# PRACTICA 2

*Laura Guzman*

*28 maig de 2019*

- PRÀCTICA 2
- 1. Descripció del dataset. Perquè és important i quina pregunta/problema pretén respondre?
- 2. Integració i selecció de les dades d'interès a analitzar.
- 3. Neteja de les dades.
    - 3.1. Les dades contenen zeros o elements buits? Com gestionaries aquests casos?
    - 3.2. Identificació i tractament de valors extrems.
- 2B. Creem variables
- 4. Anàlisi de les dades.
    - 4.0 Anàlisi descriptiu de les dades
    - 4.1. Selecció dels grups de dades que es volen analitzar/comparar (planificació dels anàlisis a aplicar).
    - 4.2. Comprovació de la normalitat i homogeneïtat de la variància.
    - 4.3. Aplicació de proves estadístiques per comparar els grups de dades. En funció de les dades i de l'objectiu de l'estudi, aplicar proves de contrast d'hipòtesis,correlacions, regressions, etc. Aplicar almenys tres mètodes d'anàlisi diferents.
        - 4.3.1Contrast d'hipòtesis
        - 4.3.2 Regressió logarítmica:
        - 4.3.3 Arbre
    - 5. Representació dels resultats a partir de taules i gràfiques.
    - 6. Resolució del problema. A partir dels resultats obtinguts, quines són les conclusions? Els resultats permeten respondre al problema?
- 7. Codi: Cal adjuntar el codi, preferiblement en R, amb el que s'ha realitzat la neteja, anàlisi i representació de les dades. Si ho preferiu, també podeu treballar en Python.

# PRÀCTICA 2

# 1. Descripció del dataset. Perquè és important i quina pregunta/problema pretén respondre?

El dataset conté les dades de la supervivència dels passatgers del titànic segons els atributs dels passatgers.

Volem saber la probabilitat de supervivència segons al grup que pertanyia el passatger. En particular, volem predir quins passatgers van sobreviure.

Primer de tot carreguem les dades. Les dades les tenim separades en dos: el conjunt d'entrenament i el conjunt de test.

```
#Carreguem arxius de dades
train <-read.csv(file="C:/Users/Laura/Desktop/UOC/Tipologia i cicle de vida de les dades/PRAC 2 Neteja i validació/titanic/train.csv", header=TRUE, sep=",",stringsAsFactors = FALSE)

test <-read.csv(file="C:/Users/Laura/Desktop/UOC/Tipologia i cicle de vida de les dades/PRAC 2 Neteja i validació/titanic/test.csv", header=TRUE, sep=",",stringsAsFactors = FALSE)
```

Comencem per descriure el dataset train:

```
#Resum dades train

#Dimensió train
dim(train)
```

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

```
# Resum train
summary(train)
```

```
##   PassengerId       Survived          Pclass          Name
##  Min.   :  1.0   Min.   :0.0000   Min.   :1.000   Length:891
##  1st Qu.:223.5   1st Qu.:0.0000   1st Qu.:2.000   Class :character
##  Median :446.0   Median :0.0000   Median :3.000   Mode  :character
##  Mean   :446.0   Mean   :0.3838   Mean   :2.309
##  3rd Qu.:668.5   3rd Qu.:1.0000   3rd Qu.:3.000
##  Max.   :891.0   Max.   :1.0000   Max.   :3.000
##
##      Sex                 Age            SibSp           Parch
##  Length:891         Min.   : 0.42   Min.   :0.000   Min.   :0.0000
##  Class :character   1st Qu.:20.12   1st Qu.:0.000   1st Qu.:0.0000
##  Mode  :character   Median :28.00   Median :0.000   Median :0.0000
##                     Mean   :29.70   Mean   :0.523   Mean   :0.3816
##                     3rd Qu.:38.00   3rd Qu.:1.000   3rd Qu.:0.0000
##                     Max.   :80.00   Max.   :8.000   Max.   :6.0000
##                     NA's   :177
##     Ticket               Fare           Cabin             Embarked
##  Length:891         Min.   :  0.00   Length:891         Length:891
##  Class :character   1st Qu.:  7.91   Class :character   Class :character
##  Mode  :character   Median : 14.45   Mode  :character   Mode  :character
##                     Mean   : 32.20
##                     3rd Qu.: 31.00
##                     Max.   :512.33
##
```

```
# Tipus de dades train
str(train)
```

```
## 'data.frame':    891 obs. of  12 variables:
##  $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Survived   : int  0 1 1 1 0 0 0 0 1 1 ...
##  $ Pclass     : int  3 1 3 1 3 3 1 3 3 2 ...
##  $ Name       : chr  "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley (Flore
nce Briggs Thayer)" "Heikkinen, Miss. Laina" "Futrelle, Mrs. Jacques Heath (Lily Ma
y Peel)" ...
##  $ 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" ...
```

Veiem que el conjunt de train compta amb 891 mostres que tenen 12 atributs. Aquests 12 atributs són:

-PassengerId, identifica el passetger, enter.

-Survived, classe a predir, 0 és igual a No. Hauria de ser un factor i apareix com enter

-Pclass, classe de tiquet ( primera, segona o tercera), també hauria de ser un factor.

-Name, nom del passatger, caràcter, no és rellevant.

-Sex, sexe. És un caràcter i hauria de ser un factor.

-Age, edat. També podria ser un factor o es podria posar en intervals.

-SibSp (nombre de germans/conjugues a bord), enter.

-Parch (nombre de pares/fills a bord), enter.

-Ticket (número de tiquet), caràcter, irrellevant.

-Fare (tarifa), nombre.

-Cabin (número de cabina), caràcter, irrellevant.

-Embarked (port embarcació:C = Cherbourg, Q = Queenstown, S = Southampton).És un caràcter i hauria de ser factor.

Anem a descriure el conjunt de test:

```
#Resum dades test

#Dimensió test
dim(test)
```

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

```
# Resum test
summary(test)
```

```
##   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
```

```
# Tipus de dades test
str(test)
```

```
## '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)" "Myle
s, Mr. Thomas Francis" "Wirz, Mr. Albert" ...
##  $ 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" ...
```

Veiem que el conjunt de test compta amb 418 mostres que tenen 11 atributs (en aquest cas no tenim el valor “Survived”, que és el que predim). Aquests 11 atributs són: PassengerId, Pclass, Name, Sex, Age, SibSp, Parch, Ticket, Fare, Cabin i Embarked. Aquest conjunt de dades només l’utilitzarem per fer la predicció.

# 2. Integració i selecció de les dades d'interès a analitzar.

Primer de tot, discretitzem les variables que haurien de ser factor i són un altre tipus.

```
# Per a quines variables tindria sentit un procés de discretització?
apply(train,2, function(x) length(unique(x)))
```

```
## PassengerId    Survived      Pclass        Name         Sex         Age
##         891           2           3         891           2          89
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##           7           7         681         248         148           4
```

Té sentit discretitzar les variables Survived, Pclass, Sex i Embarked (que són les que ja havíem mencionat al descriure les dades.)

```
# Discretitzem les variables amb poques classes
cols<-c("Survived","Pclass","Sex","Embarked")
for (i in cols){
  train[,i] <- as.factor(train[,i])
}
#En test

cols<-c("Pclass","Sex","Embarked")
for (i in cols){
  test[,i] <- as.factor(test[,i])
}
```

```
#Carreguem llibreries
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.5.3
```

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.5.3
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

Tractem els dos fitxers (train i test) junts, ja que si els tractem per separat llavors podriem estar assignant valors de manera diferent a cada grup i estar canviant la mostra. Diferenciem els registres de test de train a través del valor de Survived

```
#Creem variable Survived a test
test$Survived <- NA

#Combinem
total <- bind_rows(train,test)
```

```
## Warning in bind_rows_(x, .id): Unequal factor levels: coercing to character
```

```
## Warning in bind_rows_(x, .id): binding character and factor vector,
## coercing into character vector

## Warning in bind_rows_(x, .id): binding character and factor vector,
## coercing into character vector
```

Per a l'estudi, utilitzarem les variables Survived (variable a predir) i PClass, Sex, Age, SibSp, Parch, Fare i Embarked. Descartem PassengerId ja que no és cap característica, també descartem el nom ja que el nom ens diu si és home o dona o nen, però això també ho podem saber a través de l'edat i el sexe, descartem ticket i cabin ja que son variables que tenen valors molt dispersos (per exemple eb cabin hi ha moltes cabines diferents i la seva numeració no segueix cap tipus de relació lineal) i que no ens aporten molt. El que si que utilitzardm de cabin serà la lletra de la coberta.

```
#Treiem variables que no utilitzarem
total <- select(total, -Name, -Ticket) #, -PassengerId no el treiem perquè el necess
item pels resultats de test per penjar

#Comprovem
summary(total)
```

```
##   PassengerId   Survived   Pclass      Sex           Age
##  Min.   :   1   0   :549   1:323   female:466   Min.   : 0.17
##  1st Qu.: 328   1   :342   2:277   male  :843   1st Qu.:21.00
##  Median : 655   NA's:418   3:709                Median :28.00
##  Mean   : 655                                   Mean   :29.88
##  3rd Qu.: 982                                   3rd Qu.:39.00
##  Max.   :1309                                   Max.   :80.00
##                                                 NA's   :263
##      SibSp            Parch            Fare            Cabin
##  Min.   :0.0000   Min.   :0.000   Min.   :  0.000   Length:1309
##  1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:  7.896   Class :character
##  Median :0.0000   Median :0.000   Median : 14.454   Mode  :character
##  Mean   :0.4989   Mean   :0.385   Mean   : 33.295
##  3rd Qu.:1.0000   3rd Qu.:0.000   3rd Qu.: 31.275
##  Max.   :8.0000   Max.   :9.000   Max.   :512.329
##                                   NA's   :1
##    Embarked
##  Length:1309
##  Class :character
##  Mode  :character
##
##
##
##
```

```
head(total)
```

```
##   PassengerId Survived Pclass    Sex Age SibSp Parch    Fare Cabin
## 1           1        0      3   male  22     1     0  7.2500
## 2           2        1      1 female  38     1     0 71.2833   C85
## 3           3        1      3 female  26     0     0  7.9250
## 4           4        1      1 female  35     1     0 53.1000  C123
## 5           5        0      3   male  35     0     0  8.0500
## 6           6        0      3   male  NA     0     0  8.4583
##   Embarked
## 1        S
## 2        C
## 3        S
## 4        S
## 5        S
## 6        Q
```

# 3. Neteja de les dades.

## 3.1. Les dades contenen zeros o elements buits? Com gestionaries aquests casos?

```
#Mirem si conté valors buits
colSums(is.na(total))
```

```
## PassengerId    Survived      Pclass         Sex         Age       SibSp
##           0         418           0           0         263           0
##       Parch        Fare       Cabin    Embarked
##           0           1           0           0
```

```
colSums(total=="")
```

```
## PassengerId    Survived      Pclass         Sex         Age       SibSp
##           0          NA           0           0          NA           0
##       Parch        Fare       Cabin    Embarked
##           0          NA        1014           2
```

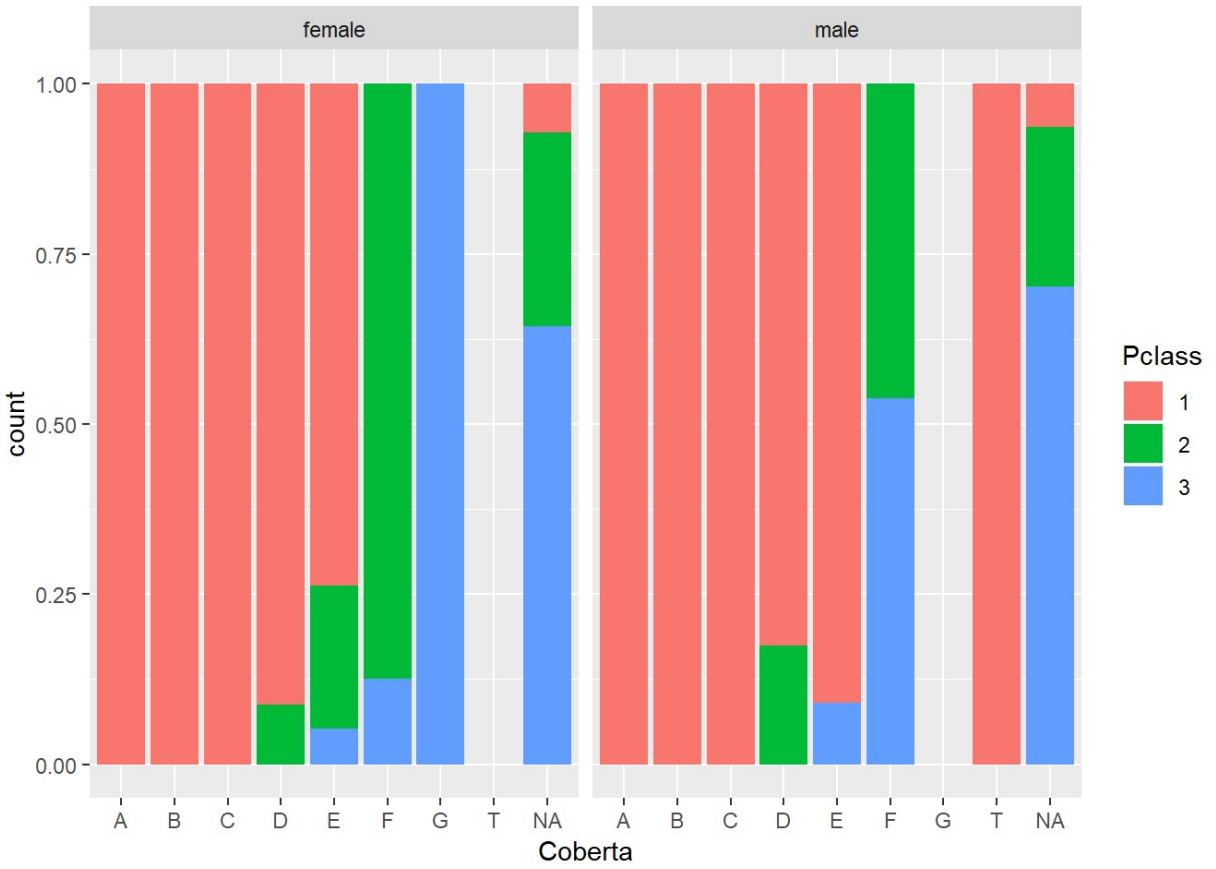
Les variables que contenen valors buits són Age , Embarked, Fare i Cabin. Els valors buits de Survived són els registres de test.

Creem variable coberta i eliminem cabin i tractem valors buits de coberta Coberta és la lletra de cabin

```
total$Coberta<-factor(sapply(total$Cabin, function(x) {strsplit(x, NULL)[[1]][1]}))
total <- select(total, -Cabin)
```

Mirem valors coberta segons pclass

```
ggplot(total,aes(x=Coberta,fill=Pclass))+geom_bar(position='fill')+facet_wrap(~Sex)
```




Veiem que els de primera classe anaven a A,B,C i T íntegrament i compartien D i E. Veiem que els de segona classe compartien D , E i F. Veiem que els de tercera classe estaven a G i compartien F i E.

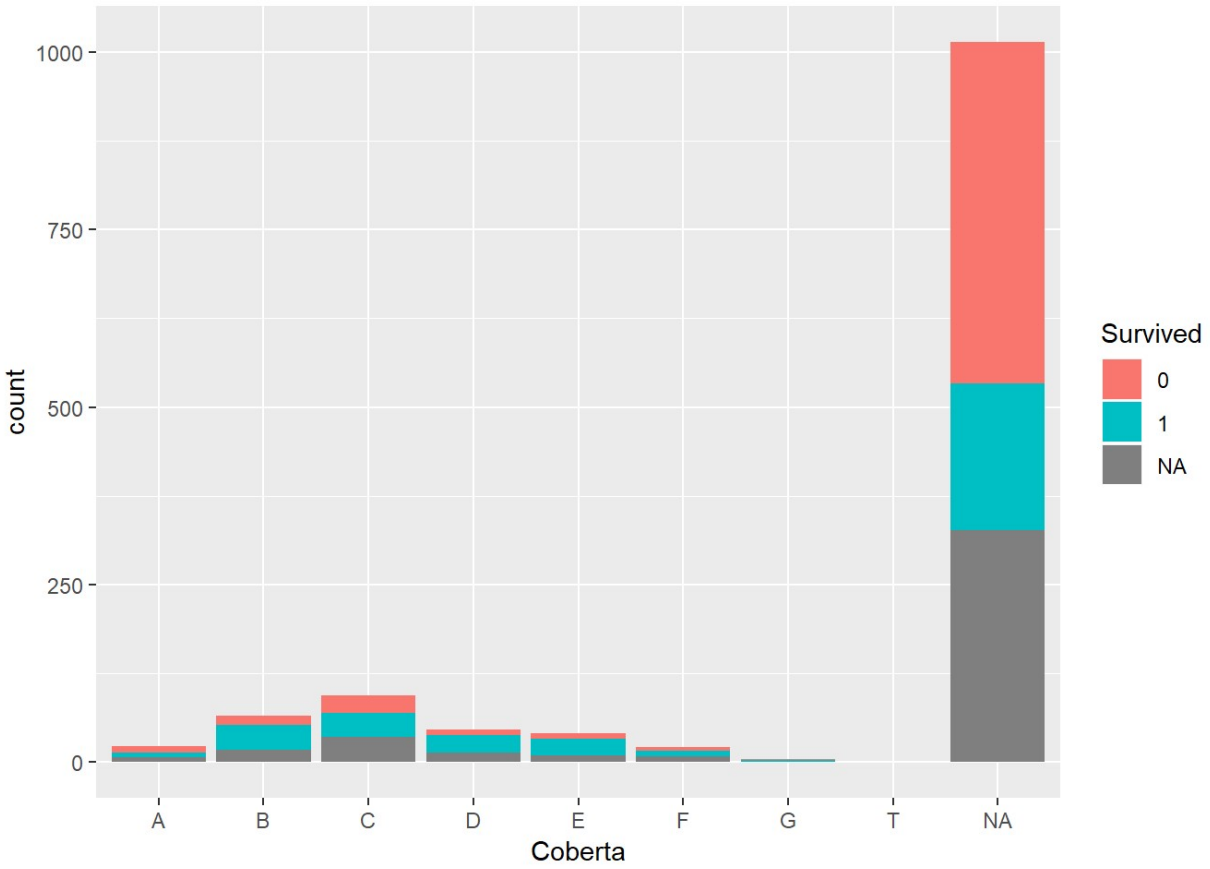
Veiem que a T no hi havia dones i a G no hi havia homes.

Veiem que les dones de primera classe estan a A,B, C, D i E en més d'un 50% Veiem que les dones de segona classe estan a F en més d'un 50% Veiem que les dones de tercera classe estan a G en més d'un 50%.

Veiem que els homes de primera classe estan a A,B,C,D,E en més d'un 50%. Veiem que els homes de segona classe estan a F en un 50%. Veiem que els homes de tercera classe estan a F en un 50%.

És dificil imputar una lletra segons la classe i mirem segons embarked i tampoc, de manera que ho tractem posant NA i creant un nou grup i utilitzant addna:

```
total$Coberta[total$Coberta==''']<-NA
total$Coberta <- factor(total$Coberta)
total$Coberta <- addNA(total$Coberta)

#Visualitzem;

ggplot(data=total,aes(x=Coberta,fill=Survived))+geom_bar()
```




Pels valors buits d'embarked, primer comprovem quins són:

```
#Comprovem quins son els valors buits d'embarked
total[total$Embarked=="",]
```

```
##     PassengerId Survived Pclass    Sex Age SibSp Parch Fare Embarked
## 62           62        1      1 female  38     0     0   80
## 830         830        1      1 female  62     0     0   80
##     Coberta
## 62        B
## 830       B
```

```
table(total$Embarked)
```

```
##
##       C   Q   S
##   2 270 123 914
```

```
table(total$Survived,total$Embarked)
```

```
##
##           C   Q   S
##   0   0  75  47 427
##   1   2  93  30 217
```

```
table(total$Sex,total$Embarked)
```

```
##
##                C   Q   S
##   female   2 113  60 291
##   male     0 157  63 623
```

```
table(total$Pclass,total$Embarked)
```

```
##
##           C   Q   S
##   1   2 141   3 177
##   2   0  28   7 242
##   3   0 101 113 495
```

Veiem que de 464 dones, 113 embarquen a C, 60 a Q i 291 a S.

Veiem que de 321 persones de primera classe, 141 embarquen a C, 3 a Q i 177 a S.

Veiem que de 340 que son supervivents, 93 embarquen a C, 30 a Q i 217 a S.

Com que els valors buits d'embarked són dones de primera classe i supervivents, llavors els hi assignarem el valor S ja que és el que té més freqüència en el seu cas.

```
# Prenem el valor "S" pels valors buits de la variable "Embarked"
total$Embarked<- as.character(total$Embarked)# El posem numeric perquè no quedi el f
actor "" (buit)
total$Embarked[total$Embarked==""]<-"S"
total$Embarked<- as.factor(total$Embarked) #El tornem a factor
```

Fare: Com que veiem que el registre que li falta fare és de classe 4 i va embarcar a S, utilitzem només els registres que compleixen aquestes condicions per fer la mitjana i assignar-la al valor de fare buit.

```
# Comprovem el valor de Fare que falta
total[is.na(total$Fare) ,]
```

```
##      PassengerId Survived Pclass  Sex  Age SibSp Parch Fare Embarked
## 1044        1044     <NA>      3 male 60.5     0     0   NA        S
##      Coberta
## 1044    <NA>
```

```
#Fare, posem la mitjana segons pclass =3, embarked=S i el sexe= male
total$Fare[is.na(total$Fare) ] <- mean(total$Fare[total$Pclass=='3' & total$Embarked
=='S' & total$Sex=='male'],na.rm=T)
```

Pels valors buits de age, en un principi voliem posar la mitjana d'edat segons el sexe, però al haver-hi tants valors buits, hem preferit utilitzar la funció k Nearest Neighbor

```
#Age assignem valor per knn

#Fem grups de train ( valors que no son buits de Age) i test ( valors buits d'Age)
i els valors resultat d'edat de train

age_nobuit<-total[!is.na(total$Age),]
age_buit<-total[is.na(total$Age),]
age_solucio<-age_nobuit$Age


#Embarked, sex , pclass i Coberta els binaritzem
library(fastDummies)
```

```
## Warning: package 'fastDummies' was built under R version 3.5.3
```

```
age_nobuit<- fastDummies::dummy_cols(age_nobuit, select_columns =c("Embarked", "Se
x", "Pclass"))
age_buit<- fastDummies::dummy_cols(age_buit, select_columns =c("Embarked", "Sex", "P
class"))

#Eliminem les variables originals que no estan binaritzades
age_nobuit<-select(age_nobuit,-Survived,-Age,-Embarked,-Sex,-Pclass,-Coberta)
age_buit<-select(age_buit,-Survived,-Age,-Embarked,-Sex,-Pclass,-Coberta)


library(class)
```

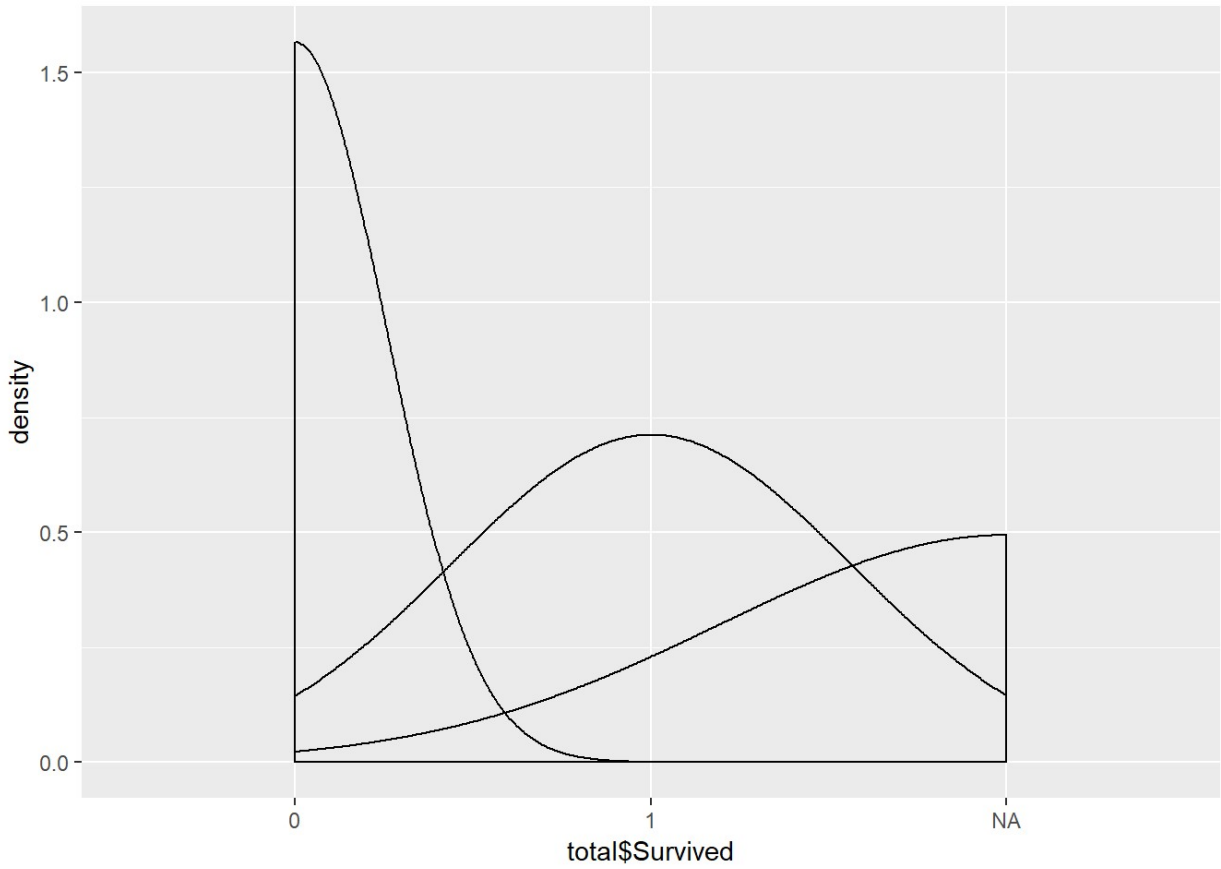
```
## Warning: package 'class' was built under R version 3.5.3
```

```
#Utilitzem com a k l'arrel del nombre de registres de la mostra i apliquem knn
ageknn<-knn(age_nobuit,age_buit,age_solucio,k=36) #ageknn seran les edats que ens fa
lten


#Assignem les edats resultants als valors nuls
total$Age[is.na(total$Age)]<-ageknn

#Si haguéssim volgut assignar als valors buits de Age la mitjana segons el sexe, hag
uéssim fet:

    # Agafem la mitjana segons el sexe pels valors buits de la variable "Age"
    #total$Age[is.na(total$Age) & total$Sex=='female' ] <-   mean (total$Age [total
$Sex=='female'], na.rm=T)

    #total$Age[is.na(total$Age) & total$Sex=='male' ] <- mean(total$Age[total$Sex=
='male'], na.rm=T)
```

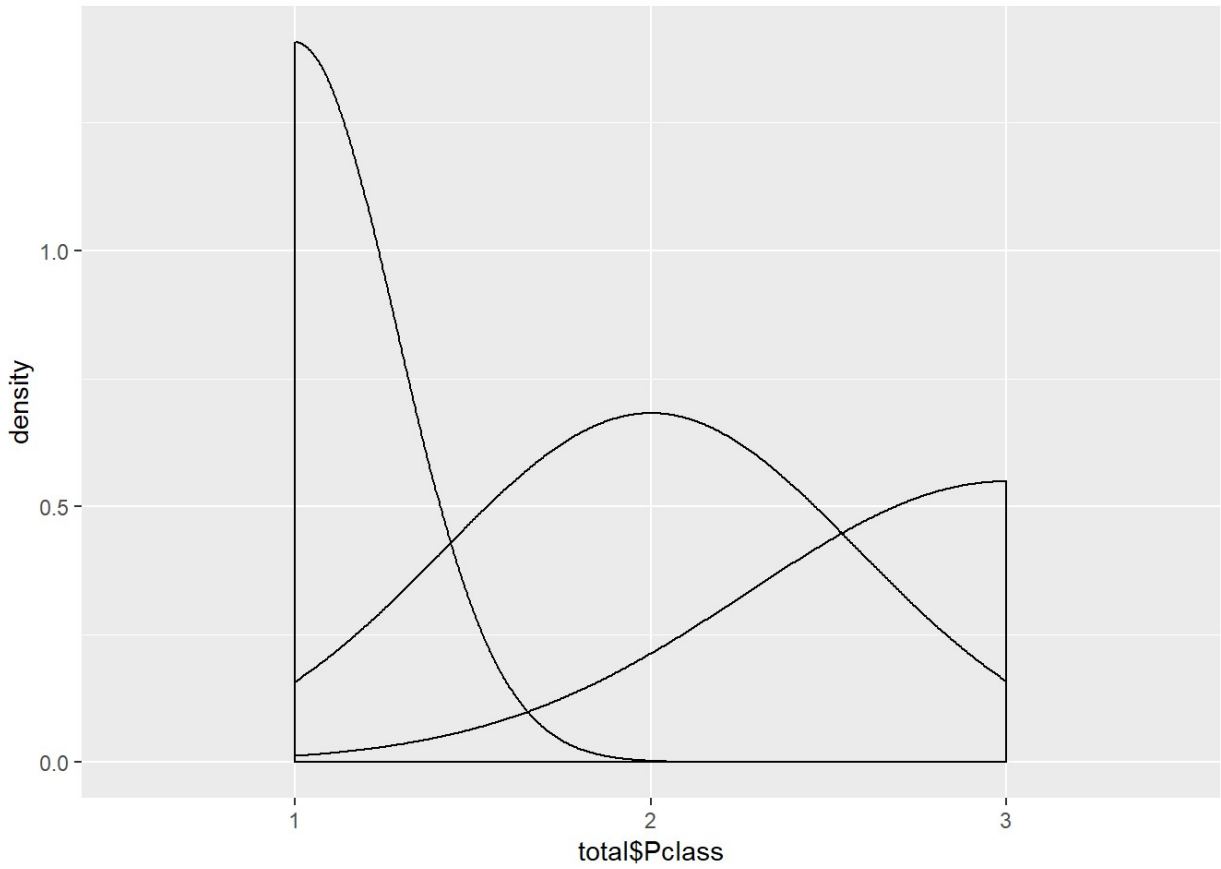
# 3.2. Identificació i tractament de valors

# extrems.

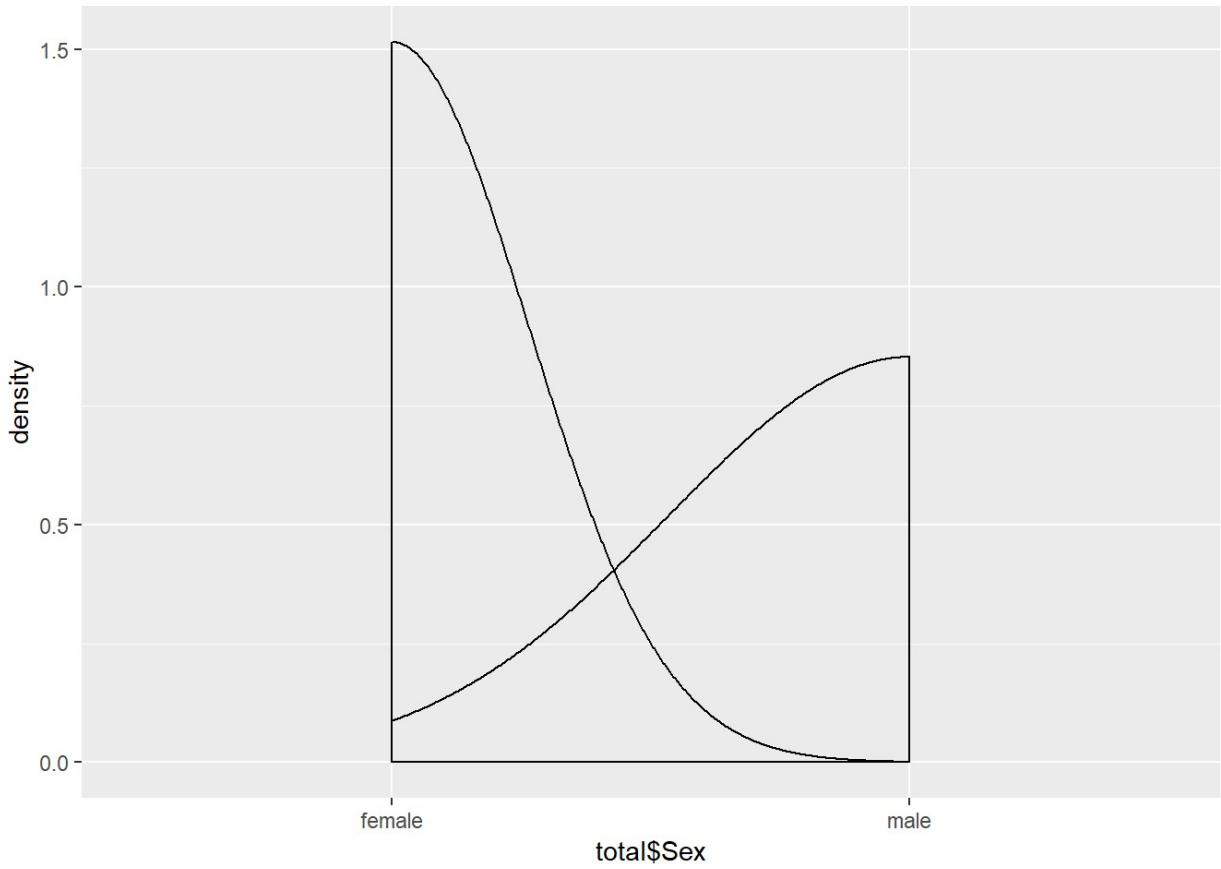
```
#Primer de tot busquem si hi ha valors sentinella (valors extrems per representar un
a situació especial)

ggplot(mapping= aes(x=total$Survived))+ geom_density()
```



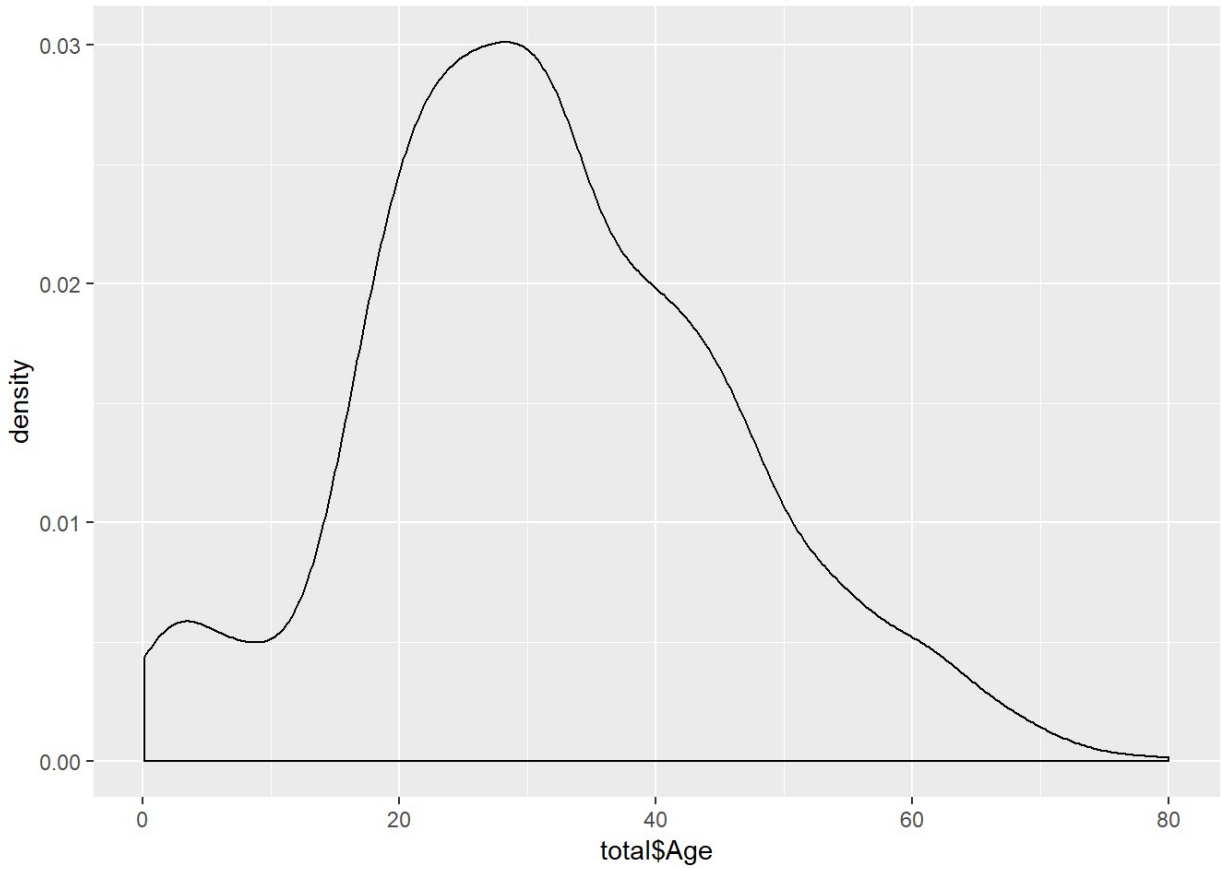

```
ggplot(mapping= aes(x=total$Pclass))+ geom_density()
```



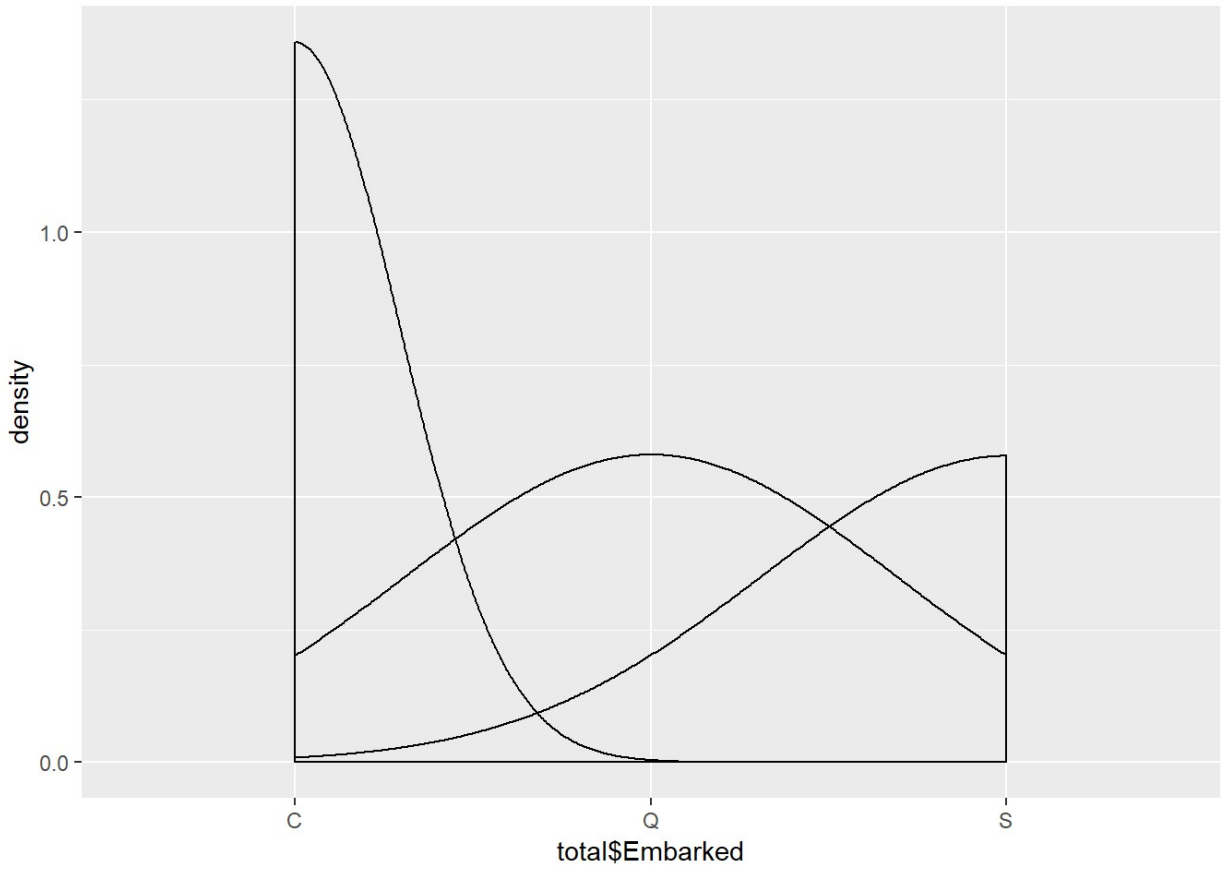

```
ggplot(mapping= aes(x=total$Sex))+ geom_density()
```



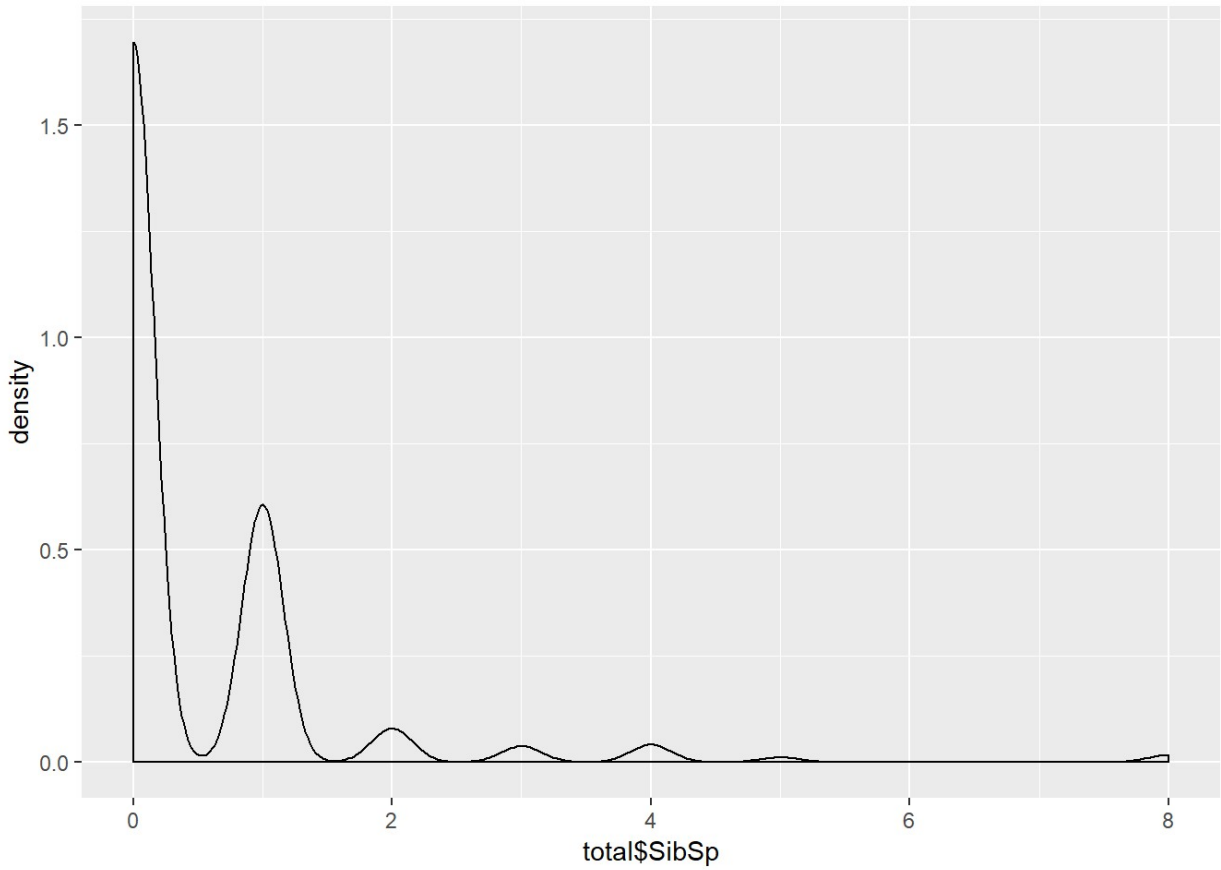
```
ggplot(mapping= aes(x=total$Age))+ geom_density()
```



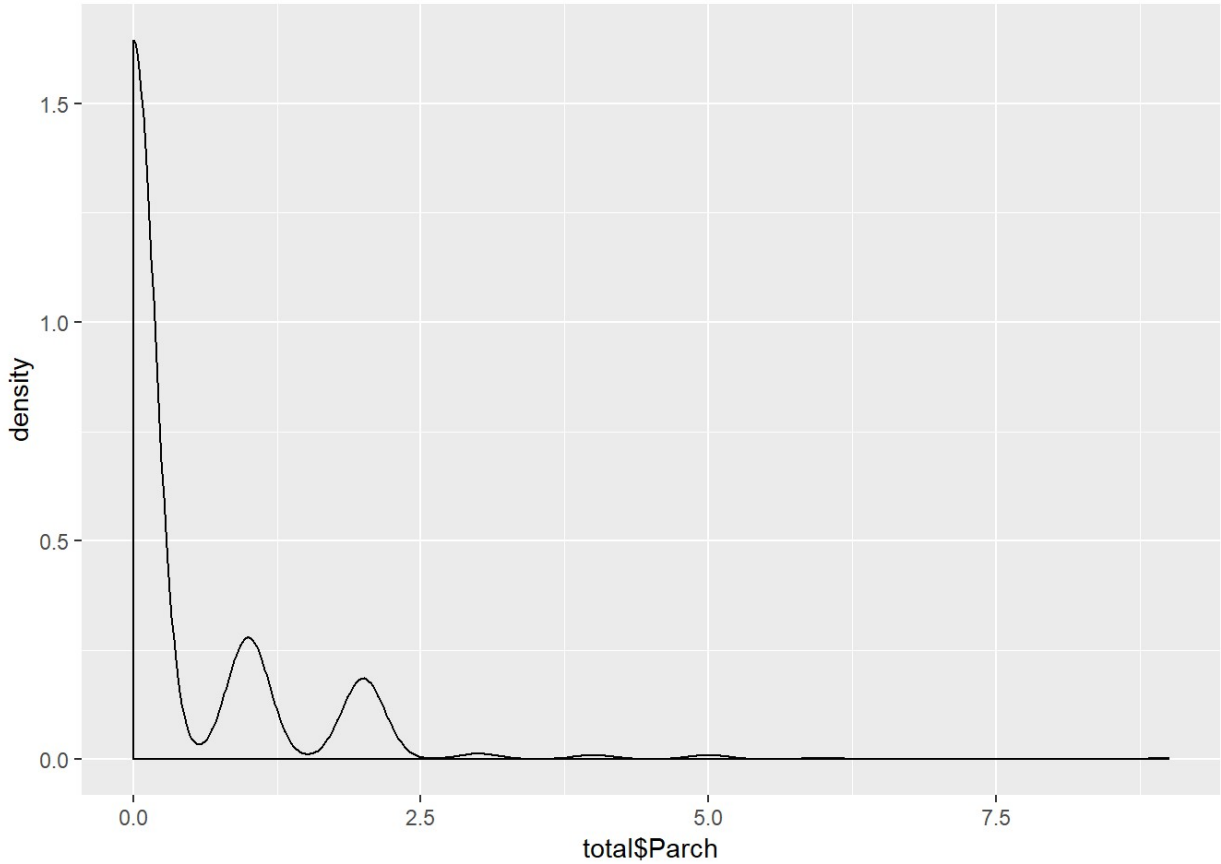

```
ggplot(mapping= aes(x=total$Embarked))+ geom_density()
```




```
ggplot(mapping= aes(x=total$SibSp))+ geom_density()
```



```
ggplot(mapping= aes(x=total$Parch))+ geom_density()
```




Veiem que en principi cada variable segueix una seqüència lògica, a part de Parch i SibSp que tenen acumulada la majoria a 0, que vol dir que no viatjaven amb família, però té sentit.

Valors atípics:

```
summary(total)
```

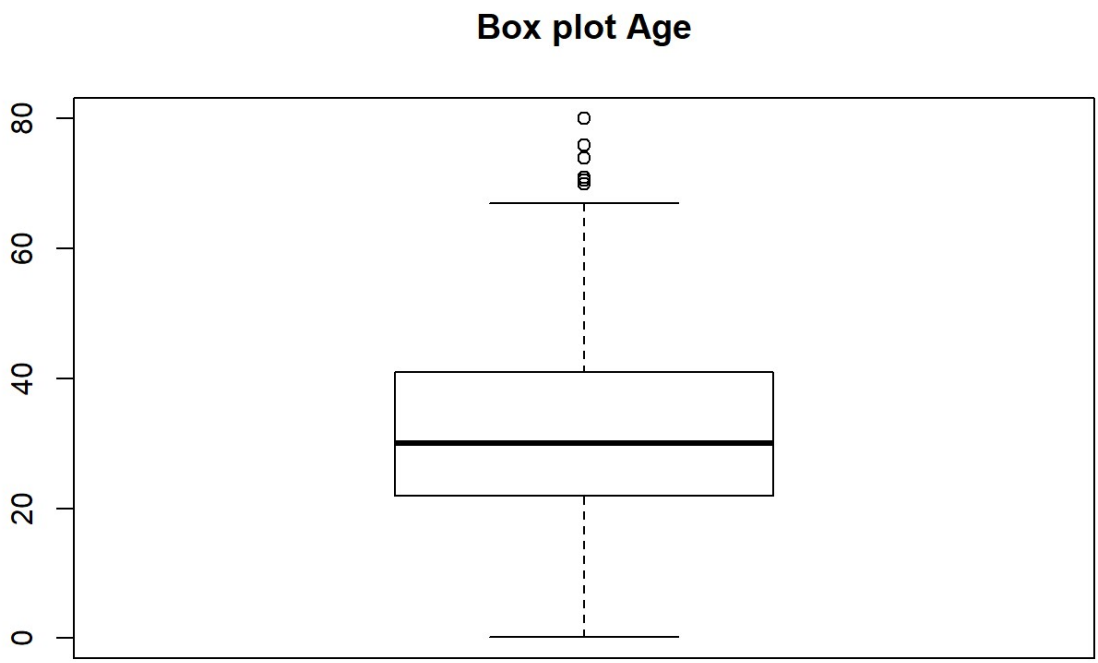
```
##   PassengerId   Survived   Pclass      Sex           Age
##  Min.   :   1   0   :549   1:323   female:466   Min.   : 0.17
##  1st Qu.: 328   1   :342   2:277   male  :843   1st Qu.:22.00
##  Median : 655   NA's:418   3:709                Median :30.00
##  Mean   : 655                                   Mean   :31.75
##  3rd Qu.: 982                                   3rd Qu.:41.00
##  Max.   :1309                                   Max.   :80.00
##
##      SibSp            Parch            Fare         Embarked
##  Min.   :0.0000   Min.   :0.000   Min.   :  0.000   C:270
##  1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:  7.896   Q:123
##  Median :0.0000   Median :0.000   Median : 14.454   S:916
##  Mean   :0.4989   Mean   :0.385   Mean   : 33.280
##  3rd Qu.:1.0000   3rd Qu.:0.000   3rd Qu.: 31.275
##  Max.   :8.0000   Max.   :9.000   Max.   :512.329
##
##     Coberta
##  NA     :1014
##  C      :  94
##  B      :  65
##  D      :  46
##  E      :  41
##  A      :  22
##  (Other):  27
```

Sabem que Survived només conté valors 0 i 1, Plass conté 1,2 i 3, Sex conté masculí i femení, Embarked conté C, S o Q.

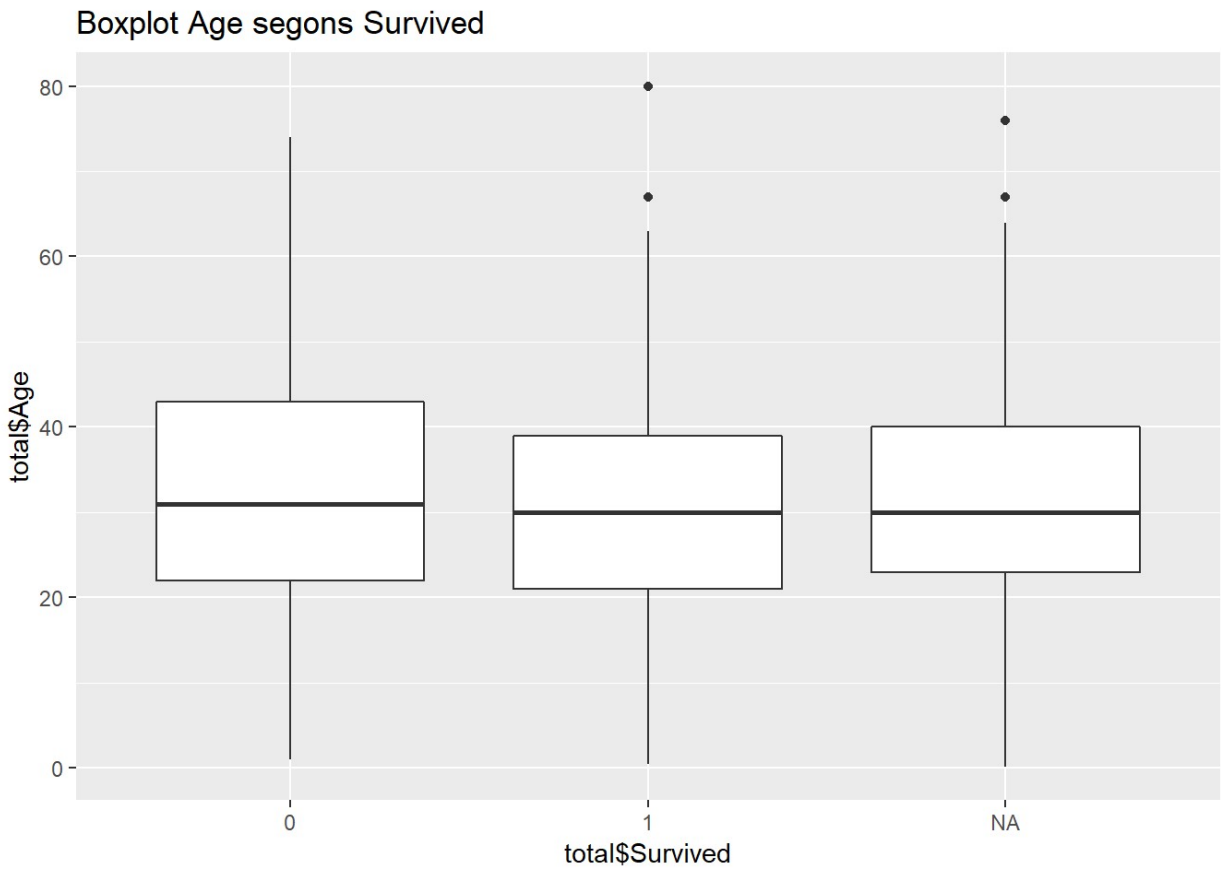
```
#Valors atípics
boxplot.stats(total$Age)
```

```
## $stats
## [1]  0.17 22.00 30.00 41.00 67.00
##
## $n
## [1] 1309
##
## $conf
## [1] 29.17026 30.82974
##
## $out
## [1] 71.0 70.5 71.0 80.0 70.0 70.0 74.0 70.0 76.0
```

```
#Boxplot
boxplot(total$Age, main="Box plot Age")
```




```
ggplot(data=total, aes(total$Survived, total$Age)) + geom_boxplot()+ ggtitle('Boxplo
t Age segons Survived')
```



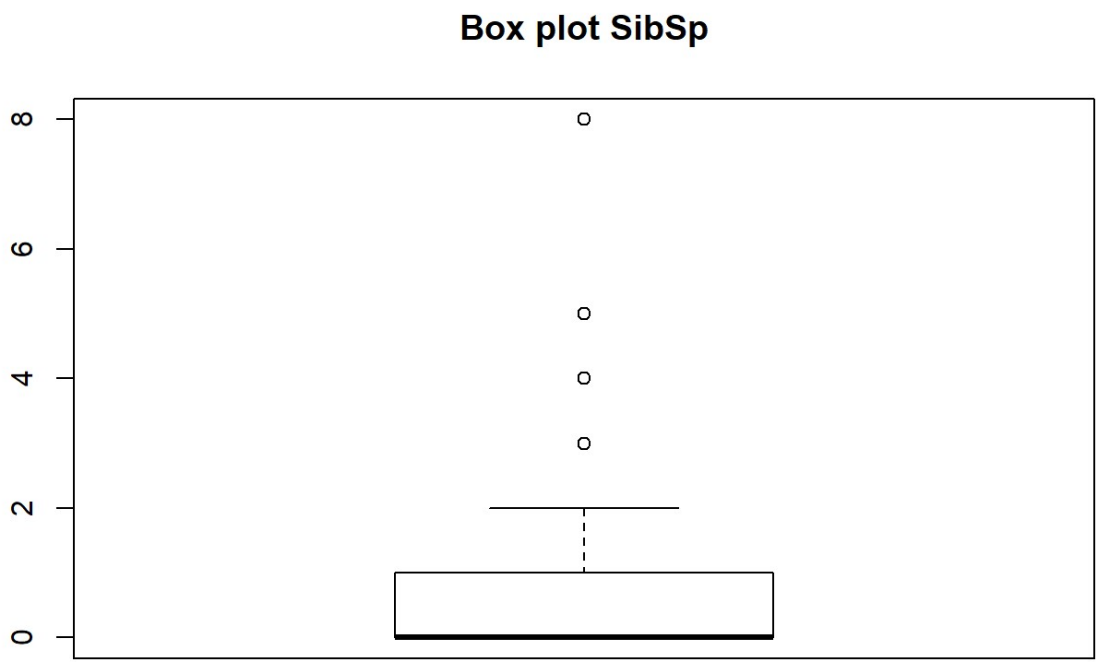
Veiem que són atípics els valors per sota 3 anys i els de sobre de 54, tot i que no ho tractem ja que creiem que són coses de les dades i no perquè siguin mostres especials (extremes).

Quan mirem Age segons Survived, veiem que només hi ha outliers a 1 i NA. No els eliminem ja que a la hora de predir, si al test tenim gent d'aquesta edat ( almenys al gràfic veiem que n'hi ha de semblant), llavors no prediriem bé el resultat.

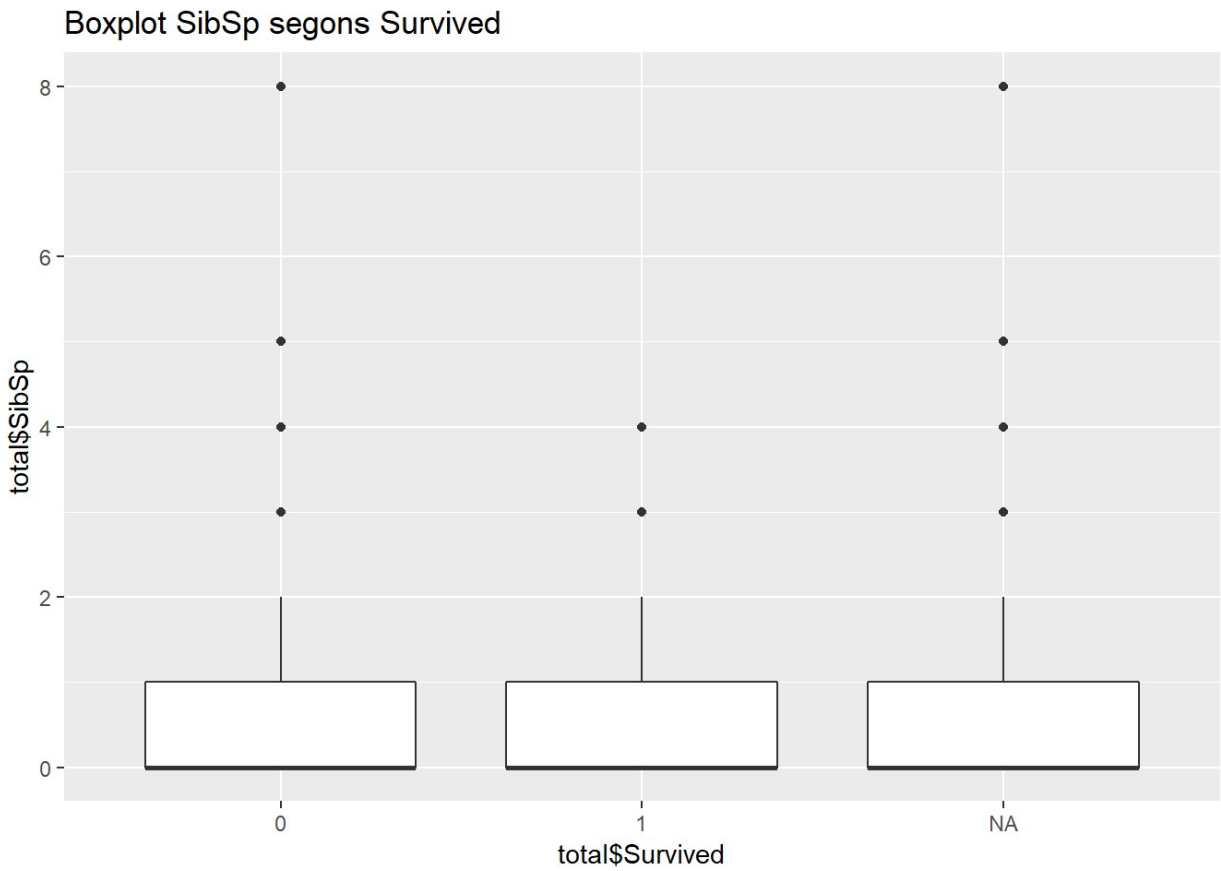
```
#Valors atípics
boxplot.stats(total$SibSp)
```

```
## $stats
## [1] 0 0 0 1 2
##
## $n
## [1] 1309
##
## $conf
## [1] -0.04367041  0.04367041
##
## $out
##  [1] 3 4 3 3 4 5 3 4 5 3 3 4 8 4 4 3 8 4 8 3 4 4 4 4 8 3 3 5 3 5 3 4 4 3 3
## [36] 5 4 3 4 8 4 3 4 8 4 8 3 4 5 3 4 8 4 8 4 3 3
```

```
#Boxplot
boxplot(total$SibSp, main="Box plot SibSp")
```



```
ggplot(data=total, aes(total$Survived, total$SibSp)) + geom_boxplot()+ ggtitle('Boxp
lot SibSp segons Survived')
```



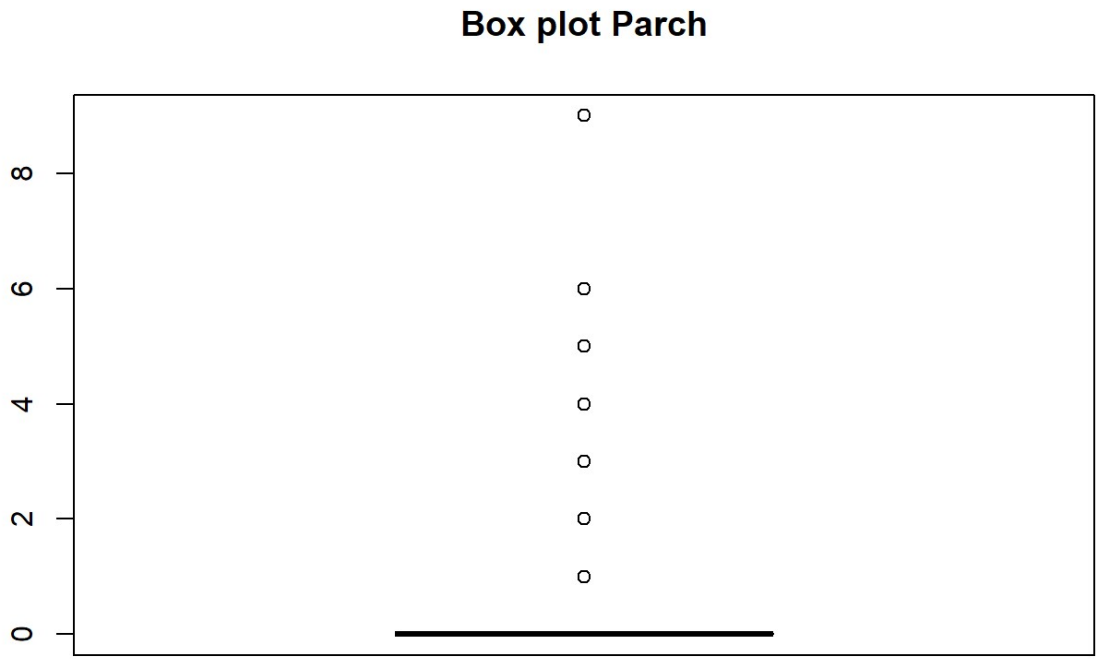

Veiem que els valors per sobre de 2 serien atípics.

Si dividim el boxplot per survived, veiem que els NA(valors de test) contenen valors ouliers que també estan a train , de manera que no eliminem els outliers ja que ens seran utils per a la predicció.

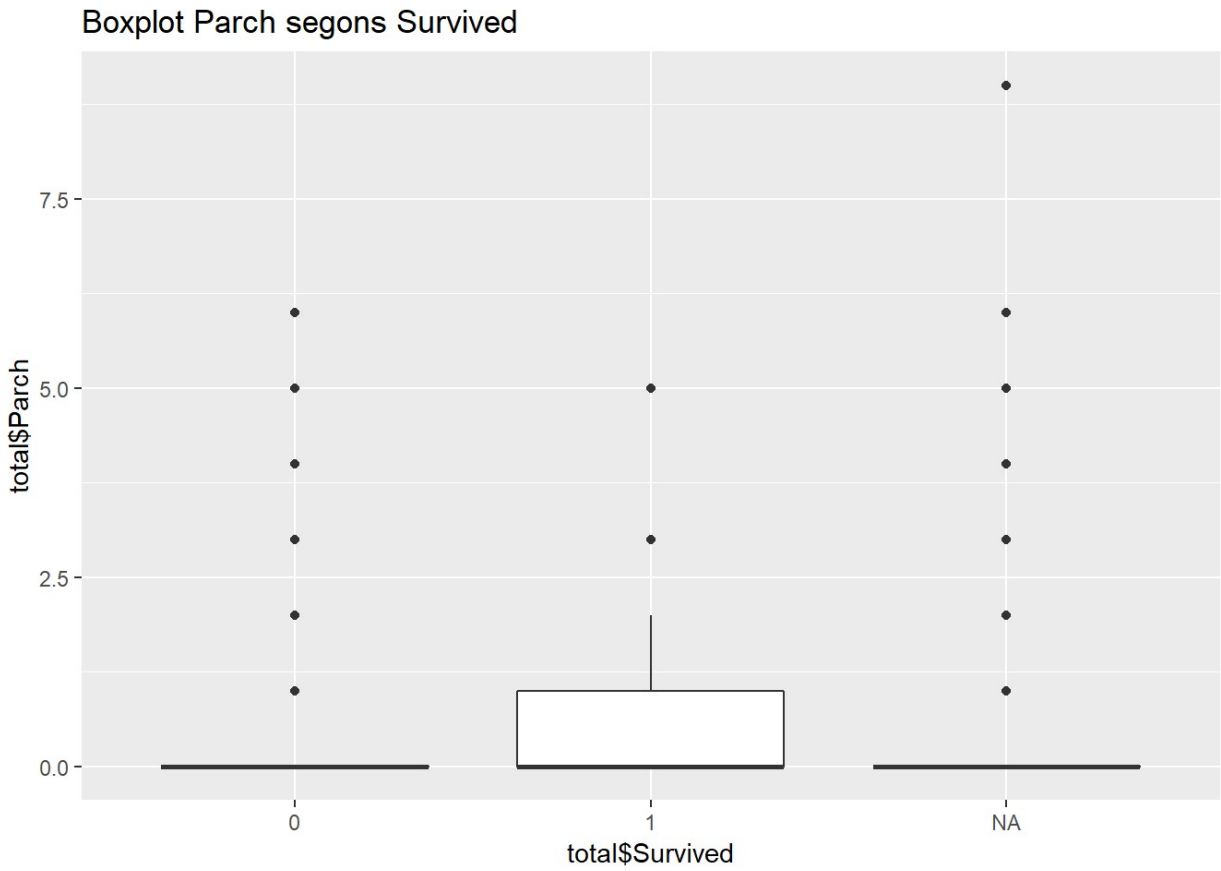
```
#Valors atípics
boxplot.stats(total$Parch)
```

```
## $stats
## [1] 0 0 0 0 0
##
## $n
## [1] 1309
##
## $conf
## [1] 0 0
##
## $out
##   [1] 1 2 1 5 1 1 5 2 2 1 1 2 2 2 1 2 2 2 3 2 2 1 1 1 1 2 1 1 2 2 1 2 2 2 1
##  [36] 2 1 1 2 1 4 1 1 1 1 2 2 1 2 1 1 1 2 1 1 2 2 2 1 1 2 2 1 2 1 1 1 1 1 1
##  [71] 1 2 1 2 2 1 1 2 1 1 2 1 1 1 1 2 1 1 1 4 1 1 2 2 2 2 2 1 1 1 2 2 1 1 2
## [106] 2 3 4 1 2 1 1 2 1 2 1 2 1 1 2 2 1 1 1 1 2 2 2 2 2 2 1 1 2 1 4 1 1 2 1
## [141] 2 1 1 2 5 2 1 1 1 2 1 5 2 1 1 1 2 1 6 1 2 1 2 1 1 1 1 1 1 1 3 2 1 1 1
## [176] 1 2 1 2 3 1 2 1 2 2 1 1 2 1 2 1 2 1 1 1 2 1 1 2 1 2 1 1 1 1 3 2 1 1 1
## [211] 1 5 2 1 1 1 1 3 1 2 2 1 2 1 2 1 2 4 1 1 2 1 1 1 4 6 2 3 1 1 2 2 2 1 1
## [246] 2 5 2 3 2 1 1 1 2 1 2 2 2 1 2 1 1 2 1 2 1 2 1 2 2 1 1 1 1 1 2 1 1 2 1
## [281] 1 1 2 1 2 9 1 1 1 2 2 2 1 9 1 1 2 2 1 1 2 1 1 1 1 1 1
```

```
#Boxplot
boxplot(total$Parch, main="Box plot Parch")
```




```
ggplot(data=total, aes(total$Survived, total$Parch)) + geom_boxplot()+ ggtitle('Boxp
lot Parch segons Survived')
```



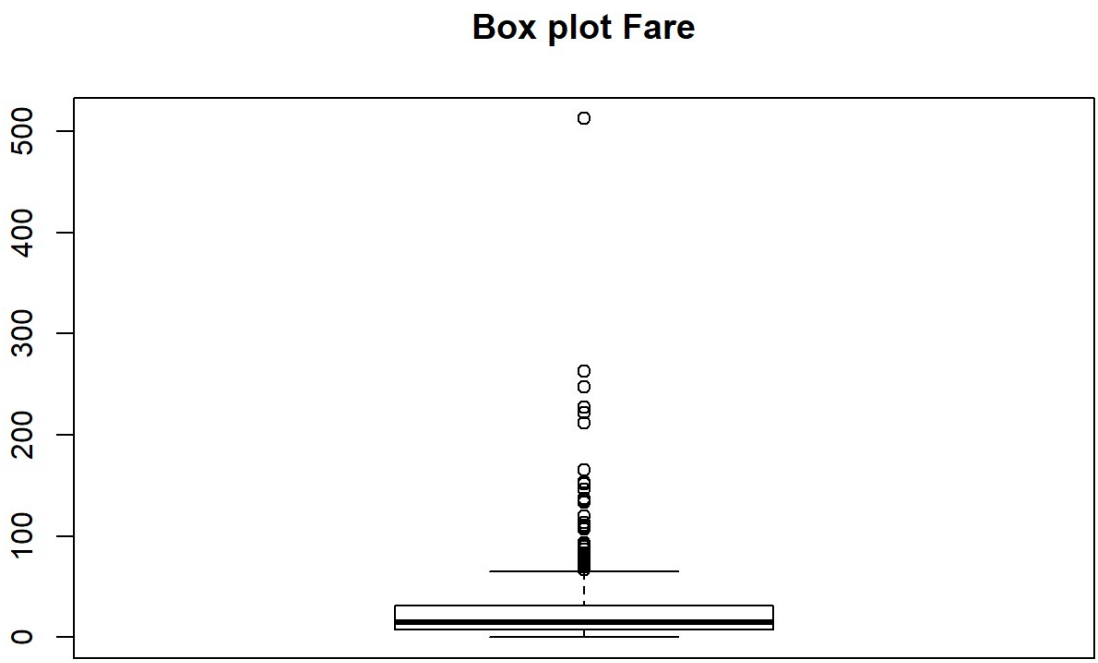

Veiem que el valor majoritari de parch és 0.

Quan dividim els valors segons survived, veiem que els NA també contenen outliers i que la distribució dels NA és molt semblant a la dels 0. No elminimen els outliers pel mateix motiu que en els anteriors: a la hora de predir test ens aniran bé.

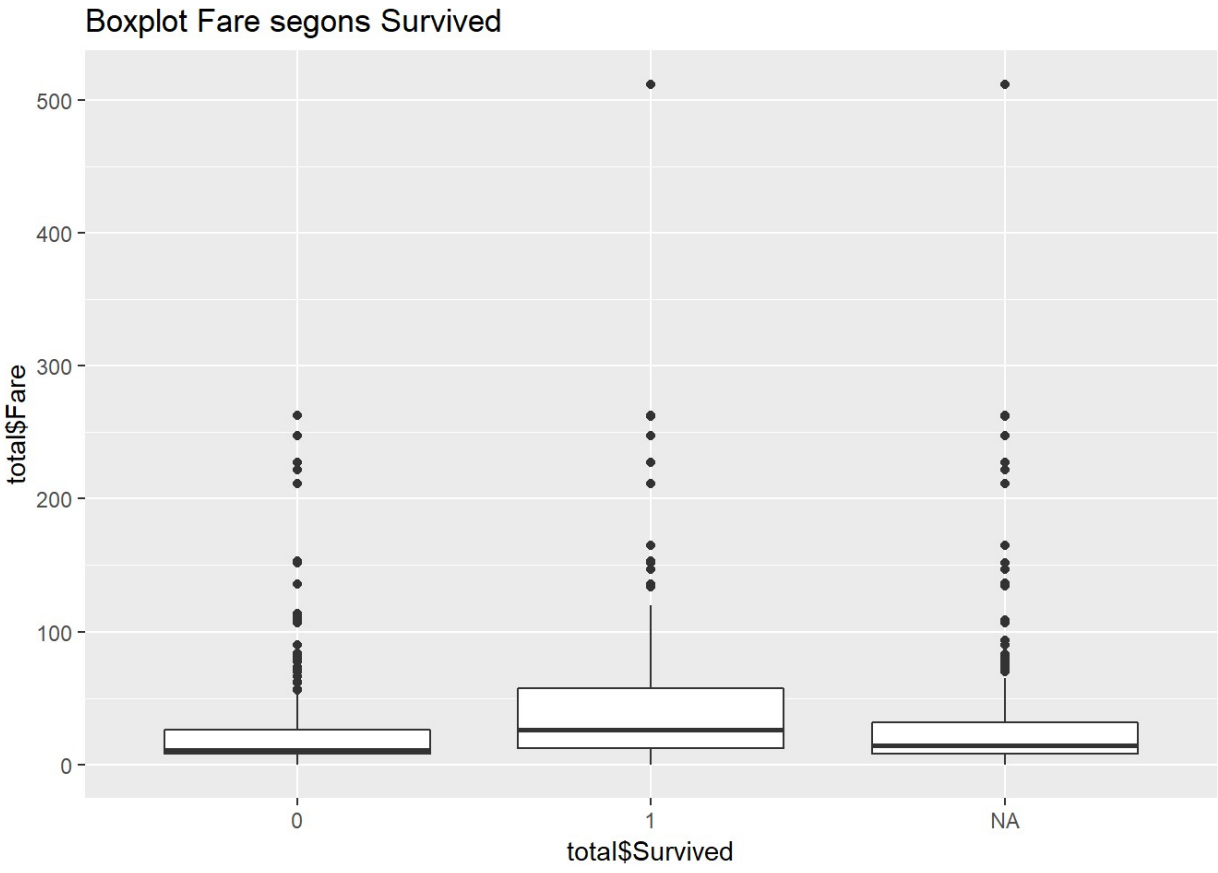
```
#Valors atípics
boxplot.stats(total$Fare)
```

```
## $stats
## [1]  0.0000  7.8958 14.4542 31.2750 65.0000
##
## $n
## [1] 1309
##
## $conf
## [1] 13.43322 15.47518
##
## $out
##   [1]  71.2833 263.0000 146.5208  82.1708  76.7292  80.0000  83.4750
##   [8]  73.5000 263.0000  77.2875 247.5208  73.5000  77.2875  79.2000
##  [15]  66.6000  69.5500  69.5500 146.5208  69.5500 113.2750  76.2917
##  [22]  90.0000  83.4750  90.0000  79.2000  86.5000 512.3292  79.6500
##  [29] 153.4625 135.6333  77.9583  78.8500  91.0792 151.5500 247.5208
##  [36] 151.5500 110.8833 108.9000  83.1583 262.3750 164.8667 134.5000
##  [43]  69.5500 135.6333 153.4625 133.6500  66.6000 134.5000 263.0000
##  [50]  75.2500  69.3000 135.6333  82.1708 211.5000 227.5250  73.5000
##  [57] 120.0000 113.2750  90.0000 120.0000 263.0000  81.8583  89.1042
##  [64]  91.0792  90.0000  78.2667 151.5500  86.5000 108.9000  93.5000
##  [71] 221.7792 106.4250  71.0000 106.4250 110.8833 227.5250  79.6500
##  [78] 110.8833  79.6500  79.2000  78.2667 153.4625  77.9583  69.3000
##  [85]  76.7292  73.5000 113.2750 133.6500  73.5000 512.3292  76.7292
##  [92] 211.3375 110.8833 227.5250 151.5500 227.5250 211.3375 512.3292
##  [99]  78.8500 262.3750  71.0000  86.5000 120.0000  77.9583 211.3375
## [106]  79.2000  69.5500 120.0000  93.5000  80.0000  83.1583  69.5500
## [113]  89.1042 164.8667  69.5500  83.1583  82.2667 262.3750  76.2917
## [120] 263.0000 262.3750 262.3750 263.0000 211.5000 211.5000 221.7792
## [127]  78.8500 221.7792  75.2417 151.5500 262.3750  83.1583 221.7792
## [134]  83.1583  83.1583 247.5208  69.5500 134.5000 227.5250  73.5000
## [141] 164.8667 211.5000  71.2833  75.2500 106.4250 134.5000 136.7792
## [148]  75.2417 136.7792  82.2667  81.8583 151.5500  93.5000 135.6333
## [155] 146.5208 211.3375  79.2000  69.5500 512.3292  73.5000  69.5500
## [162]  69.5500 134.5000  81.8583 262.3750  93.5000  79.2000 164.8667
## [169] 211.5000  90.0000 108.9000
```

```
#Boxplot
boxplot(total$Fare, main="Box plot Fare")
```




```
ggplot(data=total, aes(total$Survived, total$Fare)) + geom_boxplot()+ ggtitle('Boxpl
ot Fare segons Survived')
```



Veiem que la tarifa, a partir de 65 ja no hi ha tanta densitat. Destacar que n'hi ha un outlier que és extrem, 512.392, però com que no serà variable de l'estudi, el deixem.

Veiem que l'outlier extrem, està tant en els sobrevivents com en el NA (test), llavors els outliers ens serviran per predir millor.

Comprovem qui té el valor de 512:

```
#Comprovem qui és el que té aquest valor:
total[total$Fare>300,]
```

```
##      PassengerId Survived Pclass    Sex Age SibSp Parch     Fare Embarked
## 259          259        1      1 female  35     0     0 512.3292        C
## 680          680        1      1   male  36     0     1 512.3292        C
## 738          738        1      1   male  35     0     0 512.3292        C
## 1235        1235     <NA>      1 female  58     0     1 512.3292        C
##      Coberta
## 259     <NA>
## 680        B
## 738        B
## 1235       B
```

Com que veiem que un d'ells forma part de test, no els podem eliminar. Tampoc els canviarem el valor ja que èr exemple en un primer moment hem pensat que el valor podria ser per exemple 51.23, però al ser de primera classe ho descartem.

# 2B. Creem variables

Creem variables dummy per sex.

Creem la variable Family, que és la suma de SibSp i Parch

Creem la variable Alone, que indica si té family o no

Creem la variable Agec, que és age dividit en dos intervals: Nens i Adults

Creem la variable Agec3, que és Age dividit en 3 intervals: Nens, Adults i Avis

Creem la variable Farebin, que és Fare dividit en 6 intervals.

```
#Crear variables dummy per sex i embarked
total$SexR<-relevel(total$Sex,"female")
#total$EmbarkedR<-relevel(total$Embarked,"C")


#Creem variable family, que és el nombre de familiars
total$Family<-total$SibSp+total$Parch

total$Family<-as.numeric(total$Family)

#Creem variable Alone, que és si viatjava sol o acompanyat
total$Alone[total$Family>0]<-"1"
total$Alone[total$Family==0]<-"0"

total$Alone<- as.factor(total$Alone)

#Creem variable Agec, que és la variable Age amb dues categories segons si és nen o
adult
total$Agec[total$Age<18]<-"Nen"
total$Agec[total$Age>=18]<-"Adult"

total$Agec<- as.factor(total$Agec)

#Creem variable Agec3, amb 3 categories
total$Agec3<-"Adult"
total$Agec3[total$Age<14]<-"Nen"
total$Agec3[total$Age>=60]<-"Avi"
total$Agec3<- as.factor(total$Agec3)

library(car)
```

```
## Warning: package 'car' was built under R version 3.5.3
```

```
## Loading required package: carData
```

```
## Warning: package 'carData' was built under R version 3.5.2
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
#Mirem valors Fare
summary(total$Fare)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   7.896  14.454  33.280  31.275 512.329
```

```
#Mirem valors últim quartil Fare
summary(total$Fare[total$Fare>31.275])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   31.39   49.75   69.55   93.55  107.66  512.33
```

```
#Creem intervals a Fare
total$Farebin<-Recode(total$Fare,"0.0:7.896='Fare5';7.8961:14.454='Fare4';14.4541:3
1.275='Fare3';31.275: 49.75='Fare2'; 49.751:93.55='Fare1';93.551:512.34='Fare0'", le
vels=c('Fare0','Fare1','Fare2','Fare3','Fare4','Fare5' ))
total$Farebin<-as.factor(total$Farebin)
```

I comprovem les dades que hem creat:

```
summary(total)
```

```
##   PassengerId   Survived   Pclass      Sex           Age
##  Min.   :   1   0   :549   1:323   female:466   Min.   : 0.17
##  1st Qu.: 328   1   :342   2:277   male  :843   1st Qu.:22.00
##  Median : 655   NA's:418   3:709                Median :30.00
##  Mean   : 655                                   Mean   :31.75
##  3rd Qu.: 982                                   3rd Qu.:41.00
##  Max.   :1309                                   Max.   :80.00
##
##      SibSp            Parch            Fare         Embarked
##  Min.   :0.0000   Min.   :0.000   Min.   :  0.000   C:270
##  1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:  7.896   Q:123
##  Median :0.0000   Median :0.000   Median : 14.454   S:916
##  Mean   :0.4989   Mean   :0.385   Mean   : 33.280
##  3rd Qu.:1.0000   3rd Qu.:0.000   3rd Qu.: 31.275
##  Max.   :8.0000   Max.   :9.000   Max.   :512.329
##
##     Coberta         SexR         Family        Alone      Agec
##  NA     :1014   female:466   Min.   : 0.0000   0:790   Adult:1148
##  C      :  94   male  :843   1st Qu.: 0.0000   1:519   Nen  : 161
##  B      :  65                Median : 0.0000
##  D      :  46                Mean   : 0.8839
##  E      :  41                3rd Qu.: 1.0000
##  A      :  22                Max.   :10.0000
##  (Other):  27
##    Agec3       Farebin
##  Adult:1150   Fare0: 84
##  Avi  :  58   Fare1:158
##  Nen  : 101   Fare2: 81
##               Fare3:338
##               Fare4:311
##               Fare5:337
##
```

Finalment tornem a separar els dos datasets en test i train

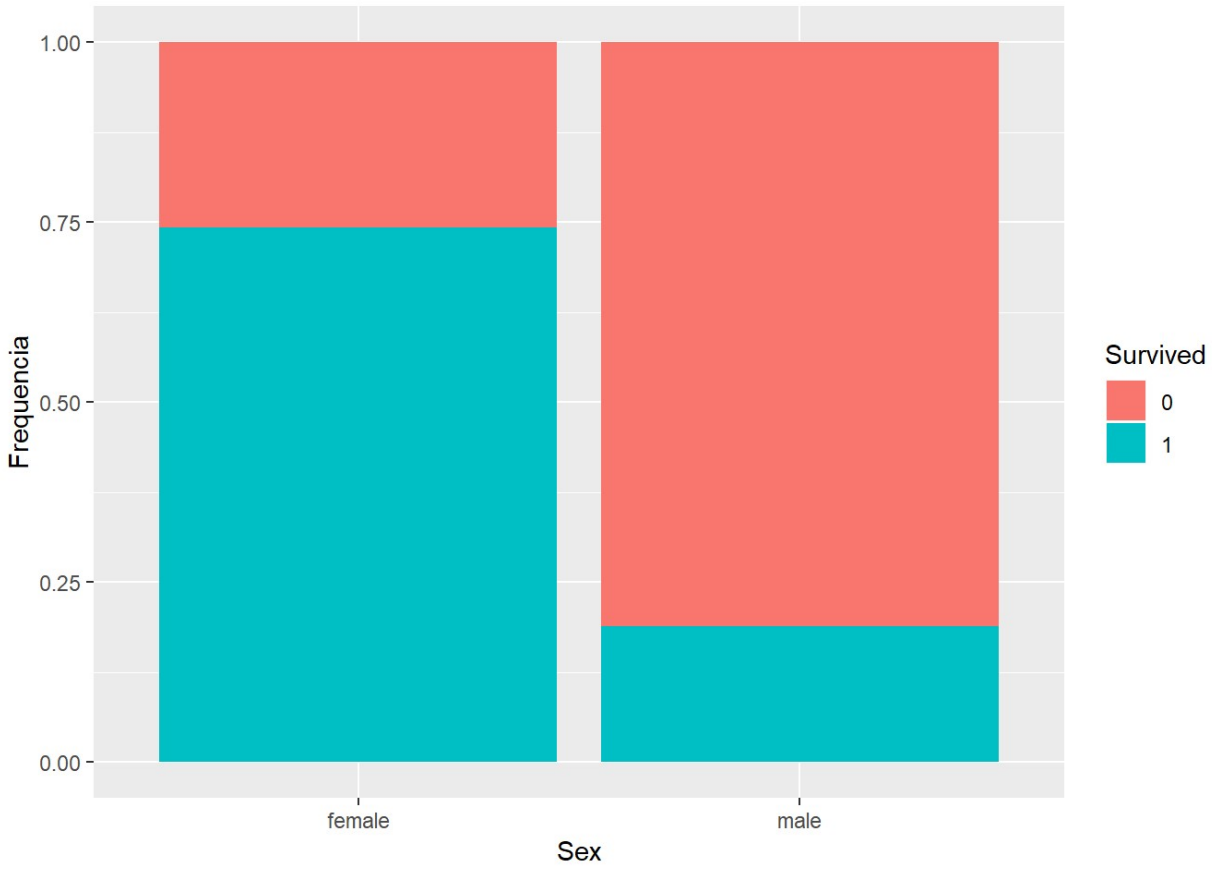
```
train<-total[!is.na(total$Survived),]
test<-total[is.na(total$Survived),]
```

# 4. Anàlisi de les dades.

## 4.0 Anàlisi descriptiu de les dades

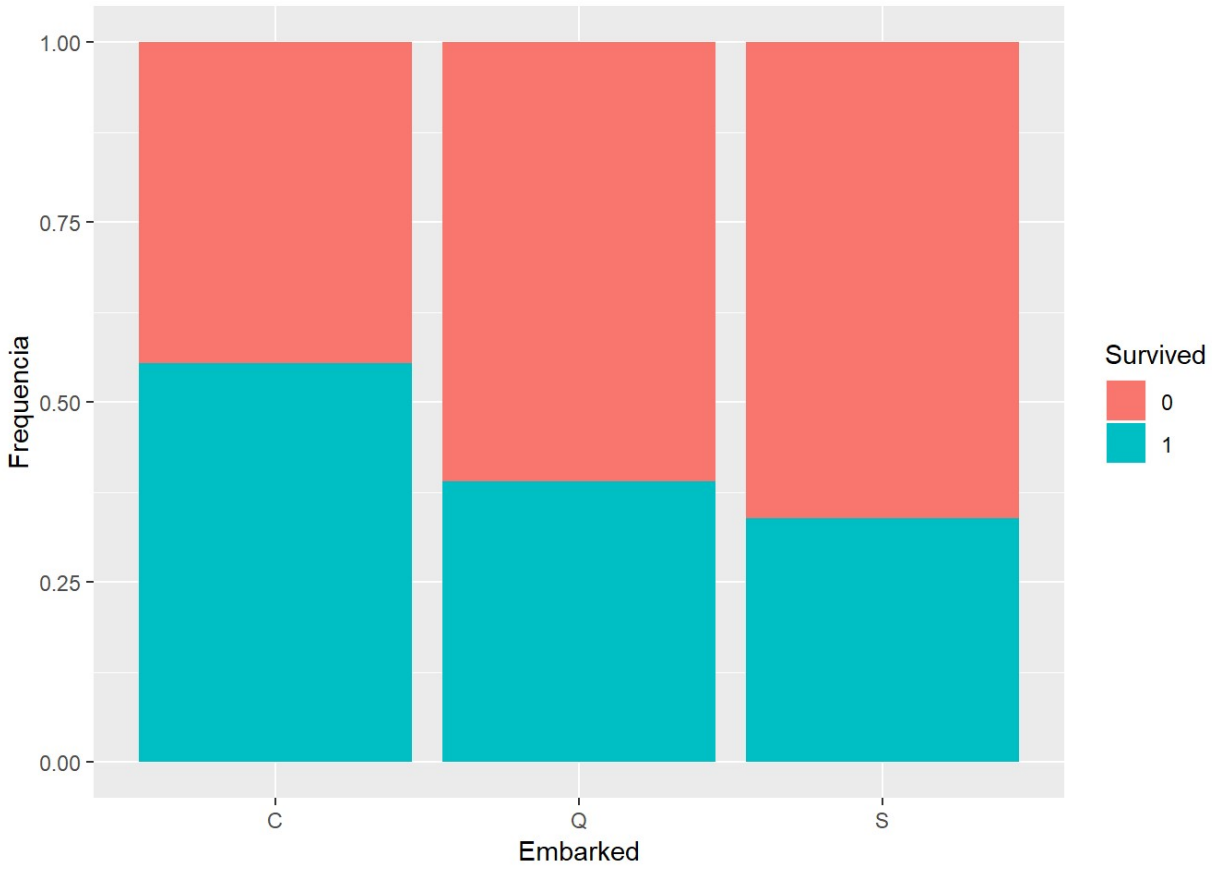
Analitzem relacions entre variables

```
library(ggplot2)

# Visualitzem la relació entre "sex" y "survival" (freq.):
ggplot(data = train,aes(x=Sex,fill=Survived))+geom_bar(position="fill")+ylab("Freque
ncia")
```



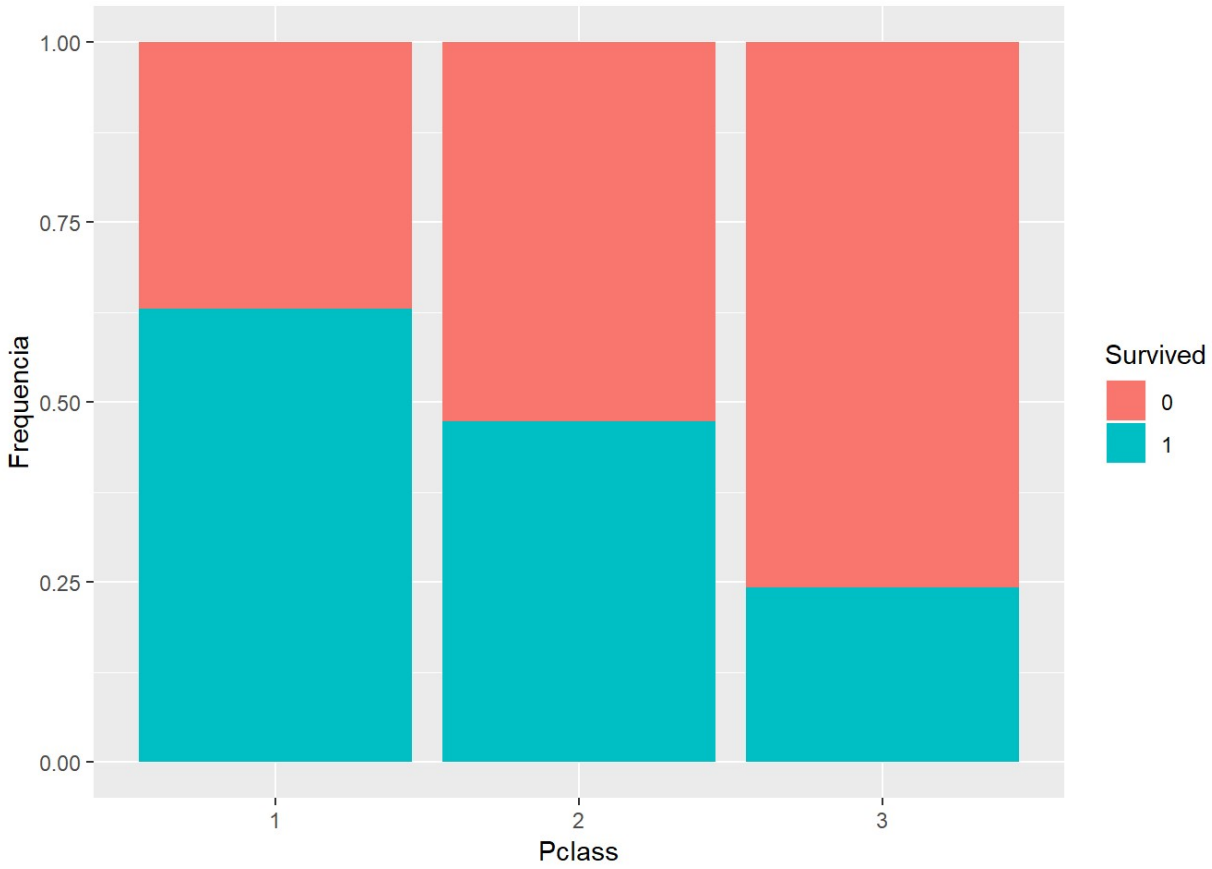

Veiem que les dones tenen més probabilitat de sobreviure que els homes.

```
# Visualitzem Survival en funció de Embarked (Freqüència)
ggplot(data = train,aes(x=Embarked,fill=Survived))+geom_bar(position="fill")+ylab("F
requencia")
```



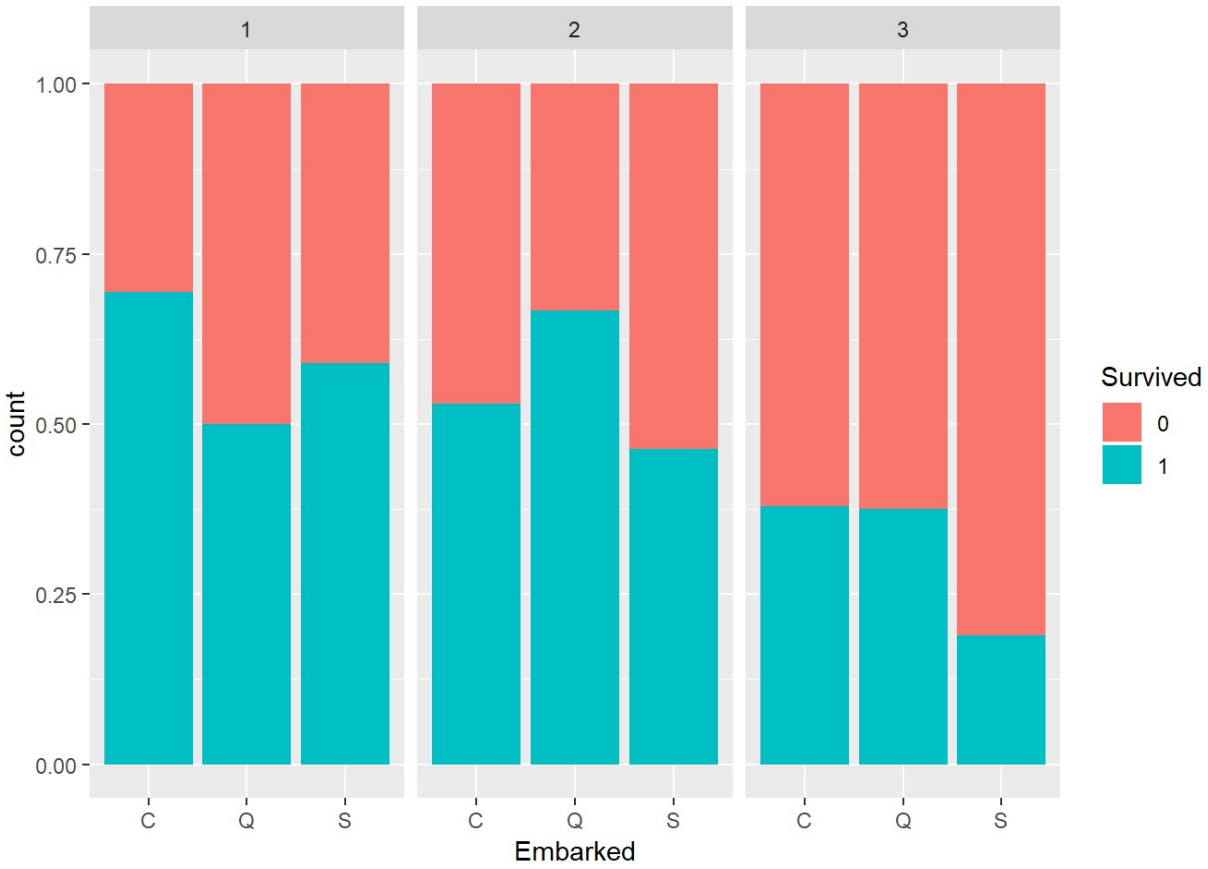

Veiem que els que han embarcat a C han sobreviscut més. Els de la categoria sense nom són els que estan en blanc.

```
# Visualitzem Survival en funció de Pclass (Freqüència)
ggplot(data = train,aes(x=Pclass,fill=Survived))+geom_bar(position="fill")+ylab("Fre
quencia")
```

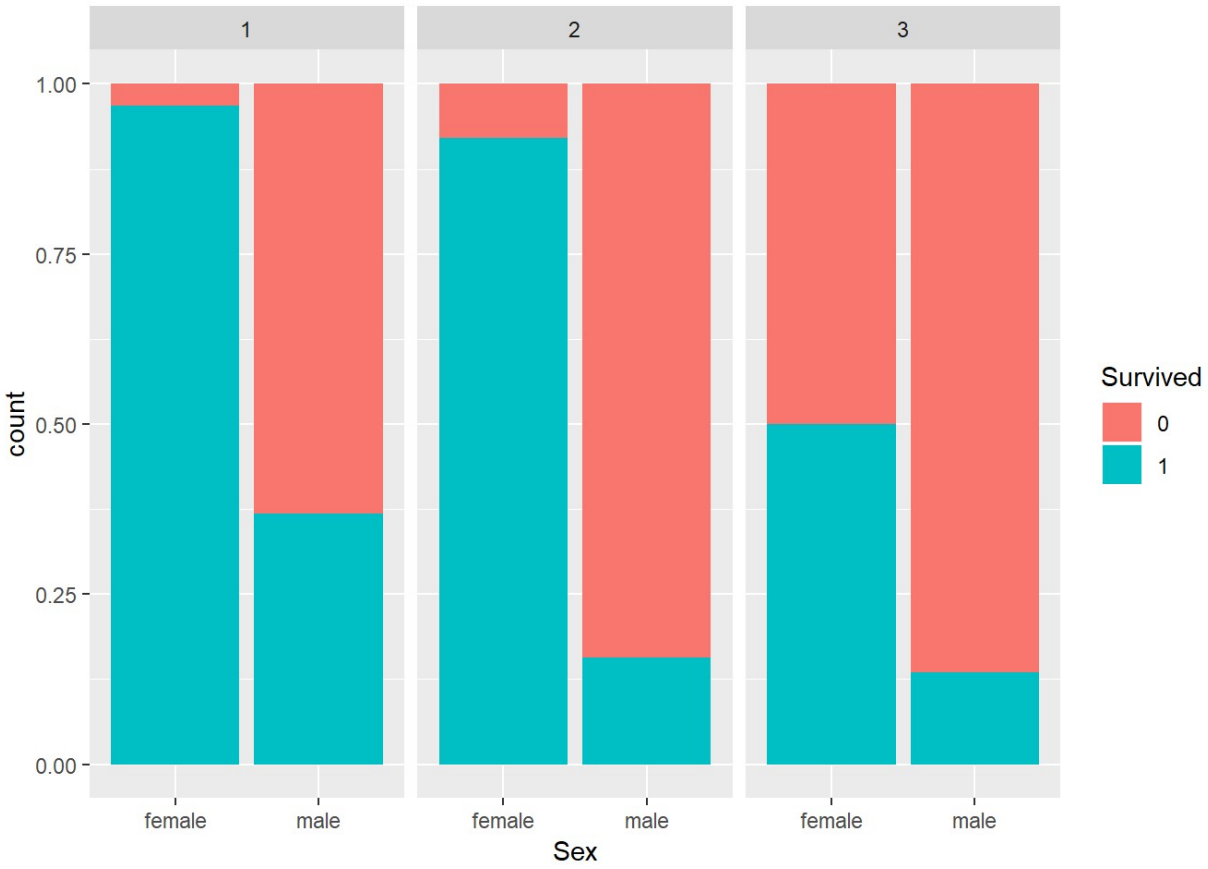



Veiem que els de primera classe són els que més van sobreviure

```
#  Supervivents segons Embarked i Pclass:
ggplot(data = train,aes(x=Embarked,fill=Survived))+geom_bar(position="fill")+facet_w
rap(~Pclass)
```



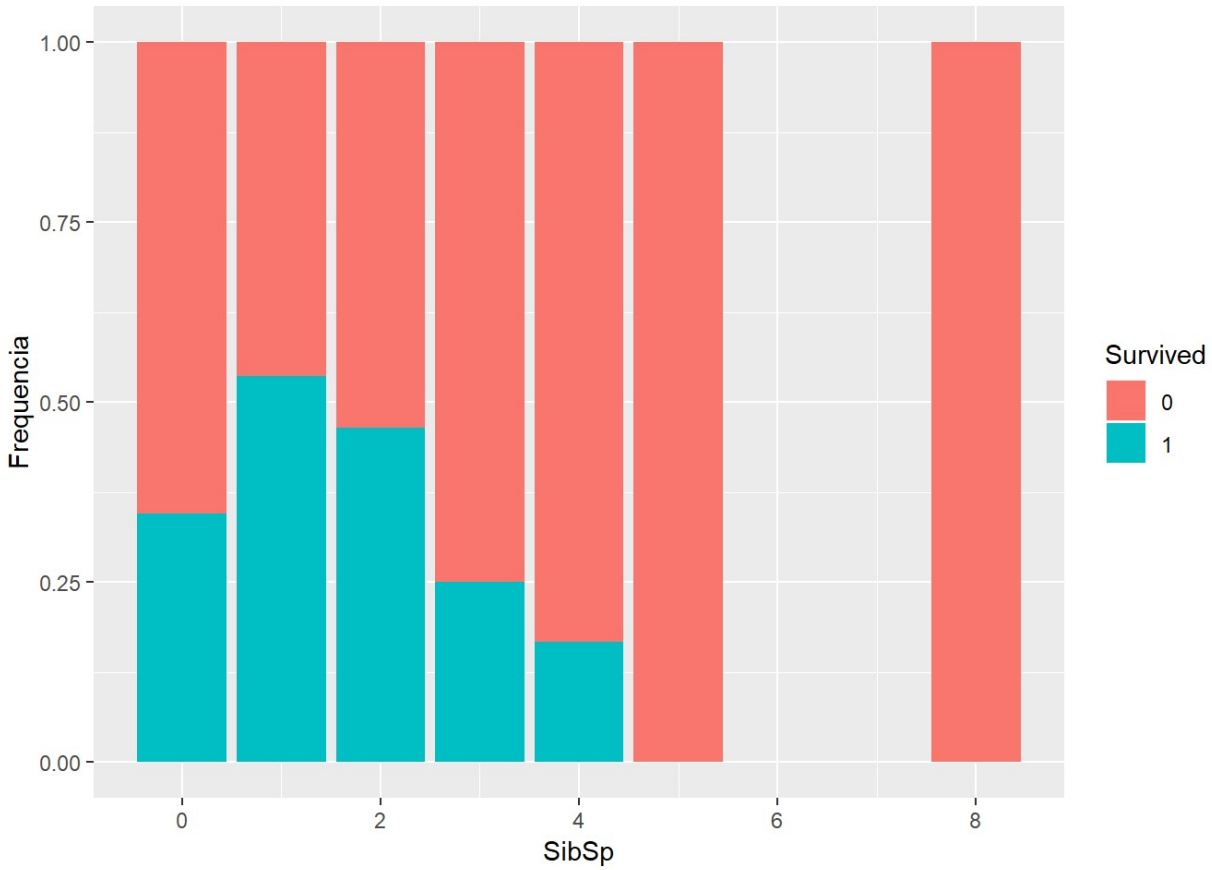

```
#  Supervivents segons Sex i Pclass:
ggplot(data = train,aes(x=Sex,fill=Survived))+geom_bar(position="fill")+facet_wrap(~
Pclass)
```




Veiem que les dones de primera i segona classe tenen una supervivència de més del 85%, en canvi les de tercera classe tenen un índex de supervivència del 50%.

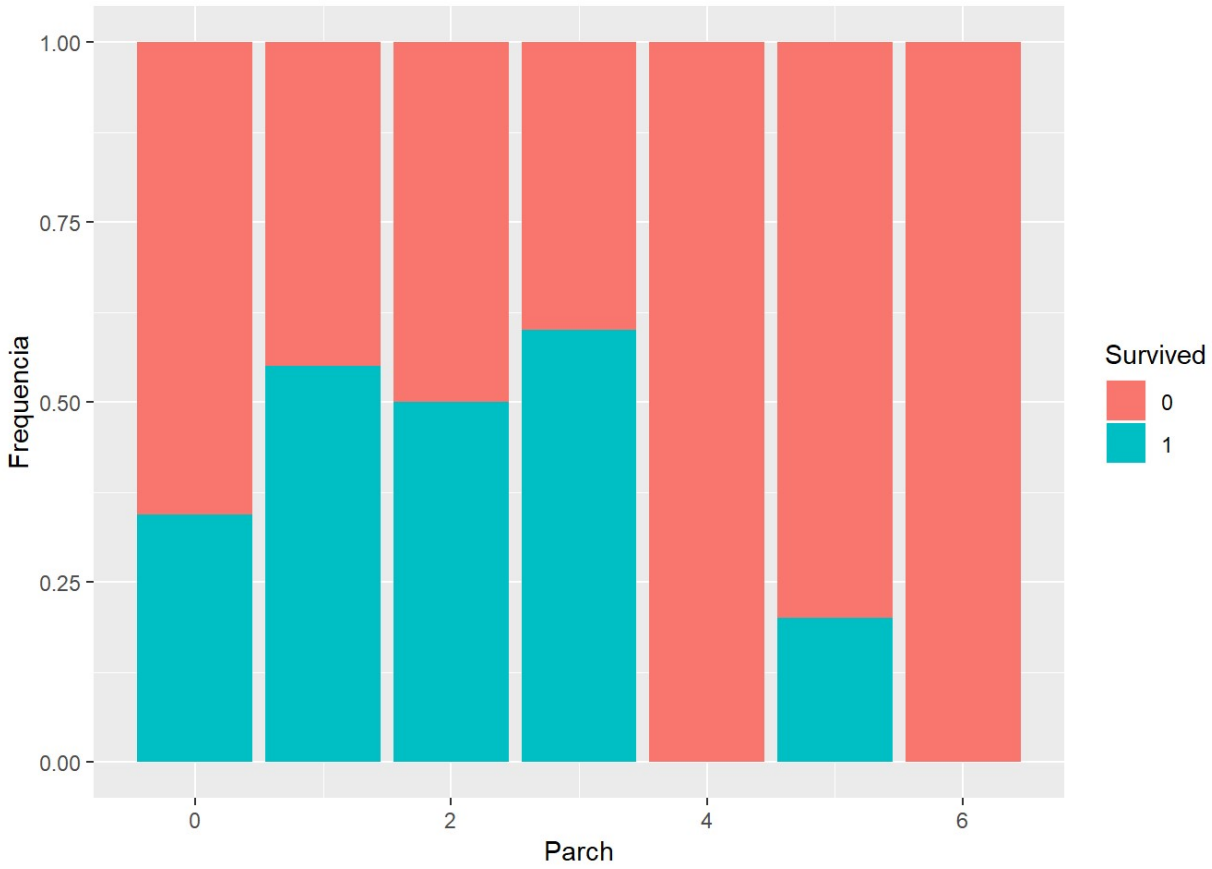
Veiem que els homes de primera classe tenen una supervivència d'un 37% i que els homes de segona i tercera classe tenen menys supervivència, d'un 15% aproximadament.

```
# Visualitzem Survival en funció de sibsp (Freqüència)
ggplot(data = train,aes(x=SibSp,fill=Survived))+geom_bar(position="fill")+ylab("Freq
uencia")
```




Veiem que a partir de SibSp>4 l'íned de supervivència és 0%. Els que tenen més supervivència són els que tenen SibSp 1 o 2.

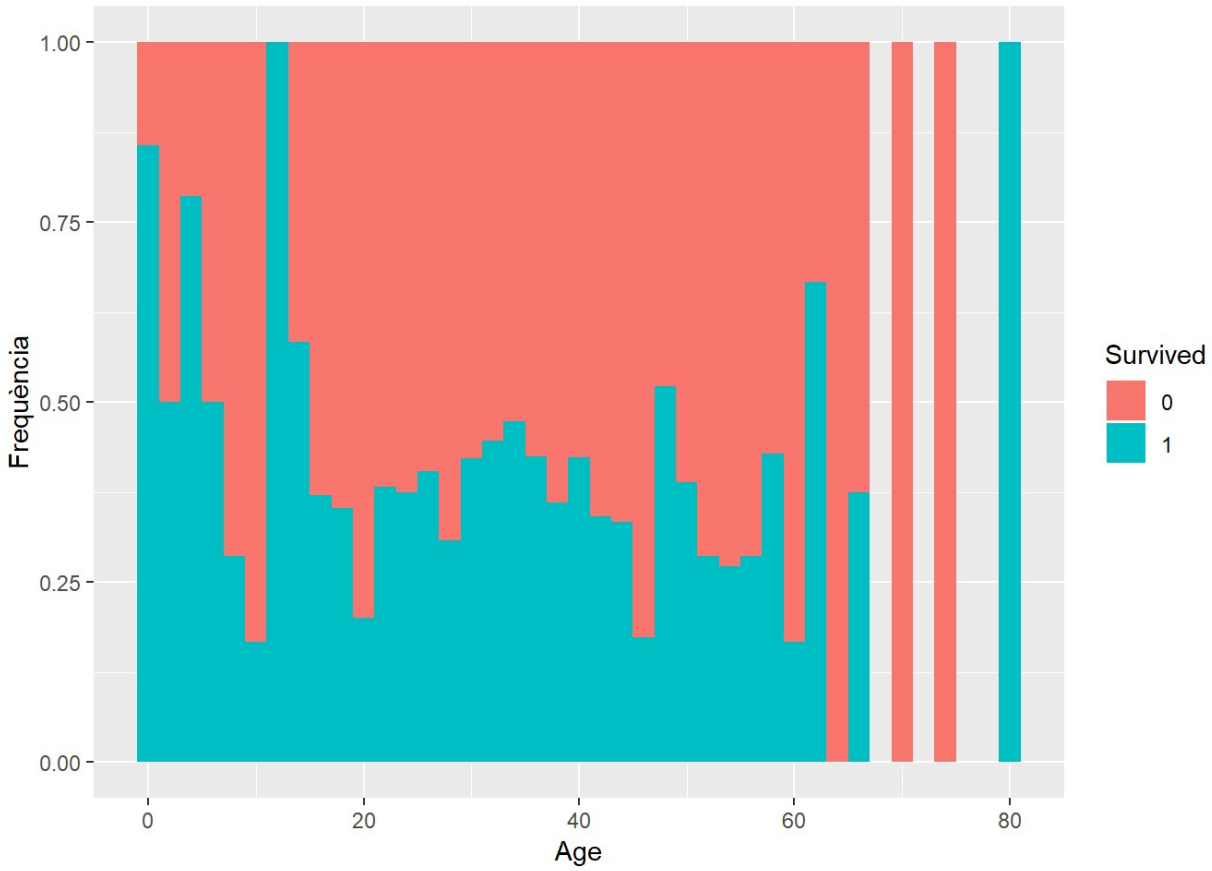
```
# Visualitzem Survival en funció de Parch (Freqüència)
ggplot(data = train,aes(x=Parch,fill=Survived))+geom_bar(position="fill")+ylab("Freq
uencia")
```




Veiem que els que tenen més supervivència són els que tenen parch 3 i 1.

```
#Survival respecte de age (freq):
ggplot(data = train[!(is.na(train$Age)),],aes(x=Age,fill=Survived))+geom_histogram(b
inwidth =2, position="fill")+ylab("Frequència")
```

```
## Warning: Removed 8 rows containing missing values (geom_bar).
```




Veiem que per sota de 16 anys l'índex de supervivència és alt: de 0 a 8 anys és 50% o més i de 13 a 16 anys també.

També és força alt, però menys del 50% pels voltants dels 21-44 anys. A partir dels 46 anys fins als 62 també hi ha algunts punts alts de supervivència.

# 4.1. Selecció dels grups de dades que es volen analitzar/comparar (planificació dels anàlisis a aplicar).

Com que el que volem és predir la supervivència de les dades de test a partir de train, Utilitzarem principalment les dades de train, menys quan predim els valors de test, que utilitzarem test. A train hi tenim les següents variables:

```
summary(train)
```

```
##   PassengerId    Survived Pclass      Sex           Age
##  Min.   :  1.0   0:549    1:216   female:314   Min.   : 0.42
##  1st Qu.:223.5   1:342    2:184   male  :577   1st Qu.:22.00
##  Median :446.0            3:491                Median :30.50
##  Mean   :446.0                                 Mean   :31.76
##  3rd Qu.:668.5                                 3rd Qu.:42.00
##  Max.   :891.0                                 Max.   :80.00
##
##      SibSp           Parch             Fare        Embarked    Coberta
##  Min.   :0.000   Min.   :0.0000   Min.   :  0.00   C:168    NA     :687
##  1st Qu.:0.000   1st Qu.:0.0000   1st Qu.:  7.91   Q: 77    C      : 59
##  Median :0.000   Median :0.0000   Median : 14.45   S:646    B      : 47
##  Mean   :0.523   Mean   :0.3816   Mean   : 32.20            D      : 33
##  3rd Qu.:1.000   3rd Qu.:0.0000   3rd Qu.: 31.00            E      : 32
##  Max.   :8.000   Max.   :6.0000   Max.   :512.33            A      : 15
##                                                             (Other): 18
##      SexR         Family        Alone      Agec       Agec3
##  female:314   Min.   : 0.0000   0:537   Adult:771   Adult:779
##  male  :577   1st Qu.: 0.0000   1:354   Nen  :120   Avi  : 39
##               Median : 0.0000                       Nen  : 73
##               Mean   : 0.9046
##               3rd Qu.: 1.0000
##               Max.   :10.0000
##
##   Farebin
##  Fare0: 53
##  Fare1:108
##  Fare2: 54
##  Fare3:236
##  Fare4:217
##  Fare5:223
##
```

A train podem eliminar PassengerId ( a test no perquè ho necessitem per penjar-ho a kaggle).

```
train<- select(train, -PassengerId)
```

Les altres variables (Pclass, Sex, Age, Sibsp, parch, fare, embarked, sexr,family, alone, agec, agec3 i farebin) les utilitzarem per coneixr més la mostra en relació a survived fent contrasts d'hipòtesis i llavors utilitzarem els atributs per crear alguns models predictius com un model de regressió i un arbre per predir survived.

Quan utilitzem Pclass, intentarem no utilitzar fare ja que estan relacionats directament, ja que els de classe 1 tindran un fare més elevat que els de classe 3.

Quan utilitzem SibSp i Parch, no utilitzarem ni Family ni Alone ni Farebin.

Quan utilitzem Age, no utilitzarem ni Agec ni Agec3 i quan utilitzem Sex no utilitzarem SexR.

# 4.2. Comprovació de la normalitat i homogeneïtat de la variància.

Al tenir una mostra de més de 30, considerem el Teorema del Limit Central i per tant, considerem que la mostra es distribueix normalment.

```
#Comprovem normalitat de cada variable
library(nortest)
```

```
## Warning: package 'nortest' was built under R version 3.5.2
```

```
alpha = 0.05
col.names = colnames(train)
for (i in 1:ncol(train)) {
  if (i == 1) cat("Variables que no segueixen una distribució normal:\n")
  if (is.integer(train[,i]) | is.numeric(train[,i])) {
    p_val = ad.test(train[,i])$p.value
    if (p_val < alpha) {
      cat(col.names[i])
      # Format output
      if (i < ncol(train) - 1) cat(", ")
      if (i %% 3 == 0) cat("\n")
    }
  }
}
```

```
## Variables que no segueixen una distribució normal:
## Age, SibSp, Parch,
## Fare, Family,
```

```
#Estudiem homogeneitat variancies (vol veure es si la variància d'una variable és ho
mogènia tenint en compte els diferents grups identificats per una variable categòric
a com Survived)
fligner.test(Age ~ Survived, data = train)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  Age by Survived
## Fligner-Killeen:med chi-squared = 0.00024885, df = 1, p-value =
## 0.9874
```

```
fligner.test(Parch ~ Survived, data = train)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  Parch by Survived
## Fligner-Killeen:med chi-squared = 11.253, df = 1, p-value =
## 0.0007948
```

```
fligner.test(SibSp ~ Survived, data = train)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  SibSp by Survived
## Fligner-Killeen:med chi-squared = 1.2514, df = 1, p-value = 0.2633
```

```
fligner.test(Family ~ Survived, data = train)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  Family by Survived
## Fligner-Killeen:med chi-squared = 19.647, df = 1, p-value =
## 9.317e-06
```

Si p>0.05, acceptem Ho variances de les mostres son homogènies.

Veiem que són homogènies només en SibSp i que no són homogènies en Age, Parch i Family

# 4.3. Aplicació de proves estadístiques per comparar els grups de dades. En funció de les dades i de l'objectiu de l'estudi, aplicar proves de contrast d'hipòtesis,correlacions, regressions, etc. Aplicar almenys tres mètodes d'anàlisi diferents.

## 4.3.1Contrast d'hipòtesis

### 4.3.1.1- Hi ha diferència entre la supervivència dels homes i de les dones?

Ho: mitjana home = mitjana dona

H1(0):mitjana home <> mitjana dona

H1(1): mitjana home > mitjana dona

H1(2): mitjana dona > mitjana home

```r
#Creem dues taules, una amb les dades de survived d'homes i l'altre de dones
train_dona <- as.numeric(as.character(train$Survived[train$Sex=="female"]))
train_home <- as.numeric(as.character(train$Survived[train$Sex=="male"]))

#Primer mirem si variancies son iguals:
var.test(train_home, train_dona, conf.level=.95)
```

```
##
##  F test to compare two variances
##
## data:  train_home and train_dona
## F = 0.7993, num df = 576, denom df = 313, p-value = 0.02218
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.6558632 0.9685628
## sample estimates:
## ratio of variances
##           0.799295
```

```r
#Fem el test per comprovar si home=dona i H1(0)
t.test(train_home,train_dona,var.equal=FALSE)
```

```
##
##  Welch Two Sample t-test
##
## data:  train_home and train_dona
## t = -18.672, df = 584.43, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.6113121 -0.4949481
## sample estimates:
## mean of x mean of y
## 0.1889081 0.7420382
```

```r
 #Com que veiem que no són iguals, fem un test per mirar si home>dona H1(1)
t.test(train_home,train_dona,var.equal=FALSE, alternative='greater')
```

```
##
##  Welch Two Sample t-test
##
## data:  train_home and train_dona
## t = -18.672, df = 584.43, p-value = 1
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  -0.6019342        Inf
## sample estimates:
## mean of x mean of y
## 0.1889081 0.7420382
```

```
 #Com que veiem que no són iguals, fem un test per mirar si dona>home H1(2)
t.test(train_dona,train_home,var.equal=FALSE, alternative='greater')
```

```
##
##  Welch Two Sample t-test
##
## data:  train_dona and train_home
## t = 18.672, df = 584.43, p-value < 2.2e-16
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  0.5043259       Inf
## sample estimates:
## mean of x mean of y
## 0.7420382 0.1889081
```

I veiem que les variàncies són diferents, ja que p (0.02218) és més petit que 0.05, rebutjant la hipòtesis nul·la de que les variancies son iguals.

En el t-test, veiem que p =2.2e-16 < 0.05, per tant rebutjem la Hipòtesi nul·la, de manera que acceptem la Hipòtesi alternativa de que la mitjana de supervivència de l'home és dierent que la de la dona, que vol dir que en un 95% de confiança obtenim que la mitjana de la supervivència dels homes és diferent a la mitjana de supervivència de les dones.

Ara comprovem quina mitjana és més gran:

En el segon test on mirem si H1(1):mitjana home > mitjana dona, llavors veiem que p=1>0.05, per tant no podem rebutjar la hipòtesi nul·la que la mitjana de home en supervivència és igual que la de la dona, però si que rebutjem que mitjana_home>mitjana_dona

En el tercer test on mirem si H1(1):mitjana dona > mitjana home, llavors veiem que p=2.2e-16<0.05 de manera que rebutjem la hipòtesi nul·la i acceptem l'hipòtesi alternativa que la mitjana de la dona és més gran que la de la dona, que vol dir que hi ha més hi ha mes 0 ( no supervivents) i per tant, que les dones tenien més probabilitat de sobreviure.

### 4.3.1.2- Hi ha diferència entre la supervivència dels nens i dels adults?

Ho: mitjana adult = mitjana nen

H1(0):mitjana adult <> mitjana nen

H1(1): mitjana adult > mitjana nen

H1(2): mitjana nen > mitjana adult

```
#Creem dues taules, una amb les dades de survived d'adults i l'altre de nens
train_nen <- as.numeric(as.character(train$Survived[train$Agec=="Nen"]))
train_adult <- as.numeric(as.character(train$Survived[train$Agec=="Adult"]))

#Primer mirem si variancies son iguals:
var.test(train_adult, train_nen, conf.level=.95)
```

```
##
##  F test to compare two variances
##
## data:  train_adult and train_nen
## F = 0.91961, num df = 770, denom df = 119, p-value = 0.5213
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.6895079 1.1928740
## sample estimates:
## ratio of variances
##          0.9196084
```

```
#Fem el test per comprovar si nen=adult i H1(0)
t.test(train_adult,train_nen,var.equal=TRUE)
```

```
##
##  Two Sample t-test
##
## data:  train_adult and train_nen
## t = -3.2316, df = 889, p-value = 0.001276
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.24672735 -0.06027654
## sample estimates:
## mean of x mean of y
## 0.3631647 0.5166667
```

```
 #Com que veiem que no són iguals, fem un test per mirar si home>dona H1(1)
t.test(train_adult,train_nen,var.equal=TRUE, alternative='greater')
```

```
##
##  Two Sample t-test
##
## data:  train_adult and train_nen
## t = -3.2316, df = 889, p-value = 0.9994
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  -0.2317142        Inf
## sample estimates:
## mean of x mean of y
## 0.3631647 0.5166667
```

```
 #Com que veiem que no són iguals, fem un test per mirar si dona>home H1(2)
t.test(train_nen,train_adult,var.equal=TRUE, alternative='greater')
```

```
##
##  Two Sample t-test
##
## data:  train_nen and train_adult
## t = 3.2316, df = 889, p-value = 0.0006381
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  0.07528973        Inf
## sample estimates:
## mean of x mean of y
## 0.5166667 0.3631647
```

I veiem que les variàncies són iguals, ja que p (0.5234) és més gran que 0.05, acceptant la hipòtesis nul·la de que les variancies son iguals.

En el primer t-test, veiem que p =0.0004393 < 0.05, per tant rebutjem la Hipòtesi nul·la, de manera que acceptem la Hipòtesi alternativa de que la mitjana de supervivència de l'adult és dierent que la del nen, que vol dir que en un 95% de confiança obtenim que la mitjana de la supervivència dels adults és diferent a la mitjana de supervivència dels nens.

Ara comprovem quina mitjana és més gran:

En el segon test on mirem si H1(1):mitjana adult > mitjana nen, llavors veiem que p=0.9998>0.05, per tant no podem rebutjar la hipòtesi nul·la que la mitjana d'adult en supervivència és igual que la del nen, però si que rebutjem que mitjana_adult>mitjana_nen

En el tercer test on mirem si H1(1):mitjana nen > mitjana adult, llavors veiem que p=0.0002196<0.05 de manera que rebutjem la hipòtesi nul·la i acceptem l'hipòtesi alternativa que la mitjana dels nens és més gran que la dles adults, que vol dir que en adults hi ha més 0 ( no supervivents) i per tant, que els nens tenien més probabilitat de sobreviure.

### 4.3.1.3- Hi ha diferència entre la supervivència dels que anaven sols i dels que anaven acompanyats?

Ho: mitjana sol = mitjana acompanyat

H1(0):mitjana sol <> mitjana acompanyat

H1(1): mitjana sol > mitjana acompanyat

H1(2): mitjana acompanyat > mitjana sol

```
#Creem dues taules, una amb les dades de survived dels que van sols i l'altre dels q
ue van acompanyats
train_sol <- as.numeric(as.character(train$Survived[train$Alone=="0"]))
train_acompanyat <- as.numeric(as.character(train$Survived[train$Alone=="1"]))

#Primer mirem si variancies son iguals:
var.test(train_sol, train_acompanyat, conf.level=.95)
```

```
##
##  F test to compare two variances
##
## data:  train_sol and train_acompanyat
## F = 0.8449, num df = 536, denom df = 353, p-value = 0.07955
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.6972027 1.0200358
## sample estimates:
## ratio of variances
##          0.8449033
```

```
#Fem el test per comprovar si nen=adult i H1(0)
t.test(train_sol,train_acompanyat,var.equal=TRUE)
```

```
##
##  Two Sample t-test
##
## data:  train_sol and train_acompanyat
## t = -6.193, df = 889, p-value = 9.009e-10
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.2661628 -0.1380603
## sample estimates:
## mean of x mean of y
## 0.3035382 0.5056497
```

```
 #Com que veiem que no són iguals, fem un test per mirar si home>dona H1(1)
t.test(train_sol,train_acompanyat,var.equal=TRUE, alternative='greater')
```

```
##
##  Two Sample t-test
##
## data:  train_sol and train_acompanyat
## t = -6.193, df = 889, p-value = 1
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  -0.2558478        Inf
## sample estimates:
## mean of x mean of y
## 0.3035382 0.5056497
```

```
 #Com que veiem que no són iguals, fem un test per mirar si dona>home H1(2)
t.test(train_acompanyat,train_sol,var.equal=TRUE, alternative='greater')
```

```
##
##  Two Sample t-test
##
## data:  train_acompanyat and train_sol
## t = 6.193, df = 889, p-value = 4.505e-10
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  0.1483752       Inf
## sample estimates:
## mean of x mean of y
## 0.5056497 0.3035382
```

I veiem que les variàncies són iguals, ja que p (0.07955) és més gran que 0.05, acceptant la hipòtesis nul·la de que les variancies son iguals.

En el primer t-test, veiem que p = 9.009e-10 < 0.05, per tant rebutjem la Hipòtesi nul·la, de manera que acceptem la Hipòtesi alternativa de que la mitjana de supervivència dels que van sols és dierent que la dels que van acompanyats, que vol dir que en un 95% de confiança obtenim que la mitjana de la supervivència dels que van sols és diferent a la mitjana de supervivència dels que van acompanyats.

Ara comprovem quina mitjana és més gran:

En el segon test on mirem si H1(1):mitjana sol > mitjana acompanyat, llavors veiem que p=1>0.05, per tant no podem rebutjar la hipòtesi nul·la que la mitjana dels que van sols en supervivència és igual que la dels que van acompanyats, però si que rebutjem que mitjana_sol>mitjana_acompanyat

En el tercer test on mirem si H1(1):mitjana acompanyat > mitjana sol, llavors veiem que p=4.505e-10<0.05 de manera que rebutjem la hipòtesi nul·la i acceptem l'hipòtesi alternativa que la mitjana dels que van acompanyats és més gran que la dels que van sols, que vol dir que en els que van sols hi ha més 0 ( no supervivents) i per tant, que els que van acompanyats tenien més probabilitat de sobreviure.

# 4.3.2 Regressió logarítmica:

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.5.3
```

```
## Loading required package: lattice
```

```
#Creem Model regressió
lm_1=glm(Survived~Pclass+SexR+Age+Embarked+SibSp,train,family=binomial())
lm_2=glm(Survived~Pclass+SexR+Age+Embarked+Family,train,family=binomial())
lm_3=glm(Survived~Pclass+SexR+Age+Embarked+Parch+SibSp,train,family=binomial())
lm_4=glm(Survived~Pclass+SexR+Agec+Embarked+Parch+SibSp,train,family=binomial())
lm_5=glm(Survived~Pclass+SexR+Agec3+Embarked+Parch+SibSp,train,family=binomial())
lm_6=glm(Survived~Pclass+SexR+Agec3+Embarked+SibSp,train,family=binomial())
lm_7=glm(Survived~Pclass+SexR+Agec3+Embarked+Family,train,family=binomial())
lm_8=glm(Survived~Pclass+SexR+Agec3+Embarked+Alone,train,family=binomial())
lm_9=glm(Survived~Pclass+SexR+Agec3+Embarked+Family+Fare,train,family=binomial())
lm_10=glm(Survived~Pclass+SexR+Agec3+Embarked+Alone+Fare,train,family=binomial()) #A
lone té relació oamb family, que aquest té relació amb SibSp i Parch
lm_11=glm(Survived~Pclass+SexR+Agec3+Embarked+Alone+Farebin,train,family=binomial())

lm_1b=glm(Survived~Pclass+SexR+Agec3+Embarked+SibSp,train,family=binomial())
lm_1c=glm(Survived~Pclass+SexR+Age+Embarked+Alone,train,family=binomial())
lm_1d=glm(Survived~Pclass+SexR+Age+Embarked+Alone+Fare,train,family=binomial())
lm_1e=glm(Survived~SexR+Age+Embarked+Alone+Fare,train,family=binomial())
lm_1eb=glm(Survived~SexR+Age+Embarked+Fare,train,family=binomial())

lm_12=glm(Survived~Pclass+SexR+Age+Embarked+SibSp+Coberta,train,family=binomial())



AIC(lm_1,lm_2,lm_3,lm_4,lm_5,lm_6,lm_7,lm_8,lm_9, lm_10, lm_11, lm_1b, lm_1c, lm_1
d, lm_1e, lm_1eb, lm_12)
```

```
##        df      AIC
## lm_1    8 801.0351
## lm_2    8 801.6301
## lm_3    9 802.3650
## lm_4    9 810.2041
## lm_5   10 799.2917
## lm_6    9 800.1173
## lm_7    9 798.5235
## lm_8    9 816.2568
## lm_9   10 797.8293
## lm_10  10 818.0439
## lm_11  14 819.3214
## lm_1b   9 800.1173
## lm_1c   8 811.9280
## lm_1d   9 813.9208
## lm_1e   7 883.0040
## lm_1eb  6 881.6489
## lm_12  16 799.6287
```

Veiem que el millor model és el lm_1e, de manera que l'utilitzem per fer les prediccions:

lm_1e:

```
#Mostrem model
lm_1e
```

```
##
## Call:  glm(formula = Survived ~ SexR + Age + Embarked + Alone + Fare,
##     family = binomial(), data = train)
##
## Coefficients:
## (Intercept)     SexRmale          Age    EmbarkedQ    EmbarkedS
##     1.75479     -2.45365     -0.01505     -0.62404     -0.67457
##      Alone1         Fare
##    -0.15478      0.01088
##
## Degrees of Freedom: 890 Total (i.e. Null);  884 Residual
## Null Deviance:       1187
## Residual Deviance: 869   AIC: 883
```

```
summary(lm_1e)
```

```
##
## Call:
## glm(formula = Survived ~ SexR + Age + Embarked + Alone + Fare,
##     family = binomial(), data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2892  -0.6178  -0.5358   0.7968   2.1684
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.754789   0.337380   5.201 1.98e-07 ***
## SexRmale    -2.453646   0.181860 -13.492  < 2e-16 ***
## Age         -0.015048   0.005999  -2.508  0.01213 *
## EmbarkedQ   -0.624038   0.350900  -1.778  0.07534 .
## EmbarkedS   -0.674572   0.221552  -3.045  0.00233 **
## Alone1      -0.154784   0.193748  -0.799  0.42435
## Fare         0.010882   0.002570   4.234 2.29e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1186.7  on 890  degrees of freedom
## Residual deviance:  869.0  on 884  degrees of freedom
## AIC: 883
##
## Number of Fisher Scoring iterations: 5
```

```
#Predim en train, indicant que volem les probabilitats
train$pred<-predict(lm_1e,train,interval='response')
test$pred<-predict(lm_1e,test,interval='response')


#Si probabilitat>0.55 llavors train_predita=1
train$pred[train$pred>0.55]<-1
train$pred[train$pred<=0.55]<-0

test$pred[test$pred>0.55]<-1
test$pred[test$pred<=0.55]<-0

train$Survived<- as.factor(train$Survived)
train$pred<-as.factor(train$pred)

matriu_conf<-confusionMatrix(train$pred,train$Survived)
matriu_conf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 475 127
##          1  74 215
##
##                Accuracy : 0.7744
##                  95% CI : (0.7455, 0.8015)
##     No Information Rate : 0.6162
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5087
##
##  Mcnemar's Test P-Value : 0.0002446
##
##             Sensitivity : 0.8652
##             Specificity : 0.6287
##          Pos Pred Value : 0.7890
##          Neg Pred Value : 0.7439
##              Prevalence : 0.6162
##          Detection Rate : 0.5331
##    Detection Prevalence : 0.6756
##       Balanced Accuracy : 0.7469
##
##        'Positive' Class : 0
##
```

Tot i semblar un molt bon model, obtenim una precisió de tant sols 77.44% en train. Al pujar les dades a kaggle, ens dóna una precisió del test de 77,03%. El model conté les variables sexe, age, embarked, alone i fare (pclass s'ha exclòs ja que té molta relació amb Fare i Sibsp i Parch formen part de alone ) Veiem que totes les variables són significatives almenys al 90% menys Alone1.

També hem provat de fer aquest mateix model però sense la variable Alone (lm_1eb), i no millorem ni el model (AIC) ni la precisió.

Provem de predir amb el següent model que ens ha donat millors resultats, lm_11:

```
#Mostrem
lm_11
```

```
##
## Call:  glm(formula = Survived ~ Pclass + SexR + Agec3 + Embarked + Alone +
##     Farebin, family = binomial(), data = train)
##
## Coefficients:
##  (Intercept)       Pclass2       Pclass3      SexRmale      Agec3Avi
##      2.71355      -0.78285      -1.98742      -2.67518      -0.68827
##     Agec3Nen     EmbarkedQ     EmbarkedS        Alone1  FarebinFare1
##      1.53428      -0.03450      -0.64075      -0.22596       0.57175
## FarebinFare2  FarebinFare3  FarebinFare4  FarebinFare5
##     -0.44109      -0.04708       0.21845       0.01136
##
## Degrees of Freedom: 890 Total (i.e. Null);  877 Residual
## Null Deviance:       1187
## Residual Deviance: 791.3     AIC: 819.3
```

```
summary(lm_11)
```

```
##
## Call:
## glm(formula = Survived ~ Pclass + SexR + Agec3 + Embarked + Alone +
##     Farebin, family = binomial(), data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6133  -0.6718  -0.3825   0.6323   2.3930
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.71355    0.46266   5.865 4.49e-09 ***
## Pclass2      -0.78285    0.34367  -2.278  0.02273 *
## Pclass3      -1.98742    0.34675  -5.732 9.95e-09 ***
## SexRmale     -2.67518    0.20311 -13.171  < 2e-16 ***
## Agec3Avi     -0.68827    0.44916  -1.532  0.12544
## Agec3Nen      1.53428    0.34796   4.409 1.04e-05 ***
## EmbarkedQ    -0.03450    0.38182  -0.090  0.92801
## EmbarkedS    -0.64075    0.23928  -2.678  0.00741 **
## Alone1       -0.22596    0.26461  -0.854  0.39314
## FarebinFare1  0.57175    0.46287   1.235  0.21674
## FarebinFare2 -0.44109    0.55024  -0.802  0.42277
## FarebinFare3 -0.04708    0.47281  -0.100  0.92068
## FarebinFare4  0.21845    0.58212   0.375  0.70746
## FarebinFare5  0.01136    0.60679   0.019  0.98506
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1186.66  on 890  degrees of freedom
## Residual deviance:  791.32  on 877  degrees of freedom
## AIC: 819.32
##
## Number of Fisher Scoring iterations: 5
```

```
#Predim en train, indicant que volem les probabilitats
train$pred<-predict(lm_11,train,interval='response')
test$pred<-predict(lm_11,test,interval='response')

#Posant un llindar del 50%, provem:

#Si probabilitat>0.5 llavors train_predita=1
train$pred[train$pred>0.50]<-1
train$pred[train$pred<=0.50]<-0

test$pred[test$pred>0.50]<-1
test$pred[test$pred<=0.50]<-0

train$Survived<- as.factor(train$Survived)
train$pred<-as.factor(train$pred)

matriu_conf<-confusionMatrix(train$pred,train$Survived)
matriu_conf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 518 144
##          1  31 198
##
##                Accuracy : 0.8036
##                  95% CI : (0.776, 0.8292)
##     No Information Rate : 0.6162
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5572
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9435
##             Specificity : 0.5789
##          Pos Pred Value : 0.7825
##          Neg Pred Value : 0.8646
##              Prevalence : 0.6162
##          Detection Rate : 0.5814
##    Detection Prevalence : 0.7430
##       Balanced Accuracy : 0.7612
##
##        'Positive' Class : 0
##
```

En aquest model veiem que la variable Fare no és significativa, tot i que obtenim una millor precisió en train (80.47%).A kaggle obtenim una predicció de 77.03%, la mateixa que amb l'anterior.

També provem de predir amb lm_10

```
lm_10
```

```
##
## Call:  glm(formula = Survived ~ Pclass + SexR + Agec3 + Embarked + Alone +
##     Fare, family = binomial(), data = train)
##
## Coefficients:
## (Intercept)      Pclass2      Pclass3     SexRmale     Agec3Avi
##    2.716201    -0.804537    -2.018139    -2.649708    -0.683568
##    Agec3Nen    EmbarkedQ    EmbarkedS       Alone1         Fare
##    1.357713     0.003904    -0.566281    -0.215124     0.001021
##
## Degrees of Freedom: 890 Total (i.e. Null);  881 Residual
## Null Deviance:       1187
## Residual Deviance: 798   AIC: 818
```

```
summary(lm_10)
```

```
##
## Call:
## glm(formula = Survived ~ Pclass + SexR + Agec3 + Embarked + Alone +
##     Fare, family = binomial(), data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6377  -0.6963  -0.3953   0.6759   2.3616
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.716201   0.351749   7.722 1.15e-14 ***
## Pclass2     -0.804537   0.283797  -2.835  0.00458 **
## Pclass3     -2.018139   0.269269  -7.495 6.64e-14 ***
## SexRmale    -2.649708   0.198952 -13.318  < 2e-16 ***
## Agec3Avi    -0.683568   0.436114  -1.567  0.11702
## Agec3Nen     1.357713   0.332332   4.085 4.40e-05 ***
## EmbarkedQ    0.003904   0.370839   0.011  0.99160
## EmbarkedS   -0.566281   0.235055  -2.409  0.01599 *
## Alone1      -0.215124   0.207339  -1.038  0.29948
## Fare         0.001021   0.002236   0.457  0.64779
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1186.66  on 890  degrees of freedom
## Residual deviance:  798.04  on 881  degrees of freedom
## AIC: 818.04
##
## Number of Fisher Scoring iterations: 5
```

```
#Predim en train, indicant que volem les probabilitats
train$pred<-predict(lm_10,train,interval='response')
test$pred<-predict(lm_10,test,interval='response')




#Si probabilitat>0.65 llavors train_predita=1
train$pred[train$pred>0.65]<-1
train$pred[train$pred<=0.65]<-0

test$pred[test$pred>0.65]<-1
test$pred[test$pred<=0.65]<-0

train$Survived<- as.factor(train$Survived)
train$pred<-as.factor(train$pred)

matriu_conf<-confusionMatrix(train$pred,train$Survived)
matriu_conf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 521 146
##          1  28 196
##
##                Accuracy : 0.8047
##                  95% CI : (0.7771, 0.8303)
##     No Information Rate : 0.6162
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5584
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9490
##             Specificity : 0.5731
##          Pos Pred Value : 0.7811
##          Neg Pred Value : 0.8750
##              Prevalence : 0.6162
##          Detection Rate : 0.5847
##    Detection Prevalence : 0.7486
##       Balanced Accuracy : 0.7610
##
##        'Positive' Class : 0
##
```

Obtenim una precisió en train del 80.58% i a kaggle el test obté una puntuació de 0.77511, més alta que lm_11. En aquest cas veiem que Fare tampoc és significatiu.

Provem de predir amb lm_1:

```
lm_1
```

```
##
## Call:  glm(formula = Survived ~ Pclass + SexR + Age + Embarked + SibSp,
##     family = binomial(), data = train)
##
## Coefficients:
## (Intercept)      Pclass2      Pclass3     SexRmale          Age
##     4.15708     -1.04227     -2.22817     -2.69659     -0.03506
##   EmbarkedQ    EmbarkedS        SibSp
##     0.15375     -0.49886     -0.32290
##
## Degrees of Freedom: 890 Total (i.e. Null);  883 Residual
## Null Deviance:       1187
## Residual Deviance: 785   AIC: 801
```

```
summary(lm_1)
```

```
##
## Call:
## glm(formula = Survived ~ Pclass + SexR + Age + Embarked + SibSp,
##     family = binomial(), data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5701  -0.6216  -0.3959   0.6115   2.4632
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  4.157083   0.405807  10.244  < 2e-16 ***
## Pclass2     -1.042273   0.269575  -3.866  0.00011 ***
## Pclass3     -2.228168   0.249110  -8.945  < 2e-16 ***
## SexRmale    -2.696593   0.195436 -13.798  < 2e-16 ***
## Age         -0.035055   0.006915  -5.069    4e-07 ***
## EmbarkedQ    0.153746   0.385033   0.399  0.68967
## EmbarkedS   -0.498861   0.236498  -2.109  0.03491 *
## SibSp       -0.322898   0.105251  -3.068  0.00216 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1186.66  on 890  degrees of freedom
## Residual deviance:  785.04  on 883  degrees of freedom
## AIC: 801.04
##
## Number of Fisher Scoring iterations: 5
```

```
#Predim en train, indicant que volem les probabilitats
train$pred<-predict(lm_1,train,interval='response')
test$pred<-predict(lm_1,test,interval='response')




#Si probabilitat>0.55 llavors train_predita=1
train$pred[train$pred>0.55]<-1
train$pred[train$pred<=0.55]<-0

test$pred[test$pred>0.55]<-1
test$pred[test$pred<=0.55]<-0

train$Survived<- as.factor(train$Survived)
train$pred<-as.factor(train$pred)

matriu_conf<-confusionMatrix(train$pred,train$Survived)
matriu_conf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 513 135
##          1  36 207
##
##                Accuracy : 0.8081
##                  95% CI : (0.7807, 0.8334)
##     No Information Rate : 0.6162
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5708
##
##  Mcnemar's Test P-Value : 6.668e-14
##
##             Sensitivity : 0.9344
##             Specificity : 0.6053
##          Pos Pred Value : 0.7917
##          Neg Pred Value : 0.8519
##              Prevalence : 0.6162
##          Detection Rate : 0.5758
##    Detection Prevalence : 0.7273
##       Balanced Accuracy : 0.7698
##
##        'Positive' Class : 0
##
```

En aquest model obtenim una precisió en train de 81.14% i en test ( kaggle) de 0.7799. (La millor puntuació del model feta fins ara, tot i no tenir un AIC de les més altes.)

En aquest cas hem utilitzat les variables pclass, Sex, Age, Embarked i SibSp, on totes són significatives al 95%, menys EmbarkedQ, que això no vol dir que la variable Embarked no sigui significativa, sinó que EmbarkedQ en relació a la variable base d'Embarqued (EmbarquedC), no és significativa, però per exemple veiem que si que és significativa la diferència entre Embarked C i embarkedS.

Provem de predir amb el model lm_8:

```
lm_8
```

```
##
## Call:  glm(formula = Survived ~ Pclass + SexR + Agec3 + Embarked + Alone,
##     family = binomial(), data = train)
##
## Coefficients:
## (Intercept)      Pclass2      Pclass3     SexRmale     Agec3Avi
##    2.792943    -0.856982    -2.078172    -2.652464    -0.688851
##    Agec3Nen    EmbarkedQ    EmbarkedS       Alone1
##    1.356792    -0.005933    -0.579379    -0.191188
##
## Degrees of Freedom: 890 Total (i.e. Null);  882 Residual
## Null Deviance:       1187
## Residual Deviance: 798.3     AIC: 816.3
```

```
summary(lm_8)
```

```
##
## Call:
## glm(formula = Survived ~ Pclass + SexR + Agec3 + Embarked + Alone,
##     family = binomial(), data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6125  -0.6956  -0.3940   0.6770   2.3549
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.792943   0.309935   9.011  < 2e-16 ***
## Pclass2     -0.856982   0.259716  -3.300 0.000968 ***
## Pclass3     -2.078172   0.235377  -8.829  < 2e-16 ***
## SexRmale    -2.652464   0.198666 -13.351  < 2e-16 ***
## Agec3Avi    -0.688851   0.436094  -1.580 0.114200
## Agec3Nen     1.356792   0.332413   4.082 4.47e-05 ***
## EmbarkedQ   -0.005933   0.370258  -0.016 0.987216
## EmbarkedS   -0.579379   0.233158  -2.485 0.012958 *
## Alone1      -0.191188   0.200287  -0.955 0.339795
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1186.66  on 890  degrees of freedom
## Residual deviance:  798.26  on 882  degrees of freedom
## AIC: 816.26
##
## Number of Fisher Scoring iterations: 5
```

```
#Predim en train, indicant que volem les probabilitats
train$pred<-predict(lm_8,train,interval='response')
test$pred<-predict(lm_8,test,interval='response')




#Si probabilitat>0.55 llavors train_predita=1
train$pred[train$pred>0.55]<-1
train$pred[train$pred<=0.55]<-0

test$pred[test$pred>0.55]<-1
test$pred[test$pred<=0.55]<-0

train$Survived<- as.factor(train$Survived)
train$pred<-as.factor(train$pred)

matriu_conf<-confusionMatrix(train$pred,train$Survived)
matriu_conf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 520 146
##          1  29 196
##
##                Accuracy : 0.8036
##                  95% CI : (0.776, 0.8292)
##     No Information Rate : 0.6162
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5561
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9472
##             Specificity : 0.5731
##          Pos Pred Value : 0.7808
##          Neg Pred Value : 0.8711
##              Prevalence : 0.6162
##          Detection Rate : 0.5836
##    Detection Prevalence : 0.7475
##       Balanced Accuracy : 0.7601
##
##        'Positive' Class : 0
##
```

Aquest model té una precisió de 80,47% en train i de 77,51% en test (kaggle), per tant no és la millor precisió que podem obtenir. Veiem que Alone no és significatiu i que Agec Avi tampoc (al 95%).

Provem de predir amb lm_1d:

```
lm_1d
```

```
##
## Call:  glm(formula = Survived ~ Pclass + SexR + Age + Embarked + Alone +
##     Fare, family = binomial(), data = train)
##
## Coefficients:
## (Intercept)      Pclass2      Pclass3     SexRmale          Age
##   3.8637143   -0.9670387   -2.2284378   -2.6081023   -0.0307237
##   EmbarkedQ    EmbarkedS       Alone1         Fare
##   0.1384692   -0.5673501   -0.0887071    0.0001881
##
## Degrees of Freedom: 890 Total (i.e. Null);  882 Residual
## Null Deviance:       1187
## Residual Deviance: 795.9     AIC: 813.9
```

```
summary(lm_1d)
```

```
##
## Call:
## glm(formula = Survived ~ Pclass + SexR + Age + Embarked + Alone +
##     Fare, family = binomial(), data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5360  -0.6615  -0.3933   0.6325   2.4827
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  3.8637143  0.4661475   8.289  < 2e-16 ***
## Pclass2     -0.9670387  0.2913162  -3.320 0.000902 ***
## Pclass3     -2.2284378  0.2842453  -7.840 4.51e-15 ***
## SexRmale    -2.6081023  0.1970130 -13.238  < 2e-16 ***
## Age         -0.0307237  0.0066403  -4.627 3.71e-06 ***
## EmbarkedQ    0.1384692  0.3778157   0.366 0.713993
## EmbarkedS   -0.5673501  0.2366572  -2.397 0.016514 *
## Alone1      -0.0887071  0.1967124  -0.451 0.652027
## Fare         0.0001881  0.0022190   0.085 0.932455
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1186.66  on 890  degrees of freedom
## Residual deviance:  795.92  on 882  degrees of freedom
## AIC: 813.92
##
## Number of Fisher Scoring iterations: 5
```

```
#Predim en train, indicant que volem les probabilitats
train$pred<-predict(lm_1d,train,interval='response')
test$pred<-predict(lm_1d,test,interval='response')




#Si probabilitat>0.6 llavors train_predita=1
train$pred[train$pred>0.60]<-1
train$pred[train$pred<=0.60]<-0

test$pred[test$pred>0.60]<-1
test$pred[test$pred<=0.60]<-0

train$Survived<- as.factor(train$Survived)
train$pred<-as.factor(train$pred)

matriu_conf<-confusionMatrix(train$pred,train$Survived)
matriu_conf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 516 151
##          1  33 191
##
##                Accuracy : 0.7935
##                  95% CI : (0.7654, 0.8196)
##     No Information Rate : 0.6162
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.533
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9399
##             Specificity : 0.5585
##          Pos Pred Value : 0.7736
##          Neg Pred Value : 0.8527
##              Prevalence : 0.6162
##          Detection Rate : 0.5791
##    Detection Prevalence : 0.7486
##       Balanced Accuracy : 0.7492
##
##        'Positive' Class : 0
##
```

En aquest cas obtenim una precisió en train de 79.46% i una precisió en test del 76.07% i per tant tampoc és una de les millors precisions.

Per últim, provem de predir amb lm_1c:

```
lm_1c
```

```
##
## Call:  glm(formula = Survived ~ Pclass + SexR + Age + Embarked + Alone,
##     family = binomial(), data = train)
##
## Coefficients:
## (Intercept)      Pclass2      Pclass3     SexRmale          Age
##     3.87993     -0.97705     -2.24014     -2.60874     -0.03077
##   EmbarkedQ    EmbarkedS       Alone1
##     0.13690     -0.56980     -0.08474
##
## Degrees of Freedom: 890 Total (i.e. Null);  883 Residual
## Null Deviance:       1187
## Residual Deviance: 795.9     AIC: 811.9
```

```
summary(lm_1c)
```

```
##
## Call:
## glm(formula = Survived ~ Pclass + SexR + Age + Embarked + Alone,
##     family = binomial(), data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5319  -0.6620  -0.3928   0.6319   2.4871
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  3.879926   0.425328   9.122  < 2e-16 ***
## Pclass2     -0.977054   0.266307  -3.669 0.000244 ***
## Pclass3     -2.240136   0.248574  -9.012  < 2e-16 ***
## SexRmale    -2.608740   0.196845 -13.253  < 2e-16 ***
## Age         -0.030770   0.006619  -4.649 3.34e-06 ***
## EmbarkedQ    0.136897   0.377371   0.363 0.716781
## EmbarkedS   -0.569803   0.234853  -2.426 0.015257 *
## Alone1      -0.084740   0.191019  -0.444 0.657318
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1186.66  on 890  degrees of freedom
## Residual deviance:  795.93  on 883  degrees of freedom
## AIC: 811.93
##
## Number of Fisher Scoring iterations: 5
```

```
#Predim en train, indicant que volem les probabilitats
train$pred<-predict(lm_1c,train,interval='response')
test$pred<-predict(lm_1c,test,interval='response')




#Si probabilitat>0.5 llavors train_predita=1
train$pred[train$pred>0.50]<-1
train$pred[train$pred<=0.50]<-0

test$pred[test$pred>0.50]<-1
test$pred[test$pred<=0.50]<-0

train$Survived<- as.factor(train$Survived)
train$pred<-as.factor(train$pred)

matriu_conf<-confusionMatrix(train$pred,train$Survived)
matriu_conf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 511 143
##          1  38 199
##
##                Accuracy : 0.7969
##                  95% CI : (0.7689, 0.8228)
##     No Information Rate : 0.6162
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5442
##
##  Mcnemar's Test P-Value : 1.073e-14
##
##             Sensitivity : 0.9308
##             Specificity : 0.5819
##          Pos Pred Value : 0.7813
##          Neg Pred Value : 0.8397
##              Prevalence : 0.6162
##          Detection Rate : 0.5735
##    Detection Prevalence : 0.7340
##       Balanced Accuracy : 0.7563
##
##        'Positive' Class : 0
##
```

Obtenim una precisió en train del 79.8% i en test del 77,99%, una de les millors precisions en test. Tot i així, veiem que la variable Alone no és significativa.

Per tant els millors models en quant a precisió de test són lm_1c i lm_1.

Veiem que la AIC de lm_1c és 813.59 i la AIC de lm_1 és 803.1. La principal diferència entre els dos models és que un utilitza Alone (que surt de family, que és la suma de Sibsp i Parch) i l'altre SibSp . Les variables en comú són Pclass, SexR, Age i Embarked (Fare no s'utilitza ja que utilitzem Pclass, i fent aquest model utilitzant fare enlloc de pclass ens surt més AIC però menys precisió en test).

Tot i que la AIC de lm_1c és més alta que lm_1, té la variable Alone que no és explicativa, ja que no és significativa ( si Alone =1 no té diferències de Alone=0, com que només té dos grups aquesta variable, llavors ens està dient que tampoc és explicativa del model).

Veiem que en lm_1 obtenim la millor precisió en train (81.14%) enfront del 79.91% de lm1_1c, tot i que la precisió en test és la mateixa ( 77)

Veiem que en lm_1 es classifiquen 514 de 648 "0" correctament (79.32%) i 134 "0" es classifiquen incorrectament com a "1" (20.68%). Dels "1", 35 de 243 estan mal classificats com a "0" (14.4%)i 208 estan ben classificats (85.6%). Per tant, veiem que classifica millor els 1 que els 0, però pot ser degut a que el llindar era 55%.

Veiem que en lm_1c es classifiquen 512 de 654 "0" correctament (78.28%) i 142 "0" es classifiquen incorrectament com a "1" (21.72%). Dels "1", 37 de 237 estan mal classificats com a "0" (15.61%)i 200 estan ben classificats (84.39%). Per tant, veiem que classifica millor els 1 que els 0, tot i que el llindar és del 50%.

Com que veiem que lm_1 classifica millor, expliquem aquest model :

Aquest model utilitza les variables Pclass, SexR, Age, Embarked i SibSp. Veiem que totes les variables menys EmbarkedQ són significatives al 95%, cosa que vol dir que són explicatives del model.

En els resultats dels coeficients, veiem que per exemple, ser home baixa la probabilitat de sobreviure, ja que el seu pendent és -2.69 . També veiem que la probabilitat de sobreviure de la classe 2 és més baixa que la classe 1 ( pendent -1.05), i la classe 3 encara té menys probabilitats de sobreviure que la 2 (-2.23 respecte la 1).

En quant a l'embarcament, veiem que els embarcats a C tenen menys probabilitats de sobreviure que els embarcats a Q (0.09), tot i que al no ser significatiu no sabem si és així, el que sí que sabem és que els embarcats a S tenen menys probabilitats de sobreviure que els embarcats a C(-0.50).

També veiem que a més edat, menys probabilitat de sobreviure ( -0.03) o que també com més familia es tingui (Sibsp), menys probabilitats de sobreviure (-0.32)

# 4.3.3 Arbre

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.5.3
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

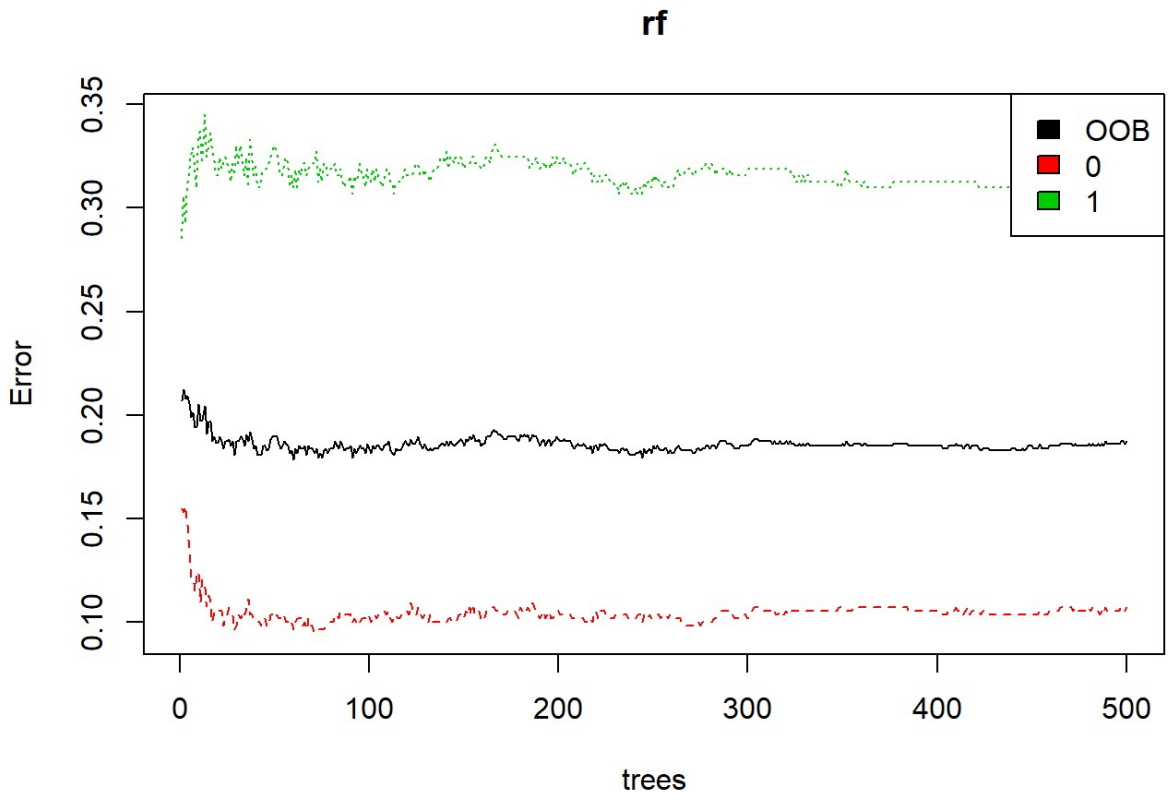
```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
equacio <- "Survived ~ Pclass + SexR + Age + SibSp + Parch + Embarked"
formula <- as.formula(equacio)


#Creem arbre
rf<- randomForest(formula=formula, data=train, ntree=500, mtry=3,nodesize=0.01*nrow
(train))
#Predim resultats
train$pred_arbre<-predict(rf,train)


#Al predir el test, ens surt error que type of predictors in new data do not match,
#Per tant primer igualem
levels(test$Age) <- levels(train$Age)
levels(test$SexF) <- levels(train$SexF)
levels(test$Pclass) <- levels(train$Pclass)
levels(test$Embarked) <- levels(train$Embarked)


#Predim en test
test$pred_arbre<- predict(rf,newdata=test)

rf
```

```
##
## Call:
##  randomForest(formula = formula, data = train, ntree = 500, mtry = 3,      nodesi
ze = 0.01 * nrow(train))
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 18.74%
## Confusion matrix:
##     0   1 class.error
## 0 490  59   0.1074681
## 1 108 234   0.3157895
```

```
plot (rf)
legend('topright', colnames(rf$err.rate), col=1:3, fill=1:3)
```




```
matriu_conf<-confusionMatrix(train$pred_arbre,train$Survived)
matriu_conf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 521  82
##          1  28 260
##
##                Accuracy : 0.8765
##                  95% CI : (0.8531, 0.8974)
##     No Information Rate : 0.6162
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.731
##
##  Mcnemar's Test P-Value : 4.341e-07
##
##             Sensitivity : 0.9490
##             Specificity : 0.7602
##          Pos Pred Value : 0.8640
##          Neg Pred Value : 0.9028
##              Prevalence : 0.6162
##          Detection Rate : 0.5847
##    Detection Prevalence : 0.6768
##       Balanced Accuracy : 0.8546
##
##        'Positive' Class : 0
##
```

La precisió d'aquest model en test és del 77.033% (kaggle) i en train és del 87.77%. Provem de fer altres models:

```
#Equacions i formules
equacio1 <- "Survived ~ Pclass + SexR + Age + SibSp + Parch + Embarked+ Fare"
formula1 <- as.formula(equacio1)

equacio2 <- "Survived ~ Pclass + Sex + Age + SibSp + Parch + Embarked"
formula2 <- as.formula(equacio2)

equacio3 <- "Survived ~ Pclass + Sex + Age + Family + Embarked"
formula3 <- as.formula(equacio3)

equacio4 <- "Survived ~ Pclass + Sex + Age + Alone + Embarked"
formula4 <- as.formula(equacio4)

equacio5 <- "Survived ~ Pclass + Sex + Age + Alone + Embarked +Fare"
formula5 <- as.formula(equacio5)

equacio6 <- "Survived ~  Sex + Alone + Embarked + Fare"
formula6 <- as.formula(equacio6)

equacio7 <- "Survived ~ Pclass + SexR + Age + Alone + Embarked +Fare"
formula7 <- as.formula(equacio7)

equacio8 <- "Survived ~ Pclass + SexR + Age + Alone + Embarked "
formula8 <- as.formula(equacio8)

equacio9 <- "Survived ~ Pclass + SexR + Age + SibSp + Parch + Embarked+ Fare+Cobert
a"
formula9 <- as.formula(equacio8)

#Creem arbres
rf1<- randomForest(formula=formula1, data=train, ntree=500, mtry=3,nodesize=0.01*nro
w(train))

rf2<- randomForest(formula=formula2, data=train, ntree=500, mtry=3,nodesize=0.01*nro
w(train))

rf3<- randomForest(formula=formula3, data=train, ntree=500, mtry=3,nodesize=0.01*nro
w(train))

rf4<- randomForest(formula=formula4, data=train, ntree=500, mtry=3,nodesize=0.01*nro
w(train))

rf5<- randomForest(formula=formula5, data=train, ntree=500, mtry=3,nodesize=0.01*nro
w(train))

rf6<- randomForest(formula=formula6, data=train, ntree=500, mtry=3,nodesize=0.01*nro
w(train))

rf7<- randomForest(formula=formula7, data=train, ntree=500, mtry=3,nodesize=0.01*nro
w(train))

rf8<- randomForest(formula=formula8, data=train, ntree=500, mtry=3,nodesize=0.01*nro
```

```
w(train))

rf9<- randomForest(formula=formula9, data=train, ntree=500, mtry=3,nodesize=0.01*nro
w(train))


mean(rf$err.rate)
```

```
## [1] 0.2023208
```

```
mean(rf1$err.rate)
```

```
## [1] 0.1841298
```

```
mean(rf2$err.rate)
```

```
## [1] 0.1949982
```

```
mean(rf3$err.rate)
```

```
## [1] 0.2028992
```

```
mean(rf4$err.rate)
```

```
## [1] 0.2206217
```

```
mean(rf5$err.rate)
```

```
## [1] 0.1922454
```

```
mean(rf6$err.rate)
```

```
## [1] 0.2198403
```

```
mean(rf7$err.rate)
```

```
## [1] 0.196523
```

```
mean(rf8$err.rate)
```

```
## [1] 0.2150754
```

```
mean(rf9$err.rate)
```

```
## [1] 0.2147388
```

Veiem que rf1, rf2 , rf6 i rf8 són els que tenen menys error, els mirem:

```
rf1
```

```
##
## Call:
##  randomForest(formula = formula1, data = train, ntree = 500, mtry = 3,      nodes
ize = 0.01 * nrow(train))
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 16.61%
## Confusion matrix:
##     0   1 class.error
## 0 499  50  0.09107468
## 1  98 244  0.28654971
```

```
rf2
```

```
##
## Call:
##  randomForest(formula = formula2, data = train, ntree = 500, mtry = 3,      nodes
ize = 0.01 * nrow(train))
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 17.85%
## Confusion matrix:
##     0   1 class.error
## 0 496  53  0.09653916
## 1 106 236  0.30994152
```

```
rf6
```

```
##
## Call:
##  randomForest(formula = formula6, data = train, ntree = 500, mtry = 3,      nodes
ize = 0.01 * nrow(train))
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 20.88%
## Confusion matrix:
##     0   1 class.error
## 0 479  70   0.1275046
## 1 116 226   0.3391813
```

```
rf8
```

```
##
## Call:
##  randomForest(formula = formula8, data = train, ntree = 500, mtry = 3,      nodes
ize = 0.01 * nrow(train))
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 19.53%
## Confusion matrix:
##     0   1 class.error
## 0 496  53  0.09653916
## 1 121 221  0.35380117
```

Veiem que rf1 és el que té un OOB estimate of error rate més baix (17.17%)

Veiem que rf1 classifica bé 498 “0” (90.71%) i en classifica malament 51 (9.29%) i que classifica 240 “1” correctament(70.18%) i en classifica malament 102 (29.82%).

Provem aquest mateix model canviant alguns paràmetres:

```
rf1a<- randomForest(formula=formula1, data=train, ntree=1000, mtry=3,nodesize=0.01*n
row(train))

rf1b<- randomForest(formula=formula1, data=train, ntree=1000, mtry=4,nodesize=0.01*n
row(train))

rf1c<- randomForest(formula=formula1, data=train, ntree=500, mtry=4,nodesize=0.01*nr
ow(train))

rf1d<- randomForest(formula=formula1, data=train, ntree=500, mtry=4)

rf1e<- randomForest(formula=formula1, data=train, ntree=500, mtry=4, importance=TRU
E)

rf1a
```

```
##
## Call:
##  randomForest(formula = formula1, data = train, ntree = 1000,      mtry = 3, node
size = 0.01 * nrow(train))
##                Type of random forest: classification
##                      Number of trees: 1000
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 17.06%
## Confusion matrix:
##     0   1 class.error
## 0 497  52  0.09471767
## 1 100 242  0.29239766
```

```
rf1b
```

```
##
## Call:
##  randomForest(formula = formula1, data = train, ntree = 1000,      mtry = 4, node
size = 0.01 * nrow(train))
##                Type of random forest: classification
##                      Number of trees: 1000
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 16.72%
## Confusion matrix:
##     0   1 class.error
## 0 495  54  0.09836066
## 1  95 247  0.27777778
```

```
rf1c
```

```
##
## Call:
##  randomForest(formula = formula1, data = train, ntree = 500, mtry = 4,      nodes
ize = 0.01 * nrow(train))
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 16.84%
## Confusion matrix:
##     0   1 class.error
## 0 496  53  0.09653916
## 1  97 245  0.28362573
```

```
rf1d
```

```
##
## Call:
##  randomForest(formula = formula1, data = train, ntree = 500, mtry = 4)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 17.62%
## Confusion matrix:
##     0   1 class.error
## 0 485  64   0.1165756
## 1  93 249   0.2719298
```

```
rf1e
```

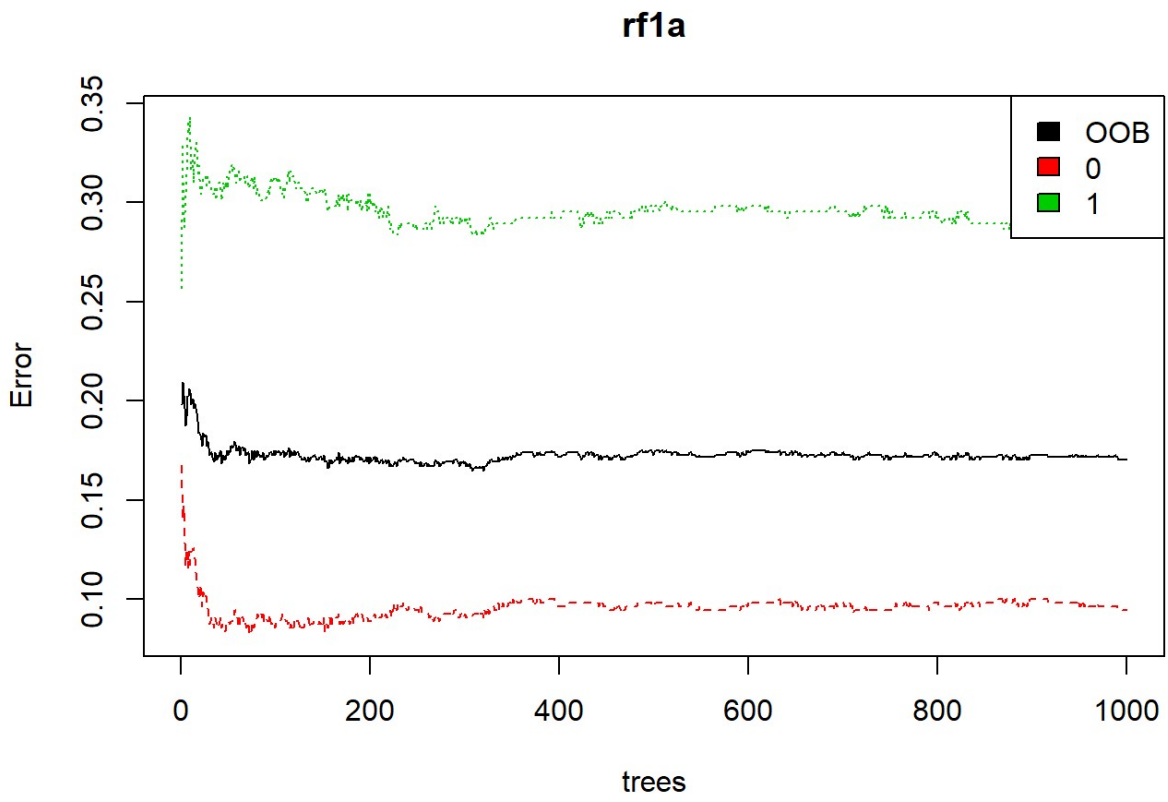
```
##
## Call:
##  randomForest(formula = formula1, data = train, ntree = 500, mtry = 4,      impor
tance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 17.28%
## Confusion matrix:
##     0   1 class.error
## 0 484  65   0.1183971
## 1  89 253   0.2602339
```

Veiem que rf1a és el que té un error rate més baix, del 17.28%, per tant l'utilitzem per fer les prediccions.

```
#Predim resultats en train
train$pred_arbre<-predict(rf1a,train)


#Al predir el test, ens surt error que type of predictors in new data do not match,
#Per tant primer igualem
levels(test$Age) <- levels(train$Age)
levels(test$SexF) <- levels(train$SexF)
levels(test$Pclass) <- levels(train$Pclass)
levels(test$Embarked) <- levels(train$Embarked)


#Predim en test
test$pred_arbre<- predict(rf1a,newdata=test)

#Mostrem model
rf1a
```

```
##
## Call:
##  randomForest(formula = formula1, data = train, ntree = 1000,      mtry = 3, node
size = 0.01 * nrow(train))
##                Type of random forest: classification
##                      Number of trees: 1000
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 17.06%
## Confusion matrix:
##     0   1 class.error
## 0 497  52  0.09471767
## 1 100 242  0.29239766
```

```
plot (rf1a)
legend('topright', colnames(rf1a$err.rate), col=1:3, fill=1:3)
```




```
#Mostrem precisió en train
matriu_conf<-confusionMatrix(train$pred_arbre,train$Survived)
matriu_conf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 529  63
##          1  20 279
##
##                Accuracy : 0.9068
##                  95% CI : (0.8858, 0.9251)
##     No Information Rate : 0.6162
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7983
##
##  Mcnemar's Test P-Value : 4.025e-06
##
##             Sensitivity : 0.9636
##             Specificity : 0.8158
##          Pos Pred Value : 0.8936
##          Neg Pred Value : 0.9331
##              Prevalence : 0.6162
##          Detection Rate : 0.5937
##    Detection Prevalence : 0.6644
##       Balanced Accuracy : 0.8897
##
##        'Positive' Class : 0
##
```

Veiem que la precisió en train és del 90.68%. i que classifica correctament els “0” en un 89,49% i els “1” en un 93.02%.

Tot i així, al estar comparant una predicció que s’ha fet amb les mateixes dades d’entrenament, hem de mirar l’oob score: tenim un 17.28% d’error.

Tot i haver millorat la precisió en train, veiem que la precisió en test no ha augmentat : 77.03% en kaggle.

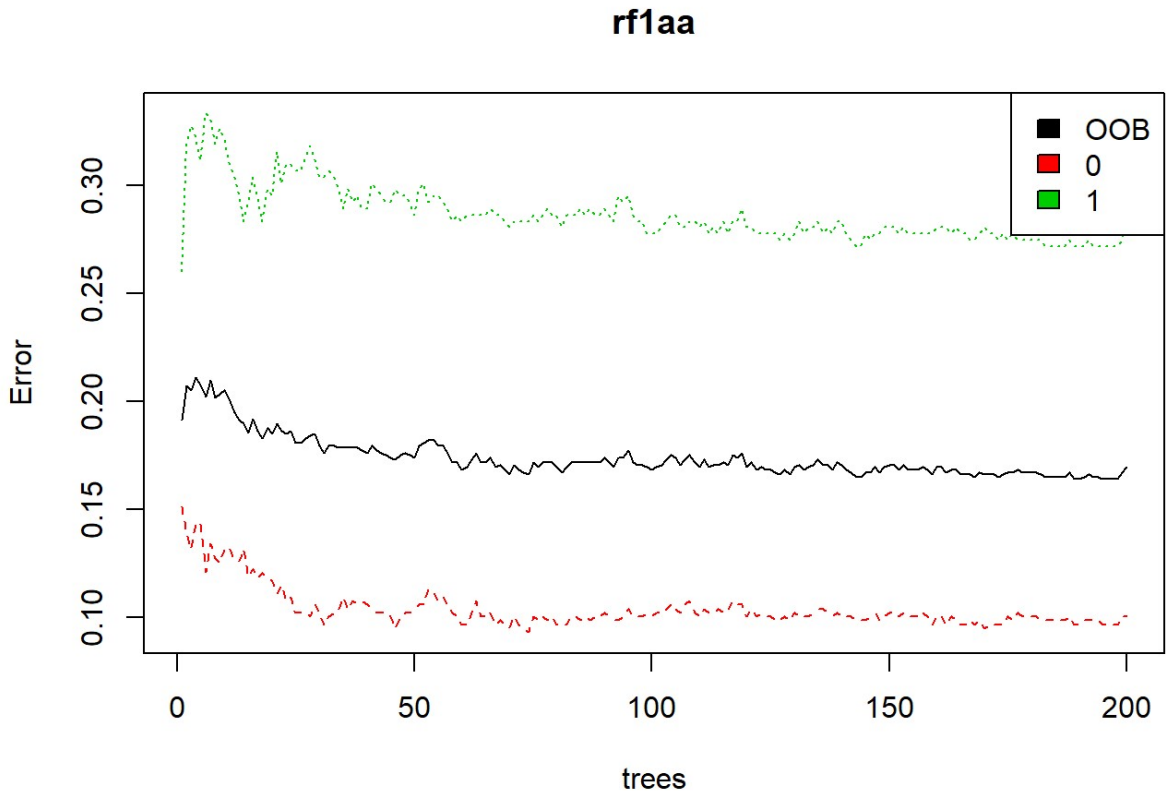
Amb això el que pensem és que hi ha hagut sobreentrenament de les dades i que el model sap explicar molt bé train però al que li canvien les dades ja no ho explica bé.

Provem amb el mateix model i menys arbres:

```
rf1aa<- randomForest(formula=formula1, data=train, ntree=200, mtry=3,nodesize=0.01*n
row(train))


#Predim resultats en train
train$pred_arbre<-predict(rf1aa,train)


#Al predir el test, ens surt error que type of predictors in new data do not match,
#Per tant primer igualem
levels(test$Age) <- levels(train$Age)
levels(test$SexF) <- levels(train$SexF)
levels(test$Pclass) <- levels(train$Pclass)
levels(test$Embarked) <- levels(train$Embarked)


#Predim en test
test$pred_arbre<- predict(rf1aa,newdata=test)

#Mostrem model
rf1aa
```

```
##
## Call:
##  randomForest(formula = formula1, data = train, ntree = 200, mtry = 3,      nodes
ize = 0.01 * nrow(train))
##                Type of random forest: classification
##                      Number of trees: 200
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 16.95%
## Confusion matrix:
##     0   1 class.error
## 0 494  55   0.1001821
## 1  96 246   0.2807018
```

```
plot (rf1aa)
legend('topright', colnames(rf1aa$err.rate), col=1:3, fill=1:3)
```




Amb aquest model amb menys arbres, veiem que la precisió en test ha baixat : 75.598%.

Finalment, provem de fer un arbre condicional:

```
library(party)
```

```
## Warning: package 'party' was built under R version 3.5.3
```

```
## Loading required package: grid
```

```
## Loading required package: mvtnorm
```

```
## Warning: package 'mvtnorm' was built under R version 3.5.2
```

```
## Loading required package: modeltools
```

```
## Warning: package 'modeltools' was built under R version 3.5.2
```

```
## Loading required package: stats4
```

```
##
## Attaching package: 'modeltools'
```

```
## The following object is masked from 'package:car':
##
##     Predict
```

```
## Loading required package: strucchange
```

```
## Warning: package 'strucchange' was built under R version 3.5.3
```

```
## Loading required package: zoo
```

```
## Warning: package 'zoo' was built under R version 3.5.3
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: sandwich
```

```
## Warning: package 'sandwich' was built under R version 3.5.3
```

```
forest<-cforest(Survived~Family+Pclass+Sex+Age+SibSp+Parch+Fare+Embarked,data=train,
controls=cforest_unbiased(ntree=500, mtry=3))
```

```
forest
```

```
##
##   Random Forest using Conditional Inference Trees
##
## Number of trees:  500
##
## Response:  Survived
## Inputs:  Family, Pclass, Sex, Age, SibSp, Parch, Fare, Embarked
## Number of observations:  891
```

```
#Predim en test
test$pred_arbre<- predict(forest,newdata=test)
```

Amb aquesta predicció obtenim un 77.99% de precisió en test (kaggle).

# 5. Representació dels resultats a partir de

# taules i gràfiques.

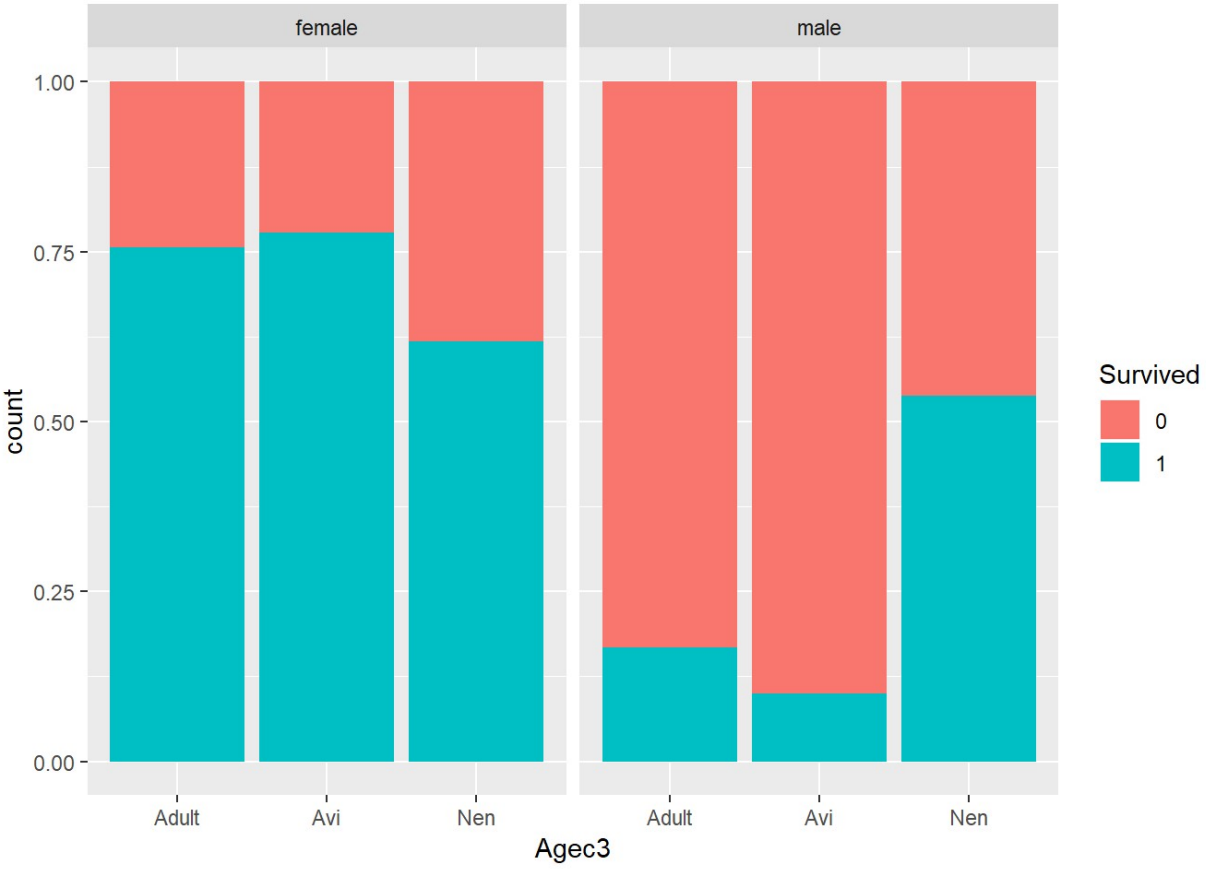
```
table(train$Survived)
```

```
##
##   0   1
## 549 342
```

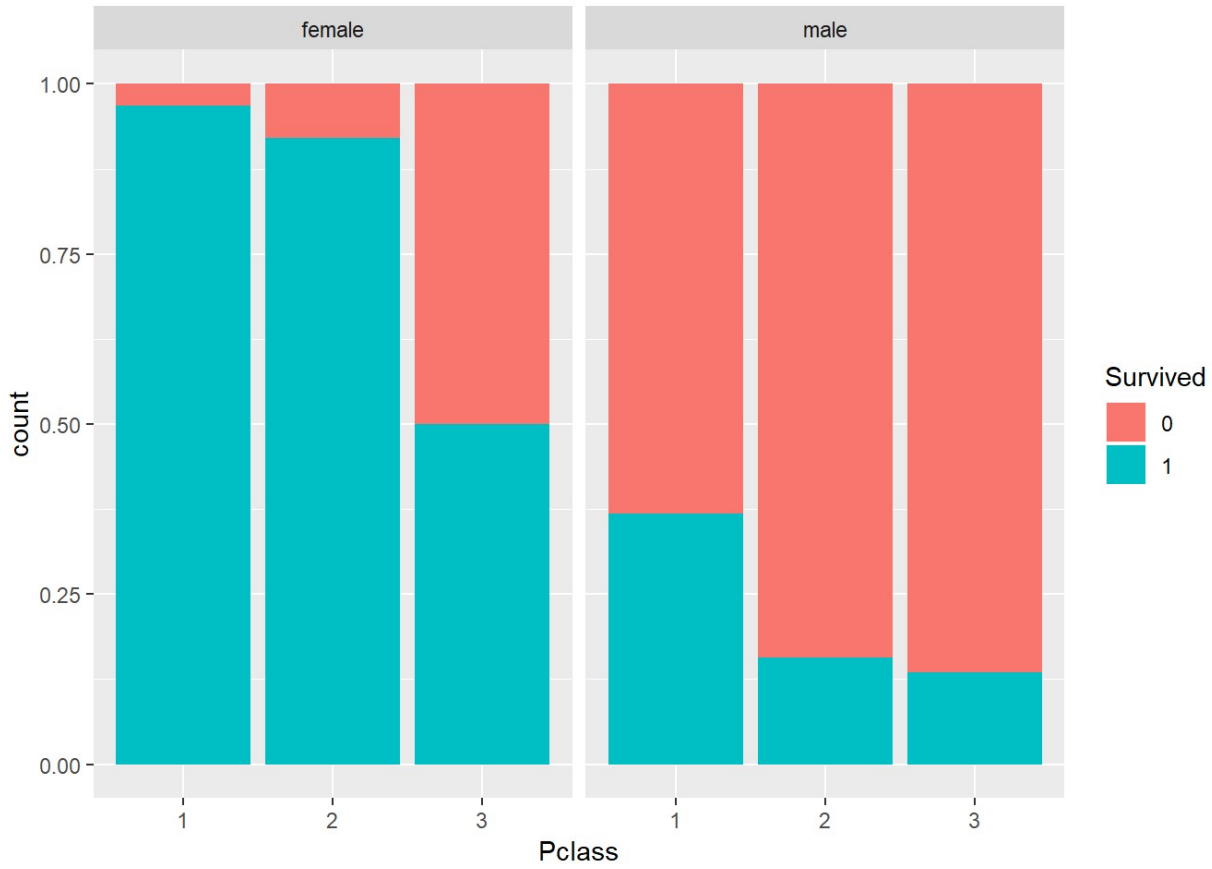
En train, sabem que tenim 891 registres, dels quals 549 no sobreviuen i 342 sobreviuen.

D'aquests, sabem que si són dona o nen de primera o segona classe, tenen més probabilitat de sobreviure que si són home i adult o avi de segona o tercera classe:

```
ggplot(data = train,aes(x=Agec3,fill=Survived))+geom_bar(position="fill")+facet_wrap(~Sex)
```

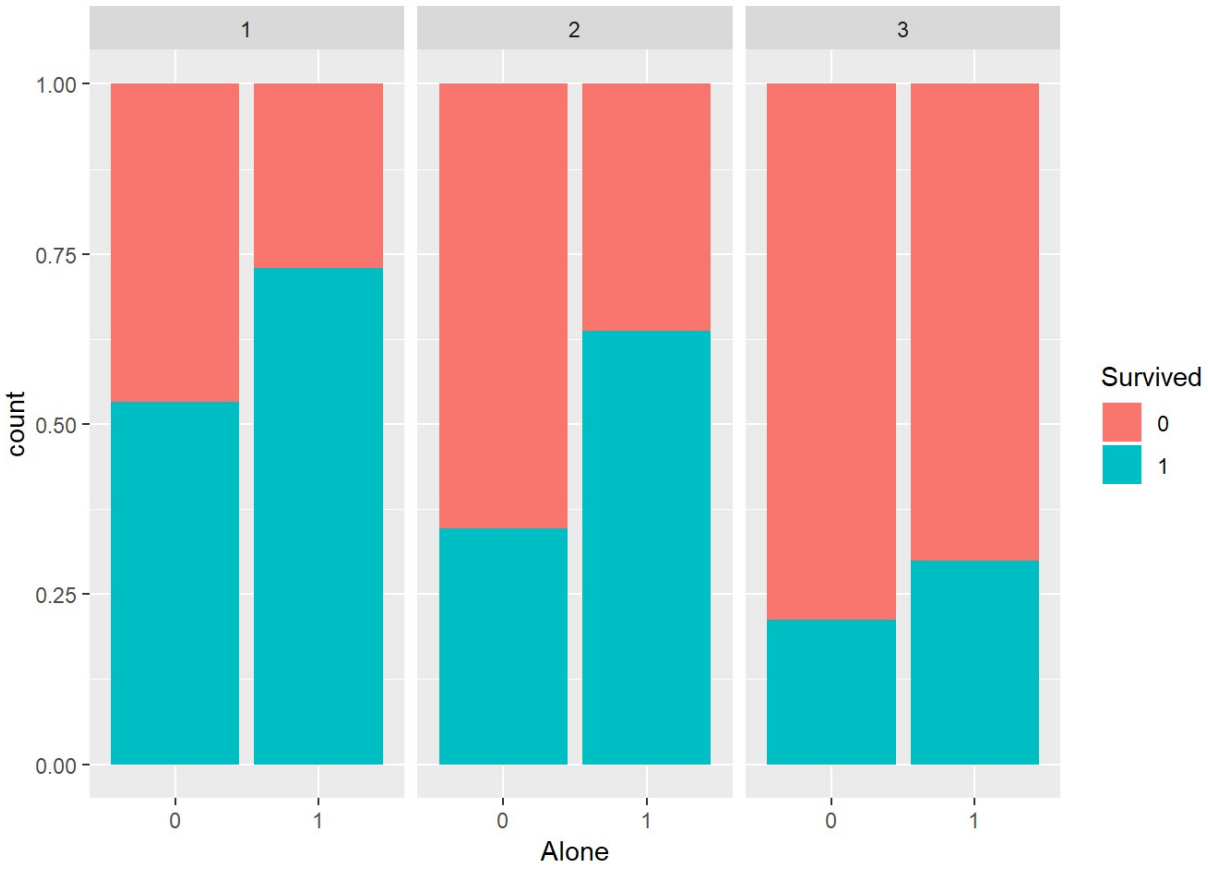



```
ggplot(data = train,aes(x=Pclass,fill=Survived))+geom_bar(position="fill")+facet_wrap(~Sex)
```



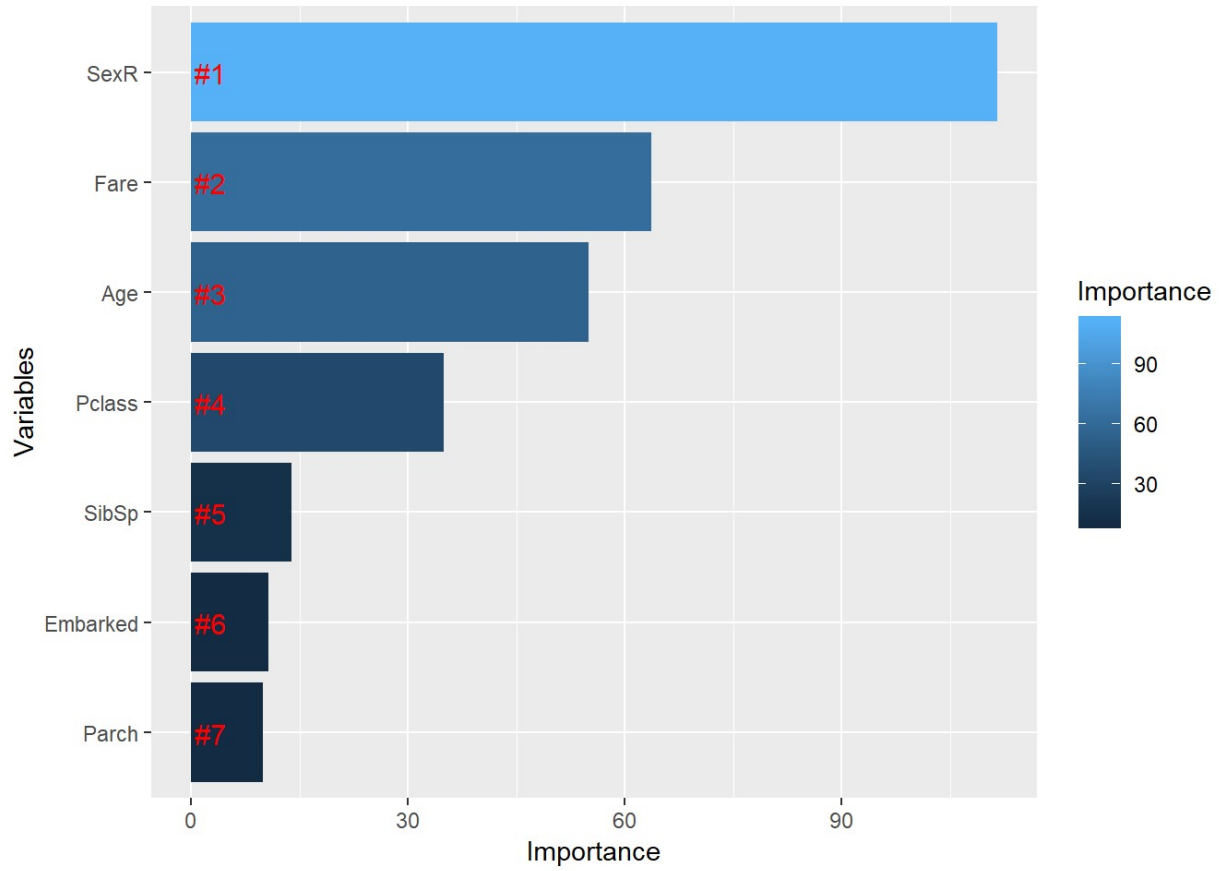

També sabem que els que anaven acompanyats tenen més supervivència que els que anaven sols:

```
ggplot(data = train,aes(x=Alone,fill=Survived))+geom_bar(position="fill")+facet_wrap
(~Pclass)
```




També sabem que les variables que tenen més importància en el model són: (utilitzem l'arbre creat amb el model rf1a per veure les importàncies de les variables) SexR, Fare, Age i Pclass.

```
# Obtenir importancia
importance    <- importance(rf1a)
varImportance <- data.frame(Variables = row.names(importance), Importance = round(im
portance[ ,'MeanDecreaseGini'],2))

# Crear un rang basat en l'importància
rankImportance <- varImportance %>%
  mutate(Rank = paste0('#',dense_rank(desc(Importance))))

# Visaulitzem importància relativa de les variables
ggplot(rankImportance, aes(x = reorder(Variables, Importance),
    y = Importance, fill = Importance)) +
  geom_bar(stat='identity') +
  geom_text(aes(x = Variables, y = 0.5, label = Rank),
    hjust=0, vjust=0.55, size = 4, colour = 'red') +
  labs(x = 'Variables') +
  coord_flip()
```




`

# 6. Resolució del problema. A partir dels resultats obtinguts, quines són les conclusions? Els resultats permeten respondre al problema?

Amb l'estudi de les dades del titànic el que volíem saber era la probabilitat de supervivència segons el grup al que pertanyia al passatger i en particular volíem predir quins passatgers van sobreviure.

En primer lloc hem fet una neteja de les dades i hem creat algunes variables, alhora que analitzavem els valors que agafaven aquestes dades. Llavors hem fet un anàlisi visual per saber , prèvi a crear el model, quines eren les característiques que feien que es sobrevisqués i quines relacions hi ha entre les dades.

De seguida hem vist que les dones tenen més supervivència que els homes, que els embarcats a C tenien més supervivència que els embarcats a Q i S, que els de primera classe tenien més supervivència que els de segona classe i aquests, més supervivència que els de tercera classe.

També hem vist que els que tenen SibSp entre 0 i 4 tenien una freqüència de supervivència del 20 al 50% i que els que tenien més SibSp no sobrevivien. La variable Parch també és molt igual: Tenen una freqüència de supervivència del 20 al 60% els que tenen Parch de 0 a 3 o 5, sinó no sobreviuen.

En quant a l'edat, hem vist que els nens tenen més probabilitat de sobreviure que els adults.

Llavors hem fet tres contrasts d'hipòtesis:

- Hi ha diferència entre la supervivència dels homes i de les dones?
- Hi ha diferència entre la supervivència dels nens i dels adults?
- Hi ha diferència entre la supervivència dels que anaven sols i dels que anaven acompanyats?

I la resposta en tots és que si, que sí que hi ha diferència entre els grups. Les dones tenen més supervivència que els homes, els nens tenen més supervivència que els adults i els que anaven acompanyats tenen més supervivència que els que anaven sols.

Llavors hem continuat amb una regressió, on hem predit els valors de test a partir del següent model (model amb millor precisió de test):

Survived ~ Pclass + SexR + Age + Embarked + Alone

On hem vist que les variables que més influeixen són Pclass i Sex (ja que són les que tenen més pendent) i també hem vist que tens més probabilitat de sobreviure si ets de primera classe, dona, ets jove, et vas embarcar a C i no anaves sol i tens menys probabilitatss de sobreviure si ets de tercera classe, home, amb molta edat, embarcat a S i vas sol.

En els models de regressió creats, hem vist que tot i tenir algunes AIC altes, el model que prediu millor test no és el que té l'AIC més alt.

Finalment, hem creat un arbre per classificar on hem vist que les variables més importants que utilitza són SexR, Fare, Age i Pclass.

Hem provat varis models de regressió i arbres sense millorar la precisió de test de 77.99%, de manera que crec que sense tractar més les dades no crec que millorem més la precisió.

# 7. Codi: Cal adjuntar el codi, preferiblement en R, amb el que s'ha realitzat la neteja, anàlisi i representació de les dades. Si ho preferiu, també podeu treballar en Python.

Extraiem les dades en csv per penjar a kaggle i comprovar la precisió.

```
write.csv(test[,c("PassengerId", "pred_arbre")], file="C:/Users/Laura/Desktop/UOC/Tipologia i cicle de vida de les dades/PRAC 2 Neteja i validació/titanic/test_predit.csv")
```