

ADA 442: Statistical Learning

Heart Disease Prediction Using Different Models

Tolgay Atınç Uzun 15820640848 Gökmen Çağlar 12590403284 Ufuk Özkul 58702042132

06 Haziran, 2022

Contents

1 Abstract & Introduction	2
2 Used Libraries	2
3 Methodology	2
4 Data	2
4.1 Numerical Data analysis	3
4.2 Histogram Analysis	4
5 Preprocessing Step	7
6 Detailed Data Analysis	11
6.1 Categorical Data Analysis	13
7 Data Partioning	16
8 Logisitic Regression	16
8.1 Result Analysis	19
9 Decision Tree	21
9.1 Result Analysis	22
9.2 Pruning	23
9.3 Result Analysis	25
9.4 Rpart Library Desicion Tree	26
10 Comparison	28
11 References	30

1 Abstract & Introduction

The aim of this project is to predict heart disease from the medical scanning and blood sample data. It has been stated that the number of patients with heart failure worldwide was 64.3 million in 2017. This alone accounts great expenses on the global society due to specialized methods and equipments that are employed for diagnosis of heart disease. Predicting the condition with machine learning techniques would help reducing the costs as well as making the process simpler for the healthcare workers.

2 Used Libraries

```
set.seed(84485)
library(caret)
library(ISLR2)
library(corrplot)
library(ggplot2)
library(tree)
library(mice)
library(boot)
library(tidyverse)
library(caret)
library(rpart)
```

3 Methodology

In this report, the Logistic Regression and Decision Tree models are used. Specifically Logistic Regression has been trained by 2 different approach. First one is trained using all predictors and the second one trained using specifically selected predictors. In Decision trees part, unpruned, pruned and 10-fold rpart decision tree was trained.

4 Data

```
heartData <- read.csv(file = 'heart.csv')
head(heartData)
```

```
##   Age Sex ChestPainType RestingBP Cholesterol FastingBS RestingECG MaxHR
## 1  40  M           ATA        140         289          0    Normal   172
## 2  49  F           NAP        160         180          0    Normal   156
## 3  37  M           ATA        130         283          0         ST     98
## 4  48  F           ASY        138         214          0    Normal   108
## 5  54  M           NAP        150         195          0    Normal   122
## 6  39  M           NAP        120         339          0    Normal   170
##   ExerciseAngina Oldpeak ST_Slope HeartDisease
## 1              N     0.0      Up           0
## 2              N     1.0     Flat           1
## 3              N     0.0      Up           0
## 4              Y     1.5     Flat           1
## 5              N     0.0      Up           0
## 6              N     0.0      Up           0
```

4.1 Numerical Data analysis

```
# Get the amount of columns and number of rows
dim(heartData)
```

```
## [1] 918 12
```

There are 12 predictor columns in the dataset and 918 rows.

```
summary(heartData)
```

```
##      Age      Sex      ChestPainType      RestingBP
##  Min.   :28.00  Length:918      Length:918      Min.    : 0.0
## 1st Qu.:47.00  Class :character  Class :character 1st Qu.:120.0
## Median :54.00  Mode  :character  Mode  :character Median :130.0
## Mean   :53.51                                     Mean   :132.4
## 3rd Qu.:60.00                                     3rd Qu.:140.0
## Max.    :77.00                                     Max.    :200.0
## Cholesterol      FastingBS      RestingECG      MaxHR
##  Min.   : 0.0  Min.   :0.0000  Length:918      Min.    : 60.0
## 1st Qu.:173.2 1st Qu.:0.0000  Class :character 1st Qu.:120.0
## Median :223.0 Median :0.0000  Mode  :character Median :138.0
## Mean   :198.8 Mean   :0.2331                                     Mean   :136.8
## 3rd Qu.:267.0 3rd Qu.:0.0000                                     3rd Qu.:156.0
## Max.    :603.0 Max.    :1.0000                                     Max.    :202.0
## ExerciseAngina      Oldpeak      ST_Slope      HeartDisease
##  Length:918      Min.   :-2.6000  Length:918      Min.    :0.0000
##  Class :character 1st Qu.: 0.0000  Class :character 1st Qu.:0.0000
##  Mode  :character Median   : 0.6000  Mode  :character Median :1.0000
##                                     Mean    : 0.8874  Mean   :0.5534
##                                     3rd Qu.: 1.5000  3rd Qu.:1.0000
##                                     Max.     : 6.2000  Max.    :1.0000
```

The type of the predictors should be examined for getting a better insight about the data. There are numerous columns which have categorical value type as well as numeric value type.

The columns which have categorical values and contain characters are: Sex, ChestPainType, RestingECG, ExerciseAngina, ST_Slope. In the dataset, HeartDisease column is originally given as numeric but it denotes whether heart disease exists or not. It might be included as a categorical value. A similar case is also observed in the FastingBS column, too.

The columns that are numeric: Age, RestingBP, Cholesterol, MaxHR and Oldpeak. Since these columns are composed of numeric values, getting statistical descriptions are easier.

When the mean and the median are the same, the predictor is likely to be evenly distributed for each value. This is evident in predictors such as Age and MaxHR. However, if the distribution of data is left skewed, the mean is less than the median and if it is right skewed, the mean is greater than median. For instance, the column "Oldpeak" shows right skewness.

The first quartile gives data samples which are found under lowest 25% when they are arranged in increasing order. The third quartile, is the value under which three quartile out of four are found when arranged in increasing order. This helps visualising the spread of the data. In cholesterol column, some samples deviate from mean which reside on the higher standard deviation area. Also, comparing maximum value with third quartile can be valuable.

Minimum and maximum values for the predictors seem ordinary except resting blood pressure(RestingBP) and cholesterol levels. Having the value “0” for both of them constitutes to extraordinary situations. Assuming this dataset was compiled using only alive people, the “0” value for resting blood pressure should definitely be removed. The cholesterol levels are generally not 0 so they should be preprocessed, too.

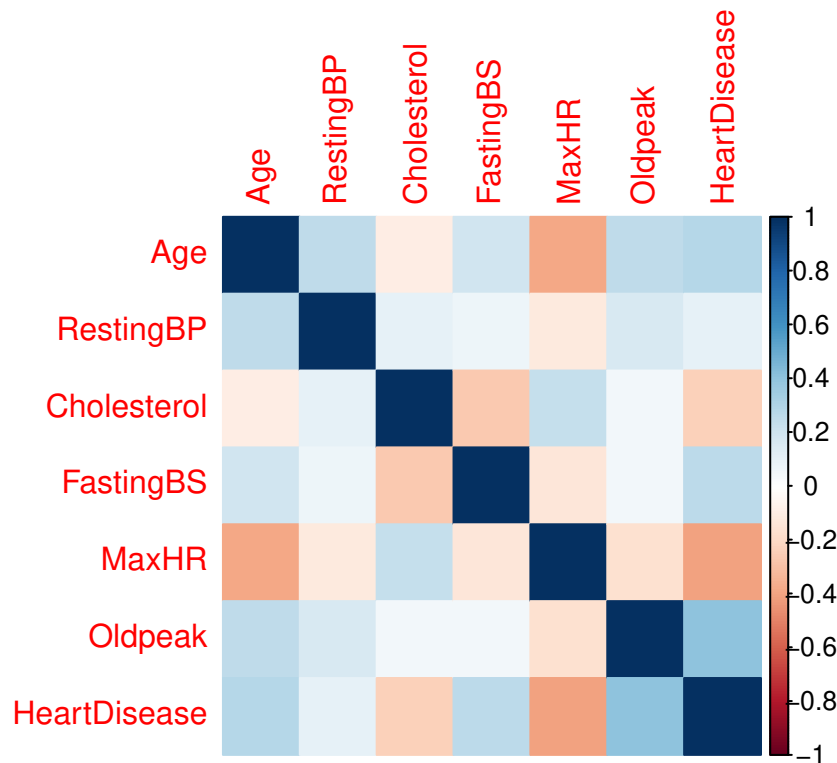
4.1.1 Correlation

To determine the redundant predictors, if exist, the correlation matrix should be analyzed. For this purpose the correlation values for different numeric columns are needed. To obtain those values only numeric data columns will be used.

```
# Determine numeric columns
num_cols <- unlist(lapply(heartData, is.numeric))

# Select only numeric columns
data_num <- heartData[, num_cols]

#correlation matrix creation
correlation_matrix <- cor(data_num)
corrplot(correlation_matrix, method = "color")
```



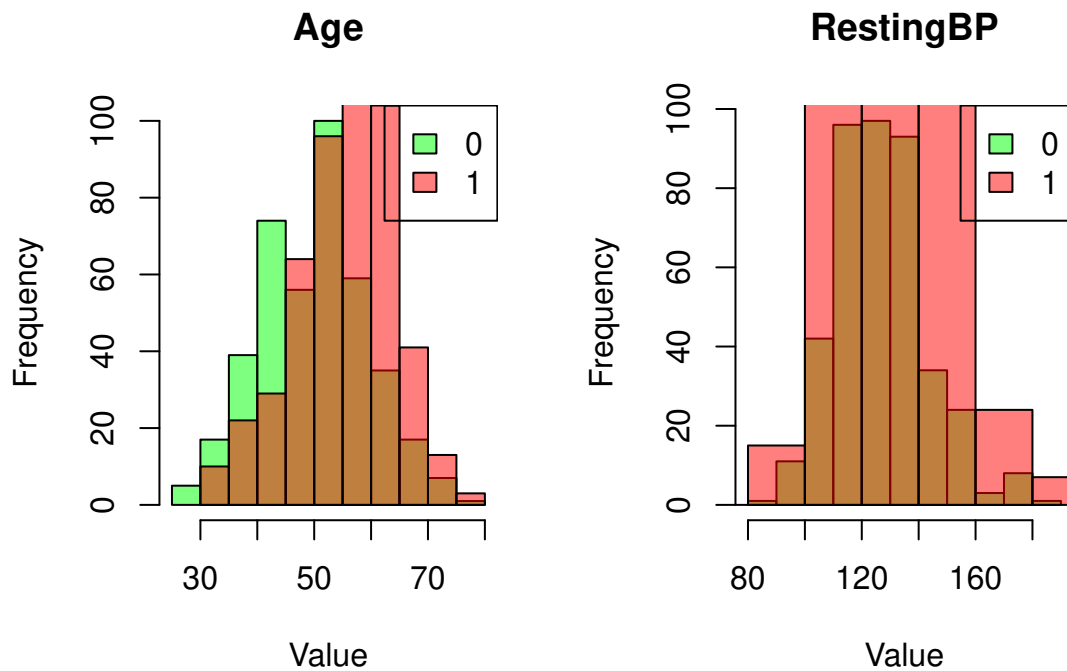
From the above correlation matrix it can be conclude that, non of the numeric columns have strong correlation with other columns. This means the Heart Disease dataset do not have redundant predictors.

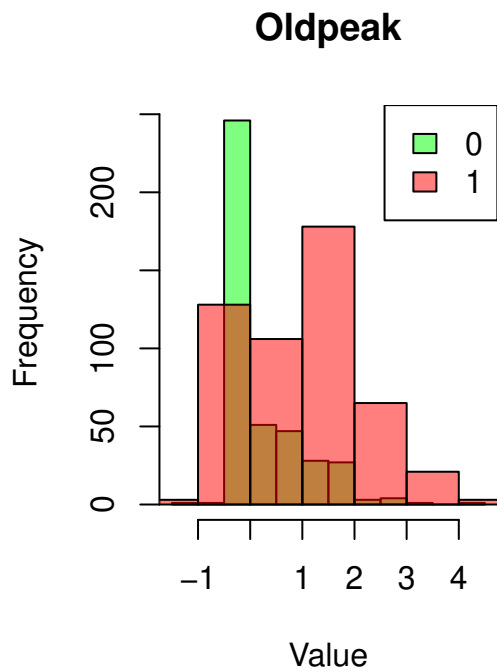
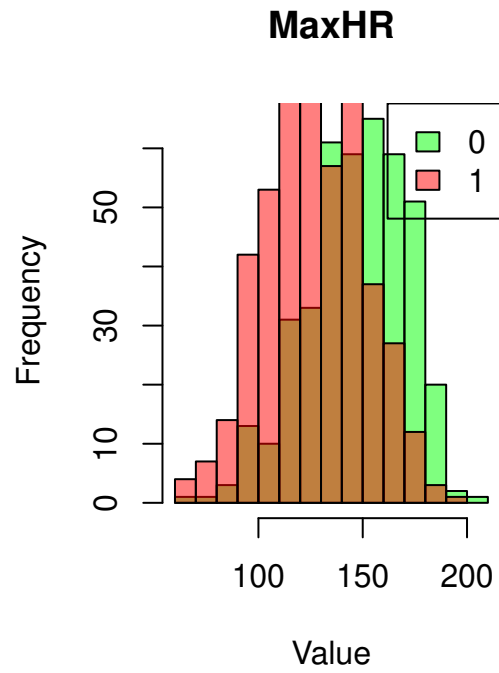
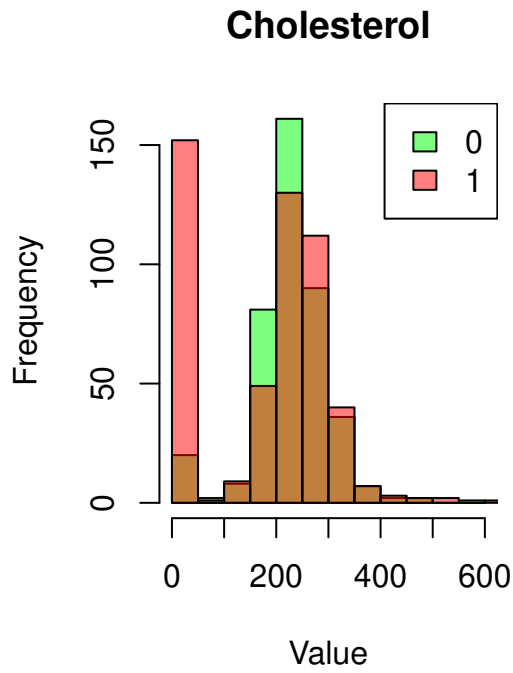
4.2 Histogram Analysis

In the histograms, the green columns symbolize the healthy patients, the red columns symbolizes the patients who have heart disease. Lastly the brown columns symbolizes the intersection of both of healthy and diseased

patients, it contains both green and red columns.

```
histogramDrawer = function(column, name){  
  
  healthyArr = c()  
  diseased = c()  
  for(a in 1:length(heartData$HeartDisease)){  
  
    result = heartData$HeartDisease[a]  
  
    if(result == 0){  
      healthyArr = append(healthyArr, column[a])  
    }  
    else{  
      diseased = append(diseased, column[a])  
    }  
  }  
  hist(healthyArr, col = rgb(0,1,0, 0.5), main = name, xlab = "Value")  
  hist(diseased, add = TRUE, col = rgb(1,0,0, 0.5))  
  legend('topright', c('0', '1'), fill=c(rgb(0,1,0, 0.5), rgb(1,0,0, 0.5)))  
}
```





4.2.1 Age Analysis

When the dataset's first column, Age, examined, it is noticed that when the age increases, the number of patients who have heart disease is rising. That can be obtained from the above graph. Before the age 45, the

count of healthy patients is larger than those who are likely to have a heart disease. If the age of patients are between 45 and 55, the number of those with heart disease are close to the number of healthy patients. After the age 55, the most of the patients are more likely to have the heart disease. This column is ideal for model training, since the heart disease distribution on the age factor is acceptable.

4.2.2 Resting BP Analysis

The blood pressure value between 80-120 mm Hg is assumed as normal blood pressure. Blood pressure between 120-130 mm Hg considered as elevated blood pressure, and lastly blood pressure more than 130 mm Hg is considered as high blood pressure. When the resting blood pressure histogram is considered, the patients whose resting blood pressure which is found in either low or high ends are more likely to have heart disease such as in 100-110 mm Hg and 140-180 mm Hg range. In other words, patients who have high blood pressure are more likely to have heart disease.

4.2.3 Cholesterol Analysis

Values between 50 to 600 mm/dl are mixed according to heart disease. It is hard to recognize which value range has clear color, since it contains both of no heart disease and heart disease. But it is obvious that patient's cholesterol values in 0 to 50 mm/dl are have heart disease. Because of that reason this column is suitable for model training.

4.2.4 MaxHR Analysis

From the above histogram it can be obtained that if the maximum heart rate value is more than 150, most of the patients have no heart disease. Same way if the maximum heart rate value is lower than 130, most of the patients have heart disease. This column is good choice for model training.

4.2.5 Oldpeak Analysis

From the last histogram, patient whose old peak values except -0.5 to 0 range are more likely to have a heart disease. This column is suitable for the model training since the old peak data distribution shows that heart diseased patients.

As a last word, all of the above histogram's data distribution is similar to the bell. Yet, there are significant exceptions so the preprocess operation should be done on the data because of the extremities such as RestingBP and Cholesterol columns.

5 Preprocessing Step

As discussed above, the data has 12 columns which are all related for prediction quality of the heart disease. It does not have database ID key column that are generally used for patient identification. Therefore, none of the columns should be dropped.

There are various methods for dealing with NA values. Simply, rows containing NA values can be dropped. However, since a crucial data regarding human health is concerned, filling them using imputation would be more suitable for as the specific rows would still contain useful information aside from the NA feature.

```
sum(is.na(heartData$Cholesterol))
```

```
## [1] 0
```

In the R chunk above, the total values for NA values are seen to be 0. At first glance, null values seem not to exist in the dataset. Therefore, one might assume not to employ necessary techniques for handling the missing data. In spite of that, the 0 values in Cholesterol can be interpreted as NA values since the history of the patients are not known. Normally, the distribution of cholesterol follows gaussian normal distribution but in this data the most of the samples have 0 value, which is not quite possible from the general human population perspective.

```
sum(heartData$Cholesterol == 0)
```

```
## [1] 172
```

In Cholesterol column, the count of 0 values are 172. It has considerable amount of 0s, so dropping them creates critical penalty for prediction operation. To prevent this penalty, the regression imputation process will be applied to the Cholesterol column. By this way, all of the zero values which correspond to NA, are going to be replaced with the result of the regression model. The model has been fit using the present values in other columns and the cholesterol value has been imputed.

However, the “complete” method obtained from mice library only works on NA values. Therefore, all of the 0 values in cholesterol should be converted to NA in order to make use of complete function.

```
heartData$Cholesterol[heartData$Cholesterol == 0] <- NA
imp <- mice(heartData, method = "norm.predict", m = 1, maxit=1)
```

```
##
## iter imp variable
## 1 1 Cholesterol
```

```
# Store data
heartDisease <- complete(imp)
heartData$Cholesterol <- as.integer(heartDisease$Cholesterol)
```

After the conversion process, the cholesterol values are casted to integer since the original cholesterol column data type is integer. This preserves the integrity of the data as the discrepancy between the float and integer has been mitigated.

```
sum(heartData$Cholesterol == 0)
```

```
## [1] 0
```

```
sum(is.na(heartData$Cholesterol))
```

```
## [1] 0
```

All the 0 values in Cholesterol column are replaced according to imputation calculation as seen above.

The outliers in the data are important in a sense that they carry information on the prediction quality. Although this is true, the impossible cases that are discussed in data familiarization part must be eliminated since the statistical descriptions validate this claim.


```
lower_bound <- quantile(heartData$RestingBP, 0.001)
lower_bound
```

```
## 0.1%
## 73.36
```

```
upper_bound <- quantile(heartData$RestingBP, 0.999)
upper_bound
```

```
## 99.9%
## 200
```

```
removedData <- subset(heartData, heartData$RestingBP <= lower_bound | heartData$RestingBP >= upper_bound)
heartData <- subset(heartData, heartData$RestingBP > lower_bound & heartData$RestingBP < upper_bound)
```

```
dim(heartData)
```

```
## [1] 913 12
```

```
removedData
```

```
##      Age Sex ChestPainType RestingBP Cholesterol FastingBS RestingECG MaxHR
## 242  54  M          ASY         200         198         0      Normal   142
## 366  64  F          ASY         200         268         0      Normal   140
## 400  61  M          NAP         200         266         1          ST    70
## 450  55  M          NAP         0         217         0      Normal   155
## 733  56  F          ASY         200         288         1         LVH   133
##      ExerciseAngina Oldpeak ST_Slope HeartDisease
## 242                Y    2.0     Flat            1
## 366                Y    1.0     Flat            1
## 400                N    0.0     Flat            1
## 450                N    1.5     Flat            1
## 733                Y    4.0     Down            1
```

After the outlier elimination using predetermined quantiles, the minimum and maximum values become:

```
min(heartData$RestingBP)
```

```
## [1] 80
```

```
max(heartData$RestingBP)
```

```
## [1] 192
```

RestingBP columns's preprocess is completed.

Some range of the features can be different from each other and this might cause loss of prediction quality. Large values can affect the computed weight in the model, conflicting with smaller values' effect. Therefore, min max scaling is used.

```

minMax <- function(x) {
  (x - min(x)) / (max(x) - min(x))
}

arr = c(1,4,5,8,10)

#normalize data using custom function
for(x in arr) {
  heartData[,x] = minMax(heartData[,x])
}

```

Character variables in our dataset causes some problems for tree and cv.tree methods. It directly converts these character to NA values. To avoid this we applied factor method on them. This way we get better results for our methods.

```
summary(heartData)
```

```

##      Age          Sex      ChestPainType      RestingBP
##  Min.   :0.0000   Length:913   Length:913   Min.   :0.0000
##  1st Qu.:0.3878   Class  :character   Class  :character   1st Qu.:0.3571
##  Median :0.5306   Mode   :character   Mode   :character   Median :0.4464
##  Mean   :0.5201                                     Mean   :0.4665
##  3rd Qu.:0.6531                                     3rd Qu.:0.5357
##  Max.   :1.0000                                     Max.   :1.0000
##  Cholesterol      FastingBS      RestingECG      MaxHR
##  Min.   :0.0000   Min.   :0.0000   Length:913   Min.   :0.0000
##  1st Qu.:0.2490   1st Qu.:0.0000   Class  :character   1st Qu.:0.4225
##  Median :0.3069   Median :0.0000   Mode   :character   Median :0.5493
##  Mean   :0.3094   Mean   :0.2322                                     Mean   :0.5413
##  3rd Qu.:0.3514   3rd Qu.:0.0000                                     3rd Qu.:0.6761
##  Max.   :1.0000   Max.   :1.0000                                     Max.   :1.0000
##  ExerciseAngina    Oldpeak      ST_Slope      HeartDisease
##  Length:913        Min.   :0.0000   Length:913   Min.   :0.0000
##  Class  :character  1st Qu.:0.2955   Class  :character   1st Qu.:0.0000
##  Mode   :character  Median :0.3523   Mode   :character   Median :1.0000
##                                     Mean   :0.3958                                     Mean   :0.5509
##                                     3rd Qu.:0.4659                                     3rd Qu.:1.0000
##                                     Max.   :1.0000                                     Max.   :1.0000

```

```

heartData$ST_Slope <- as.factor(heartData$ST_Slope)
heartData$ChestPainType <- as.factor(heartData$ChestPainType)
heartData$ExerciseAngina <- as.factor(heartData$ExerciseAngina)
heartData$Sex <- as.factor(heartData$Sex)
heartData$RestingECG <- as.factor(heartData$RestingECG)

```

```
summary(heartData)
```

```

##      Age      Sex      ChestPainType      RestingBP      Cholesterol
##  Min.   :0.0000  F:191  ASY:493   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.3878  M:722  ATA:173   1st Qu.:0.3571   1st Qu.:0.2490

```

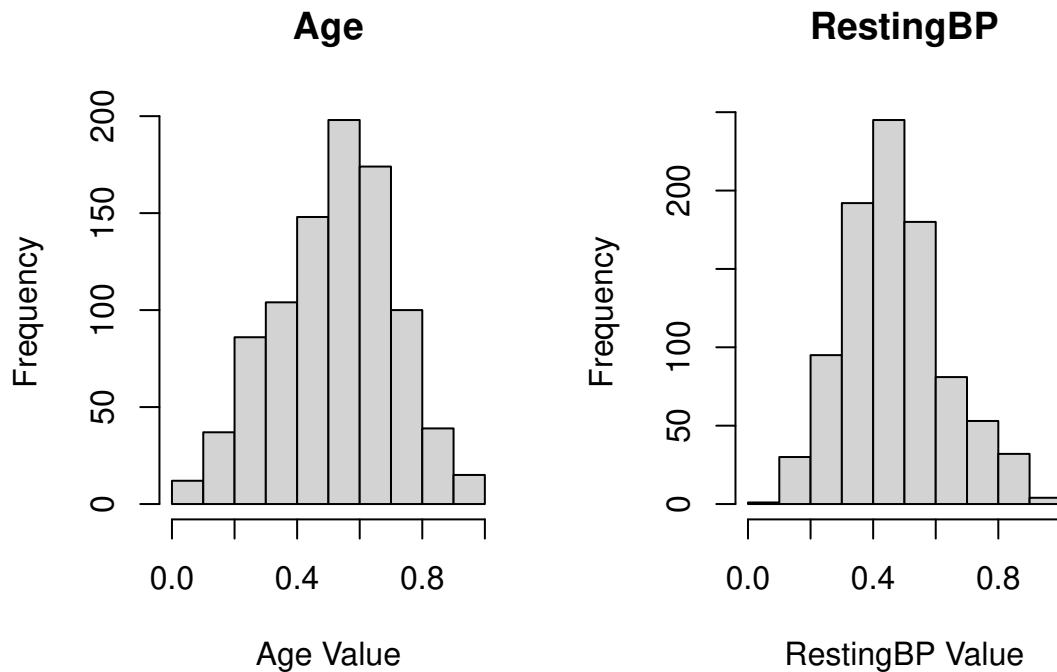
```

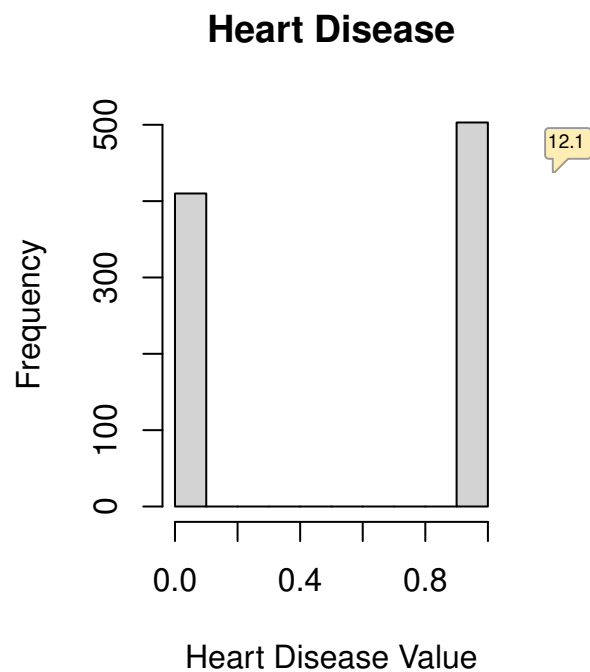
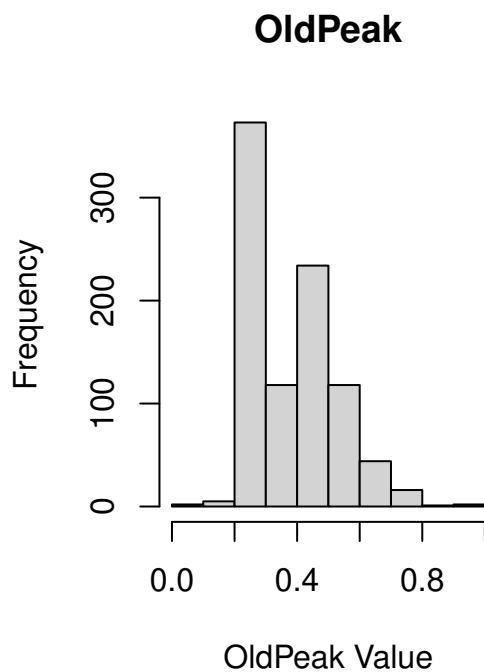
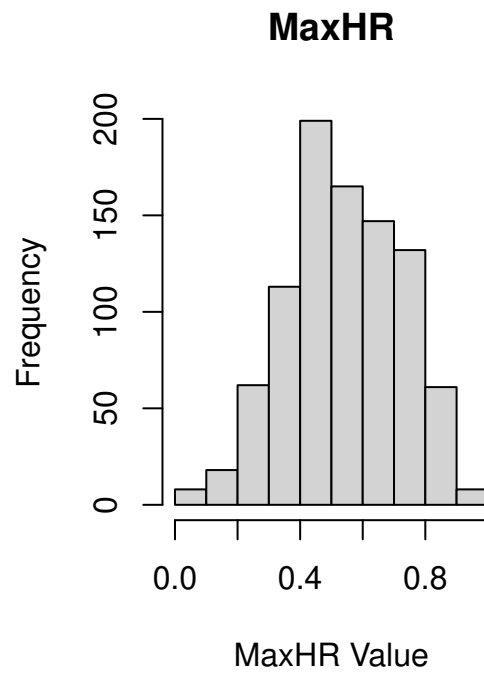
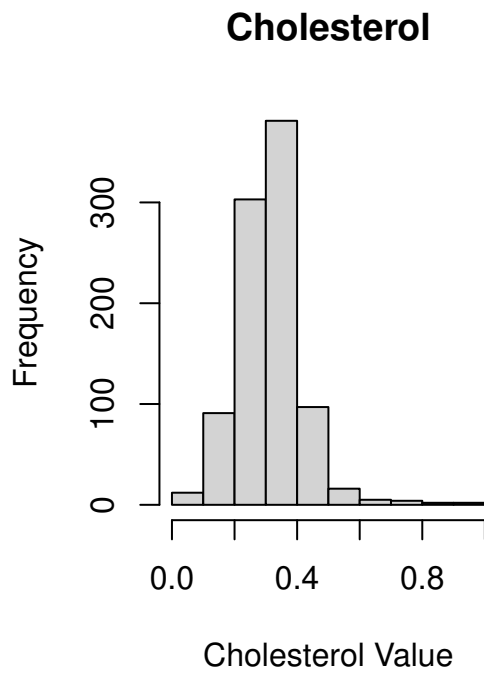
## Median :0.5306          NAP:201          Median :0.4464      Median :0.3069
## Mean   :0.5201          TA : 46           Mean   :0.4665      Mean   :0.3094
## 3rd Qu.:0.6531          3rd Qu.:0.5357      3rd Qu.:0.3514
## Max.   :1.0000          Max.   :1.0000      Max.   :1.0000
## FastingBS      RestingECG      MaxHR      ExerciseAngina      Oldpeak
## Min.   :0.0000      LVH   :187      Min.   :0.0000      N:545              Min.   :0.0000
## 1st Qu.:0.0000      Normal:549      1st Qu.:0.4225      Y:368              1st Qu.:0.2955
## Median :0.0000      ST    :177      Median :0.5493          Median :0.3523
## Mean   :0.2322          Mean   :0.5413          Mean   :0.3958
## 3rd Qu.:0.0000          3rd Qu.:0.6761          3rd Qu.:0.4659
## Max.   :1.0000          Max.   :1.0000          Max.   :1.0000
## ST_Slope      HeartDisease
## Down: 62      Min.   :0.0000
## Flat:456      1st Qu.:0.0000
## Up   :395      Median :1.0000
##                      Mean   :0.5509
##                      3rd Qu.:1.0000
##                      Max.   :1.0000

```

6 Detailed Data Analysis

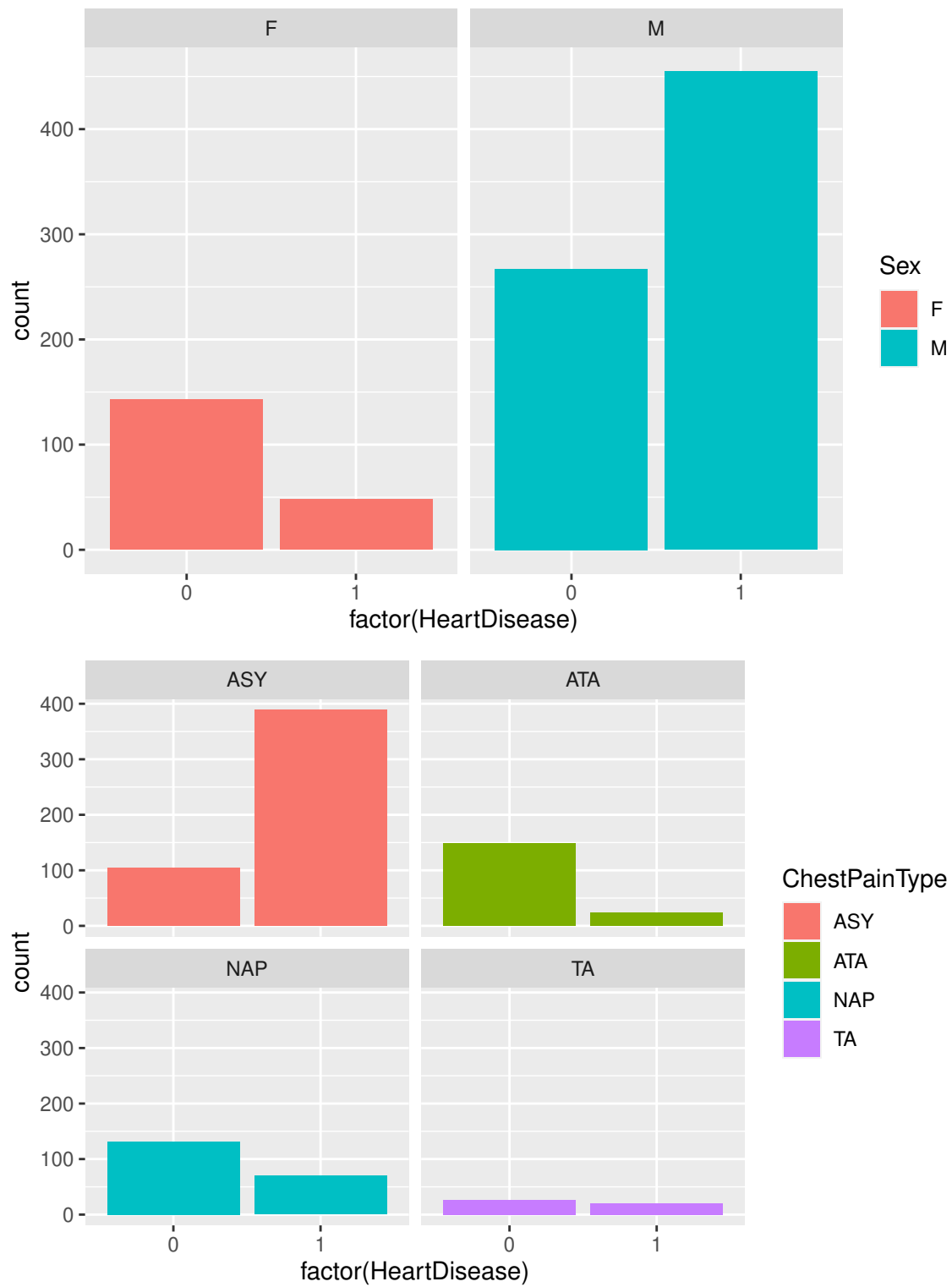
Drawing histogram of the data can give insight about the distribution of the data.

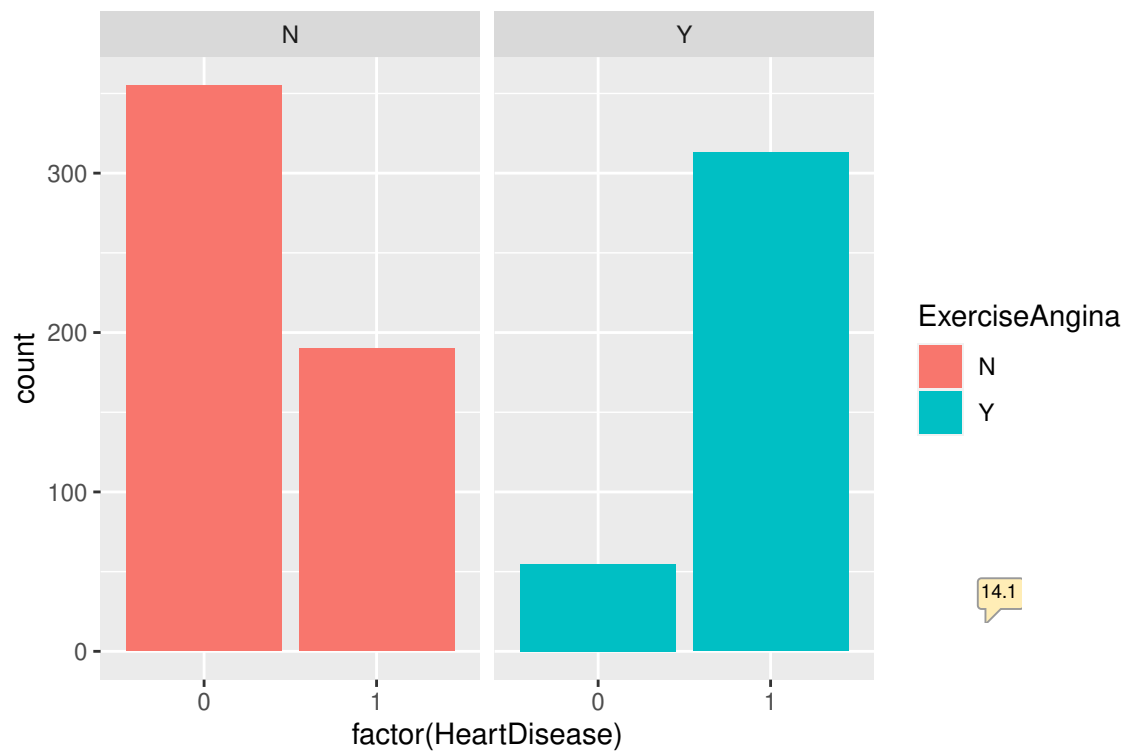
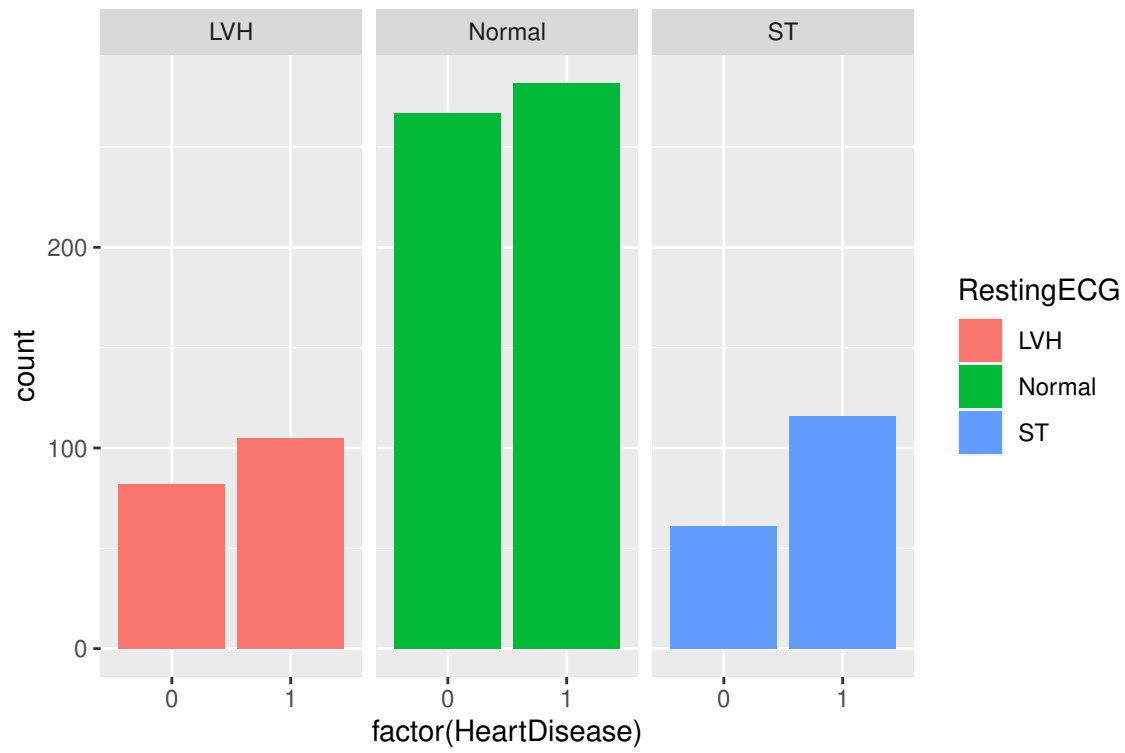




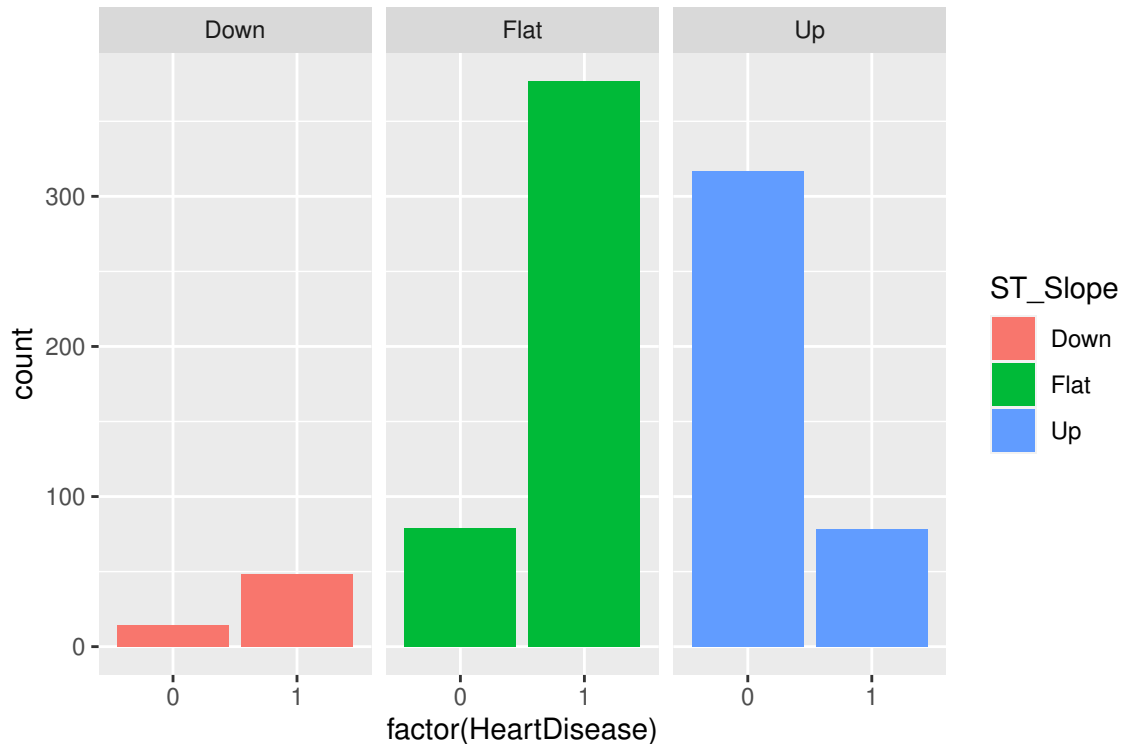
Since the data is nearly equally distributed for diseased and non-diseased, there is no need for oversampling or undersampling. Nearly all histogram charts resemble a bell curve, which satisfies the assumption of normal distribution. However, in cholesterol data a considerable amount of the data is clustered on 0-50 interval.

6.1 Categorical Data Analysis





14.1



Categorical features are split into half according to heart disease value. 0 symbolizes the non-existence of heart disease and 1 symbolizes the existence of heart disease. The given analysis are mostly estimates from the above plots.

6.1.1 Sex Analysis

When the first plot examined, the following conclusion can be drawn; male patients are more likely to have a heart disease. There around 700 male patients. More than half of these patients have heart disease. In addition female patients are likely to have no heart disease as the “0” values for females are higher than “1” values. Additionally, there are around 200 female patients in the data where nearly 50 of them have heart disease. This column will be a good fit for the model training phase since, this column make a clear distinction between the heart disease.

6.1.2 Chest Pain Type Analysis

There are 4 types of chest pain which are ASY, ATA, NAP, TA. From the above chest pain type plot, it is clearly seen that ASY chest pain type distinguishes the heart disease. In ASY sub category around 100 patients have no heart disease and patients that have heart disease, ratio is up to 4 times larger than healthy patients. Same way the ATA chest pain type clearly splits the healthy and diseased patients. Also NAP chest pain type classifies the healthy and diseased ones but not good as ASY and ATA. On the other hand the TA chest pain type is not successful enough while determining the heart disease. In summary, this column is suitable for the model training data because first 3 chest pain type can be used in predicting the heart disease.

6.1.3 Resting ECG Analysis

There are 3 types of resting ecg which is LVH, Normal and ST. The LVH resting ecg type slightly classify the healthy and diseased patients. The normal resting ecg type is similar to the LVH type. Both of them

can slightly useful when making categorization. Lastly the ST resting ecg type can determine the patients who have heart disease since its the diseased patients are nearly twice as healthy patients, around 60 healthy and around 120 diseased category. In sum this column especially ST type is suitable for the model training due to fact that its categorical values shows the distinction between healthy and diseased patients.

6.1.4 Exercise Angina Analysis


Angina appear when heart do not get enough blood. There are 2 type which is no and yes. In no sub category, there are around 300 patients whose do not have heart disease and around 200 patients have a heart disease. This sub category may be useful when deciding the heart disease result of a patient. The yes sub category have around 50 patients who do not have heart disease and around 330 patients have heart disease. The difference between the healthy and diseased patients count is huge. This yes sub category makes better classification than the no sub category. This column can be very useful when deciding the occurrence of heart disease.

6.1.5 ST Slope Analysis

There are 3 types of ST slope which is down slopping, flat and up sloping. In the down sub category around 20 healthy patients and 50 patients have heart disease. This sub category can be slightly useful on deciding the heart disease. The second subcategory is the flat type having around 80 patients. These patients do not have heart disease and around 380 patient have heart disease. This sub category is suitable for the model training part since its data distribution is shows the separation clearly. The last type is up slope. This sub category has around 320 patients who do not have heart disease and around 80 patients who have heart disease. This huge difference can be useful when making classification. As a last word, this column is suitable for the model training phase.

7 Data Partioning

The data is split according to the general rule of thumb, 80% for training data and 20% for testing data.




```
# Creation of sample for model training and testing
row.number <- sample(1:nrow(heartData), 0.8*nrow(heartData))

train= heartData[row.number,]
test = heartData[-row.number,]
```

8 Logisitic Regression

The multiple logistic regression model has been fit using all the features. The summary of the trained model is given as:



```
glmModel <- glm(data = train, formula = HeartDisease ~ ., family = "binomial")
summary(glmModel)

##
## Call:
## glm(formula = HeartDisease ~ ., family = "binomial", data = train)
##
## Deviance Residuals:
```



```
##      Min      1Q   Median      3Q      Max
## -2.9268 -0.3978  0.2001   0.4631  2.6637
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.88621    1.10771  -1.703 0.088606 .
## Age           0.85756    0.72054   1.190 0.233986
## SexM          1.67147    0.31044   5.384 7.28e-08 ***
## ChestPainTypeATA -1.76357    0.37549  -4.697 2.64e-06 ***
## ChestPainTypeNAP -1.39335    0.28892  -4.823 1.42e-06 ***
## ChestPainTypeTA  -1.17743    0.50993  -2.309 0.020942 *
## RestingBP       0.06260    0.78303   0.080 0.936285
## Cholesterol     0.95455    1.15940   0.823 0.410332
## FastingBS       1.41057    0.29339   4.808 1.53e-06 ***
## RestingECGNormal -0.01001    0.30413  -0.033 0.973752
## RestingECGST     -0.19465    0.38091  -0.511 0.609345
## MaxHR          -1.42388    0.78355  -1.817 0.069183 .
## ExerciseAnginaY  0.90109    0.26496   3.401 0.000672 ***
## Oldpeak         2.72011    1.13284   2.401 0.016344 *
## ST_SlopeFlat     0.84135    0.47635   1.766 0.077355 .
## ST_SlopeUp       -1.38551    0.49999  -2.771 0.005587 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 1004.07  on 729  degrees of freedom
## Residual deviance:  487.57  on 714  degrees of freedom
## AIC: 519.57
##
## Number of Fisher Scoring iterations: 5
```

```
model_glm_pred = ifelse(predict(glmModel, newdata = test, type = "response") > 0.5, 1, 0)
```

A residual is the difference between an observed value and a predicted value in a model. The median of the deviance residuals is 0.2001. The minimum value is -2.9268 and maximum is 2.6637. First quartile range is -0.3978 and third quartile is 0.4631.

The $\text{Pr}(>|z|)$ column contains the p-value linked with the z value and lower p-values would indicate that the variable in the model is relevant. At the first glance, Sex, ChestPainTypeATA, ChestPainTypeNAP, Cholesterol, FastingBS, ExerciseAnginaY predictors have better chance of changing the outcome of the null hypothesis. Possible hypothesis would be “the patient has heart disease” and these variables are influential.

From the above model summary, all the columns that have stars will used to train a new model.

```
specificModel <- glm(data = train, formula = HeartDisease ~ Sex + ChestPainType + ExerciseAngina + ST_S
specificModel_pred = ifelse(predict(specificModel, newdata = test, type = "response") > 0.5, 1, 0)
```

After training the models the best error rate is evaluated using cross validation. The cross validation and analysis for the model is given below.

As a remark, separate validation set was not used as the it can host high variability depending on the included samples, as well as it reduces the number of training samples when divided.

Three cross validation was applied, namely, leave-one-out cross validation, 5-fold cross validation and 10-fold cross validation. The reason why three different techniques are employed can be attributed to variance-bias trade-off. LOOCV partitions the data into 1 sample for validation and m-1 samples for training. Because of that, the variance is generally higher when compared to the 5-fold and 10-fold cross validation. In 5-fold cross validation, the data is split to 5 equal chunks and the validation is done one of the chunks for each fold, which is similar in 10-fold cross validation, too.

```
# leave-one-out cross-validation estimate without any extra model-fitting
cv_loo <- cv.glm(train, glmModel)

cv_5fold <- cv.glm(train, glmModel, K = 5) # Corresponds 5-fold CV

cv_10fold <- cv.glm(train, glmModel, K = 10) # Corresponds 10-fold CV
```

The delta array bring the raw cross-validation prediction error and the second component gives the bias corrected cross-validation error. The errors for cross validation is calculated by given formula.

$$MSE = \frac{1}{m} \sum_{i=1}^m (y - \hat{y})^2$$

\hat{y} symbolizes the predicted values

y symbolizes the actual values

m symbolizes the number of data points

```
cv_loo$delta
```

```
## [1] 0.1083402 0.1083350
```

```
cv_5fold$delta
```

```
## [1] 0.1094223 0.1083839
```

```
cv_10fold$delta
```

```
## [1] 0.1107416 0.1102618
```

```
## [1] "LOOCV Raw error: 0.1083402 and the bias corrected error: 0.1083350"
```

```
## [1] "5-Fold Raw error: 0.1094223 and the bias corrected error: 0.1083839" ?
```

```
## [1] "10-Fold Raw error: 0.1107416 and the bias corrected error: 0.1102618"
```

Among the three cross validation folds, the lowest raw error comes from LOOCV with 0.1083402, followed by 5-fold cross validation with 0.1094223 and lastly 10-fold has the highest raw error with 0.1107416.

When the raw error is adjusted according to bias values, again, lowest bias corrected error comes from LOOCV with 0.1083350, followed by with a small difference by 0.0000489, the adjusted error for 5-fold cross validation comes as 0.1083839 and lastly 10-fold has the highest raw error with 0.1102618.

Initially, the suitable cross validation for seems to be LOOCV since it has both low raw error and bias corrected error. It seems to be the best cross validation model for heart disease prediction case. Yet, it has the risk of overfitting the data and because of that, the error rates can be deceiving. Additionally, the error difference between LOOCV and 5-fold CV can be ignored as the difference is only measurable with sensitive

tools. Also, LOOCV requires the most computation power which is a clear disadvantage if the data was much more larger.

Moreover, the 10-fold cross validation has the highest error rate and the needs more computation power when compared with 5-fold CV.

With the mentioned reasons, one can prefer the 5-fold cross validation over LOOCV and 10-fold.

8.1 Result Analysis

FN (Type2 error) is predicted negative and it is false.(Wrong prediction). FP (Type1 error) is predicted positive and it is false.(Wrong prediction). TN is predicted negative and it is true. TP is predicted positive and it is true.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$


$$\text{Precision} = \frac{TP}{TP+FP} \text{ or } \frac{\text{TruePositive}}{\text{PredictiveResults}}$$

Precision (Pos Pred Value in R language) symbolizes the quantity of the predicted correctly in predicted positive classes.

$$\text{Recall} = \frac{TP}{TP+FN} \text{ or } \frac{\text{TruePositive}}{\text{ActualResults}}$$

Recall (Sensitivity in R language) symbolizes the quantity of the predicted correctly in all positive classes.

When the real life considered, if a patient has a heart disease, this accepted as a negative situation. If patient has no heart disease, it is a positive situation. Assume that when looking the confusion matrix the 0 0 index is TP.



```
#confusion matrix creation
testDiseased_factor = factor(test$HeartDisease, levels=c(0, 1))
model_factor = factor(model_glm_pred, levels=c(0, 1))

confusionMatrix(data= model_factor, reference= testDiseased_factor)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0   1
##           0 69 11
##           1 14 89
##
##           Accuracy : 0.8634
##           95% CI : (0.805, 0.9096)
##           No Information Rate : 0.5464
##           P-Value [Acc > NIR] : <2e-16
##
##           Kappa : 0.7235
##
##           McNemar's Test P-Value : 0.6892
##
##           Sensitivity : 0.8313
##           Specificity : 0.8900
##           Pos Pred Value : 0.8625
##           Neg Pred Value : 0.8641
##           Prevalence : 0.4536
##           Detection Rate : 0.3770
```

```
## Detection Prevalence : 0.4372
## Balanced Accuracy : 0.8607
##
## 'Positive' Class : 0
##
```

From the above logistic regression's confusion matrix it can be obtained that, Logistic regression model has 86.34% accuracy. This is obtained from; sum of the TPs and TNs (correct predictions) are $69 + 89 = 158$ out of 183 total cases. $158/183 \approx 0.86338$.

Recall value can be obtained from TP count divided to sum of the TP and FN which is: $69/(69+14) = 69/83 \approx 0.8313$

Precision value can be obtained from TP count divided to sum of the TP and FP which is: $69/(69+11) = 69/80 \approx 0.8625$

```
specificModel_factor = factor(specificModel_pred, levels=c(0, 1))

confusionMatrix(data= specificModel_factor, reference= testDiseased_factor)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##           0 70 13
##           1 13 87
##
##           Accuracy : 0.8579
##           95% CI : (0.7988, 0.905)
## No Information Rate : 0.5464
## P-Value [Acc > NIR] : <2e-16
##
##           Kappa : 0.7134
##
## Mcnemar's Test P-Value : 1
##
##           Sensitivity : 0.8434
##           Specificity : 0.8700
##           Pos Pred Value : 0.8434
##           Neg Pred Value : 0.8700
##           Prevalence : 0.4536
##           Detection Rate : 0.3825
## Detection Prevalence : 0.4536
##           Balanced Accuracy : 0.8567
##
##           'Positive' Class : 0
##
```



The above confusion matrix belongs to the model that trained for the specific columns and its Recall value is better than normal logistic regression model but Precision value is lower than normal logistic regression model. To determine which one is suitable for the heart disease prediction case the F score should be analyzed.

```
## [1] "F score for Logistic model is: 0.84663"
```

```
## [1] "F score for Specific Logistic model is: 0.84337"
```

From the above comparison the normal logistic regressions F score is better by small difference. As a result there is no need to involve the Specific logistic regression model to comparison part.

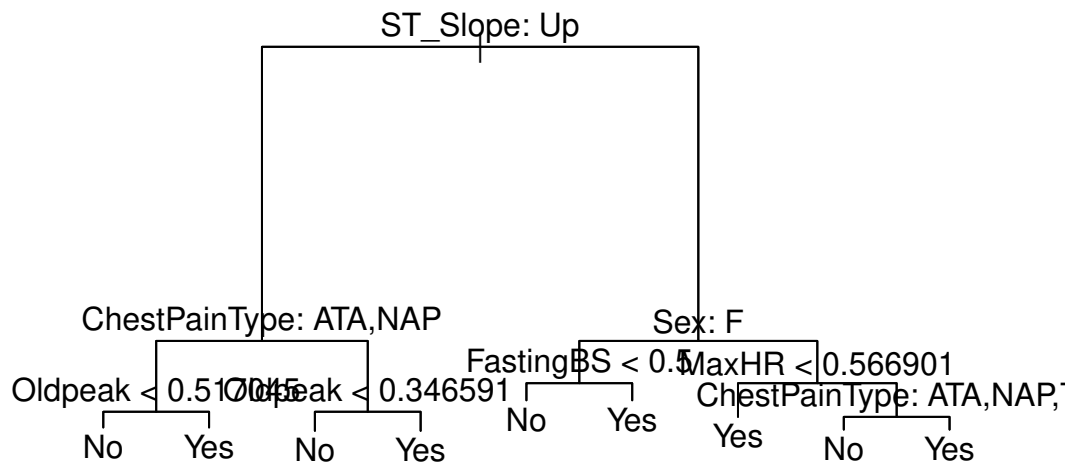
9 Decision Tree

At this point the response value is binary categorical response so that there is no need to make further operations for the HeartDisease data. While creating the tree we do not need to drop any columns and we need to use all the features to get the best value. Because of the response data type (integer) when we directly apply the tree method we get Regression tree output, since this is not what we desire we convert this '0' and '1' values of response HeartDisease data into Yes and No with using factor method. After applying it we observe a classification tree like we expected.

With the summary method we can observe the variables to produce this tree but since these variables change based on the training data, deep and detailed discussion is not needed about it. To avoid these meaningless changes based on the random selection of training data we should apply cross validation to look at the overall performance.

```
# Creation of sample for subset usage of tree
heartData$HeartDisease <- factor(ifelse(heartData$HeartDisease < 0.5, "No", "Yes"))
HeartDisease.test <- heartData$HeartDisease[-row.number]

# To to implement tree
tree.heartData <- tree(HeartDisease~., heartData, subset = row.number)
#to Plot
plot(tree.heartData)
text(tree.heartData , pretty = 0)
```



```
summary(tree.heartData)
```

```
##
## Classification tree:
## tree(formula = HeartDisease ~ ., data = heartData, subset = row.number)
## Variables actually used in tree construction:
## [1] "ST_Slope"      "ChestPainType" "Oldpeak"      "Sex"
## [5] "FastingBS"     "MaxHR"
## Number of terminal nodes: 9
## Residual mean deviance: 0.6701 = 483.1 / 721
## Misclassification error rate: 0.1438 = 105 / 730
```

9.1 Result Analysis

Without pruning the data we can observe the confusion matrix of the largest tree.

```
# Predictions on largest tree
tree.pred <- predict(tree.heartData , test , type = "class")

confusionMatrix(data=tree.pred, reference=HeartDisease.test)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction No Yes
##           No  72  30
##           Yes  11  70
```

```
##
##           Accuracy : 0.776
##           95% CI : (0.7086, 0.8342)
##      No Information Rate : 0.5464
##      P-Value [Acc > NIR] : 9.168e-11
##
##           Kappa : 0.5566
##
##  McNemar's Test P-Value : 0.004937
##
##      Sensitivity : 0.8675
##      Specificity : 0.7000
##      Pos Pred Value : 0.7059
##      Neg Pred Value : 0.8642
##      Prevalence : 0.4536
##      Detection Rate : 0.3934
##      Detection Prevalence : 0.5574
##      Balanced Accuracy : 0.7837
##
##      'Positive' Class : No
##
```

From the above Decision Tree's confusion matrix it can be obtained that, Decision Treen model has 77.6% accuracy. This is obtained from; sum of the TPs and TNs (correct predictions) are $72 + 70 = 142$ out of 183 total cases. $142/183 \approx 0.7759$.

Recall value can obtained from TP count divided to sum of the TP and FN which is: $72/(72 + 11) = 72/83 \approx 0.86746$

Precision value can be obtained from TP count divided to sum of the TP and FP which is: $72/(72 + 30) = 72/102 \approx 0.70588$

9.2 Pruning

Now we can make pruning operations to get minimum misclassification error rate because of this reason I will use `prune.misclass` method. From this method we can observe the `k` variable which is `alpha`, the penalty term, but to select pruning level, we will create the plots and we can see the joint behaviour of the size-deviance, `k`-deviance and finally `k`-size plots. From the dev and size plot we can observe the min value of the deviance and the respected size of it. So size of 4, 6, 7 have the lowest deviances. For further operations, we will use size 4.

```
# To decide the prunning level
cv.heartData <- cv.tree(tree.heartData , FUN = prune.misclass)
names(cv.heartData)
```

```
## [1] "size"  "dev"   "k"     "method"
```

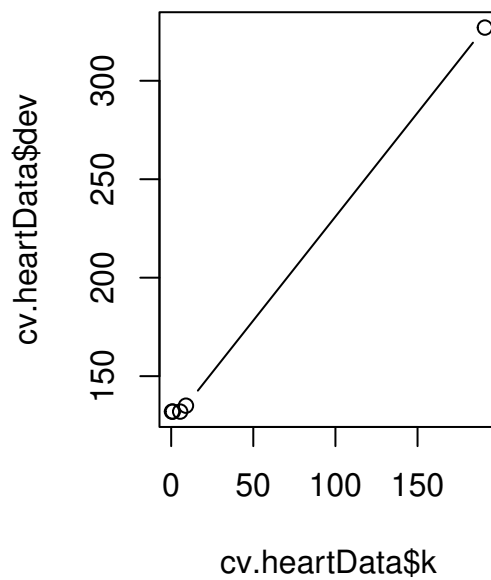
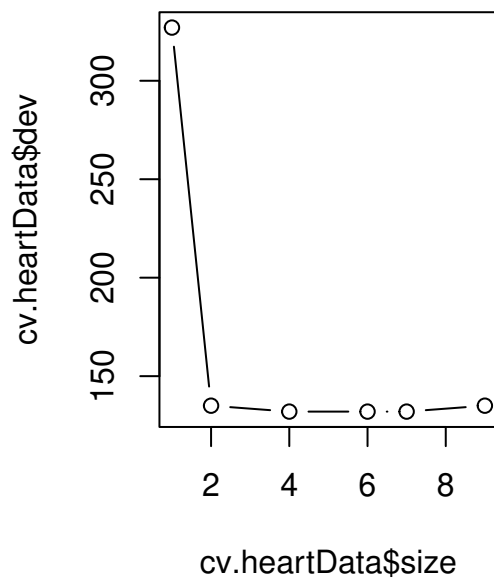
```
cv.heartData
```

```
## $size
## [1] 9 7 6 4 2 1
##
## $dev
```

```
## [1] 135 132 132 132 135 327
##
## $k
## [1] -Inf  0.5  1.0  5.5  9.0 191.0
##
## $method
## [1] "misclass"
##
## attr("class")
## [1] "prune"          "tree.sequence"
```

```
par(mfrow = c(1, 2))

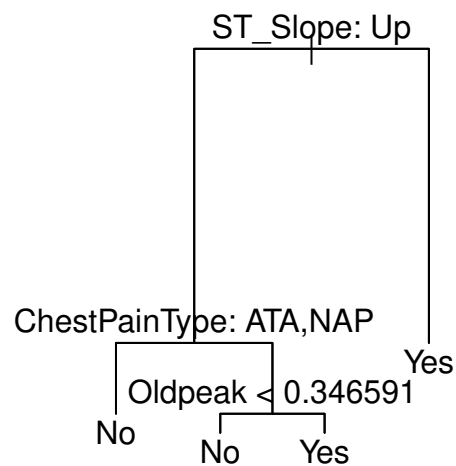
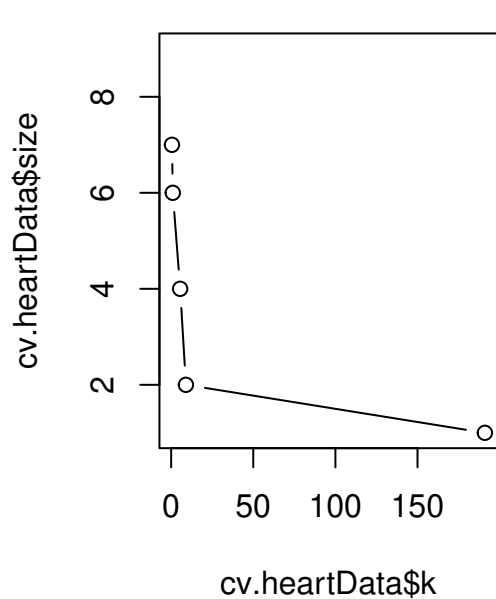
plot(cv.heartData$size , cv.heartData$dev, type = "b")
plot(cv.heartData$k, cv.heartData$dev, type = "b")
```



```
plot(cv.heartData$k, cv.heartData$size, type = "b")
```

```
prune.heartData <- prune.misclass(tree.heartData , best = 4)
```

```
#Pruned tree
plot(prune.heartData)
text(prune.heartData , pretty = 0)
```

9.3 Result Analysis

After pruning the tree now we can make predictions with using the test data and the pruned model to calculate accuracy. In this sense, we can observe the output of the accuracy based on the sample selection. Sometimes we can observe better output for pruned data but sometimes not so it may depend data's characteristics.

```
tree.pred.Pruned <- predict(prune.heartData, test, type = "class")
confusionMatrix(data=tree.pred.Pruned, reference=HeartDisease.test)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction No Yes
##           No  57   6
##           Yes  26  94
##
##           Accuracy : 0.8251
##           95% CI : (0.7622, 0.8772)
##           No Information Rate : 0.5464
##           P-Value [Acc > NIR] : 1.715e-15
##
##           Kappa : 0.6399
##
##           McNemar's Test P-Value : 0.0007829
##
```

```
##          Sensitivity : 0.6867
##          Specificity : 0.9400
##          Pos Pred Value : 0.9048
##          Neg Pred Value : 0.7833
##          Prevalence : 0.4536
##          Detection Rate : 0.3115
##          Detection Prevalence : 0.3443
##          Balanced Accuracy : 0.8134
##
##          'Positive' Class : No
##
```


From the above Pruned Decision Tree's confusion matrix it can be obtained that, Pruned Decision Tree model has 82.51% accuracy. This is obtained from; sum of the TPs and TNs (correct predictions) are $57 + 94 = 151$ out of 183 total cases. $151/183 \approx 0.82513$.

Recall value can be obtained from TP count divided to sum of the TP and FN which is: $57/(57+26) = 57/83 \approx 0.68674$

Precision value can be obtained from TP count divided to sum of the TP and FP which is: $57/(57+6) = 57/63 \approx 0.90476$

9.4 Rpart Library Decision Tree


At this point we will create our decision tree with using the Rpart library. Since our accept feature Heart-Disease is int value on train, the rparts library detects this tree as a regression tree so that factor operation is applied on it beforehand. Meanwhile we will apply the 10 fold cross validation to create our model. Furthermore we use tuneLength. Tunelenght property helps the system to tune our model automatically. Furthermore, it indicates the number of different values to try for each tuning parameter. We selected it as 5 To make observations.



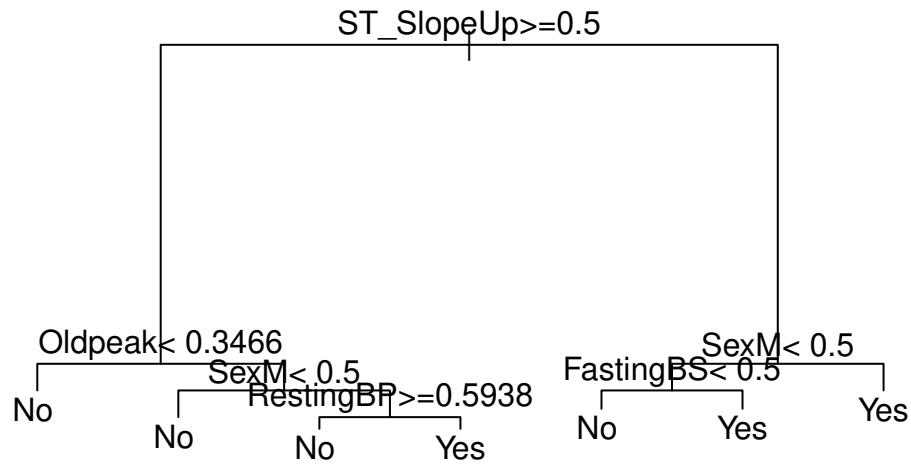
```
train$HeartDisease <- factor(ifelse(train$HeartDisease < 0.5, "No", "Yes"))
test$HeartDisease <- factor(ifelse(test$HeartDisease < 0.5, "No", "Yes"))

# Try also 10-fold CV below !
model2 <- train(
  HeartDisease ~., data = train, method = "rpart",
  trControl = trainControl("cv", number = 10),
  tuneLength = 5
)
```

As we can see here, rpart algorithm decided to go with 7 leaf nodes although it reached this conclusion, the tuneLength variable significantly changes number of leaf nodes. But at the end we get the pruned and 10 fold cross validated Decision Tree with around 0.8033 accuracy. So, we can say that this library performs better when compared with the unpruned tree in tree library but it performs worse than the manually pruned tree from the tree library.



```
# Plot the final tree model
par(xpd = NA) # Avoid clipping the text in some device
plot(model2$finalModel)
text(model2$finalModel, digits = 3)
```



```
# Decision rules in the model, as a pruned tree
```

```
model2$finalModel
```

```
## n= 730
##
## node), split, n, loss, yval, (yprob)
##      * denotes terminal node
##
## 1) root 730 327 Yes (0.4479452 0.5520548)
##    2) ST_SlopeUp>=0.5 321 65 No (0.7975078 0.2024922)
##      4) Oldpeak< 0.3465909 248 28 No (0.8870968 0.1129032) *
##      5) Oldpeak>=0.3465909 73 36 Yes (0.4931507 0.5068493)
##        10) SexM< 0.5 18 2 No (0.8888889 0.1111111) *
##        11) SexM>=0.5 55 20 Yes (0.3636364 0.6363636)
##          22) RestingBP>=0.59375 15 4 No (0.7333333 0.2666667) *
##          23) RestingBP< 0.59375 40 9 Yes (0.2250000 0.7750000) *
##    3) ST_SlopeUp< 0.5 409 71 Yes (0.1735941 0.8264059)
##      6) SexM< 0.5 62 31 No (0.5000000 0.5000000)
##        12) FastingBS< 0.5 51 20 No (0.6078431 0.3921569) *
##        13) FastingBS>=0.5 11 0 Yes (0.0000000 1.0000000) *
##      7) SexM>=0.5 347 40 Yes (0.1152738 0.8847262) *
```

```
# Make predictions on the test data
```

```
predicted.classes <- model2 %>% predict(test)
```

```
# Confusion matrix idea
```

```
confusionMatrix(data=predicted.classes, reference=HeartDisease.test)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction No Yes
##           No  63  16
##           Yes 20  84
##
##           Accuracy : 0.8033
##           95% CI : (0.7382, 0.8583)
##           No Information Rate : 0.5464
##           P-Value [Acc > NIR] : 3.032e-13
##
##           Kappa : 0.6015
##
## Mcnemar's Test P-Value : 0.6171
##
##           Sensitivity : 0.7590
##           Specificity : 0.8400
##           Pos Pred Value : 0.7975
##           Neg Pred Value : 0.8077
##           Prevalence : 0.4536
##           Detection Rate : 0.3443
##           Detection Prevalence : 0.4317
##           Balanced Accuracy : 0.7995
##
##           'Positive' Class : No
##
```

✓ Rpart library's 10 fold Decision Tree's confusion matrix is obtained above. This model has 80.33% accuracy. This is obtained from; sum of the TPs and TNs (correct predictions) are $63 + 84 = 147$ out of 183 total cases. $147/183 \approx 0.80327$.

Recall value can be obtained from TP count divided to sum of the TP and FN which is: $63/(63+20) = 63/83 \approx 0.75903$

Precision value can be obtained from TP count divided to sum of the TP and FP which is: $63/(63+16) = 63/79 \approx 0.79746$

```
sprintf("F score for 10 fold Decision Tree model is: %.5f", F_meas(data=predicted.classes, reference=He
```

```
## [1] "F score for 10 fold Decision Tree model is: 0.77778"
```

✓ The rparts 10 fold decision tree's recall value is lower than the normal decision tree's recall value(0.7590 vs 0.8675). In heart disease prediction case the FN which directly affects the Recall value, count is important. This model predicted patients who has disease as healthy category. So this model makes lots of misclassification in predicting patients who have heart diseased.

Heart disease prediction case is part of healthcare, and healthcare cannot be tolerated. Because of that FN count must be low. This model is not suitable for heart disease prediction case.

10 Comparison

$$F \text{ score} = \frac{2 * Recall * Precision}{Recall + Precision}$$

F score symbolizes the harmonic mean of recall and precision. The reason of why f score used is arithmetic mean cannot reduce the outliers more than harmonic mean.

```
## [1] "Recall value for Unpruned Decision Tree model is: 0.86747"

## [1] "Precision value for Unpruned Decision Tree model is: 0.70588"

## [1] "F score for Unpruned Decision Tree model is: 0.77838"

## [1] "Recall value for Pruned Decision Tree model is: 0.68675"

## [1] "Precision value for Pruned Decision Tree model is: 0.90476"

## [1] "F score for Pruned Decision model Tree is: 0.78082"

## [1] "Recall value for Rpart 10-fold Decision Tree model is: 0.75904"

## [1] "Precision value for Rpart 10-fold Decision Tree model is: 0.79747"

## [1] "F score for Rpart 10-fold Decision model Tree is: 0.77778"

## [1] "Recall value for Logistic model is: 0.83133"

## [1] "Precision value for Logistic model is: 0.86250"

## [1] "F score for Logistic model is: 0.84663"
```

Recall value will be considerable where the case is intolerable to false negative. The healthcare topics often focus this value. Precision value will be considerable where false positive is more important. The intricate topics often focus on this value. F score's main purpose is to reduce the effect of variation in recall and precision values. For example, if Recall value is high but Precision value is low, the F score will be low. If Precision value is high but Recall value is low, again F score will be low. F score will be high only when both of Recall and Precision values are high. Because of this reason, F score is more valuable when deciding the better model rather than arithmetic mean (accuracy).

When it comes to health, the priority is to detect heart disease correctly. To decide which model is suitable for heart disease case F score, specificity and recall values will be compared.

Among the above 4 models, Unpruned Decision Tree's recall is the best with 0.8675 points, followed by Logistic Regression's recall value 0.8313. Followed by 10-fold rpart Tree with the value 0.7590 the last one is Pruned Decision Tree's recall value 0.6868.

For the precision values, the pruned decision tree's value is the best. It has the value of 0.9048. The next model is the logistic regression with 0.8625 which is followed by the 10-fold rpart decision tree which has the precision value of 0.7975 and the last one is the Unpruned Decision Tree having 0.7059.

From the F score perspective, the best model is again Logistic Regression by 0.8466 points, followed by Pruned Decision model with 0.7808 and the Unpruned Decision Tree model with 0.7784 points. The worst F score belongs to the 10-fold Rpart Decision Tree 0.7778

In addition to that Logistic regression's accuracy value is the best with 0.8634 points followed by Pruned Decision Tree model with 0.8251 points. The 10-fold Rpart Decision Tree's accuracy is 0.8033 and the last model is Unpruned Decision Tree model by 0.776 points.

The best model can be selected according to various parameters. If the absolute percentages of correct classifications were important, then, the accuracy can be the main chosen for model selection. However accuracy does not give information about the false negatives and false positives. Predicting diseased patients as healthy is a dangerous outcome and it is not acceptable in this case so the models which have high FN count will not be satisfactory for heart disease prediction. High FN count can be measured using recall values so the two models with lowest recall values will be eliminated. The Pruned Decision Tree and Rpart Decision Tree will be discarded.

The remaining models will be compared according to their F scores. The logistic regression have the highest F score whereas the Unpruned Decision Tree have less F score. Due to this it should not be used in heart disease detection. From the above observations, it can be concluded that, the models with high recall and high F score is more suitable for heart disease case. Therefore, the best model for this heart disease is Logistic Regression model. The reason of it may the data distribution is better separated by Logistic Regression line which is similar to the “S” character. In the end, this model exhibits better performance than Decision Tree and Pruned Decision Tree models.

11 References

<https://www.kaggle.com/datasets/fedesoriano/heart-failure-prediction>

Index of comments

12.1 For this one, barplot can be considered indeed since it is a categorical one. Keep in mind for later studies

14.1 Good visualizations