Índice general

Parte 1

Introducción a R

Los siguientes apuntes consiste en una breve introducción a R para poder seguir de forma adecuada el curso de *Técnicas de Análisis Multivariante y Aplicaciones* del *Máster Universitario en Estadística Aplicada* de la Universidad de Granada. Estos apuntes no pretenden ser un curso de R sino una breve introducción a utilidades que pueden ser necesitadas a la hora de establecer diferentes técnicas de análisis multivariante.

Antes de entrar, describiremos R como:

- R es un lenguaje de programación para Estadística.
- Es un lenguaje que permite el almacenamiento, manejo y tratamiento estadístico de los datos.
- Es un lenguaje interpretado, es decir, se ejecuta cada orden conforme la escribimos.
- Es un lenguaje vectorial.

Las principales ventajas de R son:

- Es gratuito.
- Totalmente programable pudiendo añadir nuevos paquetes por parte de los usuarios.
- Es multiplataforma.
- Esta destinada a objeto (manipulan los datos de entrada).

Dentro de estas funciones debemos destacar que existen dos tipos de perfiles:

1. Perfil usuario: utiliza las funciones ya creadas por otros usuarios. Estas funciones o paquetes se denomina *library* y se deben descargar y cargar cada vez que se ejecute el programa.
2. Perfil programador: crea nuevas funciones y las pone a disposición de la comunidad R

Para descargar e instalar R solo tenemos que acceder a la sección de descargar de la página oficial https://www.r-project.org/, en la siguiente ventana hay que elegir un *mirror* desde donde descargarlo, elegiremos el de España https://cran.rediris.es y finalmente la plataforma que tengamos.

Una vez instalada la interface de R, esta es muy sencilla y sin apenas opciones, esta destinada a programar, ya sea a nivel usuario o programador. En esta interfaz debemos tener en cuenta lo siguiente:

- R evalúa expresiones enteras.
- R distingue entre mayúsculas y minúsculas.
- Todas las órdenes que empiecen por # se consideran comentarios y no se ejecutan.

Se pueden usar editores de R que pueden facilitar el trabajo como puede ser Rstudio, el cuál incluye una consola; editor de sintaxis que apoya la ejecución de código; así como herramientas para el trazado, la depuración y la gestión del espacio de trabajo (https://rstudio.com/).

1.1. Primeras órdenes

- Asignación: $<-$ es equivalente a $=$; y es la forma de asignar valores. Por ejemplo con $x <- 3$ asignaremos el valor de 3 a la variable x.
- R distingue entre mayúsculas y minúsculas $x <- 3$ no es igual a $X <- 3$.
- Se pueden realizar las operaciones aritméticas comunes:
  - Potencia.
  - Producto $*$ y división /.
  - Suma + y resta $-$.
- Operadores de la lógica:
  - ! no.
  - ==, != igual, distinto.
  - >, >= mayor, mayor o igual.
  - <, <= menor, menor o igual.
  - | o.
  - & y.

Algunos ejemplos de estas operaciones son:

```r
3+2 #suma
```

```
## [1] 5
```

```r
3*2 #multiplicación
```

```
## [1] 6
```

```r
3/2 #división
```

```
## [1] 1.5
```

```r
3 ^ 2 #potencia
```

```
## [1] 9
```

```r
2>3 #lógica mayor que
```

```
## [1] FALSE
```

```r
# las ordenes lógica devuelven un operador  TRUE o FALSE según se verifique o no
```

1.2. Objetos

Los tipos de objetos con los que se trabaja en R se pueden clasificar en:

- funciones
- vectores
- listas.

Los datos son fundamentalmente de tipo:

- matrix.
- data.frame.
- factor.

Existen una serie de valores especiales que son:

- Na (not avaible): valor no disponible (perdido).
- NaN (not a number): valor numérico, imposible o sin sentido.
- Inf: valor infinito.

1.2.1. Funciones

Una función es un orden para realizar un procedimiento determinado. Esta formada por una serie de argumentos con los que se pueden personalizar las salidas. Tiene la forma:

nombre de la función(argumento1, argumento2, argumento3.....)

por ejemplo unas orden básica es la operación logaritmo:

```
log(2)

## [1] 0.6931472

log(2,10)        #logaritmo de 2 con base 10

## [1] 0.30103

log(x=2,base=10)

## [1] 0.30103
```

Las funciones y bibliotecas pueden estar descargadas o tener que descargarlas. Si la biblioteca no la hemos usado nunca hay que descargarla con la opción INSTALAR PAQUETES y luego cada vez que se use hay que activarla con la orden *library*(*paquete*).

1.2.2. Vectores

Los vectores son:

- Conjuntos ordenados de valores.
- Se referencia sobre el nombre del vector.
- Se puede seleccionar un individuo o un conjunto de ellos referenciando su posición.

Para signar valores a un vector podemos utilizar las siguientes formas:

- Valor concreto $x < -3$.
- Conjunto de valores con la orden $c()$.
- Secuencia con las órdenes: : o rep().

```
a<-1.5    # asigna 1.5 al la variable a
a
```

```
## [1] 1.5

b<-c(3,4,5,7)    # crea un vector con los valores 3,4,5 y 7
b

## [1] 3 4 5 7

c<-2:8   # serie del 2 al 8
c

## [1] 2 3 4 5 6 7 8

d<-5:2
d

## [1] 5 4 3 2

e<-seq(from=1,to=10,by=3)       # serie del 1 al 10 saltando de 3 en 3
e

## [1]  1  4  7 10

f<-seq(2,10,by=2)
f

## [1]  2  4  6  8 10

g<-rep(5,4)      # repite el 5, 4 veces
g

## [1] 5 5 5 5

h<-rep(b,4)      # repite el vector b 4 veces
h

##  [1] 3 4 5 7 3 4 5 7 3 4 5 7 3 4 5 7

c(a,b)

## [1] 1.5 3.0 4.0 5.0 7.0
```

Para referenciar a un/os individuo/s de un vector se hace por el subindice de la forma:

- Para indicar el elemento n del vector x: x[n].

- Para indicar los elementos n1 y el n2 del vector x: x[c(n1,n2)].
- Para indicar los elementos entre el n1 y el n2 del vector x: x[n1,n2].
- La función *length(x)* nos da la longitud del vector x.
- Sobre los vectores se pueden realizar operaciones que se realizan sobre todos los elementos del vector.

```
h[4]    # elemento 4 del vector h

## [1] 7

h[c(3,6)] # elementos 3 y 6 del vector h

## [1] 5 4

h[3:6]  # elementos del 3 al 6

## [1] 5 7 3 4

length(h) # longitud del vector h

## [1] 16

h<5     # elementos de h menores de 5

##  [1]  TRUE  TRUE FALSE FALSE  TRUE  TRUE FALSE FALSE  TRUE
TRUE FALSE FALSE  TRUE  TRUE FALSE FALSE

h+5     # suma 5 a TODOS los elementos del vector h

##  [1]  8  9 10 12  8  9 10 12  8  9 10 12  8  9 10 12
```

1.2.3. Matrices

Las matrices son un conjunto bidimensional de vectores del mismo tipo de datos. Los elementos de la matriz están ordenados por filas y columnas usando los subindices para citar a los individuos. Se crean con la función

$$matrix(data = NA,\ nrow = 1,\ ncol = 1,\ byrow = FALSE, dimnames = NULL)$$

donde

- data= conjunto de datos.

- nrow=número de filas, ncol=número de columnas.
- byrow=FALSE se ordena por columnas; si es TRUE se ordena por filas.

Las ordenes fundamentales sobre las matrices son:

```
i<-matrix(1:12,nrow=3,ncol=4)
#crea una matriz con elementos del 1 al 12 de 3 filas y 4 columnas
i

##      [,1] [,2] [,3] [,4]
## [1,]    1    4    7   10
## [2,]    2    5    8   11
## [3,]    3    6    9   12

j<-matrix(1:12,nrow=3,ncol=4,byrow=T)
#crea una matriz con elementos del 1 al 12 de tres filas y cuatro
columnas rellenando por filas
j

##      [,1] [,2] [,3] [,4]
## [1,]    1    2    3    4
## [2,]    5    6    7    8
## [3,]    9   10   11   12

dim(i)  # dimensiones de la matriz i

## [1] 3 4

i[,1]           # columnas 1 de la matriz i

## [1] 1 2 3

i[3,]           #fila 3 de la matriz i

## [1]  3  6  9 12

j[2,2]  # elemento de la fila 2 columna 2 de la matriz j

## [1] 6

rownames(i)=c("f1", "f2","f3") #asigna nombres a las filas
colnames(i)=c("c1","c2","c3","c4") # asigna nombre a las columnas
i
```

```
##    c1 c2 c3 c4
## f1  1  4  7 10
## f2  2  5  8 11
## f3  3  6  9 12

m<-rbind(i,c(1,2,3,4)) # añade a i la fila con los valores 1,2,3,4
m

##    c1 c2 c3 c4
## f1  1  4  7 10
## f2  2  5  8 11
## f3  3  6  9 12
##     1  2  3  4

m1<-cbind(i,c(1,2,3)) # añade a i la columna con los valores 1,2,3
m1

##    c1 c2 c3 c4
## f1  1  4  7 10 1
## f2  2  5  8 11 2
## f3  3  6  9 12 3

rbind(m,m) # crea una matriz con las matrices  m y m añadiendo filas

##    c1 c2 c3 c4
## f1  1  4  7 10
## f2  2  5  8 11
## f3  3  6  9 12
##     1  2  3  4
## f1  1  4  7 10
## f2  2  5  8 11
## f3  3  6  9 12
##     1  2  3  4

cbind(m1,m1) # crea una matriz con las matrices m1 y m1 añadiendo columnas

##    c1 c2 c3 c4   c1 c2 c3 c4
## f1  1  4  7 10 1  1  4  7 10 1
## f2  2  5  8 11 2  2  5  8 11 2
## f3  3  6  9 12 3  3  6  9 12 3
```

1.2.4. Arrays

Es un conjunto n dimensional de vectores o matrices. Sigue la formula:

$$array(data = NA,\ dim = length(data))$$

```
j<-array(1:24,dim=c(2,3,4))
j

## , , 1
##
##      [,1] [,2] [,3]
## [1,]    1    3    5
## [2,]    2    4    6
##
## , , 2
##
##      [,1] [,2] [,3]
## [1,]    7    9   11
## [2,]    8   10   12
##
## , , 3
##
##      [,1] [,2] [,3]
## [1,]   13   15   17
## [2,]   14   16   18
##
## , , 4
##
##      [,1] [,2] [,3]
## [1,]   19   21   23
## [2,]   20   22   24
```

1.2.5. Factor

Es un vector de variables categóricas (no numéricas) útil para identificar individuos

```
k<-c("h","h","m","m","m","h","h") #creamos una variable de sexo
fk<-factor(k) # con la orden factor la convertimos en un factor
levels(fk)
```

```
## [1] "h" "m"

pesok<-c(74,78,58,51,59,87,73) #pesos de esos individuos
pesok[fk=="h"] # podemos trabajar solo con los que sean h

## [1] 74 78 87 73
```

1.2.6. Data Frame

Un Data Frame es la estructura básica estadística. Se compone de vectores o factores todos de igual longitud. La información aparece en forma de tabla donde cada fila es un individúo y cada columna una variable.

Los elementos pueden ser de varios tipos y al crearse todos los caracteres se convierten en factores.

Se construye con la orden

$$d\text{ata.frame(vector1,vector2,.....vectork)}$$

```
dataframe<-data.frame(var1=c(10,20,30,40),var2=d,var3=c(1:4),var4=c("h","m","m","h"))
dataframe

##   var1 var2 var3 var4
## 1   10    5    1    h
## 2   20    4    2    m
## 3   30    3    3    m
## 4   40    2    4    h
```

si tuviéramos el nombre de las variables sería

```
individuos=(c("in1","in2","in3","in4"))
dataframe<-data.frame(df1=c(10,20,30,40),df2=d,df3=c(1:4),
                      sexo=c("h","m","m","h"),row.names=individuos)
dataframe

##     df1 df2 df3 sexo
## in1  10   5   1    h
## in2  20   4   2    m
## in3  30   3   3    m
## in4  40   2   4    h
```

Las características más importantes de los Data Frame son:

```
row.names(dataframe) # nombres de los individuos

## [1] "in1" "in2" "in3" "in4"

names(dataframe) # nombre de las variables

## [1] "df1"  "df2"  "df3"  "sexo"

dataframe$df1 # para hacer referencia a cada variable

## [1] 10 20 30 40

dataframe[1,3] # para hacer referencia al dato de la fila 1 columna 3

## [1] 1

dataframe[1]  # para hacer referencia a la variable 1

##     df1
## in1  10
## in2  20
## in3  30
## in4  40

dataframe[1,] # para hacer referencia al individuo 1

##     df1 df2 df3 sexo
## in1  10   5   1    h
```

1.2.7. Tablas

Se pueden crear factores a partir de variables cuantitativas con el comando cut

```
Edades<-c(26,21,30,31,25,38,41,32,28,45,42,19,36) # edades de 13 individuos
Edades2<-cut(Edades, breaks=3) # divide las edades en 3 intervalos
Edades2

##  [1] (19,27.7]   (19,27.7]   (27.7,36.3] (27.7,36.3] (19,27.7] (36.3,45] (36.3,45]
##  [8] (27.7,36.3] (27.7,36.3] (36.3,45]   (36.3,45]   (19,27.7] (27.7,36.3]
## Levels: (19,27.7] (27.7,36.3] (36.3,45]
```

```
table(Edades2) # calcula la frecuencia por intervalos

## Edades2
##   (19,27.7] (27.7,36.3]   (36.3,45]
##           4           5           4

Edades2

##  [1] (19,27.7]   (19,27.7]   (27.7,36.3] (27.7,36.3] (19,27.7] (36.3,45] (36.3,45]
##  [8] (27.7,36.3] (27.7,36.3] (36.3,45]   (36.3,45]   (19,27.7] (27.7,36.3]
## Levels: (19,27.7] (27.7,36.3] (36.3,45]

Edades3<-cut(Edades,breaks=c(18,29,39,49)) # límites de los intervalos
Edades3

##  [1] (18,29] (18,29] (29,39] (29,39] (18,29] (29,39] (39,49] (29,39]
(18,29] (39,49] (39,49] (18,29] (29,39]
## Levels: (18,29] (29,39] (39,49]

table(Edades3)

## Edades3
## (18,29] (29,39] (39,49]
##       5       5       3

Edades3

##  [1] (18,29] (18,29] (29,39] (29,39] (18,29] (29,39] (39,49] (29,39] (18,29] (39,49] (39,49]
## Levels: (18,29] (29,39] (39,49]
```

Por ejemplo para crear tablas multidimensionales

```
altura<-c(178,168,181,178,169,168,175,178,169,181,179,165,160,178)
#alturas de 14 individuos
peso<-c(81,59,82,76,61,58,79,87,58,81,72,54,50,83)  #peso de 14 individuos
edad<-c(21,22,21,23,24,25,26,21,22,21,23,21,22,24)   # edad 14 individuos
taltura<-cut(altura,breaks=c(150,160,170,180,190))   # creamos el factor altura
tpeso<-cut(peso,breaks=c(50,60,70,80,90))   # creamos el factor peso
tedad<-cut(edad,breaks=c(20,24,28))       # creamos el factor edad
table(taltura,tpeso,tedad)

## , , tedad = (20,24]
```

```
##
##            tpeso
## taltura     (50,60] (60,70] (70,80] (80,90]
##   (150,160]       0       0       0       0
##   (160,170]       3       1       0       0
##   (170,180]       0       0       2       3
##   (180,190]       0       0       0       2
##
## , , tedad = (24,28]
##
##            tpeso
## taltura     (50,60] (60,70] (70,80] (80,90]
##   (150,160]       0       0       0       0
##   (160,170]       1       0       0       0
##   (170,180]       0       0       1       0
##   (180,190]       0       0       0       0

#obtenemos la tabla de alturas y pesos por cada intervalos de edad
```

1.3. Apertura de ficheros

En esta sección describiremos brevemente como abrir ficheros de algunas de las fuentes más comunes que podemos encontrarnos, concretamente ficheros de tipo texto; de Excell o de SPSS.

1.3.1. Apertura fichero de texto

La sintaxis para abrir ficheros de texto es la siguiente:

```
read.table(file, header = FALSE, sep = "", quote = "\"'", dec = ".",
numerals = c("allow.loss", "warn.loss", "no.loss"),  row.names, col.names,
as.is = !stringsAsFactors, na.strings = "NA", colClasses = NA, nrows = -1,
skip = 0, check.names = TRUE, fill = !blank.lines.skip,
strip.white = FALSE, blank.lines.skip = TRUE,
comment.char = "#", allowEscapes = FALSE, flush = FALSE,
stringsAsFactors = default.stringsAsFactors(),
fileEncoding = "", encoding = "unknown", text, skipNul = FALSE)
```

Donde los principales atributos de esta función son:

- file: dirección donde se encuentra el fichero que deseamos abrir.
- header=TRUE: el fichero contiene el nombre de las variables. En caso contrario pondremos F (FALSE).
- sep: el separador entre los datos, por defecto es un espacio en blanco "" pero puede ser doble espacio, tabulador...
- Row.names y col.names si tenemos un fichero con los nombres de los individuos y variables.
- na.string='NA': los caracteres se consideran con valores NA y en las variables numéricas como valores perdidos.

Por ejemplo vamos a abrir un fichero de tipo texto denominado datos.txt que se encuentra en la plataforma:

```
datos <- read.table("D:/Escritorio/tema 0 master/datos/datos.txt",
                    header = TRUE, sep = "", na.strings = "NA", dec = ".",
                    strip.white = TRUE)
datos
```

```
##    nombre sexo edad peso altura
## 1    Juan    H   26   81    179
## 2  Andres    H   21   77    176
## 3 Yolanda    M   22   56    167
## 4     Eva    M   22   53    165
## 5 Antonio    H   27   76    179
```

1.3.2. Apertura ficheros Excell

PAra abrir un fichero de tipo Excell, la orden es:

```
read_excel(path, sheet = NULL, range = NULL, col_names = TRUE,
col_types = NULL, na = "", trim_ws = TRUE, skip = 0,
n_max = Inf, guess_max = min(1000, n_max),
progress = readxl_progress(), .name_repair = "unique")
```

Los principales atributos son:

- path: ruta al fichero.
- colnames=True: si tiene los nombres del las variables.

Abrimos el fichero datosexcel.xlsx

```
library("readxl")
datosExcel <- read_excel("D:/Escritorio/tema 0 master/datos/datosexcel.xlsx")
datosExcel

## # A tibble: 5 x 5
##   nombre  sexo   edad  peso altura
##   <chr>   <chr> <dbl> <dbl>  <dbl>
## 1 juan    h        NA    81    179
## 2 andres  h        21    77    176
## 3 yolanda m        22    56    167
## 4 eva     m        22    53    165
## 5 antonio h        27    76    179
```

1.3.3. Ficheros de SPSS

La sinstasis para abrir ficheros de SPSS es:

```
read.spss(file, use.value.labels = TRUE, to.data.frame = FALSE,
max.value.labels = Inf, trim.factor.names = FALSE,
trim_values = TRUE, reencode = NA, use.missings = to.data.frame,
sub = ".", add.undeclared.levels = c("sort", "append", "no"),
duplicated.value.labels = c("append", "condense"),
duplicated.value.labels.infix = "_duplicated_", ...)
```

Abriremos el fichero datos.sav

```
library(foreign)
datosSPSS <- read.spss("D:/Escritorio/tema 0 master/datos/datos.sav", to.data.frame=TRUE,
use.value.labels = TRUE)
 head(datosSPSS)

##         CCAA       P0      p1       p2                                        p301
## 1 País Vasco Española Regular     Mala 0 Con toda seguridad, no lo votaría nunca
## 2 País Vasco Española Regular  Regular                                          7
## 3 País Vasco Española   Buena     Mala                                          6
## 4 País Vasco Española    Mala Muy mala                                          4
## 5 País Vasco Española    Mala     Mala                                          6
## 6 País Vasco Española Regular Muy mala 0 Con toda seguridad, no lo votaría nunca
##                                       p302      p401      p402     p5 p6       p7
## 1 0 Con toda seguridad, no lo votaría nunca         3 1 Muy mal  Mujer 61 Casado/a
```

```
## 2                                          5         8         4  Mujer 71 Casado/a
## 3                                          3         5 1 Muy mal  Mujer 42 Casado/a
## 4                                          7         4         6 Hombre 70 Casado/a
## 5 0 Con toda seguridad, no lo votaría nunca         7 1 Muy mal  Mujer 56 Casado/a
## 6 0 Con toda seguridad, no lo votaría nunca 1 Muy mal 1 Muy mal  Mujer 62 Casado/a
```

Parte 2

Análisis previos de los datos

Vamos a utilizar un fichero bastante descrito en la bibliografía de análisis multivariante que es el fichero car.txt. Este fichero ontiene los datos para 398 modelos de vehículos y diferentes características. También tendremos el mismo fichero con los precios de esos coches. Si abrimos los ficheros tendremos:

```
coches <- read.table("D:/Escritorio/tema 0 master/datos/car.txt",header = TRUE,
sep = ",", na.strings = "?", dec = ".", strip.white = TRUE)
precios<- read.table("D:/Escritorio/tema 0 master/datos/precios.txt",header = TRUE,
sep = ",", na.strings = "?", dec = ".", strip.white = TRUE)
#cada alumno pondrá la dirección donde lo tenga almacenado
#podemos ver los cabeceras de los ficheros

head(coches)
```

```
##   ID consumo cilindros desplazamiento  CV peso aceleracion fecha
## 1  1      18         8            307 130 3504        12.0    70
## 2  2      15         8            350 165 3693        11.5    70
## 3  3      18         8            318 150 3436        11.0    70
## 4  4      16         8            304 150 3433        12.0    70
## 5  5      17         8            302 140 3449        10.5    70
## 6  6      15         8            429 198 4341        10.0    70

##  origen                    mdoelo
##  1 chevrolet chevelle malibu
##  1         buick skylark 320
##  1        plymouth satellite
##  1             amc rebel sst
##  1               ford torino
##  1          ford galaxie 500
```

```
head(precios)

##   ID   precio
## 1  1 25561.59
## 2  2 24221.42
## 3  3 27240.84
## 4  4 33684.97
## 5  5 20000.00
## 6  6 30000.00
```

El fichero coches contiene para 398 modelos de vehículos el consumo, los cilindros, km de autonomía, CV, el peso, la aceleración, el año de producción, el origen y el modelo el fichero precios el valor de esos vehículos en el mismo orden. También un código de identificación

Nuestro primer paso será el de unificar estos ficheros, para ellos además de las órdenes vistas en el capítulo anterior podemos usar la library(dplyr) y la orden join (puede ser usada por la derecha, izquierda...) en donde indicaremos la variable que los identifica.

```
library(dplyr)
coches2<-left_join(coches,precios,by="ID")
#el dataframe coches2 contiene unificado los dos ficheros
```

2.1. Estadadísticos descriptivos

El primer paso antes de realizar cualquier tipo de análisis estadístico es el resumen y la depuración de los datos. Para ello son útiles las herramientas de la estadística descriptiva. Como primer paso hemos de saber que tipo de datos tenemos y un breve resumen de la información de las variables:

```
# con esta primera opción vemos el tipo de dato que es cada variable: entero,
                                          numérico o de cadena de caracteres
str(coches2)

## 'data.frame': 398 obs. of  11 variables:
##  $ ID            : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ consumo       : num  18 15 18 16 17 15 14 14 14 15 ...
##  $ cilindros     : int  8 8 8 8 8 8 8 8 8 8 ...
##  $ desplazamiento: num  307 350 318 304 302 429 454 440 455 390 ...
##  $ CV            : int  130 165 150 150 140 198 220 215 225 190 ...
##  $ peso          : int  3504 3693 3436 3433 3449 4341 4354 4312 4425 3850 ...
```

```
##  $ aceleracion   : num  12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
##  $ fecha         : int  70 70 70 70 70 70 70 70 70 70 ...
##  $ origen        : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ mdoelo        : chr  "chevrolet chevelle malibu" "buick skylark 320"  ...
##  $ precio        : num  25562 24221 27241 33685 20000 ...
```

```
# breve resumen con los estadísticos básicos
summary(coches2)
```

```
##       ID           consumo     cilindros   desplazamiento        CV             peso
## Min.   :  1.0 Min.   : 9.00  Min.   :3.000 Min.   : 68.0 Min.   : 46.0   Min.   :1613
## 1stQu.:100.2 1st Qu.:17.50  1st Qu.:4.000 1st Qu.:104.2 1st Qu.: 75.0   1st Qu.:2224
## Median:199.5 Median :23.00  Median :4.000 Median :148.5 Median : 93.5   Median :2804
## Mean   :199.5 Mean   :23.51  Mean   :5.455 Mean   :193.4 Mean   :104.5   Mean   :2970
## 3rdQu.:298.8 3rd Qu.:29.00  3rd Qu.:8.000 3rd Qu.:262.0 3rd Qu.:126.0   3rd Qu.:3608
## Max.   :398.0 Max.   :46.60  Max.   :8.000 Max.   :455.0 Max.   :230.0   Max.   :5140
##                                                                NA's   :6
##   aceleracion        fecha           origen         mdoelo              precio
##  Min.   : 8.00   Min.   :70.00   Min.   :1.000   Length:398         Min.   : 1598
##  1st Qu.:13.82   1st Qu.:73.00   1st Qu.:1.000   Class :character   1st Qu.:23110
##  Median :15.50   Median :76.00   Median :1.000   Mode  :character   Median :30000
##  Mean   :15.57   Mean   :76.01   Mean   :1.573                      Mean   :29684
##  3rd Qu.:17.18   3rd Qu.:79.00   3rd Qu.:2.000                      3rd Qu.:36430
##  Max.   :24.80   Max.   :82.00   Max.   :3.000                      Max.   :53746
##
```

Como podemos observar hay alguna variable que no se ha importado de forma correcta o no como desearíamos. Por ejemplo las variables Cilindros, CV, Peso, Aceleación y precio deberían ser un numéricas y las variables Origen y Fecha factores. Concretamente la variable origen tiene como valores 1 en el caso de ser de EEUU, 2 si es Europeo y 3 si es Japones. Nuestro siguiente paso será recodificar esas variables:

```
#creamos la variable con las etiquetas y las añadimos
origen<- c("EEUU","Europa","Japon")
#convertimos la variable origen en un factor con la original y las etiquetas de origen
coches2$origen<-factor(coches2$origen,labels=origen2)

# convertiremos la variable fecha en un factor
coches2$fecha<-as.factor(coches2$fecha)
# el resto de variables las convertimos en numéricas
```

```
coches2$cilindros<-as.numeric(coches2$cilindros)
coches2$CV<-as.numeric(coches2$CV)
coches2$peso<-as.numeric(coches2$peso)
coches2$aceleracion<-as.numeric(coches2$aceleracion)
coches2$precio<-as.numeric(coches2$precio)
str(coches2)

## 'data.frame': 398 obs. of  11 variables:
##  $ ID            : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ consumo       : num  18 15 18 16 17 15 14 14 14 15 ...
##  $ cilindros     : num  8 8 8 8 8 8 8 8 8 8 ...
##  $ desplazamiento: num  307 350 318 304 302 429 454 440 455 390 ...
##  $ CV            : num  130 165 150 150 140 198 220 215 225 190 ...
##  $ peso          : num  3504 3693 3436 3433 3449 ...
##  $ aceleracion   : num  12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
##  $ fecha         : Factor w/ 13 levels "70","71","72",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ origen        : Factor w/ 3 levels "EEUU","Europa",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ mdoelo        : chr  "chevrolet chevelle malibu" "buick skylark 320"  ...
##  $ precio        : num  25562 24221 27241 33685 20000 ...
```

Otro paso previo antes de empezar a trabajar es determinar si existen valores perdidos en alguna variable:

```
which(is.na(coches2))
# nos indica que individuos son perdidos

## [1] 1625 1719 1923 1929 1947 1967

colSums(is.na(coches2))

##   ID        consumo  cilindros desplazamiento     CV           peso    aceleracion
##    0              0          0              0      6              0              0
##  aceleracion fecha      origen     mdoelo         precio
##    0              0            0          0             0

head(coche2)

## Error in head(coche2): object 'coche2ñot found
```

Podemos ver como hay 6 valores perdidos en la variable CV y los individuos que son. Dependiendo de las circunstancias podemos sustituirlos por algún valor, eliminar ese individuo o que trabaje R y

en los casos en los que sea necesario no utilice ese individuos.

```
# si decidimos eliminar los individuos la muestra se reduciría en 6 unidades
coches3<-na.omit(coches2)
dim(coches2)
#el fichero coches2 tienes 398 individuos en 11 variables

## [1] 398  11

dim(coches3)
#el fichero coches3 tienes 398 individuos en 11 variables

## [1] 392  11
```

Ya comentamos que con la orden summary podemos obtener un breve resumen de cada variable. Si estuviéramos interesados en otras medidas descriptivas como la moda, desviación típica...

```
library(modeest)
c(mean(coches3$CV),sd(coches3$CV),IQR(coches3$CV),mfv(coches3$CV),
quantile(coches3$CV,c(0.25,0.5,0.75),na.rm = TRUE))

##                                              25%       50%       75%
## 104.46939  38.49116  51.00000 150.00000  75.00000  93.50000 126.00000

c(range(coches3$CV),min(coches3$CV),max(coches3$CV),var(coches3$CV))

## [1]   46.000  230.000   46.000  230.000 1481.569
```

2.1.1. Tablas de frecuencias

En las variables cualitativas (de factor) suele ser común obtener la tabla de frecuencias para poder obtener información de tipo descritivo. en este caso trabajaremos con la variable Origen.

```
table(coches3$origen)#para obtener la tabla de frecuencias

##
##   EEUU Europa  Japon
##    245     68     79

# porcentajes
prop.table(table(coches3$origen))
```

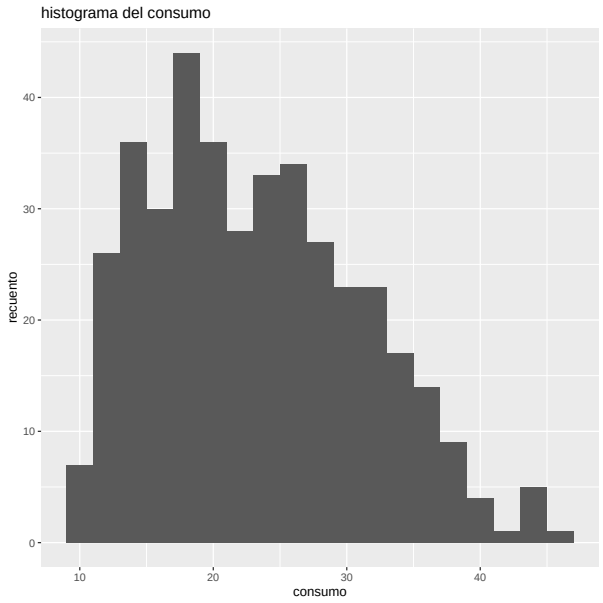
```
##
##      EEUU    Europa     Japon
## 0.6250000 0.1734694 0.2015306

#porcentajes en tanto por ciento
prop.table(table(coches3$origen))*100

##
##     EEUU   Europa    Japon
## 62.50000 17.34694 20.15306

#frecuencias acumuladas
cumsum(table(coches3$origen))

##   EEUU Europa  Japon
##    245    313    392

cumsum(prop.table(table(coches3$origen)))

##      EEUU    Europa     Japon
## 0.6250000 0.7984694 1.0000000
```

2.2. Representaciones gráficas

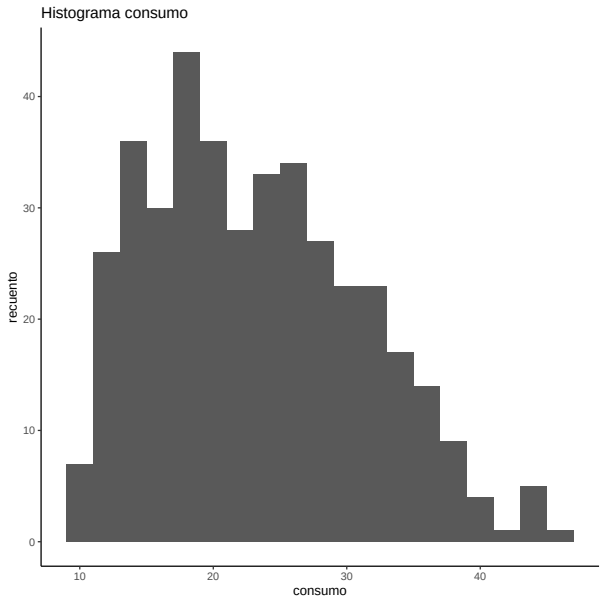
En esta sección se va a utilizar la biblioteca library(ggplot2) y dentro de ellas algunas funciones como son qplot o ggplot.

2.2.1. Histograma

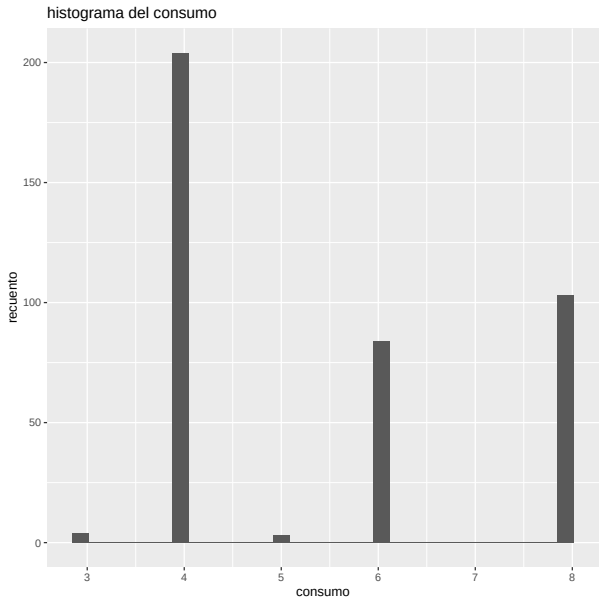
```
library(ggplot2)
#vamos a realizar un histograma del consumo de combustible
qplot(coches2$consumo, xlab = 'consumo', ylab = 'recuento', binwidth = 2,
      main='histograma del consumo')
```




```
# otra opción de hacerlo es la orden ggplot donde se van añadiendo características con la orden
#  con la orden +
ggplot(coches2, aes(consumo))    +
  geom_histogram(binwidth = 2) +
  labs(title = "Histograma consumo ", y = "recuento")  +
  theme_classic()
```




```
qplot(coches2$cilindros, xlab = 'consumo', ylab = 'recuento',
      main='histograma del consumo')
```

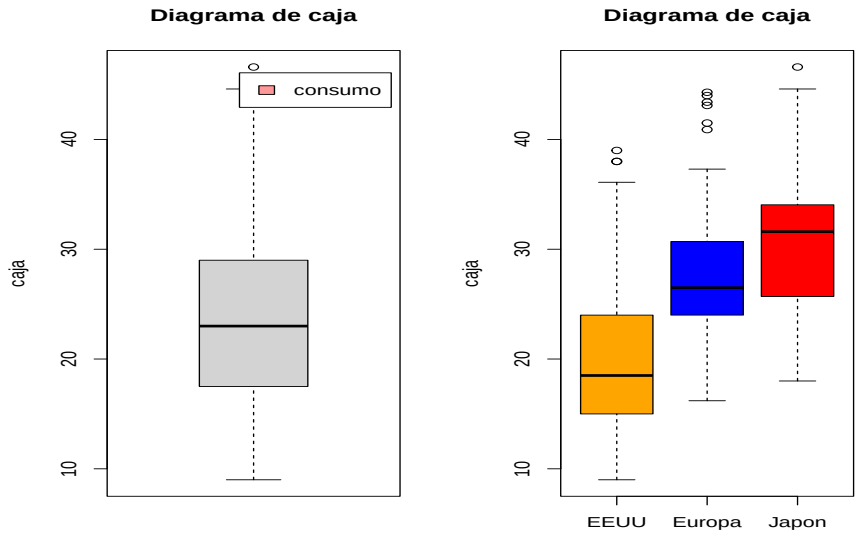


en estos histogramas podemos ver como se alejan de la curva de la normal (la campana de Gauss).

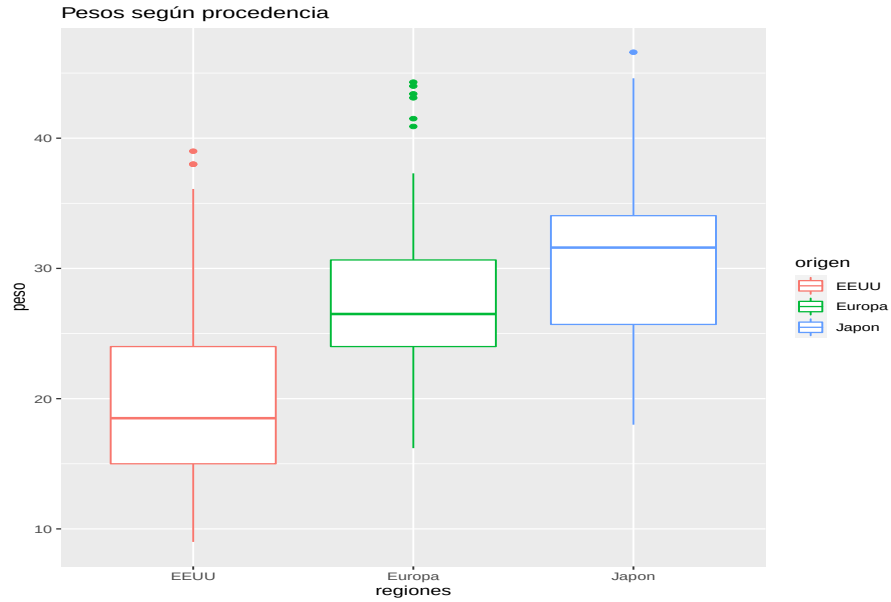
2.2.2. Diagramas de caja

El diagrama de caja con bigotes,es muy útil para la detección de puntos anómalos se puede obtener mediante la orden boxplot o en el paquete ggplot con la orden geom_boxplot()

```
# En primer lugar activamos una salida donde podamos crear dos gráficos
#  dentro de la misma imagen
par(mfrow = c(1, 2))
# en este caso tendremos el diagrama de caja del consumo
boxplot(coches2$consumo, main="Diagrama de caja",  xlab="", ylab="caja")
legend("topright", legend = "consumo", # Posición y título
       fill = rgb(1, 0, 0, alpha = 0.4),  # Color
       inset = c(0.03, 0.05), # Cambiamos los márgenes
       bg = "white") # Color de fondo de la leyenda

# si queremos hacerlo según una variable de factor y añadir colores
boxplot(coches2$consumo~coches2$origen ,main="Diagrama de caja",  xlab="",
ylab="caja", col=c("orange","blue","red"))
```

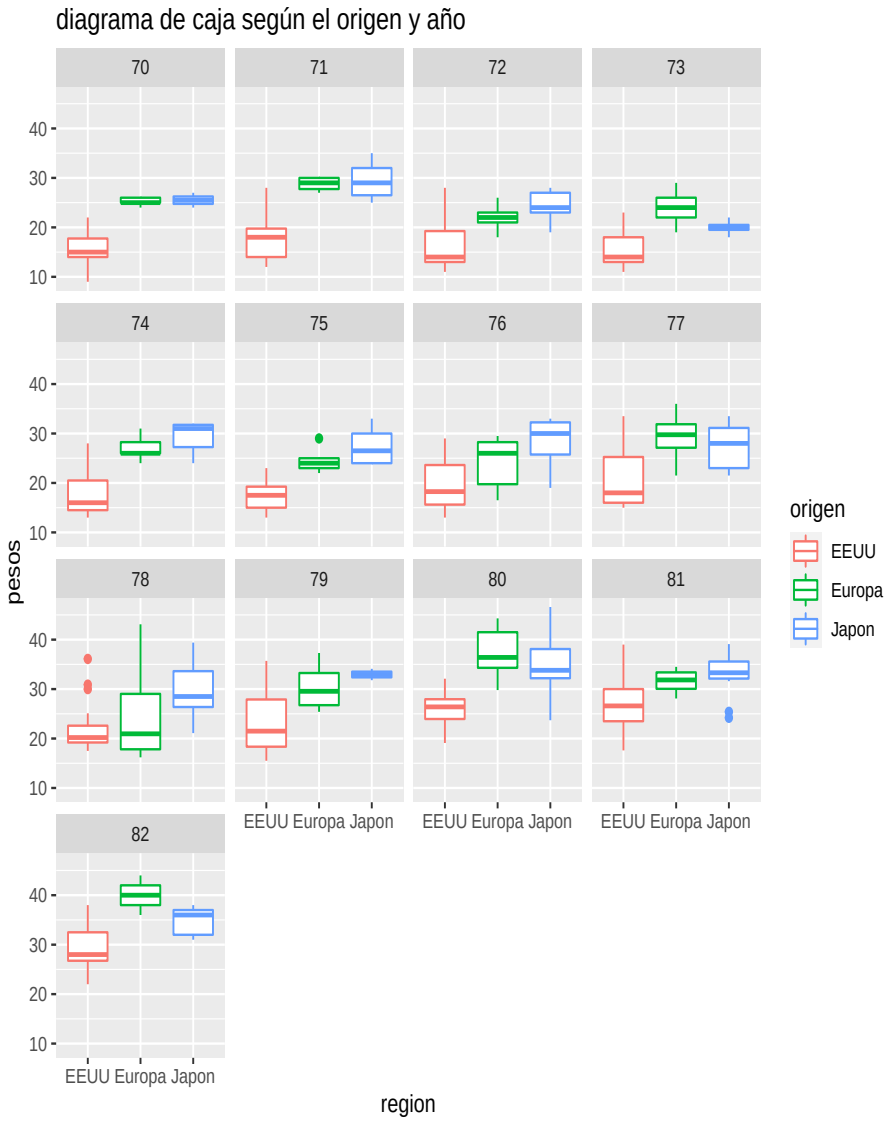



```
#Otra forma de realizar los diagramas de caja
ggplot(data = coches2, aes(x = origen, y = consumo,col=origen)) +
          geom_boxplot() +
          xlab('regiones') +
          ylab('peso') +
          ggtitle('Pesos según procedencia')
```




```
#Si quisiéramos añadir el año de procedencia
ggplot(data = coches2, aes(x = origen, y = consumo,col=origen)) +
          geom_boxplot() +
```

```
        xlab('region') +
  facet_wrap(~ fecha)+
        ylab('pesos') +
        ggtitle('diagrama de caja según el origen y año')
```



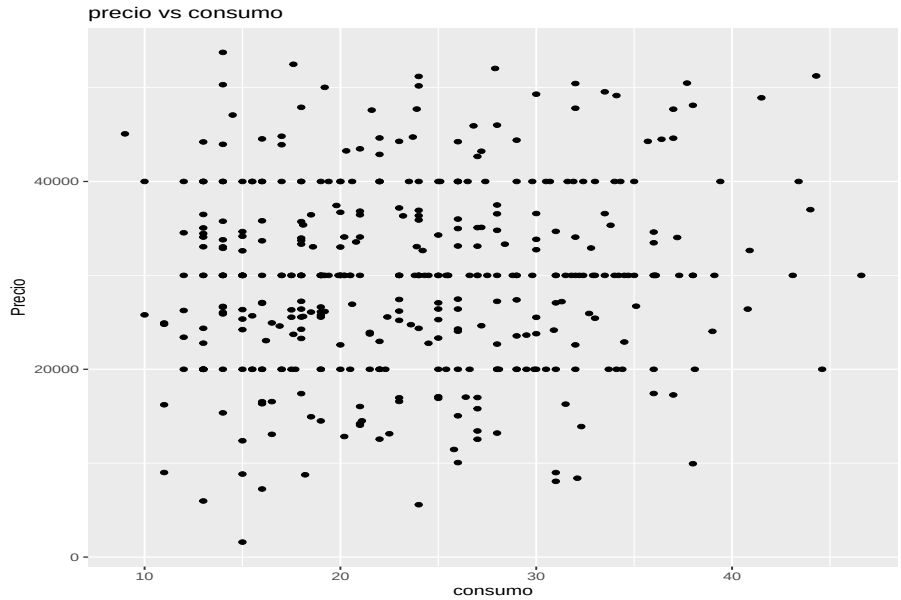
En esta opción podemos ver como el peso de los Japones es mayor que los Europeos y los Americanos y los valores anómalos de cada uno de ellos.

2.2.3. Diagramas de puntos

Suele ser muy útil ver los gráficos bidimensionales para poder detectar anomalías

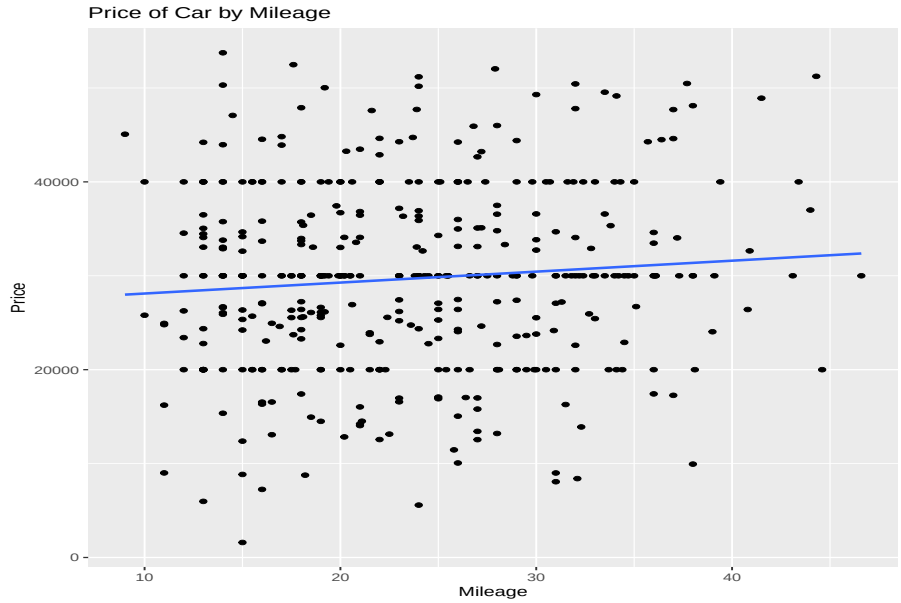
```
ggplot(coches2, aes(x = consumo, y = precio)) +
  geom_point() +
```

```
  xlab("consumo") +
  ylab("Precio") +
  ggtitle("precio vs consumo")
```



Si quisiéramos añadir la recta de regresión:

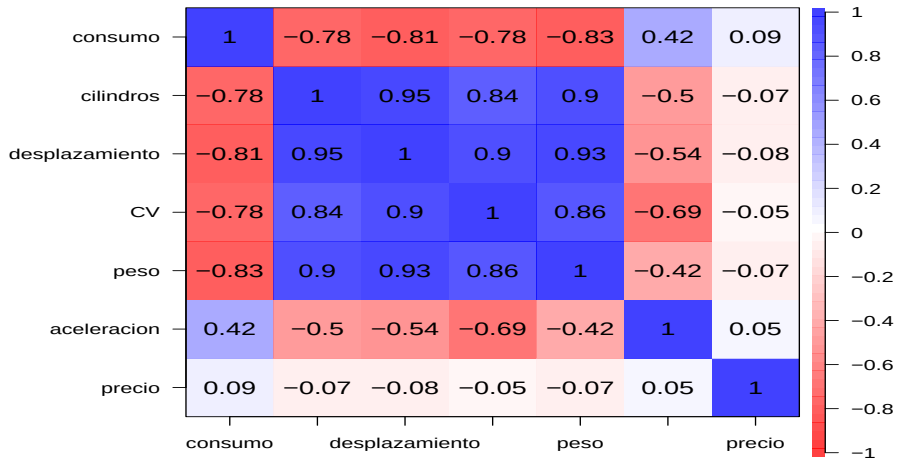
```
ggplot(coches2, aes(x = consumo, y = precio)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  xlab("Mileage") +
  ylab("Price") +
  ggtitle("Price of Car by Mileage")
```



2.2.4. Correlaciones

También es útil como medida descriptiva analizar las correlaciones de las variables:

```
vehiculos <- coches2[complete.cases(coches2),]# quitamos los valores perdidos
vehiculos <- cars[,-1] # quitammos la variable identificadora
vehiculos2 <- sapply(vehiculos, is.numeric) # determinamos cuales son numéricas
cor(vehiculos[,vehiculos2])
# calculamos las correlaciones entre vehículos para las variables que son numéricas

##                cilindros esplazamiento       CV      peso aceleracion    precio
##cilindros       1.0000000  0.9508233  0.8429833  0.8975273 -0.5046833 -0.0653417
##desplazamiento  0.9508233  1.0000000  0.8972570  0.9329944 -0.5438005 -0.0815681
##CV              0.8429833  0.8972570  1.0000000  0.8645377 -0.6891955 -0.0537003
##peso            0.8975273  0.9329944  0.8645377  1.0000000 -0.4168392 -0.0721973
##aceleracion    -0.5046833 -0.5438005 -0.6891955 -0.4168392  1.0000000  0.0482044
##precio         -0.0653417 -0.0815681 -0.0537003 -0.0721973  0.0482044  1.0000000
##


library(psych)
corPlot(correlations, order = "hclust") #obtenemos un gráfico de calor con las p-valores
```



Parte 3

Hipótesis básicas en Análisis Multivariante

Casi todas las técnicas de análisis multivariante requieren el estudio de una serie de supuestos básicos. Algunas técnicas son muy exigentes en el estudio y cumplimiento de estas hipótesis. Aunque en cada tema se estudiarán los supuestos básicos que ha de cumplir cada técnica, dada la importancia de alguna de ellas se estudiarán de forma independiente en este tema.

3.1. Normalidad

Aunque la hipótesis que hemos de verificar es la normalidad multivariante, en primer lugar estudiaremos la normalidad univariante. Es importante destacar que el que todas las variables sean normales univariantemente no implica que lo sean de forma multivariante.

3.1.1. Normalidad univariante

Una primer acercamiento a esta hipótesis consiste en comprobar la asimetría y curtosis de las distribuciones. Si la distribución fuera normal, estos dos valores deben ser 0.

```
library(pastecs)


#lo hacemos con la variable vehículos que hemos seleccionado antes
round(stat.desc(vehiculos[,vehiculos2],basic=FALSE,norm=TRUE),digits=3)


##              cilindros desplazamiento       CV       peso aceleracion       precio
## median           4.000        151.000   93.500   2803.500      15.500    30000.000
## mean             5.472        194.412  104.469   2977.584      15.541    29659.660
## SE.mean          0.086          5.285    1.944     42.901       0.139      499.411
## CI.mean.0.95     0.169         10.391    3.822     84.346       0.274      981.868
## var              2.910      10950.368 1481.569 721484.709       7.611 97769434.821
```

```
## std.dev          1.706        104.644   38.491    849.403       2.759     9887.843
## coef.var         0.312          0.538    0.368      0.285       0.178        0.333
## skewness         0.504          0.696    1.079      0.516       0.289        0.024
## skew.2SE         2.046          2.825    4.377      2.092       1.174        0.097
## kurtosis        -1.404         -0.795    0.654     -0.825       0.406       -0.298
## kurt.2SE        -2.855         -1.617    1.330     -1.678       0.825       -0.606
## normtest.W       0.751          0.882    0.904      0.941       0.992        0.986
## normtest.p       0.000          0.000    0.000      0.000       0.031        0.001
```

```
#otra forma de hacerlo es indicando las variables
round(stat.desc(coches2[,c("cilindros","desplazamiento","CV","peso","aceleracion","precio")],
basic=FALSE,norm=TRUE),digits=3)
```

```
##              cilindros desplazamiento       CV       peso aceleracion       precio
## median           4.000        148.500   93.500   2803.500      15.500    30000.000
## mean             5.455        193.426  104.469   2970.425      15.568    29684.473
## SE.mean          0.085          5.227    1.944     42.448       0.138      492.847
## CI.mean.0.95     0.168         10.275    3.822     83.452       0.272      968.916
## var              2.893      10872.199 1481.569 717140.991       7.605 96673401.656
## std.dev          1.701        104.270   38.491    846.842       2.758     9832.263
## coef.var         0.312          0.539    0.368      0.285       0.177        0.331
## skewness         0.523          0.714    1.079      0.527       0.277        0.018
## skew.2SE         2.138          2.919    4.377      2.154       1.131        0.075
## kurtosis        -1.383         -0.764    0.654     -0.802       0.382       -0.277
## kurt.2SE        -2.833         -1.564    1.330     -1.643       0.783       -0.567
## normtest.W       0.749          0.880    0.904      0.941       0.992        0.987
## normtest.p       0.000          0.000    0.000      0.000       0.040        0.001
```

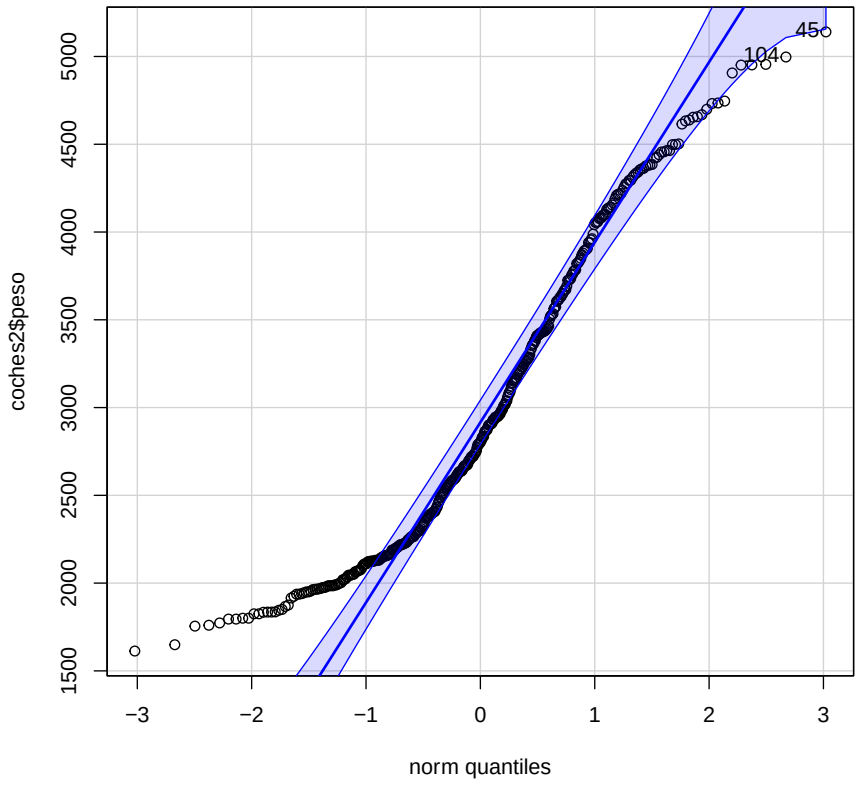
```
#la orden round redondea
```

En este caso valores superiores, en valor absoluto a 1, en skew.2se o kurt.2se indicarán que se aleja de la normalidad por lo que a priori parece que no verifican la hipótesis de normalidad.

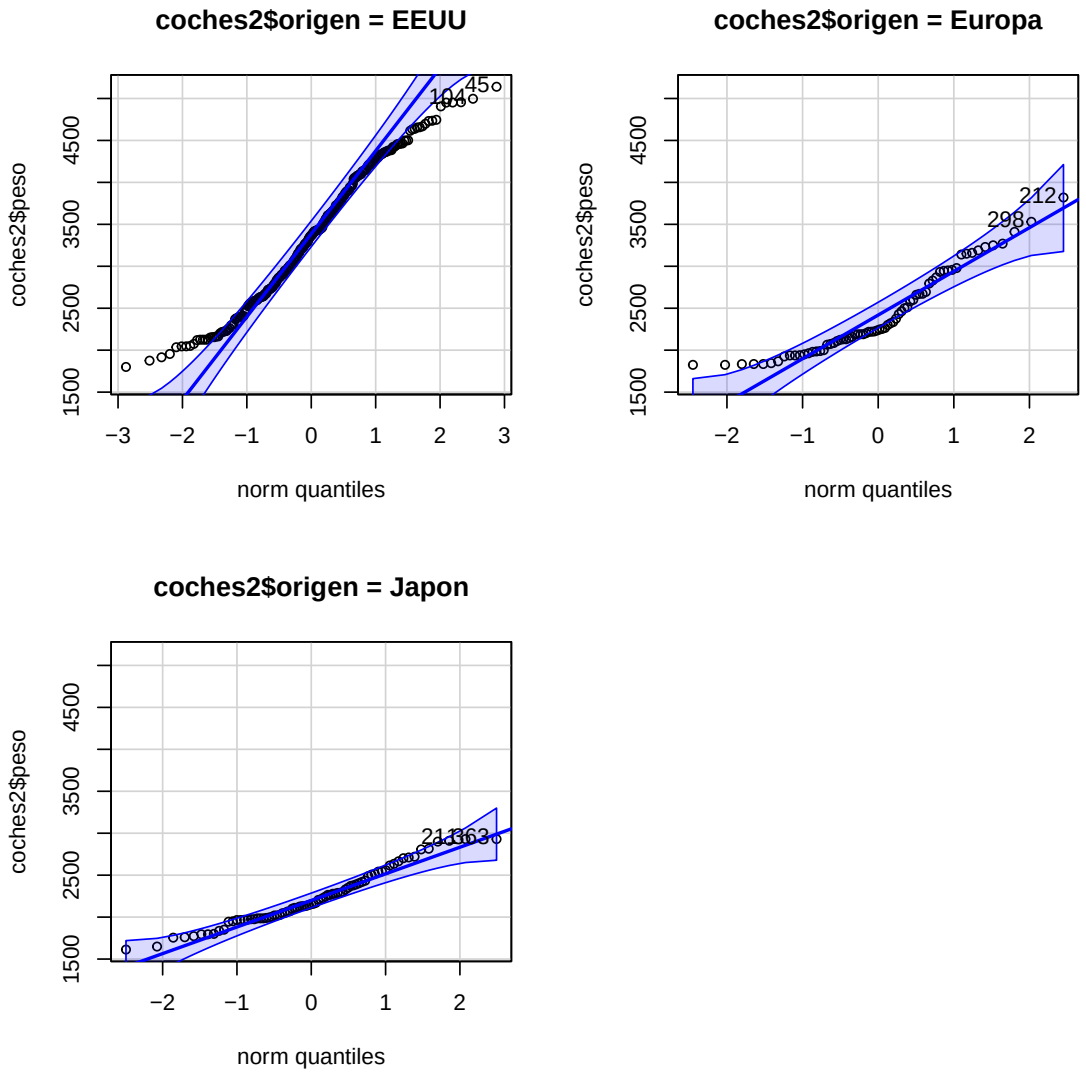
Otra forma de comprobar la normalidad es mediante los gráficos Q-Q, aunque, como en la mayor parte de los gráficos, la interpretación gráfica es muy subjetiva.

Por ejemplo estudiaremos el gráfico qq de forma individual y por los grupos marcados por la variable origen:

```
library(car)


qqPlot(coches2$peso)
```



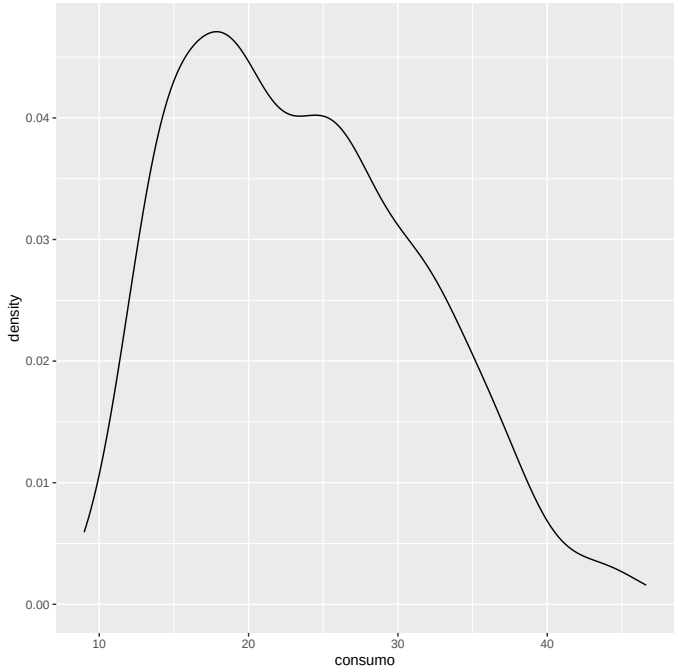

```
qqPlot(coches2$peso, groups = coches2$origen)
```



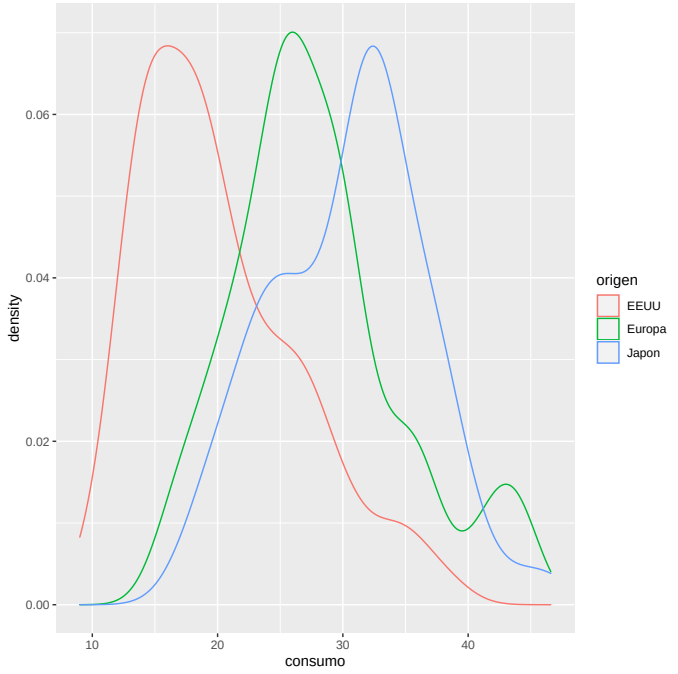
Tanto para la variable Precio, como para los casos de fabricación americana los valores distan mucho de acercarse a los cuartiles de la distribución normal (recta del interior de los gráficos), para el precio de los fabricados en Europa y Japón si parece que se cumpla la hipótesis de normalidad.

También podemos obtener el gráfico de la densidad de frecuencias para comprobar si se aproximan a la normalidad

```
library(ggplot2)
ggplot(coches2, aes(consumo)) +geom_density()
```




```
ggplot(coches2, aes(consumo,color=origen)) +geom_density()
```



Una vez visto que parece improbable que se cumpla la normalidad, estudiaremos el mejor método para comprobarlo, un test de hipótesis.

De forma muy rápida podríamos decir que un test de hipótesis es un procedimiento estadístico para decidir entre una hipótesis a la que llamamos hipótesis nula H_0 o su contraria denominada

hipótesis alternativa H_1. El procedimiento en la práctica para resolver es si el p-valor asociado al caso concreto es menor que un nivel α fijado con anterioridad. Este valor α es la probabilidad de que elijamos H_1 siendo cierta H_0. Este valor generalmente es de $\alpha = 0.05$ (contraste del hipótesis al 95 %).

Para contrastar la normalidad se suelen usar dos estadísticos clásicos, el de Shapiro-Wilks o el de Kolmogorov-Smirnof. El primero para muestras pequeñas (menor de 50) y el segundo para muestras grandes. Aunque existen otros test estos son los mas comunes.

```
library(nortest)
vehiculosnumerico<-coches2[,c("cilindros",
"desplazamiento","CV","peso","aceleracion","precio")]
shapiro.test(coches2$cilindros) # solo lo hace para una variable
```

```
##
##  Shapiro-Wilk normality test
##
## data:  coches2$cilindros
## W = 0.74873, p-value < 2.2e-16
```

```
lapply(vehiculosnumerico, shapiro.test) # para hacer para una matriz
```

```
## $cilindros
##
##  Shapiro-Wilk normality test
##
## data:  X[[i]]
## W = 0.74873, p-value < 2.2e-16
##
##
## $desplazamiento
##
##  Shapiro-Wilk normality test
##
## data:  X[[i]]
## W = 0.88042, p-value < 2.2e-16
##
```

```
##
## $CV
##
##  Shapiro-Wilk normality test
##
## data:  X[[i]]
## W = 0.9041, p-value = 5.022e-15
##
##
## $peso
##
##  Shapiro-Wilk normality test
##
## data:  X[[i]]
## W = 0.94136, p-value = 1.97e-11
##
##
## $aceleracion
##
##  Shapiro-Wilk normality test
##
## data:  X[[i]]
## W = 0.99238, p-value = 0.03987
##
##
## $precio
##
##  Shapiro-Wilk normality test
##
## data:  X[[i]]
## W = 0.98656, p-value = 0.0009603


lapply(vehiculosnumerico, lillie.test) #


## $cilindros
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
```

```
##
## data:  X[[i]]
## D = 0.32641, p-value < 2.2e-16
##
##
## $desplazamiento
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  X[[i]]
## D = 0.18308, p-value < 2.2e-16
##
##
## $CV
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  X[[i]]
## D = 0.16357, p-value < 2.2e-16
##
##
## $peso
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  X[[i]]
## D = 0.093434, p-value = 6.95e-09
##
##
## $aceleracion
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  X[[i]]
## D = 0.050837, p-value = 0.01528
##
##
## $precio
```

```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  X[[i]]
## D = 0.11534, p-value = 7.381e-14
```

En estos contrastes, que salían en la parte final del sumario estadistico que se obtuvo anteriormente, podemos ver como ninguna de las variables son normales.

```
ggplot(coches2$precio,coches2$origen,shapiro.tes)


by(coches2$precio,coches2$origen,shapiro.test)
coches2$origen: 1

Shapiro-Wilk normality test

data:  dd[x, ]
W = 0.9875, p-value = 0.02901

------------------------------------------------------------------------------
coches2$origen: 2

Shapiro-Wilk normality test

data:  dd[x, ]
W = 0.98003, p-value = 0.3269

------------------------------------------------------------------------------
coches2$origen: 3

Shapiro-Wilk normality test

data:  dd[x, ]
W = 0.97454, p-value = 0.1152
```

Como vimos en las representaciones gráficas el precio para los vehículos fabricados en Europa y Japón si siguen una distribución normal.

3.1.2. Normalidad Multivariante

Existen diferentes test para contrastar la normalidad multivariante, como son los test de MArdia, Henze-Zirkler o Royston. PAra ello usamos la biblioteca MVN

```
library(MVN)
library(MSQC)
mvn(vehiculosnumerico,univariateTest = "Lillie",multivariatePlot="qq")
# con este expresión realiza el test de Henze-Zirkler
```

```
mvn(vehiculosnumerico,mvnTest = "mardia",univariateTest = "Lillie")
# con esta exppresión se realzia el test de mardia
```

```
## $multivariateNormality
##              Test        Statistic               p value Result
## 1 Mardia Skewness 713.377657544392 1.00212509328992e-114     NO
## 2 Mardia Kurtosis 18.1282020438843                     0     NO
## 3             MVN             <NA>                  <NA>     NO
##
## $univariateNormality
##                              Test       Variable Statistic   p value Normality
## 1 Lilliefors (Kolmogorov-Smirnov)   cilindros       0.3238  <0.001      NO
## 2 Lilliefors (Kolmogorov-Smirnov) desplazamiento    0.1815  <0.001      NO
## 3 Lilliefors (Kolmogorov-Smirnov)       CV          0.1636  <0.001      NO
## 4 Lilliefors (Kolmogorov-Smirnov)      peso         0.0946  <0.001      NO
## 5 Lilliefors (Kolmogorov-Smirnov)  aceleracion      0.0513  0.0149      NO
## 6 Lilliefors (Kolmogorov-Smirnov)     precio        0.1164  <0.001      NO
##
## $Descriptives
##                   n         Mean     Std.Dev  Median      Min      Max      25th
## cilindros       392     5.471939    1.705783     4.0    3.000     8.00     4.000
## desplazamiento 392   194.411990  104.644004   151.0   68.000   455.00   105.000
## CV              392   104.469388   38.491160    93.5   46.000   230.00    75.000
## peso            392  2977.584184  849.402560  2803.5 1613.000  5140.00  2225.250
## aceleracion     392    15.541327    2.758864    15.5    8.000    24.80    13.775
## precio          392 29659.659644 9887.842779 30000.0 1598.073 53745.94 22956.440
##                   75th       Skew       Kurtosis
## cilindros          8.000   0.50422726   -1.4038700
```

```
## desplazamiento   275.750   0.69630832   -0.7949853
## CV               126.000   1.07901906    0.6541069
## peso            3614.750   0.51561602   -0.8253788
## aceleracion       17.025   0.28935919    0.4058767
## precio         36383.155   0.02397518   -0.2982370
```

```
mvn(vehiculosnumerico,mvnTest = "royston",univariateTest = "Lillie")# test de royston
```

```
## $multivariateNormality
##      Test        H      p value MVN
## 1 Royston 152.8706 8.577626e-33  NO
##
## $univariateNormality
##                              Test       Variable Statistic   p value Normality
## 1 Lilliefors (Kolmogorov-Smirnov)   cilindros       0.3238  <0.001      NO
## 2 Lilliefors (Kolmogorov-Smirnov) desplazamiento    0.1815  <0.001      NO
## 3 Lilliefors (Kolmogorov-Smirnov)       CV          0.1636  <0.001      NO
## 4 Lilliefors (Kolmogorov-Smirnov)      peso         0.0946  <0.001      NO
## 5 Lilliefors (Kolmogorov-Smirnov)  aceleracion      0.0513  0.0149      NO
## 6 Lilliefors (Kolmogorov-Smirnov)     precio        0.1164  <0.001      NO
##
## $Descriptives
##                   n         Mean     Std.Dev  Median      Min      Max      25th
## cilindros       392     5.471939    1.705783     4.0    3.000     8.00     4.000
## desplazamiento 392   194.411990  104.644004   151.0   68.000   455.00   105.000
## CV              392   104.469388   38.491160    93.5   46.000   230.00    75.000
## peso            392  2977.584184  849.402560  2803.5 1613.000  5140.00  2225.250
## aceleracion     392    15.541327    2.758864    15.5    8.000    24.80    13.775
## precio          392 29659.659644 9887.842779 30000.0 1598.073 53745.94 22956.440
##                      75th       Skew    Kurtosis
## cilindros           8.000 0.50422726  -1.4038700
## desplazamiento    275.750 0.69630832  -0.7949853
## CV                126.000 1.07901906   0.6541069
## peso             3614.750 0.51561602  -0.8253788
## aceleracion        17.025 0.28935919   0.4058767
## precio          36383.155 0.02397518  -0.2982370
```

```
mvn(vehiculosnumerico[,c("desplazamiento","CV")],mvnTest = "royston",
univariateTest = "Lillie",univariatePlot="qq",multivariatePlot = "persp")# test de royston
```

```
## $multivariateNormality
##      Test        H      p value MVN
## 1 Royston 95.56293 5.087419e-22  NO
##
## $univariateNormality
##                              Test       Variable Statistic   p value Normality
## 1 Lilliefors (Kolmogorov-Smirnov) desplazamiento    0.1815  <0.001      NO
## 2 Lilliefors (Kolmogorov-Smirnov)       CV          0.1636  <0.001      NO
##
## $Descriptives
##                   n   Mean Std.Dev Median Min Max 25th  75th    Skew  Kurtosis
## desplazamiento 392 194.41 104.644  151.0  68 455 105 275.75 0.69630 -0.794985
## CV             392 104.46  38.491   93.5  46 230  75 126.00 1.07901  0.654106
```

Algunas de las soluciones para corregir el defecto de la normalidad es tomar transformaciones de los datos por ejemplo logaritmos, raíz cuadrada...

```
mvn(vehiculosnumerico,univariateTest = "Lillie",transform = "sqrt")
```

```
## $multivariateNormality
##            Test       HZ p value MVN
## 1 Henze-Zirkler 2.620966       0  NO
##
## $univariateNormality
##                              Test       Variable Statistic   p value Normality
## 1 Lilliefors (Kolmogorov-Smirnov)   cilindros       0.3269  <0.001      NO
## 2 Lilliefors (Kolmogorov-Smirnov) desplazamiento    0.1594  <0.001      NO
## 3 Lilliefors (Kolmogorov-Smirnov)       CV          0.1316  <0.001      NO
## 4 Lilliefors (Kolmogorov-Smirnov)      peso         0.0868  <0.001      NO
## 5 Lilliefors (Kolmogorov-Smirnov)  aceleracion      0.0410  0.1134      YES
## 6 Lilliefors (Kolmogorov-Smirnov)     precio        0.1324  <0.001      NO
##
## $Descriptives
##                   n       Mean     Std.Dev     Median        Min        Max
```

```
## cilindros      392   2.311918  0.3567862   2.000000  1.732051   2.828427
## desplazamiento 392  13.459603  3.6448560  12.288206  8.246211  21.330729
## CV             392  10.063940  1.7873567   9.669505  6.782330  15.165751
## peso           392  54.027807  7.6635566  52.948077 40.162171  71.693793
## aceleracion    392   3.926648  0.3508259   3.937004  2.828427   4.979960
## precio         392 169.517613 30.4269664 173.205081 39.975910 231.831702
##                        25th       75th        Skew     Kurtosis
## cilindros       2.000000   2.828427  0.42623636  -1.47349434
## desplazamiento 10.246951  16.599638  0.45004313  -1.17288794
## CV              8.660254  11.224709  0.72712152  -0.07432487
## peso           47.172554  60.122786  0.33141622  -1.01244499
## aceleracion     3.711464   4.126133 -0.02204162   0.33412635
## precio        151.513797 190.743659 -0.57117227   0.66121817
```

Por todos los test comprobamos que no se da la normalidad multivariante.

3.1.3. Homocedasticidad

La homocedasticidad consiste en comprobar si la dispersión (varianza) de una variable es igual para los grupos que marque una variable de factor. Para ello utilizaremos el test de Levene que contrasta la igualdad de varianzas para los k grupos de una variable:

$$H_0 : \sigma_1^2 = \sigma_2^2 = \ldots = \sigma_k^2$$

$$H_1 : \sigma_i^2 \neq \sigma_j^2$$

En R tendremos:

```
library(car)
lapply(vehiculosnumerico,leveneTest,coches2$origen)
```

```
## $cilindros
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value    Pr(>F)
## group   2  110.44 < 2.2e-16 ***
##       395
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## $desplazamiento
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value    Pr(>F)
## group   2  109.47 < 2.2e-16 ***
##       395
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## $CV
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value    Pr(>F)
## group   2  26.527 1.586e-11 ***
##       389
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## $peso
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value    Pr(>F)
## group   2  43.602 < 2.2e-16 ***
##       395
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## $aceleracion
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value  Pr(>F)
## group   2  4.2256 0.01528 *
##       395
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## $precio
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value Pr(>F)
## group   2  0.4084  0.665
##       395
```

En los casos en los que el p-valor sea menor que 0.05 se rechazará la hipótesis de igualdad de varianzas para esos grupos.

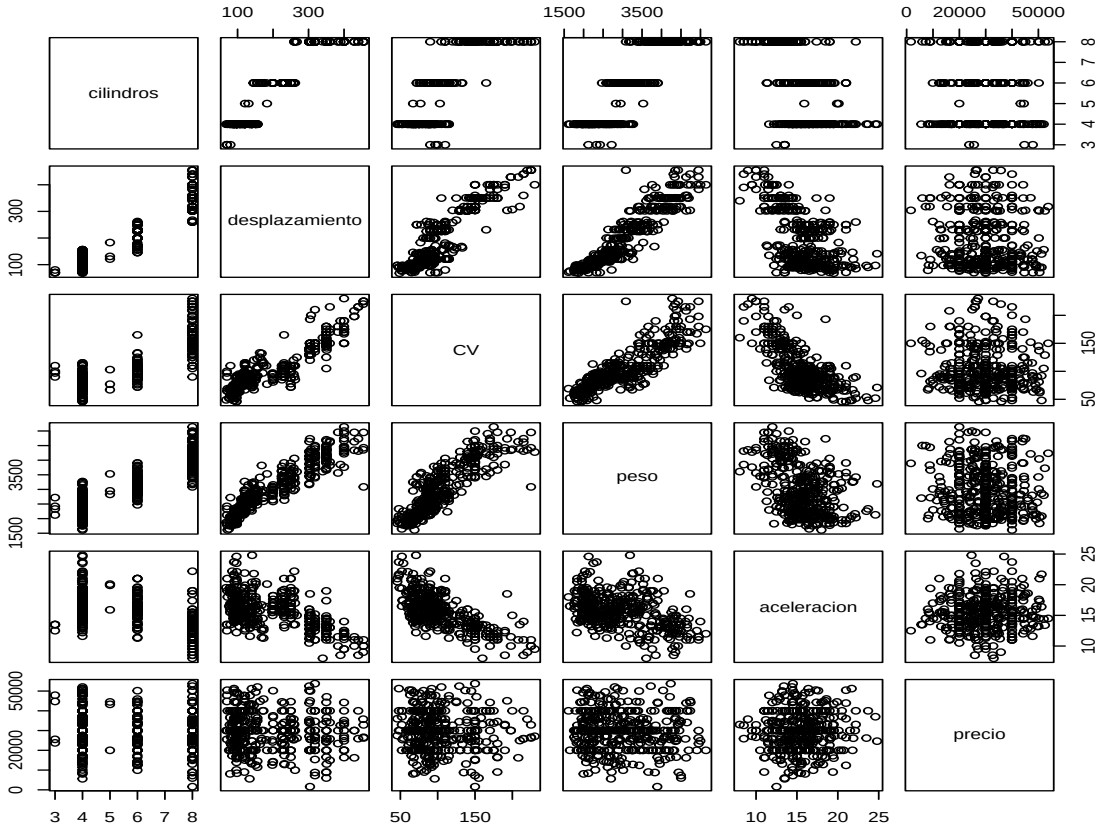
Otra opción es querer realizar el test de igualdad de matrices de varianzas covarianzas, el test más usual es el test de la M de box:

```
library(biotools)


boxM(coches3[,c("cilindros","desplazamiento","CV","peso","aceleracion"
,"precio")],coches3$origen)


##
##  Box's M-test for Homogeneity of Covariance Matrices
##
## data:coches3[,c("cilindros", "desplazamiento", "CV", "peso", "aceleracion","precio")]
## Chi-Sq (approx.) = 523.73, df = 42, p-value < 2.2e-16
```

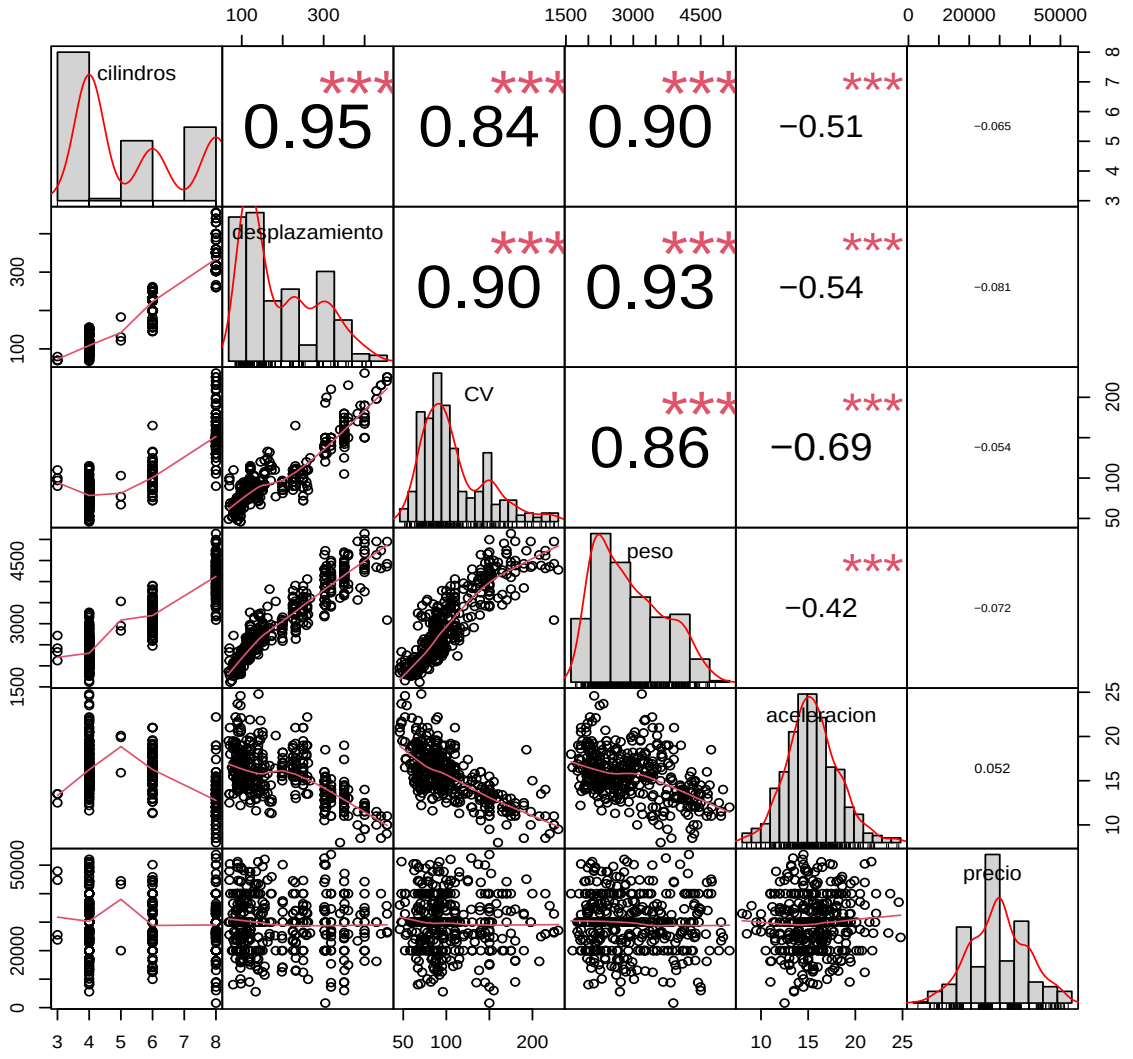
3.1.4. Linealidad

La ultima de las hipótesis que contrastaremos es la de linealidad, para ellos hacemos las siguientes gráficas:

```
library("PerformanceAnalytics")


plot(vehiculosnumerico)
```




```
chart.Correlation(vehiculosnumerico, histogram=TRUE)
```


cilindros
desplazamiento
CV
peso
aceleracion
precio
0.95 ***
0.84 ***
0.90 ***
−0.51 ***
−0.065
0.90 ***
0.93 ***
−0.54 ***
−0.081
0.86 ***
−0.69 ***
−0.054
−0.42 ***
−0.072
0.052