1 - Oggetti di R

Massimo Aria

March, 2018

Help

```
# per ottenere informazioni sul comando mean
help(mean)
```

```
## starting httpd help server ... done
```

```
# o in alternativa
?mean
```

Vettori e Variabili

```
x <- 4  # assegna alla variabile x il valore 4
x
```

```
## [1] 4
```

```
y <- c(2,7,4,1) # crea un vettore y contenente i numeri 2,7,4 e 1
y
```

```
## [1] 2 7 4 1
```

```
c(2,7,4,1) -> y # è un espressione equivalente alla precedente

ls()   # elenca gli oggetti presenti nel workspace di R
```

```
## [1] "x" "y"
```

Matrici e operazioni algebriche

```
x*y  # esegue il prodotto scalare tra x e y
```

```
## [1]  8 28 16  4
```

```
y*y # esegue il prodotto termine a termine tra i due vettori y e y
```

```
## [1]  4 49 16  1
```

```
y^2 # esegue il quadrato termine per termine degli elementi di y
```

```
## [1]  4 49 16  1
```

```
y**2 # l'espressione è equivalente alla precedente
```

```
## [1]  4 49 16  1
```

```
# per eseguire operazioni di algebra matriciale si utilizza l'operatore %*%
t(y) %*% y
```

```
##      [,1]
## [1,]   70
```

```
# CURIOSITA': R per convenzione converte automaticamente il primo vettore in modo che l'operatore matri

y%*%y # da lo stesso risultato dell'espressione precedente perchè il primo vettore è utilizato come col
```

```
##      [,1]
## [1,]   70
```

```
y%*%t(y)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    4   14    8    2
## [2,]   14   49   28    7
## [3,]    8   28   16    4
## [4,]    2    7    4    1
```

```
# creazione di una matrice con il comando matrix

a <- matrix(1:30,5,6) # crea una matrice di dimensione 5 x 6 riempiendola con i numeri da 1 a 30
a
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]    1    6   11   16   21   26
## [2,]    2    7   12   17   22   27
## [3,]    3    8   13   18   23   28
## [4,]    4    9   14   19   24   29
## [5,]    5   10   15   20   25   30
```

```
1:30 # costruisce una sequenza di numeri da 1 a 30 (con l'operatore :)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
## [24] 24 25 26 27 28 29 30
```

```
?matrix # ulteriori informazioni sul comando matrix

a <- matrix(1:30,5,6) # riempimento per colonna
a
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]    1    6   11   16   21   26
## [2,]    2    7   12   17   22   27
## [3,]    3    8   13   18   23   28
## [4,]    4    9   14   19   24   29
## [5,]    5   10   15   20   25   30
```

```
a <- matrix(1:30,5,6, byrow=TRUE) # riempimento per riga
a
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]    1    2    3    4    5    6
## [2,]    7    8    9   10   11   12
## [3,]   13   14   15   16   17   18
## [4,]   19   20   21   22   23   24
## [5,]   25   26   27   28   29   30
```

```
# Se la lunghezza dell'oggetto da inserire nella matrice non è sufficiente allora R completa automatica
matrix(c(1,2,3,4),2,4)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    3    1    3
## [2,]    2    4    2    4
```

```
matrix(c(1,2,3),2,4)  # il comando genera un avvertimento (warning) sulla modalità di completamento del
```

```
## Warning in matrix(c(1, 2, 3), 2, 4): data length [3] is not a sub-multiple
```

```
## or multiple of the number of rows [2]
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    3    2    1
## [2,]    2    1    3    2
```

```
t(a)  # esegue la trasposta di una matrice o di un vettore
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    7   13   19   25
## [2,]    2    8   14   20   26
## [3,]    3    9   15   21   27
## [4,]    4   10   16   22   28
## [5,]    5   11   17   23   29
## [6,]    6   12   18   24   30
```

```
matrix(a,2,3) # genera una matrice piena di zeri
```

```
##      [,1] [,2] [,3]
## [1,]    1   13   25
## [2,]    7   19    2
```

```
matrix(NA,2,3) # genera una matrice di valori mancanti (NA)
```

```
##      [,1] [,2] [,3]
## [1,]   NA   NA   NA
## [2,]   NA   NA   NA
```

```
# Se si intende specificare la natura di un vettore riga o colonna, lo si può creare con il comando mat

x <- matrix(c(1,2,3,4),1,4)  # x è un vettore riga di dimensione 4 (o più correttamente una matrice di
x
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    3    4
```

```
y <- matrix(c(1,2,3,4),4,1)  # x è un vettore colonna di dimensione 4 (o più correttamente una matrice
y
```

```
##      [,1]
## [1,]    1
## [2,]    2
## [3,]    3
## [4,]    4
```

```
# Questo camando fornisce un errore perchè x e y sono due matrici per le quali non è possibile usare un

#x*y

# mentre è consentito
t(x)*y
```

```
##      [,1]
## [1,]    1
## [2,]    4
## [3,]    9
## [4,]   16
```

```r
# oppure

y %*% x
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    3    4
## [2,]    2    4    6    8
## [3,]    3    6    9   12
## [4,]    4    8   12   16
```

```r
# oppure

x %*% y
```

```
##      [,1]
## [1,]   30
```

```r
#Classiche operazioni algebriche tra vettori
x <- c(1,2,3,4)
y <- c(2,4,6,8)

v1 <- x+y
v1
```

```
## [1]  3  6  9 12
```

```r
v2 <- x-y
v2
```

```
## [1] -1 -2 -3 -4
```

```r
v3 <- x*y
v3
```

```
## [1]  2  8 18 32
```

```r
v4 <- x/y
v4
```

```
## [1] 0.5 0.5 0.5 0.5
```

```r
#Per ulteriori informazioni sugli operatori
?"+"
```

ACCEDERE A ELEMENTI DI UN VETTORE

```r
x <- 1:4

# così come

x <- -3:8

# produce una sequenza di valori e li assegna a x
# il comando generale per creare sequenze è seq
?seq

seq(from=-3,to=6,by=2)
```

```
## [1] -3 -1  1  3  5
```

```r
#oppure
seq(-3,6,2)
```

```
## [1] -3 -1  1  3  5
```

```r
seq(-3,-1,0.33)  # crea una sequenza di numeri equidistanti da -3 a -1, con distanza pari a 0.33
```

```
## [1] -3.00 -2.67 -2.34 -2.01 -1.68 -1.35 -1.02
```

```r
seq(-3,1,length=10) #crea un vettore di lunghezza 10 di numeri equidistanti da -3 a 1
```

```
##  [1] -3.0000000 -2.5555556 -2.1111111 -1.6666667 -1.2222222 -0.7777778
##  [7] -0.3333333  0.1111111  0.5555556  1.0000000
```

```r
# Creiamo una matrice e un  vettore attraverso una sequenza

A <- matrix(seq(-3,8),2,6)
A
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]   -3   -1    1    3    5    7
## [2,]   -2    0    2    4    6    8
```

```r
sequenza <- seq(-3,8)

# per accedere ad un elemento del vettore
sequenza[3]
```

```
## [1] -1
```

```r
# per accedere ad un elemento della matrice
A[2,3]
```

```
## [1] 2
```

```r
# per accedere all'intera seconda riga della matrice A
A[2,]
```

```
## [1] -2  0  2  4  6  8
```

```r
# oppure sesta colonna
A[,6]
```

```
## [1] 7 8
```

```r
# Per cercare elementi all'interno di un vettore/matrice
which(sequenza<2)
```

```
## [1] 1 2 3 4 5
```

```r
which((sequenza >= -1) & (sequenza < 5))
```

```
## [1] 3 4 5 6 7 8
```

```r
z <- which((sequenza >= -1) & (sequenza < 5))

sequenza[z]
```

```
## [1] -1  0  1  2  3  4
```

```r
# restituzione di valori logici TRUE/FALSE

sequenza < 3.2
```

```
##  [1]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE
## [12] FALSE
```

```r
# il comando which non fa altro che restituire la posizione dei valori per cui la condizione è TRUE

sequenza[sequenza < 3.2] #permette di ottenre lo stesso risultato precedente
```

```
## [1] -3 -2 -1  0  1  2  3
```

```r
# WORKSPACE

# il comando ls consente di visualizzare gli oggetti presenti nel workspace
?ls

ls()
```

```
##  [1] "a"        "A"        "sequenza" "v1"       "v2"       "v3"
##  [7] "v4"       "x"        "y"        "z"
```

```r
# i comandi getwd e setwd consentono di visualizzare e cambiare la directory di lavoro di R
?getwd
?setwd

getwd()
```

```
## [1] "C:/Users/Massimo/Documents/R/work/Laboratorio di Analisi Statistica con R/2018"
```

```r
# per salvare uno o più oggetti in un file
save(x,A, file="prova.RData")

# per rimuovere oggetti dal workspace
rm(x,A)

ls()
```

```
## [1] "a"        "sequenza" "v1"       "v2"       "v3"       "v4"
## [7] "y"        "z"
```

```r
# per cancellare completamente il workspace
rm(list=ls())

ls()
```

```
## character(0)
```

```r
#per ricaricare oggetti salvati in precedenza
load("prova.RData")

ls()
```

```
## [1] "A" "x"
```

TIPOLOGIE DI OGGETTI

```r
# In R sono definiti diversi tipi di oggetti:
# character
```

```r
# numeric
# integer
# logical
# complex

# gli oggetti possono essere dichiarati in maniera implicita o esplicita
x <- 1:3 # si crea un oggetto numerico denominato x

# che equivale al comando esplicito
x <- numeric(3)

# allo stesso modo

x <- character(10)

?integer
?complex
?logical

# LISTE

# è un contenitore di oggetti di natura diversa
?list

s <- c("tizio", "caio", "sempronio")
s
```

```
## [1] "tizio"     "caio"      "sempronio"
```

```r
log <- c(TRUE,TRUE,FALSE,FALSE,FALSE)
log
```

```
## [1]  TRUE  TRUE FALSE FALSE FALSE
```

```r
A <- matrix(1,4,2)

LISTA <- list(NOMI=s,BOOL=log, MATRICE=A)
LISTA
```

```
## $NOMI
## [1] "tizio"     "caio"      "sempronio"
##
## $BOOL
## [1]  TRUE  TRUE FALSE FALSE FALSE
##
## $MATRICE
##      [,1] [,2]
## [1,]    1    1
## [2,]    1    1
## [3,]    1    1
## [4,]    1    1
```

```r
LISTA$BOOL  # accede all'oggetto BOOL contenuto nella lista
```

```
## [1]  TRUE  TRUE FALSE FALSE FALSE
```

```
# in alternativa
LISTA[["BOOL"]]
```

```
## [1]  TRUE  TRUE FALSE FALSE FALSE
```

```
LISTA$MATRICE[2,1]
```

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

```
str(LISTA)  #fornisce la struttura dell'oggetto
```

```
## List of 3
##  $ NOMI   : chr [1:3] "tizio" "caio" "sempronio"
##  $ BOOL   : logi [1:5] TRUE TRUE FALSE FALSE FALSE
##  $ MATRICE: num [1:4, 1:2] 1 1 1 1 1 1 1 1
```

TIPI DI DATI "STATISTICI"

```
# VARIABILI QUALITATIVE SU SCALA NOMINALE e ORDINALE

sesso <- c("U","U","U","D","D","D","D")
eta <- c("giovane","giovane","adulto","adulto","anziano","giovane","anziano")

# così come dichiarati, gli oggett sesso e eta sono due tipi carattere
str(sesso)
```

```
##  chr [1:7] "U" "U" "U" "D" "D" "D" "D"
```

```
str(eta)
```

```
##  chr [1:7] "giovane" "giovane" "adulto" "adulto" "anziano" "giovane" ...
```

```
# in R è possibile transformare questi oggetti in una variabile nominale attraverso il comando FACTOR

sesso2 <- factor(sesso)
str(sesso2)
```

```
##  Factor w/ 2 levels "D","U": 2 2 2 1 1 1 1
```

```
sesso2
```

```
## [1] U U U D D D D
## Levels: D U
```

```
eta2 <- factor(eta)
str(eta2)
```

```
##  Factor w/ 3 levels "adulto","anziano",..: 3 3 1 1 2 3 2
```

```
eta2
```

```
## [1] giovane giovane adulto  adulto  anziano giovane anziano
## Levels: adulto anziano giovane
```

```
# tra i livelli della variabile eta esiste una relazione d'ordine, quindi la variabile è in realtà di t

ordered(eta2,levels=c("giovane","adulto","anziano"))
```

```
## [1] giovane giovane adulto  adulto  anziano giovane anziano
## Levels: giovane < adulto < anziano
```

```r
# o direttamente con
eta2 <- factor(eta, levels=c("giovane","adulto","anziano"),ordered=TRUE)
eta2
```

```
## [1] giovane giovane adulto  adulto  anziano giovane anziano
## Levels: giovane < adulto < anziano
```

```r
# o ancora più semplicemente
eta2=ordered(eta,levels=c("giovane","adulto","anziano"))
eta2
```

```
## [1] giovane giovane adulto  adulto  anziano giovane anziano
## Levels: giovane < adulto < anziano
```

```r
# è possibile creare un fattore anche da vettori numerici e aggiungere i livelli successivamente
sesso3 <- c(1,1,1,2,2,2,2)
sesso3
```

```
## [1] 1 1 1 2 2 2 2
```

```r
sesso4 <- factor(sesso3)
sesso4
```

```
## [1] 1 1 1 2 2 2 2
## Levels: 1 2
```

```r
levels(sesso4) <- c("U","D")
sesso4
```

```
## [1] U U U D D D D
## Levels: U D
```

```r
# se dalle etichette si vuole risalire ai codici
unclass(sesso4)
```

```
## [1] 1 1 1 2 2 2 2
## attr(,"levels")
## [1] "U" "D"
```

LA MATRICE DEI DATI (DATAFRAME)

```r
reddito <- c(1000,1200,900,1001,2000,1100,900)
reddito
```

```
## [1] 1000 1200  900 1001 2000 1100  900
```

```r
dati <- data.frame(sesso=sesso2,eta=eta2,reddito=reddito)
dati
```

```
##   sesso     eta reddito
## 1     U giovane    1000
## 2     U giovane    1200
## 3     U  adulto     900
## 4     D  adulto    1001
## 5     D anziano    2000
## 6     D giovane    1100
## 7     D anziano     900
```

```r
class(dati)
```

```
## [1] "data.frame"
```

```r
summary(dati)  # fornisce una breve sintesi descrittiva del dataframe
```

```
##  sesso      eta       reddito
##  D:4   giovane:3   Min.   : 900
##  U:3   adulto :2   1st Qu.: 950
##        anziano:2   Median :1001
##                    Mean   :1157
##                    3rd Qu.:1150
##                    Max.   :2000
```

```r
# per accedere agli elementi di un data frame

dati$sesso  # fornisce l'intera variabile sesso
```

```
## [1] U U U D D D D
## Levels: D U
```

```r
# in alternativa
dati["sesso"]
```

```
##   sesso
## 1     U
## 2     U
## 3     U
## 4     D
## 5     D
## 6     D
## 7     D
```

```r
# oppure
dati[,1]
```

```
## [1] U U U D D D D
## Levels: D U
```

```r
# Per utilizzare le variabili contenute in un data frame senza doverle richiamare continuamente dal dat

?attach
attach(dati)  # in questo modo è possibili richiamare le variabili automaticamente
```

```
## The following objects are masked _by_ .GlobalEnv:
##
##     eta, reddito, sesso
```

```r
eta
```

```
## [1] "giovane" "giovane" "adulto"  "adulto"  "anziano" "giovane" "anziano"
```

```r
reddito
```

```
## [1] 1000 1200  900 1001 2000 1100  900
```

```r
sesso
```

```
## [1] "U" "U" "U" "D" "D" "D" "D"
```

```r
# al termine
detach(dati)
```

DATA FRAME GIA' CONTENUTI IN R

```r
# Il comando DATA consente di accedere ai numerosi data frame già presenti di default in R
?data

data()

data(Titanic)
df=as.data.frame(Titanic)
summary(df)
```

```
##   Class       Sex         Age     Survived      Freq
##  1st :8   Male  :16   Child:16   No :16   Min.   :  0.00
##  2nd :8   Female:16   Adult:16   Yes:16   1st Qu.:  0.75
##  3rd :8                                  Median : 13.50
##  Crew:8                                  Mean   : 68.78
##                                          3rd Qu.: 77.00
##                                          Max.   :670.00
```

IMPORTARE DATA FRAME DA FILE DI TESTO O DAL WEB

```r
# Attraverso il comando read.table è possibile importare data frame in R
?read.table
auto=NA
auto=read.table("https://archive.ics.uci.edu/ml/machine-learning-databases/auto-mpg/auto-mpg.data", hea

str(auto)
```

```
## 'data.frame':    398 obs. of  9 variables:
##  $ V1: num  18 15 18 16 17 15 14 14 14 15 ...
##  $ V2: int  8 8 8 8 8 8 8 8 8 8 ...
##  $ V3: num  307 350 318 304 302 429 454 440 455 390 ...
##  $ V4: Factor w/ 94 levels "?","100.0","102.0",..: 17 35 29 29 24 42 47 46 48 40 ...
##  $ V5: num  3504 3693 3436 3433 3449 ...
##  $ V6: num  12 11.5 11 12 10.5 10 9 8.5 10 8.5 ...
##  $ V7: int  70 70 70 70 70 70 70 70 70 70 ...
##  $ V8: int  1 1 1 1 1 1 1 1 1 1 ...
##  $ V9: Factor w/ 305 levels "amc ambassador brougham",..: 50 37 232 15 162 142 55 224 242 2 ...
```

```r
# il comando NAMES consente di aggiungere/modificar eil nome delle variabili contenute in un data frame

names(auto)=c("mpg","cylinders","displacement","hp","weight","acceleration","year","origin","car_name")
```