DA5030 : Term Project

Nidhi Hookeri

2020-04-19

The project includes the dataset (Indian Liver Patient Dataset) obtained from https://www.kaggle.com/jeevannagaraj/indian-liver-patient-dataset. The dataset is downloaded as .csv format and is loaded onto R studio cloud. This is a case of classfication where using atleast 3 models, based on the performation of the model, ranking of each model will take place. In the dataset, 'is_patient' is a column that contains 1 or 2: 1 indicates the presence of liver disease in the patient and 2 indicates otherwise.

The goal includes checking the performance (of classification) of algorithms with respect to the dependent variable:'is_patient' based on important features selected from the dataset.

Data understanding:

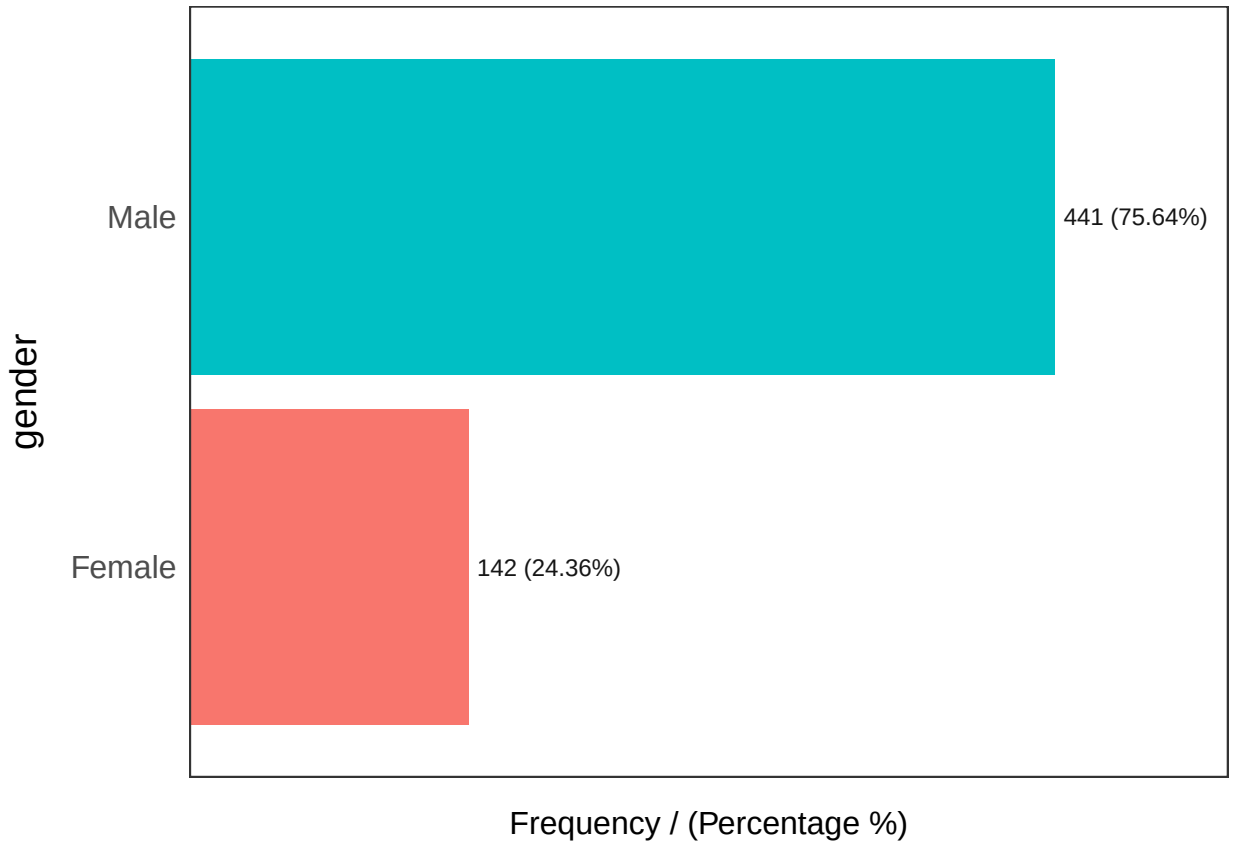
Includes exploring data, verifying quality of data, checking the presence of NA values.

-"funModeling" is a package that can be used in the process of data understanding, it comes with various function such as 'freq', 'plot_num', 'profling_num'

```
#Loading the dataset
liverdata = read.csv("Indian Liver Patient Dataset (ILPD).csv")

#Exploring data - Number of observations (rows) and variables and the
#head of first cases with it's data structure
str(liverdata)
```

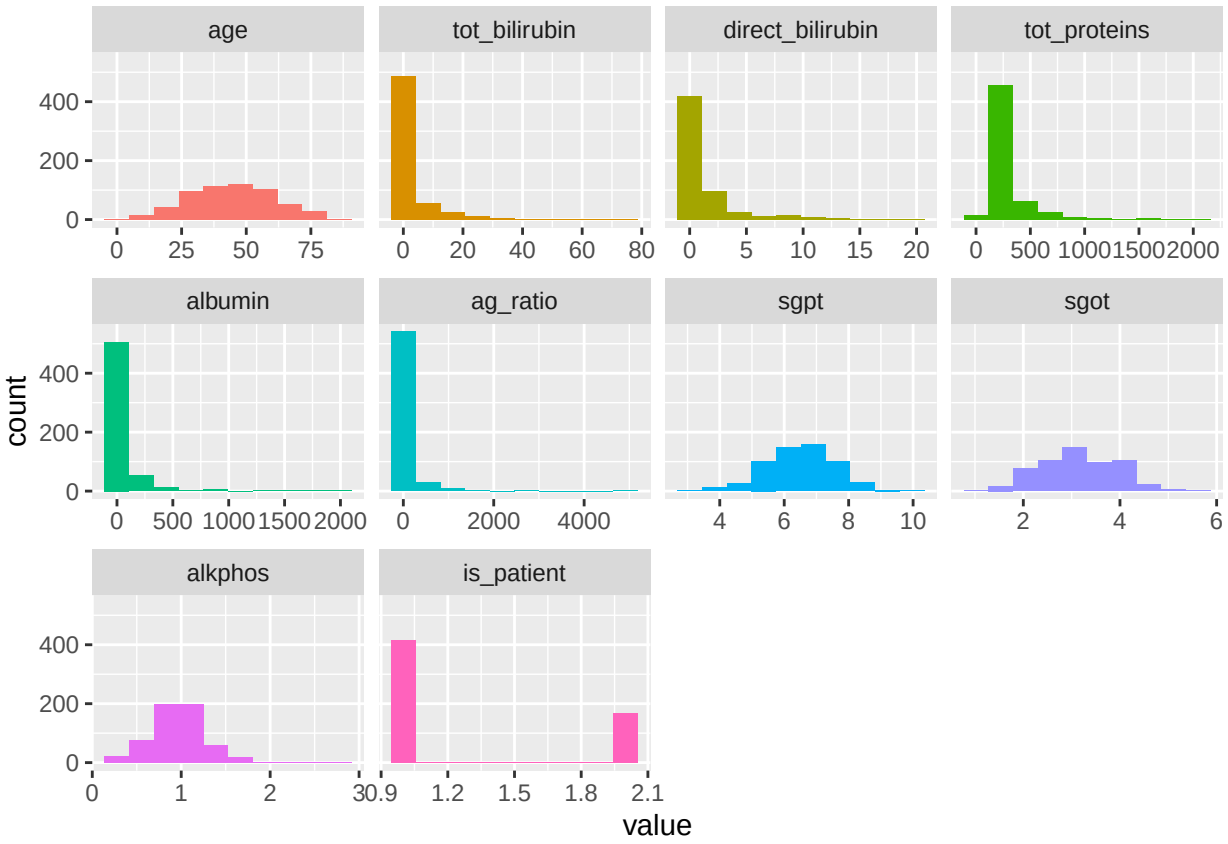
```
## 'data.frame':    583 obs. of  11 variables:
##  $ age             : int  65 62 62 58 72 46 26 29 17 55 ...
##  $ gender          : Factor w/ 2 levels "Female","Male": 1 2 2 2 2 2 1 1 2 2 ...
##  $ tot_bilirubin   : num  0.7 10.9 7.3 1 3.9 1.8 0.9 0.9 0.9 0.7 ...
##  $ direct_bilirubin: num  0.1 5.5 4.1 0.4 2 0.7 0.2 0.3 0.3 0.2 ...
##  $ tot_proteins    : int  187 699 490 182 195 208 154 202 202 290 ...
##  $ albumin         : int  16 64 60 14 27 19 16 14 22 53 ...
##  $ ag_ratio        : int  18 100 68 20 59 14 12 11 19 58 ...
##  $ sgpt            : num  6.8 7.5 7 6.8 7.3 7.6 7 6.7 7.4 6.8 ...
##  $ sgot            : num  3.3 3.2 3.3 3.4 2.4 4.4 3.5 3.6 4.1 3.4 ...
##  $ alkphos         : num  0.9 0.74 0.89 1 0.4 1.3 1 1.1 1.2 1 ...
##  $ is_patient      : int  1 1 1 1 1 1 1 1 2 1 ...
```

```
#Analyzing categorical variable
#install.packages("funModeling")
library(funModeling)
freq(liverdata)
```

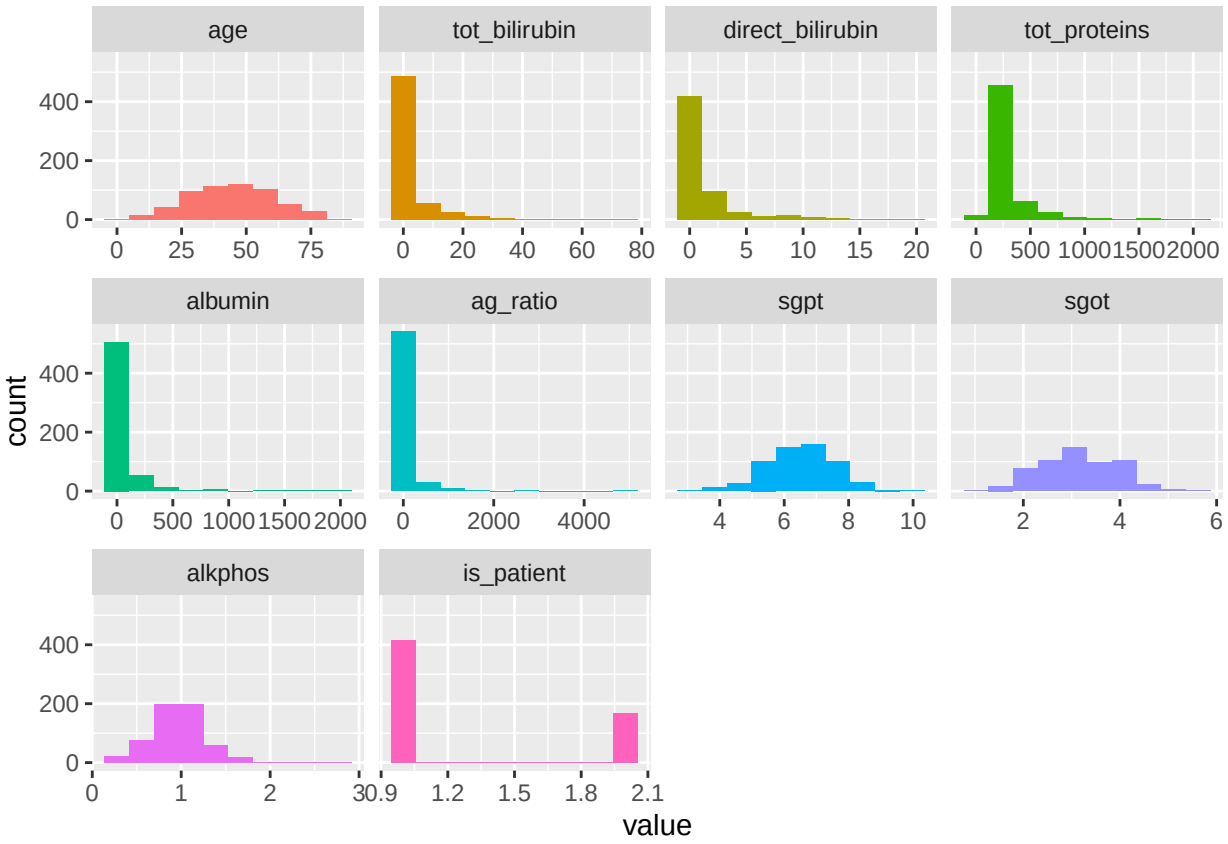



```
##   gender frequency percentage cumulative_perc
## 1   Male       441      75.64           75.64
## 2 Female       142      24.36          100.00
```

```
#Analyzing numeric variable
plot_num(liverdata)
```




```
#The graphs will get exported to current directory
plot_num(liverdata, path_out = ".")
```




```
#Quantitative analysis
#skewness, range98, standard deviation can be looked at to understand #the data
profiling_num(liverdata)[,c(1,3,12,13,15)]
```

```
##            variable     std_dev    skewness   kurtosis
## 1               age  16.1898333 -0.02930965   2.434452
## 2     tot_bilirubin   6.2095217  4.89483852  39.835553
## 3  direct_bilirubin   2.8084976  3.20413176  14.245125
## 4      tot_proteins 242.9379892  3.75541223  20.590666
## 5           albumin 182.6203560  6.53232946  53.136430
## 6         ag_ratio 288.9185291 10.51902355 152.618375
## 7              sgpt   1.0854515 -0.28493665   3.220771
## 8              sgot   0.7955188 -0.04357225   2.605140
## 9           alkphos   0.3195921  0.98972687   6.243282
## 10       is_patient   0.4524902  0.94470115   1.892460
##                      range_98
## 1                  [9.64, 75]
## 2  [0.582, 28.2039999999999]
## 3               [0.1, 12.962]
## 4             [97.82, 1555.4]
## 5  [11.82, 1003.99999999998]
## 6     [12, 976.199999999995]
## 7  [3.682, 8.61799999999999]
## 8                [1.482, 4.9]
## 9             [0.3856, 1.811]
## 10                    [1, 2]
```

```r
#Checking the presence of missing values
summary(is.na(liverdata))
```

```
##     age            gender        tot_bilirubin   direct_bilirubin
##  Mode :logical   Mode :logical   Mode :logical   Mode :logical
##  FALSE:583       FALSE:583       FALSE:583       FALSE:583
##
##  tot_proteins     albumin         ag_ratio          sgpt
##  Mode :logical   Mode :logical   Mode :logical   Mode :logical
##  FALSE:583       FALSE:583       FALSE:583       FALSE:583
##
##     sgot          alkphos        is_patient
##  Mode :logical   Mode :logical   Mode :logical
##  FALSE:583       FALSE:579       FALSE:583
##                  TRUE :4
```

Result Interpretation:

-The dataset contains 583 samples, in which 416 are patients of liver disease and 167 are otherwise with 11 features that contain both categorical and numerical variables

-'funModelling' package allows different functions that can help in understanding the so that, correct implementations of the algorithms can happen.

-'freq' is a function that allows to statistically analyze the categorical variable present in the dataset. It automatically picks out the categorical variable and performs it's course of action.

Quantative analysis: There seems to be a lot of spread in the data in tot_proteins, albumin and ag_ratio.

-'age', 'sgpt', 'sgot' follow minutely left-skewned distribution [range: -0.02 to -0.04]

-range98 talks about the range of where 98% of the data points lie

Presence of NA values: There are 4 NA's that are present in the dataset, all the 4 are present in the column Alkaline Phosphatase.

Source: https://blog.datascienceheroes.com/exploratory-data-analysis-data-preparation-with-funmodeling/

```r
#Detection of outliers
x = which(abs(scale(liverdata$direct_bilirubin)) > 3, liverdata$direct_bilirubin)

x1 = which(abs(scale(liverdata$sgpt)) > 3, liverdata$sgpt)

x2 = which(abs(scale(liverdata$sgot)) > 3, liverdata$sgot)

x3 = which(abs(scale(liverdata$alkphos)) > 3, liverdata$alkphos)
```

There are a few outliers that are not significantly differing from the data. Hence, the outliers are kept as the other features in that row are important.

Data Preparation:

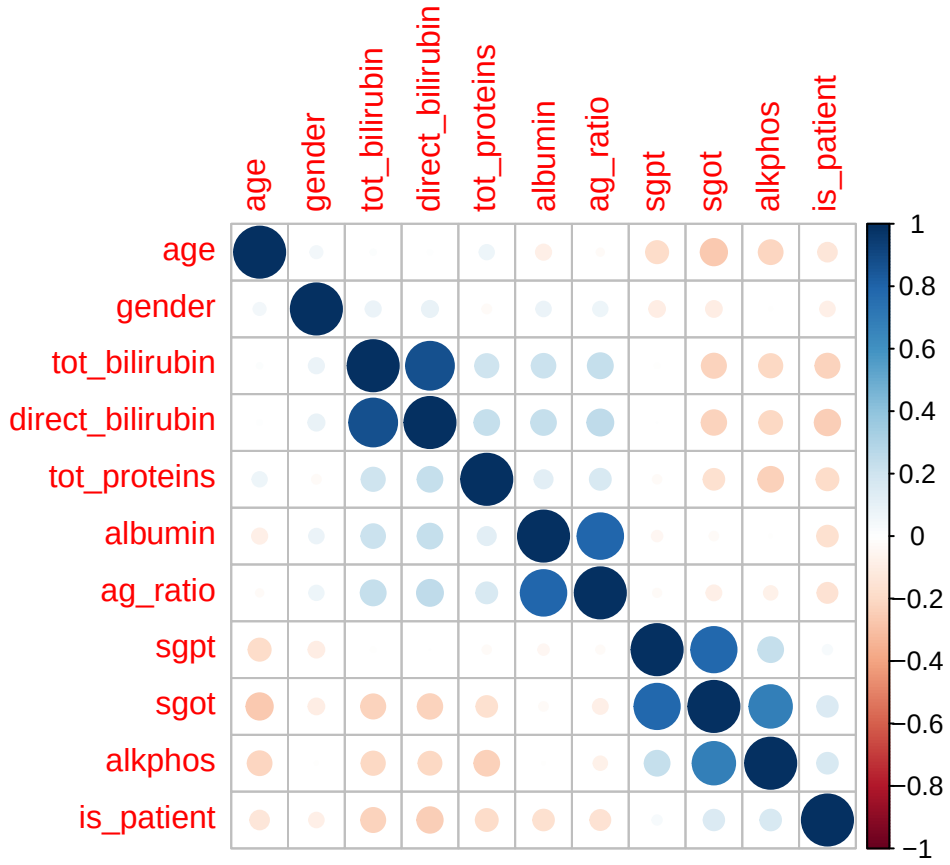
This step entails selecting data, cleaning data. Selecting data: Important features are selected to proceed with, data transformation, categorical variables are converted into numerical variable Cleaning data: The NA rows are omitted

```r
#Removing the rows that contain NA, since, there are only 4 rows that have NA value out of 583 observat
na_liverdata = na.omit(liverdata)

#Converting categorical variable into numeric variable
#Gender is a categorical variable. Female is 1 and Male is 2.
```

```r
na_liverdata$gender = as.numeric(na_liverdata$gender)

#Feature selection
#Best method for numeric input, numeric output is Pearson correlation
#Checking co-relation between the variables of the dataset
correlation = cor(na_liverdata)
correlation
```

```
##                          age       gender tot_bilirubin direct_bilirubin
## age               1.000000000  0.055880710   0.011000374     6.784303e-03
## gender            0.055880710  1.000000000   0.088067618     9.916008e-02
## tot_bilirubin     0.011000374  0.088067618   1.000000000     8.744810e-01
## direct_bilirubin  0.006784303  0.099160081   0.874480969     1.000000e+00
## tot_proteins      0.078878350 -0.029367677   0.205739173     2.340076e-01
## albumin          -0.087799162  0.081338710   0.213375493     2.331801e-01
## ag_ratio         -0.020498946  0.079420992   0.237323055     2.570224e-01
## sgpt             -0.186248122 -0.095149286  -0.007905923     3.270877e-05
## sgot             -0.264210935 -0.095578941  -0.222086570    -2.284092e-01
## alkphos          -0.216408346 -0.003424034  -0.206267186    -2.001247e-01
## is_patient       -0.133163583 -0.081349445  -0.220217940    -2.462729e-01
##                  tot_proteins     albumin    ag_ratio          sgpt        sgot
## age                 0.07887835 -0.08779916 -0.02049895 -1.862481e-01 -0.26421094
## gender             -0.02936768  0.08133871  0.07942099 -9.514929e-02 -0.09557894
## tot_bilirubin       0.20573917  0.21337549  0.23732305 -7.905923e-03 -0.22208657
## direct_bilirubin    0.23400757  0.23318008  0.25702239  3.270877e-05 -0.22840915
## tot_proteins        1.00000000  0.12477671  0.16657999 -2.706202e-02 -0.16341865
## albumin             0.12477671  1.00000000  0.79186215 -4.243210e-02 -0.02865750
## ag_ratio            0.16657999  0.79186215  1.00000000 -2.575101e-02 -0.08491457
## sgpt               -0.02706202 -0.04243210 -0.02575101  1.000000e+00  0.78311217
## sgot               -0.16341865 -0.02865750 -0.08491457  7.831122e-01  1.00000000
## alkphos            -0.23416650 -0.00237499 -0.07003983  2.348872e-01  0.68963234
## is_patient         -0.18336330 -0.16311694 -0.15183446  3.361377e-02  0.15976956
##                      alkphos  is_patient
## age               -0.216408346 -0.13316358
## gender            -0.003424034 -0.08134945
## tot_bilirubin     -0.206267186 -0.22021794
## direct_bilirubin  -0.200124685 -0.24627293
## tot_proteins      -0.234166499 -0.18336330
## albumin           -0.002374990 -0.16311694
## ag_ratio          -0.070039828 -0.15183446
## sgpt               0.234887181  0.03361377
## sgot               0.689632342  0.15976956
## alkphos            1.000000000  0.16313136
## is_patient         0.163131363  1.00000000
```

```r
#Visually representing correlation
#install.packages("corrplot")
library(corrplot)
corrplot(correlation)
```



Result Interpretation:

-The 4 NA values are omitted leaving behind a total of 579 samples.

Result of correlation matrix and plot:

According to the correlation matrix above, there is a co-relation between the direct_bilirubin and total_bilirubin, between sgot(Alanine Aminotransferase) and sgpt(Aspartate Aminotransferase) and between Alkaline Phosphatase and Aspartate aminotransferase.

The correlation between the direct and total is obvious, because as total bilirubin is the sum of direct and indirect bilirubin, so they have linear proportionality.

The relationship between sgpt and sgot also is strong since, both of them play a vital role in causing liver disease. Elevated levels of transferases cause liver disease. The correlation co-efficient is 0.78.

Elevated levels of alkaline phosphatase also causes liver damage. The corelation coeffecient between alkaline phosphatase and Alanine Aminotransferase(sgot) is 0.69

After studying the correlation between variables, we can say that, aspartate aminotransferase, alanine aminotransferase, alkaline phosphatase and direct bilirubin are leading factors in causing liver disease.

Principle Component Analysis

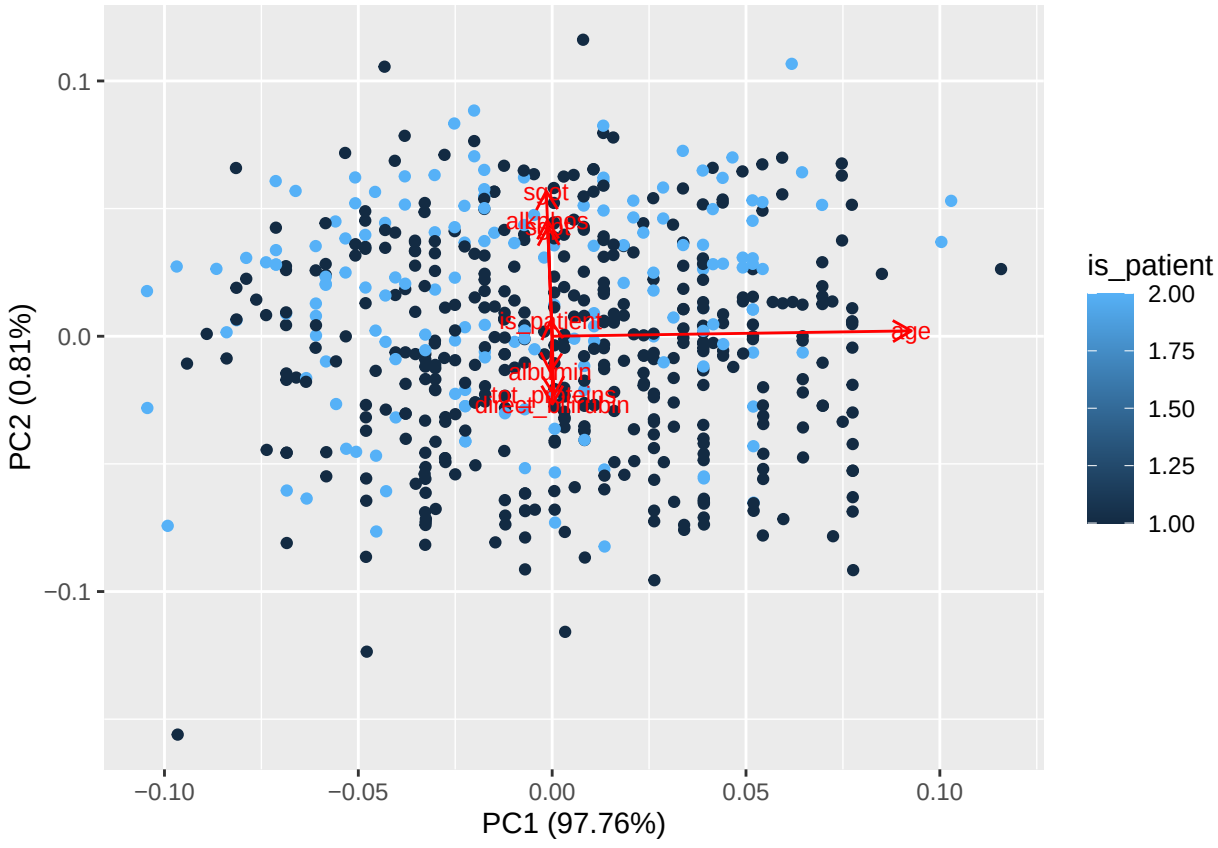
PCA helps in reducing the dimension of the dataset by decreasing the number of variables. From this, we can infer the variation criteria observed in each principal component.

```
#Selecting data
features = na_liverdata[,c(4,5,6,8,9,10)]
#Scaling the data
features_scaled = as.data.frame(scale(features))
```

```
#Adding age and is_patient columns to the scaled dataframe
features_scaled$age = na_liverdata$age
features_scaled$is_patient = na_liverdata$is_patient

#Computing PCA
features_pca = prcomp(features_scaled)
features_summ = summary(features_pca)
features_summ
```

```
## Importance of components:
##                            PC1     PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     16.2270 1.47272 1.15029 0.96609 0.86554 0.78989 0.42325
## Proportion of Variance  0.9776 0.00805 0.00491 0.00347 0.00278 0.00232 0.00067
## Cumulative Proportion   0.9776 0.98565 0.99056 0.99403 0.99681 0.99913 0.99979
##                            PC8
## Standard deviation     0.23728
## Proportion of Variance 0.00021
## Cumulative Proportion  1.00000
```

```
#Plotting PCA
#install.packages("ggfortify")
library(ggfortify)
autoplot(features_pca, data = features_scaled,
         colour = 'is_patient', loadings = TRUE, loadings.label = TRUE, loadings.label.size  = 3)
```



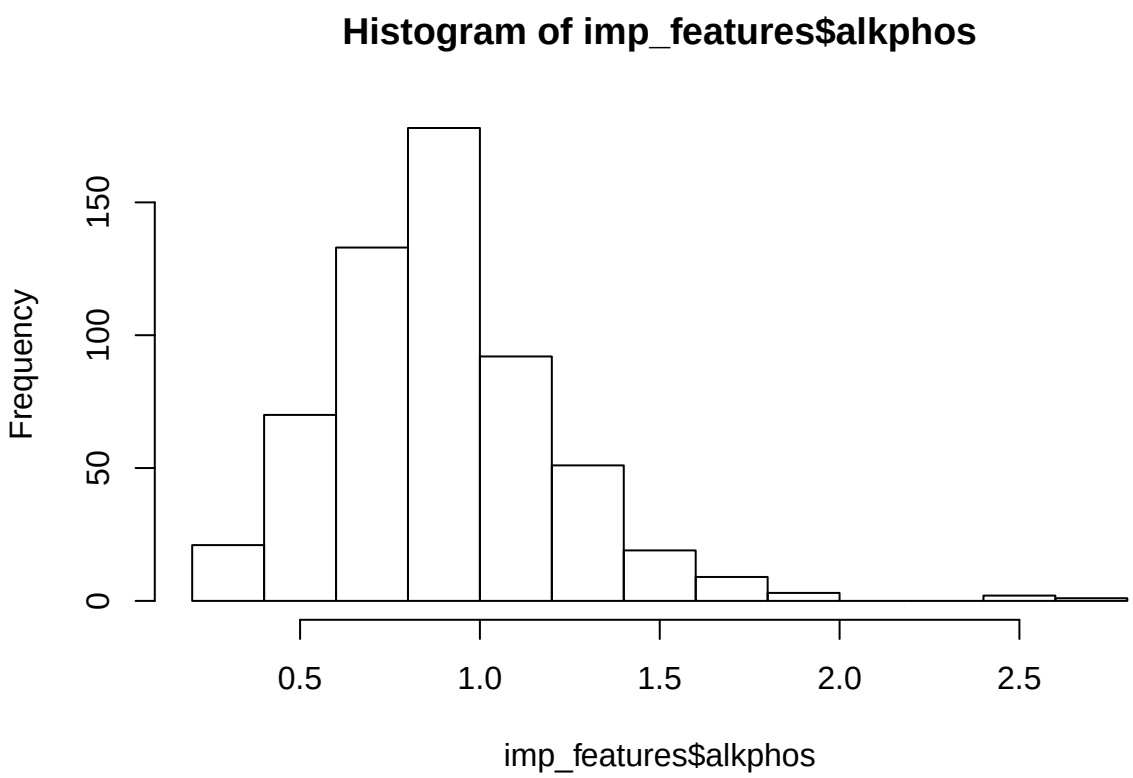
Result Interpretation:

It is clear from the result that about 97% variation is observed in PC1. With respect to 'is_patient', 'age' is

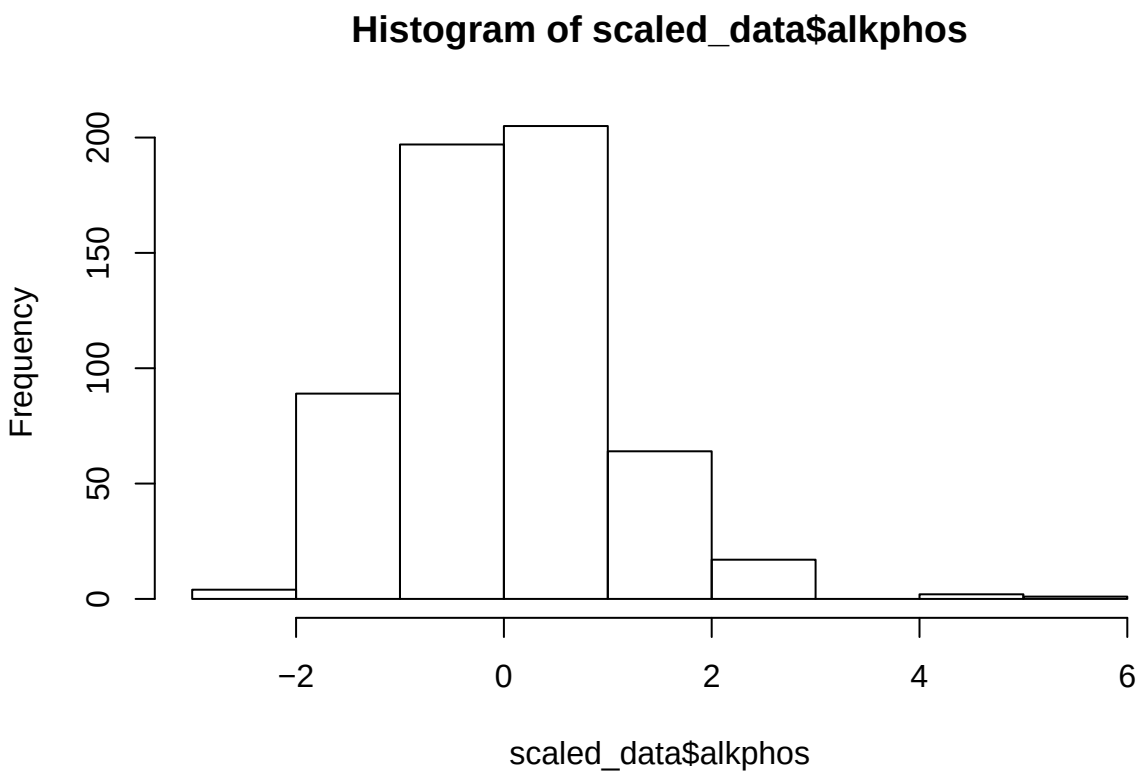
proportional to 'sgot' and 'alkphos' and is negatively proportional to the rest.

Direct_bilirubin is an important factor that causes liver disease but it is also seen that sgpt, sgot and alkphos also contribute to the cause of liver disease from correlation, hence, the mentioned three are included in the further analysis.

```
#Extracting data from important features
#Important features are direct bilirubin, sgpt, sgot and alkphos
imp_features = na_liverdata[,c(4, 8, 9, 10)]
#summary(imp_features)
hist(imp_features$alkphos)
```




```
#Scaling the data
scaled_data = as.data.frame(scale(imp_features))
#summary(scaled_data)
#Representation of the scaled data
hist(scaled_data$alkphos)
```




```
#Adding the age and gender column to the scaled data
scaled_data$age = na_liverdata$age
scaled_data$gender = na_liverdata$gender
scaled_data$is_patient = na_liverdata$is_patient
```

Modeling:

1) Splitting the dataset into training and testing data
2) Naive Bayes, SVM algorithm, Decision tree algorithms were implemented - these are the best models used for numeric variables.
3) Evaluation of the model and accuracy of the model is calculated alongside

```
#Splitting the dataset into training and testing data (80-20%)
sample = sample(nrow(scaled_data), 0.80*nrow(scaled_data))
train = scaled_data[sample,]
test = scaled_data[-sample,]
```

Model 1 - Naive Bayes Algorithm:

This is an algorithm that is used for classification. This algorithm assumes that the presence of one feature in a class is completely unrelated to the presence of all other features.

```
set.seed(234)
#Using the package e1071 for Naive Bayes
library(e1071)

#Factorizing the dataset
train_fac = as.data.frame(lapply(train, factor))
test_fac = as.data.frame(lapply(test, factor))

#Using the function to classify
naive_model = naiveBayes(is_patient~. , data = train_fac)
```

```r
#Predicting for test dataset
pred_naive = predict(naive_model, test$is_patient, type = "class")

#Confusion matrix to check accuracy
naive_table = table(pred_naive, test_fac$is_patient)

naive_acc = sum(diag(naive_table)) / sum(naive_table)
cat("The accuracy of the classification using Naive Bayes model is", naive_acc*100,"%")
```

```
## The accuracy of the classification using Naive Bayes model is 74.13793 %
```

Result Interpretation:

Naive Bayes is an algorithm that doesnot consider the dependency relationship between feautres. Hence, this is a good algorithm to begin with. After attempting to check the correct classification rate, it's seen that 81 out of 116 are correctly classified as patients with liver disease with 35 misclassifications.

Model 2 - SVM algorithm:

SVM works on the basis of creating hyperplanes by which it classifies the data points. The algorithm can be used both for regression and classification but is widely used for classification.

```r
#Loading library
library(e1071)

#Using svm function from the library
svm_model = svm(is_patient~., data = train, type = 'C-classification', kernel = 'linear')
summary(svm_model)
```

```
##
## Call:
## svm(formula = is_patient ~ ., data = train, type = "C-classification",
##     kernel = "linear")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  1
##
## Number of Support Vectors:  288
##
##  ( 153 135 )
##
##
## Number of Classes:  2
##
## Levels:
##  1 2
```

```r
#Predicting with the test data
svm_pred = predict(svm_model, test[,1:6])

#Creating a confusion matrix
svm_table = table(test[,7], svm_pred)
```

```
svm_acc = sum(diag(svm_table)) / sum(svm_table)
cat("The accuracy of the SVM model is", svm_acc*100,"%")
```

```
## The accuracy of the SVM model is 74.13793 %
```

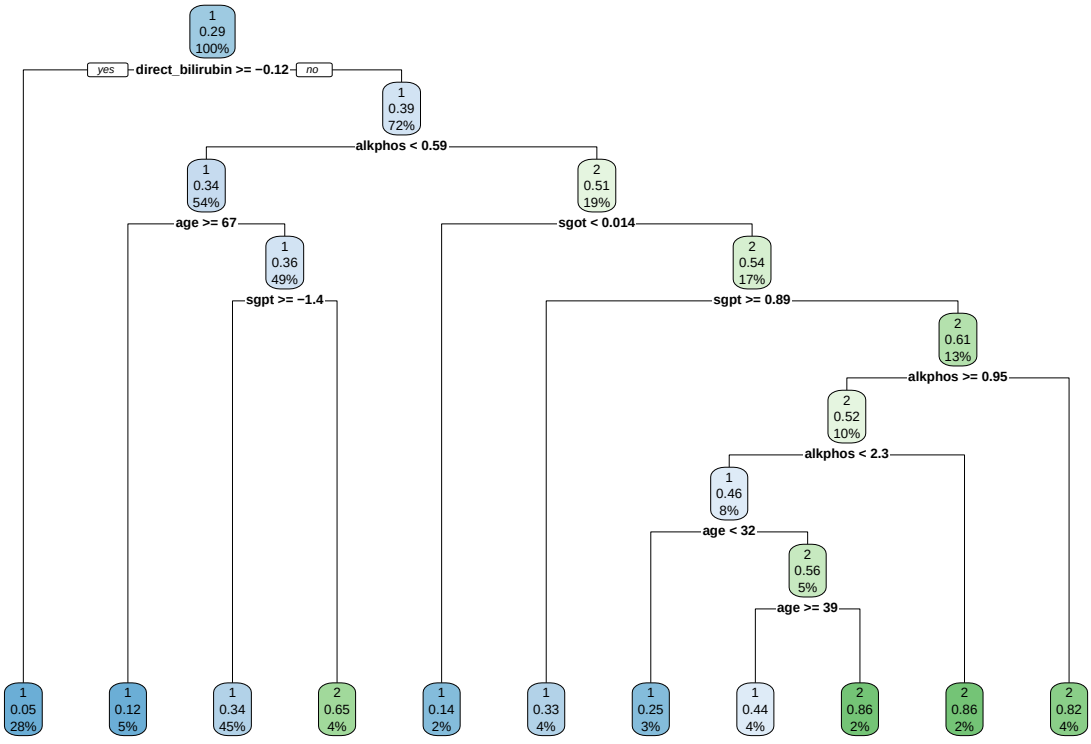
Result Interpretation:

There are a total of 83 out of 116 correctly classified cases.

Model 3 - Decision tree:

It is a supervised machine learning algorithm where the data is continuously split according to certain parameters. The algorithm was used to check classification of patients with liver disease or otherwise.

```
#install.packages("rpart")
#install.packages("caret")
library(rpart)
library(caret)
#Using the function rpart to build decision tree
dt_model = rpart(is_patient~., data = train, method = "class")

#Visualization of the built decision tree
#install.packages("rpart.plot")
library(rpart.plot)
rpart.plot(dt_model)
```



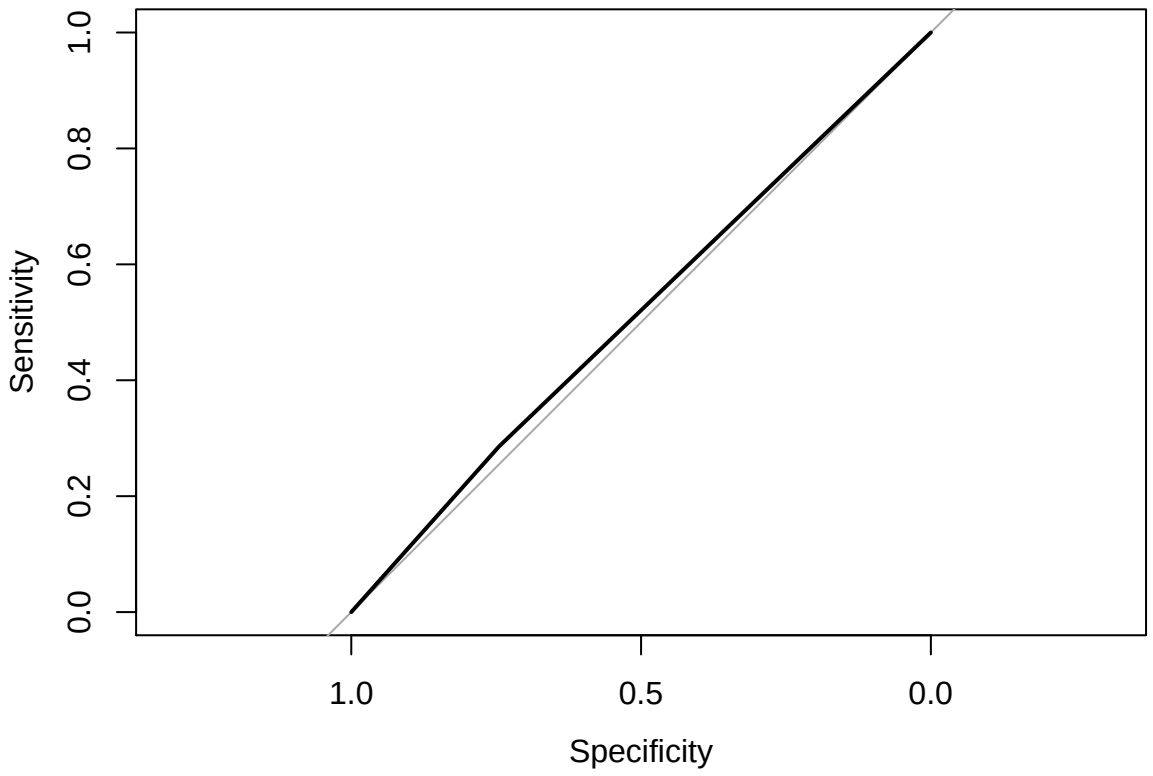

```
#Predicting with test data : Predict the class (1/2) of the test set
dt_pred = predict(dt_model, test, type = "class")

#Table to count how many are diagnosed with liver condition and not and
#compare to the correct classification
dt_table = table(dt_pred, test$is_patient)
dt_table
```

```
##
## dt_pred  1  2
##       1 76 26
##       2 10  4
```

```
#Accuracy calculation
dt_acc = sum(diag(dt_table)) / sum(dt_table)

cat("The accuarcy of classification using decision tree is", dt_acc*100,"%")
```

```
## The accuarcy of classification using decision tree is 68.96552 %
```

```
#ROC plot and AUC
#install.packages("pROC")
library(pROC)
dt_roc = roc(dt_pred, test$is_patient)
plot(dt_roc)
```




```
dt_auc = auc(dt_roc)
cat("The AUC of the model is",dt_auc)
```

```
## The AUC of the model is 0.5154062
```

Result Interpretation:

76 samples are classified correctly with 40 misclassifications.

AUC : Area under curve is a very well used method for model evaluation. From the evaluation, we can infer the capacity of the model to classify. Higher the AUC (closer to 1), higher the ability to correctly classify the data.

Random Forest:

Random forest chooses a random subset of features and builds many Decision Trees. The model averages

out all the predictions of the Decisions trees -It is an ensemble model that is used on this data to check classification of the data -Random Forest randomly resamples the data and builds multiple decision tree and finally averages the result

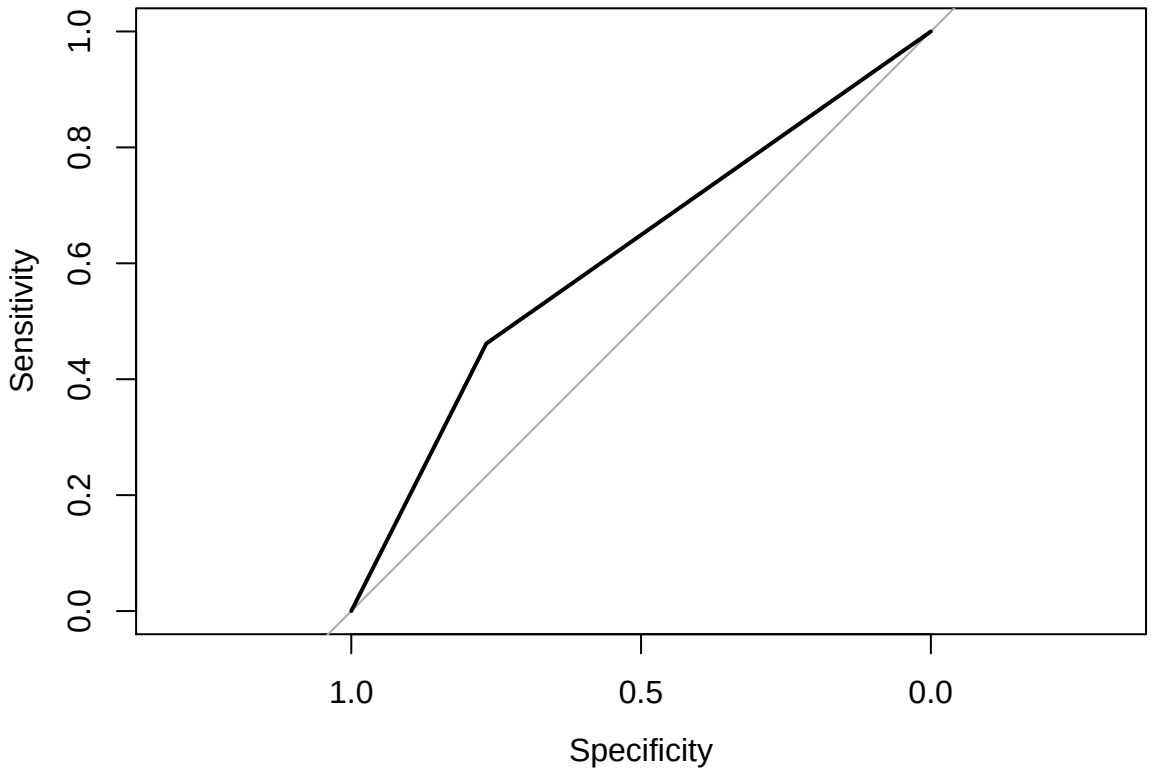
```
set.seed(1223)
#install.packages('randomForest')
library(randomForest)

#Converting the dependent function into factor for classification
trainx = train
testx = test
trainx$is_patient = factor(trainx$is_patient)
testx$is_patient = factor(testx$is_patient)

#Implementing randomForest algorithm on the train data
rf_model = randomForest(is_patient~., data = trainx, proximity = T)

#Predicting on the test data
rf_pred = predict(rf_model, testx)

#Creating a table to interpret classifications and misclassification
rf_table = table(rf_pred, testx$is_patient)
rf_table
```

```
##
## rf_pred  1  2
##       1 79 24
##       2  7  6
```

```
rf_acc = sum(diag(rf_table)) / sum(rf_table)
cat("The accuracy of the model is", rf_acc*100,"%")
```

```
## The accuracy of the model is 73.27586 %
```

```
#ROC plot and AUC
rf_roc = roc(rf_pred, test$is_patient)
plot(rf_roc)
```




```
rf_auc = auc(rf_roc)
cat("The AUC of the model is",rf_auc)
```

```
## The AUC of the model is 0.6142644
```

Result Interpretation:

The total number of misclassifications is 35 which is lesser than the total mis-classifications of decision tree model (presented above). 81 samples are classified correctly. The accuracy of the model is significantly higher.

Classification evaluation:

Precision and recall calculation:

Precision is defined as the fraction of correct predictions for a certain class, whereas recall is the fraction of instances of a class that were correctly predicted.

```
#Calculating precision and recall
#For Decision Tree
dt_prec = as.data.frame(diag(dt_table) / colSums(dt_table))[1,]
dt_recall = as.data.frame(diag(dt_table) / rowSums(dt_table))[1,]

#For SVM
svm_prec = as.data.frame(diag(svm_table) / colSums(svm_table))[1,]
svm_recall = as.data.frame(diag(svm_table) / rowSums(svm_table))[1,]

#For Naive Bayes
naive_prec = as.data.frame(diag(naive_table) / colSums(naive_table))[1,]
naive_recall = as.data.frame(diag(naive_table) / rowSums(naive_table))[1,]

#For Random Forest
rf_prec = as.data.frame(diag(rf_table) / colSums(rf_table))[1,]
```

```r
rf_recall = as.data.frame(diag(rf_table) / rowSums(rf_table))[1,]

finale = data.frame("Precision"=c(rf_prec, dt_prec, svm_prec, naive_prec), "Recall"=c(rf_recall, dt_reca
rownames(finale) = c("Random Forest", "Decision tree", "SVM", "Naive Bayes")
finale
```

```
##               Precision    Recall Accuracy
## Random Forest 0.9186047 0.7669903 73.27586
## Decision tree 0.8837209 0.7450980 68.96552
## SVM           0.7413793 1.0000000 74.13793
## Naive Bayes   1.0000000 0.7413793 74.13793
```

Result Interpretation:

For binary classification, the positive class is looked at while reporting metrics for precision and recall.

Source: https://blog.revolutionanalytics.com/2016/03/com_class_eval_metrics_r.html

References:

1)https://www.analyticsvidhya.com/blog/2015/11/beginners-guide-on-logistic-regression-in-r/? 2)https://towardsdatascience.com/decision-tree-classification-de64fc4d5aac