Aprendizaje Supervisado en R

Estudio de los algoritmos de clasificación:

- **Árboles de decisión con rpart**
- **Evaluación mediante curvas ROC**
- **Knn**

Ejercicio 1: datos del titanic y rpart

Se le proporciona un fichero de nombre "train.csv" que contiene 4 columnas con valores numéricos y categóricos. Las variables son:

- Sobrevivió, 0 ó 1, donde 0 es no sobrevive.
- Clase: los pasajeros tienen billete de 1º, 2º o 3º.
- Sexo: variable categórica con el género.
- Y finalmente edad.

Estos datos representan las variables de 500 pasajeros del crucero Titanic. El objetivo es crear un modelo de predicción de la supervivencia de los pasajeros. Para ello realice los siguientes pasos:

1. Cargue los datos en una variable de nombre train que será un dataframe con 4 variables o atributos y 500 observaciones (pasajeros) que vienen codificados en el fichero. Para ello:

```
train<-read.delim("train.csv", sep = "\t", head = TRUE)
rownames(train) <- train$id
train$id <- NULL
```

   Inspeccione el conjunto de entrenamiento (funciones head, str, dim).

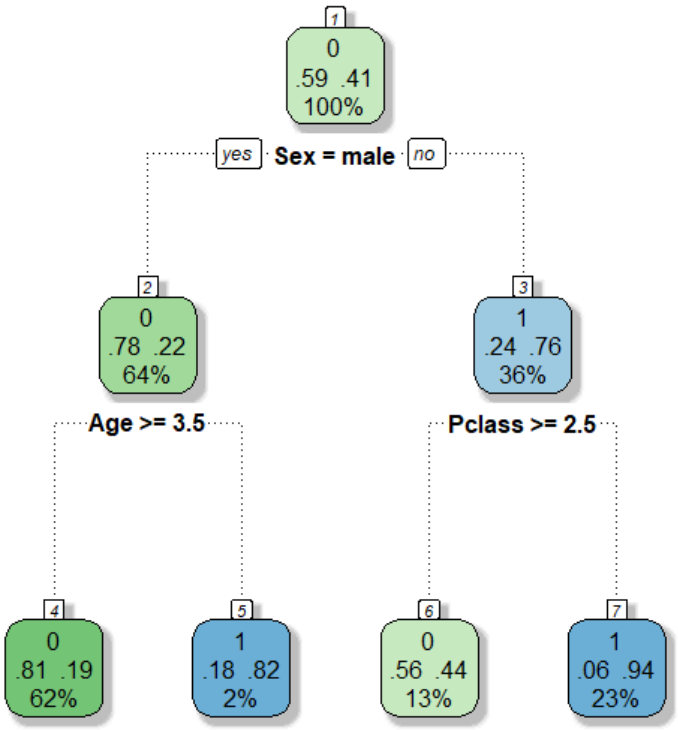
2. Cargue los siguientes paquetes para crear árboles de decisión.

```
install.packages("rattle")

library(rpart)
library(rattle)
library(RColorBrewer)
```

3. La función para crear árboles de decisión que vamos a utilizar es rpart. Teclee en la consola help("rpart") y estudie como aplicarla. Utilice los parámetros:
   - Formula, indicando que el objetivo a predecir es la supervivencia en función del resto de atributros.
   - Indique el conjunto de entrenamiento.
   - El parámetro method indique que va a realizar clasificación sobre variables categóricas o factor en R.
   - Visualice el resultado usando la función plot.

```
fancyRpartPlot(tree)
```

- Analizamos el resultado. Indique cuál de las siguientes afirmaciones es correcta, (recuerda, 0 no sobrevive 1 vive) :
  - El árbol predecirá que las pasajeras de la clase 3 no sobrevivirán, aunque está cerca.
  - La clase mayoritaria del nodo raíz es positiva, lo que denota supervivencia.
  - El atributo que sigue al género es hombre es una variable categórica.



4. Cargue el conjunto de datos de test del fichero con nombre “test.csv” de igual modo en el que se realizó la carga de los datos de entrenamiento. Realice la predicción con el conjunto de test para construir la matriz de confusión y calcular la accuracy del modelo, utilizando las funciones predict y table respectivamente.

5. Copie el siguiente código

```
set.seed(1)
tree <- rpart(Survived ~., train, method="class",
control=rpart.control(cp=0.00001))

fancyRpartPlot(tree)
```

¿Qué observa? El modelo resultante funciona bien si calculamos su accuracy pero es difícil de interpretar. Para arreglar esta situación pode el árbol con la función prune y vuelva a dibujarlo.

6. ¿Los árboles construidos qué heurística aplican? Para modificar la heurística utilice en la

función rpart el parámetro de entrada Split y genere el árbol con nombre tree_i.

Ejercicio 2: datos del titanic y evaluación mediante curva ROC

Utilizando el ejercicio anterior, su árbol de decisión creado, en lugar de obtener la clase como resultado de predicción vamos a obtener la probabilidad de que se obtenga una clase con el siguiente código:

```
all_probs <- predict(tree, test, type="prob")

summary(all_probs)

all_probs[,2]
probs <- all_probs[,2]
```

Vamos a realizar una análisis ROC y para ello siga los siguientes pasos:

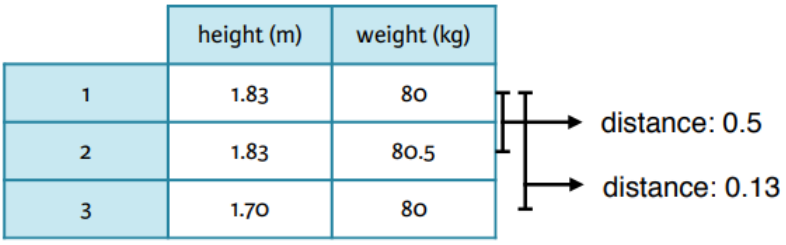
1. Cargue la librería ROCR. Llame a la función prediction pasando como argumentos las probabilidades y la columna del atributo clase. El resultado de esta función, juntos con "tpr" y "fpr" utilícelo como argumentos de entradas de la función performance. Finalmente llame a la función plot dando como entrada el resultado de la función anterior.

2. Para obtener el valor de AUC o área bajo la curva debemos escribir el siguiente código:

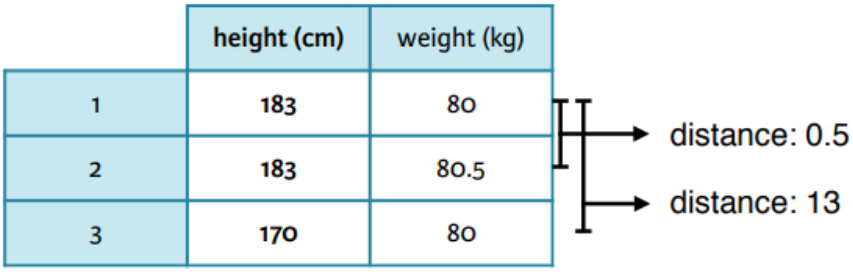
```
perf_auc <- performance(pred, "auc")

print(perf_auc@y.values[[1]])
```

Ejercicio 3: datos del titanic y knn

La técnica de clasificación knn se basa en la distancia de la observación a clasificar en función de los vecinos más cercanos. Observe el siguiente ejemplo en donde tenemos 3 observaciones con dos atributos, la altura y peso de las personas que constituyen la población de estudio. Si calculamos la distancia entre las observaciones 1-2 y 1-3 vemos que son cercanas.



Sin embargo, al cambiar la escala del atributo altura de metro a centímetros se observa que la distancia entre 1-3 es mucho mayor que entre 1-2. Luego la escala de los atributos influye en el cálculo de distancias.



Para evitar este problema se normaliza el valor de todos los atributos entre 0 y 1. De manera que dado el vector X se aplica la siguiente ecuación para cada uno de sus valores x:

$$x = [x - \min(X)] / [\max(X) - \min(X)]$$

El objetivo es crear un modelo de predicción de la supervivencia de los pasajeros del Titanic utilizando una técnica de aprendizaje basado en dinstancia, el knn. Para ello sigua los siguientes pasos:

1. La función knn tiene como parámetro el conjunto de entrenamiento, el de test y el vector de clases, para ello realice lo siguiente:

```
train_labels <- train$Survived
test_labels <- test$Survived

knn_train <- train
knn_test <- test

knn_train$Survived <- NULL
knn_test$Survived <- NULL
```

2. Normalice los atributos numéricos, es decir, la clase y edad del pasajero.

3. Llame a la función knn y obtenga la matriz de confusión. A partir de ella calcule el accuracy del modelo.

4. Vamos a realizar un estudio para averiguar cuál es el mejor valor de K para este conjunto de datos. Con este objetivo vamos a generar un vector de nombre range con todos los k a analizar y un vector de nombre acc donde almacenaremos los valores de accuracy de todos los modelos. Para ello copie el siguiente código:

```
set.seed(1)

library(class)
range <- 1:round(0.2 * nrow(knn_train))
accs <- rep(0, length(range))
```

Escriba un bucle en donde para cada valor de k llame a la función knn y calcule su valor de accuracy. Finalmente visualizamos los accuracy obtenidos y nos quedamos con el k cuyo accuracy es máximo:

```
plot(range, accs, xlab = "k")
which.max(accs)
```