Data Science Capstone Project CYO

Muhammad Ameer Hamza

4/19/2020

Introduction

Heart disease is one of the most common and widespread diseases in the world. According to the estimate of World Health Organization twelve million deaths occur world wide due to heart diseases. Half the deaths in the United States and other developed countries are due to cardio vascular diseasesThe early prognosis of cardiovascular diseases can aid in making decisions on lifestyle changes in high risk patients and in turn reduce the complications. This research intends to pinpoint the most relevant/risk factors of heart disease as well as predict the overall risk

Some attempts will be made to find out whether there is a machine learning technique that identifies those patients who are more likely to be diagnoed with coronary heart disease in next ten years.

Overview

Data set:

The dataset is publically available on the Kaggle website, and it is from an ongoing cardiovascular study on residents of the town of Framingham, Massachusetts. The classification goal is to predict whether the patient has 10-year risk of future coronary heart disease (CHD).The dataset provides the patients' information. It includes over 4,000 records and 15 attributes. Each attribute is a potential risk factor.

Variables:

Following is an overview of the variables used and method used:

- framingham : original dataset as available on Kaggle
- fullData: data frame used for analysis. It is derived from framingham after removal of NAs and columns not required.
- test: data frame to check accuracy of the models
- train: data frame to train models.

Method:

After division into test and train data frames, we will be analysing all the potential risk factors one after another to generate insights of the data. Once insights are generated, following machine learning models will used to generate confusin matrices which will then be compared:

- Naive Bayes
- Linear Classifier
- Logistic Regression
- K-Nearest Neighbours
- Random Forest

Data Analysis:

Workspace

First we are going to download the libraries that will be used:

```
#Downloading libraries#

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

```
## Loading required package: tidyverse
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.0     v purrr   0.3.3
## v tibble  3.0.0     v dplyr   0.8.5
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

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

```
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
```

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

```
## Loading required package: RCurl
```

```
##
## Attaching package: 'RCurl'
```

```
## The following object is masked from 'package:tidyr':
##
##     complete
```

```
# Loading libraries#

library(caret)
library(tidyverse)
library(dplyr)
library(RCurl)
```

Data Exploration

Downloading and Splitting the Data

The first step is to download the data and import it into a data frame. The csv file is linked from a github repository. Follwing chunk of code will download the csv file and import into a data frame.

```
x <- getURL("https://raw.githubusercontent.com/mhmdamrhamza/dscapstoneharvardxcyo/master/framingham.csv")
framingham <- read.csv(text = x)
```

Cleansing the Data

To perform analysis correctly, some adjustments are made:

- The data contains education and Smoking column. These columns are removed as education is not a potential risk factor. Also, smoking column has values 1 for smoker and 0 for non-smokers. Rather than using this column we will be using cigsperday column which has better input and easier to process.
- Male column has entries 1 for males and 0 for females. We will be adding sex column having factors male and female
- All rows with NA values will be removed to execute analysis without any errors.
- TenYearCHD is set as parameter to evaluate performance.

```
#removing education and smoking column#

fullData <- framingham[,-c(3,4)]

#adding sex column#

fullData$sex <- ifelse(fullData$male == 1, "male", "female")

#removing rows with NA values in any of the columns#

fullData <- subset(fullData, complete.cases(fullData))


#defining result parameter#

fullData <- fullData %>%
  mutate(TenYearCHD = as.character(TenYearCHD)) %>%
  mutate(TenYearCHD = replace(TenYearCHD, TenYearCHD == '0', 'No')) %>%
  mutate(TenYearCHD = replace(TenYearCHD, TenYearCHD == '1', 'Yes')) %>%
  mutate(TenYearCHD = as.factor(TenYearCHD))
```

Next, the data is divided into test and train sets:

```
#generating test and train sets#

set.seed(1)
trainIndex <- createDataPartition(fullData$TenYearCHD, p=.7, list=FALSE)
train <- fullData[trainIndex,]
test <- fullData[-trainIndex,]
```

To have an idea of the outcomes we will list down the number of positive and negative outcomes:

```
## # A tibble: 2 x 2
##   TenYearCHD     n
##   <fct>      <int>
## 1 No          3177
## 2 Yes          572
```

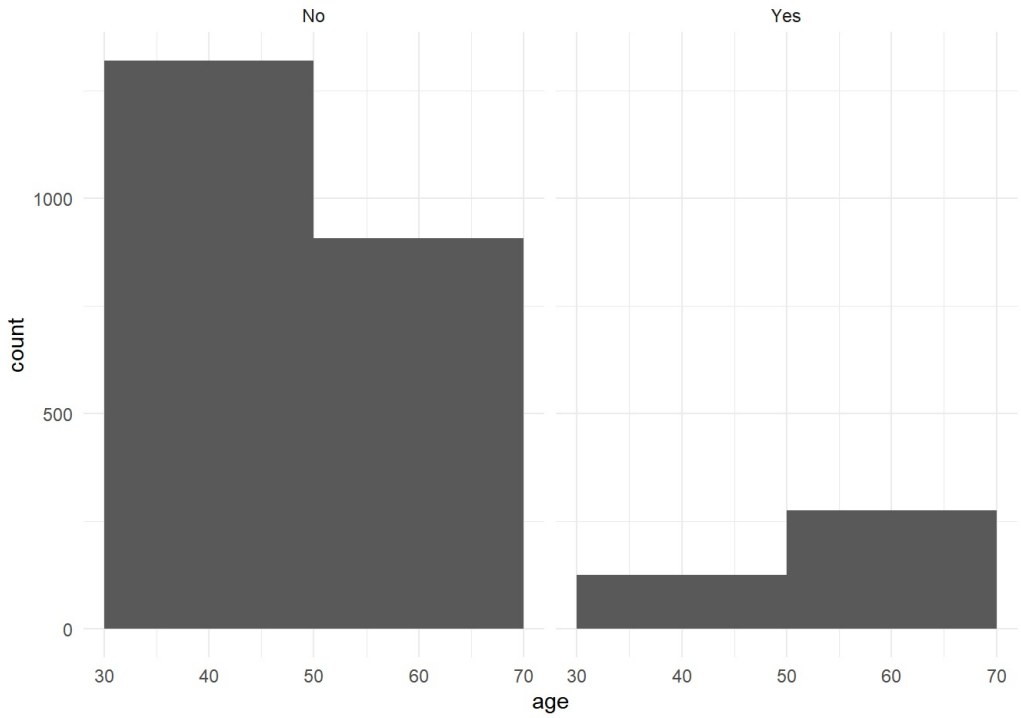
As it is clear that number of negative outcomes is significantly greater, we will be focusing more on the patterns of each risk factor.

Exploration of each potential risk factor

Age

Age is on of the most important potential risk factor for majority of the diseases:

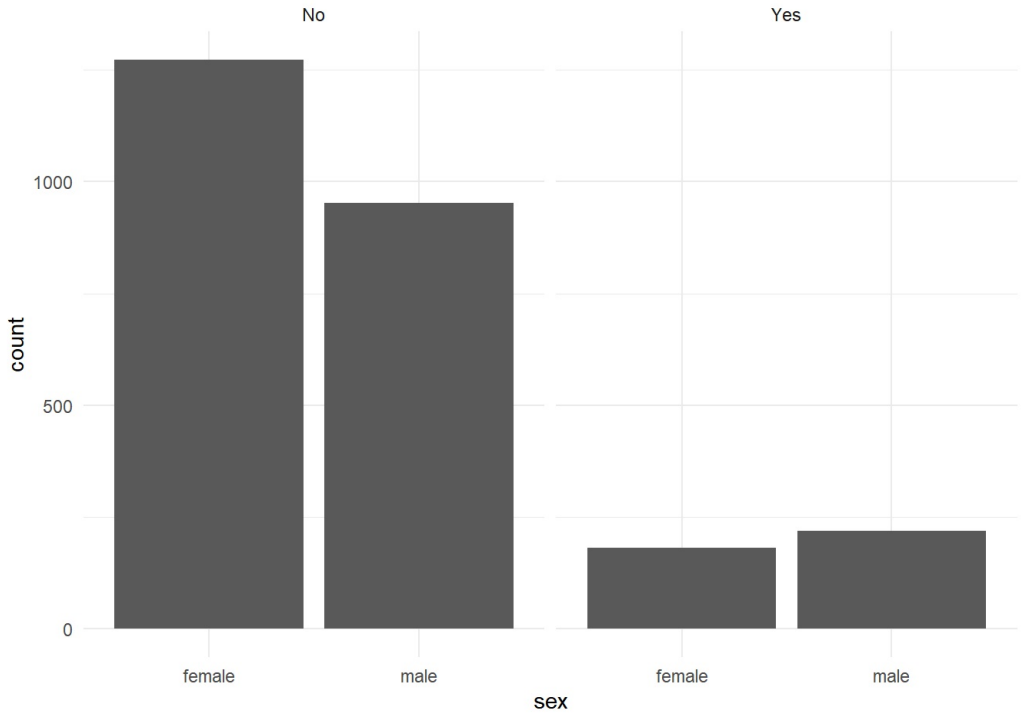
```
train %>%
  ggplot(aes(x = age)) +
    geom_histogram(binwidth = 20) +
    theme_minimal() +
    facet_grid(~ TenYearCHD)
```



The graph shows age has a direct impact on the risk of the diseases. For group wtih negative outcome (TenYearCHD = No) higher age group has less count implying that aged people have more chances of heart disease. For other group, the distribution is different implying the same outcome.

Sex

```
train %>%
  ggplot(aes(x = sex)) +
    geom_bar() +
    theme_minimal() +
    facet_grid(~ TenYearCHD)
```

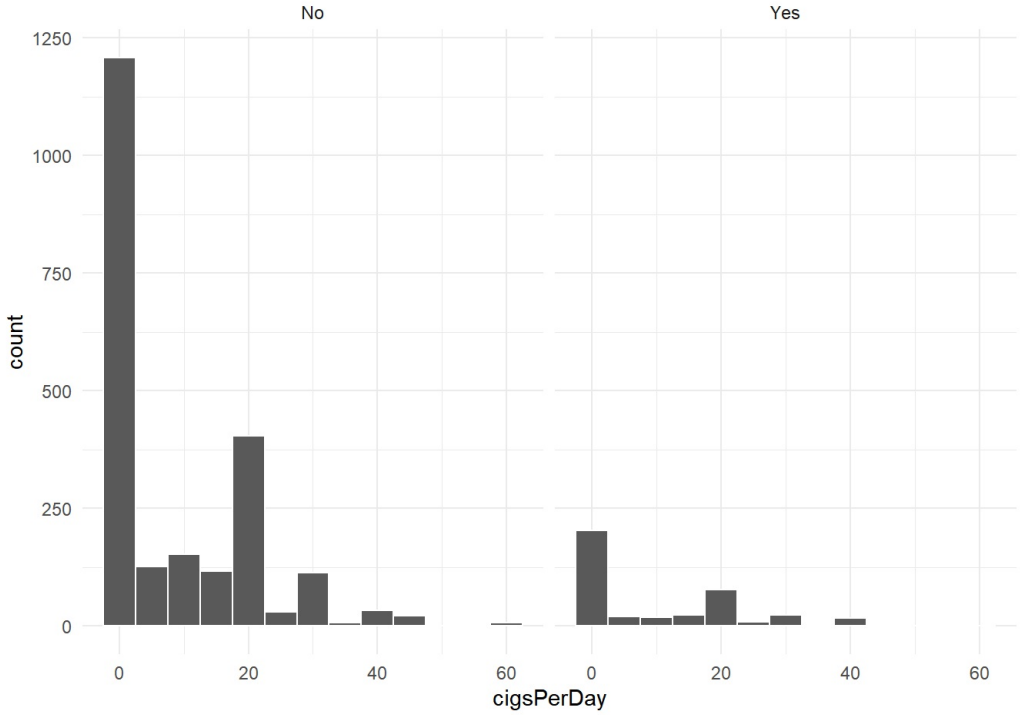


It seems that women appear less frequently in positive outcome groups and more frequently in negative outcome group. Both these observations imply that women are less likely to be tested positive in ten years.

Smoking

Heart dieseases are the second most known disease attributed with the smoking after lungs disorder:

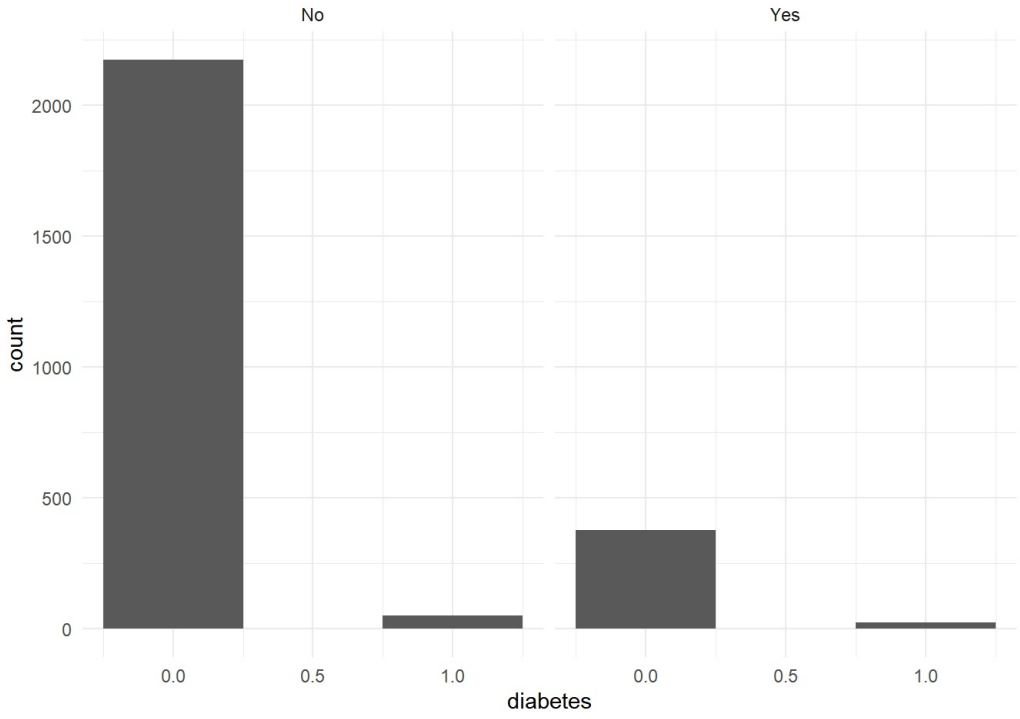
```
train %>%
  ggplot(aes(x = cigsPerDay)) +
    geom_histogram(binwidth = 5, color = "white") +
    theme_minimal() +
    facet_grid(~ TenYearCHD)
```



Not surprisingly in both conditions, people with more number of cigerettes per day are associated with the risk more as compared to those who smoked small number of cigeretts or no cigerettes at all.

Diabetes

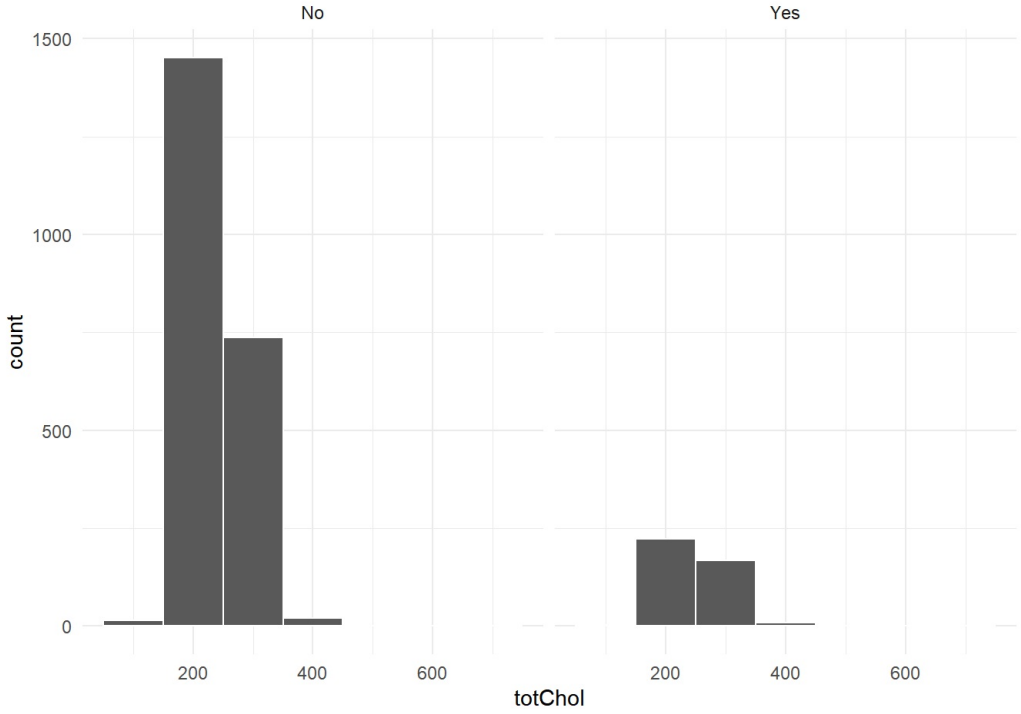
```
train %>%
  ggplot(aes(x = diabetes)) +
    geom_histogram(binwidth = 0.5) +
    theme_minimal() +
    facet_grid(~ TenYearCHD)
```



Since value in diabetes column is 0 for no and 1 for yes, high count of zero in outcome ‘no’ column shows that people with diabetes are more likely to have heart disease in ten years. Same is the observation for outcome ‘yes’ i.e. count of people having diabetes and associated to high chance of heart disease is high.

Cholestrol

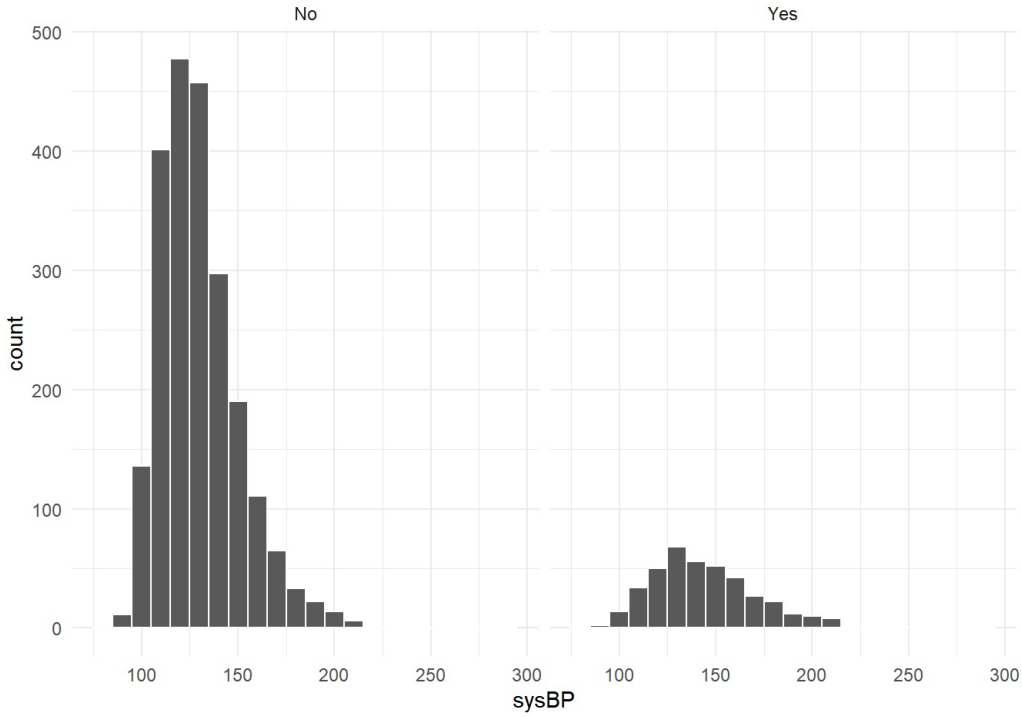
```
train %>%
  ggplot(aes(x = totChol)) +
    geom_histogram(binwidth = 100, color = "white") +
    theme_minimal() +
    facet_grid(~ TenYearCHD)
```



Both groups of people follow same pattern although difference cholestrol value for ‘outcome = yes’ group is higher.

Systolic Bloodpressure:

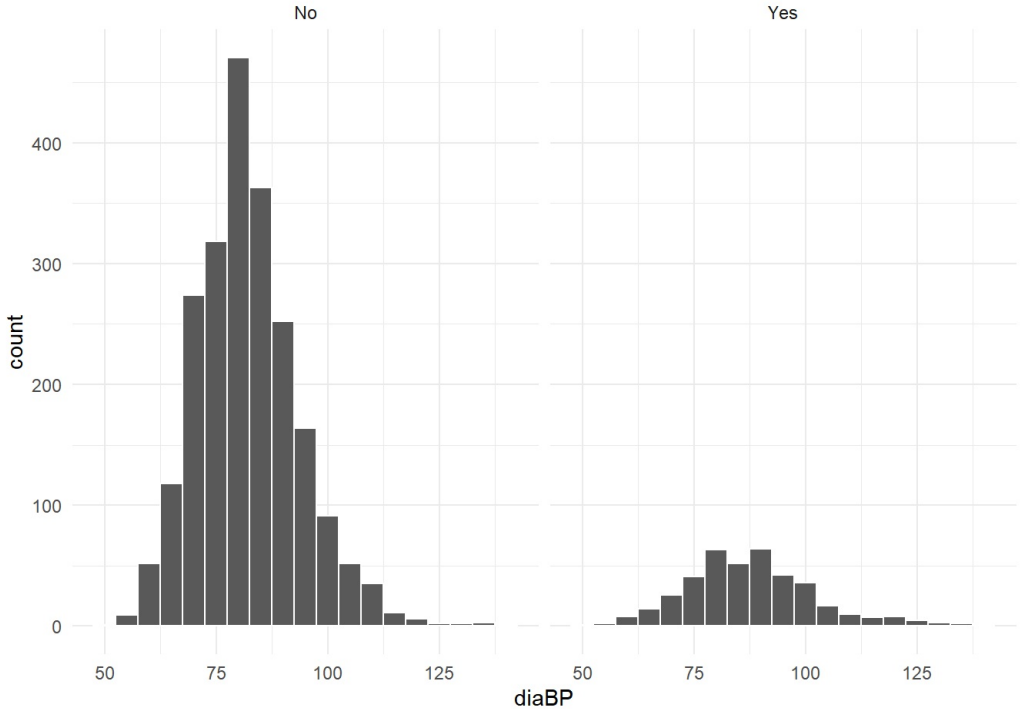
```
train %>%
  ggplot(aes(x = sysBP)) +
    geom_histogram(binwidth = 10, color = "white") +
    theme_minimal() +
    facet_grid(~ TenYearCHD)
```



The distribution is skewed left and have similar patterns for both outcome groups.

Diastolic Blood Pressure

```
train %>%
  ggplot(aes(x = diaBP)) +
    geom_histogram(binwidth = 5, color = "white") +
    theme_minimal() +
    facet_grid(~ TenYearCHD)
```

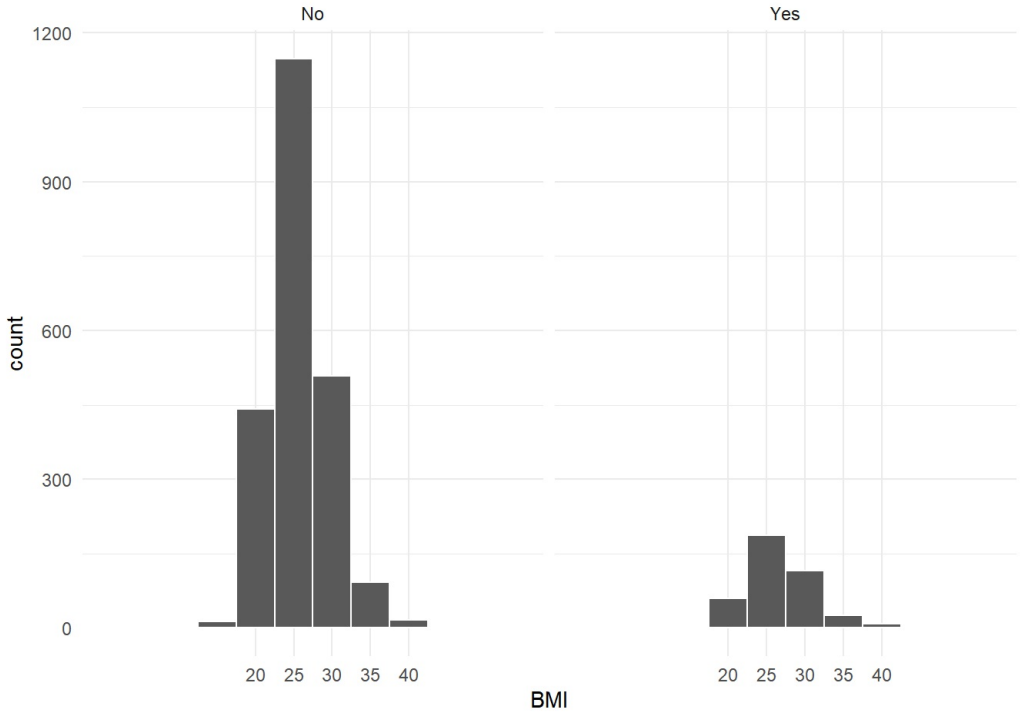


Once again both outcome groups have same same distribution and no significant effect of the parameter is observed.

BMI:

Body mass index is a measure of obesity. A BMI value of more than 30 means the person is obese. Obesity is widely associated with heart diseases:

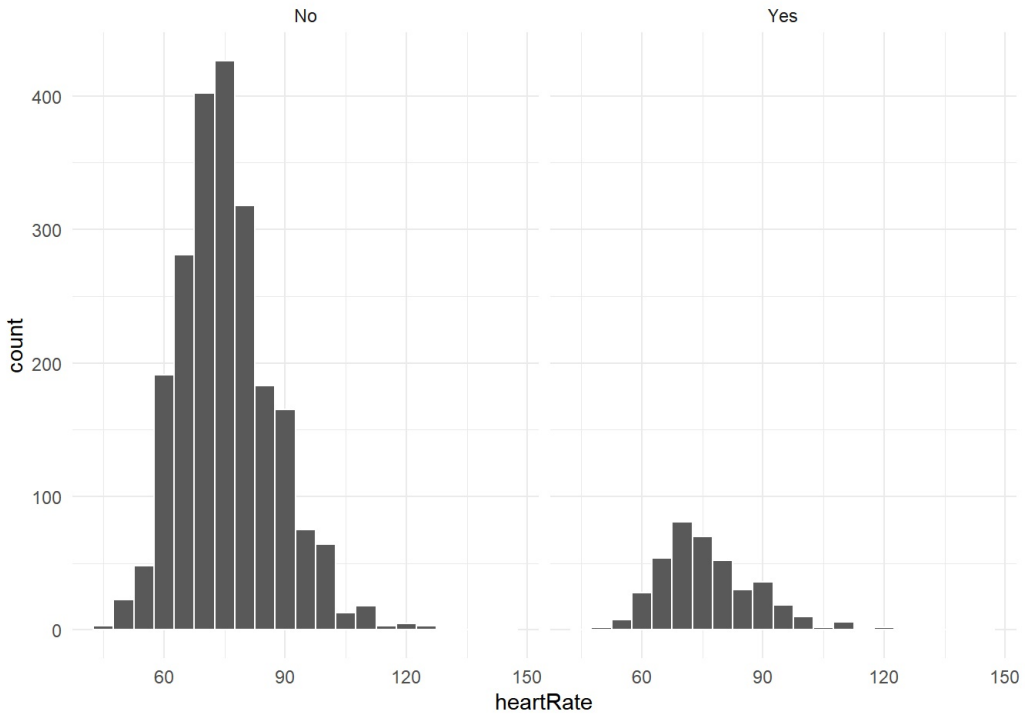
```
train %>%
  ggplot(aes(x = BMI)) +
    geom_histogram(binwidth = 5, color = "white") +
  scale_x_discrete(limits = c(seq(20,40,5))) +
    theme_minimal() +
    facet_grid(~ TenYearCHD)
```



Persons with BMI around twenty-five has the most chances of being diagnosed with the heart disease. This needs to be explored more.

HeartRate:

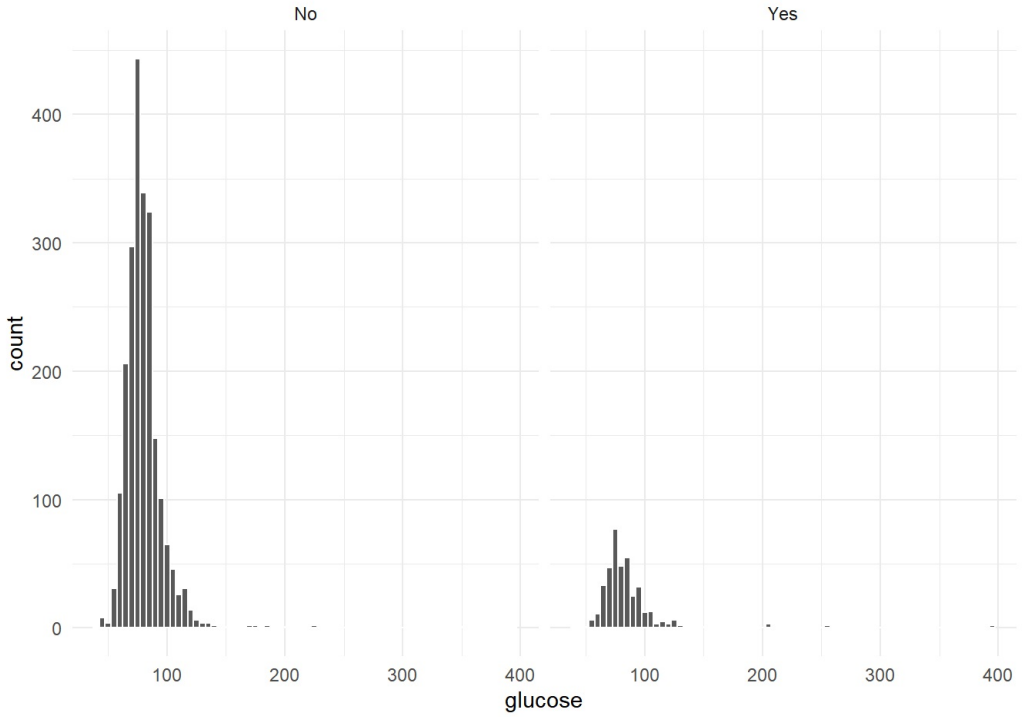
```
train %>%
  ggplot(aes(x = heartRate)) +
    geom_histogram(binwidth = 5, color = "white") +
    theme_minimal() +
    facet_grid(~ TenYearCHD)
```



Once again, normal distribution is observed in the both outcome sets.

Glucose

```
train %>%
  ggplot(aes(x = glucose)) +
    geom_histogram(binwidth = 5, color = "white")  +
    theme_minimal() +
    facet_grid(~ TenYearCHD)
```



Both sets show similar distribution

indicating that glucose might not have too significant effect on the prediction.

Clearly there are outliers in the data as well which we will be removing in the coming part.

Data Preparation

From the description of the data above it is evident that some predictors are most likely correlated for example usually for a patient both kinds of blood pressure are equally varied. By excluding categorical columns we can quickly find correleations as follows:

```
cor(subset(train, select = -c(sex, TenYearCHD,male,age)))
```

```
##                  cigsPerDay      BPMeds prevalentStroke prevalentHyp
## cigsPerDay       1.00000000 -0.05489931     -0.04117553  -0.08491094
## BPMeds          -0.05489931  1.00000000      0.14127539   0.26760525
## prevalentStroke -0.04117553  0.14127539      1.00000000   0.09182765
## prevalentHyp    -0.08491094  0.26760525      0.09182765   1.00000000
## diabetes        -0.02401496  0.04613646      0.01345869   0.07418360
## totChol         -0.02961859  0.09403313      0.02992410   0.16080024
## sysBP           -0.11559759  0.27762951      0.08421140   0.69402269
## diaBP           -0.06534743  0.19586779      0.07099070   0.61671289
## BMI             -0.10509428  0.11291053      0.05708226   0.31993266
## heartRate        0.06052876  0.02373722     -0.01859039   0.14650821
## glucose         -0.04956850  0.05508546      0.02027462   0.08239137
##                    diabetes     totChol       sysBP       diaBP         BMI
## cigsPerDay      -0.02401496 -0.02961859 -0.11559759 -0.06534743 -0.10509428
## BPMeds           0.04613646  0.09403313  0.27762951  0.19586779  0.11291053
## prevalentStroke  0.01345869  0.02992410  0.08421140  0.07099070  0.05708226
## prevalentHyp     0.07418360  0.16080024  0.69402269  0.61671289  0.31993266
## diabetes         1.00000000  0.03837966  0.08370725  0.02797652  0.05949606
## totChol          0.03837966  1.00000000  0.20504563  0.16892392  0.12699449
## sysBP            0.08370725  0.20504563  1.00000000  0.79213011  0.33691841
## diaBP            0.02797652  0.16892392  0.79213011  1.00000000  0.39140306
## BMI              0.05949606  0.12699449  0.33691841  0.39140306  1.00000000
## heartRate        0.06887305  0.08881872  0.18578187  0.16820206  0.06463631
## glucose          0.60597427  0.05591973  0.12027978  0.05101439  0.06235936
##                  heartRate     glucose
## cigsPerDay       0.06052876 -0.04956850
## BPMeds           0.02373722  0.05508546
## prevalentStroke -0.01859039  0.02027462
## prevalentHyp     0.14650821  0.08239137
## diabetes         0.06887305  0.60597427
## totChol          0.08881872  0.05591973
## sysBP            0.18578187  0.12027978
## diaBP            0.16820206  0.05101439
## BMI              0.06463631  0.06235936
## heartRate        1.00000000  0.10019763
## glucose          0.10019763  1.00000000
```

from the data above, the parameters with high correlated predictors are:

- sysBP and diaBP

```
train %>%
  ggplot(aes(x = sysBP, y = diaBP, color = prevalentHyp, shape = TenYearCHD,  )) +
    geom_point() +
    theme_minimal() +
    facet_grid(~ TenYearCHD)
```

Both groups have same correlation with outliers in the similar range. For this reason one of the parameter can be omitted from our model.

```
train <- train %>% subset(select = -c(sysBP))
test <- test %>% subset(select = -c(sysBP))
```

Machine Learning Methods

For developing machine learning models, there are some signs of correlations. The aim of this project is to correctly predict the diagnoses outcome for the people.

Sensitivity (the proportion of patients who were in the disease group correctly identified) and specificity (the proportion of actual negatives, which are healthypatients, identified as such) are the two other parameters which can be used to assess the accuracy of the model.

We are going to save results after each model to allow comparisons:

```
results <- data.frame(Model = character(),
                      Accuracy = double(),
                      Sensitivity = double(),
                      Specificity = double(),
                      stringsAsFactors = FALSE)
```

Naive Bayes

Naive Bayes is one of the most common models. It assumes This calculates a series of conditional probabilities, and the probabilities of each possible outcome.

```
nb_model = train(TenYearCHD ~ ., data = train, method = "nb")
predictions = predict(nb_model, newdata = test)
confusionMatrix <- confusionMatrix(predictions, test$TenYearCHD)
results[nrow(results) + 1, ] <- c(as.character('Naive Bayes (nb)'),
                                  confusionMatrix$overall['Accuracy'],
                                  confusionMatrix$byClass['Sensitivity'],
                                  confusionMatrix$byClass['Specificity'])
rm(nb_model, predictions)
confusionMatrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  951 167
##        Yes   2   4
##
##                Accuracy : 0.8496
##                  95% CI : (0.8274, 0.87)
##     No Information Rate : 0.8479
##     P-Value [Acc > NIR] : 0.4542
##
##                   Kappa : 0.0352
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.99790
##             Specificity : 0.02339
##          Pos Pred Value : 0.85063
##          Neg Pred Value : 0.66667
##              Prevalence : 0.84786
##          Detection Rate : 0.84609
##    Detection Prevalence : 0.99466
##       Balanced Accuracy : 0.51065
##
##        'Positive' Class : No
##
```

Linear Classifier

```
lc_model = train(TenYearCHD ~ ., data = train, method = "glmboost")
predictions = predict(lc_model, newdata = test)
confusionMatrix <- confusionMatrix(predictions, test$TenYearCHD)
results[nrow(results) + 1, ] <- c(as.character('Linear Classifier (glmboost)'),
                                  confusionMatrix$overall['Accuracy'],
                                  confusionMatrix$byClass['Sensitivity'],
                                  confusionMatrix$byClass['Specificity'])
rm(lc_model, predictions)
confusionMatrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  951 170
##        Yes   2   1
##
##                Accuracy : 0.847
##                  95% CI : (0.8246, 0.8675)
##     No Information Rate : 0.8479
##     P-Value [Acc > NIR] : 0.5533
##
##                   Kappa : 0.0063
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.997901
##             Specificity : 0.005848
##          Pos Pred Value : 0.848350
##          Neg Pred Value : 0.333333
##              Prevalence : 0.847865
##          Detection Rate : 0.846085
##    Detection Prevalence : 0.997331
##       Balanced Accuracy : 0.501875
##
##        'Positive' Class : No
##
```

Although it has almost the same accuracy as Naive Bayes model, the confusion marix also shows a significant number of patients predicted yes when they are no.

Here, the sensitivity (the number of patients correctly identified) was close to 1 (which would otherwise be excellent), but the sensitivity (those without the disease categorised as such) was close to zero.

In healthcare systems it is an error to predict care for a patient when it is not required.

The other problem is that in these data, the group with negative outcome are larger than the those preicted ‘yes’. This gives the unusually large accuracy metric.

Logistic Regression

Logistic regression is another very common machine learning approach . Using the Bayesian Generalized Linear Model, and including the actual prevalence in the confusion matrix we can generate results as follows:

```
lr_model = train(TenYearCHD ~ ., data = train, method = "bayesglm")
predictions = predict(lr_model, newdata = test)
confusionMatrix <- confusionMatrix(predictions, test$TenYearCHD, prevalence = 0.06)
results[nrow(results) + 1, ] <- c(as.character('Logistic Regression (bayesglm)'),
                                  confusionMatrix$overall['Accuracy'],
                                  confusionMatrix$byClass['Sensitivity'],
                                  confusionMatrix$byClass['Specificity'])
rm(lr_model, predictions)
confusionMatrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  949 168
##        Yes   4   3
##
##                Accuracy : 0.847
##                  95% CI : (0.8246, 0.8675)
##     No Information Rate : 0.8479
##     P-Value [Acc > NIR] : 0.5533
##
##                   Kappa : 0.022
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.99580
##             Specificity : 0.01754
##          Pos Pred Value : 0.06077
##          Neg Pred Value : 0.98496
##              Prevalence : 0.06000
##          Detection Rate : 0.84431
##    Detection Prevalence : 0.99377
##       Balanced Accuracy : 0.50667
##
##        'Positive' Class : No
##
```

This shows a further improvement. The PPV of 0.06 is lower, but the NPV of 0.99 represents a far larger group of patients

K-Nearest Neighbours

The next method to consider is the “Nearest Neighbours”, here the K-nearest neighbours. This will find the k closest matching points from the training data. These will have their results averaged to find the predicted outcome.

This is a powerful idea, as the nearest neighbours to any patient will be patients with similar profiles (demographics and blood results), so may be more likely to have the same outcome.

The value of K is a tuning parameter, and the following code will attempt to find the most appropriate value.

```
knn_model = train(TenYearCHD ~ ., data = train, method = "knn", preProcess=c('knnImpute'))
knn_model
```

```
## k-Nearest Neighbors
##
## 2625 samples
##   13 predictor
##    2 classes: 'No', 'Yes'
##
## Pre-processing: nearest neighbor imputation (13), centered (13), scaled (13)
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 2625, 2625, 2625, 2625, 2625, 2625, ...
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
##   5  0.7992475  0.09073831
##   7  0.8130868  0.08488891
##   9  0.8195830  0.05701132
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 9.
```

The value of 9 is settled on here. It's worth noting that in a more widespread deployment, where more pleniful training data are available, this could be different.

```
predictions = predict(knn_model, newdata = test)
confusionMatrix <- confusionMatrix(predictions, test$TenYearCHD, prevalence = 0.06)
results[nrow(results) + 1, ] <- c(as.character('K-nearest neighbours (knn)'),
                                  confusionMatrix$overall['Accuracy'],
                                  confusionMatrix$byClass['Sensitivity'],
                                  confusionMatrix$byClass['Specificity'])
rm(knn_model, predictions)
confusionMatrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  937 166
##        Yes  16   5
##
##                Accuracy : 0.8381
##                  95% CI : (0.8152, 0.8592)
##     No Information Rate : 0.8479
##     P-Value [Acc > NIR] : 0.8305
##
##                   Kappa : 0.0195
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.98321
##             Specificity : 0.02924
##          Pos Pred Value : 0.06072
##          Neg Pred Value : 0.96465
##              Prevalence : 0.06000
##          Detection Rate : 0.83363
##    Detection Prevalence : 0.98132
##       Balanced Accuracy : 0.50623
##
##        'Positive' Class : No
##
```

Despite a more realistic range of predictions, there is no improvement to the accuracies, sensitivity or specificity.

Random Forest

The final tested approach is a random forest. This will create a "forest" of randomly chosen decision trees, which result in a classification.

The results of these trees will then be compared, and the best performing tree selected.

```
rf_model = train(TenYearCHD ~ ., data = train, method = "rf")
predictions = predict(rf_model, newdata = test)
confusionMatrix <- confusionMatrix(predictions, test$TenYearCHD, prevalence = 0.06)
results[nrow(results) + 1, ] <- c(as.character('Random Forest (rf)'),
                                  confusionMatrix$overall['Accuracy'],
                                  confusionMatrix$byClass['Sensitivity'],
                                  confusionMatrix$byClass['Specificity'])
rm(rf_model, predictions)
confusionMatrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  952 170
##        Yes   1   1
##
##                Accuracy : 0.8479
##                  95% CI : (0.8255, 0.8684)
##     No Information Rate : 0.8479
##     P-Value [Acc > NIR] : 0.5204
##
##                   Kappa : 0.0081
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.998951
##             Specificity : 0.005848
##          Pos Pred Value : 0.060272
##          Neg Pred Value : 0.988676
##              Prevalence : 0.060000
##          Detection Rate : 0.846975
##    Detection Prevalence : 0.998221
##       Balanced Accuracy : 0.502399
##
##        'Positive' Class : No
##
```

```
rm(confusionMatrix)
```

Results

```
results %>% arrange(Accuracy)
```

```
##                            Model          Accuracy       Sensitivity
## 1     K-nearest neighbours (knn) 0.838078291814947 0.983210912906611
## 2   Linear Classifier (glmboost) 0.846975088967972 0.997901364113326
## 3 Logistic Regression (bayesglm) 0.846975088967972 0.995802728226653
## 4             Random Forest (rf) 0.847864768683274 0.998950682056663
## 5               Naive Bayes (nb) 0.849644128113879 0.997901364113326
##           Specificity
## 1  0.0292397660818713
## 2 0.00584795321637427
## 3  0.0175438596491228
## 4 0.00584795321637427
## 5  0.0233918128654971
```

As we have a table of the all the methods used and their outcomes, a good practice is to consider the positives and the negatives of each combination of the results:

Models that are highly sensitive, but less specific (the linear classifier, logistic regression and k-nearest neighbours) would mean patients are very likely to be flagged correctly however many patients who do not have significant risk will also be flagged.

Models with high speficicity but low sensitivity (none of these models behaved this way) would be good for identifying patients who are very likely to be suffering with the coronary heart disease. A ‘hit’ on a tool like this would indicate that the patient is almost certainly in the group with high chance of heart disease diagnosed in ten years.

Models (like Naive Bayes), which perform well in both sensitivity and specificty provide a compromise. Not all patients will be spotted correctly , but there will be fewer patients missed who may otherwise be flagged.

Conclusion

Considering all parameters carefully, it can be deduced that no model has clear superiority in terms pf perfect combination.

It is possible that KNN would most likely improve in performance a little with a larger training data set.

Since the idea of this project was to aid analysis of blood results, but not replace the role of the physician in that process, a combination of high sensitivity and low specificity might be useful. It will lead to many patients being highlighted unnecessarily, but will allow a doctor to quickly note that the blood results (and wider patient presentation) need attention with regard to their liver.

As the idea of the project is to flag the patients by aiding in analysis a combination of highsensitivity and low specificity might be useful.It will lead to many patients being highlighted unnecessarily, but will allow a doctor and the person in consideration to quicklynote that person need attention with regard to their heart.

Despite this, there is a risk that, especially in public-sector healthcare systems, this could be a huge increase in workload for clinicians, who would need to review significant numbers of patients who have no illness, no symptoms and no significant benefit from specialist care.

There is also a risk of ‘alert fatigue’, where doctors get so used to seeing a warning marker that a patient may need specialist care that they begin ignoring the warnings altogether.

For this reason, the Naive Bayes approach, using the `nb` method of the `caret` package, appears to offer the best combination of accuracy, sensitivity and specificity.