Tema 1: Preprocesado y análisis estadístico

MsC. Edmond Géraud

Lectura de datos

Primero de todo necesitamos cargar las librerías necesarias

```r
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
```

```r
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

```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_Weig
[4] "Female,21,1.52,56,yes,no,3,3,Sometimes,yes,3,yes,3,0,Sometimes,Public_Transportation,No
[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

```r
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.

```r
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.

```r
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)
```

```r
  return(datos)

}
```

```r
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
 $ 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
 $ MTRANS                        : Factor w/ 5 levels "Automobile","Bike",..: 4 4 4 5 4 1 3
 $ NObeyesdad                    : Factor w/ 7 levels "Insufficient_Weight",..: 2 2 2 6 7 2
```

Familiarización con los datos

[…]he attributes related with eating habits are: Frequent consumption of high caloric food (FAVC), Frequency of consumption of vegetables (FCVC), Number of main meals (NCP), Consumption of food between meals (CAEC), Consumption of water daily (CH20), and Consumption of alcohol (CALC). The attributes related with the physical condition are: Calories consumption monitoring (SCC), Physical activity frequency (FAF), Time using technology devices (TUE), Transportation used (MTRANS), other variables obtained were: Gender, Age, Height and Weight. Finally, all data was labeled and the class variable NObesity was created with the values of: Insufficient Weight, Normal Weight, Overweight Level I, Overweight Level II, Obesity Type I, Obesity Type II and Obesity Type III […]

Estadística numérica, gráfica e inferencial

La VI y VD son categóricas

Numérica

Los datos los componen 17 columnas y 2111 observaciones. Podemos realizar un summary de los datos.

```
summary(datos)
```

```
    Gender           Age             Height           Weight      
 Female:1043   Min.   :14.00   Min.   :1.450   Min.   : 39.00  
 Male  :1068   1st Qu.:19.95   1st Qu.:1.630   1st Qu.: 65.47  
               Median :22.78   Median :1.700   Median : 83.00  
               Mean   :24.31   Mean   :1.702   Mean   : 86.59  
               3rd Qu.:26.00   3rd Qu.:1.768   3rd Qu.:107.43  
               Max.   :61.00   Max.   :1.980   Max.   :173.00  

 family_history_with_overweight  FAVC           FCVC            NCP       
 no : 385                       no : 245   Min.   :1.000   Min.   :1.000  
 yes:1726                       yes:1866   1st Qu.:2.000   1st Qu.:2.659  
                                           Median :2.386   Median :3.000  
                                           Mean   :2.419   Mean   :2.686  
                                           3rd Qu.:3.000   3rd Qu.:3.000  
                                           Max.   :3.000   Max.   :4.000  

         CAEC       SMOKE          CH2O        SCC            FAF        
 Always    :  53   no :2067   Min.   :1.000   no :2015   Min.   :0.0000  
 Frequently: 242   yes:  44   1st Qu.:1.585   yes:  96   1st Qu.:0.1245  
 no        :  51              Median :2.000              Median :1.0000  
 Sometimes :1765              Mean   :2.008              Mean   :1.0103  
                              3rd Qu.:2.477              3rd Qu.:1.6667  
                              Max.   :3.000              Max.   :3.0000  

      TUE                 CALC                         MTRANS    
 Min.   :0.0000   Always    :   1   Automobile           : 457  
 1st Qu.:0.0000   Frequently:  70   Bike                 :   7  
 Median :0.6253   no        : 639   Motorbike            :  11  
 Mean   :0.6579   Sometimes :1401   Public_Transportation:1580  
 3rd Qu.:1.0000                     Walking              :  56  
 Max.   :2.0000                                                 
```

```
                NObeyesdad
 Insufficient_Weight:272
 Normal_Weight      :287
 Obesity_Type_I     :351
 Obesity_Type_II    :297
 Obesity_Type_III   :324
 Overweight_Level_I :290
 Overweight_Level_II:290
```

Le podemos preguntar al conjunto de datos, cuantas variables son factores:

```r
factores <- colnames(datos)[which(unlist(lapply(datos,is.factor)))]
```

Nos podemos preguntar, en este conjunto de datos, cómo está relacionado el género con los distintos niveles de obesidad

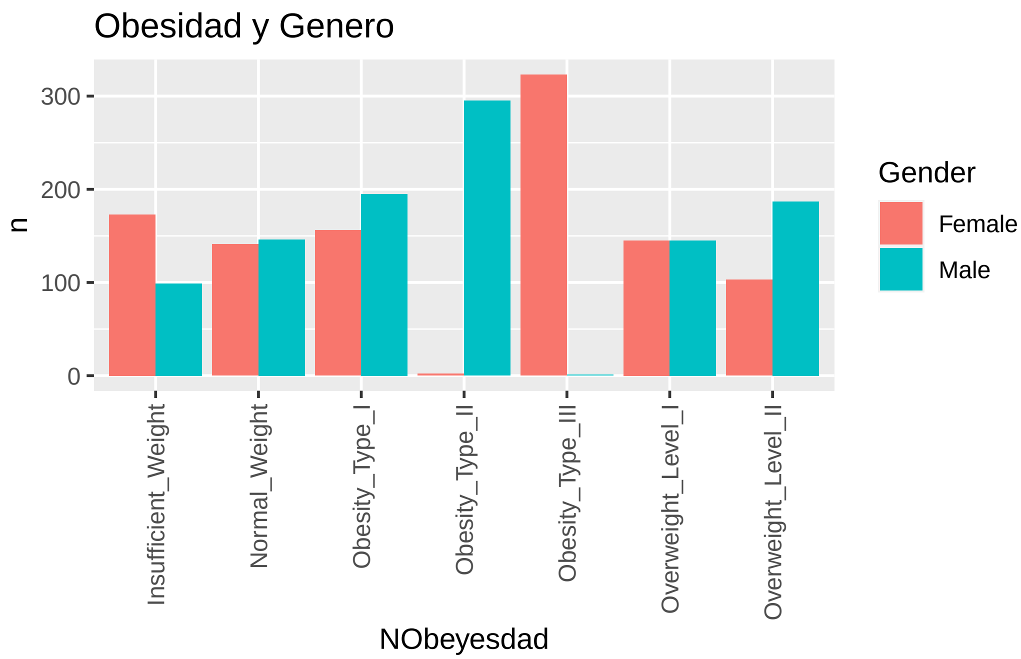
```r
(obesidad.genero  <- datos %>% group_by(Gender,NObeyesdad) %>% reframe(n=n()) )
```

```
# A tibble: 14 x 3
   Gender NObeyesdad              n
   <fct>  <fct>               <int>
 1 Female Insufficient_Weight   173
 2 Female Normal_Weight         141
 3 Female Obesity_Type_I        156
 4 Female Obesity_Type_II         2
 5 Female Obesity_Type_III      323
 6 Female Overweight_Level_I    145
 7 Female Overweight_Level_II   103
 8 Male   Insufficient_Weight    99
 9 Male   Normal_Weight         146
10 Male   Obesity_Type_I        195
11 Male   Obesity_Type_II       295
12 Male   Obesity_Type_III        1
13 Male   Overweight_Level_I    145
14 Male   Overweight_Level_II   187
```

Gráfica

Antes sería necesario graficar qué es lo que observamos, mediante unas barras

```r
ggplot(obesidad.genero,aes(y=n,x=NObeyesdad,fill=Gender)) + geom_bar(stat = "identity",pos
```



Sin embargo, al revisar el grafico, observamos que tienen números muy dispares, por tantos niveles en la clasificacion de la obesidad.

```r
datos$obesidad  <- as.character(datos$NObeyesdad)
datos$obesidad[datos$obesidad=="Insufficient_Weight"]
```

```
 [1] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [4] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [7] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[10] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[13] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[16] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[19] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[22] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[25] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[28] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[31] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[34] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[37] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
```

```
 [40] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [43] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [46] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [49] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [52] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [55] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [58] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [61] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [64] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [67] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [70] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [73] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [76] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [79] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [82] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [85] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [88] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [91] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [94] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
 [97] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[100] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[103] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[106] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[109] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[112] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[115] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[118] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[121] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[124] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[127] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[130] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[133] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[136] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[139] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[142] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[145] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[148] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[151] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[154] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[157] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[160] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[163] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[166] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
```

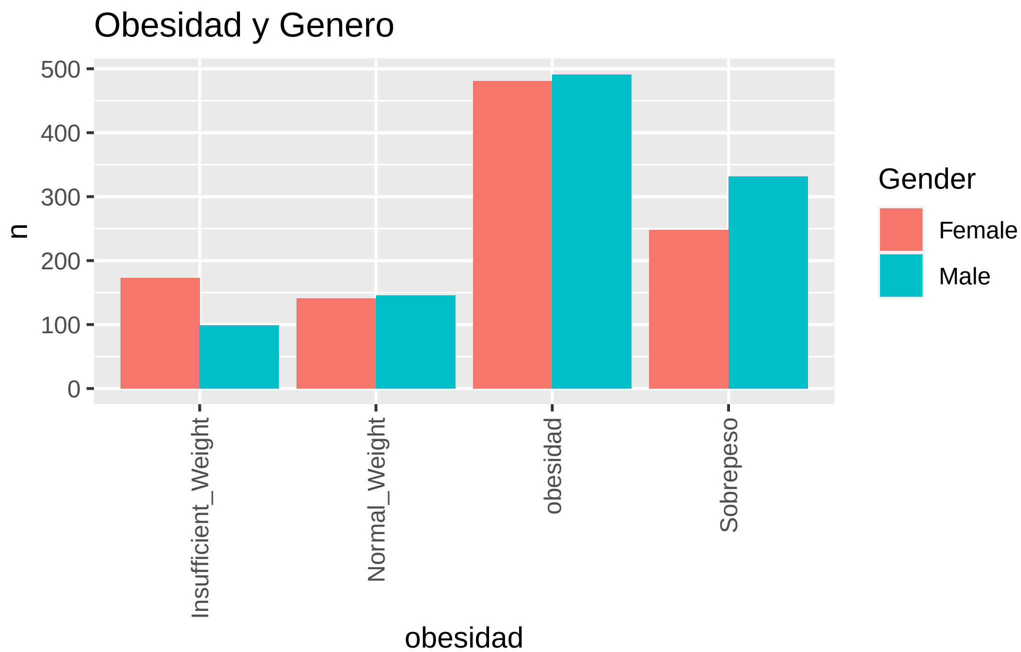
```
[169] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[172] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[175] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[178] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[181] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[184] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[187] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[190] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[193] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[196] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[199] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[202] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[205] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[208] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[211] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[214] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[217] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[220] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[223] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[226] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[229] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[232] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[235] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[238] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[241] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[244] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[247] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[250] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[253] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[256] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[259] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[262] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[265] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[268] "Insufficient_Weight" "Insufficient_Weight" "Insufficient_Weight"
[271] "Insufficient_Weight" "Insufficient_Weight"
```

```
datos$obesidad[datos$obesidad=="Obesity_Type_I" |
                 datos$obesidad=="Obesity_Type_II"|
                 datos$obesidad=="Obesity_Type_III"] <- "obesidad"
datos$obesidad[datos$obesidad=="Overweight_Level_I"|
                 datos$obesidad=="Overweight_Level_II"] <- "Sobrepeso"
```

```
(obesidad.genero  <- datos %>% group_by(Gender,obesidad) %>% reframe(n=n()) )
```

```
# A tibble: 8 x 3
  Gender obesidad                 n
  <fct>  <chr>                <int>
1 Female Insufficient_Weight    173
2 Female Normal_Weight          141
3 Female Sobrepeso              248
4 Female obesidad               481
5 Male   Insufficient_Weight     99
6 Male   Normal_Weight          146
7 Male   Sobrepeso              332
8 Male   obesidad               491
```

```
ggplot(obesidad.genero,aes(y=n,x=obesidad,fill=Gender)) + geom_bar(stat = "identity",posit
```



Inferencial

Ahora ya nos podemos preguntar si los niveles en los que se le clasifica el peso a las personas esta mas ligado al sexo o no

```
chisq.test(table(datos$obesidad,datos$Gender))
```

```

	Pearson's Chi-squared test

data:  table(datos$obesidad, datos$Gender)
X-squared = 32.196, df = 3, p-value = 4.758e-07
```

Efectivamente como el p valor es menor que 0.05, se rechaza la hipotesis nula de no asocicion entre el genero y la clasificacion del peso.

Si bien queremos preguntarnos si la obesidad, ya sea de tipo I o II, esta mas ligada a los hombres, que aparenta serlo, debemos confirmarlo con el test chi cuadrado

```
subsetdatos <- datos[datos$obesidad=="obesidad",c("obesidad","Gender")]
subsetdatos$obesidad  <- as.factor(as.character(subsetdatos$obesidad))
chisq.test(table(subsetdatos$obesidad,subsetdatos$Gender))
```

```

	Chi-squared test for given probabilities

data:  table(subsetdatos$obesidad, subsetdatos$Gender)
X-squared = 0.10288, df = 1, p-value = 0.7484
```

En realidad no hay asociacion entre la obesidad y el genero.

La VI es categórica y la VD es numérica

En este caso, estaríamos pensando en una diferencia de medias o medianas.

SI LA VARIABLE CATEGÓRICA TIENE 2 NIVELES.

Nos podemos preguntar si existen diferencias en la altura respecto al género

Es decir, si realizamos una prueba de diferencia de medias, la hipótesis sería

$$H_0 : \mu_h = \mu_m H_0 : \mu_h - \mu_m = 0$$

```r
altura <- datos$Height
genero <- datos$Gender
```

Cómo hemos mencionado en clase, tenemos ciertas suposiciones a seguir. Donde siempre optaremos en una primera instancia por modelos paramétricos, antes de los no paramétricos, si se cumplen dichas suposicones

1. **T-TEST**

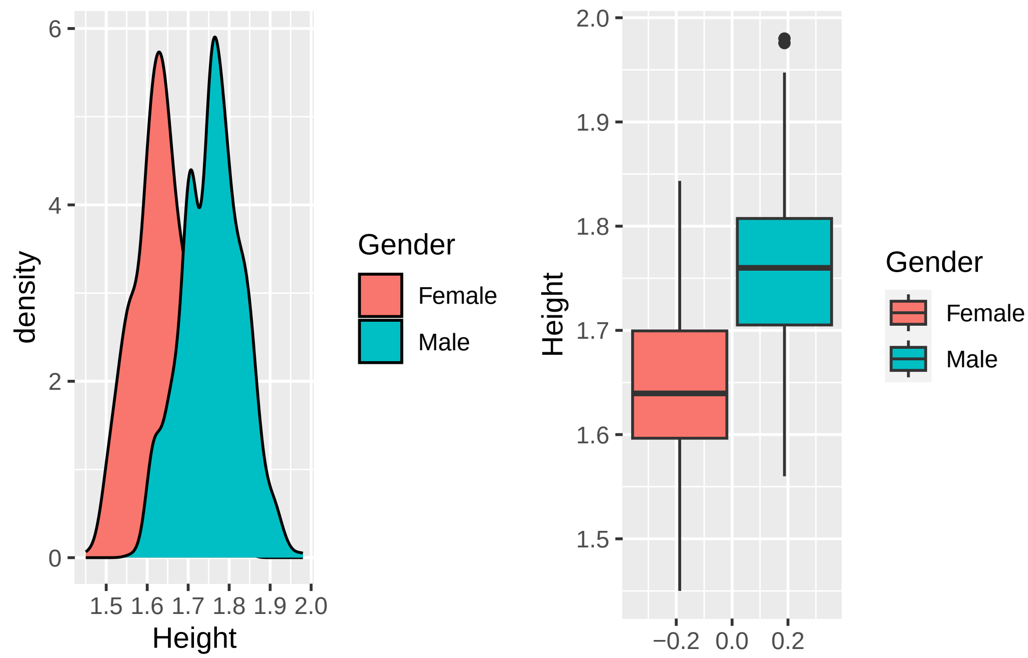
    1. La normalidad de los residuos, o la normalidad entre los niveles tiene que cumplirse
    2. Segun si tenemos varianzas iguales o no procederemos de otro modo

```r
lista.altura <- split(altura,genero)
lapply(lista.altura,ks.test,"pnorm")
```

```
Warning in ks.test.default(X[[i]], ...): ties should not be present for the
Kolmogorov-Smirnov test

Warning in ks.test.default(X[[i]], ...): ties should not be present for the
Kolmogorov-Smirnov test

$Female

	Asymptotic one-sample Kolmogorov-Smirnov test

data:  X[[i]]
D = 0.92865, p-value < 2.2e-16
alternative hypothesis: two-sided


$Male

	Asymptotic one-sample Kolmogorov-Smirnov test

data:  X[[i]]
D = 0.94333, p-value < 2.2e-16
alternative hypothesis: two-sided
```

En una primera instancia, la prueba de kolmogorov nos dice que no siguen una normal los datos. No obstante si graficamos la densidad… observamos cosas diferentes

```
p1 <- ggplot(datos,aes(x=Height,fill=Gender))+geom_density()
p2 <- ggplot(datos,aes(y=Height,fill=Gender))+geom_boxplot()
ggarrange(p1,p2)
```



De hecho, al ser tantnas observaciones, nos podemos fiar que los datos siguen una normal. Además, la media y la mediana estan bastante cerca entre sí.

```
lapply(lista.altura,summary)
```

```
$Female
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.450   1.597   1.640   1.643   1.700   1.843

$Male
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.560   1.705   1.760   1.759   1.807   1.980
```

Ahora tendríamos que ver la igualdad de varianzas

```
leveneTest(altura ~ genero)
```

```
Levene's Test for Homogeneity of Variance (center = median)
        Df F value Pr(>F)
group    1  1.3857 0.2393
      2109
```

Efectivamente la prueba de levene nos confirma que tenemos varianzas iguales. Entonces procedemos

```r
t.test(altura ~ genero,var.equal=T)
```

```

	Two Sample t-test

data:  altura by genero
t = -36.144, df = 2109, p-value < 2.2e-16
alternative hypothesis: true difference in means between group Female and group Male is not e
95 percent confidence interval:
 -0.1216535 -0.1091316
sample estimates:
mean in group Female   mean in group Male 
            1.643298             1.758690 
```

Y efectivamente como observamos en el grafico son diferentes.