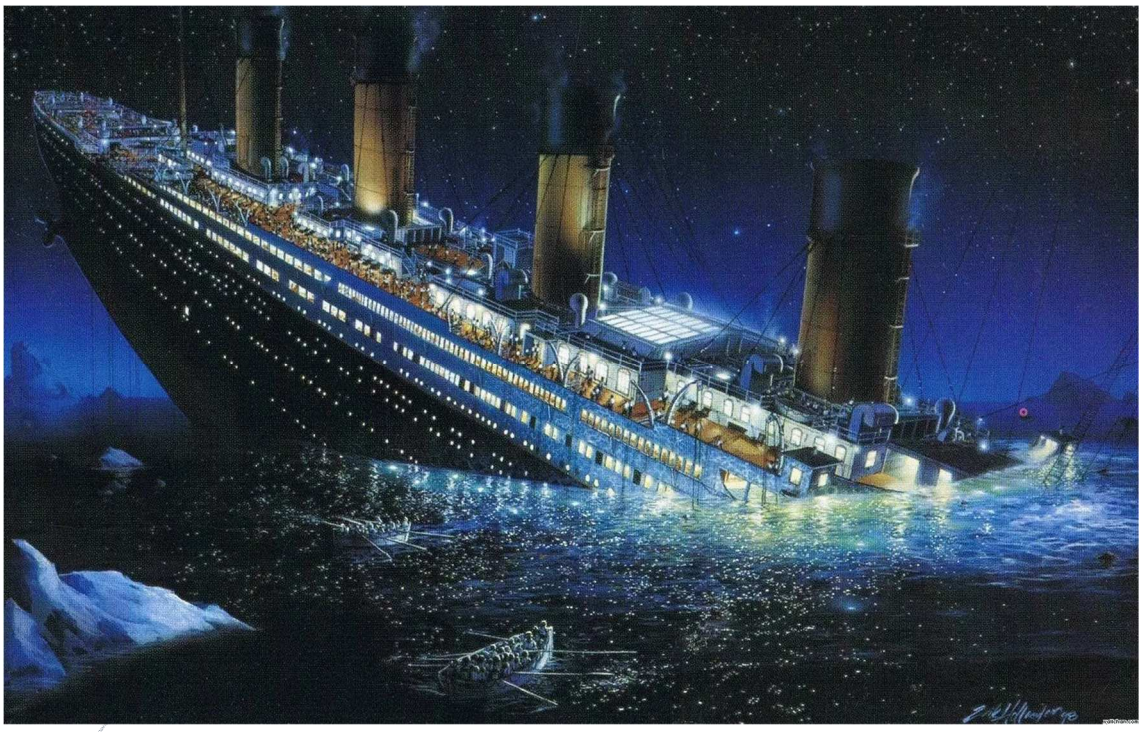
PRÁCTICA 2

Limpieza y Análisis de datos



Autores: Rosa M. Suárez y Javier Fernández Martínez

(TIPOLOGÍA Y CICLO DE VIDA DE LOS DATOS)

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

El dataset elegido es el conjunto de datos del desastre del Titanic obtenido de Kaggle (https://www.kaggle.com/c/titanic).

En este dataset tenemos información acerca de los pasajeros que iban en el Titanic, naufragado en 1912 y en el que murieron 1500 personas. Además de información descriptiva del tipo de pasajero tenemos también el indicador de si sobrevivió al desastre o no.

El objetivo es decidir si las variables aportadas son suficientes para crear un modelo predictivo que prediga la supervivencia o no de un pasajero. El dataset nos permite sacar algunas conclusiones de las características de los pasajeros supervivientes, incluso se podría intentar agruparlos en clusteres.

Para este cometido disponemos de 12 variables y 891 observaciones.

Descripción del dataset:

• Passengerld: Contador de pasajeros del 1 al 891.
• Survived: esta variable toma dos valores e indica si el pasajero sobrevivió. (0=" No", 1="Si").
• pClass: clase del ticket. 1=1st (clase alta), 2=2nd (clase media) y 3=3rd (clase baja).
• Name: nombre completo del pasajero.
• Sex: sexo del pasajero (Female o Male).
• Age: edad del pasajero. 3
•SibSp: número de hermanos/hermanas, hermanastros/hermanastros y marido o esposa del pasajero que también iban a bordo.
• Parch: número de hijas, hijos, padre y madre del pasajero a bordo del Titanic.
• Ticket: El número del ticket del pasajero.
• Fare: Es la tarifa del pasajero en dólares.
• Cabin: Código identificativo de la cabina.
• Embarked: el puerto en el que embarcó el pasajero (C = Cherbourg, Q = Queenstown, S = Southampton).

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

Del enlace indicado en Kaggle podemos descargar dos ficheros: train.csv y test.csv. Para nuestro proyecto vamos a trabajar únicamente con los datos de train.csv. El motivo es que el conjunto de datos que viene en el fichero test.csv no trae la variable objetivo, por tanto a la hora de construir modelos no podemos evaluarlos ya que no sabremos si el modelo está funcionando correctamente o no. Podríamos llegar a utilizar esos datos para hacer imputaciones de valores (si vamos a obtener medias, etc) pero en principio nosotros no lo vamos a utilizar. Es decir nosotros vamos a realizar toda la integración y limpieza de datos sobre el dataframe generado a partir de train.csv.

Luego más adelante, nosotros dividiremos nuestro dataframe (compuesto únicamente por registros del fichero train.csv) en 2 dataframe "train" y "test" que utilizaremos para entrenar y evaluar a los modelos construidos (pero no confundir este "test" con el fichero test.csv que no utilizamos).

Una vez leídos los datos en un dataframe (al que llamamos "data") podemos ver su dimensionalidad, los nombres y tipos de las variables.

```
> #Vemos las dimensiones del dataframe: Tenemos 891 registros con 12 variables
> #La descripción de cada una de las variables se puede encontrar
> #en el pdf complementario
> dim.data.frame(data)
[1] 891  12
```

```
> sapply(data, function(x) class(x))
PassengerId    Survived      Pclass        Name         Sex         Age       SibSp
  "numeric"   "numeric"   "numeric" "character" "character"   "numeric"   "numeric"
      Parch      Ticket        Fare       Cabin    Embarked
  "numeric" "character"   "numeric" "character" "character"
```

Además podemos hacer una comprobación de si hay registros duplicados:

```
#1) Comprobemos la unicidad de los registros:
#Con el comando unique() eliminamos las muestras duplicadas  y vemos que sigue teniendo
#891 muestras. Por tanto, no hay duplicados. OK
data_unique<-unique(data)
dim.data.frame(data_unique)
1] 891  12
remove(data_unique)
```

En este caso hemos comprobado que la dimensionalidad ha quedado exactamente igual que cuando cargamos el dataset, por tanto no hay registros duplicados (nos referimos a completos duplicados).

Y podemos ver también una muestra de los primeros registros de nuestro dataframe:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
	<dbl>	<dbl>	<dbl>	<chr>	<chr>	<dbl>	<dbl>	<dbl>	<chr>	<dbl>	<chr>	<chr>
1	1	0	3	Braund~	male	22	1	0	A/5 2~	7.25	NA	S
2	2	1	1	Cuming~	fema~	38	1	0	PC 17~	71.3	C85	C
3	3	1	3	Heikki~	fema~	26	0	0	STON/~	7.92	NA	S
4	4	1	1	Futrel~	fema~	35	1	0	113803	53.1	C123	S
5	5	0	3	Allen,~	male	35	0	0	373450	8.05	NA	S
6	6	0	3	Moran,~	male	NA	0	0	330877	8.46	NA	Q

Hay algunas variables que no nos van a interesar por ser identificativas de cada registro: Se trata del "**PassengerId**", "**Ticket**" y "**Name**". No nos interesa tener unívocamente identificado cada caso para realizar ningún tipo de análisis, no aportan nada. Pero sí que es cierto que la variable "**Ticket**" nos va a servir para conocer el precio unitario que han pagado los pasajeros, ya que la variable "**Fare**" es la tarifa pagada en el ticket, pero dentro del mismo ticket pueden estar incluidas varias personas. Entonces inicialmente vamos a obtener el nº de personas que están incluidas en un mismo ticket.

```
#Pero la variable Ticket aporta la agrupación de las personas que viajan con un mismo ticket,
#es decir podemos calcular el precio por persona diviendo Fare / num

#Agrupamos por Ticket y Fare y obtenemos el nº de filas
nrow(table(data$Ticket, data$Fare)) |#Devuelve 681 filas
length(unique(data$Ticket))          #Devuelve 681 valores únicos de Ticket
#También se puede obtener así
data %>%
  group_by(Ticket, Fare) %>%
  filter(row_number() == 1)

#Concluimos que se puede obtener el Precio por Persona, dividiendo Fare / integrantes del Ticket
ticket_personas <- as.data.frame(data %>%
                                 group_by(Ticket) %>%
                                 dplyr::summarize(PersonasTicket=n()))
#Tenemos un nuevo dataframe con el Ticket y nº de registros (personas) para ese ticket
df_status(ticket_personas)   #Las columnas son Ticket y PersonasTicket
```

Hemos obtenido un nuevo dataframe (llamado "ticket_personas") con las variables "**Ticket**" y "**PersonasTicket**".

```
         variable q_zeros p_zeros q_na p_na q_inf p_inf      type unique
1          Ticket       0       0    0    0     0     0 character    681
2 PersonasTicket       0       0    0    0     0     0   integer      7
```

Este nuevo dataframe (ticket_personas) lo vamos a combinar con nuestro dataframe original (un join) a través del nº de ticket (variable "**Ticket**") para poder incorporar el nº de personas al dataframe original.

```
#Ahora vamos a hacer un Merge entre nuestro Dataset original y el de Tickets/Personas
#Lo hacemos a través de la columna Ticket (aparece la nueva columnas PersonasTicket que usamos para
#calular, y luego eliminaremos)
data <- merge(data, ticket_personas, by = "Ticket")
remove(ticket_personas)
```

Y una vez que lo hemos juntado, podemos eliminar el dataframe generado ("ticket_personas").
Ahora tenemos en nuestro dataframe original "data" una nueva variable "**PersonasTicket**". Esa variable será utilizada para dividir la tarifa del ticket "**Fare**" entre esta nueva columna incorporada. Con ello obtenemos una nueva variable que le llamamos "**Price**" y es el precio por persona que se paga en el billete (obteniendo un precio por persona lineal por billete).

```
#Y ahora generamos la nueva columna de precio (Price para mantener el idioma de columnas)
data$Price <- data$Fare / data$PersonasTicket
```

Y ya una vez generada la variable que queríamos, ahora sí procedemos a eliminar aquellas variables que decíamos que íbamos a eliminar (además ahora también añadimos "**PersonasTicket**" a la lista de variables a eliminar, ya que ya la hemos utilizado para generar "**Price**").

```
#Y después de esto ahora sí finalmente nos disponemos a eliminar aquellas columnas que no vamos
#a utilizar en nuestro análisis
data<-select(data,-PassengerId,-Name,-Ticket,-PersonasTicket)
```

Otra variable que podría ser interesante es la variable "**FamilySize**" que englobla a las variables "**Parch**" y "**SubSp**" ya que estas variables se refieren a los lazos familizares. Vamos a incluir al pasajero en cuestión por tanto esta variable siempre vale como mínimo 1.

```
#Vamos a definir el tamaño familiar de la gente que viaja, sumando SibSp y Parch
#SibSp: (Number of Siblings/Spouses Aboard)
#   Sibling: Brother, Sister, Stepbrother, or Stepsister of Passenger Aboard Titanic
#   Spouse: Husband or Wife of Passenger Aboard Titanic
#Parch: (Number of Parents/Children Aboard)
#   Parent: Mother or Father of Passenger Aboard Titanic
#   Child: Son, Daughter, Stepson, or Stepdaughter of Passenger Aboard Titanic

#Incluimos al pasajero en cuestión. Es decir siempre va de 1 a N la columna FamilySize
data$FamilySize = data$SibSp + data$Parch + 1
```

Tenemos algunas variables que, aunque a priori aparecen como "numeric" o "character" deberíamos convertir a "factor". Estas variables son: "**Survived**", "**Pclass**", "**Embarked**" y "**Sex**". Tienen un número finito de valores y aunque puedan ser numéricas, ese número no nos aporta información:

```
data$Sex<-as.factor(data$Sex)
data$Survived<-as.factor(data$Survived)
data$Pclass<-as.factor(data$Pclass)
data$Embarked<-as.factor(data$Embarked)
```

La variable "Cabin" también es de tipo "character", pero como es candidata a ser eliminada (lo veremos más adelante), por ahora no le hacemos ningún tipo de transformación.

Como numéricos quedarían "**Age**", "**SubSp**", "**Parch**", "**FamilySize**", "**Fare**" y "**Price**".

```
> summary(data)
 Survived Pclass      Sex           Age            SibSp           Parch
 0:549    1:216   female:314   Min.   : 0.42   Min.   :0.000   Min.   :0.0000
 1:342    2:184   male  :577   1st Qu.:20.12   1st Qu.:0.000   1st Qu.:0.0000
          3:491                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
      Fare           Cabin           Embarked       Price          FamilySize
 Min.   :  0.00   Length:891         C   :168   Min.   :  0.000   Min.   : 1.000
 1st Qu.:  7.91   Class :character   Q   : 77   1st Qu.:  7.763   1st Qu.: 1.000
 Median : 14.45   Mode  :character   S   :644   Median :  8.850   Median : 1.000
 Mean   : 32.20                      NA's:  2   Mean   : 17.789   Mean   : 1.905
 3rd Qu.: 31.00                                 3rd Qu.: 24.288   3rd Qu.: 2.000
 Max.   :512.33                                 Max.   :221.779   Max.   :11.000
```

3. Limpieza de los datos.

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

Para la gestión de nulos y vacíos, además del comando summary ya mostrado anteriormente nos será muy útil ver la salida de "df_status", que nos muestra un resumen del estado de nuestras 8 variables actuales:

```
> df_status(data)
     variable q_zeros p_zeros q_na  p_na q_inf p_inf      type unique
1    Survived     549   61.62    0  0.00     0     0    factor      2
2      Pclass       0    0.00    0  0.00     0     0    factor      3
3         Sex       0    0.00    0  0.00     0     0    factor      2
4         Age       0    0.00  177 19.87     0     0   numeric     88
5       SibSp     608   68.24    0  0.00     0     0   numeric      7
6       Parch     678   76.09    0  0.00     0     0   numeric      7
7        Fare      15    1.68    0  0.00     0     0   numeric    248
8       Cabin       0    0.00  687 77.10     0     0 character    147
9    Embarked       0    0.00    2  0.22     0     0    factor      3
10      Price      15    1.68    0  0.00     0     0   numeric    248
11 FamilySize       0    0.00    0  0.00     0     0   numeric      9
```

En esta table de resultados vemos que "**Age**" tiene un 19.87% de nulos y "**Cabin**" tiene más de un 77%. Estos son los casos más llamativos que tenemos que solucionar. También tenemos 2 nulos en "**Embarked**".

Respecto a los ceros: Tenemos un alto número de ceros en "**SibSp**" y "**Parch**", pero son valores válidos para estas variables, pues cuentan número de acompañantes (familiares) del pasajero. Hay también un % pequeño de ceros en "**Fare**" (tarifa), podría tener algún sentido (podría ser por ejemplo algún premio y por tanto el ticket no tenía coste) por lo que en principio vamos a dejar presentes estos ceros.

Empezamos por el a priori más sencillo de solucionar, concretamente la variable "**Embarked**" que solamente tiene 2 valores nulos. Dado que son muy pocos casos y además la variable solo toma 3 valores vamos a imputar el valor que más se repite: Embarked=S (Southampton).
Podemos hacerlo así:

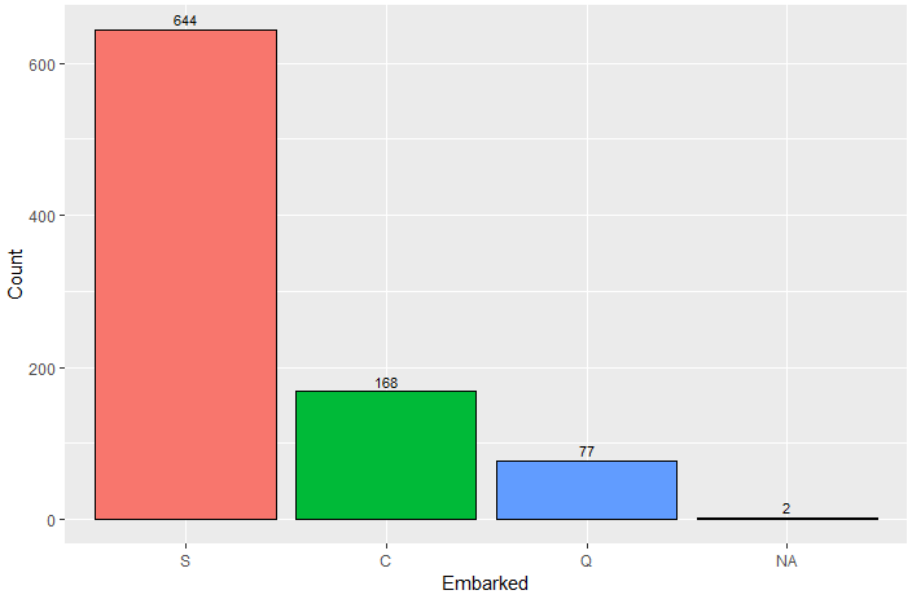
```
> data %>% group_by(Embarked) %>% count(Embarked)
# A tibble: 4 x 2
# Groups:   Embarked [4]
  Embarked     n
  <fct>    <int>
1 C          168
2 Q           77
3 S          644
4 NA           2
```

O también así:

```
> dat_Embarked = sort(table(data$Embarked,useNA = "ifany"),decreasing = TRUE)
> dat_Embarked

   S    C    Q <NA>
 644  168   77    2
```

Y gráficamente:

```
> dat_plot = as.data.frame(dat_Embarked)
> ggplot(dat_plot, aes(x=Var1, y=Freq, fill=Var1)) +
+   geom_bar(stat = "identity", color="black") +
+   geom_text(aes(label=Freq), vjust=-0.4, color="black", size=3) +
+   labs(x='Embarked', y='Count') + theme(legend.position='none')
```



Y por tanto finalmente asignamos el valor más frecuente ("S" correspondiente a Southampton).

```
#Es decir asignar el más frecuente (o sea "S")
data$Embarked[is.na(data$Embarked)]<-names(dat_Embarked[1])
```

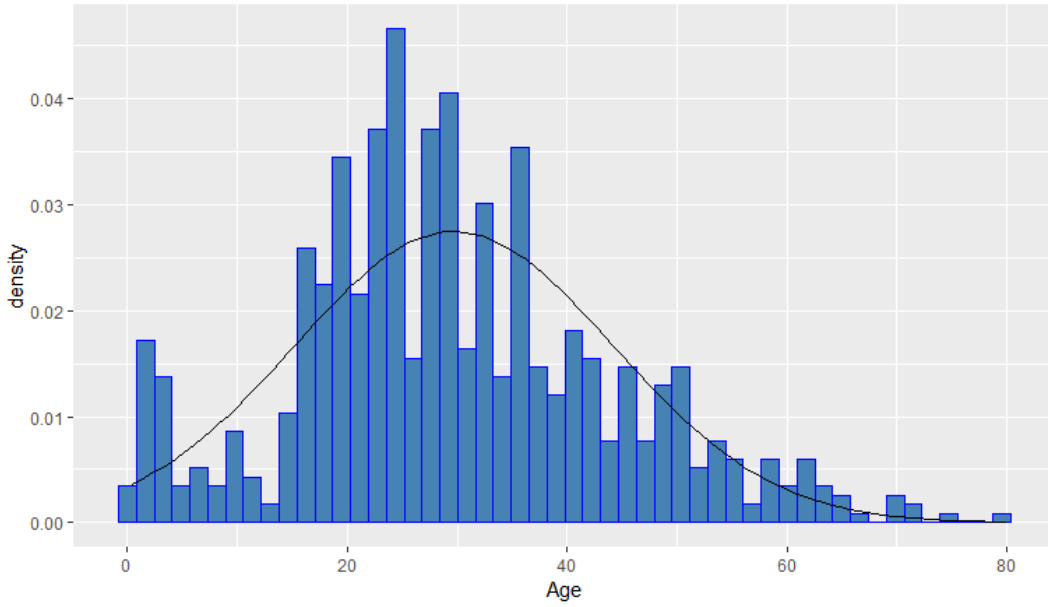
Respecto a la variable "**Cabin**" dado que el número de nulos es muy elevado, un 77%, hemos decidido eliminar esta variable. Tendríamos que imputar valores a una parte muy importante del dataset (con su consiguiente posible error), y además no parece que pueda ser una variable determinante para predecir la supervivencia.

Y llegamos a la variable "**Age**" que tenía algo más de un 19% de valores nulos (concretamente 177 registros). Para solucionar estos 177 casos nulos parecería sencillo calcular la media del resto de registros, e imputarla, pero vamos a estudiar si podemos delimitar un poco la media a aplicar para obtener un resultado más depurado.

Lo que vamos a hacer son diferentes simulaciones de imputación de valores, concretamente vamos a hacer 4, y luego nos vamos a quedar con una sola de ellas.

De entrada, y antes de imputar ningún valor, vamos a intentar ver cómo es la distribución de la variable "**Age**" (por supuesto solamente teniendo en cuenta en los registros donde hay valores, es decir algo más del 80% del dataframe):

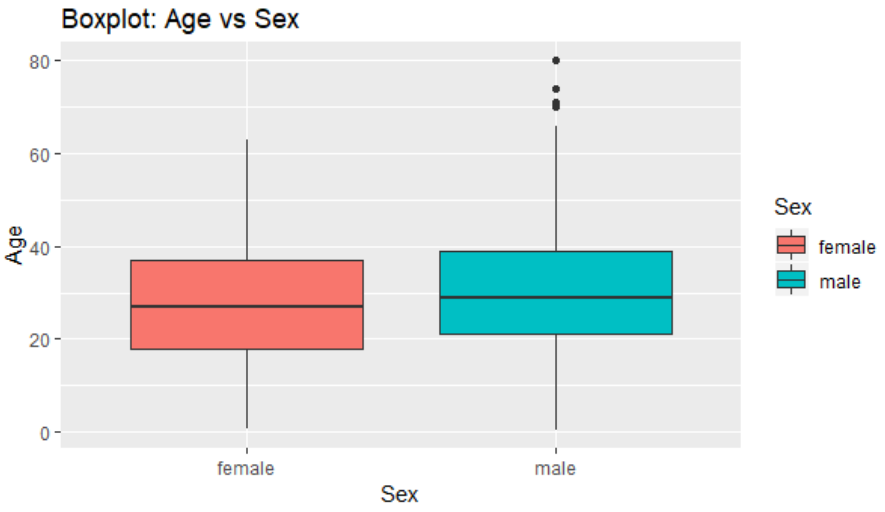
```
#Nos quedamos inicialmente un data.frame con los registros que
#no tienen valores nulos en Age
data_NoNA = data[which(!is.na(data$Age)),]

#Comprobamos gráficamente como se distribuye la variable Age
#(en el dataset que no son nulos)
ggplot(data_NoNA, aes(Age)) +
  geom_histogram(aes(y = ..density..), bins=50, fill="steelblue", color="blue") +
  stat_function(fun = dnorm, args = list(mean = mean(data_NoNA$Age),
                                         sd = sd(data_NoNA$Age)))
```



Vamos a comprobar si vemos alguna relación entre la edad y algunas del resto de las variables, para tenerlas en cuenta a la hora de imputar valores.

"**Age**" en función de "**Sex**":

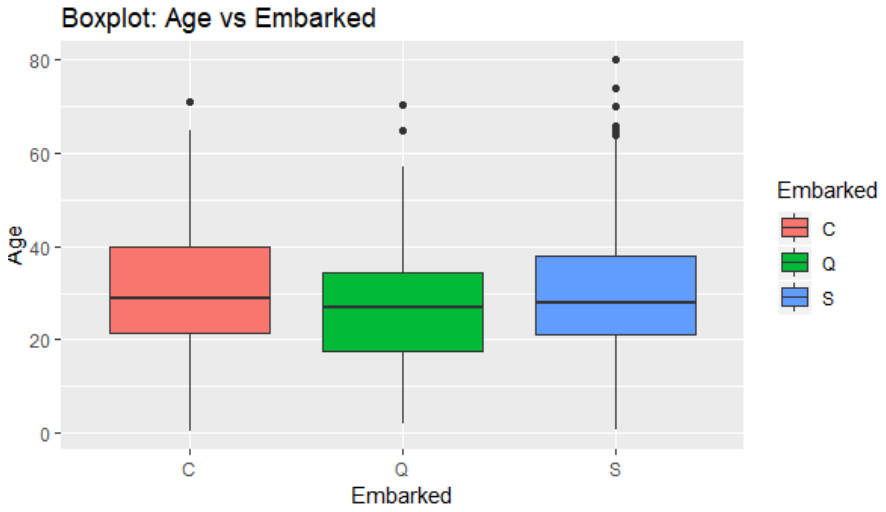
```
#Comprobamos gráficamente la relación entre la edad y el género
#(no se aprecian diferencias)
titulo <- 'Age vs Sex'
ggplot(data_NoNA, aes(y=Age, x=Sex, fill=Sex)) + geom_boxplot() +
  labs(title = paste0('Boxplot: ',titulo)) + ylab("Age") + xlab("Sex")
```



Podemos ver que no se aprecian casi diferencias de edad en función del género.

"**Age**" en función de "**Embarked**":

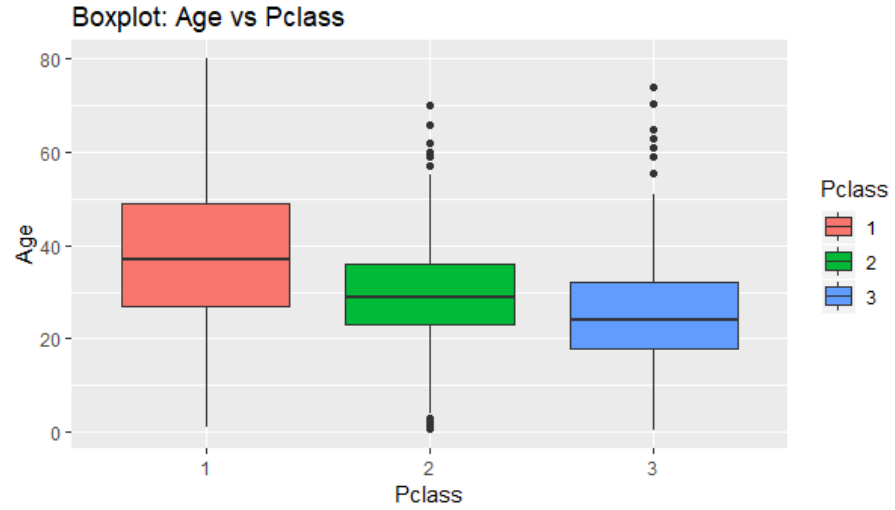
```
#Comprobamos gráficamente la relación entre la edad y el puerto de Embarque
#(no se aprecian casi diferencias)
titulo <- 'Age vs Embarked'
ggplot(data_NoNA, aes(y=Age, x=Embarked, fill=Embarked)) + geom_boxplot() +
  labs(title = paste0('Boxplot: ',titulo)) + ylab("Age") + xlab("Embarked")
```



En cuanto a las medias de edad, tampoco parece significativo el hecho de que haya salido de un puerto u otro.

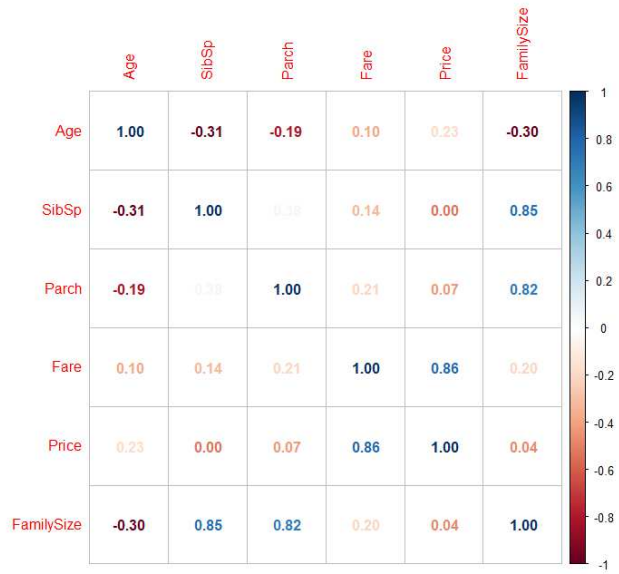
"**Age**" en función de "**Pclass**":

```
#En cambio en el siguiente gráfico vemos que la edad tiene bastante relación
#con la clase #en que se viaja, puesto que a mayor edad  mejor clase
#(clase alta). Por lo que según esto podríamos hacer una imputación
#por clase y el resultado podría ser más acertado:
titulo <- 'Age vs Pclass'
ggplot(data_NoNA, aes(y=Age, x=Pclass, fill=Pclass)) + geom_boxplot() +
  labs(title = paste0('Boxplot: ',titulo)) + ylab("Age") + xlab("Pclass")
```



Hemos detectado una relación entre la edad y la clase en que viajaban los pasajeros, los pasajeros de clase 1 (alta) tenían generalmente mayor edad que los de clase 2 (media) e igualmente sucede con los de clase 3 (baja).

Y también hacemos una comprobación de la correlación de las variables numéricas.



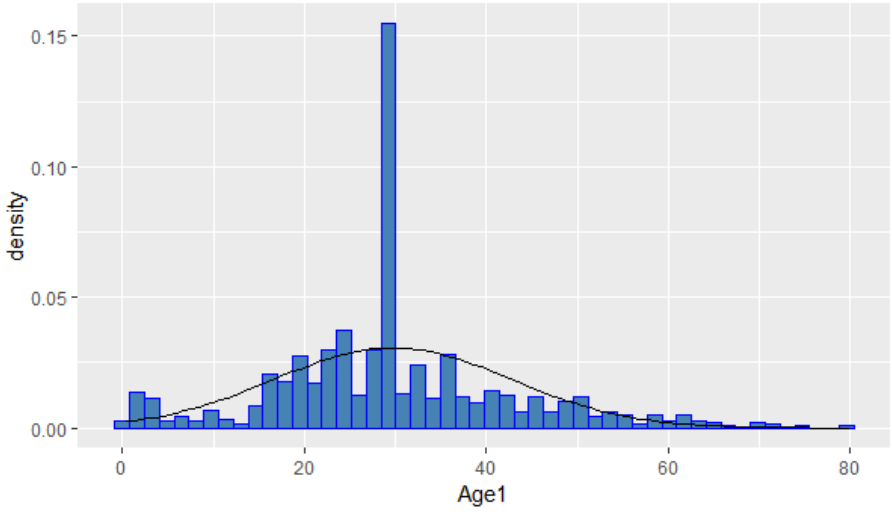
Podemos ver que existe correlación negativa con "**SibSp**" y con "**Parch**" (también con "**FamilySize**", pero eso es completamente normal porque "**FamilySize**" es una transformación lineal de las otras 2). También hay correlación positiva con "**Price**", pero entendemos que podemos tener en cuenta la parte familiar por el tema de esposa, hijos, hermanos, etc. puede tener efecto en la edad.

Entonces empezamos con los 4 casos que comentábamos para la imputación de valores en la variable "Age":

Caso 1: Imputar la media de "**Age**" a todos los elementos faltantes

```
#Caso 1: Imputando la media a todos los elementos faltantes
data$Age1[is.na(data$Age1)] <- mean(data_NoNA$Age)

ggplot(data, aes(Age1)) +
  geom_histogram(aes(y = ..density..), bins=50, fill="steelblue", color="blue") +
  stat_function(fun = dnorm, args = list(mean = mean(data$Age1),
                                         sd = sd(data$Age1)))
```

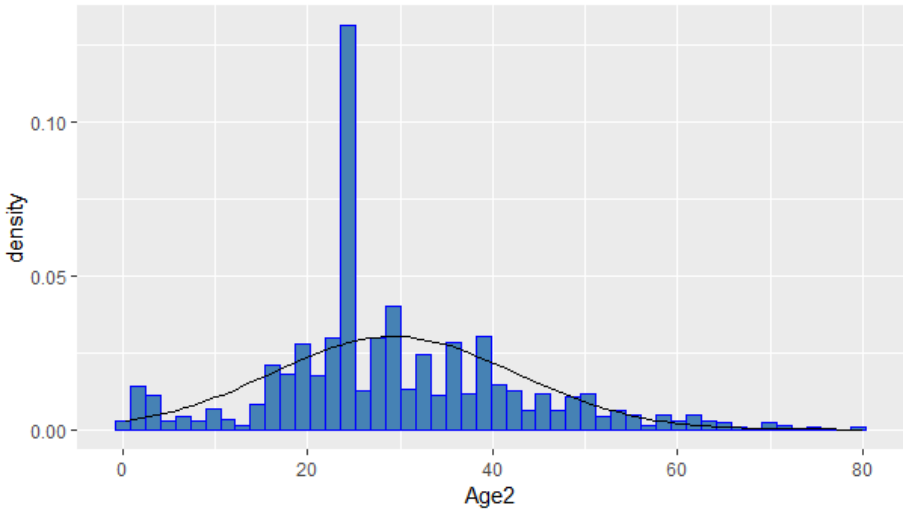
La distribución de esta nueva variable "Age1" sería la siguiente:



Caso 2: Imputar la media de "**Age**", pero por cada una de las clases ("**Pclass**") ya que hemos visto que hay una relación entre ambas variables.

```
#Caso 2: Imputando la media pero por cada Pclass
data$Age2[is.na(data$Age2)&data$Pclass==1]<-mean(data$Age2[!is.na(data$Age2)&data$Pclass==1])
data$Age2[is.na(data$Age2)&data$Pclass==2]<-mean(data$Age2[!is.na(data$Age2)&data$Pclass==2])
data$Age2[is.na(data$Age2)&data$Pclass==3]<-mean(data$Age2[!is.na(data$Age2)&data$Pclass==3])

ggplot(data, aes(Age2)) +
  geom_histogram(aes(y = ..density..), bins=50, fill="steelblue", color="blue") +
  stat_function(fun = dnorm, args = list(mean = mean(data$Age2), sd = sd(data$Age2)))
```

La distribución de esta nueva variable "Age2" sería la siguiente:



Caso 3: Imputar la datos en "**Age**", pero teniendo en cuenta "**Pclass**", "**Parch**" y "**SibSp**", ya que hemos visto correlación entre las variables.

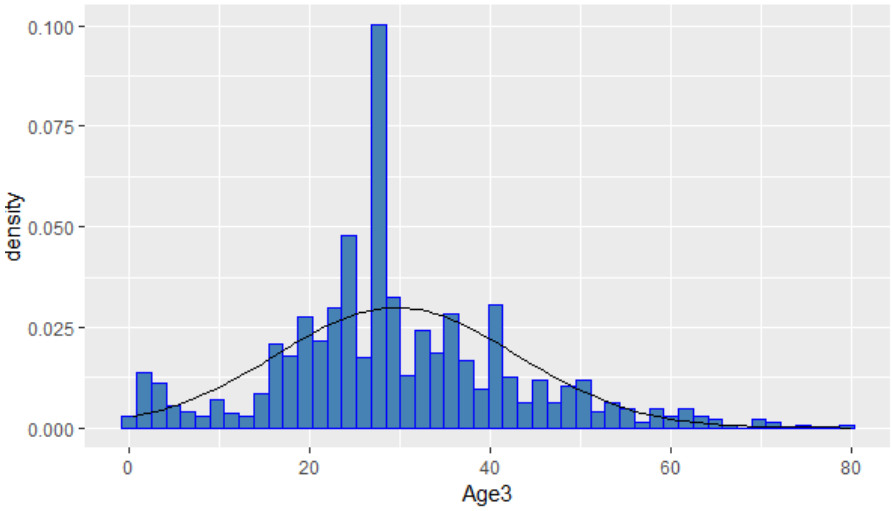
Para este caso lo que hacemos es tener en cuenta las 3 variables y generar una agrupación de datos calculando la media para las combinaciones (recordemos que son variables discretas).

Una vez que obtenemos esos valores medios, los imputamos. Luego verificamos si quedó algún valor sin imputar (que serán muy pocos) y para esos pocos casos faltantes, imputar los valores por cada clase (como hicimos en el caso 2).

```
#Caso 3: Imputando datos de Age, teniendo en cuenta Pclass, Parch y SibSp

medias_clase_fam <- as.data.frame(data_NoNA %>%
                                  group_by(Pclass, SibSp, Parch) %>%
                                  dplyr::summarize(Media_clase_fam=mean(Age)))

#Ahora vamos a hacer un Merge entre nuestro Dataset original y el de Medias
#(por Pclass,SibSp y Parch)
#Lo hacemos a través de las columnas usadas en la agrupación
#Aparecerá una nueva columna (llamada Media_clase_fam)
#El merge es un LEFT JOIN ya que puede que no existan todas las combinaciones en
#medias_clase_fam
data <- merge(data, medias_clase_fam, by = c("Pclass", "SibSp", "Parch"), all.x = TRUE)
data$Age3[is.na(data$Age3)] <- data$Media_clase_fam[is.na(data$Age3)]
df_status(data)
#Como es posible que nos hayan quedado alguno sin poder asignar (por no existir la combinación
# Pclass + SibSp + Parch) entonces a los que faltan (7) los asignamos directamente
#por la Pclass sin tener en cuenta los otros 2 valores
data$Age3[is.na(data$Age3)&data$Pclass==1]<-mean(data$Age3[!is.na(data$Age3)&data$Pclass==1])
data$Age3[is.na(data$Age3)&data$Pclass==2]<-mean(data$Age3[!is.na(data$Age3)&data$Pclass==2])
data$Age3[is.na(data$Age3)&data$Pclass==3]<-mean(data$Age3[!is.na(data$Age3)&data$Pclass==3])
```

Después de la imputación de valores en "**Age3**" nos queda la siguiente distribución:

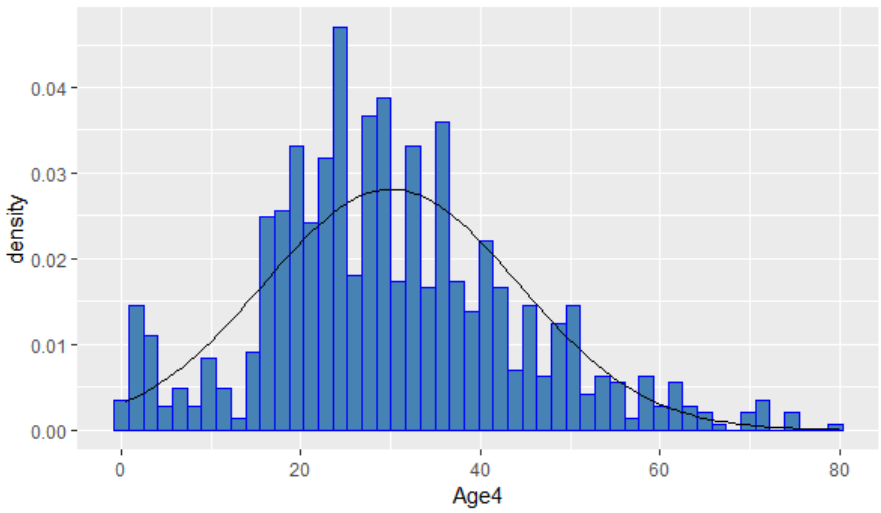
```
#Ya tenemos todos los valores de Age (caso 3) imputados.
ggplot(data, aes(Age3)) +
  geom_histogram(aes(y = ..density..), bins=50, fill="steelblue", color="blue") +
  stat_function(fun = dnorm, args = list(mean = mean(data$Age3), sd = sd(data$Age3)))
```



Caso 4: Imputando datos de "**Age**", con MICE (Multivariate Imputation via Chained Equations). En este caso se predicen los valores de Age, con el resto de valores observados (usamos para este caso "Pclass", "SibSp", "Parch", "Sex" y "Age").

```
#Caso 4: Imputando datos de Age, con MICE (Multivariate Imputation via Chained Equ
#En este caso se predicen los valores con el resto de valores observados
columnas <- c('Pclass', 'SibSp', 'Parch', 'Sex', 'Age')
mice_imputar <- mice(data = data[,columnas], method = "rf")
mice_completo <- mice::complete(mice_imputar)
data$Age4[is.na(data$Age4)]<-mice_completo$Age[is.na(data$Age4)]
df_status(data)
#Ya tenemos todos los valores de Age (caso 4) imputados.
ggplot(data, aes(Age4)) +
  geom_histogram(aes(y = ..density..), bins=50, fill="steelblue", color="blue") +
  stat_function(fun = dnorm, args = list(mean = mean(data$Age4),
                                         sd = sd(data$Age4)))
```

Después de la imputación nos queda la siguiente distribución:



Y estos son los resultados resumen de las variables “Age” que hemos calculado:

```
> summary(data$Age)  #Este es el original
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
   0.42   20.12   28.00   29.70   38.00   80.00     177
> summary(data$Age1)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.42   22.00   29.70   29.70   35.00   80.00
> summary(data$Age2)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.42   22.00   26.00   29.29   37.00   80.00
> summary(data$Age3)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.42   22.00   28.24   29.60   37.00   80.00
> summary(data$Age4)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.42   21.00   29.00   30.04   38.00   80.00
```

Nosotros nos decantamos por la 4ª imputación, ya que la distribución se parece mucho más a la original, hemos tenido en cuenta más variables, y los valores de media y mediana no se ven muy resentidos.

Viendo nuevamente los resultados con “summary”, hemos eliminado los nulos y tenemos un nuevo valor de media y mediana para “Age”.

```
> summary(data)
 Pclass      SibSp           Parch        Survived     Sex           Age
 1:216   Min.   :0.000   Min.   :0.0000   0:549    female:314   Min.   : 0.42
 2:184   1st Qu.:0.000   1st Qu.:0.0000   1:342    male  :577   1st Qu.:21.00
 3:491   Median :0.000   Median :0.0000                         Median :29.00
         Mean   :0.523   Mean   :0.3816                         Mean   :30.04
         3rd Qu.:1.000   3rd Qu.:0.0000                         3rd Qu.:38.00
         Max.   :8.000   Max.   :6.0000                         Max.   :80.00
      Fare         Embarked     Price           FamilySize
 Min.   :  0.00   C:168    Min.   :  0.000   Min.   : 1.000
 1st Qu.:  7.91   Q: 77    1st Qu.:  7.763   1st Qu.: 1.000
 Median : 14.45   S:646    Median :  8.850   Median : 1.000
 Mean   : 32.20            Mean   : 17.789   Mean   : 1.905
 3rd Qu.: 31.00            3rd Qu.: 24.288   3rd Qu.: 2.000
 Max.   :512.33            Max.   :221.779   Max.   :11.000
```

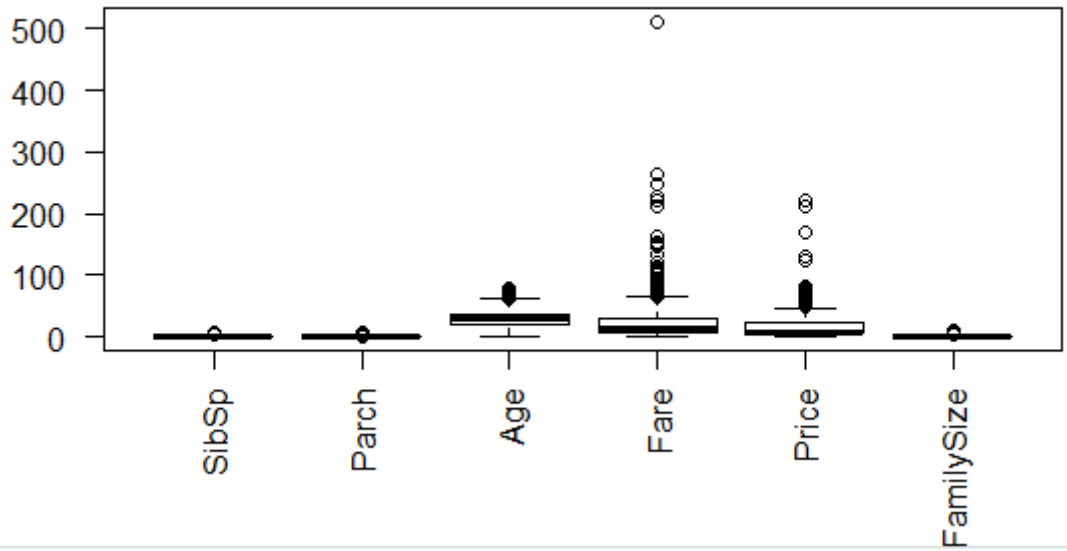
Y con “df_status” vemos también como nos han quedado los datos después de eliminar variables, y de imputar valores en las variables que le faltaban algunos valores. El dataset queda libre de nulos, y con los ceros que hemos aceptado que tiene que mantener y que tienen sentido.

```
> df_status(data)
      variable q_zeros p_zeros q_na p_na q_inf p_inf    type unique
1       Pclass       0    0.00    0    0     0     0  factor      3
2        SibSp     608   68.24    0    0     0     0 numeric      7
3        Parch     678   76.09    0    0     0     0 numeric      7
4     Survived     549   61.62    0    0     0     0  factor      2
5          Sex       0    0.00    0    0     0     0  factor      2
6          Age       0    0.00    0    0     0     0 numeric     88
7         Fare      15    1.68    0    0     0     0 numeric    248
8     Embarked       0    0.00    0    0     0     0  factor      3
9        Price      15    1.68    0    0     0     0 numeric    248
10 FamilySize       0    0.00    0    0     0     0 numeric      9
```

3.2. Identificación y tratamiento de valores extremos

Se considera un valor extremo o outlier a un valor fuera de rango. Son valores que se salen de la escala esperada visualizando el resto de las observaciones. A lo largo de los años la identificación de valores extremos ha dado lugar a muchas y controvertidas discusiones. Generalmente en la actualidad el criterio más habitual es considerar un valor extremo a aquel que se encuentra alejado de la media 3 veces la desviación típica. De todos modos, es necesario conocer la naturaleza de los datos para identificarlos correctamente y diferenciar un outiler de un valor realmente posible.

En nuestro caso vamos a comenzar por separar aquellas variables que son numéricas y verlas en un gráfico bloxplot:



Pero de esas variables solamente 3 son variables continuas, concretamente "**Age**", "**Price**" y "**Fare**".

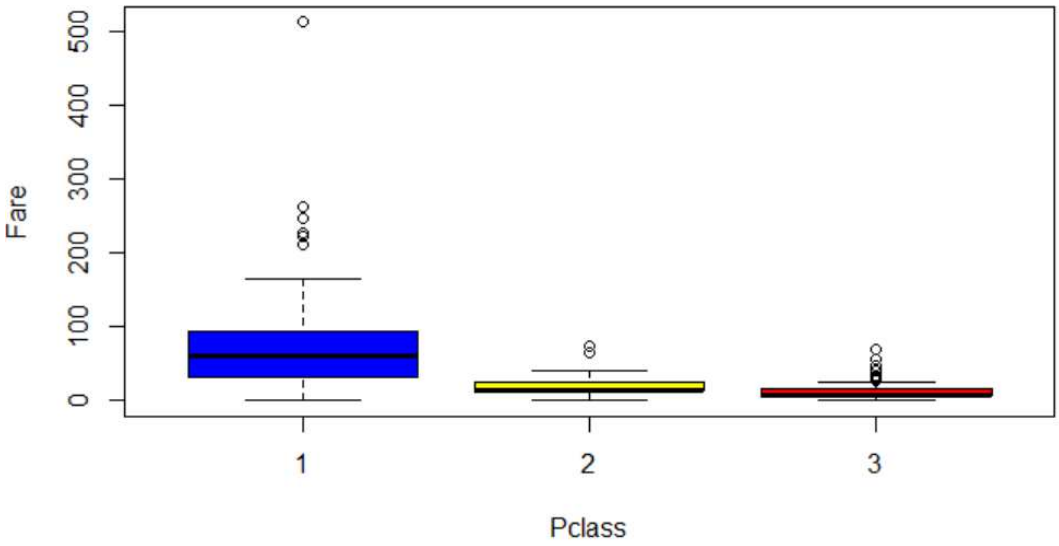
Aparentemente parece que hay valores extremos en la variable "**Fare**", veamos con el criterio de 3 veces la desviación típica cuántos nos identifican:

```
> data_out<-as.data.frame(data_num$Fare)
> data_out$outlier<-FALSE
> for (i in 1:ncol(data_out)-1){
+   columna = data_out[,i]
+   if (is.numeric(columna)){
+     media = mean(columna)
+     desviacion = sd(columna)
+     data_out$outlier = ( columna>(media+3*desviacion) | columna<(media-3*desviacion))
+   }
+ }
> # Marcamos los TRUE y FALSE
> table(data_out$outlier)

FALSE  TRUE
  871    20
```

Identificamos con este criterio 20 posibles outliers, pero los valores que nos indica el boxplot.stats y teniendo en cuenta que el máximo es 512, no parecen exagerados.

```
> boxplot.stats(data$Fare)$out
  [1]  86.5000  86.5000  86.5000 151.5500 227.5250 227.5250 211.3375  80.0000
  [9]  80.0000 135.6333  79.2000  79.2000  93.5000 512.3292  69.3000 221.7792
 [17] 227.5250 134.5000 110.8833 135.6333  83.1583 135.6333 106.4250 153.4625
 [25]  76.2917  78.8500  77.9583  69.3000  79.2000 146.5208 512.3292  76.7292
 [33] 153.4625 153.4625  77.2875  77.2875 512.3292  83.1583 211.3375 247.5208
 [41] 211.3375 247.5208 164.8667  71.0000  79.6500 113.2750 110.8833 211.5000
 [49]  81.8583  76.7292  82.1708  90.0000  66.6000  89.1042  77.9583  90.0000
 [57] 113.2750 113.2750  66.6000 106.4250  78.2667  83.4750  83.4750 133.6500
 [65]  89.1042 227.5250  91.0792 108.9000  77.9583  90.0000  82.1708  75.2500
 [73]  78.8500  71.2833 146.5208  76.7292  91.0792  78.2667 108.9000  79.6500
 [81]  71.0000  79.6500 164.8667 110.8833 134.5000 110.8833  79.2000  83.1583
 [89]  93.5000 120.0000 120.0000 120.0000 151.5500 151.5500 151.5500 120.0000
 [97] 263.0000 133.6500  90.0000 262.3750 262.3750 263.0000 263.0000 263.0000
[105]  73.5000  73.5000  73.5000  73.5000  73.5000  69.5500  69.5500  69.5500
[113]  69.5500  69.5500  69.5500  69.5500
```

Además, los valores están bien distribuidos, los más altos en las clases altas. Con lo que no vamos a realizar ninguna acción con estos outliers, puede que incluso aporten información importante, aunque no sea para nuestro objetivo, sí podrían ser útiles para resolver algún otro problema:



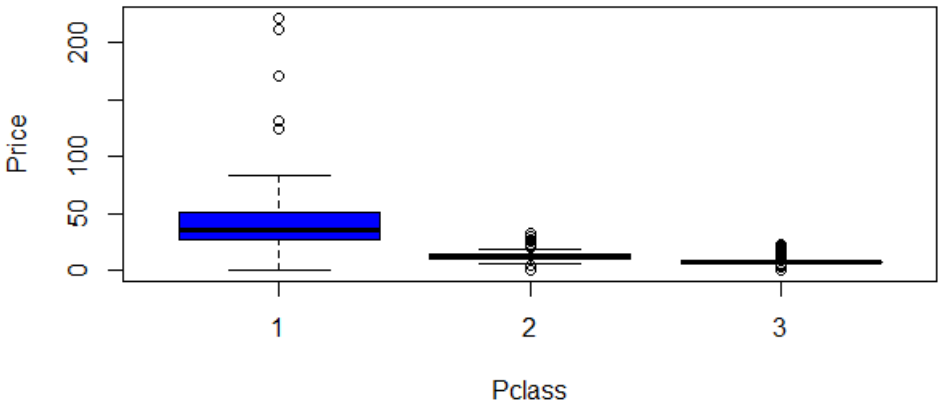
Si hacemos lo mismo con "**Price**", y comprobamos cuantos salen fuera 3 veces desviación típica, salen 13 outliers:

```
> data_out<-as.data.frame(data_num$Price)
> data_out$outlier<-FALSE
> for (i in 1:ncol(data_out)-1){
+   columna = data_out[,i]
+   if (is.numeric(columna)){
+     media = mean(columna)
+     desviacion = sd(columna)
+     data_out$outlier = ( columna>(media+3*desviacion) | columna<(media-3*desviacion))
+   }
+ }
> # Marcamos los TRUE y FALSE
> table(data_out$outlier)

FALSE  TRUE
  878    13
```

Con boxplot.stats en cambio nos devuelve los siguientes outliers, donde el valor máximo es 221.

```
> boxplot.stats(data$Price)$out
 [1]  56.88125  56.88125  70.44583  50.00000  50.49580 170.77640  49.50420
 [8] 221.77920  56.88125  67.25000  53.21250  51.86250  51.15417  76.29170
[15]  79.20000  73.26040 170.77640  49.50420  51.15417  61.97920  51.15417
[22] 170.77640  70.44583  61.37920 123.76040  70.44583 123.76040  63.35830
[29]  82.43335  49.50000 211.50000  81.85830  51.86250  59.40000  53.21250
[36]  66.82500  61.17500  53.10000  56.88125  54.45000  52.55420  52.00000
[43]  55.44170  75.25000  71.28330  73.26040  54.45000  82.43335  67.25000
[50]  79.20000  83.15830  65.75000  66.82500  51.47920 131.18750 131.18750
[57]  65.75000  65.75000  65.75000
```

Y al igual que pasaba con “Fare”, los valores más altos en las clases altas. Con lo cual adoptamos la misma solución que con Fare, los vamos a dejar así.



Podemos seguir el mismo procedimiento para la variable Age:

```
> ##veamos siguiendo el mismo procedimiento qué resultados tenemos con la edad:
> data_out<-as.data.frame(data_num$Age)
> data_out$outlier<-FALSE
> for (i in 1:ncol(data_out)-1){
+   columna = data_out[,i]
+   if (is.numeric(columna)){
+     media = mean(columna)
+     desviacion = sd(columna)
+     data_out$outlier = ( columna>(media+3*desviacion) | columna<(media-3*desviacion))
+   }
+ }
> # Marcamos los TRUE y FALSE
> table(data_out$outlier)

FALSE  TRUE
  887     4
> #Podríamos tener 4 outliers, pero según los valores que nos devuelve la función
> #boxplot.stats todos no los consideraremos outliers, son razonables
> boxplot.stats(data$Age)$out
 [1] 64.0 80.0 74.0 71.0 65.0 71.0 65.0 70.0 64.0 70.0 66.0 65.0 71.0 71.0 74.0 74.0
[17] 71.0 70.5
>
```

Donde vemos que podríamos tener 4 valores outliers, pero observando los valores devueltos por boxplot.stats que son totalmente normales no los vamos a considerar outliers.

Por tanto no hemos detectado outliers que haya que tratar.

4. Análisis de los datos.

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

Del dataset completo nos interesa poder diferentes análisis en función de diferentes subconjuntos de datos, como puede ser el género, la clase en la que viajan los pasajeros, el puerto en el que han embarcado, incluso se pueden definir grupos por edad, y ver realmente si es cierto y se cumplió aquello que dicen en las películas "las mujeres y los niños primero" y poder comprobar si realmente los niños tienen mejor índice de supervivencia que los adultos.

Podemos definir diferentes agrupaciones que podemos llegar más adelante para estudiar los casos por grupos. Algunos ejemplos pueden ser:

```
#Definimos posibles grupos para analizar
#Dentro del dataset vamos a añadir una variable Child
edad_corte = 8
data$Child[data$Age <= edad_corte] <- 1
data$Child[data$Age > edad_corte] <- 0
data$Child <- as.factor(data$Child)

#Agrupando por el género
Mujeres <- data[which(data$Sex=='female'),]
Hombres <- data[which(data$Sex=='male'),]

#Agrupando por Embarked
EmbarqueC <- data[which(data$Embarked=='C'),]
EmbarqueQ <- data[which(data$Embarked=='Q'),]
EmbarqueS <- data[which(data$Embarked=='S'),]

#Por clase
FirstClass <- data[which(data$Pclass==1),]
SecondClass <- data[which(data$Pclass==2),]
ThirdClass <- data[which(data$Pclass==3),]

#Definir grupos por edades. Añadir una columna al dataset (AgeInterval)
intervalos_edad <- c(0,13,18,40,55)
data$AgeInterval <- findInterval(data$Age,intervalos_edad)
#data$AgeInterval <- as.ordered(data$AgeInterval)
data$AgeInterval <- as.factor(data$AgeInterval)
table(data$AgeInterval)
```

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

Vamos a verificar si las variables cuantitativas continuas siguen una distribución normal. Algunos test estadísticos requieren que las variables que van a ser analizadas sigan una distribución normal, por tanto tenemos que conocer cuáles son las distribuciones de nuestras variables continuas.

Realmente las únicas variables cuantitativas continuas que tenemos en el dataset original son las variables "**Age**" y "**Fare**". De todas formas, como nosotros hemos generado una nueva variable a partir de "**Fare**", que le hemos llamado "**Price**", y que por tanto también es una variable cualitativa continua.

En general la prueba de Shapiro-Wilk es considerada una prueba muy potente para contrastar la normalidad de distribuciones. Se asume como hipótesis nula que la población sigue una distribución normal. Si el p-valor obtenido es inferior al nivel de significancia (normalmente α = 0,05) entonces se rechaza la hipótesis nula (y por tanto se concluye que los datos no vienen de una distribución normal). En cambio, si el p-valor es superior al nivel de significancia, entonces no se puede rechazar la hipótesis nula y se asume que los datos siguen una distribución normal.

De todas formas, para poder tener más seguridad, vamos a aplicar otros 2 métodos, la prueba de Anderson-Darling y la prueba de Kolmogorov-Smirnov (conocida también como K-S)

```
tabla.normalidad <- data.frame('Variable'=character(),
                               'Test de Normalidad'=character(),
                               'Valor Estadístico'=numeric(),
                               'p-Value'=numeric(),
                               stringsAsFactors=FALSE)
str(tabla.normalidad)

#Vamos a comprobar la normalidad de las variables continuas
for (i in 1:length(var.continuas)) {
  variable=var.continuas[i]
  #Test Shapiro-Wilk
  test = shapiro.test(data[,variable])
  tabla.normalidad[nrow(tabla.normalidad)+1,] = c(variable, test$method,
                                                  test$statistic, test$p.value)

  #Test Anderson-Darling
  test = ad.test(data[,variable])
  tabla.normalidad[nrow(tabla.normalidad)+1,] = c(variable, test$method,
                                                  test$statistic, test$p.value)

  #Test Kolmogorov-Smirnov'
  test = ks.test(data[,variable], "pnorm", mean=mean(data[,variable]), sd=sd(data[,variable]))
  tabla.normalidad[nrow(tabla.normalidad)+1,] = c(variable, test$method,
                                                  test$statistic, test$p.value)

}
#Tabla de los test de normalidad (Shapiro-Wilk, Anderson-Darling y Kolmogorov-Smirnov)
kable(tabla.normalidad)
```

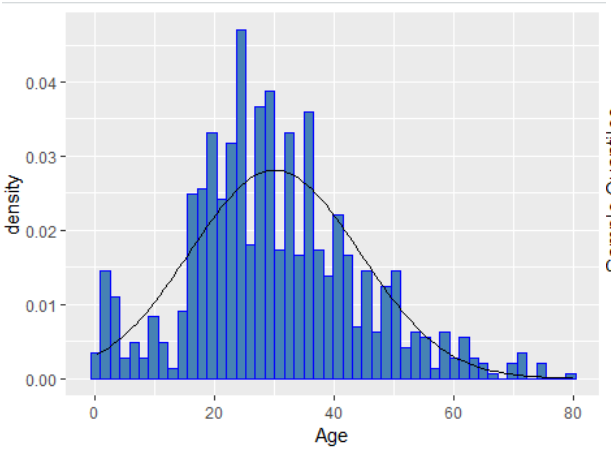
Y los resultados son los que siguen a continuación:

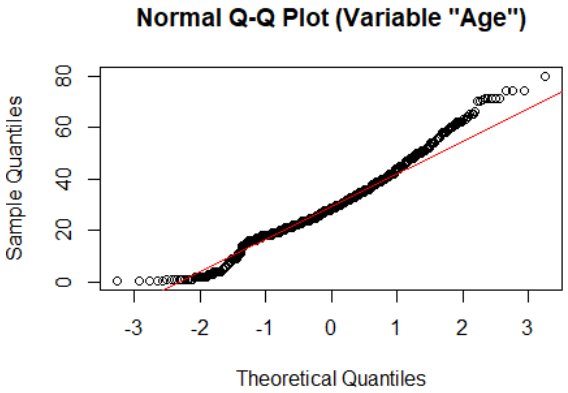
```
> kable(tabla.normalidad)
```

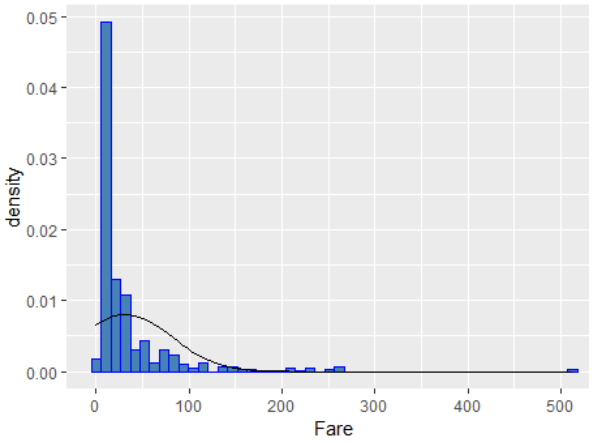
Variable	Test.de.Normalidad	Valor.Estadístico	p.Value
Age	Shapiro-Wilk normality test	0.980142051119058	1.17959231458724e-09
Age	Anderson-Darling normality test	4.77336848791265	7.96504458600651e-12
Age	One-sample Kolmogorov-Smirnov test	0.0599784647259898	0.00328808961389948
Fare	Shapiro-Wilk normality test	0.521891302117355	1.08404452322613e-43
Fare	Anderson-Darling normality test	122.169627214592	3.7e-24
Fare	One-sample Kolmogorov-Smirnov test	0.281848040985975	0
Price	Shapiro-Wilk normality test	0.566484900244852	3.00259150054125e-42
Price	Anderson-Darling normality test	110.325199878766	3.7e-24
Price	One-sample Kolmogorov-Smirnov test	0.26829589819932	0

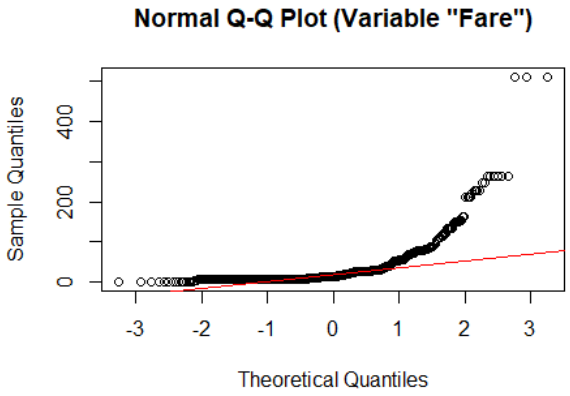
Con estos resultados se pude decir que ninguna de las 3 variables sigue una distribución normal, en todos los casos el p-value ha sido inferior a 0.05 y por tanto se han rechazado la hipótesis nula (que la variable sigue una distribución normal).

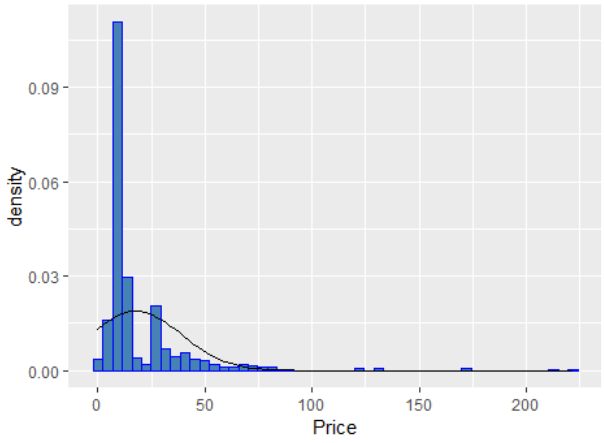
Aún así nos parece interesante poder ver gráficamente la distribución de cada una de las variables, a través de su histograma, curva de densidad y gráficas Q-Q.

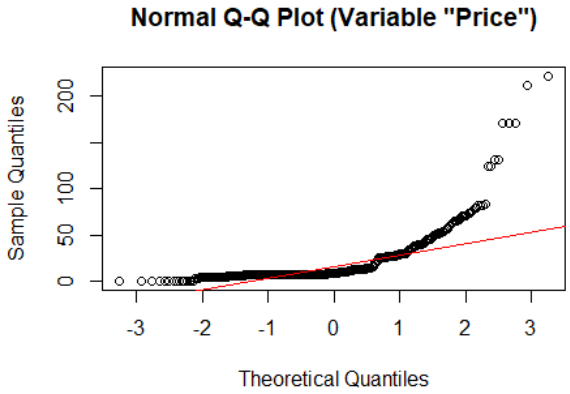












Para la comparación de la varianza, es decir para comprobar si las varianzas entre los grupos a comparar son iguales (homocedasticidad), se pueden usar por ejemplo el test de Levene, cuando los datos siguen una distribución normal, o por ejemplo el test de Fligner-Killeen, que es la

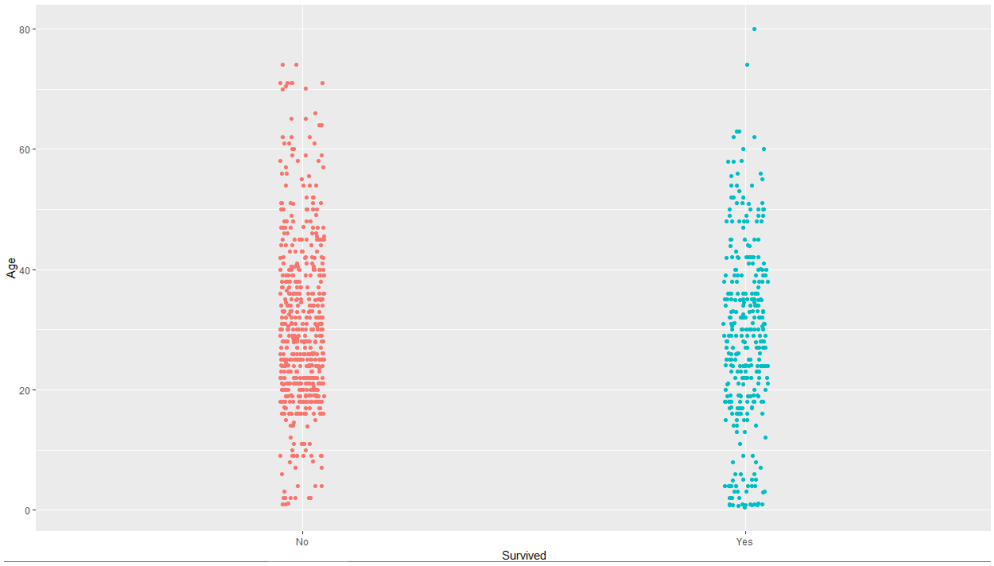
alternativa no paramétrica que se utiliza cuando los datos no siguen una distribución normal (o cuando hay problemas con outliers no resueltos).
En ambos casos, la hipótesis nula asume la igualdad de varianzas en los diferentes grupos de datos, con lo que si el p-valor obtenido es inferior al nivel de significancia (generalmente α = 0,05) entonces se rechaza la hipótesis nula y se concluye que hay heterocedasticidad.

Las variables, que hemos comprobado anteriormente, no siguen una distribución normal, pero si que es cierto que la variable "**Age**" está próxima a una distribución normal, con lo cual podemos utilizar el test de Levene para la comprobación de varianzas.

Vamos a comprobar si las varianzas son iguales cuando comprobamos "**Age**" y el grupo es "**Survived**", es decir estamos comprobando la homogeneidad de varianzas de la edad en los grupos de supervivientes y no supervivientes.

```
> #Comprobar las varianzas edad entre los grupos sobreviven/mueren
> leveneTest(data = data, Age ~ Survived, center = mean)
Levene's Test for Homogeneity of Variance (center = mean)
       Df F value Pr(>F)
group   1  0.8516 0.3563
      889
```

En este caso el p-value es superior a 0.05 y por tanto asumimos que hay homogeneidad de varianzas entre los grupos.
Además, para este caso, vamos a ver la distribución de las muestras de una forma gráfica:



De todas formas, si tenemos la sensación de haber sido muy atrevidos por haber utilizado el test de Levene, por el hecho de que la distribución que sigue la variable "**Age**" no es normal (tal y como nos han dicho los test de normalidad aplicados), podemos aplicar el test no paramétrico Fligner-Killeen, y despejamos dudas:

```
> fligner.test(Age ~ Survived, data=data)

        Fligner-Killeen test of homogeneity of variances

data:  Age by Survived
Fligner-Killeen:med chi-squared = 1.0371, df = 1, p-value = 0.3085
```

Y en este caso el resultado de este test vuelve a confirmar que las varianzas de ambas muestras son homogéneas (porque el p-value es superior a 0.05, y por tanto no se rechaza la hipótesis nula).

Podemos también comprobar cómo se comportan las varianzas cuando se trata de la variable edad "**Age**", pero los grupos son "**Pclass**" y "**Embarked**". Y los resultados son los que siguen a continuación:

(destacar que estas 2 formas de ejecutar la llamada, son equivalentes)

```
#Estudio homogeneidad de varianzas entre Age y como grupo Embarked
leveneTest(data = data, Age ~ Embarked, center = mean)
leveneTest(y=data$Age, group = data$Embarked, center=mean)
```

Y como resultado obtenemos el siguiente:

```
> leveneTest(data = data, Age ~ Embarked, center = mean)
Levene's Test for Homogeneity of Variance (center = mean)
       Df F value Pr(>F)
group   2  1.5535 0.2121
      888
```

Por tanto, vemos que nuevamente podemos afirmar que hay homogeneidad de varianzas de Age en los grupos que define la variable Embarked.

En cambio, cuando controlamos la homogeneidad de varianzas de "**Age**" con otro grupo, y en este caso usamos "**Pclass**", el resultado es diferente:

(aplicando el test paramétrico de Levene)

```
> leveneTest(data = data, Age ~ Pclass, center = mean)
Levene's Test for Homogeneity of Variance (center = mean)
       Df F value   Pr(>F)
group   2  6.2316 0.002053 **
      888
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

(aplicando el test no paramétrico de Fligner-Killeen)

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

        Fligner-Killeen test of homogeneity of variances

data:  Age by Pclass
Fligner-Killeen:med chi-squared = 13.372, df = 2, p-value = 0.001248
```

Tanto con un test paramétrico, como con el no-paramétrico, se rechaza la hipótesis nula, es decir hay heterogeneidad de varianzas entre las muestras de "**Age**" cuando se agrupan por "**Pclass**".

Y si el test lo hacemos entre "**Age**" agrupado por género "**Sex**", los resultados son los siguientes:

```
> #Estudio de la homogeneidad de varianzas de Age en función de Sex
> leveneTest(y=data$Age, group=data$Sex, center=mean)
Levene's Test for Homogeneity of Variance (center = mean)
       Df F value Pr(>F)
group   1  0.1178 0.7316
      889
> fligner.test(Age ~ Sex, data=data)

        Fligner-Killeen test of homogeneity of variances

data:  Age by Sex
Fligner-Killeen:med chi-squared = 0.049434, df = 1, p-value = 0.8241
```

Tanto con el test de Levene, como con el test Fligner-Killeen, en p-value obtenido es claramente superior a 0.05 y entonces podemos afirmar que sí hay homogeneidad de varianzas para "**Age**" cuando está agrupado por "**Sex**".

Cambiamos de variable continua, y pasamos a la comprobación de "**Fare**" (y también de "**Price**"). En la comprobación de homogeneidad varianzas en relación con los diferentes grupos, los resultados indican que hay heterogeneidad de varianza de esas variables respecto a los grupos con los que se ha aplicado el test no paramétrico.
(Omitimos los resultados de Price ya que son muy similares a los que se publican con Fare, y en todos los casos los p-value son valores muy inferiores a 0.05)

```
> fligner.test(Fare ~ Sex, data=data)

        Fligner-Killeen test of homogeneity of variances

data:  Fare by Sex
Fligner-Killeen:med chi-squared = 55.949, df = 1, p-value = 7.436e-14

> fligner.test(Fare ~ Survived, data=data)

        Fligner-Killeen test of homogeneity of variances

data:  Fare by Survived
Fligner-Killeen:med chi-squared = 96.253, df = 1, p-value < 2.2e-16

> fligner.test(Fare ~ Embarked, data=data)

        Fligner-Killeen test of homogeneity of variances

data:  Fare by Embarked
Fligner-Killeen:med chi-squared = 133.23, df = 2, p-value < 2.2e-16

> fligner.test(Fare ~ Pclass, data=data)

        Fligner-Killeen test of homogeneity of variances

data:  Fare by Pclass
Fligner-Killeen:med chi-squared = 365.8, df = 2, p-value < 2.2e-16
```

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

Podemos calcular si hay diferencias significativas entre la media de edad, de mujeres y de hombres. En este caso podemos aplicar el test paramétrico t-test de Student, que requiere que las muestras a comparar sigan una distribución normal. Por un lado vimos que la variable "**Age**" no sigue exactamente una distribución normal, pero aplicando el teorema del límite central con una muestra suficientemente grande (mayor de 30, y en este caso lo es de sobra) se puede asumir que la variable sigue una distribución normal. En caso de que no fuese normal, hubiésemos aplicado una prueba no paramétrica (como el test de Mann-Whitney). En R el comando "t.test" nos permite ejecutar la prueba t de Student, pero también con ese mismo comando realiza el test de Welch cuando las varianzas entre las muestras son diferentes. Por tanto, inicialmente habría que comprobar si las varianzas entre ambas muestras son iguales, para determinar si aplicar uno u otro. Eso ya lo hicimos en el apartado anterior, en el que aplicamos Levene y Fligner-Killeen. También la variable sigue una distribución normal, se puede aplicar el F-test (que no aplicamos en su momento y lo podemos aplicar por ejemplo ahora).

(Haciendo el test de igualdad de varianzas con var.test para el F test)

```
> #Antes comparamos si las varianzas son iguales
> var.test(Mujeres$Age, Hombres$Age)

        F test to compare two variances

data:  Mujeres$Age and Hombres$Age
F = 0.95658, num df = 313, denom df = 576, p-value = 0.6629
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 0.7894069 1.1657769
sample estimates:
ratio of variances
         0.9565807
```

(Se vuelve a comprobar como ya habíamos hecho antes que las varianzas son iguales, es decir con tres test diferentes hemos comprobado que las varianzas de edad en los grupos mujeres y hombres, son iguales)

En el test que vamos a aplicar para comparar las medias de los 2 grupos, la hipótesis nula dice que no hay diferencias significativas entre la media de edades entre hombres y mujeres.
Vamos a utilizar por tanto el t-Test indicando que las varianzas de los grupos son iguales.

```
> #Comparamos la media de edades entre mujeres y hombres
> t.test(Mujeres$Age, Hombres$Age, var.equal = TRUE)

        Two Sample t-test

data:  Mujeres$Age and Hombres$Age
t = -2.8395, df = 889, p-value = 0.004622
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -4.7593836 -0.8690357
sample estimates:
mean of x mean of y
 28.09076  30.90497
```

En este caso rechazamos la hipótesis nula, el p-value es inferior a 0.05, y rechazamos la hipótesis nula. Por tanto, afirmamos que la media de edad por género sí tiene una diferencia significativa.

Ahora entonces vamos a establecer una nueva hipótesis alternativa, y es que la media de edad de las mujeres es menor que la de los hombres. Por tanto, la hipótesis nula es que la edad de las mujeres es mayor o igual a la de los hombres:

```
> t.test(Mujeres$Age, Hombres$Age, alternative = "less", var.equal = TRUE)

        Two Sample t-test

data:  Mujeres$Age and Hombres$Age
t = -2.8395, df = 889, p-value = 0.002311
alternative hypothesis: true difference in means is less than 0
95 percent confidence interval:
     -Inf -1.18229
sample estimates:
mean of x mean of y
 28.09076  30.90497
```

Nuevamente se vuelve a rechazar la hipótesis nula, solo que esta vez la hipótesis alternativa era “less”, es decir que podemos afirmar que la media de edad de las mujeres del Titanic es inferior a la media de edad de los hombres a bordo.

Podemos comprobar la relación entre la supervivencia y el género. Es decir, vamos a comprobar si el género importa a la hora de la supervivencia, es decir si ser mujer o ser hombre es significativo a la hora de valorar la supervivencia. Para ello vamos a aplicar el test exacto de Fisher que analiza tablas de contingencia.

En este caso la hipótesis nula es que la proporción de mujeres que mueren coincide con la proporción de hombres que mueren en el accidente del Titanic.

```
> #Proporcion de mujeres que mueren (no sobreviven) coindice con la proporcion
> #de los hombres que viven
> fisher.test(table(data$Sex, data$Survived)) #p-Value < 0.05 se rechaza hipotesis nula

        Fisher's Exact Test for Count Data

data:  table(data$Sex, data$Survived)
p-value < 2.2e-16
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
 0.0575310 0.1138011
sample estimates:
odds ratio
0.08128333
```

En este caso rechazamos la hipótesis nula (el p-value es inferior a 0.05), es decir sí que son diferentes las proporciones de supervivencia entre hombres y mujeres.

Una vez que hemos visto que el género sí que importa a la hora de las probabilidades de supervivencia, podemos entonces intentar ver quien tiene más probabilidades de supervivencia, si las mujeres o los hombres. Entonces ahora añadimos una condición a la hipótesis alternativa, le ponemos “less” para indicar (en la hipótesis nula) que el primer grupo (en este caso las mujeres ya que en orden alfabético es female y male) tiene menor proporción (probabilidad) si se rechaza la hipótesis nula. Es decir, la hipótesis nula dice que “la proporción de mujeres que mueren es mayor que la de los hombres” (porque la hipótesis nula precisamente es “less” menor, y ya vimos que la igualdad no se cumplía)

```
> #Proporcion de mujeres que mueren (no sobreviven) es mayor que la de hombre que mueren?
> fisher.test(table(data$Sex, data$Survived), alternative = 'less')

        Fisher's Exact Test for Count Data

data:  table(data$Sex, data$Survived)
p-value < 2.2e-16
alternative hypothesis: true odds ratio is less than 1
95 percent confidence interval:
 0.0000000 0.1081391
sample estimates:
odds ratio
0.08128333
```

En este caso rechazamos la hipótesis nula (el p-value es inferior a 0.05), es decir se cumple la hipótesis alternativa “la proporción de mujeres que mueren es inferior a la de los hombres” (o dicho con otras palabras, la proporción de mujeres que sobreviven es más alta que la de los hombres).

Tengamos en cuenta una cosa, la tabla de contingencia que estamos evaluando es la siguente:

```
> table(data$Sex, data$Survived)

          0   1
  female  81 233
  male   468 109
```

El odd ratio de la tabla de contingencia se calcula así.

```
> tab.contingencia =  table(data$Sex, data$Survived)
> odd_ratio = (tab.contingencia[1,1]/tab.contingencia[1,2]) /
+             (tab.contingencia[2,1]/tab.contingencia[2,2])
> odd_ratio
[1] 0.08096732
```

Como veíamos antes en el test de Fisher, hipótesis alternativa se hace verdadera cuando el odd_ratio se vuelve 1 (es decir cuando las proporciones son iguales). Luego cuando a la hipótesis alternativa le ponemos “less”, esta se hace verdadera cuando odd_ratio es menor que 1 (como fue el caso que comprobamos). Y si ponemos “greater” la hipótesis alternativa se hace verdadera si el odd_ratio es mayor que 1.

Podemos obtener intervalos de confianza, por ejemplo podemos obtener un intervalo de confianza al 99% de que la media de las edades de

diferentes muestras, van a estar con una confianza del 99% entre los valores 28.8 y 31.26 de media de edad.

```
> #Intervalo de confianza, al 99%
> test = t.test(data$Age, conf.level = 0.99)
> test$conf.int
[1] 28.80817 31.26348
attr(,"conf.level")
[1] 0.99
```

Podemos querer saber por ejemplo la relación que existe (si es que existe entre la clase "**Pclass**" y la supervivencia "**Survived**". Al ser 2 variables categóricas vamos a aplicar por ejemplo el test Chi-Cuadrado. Aquí la hipótesis nula nos dice que las variables son independientes. Vamos a comprobarlo:

```
> #Comparamos la relación que hay entre Pclass y Survived
> #Hipótesis nula H0 --> Las variables son independientes
> test_chisq <- chisq.test(data$Pclass, data$Survived)
> test_chisq

        Pearson's Chi-squared test

data:  data$Pclass and data$Survived
X-squared = 102.89, df = 2, p-value < 2.2e-16

> test_chisq$p.value
[1] 4.549252e-23
```

Vemos que el p-value obtenido es inferior a 0.05, es decir rechazamos la hipótesis nula, y por tanto afirmamos que las variables no son independientes.
De los datos que nos devuelve, aparte del p-value, obtenemos df que son los grados de libertad y que lo podemos obtener a través de la tabla de contingencias, y el estadístico de contraste. Si es el estadístico de contraste obtenido supera el valor crítico (calculado abajo) entonces se rechaza la hipótesis nula.

```
#Los grados de libertad los podemos obtener de la tabla de contingencias:
#table(data$Pclass, data$Survived) donde hay que multiplicar (filas-1)*(columnas-1)
#Que es lo mismo que obtener los diferentes valores de cada variable (menos 1)
#y multiplicarlos

> df_chisq <- (length(levels(data$Pclass))-1) * (length(levels(data$Survived))-1)
> valorcritico_chisq <- qchisq(p=0.05, df=df_chisq,lower.tail = FALSE)
> test_chisq$statistic
X-squared
  102.889
> #Si el estadistico chi-cuadrado calculado es superior al valor critico, entonces se rechaza
> #la hipótesis nula, y por tanto concluimos que las variables sí son dependientes
> cat(sprintf("El estadistico obtenido es: %f, y el valor crítico es: %f",
+             test_chisq$statistic, valorcritico_chisq))
El estadistico obtenido es: 102.888989, y el valor crítico es: 5.991465
```

De la misma forma, aplicando el test Chi-Cuadrado podemos saber si existe o no relación entre las variables "**Embarked**" y "**Survived**"

```
> ########################################################################
> #Test de independencia entre Embarked y Survived
> #Hipótesis nula: Embarked y Survived son independientes
> test_chisq <- chisq.test(data$Embarked, data$Survived)
> test_chisq

        Pearson's Chi-squared test

data:  data$Embarked and data$Survived
X-squared = 25.964, df = 2, p-value = 2.301e-06

> test_chisq$p.value
[1] 2.300863e-06
> df_chisq <- (length(levels(data$Pclass))-1) * (length(levels(data$Survived))-1)
> valorcritico_chisq <- qchisq(p=0.05, df=df_chisq,lower.tail = FALSE)
> test_chisq$statistic
X-squared
 25.96445
> cat(sprintf("El estadístico obtenido es: %f, y el valor crítico es: %f",
+             test_chisq$statistic, valorcritico_chisq))
El estadístico obtenido es: 25.964453, y el valor crítico es: 5.991465
```

Como podemos comprobar se rechaza también la hipótesis nula, por tanto las variables sí que son dependientes.

Con la variable nueva que habíamos creado “Child” (que la definimos como los niños de edad menor o igual a 8 años), vamos a comprobar si tienen las mismas probabilidades de supervivencia que el resto de pasajeros. Para ello aplicamos el test de Fisher:

```
> ############################################################
> #Proporcion de niños que mueren (no sobreviven) coindice con la proporcion
> #de los que no son niños (hemos hecho un corte a los 8 años)
> fisher.test(table(data$Child, data$Survived)) #p-value < 0.05 se rechaza hipotesis nula

        Fisher's Exact Test for Count Data

data:  table(data$Child, data$Survived)
p-value = 8.901e-06
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
 1.918487 6.514538
sample estimates:
odds ratio
  3.481729
```

De entrada, vemos que se rechaza la hipótesis nula, es decir que las proporciones no coinciden, vamos a ver entonces si tienen o no más probabilidades de supervivencia.

Tal y como es nuestra tabla de contingencias, nuestra hipótesis nula nueva será “la probabilidad de muerte de los mayores de 8 años (no-niños) es menor que la probabilidad de muerte de los niños (hasta 8 años)”. Entonces ejecutamos lo siguiente:

```
> #Proporcion de no-niños que mueren (no sobreviven) es menor a
> #la proporcion de niños (hasta 8 años) que mueren?
> fisher.test(table(data$Child, data$Survived), alternative="greater")

        Fisher's Exact Test for Count Data

data:  table(data$Child, data$Survived)
p-value = 7.482e-06
alternative hypothesis: true odds ratio is greater than 1
95 percent confidence interval:
 2.093434      Inf
sample estimates:
odds ratio
  3.481729
```

Se rechaza la hipótesis nula, y se acepta la alternativa, es decir que la probabilidad de muerte de los no-niños es mayor que la de los niños (o sea que los niños tienen más posibilidades de supervivencia).

```
> tab.contingencia =  table(data$Child, data$Survived)
> tab.contingencia

     0   1
  0 530 304
  1  19  38
> odd_ratio = (tab.contingencia[1,1]/tab.contingencia[1,2]) /
+   (tab.contingencia[2,1]/tab.contingencia[2,2])
> odd_ratio
[1] 3.486842
```

Supongamos que ahora queremos saber si la media de edad de las personas que murieron es coincidente con la media de edad de las personas que han podido sobrevivir. Para ello aplicamos el test t-Test:

```
#Edad media de los muertos equivalente a la media de los supervivientes
#Son equivalentes las 2 siguientes instrucciones
t.test(Age~Survived,data = data)
t.test(data$Age[which(data$Survived==0)], data$Age[which(data$Survived==1)])
```

```
> t.test(data$Age[which(data$Survived==0)], data$Age[which(data$Survived==1)])

        Welch Two Sample t-test

data:  data$Age[which(data$Survived == 0)] and data$Age[which(data$Survived == 1)]
t = 1.9782, df = 698.37, p-value = 0.0483
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.01460646 3.88932221
sample estimates:
mean of x mean of y
 30.78506  28.83310
```

En este caso vemos que el p-valor es inferior por muy poco a 0.05, lo cual nos lleva a rechazar la hipótesis nula (que las medias de edad son coincidentes para fallecidos y supervivientes), pero también es cierto que esta variable “Age” no sigue un distribución normal tal y como ya hemos comentado. Entonces en este caso vamos a aplicar el test no paramétrico U de Mann-Whitney (en este caso para muestras no apareadas).

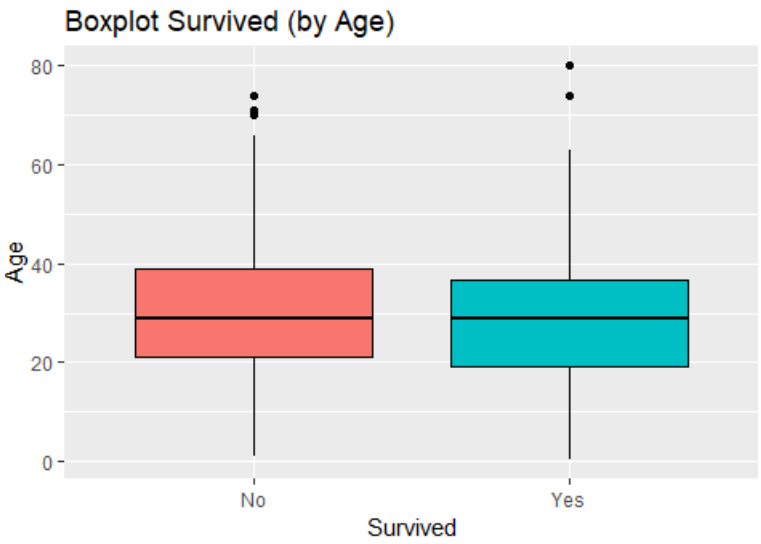
```
#Prueba no paramétrica Mann-Whitney
wilcox.test(x = data$Age[which(data$Survived==0)],
            y = data$Age[which(data$Survived==1)],
            paired=FALSE)
```

```
> #Prueba no paramétrica Mann-Whitney
> wilcox.test(x = data$Age[which(data$Survived==0)],
+             y = data$Age[which(data$Survived==1)],
+             paired=FALSE)

        Wilcoxon rank sum test with continuity correction

data:  data$Age[which(data$Survived == 0)] and data$Age[which(data$Survived == 1)]
W = 98372, p-value = 0.2291
alternative hypothesis: true location shift is not equal to 0
```

En este caso la prueba no-paramétrica nos devuelve un p-value por encima de 0.05 y por tanto no rechazamos la hipótesis nula, es decir que podemos decir que la media de edad es coincidente entre los muertos y los supervivientes del accidente del Titanic.

De hecho gráficamente podemos ver:

```
#Boxplot de supervivencia (por edad)
ggplot(data = data, aes(x=Survived, y=Age))+
  scale_x_discrete(name ="Survived", labels=c("0"="No","1"="Yes"))+
  geom_boxplot(color="black", fill=c(colores_defecto_ggplot[1],colores_defecto_ggplot[2])) +
  ggtitle("Boxplot Survived (by Age)")
```



Si queremos por ejemplo saber si la edad "**Age**" es independiente del puerto de embarque "**Embarked**" entonces podemos realizar un test Anova, donde la hipótesis nula nos dice que la media de la edad es independiente del puerto de embarque, es decir que la media de edad de la gente que subió en "C" (Cherbourg), coincide con la media de los que embarcaron en "Q" (Queenstown) y también con los que embarcaron en "S" (Southampton). Y por el contrario la hipótesis nula es que alguna de las medias es diferente.

```
> modelo_anova <- aov(formula = Age ~ Embarked, data = data)
> resumen_anova <- summary(modelo_anova)
> resumen_anova
             Df Sum Sq Mean Sq F value Pr(>F)
Embarked      2    330   164.8   0.817  0.442
Residuals   888 179029   201.6
```

Vamos a explicar un poco estos resultados: De entrada podemos decir que vemos 2 filas, una que pone "Embarked" y otra que pone "Residuals". La primera fila corresponde a todo lo relativo a la varianza explicada (la de la

variable independiente “Embarked”) y la segunda fila relativo a la varianza no explicada o residual.

Df son los grados de libertad, para el caso de la varianza explicada (la de la variable independiente) es k-1. En nuestro caso como “Embarked” tiene 3 valores posibles, entonces Df=2.
Para la parte de la varianza residual, es n - k, por tanto el valor Df=888 (891 son los registros del dataframe, si le restamos los 2 de antes quedan los 888).
Sum Sq es la suma de la diferencia de los cuadrados, conocidos como SCDe (variación entre-grupos) y SCDi (variación intra-grupos). La función nos ha devuelto 330 y 179029 respectivamente.

Mean Sq es la media cuadrática referente a entre-grupos e intra-grupos. O lo que también llamamos como Varianza explicada (línea superior, es decir 164.8) y varianza no explicada (línea inferior, es decir 201.6).
Corresponde al cociente entre la suma de diferencias de cuadrados y los grados de libertad, es decir realmente Mean Sq=Sum Sq/Df, es decir:

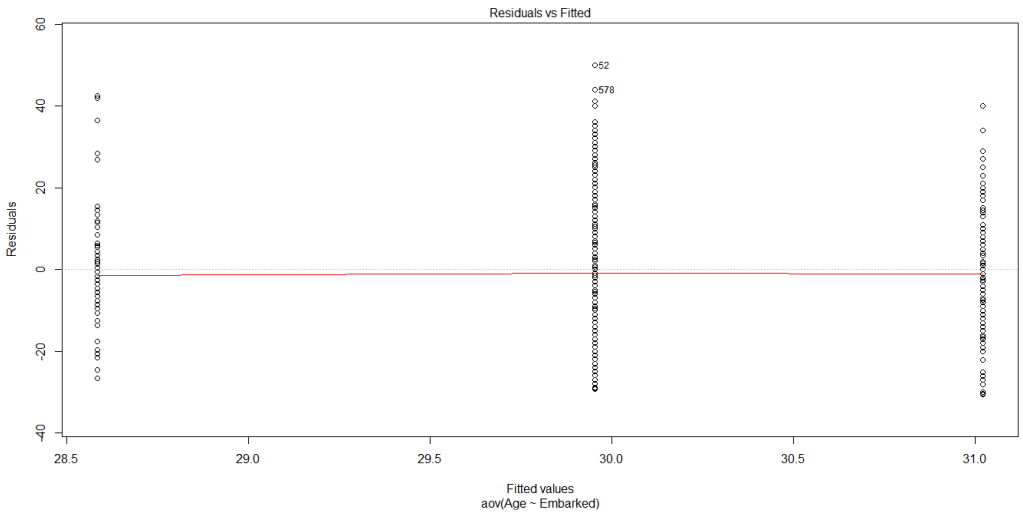
para la media cuadrática entregrupos 330/2=165 (realmente 164.8) y
para la media cuadrática intragrupos 179029/888=201.6

F Es el estadístico (Fisher-Snedecor), que es el cociente entre la varianza explicada entre la varianza no explicada. Por tanto es el cociente entre los valores *Mean Sq*, siendo el numerador la fila superior (164.8) y el denominador la fila inferior (201.6). El valor del cociente es 0.817, es decir el estadístico F.

Pr(>F) Es la probabilidad asociada, el p-valor que nos indicará si se rechaza o no la hipotesis nula. En este caso el valor es: 0.442 que es mayor que 0.05 y nos permite aceptar (no rechazar) la hipótesis nula que decía que las medias de la variable “**Age**” en función del valor de “**Embarked**” eran iguales.

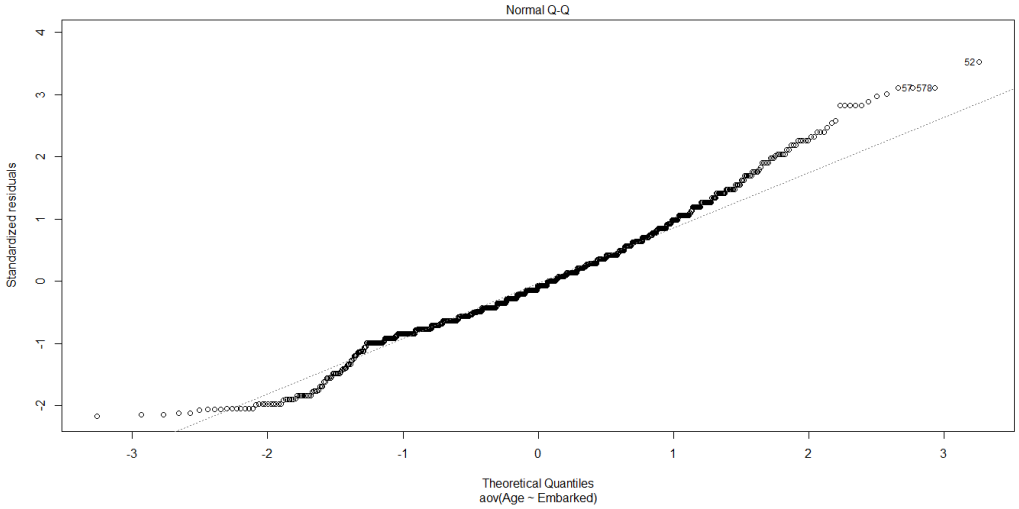
Ahora vamos a comprobar los gráficos que nos devuelve el modelo para ver si se han cumplido las condiciones para aplicar Anova.

Los ponemos a continuación:

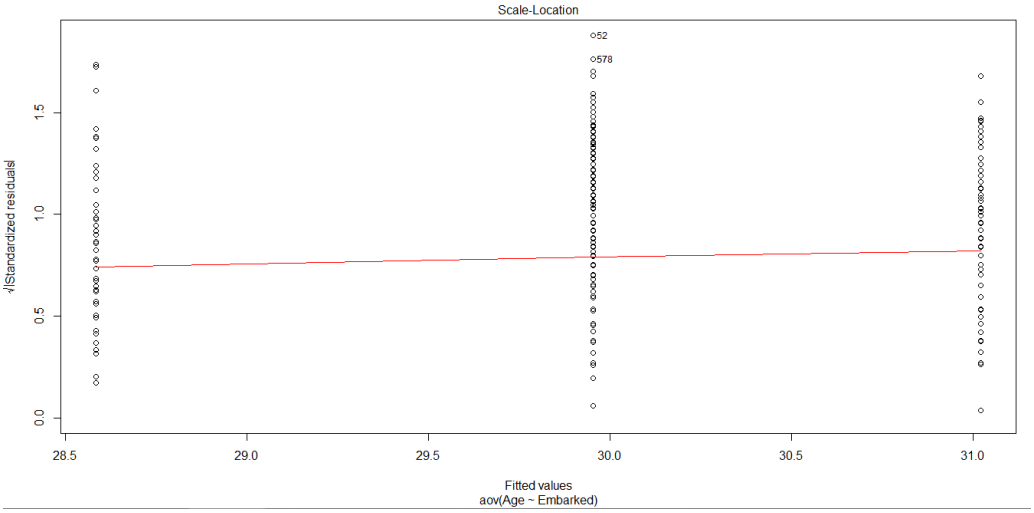


Esta gráfica nos muestra si los residuos tienen patrones no lineales. Puede existir una relación no lineal entre las variables explicativas y la variable explicada, y ese patrón podría verse en esta gráfica si el modelo no captura esa relación no lineal. Si los residuos están igualmente distribuidos alrededor de una línea horizontal sin patrones diferentes, es una buena señal de que no hay relaciones no lineales.

En nuestro caso concreto, se ve una línea casi horizontal (hay una mini curva) y los residuos se distribuyen alrededor de dicha línea. Podemos decir que no hay indicios de relaciones no lineales.



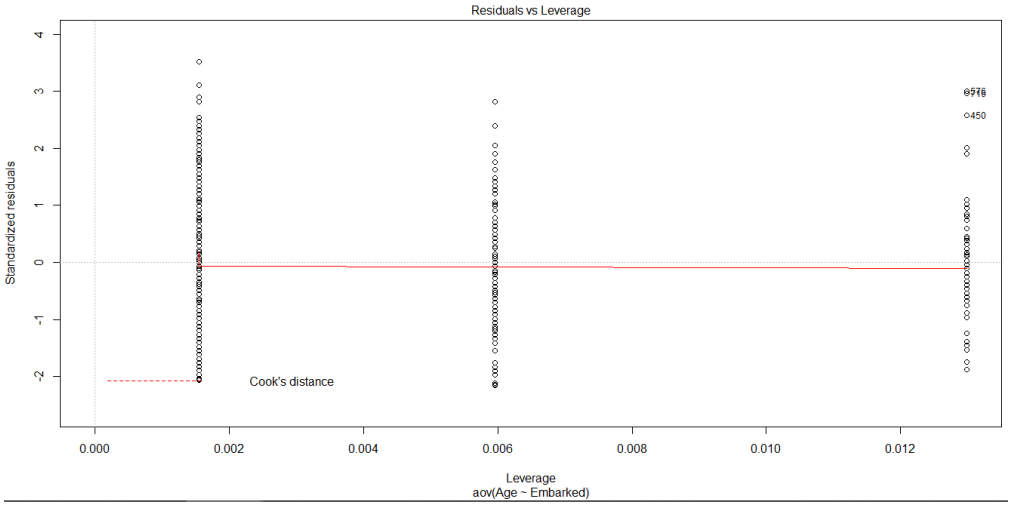
Si no nos ponemos muy estrictos en la exigencia, con el gráfico Normal Q-Q podemos ver que la variable dependiente sigue una distribución aproximadamente normal. La mayoría de los puntos se encuentran alineados en la línea de puntos y solamente es en los extremos donde esa situación no se cumple.



La gráfica Scale-Location también es conocida con el nombre de gráfica Spread-Location. Aquí vemos si los residuos se reparten de forma equitativa. De esa forma es como se verifica la asunsión de igual de

varianza (homocedasticidad). Una buena señal es si se ve una línea horizontal con los residuos distribuidos (al azar) de una forma similar.

En nuestro ejemplo vemos casi una recta horizontal y los puntos se distribuyen de forma muy similar a los lados de la línea y en los extremos también parece un gráfico simétrico (tanto en horizontal como en vertical). Es decir que el gráfico nos está indicando que se cumple la homocedasticidad en nuestro modelo.



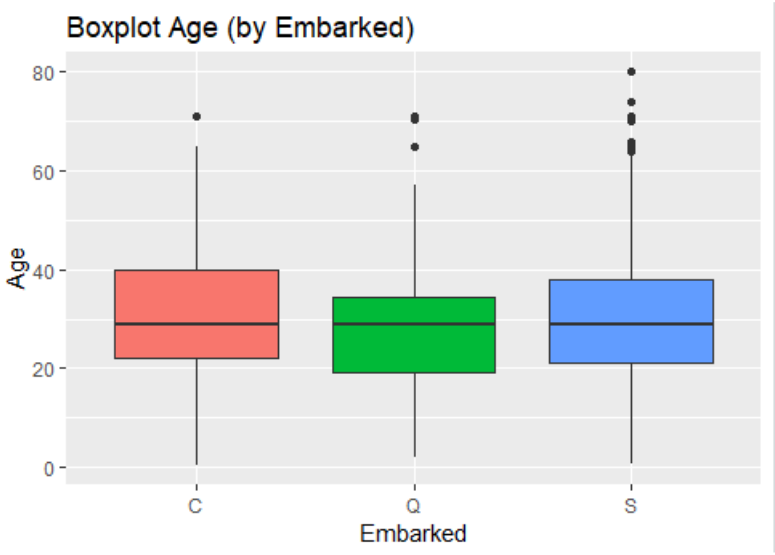
Esta gráfica sirve para ayudarnos a encontrar casos influyentes (es decir, sujetos) si es que los hay. No todos los valores atípicos son influyentes en el análisis de regresión lineal. En el caso de existir valores extremos (outliers), puede que no sean influyentes para determinar la línea de regresión. Eso significa que los resultados no serían muy diferentes si los incluimos o excluimos del análisis, es decir no son influyentes. Por otra parte, algunos casos podrían ser muy influyentes, incluso aunque parezca que están dentro de un rango razonable de los valores. Pueden ser casos extremos contra una línea de regresión y pueden alterar los resultados si los excluimos del análisis. Es decir, estos no están alineados (tendencia) con la mayoría de los casos.

Tenemos que buscar casos que estén fuera de una línea discontinua (que es la distancia de Cook). Cuando encontremos casos fuera de la distancia de Cook (es decir que tienen puntuación alta de distancia de Cook) se consideran casos influyentes en los resultados de la regresión. Y esa regresión se verá afectada si excluimos esos casos.

Después de este análisis casi y dar por aceptadas las condiciones del cumplimiento de Anova, podemos afirmar que las medias de edad en los diferentes puertos de Embarke es coincidente.

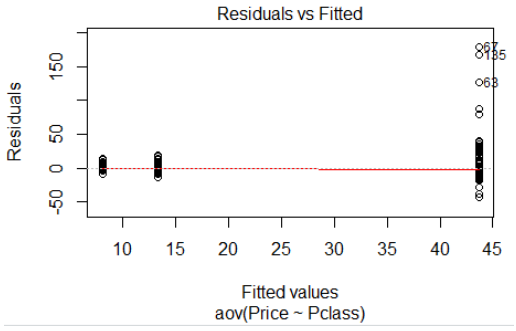
Y de hecho, si vemos el Boxplot de la variable “Age” en función de la variable “Embarked”:

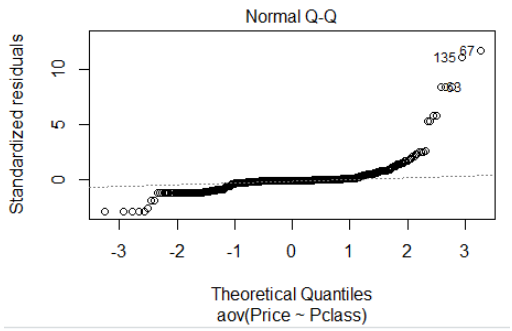
```
#Boxplot de Age (por Embarked)
ggplot(data, aes(x=Embarked, y=Age)) +
  geom_boxplot(fill=(hue_pal()(3))) +
  ggtitle("Boxplot Age (by Embarked)")
```

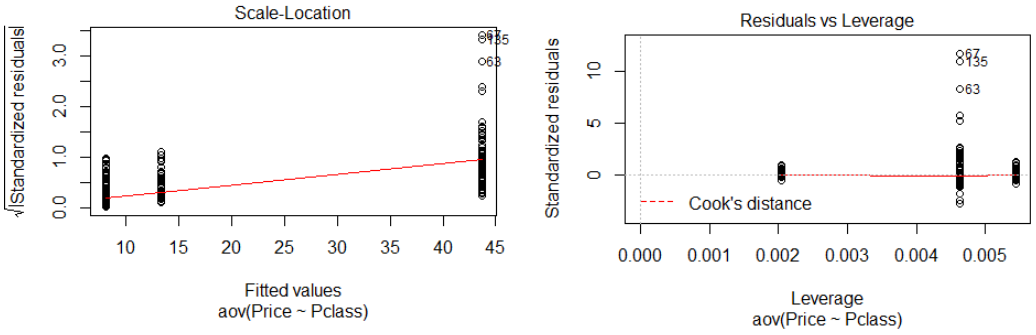


Si procuramos hacer algo similar, un test Anova pero con la variables "**Price**" y "**Pclass**", intentado comprobar con la hipótesis nula que el precio medio pagado es igual para cada una de las 3 clases.

```
> #Modelo anova de Price en función de Pclass (el p-value es inferior a 0.05)
> #Además no se cumplen las condiciones
> modelo_anova <- aov(formula = Price ~ Pclass, data = data)
> resumen_anova <- summary(modelo_anova)
> resumen_anova
             Df Sum Sq Mean Sq F value Pr(>F)
Pclass        2 194362   97181   418.3 <2e-16 ***
Residuals   888 206326     232
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> plot(modelo_anova)
Hit <Return> to see next plot:
```





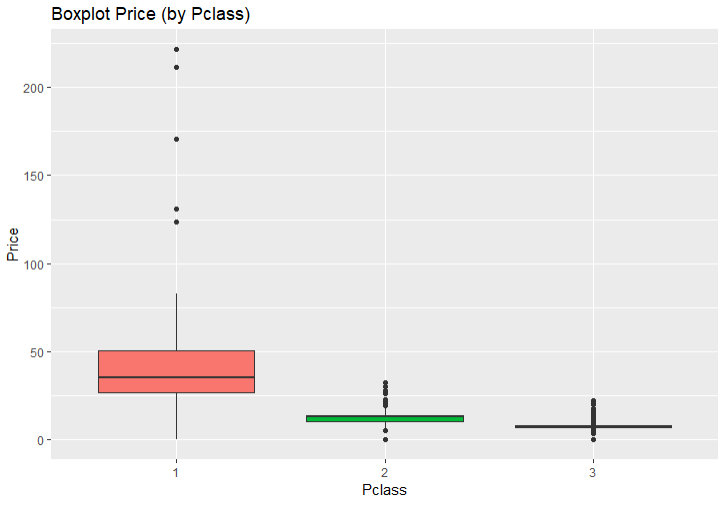


Además de que el p-value devuelto es inferior a 0.05, lo más importante es que no se cumplen las condiciones para aplicar Anova, y por tanto este test no tiene valor. Vamos a intentar entonces aplicar un test no paramétrico. En este caso vamos a utilizar el test de Kruskal Wallis.

```
> #Test no paramétrico, Kruskal Wallis
> modelo_krustal <- kruskal.test(formula=Price ~ Pclass, data=data)
> modelo_krustal

        Kruskal-Wallis rank sum test

data:  Price by Pclass
Kruskal-Wallis chi-squared = 566.65, df = 2, p-value < 2.2e-16
```

El resultado de este test también arroja un p-value inferior a 0.05 y por tanto rechazamos la hipótesis nula. Es decir que la media del precio pagado por billete no es independiente de la clase de turista.
Lo podemos comprobar también con el boxplot relativo a ello:

```
#Boxplot de Price (por Pclass)
ggplot(data, aes(x=Pclass, y=Price)) +
  geom_boxplot(fill=(hue_pal()(3))) +
  ggtitle("Boxplot Price (by Pclass)")
```



Modelo de regresión logística

Vamos a crear un modelo de regresión logística, que es un tipo de análisis de regresión que se utiliza para predecir el resultado de una variable categórica, en función de las variables independientes o predictoras. En nuestro caso la variable objetivo "**Survived**" es una variable categórica binaria, es decir que tomar valor de 0 o 1 (verdadero o falso). En nuestro caso concreto, la variable toma el valor 1 cuando el pasajero ha sobrevivido al accidente, y 0 en caso contrario.

De entrada, vamos a utilizar como variables explicativas las variables que traía por defecto el dataset original (menos las que hemos eliminado en el proceso de transformación y limpieza de datos que hemos comentado anteriormente).

Antes de nada, vamos a crear para los modelos un conjunto de entrenamiento y un conjunto de test. Es decir de nuestro dataframe original, vamos a realizar una partición, un 75% de datos para el entrenamiento, y el 25% restante para validar los modelos.

```
> #Creamos el conjunto de entramiento y de test (75-25)
> set.seed(123)
> index_train <- createDataPartition(data$Survived, p = .75, list = FALSE)
> data_train <- data[index_train,]
> data_test <- data[-index_train,]
```

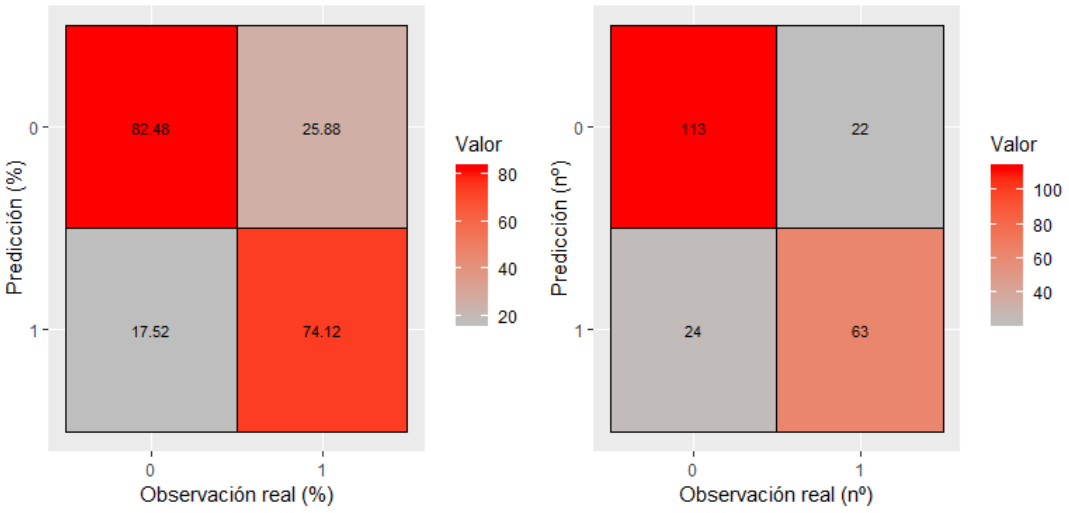
Como decíamos, vamos a crear un primer modelo (glm1) con los variables "originales". En este caso esas variables son: "**Pclass**", "**SibSp**", "**Parch**", "**Sex**", "**Age**", "**Fare**" y "**Embarked**".

```
> modelo_glm1 <- glm(formula = Survived ~ Pclass+SibSp+Parch+Sex+Age+Fare+Embarked,
+                    data = data_train, family = binomial(link = "logit"))
> summary(modelo_glm1)

Call:
glm(formula = Survived ~ Pclass + SibSp + Parch + Sex + Age +
    Fare + Embarked, family = binomial(link = "logit"), data = data_train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.5799  -0.6524  -0.3682   0.6153   2.8782

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  4.058334   0.540930   7.503 6.26e-14 ***
Pclass2     -0.844282   0.341197  -2.474   0.0133 *
Pclass3     -2.340607   0.351271  -6.663 2.68e-11 ***
SibSp       -0.313868   0.122887  -2.554   0.0106 *
Parch       -0.135616   0.139358  -0.973   0.3305
Sexmale     -2.647942   0.233145 -11.357  < 2e-16 ***
Age         -0.037148   0.008419  -4.413 1.02e-05 ***
Fare         0.002366   0.002748   0.861   0.3892
EmbarkedQ    0.127208   0.438947   0.290   0.7720
EmbarkedS   -0.466026   0.281394  -1.656   0.0977 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 891.19  on 668  degrees of freedom
Residual deviance: 579.81  on 659  degrees of freedom
AIC: 599.81

Number of Fisher Scoring iterations: 5
```

Se puede observar en este modelo que tanto las variables "**Fare**" como "**Parch**" no son significativas, es decir no están aportando a la hora de predecir la variable "**Survived**".
Además, podemos comprobar el "sentido" de la significación. Por ejemplo, vemos como significativa la variable dummy (con las variables categóricas se crean automáticamente tantas variables dummy como niveles – 1. En el caso de la variable Sex, al tener 2 valores ha creado la variable "Sexmale", es decir la parte correspondiente a hombres, mientras que la parte de mujeres forma parte del Intercept del modelo). Concretamente vemos que "Sexmale" es una variable significativa, pero con valor negativo (de las más negativas junto con "Pclass3"). Eso significa que esas variables son una influencia "negativa" de cara a la supervivencia. Es decir, esas variables contribuyen negativamente a la supervivencia, se puede ver por ejemplo que Pclass3 afecta más negativamente que Pclass2.

```
> csstab_glm1 <- CrossTable(data_test$Survived, predict_glm1,
+                           prop.chisq  = FALSE, prop.c = FALSE, prop.r =FALSE,
+                           dnn = c('Reality', 'Prediction'))

   Cell Contents
|-------------------------|
|                       N |
|         N / Table Total |
|-------------------------|

Total Observations in Table:  222

             | Prediction
     Reality |         0 |         1 | Row Total |
-------------|-----------|-----------|-----------|
           0 |       113 |        24 |       137 |
             |     0.509 |     0.108 |           |
-------------|-----------|-----------|-----------|
           1 |        22 |        63 |        85 |
             |     0.099 |     0.284 |           |
-------------|-----------|-----------|-----------|
Column Total |       135 |        87 |       222 |
-------------|-----------|-----------|-----------|

> pct.correcto_glm1 <- 100 * sum(diag(mat.confusion_glm1)) / sum(mat.confusion_glm1)
> print(sprintf("El %% de registros correctamente clasificados es: %.4f %%",
+               pct.correcto_glm1))
[1] "El % de registros correctamente clasificados es: 79.2793 %"
```

Gráficamente la matriz de confusión tanto en porcentaje como en muestras:

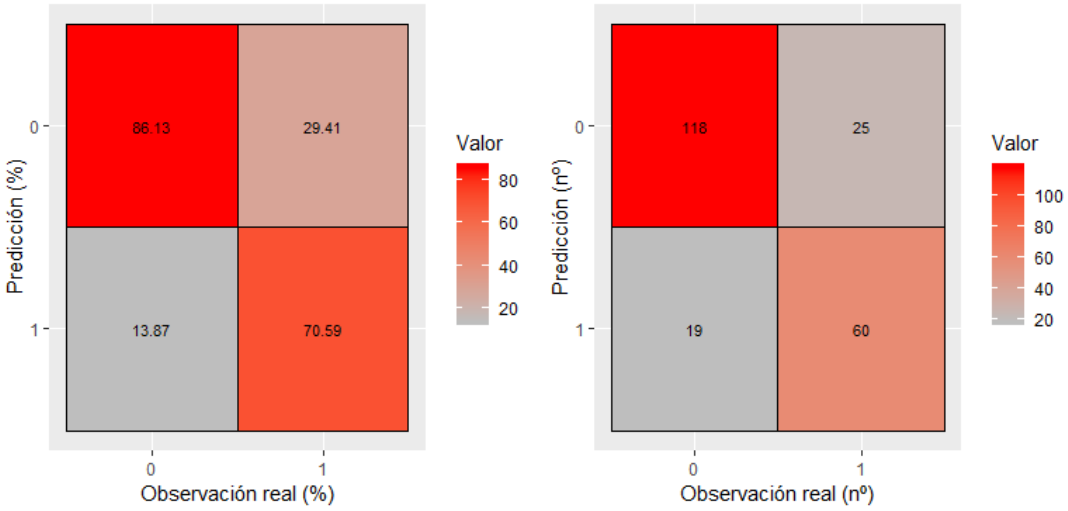


Vamos a construir otro nuevo modelo (glm2), similar pero esta vez partiendo de los resultados del modelo anterior, le vamos a quitar aquellas variables que vimos que no eran significativas para el modelo de predicción. Por tanto tendremos un modelo2 cuyas variables a utilizar serán "**Pclass**", "**SibSp**", "**Sex**" y "**Age**".

```
> modelo_glm2 <- glm(formula = Survived ~ Pclass+SibSp+Sex+Age,
+                    data = data_train, family = binomial(link = "logit"))
> summary(modelo_glm2)

Call:
glm(formula = Survived ~ Pclass + SibSp + Sex + Age, family = binomial(link = "logit"),
    data = data_train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.6458  -0.6431  -0.3841   0.6048   2.4529

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.893156   0.441492   8.818  < 2e-16 ***
Pclass2     -1.080948   0.299354  -3.611 0.000305 ***
Pclass3     -2.466696   0.287568  -8.578  < 2e-16 ***
SibSp       -0.351952   0.118704  -2.965 0.003027 **
Sexmale     -2.712596   0.225025 -12.055  < 2e-16 ***
Age         -0.035839   0.008085  -4.433 9.31e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 891.19  on 668  degrees of freedom
Residual deviance: 587.63  on 663  degrees of freedom
AIC: 599.63

Number of Fisher Scoring iterations: 5
```

Ahora podemos comprobar que todas las variables utilizadas son variables significativas para el modelo. Además, el modelo parece haber mejorado un poco en cuanto a la predicción:

```
> pct.correcto_glm2 <- 100 * sum(diag(mat.confusion_glm2)) / sum(mat.confusion_glm2)
> print(sprintf("El %% de registros correctamente clasificados es: %.4f %%",
+               pct.correcto_glm2))
[1] "El % de registros correctamente clasificados es: 80.1802 %"
>
```



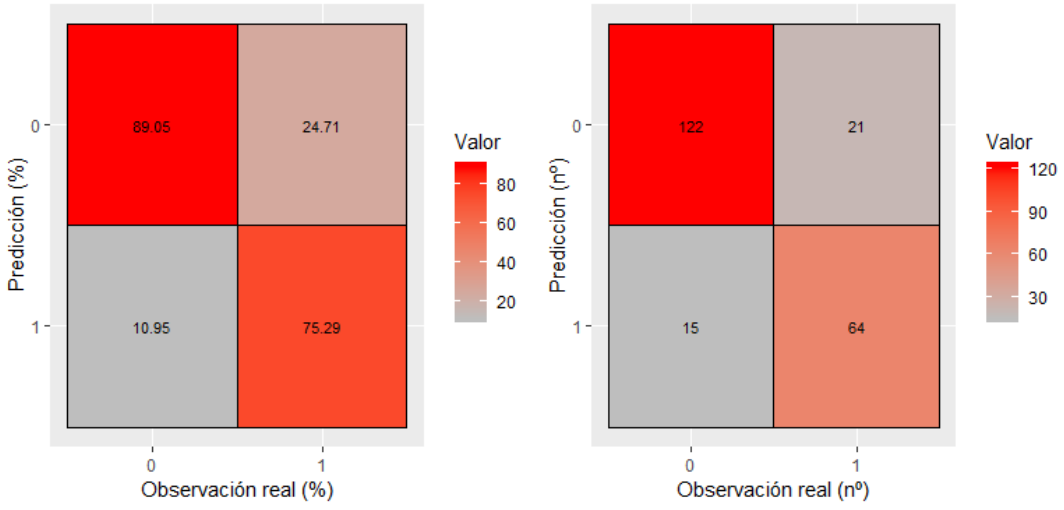
Hasta ahora hemos utilizado variables del dataset "original". Vamos a construir otro modelo (glm3), pero esta vez vamos a intentar utilizar alguna de las variables que hemos construido para ayudar a predecir la supervivencia. Para este caso vamos a utilizar las variables nuevas "**Child**" y también "**FamilySize**". Además de esas 2 variables, vamos a utilizar otras 3 variables originales, que son: "**Pclass**", "**Sex**" y "**Age**".

```
> modelo_glm3 <- glm(formula = Survived ~ Sex+Pclass+Age+Child+FamilySize,
+                    data = data_train, family = binomial(link = "logit"))
> summary(modelo_glm3)

Call:
glm(formula = Survived ~ Sex + Pclass + Age + Child + FamilySize,
    family = binomial(link = "logit"), data = data_train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.8265  -0.6247  -0.3886   0.5687   2.4560

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.873193   0.493812   7.843 4.38e-15 ***
Sexmale     -2.803518   0.232082 -12.080  < 2e-16 ***
Pclass2     -1.078328   0.298521  -3.612 0.000304 ***
Pclass3     -2.409134   0.286233  -8.417  < 2e-16 ***
Age         -0.024393   0.009062  -2.692 0.007104 **
Child1       1.344034   0.515947   2.605 0.009188 **
FamilySize  -0.298147   0.083020  -3.591 0.000329 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 891.19  on 668  degrees of freedom
Residual deviance: 580.39  on 662  degrees of freedom
AIC: 594.39

Number of Fisher Scoring iterations: 5
```

Al igual que pasaba en el modelo2, en este caso todas las variables utilizadas han resultado significativas, y además el AIC del modelo ha mejorado ligeramente. Además la capacidad predictiva del modelo ha aumentado un poco:

```
> pct.correcto_glm3 <- 100 * sum(diag(mat.confusion_glm3)) / sum(mat.confusion_glm3)
> print(sprintf("El %% de registros correctamente clasificados es: %.4f %%",
+               pct.correcto_glm3))
[1] "El % de registros correctamente clasificados es: 83.7838 %"
```

Y la matriz de confusión resultante es:



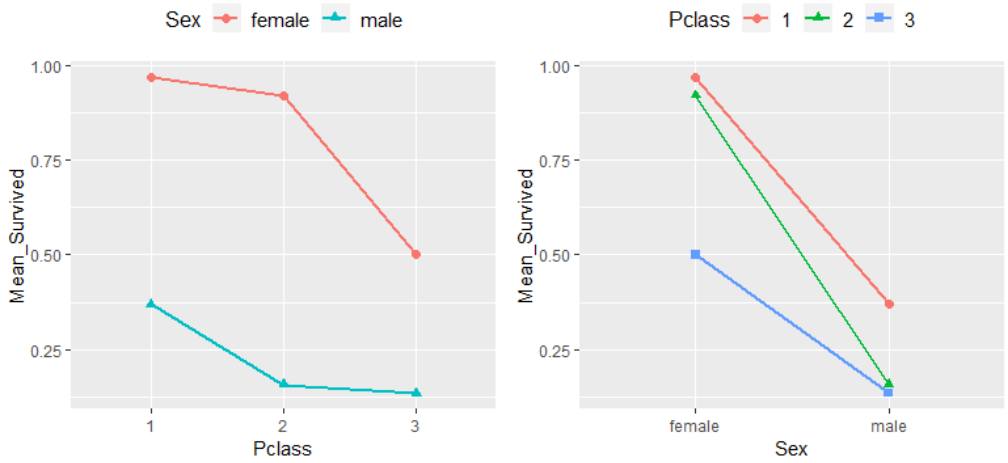
Antes de construir un nuevo modelo de regresión logística, vamos a intentar averiguar a través de unas gráficas (gráficas de perfil) si encontramos que algunas variables independientes afectan a la variable dependiente). Cuando los diferentes valores de una variable explicativa (diferentes valores de un factor) dan como resultado diferentes valores de la variable explicada (es decir obtenemos diferentes respuestas), entonces decimos que tenemos un efecto principal. La ausencia de efectos principales, gráficamente implican líneas horizontales, es decir que el valor da respuesta no varía, independientemente que varíe el valor del factor. En cambio, cuando la línea no es horizontal, entonces hay un efecto principal. Eso significa que los diferentes valores del factor afectan la respuesta de manera diferente. Y además cuanta más inclinada sea la pendiente de la línea, mayor será la magnitud del efecto principal.

Además, con estos gráficos, podremos visualmente comprobar si hay interacciones entre las variables independientes. Gráficamente esa situación se da cuando las líneas de los gráficos no son paralelas (con independencia de la pendiente que tengan ellas). Cuando las líneas se cruzan o cambian la pendiente, estamos ante situaciones que puede ser una interacción.

```
> #Para ello antes de nada vamos a definir una variable 'objetivo' pero numérica
> data$SurvivedNum <- as.integer(as.character(data$Survived))
> #Agrupamos el dataset por las variables Sex y Pclass
> data.agrup <- data %>% group_by(Sex, Pclass)
> #Calculamos la media creando un campo 'Mean_Survived' de la media
> data.mediasurvived <- summarise(.data = data.agrup, Mean_Survived = mean(SurvivedNum))
> kable(data.mediasurvived)
```

Sex	Pclass	Mean_Survived
female	1	0.9680851
female	2	0.9210526
female	3	0.5000000
male	1	0.3688525
male	2	0.1574074
male	3	0.1354467

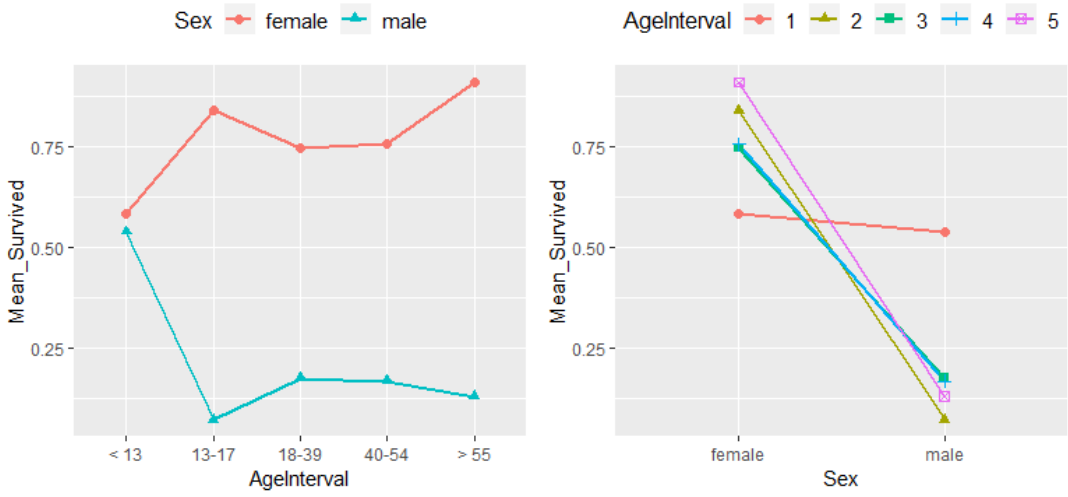
```
plot1 <- ggplot(data=data.mediasurvived, aes(x=Pclass, y = Mean_Survived, group=Sex))+
  geom_line(aes(color=Sex), size=1) +
  geom_point(aes(color=Sex, shape=Sex), size=2) +
  theme(legend.position="top") +
  theme(legend.title = element_text(size=12), legend.text = element_text(size = 11))
plot2 <- ggplot(data=data.mediasurvived, aes(x=Sex, y = Mean_Survived, group=Pclass))+
  geom_line(aes(color=Pclass), size=1) +
  geom_point(aes(color=Pclass, shape=Pclass), size=2) +
  theme(legend.position="top", legend.title = element_text(size=12)) +
  theme(legend.text = element_text(size = 11))
#library(gridExtra)
grid.arrange(plot1, plot2, ncol=2)
```



En este ejemplo tanto la variable "**Sex**" como la variable "**Pclass**" producen efecto en la variable "**Mean_Survived**". También podemos comprobar como se produce interacción entre las variables Sex y Pclass respecto a Mean_Survived. Podemos ver por ejemplo como hay un descenso muy pronunciado cuando pasamos de mujeres a hombres y estamos tratando la clase 2ª. Y también como el pasar a tercera clase afecta muy negativamente en el caso de las mujeres, ya que aunque no es una línea paralela, el hecho de estar en 1ª o 2ª clase no parece demasiado importante en el caso de las mujeres, pero al pasar a 3ª clase la caída de la media de supervivencia es muy importante.

De mismo modo podemos hacer esto con otras variables independientes y ver efectos e interacciones entre ellas. Nosotros escogimos otra, que es la variable nueva que definimos "**AgeInterval**" vs "**Sex**". En este caso llama la atención el impacto de cuando pasamos el primer intervalo (menores o iguales a 12 años) al siguiente intervalo (de 13 a 17 años).

En ese caso la proporción de las chicas aumenta mientras que en el caso de los hombres disminuye dástricamente. Además no parece haber apenas diferencia en el primer tramo (es decir los niños, en la misma proporción, se salvan independencia del género)

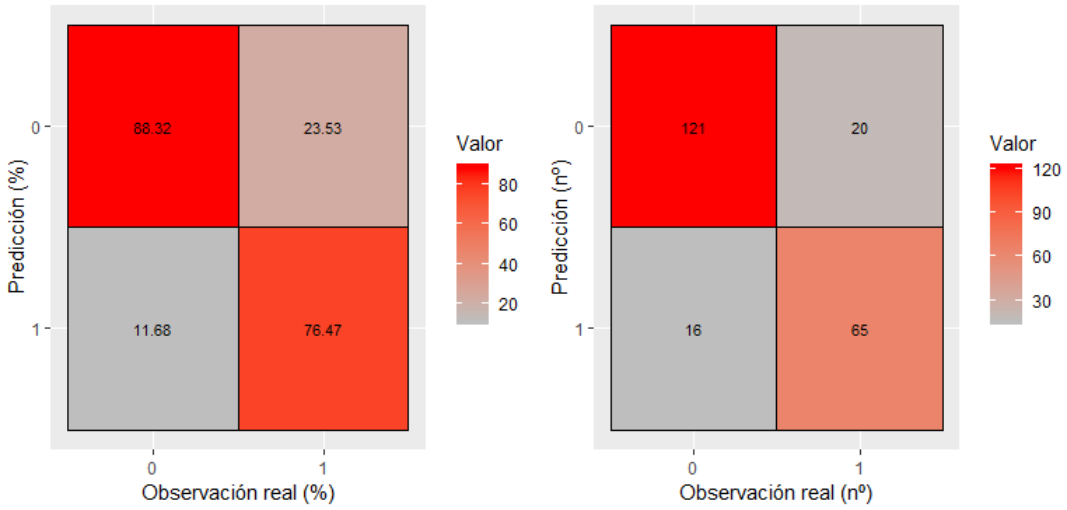


Vamos a construir ahora un 4º modelo de regresión logística, en el que además de algunas variables que ya hemos utilizado en otros modelos, usemos algunas interacciones, por ejemplo los 2 que acabamos de comentar: "AgeInterval":"Sex" y también hemos añadido la interacción que generan "Sex":"Pclass". A estas variables le sumamos también las variables independientes "**Sex**", "**Parch**" y "**SibSp**".

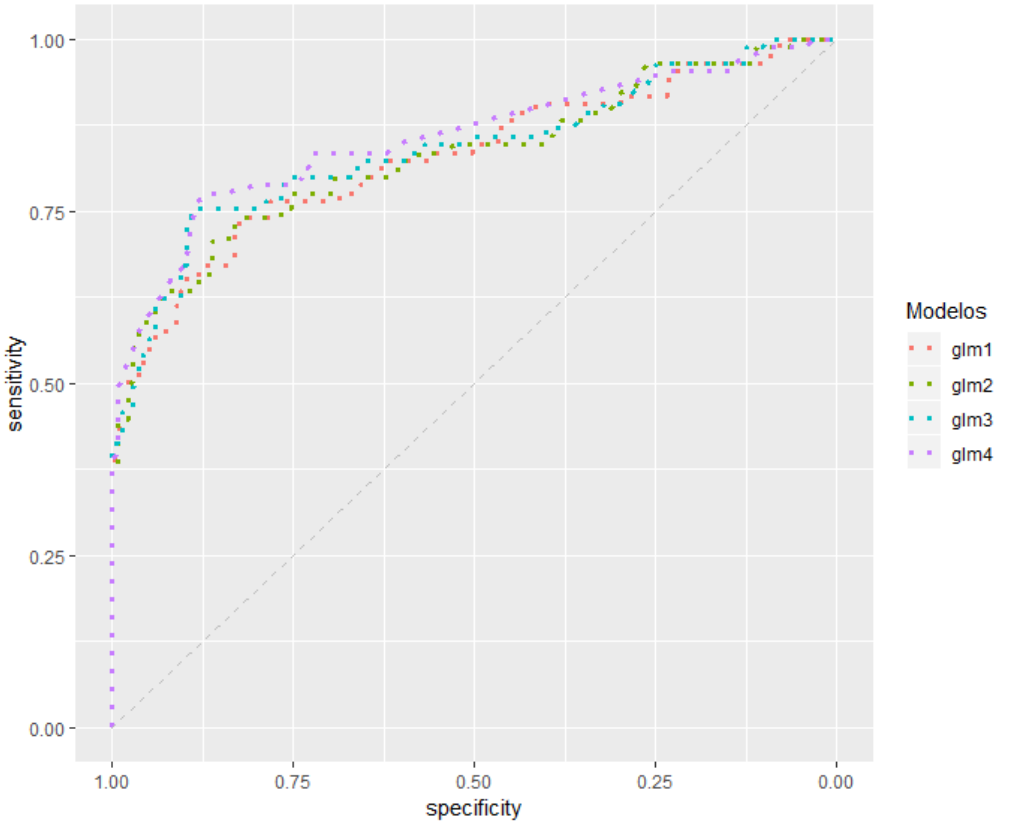
Y con esos datos construimos el modelo que sigue a continuación:

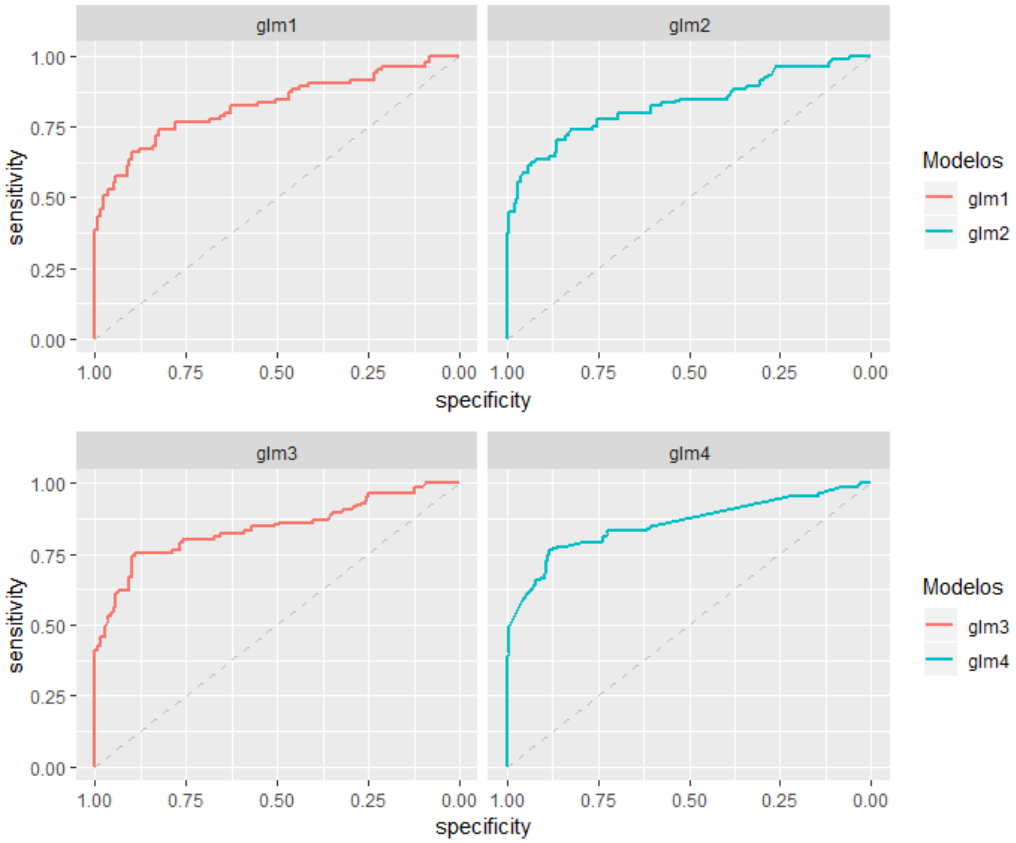
```
> modelo_glm4 <- glm(formula = Survived ~ Sex+Parch+SibSp+AgeInterval:Sex+Sex:Pclass,
+                    data = data_train, family = binomial(link = "logit"))
> summary(modelo_glm4)

Call:
glm(formula = Survived ~ Sex + Parch + SibSp + AgeInterval:Sex +
    Sex:Pclass, family = binomial(link = "logit"), data = data_train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.6250  -0.5204  -0.4157   0.3992   2.3904

Coefficients:
                       Estimate Std. Error z value Pr(>|z|)
(Intercept)              4.2863     0.8383   5.113 3.17e-07 ***
Sexmale                 -1.0710     0.9521  -1.125  0.26067
Parch                   -0.2200     0.1592  -1.382  0.16705
SibSp                   -0.4336     0.1350  -3.212  0.00132 **
Sexfemale:AgeInterval2   0.5086     0.8104   0.628  0.53027
Sexmale:AgeInterval2    -3.5392     0.9040  -3.915 9.04e-05 ***
Sexfemale:AgeInterval3  -0.6796     0.5469  -1.243  0.21405
Sexmale:AgeInterval3    -3.0646     0.5380  -5.696 1.23e-08 ***
Sexfemale:AgeInterval4  -1.1432     0.6923  -1.651  0.09867 .
Sexmale:AgeInterval4    -3.7630     0.6416  -5.865 4.49e-09 ***
Sexfemale:AgeInterval5  -0.4050     1.3130  -0.308  0.75773
Sexmale:AgeInterval5    -4.8723     0.8533  -5.710 1.13e-08 ***
Sexfemale:Pclass2       -0.7342     0.7618  -0.964  0.33513
Sexmale:Pclass2         -1.8033     0.4294  -4.200 2.67e-05 ***
Sexfemale:Pclass3       -3.4032     0.6483  -5.250 1.52e-07 ***
Sexmale:Pclass3         -2.0815     0.3468  -6.002 1.95e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 891.19  on 668  degrees of freedom
Residual deviance: 534.50  on 653  degrees of freedom
AIC: 566.5

Number of Fisher Scoring iterations: 5
> pct.correcto_glm4 <- 100 * sum(diag(mat.confusion_glm4)) / sum(mat.confusion_glm4)
> print(sprintf("El %% de registros correctamente clasificados es: %.4f %%",
+               pct.correcto_glm4))
[1] "El % de registros correctamente clasificados es: 83.7838 %"
> csstab_glm4 <- CrossTable(data_test$Survived, predict_glm4,
+                           prop.chisq  = FALSE, prop.c = FALSE, prop.r =FALSE,
+                           dnn = c('Reality', 'Prediction'))
```

La matriz de confusión queda como sigue:



Y continuación mostramos las curvas ROC de los modelos que acabamos de generar:

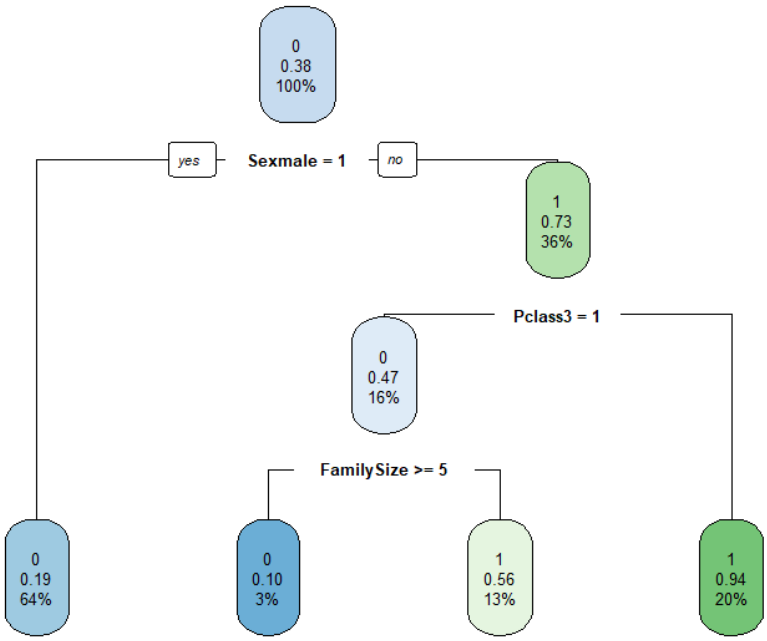

1.00
0.75
0.50
0.25
0.00
sensitivity
1.00
0.75
0.50
0.25
0.00
specificity
Modelos
glm1
glm2
glm3
glm4


glm1
glm2
glm3
glm4
1.00
0.75
0.50
0.25
0.00
sensitivity
1.00
0.75
0.50
0.25
0.00
specificity
Modelos
glm1
glm2
glm3
glm4

Árbol de decisión:

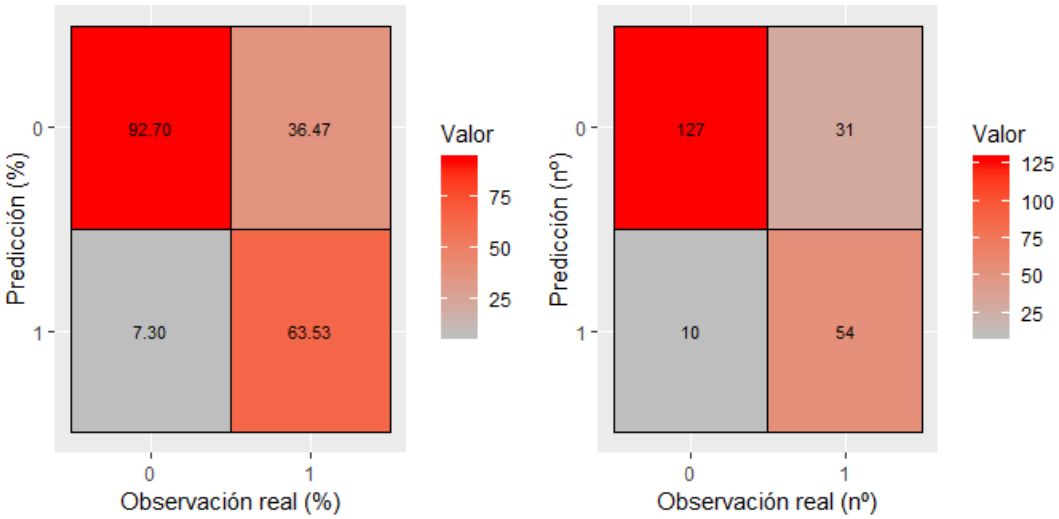
Pensamos también en crear un modelo de árbol de decisión. Lo mejor de estos modelos es su sencillez, ya que al final son reglas (condiciones) las que no llevan a clasificar las muestras. He aquí un ejemplo, realizado también con validación cruzada de datos.

```
#################################################################
#Arbol de decision
#################################################################
tctrl <- caret::trainControl(method = "cv",number=10, repeats = 3)
model_tree1 <- caret::train(Survived~Sex+FamilySize+Child+Age+Pclass,
                            data=data_train,
                            method="rpart",
                            trControl = tctrl)
summary(model_tree1)
#library(rpart.plot)
rpart.plot(model_tree1$finalModel)

#Realizamos la predicción con el modelo construido con rpart y el dataset test
predict_tree1 <-predict(model_tree1, data_test)

mat.confusion_tree1 <- table(data_test$Survived, predict_tree1)
mat.confusion_tree1
pct.correcto_tree1 <- 100 * sum(diag(mat.confusion_tree1)) / sum(mat.confusion_tree1)
print(sprintf("El %% de registros correctamente clasificados es: %.4f %%",
              pct.correcto_tree1))
csstab_tree1 <- CrossTable(data_test$Survived, predict_tree1,
                           prop.chisq  = FALSE, prop.c = FALSE, prop.r =FALSE,
                           dnn = c('Reality', 'Prediction'))
print(plot_matriz_confusion(csstab_tree1))
```




```
> pct.correcto_tree1 <- 100 * sum(diag(mat.confusion_tree1)) / sum(mat.confusion_tree1)
> print(sprintf("El %% de registros correctamente clasificados es: %.4f %%",
+               pct.correcto_tree1))
[1] "El % de registros correctamente clasificados es: 81.5315 %"
```

Y la matriz de confusión es la siguiente:

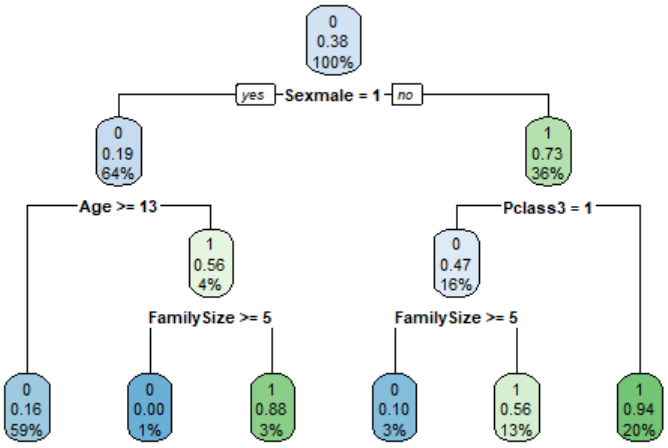


Veremos que, manteniendo exactamente las mismas variables predictivas, pero cambiando un parámetro en el modelo (concretamente maxdepth, que es la profundidad máxima del árbol) este puede llegar a mejorar los resultados obtenidos:

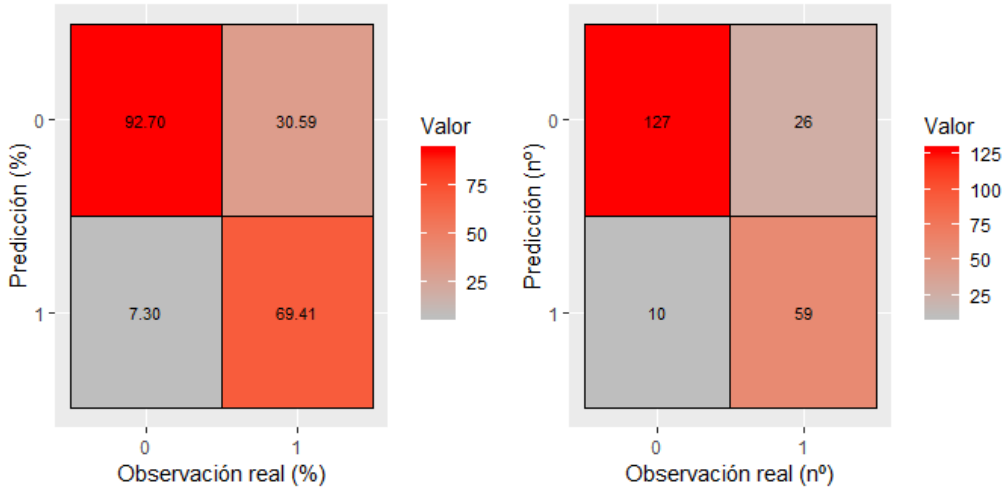
```
#####################################################################
# Arbol decision2
#####################################################################
tctrl <- caret::trainControl(method = "repeatedcv",
                             number=10, repeats = 3)
tgrid <- data.frame(maxdepth = seq(2,10,1))

model_tree2<- train(Survived ~ Sex+FamilySize+Child+Age+Pclass,
                    data=data_train,
                    trControl=tctrl,
                    tuneGrid=tgrid,
                    method="rpart2")

rpart.plot(model_tree2$finalModel)
summary(model_tree2)

#Realizamos la predicción con el modelo construido con rpart y el dataset test
predict_tree2 <-predict(model_tree2, data_test)
mat.confusion_tree2 <- table(data_test$Survived, predict_tree2)
mat.confusion_tree2
pct.correcto_tree2 <- 100 * sum(diag(mat.confusion_tree2)) / sum(mat.confusion_tree2)
print(sprintf("El %% de registros correctamente clasificados es: %.4f %%",
              pct.correcto_tree2))
print(paste0("El mejor parámetro para el árbol de clasificación ha sido: ",
           names(model_tree2$bestTune),"=", model_tree2$bestTune))
csstab_tree2 <- CrossTable(data_test$Survived, predict_tree2,
                           prop.chisq  = FALSE, prop.c = FALSE, prop.r =FALSE,
                           dnn = c('Reality', 'Prediction'))
print(plot_matriz_confusion(csstab_tree2))
```

Los resultados de la predicción han mejorado algo respecto al árbol de clasificación 1:

```
> print(sprintf("El %% de registros correctamente clasificados es: %.4f %%",
+               pct.correcto_tree2))
[1] "El % de registros correctamente clasificados es: 83.7838 %"
> print(paste0("El mejor parámetro para el árbol de clasificación ha sido: ",
+             names(model_tree2$bestTune),"=", model_tree2$bestTune))
[1] "El mejor parámetro para el árbol de clasificación ha sido: maxdepth=5"
```

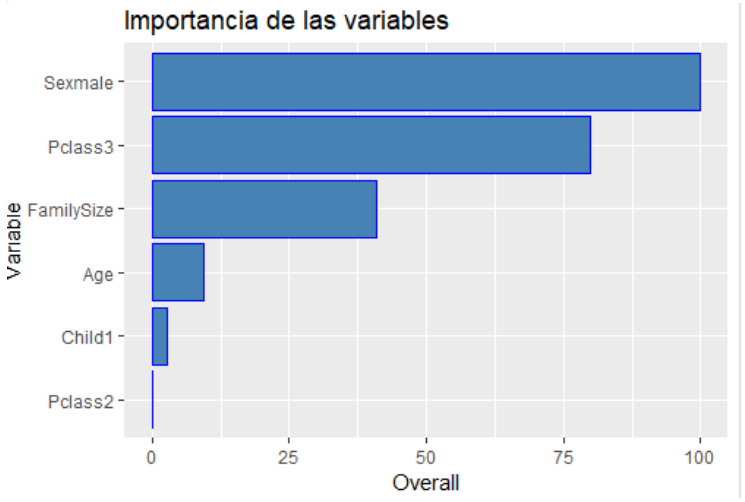


La matriz de confusión obtenida es la siguiente:



Podemos ver la importancia que el árbol le ha dado a las variables utilizadas para su construcción:

```
datosVarImp <- varImp(model_tree2)$importance %>%
  as.data.frame() %>%
  rownames_to_column() %>%
  arrange(Overall)
datosVarImp$Variable <- reorder(datosVarImp$rowname, datosVarImp$Overall)
ggplot(datosVarImp)+
  geom_col(aes(x = Variable, y = Overall), fill="steelblue", color="blue")+
  coord_flip() + ggtitle("Importancia de las variables")
```



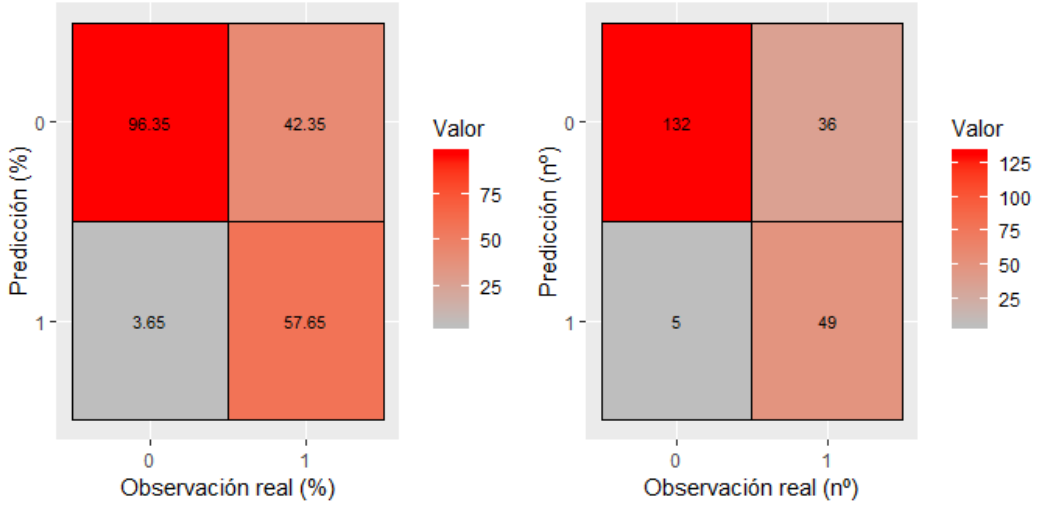
Random Forest

Vamos ahora a crear un modelo utilizando Random Forest, que consiste en una combinación de árboles predictores. Cada árbol utiliza muestras con reemplazo del conjunto de datos que se le pasa al modelo. Y también cada árbol utiliza números diferentes de variables, para de esa forma dar también opciones a algunas variables que podrían quedar eclipsadas por otras con más relevancia. Y de ahí que luego se pueda obtener la importancia relativa de cada variable (el clasificador altera la variable y mide el impacto). El uso de esta técnica nos va a permitir, respecto a los árboles de clasificación, un sistema más estable. No significa que vaya a tener mejores resultados, pero sí que es más robusto, es menos sensible a la variación de los datos del conjunto de entrada.

```
##############################################################################
tctrl <- caret::trainControl(method = "repeatedcv",
                             number=10, repeats = 3)
model_rf <- caret::train(Survived ~ Sex+Parch+SibSp+Child+Age+Pclass,
                         data = data_train,
                         method = "rf",
                         trControl = tctrl,
                         verbose = FALSE)
plot(model_rf)
predict_rf <- predict(model_rf, data_test)
mat.confusion_rf <- table(data_test$Survived, predict_rf)
mat.confusion_rf
pct.correcto_rf<-100 * sum(diag(mat.confusion_rf)) / sum(mat.confusion_rf)
print(sprintf("El %% de registros correctamente clasificados es: %.4f %%",
              pct.correcto_rf))
csstab_rf <- CrossTable(data_test$Survived, predict_rf,
                        prop.chisq  = FALSE, prop.c = FALSE, prop.r =FALSE,
                        dnn = c('Reality', 'Prediction'))
print(plot_matriz_confusion(csstab_rf))
#Importancia de las variables
datosVarImp <- varImp(model_rf)$importance %>%
  as.data.frame() %>%
  rownames_to_column() %>%
  arrange(Overall)
datosVarImp$Variable <- reorder(datosVarImp$rowname, datosVarImp$Overall)
ggplot(datosVarImp)+
  geom_col(aes(x = Variable, y = Overall), fill="steelblue", color="blue")+
  coord_flip() + ggtitle("Importancia de las variables")
```

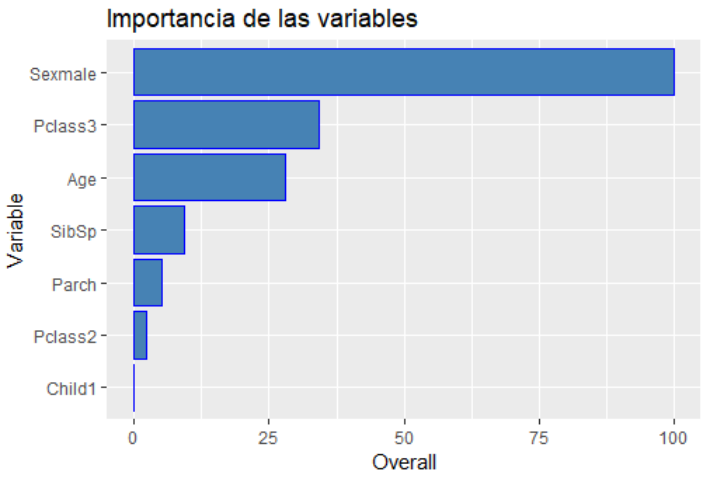
```
> pct.correcto_rf<-100 * sum(diag(mat.confusion_rf)) / sum(mat.confusion_rf)
> print(sprintf("El %% de registros correctamente clasificados es: %.4f %%",
+               pct.correcto_rf))
[1] "El % de registros correctamente clasificados es: 81.5315 %"
```

La matriz de confusión es la siguiente:



Y la importancia de las variables es la siguiente:

```
#Importancia de las variables
datosVarImp <- varImp(model_rf)$importance %>%
  as.data.frame() %>%
  rownames_to_column() %>%
  arrange(Overall)
datosVarImp$variable <- reorder(datosVarImp$rowname, datosVarImp$Overall)
ggplot(datosVarImp)+
  geom_col(aes(x = variable, y = Overall), fill="steelblue", color="blue")+
  coord_flip() + ggtitle("Importancia de las variables")
```



Gradient Boosting

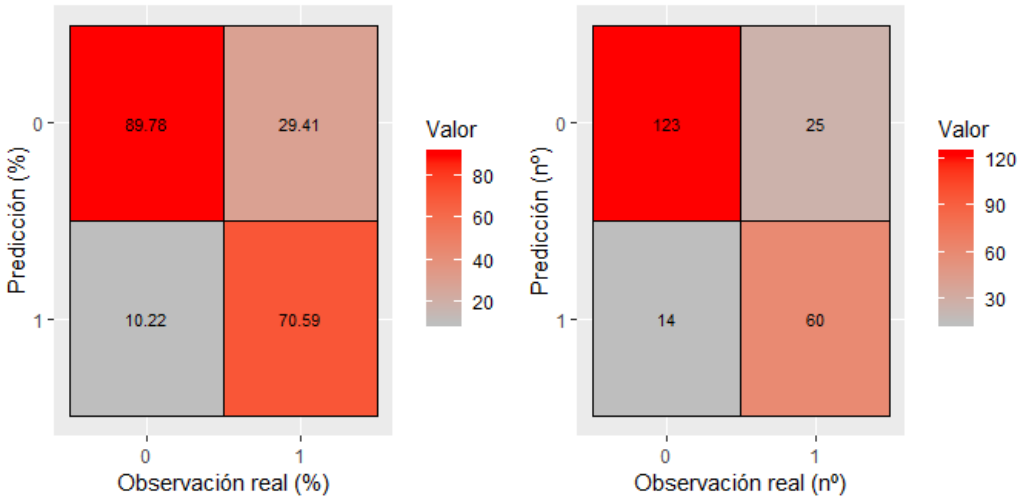
Ya por último en cuanto a modelos, vamos a crear ahora un modelo basado en boosting, que consiste básicamente en ir creando clasificadores encadenados y que van potenciando los errores, dándoles más peso, para los siguientes clasificadores. Generalmente los clasificadores son árboles de decisión. Para este caso concreto hemos utilizado xgboost, y hemos tenido que convertir las variables categóricas en variables de tipo numérico (integer) para que el modelo funcione.

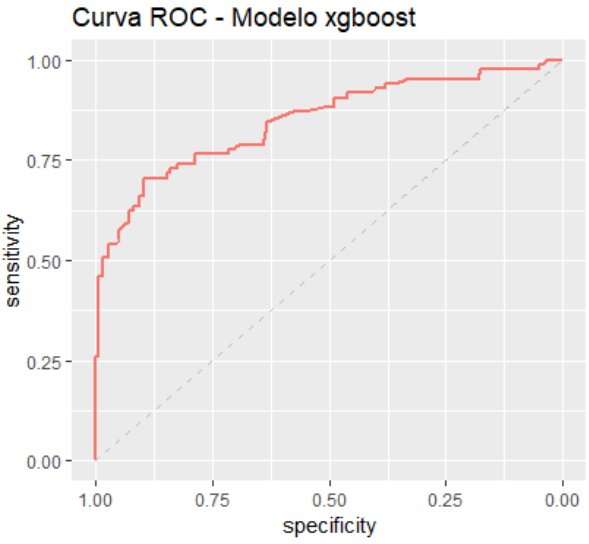
```
train_xgb <- data_train
test_xgb <- data_test

train_xgb$Sex <- as.integer(ifelse((train_xgb$Sex == "male"),1,0))
test_xgb$Sex <- as.integer(ifelse((test_xgb$Sex == "male"),1,0))
train_xgb$Pclass <- as.integer(as.character(train_xgb$Pclass))
test_xgb$Pclass <- as.integer(as.character(test_xgb$Pclass))
train_xgb$Embarked <- as.integer(ifelse((train_xgb$Embarked == "C"),0,
                                        ifelse(train_xgb$Embarked=='Q',1,2)))
test_xgb$Embarked <- as.integer(ifelse((test_xgb$Embarked == "C"),0,
                                       ifelse(test_xgb$Embarked=='Q',1,2)))
train_xgb$Child <- as.integer(as.character(train_xgb$Child))
test_xgb$Child <- as.integer(as.character(test_xgb$Child))
train_xgb$AgeInterval <- as.integer(as.character(train_xgb$AgeInterval))
test_xgb$AgeInterval <- as.integer(as.character(test_xgb$AgeInterval))
train_xgb$Survived <- as.integer(as.character(train_xgb$Survived))
test_xgb$Survived <- as.integer(as.character(test_xgb$Survived))

df_status(train_xgb)  #Ya están convertidas las variables categóricas a integer
```

```
predictores <- c('Sex', 'Pclass','AgeInterval','Child', 'Fare')
var.objetivo <- train_xgb$Survived
predictores <- c('Sex', 'Child', 'Pclass', 'AgeInterval','Fare') #82.43
model_xgb <- xgboost(data = as.matrix(train_xgb[,predictores]),
                     label = var.objetivo,
                     objective = "binary:logistic",
                     eval_metric = "logloss",
                     max_depth = 8,
                     nfold=4,
                     nrounds = 80)
predict_xgb <- predict(model_xgb, as.matrix(test_xgb[,predictores]))
rxgb = pROC::roc(response=test_xgb$Survived, predictor = predict_xgb)
plot(rxgb, lty=2, lwd=2)
print(rxgb$auc)
valor_threshold <- coords(rxgb, "best", ret = "threshold")

predict_xgb <- ifelse(predict_xgb <= valor_threshold, 0, 1)
mat.confusion_xgb <- table(test_xgb$Survived, predict_xgb)
mat.confusion_xgb
porcentaje.correcto_xgb <- 100 * sum(diag(mat.confusion_xgb)) / sum(mat.confusion_xgb)
print(sprintf("El %% de registros correctamente clasificados es: %.4f %%",
              porcentaje.correcto_xgb))
```

```
> porcentaje.correcto_xgb <- 100 * sum(diag(mat.confusion_xgb)) / sum(mat.confusion_xgb)
> print(sprintf("El %% de registros correctamente clasificados es: %.4f %%",
+               porcentaje.correcto_xgb))
[1] "El % de registros correctamente clasificados es: 82.4324 %"
```

La matriz de confusión del modelo es:

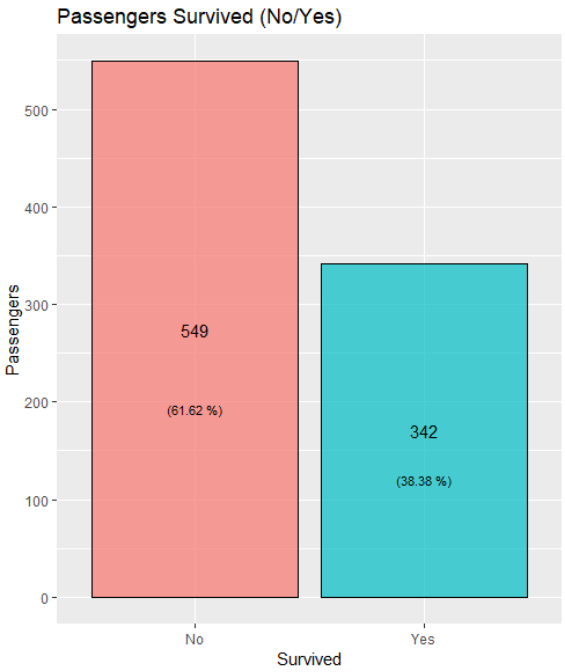


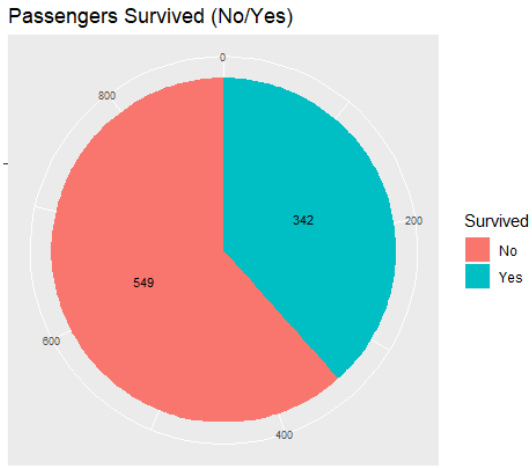


5. Representación de los resultados a partir de tablas y graficas.

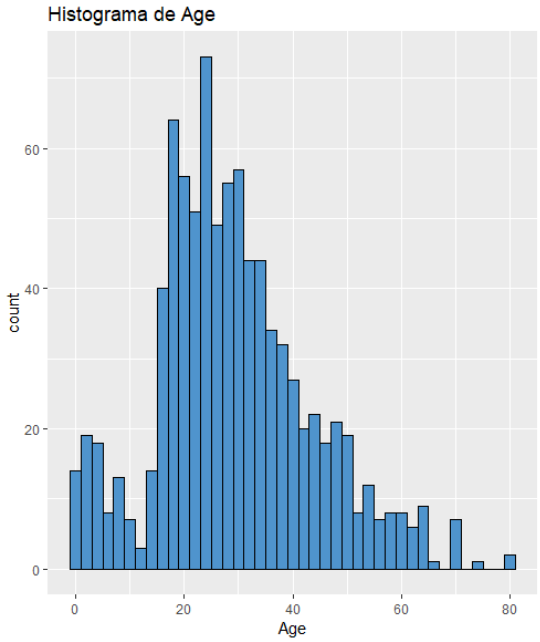
A lo largo de las etapas anteriores se han ido añadiendo tablas y gráficos que han representado los resultados que se han querido mostrar. Adicionalmente se añaden aquí otros gráficos que pueden ser de interés: Inicialmente vamos a poner los gráficos del análisis descriptivo de los datos. *(En lo que respecta a los gráficos y para que no sea muy pesado de visualizar hemos decido omitir el código que genera dichos gráficos, aunque están todos comentados y en el mismo orden que figuran en este documento).*

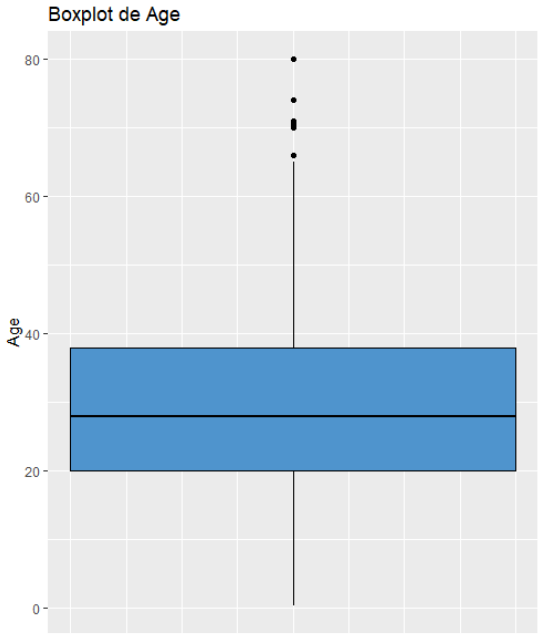
*(Variable "**Survived**")*



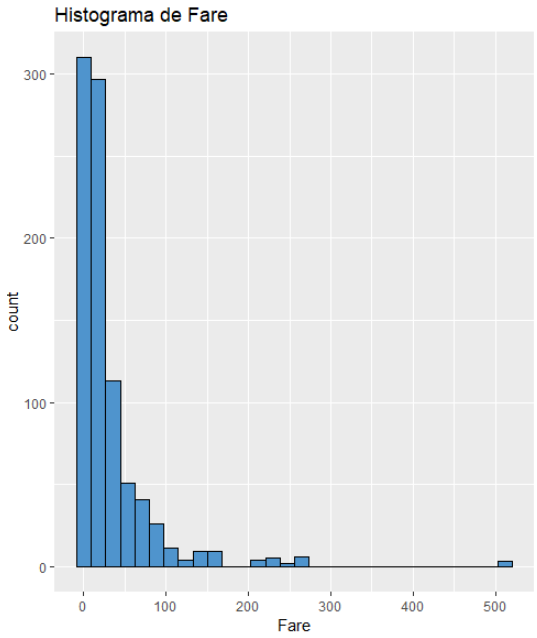


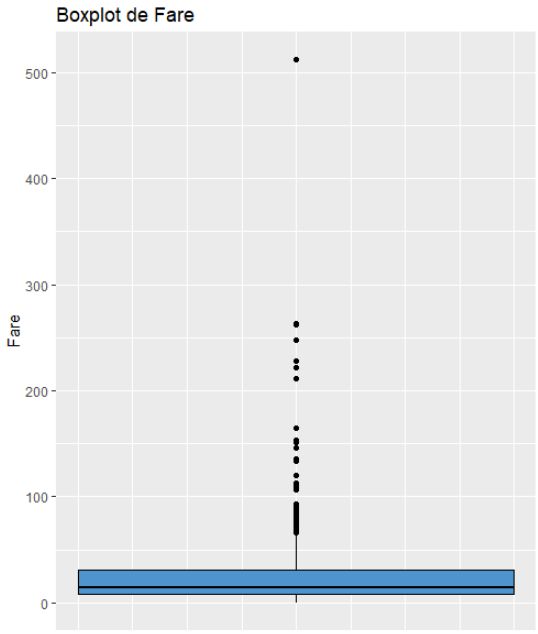
*(Variable "**Age**")*



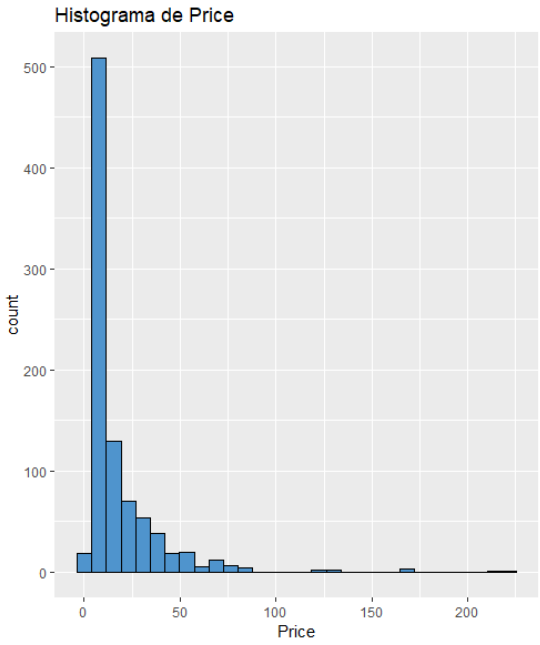


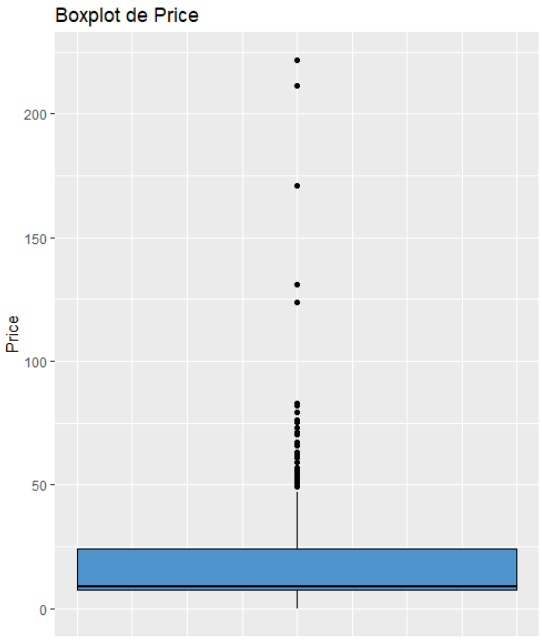
*(Variable "**Fare**")*



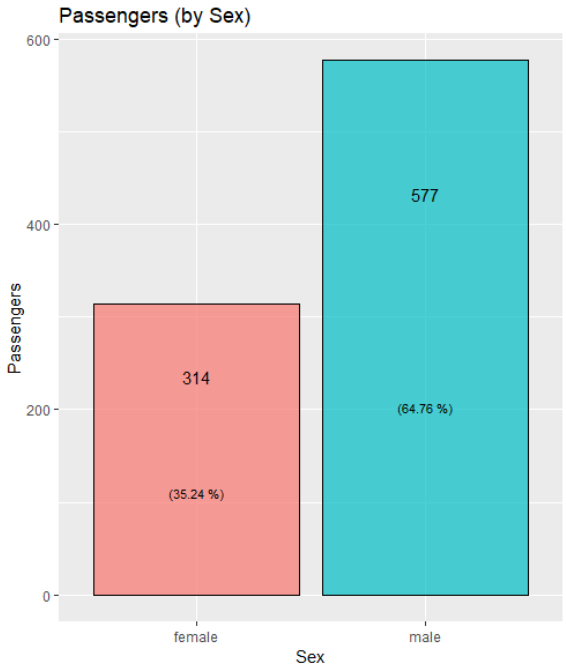


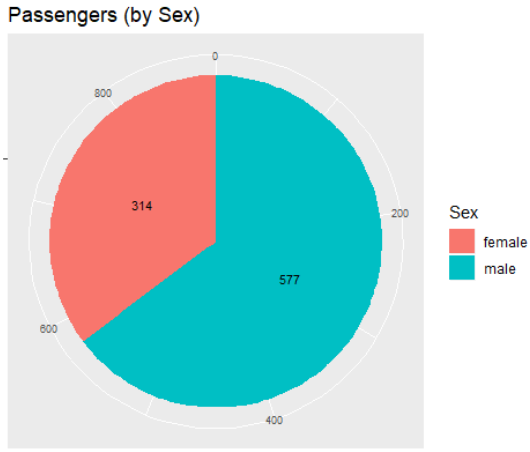
*(Variable "**Price**" que no pertenecía al dataset original)*



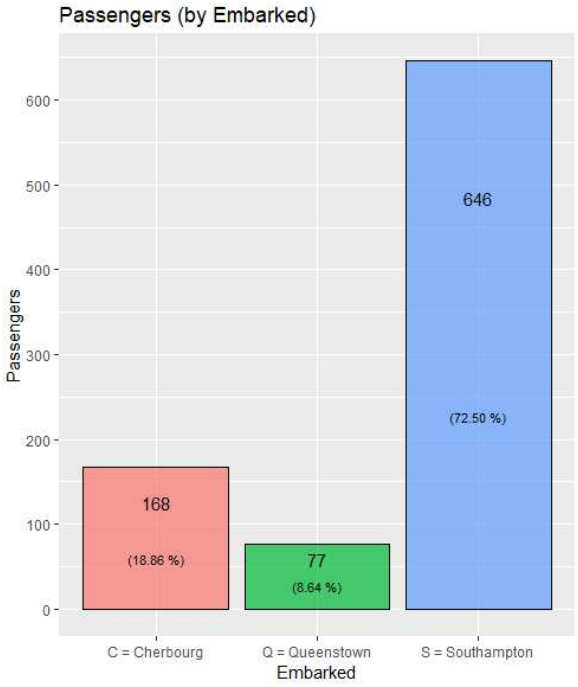


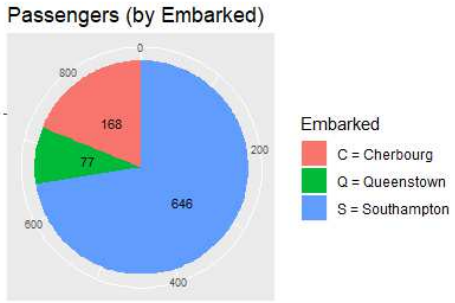
*(Variable "**Sex**")*

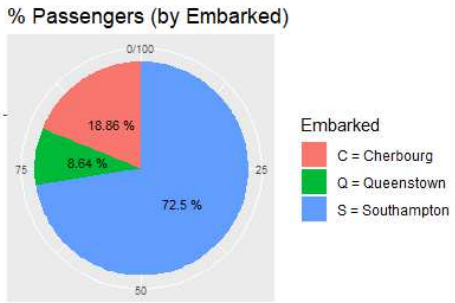




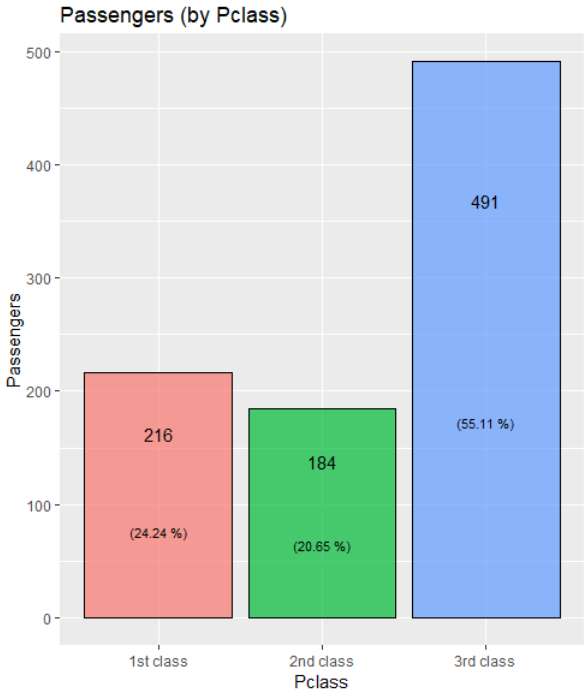
*(Variable "**Embarked**")*

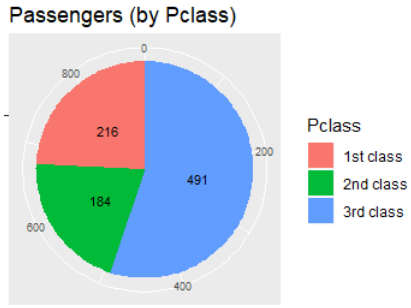


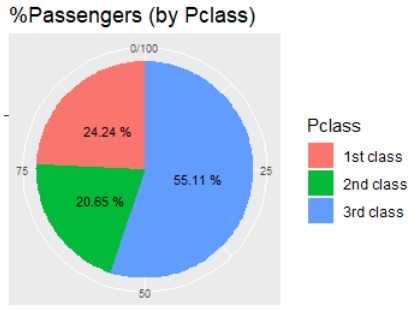




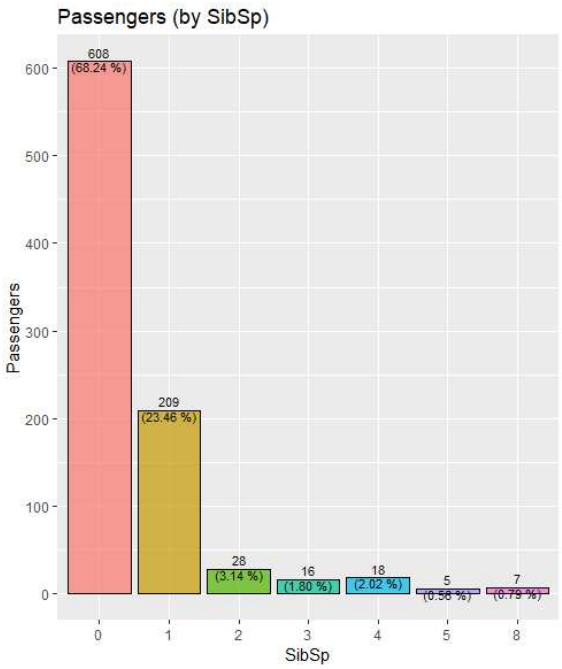
*(Variable "**Pclass**")*

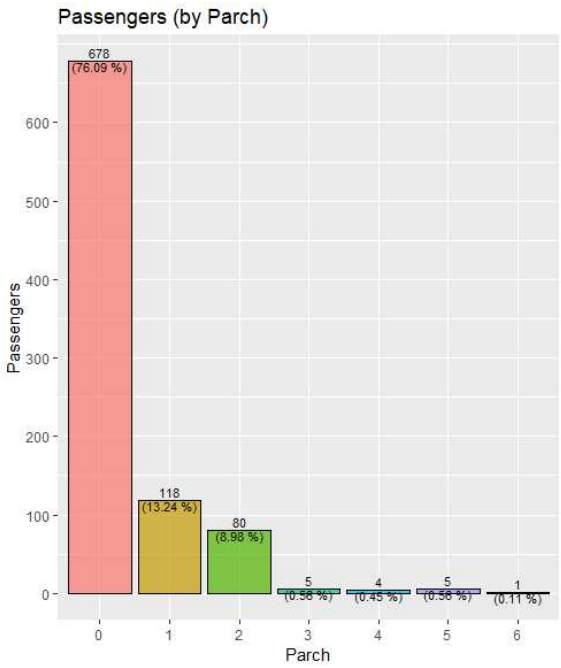




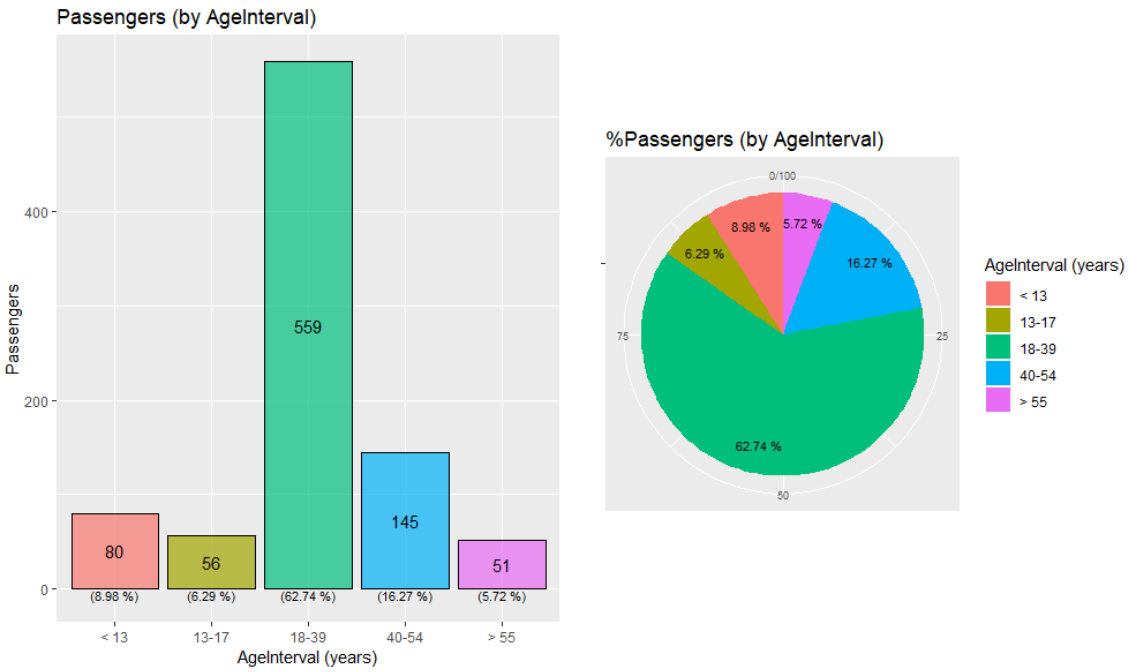


*(Variables "**SibSp**" y "**Parch**". La variable "**FamilySize**" es la suma de ambas +1)*

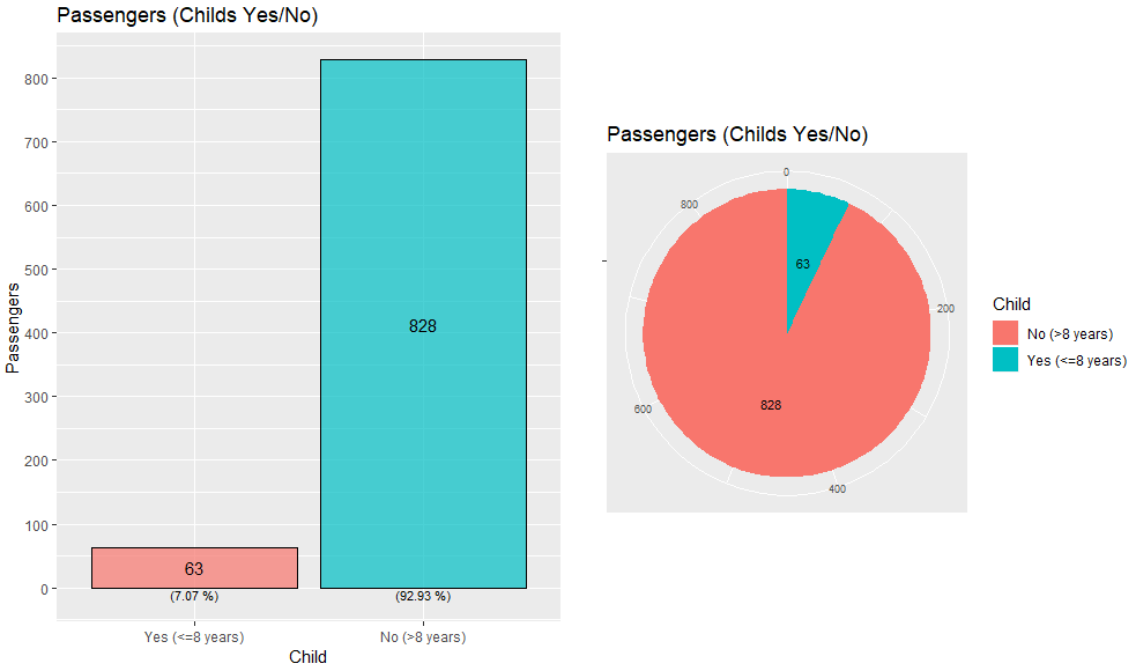




*(Variable "**AgeInterval**", que no pertenecía al dataset original)*

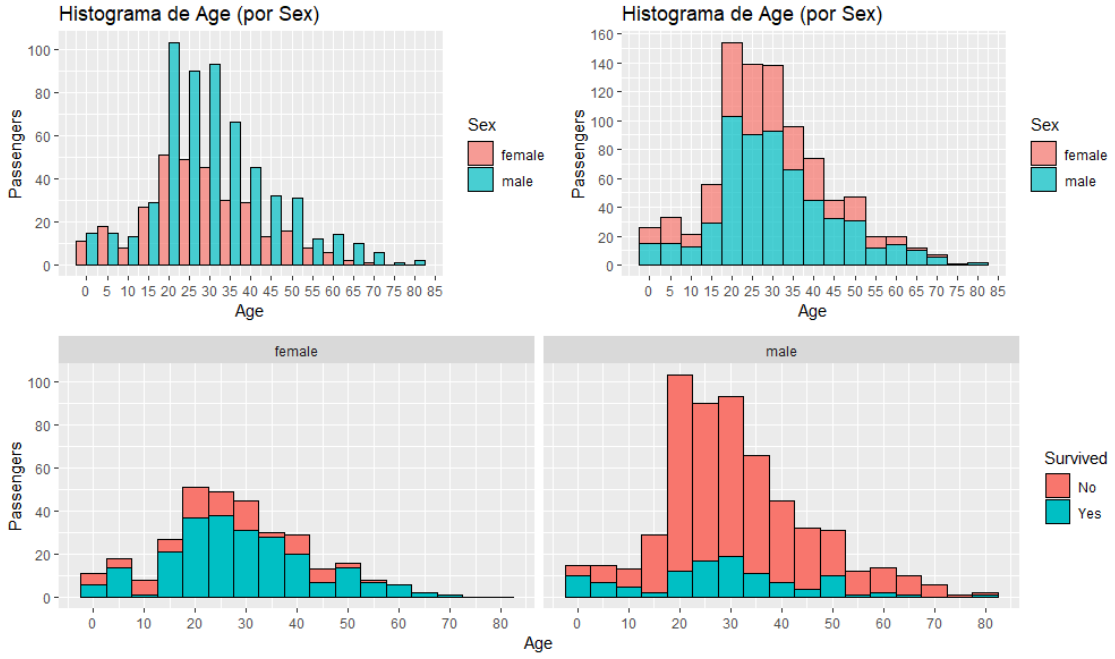


*(Variable "**Child**", que no pertenecía al dataset original)*

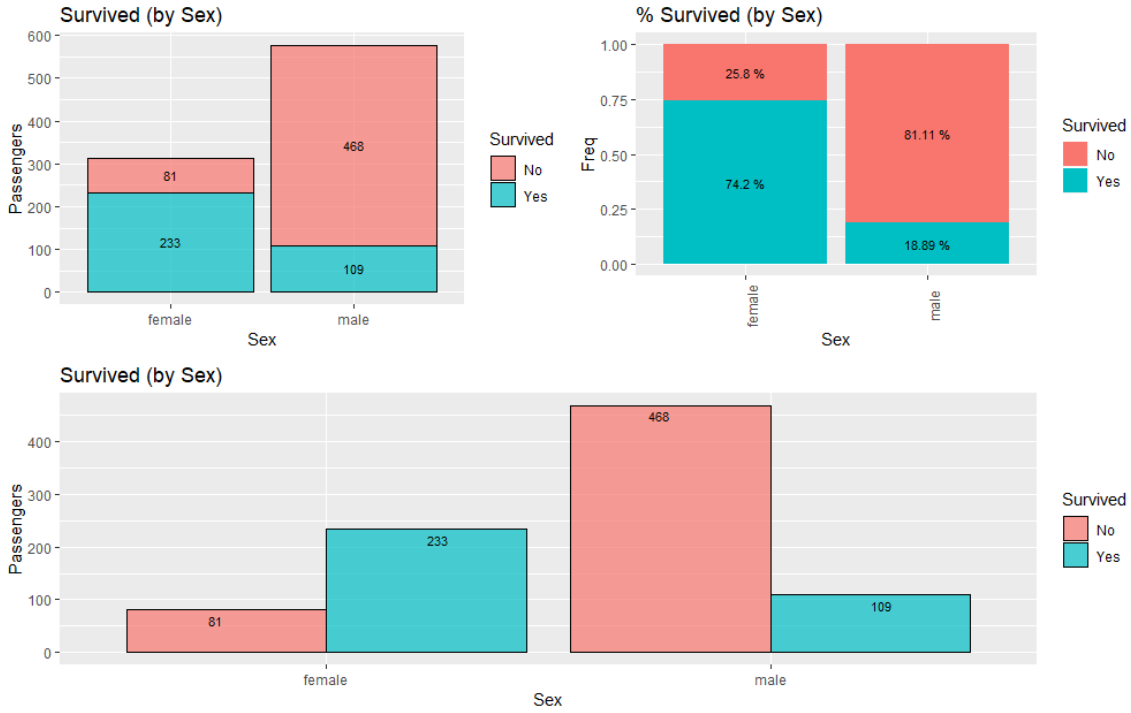


Una vez que ha hemos visto gráficamente cómo son las variables de nuestro dataset, ahora vamos a ver cómo interactúan unas con otras, donde podremos ver realidades desde diferentes puntos de vista, y que han sido objeto de análisis en ese trabajo.

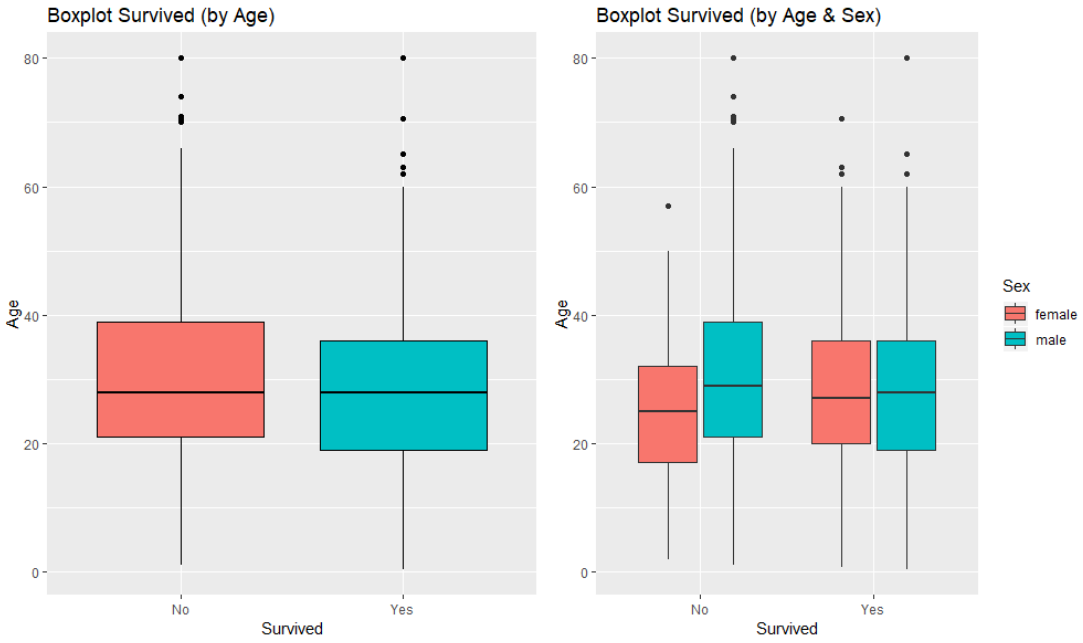
*(En estos gráficos podemos ver cómo es la distribución de la variable "**Age**" pero teniendo en cuenta el género "**Sex**". También podemos ver cómo es la distribución ya separadas por sexo, pero ahora teniendo en cuenta la variable "**Survived**". Aquí ya se puede apreciar que las mujeres tienen un índice de supervivencia muy superior al de los hombres)*



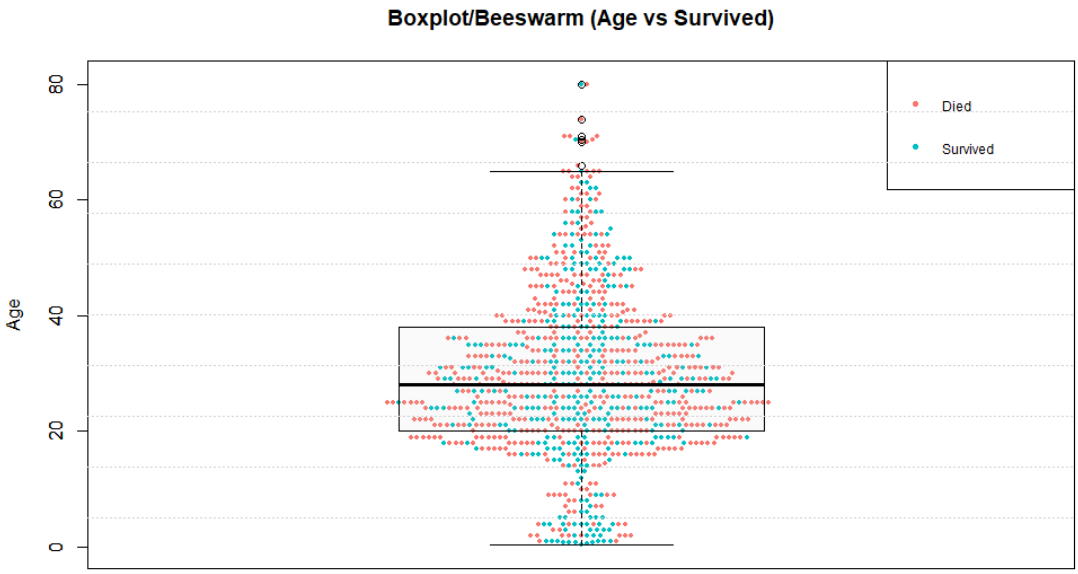
(Y los siguientes gráficos lo confirman, las mujeres tienen muchas más posibilidades de sobrevivir, de hecho las gráficas de proporciones están casi invertidas)



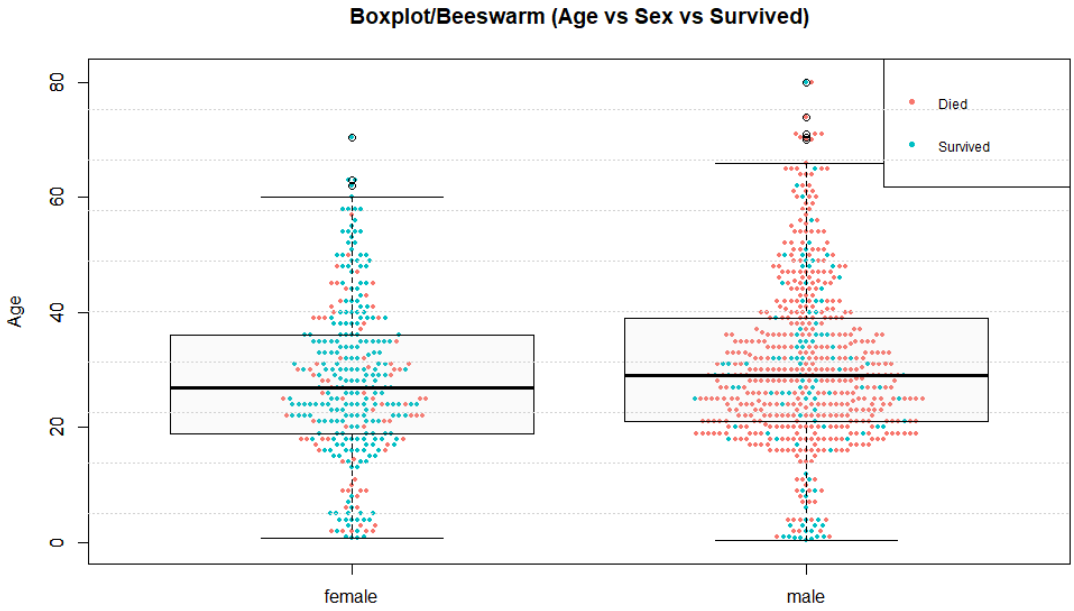
(Ahora se representan unos boxplots dónde podemos ver que la media de edad de las personas que mueren y que sobreviven es bastante parecida. También podemos ver que sí se aprecian diferencias en cuanto a la edad cuando visualizamos los datos separados por género. Apenas se observan diferencias en la media de edad de las personas que sobrevivieron (en cuanto al sexo), pero sí se puede apreciar que la media de edad de las personas que fallecieron era como 5 años más alta en el caso de los hombres)



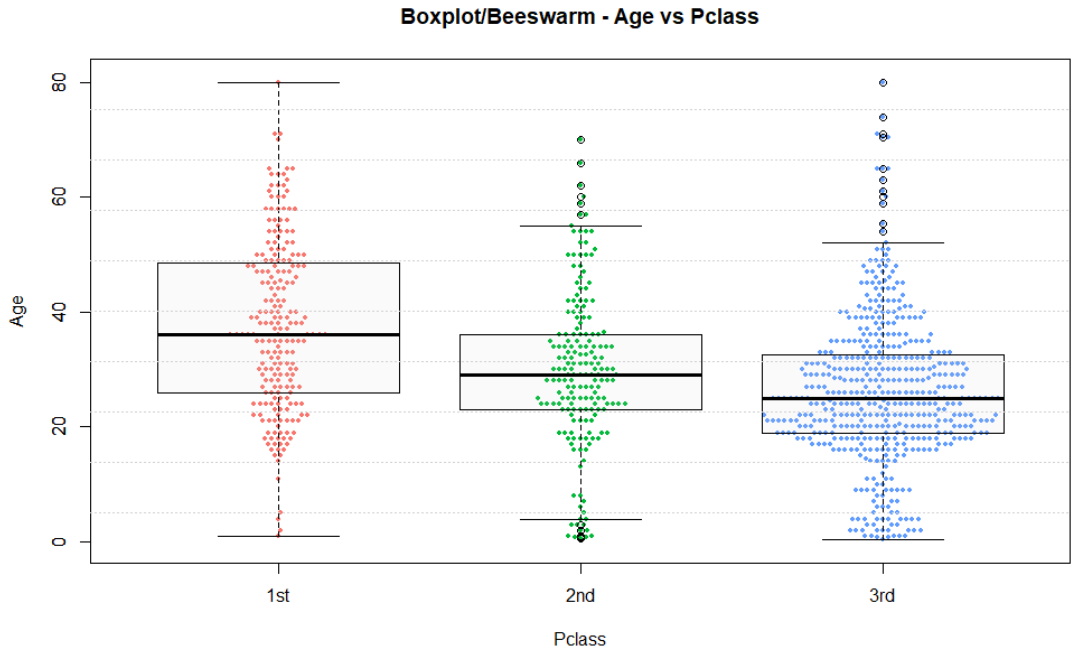
*(Este gráfico nos permite visualizar un boxplot de la variable "**Age**" pero nos enseña las diferentes muestras, por colores en función de si han sobrevivido o fallecido. De esta forma podemos ver si hay tendencias, es decir si unas edades parecen más propensas que otras para salvarse o morir)*



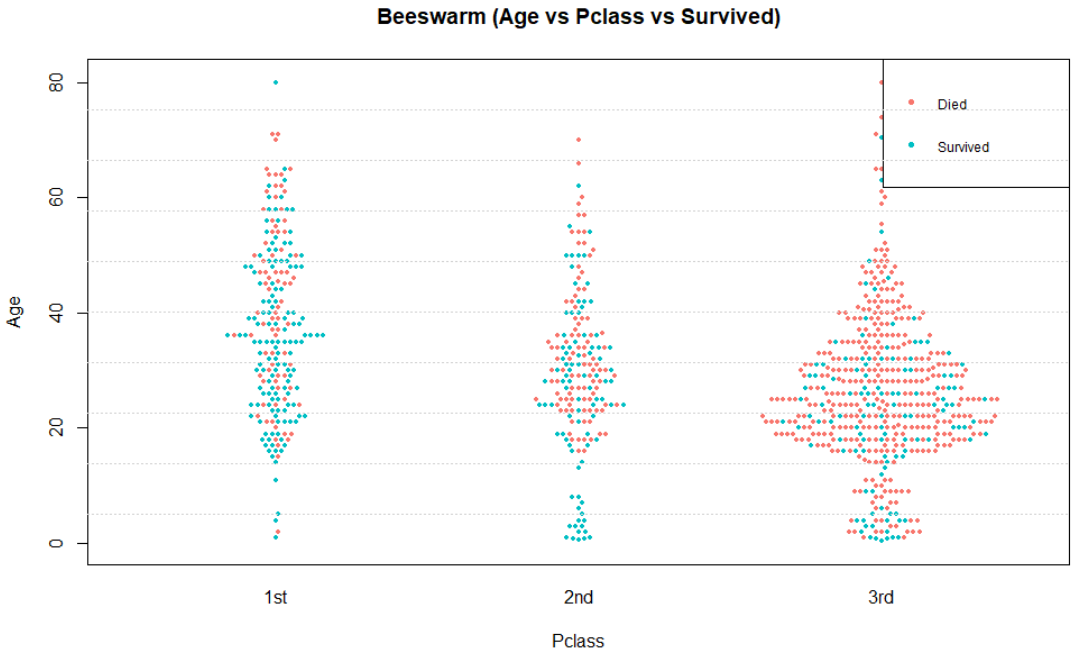
*(Este gráfico es como el anterior, pero dividido por "**Sex**". Podemos ver que los niños más o menos se salvan a partes iguales, pero en cambio se nota mucho que en las edades más adultas apenas hay supervivencia masculina, y en cambio en la femenina eso no ocurre. De hecho, las muertes femeninas se distribuyen a lo largo de todas las edades sin grandes acumulaciones).*



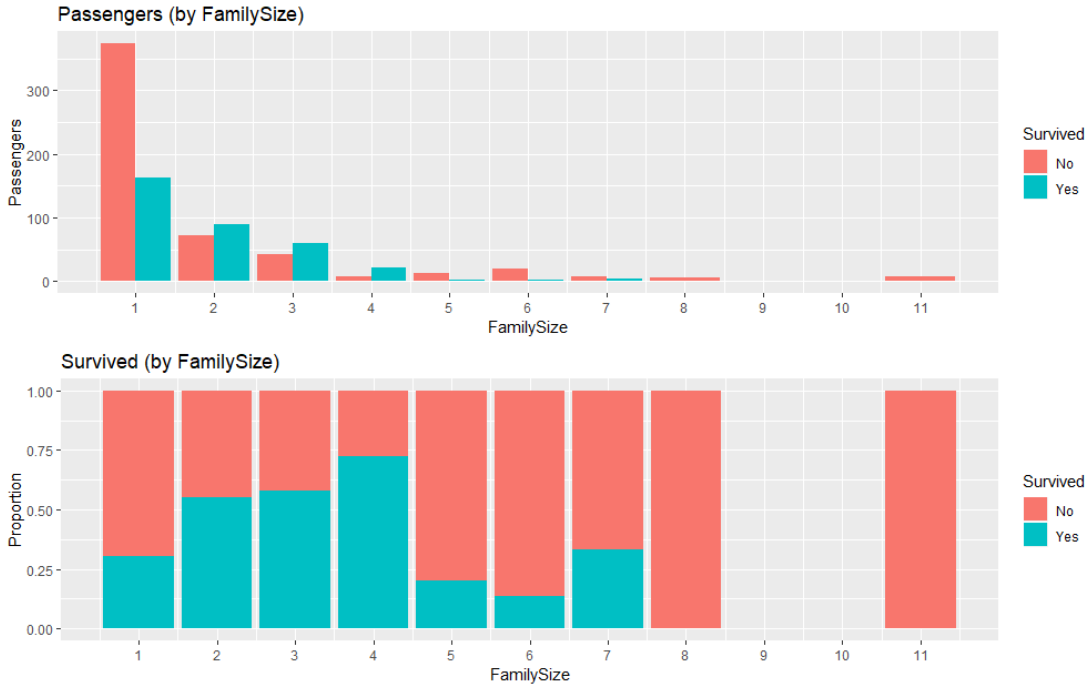
*(Este tipo de gráfico con el que podemos además superponer el boxplot, nos permite ver como se distribuyen los datos, en este caso la edad "**Age**" en los diferentes grupos, por ejemplo en este caso la clase "**Pclass**". Se puede ver como la primera clase es una columna estilizada, es decir los datos se distribuyen casi uniformemente desde aproximadamente los 20 años hasta los 60. En cambio en la 3ª clase se ven los datos muy agrupados entre los 15 y los 40 años).*



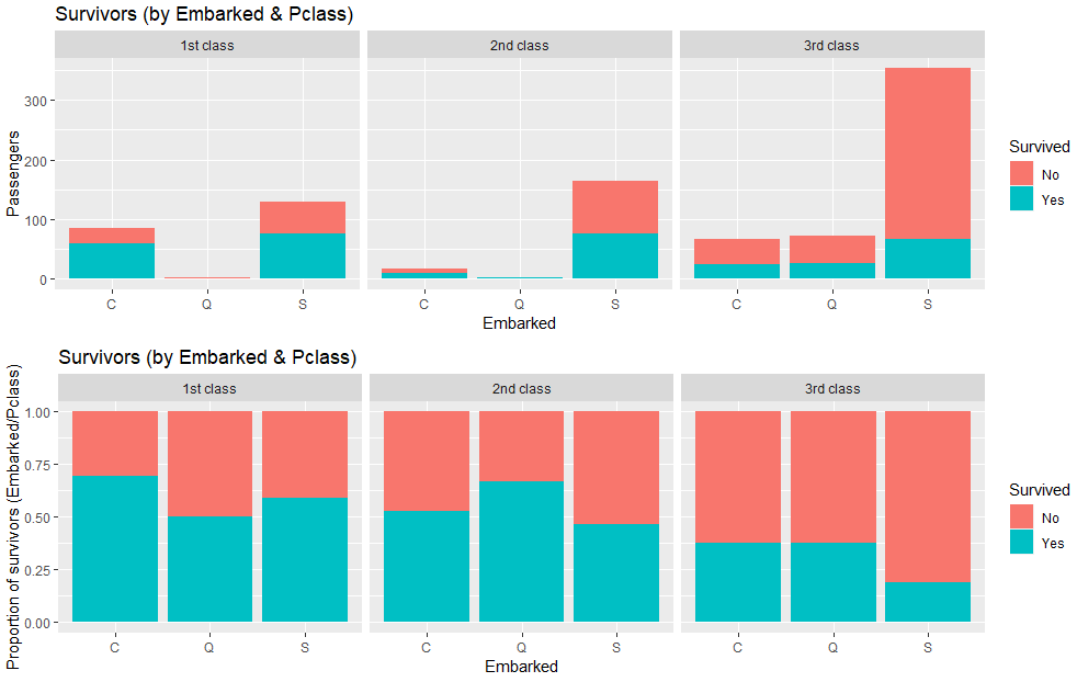
*(Ahora vemos el mismo gráfico de antes (sin el boxplot) pero añadiendo la variable "**Survived**" en modo de color. Llama la atención que en la zona de los niños, en la 2ª clase se salvan todos, en la primera clase solamente muere 1, en cambio en la tercera clase mueren aproximadamente la mitad. O sea que al final aquello de las mujeres y los niños primero, se cumple a medias, porque la clase en la que se viaja también está afectando).*



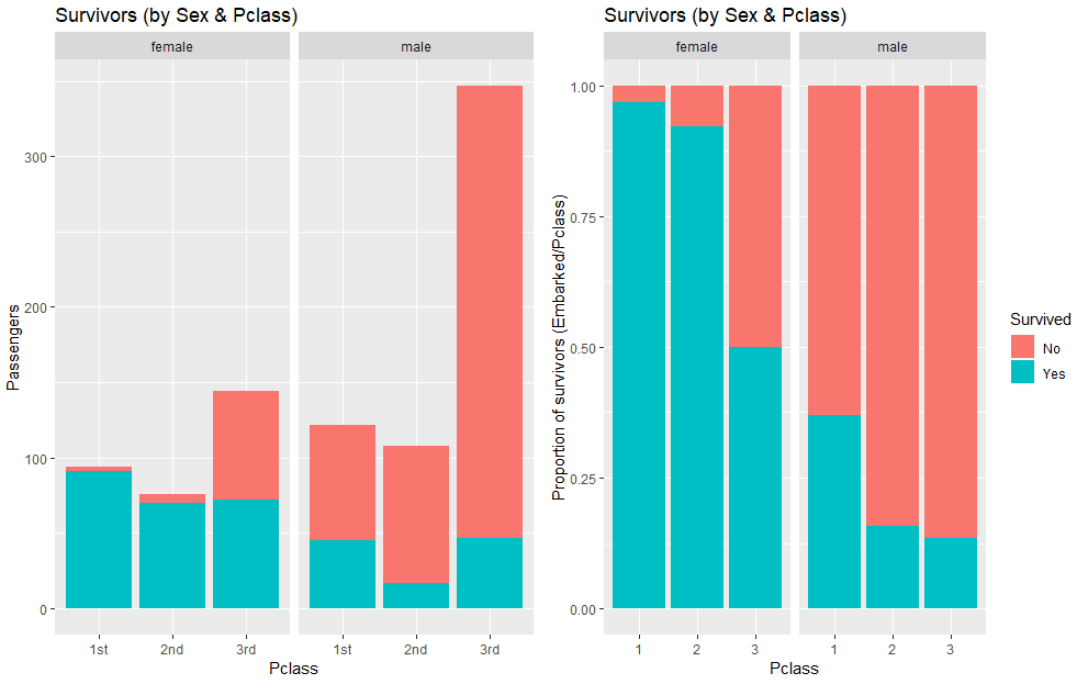
*(Esta variable "**FamilySize**" es la suma de "**SibSp**" y "**Parch**". Aquí podemos ver que mas de 500 personas viajaban "solas", o al menos sin nadie de parentesco, y de esas personas solo sobrevivió aproximadamente el 30%. El hecho de viajar "solo" normalmente es condición de mayoría de edad, y vimos que eso es un hándicap a la hora de tener más opciones de supervivencia).*



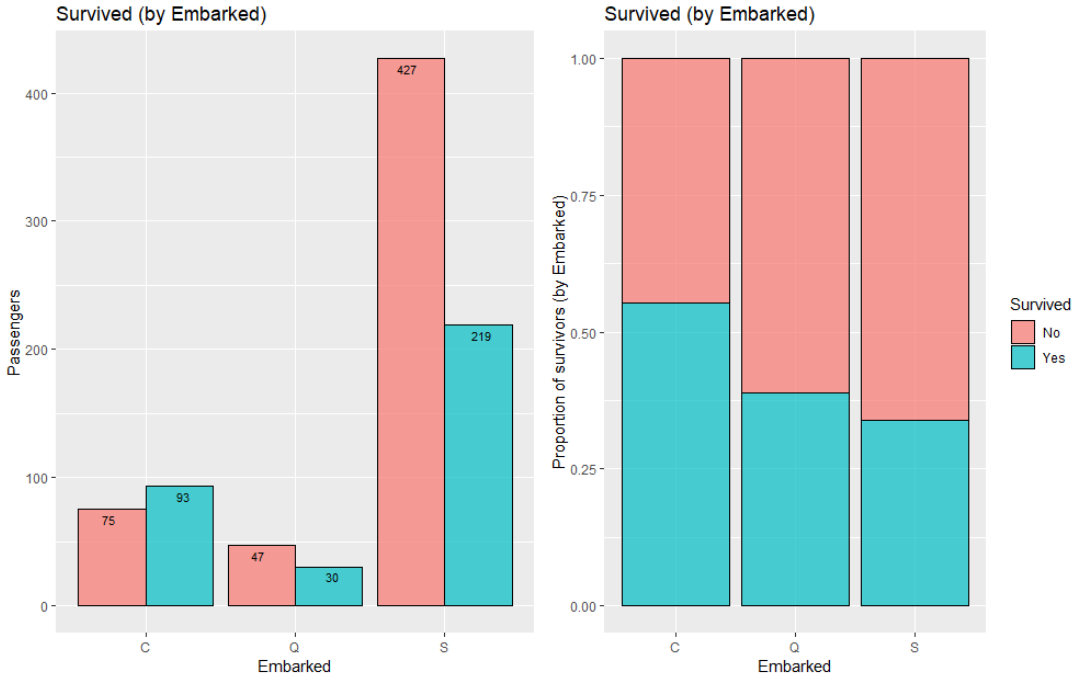
*(En este gráfico combinamos las variables "**Embarked**", "**Pclass**" y "**Survived**". En este caso en cuanto a la clase en la que se viaja, no parece que el puerto en el que se haya embarcado sea un factor decisivo a la hora de la supervivencia).*



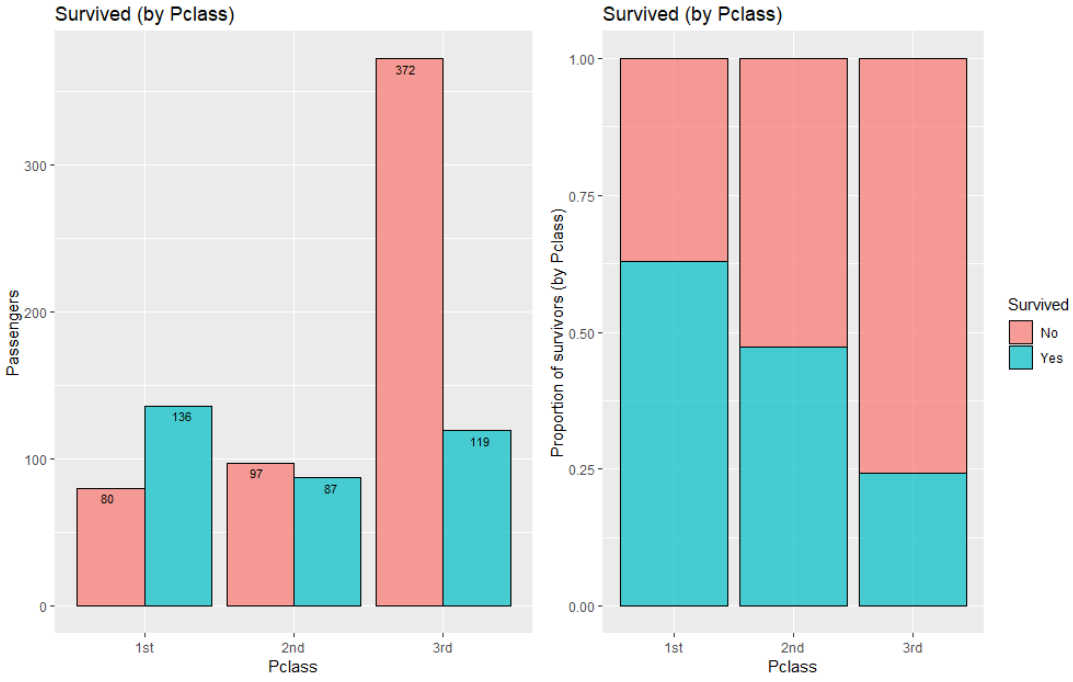
*(Ahora tenemos un gráfico muy similar pero esta vez las variables son "**Sex**", "**Pclass**" y "**Survived**". Aquí llama la atención entre las mujeres que en 1ª y 2ª clase hay apenas diferencias en cuanto a la mortalidad, pero sube al 50% en 3ª clase). Y en cambio en los hombres las diferencias son mínimas entre la 2ª y 3ª clase proporcionalmente).*



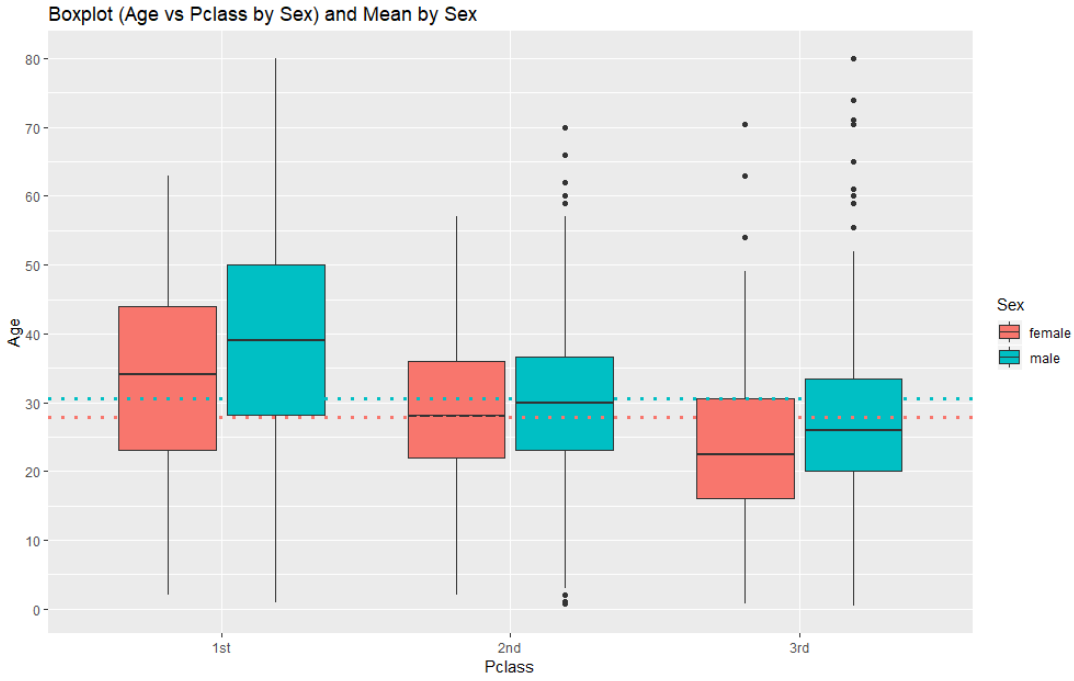
*(Antes, cuando veíamos la supervivencia de "**Embarked**" en función de "Pclass", no pareció que fuese muy determinante). Cuando vemos los datos solos, vemos que los puertos de embarque Q y S tienen aproximadamente la misma proporción de supervivencia, y en cambio el puerto C- Cherbourg tiene más de un 50% de porcentaje de supervivencia).*



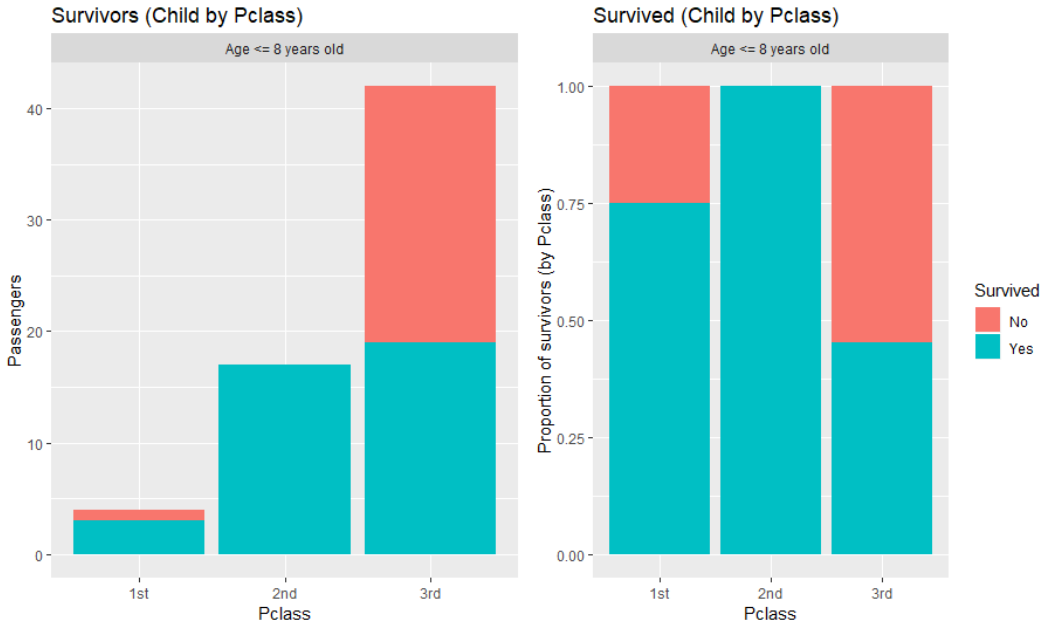
*(Y si hacemos lo mismo, pero esta vez con la variable "**Pclass**", vemos como claramente hay un mejor índice de supervivencia cuando se viaja en las clases más altas, parece ser un factor bastante diferencial).*



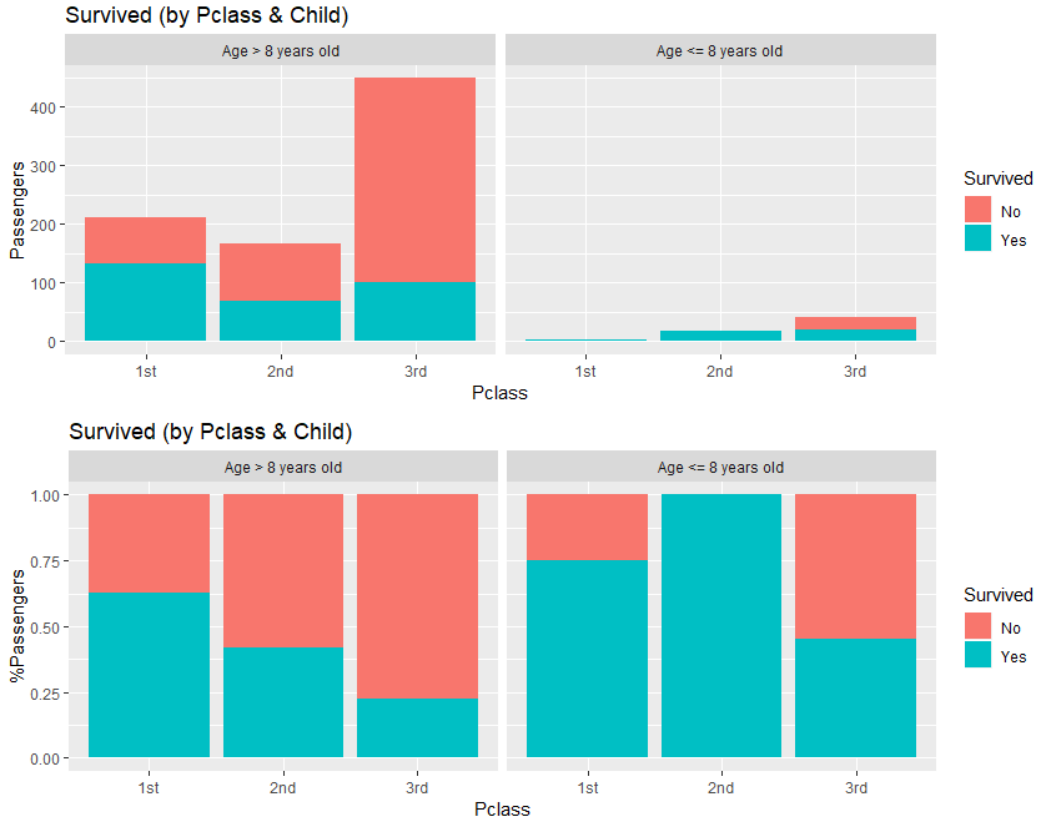
*(Ahora vemos un boxplot de la variable "**Age**" que la hemos separados en los grupos de "Pclass" y "Sex". Además hemos obtenido la media de edad de hombres y de mujeres, y así podemos comparar como se distribuyen los datos. Vemos que la gente que viaja en primera clase es de media mayor que la media de edad de los pasajeros en ambos sexos. Y en cambio en la 3ª clase la cosa es al revés, es decir son de media más jóvenes. Y vemos que la media por género de los pasajeros coincide casi exactamente con la media de los pasajeros de 2ª clase, tanto en mujeres como en hombres).*



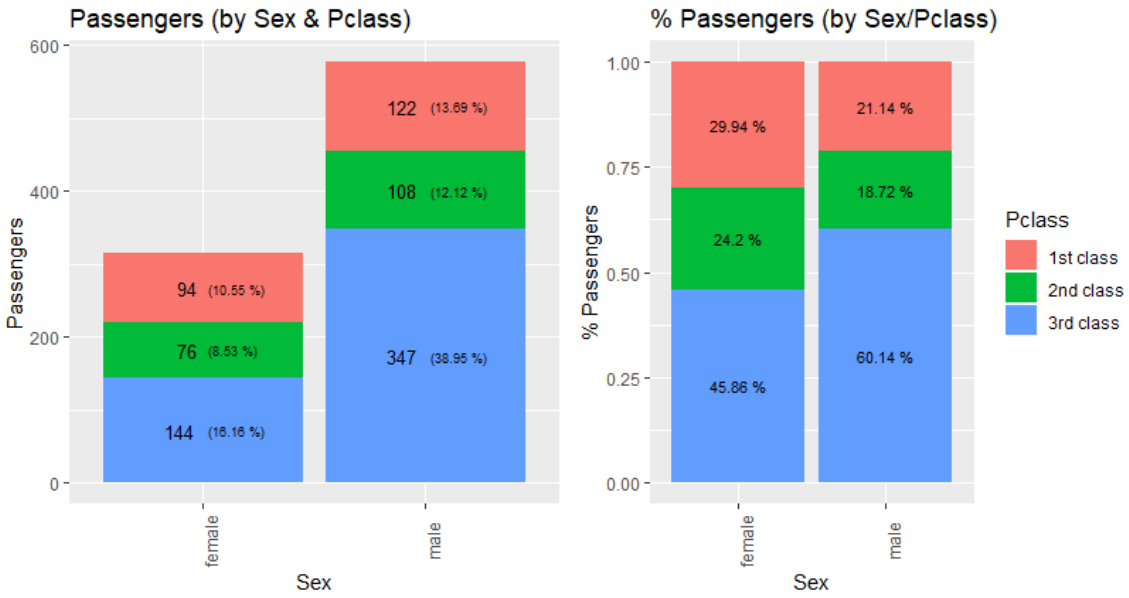
(Confirmamos que ser un niño era una ventaja a la hora de poder sobrevivir. Y sobre todo si viajas en 1ª o 2ª clase. Estos datos los podíamos intuir con aquel gráfico beeswarm que vimos anteriormente donde por edades se podían ver los supervivientes y fallecidos).



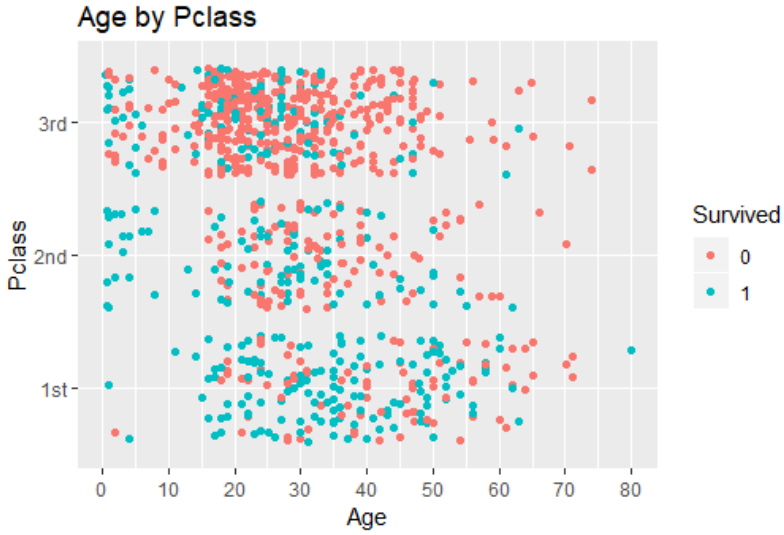
*(Y aquí en esta gráfica donde vemos los niños "**Child**" vs los que no lo son, y abajo sus respectivas proporciones, podemos observar que en todas las clases la proporción de supervivencia es mejor en el caso de los niños, que está a la derecha de la imagen).*

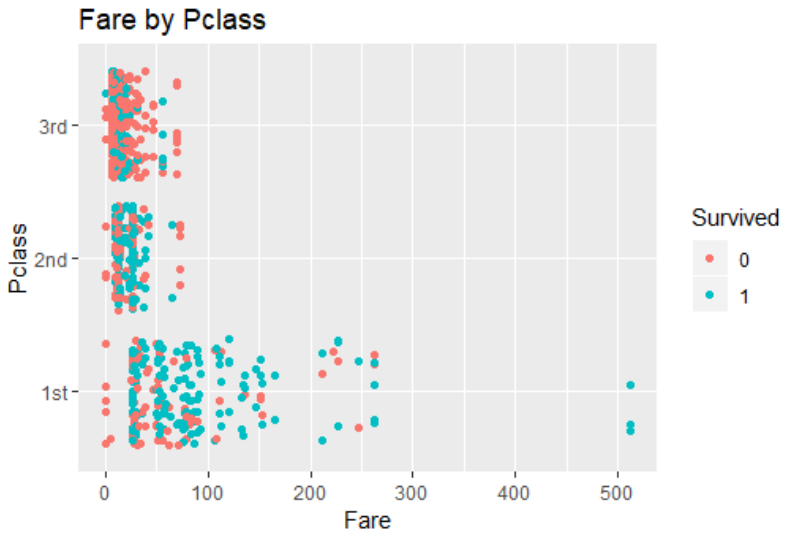


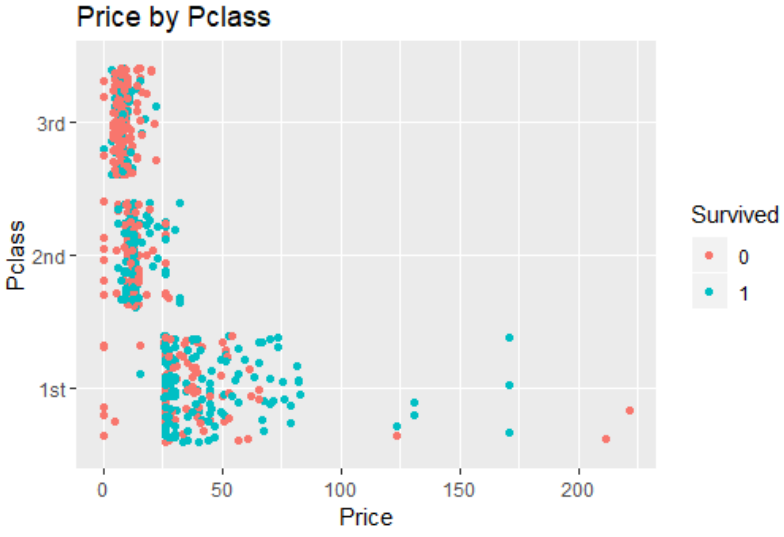
(Esta gráfica es cruza los datos entre "Sex" y "Pclass", tanto en número como en proporciones. Aquí podemos ver que la 3ª clase esta compuesta mayoritariamente por hombres, y que además esa 3ª clase tiene más hombres que en las otras 2 clases juntas. Las mujeres en cambio, casi una tercera parte viaja en 1ª clase).



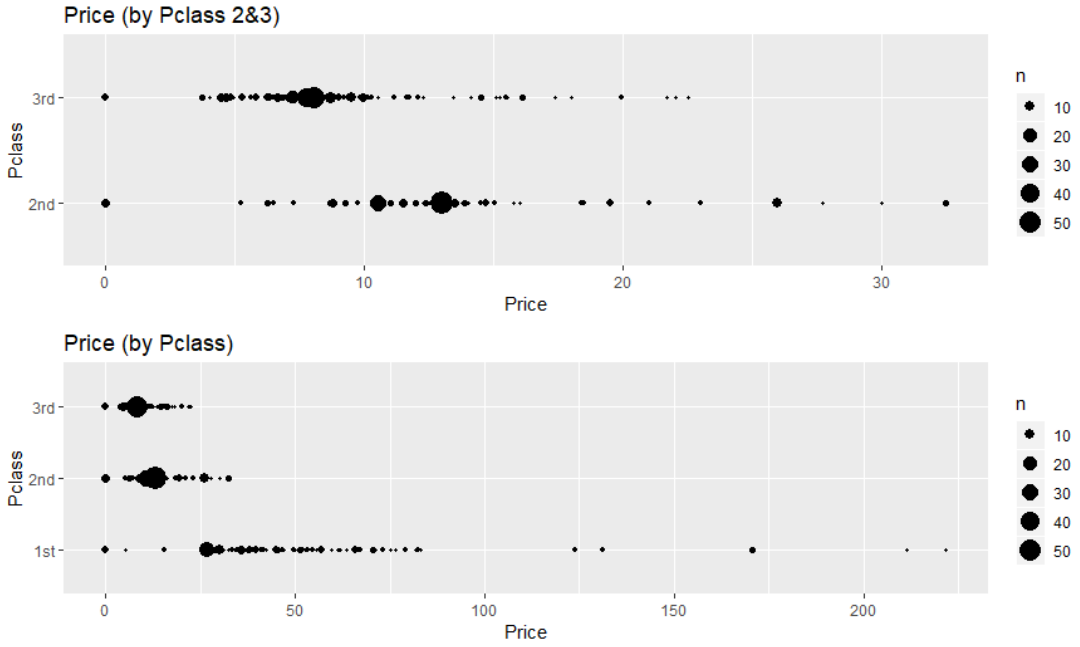
*(En estas gráficas siguientes vemos como se distribuyen en cuanto a "**Survived**" (a través de los colores) las muestras en las variables continuas "**Age**", "**Fare**" y "**Price**", separadas por los diferentes valores de la variable "**Pclass**").*



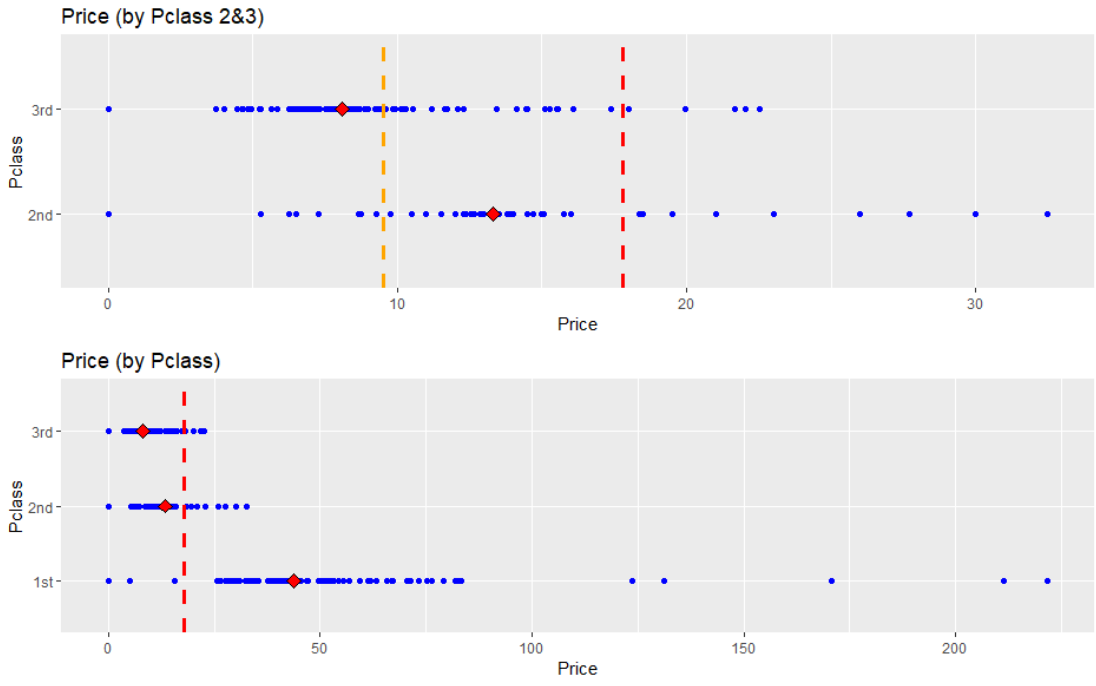




*(Aquí podemos ver como es el precio medio por pasajero "**Price**" en función de la clase "**Pclass**" en la que se viaja. En el gráfico de arriba solamente la 2ª y 3ª clase para que se distinga un poco mejor).*



*(En los siguientes gráficos también "**Price**" en función de la clase "**Pclass**", pero esta vez marcando donde está la media del precio por cada una de las clases (los puntos rojos) y luego las líneas verticales que suponen la media del precio de todos los pasajeros. En el primer gráfico solo hemos puesto la 2ª y 3ª clase para que se pueda distinguir. La línea amarilla indica la media entre ambas clases. Mientras que la línea roja puntuada es la media de todos los pasajeros. Ambas clases están alejadas del precio medio del billete).*



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

A lo largo de este proyecto se ha realizado un exhaustivo análisis del conjunto de datos dado: Comenzando por las fases de limpieza, depuración y análisis de datos (eliminando variables innecesarias, imputando datos faltantes, comprobando supuestos estadísticos…), hasta finalmente implementar un conjunto de modelos de predicción basados en variables dadas y otras creadas explícitamente a lo largo del proyecto.

Por tanto, a la pregunta de si es posible con el conjunto de datos del Titanic, predecir la supervivencia o no de un pasajero, responde cada uno de los modelos creados con resultados de precisión más bien discretos. Además, no hay ningún tipo de algoritmo que sea infalible, por tanto fuimos haciendo diferentes pruebas con modelos para ver cómo iban funcionando, cambiando algunos parámetros y también las variables utilizadas en la construcción de cada uno.

Destacaríamos tres de ellos con precisión de 83,78%: Dos regresiones logísticas usando datos enriquecidos con variables sintéticas, y un árbol de decisión con profundidad 5 para el que las variables más significativas resultaron ser las siguientes: "**Sexmale**", "**Pclass3**" y "**FamilySize**" (aunque también hemos visto que hay otras variables influyentes en los modelos, y que eran de esperar como "**Age**" y sus variables derivadas, como "**Child**" y "**AgeInterval**" y que suponen también una lógica). También hemos comprobado que la regresión logística con interacción de variables (modelo glm4) mejora la predicción de supervivientes (a costa de fallar algo más en los no supervivientes).

De hecho hemos comprobado que aunque la precisión total de ambos modelos fue equivalente, en el modelo de regresión logística ha fallado en que fallecen el 11,68% y el modelo predice que sobreviven (y sobreviven el 23.53% y el modelo predice lo contrario), mientras que en el árbol (tree2) se ha fallado en lo siguiente, fallecen realmente un 7,30% y el modelo predice que sobreviven (y sobreviven el 30,59% y el modelo predice que mueren). Nosotros consideramos que es preferible que se equivoque más en la predicción que no van a sobrevivir y finalmente sí lo hacen). Y esto parece tener sentido, ya que si observamos las 2 variables más importantes del árbol, que ha considerado como Sexmale (hombres) y Pclass3 (3ª clase) precisamente son las variables que suponen el mayor índice de mortalidad, es decir el modelo está utilizando principalmente esas, es decir tiene sentido que sea entonces capaz de predecir mejor la no supervivencia.

Finalmente hemos creado un fichero "**titanic_enriquecido.csv**" que contiene las 12 variables con las que hemos trabajado a lo largo de este proyecto, con la imputación de los datos realizada, con las variables no utilizadas del dataset origen eliminadas, y con las transformaciones que hemos consideradas necesarias.

A mayores, también hemos creado otro fichero al que hemos llamado "**titanic_enriquecido_predict.csv**", en el que además de las columnas que tiene el fichero anterior, tiene una columna nueva llamada "**Survived_Predicted**", que es la columna con la predicción de la variable "**Survived**" realizada con el modelo escogido.