PrevendoCliquesFraudulentos

Luiz Carlos Castro Cunha Junior

27/11/2019

O objetivo do desenvolvimento do projeto consiste em identificar se um clique é fraudulento ou não.

#Configurando o working directory setwd("C:\FCD\BigDataAnalytics-R-Azure\Projeto")

#Carregando as biblíotecas library(tidyverse) library(lubridate) library(corrplot) library(caret) library(rmarkdown)

Carregando uma amostra de 4.000.000 dados do dataset

dt<- read_csv('datasets/train.csv',n_max = 4000000)

Verificando os valores NA do conjunto de dados

colSums(is.na(dt)) colSums(is.na(dt))

Analisando a distribuição dos valores da variável target

table(dt$is_attributed)

Podemos observar que há 3993058 valores NA na coluna attributed_time e 3993058 valores falsos(0) na coluna is_attributed,

o que está completamente certo, pois quando is_attributed é verdadeira (1) aparece o tempo total do download na coluna attributed_time.

E quando ele é 0(falso) é atribuído NA, então a coluna attributed_time será removida.

dt$attributed_time <- NULL

Visualizando os dados

View(dt)

Criando as colunas dia, mes e hora através da coluna click_time.

Não será adicionadoa coluna ano, pois os dados são do ano de 2017

dt <- dt %>% mutate(dia = as.factor(day(click_time)), mes = month(click_time), hora = hour(click_time))

#Visualizando os dados após a mudança View(dt)

Como as colunas foram preenchidas corretamente, não há sentido em manter a coluna click_time, então iremos remover a mesma

dt$click_time <- NULL

#Verificando os valores únicos dos dados f_unicos <- function(x){ length(unique(x)) }

lapply(dt, f_unicos)

Após aplicar a função nos dados percebemos que as colunas dia e mês possuem apenas 1 valor único,

que são respectivamente: 6(dia) e 11(mês).

Removendo as duas colunas

dt$dia <- -NULLdt$mes <- NULL

#Visualizando a correlação entre as variáveis corrplot(cor(dt))

#Convertendo as variáveis hora e is_attributed para o tipo fator

dt$is_attributed <- -factor(dt$is_attributed) dt$hora <- -factor(dt$hora) str(dt)

#Visualizando o total de cliques ggplot(data = dt, aes(x = is_attributed))+ geom_bar(stat = 'count', width = 0.4, fill = c('green', 'black'))+ labs(title = "Relação entre os Cliques Fraudulentos x Cliques não Fraudulentos", x = 'Cliques')+ theme_minimal()

#Visualizando o total de cliques por hora ggplot(data = dt, aes(x = hora, fill= is_attributed))+ geom_bar()+ labs(title = 'Relação de cliques por Hora',x = "Cliques")+ theme_minimal()

#Separando os dados em treino e teste library(caret) set.seed(5000) split <- createDataPartition(y = dt$is_attributed, p = 0.70, list = F)

Criando dados de treino e de teste

dados_treino <- dt[split,] dados_teste <- dt[-split,]

str(dados_treino)

#Verificando a proporção dos dados round(prop.table(table(dados_teste$is_attributed))*100, 2)round(prop.table(table(dados_tre
* 100, 2)

#Utilizando o algoritmo de árvores de decisão library(rpart) set.seed(1234)

modelo <- rpart(is_attributed ~., data = dados_treino, method = "class")

#Criando um plot do modelo de árvores de decisão library(rpart) library(rpart.plot) rpart.plot(modelo)

#Fazendo previsões com o modelo previsoes <- predict(modelo, dados_teste, type = 'class')

#Verificando o desempenho do modelo round(prop.table(table(previsoes == dados_teste$is_attributed)) * 100, 2) summary(modelo)

#Verificando a taxa de acerto do modelo com uma confussion matrix

confusionMatrix(previsoes, dados_teste$is_attributed)

#O modelo obteve uma taxa de acurácia de 99.85%