Procesado de datos II e introducción a R

Clases de objetos

Vectores

```
(x <- c(1,4,2,42,4))
```

```
[1]  1  4  2 42  4
```

```
class(x)
```

```
[1] "numeric"
```

```
(y <- c(1,3,"ch",2,2))
```

```
[1] "1"  "3"  "ch" "2"  "2"
```

```
class(y)
```

```
[1] "character"
```

```
y[3]
```

```
[1] "ch"
```

```
  y[-3]
```

```
[1] "1" "3" "2" "2"
```

Factores

```
  (mifactor <- rep(c("A","B"),each=4))
```

```
[1] "A" "A" "A" "A" "B" "B" "B" "B"
```

```
  mifactor
```

```
[1] "A" "A" "A" "A" "B" "B" "B" "B"
```

```
  class(mifactor)
```

```
[1] "character"
```

```
  (mifactor <- as.numeric(mifactor))
```

```
Warning: NAs introduced by coercion
```

```
[1] NA NA NA NA NA NA NA NA
```

```
  (mifactor2 <- as.factor(rep(c("A","B","C"),each=5)))
```

```
 [1] A A A A A B B B B B C C C C C
Levels: A B C
```

```r
mifactor2  <- mifactor2[mifactor2!="C"]

(mifactor2 <- as.factor(as.character(mifactor2)))
```

```
 [1] A A A A A B B B B B
Levels: A B
```

Matrices

```r
(X <- rnorm(25,mean = 0,sd=1))
```

```
 [1] -0.52586631  1.55544920 -1.16994767  0.06233879  0.66985466  0.57196950
 [7] -0.78859688  1.64829941 -0.08716969 -0.11219052  0.55035999 -0.37607243
[13]  0.41567147  0.86095593  0.94024890  0.23333106 -1.46772028 -0.29323181
[19] -0.46053262 -0.84669300  0.65604840 -0.04239457 -0.93421058 -1.16304580
[25] -0.26329990
```

```r
(X <- matrix(X,byrow = T,ncol=5))
```

```
           [,1]        [,2]       [,3]        [,4]       [,5]
[1,] -0.5258663  1.55544920 -1.1699477  0.06233879  0.6698547
[2,]  0.5719695 -0.78859688  1.6482994 -0.08716969 -0.1121905
[3,]  0.5503600 -0.37607243  0.4156715  0.86095593  0.9402489
[4,]  0.2333311 -1.46772028 -0.2932318 -0.46053262 -0.8466930
[5,]  0.6560484 -0.04239457 -0.9342106 -1.16304580 -0.2632999
```

```r
(X.t <- t(X))
```

```
            [,1]        [,2]       [,3]       [,4]        [,5]
[1,] -0.52586631  0.57196950  0.5503600  0.2333311  0.65604840
[2,]  1.55544920 -0.78859688 -0.3760724 -1.4677203 -0.04239457
[3,] -1.16994767  1.64829941  0.4156715 -0.2932318 -0.93421058
[4,]  0.06233879 -0.08716969  0.8609559 -0.4605326 -1.16304580
[5,]  0.66985466 -0.11219052  0.9402489 -0.8466930 -0.26329990
```

```r
(Y <- matrix(rnorm(20,mean=0,sd=4),ncol = 2))
```

```
            [,1]       [,2]
 [1,] -1.9978863 -3.8399623
 [2,] -2.7708734 -3.3134342
 [3,] -0.5256227 -3.0051835
 [4,] -5.8085579  5.3028543
 [5,]  1.3242234  4.0915169
 [6,] -1.6228782  2.2971294
 [7,] -5.4964594  3.9847029
 [8,] -6.3507514 -0.4054161
 [9,]  4.6048871 -1.8494299
[10,] -6.0634912  2.1334161
```

```r
dim(Y)
```

```
[1] 10  2
```

```r
Y.t <- t(Y)
dim(Y.t)
```

```
[1]  2 10
```

```r
dim(Y)
```

```
[1] 10  2
```

```r
(multi <- Y.t %*% Y)
```

```
          [,1]      [,2]
[1,] 178.58624 -51.45873
[2,] -51.45873 108.90694
```

```r
colnames(multi) <- c("A","B")
rownames(multi) <- c("pepe","juan")
multi
```

```
             A         B
pepe 178.58624 -51.45873
juan -51.45873 108.90694
```

```r
multi[,2]
```

```
     pepe      juan
-51.45873 108.90694
```

```r
multi[1,]
```

```
        A         B
178.58624 -51.45873
```

```r
multi[1,2]
```

```
[1] -51.45873
```

```r
multi[-2,]
```

```
        A         B
178.58624 -51.45873
```

```r
multi[,-2]
```

```
     pepe      juan
178.58624 -51.45873
```

Data frames

```r
(midf2 <- as.data.frame(multi))
```

```
             A         B
pepe 178.58624 -51.45873
juan -51.45873 108.90694
```

```r
(midf3 <- data.frame(A=c("conejo","oso","pepe"),
                     B=as.factor(c("A","A","B")),
                     C=c(1,52,2)))
```

```
       A B  C
1 conejo A  1
2    oso A 52
3   pepe B  2
```

```r
midf3$A
```

```
[1] "conejo" "oso"    "pepe"
```

```r
colnames(midf3) <- c("papa","chaucha","monaguillo")
midf3
```

```
    papa chaucha monaguillo
1 conejo       A          1
2    oso       A         52
3   pepe       B          2
```

Listas

```r
(milista <- as.list(midf3))
```

```
$papa
[1] "conejo" "oso"    "pepe"

$chaucha
[1] A A B
Levels: A B

$monaguillo
[1]  1 52  2
```

```
milista$papa
```

```
[1] "conejo" "oso"    "pepe"
```

```
lista <- list(prueba1=c(2,4,2,1),
              prueba2 = c("ch","cjo"),
              prueba5 = as.factor(c("A","A","B","B")))

lista$preueba1
```

```
NULL
```

```
lista[[1]]
```

```
[1] 2 4 2 1
```

Lectura de datos y documetnos dinámicos en Quarto/Rmarkdown

Archivos de texto plano

.csv

```
list.files("./data/")
```

```
[1] "BreastTissue.csv"
[2] "BreastTissue.xls"
[3] "leukemia_remission.txt"
[4] "ObesityDataSet_raw_and_data_sinthetic.arff"
```

```r
list.files("./data",pattern = ".csv")
```

```
[1] "BreastTissue.csv"
```

```r
tejido_cancer.archivo <- list.files("./data/",pattern = ".csv",full.names = T)
tejido_cancer.datos <- read.csv(tejido_cancer.archivo)
head(tejido_cancer.datos,3)[,1:4] ## no empieza en 0 !!
```

```
  Class       I0     PA500        HFS
1   car 524.7941 0.1874484 0.03211406
2   car 330.0000 0.2268928 0.26529005
3   car 551.8793 0.2324779 0.06352998
```

```r
tejido.txt <- list.files("./data/",pattern = ".txt",full.names = T)
tejido.data <- read.table(tejido.txt,skip=1)
head(tejido.data)
```

```
  remiss cell smear infil  li blast  temp
1      1  0.8  0.83  0.66 1.9 1.100 0.996
2      1  0.9  0.36  0.32 1.4 0.740 0.992
3      0  0.8  0.88  0.70 0.8 0.176 0.982
4      0  1.0  0.87  0.87 0.7 1.053 0.986
5      1  0.9  0.75  0.68 1.3 0.519 0.980
6      0  1.0  0.65  0.65 0.6 0.519 0.982
```

```r
library(xlsx)
tejido <- read.xlsx(file = "./data/BreastTissue.xls",sheetIndex = 2)
```

```r
datos.param <- read.csv(list.files(params$data,pattern = ".csv",full.names = T))
```

mis datos tienen 106, 10

Ahora vamos leer archivos arrf, es lo bueno que tiene R que no se necesitan programas especiales para leer archivos complejos

Primero de todo necesitamos cargar las librerías necesarias

```
library(dplyr) # Facil manipulacion de data frames
```

```
Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
library(ggplot2)# Graficos
library(knitr)
library(ggpubr)
library(car)
```

```
Loading required package: carData


Attaching package: 'car'

The following object is masked from 'package:dplyr':

    recode
```

1. Antes, de leer los datos, necesitamos saber que extensión son para proceder con la lectura, es decir, si son .csv, .txt, u otro formato.

```r
archivo <- list.files(params$data,
                      pattern = "*.arff",
                      full.names = T,recursive = T)
# file.show(archivo)
```

Podemos osbervar, como en realidad, es un archivo de texto, denominado arff. No obstante, tenemos que convertir dicho archivo a un data frame para poder manejarlo en R.

Leemos el archivo por líneas. E imprimimos por pantalla las primeras líneas:

```r
predata <- readLines(archivo)
print(head(predata))
```

```
[1] "@relation obeyesdad-weka.filters.supervised.instance.SMOTE-C0-K5-P300.0-S1-weka.filters
[2] ""
[3] "@attribute Gender {Female,Male}"
[4] "@attribute Age numeric"
[5] "@attribute Height numeric"
[6] "@attribute Weight numeric"
```

Ahora obtenemos solamente la cabecera, la cual está compuesta del símbolo arroba

```r
filas_cabecera <- grep("@",predata)
cabecera <- predata[filas_cabecera]
print(cabecera)
```

```
 [1] "@relation obeyesdad-weka.filters.supervised.instance.SMOTE-C0-K5-P300.0-S1-weka.filters
 [2] "@attribute Gender {Female,Male}"
 [3] "@attribute Age numeric"
 [4] "@attribute Height numeric"
 [5] "@attribute Weight numeric"
 [6] "@attribute family_history_with_overweight {yes,no}"
 [7] "@attribute FAVC {yes,no}"
 [8] "@attribute FCVC numeric"
 [9] "@attribute NCP numeric"
[10] "@attribute CAEC {no,Sometimes,Frequently,Always}"
[11] "@attribute SMOKE {yes,no}"
[12] "@attribute CH2O numeric"
[13] "@attribute SCC {yes,no}"
[14] "@attribute FAF numeric"
```

```
[15] "@attribute TUE numeric"
[16] "@attribute CALC {no,Sometimes,Frequently,Always}"
[17] "@attribute MTRANS {Automobile,Motorbike,Bike,Public_Transportation,Walking}"
[18] "@attribute NObeyesdad {Insufficient_Weight,Normal_Weight,Overweight_Level_I,Overweight_
[19] "@data"
```

Si hacemos un indexado negativo de la cabecera, tenemos los datos crudos

```
predatos <- predata[-filas_cabecera]
head(predatos)
```

```
[1] ""
[2] ""
[3] "Female,21,1.62,64,yes,no,2,3,Sometimes,no,2,no,0,1,no,Public_Transportation,Normal_Weigh
[4] "Female,21,1.52,56,yes,no,3,3,Sometimes,yes,3,yes,3,0,Sometimes,Public_Transportation,Nor
[5] "Male,23,1.8,77,yes,no,2,3,Sometimes,no,2,no,2,1,Frequently,Public_Transportation,Normal_
[6] "Male,27,1.8,87,no,no,3,3,Sometimes,no,2,no,2,0,Frequently,Walking,Overweight_Level_I"
```

```
## convertimos a matriz para extraer el nombre
```

Ahora extraemos del archivo de texto plano, aquellas filas que empiecen con `"@attribute"`. Esto nos dice el nombre y el tipo de datos con los que tenemos que trabajar, al igual que la mayoría del significado de las columnas.

```
filas_cabecera <- grep("@attribute",predata)

pre_columnas <- predata[filas_cabecera]
print(pre_columnas)
```

```
 [1] "@attribute Gender {Female,Male}"
 [2] "@attribute Age numeric"
 [3] "@attribute Height numeric"
 [4] "@attribute Weight numeric"
 [5] "@attribute family_history_with_overweight {yes,no}"
 [6] "@attribute FAVC {yes,no}"
 [7] "@attribute FCVC numeric"
 [8] "@attribute NCP numeric"
 [9] "@attribute CAEC {no,Sometimes,Frequently,Always}"
[10] "@attribute SMOKE {yes,no}"
```

```
[11] "@attribute CH2O numeric"
[12] "@attribute SCC {yes,no}"
[13] "@attribute FAF numeric"
[14] "@attribute TUE numeric"
[15] "@attribute CALC {no,Sometimes,Frequently,Always}"
[16] "@attribute MTRANS {Automobile,Motorbike,Bike,Public_Transportation,Walking}"
[17] "@attribute NObeyesdad {Insufficient_Weight,Normal_Weight,Overweight_Level_I,Overweight
```

Tenemos 17 columnas…Observamos, como los datos, estan separados por un espacio, vamos a transformar la salida anterior en una matriz de caracteres. Para ello utilizamos la función `strsplit`. Esta función nos devuelve una lista de las separaciones.

```
pre_columnas.list <- strsplit(predata[filas_cabecera]," ")
print(length(pre_columnas.list))
```

```
[1] 17
```

Tenemos efectivamente. Ahora necesitamos manipular la lista para convertira el una matriz de 17X3. No obstante antes, de manipular debemos de pasar la lista a un string.

```
pre_columnas.unlist <- unlist(pre_columnas.list)
#convertimos a matriz
cabecera.raw <- matrix(pre_columnas.unlist,nrow=length(pre_columnas.list),
                   ncol=3,byrow = T)
head(cabecera.raw)
```

```
     [,1]         [,2]                              [,3]
[1,] "@attribute" "Gender"                          "{Female,Male}"
[2,] "@attribute" "Age"                             "numeric"
[3,] "@attribute" "Height"                          "numeric"
[4,] "@attribute" "Weight"                          "numeric"
[5,] "@attribute" "family_history_with_overweight" "{yes,no}"
[6,] "@attribute" "FAVC"                            "{yes,no}"
```

De la cabecera nos importan la segunda y la tercera columna que son las que tienen información

```
cabecera  <- cabecera.raw[,2:3]
## tambien la podemos convertir a data frame.
cabecera <- as.data.frame(cabecera)
```

```r
colnames(cabecera) <- c("Variable","Clase")
cabecera
```

```
                        Variable
1                         Gender
2                            Age
3                         Height
4                         Weight
5  family_history_with_overweight
6                           FAVC
7                           FCVC
8                            NCP
9                           CAEC
10                         SMOKE
11                          CH2O
12                           SCC
13                           FAF
14                           TUE
15                          CALC
16                        MTRANS
17                    NObeyesdad

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16                                                                {Automobile,Motorbike,Bi
17 {Insufficient_Weight,Normal_Weight,Overweight_Level_I,Overweight_Level_II,Obesity_Type_I,
```

Ya tenemos la cabecera, ahora vamos por los datos. Si recordamos lo habíamos guardado en la variable `predatos`. Tambiién habíamos observado que estaban separados por comas. Por lo

tanto procedemos a separarlos por dicho caracter, y a parte, sabemos que los datos se componen 17 columnas. Especificamos que se ordenen por filas, mediante el comando `byrow=T`.

```
datos <-
  as.data.frame(matrix(
    unlist(strsplit(predatos, ",")),
    ncol = nrow(cabecera),
    byrow = T
  ))
colnames(datos) <- cabecera$Variable
head(datos)
```

```
  Gender Age Height Weight family_history_with_overweight FAVC FCVC NCP
1 Female  21   1.62     64                             yes   no    2   3
2 Female  21   1.52     56                             yes   no    3   3
3   Male  23    1.8     77                             yes   no    2   3
4   Male  27    1.8     87                              no   no    3   3
5   Male  22   1.78   89.8                              no   no    2   1
6   Male  29   1.62     53                              no  yes    2   3
       CAEC SMOKE CH2O SCC FAF TUE       CALC                 MTRANS
1 Sometimes    no    2  no   0   1         no Public_Transportation
2 Sometimes   yes    3 yes   3   0  Sometimes Public_Transportation
3 Sometimes    no    2  no   2   1 Frequently Public_Transportation
4 Sometimes    no    2  no   2   0 Frequently               Walking
5 Sometimes    no    2  no   0   0  Sometimes Public_Transportation
6 Sometimes    no    2  no   0   0  Sometimes            Automobile
          NObeyesdad
1       Normal_Weight
2       Normal_Weight
3       Normal_Weight
4  Overweight_Level_I
5 Overweight_Level_II
6       Normal_Weight
```

Preprocesado de datos

En este paso, necesitamos identificar qué variables son numéricas y cuales son factores.

```
str(datos)
```

```
'data.frame':   2111 obs. of  17 variables:
 $ Gender                         : chr  "Female" "Female" "Male" "Male" ...
 $ Age                            : chr  "21" "21" "23" "27" ...
 $ Height                         : chr  "1.62" "1.52" "1.8" "1.8" ...
 $ Weight                         : chr  "64" "56" "77" "87" ...
 $ family_history_with_overweight: chr  "yes" "yes" "yes" "no" ...
 $ FAVC                           : chr  "no" "no" "no" "no" ...
 $ FCVC                           : chr  "2" "3" "2" "3" ...
 $ NCP                            : chr  "3" "3" "3" "3" ...
 $ CAEC                           : chr  "Sometimes" "Sometimes" "Sometimes" "Sometimes" ...
 $ SMOKE                          : chr  "no" "yes" "no" "no" ...
 $ CH2O                           : chr  "2" "3" "2" "2" ...
 $ SCC                            : chr  "no" "yes" "no" "no" ...
 $ FAF                            : chr  "0" "3" "2" "2" ...
 $ TUE                            : chr  "1" "0" "1" "0" ...
 $ CALC                           : chr  "no" "Sometimes" "Frequently" "Frequently" ...
 $ MTRANS                         : chr  "Public_Transportation" "Public_Transportation" "Publ
 $ NObeyesdad                     : chr  "Normal_Weight" "Normal_Weight" "Normal_Weight" "Over
```

Todas están catalogadas como caracter. Bien podemos ir variable por variable y asignar la clase a la que corresponde, o podemos realizar lo siguiente.

```r
vars.numericas <- grep("numeric",cabecera$Clase)
datos[,vars.numericas]<- apply(datos[,vars.numericas]
                               , 2,
                               as.numeric)
datos[,-vars.numericas] <- lapply(datos[,-vars.numericas],
                                  as.factor)
str(datos)
```

```
'data.frame':   2111 obs. of  17 variables:
 $ Gender                         : Factor w/ 2 levels "Female","Male": 1 1 2 2 2 2 1 2 2 2 .
 $ Age                            : num  21 21 23 27 22 29 23 22 24 22 ...
 $ Height                         : num  1.62 1.52 1.8 1.8 1.78 1.62 1.5 1.64 1.78 1.72 ...
 $ Weight                         : num  64 56 77 87 89.8 53 55 53 64 68 ...
 $ family_history_with_overweight: Factor w/ 2 levels "no","yes": 2 2 2 1 1 1 2 1 2 2 ...
 $ FAVC                           : Factor w/ 2 levels "no","yes": 1 1 1 1 1 2 2 1 2 2 ...
 $ FCVC                           : num  2 3 2 3 2 2 3 2 3 2 ...
 $ NCP                            : num  3 3 3 3 1 3 3 3 3 3 ...
 $ CAEC                           : Factor w/ 4 levels "Always","Frequently",..: 4 4 4 4 4 4 4
 $ SMOKE                          : Factor w/ 2 levels "no","yes": 1 2 1 1 1 1 1 1 1 1 ...
```

```
$ CH2O                          : num  2 3 2 2 2 2 2 2 2 2 ...
$ SCC                           : Factor w/ 2 levels "no","yes": 1 2 1 1 1 1 1 1 1 1 ...
$ FAF                           : num  0 3 2 2 0 0 1 3 1 1 ...
$ TUE                           : num  1 0 1 0 0 0 0 0 1 1 ...
$ CALC                          : Factor w/ 4 levels "Always","Frequently",..: 3 4 2 2 4 4 4
$ MTRANS                        : Factor w/ 5 levels "Automobile","Bike",..: 4 4 4 5 4 1 3 4
$ NObeyesdad                    : Factor w/ 7 levels "Insufficient_Weight",..: 2 2 2 6 7 2 2
```

Ahora bien, también podemos realizar una función con los pasos anteriores.

La siguiente función hace lo mismo que el código anterior, asignando a las variables la clase que corresponde.

```
read.arff <- function(file_name){
  archivo <- readLines(file_name)

  filas_cabecera <- grep("@attribute", predata)

  pre_columnas <- predata[filas_cabecera]
  pre_columnas.list <- strsplit(predata[filas_cabecera], " ")
  cabecera  <- cabecera.raw[, 2:3]
  cabecera <- as.data.frame(cabecera)
  colnames(cabecera) <- c("Variable", "Clase")
  datos <-
    as.data.frame(matrix(
      unlist(strsplit(predatos, ",")),
      ncol = nrow(cabecera),
      byrow = T
    ))
  colnames(datos) <- cabecera$Variable
  datos <- as.data.frame(datos)
  numericas <- grep("numeric",cabecera$Clase)
  datos[,numericas] <- lapply(datos[,numericas],as.numeric)
  datos[,-numericas] <- lapply(datos[,-numericas],as.factor)


  return(datos)

}
```

```
datos  <- read.arff(archivo)
str(datos)
```

```
'data.frame':   2111 obs. of  17 variables:
 $ Gender                         : Factor w/ 2 levels "Female","Male": 1 1 2 2 2 2 1 2 2 2 .
 $ Age                            : num  21 21 23 27 22 29 23 22 24 22 ...
 $ Height                         : num  1.62 1.52 1.8 1.8 1.78 1.62 1.5 1.64 1.78 1.72 ...
 $ Weight                         : num  64 56 77 87 89.8 53 55 53 64 68 ...
 $ family_history_with_overweight: Factor w/ 2 levels "no","yes": 2 2 2 1 1 1 2 1 2 2 ...
 $ FAVC                           : Factor w/ 2 levels "no","yes": 1 1 1 1 1 2 2 1 2 2 ...
 $ FCVC                           : num  2 3 2 3 2 2 3 2 3 2 ...
 $ NCP                            : num  3 3 3 3 1 3 3 3 3 3 ...
 $ CAEC                           : Factor w/ 4 levels "Always","Frequently",..: 4 4 4 4 4 4 4
 $ SMOKE                          : Factor w/ 2 levels "no","yes": 1 2 1 1 1 1 1 1 1 1 ...
 $ CH2O                           : num  2 3 2 2 2 2 2 2 2 2 ...
 $ SCC                            : Factor w/ 2 levels "no","yes": 1 2 1 1 1 1 1 1 1 1 ...
 $ FAF                            : num  0 3 2 2 0 0 1 3 1 1 ...
 $ TUE                            : num  1 0 1 0 0 0 0 0 1 1 ...
 $ CALC                           : Factor w/ 4 levels "Always","Frequently",..: 3 4 2 2 4 4 4
 $ MTRANS                         : Factor w/ 5 levels "Automobile","Bike",..: 4 4 4 5 4 1 3 4
 $ NObeyesdad                     : Factor w/ 7 levels "Insufficient_Weight",..: 2 2 2 6 7 2 2
```

Primero de todo necesitamos cargar las librerías necesarias

```r
library(dplyr) # Facil manipulacion de data frames
library(ggplot2)# Graficos
library(knitr)
library(ggpubr)
library(car)
```

1. Antes, de leer los datos, necesitamos saber que extensión son para proceder con la lectura, es decir, si son .csv, .txt, u otro formato.

```r
archivo <- list.files(params$data,
                      pattern = "*.arff",
                      full.names = T,recursive = T)
# file.show(archivo)
```

Podemos osbervar, como en realidad, es un archivo de texto, denominado arff. No obstante, tenemos que convertir dicho archivo a un data frame para poder manejarlo en R.

Leemos el archivo por líneas. E imprimimos por pantalla las primeras líneas:

```r
predata <- readLines(archivo)
print(head(predata))
```

```
[1] "@relation obeyesdad-weka.filters.supervised.instance.SMOTE-C0-K5-P300.0-S1-weka.filters
[2] ""
[3] "@attribute Gender {Female,Male}"
[4] "@attribute Age numeric"
[5] "@attribute Height numeric"
[6] "@attribute Weight numeric"
```

Ahora obtenemos solamente la cabecera, la cual está compuesta del símbolo arroba

```r
filas_cabecera <- grep("@",predata)
cabecera <- predata[filas_cabecera]
print(cabecera)
```

```
 [1] "@relation obeyesdad-weka.filters.supervised.instance.SMOTE-C0-K5-P300.0-S1-weka.filters
 [2] "@attribute Gender {Female,Male}"
 [3] "@attribute Age numeric"
 [4] "@attribute Height numeric"
 [5] "@attribute Weight numeric"
 [6] "@attribute family_history_with_overweight {yes,no}"
 [7] "@attribute FAVC {yes,no}"
 [8] "@attribute FCVC numeric"
 [9] "@attribute NCP numeric"
[10] "@attribute CAEC {no,Sometimes,Frequently,Always}"
[11] "@attribute SMOKE {yes,no}"
[12] "@attribute CH2O numeric"
[13] "@attribute SCC {yes,no}"
[14] "@attribute FAF numeric"
[15] "@attribute TUE numeric"
[16] "@attribute CALC {no,Sometimes,Frequently,Always}"
[17] "@attribute MTRANS {Automobile,Motorbike,Bike,Public_Transportation,Walking}"
[18] "@attribute NObeyesdad {Insufficient_Weight,Normal_Weight,Overweight_Level_I,Overweight_
[19] "@data"
```

Si hacemos un indexado negativo de la cabecera, tenemos los datos crudos

```r
predatos <- predata[-filas_cabecera]
head(predatos)
```

```
[1] ""
[2] ""
[3] "Female,21,1.62,64,yes,no,2,3,Sometimes,no,2,no,0,1,no,Public_Transportation,Normal_Weigh
[4] "Female,21,1.52,56,yes,no,3,3,Sometimes,yes,3,yes,3,0,Sometimes,Public_Transportation,Nor
[5] "Male,23,1.8,77,yes,no,2,3,Sometimes,no,2,no,2,1,Frequently,Public_Transportation,Normal_
[6] "Male,27,1.8,87,no,no,3,3,Sometimes,no,2,no,2,0,Frequently,Walking,Overweight_Level_I"
```

```r
## convertimos a matriz para extraer el nombre
```

Ahora extraemos del archivo de texto plano, aquellas filas que empiecen con `"@attribute"`. Esto nos dice el nombre y el tipo de datos con los que tenemos que trabajar, al igual que la mayoría del significado de las columnas.

```r
filas_cabecera <- grep("@attribute",predata)

pre_columnas <- predata[filas_cabecera]
print(pre_columnas)
```

```
 [1] "@attribute Gender {Female,Male}"
 [2] "@attribute Age numeric"
 [3] "@attribute Height numeric"
 [4] "@attribute Weight numeric"
 [5] "@attribute family_history_with_overweight {yes,no}"
 [6] "@attribute FAVC {yes,no}"
 [7] "@attribute FCVC numeric"
 [8] "@attribute NCP numeric"
 [9] "@attribute CAEC {no,Sometimes,Frequently,Always}"
[10] "@attribute SMOKE {yes,no}"
[11] "@attribute CH2O numeric"
[12] "@attribute SCC {yes,no}"
[13] "@attribute FAF numeric"
[14] "@attribute TUE numeric"
[15] "@attribute CALC {no,Sometimes,Frequently,Always}"
[16] "@attribute MTRANS {Automobile,Motorbike,Bike,Public_Transportation,Walking}"
[17] "@attribute NObeyesdad {Insufficient_Weight,Normal_Weight,Overweight_Level_I,Overweight_
```

Tenemos 17 columnas...Observamos, como los datos, estan separados por un espacio, vamos a transformar la salida anterior en una matriz de caracteres. Para ello utilizamos la función `strsplit`. Esta función nos devuelve una lista de las separaciones.

```
pre_columnas.list <- strsplit(predata[filas_cabecera]," ")
print(length(pre_columnas.list))
```

```
[1] 17
```

Tenemos efectivamente. Ahora necesitamos manipular la lista para convertira el una matriz de 17X3. No obstante antes, de manipular debemos de pasar la lista a un string.

```
pre_columnas.unlist <- unlist(pre_columnas.list)
#convertimos a matriz
cabecera.raw <- matrix(pre_columnas.unlist,nrow=length(pre_columnas.list),
                   ncol=3,byrow = T)
head(cabecera.raw)
```

```
     [,1]         [,2]                              [,3]
[1,] "@attribute" "Gender"                          "{Female,Male}"
[2,] "@attribute" "Age"                             "numeric"
[3,] "@attribute" "Height"                          "numeric"
[4,] "@attribute" "Weight"                          "numeric"
[5,] "@attribute" "family_history_with_overweight" "{yes,no}"
[6,] "@attribute" "FAVC"                            "{yes,no}"
```

De la cabecera nos importan la segunda y la tercera columna que son las que tienen información

```
cabecera  <- cabecera.raw[,2:3]
## tambien la podemos convertir a data frame.
cabecera <- as.data.frame(cabecera)
colnames(cabecera) <- c("Variable","Clase")
cabecera
```

```
                        Variable
1                         Gender
2                            Age
3                         Height
```

```
4                            Weight
5  family_history_with_overweight
6                              FAVC
7                              FCVC
8                               NCP
9                              CAEC
10                            SMOKE
11                             CH2O
12                              SCC
13                              FAF
14                              TUE
15                             CALC
16                           MTRANS
17                       NObeyesdad

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16                                                                {Automobile,Motorbike,Bi
17 {Insufficient_Weight,Normal_Weight,Overweight_Level_I,Overweight_Level_II,Obesity_Type_I,
```

Ya tenemos la cabecera, ahora vamos por los datos. Si recordamos lo habíamos guardado en la variable `predatos`. Tambiién habíamos observado que estaban separados por comas. Por lo tanto procedemos a separarlos por dicho caracter, y a parte, sabemos que los datos se componen 17 columnas. Especificamos que se ordenen por filas, mediante el comando `byrow=T`.

```r
datos <-
  as.data.frame(matrix(
    unlist(strsplit(predatos, ",")),
    ncol = nrow(cabecera),
```

```
    byrow = T
  ))
colnames(datos) <- cabecera$Variable
head(datos)
```

```
  Gender Age Height Weight family_history_with_overweight FAVC FCVC NCP
1 Female  21   1.62     64                            yes   no    2   3
2 Female  21   1.52     56                            yes   no    3   3
3   Male  23    1.8     77                            yes   no    2   3
4   Male  27    1.8     87                             no   no    3   3
5   Male  22   1.78   89.8                             no   no    2   1
6   Male  29   1.62     53                             no  yes    2   3
       CAEC SMOKE CH2O SCC FAF TUE       CALC                MTRANS
1 Sometimes    no    2  no   0   1         no Public_Transportation
2 Sometimes   yes    3 yes   3   0  Sometimes Public_Transportation
3 Sometimes    no    2  no   2   1 Frequently Public_Transportation
4 Sometimes    no    2  no   2   0 Frequently               Walking
5 Sometimes    no    2  no   0   0  Sometimes Public_Transportation
6 Sometimes    no    2  no   0   0  Sometimes            Automobile
          NObeyesdad
1       Normal_Weight
2       Normal_Weight
3       Normal_Weight
4  Overweight_Level_I
5 Overweight_Level_II
6       Normal_Weight
```

Preprocesado de datos

En este paso, necesitamos identificar qué variables son numéricas y cuales son factores.

```
str(datos)
```

```
'data.frame':   2111 obs. of  17 variables:
 $ Gender                        : chr  "Female" "Female" "Male" "Male" ...
 $ Age                           : chr  "21" "21" "23" "27" ...
 $ Height                        : chr  "1.62" "1.52" "1.8" "1.8" ...
 $ Weight                        : chr  "64" "56" "77" "87" ...
 $ family_history_with_overweight: chr  "yes" "yes" "yes" "no" ...
 $ FAVC                          : chr  "no" "no" "no" "no" ...
```

```
 $ FCVC                          : chr  "2" "3" "2" "3" ...
 $ NCP                           : chr  "3" "3" "3" "3" ...
 $ CAEC                          : chr  "Sometimes" "Sometimes" "Sometimes" "Sometimes" ...
 $ SMOKE                         : chr  "no" "yes" "no" "no" ...
 $ CH2O                          : chr  "2" "3" "2" "2" ...
 $ SCC                           : chr  "no" "yes" "no" "no" ...
 $ FAF                           : chr  "0" "3" "2" "2" ...
 $ TUE                           : chr  "1" "0" "1" "0" ...
 $ CALC                          : chr  "no" "Sometimes" "Frequently" "Frequently" ...
 $ MTRANS                        : chr  "Public_Transportation" "Public_Transportation" "Publ
 $ NObeyesdad                    : chr  "Normal_Weight" "Normal_Weight" "Normal_Weight" "Over
```

Todas están catalogadas como caracter. Bien podemos ir variable por variable y asignar la clase a la que corresponde, o podemos realizar lo siguiente.

```r
vars.numericas <- grep("numeric",cabecera$Clase)
datos[,vars.numericas]<- apply(datos[,vars.numericas]
                               , 2,
                               as.numeric)
datos[,-vars.numericas] <- lapply(datos[,-vars.numericas],
                                  as.factor)
str(datos)
```

```
'data.frame':   2111 obs. of  17 variables:
 $ Gender                        : Factor w/ 2 levels "Female","Male": 1 1 2 2 2 2 1 2 2 2 .
 $ Age                           : num  21 21 23 27 22 29 23 22 24 22 ...
 $ Height                        : num  1.62 1.52 1.8 1.8 1.78 1.62 1.5 1.64 1.78 1.72 ...
 $ Weight                        : num  64 56 77 87 89.8 53 55 53 64 68 ...
 $ family_history_with_overweight: Factor w/ 2 levels "no","yes": 2 2 2 1 1 1 2 1 2 2 ...
 $ FAVC                          : Factor w/ 2 levels "no","yes": 1 1 1 1 1 2 2 1 2 2 ...
 $ FCVC                          : num  2 3 2 3 2 2 3 2 3 2 ...
 $ NCP                           : num  3 3 3 3 1 3 3 3 3 3 ...
 $ CAEC                          : Factor w/ 4 levels "Always","Frequently",..: 4 4 4 4 4 4 4
 $ SMOKE                         : Factor w/ 2 levels "no","yes": 1 2 1 1 1 1 1 1 1 1 ...
 $ CH2O                          : num  2 3 2 2 2 2 2 2 2 2 ...
 $ SCC                           : Factor w/ 2 levels "no","yes": 1 2 1 1 1 1 1 1 1 1 ...
 $ FAF                           : num  0 3 2 2 0 0 1 3 1 1 ...
 $ TUE                           : num  1 0 1 0 0 0 0 0 1 1 ...
 $ CALC                          : Factor w/ 4 levels "Always","Frequently",..: 3 4 2 2 4 4 4
 $ MTRANS                        : Factor w/ 5 levels "Automobile","Bike",..: 4 4 4 5 4 1 3 4
 $ NObeyesdad                    : Factor w/ 7 levels "Insufficient_Weight",..: 2 2 2 6 7 2 2
```

Ahora bien, también podemos realizar una función con los pasos anteriores.

La siguiente función hace lo mismo que el código anterior, asignando a las variables la clase que corresponde.

```
read.arff <- function(file_name){
  archivo <- readLines(file_name)

  filas_cabecera <- grep("@attribute", predata)

  pre_columnas <- predata[filas_cabecera]
  pre_columnas.list <- strsplit(predata[filas_cabecera], " ")
  cabecera  <- cabecera.raw[, 2:3]
  cabecera <- as.data.frame(cabecera)
  colnames(cabecera) <- c("Variable", "Clase")
  datos <-
    as.data.frame(matrix(
      unlist(strsplit(predatos, ",")),
      ncol = nrow(cabecera),
      byrow = T
    ))
  colnames(datos) <- cabecera$Variable
  datos <- as.data.frame(datos)
  numericas <- grep("numeric",cabecera$Clase)
  datos[,numericas] <- lapply(datos[,numericas],as.numeric)
  datos[,-numericas] <- lapply(datos[,-numericas],as.factor)


  return(datos)

}
```

```
datos  <- read.arff(archivo)
str(datos)
```

```
'data.frame':   2111 obs. of  17 variables:
 $ Gender                         : Factor w/ 2 levels "Female","Male": 1 1 2 2 2 2 1 2 2 2 .
 $ Age                            : num  21 21 23 27 22 29 23 22 24 22 ...
 $ Height                         : num  1.62 1.52 1.8 1.8 1.78 1.62 1.5 1.64 1.78 1.72 ...
 $ Weight                         : num  64 56 77 87 89.8 53 55 53 64 68 ...
 $ family_history_with_overweight: Factor w/ 2 levels "no","yes": 2 2 2 1 1 1 2 1 2 2 ...
 $ FAVC                           : Factor w/ 2 levels "no","yes": 1 1 1 1 1 2 2 1 2 2 ...
```

```
 $ FCVC                          : num  2 3 2 3 2 2 3 2 3 2 ...
 $ NCP                           : num  3 3 3 3 1 3 3 3 3 3 ...
 $ CAEC                          : Factor w/ 4 levels "Always","Frequently",..: 4 4 4 4 4 4 4
 $ SMOKE                         : Factor w/ 2 levels "no","yes": 1 2 1 1 1 1 1 1 1 1 ...
 $ CH2O                          : num  2 3 2 2 2 2 2 2 2 2 ...
 $ SCC                           : Factor w/ 2 levels "no","yes": 1 2 1 1 1 1 1 1 1 1 ...
 $ FAF                           : num  0 3 2 2 0 0 1 3 1 1 ...
 $ TUE                           : num  1 0 1 0 0 0 0 0 1 1 ...
 $ CALC                          : Factor w/ 4 levels "Always","Frequently",..: 3 4 2 2 4 4 4
 $ MTRANS                        : Factor w/ 5 levels "Automobile","Bike",..: 4 4 4 5 4 1 3 4
 $ NObeyesdad                    : Factor w/ 7 levels "Insufficient_Weight",..: 2 2 2 6 7 2 2
```

Preguntar a los datos

```
head(datos)
```

```
  Gender Age Height Weight family_history_with_overweight FAVC FCVC NCP
1 Female  21   1.62   64.0                            yes   no    2   3
2 Female  21   1.52   56.0                            yes   no    3   3
3   Male  23   1.80   77.0                            yes   no    2   3
4   Male  27   1.80   87.0                             no   no    3   3
5   Male  22   1.78   89.8                             no   no    2   1
6   Male  29   1.62   53.0                             no  yes    2   3
       CAEC SMOKE CH2O SCC FAF TUE       CALC                MTRANS
1 Sometimes    no    2  no   0   1         no Public_Transportation
2 Sometimes   yes    3 yes   3   0  Sometimes Public_Transportation
3 Sometimes    no    2  no   2   1 Frequently Public_Transportation
4 Sometimes    no    2  no   2   0 Frequently              Walking
5 Sometimes    no    2  no   0   0  Sometimes Public_Transportation
6 Sometimes    no    2  no   0   0  Sometimes            Automobile
          NObeyesdad
1       Normal_Weight
2       Normal_Weight
3       Normal_Weight
4  Overweight_Level_I
5 Overweight_Level_II
6       Normal_Weight
```

```
cabecera
```

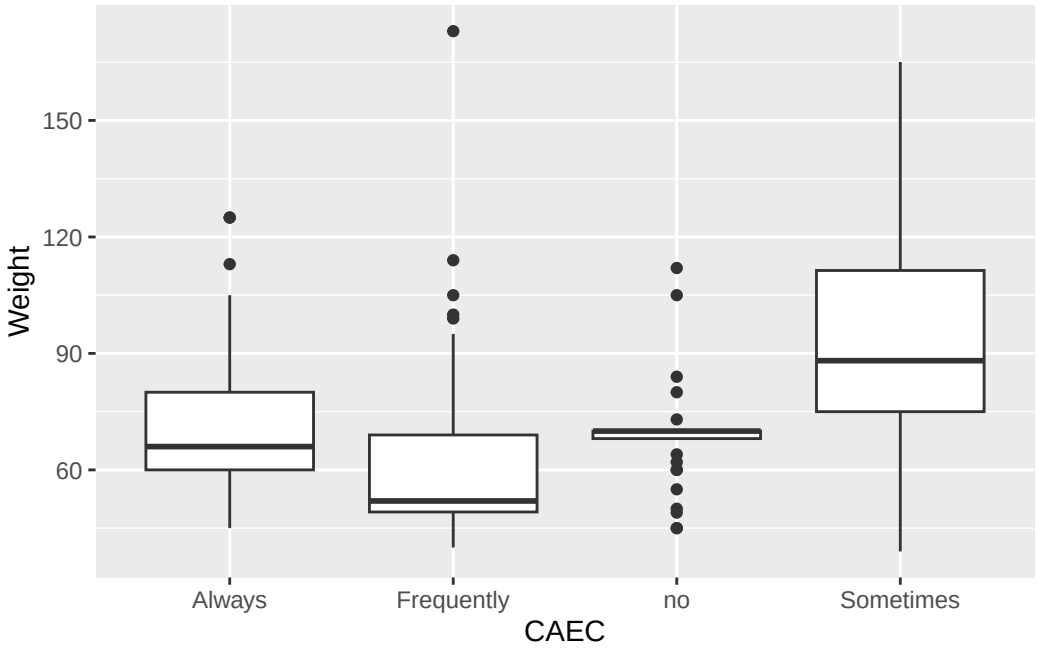
```
                          Variable
1                           Gender
2                              Age
3                           Height
4                           Weight
5  family_history_with_overweight
6                             FAVC
7                             FCVC
8                              NCP
9                             CAEC
10                           SMOKE
11                            CH2O
12                             SCC
13                             FAF
14                             TUE
15                            CALC
16                          MTRANS
17                      NObeyesdad
```

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16                                                                {Automobile,Motorbike,Bi
17 {Insufficient_Weight,Normal_Weight,Overweight_Level_I,Overweight_Level_II,Obesity_Type_I,
```

Vamos a realizar un ANOVA de 1 Vía. EL ANOVA es un modelo de regresión lineall donde las variables independientes son factores (o variables categóricas)

Pero primero tenemos que ver gráficamente

```r
library(ggplot2)
ggplot(datos,aes(y=Weight,CAEC))+geom_boxplot()
```




```r
mdl <- lm(Weight ~ CAEC,data=datos)
summary(mdl)
```

```
Call:
lm(formula = Weight ~ CAEC, data = datos)

Residuals:
    Min      1Q  Median      3Q     Max
-52.360 -15.360  -3.264  18.535 114.114

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)      71.091      3.268  21.752  < 2e-16 ***
CAECFrequently  -12.205      3.608  -3.382 0.000732 ***
CAECno           -2.188      4.667  -0.469 0.639241
CAECSometimes    20.270      3.317   6.111 1.18e-09 ***
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 23.79 on 2107 degrees of freedom
Multiple R-squared:  0.1759,    Adjusted R-squared:  0.1747
F-statistic: 149.9 on 3 and 2107 DF,  p-value: < 2.2e-16
```

```r
summary(aov(mdl))
```

```
              Df  Sum Sq Mean Sq F value Pr(>F)
CAEC           3  254594   84865   149.9 <2e-16 ***
Residuals   2107 1192818     566
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
DescTools::JarqueBeraTest(mdl$residuals-datos$Weight)
```

```
    Robust Jarque Bera Test

data:  mdl$residuals - datos$Weight
X-squared = 143247, df = 2, p-value < 2.2e-16
```

```r
pairwise.t.test(datos$Weight,datos$CAEC,"BH")
```

```
    Pairwise comparisons using t tests with pooled SD

data:  datos$Weight and datos$CAEC

           Always  Frequently no
Frequently 0.0011  -          -
no         0.6392  0.0076     -
Sometimes  2.4e-09 < 2e-16    1.2e-10

P value adjustment method: BH
```

```r
kruskal.test(Weight~CAEC,data=datos)
```

```
    Kruskal-Wallis rank sum test

data:  Weight by CAEC
Kruskal-Wallis chi-squared = 396.97, df = 3, p-value < 2.2e-16
```

```r
pairwise.wilcox.test(datos$Weight,datos$CAEC,p.adjust.method = "BH")
```

```
    Pairwise comparisons using Wilcoxon rank sum test with continuity correction

data:  datos$Weight and datos$CAEC

           Always  Frequently no
Frequently 7.5e-07 -          -
no         0.55    7.5e-07    -
Sometimes  6.3e-09 < 2e-16    1.6e-11

P value adjustment method: BH
```

Realizaremos una demostración de cómo el modelo lineal es una generalizacion del ANOVA y esta del t test

```r
mdl2  <- lm(datos$Weight~datos$SMOKE)
summary(aov(mdl2))
```

```
              Df  Sum Sq Mean Sq F value Pr(>F)
datos$SMOKE    1     959   959.5   1.399  0.237
Residuals   2109 1446453   685.8
```

```r
t.test(datos$Weight ~datos$SMOKE,data=datos,var.equal=T)
```

```
    Two Sample t-test

data:  datos$Weight by datos$SMOKE
t = -1.1828, df = 2109, p-value = 0.237
alternative hypothesis: true difference in means between group no and group yes is not equal
95 percent confidence interval:
 -12.543654   3.105427
sample estimates:
 mean in group no mean in group yes
         86.48770          91.20681
```