Actividad: PRA2: Limpieza y análisis de datos

Reison A. Torres Urina

Diciembre 2019

Índice

1. Detalles de la actividad

1.1. Descripción

En esta practica se elabora un caso práctico orientado a aprender a identificar los datos relevantes para un proyecto analítico y usar las herramientas de integración, limpieza, validación y análisis de las mismas.

1.2. Objetivos

Los objetivos concretos de esta práctica son:

- Aprender a aplicar los conocimientos adquiridos y su capacidad de resolución de problemas en entornos nuevos o poco conocidos dentro de contextos más amplios o multidisciplinares.
- Saber identificar los datos relevantes y los tratamientos necesarios (integración, limpieza y validación) para llevar a cabo un proyecto analítico.
- Aprender a analizar los datos adecuadamente para abordar la información contenida en los datos.
- Identificar la mejor representación de los resultados para aportar conclusiones sobre el problema planteado en el proceso analítico.
- Actuar con los principios éticos y legales relacionados con la manipulación de datos en función del ámbito de aplicación.
- Desarrollar las habilidades de aprendizaje que les permitan continuar estudiando de un modo que tendrá que ser en gran medida autodirigido o autónomo.
- Desarrollar la capacidad de búsqueda, gestión y uso de información y recursos en el ámbito de la ciencia de datos.

1.3. Competencias

En esta práctica se desarrollan las siguientes competencias del Máster de Data Science:

- Capacidad de analizar un problema en el nivel de abstracción adecuado a cada situación y aplicar las habilidades y conocimientos adquiridos para abordarlo y resolverlo.
- Capacidad para aplicar las técnicas específicas de tratamiento de datos (integración, transformación, limpieza y validación) para su posterior análisis.

2. Solución

2.1. Descripción del dataset

2.1.1. Carga de los datos

Cargamos el conjunto de datos que se encuentran en los archivos **train.csv**, **test.csv** y **gender_submission.csv** en formato CSV, y representan los datos de los pasajeros que abordaron el Titanic.

Estos datos estarán representados en R por un dataframe para facilitar la manipulación de los mismos en nuestro análisis.

```
# cargamos paquetes R que vamos a utilizar durante nuestro anlisis

if(!require(ggplot2)){
    #install.packages('ggplot2', repos='http://cran.us.r-project.org')
    library(ggplot2)
}

if(!require(ggpubr)){
    #install.packages('ggpubr', repos='http://cran.us.r-project.org')
    library(ggpubr)
}

library(dplyr)
#library(Hmisc)
#library(corrplot)

# Carga del dataset contenido en el archivo train.csv
titanic.train <- read.csv("../datos/train.csv",stringsAsFactors = FALSE, header=T, sep=",")
# Carga del dataset contenido en el archivo test.csv
titanic.test <- read.csv("../datos/test.csv",stringsAsFactors = FALSE, header=T, sep=",")
# Carga del dataset contenido en el archivo gender_submission.csv
titanic.test.survived <- read.csv("../datos/gender_submission.csv",stringsAsFactors = FALSE, header=T, s
```

2.1.2. Descripción

Los datos seleccionados, fueron obtenidos del sitio de data science, **www.Kaggle.com**, en el encontramos una variedad de dataset Open Data. El conjunto de datos seleccionados para desarrollar esta actividad es **Titanic: Machine Learning from Disaster**, en este dataset encontramos, los datos de los pasajeros, que abordaron el Titanic en su viaje inaugural.

Los datos de este dataset se encuentran divididos en dos archivos train.csv con 891 observaciones y test.cvs con 418 observaciones para un total de 1309. El conjunto de datos esta descripto por un conjunto de 12 variables. Las característica presenten en este dataset, nos permitirá cumplir los objetivos propuestos en esta actividad.

Variables contenidas en el dataset **train.csv**:

```
str(titanic.train)
```

```
## 'data.frame':    891 obs. of  12 variables:
##  $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Survived   : int  0 1 1 1 0 0 0 0 1 1 ...
##  $ Pclass     : int  3 1 3 1 3 3 1 3 3 2 ...
##  $ Name       : chr  "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley (Florence Briggs Thayer)"
##  $ Sex        : chr  "male" "female" "female" "female" ...
##  $ Age        : num  22 38 26 35 35 NA 54 2 27 14 ...
##  $ SibSp      : int  1 1 0 1 0 0 0 3 0 1 ...
##  $ Parch      : int  0 0 0 0 0 0 0 1 2 0 ...
##  $ Ticket     : chr  "A/5 21171" "PC 17599" "STON/O2. 3101282" "113803" ...
##  $ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Cabin      : chr  "" "C85" "" "C123" ...
##  $ Embarked   : chr  "S" "C" "S" "S" ...
```

Variables contenidas en el dataset **test.cvs**:

```
str(titanic.test)
```

```
## 'data.frame':    418 obs. of  11 variables:
##  $ PassengerId: int  892 893 894 895 896 897 898 899 900 901 ...
##  $ Pclass     : int  3 3 2 3 3 3 3 2 3 3 ...
##  $ Name       : chr  "Kelly, Mr. James" "Wilkes, Mrs. James (Ellen Needs)" "Myles, Mr. Thomas Francis
##  $ Sex        : chr  "male" "female" "male" "male" ...
##  $ Age        : num  34.5 47 62 27 22 14 30 26 18 21 ...
##  $ SibSp      : int  0 1 0 0 1 0 0 1 0 2 ...
##  $ Parch      : int  0 0 0 0 1 0 0 1 0 0 ...
##  $ Ticket     : chr  "330911" "363272" "240276" "315154" ...
##  $ Fare       : num  7.83 7 9.69 8.66 12.29 ...
##  $ Cabin      : chr  "" "" "" "" ...
##  $ Embarked   : chr  "Q" "S" "Q" "S" ...
```

Variables contenidas en el dataset **gender_submission.csv**:

```
str(titanic.test.survived)
```

```
## 'data.frame':    418 obs. of  2 variables:
##  $ PassengerId: int  892 893 894 895 896 897 898 899 900 901 ...
##  $ Survived   : int  0 1 0 0 1 0 1 0 1 0 ...
```

Este dataset **gender_submission.csv** contiene la variable de survived, que luego utilizaremos para agregar al dataset **titanic.test**.

A continuación describimos el conjunto de variables que conforman este dataset:

- **PassengerId:** Número consecutivo que identifica al pasajero.
- **Name:** Nombre del pasajero.
- **Sex:** Define el xeso del pasajero.

- **pclass:** Nivel socioeconómico del pasajero (1st = Upper, 2nd = Middle, 3rd = Lower).
- **age:** Edad del pasajero en años.
- **sibsp: Familiar abordo del Titanic. Define la relación familiar de la siguiente forma:**
  Hermanos => 1 = hermana, 2 = hermano, 3 = hermanastro, 4 = hermanastra
  Esposos => 5 = esposo, 6 = esposa, 7 = amantes y 8 = novio
- **parch: Familiar abordo del Titanic. Define la relación familiar de la siguiente forma:**
  Padres => 1 = madre, 2 = padre
  Hijos => 3 = hija, 4 = hijo, 5 = hijastra, 6 = hijastro
- **ticket:** Número del boleto de abordaje.
- **fare:** Precio del boleto.
- **cabin:** Número de la cabina.
- **embarked:** Puerto de embarque (C = Cherbourg, Q = Queenstown, S = Southampton).
- **survived:** Pasajero superviviente (0 = No, 1 = Yes)

2.1.3. Importancia y objetivos de los análisis

A partir de este conjunto de datos se plantea la problemática de determinar qué variables influyen más en la supervivencia de un pasajero en el naufragio del Titanic. Además, se podrá proceder a crear modelos de regresión que permitan predecir si un pasajero sobrevive o no en función de sus características y contrastes de hipótesis que ayuden a identificar propiedades interesantes en las muestras que puedan ser inferidas con respecto a la población.

Este tipo de análisis pueden ser utilizados por las aseguradoras del sector turístico, para determinar el riesgo que puede tener un turista al viajar en los trasatlánticos. Y asi poder ofreser las cobertura del seguro.

2.2. Integración y selección de los datos de interés a analizar

2.2.1. Integración

Con el fin de tener una estructura de datos coherente y única que contenga mayor cantidad de información, combinaremos los datos procedentes de los dataset train.csv y test.cvs. Luego realizaremos una fusión horizontal para añadir el atributo **survived**, debido a que el dataset **test.cvs** no presenta este atributo. Este valor será extraído del dataset **gender_submission.csv**.

```
# Realizamos una fusión horizontal entre los dataset titanic.test y titanic.test.survived para agregar
titanic.test <- inner_join(titanic.test, titanic.test.survived, by ="PassengerId")

#Creamos el dataset titanc.data con la combinacion de los datos de los dataset  titanic.train y titanic
titanc.data <- bind_rows(titanic.train,titanic.test)

# Eliminamos los dataset temporales
rm(titanic.test.survived)
rm(titanic.test)
```

```
rm(titanic.train)

# Verificamos la estructura del dataset con los datos combinados
str(titanc.data)
```

```
## 'data.frame':    1309 obs. of  12 variables:
##  $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Survived   : int  0 1 1 1 0 0 0 0 1 1 ...
##  $ Pclass     : int  3 1 3 1 3 3 1 3 3 2 ...
##  $ Name       : chr  "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley (Florence Briggs Thayer)"
##  $ Sex        : chr  "male" "female" "female" "female" ...
##  $ Age        : num  22 38 26 35 35 NA 54 2 27 14 ...
##  $ SibSp      : int  1 1 0 1 0 0 0 3 0 1 ...
##  $ Parch      : int  0 0 0 0 0 0 0 1 2 0 ...
##  $ Ticket     : chr  "A/5 21171" "PC 17599" "STON/O2. 3101282" "113803" ...
##  $ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Cabin      : chr  "" "C85" "" "C123" ...
##  $ Embarked   : chr  "S" "C" "S" "S" ...
```

2.2.2. Selección de los datos

La gran mayoría de las variables contenidas en el conjunto de datos corresponde con características de los pasajeros que abordaron el Titanic, por lo que serán tenidas en cuenta para realizar nuestro análisis. Sin embargo, podremos prescindir de las variables (**PassengerId**,**Name**) dado que estos atributos no aportan una carasterisitica al pasajero, y no influye en la resolución de nuestro problema.

```
# Eliminamos del dataset las variables "PassengerId" y "Name"
titanc.data <- titanc.data[,!(colnames(titanc.data) %in% c("PassengerId","Name"))]

# Verificamos la estructura del dataset
str(titanc.data)
```

```
## 'data.frame':    1309 obs. of  10 variables:
##  $ Survived: int  0 1 1 1 0 0 0 0 1 1 ...
##  $ Pclass  : int  3 1 3 1 3 3 1 3 3 2 ...
##  $ Sex     : chr  "male" "female" "female" "female" ...
##  $ Age     : num  22 38 26 35 35 NA 54 2 27 14 ...
##  $ SibSp   : int  1 1 0 1 0 0 0 3 0 1 ...
##  $ Parch   : int  0 0 0 0 0 0 0 1 2 0 ...
##  $ Ticket  : chr  "A/5 21171" "PC 17599" "STON/O2. 3101282" "113803" ...
##  $ Fare    : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Cabin   : chr  "" "C85" "" "C123" ...
##  $ Embarked: chr  "S" "C" "S" "S" ...
```

2.3. Limpieza de los datos

2.3.1. Discretización y conversion de tipos de datos

Al cargar los archivos con la función read.csv(), esta de manera automática asigna el tipo de variable en el dataset, en ciertas ocasiones los tipos asignados, no son los correctos. A continuación visualizamos los tipos de variables asignados al dataset, para luego decidir si se requiere una conversión de tipo.

```
# Tipos de variables
titanc.data.ctype <- sapply(titanc.data,class)
```

```
titanc.data.ctype <- data.frame(variables = names(titanc.data.ctype),tipo = as.vector(titanc.data.ctype)

titanc.data.ctype
```

```
##    variables      tipo
## 1   Survived   integer
## 2     Pclass   integer
## 3        Sex character
## 4        Age   numeric
## 5      SibSp   integer
## 6      Parch   integer
## 7     Ticket character
## 8       Fare   numeric
## 9      Cabin character
## 10  Embarked character
```

```
rm(titanc.data.ctype)
```

En este paso realizamos un analisis sobre las variables, que en R han sido cargadas como continuas pero en realidad son discretas (factor).Para esto realizamos un análisis de discretizacion sobre los atributos, para identificar que variables tienen sentido discretizar.

```
#summary(titanc.data[,titanc.data.ctype[titanc.data.ctype$tipo == "numeric",]$variables])
# Identificar el número de clases que se encuentra en cada variable del dataset
apply(titanc.data,2, function(x) length(unique(x)))
```

```
## Survived   Pclass      Sex      Age    SibSp    Parch   Ticket     Fare
##        2        3        2       99        7        8      929      282
##    Cabin Embarked
##      187        4
```

Con el fin de facilitar la interpretar y comparar los resultados de diferentes grupos de datos, procedemos a discretizar a las variables con pocas clases:

```
cols<-c("Survived","Pclass","Sex","SibSp","Parch","Embarked")
for (i in cols){
  titanc.data[,i] <- as.factor(titanc.data[,i]) # Conversion de variable a tipo factor
}

levels(titanc.data[,"Survived"]) <- c("No","Si")
levels(titanc.data[,"Pclass"]) <- c("Upper","Middle", "Lower")
levels(titanc.data[,"Embarked"]) <- c("?","Cherbourg", "Queenstown", "Southampton")

str(titanc.data)
```

```
## 'data.frame':    1309 obs. of  10 variables:
##  $ Survived: Factor w/ 2 levels "No","Si": 1 2 2 2 1 1 1 1 2 2 ...
##  $ Pclass  : Factor w/ 3 levels "Upper","Middle",..: 3 1 3 1 3 3 1 3 3 2 ...
##  $ Sex     : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 2 1 1 ...
##  $ Age     : num  22 38 26 35 35 NA 54 2 27 14 ...
##  $ SibSp   : Factor w/ 7 levels "0","1","2","3",..: 2 2 1 2 1 1 1 4 1 2 ...
##  $ Parch   : Factor w/ 8 levels "0","1","2","3",..: 1 1 1 1 1 1 1 2 3 1 ...
##  $ Ticket  : chr  "A/5 21171" "PC 17599" "STON/O2. 3101282" "113803" ...
##  $ Fare    : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Cabin   : chr  "" "C85" "" "C123" ...
##  $ Embarked: Factor w/ 4 levels "?","Cherbourg",..: 4 2 4 4 4 3 4 4 4 2 ...
```

2.3.2. Tratamientos de ceros o elementos vacíos

Los datos vacíos o no definidos pueden presentarse en distintos formatos, típicamente “”, ? ,“ “ o NA (Not Available en inglés), pero en algunos contextos pueden incluso tomar valores numéricos como 0 o 999.

A continuación inspeccionaremos, que atributos de nuestro dataset, tienen una cantidad alta de valores no disponibles o valores faltantes en los diferentes formatos (“”,?, o NA):

```
# Funcion: Explorar atributos con valores faltante
# Parmetros:
# 1. dataset: conjunto de datos con los atributos a explorar
hasValoresFaltantes <- function(dataset){
  # Verificar si existen variables cuantitativas con valores NA
  variablesWithNA <- colSums(is.na(dataset))

  # Verificar si existen variables con cadenas vacias
  variablesWithEmpaty <- colSums(dataset=="")
  variablesWithEmpaty[is.na(variablesWithEmpaty)] <- 0

  # Verificar si existen variables con valores desconocidos ("?").
  variablesWithQuestionMark <- colSums(dataset=="?")
  variablesWithQuestionMark[is.na(variablesWithQuestionMark)] <- 0

  # Verificar si existen variables con valores desconocidos (" ").
  variablesWithSpace <- colSums(dataset==" ")
  variablesWithSpace[is.na(variablesWithSpace)] <- 0

  df <- data.frame(variables = names(variablesWithNA),"NA" = as.vector(variablesWithNA),stringsAsFactors

  df = bind_cols(df,"Empaty" = as.vector(variablesWithEmpaty))
  df = bind_cols(df,"?" = as.vector(variablesWithQuestionMark))
  df = bind_cols(df,"Space" = as.vector(variablesWithSpace))

  df
  #ls <- list(valoresFaltantes = df);
  #ls$totalMuestras <- dim(dataset)[1]
  #ls
}
```

```
# Verificar si existen variables con valores faltantes
hasValoresFaltantes(titanc.data)
```

```
##    variables NA. Empaty ? Space
## 1   Survived   0      0 0     0
## 2     Pclass   0      0 0     0
## 3        Sex   0      0 0     0
## 4        Age 263      0 0     0
## 5      SibSp   0      0 0     0
## 6      Parch   0      0 0     0
## 7     Ticket   0      0 0     0
## 8       Fare   1      0 0     0
## 9      Cabin   0   1014 0     0
## 10  Embarked   0      0 2     0
```

Al observar el resultado del análisis anterior, podemos identificar que para las variables Age y Fare presenta valores faltantes (NA). Para la variable Cabin se identifica que presenta una cantidad alta de valores faltantes

en el formato vacío (“”). y para la variable Embarked se identifica valores faltantes en el formato “?”.

Llegados a este punto debemos decidir cómo manejar estos registros que contienen valores desconocidos:

Para el atributo **Embarked** realizamos un análisis de proporción de valores faltantes y lo actualizaremos en función del valor mas frecuente. Existen 2 casos con valor faltante con formato “?”, con una proporción del 0.15 %, el valor más frecuentes es “Southampton” con una proporción del 56.98 % .

```
arrange(data.frame(round(prop.table(table(titanc.data$Embarked)),4)*100),-Freq)
```

```
##          Var1  Freq
## 1 Southampton 69.82
## 2   Cherbourg 20.63
## 3  Queenstown  9.40
## 4           ?  0.15
```

```
# actualizamos los valores faltantes con el valor más frecuente
titanc.data$Embarked[titanc.data$Embarked=="?"] <- "Southampton"
```

Para el atributo **Cabin** realizamos un análisis de proporción de valores faltantes. Existen 1014 casos con valor faltante con formato vacío (“”), con una proporción del 77.46 %, esto corresponde a más de la mitad de las observaciones. Si intentamos completar los valores faltantes, por alguna de las técnicas de imputación de valores perdidos, debido a la alta cantidad de valores faltantes, en este atributo, nos puede generar sesgos en los datos de este atributo. De acuerdo a esto, se decide eliminar el atributo **Cabin** del dataset en estudio.

```
data.frame(Total=sort(colSums(titanc.data == ""), decreasing = TRUE),Porcentaje = sort(round(colMeans(t
```

```
##          Total Porcentaje
## Cabin     1014      77.46
## Survived     0       0.00
## Pclass       0       0.00
## Sex          0       0.00
## SibSp        0       0.00
## Parch        0       0.00
## Ticket       0       0.00
## Embarked     0       0.00
```

```
# Eliminamos la variable Cabin
titanc.data <- titanc.data[, !(names(titanc.data) %in% c("Cabin"))]
```

2.3.3. Identificación y tratamiento de valores extremos

2.4. Código fuente y dataset

3. Recursos