Práctica 2: Limpieza y análisis de datos

Pedro Uceda Martínez, Pablo Campillo Sánchez

22 de diciembre, 2020

1. Descripción del dataset

Durante esta práctica vamos a tratar el *dataset* base de la competición **Titanic - Machine Learning from Disaster**. En este conjunto de datos se nos presenta, para cada pasajero del tan famoso trasatlántico, sus datos personales más importantes, así como otros relacionados con su embarque en el Titanic, y si finalmente sobrevivieron al naufragio del mismo.

De este modo, este estudio es interesante dado que examinaremos qué posibles factores pudieron influir en la supervivencia de los pasajeros. Así, podremos, por ejemplo, ver si solamente la clase del billete el género (mujeres) y la edad (niños) condicionaron que un viajero se salvase tal y como hemos visto en la gran pantalla o bien hubiera habido otros factores que pudieran haber determinado la supervivencia del pasajero, como el número de billete.

Las variables de las que disponemos, para cada pasajero, son:

- **PassengerId**: Identificador artifical del pasajero.
- **Survived**: Si sobrevivió (1) o no (0).
- **Pclass**: Clase del pasaje.
- **Name**: Nombre del pasajero.
- **sex**: Sexo del viajero.
- **Age**: Edad, en años.
- **SibSp**: Número de hermanos o esposas a bordo del Titanic
- **Parch**: Número de padres / hijos a bordo del Titanic
- **ticket**: Número de ticket
- **fare**: Tarifa del pasaje
- **cabin**: Número de camarote
- **embarked**: Puerto desde el que embarcó el pasajero. Las posibles opciones son: Cherbourg(C), Queenstown(Q) o Southampton(s).

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

2.1 Carga de los datos y selección

A continuación procedemos a cargar el **dataset**, sin **factors**, para evitar tratar los nombres de los pasajeros como tales.

```
ds <- read.csv(file = "train.csv", header=TRUE)

str(ds)
```

```
## 'data.frame':    891 obs. of  12 variables:
##  $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Survived   : int  0 1 1 1 0 0 0 0 1 1 ...
##  $ Pclass     : int  3 1 3 1 3 3 1 3 3 2 ...
```

```
##  $ Name       : chr  "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley (Florence Briggs Thayer)"
##  $ Sex        : chr  "male" "female" "female" "female" ...
##  $ Age        : num  22 38 26 35 35 NA 54 2 27 14 ...
##  $ SibSp      : int  1 1 0 1 0 0 0 3 0 1 ...
##  $ Parch      : int  0 0 0 0 0 0 0 1 2 0 ...
##  $ Ticket     : chr  "A/5 21171" "PC 17599" "STON/O2. 3101282" "113803" ...
##  $ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Cabin      : chr  "" "C85" "" "C123" ...
##  $ Embarked   : chr  "S" "C" "S" "S" ...
```

Los atributos PassengerId y Name no serán objeto de análisis. Para el resto, tenemos las variables cuantitativas Age, SibSp, Parch y Fare, todas correctamente tratadas como int o num.

También estan las variables cualitativas Ticket, PClass, Sex y Cabin, cargadas como cadena de caracteres. Nótese que Cabin es susceptible de ser dividida en letra y número.

2.2 Transformación de los datos

Para más claridad de los datos, procedemos a realizar las siguientes transformaciones: - Transformamos el campo dicotómico Survived a Yes(1) y Not(0). - Transformamos el campo cualitativo categórico Embarked a un factor con 3 posibles valores, cada uno con el nombre del puerto.

```
ds$Survived <- factor(ds$Survived, levels=sort(c(0,1)), labels = c("Not", "Yes"))
ds$Embarked <- factor(ds$Embarked, levels=sort(c("C", "Q", "S")), labels = c("Cherbourg", "Queenstown",



str(ds)
```

```
## 'data.frame':    891 obs. of  12 variables:
##  $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Survived   : Factor w/ 2 levels "Not","Yes": 1 2 2 2 1 1 1 1 2 2 ...
##  $ Pclass     : int  3 1 3 1 3 3 1 3 3 2 ...
##  $ Name       : chr  "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley (Florence Briggs Thayer)"
##  $ Sex        : chr  "male" "female" "female" "female" ...
##  $ Age        : num  22 38 26 35 35 NA 54 2 27 14 ...
##  $ SibSp      : int  1 1 0 1 0 0 0 3 0 1 ...
##  $ Parch      : int  0 0 0 0 0 0 0 1 2 0 ...
##  $ Ticket     : chr  "A/5 21171" "PC 17599" "STON/O2. 3101282" "113803" ...
##  $ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Cabin      : chr  "" "C85" "" "C123" ...
##  $ Embarked   : Factor w/ 3 levels "Cherbourg","Queenstown",..: 3 1 3 3 3 2 3 3 3 1 ...
```

A continuación analizamos cada uno de los distintos atributos:

```
summary(ds)
```

```
##   PassengerId    Survived      Pclass          Name
##  Min.   :  1.0   Not:549   Min.   :1.000   Length:891
##  1st Qu.:223.5   Yes:342   1st Qu.:2.000   Class :character
##  Median :446.0             Median :3.000   Mode  :character
##  Mean   :446.0             Mean   :2.309
##  3rd Qu.:668.5             3rd Qu.:3.000
##  Max.   :891.0             Max.   :3.000
##
##      Sex                 Age            SibSp           Parch
##  Length:891         Min.   : 0.42   Min.   :0.000   Min.   :0.0000
##  Class :character   1st Qu.:20.12   1st Qu.:0.000   1st Qu.:0.0000
```

```
##  Mode  :character   Median :28.00   Median :0.000   Median :0.0000
##                     Mean   :29.70   Mean   :0.523   Mean   :0.3816
##                     3rd Qu.:38.00   3rd Qu.:1.000   3rd Qu.:0.0000
##                     Max.   :80.00   Max.   :8.000   Max.   :6.0000
##                     NA's   :177
##     Ticket              Fare           Cabin                  Embarked
##  Length:891         Min.   :  0.00   Length:891         Cherbourg  :168
##  Class :character   1st Qu.:  7.91   Class :character   Queenstown : 77
##  Mode  :character   Median : 14.45   Mode  :character   Southampton:644
##                     Mean   : 32.20                      NA's       :  2
##                     3rd Qu.: 31.00
##                     Max.   :512.33
##
```

Vemos que los campos Age y Embarked tienen 177 y 2 valores nulos, respectivamente. Como no tiene sentido interpretarlos como 0 años o ningún puerto, sustituimos estos campos por la mediana para que afecten en la medida de lo posible al análisis.

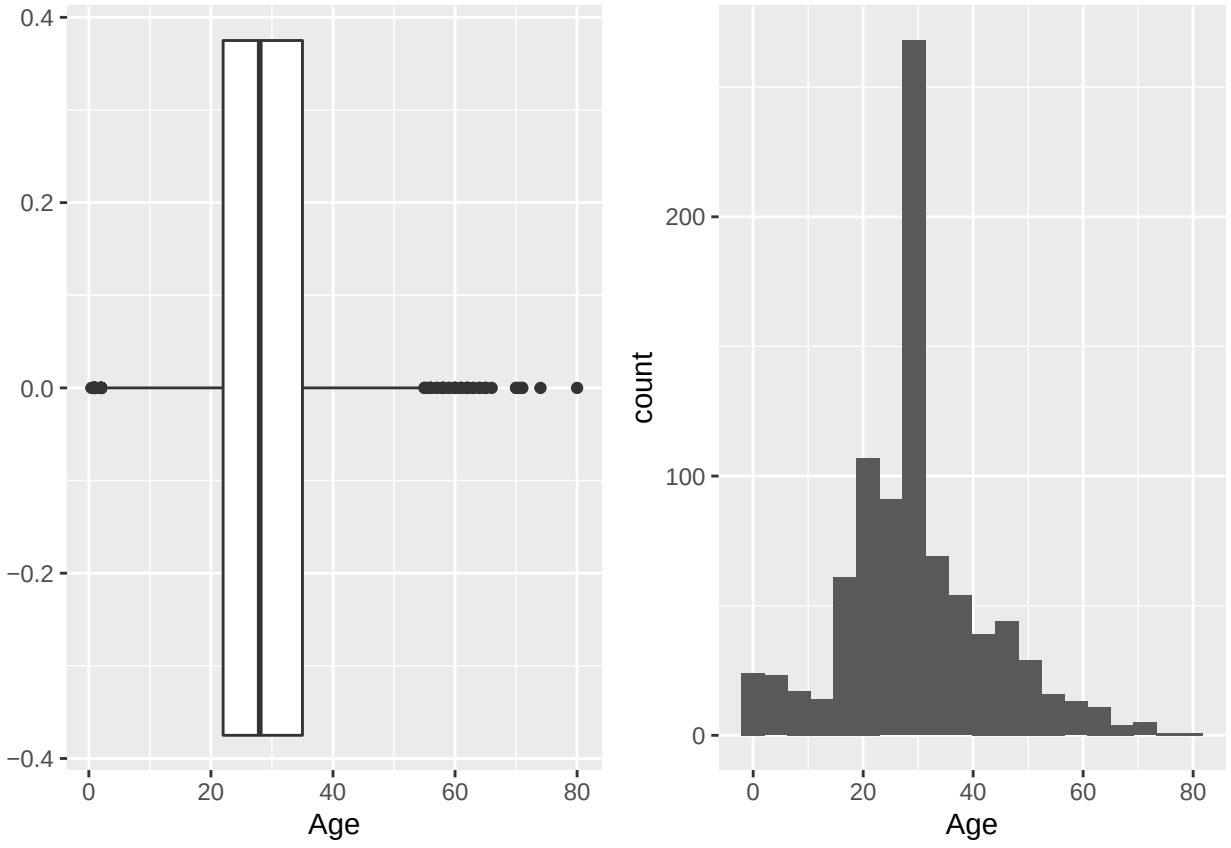
```
age_median <- median(ds$Age, na.rm = TRUE)

ds[, 'Age'][is.na(ds[,'Age'])] <- age_median

embarked_most_frequent <-  levels(ds$Embarked)[which.max(ds$Embarked)]

ds[, 'Embarked'][is.na(ds[,'Embarked'])] <- embarked_most_frequent

summary(ds)
```

```
##   PassengerId    Survived      Pclass          Name
##  Min.   :  1.0   Not:549   Min.   :1.000   Length:891
##  1st Qu.:223.5   Yes:342   1st Qu.:2.000   Class :character
##  Median :446.0             Median :3.000   Mode  :character
##  Mean   :446.0             Mean   :2.309
##  3rd Qu.:668.5             3rd Qu.:3.000
##  Max.   :891.0             Max.   :3.000
##      Sex                 Age            SibSp           Parch
##  Length:891         Min.   : 0.42   Min.   :0.000   Min.   :0.0000
##  Class :character   1st Qu.:22.00   1st Qu.:0.000   1st Qu.:0.0000
##  Mode  :character   Median :28.00   Median :0.000   Median :0.0000
##                     Mean   :29.36   Mean   :0.523   Mean   :0.3816
##                     3rd Qu.:35.00   3rd Qu.:1.000   3rd Qu.:0.0000
##                     Max.   :80.00   Max.   :8.000   Max.   :6.0000
##     Ticket              Fare           Cabin                  Embarked
##  Length:891         Min.   :  0.00   Length:891         Cherbourg  :170
##  Class :character   1st Qu.:  7.91   Class :character   Queenstown : 77
##  Mode  :character   Median : 14.45   Mode  :character   Southampton:644
##                     Mean   : 32.20
##                     3rd Qu.: 31.00
##                     Max.   :512.33
```

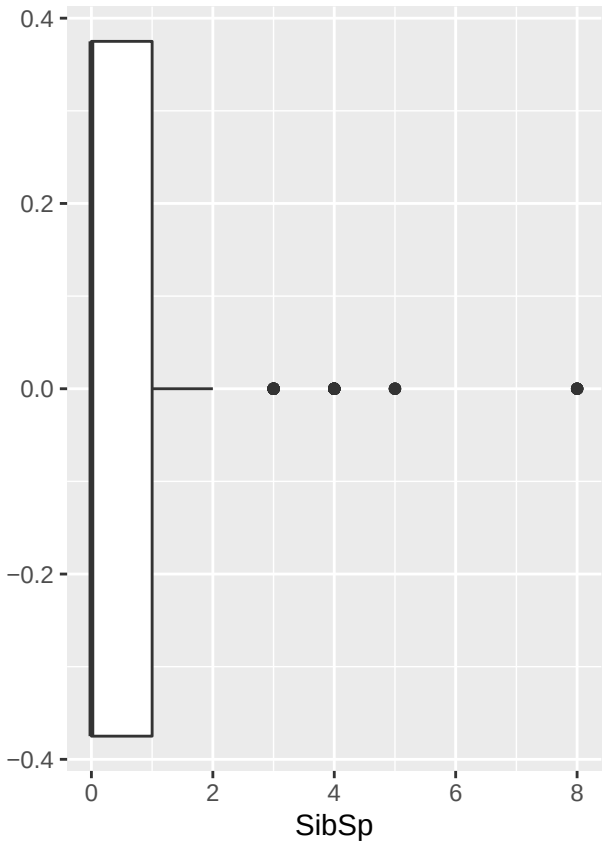
```
#Visualización de variables cuantitativas

#Age
gAge1 <- ggplot(ds, aes(x=Age)) + geom_boxplot()

gAge2 <- ggplot(ds, aes(x=Age)) + geom_histogram(bins=20)
```

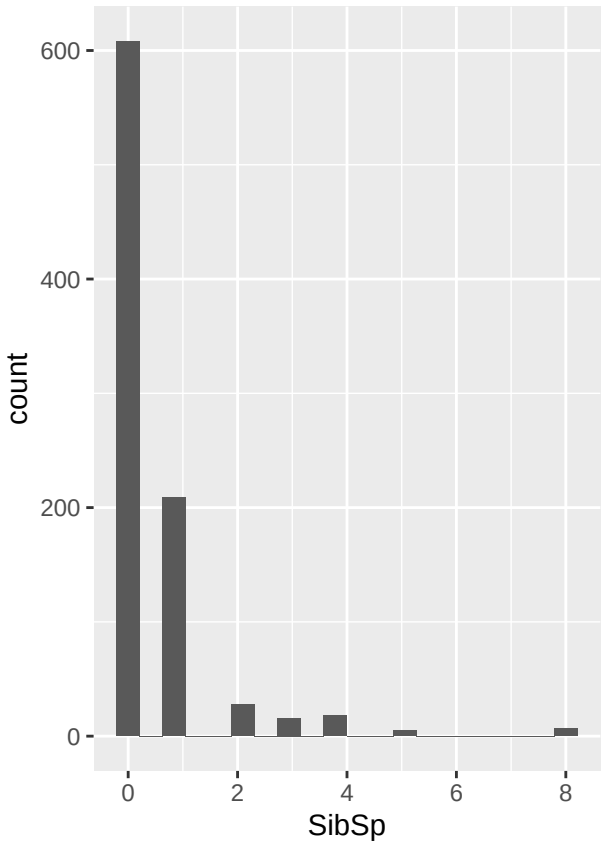
```r
#SibSp

gSibSp1 <- ggplot(ds, aes(x=SibSp)) + geom_boxplot()

gSibSp2 <- ggplot(ds, aes(x=SibSp)) + geom_histogram(bins=20)

#Parch

gParch1 <- ggplot(ds, aes(x=Parch)) + geom_boxplot()

gParch2 <- ggplot(ds, aes(x=Parch)) + geom_histogram(bins=20)


#Fare

gFare1 <- ggplot(ds, aes(x=Fare)) + geom_boxplot()

gFare2 <- ggplot(ds, aes(x=Fare)) + geom_histogram(bins=20)
```

```r
grid.arrange(gAge1,gAge2,nrow=1)
```

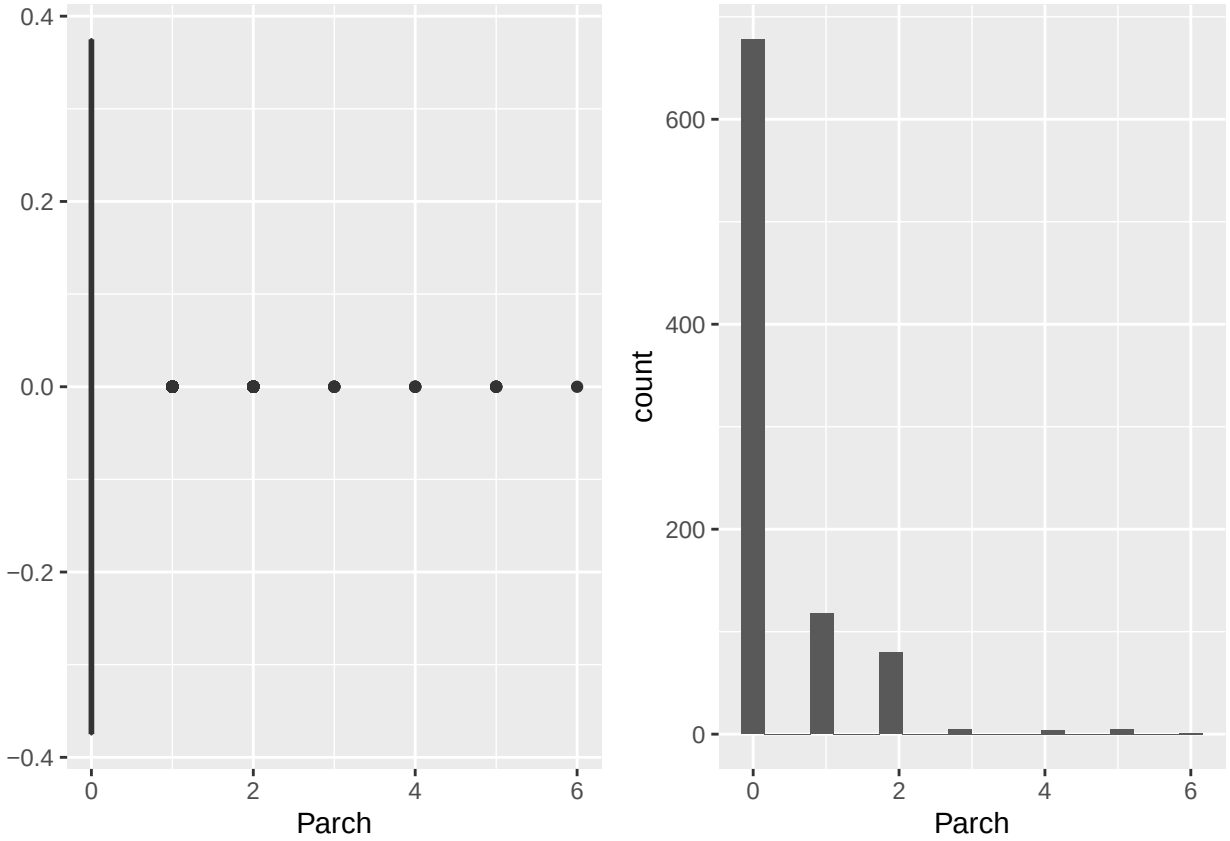



```r
grid.arrange(gSibSp1,gSibSp2,nrow=1)
```

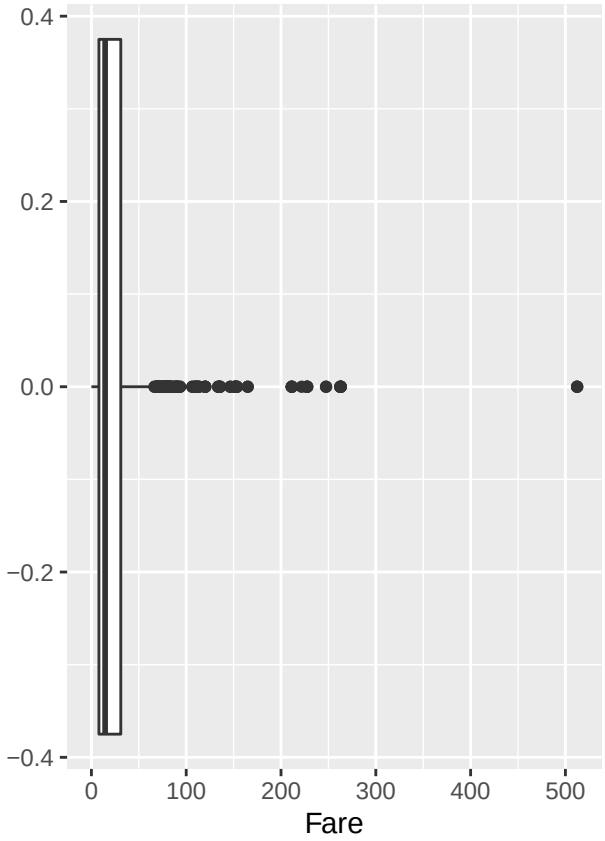


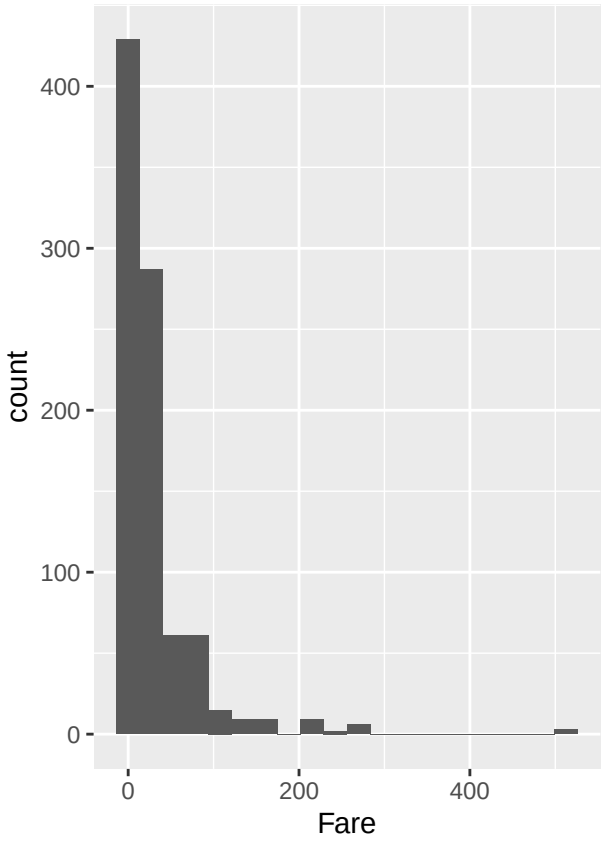



```
grid.arrange(gParch1,gParch2,nrow=1)
```




```
grid.arrange(gFare1,gFare2,nrow=1)
```






```
#Visualizacion de variables cuantitativas

#Survived
sumSurvived <- summarize( group_by(ds, Survived), n=length(Survived), Fare=mean(Fare))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
gSurvived1 <- ggplot( sumSurvived, aes(x="", y=n, fill=Survived)) +
geom_bar(width = 1, stat = "identity") +
coord_polar("y", start=0) + ggtitle("Survived")

#PClass and Survived
sumPClass <- summarize( group_by(ds, Pclass), n=length(Pclass), Survived=mean(Survived))
```

```
## Warning in mean.default(Survived): argument is not numeric or logical: returning
## NA

## Warning in mean.default(Survived): argument is not numeric or logical: returning
## NA

## Warning in mean.default(Survived): argument is not numeric or logical: returning
## NA

## `summarise()` ungrouping output (override with `.groups` argument)
```

```
gPClass1 <- ggplot( sumPClass, aes(x="", y=n, fill=Pclass)) +
geom_bar(width = 1, stat = "identity") +
coord_polar("y", start=0) + ggtitle("PClass")
```

```
gPClass2 <- ds %>%
  group_by(Survived, Pclass) %>%
  tally() %>%
  group_by(Survived) %>%
  mutate(x = n / sum(n)) %>%
  ggplot() +
    geom_col(aes(
      x = factor(Survived),
      y = x,
      fill = factor(Pclass)
      ), position = "stack")

#Sex and Survived
sumSex <- summarize( group_by(ds, Sex), n=length(Sex), Survived=mean(Survived))
```

```
## Warning in mean.default(Survived): argument is not numeric or logical: returning
## NA

## Warning in mean.default(Survived): argument is not numeric or logical: returning
## NA

## `summarise()` ungrouping output (override with `.groups` argument)
```

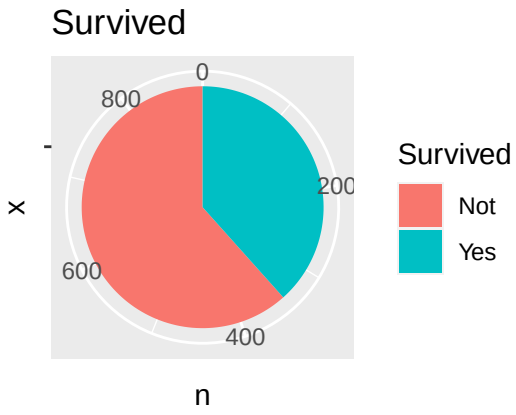
```
gSex1 <- ggplot( sumSex, aes(x="", y=n, fill=Sex)) +
geom_bar(width = 1, stat = "identity") +
coord_polar("y", start=0) + ggtitle("Sex")

gSex2 <- ds %>%
  group_by(Survived, Sex) %>%
  tally() %>%
  group_by(Survived) %>%
  mutate(x = n / sum(n)) %>%
  ggplot() +
    geom_col(aes(
      x = factor(Survived),
      y = x,
      fill = factor(Sex)
      ), position = "stack")

#Embarked and Survived
sumEmbarked <- summarize( group_by(ds, Embarked), n=length(Embarked))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

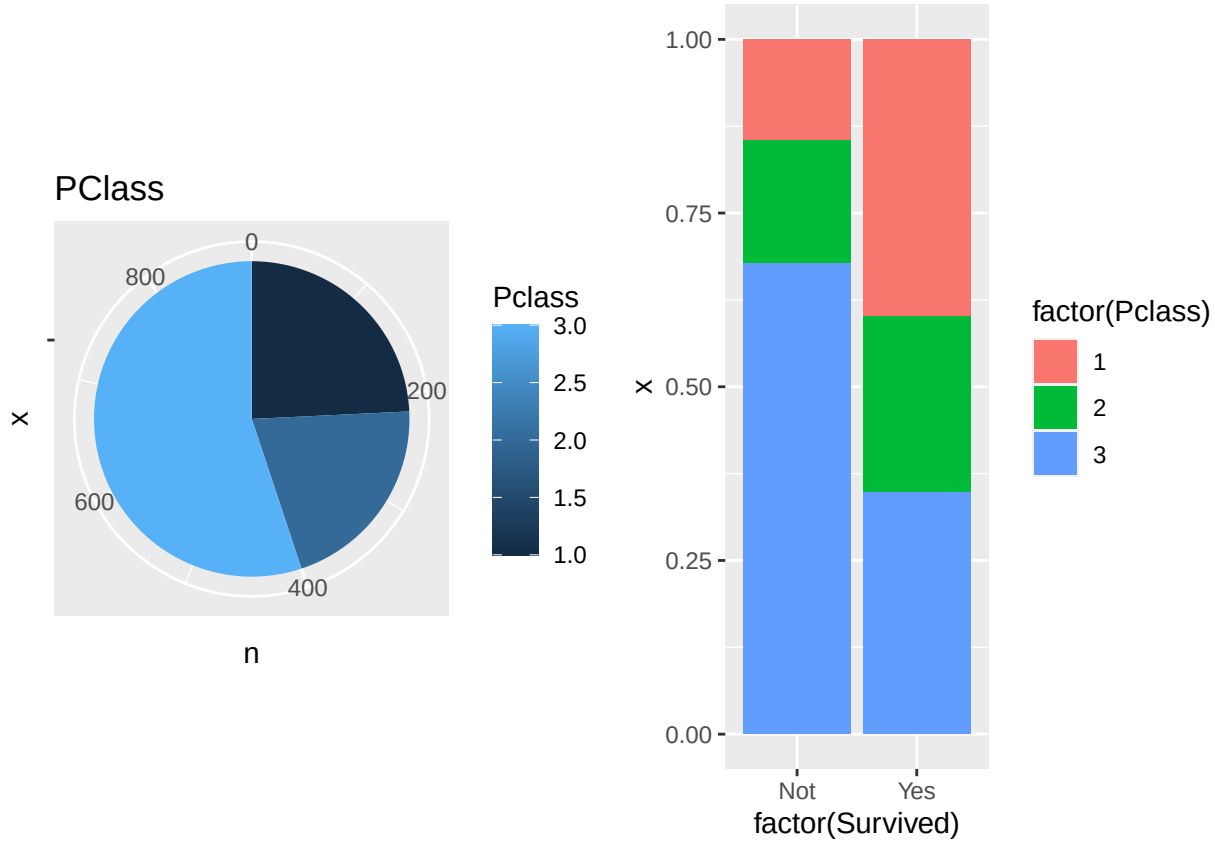
```
gEmbarked1 <- ggplot( sumEmbarked, aes(x="", y=n, fill=Embarked)) +
geom_bar(width = 1, stat = "identity") +
coord_polar("y", start=0) + ggtitle("Embarked")


gEmbarked2 <- ds %>%
  group_by(Survived, Embarked) %>%
  tally() %>%
  group_by(Survived) %>%
  mutate(x = n / sum(n)) %>%
  ggplot() +
```

```
    geom_col(aes(
      x = factor(Embarked),
      y = x,
      fill = factor(Survived)
      ), position = "stack")
```

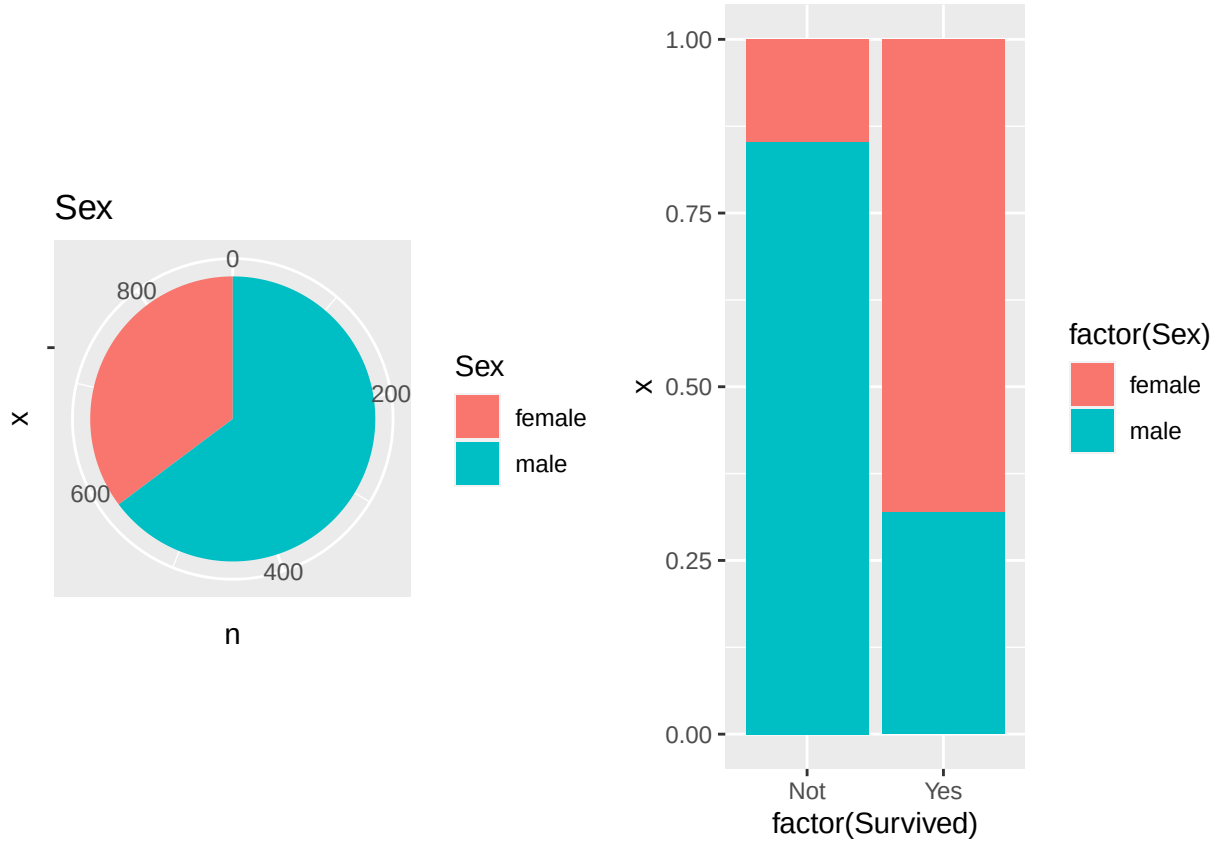
```
grid.arrange(gSurvived1, nrow=2)
```



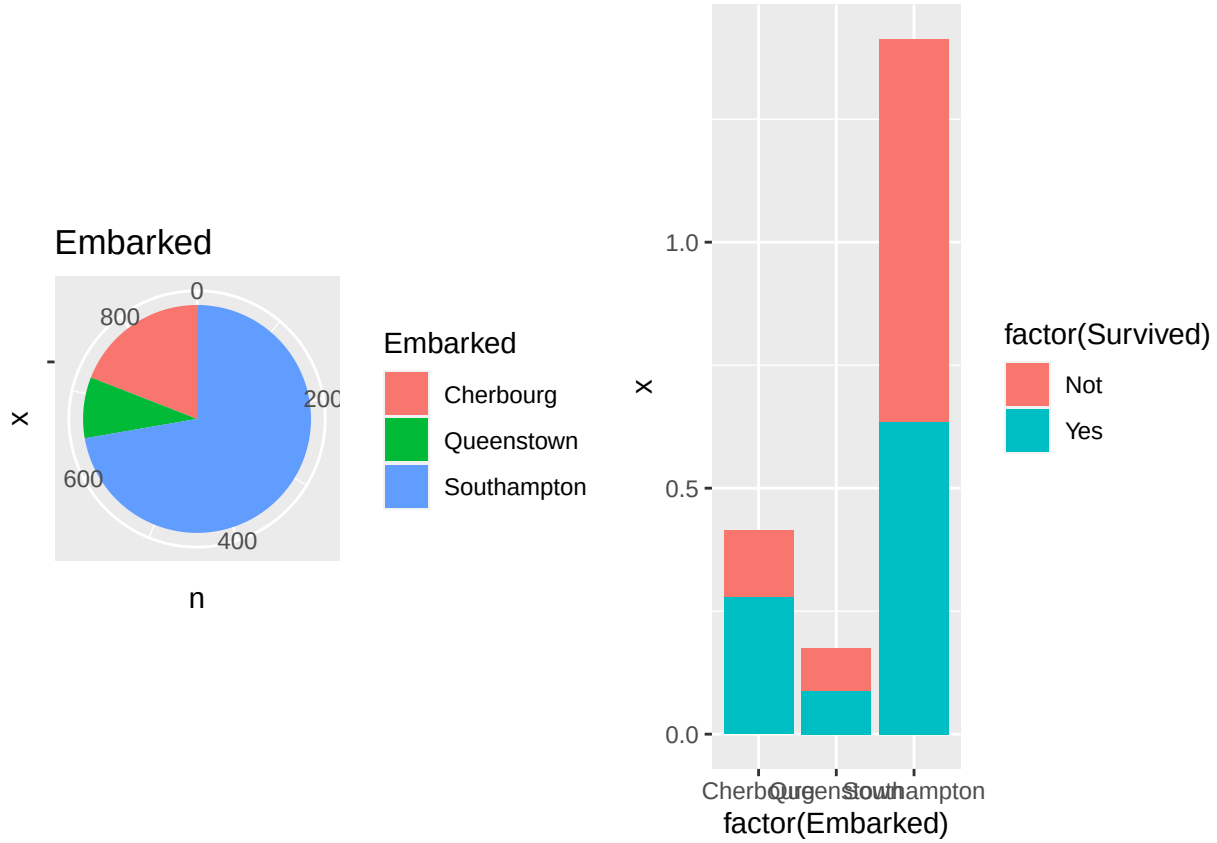

```
grid.arrange(gPClass1,gPClass2, nrow=1)
```




```
grid.arrange(gSex1, gSex2, nrow=1)
```




```
grid.arrange(gEmbarked1, gEmbarked2, nrow=1)
```



#2.3 Descipción estadística descriptiva

TODO: Describir cómo se distribuyen los datos y como podría saltar a la vista correlaciones. Da idea del ejercicio 4.

#3. Limpieza de datos

3.1 Elementos vacíos

TODO: En el ejercicio 1 se ha pintado el campo Age y el campo Embarked ya sin elementos vacíos. Traer aqui y pintar de nuevo, con un summary para demostrar que han desaparecido.

3.2 Identificación y tratamiento de valores extremos.