

Tipología y ciclo de vida de los datos. Práctica 2

Baltasar Boix / Yago Ezcurra

13/5/2021

Contents

1	Titanic - Machine Learning from Disaster. Kaggle competition.	2
2	Descripción del dataset. ¿Por qué es importante y qué pregunta/problema pretende responder?	2
2.1	Lectura y análisis previo del dataset.	2
3	Integración y selección de los datos de interés a analizar.	4
3.1	Selección de los datos de interés	6
4	Limpieza de los datos.	6
4.1	¿Los datos contienen ceros o elementos vacíos? ¿Cómo gestionarías cada uno de estos casos? .	6
4.2	Identificación y tratamiento de valores extremos.	8
5	Análisis de los datos.	8
5.1	Selección de los grupos de datos que se quieren analizar/comparar (planificación de los análisis a aplicar).	8
5.2	Comprobación de la normalidad y homogeneidad de la varianza.	9
5.3	Aplicación de pruebas estadísticas para comparar los grupos de datos. En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes.	10
6	Representación de los resultados a partir de tablas y gráficas.	12
7	Resolución del problema. A partir de los resultados obtenidos, ¿cuáles son las conclusiones? ¿Los resultados permiten responder al problema?	12
8	Contribuyentes	14
9	References	14

1 Titanic - Machine Learning from Disaster. Kaggle competition.

```
library(dplyr)
library(ggplot2)
library(tidyverse)
require(laers)
library(GGally)
library(knitr)
library(kableExtra)
library(gridExtra)
library(DescTools)
library(caret)
library(VIM)
```

2 Descripción del dataset. ¿Por qué es importante y qué pregunta/problema pretende responder?

2.1 Lectura y analisis previo del dataset.

Data Dictionary

Variable	Definition	Key
survival	Survival	0 = No, 1 = Yes
pclass	Ticket class	1 = 1st, 2 = 2nd, 3 = 3rd
sex	Sex	
Age	Age in years	
sibsp	# of siblings / spouses aboard the Titanic	
parch	# of parents / children aboard the Titanic	
ticket	Ticket number	
fare	Passenger fare	
cabin	Cabin number	
embarked	Port of Embarkation	C = Cherbourg, Q = Queenstown, S = Southampton

Variable Notes

pclass: A proxy for socio-economic status (SES)

1st = Upper

2nd = Middle

3rd = Lower

age: Age is fractional if less than 1. If the age is estimated, is it in the form of xx.5

sibsp: The dataset defines family relations in this way...

Sibling = brother, sister, stepbrother, stepsister

Spouse = husband, wife (mistresses and fiancés were ignored)

parch: The dataset defines family relations in this way...

Parent = mother, father

Child = daughter, son, stepdaughter, stepson

Some children travelled only with a nanny, therefore parch=0 for them.

Figure 1: Descripción del dataset obtenido en Kaggle

```
df <- read_csv('../data/train.csv')
df_t <- read_csv('../data/test.csv')
df_t$Survived <- NA
df <- bind_rows(df, df_t)
df$Survived <- factor(df$Survived)
df$Pclass <- factor(df$Pclass)
df$Sex <- factor(df$Sex)
df$SibSp <- factor(df$SibSp)
df$Parch <- factor(df$Parch)
df$Embarked <- factor(df$Embarked)
summary(df)
```

```
## PassengerId  Survived  Pclass    Name                Sex
## Min.       : 1      0 :549    1:323  Length:1309      female:466
## 1st Qu.: 328      1 :342    2:277  Class :character  male :843
## Median : 655      NA's:418  3:709  Mode  :character
## Mean      : 655
## 3rd Qu.: 982
## Max.      :1309
##
##      Age      SibSp      Parch      Ticket                Fare
## Min.   : 0.17   0:891    0      :1002  Length:1309      Min.   : 0.000
## 1st Qu.:21.00  1:319    1      : 170  Class :character  1st Qu.: 7.896
## Median :28.00  2: 42    2      : 113  Mode  :character  Median :14.454
## Mean    :29.88  3: 20    3      :   8                      Mean  :33.295
## 3rd Qu.:39.00  4: 22    4      :   6                      3rd Qu.:31.275
## Max.    :80.00  5:  6    5      :   6                      Max.   :512.329
## NA's    :263   8:  9  (Other):  4                      NA's   :1
##      Cabin      Embarked
## Length:1309      C :270
## Class :character  Q :123
## Mode  :character  S :914
##                  NA's: 2
##
##
##
##df_str(df)
```

3 Integración y selección de los datos de interés a analizar.

Creamos la variable dicotómica `Child` para diferenciar los niños de los adultos (>12años).

Creamos la variable `n_ticket` con el número de personas que viajan con el mismo ticket.

Separamos del `Name` el título (`title_name`) y el primer apellido (`first_name`).

Simplificamos `title_name` en cuatro niveles.

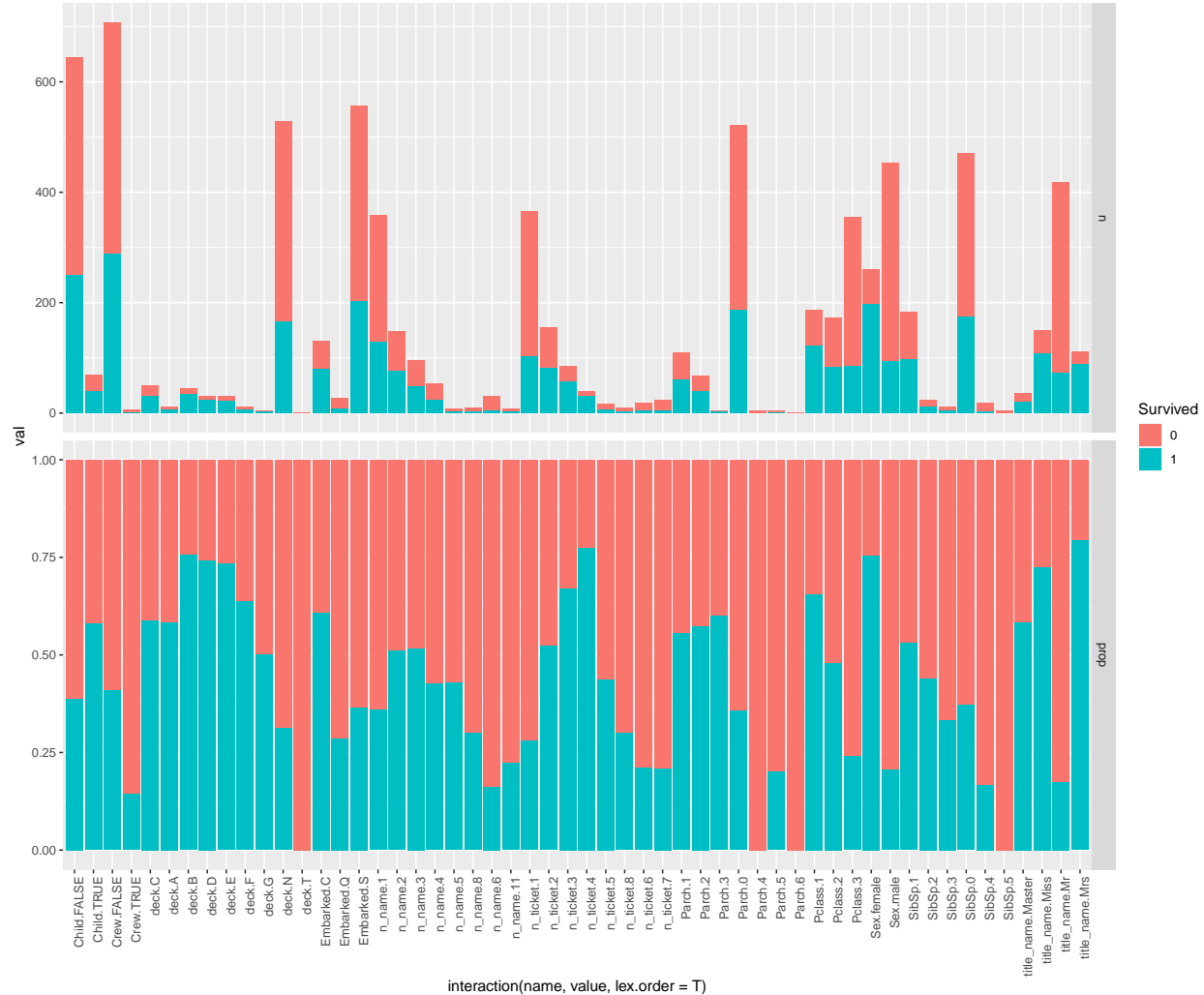
Creamos la variable `Crew`. TRUE para la personas con `Fare==0` que consideramos que viajan como tripulación.

```
df <- df %>%
  mutate(Child=factor(Age<=12))
df <- left_join(df, df %>%
  group_by(Ticket) %>%
  summarize(n_ticket=n())) %>%
  mutate(n_ticket=factor(n_ticket))
df <- df %>%
  separate(Name, c('first_name', 'rest_name'), sep=', ', remove=F) %>%
  separate(rest_name, c('title_name', 'rest_name'), sep='\\.') %>%
  select(-rest_name)
df$title_name[df$title_name %in% c('Capt', 'Col', 'Don', 'Dr', 'Jonkheer',
  'Major', 'Rev', 'Sir')] <- 'Mr'
df$title_name[df$title_name %in% c('Lady', 'Mme', 'the Countess', 'Dona')] <- 'Mrs'
df$title_name[df$title_name %in% c('Mlle', 'Ms')] <- 'Miss'
df$title_name <- factor(df$title_name)
df <- left_join(df, df %>%
```

```

    group_by(Pclass, first_name) %>%
      summarize(n_name=n()) %>%
mutate(n_name=factor(n_name))
df <- df %>%
  mutate(deck=if_else(is.na(Cabin), 'N', str_sub(Cabin, 1, 1))) %>%
  mutate(deck=factor(deck))
df$Fare[is.na(df$Fare)] <- df %>% filter(Pclass==3) %>% summarize(mean(Fare, na.rm=T)) %>% pull()
df$Embarked[is.na(df$Embarked)] <- 'S'
df <- df %>%
  mutate(Crew=factor(if_else(Fare==0, TRUE, FALSE)))
df %>%
  select(where(is.factor)) %>%
  na.omit() %>%
  pivot_longer(-Survived) %>%
  group_by(name, value, Survived) %>%
  summarize(n=n()) %>%
  mutate(prop=prop.table(n)) %>%
  pivot_longer(c(n,prop), names_to='tipo', values_to='val') %>%
  ggplot(aes(x=interaction(name,value, lex.order = T), y=val, fill=Survived)) +
    geom_bar(stat='identity', position='stack') +
    facet_grid(tipo ~ ., scale='free_y') +
    theme(axis.text.x = element_text(angle = 90, vjust = 1, hjust=1))

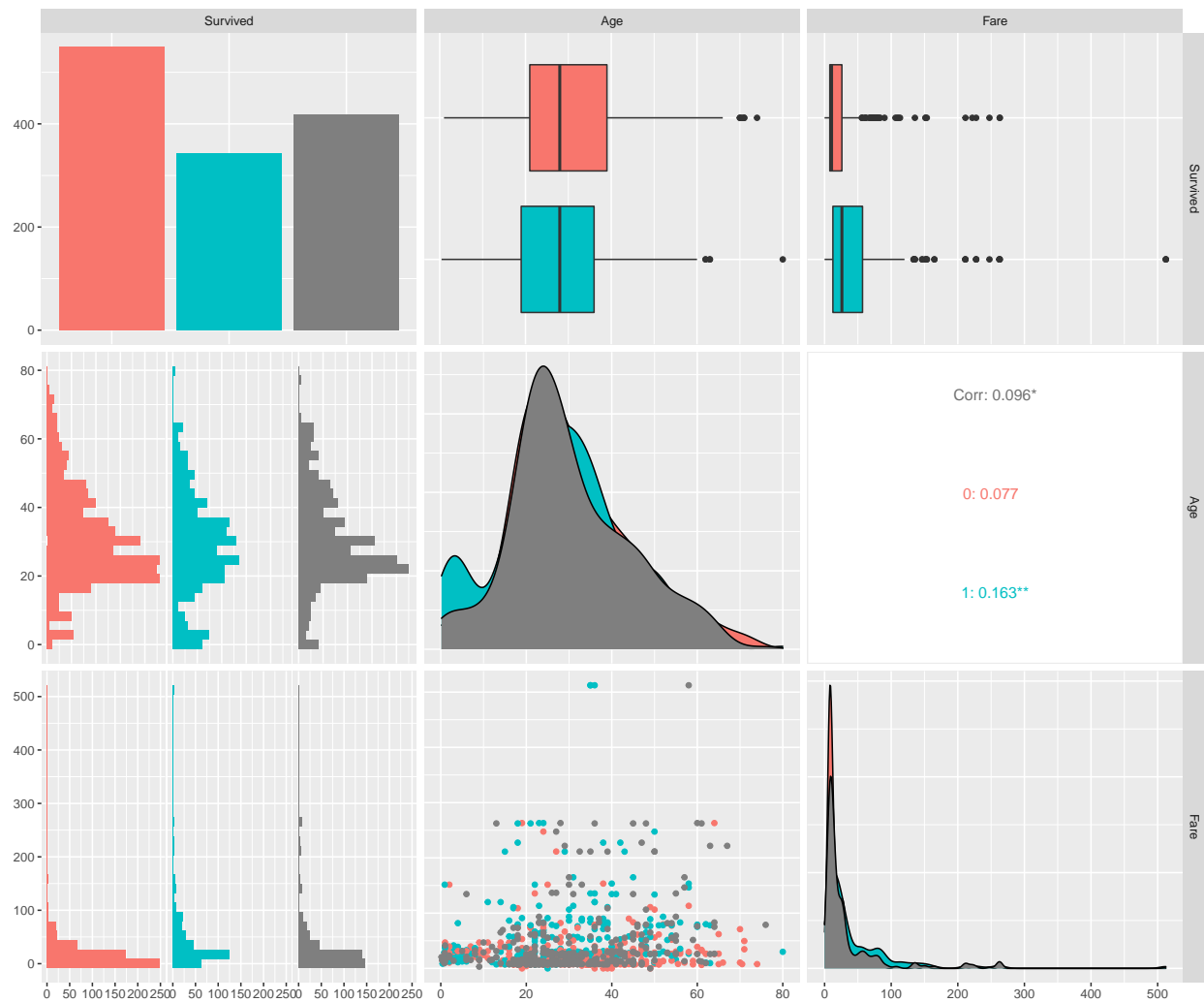
```



```

df %>%
  select(Survived, where(is.numeric), -PassengerId) %>%
  ggpairs(aes(color=Survived))

```



3.1 Selección de los datos de interes

La gran mayoría de los atributos presentes se corresponden con las características que representan a todos los tripulantes del Titanic, por lo que será conveniente tenerlos en consideración durante la realización de los análisis. Sin embargo, podemos prescindir de los campos (PassengerId, Name, first_name, Cabin y Ticket) dado que no son atributos a los que se les ha extraído la información relevante en el paso anterior, por tanto, nos añadirían redundancia a la hora de resolver nuestro problema.

```
# Eliminamos las siguientes columnas
#df$PassengerId <-NULL
#df$Name <-NULL
#df$Cabin <- NULL
```

4 Limpieza de los datos.

4.1 ¿Los datos contienen ceros o elementos vacíos? ¿Cómo gestionarías cada uno de estos casos?

En primer lugar analizamos la cantidad de valores nulos que existen por cada atributo.

```
# Números de valores desconocidos por campo
sapply(df, function(x) sum(is.na(x)))
```

```
## PassengerId    Survived    Pclass      Name first_name title_name
##           0         418         0         0         0         0
##           Sex         Age      SibSp      Parch      Ticket      Fare
##           0         263         0         0         0         0
##           Cabin Embarked      Child  n_ticket  n_name      deck
##          1014         0         263         0         0         0
##           Crew
##           0
```

Llegados a este punto debemos decidir cómo manejar estos registros que contienen valores desconocidos para algún campo. Una opción podría ser eliminar esos registros que incluyen este tipo de valores, pero ello supondría desaprovechar información. Por lo que aplicaremos a continuación distintas técnicas para imputar esos valores.

Sustituimos los NA's de la variable Child con los siguientes criterios:

- Asumimos que los viajeros con ticket unipersonal con SibSp==0 (sin hermanos o esposa a bordo) y Parch==0 (sin familia) son niños.
- Asumimos que las personas con title_name=='Mrs' (mujeres casadas) no son niños.
- Asumimos que las personas con title_name=='Master' son niños.
- Asumimos que las personas con SibSp>0 y Parch>0 son niños.

```
df <- df %>%
  mutate(Child = if_else(SibSp == 0 & Parch == 0 & is.na(Child) & n_ticket == 1,
                        FALSE, as.logical(Child))) %>%
  mutate(Child = if_else(title_name == 'Master' & is.na(Child),
                        TRUE, as.logical(Child))) %>%
  mutate(Child = if_else(title_name == 'Mrs' & is.na(Child),
                        FALSE, as.logical(Child))) %>%
  mutate(Child = if_else(SibSp != '0' & Parch != '0' & is.na(Child),
                        TRUE, as.logical(Child))) %>%
  mutate(Child = if_else(is.na(Child), FALSE, as.logical(Child))) %>%
  mutate(Child=factor(Child))
prop.table(table(df$Survived, df$Child, dnn = c('Survived', 'Child')), margin = 2)
```

```
##           Child
## Survived  FALSE    TRUE
##           0 0.6290323 0.4941176
##           1 0.3709677 0.5058824
```

Por último, se empleará un método de imputación de valores basado en la similitud o diferencia entre los registros: la imputación basada en k vecinos más próximos (en inglés, kNN-imputation). La elección de esta alternativa se realiza bajo la hipótesis de que nuestros registros guardan cierta relación. No obstante, es mejor trabajar con datos “aproximados” que, con los propios elementos vacíos, ya que obtendremos análisis con menor margen de error.

```
# Imputación de valores mediante la función kNN() del paquete VIM
df$Age <- kNN(df)$Age
sapply(df, function(x) sum(is.na(x)))
```

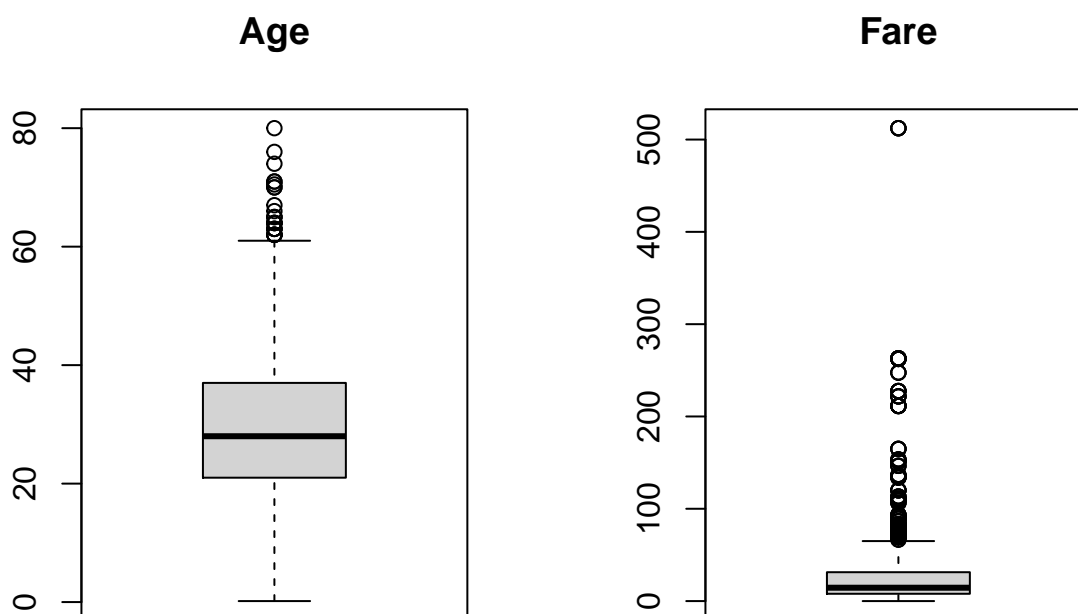
```
## PassengerId    Survived    Pclass      Name first_name title_name
##           0         418         0         0         0         0
##           Sex         Age      SibSp      Parch      Ticket      Fare
##           0         0         0         0         0         0
##           Cabin Embarked      Child  n_ticket  n_name      deck
##          1014         0         0         0         0         0
##           Crew
##           0
```

4.2 Identificación y tratamiento de valores extremos.

Los valores extremos o outliers son aquellos datos que se encuentran muy alejados de la distribución normal de una variable o población. Para identificarlos, podemos hacer uso de dos vías: (1) representar un diagrama de caja por cada variable y ver qué valores distan mucho del rango intercuartílico (la caja) o (2) utilizar la función `boxplots.stats()` de R.

A continuación se muestra los valores atípicos para las variables cuantitativas (Age y Fare) mediante la representación de un diagrama de caja.

```
# Identificación de valores extremos
par(mfrow=c(1,2))
boxplot(df$Age, main= "Age")
boxplot(df$Fare, main= "Fare")
```



Para los resultados de la característica edad (Age) los valores extremos son valores reales y están en un rango normal. No hay edades mayores a 80 ni menores a 0. Por tanto, son valores que pueden darse.

Para los resultados de la característica tarifa (Fare) sí que vemos una mayor discrepancia en la distribución con precios 5 veces más grandes que la media. Estos valores están relacionados con la clase y el número de pasajeros. Por tanto, son valores que pueden darse.

5 Análisis de los datos.

5.1 Selección de los grupos de datos que se quieren analizar/comparar (planificación de los análisis a aplicar).

A continuación, se seleccionan los grupos dentro de nuestro conjunto de datos que pueden resultar interesantes para analizar y/o comparar. En este caso nos interesa analizar las diferencias de supervivencia entre hombres

y mujeres, distintas clases y si son niños o no.

```
# Agrupación por sexo
titanic.male <- df[df$Sex == "male",]
titanic.female <- df[df$Sex == "female",]

# Agrupación por clase
titanic.first <- df[df$Pclass == "1",]
titanic.second <- df[df$Pclass == "2",]
titanic.third <- df[df$Pclass == "3",]

# Agrupación por rango de edad
titanic.child <- df[df$Child == TRUE,]
titanic.adult <- df[df$Child == FALSE,]
```

5.2 Comprobación de la normalidad y homogeneidad de la varianza.

de una población distribuida normalmente, utilizaremos la prueba de normalidad de Shapiro-Wilk, al tratarse de uno de los métodos más potentes para contrastar la normalidad. Asumiendo como hipótesis nula que la población está distribuida normalmente, si el p-valor es menor al nivel de significancia, generalmente $\alpha = 0.05$, entonces la hipótesis nula es rechazada y se concluye que los datos no cuentan con una distribución normal.

```
# Shapiro-Wilk test para Age
shapiro.test(df$Age)

##
##  Shapiro-Wilk normality test
##
## data:  df$Age
## W = 0.97635, p-value = 7.966e-14

# Shapiro-Wilk test para Fare
shapiro.test(df$Fare)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  df$Fare
## W = 0.52766, p-value < 2.2e-16
```

Dado los resultados anteriores, se observa que para la variable Age y la variable Fare consideradas sus correspondientes p-valores son inferiores al nivel de significación $\alpha = 0.05$. Por tanto, rechazamos la hipótesis nula y concluimos con un 95% de confianza que los datos no se distribuyen normalmente.

Seguidamente, pasamos a estudiar la homogeneidad de varianzas, mediante la aplicación de un test de Fligner-Killeen. Se trata de un test no paramétrico que compara las varianzas basándose en la mediana. Para realizar el test de Fligner-Killeen se utiliza la función *fligner.test()*

A continuación, se muestra la aplicación del test Fligner-Killeen para la característica cuantitativa Edad (Age) en función de la Supervivencia (Survived):

```
# Test Fligner-Killeen para Age
fligner.test(Age ~ Survived, data = df)

##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  Age by Survived
## Fligner-Killeen: med chi-squared = 3.3483, df = 1, p-value = 0.06728
```

Puesto que obtenemos un p-valor superior al nivel de significación ($\alpha = 0.05$), aceptamos la hipótesis nula, es decir, de que las varianzas de ambas muestras son homogéneas.

A continuación, se muestra la aplicación del test Fligner-Killeen para característica precio (Fare) en función de la Supervivencia (Survived):

```
# Test fligner para SibSp
fligner.test(Fare ~ Survived, data = df)
```

```
##
## Fligner-Killeen test of homogeneity of variances
##
## data: Fare by Survived
## Fligner-Killeen:med chi-squared = 96.253, df = 1, p-value < 2.2e-16
```

Puesto que obtenemos un p-valor inferior al nivel de significación, rechazamos la hipótesis nula (H_0), es decir, podemos concluir que las varianzas son significativamente diferentes.

5.3 Aplicación de pruebas estadísticas para comparar los grupos de datos. En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes.

Para determinar la asociación de la variable `Survived` con el resto de variables categóricas utilizamos el test χ^2 y el coeficiente CramerV. Según el test χ^2 todas las variables son significativas con un 95% de confianza. La bibliografía indica que un valor de CramerV > 0.1 es significativo. Las variables `Child` y `Crew` no lo cumplen por poco.

Para determinar la asociación de la variable `Survived` con las variables numéricas utilizaremos el t.test (H_0 : la media del subset con `Survived==1` es igual a la media con `Survived==0`). En el caso de las variables `Age` y `Fare` podemos descartar que tengan la misma media con un 95% de confianza.

Ajustamos un modelo GLM y minimizamos el Aic con la función `step` (búsqueda de las variables más significativas).

Los dos primeros métodos identifican la asociación de cada variable individualmente. GLM tiene en cuenta la colinearidad de las variables.

Las más significativas son: `Pclass`, `title_name`, `Age` y `n_ticket`.

```
df %>%
  select(Survived, where(is.factor)) %>%
  filter(!is.na(Survived)) %>%
  pivot_longer(~Survived) %>%
  group_by(name) %>%
  summarize(cramerv=CramerV(x=Survived, y=factor(value)),
            chisq.pvalue=chisq.test(Survived, factor(value))$p.value) %>%
  mutate(`signif 95%`=chisq.pvalue < 0.05) %>%
  arrange(chisq.pvalue) %>%
  kable(format='latex', digits=4, caption='CramerV y chisq.test') %>%
  kable_styling(full_width = F, latex_options = "HOLD_position")
```

Table 1: CramerV y chisq.test

name	cramerv	chisq.pvalue	signif 95%
title_name	0.5708	0.0000	TRUE
Sex	0.5434	0.0000	TRUE
Pclass	0.3398	0.0000	TRUE
n_ticket	0.3388	0.0000	TRUE
deck	0.3336	0.0000	TRUE
SibSp	0.2045	0.0000	TRUE
Embarked	0.1707	0.0000	TRUE
n_name	0.1979	0.0000	TRUE
Parch	0.1770	0.0001	TRUE
Child	0.0815	0.0206	TRUE
Crew	0.0853	0.0226	TRUE

```
df %>%
  select(Survived, where(is.numeric), -PassengerId) %>%
  filter(!is.na(Survived)) %>%
  pivot_longer(-Survived) %>%
  group_by(name, Survived) %>%
  summarize(value_list = list(value)) %>%
  pivot_wider(names_from=Survived, values_from=value_list) %>%
  mutate(vartest.pval=var.test(unlist(`0`), unlist(`1`))$p.val) %>%
  mutate(ttest.pval=t.test(unlist(`0`), unlist(`1`),
    var.equal=vartest.pval>0.05)$p.val) %>%
  mutate(`signif 95%`=ttest.pval < 0.05) %>%
  select(`0`, `1`) %>%
  kable(format='latex', digits=4, caption='t.test') %>%
  kable_styling(full_width = F, latex_options = "HOLD_position")
```

Table 2: t.test

name	vartest.pval	ttest.pval	signif 95%
Age	0.2075	0.038	TRUE
Fare	0.0000	0.000	TRUE

```
# GLM
df0 <- df %>%
  select(-PassengerId, -Name, -first_name, -Ticket, -Cabin) %>%
  na.omit()
fit <- glm(data=df0, Survived ~ ., family=binomial(link = "logit"))
step_fit <- step(fit, direction='both', trace=0)
summary(step_fit)

##
## Call:
## glm(formula = Survived ~ Pclass + title_name + Sex + Age + SibSp +
##     Parch + Fare + n_ticket + n_name, family = binomial(link = "logit"),
##     data = df0)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.2614  -0.4672  -0.2992   0.4569   2.6211
##
## Coefficients: (1 not defined because of singularities)
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   23.99703  3956.18050   0.006 0.995160
## Pclass2       -1.44944    0.36774  -3.941 8.10e-05 ***
## Pclass3       -2.57274    0.38580  -6.669 2.58e-11 ***
## title_nameMiss -19.73389  3956.18049  -0.005 0.996020
```

```
## title_nameMr      -4.66066      0.83524    -5.580 2.40e-08 ***
## title_nameMrs     -18.88969 3956.18050    -0.005 0.996190
## Sexmale           -18.21254 3956.18041    -0.005 0.996327
## Age               -0.04073      0.01075    -3.789 0.000151 ***
## SibSp1             0.29142      0.35415      0.823 0.410579
## SibSp2             1.22251      0.71822      1.702 0.088730 .
## SibSp3             0.32039      1.13234      0.283 0.777220
## SibSp4            -18.17148   839.89941    -0.022 0.982739
## SibSp5            -44.08829 1695.15924    -0.026 0.979251
## SibSp8            -36.51632 1440.36402    -0.025 0.979774
## Parch1            -0.05667      0.40483    -0.140 0.888669
## Parch2             0.42621      0.62015      0.687 0.491912
## Parch3             1.59415      1.63027      0.978 0.328153
## Parch4            -13.02690 1625.15898    -0.008 0.993604
## Parch5            -15.88408   839.90167    -0.019 0.984911
## Parch6            -42.62142 4044.35354    -0.011 0.991592
## Fare              0.00870      0.00443      1.964 0.049529 *
## n_ticket2         -0.63038      0.36189    -1.742 0.081524 .
## n_ticket3         -0.22091      0.50212    -0.440 0.659969
## n_ticket4         -0.19257      0.77828    -0.247 0.804575
## n_ticket5         -2.77100      0.90102    -3.075 0.002102 **
## n_ticket6         -4.85743      1.36294    -3.564 0.000365 ***
## n_ticket7         -4.32455      1.88808    -2.290 0.021995 *
## n_ticket8          3.21451      0.90126      3.567 0.000362 ***
## n_ticket11         NA          NA          NA          NA
## n_name2           -0.40455      0.37439    -1.081 0.279894
## n_name3           -0.71893      0.46918    -1.532 0.125439
## n_name4           -1.26050      0.58802    -2.144 0.032062 *
## n_name5           -1.89627      1.44918    -1.309 0.190697
## n_name6           -0.94818      1.06019    -0.894 0.371134
## n_name8            20.35827   839.89981      0.024 0.980662
## n_name11           16.97264   839.89768      0.020 0.983877
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 1186.66  on 890  degrees of freedom
## Residual deviance:  642.17  on 856  degrees of freedom
## AIC: 712.17
##
## Number of Fisher Scoring iterations: 16
```

6 Representación de los resultados a partir de tablas y gráficas.

7 Resolución del problema. A partir de los resultados obtenidos, ¿cuáles son las conclusiones? ¿Los resultados permiten responder al problema?

Claramente la supervivencia en el naufragio del Titanic se priorizó a las mujeres, los niños y a los pasajeros de primera clase.

El modelo SVM ajustado para participar en el concurso de Kaggle consigue una precisión aceptable (determinada por cross validation de 10 grupos).

Por último, generamos el fichero 'My_submission.csv' que mandaremos a Kaggle.

```
df0 <- df %>%
  select(-PassengerId, -Name, -Cabin) %>%
  rowwise() %>%
  mutate(Age=if_else(is.na(Age), if_else(as.logical(Child), runif(1, 0, 12), runif(1, 13, 60)), Age)) %>%
  mutate(first_name=factor(first_name)) %>%
  mutate(Ticket=factor(Ticket)) %>%
  na.omit()
```

```

set.seed(200560)
train_control <- trainControl(method="cv", number=10)
# Fit the model
svm1 <- train(Survived ~ ., data = df0, method = 'svmLinear', trControl = train_control,
              tuneGrid = expand.grid(C = seq(0, 2, length = 10)))
svm1

## Support Vector Machines with Linear Kernel
##
## 891 samples
## 15 predictor
## 2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 802, 801, 802, 802, 803, 802, ...
## Resampling results across tuning parameters:
##
##  C          Accuracy  Kappa
##  0.0000000      NaN      NaN
##  0.2222222  0.8317058  0.6393967
##  0.4444444  0.8463509  0.6722825
##  0.6666667  0.8474370  0.6774003
##  0.8888889  0.8463512  0.6749678
##  1.1111111  0.8452276  0.6724751
##  1.3333333  0.8463387  0.6750447
##  1.5555556  0.8452151  0.6728147
##  1.7777778  0.8440787  0.6701865
##  2.0000000  0.8440787  0.6701865
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was C = 0.6666667.

confusionMatrix(predict(svm1), df0$Survived)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0    1
##           0 546   7
##           1   3 335
##
##           Accuracy : 0.9888
##           95% CI : (0.9795, 0.9946)
##           No Information Rate : 0.6162
##           P-Value [Acc > NIR] : <2e-16
##
##           Kappa : 0.9762
##
##  Mcnemar's Test P-Value : 0.3428
##
##           Sensitivity : 0.9945
##           Specificity : 0.9795
##           Pos Pred Value : 0.9873
##           Neg Pred Value : 0.9911
##           Prevalence : 0.6162
##           Detection Rate : 0.6128
##           Detection Prevalence : 0.6207
##           Balanced Accuracy : 0.9870
##
##           'Positive' Class : 0
##

df1 <- df %>%
  rowwise() %>%
  mutate(Age=if_else(is.na(Age), if_else(as.logical(Child), runif(1, 0, 12), runif(1, 13, 60)), Age)) %>%
  filter(is.na(Survived))
df1$Survived <- predict(svm1, df1)

```

```
write_csv((df1 %>% select(PassengerId, Survived)), file='My_submission.csv')
```

8 Contribuyentes

Contribuciones	Firma
Investigación previa	
Redacción de las respuestas	
Desarrollo código	

9 References