Detecção de Fraudes no Tráfego de Cliques em Propagandas de Aplicações Mobile

Gilson Santana

02/11/2020

Informações gerais

Projeto da DSA como parte do treiamento Big Data Analytics com R e Microsoft Azure Machine Learning.

O projeto consiste na criação de modelo de Machine Learning que possa prever se um click para download de aplicativo é ou não fraudulento. Para esse trabalho foi utilizado a base de dados train_sample.csv, disponível no link https://www.kaggle.com/c/talkingdata-adtracking-fraud-detection/data. Maiores detalhe do desafio também pode ser obtido no site do Kaggle.

Conhecendo os dados

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ROSE)
```

```
## Loaded ROSE 0.0-3
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
#Diretório de trabalho
setwd('D:/OneDrive/EstudosTecnicos/CienciaDados/DtScienceTrab/BigDataRAzure/Cap20/Projeto01')

# Carregar dados - Utilizada a base train-sample.csv devido ao tamanho da base train
dfDados<- read.csv('train_sample.csv', stringsAsFactors= F, header =T)

str(dfDados)
```

```
## 'data.frame':    100000 obs. of  8 variables:
##  $ ip             : int  87540 105560 101424 94584 68413 93663 17059 121505 192967 143636 ...
##  $ app            : int  12 25 12 13 12 3 1 9 2 3 ...
##  $ device         : int  1 1 1 1 1 1 1 1 2 1 ...
##  $ os             : int  13 17 19 13 1 17 17 25 22 19 ...
##  $ channel        : int  497 259 212 477 178 115 135 442 364 135 ...
##  $ click_time     : chr  "2017-11-07 09:30:38" "2017-11-07 13:40:27" "2017-11-07 18:05:24" "2017-11-0
##  $ attributed_time: chr  "" "" "" "" ...
##  $ is_attributed  : int  0 0 0 0 0 0 0 0 0 0 ...
```

```
# View(dfDados)
summary(dfDados)
```

```
##        ip             app              device             os        
##  Min.   :     9   Min.   :  1.00   Min.   :   0.00   Min.   :  0.00  
##  1st Qu.: 40552   1st Qu.:  3.00   1st Qu.:   1.00   1st Qu.: 13.00  
##  Median : 79827   Median : 12.00   Median :   1.00   Median : 18.00  
##  Mean   : 91256   Mean   : 12.05   Mean   :  21.77   Mean   : 22.82  
##  3rd Qu.:118252   3rd Qu.: 15.00   3rd Qu.:   1.00   3rd Qu.: 19.00  
##  Max.   :364757   Max.   :551.00   Max.   :3867.00   Max.   :866.00  
##     channel       click_time        attributed_time    is_attributed    
##  Min.   :  3.0   Length:100000      Length:100000      Min.   :0.00000  
##  1st Qu.:145.0   Class :character   Class :character   1st Qu.:0.00000  
##  Median :258.0   Mode  :character   Mode  :character   Median :0.00000  
##  Mean   :268.8                                         Mean   :0.00227  
##  3rd Qu.:379.0                                         3rd Qu.:0.00000  
##  Max.   :498.0                                         Max.   :1.00000
```

```
# Tratar valores NA - Não tem valores missing
any(is.na(dfDados))
```

```
## [1] FALSE
```

Análise de atributos

```
# Relação entre attributed_time e is_attributed. Se baixado implica na existendia do atributed_time, as
# essa variável consequência da variável target, não pode figurar como perditora
    nrow(dfDados %>% filter(attributed_time == ''))
```

```
## [1] 99773
```

```
    nrow(dfDados %>% filter(is_attributed == '0'))
```

```
## [1] 99773
```

```
    nrow(dfDados %>% filter(attributed_time == '' & is_attributed == '0'))
```

```
## [1] 99773
```

```
    dfDados$attributed_time <- NULL


# click_time  - caberia uma análise de série temporal
    dfDados$dt <- date(dfDados$click_time)
    dfDados$day <- day(dfDados$dt)
    dfDados$month <- month(dfDados$dt)
    dfDados$weekday <- wday(dfDados$dt)
```

```
    dfDados$hour <- hour(dfDados$click_time)
    str(dfDados)
```

```
## 'data.frame':    100000 obs. of  12 variables:
##  $ ip           : int  87540 105560 101424 94584 68413 93663 17059 121505 192967 143636 ...
##  $ app          : int  12 25 12 13 12 3 1 9 2 3 ...
##  $ device       : int  1 1 1 1 1 1 1 1 2 1 ...
##  $ os           : int  13 17 19 13 1 17 17 25 22 19 ...
##  $ channel      : int  497 259 212 477 178 115 135 442 364 135 ...
##  $ click_time   : chr  "2017-11-07 09:30:38" "2017-11-07 13:40:27" "2017-11-07 18:05:24" "2017-11-07
##  $ is_attributed: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ dt           : Date, format: "2017-11-07" "2017-11-07" ...
##  $ day          : int  7 7 7 7 9 9 9 7 8 8 ...
##  $ month        : num  11 11 11 11 11 11 11 11 11 11 ...
##  $ weekday      : num  3 3 3 3 5 5 5 3 4 4 ...
##  $ hour         : int  9 13 18 4 9 1 1 10 9 12 ...
```

Correlação

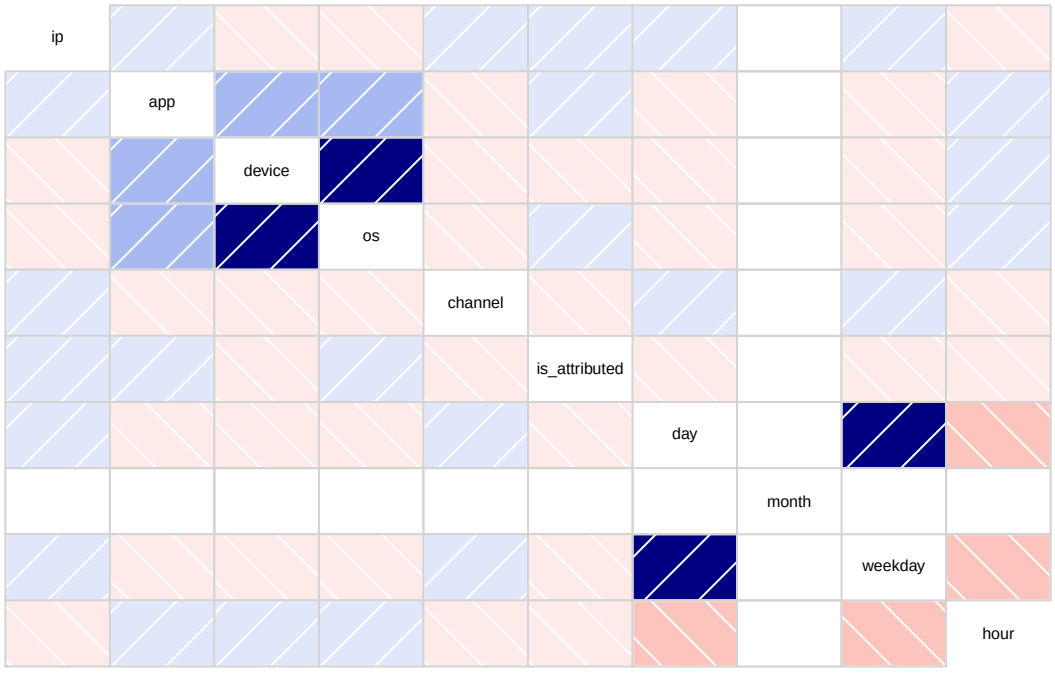
```
# Análise de correlação
    library(corrgram)
```

```
## Registered S3 method overwritten by 'seriation':
##   method         from
##   reorder.hclust gclus
```

```
    corrgram(dfDados)
```

```
## Warning in cor(x, use = "pairwise.complete.obs", method = cor.method): o desvio
## padrão é zero

## Warning in cor(x, y, use = "pair", method = cor.method): o desvio padrão é zero

## Warning in cor(x, y, use = "pair", method = cor.method): o desvio padrão é zero

## Warning in cor(x, y, use = "pair", method = cor.method): o desvio padrão é zero

## Warning in cor(x, y, use = "pair", method = cor.method): o desvio padrão é zero

## Warning in cor(x, y, use = "pair", method = cor.method): o desvio padrão é zero

## Warning in cor(x, y, use = "pair", method = cor.method): o desvio padrão é zero

## Warning in cor(x, y, use = "pair", method = cor.method): o desvio padrão é zero

## Warning in cor(x, y, use = "pair", method = cor.method): o desvio padrão é zero

## Warning in cor(x, y, use = "pair", method = cor.method): o desvio padrão é zero

## Warning in cor(x, y, use = "pair", method = cor.method): o desvio padrão é zero

## Warning in cor(x, y, use = "pair", method = cor.method): o desvio padrão é zero

## Warning in cor(x, y, use = "pair", method = cor.method): o desvio padrão é zero
```

```
## Warning in cor(x, y, use = "pair", method = cor.method): o desvio padrão é zero

## Warning in cor(x, y, use = "pair", method = cor.method): o desvio padrão é zero

## Warning in cor(x, y, use = "pair", method = cor.method): o desvio padrão é zero

## Warning in cor(x, y, use = "pair", method = cor.method): o desvio padrão é zero

## Warning in cor(x, y, use = "pair", method = cor.method): o desvio padrão é zero

## Warning in cor(x, y, use = "pair", method = cor.method): o desvio padrão é zero
```




```
    dfDados %>% select(month) %>% distinct(month) #somente mês 11, é como uma constante
```

```
##   month
## 1    11
```

```
    dfDados$month <- NULL


# Transformar variáveis em categóricas
    toFactor<- function(df, var) {
        for(v in var) df[,v]= factor(df[,v])
        return(df)
    }
    VarToFactor<- c('app','device','os','channel','is_attributed', 'ip','day', 'weekday', 'hour')
    dfDados<- toFactor(dfDados, VarToFactor)
```

```r
# Divisão dos dados
linhas <- sample(1:nrow(dfDados), 0.7 * nrow(dfDados))
dfTrain <- dfDados[linhas,]
dfTest <- dfDados[-linhas,]
```

Balanceamento de classe

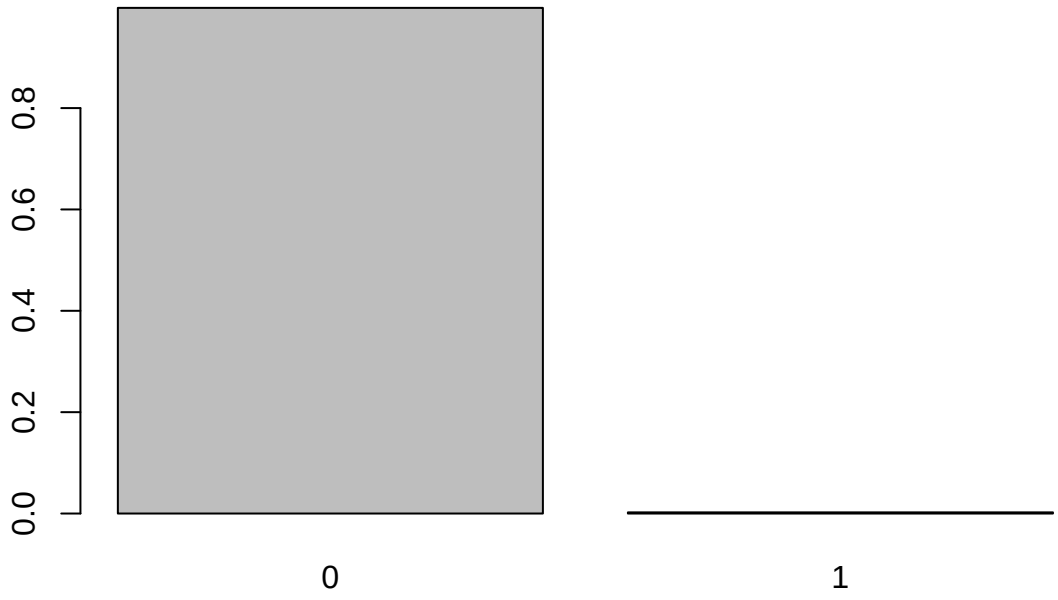
```r
summary(dfTrain$is_attributed)
```

```
##     0     1
## 69842   158
```

```r
prop.table(table(dfTrain$is_attributed))
```

```
##
##           0           1
## 0.997742857 0.002257143
```

```r
barplot(prop.table(table(dfTrain$is_attributed)))
```




```r
dfTrainBal <- ROSE(is_attributed ~ ip +
                     app +
                     device +
                     os +
                     channel +
                     day +
```

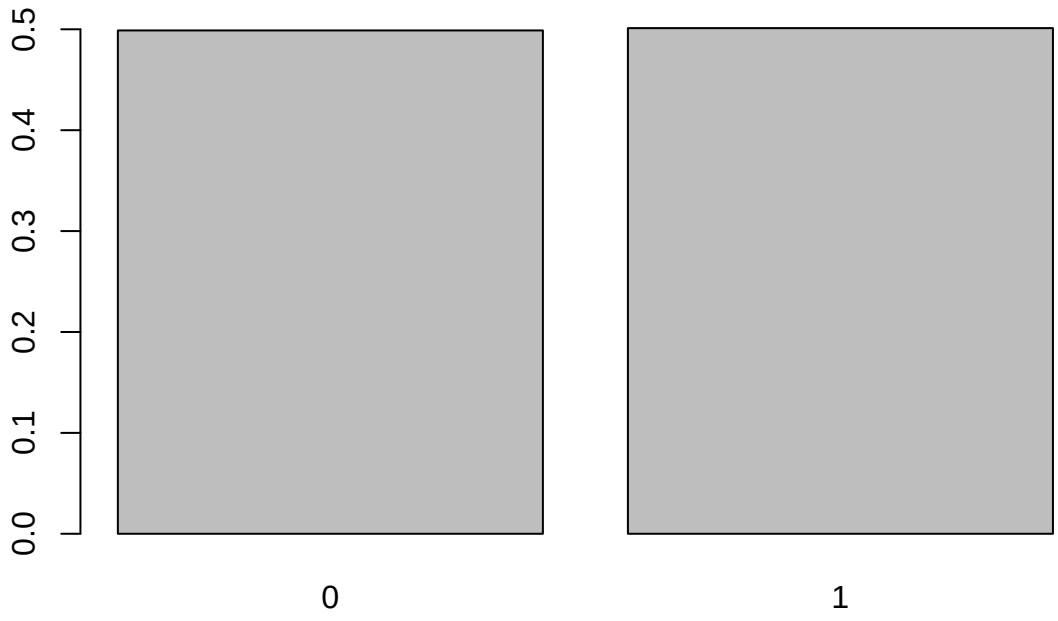
```
                        weekday +
                        hour,
                    data = dfTrain, seed = 1)$data

    # Nova Proporção
    summary(dfTrainBal$is_attributed)
```

```
##     0     1
## 34919 35081
```

```
    prop.table(table(dfTrainBal$is_attributed))
```

```
##
##         0         1
## 0.4988429 0.5011571
```

```
    barplot(prop.table(table(dfTrainBal$is_attributed)))
```




```
    any(is.na(dfTrainBal))
```

```
## [1] FALSE
```

```
    # Balancear Teste
    dfTestBal <- ROSE(is_attributed ~ ip +
                          app +
                          device +
                          os +
                          channel +
```

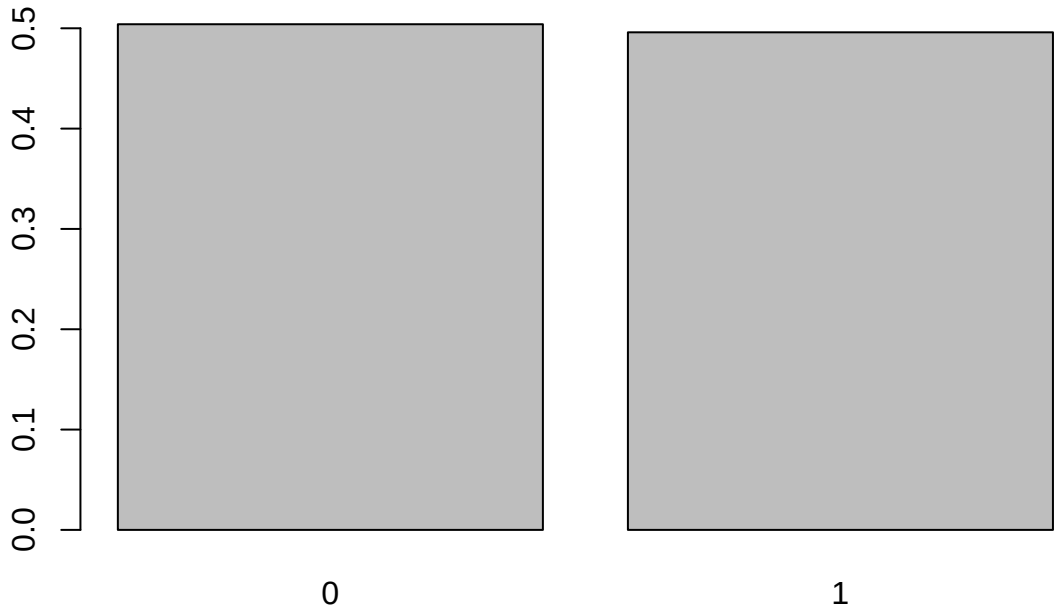
```
                            day +
                            weekday +
                            hour,
                        data = dfTest, seed = 1)$data
    summary(dfTestBal$is_attributed)
```

```
##     0     1
## 15121 14879
```

```
    prop.table(table(dfTestBal$is_attributed))
```

```
##
##         0         1
## 0.5040333 0.4959667
```

```
    barplot(prop.table(table(dfTestBal$is_attributed)))
```




```
    any(is.na(dfTestBal))
```

```
## [1] FALSE
```

Treinando modelos

OBS: O Radom Forest não aceita trabalhar com variáveis categóricas com mais de 53 níveis. Por esse motivo, algumas Variáveis foram ajustadas para o tipo character.

```
# Treinando modelos
    library(C50) #algorítimo C5.0
```

```
    library(e1071) #naiveBayes
    library(randomForest)
```

```
## randomForest 4.6-14
```

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

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

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
    #Árvore de decisão - algorítimo C5.0
    m.Arvore1 <- C5.0(is_attributed ~ ., data = dfTrainBal, rules = TRUE)

    # naiveBayes
    m.Naive1<- naiveBayes(is_attributed ~ ., data = dfTrainBal, laplace = 0)

    # Radom Forest - Não aceita trabalhar com factor com mais de 53 níveis. Variáveis
    # ajustadas para o tipo character.
    dfTrainBalRandom<- dfTrainBal
    dfTrainBalRandom$ip<- as.character(dfTrainBalRandom$ip)
    dfTrainBalRandom$app<- as.character(dfTrainBalRandom$app)
    dfTrainBalRandom$device<- as.character(dfTrainBalRandom$device)
    dfTrainBalRandom$os<- as.character(dfTrainBalRandom$os)
    dfTrainBalRandom$channel<- as.character(dfTrainBalRandom$channel)
    str(dfTrainBalRandom)
```

```
## 'data.frame':    70000 obs. of  9 variables:
##  $ ip           : chr  "25818" "17149" "107653" "116326" ...
##  $ app          : chr  "2" "9" "12" "2" ...
##  $ device       : chr  "1" "1" "1" "1" ...
##  $ os           : chr  "6" "37" "18" "13" ...
##  $ channel      : chr  "205" "466" "265" "469" ...
##  $ is_attributed: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ day          : Factor w/ 4 levels "6","7","8","9": 3 2 3 3 2 1 4 4 3 4 ...
##  $ weekday      : Factor w/ 4 levels "2","3","4","5": 3 2 3 3 2 1 4 4 3 4 ...
##  $ hour         : Factor w/ 24 levels "0","1","2","3",..: 10 16 13 10 16 18 2 4 8 7 ...
```

```
    m.Random1 <- randomForest( is_attributed ~ ip +
                                   app +
                                   device +
                                   os +
                                   channel +
                                   day +
                                   weekday +
                                   hour,
                               data = dfTrainBalRandom,
                               ntree = 100, nodesize = 10)
```

Predição e avaliação

```
# Predições
     p.Arvore1<- predict(m.Arvore1, dfTestBal)
     p.Naive1<- predict(m.Naive1, dfTestBal)

     dfTestBalRandom<- dfTestBal
     dfTestBalRandom$ip<- as.character(dfTestBalRandom$ip)
     dfTestBalRandom$app<- as.character(dfTestBalRandom$app)
     dfTestBalRandom$device<- as.character(dfTestBalRandom$device)
     dfTestBalRandom$os<- as.character(dfTestBalRandom$os)
     dfTestBalRandom$channel<- as.character(dfTestBalRandom$channel)
     p.Random1<- predict(m.Random1, dfTestBalRandom)

#Avaliando predições
     library(caret)
```

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

```
##
## Attaching package: 'lattice'
```

```
## The following object is masked from 'package:corrgram':
##
##     panel.fill
```

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

```
##
## Attaching package: 'ggplot2'
```

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

```
     confusionMatrix(dfTestBal$is_attributed, p.Arvore1)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0 14234   887
##          1  4497 10382
##
##                Accuracy : 0.8205
##                  95% CI : (0.8161, 0.8249)
##     No Information Rate : 0.6244
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6403
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.7599
##             Specificity : 0.9213
##          Pos Pred Value : 0.9413
##          Neg Pred Value : 0.6978
```

```
##              Prevalence : 0.6244
##          Detection Rate : 0.4745
##    Detection Prevalence : 0.5040
##       Balanced Accuracy : 0.8406
##
##        'Positive' Class : 0
##
```

```
confusionMatrix(dfTestBal$is_attributed, p.Naive1)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0 14116  1005
##          1  2009 12870
##
##                Accuracy : 0.8995
##                  95% CI : (0.8961, 0.9029)
##     No Information Rate : 0.5375
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7989
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.8754
##             Specificity : 0.9276
##          Pos Pred Value : 0.9335
##          Neg Pred Value : 0.8650
##              Prevalence : 0.5375
##          Detection Rate : 0.4705
##    Detection Prevalence : 0.5040
##       Balanced Accuracy : 0.9015
##
##        'Positive' Class : 0
##
```

```
confusionMatrix(dfTestBal$is_attributed, p.Random1)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0 15105    16
##          1 14443   436
##
##                Accuracy : 0.518
##                  95% CI : (0.5124, 0.5237)
##     No Information Rate : 0.9849
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.0285
##
##  Mcnemar's Test P-Value : <2e-16
```

```
##
##             Sensitivity : 0.5112
##             Specificity : 0.9646
##          Pos Pred Value : 0.9989
##          Neg Pred Value : 0.0293
##              Prevalence : 0.9849
##          Detection Rate : 0.5035
##    Detection Prevalence : 0.5040
##       Balanced Accuracy : 0.7379
##
##        'Positive' Class : 0
##
```

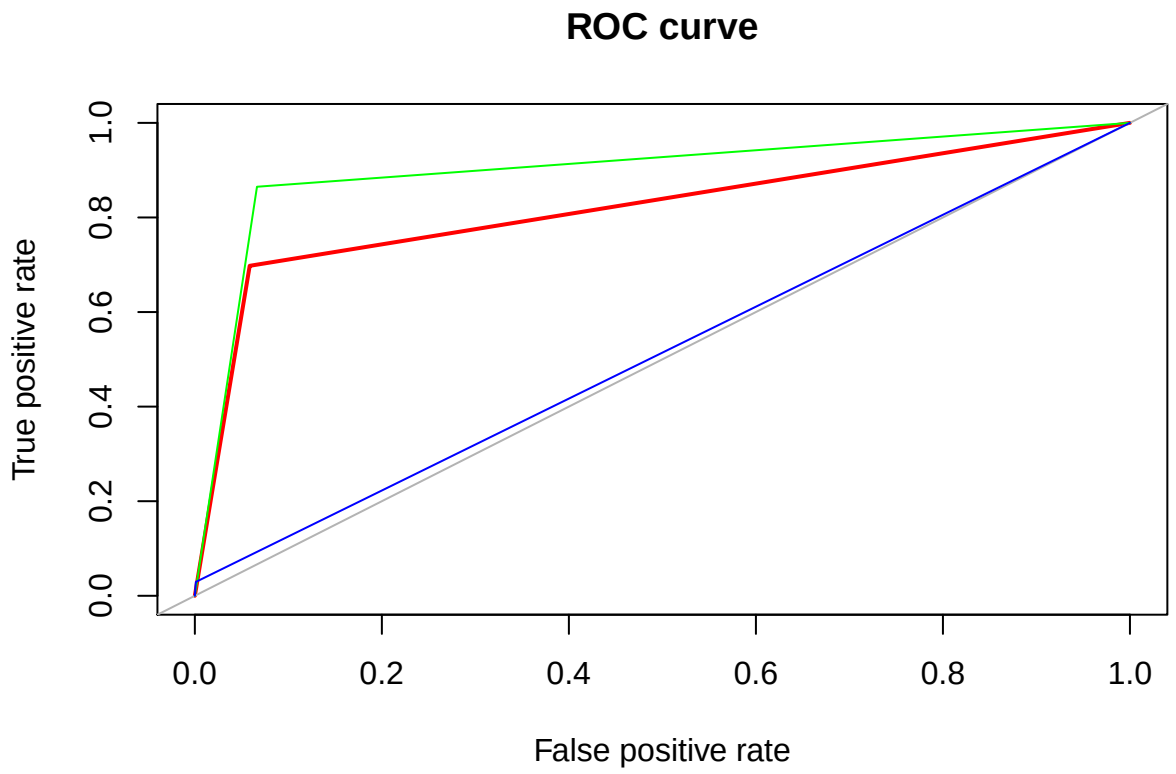
```
# ROC Curves com o ROSE
    roc.curve(dfTestBal$is_attributed, p.Arvore1, plotit = T, col = "red", add.roc = F)
```

```
## Area under the curve (AUC): 0.820
```

```
    roc.curve(dfTestBal$is_attributed, p.Naive1, plotit = T, col = "green", add.roc = T)
```

```
## Area under the curve (AUC): 0.899
```

```
    roc.curve(dfTestBal$is_attributed, p.Random1, plotit = T, col = "blue", add.roc = T)
```




```
## Area under the curve (AUC): 0.514
```

Conclusões

O moldelo baseado no Naive apresentou melhor acurácia, poderia seguir com um refinamento do processo de otimização

O C5.0 vem em seguida. Apresenta também como candidado a seguir com oma otimização

O modelo de Radom Forest não conseguiu rodar com as variáveis categóricas (factor) com mais de 53 níveis. Assim, algumas variáveis foram convertidas para caracter. isso pode ter influenciando no seu baixo desempenho.