vectores, matrices, data frames & listas

Armando Ocampo

Vectores

Los vectores son representaciones lineales de la información. Es decir, solo tienen una dimensión. Los vectores pueden tener elementos de tipo caracter, numericos continuos, numericos enteros, fechas, o una mezcla de varios elementos. Para unir varios elemntos utilizamos la funcion c().

Nota: si no saber utilizar una función, recuerda que puedes buscarla en internet o con los siguientes comandos ?c help(c)

```
# vector tipo caracter
c("a", "b", "c", "d")
# vector numerico continuo <-
c(1.2,1.3,1.4,1.5)
# vector numerico entero
c(1,2,3,4,5,6,7)
# vector mixto
c("a", "b",1, "c",1.3,5)
```

Nota 1: los elementos de tipo caracter van entre comillas, los vectores de tipo numerico se escriben sin comillas. Nota 2: puedes o no colocar un espacio entre cada elemento, el agregar un espacio no afecta el vector

Hasta este momento nuestros vectores solo se han proyectado en la consola. Para guardarlos en el ambiente utilizamos "<-" o "=". Recuerda que el atajo para asignar es "alt" + "-". Vamos a crear un vector con numeros del 1 al 5. Llamaremos a este vector *numeros*

```
numeros <- c(1,2,3,4,5)
```

Este vector con una secuencia numerica corta es fácil de realizar de forma manual, pero ¿qué sucede si nos piden realizar un vector con numeros del 1 al 100? Para esto tenemos un atajo, los dos puntos ":" permiten hacer secuencias solo necesitamos el valor minimo y el valos maximo

```
c(1:100)
```

De hecho, podemos hacer secuencias con diferente rango, seprando cada secuencia con un coma.

```
c(1:5,15:40,50:100)
```

Nota: las comas sirven para separar argumentos, solo debes tener cuidado que no omitas o pongas una coma extra cuando haces tu codigo.

Los vectores numericos tambien pueden utilizarse en expresiones matemáticas. Si sumamos un numero a un vector, el numero se sumará a cada uno de los elementos del vector, generando un nuevo vector con el producto de la expresión matemática.

```r
# sumando 5 a cada elemnto del vector numeros
numeros + 5
# restando 15 a cada elemento del vector numeros
numeros - 15
# multiplicando por 10
numeros * 10
```

Este resultado solo se observa en la consola, podemos guardarlo asignandolo a un objeto

```r
vectorcito <- numeros + 5
```

Tambien podemos seleccionar un elemento especifico de cada vector a partir del uso de []. Para seleccionar el tercer elemento del vector numeros se realiza lo siguiente

```r
numeros[3]
# seleccion del cuarto elemento del vector vectorcito
vectorcito[4]
# si queremos conocer dos elementos utilizamos la funcion concatenar c() dentro de los
# corchetes. Por ejemplo, seleccionaremos los elementos tres, cuatro y cinco  del vector
#vectorcito
vectorcito[c(3,4,5)]
# e incluso podemos realizar comparaciones y saber que elemento cumple cada condicon

# en el siguiente ejemplo conoceremos los elementos que son mayor que 8 en el vector
#vectorcito
vectorcito > 8
# de esta forma podemos seleccionar los elementos que cumplen la condicion
vectorcito[c(4,5)]
```

tarea: realiza un vector de tipo caracter con 5 nombres y guardalo con el nombre personas, re aliza un vector con numeros del 5 al 35, realiza un vector de tipo caracter con 5 animales.

Matrices

Las matrices son elementos rectangulares en dos dimensiones que almacenan informacion de un solo tipo. Para crear una matriz se necesita la funcion matrix(). Si no sabes utilizar esta funcion utiliza el comando help(matrix) o ?matrix

```r
matrix(data = (1:9), nrow = 3, byrow = TRUE)

matrix(data = (1:9), nrow = 3, byrow = FALSE)
```

Esta informacion se puede almacenar como un objeto

```r
mi_primer_matriz <- matrix(data = (1:9), nrow = 3, byrow = TRUE)
```

Las matrices tambien pueden tener una evaluacion a partir de funciones aritméticas. Cuando sumamos una matriz con un numero, este se sumará a cada elemento de la matriz

```r
mi_primer_matriz + 5
# tambien podemos hacer restas
mi_primer_matriz - 20
# multiplicaciones
mi_primer_matriz * 50
```

Al igual que los vectores, podemos seleccionar un elemento en especifico dentro de la matriz a partir del uso de corchetes []. Recordando que las matrices son elementos en dos dimensiones, necesitamos colocar dos numeros dentro de los corchetes. El primero corresponde a las filas y el segundo a las columnas. A continuacion seleccionaresmos el elemento de la fila 2 de la columna 3 del elemento mi_primer_matriz

```r
mi_primer_matriz[2,3]
```

De la misma forma, podemos hacer compraciones. Evaluaremos que elementos de mi_primer_matriz son mayores a 6

```r
mi_primer_matriz > 6
# en este caso vemos que toda la fila 3 cumple con esta condicion, para seleccionar una
#fila completa colocamos la fila en el corchete y el segundo elemnto lo dejamos en blanco
mi_primer_matriz[3,]
# con las columnas se hace algo similar, solo que el primer elemento se deja en blanco y el
# segundo numero es la columna que seleccionaresmos. A continuacion seleccionaremos los
# elementos de la columna 2
mi_primer_matriz[,2]
# No obstante, tambien podemos colocar la condicion dentro de los corchetes y saber
# los elementos que se cumple la condicion
mi_primer_matriz[mi_primer_matriz > 6]
# podemos hacer la condicion como queramos
mi_primer_matriz[mi_primer_matriz == 8]
mi_primer_matriz[mi_primer_matriz > 4]
# queremos obtener los elementos mayores a 4 y menores a 8
mi_primer_matriz[mi_primer_matriz > 4 & mi_primer_matriz < 8]
```

Data frames

Los data frames son elementos en dos dimensiones que pueden guardar objetos de cualquier naturaleza. Es decir, elementos numericos enteros, continuos, caracter, etc. Existen dataframes precargados en R para hacer pruebas estadísitcas, como mtcars e iris

```r
mtcars
?mtcars
iris
?iris
```

Tambien podemos construir un data frame a partir de vectores. Realizaremos tres vectores con nombres, edades, color favorito y numero de mascotas

```r
nombres <- c("juan", "pedro", "luis")
edad <- c(20,35,40)
color <- c("rojo", "azul", "verde")
mascotas <- c(1,2,1)
```

```r
# el dataframe o df se construye con la funcion data.frame(), dentro de la funcion
# se coloan los vectores que conformarán el df
# nombraremos a nuestro df como mi_primer_df
mi_primer_df <- data.frame(nombres,edad,color,mascotas)
mi_primer_df
```

Al tener elementos de diferente naturaleza a veces es complicado evaluar columna por columna para saber la naturaleza de nuestras variables. En este caso te recomiendo evaluar de forma rapida tu data set con las siguientes funciones

```r
str(mi_primer_df)
summary(mi_primer_df)
```

Esto se puede realizar con cualquier elemento, no solo con los df

```r
str(mi_primer_matriz)
summary(mi_primer_matriz)
str(color)
summary(color)
str(iris)
summary(iris)
```

Al tener elementos de diferente naturaleza no podemos realizar operaciones matematicas de forma directa. Sin embargo podemos seleccionar una columna del df utilizando "$"

```r
# conocer los nombres de las columnas de un df
names(mi_primer_df)
# seleccionar la columna edad
mi_primer_df$edad
# multiplicar el vector de edad por 10
mi_primer_df$edad * 10

# esto se puede hacer con cualquier df
names(iris)
names(mtcars)
# y seleccionar cualquier columna de la misma manera
iris$Sepal.Length
# o gurdar el contenido de la columna en un nuevo objeto
vector_iris <- iris$Petal.Width
vector_iris
```

Y si tiene las mismas dimensiones puedes agregar un nuevo vector al df a partir de la funcion rbind(), el nombre viene de row bind o pegar fila

```r
nuevo_elemento <- c("hugo", 40, "gris", 0)
df_modificado <- rbind(mi_primer_df, nuevo_elemento)
```

o pegar una nueva columna con la funcion cbind() de column bind o pegar columna

```r
# agregaremos una columna de área de estudio a df_modificado y lo nombraremos df_completo
carrera <- c("ciencias", "biologia", "matematicas", "medicina")
df_completo <- cbind(df_modificado, carrera)
df_completo
```

Listas

las listas son elementos que permiten guardar objetos de diferente naturaleza incluyendo df, matrices, vectores, objetos JSON, etc. Las listas se generan con las funcion list(), dentro de la funcion se colocan los elementos que se van a guaradar en la lista

```
mi_primer_lista <- list(df_completo, df_modificado, carrera, color, mascotas, mi_primer_matriz)
mi_primer_lista
```

De esta forma podemos almacenar de forma eficiente nuestros elementos. Para seleccionar los elementos de la lista se utilizan los [] y los [[]]