# Práctica 2

*Jorge Cutipa Musaja*

*6 de enero, 2019*

# 1. Descripción del dataset

En abril de 1912, durante su viaje inaugural, el Titanic se hundió después de chocar con un iceberg, matando a 1502 de 2224 pasajeros y tripulantes. Una de las razones por las que el naufragio llevó a tal pérdida de vidas humanas fue que no había suficientes botes salvavidas para los pasajeros y la tripulación. Aunque siempre habrá algún elemento de azar a considerar, algunos grupos de personas tenían más probabilidades de sobrevivir al hundimiento que otros.

El objetivo de esta práctica es elaborar un árbol de clasificación, para predecir que grupos de **pasajeros** tienen - o mejor dicho, tenían - una mayor probabilidad de sobrevivir al hundimiento del Titanic.

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

Los elementos del dataset son:

```
colnames(datos)
```

```
##  [1] "PassengerId" "Survived"    "Pclass"      "Name"        "Sex"

##  [6] "Age"         "SibSp"       "Parch"       "Ticket"      "Fare"

## [11] "Cabin"       "Embarked"
```

Donde:
- PassengerId: Identificador - Survived: Supervivencia 0 = No, 1 = Sí
- Pclass: Clase de Ticket 1 = 1ra, 2 = 2da, 3 = 3ra
- Name: Nombre del pasajero
- Sex: Sexo del pasajero
- Age: Edad del pasajero (en años)
- SibSp: # de hermanos / cónyuges a bordo del Titanic
- Parch: # de padres / hijos a bordo del Titanic
- Ticket: Número de boleto
- Fare: Tarifa
- Cabin: Número de cabina
- Embarked: Puerto de Embarque C = Cherbourg, Q = Queenstown, S = Southampton

Una descripción de los tipos de datos que R asignó es lo que visualizaremos a continuación:

```
glimpse(datos)
```

1. Descripción del dataset

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

3. Limpieza de los datos

4. Análisis de los datos

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

6. Resolución del problema

```
## Observations: 891
## Variables: 12
## $ PassengerId <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 1
5,...
## $ 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 Br
a...
## $ Sex         <chr> "male", "female", "female", "female", "male", "ma
l...
## $ 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", "113
8...
## $ 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",
...
```

Como se observa, Survived y Pclass han sido consideradas como variables numéricas, cuando en realidad son de tipo factor. Corregiremos esto.

```
datos$Survived <- factor(datos$Survived)
datos$Pclass <- factor(datos$Pclass)
```

Además, dado que SibSp y Parch toman una cantidad finita de valores, las consideraremos como tipo factor.

```
datos$SibSp <- factor(datos$SibSp)
datos$Parch <- factor(datos$Parch)
```

Asimismo, las variables Sex y Embarked, consideradas como tipo character, son en realidad del tipo factor.

```
datos$Sex <- factor(datos$Sex)
datos$Embarked <- factor(datos$Embarked)
```

Por tanto, los tipos de datos del dataset quedan así:

```
glimpse(datos)
```

```
## Observations: 891
## Variables: 12
## $ PassengerId <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 1
5,...
## $ Survived    <fct> 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1,
0,...
## $ Pclass      <fct> 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 Br
a...
## $ Sex         <fct> male, female, female, female, male, male, male, m
a...
## $ Age         <dbl> 22, 38, 26, 35, 35, NA, 54, 2, 27, 14, 4, 58, 20,
...
## $ SibSp       <fct> 1, 1, 0, 1, 0, 0, 0, 3, 0, 1, 1, 0, 0, 1, 0, 0,
4,...
## $ Parch       <fct> 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", "113
8...
## $ Fare        <dbl> 7.2500, 71.2833, 7.9250, 53.1000, 8.0500, 8.4583,
...
## $ Cabin       <chr> "", "C85", "", "C123", "", "", "E46", "", "", "",
...
## $ Embarked    <fct> S, C, S, S, S, Q, S, S, S, C, S, S, S, S, S, S,
Q,...
```

En resumen, en total, se tienen 891 registros y 12 variables, incluyendo a la variable objetivo: Survived.

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

Como realizaremos un árbol de clasificación, no es necesario hacer una preselección de variables relevantes. Lo único que haremos será eliminar variables que no aportan información: nombre del pasajero, número de ticket, número de cabina y el identificador. En el caso de los tres primeros, son datos tipo caracter que no pueden ser tomados como factores, por ello su eliminación. En el caso de identificador, si bien es del tipo numérico, cumple el mismo propósito que el nombre del pasajero, y como no tiene sentido tomar como un factor el identificador, se elimina.

```
datos <- datos[,colnames(datos)!="Name"]
datos <- datos[,colnames(datos)!="Ticket"]
datos <- datos[,colnames(datos)!="Cabin"]
datos <- datos[,colnames(datos)!="PassengerId"]
```

Ahora, en total, se tienen 891 registros y 8 variables, incluyendo a la variable objetivo: Survived.

# 3. Limpieza de los datos

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

Podemos contabilizar los elementos vacíos de las variables

```
datos.na = data.frame(sapply(datos, complete.cases))
sapply(-datos.na+1, sum)
```

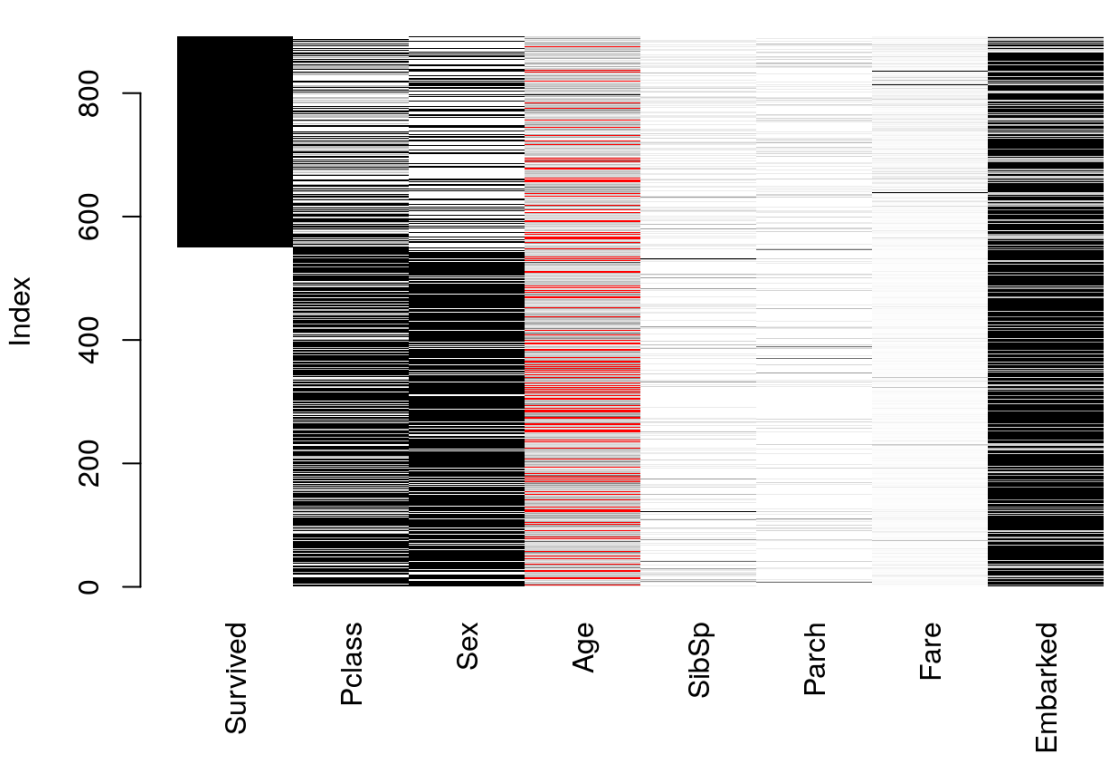
```
## Survived   Pclass      Sex      Age    SibSp    Parch     Fare Embarked

##        0        0        0      177        0        0        0        0
```

Como se observa, el número de valores perdidos para Age es relativamente alto (19.9%) . Una opción sería imputar los valores faltantes en Age con la ayuda de algún algoritmo; otra opción es eliminar todos los registros con valores perdidos en Age, pues aún eliminando estos registros, tendríamos suficiente muestra para construir nuestro árbol de clasificación.

Antes de decidir si imputaremos valores para Age o si eliminaremos los registros con valores perdidos en Age, conviene verificar si los valores perdidos no siguen algún patrón. Para ello, utilizaremos una matrixplot.

```
matrixplot(datos, sortby = "Survived")
```




Como podemos observar, ordenando los datos respecto a nuestra variable objetivo, los valores perdidos en Age (representados en color rojo) no siguen ningún patrón.

Imputaremos los datos con el algoritmo random forest, mediante el cual se generan N árboles de clasificación en forma aleatoria, con información del dataset, para imputar los datos faltantes.

1. Descripción del dataset
2. Integración y selección de los datos de interés a analizar
3. Limpieza de los datos
4. Análisis de los datos
5. Representación de los resultados a partir de tablas y gráficas
6. Resolución del problema

```
datos.imputed <- rfImpute(Survived ~ ., datos)
```

```
## ntree      OOB      1      2
##   300:  17.40%  8.56% 31.58%
## ntree      OOB      1      2
##   300:  17.40%  8.56% 31.58%
## ntree      OOB      1      2
##   300:  16.16%  8.38% 28.65%
## ntree      OOB      1      2
##   300:  16.72%  8.93% 29.24%
## ntree      OOB      1      2
##   300:  16.95%  8.74% 30.12%
```

Como podemos verificar, ya no tenemos valores perdidos.

```
datos.na = data.frame(sapply(datos.imputed, complete.cases))
sapply(-datos.na+1, sum)
```

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

##        0        0        0        0        0        0        0        0
```

También, debemos verificar que, para el caso de las variables categóricas (factor), todos sus valores están correctamente etiquetados. Es de decir, que no se tengan valores en las categorías del tipo " “.

```
datos.factor <- datos.imputed[sapply(datos.imputed, is.factor)]
apply(datos.factor, 2, table)
```

```
## $Survived
##
##   0   1
## 549 342
##
## $Pclass
##
##   1   2   3
## 216 184 491
##
## $Sex
##
## female   male
##    314    577
##
## $SibSp
##
##   0   1   2   3   4   5   8
## 608 209  28  16  18   5   7
##
## $Parch
##
##   0   1   2   3   4   5   6
## 678 118  80   5   4   5   1
##
## $Embarked
##
##       C   Q   S
##   2 168  77 644
```

Como se observa, para el caso de Embarked, dos de sus valores no están etiquetados. Elimaremos estos dos registros del dataset.

```
datos.imputed <- filter(datos.imputed, datos.imputed$Embarked=="C" | dato
s.imputed$Embarked=="Q" | datos.imputed$Embarked=="S")
datos.imputed$Embarked <- factor(datos.imputed$Embarked)
```

Por tanto, en total, se tienen 889 registros y 8 variables, incluyendo a la variable objetivo: Survived.

## 3.2. Identificación y tratamiento de valores extremos.

Para identificar valores extremos realizaremos boxplot de cada variable numérica en el dataset.

1. Descripción del dataset

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

3. Limpieza de los datos

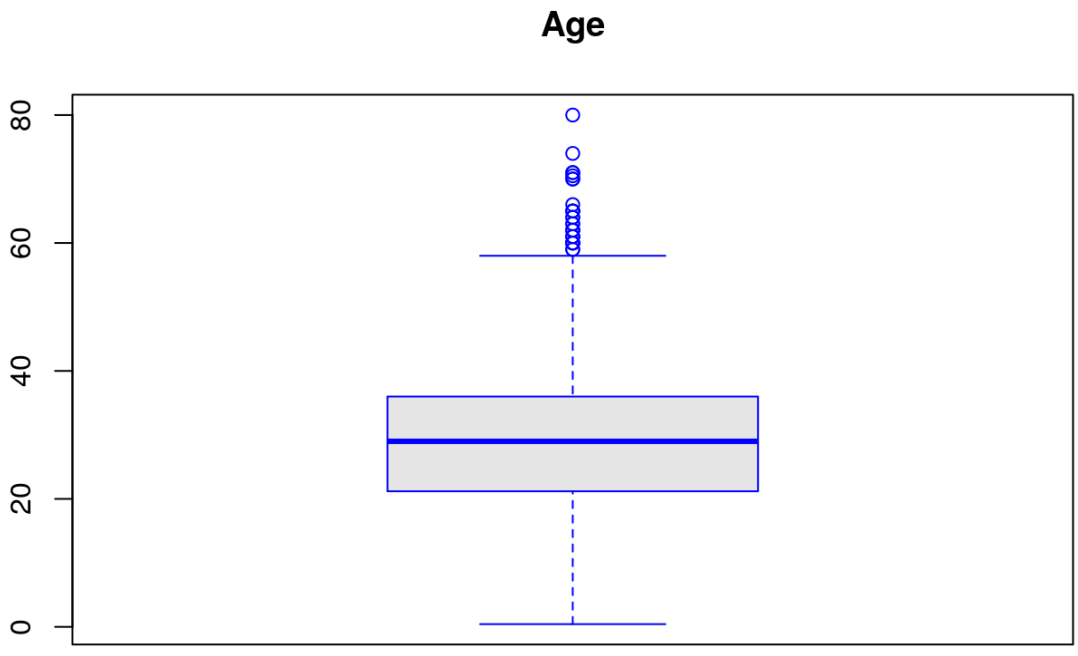
4. Análisis de los datos

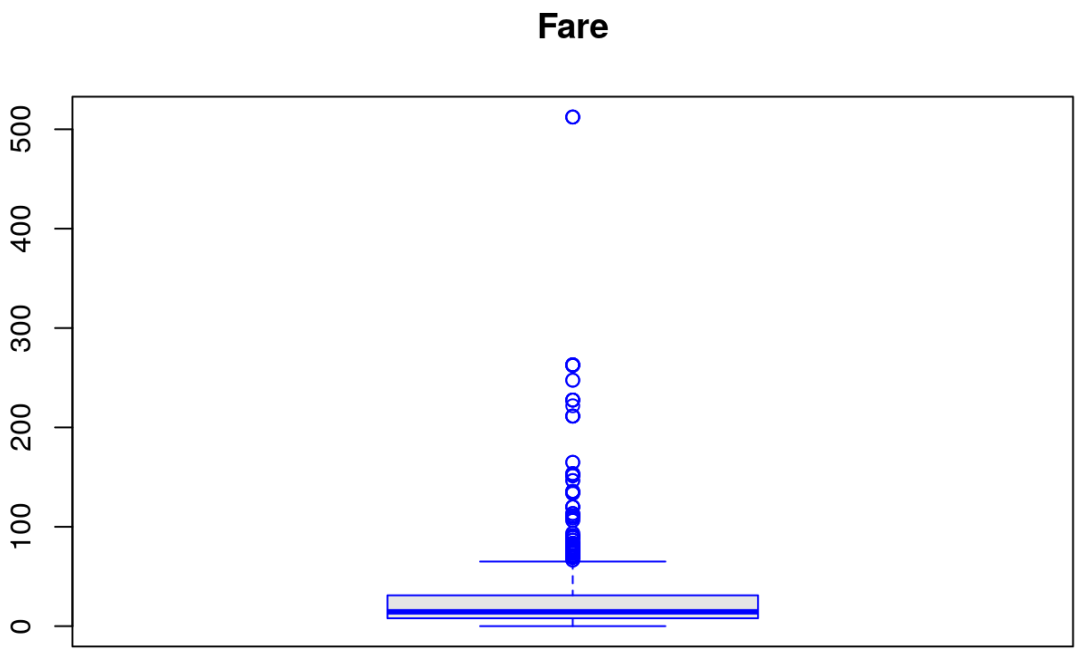
5. Representación de los resultados a partir de tablas y gráficas

6. Resolución del problema

```
datos.numeric <- datos.imputed[sapply(datos.imputed, is.numeric)]

for (i in 1:dim(datos.numeric)[2]){
  boxplot(datos.numeric[,i], main=names(datos.numeric)[i], border = "blue"
, col = "grey90")
  }
```




Gráficamente, podemos interpretar que tanto para Age como para Fare se observan una gran cantidad de valores extremos; sin embargo, será mejor inspeccionar al detalle cuántos valores extremos tenemos en estas dos variables.

```
#Valores extremos para Age
boxplot.stats(datos.imputed$Age)$out
```

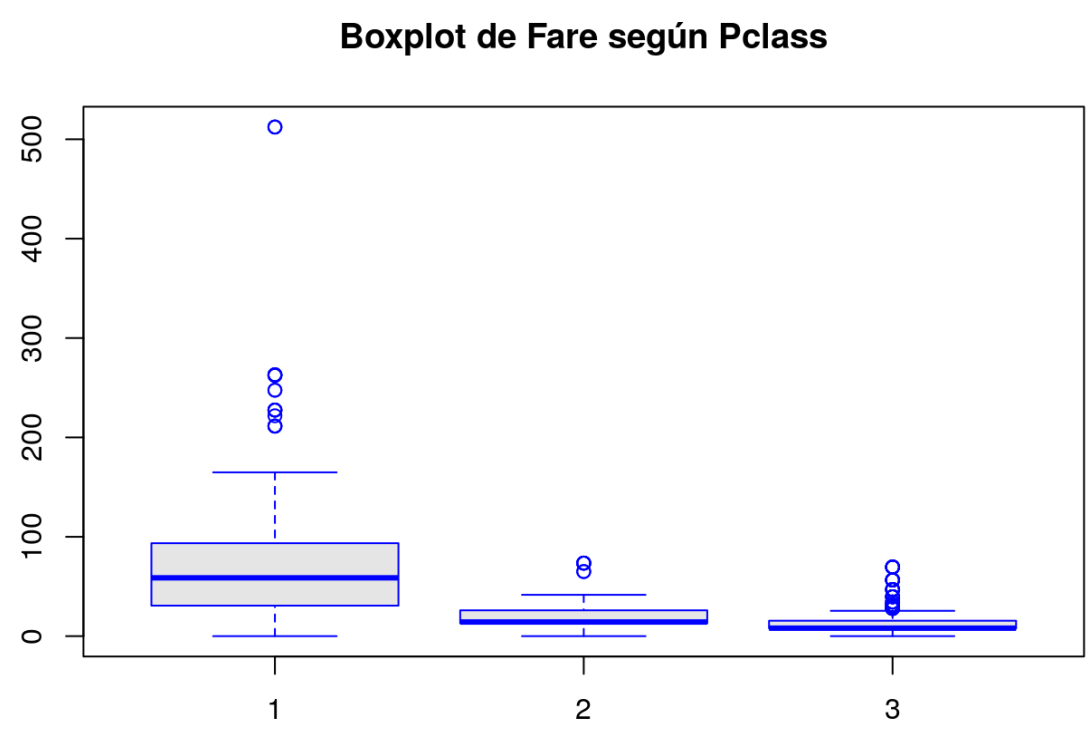
```
##  [1] 66.0 65.0 59.0 71.0 70.5 61.0 59.0 62.0 63.0 65.0 61.0 60.0 64.0 6
5.0
## [15] 63.0 71.0 64.0 62.0 62.0 60.0 61.0 80.0 70.0 60.0 60.0 70.0 74.0
```

```
#Valores extremos para Fare
boxplot.stats(datos.imputed$Fare)$out
```

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

Podemos observar que en el caso de Fare es donde se tiene un elevado número de valores extremos (114). Una solución podría ser recortar la muestra, eliminando los registros con valores extremos en Fare; sin embargo, debemos recordar que el problema de los valores extremos se extiende cuando debemos hacer contrastes de hipótesis, correlaciones, regresiones, etc. En nuestro caso, un árbol de clasificación es un método computacional, y si bien es cierto que se rige bajo ciertas normas estadísticas, que no existan valores extremos en las variables no es un requisito necesario para su elaboración. Por tanto, mantendremos el dataset sin ningún cambio. Además, se deben eliminar los valores extremos cuando estos representan valores no posibles o poco posibles en la variable de estudio; es decir, cuando que presumimos que se ha registradon por error estos valores extremos, y por tanto, es conveniente eliminarlos. No es así en nuestro caso; pues como observamos, en el caso de Age, estos valores son completamente posibles, y en el caso de Fare, también. Esto último se puede visualizar relacionando un boxplot de Fare según Pclass (Clase de Ticket).

```
boxplot(datos.imputed$Fare ~ datos.imputed$Pclass, main="Boxplot de Fare s
egún Pclass", border = "blue", col = "grey90")
```




# 4. Análisis de los datos

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

Seleccionamos los conjuntos de entrenamiento (train) y prueba (test):

```
set.seed(32)
ids <- createDataPartition(datos.imputed$Survived,
                           p = 0.7,
                           list = F)

train <- datos.imputed[ids,]
test <- datos.imputed[-ids,]
```

Por tanto, del dataset original de 889 registros, se han tomado 623 registros para el dataset train y 266 registros para el dataset test.

1. Descripción del dataset

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

3. Limpieza de los datos

4. Análisis de los datos

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

6. Resolución del problema

Respecto al pre-análisis, si bien es cierto que no podemos realizar correlaciones con las variables categóricas (tipo factor), sí podemos hacer tablas con ellas y hacer un barplot de ellas.

```
datos.factor <- datos.imputed[sapply(datos.imputed, is.factor)]

for (i in 1:5){
  tabla <- datos.factor[,c(1,i+1)]
  p <- ggplot(tabla, aes(x=tabla[,2], y="", fill=Survived)) +
    geom_bar(stat="identity") + ylab("") + xlab("") +
    scale_fill_manual(values=c("#969696", "#252525")) +
    ggtitle(paste0("Barplot de ",colnames(tabla)[2],", según Survived"))
  plot(p)
  }
```

1. Descripción del dataset

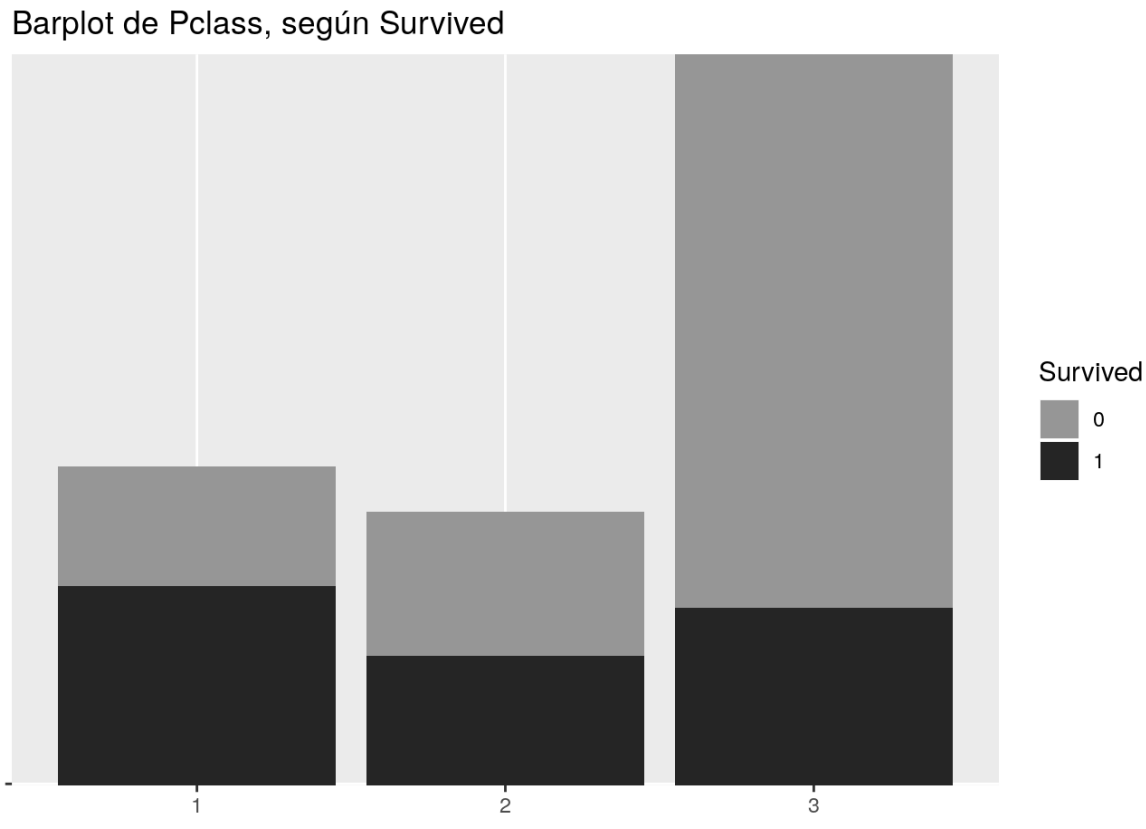
2. Integración y selección de los datos de interés a analizar

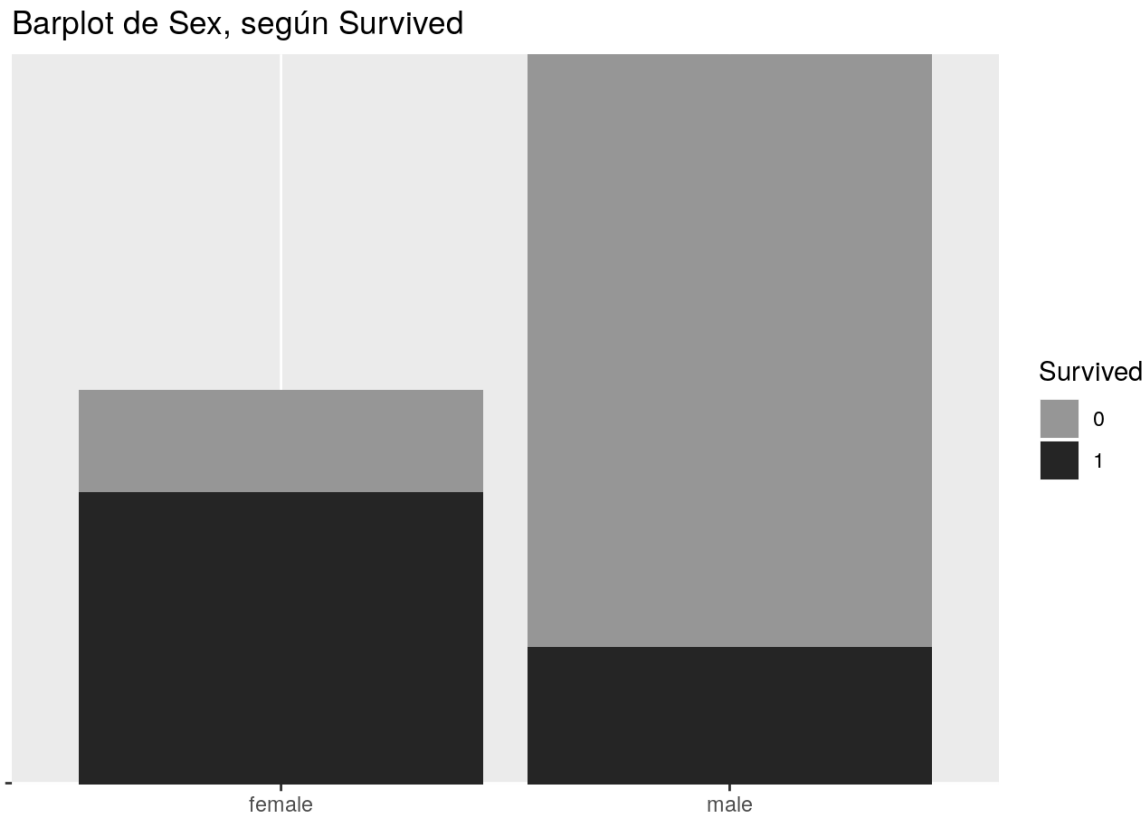
3. Limpieza de los datos

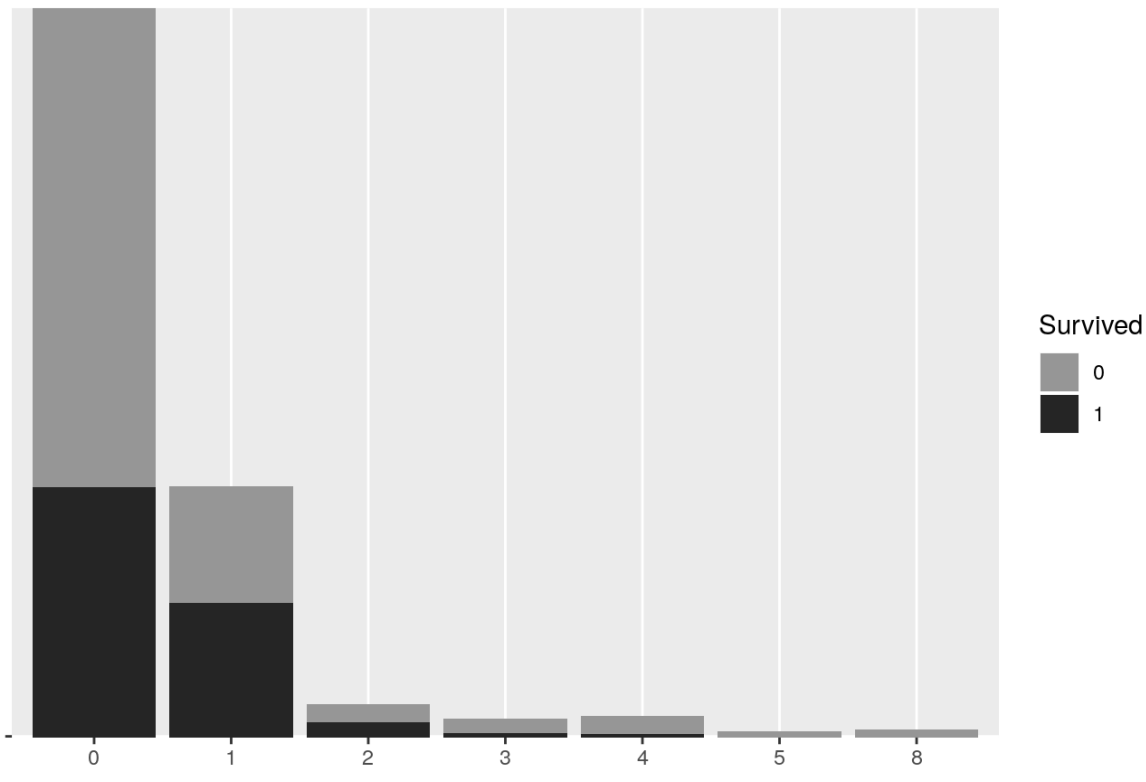
4. Análisis de los datos

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

6. Resolución del problema



1. Descripción del dataset

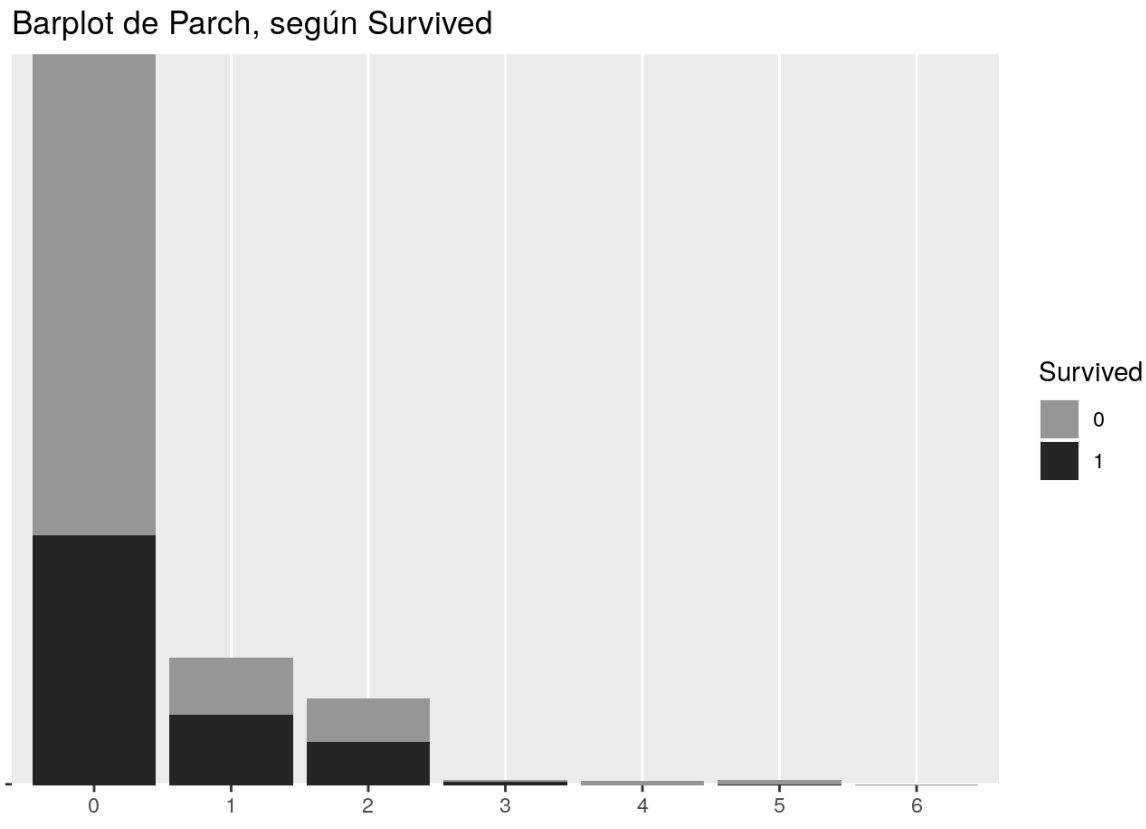
2. Integración y selección de los datos de interés a analizar

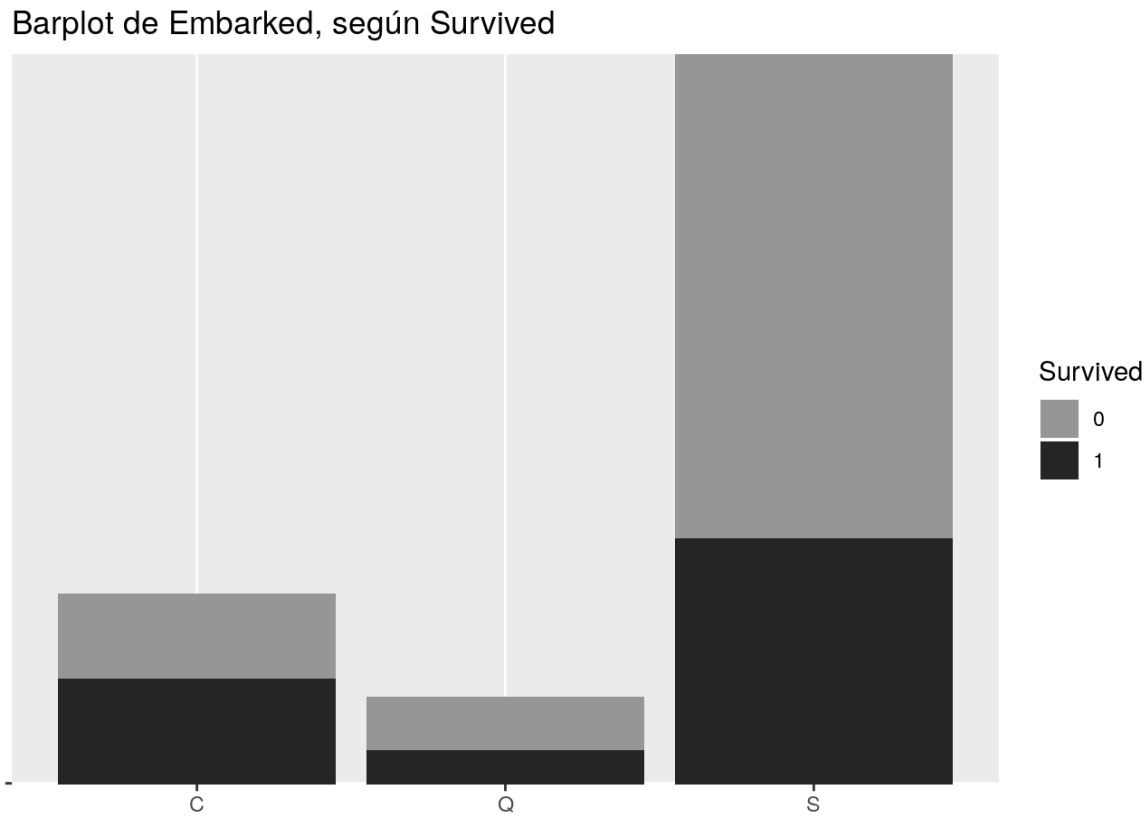
3. Limpieza de los datos

4. Análisis de los datos

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

6. Resolución del problema




Como se puede apreciar en los gráficos, se esperan las siguientes relaciones:
- Mayor probabilidad de sobrevivir si la clase de Ticket (Pclass) es 1 = 1ra.
- Mayor probabilidad de no sobrevivir si la clase de ticket (Pclass) es 3 = 3ra. - Mayor probabilidad de sobrevivir si se es mujer.
- Mayor probabilidad de no sobrevivir si se es hombre.
- Mayor probabilidad de no sobrevivir si el número de hermanos / cónyuges a bordo del Titanic (SibSp) es cero.
- Mayor probabilidad de no sobrevivir si el número de padres / hijos a bordo del Titanic (Parch) es cero.
- Mayor probabilidad de no sobrevivir si el puerto de embarque (Embarked) es S = Southampton.

Y respecto a las variables numéricas, podemos realizar un boxplot de ellas, diferenciandolas según Survived.

```
boxplot(datos.imputed$Age ~ datos.imputed$Survived, main="Boxplot de Age s
egún Survived", border = "blue", col = "grey90")
```

1. Descripción del dataset

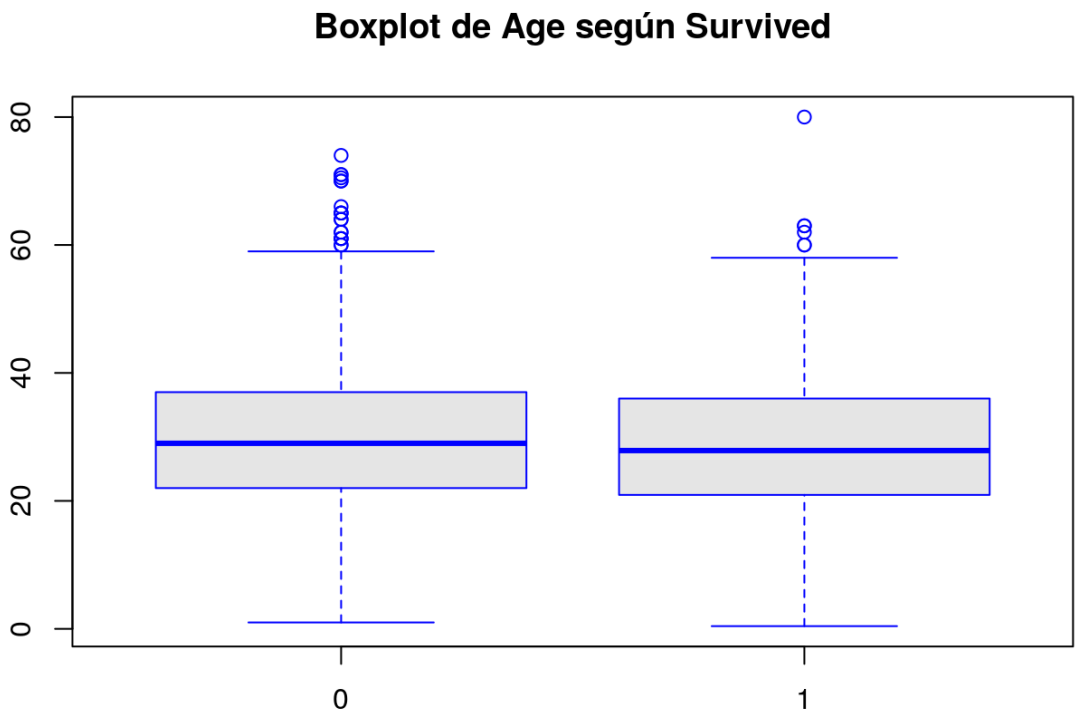
2. Integración y selección de los datos de interés a analizar

3. Limpieza de los datos

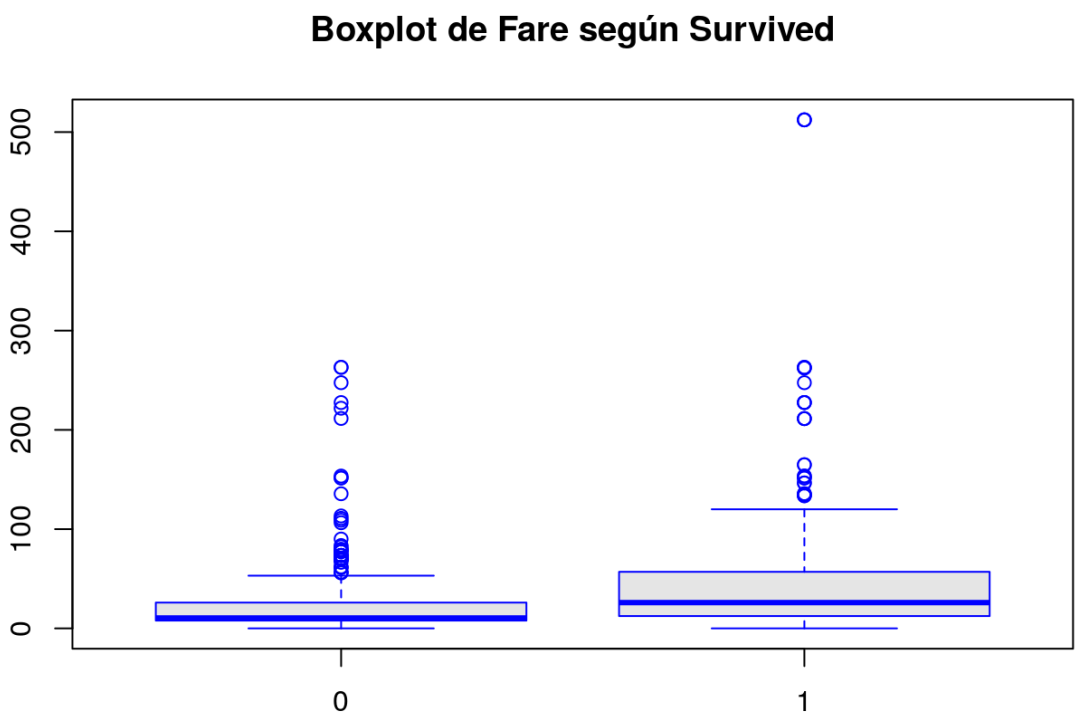
4. Análisis de los datos

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

6. Resolución del problema




```
boxplot(datos.imputed$Fare ~ datos.imputed$Survived, main="Boxplot de Fare
 según Survived", border = "blue", col = "grey90")
```




De estos últimos gráficos, se puede apreciar que, con respecto a Age, no se aprecia mayor diferencia cuando se agrupan los datos según Survived; mientras que, para el caso de Fare, se aprecia cierta dierencia cuando se agrupan los datos según Survived.

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

Para evaluar la normalidad en las variables numéricas del dataset utilizaremos el test de Kolmogorov-Smirnov.

```
datos.numeric <- datos.imputed[sapply(datos.imputed, is.numeric)]
apply(datos.numeric, 2, lillie.test)
```

```
## $Age
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  newX[, i]
## D = 0.074327, p-value = 1.383e-12
##
##
## $Fare
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  newX[, i]
## D = 0.28257, p-value < 2.2e-16
```

Como se observa, tanto para Age como para Fare, el test arroja por resultado No Normalidad. Como se conoce, las consecuencias de la no normalidad afectan principalmente a los test de hipótesis paramétricos y a los modelos de regresión: los estimadores mínimo-cuadráticos no son eficientes (de mínima varianza) y los intervalos de confianza de los parámetros del modelo y los contrastes de significancia son solamente aproximados y no exactos.

Podríamos aplicar una normalización a Age y Fare, con la desventaja que perderíamos interpretabilidad al realizar el árbol de clasificación y, dado que no es necesario que los datos numéricos sean normales para aplicar el árbol de clasificación, optaremos por no transformar Age ni Fare.

Ahora, aplicaremos el test respectivo para probar la homocesasticidad de las variables numéricas en los niveles del factor Survived. Deberíamos de usar, en presencia de normalidad en las variables un F-test, pero dado que no cumplen esta condición, lo más recomendable es aplicar el test de Levene, utilizando la mediana.

```
leveneTest(y = datos.imputed$Age, group = datos.imputed$Survived, center =
 "median")
```

```
## Levene's Test for Homogeneity of Variance (center = "median")
##        Df F value  Pr(>F)
## group   1   5.214 0.02264 *
##       887
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
leveneTest(y = datos.imputed$Fare, group = datos.imputed$Survived, center
= "median")
```

```
## Levene's Test for Homogeneity of Variance (center = "median")
##        Df F value    Pr(>F)
## group   1  44.372 4.759e-11 ***
##       887
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Como se observa, tanto para el caso de Age como Fare, el test de Levene arroja heterocedasticidad de varianza. Es decir que el test encuentra diferencias significativas entre las varianzas de los dos grupos (Survived==1 y Survived==0) para las variables Age y Fare.

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

Entrenamos el árbol de clasificación

```
tree <- rpart(Survived ~ ., data=train)
```

Mostramos los resultados del entrenamiento

```
tree
```

```
## n= 623
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##   1) root 623 238 0 (0.61797753 0.38202247)
##     2) Sex=male 403  79 0 (0.80397022 0.19602978)
##       4) Age>=9.5 378  64 0 (0.83068783 0.16931217) *
##       5) Age< 9.5 25  10 1 (0.40000000 0.60000000)
##        10) SibSp=3,4,5 11   1 0 (0.90909091 0.09090909) *
##        11) SibSp=0,1,2 14   0 1 (0.00000000 1.00000000) *
##     3) Sex=female 220  61 1 (0.27727273 0.72272727)
##       6) Pclass=3 103  49 0 (0.52427184 0.47572816)
##        12) Fare>=24.80835 21   2 0 (0.90476190 0.09523810) *
##        13) Fare< 24.80835 82  35 1 (0.42682927 0.57317073)
##          26) Fare>=7.9021 54  26 0 (0.51851852 0.48148148)
##            52) Fare< 15.3729 34  12 0 (0.64705882 0.35294118)
##             104) Fare>=13.90835 9   0 0 (1.00000000 0.00000000) *
##             105) Fare< 13.90835 25  12 0 (0.52000000 0.48000000)
##               210) Age>=19 18   5 0 (0.72222222 0.27777778) *
##               211) Age< 19 7   0 1 (0.00000000 1.00000000) *
##            53) Fare>=15.3729 20   6 1 (0.30000000 0.70000000) *
##          27) Fare< 7.9021 28   7 1 (0.25000000 0.75000000) *
##       7) Pclass=1,2 117   7 1 (0.05982906 0.94017094) *
```

Como puede observarse, las variables Parch y Embarked no han sido consideradas por el algoritmo del árbol de clasificación; por tanto, es preferible eliminarlas del dataset.

```
datos.imputed <- datos.imputed[,colnames(datos.imputed)!="Parch"]
datos.imputed <- datos.imputed[,colnames(datos.imputed)!="Embarked"]
```

Creamos nuevamente los datasets train y test.

```
set.seed(32)
ids <- createDataPartition(datos.imputed$Survived,
                           p = 0.7,
                           list = F)

train <- datos.imputed[ids,]
test <- datos.imputed[-ids,]
```

Realizamos nuevamente el entrenamiento del árbol de clasificación.

```
tree <- rpart(Survived ~ ., data=train)
```

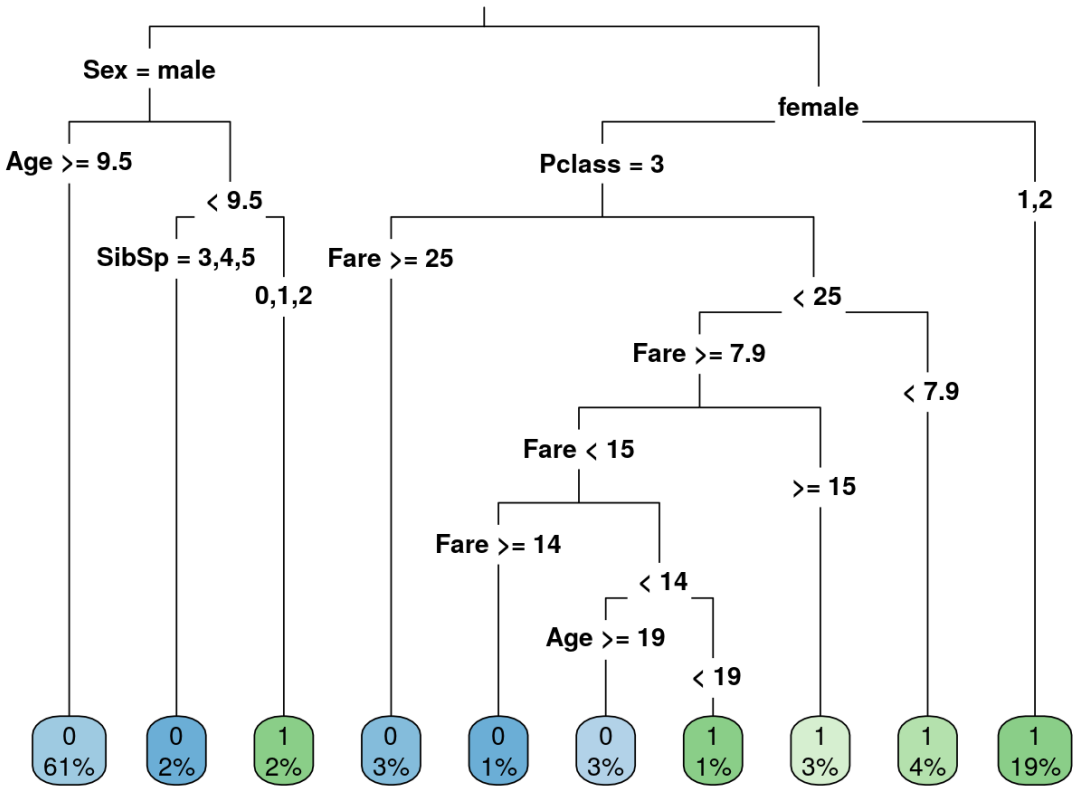
Mostramos los resultados del entrenamiento

```
tree
```

```
## n= 623
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##   1) root 623 238 0 (0.61797753 0.38202247)
##     2) Sex=male 403  79 0 (0.80397022 0.19602978)
##       4) Age>=9.5 378  64 0 (0.83068783 0.16931217) *
##       5) Age< 9.5 25  10 1 (0.40000000 0.60000000)
##        10) SibSp=3,4,5 11   1 0 (0.90909091 0.09090909) *
##        11) SibSp=0,1,2 14   0 1 (0.00000000 1.00000000) *
##     3) Sex=female 220  61 1 (0.27727273 0.72272727)
##       6) Pclass=3 103  49 0 (0.52427184 0.47572816)
##        12) Fare>=24.80835 21   2 0 (0.90476190 0.09523810) *
##        13) Fare< 24.80835 82  35 1 (0.42682927 0.57317073)
##          26) Fare>=7.9021 54  26 0 (0.51851852 0.48148148)
##            52) Fare< 15.3729 34  12 0 (0.64705882 0.35294118)
##             104) Fare>=13.90835 9   0 0 (1.00000000 0.00000000) *
##             105) Fare< 13.90835 25  12 0 (0.52000000 0.48000000)
##               210) Age>=19 18   5 0 (0.72222222 0.27777778) *
##               211) Age< 19 7   0 1 (0.00000000 1.00000000) *
##            53) Fare>=15.3729 20   6 1 (0.30000000 0.70000000) *
##          27) Fare< 7.9021 28   7 1 (0.25000000 0.75000000) *
##       7) Pclass=1,2 117   7 1 (0.05982906 0.94017094) *
```

Para una mejor interpretación, mostramos gráficamente el árbol de clasificación.

```
rpart.plot(tree, extra = 100, type = 3)
```




Cada nodo resultante refleja la predicción para Survived. De todos ellos, son dos los que deben llamarnos la atención, pues en conjunto representan el 80% de la predicción para Survived. El primero de ellos es 61%, que quiere decir que si se es hombre y se tiene una edad mayor o igual

a 9.5 años, se tiene una probabilidad de no supervivencia del 61%. El segundo, 19%, significa que si se es mujer y el ticket es de 1ra o 2da clase, se tiene una probabilidad de supervivencia del 19%.

Aplicamos el árbol de clasificación al test.

```
newdata <- test[,colnames(test)!="Survived"]
prediccion <- predict(tree, newdata=newdata, type="class")
```

Mostramos los resultados de aplicar el árbol de clasificación al test.

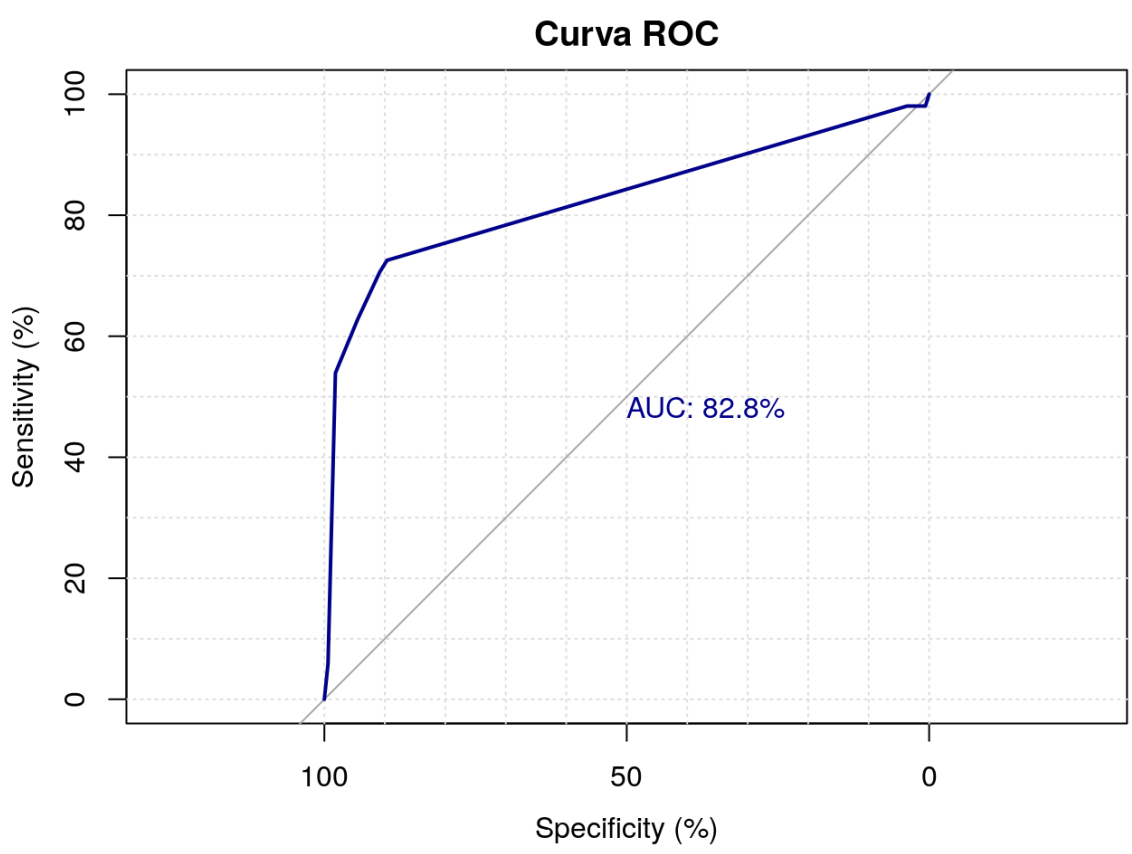
```
confusionMatrix(data=prediccion, reference=test$Survived, positive="1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 149  30
##          1  15  72
##
##                Accuracy : 0.8308
##                  95% CI : (0.7803, 0.8738)
##     No Information Rate : 0.6165
##     P-Value [Acc > NIR] : 2.211e-14
##
##                   Kappa : 0.632
##  Mcnemar's Test P-Value : 0.03689
##
##             Sensitivity : 0.7059
##             Specificity : 0.9085
##          Pos Pred Value : 0.8276
##          Neg Pred Value : 0.8324
##              Prevalence : 0.3835
##          Detection Rate : 0.2707
##    Detection Prevalence : 0.3271
##       Balanced Accuracy : 0.8072
##
##        'Positive' Class : 1
##
```

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

A fin de representar la precisión de la predicción utilizando el árbol de clasificación, graficaremos una curva ROC.

```
prob_tree <- predict(tree, newdata=test, type="prob")
ROC <- roc(test$Survived, prob_tree[,2], percent = T, smooth = F, auc = T,
 ci = F)
plot(ROC, print.auc = T, col = "darkblue", main="Curva ROC", grid = T)
```




Y donde el intervalo de confianza para el área bajo la curva es:

```
ci.auc(ROC)
```

```
## 95% CI: 77.57%-87.93% (DeLong)
```

Por otra parte, de https://www.kaggle.com/c/titanic (https://www.kaggle.com/c/titanic) se descargaron dos dataset. El primero es el que hemos venido utilizando para hacer el train y test. El segundo dataset es el siguiente:

```
glimpse(aplicar)
```

```
## Observations: 418
## Variables: 11
## $ PassengerId <int> 892, 893, 894, 895, 896, 897, 898, 899, 900, 901,
...
## $ Pclass      <int> 3, 3, 2, 3, 3, 3, 3, 2, 3, 3, 3, 1, 1, 2, 1, 2,
2,...
## $ Name        <chr> "Kelly, Mr. James", "Wilkes, Mrs. James (Ellen Ne
e...
## $ Sex         <chr> "male", "female", "male", "male", "female", "mal
e"...
## $ Age         <dbl> 34.5, 47.0, 62.0, 27.0, 22.0, 14.0, 30.0, 26.0, 1
8...
## $ SibSp       <int> 0, 1, 0, 0, 1, 0, 0, 1, 0, 2, 0, 0, 1, 1, 1, 1,
0,...
## $ Parch       <int> 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0,...
## $ Ticket      <chr> "330911", "363272", "240276", "315154", "310129
8",...
## $ Fare        <dbl> 7.8292, 7.0000, 9.6875, 8.6625, 12.2875, 9.2250,
7...
## $ Cabin       <chr> "", "", "", "", "", "", "", "", "", "", "", "",
"B...
## $ Embarked    <chr> "Q", "S", "Q", "S", "S", "S", "Q", "S", "C", "S",
...
```

Como puede observarse, no contiene a la variable Survived. Es decir, que una vez que utilicemos el árbol de clasificación no podremos elaborar una matriz de confusión ni tampoco evaluar el Accuracy; sin embargo, la predicción que obtengamos nos servirá para participar en la competencia de Kaggle.

Para aplicar el árbol modelado primero debemos convertir en factor algunas variables.

```
aplicar$Pclass <- factor(aplicar$Pclass)
aplicar$SibSp <- factor(aplicar$SibSp)
aplicar$Sex <- factor(aplicar$Sex)
```

Aplicamos el árbol de clasificación y guardamos la predicción en un nuevo dataset.

```
pred <- predict(tree, newdata=aplicar, type="class")
pred <- cbind(aplicar$PassengerId, pred)
colnames(pred) <- c("PassengerId","Survived")
pred <- data.frame(pred)
pred$Survived <- ifelse(pred$Survived==1, 0,
                        ifelse(pred$Survived==2, 1,
                               NA))
write.csv(pred, "prediccion.csv", row.names = F)
```

Las predicciones para este dataset son:

```
table(pred$Survived)
```

```
##
##   0   1
## 281 137
```

```
prop.table(table(pred$Survived))
```

```
##
##         0         1
## 0.6722488 0.3277512
```

# 6. Resolución del problema

Podemos afirmar que las variables más importantes para el árbol de clasificación son: Sex, Age y Pclass. Por tanto, son las que afectan más a la supervivencia. Sin embargo, cabe aclarar el sentido de esta conclusión, pues afectan en gran medida a la supervivencia la interacción de

ellas y no cada una de ellas por sí misma. Por ejemplo, el hecho de ser mujer por sí solo no es garantía de supervivencia, sin embargo, si se es mujer y se pertenece a la 1ra o 2da clase, entonces sí se tiene una buena probabilidad de sobrevivir.

Otro aspecto a mencionar es que las suposiciones producto del análisis de los gráficos del punto 4.1 han resultado ser verdaderas, salvo las relacionadas a las variables Parch y Embarked, que finalmente no fueron consideradas por el algoritmo del árbol de clasificación.

Y respecto a la precisión de predicción del árbol de clasificación, podemos afirmar que esta ha sido satisfactoria, pues alcanza un valor superior al 80%.

Por último, debemos mencionar que, una mejora del árbol de clasificación puede lograrse realizando una poda y/o construyendo un bosque.