Heart Disease Prediction

Leonardo Ribeiro

08/10/2021

Heart Disease in Patients

The purpose of this work is to formulate a Machine Learning (ML) model that points out, with reasonable precision, if a patient has a heart disease. In order do to so, a data set provided by Kaggle will be used. Next, the variables in the data set are be described:

Field	Description
Age	age of the patient [years]
Sex	sex of the patient [M: Male, F: Female]
ChestPainType	chest pain type [TA: Typical Angina, ATA: Atypical Angina, NAP: Non-Anginal Pain, ASY: Asymptomatic]
RestingBP	resting blood pressure [mm Hg]
Cholesterol	serum cholesterol [mm/dl]
FastingBS	fasting blood sugar [1: if FastingBS $>$ 120 mg/dl, 0: otherwise]
RestingECG	resting electrocardiogram results [Normal: Normal, ST: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of $>$ 0.05 mV), LVH: showing probable or definite left ventricular hypertrophy by Estes' criteria
MaxHR	maximum heart rate achieved [Numeric value between 60 and 202]
ExerciseAngina	exercise-induced angina [Y: Yes, N: No]
Oldpeak	oldpeak = ST [Numeric value measured in depression]
ST_Slope	the slope of the peak exercise ST segment [Up: upsloping, Flat: flat, Down: downsloping]
HeartDisease	output class [1: heart disease, 0: Normal]

For more details about each variable and its sources, visit **this link**.

The Goal

A patient arrives at a hospital to perform a battery of tests. The machine learning model must predict if he/she has a heart disease.

Loading the Data

The first step is to load the data set of interest:

```
file <- list.files(pattern = ".csv")
raw_data <- read.csv(file)
dim(raw_data)
```

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

```
colnames(raw_data)
```

```
##  [1] "Age"            "Sex"            "ChestPainType"  "RestingBP"
##  [5] "Cholesterol"    "FastingBS"      "RestingECG"     "MaxHR"
##  [9] "ExerciseAngina" "Oldpeak"        "ST_Slope"       "HeartDisease"
```

The data ensemble has 918 rows and 12 columns (representing characteristics of the patients). The column HeartDisease is the one that indicates if a a patient is sick (1) or healthy (0). Therefore, the ML model must predict the value present on this column.

Exploratory Data Analysis

The variables RestingBP (resting electrocardiogram results) and Cholesterol have the following characteristics:

```
summary(raw_data[, c("Cholesterol", "RestingBP")])
```

```
##   Cholesterol      RestingBP
##  Min.   :  0.0   Min.   :  0.0
##  1st Qu.:173.2   1st Qu.:120.0
##  Median :223.0   Median :130.0
##  Mean   :198.8   Mean   :132.4
##  3rd Qu.:267.0   3rd Qu.:140.0
##  Max.   :603.0   Max.   :200.0
```

Thus, both columns contain the value zero. It suggests that this data was not collected for all the subjects.

```
nrow(raw_data[raw_data$RestingBP == 0.0 ,])
```

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

```
nrow(raw_data[raw_data$Cholesterol == 0.0 ,])
```

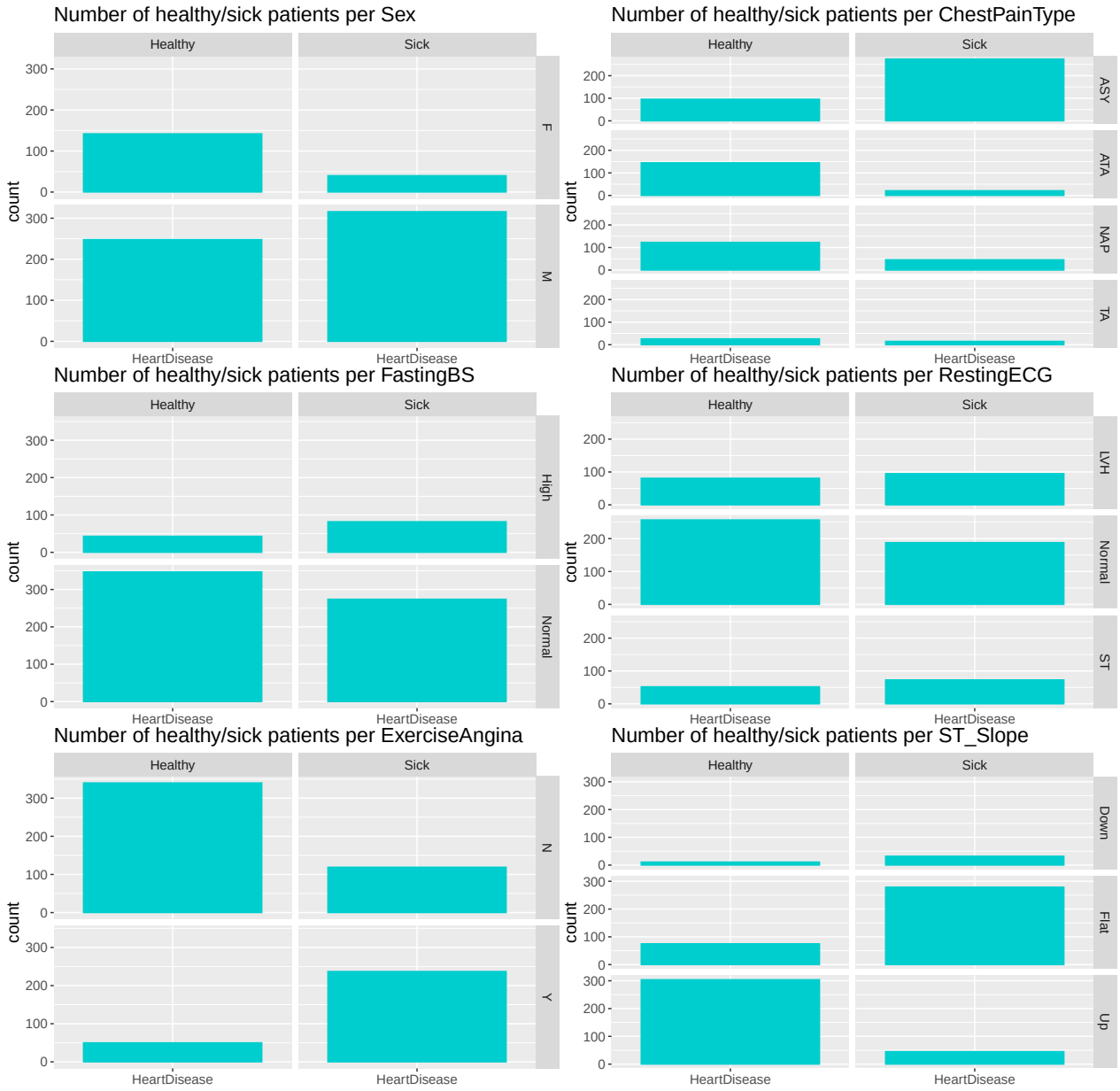
```
## [1] 172
```

```
raw_data <- raw_data[raw_data$RestingBP != 0.0,]
raw_data <- raw_data[raw_data$Cholesterol != 0.0,]
```

There is only a single patient for whom the variable RestingBP is zero, but there are 172 subjects for which the Cholesterol index was not registered. In order to avoid statistical aberrations, the corresponding rows will be discarded. Therefore, the total number of lines will be reduced by almost 19%.

The next step is to evaluate how the variables ChestPainType, FastingBS, RestingECG, ExerciseAngina and ST_Slope are related with the target column, HeartDisease.

```
library(ggplot2)
raw_data$HeartDisease = sapply(raw_data$HeartDisease, function(x) {
  ifelse(x == '0', "Healthy", "Sick")
})
raw_data$FastingBS = sapply(raw_data$FastingBS, function(x) {
  ifelse(x == '0', "Normal", "High")
})
cols_of_interest <- c('Sex', 'ChestPainType', 'FastingBS', 'RestingECG',
                      'ExerciseAngina', 'ST_Slope')
dv.plot_multiple_bars_II(raw_data, cols_of_interest, 'HeartDisease', title = 'Number of healthy/sick pa
```



The first noticeable fact is that the majority of women that seek hospitals in order to perform the exams analysed here are healthy, while the opposite is true for men. This indicates that the former group has a greater care with their personal health when compared with the later.

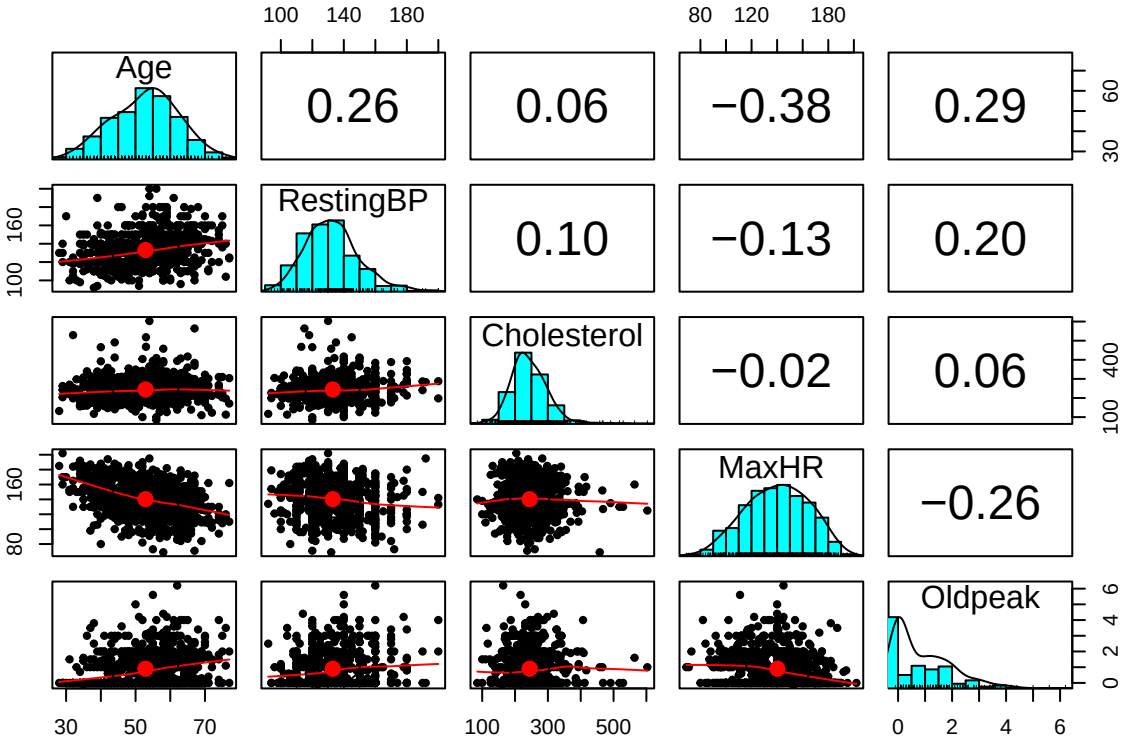
Another interesting point is that the majority of asymptomatic patients in the data set are sick. This may be an indicator that those exams are routinely sought by people during their heart disease treatment. Furthermore, the largest part of subjects with a high concentration of sugar on their blood are sick, while the majority of the ones with normal levels are healthy.

The variable RestingECG also seems to be relevant. The majority of patients with value 'normal' for this variable are healthy, while the opposite is true. Moreover, just like RestingECG, the value on the column ExerciseAngina also seems to be an important factor regarding the presence of a heart disease.

Finally, another relevant indicator is the ST_Slope. The largest part of subjects with value 'Up' on this column are healthy, while most of the ones with 'Down' or 'Flat' values are sick.

Even tough all the variables studied so far have a qualitative nature, the data set also contains quantitative data. Those are related to each other in the following manner:

```
library(psych)
pairs.panels(raw_data[c('Age', 'RestingBP', 'Cholesterol', 'MaxHR', 'Oldpeak')])
```



Despite the fact that there is no strong correlation among the predictor variables, the column 'Age' presents a weak correlation with the others - a negative one with MaxHR and a positive one Oldpeak and RestingBp. In fact, the correlation with the later is well known since de 80s (LONDEREE and MOESCHBERGER (1982) Effect of age and other factors on HRmax. Research Quarterly for Exercise & Sport, 53 (4), p. 297-304). On the other hand, it is interesting to notice how Cholesterol has essentially no correlation with the remaining variables.

Data Manipulation

Assuming that small variations regarding the column Age are irrelevant, its values will be divided in ranges:

```
raw_data <- dm.range_divide(raw_data, 'Age', 5, 1)
levels(raw_data$Age_range) <- c('28-37', '38-47', '48-57', '58-67', '68-77')
```

Besides Age, the columns RestingBP, Cholesterol, MaxHR and Oldpeak are numeric too. Therefore, their variables will also be converted to factor, as well as the remaining columns:

```
raw_data <- dm.range_divide(raw_data, 'RestingBP', 5, 1)

raw_data <- dm.range_divide(raw_data, 'MaxHR', 5, 1)

raw_data <- dm.range_divide(raw_data, 'Oldpeak', 5, .1)

raw_data <- dm.range_divide(raw_data, 'Cholesterol', 5, 1)

fac_cols <- c('Sex', 'ChestPainType', 'FastingBS', 'RestingECG',
```

```
              'ExerciseAngina', 'ST_Slope', 'HeartDisease')

raw_data = dm.factorize_cols(raw_data, fac_cols)
```

Up to this point, not a single column was removed from the data set - some were added and others had their variable type changed. Now, in order to focus on the data of interest, the columns Age, RestingBP, Cholesterol, MaxHR and Oldpeak will be dropped. Additionally, in order for the model to not to differentiate between male and female patients, the column Sex will also be removed:

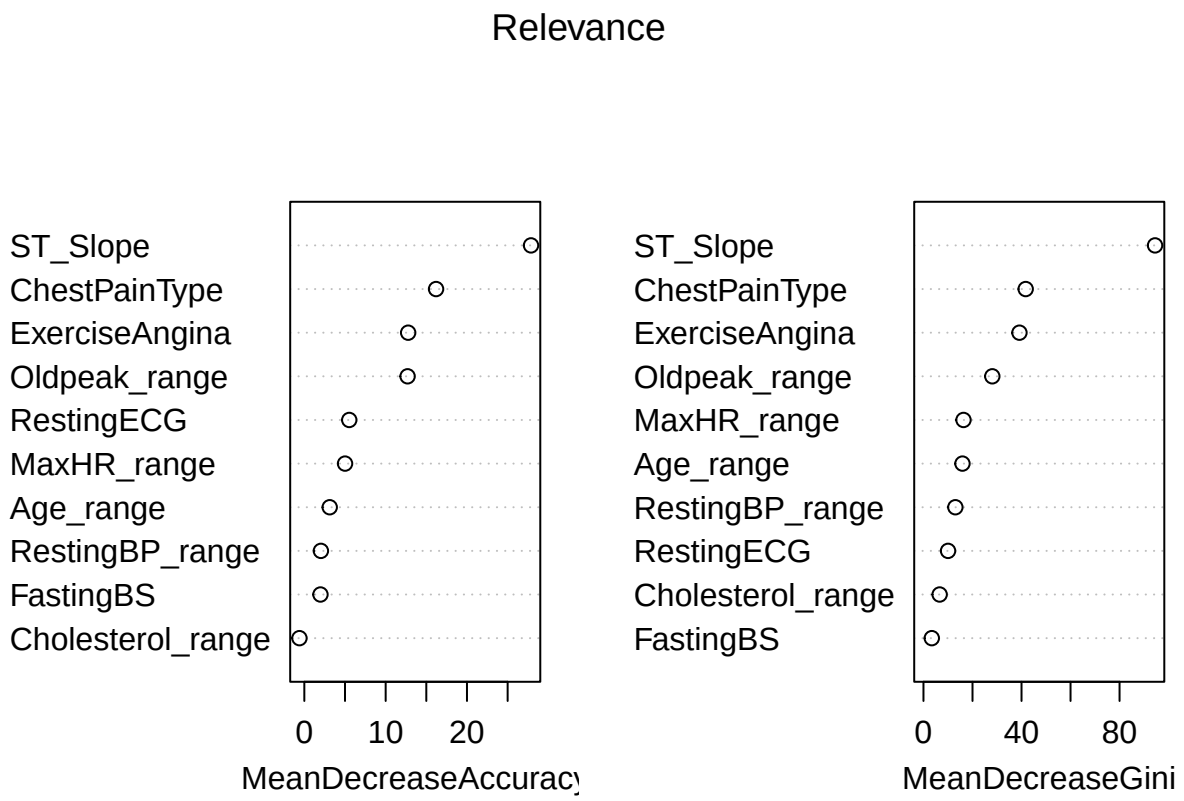
```
cols_to_drop <- c('Sex', 'Age', 'RestingBP', 'Cholesterol', 'MaxHR', 'Oldpeak')
data <- dm.drop_cols(raw_data, cols_to_drop)
```

Now that the data was cleaned, discussed and manipulated, the ML model can be built.

Building the Machine Learning model

Using randomForest(), one can identify the relevance of each variable before starting the construction of the model itself:

```
library(randomForest)
Relevance <- randomForest(HeartDisease ~., data = data, ntree = 100,
                          nodesize = 10, importance = T)
varImpPlot(Relevance)
```



Accordingly to both criteria used above, the 4 most relevant variables are the same: ST_Slope, ChestPainType, Oldpeak_range and ExerciseAngina. Nonetheless, since the data set has a relatively small amount of data, even the less influential columns will be kept.

The next step is to divide the data set into a pair: a training set (with 80% of the data) and a testing set (with the remaining 20%). Then, by using the former, the ML model can be built:

```
random_indexes <- de.get_random_row_indexes(data, 80)
trainSet <- data[random_indexes, ]
testSet <- data[-random_indexes, ]
model <- randomForest(HeartDisease ~., data = trainSet,
                      ntree = 100, nodesize = 10)
model
```

```
##
## Call:
##  randomForest(formula = HeartDisease ~ ., data = trainSet, ntree = 100,      nodesize = 10)
##                Type of random forest: classification
##                      Number of trees: 100
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 15.1%
## Confusion matrix:
##         Healthy Sick class.error
## Healthy     261   49   0.1580645
## Sick         41  245   0.1433566
```

This result shows that the ML model has a small error rate - around 15%. Evidently, for something as important as the health of an individual, this error is way too large. Nonetheless, now the test data will be used to evaluate the precision of the model:

```
library(caret)
prediction = predict(model, newdata = testSet)
comparisonTable = data.frame(observed = testSet$HeartDisease,
                             predicted = prediction)
confusionMatrix(comparisonTable$observed, comparisonTable$predicted)
```

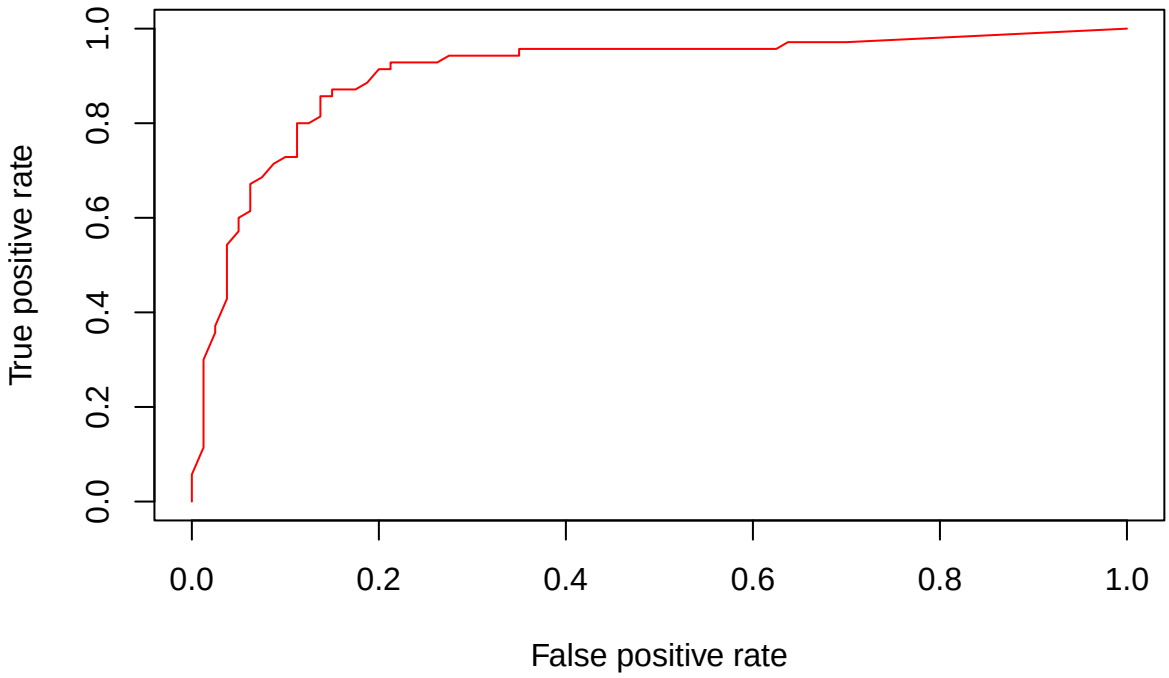
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction Healthy Sick
##    Healthy      68   12
##    Sick          9   61
##
##                Accuracy : 0.86
##                  95% CI : (0.794, 0.9112)
##     No Information Rate : 0.5133
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.7195
##
##  Mcnemar's Test P-Value : 0.6625
##
##             Sensitivity : 0.8831
##             Specificity : 0.8356
##          Pos Pred Value : 0.8500
##          Neg Pred Value : 0.8714
##              Prevalence : 0.5133
##          Detection Rate : 0.4533
##    Detection Prevalence : 0.5333
##       Balanced Accuracy : 0.8594
##
```

```
##         'Positive' Class : Healthy
##
```

Therefore, using the ML model that was just created to predict the current state of a patient, an accuracy between 84% and 90% was obtained. This model could be optimized and techniques could be implemented in order to improve its accuracy, but that is beyond the scope of this text.

Additionally, another important feature when evaluating a ML model is the ROC curve:

```
library(ROCR)
class1 <- predict(model, newdata = testSet, type = 'prob')
class2 <- testSet$HeartDisease
pred <- prediction(class1[,2], class2)
perf <- performance(pred, 'tpr', 'fpr')
auc <- dv.get_auc(pred)
plot(perf, col = rainbow(10))
```




```
print(auc)
```

```
## [1] 0.906875
```

Since the area under the curve is above 0.9, one can say the ML model created has an excellent overall performance.

Conclusion

The Machine Learning model constructed during this study was able to predict the presence of a heart disease in the large majority of the cases. Other models could be created, removing potentially problematic columns from the data set for instance, but the very high value for the area under the AOC curve and the above 84% accuracy are enough for the purposes of this demonstration.