Projeto 01 - Detecção de Fraudes no Tráfego de Cliques em Propagandas de Aplicações Mobile

Lorena França Almeida

22 de Agosto, 2021

Definição de Diretório e Importação da Amostra

```
# Definindo o diretório de trabalho
setwd("C:/FCD/BigDataRAzure/Cap20/Projeto01")

# Checando o diretório atual de trabalho
getwd()
```

```
## [1] "C:/FCD/BigDataRAzure/Cap20/Projeto01"
```

```
# Importando arquivo da amostra reduzida
amostra1 <- read.csv("train_sample.csv")
head(amostra1)
```

```
##       ip app device os channel          click_time attributed_time
## 1  87540  12      1 13     497 2017-11-07 09:30:38
## 2 105560  25      1 17     259 2017-11-07 13:40:27
## 3 101424  12      1 19     212 2017-11-07 18:05:24
## 4  94584  13      1 13     477 2017-11-07 04:58:08
## 5  68413  12      1  1     178 2017-11-09 09:00:09
## 6  93663   3      1 17     115 2017-11-09 01:22:13
##   is_attributed
## 1             0
## 2             0
## 3             0
## 4             0
## 5             0
## 6             0
```

Data Munging e Análise Exploratória

```
# Explorando os dados
summary(amostra1)
```

```
##        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
##  Min.   :  3.0   2017-11-08 12:01:02:    7                      :99773
##  1st Qu.:145.0   2017-11-07 04:36:16:    6   2017-11-06 17:19:04:    1
##  Median :258.0   2017-11-07 05:00:11:    6   2017-11-06 21:30:47:    1
##  Mean   :268.8   2017-11-08 13:32:05:    6   2017-11-06 23:16:28:    1
##  3rd Qu.:379.0   2017-11-09 14:46:23:    6   2017-11-06 23:58:31:    1
##  Max.   :498.0   2017-11-06 23:27:07:    5   2017-11-07 00:15:11:    1
##                  (Other)            :99964   (Other)            :  222
##  is_attributed
##  Min.   :0.00000
##  1st Qu.:0.00000
##  Median :0.00000
##  Mean   :0.00227
##  3rd Qu.:0.00000
##  Max.   :1.00000
##
```

```
str(amostra1)
```

```
## '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     : Factor w/ 80350 levels "2017-11-06 16:00:00",..: 17416 23124 27845 11509 70546 5
##  $ attributed_time: Factor w/ 228 levels "","2017-11-06 17:19:04",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ is_attributed  : int  0 0 0 0 0 0 0 0 0 0 ...
```

```
# Convertendo a variável target para fator
amostra1$is_attributed <- as.factor(amostra1$is_attributed)

# Capturando a hora da variável click_time
library("lubridate")
```

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

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

```
amostra1$click_time <- as_datetime(amostra1$click_time)
amostra1$click_time <- format(amostra1$click_time,"%H")

# Visualizando a frequência de algumas variáveis

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
```

```
head(count(amostra1, app, sort = TRUE), 25)
```

```
##    app     n
## 1    3 18279
## 2   12 13198
## 3    2 11737
## 4    9  8992
## 5   15  8595
## 6   18  8315
## 7   14  5359
## 8    1  3135
## 9   13  2422
## 10   8  2004
## 11  21  1979
## 12  11  1927
## 13  26  1633
## 14  23  1454
## 15   6  1303
## 16  64  1079
## 17   7   981
## 18  20   911
## 19  25   804
## 20  28   720
## 21  24   704
## 22  27   696
## 23  19   478
## 24  10   388
## 25  22   386
```

```
head(count(amostra1, channel, sort = TRUE), 25)
```

```
##    channel    n
## 1      280 8114
## 2      245 4802
## 3      107 4543
## 4      477 3960
## 5      134 3224
## 6      259 3130
## 7      265 3013
## 8      153 2954
## 9      178 2936
## 10     121 2472
## 11     205 2369
## 12     145 1964
## 13     442 1941
## 14     459 1921
## 15     379 1833
## 16     439 1528
## 17     128 1486
## 18     466 1483
## 19     135 1473
## 20     480 1468
```

```
## 21     469 1458
## 22     489 1426
## 23     237 1408
## 24     122 1366
## 25     140 1328
```

```
head(count(amostra1, ip, sort = TRUE), 25)
```

```
##        ip   n
## 1    5348 669
## 2    5314 616
## 3   73487 439
## 4   73516 399
## 5   53454 280
## 6  114276 219
## 7   26995 218
## 8   95766 205
## 9   17149 186
## 10 100275 173
## 11 105475 167
## 12 105560 149
## 13 111025 137
## 14  43793 135
## 15  86767 134
## 16 137052 128
## 17   5178 117
## 18  49602 116
## 19  48170 112
## 20  48282 112
## 21   5147 109
## 22 201182 104
## 23  45745 100
## 24 209663  96
## 25 114220  95
```

```
nrow(distinct(amostra1, ip))
```

```
## [1] 34857
```

```
round((nrow(distinct(amostra1, ip))/100000)*100, 1)
```

```
## [1] 34.9
```

Temos 34.857 IPs distintos nessa amostra, em um total de 100.000 registros, representando 34,9% da amostra.

```
# IPs e suas respectivas quantidades de clicks
contador <- count(amostra1, ip, sort = TRUE)
head(contador)
```

```
##       ip   n
## 1   5348 669
## 2   5314 616
## 3  73487 439
## 4  73516 399
## 5  53454 280
## 6 114276 219
```

```r
nrow(contador)
```

```
## [1] 34857
```

```r
# Quantidade de IPs que clicaram mais de uma vez
IpMaiorQueUm <- subset(contador, n > 1, sort = TRUE)
head(IpMaiorQueUm)
```

```
##       ip   n
## 1   5348 669
## 2   5314 616
## 3  73487 439
## 4  73516 399
## 5  53454 280
## 6 114276 219
```

```r
nrow(IpMaiorQueUm)
```

```
## [1] 17434
```

```r
# Quantidade de IPs que clicaram somente uma vez
IpIgualAUm <- subset(contador, n == 1, sort = TRUE)
head(IpIgualAUm)
```

```
##       ip n
## 17435  9 1
## 17436 19 1
## 17437 25 1
## 17438 31 1
## 17439 33 1
## 17440 85 1
```

```r
nrow(IpIgualAUm)
```

```
## [1] 17423
```

```r
# Confirmando total de IPs da divisão
nrow(IpIgualAUm) + nrow(IpMaiorQueUm)
```

```
## [1] 34857
```

```r
# Inserindo a coluna is_attributed ao subset de IPs que clicaram somente uma vez
juncao <- left_join(IpIgualAUm, amostra1 %>% select(ip, is_attributed),
                    by = c("ip" = "ip"))

head(juncao)
```

```
##   ip n is_attributed
## 1  9 1             0
## 2 19 1             0
## 3 25 1             0
## 4 31 1             0
## 5 33 1             0
## 6 85 1             0
```

```r
nrow(juncao)
```

```
## [1] 17423
```

```r
# Quantidade de IPs que clicaram só uma vez e não fizeram download
unicoNok <- subset(juncao, is_attributed == 0, sort = TRUE)
head(unicoNok)
```

```
##   ip n is_attributed
## 1  9 1             0
## 2 19 1             0
## 3 25 1             0
## 4 31 1             0
## 5 33 1             0
## 6 85 1             0
```

```r
nrow(unicoNok)
```

```
## [1] 17273
```

```r
# Quantidade de IPs que clicaram só uma vez e fizeram download
unicoOk <- subset(juncao, is_attributed == 1, sort = TRUE)
head(unicoOk)
```

```
##         ip n is_attributed
## 192   2948 1             1
## 394   6192 1             1
## 528   7909 1             1
## 1008 15195 1             1
## 1409 20806 1             1
## 1664 24458 1             1
```

```r
nrow(unicoOk)
```

```
## [1] 150
```

```r
# Confirmando divisão das quantidades de IPs que clicaram somente uma vez
nrow(unicoNok) + nrow(unicoOk)
```

```
## [1] 17423
```

```r
# Percentual de IPs que clicaram somente uma vez e fizeram download
(nrow(unicoOk)/nrow(IpIgualAUm))*100
```

```
## [1] 0.860931
```

```r
# Dos IPs que clicaram mais de uma vez, quantas foram as vezes que fizeram e que
# não fizeram download?
head(IpMaiorQueUm)
```

```
##       ip   n
## 1   5348 669
## 2   5314 616
## 3  73487 439
## 4  73516 399
## 5  53454 280
## 6 114276 219
```

```r
nrow(IpMaiorQueUm)
```

```
## [1] 17434
```

```r
SemDown <- as.data.frame(amostra1 %>%
  filter(is_attributed == 0) %>%
```

```
  group_by(ip) %>%
  summarise(is_attributed = n()) %>%
              arrange(desc(is_attributed)))

head(SemDown)
```

```
##       ip is_attributed
## 1   5348           666
## 2   5314           613
## 3  73487           439
## 4  73516           399
## 5  53454           280
## 6 114276           219
```

```
ComDown <- as.data.frame(amostra1 %>%
                           filter(is_attributed == 1) %>%
                           group_by(ip) %>%
                           summarise(is_attributed = n()) %>%
                           arrange(desc(is_attributed)))

head(ComDown)
```

```
##     ip is_attributed
## 1 5314             3
## 2 5348             3
## 3 2948             1
## 4 4865             1
## 5 5281             1
## 6 5729             1
```

```
IpsRepetidos <- left_join(
                          left_join(IpMaiorQueUm, SemDown %>%
                          select(ip, "SemDown" = is_attributed),
                          by = c("ip" = "ip")),
                          ComDown %>%
                          select(ip, "ComDown" = is_attributed),
                          by = c("ip" = "ip"))

head(IpsRepetidos)
```

```
##       ip   n SemDown ComDown
## 1   5348 669     666       3
## 2   5314 616     613       3
## 3  73487 439     439      NA
## 4  73516 399     399      NA
## 5  53454 280     280      NA
## 6 114276 219     219      NA
```

```
nrow(IpsRepetidos)
```

```
## [1] 17434
```

```
nrow(subset(IpsRepetidos, ComDown >= 1))
```

```
## [1] 73
```

```
(nrow(subset(IpsRepetidos, ComDown >= 1))/
    nrow(IpsRepetidos))*100
```

```
## [1] 0.418722
```

```
# Dos IPs que clicaram mais de uma vez, 0,42% fizeram ao menos um download.

# Quantidade total de downloads da amostra
nrow(subset(IpsRepetidos, ComDown >= 1))+nrow(unicoOk)
```

```
## [1] 223
```

```
# Do total de downloads da amostra, segue o percentual dos que foram feitos por
# IPs que clicaram somente uma vez:
(nrow(unicoOk)/(nrow(subset(IpsRepetidos, ComDown >= 1))+nrow(unicoOk)))*100
```

```
## [1] 67.26457
```

```
# Do total de downloads da amostra, segue o percentual dos que foram feitos por
# IPs que clicaram mais de uma vez:
(nrow(subset(IpsRepetidos, ComDown >= 1))/
(nrow(subset(IpsRepetidos, ComDown >= 1))+nrow(unicoOk)))*100
```

```
## [1] 32.73543
```

Isso mostra que, da nossa quantidade de downloads feitos, 67% foram realizados por IPs que clicaram somente uma vez e 33% realizados por IPs que clicaram mais de uma vez. No entanto, destes IPs que clicaram mais de uma vez, de forma geral, a quantidade de clicks sem download é muito maior do que a quantidade de clicks com download, podendo supor que os IPs que clicam mais de uma vez possuem uma probabilidade maior de clicks fraudulentos do que os IPs que clicam apenas uma vez. Tivemos 73 IPs nessa amostra que clicaram mais de uma vez e realizaram pelo menos um download:

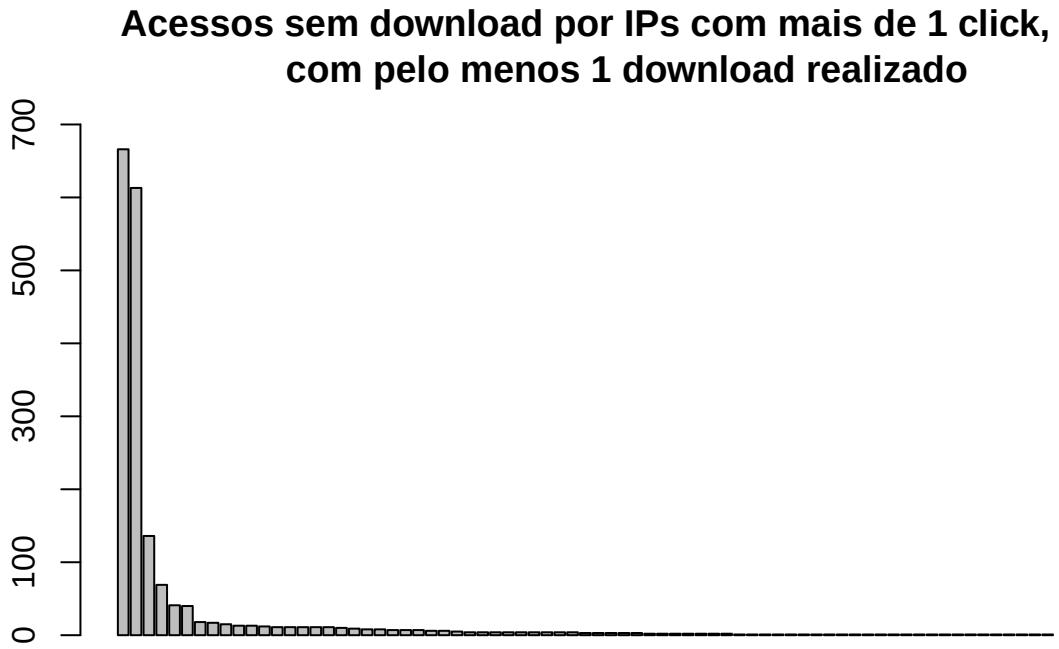
```
clicsokipsduplos <- head(IpsRepetidos[order(IpsRepetidos$ComDown,
                                            decreasing = TRUE),],73)
clicsokipsduplos
```

```
##           ip   n SemDown ComDown
## 1       5348 669     666       3
## 2       5314 616     613       3
## 13    111025 137     136       1
## 50     44067  70      69       1
## 91      5729  42      41       1
## 95    118252  41      40       1
## 324    93542  19      18       1
## 348   102174  18      17       1
## 421   102511  16      15       1
## 522    40654  14      13       1
## 527    48337  14      13       1
## 588     7350  13      12       1
## 705     6481  12      11       1
## 708     8180  12      11       1
## 713    12506  12      11       1
## 724    17460  12      11       1
## 737    35308  12      11       1
## 992    88914  11      10       1
## 1160   55142  10       9       1
## 1519   76989   9       8       1
```

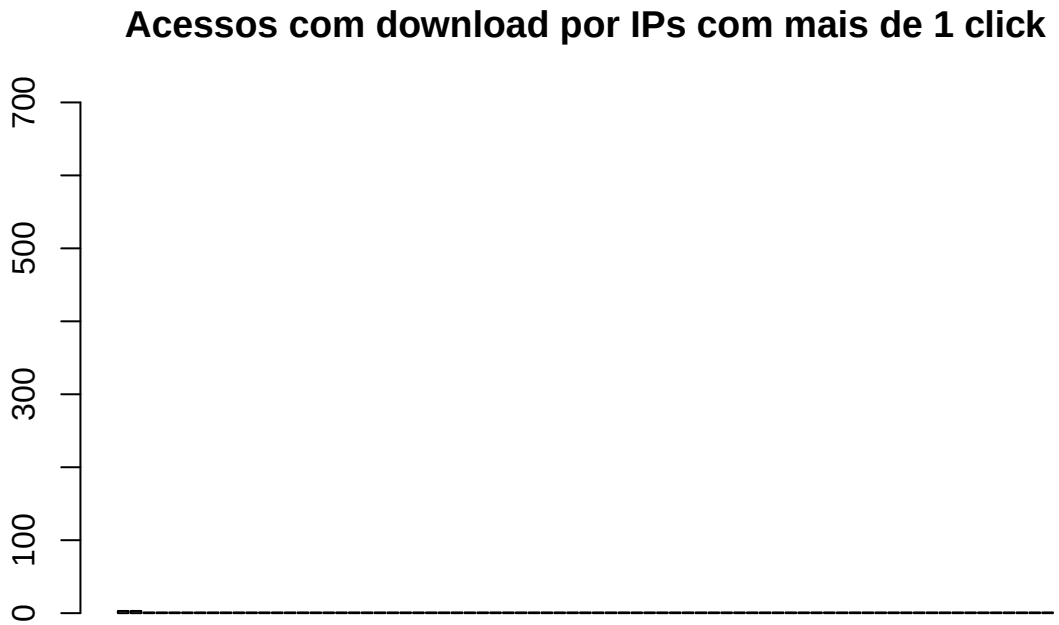
```
## 1629  115265   9       8       1
## 1920   69145   8       7       1
## 1929   71579   8       7       1
## 1978   86812   8       7       1
## 2332   48072   7       6       1
## 2510   85536   7       6       1
## 3394  110652   6       5       1
## 3592    5281   5       4       1
## 3671   14888   5       4       1
## 3684   16535   5       4       1
## 3804   31859   5       4       1
## 4059   60666   5       4       1
## 4486  113865   5       4       1
## 4510  116718   5       4       1
## 4540  120148   5       4       1
## 4712  196491   5       4       1
## 4895   11467   4       3       1
## 5104   29016   4       3       1
## 5182   34399   4       3       1
## 6280  116785   4       3       1
## 6713  258509   4       3       1
## 7909   48733   3       2       1
## 8004   53389   3       2       1
## 8251   63776   3       2       1
## 8788   89457   3       2       1
## 9084  103059   3       2       1
## 9979  184467   3       2       1
## 10255 242468   3       2       1
## 10527   4865   2       1       1
## 10810  11967   2       1       1
## 11046  18204   2       1       1
## 11182  22109   2       1       1
## 11502  31277   2       1       1
## 11552  32463   2       1       1
## 11801  38943   2       1       1
## 11847  40352   2       1       1
## 11911  42074   2       1       1
## 12497  56317   2       1       1
## 12584  58560   2       1       1
## 13191  75443   2       1       1
## 13944  95207   2       1       1
## 14282 104110   2       1       1
## 14624 112271   2       1       1
## 14954 120259   2       1       1
## 15388 137509   2       1       1
## 15753 157074   2       1       1
## 16137 177454   2       1       1
## 16185 180418   2       1       1
## 16313 186843   2       1       1
## 16441 193539   2       1       1
## 16636 202255   2       1       1
## 16794 210641   2       1       1
## 17165 268081   2       1       1
```

A quantidade de acessos dos 73 IPs com mais de um click sem download pode ser comparada com a quantidade de acessos com download dos mesmos IPs através dos gráficos de barras a seguir:

```
barplot(clicsokipsduplos$SemDown, ylim = c(0,700),
        main = "Acessos sem download por IPs com mais de 1 click,
        com pelo menos 1 download realizado")
```




```
barplot(clicsokipsduplos$ComDown, ylim = c(0,700),
        main = "Acessos com download por IPs com mais de 1 click")
```



Ou seja, a classificação de um IP como repetido ou não nesse espaço de tempo pode ser significativa para a criação do modelo preditivo. Portanto, é criada uma coluna ao dataset da amostra classificando o IP como repetido = TRUE ou não repetido (único) = FALSE.

```
amostra1$IpRepetido <- amostra1$ip %in% IpMaiorQueUm$ip
head(amostra1)
```

```
##       ip app device os channel click_time attributed_time is_attributed
## 1  87540  12      1 13     497         09                              0
## 2 105560  25      1 17     259         13                              0
## 3 101424  12      1 19     212         18                              0
## 4  94584  13      1 13     477         04                              0
## 5  68413  12      1  1     178         09                              0
## 6  93663   3      1 17     115         01                              0
##   IpRepetido
## 1       TRUE
## 2       TRUE
## 3       TRUE
## 4       TRUE
## 5       TRUE
## 6       TRUE
```

```
nrow(amostra1[amostra1$IpRepetido == FALSE,])
```

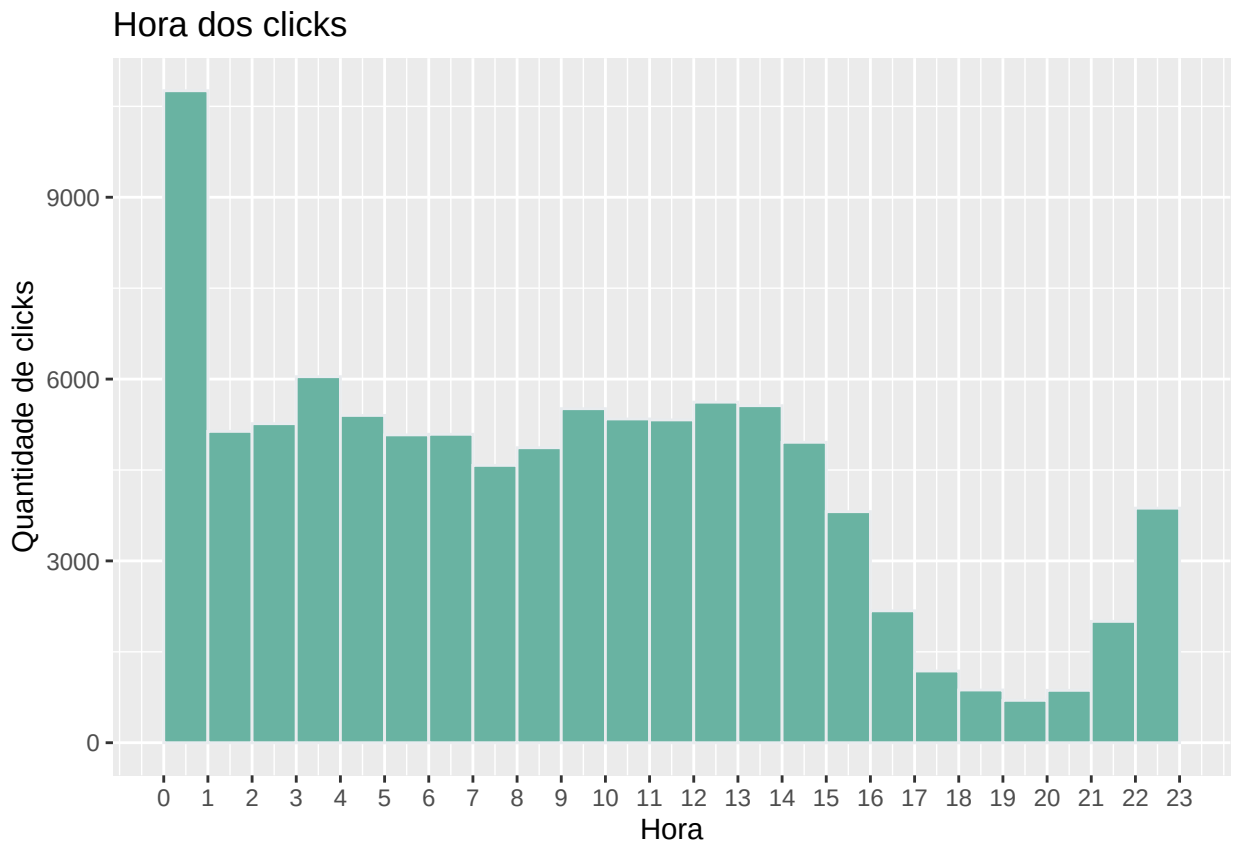
```
## [1] 17423
```

```
# Convertendo a variável IpRepetido para fator
amostra1$IpRepetido <- as.factor(amostra1$IpRepetido)
```

```
# Observando os clicks por horário
library("ggplot2")

# Convertendo a variável click_time para integer
amostra1$click_time <- as.integer(amostra1$click_time)

ggplot(amostra1, aes(click_time)) +
  geom_histogram(fill="#69b3a2", color="#e9ecef", alpha=1,
  breaks = seq(0, 23, by = 1)) +
  labs(x = "Hora",
       title = "Hora dos clicks",
       y = "Quantidade de clicks") +
  theme_update(plot.title = element_text(hjust = 0.5)) +
  scale_x_continuous(breaks = seq(0, 23, by=1))
```




```
# Convertendo a variável click_time para fator
amostra1$click_time <- as.factor(amostra1$click_time)

# Checando as conversões das variáveis
str(amostra1)
```

```
## 'data.frame':    100000 obs. of  9 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     : Factor w/ 24 levels "0","1","2","3",..: 10 14 19 5 10 2 2 11 10 13 ...
##  $ attributed_time: Factor w/ 228 levels "","2017-11-06 17:19:04",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ is_attributed  : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ IpRepetido     : Factor w/ 2 levels "FALSE","TRUE": 2 2 2 2 2 2 2 2 2 1 ...
```

Feature Selection

```
library("randomForest")
```

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

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

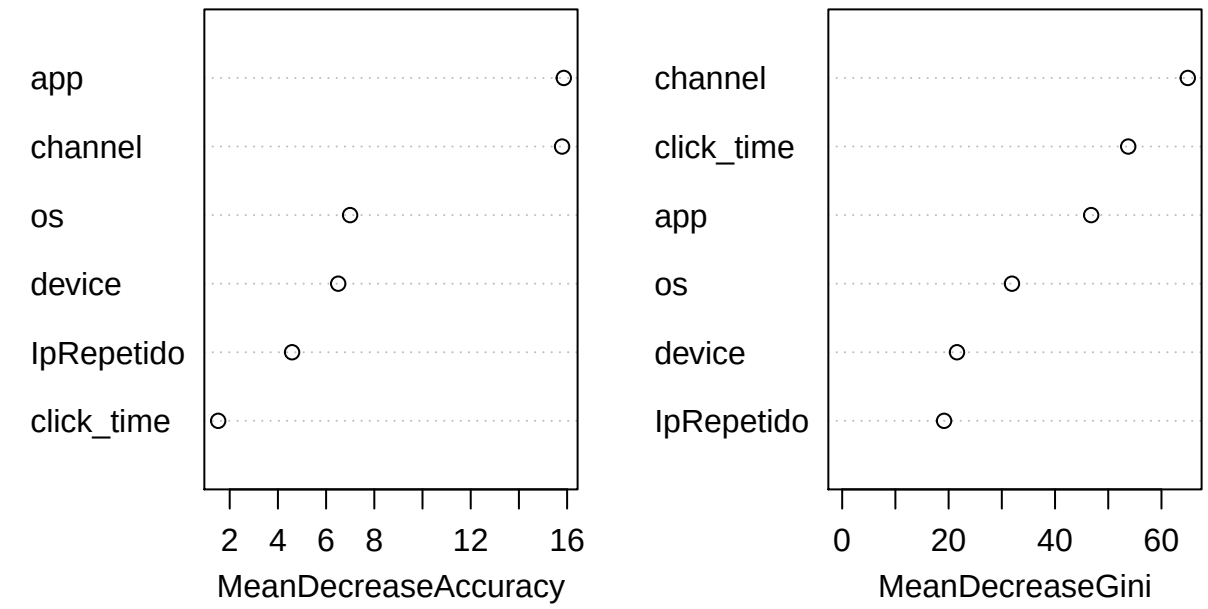
```
##
## Attaching package: 'randomForest'
```

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

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

```
modelo <- randomForest(is_attributed ~ . - ip  - attributed_time,
                       data = amostra1,
                       ntree = 100, nodesize = 10, importance = T)

varImpPlot(modelo, main = "Importância das Variáveis")
```




```
# Variáveis selecionadas para construção do modelo
selectedSample <-
  amostra1[, c('app', 'device', 'os', 'channel', 'IpRepetido', 'is_attributed')]
```

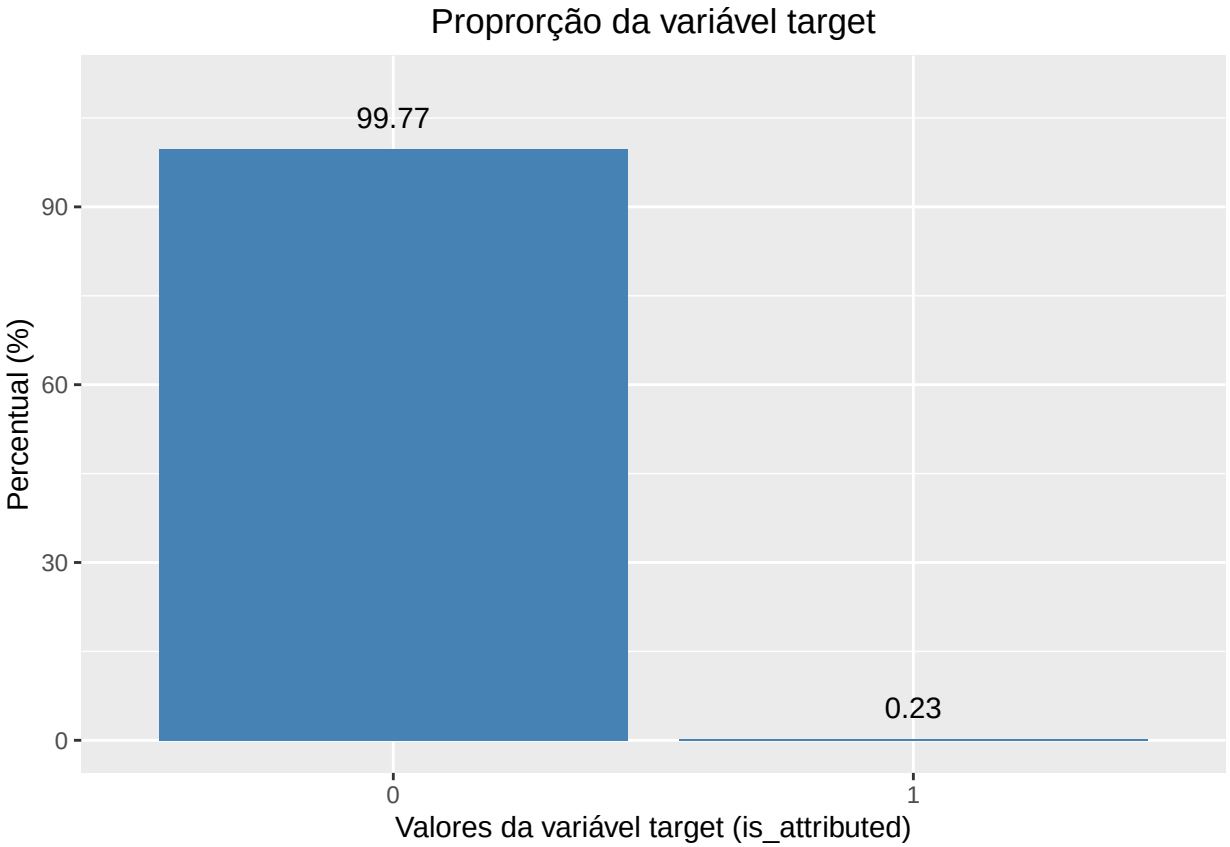
Balanceamento de Classe

```
# Checando proporção da variável target
round(prop.table(table(selectedSample$is_attributed))*100, 2)
```

```
##
##     0     1
## 99.77  0.23
```

```
df <- as.data.frame(round(prop.table(table(selectedSample$is_attributed))*100, 2))

ggplot(data=df, aes(x=Var1, y=Freq)) +
  geom_text(aes(x = Var1, y = Freq), label = df$Freq, vjust = -1) +
  geom_bar(stat="identity", fill="steelblue") +
  labs(x = "Valores da variável target (is_attributed)",
       title = "Proprorção da variável target",
       y = "Percentual (%)") +
  theme_update(plot.title = element_text(hjust = 0.5)) +
  scale_y_continuous(limits = c(0,110))
```



Nota-se que a variável está bastante desbalanceada, tendo cerca de 99,77% de registros para a classe 0 e 0,23% para a classe 1.

```
# Dividindo os dados em treino e teste - 70:30 ratio
set.seed(2021)
indexes <- sample(1:nrow(selectedSample), size = 0.7 * nrow(selectedSample))
train.data <- selectedSample[indexes,]
test.data <- selectedSample[-indexes,]

# Checando proporção da divisão para cada valor da classe
prop.table(table(train.data$is_attributed))
```

```
##
##      0      1
## 0.9976 0.0024
```

```
prop.table(table(test.data$is_attributed))
```

```
##
##           0           1
## 0.998033333 0.001966667
```

```
# Balanceando a classe
library(ROSE)
```

```
## Loaded ROSE 0.0-4
```

```
# Aplicando ROSE em dados de treino e checando a proporção de classes
rose_treino <- ROSE(is_attributed ~ ., data = train.data, seed = 1)$data
```

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

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

```
# Aplicando ROSE em dados de teste e checando a proporção de classes
rose_teste <- ROSE(is_attributed ~ ., data = test.data, seed = 1)$data
prop.table(table(rose_teste$is_attributed))
```

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

Com uma proporção mais balanceada, segue-se com a criação do modelo.

Criação e Score do Modelo Supervisionado de Machine Learning

```
# Ativando pacote para modelo de árvore de decisão
library(C50)

# Criando o modelo com os dados de treino balanceados
set.seed(2021)
modelotree <- C5.0(is_attributed ~ ., data = rose_treino)

# Gerando previsões nos dados de teste balanceados
set.seed(2021)
previsoestree <- data.frame(observado = rose_teste$is_attributed,
                            previsto = predict(modelotree, newdata = rose_teste))
```

Avaliação do Modelo

```
# Avaliando o modelo
library(caret)
```

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

```
confusionMatrix(previsoestree$observado, previsoestree$previsto)
```

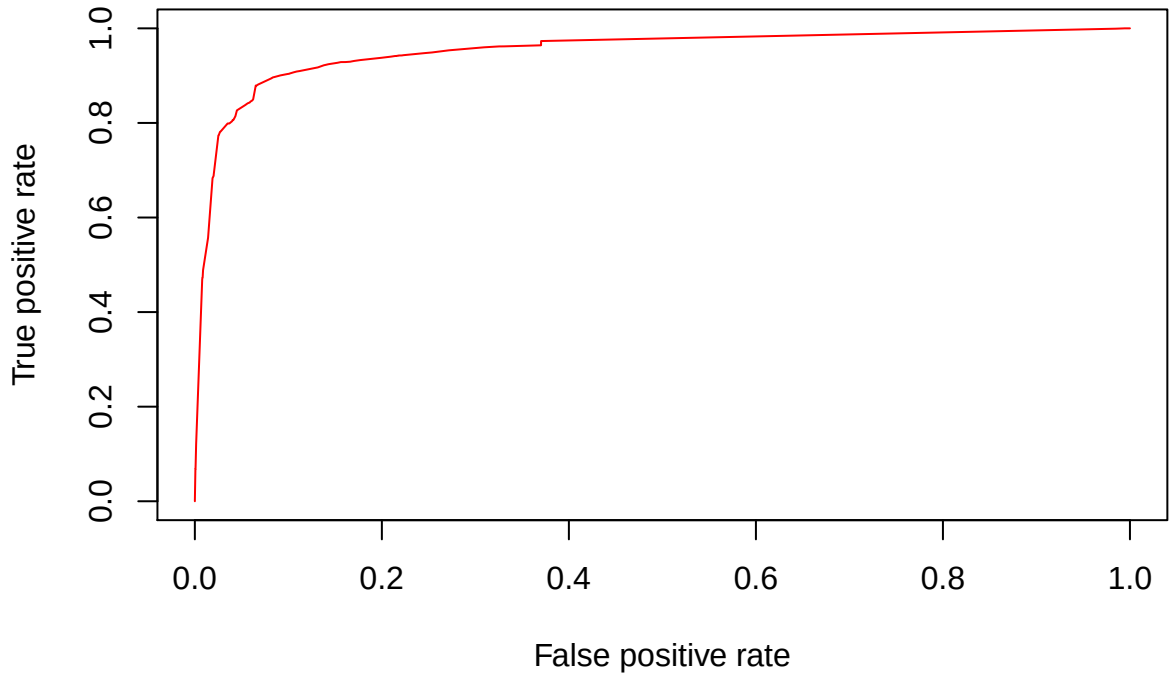
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0 13133  1988
##          1  1229 13650
##
##                Accuracy : 0.8928
##                  95% CI : (0.8892, 0.8962)
##     No Information Rate : 0.5213
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7856
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
```

```
##             Sensitivity : 0.9144
##             Specificity : 0.8729
##          Pos Pred Value : 0.8685
##          Neg Pred Value : 0.9174
##              Prevalence : 0.4787
##          Detection Rate : 0.4378
##    Detection Prevalence : 0.5040
##       Balanced Accuracy : 0.8937
##
##        'Positive' Class : 0
##
```

```r
# Acurácia de 89,28%

# Gerando as classes de dados para curva ROC
classe1 <- predict(modelotree, newdata = rose_teste, type = 'prob')
classe2 <- rose_teste$is_attributed

# Gerando a curva ROC
library("ROCR")
pred <- prediction(classe1[,2], classe2)
perf <- performance(pred, "tpr","fpr")
curva <- plot(perf, col = rainbow(10))
```




```r
# Verificando AUC:
library("pROC")
```

```
## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
calcAUC <- roc(classe2, classe1[,2])
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```
auc(calcAUC)
```

```
## Area under the curve: 0.9527
```

```
# AUC de 95,27%
```

Conclusão: Foi criado um modelo de árvore de decisão com acurácia de 89% e área abaixo da curva de 95%.