PRACTICA_04

Nelson de Jesus Magaña Godinez

3/8/2022

Contents

Generalmente los datos suelen leerse desde archivos externos y no teclearse desde la consola. Las capacidades de lectura de archivos de R son sencillas y sus requisitos son bastante estrictos, por lo que hay que tenerlas muy en cuenta, de lo contrario los resultados en la lectura no serán los esperados.

1. USO DE LA FUNCIÓN READ.TABLE().

Ejemplo: Guardar (escribir) determinados datos en un archivo de texto (ASCII) y luego recuperar (leer) dicho archivo desde R.

1º) Cambiar el directorio de trabajo a su directorio de trabajo, en el cual ha almacenado sus prácticas, desde el menú File.

2º) Abrir el R Editor para crear un nuevo script desde el menú File.

3º) En la ventana del R Editor, teclee los datos tal como se muestra:

Observaciones:

- La primera línea del archivo debe contener el nombre de cada objeto o variable.
- En cada una de las siguientes líneas, el primer elemento es la etiqueta de la fila, y a continuación deben aparecer los valores de cada variable.
- Si el archivo tiene un elemento menos en la primera línea que en las restantes, obligatoriamente será el diseño anterior el que se utilice.
- A menudo no se dispone de etiquetas de filas. En ese caso, también es posible la lectura y el programa añadirá unas etiquetas predeterminadas.
- La última línea debe finalizar con ENTER para que R reconozca el fin del archivo.

4º) Oprimir con el puntero del ratón el icono que representa un disquete (Save script as) y guarde el archivo con el nombre "datos01.txt". También puede darle el nombre de "datos01.dat" (otro formato soportado por la función read.table), e incluso puede leer datos directamente desde una página de internet, solamente proporcionando la dirección URL completa.

El archivo no se guarda con extensión .R, porque no es un script, sino un archivo de datos. en formato ".dat" o ".txt"

También puede realizar estos pasos utilizando un editor de texto como NotePad o WordPad.

5º) Recuperar los objetos o datos guardados en el archivo "datos01.txt"

```
entrada1 <- read.table("datos01.txt", header = T);
entrada1
```

```
##   Edad Estatura Peso Sexo
## 1   26     1.65  146    F
## 2   21     1.73  158    M
## 3   21     1.81  167    M
## 4   20     1.70  152    F
```

```
entrada2 <- read.table("datos01.dat", header = T);
entrada2
```

```
##   Edad Estatura Peso Sexo
## 1   26     1.65  146    F
## 2   21     1.73  158    M
## 3   21     1.81  167    M
## 4   20     1.70  152    F
```

```
# No existe diferencia entre ambos archivos a la hora de leerlos
```

NOTA: La función read.table() lee los datos y los almacena en una hoja de datos (data.frame), si quiere hacer operaciones debe conectar esta hoja a la trayectoria de búsqueda.

6º) Leer los datos contenidos en el archivo "mexico.dad"

```
Mexico <- read.table("mexico.dat");
#Mexico
# Note que la instrucción header=T es por defecto y puede omitirla
#(R reconocerá siempre que en la primera línea se encuentran los
# nombres de las variables).
```

La sintaxis completa de la función read.table() es

```
#read.table(file, header = FALSE, sep = "", quote = " \ "'", dec = ".", row.names, col.names, as.is = F
```

2. USO DE LA FUNCIÓN SCAN().

La función scan() es más flexible que read.table() y permite realizar lecturas más complejas, como puede consultar en la ayuda: help(scan)

- *Ejemplo 1:* Leer sólo las dos primeros objetos o columnas del archivo "datos01.txt"

```
Edat1 <- scan("datos01.txt", list(X1=0, X2=0), skip = 1, flush = TRUE,
              quiet = TRUE);
Edat1
```

```
## $X1
## [1] 26 21 21 20
##
## $X2
## [1] 1.65 1.73 1.81 1.70
```

```
Edat2 <- scan("datos01.dat", list(X1=0, X2=0),
              skip = 1, flush = TRUE, quiet = TRUE);
Edat2
```

```
## $X1
## [1] 26 21 21 20
##
## $X2
## [1] 1.65 1.73 1.81 1.70
```

Observe que en list(X1=0, X2=0) se les da el nombre a las dos primeras columnas o variables (puede darle el nombre que crea más conveniente) y se indica que son variables numéricas; sin embargo, del archivo únicamente se leen las dos primeras columnas, si se quisiera leer las columnas primera y tercera, nos veríamos obligados a leer las tres primeras.

```
Edat3 <- scan("datos01.txt", list(X1=0, X2=0, X3=0), skip = 1, flush = TRUE,
              quiet = TRUE);
Edat3 # Si solo se sustituye X3 en X2 entonces se muestra siempre X@
```

```
## $X1
## [1] 26 21 21 20
##
## $X2
## [1] 1.65 1.73 1.81 1.70
##
## $X3
## [1] 146 158 167 152
```

Note que si escribimos list(0, 0), indica que se leerán las dos primeras columnas del archivos y que los datos leídos son numéricos (asigna nombres por defecto). Para indicar que los datos que se leen son cadenas se utiliza " " en lugar de 0.

```
Edat4 <- scan("datos01.txt", list(0, 0,0, ""), skip = 1, flush = TRUE,
              quiet = TRUE);
Edat4
```

```
## [[1]]
## [1] 26 21 21 20
##
## [[2]]
## [1] 1.65 1.73 1.81 1.70
##
## [[3]]
## [1] 146 158 167 152
##
## [[4]]
## [1] "F" "M" "M" "F"
```

- *Ejemplo 2:* Crear un archivo con la función cat() y luego recuperarlo

```
datos02.txt <- cat("TITULO Linea extra", "2,3,5,7", "11,13,17", file = "datos02.txt", sep = "\n")
datos02.txt
```

```
## NULL
```

El archivo lo recuperamos con la función scan():

```
#pp <- scan("datos02.txt", skip = 1, quiet = TRUE);
#pp
# La función scan es muy útil cuando en el archivo de datos a importar cada línea representa un único c
```

La sintaxis completa de la función scan() es:

```
#scan(file = "", what = double(0), nmax = -1, n = -1, sep = "", quote = if (sep=="\n") "" else "'\"", d
#quiet = FALSE, blank.lines.skip = TRUE, multi.line = TRUE, comment.char = "#")
```

3. USO DE LA FUNCIÓN READ.CSV().

Leer un conjunto de datos de Microsoft Excel pero los datos no están almacenados en el formato conocido de Excel ".xls", sino más bien un formato menos conocido como ".csv".

1º) Ingresar al Microsoft Excel y crear la hoja de datos siguiente:

Observe que debe guardar la hoja Excel en su directorio de trabajo y que el archivo debe ser de tipo: CSV(delimitado por comas)

2º) Regresar al entorno de R y recuperar el archivo "HojaE1.csv".

```
hojaR <- read.csv("HojaE1.csv", sep = ";", strip.white = TRUE)
hojaR
```

```
##        Producto Cantidad.S1 Cantidad.S2 Cantidad.S3 Cantidad.S4
## 1      Desayuno         132         125         142         120
## 2     Almuerzos         120         125         122         114
## 3         Cenas         115         105         130         108
## 4 Tazas de café         200         180         210         140
## 5      Gaseosas          75          90          62          60
```

Note que R ha reemplazado "–" en los encabezados de las columnas por "."; en general reemplazará cualquier carácter.

Puede investigar el tipo de objeto que es hojaR con:

```
is.matrix(hojaR);
```

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

```
is.list(hojaR);
```

```
## [1] TRUE
```

```
is.data.frame(hojaR)
```

```
## [1] TRUE
```

Acceda a la componente Producto de hojaR con:

```
hojaR$Producto
```

```
## [1] "Desayuno"      "Almuerzos"     "Cenas"         "Tazas de café"
## [5] "Gaseosas"
```

Observe que R toma está columna (variable de caracteres) como un Factor Nominal, verifíquelo tecleando:

```
is.vector(hojaR$Producto);
```

```
## [1] TRUE
```

```
is.factor(hojaR$Producto)
```

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

¿Qué tipo de objeto es la columna Cantidad.S1?

```
is.vector(hojaR$Cantidad.S1); # Es un vector
```

```
## [1] TRUE
```

```
is.factor(hojaR$Cantidad.S1)
```

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

4. USO DEL PAQUETE RODBC.

Si por el contrario los datos a los cuales deseamos realizar el análisis estadístico se encuentran en formato XLS (versión 2003 de Microsoft Excel), debemos de seguir los siguientes pasos (Ilustraremos el procedimiento con el archivo "contaminación_mexico.xls"):

- Instalar el paquete RODBC, con la siguiente instrucción install.packages(c("RODBC")) o desde el menú como en el caso de la instalación del paquete Foreing. # Con este procedimiento se instalan los paquetes directamente desde internet, es necesario para ello contar con una conexión a internet en el momento. Posteriormente se selecciona un mirror (un servidor desde el cual se descargarán los paquetes), y finalmente buscar el paquete deseado del listado.

```
#install.packages("RODBC")
```

- Cargar el paquete con la siguiente instrucción: library(RODBC)

```
#library(RODBC)
```

- Seleccionar el archivo (el cual puede contener más de una hoja de datos)

```
#"contaminación_mexico.xls", con la instrucción:
#datos.xls <- odbcConnectExcel(file.choose())
```

- Seleccionar la hoja en la cual se encuentran los datos

```
#datoshoja1.xls <- sqlFetch(contaminacion_mexico,"contaminacion_mexico")
library(readxl)
contaminacion_mexico <- read_excel("C:/Users/pc 1/Desktop/PAQUETE R/PRACTICAS-_1/contaminacion_mexico.x
    sheet = "contaminacion_mexico")
#View(contaminacion_mexico)
# Con esta instrucción se indica la hoja en la cual se encuentran los datos con los que se
```

desea trabajar (contaminación_mexico) o cargar en R. Siempre es necesario especificarlo.

- Realizar los análisis o cálculos correspondientes.

5. IMPORTAR DATOS DE SPSS HACIA R.

A parte de leer archivos en formato texto y delimitados por comillas, R permite leer datos en una gran variedad de formato entre ellos se encuentra archivos el formato de SPSS ".sav". Para poder leerlos primero debemos de cargar el paquete correspondiente en el cual se encuentran la función que nos permitirá leer los ficheros de datos. Para el caso de SPSS, debe cargar el paquete foreign. El cual es necesario para lectura y escritura de datos.

Para leer los datos se usa la siguiente función Read.spss("nombreArchivo", use.values.labels.=FALSE, max.value.label=Inf, to.data.frame=T); donde use.values.labels=TRUE significa que si en el archivo existen variables categóricas que han sido previamente codificadas con su respectiva etiqueta, entonces se leerán directamente las etiquetas y no los valores de esta (por ejemplo, si 1 representa Femenino, se leerá Femenino en lugar de 1). to.data.frame =T indica que los datos serán almacenados en un data.frame, muy recomendable para análisis estadístico. Puede consultar más ayuda de la función con la instrucción help(read.spss).

- Instalar el paquete foreign, con la siguiente instrucción

```
#install.packages(c("foreign"))
```

o desde el menú como en el caso de la instalación del paquete Foreing.

- Cargar el paquete con la siguiente instrucción:

```
library(foreign)
```

- Leer el contenido del archivo "demo.sav", con la instrucción:

```
library(haven)
demo <-read_sav("demo.sav")
demo
```

```
## # A tibble: 6,400 x 29
##      age      marital address income  inccat   car  carcat      ed employ retire
##    <dbl>    <dbl+lbl>   <dbl>  <dbl> <dbl+l> <dbl> <dbl+l> <dbl+l>  <dbl> <dbl+>
##  1    55 1 [Married]        12     72 3 [$50~  36.2 3 [Lux~ 1 [Did~     23 0 [No]
##  2    56 0 [Unmarrie~       29    153 4 [$75~  76.9 3 [Lux~ 1 [Did~     35 0 [No]
##  3    28 1 [Married]         9     28 2 [$25~  13.7 1 [Eco~ 3 [Som~      4 0 [No]
##  4    24 1 [Married]         4     26 2 [$25~  12.5 1 [Eco~ 4 [Col~      0 0 [No]
##  5    25 0 [Unmarrie~        2     23 1 [Und~  11.3 1 [Eco~ 2 [Hig~      5 0 [No]
##  6    45 1 [Married]         9     76 4 [$75~  37.2 3 [Lux~ 3 [Som~     13 0 [No]
##  7    42 0 [Unmarrie~       19     40 2 [$25~  19.8 2 [Sta~ 3 [Som~     10 0 [No]
##  8    35 0 [Unmarrie~       15     57 3 [$50~  28.2 2 [Sta~ 2 [Hig~      1 0 [No]
##  9    46 0 [Unmarrie~       26     24 1 [Und~  12.2 1 [Eco~ 1 [Did~     11 0 [No]
## 10    34 1 [Married]         0     89 4 [$75~  46.1 3 [Lux~ 3 [Som~     12 0 [No]
## # ... with 6,390 more rows, and 19 more variables: empcat <dbl+lbl>,
## #   jobsat <dbl+lbl>, gender <chr+lbl>, reside <dbl>, wireless <dbl+lbl>,
## #   multline <dbl+lbl>, voice <dbl+lbl>, pager <dbl+lbl>, internet <dbl+lbl>,
## #   callid <dbl+lbl>, callwait <dbl+lbl>, owntv <dbl+lbl>, ownvcr <dbl+lbl>,
## #   owncd <dbl+lbl>, ownpda <dbl+lbl>, ownpc <dbl+lbl>, ownfax <dbl+lbl>,
## #   news <dbl+lbl>, response <dbl+lbl>
## # i Use `print(n = ...)` to see more rows, and `colnames()` to see all variable names
```

- Realizar los análisis o cálculos correspondientes.