Proyecto Data Science

José Alberto Félix Rodríguez

2020-11-12

El Proyecto de Data Science consiste en:

1)Planteamiento de preguntas del proyecto junto con un análisis exploratorio y una gráfica.

2)Entrenamiento del modelo de clasificación de KNN.

##### Entrenamiento del Modelo de Clasificación de KNN

Cargamos los paquetes a utilizar.

require(class)

## Loading required package: class

require(gmodels)

## Loading required package: gmodels

require(dplyr)

## Loading required package: 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

library(ggplot2)  
library(tidyverse)

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

## v tibble 3.0.4 v purrr 0.3.4  
## 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(dplyr)

Abrimos las bases de datos que utilizaremos y las guardamos en una variable:

dbv <- "C:\\Users\\Beto Felix\\Desktop\\Beto\\SPPC\\Curso Análisis de Datos\\klustera\\v.csv"  
dbe <- "C:\\Users\\Beto Felix\\Desktop\\Beto\\SPPC\\Curso Análisis de Datos\\klustera\\e.csv"  
  
## Guardar base de datos en variable  
e <- read.csv(paste(dbe, sep="/"), stringsAsFactors = FALSE)  
v <- read.csv(paste(dbv, sep="/"), stringsAsFactors = FALSE)

Quitamos la primera fila de la base de datos “e”

str(e)

## 'data.frame': 249556 obs. of 9 variables:  
## $ X : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ device\_mac : chr "00:00:00:6a:17:ce" "00:00:00:6a:17:ce" "00:00:00:6a:78:ae" "00:00:02:fa:9c:12" ...  
## $ branch\_office : int 1 1 1 1 3 3 2 3 1 1 ...  
## $ month\_tz : int 11 11 10 11 11 11 11 11 11 11 ...  
## $ day\_tz : int 2 2 30 13 4 3 4 5 19 8 ...  
## $ day\_of\_week\_tz: chr "Wednesday" "Wednesday" "Sunday" "Sunday" ...  
## $ hour\_tz : int 0 1 20 12 19 5 19 17 16 11 ...  
## $ visitor : chr "true" "true" "false" "true" ...  
## $ tiempodeses : int 2676 2676 0 2272 0 0 0 0 0 0 ...

e <- e[-1]  
str(e)

## 'data.frame': 249556 obs. of 8 variables:  
## $ device\_mac : chr "00:00:00:6a:17:ce" "00:00:00:6a:17:ce" "00:00:00:6a:78:ae" "00:00:02:fa:9c:12" ...  
## $ branch\_office : int 1 1 1 1 3 3 2 3 1 1 ...  
## $ month\_tz : int 11 11 10 11 11 11 11 11 11 11 ...  
## $ day\_tz : int 2 2 30 13 4 3 4 5 19 8 ...  
## $ day\_of\_week\_tz: chr "Wednesday" "Wednesday" "Sunday" "Sunday" ...  
## $ hour\_tz : int 0 1 20 12 19 5 19 17 16 11 ...  
## $ visitor : chr "true" "true" "false" "true" ...  
## $ tiempodeses : int 2676 2676 0 2272 0 0 0 0 0 0 ...

Verificamos que no haya datos NA en la base de datos “e”

sapply(e, function(x) sum(is.na(x)))

## device\_mac branch\_office month\_tz day\_tz day\_of\_week\_tz   
## 0 0 0 0 0   
## hour\_tz visitor tiempodeses   
## 0 0 0

Cambiamos a factor la columna de visitor y convertimos su contenido en Visitante y No Visitante. También sacamos en porcentaje cuántos Visitantes y cuántos No Visitantes hay

e$visitor <- factor(e$visitor, levels = c("true", "false"),  
 labels = c("Visitante", "No Visitante"))  
  
round(prop.table(table(e$visitor)) \* 100, digits = 1)

##   
## Visitante No Visitante   
## 34.2 65.8

Exploramos los valores de los atributos que tienen un mínimo y un máximo muy separados entre si y normalizamos esos valores.

summary(e[c("tiempodeses", "day\_tz", "hour\_tz")])

## tiempodeses day\_tz hour\_tz   
## Min. : 0 Min. : 1.0 Min. : 0.00   
## 1st Qu.: 0 1st Qu.: 7.0 1st Qu.:10.00   
## Median : 0 Median :13.0 Median :14.00   
## Mean : 2375 Mean :14.8 Mean :13.64   
## 3rd Qu.: 421 3rd Qu.:24.0 3rd Qu.:18.00   
## Max. :68062 Max. :31.0 Max. :23.00

normalize <- function(x) {  
 return ((x - min(x)) / (max(x) - min(x)))  
}  
  
e\_n <- as.data.frame(lapply(e[c(4,6,8)], normalize)) #list apply  
head(e\_n)

## day\_tz hour\_tz tiempodeses  
## 1 0.03333333 0.00000000 0.03931709  
## 2 0.03333333 0.04347826 0.03931709  
## 3 0.96666667 0.86956522 0.00000000  
## 4 0.40000000 0.52173913 0.03338133  
## 5 0.10000000 0.82608696 0.00000000  
## 6 0.06666667 0.21739130 0.00000000

Separamos la base de datos en un database de prueba y en otro de entrenamiento, guardando la clasificación que tuvieron en otra variable.

nfilas <- floor(nrow(e\_n) \* .80)  
set.seed(123)  
index <- sample(7:nrow(e\_n), nfilas) # 80%  
e\_train <- e\_n[1:80000, ] # Obtener solo las muestras  
e\_test <- e\_n[80001:90000, ] # Todo menos las muestras  
  
# Guardamos la clasificaciÃ³n de cada uno (B o M) de la primera columna  
e\_train\_labels <- e[1:80000, 7]  
e\_test\_labels <- e[80001:90000, 7]  
str(e\_train\_labels)

## Factor w/ 2 levels "Visitante","No Visitante": 1 1 2 1 2 2 2 2 2 2 ...

Se realiza la predicción con un número de veces que se analizara siendo “K” ese número y realizamos una tabla para verificar el resultado de nuestra predicción.

e\_test\_pred <- knn(train = e\_train, test = e\_test, cl = e\_train\_labels, k = 30)  
  
## ----------- Evaluamos los resultados del modelo   
# Creamos una tabla para compara predicciones vs real  
CrossTable(x = e\_test\_labels, y = e\_test\_pred, prop.r=FALSE, prop.chisq = FALSE)

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | N / Col Total |  
## | N / Table Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 10000   
##   
##   
## | e\_test\_pred   
## e\_test\_labels | Visitante | No Visitante | Row Total |   
## --------------|--------------|--------------|--------------|  
## Visitante | 2714 | 806 | 3520 |   
## | 0.995 | 0.111 | |   
## | 0.271 | 0.081 | |   
## --------------|--------------|--------------|--------------|  
## No Visitante | 15 | 6465 | 6480 |   
## | 0.005 | 0.889 | |   
## | 0.002 | 0.646 | |   
## --------------|--------------|--------------|--------------|  
## Column Total | 2729 | 7271 | 10000 |   
## | 0.273 | 0.727 | |   
## --------------|--------------|--------------|--------------|  
##   
##

Agregamos la predicción a la base “v” en una nueva columna llamada “Visitor”

v$visitor <- e\_test\_pred  
summary(v$visitor)

## Visitante No Visitante   
## 24561 65439

##### ¿Cuál es el mes con mayor número de visitantes?

Para sacar el mes con el mayor número de visitantes, primero se realizará un filtro sobre la columna “Visitor” para dejar solo los valores con “Visitantes”. Luego se agrupara por mes y por último, se contará los registros que salieron en las agrupaciones.

mes <- v %>%  
 filter(visitor=="Visitante") %>%  
 group\_by(month\_tz) %>%  
 count() %>%  
 mutate(total=n)

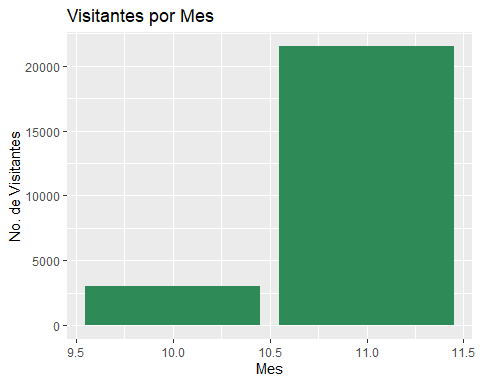
Se obtiene los siguientes datos:

mes

## # A tibble: 2 x 3  
## # Groups: month\_tz [2]  
## month\_tz n total  
## <int> <int> <int>  
## 1 10 3031 3031  
## 2 11 21530 21530

Se realiza la gráfica de los datos anteriores

gr <- ggplot(mes, aes(x=month\_tz, y=total)) +  
 geom\_col(fill="seagreen")+labs(title = "Visitantes por Mes", x="Mes", y="No. de Visitantes")  
gr



###### ¿Cuál es el día de la semana con mayor número de visitantes?

Para sacar el día de la semana con el mayor número de visitantes, primero se realizará un filtro sobre la columna “Visitor” para dejar solo los valores con “Visitantes”. Luego se agrupara por día y por último, se contará los registros que salieron en las agrupaciones.

dia <- v %>%  
 filter(visitor=="Visitante") %>%  
 group\_by(day\_of\_week\_tz) %>%  
 count() %>%  
 mutate(total=n)

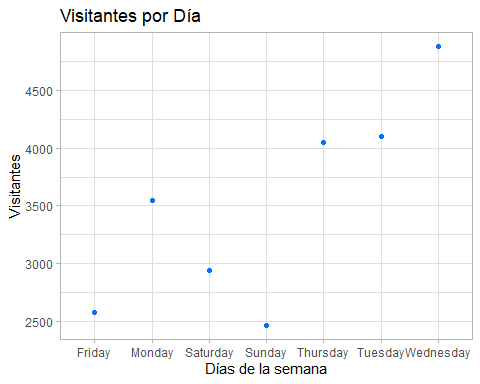
Se obtiene los siguientes datos:

dia

## # A tibble: 7 x 3  
## # Groups: day\_of\_week\_tz [7]  
## day\_of\_week\_tz n total  
## <chr> <int> <int>  
## 1 Friday 2580 2580  
## 2 Monday 3545 3545  
## 3 Saturday 2941 2941  
## 4 Sunday 2465 2465  
## 5 Thursday 4049 4049  
## 6 Tuesday 4102 4102  
## 7 Wednesday 4879 4879

Se realiza la gráfica de los datos anteriores

gr <- ggplot(dia, aes(x=day\_of\_week\_tz, y=total)) +  
 geom\_point(color="#0570F1")+ labs(title = "Visitantes por Día",x="Días de la semana",y="Visitantes")+  
 theme\_light()  
gr



##### ¿Cuál es la franquicia que cuenta con el mayor número de visitantes?

Para sacar la franquicia que obtuvo el mayor número de visitantes, primero se realizará un filtro sobre la columna “Visitor” para dejar solo los valores con “Visitantes”. Luego se agrupara por franquicia y por último, se contará los registros que salieron en las agrupaciones.

branch <- v %>%  
 filter(visitor=="Visitante") %>%  
 group\_by(branch\_office) %>%  
 count() %>%  
 mutate(total=n)

Se obtiene los siguientes datos:

branch

## # A tibble: 3 x 3  
## # Groups: branch\_office [3]  
## branch\_office n total  
## <int> <int> <int>  
## 1 1 12541 12541  
## 2 2 2382 2382  
## 3 3 9638 9638

Se realiza la gráfica de los datos anteriores

gr <- ggplot(branch, aes(x=branch\_office, y=total)) +  
 geom\_col(fill="royalblue")+labs(title = "Visitantes por Franquicia", x="Franquicia", y="No. de Visitantes")  
gr

