R-PL3

Gabriel López, Sergio Sanz, Álvaro Zamorano

26 de octubre de 2019

1. Ejercicio realizado en clase.

Para obtener la **función de clasificación** mediante el algoritmo construcción de **árboles de decisión de Hunt** es necesario usar los paquetes `rpart` y `tree`. Estos paquetes hay que descargarlos desde la página de CRAN y para instalarlos hay que ejecutar el siguiente código:

```
> install.packages("./Paquetes/rpart_4.1-15.zip")

package 'rpart' successfully unpacked and MD5 sums checked

> install.packages("./Paquetes/tree_1.0-40.zip")

package 'tree' successfully unpacked and MD5 sums checked
```

De esta forma, los paquete únicamente estarán instalados. Para poder usarlos es necesario cargarlos:

```
> library(rpart)
> library(tree)
```

- Con rpart obtendremos las particiones recursivas para la clasificación y los árboles de decisión.
- Con tree, los árboles de clasificación y regresión.

1.1. Función de clasificación.

Los datos a usar en este primer ejercicio se componen de 9 calificaciones de estudiantes compuestas por Teoría, Laboratorio, Prácticas y Calificación Global.

Para introducir estos datos en el algoritmo a usar es necesario tener un fichero `.txt` con el siguiente aspecto.

Suceso	Teoría	Lab	Prac	Calif
s1	A	A	B	Ap
s2	A	B	D	Ss
s3	D	D	C	Ss
s4	D	D	A	Ss
s5	B	C	B	Ss
s6	C	B	B	Ap
s7	B	B	A	Ap
s8	C	D	C	Ss
s9	B	A	C	Ss

Procedemos a leer dicho fichero .txt mediante el uso de la función *read.table.*

```
> calificaciones<-read.table("./Datos/Calificaciones.txt")
```

Para asegurarnos de que todo irá bien a la hora de realizar la clasificación, convertimos los datos leídos en un **dataframe.**

```
> muestra<-data.frame(calificaciones)
```

Nuestros datos ya se encuentran preparados para aplicarles la función **rpart.** Es importante destacar el uso de minsplit ya que disponemos de una muestra con un número muy reducido de datos.

```
> clasificacion<-rpart(Calif~.,data=muestra,method="class",minsplit=1)
> clasificacion

n= 9

node), split, n, loss, yval, (yprob)
      * denotes terminal node

1) root 9 3 Ss (0.3333333 0.6666667)
  2) Lab=A,B 5 2 Ap (0.6000000 0.4000000)
    4) Prac=A,B 3 0 Ap (1.0000000 0.0000000) *
    5) Prac=C,D 2 0 Ss (0.0000000 1.0000000) *
  3) Lab=C,D 4 0 Ss (0.0000000 1.0000000) *
```

Para mostrar el árbol de clasificación hacemos uso de una función que hemos definido, pero para poder usarla en primer lugar es necesario instalar el paquete **rpart.plot.**

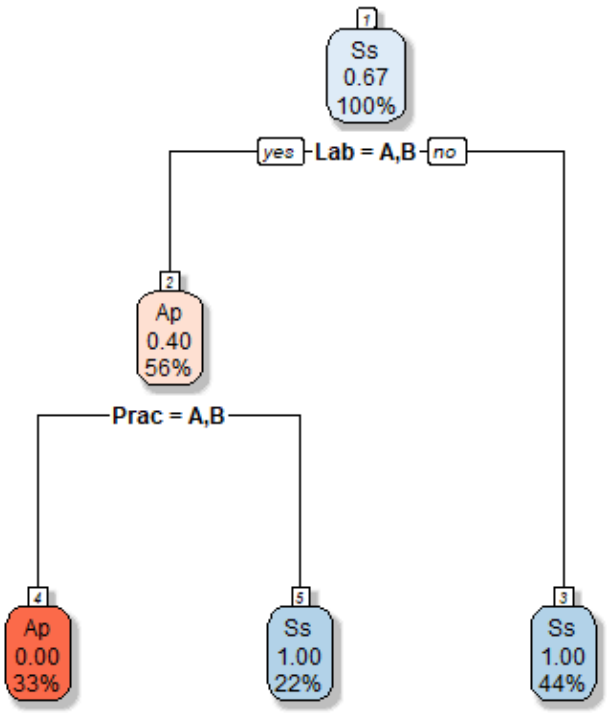
```
> install.packages("./Paquetes/rpart.plot_3.0.8.zip")

package 'rpart.plot' successfully unpacked and MD5 sums checked

> library(rpart.plot)
```

Dicha función es:

```
> source("Funciones/plotTree.R")
> plotTree

function (tree, ruta)
{
    png(paste("./tmp/", ruta, sep = ""))
    rpart.plot(tree, box.palette = "RdBu", shadow.col = "gray",
        nn = TRUE)
    dev.off()
}
```

Procedemos a su ejecución.

```
> plotTree(clasificacion, "classTree.png")
```



Por último, se aplica la función **tree** a nuestros datos.

```
> clasificaciontree<-tree(Calif~.,data=muestra,mincut=1,minsize=2)
> clasificaciontree
```

```
node), split, n, deviance, yval, (yprob)
      * denotes terminal node

1) root 9 11.46 Ss ( 0.3333 0.6667 )
  2) Lab: A,B 5  6.73 Ap ( 0.6000 0.4000 )
    4) Prac: A,B 3  0.00 Ap ( 1.0000 0.0000 ) *
    5) Prac: C,D 2  0.00 Ss ( 0.0000 1.0000 ) *
  3) Lab: C,D 4  0.00 Ss ( 0.0000 1.0000 ) *
```

1.2. Análisis de regresión lineal.

En este caso trabajaremos con datos de planetas, en concreto su Radio y su Diámetro. Los planetas de los que se tienen los datos son: Mercurio, Venus, Tierra y Marte, y el .txt del que se leen dichos datos tiene el aspecto que sigue.

Planeta	Radio	Diámetro
Mercurio	2.4	5.4
Venus	6.1	5.2
Tierra	6.4	5.5
Marte	3.4	3.9

Al igual que anteriormente, es necesario leer dicho fichero y pasarlo a data-frame.

```
> planetas<-read.table("./Datos/Planetas.txt")
> muestraP<-data.frame(planetas)
```

El análisis de regresión se hace mediante el uso de la función **lm** contenida en el paquete stats. Cabe destacar que el primero de sus argumentos es de tipo *fórmula* donde una expresión de la forma y ~ model se interpreta como una especificación de que la respuesta `y` está modelada por un predictor lineal especificado simbólicamente por model, es decir, en nuestro caso model=x por lo que su ejecución queda como:

```
> regresionP<-lm(D~R,data=muestraP)
> regresionP

Call:
lm(formula = D ~ R, data = muestraP)

Coefficients:
(Intercept)            R
     4.3624       0.1394
```

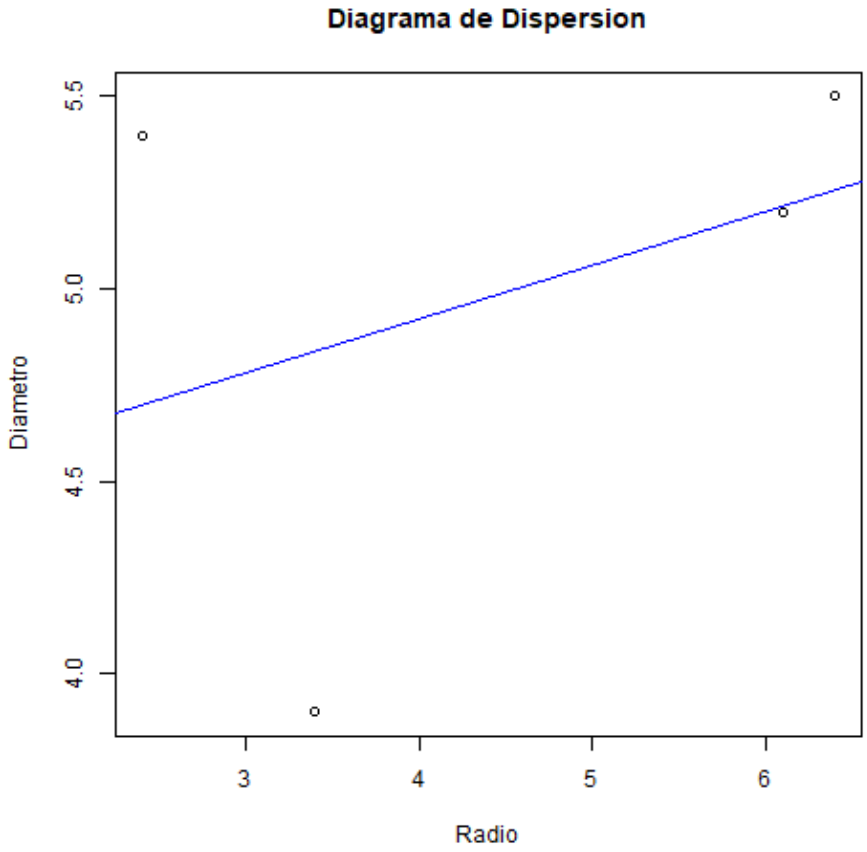
De acuerdo a la ecuación de una recta `y=a+b*x`, el primero de los coeficientes es el término independiente (a), y el segundo de ellos la b.

Para mostrar el gráfico de dispersión y la recta de ajuste es necesario hacer uso de varias librerías.

```
> library(foreign)
> library(ggplot2)
> library(psych)
```

Estas librerías se usan en funciones externas usadas para representar los gráficos requeridos.

```
> source("Funciones/plotDisp.R")
> plotDisp(planetas,regresionP,"Radio","Diametro","regPlanetas.png")
```



2. Segunda parte

2.1. Función de clasificación.

En este ejercicio usaremos datos correspondientes a ventas de coches, en concreto son 10 muestras compuestas por: TipoCarnet, NúmeroRuedas, NúmeroPasajeros y TipoVehículo.

Para introducir estos datos en el algoritmo a usar es necesario tener un fichero `.txt` con el siguiente aspecto.

Vehículo	TipoCarnet	NúmeroRuedas	NúmeroPasajeros	TipoVehículo
v1	B	4	5	Coche
v2	A	2	2	Moto
v3	N	2	1	Bicicleta
v4	B	6	4	Camión
v5	B	4	6	Coche
v6	B	4	4	Coche
v7	N	2	2	Bicicleta
v8	B	2	1	Moto
v9	B	6	2	Camión
v10	N	2	1	Bicicleta

Procedemos a leer dicho fichero como anteriormente y pasarlo a dataframe.

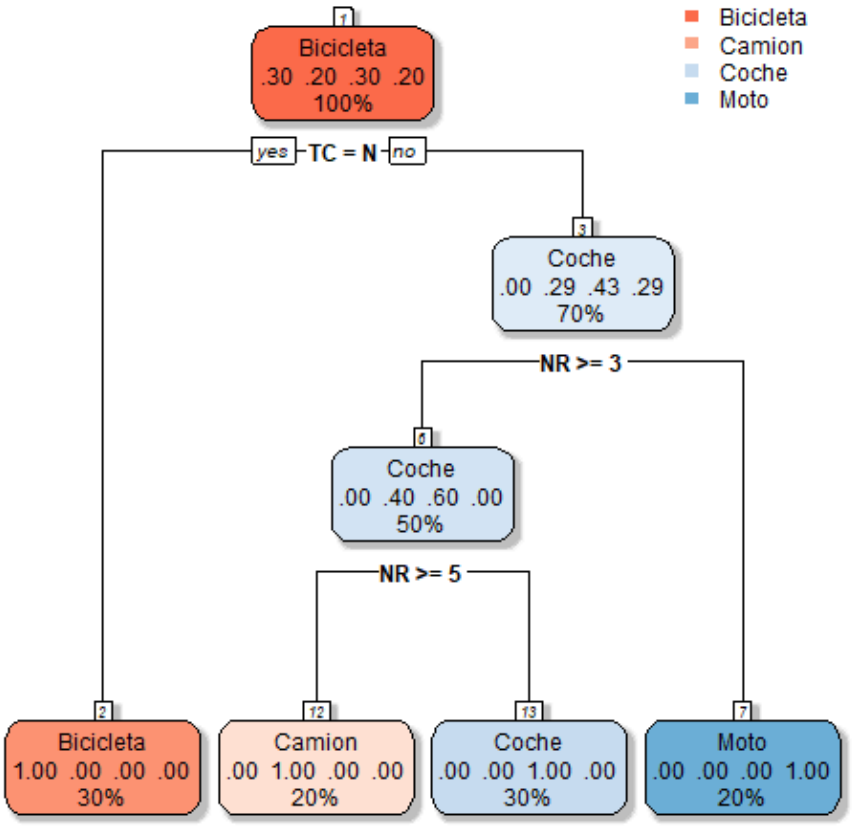
```
> vehiculos<-read.table("./Datos/Vehiculos.txt")
> muestraV<-data.frame(vehiculos)
```

Nuestros datos ya se encuentran preparados para aplicarles la función **rpart.** La clasificación a obtener será el tipo de vehículo al que pertenece cada uno de ellos.

```
> clasV<-rpart(TV~.,data=muestraV,method="class",minsplit=1)
> clasV

n= 10

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 10 7 Bicicleta (0.3000000 0.2000000 0.3000000 0.2000000)
   2) TC=N 3 0 Bicicleta (1.0000000 0.0000000 0.0000000 0.0000000) *
   3) TC=A,B 7 4 Coche (0.0000000 0.2857143 0.4285714 0.2857143)
     6) NR>=3 5 2 Coche (0.0000000 0.4000000 0.6000000 0.0000000)
      12) NR>=5 2 0 Camion (0.0000000 1.0000000 0.0000000 0.0000000) *
      13) NR< 5 3 0 Coche (0.0000000 0.0000000 1.0000000 0.0000000) *
     7) NR< 3 2 0 Moto (0.0000000 0.0000000 0.0000000 1.0000000) *
```

Procedemos a mostrar el árbol de clasificación.

```
> plotTree(clasV, "classTreeV.png")
```



Por último, se aplica la función **tree** a nuestros datos.

```
> classTV<-tree(TV~.,data=muestraV,mincut=1,minsize=2)
> classTV

node), split, n, deviance, yval, (yprob)
      * denotes terminal node

1) root 10 27.32 Bicicleta ( 0.3 0.2 0.3 0.2 )
  2) NR < 3 5  6.73 Bicicleta ( 0.6 0.0 0.0 0.4 )
    4) TC: A,B 2  0.00 Moto ( 0.0 0.0 0.0 1.0 ) *
    5) TC: N 3  0.00 Bicicleta ( 1.0 0.0 0.0 0.0 ) *
  3) NR > 3 5  6.73 Coche ( 0.0 0.4 0.6 0.0 )
    6) NR < 5 3  0.00 Coche ( 0.0 0.0 1.0 0.0 ) *
    7) NR > 5 2  0.00 Camion ( 0.0 1.0 0.0 0.0 ) *
```

2.2. Análisis de regresión lineal.

En este caso tenemos que hacer un análisis de regresión lineal para 4 muestras distintas compuestas por pares de datos.

Como en ocasiones anteriores, procedemos a leer dichas muestras y pasarlas a dataframe.

```
> pares<-read.table("./Datos/Pares.txt")
> muestraPS<-data.frame(pares)
```

El análisis de regresión se hace mediante el uso de la función **lm.** En este caso será neceario hacer cuatro análisis diferentes.

```
> (r1<-lm(V2~V1,data=muestraPS))

Call:
lm(formula = V2 ~ V1, data = muestraPS)

Coefficients:
(Intercept)           V1
     3.0001       0.5001

> (r2<-lm(V4~V3,data=muestraPS))

Call:
lm(formula = V4 ~ V3, data = muestraPS)

Coefficients:
(Intercept)           V3
      3.001        0.500

> (r3<-lm(V6~V5,data=muestraPS))

Call:
lm(formula = V6 ~ V5, data = muestraPS)

Coefficients:
(Intercept)           V5
     3.0025       0.4997

> (r4<-lm(V8~V7,data=muestraPS))

Call:
lm(formula = V8 ~ V7, data = muestraPS)

Coefficients:
(Intercept)           V7
     3.0017       0.4999
```

De acuerdo a la ecuación de una recta `y=a+b*x`, el primero de los coeficientes es el término independiente (a), y el segundo de ellos la b.

Mostraremos estos análisis en una misma figura mediante el uso de:

```
> source("Funciones/plotDisp2.R")
> plotDisp2
```

```
function (data, r1, r2, r3, r4, ruta)
{
    png(paste("./tmp/", ruta, sep = ""))
    par(mfrow = c(2, 2))
    plot(data[, 1], data[, 2], main = "Muestra 1")
    abline(r1, col = "red")
    plot(data[, 3], data[, 4], main = "Muestra 2")
    abline(r2, col = "blue")
    plot(data[, 5], data[, 6], main = "Muestra 3")
    abline(r3, col = "green")
    plot(data[, 7], data[, 8], main = "Muestra 4")
    abline(r4, col = "yellow")
    dev.off()
}
```

Procedemos a su ejecución.

```
> plotDisp2(pares, r1, r2, r3, r4, "rPares.png")
```