Heart Disease Project

Akash Kumar

June 17, 2019

Contents

Introduction

Background

Everyday, more and more people are at risk of getting heart disease. So, it is of great importance for doctors to be able to predict who will get heart disease so that appropriate measures can be taken before the condition becomes serious.

Inspiration

Many people have always wondered how doctors are able to predict certain diseases in patients using data about the patient's health. Through this project, we would like to gain some insight on how exactly that process works.

Focus

The focus of this project is to predict heart disease in patients with as high of an accuracy as possible.

Dataset

For this assignment, we will be using the cleaned version of the Cleveland database from the Kaggle website.

Goal

The goal of the project is to develop and train models to predict heart disease in patients with as high of an accuracy as possible. The accuracy of the predictions must be optimized such that the **accuracy >= 0.7**.
For this project, we will be building these models using R (version 3.6.0).

Setup and Cleaning

The following code is used to create the heart set, which is split into the train set and test set. Due to the small size of the dataset, we have opted not to create a validation set.

```r
# Install packages:
if(!require(tidyverse))
  install.packages("tidyverse", repos = "http://cran.us.r-project.org")
if(!require(caret))
  install.packages("caret", repos = "http://cran.us.r-project.org")
if(!require(pROC))
  install.packages("pROC", repos = "http://cran.us.r-project.org")
if(!require(randomForest))
  install.packages("randomForest", repos = "http://cran.us.r-project.org")
if(!require(klaR))
  install.packages("klaR", repos = "http://cran.us.r-project.org")
if(!require(kernlab))
  install.packages("kernlab", repos = "http://cran.us.r-project.org")

# Load the necessary libraries:
library(tidyverse)
library(caret)
library(pROC)
library(randomForest)
library(klaR)
library(kernlab)

# Heart Disease UCI dataset:
# https://www.kaggle.com/ronitf/heart-disease-uci
# https://archive.ics.uci.edu/ml/datasets/Heart+Disease

# heart set:
heart <- read.csv("heart.csv")
colnames(heart)[c(1,3,4,6,7,8,9,10,11,12)] <- c("age", "chest_pain", "resting_bps",
                              "blood_sugar_120", "resting_ecg",
                              "max_bpm", "angina", "st_oldpeak", "st_slope",
                              "vessel_count")
```

```r
heart <- heart[, c(1,2,4,8,5,6,7,3,9,10,11,12,13,14)] # Reorder some columns.
heart <- na.omit(heart) # Remove NAs.
head(heart)
```

```
##   age sex resting_bps max_bpm chol blood_sugar_120 resting_ecg chest_pain
## 1  63   1         145     150  233               1           0          3
## 2  37   1         130     187  250               0           1          2
## 3  41   0         130     172  204               0           0          1
## 4  56   1         120     178  236               0           1          1
## 5  57   0         120     163  354               0           1          0
## 6  57   1         140     148  192               0           1          0
##   angina st_oldpeak st_slope vessel_count thal target
## 1      0        2.3        0            0    1      1
## 2      0        3.5        0            0    2      1
## 3      0        1.4        2            0    2      1
## 4      0        0.8        2            0    2      1
## 5      1        0.6        2            0    2      1
## 6      0        0.4        1            0    1      1
```

```r
glimpse(heart)
```

```
## Observations: 303
## Variables: 14
## $ age             <int> 63, 37, 41, 56, 57, 57, 56, 44, 52, 57, 54, 48...
## $ sex             <int> 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0...
## $ resting_bps     <int> 145, 130, 130, 120, 120, 140, 140, 120, 172, 1...
## $ max_bpm         <int> 150, 187, 172, 178, 163, 148, 153, 173, 162, 1...
## $ chol            <int> 233, 250, 204, 236, 354, 192, 294, 263, 199, 1...
## $ blood_sugar_120 <int> 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0...
## $ resting_ecg     <int> 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1...
## $ chest_pain      <int> 3, 2, 1, 1, 0, 0, 1, 1, 2, 2, 0, 2, 1, 3, 3, 2...
## $ angina          <int> 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0...
## $ st_oldpeak      <dbl> 2.3, 3.5, 1.4, 0.8, 0.6, 0.4, 1.3, 0.0, 0.5, 1...
## $ st_slope        <int> 0, 0, 2, 2, 2, 1, 1, 2, 2, 2, 2, 2, 2, 1, 2, 1...
## $ vessel_count    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ thal            <int> 1, 2, 2, 2, 2, 1, 2, 3, 3, 2, 2, 2, 2, 2, 2, 2...
## $ target          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
```

```r
summary(heart)
```

```
##       age             sex          resting_bps       max_bpm     
##  Min.   :29.00   Min.   :0.0000   Min.   : 94.0   Min.   : 71.0  
##  1st Qu.:47.50   1st Qu.:0.0000   1st Qu.:120.0   1st Qu.:133.5  
##  Median :55.00   Median :1.0000   Median :130.0   Median :153.0  
##  Mean   :54.37   Mean   :0.6832   Mean   :131.6   Mean   :149.6  
##  3rd Qu.:61.00   3rd Qu.:1.0000   3rd Qu.:140.0   3rd Qu.:166.0  
##  Max.   :77.00   Max.   :1.0000   Max.   :200.0   Max.   :202.0  
##       chol       blood_sugar_120   resting_ecg       chest_pain   
##  Min.   :126.0   Min.   :0.0000   Min.   :0.0000   Min.   :0.000  
##  1st Qu.:211.0   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.000  
##  Median :240.0   Median :0.0000   Median :1.0000   Median :1.000  
##  Mean   :246.3   Mean   :0.1485   Mean   :0.5281   Mean   :0.967  
##  3rd Qu.:274.5   3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:2.000  
##  Max.   :564.0   Max.   :1.0000   Max.   :2.0000   Max.   :3.000  
##      angina         st_oldpeak      st_slope      vessel_count   
##  Min.   :0.0000   Min.   :0.00   Min.   :0.000   Min.   :0.0000  
##  1st Qu.:0.0000   1st Qu.:0.00   1st Qu.:1.000   1st Qu.:0.0000  
##  Median :0.0000   Median :0.80   Median :1.000   Median :0.0000  
##  Mean   :0.3267   Mean   :1.04   Mean   :1.399   Mean   :0.7294
##  3rd Qu.:1.0000   3rd Qu.:1.60   3rd Qu.:2.000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :6.20   Max.   :2.000   Max.   :4.0000
##       thal           target
##  Min.   :0.000   Min.   :0.0000
##  1st Qu.:2.000   1st Qu.:0.0000
##  Median :2.000   Median :1.0000
##  Mean   :2.314   Mean   :0.5446
##  3rd Qu.:3.000   3rd Qu.:1.0000
##  Max.   :3.000   Max.   :1.0000
```

```r
dim(heart) # 303 rows, 14 columns.
```

```
## [1] 303  14
```

```r
# train set will be 90% of heart set
# test set will be 10% of heart set
set.seed(1, sample.kind = "Rounding")
test_index <- createDataPartition(y = heart$age, times = 1, p = 0.1, list = FALSE)
train <- heart[-test_index,]
test <- heart[test_index,]
```

Methods and Analysis

Exploratory Data Analysis (EDA)

Now, let us start our EDA on the Cleveland dataset.

```r
train <- train %>% arrange(age) # Arrange rows according to age.
# Take a glimpse at the train set.
head(train)
```

```
##   age sex resting_bps max_bpm chol blood_sugar_120 resting_ecg chest_pain
## 1  29   1         130     202  204               0           0          1
## 2  34   1         118     174  182               0           0          3
## 3  34   0         118     192  210               0           1          1
## 4  35   0         138     182  183               0           1          0
## 5  35   1         122     174  192               0           1          1
## 6  35   1         126     156  282               0           0          0
##   angina st_oldpeak st_slope vessel_count thal target
## 1      0        0.0        2            0    2      1
## 2      0        0.0        2            0    2      1
## 3      0        0.7        2            0    2      1
## 4      0        1.4        2            0    2      1
## 5      0        0.0        2            0    2      1
## 6      1        0.0        2            0    3      0
```

```r
glimpse(train)
```

```
## Observations: 271
## Variables: 14
## $ age             <int> 29, 34, 34, 35, 35, 35, 37, 37, 38, 38, 38, 39...
## $ sex             <int> 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1...
## $ resting_bps     <int> 130, 118, 118, 138, 122, 126, 130, 120, 138, 1...
## $ max_bpm         <int> 202, 174, 192, 182, 174, 156, 187, 170, 173, 1...
## $ chol            <int> 204, 182, 210, 183, 192, 282, 250, 215, 175, 1...
## $ blood_sugar_120 <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ resting_ecg     <int> 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1...
## $ chest_pain      <int> 1, 3, 1, 0, 1, 0, 2, 2, 2, 2, 3, 2, 2, 2, 0, 3...
## $ angina          <int> 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1...
## $ st_oldpeak      <dbl> 0.0, 0.0, 0.7, 1.4, 0.0, 0.0, 3.5, 0.0, 0.0, 0...
## $ st_slope        <int> 2, 2, 2, 2, 2, 2, 0, 2, 2, 2, 1, 2, 2, 1, 1, 2...
## $ vessel_count    <int> 0, 0, 0, 0, 0, 0, 0, 0, 4, 4, 0, 0, 0, 0, 0, 0...
## $ thal            <int> 2, 2, 2, 2, 2, 3, 2, 2, 2, 2, 3, 2, 2, 2, 3, 3...
## $ target          <int> 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1...
```

```
summary(train)
```

```
##       age             sex          resting_bps       max_bpm
##  Min.   :29.00   Min.   :0.00  Min.   : 94.0    Min.   : 71.0
##  1st Qu.:47.50   1st Qu.:0.00  1st Qu.:120.0    1st Qu.:137.5
##  Median :55.00   Median :1.00  Median :130.0    Median :153.0
##  Mean   :54.39   Mean   :0.69  Mean   :131.9    Mean   :150.3
##  3rd Qu.:61.00   3rd Qu.:1.00  3rd Qu.:140.0    3rd Qu.:167.5
##  Max.   :77.00   Max.   :1.00  Max.   :200.0    Max.   :202.0
##       chol       blood_sugar_120   resting_ecg       chest_pain
##  Min.   :126.0   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:211.0   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :243.0   Median :0.0000   Median :1.0000   Median :1.0000
##  Mean   :248.0   Mean   :0.1513   Mean   :0.5203   Mean   :0.9594
##  3rd Qu.:276.5   3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:2.0000
##  Max.   :564.0   Max.   :1.0000   Max.   :2.0000   Max.   :3.0000
##      angina         st_oldpeak       st_slope     vessel_count
##  Min.   :0.0000   Min.   :0.000   Min.   :0.000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:1.000   1st Qu.:0.0000
##  Median :0.0000   Median :0.800   Median :1.000   Median :0.0000
##  Mean   :0.3358   Mean   :1.067   Mean   :1.413   Mean   :0.7565
##  3rd Qu.:1.0000   3rd Qu.:1.800   3rd Qu.:2.000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :6.200   Max.   :2.000   Max.   :4.0000
##       thal           target
##  Min.   :0.000   Min.   :0.0000
##  1st Qu.:2.000   1st Qu.:0.0000
##  Median :2.000   Median :1.0000
##  Mean   :2.336   Mean   :0.5387
##  3rd Qu.:3.000   3rd Qu.:1.0000
##  Max.   :3.000   Max.   :1.0000
```

```
dim(train) # 271 rows, 14 columns.
```

```
## [1] 271  14
```

```
# Total population:
patient_pop <- train %>% nrow()
patient_pop # 271 patients
```

```
## [1] 271
```

```
# Total number of men and women: 1 = male, 0 = female
men <- sum(train$sex)
men    # 187 men
```

```
## [1] 187
```

```
women <- patient_pop - men
women # 84 women
```

```
## [1] 84
```

```
# Number of patients with heart disease:
heart_disease <- sum(train$target)
heart_disease # 146 heart disease patients
```

```
## [1] 146
```

```
# Number of patients without heart disease:
healthy <- patient_pop - heart_disease
healthy       # 125 healthy patients
```

```
## [1] 125
```

```
# Number of men and women with heart disease:
hd_men <- train %>% filter(sex == 1 & target == 1) %>% nrow()
hd_men   # 85 hd men
```

```
## [1] 85
```

```
hd_women <- train %>% filter(sex == 0 & target == 1) %>% nrow()
hd_women # 61 hd women
```

```
## [1] 61
```

```
# Proportion of men and women with heart disease:
hd_men_prop <- hd_men / men
hd_men_prop   # 45.45%
```

```
## [1] 0.4545455
```

```
hd_women_prop <- hd_women / women
hd_women_prop # 72.62%
```

```
## [1] 0.7261905
```

```
# Age:
mean_age <- mean(train$age)
mean_age     # 54 years old
```

```
## [1] 54.38745
```

```
# Mode function:
getmode <- function(x) {
  uniqv <- unique(x)
  uniqv[which.max(tabulate(match(x, uniqv)))]
}
mode_age <- getmode(train$age)
mode_age     # Most patients are 58 years old
```

```
## [1] 58
```

```
# Youngest age:
youngest_age <- min(train$age)
youngest_age # 29 years old
```

```
## [1] 29
```

```
# Oldest age:
oldest_age <- max(train$age)
oldest_age   # 77 years old
```

```
## [1] 77
```

Here is a brief summary of the train dataset:

The train dataset contains 271 entries (out of the 303 entries in the heart dataset) and 14 variables.

- *age*: age of the patient.
- *sex*: gender of the patient (0 = female, 1 = male).
- *resting_bps*: resting blood pressure (in mm Hg).
- *max_bpm*: max heart rate achieved.
- *chol*: serum cholestoral in mg/dl.
- *blood_sugar_120*: resting blood sugar (0 = false, 1 = true).
- *resting_ecg*: resting electrocardiographic results.
- *chest_pain*: chest pain type (0-3).
- *angina*: exercise induced angina (0 = no, 1 = yes).
- *st_oldpeak*: ST depression induced by exercise relative to rest.
- *st_slope*: the slope of the peak exercise ST segment.
- *vessel_count*: number of major vessels (0-3) colored by flourosopy.
- *thal*: 3 = normal; 6 = fixed defect; 7 = reversable defect.
- *target*: presence of heart disease (0 = no, 1 = yes).

Of the 271 patients in the train set, 146 have heart disease and 125 are healthy. Since there are more patients with heart disease than healthy patients there might be a tendency for our models to predict false positives. If patients get falsely marked with heart disease, they may receive unnecessary treatment, resulting in the loss of money, medical supplies, and more importantly, the patient's health. So we must be careful to ensure that specificity is high in our models. Also note that there are more men (187) than women (84). This means that our models run the risk of being more biased towards men than women. The proportion of men with heart disease is 45.45% whereas the proportion of women with heart disease is 72.62%. This suggests that women are at greater risk of developing heart disease than men. The youngest age in the set is 29 and the oldest age in the set is 77, with the mean age being 54. So patients outside of this age range may receive inaccurate diagnosis.

Plots and Visualizations

Before going forward, we need to consider some factors that might influence the onset of heart disease. Here are 6 such potential factors:

1. Age

2. Sex

3. Max BPM

4. Cholesterol

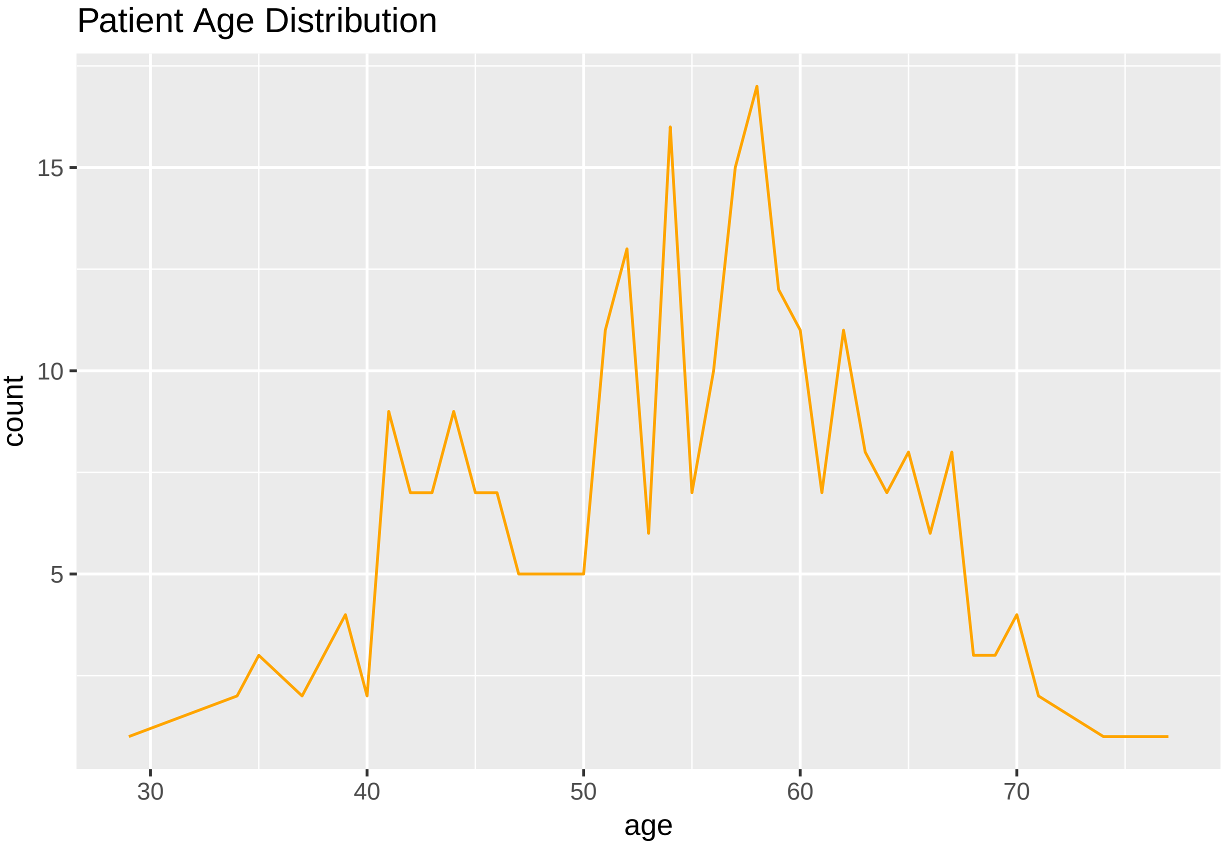
5. Blood sugar

6. Number of major vessels

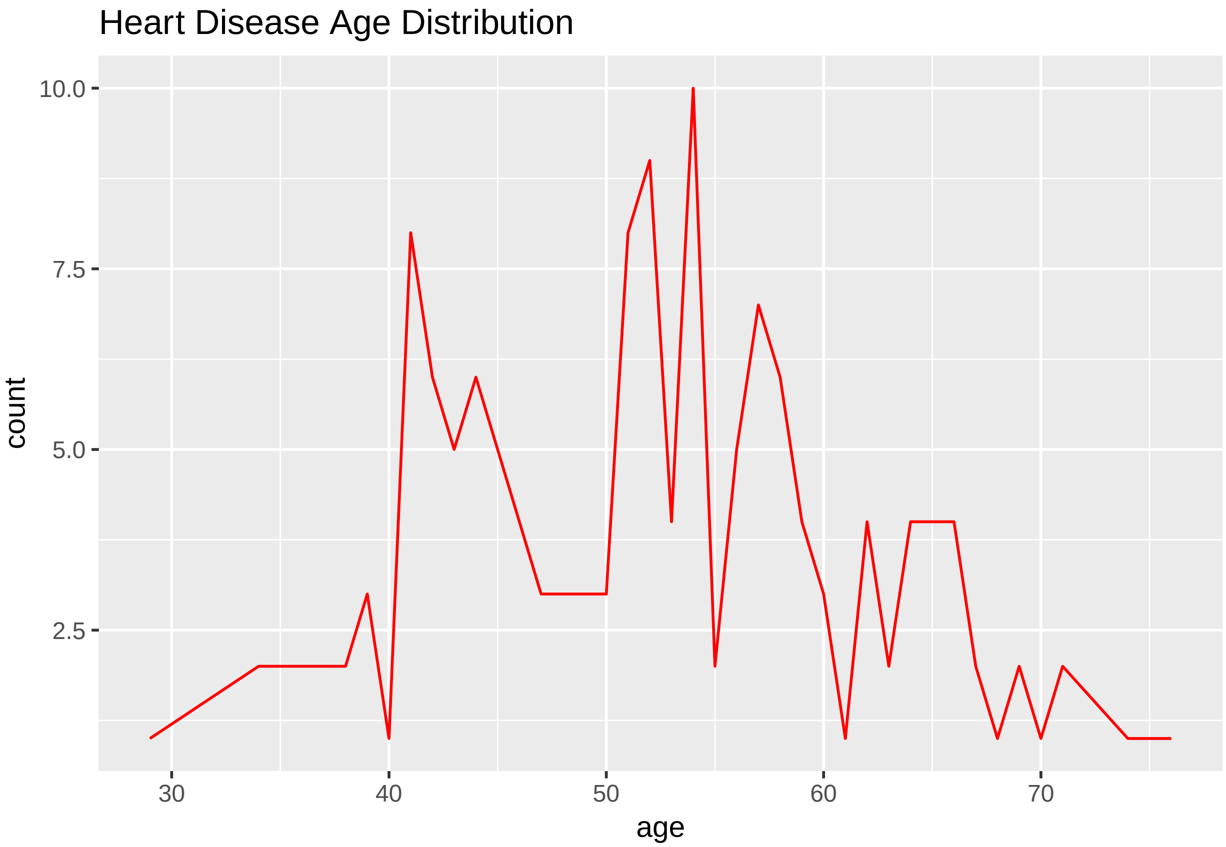
The following plots and visualizations will serve to highlight the 6 above mentioned factors and further aid in our EDA.

1. Age

First, we want to visualize the age Distribution using a histogram.



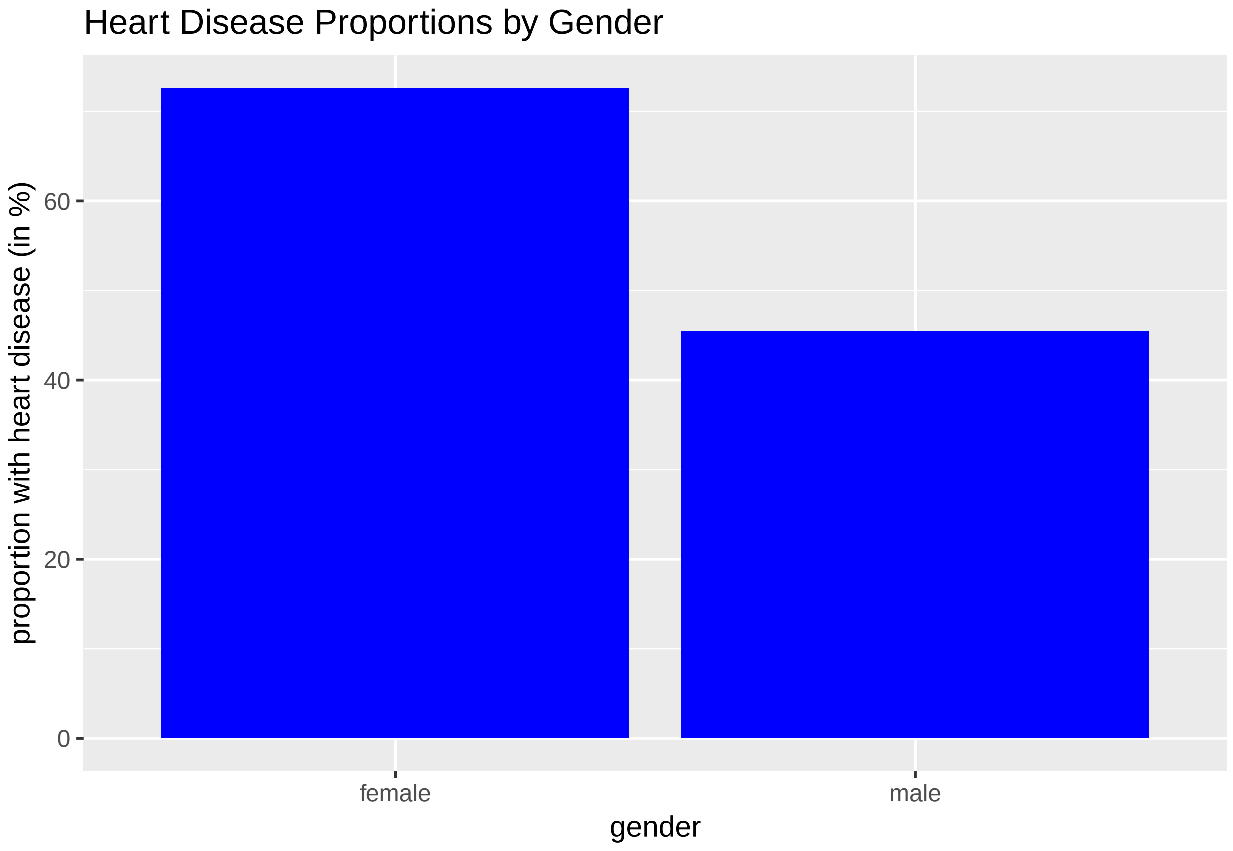
From this histogram, we can see that the ages are normally distributed with a peak at age 58 (mode). However, we want to know which ages are at greater risk of heart disease.



Heart disease seems to affect people of all ages and one can even argue that risk of heart disease increases with age. The reason there are few people over age 65 with heart disease is that old people do not represent the majority of the population. If we were to graph the proportions of a certain age with heart disease, we would find that the disease proportion will increase as age increases. After all, older people are weaker, fragile, and more susceptible to disease.

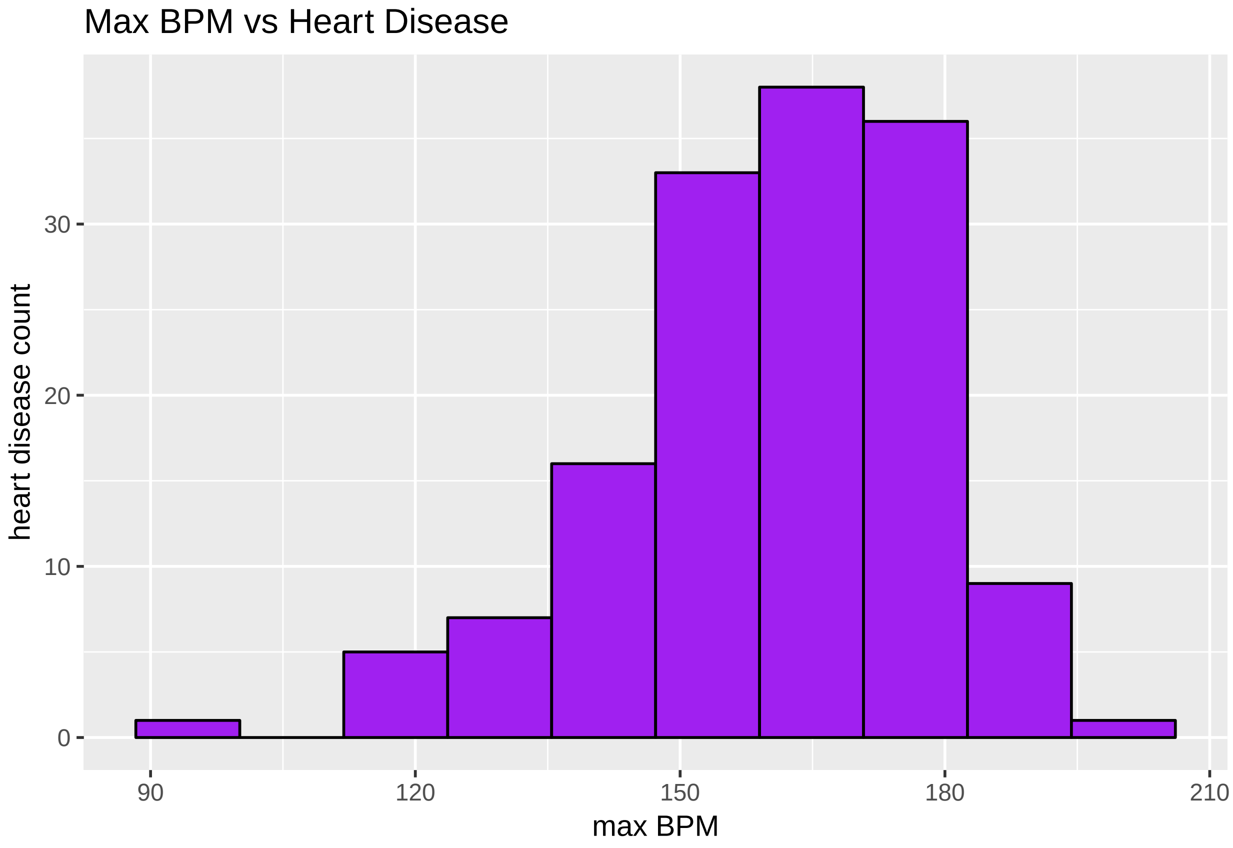
2. Sex

Women are at greater risk of developing heart disease than men. At the same time, they also tend to live longer, and this might play a factor as old people are more likely of developing heart disease, as we mentioned earlier.



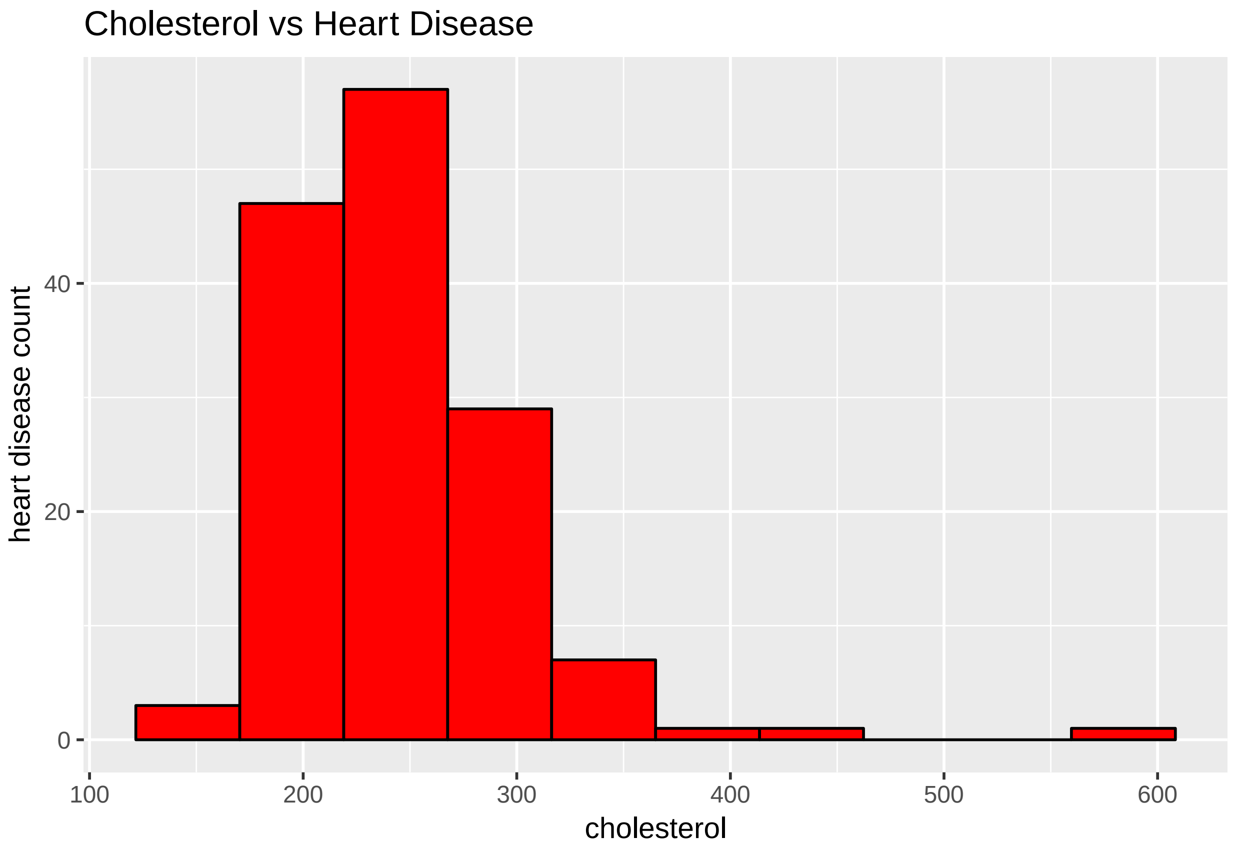
3. Max BPM

The higher the patient's max heart rate, the greater their chance of developing heart disease.



4. Cholesterol

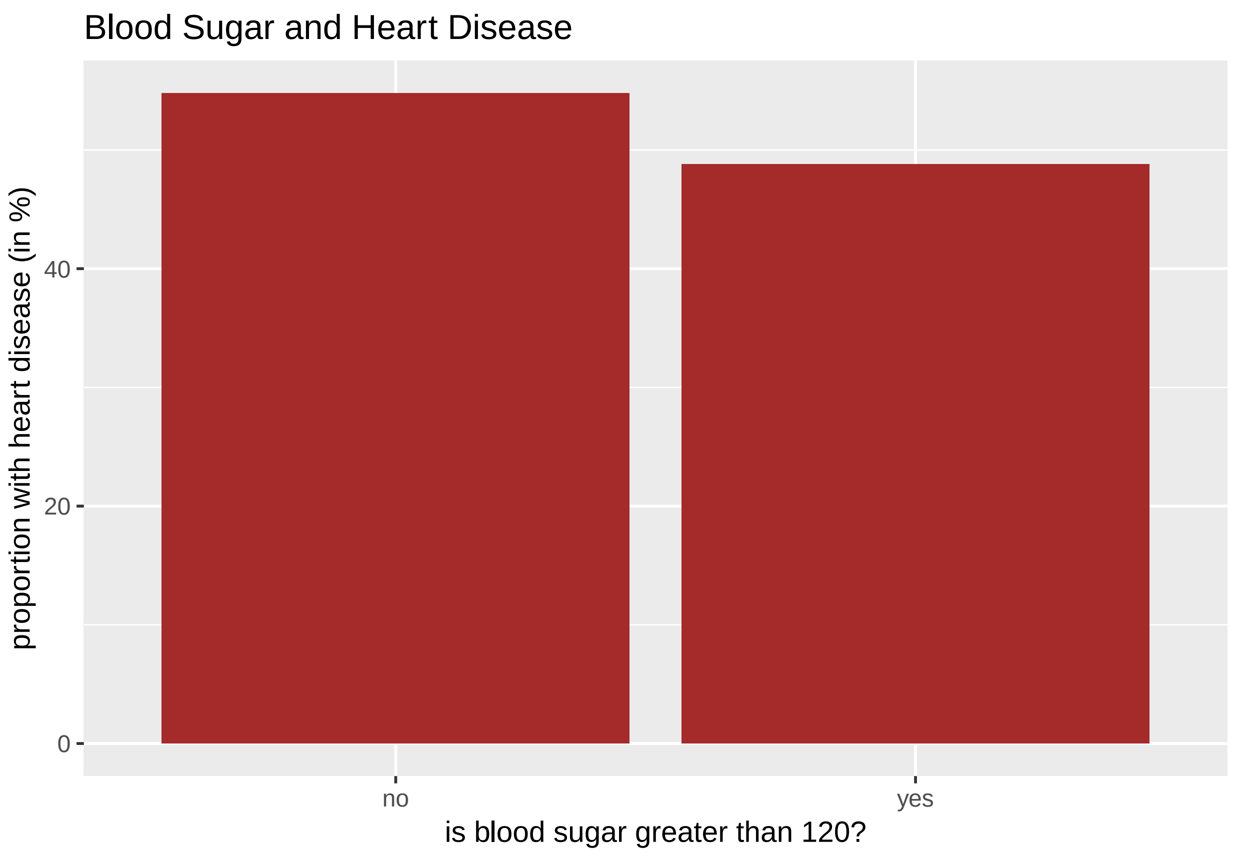
Patients with high cholesterol are at great risk of developing heart disease. However, it is hard to find patients with extremely high cholesterol (over 400 mg/dl), so their numbers are less.



Patients with cholesterol levels between 150 and 350 mg/dl are found to be the majority with heart disease.

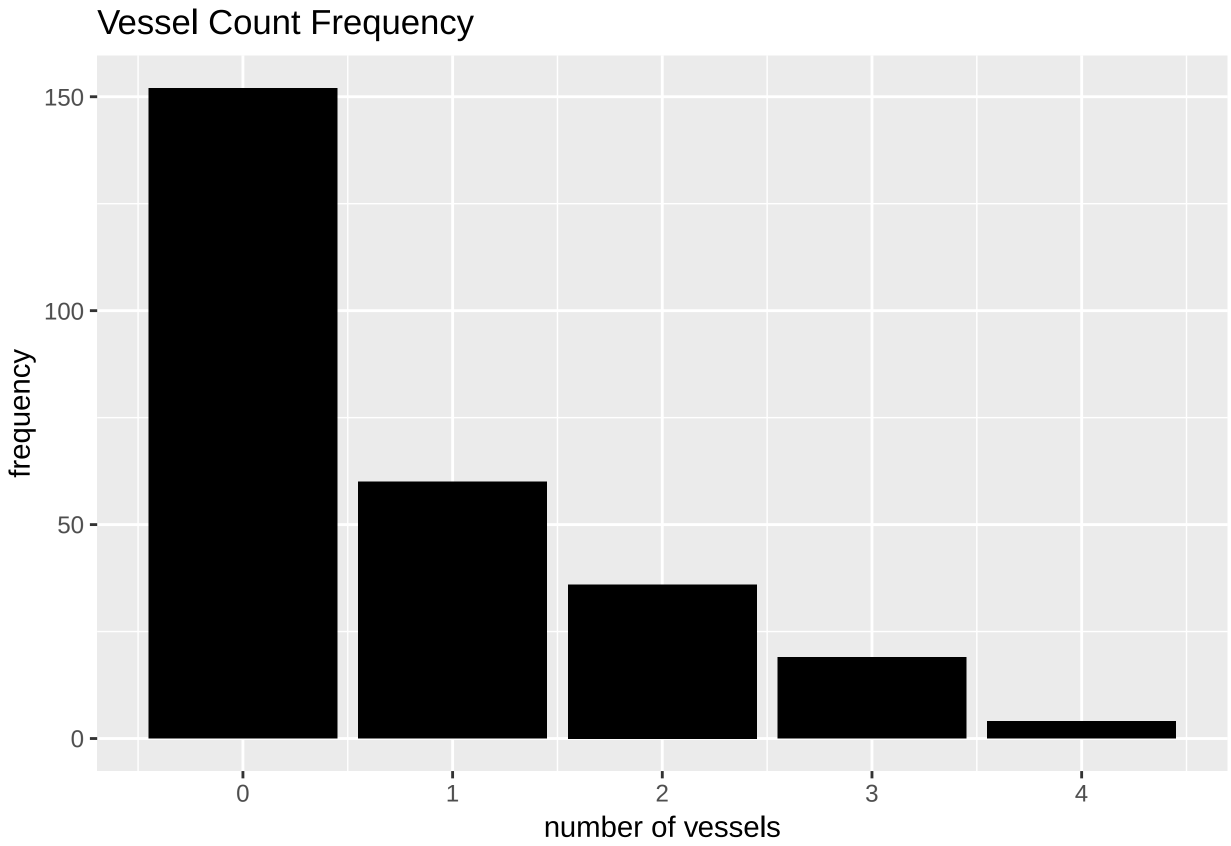
5. Blood Sugar

A fasting blood sugar over 120 mg/dl is usually considered as an indication of diabetes. However, there does not seem to be much correlation between the level of blood sugar and heart disease. Other factors like age and sex serve as better predictors.

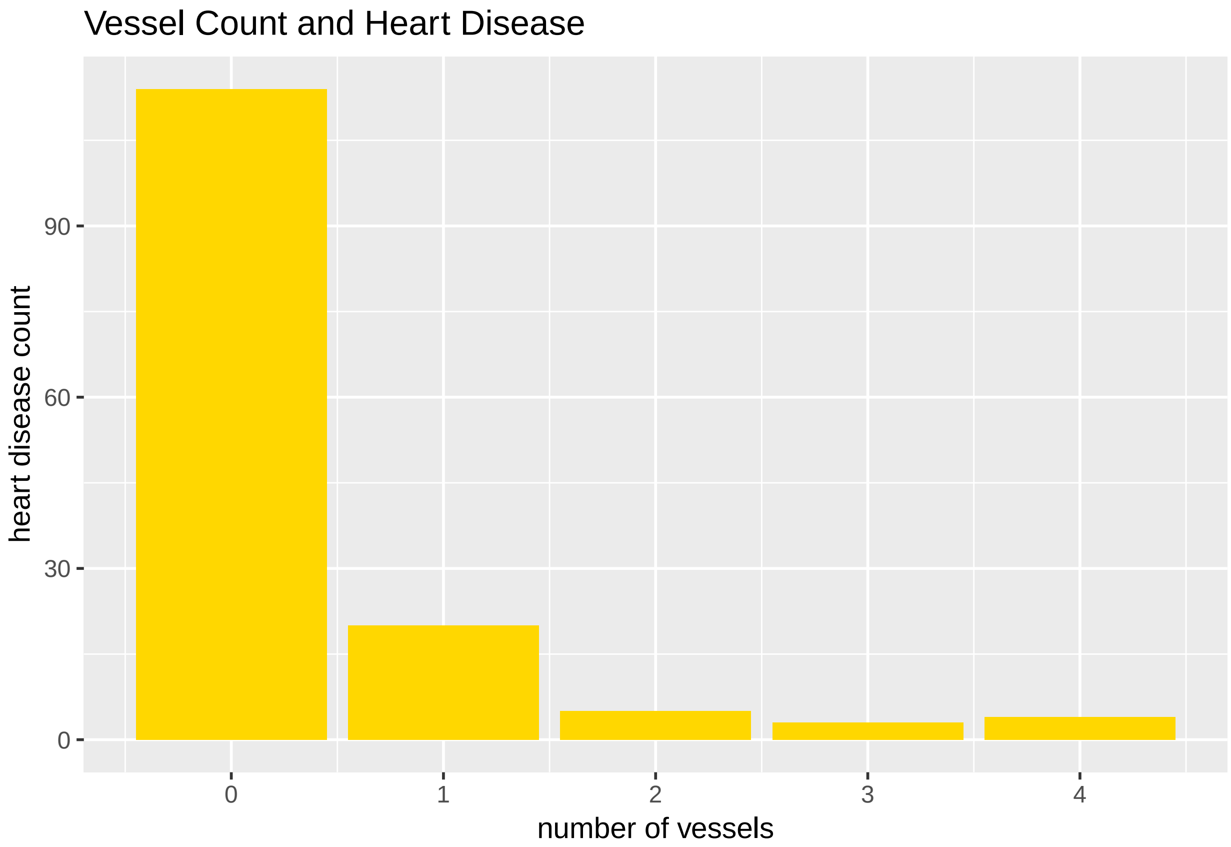


6. Number of major vessels

We want to see which vessel count represents the majority. To do this, we will use a vessel count frequency distribution. There can be anywhere between 0 to 4 major blood vessels colored by flourosopy.



The majority of patients have 0 major vessels colored by flourosopy.



Looking at the vessel count for heart disease patients, we see that as the patients contain more healthy major vessels, the lesser their risk of getting heart disease. At the same time, the frequency of each of the vessel counts seem to have decreased in the same proportions. That is why this frequency bar plot seems very similar in shape to the previous one. Nonetheless, a correlation does seem to exist.

Modelling and Training

Models

For this project, 4 prediction models have been built:

- Model 1: Naive Bayes
- Model 2: Logistic Regression
- Model 3: Support Vector Machine (SVM)
- Model 4: Random Forest

These models will be trained on the train dataset and then tested on the test dataset. Please note that no validation set was created as the original heart dataset is small in size.

Setup for Modelling

Before we start building our models, we need to do a few things:

```r
AUC = list()      # List of AUCs
accuracy = list() # List of model accuracies
# Change some columns in train and test sets to factor:
fac_col <- c(2,3,6,7,9,11,12,13,14)
for(i in fac_col) {
  train[,i] = as.factor(train[,i])
}
for(i in fac_col) {
  test[,i] = as.factor(test[,i])
}
# Remove any troublesome columns: (zero variances between variables)
train[[3]] <- NULL
test[[3]] <- NULL
# Set target level:
levels(train$target) <- make.names(levels(factor(train$target)))
levels(test$target) <- make.names(levels(factor(train$target)))
# Train control:
train_control <- trainControl(method = "repeatedcv",
                              number = 10,
                              repeats = 10,
                              classProbs = TRUE, # Estimate class probabilities.
                              summaryFunction = twoClassSummary)
```

Model 1: Naive Bayes

```r
set.seed(1, sample.kind = "Rounding")
naive_bayes <- train(target ~ ., data = train,
                     method = "nb", family = "binomial")
nb_prediction <- predict(naive_bayes, test)
nb_prediction_prob <- predict(naive_bayes, test, type = "prob")[2]
# Confusion matrix:
nb_conf_mat <- confusionMatrix(nb_prediction, as.factor(test[,"target"]))
nb_conf_mat
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction X0 X1
##         X0  3  1
##         X1 10 18
##
##                Accuracy : 0.6562
##                  95% CI : (0.4681, 0.8143)
##     No Information Rate : 0.5938
##     P-Value [Acc > NIR] : 0.29782
##
##                   Kappa : 0.2
##
##  Mcnemar's Test P-Value : 0.01586
##
##             Sensitivity : 0.23077
##             Specificity : 0.94737
##          Pos Pred Value : 0.75000
##          Neg Pred Value : 0.64286
##              Prevalence : 0.40625
##          Detection Rate : 0.09375
##    Detection Prevalence : 0.12500
##       Balanced Accuracy : 0.58907
##
##        'Positive' Class : X0
##
```

```
#ROC curve:
AUC$naive_bayes <- roc(as.numeric(test$target),
                   as.numeric(as.matrix((nb_prediction_prob))))$auc
AUC
```

```
## $naive_bayes
## Area under the curve: 0.7004
```

```
# Accuracy:
accuracy$naive_bayes <- nb_conf_mat$overall["Accuracy"]
accuracy
```

```
## $naive_bayes
## Accuracy
##  0.65625
```

The accuracy for the naive bayes model is **0.65625**.
The sensitivity is **0.23077**.
The specificity is **0.94737**.
The accuracy is below our set goal of **0.7** and the sensitivity is too low for practical purposes. The specificity, however, is very good and this will help to greatly avoid predicting false positives.

Model 2: Logistic Regression

```
set.seed(1, sample.kind = "Rounding")
log_reg <- train(target ~ ., data = train,
                 method = "glm", family = "binomial")
log_reg_prediction <- predict(log_reg, test)
log_reg_prediction_prob <- predict(log_reg, test, type = "prob")[2]
```

```r
# Confusion matrix:
log_reg_conf_mat <- confusionMatrix(log_reg_prediction, as.factor(test[,"target"]))
log_reg_conf_mat
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction X0 X1
##         X0  6  5
##         X1  7 14
##
##                Accuracy : 0.625
##                  95% CI : (0.4369, 0.789)
##     No Information Rate : 0.5938
##     P-Value [Acc > NIR] : 0.4331
##
##                   Kappa : 0.2033
##
##  Mcnemar's Test P-Value : 0.7728
##
##             Sensitivity : 0.4615
##             Specificity : 0.7368
##          Pos Pred Value : 0.5455
##          Neg Pred Value : 0.6667
##              Prevalence : 0.4062
##          Detection Rate : 0.1875
##    Detection Prevalence : 0.3438
##       Balanced Accuracy : 0.5992
##
##        'Positive' Class : X0
##
```

```r
#ROC curve:
AUC$log_reg <- roc(as.numeric(test$target),
                  as.numeric(as.matrix((log_reg_prediction_prob))))$auc
AUC
```

```
## $naive_bayes
## Area under the curve: 0.7004
##
## $log_reg
## Area under the curve: 0.5951
```

```r
# Accuracy:
accuracy$log_reg <- log_reg_conf_mat$overall["Accuracy"]
accuracy
```

```
## $naive_bayes
## Accuracy
##  0.65625
##
## $log_reg
## Accuracy
##    0.625
```

The accuracy for the logistic regression model is **0.625**.
The sensitivity is **0.4615**.
The specificity is **0.7368**.
The accuracy is below our set goal of **0.7** and even though the sensitivity is better than that of model 1, it is still too low for practical purposes. By increasing the sensitivity, we have to pay the price of decreased specificity.

Model 3: Support Vector Machine (SVM)

```r
set.seed(1, sample.kind = "Rounding")
svm <- train(target ~ ., data = train,
             method = "svmLinear",
             preProcess = c("center", "scale"),
             trControl = train_control,
             family = "binomial",
             tuneLength = 8,
             metric = "ROC")
svm_prediction <- predict(svm, test)
svm_prediction_prob <- predict(svm, test, type="prob")[2]
# Confusion matrix:
svm_conf_mat <- confusionMatrix(svm_prediction, test[,"target"])
svm_conf_mat
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction X0 X1
##         X0  7  4
##         X1  6 15
##
##                Accuracy : 0.6875
##                  95% CI : (0.4999, 0.8388)
##     No Information Rate : 0.5938
##     P-Value [Acc > NIR] : 0.1848
##
##                   Kappa : 0.3361
##
##  Mcnemar's Test P-Value : 0.7518
##
##             Sensitivity : 0.5385
##             Specificity : 0.7895
##          Pos Pred Value : 0.6364
##          Neg Pred Value : 0.7143
##              Prevalence : 0.4062
##          Detection Rate : 0.2188
##    Detection Prevalence : 0.3438
##       Balanced Accuracy : 0.6640
##
##        'Positive' Class : X0
##
```

```r
#ROC curve:
AUC$svm <- roc(as.numeric(test$target),
               as.numeric(as.matrix((svm_prediction_prob))))$auc
AUC
```

```
## $naive_bayes
## Area under the curve: 0.7004
##
## $log_reg
## Area under the curve: 0.5951
##
## $svm
## Area under the curve: 0.7085
```

```
# Accuracy:
accuracy$svm <- svm_conf_mat$overall["Accuracy"]
accuracy
```

```
## $naive_bayes
## Accuracy
##  0.65625
##
## $log_reg
## Accuracy
##    0.625
##
## $svm
## Accuracy
##   0.6875
```

The accuracy for the SVM model is **0.6875**.
The sensitivity is **0.5385**.
The specificity is **0.7895**.
The accuracy is very close to our set goal of **0.7** but not quite there yet. This time, both the sensitivity and specificity have increased from the previous model, thus increasing true positives and decreasing false positives.

Model 4: Random Forest

```
set.seed(1, sample.kind = "Rounding")
rf <- train(target ~ ., data = train,
            method = "rf",
            ntree = 500,
            preProcess = c("center", "scale"),
            trControl = train_control,
            family = "binomial",
            tuneLength = 8,
            metric = "ROC")
# Alternative method:
#rf <- randomForest(target ~ .,data = train, ntree = 500)
rf_prediction <- predict(rf, test)
rf_prediction_prob = predict(rf, test, type="prob")[2]
# Confusion matrix:
rf_conf_mat <- confusionMatrix(rf_prediction, test[,"target"])
rf_conf_mat
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction X0 X1
##         X0  7  3
##         X1  6 16
##
##                Accuracy : 0.7188
##                  95% CI : (0.5325, 0.8625)
##     No Information Rate : 0.5938
##     P-Value [Acc > NIR] : 0.1022
##
##                   Kappa : 0.395
##
##  Mcnemar's Test P-Value : 0.5050
##
```

```
##             Sensitivity : 0.5385
##             Specificity : 0.8421
##          Pos Pred Value : 0.7000
##          Neg Pred Value : 0.7273
##              Prevalence : 0.4062
##          Detection Rate : 0.2188
##    Detection Prevalence : 0.3125
##       Balanced Accuracy : 0.6903
##
##        'Positive' Class : X0
##
```

```
#ROC curve:
AUC$rf <- roc(as.numeric(test$target),
              as.numeric(as.matrix((rf_prediction_prob))))$auc
AUC
```

```
## $naive_bayes
## Area under the curve: 0.7004
##
## $log_reg
## Area under the curve: 0.5951
##
## $svm
## Area under the curve: 0.7085
##
## $rf
## Area under the curve: 0.7247
```

```
# Accuracy:
accuracy$rf <- rf_conf_mat$overall["Accuracy"]
accuracy
```

```
## $naive_bayes
## Accuracy
##  0.65625
##
## $log_reg
## Accuracy
##    0.625
##
## $svm
## Accuracy
##   0.6875
##
## $rf
## Accuracy
##  0.71875
```

The accuracy for the random forest model is **0.7188**.
The sensitivity is **0.5385**.
The specificity is **0.8421**.
This time, the accuracy has passed our set goal of **0.7**. Also, we were able to increase the specificity with causing any decrease in sensitivity.

Results

Model 1 Accuracy:

```
## Accuracy
##  0.65625
```

Model 2 Accuracy:

```
## Accuracy
##    0.625
```

Model 3 Accuracy:

```
## Accuracy
##   0.6875
```

Model 4 Accuracy:

```
## Accuracy
##  0.71875
```

Out of the 4 prediction models we have built for this project, Model 4, which implemented a random forest, managed to achieve an accuracy greater than the target accuracy of **0.7**. The accuracy achieved by that model is **0.71875**. Therefore, our goal has been achieved.

Conclusion

For this project, we built and used 4 machine learning models to predict heart disease in patients. We started off with a relatively simple naive bayes model, then a logistic regression model, then built an SVM model, and finally achieved an accuracy greater than **0.7** with our random forest model. The best of the models is the random forest model and the worst is the logistic regression model.

Based on what we have observed, it can be concluded that certain factors like *age* and *sex* alone have enough predictive power to determine the onset of heart disease in patients with a reasonable degree of accuracy.

In the future, we could try implementing more models like FLD, boosted trees, least squares, AdaBoost, ensemble, etc. The goal would be to create models which address some of the shortcomings of our current models and hopefully achieve an accuracy greater than or equal to **0.85**.