Informe _RMarkdown

R Markdown

Empezamos por importar el dataset

```
Data_Original <- read.csv('train.csv', stringsAsFactors = FALSE, header =
TRUE)
```

Observamos los primeros resultados del dataset con la función head()

```
head(Data_Original)
```

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

Observamos los tipos de variables que tenemos y sus valores

```
str(Data_Original)
```

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

Análisis estadístico de la distribución de los valores

```
summary(Data_Original)
```

```
##   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
##
```

Eliminamos del dataframe las columnas que no nos interesan para el análisis: Ticket, Cabin y Embarked

```
borrar <- c("Ticket","Cabin", "Embarked")
df <- Data_Original[ , !(names(Data_Original) %in% borrar)]
```

Comprobamos que la eliminación ha sido correcta

```
head(df, n=5)
```

```
##   PassengerId Survived Pclass
## 1           1        0      3
## 2           2        1      1
## 3           3        1      3
## 4           4        1      1
## 5           5        0      3
##                                                  Name    Sex Age SibSp
Parch
## 1                             Braund, Mr. Owen Harris   male  22     1
0
## 2 Cumings, Mrs. John Bradley (Florence Briggs Thayer) female  38     1
0
## 3                              Heikkinen, Miss. Laina female  26     0
0
## 4        Futrelle, Mrs. Jacques Heath (Lily May Peel) female  35     1
0
## 5                            Allen, Mr. William Henry   male  35     0
0
##      Fare
## 1  7.2500
## 2 71.2833
## 3  7.9250
## 4 53.1000
## 5  8.0500
```

Para continuar con el análisis, comprobamos la presencia de valores NA

```
colSums(is.na(Data_Original))
```

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

Vemos que tenemos valores vacíos en la variable Age; los reemplazamos por la media de la edad

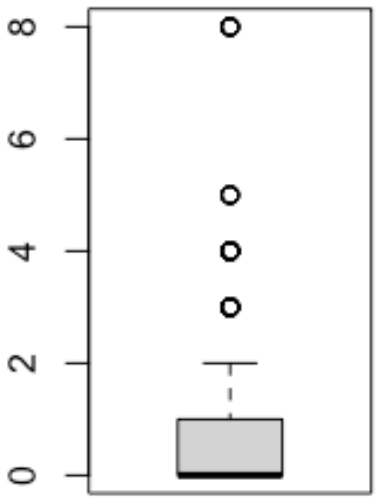
```
df$Age[is.na(df$Age)] <- mean(df$Age, na.rm=T)
```

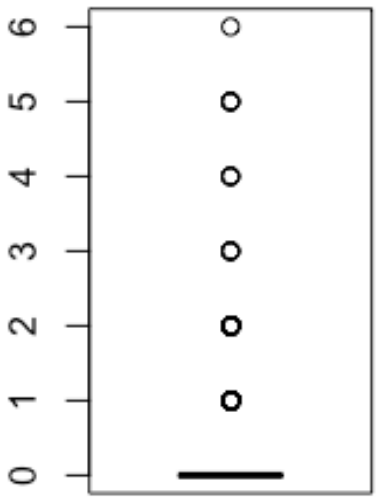
Comprobamos que no quedan valores vacios en las variables elegidas

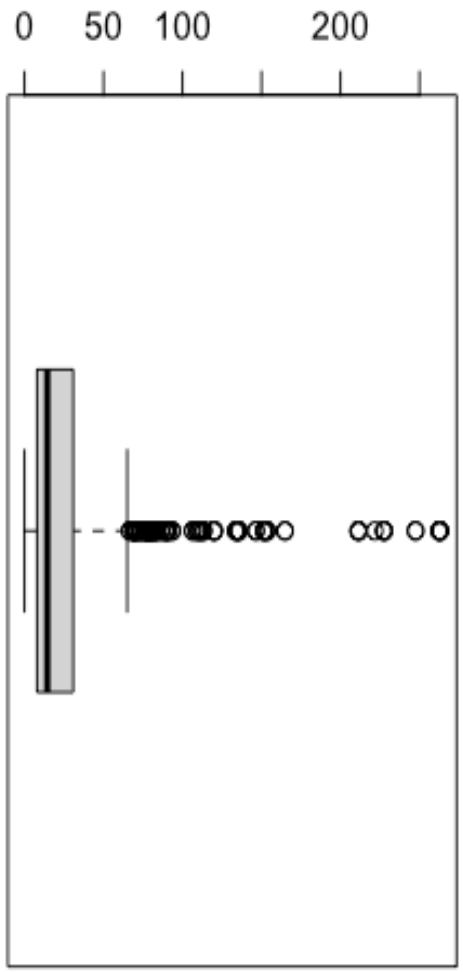
```
colSums(df=="")
```

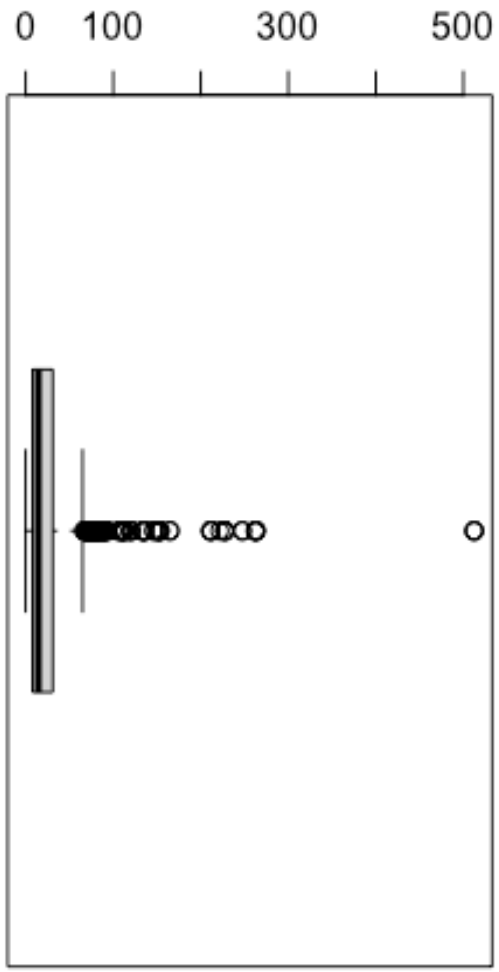
```
## PassengerId    Survived      Pclass        Name         Sex         Age
##           0           0           0           0           0           0
##       SibSp       Parch        Fare
##           0           0           0
```

```
## [1] "integer"
```






0
50
100
200


0
100
300
500

Para acabar de limpiar la variable 'Fare', redondeamos todos sus registros a dos decimales

```
df2$Fare <- round(df2$Fare, 2)
```

Comenzamos la factorización

Primero, comprobamos qué variables son las que debemos binarizar

```
apply(df2, 2, function(x) length(unique(x)))
```

```
## PassengerId    Survived      Pclass        Name         Sex         Age
##         888           2           3         888           2          71
##       SibSp       Parch        Fare
##           7           7         236
```

Y binarizamos la variable 'sex', para registrarla en vez de como 'male' y 'female', como (0,1)

```
df2 <- transform(df2, Sex = ifelse(Sex== "male", 0, 1))
```

Después de los cambios, analizamos la nueva estructura del conjunto de datos

```
str(df2)
```

```
## 'data.frame':    888 obs. of  9 variables:
##  $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Survived   : int  0 1 1 1 0 0 0 0 1 1 ...
##  $ Pclass     : int  3 1 3 1 3 3 1 3 3 2 ...
##  $ Name       : chr  "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley
(Florence Briggs Thayer)" "Heikkinen, Miss. Laina" "Futrelle, Mrs. Jacques
Heath (Lily May Peel)" ...
##  $ Sex        : num  0 1 1 1 0 0 0 0 1 1 ...
##  $ Age        : int  22 38 26 35 35 29 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 ...
##  $ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
```

El siguiente paso es crear una nueva variable ('Fam') a partir de la combinación de las variables SibSp y Parch. Esta variable nos indica si el sujeto tiene (1) o no tiene (0) familia.

```
df3 <- transform(df2, Fam = ifelse(SibSp==0 & Parch ==0, 0, 1))
```

Comprobamos que la variable 'Fam' se ha creado correctamente, tomando valor 1 si el sujeto tiene algún familiar (si SibSp o Parch es distinto de cero) y valor 0 si no tienen ninguno (SibSp y Parch son iguales a cero).

```
summary(df3$Fam)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  0.0000  0.3975  1.0000  1.0000
```

```
unique(df3$Fam[df3$SibSp==0 & df3$Parch == 0])
```

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

Habiendo comprobado que hemos creado con éxito la variable Fam, vamos a eliminar las variables 'SibSp' y 'Parch', ya que el contenido de interés analítico en este caso se encuentra recogido en la nueva variable 'Fam'.

```
borrar <- c("SibSp","Parch")
df_final <- df3[ , !(names(df3) %in% borrar)]
```

Y comprobamos que se han borrado correctamente:

```
head(df_final)
```

```
##   PassengerId Survived Pclass
## 1           1        0      3
## 2           2        1      1
## 3           3        1      3
## 4           4        1      1
## 5           5        0      3
## 6           6        0      3
##                                                 Name Sex Age  Fare Fam
## 1                            Braund, Mr. Owen Harris   0  22  7.25   1
## 2 Cumings, Mrs. John Bradley (Florence Briggs Thayer)   1  38 71.28   1
## 3                             Heikkinen, Miss. Laina   1  26  7.92   0
## 4        Futrelle, Mrs. Jacques Heath (Lily May Peel)   1  35 53.10   1
## 5                           Allen, Mr. William Henry   0  35  8.05   0
## 6                                   Moran, Mr. James   0  29  8.46   0
```

Antes de exportar nuestros datos, vamos a factorizar las variables que lo requieran para evitar futuros conflictos.

```
str(df_final)
```

```
## 'data.frame':    888 obs. of  8 variables:
##  $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Survived   : int  0 1 1 1 0 0 0 0 1 1 ...
##  $ Pclass     : int  3 1 3 1 3 3 1 3 3 2 ...
##  $ Name       : chr  "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley
```

```
(Florence Briggs Thayer)" "Heikkinen, Miss. Laina" "Futrelle, Mrs. Jacques
Heath (Lily May Peel)" ...
##  $ Sex         : num  0 1 1 1 0 0 0 0 1 1 ...
##  $ Age         : int  22 38 26 35 35 29 54 2 27 14 ...
##  $ Fare        : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Fam         : num  1 1 0 1 0 0 0 1 1 1 ...
```

Vemos que las variables que no son leídas como factores son 'Survived', 'Pclass', 'Sex' y 'Fam', las factorizamos y nuestra base de datos queda lista para el análisis

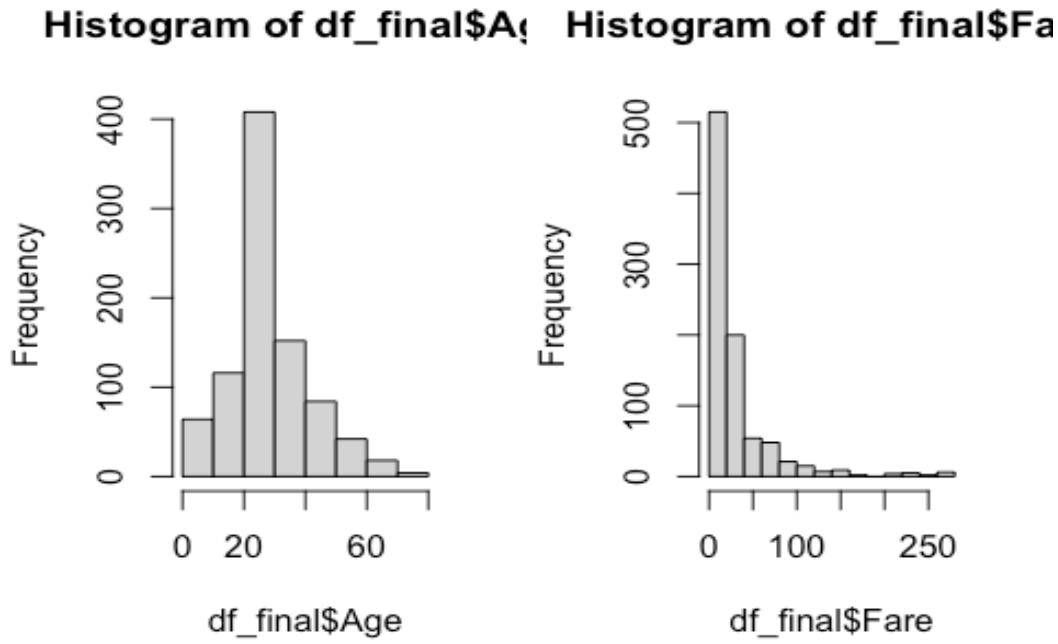
```
cols<-c("Survived", "Pclass", "Sex", "Fam")
for (i in cols){
  df_final[,i] <- as.factor(df_final[,i])
}

str(df_final)
```

```
## 'data.frame':    888 obs. of  8 variables:
##  $ PassengerId: int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Survived   : Factor w/ 2 levels "0","1": 1 2 2 2 1 1 1 1 2 2 ...
##  $ Pclass     : Factor w/ 3 levels "1","2","3": 3 1 3 1 3 3 1 3 3 2 ...
##  $ Name       : chr  "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley
(Florence Briggs Thayer)" "Heikkinen, Miss. Laina" "Futrelle, Mrs. Jacques
Heath (Lily May Peel)" ...
##  $ Sex        : Factor w/ 2 levels "0","1": 1 2 2 2 1 1 1 1 2 2 ...
##  $ Age        : int  22 38 26 35 35 29 54 2 27 14 ...
##  $ Fare       : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Fam        : Factor w/ 2 levels "0","1": 2 2 1 2 1 1 1 2 2 2 ...
```

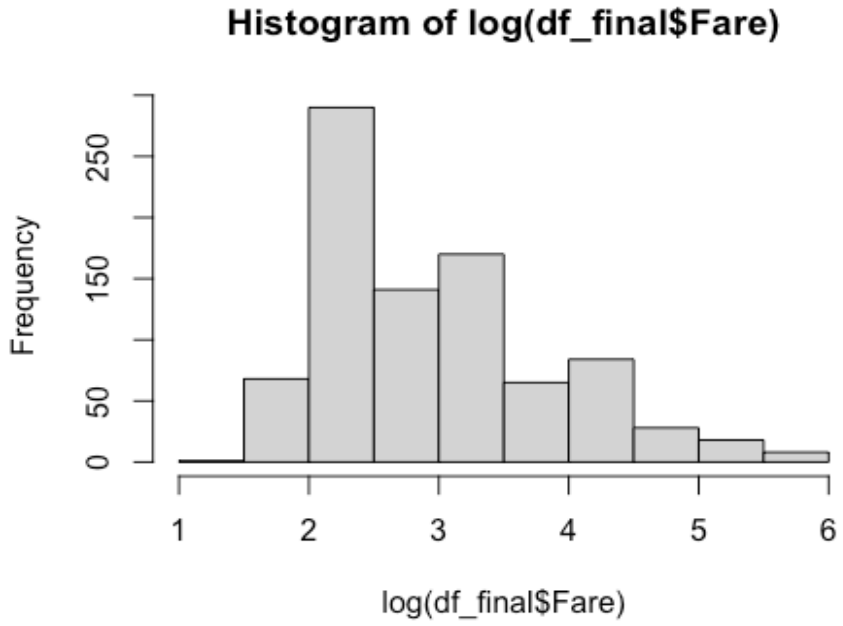
```
write.csv(df_final, "Data Final.csv", row.names=FALSE)
```

Antes de comenzar con el análisis, vamos a observar cómo se comportan las variables finales con las que trabajaremos. Usaremos primero histogramas para hacernos una primera idea.

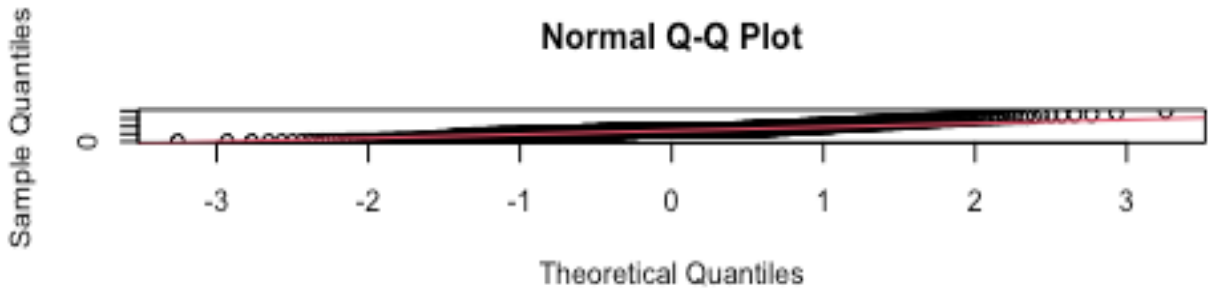
```
par(mfrow=c(1,2))
hist(df_final$Age)
hist(df_final$Fare)
```

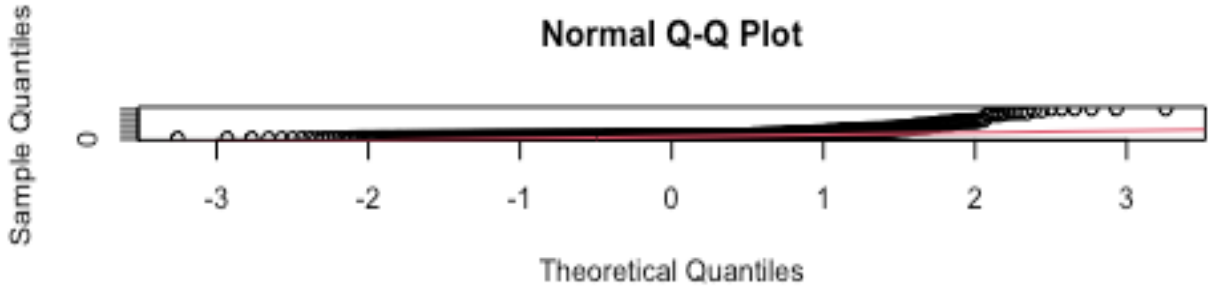



```
De la representación anterior, vemos que la variable 'Fare' (precio del billete) requeriría de una escala logarítmica para poder observar mejor cómo se comportan sus registros.

hist(log(df_final$Fare))
```



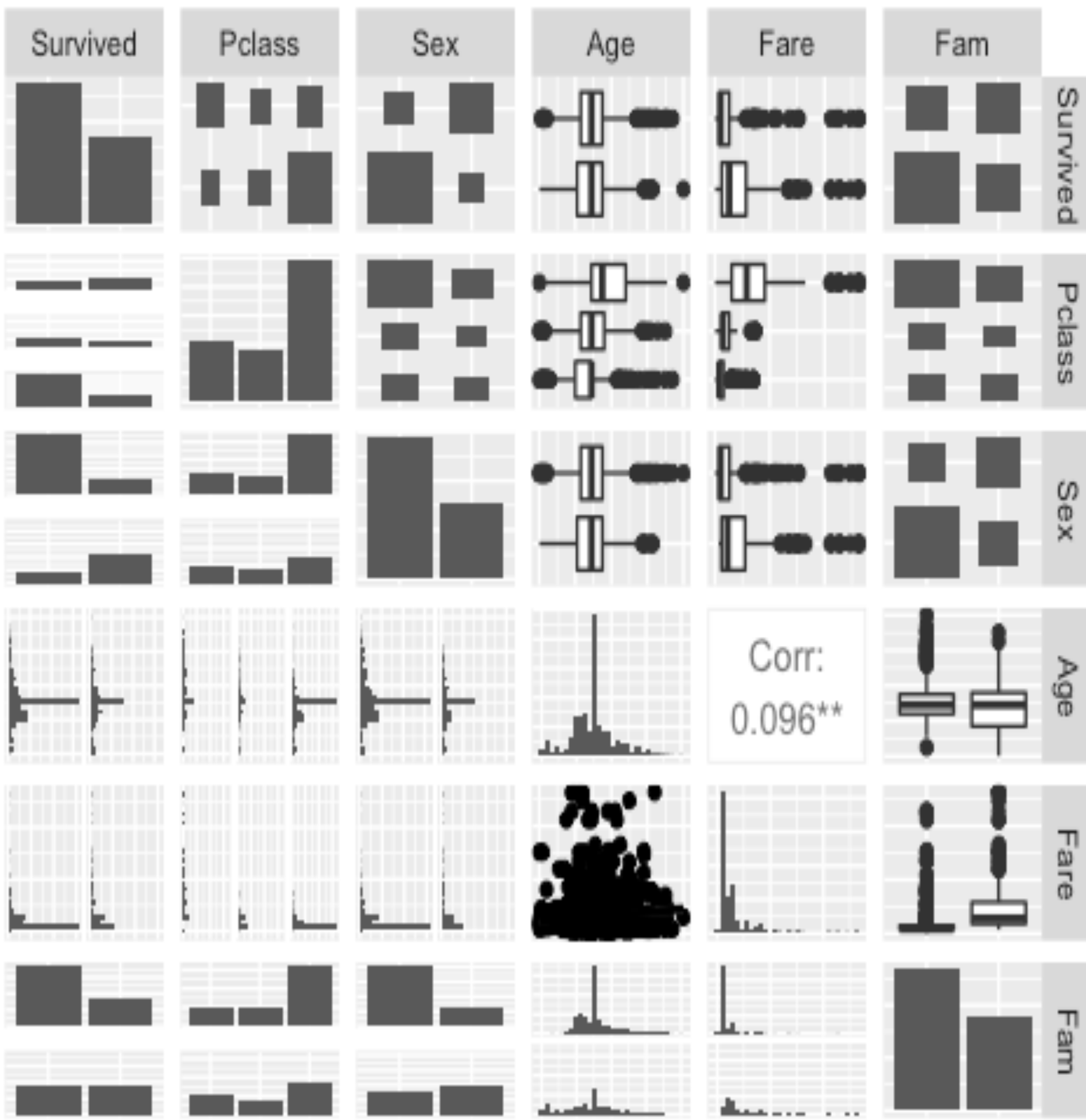
```
Hacemos una primera representación de las correlaciones entre variables para
ir haciéndonos una idea de cómo se comportan los registros que tenemos.

vars <- df_final[, c("Survived", "Pclass", "Sex", "Age", "Fare", "Fam")]



Análisis de la normalidad de Age


par(mfrow=c(3,1))
qqnorm(df_final$Age);qqline(df_final$Age,col=2)
qqnorm(df_final$Fare);qqline(df_final$Fare,col=2)
```






```
# install.packages("GGally")

library(GGally)

## Loading required package: ggplot2

## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2

library(ggplot2)

ggpairs(vars, lower = list(continuous="smooth"), diag =
list(continuous="barDiag"), axisLabels = "none")
```

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



Graficamos algunas variables en función de survived, para observar tendencias en las relaciones y guiarnos para la construcción de nuestros análisis

```
library(ggplot2)
# install.packages("dplyr")
library(dplyr)
```

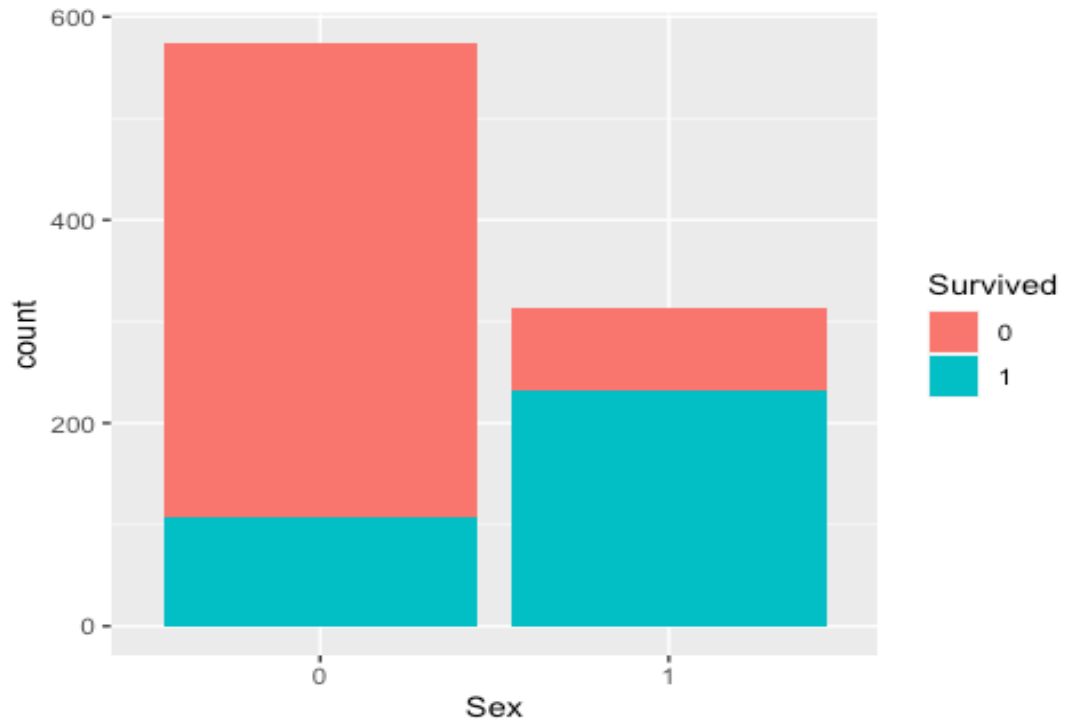
```
##
## Attaching package: 'dplyr'

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

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

```
filas=dim(df_final)
```

Empezamos por ver la relación entre el género y la supervivencia (Sex-Survival)

```
ggplot(data=df_final[1:filas,],aes(x=Sex,fill=Survived))+geom_bar()
```

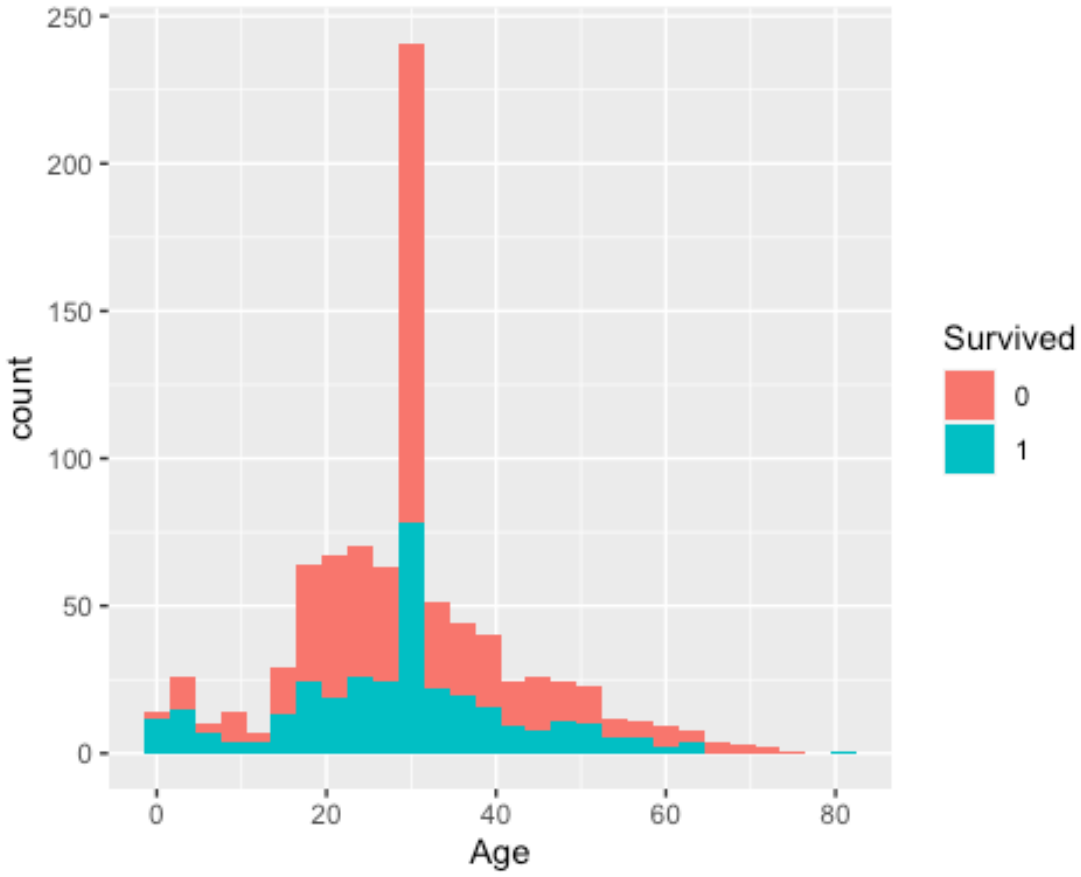
```
## Warning in 1:filas: numerical expression has 2 elements: only the first
used
```



Y graficamos también la supervivencia en función de la edad

```
ggplot(data =
df_final[!(is.na(df_final[1:filas,]$Age)),],aes(x=Age,fill=Survived))+geom_hi
stogram(binwidth =3)
```

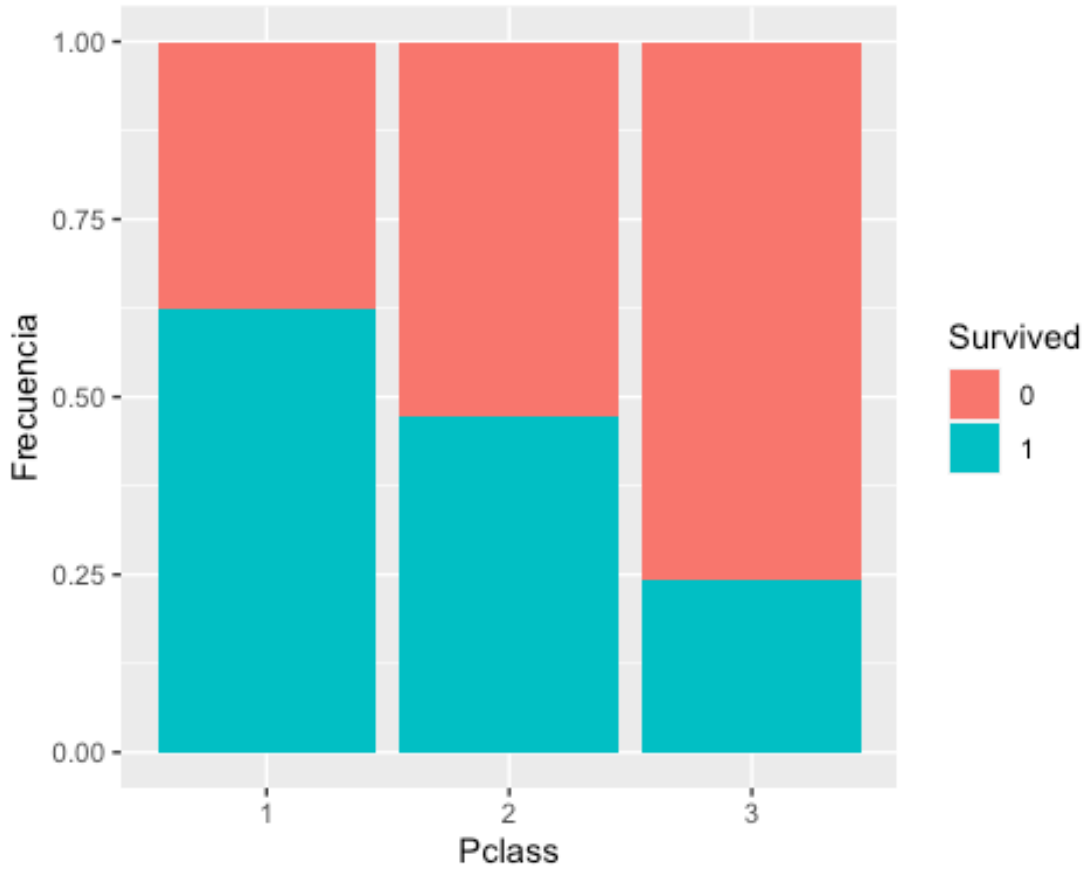
```
## Warning in 1:filas: numerical expression has 2 elements: only the first
used
```



Probamos a continuación a graficar Survival en función de Pclass

```
ggplot(data =
df_final[1:filas,],aes(x=Pclass,fill=Survived))+geom_bar(position="fill")+yla
b("Frecuencia")
```

```
## Warning in 1:filas: numerical expression has 2 elements: only the first
used
```



Obtenemos, además, una matriz de porcentajes de freccuencia. Vemos que la probabilidad de sobrevivir si el billete es de primera clase es de 62.44%, si es de segunda clase 47.28%, y si es de tercera clase 24,24%.

```
t<-table(df_final[1:filas,]$Pclass, df_final[1:filas,]$Survived)
```

```
## Warning in 1:filas: numerical expression has 2 elements: only the first
used
```

```
## Warning in 1:filas: numerical expression has 2 elements: only the first
used
```

```
for (i in 1:dim(t)[1]){
    t[i,]<-t[i,]/sum(t[i,])*100
}
```

```
t
```

```
##
##            0        1
##   1 37.55869 62.44131
```

```
##   2 52.71739 47.28261
##   3 75.76375 24.23625
```

Comenzamos con el análisis de nuestro dataset, compuesto de tres partes:

1 - Correlación + regresión simple: ¿Existe una correlación entre pagar más por el billete y aumentar tu probabilidad de sobrevivir? ¿Cuánto afecta el precio del billete a la probabilidad de sobrevivir?

2 - Regresión: ¿Afectaron las características identitarias de una persona (su edad, género, clase y si tiene o no familia) a su probabilidad de sobrevivir?

3 - Métodos de asociación. Dadas las diferentes combinaciones de valores para cada registro, ¿es posible establecer unas reglas en forma de condicional que representen la probabilidad de diferentes darse determinadas combinaciones de eventos?

CORRELACIÓN

Cuando uno paga más por un billete de primera, puede pensar que paga no solo por lujo sino por un aumeno de la seguridad. Comprobamos si efectivamente existe una correlación entre el precio del billete (Fare) y la supervivencia (Survived)

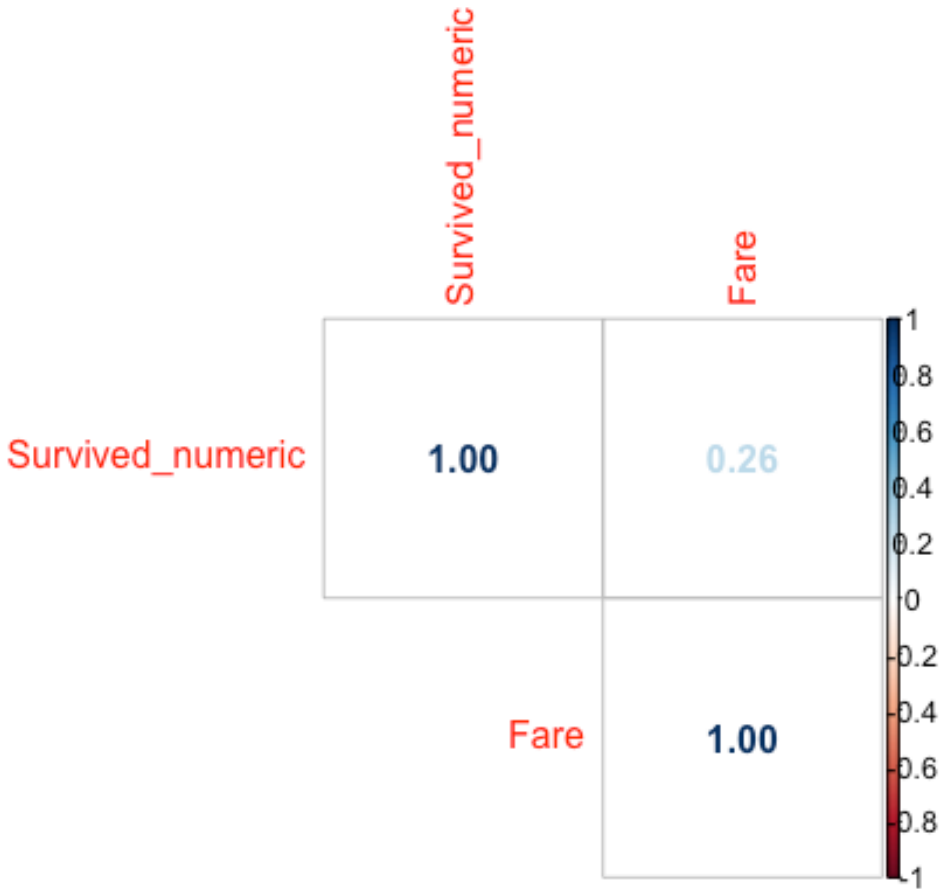
Con este fin, registramos la variable Survived como numérica

```
df_final$Survived_numeric <- as.numeric(df_final$Survived)
```

```
var.cor <- select(df_final, Survived_numeric, Fare)
correlacion1 <- cor(var.cor, method = "pearson")
# install.packages("corrplot")
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
corrplot(correlacion1, method = "number", type = "upper")
```



Realizamos la regresión simple que evalúa de forma más precisa de qué manera interactúan estas dos variables

```
regresion_simple <- glm(formula = Survived~Fare, data = df_final, family =
binomial)
summary(regresion_simple)
```

```
##
## Call:
## glm(formula = Survived ~ Fare, family = binomial, data = df_final)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4899  -0.8885  -0.8531   1.3458   1.5941
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.941125   0.095192  -9.887  < 2e-16 ***
## Fare         0.015189   0.002236   6.794 1.09e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1180.9  on 887  degrees of freedom
## Residual deviance: 1117.6  on 886  degrees of freedom
## AIC: 1121.6
##
## Number of Fisher Scoring iterations: 4
```

```
# install.packages("pROC")
```

```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

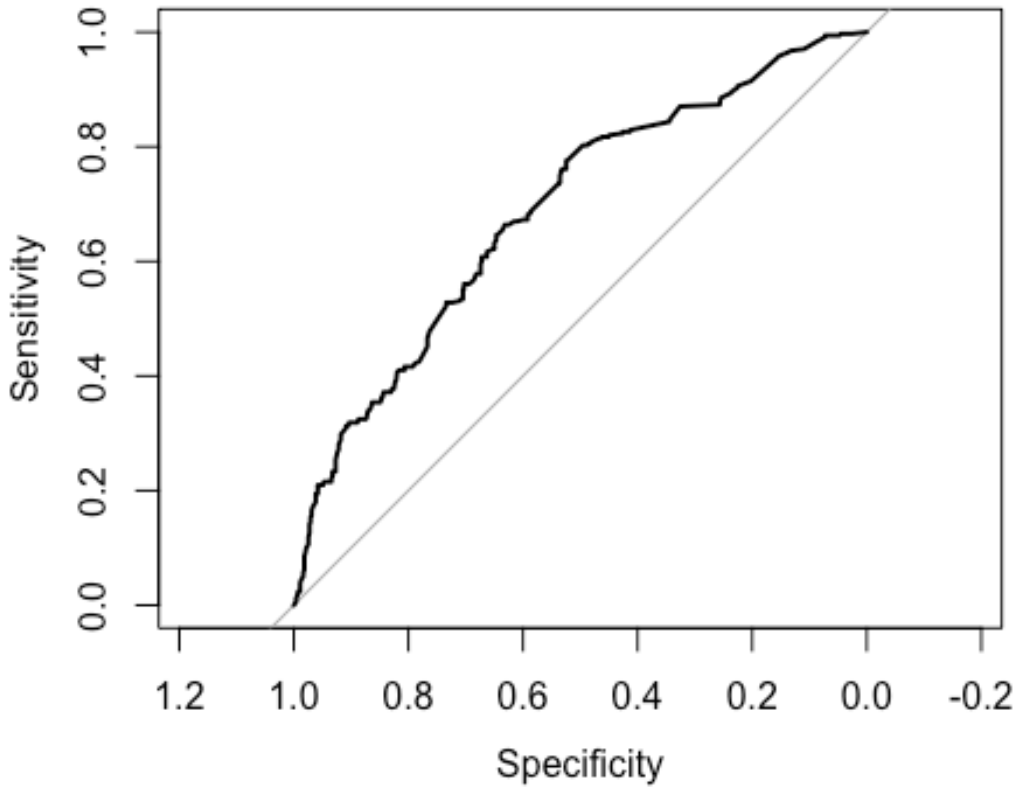
```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
probabilidad_bajo=predict(regresion_simple, df_final, type="response")
r=roc(df_final$Survived, probabilidad_bajo, data = df_final)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
plot(r)
```




```
auc(r)
```

```
## Area under the curve: 0.6894
```

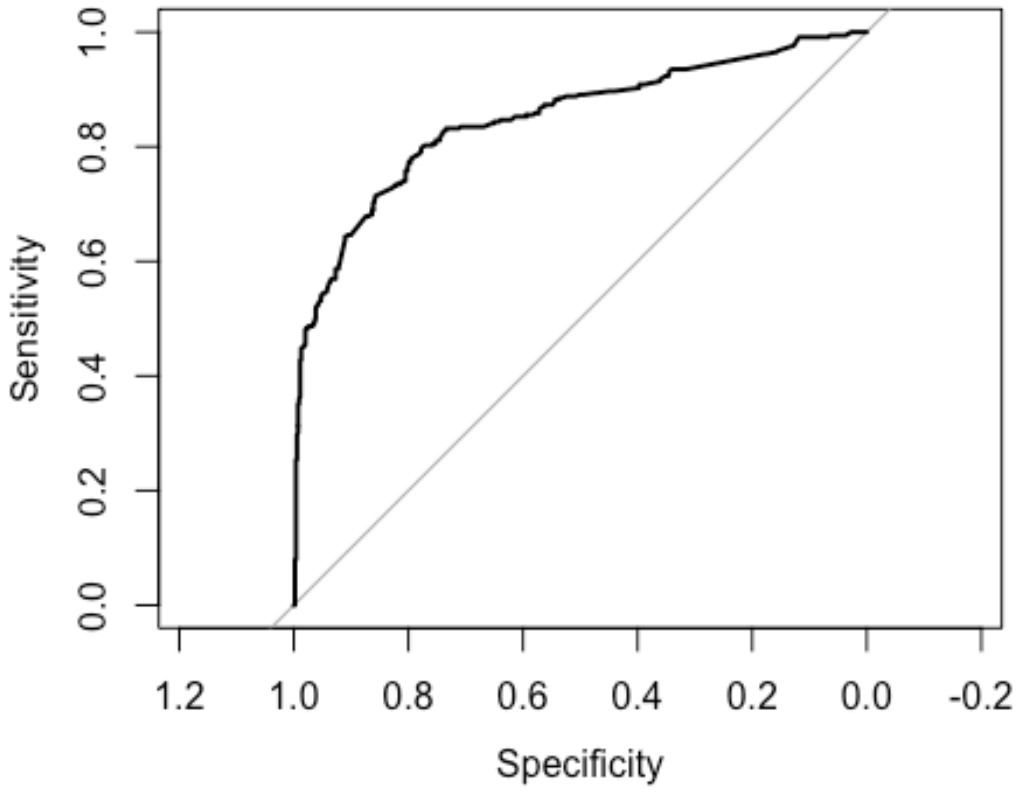
Como el área bajo la curva está entre 0.6 y 0.8, podemos considerar que el modelo discrimina de manera adecuada.

PARTE 2 - Realizamos una regresión logística múltiple para ver de qué manera los rasgos identitarios de la persona afectan a sus probabilidades de supervivencia. Lo evaluamos con las variables: sex, age, pclass (proxy de la clase social) y Fam.

```
regresion_multiple <- glm(formula = Survived~Age+Sex+Pclass+Fam, data =
df_final, family = binomial)
summary(regresion_multiple)
```

```
##
## Call:
## glm(formula = Survived ~ Age + Sex + Pclass + Fam, family = binomial,
##     data = df_final)
##
## Deviance Residuals:
```

```
##     Min       1Q   Median       3Q      Max
## -2.6326  -0.6498  -0.4265   0.6237   2.4271
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.939319   0.352065   2.668  0.00763 **
## Age         -0.033740   0.007533  -4.479 7.50e-06 ***
## Sex1         2.639080   0.194296  13.583  < 2e-16 ***
## Pclass2     -1.095095   0.259563  -4.219 2.45e-05 ***
## Pclass3     -2.312422   0.244644  -9.452  < 2e-16 ***
## Fam1        -0.077492   0.188340  -0.411  0.68074
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1180.89  on 887  degrees of freedom
## Residual deviance:  801.61  on 882  degrees of freedom
## AIC: 813.61
##
## Number of Fisher Scoring iterations: 5

library(pROC)

probabilidad_bajo=predict(regresion_multiple, df_final, type="response")
r=roc(df_final$Survived, probabilidad_bajo, data = df_final)

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

plot(r)
```




```
auc(r)

## Area under the curve: 0.8479
```

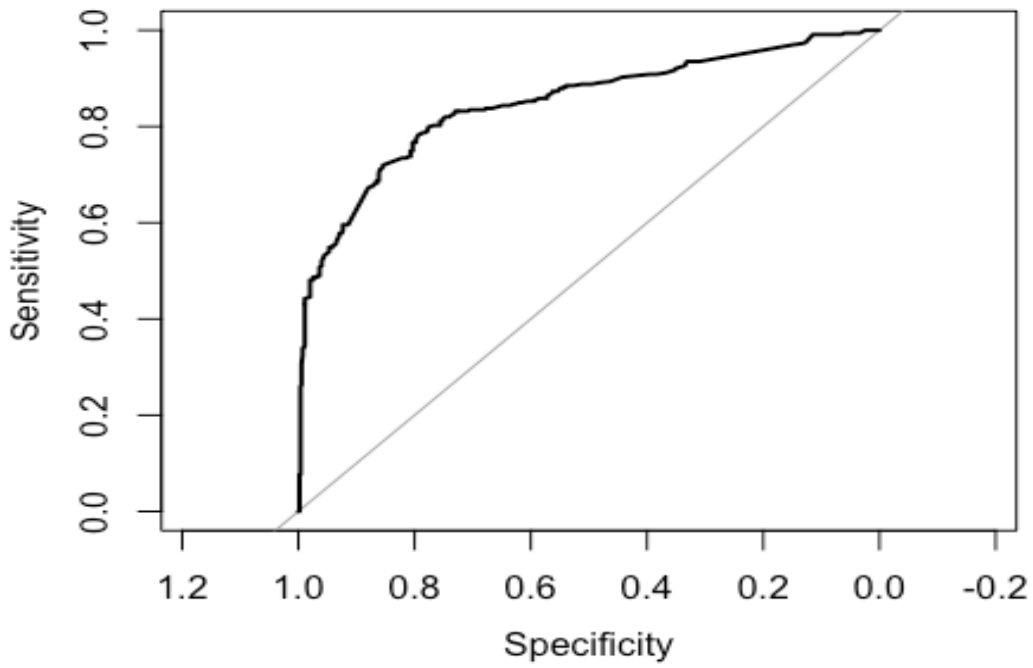
Como el área bajo la curva está entre 0.8 y 0.9, se estima que el modelo discrimina de forma excelente.

Vemos que la variable que no afecta de forma significativa es Fam, por lo tanto vamos a realizar de nuevo la regresión sin considerar esta variable para ver si podemos ajustar aún más nuestros resultados

```
regresion_multiple2 <- glm(formula = Survived~Age+Sex+Pclass, data =
df_final, family = binomial)
summary(regresion_multiple2)

##
## Call:
## glm(formula = Survived ~ Age + Sex + Pclass, family = binomial,
##     data = df_final)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6342  -0.6666  -0.4225   0.6282   2.4309
##
```

```
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.886501   0.327553   2.706   0.0068 **
## Age         -0.033123   0.007373  -4.493 7.04e-06 ***
## Sex1         2.617564   0.186685  14.021  < 2e-16 ***
## Pclass2     -1.085267   0.258301  -4.202 2.65e-05 ***
## Pclass3     -2.297093   0.241673  -9.505  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1180.89  on 887  degrees of freedom
## Residual deviance:  801.78  on 883  degrees of freedom
## AIC: 811.78
##
## Number of Fisher Scoring iterations: 5
```

```
library(pROC)

probabilidad_bajo=predict(regresion_multiple2, df_final, type="response")
r=roc(df_final$Survived, probabilidad_bajo, data = df_final)

## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

plot(r)
```



```
auc(r)
```

```
## Area under the curve: 0.8474
```

Vemos que el modelo no mejora significativamente

PARTE 3 MÉTODOS DE ASOCIACIÓN

En esta tercera parte vamos a obtener reglas de asociacion a partir de una selección de variables categóricas del dataset.
Dichas reglas nos ayudarán a comprender cómo la información del data set se relaciona entre si.

```
#install.packages("arules")
library(arules)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'arules'
```

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

```
## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

```
df_final1<- select(df_final, "Fam", "Sex", "Pclass", "Survived")
titanic_rules <- apriori(df_final1, parameter = list(support = 0.01,
confidence = 0.5))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.5    0.1    1 none FALSE            TRUE       5    0.01      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 8
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[9 item(s), 888 transaction(s)] done [0.00s].
## sorting and recoding items ... [9 item(s)] done [0.00s].
```

```
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [122 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
inspect(head(sort(titanic_rules, by = "confidence"), 5))
```

```
##     lhs                                rhs          support    confidence
## [1] {Fam=0, Pclass=1, Survived=0} => {Sex=0}      0.05630631 0.9803922
## [2] {Fam=0, Sex=1, Pclass=1}      => {Survived=1} 0.03603604 0.9696970
## [3] {Sex=1, Pclass=1}             => {Survived=1} 0.10135135 0.9677419
## [4] {Fam=1, Sex=1, Pclass=1}      => {Survived=1} 0.06531532 0.9666667
## [5] {Pclass=1, Survived=0}        => {Sex=0}      0.08671171 0.9625000
##     coverage   lift     count
## [1] 0.05743243 1.514066 50
## [2] 0.03716216 2.540091 32
## [3] 0.10472973 2.534970 90
## [4] 0.06756757 2.532153 58
## [5] 0.09009009 1.486435 77
```

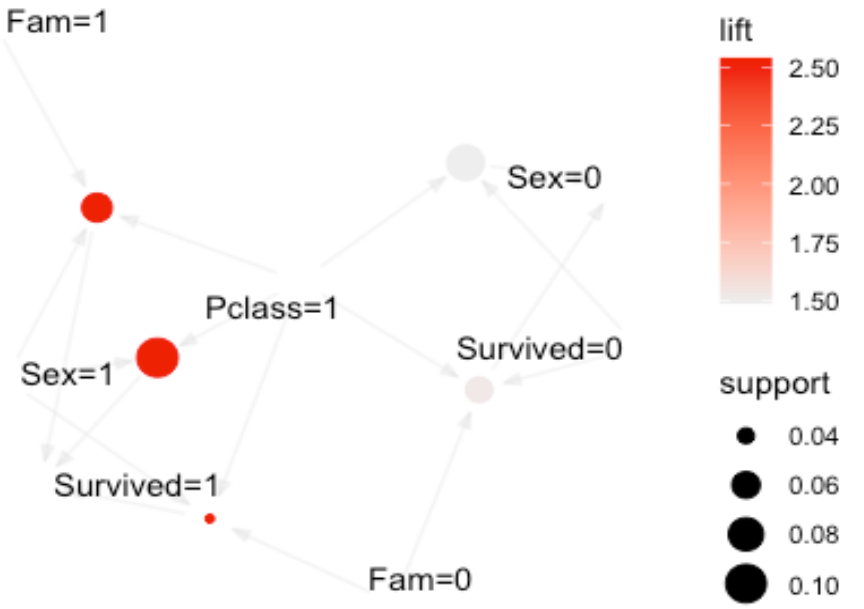
Como vemos se ha generado un set de reglas con diferente soporte, confianza y lift. El soporte indica cuantas veces se han encontrado las reglas {lsh => rhs} en el dataset, cuanto más alto mejor. La confianza habla de la probabilidad de que {rhs} se de en función de {lhs}. El lift es un parámetro que nos indica cuánto de aleatoriedad hay en las reglas. Un lift de 1 o menos es que las reglas son completamente fruto del azar.

Representación visual

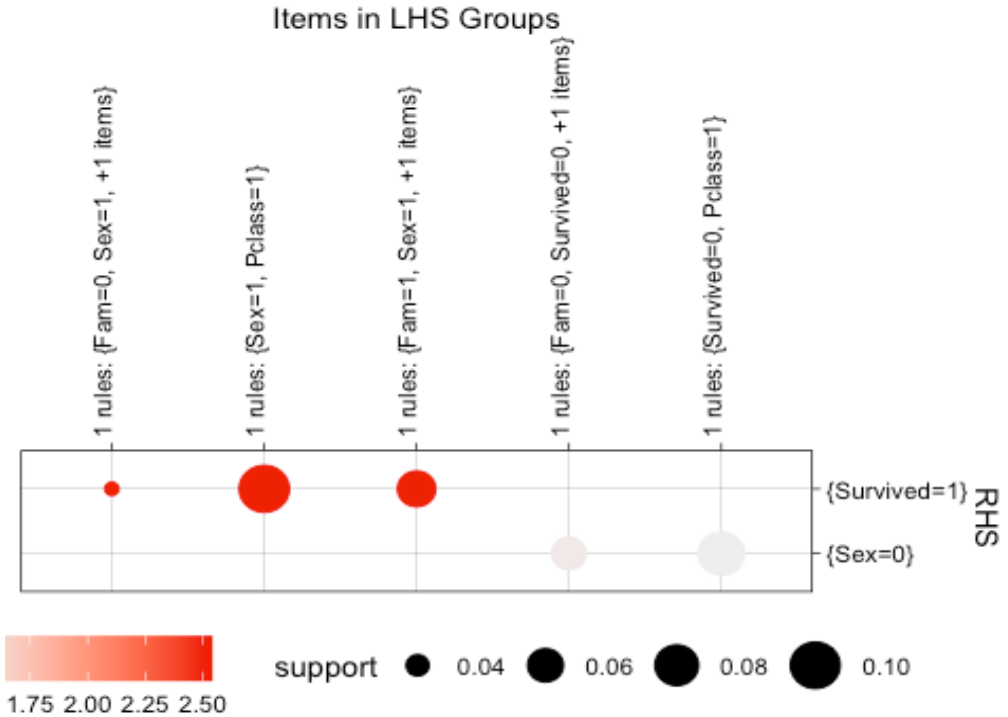
```
# install.packages("arulesViz")
```

```
library(arulesViz)
```

```
Creamos un subconjunto de 5 reglas de entre todas las reglas posibles.
subrules <- head(sort(titanic_rules, by = "confidence"), 5)
plot(subrules, method="graph")
```




```
plot(subrules, method = "grouped", control = list(k = 5))
```




```
plot(subrules, method="paracoord",  control=list(alpha=.5, reorder=FALSE))
```


Parallel coordinates plot for 5 rules
Survived=1
Sex=0
Sex=1
Fam=1
Fam=0
Survived=0
Pclass=1
3
2
1
rhs
Position