Tipología y ciclo de vida de los datos: PRA2 - Limpieza y análisis de datos

Autor: Giovanny Caluña y Ivan Ovalle

Mayo 2021

# Detalles de la actividad

## Descripción

En esta práctica 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.

## Objetivos

Los objetivos que se persiguen mediante el desarrollo de esta actividad práctica son los siguientes: - 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 permita 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.

## Competencias

Las competencias que se desarrollan son:

* 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.

# Desarrollo

## Descripción del dataset

Para el desarrollo de este proyecto se seleccionó el data set llamado: Titanic: Machine Learning. Obtenido de: (<https://www.kaggle.com/c/titanic>). El data set cuenta con la informacion de diferentes tripulantes que estuvieron en el Titanic el dia de su hundimineto. El data set cuenta con 12 atributos que ayudan a describir a cada pasajero: - PassengerID: Id del pasajero registrado. - Survived: Si el pasajero sobrevivió o no al incidente(0 = No, 1 = Si). - Pclass: Clase del ticket abordo del Titanic. Ej: 1,2 o 3. - Name: Nombre del pasajero. - Sex: Genero del pasajero. - Age: Edad del pasajero. - SibSp: Número de hermanos o conyuges a bordo del Titanic. - Parch: Número de padres o niños a bordo del Titanic. - Ticket: Número del ticket. - Fare: Tarifa del pasajero (costo del ticket). - Cabin: Número de cabina abordo del titanic. - Embarked: Puerto de embarcación (C = Cherbourg, Q = Queenstown, S = Southampton)

# Importancia y objetivos de los análisis

Tomando en cuenta la informacion que nos provee el data set, el siguiente trabajo tratará de desarrollar un modelo de clasificacion supervisado XXXXXXXXX, que nos ayudará a predecir si un pasajero sobrevivió o no al hundimiento del Titanic, basado en sus atributos. Este modelo nos ayudara a ratificar o contrastar las diferentes hipotesis que se deprenden en el analisis visual de los datos.

# Limpieza de los datos

El primer paso para elaborar nuestro modelo de clasificaion es: la limpieza de los datos. En esta etapa vamos a aplicar las diferentes tecnicas de limpieza de datos que nos perimitirá corregir posibles inconsistencias, valores nulos y atributos innecesarios.

# Eliminación de atributos

Para cargar los datos en un data frame compatible con nuestro entorno, leemos el fichero de tipo .csv, utilizando la función *read.csv* de R.

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

## Loading required package: ggplot2

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

## Loading required package: ggpubr

## Warning: package 'ggpubr' was built under R version 4.0.5

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

## Loading required package: grid

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

## Loading required package: gridExtra

if(!require(C50)){  
 install.packages('C50', repos='http://cran.us.r-project.org')  
 library(C50)  
}

## Loading required package: C50

## Warning: package 'C50' was built under R version 4.0.5

passengers<-read.csv("./train.csv",header=T,sep=",")

Ahora utilizaremos la funcion *sapply*, que nos proporcionará el tipo de dato que maneja cada atributo.

sapply(passengers, function(x) class(x))

## PassengerId Survived Pclass Name Sex Age   
## "integer" "integer" "integer" "character" "character" "numeric"   
## SibSp Parch Ticket Fare Cabin Embarked   
## "integer" "integer" "character" "numeric" "character" "character"

Como podemos observar en la tabla superior, en nuestro data set podemos encontrar 3 tipos de datos: integer, character y numeric. Ahora procedemos a utilizar la funcion *summary* que nos ayudará con datos estadisticos generales de cada atributo.

summary(passengers)

## PassengerId Survived Pclass Name   
## Min. : 1.0 Min. :0.0000 Min. :1.000 Length:891   
## 1st Qu.:223.5 1st Qu.:0.0000 1st Qu.:2.000 Class :character   
## Median :446.0 Median :0.0000 Median :3.000 Mode :character   
## Mean :446.0 Mean :0.3838 Mean :2.309   
## 3rd Qu.:668.5 3rd Qu.:1.0000 3rd Qu.:3.000   
## Max. :891.0 Max. :1.0000 Max. :3.000   
##   
## Sex Age SibSp Parch   
## Length:891 Min. : 0.42 Min. :0.000 Min. :0.0000   
## Class :character 1st Qu.:20.12 1st Qu.:0.000 1st Qu.:0.0000   
## Mode :character Median :28.00 Median :0.000 Median :0.0000   
## Mean :29.70 Mean :0.523 Mean :0.3816   
## 3rd Qu.:38.00 3rd Qu.:1.000 3rd Qu.:0.0000   
## Max. :80.00 Max. :8.000 Max. :6.0000   
## NA's :177   
## Ticket Fare Cabin Embarked   
## Length:891 Min. : 0.00 Length:891 Length:891   
## Class :character 1st Qu.: 7.91 Class :character Class :character   
## Mode :character Median : 14.45 Mode :character Mode :character   
## Mean : 32.20   
## 3rd Qu.: 31.00   
## Max. :512.33   
##

Como podemos observar, se obtiene la el minimo, la media y la mediana de todos los atributos de tipo integer y numeric. Ademas, en el atributo Age (Edad del pasajero) podemos observar un campo extra llamado NA´s. Este dato nos indica el numero de valores nulos o vacios que contienes este campo, en este caso 177.

Para la eliminacion de atributos innecesarios utilizaremos la funcion *str* de R que nos dara diferente ejemplos de cada atributo asi como su formato.

str(passengers)

## '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)" "Heikkinen, Miss. Laina" "Futrelle, Mrs. Jacques Heath (Lily May Peel)" ...  
## $ 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" ...

El primer atributo que se eliminará es: Passenger Id. Este atributo nos indica un numero asignado de manera incremetal a cada passajero por lo que no nos aporta informacion relevante para el analisi y entrenamiento del modelo. El segundo atributo eliminado será Name, este atributo de tipo character al ser diferente para cada pasajero, no proporciona informacion util para la clasificacion. De la misma manera, el atributo ticket, da un valor alfanumerico especifico para cada pasajero por lo que tambien se eliminará. Por ultimo, despues de un analisis visual sobre el data set, se detecto un alto numero de valores nulos sobre el campo Cabin y los pocos registros con los que se cuenta dan un valor especifico por pasajero. Para eliminar los atributos mecionados, se procede a ejecutar lo siguiente:

passengers$PassengerId<- NULL  
passengers$Name <- NULL  
passengers$Ticket <- NULL  
passengers$Cabin <- NULL

## Valores nulos

Ahora procederemos a tratar los valores nulos. Gracias a una inspeccion “manual” de los datos se a conseguido identificar valores nulos en el campo Age de tipo NA y valores nulos en el campo Fare de tipo “0”.

Aplicamos el siguiente comando que nos ayudara a contabilizar los valores nulos en cada atributo:

#Encuentra valos NA  
colSums(is.na(passengers))

## Survived Pclass Sex Age SibSp Parch Fare Embarked   
## 0 0 0 177 0 0 0 0

#Encuentra valores de tipo numeric igual a 0  
colSums(passengers == 0)

## Survived Pclass Sex Age SibSp Parch Fare Embarked   
## 549 0 0 NA 608 678 15 0

Con la funcion *is.na* podmeos observar que el data set cuenta con varios datos vacios o nulos en el atributo Age (edad). Debido al alto numero de valores nulos y a su alta impacto en el analisis, vamos a revisar de manera detenida el atributo.

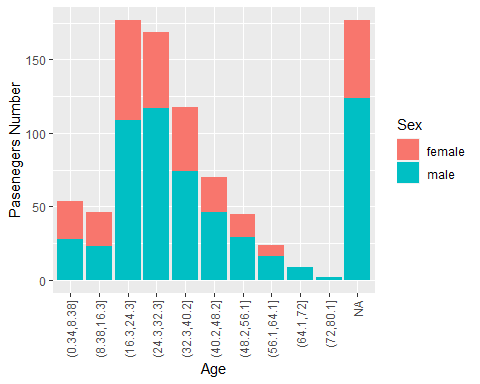
Con el siguiente comando, vamos a ordenar los valores que toma el atributo basado en el numero de apariciones en el data set.

sort(table(passengers$Age), decreasing = TRUE)

##   
## 24 22 18 19 28 30 21 25 36 29 26 27 32 35 16 31   
## 30 27 26 25 25 25 24 23 22 20 18 18 18 18 17 17   
## 20 23 33 34 39 17 40 42 45 38 2 4 50 44 47 48   
## 15 15 15 15 14 13 13 13 12 11 10 10 10 9 9 9   
## 9 54 1 51 3 14 37 41 49 52 15 43 58 5 8 11   
## 8 8 7 7 6 6 6 6 6 6 5 5 5 4 4 4   
## 56 60 62 6 7 46 61 65 0.75 0.83 10 13 28.5 30.5 32.5 40.5   
## 4 4 4 3 3 3 3 3 2 2 2 2 2 2 2 2   
## 45.5 55 57 59 63 64 70 71 0.42 0.67 0.92 12 14.5 20.5 23.5 24.5   
## 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1   
## 34.5 36.5 53 55.5 66 70.5 74 80   
## 1 1 1 1 1 1 1 1

La primera fila representa los valores con mayor frecuencia, aqui se encuentran valores en el rango de 20 a 30 años aproximadamente. Para poder visualizar la infomracion de manera grafica, cargamos las librearias necesarias y procedemos a plotear la informacion con el siguiente comando:

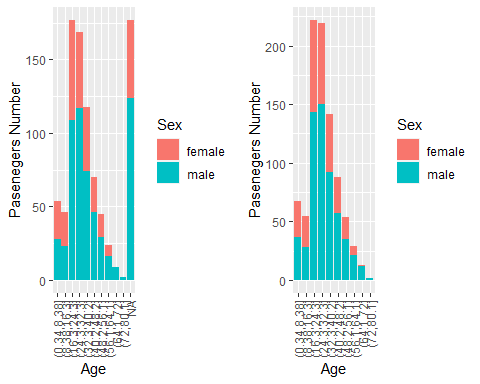
ggplot(passengers, aes(cut(Age,10), fill = Sex)) + geom\_bar()+ guides(x = guide\_axis(angle = 90))+labs(x="Age", y="Pasenegers Number")

 Como se puede observar en el gráfico, la mayor parte de pasajeros se encuentra distribuido en edades que van desde los 16.3 años hasta los 32.3. Ademas, podemos apreciar que el numero de hombres y mujeres mantienen una relacion de proporcion en todos los intervalos. Por esta razon, se procede a llenar los datos faltantes del atibuto edad con valores aleatorios obtenidos del mismo data set, sin distincion si es hombre o mujer. Para esto se ejecuta el siguiente comando:

#Se fija una semilla para que el trabajo se pueda repetir sin alterar los resultados  
set.seed(873465)  
#Guardamos el atributo edad para comparar la distribucion una vez que los valores nulos sean reemplazados.  
tmpAge <- passengers$Age  
#Creamos una variable que contenga todas las edades de los pasajeros   
tmp <- passengers$Age[!is.na(passengers$Age)]   
#Reemplazamos los valores nulos con un valor aleatorio de nuestra variable tmp.  
passengers$Age[is.na(passengers$Age)] <- sample(tmp, size = 177, replace = FALSE)

Una vez reemplazados los valores procedemos a graficarlos para comparar los resultados y verificar si hubo alguna alteracion significativa en la distribucion de datos.

plotAfter <- ggplot(passengers, aes(cut(Age,10), fill = Sex)) + geom\_bar()+ guides(x = guide\_axis(angle = 90))+labs(x="Age", y="Pasenegers Number")  
plotBefore <- ggplot(passengers, aes(cut(tmpAge,10), fill = Sex)) + geom\_bar()+ guides(x = guide\_axis(angle = 90))+labs(x="Age", y="Pasenegers Number")  
grid.arrange(plotBefore,plotAfter,ncol=2)

 Como se puede observar en el grafico derecho, despues de la asignacion de valores, ya no existen valores nulos. Ademas, a pesar del alto numero de valores nulos con el metodo aplicado la diferencia en la distribucion de edades es minima.

Ahora procedemos a analizar el atributo fare. Como se calculó en el apartado anterior, existen 15 valores del atributo fare que son 0. Para reemplazar este valor, vamos a calcular la mediana de los demas registros.

tmpFare <- passengers$Fare[passengers$Fare !=0]  
summary(tmpFare)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 4.013 7.925 14.500 32.756 31.275 512.329

passengers$Fare[passengers$Fare==0] <- median(tmpFare)  
summary(passengers$Fare)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 4.013 7.925 14.500 32.448 31.000 512.329

Como se puede observar, el reemplazo de valores apenas afecta la media global del data set

## Normalizacion de datos

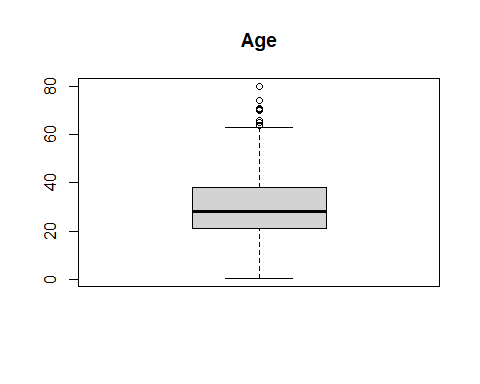
Ahora procedemos a normalizar los atributos. Como se pudo observar en el apartado anterior, solo el atributo fare es de tipo numeric, que se normalizará con la finalidad de eliminar o disminuir grandes cambios debido a la diferencia entre valores del mismo atributo. Para esto realizamos la siguiente operacion:

passengers$Fare <- scale(passengers$Fare)

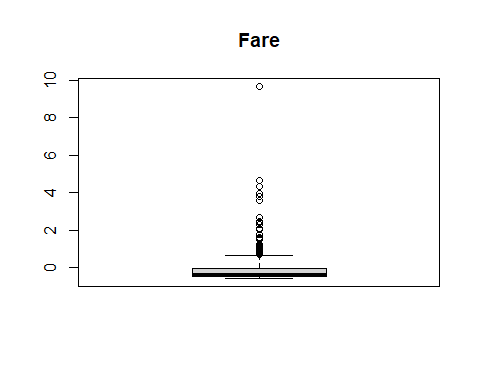
## Outliers

En esta etapa trataremos de identificar outliers en nuestro data set. A continuacionanalizamos los atributos age y fare ya que los demos atributos son de tipo categorico. Para esto creamos un boxplot para los atributos mencionados.

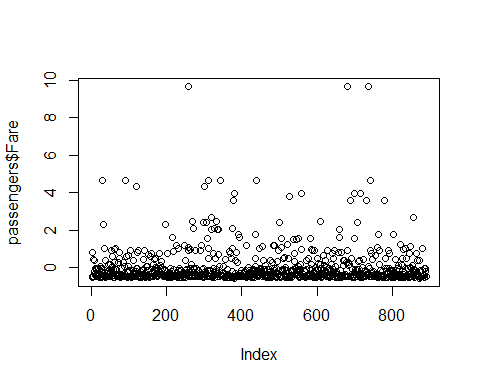
AgeOutliers<-boxplot(passengers$Age,main="Age")



FareOutliers<-boxplot(passengers$Fare,main="Fare")

 En el caso de el atributo Age podemos observar que, se considera como outlier a las personas mayores a 65 años aproximadamente. Esto se debe a que, como se menciono anteriormente, la mayoria de pasajerons tenia una edad entre 20-30 años. En este caso no se realizara ninguna operacion ya que esta informacion es real y muy importante para el analisis. Por otro lado, con el atributo Fare si podemos distinguir claramente 3 grupos de outliers que a priori se pueden considerar normales. Para visualizar los registros de mejor manera ploteamos este atributo de la siguiente manera:

plot(passengers$Fare)



A pesar de que no se tiene evidencia para asumir que los valores son reales, alterados o erroneos, procederemos a eliminarlos. Esto se lleva acabo con la finalidad de eliminar 3 registros que alteran la distribucion de los datos y por su cantidad su eliminacion no representa una perdida significante de informacion. Para eliminar los outliers procedemos a aplicar lo siguiente:

#cargamos la libreria dplyr  
library(dplyr)

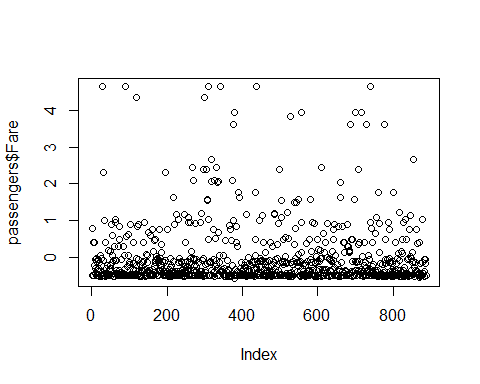
##   
## Attaching package: 'dplyr'

## The following object is masked from 'package:gridExtra':  
##   
## combine

## The following objects are masked from 'package:stats':  
##   
## filter, lag

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

#Reemplazamos nuestro dataset, donde ahora se filtraran los valores que no cumplan la condicion (los 3 registros mencionados anteriormente)  
passengers <- filter(passengers, Fare<8)  
#Ploteamos el atributo Fare del data set temporal  
plot(passengers$Fare)



# Analisis de componentes

Ahora que tenemos un data set “limpio”, procederemos a analizar los atributos para determinar los componentes principales con la funcion *prcomp*.

library(tidyverse)

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

## v tibble 3.1.0 v purrr 0.3.4  
## v tidyr 1.1.3 v stringr 1.4.0  
## v readr 1.4.0 v forcats 0.5.1

## Warning: package 'readr' was built under R version 4.0.5

## -- Conflicts ------------------------------------------ tidyverse\_conflicts() --  
## x dplyr::combine() masks gridExtra::combine()  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

passengers.pca <- prcomp(passengers[,c(1,2,4,5,6,7)],na.rn=TRUE, scale = TRUE)

## Warning: In prcomp.default(passengers[, c(1, 2, 4, 5, 6, 7)], na.rn = TRUE,   
## scale = TRUE) :  
## extra argument 'na.rn' will be disregarded

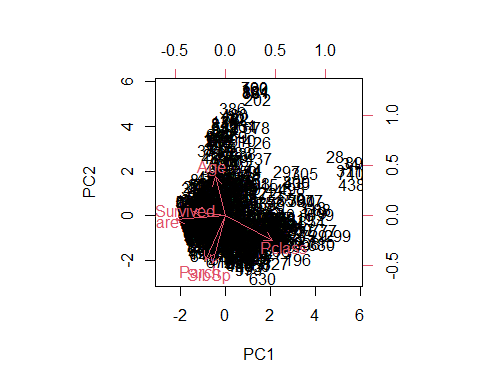
passengers.pca $rotation <- -1\*passengers.pca $rotation  
passengers.pca

## Standard deviations (1, .., p=6):  
## [1] 1.3833458 1.2730747 0.9837499 0.8054998 0.7427308 0.5453368  
##   
## Rotation (n x k) = (6 x 6):  
## PC1 PC2 PC3 PC4 PC5 PC6  
## Survived -0.3994202 0.05652932 -0.71529564 0.4382507 0.34369807 0.1242096  
## Pclass 0.5860537 -0.30656422 -0.05145768 0.2406666 0.06246848 0.7057541  
## Age -0.1318453 0.48760386 0.59950362 0.5308577 0.27981854 0.1592054  
## SibSp -0.1603597 -0.59882902 0.27659492 -0.1400612 0.71042569 -0.1219098  
## Parch -0.2526336 -0.55133565 0.17218716 0.5652974 -0.50568605 -0.1651586  
## Fare -0.6245658 -0.04998088 0.14193834 -0.3592043 -0.19811076 0.6473001

Como podemos observar, en el componenete PC1 y PC6 se muestra que practicamente toda la variacion se encuentra dada por el atributo PClass. En el caso del componente PC2, PC3 y PC4 la edad provee la variacion. Para el componente PC5 la variacion se encuetra determinada por el atributo SibSp. Finalmente, podemos ver que en el PC6 ademas de la edad el atributo Fare hace varia el atributo.

Para visulizar esto resultados aplicamos el siguiente comando:

biplot(passengers.pca, scale = 0)



Como podemos observar, el atributo Pclass toma los valores mas alejados en el eje de la PC1 y por otro lado la edad en el eje Y de la PC2. De esta manera se representa el emfasis que tiene cada atributo en la PCs.