Data Science Academy - Projeto com Feedback 01

Henrique Jordão Figueiredo Alves

07 Maio, 2021

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

Este projeto tem como objetivo criar um algoritmo de aprendizagem de máquina que possa prever se um usuário fará o download de um aplicativo depois de clicar em um anúncio de aplicativo para dispositivos móveis.

https://www.kaggle.com/c/talkingdata-adtracking-fraud-detection/data

Para este projeto foi utilizado uma amostragem menor dos dados disponíveis.

Ao testar o arquivo .R, favor modificar o diretório da pasta raiz do projeto para onde os arquivos do mesmo estão localizados em sua máquina.

Etapa 1: Carregando os dados

```
# Carregando os pacotes necessários para o projeto
library(tidyr)
library(lubridate)
```

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

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

```
library(ggplot2)
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
```

```
dados <- read.csv("train_sample.csv")
str(dados)
```

```
## '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 59
##  $ 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 ...
```

```
head(dados)
```

```
##       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
```

Etapa 2: Limpando os Dados

```
# Criando função para conversão de variáveis categóricas
to.factors <- function(df, variables){
  for (variable in variables){
    df[[variable]] <- as.factor(df[[variable]])
  }
  return(df)
}

dados <- dados[,-7]

# Separando Data e Hora
dados <- separate(dados, click_time, c("Date","Time"), sep = ' ', remove = TRUE)

# Variáveis do tipo fator
categorical.vars <- c('is_attributed', 'device', 'os', 'channel', 'app', 'ip')

# Convertendo as variáveis para o tipo fator (categórica)
dados <- to.factors(df = dados, variables = categorical.vars)
str(dados)
```

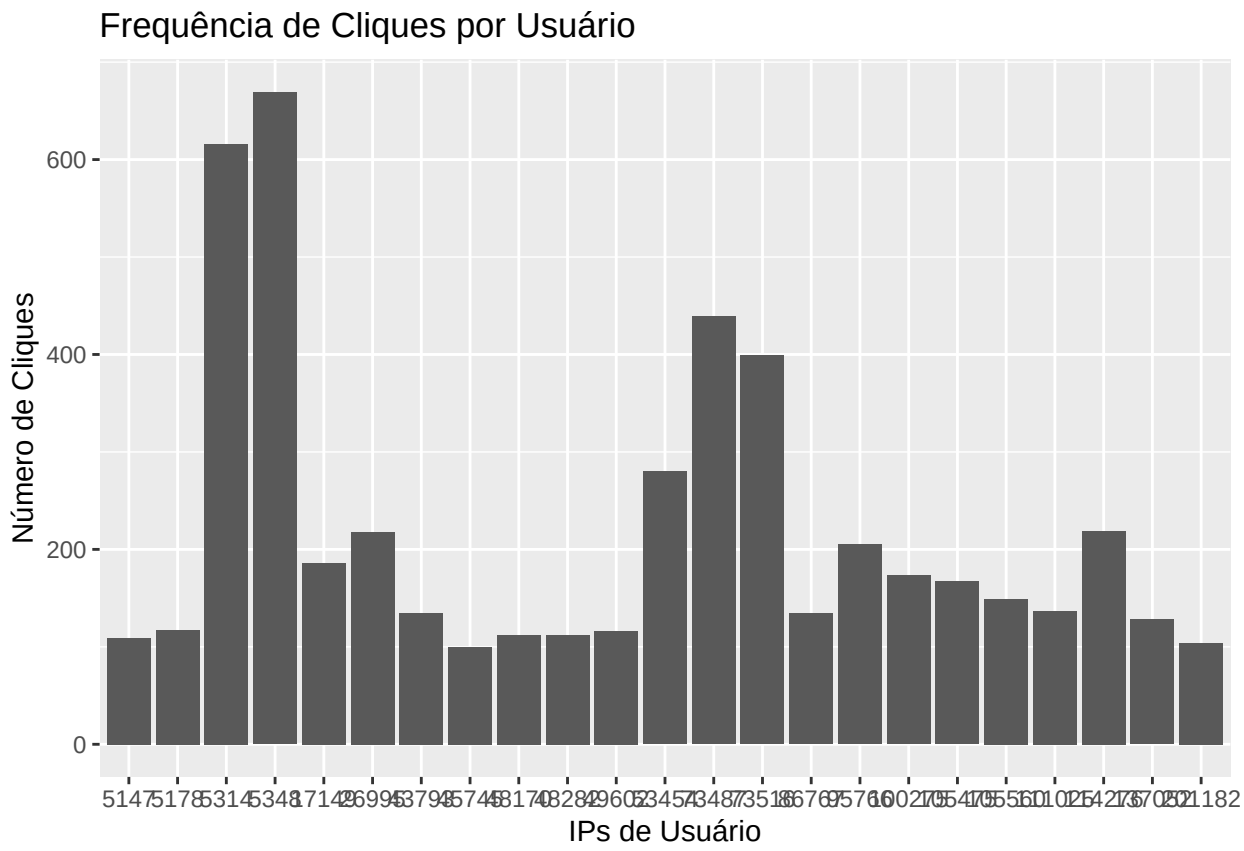
```
## 'data.frame':    100000 obs. of  8 variables:
##  $ ip           : Factor w/ 34857 levels "9","10","19",..: 15221 18449 17664 16497 11853 16301 2974 1
##  $ app          : Factor w/ 161 levels "1","2","3","4",..: 12 25 12 13 12 3 1 9 2 3 ...
##  $ device       : Factor w/ 100 levels "0","1","2","4",..: 2 2 2 2 2 2 2 2 3 2 ...
##  $ os           : Factor w/ 130 levels "0","1","2","3",..: 14 18 20 14 2 18 18 26 23 20 ...
##  $ channel      : Factor w/ 161 levels "3","4","5","13",..: 160 68 53 147 46 21 35 127 101 35 ...
##  $ Date         : chr  "2017-11-07" "2017-11-07" "2017-11-07" "2017-11-07" ...
##  $ Time         : chr  "09:30:38" "13:40:27" "18:05:24" "04:58:08" ...
##  $ is_attributed: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
```

Etapa 3: Análise Exploratória dos Dados

```
ip_freq <- as.data.frame(table(dados$ip))
names(ip_freq) <- c("ip","freq")
str(ip_freq)
```

```
## 'data.frame':    34857 obs. of  2 variables:
##  $ ip  : Factor w/ 34857 levels "9","10","19",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ freq: int  1 3 1 4 1 5 1 1 3 3 ...
```

```
# Plot frequência de cliques por IP
ip_freq %>% filter(freq >= 100) %>%
ggplot(aes(x =ip, y = freq)) + geom_bar(stat = "identity") +
  ylab("Número de Cliques") + xlab("IPs de Usuário") +
  ggtitle("Frequência de Cliques por Usuário")
```



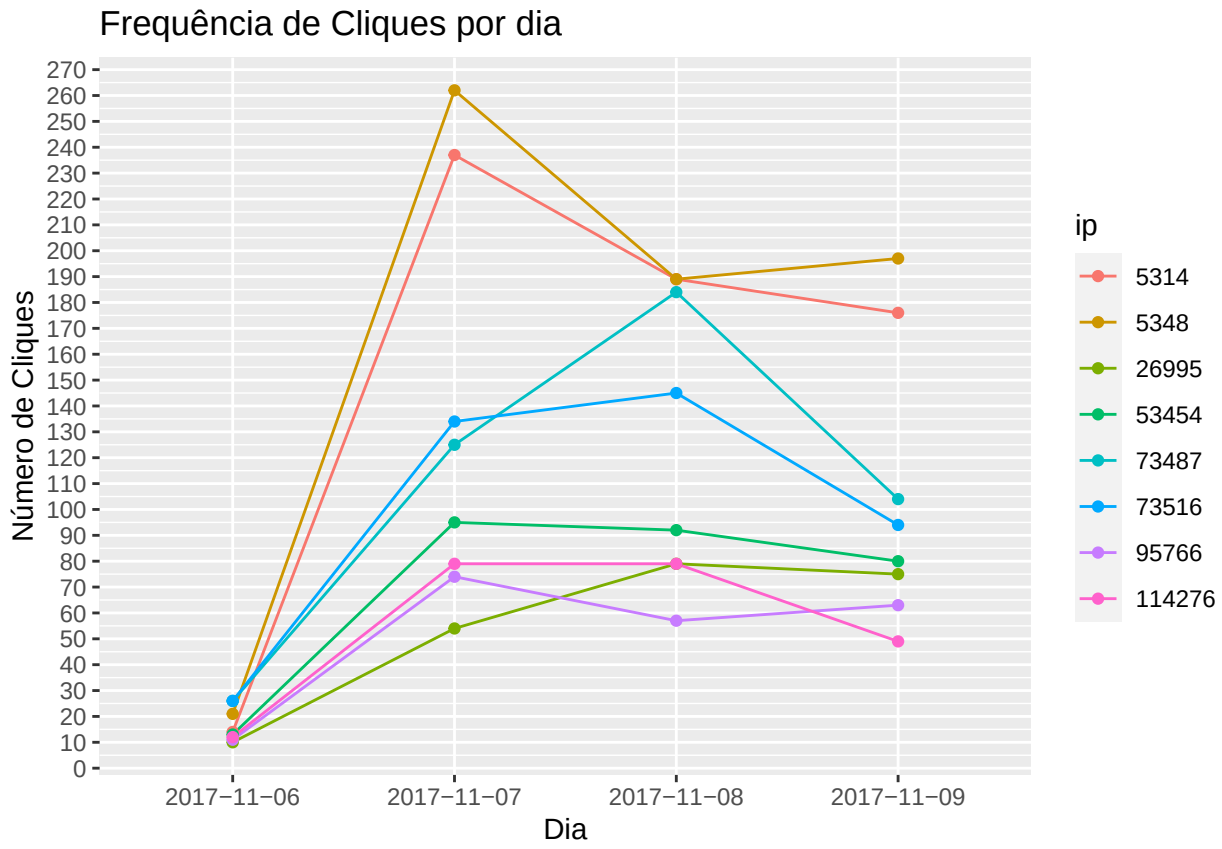

```
# Criando vetor de IPs com maior suspeita de fraude (mais de 200 cliques)
ip_susp <- ip_freq %>% filter(freq >= 200)
ip_susp <- ip_susp$ip

# Tabela de cliques por Ips suspeitos de acordo com a data
dados_bydate <- dados %>% group_by(ip, Date) %>% summarise(Freq = n())
```

```
## `summarise()` has grouped output by 'ip'. You can override using the `.groups` argument.
```

```
dados_bydate <- dados_bydate %>% filter(ip %in% ip_susp)

# Plot cliques de IPs suspeitos por data
dados_bydate %>%
  ggplot(aes(x = Date, y = Freq, group = ip, color = ip)) +
  ylab("Número de Cliques") + xlab("Dia") +
  geom_line() + geom_point() +
  scale_y_continuous(breaks = seq(0, 300, by = 10)) +
  ggtitle("Frequência de Cliques por dia")
```




```
# Teste Qui-Quadrado

chisq.test(dados$is_attributed, dados$ip)
```

```
## Warning in chisq.test(dados$is_attributed, dados$ip): Chi-squared approximation
## may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  dados$is_attributed and dados$ip
## X-squared = 74845, df = 34856, p-value < 2.2e-16
```

Etapa 4: Construindo Modelos Preditivos

```
# Excluindo Ips com menos de 50 cliques

ip_freq <- ip_freq %>% filter(freq >= 50)
dados <- dados %>% filter(ip %in% ip_freq$ip)
str(dados)
```

```
## 'data.frame':    8457 obs. of  8 variables:
##  $ ip           : Factor w/ 34857 levels "9","10","19",..: 18449 20015 6263 919 8450 671 13926 927 9
```

```
##  $ app          : Factor w/ 161 levels "1","2","3","4",..: 25 6 2 2 2 12 11 8 28 1 ...
##  $ device       : Factor w/ 100 levels "0","1","2","4",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ os           : Factor w/ 130 levels "0","1","2","3",..: 18 21 14 3 20 20 20 12 20 20 ...
##  $ channel      : Factor w/ 161 levels "3","4","5","13",..: 68 29 49 147 26 71 151 39 35 13 ...
##  $ Date         : chr  "2017-11-07" "2017-11-08" "2017-11-07" "2017-11-09" ...
##  $ Time         : chr  "13:40:27" "14:46:16" "00:54:09" "07:33:41" ...
##  $ is_attributed: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
```

```
table(dados$is_attributed)
```

```
##
##    0    1
## 8449    8
```

```
# Modelo RandomForest
set.seed(40)
dados[,'index'] <- ifelse(runif(nrow(dados)) < 0.7,1,0)

# Dados de treino e teste
trainset <- dados[dados$index==1,]
testset <- dados[dados$index==0,]

# Obter o índice
trainColNum <- grep('index', names(trainset))

# Remover o índice dos datasets
trainset <- trainset[,-trainColNum]
testset <- testset[,-trainColNum]

# Cria o modelo
library(rpart)
modelo_rf_v1 = rpart(is_attributed ~ ip, data = trainset, control = rpart.control(cp = .0005))

# Previsões nos dados de teste
tree_pred = predict(modelo_rf_v1, testset, type='class')

# Percentual de previsões corretas com dataset de teste
mean(tree_pred==testset$is_attributed)
```

```
## [1] 0.9992272
```

```
# Confusion Matrix
table(tree_pred, testset$is_attributed)
```

```
##
## tree_pred    0    1
##         0 2586    2
##         1    0    0
```

Fim

www.github.com/henryjordan