10ImportarDatos_cheatsheet.R

moka

2023-04-20

```
# Autor: Monika Avila Marquez, Ph.D.
# Fecha: 12.04.2023
# Objetivo: Importar datos que no estan en R de excel, csv, etc.
# Referencia: Basado en R Programming Fundamentals, StanfordOnline XDFS112
```

```
# Limpiar el espacio de trabajo
rm(list=ls())
```

```
# Configurar el directorio
```

```
midirectorio<-setwd("~/Dropbox/0.POST-PHD/GOALS/2.CODE/R/Ecomienza/10ImportarDatos")
midirectorio
```

```
## [1] "/Users/moka/Dropbox/0.POST-PHD/GOALS/2.CODE/R/Ecomienza/10ImportarDatos"
```

```
# Obetener los archivos disponibles en el directorio
dir()
```

```
##  [1] "10ImportarDatos_cheatsheet.aux"      "10ImportarDatos_cheatsheet.html"
##  [3] "10ImportarDatos_cheatsheet.R"        "10ImportarDatos_cheatsheet.spin.R"
##  [5] "10ImportarDatos_cheatsheet.spin.Rmd" "10ImportarDatos_cheatsheet.tex"
##  [7] "10ImportarDatos_conceptos.aux"       "10ImportarDatos_conceptos.tex"
##  [9] "10ImportarDatos_lab.aux"             "10ImportarDatos_lab.log"
## [11] "10ImportarDatos_lab.tex"
```

```
list.files()
```

```
##  [1] "10ImportarDatos_cheatsheet.aux"      "10ImportarDatos_cheatsheet.html"
##  [3] "10ImportarDatos_cheatsheet.R"        "10ImportarDatos_cheatsheet.spin.R"
##  [5] "10ImportarDatos_cheatsheet.spin.Rmd" "10ImportarDatos_cheatsheet.tex"
##  [7] "10ImportarDatos_conceptos.aux"       "10ImportarDatos_conceptos.tex"
##  [9] "10ImportarDatos_lab.aux"             "10ImportarDatos_lab.log"
## [11] "10ImportarDatos_lab.tex"
```

```
# Importar datos en R
```

```
# 1. Archivo txt (archivo tipo ascii, con extension .txt): este tipo. de archivo es lisible por humanos
# de archivos son mas dificiles de leer.
  # Ver el archivo
    #file.show("~/Dropbox/0.POST-PHD/GOALS/2.CODE/R/Ecomienza/datos/creados/matH2.txt")
  # Importar los datos
    datostxt<-read.table("~/Dropbox/0.POST-PHD/GOALS/2.CODE/R/Ecomienza/datos/creados/matH2.txt")
  # Ver los datos
    datostxt
```

```
##           v1        v2
## I1 0.2740892 0.6317251
```

```
## I2 0.5920839 0.1891507
```

```
  # Ver que tipo de datos es
    class(datostxt)
```

```
## [1] "data.frame"
```

```
    str(datostxt)
```

```
## 'data.frame':    2 obs. of  2 variables:
##  $ v1: num  0.274 0.592
##  $ v2: num  0.632 0.189
```

```
    is.matrix(datostxt)
```

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

```
    # We call this a tidy table because each row is an observation and each column is a variable
# 2. Archivo csv (comma separated variables): la ventaja es que se puede ver
    #este archivo en R
    ?dir
    dir("datos/raw")
```

```
## character(0)
```

```
    # Ver los datos
    file.show("~/Dropbox/0.POST-PHD/GOALS/2.CODE/R/Ecomienza/datos/raw/Haplotype.csv")
    # Importar datos
    ?read.csv # Default para el separadores ,
    Hap<-read.csv("~/Dropbox/0.POST-PHD/GOALS/2.CODE/R/Ecomienza/datos/raw/Haplotype.csv")
    head(Hap)
```

```
##   Individual DYS19 DXYS156Y DYS389m DYS389n DYS389p DYS389q DYS390m DYS390n DYS390p DYS390q DYS392
## 1         H1    14       12       4      12       3      10       8      10       1       4     15
## 2         H3    15       13       4      13       3       9       8      10       1       4     13
## 3         H4    15       11       5      11       3      10       8      10       1       4     11
## 4         H5    17       13       4      11       3      10       7      10       1       4     14
## 5         H7    13       12       5      12       3      11       8      11       1       4     14
## 6         H8    16       11       5      12       3      10       8      10       1       4     11
##   DYS393 YAPbcbc SRY1532bb X92R7bb
## 1     13       0         1       1
## 2     12       0         1       1
## 3     14       0         1       1
## 4     12       0         1       1
## 5     14       0         1       1
## 6     15       0         1       1
```

```
      # Debugging read.csv output
      # Problema, el nombre de las filas es considerado como una variable
      # llamada individual, pero inicialmente este es solamente un identificador
      # Para evitar esto, podemos poner las opciones de rownames, y colnames.
    Hap1<-read.csv("~/Dropbox/0.POST-PHD/GOALS/2.CODE/R/Ecomienza/datos/raw/Haplotype.csv",row.names = 1
    head(Hap1)
```

```
##    DYS19 DXYS156Y DYS389m DYS389n DYS389p DYS389q DYS390m DYS390n DYS390p DYS390q DYS392 DYS393
## H1    14       12       4      12       3      10       8      10       1       4     15     13
## H3    15       13       4      13       3       9       8      10       1       4     13     12
## H4    15       11       5      11       3      10       8      10       1       4     11     14
## H5    17       13       4      11       3      10       7      10       1       4     14     12
```

```
## H7    13       12       5      12       3      11       8      11       1       4     14     14
## H8    16       11       5      12       3      10       8      10       1       4     11     15
##    YAPbcbc SRY1532bb X92R7bb
## H1       0         1       1
## H3       0         1       1
## H4       0         1       1
## H5       0         1       1
## H7       0         1       1
## H8       0         1       1
```

```
    # Ahora, ya no tenemos la variable individual como variable factor.
    # Utilizar la funcion scan
    scan("~/Dropbox/0.POST-PHD/GOALS/2.CODE/R/Ecomienza/datos/raw/Haplotype.csv",nlines=5,what="")
```

```
## [1] "Individual,DYS19,DXYS156Y,DYS389m,DYS389n,DYS389p,DYS389q,DYS390m,DYS390n,DYS390p,DYS390q,DYS391
## [2] "H1,14,12,4,12,3,10,8,10,1,4,15,13,0,1,1"
## [3] "H3,15,13,4,13,3,9,8,10,1,4,13,12,0,1,1"
## [4] "H4,15,11,5,11,3,10,8,10,1,4,11,14,0,1,1"
## [5] "H5,17,13,4,11,3,10,7,10,1,4,14,12,0,1,1"
```

```
    ?scan
# 3. Archivo Excel con extensiones .xlsx, o .xls
    #install.packages("readxl")
    library(readxl)
    ?read_excel
    Hap2<-read_excel("~/Dropbox/0.POST-PHD/GOALS/2.CODE/R/Ecomienza/datos/raw/Haplotype.xlsx")
    str(Hap2)
```

```
## tibble [24 x 16] (S3: tbl_df/tbl/data.frame)
##  $ Individual: chr [1:24] "H1" "H3" "H4" "H5" ...
##  $ DYS19     : num [1:24] 14 15 15 17 13 16 16 15 16 15 ...
##  $ DXYS156Y  : num [1:24] 12 13 11 13 12 11 11 12 13 12 ...
##  $ DYS389m   : num [1:24] 4 4 5 4 5 5 5 4 4 4 ...
##  $ DYS389n   : num [1:24] 12 13 11 11 12 12 11 12 12 12 ...
##  $ DYS389p   : num [1:24] 3 3 3 3 3 3 3 3 3 3 ...
##  $ DYS389q   : num [1:24] 10 9 10 10 11 10 10 10 10 10 ...
##  $ DYS390m   : num [1:24] 8 8 8 7 8 8 8 8 8 8 ...
##  $ DYS390n   : num [1:24] 10 10 10 10 11 10 10 10 11 9 ...
##  $ DYS390p   : num [1:24] 1 1 1 1 1 1 1 1 1 1 ...
##  $ DYS390q   : num [1:24] 4 4 4 4 4 4 4 4 4 4 ...
##  $ DYS392    : num [1:24] 15 13 11 14 14 11 11 14 12 14 ...
##  $ DYS393    : num [1:24] 13 12 14 12 14 15 14 13 12 13 ...
##  $ YAPbcbc   : num [1:24] 0 0 0 0 0 0 0 0 0 0 ...
##  $ SRY1532bb : num [1:24] 1 1 1 1 1 1 1 1 1 1 ...
##  $ 92R7bb    : num [1:24] 1 1 1 1 1 1 1 1 1 1 ...
```

```
    class(Hap2)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
    # read_excel uploads the data in a different format, and we can just convert it to dataframe
    # Convert it to a data frame
    Hap2df<-as.data.frame(Hap2)
    str(Hap2df)
```

```
## 'data.frame':    24 obs. of  16 variables:
##  $ Individual: chr  "H1" "H3" "H4" "H5" ...
```

```
##  $ DYS19     : num  14 15 15 17 13 16 16 15 16 15 ...
##  $ DXYS156Y  : num  12 13 11 13 12 11 11 12 13 12 ...
##  $ DYS389m   : num  4 4 5 4 5 5 5 4 4 4 ...
##  $ DYS389n   : num  12 13 11 11 12 12 11 12 12 12 ...
##  $ DYS389p   : num  3 3 3 3 3 3 3 3 3 3 ...
##  $ DYS389q   : num  10 9 10 10 11 10 10 10 10 10 ...
##  $ DYS390m   : num  8 8 8 7 8 8 8 8 8 8 ...
##  $ DYS390n   : num  10 10 10 10 11 10 10 10 11 9 ...
##  $ DYS390p   : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ DYS390q   : num  4 4 4 4 4 4 4 4 4 4 ...
##  $ DYS392    : num  15 13 11 14 14 11 11 14 12 14 ...
##  $ DYS393    : num  13 12 14 12 14 15 14 13 12 13 ...
##  $ YAPbcbc   : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ SRY1532bb : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ 92R7bb    : num  1 1 1 1 1 1 1 1 1 1 ...
```

```
# 4. Importar datos de otros softwares
    #install.packages("foreign")
    library(foreign)
    # spss .sav, read.spss
    # stata .dta, read.dta
    ?read.dta
# 5. Scraping la web y leyendo datos direcatmente (se necesita conexion al internet)
    # Datos que vienen de un Github
    births<-read.csv(url("https://raw.githubusercontent.com/fivethirtyeight/data/master/births/US_births
    head(births)
```

```
##   year month date_of_month day_of_week births
## 1 2000     1             1           6   9083
## 2 2000     1             2           7   8006
## 3 2000     1             3           1  11363
## 4 2000     1             4           2  13032
## 5 2000     1             5           3  12558
## 6 2000     1             6           4  12466
```

```
    # Nota: Scraping datos con paquete rvest
# Bonus: Guardar acrhivo
    write.csv(births,file="~/Dropbox/0.POST-PHD/GOALS/2.CODE/R/Ecomienza/datos/procesados/births1.csv")
    file.show("~/Dropbox/0.POST-PHD/GOALS/2.CODE/R/Ecomienza/datos/procesados/births1.csv")
    # Mejores practicas creando archivos de datos: no espacios en nombre de variables
    # csv ocupa bastante espacio, se puede crear un archivo binario usando
    # save file en binary R formato
    save(births,file="~/Dropbox/0.POST-PHD/GOALS/2.CODE/R/Ecomienza/datos/procesados/births1.Rdata")
    load("~/Dropbox/0.POST-PHD/GOALS/2.CODE/R/Ecomienza/datos/procesados/births1.Rdata")
    # Formato binario de R es mas eficiente y rapido de leer
    # Recursos: R import data
    # https://cran.r-project.org/doc/manuals/r-devel/R-data.html

    # No quitar el comentario de la linea inferior. Solamente copiar en la consola para que ejecute
    #rmarkdown::render("10ImportarDatos_cheatsheet.R",c("pdf_document","html_document"))
```