Implementation of Algorithms

```
#----------------- Loading Libraries And Normalization of Data ------------------#
#Loding Libraries
library(class)
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
library(C50)
library(gmodels)
library(stats)
library(cluster)
library(fpc)
library(plotly)
```

```
##
## Attaching package: 'plotly'
```

```
## The following object is masked from 'package:ggplot2':
##
##     last_plot
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following object is masked from 'package:graphics':
##
##     layout
```

```
library(e1071)
library(randomForest)
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(caTools)
```

```
## Warning: package 'caTools' was built under R version 3.3.2
```

```
#----------------- load orignal dataset ------------------------------------#
data.liver <- read.csv("liver.csv",TRUE,)
str(data.liver)
```

```
## '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 ...
##  $ TB            : num  0.7 10.9 7.3 1 3.9 1.8 0.9 0.9 0.9 0.7 ...
##  $ DB            : num  0.1 5.5 4.1 0.4 2 0.7 0.2 0.3 0.3 0.2 ...
##  $ Alkphos       : int  187 699 490 182 195 208 154 202 202 290 ...
##  $ Sgpt          : int  16 64 60 14 27 19 16 14 22 53 ...
##  $ Sgot          : int  18 100 68 20 59 14 12 11 19 58 ...
##  $ TP            : num  6.8 7.5 7 6.8 7.3 7.6 7 6.7 7.4 6.8 ...
##  $ ALB           : num  3.3 3.2 3.3 3.4 2.4 4.4 3.5 3.6 4.1 3.4 ...
##  $ A.G.Ratio     : num  0.9 0.74 0.89 1 0.4 1.3 1 1.1 1.2 1 ...
##  $ Selector.field: int  1 1 1 1 1 1 1 1 2 1 ...
```

```
# Removing Gender column
data.liver <- data.liver[,c(1,3:11)]


# selecting the column of missing values and defining ina variable
x <- c(data.liver$A.G.Ratio)
summary(x)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.3000  0.7000  0.9471  0.9471  1.1000  2.8000
```

```
#------------------------ Mean without NA values ----------------------------#
result.mean<-mean(x,na.rm = TRUE)
print(result.mean)
```

```
## [1] 0.9470639
```

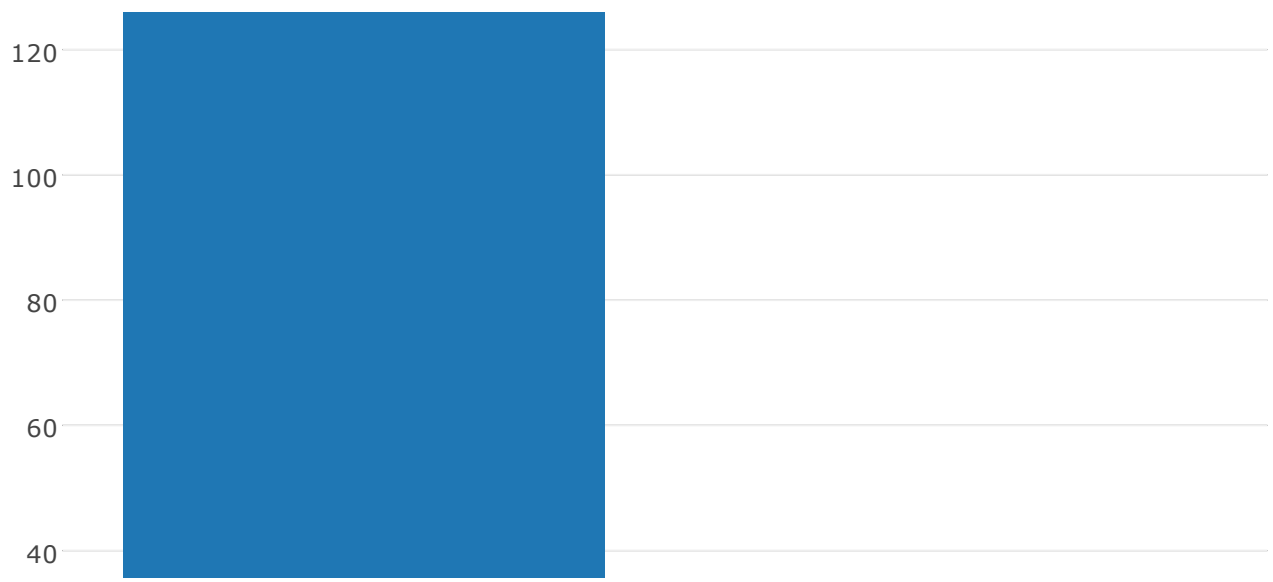
```
liver_normal <- data.liver[,c(1:9)]

# Function for Normalization
normalize <- function(w) {
  return((w-min(w)) / (max(w) - min(w)))
}

# Normalizing the dataset for calculations
liver_normal <- as.data.frame(lapply(liver_normal,normalize))
Selector.field <- as.factor(data.liver$Selector.field)
liver_normal<-cbind(liver_normal,Selector.field)
str(liver_normal)
```

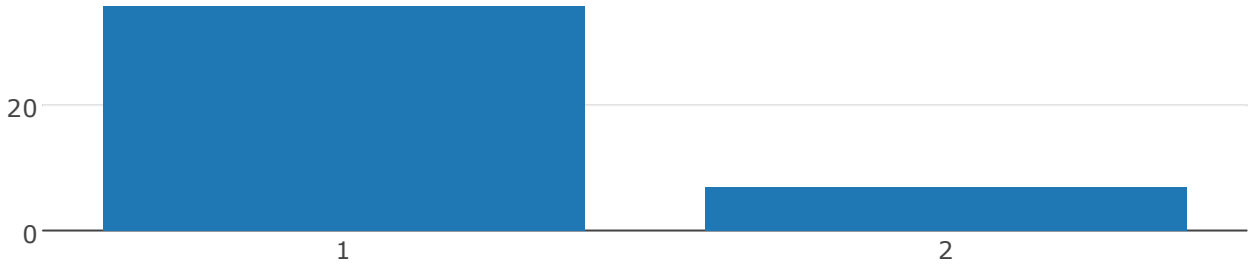
```
## 'data.frame':    583 obs. of  10 variables:
##  $ Age           : num  0.709 0.674 0.674 0.628 0.791 ...
##  $ TB            : num  0.00402 0.14075 0.09249 0.00804 0.04692 ...
##  $ DB            : num  0 0.2755 0.2041 0.0153 0.0969 ...
##  $ Alkphos       : num  0.0606 0.3107 0.2086 0.0581 0.0645 ...
##  $ Sgpt          : num  0.00302 0.02714 0.02513 0.00201 0.00854 ...
##  $ Sgot          : num  0.00163 0.0183 0.01179 0.00203 0.00996 ...
##  $ TP            : num  0.594 0.696 0.623 0.594 0.667 ...
##  $ ALB           : num  0.522 0.5 0.522 0.543 0.326 ...
##  $ A.G.Ratio     : num  0.24 0.176 0.236 0.28 0.04 0.4 0.28 0.32 0.36 0.28 ...
##  $ Selector.field: Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 2 1 ...
```

```
#--------------------------- Data Partitioning for KNN Algorithm  -------------------#
trainer <- liver_normal[1:450,c(1:9)]
tester <- liver_normal[451:583,c(1:9) ]
trainer_target <- liver_normal[1:450,10]
tester_target <- liver_normal[451:583,10]

#################################################################################
#                              Applying KNN Algorithm                           #
#################################################################################

knn_model <- knn(train = trainer, test = tester,
          cl=trainer_target, k=sqrt(583))

# Printing and plotting model results
print(knn_model)
```

```
##   [1] 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1
##  [36] 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [71] 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1
## [106] 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1
## Levels: 1 2
```

```
x<-rchisq(100,5,0)
plot_ly(x=knn_model,type = 'histogram')
```






```
# Summary of the Algorithm
summary(knn_model)
```

```
##   1   2
## 126   7
```

```
#Finding Results using confusion matrix
confusionMatrix(knn_model, tester_target)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1 92 34
##          2  5  2
##
##                Accuracy : 0.7068
##                  95% CI : (0.6216, 0.7825)
##     No Information Rate : 0.7293
##     P-Value [Acc > NIR] : 0.7552
##
##                   Kappa : 0.0054
##  Mcnemar's Test P-Value : 7.34e-06
##
##             Sensitivity : 0.94845
##             Specificity : 0.05556
##          Pos Pred Value : 0.73016
##          Neg Pred Value : 0.28571
##              Prevalence : 0.72932
##          Detection Rate : 0.69173
##    Detection Prevalence : 0.94737
##       Balanced Accuracy : 0.50200
##
##        'Positive' Class : 1
##
```

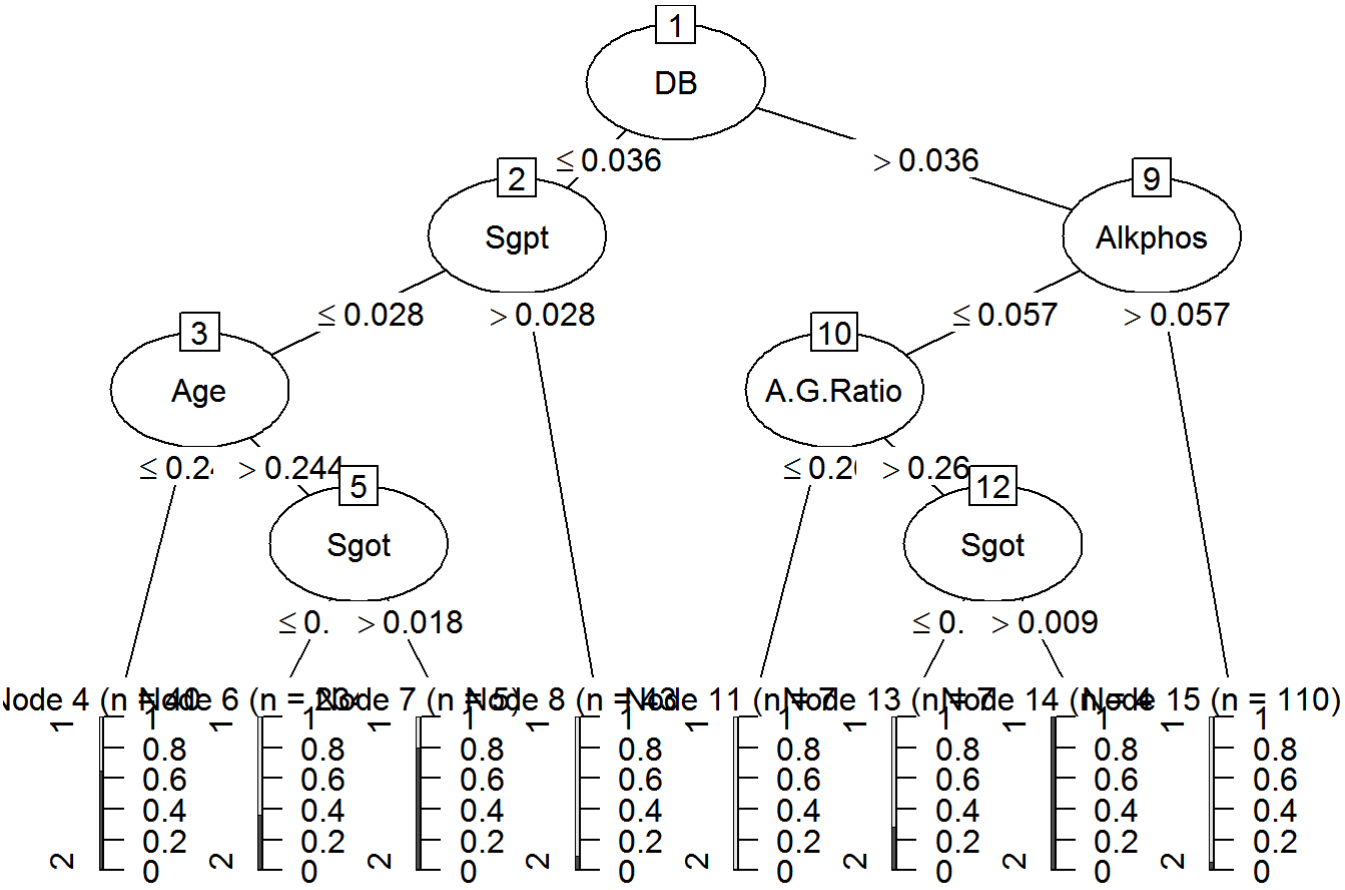
```
#--------------- Validating Data Partitioning for this C5.0 Algorithm ------------------#
trainc50 <- liver_normal[1:450,c(1:9)]
testc50 <- liver_normal[451:583,]
train_targetc50 <- liver_normal[1:450,10]
test_targetc50 <- liver_normal[451:583,10]


#############################################################################
#                                C50 ALGORITHM                              #
#############################################################################

C50_model<- C5.0(trainc50,train_targetc50)

# print result of Algorithm
print(C50_model)
```

```
##
## Call:
## C5.0.default(x = trainc50, y = train_targetc50)
##
## Classification Tree
## Number of samples: 450
## Number of predictors: 9
##
## Tree size: 8
##
## Non-standard options: attempt to group attributes
```

```
# Summary of the Algorithm
summary(C50_model)
```

```
##
## Call:
## C5.0.default(x = trainc50, y = train_targetc50)
##
##
## C5.0 [Release 2.07 GPL Edition]      Fri Dec 02 20:04:43 2016
## -------------------------------
##
## Class specified by attribute `outcome'
##
## Read 450 cases (10 attributes) from undefined.data
##
## Decision tree:
##
## DB <= 0.03571429:
## :...Sgpt > 0.02763819: 1 (43/4)
## :   Sgpt <= 0.02763819:
## :   :...Age <= 0.244186: 2 (42/14)
## :       Age > 0.244186:
## :       :...Sgot <= 0.0182964: 1 (232/83)
## :           Sgot > 0.0182964: 2 (5/1)
## DB > 0.03571429:
## :...Alkphos > 0.0566683: 1 (110/6)
##     Alkphos <= 0.0566683:
##     :...A.G.Ratio <= 0.26: 1 (7)
##         A.G.Ratio > 0.26:
##         :...Sgot <= 0.008944907: 1 (7/2)
##             Sgot > 0.008944907: 2 (4)
##
##
## Evaluation on training data (450 cases):
##
##      Decision Tree
##    ----------------
##    Size      Errors
##
##       8  110(24.4%)   <<
##
##
##     (a)   (b)    <-classified as
##    ----  ----
##     304    15    (a): class 1
##      95    36    (b): class 2
##
##
##  Attribute usage:
##
##  100.00% DB
##   71.56% Sgpt
##   62.00% Age
##   55.11% Sgot
##   28.44% Alkphos
##    4.00% A.G.Ratio
##
##
## Time: 0.0 secs
```

```
plot(C50_model,type="extended")
```




```
# Testing the model
pred_m<-predict(C50_model,testc50)
print(pred_m)
```

```
##   [1] 1 1 1 1 1 2 1 1 1 1 1 1 1 2 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1
##  [36] 1 1 1 1 1 1 2 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [71] 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [106] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 1 1 1 1 1 1 1 1 1 1
## Levels: 1 2
```

```
# Making table of Confusion matrix
CrossTable(test_targetc50, pred_m,
           prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,
           dnn = c('actual Selector.Field', 'predicted Selector.field'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  133
##
##
##                      | predicted Selector.field
## actual Selector.Field |         1 |         2 | Row Total |
## ---------------------|-----------|-----------|-----------|
##                    1 |        91 |         6 |        97 |
##                      |     0.684 |     0.045 |           |
## ---------------------|-----------|-----------|-----------|
##                    2 |        32 |         4 |        36 |
##                      |     0.241 |     0.030 |           |
## ---------------------|-----------|-----------|-----------|
##          Column Total |       123 |        10 |       133 |
## ---------------------|-----------|-----------|-----------|
##
##
```

```
# Printing the Result using Confusion Matrix
confusionMatrix(test_targetc50,pred_m)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1 91  6
##          2 32  4
##
##                Accuracy : 0.7143
##                  95% CI : (0.6295, 0.7892)
##     No Information Rate : 0.9248
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.0637
##  Mcnemar's Test P-Value : 5.002e-05
##
##             Sensitivity : 0.7398
##             Specificity : 0.4000
##          Pos Pred Value : 0.9381
##          Neg Pred Value : 0.1111
##              Prevalence : 0.9248
##          Detection Rate : 0.6842
##    Detection Prevalence : 0.7293
##       Balanced Accuracy : 0.5699
##
##        'Positive' Class : 1
##
```

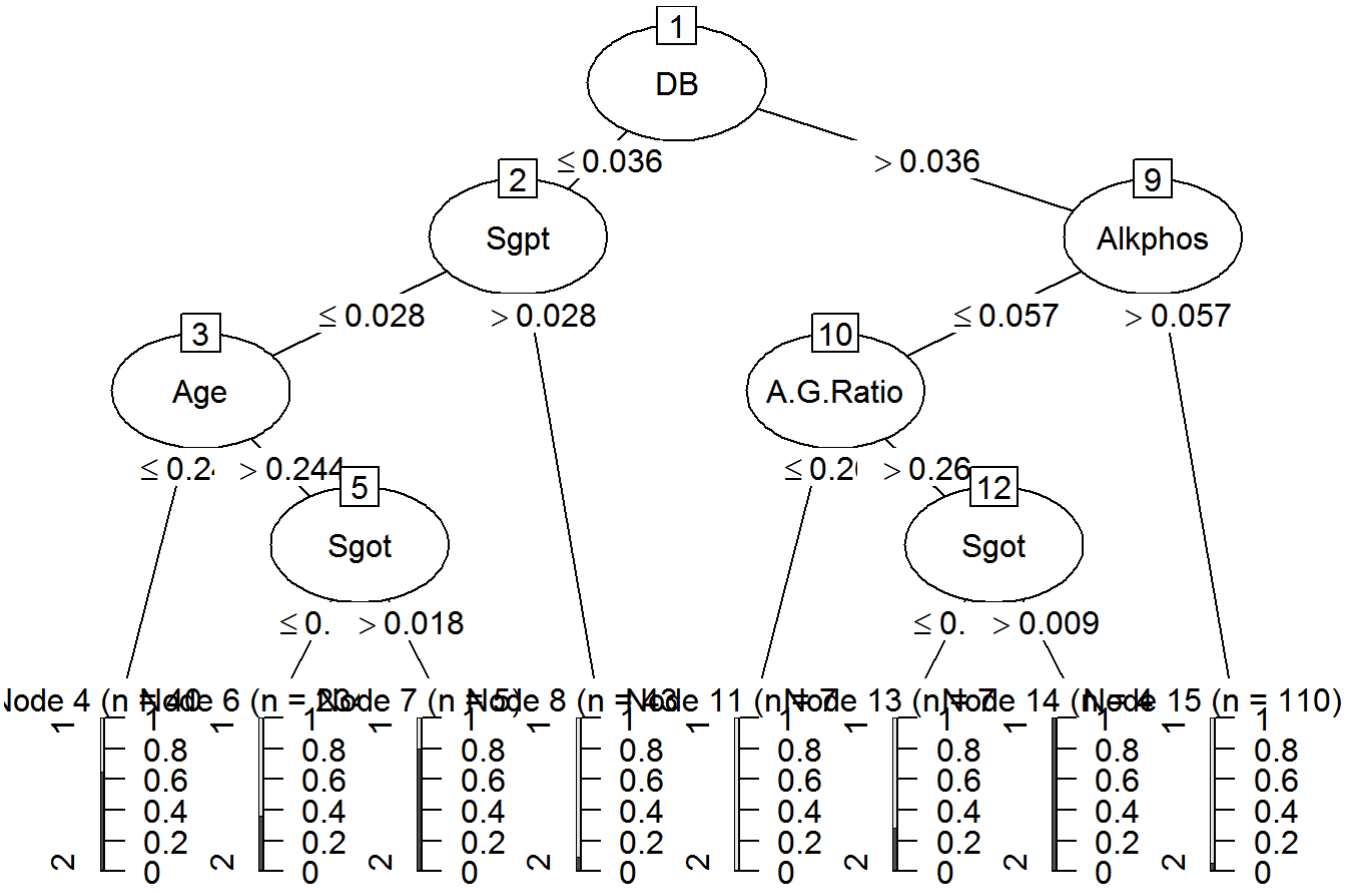
```
#-------------------------------------------------------------------------------#
# Improving the model with (adaptative) boosting
credit_boost10 <- C5.0(trainc50, train_targetc50,
                       trials = 10)
credit_boost10
```

```
##
## Call:
## C5.0.default(x = trainc50, y = train_targetc50, trials = 10)
##
## Classification Tree
## Number of samples: 450
## Number of predictors: 9
##
## Number of boosting iterations: 10 requested;  3 used due to early stopping
## Average tree size: 5.3
##
## Non-standard options: attempt to group attributes
```

```
credit_boost_pred10 <- predict(credit_boost10, testc50)
CrossTable(test_targetc50, credit_boost_pred10,
           prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,
           dnn = c('actual Selector.field', 'predicted Selector.field'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  133
##
##
##                       | predicted Selector.field
## actual Selector.field |         1 |         2 | Row Total |
## ----------------------|-----------|-----------|-----------|
##                     1 |        88 |         9 |        97 |
##                       |     0.662 |     0.068 |           |
## ----------------------|-----------|-----------|-----------|
##                     2 |        24 |        12 |        36 |
##                       |     0.180 |     0.090 |           |
## ----------------------|-----------|-----------|-----------|
##          Column Total |       112 |        21 |       133 |
## ----------------------|-----------|-----------|-----------|
##
##
```

```
plot(credit_boost10)
```




```
confusionMatrix(test_targetc50,credit_boost_pred10)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1 88  9
##          2 24 12
##
##                Accuracy : 0.7519
##                  95% CI : (0.6696, 0.8226)
##     No Information Rate : 0.8421
##     P-Value [Acc > NIR] : 0.99749
##
##                   Kappa : 0.2768
##  Mcnemar's Test P-Value : 0.01481
##
##             Sensitivity : 0.7857
##             Specificity : 0.5714
##          Pos Pred Value : 0.9072
##          Neg Pred Value : 0.3333
##              Prevalence : 0.8421
##          Detection Rate : 0.6617
##    Detection Prevalence : 0.7293
##       Balanced Accuracy : 0.6786
##
##        'Positive' Class : 1
##
```

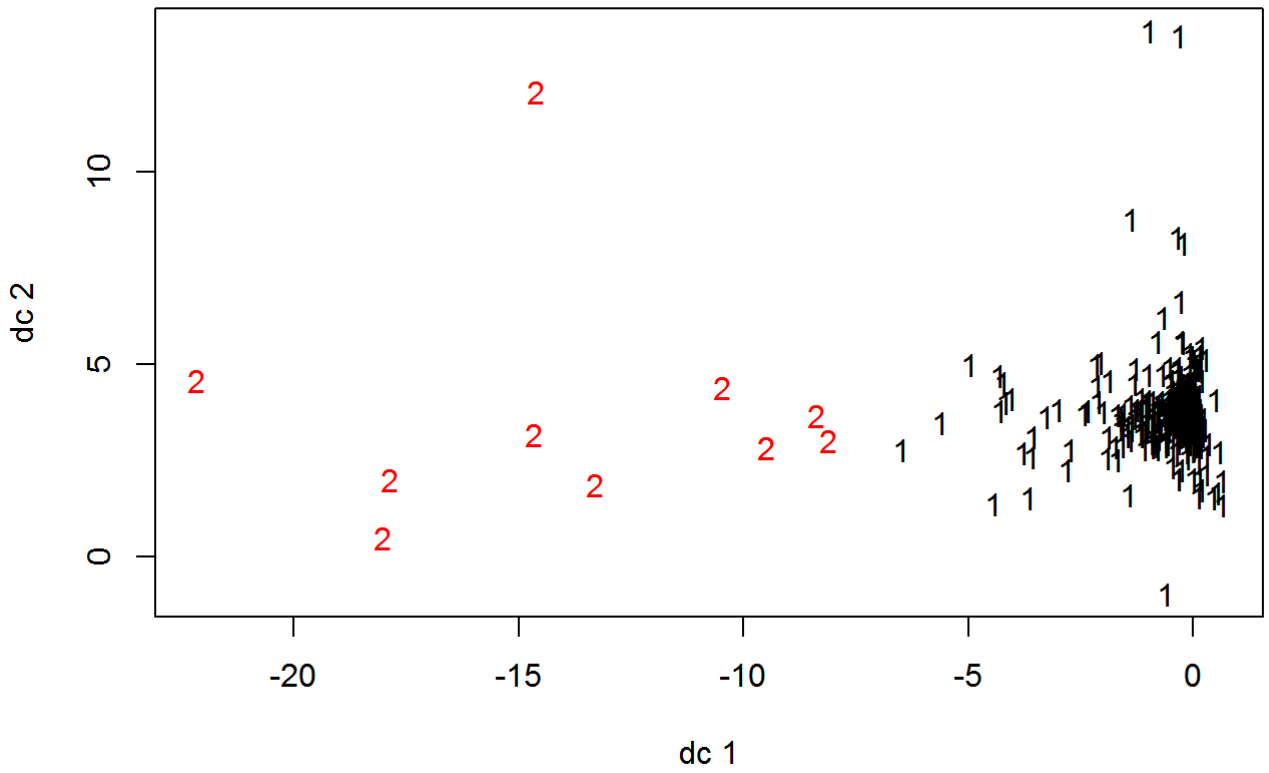
```
#------------------ Prepration of the Data to be implemented in kmeans algorithm ------------
------------#
data.liver1<-data.liver
data.liver1$Selector.field <- as.factor(data.liver1$Selector.field)
str(data.liver1)
```

```
## 'data.frame':    583 obs. of  10 variables:
##  $ Age           : int  65 62 62 58 72 46 26 29 17 55 ...
##  $ TB            : num  0.7 10.9 7.3 1 3.9 1.8 0.9 0.9 0.9 0.7 ...
##  $ DB            : num  0.1 5.5 4.1 0.4 2 0.7 0.2 0.3 0.3 0.2 ...
##  $ Alkphos       : int  187 699 490 182 195 208 154 202 202 290 ...
##  $ Sgpt          : int  16 64 60 14 27 19 16 14 22 53 ...
##  $ Sgot          : int  18 100 68 20 59 14 12 11 19 58 ...
##  $ TP            : num  6.8 7.5 7 6.8 7.3 7.6 7 6.7 7.4 6.8 ...
##  $ ALB           : num  3.3 3.2 3.3 3.4 2.4 4.4 3.5 3.6 4.1 3.4 ...
##  $ A.G.Ratio     : num  0.9 0.74 0.89 1 0.4 1.3 1 1.1 1.2 1 ...
##  $ Selector.field: Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 2 1 ...
```

```
#############################################################################################
#                                  K-means ALGORITHM                                        #
#############################################################################################

kmeans_model <- kmeans(x = subset(data.liver1, select = -Selector.field),
                       centers = 2)

# Printing and Ploting result of algorithm
str(kmeans_model)
```

```
## List of 9
##  $ cluster     : Named int [1:583] 1 1 1 1 1 1 1 1 1 1 ...
##   ..- attr(*, "names")= chr [1:583] "1" "2" "3" "4" ...
##  $ centers     : num [1:2, 1:9] 44.94 34.91 3.17 9.81 1.43 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:2] "1" "2"
##   .. ..$ : chr [1:9] "Age" "TB" "DB" "Alkphos" ...
##  $ totss       : num 1.03e+08
##  $ withinss    : num [1:2] 4.4e+07 2.0e+07
##  $ tot.withinss: num 6.4e+07
##  $ betweenss   : num 38506270
##  $ size        : int [1:2] 572 11
##  $ iter        : int 1
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

```
plotcluster(data.liver1[,-10], kmeans_model$cluster)
```




```
# Summarizing the Algorithm's Result
summary(kmeans_model)
```

```
##              Length Class  Mode
## cluster      583    -none- numeric
## centers       18    -none- numeric
## totss          1    -none- numeric
## withinss       2    -none- numeric
## tot.withinss   1    -none- numeric
## betweenss      1    -none- numeric
## size           2    -none- numeric
## iter           1    -none- numeric
## ifault         1    -none- numeric
```

```
# Making table of Comfusion matrix
mtab <- table(data.liver1$Selector.field,kmeans_model$cluster)
CrossTable(kmeans_model$cluster, data.liver1$Selector.field,
           prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,
           dnn = c('actual Selector.Field', 'predicted Selector.field'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  583
##
##
##                      | predicted Selector.field
## actual Selector.Field |         1 |         2 | Row Total |
## ---------------------|-----------|-----------|-----------|
##                    1 |       405 |       167 |       572 |
##                      |     0.695 |     0.286 |           |
## ---------------------|-----------|-----------|-----------|
##                    2 |        11 |         0 |        11 |
##                      |     0.019 |     0.000 |           |
## ---------------------|-----------|-----------|-----------|
##          Column Total |       416 |       167 |       583 |
## ---------------------|-----------|-----------|-----------|
##
##
```

```
# Printing the result using confusion matrix
confusionMatrix(mtab)
```

```
## Confusion Matrix and Statistics
##
##
##       1   2
##   1 405  11
##   2 167   0
##
##                Accuracy : 0.6947
##                  95% CI : (0.6555, 0.7319)
##     No Information Rate : 0.9811
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : -0.0367
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.7080
##             Specificity : 0.0000
##          Pos Pred Value : 0.9736
##          Neg Pred Value : 0.0000
##              Prevalence : 0.9811
##          Detection Rate : 0.6947
##    Detection Prevalence : 0.7136
##       Balanced Accuracy : 0.3540
##
##        'Positive' Class : 1
##
```

```
#------------ Preparing Data to be implemented in Naive Bayes Algorithm ----------#
train_naive <- liver_normal[1:450,c(1:9)]
train_target_naive <- liver_normal[1:450,10]
tester_naive <- liver_normal[451:583,c(1:9) ]
tester_target_naive <- liver_normal[451:583,10]

##############################################################################
#                          Applying NAIVE BAYES Algorithm                    #
##############################################################################

bayes_model <- naiveBayes(x = train_naive, y = train_target_naive)

# Summary of the Algorithm
summary(bayes_model)
```

```
##         Length Class  Mode
## apriori 2      table  numeric
## tables  9      -none- list
## levels  2      -none- character
## call    3      -none- call
```

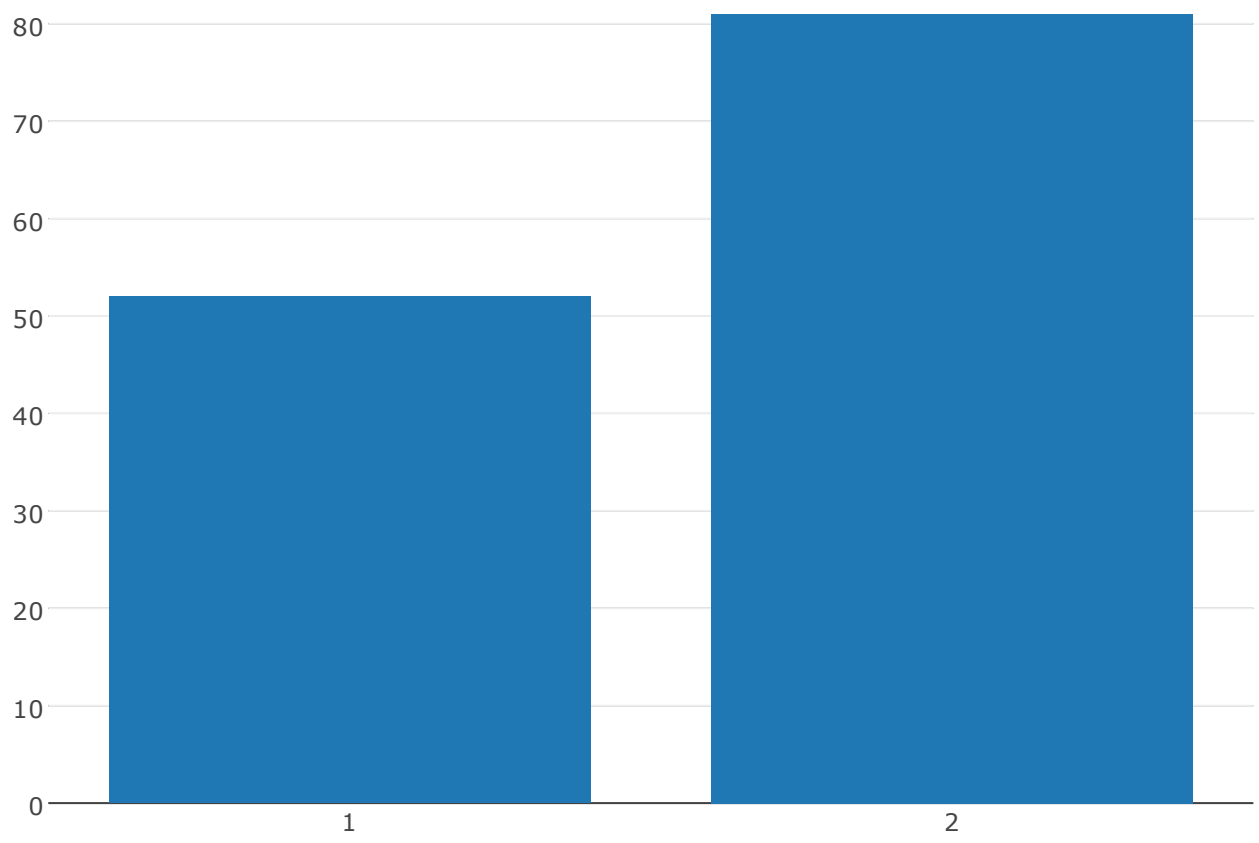
```
# structure result of Algorithm
str(bayes_model)
```

```
## List of 4
##  $ apriori: 'table' int [1:2(1d)] 319 131
##   ..- attr(*, "dimnames")=List of 1
##   .. ..$ train_target_naive: chr [1:2] "1" "2"
##  $ tables :List of 9
##   ..$ Age      : num [1:2, 1:2] 0.493 0.429 0.184 0.206
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ train_target_naive: chr [1:2] "1" "2"
##   .. .. ..$ Age               : NULL
##   ..$ TB       : num [1:2, 1:2] 0.0392 0.0109 0.0833 0.0148
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ train_target_naive: chr [1:2] "1" "2"
##   .. .. ..$ TB                : NULL
##   ..$ DB       : num [1:2, 1:2] 0.0698 0.0169 0.1257 0.0291
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ train_target_naive: chr [1:2] "1" "2"
##   .. .. ..$ DB                : NULL
##   ..$ Alkphos  : num [1:2, 1:2] 0.1317 0.0788 0.1422 0.075
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ train_target_naive: chr [1:2] "1" "2"
##   .. .. ..$ Alkphos           : NULL
##   ..$ Sgpt     : num [1:2, 1:2] 0.0472 0.0122 0.1175 0.0131
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ train_target_naive: chr [1:2] "1" "2"
##   .. .. ..$ Sgpt              : NULL
##   ..$ Sgot     : num [1:2, 1:2] 0.02631 0.00627 0.07636 0.00785
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ train_target_naive: chr [1:2] "1" "2"
##   .. .. ..$ Sgot              : NULL
##   ..$ TP       : num [1:2, 1:2] 0.539 0.551 0.161 0.153
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ train_target_naive: chr [1:2] "1" "2"
##   .. .. ..$ TP                : NULL
##   ..$ ALB      : num [1:2, 1:2] 0.477 0.539 0.172 0.167
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ train_target_naive: chr [1:2] "1" "2"
##   .. .. ..$ ALB               : NULL
##   ..$ A.G.Ratio: num [1:2, 1:2] 0.249 0.304 0.113 0.111
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ train_target_naive: chr [1:2] "1" "2"
##   .. .. ..$ A.G.Ratio         : NULL
##  $ levels : chr [1:2] "1" "2"
##  $ call   : language naiveBayes.default(x = train_naive, y = train_target_naive)
##  - attr(*, "class")= chr "naiveBayes"
```

```
# Testing the model
result<-predict(object = bayes_model,newdata = tester_naive)
print(result)
```

```
##   [1] 1 2 1 2 2 2 2 2 1 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 2 1 1 2 1
##  [36] 1 2 1 2 2 2 2 1 2 1 2 2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 1 2 2 2 1
##  [71] 2 1 2 2 2 2 1 1 2 1 2 1 2 1 2 1 1 2 2 2 2 2 2 2 1 2 1 1 1 1 2 2 2 1 2
## [106] 1 2 2 1 1 1 1 1 2 2 1 2 2 1 1 1 2 1 1 1 1 1 1 2 2 2 2 2
## Levels: 1 2
```

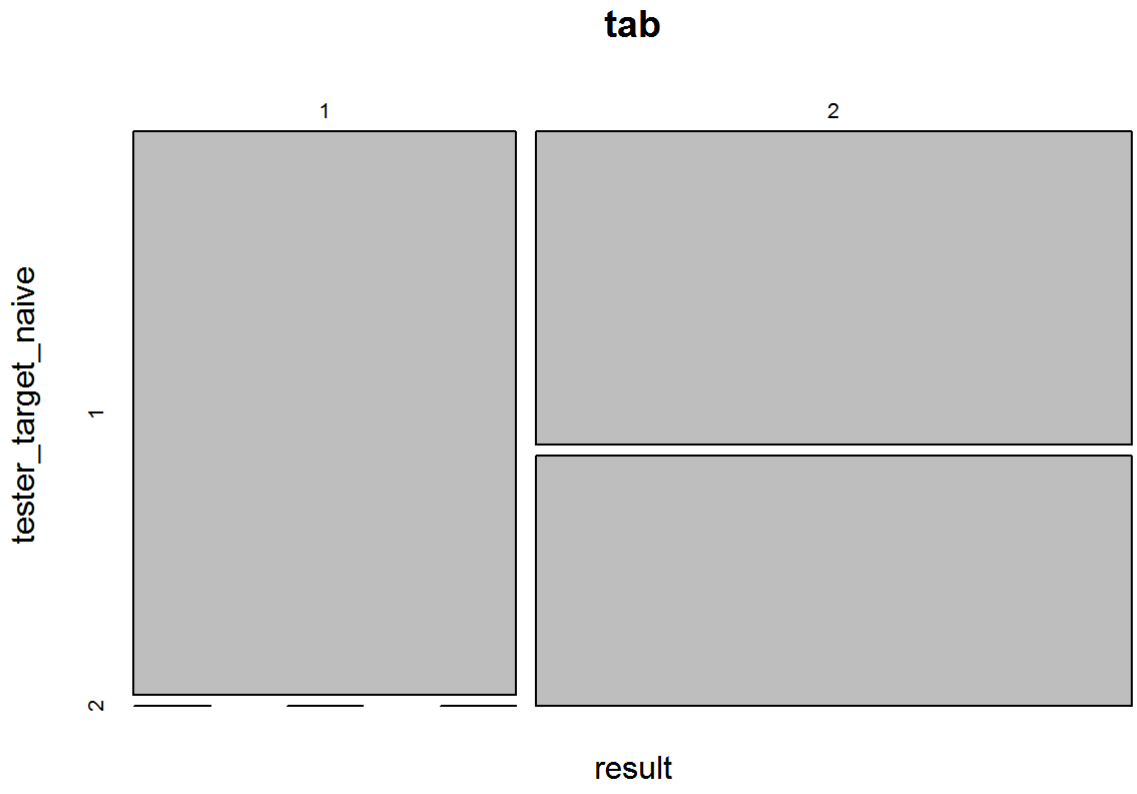
```
x<-rchisq(100,5,0)
plot_ly(x=result,type = 'histogram')
```




```
# Making table of Confusion matrix
tab<-table(result,tester_target_naive)
CrossTable(tester_target_naive, result,
           prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE,
           dnn = c('actual Selector.Field', 'predicted Selector.field'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  133
##
##
##                      | predicted Selector.field
## actual Selector.Field |         1 |         2 | Row Total |
## ---------------------|-----------|-----------|-----------|
##                    1 |        52 |        45 |        97 |
##                      |     0.391 |     0.338 |           |
## ---------------------|-----------|-----------|-----------|
##                    2 |         0 |        36 |        36 |
##                      |     0.000 |     0.271 |           |
## ---------------------|-----------|-----------|-----------|
##          Column Total |        52 |        81 |       133 |
## ---------------------|-----------|-----------|-----------|
##
##
```

```
plot(tab)
```



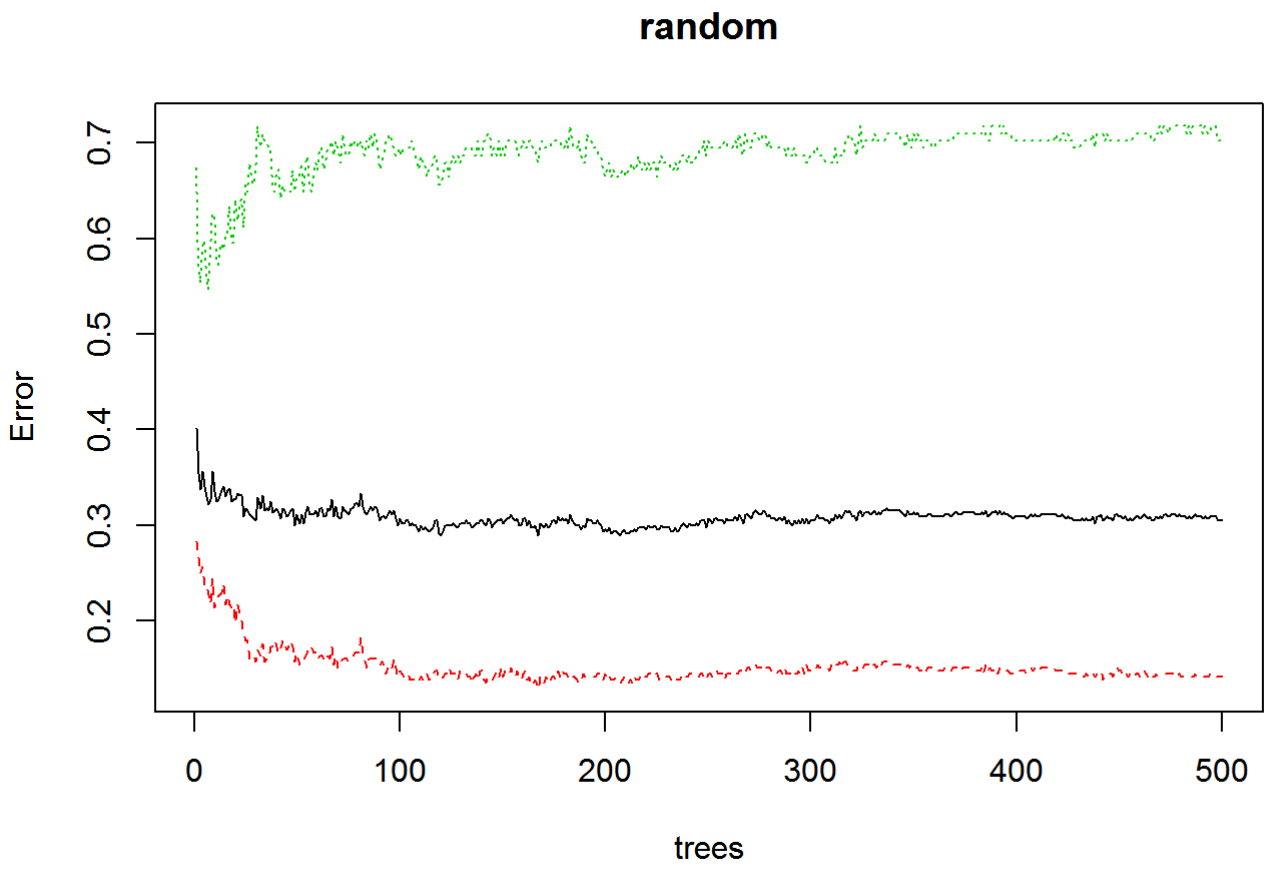
```
# Printing the result using confusion matrix
confusionMatrix(tab)
```

```
## Confusion Matrix and Statistics
##
##       tester_target_naive
## result  1  2
##      1 52  0
##      2 45 36
##
##                Accuracy : 0.6617
##                  95% CI : (0.5746, 0.7414)
##     No Information Rate : 0.7293
##     P-Value [Acc > NIR] : 0.9658
##
##                   Kappa : 0.3848
##  Mcnemar's Test P-Value : 5.412e-11
##
##             Sensitivity : 0.5361
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.4444
##              Prevalence : 0.7293
##          Detection Rate : 0.3910
##    Detection Prevalence : 0.3910
##       Balanced Accuracy : 0.7680
##
##        'Positive' Class : 1
##
```

```
#-----------------------  Prepairing Data to be implemented in this Algorithm --------------
------------#
train_rf <- liver_normal[1:450,c(1:9)]
test_rf <- liver_normal[451:583,]
train_target_rf <- liver_normal[1:450,10]
test_target_rf <- liver_normal[451:583,10]

#############################################################################
#                          Applying RANDOM FOREST Algorithm                 #
#############################################################################

random <- randomForest(train_rf,train_target_rf,ntree = 500)

# Printing and Ploting result of algorithm
print(random)
```

```
##
## Call:
##  randomForest(x = train_rf, y = train_target_rf, ntree = 500)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 30.44%
## Confusion matrix:
##     1  2 class.error
## 1 274 45   0.1410658
## 2  92 39   0.7022901
```

```
plot(random)
```




```
# Summarizing the Algorithm's Result
summary(random)
```

```
##                 Length Class  Mode
## call               4   -none- call
## type               1   -none- character
## predicted        450   factor numeric
## err.rate        1500   -none- numeric
## confusion          6   -none- numeric
## votes            900   matrix numeric
## oob.times        450   -none- numeric
## classes            2   -none- character
## importance         9   -none- numeric
## importanceSD       0   -none- NULL
## localImportance    0   -none- NULL
## proximity          0   -none- NULL
## ntree              1   -none- numeric
## mtry               1   -none- numeric
## forest            14   -none- list
## y                450   factor numeric
## test               0   -none- NULL
## inbag              0   -none- NULL
```

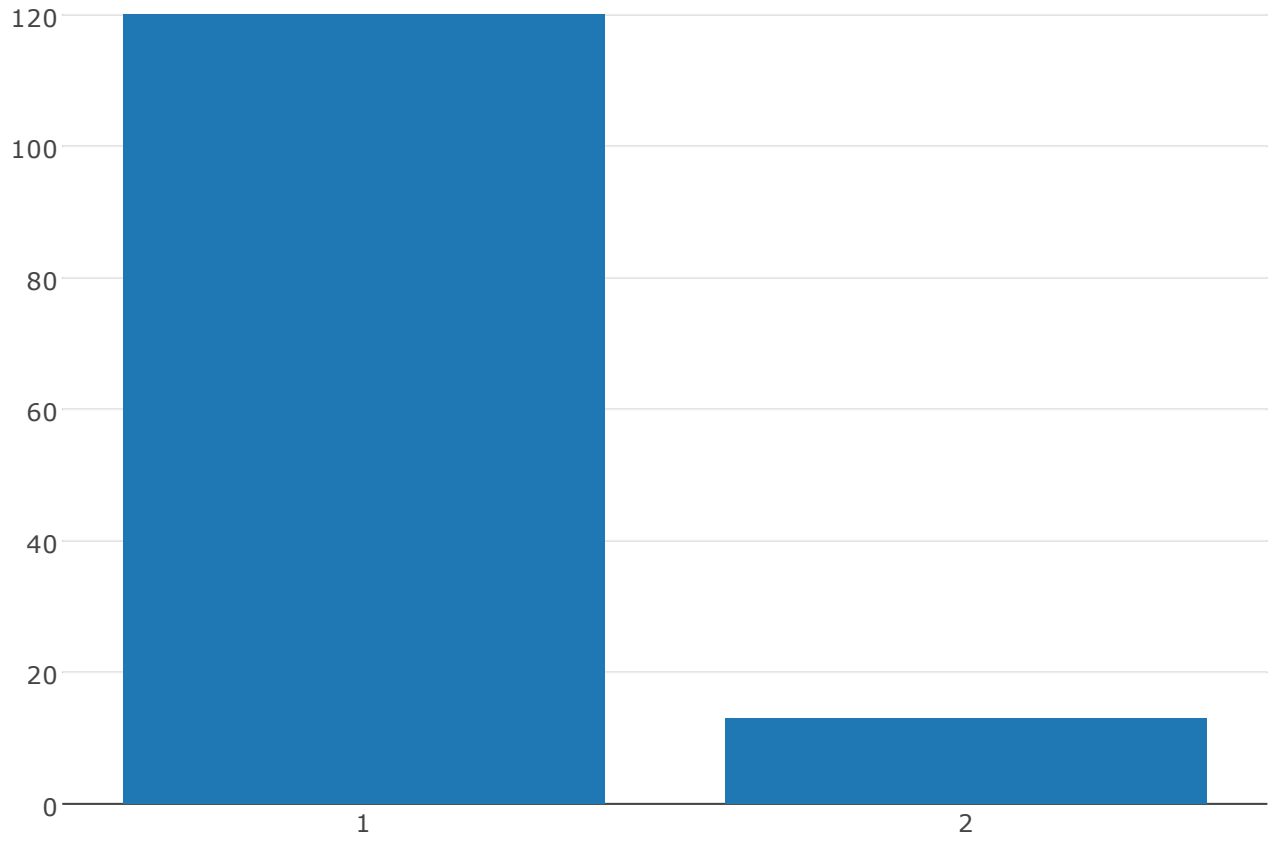
```
# Testing the model
predic <- predict(random,test_rf)
print(predic)
```

```
## 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468
##   1   1   1   1   1   2   1   1   1   1   1   1   1   2   1   1   1   1
## 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486
##   1   1   2   1   1   1   1   1   1   1   1   1   2   1   1   1   1   1
## 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504
##   1   1   1   2   1   2   1   1   1   1   2   1   2   1   1   1   1   1
## 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522
##   1   1   1   1   1   1   1   1   1   1   1   1   2   1   1   1   1   1
## 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540
##   1   2   1   1   1   1   1   1   2   1   1   1   1   1   1   1   1   1
## 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558
##   1   2   1   1   1   1   1   1   1   1   1   2   1   1   1   1   1   1
## 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
## 577 578 579 580 581 582 583
##   1   1   1   1   1   1   1
## Levels: 1 2
```

```
# Making table of Confusion matrix
rtab<-table(predic,test_target_rf)
CrossTable(test_target_rf, predic,prop.chisq = FALSE,
           prop.c = FALSE,
           prop.r = FALSE,
           dnn = c('actual Selector.Field',
                   'predicted Selector.field'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  133
##
##
##                      | predicted Selector.field
## actual Selector.Field |         1 |         2 | Row Total |
## ----------------------|-----------|-----------|-----------|
##                     1 |        90 |         7 |        97 |
##                       |     0.677 |     0.053 |           |
## ----------------------|-----------|-----------|-----------|
##                     2 |        30 |         6 |        36 |
##                       |     0.226 |     0.045 |           |
## ----------------------|-----------|-----------|-----------|
##          Column Total |       120 |        13 |       133 |
## ----------------------|-----------|-----------|-----------|
##
##
```

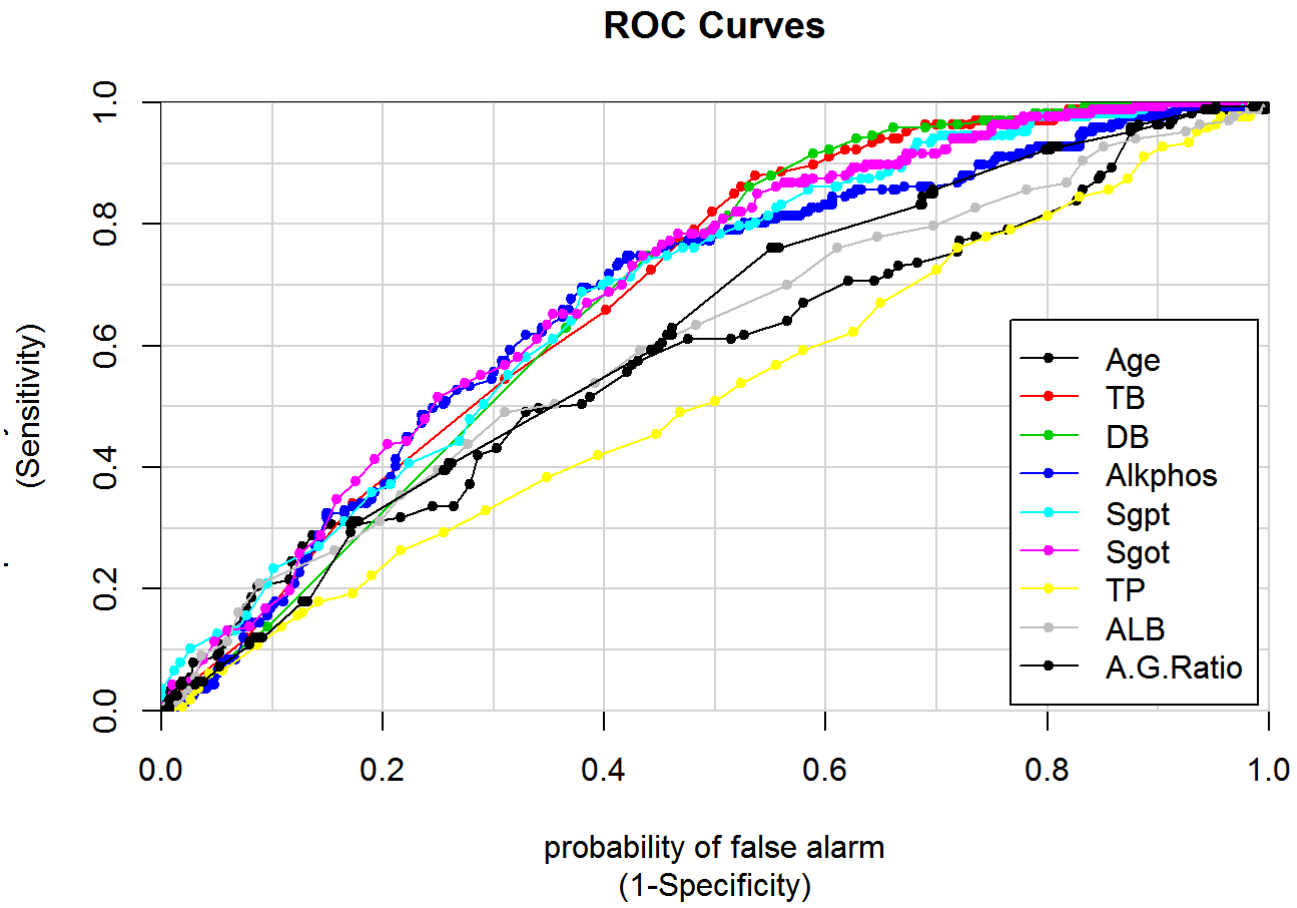
```
x<-rchisq(1000,5,0)
plot_ly(x=predic,type = 'histogram')
```



```
# Printing the result using confusion matrix
confusionMatrix(predic,test_target_rf)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1 90 30
##          2  7  6
##
##                Accuracy : 0.7218
##                  95% CI : (0.6375, 0.796)
##     No Information Rate : 0.7293
##     P-Value [Acc > NIR] : 0.6202821
##
##                   Kappa : 0.1183
##  Mcnemar's Test P-Value : 0.0002983
##
##             Sensitivity : 0.9278
##             Specificity : 0.1667
##          Pos Pred Value : 0.7500
##          Neg Pred Value : 0.4615
##              Prevalence : 0.7293
##          Detection Rate : 0.6767
##    Detection Prevalence : 0.9023
##       Balanced Accuracy : 0.5473
##
##        'Positive' Class : 1
##
```

```
#############################################################################################
# ROC curve for Whole Orignal dataset befor implementing algorithms.

colAUC(liver_normal[,-10], liver_normal[,10],
       plotROC=TRUE,alg=c("Wilcoxon","ROC"))
```




```
##               Age        TB       DB  Alkphos      Sgpt      Sgot
## 1 vs. 2 0.5827024 0.6932793 0.686075 0.674466 0.6855856 0.6972161
##                TP       ALB A.G.Ratio
## 1 vs. 2 0.5205622 0.6065897 0.6190264
```

```
cat("Total AUC: \n");
```

```
## Total AUC:
```

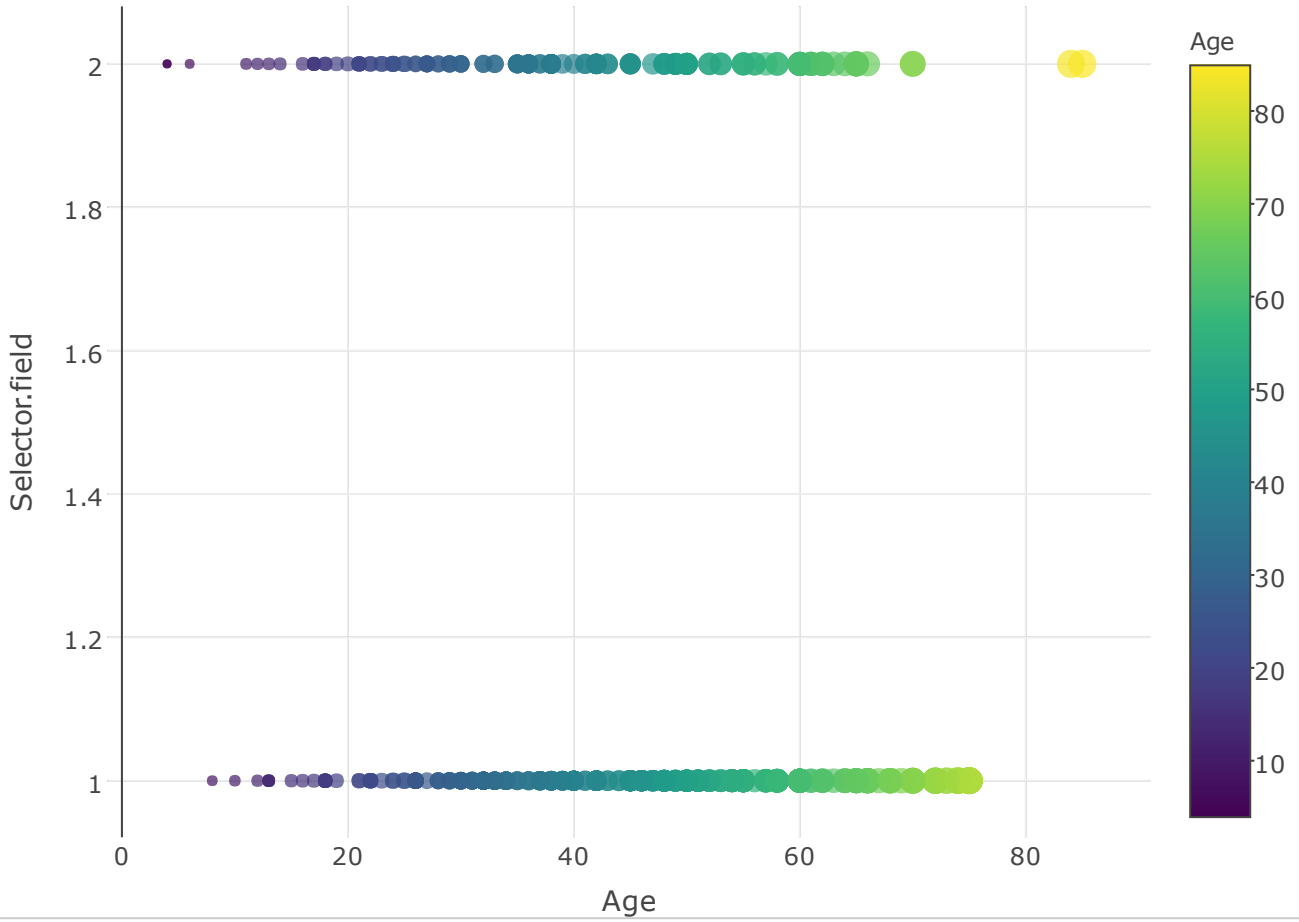
```
colMeans(colAUC(liver_normal[,-10], liver_normal[,10]))
```

```
##       Age        TB        DB    Alkphos      Sgpt      Sgot        TP
## 0.5827024 0.6932793 0.6860750 0.6744660 0.6855856 0.6972161 0.5205622
##       ALB A.G.Ratio
## 0.6065897 0.6190264
```

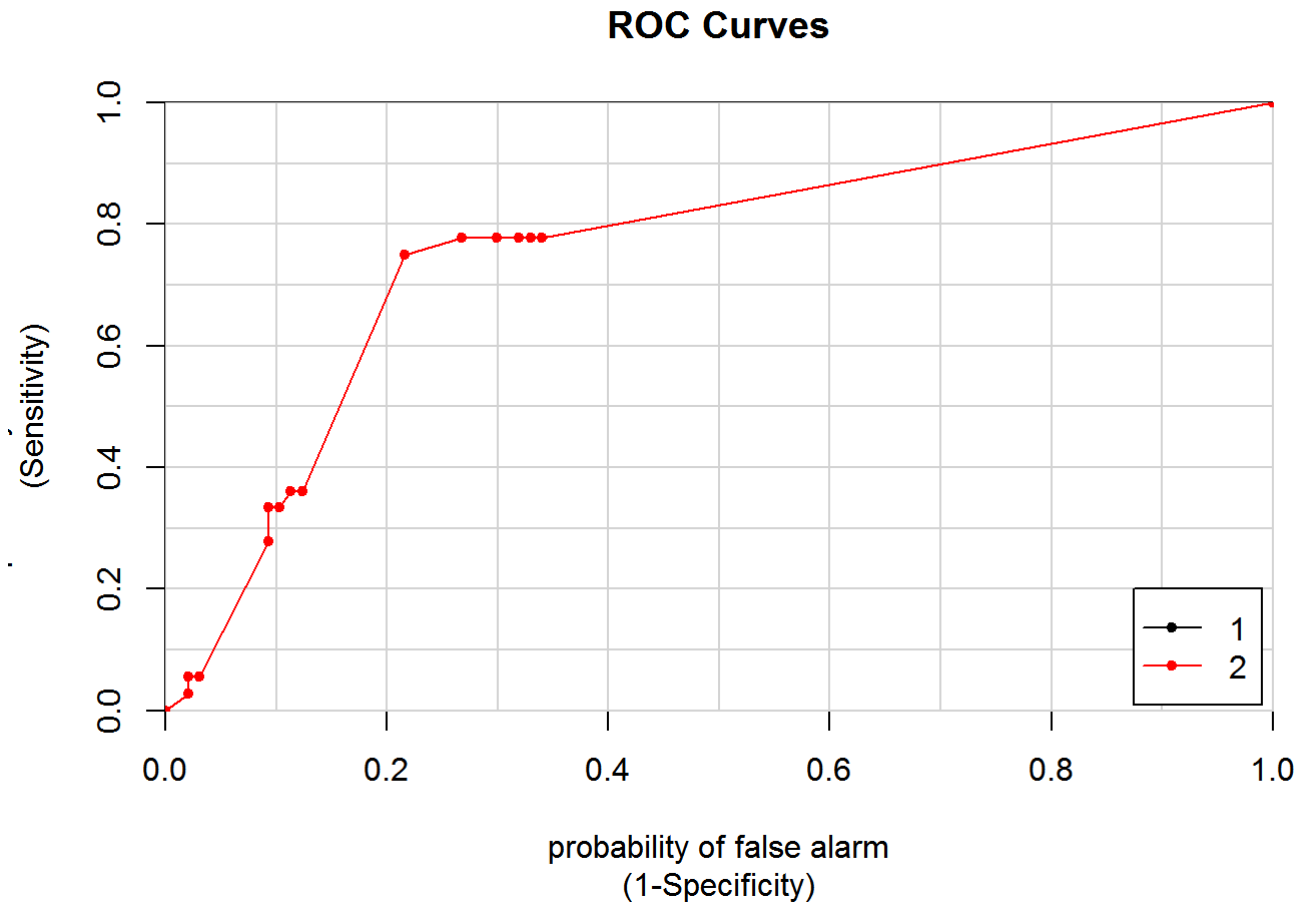
```
# Plot patients in there age groups
set.seed(100)
d <- data.liver[sample(nrow(data.liver), 500), ]
plot_ly(d, x = ~Age, y = ~Selector.field, color = ~Age,
        size = ~Age, text = ~paste("A.G.Ratio: ", A.G.Ratio))
```

```
## No trace type specified:
##   Based on info supplied, a 'scatter' trace seems appropriate.
##   Read more about this trace type -> https://plot.ly/r/reference/#scatter
```

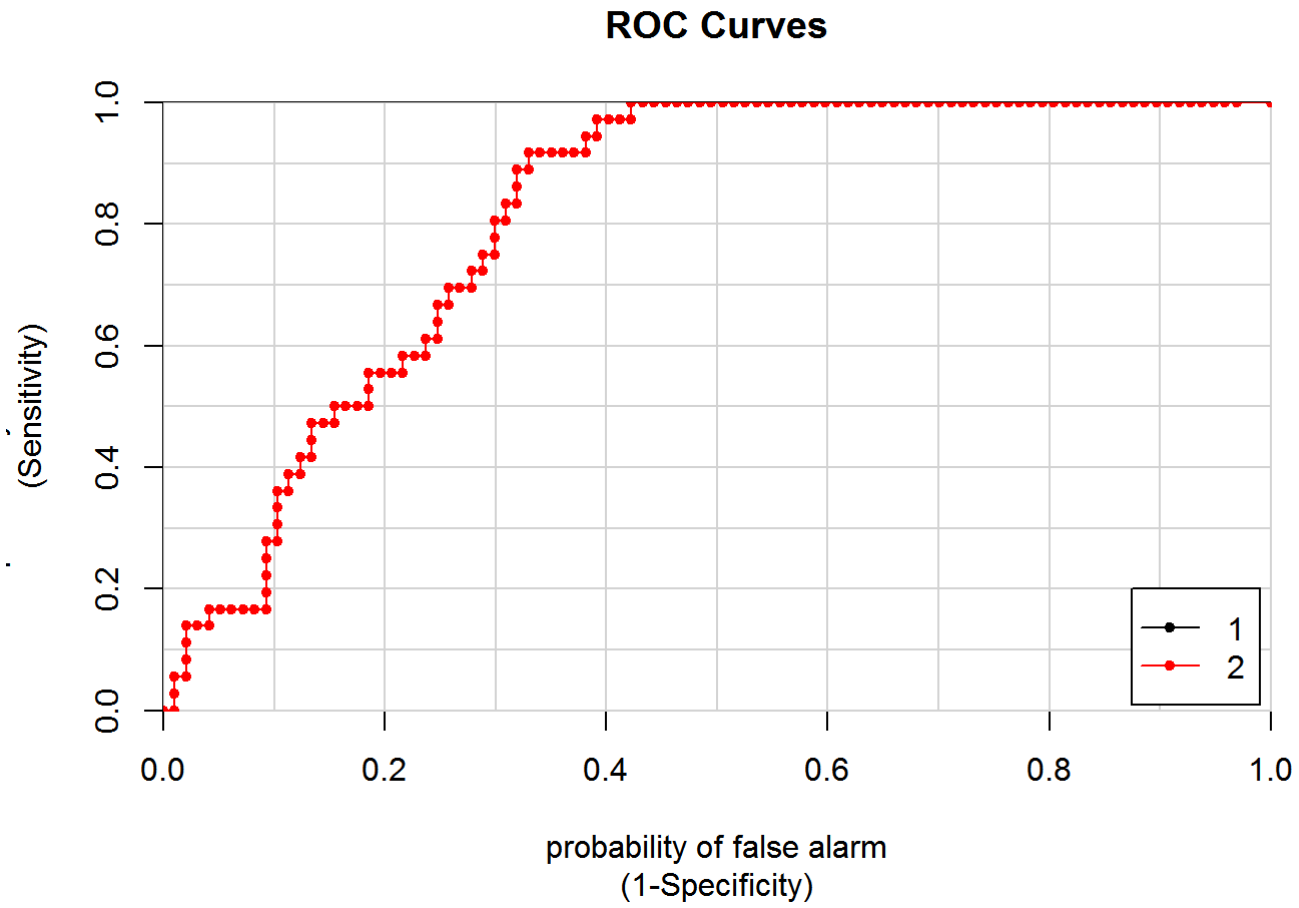
```
## No scatter mode specifed:
##   Setting the mode to markers
##   Read more about this attribute -> https://plot.ly/r/reference/#scatter-mode
```



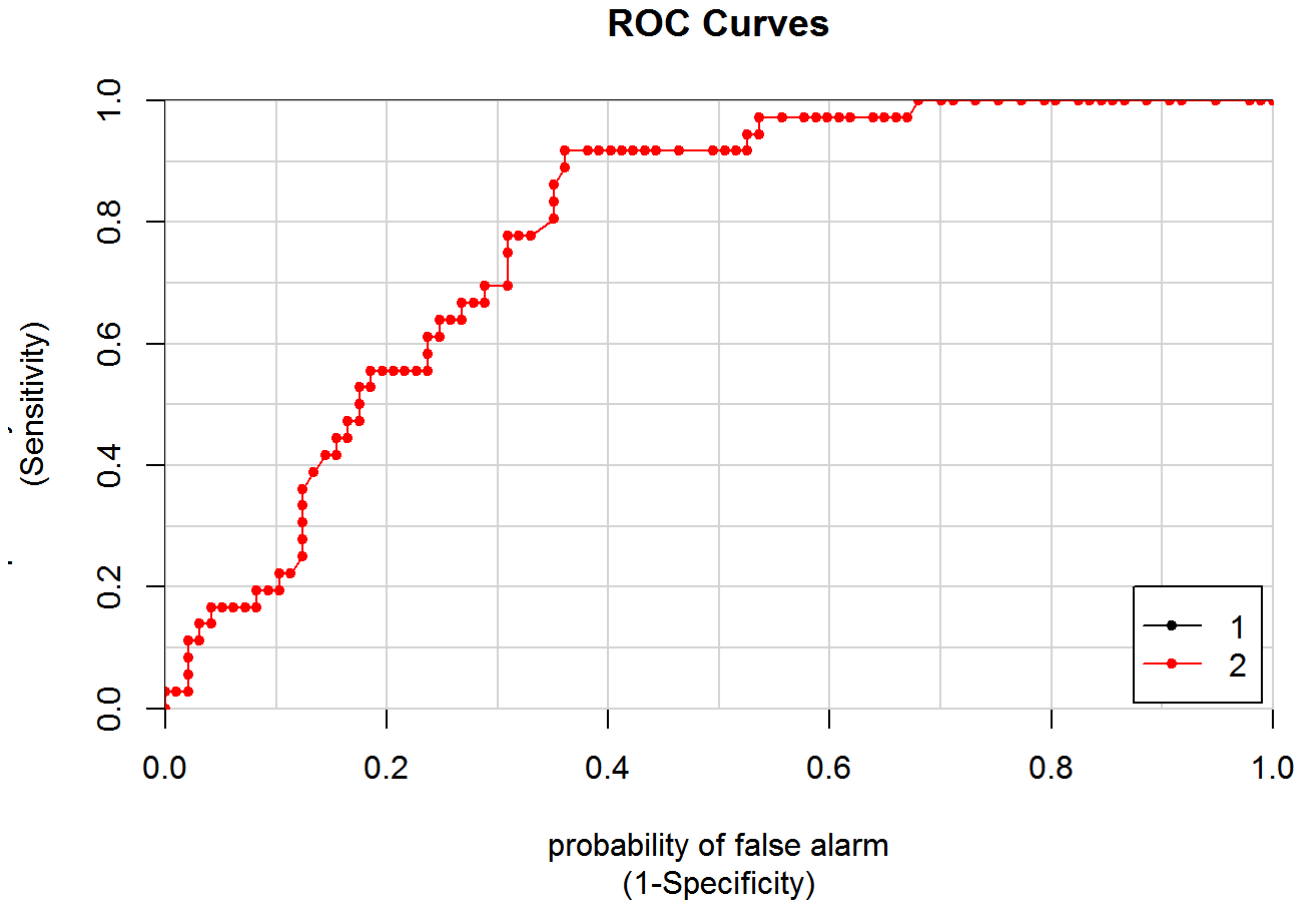

```
#--------------------- ROC Curve for C5.0 Algorithm ----------------------------#
c5.0_predict <- predict(credit_boost10,testc50,type="prob")
c5.0_ROC <- colAUC(c5.0_predict,testc50$Selector.field,
                   plotROC = TRUE,
                   alg=c("Wilcoxon","ROC"))
```




```
#--------------------- ROC curve for Naive bayes Algorithm ----------------------#
test_nb <- liver_normal[451:583,]
nb_predict <- predict(bayes_model,test_nb,type="raw")
nb_ROC <- colAUC(nb_predict,test_nb$Selector.field,
                 plotROC = TRUE,
                 alg=c("Wilcoxon","ROC"))
```




```
#--------------------- ROC curve for Random Forest -----------------------------#
rf_predict <- predict(random,test_rf,type="prob")
rf_ROC <- colAUC(rf_predict,test_rf$Selector.field,
                 plotROC = TRUE,
                 alg=c("Wilcoxon","ROC"))
```


ROC Curves
(Sensitivity)
0.0
0.2
0.4
0.6
0.8
1.0
1
2
probability of false alarm
(1-Specificity)