Predcit Heart disease

Introduction:

Several health conditions, your lifestyle, and your age and family history can increase your risk for heart disease. These are called risk factors. About half of all Americans (47%) have at least one of the three key risk factors for heart disease: high blood pressure, high cholesterol, and smoking (1). Some of the risk factors for heart disease cannot be controlled, such as your age or family history. But you can take steps to lower your risk by changing the factors you can control. 1. https://www.cdc.gov/heartdisease/risk_factors.htm

Goal: Build a machine learning model that can predict heart disease given several data point of a new patient.

Data: 14 attributes of 303 patients with each patient having a goal field that refers to the presence (or lack of) heart disease.
Source: https://www.kaggle.com/ronitf/heart-disease-uci

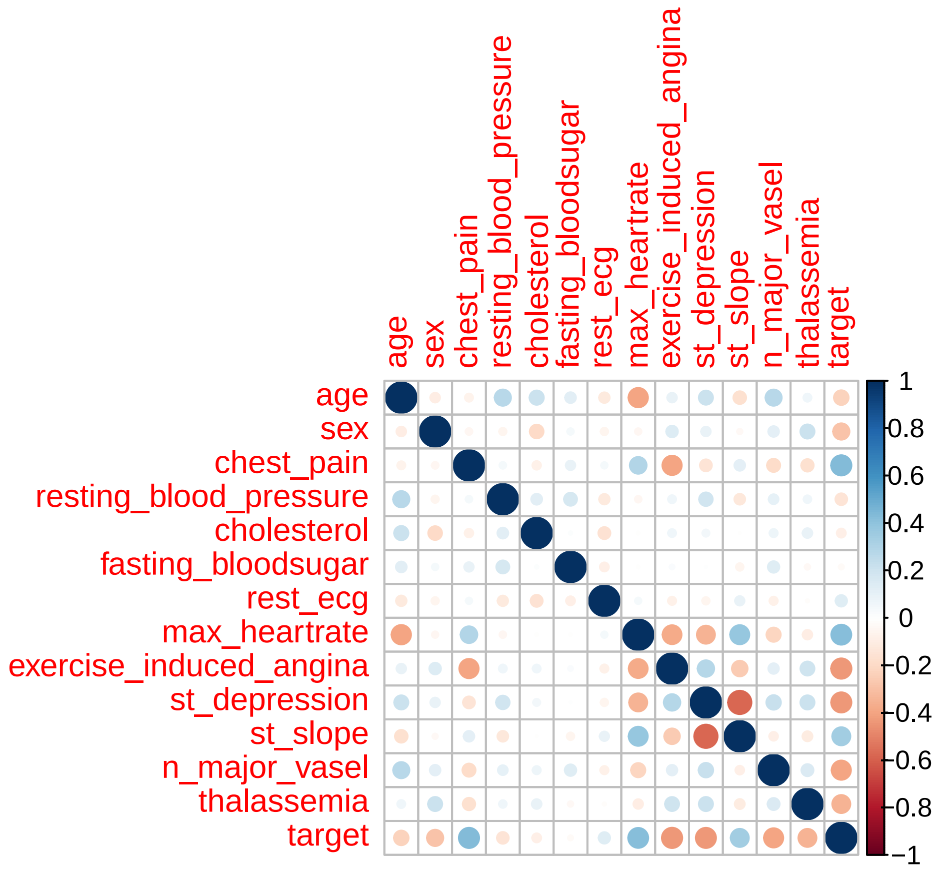
Analysis

I loaded the .csv data file then changed the coulmn names to clearer ones.

To be able to generate the model I created the test and train data sets.The training set will contain random 80% of the database samples (rows) and in accordance test will be 20% of the data.

To start undesranding the data first I created an graph that will show the corelation between all the parameters and heart disease. We can see that some paramteres are postivly correlated such as chest pain and max heart rate and some are negativly such as exercise induced angina and ST depression.

```
M<-cor(hdata)
corrplot(M, method="circle")
```



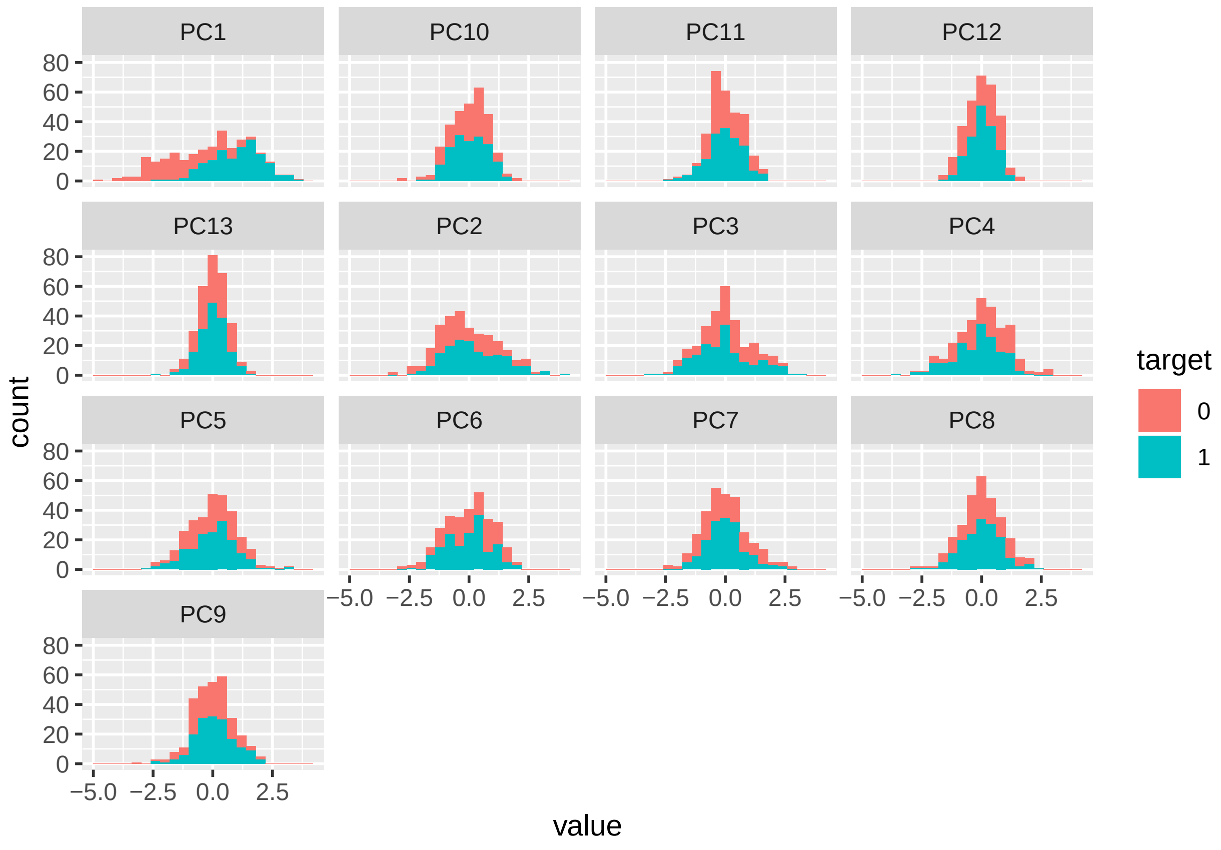
I employed the Principal component analysis (PCA) technique. It might allow to reduce the number of parameters (dimensons) and bring out strong patterns in the dataset.

```
#Principal Component Analysis

hdata.pca <- prcomp(hdata[ , 1:13], center = TRUE, scale. = TRUE)
pc_df<-data.frame(hdata.pca$x)
summary(hdata.pca)
```

```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6
## Standard deviation     1.6622 1.2396 1.10582 1.08681 1.01092 0.98489
## Proportion of Variance 0.2125 0.1182 0.09406 0.09086 0.07861 0.07462
## Cumulative Proportion  0.2125 0.3307 0.42481 0.51567 0.59428 0.66890
##                            PC7     PC8    PC9    PC10    PC11    PC12
## Standard deviation     0.92885 0.88088 0.8479 0.78840 0.72808 0.65049
## Proportion of Variance 0.06637 0.05969 0.0553 0.04781 0.04078 0.03255
## Cumulative Proportion  0.73527 0.79495 0.8503 0.89807 0.93885 0.97140
##                          PC13
## Standard deviation     0.6098
## Proportion of Variance 0.0286
## Cumulative Proportion  1.0000
```

```
df.tidy <- cbind(pc_df,data.frame(target = as.factor(hdata$target))) %>%
  gather(key = "component", value = 'value', 1:13 )

#Vizualize the pca
```

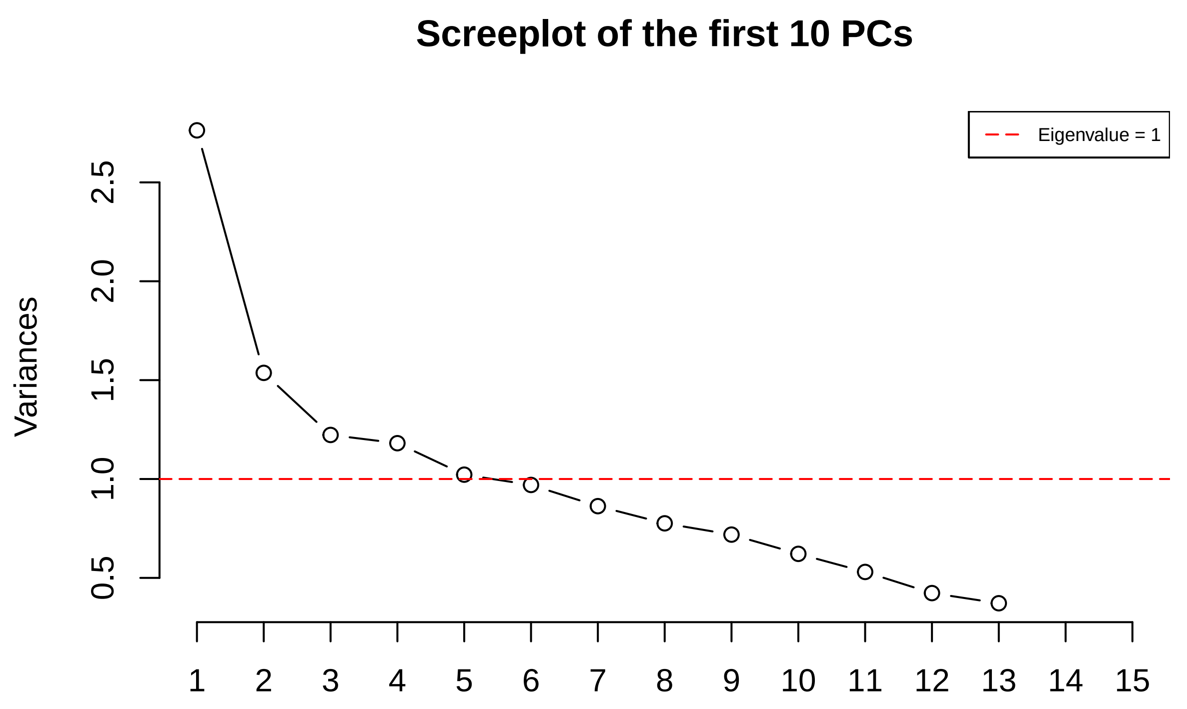
```r
ggplot(df.tidy, aes(x = value, fill = target)) +
  geom_histogram(binwidth=0.4) +
  facet_wrap(~component)
```



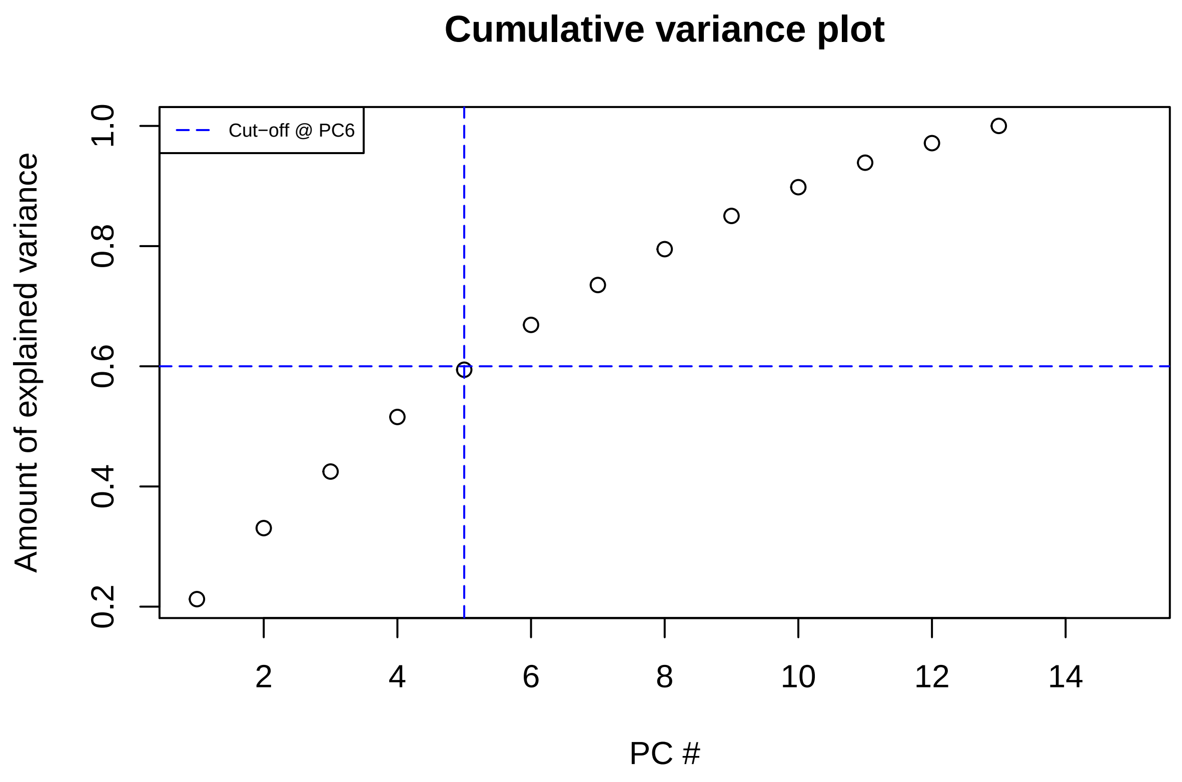
In this graph we see that several componenets contribute the most to the vraiblity.

I will use a screeplot to determine the number of factors to retain.

```r
screeplot(hdata.pca, type = "l", npcs = 15, main = "Screeplot of the first 10 PCs")
abline(h = 1, col="red", lty=5)
legend("topright", legend=c("Eigenvalue = 1"),
       col=c("red"), lty=5, cex=0.6)
```



```
cumpro <- cumsum(hdata.pca$sdev^2 / sum(hdata.pca$sdev^2))
plot(cumpro[0:15], xlab = "PC #", ylab = "Amount of explained variance", main = "Cumulative variance pl
abline(v = 5, col="blue", lty=5)
abline(h = 0.6, col="blue", lty=5)
legend("topleft", legend=c("Cut-off @ PC6"),
       col=c("blue"), lty=5, cex=0.6)
```



We notice that the first 5 componenets have an Eigenvalue > 1 and explain about 60% of the varaince. Altough it would be possible to reduce the number of dimenions I would not do it since the domention number and the row number is not that high and it possible to use all of them. (The PCA was also a good practice)

To create a machine learning algorithm I will use logistic regression since this will allow the classification into the 2 groups. The algorithm will parctice on 80% of the data.

```
fit_glm <- glm(target~.,family=binomial(),data=train_set)
summary(fit_glm)
```

```
##
## Call:
## glm(formula = target ~ ., family = binomial(), data = train_set)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5189  -0.3613   0.1697   0.5817   2.4550
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)               3.664518   2.837689   1.291 0.196574
## age                      -0.016545   0.025911  -0.639 0.523138
## sex                      -1.710843   0.507458  -3.371 0.000748 ***
## chest_pain                0.747703   0.210025   3.560 0.000371 ***
## resting_blood_pressure   -0.015418   0.011700  -1.318 0.187563
## cholesterol              -0.005616   0.004068  -1.380 0.167456
## fasting_bloodsugar       -0.156680   0.616468  -0.254 0.799373
## rest_ecg                 0.436463   0.386827   1.128 0.259186
## max_heartrate            0.025946   0.011976   2.167 0.030269 *
## exercise_induced_angina -1.276709   0.478643  -2.667 0.007645 **
## st_depression           -0.415538   0.239069  -1.738 0.082185 .
## st_slope                 0.528859   0.392930   1.346 0.178323
## n_major_vasel           -0.788073   0.210673  -3.741 0.000183 ***
## thalassemia             -0.975054   0.331549  -2.941 0.003273 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 331.04  on 239  degrees of freedom
## Residual deviance: 168.15  on 226  degrees of freedom
## AIC: 196.15
##
## Number of Fisher Scoring iterations: 6
```

```r
p_glm <- predict(fit_glm,test_set)

#calculation of accuracy for the model

mean(as.numeric(p_glm>=0)==test_set$target)
```

```
## [1] 0.8387097
```

Result

I created a machine learning model that can predict the prognosis for a new unknow patient with an accuracy of 84%. In the model summary we can see that some coefficients are more significant

Conclusion

It is possible to use classification models to create machine learning algorith based on the given data using the glm() function.