Mini Projeto - Data Science Academy

André Campos da Silva

18 de Novembro, 2020

Projeto - Detecção de Fraudes

Construir um modelo de analise de fraudes com os dados históricos,com a finalidade de determinar se um futuro clique pode ser fraudulento ou não. Usarei os dados disponibilizados no site do https://www.kaggle.com/c/talkingdata-adtracking-fraud-detection/data (https://www.kaggle.com/c/talkingdata-adtracking-fraud-detection/data)

Coletando os dados

```
# Carrego os pacotes necessários para o projeto
library('tidyverse')
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library('caret')
```

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

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library('ROSE')
```

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

```r
library('data.table')
```

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
## The following object is masked from 'package:purrr':
##
##     transpose
```

```r
library('gridExtra')
```

```
##
## Attaching package: 'gridExtra'
```

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

```r
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:gridExtra':
##
##     combine
```

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

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

```
library('DMwR')
```

```
## Loading required package: grid
```

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

```
library('e1071')
library('rpart')
library('C50')
library("ROCR")
```

```
# Carrego os dados de treino que será tratado e usado para a análise e treinamento.
train_default <- fread('Dados/train_sample.csv',data.table = FALSE, tz="UTC")
head(train_default)
```

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

```
# Faço uma verificação do formato dos dados.
glimpse(train_default)
```

```
## Rows: 100,000
## Columns: 8
## $ ip              <int> 87540, 105560, 101424, 94584, 68413, 93663, 17059, ...
## $ app             <int> 12, 25, 12, 13, 12, 3, 1, 9, 2, 3, 3, 3, 3, 6, 2, 2...
## $ device          <int> 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 2, 1, ...
## $ os              <int> 13, 17, 19, 13, 1, 17, 17, 25, 22, 19, 22, 13, 22, ...
## $ channel         <int> 497, 259, 212, 477, 178, 115, 135, 442, 364, 135, 4...
## $ click_time      <dttm> 2017-11-07 09:30:38, 2017-11-07 13:40:27, 2017-11-...
## $ attributed_time <dttm> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA...
## $ is_attributed   <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
```

Tratamento dos dados

```
# Como no dataset de teste não tem a coluna attributed_time, eu vou tira-la do dataset de treino
# na minha análise ela não é relevante, afinal não teremos ela para os testes.
train_default$attributed_time <- NULL
head(train_default)
```

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

```
# Verifico que o dataset só tem um ano e um mês para análise.
table(year(train_default$click_time))
```

```
## 
##   2017 
## 100000
```

```
table(month(train_default$click_time))
```

```
## 
##     11 
## 100000
```

```r
# Crio uma função que extrai o dia, hora e minuto da variavel click_time para outras variáveis,
# com seus respectivos nomes, e no final retiro a click_time, não peguei o ano e mês,
# pois ambos são apenas um, 11/2017 como verificado anteriormente.
create_date <- function(x, y ){
   for (i in y){
      x$click_Day <- weekdays(y)
      x$click_Hour <- hour(y)
      x$click_Minute <- minute(y)
      x$click_time <- NULL
   }
   return(x)
}
```

```r
# Faço o teste  antes para ve se vai rodar do jeito desejado.
testeF <- train_default[1:50,]
create_date(testeF, testeF$click_time)
```

```
##        ip app device os channel is_attributed click_Day click_Hour click_Minute
## 1   87540  12      1 13     497             0   Tuesday          9           30
## 2  105560  25      1 17     259             0   Tuesday         13           40
## 3  101424  12      1 19     212             0   Tuesday         18            5
## 4   94584  13      1 13     477             0   Tuesday          4           58
## 5   68413  12      1  1     178             0  Thursday          9            0
## 6   93663   3      1 17     115             0  Thursday          1           22
## 7   17059   1      1 17     135             0  Thursday          1           17
## 8  121505   9      1 25     442             0   Tuesday         10            1
## 9  192967   2      2 22     364             0 Wednesday          9           35
## 10 143636   3      1 19     135             0 Wednesday         12           35
## 11  73839   3      1 22     489             0 Wednesday          8           14
## 12  34812   3      1 13     489             0   Tuesday          5            3
## 13 114809   3      1 22     205             0  Thursday         10           24
## 14 114220   6      1 20     125             0 Wednesday         14           46
## 15  36150   2      1 13     205             0   Tuesday          0           54
## 16  72116  25      2 19     259             0 Wednesday         23           17
## 17   5314   2      1  2     477             0  Thursday          7           33
## 18 106598   3      1 20     280             0  Thursday          3           44
## 19  72065  20      2 90     259             0    Monday         23           14
## 20  37301  14      1 13     349             0    Monday         20            7
## 21  28735  12      1 19     265             0  Thursday          9           55
## 22  66918  64      1 25     459             0 Wednesday         17            1
## 23  25761   9      1 10     215             0 Wednesday          2            5
## 24   8362   7      1 19     101             0   Tuesday         10           30
## 25  45257   3      1 18     280             0   Tuesday          1           35
## 26 145896  64      1 13     459             0   Tuesday          3           58
## 27 162976   3      1 13     115             0   Tuesday         16           19
## 28  52432   1      1 13     115             0   Tuesday         17           22
## 29 135690  12      1 40     122             0   Tuesday          6           39
## 30 139137  12      1 13     497             0   Tuesday         10           11
## 31  48846   3      1 19     379             0  Thursday          3           14
## 32  70747  64      1 15     459             0   Tuesday         14            0
## 33  10831  15      1 19     386             0  Thursday         12           39
## 34  89242   1      1 27     124             0   Tuesday          9           37
## 35 140138  12      1 13     265             0   Tuesday          7            1
## 36  28411  14      1 13     442             0    Monday         23           45
## 37 127888  13      1 23     477             0    Monday         16           24
## 38  75943   2      1 53     477             0   Tuesday         13           28
## 39  87879   3      1 13     115             0   Tuesday          3           20
## 40 250933   3      1 13     280             0 Wednesday         11            9
## 41 133933  12      1 13     140             0   Tuesday         11           11
## 42  35096  18      1 19     107             0  Thursday         15           26
## 43  49431  15      1 49     245             0   Tuesday         23           11
## 44  53365  15      1 19     111             0  Thursday          4           23
## 45  92599  25      2 13     259             0  Thursday         10            4
## 46 107148  15      1 13     140             0   Tuesday          0           49
## 47 116677   9      1 13     134             0 Wednesday          0           56
## 48  11051  15      1 19     245             0 Wednesday          2           14
## 49  48240   2      1 19     122             0 Wednesday         14           40
## 50  92488  14      1 18     401             0 Wednesday         17           33
```

```
# Faço então a extração no dataset de treino.

# train <- create_date(train_default, train_default$click_time)
# tail(train)
# write_csv(train,'train-tratado.csv')

train <- fread('Dados/train-tratado.csv',data.table = FALSE, tz="UTC")
# A extração acima, demorou algumas horas, então ao terminar eu salvei o dataset tratado
# e carreguei ele novavemente para quando precisar rodar o script novamente não precisar refazer
# esse tratamento.
```

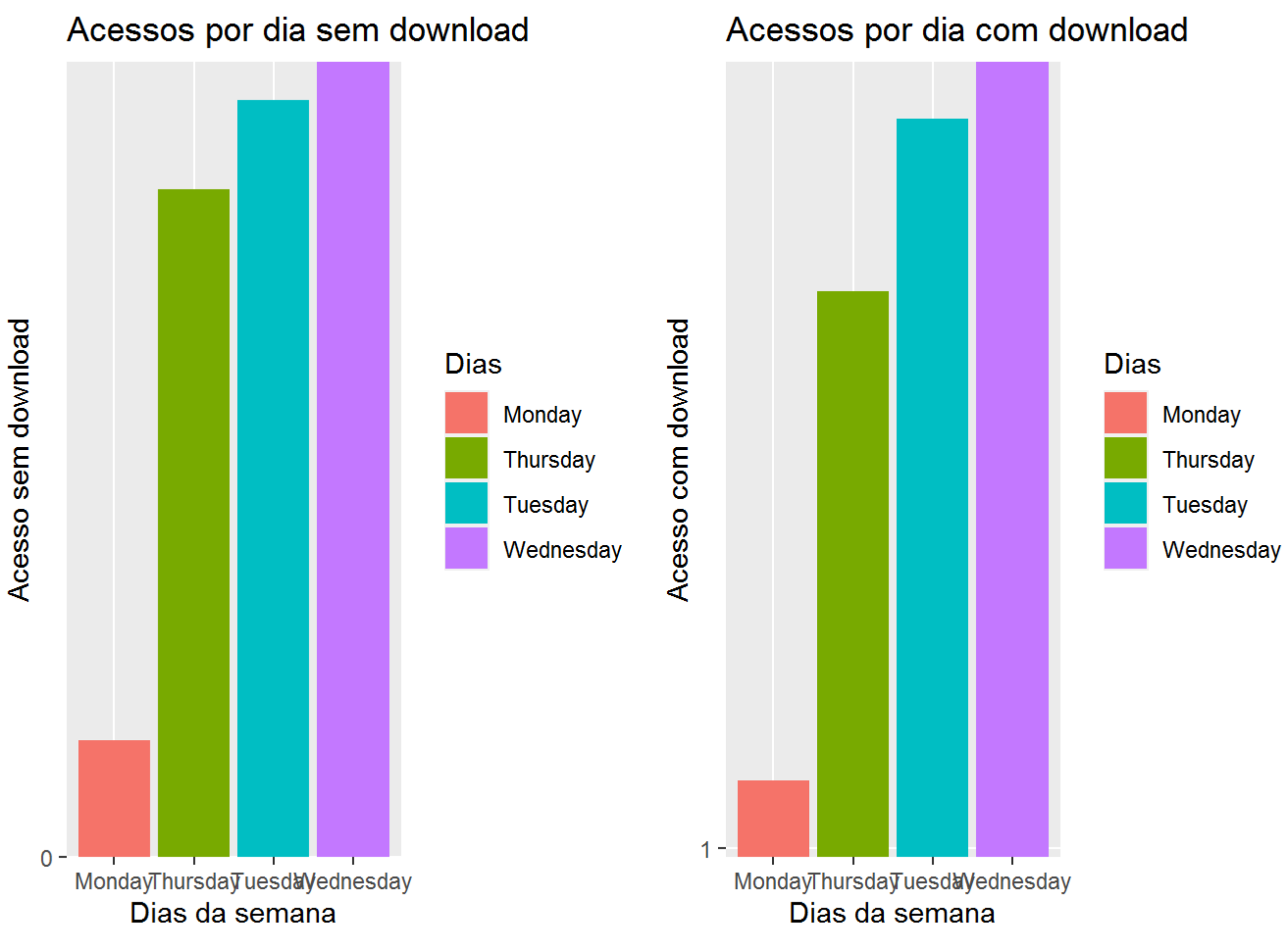
```
head(train)
```

```
##       ip app device os channel is_attributed click_Day click_Hour click_Minute
## 1  87540  12      1 13     497             0   Tuesday          9           30
## 2 105560  25      1 17     259             0   Tuesday         13           40
## 3 101424  12      1 19     212             0   Tuesday         18            5
## 4  94584  13      1 13     477             0   Tuesday          4           58
## 5  68413  12      1  1     178             0  Thursday          9            0
## 6  93663   3      1 17     115             0  Thursday          1           22
```

```
# Crio uma formula agora para converter as variáveis que estão numericas para factor.

var_convert <- names(train)

to_factor <- function(df, var){
   for (i in var){
      df[[i]] <- as.factor(df[[i]])
   }
   return(df)
}
train2 <- to_factor(train,var_convert)
```

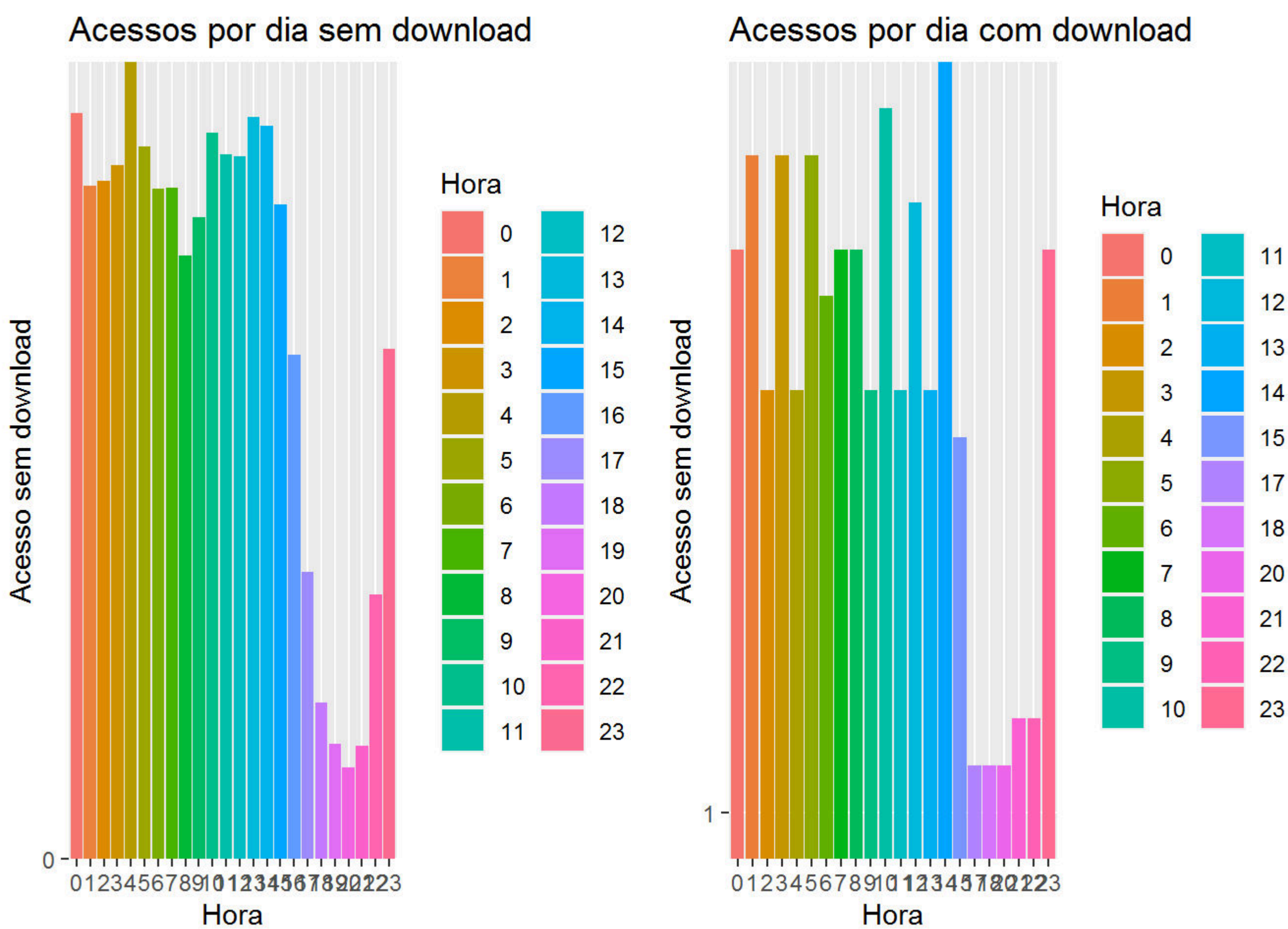
Analise Exploratória

```
# Acessos vs Acessos/Download por dia.
pl1 <- train2 %>%
   filter(is_attributed == 0) %>%
   ggplot(aes(x =click_Day, y = is_attributed, fill =click_Day)) +
   geom_bar(stat = "identity")+
   labs(title = 'Acessos por dia sem download', x = 'Dias da semana',
        y = 'Acesso sem download', fill = 'Dias')

pl2 <- train2 %>%
   filter(is_attributed == 1) %>%
   ggplot(aes(x =click_Day, y = is_attributed, fill = click_Day)) +
   geom_bar(stat = "identity")+
   labs(title = 'Acessos por dia com download', x = 'Dias da semana',
        y = 'Acesso com download', fill = 'Dias')

grid.arrange(pl1,pl2, nrow=1,ncol=2)
```



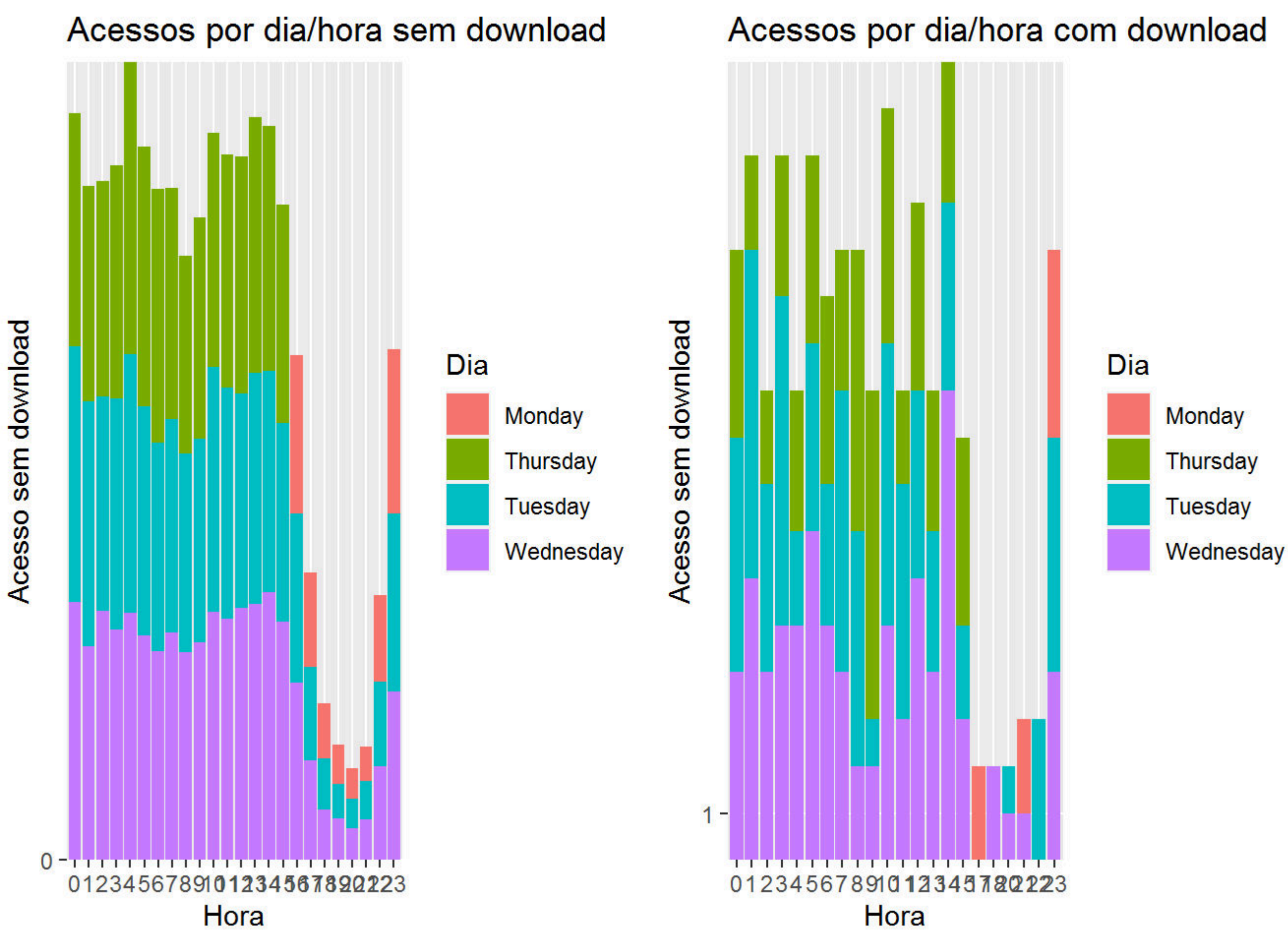

```
# Acessos sem download quanto os com download ocorrem com mais frequência na quarta-feira
```

```
# Acessos vs Acessos/Download por hora.
pl3 <- train2 %>%
   filter(is_attributed == 0) %>%
   ggplot(aes(x =click_Hour, y = is_attributed, fill =click_Hour)) +
   geom_bar(stat = "identity")+
   labs(title = 'Acessos por dia sem download', x = 'Hora',
        y = 'Acesso sem download', fill = 'Hora')

pl4 <- train2 %>%
   filter(is_attributed == 1) %>%
   ggplot(aes(x =click_Hour, y = is_attributed, fill =click_Hour)) +
   geom_bar(stat = "identity")+
   labs(title = 'Acessos por dia com download', x = 'Hora',
        y = 'Acesso sem download', fill = 'Hora')

grid.arrange(pl3,pl4, nrow=1,ncol=2)
```




```
# Acessos sem downloads quanto com downloads ocorrem com mais frequencia entre a madrugada ate o inicio
# da tarde
```

```
# Acessos dia/hora vs Acessos/Download por dia/hora.
pl5 <- train2 %>%
   filter(is_attributed == 0) %>%
   group_by(click_Day,click_Hour)%>%
   ggplot(aes(x =click_Hour, y = is_attributed, fill =click_Day)) +
   geom_bar(stat = "identity")+
   labs(title = 'Acessos por dia/hora sem download', x = 'Hora',
        y = 'Acesso sem download', fill = 'Dia')

pl6 <- train2 %>%
   filter(is_attributed == 1) %>%
   group_by(click_Day,click_Hour)%>%
   ggplot(aes(x =click_Hour, y = is_attributed, fill =click_Day)) +
   geom_bar(stat = "identity")+
   labs(title = 'Acessos por dia/hora com download', x = 'Hora',
        y = 'Acesso sem download', fill = 'Dia')

grid.arrange(pl5,pl6, nrow=1,ncol=2)
```




```
# Existe um padrão para os acessos por dia em relação as horas.
```

```
# Dispositivos mais usados / Dispositivos com mais Downloads.
pl7 <- train2 %>%
   select(is_attributed, device)%>%
   filter(is_attributed == 0) %>%
   group_by(device)%>%
   summarise(Quantidade = table(device))%>%
   filter(Quantidade > 30)%>%
   ggplot(aes(x = '', y = Quantidade, fill = device)) +
   geom_bar(width = 1, stat = "identity") +
   coord_polar("y", start = 0, direction = -1) +
   labs(title = 'Top 6 - Dispositivos mais usados',
        fill = 'Device')+
   theme(
      axis.title.x = element_blank(),
      axis.title.y = element_blank(),
      panel.border = element_blank(),
      panel.grid=element_blank(),
      axis.ticks = element_blank(),
      panel.background = element_blank(),
      axis.text.x=element_blank())
```

```
## `summarise()` regrouping output by 'device' (override with `.groups` argument)
```

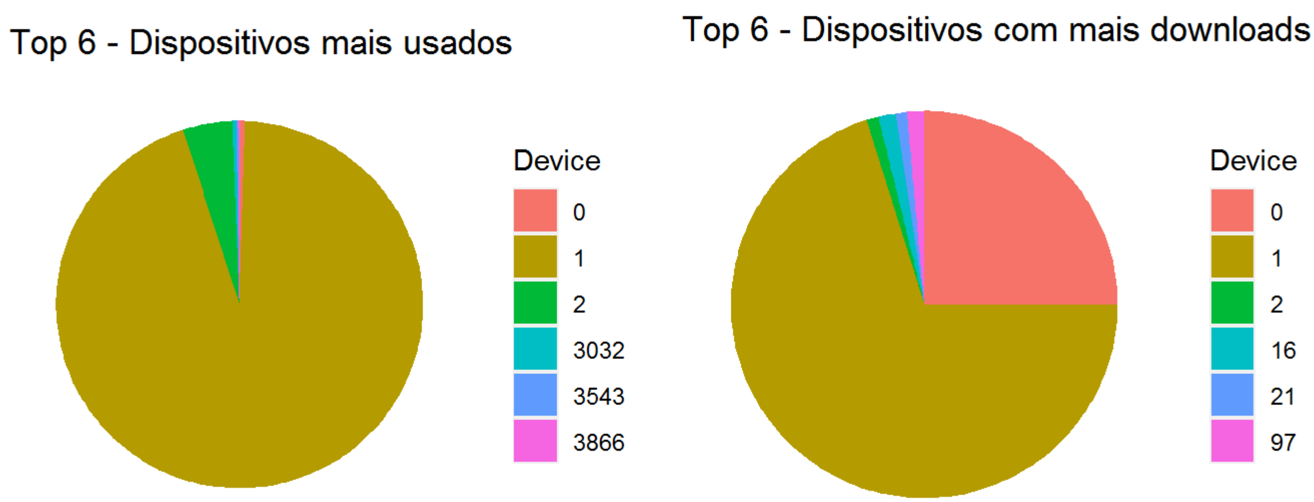
```
pl8 <- train2 %>%
   select(is_attributed, device)%>%
   filter(is_attributed == 1) %>%
   group_by(device)%>%
   summarise(Quantidade = table(device))%>%
   filter(Quantidade > 1)%>%
   ggplot(aes(x = '', y = Quantidade, fill = device)) +
   geom_bar(width = 1, stat = "identity") +
   coord_polar("y", start = 0, direction = -1) +
   labs(title = 'Top 6 - Dispositivos com mais downloads',
        fill = 'Device')+
   theme(
      axis.title.x = element_blank(),
      axis.title.y = element_blank(),
      panel.border = element_blank(),
      panel.grid=element_blank(),
      axis.ticks = element_blank(),
      panel.background = element_blank(),
      axis.text.x=element_blank())
```

```
## `summarise()` regrouping output by 'device' (override with `.groups` argument)
```

```
grid.arrange(pl7,pl8, nrow=1,ncol=2)
```

```
## Don't know how to automatically pick scale for object of type table. Defaulting to continuou
s.
```

```
## Don't know how to automatically pick scale for object of type table. Defaulting to continuou
s.
```




```
# O dispositvo 1 por ter mais acessos consequentemente tem mais downloads, e o dispotivo 0
# apesar de ter menos acesso, tem uma quantidade de download interessante.
```

```
# Canais mais usados / canais com mais downloads.
pl9 <- train2 %>%
   select(is_attributed, channel)%>%
   filter(is_attributed == 0) %>%
   group_by(channel)%>%
   summarise(Quantidade = table(channel))%>%
   filter(Quantidade > 2400)%>%
   ggplot(aes(x =reorder(channel, Quantidade), y =  Quantidade))+
   geom_bar(stat = "identity",color = "white", fill = "lightblue")+
   labs(title = 'Top 10 - Canais com mais acessos.',
        x = 'Canal', y = 'Quantidade de acessos')
```

```
## `summarise()` regrouping output by 'channel' (override with `.groups` argument)
```

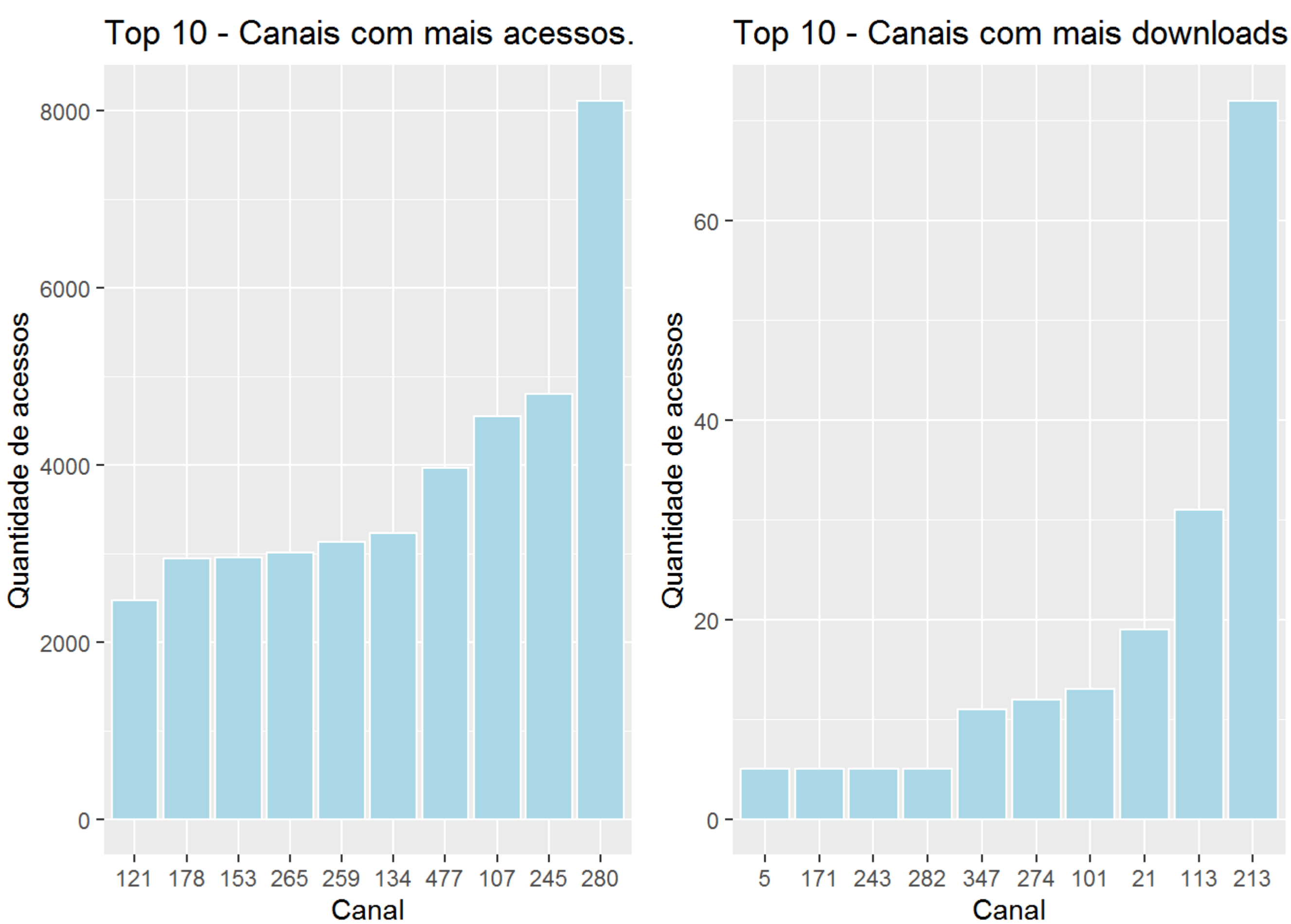
```
pl10 <- train2 %>%
   select(is_attributed, channel)%>%
   filter(is_attributed == 1) %>%
   group_by(channel)%>%
   summarise(Quantidade = table(channel))%>%
   filter(Quantidade > 4)%>%
   ggplot(aes(x =reorder(channel, Quantidade), y =  Quantidade))+
   geom_bar(stat = "identity",color = "white", fill = "lightblue")+
   labs(title = 'Top 10 - Canais com mais downloads.',
        x = 'Canal', y = 'Quantidade de acessos')
```

```
## `summarise()` regrouping output by 'channel' (override with `.groups` argument)
```

```
grid.arrange(pl9,pl10, nrow=1,ncol=2)
```

```
## Don't know how to automatically pick scale for object of type table. Defaulting to continuou
s.
```

```
## Don't know how to automatically pick scale for object of type table. Defaulting to continuou
s.
```



```
# Nenhum dos canais do top 10 mais usados estão no top 10 do canais com mais downloads.
# destacando o canal 213 com mais downloads.
```

```
# O.S mais usados / O.S com mais downloads.
pl11 <- train2 %>%
   select(is_attributed, os)%>%
   filter(is_attributed == 0) %>%
   group_by(os)%>%
   summarise(Quantidade = table(os))%>%
   filter(Quantidade > 2340)%>%
   ggplot(aes(x =reorder(os, Quantidade), y =  Quantidade))+
   geom_bar(stat = "identity",color = "white", fill = "lightblue")+
   labs(title = 'Top 10 - O.S com mais acessos.',
        x = 'O.S', y = 'Quantidade de acessos')
```

```
## `summarise()` regrouping output by 'os' (override with `.groups` argument)
```

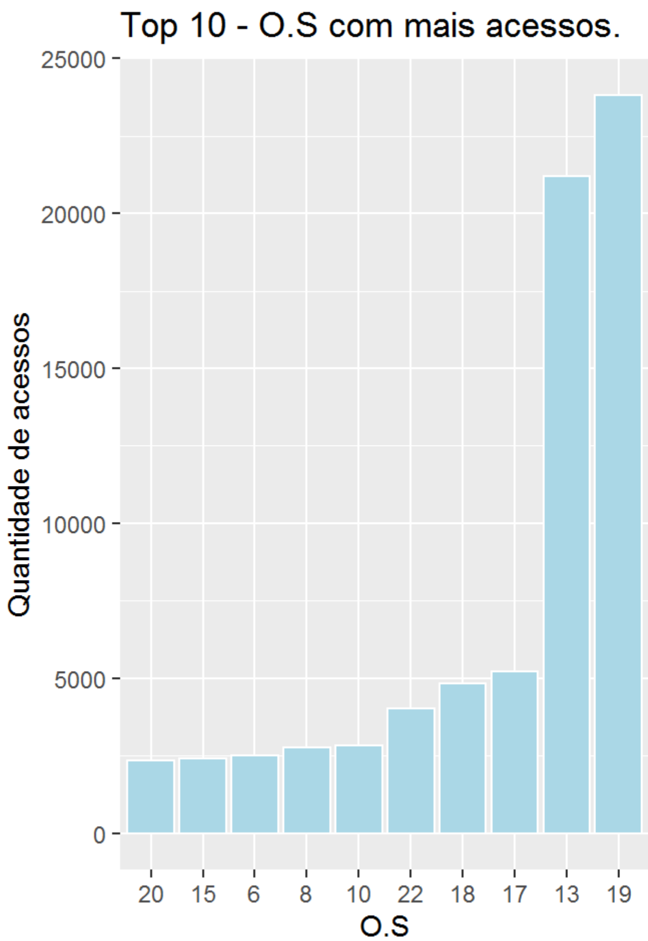
```
pl12 <- train2 %>%
   select(is_attributed, os)%>%
   filter(is_attributed == 1) %>%
   group_by(os)%>%
   summarise(Quantidade = table(os))%>%
   filter(Quantidade > 4)%>%
   ggplot(aes(x =reorder(os, Quantidade), y =  Quantidade))+
   geom_bar(stat = "identity",color = "white", fill = "lightblue")+
   labs(title = 'Top 10 - O.S com mais downloads.',
        x = 'O.S', y = 'Quantidade de acessos')
```

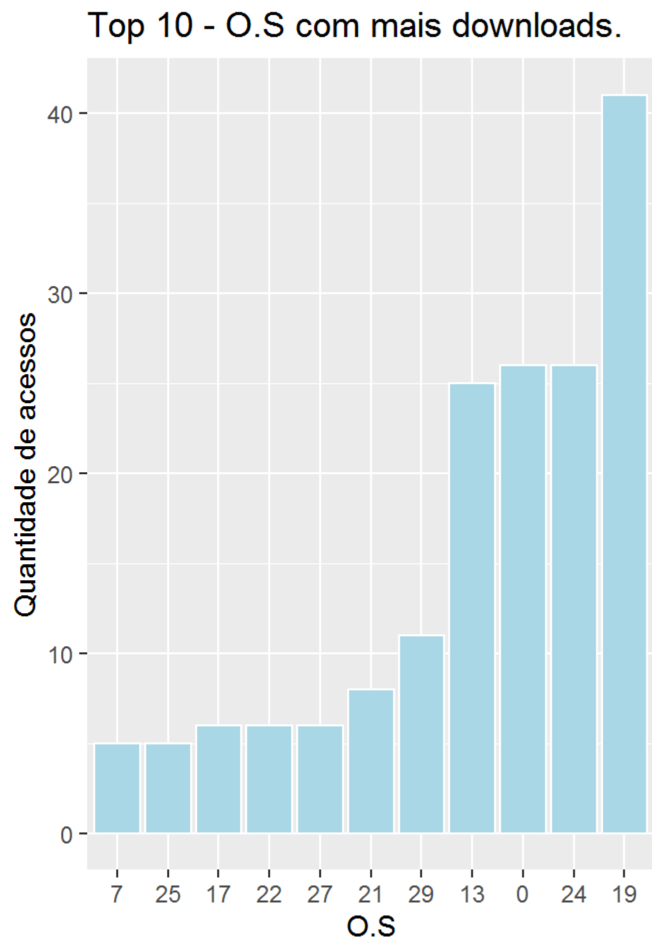
```
## `summarise()` regrouping output by 'os' (override with `.groups` argument)
```

```
grid.arrange(pl11,pl12, nrow=1,ncol=2)
```

```
## Don't know how to automatically pick scale for object of type table. Defaulting to continuou
s.
```

```
## Don't know how to automatically pick scale for object of type table. Defaulting to continuou
s.
```






```
# Acessos mais frequentes pelo O.S : 13, 19 ambos estando entre os primeiros no que mais fazem
# downloads, juntamente com o 0 e 24.
```

```
# App mais usados / App com mais downloads.
pl13 <- train2 %>%
   select(is_attributed, app)%>%
   filter(is_attributed == 0) %>%
   group_by(app)%>%
   summarise(Quantidade = table(app))%>%
   filter(Quantidade > 1999)%>%
   ggplot(aes(x =reorder(app, Quantidade), y =  Quantidade))+
   geom_bar(stat = "identity",color = "white", fill = "lightblue")+
   labs(title = 'Top 10 - App com mais acessos',
        x = 'App', y = 'Quantidade de acessos')
```

```
## `summarise()` regrouping output by 'app' (override with `.groups` argument)
```

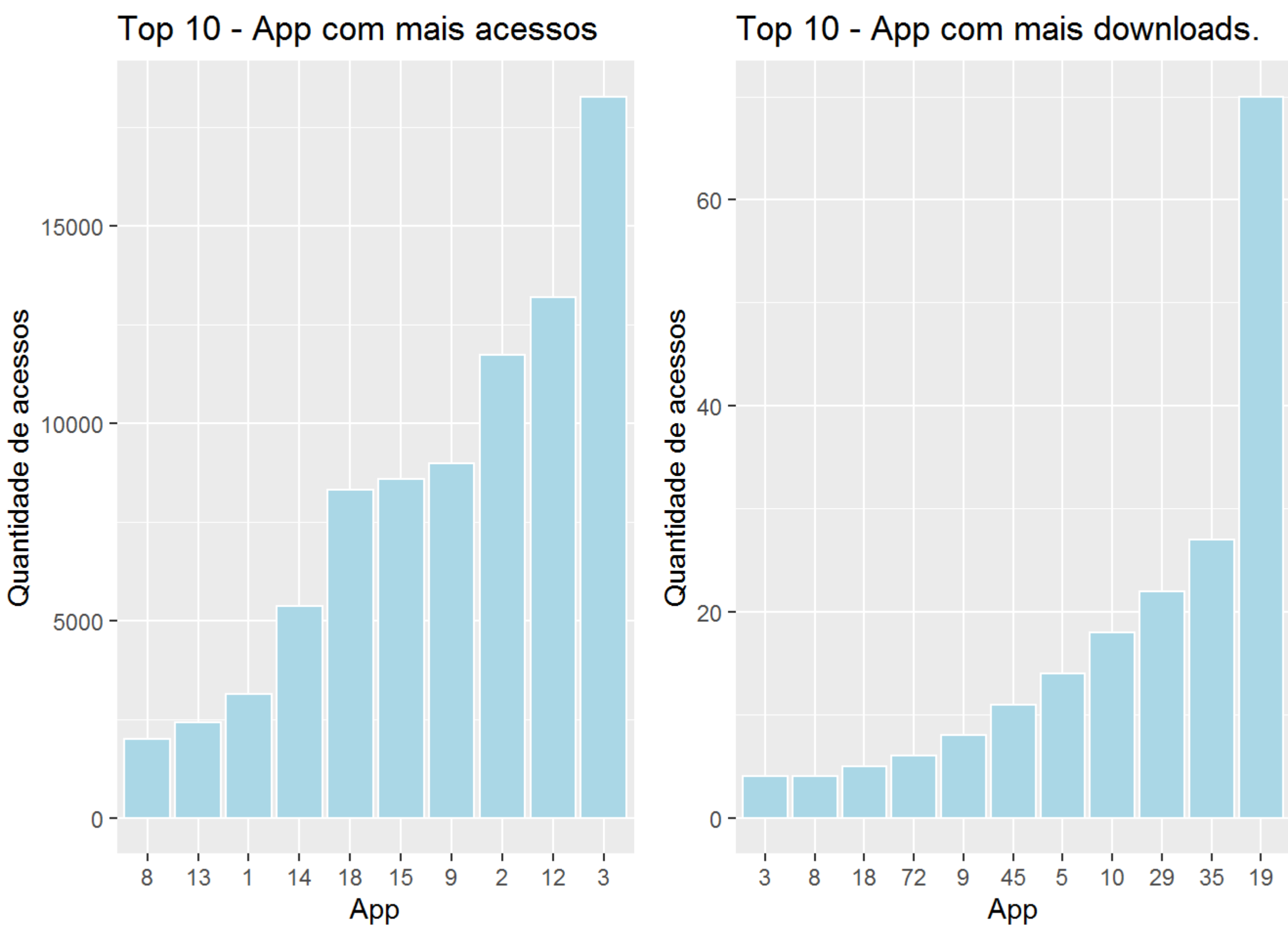
```
pl14 <- train2 %>%
   select(is_attributed, app)%>%
   filter(is_attributed == 1) %>%
   group_by(app)%>%
   summarise(Quantidade = table(app))%>%
   filter(Quantidade > 3)%>%
   ggplot(aes(x =reorder(app, Quantidade), y =  Quantidade))+
   geom_bar(stat = "identity",color = "white", fill = "lightblue")+
   labs(title = 'Top 10 - App com mais downloads.',
        x = 'App', y = 'Quantidade de acessos')
```

```
## `summarise()` regrouping output by 'app' (override with `.groups` argument)
```

```
grid.arrange(pl13,pl14, nrow=1,ncol=2)
```

```
## Don't know how to automatically pick scale for object of type table. Defaulting to continuou
s.
```

```
## Don't know how to automatically pick scale for object of type table. Defaulting to continuou
s.
```

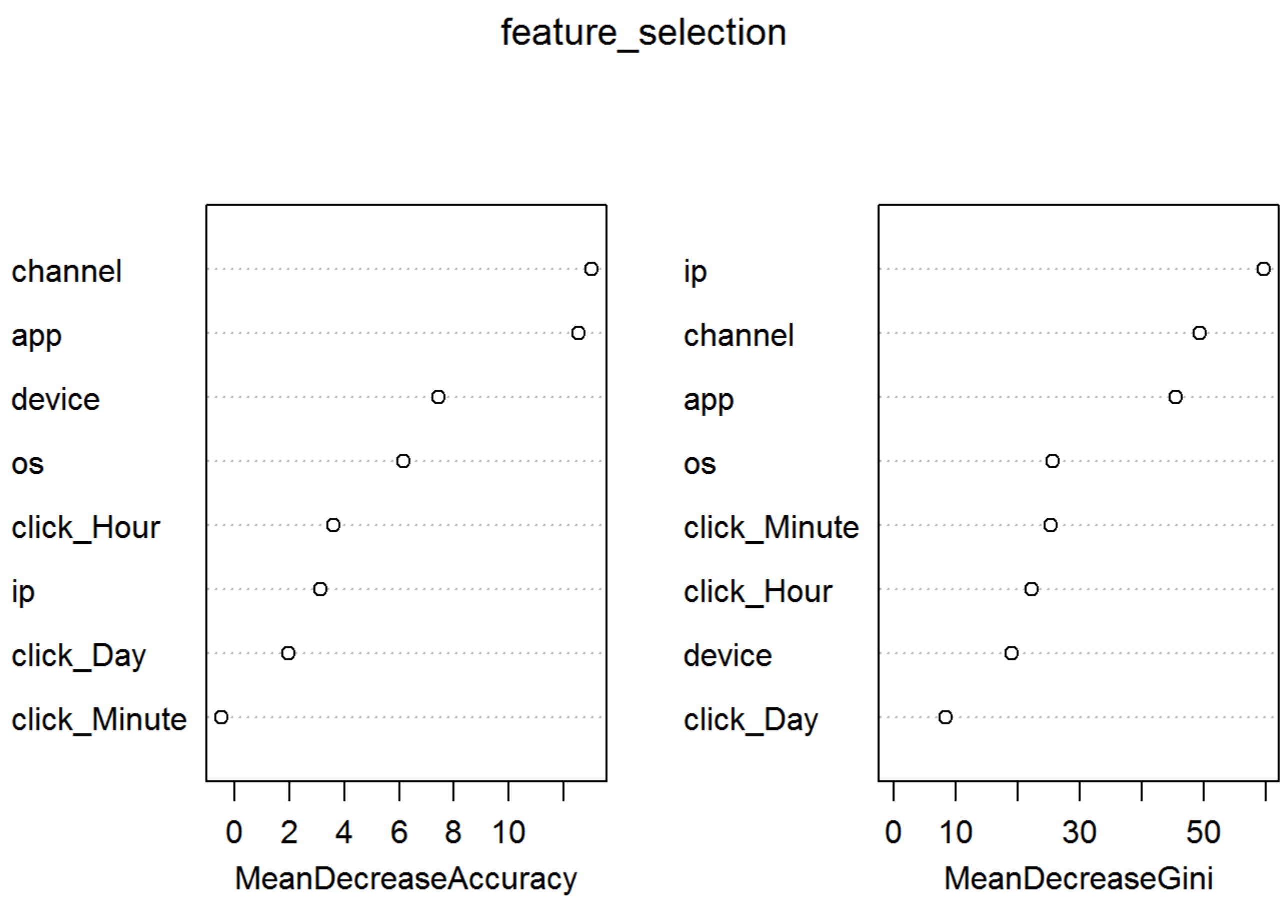


```
# Os apps com maiores acessos não aparecem entre os com mais downloads, este tendo com
# mais volume o app 19 com uma  quantidade de download maior em relação aos outros.
```

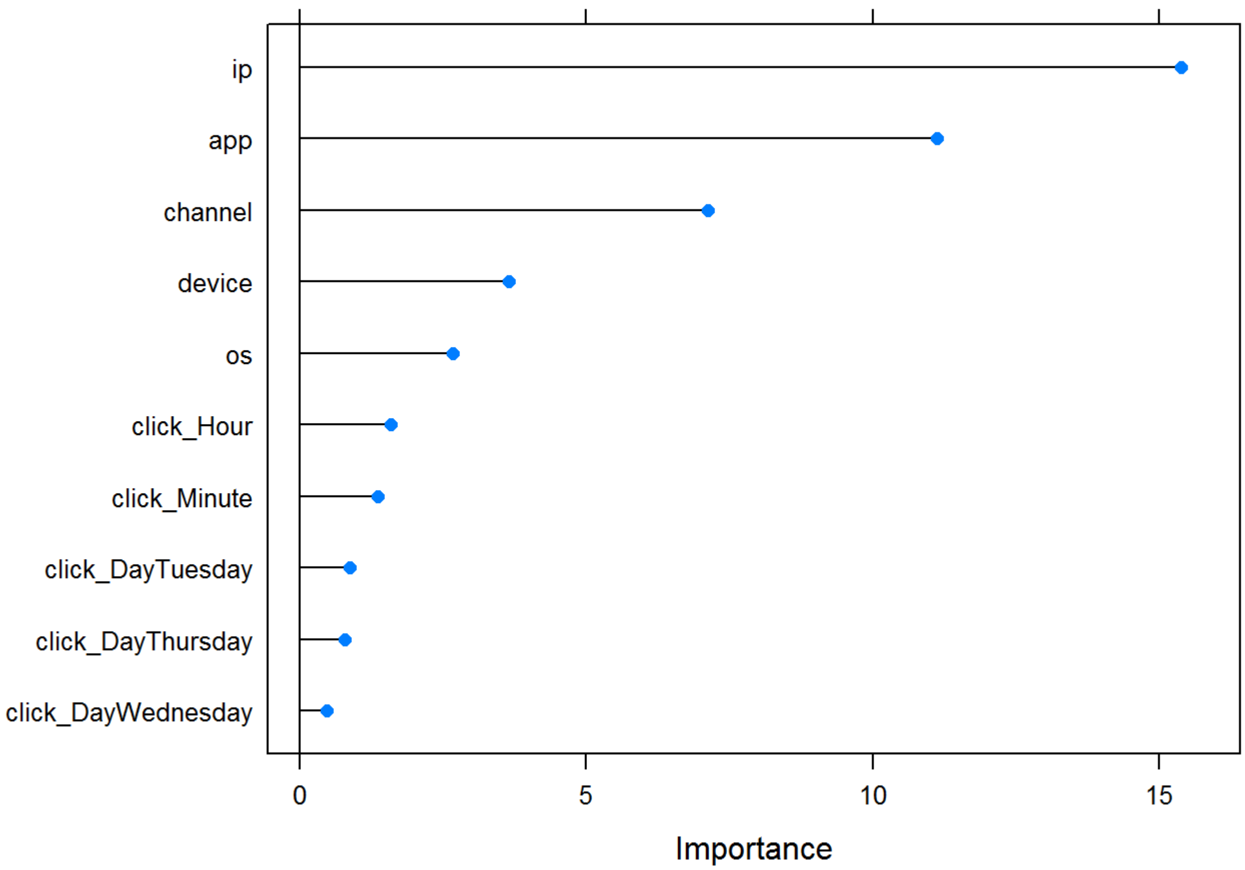
Feature Selection (Seleção de Variáveis)

```
# Após a analise exploratória dos dados, uso o random forest e o  glm para a seleção das variáve
is
# para treinar os modelos.
train$is_attributed <- as.factor(train$is_attributed)
train$click_Day <-as.factor(train$click_Day)
```

```
#Random Forest
feature_selection <- randomForest(is_attributed ~ .,
                                  data = train,
                                  ntree = 100, nodesize = 10, importance = T)

varImpPlot(feature_selection)
```



```
#GLM
control <- trainControl(method = "repeatedcv", number = 10, repeats = 2)
model <- train(is_attributed ~ . , data = train, method = "glm", trControl = control)
importance <- varImp(model, scale = FALSE)
plot(importance)
```




```
# Ambos os modelos mostraram as variáveis (ip, app, channel, device, os), como as mais relevante
s
# usarei elas para os modelos preditivos.
```

Split dos dados

```
# Faço a divisão do dados de treino e teste, usando o dataset train, e deixo o test que vou usar
# no final com o modelo de melhor performance.
intrain <- createDataPartition(train2$os,p=0.7,list=FALSE)
```

```
## Warning in createDataPartition(train2$os, p = 0.7, list = FALSE): Some classes
## have a single record ( 84, 88, 99, 106, 113, 114, 116, 127, 129, 133, 135, 137,
## 142, 151, 153, 168, 172, 174, 185, 192, 193, 199, 207, 836 ) and these will be
## selected for the sample
```

```
trainModel <- train2[intrain,]
testModel <- train2[-intrain,]
nrow(trainModel)
```

```
## [1] 70063
```

```
nrow(testModel)
```

```
## [1] 29937
```

```
table(trainModel$is_attributed)
```

```
## 
##     0     1 
## 69901   162
```

```
str(trainModel)
```

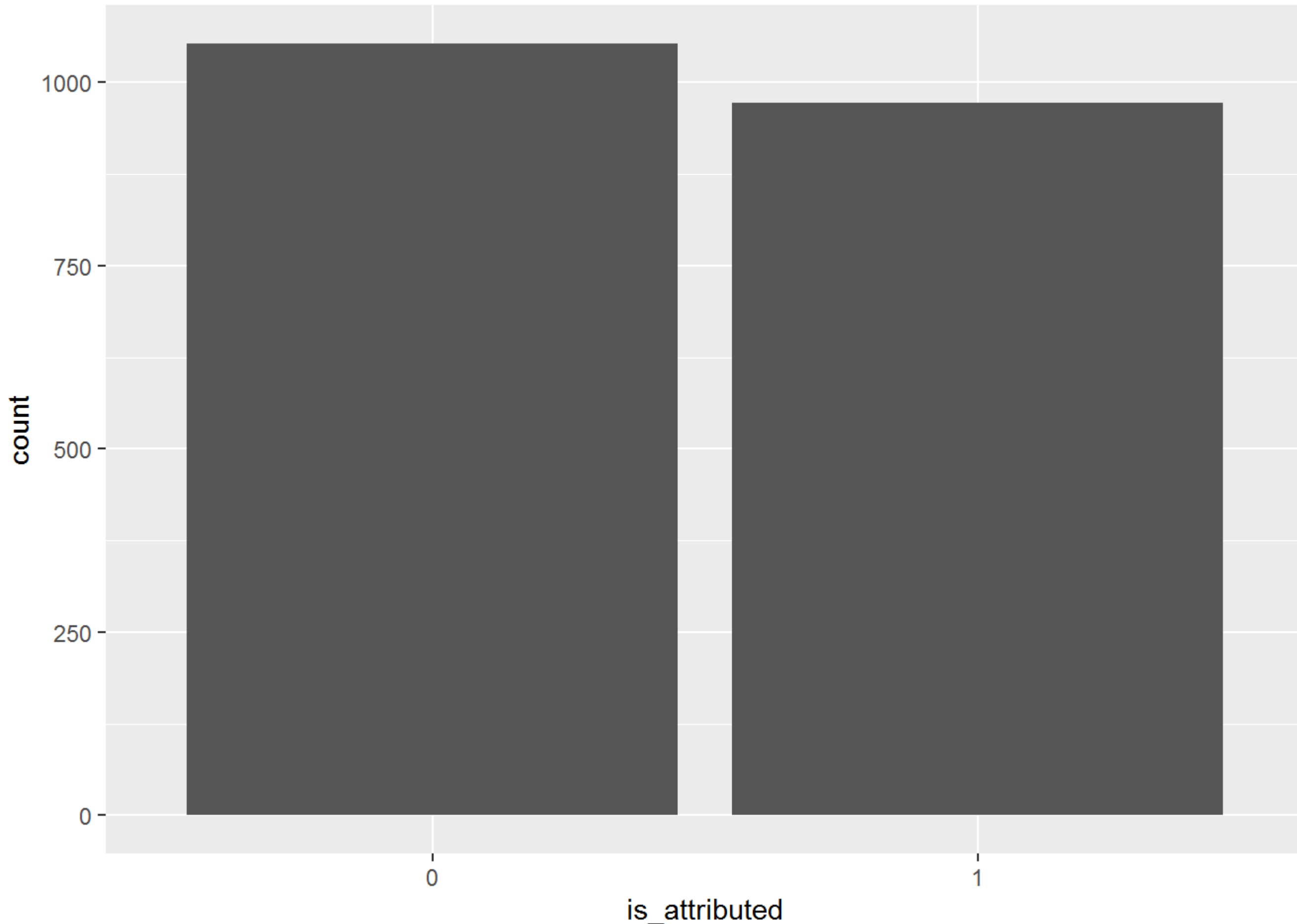
```
## 'data.frame':    70063 obs. of  9 variables:
##  $ ip           : Factor w/ 34857 levels "9","10","19",..: 18449 17664 11853 21304 12885 6028
20015 6263 12572 919 ...
##  $ app          : Factor w/ 161 levels "1","2","3","4",..: 25 12 12 9 3 3 6 2 25 2 ...
##  $ device       : Factor w/ 100 levels "0","1","2","4",..: 2 2 2 2 2 2 2 2 3 2 ...
##  $ os           : Factor w/ 130 levels "0","1","2","3",..: 18 20 2 26 23 14 21 14 20 3 ...
##  $ channel      : Factor w/ 161 levels "3","4","5","13",..: 68 53 46 127 157 157 29 49 68 147
...
##  $ is_attributed: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ click_Day    : Factor w/ 4 levels "Monday","Thursday",..: 3 3 2 3 4 3 4 3 4 2 ...
##  $ click_Hour   : Factor w/ 24 levels "0","1","2","3",..: 14 19 10 11 9 6 15 1 24 8 ...
##  $ click_Minute : Factor w/ 60 levels "0","1","2","3",..: 41 6 1 2 15 4 47 55 18 34 ...
```

Balanceamento dos dados

```
# É necessário balancear a variável target, pois está muito desbalanceada e com isso pode
# fazer com que o algoritmo faça previsões equivocadas, então uso o pacote SMOTE
# para o balanceamento.
trainModel_balanced <- SMOTE(is_attributed ~ .,trainModel, perc.over =500, perc.under=130)
table(trainModel_balanced$is_attributed)
```

```
## 
##    0    1 
## 1053  972
```

```
ggplot(trainModel_balanced, aes(x = is_attributed)) + geom_bar()
```




```
nrow(trainModel_balanced)
```

```
## [1] 2025
```

```
head(trainModel_balanced)
```

```
##           ip app device os channel is_attributed click_Day click_Hour
## 70714  81211   3      1 15     280             0 Wednesday          0
## 30777 112574  15      1 19     278             0 Wednesday          8
## 301    68568  18      1 22     107             0  Thursday          7
## 67119  43871   2      1 19     219             0 Wednesday         15
## 71269  79271   9      1 17     442             0 Wednesday         10
## 93450  48062   3      1 10     280             0 Wednesday          2
##       click_Minute
## 70714           45
## 30777           36
## 301             59
## 67119           31
## 71269            2
## 93450           34
```

```
# Converto todas as variáveis para chacacter deixando apenas a target como factor para
# usar em alguns algoritmos em outros usarei o com as variaveis factor.
var_convert2 <- c ("ip","app","device","os","channel",
                  "click_Day","click_Hour","click_Minute")

to_character <- function(df, var){
   for (i in var){
      df[[i]] <- as.character(df[[i]])
   }
   return(df)
}

trainModel_balanced2 <- to_character(trainModel_balanced,var_convert2)
testModel2 <- to_character(testModel,var_convert2)

glimpse(trainModel_balanced2)
```

```
## Rows: 2,025
## Columns: 9
## $ ip            <chr> "81211", "112574", "68568", "43871", "79271", "48062"...
## $ app           <chr> "3", "15", "18", "2", "9", "3", "3", "15", "3", "2", ...
## $ device        <chr> "1", "1", "1", "1", "1", "1", "1", "1", "1", "1", "1"...
## $ os            <chr> "15", "19", "22", "19", "17", "10", "17", "19", "18",...
## $ channel       <chr> "280", "278", "107", "219", "442", "280", "135", "245...
## $ is_attributed <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ click_Day     <chr> "Wednesday", "Wednesday", "Thursday", "Wednesday", "W...
## $ click_Hour    <chr> "0", "8", "7", "15", "10", "2", "4", "13", "14", "0",...
## $ click_Minute  <chr> "45", "36", "59", "31", "2", "34", "45", "29", "37", ...
```

```
glimpse(testModel2)
```

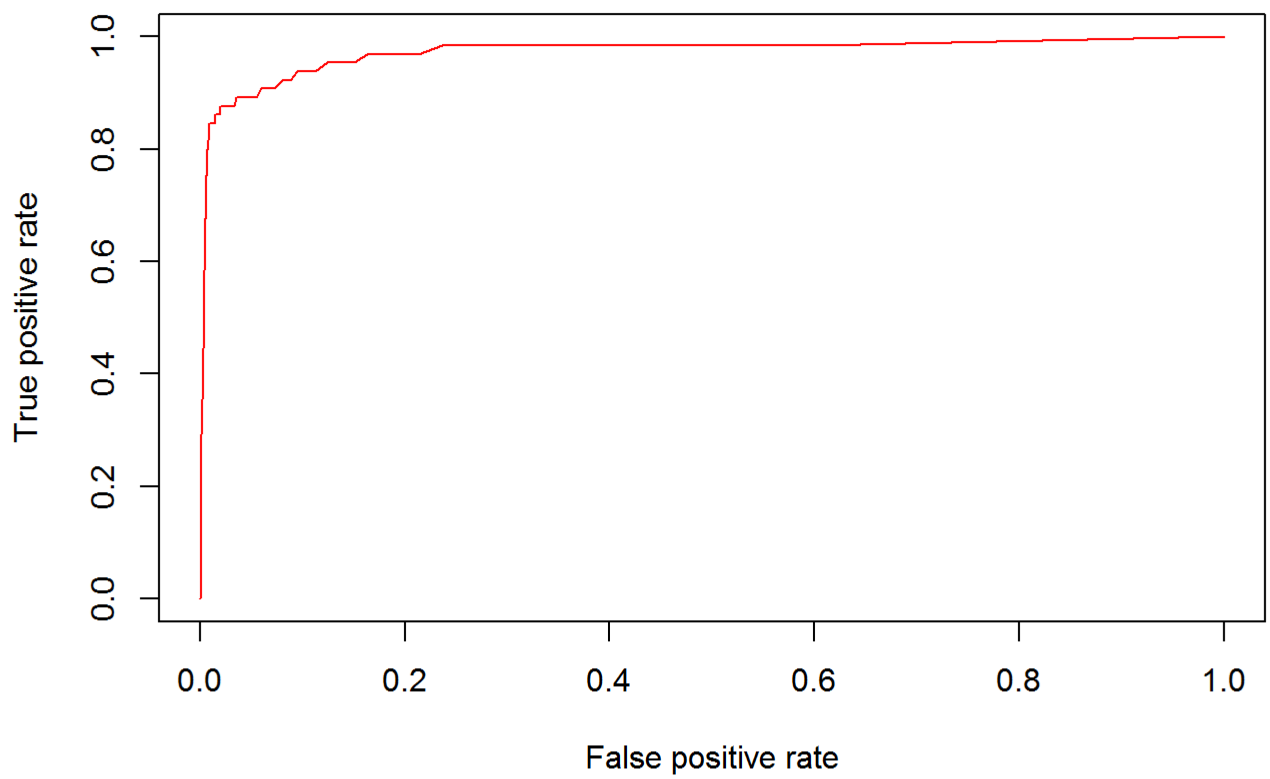
```
## Rows: 29,937
## Columns: 9
## $ ip            <chr> "87540", "94584", "93663", "17059", "192967", "143636...
## $ app           <chr> "12", "13", "3", "1", "2", "3", "3", "3", "64", "7", ...
## $ device        <chr> "1", "1", "1", "1", "2", "1", "1", "1", "1", "1", "1"...
## $ os            <chr> "13", "13", "17", "17", "22", "19", "22", "20", "25",...
## $ channel       <chr> "497", "477", "115", "135", "364", "135", "205", "280...
## $ is_attributed <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ click_Day     <chr> "Tuesday", "Tuesday", "Thursday", "Thursday", "Wednes...
## $ click_Hour    <chr> "9", "4", "1", "1", "9", "12", "10", "3", "17", "10",...
## $ click_Minute  <chr> "30", "58", "22", "17", "35", "35", "24", "44", "1", ...
```

Algoritmos de aprendizagem

```r
# Modelo com o randomForest
modelo_v1 <- randomForest(is_attributed ~ ip
                          +app
                          +channel
                          +device
                          +os,
                          data = trainModel_balanced2,
                          ntree = 100,
                          nodesize = 10)

previsao_v1 <- predict(modelo_v1, testModel2)
confusionMatrix(previsao_v1, testModel2$is_attributed)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0 29373     9
##          1   499    56
##
##                Accuracy : 0.983
##                  95% CI : (0.9815, 0.9845)
##     No Information Rate : 0.9978
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1774
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9833
##             Specificity : 0.8615
##          Pos Pred Value : 0.9997
##          Neg Pred Value : 0.1009
##              Prevalence : 0.9978
##          Detection Rate : 0.9812
##    Detection Prevalence : 0.9815
##       Balanced Accuracy : 0.9224
##
##        'Positive' Class : 0
##
```

```r
# Criando curvas ROC para o modelo
previsao_v1_ROC <- predict(modelo_v1, newdata = testModel2, type = 'prob')
targetROC2 <- testModel2$is_attributed
pred1 <- prediction(previsao_v1_ROC[,2], targetROC2)
perf1 <- performance(pred1, "tpr","fpr")
plot(perf1, col = rainbow(10))
```




```
# Modelo com o naiveBayes
modelo_v2 <- naiveBayes(is_attributed ~ ip
                        +app
                        +channel
                        +device
                        +os,
                        data = trainModel_balanced2)

previsao_v2 <- predict(modelo_v2, testModel2)
```

```
## Warning in data.matrix(newdata): NAs introduced by coercion
```

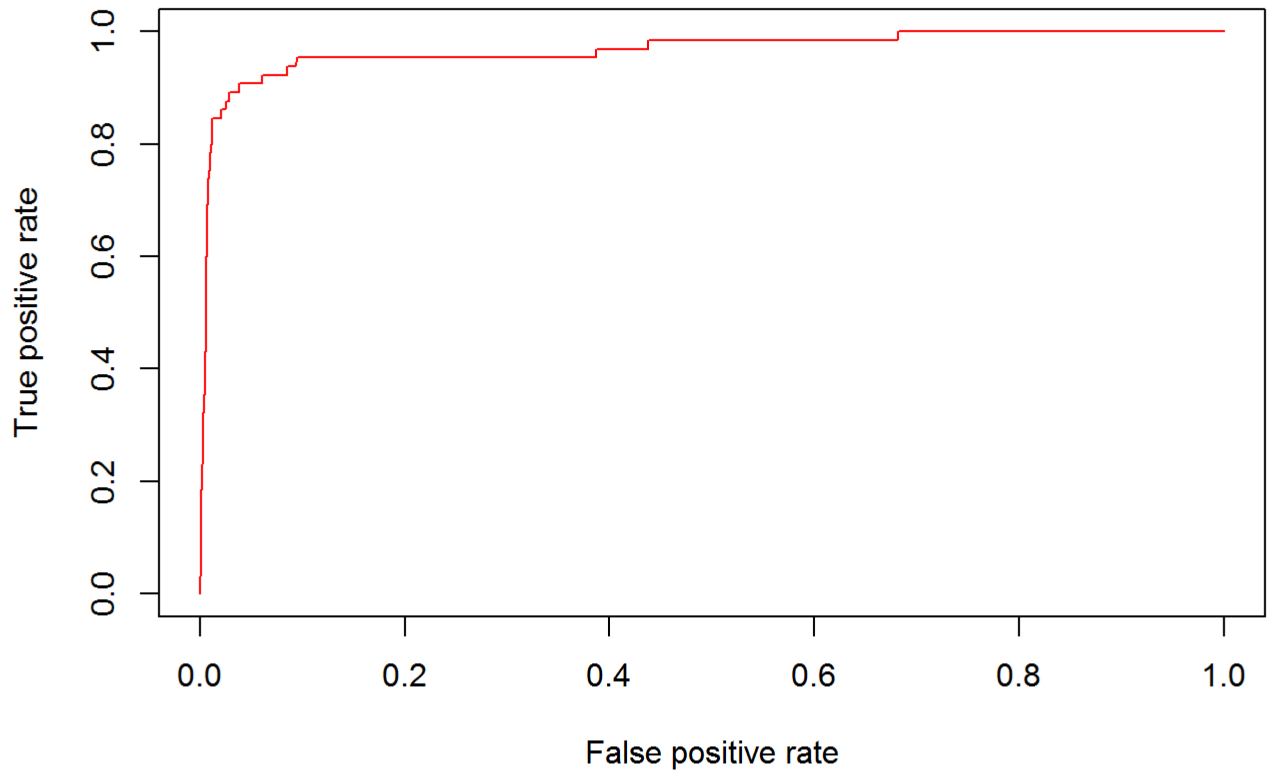
```
confusionMatrix(previsao_v2, testModel2$is_attributed)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction     0     1
##          0 29276    10
##          1   596    55
##                                           
##                Accuracy : 0.9798          
##                  95% CI : (0.9781, 0.9813)
##     No Information Rate : 0.9978          
##     P-Value [Acc > NIR] : 1               
##                                           
##                   Kappa : 0.1503          
##                                           
##  Mcnemar's Test P-Value : <2e-16          
##                                           
##             Sensitivity : 0.98005         
##             Specificity : 0.84615         
##          Pos Pred Value : 0.99966         
##          Neg Pred Value : 0.08449         
##              Prevalence : 0.99783         
##          Detection Rate : 0.97792         
##    Detection Prevalence : 0.97825         
##       Balanced Accuracy : 0.91310         
##                                           
##        'Positive' Class : 0               
## 
```

```
# Criando curvas ROC para o modelo
previsao_v2_ROC <- predict(modelo_v2, newdata = testModel2, type = 'raw')
```

```
## Warning in data.matrix(newdata): NAs introduced by coercion
```

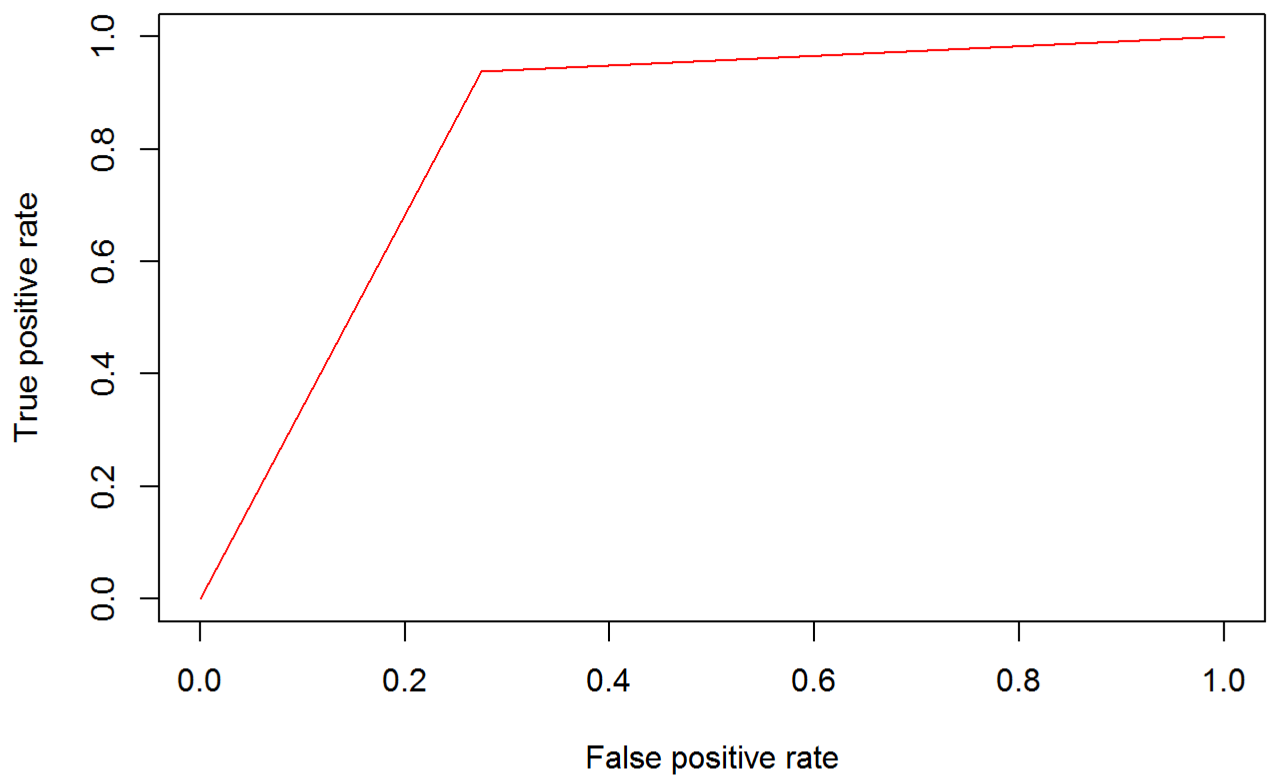
```
pred2 <- prediction(previsao_v2_ROC[,2], targetROC2)
perf2 <- performance(pred2, "tpr","fpr")
plot(perf2, col = rainbow(10))
```




```r
# Modelo com o C5.0
Cost_func <- matrix(c(0, 2, 1.5, 0), nrow = 2, dimnames = list(c("0", "1"), c("0", "1")))

modelo_v3 <- C5.0(is_attributed ~ ip
                  +app
                  +channel
                  +device
                  +os,
                  data = trainModel_balanced,
                  trials = 100,
                  cost = Cost_func)

previsao_v3 <- predict(modelo_v3, testModel)
confusionMatrix(previsao_v3, testModel$is_attributed)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction     0     1
##          0 21686     4
##          1  8186    61
##                                           
##                Accuracy : 0.7264          
##                  95% CI : (0.7213, 0.7315)
##     No Information Rate : 0.9978          
##     P-Value [Acc > NIR] : 1               
##                                           
##                   Kappa : 0.0104          
##                                           
##  Mcnemar's Test P-Value : <2e-16          
##                                           
##             Sensitivity : 0.725964        
##             Specificity : 0.938462        
##          Pos Pred Value : 0.999816        
##          Neg Pred Value : 0.007397        
##              Prevalence : 0.997829        
##          Detection Rate : 0.724388        
##    Detection Prevalence : 0.724521        
##       Balanced Accuracy : 0.832213        
##                                           
##        'Positive' Class : 0               
## 
```

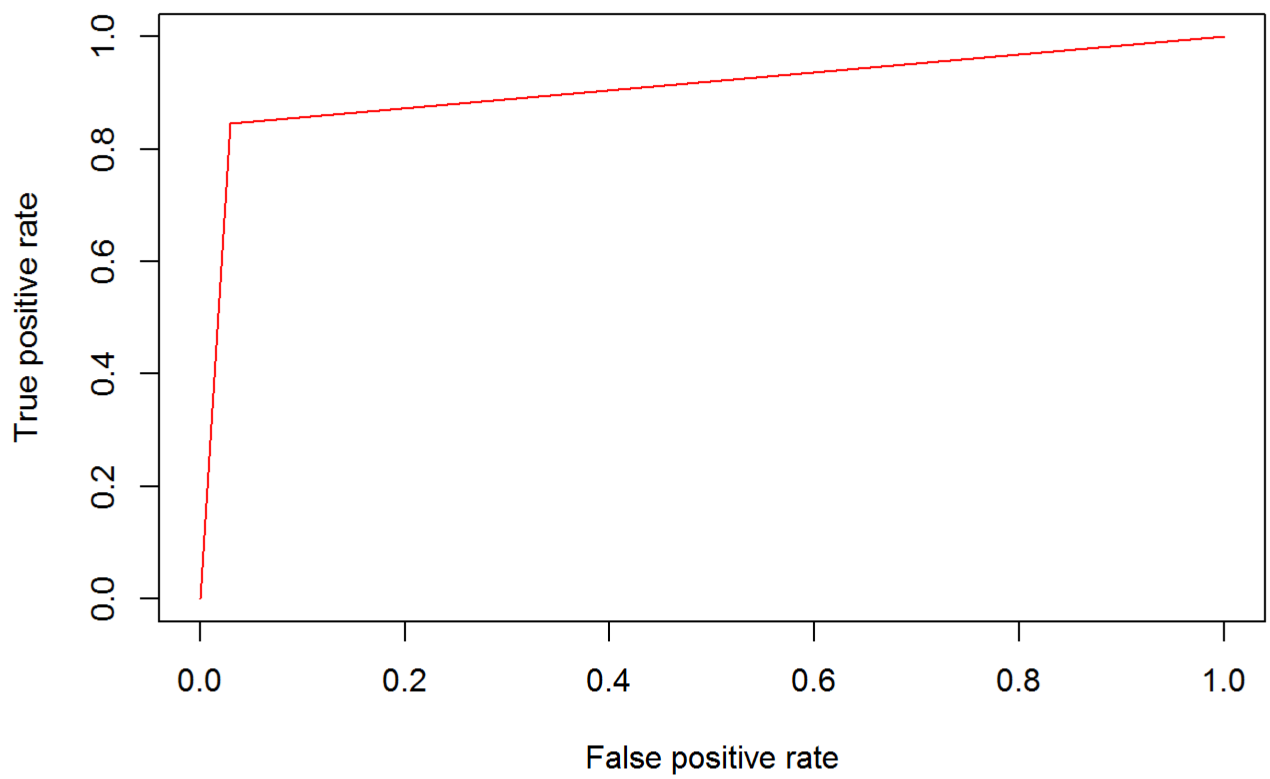
```
# Criando curvas ROC para o modelo
targetROC <- testModel$is_attributed
previsao_v3_ROC <- predict(modelo_v3, newdata = testModel,type = 'class')
pred3 <- prediction(as.numeric(previsao_v3_ROC), as.numeric(targetROC))
perf3 <- performance(pred3, "tpr","fpr")
plot(perf3, col = rainbow(10))
```




```r
# Modelo com o rpart
modelo_v4 <- rpart(is_attributed ~ ip
                   +app
                   +channel
                   +device
                   +os,
                   data = trainModel_balanced)

previsao_v4 <- predict(modelo_v4, testModel,type = 'class')
confusionMatrix(previsao_v4, testModel$is_attributed)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0 28981    10
##          1   891    55
##
##                Accuracy : 0.9699
##                  95% CI : (0.9679, 0.9718)
##     No Information Rate : 0.9978
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1052
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.97017
##             Specificity : 0.84615
##          Pos Pred Value : 0.99966
##          Neg Pred Value : 0.05814
##              Prevalence : 0.99783
##          Detection Rate : 0.96807
##    Detection Prevalence : 0.96840
##       Balanced Accuracy : 0.90816
##
##        'Positive' Class : 0
##
```

```
# Criando curvas ROC para o modelo
previsao_v4_ROC <- predict(modelo_v4, newdata = testModel,type = 'prob')
pred4 <- prediction(previsao_v4_ROC[,2], targetROC)
perf4 <- performance(pred4, "tpr","fpr")
plot(perf4, col = rainbow(10))
```



Rodando o algoritmo em produção

```
# Todos os 4 modelos tiveram resultados satisfatórios, eu decido por usar o randomForest em prod
ução.

# Carrego o data set de test simulando como se fossem dados novos e faço os tratamentos para rod
ar no algoritmo e adiciono
# o resultado previsto ao dataset e imprimo as primeiras e ultimas linhas.
test_default <- fread('Dados/test_default.csv', data.table = FALSE)
test_default$click_id <- NULL
test_default$click_time <- NULL
var_convert3 <- c ("ip","app","device","os","channel")
test<- to_character(test_default,var_convert3)

previsao_prod <- predict(modelo_v1, test)
table(previsao_prod)
```

```
## previsao_prod
##        0        1
## 18382442   408027
```

```
test$is_attributed <- previsao_prod
head(test)
```

```
##       ip app device os channel is_attributed
## 1   5744   9      1  3     107             0
## 2 119901   9      1  3     466             0
## 3  72287  21      1 19     128             0
## 4  78477  15      1 13     111             0
## 5 123080  12      1 13     328             0
## 6 110769  18      1 13     107             0
```

```
tail(test)
```

```
##              ip app device os channel is_attributed
## 18790464  69245  12      1 13     135             0
## 18790465  99442   9      1 13     127             0
## 18790466  88046  23      1 37     153             0
## 18790467  81398  18      1 17     265             0
## 18790468 123236  27      1 13     122             0
## 18790469  73516  12      2 27     265             0
```