Anexo VI. Análisis univariante y multivariante: librería handy_functions

A continuación se presenta la biblioteca creada por nosotros para realizar el análisis univariante y multivariante de nuestro conjunto de datos, que hemos llamado *handy_functions.*

Hemos creído conveniente diseñar funciones en vez de un programa principal por dos principales motivos:

- Dada la naturaleza de las funciones (orientadas a variables de nuestro conjunto de datos más que al conjunto de datos en global), podemos aplicarlas sobre variables específicas para concretar nuestro análisis.

- Al ser funciones por lo general de un ámbito muy amplio (es decir, poco concretas), pueden ser utilizadas para otros proyectos en R.

La librería cuenta con cuatro funciones, detalladas dentro del código.

```
library(dplyr)    #Librería usada para manipulación de datos
library(e1071)    #Librería para obtención de parámetros estadísticos
library(dbplyr)   #Librería usada para manipulación de datos

show_description = function(aux_df, name){

  #Descripción: esta función recibe un dataframe auxiliar de información
  #             de las variables del dataframe principal y un nombre de
  #             de variable y devuelve la descripción de esta variable
  #
  #Inputs:
  #    - aux_df: dataframe auxiliar que contiene los siguientes campos:
  #             * Nombre de la variable
  #             * Tipo de variable
  #             * Descripción de la variable. Esta columna debe llamarse
  #               "description"
  #    - name: columna cuya descripción se desea conocer
  #
  #Outputs:
  #    - Descripción de la variable deseada
  #
  #Notas:
  #
  #    -  Esta función se comporta realmente como un atajo para no tener que
  #       escribir toda la instrucción, ya que realmente esta función solo
  #       realiza un acceso a una posición de la columna "description"

  return (aux_df[aux_df["variable"] == name,]$description)
}
```

```r
univ_cat = function(df, name, complete = FALSE, n = 10, labels = TRUE,
                    save = FALSE, file = ""){

 #Descripción: esta función recibe un dataframe, y el nombre de una variable
 #             categórica y grafica un diagrama de barras de la frecuencia
 #             absoluta de las categorías más frecuentes en dicha variable.
 #Inputs:
 #     - df: dataframe con un conjunto de datos
 #     - name: nombre de la variable categórica
 #     - complete: valor lógico que indica si se grafican todas las categorías
 #                 de la variable. El valor por defecto es FALSE
 #     - n: entero que indica el nº de categorías de la variable a graficar.
 #          El valor por defecto es 10
 #     - labels: valor lógico que indica si los ejes x e y de la gráfica se
 #               muestran o no. El valor por defecto es TRUE
 #     - save: valor lógico que indica si el diagrama obtenido se guarda como
 #             imagen .jpeg. El valor por defecto es TRUE
 #     - file: cadena de caracteres que sirve de ruta en el ordenador para
 #             guardar la imagen. El valor por defecto es una cadena vacía,
 #             que sirve a la función para crear una ruta específica en el
 #             caso de que no se proporcione un valor
 #
 #Outputs:
 #     - t.freq: tabla de frecuencias con las n categorías más frecuentes
 #
 #Notas:
 #     - la ruta creada por la función si no se le pasa un valor concreto es
 #       muy específica, y se requiere la creación de una carpeta llamada
 #       "Plots" en el directorio de trabajo, por lo que se recomienda indicar
 #       una ruta siempre que se quiera guardar la gráfica como imagen

 if (file == ""){
   file = paste(paste("Plots/Hist_Freq",name, sep = "_"), ".jpeg", sep = "")

#Se genera una ruta específica si no hay valor en file
 }
 t.freq = as.data.frame(table(df[,name]))
 colnames(t.freq) = c(name,"Frequency")

 if (!complete & nrow(t.freq) >= n){
   t.freq = top_n(t.freq, n, Frequency)    #Se escogen las n categorías más
                                           #frecuentes
 }

 if (!save){
   par(mar = c(10,4,4,2))
   if (labels){
     barplot(t.freq$Frequency, names.arg = t.freq[[name]], las = 2, col = "blue",
             main = paste("Histograma del top",min(n,nrow(t.freq)), "de", name),
             xlab = name, ylab = "Frecuencia absoluta")
   } else {
```

```
      barplot(t.freq$Frequency, names.arg = t.freq[[name]], las = 2, col = "blue",
              main = paste("Histograma del top",min(n,nrow(t.freq)), "de", name))
    }
  }

  else {
    jpeg(file)
    par(mar = c(10,4,4,2))
    if (labels){
      barplot(t.freq$Frequency, names.arg = t.freq[[name]], las = 2, col = "blue",
              main = paste("Histograma del top",min(n,nrow(t.freq)), "de", name),
              xlab = name, ylab = "Frecuencia absoluta")
    } else {
      barplot(t.freq$Frequency, names.arg = t.freq[[name]], las = 2, col = "blue",
              main = paste("Histograma del top",min(n,nrow(t.freq)), "de", name))
    }
    dev.off()
  }
  return (t.freq)
}

univ_num = function(df, name, labels = TRUE, save = FALSE, file = ""){

  #Descripción: esta función recibe un dataframe, y el nombre de una variable
  #             numérica y grafica un histograma de frecuencias absolutas
  #             y un diagrama de caja y bigote, así como muestra el coef. De
  #             kurtosis y el de skewness.
  #Inputs:
  #     - df: dataframe con un conjunto de datos
  #     - name: nombre de la variable numérica
  #     - labels: valor lógico que indica si los ejes x e y de las gráficas se
  #               muestran o no. El valor por defecto es TRUE
  #     - save: valor lógico que indica si las figuras obtenidas se guardan como
  #             imagen .jpeg. El valor por defecto es TRUE
  #     - file: cadena de caracteres que sirve de ruta en el ordenador para
  #             guardar la imagen. El valor por defecto es una cadena vacía,
  #             que sirve a la función para crear una ruta específica en el
  #             caso de que no se proporcione un valor
  #
  #Outputs:
  #     - plot_info: pareja con información de los diagramas graficados
  #
  #Notas:
  #     - la ruta creada por la función si no se le pasa un valor concreto es
  #       muy específica, y se requiere la creación de una carpeta llamada
  #       "Plots" en el directorio de trabajo, por lo que se recomienda indicar
  #       una ruta siempre que se quiera guardar la gráfica como imagen
```

```
  if (file == ""){
    file = paste(paste("Plots/General_info", name, sep = "_"), ".jpeg", sep = "")
    print(file)
  }
  data = df[[name]]
  coef_k = kurtosis(data, na.rm = T)
  coef_skew = skewness(data, na.rm = T)
  if (!save){
    par(mfrow = c(1,2), mar = c(5,5,6,5))
    if (labels){
      p1 = hist(data, col = "yellow", xlab = name, ylab = "Frecuencia absoluta",
           main = paste("Histograma de\n", name),
           sub = paste("Coef. kurtosis: ", round(coef_k,3)))
      p2 = boxplot(data, col = "green", ylab = name,
              main = paste("Diagrama de caja y\nbigotes de", name),
              sub = paste("Coef. simetria: ", round(coef_skew,3)))
    }
    else{
      hist(data, col = "yellow")
      boxplot(data, col = "green")
    }
  }

  else {
    jpeg(file)
    par(mfrow = c(1,2), mar = c(5,5,6,5))
    if (labels){
      p1 = hist(data, col = "yellow", xlab = name, ylab = "Frecuencia absoluta",
           main = paste("Histograma de\n", name),
           sub = paste("Coef. kurtosis: ", round(coef_k,3)))
      p2 = boxplot(data, col = "green", ylab = name,
              main = paste("Diagrama de caja y\nbigotes de", name),
              sub = paste("Coef. simetria: ", round(coef_skew,3)))
    }
    else{
      hist(data, col = "yellow")
      boxplot(data, col = "green")
    }
    dev.off()
  }

  return (c(p1,p2))
}
```

```r
corplots = function(df, cor.matrix, n = 6, threshold = 0.6, save = F){

  #Descripción: esta función recibe un dataframe, una matriz de correlaciones
  #             y gráfica un diagrama de dispersión para cada par de variables
  #             que estén altamente correlacionadas
  #
  #Inputs:
  #      - df: dataframe con el conjunto de datos
  #      - cor.matrix: matriz de correlaciones de las variables del dataframe
  #      - n: nº máximo de par de variables a graficar
  #      - threshold: entero entre 0 y 1 que indica el umbral a partir del cual
  #                   dos variables se consideran correlacionadas (la comparación
  #                   se hace en valor absoluto, para tener en cuenta las que
  #                   se encuentren correlacionadas negativamente)
  #      - save: valor lógico que indica si las figuras obtenidas se guardan como
  #              imagen .jpeg. El valor por defecto es TRUE


  count = 0
  indexes = which(abs(cor.matrix) > threshold)
  for (i in indexes){
    k = arrayInd(i, dim(cor.matrix))
    row = rownames(cor.matrix)[k[,1]]
    col = colnames(cor.matrix)[k[,2]]
    if (row != col){
      if (count > n){
        break
      } else {
        print(paste(row, "<-->",col))
        count = count + 1
        if (save){
          jpeg(paste(paste("Plots/Corr", row, col, sep = "_"), ".jpeg", sep = ""))
          plot(as.data.frame(df[,row])[,1], as.data.frame(df[,col])[,1],
               xlab = row, ylab = col, pch = ".",
               main = paste("Gráfico de dispersión de",row,"frente
                            a",col,"\n","Umbral",threshold),
               sub = paste("Coeficiente de correlación",cor.matrix[i]))
          dev.off()
        } else {
          plot(as.data.frame(df[,row])[,1], as.data.frame(df[,col])[,1],
               xlab = row, ylab = col, pch = ".",
               main = paste("Gráfico de dispersión de",row,"frente
                            a",col,"\n","Umbral",threshold),
               sub = paste("Coeficiente de correlación",cor.matrix[i]))
        }
      }
    }
  }
}
```