Ejercicios Listas

David Criado Ramón

10/11/2019

Introducción a R (5)

1. Lists

Las listas son colecciones de objetos que pueden tener modos diferentes (e.g. numéricos, vectores, arrays..)

```
my_list <- list(name="Fred", wife="Mary", no.children=3, child.ages=c(4,7,9))
```

- Imprime los atributos y los nombres de todos los componentes de la lista.

```
attributes(my_list)
```

```
## $names
## [1] "name"        "wife"        "no.children" "child.ages"
```

```
my_list
```

```
## $name
## [1] "Fred"
##
## $wife
## [1] "Mary"
##
## $no.children
## [1] 3
##
## $child.ages
## [1] 4 7 9
```

- Devuelve el Segundo componente de la lista. El operador [[..]] devuelve el objeto almacenano en ese componente.

```
my_list[[2]]
```

```
## [1] "Mary"
```

- ¿Que pasa si usamos '[..]'?

```
my_list[2]
```

```
## $wife
## [1] "Mary"
```

Podemos observar tanto el nombre del atributo (wife) como el valor asociado.

- Llama al componente_list[[2]] por su nombre.

```
my_list$wife
```

```
## [1] "Mary"
```

- Recupera el Segundo element del cuarto componente de la lista

```
my_list[[4]][2]
```

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

- Imprime la longitud, del cuarto elemento de la lista.

```
length(my_list[[4]])
```

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

- Reemplaza su contenido por un vector de 12 numeros del 1 al 12

```
my_list[[4]] <- 1:12
```

- Elimina el componente wife

```
my_list$wife <- NULL
```

- Añade un element más a la lista.

```
my_list$otro_elemento <- 5
```

- Convertir lista a data frame o matriz.

```
unlist(my_list);
```

```
##          name   no.children   child.ages1   child.ages2   child.ages3
##        "Fred"           "3"           "1"           "2"           "3"
##   child.ages4   child.ages5   child.ages6   child.ages7   child.ages8
##           "4"           "5"           "6"           "7"           "8"
##   child.ages9  child.ages10  child.ages11  child.ages12 otro_elemento
##           "9"          "10"          "11"          "12"           "5"
```

```
data.frame(unlist(my_list));
```

```
##               unlist.my_list.
## name                     Fred
## no.children                 3
## child.ages1                 1
## child.ages2                 2
## child.ages3                 3
## child.ages4                 4
## child.ages5                 5
## child.ages6                 6
## child.ages7                 7
## child.ages8                 8
## child.ages9                 9
## child.ages10               10
## child.ages11               11
## child.ages12               12
## otro_elemento               5
```

```
matrix(unlist(my_list));
```

```
##       [,1]
##  [1,] "Fred"
##  [2,] "3"
##  [3,] "1"
##  [4,] "2"
##  [5,] "3"
```

```
##  [6,] "4"
##  [7,] "5"
##  [8,] "6"
##  [9,] "7"
## [10,] "8"
## [11,] "9"
## [12,] "10"
## [13,] "11"
## [14,] "12"
## [15,] "5"
```

2. table()

Vamos a trabajar con el dataset iris. La función table() cuenta el numero de elementos repetidos en un vector. Es la función más básica de clustering. Cuenta el número de entradas idénticas en la variable Sepal.Length del dataset iris.

```
table(iris$Sepal.Length, iris$Species)
```

```
##
##       setosa versicolor virginica
##   4.3      1          0         0
##   4.4      3          0         0
##   4.5      1          0         0
##   4.6      4          0         0
##   4.7      2          0         0
##   4.8      5          0         0
##   4.9      4          1         1
##   5        8          2         0
##   5.1      8          1         0
##   5.2      3          1         0
##   5.3      1          0         0
##   5.4      5          1         0
##   5.5      2          5         0
##   5.6      0          5         1
##   5.7      2          5         1
##   5.8      1          3         3
##   5.9      0          2         1
##   6        0          4         2
##   6.1      0          4         2
##   6.2      0          2         2
##   6.3      0          3         6
##   6.4      0          2         5
##   6.5      0          1         4
##   6.6      0          2         0
##   6.7      0          3         5
##   6.8      0          1         2
##   6.9      0          1         3
##   7        0          1         0
##   7.1      0          0         1
##   7.2      0          0         3
##   7.3      0          0         1
##   7.4      0          0         1
##   7.6      0          0         1
```

```
##   7.7      0          0         4
##   7.9      0          0         1
```

3. Como ordenar datos, hacer selecciones con if, calcular condicionales totales, transponer columnas y filas

Vamos a utilizar el datasets mtcars.

- Ordena este data set de forma ascendente según su valor de hp.

```
mtcars %>% arrange(hp)
```

```
##     mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## 1  30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## 2  24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## 3  33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## 4  32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## 5  27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## 6  26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## 7  22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## 8  22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## 9  21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## 10 18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## 11 21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
## 12 21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## 13 21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## 14 21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## 15 30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## 16 19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## 17 17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## 18 15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## 19 15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## 20 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## 21 19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## 22 19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## 23 16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## 24 17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## 25 15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## 26 10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## 27 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
## 28 14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## 29 14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## 30 13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## 31 15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## 32 15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
```

- Hazlo ahora de forma descendente

```
mtcars %>% arrange(desc(hp))
```

```
##     mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## 1  15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## 2  15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## 3  14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## 4  13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## 5  14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
```

```
## 6  10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
## 7  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## 8  16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## 9  17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## 10 15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## 11 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## 12 19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## 13 19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## 14 15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## 15 15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## 16 19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## 17 17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## 18 30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## 19 21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## 20 21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## 21 21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## 22 21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
## 23 18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## 24 21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## 25 22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## 26 22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## 27 26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## 28 32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## 29 27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## 30 33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## 31 24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## 32 30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
```

- Calcula la media de la columna mpg.

```
mean(mtcars$mpg, na.rm=T)
```

```
## [1] 20.09062
```

- Calcula la media de mpg para aquellos datos cuyo valor de hp sea menor que 150 y por separado para aquellos cuyo valor de hp sea mayor o igual a 150

```
mean(subset(mtcars, mtcars$hp < 150)$mpg, na.rm=T)
```

```
## [1] 24.22353
```

```
mean(subset(mtcars, mtcars$hp >= 150)$mpg, na.rm=T)
```

```
## [1] 15.40667
```

- Busca los valores únicos de la columna cyl de mtcars. PISTA: unique()

```
unique(mtcars$cyl)
```

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

- Obtén los datos de mpg cyl disp hp para "Toyota Corolla"

```
mtcars %>% filter(rownames(.) == "Toyota Corolla") %>%
           select(mpg, cyl, disp, hp)
```

```
##    mpg cyl disp hp
## 1 33.9   4 71.1 65
```

- Crea una nueva variable mpgClass de tipo categórico cuyo valor es “Low“ si el valor de mpg es menor que la media de la columna mpg y “High” si es mayor que la media de mpg.

```
mtcars %>% mutate(mpgClass = factor(ifelse(mpg < mean(mpg), "Low", "High")))
```

```
##     mpg cyl  disp  hp drat    wt  qsec vs am gear carb mpgClass
## 1  21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4     High
## 2  21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4     High
## 3  22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1     High
## 4  21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1     High
## 5  18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2      Low
## 6  18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1      Low
## 7  14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4      Low
## 8  24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2     High
## 9  22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2     High
## 10 19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4      Low
## 11 17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4      Low
## 12 16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3      Low
## 13 17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3      Low
## 14 15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3      Low
## 15 10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4      Low
## 16 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4      Low
## 17 14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4      Low
## 18 32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1     High
## 19 30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2     High
## 20 33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1     High
## 21 21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1     High
## 22 15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2      Low
## 23 15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2      Low
## 24 13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4      Low
## 25 19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2      Low
## 26 27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1     High
## 27 26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2     High
## 28 30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2     High
## 29 15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4      Low
## 30 19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6      Low
## 31 15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8      Low
## 32 21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2     High
```