Tipología y ciclo de vida. PRACTICA 2: LIMPIEZA Y VALIDACIÓN DE LOS DATOS

Edison Marcelo Muzo Oyana

1 de June, 2019

Contents

1 Descripción del dataset

```
# Lectura de datos de entrenamiento y prueba.
data_path <- 'titanic'
train_file <- 'train.csv'
test_file <- 'test.csv'
gender_file <- 'gender_submission.csv'

train_data <- read.csv(paste(data_path, train_file, sep="/"), header = TRUE, stringsAsFactors = FALSE)
test_data <- read.csv(paste(data_path, test_file, sep="/"), header = TRUE, stringsAsFactors = FALSE)
#gender_data <- read.csv(paste(data_path, gender_file, sep="/"), header = TRUE, stringsAsFactors = FALS

# Conjunto de datos completo.
full_data  <- bind_rows(train_data, test_data) # bind training & test data
```

El conjunto de datos objeto de análisis se ha obtenido a partir Titanic que contiene datos sobre la supervivencia de pasajeros abordo del Titanic. Los datos se han dividido en dos grupos:

1. El conjunto de datos de entrenamiento (train.csv). Está constituido por 891 características (columnas) que presentan 12 pasajeros (filas o registros).
2. El conjunto de datos de pruebas (test.csv). Está constituido por 418 características (columnas) que presentan 11 pasajeros (filas o registros).

También se incluye un conjunto de predicciones (gender_submission.csv) que asumen que todos y solo las pasajeras mujeres sobreviven.

Los campos de este conjunto de datos son los siguientes:

Nombre de la Variable	Descripción	Valores
Survived	Survived (1) or died (0)	Survived (1) or died (0)
Pclass	Clase del Pasajero	1 = 1st, 2 = 2nd, 3 = 3rd
Name	Nombre del Pasajero	Caracteres
Sex	Sexo del Pasajero	female or male
Age	Edad del Pasajero	Numérico
SibSp	Número de hermanos / cónyuges a bordo	Numérico
Parch	Número de padres / hijos a bordo	Numérico
Ticket	Número del Ticket	Caracteres
Fare	Tarifa	Caracteres
Cabin	Cabina	Caracteres
Embarked	Puerto de embarque	C = Cherbourg, Q = Queenstown, S = Southampton

Para este trabajo se utilizan los **conjunto de datos entrenamiento** y **conjunto de datos pruebas** como un solo conjunto de datos. Por tanto, este conjunto de datos contiene 1309 registros y 12 características

Del análisis de los ficheros train.csv y test.csv podemos extraer la siguiente información:

1. Las columnas tienen nombres (nombres de las variables).
2. El separador de columnas es el carácter ***coma*** (,).
3. Las cadenas de caracteres están delimitadas por el carácter ***comilla doble*** (").
4. Algunas cadenas de caracteres tienen espacios en blanco al inicio y/o al final.
5. Los valores decimales tienen el separador decimal ***punto*** (.).
6. El resto de las columnas parecen ser números.

2.2 Importancia y objetivos de los análisis.

A partir de este conjunto de datos se plantea la problemática de determinar qué variables influyeron más sobre la supervivencia de los pasajeros abordo del Titanic. Además, se podrá proceder a crear modelos de aprendizaje automático que permitan predecir la supervivencia de una persona 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.

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

En primer lugar, inspeccionamos el conjunto de datos sin ningún tipo de pre-procesamiento, para ello se utiliza la función **sr()**.

```
# Visualizamos los datos cargados
str(full_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" ...
```

De este conjunto de datos extraemos las siguientes conclusiones:

1. La característica `PassengerId` se puede eliminar del conjunto de datos ya que no contribuye a la supervivencia.
2. La característica `Ticket` también se puede eliminar del conjunto de datos ya que no parece contribuir a la supervivencia.
3. De la característica `Name` se puede extraer el título (por ejemplo, 'Miss', 'Mrs', etc) y el apellido de la familia y pueden aportar información adicional para determinar la supervivencia.
4. De la característica `Cabina` se pueden crear grupos según la letra inicial de la cabina y pueden aportar información adicional para determinar la supervivencia. En los casos que un valor tenga múltiples cabinas a proiri parecen compartir la misma letra y solo cambia el número de cambina, así que también nos quedamos con la primera letra.
5. De las características `SibSp` y `Parch` se puede combinar para obtener el tamaño de la familia y puede aportar información adicional para determinar la supervivencia.

El resto de características (`Pclass`,`Sex`,`Age`,`SibSp`,`Parch`,`Fare` y `Embarked`) del conjunto de datos serán considerados durante la realización de los análisis .

3. Limpieza de los datos

El conjunto de datos (train + test) contiene 1309 registros y 12 variables. Los nombres de las características son: PassengerId, Survived, Pclass, Name, Sex, Age, SibSp, Parch, Ticket, Fare, Cabin, Embarked. Antes de comenzar con la tarea de la limpieza de los datos vamos a idetificar los **tipos de datos de variables**, para ello se puede usar las funciones `str()` o `glimpse()`. Para mostar esta información en forma de tabla que facilita el análisis, se utiliza la función `sapply(dataset, class)`.

```
# Inspeccionamos la estructura del conjunto de datos
str(full_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" ...
```

```
# Inspeccionamos el conjunto de datos
glimpse(full_data)
```

```
## Observations: 1,309
## Variables: 12
## $ PassengerId <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,...
## $ Survived    <int> 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0,...
## $ Pclass      <int> 3, 1, 3, 1, 3, 3, 1, 3, 3, 2, 3, 1, 3, 3, 3, 2, 3,...
```

```
## $ Name        <chr> "Braund, Mr. Owen Harris", "Cumings, Mrs. John Bra...
## $ Sex         <chr> "male", "female", "female", "female", "male", "mal...
## $ Age         <dbl> 22, 38, 26, 35, 35, NA, 54, 2, 27, 14, 4, 58, 20, ...
## $ SibSp       <int> 1, 1, 0, 1, 0, 0, 0, 3, 0, 1, 1, 0, 0, 1, 0, 0, 4,...
## $ Parch       <int> 0, 0, 0, 0, 0, 0, 0, 1, 2, 0, 1, 0, 0, 5, 0, 0, 1,...
## $ Ticket      <chr> "A/5 21171", "PC 17599", "STON/O2. 3101282", "1138...
## $ Fare        <dbl> 7.2500, 71.2833, 7.9250, 53.1000, 8.0500, 8.4583, ...
## $ Cabin       <chr> "", "C85", "", "C123", "", "", "E46", "", "", "", ...
## $ Embarked    <chr> "S", "C", "S", "S", "S", "Q", "S", "S", "S", "C", ...
```

```
# Mostramos en forma de tabla
column_classes <- sapply(full_data, class)
data <- data.frame(Variables = names(column_classes), Clases=unname(column_classes))
kable(data) %>%
    kable_styling(bootstrap_options = "striped", full_width = F)
```

Variables	Clases
PassengerId	integer
Survived	integer
Pclass	integer
Name	character
Sex	character
Age	numeric
SibSp	integer
Parch	integer
Ticket	character
Fare	numeric
Cabin	character
Embarked	character

A la vista de los resultados anteriores se identifican las siguientes conversiones:

- La característica `Survived` debería ser un factor debido a que es cualitativa con dos valores: `1` y `0`.
- La característica `PClass` debería ser un factor debido a que es cualitativa con tres valores: `1`, `2` y `3`.
- La característica `Sex` debería ser un factor debido a que es cualitativa con dos valores: `male` y `female`.
- La característica `Embarked` debería ser un factor debido a que es cualitativa con tres valores: `C`, `Q`, y `S`. Además, hay que cambiar los valores vacios a `NA`.
- En la característica `Cabin` hay que cambiar los valores vacios a `NA`.

Además, se requiere extraer información de las siguientes características:

- De la característica `Name` se extraer el título y el apellido de la familia.
- De la característica `Cabina` se extrae el grupo de la cabina.
- De las características `SibSp` y `Parch` se combinan para obtener el tamaño de la familia.

Conversiones

En primer lugar, convertimos a factores las características `Survived`, `PClass`, `Sex` y `Embarked`. Convertimos los valores vacíos a `NA` en las características `Embarked` y `Cabin`. Finalmente, visualizalos los tipos de las características para comprobar las conmversiones.

```
# Conversion a Factores
full_data$Survived <- as.factor(full_data$Survived)
full_data$Pclass <- as.factor(full_data$Pclass)
full_data$Sex <- as.factor(str_to_upper(str_trim(full_data$Sex)))
levels(full_data$Sex)
```

```
## [1] "FEMALE" "MALE"
```

```
levels(full_data$Sex) <- c("F", "M")
full_data$Embarked <- factor(full_data$Embarked, exclude = '')


# Conversion de vacios a NA.
full_data$Cabin <- str_trim(full_data$Cabin)
full_data$Cabin[full_data$Cabin == ''] <- NA
full_data$Ticket <- str_trim(full_data$Ticket)
full_data$Ticket[full_data$Ticket == ''] <- NA


# Mostratamos el resultado de las conversiones:
str(full_data)
```

```
## 'data.frame':    1309 obs. of  12 variables:
##  $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ 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 ...
##  $ Name       : chr  "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley (Florence Briggs Thayer)"
##  $ Sex        : Factor w/ 2 levels "F","M": 2 1 1 1 2 2 2 2 1 1 ...
##  $ 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  NA "C85" NA "C123" ...
##  $ Embarked   : Factor w/ 3 levels "C","Q","S": 3 1 3 3 3 2 3 3 3 1 ...
```

```
# Visualizamos la tabla
column_classes <- sapply(full_data, class)
data <- data.frame(Variables = names(column_classes), Clases=unname(column_classes))
kable(data) %>%
    kable_styling(bootstrap_options = "striped", full_width = F)
```

Variables	Clases
PassengerId	integer
Survived	factor
Pclass	factor
Name	character
Sex	factor
Age	numeric
SibSp	integer
Parch	integer
Ticket	character
Fare	numeric
Cabin	character
Embarked	factor

Después de estas tranformaciones tenemos las siguientes distribución de variables:

- Variables categoricas: `Survived`, `Sex`, `Embarked`, y `Pclass`.
- Variables numéricas continuas: `Age`, `Fare`.
- Varibales numéricas discretas: `SibSp`, `Parch`.

- Variables con caracteres: `Name`, `Ticket` y `Cabin`.
  - `Name`: Caracteres alfanuméricos.
  - `Ticket`: Mezcla de caracteres especiales y alfanuméricos.
  - `Cabin`: Caracteres alfanuméricos.

Característica Nombre (Name)

La variable nombre del pasajero podemos dividirla en variables significativas adicionales que pueden alimentar predicciones o ser usadas en la creación de nuevas variables adicionales. Por ejemplo, el título del pasajero está contenido dentro de la variable de nombre del pasajero (Por ejemplo, 'Mr', 'Miss') y podemos usar el apellido para representar a las familias.

```
# Grab title from passenger names
full_data$Title <- gsub('(.*, )|(\\..*)', '', full_data$Name)

# Show title counts by sex
table(full_data$Sex, full_data$Title)
```

```
##
##     Capt Col Don Dona  Dr Jonkheer Lady Major Master Miss Mlle Mme  Mr Mrs
##   F    0   0   0    1   1        0    1     0      0  260    2   1   0 197
##   M    1   4   1    0   7        1    0     2     61    0    0   0 757   0
##
##      Ms Rev Sir the Countess
##   F   2   0   0            1
##   M   0   8   1            0
```

```
# Titles with very low cell counts to be combined to "rare" level
rare_title <- c('Dona', 'Lady', 'the Countess','Capt', 'Col', 'Don',
                'Dr', 'Major', 'Rev', 'Sir', 'Jonkheer')

# Also reassign mlle, ms, and mme accordingly
full_data$Title[full_data$Title == 'Mlle']        <- 'Miss'
full_data$Title[full_data$Title == 'Ms']          <- 'Miss'
full_data$Title[full_data$Title == 'Mme']         <- 'Mrs'
full_data$Title[full_data$Title %in% rare_title]  <- 'Rare Title'

# Conversion a factor
full_data$Title <- as.factor(full_data$Title)

# Show title counts by sex again
table(full_data$Sex, full_data$Title)
```

```
##
##     Master Miss  Mr Mrs Rare Title
##   F      0  264   0 198          4
##   M     61    0 757   0         25
```

```
# Finally, grab surname from passenger name
full_data$Surname <- sapply(full_data$Name,
                      function(x) strsplit(x, split = '[,.]')[[1]][1])

# Conversion a factor
full_data$Surname <- as.factor(full_data$Surname)
```

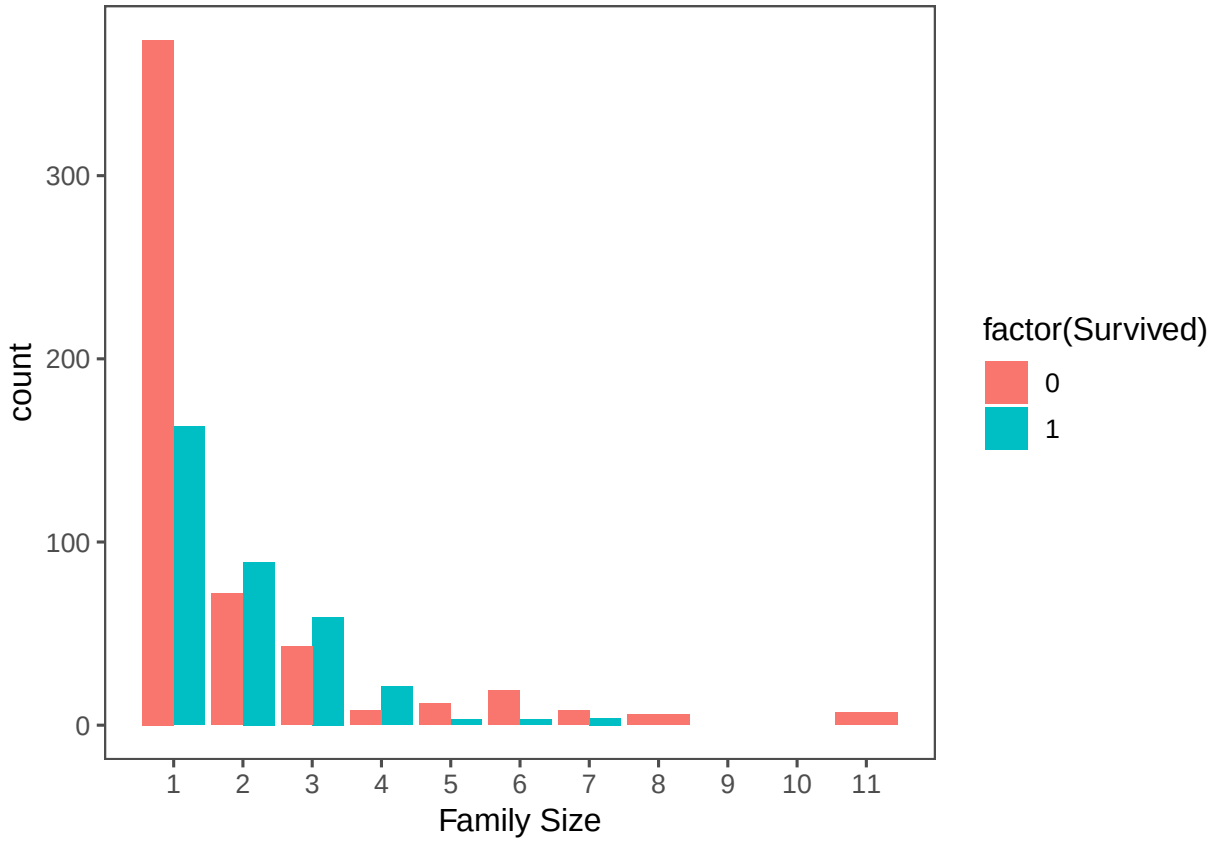
Característica Tamaño de a familia.

Podemos combinar los valores de las características `SibSp` y `Parch` para crear una característica discreta con el tamaño de la variable `FsizeD`.

```
# Create a family size variable including the passenger themselves
full_data$Fsize <- full_data$SibSp + full_data$Parch + 1
```

Visualizamos la posible relación entre el tamaño de la familia y la suervivencia.

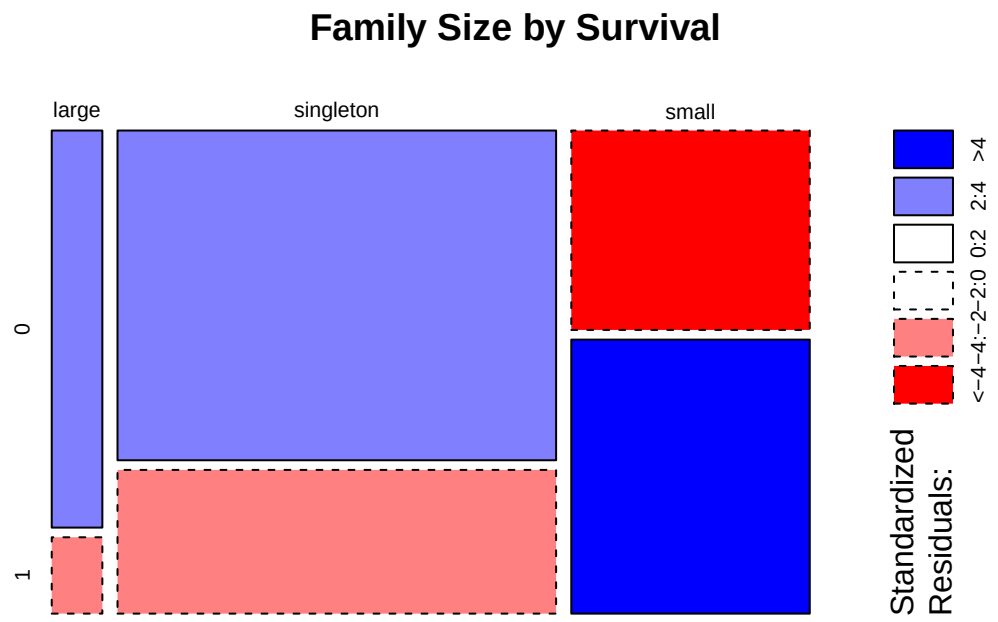
```
# Use ggplot2 to visualize the relationship between family size & survival
ggplot(full_data[1:891,], aes(x = Fsize, fill = factor(Survived))) +
  geom_bar(stat='count', position='dodge') +
  scale_x_continuous(breaks=c(1:11)) +
  labs(x = 'Family Size') +
  theme_few()
```



Dado los resultados anteriores, podemos observar que hay una penalización de supervivencia para los solteros y aquellos con un tamaño de familia superior a 4. Se puede discretizar esta variable en tres niveles, lo que será útil ya que hay comparativamente menos familias grandes.

```
# Discretize family size
full_data$FsizeD[full_data$Fsize == 1] <- 'singleton'
full_data$FsizeD[full_data$Fsize < 5 & full_data$Fsize > 1] <- 'small'
full_data$FsizeD[full_data$Fsize > 4] <- 'large'

full_data$FsizeD <- as.factor(full_data$FsizeD)

# Show family size by survival using a mosaic plot
```

```
mosaicplot(table(full_data$FsizeD, full_data$Survived), main='Family Size by Survival', shade=TRUE)
```



Característica Cabina (Cabin)

De la variable cabina (**Cabine**) podemos extrear alguna información potencialmente útil. Para ello se va discretizar esta variable según la primera letra de la cabina. Existen registros donde la cabina tiene múltiples valores pero a priori en estos casos la letra inicial de la cabina es la misma variando el número.

```
# This variable appears to have a lot of missing values
head(full_data)$Cabin
```

```
## [1] NA     "C85"  NA     "C123" NA     NA
```

```
# The first character is the deck. For example:
strsplit(full_data$Cabin[2], NULL)[[1]]
```

```
## [1] "C" "8" "5"
```

```
# Create a Deck variable. Get passenger deck A - F:
full_data$Deck<-factor(sapply(full_data$Cabin, function(x) strsplit(x, NULL)[[1]][1]))
```

Característica Ticket

De la variable cabina (**Ticket**) podemos extrear alguna información potencialmente útil. Varios pasajeros estan asociados a un ticket. Para ello se va eliminar caracteres no alfanuméricos y se tranformarán en factores sus valores.

```
# Eliminamos el punto y la barra inclinada
full_data$Ticket <- gsub('\\.|/|\\s', "", full_data$Ticket)
```

```
# Convertimos en facto
full_data$Ticket <- as.factor(full_data$Ticket)
```

3.1 Ceros o elementos vacíos

Para analizar los datos Nulo e incompletos visualizamos un resumen de los variables:

```
summary(full_data)
```

```
##   PassengerId    Survived   Pclass      Name              Sex
##  Min.   :   1   0   :549   1:323   Length:1309        F:466
##  1st Qu.: 328   1   :342   2:277   Class :character   M:843
##  Median : 655   NA's:418   3:709   Mode  :character
##  Mean   : 655
##  3rd Qu.: 982
##  Max.   :1309
##
##       Age            SibSp            Parch           Ticket
##  Min.   : 0.17   Min.   :0.0000   Min.   :0.000   CA2343 :  11
##  1st Qu.:21.00   1st Qu.:0.0000   1st Qu.:0.000   1601   :   8
##  Median :28.00   Median :0.0000   Median :0.000   CA2144 :   8
##  Mean   :29.88   Mean   :0.4989   Mean   :0.385   3101295:   7
##  3rd Qu.:39.00   3rd Qu.:1.0000   3rd Qu.:0.000   347077 :   7
##  Max.   :80.00   Max.   :8.0000   Max.   :9.000   347082 :   7
##  NA's   :263                                      (Other):1261
##       Fare            Cabin           Embarked          Title
##  Min.   :  0.000   Length:1309        C   :270   Master    : 61
##  1st Qu.:  7.896   Class :character   Q   :123   Miss      :264
##  Median : 14.454   Mode  :character   S   :914   Mr        :757
##  Mean   : 33.295                      NA's:  2   Mrs       :198
##  3rd Qu.: 31.275                                 Rare Title: 29
##  Max.   :512.329
##  NA's   :1
##       Surname         Fsize              FsizeD         Deck
##  Andersson:  11   Min.   : 1.000   large    : 82   C      :  94
##  Sage     :  11   1st Qu.: 1.000   singleton:790   B      :  65
##  Asplund  :   8   Median : 1.000   small    :437   D      :  46
##  Goodwin  :   8   Mean   : 1.884                   E      :  41
##  Davies   :   7   3rd Qu.: 2.000                   A      :  22
##  Brown    :   6   Max.   :11.000                   (Other):  27
##  (Other)  :1258                                    NA's   :1014
```

```
# Visualizar numero de nulos en las variables.
mv_colnames <- colSums(is.na(full_data))
mv_colnames <- mv_colnames[mv_colnames > 0]
data <- data.frame(Variables = names(mv_colnames), Missing=unname(mv_colnames))
kable(data) %>%
    kable_styling(bootstrap_options = "striped", full_width = F)
```

Variables	Missing
Survived	418
Age	263
Fare	1
Cabin	1014
Embarked	2
Deck	1014

Las variables de interes que tienen valores perdidos ordendas de mayor a menor son: Cabin > Age > Embarked.

Característica Embarque

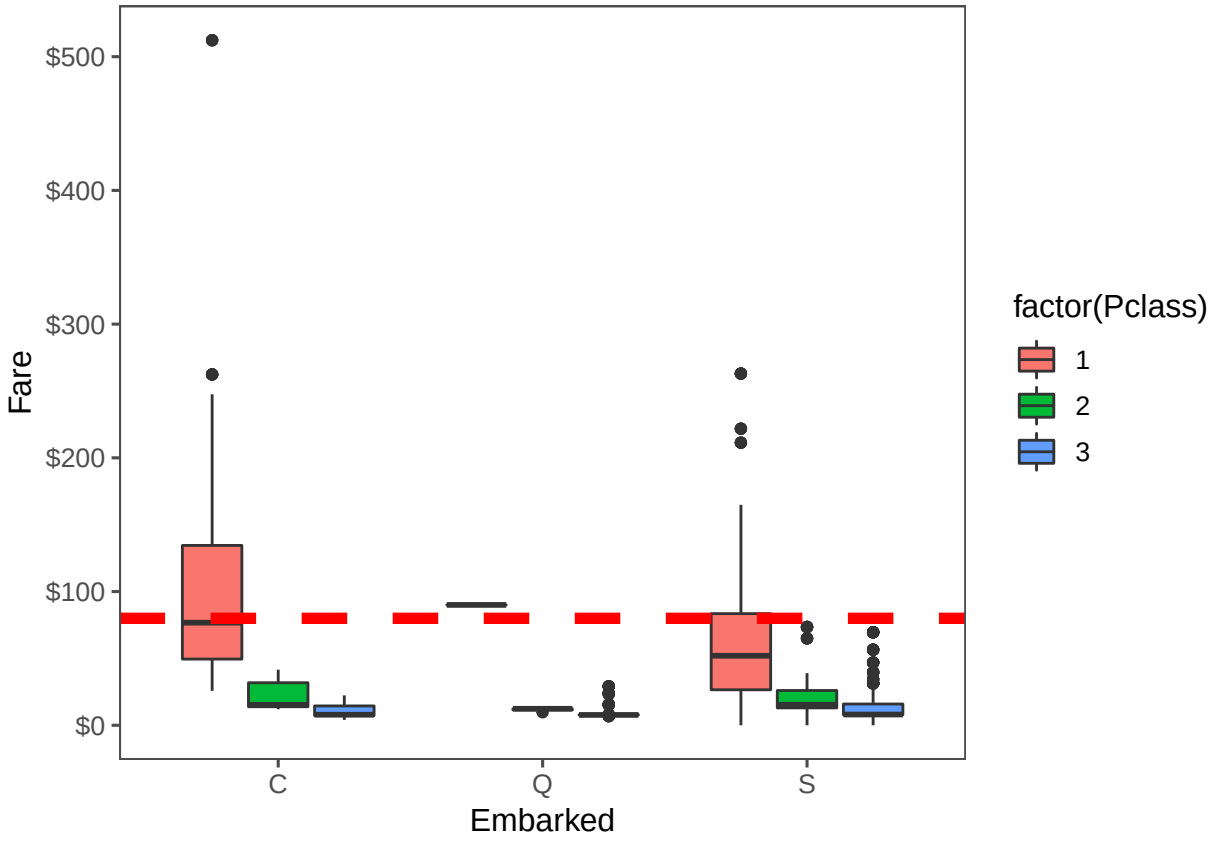
Visualizamos los datos que tienen valores perdidos en la variable Embarque (**Embarked**).

```
# Passengers 62 and 830 are missing Embarkment

miss_embark_index <- which(is.na(full_data$Embarked))
miss_embark <- full_data[miss_embark_index,]
miss_embark
```

```
##     PassengerId Survived Pclass                                     Name
## 62           62        1      1                     Icard, Miss. Amelie
## 830         830        1      1 Stone, Mrs. George Nelson (Martha Evelyn)
##     Sex Age SibSp Parch Ticket Fare Cabin Embarked Title Surname Fsize
## 62    F  38     0     0 113572   80   B28     <NA>  Miss   Icard     1
## 830   F  62     0     0 113572   80   B28     <NA>   Mrs   Stone     1
##        FsizeD Deck
## 62  singleton    B
## 830 singleton    B
```

Podemos inferir sus valores de embarque en función de los datos actuales que podamos imaginar que pueden ser relevantes: clase de pasajero y tarifa. Se observa que ambos pagaron $ 80 y estaban en la clase 1.

```
# Get rid of our missing passenger IDs
embark_fare <- full_data %>%
  filter(!is.na(Embarked))

# Use ggplot2 to visualize embarkment, passenger class, & median fare
ggplot(embark_fare, aes(x = Embarked, y = Fare, fill = factor(Pclass))) +
  geom_boxplot() +
  geom_hline(aes(yintercept=80),
    colour='red', linetype='dashed', lwd=2) +
  scale_y_continuous(labels=dollar_format()) +
  theme_few()
```



Dado los resultados anteriores, se observa que la tarifa mediana para un pasajero de 1ra clase que sale de Charbourg ('C') coincide muy bien con los $ 80 pagados por los pasajeros con valores perdidos en el embarque. Por tanto, podemos asignarles el vamore 'C'.

```
# Since their fare was $80 for 1st class, they most likely embarked from 'C'
full_data$Embarked[miss_embark_index] <- 'C'

# Comprobamos el resultado
sum(is.na(full_data$Embarked))
```

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

Característica Tarifa

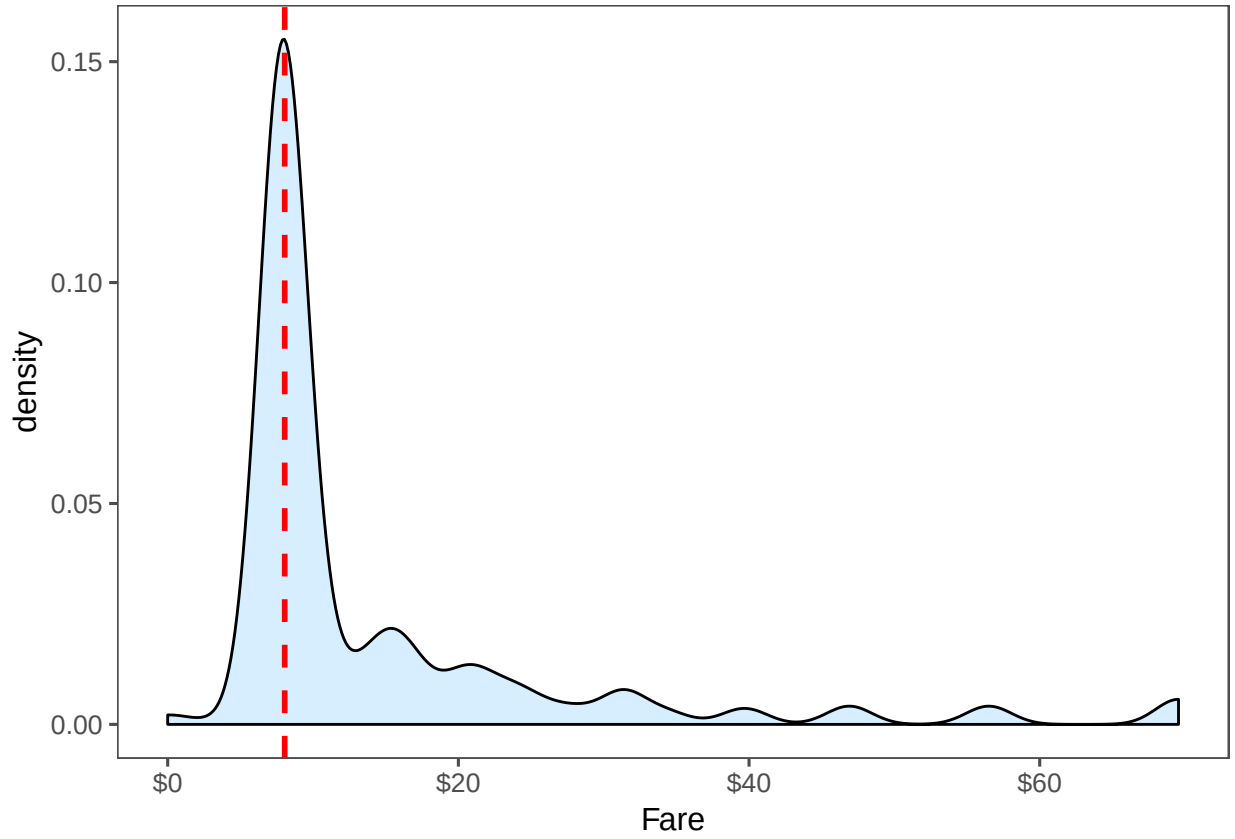
Visualizamos los datos que tienen valores perdidos en la variable Tarifa (**Fare**).

```
# Show row 1044
miss_fare_index <- which(is.na(full_data$Fare))
miss_fare <- full_data[miss_fare_index,]
miss_fare
```

```
##      PassengerId Survived Pclass               Name Sex  Age SibSp Parch
## 1044        1044     <NA>      3 Storey, Mr. Thomas   M 60.5     0     0
##      Ticket Fare Cabin Embarked Title Surname Fsize    FsizeD Deck
## 1044   3701   NA  <NA>        S    Mr  Storey     1 singleton <NA>
```

El pasajero esta asignado a la tercera clase que partió de Southampton ("S"). Visualizamos las tarifas entre todos los demás que comparten su clase y embarque (n = 495).

```
# Get rid of our missing passenger IDs
pclass_embark <- full_data %>%
  filter(Pclass == '3' & Embarked == 'S')

ggplot(pclass_embark,
  aes(x = Fare)) +
  geom_density(fill = '#99d6ff', alpha=0.4) +
  geom_vline(aes(xintercept=median(Fare, na.rm=T)),
    colour='red', linetype='dashed', lwd=1) +
  scale_x_continuous(labels=dollar_format()) +
  theme_few()
```



Dado los resultados obtenidos, parece bastante razonable reemplazar el valor perdido de la tarifa por la mediana de su clase y embarque, que es de $ 8.05.

```
# Replace missing fare value with median fare for class/embarkment
full_data$Fare[miss_fare_index] <- median(full_data[full_data$Pclass == '3' & full_data$Embarked == 'S'

# Comprobamos el resultado
sum(is.na(full_data$Fare))
```

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

```
str(full_data)
```

```
## 'data.frame':    1309 obs. of  17 variables:
##  $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ 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 ...
##  $ Name       : chr  "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley (Florence Briggs Thayer)"
##  $ Sex        : Factor w/ 2 levels "F","M": 2 1 1 1 2 2 2 2 1 1 ...
##  $ 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     : Factor w/ 929 levels "110152","110413",..: 720 816 906 66 650 374 110 542 478 175 ..
##  $ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Cabin      : chr  NA "C85" NA "C123" ...
##  $ Embarked   : Factor w/ 3 levels "C","Q","S": 3 1 3 3 3 2 3 3 3 1 ...
##  $ Title      : Factor w/ 5 levels "Master","Miss",..: 3 4 2 4 3 3 3 1 4 4 ...
##  $ Surname    : Factor w/ 875 levels "Abbing","Abbott",..: 101 183 335 273 16 544 506 614 388 565 ..
##  $ Fsize      : num  2 2 1 2 1 1 1 5 3 2 ...
##  $ FsizeD     : Factor w/ 3 levels "large","singleton",..: 3 3 2 3 2 2 2 1 3 3 ...
##  $ Deck       : Factor w/ 8 levels "A","B","C","D",..: NA 3 NA 3 NA NA 5 NA NA NA ...
```

Característica Edad

Finalmente, la variable Edad (**Age**) tiene bastantes valores perdidos. Para calcular los valores perdidos se utiliza un modelo de predicción de edades basado en otras variables.

```
"# Make variables factors into factors

# Set a random seed
set.seed(129)

# Perform mice imputation, excluding certain less-than-useful variables:
mice_mod <- mice(full_data[, !names(full_data) %in% c('PassengerId','Name','Ticket','Cabin','Family','Su

# Save the complete output
mice_output <- complete(mice_mod)"
```

```
## [1] "# Make variables factors into factors\n\n# Set a random seed\nset.seed(129)\n\n# Perform mice im
```

Comparamos los resultados de la distribución original de la edad con los del modelo.

```
"
# Plot age distributions
par(mfrow=c(1,2))
hist(full_data$Age, freq=F, main='Age: Original Data',
  col='darkgreen', ylim=c(0,0.04))
hist(mice_output$Age, freq=F, main='Age: MICE Output',
  col='lightgreen', ylim=c(0,0.04))
"
```

```
## [1] "\n# Plot age distributions\npar(mfrow=c(1,2))\nhist(full_data$Age, freq=F, main='Age: Original D
```

Dado los resultados anteriores, se observa una leve mejora en la distribución. Por tanto, se remplaza los datos originales de la edad con los obtenidos con el modelo `mice`.

```
"# Replace Age variable from the mice model.
full_data$Age <- mice_output$Age

# Show new number of missing Age values
sum(is.na(full_data$Age))"
```

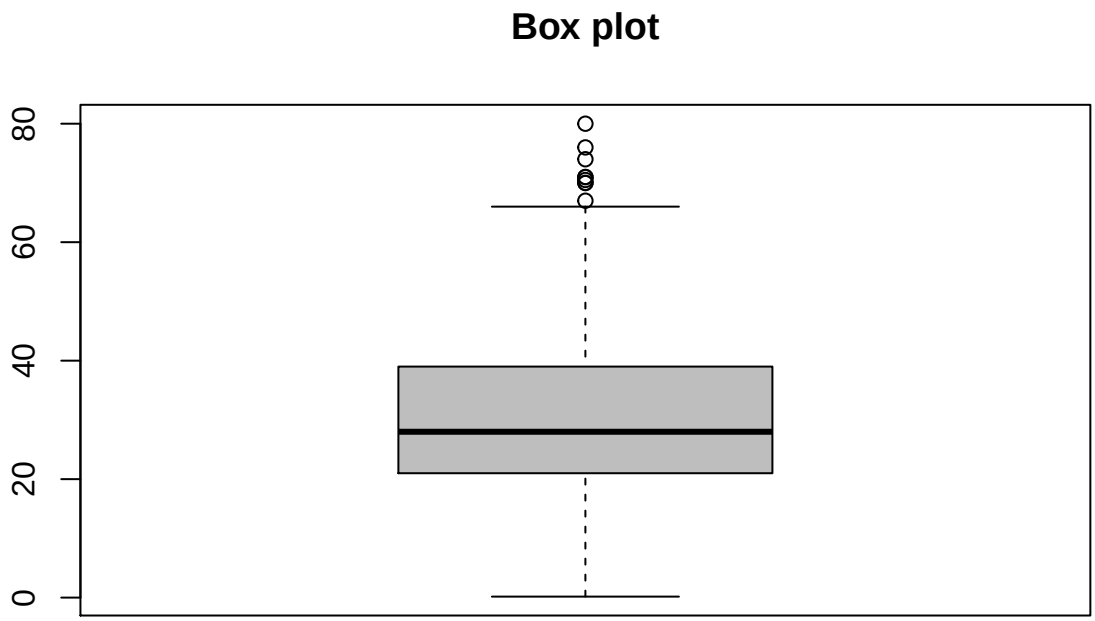
```
## [1] "# Replace Age variable from the mice model.\nfull_data$Age <- mice_output$Age\n\n# Show new numb
```

3.2. Identificación y tratamiento de valores extremos

Los valores extremos o **outliers** son aquellos que parecen no ser congruentes sin los comparamos con el resto de los datos. Para identificarlos se representará un diagrama de caja por cada variable y ver qué valores distan mucho del rango intercuartílico (la caja), para ello se utilizará la función `boxplots.stats()`.

Así, se mostrarán sólo los valores atípicos para variables cuantitativas: `Age`, `Fare`, `SibSp`, `Parch`, y `Fsize`.

```
# Visualizamos boxplot
boxplot(full_data$Age, main="Box plot", col="gray")
```

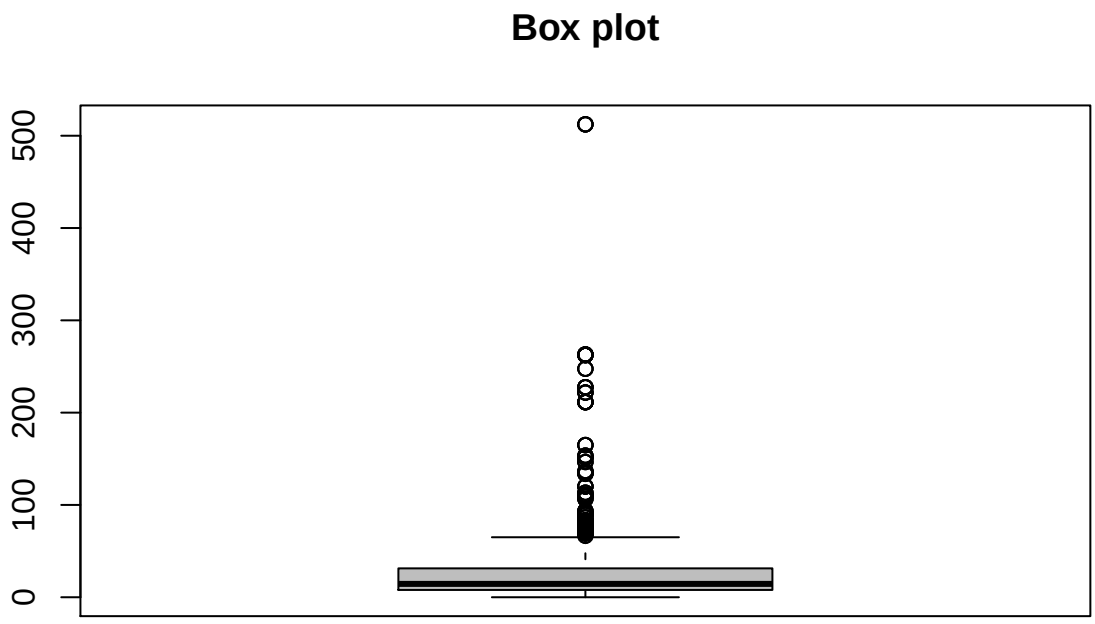



```
boxplot.stats(full_data$Age)$out
```

```
## [1] 71.0 70.5 71.0 80.0 70.0 70.0 74.0 67.0 76.0
```

Para los resultados de la característica **Edad**, si si revisamos de forma aleatoria los datos de los pasajeros se comprueba que los valores extremos están un rango normal. Por ejemplo, ningúno es menor que cero o mayor que 100. Un pasajero con 100 años viajando es poco usual. Por tanto, son valores que perfectamente pueden darse.

```
# Visualizamos boxplot
boxplot(full_data$Fare, main="Box plot", col="gray")
```

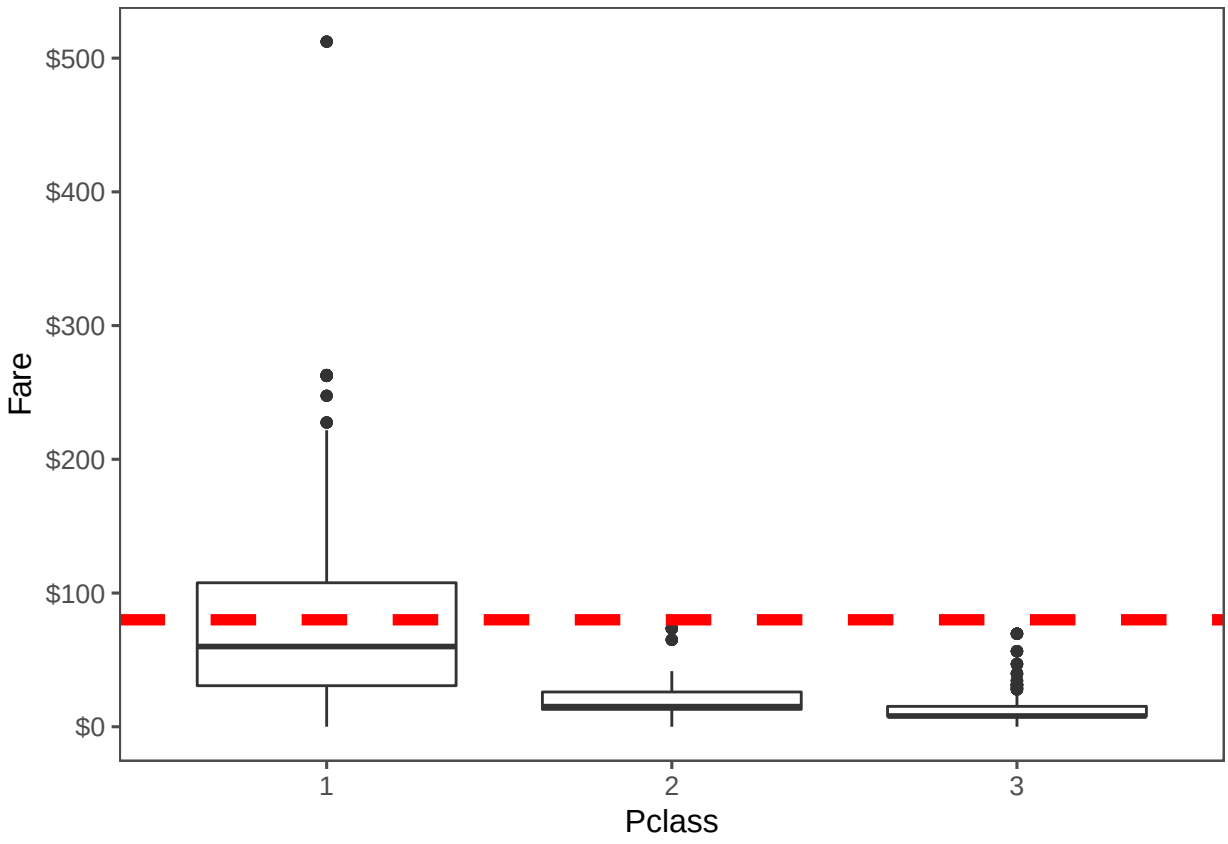



```
boxplot.stats(full_data$Fare)$out
```

```
##   [1]  71.2833 263.0000 146.5208  82.1708  76.7292  80.0000  83.4750
##   [8]  73.5000 263.0000  77.2875 247.5208  73.5000  77.2875  79.2000
##  [15]  66.6000  69.5500  69.5500 146.5208  69.5500 113.2750  76.2917
##  [22]  90.0000  83.4750  90.0000  79.2000  86.5000 512.3292  79.6500
##  [29] 153.4625 135.6333  77.9583  78.8500  91.0792 151.5500 247.5208
##  [36] 151.5500 110.8833 108.9000  83.1583 262.3750 164.8667 134.5000
##  [43]  69.5500 135.6333 153.4625 133.6500  66.6000 134.5000 263.0000
##  [50]  75.2500  69.3000 135.6333  82.1708 211.5000 227.5250  73.5000
##  [57] 120.0000 113.2750  90.0000 120.0000 263.0000  81.8583  89.1042
##  [64]  91.0792  90.0000  78.2667 151.5500  86.5000 108.9000  93.5000
##  [71] 221.7792 106.4250  71.0000 106.4250 110.8833 227.5250  79.6500
##  [78] 110.8833  79.6500  79.2000  78.2667 153.4625  77.9583  69.3000
##  [85]  76.7292  73.5000 113.2750 133.6500  73.5000 512.3292  76.7292
##  [92] 211.3375 110.8833 227.5250 151.5500 227.5250 211.3375 512.3292
##  [99]  78.8500 262.3750  71.0000  86.5000 120.0000  77.9583 211.3375
## [106]  79.2000  69.5500 120.0000  93.5000  80.0000  83.1583  69.5500
## [113]  89.1042 164.8667  69.5500  83.1583  82.2667 262.3750  76.2917
## [120] 263.0000 262.3750 262.3750 263.0000 211.5000 211.5000 221.7792
## [127]  78.8500 221.7792  75.2417 151.5500 262.3750  83.1583 221.7792
## [134]  83.1583  83.1583 247.5208  69.5500 134.5000 227.5250  73.5000
## [141] 164.8667 211.5000  71.2833  75.2500 106.4250 134.5000 136.7792
## [148]  75.2417 136.7792  82.2667  81.8583 151.5500  93.5000 135.6333
## [155] 146.5208 211.3375  79.2000  69.5500 512.3292  73.5000  69.5500
## [162]  69.5500 134.5000  81.8583 262.3750  93.5000  79.2000 164.8667
```

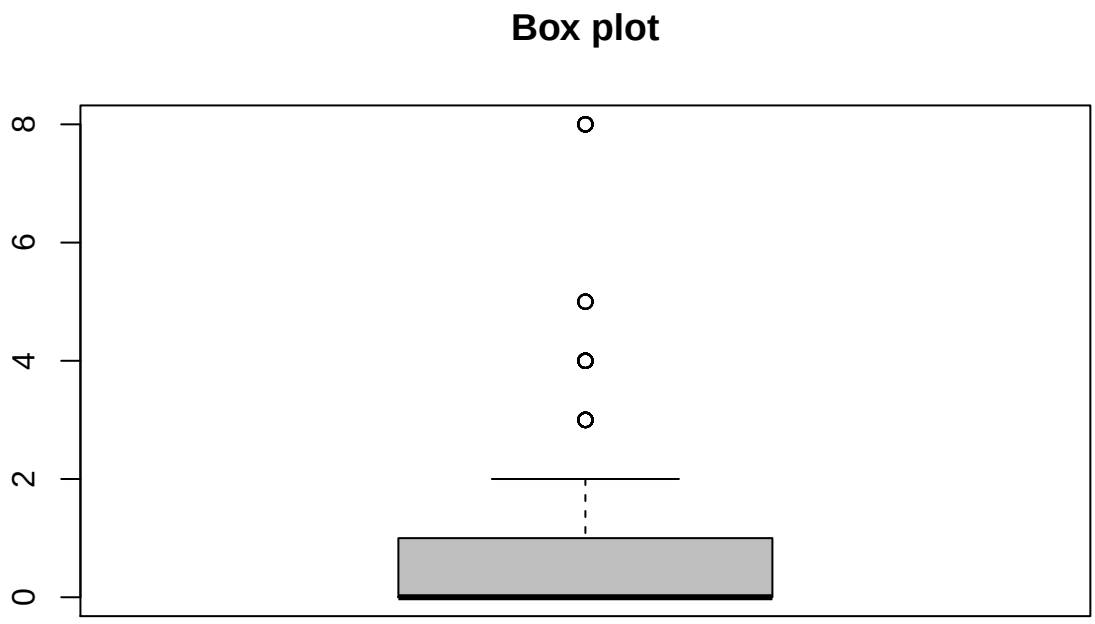
```
## [169] 211.5000  90.0000 108.9000
```

```
# Use ggplot2 to visualize Pclass, passenger class, & median fare
ggplot(full_data, aes(x = Pclass, y = Fare)) +
  geom_boxplot() +
  geom_hline(aes(yintercept=80),
    colour='red', linetype='dashed', lwd=2) +
  scale_y_continuous(labels=dollar_format()) +
  theme_few()
```



Para los resultados de la característica **Tarifa**, si revisamos de forma aleatoria los datos de los pasajeros se comprueba que los valores extremos estan asociados a un mismo ticket en un clase de pasejero especifica. Mientras mejor es la clase y mayor es el número de pasajeros, más alta es la tarifa. Por tanto, son valores que perfectamente pueden darse.

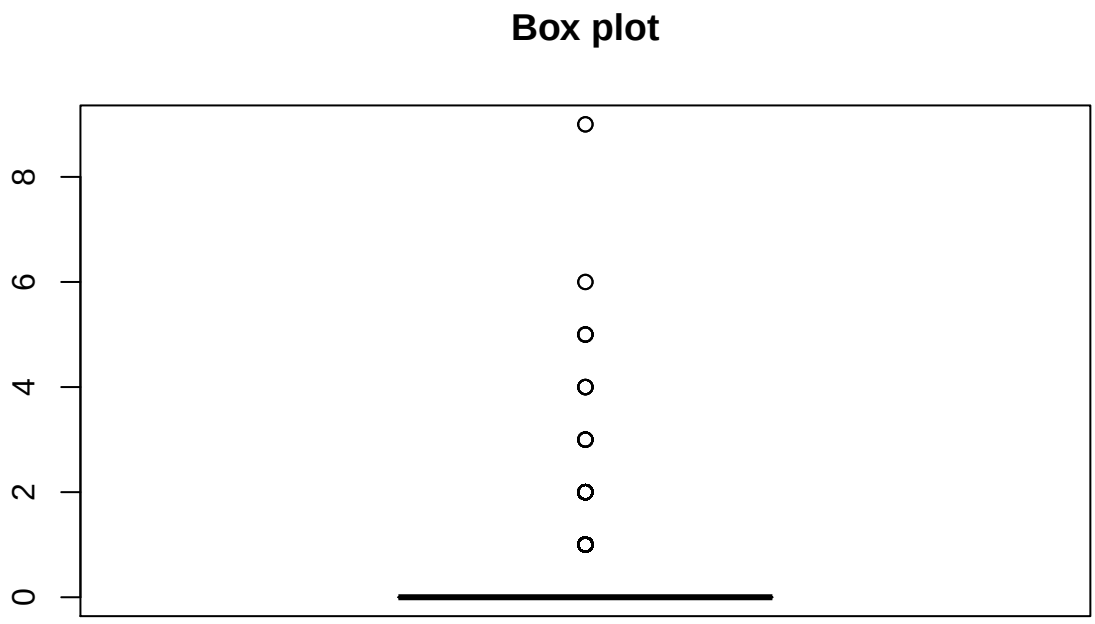
```
boxplot(full_data$SibSp, main="Box plot", col="gray")
```




```
boxplot.stats(full_data$SibSp)$out
```

```
##  [1] 3 4 3 3 4 5 3 4 5 3 3 4 8 4 4 3 8 4 8 3 4 4 4 4 8 3 3 5 3 5 3 4 4 3 3
## [36] 5 4 3 4 8 4 3 4 8 4 8 3 4 5 3 4 8 4 8 4 3 3
```

```
boxplot(full_data$Parch, main="Box plot", col="gray")
```




```
boxplot.stats(full_data$Parch)$out
```

```
##   [1] 1 2 1 5 1 1 5 2 2 1 1 2 2 2 1 2 2 2 3 2 2 1 1 1 1 2 1 1 2 2 1 2 2 2 1
##  [36] 2 1 1 2 1 4 1 1 1 1 2 2 1 2 1 1 1 2 1 1 2 2 2 1 1 2 2 1 2 1 1 1 1 1 1
##  [71] 1 2 1 2 2 1 1 2 1 1 2 1 1 1 1 2 1 1 1 4 1 1 2 2 2 2 2 1 1 1 2 2 1 1 2
## [106] 2 3 4 1 2 1 1 2 1 2 1 2 1 1 2 2 1 1 1 1 2 2 2 2 2 2 1 1 2 1 4 1 1 2 1
## [141] 2 1 1 2 5 2 1 1 1 2 1 5 2 1 1 1 2 1 6 1 2 1 2 1 1 1 1 1 1 1 3 2 1 1 1
## [176] 1 2 1 2 3 1 2 1 2 2 1 1 2 1 2 1 2 1 1 1 2 1 1 2 1 2 1 1 1 1 3 2 1 1 1
## [211] 1 5 2 1 1 1 1 3 1 2 2 1 2 1 2 1 2 4 1 1 2 1 1 1 4 6 2 3 1 1 2 2 2 1 1
## [246] 2 5 2 3 2 1 1 1 2 1 2 2 2 1 2 1 1 2 1 2 1 2 1 2 2 1 1 1 1 2 1 1 2 1
## [281] 1 1 2 1 2 9 1 1 1 2 2 2 1 9 1 1 2 2 1 1 2 1 1 1 1 1 1
```

Para los resultados de las características **Número de hermanos / cónyuges a bordo (SibSp)** y **Número de padres / hijos a bordo(Parch)**; si revisamos de forma aleatoria los datos de los pasajeros se comprueba que los valores extremos están un rango normal. Por ejemplo, ningúno es menor que cero o mayor que 15. Una familia con más de 20 indiviuos viajando junto es poco habitual. Por tanto, son valores que perfectamente pueden darse.

2.3. Exportación de los datos preprocesados

Volvemos a revisar las características un vez más.

```
summary(full_data)
```

```
##   PassengerId   Survived   Pclass      Name              Sex
##  Min.   :   1   0   :549   1:323   Length:1309        F:466
##  1st Qu.: 328   1   :342   2:277   Class :character   M:843
##  Median : 655   NA's:418   3:709   Mode  :character
```

```
##  Mean   : 655
##  3rd Qu.: 982
##  Max.   :1309
##
##       Age            SibSp            Parch           Ticket
##  Min.   : 0.17   Min.   :0.0000   Min.   :0.000   CA2343 :  11
##  1st Qu.:21.00   1st Qu.:0.0000   1st Qu.:0.000   1601   :   8
##  Median :28.00   Median :0.0000   Median :0.000   CA2144 :   8
##  Mean   :29.88   Mean   :0.4989   Mean   :0.385   3101295:   7
##  3rd Qu.:39.00   3rd Qu.:1.0000   3rd Qu.:0.000   347077 :   7
##  Max.   :80.00   Max.   :8.0000   Max.   :9.000   347082 :   7
##  NA's   :263                                      (Other):1261
##       Fare            Cabin           Embarked        Title
##  Min.   :  0.000   Length:1309        C:272    Master    : 61
##  1st Qu.:  7.896   Class :character   Q:123    Miss      :264
##  Median : 14.454   Mode  :character   S:914    Mr        :757
##  Mean   : 33.276                               Mrs       :198
##  3rd Qu.: 31.275                               Rare Title: 29
##  Max.   :512.329
##
##       Surname          Fsize              FsizeD         Deck
##  Andersson:  11   Min.   : 1.000   large    : 82   C      :  94
##  Sage     :  11   1st Qu.: 1.000   singleton:790   B      :  65
##  Asplund  :   8   Median : 1.000   small    :437   D      :  46
##  Goodwin  :   8   Mean   : 1.884                   E      :  41
##  Davies   :   7   3rd Qu.: 2.000                   A      :  22
##  Brown    :   6   Max.   :11.000                   (Other):  27
##  (Other)  :1258                                    NA's   :1014
```

De la información anterior se concluye:

- La variable `PassengerId` se puede eliminarse del conjunto de datos ya que no contribuye a la supervivencia.
- La variable `Name` se puede eliminar debido a que extraído su información en las caracteristcias `Title` y `Surname`.
- La variable `Cabin` se puede eliminar debido a que extraído su información en la `Deck`.
- La variable `Fsize` se puede eliminar debido a que uso como una combinación `SibSp` y `Parch`.

Se seleccionan las siguientes caracteristicas: `Age`, `Sex`, `SibSp`, `Parch`, `Pclass`, `Fare`, `Ticket`, `Title`, `Surname`, `Deck`, y `FSizeD`.

```
# Seleccion de caracteristicas de interes
cleaning_full_data <- select(full_data, -PassengerId, -Name, -Cabin, -Fsize)

# Visualizamos los datos limpios:
summary(cleaning_full_data)
```

```
##  Survived   Pclass  Sex          Age            SibSp
##  0   :549   1:323   F:466   Min.   : 0.17   Min.   :0.0000
##  1   :342   2:277   M:843   1st Qu.:21.00   1st Qu.:0.0000
##  NA's:418   3:709           Median :28.00   Median :0.0000
##                             Mean   :29.88   Mean   :0.4989
##                             3rd Qu.:39.00   3rd Qu.:1.0000
##                             Max.   :80.00   Max.   :8.0000
##                             NA's   :263
##      Parch           Ticket          Fare         Embarked
```

```
##  Min.   :0.000   CA2343 :  11   Min.   :  0.000   C:272
##  1st Qu.:0.000   1601   :   8   1st Qu.:  7.896   Q:123
##  Median :0.000   CA2144 :   8   Median : 14.454   S:914
##  Mean   :0.385   3101295:   7   Mean   : 33.276
##  3rd Qu.:0.000   347077 :   7   3rd Qu.: 31.275
##  Max.   :9.000   347082 :   7   Max.   :512.329
##                  (Other):1261
##         Title          Surname           FsizeD         Deck
##  Master     : 61   Andersson:  11   large    : 82   C      :  94
##  Miss       :264   Sage     :  11   singleton:790   B      :  65
##  Mr         :757   Asplund  :   8   small    :437   D      :  46
##  Mrs        :198   Goodwin  :   8                   E      :  41
##  Rare Title: 29   Davies   :   7                   A      :  22
##                   Brown    :   6                   (Other):  27
##                   (Other)  :1258                   NA's   :1014
```

```
# Split the data back into a train set and a test set
cleaning_train_data <- cleaning_full_data[1:nrow(train_data),]
cleaning_test_data <- cleaning_full_data[(nrow(train_data) + 1):nrow(full_data),]


# Exportación de los datos limpios en .csv
output_path <- 'output'
cleaning_train_file <- 'cleaning_train.csv'
cleaning_test_file <- 'cleaning_test.csv'
cleaning_test_file <- 'cleaning_full.csv'

write.csv(cleaning_train_data, paste(output_path, cleaning_train_file, sep = '/'))
write.csv(cleaning_test_data, paste(output_path, cleaning_test_file, sep = '/'))
write.csv(cleaning_full_data, paste(output_path, cleaning_test_file, sep = '/'))
```

Dividimos el conjunto de datos limpio en dos conjuntos:

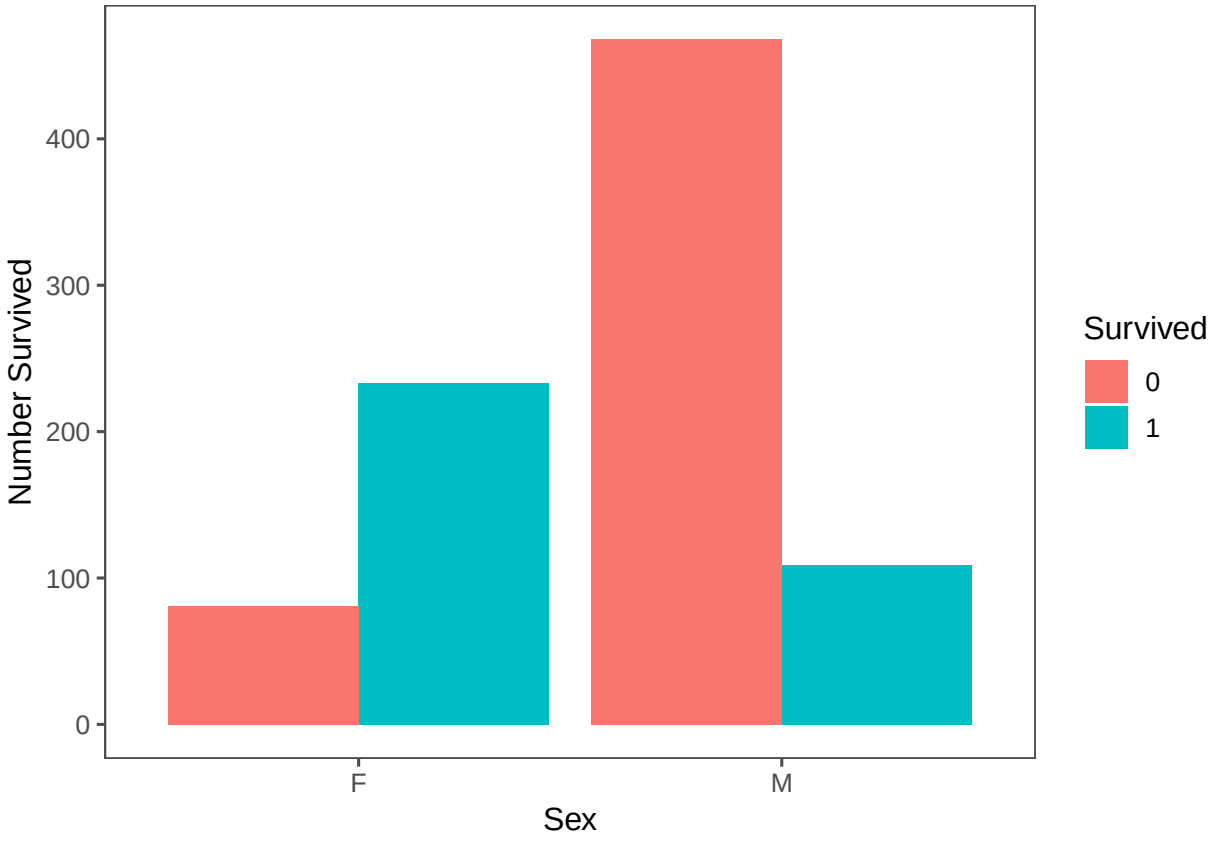
- El conjunto de datos de entrenamiento limpio se almacena en el fichero cleaning_train.csv y está constituido por 891 características y 12 pasajeros.
- El conjunto de datos de pruebas limpio se almacena en el fichero cleaning_full.csv y está constituido por 418 características y 11 pasajeros.

4. Análisis de los datos.

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

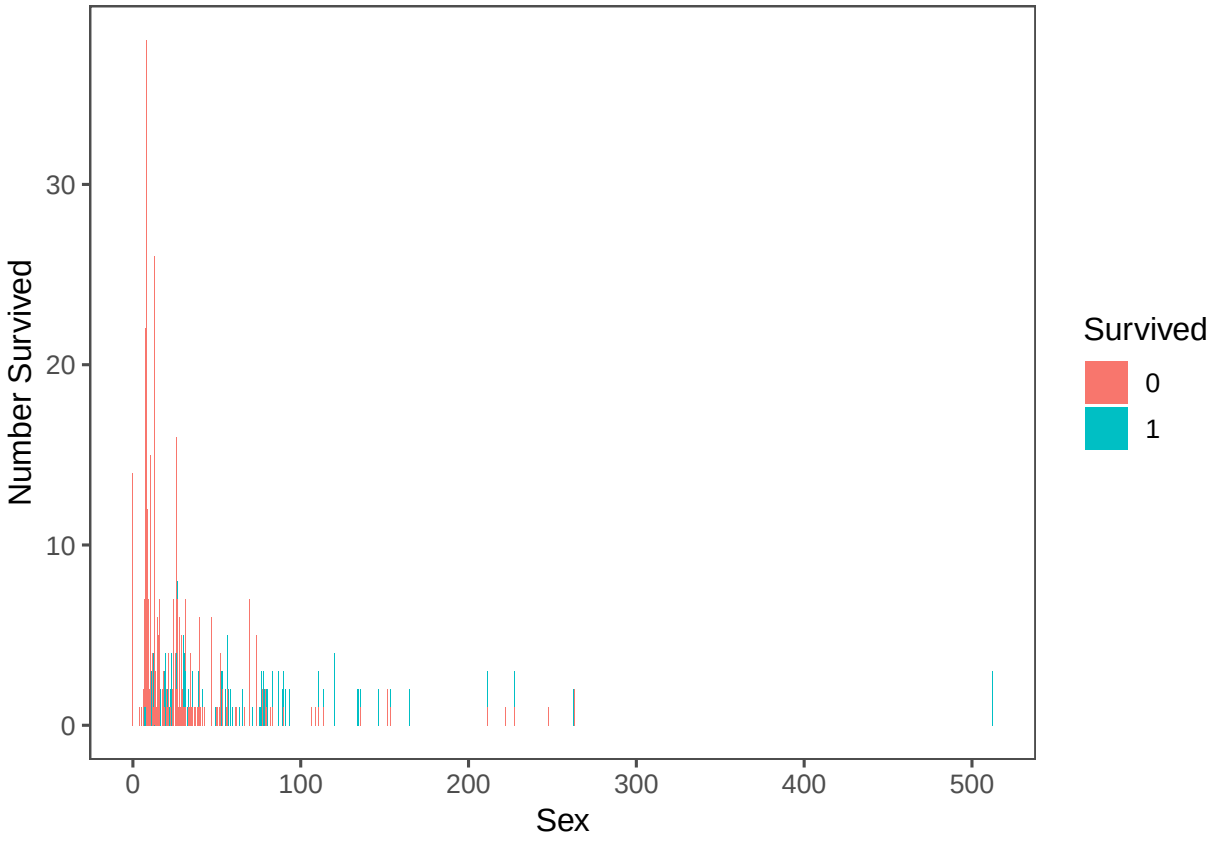
```
#filtered_mydate <- select(mydata, Age, SibSp, Parch, Fare)
#par(mfrow=c(2,2))
#for(i in 1:ncol(filtered_mydate)) {
#  if (is.numeric(filtered_mydate[,i])){
#    boxplot(filtered_mydate[,i], main = colnames(filtered_mydate)[i], width = 100)
#  }
#}

for(varname in c('Sex', 'Age', 'Fare', 'Deck')) {
    ggplot(cleaning_train_data, aes_string(x=varname, fill = 'Survived') ) +
      geom_bar(position=position_dodge()) +
```

```
      theme_few()
}

ggplot(cleaning_train_data, aes(x = Sex, fill = Survived)) +
    geom_bar(position=position_dodge()) +
    labs(y = "Number Survived", x = "Sex") +
    theme_few()
```




```
ggplot(cleaning_train_data, aes(x = Fare, fill = Survived)) +
    geom_bar(position=position_dodge()) +
    labs(y = "Number Survived", x = "Sex") +
    theme_few()
```

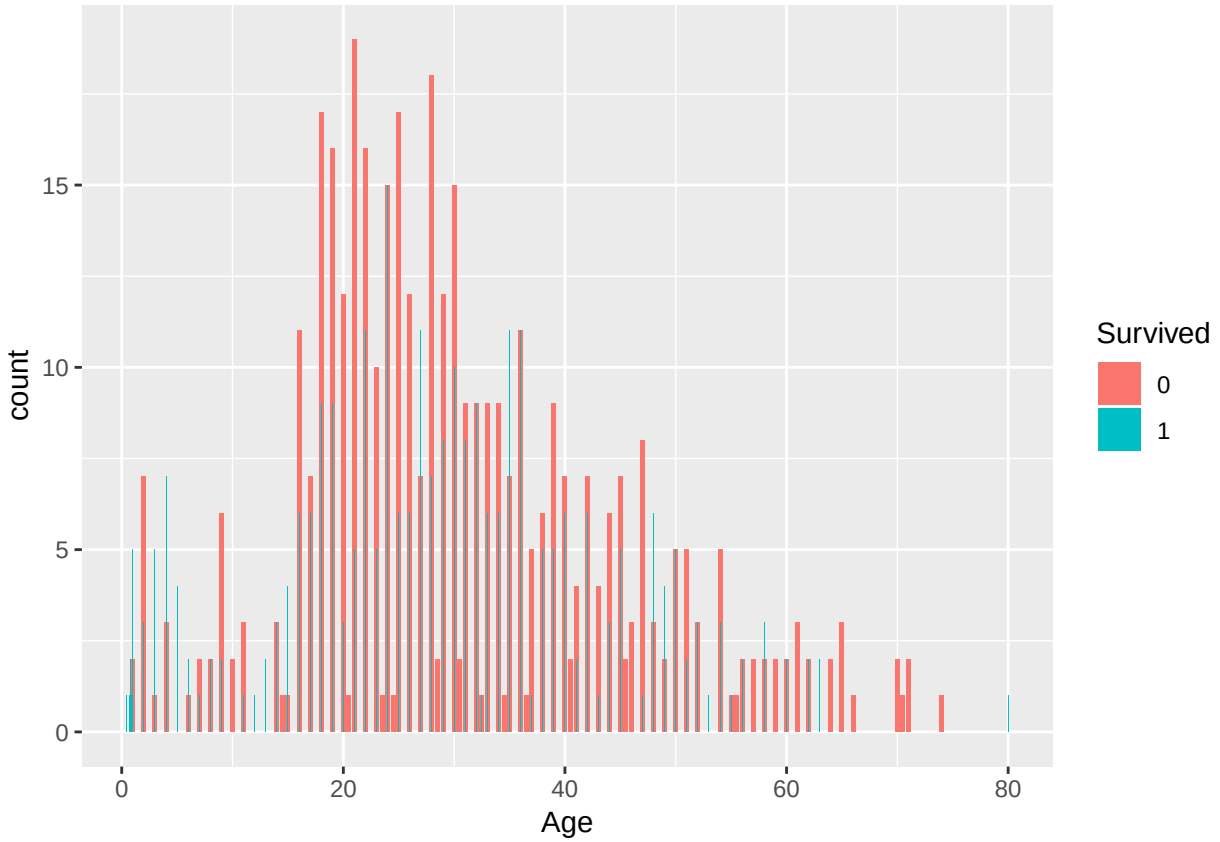
```
## Warning: position_dodge requires non-overlapping x intervals
```



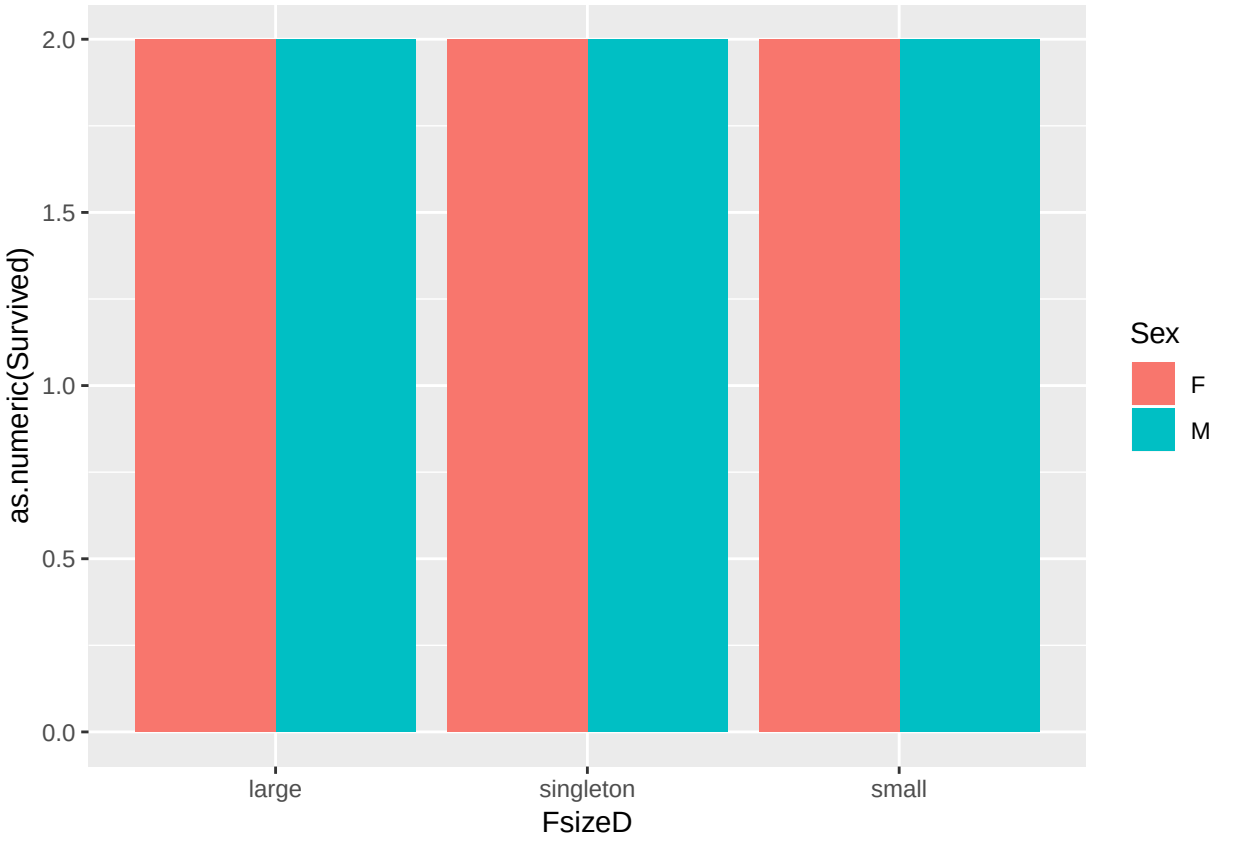

```
ggplot(cleaning_train_data, aes(x = Age, fill = Survived)) +
  geom_bar(position=position_dodge())
```

```
## Warning: Removed 177 rows containing non-finite values (stat_count).
```

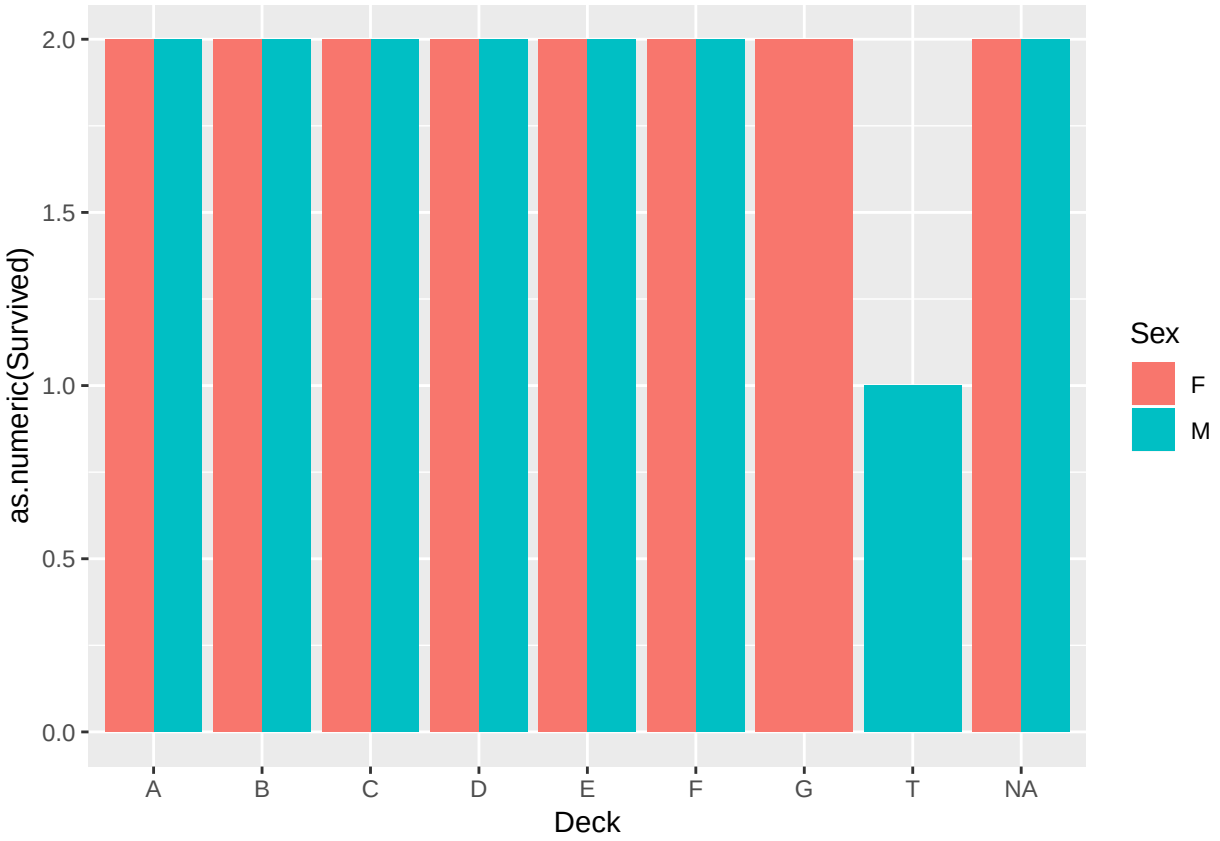
```
## Warning: position_dodge requires non-overlapping x intervals
```




```
ggplot(cleaning_train_data, aes(x = FsizeD, y = as.numeric(Survived), fill = Sex)) +
  geom_bar(stat = "identity", position=position_dodge()) # +
```




```
#  scale_y_continuous(labels=dollar_format()) +
#  theme_few()

ggplot(cleaning_train_data, aes(x = Deck, y = as.numeric(Survived), fill = Sex)) +
  geom_bar(stat = "identity",  position=position_dodge()) #+
```




```
#  scale_y_continuous(labels=dollar_format()) +
#  theme_few()

#ggplot(cleaning_train_data, aes(x = Embarked, y = Survived, fill = Sex)) +
#  geom_col(stat = "identity", position=position_dodge()) #+
#  scale_y_continuous(labels=dollar_format()) +
#  theme_few()
```

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

4.3. Aplicación de pruebas estadísticas para comparar los grupos de datos.

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

6. Resolución del problema.