Minería de datos: PEC3 - Clasificación con árboles de decisión

Autor: Nombre estudiante

Diciembre 2023

Contents

Recursos básicos

Esta Prueba de Evaluación Continuada (PEC) cubre principalmente el material didáctico de modelos supervisados y evaluación de modelos.

Complementarios:

- Material docente "Creación y evaluación de modelos no supervisados" proporcionado por la UOC.
- Fichero titanic.csv.

- R package C5.0 (Decision Trees and Rule-Based Models): https://cran.r-project.org/web/packages/C50/index.html
- Fichero de "German Credit": credit.csv (se obtuvo de https://www.kaggle.com/shravan3273/credit-approval)

La descripción de las variables se puede ver en https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data)

La variable "default" es el target siendo 1 = "No default" y 2 = "Default". Se deben utilizar estos datos para la realización de los ejercicios.

Ejemplo ilustrativo

En este ejercicio vamos a seguir los pasos del ciclo de vida de un proyecto de minería de datos, para el caso de un algoritmo de clasificación usaremos un árbol de decisión, **que es el algoritmo supervisado que vamos a tratar en esta asignatura**. Primero y a modo de ejemplo sencillo lo haremos con el archivo titanic.csv, que se encuentra adjunto en el aula. Este archivo contiene un registro por cada pasajero que viajaba en el Titanic. En las variables se caracteriza si era hombre o mujer, adulto o menor (niño), en qué categoría viajaba o si era miembro de la tripulación. Se mostrará un ejemplo sencillo de solución con estos datos pero los alumnos deberéis responder a las preguntas de la rúbrica para otro conjunto: German Credit. Para este conjunto, tomaréis como referencia la variable "default" que indica el impago de créditos.

Objetivos:

- Estudiar los datos, por ejemplo: ¿Número de registros del fichero? ¿Distribuciones de valores por variables? ¿Hay campos mal informados o vacíos?
- Preparar los datos. En este caso ya están en el formato correcto y no es necesario discretizar ni generar atributos nuevos. Hay que elegir cuáles son las variables que se utilizarán para construir el modelo y cuál es la variable que clasifica. En este caso la variable por la que clasificaremos es el campo de si el pasajero sobrevivia o no.
- Instalar, si es necesario, el paquete C5.0 Se trata de una implementación más moderna del algoritmo ID3 de Quinlan. Tiene los principios teóricos del ID3 más la poda automática. Con este paquete generar un modelo de minería.
- ¿Cuál es la calidad del modelo?
- Generar el árbol gráfico.
- Generar y extraer las reglas del modelo.
- En función del modelo, el árbol y las reglas: ¿Cuál es el conocimiento que obtenemos?
- Probar el modelo generado presentándole nuevos registros. ¿Clasifica suficientemente bien?

A continuación, se plantean los puntos a realizar en la PEC 3 y, tomando como ejemplo el conjunto de datos de Titanic, se obtendrán, a modo de ejemplo, algunos resultados que pretender servir a modo de inspiración para los estudiantes. Los estudiantes deberán utilizar el conjunto de datos de "German Credit Data" que se pueden conseguir en este enlace: https://www.kaggle.com/shravan3273/credit-approval

Este recurso puede ser útil para profundizar sobre el paquete IML: https://uc-r.github.io/iml-pkg

Revisión de los datos, extracción visual de información y preparación de los datos

Carga de los datos:

```
data<-read.csv("./titanic-1.csv",header=T,sep=",")
attach(data)
```

Análisis inicial

Empezaremos haciendo un breve análisis de los datos ya que nos interesa tener una idea general de los datos que disponemos.

Exploración de la base de datos

Primero calcularemos las dimensiones de nuestra base de datos y analizaremos qué tipos de atributos tenemos.

Para empezar, calculamos las dimensiones de la base de datos mediante la función dim(). Obtenemos que disponemos de 2201 registros o pasajeros (filas) y 4 variables (columnas).

```
dim(data)
```

```
## [1] 2201    4
```

¿Cuáles son esas variables? Gracias a la función str() sabemos que las cuatro variables son categóricas o discretas, es decir, toman valores en un conjunto finito. La variable CLASS hace referencia a la clase en la que viajaban los pasajeros (1^a, 2^a, 3^a o crew), AGE determina si era adulto o niño (Adulto o Menor), la variable SEX si era hombre o mujer (Hombre o Mujer) y la última variable (SURVIVED) informa si el pasajero murió o sobrevivió en el accidente (Muere o Sobrevive).

```
str(data)
```

```
## 'data.frame':    2201 obs. of  4 variables:
##  $ CLASS   : chr  "1a" "1a" "1a" "1a" ...
##  $ AGE     : chr  "Adulto" "Adulto" "Adulto" "Adulto" ...
##  $ SEX     : chr  "Hombre" "Hombre" "Hombre" "Hombre" ...
##  $ SURVIVED: chr  "Sobrevive" "Sobrevive" "Sobrevive" "Sobrevive" ...
```

Vemos que las variables están definidas como carácter, así que las transformamos a tipo factor.

```
data[] <- lapply(data, factor)
str(data)
```

```
## 'data.frame':    2201 obs. of  4 variables:
##  $ CLASS   : Factor w/ 4 levels "1a","2a","3a",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ AGE     : Factor w/ 2 levels "Adulto","Menor": 1 1 1 1 1 1 1 1 1 1 ...
##  $ SEX     : Factor w/ 2 levels "Hombre","Mujer": 1 1 1 1 1 1 1 1 1 1 ...
##  $ SURVIVED: Factor w/ 2 levels "Muere","Sobrevive": 2 2 2 2 2 2 2 2 2 2 ...
```

Es de gran interés saber si tenemos muchos valores nulos (campos vacíos) y la distribución de valores por variables. Es por ello recomendable empezar el análisis con una visión general de las variables. Mostraremos para cada atributo la cantidad de valores perdidos mediante la función summary.

```
summary(data)
```

```
##   CLASS         AGE           SEX            SURVIVED
##  1a  :325   Adulto:2092   Hombre:1731   Muere    :1490
##  2a  :285   Menor : 109   Mujer : 470   Sobrevive: 711
##  3a  :706
##  crew:885
```

Como parte de la preparación de los datos, miraremos si hay valores missing.

```
missing <- data[is.na(data),]
dim(missing)
```

```
## [1] 0 4
```

Observamos fácilmente que no hay valores missing y, por tanto, no deberemos preparar los datos en este sentido. En caso de haberlos, habría que tomar decisiones para tratar los datos adecuadamente.

Disponemos por tanto de un data frame formado por cuatro variables categóricas sin valores nulos.

Visualización

Para un conocimiento mayor sobre los datos, tenemos a nuestro alcance unas herramientas muy valiosas: las herramientas de visualización. Para dichas visualizaciones, haremos uso de los paquetes ggplot2, gridExtra y grid de R.

```
if(!require(ggplot2)){
    install.packages('ggplot2', repos='http://cran.us.r-project.org')
    library(ggplot2)
}
```

```
## Loading required package: ggplot2
```

```
if(!require(ggpubr)){
    install.packages('ggpubr', repos='http://cran.us.r-project.org')
    library(ggpubr)
}
```

```
## Loading required package: ggpubr
```

```
if(!require(grid)){
    install.packages('grid', repos='http://cran.us.r-project.org')
    library(grid)
}
```

```
## Loading required package: grid
```

```
if(!require(gridExtra)){
    install.packages('gridExtra', repos='http://cran.us.r-project.org')
    library(gridExtra)
}
```

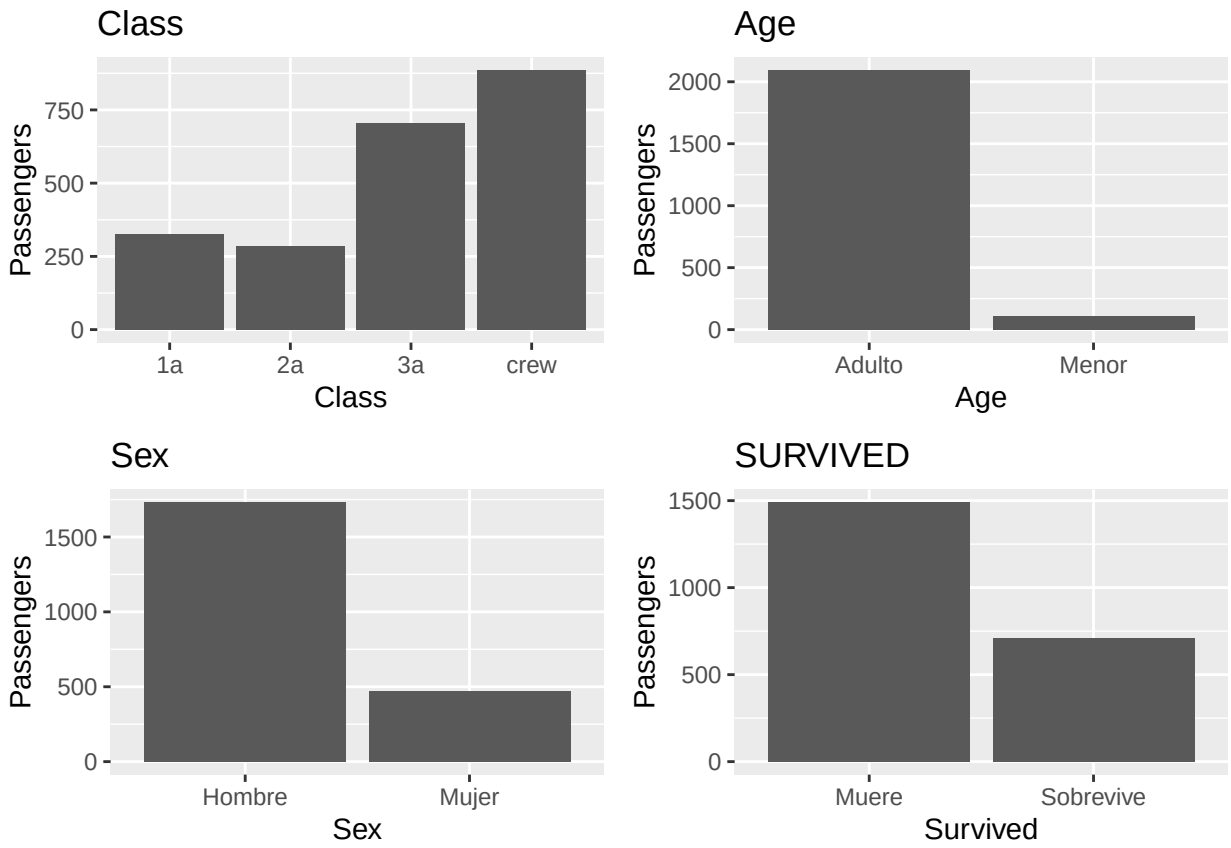
```
## Loading required package: gridExtra
```

```
if(!require(C50)){
    install.packages('C50', repos='http://cran.us.r-project.org')
    library(C50)
}
```

```
## Loading required package: C50
```

Siempre es importante analizar los datos que tenemos ya que las conclusiones dependerán de las características de la muestra.

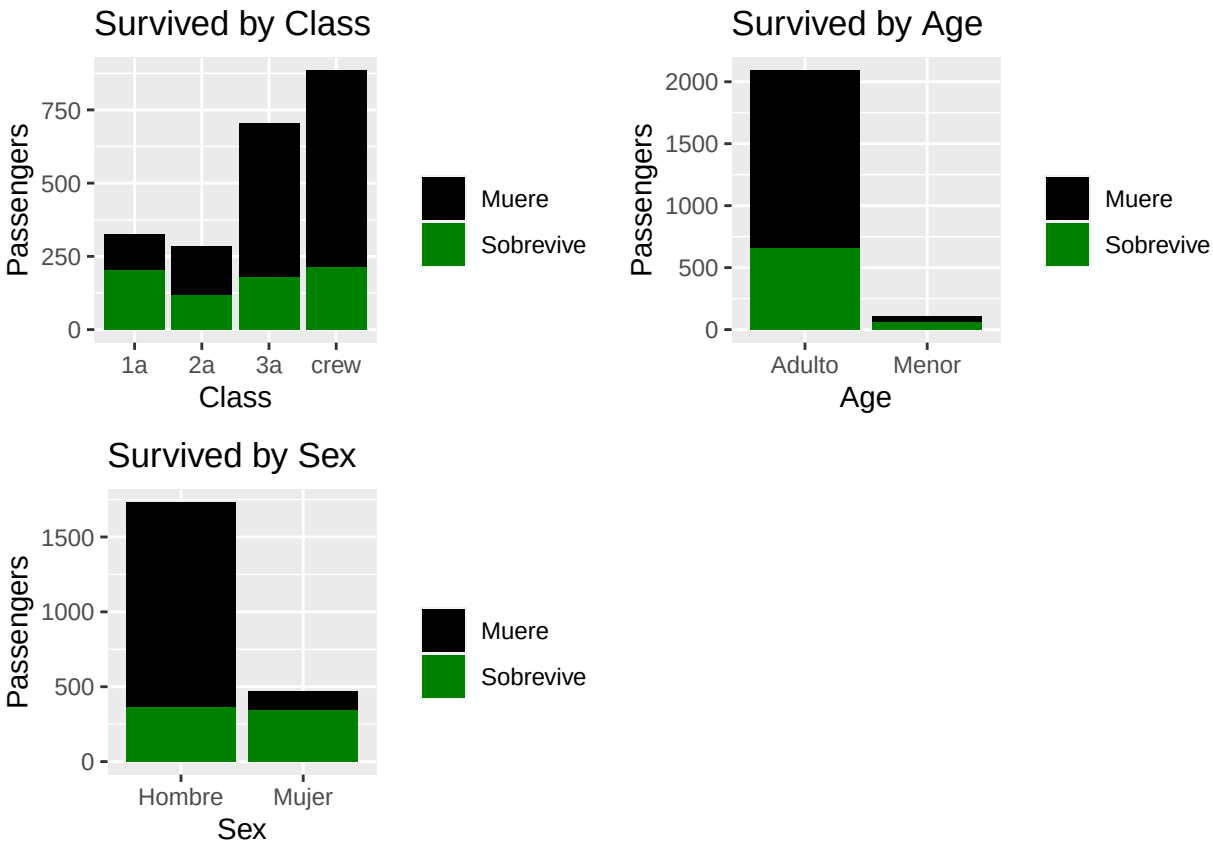
```
grid.newpage()
plotbyClass<-ggplot(data,aes(CLASS))+geom_bar() +labs(x="Class", y="Passengers")+ guides(fill=guide_leg
plotbyAge<-ggplot(data,aes(AGE))+geom_bar() +labs(x="Age", y="Passengers")+ guides(fill=guide_legend(ti
plotbySex<-ggplot(data,aes(SEX))+geom_bar() +labs(x="Sex", y="Passengers")+ guides(fill=guide_legend(ti
plotbySurvived<-ggplot(data,aes(SURVIVED))+geom_bar() +labs(x="Survived", y="Passengers")+ guides(fill=
grid.arrange(plotbyClass,plotbyAge,plotbySex,plotbySurvived,ncol=2)
```



Claramente vemos cómo es la muestra analizando la distribución de las variables disponibles. De cara a los informes, es mucho más interesante esta información que la obtenida en summary, que se puede usar para complementar.

Nos interesa describir la relación entre la supervivencia y cada uno de las variables mencionadas anteriormente. Para ello, por un lado graficaremos mediante diagramas de barras la cantidad de muertos y supervivientes según la clase en la que viajaban, la edad o el sexo. Por otro lado, para obtener los datos que estamos graficando utilizaremos el comando table para dos variables que nos proporciona una tabla de contingencia.

```
grid.newpage()
plotbyClass<-ggplot(data,aes(CLASS,fill=SURVIVED))+geom_bar() +labs(x="Class", y="Passengers")+ guides(
plotbyAge<-ggplot(data,aes(AGE,fill=SURVIVED))+geom_bar() +labs(x="Age", y="Passengers")+ guides(fill=gu
plotbySex<-ggplot(data,aes(SEX,fill=SURVIVED))+geom_bar() +labs(x="Sex", y="Passengers")+ guides(fill=gu
grid.arrange(plotbyClass,plotbyAge,plotbySex,ncol=2)
```



De estos gráficos obtenemos información muy valiosa que complementamos con las tablas de contingencia (listadas abajo). Por un lado, la cantidad de pasajeros que sobrevivieron es similar en hombres y mujeres (hombres: 367 y mujeres 344). No, en cambio, si tenemos en cuenta el porcentaje respecto a su sexo. Es decir, pese a que la cantidad de mujeres y hombres que sobrevivieron es pareja, viajaban más hombres que mujeres (470 mujeres y 1731 hombres), por lo tanto, la tasa de muerte en hombres es muchísimo mayor (el 78,79% de los hombres murieron mientras que en mujeres ese porcentaje baja a 26,8%).

En cuanto a la clase en la que viajaban, los pasajeros que viajaban en primera clase fueron los únicos que el porcentaje de supervivencia era mayor que el de mortalidad. El 62,46% de los viajeros de primera clase sobrevivió, el 41,4% de los que viajaban en segunda clase mientras que de los viajeros de tercera y de la tripulación solo sobrevivieron un 25,21% y 23,95% respectivamente. Para finalizar, destacamos que la presencia de pasajeros adultos era mucho mayor que la de los niños (2092 frente a 109) y que la tasa de supervivencia en niños fue mucho mayor (52,29% frente a 31,26%), no podemos obviar, en cambio, que los únicos niños que murieron fueron todos pasajeros de tercera clase (52 niños).

```
tabla_SST <- table(SEX, SURVIVED)
tabla_SST
```

```
##         SURVIVED
## SEX      Muere Sobrevive
##   Hombre  1364       367
##   Mujer    126       344
```

```
prop.table(tabla_SST, margin = 1)
```

```
##         SURVIVED
## SEX          Muere Sobrevive
##   Hombre 0.7879838 0.2120162
##   Mujer  0.2680851 0.7319149
```

```r
tabla_SCT <- table(CLASS,SURVIVED)
tabla_SCT
```

```
##       SURVIVED
## CLASS  Muere Sobrevive
##   1a     122       203
##   2a     167       118
##   3a     528       178
##   crew   673       212
```

```r
prop.table(tabla_SCT, margin = 1)
```

```
##       SURVIVED
## CLASS      Muere Sobrevive
##   1a   0.3753846 0.6246154
##   2a   0.5859649 0.4140351
##   3a   0.7478754 0.2521246
##   crew 0.7604520 0.2395480
```

```r
tabla_SAT <- table(AGE,SURVIVED)
tabla_SAT
```

```
##         SURVIVED
## AGE      Muere Sobrevive
##   Adulto  1438       654
##   Menor     52        57
```

```r
prop.table(tabla_SAT, margin = 1)
```

```
##         SURVIVED
## AGE          Muere Sobrevive
##   Adulto 0.6873805 0.3126195
##   Menor  0.4770642 0.5229358
```

```r
tabla_SAT.byClass <- table(AGE,SURVIVED,CLASS)
tabla_SAT.byClass
```

```
## , , CLASS = 1a
##
##         SURVIVED
## AGE      Muere Sobrevive
##   Adulto   122       197
##   Menor      0         6
##
## , , CLASS = 2a
##
##         SURVIVED
## AGE      Muere Sobrevive
##   Adulto   167        94
##   Menor      0        24
##
## , , CLASS = 3a
##
##         SURVIVED
## AGE      Muere Sobrevive
##   Adulto   476       151
##   Menor     52        27
```

```
##
## , , CLASS = crew
##
##          SURVIVED
## AGE       Muere Sobrevive
##   Adulto   673       212
##   Menor      0         0
```

Test estadísticos de significancia

Los resultados anteriores muestran los datos de forma descriptiva, podemos añadir algún test estadístico para validar el grado de significancia de la relación. La librería "DescTools" nos permite instalarlo fácilmente.

```
if(!require(DescTools)){
    install.packages('DescTools', repos='http://cran.us.r-project.org')
    library(DescTools)
}
```

```
## Loading required package: DescTools
```

```
Phi(tabla_SST)
```

```
## [1] 0.4556048
```

```
CramerV(tabla_SST)
```

```
## [1] 0.4556048
```

```
Phi(tabla_SAT)
```

```
## [1] 0.09757511
```

```
CramerV(tabla_SAT)
```

```
## [1] 0.09757511
```

```
Phi(tabla_SCT)
```

```
## [1] 0.2941201
```

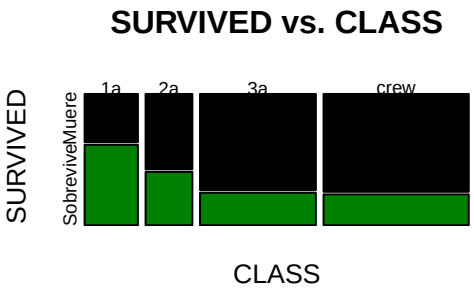
```
CramerV(tabla_SCT)
```

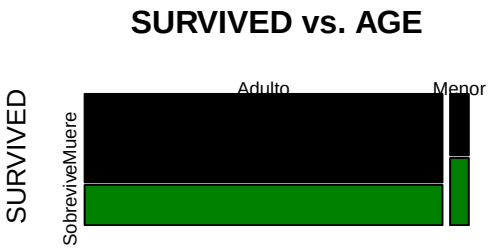
```
## [1] 0.2941201
```

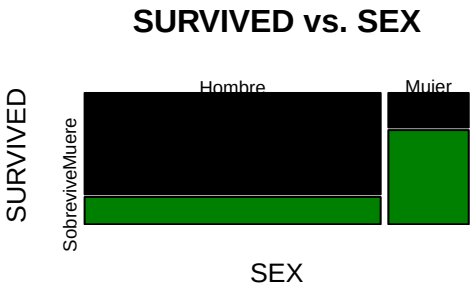
Valores de la V de Cramér (https://en.wikipedia.org/wiki/Cramér%27s_V) y Phi (https://en.wikipedia.org/wiki/Phi_coefficient) entre 0.1 y 0.3 nos indican que la asociación estadística es baja, y entre 0.3 y 0.5 se puede considerar una asociación media. Finalmente, si los valores fueran superiores a 0.5 (no es el caso), la asociación estadística entre las variables sería alta. Como se puede apreciar, los valores de Phi y V coinciden. Esto ocurre en el contexto de analizar tablas de contingencia 2x2.

Una alternativa interesante a las barras de diagramas, es el plot de las tablas de contingencia. Obtenemos la misma información pero para algunos receptores puede resultar más visual.

```
par(mfrow=c(2,2))
plot(tabla_SCT, col = c("black","#008000"), main = "SURVIVED vs. CLASS")
plot(tabla_SAT, col = c("black","#008000"), main = "SURVIVED vs. AGE")
plot(tabla_SST, col = c("black","#008000"), main = "SURVIVED vs. SEX")
```







Nuestro objetivo es crear un árbol de decisión que permita analizar qué tipo de pasajero del Titanic tenía probabilidades de sobrevivir o no. Por lo tanto, la variable por la que clasificaremos es el campo de si el pasajero sobrevivió o no. De todas maneras, al imprimir las primeras (con head) y últimas 10 (con tail) filas nos damos cuenta de que los datos están ordenados.

```
head(data,10)
```

```
##    CLASS    AGE    SEX  SURVIVED
## 1     1a Adulto Hombre Sobrevive
## 2     1a Adulto Hombre Sobrevive
## 3     1a Adulto Hombre Sobrevive
## 4     1a Adulto Hombre Sobrevive
## 5     1a Adulto Hombre Sobrevive
## 6     1a Adulto Hombre Sobrevive
## 7     1a Adulto Hombre Sobrevive
## 8     1a Adulto Hombre Sobrevive
## 9     1a Adulto Hombre Sobrevive
## 10    1a Adulto Hombre Sobrevive
```

```
tail(data,10)
```

```
##      CLASS    AGE   SEX  SURVIVED
## 2192  crew Adulto Mujer Sobrevive
## 2193  crew Adulto Mujer Sobrevive
## 2194  crew Adulto Mujer Sobrevive
## 2195  crew Adulto Mujer Sobrevive
## 2196  crew Adulto Mujer Sobrevive
## 2197  crew Adulto Mujer Sobrevive
## 2198  crew Adulto Mujer Sobrevive
## 2199  crew Adulto Mujer     Muere
## 2200  crew Adulto Mujer     Muere
## 2201  crew Adulto Mujer     Muere
```

Preparación de los datos para el modelo

Para la futura evaluación del árbol de decisión, es necesario dividir el conjunto de datos en un conjunto de entrenamiento y un conjunto de prueba. El conjunto de entrenamiento es el subconjunto del conjunto original de datos utilizado para construir un primer modelo; y el conjunto de prueba, el subconjunto del conjunto original de datos utilizado para evaluar la calidad del modelo.

Lo más correcto será utilizar un conjunto de datos diferente del que utilizamos para construir el árbol, es decir, un conjunto diferente del de entrenamiento. No hay ninguna proporción fijada con respecto al número relativo de componentes de cada subconjunto, pero la más utilizada acostumbra a ser 2/3 para el conjunto de entrenamiento y 1/3, para el conjunto de prueba.

La variable por la que clasificaremos es el campo de si el pasajero sobrevivió o no, que está en la cuarta columna. De esta forma, tendremos un conjunto de datos para el entrenamiento y uno para la validación

```
set.seed(666)
y <- data[,4]
X <- data[,1:3]
```

De forma dinámica podemos definir una forma de separar los datos en función de un parámetro. Así, definimos un parámetro que controla el split de forma dinámica en el test.

```
split_prop <- 3
indexes = sample(1:nrow(data), size=floor(((split_prop-1)/split_prop)*nrow(data)))
trainX<-X[indexes,]
trainy<-y[indexes]
testX<-X[-indexes,]
testy<-y[-indexes]
```

Después de una extracción aleatoria de casos es altamente recomendable efectuar un análisis de datos mínimo para asegurarnos de no obtener clasificadores sesgados por los valores que contiene cada muestra. En este caso, verificaremos que la proporción del supervivientes es más o menos constante en los dos conjuntos:

```
summary(trainX);
```

```
##   CLASS         AGE           SEX
##  1a  :208   Adulto:1395   Hombre:1153
##  2a  :185   Menor :  72   Mujer : 314
##  3a  :477
##  crew:597
```

```
summary(trainy)
```

```
##     Muere Sobrevive
##       997       470
```

```
summary(testX)
```

```
##   CLASS         AGE          SEX
##  1a  :117   Adulto:697   Hombre:578
##  2a  :100   Menor : 37   Mujer :156
##  3a  :229
##  crew:288
```

```
summary(testy)
```

```
##     Muere Sobrevive
##       493       241
```

Verificamos fácilmente que no hay diferencias graves que puedan sesgar las conclusiones.

Creación del modelo, calidad del modelo y extracción de reglas

Se crea el árbol de decisión usando los datos de entrenamiento (no hay que olvidar que la variable outcome es de tipo factor):

```
trainy <-  as.factor(trainy)
model <- C50::C5.0(trainX, trainy,rules=TRUE )
summary(model)
```

```
##
## Call:
## C5.0.default(x = trainX, y = trainy, rules = TRUE)
##
##
## C5.0 [Release 2.07 GPL Edition]      Tue Dec 19 20:10:27 2023
## -------------------------------
##
## Class specified by attribute `outcome'
##
## Read 1467 cases (4 attributes) from undefined.data
##
## Rules:
##
## Rule 1: (1153/243, lift 1.2)
##  SEX = Hombre
##  ->  class Muere  [0.789]
##
## Rule 2: (477/123, lift 1.1)
##  CLASS = 3a
##  ->  class Muere  [0.741]
##
## Rule 3: (178/15, lift 2.8)
##  CLASS in {1a, 2a, crew}
##  SEX = Mujer
##  ->  class Sobrevive  [0.911]
##
## Default class: Muere
##
##
## Evaluation on training data (1467 cases):
##
##          Rules
##    ----------------
##      No      Errors
##
##       3  322(21.9%)   <<
##
##
##     (a)   (b)    <-classified as
##    ----  ----
##     982    15    (a): class Muere
##     307   163    (b): class Sobrevive
##
##
##  Attribute usage:
```

```
##
##   90.73% SEX
##   44.65% CLASS
##
##
## Time: 0.0 secs
```

Errors muestra el número y porcentaje de casos mal clasificados en el subconjunto de entrenamiento. El árbol obtenido clasifica erróneamente 322 de los 1467 casos dados, una tasa de error del 21.9%.

A partir del árbol de decisión de dos hojas que hemos modelado, se pueden extraer las siguientes reglas de decisión (gracias a rules=TRUE podemos imprimir las reglas directamente):

SEX = "Hombre" → Muere. Validez: 78,9%

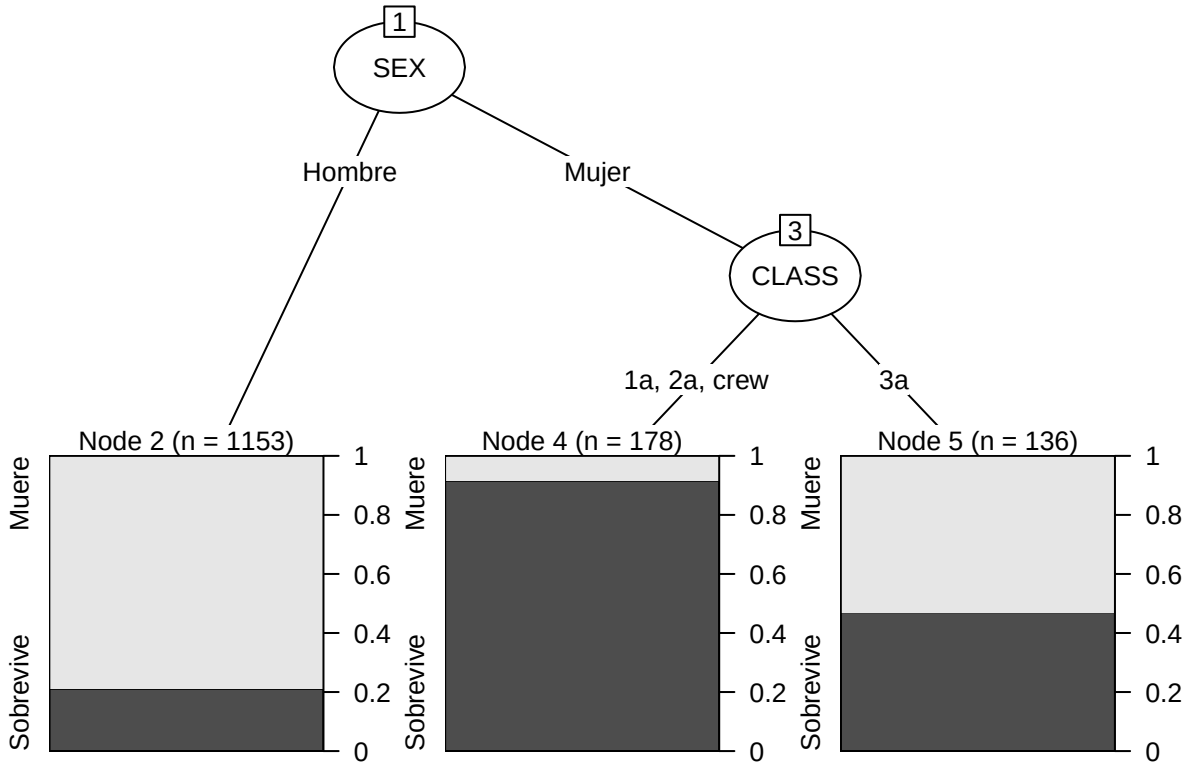
CLASS "3ª" → Muere. Validez: 74,1%

CLASS "1ª", "2ª", "crew" y SEX = "Mujer" → Sobrevive. Validez: 91,1%

Por tanto, podemos concluir que el conocimiento extraído y cruzado con el análisis visual se resume en "las mujeres y los niños primero a excepción de que fueras de 3ª clase".

A continuación, mostramos el árbol obtenido.

```
model <- C50::C5.0(trainX, trainy)
plot(model,gp = gpar(fontsize = 9.5))
```



Validación del modelo con los datos reservados

Una vez tenemos el modelo, podemos comprobar su calidad prediciendo la clase para los datos de prueba que nos hemos reservado al principio.

```
predicted_model <- predict( model, testX, type="class" )
print(sprintf("La precisión del árbol es: %.4f %%",100*sum(predicted_model == testy) / length(predicted_
```

```
## [1] "La precisión del árbol es: 78.8828 %"
```

Cuando hay pocas clases, la calidad de la predicción se puede analizar mediante una matriz de confusión que identifica los tipos de errores cometidos.

```
mat_conf<-table(testy,Predicted=predicted_model)
mat_conf
```

```
##            Predicted
## testy       Muere Sobrevive
##   Muere       488         5
##   Sobrevive   150        91
```

Otra manera de calcular el porcentaje de registros correctamente clasificados usando la matriz de confusión:

```
porcentaje_correct<-100 * sum(diag(mat_conf)) / sum(mat_conf)
print(sprintf("El %% de registros correctamente clasificados es: %.4f %%",porcentaje_correct))
```

```
## [1] "El % de registros correctamente clasificados es: 78.8828 %"
```

Además, tenemos a nuestra disposición el paquete gmodels para obtener información más completa:

```
if(!require(gmodels)){
    install.packages('gmodels', repos='http://cran.us.r-project.org')
    library(gmodels)
}
```

```
## Loading required package: gmodels
```

```
## Registered S3 method overwritten by 'gdata':
##   method         from
##   reorder.factor DescTools
```

```
CrossTable(testy, predicted_model,prop.chisq  = FALSE, prop.c = FALSE, prop.r =FALSE,dnn = c('Reality',
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  734
##
##
##              | Prediction
##      Reality |     Muere | Sobrevive | Row Total |
## -------------|-----------|-----------|-----------|
##        Muere |       488 |         5 |       493 |
##              |     0.665 |     0.007 |           |
## -------------|-----------|-----------|-----------|
##    Sobrevive |       150 |        91 |       241 |
##              |     0.204 |     0.124 |           |
## -------------|-----------|-----------|-----------|
## Column Total |       638 |        96 |       734 |
## -------------|-----------|-----------|-----------|
##
```

##

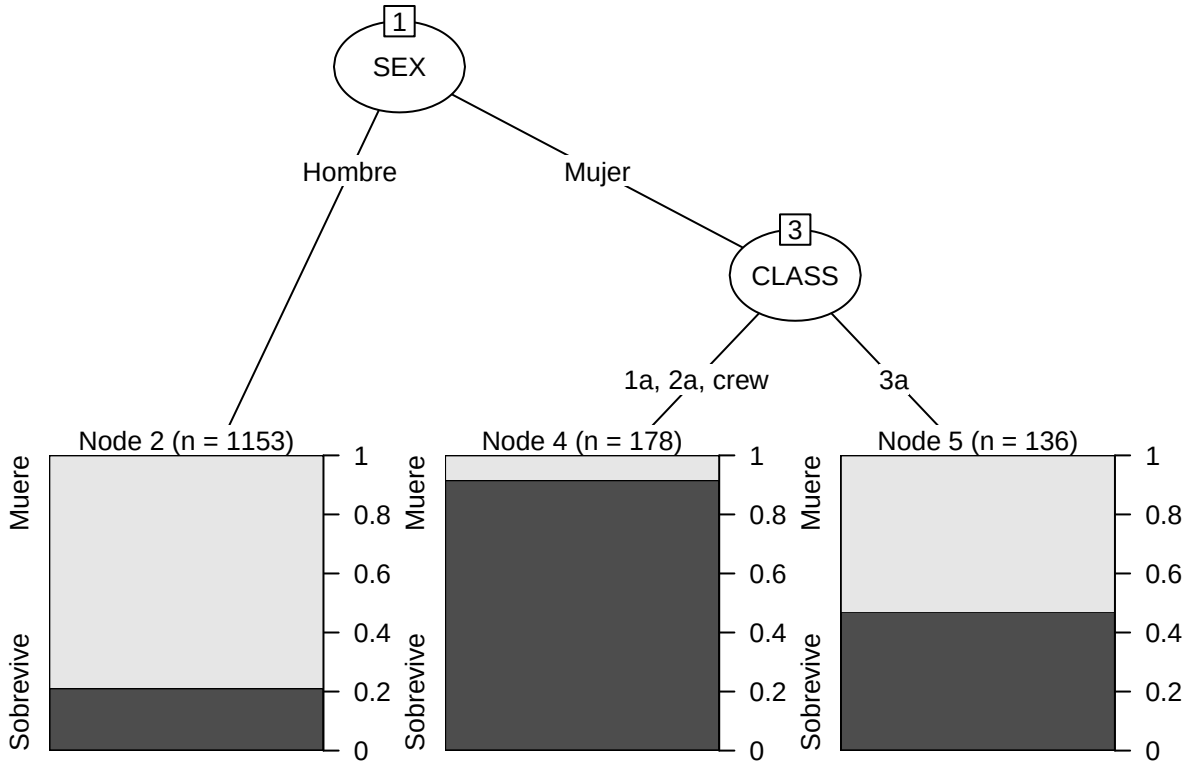
Prueba con una variación u otro enfoque algorítmico

Variaciones del paquete C5.0

En este apartado buscaremos probar con las variaciones que nos ofrece el paquete C5.0 para analizar cómo afectan a la creación de los árboles generados. Existen muchas posibles variaciones con otras funciones que podéis investigar. La idea es seguir con el enfoque de árboles de decisión explorando posibles opciones. Una vez tengamos un método alternativo, debemos analizar cómo se modifica el árbol y cómo afecta a la capacidad predictiva en el conjunto de test.

A continuación, utilizamos otro enfoque para comparar los resultados: incorpora como novedad "adaptative boosting", basado en el trabajo Rob Schapire and Yoav Freund (1999). La idea de esta técnica es generar varios clasificadores, con sus correspondientes arboles de decisión y su ser de reglas. Cuando un nuevo caso va a ser clasificado, cada clasificador vota cual es la clase predicha. Los votos son sumados y determina la clase final.

```
modelo2 <- C50::C5.0(trainX, trainy, trials = 10)
plot(modelo2,gp = gpar(fontsize = 9.5))
```



En este caso, dada la simplicidad del conjunto de ejemplo, no se aprecian diferencias, pero aparecerán en datos de mayor complejidad y modificando el parámetro "trials" se puede intentar mejorar los resultados.

Vemos a continuación cómo son las predicciones del nuevo árbol:

```
predicted_model2 <- predict( modelo2, testX, type="class" )
print(sprintf("La precisión del árbol es: %.4f %%",100*sum(predicted_model2 == testy) / length(predicte
```

```
## [1] "La precisión del árbol es: 75.0681 %"
```

Observamos como se modifica levemente la precisión del modelo a mejor.

```
mat_conf<-table(testy,Predicted=predicted_model2)
mat_conf
```

```
##            Predicted
## testy       Muere Sobrevive
##   Muere       438        55
##   Sobrevive   128       113
```

Otra manera de calcular el porcentaje de registros correctamente clasificados usando la matriz de confusión:

```
porcentaje_correct<-100 * sum(diag(mat_conf)) / sum(mat_conf)
print(sprintf("El %% de registros correctamente clasificados es: %.4f %%",porcentaje_correct))
```

```
## [1] "El % de registros correctamente clasificados es: 75.0681 %"
```

El algoritmo C5.0 incorpora algunas opciones para ver la importancia de las variables (ver documentación para los detalles entre los dos métodos):

```
importancia_usage <- C50::C5imp(modelo2, metric = "usage")
importancia_splits <- C50::C5imp(modelo2, metric = "splits")
importancia_usage
```

```
##       Overall
## CLASS  100.00
## SEX    100.00
## AGE     93.73
```

```
importancia_splits
```

```
##       Overall
## CLASS      40
## SEX        40
## AGE        20
```

Curiosamente y aunque el conjunto de datos es muy sencillo, se aprecian diferencias en los métodos de importancia de las variables. Se recomienda en vuestro ejercicio mejorar la visualización de los resultados con la función ggplo2 o similar.

Interpretación de las variables en las predicciones.

Nos interesa saber para las predicciones que variable son las que tienen más influencia. Así, probaremos con un enfoque algorítmico de Random Forest y obtendremos métricas de interpretabilidad con la librería IML (https://cran.r-project.org/web/packages/iml/iml.pdf). As:

```
if(!require(randomForest)){
  install.packages('randomForest',repos='http://cran.us.r-project.org')
  library(randomForest)
}
```

```
## Loading required package: randomForest
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:gridExtra':
##
```

```
##     combine
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
if(!require(iml)){
  install.packages('iml', repos='http://cran.us.r-project.org')
  library(iml)
}
```

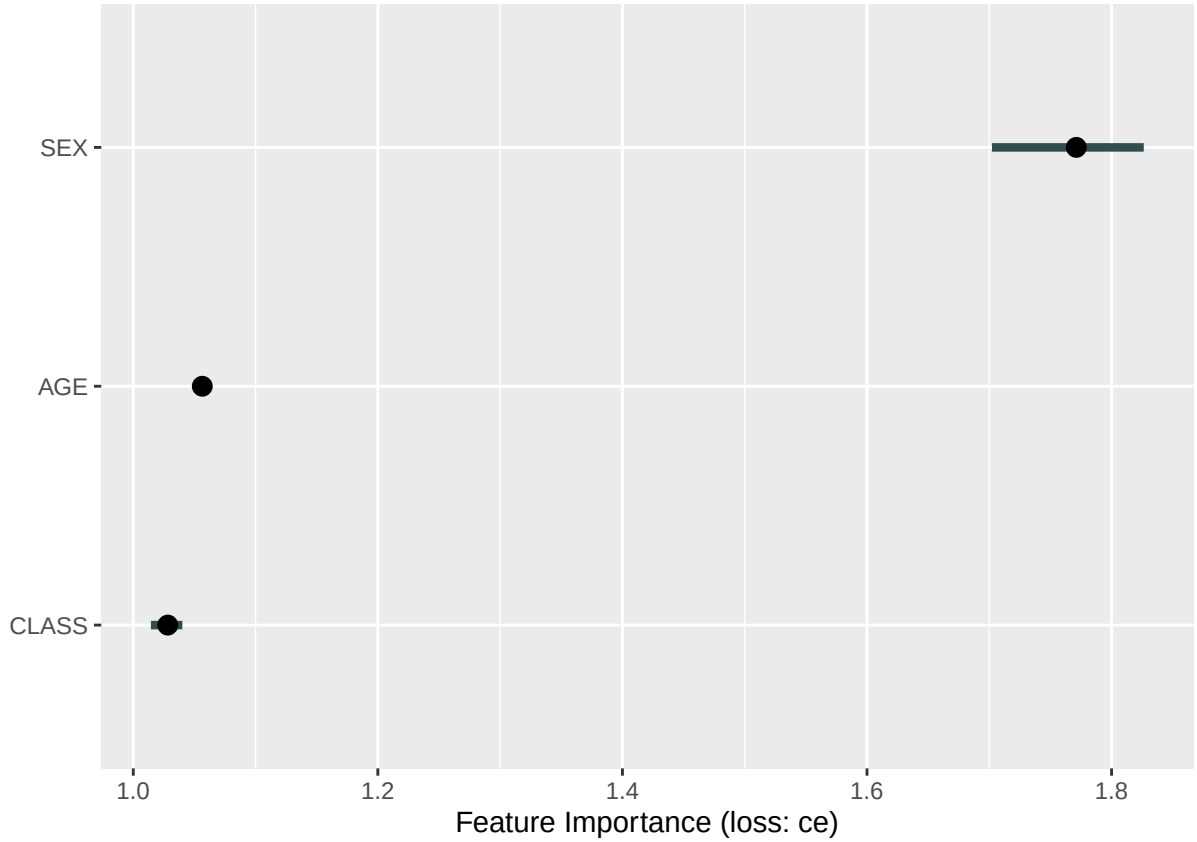
```
## Loading required package: iml
```

Empezamos ejecutado un Random Forest:

```
train.data <- as.data.frame(cbind(trainX,trainy))
colnames(train.data)[4] <- "SURVIVED"
rf <-  randomForest(SURVIVED ~ ., data = train.data, ntree = 50)
```

Podemos medir y graficar la importancia de cada variable para las predicciones del random forest con *FeatureImp*. La medida se basa funciones de pérdida de rendimiento que en nuestro caso será con el objetivo de clasificación ("ce").

```
X <- train.data[which(names(train.data) != "SURVIVED")]
predictor <- Predictor$new(rf, data = X, y = train.data$SURVIVED)
imp <- FeatureImp$new(predictor, loss = "ce")
plot(imp)
```




```
imp$results
```

```
##   feature importance.05 importance importance.95 permutation.error
```

```
## 1     SEX      1.702194   1.771160      1.826332         0.3851397
## 2     AGE      1.053292   1.056426      1.062069         0.2297205
## 3   CLASS      1.014420   1.028213      1.040125         0.2235855
```

Adicionalmente, podemos también dibujar los efectos locales acumulados (ALE) de la variable usando la libreria *patchwork*:

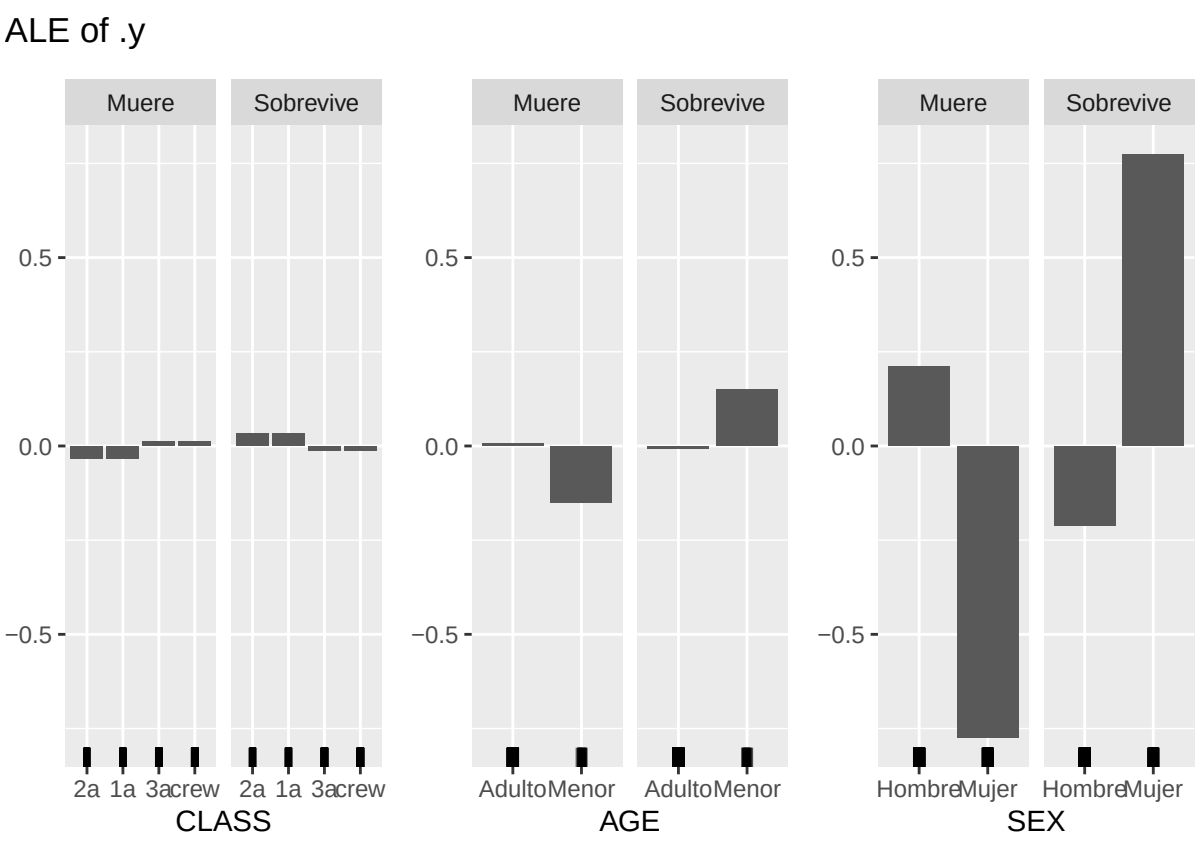
```
if(!require(patchwork)){
    install.packages('patchwork',repos='http://cran.us.r-project.org')
    library(patchwork)
}
```

```
## Loading required package: patchwork
```

```
##
## Attaching package: 'patchwork'
```

```
## The following object is masked from 'package:MASS':
##
##     area
```

```
effs <- FeatureEffects$new(predictor)
plot(effs)
```



Como podemos ver, el género es la variable con más importancia para las predicciones, siendo las mujeres mucho más propensas a sobrevivir. Nota: Se espera que los alumnos profundicen en la función de cara a la resolución de los ejercicios.

Enunciado del ejercicio

Para el conjunto de datos German Credit, los alumnos deben completar aquí la solución a la PEC3 que consiste de los siguientes apartados. Notad que se detalla el contenido necesario para cada apartado en la Sección 4 (Rúbrica).

El formato de entrega es como en las anteriores PECs: **usernameestudiant-PECn.html** (o PDF/Word) y el código **Rmd**.

Se debe entregar la PEC en el buzón de entregas del aula, como en las anteriores PECs.

Realizar un primer análisis descriptivo y de correlaciones. Es importante en este apartado entender bien los datos antes de seguir con los análisis posteriores. Lista todo lo que te haya sorprendido de los datos

Realizar un primer árbol de decisión. Puedes decidir utilizar todas las variables o, de forma justificada, quitar alguna para el ajuste del modelo

Con el árbol obtenido, realiza una breve explicación de las reglas obtenidas así como de todos los puntos que te parezcan interesantes. Un elemento a considerar es, por ejemplo, cuantas observaciones caen dentro de cada regla

Una vez tengas un modelo válido, procede a realizar un análisis de la bondad de ajuste sobre el conjunto de test y matriz de confusión. ¿Te parece un modelo suficientemente bueno como para utilizarlo? Justifica tu respuesta considerando todos los posibles tipos de error

Con un enfoque parecido a los puntos anteriores y considerando las mismas variables, enriquece el ejercicio mediante el ajuste de modelos de árbol de decisión complementarios. ¿Es el nuevo enfoque mejor que el original? Justifica la respuesta

Haz un resumen de las principales conclusiones de todos los análisis y modelos realizados

---

Rúbrica

- (Obligatorio) Se debe realizar un breve informe (Html) donde se respondan a las preguntas concretas, mostrando en primer lugar el código utilizado, luego los resultados y posteriormente los comentarios que se consideren pertinentes para cada apartado.

- 10% Hay un estudio sobre los datos de los que se parte, las variables que componen los datos. Los datos son preparados correctamente.
- 10% Se realiza un análisis descriptivo univariante (o análisis de relevancia) de algunas variables una vez se han tratado vs el target a nivel gráfico, comentando las que aparentemente son más interesantes. Análogamente se realiza un análisis de correlaciones.
- 20% Se aplica un árbol de decisión de forma correcta y se obtiene una estimación del error, mostrando gráficamente el árbol obtenido. La visualización debe ser comprensible y adecuada al problema a resolver.
- 15% Se explican las reglas que se obtienen en términos concretos del problema a resolver.

- 10% Se usa el modelo para predecir con muestras no usadas en el entrenamiento (holdout) y se obtiene una estimación del error. En base a la matriz de confusión, se comentan los tipos de errores y se valora de forma adecuada la capacidad predictiva del algoritmo.
- 10% Se prueba otro modelo de árbol o variantes diferentes del C50 y se comparan los resultados obtenidos, valorando si son mejores. Se recomienda experimentar usando un Random Forest (librería de R *randomForest* (https://cran.r-project.org/web/packages/randomForest/randomForest.pdf)).
- 10% Con los resultados obtenidos anteriormente, se presentan unas conclusiones contextualizadas donde se expone un resumen de los diferentes modelos utilizados (al menos 3) así como el conocimiento adquirido tras el trabajo realizado y los descubrimientos más importantes realizados en el conjunto de datos.
- 10% Utiliza métricas de explicabilidad como las comentadas en el ejemplo para obtener conclusiones de los datos.
- 5% Se presenta el código y es fácilmente reproducible.