NaiveBayes & DecisionTree Classifiers

Team.8

17 Januar 2018

Business Intelligence - Homework.2

Classification on bank-full dataset for predicting the subscribtion of clients

```
#set working directory and import Data
setwd("~/R/work-space")
dataset <- read.csv("bank-full.csv",
                    header = TRUE,
                    sep = ";",
                    stringsAsFactors = TRUE)
```

[1]- Data Exploration

```
#view the structure of data
str(dataset)
```

```
## 'data.frame':    45211 obs. of  17 variables:
##  $ age      : int  58 44 33 47 33 35 28 42 58 43 ...
##  $ job      : Factor w/ 12 levels "admin.","blue-collar",..: 5 10 3 2 12 5 5 3 6 10 ...
##  $ marital  : Factor w/ 3 levels "divorced","married",..: 2 3 2 2 3 2 3 1 2 3 ...
##  $ education: Factor w/ 4 levels "primary","secondary",..: 3 2 2 4 4 3 3 3 1 2 ...
##  $ default  : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 2 1 1 ...
##  $ balance  : int  2143 29 2 1506 1 231 447 2 121 593 ...
##  $ housing  : Factor w/ 2 levels "no","yes": 2 2 2 2 1 2 2 2 2 2 ...
##  $ loan     : Factor w/ 2 levels "no","yes": 1 1 2 1 1 1 2 1 1 1 ...
##  $ contact  : Factor w/ 3 levels "cellular","telephone",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ day      : int  5 5 5 5 5 5 5 5 5 5 ...
##  $ month    : Factor w/ 12 levels "apr","aug","dec",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ duration : int  261 151 76 92 198 139 217 380 50 55 ...
##  $ campaign : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ pdays    : int  -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 ...
##  $ previous : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ poutcome : Factor w/ 4 levels "failure","other",..: 4 4 4 4 4 4 4 4 4 4 ...
##  $ y        : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
```

```
#view the summary of data
summary(dataset)
```

```
##       age                 job          marital          education
##  Min.   :18.00   blue-collar:9732   divorced: 5207   primary  : 6851
##  1st Qu.:33.00   management :9458   married :27214   secondary:23202
##  Median :39.00   technician :7597   single  :12790   tertiary :13301
##  Mean   :40.94   admin.     :5171                    unknown  : 1857
##  3rd Qu.:48.00   services   :4154
##  Max.   :95.00   retired    :2264
```

```
##                   (Other)    :6835
##  default        balance       housing      loan            contact
##  no :44396   Min.   : -8019   no :20081   no :37967   cellular :29285
##  yes:  815   1st Qu.:    72   yes:25130   yes: 7244   telephone: 2906
##              Median :   448                           unknown  :13020
##              Mean   :  1362
##              3rd Qu.:  1428
##              Max.   :102127
##
##       day            month          duration         campaign
##  Min.   : 1.00   may    :13766   Min.   :   0.0   Min.   : 1.000
##  1st Qu.: 8.00   jul    : 6895   1st Qu.: 103.0   1st Qu.: 1.000
##  Median :16.00   aug    : 6247   Median : 180.0   Median : 2.000
##  Mean   :15.81   jun    : 5341   Mean   : 258.2   Mean   : 2.764
##  3rd Qu.:21.00   nov    : 3970   3rd Qu.: 319.0   3rd Qu.: 3.000
##  Max.   :31.00   apr    : 2932   Max.   :4918.0   Max.   :63.000
##                  (Other): 6060
##      pdays          previous           poutcome       y
##  Min.   : -1.0   Min.   :  0.0000   failure: 4901   no :39922
##  1st Qu.: -1.0   1st Qu.:  0.0000   other  : 1840   yes: 5289
##  Median : -1.0   Median :  0.0000   success: 1511
##  Mean   : 40.2   Mean   :  0.5803   unknown:36959
##  3rd Qu.: -1.0   3rd Qu.:  0.0000
##  Max.   :871.0   Max.   :275.0000
##
```

```r
#encoding the target vaiable (y) as a factor of two levels
dataset$y = factor(dataset$y,
                   levels = c('yes', 'no'),
                   labels = c(1, 0))
```

```r
#proportions of target vaiable (y)
table(dataset$y)
```

```
##
##     1     0
##  5289 39922
```

```r
prop.table(table(dataset$y))
```

```
##
##         1         0
## 0.1169848 0.8830152
```

[2]- Classifiers Preparation

```r
#splitting the dataset into trainingSet and testSet
#install.packages('caret')
library(caret)
set.seed(123)

#split the dataset to 75% for training and 25% for testing
splitSet <- createDataPartition(y = dataset$y, p = 0.75, list = FALSE)
trainSet <- dataset[splitSet,]
```

```
testSet <- dataset[-splitSet,]
```

```
#check the number of rows in training set and test set
nrow(trainSet)
```

```
## [1] 33909
```

```
nrow(testSet)
```

```
## [1] 11302
```

```
#proportions of trainSet and testSet
prop.table(table(trainSet$y))
```

```
##
##         1         0
## 0.1169896 0.8830104
```

```
prop.table(table(testSet$y))
```

```
##
##         1         0
## 0.1169704 0.8830296
```

[3]- NaiveBayes Model

```
#importing e1071 library for creating our NaiveBayes model
library(e1071)
library(rminer)
```

```
#create the naiveBayes classifier model
nb_model <- naiveBayes(y ~ ., data = trainSet)
nb_model
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##         1         0
## 0.1169896 0.8830104
##
## Conditional probabilities:
##    age
## Y       [,1]     [,2]
##   1 41.57600 13.55578
##   0 40.80429 10.16623
##
##    job
## Y        admin. blue-collar entrepreneur   housemaid  management
##   1 0.114696244 0.133854298  0.022687169 0.021426771 0.245525586
##   0 0.113218890 0.225302251  0.034533431 0.028488411 0.204695745
##    job
```

```
## Y       retired self-employed    services     student  technician
##   1 0.099067305   0.036299471 0.069826065 0.055205445 0.153516511
##   0 0.042715917   0.035234787 0.095451206 0.016465166 0.169427560
##    job
## Y    unemployed     unknown
##   1 0.040080665 0.007814469
##   0 0.027920647 0.006545989
##
##    marital
## Y    divorced   married    single
##   1 0.1199899 0.5177716 0.3622385
##   0 0.1137867 0.6120500 0.2741634
##
##    education
## Y      primary  secondary   tertiary    unknown
##   1 0.11343585 0.46004537 0.37811949 0.04839929
##   0 0.15676975 0.51927059 0.28298043 0.04097923
##
##    default
## Y           no         yes
##   1 0.990420973 0.009579027
##   0 0.980328635 0.019671365
##
##    balance
## Y       [,1]     [,2]
##   1 1797.897 3556.474
##   0 1309.445 3004.460
##
##    housing
## Y         no       yes
##   1 0.6307033 0.3692967
##   0 0.4199452 0.5800548
##
##    loan
## Y         no       yes
##   1 0.9082430 0.0917570
##   0 0.8316412 0.1683588
##
##    contact
## Y     cellular  telephone    unknown
##   1 0.82606504 0.07108646 0.10284850
##   0 0.62607708 0.06342262 0.31050030
##
##    day
## Y       [,1]     [,2]
##   1 15.18125 8.553128
##   0 15.93845 8.317354
##
##    month
## Y           apr         aug         dec         feb         jan
##   1 0.111167129 0.132593900 0.018149735 0.083942526 0.026216284
##   0 0.058713513 0.140972547 0.002939015 0.055540712 0.031728007
##    month
## Y           jul         jun         mar         may         nov
```

```
##   1 0.114948324 0.100579783 0.049407613 0.177968238 0.073859340
##   0 0.158105671 0.119464298 0.005610848 0.319784918 0.089305992
##    month
## Y           oct         sep
##   1 0.059742879 0.051424250
##   0 0.010186360 0.007648120
##
##    duration
## Y       [,1]     [,2]
##   1 534.7046 387.2517
##   0 222.3071 208.3119
##
##    campaign
## Y       [,1]     [,2]
##   1 2.117721 1.919196
##   0 2.862334 3.251196
##
##    pdays
## Y       [,1]      [,2]
##   1 70.45828 119.29641
##   0 35.96239  96.12755
##
##    previous
## Y        [,1]     [,2]
##   1 1.1724225 2.597958
##   0 0.4868412 1.753584
##
##    poutcome
## Y      failure      other    success    unknown
##   1 0.11872952 0.06024704 0.18805142 0.63297202
##   0 0.10517000 0.03850778 0.01375994 0.84256229
```

```r
#making predict on testSet using our naiveBayes model
nb_prediction <- predict(nb_model, testSet, type = "class")
```

[4]- NaiveBayes Confusion Matrix

```r
confusionMatrix(nb_prediction, testSet$y)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    1    0
##          1  663  848
##          0  659 9132
##
##                Accuracy : 0.8667
##                  95% CI : (0.8603, 0.8729)
##     No Information Rate : 0.883
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.3922
##  Mcnemar's Test P-Value : 1.28e-06
```

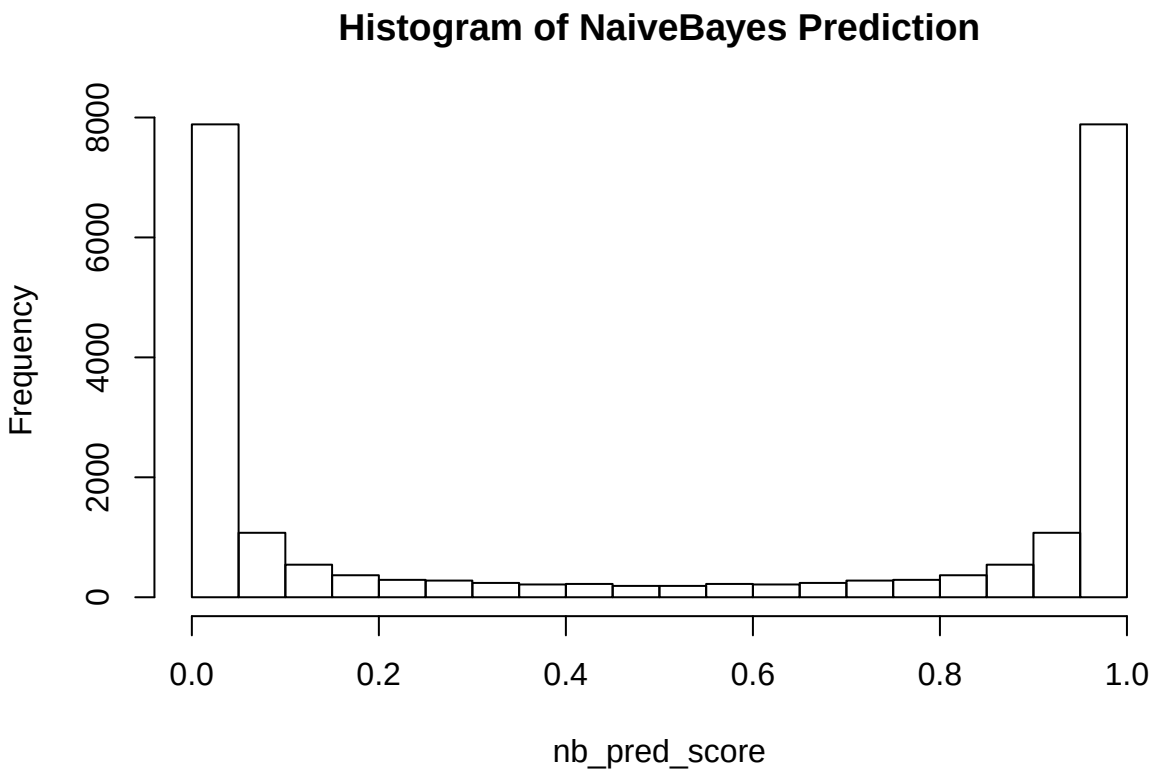
```
##
##             Sensitivity : 0.50151
##             Specificity : 0.91503
##          Pos Pred Value : 0.43878
##          Neg Pred Value : 0.93269
##              Prevalence : 0.11697
##          Detection Rate : 0.05866
##    Detection Prevalence : 0.13369
##       Balanced Accuracy : 0.70827
##
##        'Positive' Class : 1
##
```

```r
#view some metrics about our naiveBayes prediction like Accuracy and TPR compared
#to the actual value of 'y' variable
mmetric(testSet$y, nb_prediction, c("ACC", "PRECISION", "TPR", "F1"))
```

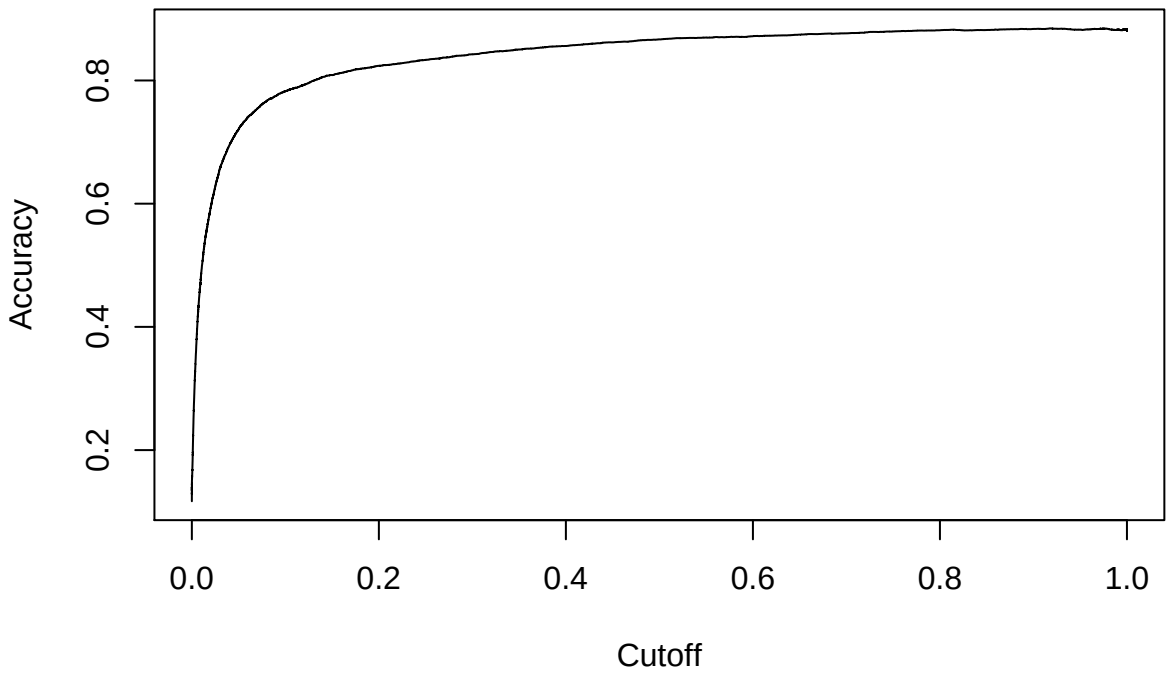
```
##        ACC PRECISION1 PRECISION2       TPR1       TPR2        F11
##   86.66608   43.87823   93.26933   50.15129   91.50301   46.80551
##        F12
##   92.37772
```

[5]- NaiveBayes Model Evaluation

```r
library(ROCR)
nb_pred_score = predict(nb_model, newdata = testSet, type = 'raw')
#plotting the histogram of naiveBayes prediction
hist(nb_pred_score, main = "Histogram of NaiveBayes Prediction")
```

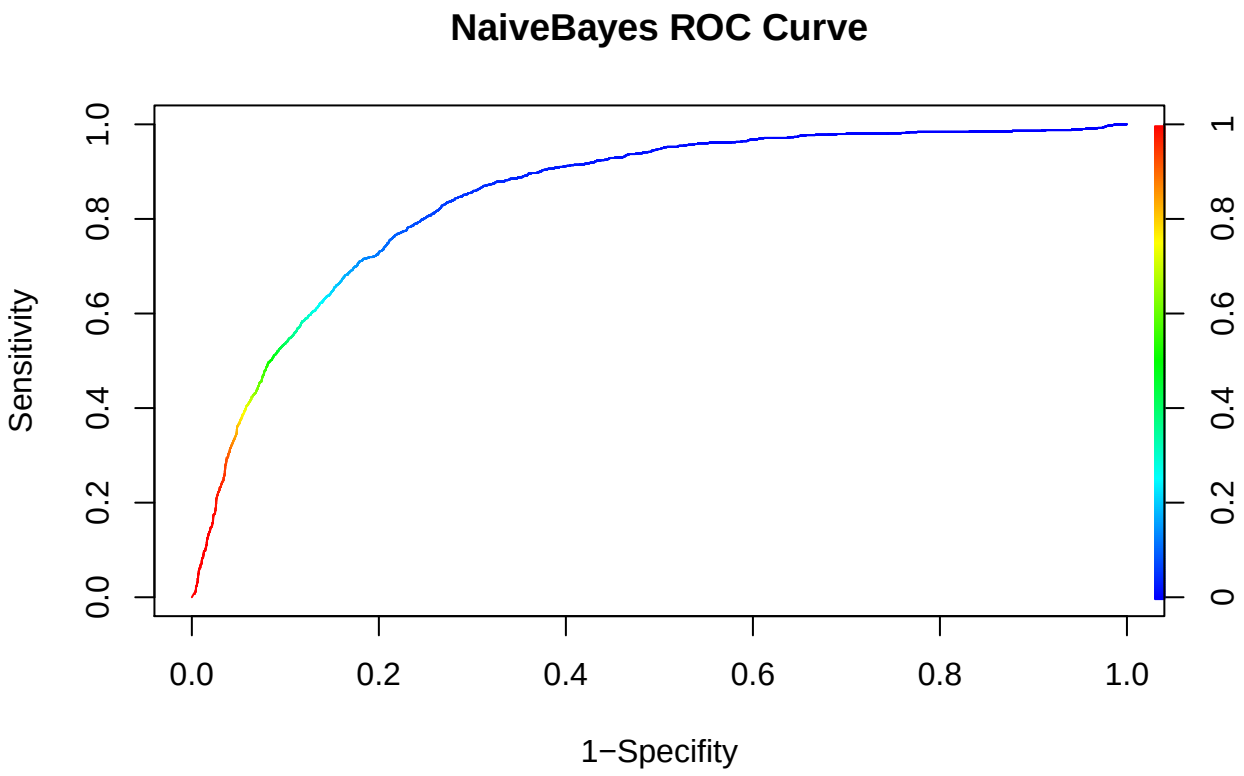



```
nb_pred = prediction(nb_pred_score[,1], testSet$y)
nb_eval = performance(nb_pred, "acc")
#plotting the performance of naiveBayes prediction
plot(nb_eval)
```



[6]- NaiveBayes ROC Curve

```
nb_roc = performance(nb_pred, "tpr", "fpr")
plot(nb_roc, colorize=T, main="NaiveBayes ROC Curve", ylab="Sensitivity", xlab="1-Specifity")
```



[7]- DecisionTree Model

```
#importing C50 library for creating our DecisionTree model
#install.packages('C50')
library(C50)
```

```
#install.packages('C50')
library(C50)
dt_model <- C5.0(trainSet[-17], trainSet$y)
dt_model
```

```
##
## Call:
## C5.0.default(x = trainSet[-17], y = trainSet$y)
##
## Classification Tree
## Number of samples: 33909
## Number of predictors: 16
##
## Tree size: 336
##
## Non-standard options: attempt to group attributes
```

```
#making predict on testSet using our model
dt_prediction <- predict(dt_model, testSet, type = "class")
```

[8]- DecisionTree Confusion Matrix

```
confusionMatrix(dt_prediction, testSet$y)
```

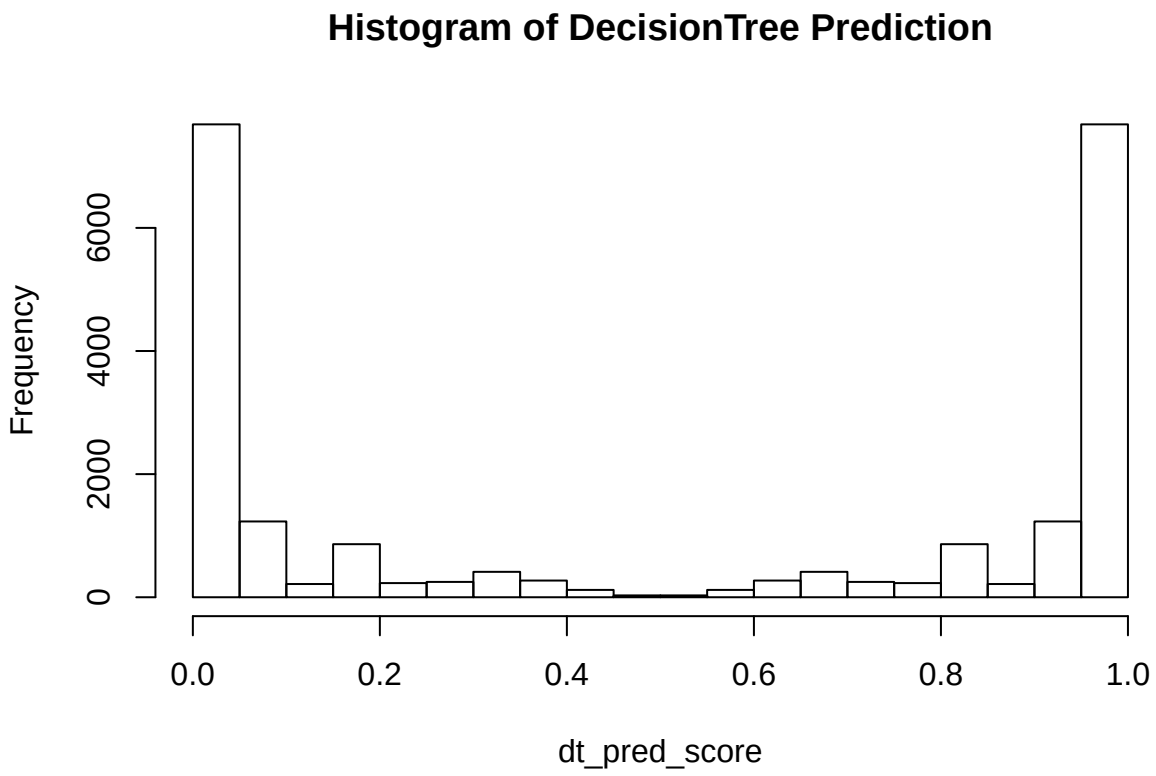
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    1    0
##          1  615  379
##          0  707 9601
##
##                Accuracy : 0.9039
##                  95% CI : (0.8983, 0.9093)
##     No Information Rate : 0.883
##     P-Value [Acc > NIR] : 7.12e-13
##
##                   Kappa : 0.4788
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.46520
##             Specificity : 0.96202
##          Pos Pred Value : 0.61871
##          Neg Pred Value : 0.93141
##              Prevalence : 0.11697
##          Detection Rate : 0.05442
##    Detection Prevalence : 0.08795
##       Balanced Accuracy : 0.71361
##
##        'Positive' Class : 1
##
```

```
#view some metrics about our decsionTree prediction like Accuracy and TPR compared
#to the actual value of 'y' variable
mmetric(testSet$y, dt_prediction, c("ACC", "PRECISION", "TPR", "F1"))
```

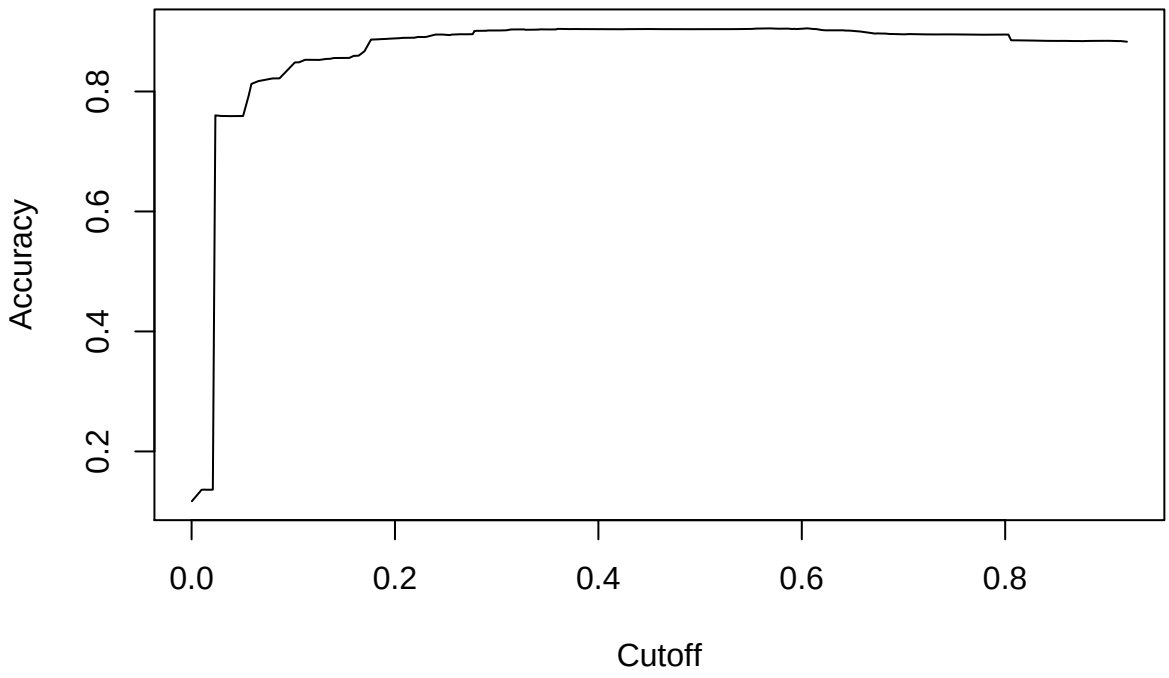
```
##        ACC PRECISION1 PRECISION2       TPR1       TPR2        F11
##   90.39108   61.87123   93.14125   46.52042   96.20240   53.10881
##        F12
##   94.64708
```

[9]- DecisionTree Model Evaluation

```
library(ROCR)
dt_pred_score = predict(dt_model, newdata = testSet, type = 'prob')
#plotting the histogram of decisionTree prediction
hist(dt_pred_score, main = "Histogram of DecisionTree Prediction")
```

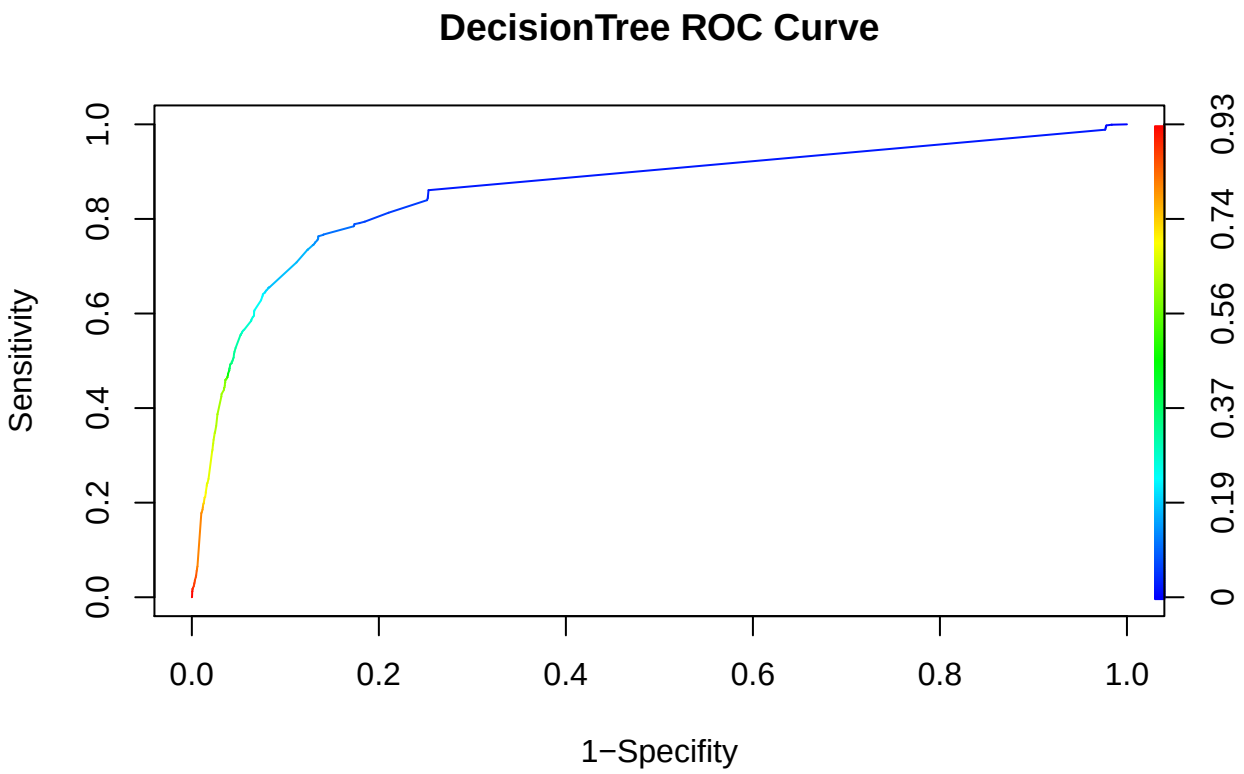



```
dt_pred = prediction(dt_pred_score[,1], testSet$y)
dt_eval = performance(dt_pred, "acc")
#plotting the performance of decisionTree prediction
plot(dt_eval)
```



[10]- DecisionTree ROC Curve

```
dt_roc = performance(dt_pred, "tpr", "fpr")
plot(dt_roc, colorize=T, main="DecisionTree ROC Curve", ylab="Sensitivity", xlab="1-Specifity")
```


DecisionTree ROC Curve
Sensitivity
1−Specifity
0.0
0.2
0.4
0.6
0.8
1.0
0
0.19
0.37
0.56
0.74
0.93