Pràctica 2: Neteja i validació de dades

Tipologia i cicle de vida de les dades

Jesús Marí i Víctor Boix

Data de presentació: 11/06/2019

Índex

---

1. Descripció del dataset

El dataset utilitzat en aquesta pràctica s'ha obtingut de la pàgina web de *Kaggle* (https://www.kaggle.com/c/titanic) i conté les dades dels passatgers del Titanic durant el desastre del 1912. L'objectiu d'aquest conjunt de dades és realitzar una predicció sobre la supervivència d'alguns passatgers a partir de les seves característiques. Per tant, es tracta d'un problema de classificació d'aprenentatge supervisat, on cal construir un model capaç de determinar el valor de l'atribut *Survived* (variable objectiu o dependent) a partir de la resta d'atributs (variables independents).

Les conjunt de dades està format per dos fitxers CSV:

- *Titanic_train.csv*. Conjunt d'entrenament que servirà per entrenar el model, per tant, conté l'atribut objectiu o classe *Survived*. Està format per 891 registres i 12 atributs.
- *Titanic_test.csv*. Conjunt de prova, conté els registres sobre els que cal realitzar la predicció. Està format per 418 registres i 11 atributs.

La informació que contenen els atributs és la següent:

- *PassengerId*. Identificador únic del viatger al dataset (nombre enter).
- *Survived*. Atribut que indica si el passatger va sobreviure a la catàstrofe (1: va sobreviure, 0: va morir). Només disponible al conjunt *train*.

- *Pclass*. Indica la classe en què viatjava el passatger (1: primera classe, 2: segona classe, 3: tercera classe).
- *Name*. Nom complet del viatger (cadena de text).
- *Sex*. Sexe del viatger (male: home, female: dona).
- *Age*. Edat del viatger en anys (nombre real).
- *SibSp*. Nombre de germans, germanes o marit/muller a bord del vaixell (nombre enter).
- *Parch*. Nombre de pares o fills a bord del vaixell (nombre enter).
- *Ticket*. Nombre del tiquet (cadena de text).
- *Fare*. Import pagat pel bitllet (nombre real).
- *Cabin*. Cabina on s'allotjava el passatger (cadena de text).
- *Embarked*. Port d'embarcament del viatger (C: Cherbourg, Q: Queenstown, S: Southampton).

2. Integració i selecció de les dades

Per tal de realitzar els processos de neteja i anàlisi de les dades carregarem els dos fitxers en dos dataframes que anomenarem *train* i *test*. Totes les operacions de neteja les realitzarem sobre els dos dataframes.

```
# Càrrega dels fitxers de dades
train <- read.csv('../data/Titanic_train.csv')
test <- read.csv('../data/Titanic_test.csv')
```

L'atribut *PassengerId* és un identificador numèric dels registres del dataframe que no ens aporta cap informació sobre els passatgers, per tant, l'eliminarem, ja que per identificar els registres ja disposem de l'índex.

```
# Eliminem l'atribut PassengerId a train i test
train <- train[,-1]
test <- test[,-1]
```

La resta d'atributs els mantindrem perquè ens donen informació sobre el viatger i poden resultar útils per al model. Podem fer una visualització dels primers registres amb la funció *head()*.

```
# Primers registres del dataframe train
head(train)
```

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

3. Neteja de les dades

Comencem examinant el dataframe amb la funció *str()*, que ens mostra el tipus de dada i els valors dels primers registres per a cada atribut.

```r
# Examinem el conjunt train
str(train)
```

```
## 'data.frame':    891 obs. of  11 variables:
##  $ 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 ...
```

```r
# Examinem el conjunt test
str(test)
```

```
## 'data.frame':    418 obs. of  10 variables:
##  $ 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 58 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 ...
```

L'atribut *Name* és de tipus factor, però té tant nivells com registres, per tant el convertirem a character.

```r
# Conversió de Name a character
train[,'Name'] <- as.character(train[,'Name'])
test[,'Name'] <- as.character(test[,'Name'])
```

Els tipus de dada per a cada atribut són:

- *Factor*: Sex, Ticket, Cabin, Embarked
- *int*: Survived, Pclass, SibSp, Parch
- *num*: Age, Fare
- *chr*: Name

Abans d'analitzar els elements buit i extrems pot resultar útil mostrar un resum de les dades amb la funció *summary()*.

```r
# Resum de les dades train
summary(rbind(train[-1],test))
```

```
##      Pclass          Name              Sex           Age       
##  Min.   :1.000   Length:1309        female:466   Min.   : 0.17  
##  1st Qu.:2.000   Class :character   male  :843   1st Qu.:21.00  
##  Median :3.000   Mode  :character                Median :28.00  
##  Mean   :2.295                                   Mean   :29.88  
##  3rd Qu.:3.000                                   3rd Qu.:39.00  
##  Max.   :3.000                                   Max.   :80.00  
##                                                  NA's   :263
```

```
##      SibSp           Parch           Ticket          Fare
##  Min.   :0.0000   Min.   :0.000   CA. 2343:  11   Min.   :  0.000
##  1st Qu.:0.0000   1st Qu.:0.000   1601    :   8   1st Qu.:  7.896
##  Median :0.0000   Median :0.000   CA 2144 :   8   Median : 14.454
##  Mean   :0.4989   Mean   :0.385   3101295 :   7   Mean   : 33.295
##  3rd Qu.:1.0000   3rd Qu.:0.000   347077  :   7   3rd Qu.: 31.275
##  Max.   :8.0000   Max.   :9.000   347082  :   7   Max.   :512.329
##                                   (Other) :1261   NA's   :1
##              Cabin      Embarked
##                 :1014    :  2
##  C23 C25 C27    :   6   C:270
##  B57 B59 B63 B66:   5   Q:123
##  G6             :   5   S:914
##  B96 B98        :   4
##  C22 C26        :   4
##  (Other)        : 271
```

3.1. Elements buits

Per començar examinem els elements buits a les columnes dels dos conjunts de dades. Podem veure que tenim molts valors NA a la columna *Age* dels dos dataframes i un a la columna *Fare* del conjunt de test.

```
# Nombre de valors NA per atribut
sapply(train, function(x) sum(is.na(x)))
```

```
## Survived   Pclass     Name      Sex      Age    SibSp    Parch   Ticket
##        0        0        0        0      177        0        0        0
##     Fare    Cabin Embarked
##        0        0        0
```

```
sapply(test, function(x) sum(is.na(x)))
```

```
##   Pclass     Name      Sex      Age    SibSp    Parch   Ticket     Fare
##        0        0        0       86        0        0        0        1
##    Cabin Embarked
##        0        0
```

A més, els atributs *Cabin* i *Embarked* també contenen valors buits.

```
# Nombre de valors buits per atribut
sapply(train, function(x) nrow(train[x=='',]))
```

```
## Survived   Pclass     Name      Sex      Age    SibSp    Parch   Ticket
##        0        0        0        0      177        0        0        0
##     Fare    Cabin Embarked
##        0      687        2
```

```
sapply(test, function(x) nrow(test[x=='',]))
```

```
##   Pclass     Name      Sex      Age    SibSp    Parch   Ticket     Fare
##        0        0        0       86        0        0        0        1
##    Cabin Embarked
##      327        0
```

Fare

Per imputar el valor buit del preu del bitllet (*Fare*) podem utilitzar la informació de la classe del viatger (*Pclass*), ja que és lògic pensar que els bitllets de millor classe seran més cars. Això ho podem confirmar

mostrant la mitjana de *Fare* per a cada valor de *Pclass*.

```
# Mitjana de Fare per cada valor de Pclass als conjunts train i test conjuntament
aggregate(Fare~Pclass, data=rbind(train[,-1],test), mean)
```

```
##   Pclass     Fare
## 1      1 87.50899
## 2      2 21.17920
## 3      3 13.30289
```

Com veiem, hi ha una diferència significativa entre la mitjana del preu de bitllet per a cada classe. Utilitzarem aquesta circumstància per imputar el valor NA de *Fare* amb la mitjana de valors per als viatgers de la mateixa classe.

```
# Imputem el valor NA de Fare amb la mitjana de valors de la mateixa classe
test[is.na(test$Fare),'Fare'] <- aggregate(Fare~Pclass, data=rbind(train[,-1],test),
                                           mean)[test[is.na(test$Fare),'Pclass'],'Fare']
```

Cabin

En el cas de l'atribut *Cabin*, substituirem els valors buits pel marcador 'NO'.

```
levels(train$Cabin)[levels(train$Cabin)==''] <- 'NO'
levels(test$Cabin)[levels(test$Cabin)==''] <- 'NO'
```

Embarked

Per imputar els valors buits del port d'embarcament (*Embarked*) podem aprofitar la informació del bitllet. Si consultem les dades dels viatgers que tenen un valor buit a *Embarked* veiem que tenen el mateix número de bitllet.

```
# Ticket dels viatgers amb valor buit a Embarked
train[train$Embarked=='',c("Embarked", "Ticket")]
```

```
##     Embarked Ticket
## 62          113572
## 830         113572
```

Si ampliem la cerca i mostrem el valor d'*Embarked* per a tots els viatgers amb un número de bitllet semblant (113XXX) veiem que la majoria han pujat a Southampton ('S'). Per tant assignarem aquest valor als dos valors buits d'*Embarked*.

```
# Viatgers amb un número de ticket semblant a l'anterior (113XXX) agrupats pel valor d'Embarked
 summary(train[grep("113.*",train$Ticket),"Embarked"])
```

```
##     C  Q  S
##  2  7  0 42
```

```
# Assignem el valor 'S' als valors buits d'Embarked
levels(train$Embarked) <- c("S", "C", "Q", "S")
```

Age

Finalment, per imputar el valor buit de l'edat (*Age*) farem servir el mètode basat en els k-veïns més propers (KNN). El que fa l'algorisme knn-imputation és identificar les k-observacions més properes i assignar la mitjana ponderada a l'atribut amb valor NA. En el nostre cas utilitzarem k=3 i tindrem en compte tots els atributs excepte *Name*.

```
# Imputació de valors amb VIM
train$Age <- kNN(train[,-3], k=3)$Age
test$Age <- kNN(test[,-2], k=3)$Age
```

Per acabar, comprovem que ja no hi ha cap valor NA o buit.

```
# Nombre de valors NA per atribut
sapply(train, function(x) sum(is.na(x)))
```

```
## Survived   Pclass     Name      Sex      Age    SibSp    Parch   Ticket
##        0        0        0        0        0        0        0        0
##     Fare    Cabin Embarked
##        0        0        0
```

```
sapply(test, function(x) sum(is.na(x)))
```

```
##   Pclass     Name      Sex      Age    SibSp    Parch   Ticket     Fare
##        0        0        0        0        0        0        0        0
##    Cabin Embarked
##        0        0
```

```
# Nombre de valors buits per atribut
sapply(train, function(x) nrow(train[x=='',]))
```

```
## Survived   Pclass     Name      Sex      Age    SibSp    Parch   Ticket
##        0        0        0        0        0        0        0        0
##     Fare    Cabin Embarked
##        0        0        0
```

```
sapply(test, function(x) nrow(test[x=='',]))
```

```
##   Pclass     Name      Sex      Age    SibSp    Parch   Ticket     Fare
##        0        0        0        0        0        0        0        0
##    Cabin Embarked
##        0        0
```

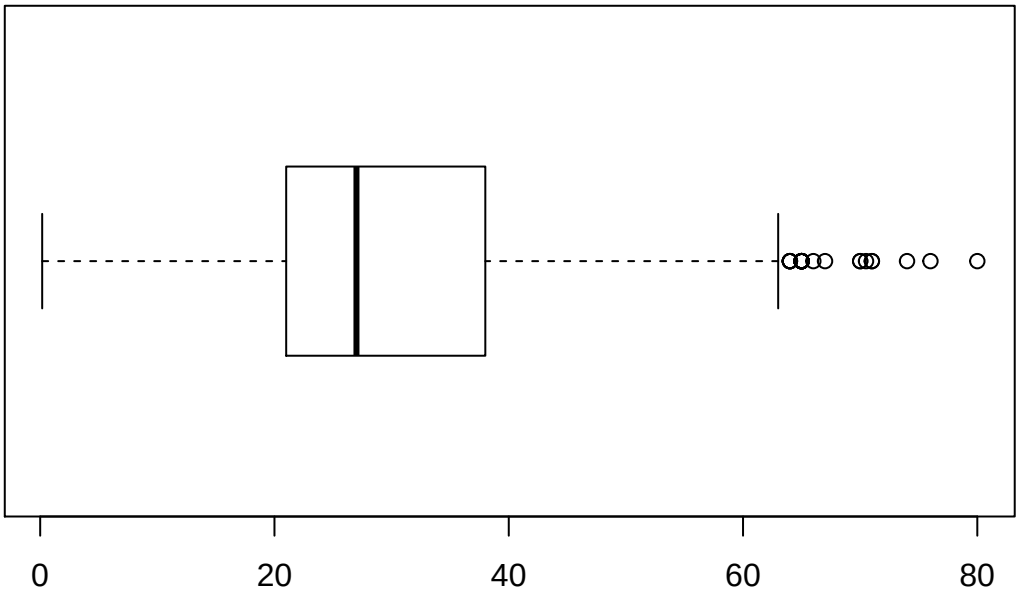
3.2. Valors extrems

Els valors extrems o *outliers* són aquells valors que es troben tan allunyats de la resta de valors que ens poden fer pensar que són erronis o tenen un origen diferent. Generalment es consideren *outliers* els valors que estan a més de 3 desviacions estàndards de la mitjana de la població. A continuació estudiarem els valors extrems per als atributs de tipus numèric, tant enters com reals.

Age

Si representem el diagrama de caixa de l'atribut *Age* veiem que la majoria de passatgers se situen entre els 20 i els 40 anys, però també apareixen alguns punts aïllats per sobre dels 60 anys.

```
# Diagrama de caixa de l'atribut Age
A.Age <- c(train$Age,test$Age)
boxplot(A.Age, horizontal = TRUE)
```



Si mostrem els valors extrems per a aquest atribut obtenim molts valors entre 60 i 80.

```
# Possibles outliers a l'atribut Age
boxplot.stats(A.Age)$out
```

```
##  [1] 66.0 65.0 71.0 70.5 65.0 65.0 65.0 65.0 65.0 65.0 64.0 65.0 65.0 71.0
## [15] 64.0 65.0 65.0 80.0 70.0 70.0 65.0 65.0 74.0 67.0 76.0 64.0 64.0 64.0
```

Si calculem els valors que es troben per sobre de 3 desviacions estàndard el nombre d'*outliers* disminueix a només 3.

```
# Edats superiors a 3 desviacions estàndard per sobre de la mitjana
A.Age[A.Age > (mean(A.Age) + 3 * sd(A.Age))]
```

```
## [1] 80 74 76
```

```
# Rang total de l'atribut Age
range(A.Age)
```

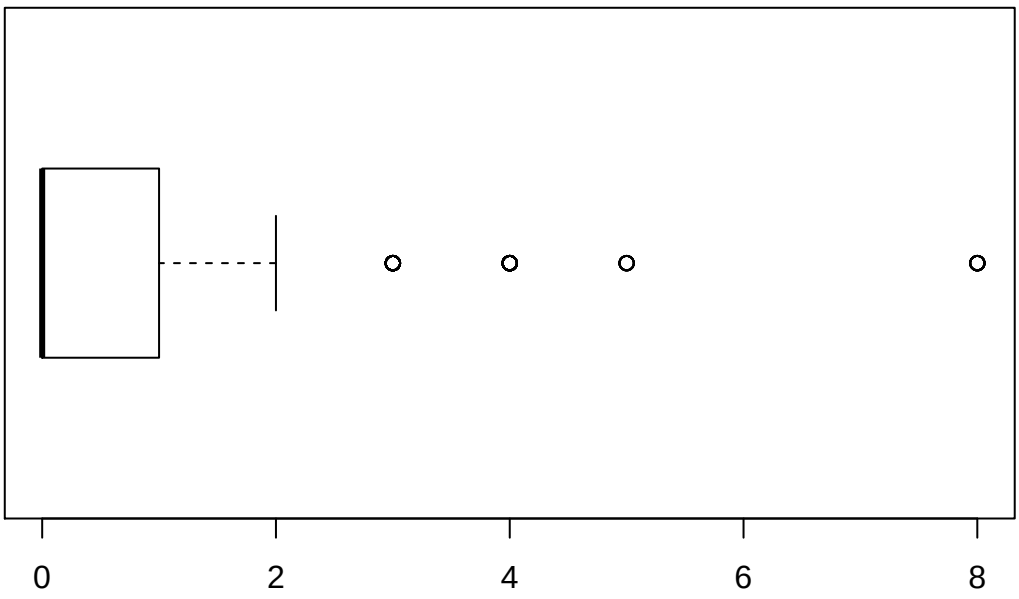
```
## [1]  0.17 80.00
```

Tot i que hem detectat 3 possibles outliers, tots els passatgers se situen entre els 0 i els 80 anys, que són edats perfectament possibles i per tant mantindrem tots els registres sense cap modificació.

SibSp

L'atribut *SibSp* ens indica el nombre de germans o marit/muller de cada passatger. Si mostrem la distribució de valors en una diagrama de caixa veiem que la majoria de valors se situen entre 0 i 1.

```
# Diagrama de caixa de l'atribut SibSp
A.SibSp <- c(train$SibSp, test$SibSp)
boxplot(A.SibSp, horizontal = TRUE)
```



Repetint les anàlisis anteriors veiem que la funció *boxplot* detecta com a extrems els valors superiors a 2, si calculem els valors superiors a 3 desviacions estàndard trobem els valors superiors a 3, mentre que el valor més alt de l'atribut és de 8. Tots aquests valors són perfectament possibles com a nombre de fills i per tant també els mantidrem.

```
# Possibles outliers a l'atribut SibSp
boxplot.stats(A.SibSp)$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
```

```
# Valors per sobre de 3 SD
A.SibSp[A.SibSp > (mean(A.SibSp) + 3 * sd(A.SibSp))]
```

```
##  [1] 4 4 5 4 5 4 8 4 4 8 4 8 4 4 4 4 8 5 5 4 4 5 4 4 8 4 4 8 4 8 4 5 4 8 4
## [36] 8 4
```

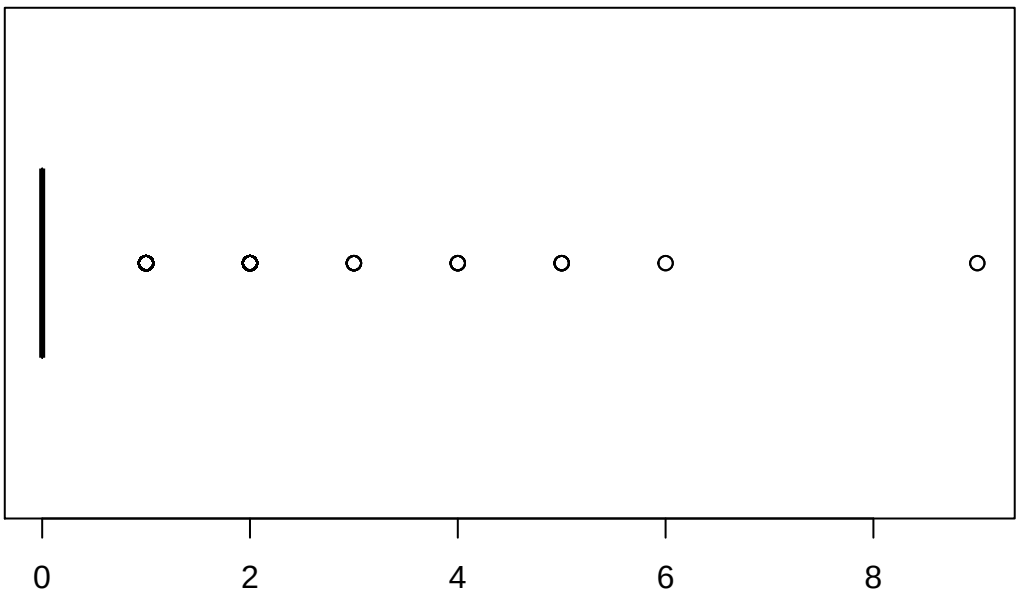
```
# Rang total de l'atribut
range(A.SibSp)
```

```
## [1] 0 8
```

Parch

L'atribut *Parch* indica el nombre de pares o fills al vaixell per a cada passatger. Si repetim els càlculs obtenim un resultat molt semblant a l'anterior. En aquest cas, la majoria de registres tenen un valor de 0, això fa que la mitjana sigui molt baixa i la resta de valors apareguin com a outliers. Malgrat tot, els rang de l'atribut se situa entre 0 i 9 fills, que són valors possibles i que encaixen amb els resultats de *SibSp*, per tant també els mantindrem.

```
# Diagrama de caixa de l'atribut Parch
A.Parch <- c(train$Parch, test$Parch)
boxplot(A.Parch, horizontal = TRUE)
```




```
# Possibles outliers a l'atribut Parch
boxplot.stats(A.Parch)$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
```

```
# Valors per sobre de 3 SD
A.Parch[A.Parch > (mean(A.Parch) + 3 * sd(A.Parch))]
```

```
##  [1] 5 5 3 4 4 3 4 4 5 5 6 3 3 3 5 3 4 4 6 3 5 3 9 9
```

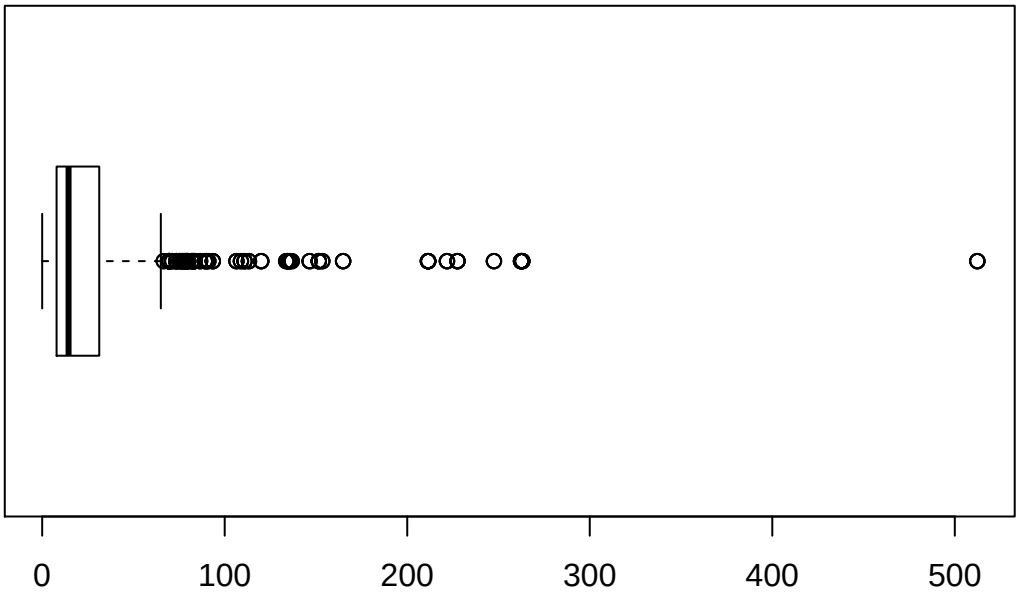
```
# Rang total de l'atribut
range(A.Parch)
```

```
## [1] 0 9
```

Fare

Per acabar estudiarem els valors extrem de l'atribut *Fare*, que indica el preu del bitllet de cada passatger. En aquest cas veiem que la majoria de bitllets se situen per sota de 50, tenim molts possibles *outliers* per sobre de 80 i un valor màxim superior als 500.

```
# Diagrama de caixa de l'atribut Fare
A.Fare <- c(train$Fare, test$Fare)
boxplot(A.Fare, horizontal = TRUE)
```




```
# Possibles outliers a l'atribut Fare
boxplot.stats(A.Fare)$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
```

```
# Valors per sobre de 3 SD
A.Fare[A.Fare > (mean(A.Fare) + 3 * sd(A.Fare))]
```

```
##  [1] 263.0000 263.0000 247.5208 512.3292 247.5208 262.3750 263.0000
##  [8] 211.5000 227.5250 263.0000 221.7792 227.5250 512.3292 211.3375
## [15] 227.5250 227.5250 211.3375 512.3292 262.3750 211.3375 262.3750
## [22] 263.0000 262.3750 262.3750 263.0000 211.5000 211.5000 221.7792
```

```
## [29] 221.7792 262.3750 221.7792 247.5208 227.5250 211.5000 211.3375
## [36] 512.3292 262.3750 211.5000
```

```r
# Rang total de l'atribut
range(A.Fare)
```

```
## [1]   0.0000 512.3292
```

Com en els casos anteriors, per a aquest atribut també hem decidit mantenir tots els valors.

3.3. Conversió de dades

A continuació transformarem alguns atributs per tal de reduir-ne el nombre de categories o obtenir-ne informació que pugui resultat interessant per a la fase d'anàlisi.

Name

A l'atribut *Name* n'extraurem el títol de tractament, ja que indica l'estatus del viatger i pot influir en la seva supervivencia.

```r
# Extraiem el títol de l'atribut Name
train$Title <- sub(".*\\s([A-Za-z]+)\\..*", "\\1", train$Name)
test$Title <- sub(".*\\s([A-Za-z]+)\\..*", "\\1", test$Name)

# Eliminem l'atribut Name perquè ja no ens serà útil
train <- train[,-3]
test <- test[,-2]

# Nombre de registres per categoria de Title
table(c(train$Title, test$Title))
```

```
##
##     Capt      Col Countess      Don     Dona       Dr Jonkheer        L
##        1        4        1        1        1        8        1        1
##     Lady    Major   Master     Miss     Mlle      Mme       Mr      Mrs
##        1        2       61      260        2        1      757      196
##       Ms      Rev      Sir
##        2        8        1
```

A continuació, com que el nombre de categories és força gran i algunes tenen molt pocs registres, unirem els títols poc freqüents ens una mateixa categoria 'other'. Per acabar, convertim l'atribut a tipus factor.

```r
# Marquem com a 'other' els títols menys freqüents
others <- c("Capt","Countess","Don","Dona","Jonkheer","L","Lady","Mme","Sir",
            "Major","Mlle","Ms","Col","Dr","Rev")
train[train$Title %in% others, 'Title'] = 'other'
test[test$Title %in% others, 'Title'] = 'other'

# Convertim Title a factor
train$Title <- as.factor(train$Title)
test$Title <- as.factor(test$Title)
```

Cabin

L'atribut cabina conté més de 100 nivells. Una manera de reduir-los pot ser eliminar els números i conservar únicament la lletra, ja que segurament indica el tipus de cabina o la zona on s'allotjaven els passatgers i pot resultar significativa.

```
# Extraiem la lletra de l'atribut Cabin
levels(train$Cabin)[-1] <- sub("([A-T]).*","\\1", levels(train$Cabin)[-1])
levels(test$Cabin)[-1] <- sub("([A-T]).*","\\1", levels(test$Cabin)[-1])

# Freqüència de supervivents per cabina
table(train[,c("Survived","Cabin")])
```

```
##          Cabin
## Survived  NO   A   B   C   D   E   F   G   T
##        0 481   8  12  24   8   8   5   2   1
##        1 206   7  35  35  25  24   8   2   0
```

Com veiem, les cabines A, F, G i T tenen molt pocs passatgers i un percentatge de supervivents força semblant. Per evitar tenir cabines amb pocs registres i millorar l'eficàcia de les anàlisis unirem aquestes cabines en una mateixa categoria que anomenarem X.

```
# Reanomenem a X les cabines A, F, G i T
levels(train$Cabin) <- c("NO", "X", "B", "C", "D", "E", "X", "X", "X")
levels(test$Cabin) <- c("NO", "X", "B", "C", "D", "E", "X", "X", "X")
```

Ticket

Finalment, tot i que l'atribut *Ticket* també conté molts nivells i sembla poc útil, podem provar d'extreure'n alguna informació interessant. A continuació mostrem les dades d'alguns passatgers que comparteixen el valor de *Ticket*.

```
# Valors més freqüents de Ticket
sort(table(train$Ticket), decreasing = TRUE)[c(1:5)]
```

```
##
##     1601   347082 CA. 2343  3101295   347088
##        7        7        7        6        6
```

```
# Passatgers amb el Ticket més freqüent
train[train$Ticket=='1601',]
```

```
##     Survived Pclass  Sex Age SibSp Parch Ticket    Fare Cabin Embarked
## 75         1      3 male  32     0     0   1601 56.4958    NO        S
## 170        0      3 male  28     0     0   1601 56.4958    NO        S
## 510        1      3 male  26     0     0   1601 56.4958    NO        S
## 644        1      3 male  32     0     0   1601 56.4958    NO        S
## 693        1      3 male  32     0     0   1601 56.4958    NO        S
## 827        0      3 male  28     0     0   1601 56.4958    NO        S
## 839        1      3 male  32     0     0   1601 56.4958    NO        S
##     Title
## 75     Mr
## 170    Mr
## 510    Mr
## 644    Mr
## 693    Mr
## 827    Mr
## 839    Mr
```

Com veiem, els passatgers amb un mateix valor per a *Ticket* també comparteixen moltes característiques, perquè segurament formaven una família o viatjaven junts. Això ho aprofitarem per calcular la mida del grup de persones que viatjaven conjuntament i crear l'atribut *Group*.

```
# Creem una taula de freqüències per als tickets
tickets <- as.data.frame(table(rbind(train["Ticket"],test["Ticket"])))
colnames(tickets) <- c("Ticket","Group")

# Assignem el valor de la freqüència a un nou atribut "Group"
train <- join(train, tickets, by="Ticket")
test <- join(test, tickets, by="Ticket")

# Eliminem l'atribut Ticket
train <- train[-7]
test <- test[-6]
```

Fare

Finalment, com que el preu del bitllet sembla ser el mateix per a cada ticket, és a dir, sembla tractar-se de l'import per a tot el grup, dividirem l'import del passatge (*Fare*) per les dimensions del grup (*Group*) per tal d'obtenir el preu individual del bitllet.

```
# Càlcul del preu individual del bitllet
train$Fare <- train$Fare / train$Group
test$Fare <- test$Fare / test$Group
```

3.4. Exportació de dades

Una vegada hem netejat les dades les exportarem en un nou fitxer.

```
# Exportació dels conjunt train i test nets
write.csv(train, '../data/Titanic_train_clean.csv')
write.csv(test, '../data/Titanic_test_clean.csv')
```

4. Anàlisi de dades

4.1. Selecció de dades

La fase d'anàlisi s'enfocarà en estudiar les dades des de tres punts de vista i intentant respondre tres preguntes diferents:

1. Quines variables numèriques influeixen més en la supervivència dels passatgers?
2. La probabilitat de sobreviure és significativament diferent entre les diferents categories?
3. Quina és la probabilitat de sobreviure per als passatgers del conjunt de test?

Per a respondre la primera pregunta treballarem sobre el conjunt *train*, que és l'únic que conté el valor de la variable objectiu (*Survived*), i estudiarem el grau de correlació entre *Survived* i la resta de variables quantitatives (*Pclass*, *Age*, *SibSp*, *Parch*, *Fare* i *Group*).

La segona pregunta se centra en les variables qualitatives (*Sex*, *Cabin*, *Embarked* i *Title*) on avaluarem si hi ha diferències significatives en el percentatge de supervivents entre els diferents grups definits per aquestes variables. Per fer aquesta anàlisi utilitzarem el test de χ^2 amb la funció *chisq.test*.

Finalment, la tercera pregunta és un problema de classificació. Treballarem amb el conjunt *train* per crear un model de regressió logística a partir de totes les variables, tant les numèriques com les categòriques, i l'utilitzarem per efectuar prediccions sobre el conjunt de *test*.

```r
# Variables numèriques
var_num <- c("Pclass","Age","SibSp","Parch","Fare","Group")

# Variables categòriques
var_cat <- c("Sex","Cabin","Embarked","Title")
```

Abans de fer totes aquestes anàlisis comprovarem la normalitat i homocedasticitat de les variables quantitatives.

4.2. Normalitat i homogeneïtat de la variància

Per fer les anàlisis de normalitat i homogeneïtat de la variància ens centrarem en els atributs de tipus *numeric*, que són *Age* i *Fare*. En primer lloc unirem els dos conjunts de dades en un mateix dataframe per facilitar-ne l'estudi.

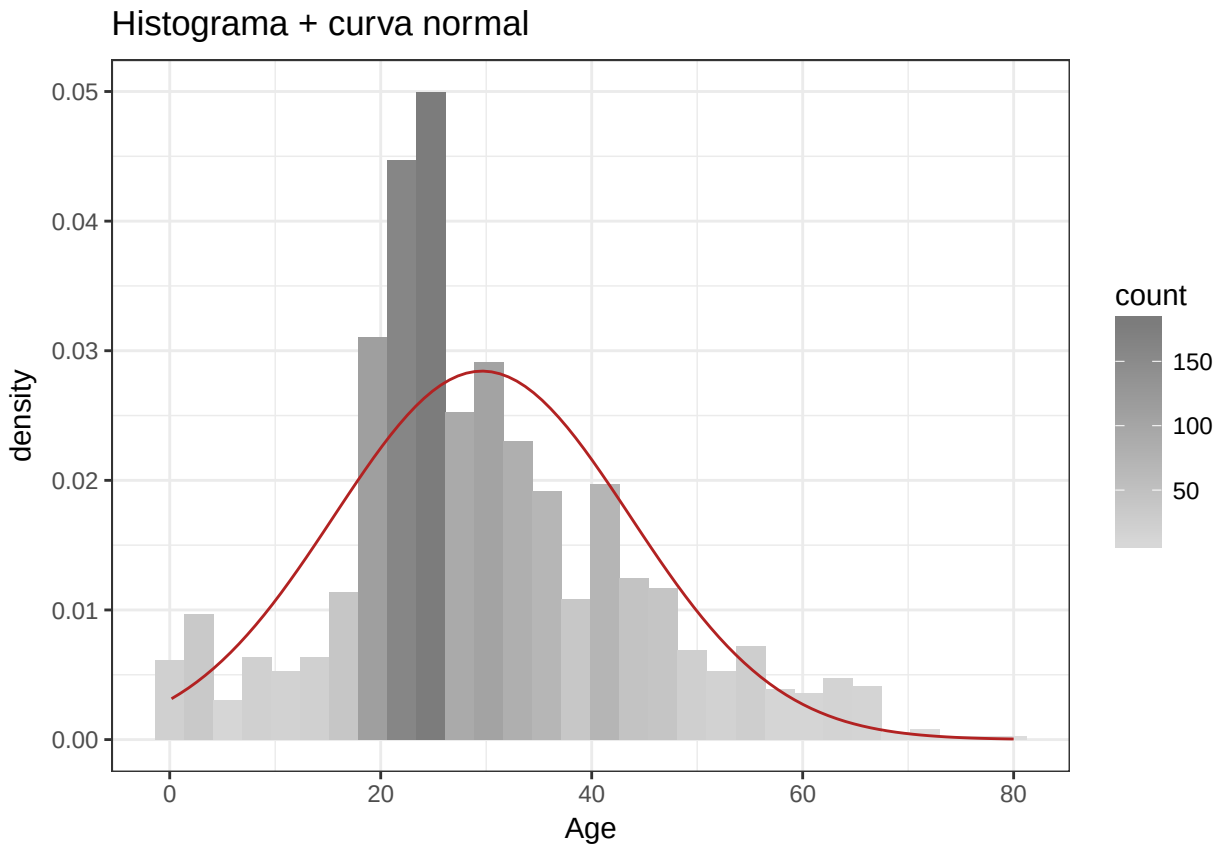
```r
# Unim les dades en un dataframe
test_and_train <- rbind(train[c("Age","Fare")],test[c("Age","Fare")])
```

Age

En primer lloc, analitzarem la normalitat de la distribució de l'atribut *Age*, primer de manera gràfica i després amb el test de Shapiro-Wilk.

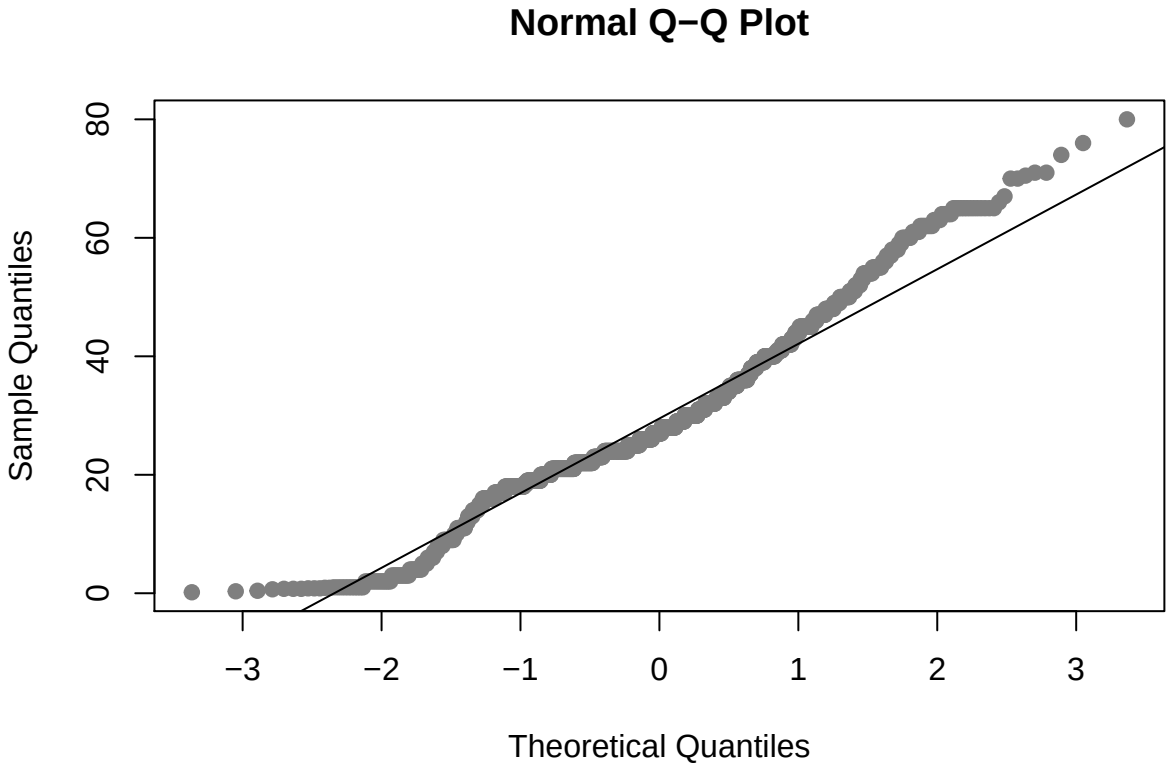
```r
# Representem les dades en un histograma
ggplot(data = test_and_train, aes(x = Age)) +
  geom_histogram(aes(y = ..density.., fill = ..count..)) +
  scale_fill_gradient(low = "#DCDCDC", high = "#7C7C7C") +
  stat_function(fun = dnorm, colour = "firebrick",
                args = list(mean = mean(test_and_train$Age),
                            sd = sd(test_and_train$Age))) +
  ggtitle("Histograma + curva normal") +
  theme_bw()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Com veiem, tot i que la majoria de franges d'edat s'acosten a una distribució normal, l'interval aproximat entre 18 i 25 anys és molt més freqüent que la resta i s'allunya molt d'aquesta distribució. La gràfica Q-Q segueix la línia de normalitat a la regió central però s'allunya als extrems.

```
qqnorm(test_and_train$Age, pch = 19, col = "gray50")
qqline(test_and_train$Age)
```




```
# Test de normalitat de Shapiro-Wilk
shapiro.test(test_and_train$Age)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  test_and_train$Age
## W = 0.96958, p-value = 5.777e-16
```

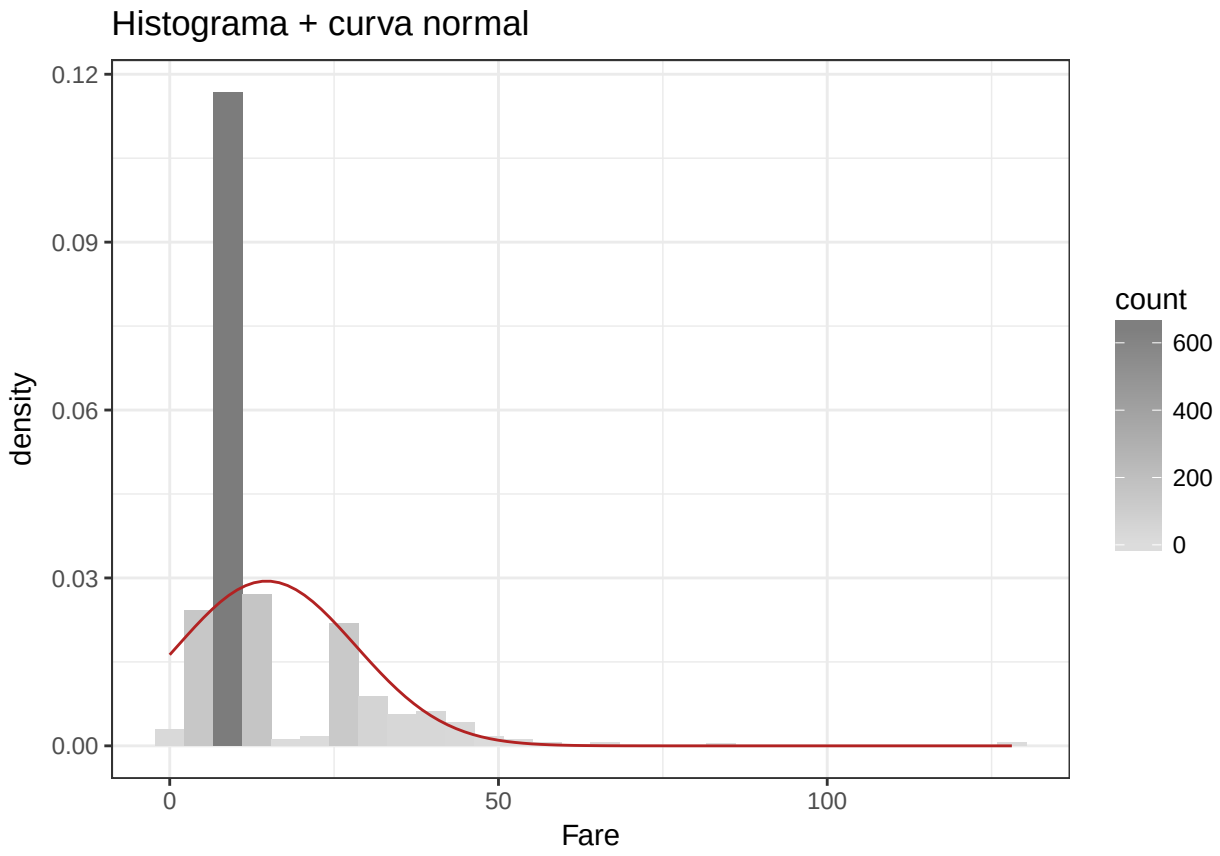
```
library(nortest)
# Test de normalitat d'Anderson-Darling
ad.test(test_and_train$Age)
```

```
##
##  Anderson-Darling normality test
##
## data:  test_and_train$Age
## A = 14.657, p-value < 2.2e-16
```

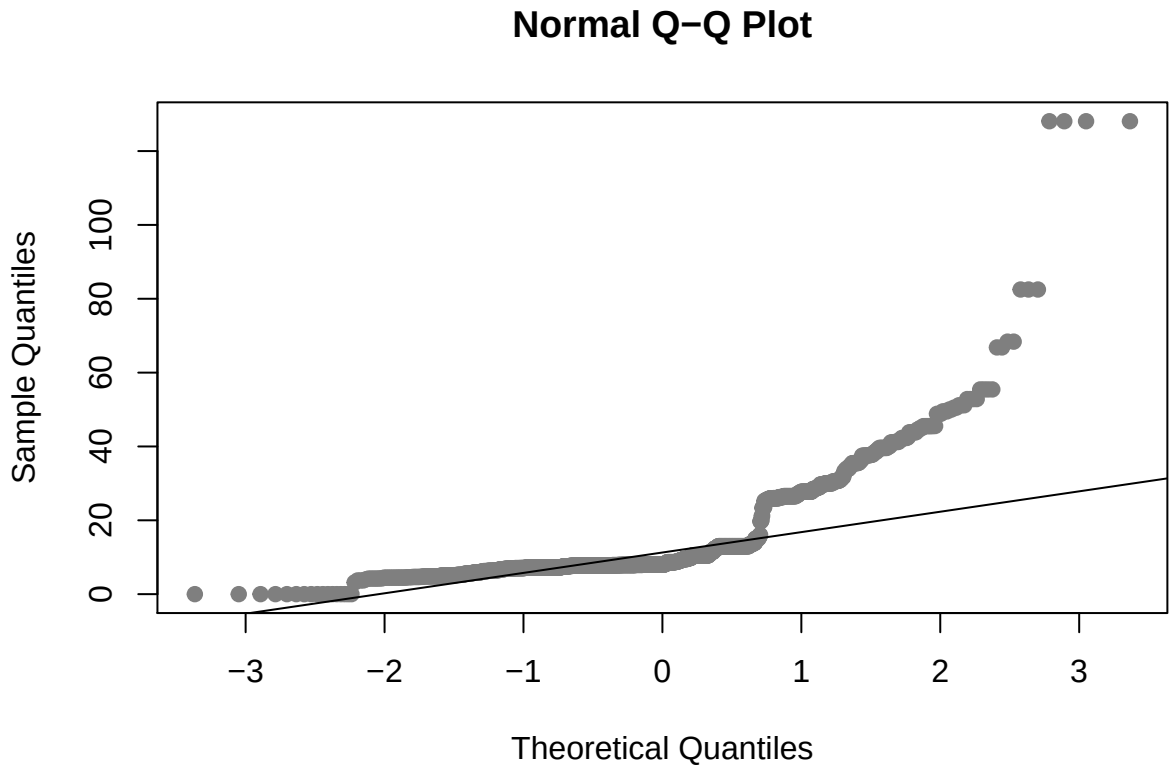
Fare

```
# Representem les dades en un histograma
ggplot(data = test_and_train, aes(x = Fare)) +
  geom_histogram(aes(y = ..density.., fill = ..count..)) +
  scale_fill_gradient(low = "#DCDCDC", high = "#7C7C7C") +
  stat_function(fun = dnorm, colour = "firebrick",
                args = list(mean = mean(test_and_train$Fare),
                            sd = sd(test_and_train$Fare))) +
  ggtitle("Histograma + curva normal") +
  theme_bw()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
qqnorm(test_and_train$Fare, pch = 19, col = "gray50")
qqline(test_and_train$Fare)
```



```r
# Test de normalitat de Shapiro-Wilk
shapiro.test(test_and_train$Fare)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  test_and_train$Fare
## W = 0.67421, p-value < 2.2e-16
```

```r
library(nortest)
# Test de normalitat d'Anderson-Darling
ad.test(test_and_train$Fare)
```

```
##
##  Anderson-Darling normality test
##
## data:  test_and_train$Fare
## A = 139.02, p-value < 2.2e-16
```

4.3. Proves estadístiques

4.3.1. Anàlisi de correlació

A la primera anàlisi ens preguntem quines variables numèriques influeixen més en la supervivència dels passatgers. Per respondre-la utilitzarem un anàlisi de correlació entre les diferents variables quantitatives per al conjunt *train*, que és l'únic que conté l'atribut *Survived*. Per mesurar el grau de correlació utilitzarem el coeficient de *Pearson*.

```r
# Correlació entre variables numèriques
cor(train[c("Survived", var_num)], method="pearson")
```

```
##             Survived      Pclass         Age       SibSp       Parch
## Survived  1.00000000 -0.33848104 -0.08515946 -0.03532250  0.08162941
## Pclass   -0.33848104  1.00000000 -0.36958135  0.08308136  0.01844267
## Age      -0.08515946 -0.36958135  1.00000000 -0.31016335 -0.20096781
## SibSp    -0.03532250  0.08308136 -0.31016335  1.00000000  0.41483770
## Parch     0.08162941  0.01844267 -0.20096781  0.41483770  1.00000000
## Fare      0.28833741 -0.76298054  0.30487621 -0.07352869 -0.03644245
## Group     0.06496221 -0.03989326 -0.27476427  0.72733097  0.63836059
##                Fare       Group
## Survived  0.28833741  0.06496221
## Pclass   -0.76298054 -0.03989326
## Age       0.30487621 -0.27476427
## SibSp    -0.07352869  0.72733097
## Parch    -0.03644245  0.63836059
## Fare      1.00000000  0.05493144
## Group     0.05493144  1.00000000
```

4.3.2. Contrast d'hipòtesis

A la segona anàlisi ens preguntem si les diferents categories de les variables qualitatives (*Sex*, *Cabin*, *Embarked* i *Title*) influeixen en la probabilitat de sobreviure. Atès que la variable *Survived* també és qualitativa, emprarem el test *Chi quadrat* amb la hipotesi nul · la que diu que les variables examinades no són independents. Primer calcularem les freqüències de *Survived* per a cada categoria i després aplicarem la funció *chisq.test*.

```r
# Test chi quadrat per a totes les variables qualitatives
for (col in var_cat){
  # Creem una taula de freqüències
  taula = table(train[, c("Survived", col)])
  print(taula)
  # Apliquem el test
  print(chisq.test(taula))
}
```

```
##          Sex
## Survived female male
##        0     81  468
##        1    233  109
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  taula
## X-squared = 260.72, df = 1, p-value < 2.2e-16
##
##          Cabin
## Survived  NO   X   B   C   D   E
##        0 481  16  12  24   8   8
##        1 206  17  35  35  25  24
##
##  Pearson's Chi-squared test
##
## data:  taula
## X-squared = 97.337, df = 5, p-value < 2.2e-16
##
##          Embarked
## Survived   S   C   Q
##        0 427  75  47
##        1 219  93  30
##
##  Pearson's Chi-squared test
##
## data:  taula
## X-squared = 25.964, df = 2, p-value = 2.301e-06
##
##          Title
## Survived Master Miss  Mr Mrs other
##        0     17   55 436  26    15
##        1     23  127  81  98    13
##
##  Pearson's Chi-squared test
##
## data:  taula
## X-squared = 282.24, df = 4, p-value < 2.2e-16
```

A L'execució podem veure com el p-valor és inferior al 0.05 en tots els casos, per tant, rebutgem la hipòtesi nul · la. Això vol dir que podem afirmar que hi ha diferències significatives entre el nombre de supervivents per a les diferents categories de *Sex*, *Cabin*, *Embarked* i *Title*.

4.3.3. Regressió logística

Finalment, calcularem la probabilitat de supervivència per als passatgers del conjunt *test*. En el nostre cas, com que la variable a predir (dependent) és dicotòmica, utilitzarem una regressió logística. Per generar el model de regressió logística utilitzarem la funció *regLog* amb totes les variables disponibles, tant les categòriques com les numèriques. Les variables numèriques no requereixen cap tipus de transformació, però les categòriques s'hauran de binaritzar, és a dir, haurem de crear un atribut de tipus binari per a cada categoria; malgrat tot, la funció *regLog* ja realitza aquesta transformació i per tant no serà necessari modificar cap atribut.

```
# Generem el model de regressió logística
regLog <- glm(Survived ~ Pclass + Age + SibSp + Parch + Fare + Group + Sex + Cabin + Embarked + Title,
              data = train, family = binomial(link="logit"))
regLog$coefficients
```

```
##  (Intercept)       Pclass          Age        SibSp        Parch
##  20.05843180  -0.87476912  -0.03753499  -0.64035704  -0.40423901
##         Fare        Group      Sexmale       CabinX       CabinB
##   0.01496888   0.09201631 -15.84479517   0.38352589   0.48750799
##       CabinC       CabinD       CabinE    EmbarkedC    EmbarkedQ
##   0.09489127   1.10598688   1.46541335   0.39266548   0.42530878
##    TitleMiss      TitleMr     TitleMrs   Titleother
## -16.21961446  -3.24855574 -15.26728882  -3.00201424
```

Els coeficients obtinguts són els paràmetres de la funció que modelitza la probabilitat de supervivència per a cada atribut. A partir d'aquests paràmetres podem fer una predicció de la probabilitat de supervivència del conjunt *test* amb la funció *predict*.

```
# Calculem la probabilitat de supervivència dels passatgers del conjunt test
prob_Survived <- predict(regLog, test, type="response")
head(cbind(test, prob_Survived),3)
```

```
##   Pclass    Sex  Age SibSp Parch   Fare Cabin Embarked Title Group
## 1      3   male 34.5     0     0 7.8292    NO        Q    Mr     1
## 2      3 female 47.0     1     0 7.0000    NO        S   Mrs     1
## 3      2   male 62.0     0     0 9.6875    NO        Q    Mr     1
##   prob_Survived
## 1    0.08950743
## 2    0.48978459
## 3    0.07949046
```

5. Representació de dades

6. Resolució del problema

7. Recursos