PRAC 2: Titanic

Oriol Roqué Paniagua

0. Introducció

En aquesta pràctica realitzarem un estudi de modelització predictiva sobre les defuncions ocorregudes en l'enfonsament del Titanic. L'objectiu serà crear un model amb el conjunt de dades de train que ens permeti a posteriori determinar la supervivència o no dels passatgers del conjunt de test. Finalment, evaluarem la capacitat predictora del model realitzant una comparativa entre la predicció i la realitat.

Comencem per carregar les llibreries.

```
# libraries
library(readr)      # load csv data
library(ggplot2)    # graphical results
library(sqldf)      # dataframes as sql tables
library(GGally)     # visual correlation
library(ISLR)       # logisitic regression
library(DMwR)       # knn
library(neuralnet) # neural networks
library(pROC)       # ROC curve
```

1. Descripció del dataset

Les dades per realitzar aquest estudi provenen de Kaggle (https://www.kaggle.com/c/titanic/data). Aquestes estan distribuides en 3 fitxers, que es corresponen a:

1. train: conté 891 registres amb les variables predictores i la variable objectiu. Servirà per entrenar els models.
2. test: conté 418 registres amb les variables predictores. Servirà per generar prediccions dels models.
3. gender_submission: conté els mateixos 418 registres que test però únicament amb la variable objectiu. Es correspon a un exemple de com s'han de fer les càrregues per entrar a la competició activa de Kaggle.

Cal comentar que, degut al caràcter de competició de Kaggle, el fitxer de test que proporcionen no té la variable objectiu. Per aquest motiu, quan analitzem les dades i realitzem els models, anomenarem a train a un subconjunt del fitxer de train, test al subconjunt restant del fitxer de train i utilitzarem el fitxer de test únicament per generar les prediccions per la competició. No obstant, de cares a realitzar la neteja de les dades, utilitzarem ambdós fitxers.

Carreguem els fitxers i realitzem un anàlisi previ de les dades.

```
# load train data with objective
df_train <- read.csv("train.csv", sep = ",", stringsAsFactors = TRUE)

# load test data without objective
df_test <- read.csv("test.csv", sep = ",", stringsAsFactors = TRUE)

# load test objective data
df_gender_submission <- read.csv("gender_submission.csv", sep = ",",
                                 stringsAsFactors = TRUE)

# summary
summary(df_train)
```

```
##   PassengerId       Survived          Pclass
```

```
##  Min.   :  1.0   Min.   :0.0000   Min.   :1.000
##  1st Qu.:223.5   1st Qu.:0.0000   1st Qu.:2.000
##  Median :446.0   Median :0.0000   Median :3.000
##  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
##
##                                     Name         Sex           Age
##  Abbing, Mr. Anthony                   :  1   female:314   Min.   : 0.42
##  Abbott, Mr. Rossmore Edward           :  1   male  :577   1st Qu.:20.12
##  Abbott, Mrs. Stanton (Rosa Hunt)      :  1                Median :28.00
##  Abelson, Mr. Samuel                   :  1                Mean   :29.70
##  Abelson, Mrs. Samuel (Hannah Wizosky):  1                3rd Qu.:38.00
##  Adahl, Mr. Mauritz Nils Martin        :  1                Max.   :80.00
##  (Other)                               :885                NA's   :177
##      SibSp           Parch             Ticket         Fare
##  Min.   :0.000   Min.   :0.0000   1601    :  7   Min.   :  0.00
##  1st Qu.:0.000   1st Qu.:0.0000   347082  :  7   1st Qu.:  7.91
##  Median :0.000   Median :0.0000   CA. 2343:  7   Median : 14.45
##  Mean   :0.523   Mean   :0.3816   3101295 :  6   Mean   : 32.20
##  3rd Qu.:1.000   3rd Qu.:0.0000   347088  :  6   3rd Qu.: 31.00
##  Max.   :8.000   Max.   :6.0000   CA 2144 :  6   Max.   :512.33
##                                   (Other) :852
##          Cabin     Embarked
##             :687    :  2
##  B96 B98    :  4   C:168
##  C23 C25 C27:  4   Q: 77
##  G6         :  4   S:644
##  C22 C26    :  3
##  D          :  3
##  (Other)    :186
```

```
str(df_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       : Factor w/ 891 levels "Abbing, Mr. Anthony",..: 109 191 358 277 16 559 520 629 417 581
##  $ Sex        : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 2 1 1 ...
##  $ 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     : Factor w/ 681 levels "110152","110413",..: 524 597 670 50 473 276 86 396 345 133 ...
##  $ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Cabin      : Factor w/ 148 levels "","A10","A14",..: 1 83 1 57 1 1 131 1 1 1 ...
##  $ Embarked   : Factor w/ 4 levels "","C","Q","S": 4 2 4 4 4 3 4 4 4 2 ...
```

```
str(df_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       : Factor w/ 418 levels "Abbott, Master. Eugene Joseph",..: 210 409 273 414 182 370 85 5
##  $ Sex        : Factor w/ 2 levels "female","male": 2 1 2 2 1 2 1 2 1 2 ...
```

```
##  $ 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     : Factor w/ 363 levels "110469","110489",..: 153 222 74 148 139 262 159 85 101 270 ...
##  $ Fare       : num  7.83 7 9.69 8.66 12.29 ...
##  $ Cabin      : Factor w/ 77 levels "","A11","A18",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Embarked   : Factor w/ 3 levels "C","Q","S": 2 3 2 3 3 3 2 3 1 3 ...
```

Veiem que el dataset consta de 12 atributs, que són:

1. PassengerId: Identificador únic del passatger
2. Survived: Variable objectiu que determina si el passatger va sobreviure (1) o no (0)
3. Pclass: Classe del ticket
4. Name: Nom del passatger
5. Sex: Sexe
6. Age: Edat
7. SibSp: Nombre de germans o cònjugues a bord del Titànic
8. Parch: Nombre de pares o fills a bord del Titànic
9. Ticket: Tipus de ticket
10. Fare: Tarifa del ticket
11. Cabin: Nombre de cabina
12. Embarked: On va embarcar el passatger

2. Integració i selecció de dades

En aquest apartat seleccionarem les variables que ens interessen per l'estudi i en crearem de noves. Abans però, integrarem els 3 fitxers en un de sol, de manera que els canvis que generem en aquest apartat i en la neteja de dades siguin comuns tant al conjunt de test com al de train. Mantindrem la distinció de la pertanença de cada registre al conjunt original mitjançant la variable Type, que contindrà el valor TEST o TRAIN. Quan haguem de fer l'anàlisi, tornarem a separar els conjunts de dades. Aquí gender_submission únicament actuarà com a columna 'falsa' de manera que els valors que contindrà Survived pel cas de Test no es corresponen amb la realitat.

Per a fer la manipulació de les dades, treballarem amb la llibreria sqldf, que permet tractar dataframes com a taules d'un sistema gestor de bases de dades relacional, de manera que es treballa directament amb SQL.

```
# build unique dataset, with a new variable to determine if the row is used for test
# or for train purpose
df_test_complete <- sqldf("SELECT
                            'TEST' as Type,
                            t.PassengerId,
                            o.Survived,
                            t.PClass,
                            t.Name,
                            t.Sex,
                            t.Age,
                            t.SibSp,
                            t.Parch,
                            t.Ticket,
                            t.Fare,
                            t.Cabin,
                            t.Embarked
                          FROM df_test t INNER JOIN df_gender_submission o
                          ON t.PassengerId = o.PassengerId")

df_train_complete <- sqldf("SELECT
```

```
                              'TRAIN' as Type,
                              t.*
                             FROM df_train as t")

df_titanic <- sqldf("SELECT * FROM df_train_complete
                     UNION
                    SELECT * FROM df_test_complete")

# summary
summary(df_titanic)
```

```
##      Type            PassengerId      Survived          Pclass
##  Length:1309        Min.   :   1   Min.   :0.0000   Min.   :1.000
##  Class :character   1st Qu.: 328   1st Qu.:0.0000   1st Qu.:2.000
##  Mode  :character   Median : 655   Median :0.0000   Median :3.000
##                     Mean   : 655   Mean   :0.3774   Mean   :2.295
##                     3rd Qu.: 982   3rd Qu.:1.0000   3rd Qu.:3.000
##                     Max.   :1309   Max.   :1.0000   Max.   :3.000
##
##      Name               Sex           Age            SibSp
##  Length:1309        female:466   Min.   : 0.17   Min.   :0.0000
##  Class :character   male  :843   1st Qu.:21.00   1st Qu.:0.0000
##  Mode  :character                Median :28.00   Median :0.0000
##                                  Mean   :29.88   Mean   :0.4989
##                                  3rd Qu.:39.00   3rd Qu.:1.0000
##                                  Max.   :80.00   Max.   :8.0000
##                                  NA's   :263
##      Parch          Ticket               Fare            Cabin
##  Min.   :0.000   Length:1309        Min.   :  0.000   Length:1309
##  1st Qu.:0.000   Class :character   1st Qu.:  7.896   Class :character
##  Median :0.000   Mode  :character   Median : 14.454   Mode  :character
##  Mean   :0.385                      Mean   : 33.295
##  3rd Qu.:0.000                      3rd Qu.: 31.275
##  Max.   :9.000                      Max.   :512.329
##                                     NA's   :1
##    Embarked
##  Length:1309
##  Class :character
##  Mode  :character
##
##
##
##
```

```
str(df_titanic)
```

```
## 'data.frame':    1309 obs. of  13 variables:
##  $ Type       : chr  "TEST" "TEST" "TEST" "TEST" ...
##  $ PassengerId: int  892 893 894 895 896 897 898 899 900 901 ...
##  $ Survived   : int  0 1 0 0 1 0 1 0 1 0 ...
##  $ Pclass     : int  3 3 2 3 3 3 3 2 3 3 ...
##  $ Name       : chr  "Kelly, Mr. James" "Wilkes, Mrs. James (Ellen Needs)" "Myles, Mr. Thomas Francis
##  $ Sex        : Factor w/ 2 levels "female","male": 2 1 2 2 1 2 1 2 1 2 ...
##  $ 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 tenim valors nuls en Age, Fare i Embarked, éssent el valor S el que es repeteix més per a aquesta última variable.

Eliminem l'atribut Name de l'anàlisi, doncs no és rellevant de cara a la determinació de la supervivència. Pel cas que volem estudiar, simplificarem el joc de dades treient també Ticket i Cabin. Creem l'atribut FamSize que indica la grandària de la família a partir dels atributs SibSp i Parch, eliminant-los del futur joc de dades. Finalment, aprofitem per codificar Sex com a variable numèrica dicotòmica, éssent 'male' igual a 1 i 'female' igual a 0.

```
# select only interesting variables
df_titanic <- sqldf("SELECT
                      t.PassengerId,
                      t.Type,
                      t.Survived,
                      t.PClass,
                      CASE WHEN t.Sex = 'male' THEN 1 ELSE 0 END AS Sex,
                      t.Age,
                      t.SibSp + t.Parch + 1 as FamSize,
                      t.Fare,
                      t.Embarked
                    FROM df_titanic t")

# summary
summary(df_titanic)
```

```
##   PassengerId       Type              Survived          Pclass
##  Min.   :   1   Length:1309        Min.   :0.0000   Min.   :1.000
##  1st Qu.: 328   Class :character   1st Qu.:0.0000   1st Qu.:2.000
##  Median : 655   Mode  :character   Median :0.0000   Median :3.000
##  Mean   : 655                      Mean   :0.3774   Mean   :2.295
##  3rd Qu.: 982                      3rd Qu.:1.0000   3rd Qu.:3.000
##  Max.   :1309                      Max.   :1.0000   Max.   :3.000
##
##       Sex              Age           FamSize           Fare
##  Min.   :0.000   Min.   : 0.17   Min.   : 1.000   Min.   :  0.000
##  1st Qu.:0.000   1st Qu.:21.00   1st Qu.: 1.000   1st Qu.:  7.896
##  Median :1.000   Median :28.00   Median : 1.000   Median : 14.454
##  Mean   :0.644   Mean   :29.88   Mean   : 1.884   Mean   : 33.295
##  3rd Qu.:1.000   3rd Qu.:39.00   3rd Qu.: 2.000   3rd Qu.: 31.275
##  Max.   :1.000   Max.   :80.00   Max.   :11.000   Max.   :512.329
##                  NA's   :263                      NA's   :1
##    Embarked
##  Length:1309
##  Class :character
##  Mode  :character
##
##
##
##
```

```
str(df_titanic)
```

```
## 'data.frame':    1309 obs. of  9 variables:
##  $ PassengerId: int  892 893 894 895 896 897 898 899 900 901 ...
##  $ Type       : chr  "TEST" "TEST" "TEST" "TEST" ...
##  $ Survived   : int  0 1 0 0 1 0 1 0 1 0 ...
##  $ Pclass     : int  3 3 2 3 3 3 3 2 3 3 ...
##  $ Sex        : int  1 0 1 1 0 1 0 1 0 1 ...
##  $ Age        : num  34.5 47 62 27 22 14 30 26 18 21 ...
##  $ FamSize    : int  1 2 1 1 3 1 1 3 1 3 ...
##  $ Fare       : num  7.83 7 9.69 8.66 12.29 ...
##  $ Embarked   : chr  "Q" "S" "Q" "S" ...
```

Amb el dataset preparat, procedim a realitzar la neteja de les dades.

3. Neteja de les dades

A continuació realitzem una neteja de les dades tenint en compte possibles valors nuls i valors extrems. Depenent del resultat que n'obtenim, és possible que haguem de modificar el dataset.

3.1. Elements buits

Hem vist en les inspeccions del dataset de l'apartat anterior que els valors nuls o buits corresponien a Embarked, Fare i Age.

Realitzem una imputació "manual" als passatgers que no tenen informat el valor Embarked a S, doncs és el valor que més es repeteix. Aprofitem per descomposar l'atribut categòric en 3 atributs numèrics per a cada un dels valors que prèn la variable. Això ens permetrà utilitzar aquesta variable en els models posteriors.

```
# NA/empty string variables
df_titanic$Embarked[df_titanic$Embarked==""] = 'S'

# distribute Embarked
df_titanic <- sqldf("SELECT
                      t.PassengerId,
                      t.Type,
                      t.Survived,
                      t.PClass,
                      t.Sex,
                      t.Age,
                      t.FamSize,
                      t.Fare,
                      CASE WHEN t.Embarked = 'S' THEN 1 ELSE 0 END AS Embarked_S,
                      CASE WHEN t.Embarked = 'C' THEN 1 ELSE 0 END AS Embarked_C,
                      CASE WHEN t.Embarked = 'Q' THEN 1 ELSE 0 END AS Embarked_Q
                    FROM df_titanic t")
```

Realitzem una imputació de l'atribut Fare mitjançant la mediana dels valors. Utilitzem la mediana i no la mitjana ja que aquestes disten molt una de l'altra, tal com es pot veure en els summary dels datasets anteriors, i pot indicar la presència d'algun valor extrem, que analitzarem a posteriori.

```
df_titanic$Fare[is.na(df_titanic$Fare)==TRUE] = median(df_titanic$Fare, na.rm = TRUE)
```

Finalment, realitzem una imputació de l'atribut Age per aquells passatgers que ho tenen en nul mitjançant el mètode dels 3 veïns més propers. Per fer-ho, utilitzarem la funció knnImputation, reescalant els valors de les variables i utilitzant com a mètode la mediana.

```
knnOutput <- knnImputation(df_titanic[,
                c('Age','Pclass','FamSize','Fare','Sex',
                  'Embarked_S','Embarked_C','Embarked_Q')],
                k = 3, scale = T, meth = "median")
df_titanic$Age <- knnOutput$Age
```

Mostrem un resum de les dades per assegurar-nos que no queden valors nuls.

```
# summary
summary(df_titanic)
```

```
##   PassengerId       Type              Survived          Pclass
##  Min.   :   1   Length:1309        Min.   :0.0000   Min.   :1.000
##  1st Qu.: 328   Class :character   1st Qu.:0.0000   1st Qu.:2.000
##  Median : 655   Mode  :character   Median :0.0000   Median :3.000
##  Mean   : 655                      Mean   :0.3774   Mean   :2.295
##  3rd Qu.: 982                      3rd Qu.:1.0000   3rd Qu.:3.000
##  Max.   :1309                      Max.   :1.0000   Max.   :3.000
##       Sex             Age           FamSize           Fare
##  Min.   :0.000   Min.   : 0.17   Min.   : 1.000   Min.   :  0.000
##  1st Qu.:0.000   1st Qu.:21.00   1st Qu.: 1.000   1st Qu.:  7.896
##  Median :1.000   Median :28.00   Median : 1.000   Median : 14.454
##  Mean   :0.644   Mean   :29.83   Mean   : 1.884   Mean   : 33.281
##  3rd Qu.:1.000   3rd Qu.:38.00   3rd Qu.: 2.000   3rd Qu.: 31.275
##  Max.   :1.000   Max.   :80.00   Max.   :11.000   Max.   :512.329
##    Embarked_S       Embarked_C       Embarked_Q
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.00000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.00000
##  Median :1.0000   Median :0.0000   Median :0.00000
##  Mean   :0.6998   Mean   :0.2063   Mean   :0.09396
##  3rd Qu.:1.0000   3rd Qu.:0.0000   3rd Qu.:0.00000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.00000
```

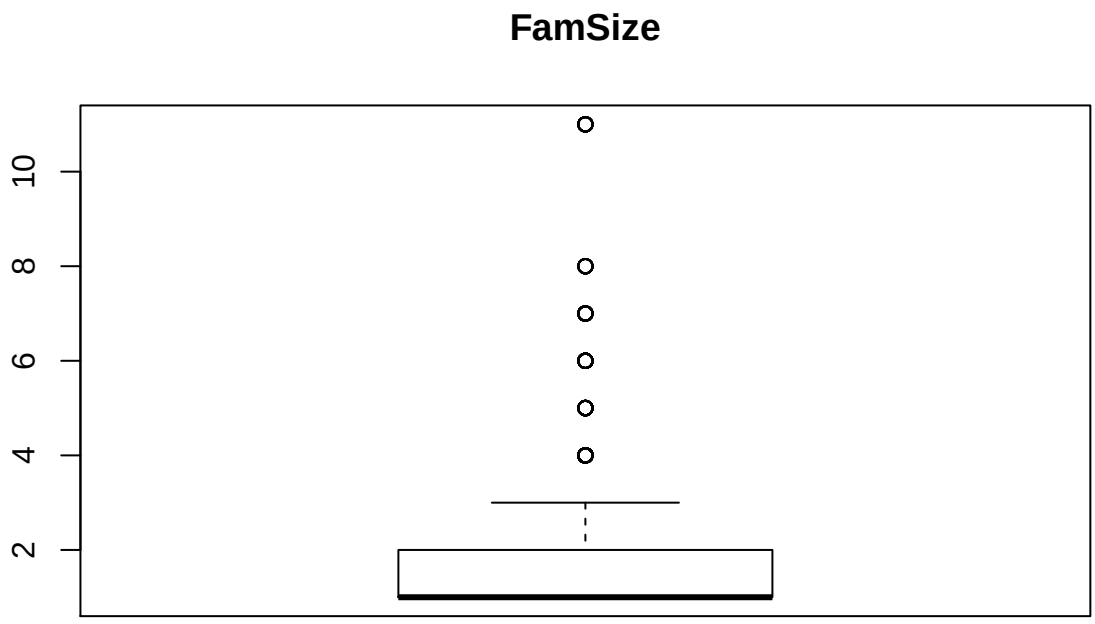
```
str(df_titanic)
```

```
## 'data.frame':    1309 obs. of  11 variables:
##  $ PassengerId: int  892 893 894 895 896 897 898 899 900 901 ...
##  $ Type       : chr  "TEST" "TEST" "TEST" "TEST" ...
##  $ Survived   : int  0 1 0 0 1 0 1 0 1 0 ...
##  $ Pclass     : int  3 3 2 3 3 3 3 2 3 3 ...
##  $ Sex        : int  1 0 1 1 0 1 0 1 0 1 ...
##  $ Age        : num  34.5 47 62 27 22 14 30 26 18 21 ...
##  $ FamSize    : int  1 2 1 1 3 1 1 3 1 3 ...
##  $ Fare       : num  7.83 7 9.69 8.66 12.29 ...
##  $ Embarked_S : int  0 1 0 1 1 1 0 1 0 1 ...
##  $ Embarked_C : int  0 0 0 0 0 0 0 0 1 0 ...
##  $ Embarked_Q : int  1 0 1 0 0 0 1 0 0 0 ...
```

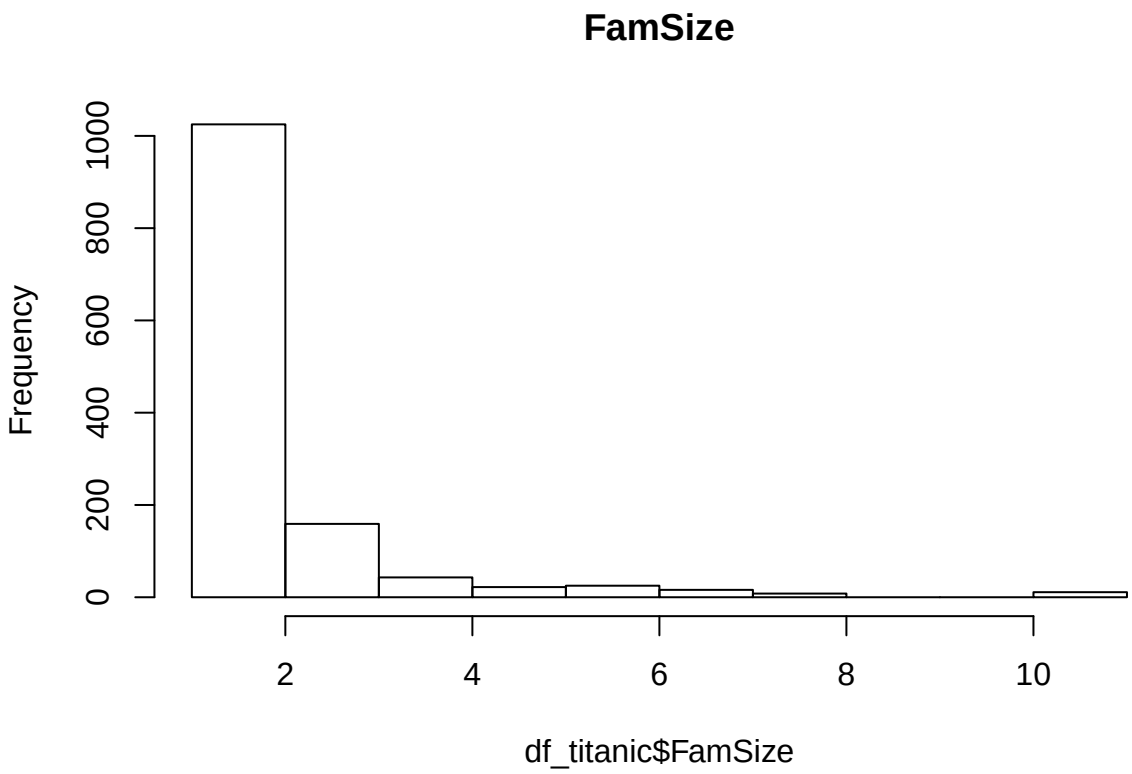
3.2. Valors extrems

Analitzem els valors extrems en els atributs FamSize, Fare i Age, doncs la resta de variables són dicotòmiques (a excepció de Pclass, que té 3 valors possibles). Ho farem de manera visual mitjançant diagrames de caixa i histogrames.

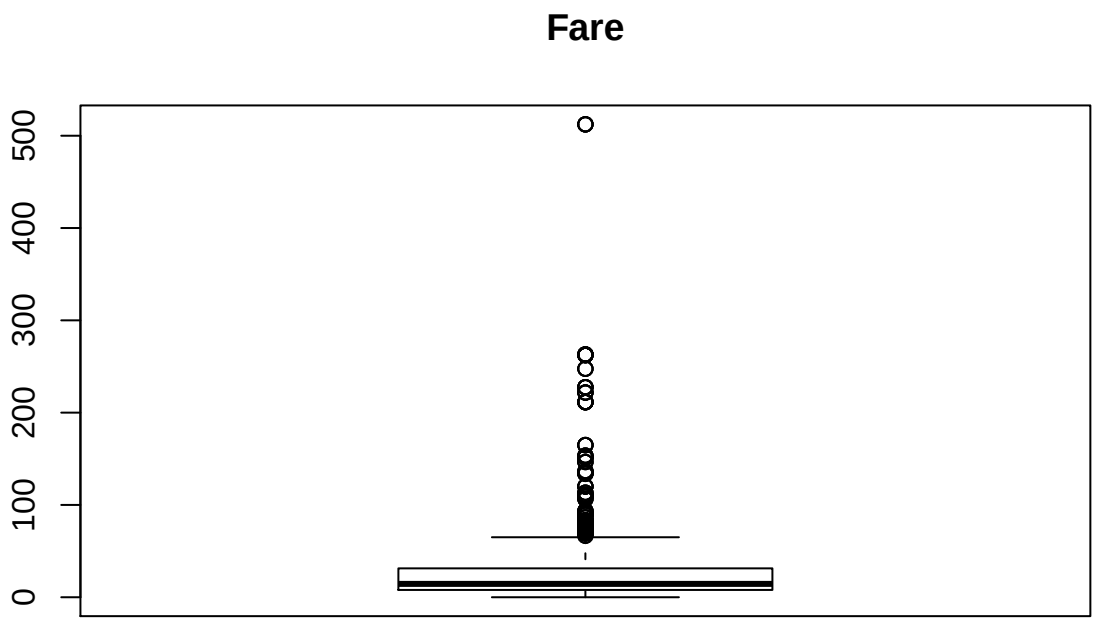
```
# FamSize
boxplot(df_titanic[,'FamSize'], main='FamSize')
```



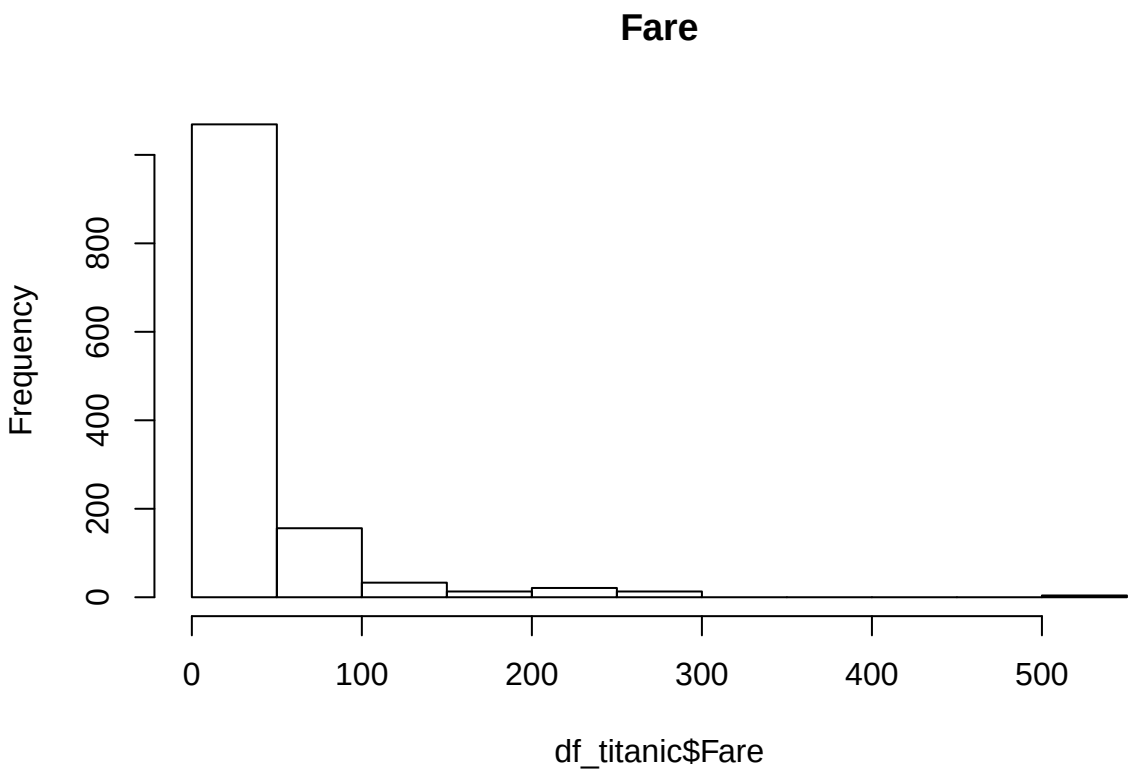

```
hist(df_titanic$FamSize, main='FamSize')
```



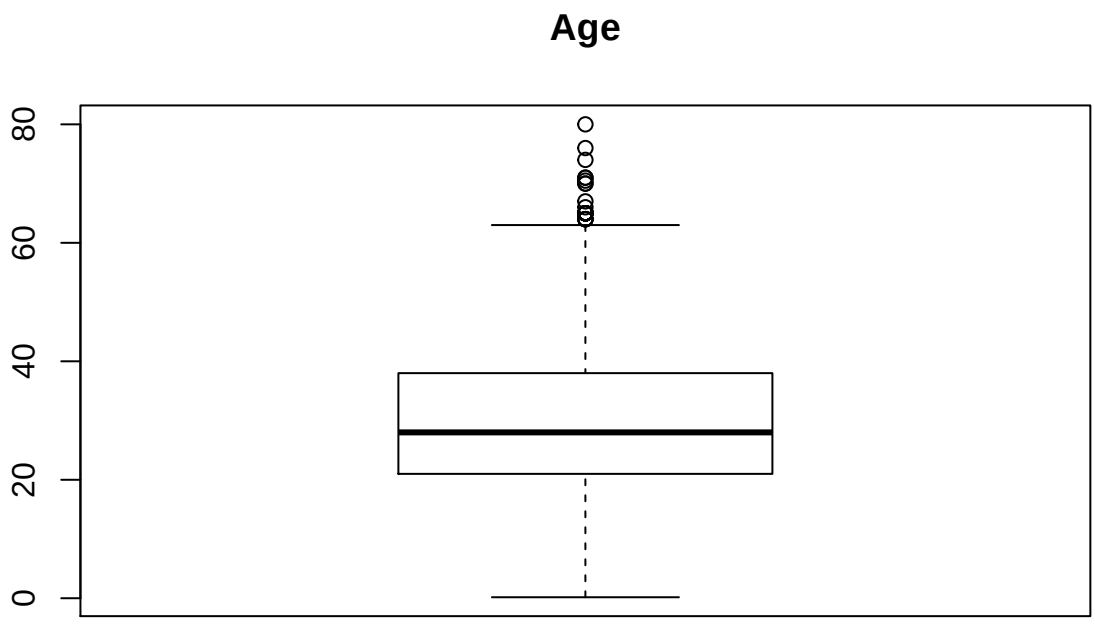

```
# Fare
boxplot(df_titanic[,'Fare'], main='Fare')
```



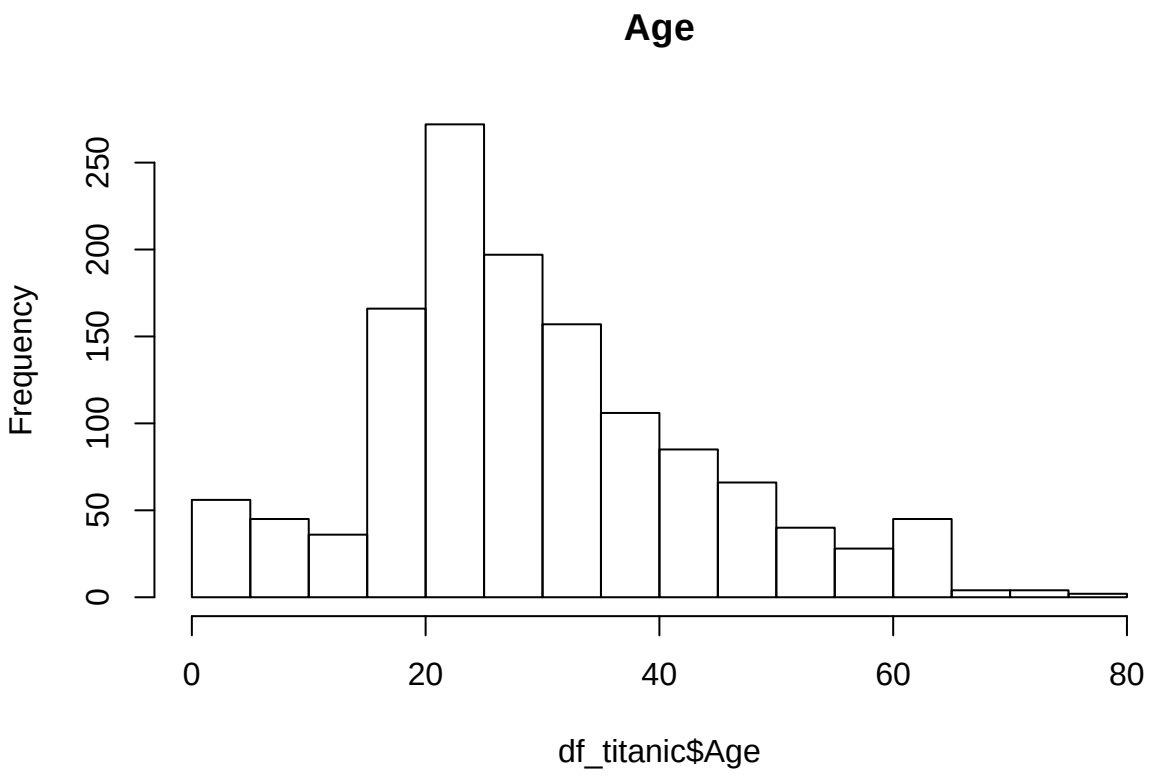

```
hist(df_titanic$Fare, main='Fare')
```




```
# Age
boxplot(df_titanic[,'Age'], main='Age')
```




```
hist(df_titanic$Age, main='Age')
```



Per a FamSize, veiem que la majoria de valors es troben en 1, és a dir, passatgers que viatjaven sols. La resta és més reduïda, tenint outliers que fins i tot arriben a 10 familiars. En aquest cas, optem per codificar l'atribut FamSize de nou amb dos únics valors: si viatjava sense família o acompanyat, independentment de la quantitat de membres de la família. El nou atribut es dirà Alone.

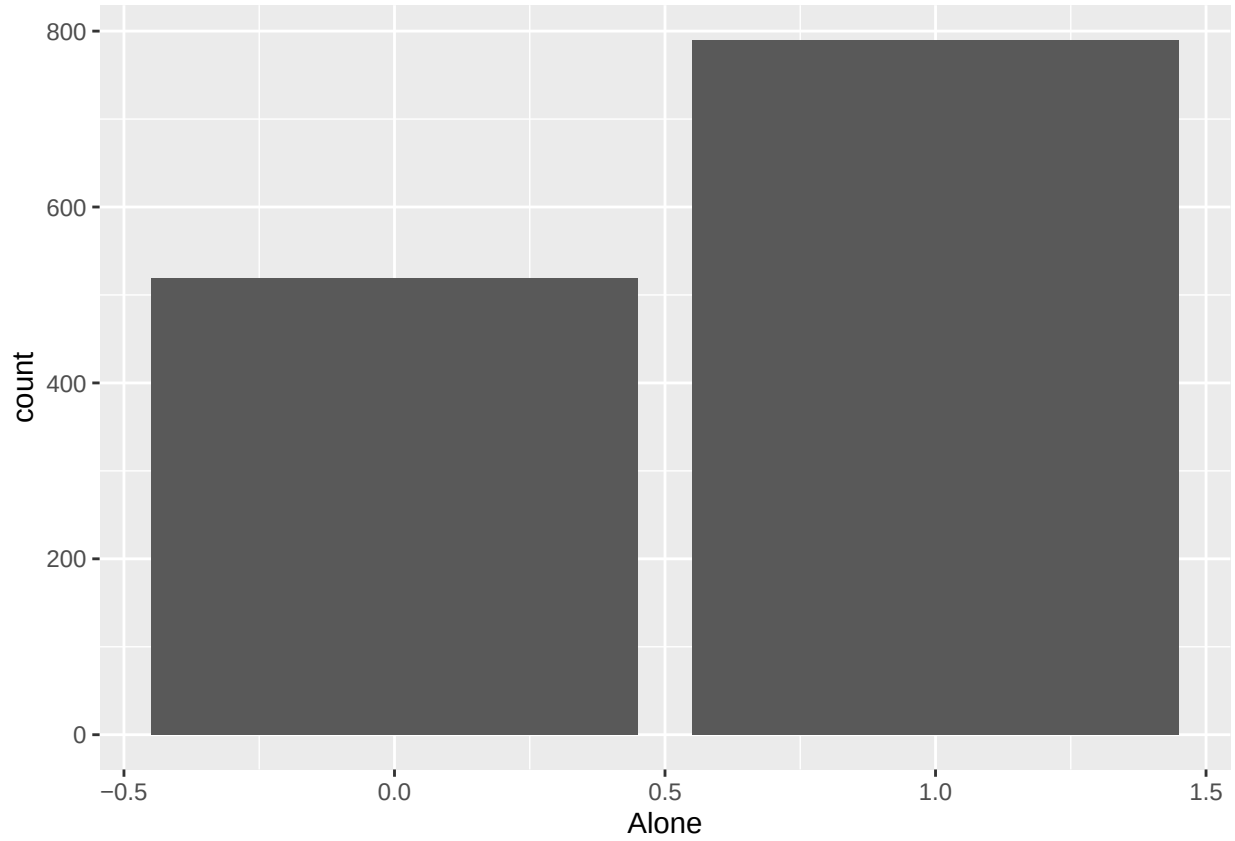
Pel que fa a Fare, l'histograma ens recorda a una caiguda exponencial del preu del bitllet. Per tal de normalitzar la variable, treballarem amb el logaritme natural de Fare com a nova variable predictiva.

Finalment, Age mostra una disitribució més centrada a l'entorn dels 25-30 anys. Aquí tractarem els outliers superiors substituint aquells valors superiors a 2.5 vegades el rang interquartílic de la distribució respecte la mediana, per suavitzar lleugerament la distribució. D'aquesta manera mantenim la possibilitat que puguin viatjar persones d'edat avançada però limitem l'efecte que ens pugui produir als models posteriors.

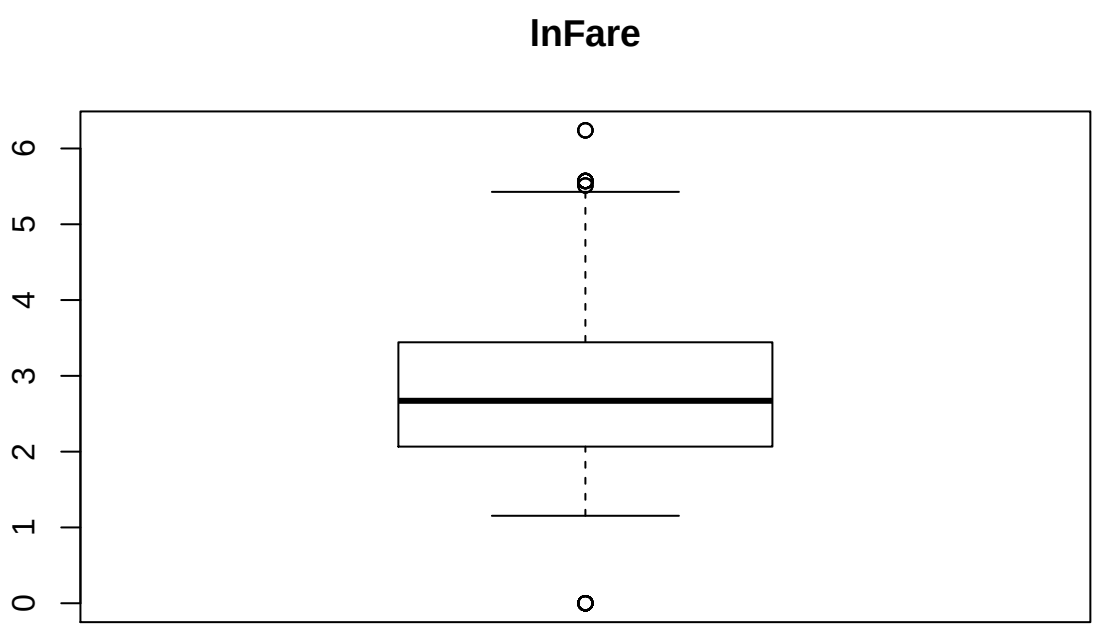
A continuació apliquem els canvis comentats:

```
# calculate superior age limit
age_sup_limit <- median(df_titanic$Age) + (quantile(df_titanic$Age,0.75) -
                 quantile(df_titanic$Age,0.25)) * 2.5

# create logarithmic variable with fare
df_titanic$lnFare <- log(df_titanic$Fare)

# reshape dataset
df_titanic <- sqldf(paste("SELECT
                      t.PassengerId,
                      t.Type,
                      t.Survived,
                      t.PClass,
```

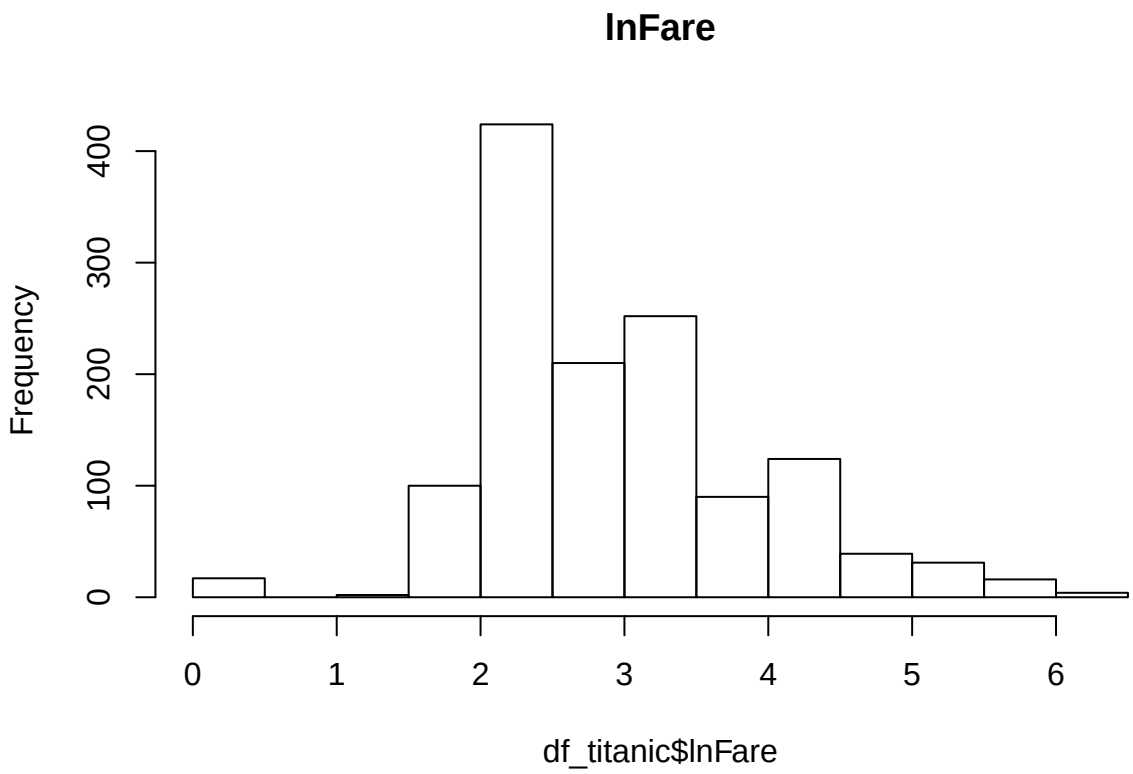
```
                    t.Sex,
                    CASE
                      WHEN t.Age >", age_sup_limit," THEN ", age_sup_limit, "
                      ELSE t.Age
                    END AS Age,
                    CASE
                      WHEN t.FamSize > 1 THEN 0
                      ELSE 1
                    END AS Alone,
                    CASE
                      WHEN lnFare IS NULL OR lnFare <0 THEN 0
                      ELSE lnFare
                    END AS lnFare,
                    t.Embarked_S,
                    t.Embarked_C,
                    t.Embarked_Q
                  FROM df_titanic t", sep=''))

# Alone disitribution
ggplot(data = df_titanic, aes(Alone)) + geom_bar()
```



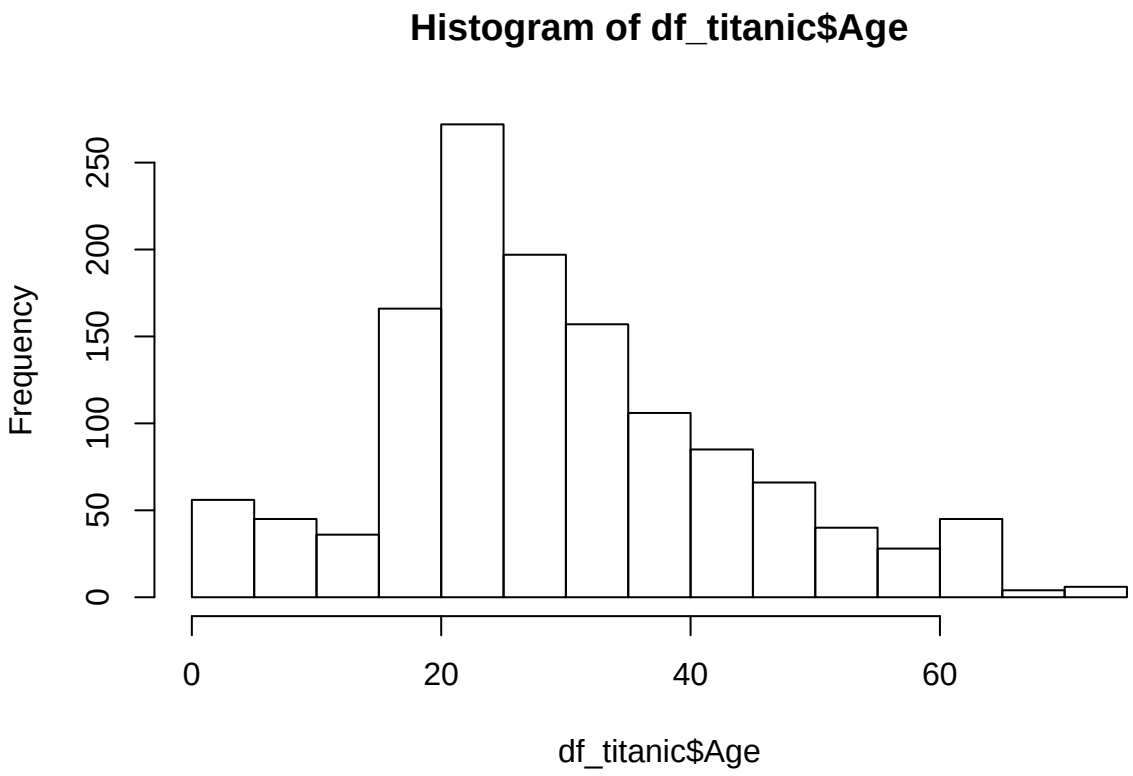

```
# new Fare distribution
boxplot(df_titanic[,'lnFare'], main='lnFare')
```




```
hist(df_titanic$lnFare, main='lnFare')
```




```
# new Age
hist(df_titanic$Age)
```



Amb les dades tractades, guardarem el fitxer complet per a usos posteriors.

```
# save new file
write.csv(df_titanic, "titanic_clean.csv")
```

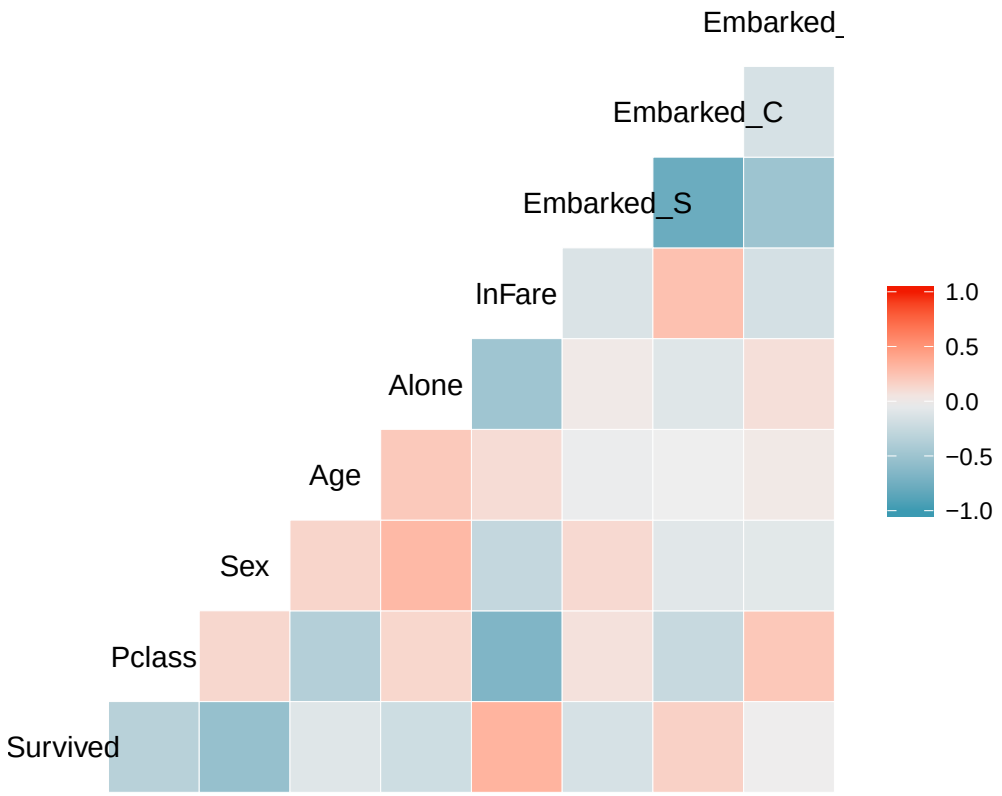
4. Anàlisi de les dades

En primer lloc, desfem els canvis per tornar a treballar únicament amb el fitxer de train. El dataset que utilitzarem s'anomenarà df_titanic_model, fent referència a què és el que utilitzarem per realitzar els models.

```
# restore train file
df_titanic_model <- subset(df_titanic, Type == 'TRAIN', select = -c(Type,PassengerId))
df_titanic_test <- subset(df_titanic, Type == 'TEST', select = -Type)

# clear data environment
rm(list=setdiff(ls(), c('df_titanic_model','df_titanic_test')))
```

A continuació, revisem les correlacions entre les variables del joc de dades visualment.

```
# show correlation of numeric variables visually
ggcorr(df_titanic_model)
```



Podem fer-nos una idea de com es correlacionen les diverses variables amb Survived, que correspon a la nostra variable objectiu. Per altra banda, també veiem correlacions interessants com són la classe i el cost del bitllet, que és negativa i bastant elevada, degut a què la primera classe té un preu més elevat que les altres classes. També podem veure com dels passatgers que viatjaven sols hi ha més predominància d'homes que no pas de dones.

4.1. Selecció dels grups de dades

Seleccionarem com a grups els següents casos:

1. Supervivència: 1 sobreviu i 0 no sobreviu.
2. Sexe: 1 male i 0 female.
3. Classe del bitllet: 1, 2 o 3, fent referència a 1ra, 2na i 3ra classe.
4. Si viatjava sense família: 1 viatja sense família, 0 viatja amb família.

```
# survived
titanic.survived <- df_titanic_model[df_titanic_model$Survived == 1,]
titanic.died <- df_titanic_model[df_titanic_model$Survived == 0,]

# sex
titanict.male <- df_titanic_model[df_titanic_model$Sex == 1,]
titanic.female <- df_titanic_model[df_titanic_model$Sex == 0,]

# class
titanic.first <- df_titanic_model[df_titanic_model$Pclass == 1,]
titanic.second <- df_titanic_model[df_titanic_model$Pclass == 2,]
titanic.third <- df_titanic_model[df_titanic_model$Pclass == 3,]
```

```r
# alone
titanic.alone <- df_titanic_model[df_titanic_model$Alone == 1,]
titanic.not_alone <- df_titanic_model[df_titanic_model$Alone == 0,]
```

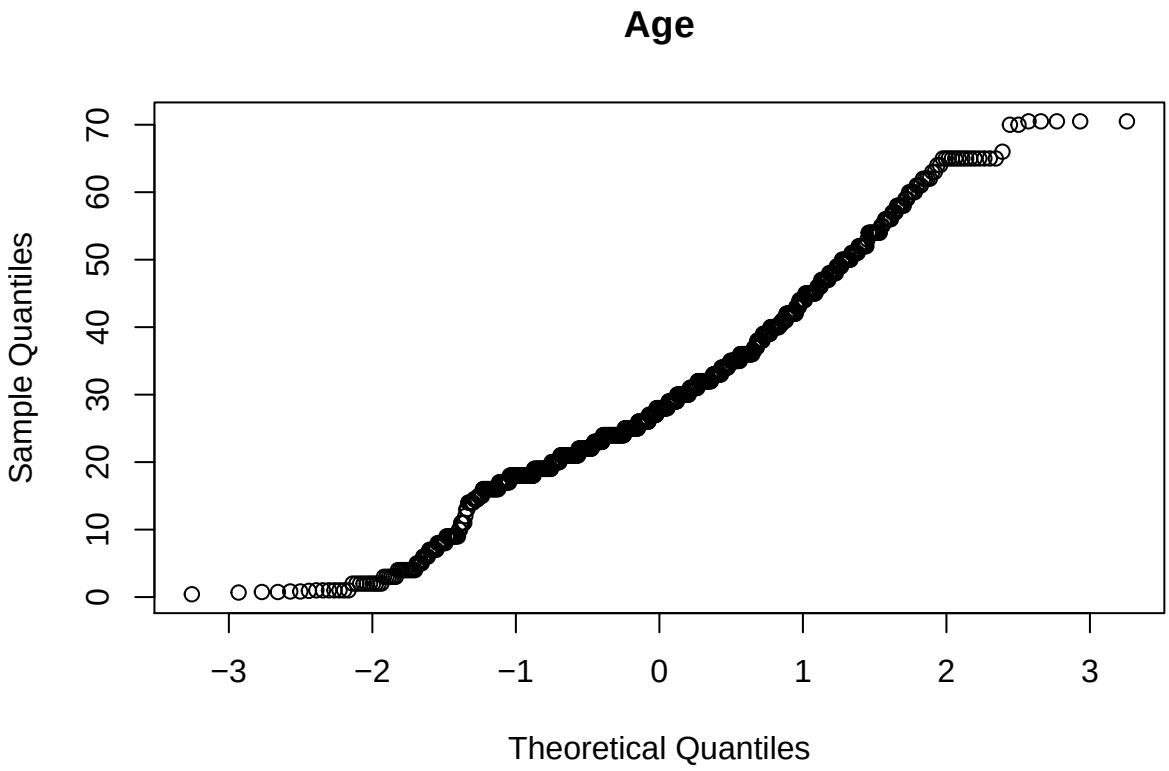
Utilitzarem alguns d'aquests grups per als anàlisis posteriors.

4.2. Normalitat

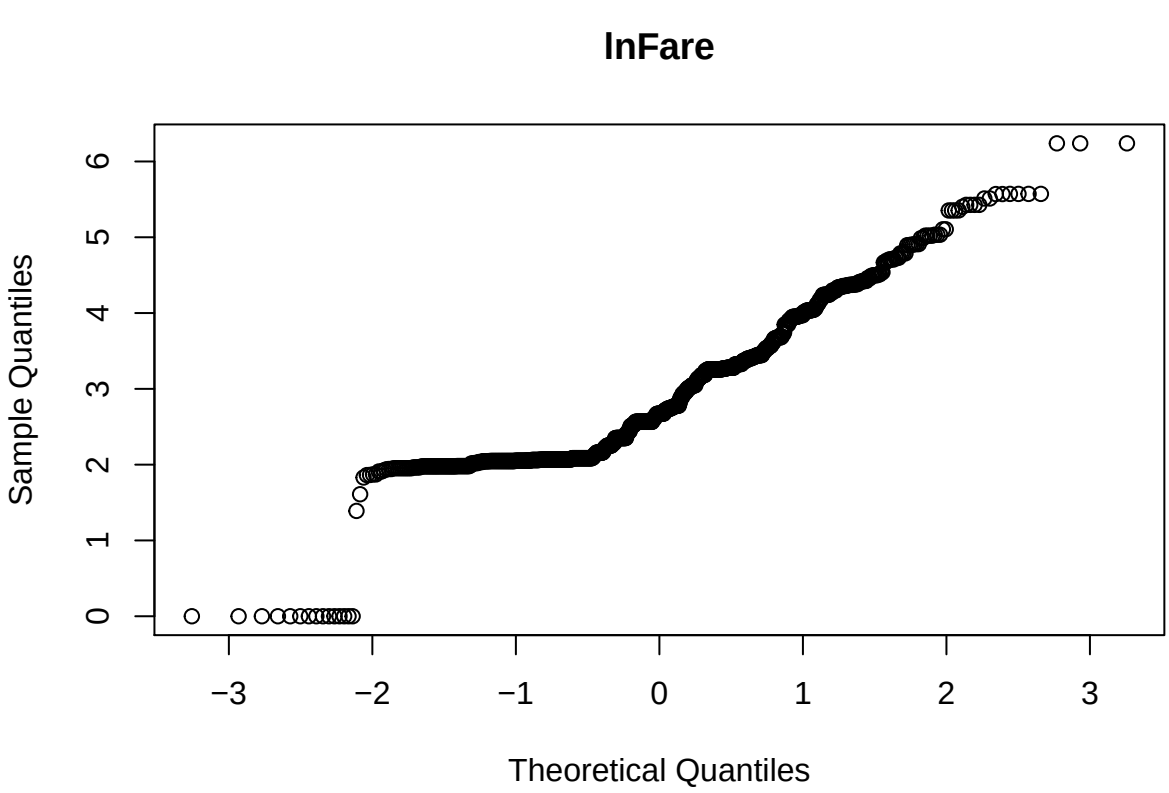
Comprovem la normalitat de les variables assumint un nivell de significació del 5%. Podem anticipar que les variables dicotòmiques o discretes de pocs valors diferents no compliran la normalitat. Ens centrarem a analitzar Age i lnFare, que són numèriques contínues. Per a fer-ho, utilitzarem el test de Shapiro-Wilk.

```r
# variables to check
var_to_check <- colnames(df_titanic_model)
for (variable in var_to_check){
  print(shapiro.test(df_titanic_model[,variable]))

  if (variable == 'lnFare' | variable == 'Age') {
    # normal Q-Q Plot
    qqnorm(df_titanic_model[,variable], main = variable)
  }
}
```

```
##
##  Shapiro-Wilk normality test
##
## data:  df_titanic_model[, variable]
## W = 0.61666, p-value < 2.2e-16
##
##
##  Shapiro-Wilk normality test
##
## data:  df_titanic_model[, variable]
## W = 0.71833, p-value < 2.2e-16
##
##
##  Shapiro-Wilk normality test
##
## data:  df_titanic_model[, variable]
## W = 0.6041, p-value < 2.2e-16
##
##
##  Shapiro-Wilk normality test
##
## data:  df_titanic_model[, variable]
## W = 0.9734, p-value = 1.106e-11
```




```
##
##  Shapiro-Wilk normality test
##
## data:  df_titanic_model[, variable]
## W = 0.62106, p-value < 2.2e-16
##
##
##  Shapiro-Wilk normality test
##
## data:  df_titanic_model[, variable]
## W = 0.92293, p-value < 2.2e-16
```




```
##
##  Shapiro-Wilk normality test
##
## data:  df_titanic_model[, variable]
## W = 0.55834, p-value < 2.2e-16
##
##
##  Shapiro-Wilk normality test
##
## data:  df_titanic_model[, variable]
## W = 0.47668, p-value < 2.2e-16
##
##
##  Shapiro-Wilk normality test
##
## data:  df_titanic_model[, variable]
## W = 0.31427, p-value < 2.2e-16
```

Podem veure que el test no resulta en normalitat, doncs el p-valor és menor al nivell de significació. No obstant, com que la mida de la mostra és gran (n>>30) tant per train (891 registres) com per test (418 registres), aplicarem el teorema del límit central i assumirem normalitat. De fet, el Q-Q plot de la variable Age no dista tant de la normalitat, que correspon a una línia recta diagonal, però pel que fa a lnFare, el dibuix dista més d'una línia recta.

4.3. Homoscedasticitat

A continuació comprovarem l'homogeneïtat de la variància de la variable Age per a cada possible valor de Sex. Ho realitzarem a partir del test de Fligner-Killeen.

```
fligner.test(formula =  Age ~ Sex, data = df_titanic_model)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  Age by Sex
## Fligner-Killeen:med chi-squared = 0.4568, df = 1, p-value = 0.4991
```

En aquest cas obtenim un p-valor superior a un nivell de significació del 5%. Per tant, com que la hipòtesi nul · la del test correspon a l'homoscedasticitat i no la rebutgem, podem concloure que la variable Age presenta variàncies estadísticament semblants pels grups female i male.

4.4. Proves estadístiques

Ens centrarem en generar tres models predictius amb les dades de train que permetin determinar amb una certa precisió la supervivència o no dels passatgers que corresponen al conjunt de test. Comencem per separar el conjunt en 2, un de train i un de test. Realitzarem una partició aleatòria del 70% dels valors a train i el 30% restant a test.

```
# set sample size
smp_size <- floor(0.7 * nrow(df_titanic_model))

# set seed to make partition reproducible
set.seed(101)
train_ind <- sample(seq_len(nrow(df_titanic_model)), size = smp_size)

# create train and test datasets
train <- df_titanic_model[train_ind,]
test <- df_titanic_model[-train_ind,]
```

4.4.1. Regressió lineal multivariable

Comencem per generar una regressió lineal de totes les variables que entren en joc per tal de predir la supervivència. Amb el model generat a partir de les dades de train, crearem una predicció amb les dades de test. Agafarem com a criteri per discernir els 1 i 0 de Survived el valor més proper a l'enter de la predicció. És a dir, els valors predits superiors a 0.5 seran 1 i indicaran que el passatger sobreviu i els menors, 0 indicant que no sobreviu.

```
# linear multivariate model with train data, using all variables
fit_lm <- lm(formula = Survived ~ Pclass + Sex + Age +
               Alone + lnFare + Embarked_S + Embarked_C + Embarked_Q,
          data = train)
summary(fit_lm)
```

```
##
## Call:
## lm(formula = Survived ~ Pclass + Sex + Age + Alone + lnFare +
##     Embarked_S + Embarked_C + Embarked_Q, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.04253 -0.25558 -0.07486  0.27627  1.02139
```

```
##
## Coefficients: (1 not defined because of singularities)
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.336252   0.164171   8.139 2.21e-15 ***
## Pclass      -0.197296   0.028525  -6.917 1.16e-11 ***
## Sex         -0.451229   0.035687 -12.644  < 2e-16 ***
## Age         -0.005133   0.001246  -4.119 4.32e-05 ***
## Alone        0.024458   0.040241   0.608    0.544
## lnFare      -0.002604   0.025224  -0.103    0.918
## Embarked_S  -0.073085   0.058374  -1.252    0.211
## Embarked_C   0.008300   0.069776   0.119    0.905
## Embarked_Q         NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3966 on 615 degrees of freedom
## Multiple R-squared:  0.3494, Adjusted R-squared:  0.342
## F-statistic: 47.18 on 7 and 615 DF,  p-value: < 2.2e-16
```

```r
# generate prediction with test data
pred.lm <- data.frame(predict.lm(fit_lm, newdata = test, interval = "prediction"))

# set to Survived=1 all predictions greater than 0.5, 0 otherwise
predicted.lm <- data.frame(ifelse(pred.lm$fit<0.5,0,1))
colnames(predicted.lm) <- c('Predicted')

# calculate Mean Standard Error to compare it with other models
MSE.lm <- sum((test$Survived - predicted.lm$Predicted)^2)/nrow(test)
```

Veiem que el model generat té un coeficient de R quadrat de 0.34, que no és massa bo. De les variables que li hem passat al model, les més significatives de cares als coeficients determinats són Pclass, Sex i Age. Embarked_S dóna significant al 90%, fet pel qual al proper model únicament considerarem les 4 anteriors. Veiem també que per a Embarked_Q no s'ha definit un coeficient degut a singularitats, així que l'algoritme directament l'ha descartat del model.

Si analitzem els primers coeficients, veiem que cada passatger té inicialment un 1.34 en 'valor' de supervivència, i que aquest es veurà afectat depenent els valors que prenguin la resta de variables. En cas que sigui de 1era classe, es reduirà el valor en -0.20, si és 2na classe en -0.39 i si és 3era classe en -0.58. A més, si és un home es redueix un -0.45 més. Finalment, es redueix un -0.005 per cada any viscut del passatger.

4.4.2. Regressió logística multivariable

A continuació realitzem un model de regressió logística multivariable, però considerant únicament les variables Pclass, Age i Sex com a variables predictores.

```r
# logistic regression survived with train data, using Pclass, Age, Sex & Alone variables
fit_lg <- glm(Survived ~ Pclass + Sex + Age,
              data = train)
summary(fit_lg)
```

```
##
## Call:
## glm(formula = Survived ~ Pclass + Sex + Age, data = train)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
```

```
## -1.07244  -0.27253  -0.08619   0.25709   1.07131
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.277871   0.068610  18.625  < 2e-16 ***
## Pclass      -0.195889   0.020245  -9.676  < 2e-16 ***
## Sex         -0.451293   0.034414 -13.114  < 2e-16 ***
## Age         -0.004773   0.001194  -3.996 7.22e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.1575922)
##
##     Null deviance: 148.65  on 622  degrees of freedom
## Residual deviance:  97.55  on 619  degrees of freedom
## AIC: 622.84
##
## Number of Fisher Scoring iterations: 2
```

```r
# generate prediction using test data
pred.lg <- data.frame(predict.glm(fit_lg, newdata = test, type = "response"))
colnames(pred.lg) <- 'fit'

# set to Survived=1 all predictions greater than 0.5, 0 otherwise
predicted.lg <- data.frame(ifelse(pred.lg$fit<0.5,0,1))
colnames(predicted.lg) <- c('Predicted')

# calculate Mean Standard Error to compare it with other models
MSE.lg <- sum((test$Survived - predicted.lg$Predicted)^2)/nrow(test)
```

Per aquest model, obtenim un AIC de 622. Veiem que els coeficients estimats no disten gaire del model anterior, però els coeficients tenen menys error associat, fet que ens indica que probablement el model s'ajustarà millor. Ho veurem en el següent apartat a l'hora de mostrar i comparar els resultats.

4.4.3. Classificació amb xarxa neuronal

Finalment, pel tercer model utilitzarem una xarxa neuronal mitjançant la llibreria neuralnet. En primer lloc, escalarem totes les dades del dataset utilitzant el rang (valors mínim i màxim per a cada variable). Realitzarem una xarxa neuronal d'una sola capa i de 3 neurones. Hi afegirem com a fórmula la mateixa que en l'apartat anterior, és a dir, Survived en funció de Pclass, Sex i Age. L'algoritme utilitzat serà el predeterminat 'rprop+', que correspon a Resilient Backpropagation with Weight Backtracking i la funció d'error també al predeterminat de suma d'errors quadrats. Finalment, realitzarem la predicció amb el conjunt de test i calcularem el MSE com en els apartats anteriors.

```r
# set seed for reproducibility
set.seed(10101)

# get min and max of each variable
max = apply(df_titanic_model, 2 , max)
min = apply(df_titanic_model, 2 , min)

# scale dataset using range (min-max)
scaled_train = as.data.frame(scale(train, center = min, scale = max - min))
scaled_test = as.data.frame(scale(test, center = min, scale = max - min))

# get names of variables
```

```r
n <- names(scaled_train)

# generate a single layer neural network with 3 neurons using train data
nn <- neuralnet(Survived ~ Pclass + Sex + Age,data = scaled_train,
                hidden = 3, linear.output=TRUE)

# show neural network visually
plot(nn)

# generate prediction
pred.nn <- data.frame(compute(nn,subset(x = scaled_test, select = -c(Survived))))

# set to Survived=1 all predictions greater than 0.5, 0 otherwise
predicted.nn <- data.frame(ifelse(pred.nn$net.result<0.5,0,1))
colnames(predicted.nn) <- c('Predicted')

# calculate Mean Standard Error to compare it with other models
MSE.nn <- sum((scaled_test$Survived - predicted.nn$Predicted)^2)/nrow(test)
```

Podem veure en el gràfic de la xarxa neuronal les 3 variables input Pclass, Sex i Age, les 3 neurones en la única capa neuronal i l'output de Survived. Les fletxes i els valors indiquen els pesos sinàptics entre nodes i les fletxes en blau indiquen el bias en cada connexió.

5. Representació dels resultats

A continuació comparem els resultats dels 3 models anteriors. En primer lloc, generem taules de contrast entre els valors predits per cada model i els valors reals del conjunt de test.

```r
# check OKs & KOs linear model
table(predicted.lm$Predicted,test$Survived)
```

```
##
##       0   1
##   0 151  21
##   1  20  76
```

```r
cat('MSE lm: ', MSE.lm)
```

```
## MSE lm:  0.1529851
```

```r
# check OKs & KOs logistic model
table(predicted.lg$Predicted,test$Survived)
```

```
##
##       0   1
##   0 153  21
##   1  18  76
```

```r
cat('MSE lg: ', MSE.lg)
```

```
## MSE lg:  0.1455224
```

```r
# check OKs & KOs neural network model
table(predicted.nn$Predicted,scaled_test$Survived)
```

```
##
##       0   1
##   0 160  39
```

```
##   1  11  58
```

```
cat('MSE nn: ', MSE.nn)
```

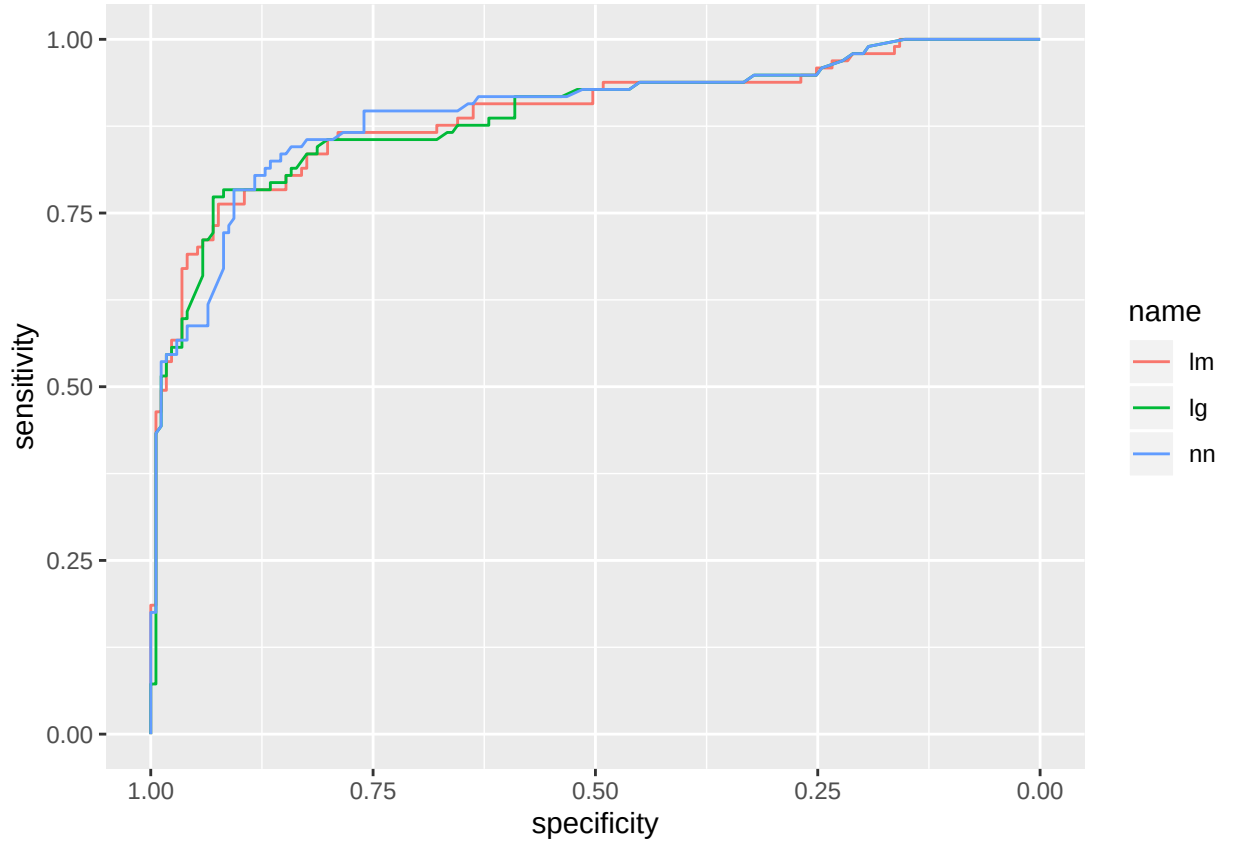
```
## MSE nn:  0.1865672
```

En les taules podem veure la relació entre encerts i errors de cada model predictiu enfront els valors reals. Veiem que tant el model logístic com el lineal són els que generen més encerts en supervivència. No obstant, el model lineal genera més supervivents dels que realment ho són. Es tractaria d'un model més 'optimista' que el logístic. Finalment, el model de xarxes neuronals és el que encerta el màxim nombre de defuncions, però falla molt a l'hora d'encertar supervivents. Es tractaria d'un model 'pessimista'.

Si revisem els MSE, veiem que el model amb menys error correspon al logístic, seguit del lineal i finalment tindríem el de la xarxa neuronal.

Finalment, representarem la corba ROC dels tres models.

```
# generate roc curves for each model
roc_lm <- roc(response = test$Survived, predictor = pred.lm$fit)
roc_lg <- roc(response = test$Survived, predictor = pred.lg$fit)
roc_nn <- roc(response = test$Survived, predictor = pred.nn$net.result)

ggroc(data = list(lm=roc_lm, lg=roc_lg, nn=roc_nn), legacy.axis = TRUE)
```




```
cat('AUC lm: ', roc_lm$auc,'\nAUC lg: ', roc_lg$auc,'\nAUC nn: ', roc_nn$auc)
```

```
## AUC lm:  0.8918129
## AUC lg:  0.8901549
## AUC nn:  0.8950383
```

Amb aquesta representació i el càlcul de l'AUC (Area Under the Curve), veiem no obstant que la millor corba és la del model de xarxa neuronal (nn), seguida del model lineal (lm) i del model logístic (lg). Això es deu a què el model de la xarxa neuronal és millor en especificitats menors al 85%, mentre que el model lineal supera o iguala al logístic a excepció de la regió del 80% d'especificitat. En qualsevol cas, els 3 valors són molt semblants.

6. Resolució del problema

En aquest estudi hem pogut determinar les variables que més van influenciar en la supervivència o no dels passatgers del Titanic. Després d'haver realitzat tres models predictius a partir de les dades d'entrenament, hem calculat les prediccions i comparat amb el dataset de test i hem arribat a les següents conclusions:

1. La supervivència o no d'un passatger ve influenciada principalment per la classe en la què viatjava, el sexe i l'edat. En general, viatjant amb una classe baixa, ser home i tenir una edat avançada implica més predisposició a no salvar-se. S'ha aconseguit crear un model de regressió logística de 3 variables que és capaç de predir la supervivència del conjunt de test amb un MSE menor al 15%.
2. Les variables Fare (o lnFare), Alone o Embarked no són tant significatives per determinar la supervivència. La resta de variables tenen una major o menor correlació amb la supervivència, però aquesta acaba sent absorbida per altres variables, com per exemple per la classe en la què viatjaven.
3. Malgrat haver realitzat l'estudi per Alone, al tractar-se d'una variable dicotòmica construïda a partir de FamSize, que a la vegada és una variable construïda a partir de SibSp i Parch; no podem assegurar que el nombre de fills, germans, cònjugues o altres relacions familiars influenciï positiva o negativament en la supervivència. Un enfoc diferent hagués pogut ser estudiar aquesta relació.
4. No podem extreure conclusions segons el Ticket o Cabin, doncs són variables que hem descartat per fer l'anàlisi. També serien possibles variables a estudiar.
5. El model logístic ha resultat ser el més bo comparant-ho amb el lineal multivariant amb totes les variables i el de la xarxa neuronal. No obstant, el model lineal ha estat precursor al logístic i no s'ha estudiat com treballaria únicament amb les variables Pclass, Sex i Age com en els altres dos casos. Per altra banda, el model de la xarxa neuronal pot ser millorat modificant-ne els paràmetres com el nombre de capes neuronals, nombre de neurones, algoritme utilitzat per entrenar la xarxa, funció d'error, variables predictores a utilitzar, tipus d'escalat en les variables predictores, etc. Treballar aquest model més a fons pot fer trobar una combinació de paràmetres que resultin en una millor predicció.

Finalment generem la predicció amb el model de regressió logística amb els valors de test, per a entrar en la competició de Kaggle.

```
# generate prediction using true test data
pred.lg_2 <- data.frame(predict.glm(fit_lg, newdata = df_titanic_test, type = "response"))
colnames(pred.lg_2) <- 'fit'

# set to Survived=1 all predictions greater than 0.5, 0 otherwise
predicted.lg_2 <- data.frame(ifelse(pred.lg_2$fit<0.5,0,1))

# generate submission dataframe
orp_submission <- cbind(subset(df_titanic_test,select=PassengerId),predicted.lg_2)
colnames(orp_submission) <- c('PassengerId','Survived')

# save new file
write.csv(orp_submission, "orp_submission.csv")
```