Sección 5 - Funciones loops

Ana Cecilia Plavan, Andrés Andruskevicius

Funciones para loops

En R existen muchas funciones manipular y trabajar sobre data frames y vectores:

- Familia Apply:
  - apply
  - sapply
  - tapply
  - mapply
  - lapply

Familia apply

- La familia de funciones apply es usada para aplicar una función a cada elemento de una estructura de datos.
- En particular, es usada para aplicar funciones en matrices, data frames, arrays y listas.

Función apply()

- apply aplica una función a todos los elementos de un data frame.

```
apply(X=dataframe, MARGIN, FUN)
```

  - MARGIN: Dimensión, bloque de elementos a los que se aplicara la función.
    - 1:Filas, 2:Columnas
  - FUN: Función a aplicar

Ejemplo apply()

```
> df <- data.frame(x = 1:4, y = 5:8, z = 9:12)
```

X	Y	z
1	5	9
2	6	10
3	7	11
4	8	12

```
> apply(df,2,max)
 x  y  z
 4  8 12
```

2:Columnas

1:Filas

```
> apply(df,1,max)
[1]  9 10 11 12
```

```
> apply(df,
c(2,1),
function(x){x^2})
   [,1][,2][,3] [,4]
x   1    4    9   16
y   25   36   49   64
z   81   100 121 144
```

Función sapply()

- Se diferencia de apply en que esta opera directamente con los vectores

```
> sapply(df, mean)
   x    y    z
 2.5  6.5 10.5
```

- Obtenemos el mismo resultado realizando:

```
> apply(df,2,mean)
   x    y    z
 2.5  6.5 10.5
```

Funcion tapply()

Aplica una función sobre un vector

- vector - parámetro 1
- Agrupado por factores - parámetro 2
- función - parámetro 3

```
> tapply(murders$total, murders$region, sum)
  Northeast South North Central West
       1469     4195      1828      1911
```

Otas funciones para loops

En R existen muchas otras funciones para realizar loops con tareas específicas como ser:

- split
- cut
- quantile
- reduce
- identical
- Unique
- group_by

Split()

- La función split divide los datos de entrada (x) en diferentes grupos (f)

```
regiones <- split(murders,f=murders$region)
```

```
$West
        state abb region population total
2      Alaska  AK   West     710231    19
3     Arizona  AZ   West    6392017   232
5  California  CA   West   37253956  1257
6    Colorado  CO   West    5029196    65
12     Hawaii  HI   West    1360301     7
13      Idaho  ID   West    1567582    12
27    Montana  MT   West     989415    12
29     Nevada  NV   West    2700551    84
32 New Mexico  NM   West    2059179    67
38     Oregon  OR   West    3831074    36
45       Utah  UT   West    2763885    22
48 Washington  WA   West    6724540    93
51    Wyoming  WY   West     563626     5
```

```
$`North Central`
         state abb         region population total
14     Illinois  IL North Central   12830632   364
15      Indiana  IN North Central    6483802   142
16         Iowa  IA North Central    3046355    21
17       Kansas  KS North Central    2853118    63
23     Michigan  MI North Central    9883640   413
24    Minnesota  MN North Central    5303925    53
26
28
35
36
42
50
```

```
$Northeast
          state abb    region population total
7   Connecticut  CT Northeast    3574097    97
20        Maine  ME Northeast    1328361    11
22 Massachusetts  MA Northeast    6547629   118
30 New Hampshire  NH Northeast    1316470     5
31    New Jersey  NJ Northeast    8791894   246
33      New York  NY Northeast   19378102   517
39  Pennsylvania  PA Northeast   12702379   457
40  Rhode Island  RI Northeast    1052567    16
46       Vermont  VT Northeast     625741     2
```

```
$South
                state abb region population total
1             Alabama  AL  South    4779736   135
4            Arkansas  AR  South    2915918    93
8            Delaware  DE  South     897934    38
9  District of Columbia  DC  South     601723    99
10            Florida  FL  South   19687653   669
11            Georgia  GA  South    9920000   376
18           Kentucky  KY  South    4339367   116
19          Louisiana  LA  South    4533372   351
21           Maryland  MD  South    5773552   293
25        Mississippi  MS  South    2967297   120
34     North Carolina  NC  South    9535483   286
37           Oklahoma  OK  South    3751351   111
41     South Carolina  SC  South    4625364   207
43          Tennessee  TN  South    6346105   219
44              Texas  TX  South   25145561   805
47           Virginia  VA  South    8001024   250
49      West Virginia  WV  South    1852994    27
```

Grupos obtenidos

- Accedemos a los grupos del data set original

```
> regiones$Northeast
           state abb    region population total
7    Connecticut  CT Northeast    3574097    97
20         Maine  ME Northeast    1328361    11
22 Massachusetts  MA Northeast    6547629   118
30 New Hampshire  NH Northeast    1316470     5
31    New Jersey  NJ Northeast    8791894   246
33      New York  NY Northeast   19378102   517
39  Pennsylvania  PA Northeast   12702379   457
40  Rhode Island  RI Northeast    1052567    16
46       Vermont  VT Northeast     625741     2
```

- Accedemos a los vectores del grupo

```
> regiones$Northeast$population
[1]  3574097  1328361  6547629  1316470  8791894
 19378102 12702379  1052567   625741
```

Función cut()

- Categoriza los valores de una variable continua en diferentes niveles de un factor.

```
cut(x, breaks, labels, right=TRUE)
```

- x = data set
- breaks = Vector de cortes
- labels = Las etiquetas que tomarán los cortes.
- right = El tipo de intervalos que se usará, por defecto es TRUE, cerrado a la derecha.

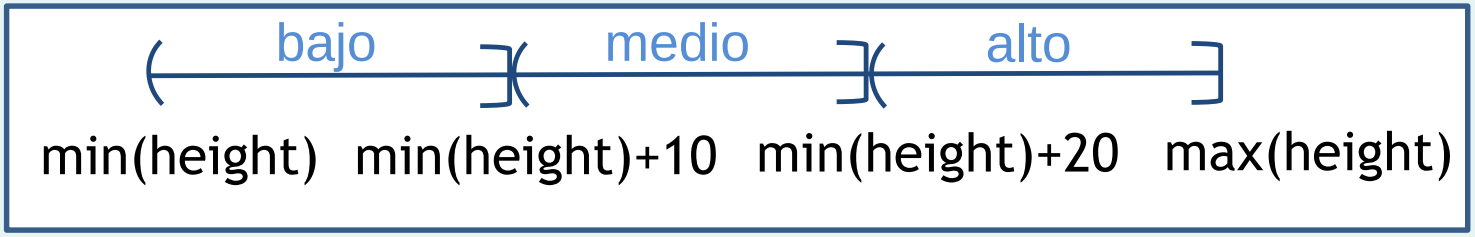
Incluyendo por defecto el valor mínimo de cada corte.

Ejemplo cut()

Categorizamos las alturas en 3 niveles: bajo, medio, alto

```
> head(heights)
     sex height
1   Male     75
2   Male     70
3   Male     68
4   Male     74
5   Male     61
6 Female     65
```

```
> head(heights)
     sex height EstAltura
1   Male     75      alto
2   Male     70     medio
3   Male     68     medio
4   Male     74      alto
5   Male     61     medio
6 Female     65     medio
```




```
> alturas<-heights$height
> categorias<-cut(alturas,breaks =
c(min(alturas),min(alturas)+10,min(alturas)+20,
max(alturas)),labels=c("bajo","medio","alto"))
> heights<-mutate(heights,EstAltura=categorias)
```

Función unique()

- Elimina valores duplicados en un vector

```
unique(x)
```

- x = data set

```
> x<-c(1,2,3,2,4,5,1,6,8,9,8,6)

> unique(x)
[1] 1 2 3 4 5 6 8 9
```

Función separate()

- Divide o Separa una cadena de caracteres con un patrón regular.

```
separate(x, col, into, sep)
```

- x = data set
- col = Nombre de columna o vector a dividir
- Into = Nombre de columnas a generar
- sep = separador o símbolo divisor

Ejemplo separate()

```
estudiantes<-
data.frame("nombres"=c("Ana","Juan","Pepe"),"ejercicio"=c("Nada
rapido","Corre lento","Bicy medio"))
```

	nombres	ejercicio
1	Ana	Nada rapido
2	Juan	Corre lento
3	Pepe	Bicy medio

```
estudiantes<-
estudiantes%>%separate(ejercicio,
c("dicipina","velocidad"))
```

	nombres	dicipina	velocidad
1	Ana	Nada	rapido
2	Juan	Corre	lento
3	Pepe	Bicy	medio

names - rename

- names(): Lista los nombres de las columnas de un data frame

```
> names(murders)
[1] "state"      "abb"        "region"     "population"
[5] "total"      "rate"
```

- rename(): Cambia los nombres de las variables del data frame.

```
> rename(murders,estado=state,poblacion=population,indice=rate)
                estado abb        region poblacion total
1          Connecticut  CT     Northeast   3574097    97
2                Maine  ME     Northeast   1328361    11
3        Massachusetts  MA     Northeast   6547629   118
4        New Hampshire  NH     Northeast   1316470     5
5           New Jersey  NJ     Northeast   8791894   246
6             New York  NY     Northeast  19378102   517
```

Función arrange

- Reordena las filas de un data frame según una o más columnas en orden ascendente o descendente.

```
arrange(x, ord): dplyr
```

  - x: Data frame a ordenar
  - Ord: valores de columnas seleccionadas. Para ordenar en orden descendente se debe de utilizar el parámetro desc:

```
arrange(x, desc(ord))
```

- Recordar que sort() ordenaba vectores o factores

Ejemplo arrange

- Ordenamos el data set de murders alfabéticamente por región y luego por nombre de estado

```
> murders<-murders%>%arrange(region,state)
```

	state	abb	region	population	total	rate
1	Connecticut	CT	Northeast	3574097	97	2.7139722
2	Maine	ME	Northeast	1328361	11	0.8280881
3	Massachusetts	MA	Northeast	6547629	118	1.8021791
4	New Hampshire	NH	Northeast	1316470	5	0.3798036
5	New Jersey	NJ	Northeast	8791894	246	2.7980319
6	New York	NY	Northeast	19378102	517	2.6679599
7	Pennsylvania	PA	Northeast	12702379	457	3.5977513
8	Rhode Island	RI	Northeast	1052567	16	1.5200933
9	Vermont	VT	Northeast	625741	2	0.3196211
10	Alabama	AL	South	4779736	135	2.8244238

Imprimir n valores

- En el caso de grandes data sets la función head(x,n) imprime las primeras n filas y sus respectivos nombres
- tail(x,n) imprime las últimas n filas y sus respectivos nombres
- Para imprimir y filtrar el contenido de acuerdo a otra cantidad o valor utilizamos top_n

```
top_n(x, n, wt)
```

  - x: DataFrame
  - n: cantidad de filas a mostrar
  - wt: columna a filtrar. Este argumento es opcional en caso de no ingresarlo se filtra de acuerdo a la ultima columna del data frame

Ejemplo:

head
vs.
top_n

```
> notas
    alumnos calificacion edades
1       Ana            D       9
2     Pablo            B       8
3     Pedro          MBS       9
4 Marianela          MBS       9
5      <NA>            B       7
6     Maria         <NA>      NA
7      Pepe         <NA>      NA
> head(notas)
    alumnos calificacion edades
1       Ana            D       9
2     Pablo            B       8
3     Pedro          MBS       9
4 Marianela          MBS       9
5      <NA>            B       7
6     Maria         <NA>      NA
> top_n(notas,6,alumnos)
    alumnos calificacion edades
1       Ana            D       9
2     Pablo            B       8
3     Pedro          MBS       9
4 Marianela          MBS       9
5     Maria         <NA>      NA
6      Pepe         <NA>      NA
```

Imprimir un resumen estadístico de los datos

- Visualizar los mínimos, máximos, promedios, valores medios y cuartiles de cada columna:

```
summary(x)
```

x: DataFrame

```
> summary(murders)
    state                abb
 Length:51          Length:51
 Class :character   Class :character
 Mode  :character   Mode  :character



           region     population
 Northeast     : 9   Min.    :  563626
 South         :17   1st Qu.: 1696962
 North Central:12   Median : 4339367
 West          :13   Mean   : 6075769
                     3rd Qu.: 6636084
                     Max.   :37253956

     total
 Min.    :   2.0
 1st Qu.:  24.5
 Median :  97.0
 Mean   : 184.4
 3rd Qu.: 268.0
 Max.   :1257.0
```

Crear un subset de valores

- En el caso que queramos seleccionar datos y filtrarlos por una condición lógica:

```
subset(x, subset, select)
```

  - x: DataFrame
  - subset: Condición lógica para realizar fitro de datos
  - select: Columnas del data frame a seleccionar o mostrar

Ejemplo subset()

```
> subset(murders , region=="South", c(state,region))
```

	state	region
1	Alabama	South
4	Arkansas	South
8	Delaware	South
9	District of Columbia	South
10	Florida	South
11	Georgia	South
18	Kentucky	South
19	Louisiana	South
21	Maryland	South
25	Mississippi	South
34	North Carolina	South
37	Oklahoma	South
41	South Carolina	South
43	Tennessee	South
44	Texas	South
47	Virginia	South
49	West Virginia	South