Guía de Métodos Estadísticos 2020

Pablo Cortes; Francisco Fernández

2020-01-23

Contents

Bienvenidos

La presente Guía tiene el objetivo de acompañarlos en los laboratorios de Métodos Estadísticos durante todo el presente semestre de este año. A través de esta guía tendrán acceso completo a toda la materia que veremos durante el semestre y adicionalmente podrán practicar todo lo que acá veamos.

Chapter 1

Software estadístico R

R es un programa estadístico *open source* de gran versatilidad que permite analizar una amplia gama de problemas cuantitativos. Si bien R incorpora un lenguaje de programación que puede ser extremadamente complejo, vale la pena familiarizarse con esa herramienta que puede llegar a ser muy útil en el futuro, tanto dentro como fuera de la academia.

1.1 Aspectos básicos de R

1. R distingue mayúsculas y minúsculas.
2. Para asignar contenido a un objeto usamos `<-`. Por ejemplo, `x <- 10` asigna a `x` el valor `10`. En lugar de `<-` también podemos usar `=`. 3.Para ver el contenido de un objeto simplemente escribimos su nombre.
3. Para obtener ayuda usamos el comando help. Por ejemplo, `help(mean)` para obtener ayuda sobre el comando mean que calcula la media.
4. El GUI o interfaz gráfica del R tiene dos partes principales: la consola y el script.

[poner foto acá]

La ***consola*** es el corazón de R, allí podemos pedirle cosas y es donde se nos entregan los resultados. También nos avisa de posibles errores (generalmente en color rojo). La consola es lo primero que observamos cuando abrimos el programa. Cuando la consola tiene el cursor `>` significa que le podemos dar comandos para ejecutar. Si es que tiene el símbolo `+` quiere decir que nos falta completar el comando anterior.

Un ***script*** corresponde a una hoja para escribir comandos. Nos sirve para escribir solo los comandos, y cuando seleccionamos y presionamos *<Ctrl + R>* (juntos) se ejecuta el comando que hemos escrito y los resultados se visualizaran

en la consola Dependiendo del sistema operatio utilizado, la combinación podría ser <Ctrl +Enter>. El script es práctico por que no solo podemos escribir comandos sino también notas personales. Las notas tienen que estar precedidas por el `#`.

1.2 R Studio

RStudio es una interfaz que permite acceder de manera sencilla a todas las funciones de R. Para utilizar RStudio se requiere haber instalado R previamente. La instalación de RStudio se puede realizar desde la página oficial del programa.

1.2.1 Conociendo a Rstudio

Una vez instalados R y RStudio procedemos a ejecutar el programa RStudio desde cualquiera de los iconos que genera y se mostrará la siguiente pantalla:

[Figura 2]

Esta pantalla está dividida en tres partes:

1. La ventana de la izquierda donde está el prompt “>” , llamada Consola, es el espacio de trabajo.

2. La ventana de la derecha se divide en dos:

- En la ventana superior derecha se encuentra el historial de objetos almacenados en memoria. Desde esta ventana también podemos:

a) Limpiar nuestro historial

b) Importar datos

c) Muestra los comandos y funciones implementadas de los informes con los que se han trabajado.

3. En la ventana inferior de la derecha RStudio muestra el directorio de trabajo, los gráficos que se van generando, paquetes para cargarlos e instalarlos directamente, ayuda y un visor HTML. Estas pestañas se irán describiendo a lo largo del documento

1.3 Trabajemos en RStudio

1.3.1 Objetos de Datos

R es un lenguaje orientado a trabajar con objetos de datos (numéricos, caracteres y lógicos), los cuales se guardan en la memoria activa del ordenador (consola). Dichos objetos pueden ser:

1. *Escalares*: Números reales que sirven para describir un fenómeno físico.
2. *Vectores*: Lista ordenada de elementos, principalmente números reales.
3. *Factores*: Es un tipo especial de vector que almacena datos de caracteres en forma de variables cualitativas.

Para familiarizarnos con la sintaxis de R, escribamos la frase "Metodos estadisticos". En la consola de RStudio debemos escribir lo siguiente y luego presionar <Ctrl + Enter>:

```
x <- "Metodos"
```

Ahora escribamos x para ver que sucede:

```
x
```

```
## [1] "Metodos"
```

Lo que acabamos de hacer es asignar la palabra Metodos al objeto `x` mediante el uso del operador `<-` Ahora, asignaremos la palabra estadisticos al objeto `y`:

```
y <- "estadisticos"
y
```

```
## [1] "estadisticos"
```

Por último, combinaremos las dos palabras y las almacenaremos en un nuevo objeto que llamaremos `frase`:

```
frase <- paste(x,y)
frase
```

```
## [1] "Metodos estadisticos"
```

IMPORTANTE: Los datos de caracteres siempre deben estar encerrados entre comillas ("")

1.3.2 Datos Numéricos

El objeto mas simple que podemos crear es aquel con contiene solamente un numero real. Para ello, asignamos un valor a un objeto mediante el uso del operador `<-`

Generaremos un objeto de nombre `x` con el valor `5`:

```
x<-5
x
```

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

R permite realizar un sin numero de operaciones algebraicas con nuestros objetos. Dichos operaciones incluyen la adición (+), sustracción (-), multiplicación (*), división (/) y potenciación (^).

```
a<-15
a
```

```
## [1] 15
```

```
b <- 2
b
```

```
## [1] 2
```

```
a+b
```

```
## [1] 17
```

```
b-a
```

```
## [1] -13
```

```
a-b
```

```
## [1] 13
```

```
(a+b)/a
```

```
## [1] 1.133333
```

```
(a*b)/(b^b)
```

```
## [1] 7.5
```

Otras funciones matemáticas de importancia para nuestro curso son: la raíz cuadrada (sqrt), función exponencial (exp) y función logarítmica natural (log).

```
sqrt(9)
```

```
## [1] 3
```

```
c <- 9
c
```

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

```
sqrt(c)
```

```
## [1] 3
```

```
exp(2)
```

```
## [1] 7.389056
```

```
log(10)
```

```
## [1] 2.302585
```

1.3.3 Vectores

Un vector representa una secuencia ordenada de elementos (datos) del mismo tipo. Es posible construir vectores de tipo numérico y caracteres. Para nuestros propósitos, los vectores podrán ser considerados como variables.

Generaremos un vector de nombre `vector1` que contenga tres datos numéricos:

```
vector1<-c(1,5,7)
vector1
```

```
## [1] 1 5 7
```

Ahora, generaremos un vector de nombre `vector2` que contenga tres caracteres:

```
vector2<-c("cerezo","peral","vid")
vector2
```

```
## [1] "cerezo" "peral"  "vid"
```

Intentemos generar un vector de nombre vector3 a partir de los vectores creados:

eyJsYW5ndWFnZSI6InIiLCJzYW1wbGUiOiJ2ZWN0b3IxPC1jKDEsNSw3KVxudmVjdG9yMjwtYyhI

IMPORTANTE: Los datos son concatenados (combinados) utilizando el comando `c()`

1.3.4 Funciones para generar vectores

Las funciones `seq` y `rep` nos permiten crear patrones de elementos. `seq` Crea una secuencia de números equiespaciados. Dentro del comando `seq` el comienzo (`from`), el fin (`to`), el espacio entre dos números consecutivos (`by`) o la cantidad de números en la secuencia (`length`) pueden ser especificados.

Por ejemplo:

```
seq(from= 2, to= 8, by=2)
```

```
## [1] 2 4 6 8
```

```
seq(from=2, to= 8,length=3)
```

```
## [1] 2 5 8
```

Por otra parte, el comando `rep` repite un elemento (`x`) una cantidad determinada de veces (`times`) o hasta lograr una longitud especificada (`length.out`).

Por ejemplo:

```
rep (0,5)
```

```
## [1] 0 0 0 0 0
```

```
rep(0:2,3)
```

```
## [1] 0 1 2 0 1 2 0 1 2
```

```
rep(1:2,length.out=7)
```

```
## [1] 1 2 1 2 1 2 1
```

You can write citations, too. For example, we are using the **bookdown** package (**?**) in this sample book, which was built on top of R Markdown and **knitr** (Xie, 2015).

Chapter 2

Tipos de datos en R

2.1 Objetos en R

Hasta ahora nos hemos dedicado a escribir algunas instrucciones para que R las ejecute. A lo largo del curso aprenderemos muchas funciones, sin embargo existen aspectos críticos que debemos saber antes de seguir avanzando.

Lo primero que debemos saber (y que nos evitará que surjan en la consola los nunca agradables errores) es que todos los elementos que maneja R son objetos: un valor numérico es un objeto, un vector es un objeto, una base de datos es un objeto, una función es un objeto, un gráfico es… un objeto.

En este laboratorio exploraremos algunos tipos de objetos y sus propiedades básicas para trabajar en R.

2.1.1 Propiedades de los objetos

1. Los objetos están compuestos por uno o más elementos. Los elementos pueden ser caracteres alfabéticos y/o numéricos. En este curso, los elementos serán considerados datos.

2. R guarda los objetos en la memoria activa del ordenador con un nombre especifico. Para ello, asignamos un valor a un objeto mediante el uso del operador `<-`.

Generemos un objeto de nombre Asignatura que contendrá las palabras *Metodos* y *estadisticos*. En el script de RStudio debemos escribir lo siguiente y luego presionar <Ctrl + enter>:

```
Asignatura<-c("Metodos", "estadisticos")
```

Ahora escribamos *asignatura* y veamos que sucede:

```
#asignatura
```

Es momento de felicitarnos a nosotros mismos, ya que acabamos de cometer uno de los errores más recurrentes y básicos que se cometen al trabajar en R. ¿Qué fue lo que sucedió?. R discrimina entre letras mayúsculas y minúsculas para el nombre de un objeto, por lo cual no es lo mismo escribir *asignatura* que *Asignatura*.

Ahora escribamos *Asignatura* y veamos que sucede:

```
Asignatura
```

```
## [1] "Metodos"      "estadisticos"
```

IMPORTANTE: Los datos de caracteres siempre deben estar encerrados entre comillas (`""`) y los datos son concatenados (combinados) utilizando el comando `c()`

2.2 Tipos de Objetos

En este curso nos ocuparemos de aquellos objetos que R utiliza para representar datos: valores, vectores, y dataframes.

2.2.1 Objetos de valores numéricos

El objeto mas simple que podemos crear es aquel con contiene solamente un numero real. Generaremos un objeto de nombre `x` con el valor `5`:

```
x<-5
x
```

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

R permite realizar un sin número de operaciones algebraicas con nuestros objetos. Dichos operaciones incluyen la adición (`+`), sustracción (`-`), multiplicación (`*`), división (`/`) y potenciación (`^`).

```
# crear objeto a
a<-15
a
```

```
## [1] 15
```

```
# crear objeto b
b<-2
b
```

```
## [1] 2
```

```
# sumar a + b
a+b
```

```
## [1] 17
```

```
# restar b-a
b-a
```

```
## [1] -13
```

```
# restar a - b
a-b
```

```
## [1] 13
```

```
# otras operaciones
(a+b)/a
```

```
## [1] 1.133333
```

```
(a*b)/(b^b)
```

```
## [1] 7.5
```

Otras funciones matemáticas de importancia para nuestro curso son: la raíz cuadrada (`sqrt`), función exponencial (`exp`) y función logarítmica natural (`log`).

```
# Raíz cuadrada
sqrt(9)
```

```
## [1] 3
```

```
# exponencial
exp(2)
```

```
## [1] 7.389056
```

```
# logaritmo
log(10)
```

```
## [1] 2.302585
```

2.2.2 Data frames

Un `data.frame` es un tipo especial de objeto que permite organizar diferentes tipos de vectores (alfanuméricos). La estructura de un `data.frame` es muy similar a una hoja de datos, en donde la información se organiza en filas (observaciones de cada vector) y columnas (vector).

El siguiente ejemplo nos muestra como crear un `data.frame` llamado `Notas` a partir de los datos obtenidos de una muestra a 8 alumnos, para cada una de las cuales se ha registrado su edad, género y nota obtenida en la primera prueba parcial del curso Métodos Estadísticos.

Primero, debemos generar los 3 vectores (variables) que utilizaremos para crear nuestro `data.frame`:

```
edad <- c(22, 21, 21, 25, 19, 22, 23,24)
genero <- c("M","F","F","M","M","F","F","M")
nota<-c(3.4,6.0,5.1,4.5,4.6,6.1,4.0,4.5)
```

A continuación, generaremos el `data.frame` llamado `Notas`. Es importante tener presente que el lenguaje de R es bastante intuitivo, a tal punto que la función que permite generar un data.frame es `data.frame ()`

```
Notas <- data.frame(edad,genero,nota)
```

Un fuerte aplauso para nosotros mismos, acabamos de generar nuestro primer `data.frame`. Para corroborar si lo hicimos bien, y al mismo tiempo ver su contenido, simplemente tenemos que llamarlo su nombre:

```
Notas
```

```
##   edad genero nota
## 1   22      M  3.4
## 2   21      F  6.0
## 3   21      F  5.1
## 4   25      M  4.5
## 5   19      M  4.6
## 6   22      F  6.1
## 7   23      F  4.0
## 8   24      M  4.5
```

En caso de que quisiéramos saber la estructura de nuestro data.frame, podemos utilizar la función: `str()`

```
str(Notas)
```

```
## 'data.frame':    8 obs. of  3 variables:
##  $ edad  : num  22 21 21 25 19 22 23 24
##  $ genero: Factor w/ 2 levels "F","M": 2 1 1 2 2 1 1 2
##  $ nota  : num  3.4 6 5.1 4.5 4.6 6.1 4 4.5
```

R nos acaba de confirmar que Notas es un data.frame de tres variables con 8 observaciones cada una. R también nos informa además de que dos variables son numéricas y la tercera, el genero, es un factor con dos valores, “F” y “M”.

En caso de que quisiéramos que R nos entregue el nombre de las variables contenidas en el Notas podemos utilizar la función `names()`:

```
names(Notas)
```

```
## [1] "edad"   "genero" "nota"
```

2.2.3 Valores Perdidos

En algunos casos los elementos que componen un objeto son desconocidos. En estos casos, debemos especificar que el elemento desconocido se encuentra “not available” (`NA`) , entonces a esa observación le asignamos el valor especial `NA`. Es importante tener presente que una operación con elementos `NA` resulta `NA`, ya que por defecto R no incluyen la especificación que omite o remueve las observaciones faltantes. Para estar seguros de esto, podemos generar un nuevo `data.frame` llamado `Notas1` a partir de los datos obtenidos de una muestra a 8 alumnos.

```
edad1 <- c(22, 21, 21, 25, 19, 22, 23,24)
genero1 <- c("M","F","F","M","M","F","F","M")
nota1<-c(3.4,6.0,5.1,4.5,4.6,6.1,4.0,NA)
Notas1 <- data.frame(edad1,genero1,nota1)
```

Apliquemos una operación básica a la información contenida en Notas1. Podemos estimar la nota promedio de los 8 alumnos, para ello utilizaremos la función `mean()`, la cual exploraremos más adelante en nuestro curso:

```
mean(Notas1$nota1)
```

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

Lo que obtuvimos como resultado fue `NA`. Esto se debe a que no le especificamos a R que debe omitir los NAs en nuestro data.frame. Una forma de solucionar este problema es utilizar la operación `na.rm=TRUE`, la cual especifica que la operación se efectué con los datos válidos excluyendo los NAs:

```
mean(Notas1$nota1, na.rm=TRUE)
```

```
## [1] 4.814286
```

Magia! Con una simple orden acabamos de solucionar un simple pero doloroso problema.

Bibliography

Xie, Y. (2015). *Dynamic Documents with R and knitr*. Chapman and Hall/CRC, Boca Raton, Florida, 2nd edition. ISBN 978-1498716963.