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

Autor: Eduardo Béjar Feijoó

Enero 2022

Contents

Presentación

Tipología y ciclo de vida de datos: Práctica 2 - Limpieza y análisis de datos **Autor: Eduardo Béjar Feijoó Fecha: Enero 2022**

Descripción de la Práctica a realizar

El objetivo de esta actividad será el tratamiento de un dataset, que puede ser el creado en la práctica 1 o bien cualquier dataset libre disponible en Kaggle (https://www.kaggle.com).

Algunos ejemplos de dataset con los que podéis trabajar son:

- Red Wine Quality (https://www.kaggle.com/uciml/red-wine-quality-cortez-et-al-2009)
- Titanic: Machine Learning from Disaster (https://www.kaggle.com/c/titanic)

El último ejemplo corresponde a una competición activa de Kaggle de manera que, opcionalmente, podéis aprovechar el trabajo realizado durante la práctica para entrar en esta competición.

Siguiendo las principales etapas de un proyecto analítico, las diferentes tareas a realizar (y justificar) son las siguientes:

##1. Descripción del dataset. ¿Por qué es importante y qué pregunta/problema pretende responder?

Para esta práctica seleccioné el dataset Titanic, en base a lo sugerido en el enunciado. En Kaggle este dataset está dividido en 2, uno de entrenamiento y otro de prueba. Este dataset es importante para responder preguntas como cuál era el perfil de los pasajeros que iban en el Titanic al momento de su naufragio, o cómo predecir si un pasajero sobrevivió o no al naufragio según su perfil de edad, sexo, o clase en la que viajaba.

```
# https://cran.r-project.org/web/packages/dplyr/index.html
if (!require('dplyr')) install.packages('dplyr'); library('dplyr')

TitanicTrain <- read.csv('titanic/train.csv',head(T))
TitanicTest <- read.csv('titanic/test.csv',head(T))
```

##2. Integración y selección de los datos de interés a analizar.

Analizamos la estructura de los dos datasets (train y test):

```
str(TitanicTrain)
```

```
## '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" ...
```

```
str(TitanicTest)
```

```
## '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" ...
```

Observamos que en el dataset de Train hay 12 columnas y en el de Test hay 11. Esto se debe a que este conjunto de datos es parte de la competencia de Kaggle para predecir si un pasajero sobrevivió o falleció en el percance del Titanic. De allí que en el dataset de Test no está la columna Survived.

Para poder realizar la integración vertical de los dos conjuntos de datos para trabajar esta Práctica, agregamos al dataset Test una columna survived con valor de No Disponible (NA - Not Available).

```
TitanicTest[,"Survived"] <- NA
```

Revisamos nuevamente la estructura de los conjuntos de datos:

```
str(TitanicTrain)
```

```
## '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" ...
```

```
str(TitanicTest)
```

```
## 'data.frame':    418 obs. of  12 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" ...
##  $ Survived    : logi  NA NA NA NA NA NA ...
```

Vemos que ahora ambos datasets cuentan con 12 columnas. Procedemos a integrarlos verticalmente para trabajar el análisis con un solo dataset:

```
TitanicDataset <- rbind(TitanicTrain, TitanicTest)
```

Revisamos la estructura del dataset resultante:

```
str(TitanicDataset)
```

```
## '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" ...
```

Verificamos que el dataset resultante cuenta con el total de la suma de las filas de los dos datasets originales, esto es 1.309 filas.

Para efectos del análisis de esta Práctica, seleccionamos solamente las columnas de PassengerId (Id de Pasajero), Pclass (Clase en la que viajaba el pasajero), Sex (Sexo), Age (Edad en años), y Embarked (Puerto de embarque el pasajero).

```
TitanicAnalisis <- TitanicDataset %>% select(PassengerId, Pclass, Sex, Age, Embarked)
str(TitanicAnalisis)
```

```
## 'data.frame':    1309 obs. of  5 variables:
##  $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ 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 ...
##  $ Embarked   : chr  "S" "C" "S" "S" ...
```

##3. Limpieza de los datos.

##3.1. ¿Los datos contienen ceros o elementos vacíos? ¿Cómo gestionarías cada uno de estos casos?

Revisamos si las columnas del dataset de análisis tienen ceros o elementos vacíos:

```
colSums(is.na(TitanicAnalisis))
```

```
## PassengerId      Pclass         Sex         Age    Embarked
##           0           0           0         263           0
```

```
colSums(TitanicAnalisis=="")
```

```
## PassengerId      Pclass         Sex         Age    Embarked
##           0           0           0          NA           2
```

Encontramos que para la columna Age (Edad) existen 263 filas con valor de NA (No Disponible), y para Embarked (Puerto de embarque el pasajero) existen 2 filas con valores vacíos.

Para el caso de Age y Embarked, imputamos valores en base a la similitud o diferencia del resto de valores mediante kNN (k-Nearest Neighbour):

```
if (!require('VIM')) install.packages('VIM'); library('VIM')
if (!require('Rcpp')) install.packages('Rcpp'); library('Rcpp')

suppressWarnings(suppressMessages(library(VIM)))
suppressWarnings(suppressMessages(library(Rcpp)))
TitanicAnalisis$Age <- kNN(TitanicAnalisis)$Age
```

Revisamos nuevamente si las columnas del dataset de análisis tienen ceros o elementos vacíos:

```
colSums(is.na(TitanicAnalisis))
```

```
## PassengerId      Pclass         Sex         Age    Embarked
##           0           0           0           0           0
```

```
colSums(TitanicAnalisis=="")
```

```
## PassengerId      Pclass         Sex         Age    Embarked
##           0           0           0           0           2
```

Observamos que ya no existen valores NA en Age. Vemos ahora los 2 valores vacíos de Embarked. Considerando que se refiere a Puertos de embarque de pasajeros y que solo son 2 valores, eliminamos esas 2 filas para trabajar con los datos.

```
TitanicAnalisis <- TitanicAnalisis %>% dplyr::filter(!(Embarked==""))

colSums(is.na(TitanicAnalisis))
```

```
## PassengerId      Pclass         Sex         Age    Embarked
##           0           0           0           0           0
```

```
colSums(TitanicAnalisis=="")
```

```
## PassengerId      Pclass         Sex         Age    Embarked
##           0           0           0           0           0
```

Vemos que ya no se reportan valores NA ni vacíos, y que contamos con 1307 filas y 5 columnas. Almacenamos este dataset resultante en un nuevo archivo CSV.

```
write.csv(TitanicAnalisis,"TitanicDatosLimpios.csv", row.names = TRUE)
```

##3.2. Identificación y tratamiento de valores extremos.

Revisamos si existen valores extremos o outliers, en especial en la columna Age.

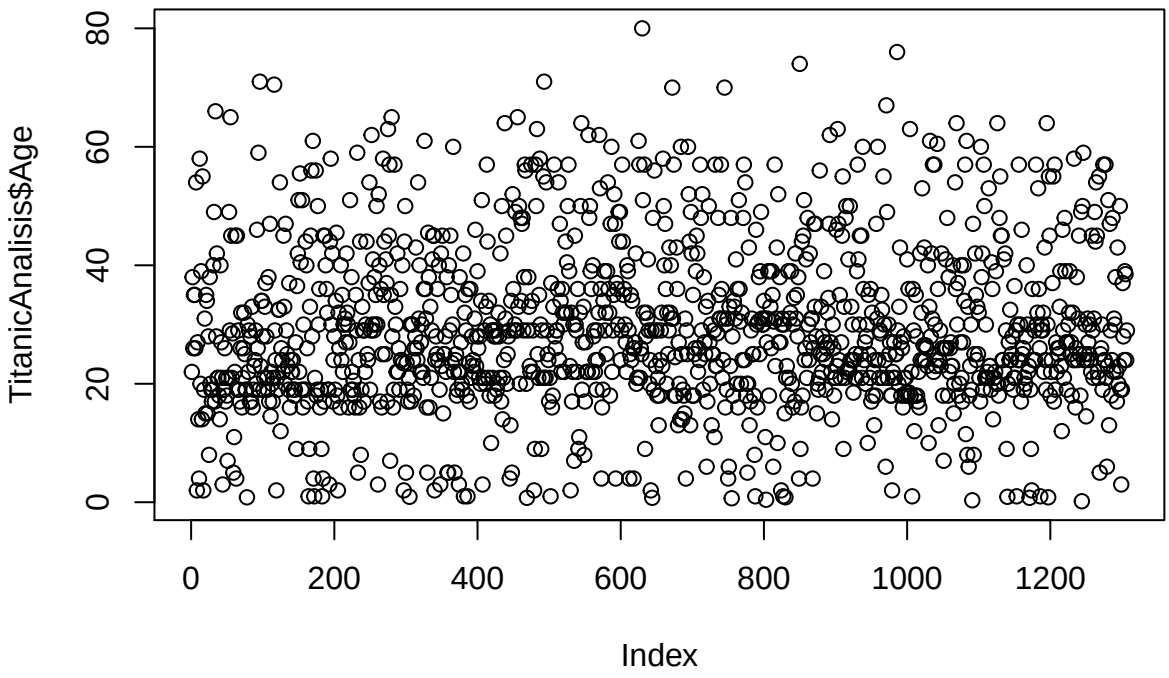
```
summary(TitanicAnalisis)
```

```
##   PassengerId         Pclass          Sex                 Age
##  Min.   :   1.0   Min.   :1.000   Length:1307        Min.   : 0.17
##  1st Qu.: 328.5   1st Qu.:2.000   Class :character   1st Qu.:21.00
##  Median : 655.0   Median :3.000   Mode  :character   Median :27.00
##  Mean   : 655.3   Mean   :2.297                      Mean   :29.30
##  3rd Qu.: 982.5   3rd Qu.:3.000                      3rd Qu.:36.75
##  Max.   :1309.0   Max.   :3.000                      Max.   :80.00
##    Embarked
##  Length:1307
##  Class :character
##  Mode  :character
##
##
##
```

Observamos que en Age el valor máximo es de 80, lo cual es un valor posible para edad, por lo que no se identifican valores extremos u outliers para tratar. Para el caso de PassengerId, este dato no es relevante en términos numéricos sino para identificar registros distintos. Pclass solo tiene 3 valores posibles (1, 2 y 3) y Sex y Embarked son variables categóricas.

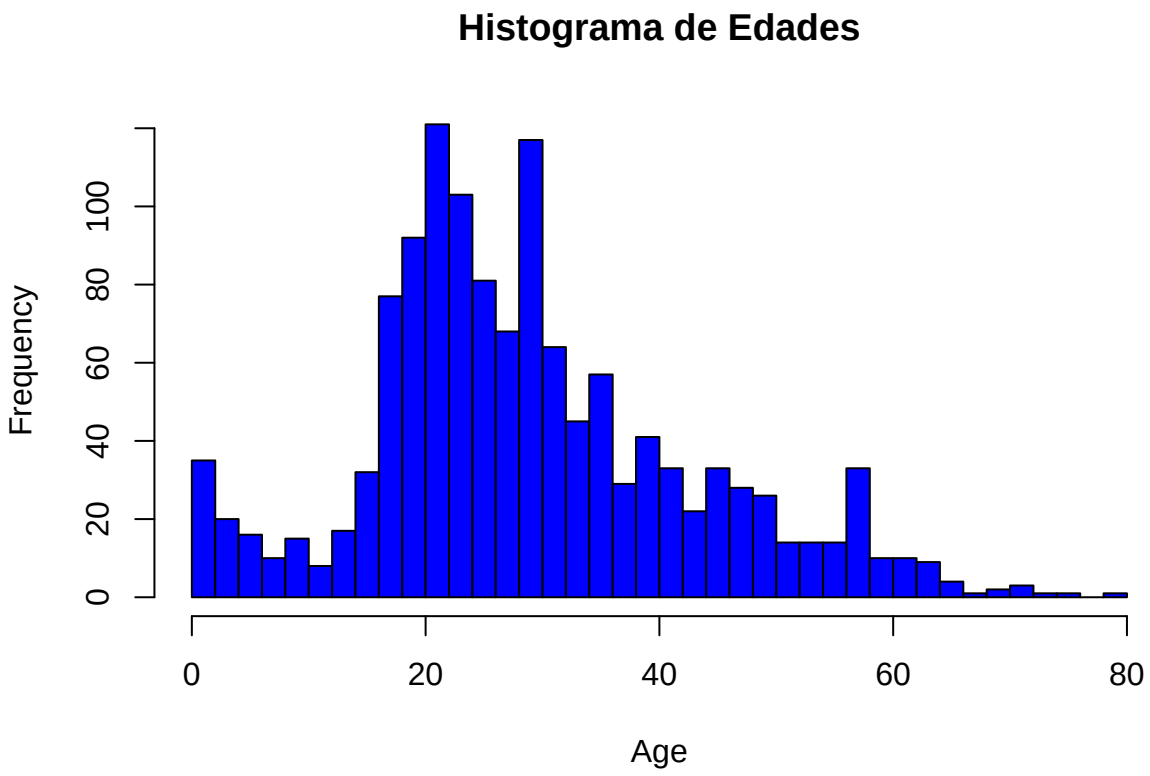
Revisamos los datos de Edad de manera gráfica:

```
plot(TitanicAnalisis$Age)
```



Comprobamos que no se observan puntos fuera del rango posible. Ahora visualizamos las edades como histograma:

```
hist(TitanicAnalisis$Age,
  xlab = "Age",
  col="blue",
  main = "Histograma de Edades",
  breaks = sqrt(nrow(TitanicAnalisis))
)
```



De igual manera, comprobamos que no se observan puntos fuera del rango posible.

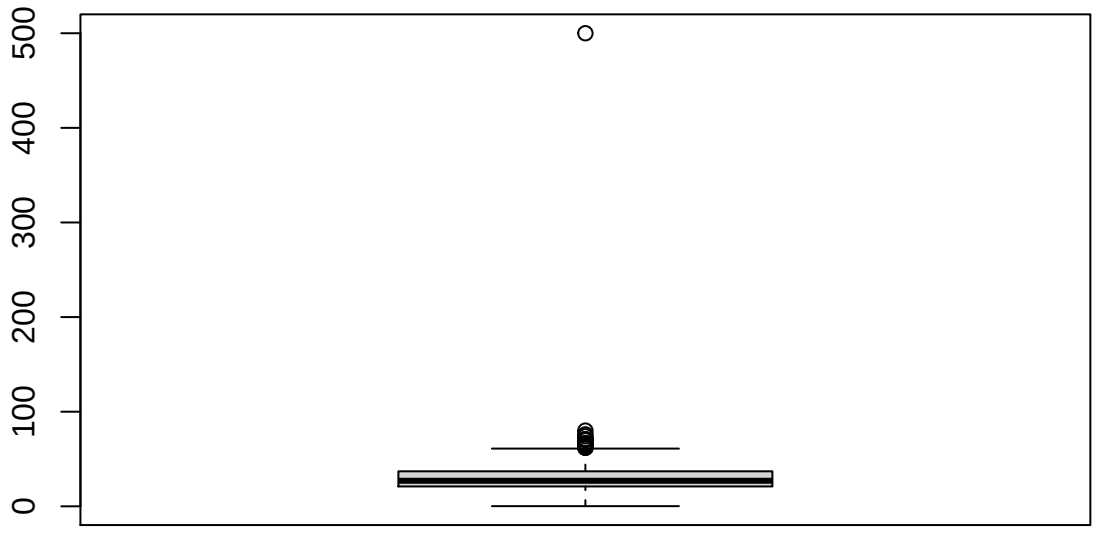
Solamente para efectos académicos de la Práctica y sobre valores extremos, agregamos un valor al dataset con Edad de 500 para observar y probar lo que ocurre cuando se tienen valores extremos u outliers (Es decir este paso no es necesario cuando se analizan datos.):

```
if (!require('tidyverse')) install.packages('tidyverse'); library('tidyverse')

TitanicAnalisis[nrow(TitanicAnalisis) + 1,] = list(PassengerId = 1310, Pclass = 1, Sex = "M", Age=500, 

summary(TitanicAnalisis)
```

```
##   PassengerId         Pclass          Sex                 Age        
##  Min.   :   1.0   Min.   :1.000   Length:1308        Min.   :  0.17  
##  1st Qu.: 328.8   1st Qu.:2.000   Class :character   1st Qu.: 21.00  
##  Median : 655.5   Median :3.000   Mode  :character   Median : 27.00  
##  Mean   : 655.8   Mean   :2.296                      Mean   : 29.66  
##  3rd Qu.: 983.2   3rd Qu.:3.000                      3rd Qu.: 37.00  
##  Max.   :1310.0   Max.   :3.000                      Max.   :500.00  
##    Embarked        
##  Length:1308        
##  Class :character   
##  Mode  :character   
##                     
##                     
##
```

Está agregado el outlier de 500 en Edad. Lo vemos gráficamente:

```
boxplot(TitanicAnalisis$Age)$out
```




```
##  [1]  66.0  65.0  71.0  70.5  62.0  63.0  65.0  64.0  65.0  63.0  71.0  64.0
## [13]  62.0  62.0  80.0  70.0  70.0  74.0  62.0  63.0  67.0  76.0  63.0  64.0
## [25]  64.0  64.0 500.0
```

Apreciamos en el gráfico de caja, en la parte superior el valor extremo de 500. También se observan ciertos valores que constan fuera de la caja y que podrían ser considerados también como outliers; sin embargo, se trata de valores de edad de hasta 80 años que son perfectamente posibles.

Para ilustrarlo, utilizamos la función stats de boxplot para encontrar los outliers que se identifican:

```
boxplot.stats(TitanicAnalisis$Age)$out
```

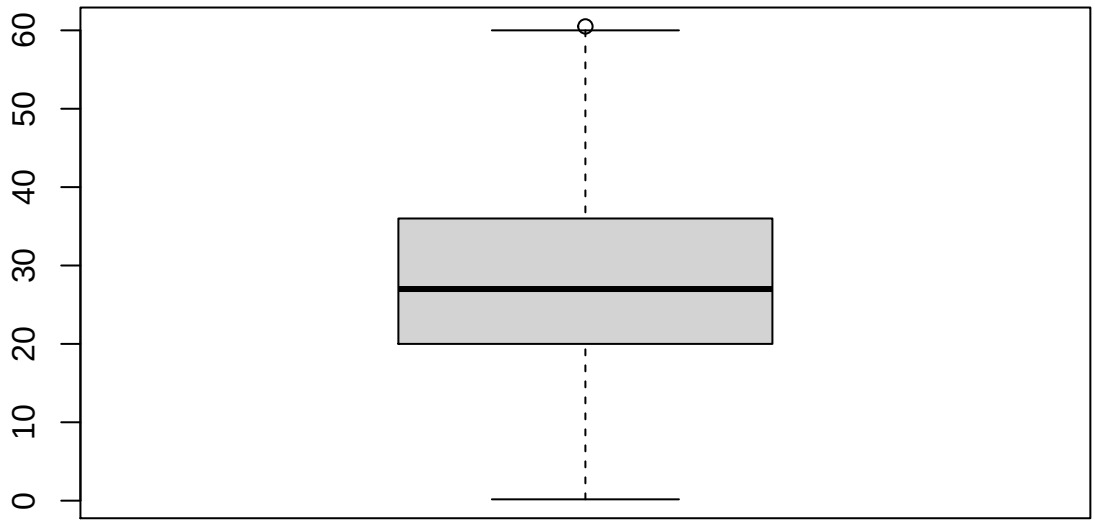
```
##  [1]  66.0  65.0  71.0  70.5  62.0  63.0  65.0  64.0  65.0  63.0  71.0  64.0
## [13]  62.0  62.0  80.0  70.0  70.0  74.0  62.0  63.0  67.0  76.0  63.0  64.0
## [25]  64.0  64.0 500.0
```

Para eliminar los outliers se pueden aplicar varias técnicas como la distancia de Mahalanobis o la distancia de Cook. Probamos con la distancia de Mahalanobis:

```r
#Calculo de distancia Mahalanobis para eliminar outliers
Q1 <- quantile(TitanicAnalisis$Age, .25)
Q3 <- quantile(TitanicAnalisis$Age, .75)
IQR <- IQR(TitanicAnalisis$Age)

#Creo nuevo dataset sin outliers
TitanicSinOutliers <- subset(TitanicAnalisis, TitanicAnalisis$Age > (Q1 - 1.5*IQR) & TitanicAnalisis$Age
dim(TitanicSinOutliers)
```

```
## [1] 1276    5
```

```r
boxplot(TitanicSinOutliers$Age)
```



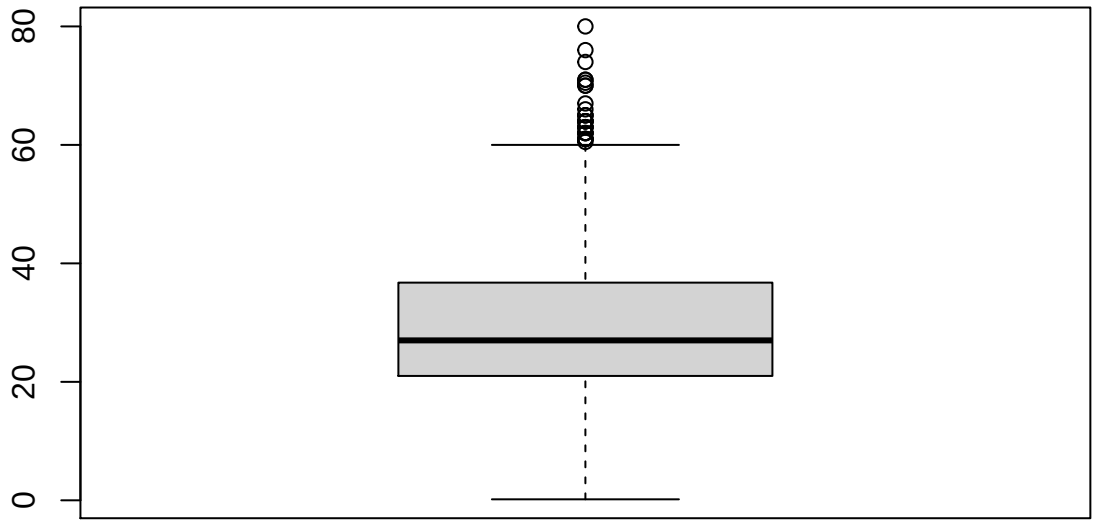
De esta manera se habrían eliminado los valores extremos del dataset. Considerando que no vamos a eliminar los valores que se detectaron como extremos, debido a que son edades perfectamente posibles, solamente eliminamos el valor que agregamos de Edad 500 para probar, y continuar analizando el conjunto de datos:

```r
TitanicAnalisis<-subset(TitanicAnalisis, PassengerId!=1310)
summary(TitanicAnalisis)
```

```
##   PassengerId         Pclass          Sex                 Age
##  Min.   :   1.0   Min.   :1.000   Length:1307        Min.   : 0.17
##  1st Qu.: 328.5   1st Qu.:2.000   Class :character   1st Qu.:21.00
##  Median : 655.0   Median :3.000   Mode  :character   Median :27.00
```

```
##  Mean   : 655.3   Mean   :2.297                    Mean   :29.30
##  3rd Qu.: 982.5   3rd Qu.:3.000                    3rd Qu.:36.75
##  Max.   :1309.0   Max.   :3.000                    Max.   :80.00
##    Embarked
##  Length:1307
##  Class :character
##  Mode  :character
##
##
##
```

```
boxplot(TitanicAnalisis$Age)
```



Como se indicó, a pesar de que en el gráfico se observan valores fuera de la caja, reiteramos que las edades de hasta 80 años que estos representan son perfectamente posibles.

##4. Análisis de los datos.

##4.1. Selección de los grupos de datos que se quieren analizar/comparar (planificación de los análisis a aplicar).

Considerando que en el dataset de Análisis (TitanicAnalisis) se cuentan con las variables PassengerId, Pclass, Sex, Age y Embarked, realizaremos los siguientes análisis:

```
# Agrupación por clases
# 1 = 1st, 2 = 2nd, 3 = 3rd
Titanic.PrimeraClase <- TitanicAnalisis[TitanicAnalisis$Pclass == 1,]
Titanic.SegundaClase <- TitanicAnalisis[TitanicAnalisis$Pclass == 2,]
```

```
Titanic.TerceraClase <- TitanicAnalisis[TitanicAnalisis$Pclass == 3,]

# Agrupacion por Puerto de Embarque
# C = Cherbourg, Q = Queenstown, S = Southampton
Titanic.Cherbourg <- TitanicAnalisis[TitanicAnalisis$Embarked == "C",]
Titanic.Queenstown <- TitanicAnalisis[TitanicAnalisis$Embarked == "Q",]
Titanic.Southampton <- TitanicAnalisis[TitanicAnalisis$Embarked == "S",]

#Agrupacion por Sexo
Titanic.Male <- TitanicAnalisis[TitanicAnalisis$Sex == "male",]
Titanic.Female <- TitanicAnalisis[TitanicAnalisis$Sex == "female",]
```

##4.2. Comprobación de la normalidad y homogeneidad de la varianza.

Realizamos la comprobación de la normalidad, utilizando como referencia la función incluida en el Ejemplo de la Práctica provisto, mediante la prueba de normalidad Anderson-Darling.

Tal como se menciona en el Ejemplo, Si los valores de p que se obtienen son superiores al alpha definido de 0,05 entonces esa variable sigue una distribución normal.

```
if (!require('nortest')) install.packages('nortest'); library('nortest')

alpha = 0.05
col.names = colnames(TitanicAnalisis)
for (i in 1:ncol(TitanicAnalisis)) {
  if (i == 1) cat("Variables que NO siguen una distribución normal:\n")
  if (is.integer(TitanicAnalisis[,i]) | is.numeric(TitanicAnalisis[,i])) {
    p_val = ad.test(TitanicAnalisis[,i])$p.value
    if (p_val < alpha) {
      cat(col.names[i])

      # Format output
      if (i < ncol(TitanicAnalisis) - 1) cat(", ")
      if (i %% 3 == 0) cat("\n")
    }
  }
}
```

```
## Variables que NO siguen una distribución normal:
## PassengerId, Pclass, Age
```

En base al resultado obtenido se encuentra que las variables PassengerId, Pclass y Age no siguen una distribución normal. Cabe destacar que Sex y Embarked son variables categóricas.

Ahora, realizaremos la comprobación de homogeneidad de la varianza aplicando la prueba Fligner-Killeen. Siguiendo el mismo ejemplo, realizaremos el análisis utilizando las variables Age y Pclass.

```
fligner.test(Age ~ Pclass, data = TitanicAnalisis)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  Age by Pclass
## Fligner-Killeen:med chi-squared = 65.135, df = 2, p-value = 7.18e-15
```

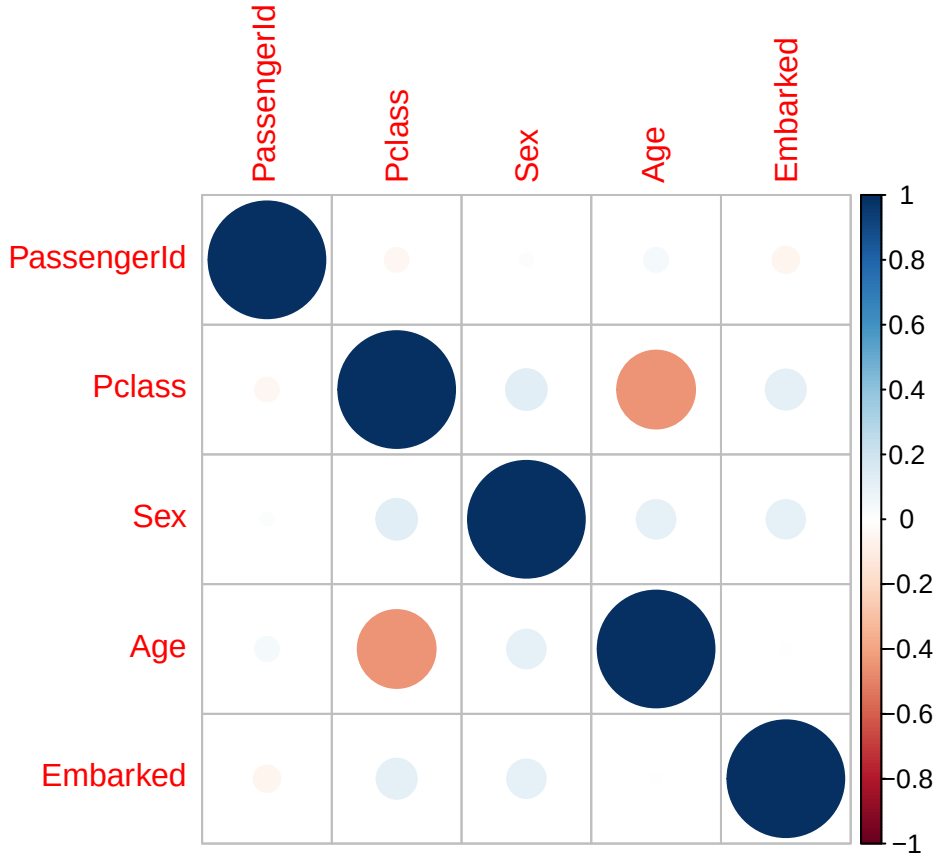
Considerando que el p-valor obtenido no es superior a 0,05 entonces las varianzas de ambas muestras no son homogéneas.

##4.3. Aplicación de pruebas estadísticas para comparar los grupos de datos. En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes.

###Correlaciones

Encontramos la matriz de correlación del dataset:

```
if (!require('corrplot')) install.packages('corrplot'); library('corrplot')

TitanicCorrelacion <- data.matrix(TitanicAnalisis)
#summary(TitanicCorrelacion)
MatrizCorrelacion <- cor(TitanicCorrelacion, method = "spearman")
corrplot(MatrizCorrelacion)
```



En base a la matriz de correlación encontramos que no existe ningun correlación que se destaque entre las variables. Solamente se aprecia una posible correlación negativa entre Edad y Clase que implicaría que mientras menor edad tienen las personas mayor el valor de su clase. En este caso significaría que en la Tercera Clase del Titanic viajaron personas más jóvenes que en la Primera Clase.

###Pruebas de contraste de hipótesis

Revisemos ahora las posibles relaciones entre variables para validar o descartar hipótesis. Considerando que en este dataset tenemos 1 variable numérica (Age, ya que no se considera PassengerId) y 3 variables categóricas (Pclass, Sex, Embarked), trabajaremos encontrando asociaciones entre las categóricas.

Para trabajar con las edades de forma categórica, las discretizamos en grupos para agruparlas para el análisis:

```
# Agregro columna de edades discretizadas
TitanicAnalisis$GruposEdad <- cut(TitanicAnalisis$Age, breaks = c(0,10,20,30,40,50,60,70,100), labels =
head(TitanicAnalisis)
```

```
##   PassengerId Pclass    Sex Age Embarked GruposEdad
## 1           1      3   male  22        S      20-29
## 2           2      1 female  38        C      30-39
## 3           3      3 female  26        S      20-29
## 4           4      1 female  35        S      30-39
## 5           5      3   male  35        S      30-39
## 6           6      3   male  26        Q      20-29
```

Ahora revisamos las hipótesis en base a relaciones:

Hipótesis 1: Las personas que se embarcaron en Southampton prefirieron viajar en Primera Clase.

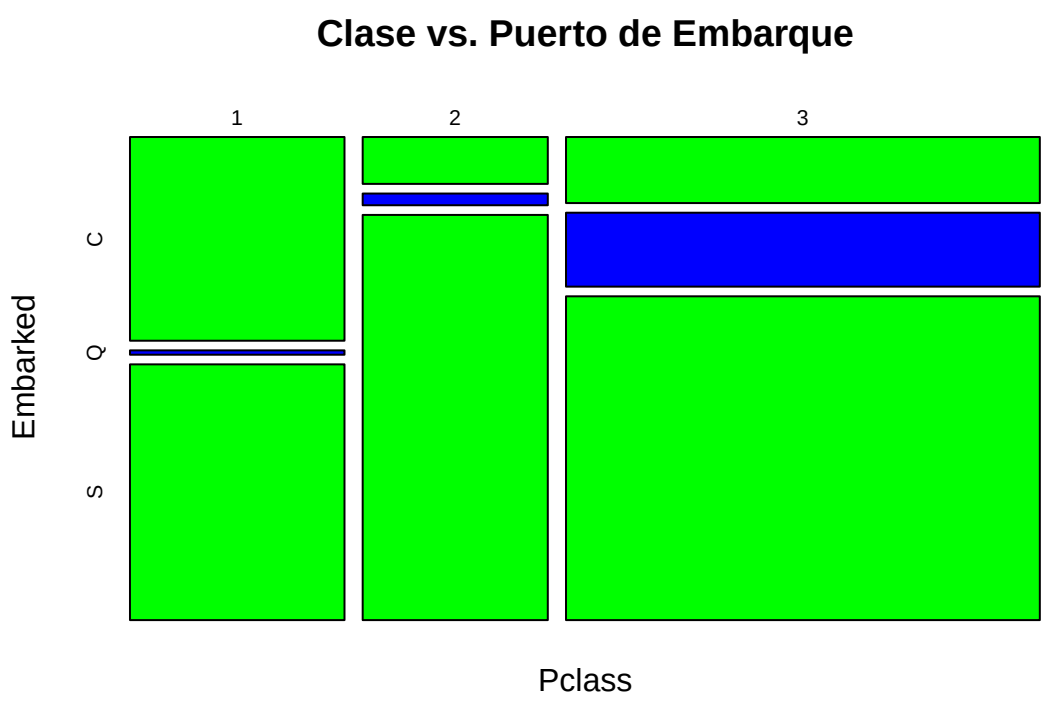
Relación entre Clase y Puerto de Embarque

```
# Tablas de contingencia
attach(TitanicAnalisis)
#table(Pclass)
#table(Sex)
#table(Embarked)

# Relación entre Pclass y Embarked
tablaPclassEmbarked <- table(Pclass,Embarked)
tablaPclassEmbarked
```

```
##        Embarked
## Pclass   C   Q   S
##      1 141   3 177
##      2  28   7 242
##      3 101 113 495
```

Ahora lo graficamos:

```
plot(tablaPclassEmbarked, col = c("green", "blue"), main = "Clase vs. Puerto de Embarque")
```



Buscamos si existe relación estadísticamente significativa entre ambas aplicando la prueba de chi cuadrado:

```
chisq.test(tablaPclassEmbarked)
```

```
##
##  Pearson's Chi-squared test
##
## data:  tablaPclassEmbarked
## X-squared = 205.79, df = 4, p-value < 2.2e-16
```

Considerando que el p-valor obtenido no es superior a 0,05 entonces rechazamos esta hipótesis de posible relación entre ambas. Incluso se observa en el gráfico que quienes embarcaron en Southampton en su mayoría viajaron en tercera clase. En conclusión, las personas que se embarcaron en Southampton NO prefirieron viajar en Primera Clase.

Hipótesis 2: Las mayoría de mujeres que viajaron en el Titanic prefirieron hacerlo en primera clase.

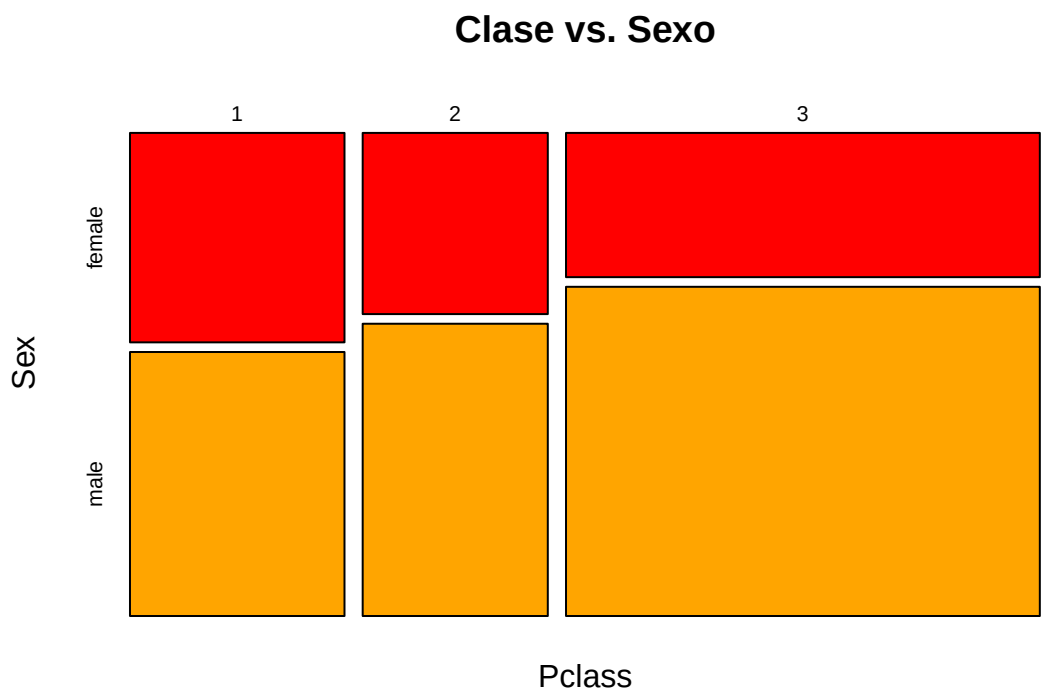
Relación entre Clase y Sexo

```
# Relación entre Pclass y Sexo
tablaPclassSex <- table(Pclass,Sex)
tablaPclassSex
```

```
##       Sex
## Pclass female male
##      1    142  179
##      2    106  171
##      3    216  493
```

Ahora lo graficamos:

```
plot(tablaPclassSex, col = c("red", "orange"), main = "Clase vs. Sexo")
```



Buscamos si existe relación estadísticamente significativa entre ambas aplicando la prueba de chi cuadrado:

```
chisq.test(tablaPclassSex)
```

```
##
##  Pearson's Chi-squared test
##
## data:  tablaPclassSex
## X-squared = 19.475, df = 2, p-value = 5.902e-05
```

Considerando que el p-valor obtenido no es superior a 0,05 entonces rechazamos esta hipótesis de posible relación entre ambas. En conclusión, la mayoría de personas que viajó en primera clase según los datos fueron hombres.

Hipótesis 3: Las mayoría de mujeres que viajaron en el Titanic se embarcaron en Queenstown.

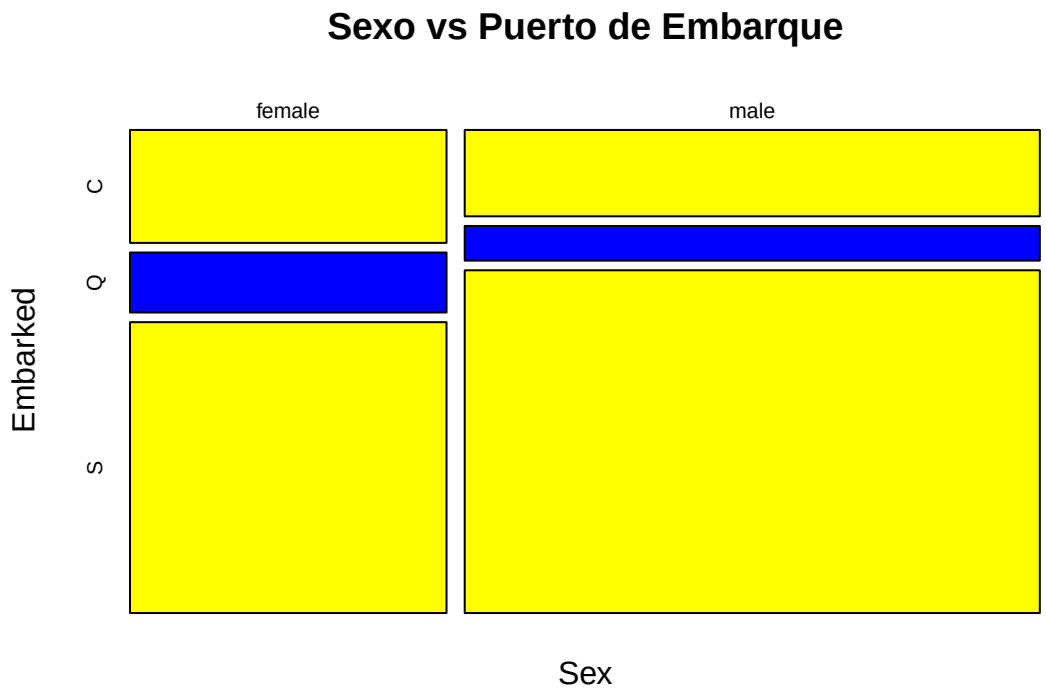
Relación entre Sexo y Puerto de Embarque

```
# Relación entre Sexo y Embarked
tablaSexEmbarked <- table(Sex,Embarked)
tablaSexEmbarked
```

```
##          Embarked
## Sex         C   Q   S
##   female  113  60 291
##   male    157  63 623
```

Ahora lo graficamos:

```
plot(tablaSexEmbarked, col = c("yellow", "blue"), main = "Sexo vs Puerto de Embarque")
```



Buscamos si existe relación estadísticamente significativa entre ambas aplicando la prueba de chi cuadrado:

```
chisq.test(tablaSexEmbarked)
```

```
##
##  Pearson's Chi-squared test
##
## data:  tablaSexEmbarked
## X-squared = 19.584, df = 2, p-value = 5.589e-05
```

Considerando que el p-valor obtenido no es superior a 0,05 entonces rechazamos esta hipótesis de posible relación entre ambas. En conclusión, la mayoría de mujeres que viajaron en el Titanic no se embarcaron en Queenstown. En la gráfica se aprecia que lo hicieron en Southampton.

Hipótesis 4: Las mayoría de personas jovenes que viajaron en el Titanic fueron mujeres.

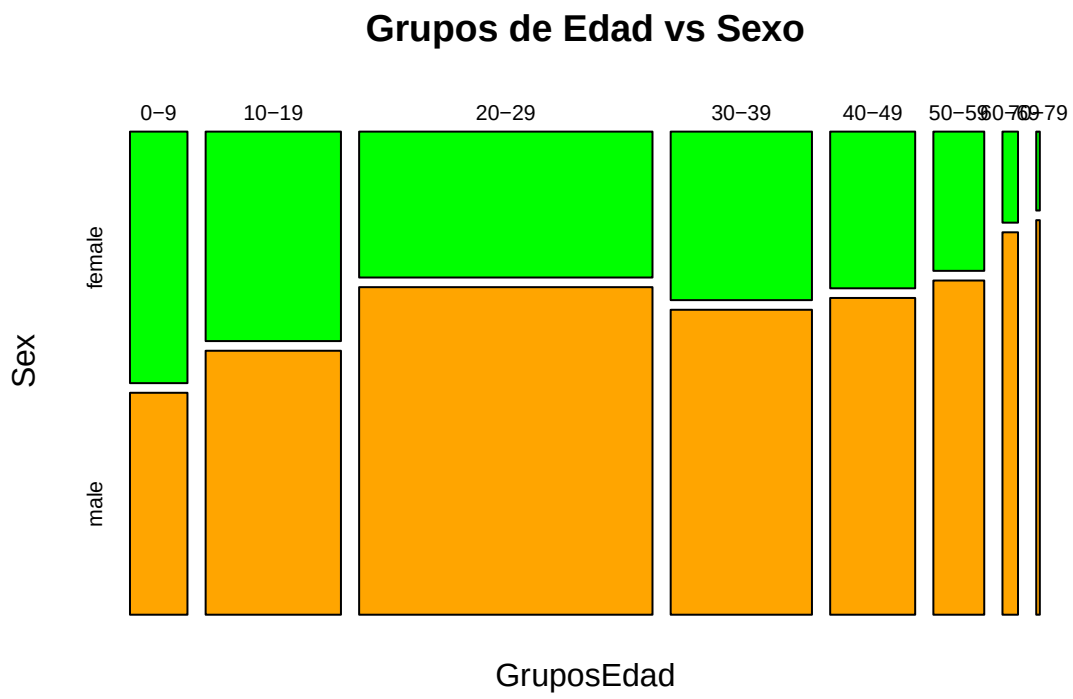
Relación entre Grupos de Edad y Sexo

```r
# Relación entre Grupos de Edad y Sexo
tablaGruposEdadSex <- table(GruposEdad,Sex)
tablaGruposEdadSex
```

```
##           Sex
## GruposEdad female male
##      0-9       51   45
##      10-19    100  126
##      20-29    151  339
##      30-39     84  152
##      40-49     47   95
##      50-59     25   60
##      60-69      5   21
##      70-79      1    5
```

Ahora lo graficamos:

```r
plot(tablaGruposEdadSex, col = c("green", "orange"), main = "Grupos de Edad vs Sexo")
```



Buscamos si existe relación estadísticamente significativa entre ambas aplicando la prueba de chi cuadrado:

```r
chisq.test(tablaGruposEdadSex)
```

```
##
```

```
##  Pearson's Chi-squared test
##
## data:  tablaGruposEdadSex
## X-squared = 30.94, df = 7, p-value = 6.377e-05
```

Considerando que el p-valor obtenido no es superior a 0,05 entonces rechazamos esta hipótesis de posible relación entre ambas. En conclusión, como se observa en la gráfica, la mayoría de personas jovenes fueron hombres.

###Regresiones

Para esta práctica realizaremos la predicción de los supervivientes para el dataset de Test, a partir de un modelo de regresión logística, al tratarse de una predicción de datos binarios (Survived 1 o 0). Comenzamos llamando nuevamente a los datasets de Train y Test:

```
TitanicTestRegresion <- read.csv('titanic/test.csv',head(T))
TitanicTrainRegresion <- read.csv('titanic/train.csv',head(T))
```

Nuevamente realizamos su limpieza y preparación:

```
print("Train - NA")
```

```
## [1] "Train - NA"
```

```
colSums(is.na(TitanicTrainRegresion))
```

```
## PassengerId    Survived      Pclass        Name         Sex         Age
##           0           0           0           0           0         177
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           0           0           0           0
```

```
print("Train - Vacíos")
```

```
## [1] "Train - Vacíos"
```

```
colSums(TitanicTrainRegresion=="")
```

```
## PassengerId    Survived      Pclass        Name         Sex         Age
##           0           0           0           0           0          NA
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           0           0         687           2
```

```
print("Test - NA")
```

```
## [1] "Test - NA"
```

```
colSums(is.na(TitanicTestRegresion))
```

```
## PassengerId      Pclass        Name         Sex         Age       SibSp
##           0           0           0           0          86           0
##       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           1           0           0
```

```
print("Test - Vacíos")
```

```
## [1] "Test - Vacíos"
```

```
colSums(TitanicTestRegresion=="")
```

```
## PassengerId      Pclass        Name         Sex         Age       SibSp
##           0           0           0           0          NA           0
##       Parch      Ticket        Fare       Cabin    Embarked
##           0           0          NA         327           0
```

Encontramos que para el caso del dataset Train, Fare tiene 1 valor NA y Age 86. Además, Cabin tiene 327 valores vacíos.

Para Test, Age tiene 177 NA y vacíos Cabin 687 y Embarked 2.

Procedemos a imputar valores a los NA y vacíos, en base a la similitud o diferencia del resto de valores mediante kNN (k-Nearest Neighbour):

```
TitanicTrainRegresion$Age <- kNN(TitanicTrainRegresion)$Age
TitanicTrainRegresion$Fare <- kNN(TitanicTrainRegresion)$Fare

TitanicTestRegresion$Age <- kNN(TitanicTestRegresion)$Age
TitanicTestRegresion$Fare <- kNN(TitanicTestRegresion)$Fare
```

Verificamos nuevamente los datasets:

```
print("Train - NA")
```

```
## [1] "Train - NA"
```

```
colSums(is.na(TitanicTrainRegresion))
```

```
## PassengerId    Survived      Pclass        Name         Sex         Age
##           0           0           0           0           0           0
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           0           0           0           0
```

```
print("Train - Vacíos")
```

```
## [1] "Train - Vacíos"
```

```
colSums(TitanicTrainRegresion=="")
```

```
## PassengerId    Survived      Pclass        Name         Sex         Age
##           0           0           0           0           0           0
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           0           0         687           2
```

```r
print("Test - NA")
```

```
## [1] "Test - NA"
```

```r
colSums(is.na(TitanicTestRegresion))
```

```
## PassengerId      Pclass        Name         Sex         Age       SibSp
##           0           0           0           0           0           0
##       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           0           0           0
```

```r
print("Test - Vacíos")
```

```
## [1] "Test - Vacíos"
```

```r
colSums(TitanicTestRegresion=="")
```

```
## PassengerId      Pclass        Name         Sex         Age       SibSp
##           0           0           0           0           0           0
##       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           0         327           0
```

Para el caso de la variables Cabin no la utilizaremos para el análisis por lo que se elimina la columna. También eliminamos Ticket que no es relevante.

Y para el caso de Embarked, que en Train tiene 2 vacíos, eliminamos esas 2 filas vacías.

```r
TitanicTrainRegresion <- TitanicTrainRegresion %>% select(-PassengerId, -Ticket, -Cabin, -Name)
TitanicTrainRegresion <- TitanicTrainRegresion %>% dplyr::filter(!(Embarked==""))

TitanicTestRegresion <- TitanicTestRegresion %>% select(-PassengerId, -Ticket, -Cabin, -Name)

print("Train - NA")
```

```
## [1] "Train - NA"
```

```r
colSums(is.na(TitanicTrainRegresion))
```

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

```r
print("Train - Vacíos")
```

```
## [1] "Train - Vacíos"
```

```r
colSums(TitanicTrainRegresion=="")
```

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

```
print("Test - NA")
```

```
## [1] "Test - NA"
```

```
colSums(is.na(TitanicTestRegresion))
```

```
##   Pclass      Sex      Age    SibSp    Parch     Fare Embarked
##        0        0        0        0        0        0        0
```

```
print("Test - Vacíos")
```

```
## [1] "Test - Vacíos"
```

```
colSums(TitanicTestRegresion=="")
```

```
##   Pclass      Sex      Age    SibSp    Parch     Fare Embarked
##        0        0        0        0        0        0        0
```

De esta manera ya tenemos los datasets de Train y Test limpios para el análisis.

Ahora transformamos en el dataset de entrenamiento las variables categóricas a factores para poderlas trabajar en el modelo:

```
TitanicTrainRegresion$Survived <- factor(TitanicTrainRegresion$Survived)
TitanicTrainRegresion$Sex <- factor(TitanicTrainRegresion$Sex)
TitanicTrainRegresion$Pclass <- factor(TitanicTrainRegresion$Pclass)
TitanicTrainRegresion$SibSp <- factor(TitanicTrainRegresion$SibSp)
TitanicTrainRegresion$Parch <- factor(TitanicTrainRegresion$Parch)
TitanicTrainRegresion$Embarked <- factor(TitanicTrainRegresion$Embarked)
str(TitanicTrainRegresion)
```

```
## 'data.frame':    889 obs. of  8 variables:
##  $ Survived: Factor w/ 2 levels "0","1": 1 2 2 2 1 1 1 1 2 2 ...
##  $ Pclass  : Factor w/ 3 levels "1","2","3": 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 28.5 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/ 7 levels "0","1","2","3",..: 1 1 1 1 1 1 1 2 3 1 ...
##  $ Fare    : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Embarked: Factor w/ 3 levels "C","Q","S": 3 1 3 3 3 2 3 3 3 1 ...
```

Ahora procedemos a entrenar el modelo con los datos del dataset de Train:

```
log.model <- glm(Survived~., family=binomial(link='logit'), data=TitanicTrainRegresion)
summary(log.model)
```

```
##
## Call:
## glm(formula = Survived ~ ., family = binomial(link = "logit"),
```

```
##     data = TitanicTrainRegresion)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9008  -0.5813  -0.3918   0.5808   2.5356
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  4.305e+00  5.193e-01   8.290  < 2e-16 ***
## Pclass2     -1.217e+00  3.128e-01  -3.891 9.99e-05 ***
## Pclass3     -2.377e+00  3.203e-01  -7.422 1.15e-13 ***
## Sexmale     -2.610e+00  2.058e-01 -12.682  < 2e-16 ***
## Age         -4.959e-02  8.483e-03  -5.846 5.05e-09 ***
## SibSp1       2.930e-02  2.274e-01   0.129  0.89751
## SibSp2      -4.073e-01  5.464e-01  -0.745  0.45599
## SibSp3      -2.349e+00  7.210e-01  -3.258  0.00112 **
## SibSp4      -1.748e+00  7.612e-01  -2.296  0.02168 *
## SibSp5      -1.598e+01  9.673e+02  -0.017  0.98682
## SibSp8      -1.622e+01  8.186e+02  -0.020  0.98419
## Parch1       3.151e-01  2.904e-01   1.085  0.27791
## Parch2      -8.242e-02  3.880e-01  -0.212  0.83179
## Parch3       4.862e-01  1.048e+00   0.464  0.64284
## Parch4      -1.574e+01  1.054e+03  -0.015  0.98808
## Parch5      -9.383e-01  1.172e+00  -0.801  0.42333
## Parch6      -1.621e+01  2.400e+03  -0.007  0.99461
## Fare         2.091e-03  2.543e-03   0.822  0.41103
## EmbarkedQ   -1.989e-01  4.054e-01  -0.491  0.62362
## EmbarkedS   -2.775e-01  2.477e-01  -1.121  0.26244
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1182.82  on 888  degrees of freedom
## Residual deviance:  748.02  on 869  degrees of freedom
## AIC: 788.02
##
## Number of Fisher Scoring iterations: 15
```

Vemos que el modelo indica que las variables Pclass2, Pclass3, Sexmale y Age (señaladas con ***) son las que más afectan al modelo.

Ahora preparamos el dataset Test, transformando también a factor las variables categóricas:

```
TitanicTestRegresion$Sex <- factor(TitanicTestRegresion$Sex)
TitanicTestRegresion$Pclass <- factor(TitanicTestRegresion$Pclass)
TitanicTestRegresion$SibSp <- factor(TitanicTestRegresion$SibSp)
TitanicTestRegresion$Parch <- factor(TitanicTestRegresion$Parch)
TitanicTestRegresion$Embarked <- factor(TitanicTestRegresion$Embarked)
str(TitanicTestRegresion)
```

```
## 'data.frame':    418 obs. of  7 variables:
##  $ Pclass  : Factor w/ 3 levels "1","2","3": 3 3 2 3 3 3 3 2 3 3 ...
##  $ Sex     : Factor w/ 2 levels "female","male": 2 1 2 2 1 2 1 2 1 2 ...
```

```
##  $ Age     : num  34.5 47 62 27 22 14 30 26 18 21 ...
##  $ SibSp   : Factor w/ 7 levels "0","1","2","3",..: 1 2 1 1 2 1 1 2 1 3 ...
##  $ Parch   : Factor w/ 8 levels "0","1","2","3",..: 1 1 1 1 2 1 1 2 1 1 ...
##  $ Fare    : num  7.83 7 9.69 8.66 12.29 ...
##  $ Embarked: Factor w/ 3 levels "C","Q","S": 2 3 2 3 3 3 2 3 1 3 ...
```

En las pruebas iniciales de esta Práctica, la prueba del modelo retornó error debido a que la variable Parch en el dataset de Test cuenta con un nivel adicional (Train$Parch tiene 7 y Test$Parch tiene 8). Revisando en Excel el dataset encontramos que solo son 2 registros que tienen Parch con nivel 9, por lo que procedemos a eliminarlos del test para esta práctica:

```
TitanicTestRegresion <- TitanicTestRegresion %>% dplyr::filter(!(Parch==9))
```

Procedemos a probar el modelo con el dataset Test:

```
fitted.probabilities <- predict(log.model, TitanicTestRegresion, type='response')
fitted.results <- ifelse(fitted.probabilities>0.5, 1,0)
TitanicTestRegresion$Survived_prediccion <- fitted.results
head(TitanicTestRegresion)
```

```
##   Pclass    Sex  Age SibSp Parch    Fare Embarked Survived_prediccion
## 1      3   male 34.5     0     0  7.8292        Q                   0
## 2      3 female 47.0     1     0  7.0000        S                   0
## 3      2   male 62.0     0     0  9.6875        Q                   0
## 4      3   male 27.0     0     0  8.6625        S                   0
## 5      3 female 22.0     1     1 12.2875        S                   1
## 6      3   male 14.0     0     0  9.2250        S                   0
```

Vemos que en el dataset de Test se agregó la columna Survived_prediccion con la predicción de si sobrevivió o no el pasajero.

Ahora realizamos cross-validation del modelo utilizando 10-fold cross-validation:

```
if (!require('caret')) install.packages('caret'); library('caret')
if (!require('naivebayes')) install.packages('naivebayescaret'); library('naivebayes')

set.seed(100)
trctrl <- trainControl(method = "cv", number = 10, savePredictions=TRUE)

nb_fit <- train(factor(Survived) ~., data = TitanicTrainRegresion, method = "naive_bayes", trControl=tr
nb_fit
```

```
## Naive Bayes
##
## 889 samples
##   7 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 800, 800, 800, 800, 800, 801, ...
## Resampling results across tuning parameters:
```

```
##
##   usekernel  Accuracy   Kappa
##   FALSE      0.4004724  0.0184075
##    TRUE      0.6175434  0.0000000
##
## Tuning parameter 'laplace' was held constant at a value of 0
## Tuning
##  parameter 'adjust' was held constant at a value of 1
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were laplace = 0, usekernel = TRUE
##  and adjust = 1.
```

A partir del 10-fold cross-validation encontramos que el modelo tiene una precisión del 61,75%.

Ahora guardamos la predicción en un archivo CSV:

```
write.csv(TitanicTestRegresion,"TitanicPrediccion.csv", row.names = TRUE)
```

##5. Representación de los resultados a partir de tablas y gráficas.

Hemos utilizado gráficas en el desarrollo de esta práctica.

##6. Resolución del problema. A partir de los resultados obtenidos, ¿cuáles son las conclusiones? ¿Los resultados permiten responder al problema?

Si bien el modelo predice si las personas del dataset de Test sobrevivieron o no al naufragio del Titanic, el que tenga 61% de precisión hace que no sea tan confiable y que se requiera probar con otros modelos. Por otro lado, en el caso de las relaciones los resultados son más claros ya que se pudo validar o rechazar a las hipótesis planteadas.

##7. Código: Hay que adjuntar el código, preferiblemente en R, con el que se ha realizado la limpieza, análisis y representación de los datos. Si lo preferís, también podéis trabajar en Python.

El código se desarrolló en R desde RStudio.

Contribuciones	Firma
Investigación Previa	Eduardo Béjar
Redacción de las respuestas	Eduardo Béjar
Desarrollo de código	Eduardo Béjar