Dataframes en R

Victor Lopez

2023-01-14

Conceptos

1. Observacion: es cada una de las filas
2. Variable: es cada una de las columnas

Dataframes por defecto

```
data() # Ventana con los DF en la sesion actual de R
data(package=.packages(all.available = TRUE)) # DF que vienen en los paquetes
```

Info de un DF

```
head(df, n)
tail(df, n)
str(df) # Estructura
names(df) # Nombres de las columnas y cambiar nombres

rownames(df) # Mostrar y cambiar
colnames(df)
dimnames(df) # Lista con un vector de rownames y colnames
nrow(df) # Numero filas
ncol(df)
dim(df) # Vector con numero de filas y columnas
```

Obtener columnas o filas (Subdataframes)

```
df$nombre_variable[4:10] # Devuelve un vector o un factor, dependiendo de como este definida
# la columna
# Las variables son internas, no esta definidas en el enviroment de R
df[4, 3] # Para traer filas y columnas. Exactamente igual que con las matrices
df[df$Species == "setosa" & df$Sepal.Width > 4, ] # Trae las observaciones que cumplan eso y
#todas las columnas
subset(iris, Species == "setosa", select = c(1, 4)) # select es que columnas mostrar
```

Leer DF

```
df = read.table("../data/etc/dataframe.dat", # O un link
                # Si proviene de un https, usa textConnection(getURL("url)), W/ paquete RCurl
                header = FALSE, # La primera fila sera el nombre de las columnas o no
                col.names = c("nombre", "ancho", "altura"), # Asignar nombre a columnas
                sep = "", # Separador de los datos
                dec = ".", # Separador de decimales
                stringsAsFactors = TRUE) # Las columnas en strings, se convierten en factores

# En vez de usar read.table() para archivos .dat, tambien podemos usar:

read.csv()
read.xls() | read.xlsx() # necesita el paquete xls o xlsx
read.mtb() # minitab
read.spss()
```

Escribir DF

```
write.table(df, file = "./carpeta/midataframe.csv", sep = ",", dec = ".")
```

Crear DF

```
Algebra = c(1, 2, 0, 5, 4)
Analysis = c(3, 3, 2, 7, 9)
Statistics = c(1, 2, 0, 5, 4)

grades = data.frame(Alg = Algebra, An = Analysis, Stat = Statistics) # Nombre de columnas y datos
row.names(grades) = c("P1, P2", "P3", "P4", "P5")
```

Anadir filas y columnas

```
# ANADIR FILA
df[num_Fila, ] = c(...) # Si la fila seleccionada tiene antes otras sin datos, les pondra NA
df = rbind(df, c(...))

# ANADIR COLUMNA
Calculus = c(5, 3, 5, 1, 9)
df = cbind(df, Calculus)

df$Calculus = c(5, 3, 5, 1, 9)
```

Cambiar tipo de datos

```
df$Calculus = as.character(df$Calculus)
df$Calculus = as.integer(df$Calculus)
df$Calculus = as.numeric(df$Calculus)
```

Aplicar funciones a DF

```
# SAPPLY
sapply(subset(iris, select = 1:4, na.rm = TRUE), function(x){sqrt(sum(x))})
# x es variable, la cual la funcion la opera como si fuera un vector
# na.rm sirve para que el vector correspondiente a cada columna no devuelva NA en caso de un NA

# AGGREGATE
# Aplicar una funcion a variables de un DF, en funcion de los niveles de un factor
aggregate(cbind(Sepal.Length, Petal.Length) ~ Species + Nombre,
          data = iris,
          FUN = mean,
          na.rm = TRUE)
# Nos devuelve un DF que calculo el promeadio de Sepal.Length, Petal.Length, en funcion de
# Las combinaciones que hay entre los valores del factor Species y el factor Nombre
# Podemos omitir el cbind(), si solo es una variable
# Podemos omitir el "+", si solo es un factor
```

Mas metodos de DF

```
# Debido que al hacer un subdataframe, hereda la estructura del dataframe original
sdf = droplevels(sdf) # Borras los niveles sobrantes de los factores heredados

# Tidyverse
sdf = select(iris, startsWith("Petal")) # Trae las columnas que cumplan con la condicion
# Pueden ser ends_with("Petal"), contains("Petal)
```

Variables globales

```
# Estos son mas recomendables ejecutarlos en consola

attach(df) # 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 dat
# 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(df) # Para devolver la situación original, eliminando del entorno global las variables
# del data frame

# Example:
attach(iris) # En el apartado de variables se me abrira un "directorio" de nombre iris que
# contendra las variables (que son vectores) del DF
```

Ejercicios

Ejercicio 1

```
df = read.csv("http://winterolympicsmedals.com/medals.csv")
nrow(df)
```

```
## [1] 2311
```

```
tail(df)
```

```
##      Year  City  Sport Discipline NOC           Event Event.gender  Medal
## 2306 2006 Turin Skiing  Snowboard USA       Half-pipe            M   Gold
## 2307 2006 Turin Skiing  Snowboard USA       Half-pipe            M Silver
## 2308 2006 Turin Skiing  Snowboard USA       Half-pipe            W   Gold
## 2309 2006 Turin Skiing  Snowboard USA       Half-pipe            W Silver
## 2310 2006 Turin Skiing  Snowboard USA Snowboard Cross            M   Gold
## 2311 2006 Turin Skiing  Snowboard USA Snowboard Cross            W Silver
```

Ejercicio 2

```
df$Medal = as.factor(df$Medal)
str(df)
```

```
## 'data.frame':    2311 obs. of  8 variables:
##  $ Year        : int  1924 1924 1924 1924 1924 1924 1924 1924 1924 1924 ...
##  $ City        : chr  "Chamonix" "Chamonix" "Chamonix" "Chamonix" ...
##  $ Sport       : chr  "Skating" "Skating" "Skating" "Bobsleigh" ...
##  $ Discipline  : chr  "Figure skating" "Figure skating" "Figure skating" "Bobsleigh" ...
##  $ NOC         : chr  "AUT" "AUT" "AUT" "BEL" ...
##  $ Event       : chr  "individual" "individual" "pairs" "four-man" ...
##  $ Event.gender: chr  "M" "W" "X" "M" ...
##  $ Medal       : Factor w/ 3 levels "Bronze","Gold",..: 3 2 2 1 2 3 3 2 3 2 ...
```

```
nrow(subset(df, Medal == "Bronze"))
```

```
## [1] 764
```

```r
nrow(subset(df, Medal == "Gold"))
```

```
## [1] 774
```

```r
nrow(subset(df, Medal == "Silver"))
```

```
## [1] 773
```

Ejercicio 3

```r
str(df)
```

```
## 'data.frame':    2311 obs. of  8 variables:
##  $ Year        : int  1924 1924 1924 1924 1924 1924 1924 1924 1924 1924 ...
##  $ City        : chr  "Chamonix" "Chamonix" "Chamonix" "Chamonix" ...
##  $ Sport       : chr  "Skating" "Skating" "Skating" "Bobsleigh" ...
##  $ Discipline  : chr  "Figure skating" "Figure skating" "Figure skating" "Bobsleigh" ...
##  $ NOC         : chr  "AUT" "AUT" "AUT" "BEL" ...
##  $ Event       : chr  "individual" "individual" "pairs" "four-man" ...
##  $ Event.gender: chr  "M" "W" "X" "M" ...
##  $ Medal       : Factor w/ 3 levels "Bronze","Gold",..: 3 2 2 1 2 3 3 2 3 2 ...
```

```r
df$City = as.factor(df$City)
length(levels(df$City))
```

```
## [1] 17
```

Ejercicio 4

```r
nrow(subset(df, Event.gender == "M"))
```

```
## [1] 1386
```

```r
nrow(subset(df, Event.gender == "W"))
```

```
## [1] 802
```

Ejercicio 5

¿En qué año participaron más deportistas?

```r
years = unique(df$Year)
years
```

```
##  [1] 1924 1928 1932 1936 1948 1952 1956 1960 1964 1968 1972 1976 1980 1984 1988
## [16] 1992 1994 1998 2002 2006
```

```
athletesInYear = function(year){
  sum(df$Year == year)
}

atletas = sapply(years, athletesInYear)
atletas
```

```
##  [1]  49  41  42  51  68  67  72  81 103 106 105 111 115 117 138 171 183 205 234
## [20] 252
```

```
years[which.max(atletas)]
```

```
## [1] 2006
```

Ejercicio 6

El campo NOC indica el país del ganador de la medalla. ¿Qué país puede presumir de haber ganado más medallas de oro en los juegos de invierno entre 1960 y 1996?

```
df$NOC = as.character(df$NOC)
head(df)
```

```
##   Year     City      Sport     Discipline NOC           Event Event.gender
## 1 1924 Chamonix    Skating Figure skating AUT      individual            M
## 2 1924 Chamonix    Skating Figure skating AUT      individual            W
## 3 1924 Chamonix    Skating Figure skating AUT           pairs            X
## 4 1924 Chamonix  Bobsleigh      Bobsleigh BEL        four-man            M
## 5 1924 Chamonix Ice Hockey     Ice Hockey CAN      ice hockey            M
## 6 1924 Chamonix   Biathlon       Biathlon FIN military patrol            M
##    Medal
## 1 Silver
## 2   Gold
## 3   Gold
## 4 Bronze
## 5   Gold
## 6 Silver
```

```
sdf = df[df$Year >= 1960 & df$Year <= 1996 & df$Medal == "Gold",  ]

countries = unique(sdf$NOC)
countries
```

```
##  [1] "AUT" "CAN" "EUA" "FIN" "FRA" "NOR" "SUI" "SWE" "URS" "USA" "GBR" "NED"
## [13] "FRG" "GDR" "ITA" "TCH" "ESP" "JPN" "POL" "LIE" "EUN" "GER" "KOR" "KAZ"
## [25] "RUS" "UKR" "UZB"
```

```
medallasDeOro = function(country) {
  sum(sdf$NOC == country)
}

res = sapply(countries, medallasDeOro)
res
```

```
## AUT CAN EUA FIN FRA NOR SUI SWE URS USA GBR NED FRG GDR ITA TCH ESP JPN POL LIE
##  25  12   7  20  12  40  19  24  71  34   4  14  11  39  22   2   1   3   1   2
## EUN GER KOR KAZ RUS UKR UZB
##   9  19   6   1  11   1   1
```

```
countries[which.max(res)]
```

```
## [1] "URS"
```