Heart Disease predicting model

Davide Mori

26/9/2019

HarvardX: PH125.9

Data Science Professional Certificate: Capstone - "Choose Your Own Project Submission"

You can find this project on Github Website at: https://github.com/davidemori/Cardiac-Disease

Introduction

Of the 56.9 million deaths worldwide in 2016, more than half (54%) were due to the top 10 causes. Ischaemic heart disease and stroke are the world's biggest killers, accounting for a combined 15.2 million deaths in 2016. These diseases have remained the leading causes of death globally in the last 15 years(World Health Organization, 2019). The aim of this project is to analyze the causes and try to find a model to predict, on the basis of some parameters, the occurence of an heart disease.

Many datasets about Medical Sciences, and in particular about heart diseases, are free downloadable on the "WEB". For the following project we have used a subset of the "Cleveland Heart Disease Dataset" from the UCI Archives, that can be found at the following link: http://archive.ics.uci.edu/ml/datasets/heart+disease .

On this dataset we'll try to perform some exploratory analyses, and test a total of six models (5 + 1 ensemble model) to find the best fit for prediction of the heart disease in an adult population.

Methods

cleaning and process the Dataset

The first goal of our cleaning and tidying process is to download the dataset and define the names of the columns. In fact, despite the dataset is provided with a partially tidy format, the names of the variables are missing form the principal file, and hence we have to extrapolate them from another document. At the end of the process we'll have the following variables:

- Age: age in years
- Sex: (1 = male; 0 = female)
- Chest.Pain: chest pain type – Value 1: typical angina – Value 2: atypical angina – Value 3: non-anginal pain Value 4: Asymptomatic
- BP.Rest: resting blood pressure (in mm Hg on admission to the hospital)
- Chol.Liv: serum cholestoral in mg/dl
- Fast.Blood.Sugar: (fasting blood sugar $>$ 120 mg/dl) (1 = true; 0 = false)
- ECG.Rest: resting electrocardiographic results: Value 0: normal
  Value 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of $>$ 0.05 mV) Value 2: showing probable or definite left ventricular hypertrophy by Estes' criteria
- HR.Max: maximum heart rate achieved
- Angina.post.Exercise: exercise induced angina (1 = yes; 0 = no)
- OldPeak.ST: ST depression induced by exercise relative to rest

- Slope.ST: the slope of the peak exercise ST segment
- Vessels.Fluo:number of major vessels (0-3) colored by flourosopy
- Defect:3 = normal; 6 = fixed defect; 7 = reversable defect
- Disease: disease in the patient. It is integer valued from 0 (no presence) to 4. Experiments with the Cleveland database have concentrated on simply attempting to distinguish presence (values 1,2,3,4) from absence (value 0).

```r
#Load necessary libraries
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: tidyverse

## -- Attaching packages --------------------------------------- tidyverse 1.2.1 --

## v ggplot2 3.2.1     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: caret

## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```r
if(!require(data.table)) install.packages("data.table", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: data.table

##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last

## The following object is masked from 'package:purrr':
##
##     transpose
```

```r
if(!require(foreign)) install.packages("foreign")
```

```
## Loading required package: foreign
```

```r
if(!require(GGally))install.packages("GGally")
```

```
## Loading required package: GGally
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
##
## Attaching package: 'GGally'
```

```
## The following object is masked from 'package:dplyr':
##
##     nasa
```

```r
#Downloading and checking dataset
data<-read.table(
  "https://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease/processed.cleveland.data",
  sep=",")

#adding column names
colnames(data) <- c("Age",
                    "Sex",
                    "Chest.Pain",
                    "BP.Rest",
                    "Chol.Liv",
                    "Fast.Blood.Sugar",
                    "ECG.Rest",
                    "HR.Max",
                    "Angina.post.Exercise",
                    "OldPeak.ST",
                    "Slope.ST",
                    "Vessels.Fluo",
                    "Defect",
                    "Disease")
```

After the first data adjustment we have to check the structure of oure datas and check for any NAs (missing values):

```r
#Check for changes after renaming columns, and check for NAs
str(data)
```

```
## 'data.frame':    303 obs. of  14 variables:
##  $ Age                 : num  63 67 67 37 41 56 62 57 63 53 ...
##  $ Sex                 : num  1 1 1 1 0 1 0 0 1 1 ...
##  $ Chest.Pain          : num  1 4 4 3 2 2 4 4 4 4 ...
##  $ BP.Rest             : num  145 160 120 130 130 120 140 120 130 140 ...
##  $ Chol.Liv            : num  233 286 229 250 204 236 268 354 254 203 ...
##  $ Fast.Blood.Sugar    : num  1 0 0 0 0 0 0 0 0 1 ...
##  $ ECG.Rest            : num  2 2 2 0 2 0 2 0 2 2 ...
##  $ HR.Max              : num  150 108 129 187 172 178 160 163 147 155 ...
##  $ Angina.post.Exercise: num  0 1 1 0 0 0 0 1 0 1 ...
##  $ OldPeak.ST          : num  2.3 1.5 2.6 3.5 1.4 0.8 3.6 0.6 1.4 3.1 ...
##  $ Slope.ST            : num  3 2 2 3 1 1 3 1 2 3 ...
##  $ Vessels.Fluo        : Factor w/ 5 levels "?","0.0","1.0",..: 2 5 4 2 2 2 4 2 3 2 ...
##  $ Defect              : Factor w/ 4 levels "?","3.0","6.0",..: 3 2 4 2 2 2 2 2 4 4 ...
##  $ Disease             : int  0 2 1 0 0 0 3 0 2 1 ...
```

```
#Check for NAs
apply(is.na(data), 2,which)
```

```
## integer(0)
```

Despite after check there wasn't any NAs, we have seen that some variables was coded by the question mark "?", so few entries are missed. For thi reason we procede to remove them all by filtering dataset.

```
data <- data %>% filter_all(all_vars(.!="?"))
```

The dataset contains the column `disease`, that is represented by a range of five values 0 to 4, where 0 means "No disease" and the other 4 values are the level of disease progression. Since also a a value of 1 indicates heart disease, we'll convert it to a binary variable where 0 is kept if no disease are present, and 1 otherwise. Additionally, some data loaded as numeric are converted to factor for better analysis.

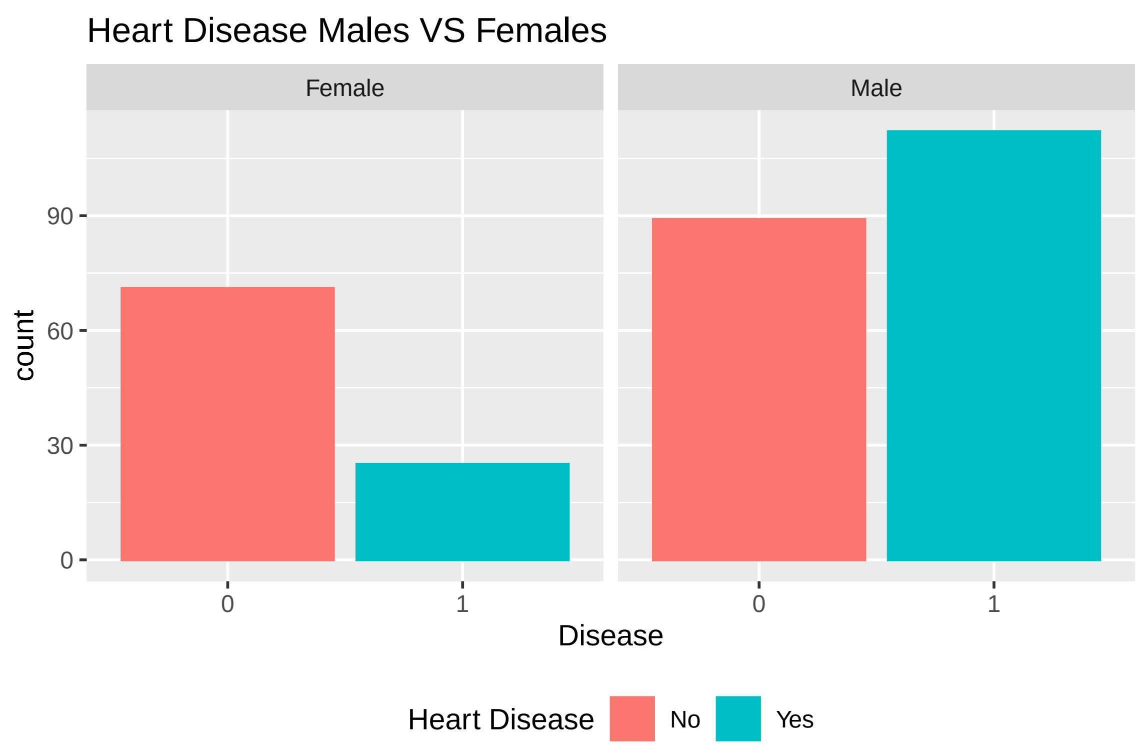
```
data$Vessels.Fluo<-as.numeric(data$Vessels.Fluo) -2
data$Defect<-factor(data$Defect)
data$Sex<-factor(data$Sex)
data$Chest.Pain<-factor(data$Chest.Pain)
data$Fast.Blood.Sugar<-as.factor(data$Fast.Blood.Sugar)
data$ECG.Rest<-as.factor(data$ECG.Rest)
data$Angina.post.Exercise<-as.factor(data$Angina.post.Exercise)
data$Slope.ST<-as.factor(data$Slope.ST)


data<-data%>%mutate(Disease=as.factor(ifelse(Disease==0,0,1)))
```

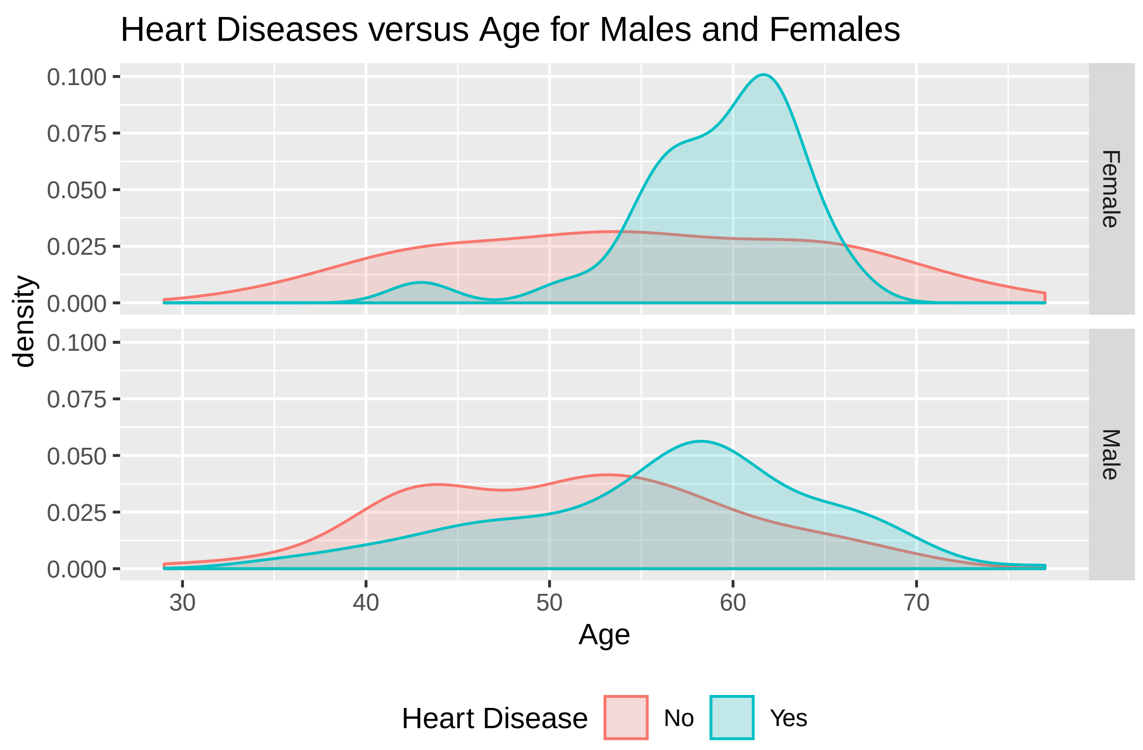
Once we have finally cleaned our dataset, we are ready to starting the explorative analysis.

Data Exploration and testing correlations

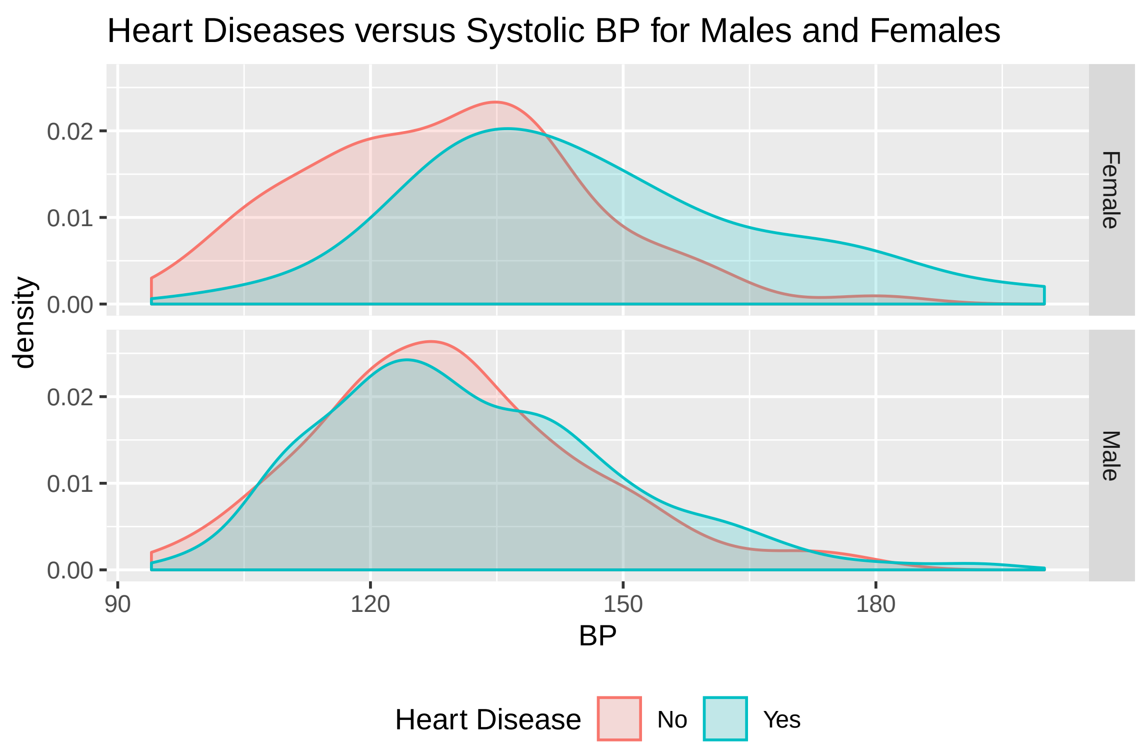
To check our data, will perform a series of density plot and bar plot to better exploring the nature of our independent variables compared to the dependent one (`disease`). Since epidemiologically it is known that the male and the female have a different distribution of heart diseases, we will plot all the variables comparing male and female sex.



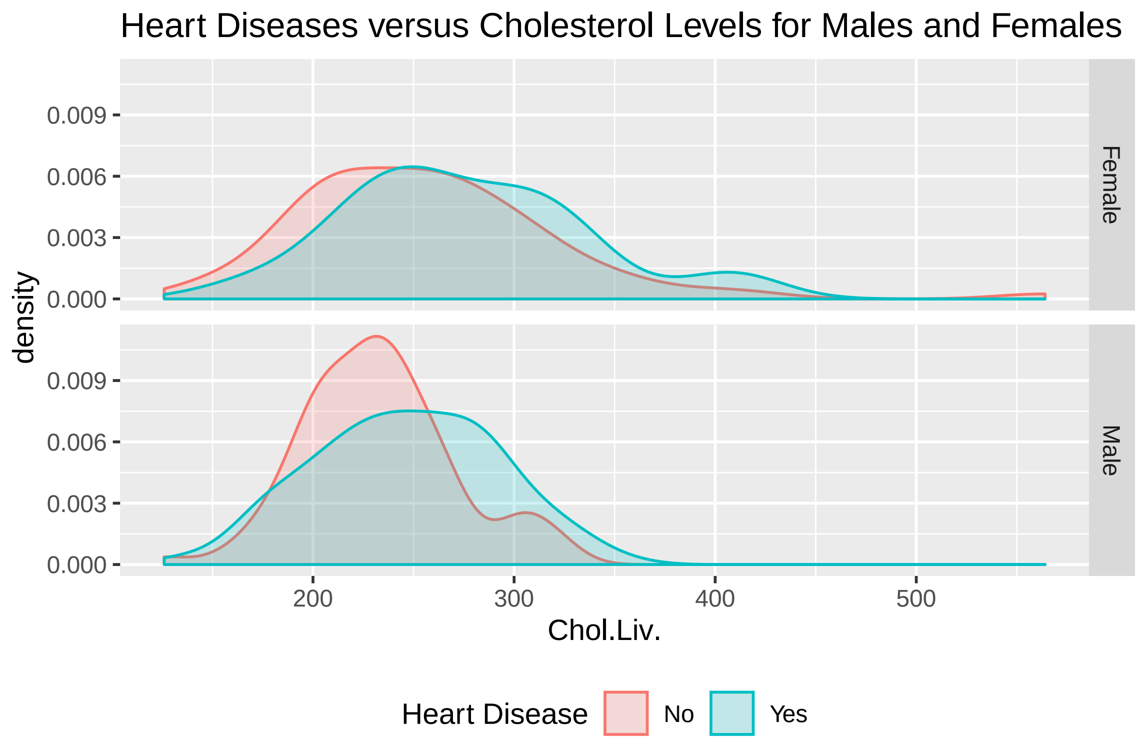
How is possible to see from the first barplot, males have a bigger incidence and a bigger prevalence of heart disease than females.

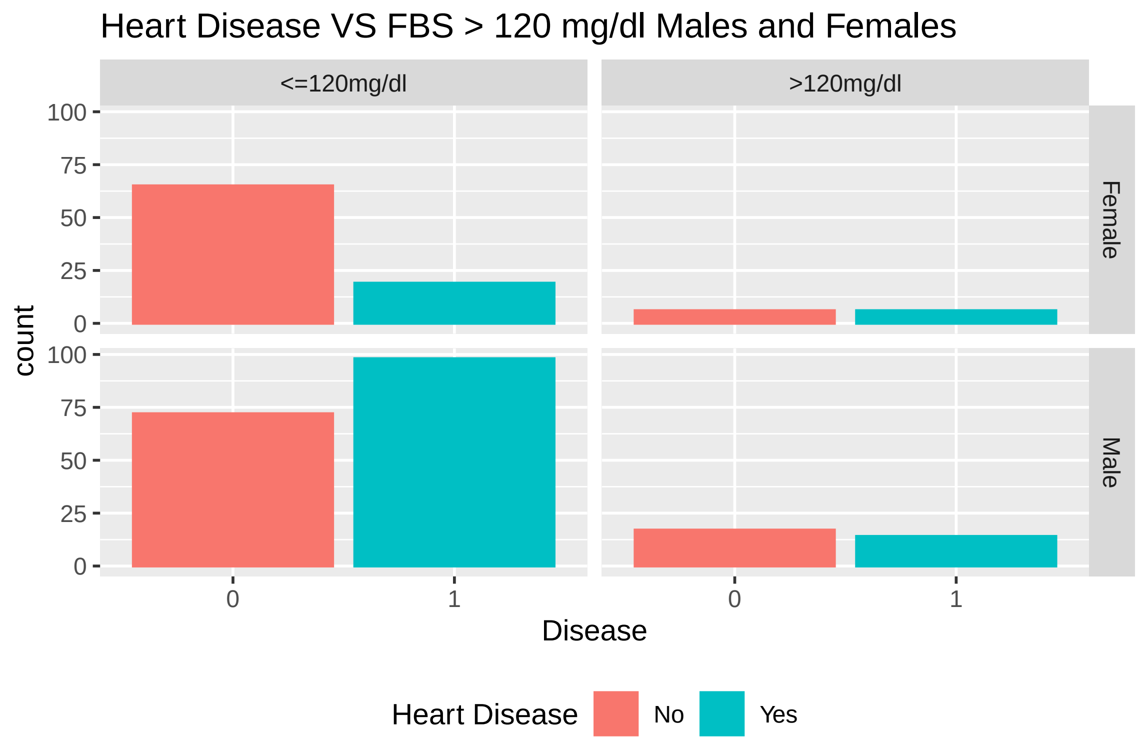


Not only the sex represent a discriminant variable for the occurrence of heart disease, age too has an important effect on it. Males and females, as is shown in the previous density plot, are both suffer a greater incidence of heart disease between the age of 50 and 70 years old.

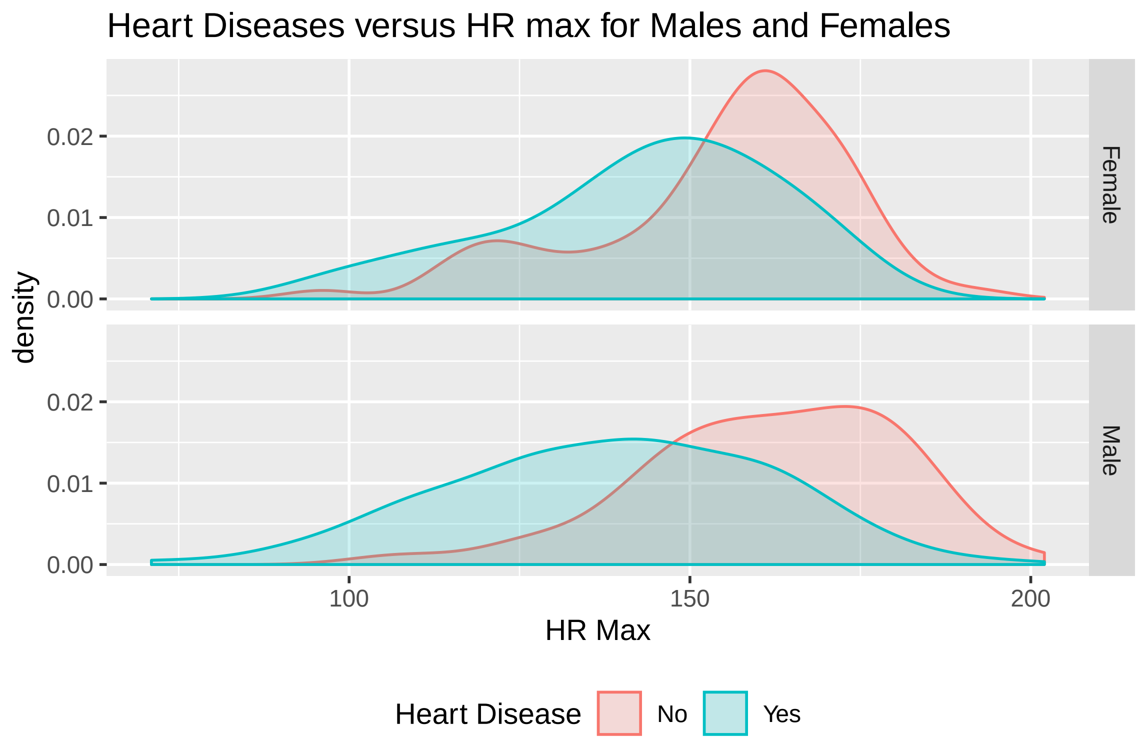


If we consider the systolic BP we can observe that there aren't great difference in terms of presence of disease, just in case of female sex, we can note a slightly increasing of incidence with a value of 150 mmHg or greater.

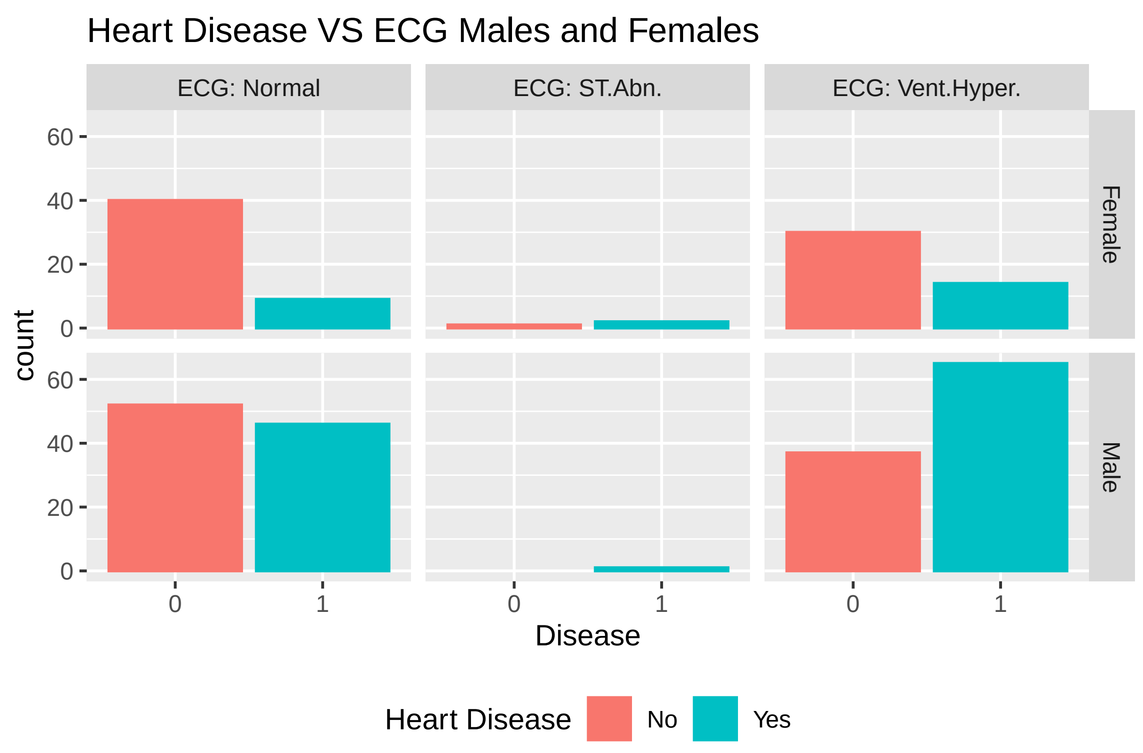




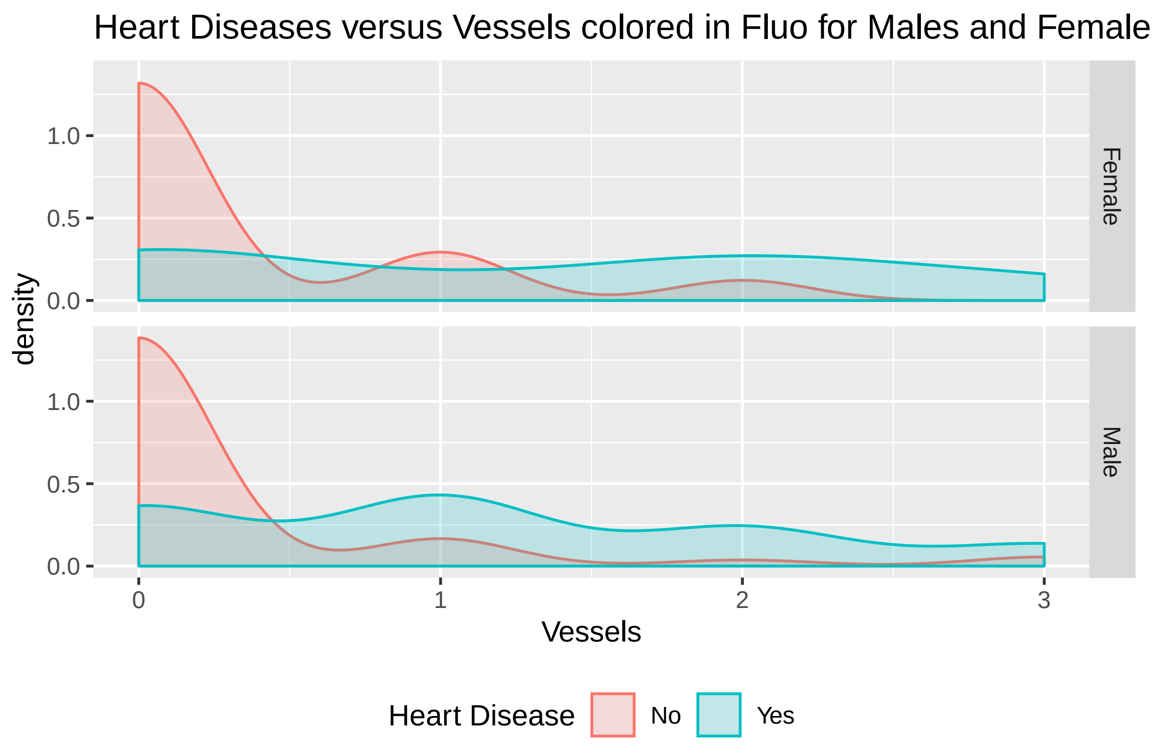
Cholesterol levels and Fast blood sugar are also seeming a non-discriminating variables, although we can observe a little modification in terms of incidence with values of Cholesterol greater than 250.



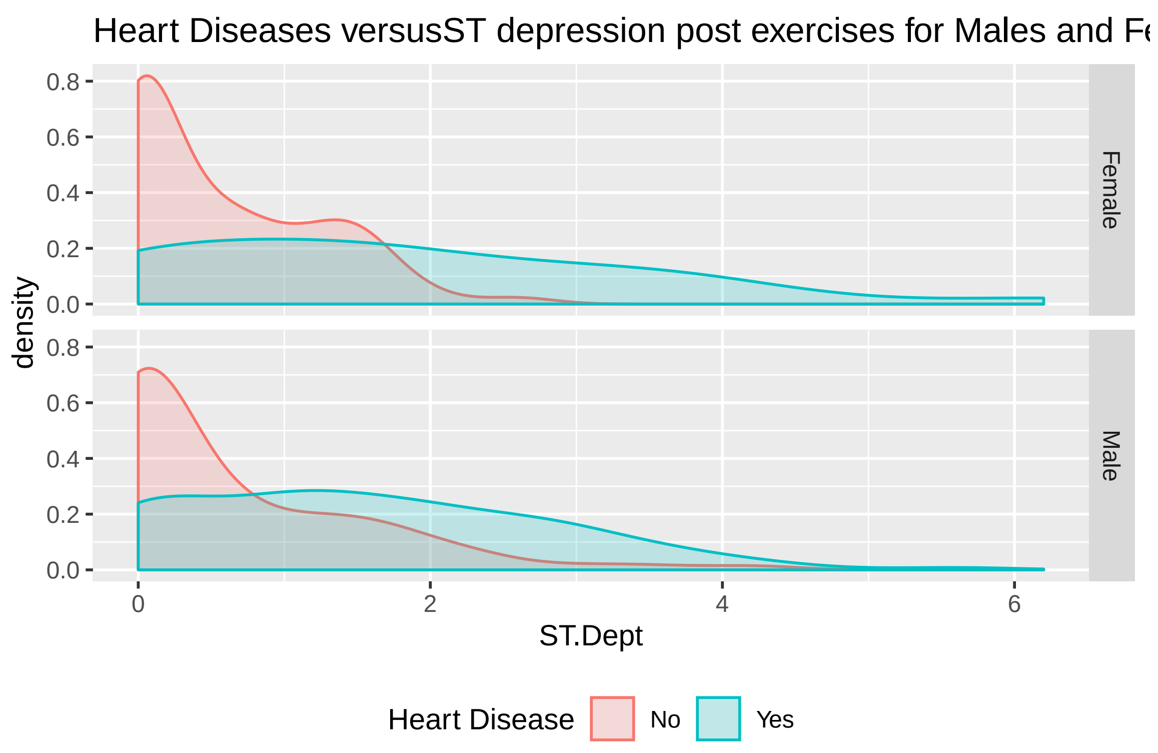
Paradoxically, as is shown in the density plot, a lower heart rate would seem to be related to an increase in the incidence of the disease.



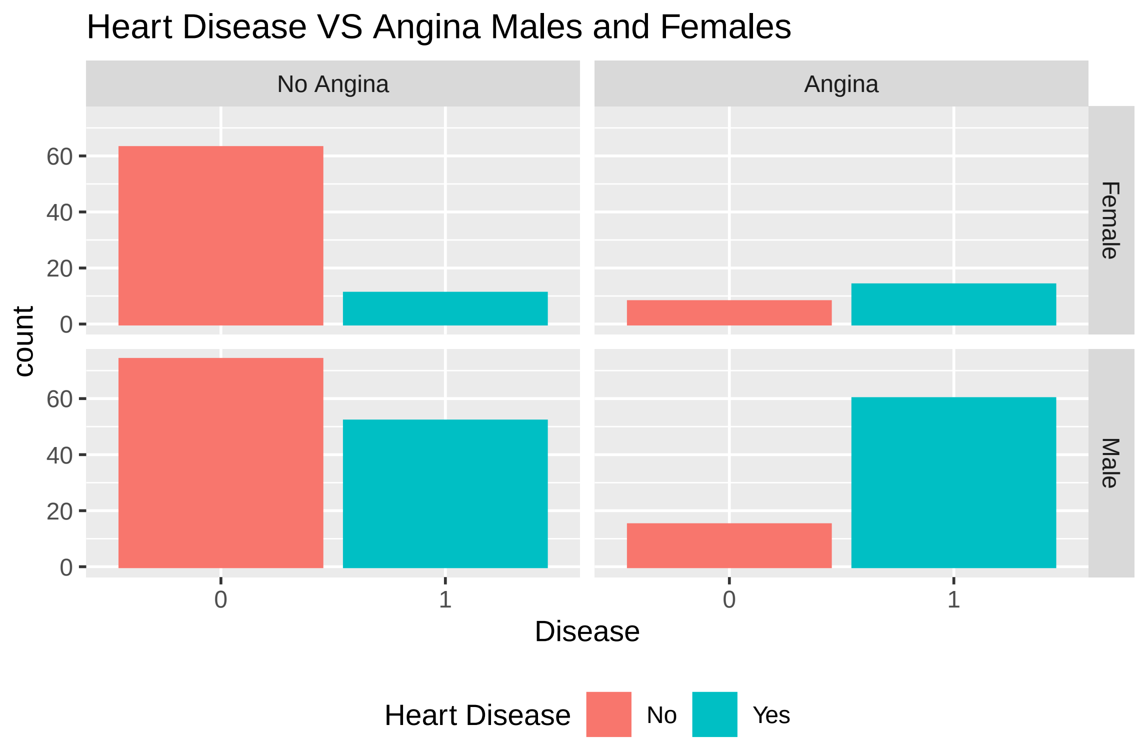
ECG seems to provide insights just for evident anomalies of cardiac rhythm.



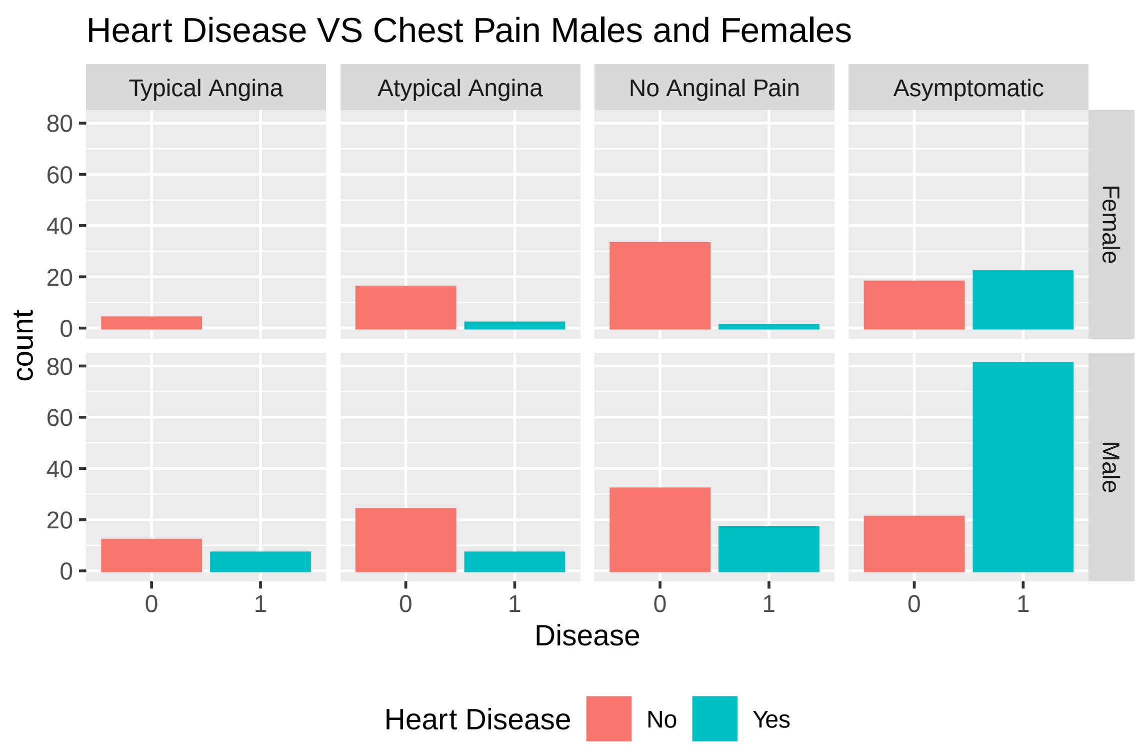
A very important information is provided by the cardiac fluor angiography.In Fact, with one or more vessels involved the incidence of heart disease increase significatively.



Heart Diseases versusST depression post exercises for Males and Fe
density
0.8
0.6
0.4
0.2
0.0
Female
Male
0
2
4
6
ST.Dept
Heart Disease
No
Yes

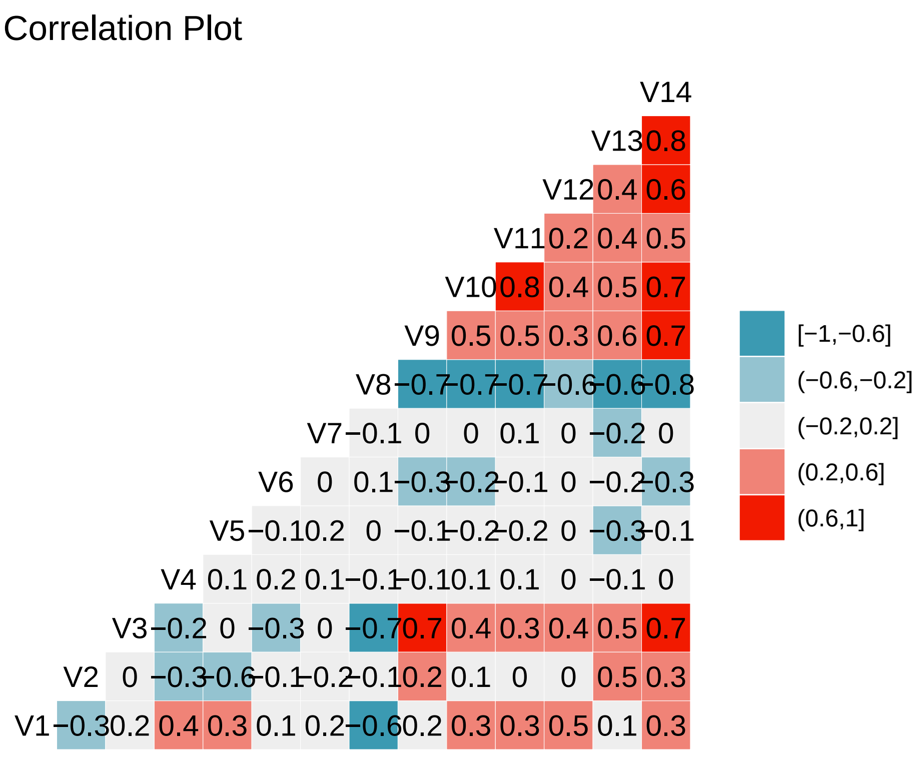


Heart Disease VS Angina Males and Females
No Angina
Angina
count
60
40
20
0
Female
Male
0
1
Disease
Heart Disease
No
Yes



While ST depression during an ECG, and an Angina pain after exercise would seem to be related to the presence of heart disease, the data about chest pain shows a counterintuitive condition, where typical angina seems to be a worst predicting element than asymptomatic condition.

Once reached this point of the analysis it is advisable to evaluate the correlations existing between the variables examined.



As we can observe our variables are not very intercorrelated, and so we can proceed with our analysis without the risk of use two or more unreliable predictors.

Creating the Training and Testing Sets

In order to predict heart disease in patients, we must separate the dataset into a training and a testing set, each containing different observations. 20% of the dataset is thus assigned to the testing set.

```
set.seed(1, sample.kind = "Rounding") #use set.seed(1) if use R version < 3.6
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```
test_index<-createDataPartition(data$Disease,times = 1, p=0.2, list=F) #Indexing for test set
test_set<-data[test_index,] #creating test set
train_set<-data[-test_index,] #crating training set
```

We are now ready to test some machine learning algorithms to create a final prediction model for our dataset. For a better develop and train our models we will provide to set a training control with the `caret` package, using a 10 fold cross validation during all over the tests.

```
# Define train control for k-fold (10-fold here) cross validation
fitControl <- trainControl(method = "cv",number = 10, savePredictions = TRUE)
```

Model 1: K-Nearest Neighbors

A first attempt was made using a k-nearest neighbors algorithm, or 'KNN'. The first step to optimize this model was to use the `tuneGrid` function in order to retrieve the highest accuracy from a range of k values (the number of "neighbors" to be considered for each datapoint).

```
#starting with a KNN Model
set.seed(1, sample.kind = "Rounding") #use set.seed(1) if use R version < 3.6
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```
fit_KNN<-train(Disease~.,
               data= train_set,
               method="knn",
               tuneGrid=data.frame(k=seq(1,30,1)),
               trControl=fitControl) #fit model and try some k's
pre_KNN<-predict(fit_KNN,test_set) #predict results
confusionMatrix(pre_KNN,test_set$Disease) #make a confusion matrix to calculate Accuracy of the model
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 23 13
##          1  9 15
```

```
##
##                Accuracy : 0.6333
##                  95% CI : (0.499, 0.7541)
##     No Information Rate : 0.5333
##     P-Value [Acc > NIR] : 0.0766
##
##                   Kappa : 0.2568
##
##  Mcnemar's Test P-Value : 0.5224
##
##             Sensitivity : 0.7188
##             Specificity : 0.5357
##          Pos Pred Value : 0.6389
##          Neg Pred Value : 0.6250
##              Prevalence : 0.5333
##          Detection Rate : 0.3833
##    Detection Prevalence : 0.6000
##       Balanced Accuracy : 0.6272
##
##        'Positive' Class : 0
##
```

```
#saving parameters and print the results
Accuracy_KNN <- confusionMatrix(pre_KNN,test_set$Disease)$overall["Accuracy"]
Sensitivity_KNN <- confusionMatrix(pre_KNN,test_set$Disease)$byClass["Sensitivity"]
Specificity_KNN <- confusionMatrix(pre_KNN,test_set$Disease)$byClass["Specificity"]
Bal.Accuracy_KNN <- confusionMatrix(pre_KNN,test_set$Disease)$byClass["Balanced Accuracy"]

results <- tribble(
    ~Method, ~Accuracy, ~Sensitivity,  ~Specificity, ~Bal.Accuracy,
    "KNN", Accuracy_KNN,  Sensitivity_KNN, Specificity_KNN,Bal.Accuracy_KNN)
results
```

```
## # A tibble: 1 x 5
##   Method Accuracy Sensitivity Specificity Bal.Accuracy
##   <chr>     <dbl>       <dbl>       <dbl>        <dbl>
## 1 KNN       0.633       0.719       0.536        0.627
```

At a first look the accuracy of this model is not very high (0.63) and in particular his Balanced Accuracy is only 0.627, all this things suggest us to continue our analyses.

Model 2: Adaptive boosting Model

The second model that we wanna try is the "Adaboost" model. Also this time we set a seed equal to 1 ande proceed to train the model on the training set. It will take few minutes to run, but crossing our fingers probably we will see an improvement.

```
#Adaboost Classification trees model
set.seed(1, sample.kind = "Rounding") #use set.seed(1) if use R version < 3.6
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```r
fit_ada<-train(Disease~.,
               data=train_set,
               method="adaboost",
               trControl=fitControl) #fit model
pre_ada<-predict(fit_ada,test_set) #predict results
confusionMatrix(pre_ada,test_set$Disease) #make a confusion matrix to calculate Accuracy of the model
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 27  7
##          1  5 21
##
##                Accuracy : 0.8
##                  95% CI : (0.6767, 0.8922)
##     No Information Rate : 0.5333
##     P-Value [Acc > NIR] : 1.609e-05
##
##                   Kappa : 0.5964
##
##  Mcnemar's Test P-Value : 0.7728
##
##             Sensitivity : 0.8438
##             Specificity : 0.7500
##          Pos Pred Value : 0.7941
##          Neg Pred Value : 0.8077
##              Prevalence : 0.5333
##          Detection Rate : 0.4500
##    Detection Prevalence : 0.5667
##       Balanced Accuracy : 0.7969
##
##        'Positive' Class : 0
##
```

```r
#saving parameters and print the results
Accuracy_ada <- confusionMatrix(pre_ada,test_set$Disease)$overall["Accuracy"]
Sensitivity_ada <- confusionMatrix(pre_ada,test_set$Disease)$byClass["Sensitivity"]
Specificity_ada <- confusionMatrix(pre_ada,test_set$Disease)$byClass["Specificity"]
Bal.Accuracy_ada <- confusionMatrix(pre_ada,test_set$Disease)$byClass["Balanced Accuracy"]

results <- tribble(
    ~Method, ~Accuracy, ~Sensitivity,  ~Specificity, ~Bal.Accuracy,
    "KNN", Accuracy_KNN,  Sensitivity_KNN, Specificity_KNN,Bal.Accuracy_KNN,
    "Adaboost", Accuracy_ada,  Sensitivity_ada, Specificity_ada,Bal.Accuracy_ada)
results
```

```
## # A tibble: 2 x 5
##   Method   Accuracy Sensitivity Specificity Bal.Accuracy
##   <chr>       <dbl>       <dbl>       <dbl>        <dbl>
## 1 KNN         0.633       0.719       0.536        0.627
## 2 Adaboost    0.8         0.844       0.75         0.797
```

We got a good result with an accuracy of 0.8 and a Balanced Accuracy of 0.796 (we note just a slightly lack in the specificity of this model). The results are good, but we want improve it and try to get a better prediction, hence we will train another model on our dataset.

Model 3: General Logistic Regression model

At this point we want to change our strategy and we will try a regression type model, in particular will perform a logistic regression model and will check if it will yield the hoping improvements.

```
#Logistic Regression Model
set.seed(1, sample.kind = "Rounding") #use set.seed(1) if use R version < 3.6
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```
fit_GLM<-train(Disease~.,
               data= train_set,
               method="glm",
               family="binomial",
               trControl=fitControl) #fit model
pre_GLM<-predict(fit_GLM,test_set) #predict results
confusionMatrix(pre_GLM,test_set$Disease) #make a confusion matrix to calculate Accuracy of the model
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 26  5
##          1  6 23
##
##                Accuracy : 0.8167
##                  95% CI : (0.6956, 0.9048)
##     No Information Rate : 0.5333
##     P-Value [Acc > NIR] : 4.344e-06
##
##                   Kappa : 0.6325
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.8125
##             Specificity : 0.8214
##          Pos Pred Value : 0.8387
##          Neg Pred Value : 0.7931
##              Prevalence : 0.5333
##          Detection Rate : 0.4333
##    Detection Prevalence : 0.5167
##       Balanced Accuracy : 0.8170
##
##        'Positive' Class : 0
##
```

```r
#saving parameters and print the results
Accuracy_GLM <- confusionMatrix(pre_GLM,test_set$Disease)$overall["Accuracy"]
Sensitivity_GLM <- confusionMatrix(pre_GLM,test_set$Disease)$byClass["Sensitivity"]
Specificity_GLM <- confusionMatrix(pre_GLM,test_set$Disease)$byClass["Specificity"]
Bal.Accuracy_GLM <- confusionMatrix(pre_GLM,test_set$Disease)$byClass["Balanced Accuracy"]

results <- tribble(
    ~Method, ~Accuracy, ~Sensitivity,  ~Specificity, ~Bal.Accuracy,
    "KNN", Accuracy_KNN,  Sensitivity_KNN, Specificity_KNN,Bal.Accuracy_KNN,
    "Adaboost", Accuracy_ada,  Sensitivity_ada, Specificity_ada,Bal.Accuracy_ada,
    "General Logistic Regression Model", Accuracy_GLM,  Sensitivity_GLM, Specificity_GLM,Bal.Accuracy_Gl
results
```

```
## # A tibble: 3 x 5
##   Method                            Accuracy Sensitivity Specificity Bal.Accuracy
##   <chr>                                <dbl>       <dbl>       <dbl>        <dbl>
## 1 KNN                                  0.633       0.719       0.536        0.627
## 2 Adaboost                             0.8         0.844       0.75         0.797
## 3 General Logistic Regressio~          0.817       0.812       0.821        0.817
```

Comparing all the three models done we can see that GLM model improve considerly the values of Accuracy, Sensitivty and Specificity. So, we are on the good way, we could stop our research, but we are "hungry" researcher and we want to check if it is still possible to improve our results.

Model 4: Decision Tree model

As fourth model we will choose a decision tree model, and also for this algorithm will use the caret package with a 10 K-Fold cross validation.

```r
#Decision tree Model
set.seed(1, sample.kind = "Rounding") #use set.seed(1) if use R version < 3.6
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```r
fit_rpart<-train(Disease~.,
                 data= train_set,
                 method="rpart",
                 trControl=fitControl) #fit model
pre_rpart<-predict(fit_rpart,test_set) #predict results
confusionMatrix(pre_rpart,test_set$Disease) #make a confusion matrix to calculate Accuracy of the model
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 28  8
##          1  4 20
##
##                Accuracy : 0.8
##                  95% CI : (0.6767, 0.8922)
##      No Information Rate : 0.5333
##      P-Value [Acc > NIR] : 1.609e-05
##
##                    Kappa : 0.5946
##
##   Mcnemar's Test P-Value : 0.3865
##
##              Sensitivity : 0.8750
##              Specificity : 0.7143
##           Pos Pred Value : 0.7778
##           Neg Pred Value : 0.8333
##               Prevalence : 0.5333
##           Detection Rate : 0.4667
##     Detection Prevalence : 0.6000
##        Balanced Accuracy : 0.7946
##
##         'Positive' Class : 0
##
```

```
#saving parameters and print the results
Accuracy_rpart <- confusionMatrix(pre_rpart,test_set$Disease)$overall["Accuracy"]
Sensitivity_rpart <- confusionMatrix(pre_rpart,test_set$Disease)$byClass["Sensitivity"]
Specificity_rpart <- confusionMatrix(pre_rpart,test_set$Disease)$byClass["Specificity"]
Bal.Accuracy_rpart <- confusionMatrix(pre_rpart,test_set$Disease)$byClass["Balanced Accuracy"]

results <- tribble(
    ~Method, ~Accuracy, ~Sensitivity,  ~Specificity, ~Bal.Accuracy,
    "KNN", Accuracy_KNN,  Sensitivity_KNN, Specificity_KNN,Bal.Accuracy_KNN,
    "Adaboost", Accuracy_ada,  Sensitivity_ada, Specificity_ada,Bal.Accuracy_ada,
    "General Logistic Regression Model", Accuracy_GLM,  Sensitivity_GLM, Specificity_GLM,Bal.Accuracy_Gl
    "Decision Tree Model", Accuracy_rpart,Sensitivity_rpart,Specificity_rpart,Bal.Accuracy_rpart)
results
```

```
## # A tibble: 4 x 5
##   Method                      Accuracy Sensitivity Specificity Bal.Accuracy
##   <chr>                          <dbl>       <dbl>       <dbl>        <dbl>
## 1 KNN                            0.633       0.719       0.536        0.627
## 2 Adaboost                       0.8         0.844       0.75         0.797
## 3 General Logistic Regressio~    0.817       0.812       0.821        0.817
## 4 Decision Tree Model            0.8         0.875       0.714        0.795
```

Again, checking the results we see a slightly worsen value of accuracy, but despite a lack in specificity we had sensibly improve our prediction sesnsitivty; for this reason, seen the possibility to increase our predictive parameters, it has meant continuing our analysis trying to improve the specificity as well as the sensitivity.

Model 5: Random Forest model

As further model we will train a Random Forest model, that represents a deepening of the Decision Tree model seen before.

```
#Random Forest model
set.seed(1, sample.kind = "Rounding") #use set.seed(1) if use R version < 3.6
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```
fit_rf<-train(Disease~.,
              data= train_set,
              method="rf",
              trControl=fitControl) #fit model
pre_rf<-predict(fit_rf,test_set) #predict results
confusionMatrix(pre_rf,test_set$Disease) #make a confusion matrix to calculate Accuracy of the model
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 28  5
##          1  4 23
##
##                Accuracy : 0.85
##                  95% CI : (0.7343, 0.929)
##     No Information Rate : 0.5333
##     P-Value [Acc > NIR] : 2.293e-07
##
##                   Kappa : 0.698
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.8750
##             Specificity : 0.8214
##          Pos Pred Value : 0.8485
##          Neg Pred Value : 0.8519
##              Prevalence : 0.5333
##          Detection Rate : 0.4667
##    Detection Prevalence : 0.5500
##       Balanced Accuracy : 0.8482
##
##        'Positive' Class : 0
##
```

```
#saving parameters and print the results
Accuracy_rf <- confusionMatrix(pre_rf,test_set$Disease)$overall["Accuracy"]
Sensitivity_rf <- confusionMatrix(pre_rf,test_set$Disease)$byClass["Sensitivity"]
Specificity_rf <- confusionMatrix(pre_rf,test_set$Disease)$byClass["Specificity"]
Bal.Accuracy_rf <- confusionMatrix(pre_rf,test_set$Disease)$byClass["Balanced Accuracy"]

results <- tribble(
    ~Method, ~Accuracy, ~Sensitivity,  ~Specificity, ~Bal.Accuracy,
    "KNN", Accuracy_KNN,  Sensitivity_KNN, Specificity_KNN,Bal.Accuracy_KNN,
    "Adaboost", Accuracy_ada,  Sensitivity_ada, Specificity_ada,Bal.Accuracy_ada,
    "General Logistic Regression Model", Accuracy_GLM,  Sensitivity_GLM, Specificity_GLM,Bal.Accuracy_Gl
    "Decision Tree Model", Accuracy_rpart,Sensitivity_rpart,Specificity_rpart,Bal.Accuracy_rpart,
    "Random Forest", Accuracy_rf,  Sensitivity_rf, Specificity_rf,Bal.Accuracy_rf)
results
```

```
## # A tibble: 5 x 5
##   Method                      Accuracy Sensitivity Specificity Bal.Accuracy
##   <chr>                          <dbl>       <dbl>       <dbl>        <dbl>
## 1 KNN                            0.633       0.719       0.536        0.627
## 2 Adaboost                       0.8         0.844       0.75         0.797
## 3 General Logistic Regressio~    0.817       0.812       0.821        0.817
## 4 Decision Tree Model            0.8         0.875       0.714        0.795
## 5 Random Forest                  0.85        0.875       0.821        0.848
```

Our efforts are payed with a strong increase of accuracy (0.85), sensitivty (0.875) and specificity (0.821). All our parmeters are improved and now, once reached the decision point , we have to choose if stop the searching, or try to make some changes to improve again. We decide to try another strategy, will ensemble the three best models to see if is possible to get a more accurate result.

Ensemble model

The ensemble strategy involves calculating the average of the results obtained with the desired models, to generate a new prediction series, that will be subsequently compared with the real results through a confusion matrix.

```r
#Try to improve our models with ensemble strategy
ensemble<-data.frame(pre_GLM,pre_rf,pre_ada)
pre_ensemble<-as.factor(ifelse(rowMeans(ensemble==1)>0.5,1,0))
confusionMatrix(pre_ensemble,test_set$Disease)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 28  5
##          1  4 23
##
##                Accuracy : 0.85
##                  95% CI : (0.7343, 0.929)
##     No Information Rate : 0.5333
##     P-Value [Acc > NIR] : 2.293e-07
##
##                   Kappa : 0.698
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.8750
##             Specificity : 0.8214
##          Pos Pred Value : 0.8485
##          Neg Pred Value : 0.8519
##              Prevalence : 0.5333
##          Detection Rate : 0.4667
##    Detection Prevalence : 0.5500
##       Balanced Accuracy : 0.8482
##
##        'Positive' Class : 0
##
```

```r
#saving parameters
Accuracy_ens <- confusionMatrix(pre_ensemble,test_set$Disease)$overall["Accuracy"]
Sensitivity_ens <- confusionMatrix(pre_ensemble,test_set$Disease)$byClass["Sensitivity"]
Specificity_ens<- confusionMatrix(pre_ensemble,test_set$Disease)$byClass["Specificity"]
Bal.Accuracy_ens <- confusionMatrix(pre_ensemble,test_set$Disease)$byClass["Balanced Accuracy"]
```

```
## # A tibble: 6 x 5
##   Method                      Accuracy Sensitivity Specificity Bal.Accuracy
##   <chr>                          <dbl>       <dbl>       <dbl>        <dbl>
## 1 KNN                            0.633       0.719       0.536        0.627
## 2 Adaboost                       0.8         0.844       0.75         0.797
## 3 General Logistic Regressio~    0.817       0.812       0.821        0.817
## 4 Decision Tree Model            0.8         0.875       0.714        0.795
## 5 Random Forest                  0.85        0.875       0.821        0.848
## 6 Ensamble Model                 0.85        0.875       0.821        0.848
```

"' As we can see the ensemble model didn't improve our results achieved from the Random Forest, so we can stop to search another algorithm or another strategy and choose as definitive approach the Random Forest model.

Conclusions

In conclusion we can say that we have achieved good predictive results through the Random Forest model. The levels of accuracy, sensitivity and specificity are quite satisfactory and allow to predict in 85% of cases the onset of heart disease in both men and women. Probably further variables to be examined could positively influence forecasts and further improve our machine learning models.