Practica 2 Tipologia y Ciclo de Vida de los Datos

Fernando Muñoz Martin y Ricardo Santos Patricio

Mayo 2021

Índice

1. Descripción del Dataset **1**
1.1. Introducción 1
1.2. Importancia y problemas para responder 2

2. Integración y selección de los datos de interés a analizar **2**
2.1. Lectura Archivo 2
2.2. Selección de datos 3

3. Limpieza de los datos **4**
3.1. Elementos vacíos 4
3.1.1. ¿Contienen los datos ceros o elementos vacíos? 4
3.1.2. ¿Cómo gestionar los casos? 6
3.2. Identificación y tratamiento de valores extremos 8

4. Análisis de los datos **11**
4.1. Selección de los grupos de datos que se quieren analizar o comparar 11
4.2. Comprobación de la normalidad y homogeneidad de la varianza. 11

1. Descripción del Dataset

```
library(dplyr)
library(caret)
library(nortest)
library(ranger)
```

1.1. Introducción

Para el desarrollo de esta práctica se ha optado por la elección del dataset: "Titanic: Machine Learning from Disaster" que se encuentra en el link: "https://www.kaggle.com/c/titanic". Este dataset contiene información relacionada con uno de los naufragios más conocidos de la historia, donde se tienen datos relativos a sus pasajeros, como edad, sexo, clase en que viajaban y, finalmente, si han consiguieron sobrevivir o no. Es un dataset cuyo uso es muy extendido para el entrenamiento de algoritmos supervisados o para árboles de decisión donde la variable objetivo es precisamente

si lograron sobrevivir o no en función de las características propias del viajero. Descripción de Columnas

Dicho conjunto de datos con 891 instancias no se puede considerar de un gran tamaño, sin embargo sí que está constituido por 12 columnas que hacen que la descripción de cada uno de los individuos sea razonablemente completa:

- **PassengerId**: id que contiene cada pasajero dentro del dataset
- **Survived**: variable que nos dice si el pasajero ha sobrevivido, valor 1, o si finalmente ha muerto, valor 0
- **Pclass**: clase en la que viajaba el pasajero
- **Name**: nombre del pasajero
- **Sex**: sexo del pasajero
- **Age**: edad del pasajero
- **SibSp**: número de hermanos y/o conyugues del pasajero a bordo;
- **Parch**: número de parientes y/o hijos/hijas del pasajero a bordo
- **Ticket**: número del ticket del pasajero
- **Fare**: precio pagado por el pasajero
- **Cabin**: cabina en la que se encontraba el pasajero
- **Embarked**: puerto de embarcación

1.2. Importancia y problemas para responder

A partir del análisis de este conjunto de datos, se pretende dar respuesta a una serie de preguntas que envuelven el accidente del titanic. En este caso, pretendemos determinar si, efectivamente, podemos decir que las mujeres y niños tenían una mayor probabilidad de haber sobrevivido o si el hecho de viajar en primera clase aportaba mayores posibilidades de supervivencia. Este análisis nos permitirá entender de qué forma afectaba la clase social o género del pasajero a la hora de decidir las condiciones sobre las que escapaban del conocido hundimiento de Titanic.

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

2.1. Lectura Archivo

El primer paso antes de poder realizar cualquier análisis consiste en la lectura del archivo o archivos de estudio. En nuestro caso, tenemos el archivo "train.csv" y leeremos sus datos a través de la función read.csv().

```
titanic_raw <- read.csv("./csv/train.csv")
```

Una vez obtenidos los datos, observamos su estructura con str() y obtenemos un resumen de los valores con summary():

```
#observamos la estructura de los datos
str(titanic_raw)
```

```
## '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
```

```
##  $ 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" ...
```

```
summary(titanic_raw)
```

```
##   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
##
```

2.2. Selección de datos

El siguiente paso, consistirá en eliminar aquellas columnas que no contengan información útil para el desarrollo de esta práctica. Entre estas columnas tenemos: • Ticket: ya que no contiene información que pueda diferenciar los pasajeros; • PassangerId: al ser simplemente un identificador del pasajero en nuestro conjunto de datos; • Cabin: ya que consiste en una variable con un gran número de valores incompletos. • Name

```
#eliminamos columnas
titanic <- subset(titanic_raw, select= -c(PassengerId, Name, Cabin, Ticket))
```

Ahora, convertiremos las variables categóricas de forma a facilitar el posterior análisis. Entre las variables que convertiremos a categóricas tenemos: Survived, Pclass, Sex y Embarked.Comprobaremos taién el número de niveles existentes.

```r
#covertimos variables categoricas
titanic$Survived <- as.factor(titanic$Survived)
titanic$Pclass <- as.factor(titanic$Pclass)
titanic$Sex <- as.factor(titanic$Sex)
titanic$Embarked <- as.factor(titanic$Embarked)

levels(titanic$Survived)
```

```
## [1] "0" "1"
```

```r
levels(titanic$Pclass)
```

```
## [1] "1" "2" "3"
```

```r
levels(titanic$Sex)
```

```
## [1] "female" "male"
```

```r
levels(titanic$Embarked)
```

```
## [1] ""  "C" "Q" "S"
```

Vemos que todas las columnas son normales a excepción de Embarked que presenta un nivel "" lo cual significara la presencia de valores vacíos.

3. Limpieza de los datos

3.1. Elementos vacíos

3.1.1. ¿Contienen los datos ceros o elementos vacíos?

Lo primero de lo que tenemos que hablar es de los ceros en nuestro dataset. Hay varias columnas en las que las que dicho valor tiene mucho sentido, luego no todo cero en nuestro enemigo.

```
* Comenzando por "Survived", un 0 es un valor FALSE indicando que no sobrevivió, luego en esta
```

- En "PClass" y en "sex" sabemos que ninguno de los valores es cero porque los hemos categorizado y podemos ver que en ninguno de los niveles aparece dicho valor, en cambio en "Embarked" sabemos que esto sí que sucede precisamente por el mismo motivo.
- Un valor 0 en "SibSp" y en "Parch" es un valor absolutamente razonable (que el pasajero no tenga hermanos o parientes a bordo), luego tampoco nos interesa buscar.
- En cambio que un ticket haya sido gratis (Fare) o que el pasajero tenga 0 años sería muy sospechoso, así que nos centraremos en estas dos columnas para la búsqueda de valores 0.

En cambio sí que queremos buscar elementos nulos en todas aquellas variables que no son categóricas, es decir en "Age", "SibSp", "Parch" y "Fare". Luego para esta segunda cuestión nos centraremos en esas.

```r
###  0's
where.ceros <- function(x){
  which(x[!is.na(x)] == 0)
  }
```

```
titanic.ceros <- lapply(titanic[,c("Fare", "Age")], where.ceros)
titanic.ceros
```

```
## $Fare
##  [1] 180 264 272 278 303 414 467 482 598 634 675 733 807 816 823
##
## $Age
## integer(0)
```

Comenzando por los ceros, podemos ver gracias a la función creada "where.ceros", que no hay ninguna edad con dicho valor, en cambio sí que vemos que muchos tickets han sido gratuitos y trataremos con ellos en el siguiente epígrafe.

```
### NA
where.na <- function(x){
  which(is.na(x) == TRUE)
}

titanic.na <- lapply(titanic[, c("Age", "SibSp", "Parch", "Fare")], where.na)
titanic.na
```

```
## $Age
##   [1]   6  18  20  27  29  30  32  33  37  43  46  47  48  49  56  65  66  77
##  [19]  78  83  88  96 102 108 110 122 127 129 141 155 159 160 167 169 177 181
##  [37] 182 186 187 197 199 202 215 224 230 236 241 242 251 257 261 265 271 275
##  [55] 278 285 296 299 301 302 304 305 307 325 331 335 336 348 352 355 359 360
##  [73] 365 368 369 376 385 389 410 411 412 414 416 421 426 429 432 445 452 455
##  [91] 458 460 465 467 469 471 476 482 486 491 496 498 503 508 512 518 523 525
## [109] 528 532 534 539 548 553 558 561 564 565 569 574 579 585 590 594 597 599
## [127] 602 603 612 613 614 630 634 640 644 649 651 654 657 668 670 675 681 693
## [145] 698 710 712 719 728 733 739 740 741 761 767 769 774 777 779 784 791 793
## [163] 794 816 826 827 829 833 838 840 847 850 860 864 869 879 889
##
## $SibSp
## integer(0)
##
## $Parch
## integer(0)
##
## $Fare
## integer(0)
```

En cuanto a los valores NA, vemos que estos sólo se encuentran en la variable Age y además son extremadamente habituales, suponiendo 177 de los 891 registros de los que disponemos, algo que sin duda afectará a la decisión que decidamos tomar en el siguiente epígrafe. Tan sólo nos queda por hablar de la variable "Embarked", que como mencionábamos presenta un nivel vacío en el que imputaremos NAs que posteriormente habrá que tratar.

3.1.2. ¿Cómo gestionar los casos?

Comenzando por los ceros, nos gustaría saber cómo se distribuyen los registros según la clase antes de tomar una decisión, así que antes de tomar una decisión vamos a observar dicha condición.

```
age.ceros <- unlist(titanic.ceros[1])

ceros.class <- titanic$Pclass[age.ceros]
table(ceros.class)
```

```
## ceros.class
## 1 2 3
## 5 6 4
```

Habíamos valorado la posibilidad de que el billete gratis fuera algún tipo de beneficio de alguna de las clases y que estuviéramos eliminando esta información, pero dado que se distribuyen casi homogéneamente por clase (5, 6 y 4), no parece que el valor pueda ser tomado por cierto. Dado que el precio de un ticket es algo que está fuertemente marcado por la clase, procedemos a imputar el precio medio condicionado a la clase en cada uno de esos ceros.

```
fareMean.byClass <- by(titanic$Fare, titanic$Pclass, mean)

titanic$Fare[age.ceros[ceros.class==1]] <- fareMean.byClass[1]
titanic$Fare[age.ceros[ceros.class==2]] <- fareMean.byClass[2]
titanic$Fare[age.ceros[ceros.class==3]] <- fareMean.byClass[3]

lapply(titanic[,c("Fare", "Age")], where.ceros) # Comprobamos que ha funcionado
```

```
## $Fare
## integer(0)
##
## $Age
## integer(0)
```

El caso de los valores NA en edad es bastante complejo. Son demasiados registros como para eliminarlos, pero a su vez no podemos imputar un valor único dado que con semejante volumen estaríamos distorsionando los resultados de eventuales análisis. Por lo que vamos a imputar los valores mediante el paquete caret, que puede funcionar realmente bien en este tipo de conjuntos. Para realizar la imputación, vamos a considerar todas las variables de las que disponemos en la creación de un modelo que nos ayudará a predecir valores posibles para esos NA. Posteriormente podremos comprobar si los resultados son razonables a partir de una comparación de las distribuciones de los registros que presentaban NAs y aquellos que no.

```
### NA's
age.na <- unlist(titanic.na[1])

predicted_age <- train(
  Age ~ Pclass + Sex + SibSp + Parch + Fare + Embarked + SibSp + Survived,
  data = titanic[-age.na, ],
  method = "ranger",
  trControl = trainControl(
```

```
    method = "cv", number = 10, verboseIter = TRUE),
  importance = 'impurity'
)
```

```
titanic$Age[age.na] <- predict(predicted_age, titanic[age.na,])

summary(titanic$Age[age.na])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   6.857  25.297  28.965  29.339  35.986  53.159
```

```
summary(titanic$Age[-age.na])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.42   20.12   28.00   29.70   38.00   80.00
```

En el caso de "Embarked" vamos a proceder de forma parecida al primer punto. Por lógica el valor de la puerta de entrada va a estar fuertemente influenciado por el precio del ticket y la clase del pasajero, así que vamos a ver a través de que puerta embarcaron aquellos con una situación similar a los registros 62 y 830 (aquellos faltantes).

```
embarked.na <- which(titanic$Embarked == "")
titanic$Embarked[embarked.na] <- NA

titanic[embarked.na, c("Pclass", "Fare")]
```

```
##     Pclass Fare
## 62       1   80
## 830      1   80
```

```
#filtramos por clase y puerta de embarque
titanic.C <- titanic[titanic$Embarked =="C" & titanic$Pclass =="1",]
titanic.Q <- titanic[titanic$Embarked =="Q" & titanic$Pclass =="1",]
titanic.S <- titanic[titanic$Embarked =="S" & titanic$Pclass =="1",]

median(titanic.C$Fare, na.rm = TRUE)
```

```
## [1] 78.2667
```

```
median(titanic.Q$Fare, na.rm = TRUE)
```

```
## [1] 90
```

```
median(titanic.S$Fare, na.rm = TRUE)
```

```
## [1] 53.1
```

Resulta que aquellos que pagaron de media 80 libras y estaban en primera clase entraron por la puerta C de forma clara, así que vamos a imputar este valor para nuestros registros faltantes y a recalcular los niveles del factor para eliminar el " ".

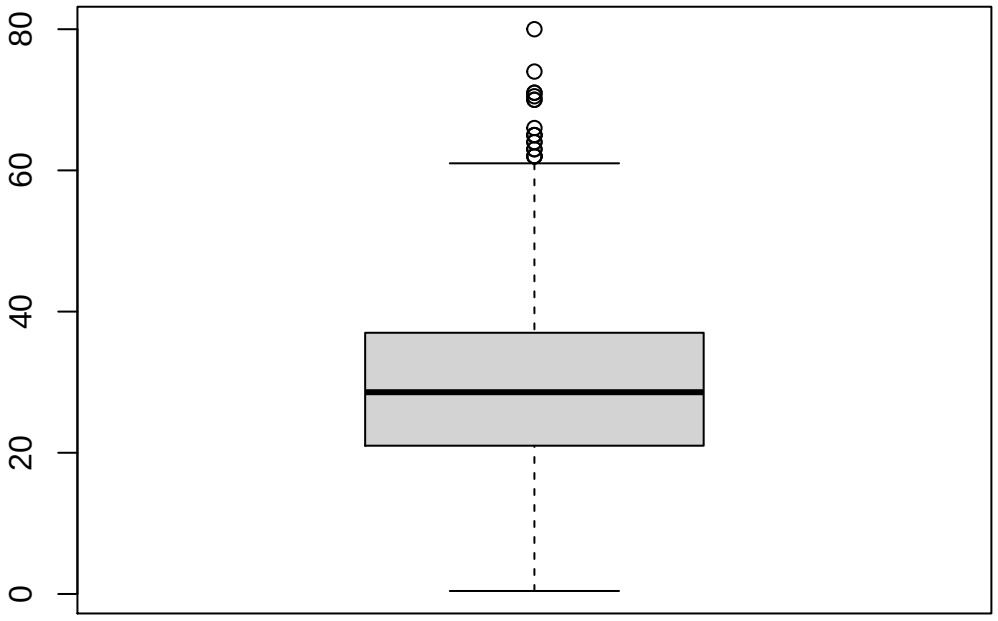
```
titanic$Embarked[embarked.na] <- "C"
levels(titanic$Embarked) <- factor(titanic$Embarked)
```

3.2. Identificación y tratamiento de valores extremos

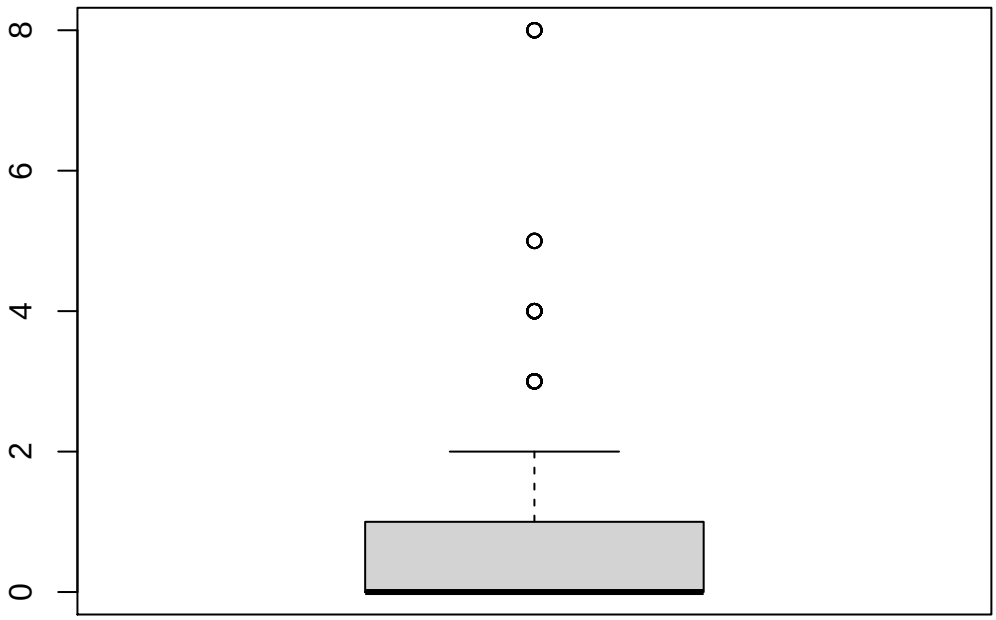
De nuevo vamos a comenzar por plantearnos qué dimensiones pueden presentar valores extremos, que en este caso sólo son las numéricas, es decir: “Age”, “SibSp”, “Fare” y “Parch”.

Antes de proceder a una imputación debemos comprobar cualitativamente si efectivamente estos valores se pueden considerar outliers, para lo que procedemos a observar los diagramas de caja correspondientes a estas dimensiones obteniendo, además, el número de outliers en cada variable.

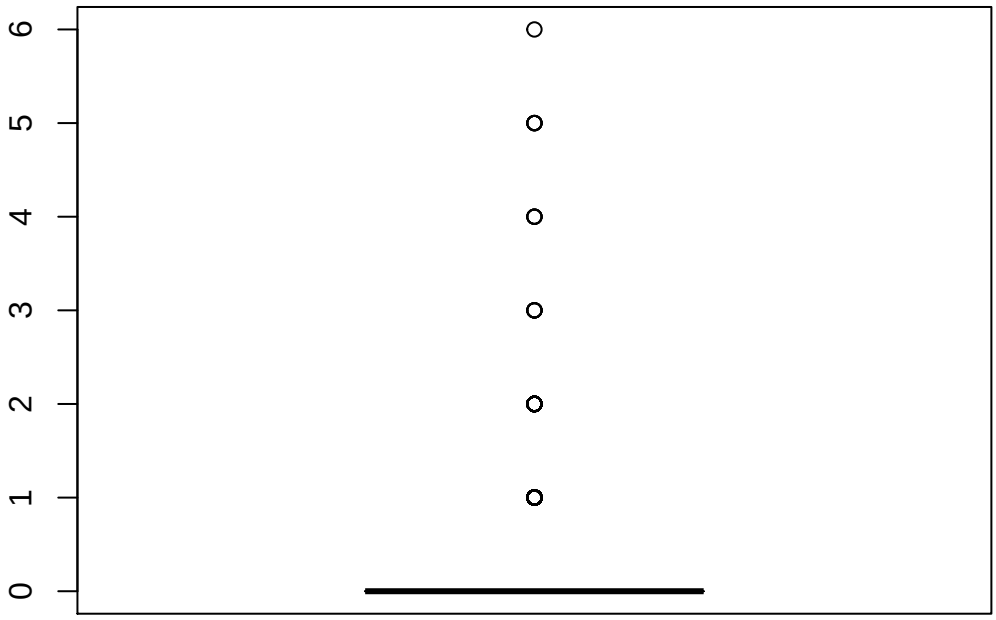
```
detect.outliers <- function(x){
  iqr <- quantile(x)
  lower.iqr <- iqr[2]
  upper.iqr <- iqr[4]
  x.iqr <- upper.iqr - lower.iqr

  upper.threshold <- (x.iqr*1.5) + upper.iqr
  lower.threshold <- lower.iqr - (x.iqr*1.5)

  values <- x < lower.threshold | x > upper.threshold
  return(values)
}

out.Age <- boxplot(titanic[,"Age"])$out
```



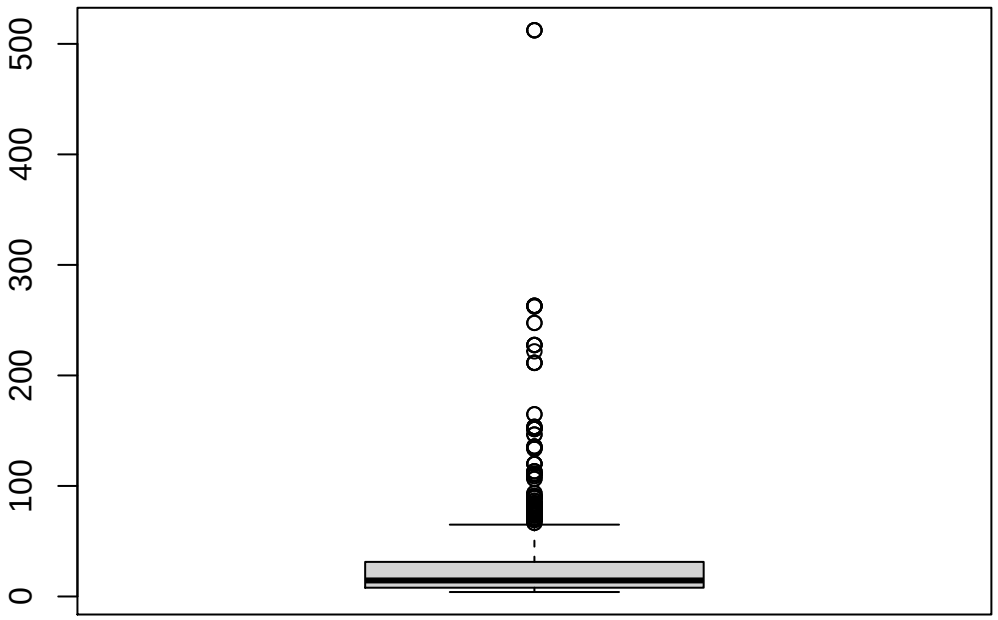

```
out.SibSp <- boxplot(titanic[,"SibSp"])$out
```




```
out.Parch <- boxplot(titanic[,"Parch"])$out
```




```
out.Fare <- boxplot(titanic[,"Fare"])$out
```




```
length(out.Age)
```

```
## [1] 19
```

```
length(out.SibSp)
```

```
## [1] 46
```

```
length(out.Parch)
```

```
## [1] 213
```

```
length(out.Fare)
```

```
## [1] 121
```

Vemos que efectivamente el dataset presenta un gran número de outliers.

Además, se podría decir que todos los valores extremos quizá a excepción de uno tienen sentido. Tenemos que alguien ha pagado más de 500 libras por una habitación cuando el segundo valor más grande era de menos de 300.

```
which(titanic$Fare > 500)
```

```
## [1] 259 680 738
```

```
titanic_raw[c(259, 680, 738), ]
```

```
##     PassengerId Survived Pclass                                Name    Sex Age
## 259         259        1      1                   Ward, Miss. Anna female  35
## 680         680        1      1 Cardeza, Mr. Thomas Drake Martinez   male  36
## 738         738        1      1             Lesurer, Mr. Gustave J   male  35
##     SibSp Parch   Ticket     Fare       Cabin Embarked
## 259     0     0 PC 17755 512.3292                   C
## 680     0     1 PC 17755 512.3292 B51 B53 B55       C
```

```
## 738      0      0 PC 17755 512.3292        B101        C
```

Viendo esta circunstancia y la más que segura existencia de información online, hemos hecho una pequeña investigación sobre dicha habitación y resulta que su valor es correcto dado que era una suite triple en la que Thomas Cardeza convivió con su madre y varios sirvientes . Debido a estas observaciones, no parece que vaya a ser necesario tratar los outliers.

4. Análisis de los datos

4.1. Selección de los grupos de datos que se quieren analizar o comparar

Todos conocemos la mítica frase "mujeres y niños primero" gracias a la película homónima, lo que no se decía tan claro en la película es que los ricos también iban primero. Está claro que uno de las selecciones tiene que ser una división de supervivientes y fallecidos, la segunda tiene que ser por sexo y la tercera será por clase. Podríamos categorizar la edad para separar también por grupos, pero corremos el riesgo de perder mucha información al establecer franjas de edad así que no lo vamos a hacer.

```
titanic.primeraClase <- titanic[titanic$Pclass == 1,]
titanic.segundaClase <- titanic[titanic$Pclass == 3,]
titanic.terceraClase <- titanic[titanic$Pclass == 3,]

titanic.mujeres <- titanic[titanic$Sex == "female",]
titanic.hombres <- titanic[titanic$Sex == "male",]

titanic.supervivientes <- titanic[titanic$Survived == 1,]
titanic.fallecidos <- titanic[titanic$Survived == 0,]
```

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

Vamos a comprobar la normalidad utilizando el test de shapiro-wilk. Este test consiste en un contraste de hipótesis en el que la hipótesis nula es la distribución normal de los datos.

```
lapply(titanic[,c("Age", "SibSp", "Parch", "Fare")], ad.test)
```

```
## $Age
##
##  Anderson-Darling normality test
##
## data:  X[[i]]
## A = 4.1836, p-value = 2.099e-10
##
##
## $SibSp
##
##  Anderson-Darling normality test
##
## data:  X[[i]]
## A = 147.36, p-value < 2.2e-16
##
```

```
##
## $Parch
##
##  Anderson-Darling normality test
##
## data:  X[[i]]
## A = 175.66, p-value < 2.2e-16
##
##
## $Fare
##
##  Anderson-Darling normality test
##
## data:  X[[i]]
## A = 121.35, p-value < 2.2e-16
```

Dado que en nuestro caso los valores p resultantes del análisis para todos los conjuntos numéricos es menor que 0.05, tenemos que rechazar la hipótesis nula y por tanto no se puede confirmar la normalidad de los datos.