

DATA FRAME

Ramon Ceballos

22/1/2021

1. Funciones en Data Frames

La mejor manera de aplicar una función a todas las columnas de un data frame es utilizar las funciones `apply()`, en este caso `sapply()`.

- `sapply(d.f, función)`: para aplicar una función a todas las columnas de un data frame en un solo paso
 - `na.rm=TRUE`: para evitar que el valor que devuelva la función para las columnas que contengan algún NA sea NA

```
#Función sapply() en DF iris para las columnas numéricas  
str(iris)
```

```
## 'data.frame':   150 obs. of  5 variables:  
## $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...  
## $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...  
## $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...  
## $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...  
## $ Species      : Factor w/ 3 levels "setosa","versicolor",...: 1 1 1 1 1 1 1 1 1 1 ...
```

```
#Selecciono las columnas de la 1 a la 4 con subset  
#aplico la función sapply() con la media  
sapply(subset(iris, select=1:4), mean)
```

```
## Sepal.Length Sepal.Width Petal.Length Petal.Width  
##      5.843333      3.057333      3.758000      1.199333
```

```
#Hacer la suma  
sapply(iris[, 1:4], sum)
```

```
## Sepal.Length Sepal.Width Petal.Length Petal.Width  
##      876.5      458.6      563.7      179.9
```

```
f = function(x){sqrt(sum(x^2))}  
sapply(iris[, 1:4], f)
```

```
## Sepal.Length Sepal.Width Petal.Length Petal.Width  
##      72.27621      37.82063      50.82037      17.38764
```

```
#Crear data frame
df = data.frame(C1 = c(1,2,NA,4), C2 = c(5,NA,2,3))
#no hacemos nada con los NA
sapply(df, mean)
```

```
## C1 C2
## NA NA
```

```
#eliminamos los NA
sapply(df, mean, na.rm = TRUE)
```

```
##      C1      C2
## 2.333333 3.333333
```

Para aplicar una función a variables de un data frame que están clasificadas según uno o más factores se utiliza **aggregate()**.

- **aggregate(variables~factors,data=d.f,FUN=función)**: para aplicar una función a variables de un data frame clasificadas por los niveles de un, o más de un, factor
 - Si queremos aplicar la función a más de una variable, tenemos que agruparlas con un **cbind**
 - Si queremos separar las variables mediante más de un factor, tenemos que agruparlos con signos **+**.

```
#Obtener la media de la longitud de sepalos y de petalos para cada especie de flor
aggregate(cbind(Sepal.Length, Petal.Length) ~ Species, data = iris, FUN = mean, na.rm = TRUE)
```

```
##      Species Sepal.Length Petal.Length
## 1      setosa      5.006      1.462
## 2 versicolor      5.936      4.260
## 3 virginica      6.588      5.552
```

```
#Empleo el Data Frame de mtcars
head(mtcars)
```

```
##      mpg cyl disp  hp drat   wt  qsec vs am gear carb
## Mazda RX4      21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag  21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710      22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive  21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant        18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

```
str(mtcars)
```

```
## 'data.frame':   32 obs. of  11 variables:
## $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
## $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
## $ disp: num  160 160 108 258 360 ...
## $ hp : num  110 110 93 110 175 105 245 62 95 123 ...
```

```
## $ drat: num 3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
## $ wt : num 2.62 2.88 2.32 3.21 3.44 ...
## $ qsec: num 16.5 17 18.6 19.4 17 ...
## $ vs : num 0 0 1 1 0 1 0 1 1 1 ...
## $ am : num 1 1 1 0 0 0 0 0 0 0 ...
## $ gear: num 4 4 4 3 3 3 3 4 4 4 ...
## $ carb: num 4 4 1 1 2 1 4 2 2 4 ...
```

```
#Transformamos aquellas variables (columnas) que les corresponda a factores
#cilindrada
mtcars$cyl = as.factor(mtcars$cyl)
#marchas
mtcars$gear = as.factor(mtcars$gear)
#nº carburadores
mtcars$carb = as.factor(mtcars$carb)
str(mtcars)
```

```
## 'data.frame': 32 obs. of 11 variables:
## $ mpg : num 21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
## $ cyl : Factor w/ 3 levels "4","6","8": 2 2 1 2 3 2 3 1 1 2 ...
## $ disp: num 160 160 108 258 360 ...
## $ hp : num 110 110 93 110 175 105 245 62 95 123 ...
## $ drat: num 3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
## $ wt : num 2.62 2.88 2.32 3.21 3.44 ...
## $ qsec: num 16.5 17 18.6 19.4 17 ...
## $ vs : num 0 0 1 1 0 1 0 1 1 1 ...
## $ am : num 1 1 1 0 0 0 0 0 0 0 ...
## $ gear: Factor w/ 3 levels "3","4","5": 2 2 2 1 1 1 1 2 2 2 ...
## $ carb: Factor w/ 6 levels "1","2","3","4",...: 4 4 1 1 2 1 4 2 2 4 ...
```

```
#Agrego con aggregate los factores creados
#Combina estos tres factores para aplicar la función a los datos
#mpg es el consumo en millas por galón
aggregate(mpg~cyl+gear+carb, data = mtcars, FUN = mean, na.rm = TRUE)
```

```
## cyl gear carb mpg
## 1 4 3 1 21.50
## 2 6 3 1 19.75
## 3 4 4 1 29.10
## 4 8 3 2 17.15
## 5 4 4 2 24.75
## 6 4 5 2 28.20
## 7 8 3 3 16.30
## 8 8 3 4 12.62
## 9 6 4 4 19.75
## 10 8 5 4 15.80
## 11 6 5 6 19.70
## 12 8 5 8 15.00
```

2. Variables globales para un DF

Para no tener que escribir continuamente el \$ para llamar a variables de un DF se emplean estas funciones.

- **attach(d.f)**: para hacer que R entienda sus variables como globales y que las podamos usar por su nombre, sin necesidad de añadir delante el nombre del data frame y el símbolo \$
 - Si ya hubiera existido una variable definida con el mismo nombre que una variable del data frame al que aplicamos **attach**, hubiéramos obtenido un mensaje de error al ejecutar esta función y no se hubiera reescrito la variable global original
- **detach(d.f)**: para devolver la situación original, eliminando del entorno global las variables del data frame