Heart Failure Risk Prediction

HarvardX Data Science Professional Certificate: Capstone project(2)

Hatem RABEH

2022-11-19

Contents

1. Scope:

This is a second project created in the process of obtaining the Data Science professional certificate from Harvardx University.

2. Introduction:

Heart failure (HF), also known as congestive heart failure (CHF), is a syndrome, a group of signs and symptoms caused by an impairment of the heart's blood pumping function. Symptoms typically include shortness of breath, excessive fatigue, and leg swelling. The shortness of breath may occur with exertion or while lying down and may wake people up during the night. Chest pain, including angina, is not usually caused by heart failure but may occur if the heart failure was caused by a heart attack. The severity of heart failure is measured by the severity of symptoms during exercise. Other conditions that may have symptoms similar to heart failure include obesity, kidney failure, liver disease, anemia, and thyroid disease. (sources: Wikipedia) In 2015, heart failure affected about 40 million people globally. Overall, around 2% of adults have heart failure and in those over the age of 65, this increases to 6–10%. Above 75 years old, rates are greater than 10%. (sources: Wikipedia) People with a high risk of heart failure (due to the presence of one or more risk factors such as hypertension, diabetes, hyperlipidemia, or already established disease) need early detection and management wherein a machine learning model can be of great help

2.1. Objective:

The main goal of this project is to explore through machine learning techniques the impact of several variables on the heart failure rate and if is it possible to predict its occurrence.

2.2. Dataset presentation:

The used data is a published data on Kaggle containing 918 observations and 12 variables.

The follwing is the dataset variables' description :

1. Age: age of the patient [years]
2. Sex: sex of the patient [M: Male, F: Female]
3. ChestPainType: chest pain type [TA: Typical Angina, ATA: Atypical Angina, NAP: Non-Anginal Pain, ASY: Asymptomatic]
4. RestingBP: resting blood pressure [mm Hg]
5. Cholesterol: serum cholesterol [mm/dl]
6. FastingBS: fasting blood sugar [1: if FastingBS > 120 mg/dl, 0: otherwise]
7. 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]
8. MaxHR: maximum heart rate achieved [Numeric value between 60 and 202]
9. ExerciseAngina: exercise-induced angina [Y: Yes, N: No]
10. Oldpeak: oldpeak = ST [Numeric value measured in depression]
11. ST_Slope: the slope of the peak exercise ST segment [Up: upsloping, Flat: flat, Down: downsloping]
12. HeartDisease: output class [1: heart disease, 0: Normal]

2.3. Method and steps to be implemented:

1) Import and read the dataset
2) DATA Wrangling
3) DATA visualization and DATA Analysis

4) Creat Models
5) Conclusion

3. Methods and analysis

3.1. Import and read the dataset

The github repo with the data set "Heart_failure_data_set" is available here: //github.com/azouri123/Own-Project-Submission_HRA_ECG-analysis-/blob/main/heart_failure_data_se

The file "Heart_failure_data_set.csv" provided in the github repo must be included in the working (project) directory for the code below to run

```
#Importing the data set:
set.seed(1, sample.kind="Rounding")
heart_failure_dataset <- read.csv("heart_failure_data_set.csv")
```

3.2. DATA Wrangling

3.2.1 presenting the dataset: The following is a presentation of the data_set:

```
head(heart_failure_dataset)
```

Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseA
40	M	ATA	140	289	0	Normal	172	N
49	F	NAP	160	180	0	Normal	156	N
37	M	ATA	130	283	0	ST	98	N
48	F	ASY	138	214	0	Normal	108	Y
54	M	NAP	150	195	0	Normal	122	N
39	M	NAP	120	339	0	Normal	170	N

The following is a summarize of the Data_set:

```
summary(heart_failure_dataset)
```

```
##       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
```

3.2.2 Missing data review : To better understand the data_set we are going to look for missing data :

```
any(is.na(heart_failure_dataset))
```

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

As described above there is no missing data in our data_set.

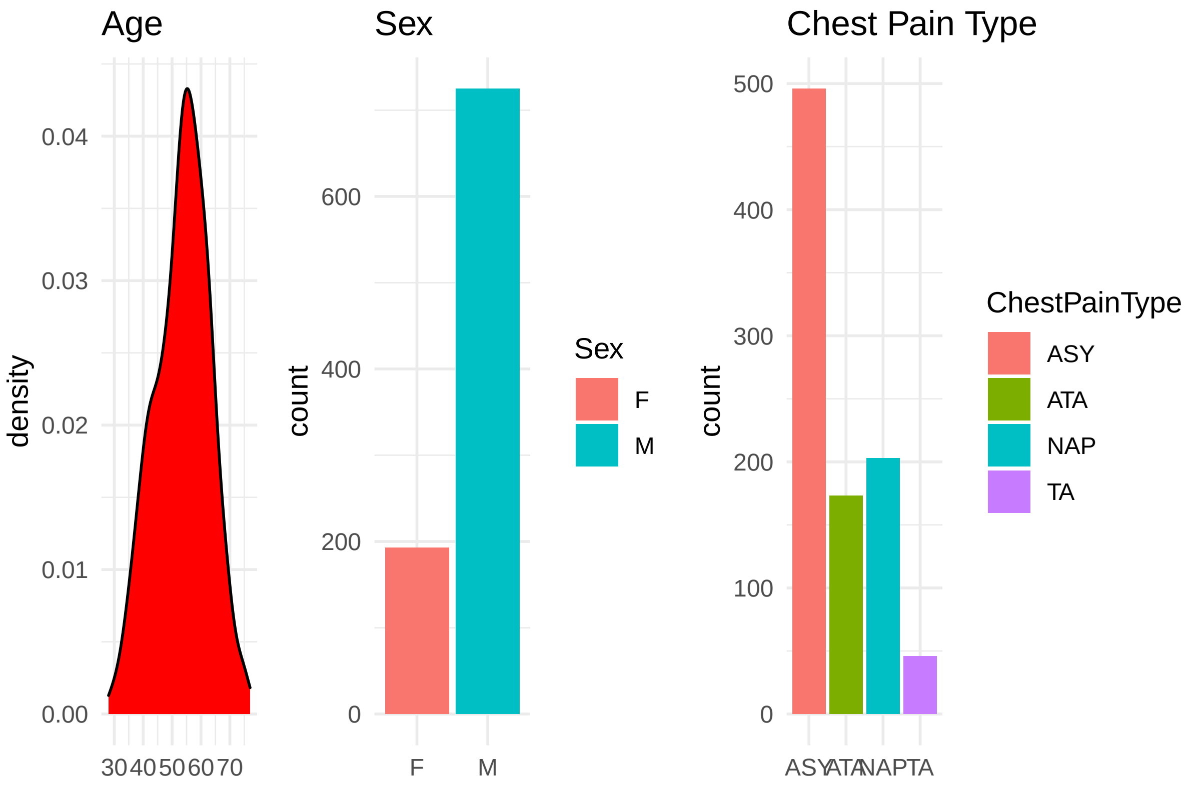
Let's now factorize the categorical variables so R doesn't process them as character strings or continuous numeric data :

```

heart_failure_dataset$Sex <- as.factor(heart_failure_dataset$Sex)
heart_failure_dataset$ChestPainType <- as.factor(heart_failure_dataset$ChestPainType)
heart_failure_dataset$FastingBS <- as.factor(heart_failure_dataset$FastingBS)
heart_failure_dataset$RestingECG <- as.factor(heart_failure_dataset$RestingECG)
heart_failure_dataset$ExerciseAngina <- as.factor(heart_failure_dataset$ExerciseAngina)
heart_failure_dataset$ST_Slope <- as.factor(heart_failure_dataset$ST_Slope)
```

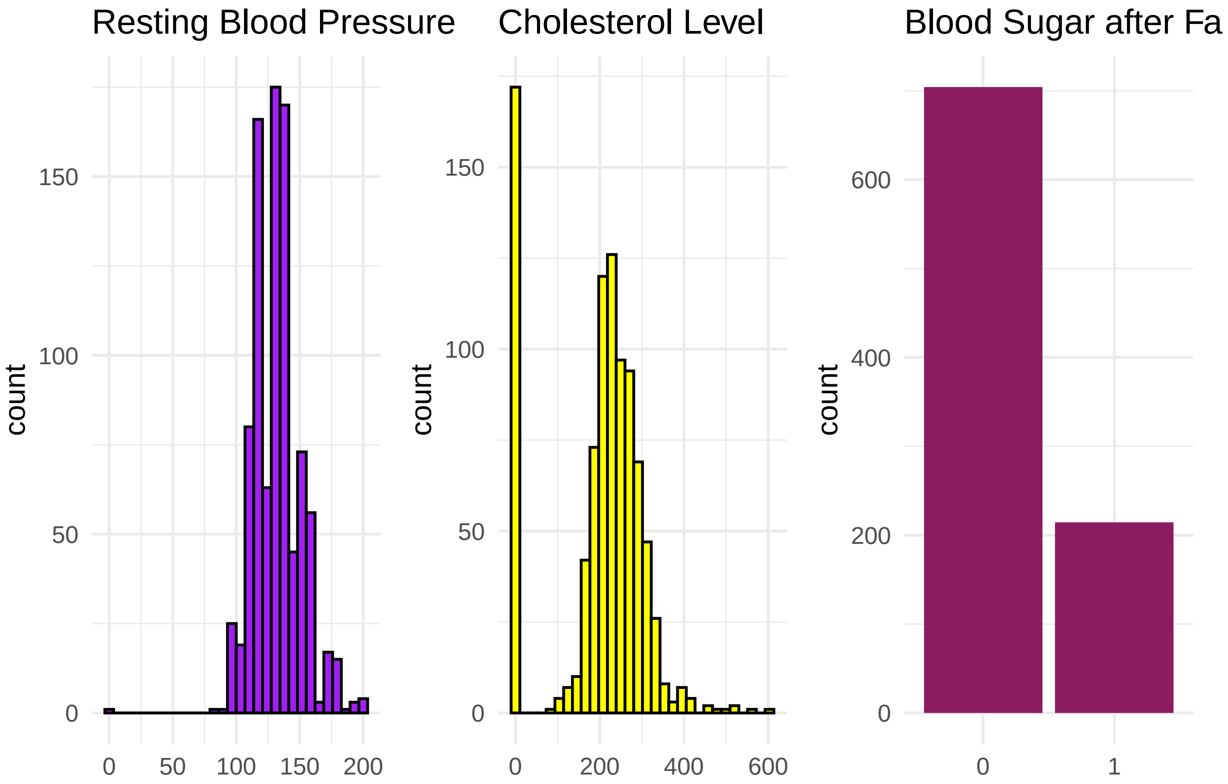
3.2.3 Harmonizing the variables

3.2.4 Check for Outlier: Outliers in a data_set could introduce bias and errors in the data analysis. Thus, a check for outliers in dataset is an important setp:

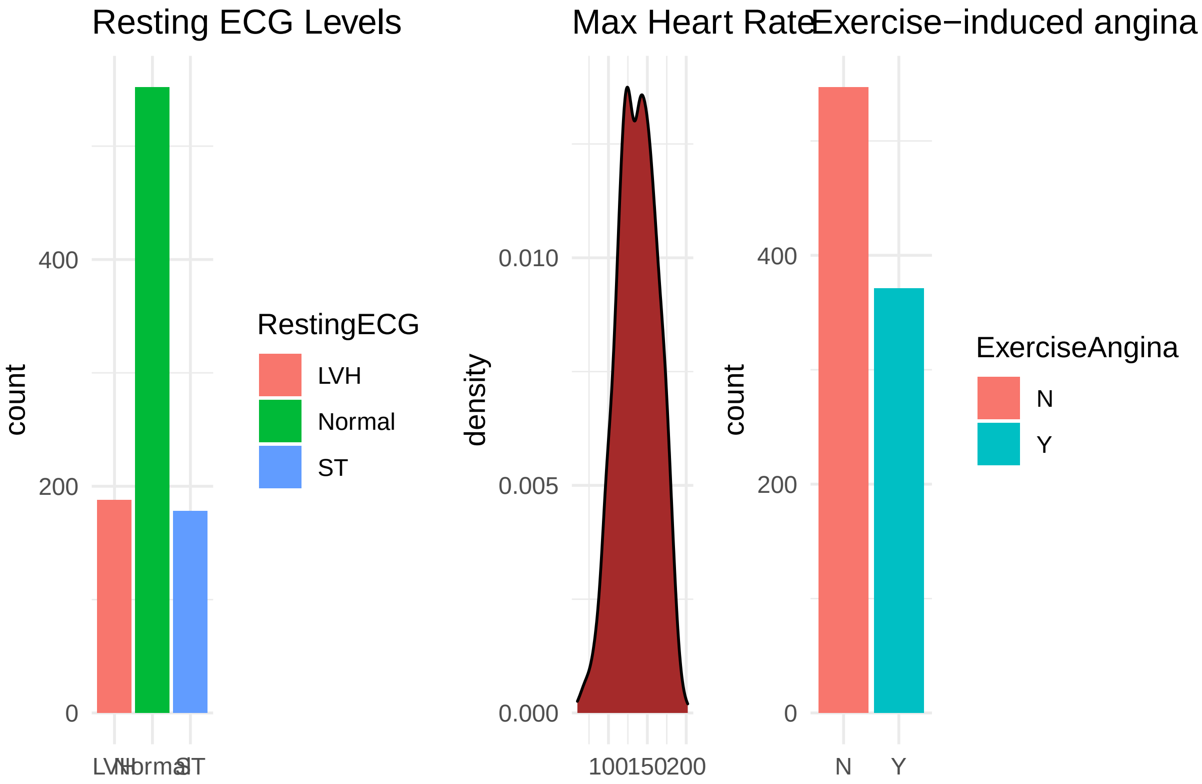
```
p1 <- heart_failure_dataset %>% ggplot(aes(Age)) + geom_density(fill = "Red") + labs(x="
p2 <- heart_failure_dataset %>% ggplot(aes(Sex)) + geom_bar(aes(fill = Sex)) + labs(x=""
p3 <- heart_failure_dataset %>% ggplot(aes(ChestPainType)) + geom_bar(aes(fill = ChestPa
p4 <- heart_failure_dataset %>% ggplot(aes(RestingBP)) + geom_histogram(colour = "black"
p5 <- heart_failure_dataset %>% ggplot(aes(Cholesterol)) + geom_histogram(colour = "blac
p6 <- heart_failure_dataset %>% ggplot(aes(FastingBS)) + geom_bar(fill = "maroon4") + la
p7 <- heart_failure_dataset %>% ggplot(aes(RestingECG)) + geom_bar(aes(fill = RestingECG
p8 <- heart_failure_dataset %>% ggplot(aes(MaxHR)) + geom_density(fill = "brown") + labs
p9 <- heart_failure_dataset %>% ggplot(aes(ExerciseAngina)) + geom_bar(aes(fill = Exerci
p10 <- heart_failure_dataset %>% ggplot(aes(Oldpeak)) + geom_histogram(color = "black",
p11 <- heart_failure_dataset %>% ggplot(aes(ST_Slope)) + geom_bar(aes(fill = ST_Slope))
p12 <- heart_failure_dataset %>% ggplot(aes(HeartDisease)) + geom_bar(fill = "purple") +

(p1 | p2 | p3)
```


Age
density
0.04
0.03
0.02
0.01
0.00
30 40 50 60 70
Sex
count
600
400
200
0
F
M
Sex
F
M
Chest Pain Type
count
500
400
300
200
100
0
ASY ATA NAP TA
ChestPainType
ASY
ATA
NAP
TA

```
(p4 | p5 | p6)
```

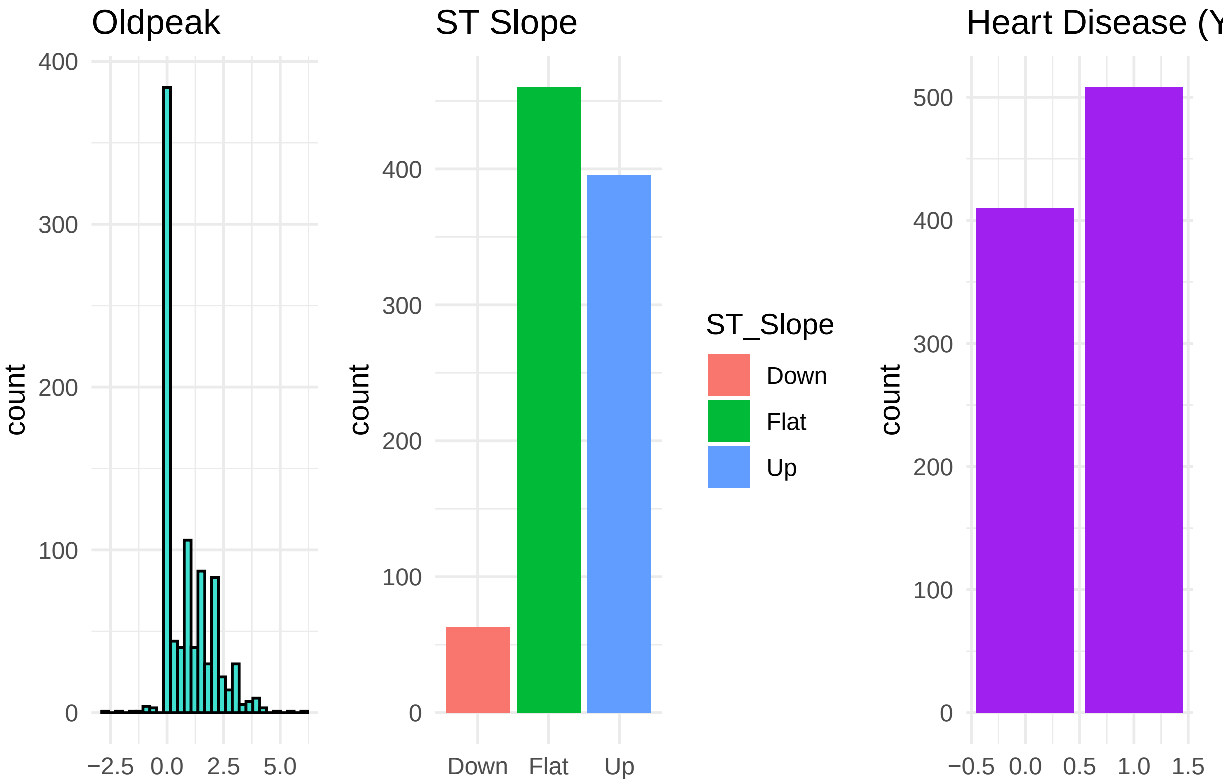

Resting Blood Pressure
count
0
50
100
150
0
50
100
150
200
Cholesterol Level
count
0
50
100
150
0
200
400
600
Blood Sugar after Fa
count
0
200
400
600
0
1

```
(p7 | p8 | p9)
```



Resting ECG Levels
count
400
200
0
LVH
Normal
ST
RestingECG
LVH
Normal
ST
Max Heart Rate
density
0.010
0.005
0.000
100
150
200
Exercise−induced angina
count
400
200
0
N
Y
ExerciseAngina
N
Y

```
(p10 | p11 | p12)
```



We conclude, that the restingBp = 0 is an outlier and is for sure an error because if RestingBp = 0 so the patient is dead. Moreover, we can not have a cholesterol level = 0. So we are going to use data imputation on this outliers.

We will start with the RestingBP:

3.2.4.1 Data imputation (RestingBP) Before imputating data we need to understand it:

```
heart_failure_dataset %>% filter(RestingBP == 0)
```

Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseA
55	M	NAP	0	0	0	Normal	155	N

The Patient have a normal restingECG so definitely there was an error. Different imputation method exist like Hot-deck, clod-deck, mean substitution or Non-negative matrix factorization.

For this, we gonna use the hot-deck imputation which was invented in the 1950s and consists simply of replacing every missing value with the last observed value in the same variable.

For that we ganna replace the RextingBp = 0 to RextingBp = NA and then use the hotdeck function.

Hot-deck in its vanilla form may break relations between variables, that is why we will impute within domains (HeartDisease)

```
heart_failure_dataset$RestingBP <- replace(heart_failure_dataset$RestingBP,heart_failure
## Verifying that there is no more 0 in the RestingBp
heart_failure_dataset %>% filter(RestingBP == 0)
```

Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseA

```
## Verifying that there is Na in the new dataset
any(is.na(heart_failure_dataset))
```

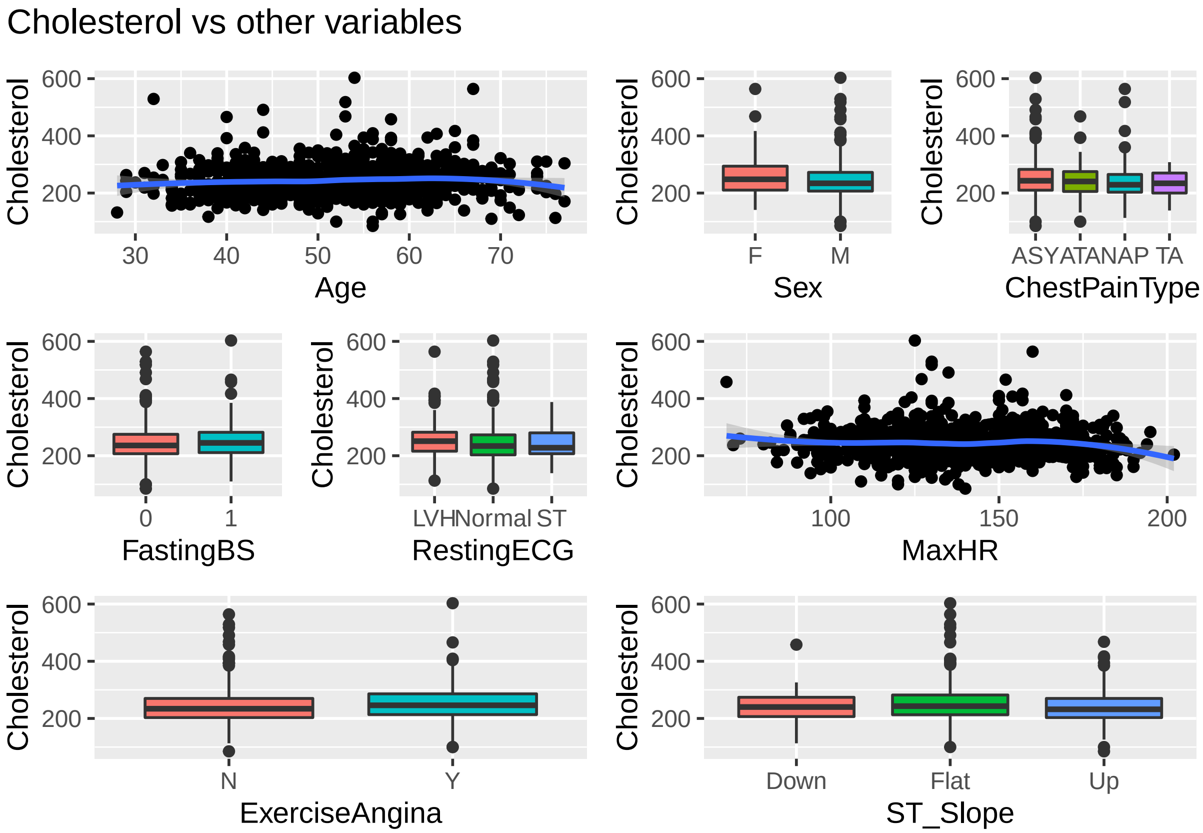
```
## [1] TRUE
```

```
## using the hotdeck function to replace the Na
heart_failure_dataset <- hotdeck(heart_failure_dataset,domain_var = "HeartDisease")
##Verifying that we do not have any NA
any(is.na(heart_failure_dataset))
```

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

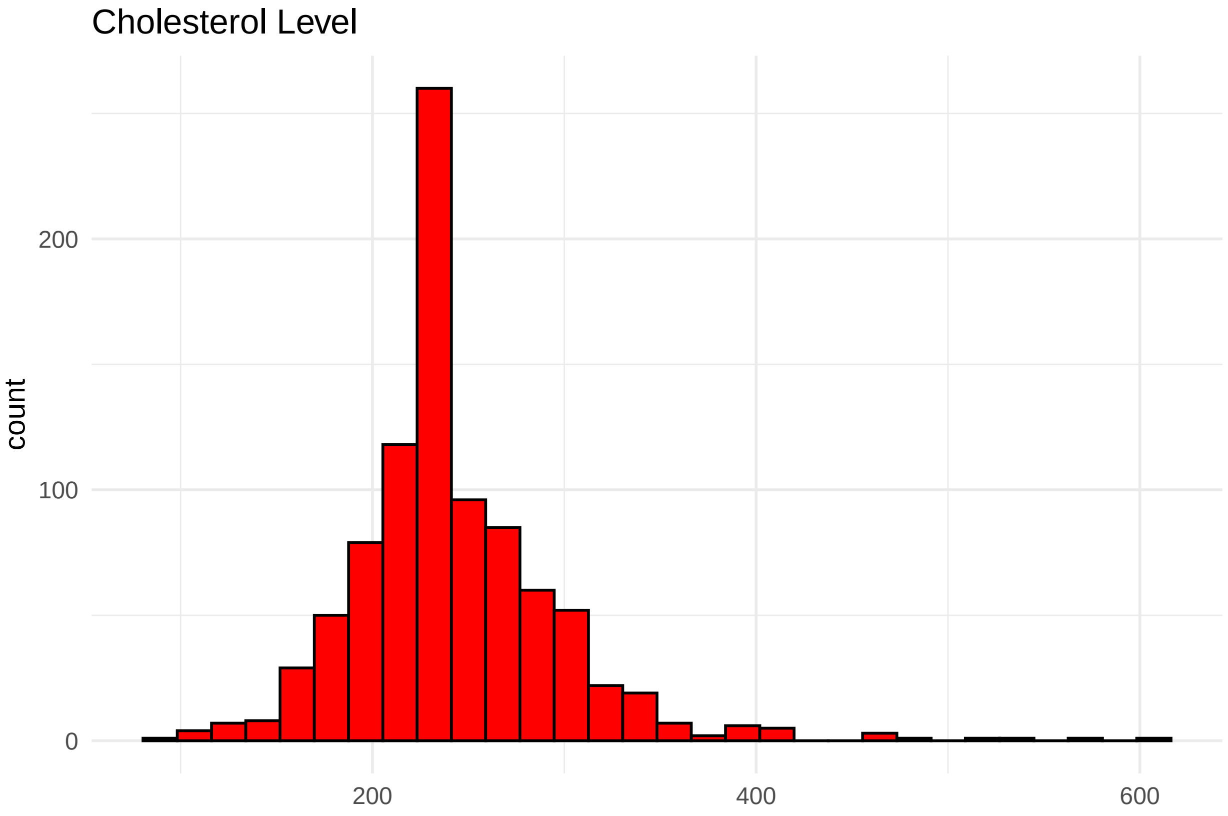
3.2.4.2 Data imputation (Cholesterol level): The other non comprehensive value deducted from the the graphs is the cholesterol level = 0 For that we will use the mean substitution, but the problem is, that method may attenuates any correlations involving the variable(s) that are imputed. So before that we will look for the variable who the most predict the cholesterol level, thus we can adjust the mean to it:

```
ch1 <- heart_failure_dataset %>% filter(Cholesterol > 0) %>% ggplot(aes(x=Age,y=Choleste
ch2 <- heart_failure_dataset %>% filter(Cholesterol > 0) %>% ggplot(aes(x=Sex,y=Choleste
ch3 <- heart_failure_dataset %>% filter(Cholesterol > 0) %>% ggplot(aes(x=ChestPainType,
ch4 <- heart_failure_dataset %>% filter(Cholesterol > 0) %>% ggplot(aes(x=FastingBS,y=Ch
ch5 <- heart_failure_dataset %>% filter(Cholesterol > 0) %>% ggplot(aes(x=RestingECG,y=C
ch6 <- heart_failure_dataset %>% filter(Cholesterol > 0) %>% ggplot(aes(x=MaxHR,y=Choles
ch7 <- heart_failure_dataset %>% filter(Cholesterol > 0) %>% ggplot(aes(x=ExerciseAngina
ch8 <- heart_failure_dataset %>% filter(Cholesterol > 0) %>% ggplot(aes(x=ST_Slope,y=Cho
design <- "
AABC
DEFF
GGHH
"

ch1 + ch2 + ch3 + ch4 + ch5 + ch6 + ch7 + ch8 +
  plot_layout(design = design) + plot_annotation(title = "Cholesterol vs other variables
```



The comparison of different variable against the cholesterol level shows that the sex is the variable that impact the most the cholesterol level. That is why we will replace the Cholesterol level = 0 with the median cholesterol level adjusted to sex :

```
m.Chol.Median <- heart_failure_dataset %>% filter(Sex == "M", Cholesterol > 0) %>% summa
f.Chol.Median <- heart_failure_dataset %>% filter(Sex == "F", Cholesterol > 0) %>% summa

heart_failure_dataset$Cholesterol[heart_failure_dataset$Sex == "M" & heart_failure_datas
heart_failure_dataset$Cholesterol[heart_failure_dataset$Sex == "F" & heart_failure_datas
#Checking the results:
heart_failure_dataset %>% ggplot(aes(Cholesterol)) + geom_histogram(colour = "black",fil
```

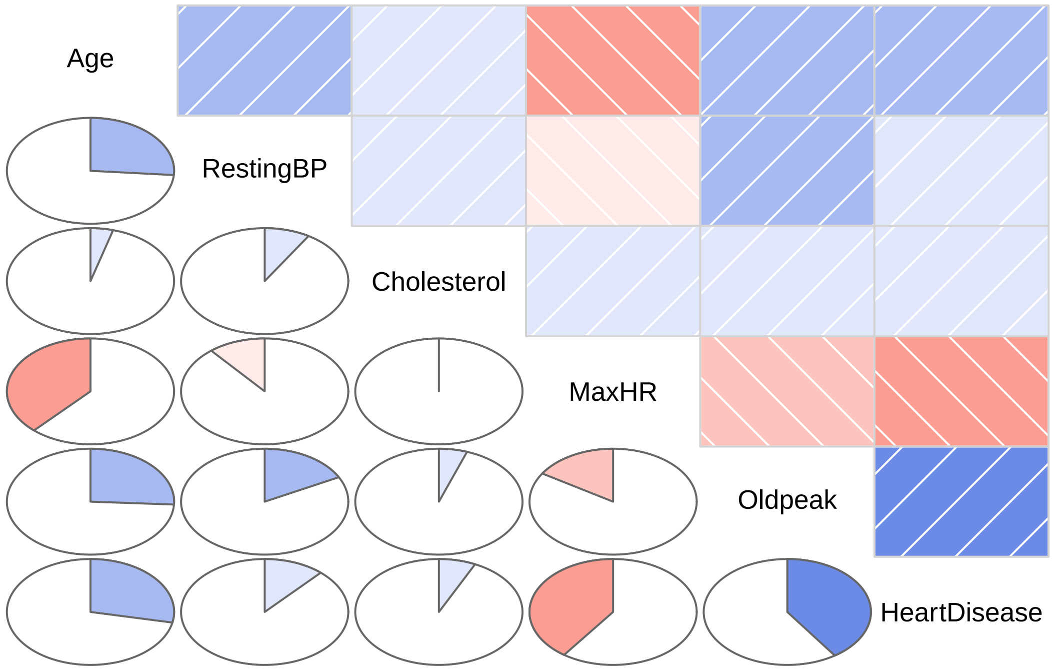


We have now a clean and tidy data_set, We can now compare the HeartDisease variable with all other variable to appraise if there is any correlation:

3.3. DATA Analysis:

We have now a clean and tidy data_set, We can now compare the HeartDisease variable with all other variable to appraise if there is any correlation:

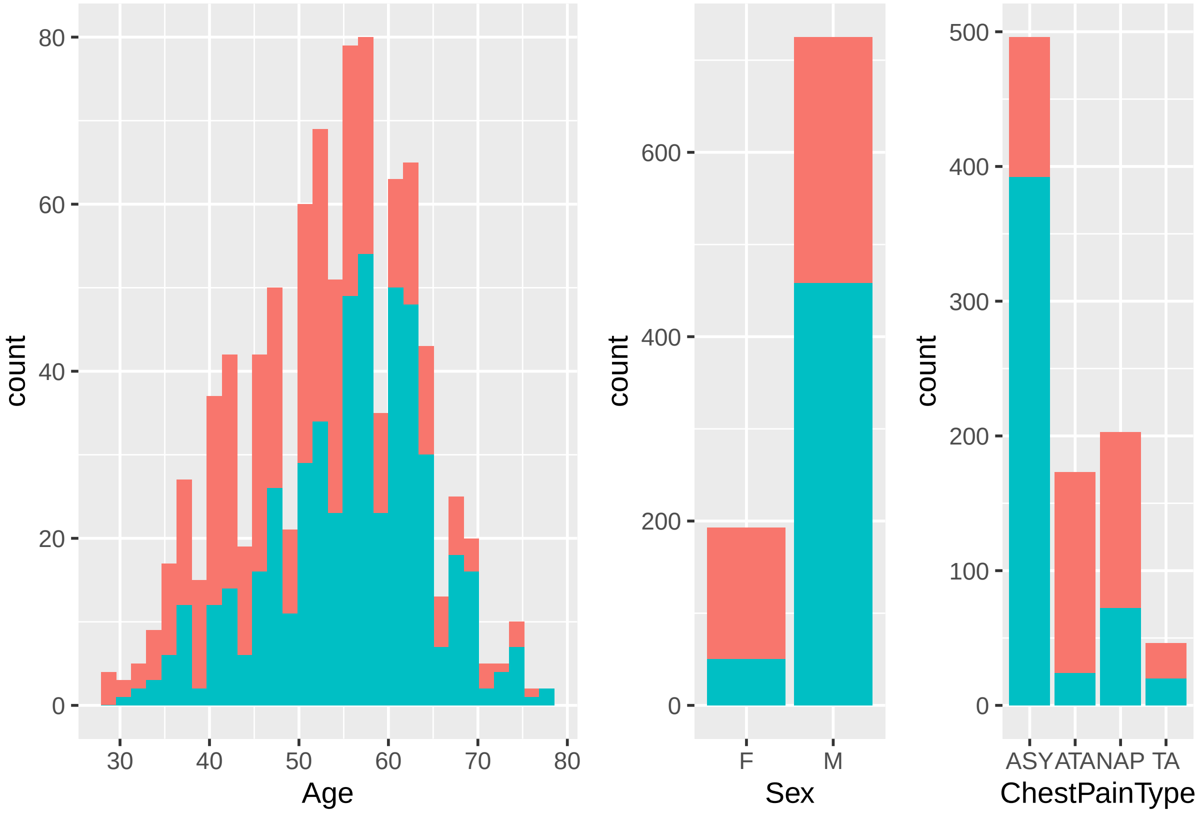
```r
heart_failure_dataset %>% corrgram(lower.panel = panel.pie ,upper.panel = panel.shade)
```



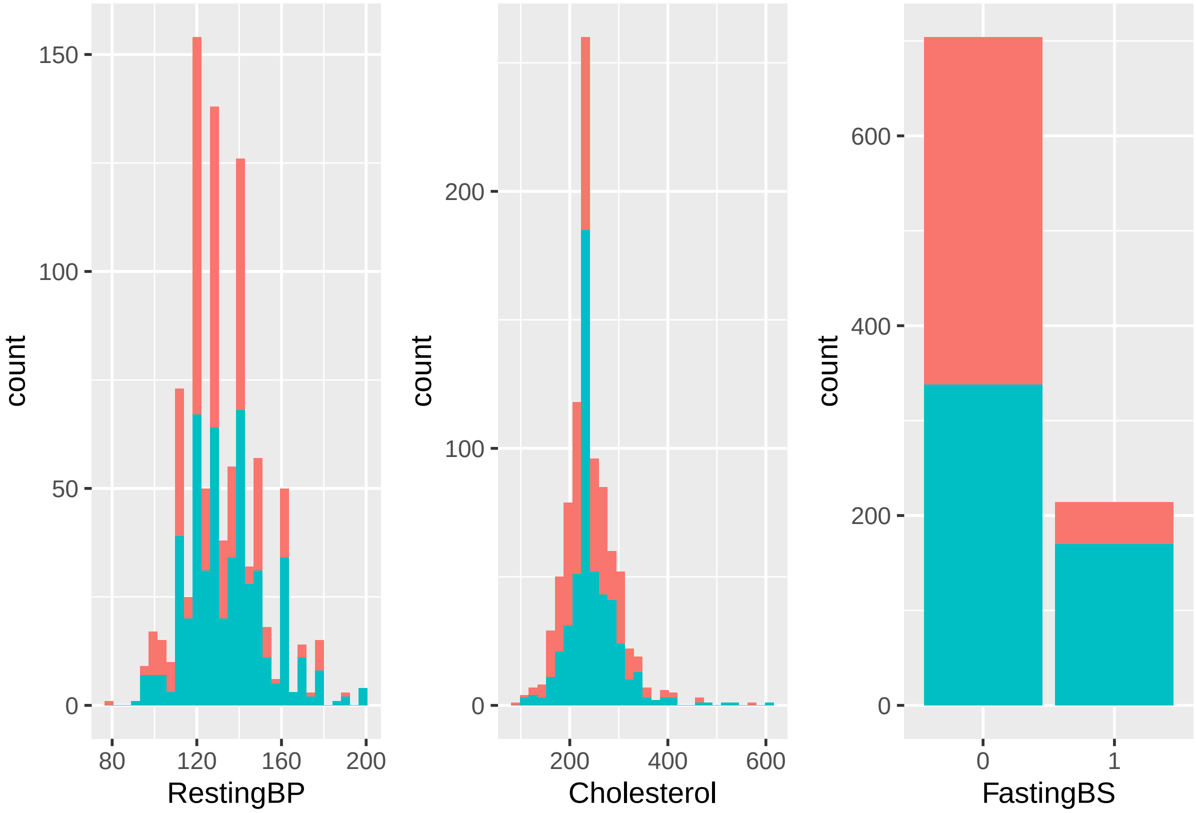
The correlogram indicates that the Oldpeak and the MaxHR has important correlation in determining if the person has high risk or not of heart disease.

To dive deep in this we goona analyse further the relation between the heart diseases and the other variables:

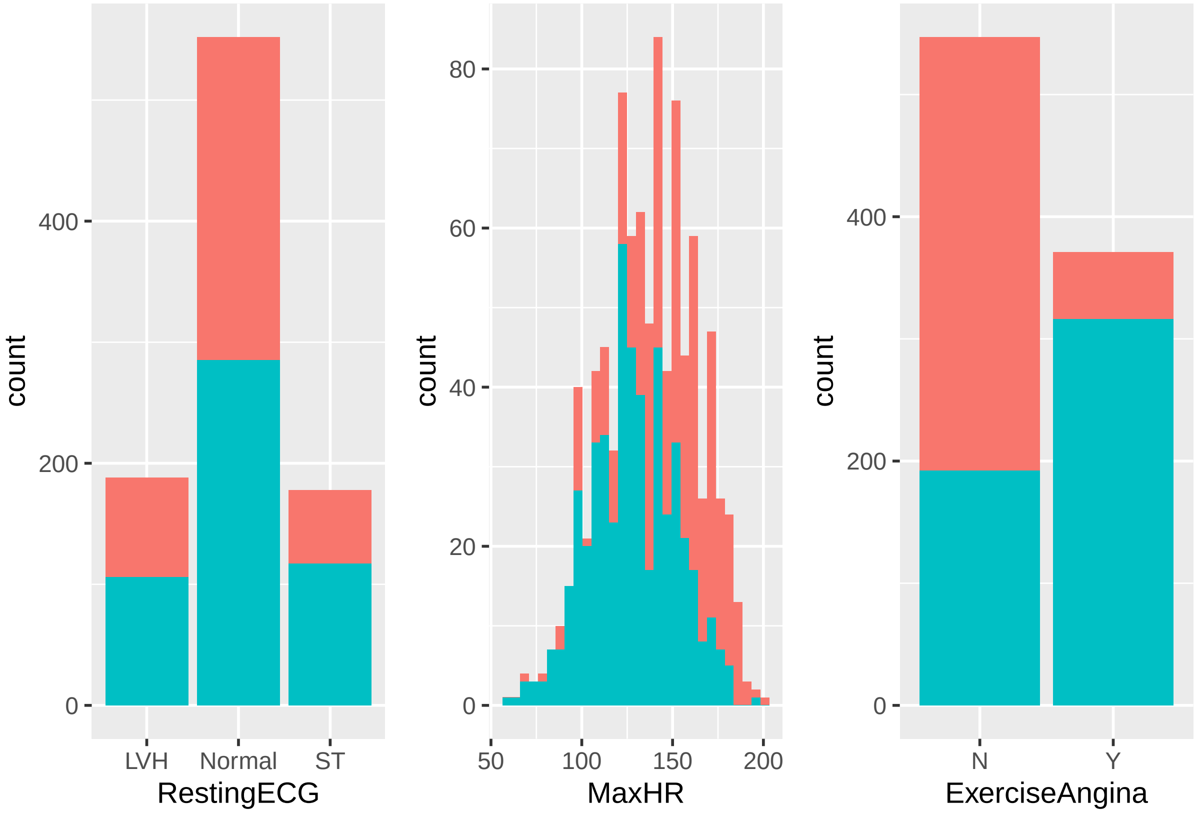
```
hd1 <- heart_failure_dataset %>% ggplot(aes(Age)) + geom_histogram(aes(fill = factor(Hea
hd2 <- heart_failure_dataset %>% ggplot(aes(Sex)) + geom_bar(aes(fill = factor(HeartDise
hd3 <- heart_failure_dataset %>% ggplot(aes(ChestPainType)) + geom_bar(aes(fill = factor
hd4 <- heart_failure_dataset %>% ggplot(aes(RestingBP)) + geom_histogram(aes(fill = fact
hd5 <- heart_failure_dataset %>% ggplot(aes(Cholesterol)) + geom_histogram(aes(fill = fa
hd6 <- heart_failure_dataset %>% ggplot(aes(FastingBS)) + geom_bar(aes(fill = factor(Hea
hd7 <- heart_failure_dataset %>% ggplot(aes(RestingECG)) + geom_bar(aes(fill = factor(He
hd8 <- heart_failure_dataset %>% ggplot(aes(MaxHR)) + geom_histogram(aes(fill = factor(H
hd9 <- heart_failure_dataset %>% ggplot(aes(ExerciseAngina)) + geom_bar(aes(fill = facto
hd10 <- heart_failure_dataset %>% ggplot(aes(Oldpeak)) + geom_histogram(aes(fill = facto
hd11 <- heart_failure_dataset %>% ggplot(aes(ST_Slope)) + geom_bar(aes(fill = factor(Hea

(hd1 + (hd2 + hd3))
```


count
80
60
40
20
0
30
40
50
60
70
80
Age
count
600
400
200
0
F
M
Sex
count
500
400
300
200
100
0
ASY
ATA
NAP
TA
ChestPainType

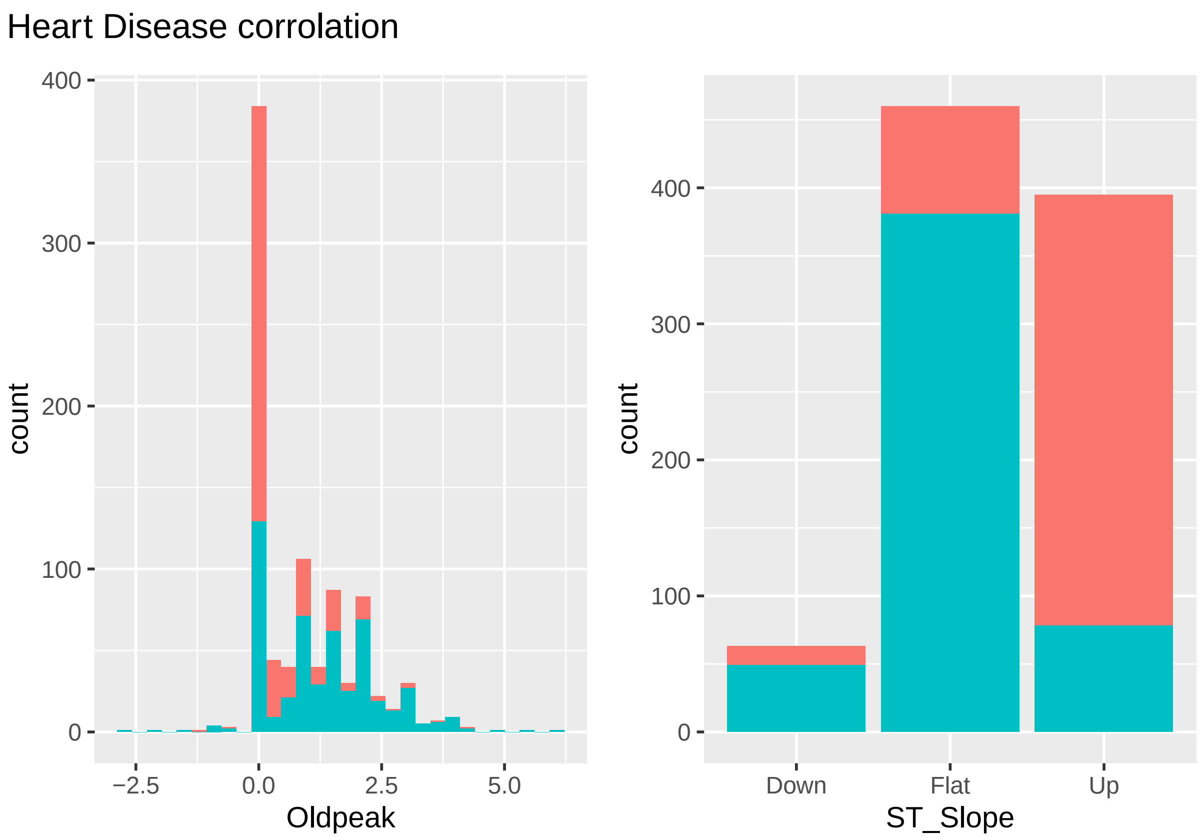
```
(hd4 + hd5 + hd6)
```


count
150
100
50
0
80
120
160
200
RestingBP
count
200
100
0
200
400
600
Cholesterol
count
600
400
200
0
0
1
FastingBS

```
(hd7 + hd8 + hd9)
```



```
(hd10 + hd11) +
  plot_annotation(title = "Heart Disease corrolation")
```



After comparing HeartDisease with other variable, we can conlcude that:

- the higher the Age the higher
- The males have higher risk
- ChestPainType ASY have higher risk)
- FastingBS >120 increase the risk)
- MaxHR the lower induce more risk)
- ExerciseAngina (Y = high risk)
- Having a ST_Slope Down or Flat is correlated with high risk

4. Results:

After cleaning then analysis our data_set we can build our Models: For this project we will use as recommended advanced methods of machine learning:

- Logistic Regression,
- Random Forest,
- Support Vector Machine,
- K-nearest neighbour & Neural Nets

### 4.1. Splitting train_set and test_set

we will create train set(80%) and HF_testset set (20%)

```
ind = createDataPartition(heart_failure_dataset$HeartDisease, times = 1, p = 0.8, list =
HF_trainset <- heart_failure_dataset[ind,]
HF_testset <- heart_failure_dataset[-ind,]
```

4.2. Logistic Regression Model:

In statistics, the logistic model (or logit model) is a statistical model that models the probability of an event taking place by having the log-odds for the event be a linear combination of one or more independent variables. In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model (the coefficients in the linear combination) (sources : Wikipedia)

```
 log.model <- glm(HeartDisease ~ ., data = HF_trainset, family = binomial(link = 'logit'
summary(log.model)
```

```
##
## Call:
## glm(formula = HeartDisease ~ ., family = binomial(link = "logit"),
##     data = HF_trainset)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8648  -0.4056   0.1979   0.5004   2.6305
##
## Coefficients: (10 not defined because of singularities)
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)             -3.204462   1.638311  -1.956 0.050470 .
## Age                      0.020976   0.014510   1.446 0.148285
## SexM                     1.807842   0.313315   5.770 7.93e-09 ***
## ChestPainTypeATA        -2.053748   0.370609  -5.542 3.00e-08 ***
## ChestPainTypeNAP        -1.527908   0.279351  -5.469 4.51e-08 ***
## ChestPainTypeTA         -1.815471   0.490899  -3.698 0.000217 ***
## RestingBP                0.007455   0.006703   1.112 0.266055
## Cholesterol              0.001341   0.002248   0.597 0.550653
## FastingBS1               1.099169   0.290900   3.779 0.000158 ***
## RestingECGNormal         0.080945   0.294050   0.275 0.783105
## RestingECGST            -0.143520   0.380640  -0.377 0.706136
## MaxHR                   -0.004240   0.005418  -0.783 0.433804
## ExerciseAnginaY          0.939439   0.263961   3.559 0.000372 ***
## Oldpeak                  0.328483   0.124973   2.628 0.008578 **
## ST_SlopeFlat             1.157674   0.458329   2.526 0.011542 *
## ST_SlopeUp              -1.095787   0.475257  -2.306 0.021129 *
## Age_impTRUE                    NA         NA      NA       NA
## Sex_impTRUE                    NA         NA      NA       NA
## ChestPainType_impTRUE          NA         NA      NA       NA
## RestingBP_impTRUE       13.091540 535.411277   0.024 0.980493
## Cholesterol_impTRUE            NA         NA      NA       NA
## FastingBS_impTRUE              NA         NA      NA       NA
## RestingECG_impTRUE             NA         NA      NA       NA
## MaxHR_impTRUE                  NA         NA      NA       NA
## ExerciseAngina_impTRUE         NA         NA      NA       NA
## Oldpeak_impTRUE                NA         NA      NA       NA
## ST_Slope_impTRUE               NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1011.26  on 734  degrees of freedom
## Residual deviance:  498.51  on 718  degrees of freedom
## AIC: 532.51
##
## Number of Fisher Scoring iterations: 12
```

```r
# Logistic Regression Model heart_failure_dataseting
glm.prediction <- predict(log.model, newdata = HF_testset, type = "response")
glm.prediction <- ifelse(glm.prediction >= 0.5, 1, 0)
#how accurate it is against our test data
table(HF_testset$HeartDisease, glm.prediction)
```

```
##    glm.prediction
##      0  1
##   0 69 11
##   1 10 93
```

```r
sum(HF_testset$HeartDisease==glm.prediction) / nrow(HF_testset)
```

```
## [1] 0.8852459
```

4.3.Random Forest Model :

Random forests or random decision forests is an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned. Random decision forests correct for decision trees' habit of overfitting to their training set. Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance (sources : Wikipedia)

```r
 rf.model <- randomForest(HeartDisease ~ ., data = HF_trainset)
importance(rf.model)
```

```
##                    IncNodePurity
## Age                  11.73613064
## Sex                   7.70909798
## ChestPainType        22.76728154
## RestingBP            11.52545038
## Cholesterol          11.58695076
## FastingBS             3.69124811
## RestingECG            3.87447166
## MaxHR                16.27487008
## ExerciseAngina       16.17785371
## Oldpeak              18.07949265
## ST_Slope             40.54914897
## Age_imp               0.00000000
## Sex_imp               0.00000000
## ChestPainType_imp     0.00000000
## RestingBP_imp         0.08216329
## Cholesterol_imp       0.00000000
## FastingBS_imp         0.00000000
## RestingECG_imp        0.00000000
## MaxHR_imp             0.00000000
## ExerciseAngina_imp    0.00000000
## Oldpeak_imp           0.00000000
## ST_Slope_imp          0.00000000
```

```
rf.prediction <- predict(rf.model, newdata = HF_testset)
rf.prediction <- ifelse(rf.prediction >= 0.5, 1, 0)

table(HF_testset$HeartDisease, rf.prediction)
```

```
##    rf.prediction
##      0  1
##   0 69 11
##   1  9 94
```

```
sum(HF_testset$HeartDisease==rf.prediction) / nrow(HF_testset)
```

```
## [1] 0.8907104
```

4.4. Support Vector Machine (linear and radial kernel):

In machine learning, support vector machines (SVMs, also support vector networks) are supervised learning models with associated learning algorithms that analyze data for classification and regression analysis. Developed at AT&T Bell Laboratories by Vladimir Vapnik with colleagues. SVMs are one of the most robust prediction methods, being based on statistical learning frameworks or VC theory proposed by Vapnik (1982, 1995) and Chervonenkis (1974). Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier (although methods such as Platt scaling exist to use SVM in a probabilistic classification setting). SVM maps training examples to points in space so as to maximise the width of the gap between the two categories. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall. (sources : Wikipedia)

```
tune.svm.linear <- tune.svm(HeartDisease ~ ., data = HF_trainset, kernel = "linear" , co
tune.svm.radial <- tune.svm(HeartDisease ~ ., data = HF_trainset, kernel = "radial" , co
svm.linear <- svm(HeartDisease ~ ., data = HF_trainset, kernel = "linear", cost = 0.01, 
svm.radial <- svm(HeartDisease ~ ., data = HF_trainset, kernel = "radial", cost = 1, gam

svm.linear.pred <- predict(svm.linear, newdata = HF_testset)
svm.linear.pred <- ifelse(svm.linear.pred >= 0.5, 1, 0)
table(HF_testset$HeartDisease,svm.linear.pred)
```

```
##    svm.linear.pred
##      0  1
##   0 64 16
##   1 11 92
```

```
sum(HF_testset$HeartDisease == svm.linear.pred)/nrow(HF_testset)
```

```
## [1] 0.852459
```

```
svm.radial.pred <- predict(svm.radial, newdata = HF_testset)
svm.radial.pred <- ifelse(svm.radial.pred >= 0.5, 1, 0)
table(HF_testset$HeartDisease,svm.radial.pred)
```

```
##    svm.radial.pred
##      0  1
##   0 70 10
##   1 11 92
```

```
sum(HF_testset$HeartDisease == svm.radial.pred)/nrow(HF_testset)
```

```
## [1] 0.8852459
```

The following is the list of results of all 3 models tested:

1) Logistic Regression Model : 0.885
2) Random Forest Model : 0.890
3) Support Vector Machine (radial) Model : 0.852
4) Support Vector Machine (linear) Model : 0.885

5. Conclusion:

The main goal of this project is to explore variables that may be related to heart failure in order to predict it using several techniques of Machine Learning.

We cleaned and analysed the data_set, and then we used several techniques of Machine Learning to build a model focused on maximizing the true predictions of Heart failure.

The machine-learning models which used to predict the heart failure are: Logistic Regression, Forest Model, Support Vector Machine.

The Forest Model can be seen as the best model.

Many thanks to Rafael Irizarry, the course instructor of HarvardX's Professional Certificate in Data Science, and to the teaching staff who were always at hand to answer questions and queries raised by students.

This edX series has been extreamly valuable. Irizarry delivered engaging lectures and provided a range of useful coding examples throughout the series.