Tipologia i cicle de vida de les dades

Pràctica 2: Neteja i anàlisi de les dades

Autors: Adrià Tarradas i Aleix Arnau Soler

Desembre 2020

Índice

1. Presentació del projecte de ciència de dades **2**
1.1. Background . . . 2
1.2. Objectiu . . . 2
1.3. Descripció del dataset . . . 2

2. Integració i comprobació **4**

3. Neteja de dades **9**
3.1. Gestió de valors buits o nul . . . 9
3.2. Creació de variables . . . 16
3.3. Gestió de valors extrems (outliers) . . . 20
4. Exploració i anàlisis de variables . . . 25
4.1. Comprobació de la normalitat variables quantitatives . . . 26
4.3. Comprovació de l'homoscedasticitat . . . 31
4.2. Distribucions de les variables entre passatgers supervivents i no supervivents . . . 33
4.3. Comparació entre dos grups de dades (variables continues) . . . 34
4.4. Comparació entre dos grups de dades (variables categoriques) . . . 34
4.6. Correlació entre variables . . . 36

5. Modelització **38**
5.1. Arbre de decisió . . . 38
5.2. Support Vector Machine . . . 42
5.3. Regressió logística . . . 44

6. Predicció **47**

7. Conclusions **48**

8. Bibliografia **49**

1. Presentació del projecte de ciència de dades

1.1. Background

L'enfonsament del Titanic és un dels naufragis més famosos de la història. El 15 d'abril de 1912, durant el seu viatge inaugural, el considerat "inenfonsable" RMS Titanic es va enfonsar després de xocar amb un iceberg. Malauradament, no hi havia prou bots salvavides per a tothom a bord, cosa que va provocar la mort de 1502 de 2224 passatgers i tripulants. Tot i que hi va haver algun element de sort per sobreviure, sembla ser que alguns grups de persones tenien més probabilitats de sobreviure que d'altres.

1.2. Objectiu

L'objectiu d'aquesta pràctica és la creació final d'un o varis models predictius que responguin a la pregunta: **"quin tipus de persones tenien més probabilitats de sobreviure?"**. Per això, caldrà primer netejar i analitzar les dades dels passatgers a bord del RMS Titanic (*i.e.* nom, edat, sexe, classe socioeconòmica, etc.) abans de construir els models per a predir quins passatgers van sobreviure a l'enfonsament del Titanic.

1.3. Descripció del dataset

El dataset que utilitzarem en aquesta pràctica és *Titanic: Machine Learning from Disaster*, disponible clicant a *aquí*.

Aquest conjunt de dades està compost per diversos atributs/característiques dels passatgers del titanic distribuïts en 2 fitxers: un dataset per entrenar els models i un altre per a testar-los. A més, s'incou un fitxer amb la predicció de la supervivencia dels passatgers (codificat com a 0: mort o 1: sobreviu) que assumeix que totes les dones, i només les dones, sobreviuen. Obviament, aquestes prediccions són irreals pero ens serveixen com a exemple de com hauria de ser el fitxer final amb les prediccions de supervivencia en cas de que es participés a una de les competició de *Kaggle*. És per aquest motiu que les dades es proporcionen ja semparades en un dataset d'entrenament i un dataset de testeig, ja que així tots els membres que participen a la cometició de la web Kaggle parteixen de la mateixa informació i dades, i per tant es poden comparar entre ells els resultats obtinguts amb els diferents models implentats.

Els fitxers que componen el dataset són:

- **train.csv**: Conté totes les dades i variables d'un subgrup dels passatgers a bord del RMS Titanic (891 passatgers i tripulants), a més de la variable que ens indica si aquell passatger va morir o sobreviure. Aquest dataset serà el que s'utilitzarà per l'entrenament d'un model.
- **test.csv**: Conté totes les dades i variables d'un subgrup més petit que l'anterior dels passatgers a bord del RMS Titanic (en aquest cas 418 passatgers i tripulants) amb l'excepció de que aquest dataset no conté la variable que ens indica si el passatger va sobreviure o no. Aquestes dades s'utilitzaran per a poder testejar els models creats amb les dades del dataset d'entrenament.
- **gender_submission.csv**: Conté la classe dels passatgers (si sobreviuen o no) vinculada a l'identificador de cada passatger, assumint que totes les dones, i només les dones, haguessin sobreviscut.

Cada passatger conté informació de les següents variables:

Taula 1: Data diccionary: resum de les variables del dataset 'Titanic: Machine Learning from Disaster'.

	Variables	Definició	Codificació
	PassangerID	Número identificador del passatger	
	Survived	Enter que indica si el passatger va sobreviure l'enfonsament o no	0 = No, 1 = Sí
	Pclass	Enter que indica el tipus de tiquet del passatger	1 = 1a classe, 2 = 2a classe, 3 = 3a classe
	Name	Títol/Nom del passatger	
	Sex	Sexe del passatger	
	Age	Edat del passatger (anys)	
	SibSp	Número de cònjuges i germans del passatger a bord	
	Parch	Número de pares i fills del passatger a bord	
	Ticket	Número del tiquet del passatger	
	Fare	Preu pagat/Tarifa del viatge	
	Cabin	Número de cabina	
	Embarked	Port en el que ha embarcat el passatger	C = Cherbourg, Q = Queenstown, S = Southampton

Notes a tenir en compte: * La variable **pclass** ens serveix com a indicador indirecte del nivell socioeconomic del passatger (alt: 1a classe, mitjà: 2a classe, baix: 3a classe). * L'edat dels menors d'1 any està codificada com la fracció d'any que tenen. Les edats és mostren com a fraccions on en alguns casos s'indica' el 'i mig' (e.g. 35.50 = 35 anys i mig) * Alguns nens viatjaven amb una mainadera, per tant és possible que per algun d'ells 'parch' sigui 0.

2. Integració i comprobació

Un cop les dades s'han descarregat des de l'enllaç proporcionat, carreguem les dades a l'entorn de treball.

```
train_raw <- read.csv("data/train.csv", sep=',', stringsAsFactors = FALSE)
test_raw <- read.csv("data/test.csv", sep=',', stringsAsFactors = FALSE)
test_class_raw <- read.csv("data/gender_submission.csv", sep=',', stringsAsFactors = FALSE)
```

Primer inspeccionem els datasets

```
head(train_raw)
```

```
##   PassengerId Survived Pclass
## 1           1        0      3
## 2           2        1      1
## 3           3        1      3
## 4           4        1      1
## 5           5        0      3
## 6           6        0      3
##                                                 Name    Sex Age SibSp Parch
## 1                            Braund, Mr. Owen Harris   male  22     1     0
## 2 Cumings, Mrs. John Bradley (Florence Briggs Thayer) female  38     1     0
## 3                             Heikkinen, Miss. Laina female  26     0     0
## 4       Futrelle, Mrs. Jacques Heath (Lily May Peel) female  35     1     0
## 5                           Allen, Mr. William Henry   male  35     0     0
## 6                                   Moran, Mr. James   male  NA     0     0
##             Ticket    Fare Cabin Embarked
## 1        A/5 21171  7.2500              S
## 2         PC 17599 71.2833   C85        C
## 3 STON/O2. 3101282  7.9250              S
## 4           113803 53.1000  C123        S
## 5           373450  8.0500              S
## 6           330877  8.4583              Q
```

```
head(test_raw)
```

```
##   PassengerId Pclass                                          Name    Sex  Age
## 1         892      3                              Kelly, Mr. James   male 34.5
## 2         893      3              Wilkes, Mrs. James (Ellen Needs) female 47.0
## 3         894      2                     Myles, Mr. Thomas Francis   male 62.0
## 4         895      3                              Wirz, Mr. Albert   male 27.0
## 5         896      3 Hirvonen, Mrs. Alexander (Helga E Lindqvist) female 22.0
## 6         897      3                    Svensson, Mr. Johan Cervin   male 14.0
##   SibSp Parch  Ticket    Fare Cabin Embarked
## 1     0     0  330911  7.8292              Q
## 2     1     0  363272  7.0000              S
## 3     0     0  240276  9.6875              Q
## 4     0     0  315154  8.6625              S
## 5     1     1 3101298 12.2875              S
## 6     0     0    7538  9.2250              S
```

```
head(test_class_raw)
```

```
##   PassengerId Survived
## 1         892        0
## 2         893        1
## 3         894        0
```

```
## 4          895        0
## 5          896        1
## 6          897        0
```

Primer de tot extreurem la variable 'Survived' del dataset d'entrenament en un format igual al dataset 'gender_submission.csv' ja que aquesta no ens farà falta de moment i així podem integrar els dos datasets en un de sol per a dur a terme la neteja, inspecció i analisis en un sol dataset.

```
train_predictions<-train_raw[,c("PassengerId","Survived")]
train_raw<-train_raw[,-2]
```

Comprovem que les dades s'han carregat correctamen i les inspeccionem. Dataset d'entrenament:

```
# Comprovem que les dimensions del dataset siguin les esperades
dim(train_raw)
```

```
## [1] 891  11
```

```
# Comprovem en quina classe es troben les variables
sapply(train_raw,class)
```

```
## PassengerId      Pclass        Name         Sex         Age       SibSp
##   "integer"   "integer" "character" "character"   "numeric"   "integer"
##       Parch      Ticket        Fare       Cabin    Embarked
##   "integer" "character"   "numeric" "character" "character"
```

```
# Comprovem l'estructura
str(train_raw)
```

```
## 'data.frame':    891 obs. of  11 variables:
##  $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ 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" ...
```

```
# Inspeccionem alguns parametres bàsics
summary(train_raw)
```

```
##   PassengerId        Pclass          Name              Sex
##  Min.   :  1.0   Min.   :1.000   Length:891         Length:891
##  1st Qu.:223.5   1st Qu.:2.000   Class :character   Class :character
##  Median :446.0   Median :3.000   Mode  :character   Mode  :character
##  Mean   :446.0   Mean   :2.309
##  3rd Qu.:668.5   3rd Qu.:3.000
##  Max.   :891.0   Max.   :3.000
##
##       Age            SibSp           Parch           Ticket
##  Min.   : 0.42   Min.   :0.000   Min.   :0.0000   Length:891
##  1st Qu.:20.12   1st Qu.:0.000   1st Qu.:0.0000   Class :character
##  Median :28.00   Median :0.000   Median :0.0000   Mode  :character
##  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
##  Min.   :  0.00   Length:891         Length:891
##  1st Qu.:  7.91   Class :character   Class :character
##  Median : 14.45   Mode  :character   Mode  :character
##  Mean   : 32.20
##  3rd Qu.: 31.00
##  Max.   :512.33
##
```

```
# Inspeccionem les dades
glimpse(train_raw)
```

```
## Rows: 891
## Columns: 11
## $ PassengerId <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, ...
## $ Pclass      <int> 3, 1, 3, 1, 3, 3, 1, 3, 3, 2, 3, 1, 3, 3, 3, 2, 3, 2, 3...
## $ Name        <chr> "Braund, Mr. Owen Harris", "Cumings, Mrs. John Bradley ...
## $ Sex         <chr> "male", "female", "female", "female", "male", "male", "...
## $ Age         <dbl> 22, 38, 26, 35, 35, NA, 54, 2, 27, 14, 4, 58, 20, 39, 1...
## $ SibSp       <int> 1, 1, 0, 1, 0, 0, 0, 3, 0, 1, 1, 0, 0, 1, 0, 0, 4, 0, 1...
## $ Parch       <int> 0, 0, 0, 0, 0, 0, 0, 1, 2, 0, 1, 0, 0, 5, 0, 0, 1, 0, 0...
## $ Ticket      <chr> "A/5 21171", "PC 17599", "STON/O2. 3101282", "113803", ...
## $ Fare        <dbl> 7.2500, 71.2833, 7.9250, 53.1000, 8.0500, 8.4583, 51.86...
## $ Cabin       <chr> "", "C85", "", "C123", "", "", "E46", "", "", "", "G6",...
## $ Embarked    <chr> "S", "C", "S", "S", "S", "Q", "S", "S", "S", "C", "S", ...
```

Dataset de testeig:

```
# Comprovem que les dimensions del dataset siguin les esperades
dim(test_raw)
```

```
## [1] 418  11
```

```
# Comprovem en quina classe es troben les variables
sapply(test_raw,class)
```

```
## PassengerId      Pclass        Name         Sex         Age       SibSp
##   "integer"   "integer" "character" "character"   "numeric"   "integer"
##       Parch      Ticket        Fare       Cabin    Embarked
##   "integer" "character"   "numeric" "character" "character"
```

```
# Comprovem l'estructura
str(test_raw)
```

```
## 'data.frame':    418 obs. of  11 variables:
##  $ PassengerId: int  892 893 894 895 896 897 898 899 900 901 ...
##  $ Pclass     : int  3 3 2 3 3 3 3 2 3 3 ...
##  $ Name       : chr  "Kelly, Mr. James" "Wilkes, Mrs. James (Ellen Needs)" "Myles, Mr. Thomas Francis
##  $ Sex        : chr  "male" "female" "male" "male" ...
##  $ Age        : num  34.5 47 62 27 22 14 30 26 18 21 ...
##  $ SibSp      : int  0 1 0 0 1 0 0 1 0 2 ...
##  $ Parch      : int  0 0 0 0 1 0 0 1 0 0 ...
##  $ Ticket     : chr  "330911" "363272" "240276" "315154" ...
##  $ Fare       : num  7.83 7 9.69 8.66 12.29 ...
##  $ Cabin      : chr  "" "" "" "" ...
```

```
##  $ Embarked   : chr  "Q" "S" "Q" "S" ...
```

```
# Inspeccionem alguns parametres bàsics
summary(test_raw)
```

```
##   PassengerId         Pclass          Name               Sex
##  Min.   : 892.0   Min.   :1.000   Length:418         Length:418
##  1st Qu.: 996.2   1st Qu.:1.000   Class :character   Class :character
##  Median :1100.5   Median :3.000   Mode  :character   Mode  :character
##  Mean   :1100.5   Mean   :2.266
##  3rd Qu.:1204.8   3rd Qu.:3.000
##  Max.   :1309.0   Max.   :3.000
##
##       Age            SibSp            Parch           Ticket
##  Min.   : 0.17   Min.   :0.0000   Min.   :0.0000   Length:418
##  1st Qu.:21.00   1st Qu.:0.0000   1st Qu.:0.0000   Class :character
##  Median :27.00   Median :0.0000   Median :0.0000   Mode  :character
##  Mean   :30.27   Mean   :0.4474   Mean   :0.3923
##  3rd Qu.:39.00   3rd Qu.:1.0000   3rd Qu.:0.0000
##  Max.   :76.00   Max.   :8.0000   Max.   :9.0000
##  NA's   :86
##       Fare            Cabin             Embarked
##  Min.   :  0.000   Length:418         Length:418
##  1st Qu.:  7.896   Class :character   Class :character
##  Median : 14.454   Mode  :character   Mode  :character
##  Mean   : 35.627
##  3rd Qu.: 31.500
##  Max.   :512.329
##  NA's   :1
```

```
# Inspeccionem les dades
glimpse(test_raw)
```

```
## Rows: 418
## Columns: 11
## $ PassengerId <int> 892, 893, 894, 895, 896, 897, 898, 899, 900, 901, 902, ...
## $ Pclass      <int> 3, 3, 2, 3, 3, 3, 3, 2, 3, 3, 3, 1, 1, 2, 1, 2, 2, 3, 3...
## $ Name        <chr> "Kelly, Mr. James", "Wilkes, Mrs. James (Ellen Needs)",...
## $ Sex         <chr> "male", "female", "male", "male", "female", "male", "fe...
## $ Age         <dbl> 34.5, 47.0, 62.0, 27.0, 22.0, 14.0, 30.0, 26.0, 18.0, 2...
## $ SibSp       <int> 0, 1, 0, 0, 1, 0, 0, 1, 0, 2, 0, 0, 1, 1, 1, 1, 0, 0, 1...
## $ Parch       <int> 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ Ticket      <chr> "330911", "363272", "240276", "315154", "3101298", "753...
## $ Fare        <dbl> 7.8292, 7.0000, 9.6875, 8.6625, 12.2875, 9.2250, 7.6292...
## $ Cabin       <chr> "", "", "", "", "", "", "", "", "", "", "", "", "B45", ...
## $ Embarked    <chr> "Q", "S", "Q", "S", "S", "S", "Q", "S", "C", "S", "S", ...
```

Les dimensions són les esperades i el format és equivalent entre els dos datasets.

```
head(train_raw)
```

```
##   PassengerId Pclass                                                Name    Sex
## 1           1      3                             Braund, Mr. Owen Harris   male
## 2           2      1 Cumings, Mrs. John Bradley (Florence Briggs Thayer) female
## 3           3      3                              Heikkinen, Miss. Laina female
## 4           4      1        Futrelle, Mrs. Jacques Heath (Lily May Peel) female
## 5           5      3                            Allen, Mr. William Henry   male
```

```
## 6           6      3                                     Moran, Mr. James   male
##   Age SibSp Parch           Ticket    Fare Cabin Embarked
## 1  22     1     0        A/5 21171  7.2500              S
## 2  38     1     0         PC 17599 71.2833   C85        C
## 3  26     0     0 STON/O2. 3101282  7.9250              S
## 4  35     1     0           113803 53.1000  C123        S
## 5  35     0     0           373450  8.0500              S
## 6  NA     0     0           330877  8.4583              Q
```

```
head(test_raw)
```

```
##   PassengerId Pclass                                          Name    Sex  Age
## 1         892      3                              Kelly, Mr. James   male 34.5
## 2         893      3              Wilkes, Mrs. James (Ellen Needs) female 47.0
## 3         894      2                     Myles, Mr. Thomas Francis   male 62.0
## 4         895      3                               Wirz, Mr. Albert   male 27.0
## 5         896      3 Hirvonen, Mrs. Alexander (Helga E Lindqvist) female 22.0
## 6         897      3                     Svensson, Mr. Johan Cervin   male 14.0
##   SibSp Parch  Ticket    Fare Cabin Embarked
## 1     0     0  330911  7.8292              Q
## 2     1     0  363272  7.0000              S
## 3     0     0  240276  9.6875              Q
## 4     0     0  315154  8.6625              S
## 5     1     1 3101298 12.2875              S
## 6     0     0    7538  9.2250              S
```

Integrem les dades dels dos datasets en un de sol.

```
dataset<-rbind(train_raw,test_raw)
head(dataset)
```

```
##   PassengerId Pclass                                                Name    Sex
## 1           1      3                             Braund, Mr. Owen Harris   male
## 2           2      1 Cumings, Mrs. John Bradley (Florence Briggs Thayer) female
## 3           3      3                              Heikkinen, Miss. Laina female
## 4           4      1        Futrelle, Mrs. Jacques Heath (Lily May Peel) female
## 5           5      3                            Allen, Mr. William Henry   male
## 6           6      3                                    Moran, Mr. James   male
##   Age SibSp Parch           Ticket    Fare Cabin Embarked
## 1  22     1     0        A/5 21171  7.2500              S
## 2  38     1     0         PC 17599 71.2833   C85        C
## 3  26     0     0 STON/O2. 3101282  7.9250              S
## 4  35     1     0           113803 53.1000  C123        S
## 5  35     0     0           373450  8.0500              S
## 6  NA     0     0           330877  8.4583              Q
```

```
dim(dataset)
```

```
## [1] 1309   11
```

A partir de la informació observada decidim canviar les classes d'algunes variables. Al canviar la classe de la variable 'Age', tots aquells passatgers menors d'1 any passaran a tenir edat = 0 (indicatiu de que tenen mesos de vida)

```
dataset <- within(dataset, {
  Pclass <- factor(Pclass)
  Sex <- factor(Sex)
  Age <- as.integer(Age)
```

```
  Embarked <- factor(Embarked)
})
```

Tornem a inspeccionar:

```
# Inspeccionem les dades
glimpse(dataset)
```

```
## Rows: 1,309
## Columns: 11
## $ PassengerId <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, ...
## $ Pclass      <fct> 3, 1, 3, 1, 3, 3, 1, 3, 3, 2, 3, 1, 3, 3, 3, 2, 3, 2, 3...
## $ Name        <chr> "Braund, Mr. Owen Harris", "Cumings, Mrs. John Bradley ...
## $ Sex         <fct> male, female, female, female, male, male, male, male, f...
## $ Age         <int> 22, 38, 26, 35, 35, NA, 54, 2, 27, 14, 4, 58, 20, 39, 1...
## $ SibSp       <int> 1, 1, 0, 1, 0, 0, 0, 3, 0, 1, 1, 0, 0, 1, 0, 0, 4, 0, 1...
## $ Parch       <int> 0, 0, 0, 0, 0, 0, 0, 1, 2, 0, 1, 0, 0, 5, 0, 0, 1, 0, 0...
## $ Ticket      <chr> "A/5 21171", "PC 17599", "STON/O2. 3101282", "113803", ...
## $ Fare        <dbl> 7.2500, 71.2833, 7.9250, 53.1000, 8.0500, 8.4583, 51.86...
## $ Cabin       <chr> "", "C85", "", "C123", "", "", "E46", "", "", "", "G6",...
## $ Embarked    <fct> S, C, S, S, S, Q, S, S, S, C, S, S, S, S, S, S, Q, S, S...
```

Observacions:

- Les dimensions són les esperades.
- De la variable 'Name' es poden extreure els títols d'algunes persones.
- La variable 'Sex' es podria codificar de forma binaria.
- La variable 'Fare' sembla representar el preu total pagat alhora de comprar més d'un tiquet junts (e.g. families) envés del tiquet individual.
- Podem crear 'dummy' variables per a les variables 'Pclass', 'Sex' i 'Embarked'
- De la variable 'cabin' es pot extreure la lletra que segurament representa diferentes zones del vaixell.
- Dels resultats de la funció summary() i glimpse() es pot veure com tenim varis valors nuls o mencants en les variables: Age, Fare, Cabin i Embarked

3. Neteja de dades

3.1. Gestió de valors buits o nul

Creem una funció que indica les variables que contenen valors nul i l'executem passant-li el dataset que hem carregat.

```
has_na <- function(dades) {
  no_na <- TRUE
  for (i in names(dades)) {
    a <- sum(is.na(dades[[i]]))
    if (a != 0) {
      no_na <- FALSE
      print(paste("La variable ", i, " té ", a, " valors nul"))
    }
    else if (a == 0 & is.character(dades[[i]])) {
      b <- length(dades[[i]][which(dades[[i]]=="")])
      if (b != 0) {
        no_na <- FALSE
        print(paste("La variable ", i, " té ", b, " valors buits"))
      }
    }
```

```
  }
  if (no_na) {
    print("No hi ha cap variable amb valors nul")
  }
}
```

Un cop creada la funció l'executem per saber quines variables tenen valors nul en els diferents datasets. Dataset d'entrenament:

```
has_na(dataset)
```

```
## [1] "La variable  Age  té  263  valors nul"
## [1] "La variable  Fare  té  1  valors nul"
## [1] "La variable  Cabin  té  1014  valors buits"
```

```
# Observant les dades veiem que la variable 'Embarked' també té 2 valors buits no identificats amb la fu
table(dataset$Embarked,useNA='always')
```

```
##
##         C    Q    S <NA>
##    2  270  123  914    0
```

```
# Aquests es troben en el passatger número 62 i 830
dataset[which(dataset$Embarked==""),]
```

```
##     PassengerId Pclass                                         Name    Sex Age
## 62           62      1                         Icard, Miss. Amelie female  38
## 830         830      1 Stone, Mrs. George Nelson (Martha Evelyn) female  62
##     SibSp Parch Ticket Fare Cabin Embarked
## 62      0     0 113572   80   B28
## 830     0     0 113572   80   B28
```

Com que només es tracta de dos valors, i aquest passatgers viatjaven sols, assumirem que aquests passatgers van embarcar al mateix port que la majoria de passatgers ('S'). De donar-se el cas de que algun d'aquests viatgers no viatges sol, també podriem assumir que que el port d'embarcament era el mateix que els seus familiars. Aquest el podriem trobar mitjançant el cognom o nom de familia del passatger.

```
dataset[62,]$Embarked<-'S'
dataset[830,]$Embarked<-'S'

table(dataset$Embarked,useNA='always')
```

```
##
##         C    Q    S <NA>
##    0  270  123  916    0
```

```
dataset$Embarked<-droplevels(dataset$Embarked)
```

En quan a la variable 'Fare', assumirem que aquest té el mateix valor mig que el que altres passatgers de viatjant en circumstancies similars van pagar.

```
# Aquests es troba en el passatger número 1044
dataset[which(is.na(dataset$Fare)),]
```

```
##      PassengerId Pclass               Name  Sex Age SibSp Parch Ticket Fare
## 1044        1044      3 Storey, Mr. Thomas male  60     0     0   3701   NA
##      Cabin Embarked
## 1044              S
```

```
dataset[1044,]$Fare<-mean(dataset[which(dataset$Pclass==3&dataset$SibSp==0&dataset$Parch==0),]$Fare,na.

dataset[1044,]$Fare
```

```
## [1] 9.096707
```

La variable 'Cabin' conté molts valors mencants, per a tant els tractarem com a mencants i no intentarem imputar els que falten.

En quan a la variable 'Age' imputarem les edats mencants segons el grup al que pertanyi cadascun dels passatgers on aquestes dades manquen. Hem decidit que fariem servir les variables 'Sex', 'Pclass' i 'Parch' igual o diferent a 0 per a crear els grups dels quals imputar les edats mancants.

```
dataset.maleC1Parch0<-dataset[which(dataset$Sex=='male'&dataset$Pclass=='1'&dataset$Parch<1),]
dataset.maleC1Parch1<-dataset[which(dataset$Sex=='male'&dataset$Pclass=='1'&dataset$Parch>=1),]
dataset.maleC2Parch0<-dataset[which(dataset$Sex=='male'&dataset$Pclass=='2'&dataset$Parch<1),]
dataset.maleC2Parch1<-dataset[which(dataset$Sex=='male'&dataset$Pclass=='2'&dataset$Parch>=1),]
dataset.maleC3Parch0<-dataset[which(dataset$Sex=='male'&dataset$Pclass=='3'&dataset$Parch<1),]
dataset.maleC3Parch1<-dataset[which(dataset$Sex=='male'&dataset$Pclass=='3'&dataset$Parch>=1),]
dataset.femaleC1Parch0<-dataset[which(dataset$Sex=='female'&dataset$Pclass=='1'&dataset$Parch<1),]
dataset.femaleC1Parch1<-dataset[which(dataset$Sex=='female'&dataset$Pclass=='1'&dataset$Parch>=1),]
dataset.femaleC2Parch0<-dataset[which(dataset$Sex=='female'&dataset$Pclass=='2'&dataset$Parch<1),]
dataset.femaleC2Parch1<-dataset[which(dataset$Sex=='female'&dataset$Pclass=='2'&dataset$Parch>=1),]
dataset.femaleC3Parch0<-dataset[which(dataset$Sex=='female'&dataset$Pclass=='3'&dataset$Parch<1),]
dataset.femaleC3Parch1<-dataset[which(dataset$Sex=='female'&dataset$Pclass=='3'&dataset$Parch>=1),]

# Comprovem que les particions estan bé
nrow(dataset.maleC1Parch0)+
nrow(dataset.maleC1Parch1)+
nrow(dataset.maleC2Parch0)+
nrow(dataset.maleC2Parch1)+
nrow(dataset.maleC3Parch0)+
nrow(dataset.maleC3Parch1)+
nrow(dataset.femaleC1Parch0)+
nrow(dataset.femaleC1Parch1)+
nrow(dataset.femaleC2Parch0)+
nrow(dataset.femaleC2Parch1)+
nrow(dataset.femaleC3Parch0)+
nrow(dataset.femaleC3Parch1)
```

```
## [1] 1309
```

Comprovem si cadascun dels grups mostre distribucions d'edat amb mitjanes diferents.

```
#maleC1Parch0
age_mean1 <- mean(dataset.maleC1Parch0$Age, na.rm = T)
age_median1 <- median(dataset.maleC1Parch0$Age, na.rm = T)

p_maleC1Parch0<-ggplot(dataset.maleC1Parch0, aes(Age)) +
  geom_density(fill = 'orchid', alpha =0.7) +
  geom_vline(aes(xintercept = age_mean1, color = 'mean'), linetype = 5) +
  geom_vline(aes(xintercept = age_median1, color = 'median'), linetype = 5) +
  scale_color_manual(name = "stats", values = c(mean = 'indianred', median = 'steelblue')) +
  ggtitle(label ="maleC1Parch0")

#maleC1Parch1
age_mean2 <- mean(dataset.maleC1Parch1$Age, na.rm = T)
```

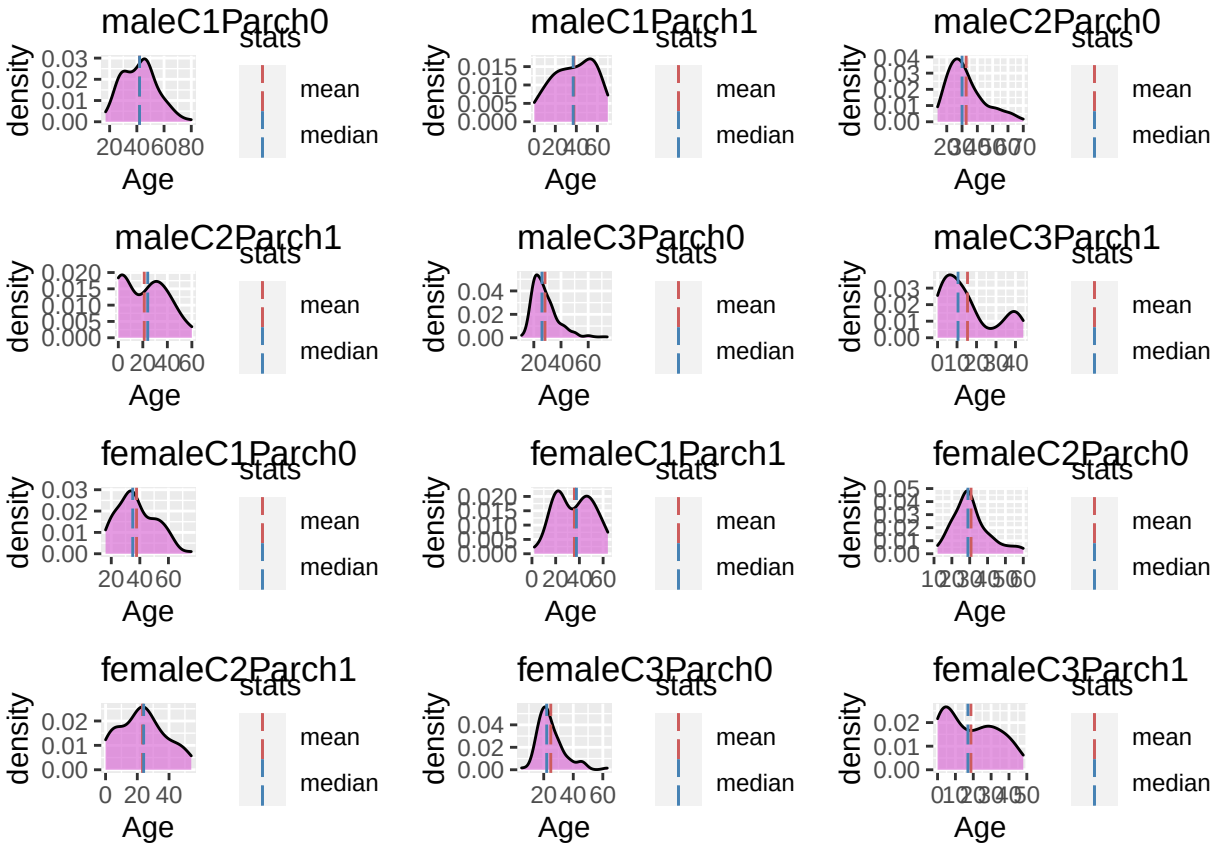
```r
age_median2 <- median(dataset.maleC1Parch1$Age, na.rm = T)

p_maleC1Parch1<-ggplot(dataset.maleC1Parch1, aes(Age)) +
  geom_density(fill = 'orchid', alpha =0.7) +
  geom_vline(aes(xintercept = age_mean2, color = 'mean'), linetype = 5) +
  geom_vline(aes(xintercept = age_median2, color = 'median'), linetype = 5) +
  scale_color_manual(name = "stats", values = c(mean = 'indianred', median = 'steelblue')) +
  ggtitle(label ="maleC1Parch1")

#maleC2Parch0
age_mean3 <- mean(dataset.maleC2Parch0$Age, na.rm = T)
age_median3 <- median(dataset.maleC2Parch0$Age, na.rm = T)

p_maleC2Parch0<-ggplot(dataset.maleC2Parch0, aes(Age)) +
  geom_density(fill = 'orchid', alpha =0.7) +
  geom_vline(aes(xintercept = age_mean3, color = 'mean'), linetype = 5) +
  geom_vline(aes(xintercept = age_median3, color = 'median'), linetype = 5) +
  scale_color_manual(name = "stats", values = c(mean = 'indianred', median = 'steelblue')) +
  ggtitle(label ="maleC2Parch0")

#maleC2Parch1
age_mean4 <- mean(dataset.maleC2Parch1$Age, na.rm = T)
age_median4 <- median(dataset.maleC2Parch1$Age, na.rm = T)

p_maleC2Parch1<-ggplot(dataset.maleC2Parch1, aes(Age)) +
  geom_density(fill = 'orchid', alpha =0.7) +
  geom_vline(aes(xintercept = age_mean4, color = 'mean'), linetype = 5) +
  geom_vline(aes(xintercept = age_median4, color = 'median'), linetype = 5) +
  scale_color_manual(name = "stats", values = c(mean = 'indianred', median = 'steelblue')) +
  ggtitle(label ="maleC2Parch1")

#maleC3Parch0
age_mean5 <- mean(dataset.maleC3Parch0$Age, na.rm = T)
age_median5 <- median(dataset.maleC3Parch0$Age, na.rm = T)

p_maleC3Parch0<-ggplot(dataset.maleC3Parch0, aes(Age)) +
  geom_density(fill = 'orchid', alpha =0.7) +
  geom_vline(aes(xintercept = age_mean5, color = 'mean'), linetype = 5) +
  geom_vline(aes(xintercept = age_median5, color = 'median'), linetype = 5) +
  scale_color_manual(name = "stats", values = c(mean = 'indianred', median = 'steelblue')) +
  ggtitle(label ="maleC3Parch0")

#maleC3Parch1
age_mean6 <- mean(dataset.maleC3Parch1$Age, na.rm = T)
age_median6 <- median(dataset.maleC3Parch1$Age, na.rm = T)

p_maleC3Parch1<-ggplot(dataset.maleC3Parch1, aes(Age)) +
  geom_density(fill = 'orchid', alpha =0.7) +
  geom_vline(aes(xintercept = age_mean6, color = 'mean'), linetype = 5) +
  geom_vline(aes(xintercept = age_median6, color = 'median'), linetype = 5) +
  scale_color_manual(name = "stats", values = c(mean = 'indianred', median = 'steelblue')) +
  ggtitle(label ="maleC3Parch1")
```

```
#femaleC1Parch0
age_mean7 <- mean(dataset.femaleC1Parch0$Age, na.rm = T)
age_median7 <- median(dataset.femaleC1Parch0$Age, na.rm = T)

p_femaleC1Parch0<-ggplot(dataset.femaleC1Parch0, aes(Age)) +
  geom_density(fill = 'orchid', alpha =0.7) +
  geom_vline(aes(xintercept = age_mean7, color = 'mean'), linetype = 5) +
  geom_vline(aes(xintercept = age_median7, color = 'median'), linetype = 5) +
  scale_color_manual(name = "stats", values = c(mean = 'indianred', median = 'steelblue')) +
  ggtitle(label ="femaleC1Parch0")

#femaleC1Parch1
age_mean8 <- mean(dataset.femaleC1Parch1$Age, na.rm = T)
age_median8 <- median(dataset.femaleC1Parch1$Age, na.rm = T)

p_femaleC1Parch1<-ggplot(dataset.femaleC1Parch1, aes(Age)) +
  geom_density(fill = 'orchid', alpha =0.7) +
  geom_vline(aes(xintercept = age_mean8, color = 'mean'), linetype = 5) +
  geom_vline(aes(xintercept = age_median8, color = 'median'), linetype = 5) +
  scale_color_manual(name = "stats", values = c(mean = 'indianred', median = 'steelblue')) +
  ggtitle(label ="femaleC1Parch1")

#femaleC2Parch0
age_mean9 <- mean(dataset.femaleC2Parch0$Age, na.rm = T)
age_median9 <- median(dataset.femaleC2Parch0$Age, na.rm = T)

p_femaleC2Parch0<-ggplot(dataset.femaleC2Parch0, aes(Age)) +
  geom_density(fill = 'orchid', alpha =0.7) +
  geom_vline(aes(xintercept = age_mean9, color = 'mean'), linetype = 5) +
  geom_vline(aes(xintercept = age_median9, color = 'median'), linetype = 5) +
  scale_color_manual(name = "stats", values = c(mean = 'indianred', median = 'steelblue')) +
  ggtitle(label ="femaleC2Parch0")

#femaleC2Parch1
age_mean10 <- mean(dataset.femaleC2Parch1$Age, na.rm = T)
age_median10 <- median(dataset.femaleC2Parch1$Age, na.rm = T)

p_femaleC2Parch1<-ggplot(dataset.femaleC2Parch1, aes(Age)) +
  geom_density(fill = 'orchid', alpha =0.7) +
  geom_vline(aes(xintercept = age_mean10, color = 'mean'), linetype = 5) +
  geom_vline(aes(xintercept = age_median10, color = 'median'), linetype = 5) +
  scale_color_manual(name = "stats", values = c(mean = 'indianred', median = 'steelblue')) +
  ggtitle(label ="femaleC2Parch1")

#femaleC3Parch0
age_mean11 <- mean(dataset.femaleC3Parch0$Age, na.rm = T)
age_median11 <- median(dataset.femaleC3Parch0$Age, na.rm = T)

p_femaleC3Parch0<-ggplot(dataset.femaleC3Parch0, aes(Age)) +
  geom_density(fill = 'orchid', alpha =0.7) +
  geom_vline(aes(xintercept = age_mean11, color = 'mean'), linetype = 5) +
  geom_vline(aes(xintercept = age_median11, color = 'median'), linetype = 5) +
  scale_color_manual(name = "stats", values = c(mean = 'indianred', median = 'steelblue')) +
```

```
  ggtitle(label ="femaleC3Parch0")

#femaleC3Parch1
age_mean12 <- mean(dataset.femaleC3Parch1$Age, na.rm = T)
age_median12 <- median(dataset.femaleC3Parch1$Age, na.rm = T)

p_femaleC3Parch1<-ggplot(dataset.femaleC3Parch1, aes(Age)) +
  geom_density(fill = 'orchid', alpha =0.7) +
  geom_vline(aes(xintercept = age_mean12, color = 'mean'), linetype = 5) +
  geom_vline(aes(xintercept = age_median12, color = 'median'), linetype = 5) +
  scale_color_manual(name = "stats", values = c(mean = 'indianred', median = 'steelblue')) +
  ggtitle(label ="femaleC3Parch1")


figure_Ages <- ggarrange(p_maleC1Parch0,p_maleC1Parch1,p_maleC2Parch0,p_maleC2Parch1,p_maleC3Parch0,p_ma
```

```
## Warning: Removed 28 rows containing non-finite values (stat_density).
## Warning: Removed 13 rows containing non-finite values (stat_density).
## Warning: Removed 132 rows containing non-finite values (stat_density).
## Warning: Removed 12 rows containing non-finite values (stat_density).
## Warning: Removed 10 rows containing non-finite values (stat_density).
## Warning: Removed 1 rows containing non-finite values (stat_density).
## Warning: Removed 3 rows containing non-finite values (stat_density).
## Warning: Removed 48 rows containing non-finite values (stat_density).
## Warning: Removed 16 rows containing non-finite values (stat_density).
```

```
figure_Ages
```




```
impute_age <- function (x) {
    x[is.na(x)] <- median(x, na.rm = T)
    return(x)
}

impute_age(dataset.maleC1Parch0$Age)
```

```
##   [1] 54 28 40 28 42 42 45 42 28 46 71 47 37 24 42 61 56 42 45 40 38 44 62 40 42
##  [26] 42 42 42 45 29 45 42 25 22 28 50 34 52 30 49 65 48 47 56 42 25 58 55 71 18
##  [51] 42 36 47 42 45 50 64 62 42 36 36 49 42 35 27 42 61 80 32 42 48 56 50 47 31
##  [76] 27 31 60 35 42 42 48 27 35 42 36 19 42 29 46 42 39 42 38 42 51 31 33 26 46
## [101] 55 41 30 42 45 24 47 31 28 32 67 49 25 36 53 42 42 42 42 48 54 42 47 42 42
## [126] 39 64 41 27 42 46 24 42 30 57 33 46 39 30 50 49 55 23 17 43
```

```
dataset.maleC1Parch0$Age<-impute_age(dataset.maleC1Parch0$Age)
dataset.maleC1Parch1$Age<-impute_age(dataset.maleC1Parch1$Age)
dataset.maleC2Parch0$Age<-impute_age(dataset.maleC2Parch0$Age)
dataset.maleC2Parch1$Age<-impute_age(dataset.maleC2Parch1$Age)
dataset.maleC3Parch0$Age<-impute_age(dataset.maleC3Parch0$Age)
dataset.maleC3Parch1$Age<-impute_age(dataset.maleC3Parch1$Age)
dataset.femaleC1Parch0$Age<-impute_age(dataset.femaleC1Parch0$Age)
dataset.femaleC1Parch1$Age<-impute_age(dataset.femaleC1Parch1$Age)
dataset.femaleC2Parch0$Age<-impute_age(dataset.femaleC2Parch0$Age)
dataset.femaleC2Parch1$Age<-impute_age(dataset.femaleC2Parch1$Age)
dataset.femaleC3Parch0$Age<-impute_age(dataset.femaleC3Parch0$Age)
dataset.femaleC3Parch1$Age<-impute_age(dataset.femaleC3Parch1$Age)
```

```r
dataset_imputed<-rbind(dataset.maleC1Parch0,dataset.maleC1Parch1,dataset.maleC2Parch0,dataset.maleC2Parc

dataset_imputed<-dataset_imputed[,c('PassengerId','Age')]
names(dataset_imputed)[2]<-'Age_imp'

dataset<-merge(dataset,dataset_imputed,by = 'PassengerId')
```

3.2. Creació de variables

3.2.1. Creació variable 'Title'

A partir del nom dels passatgers es poden extreure els titols d'algun d'ells.

```r
# Extreiem el 'titol' de cada passatger. Com que el format dels nomes sempre es el mateix podem seguir u
dataset$title <- str_sub(dataset$Name, str_locate(dataset$Name, ",")[ , 1] + 2, str_locate(dataset$Name

# combines els títols dels passatgers en grups segons si considerem que formen part de la noblesa (i.e.
# Titols nobles per homes
names_noblesa <- c("Capt", "Col", "Don", "Dr", "Jonkheer", "Major", "Rev", "Sir","Lady", "Mlle", "Mme",
dataset$title[dataset$title %in% names_noblesa] <- "noble"
dataset$title<-factor(dataset$title)
# Comprobem el resultat final
table(dataset$title)
```

```
##
## Master   Miss     Mr    Mrs  noble
##     61    260    757    197     34
```

3.2.2. Creació variable 'Zona'

De la variable 'cabin' extreiem la lletra que segurament representa la zona del vaixell on es troba la cabina.

```r
Zona <- dataset %>%
  select(Cabin) %>%
  mutate(Zona = factor(str_extract(Cabin, pattern = "^.")))

dataset$Zona<-as.character(Zona$Zona)

for (i in 1:length(dataset$Zona)){
  if(is.na(dataset[i,]$Zona)){
    dataset[i,]$Zona<-'Desconeguda'
  }
}

dataset$Zona<-factor(dataset$Zona)
```

3.2.3. Creació variable 'Familiy_Size' i 'FamilyCat'

El tamany de les families es pot inferir a partir de les variables 'SibSp' i 'Parch'. A més, tota aquella gent que comparteixi cognom és probable que formin part de la mateixa familia. Dataset d'entrenament:

```r
#Creem la variable 'Familiy_name"
dataset$Family_name <- str_replace(string = dataset$Name, pattern = ",.*", replacement = "")

#Visualitzem les dades
dataset %>%
```

```
  select(Name, Family_name) %>%
  head(10)
```

```
##                                                 Name Family_name
## 1                             Braund, Mr. Owen Harris      Braund
## 2  Cumings, Mrs. John Bradley (Florence Briggs Thayer)     Cumings
## 3                              Heikkinen, Miss. Laina   Heikkinen
## 4        Futrelle, Mrs. Jacques Heath (Lily May Peel)    Futrelle
## 5                            Allen, Mr. William Henry       Allen
## 6                                    Moran, Mr. James       Moran
## 7                             McCarthy, Mr. Timothy J    McCarthy
## 8                      Palsson, Master. Gosta Leonard     Palsson
## 9    Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)     Johnson
## 10                  Nasser, Mrs. Nicholas (Adele Achem)      Nasser
```

```
#Ceem la variable 'Family_size' (sumem 1 per assegurar-nos de que el passatger també es té en compte en
dataset <- dataset %>%
  mutate(Family_size = SibSp + Parch + 1)

#Visualitzem les dades
dataset %>%
  select(Family_name, SibSp, Parch, Family_size) %>%
   arrange(Family_name)%>%
  head(10)
```

```
##    Family_name SibSp Parch Family_size
## 1       Abbing     0     0           1
## 2       Abbott     1     1           3
## 3       Abbott     1     1           3
## 4       Abbott     0     2           3
## 5     Abelseth     0     0           1
## 6     Abelseth     0     0           1
## 7      Abelson     1     0           2
## 8      Abelson     1     0           2
## 9  Abrahamsson     0     0           1
## 10     Abrahim     0     0           1
```

Categorizem el tamany de les families segons si el passatger va sol, en familia/grup gran o petit

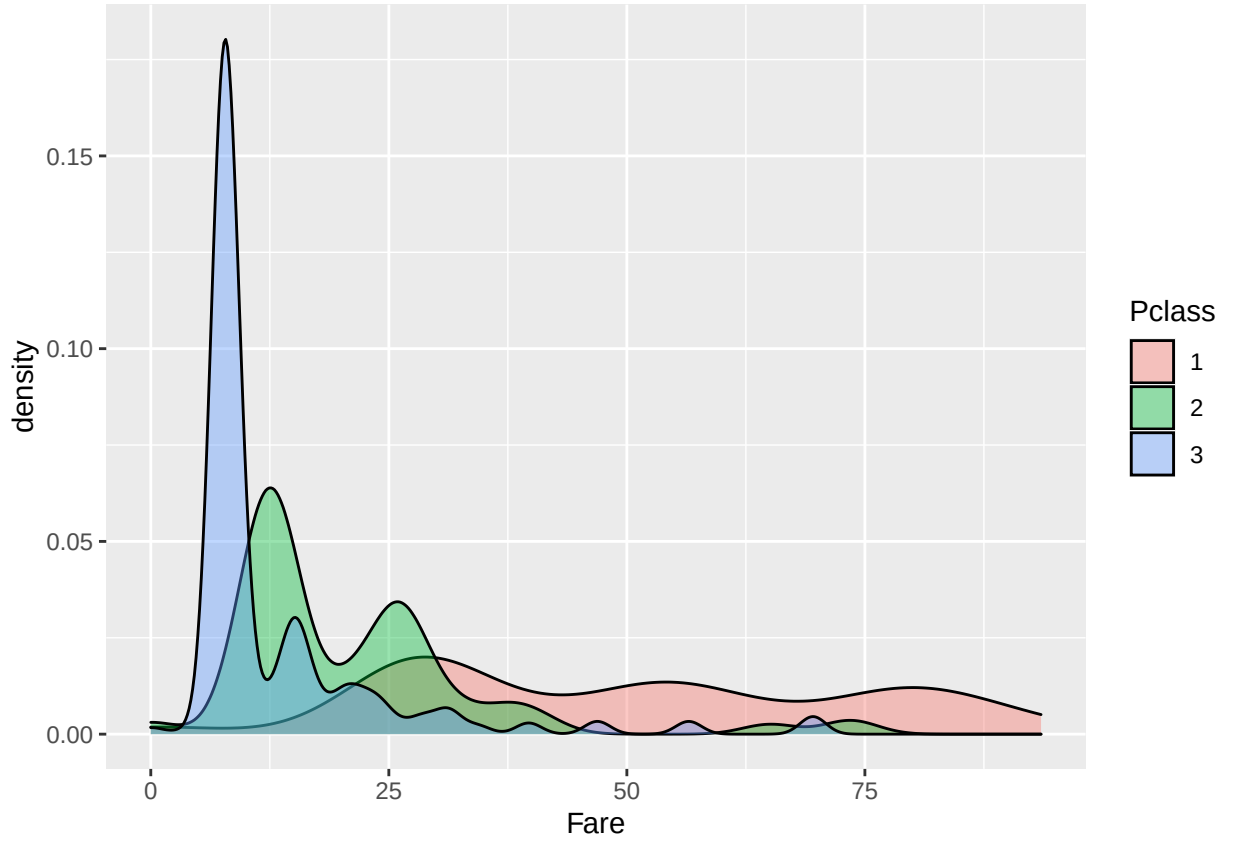
```
dataset <- dataset %>%
  mutate(FamilyCat = factor(case_when(Family_size == 1 ~ 'sol'
                             ,Family_size > 1 & Family_size < 5 ~ 'petita'
                             ,Family_size >= 5 & Family_size < 7 ~ 'moderada'
                             ,Family_size >= 7 ~ 'gran')))
```

3.2.4. Creació variable 'Price_Person'

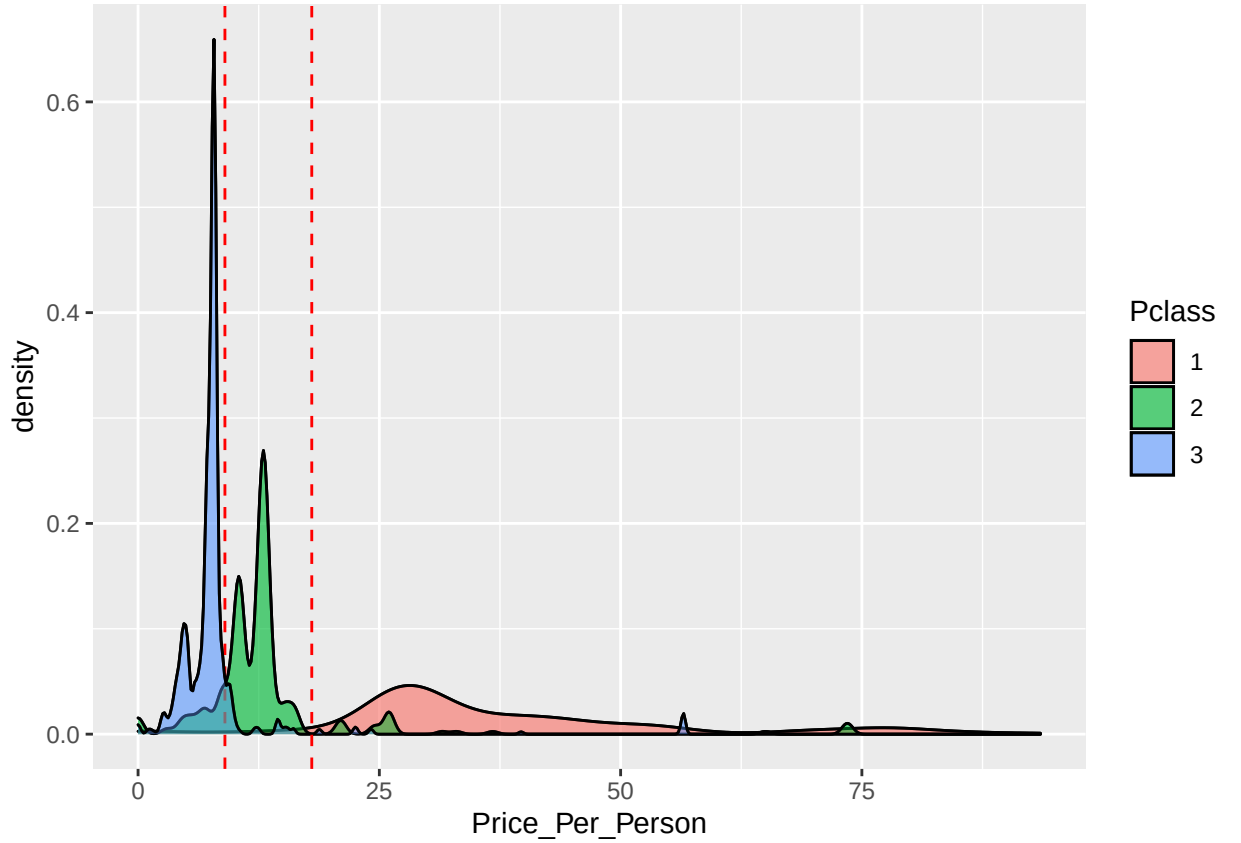
```
dataset$Price_Per_Person<-dataset$Fare/dataset$Family_size
```

Podem visualitzar com ara el preu pagat per ticket per a cada passatger, es mou dins d'un rang logic segons la classe en la que viatjaven: 1a classe = >18 dolars, 2a class = 9-18 dollars, 3a classe = < 9 dolars.

```
# Use semi-transparent fill
p_fare<-ggplot(dataset[which(dataset$Fare<100),], aes(x=Fare, fill=Pclass)) +
  geom_density(alpha=0.4)
p_fare
```




```
# Use semi-transparent fill
p_priceperson<-ggplot(dataset[which(dataset$Price_Per_Person<100),], aes(x=Price_Per_Person, fill=Pclass
  geom_density(alpha=0.4)+ geom_vline(xintercept = 9, linetype="dashed",
               color = "red", size=0.5) +
  geom_density(alpha=0.4)+ geom_vline(xintercept = 18, linetype="dashed",
               color = "red", size=0.5)
p_priceperson
```



Finalment, podem tornar a comprobar que només tenim valors mencants a la variable 'Cabin', 'Zona' que és equivalent, i 'Age', de la qual hem creat una variable nova 'Age_imp' sense valors mencants.

```
has_na(dataset)
```

```
## [1] "La variable  Age  té  263  valors nul"
## [1] "La variable  Cabin  té  1014  valors buits"
```

```
table(dataset$Embarked,useNA='always')
```

```
##
##    C    Q    S <NA>
##  270  123  916    0
```

3.2.5. Creació variable 'AgeCat'

Categorizem l'edat dels passatgers en intervals de 5 anys amb l'objectiu de poder utilitzar aquesta variable a l'hora d'aplicar mètodes que utilitzen atributs categòrics en l'etapa d'anàlisi.

Observem quin és el rang de valors que pren la variable *Age*:

```
summary(dataset$Age_imp)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   22.00   26.00   29.15   36.00   80.00
```

```
dataset["AgeCat"] <- cut(dataset$Age_imp, breaks = c(-1,10,18,55,100), labels = c("nens","adolescents",'

table(dataset$AgeCat,useNA='always')
```

```
##
##        nens adolescents      adults     ancians        <NA>
##          86         138        1027          58           0
```

3.2.6. Creació variable 'PriceCat'

Com hem vist abans, la distribució del preu per persona (Price_Per_Person) segons classe en la que viatgen els passatgers sembla indicar que tenim un preu de referencia del tiquet segons la classe en la que es viatges. Així doncs, hem decidit discretitzar la variable Price_Per_Person en tres grups: preu baix, mig i alt.

```
summary(dataset$Price_Per_Person)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   7.453   8.567  20.509  24.150 512.329
```

```
dataset["PriceCat"] <- cut(dataset$Price_Per_Person, breaks = c(-1,9,18,513), labels = c("baix","mig","a
```

3.3. Gestió de valors extrems (outliers)

Els outliers són valors extrems que es trobem molt lluny de la distribució normal o esperada d'un atribut. De forma formal es solen definir com aquells valors que es troben a més de 3 desviacions estàndards de la mitjana, tot i que altres consideren definicions més estrictes com més enllà de 6 desviacions estàndards de la mitjana. Aquests poden incrementar l'error en la variància i conseqüentment reduir el poder estadístic d'un test alhora de fer estimacions significatives, o que aquestes estimacions estiguin esbiaixades. A més, la presència d'aquests pot afectar dràsticament a la mitjana.

Així doncs, cal observar els valors extrems de les variables numèriques i considerar si es corresponen a valors errònis que cal evitar o bé són casos extrems que convé mantenir.

Per començar l'anàlisi de valors atípics hem creat una funció que mostra els diagrames de caixa de les diferents variables numèriques:

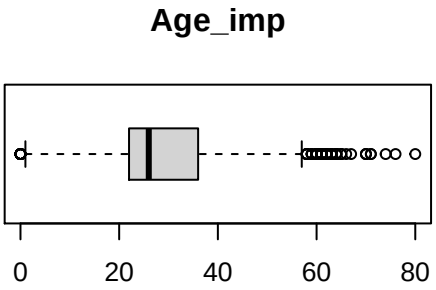
```
mostrar_diag_caixa <- function(dades, variables_numeriques) {
  mida <- ceiling(sqrt(length(variables_numeriques)))
  par(mfrow=c(2,2))
  for (i in variables_numeriques) {
    boxplot(dades[[i]], main=i, horizontal = TRUE)
  }
}
```

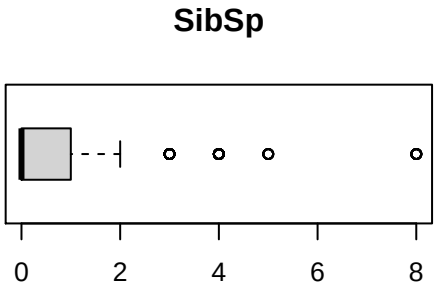
I una altra que indica textualment els valors atípics d'una variable:

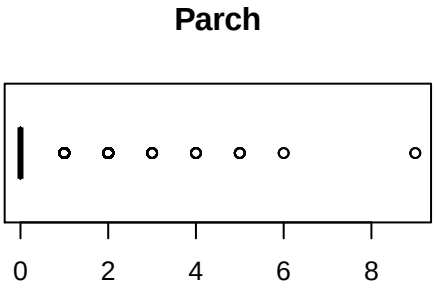
```
deteccio_valors_atipics <- function(dades,variables_numeriques) {
  for (i in variables_numeriques) {
    valors_atipics <- boxplot.stats(dades[[i]])$out
    if (length(valors_atipics != 0)) {
      print(paste("La variable", i, "té els valors atípics: "))
      print(valors_atipics)
    }
  }
}
```

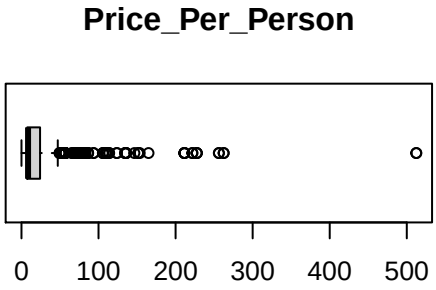
Un cop creades les funcions, les cridem passant-los hi les dades de cada fitxer i un vector amb el nom de les variables numèriques que contenen:

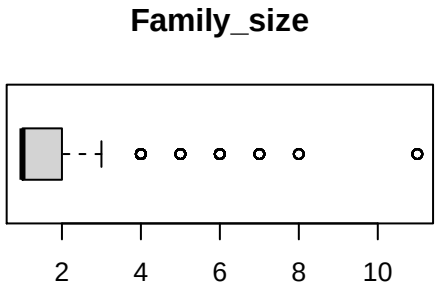
```
mostrar_diag_caixa(dataset,c("Age_imp","SibSp","Parch","Price_Per_Person","Family_size"))
```


Age_imp
0
20
40
60
80


SibSp
0
2
4
6
8


Parch
0
2
4
6
8


Price_Per_Person
0
100
200
300
400
500



Com hem vist enteriorment, tenim varies variables a considerar: Age_imp (omitim Age), SibSp, Parch, Family_size i Price_Per_Person (ommitim Fare).

Tot i que tenim varis valors que s'allunyen considerablement de la mitjana de la distribució d'aquestes variables, considerem que els valors extrems observats són llògics i esperables ja que es moun dins d'un rang de valors que esperariem, tant pel que fa a les edats, o al número de familiars a bord o tamany de familia. L'única variable que ens faria dubtar és el preu pagat per tiquet, tot i que ens falta informació per a saber si aquests preus extrems serien possible (e.g. tiquet a 1a classe comprat a última hora en la millor cabina). Tot i així, com que treballarem amb la variable PriceCat, i tots aquests valors queden categoritzats dins de la categoria de preu 'alt', no és rellevant i per tant no eliminem cap d'aquests valors extrems.

Així doncs considerem que no hi ha outliers a eliminar en les nostres dades finals.

```
deteccio_valors_atipics(dataset,c("Age_imp","SibSp","Parch","Price_Per_Person","Family_size"))
```

```
## [1] "La variable Age_imp té els valors atípics: "
##  [1] 58 66 65  0 59 71 70 61 58 59 62 58 63 65  0 61 60 64 65  0 63 58 71 64 62
## [26] 62 60 61 80  0 58 70 60 60 70  0  0 62  0 74 62 63 60 60 67 76 63 61 60 64
## [51] 61  0 60 64  0  0 64  0 58  0 59
## [1] "La variable SibSp té els valors atípics: "
##  [1] 3 4 3 3 4 5 3 4 5 3 3 4 8 4 4 3 8 4 8 3 4 4 4 4 8 3 3 5 3 5 3 4 4 3 3 5 4 3
## [39] 4 8 4 3 4 8 4 8 3 4 5 3 4 8 4 8 4 3 3
## [1] "La variable Parch té els valors atípics: "
##   [1] 1 2 1 5 1 1 5 2 2 1 1 2 2 2 1 2 2 2 3 2 2 1 1 1 1 2 1 1 2 2 1 2 2 2 1 2 1
##  [38] 1 2 1 4 1 1 1 1 2 2 1 2 1 1 1 2 1 1 2 2 2 1 1 2 2 1 2 1 1 1 1 1 1 1 2 1 2
##  [75] 2 1 1 2 1 1 2 1 1 1 1 2 1 1 1 4 1 1 2 2 2 2 2 1 1 1 2 2 1 1 2 2 3 4 1 2 1
## [112] 1 2 1 2 1 2 1 1 2 2 1 1 1 1 2 2 2 2 2 2 1 1 2 1 4 1 1 2 1 2 1 1 2 5 2 1 1
## [149] 1 2 1 5 2 1 1 1 2 1 6 1 2 1 2 1 1 1 1 1 1 1 3 2 1 1 1 1 2 1 2 3 1 2 1 2 2
```

```
## [186] 1 1 2 1 2 1 2 1 1 1 2 1 1 2 1 2 1 1 1 1 3 2 1 1 1 1 5 2 1 1 1 1 3 1 2 2 1
## [223] 2 1 2 1 2 4 1 1 2 1 1 1 4 6 2 3 1 1 2 2 2 1 1 2 5 2 3 2 1 1 1 2 1 2 2 2 1
## [260] 2 1 1 2 1 2 1 2 1 2 2 1 1 1 1 1 2 1 1 2 1 1 1 2 1 2 9 1 1 1 2 2 2 1 9 1 1
## [297] 2 2 1 1 2 1 1 1 1 1 1
## [1] "La variable Price_Per_Person té els valors atípics: "
##   [1]  51.86250  73.26040  80.00000  73.50000  56.49580  52.00000 123.76040
##   [8]  79.20000  56.49580  50.00000 146.52080  56.63750  76.29170  79.20000
##  [15]  86.50000 512.32920  76.73125 135.63330  78.85000 123.76040 110.88330
##  [22]  54.45000  56.92920  83.15830  52.47500  54.95557 135.63330  76.73125
##  [29]  66.82500 134.50000  69.30000 135.63330  70.50000 227.52500  73.50000
##  [36]  56.63750  52.00000  49.50420  86.50000  54.45000  56.49580  93.50000
##  [43] 221.77920 106.42500  53.21250 227.52500 153.46250  77.95830  69.30000
##  [50]  56.49580 256.16460  76.72920 105.66875  56.49580 113.76250 151.55000
##  [57]  49.50420 227.52500 211.33750 512.32920  52.47500  86.50000 105.66875
##  [64]  79.20000  56.49580  80.00000  56.49580  54.95557  50.49580  52.47500
##  [71]  56.49580  76.29170 262.37500  52.47500 211.50000 211.50000 110.88960
##  [78] 110.88960  75.24170 151.55000  52.47500  51.86250 221.77920  50.49580
##  [85]  82.50693 113.76250  73.50000  54.95557  70.50000  65.00000  53.21250
##  [92]  68.38960  75.24170  68.38960 135.63330  73.26040 211.33750  79.20000
##  [99] 256.16460  73.50000 134.50000 262.37500  50.00000  93.50000 164.86670
## [106]  70.50000 108.90000
## [1] "La variable Family_size té els valors atípics: "
##   [1]  5  7  6  5  7  6  4  6  4  8  6  7  8  4  5  6  4  7  5 11  6  6  6  5 11
##  [26]  7  4 11  5  7  7  6  6  4  4  5 11  6  6  5  8  4  5  4  5  6  6  4  4  4
##  [51]  4  8  5  4  4  7  7  5  4  4  7  4  4  6  6  6  4  8  8  4  6  4  5  5  4
##  [76]  4  5  4  6  4 11  4  7  6  6 11  7  4 11  6  4  5  4  4  4  6  6  5  6  4
## [101]  5  8  8  5  4  7  5  7  4 11  7  4  4  4  4  4 11  4  4 11 11  7  4  5  5
```

3.4. Datasets final

Primer de tot reduim la dimensionalitat del nostre dataset actual per a només treballar amb les variables que ens interessa analitzar. Incorporem també les dades de supervivencia en aquells passatgers pels quals coneixem si van sobreviure o no. I guardarem el fitxer com a 'dataset_titanic_clean.csv', el qual conté les dades netes que utilitzarem més endavant per als anàlisis.

```
dataset_net<-dataset[,c("PassengerId","Pclass","Sex","Zona","FamilyCat","Price_Per_Person","PriceCat","A

# El dataset final
dataset_final<-merge(dataset_net,train_predictions,by = "PassengerId", all = T)

# Convertim la variable 'Survived' en categorica
dataset_final$Survived<-factor(dataset_final$Survived)

write.csv(dataset_final,"dataset_titanic_clean.csv")
```

Podem observar com el dataset final conté 6 variables categoriques i 2 variables numériques (a banda de l'identificador del passatger i si la informació de si va sobreviure o no)

```
glimpse(dataset_final)
```

```
## Rows: 1,309
## Columns: 12
## $ PassengerId      <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,...
## $ Pclass           <fct> 3, 1, 3, 1, 3, 3, 1, 3, 3, 2, 3, 1, 3, 3, 3, 2, 3,...
## $ Sex              <fct> male, female, female, female, male, male, male, ma...
## $ Zona             <fct> Desconeguda, C, Desconeguda, C, Desconeguda, Desco...
```

```
## $ FamilyCat        <fct> petita, petita, sol, petita, sol, sol, sol, modera...
## $ Price_Per_Person <dbl> 3.625000, 35.641650, 7.925000, 26.550000, 8.050000...
## $ PriceCat         <fct> baix, alt, baix, alt, baix, baix, alt, baix, baix,...
## $ Age_imp          <dbl> 22, 38, 26, 35, 35, 26, 54, 2, 27, 14, 4, 58, 20, ...
## $ title            <fct> Mr, Mrs, Miss, Mrs, Mr, Mr, Mr, Master, Mrs, Mrs, ...
## $ Embarked         <fct> S, C, S, S, S, Q, S, S, S, C, S, S, S, S, S, S, Q,...
## $ AgeCat           <fct> adults, adults, adults, adults, adults, adults, ad...
## $ Survived         <fct> 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0,...
```

Tornem a crear els datasets d'entrenament i test inicials utilitzant les dades actuals ja netejades i processades.

```
# El dataset final d'entrenament
dataset_train<-dataset_final[which(!(is.na(dataset_final$Survived))),]
dim(dataset_train)
```

```
## [1] 891  12
```

```
# El dataset final de test
dataset_test<-dataset_final[which(is.na(dataset_final$Survived)),]
dim(dataset_test)
```

```
## [1] 418  12
```

4. Exploració i anàlisis de variables

Primer de tot seleccionem tots els grups de cada variable que ens interessa analitzar. Llavors en comprobarem la normalitat i homogeneïtat de la variància de cadascun d'ells.

```
# Grups seleccion segons classe (Pclass)
table(dataset_final$Pclass,useNA='always')
```

```
##
##    1    2    3 <NA>
##  323  277  709    0
```

```
dataset_final.Class1<-dataset_final[which(dataset_final$Pclass=='1'),]
dataset_final.Class2<-dataset_final[which(dataset_final$Pclass=='2'),]
dataset_final.Class3<-dataset_final[which(dataset_final$Pclass=='3'),]

# Grups seleccion segons sexe
table(dataset_final$Sex,useNA='always')
```

```
##
## female   male   <NA>
##    466    843      0
```

```
dataset_final.Male<-dataset_final[which(dataset_final$Sex=='male'),]
dataset_final.Female<-dataset_final[which(dataset_final$Sex=='female'),]

# Grups seleccion segons Zona
table(dataset_final$Zona,useNA='always')
```

```
##
##           A           B           C           D Desconeguda           E
##          22          65          94          46        1014          41
##           F           G           T        <NA>
##          21           5           1           0
```

```
dataset_final.ZA<-dataset_final[which(dataset_final$Zona=='A'),]
dataset_final.ZB<-dataset_final[which(dataset_final$Zona=='B'),]
dataset_final.ZC<-dataset_final[which(dataset_final$Zona=='C'),]
dataset_final.ZD<-dataset_final[which(dataset_final$Zona=='D'),]
dataset_final.ZE<-dataset_final[which(dataset_final$Zona=='E'),]
dataset_final.ZF<-dataset_final[which(dataset_final$Zona=='F'),]
dataset_final.ZG<-dataset_final[which(dataset_final$Zona=='G'),]
dataset_final.ZT<-dataset_final[which(dataset_final$Zona=='T'),]

# Grups seleccion segons Tamany familia (FamiliyCat)
table(dataset_final$FamilyCat,useNA='always')
```

```
##
##     gran moderada   petita      sol     <NA>
##       35       47      437      790        0
```

```
dataset_final.sol<-dataset_final[which(dataset_final$FamilyCat=='sol'),]
dataset_final.petita<-dataset_final[which(dataset_final$FamilyCat=='petita'),]
dataset_final.moderada<-dataset_final[which(dataset_final$FamilyCat=='moderada'),]
dataset_final.gran<-dataset_final[which(dataset_final$FamilyCat=='gran'),]

# Grups seleccion segons preu pagat (PriceCat)
table(dataset_final$PriceCat,useNA='always')
```

```
##
## baix  mig  alt <NA>
##  703  262  344    0
```

```
dataset_final.baix<-dataset_final[which(dataset_final$PriceCat=='baix'),]
dataset_final.mig<-dataset_final[which(dataset_final$PriceCat=='mig'),]
dataset_final.alt<-dataset_final[which(dataset_final$PriceCat=='alt'),]

# Grups seleccion segons títol
table(dataset_final$title,useNA='always')
```

```
##
## Master   Miss     Mr    Mrs  noble   <NA>
##     61    260    757    197     34      0
```

```
dataset_final.Master<-dataset_final[which(dataset_final$title=='Master'),]
dataset_final.Miss<-dataset_final[which(dataset_final$title=='Miss'),]
dataset_final.Mr<-dataset_final[which(dataset_final$title=='Mr'),]
dataset_final.Mrs<-dataset_final[which(dataset_final$title=='Mrs'),]
dataset_final.Noble<-dataset_final[which(dataset_final$title=='noble'),]

# Grups seleccion segons supervivencia
table(dataset_final$Survived,useNA='always')
```

```
##
##    0    1 <NA>
##  549  342  418
```

```
dataset_final.Viu<-dataset_final[which(dataset_final$Survived=='0'),]
dataset_final.Mor<-dataset_final[which(dataset_final$Survived=='1'),]
```

4.1. Comprobació de la normalitat variables quantitatives

Per a comprobar que les variables quantitatives que tenim segueixen una distribució normal utilitzarem el test de Shapiro-Wilk

```
# Price
shapiro.Price<-shapiro.test(dataset_final$Price_Per_Person)
# Age
shapiro.Age_imp<-shapiro.test(dataset_final$Age_imp)

# Price Supervivents
shapiro.Su.Price<-shapiro.test(dataset_final.Viu$Price_Per_Person)

# Price morts
shapiro.Mo.Price<-shapiro.test(dataset_final.Mor$Price_Per_Person)

# Age Supervivents
shapiro.Su.Age_imp<-shapiro.test(dataset_final.Viu$Age_imp)

# Age Morts
shapiro.Mo.Age_imp<-shapiro.test(dataset_final.Mor$Age_imp)

# Price class1
shapiro.C1.Price<-shapiro.test(dataset_final.Class1$Price_Per_Person)
```

```r
# Price class2
shapiro.C2.Price<-shapiro.test(dataset_final.Class2$Price_Per_Person)

# Price class3
shapiro.C3.Price<-shapiro.test(dataset_final.Class3$Price_Per_Person)
```

4.1.1. Resultats

```r
table_res<-matrix(c(round(mean(dataset_final$Price_Per_Person),2),
         round(mean(dataset_final$Age_imp),2),
         round(mean(dataset_final.Viu$Price_Per_Person),2),
         round(mean(dataset_final.Mor$Price_Per_Person),2),
         round(mean(dataset_final.Class1$Price_Per_Person),2),
         round(mean(dataset_final.Class2$Price_Per_Person),2),
         round(mean(dataset_final.Class3$Price_Per_Person),2),
         round(mean(dataset_final.Viu$Age_imp),2),
         round(mean(dataset_final.Mor$Age_imp),2),
         round(median(dataset_final$Price_Per_Person),2),
         round(median(dataset_final$Age_imp),2),
         round(median(dataset_final.Viu$Price_Per_Person),2),
         round(median(dataset_final.Mor$Price_Per_Person),2),
         round(median(dataset_final.Class1$Price_Per_Person),2),
         round(median(dataset_final.Class2$Price_Per_Person),2),
         round(median(dataset_final.Class3$Price_Per_Person),2),
         round(median(dataset_final.Viu$Age_imp),2),
         round(median(dataset_final.Mor$Age_imp),2),
         round(sd(dataset_final$Price_Per_Person),2),
         round(sd(dataset_final$Age_imp),2),
         round(sd(dataset_final.Viu$Price_Per_Person),2),
         round(sd(dataset_final.Mor$Price_Per_Person),2),
         round(sd(dataset_final.Class1$Price_Per_Person),2),
         round(sd(dataset_final.Class2$Price_Per_Person),2),
         round(sd(dataset_final.Class3$Price_Per_Person),2),
         round(sd(dataset_final.Viu$Age_imp),2),
         round(sd(dataset_final.Mor$Age_imp),2),
         shapiro.Price[2],
         shapiro.Age_imp[2],
         shapiro.Su.Price[2],
         shapiro.Mo.Price[2],
         shapiro.C1.Price[2],
         shapiro.C2.Price[2],
         shapiro.C3.Price[2],
         shapiro.Su.Age_imp[2],
         shapiro.Mo.Age_imp[2]),ncol=4)


colnames(table_res)<-c("Mean","Median","sd","Shapiro-Wilk (p)")
rownames(table_res)<-c("Price","Age","Price Supervivents","Price Morts","Price Class 1", "Price Class 2
table_res<-as.table(table_res)
```

Podem observar com cap de les variables continues analitzades (Age i Price) segueix una distribució normal en cap dels grups testats. El p-value del test de Shapiro-Wilk és < 0.5 en tots els casos. Per tant, en tots els casos és rebutja la hipòtesis nul · la que considera que la distribució és normal.

Taula 2: Resultats de les distribucions de les variables contínues'.

	Mean	Median	sd	Shapiro-Wilk (p)
Price	20.51	8.57	35.76	9.25115055575609e-54
Age	29.15	26	13.36	1.72081984344405e-16
Price Supervivents	13.65	7.92	18.92	4.67584912263855e-38
Price Morts	29.97	13	51.12	6.8562878170371e-31
Price Class 1	54.17	33.3	59.32	1.44437662905357e-27
Price Class 2	13.35	13	9.29	6.52564417492086e-27
Price Class 3	7.97	7.75	5.85	3.0972660079262e-44
Age Supervivents	29.7	26	13.02	2.09657419131529e-12
Age Morts	28.08	27	14.13	0.00040288352431829

Comprobem si la variable edat segueix una distribució normal en algun dels grups creats anteriorment.

```
shapiro.C1.Age_imp<-shapiro.test(dataset_final.Class1$Age_imp)
shapiro.C2.Age_imp<-shapiro.test(dataset_final.Class2$Age_imp)
shapiro.C3.Age_imp<-shapiro.test(dataset_final.Class3$Age_imp)
shapiro.Male.Age_imp<-shapiro.test(dataset_final.Male$Age_imp)
shapiro.Female.Age_imp<-shapiro.test(dataset_final.Female$Age_imp)
shapiro.ZA.Age_imp<-shapiro.test(dataset_final.ZA$Age_imp)
shapiro.ZB.Age_imp<-shapiro.test(dataset_final.ZB$Age_imp)
shapiro.ZC.Age_imp<-shapiro.test(dataset_final.ZC$Age_imp)
shapiro.ZD.Age_imp<-shapiro.test(dataset_final.ZD$Age_imp)
shapiro.ZE.Age_imp<-shapiro.test(dataset_final.ZE$Age_imp)
shapiro.ZF.Age_imp<-shapiro.test(dataset_final.ZF$Age_imp)
shapiro.ZG.Age_imp<-shapiro.test(dataset_final.ZG$Age_imp)
shapiro.sol.Age_imp<-shapiro.test(dataset_final.sol$Age_imp)
shapiro.petita.Age_imp<-shapiro.test(dataset_final.petita$Age_imp)
shapiro.moderada.Age_imp<-shapiro.test(dataset_final.moderada$Age_imp)
shapiro.gran.Age_imp<-shapiro.test(dataset_final.gran$Age_imp)
shapiro.baix.Age_imp<-shapiro.test(dataset_final.baix$Age_imp)
shapiro.mig.Age_imp<-shapiro.test(dataset_final.mig$Age_imp)
shapiro.alt.Age_imp<-shapiro.test(dataset_final.alt$Age_imp)
shapiro.Master.Age_imp<-shapiro.test(dataset_final.Master$Age_imp)
shapiro.Miss.Age_imp<-shapiro.test(dataset_final.Miss$Age_imp)
shapiro.Mr.Age_imp<-shapiro.test(dataset_final.Mr$Age_imp)
shapiro.Mrs.Age_imp<-shapiro.test(dataset_final.Mrs$Age_imp)
shapiro.Noble.Age_imp<-shapiro.test(dataset_final.Noble$Age_imp)
```

```
table_res2<-matrix(c(
  shapiro.C1.Age_imp[2],
  shapiro.C2.Age_imp[2],
  shapiro.C3.Age_imp[2],
  shapiro.Male.Age_imp[2],
  shapiro.Female.Age_imp[2],
  shapiro.ZA.Age_imp[2],
  shapiro.ZB.Age_imp[2],
  shapiro.ZC.Age_imp[2],
```

```
  shapiro.ZD.Age_imp[2],
  shapiro.ZE.Age_imp[2],
  shapiro.ZF.Age_imp[2],
  shapiro.ZG.Age_imp[2],
  shapiro.sol.Age_imp[2],
  shapiro.petita.Age_imp[2],
  shapiro.moderada.Age_imp[2],
  shapiro.gran.Age_imp[2],
  shapiro.baix.Age_imp[2],
  shapiro.mig.Age_imp[2],
  shapiro.alt.Age_imp[2],
  shapiro.Master.Age_imp[2],
  shapiro.Miss.Age_imp[2],
  shapiro.Mr.Age_imp[2],
  shapiro.Mrs.Age_imp[2],
  shapiro.Noble.Age_imp[2]),ncol=1)


colnames(table_res2)<-c("Shapiro-Wilk (p)")
rownames(table_res2)<-c("Classe 1","Classe 2","Classe 3","Homes","Dones", "Zona A", "Zona B", "Zona C",
table_res2<-as.table(table_res2)
```

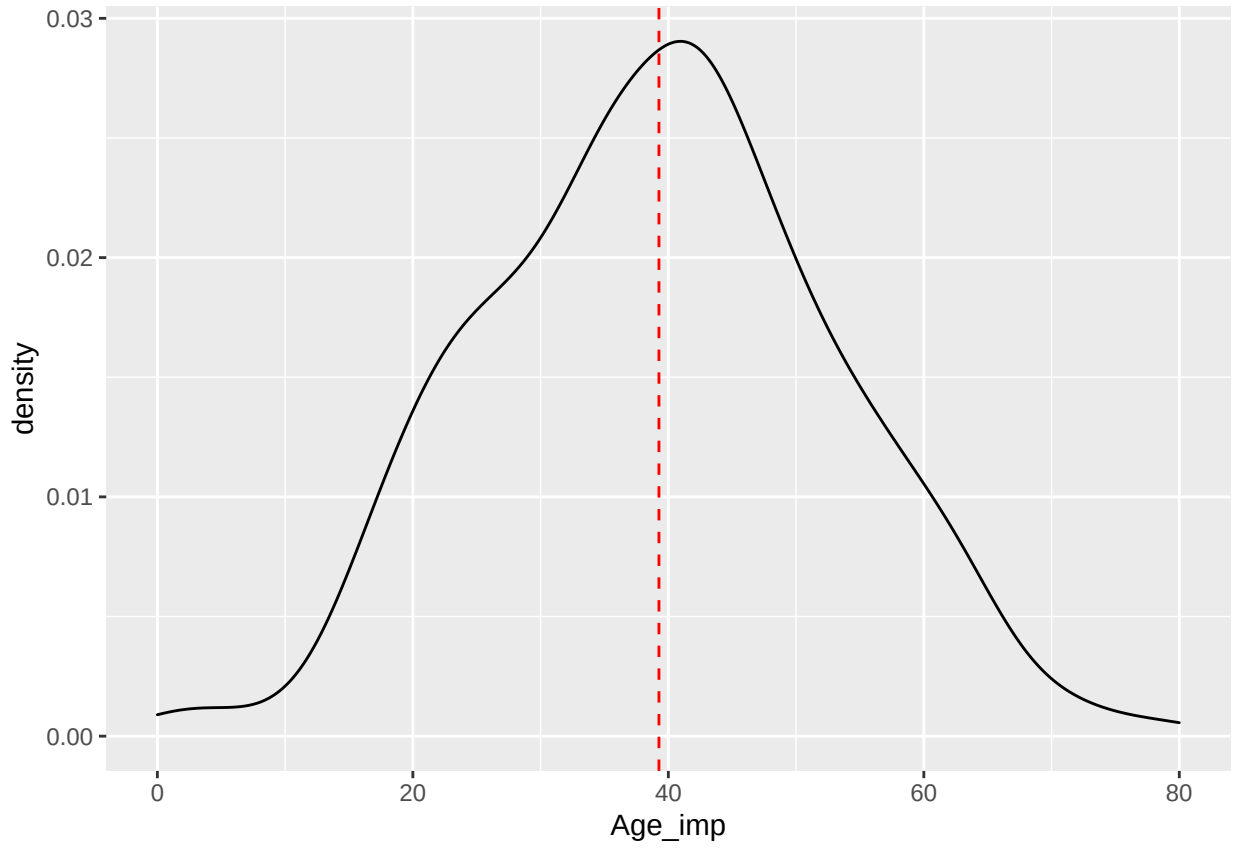
Podem observar com hi ha alguns grups on si que l'edat segeuix una distribució normal. Aquest és el cas per exemple dels passatgers de 1a classe, els que van pagar un preu alt per el tiquet (que és bastant equivalent a ser de 1a classe) i segons les zones de la cabina (amb l'excepció de la zona B).

Podem comprobar alguna d'aquestes distribucions de forma visual. Ho farem amb els passatgers que viatgen en class 1.

```
# Use semi-transparent fill
p_AgeClass1<-ggplot(dataset_final.Class1, aes(x=Age_imp)) +
  geom_density(alpha=0.4)+ geom_vline(xintercept = mean(dataset_final.Class1$Age_imp), linetype="dashed
                color = "red", size=0.5)


p_AgeClass1
```

Taula 3: Test de normalitat en la distribució de l'edat en cadascun dels grups analitzats'.

	Shapiro-Wilk (p)
Classe 1	0.556802698329019
Classe 2	2.099491307761e-05
Classe 3	8.5540861882933 2e-16
Homes	9.46702812199743e-15
Dones	1.31453223361498e-08
Zona A	0.172755572131398
Zona B	0.028731564192564 4
Zona C	0.769303386274288
Zona D	0.090057992954319 5
Zona E	0.655288626331217
Zona F	0.162609827978336
Zona G	0.076403411054259 3
Sol	5.76945371795426e-23
Familia petita	4.94059521491037e-05
Familia moderada	0.000165898968996538
Familia gran	3.29775478487587e-05
Preu tiquet baix	4.8180779286244e-14
Preu tiquet mig	7.1676500991715 2e-07
Preu tiquet alt	0.143929241502201
Master	0.000134152501775469
Miss	3.18629199703566e-07
Mr	1.82032086961577e-21
Mrs	4.98766199680562e-05
Nobles	0.147634611487388



4.3. Comprovació de l'homoscedasticitat

Per a comprobar l'homoscedasticitat (i.e. la homogeneitat de la variança) apliquem el test de Fligner-Killen (test no parametric) ja que com hem vist abans cap de les nostres variables continues segueix una distribució normal.

Aplicarem aquest test per a comprobar si la variança en les variables edat i preu del tiquet són homogenies entre tots els grups de les variables categoriques

```
# Comprovem per l'edat
flin_Pclass<-fligner.test(Age_imp ~ Pclass, data = dataset_final)
flin_Sex<-fligner.test(Age_imp ~ Sex, data = dataset_final)
flin_Zona<-fligner.test(Age_imp ~ Zona, data = dataset_final)
flin_FamilyCat<-fligner.test(Age_imp ~ FamilyCat, data = dataset_final)
flin_PriceCat<-fligner.test(Age_imp ~ PriceCat, data = dataset_final)
flin_title<-fligner.test(Age_imp ~ title, data = dataset_final)
flin_Survived<-fligner.test(Age_imp ~ Survived, data = dataset_final)

# Comprovem pel preu del tiquet
flin_Pclass_Price<-fligner.test(Price_Per_Person ~ Pclass, data = dataset_final)
flin_Sex_Price<-fligner.test(Price_Per_Person ~ Sex, data = dataset_final)
flin_Zona_Price<-fligner.test(Price_Per_Person ~ Zona, data = dataset_final)
flin_FamilyCat_Price<-fligner.test(Price_Per_Person ~ FamilyCat, data = dataset_final)
flin_PriceCat_Price<-fligner.test(Price_Per_Person ~ PriceCat, data = dataset_final)
flin_title_Price<-fligner.test(Price_Per_Person ~ title, data = dataset_final)
flin_Survived_Price<-fligner.test(Price_Per_Person ~ Survived, data = dataset_final)
```

Taula 4: Test de Fligner-Killeen de homogeneitat de la variança: edat i preu del tiquet'.

	Fligner-Killeen (p)
Edat segons classe	0.0000000
Edat segons sexe	0.0265850
Edat segons zona	0.0000719
Edat segons tamany de la familia	0.0000000
Edat segons categoria de preu	0.0000000
Edat segons titol	0.0000002
Edat segons supervivencia	0.0075484
Preu segons classe	0.0000000
Preu segons sexe	0.0000000
Preu segons zona	0.0000000
Preu segons tamany de la familia	0.0000000
Preu segons categoria de preu	0.0000000
Preu segons titol	0.0000000
Preu segons supervivencia	0.0000000

Creem la taula amb els resultats:

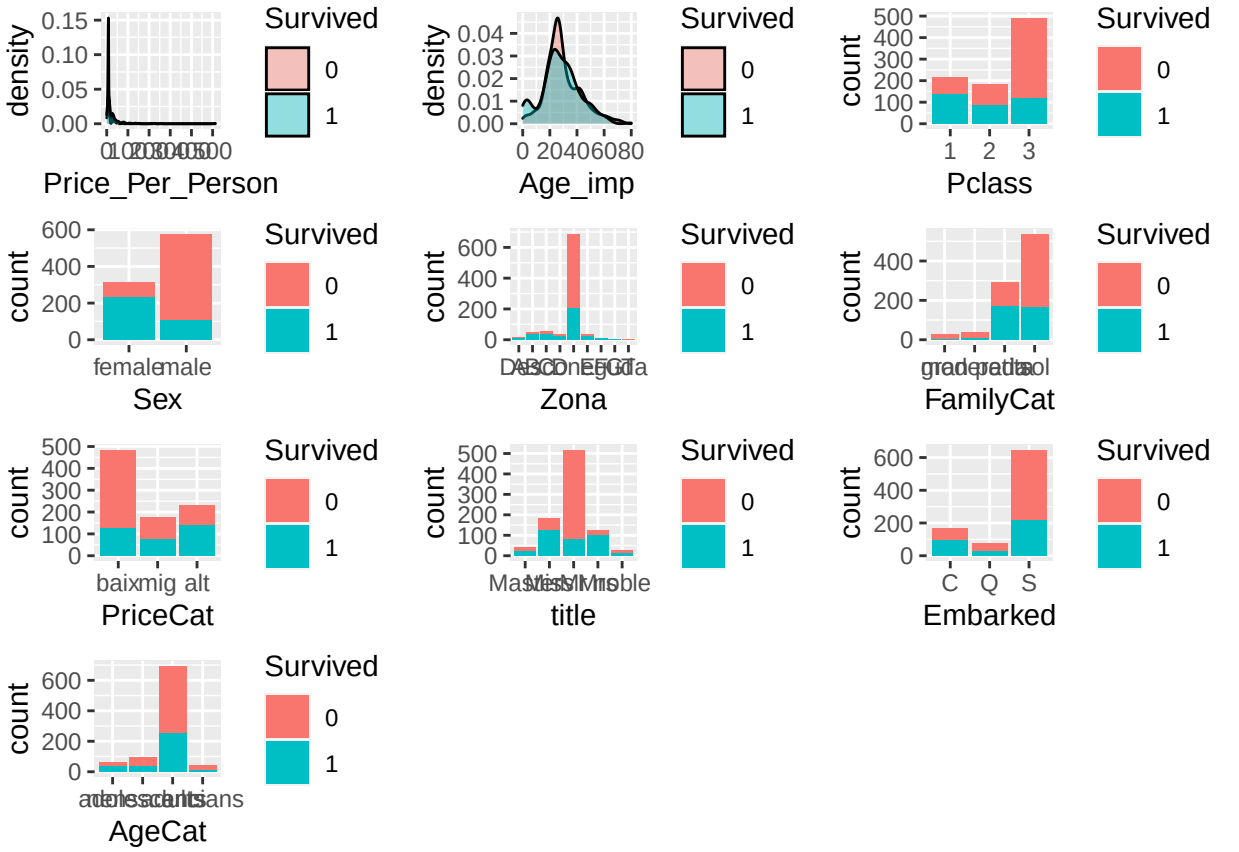
```
table_res4<-matrix(c(
  flin_Pclass$p.value,
  flin_Sex$p.value,
  flin_Zona$p.value,
  flin_FamilyCat$p.value,
  flin_PriceCat$p.value,
  flin_title$p.value,
  flin_Survived$p.value,
  flin_Pclass_Price$p.value,
  flin_Sex_Price$p.value,
  flin_Zona_Price$p.value,
  flin_FamilyCat_Price$p.value,
  flin_PriceCat_Price$p.value,
  flin_title_Price$p.value,
  flin_Survived_Price$p.value),ncol=1)

colnames(table_res4)<-c("Fligner-Killeen (p)")
rownames(table_res4)<-c("Edat segons classe","Edat segons sexe","Edat segons zona", "Edat segons tamany
table_res4<-as.table(table_res4)
```

Podem observar que tans sols la variança de l'edat a través de les diferentes zones és homogenia (acceptem la

hipotesis de que les variances són homogenies ja que el p-valor > 0.5). La variança tant de l'edat com del preu del tiquet no és homogenia entre categories de la resta de variables.

4.2. Distribucions de les variables entre passatgers supervivents i no supervivents

```
# Variables continues
p_price<-ggplot(dataset_final[which(!(is.na(dataset_final$Survived))),], aes(x=Price_Per_Person, fill=Su
  geom_density(alpha=0.4)
p_age<-ggplot(dataset_final[which(!(is.na(dataset_final$Survived))),], aes(x=Age_imp, fill=Survived)) +
  geom_density(alpha=0.4)

# Variables categoriques
p_class<-ggplot(dataset_final[which(!(is.na(dataset_final$Survived))),])+geom_bar(aes(x=Pclass, fill=Su
p_sex<-ggplot(dataset_final[which(!(is.na(dataset_final$Survived))),])+geom_bar(aes(x=Sex, fill=Survive
p_zona<-ggplot(dataset_final[which(!(is.na(dataset_final$Survived))),])+geom_bar(aes(x=Zona, fill=Survi
p_familyCat<-ggplot(dataset_final[which(!(is.na(dataset_final$Survived))),])+geom_bar(aes(x=FamilyCat,
p_priceCat<-ggplot(dataset_final[which(!(is.na(dataset_final$Survived))),])+geom_bar(aes(x=PriceCat, fi
p_title<-ggplot(dataset_final[which(!(is.na(dataset_final$Survived))),])+geom_bar(aes(x=title, fill=Surv
p_Embarked<-ggplot(dataset_final[which(!(is.na(dataset_final$Survived))),])+geom_bar(aes(x=Embarked, fi
p_AgeCat<-ggplot(dataset_final[which(!(is.na(dataset_final$Survived))),])+geom_bar(aes(x=AgeCat, fill=Su

figure_Dis <- ggarrange(p_price,p_age,p_class,p_sex,p_zona,p_familyCat,p_priceCat,p_title,p_Embarked,p_A

figure_Dis
```



4.3. Comparació entre dos grups de dades (variables continues)

Comprovem si hi ha diferencies significatives en la edat i el preu del bitllet entre la gent que va sobreviure i els que van morir.

```
wilcox.test(Price_Per_Person ~ Survived, data = dataset_final[which(!(is.na(dataset_final$Survived))),])
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  Price_Per_Person by Survived
## W = 64104, p-value = 1.551e-15
## alternative hypothesis: true location shift is not equal to 0
```

```
wilcox.test(Age_imp ~ Survived, data = dataset_final[which(!(is.na(dataset_final$Survived))),])
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  Age_imp by Survived
## W = 97870, p-value = 0.2849
## alternative hypothesis: true location shift is not equal to 0
```

Podem observar que hi ha diferencies significatives entre el preu pagat per tiquet de la gent que va sobreviure (més alt) i la que va morir (més baix) En canvi, no s'aprecien diferencies significatives entre la edat de la gent que va sobreviure i la que va morir.

4.4. Comparació entre dos grups de dades (variables categoriques)

Mirem si hi ha diferencies entre les proporcions dels diferents grups de les variables categoriques entre la gent que va sobreviure i la que va morir.

```
# Segons el sexe
sup_sex<-t(table(dataset_final[which(!(is.na(dataset_final$Survived))),]$Sex,dataset_final[which(!(is.na
sup_sex
```

```
##
##     female male
##   0     81  468
##   1    233  109
```

```
chisq.test(sup_sex)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  sup_sex
## X-squared = 260.72, df = 1, p-value < 2.2e-16
```

```
# Segons la zona
sup_zona<-t(table(dataset_final[which(!(is.na(dataset_final$Survived))),]$Zona,dataset_final[which(!(is
sup_zona
```

```
##
##       A   B   C   D Desconeguda   E   F   G   T
##   0   8  12  24   8         481   8   5   2   1
##   1   7  35  35  25         206  24   8   2   0
```

```
chisq.test(sup_zona)
```

```
## Warning in chisq.test(sup_zona): Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  sup_zona
## X-squared = 99.164, df = 8, p-value < 2.2e-16
```

```
# Segons el tamany de familia
sup_FamilyCat<-t(table(dataset_final[which(!(is.na(dataset_final$Survived))),]$FamilyCat,dataset_final[
sup_FamilyCat
```

```
##
##     gran moderada petita sol
##   0   21       31    123 374
##   1    4        6    169 163
```

```
chisq.test(sup_FamilyCat)
```

```
##
##  Pearson's Chi-squared test
##
## data:  sup_FamilyCat
## X-squared = 74.538, df = 3, p-value = 4.552e-16
```

```
# Segons el tamany del preu del bitllet
sup_PriceCat<-t(table(dataset_final[which(!(is.na(dataset_final$Survived))),]$PriceCat,dataset_final[wh
sup_PriceCat
```

```
##
##     baix mig alt
##   0  356 103  90
##   1  128  74 140
```

```
chisq.test(sup_PriceCat)
```

```
##
##  Pearson's Chi-squared test
##
## data:  sup_PriceCat
## X-squared = 79.21, df = 2, p-value < 2.2e-16
```

```
# Segons el titol que es tingui
sup_title<-t(table(dataset_final[which(!(is.na(dataset_final$Survived))),]$title,dataset_final[which(!(
sup_title
```

```
##
##     Master Miss  Mr Mrs noble
##   0     17   55 436  26    15
##   1     23  127  81  99    12
```

```
chisq.test(sup_title)
```

```
##
##  Pearson's Chi-squared test
##
## data:  sup_title
```

```
## X-squared = 283.31, df = 4, p-value < 2.2e-16
```

```
# Segons on s'ha embarcat
sup_Embarked<-t(table(dataset_final[which(!(is.na(dataset_final$Survived))),]$Embarked,dataset_final[wh
sup_Embarked
```

```
##
##       C   Q   S
##   0  75  47 427
##   1  93  30 219
```

```
chisq.test(sup_Embarked)
```

```
##
##  Pearson's Chi-squared test
##
## data:  sup_Embarked
## X-squared = 25.964, df = 2, p-value = 2.301e-06
```

```
# Segons la categoria d'edat
sup_AgeCat<-t(table(dataset_final[which(!(is.na(dataset_final$Survived))),]$AgeCat,dataset_final[which(
sup_AgeCat
```

```
##
##     nens adolescents adults ancians
##   0   26          58    438      27
##   1   38          36    256      12
```

```
chisq.test(sup_AgeCat)
```

```
##
##  Pearson's Chi-squared test
##
## data:  sup_AgeCat
## X-squared = 13.537, df = 3, p-value = 0.003608
```

Podem observar que hi ha diferencies significatives entre les diferents classes de totes les variables categoriques analitzades entre la gent que va sobreviure i la que va morir.

4.6. Correlació entre variables

Calculem la correlació entre les variables continues. Utilitzem el metode de "Spearman" ja que les dades no segueixen una distribució normal.

```
dataset_final_cont<-dataset_final[,c("Price_Per_Person","Age_imp")]

cor.test(dataset_final_cont$Price_Per_Person,dataset_final_cont$Age_imp, method = "spearman")
```

```
## Warning in cor.test.default(dataset_final_cont$Price_Per_Person,
## dataset_final_cont$Age_imp, : Cannot compute exact p-value with ties
```

```
##
##  Spearman's rank correlation rho
##
## data:  dataset_final_cont$Price_Per_Person and dataset_final_cont$Age_imp
## S = 215464945, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
```

```
## 0.4236196
```

Podem observar com l'edat és correlaciona significativament amb el preu del bitllet pagat.

```
corr.res<-cor(dataset_final_cont, method = 'spearman')

# Si tinguessim més variables podriem crear un grafic amb el codi següent:
#corrplot.mixed(corr.res,upper="circle",number.cex=.7,tl.cex=.8)
```

5. Modelització

5.1. Arbre de decisió

Construcció del model

Una manera d'analitzar el dataset és aplicant un model de classificació basat en arbre, el qual ens permetrà veure quines variables tenen més influència a l'hora de determinar la probabilitat de supervivència de cada passatger a partir de les variables que el configuren.

Les variables que s'han utilitzat per construir el model són les següents:

- Pclass
- Sex
- Embarked
- title
- FamilyCat
- AgeCat

Per poder aplicar aquest mètode, és necessari entrenar el model a generar amb els registres corresponents al dataset de train i separar les variables descriptives de la classe que es vol predir. El següent tall de codi prepara les variables que s'utilitzaran a l'etapa d'entrenament i les que s'utilitzaran per avaluar la precisió del model. Com que de les variables de test no coneixem la classe definitiva el que farem serà partir el dataset d'entrenament en dos subconjunts un dels quals servirà per entrenar el model i l'altre per avaluar-lo (2/3 dels registres per l'avaluació i el 1/3 restant per a l'entrenament).

Com que les dades estan ordenades i cal barrejar-les per tal que la tria sigui el màxim d'equitativa possible i no interfereixi amb el resultat que proporcionarà el model. Per fer-ho, apliquem el següent codi i guardem el resultat a la variable *dataset_train_random*:

```
set.seed(1912)
dataset_train_random <- dataset_train[sample(nrow(dataset_train)),]
str(dataset_train_random)
```

```
## 'data.frame':    891 obs. of  12 variables:
##  $ PassengerId     : int  347 499 28 29 144 428 87 65 2 188 ...
##  $ Pclass          : Factor w/ 3 levels "1","2","3": 2 1 1 3 3 2 3 1 1 1 ...
##  $ Sex             : Factor w/ 2 levels "female","male": 1 1 2 1 2 1 2 2 1 2 ...
##  $ Zona            : Factor w/ 9 levels "A","B","C","D",..: 5 3 3 5 5 5 5 5 3 5 ...
##  $ FamilyCat       : Factor w/ 4 levels "gran","moderada",..: 4 3 2 4 4 4 2 4 3 4 ...
##  $ Price_Per_Person: num  13 37.89 43.83 7.88 6.75 ...
##  $ PriceCat        : Factor w/ 3 levels "baix","mig","alt": 2 3 3 1 1 3 1 3 3 3 ...
##  $ Age_imp         : num  40 25 19 22 19 19 16 42 38 45 ...
##  $ title           : Factor w/ 5 levels "Master","Miss",..: 2 4 3 2 3 2 3 3 4 3 ...
##  $ Embarked        : Factor w/ 3 levels "C","Q","S": 3 3 3 2 2 3 3 1 1 3 ...
##  $ AgeCat          : Factor w/ 4 levels "nens","adolescents",..: 3 3 3 3 3 3 2 3 3 3 ...
##  $ Survived        : Factor w/ 2 levels "0","1": 2 1 1 2 1 2 1 1 2 2 ...
```

Per poder aplicar la funció relativa a la construcció del model cal diferenciar entre les variables descriptives i la classe. Això ho fem seleccionant l'índex de la columna on es troba la classe (columna 7) i la resta (columnes de la 1 a 6). Els valors corresponents a la classe els guardem a la variable y_car i els corresponents a les variables descriptives a la x_car:

```
set.seed(1912)
y <- dataset_train_random[,"Survived"]
x <- dataset_train_random[,c("Pclass","Sex","FamilyCat","PriceCat","title","Embarked","AgeCat")]
```

Un cop obtinguda la mostra desordenada, anem a dividir el conjunt d edades en mostres d'entrenament i test. El conjunt d'entrenament serà aquell que s'utilitzarà per construir el model i el conjunt de proves el que s'utilitzarà per avaluar la seva precisió. Seguidament obtenim els diferents conjunts d'entrenament i test a partir de la proporció de casos que hem triat per cada conjunt (2/3 i 1/3) utilitzant la funció *sample*:

```
set.seed(1912)
indexes = sample(1:nrow(x), size=floor((2/3)*nrow(x)))
train_x <- x[indexes,]
train_y <- y[indexes]
test_x  <- x[-indexes,]
test_y  <- y[-indexes]
```

Per poder executar el mètode cal que no existeixin valors nul, els eliminem. Per això hem creat una funció que retorna els indexs de les files que contenen algun valor nul en un dataframe.

```
indices_with_na <- function(data) {
  iwn <- c()
  for (i in c(1:nrow(data))) {
    if (any(is.na(data[i,]))) {
      iwn <- c(iwn, i)
    }
  }
  return(iwn)
}
```

Comprovem utilitzant la funció has_na que no hi ha variables del dataset amb valors nul:

```
has_na(train_x)
```

```
## [1] "No hi ha cap variable amb valors nul"
```

```
has_na(test_x)
```

```
## [1] "No hi ha cap variable amb valors nul"
```

I que tant les variables descriptives com les corresponents a la classe tenen la mateixa mida:

```
nrow(train_x) == length(train_y)
```

```
## [1] TRUE
```

```
nrow(test_x) == length(test_y)
```

```
## [1] TRUE
```

Un cop preparades les variables que s'utilitzaran, creem el model utilitzant l'algoritme C5.0 de la llibreria C50 i observem un resum del model generat.

```
model_tree <- C50::C5.0(train_x, train_y, rules=TRUE)
summary(model_tree)
```

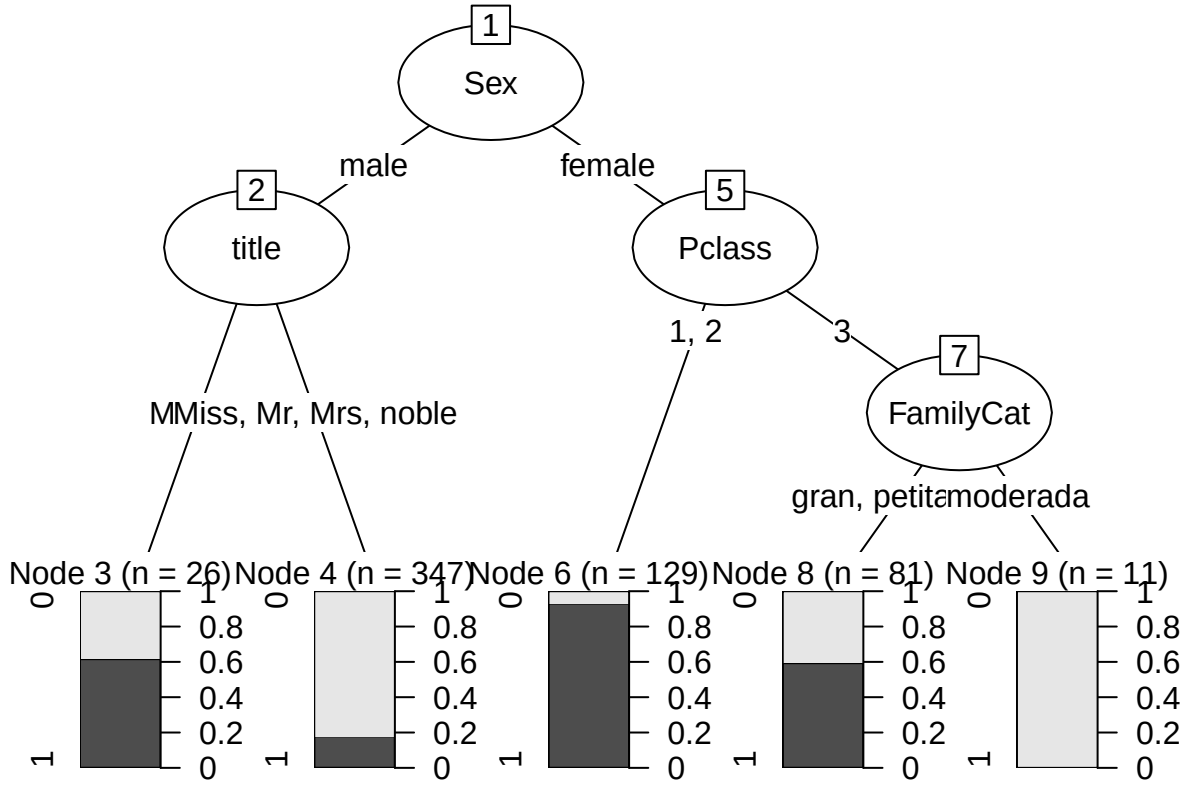
```
##
## Call:
## C5.0.default(x = train_x, y = train_y, rules = TRUE)
##
##
## C5.0 [Release 2.07 GPL Edition]      Tue Jan 05 22:28:41 2021
## -------------------------------
##
## Class specified by attribute `outcome'
```

```
##
## Read 594 cases (8 attributes) from undefined.data
##
## Rules:
##
## Rule 1: (21, lift 1.6)
##  Pclass = 3
##  FamilyCat = moderada
##  ->  class 0  [0.957]
##
## Rule 2: (347/62, lift 1.4)
##  Sex = male
##  title in {Mr, noble}
##  ->  class 0  [0.819]
##
## Rule 3: (129/9, lift 2.2)
##  Pclass in {1, 2}
##  Sex = female
##  ->  class 1  [0.924]
##
## Rule 4: (205/42, lift 1.9)
##  Sex = female
##  FamilyCat in {gran, petita, sol}
##  ->  class 1  [0.792]
##
## Rule 5: (26/10, lift 1.5)
##  title = Master
##  ->  class 1  [0.607]
##
## Default class: 0
##
##
## Evaluation on training data (594 cases):
##
##          Rules
##    ----------------
##      No      Errors
##
##       5  106(17.8%)   <<
##
##
##     (a)   (b)    <-classified as
##    ----  ----
##     304    44    (a): class 0
##      62   184    (b): class 1
##
##
##  Attribute usage:
##
##   93.77% Sex
##   62.79% title
##   38.05% FamilyCat
##   25.25% Pclass
##
```

```
##
## Time: 0.0 secs
```

Podem observar gràficament el model generat si el construïm sense mostrar les regles i el passem a la funció plot:

```
model_tree_for_plot <- C50::C5.0(train_x, train_y)
plot(model_tree_for_plot)
```



Avaluació del model

Amb el model d'arbre de decisió creat a partir de les dades del conjunt d'entrenament podem avaluar la seva eficiència intentant predir les dades que es troben al conjunt de prova. Per fer-ho utilitzem la funció *predict* de R que classifica els diferents registres del conjunt de prova (test_X) utilitzant el model que rep (model_tree).

```
predicted_test <- predict(model_tree,test_x, type="class")
```

Si observem el resultat d'aquesta funció veiem que es correspon a un vector amb les etiquetes de la classe. Per comprovar la precisió de l'arbre avaluem quantes d'aquestes etiquetes predites es corresponen al valor real que té el registre comparant-lo amb el contingut de la variable *test_y*:

```
precisio_model <- sum(predicted_test == test_y)/length(predicted_test)
print(paste("La precisió del model és de ", round(precisio_model*100,2), "%", sep=""))
```

```
## [1] "La precisió del model és de 82.15%"
```

Comentaris del model

Utilitzant el model basat en arbres de decisió hem obtingut una precisió de 82.15 %, la qual és un resultat bastant correcte i que demostra que realment existeix una influència de les variables descriptives a l'hora de

predir la classe. Tal i com hem pogut observar a partir del gràfic i el resum del model creat, aquest no ha utilitzat totes les variables ja que ha prescindit de la variable AgeCat.

També hem pogut veure que la variable que té més pes a l'hora de determinar el destí dels passatgers del titanic és el sexe, el qual té un pes del 93.77 % en el model creat. El títol nobiliari del passatger també guarda una forta relació amb el model, mantenint un 62.79 %. Finalment trobem que la dimensió de la família i classe dels tiquets també tenen una influència considerable però menor respecte les dues anteriors comentades.

5.2. Support Vector Machine

Construcció del model

Per construir un model basat en SVM utilitzarem les mateixes dades transformades que en l'apartat anterior ja que al igual que els arbres de decisió, un model basat en svm utilitza també variables categòriques.

A l'hora de cridar la funció encarregada de crear el model cal que tan la classe com les variables descriptives es trobin en el mateix dataframe així que fusionem la classe i les variables descriptives i comprovem que s'han fusionat correctament:

```
train_x["Fate"] <- train_y
str(train_x)
```

```
## 'data.frame':    594 obs. of  8 variables:
##  $ Pclass   : Factor w/ 3 levels "1","2","3": 3 3 3 3 3 1 3 1 1 2 ...
##  $ Sex      : Factor w/ 2 levels "female","male": 1 2 2 2 2 1 2 1 1 2 ...
##  $ FamilyCat: Factor w/ 4 levels "gran","moderada",..: 2 4 4 3 2 3 4 3 3 4 ...
##  $ PriceCat : Factor w/ 3 levels "baix","mig","alt": 1 1 2 1 1 2 1 3 3 2 ...
##  $ title    : Factor w/ 5 levels "Master","Miss",..: 4 3 3 3 1 4 3 4 4 3 ...
##  $ Embarked : Factor w/ 3 levels "C","Q","S": 3 3 3 3 3 3 3 3 3 3 ...
##  $ AgeCat   : Factor w/ 4 levels "nens","adolescents",..: 3 3 3 2 1 3 3 3 3 2 ...
##  $ Fate     : Factor w/ 2 levels "0","1": 1 2 1 1 1 2 1 2 1 1 ...
```

Seguidament ja podem crear el model, per la qual cosa és necessari especificar a priori la classe i les variables descriptives que intervenen en el procés. Per la creació del model basat en Support Vector Machine utilitzem la funció *svm* del paquet *E1071*.

```
SVM_model.train <- svm(Fate ~ Pclass + Sex + Embarked + title + FamilyCat + AgeCat, data = train_x, prob
```

Un cop creat el model, n'observem un resum utilitzant la funció *summary*:

```
summary(SVM_model.train)
```

```
##
## Call:
## svm(formula = Fate ~ Pclass + Sex + Embarked + title + FamilyCat +
##     AgeCat, data = train_x, probability = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  290
##
##  ( 147 143 )
##
##
```

```
## Number of Classes:  2
##
## Levels:
##  0 1
```

Avaluació del model

Un cop creat el model procedim a avaluar-lo predint els casos de test. Per fer-ho, al igual que en l'apartat anterior fem ús de la funció *predict*, la qual rep el model creat i els registres dels casos que es volen predir.

```
SVM_pred.test <- predict(SVM_model.train, test_x)
```

Una manera de visualitzar els resultats és utilitzant una matriu de confusió. La funció *CrossTable* de la llibreria *gmodels* ofereix una manera simple de presentar els resultats utilitzant una matriu de confusió però amb més detall:

```
CrossTable(test_y, SVM_pred.test, prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE, dnn = c("Realitat
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  297
##
##
##              | Predicció
##     Realitat |         0 |         1 | Row Total |
## -------------|-----------|-----------|-----------|
##            0 |       176 |        25 |       201 |
##              |     0.593 |     0.084 |           |
## -------------|-----------|-----------|-----------|
##            1 |        25 |        71 |        96 |
##              |     0.084 |     0.239 |           |
## -------------|-----------|-----------|-----------|
## Column Total |       201 |        96 |       297 |
## -------------|-----------|-----------|-----------|
##
##
```

Per avaluar la precisió del model utilitzem una matriu de confusió simple com en l'apartat anterior:

```
matriu_conf_svm <- table(test_y,Predicted=SVM_pred.test)
precisio_model_matriu_conf <- sum(diag(matriu_conf_svm)) / sum(matriu_conf_svm)
print(paste("La precisió del model és de ", round(precisio_model_matriu_conf*100,2), "%", sep=""))
```

```
## [1] "La precisió del model és de 83.16%"
```

Comentaris del model

Utilitzant el model basat en SVM obtenim una precisió del 83.16 %, lleugerament millor que el que s'ha obtingut utilitzant arbres de decisió.

5.3. Regressió logística

Construcció del model

Un altre mètode per analitzar les dades és aplicant una regressió logística, ja que aquesta ens permet realitzar una classificació a partir de la descripció d'un registre. Per crear el model podem utilitzar la funció *glm*, on cal especificar la variable a predir (*Fate*), les variables descriptives i el dataframe que s'utilitza. També cal especificar que la classe que es vol predir consta de nomes dos valors establint el valor *binomial* al paràmetre *family*.

```
model_glm <- glm(formula=Fate~Pclass+Sex+FamilyCat+PriceCat+title+Embarked+AgeCat,data=train_x,family=bi
```

Un cop creat el model podem visualitzar-ne la seva descripció utilitzant la funció *summary*:

```
summary(model_glm)
```

```
##
## Call:
## glm(formula = Fate ~ Pclass + Sex + FamilyCat + PriceCat + title +
##     Embarked + AgeCat, family = binomial(link = logit), data = train_x)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6227  -0.5517  -0.4105   0.6065   2.3356
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)        17.0780   723.7232   0.024 0.981174
## Pclass2            -0.4093     0.5539  -0.739 0.459984
## Pclass3            -0.8393     0.5717  -1.468 0.142080
## Sexmale           -15.8492   723.7226  -0.022 0.982528
## FamilyCatmoderada  -1.1502     0.8344  -1.379 0.168040
## FamilyCatpetita     1.6594     0.6940   2.391 0.016806 *
## FamilyCatsol        1.8880     0.7162   2.636 0.008389 **
## PriceCatmig         0.2328     0.4413   0.528 0.597824
## PriceCatalt         1.2238     0.5390   2.270 0.023186 *
## titleMiss         -16.4153   723.7229  -0.023 0.981904
## titleMr            -3.7152     0.7746  -4.796 1.62e-06 ***
## titleMrs          -15.9900   723.7230  -0.022 0.982373
## titlenoble         -3.9943     1.0658  -3.748 0.000179 ***
## EmbarkedQ          -0.6260     0.5041  -1.242 0.214316
## EmbarkedS          -0.4867     0.3104  -1.568 0.116871
## AgeCatadolescents  -0.5405     0.6420  -0.842 0.399858
## AgeCatadults       -0.3678     0.6461  -0.569 0.569195
## AgeCatancians      -1.8602     0.8827  -2.107 0.035075 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 805.86  on 593  degrees of freedom
## Residual deviance: 493.45  on 576  degrees of freedom
## AIC: 529.45
##
## Number of Fisher Scoring iterations: 14
```

Avaluació del model

Per avaluar el model tornem a utilitzar els registres de la variable *test_x* i els passem a la funció *predict*:

```
probs_prediction_glm <- predict(model_glm, newdata=test_x, type="response")
head(probs_prediction_glm)
```

```
##         28        251        157        217         16        360
## 0.03671277 0.09177948 0.63297456 0.70201595 0.87491594 0.67209419
```

Com podem veure la predicció d'aquest model retorna la probabilitat de que els registre pertanyin o no a la variable dictòmica que es vol predir, motiu pel qual cal realitzar una conversió als valors 0 i 1 abans de poder comparar els resultats predits amb els reals de la variable test_y:

```
prediction_glm <- c()
for (i in probs_prediction_glm) {
  if (i<0.5)
    prediction_glm <- c(prediction_glm,0)
  else
    prediction_glm <- c(prediction_glm,1)
}
```

Un cop obtinguts els valors predits amb el mateix format dels valors reals, procedim a crear una matriu de confusió per veure com s'han distribuït els encerts i les errades:

```
CrossTable(test_y, prediction_glm, prop.chisq = FALSE, prop.c = FALSE, prop.r = FALSE, dnn = c("Realita
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  297
##
##
##              | Predicció
##     Realitat |         0 |         1 | Row Total |
## -------------|-----------|-----------|-----------|
##            0 |       174 |        27 |       201 |
##              |     0.586 |     0.091 |           |
## -------------|-----------|-----------|-----------|
##            1 |        21 |        75 |        96 |
##              |     0.071 |     0.253 |           |
## -------------|-----------|-----------|-----------|
## Column Total |       195 |       102 |       297 |
## -------------|-----------|-----------|-----------|
##
##
```

Per obtenir una mesura de la bondat del model apliquem el mateix procediment que el que hem vist en els dos subapartats anteriors, fent ús de la matriu de confusió:

```
matriu_conf_glm <- table(test_y,Predicted=prediction_glm)
precisio_model_matriu_conf_glm <- sum(diag(matriu_conf_glm)) / sum(matriu_conf_glm)
print(paste("La precisió del model és de ", round(precisio_model_matriu_conf_glm*100,2), "%", sep=""))
```

```
## [1] "La precisió del model és de 83.84%"
```

Comentaris del model

Utilitzant la regressió logística obtenim una precisió del 83.84 %, lleugerament superior a la resta de modelitzacions provades. Observant el resum del model podem veure com les categòries que més influèixen en la decisió de la classe d'un passatger utilitzant aquesta modelització és quan el títol del passatger és noble o simplement Mr (ja que indica que no poseeix cap títol nobiliari i pertany al gènere masculí). També influeix que el passatger viatgi sol o amb poca companyia i el fet de si és o no ancià.

6. Predicció

Arribats a aquest punt hem volgut presentar-hi els nostres resultats a la competició activa de la pàgina Kaggle. Per fer-ho hem predit els registres corresponents al fitxer de test (*test.csv*) utilitzant el model que ens ha donat millors resultats: el model basat en la regressió logística.

```
# Obtenim les probabilitats de classe dels registes corresponents a l'apartat de test
probs_prediction_glm_competition <- predict(model_glm, newdata=dataset_test, type="response")
# Convertim les probabilitats en predicció
prediction_glm_comp <- c()
for (i in probs_prediction_glm_competition) {
  if (i<0.5)
    prediction_glm_comp <- c(prediction_glm_comp,0)
  else
    prediction_glm_comp <- c(prediction_glm_comp,1)
}
```

Un cop obtinguda la predicció la guardem en un dataframe amb l'identificador del passatger:

```
df_competition <- data.frame(dataset_test$PassengerId,prediction_glm_comp)
colnames(df_competition) <- c("PassengerId", "Survived")
```

Finalment generem el fitxer csv que conté la predicció:

```
write.csv(df_competition, "prediction_comp.csv", row.names=FALSE, quote=FALSE)
```

7. Conclusions

Taula 5: Tots els membres han contribuït de manera igualitaria en l'elaboració de la pràctica.

	Contribucions	Signa
	Investigació prèvia	AAS & ATP
	Redacció de les respostes	AAS & ATP
	Desenvolupament codi	AAS & ATP

8. Bibliografia

- Gibergans, J. (2019). Regressió lineal simple. Editorial UOC.
- Gibergans, J. (2019). Regressió lineal múltiple. Editorial UOC.
- Guillén, M., Alonso, M. T. (2019). Models de regressió logística. Editorial UOC.
- Liviano, D., Pujol, M. (2019). Models de regressió i anàlisi multivariant amb R-Commander. Editorial UOC.