Projeto 1 - FeedBack -> Analise Fraudulenta de Clicks

Marcio de Lima

15 de maio de 2019

Analise Fraudulenta de Clicks

Em resumo, neste projeto, construi um modelo de aprendizado de máquina que determinar se um clique é fraudulento ou não.

Dicionario de Dados - Descricao das Colunas

- ip: ip address of click.
- app: app id for marketing.
- device: device type id of user mobile phone (e.g., iphone 6 plus, iphone 7, huawei mate 7, etc.)
- os: os version id of user mobile phone
- channel: channel id of mobile ad publisher
- click_time: timestamp of click (UTC)
- attributed_time: if user download the app for after clicking an ad, this is the time of the app download
- is_attributed: the target that is to be predicted, indicating the app was downloaded

Definindo o tipo Modelo de Machine Learning

Baseado no problema de negocio informado acima, será criado um modelo do tipo de Classificacao de Machine Learning de Aprendizado Supervisionado.

Etapa Inicial - Carregando as bibliotecas e classes utilitarias

```
library(data.table)
library(caret)
```

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

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

```
## Registered S3 methods overwritten by 'ggplot2':
##   method         from
##   [.quosures     rlang
##   c.quosures     rlang
##   print.quosures rlang
```

```
library(dplyr)
```

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

```
## The following objects are masked from 'package:data.table':
##
##     between, first, last
```

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

```
library(ggplot2)
library(caTools)
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
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(e1071)
library(ROSE)
```

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

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

```
## Loading required package: gplots
##
## Attaching package: 'gplots'
## The following object is masked from 'package:stats':
##
##     lowess
```

```
source("Utils.R")
```

Etapa 1 - Coletando os Dados

Aqui está a coleta de dados, neste caso um arquivo csv.

```
#Carregando os dados
df <- fread("dataset/train_sample.csv", header = T, sep = ",", stringsAsFactors = FALSE)
```

Etapa 2 - Data Muning

Arrumando os dados

```
colunasFator <- c("is_attributed")
df <- to.factors(df, colunasFator)
df$click_time <- get_asPOSIXct(df, 6)
df$attributed_time <- get_asPOSIXct(df, 7)
```

Limpando a coluna ID, sem utilidade. Obs.: Poderiamos criar faixas pelo IP quebrando numa nova coluna de PAIS de origem, mas fica pro futuro.

```
df$ip <- NULL
```

Tratamento dos Campos NA's. Decisão de colocar a mesma Data do click_time

```
df$attributed_time <- ifelse(is.na(df$attributed_time), df$click_time, df$attributed_time)
df$attributed_time <- as.POSIXct(as.integer(df$attributed_time), origin = "1970-01-01")
```

Balanceamento da Variavel TARGET - Variavel TARGET sem balanceamento, modelo sera tendencioso dessa forma

```
table(df$is_attributed)
```

```
##
##     0     1
## 99773   227
```

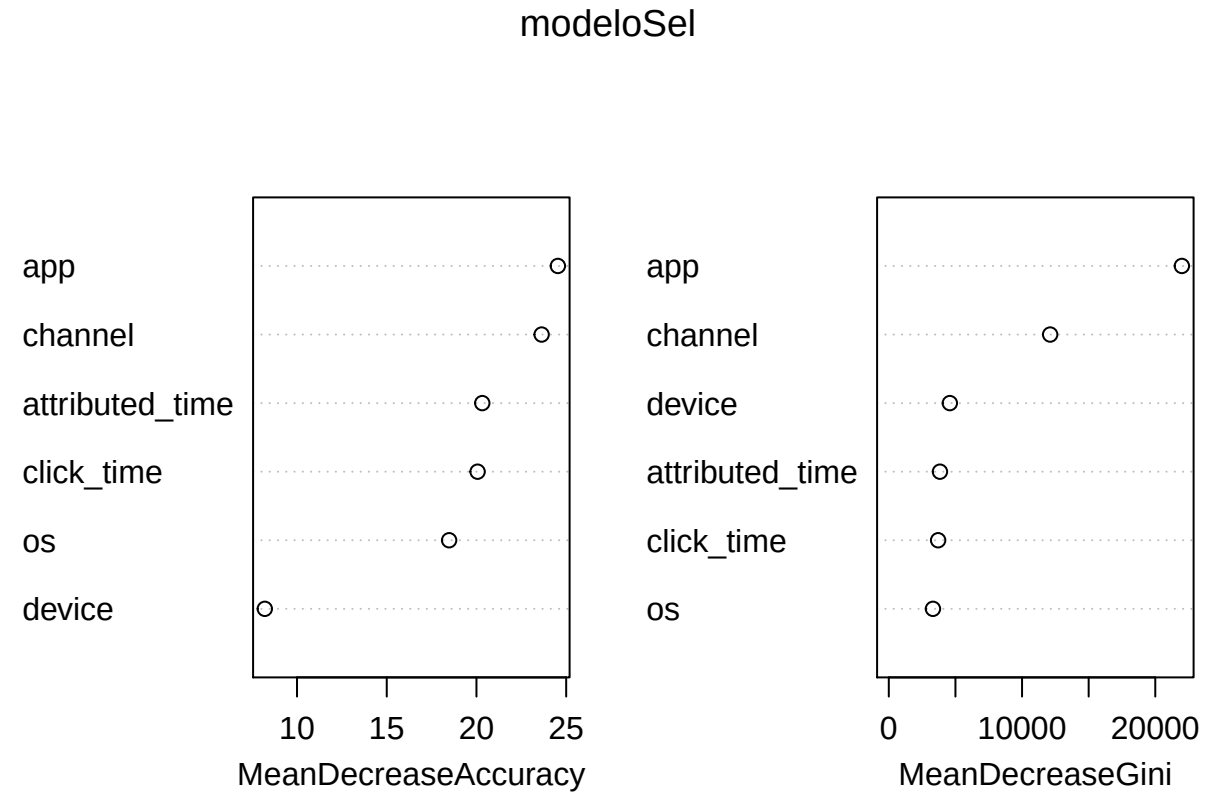
```
df2 = ovun.sample(is_attributed ~ . , data = df, method = "both", p=0.5)$data
table(df2$is_attributed)
```

```
##
##     0     1
## 50036 49964
```

Etapa 3 - Feature Selection

Mapeando as melhores variaveis para o modelo preditivo

```
modeloSel <- randomForest(is_attributed ~ . ,
                          data = df2,
                          ntree = 15,
                          nodesize = 10,
                          importance = TRUE)
varImpPlot(modeloSel)
```



Decisao de modelar com as variaveis: “app”,“channel”, “os”

Etapa 4 - Split de dados

Split de dados

```
# Funcao para gerar dados de treino e dados de teste
indice = sample.split(df2, SplitRatio = 0.7)

# Gerando dados de treino e de teste - Separando os dados
dados_treino <- df2[indice==TRUE,]
dados_teste <- df2[indice==FALSE,]
class1 <- dados_teste$is_attributed
dados_teste$is_attributed <- NULL
```

Etapa 5 - Modelo

Foram realizados vários testes com vários algoritmos, o melhor modelo e algoritmo segue abaixo

```
formula.full <- "is_attributed  ~ ."
formula.full <- as.formula(formula.full)

modelo_rf_v3 = rpart(formula.full,
                     data = dados_treino, control = rpart.control(cp = .0005))
```

```
class3 <- predict(modelo_rf_v3, dados_teste, type='class')
```

Etapa 6 - Avaliação do Modelo

Avaliando o modelo -> 99% de Acuraria => Melhor modelo comparando com SVM e com Random Forest com as variavéis selecionadas.

```
confusionMatrix(class3, class1)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0 21125     0
##          1   319 21413
##
##                Accuracy : 0.9926
##                  95% CI : (0.9917, 0.9933)
##     No Information Rate : 0.5004
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9851
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9851
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.9853
##              Prevalence : 0.5004
##          Detection Rate : 0.4929
##    Detection Prevalence : 0.4929
##       Balanced Accuracy : 0.9926
##
##        'Positive' Class : 0
##
```

Etapa 7 - Comparacao

Tabela Comparativa

- svm com kernel polynomial => 62%
- glm com binomial => 72%
- svm com kernel radial => 85%
- rpart com as variáveis selecionadas => 97%
- rpart com todas as variáveis => 99%

Muito Obrigado.

Att.

Marcio de Lima