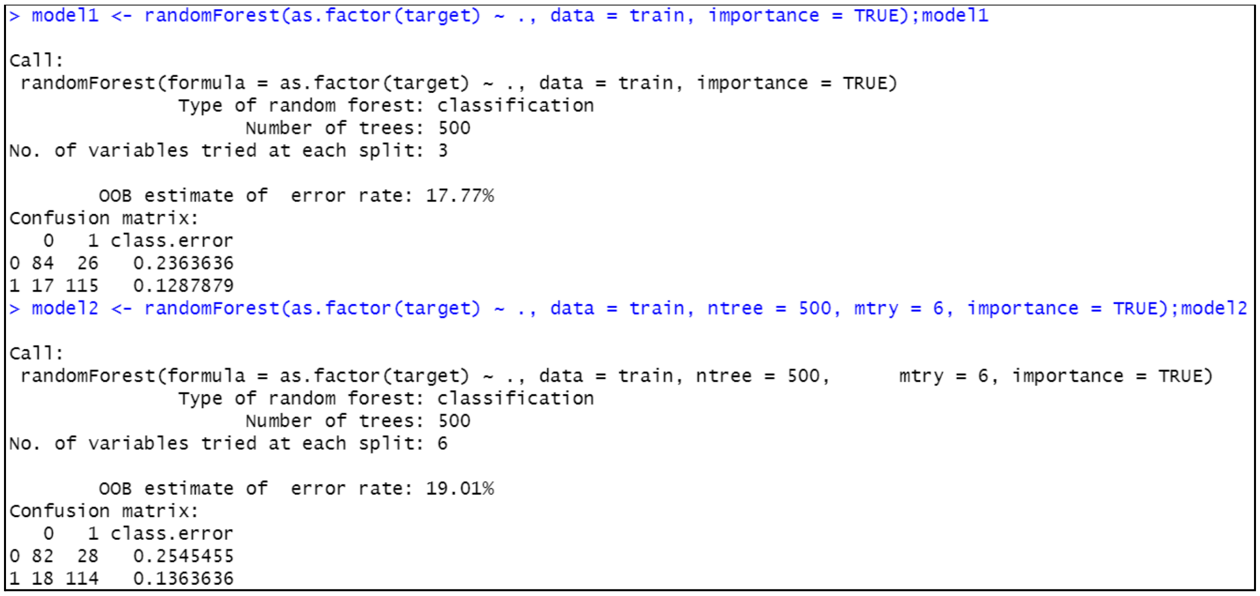
Random forest adds additional randomness to the model, while growing the trees. Instead of searching for the most important feature while splitting a node, it searches for the best feature among a random subset of features. This results in a wide diversity that generally results in a better model. Therefore, in Random Forest, only a random subset of the features is taken into consideration by the algorithm for splitting a node. You can even make trees more random by additionally using random thresholds for each feature rather than searching for the best possible thresholds (like a normal decision tree does). In this section we want to build a Random Forest model that can evaluate which of the 13 variables has the most influence on the response variable, heart disease, as well as find the best performance of this classification and observe whether Random Forest has a better performance than Decision Tree.

We split the data into a 80/20 train/test:



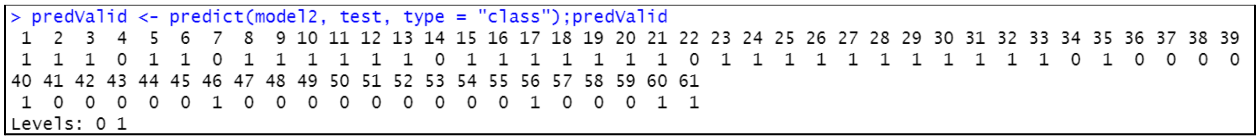
```
> sample = sample.split(heart$target, SplitRatio = .80)
> train = subset(heart, sample == TRUE)
> test  = subset(heart, sample == FALSE)
> dim(train)
[1] 242  14
> dim(test)
[1] 61 14
```

By default, the number of decision trees in the forest is 500 and the number of features used as potential candidates for each split is 3 and 6 respectively. The model will automatically attempt to classify each of the samples in the Out-Of-Bag dataset and display a confusion matrix with the results.



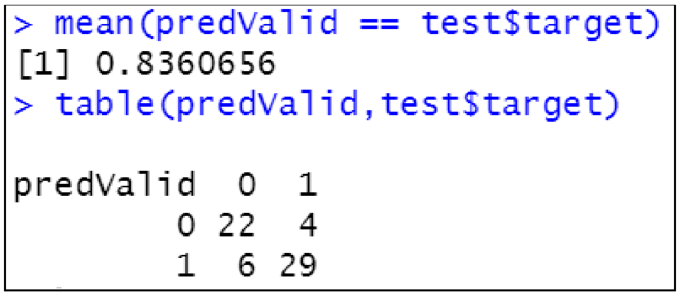
```
> model1 <- randomForest(as.factor(target) ~ ., data = train, importance = TRUE);model1

Call:
 randomForest(formula = as.factor(target) ~ ., data = train, importance = TRUE)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 3

        OOB estimate of  error rate: 17.77%
Confusion matrix:
   0   1 class.error
0 84  26   0.2363636
1 17 115   0.1287879
> model2 <- randomForest(as.factor(target) ~ ., data = train, ntree = 500, mtry = 6, importance = TRUE);model2

Call:
 randomForest(formula = as.factor(target) ~ ., data = train, ntree = 500,      mtry = 6, importance = TRUE)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 6

        OOB estimate of  error rate: 19.01%
Confusion matrix:
   0   1 class.error
0 82  28   0.2545455
1 18 114   0.1363636
```

We can see that when compared to a mtry of 6, the mtry of 3 had a lower OOB error rate, therefore a smaller split worked better for this model. Therefore we will continue using model2. For model2 we predict whether the people in our testing set have heart disease:



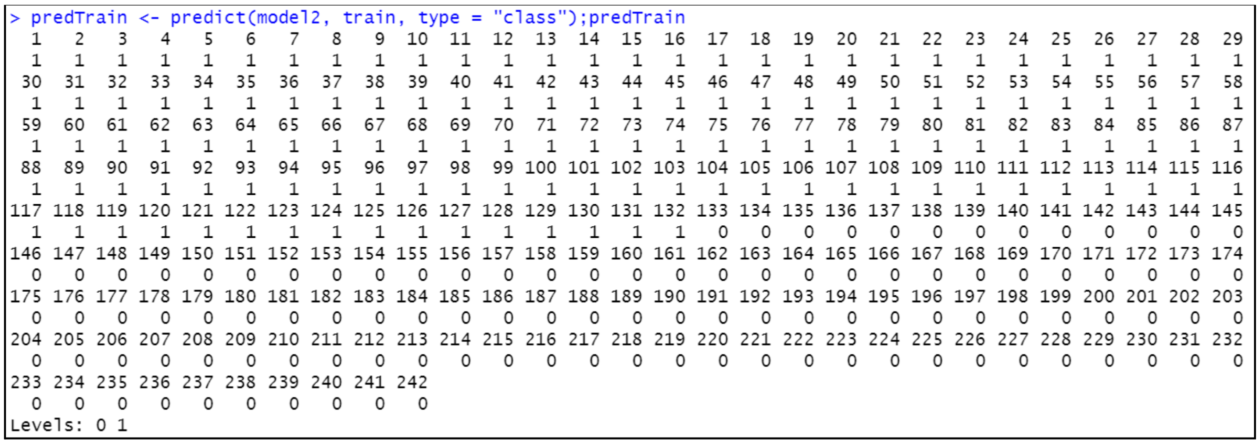
```
> predValid <- predict(model2, test, type = "class");predValid
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 
 1  1  1  0  1  1  0  1  1  1  1  1  1  0  1  1  1  1  1  1  1  0  1  1  1  1  1  1  1  1  1  1  1  0  1  0  0  0  0 
40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 
 1  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  1  0  0  0  1  1 
Levels: 0 1
```

Since this is a classification problem, we use a confusion matrix to evaluate the performance of our model. The values on the diagonal correspond to true positives and true negatives (correct predictions) whereas the others correspond to false positives and false negatives.

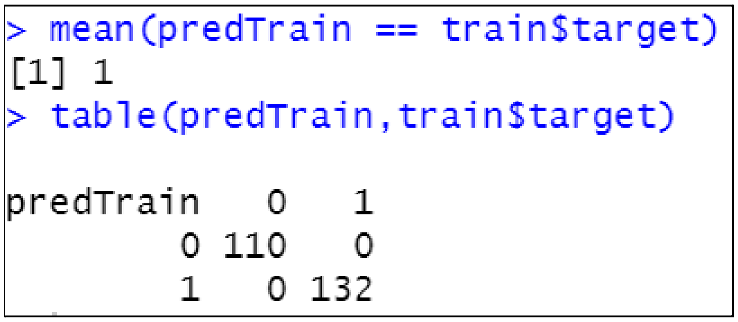


```
> mean(predValid == test$target)
[1] 0.8360656
> table(predValid,test$target)

predValid  0  1
        0 22  4
        1  6 29
```

For the case of the Validation data test, the accuracy is approximately 83.61%.



```
> predTrain <- predict(model2, train, type = "class");predTrain
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29 
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1 
 30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58 
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1 
 59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87 
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1 
 88  89  90  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1 
117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   0   0   0   0   0   0   0   0   0   0   0   0   0 
146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 
175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 
204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 
233 234 235 236 237 238 239 240 241 242 
  0   0   0   0   0   0   0   0   0   0 
Levels: 0 1
```

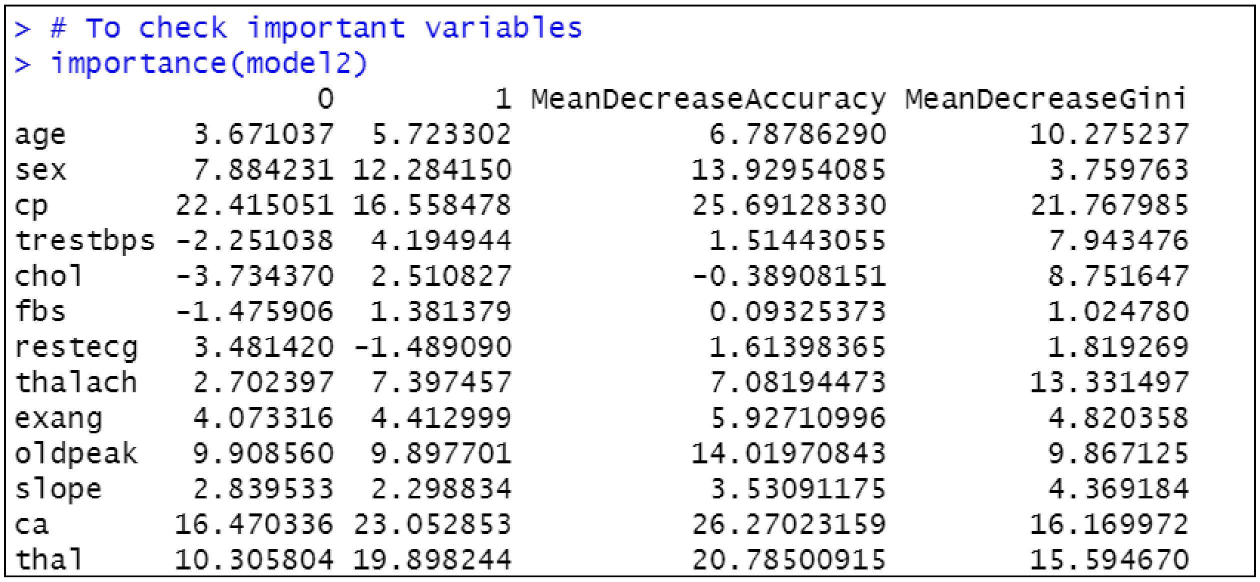


```
> mean(predTrain == train$target)
[1] 1
> table(predTrain,train$target)

predTrain   0   1
        0 110   0
        1   0 132
```

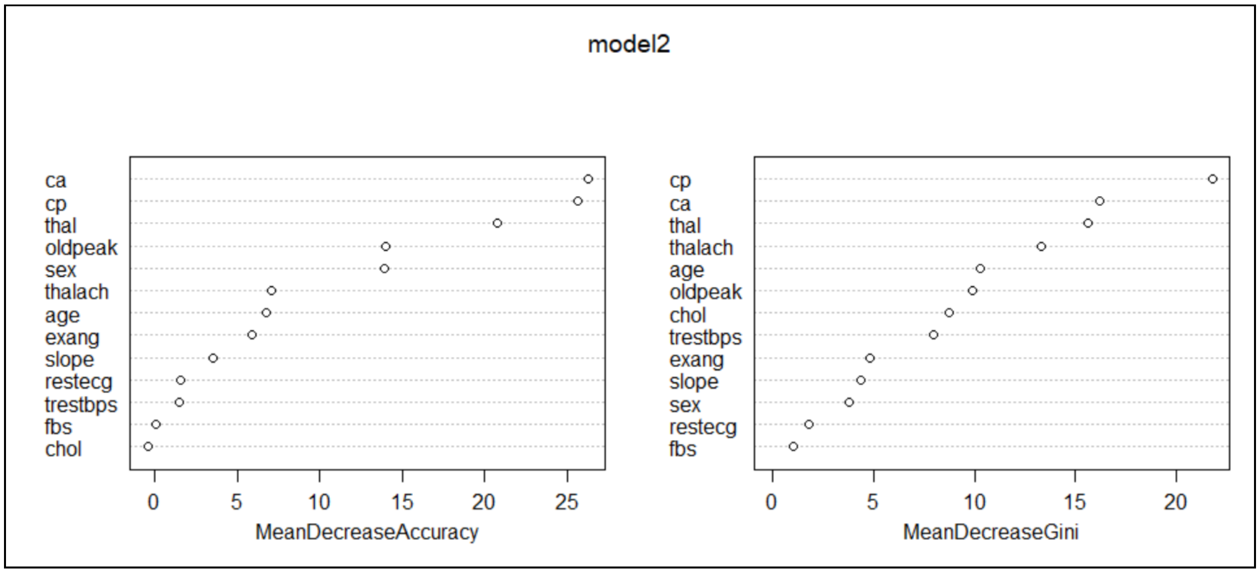
For the Train data test, the accuracy was about 100% , this can be due to overfitting which the model unfortunately can not avoid since the training performance reached 1.

After training a random forest, it is natural to ask which variables have the most predictive power. Variables with high importance are drivers of the outcome and their values have a significant impact on the outcome values. By contrast, variables with low importance might be omitted from a model, making it simpler and faster to fit and predict. Below shows the importance of 13 variables when predicting an outcome with two options. In this instance, the outcome is whether a person has a heart disease or doesn't have a heart disease.

There are two measures of importance given for each variable in the random forest. The first measure is based on how much the accuracy decreases when the variable is excluded. This is further broken down by outcome class. The second measure is based on the decrease of Gini impurity when a variable is chosen to split a node.



```
> # To check important variables
> importance(model2)
                  0         1 MeanDecreaseAccuracy MeanDecreaseGini
age        3.671037  5.723302           6.78786290        10.275237
sex        7.884231 12.284150          13.92954085         3.759763
cp        22.415051 16.558478          25.69128330        21.767985
trestbps  -2.251038  4.194944           1.51443055         7.943476
chol      -3.734370  2.510827          -0.38908151         8.751647
fbs       -1.475906  1.381379           0.09325373         1.024780
restecg    3.481420 -1.489090           1.61398365         1.819269
thalach    2.702397  7.397457           7.08194473        13.331497
exang      4.073316  4.412999           5.92710996         4.820358
oldpeak    9.908560  9.897701          14.01970843         9.867125
slope      2.839533  2.298834           3.53091175         4.369184
ca        16.470336 23.052853          26.27023159        16.169972
thal      10.305804 19.898244          20.78500915        15.594670
```



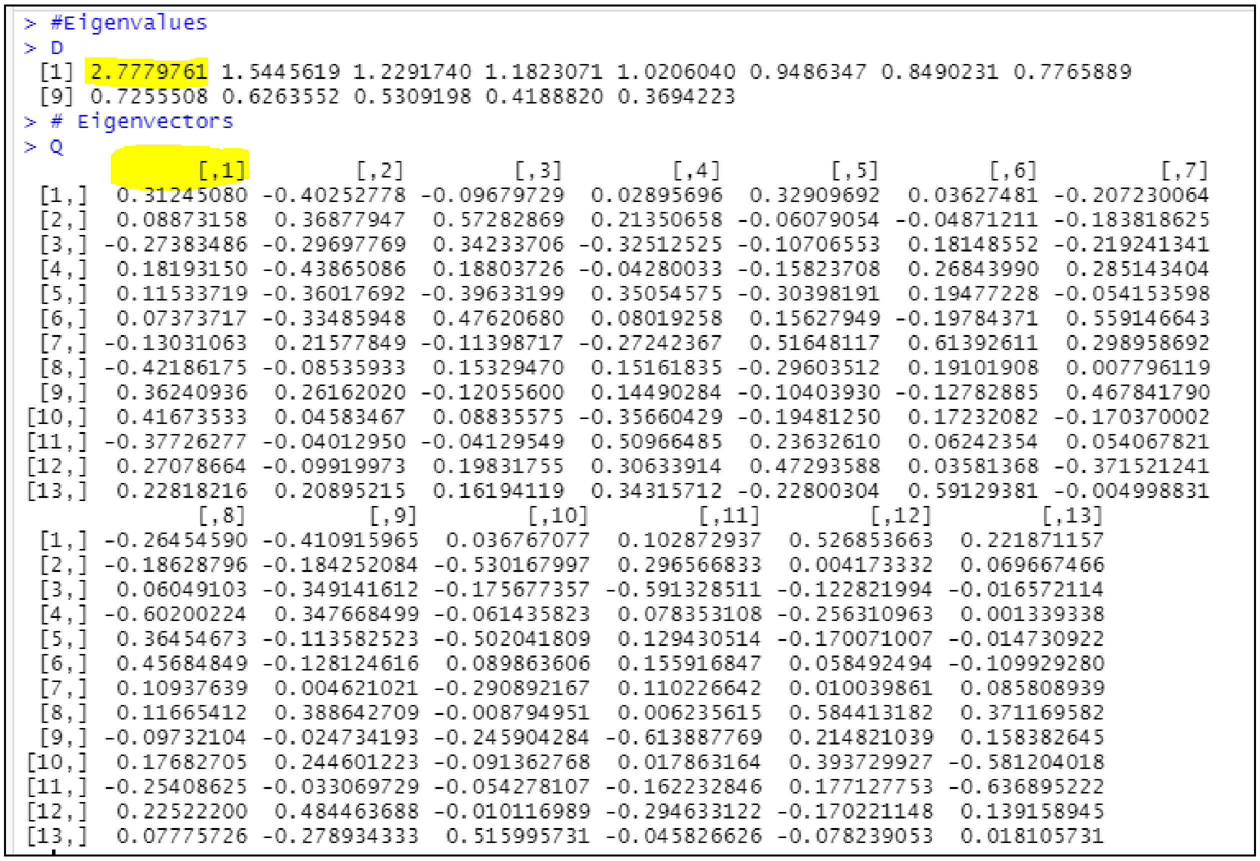
From the above right plot we can see that the cp, chest pain type has the higher decrease in Gini coefficient, followed by ca, and thal, indicating that these are by far the most important variables. This was also found to be the case in our decision tree section of the report after pruning.

Now, we have seen the implementation of Random Forest and understood the importance of the model. We compared this model with the decision tree and found that RF did better than the Decision Tree in test accuracy. Specifically, the accuracy of the Random Forest model is 83.61% measured by the test set in our case, which is better than 3.30 % if compared to the Decision Tree model (accuracy 80.33%). However, they both identified and evaluated similar results in which of the 13 variables has the most influence on the response variable, heart disease: chest pain (cp), number of major vessels (ca), and blood disorder (thal).

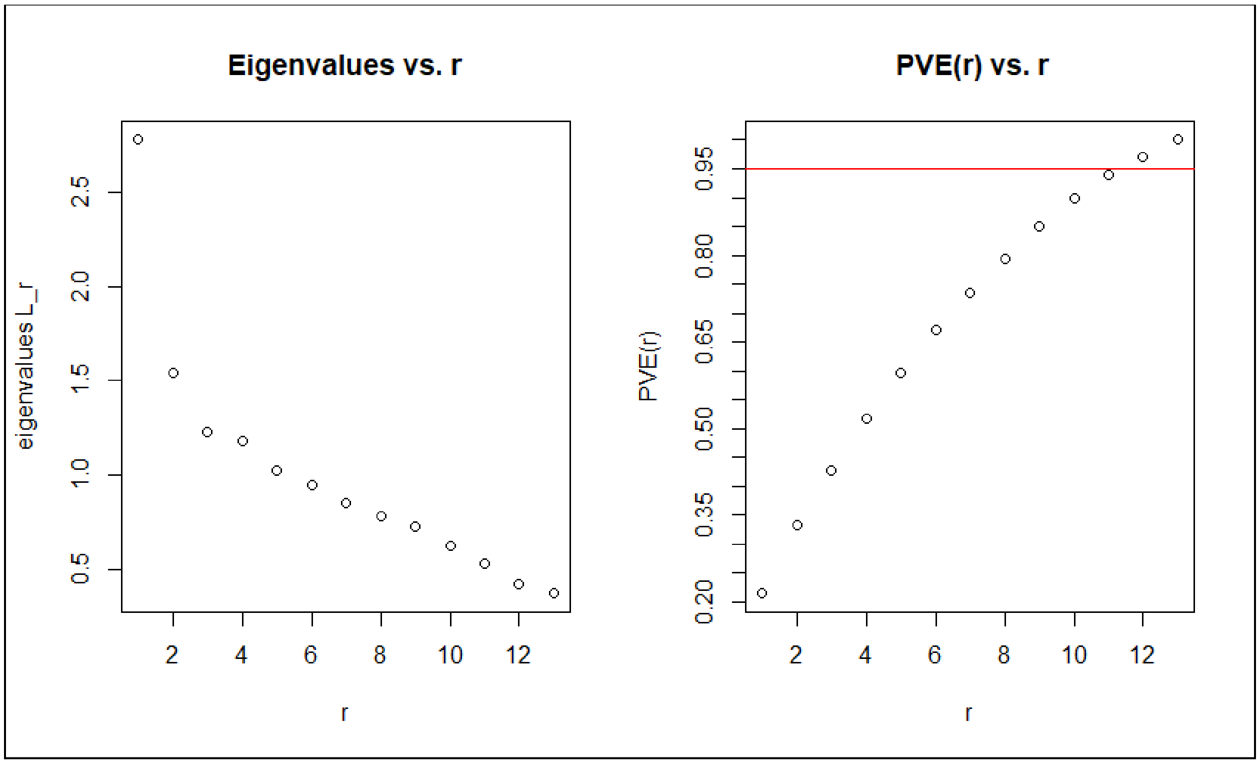
Principal Component Analysis - PCA

The main idea of principal component analysis (PCA) is to reduce the dimensionality of a data set consisting of many variables correlated with each other, either heavily or lightly, while retaining the variation present in the dataset, up to the maximum extent. The same is done by transforming the variables to a new set of variables, which are known as the principal components (or simply, the PCs) and are orthogonal, ordered such that the retention of variation present in the original variables decreases as we move down in the order. So, in this way, the 1st principal component retains maximum variation that was present in the original components. The principal components are the eigenvectors of a covariance matrix, and hence they are orthogonal. Importantly, the dataset on which PCA technique is to be used must be scaled. The results are also sensitive to the relative scaling.

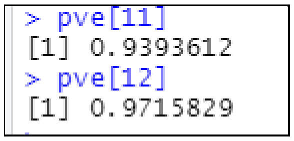
After we scale the data and define *R* as the correlation matrix, we get the eigenvalues and eigenvectors. Eigenvalues are displayed from highest to lowest, L1 > L2 > … L13, with a sum representing the overall variance. It is important to understand that each eigenvalue has a corresponding eigenvector respectfully displayed V1, V2,...,V13. Highlighted for easy reading, we can see the largest eigenvalue and our most significant is ***L1 = 2.77797*** along with its corresponding eigenvector in column 1 we define as ***V1.*** In the graphs below, on the left we see the eigenvalues plotted.



```
> #Eigenvalues
> D
 [1] 2.7779761 1.5445619 1.2291740 1.1823071 1.0206040 0.9486347 0.8490231 0.7765889
 [9] 0.7255508 0.6263552 0.5309198 0.4188820 0.3694223
> # Eigenvectors
> Q
             [,1]        [,2]        [,3]        [,4]        [,5]        [,6]         [,7]
 [1,]  0.31245080 -0.40252778 -0.09679729  0.02895696  0.32909692  0.03627481 -0.207230064
 [2,]  0.08873158  0.36877947  0.57282869  0.21350658 -0.06079054 -0.04871211 -0.183818625
 [3,] -0.27383486 -0.29697769  0.34233706 -0.32512525 -0.10706553  0.18148552 -0.219241341
 [4,]  0.18193150 -0.43865086  0.18803726 -0.04280033 -0.15823708  0.26843990  0.285143404
 [5,]  0.11533719 -0.36017692 -0.39633199  0.35054575 -0.30398191  0.19477228 -0.054153598
 [6,]  0.07373717 -0.33485948  0.47620680  0.08019258  0.15627949 -0.19784371  0.559146643
 [7,] -0.13031063  0.21577849 -0.11398717 -0.27242367  0.51648117  0.61392611  0.298958692
 [8,] -0.42186175 -0.08535933  0.15329470  0.15161835 -0.29603512  0.19101908  0.007796119
 [9,]  0.36240936  0.26162020 -0.12055600  0.14490284 -0.10403930 -0.12782885  0.467841790
[10,]  0.41673533  0.04583467  0.08835575 -0.35660429 -0.19481250  0.17232082 -0.170370002
[11,] -0.37726277 -0.04012950 -0.04129549  0.50966485  0.23632610  0.06242354  0.054067821
[12,]  0.27078664 -0.09919973  0.19831755  0.30633914  0.47293588  0.03581368 -0.371521241
[13,]  0.22818216  0.20895215  0.16194119  0.34315712 -0.22800304  0.59129381 -0.004998831
             [,8]         [,9]        [,10]        [,11]        [,12]        [,13]
 [1,] -0.26454590 -0.410915965  0.036767077  0.102872937  0.526853663  0.221871157
 [2,] -0.18628796 -0.184252084 -0.530167997  0.296566833  0.004173332  0.069667466
 [3,]  0.06049103 -0.349141612 -0.175677357 -0.591328511 -0.122821994 -0.016572114
 [4,] -0.60200224  0.347668499 -0.061435823  0.078353108 -0.256310963  0.001339338
 [5,]  0.36454673 -0.113582523 -0.502041809  0.129430514 -0.170071007 -0.014730922
 [6,]  0.45684849 -0.128124616  0.089863606  0.155916847  0.058492494 -0.109929280
 [7,]  0.10937639  0.004621021 -0.290892167  0.110226642  0.010039861  0.085808939
 [8,]  0.11665412  0.388642709 -0.008794951  0.006235615  0.584413182  0.371169582
 [9,] -0.09732104 -0.024734193 -0.245904284 -0.613887769  0.214821039  0.158382645
[10,]  0.17682705  0.244601223 -0.091362768  0.017863164  0.393729927 -0.581204018
[11,] -0.25408625 -0.033069729 -0.054278107 -0.162232846  0.177127753 -0.636895222
[12,]  0.22522200  0.484463688 -0.010116989 -0.294633122 -0.170221148  0.139158945
[13,]  0.07775726 -0.278934333  0.515995731 -0.045826626 -0.078239053  0.018105731
```



The graph on the right we take a look at the Percentage of Variance Explained which is the cumulative distribution of our variance explained. To select the optimal number of principal components we could eyeball the PVE graph or we can use trial and error in R at 95% we narrow down our result to find our result at **r = 11.** At this 95% variance, we can be attributed to just 11 out of 13 features that are most important.



```
> pve[11]
[1] 0.9393612
> pve[12]
[1] 0.9715829
```

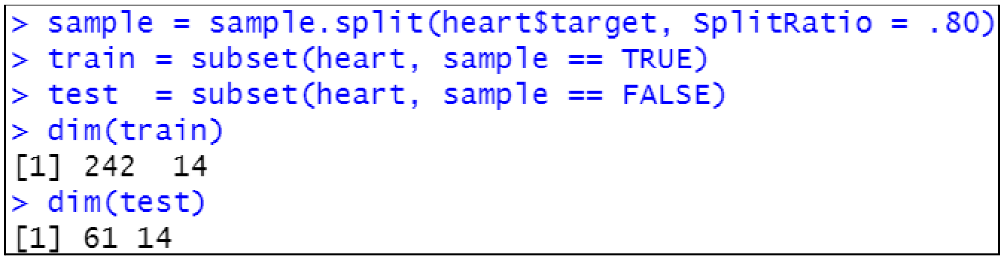
All in all, from the result we obtain, we observe that most of our features are important (11 out of 13) which means we should not reduce the dimensions in our case, in other words, PCA is not really effective with our dataset. From researches have been conducted, they show that PCA can be effectively used when the dimensions of the input features are high (e.g. a lot of variables, hundreds of features). In our case, we only have 13 features which is not a high number, therefore, what we can conclude here is that PCA may not be much effective if applying to datasets having small number of features, such as our dataset, however, it can be very effective used when the dimensions of the input features are high in other cases.

K-Nearest Neighbor (KNN)

K-Nearest Neighbors (KNN) is one of the simplest algorithms used in Machine Learning for regression and classification problems. KNN algorithms use data and classify new data points based on similarity measures (e.g. distance function). Classification is done by a majority vote to its neighbors. The data is assigned to the class which has the nearest neighbors. As you increase the number of nearest neighbors, the value of k, accuracy might increase. It uses 'feature similarity' to predict the values of new data points which further means that the new data point will be assigned a value based on how closely it matches the points in the training set.

In this section, our goal is to build a K-Nearest Neighbors (KNN) model in order to predict the response variable, heart disease, as well as find the best performance of this classification and evaluate whether K-Nearest Neighbors (KNN) has a better performance than Random Forest and Decision Tree.

Similar to what we have done in Random Forest, first we attempt to split the normalized and standardized data into a 80/20 train/test to have the same measurement:

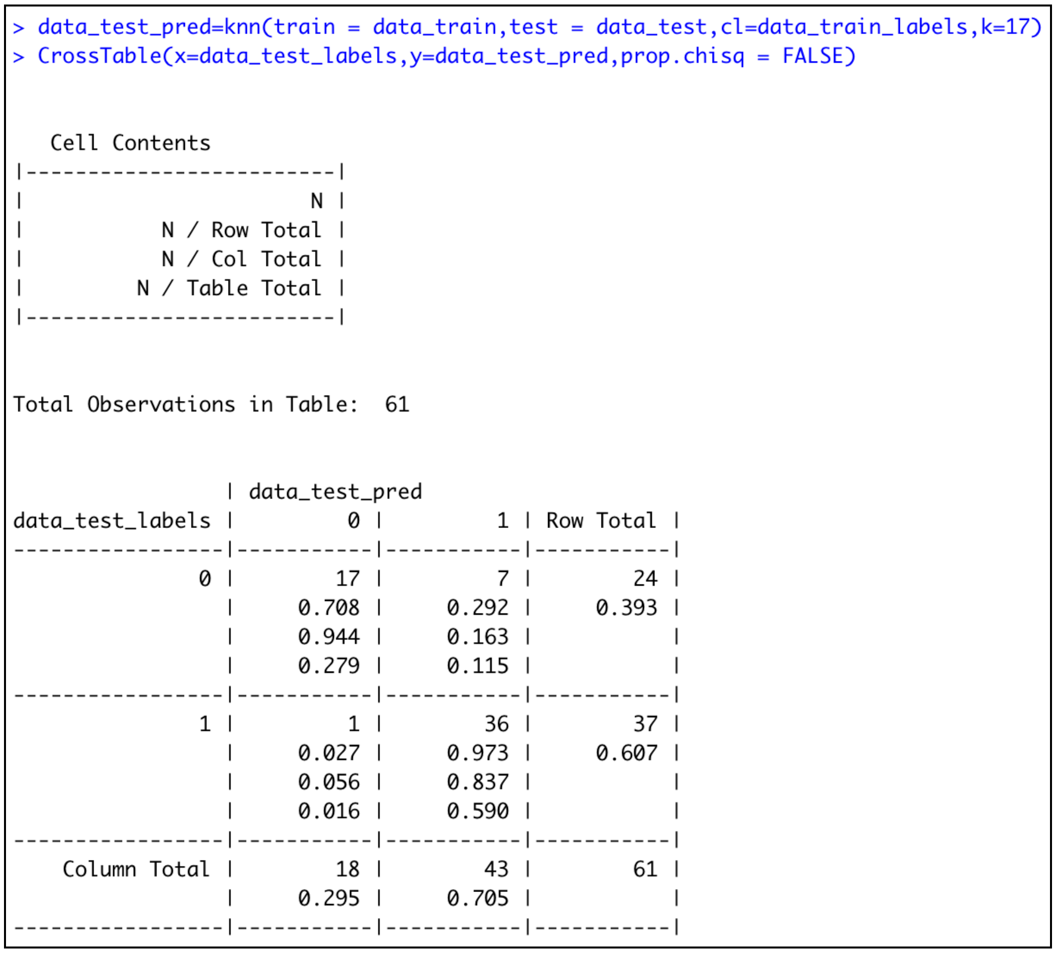


```
> sample = sample.split(heart$target, SplitRatio = .80)
> train = subset(heart, sample == TRUE)
> test  = subset(heart, sample == FALSE)
> dim(train)
[1] 242  14
> dim(test)
[1] 61 14
```

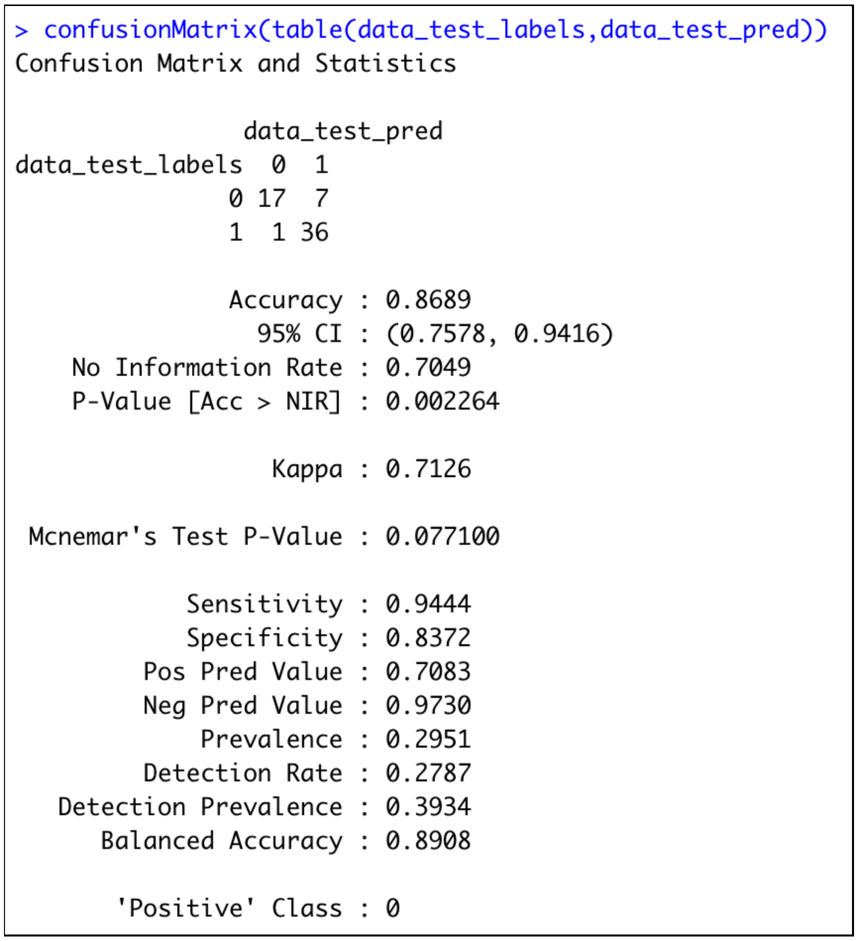
Before applying the KNN algorithm, it's needed to find out the optimal value of k as the number of nearest neighbors based on feature similarity in order to get better performance and accuracy. However, finding the optimal value of k is one of the hardest steps in the KNN algorithm, because there is no straightforward and structured method at all to calculate it.

A small value of k has a lower value of bias and a higher variance which is associated generally with a more complex model, and a large value of k has a higher value of bias and a lower variance which is associated with a simpler model. Also, choosing a large value of k is not effective and efficient, because it computationally costs a lot. Although a larger value of k will get higher accuracy, the model will also go under-fitted which means the error rate will again increase. Therefore, the optimal k should be chosen as a small value, for the reason that noise will make a higher influence on the result. Based on many researches have been conducted, researchers suggest that the optimal k value usually found is the square root of N, where N is the total number of samples.

In this case, we choose the optimal k value as the square root of the total number of samples, therefore, k is the square root of 301 samples which is **k = 17**. We apply to the KNN model and obtain the following results of cross table and confusion matrix:



```
> data_test_pred=knn(train = data_train,test = data_test,cl=data_train_labels,k=17)
> CrossTable(x=data_test_labels,y=data_test_pred,prop.chisq = FALSE)


   Cell Contents
|-------------------------|
|                       N |
|           N / Row Total |
|           N / Col Total |
|         N / Table Total |
|-------------------------|


Total Observations in Table:  61


                 | data_test_pred
data_test_labels |         0 |         1 | Row Total |
-----------------|-----------|-----------|-----------|
               0 |        17 |         7 |        24 |
                 |     0.708 |     0.292 |     0.393 |
                 |     0.944 |     0.163 |           |
                 |     0.279 |     0.115 |           |
-----------------|-----------|-----------|-----------|
               1 |         1 |        36 |        37 |
                 |     0.027 |     0.973 |     0.607 |
                 |     0.056 |     0.837 |           |
                 |     0.016 |     0.590 |           |
-----------------|-----------|-----------|-----------|
    Column Total |        18 |        43 |        61 |
                 |     0.295 |     0.705 |           |
-----------------|-----------|-----------|-----------|
```



```
> confusionMatrix(table(data_test_labels,data_test_pred))
Confusion Matrix and Statistics

                data_test_pred
data_test_labels  0  1
               0 17  7
               1  1 36

               Accuracy : 0.8689
                 95% CI : (0.7578, 0.9416)
    No Information Rate : 0.7049
    P-Value [Acc > NIR] : 0.002264

                  Kappa : 0.7126

 Mcnemar's Test P-Value : 0.077100

            Sensitivity : 0.9444
            Specificity : 0.8372
         Pos Pred Value : 0.7083
         Neg Pred Value : 0.9730
             Prevalence : 0.2951
         Detection Rate : 0.2787
   Detection Prevalence : 0.3934
      Balanced Accuracy : 0.8908

       'Positive' Class : 0
```

Since this is a classification problem, we use a confusion matrix to evaluate the performance of our model. The values on the diagonal correspond to true positives and true negatives (correct predictions) whereas the others correspond to false positives and false negatives. In this case, from the results obtained above, we have the percentage of true negatives (correct prediction of class 0) is 70. 83% and true positives (correctly prediction of class 1) is 97.30% for the test set.

All in all, the accuracy of the K-Nearest Neighbors (KNN) model is 86.89% measured by the test set in our case, which is better than 3.28% if compared to the Random Forest model (accuracy 83.61%), and outperforms 6.56% if compared to the Decision Tree model (accuracy 80.33%).

Conclusion

Heart disease is one of the major causes in real world society of people's today. In this paper we have developed an automated diagnosis system for prediction of heart disease using different analysis models. In this report we observed that our test accuracy performance increased as we moved up in our machine learning models:

Decision Tree < Random Forest < KNN

Specifically, the accuracy of the K-Nearest Neighbors (KNN) model is 86.89% measured by the test set in our case, which is better than 3.28% if compared to the Random Forest model (accuracy 83.61%), and outperforms 6.56% if compared to the Decision Tree model (accuracy 80.33%). Last but not least, in future study, we would like to further explore more Machine Learning models applying to our dataset such as Logistic Regression, XGBoost and so on, as well as explore multiple methods to apply to each model such as feature selection, adjusting parameters and so on to ultimately lower the error metric, increase the accuracy metric, or prevent overfitting to find better model(s) or potentially combine some of the existing models together to form the best ones.

In addition, we also observed that the top 3 most influencing variables are: Chest Pain (cp), Number of Major Vessels (ca), and Blood Disorder type (thal). It would be interesting to explore other machine learning techniques for this data set, as well as other coding programs other than R (such as python) to see how different or similar our results compare.

References

1. https://my.clevelandclinic.org/health/articles/11920-cholesterol-numbers-what-do-they-mean
2. https://www.heart.org/en/health-topics/high-blood-pressure/understanding-blood-pressure-readings
3. https://bolt.mph.ufl.edu/6050-6052/unit-1/one-quantitative-variable-introduction/describing-distributions/