Práctica 2: Limpieza y validación de los datos

Esteban Bordallo Valbuena

Diciembre 2018

Índice general

1 Introducción

En esta práctica se elabora un caso práctico orientado a aprender a identificar los datos relevantes para un proyecto analítico y usar las herramientas de integración, limpieza, validación y análisis de las mismas.

1.1 Competencias

En esta práctica se desarrollan las siguientes competencias del Máster de Data Science:

- Capacidad de analizar un problema en el nivel de abstracción adecuado a cada situación y aplicar las habilidades y conocimientos adquiridos para abordarlo y resolverlo.
- Capacidad para aplicar las técnicas específicas de tratamiento de datos (integración, transformación, limpieza y validación) para su posterior análisis.

1.2 Objetivos

Los objetivos concretos de esta práctica son:

- Aprender a aplicar los conocimientos adquiridos y su capacidad de resolución de problemas en entornos nuevos o poco conocidos dentro de contextos más amplios o multidisciplinares.
- Saber identificar los datos relevantes y los tratamientos necesarios (integración, limpieza y validación) para llevar a cabo un proyecto analítico.
- Aprender a analizar los datos adecuadamente para abordar la información contenida en los datos.
- Identificar la mejor representación de los resultados para aportar conclusiones sobre el problema planteado en el proceso analítico.
- Actuar con los principios éticos y legales relacionados con la manipulación de datos en función del ámbito de aplicación.
- Desarrollar las habilidades de aprendizaje que les permitan continuar estudiando de un modo que tendrá que ser en gran medida autodirigido o autónomo.
- Desarrollar la capacidad de búsqueda, gestión y uso de información y recursos en el ámbito de la ciencia de datos.

2 Desarrollo

2.1 Descripción del dataset

El conjunto de datos escogido es el del **Titanic: Machine Learning from Disaster**, obtenidos desde este enlace(https://www.kaggle.com/c/titanic). Este dataset pertenece a una competición de Kaggel y contiene un listado de 891 pasajeros del titanic con 12 características o variables.

La descripción de las características es la siguiente:

- PassengerId: número identificador de cada pasajero
- Survived: supervivencia o no al hundimiento.(0 = No, 1 = Si)
- Pclass: tipo de pasaje (1 = 1st, 2 = 2nd, 3 = 3rd)
- Name: nombre del pasajero.
- género (male = masculino, female = femenino).

- Age: edad del pasajero.
- SibSp: número de hermanos/esposas que cada pasajero tenía en el barco.
- Parch: número de padres/hijos que cada pasajero tenía en el barco.
- Ticket: número del ticket.
- Fare: tarifa del pasaje
- Cabin: número de cabina
- Embarked: puerto de embarque (C = Cherbourg, S = Southampton, Q = Queenstown)

El objetivo es conseguir adivinar, mediante el análisis de las características de los pasajeros , si estos sobrevivieron o no. Por lo tanto descubriremos que características fueron determinantes en la supervivencia de los pasajeros del Titanic ¡¡¡Mujeres y niños primero!!! ¿O tuvo influencia la clase social?

2.2 Integración y selección de los datos de interés a analizar

En primer lugar cargaremos el fichero de datos CSV en el objeto `data.frame` train:

```
# Lectura de datos
Titanic = read.csv("train.csv", header = TRUE)
head(Titanic)
```

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

```
# Tipo de dato asignado a cada campo
kable(data.frame(variables=names(sapply(Titanic, class)),
                 clase=as.vector(sapply(Titanic, class))))
```

variables	clase
PassengerId	integer
Survived	integer
Pclass	integer
Name	factor
Sex	factor
Age	numeric
SibSp	integer
Parch	integer
Ticket	factor
Fare	numeric
Cabin	factor
Embarked	factor

Las características PassengerId, y Ticket, no creo que aporten mucho al análisis que estamos realizando, por lo tanto las elimino del conjunto de datos.

```
# Suprimir PassengerId y Ticket
Tit_mod=Titanic[-c(1,9)]
```

Sin embargo voy a crear nuevas variables derivadas de las que ya tenemos. La variable **Title** agrupa los distintos títulos de tratamiento en cuatro categorías, Mr, Mrs, Miss y Master. **HasCabin** distingue entre los pasajeros con cabina y los que no tenían. **Deck** separa las cubiertas donde están situadas las cabinas. **Fam** contiene el tamaño de la familia mediante la suma de las variables SibSp y Parch, le sumamos uno para contar también al pasajero. **IsAlone** distingue entre los pasajeros que viajan solos (sin familia) de los que tienen familia.

```
# Creación de las variables Title, HasCabin, Deck, Fam e IsAlone
Mr = paste(c('Don.','Major.','Capt.','Jonkheer.','Rev.','Col.','Mr.'),
           collapse="|")
Mrs = paste(c('Countess', 'Mme.', 'Mrs.'),collapse="|")
Miss = paste(c('Mlle.', 'Ms.', 'Miss'),collapse="|")

Tit_mod$Title=as.factor(case_when(str_detect(Tit_mod$Name, Mrs) ~ 'Mrs',
                        str_detect(Tit_mod$Name, Miss) ~ 'Miss',
                        str_detect(Tit_mod$Name, Mr) ~ 'Mr',
                        str_detect(Tit_mod$Name, 'Master.') ~ 'Master',
                        str_detect(Tit_mod$Name, 'Dr.') &
                            Tit_mod$Sex == 'male'  ~ 'Mr',
                        str_detect(Tit_mod$Name, 'Dr.') &
                            Tit_mod$Sex == 'female'  ~ 'Mrs'))

Tit_mod$Deck=as.factor(case_when(str_detect(Tit_mod$Cabin, 'A') ~ 'A',
                                 str_detect(Tit_mod$Cabin, 'B') ~ 'B',
                                 str_detect(Tit_mod$Cabin, 'C') ~ 'C',
```

```
                                    str_detect(Tit_mod$Cabin, 'D') ~ 'D',
                                    str_detect(Tit_mod$Cabin, 'E') ~ 'E',
                                    str_detect(Tit_mod$Cabin, 'F') ~ 'F',
                                    str_detect(Tit_mod$Cabin, 'G') ~ 'G',
                                    TRUE ~ 'Z'))


Tit_mod$HasCabin=ifelse(Tit_mod$Cabin == "", 0, 1)

Tit_mod$Fam=(Tit_mod$SibSp+Tit_mod$Parch+1)

Tit_mod$IsAlone=ifelse(Tit_mod$Fam == 1, 1, 0)
```

Elimino las variables Name y Cabin pues ya no las usaré en el análisis.

```
# Supresión de Name y Cabin
Tit_mod=Tit_mod[-c(3,9)]
summary(Tit_mod)
```

```
##     Survived          Pclass          Sex           Age
##  Min.   :0.0000   Min.   :1.000   female:314   Min.   : 0.42
##  1st Qu.:0.0000   1st Qu.:2.000   male  :577   1st Qu.:20.12
##  Median :0.0000   Median :3.000                Median :28.00
##  Mean   :0.3838   Mean   :2.309                Mean   :29.70
##  3rd Qu.:1.0000   3rd Qu.:3.000                3rd Qu.:38.00
##  Max.   :1.0000   Max.   :3.000                Max.   :80.00
##                                                NA's   :177
##      SibSp           Parch             Fare        Embarked    Title
##  Min.   :0.000   Min.   :0.0000   Min.   :  0.00    :  2    Master: 40
##  1st Qu.:0.000   1st Qu.:0.0000   1st Qu.:  7.91   C:168    Miss  :182
##  Median :0.000   Median :0.0000   Median : 14.45   Q: 77    Mr    :537
##  Mean   :0.523   Mean   :0.3816   Mean   : 32.20   S:644    Mrs   :132
##  3rd Qu.:1.000   3rd Qu.:0.0000   3rd Qu.: 31.00
##  Max.   :8.000   Max.   :6.0000   Max.   :512.33
##
##       Deck        HasCabin          Fam            IsAlone
##  Z      :688   Min.   :0.000   Min.   : 1.000   Min.   :0.0000
##  C      : 59   1st Qu.:0.000   1st Qu.: 1.000   1st Qu.:0.0000
##  B      : 47   Median :0.000   Median : 1.000   Median :1.0000
##  D      : 33   Mean   :0.229   Mean   : 1.905   Mean   :0.6027
##  E      : 33   3rd Qu.:0.000   3rd Qu.: 2.000   3rd Qu.:1.0000
##  A      : 15   Max.   :1.000   Max.   :11.000   Max.   :1.0000
##  (Other): 16
```

2.3 Limpieza de los datos

2.3.1 Tratamiento de ceros y nulos

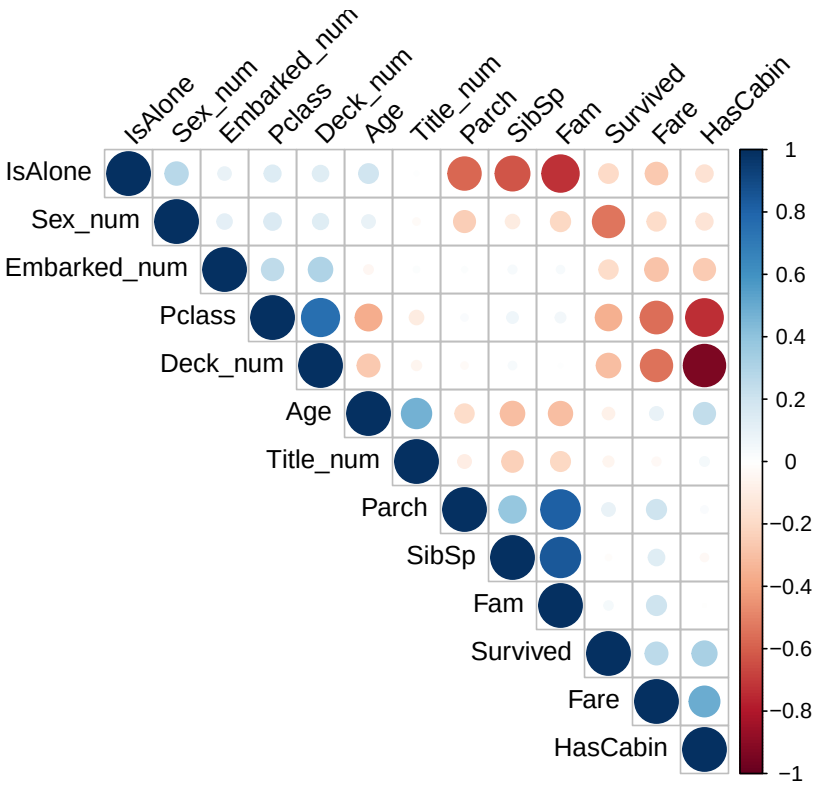
2.3.1.1 Tratamiento de NA

Como podemos ver, la falta de valores afecta solo a la variable Age. Hay 127 registros sin valor de edad, casi un 20%. Aunque se recomienda desechar características con más de un 5% de valores faltantes, mantenemos Age imputando los valores que faltan. Los métodos para imputar estos valores son los siguientes:

- Usar una constante global para completar el valor faltante.
- Usar la media o mediana de Age.
- Usar la media o la mediana del atributo de todos los registros que pertenecen a la misma clase que el registro que queremos imputar. Es decir si el registro pertenece a la clase sobrevivió, imputaremos la media de edad de todos los supervivientes.
- Calcular el valor más probable.

Voy a calcular el valor más probable usando el paquete `mice`. Para ello selecciono las variables que esten mas correlacionadas

```
# Correlación con Age
corr=Tit_mod
corr$Sex_num=as.numeric(as.factor(Tit_mod$Sex))
corr$Embarked_num=as.numeric(as.factor(Tit_mod$Embarked))
corr$Title_num=as.numeric(as.factor(Tit_mod$Title))
corr$Deck_num=as.numeric(as.factor(Tit_mod$Deck))
corr=corr[-c(3,8,9,10)]
res = cor(corr ,use = "complete.obs")

corrplot(res, type = "upper", order = "hclust",
         tl.col = "black", tl.srt = 45)
```



Las variables Pclass, Title, SibSp y Fam son las que presentan mayor correlación con Age y son las que usaré para calcular los valores de imputación.

```
# Imputación de Age
AgeCorrelation =c('Pclass', 'Title', 'SibSp', 'Fam', 'Age')

mod = mice(Tit_mod[AgeCorrelation], method='pmm', seed=2018)
```

```
##
##  iter imp variable
##   1   1  Age
##   1   2  Age
##   1   3  Age
##   1   4  Age
##   1   5  Age
##   2   1  Age
##   2   2  Age
##   2   3  Age
##   2   4  Age
##   2   5  Age
##   3   1  Age
##   3   2  Age
##   3   3  Age
##   3   4  Age
```

```
##    3    5  Age
##    4    1  Age
##    4    2  Age
##    4    3  Age
##    4    4  Age
##    4    5  Age
##    5    1  Age
##    5    2  Age
##    5    3  Age
##    5    4  Age
##    5    5  Age
```

```
Tit_mod$Age=complete(mod)$Age
```

Con la variable Age imputada crearé la variable binaria **Adult**

```
# Creación de la variable Adult
Tit_mod$Adult = if_else(Tit_mod$Age <18, 0, 1)
```

2.3.1.2 Tratamiento de valores vacios

```
summary(Tit_mod$Embarked)
```

```
##       C   Q   S
##   2 168  77 644
```

En la variable Embarked hay dos registros sin valor, los sustituimos por el valor mayoritario en este caso S.

```
# Imputación de Embarked
Tit_mod$Embarked=if_else(Tit_mod$Embarked == '',
                         'S', as.character(Tit_mod$Embarked))
Tit_mod$Embarked=as.factor(Tit_mod$Embarked)
summary(Tit_mod$Embarked)
```

```
##   C   Q   S
## 168  77 646
```

2.3.1.3 Tratamiento de ceros

La variable Fare tiene varios valores igual a cero, voy a sustituirlos por la media del valor de la variable según su etiqueta, es decir la media de la tarifa de los supervivientes a los que han sobrevivido y la media de la tarifa de los que no supervivieron a los que no lo hicieron.

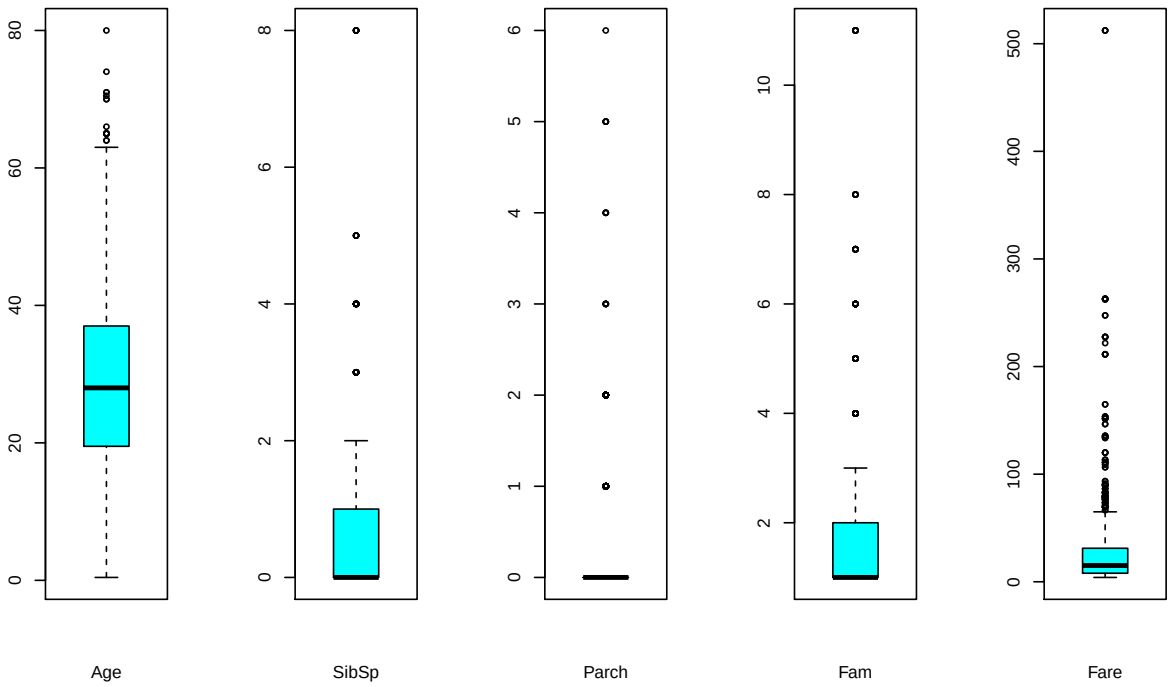
```
# Imputación de Fare
Tit_mod$Fare=case_when(Tit_mod$Fare == 0 & Tit_mod$Survived == 1 ~
                           mean(Tit_mod$Fare[Tit_mod$Survived == 1]),
                         Tit_mod$Fare == 0 & Tit_mod$Survived == 0 ~
                           mean(Tit_mod$Fare[Tit_mod$Survived == 0]),
```

```
TRUE ~ Tit_mod$Fare)
```

2.3.2 Identificación y tratamiento de valores extremos

Consideramos valores extremos aquellos que superan una vez y media el intervalo intercuartílico. Dibujo los boxplots de la variables numéricas para localizar los posibles valores extremos.

```
# Detección de outliers
par(mfrow=c(1,5))
bp_Age=boxplot(Tit_mod$Age, xlab="Age", col = "cyan")
bp_SibSp=boxplot(Tit_mod$SibSp, xlab="SibSp", col = "cyan")
bp_Parch=boxplot(Tit_mod$Parch, xlab="Parch",  col = "cyan")
bp_Fam=boxplot(Tit_mod$Fam, xlab="Fam",  col = "cyan")
bp_Fare=boxplot(Tit_mod$Fare, xlab="Fare", col = "cyan")
```

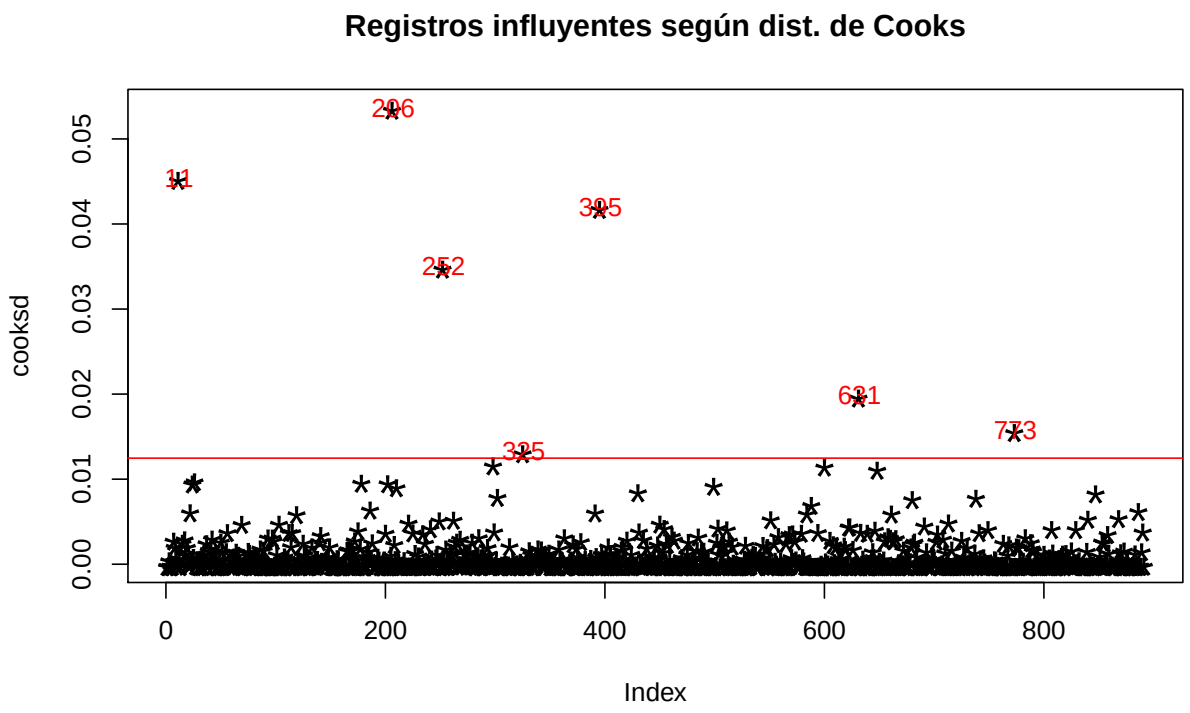


Se puede apreciar que estas características están bastante lejos de ser normales, todas están sesgadas a la izquierda. Las variables que reflejan el parentesco de los pasajeros están muy sesgadas porqué la mayoría de los pasajeros viajaban solos. A priori no parecen errores. El valor de 512,3 de la característica Fare pertenece a un único Ticket de cuatro cabinas de primera clase, por lo tanto podría no ser un error.

Para considerar cuan importantes son los registros más extremos estudiaré su importancia mediante una aproximación multivariable. Para ello se calcula la **distancia de cook** para cada registro, que mide la variación de un modelo de regresión al calcularlo sin ese registro. De esta forma se determina que registros son más influyentes. Un registro se considera influyente cuando su distancia es superior a cuatro veces la media.

Como hemos visto que los outliers no se deben a errores, voy a elevar el umbral a 10 veces la media de la distancia para encontrar los outliers realmente influyentes y eliminarlos.

```
# Influencia de los registros
mod2=lm(Survived ~ ., data=Tit_mod)
cooksd = cooks.distance(mod2)
# plot cook's distance
plot(cooksd, pch="*", cex=2, main="Registros influyentes según dist. de Cooks")
# add cutoff line
abline(h = 10*mean(cooksd, na.rm=T), col="red")
# add labels
text(x=1:length(cooksd)+1, y=cooksd,
     labels=ifelse(cooksd>10*mean(cooksd, na.rm=T),
                   names(cooksd),""), col="red")
```




```
# Determinación de los outliers influyentes
influent = as.numeric(names(cooksd)[(cooksd > 10*mean(cooksd, na.rm=T))])

out_Fares = as.numeric(rownames(Tit_mod[Tit_mod$Fare > min(bp_Fare$out),]))
out_Age = as.numeric(rownames(Tit_mod[Tit_mod$Age > min(bp_Age$out),]))
Tit_mod[intersect(influent, out_Fares),]
```

```
##     Survived Pclass  Sex Age SibSp Parch  Fare Embarked Title Deck
## 325        0      3 male  24     8     2 69.55        S    Mr    Z
##     HasCabin Fam IsAlone Adult
```

```
## 325        0  11       0     1
```

```
Tit_mod[intersect(influent, out_Age),]
```

```
##     Survived Pclass  Sex Age SibSp Parch Fare Embarked Title Deck HasCabin
## 631        1      1 male  80     0     0   30        S    Mr    A        1
##     Fam IsAlone Adult
## 631   1       1     1
```

El único outlier realmente influyente es el 631, perteneciente a un hombre de 80 años

```
# Supresión del outlier
Tit_mod=Tit_mod[-631,]
```

Exportación de los datos preprocesados Despues de realizar la integración, validación, limpieza y creción de nuevas variables, sobre los datos iniciales, guardamos el resultado en Titanic_clean.csv:

```
# Exportación a CSV
write.csv(Tit_mod, "Titanic_clean.csv")
```

2.4 Análisis de los datos

2.4.1 Selección de los grupos de datos que se quieren analizar/comparar

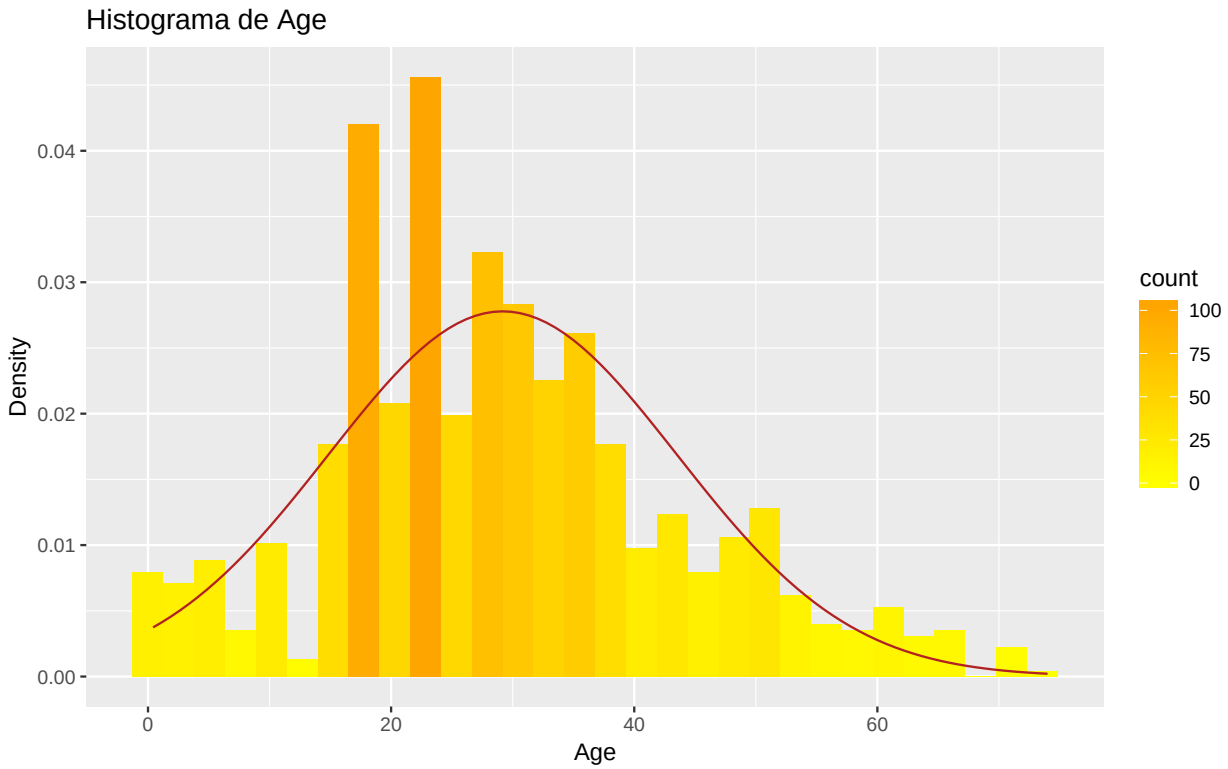
Selecciono los grupos que quiero analizar.

```
# Agrupación por supervivencia
Titanic.survived=Tit_mod[Tit_mod$Survived == 1,]
Titanic.notsurvived=Tit_mod[Tit_mod$Survived == 0,]

# Agrupación por genero
Titanic.female=Tit_mod[Tit_mod$Sex == 'female',]
Titanic.male=Tit_mod[Tit_mod$Sex == 'male',]

# Agrupación por edad
Titanic.adult=Tit_mod[Tit_mod$Adult == 1,]
Titanic.child=Tit_mod[Tit_mod$Adult == 0,]

# Agrupación por clase de pasaje
Titanic.first=Tit_mod[Tit_mod$Pclass == 1,]
Titanic.second=Tit_mod[Tit_mod$Pclass == 2,]
Titanic.third=Tit_mod[Tit_mod$Pclass == 3,]
```

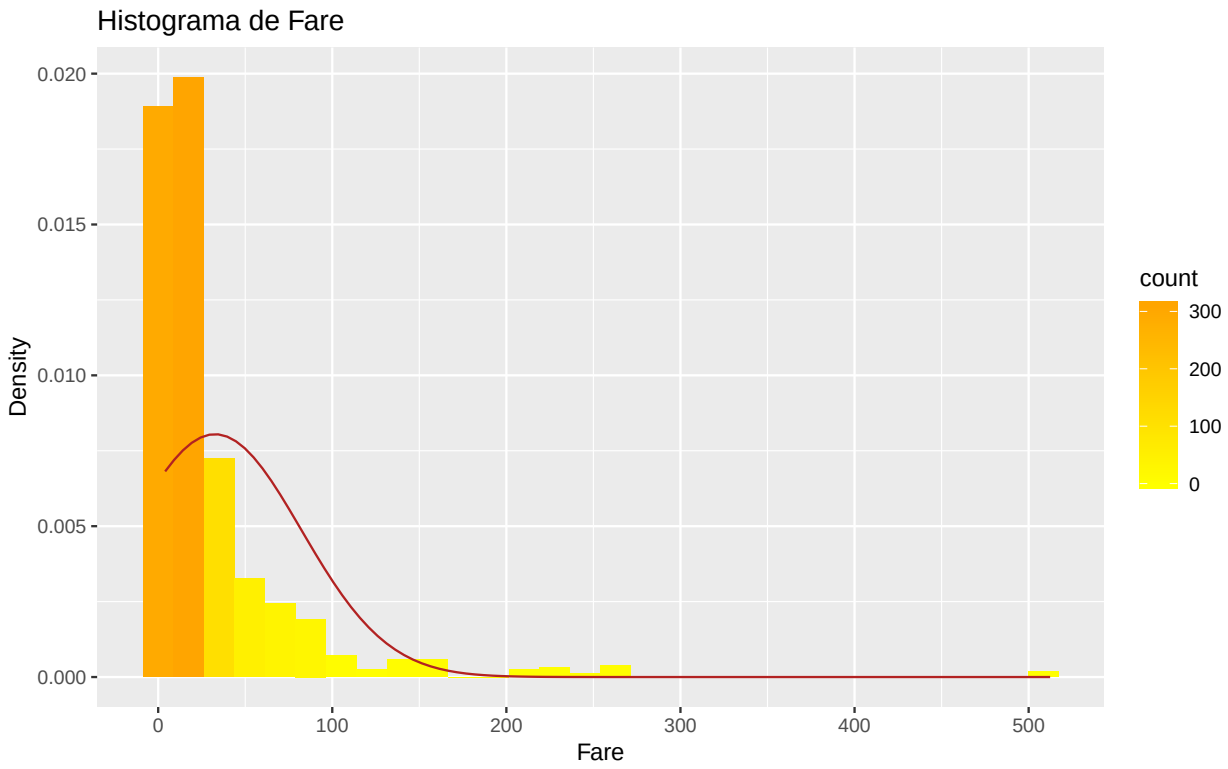
2.4.2 Comprobación de la normalidad y homogeneidad de la varianza

Para el estudio de la normalidad de las variables cuantitativas, dibujaré un histograma de estas variables (solo estuidaré Fam por ser suma de SibSp y Parch) y superpondré la curva de distribución normal con la misma media y desviación estándar que muestran los datos.

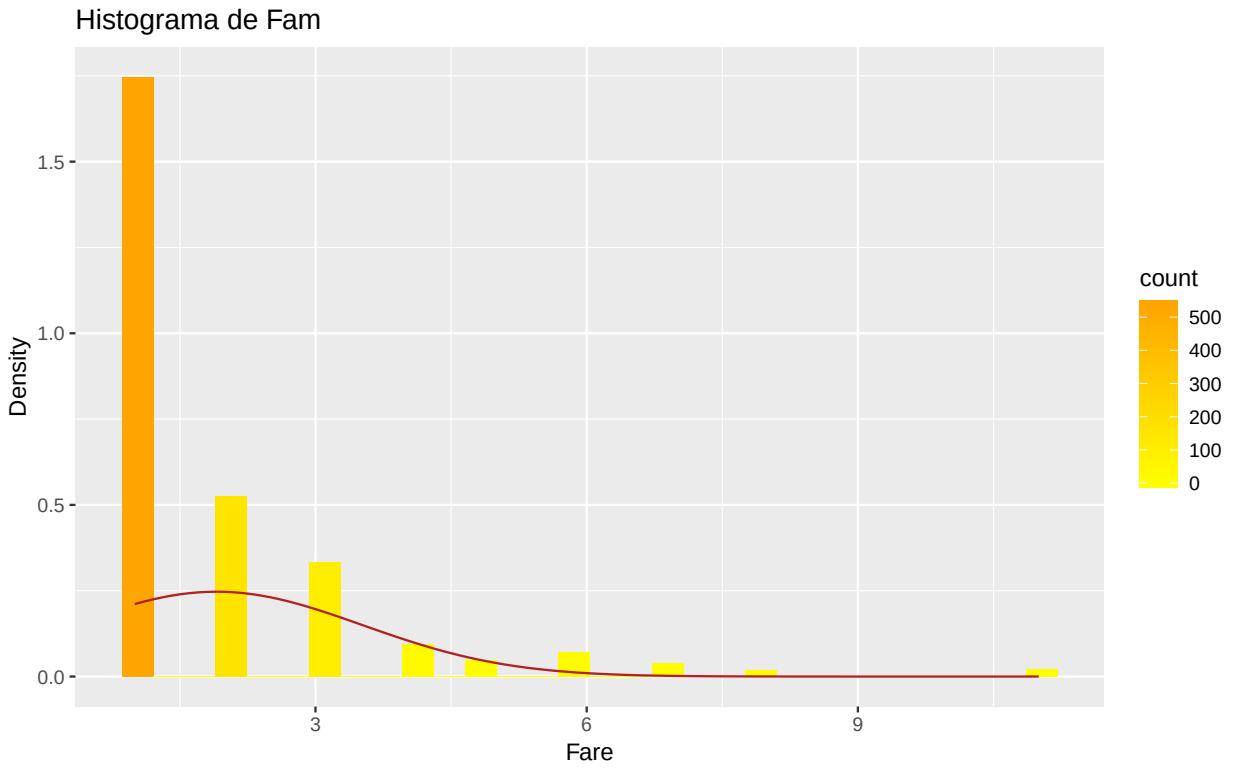
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




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



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



Como podemos comprobar, claramente Fare y Fam no son distribuciones normales. Voy a comprobar si Age tiene una distribución normal aplicando el test *Lilliefor*, una modificación del test *Kolmogorov-Smirnov* para varianza y media desconocida.

```
# Contraste de normalidad
lillie.test(x = Tit_mod$Age)
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  Tit_mod$Age
## D = 0.073104, p-value = 3.628e-12
```

Para un nivel de significancia $\alpha = 0.05$, debemos rechazar la hipótesis nula H_0 :Es una distribución normal, puesto que el p-value es < que α, es decir Age no sigue una distribución normal.

Estudiaré la homogeneidad de varianzas de Age para las agrupaciones del apartado anterior. Usaré el *test de Fligner-Killeen* pues es el recomendado para distribuciones no normales, en el caso de distribuciones normales podríamos haber usado *F-test* o el *test de Bartlet*

```
# Homogeneidad de varianzas
fligner.test(Age ~ Survived, data = Tit_mod)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
```

```
##
## data:  Age by Survived
## Fligner-Killeen:med chi-squared = 0.72674, df = 1, p-value =
## 0.3939
```

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

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  Age by Sex
## Fligner-Killeen:med chi-squared = 0.3219, df = 1, p-value = 0.5705
```

```
fligner.test(Age ~ Pclass, data = Tit_mod)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  Age by Pclass
## Fligner-Killeen:med chi-squared = 15.526, df = 2, p-value =
## 0.0004251
```

Como podemos apreciar la homogeneidad de varianzas para las agrupaciones por supervivencia y genero si se mantienen, sin embargo para las categorías de la agrupación por clase de pasaje debemos rechazar la hipótesis nula, pues el p-valor es menor que el valor de significación.

2.4.3 Aplicación de pruebas estadísticas para comparar los grupos de datos

Vamos a calcular la media de supervivencia.

```
# Cálculo de la media de supervivencia
Survived.mean=mean(Tit_mod$Survived)
print(paste0("Media de supervivencia: ", round(Survived.mean*100,2), "%"))
```

```
## [1] "Media de supervivencia: 38.31%"
```

2.4.3.1 ¡¡¡Las mujeres y los niños primero!!!

Voy a comprobar si es cierto que la probabilidad de sobrevivir es mayor para las mujeres y los menores de edad. Para ello realizaré un contraste de hipótesis para $\alpha = 0.05$ siendo las hipotesis de contraste:

$H_0 : \mu = \mu_0$

$H_1 : \mu < \mu_0$

donde μ_0 = 0.3831

Usando la agrupación Titanic.male y Titanic.adult

```r
# Contraste de hipótesis para media superivencia de hombres y adultos
t.test( Titanic.male$Survived, mu=Survived.mean, alternative="less" )
```

```
##
##  One Sample t-test
##
## data:  Titanic.male$Survived
## t = -12.02, df = 575, p-value < 2.2e-16
## alternative hypothesis: true mean is less than 0.3831461
## 95 percent confidence interval:
##       -Inf 0.2143167
## sample estimates:
## mean of x
##    0.1875
```

```r
t.test( Titanic.adult$Survived, mu=Survived.mean, alternative="less" )
```

```
##
##  One Sample t-test
##
## data:  Titanic.adult$Survived
## t = -1.523, df = 748, p-value = 0.0641
## alternative hypothesis: true mean is less than 0.3831461
## 95 percent confidence interval:
##       -Inf 0.3853164
## sample estimates:
## mean of x
## 0.3564753
```

Por lo tanto para un nivel de significación $\alpha = 0.05$ **podemos afirmar** que las mujeres primero, pero no podemos hacer lo mismo para los menores, pues el p-valor es mayor que α y por lo tanto no podemos rechazar la hipótesis nula.

2.4.3.2 ¿Tuvo influencia la clase social?

Vamos a estudiar como afectó la clase de pasaje.

```r
# Contraste de hipótesis para clase de pasaje
t.test( Titanic.third$Survived, mu=Survived.mean, alternative="less" )
```

```
##
##  One Sample t-test
##
## data:  Titanic.third$Survived
## t = -7.2725, df = 490, p-value = 7.032e-13
## alternative hypothesis: true mean is less than 0.3831461
## 95 percent confidence interval:
```

```
##       -Inf 0.2742643
## sample estimates:
## mean of x
## 0.2423625
```

La hipótesis nula es que la media de supervivencia no depende de la clase de pasaje. El p-valor menor que el nivel de significación determina que debemos rechazar la hipótesis nula, por lo tanto podemos afirmar que los pasajeros de tercera clase tuvieron una media menor de supervivencia que la media global del pasaje.

2.4.3.3 Modelos predictivos

Vamos a generar una serie de modelos predictivos para poder calcular resultado de supervivencia del concurso de Kaggel. Para ello voy ha usar validación cruzada k-fold con repetición. Este metodo evalúa el rendimiento del modelo en diferentes subconjuntos de los datos de entrenamiento y luego calcula el promedio del error de predicción. Usaré un valor de k=10 y cinco repeticiones.

```
Tit_mod$Survived=as.factor(Tit_mod$Survived)
levels(Tit_mod$Survived) = list(survived="1", notsurvived="0")
## 80% de la muesta
smp_size = floor(0.80 * nrow(Tit_mod))

## set the seed to make your partition reproducible
set.seed(2018)
train_ind = sample(seq_len(nrow(Tit_mod)), size = smp_size)

train = Tit_mod[train_ind, ]
test = Tit_mod[-train_ind, ]
```

```
# Validación cruzada k-fold
# Definición del training control
train.control = trainControl(
  method='repeatedcv', number=10, repeats=5, search = "grid",
  savePredictions = "final", index = createResample(train$Survived, 10),
  summaryFunction = twoClassSummary, classProbs = TRUE)

# Listado de modelos
mod_list = c("rf", "glm", "gbm", "glmboost", "nnet", "treebag", "svmLinear")

multi_mod = caretList(Survived ~ . , data = train, trControl = train.control,
                      methodList = mod_list, metric = 'ROC')
```

```
# Resultados
names(multi_mod) <- sapply(multi_mod, function(x) x$method)
sort(sapply(multi_mod, function(x) min(x$results$ROC)))
```

```
##      nnet svmLinear   treebag        rf       glm  glmboost       gbm
## 0.6638029 0.8387823 0.8413180 0.8470128 0.8636133 0.8649241 0.8676005
```

A tenor de los resultados, vemos que el mejor modelo es el **glmboost** con un valor mínimo ROC del **87.82%**. Calculamos la matriz de confusión para este modelo y para el random forest.

```
pred_glmboost = predict(multi_mod$glmboost, test)
pred_rf = predict(multi_mod$rf, test)
a=confusionMatrix(table(true = test$Survived, pred = pred_glmboost))
b=confusionMatrix(table(true = test$Survived, pred = pred_rf))
```

Vemos que la precisión obtenida en la predicción de los valores de test es muy similar del **80.34%** para el glmboost y del **81.46%** para el random forest. Vamos a intentar apilar los distintos modelos para intentar mejorar el resultado obtenido por cada uno de ellos por separado. Para ello crearé un nuevo conjunto de datos con las predicciones y la variable de clase, y aplicaré un modelo gbm para predecir nuevamente los valores de test.

```
predDF.train = data.frame(rf = predict(multi_mod$rf, train),
                      glm = predict(multi_mod$glm, train),
                      gbm = predict(multi_mod$gbm, train),
                      glmboost = predict(multi_mod$glmboost, train),
                      nnet = predict(multi_mod$nnet, train),
                      treebag = predict(multi_mod$treebag, train),
                      svmLinear = predict(multi_mod$svmLinear, train),
                      Survived = train$Survived)

predDF.test = data.frame(rf = predict(multi_mod$rf, test),
                      glm = predict(multi_mod$glm, test),
                      gbm = predict(multi_mod$gbm, test),
                      glmboost = predict(multi_mod$glmboost, test),
                      nnet = predict(multi_mod$nnet, test),
                      treebag = predict(multi_mod$treebag, test),
                      svmLinear = predict(multi_mod$svmLinear, test))
```

```
set.seed(2018)
stacking = train(Survived~.,data=predDF.train,method='rf')

pred_stacking = predict(stacking, predDF.test)
confusionMatrix(table(true = test$Survived, pred = pred_stacking))
```

```
## Confusion Matrix and Statistics
##
##              pred
## true          survived notsurvived
##   survived          59          17
##   notsurvived       10          92
##
##                Accuracy : 0.8483
##                  95% CI : (0.787, 0.8976)
```

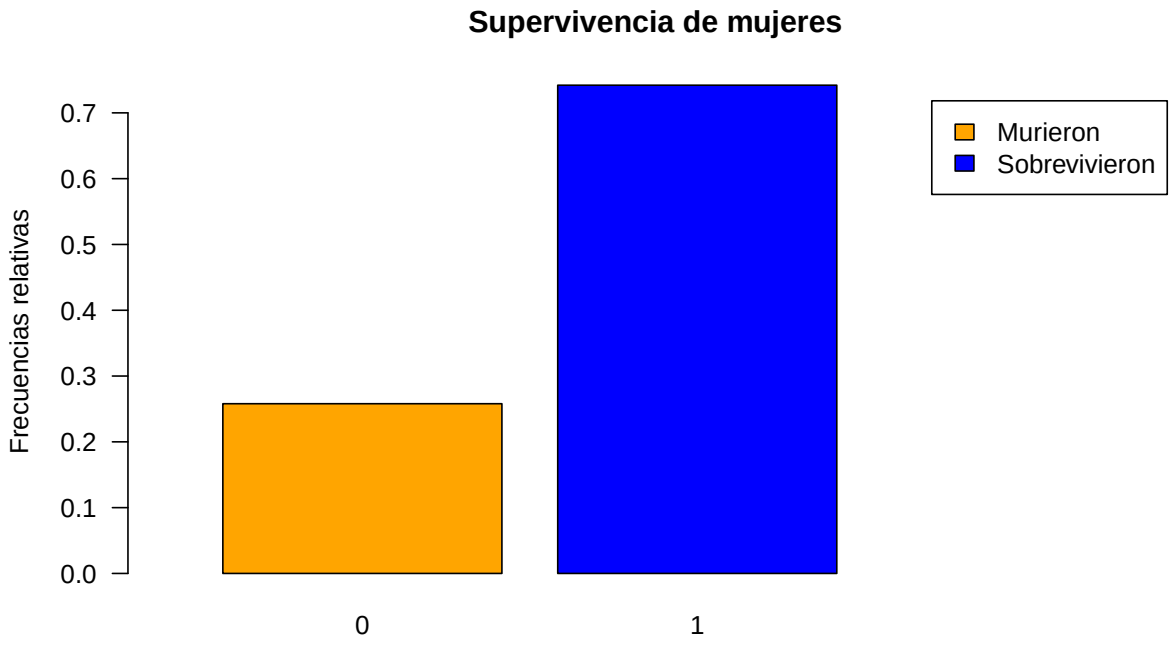
```
##     No Information Rate : 0.6124
##     P-Value [Acc > NIR] : 4.874e-12
##
##                   Kappa : 0.6863
##  Mcnemar's Test P-Value : 0.2482
##
##             Sensitivity : 0.8551
##             Specificity : 0.8440
##          Pos Pred Value : 0.7763
##          Neg Pred Value : 0.9020
##              Prevalence : 0.3876
##          Detection Rate : 0.3315
##    Detection Prevalence : 0.4270
##       Balanced Accuracy : 0.8496
##
##        'Positive' Class : survived
##
```

Como podemos ver hemos aumentado la precisión en los valores de test hasta un **84.83%**, mejorando el resultado en casi un **5%**. Con este modelo, como primera aproximación, podremos calcular los valores de supervivencia para los datos de test de la competición de Kaggle.

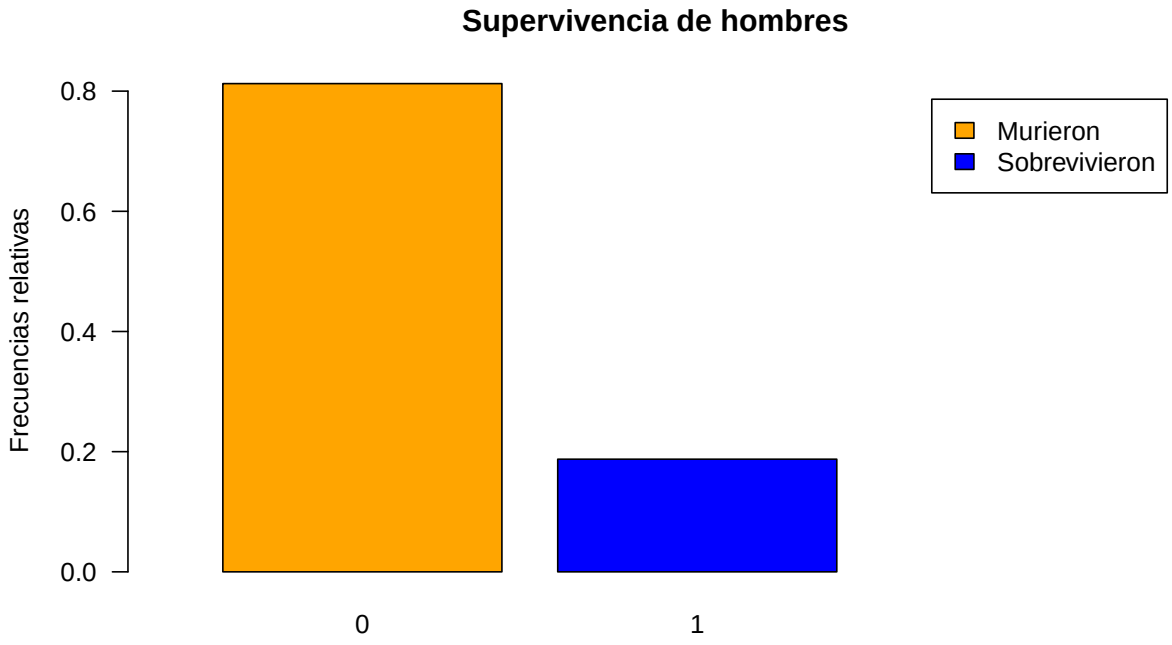
2.5 Representación de los resultados a partir de tablas y gráficas

2.5.1 Diagramas de barras

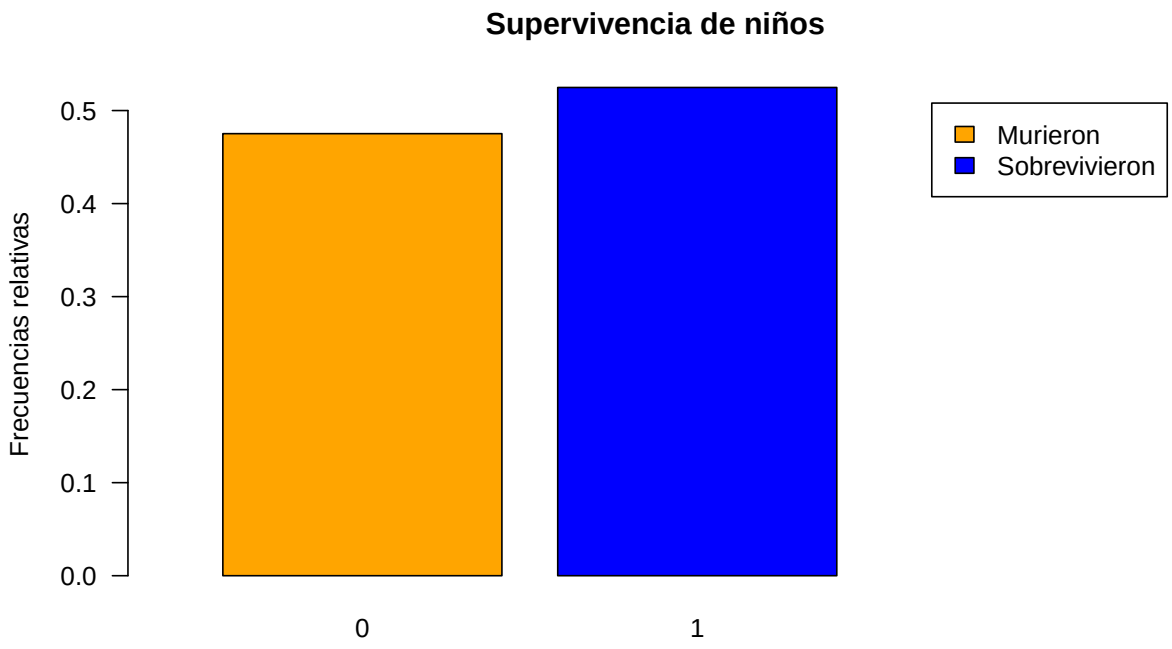
```
barplot(prop.table(table(Titanic.female$Survived)),col=c("orange","blue"),
     main="Supervivencia de mujeres", xlab="", ylab='Frecuencias relativas',
     legend.text=c("Murieron","Sobrevivieron"),xlim=c(0,3.5),las=1)
```



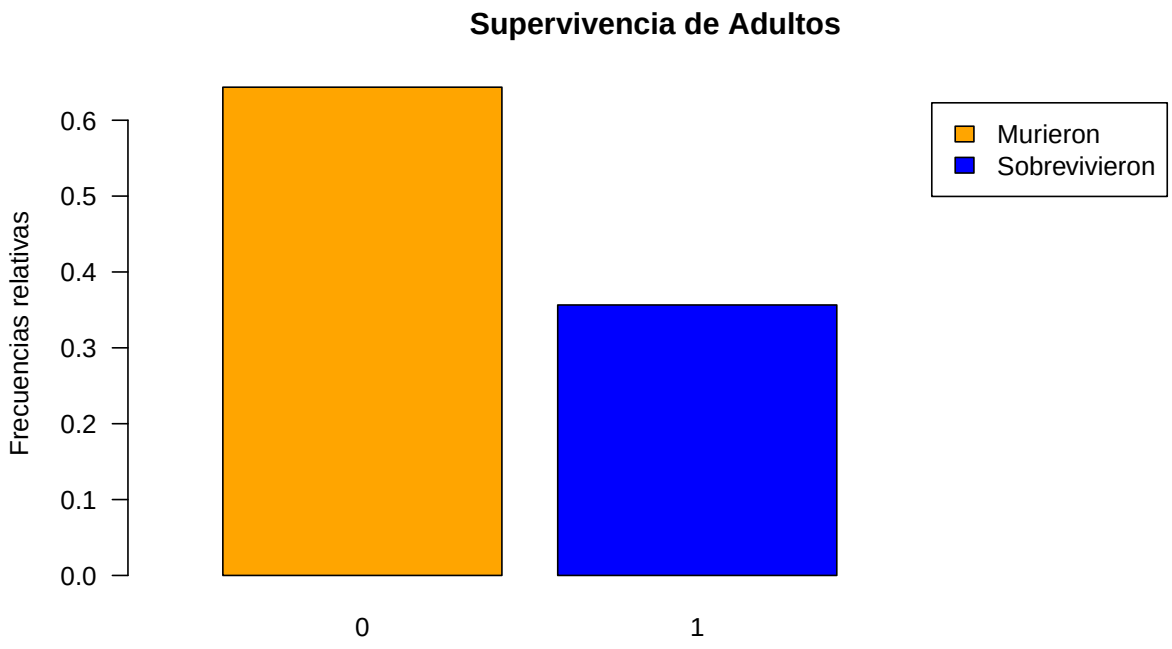

```
barplot(prop.table(table(Titanic.male$Survived)),col=c("orange","blue"),
     main="Supervivencia de hombres", xlab="", ylab='Frecuencias relativas',
     legend.text=c("Murieron","Sobrevivieron"),xlim=c(0,3.5),las=1)
```



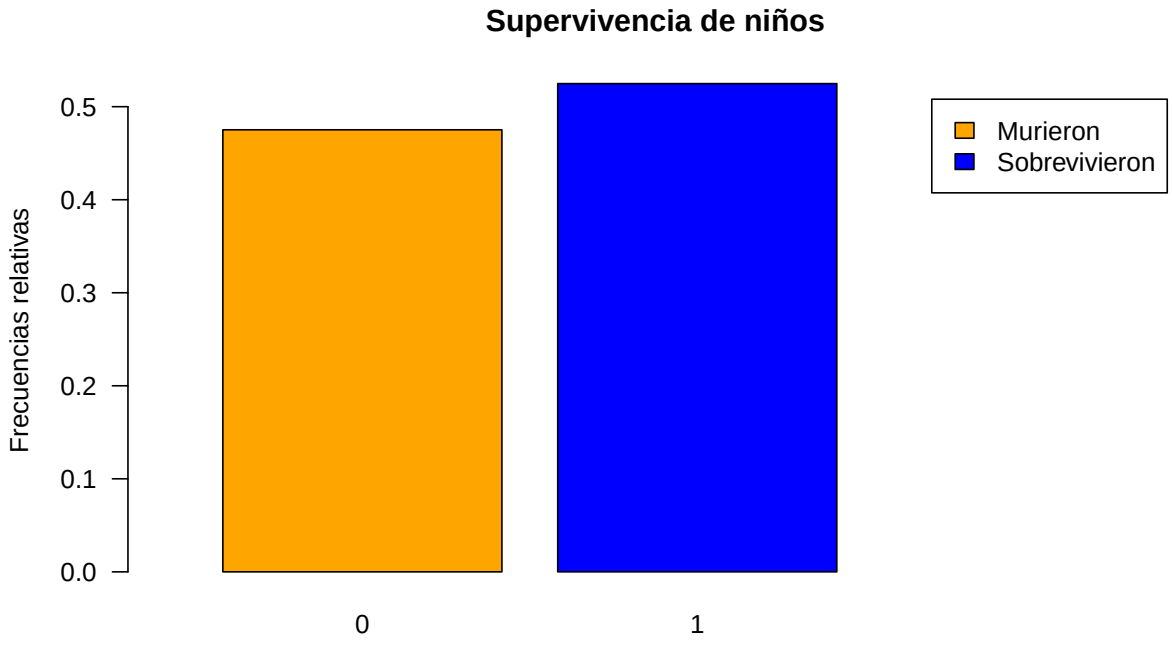
```
barplot(prop.table(table(Titanic.child$Survived)),col=c("orange","blue"),
     main="Supervivencia de niños", xlab="", ylab='Frecuencias relativas',
     legend.text=c("Murieron","Sobrevivieron"),xlim=c(0,3.5),las=1)
```



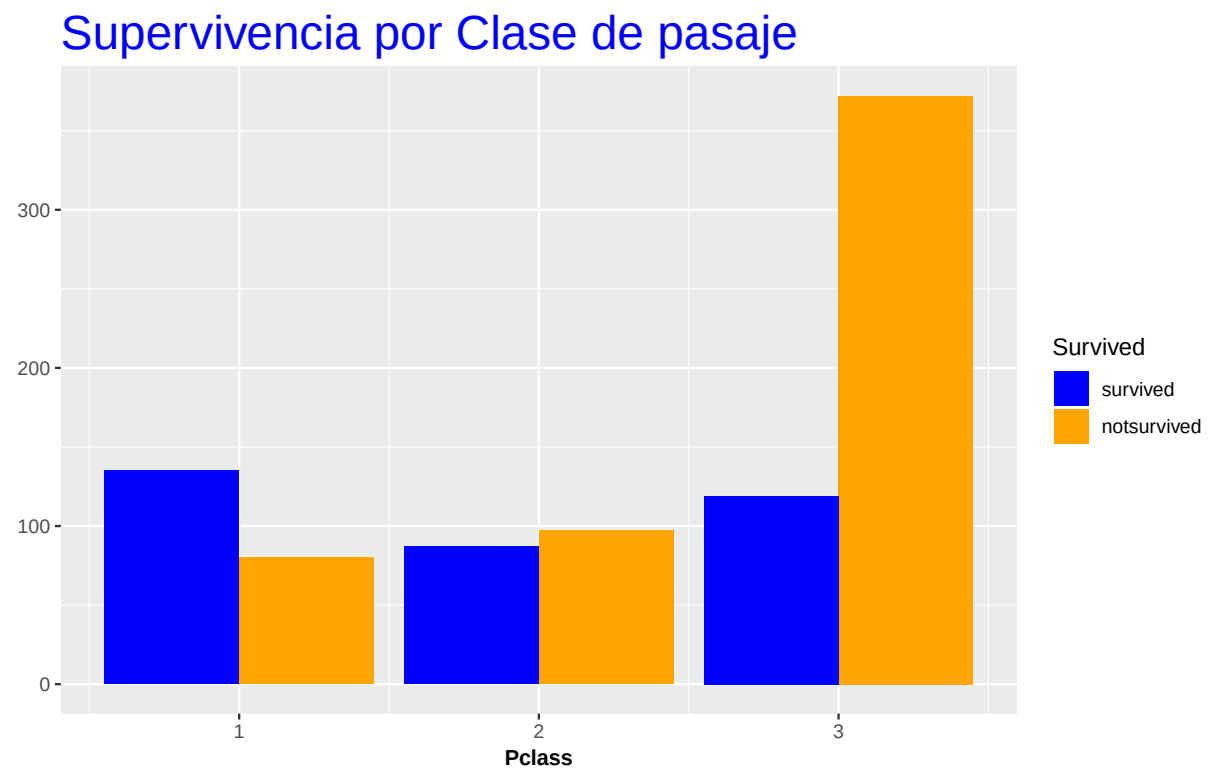

```
barplot(prop.table(table(Titanic.adult$Survived)),col=c("orange","blue"),
     main="Supervivencia de Adultos", xlab="", ylab='Frecuencias relativas',
     legend.text=c("Murieron","Sobrevivieron"),xlim=c(0,3.5),las=1)
```



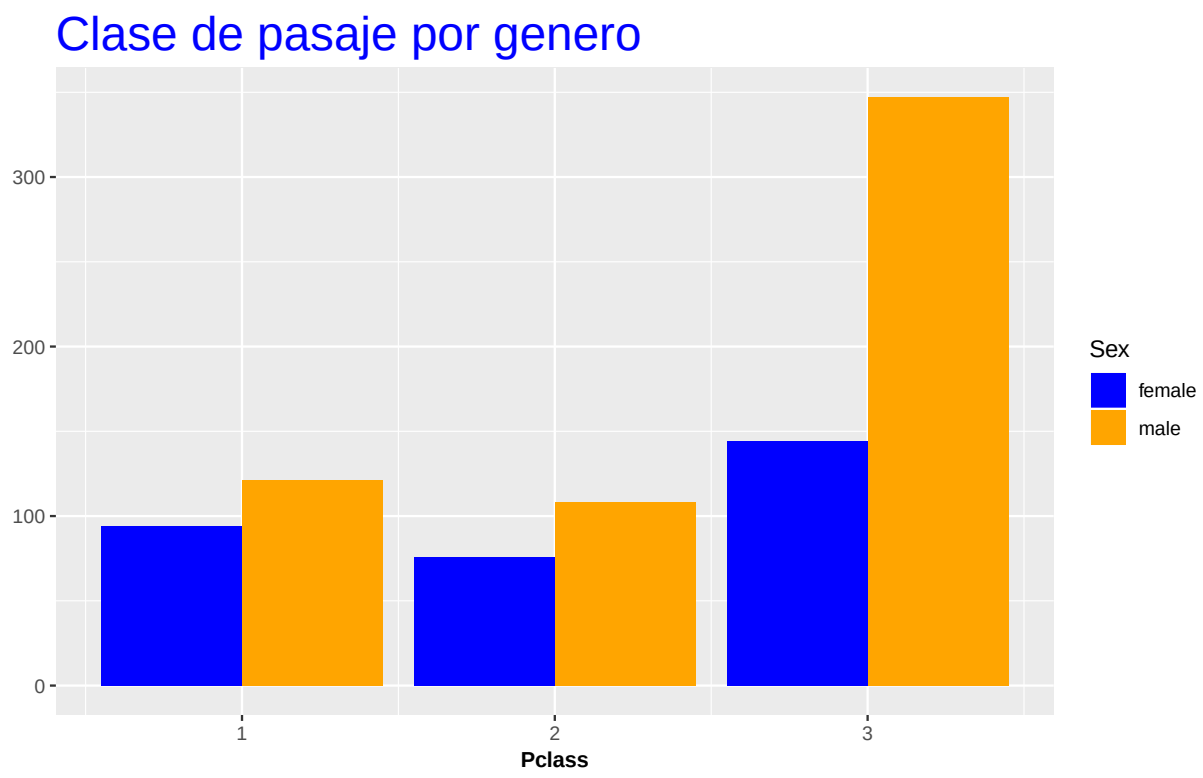

```
barplot(prop.table(table(Titanic.child$Survived)),col=c("orange","blue"),
     main="Supervivencia de niños", xlab="", ylab='Frecuencias relativas',
     legend.text=c("Murieron","Sobrevivieron"),xlim=c(0,3.5),las=1)
```



```r
g = ggplot(Tit_mod, aes(Pclass, fill=Survived) ) +
  labs(title = "Supervivencia por Clase de pasaje")+ylab("") +
  theme(plot.title = element_text(size = rel(2), colour = "blue"))

g+geom_bar(position="dodge") + scale_fill_manual(values = alpha(c("blue", "orange"), 1)
  theme(axis.title.x = element_text(face="bold", size=10))
```




```r
g = ggplot(Tit_mod, aes(Pclass, fill=Sex) ) +
  labs(title = "Clase de pasaje por genero")+ylab("") +
  theme(plot.title = element_text(size = rel(2), colour = "blue"))

g+geom_bar(position="dodge") + scale_fill_manual(values = alpha(c("blue", "orange"), 1)
  theme(axis.title.x = element_text(face="bold", size=10))
```



2.6 Resolución del problema

A partir de los resultados obtenidos podemos asegurar que en el accidente del Titanic, las mujeres se marcharon primero, pues la tasa de supervivencia de las mujeres fue muy superior a la de los hombres.

Sin embargo con los menores de edad, la tasa de supervivencia estuvo muy pareja, tanto, que estadísticamente no podemos afirmar que tuviesen un índice de supervivencia superior a la media.

A las mujeres les pudo ayudar, que el porcentaje de ellas que viajaban en primera clase es bastante mayor al porcentaje de hombres que viajaban en primera. La gran mayoría viajaba en tercera. Podemos afirmar estadísticamente que viajar en primera, tuvo una media de supervivencia superior a la media del pasaje.

La variable, mas importante para el cálculo de supervivencia es el Sexo, y los modelos individuales de predicción de la supervivencia están entorno al 80% de precisión. Esta precisión en una primera aproximación la podemos aumentar en un 5% mediante el apilado de distintos modelos individuales.

2.7 Codigo

El código en R, con el que se ha realizado la limpieza, análisis y representación de los datos se puede descargar de Github en el siguiente enlace:

3 Recursos

Titanic: Machine Learning from Disaster [en línea] [Consulta: Diciembre de 2018] https://www.kaggle.com/c/titanic/overview

Correlation matrix : A quick start guide to analyze, format and visualize a correlation matrix using R software [en línea] [Consulta: Diciembre de 2018] http://www.sthda.com/english/wiki/correlation-matrix-a-quick-start-guide-to-analyze

r-statistics.co by Selva Prabhakaran Outlier Treatment [en línea] [Consulta: Diciembre de 2018] http://r-statistics.co/Outlier-Treatment-With-R.html

Análisis de Normalidad: gráficos y contrastes de hipótesis Joaquín Amat Rodrigo j.amatrodrigo@gmail.com Enero, 2016 [en línea] [Consulta: Diciembre de 2018] https://rpubs.com/Joaquin_AR/218465

Mejorando la exactitud en la clasificación mediante ensamble de modelos Oct 22, 2016 [en línea] [Consulta: Diciembre de 2018] http://amsantac.co/blog/es/2016/10/22/model-stacking-classification-r-es.html

Dalgaard, Peter. 2002. *Statistics and Computing*. New York: «Springer».

W. Osborne, Jason. 2013. *Best Practices in Data Cleaning*. United States of America: «SAGE».