---

title: “Ad Fraud Machine Learning Classification challenge” author: “Carlos A Costa” date: “06/02/2020” output: pdf_document

Problema de negócio: Detectar fraudes no Tráfego de Cliques em Propagandas de Aplicações Mobile

Descrição: Projeto realizado com o objetivo de criar um modelo de machine learning para determinar a possibilidade de um usuário realizar o download de um aplicativo infectado, após o click em um anúncio fraudulento.

Dataset original disponível em: https://www.kaggle.com/c/talkingdata-adtracking-fraud-detection/data

```
library(caret)
```

```
## Loading required package: lattice

## Loading required package: ggplot2
```

```
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 3.6.2

## corrplot 0.84 loaded
```

```
library(gmodels)
```

```
## Warning: package 'gmodels' was built under R version 3.6.2
```

```
library(ROSE)
```

```
## Warning: package 'ROSE' was built under R version 3.6.2

## Loaded ROSE 0.0-3
```

```
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:ggplot2':
##
##     margin
```

```
library(DMwR)
```

```
## Warning: package 'DMwR' was built under R version 3.6.2

## Loading required package: grid

## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
library(rpart)
library(ROCR)
```

```
## Warning: package 'ROCR' was built under R version 3.6.2
```

```
## Loading required package: gplots
```

```
## Warning: package 'gplots' was built under R version 3.6.2
```

```
##
## Attaching package: 'gplots'
```

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

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.6.2
```

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

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

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

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

```
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 3.6.2
```

Etapa 1 - Coletando os Dados

Leitura do arquivo original(200 milhões de observações) reduzido para 30000 observações apenas, pois o hardaware do notebook tem apenas 4 GB RAM.

```
sample <- read.csv(file="train.csv",header=TRUE,sep=",",nrows = 30000,stringsAsFactors = FALSE)
```

Arquivo com 30000 observações

```
write.csv(sample,"treino1.csv")

treino1 <- read.csv(file="treino1.csv",header=TRUE, sep=",")
```

```
View(head(treino1))
str(treino1)
```

```
## 'data.frame':    30000 obs. of  9 variables:
##  $ X              : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ ip             : int  83230 17357 35810 45745 161007 18787 103022 114221 165970 74544 ...
##  $ app            : int  3 3 3 14 3 3 3 3 3 64 ...
##  $ device         : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ os             : int  13 19 13 13 13 16 23 19 13 22 ...
##  $ channel        : int  379 379 379 478 379 379 379 379 379 459 ...
##  $ click_time     : Factor w/ 456 levels "2017-11-06 14:32:21",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ attributed_time: Factor w/ 55 levels "","2017-11-06 16:00:47",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ is_attributed  : int  0 0 0 0 0 0 0 0 0 0 ...
```

```r
summary(treino1)
```

```
##        X               ip              app              device
##  Min.   :    1   Min.   :    92   Min.   :  1.00   Min.   :   0.0
##  1st Qu.: 7501   1st Qu.: 43449   1st Qu.:  3.00   1st Qu.:   1.0
##  Median :15000   Median : 83494   Median : 10.00   Median :   1.0
##  Mean   :15000   Mean   : 87872   Mean   : 12.37   Mean   :  31.8
##  3rd Qu.:22500   3rd Qu.:121176   3rd Qu.: 15.00   3rd Qu.:   1.0
##  Max.   :30000   Max.   :212743   Max.   :538.00   Max.   :3032.0
##
##        os            channel                     click_time
##  Min.   :  0.00   Min.   :  3.0   2017-11-06 16:00:34: 1056
##  1st Qu.: 13.00   1st Qu.:137.0   2017-11-06 16:00:35: 1033
##  Median : 18.00   Median :215.0   2017-11-06 16:00:33: 1011
##  Mean   : 26.94   Mean   :245.8   2017-11-06 16:00:25:  909
##  3rd Qu.: 19.00   3rd Qu.:347.0   2017-11-06 16:00:18:  893
##  Max.   :607.00   Max.   :498.0   2017-11-06 16:00:28:  887
##                                   (Other)            :24211
##            attributed_time  is_attributed
##                    :29946   Min.   :0.0000
##  2017-11-06 16:00:47:    1   1st Qu.:0.0000
##  2017-11-06 16:01:03:    1   Median :0.0000
##  2017-11-06 16:01:05:    1   Mean   :0.0018
##  2017-11-06 16:01:18:    1   3rd Qu.:0.0000
##  2017-11-06 16:01:22:    1   Max.   :1.0000
##  (Other)            :   49
```

```r
any(is.na(treino1))
```

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

2 ETAPA - Explorando relacionamento entre as variáveis: Matriz de Correlação

Pré-Processamento

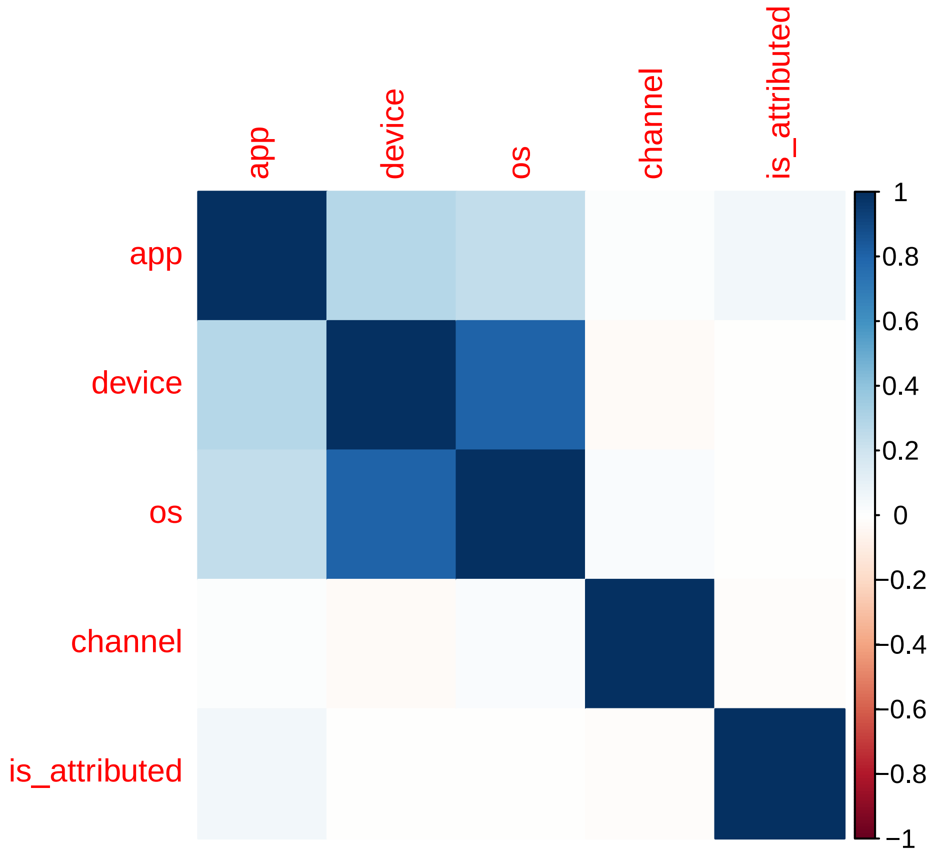
As colunas "click_time", "attributed_time", "X" e "ip" foram desconsideradas para a modelo

```r
treino1$click_time <-NULL
treino1$attributed_time <-NULL
treino1$X <-NULL
treino1$ip <-NULL
```

```r
cor_data <-cor(treino1)
```

Análise: As variáveis "os" e "device" possuem forte correlação; "device" e "app" também, mas em menor grau.

```r
corrplot(cor_data, method = 'color')
```

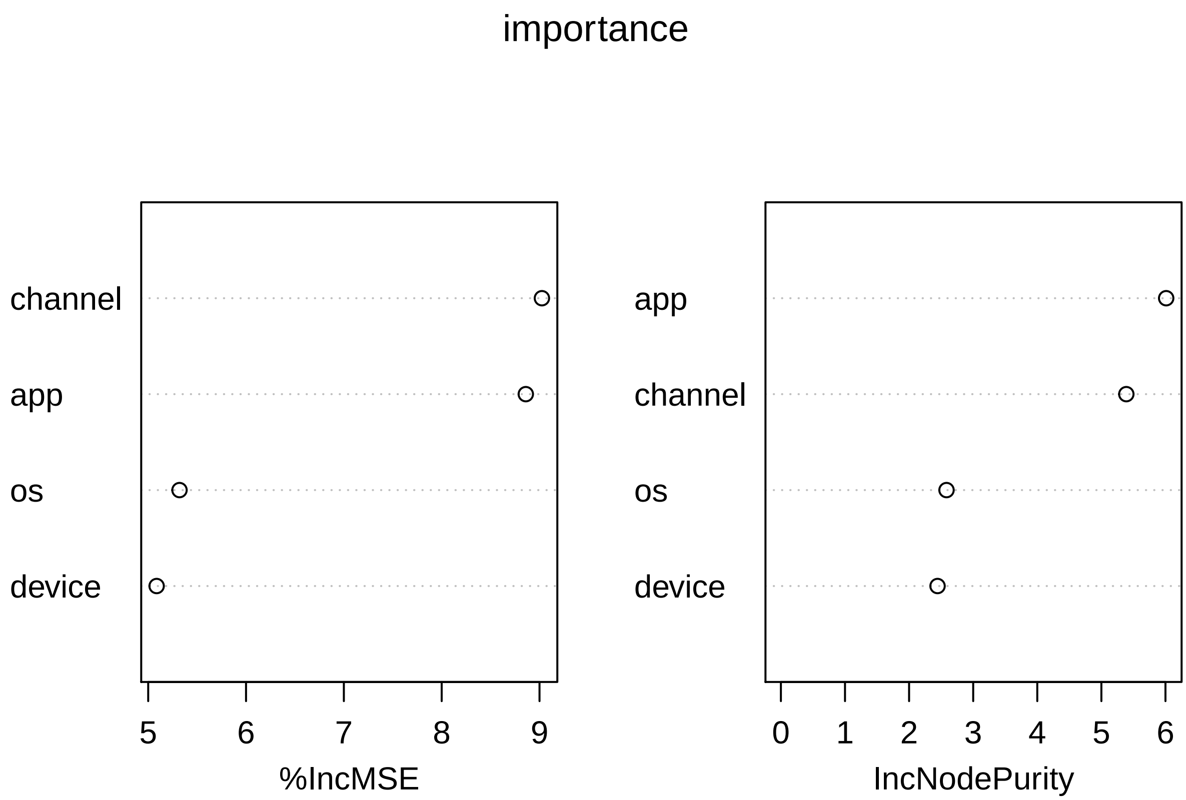


Modelo randomForest para criar um plot de importância das variáveis

```
importance <- randomForest(is_attributed ~.,
                           data = treino1,
                           ntree = 100, nodesize = 10, importance = T)
```

```
## Warning in randomForest.default(m, y, ...): The response has five or fewer
## unique values. Are you sure you want to do regression?
```

```
varImpPlot(importance)
```



Convertendo para factor

```
treino1$is_attributed <- as.factor(treino1$is_attributed)
```

Divisao dos dados (Data Split)

```
split1 <- createDataPartition(y = treino1$is_attributed, p = 0.7, list = FALSE)
```

Criando dados de treino e de teste

```
dados_treino <- treino1[split1,]
dados_teste <- treino1[-split1,]
```

Verificando distribuição da variável target, observa-se que a variável target possui 99% dos dados classificados como "0"(não realizou download) e 1% como "1"(realizou download). Por conseguinte, é necessário realizar técnicas para balancear a variável a fim de evitar o OVERFITTING(Sobreajuste) do modelo preditivo.

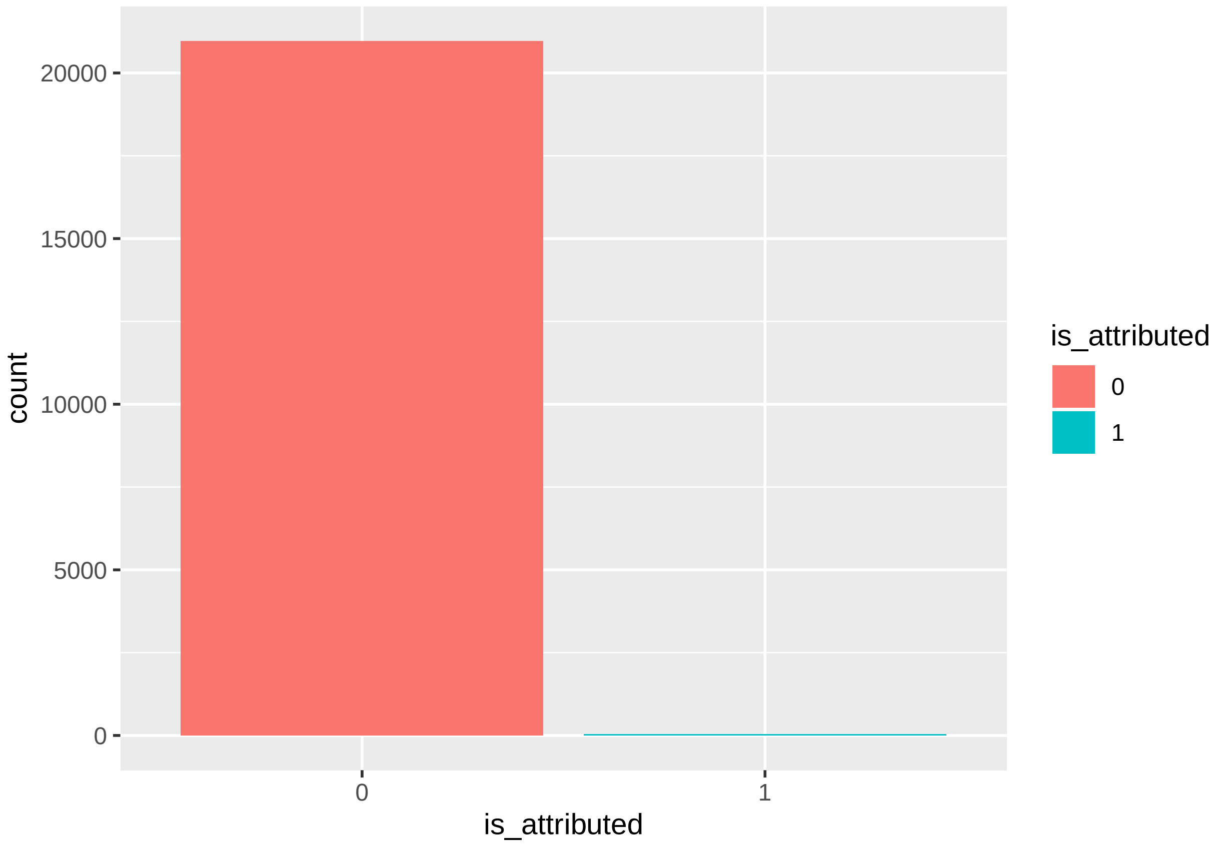
```
table(dados_treino$is_attributed)
```

```
##
##     0     1
## 20963    38
```

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

```
##
##           0           1
## 0.998190562 0.001809438
```

```r
ggplot(dados_treino,aes(x=is_attributed, fill=is_attributed)) +
  geom_bar()
```



Realizando diferentes técnicas de balanceamento para a variável preditora

Over sampling

```r
data_balanced_over <- ovun.sample(is_attributed ~ ., data = dados_treino, method = "over",N = 41926)$da
table(data_balanced_over$is_attributed)
```

```
##
##     0     1
## 20963 20963
```

Método ROSE

```r
data.rose <- ROSE(is_attributed ~ ., data = dados_treino,hmult.majo=0.25, hmult.mino=0.5)$data
```

Etapa 4: Treinando o modelo com diferentes algorítimos de machine learning

Algorítimo Decision Tree

```r
tree.rose <- rpart(is_attributed ~ ., data = data.rose)
tree.over <- rpart(is_attributed ~ ., data = data_balanced_over)
```

ETAPA 5: Validando os modelos de machine learning

Método Rose

```r
pred.tree.rose <- predict(tree.rose, newdata = dados_teste, type='class')
```

Método Over sampling

```
pred.tree.over <- predict(tree.over, newdata = dados_teste,type='class')
```

Método SMOTE

```
ctrl <- trainControl(verboseIter = FALSE,
                     sampling = "smote")


model_rf_smote <- caret::train(is_attributed ~ .,
                               data = dados_treino,
                               method = "rf",
                               preProcess = c("scale", "center"),
                               trControl = ctrl)

final_smote <- predict(model_rf_smote, newdata = dados_teste,type='raw')
```

Modelo Support Vector Machines sob os dados balanceados (ROSE)

```
modelo_svm_v1 <- svm(is_attributed ~ .,
                     data = data.rose,
                     type = 'C-classification',
                     kernel = 'radial')

pred.svm.rose <-predict(modelo_svm_v1,dados_teste,type='raw')
```

#Modelo Logistic Regression

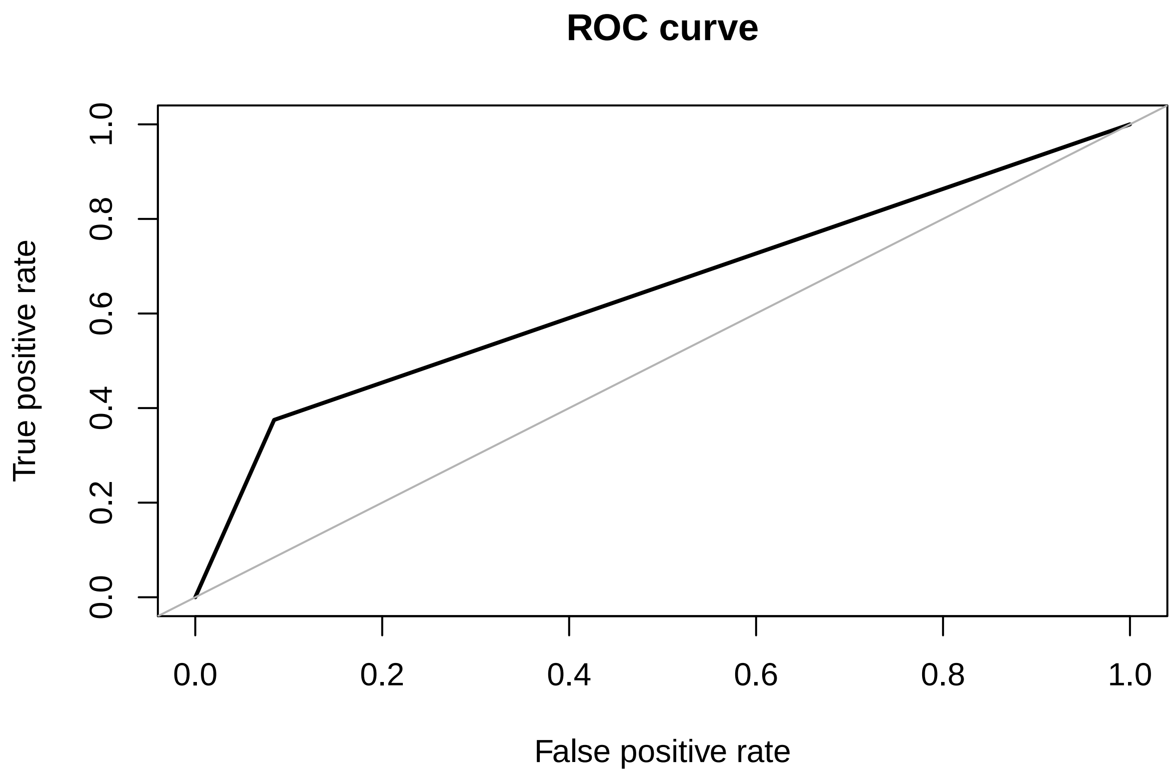
```
glm <- glm(is_attributed ~.,data.rose, family=binomial(link='logit'))

glm.pred <- predict(glm,dados_teste,type='response')
glm.pred <- ifelse(glm.pred >0.5,1,0)
glm.pred <- as.factor(glm.pred)
```

#Avaliação de desempenho dos modelos

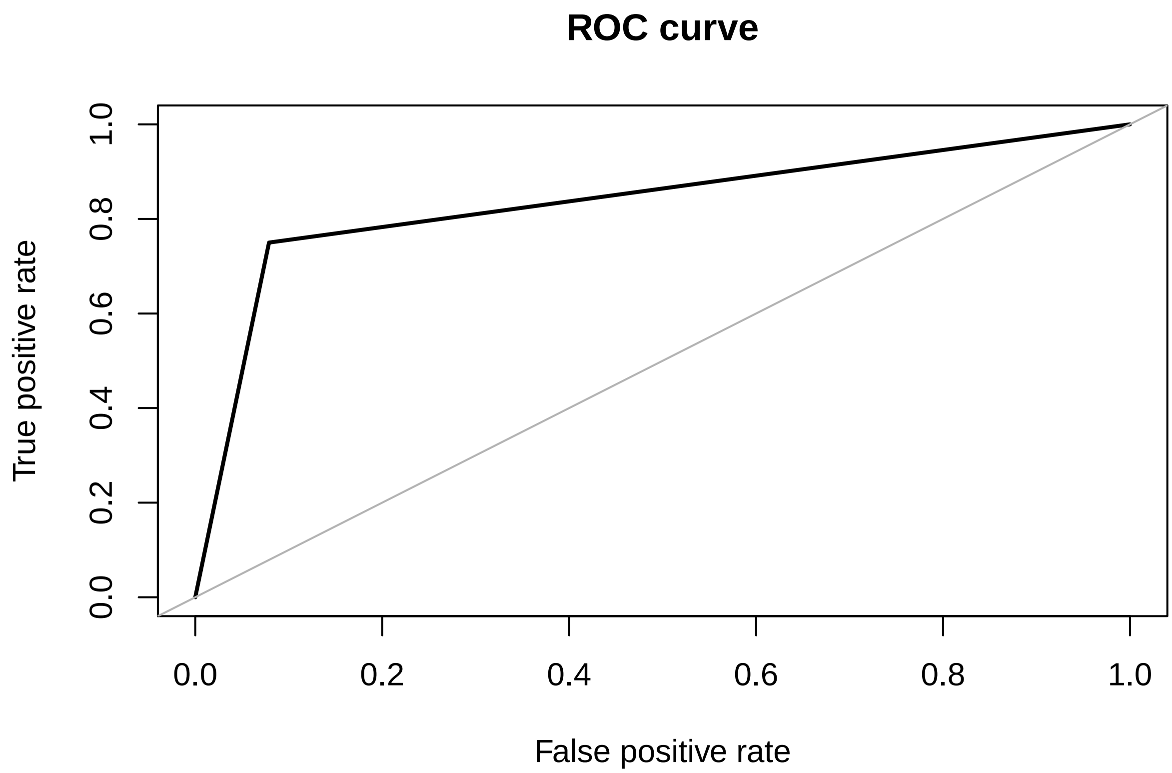
#Modelo : Classification Decision Tree

```
roc.curve(dados_teste$is_attributed,pred.tree.rose)
```




```
## Area under the curve (AUC): 0.645
```

```r
roc.curve(dados_teste$is_attributed,pred.tree.over)
```




```
## Area under the curve (AUC): 0.836
```

```r
confusionMatrix(dados_teste$is_attributed,pred.tree.rose)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 8225  758
##          1   10    6
##
##                Accuracy : 0.9147
##                  95% CI : (0.9087, 0.9204)
##     No Information Rate : 0.9151
##     P-Value [Acc > NIR] : 0.5696
##
##                   Kappa : 0.0119
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.998786
##             Specificity : 0.007853
##          Pos Pred Value : 0.915618
##          Neg Pred Value : 0.375000
##              Prevalence : 0.915102
##          Detection Rate : 0.913990
##    Detection Prevalence : 0.998222
```

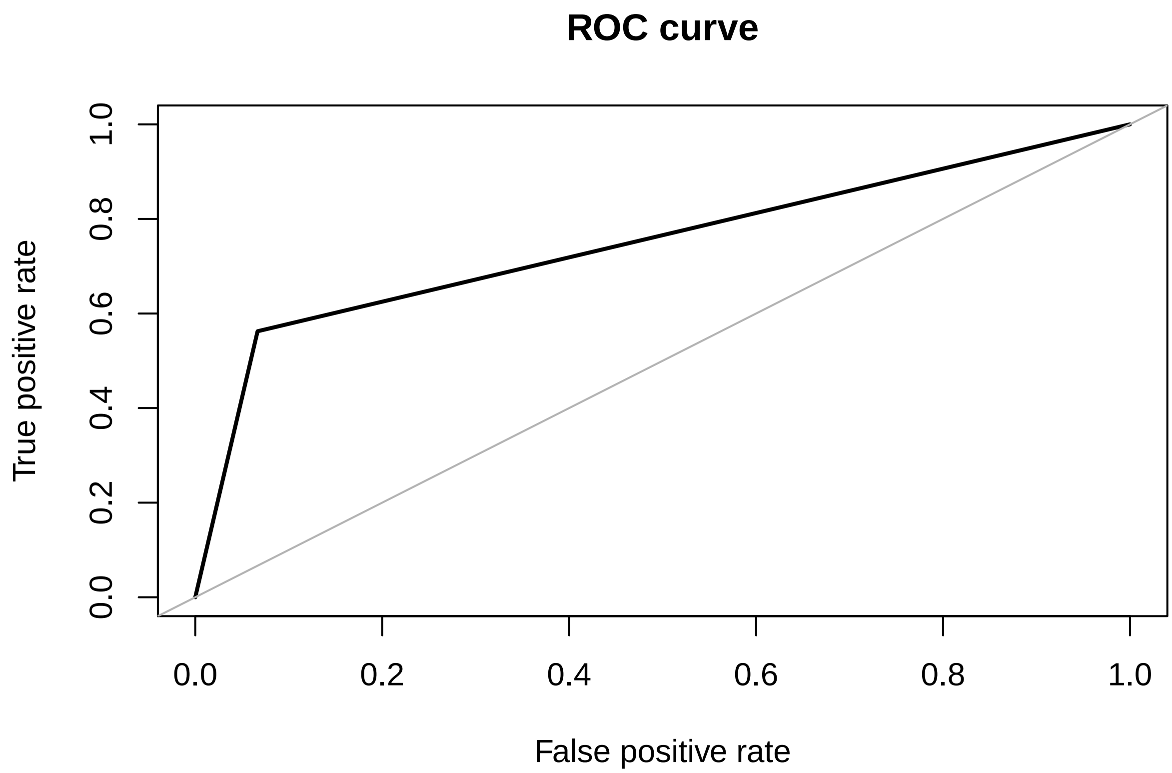
```
##       Balanced Accuracy : 0.503320
##
##        'Positive' Class : 0
##
```

```r
confusionMatrix(dados_teste$is_attributed,pred.tree.over)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 8274  709
##          1    4   12
##
##                Accuracy : 0.9208
##                  95% CI : (0.915, 0.9263)
##     No Information Rate : 0.9199
##     P-Value [Acc > NIR] : 0.3873
##
##                   Kappa : 0.0292
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.99952
##             Specificity : 0.01664
##          Pos Pred Value : 0.92107
##          Neg Pred Value : 0.75000
##              Prevalence : 0.91988
##          Detection Rate : 0.91944
##    Detection Prevalence : 0.99822
##       Balanced Accuracy : 0.50808
##
##        'Positive' Class : 0
##
```

#Modelo : Support Vector Machine

```r
roc.curve(dados_teste$is_attributed,pred.svm.rose)
```




```
## Area under the curve (AUC): 0.748
```

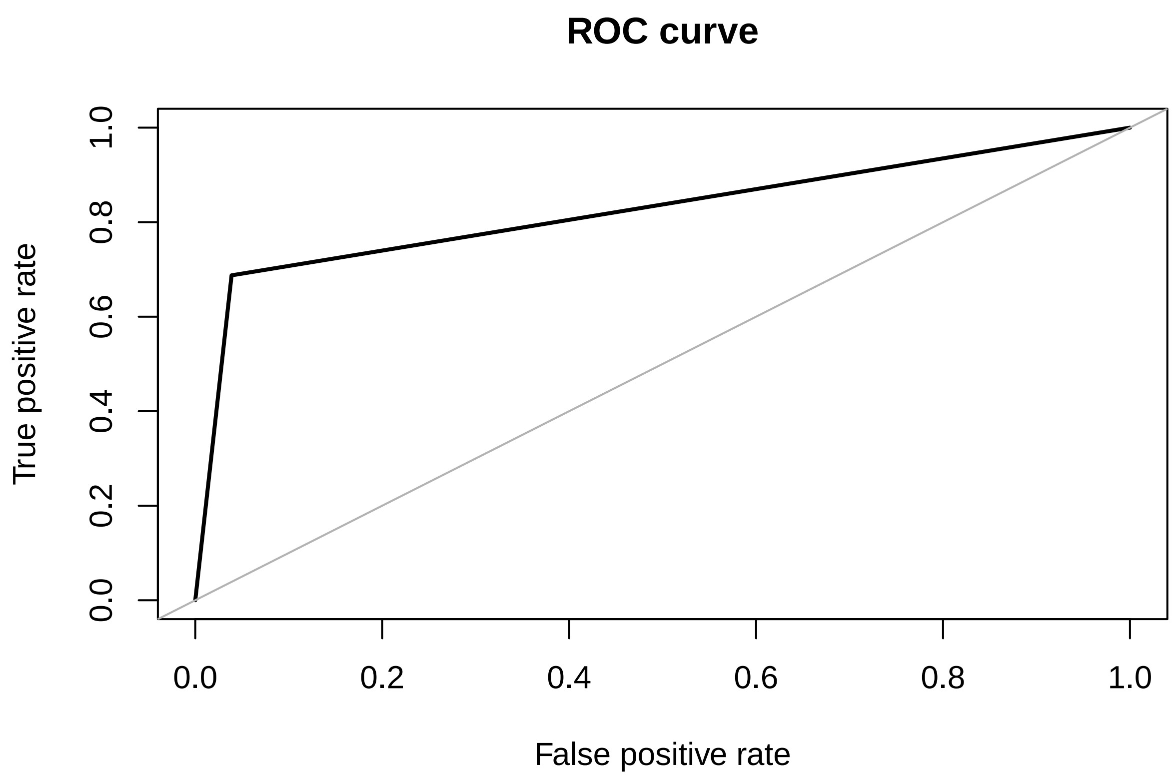
```
confusionMatrix(dados_teste$is_attributed,pred.svm.rose)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 8384  599
##          1    7    9
##
##                Accuracy : 0.9327
##                  95% CI : (0.9273, 0.9378)
##     No Information Rate : 0.9324
##     P-Value [Acc > NIR] : 0.4773
##
##                   Kappa : 0.0255
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9992
##             Specificity : 0.0148
##          Pos Pred Value : 0.9333
##          Neg Pred Value : 0.5625
##              Prevalence : 0.9324
##          Detection Rate : 0.9317
##    Detection Prevalence : 0.9982
```

```
##       Balanced Accuracy : 0.5070
##
##        'Positive' Class : 0
##
```

#Modelo : Random Forest

```r
roc.curve(dados_teste$is_attributed,final_smote)
```




```
## Area under the curve (AUC): 0.824
```

```r
confusionMatrix(dados_teste$is_attributed,final_smote)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 8634  349
##          1    5   11
##
##                Accuracy : 0.9607
##                  95% CI : (0.9564, 0.9646)
##     No Information Rate : 0.96
##     P-Value [Acc > NIR] : 0.3866
##
##                   Kappa : 0.0553
##
##  Mcnemar's Test P-Value : <2e-16
```

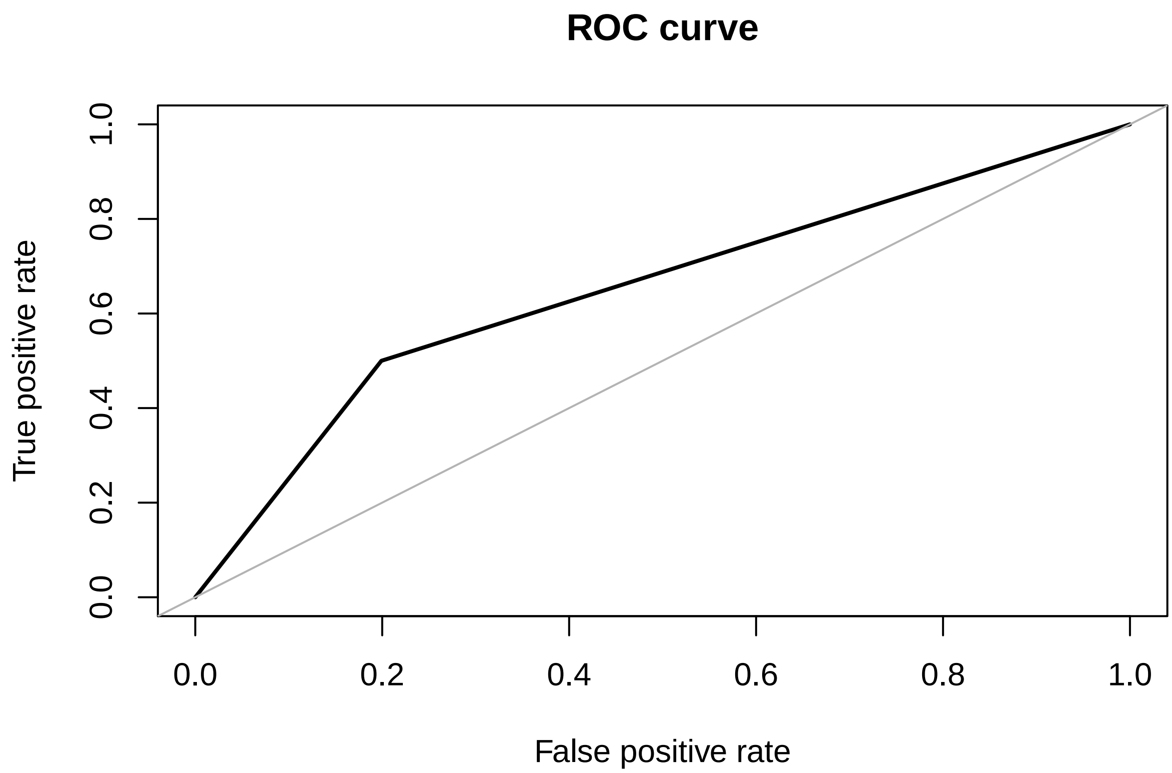
```
##
##             Sensitivity : 0.99942
##             Specificity : 0.03056
##          Pos Pred Value : 0.96115
##          Neg Pred Value : 0.68750
##              Prevalence : 0.96000
##          Detection Rate : 0.95944
##    Detection Prevalence : 0.99822
##       Balanced Accuracy : 0.51499
##
##        'Positive' Class : 0
##
```

#Modelo : Logistic Regression

```r
confusionMatrix(dados_teste$is_attributed,glm.pred)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 7194 1789
##          1    8    8
##
##                Accuracy : 0.8003
##                  95% CI : (0.7919, 0.8085)
##     No Information Rate : 0.8003
##     P-Value [Acc > NIR] : 0.5063
##
##                   Kappa : 0.0053
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.998889
##             Specificity : 0.004452
##          Pos Pred Value : 0.800846
##          Neg Pred Value : 0.500000
##              Prevalence : 0.800311
##          Detection Rate : 0.799422
##    Detection Prevalence : 0.998222
##       Balanced Accuracy : 0.501671
##
##        'Positive' Class : 0
##
```

```r
roc.curve(dados_teste$is_attributed,glm.pred)
```




```
## Area under the curve (AUC): 0.650
```