Cómo sobrevivir a una pelea con R

Ejercicios

Julio 2019

Introducción

En los ejercicios de la primera sesión vamos a trabajar sobre los diferentes objetos de `R` y su manipulación. Antes de empezar:

1. **Abre el script** `sesion1_ejercicios.R`. **Limpia** el espacio de datos ejecutando `rm(list = ls())`. Con este comando eliminarás del espacio de trabajo todos los datos (objetos) que estén disponibles evitando posibles confusiones[1].
2. **Carga los paquetes** que necesitas para realizar la práctica, ejecutando las líneas de `library()`. En caso de que alguno de ellos no esté instalado, instalaló utilizando `install.packages("package")`[2].

[1] Puedes usar el atajo `Ctrl + Enter` para ejecutar una línea de código en `RStudio`

[2] Usa comillas (" ") a la hora de instalar los paquetes con la función `install.packages()`

A. Tipos de objetos

A.1 Crea **dos vectores** `(x, y)`, ambos de tres elementos. El primero debe contener los valores `2, 4, 6` y el segundo `3, 4, 9`. Para ello utiliza la función `c()`. Calcula la **media de cada vector** y **almacena los resultados** como `mean_x` y `mean_y`. Después suma `mean_x` al vector `x`, ¿Cómo se ha producido la suma? Anota la respuesta en el script.

`c(...)` Devuelve un vector que combina los elementos pasados en la función.

`mean(x, na.rm = FALSE)` Devuelve la media aritmética del vector `x`.

A.2 Une los vectores `x` e `y` en un único vector, que será asignado a `z`. A partir del vector `z` **crea una matriz de dos columnas** y almacénala como `matriz_a`.

`matrix(data, nrow, ncol, byrow)` Devuelve un objeto de tipo `matrix`.

A.3 Crea un **data frame** (`df`) en el que la `var1` sea el vector `x`, la `var2` sea `y` y la `var3` sea `z`. Utiliza la función `data.frame()`. Imprime el df, ¿qué ha ocurrido con la `var_1` y la `var_2`?

`data.frame(..., stringsAsFactors = TRUE)` Devuelve un data frame con los elementos especificados en `...`.

A.4 Crea una **lista** (`storage`) que contenga todos los elementos que has creado hasta el momento: `x`, `y`, `mean_x`, `mean_y`, `z`, `matriz_a` y `df`. Utiliza la función `list()`. Al finalizar imprime el resultado.

`list(...)` Devuelve una lista con los elementos especificados en `...`.

B. Indexar objetos

B.1 A continuación tienes un vector con los `ingresos` de una serie de individuos. Utiliza la función `mean()` para **calcular la media de**

los ingresos de los hombre y las mujeres por separado. ¿Qué sistema has utilizado para seleccionar los subgrupos: posicional o por nombre? ¿Por qué?

```
ingresos <- c(Marcos = 1500, Marta = 200,
              Juan = 2050, Isabel = 300)
```

B.2 El data frame `mtcars` viene precargado en R[3]. Para saber cómo es el conjunto de datos `mtcars` ejecuta la siguiente línea de código, que te permitirá observar los 5 primeros casos:

[3] `R` y alguno de sus paquetes traen datos precargados con el fin de facilitar la replicabilidad de los ejemplos. Puedes consultar un listado de estos objetos con la función `data()`. Para acceder a los datos solo tienes que utilizar su nombre, por ejemplo, `mtcars`.

```
head(x = mtcars, n = 5)
```

`head(df, n = 6)` Devuelve un data frame con las primeras 6 observaciones por defecto.

	mpg	cyl	disp	hp	drat	wt
Mazda RX4	21.0	6	160	110	3.90	2.620
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875
Datsun 710	22.8	4	108	93	3.85	2.320
Hornet 4 Drive	21.4	6	258	110	3.08	3.215
Hornet Sportabout	18.7	8	360	175	3.15	3.440

El data frame `mtcars` cuenta con 11 variables y los nombres de las filas, a los que puedes acceder con la función `rownames(df)`. Realiza una **selección de columnas**, es decir, variables. Selecciona las variables `mpg`, `cyl` y `wt` y guarda el data frame resultante como `mtcars_small`. Posteriormente calcula la media de la variable `mpg`.

`rownames(df)` Devuelve un vector de tipo character con los nombres de las filas o su posición.

B.3 Del data frame `mtcars_small` calcula la media de la variable `mpg` para los 5 primeros casos **indexando de dos formas diferentes**.

B.4 Ahora selecciona una **muestra aleatoria de 7 coches** del conjunto de datos `mtcars_small`. Para seleccionar una muestra aleatoria puedes usar la función `sample()`. Esta función creará un vector con los elementos seleccionados. Por lo general, cuando se extrae una muestra de un data frame con `sample()`se utilizan los nombres de las filas como referente. A los nombres de las filas se accede con la función `rownames()`. Fíjate en el siguiente ejemplo:

`sample(x, size, replace, prob)` Devuelve un vector con los `n` elementos seleccionados en la muestra.

```
set.seed(123)
sample(x = rownames(mtcars_small), size =  2)
```

```
## [1] "Maserati Bora"      "Cadillac Fleetwood"
```

Ahora selecciona la muestra y guárdala como `mtcars_small_sample`.

C. Modificar vectores

Para **modificar vectores dentro de un data frame** se utiliza `$` para acceder a las variables o vectores, por ejemplo:

```
mtcars$new <- mtcars$mpg + mtcars$wt
```

La variable `mtcars$new` será creada dentro de `mtcars`, siendo la suma de `mpg` y `wt` para cada caso del data frame.

C.1 Otro conjunto de datos precargado en `R` es `gss_cat`. Utiliza la función `head()` para **explorar los datos**. Recodifica la variable `race` de forma que tenga dos categorías `white` y `Non-white`. Puedes usar la función `levels()` para explorar los niveles del factor y la función `recode()` para hacer la recodificación. Almacena la nueva variable dentro de `gss_cat` como `race_recode`. Usa la función `table()` para comprobar la recodificación.

`levels(x)` Devuelve un vector con los niveles de un factor `x`.

`recode(x, ..., .default, .missing)` Devuelve un vector recodificado.

`table(x, y)` Devuelve un objeto tipo `table`.

`fct_relevel(x, ...)` Devuelve un factor con las categorías reordenadas.

C.2 Haz un `table()` de la variable `marital`, como ves la respuesta `No answer` es la primera opción. **Reordena los niveles de la variable** `marital` de forma que las categorías `"Married"`, `"Divorced"` y `"Separated"` sean los tres primeros niveles y `"No answer"` el último. Sustituye la antigua versión de `marital` por la nueva.

C.3 Convierte el nivel **"No answer"** de la variable `marital` en `NA`. Determina el número de valores perdidos utilizando la función `is.na()` y `sum()`.

`is.na(x)` Devuelve un vector lógico.

`sum(x, na.rm = FALSE)` Devuelve la suma de los elementos de `x`.

C.4 A continuación tienes un vector de tipo character (`datos`). **Separa los nombres de la edad** y crea un data frame con dos variables: `nombre` y `edad`. Para conocer la longitud de las cadenas de texto puedes utilizar la función `str_length()`. Para crear el data frmae utiliza la función `cbind()` que sirve para combinar columnas. Guarda el nuevo objeto con el nombre `datos_edad_nombre`.

```
datos <- c("Juan Antonio, 28", "María Jesús, 29",
           "Pedro Sarmiento, 21", "Josefa Maura, +99")
```

`str_length(string)` Devuelve un vector con la longitud de la `string`.

`str_locate(string, pattern)` Devuelve una matriz con la posición inicial y final de la `pattern` en cada `string`.

`str_sub(string, start, end)` Devuelve un vector con las cadenas extraídas.

`cbind(x, y)` Devuelve un objeto que combina als columnas de `x` e `y`.

D. Transformar objetos

D.1 ¿Qué tipo de objeto es `datos_edad_nombre`? Conviértelo en un data frame y asignalo al mismo objeto. Imprime el objeto resultante.

`class(x)` Devuelve la denominación tipo de objeto.

D.2 Observa qué tipo de variables son `nombre` y `edad` del conjunto `datos_edad_nombre`. Convierte la variable `nombre` en un vector de

`as.data.frame(x)` Devuelve un objeto tipo data frame.

tipo character y la variable `edad` en numérica. Al transformar las variables utiliza los nombres `edad_numeric` y `nombre_character` ¿Qué ha ocurrido con la variable `edad`? ¿Y con `nombre`?

`str(x)` Devuelve la estructura de `x`.

`as.character(x)` Devuelve un objeto tipo character.

`as.numeric(x)` Devuelve un objeto tipo numeric.

D.3 Una posible solución al problema con `edad_numeric` es hacer una doble conversión con el fin de que los niveles de `edad` sean leídos en vez de los valores de la variable. Crea una nueva versión de la variable `edad` que se llame `edad_numeric2`. Para ello, en primer lugar, convierte el vector en charater para posteriormente convertirlo en numeric.