Introduzione a R

Vincenzo Nardelli - vincenzo.nardelli01@icatt.it

Luiss Business School 21/06/2019

Introduzione

R è un linguaggio di programmazione e un ambiente di sviluppo specifico per l'analisi statistica dei dati. È possibile eseguire semplici operazioni inserendo l'input nella console

```
3 + 5
```

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

```
12 / 7
```

```
## [1] 1.714286
```

È possibile salvare i risultati delle operazioni in variabili.

```
result <- 3 + 5
```

Per poter vedere cosa quale è il valore di una variabile ci sono due metodi.

```
result
```

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

```
print(result)
```

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

Possiamo utilizzare il valore salvato in una variabile per effettuare altre operazioni. Ma per sovrascriverne il valore dobbiamo salvarlo con lo stesso nome.

```
result * 3.1415
```

```
## [1] 25.132
```

```
print(result)
```

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

```
result <- result * 3.1415
print(result)
```

```
## [1] 25.132
```

Funzioni

R contiene numerose funzioni a cui possiamo passare i nostri dati. Le funzioni sono delle parti di codice che semplificano l'esecuzione di una complicata serie di comandi. Molte funzioni sono predefinite e presenti in "R base" ma possono essere importate da pacchetti esterni o definite direttamente dall'utente. Ogni funzione ha degli input (argomenti) e degli output che possono essere stampati o salvati in una variabile.

```
sqrt(result)
```

```
## [1] 5.013183
```

```r
rad_result <- sqrt(result)

round(result)
```

```
## [1] 25
```

Gli argomenti di ogni funzione e maggiori dettagli sul funzionamento sono presenti nella documentazione. Per richiamarla direttamente dalla riga di comando è sufficiente eseguire il comdando "?"

```r
?round
```

Variabili non numeriche

R può gestire anche variabili non numeriche, come bool o stringhe.

```r
2 == 3
```

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

```r
string <- "questa è una frase"
print(string)
```

```
## [1] "questa è una frase"
```

Ma alcune le funzioni sono disponibili solo per alcune tipologie di dato...

```r
string + 2
```

```
## Error in string + 2: non-numeric argument to binary operator
```

```r
"2" + "3"
```

```
## Error in "2" + "3": non-numeric argument to binary operator
```

```r
class(string)
```

```
## [1] "character"
```

```r
class(result)
```

```
## [1] "numeric"
```

Strutture dati

Vettori

```r
vector1 <- c(1, 3, 8, 13)
print(vector1)
```

```
## [1]  1  3  8 13
```

```r
vector2 <- c("cane", "casa", "sole")
print(vector2)
```

```
## [1] "cane" "casa" "sole"
```

```r
class(vector1)
```

```
## [1] "numeric"
```

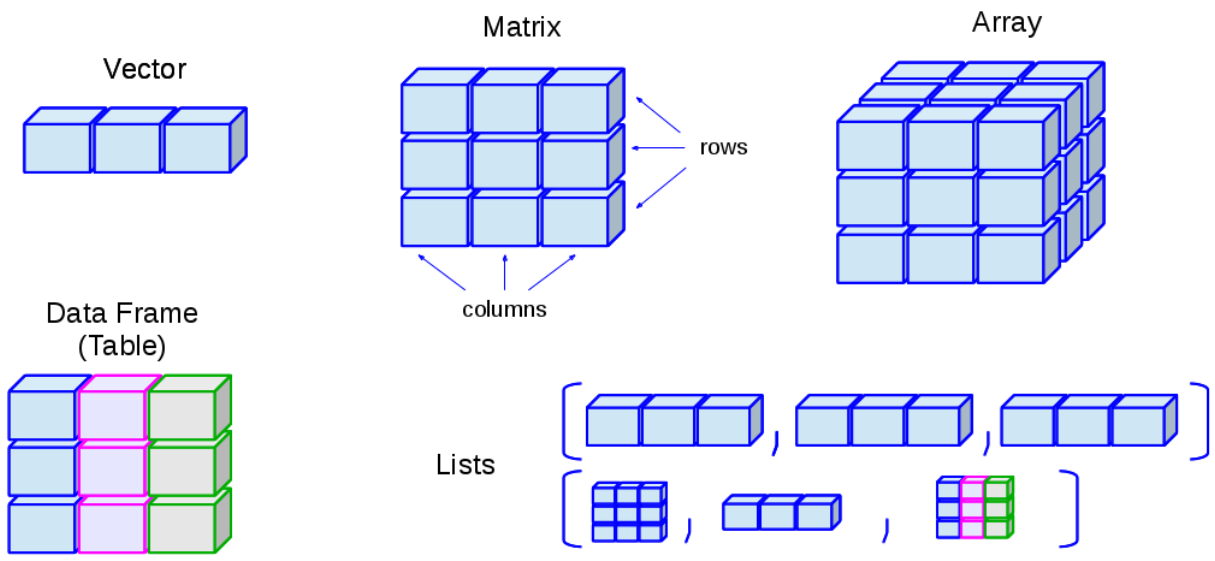

Figure 1: Strutture dati

```
class(vector2)
```

```
## [1] "character"
```

Due funzioni importanti per manipolare vettori

```
length(vector1)
```

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

```
length(vector2)
```

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

```
str(vector1)
```

```
##  num [1:4] 1 3 8 13
```

```
str(vector2)
```

```
##  chr [1:3] "cane" "casa" "sole"
```

È possibile aggiungere all'inizio o alla fine degli altri elementi.

```
vector1 <- c(34, vector1, 89)
print(vector1)
```

```
## [1] 34  1  3  8 13 89
```

Subset

È possibile estrarre un subset di un vettore tramite gli indici.

```
vector1[1]
```

```
## [1] 34
```

```r
vector1[3]
```

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

```r
vector1[4:6]
```

```
## [1]  8 13 89
```

```r
vector1[c(1, 6)]
```

```
## [1] 34 89
```

Oppure in alternativa è possibile fare un subset condizionale utilizzando un vettore booleano della stessa lunghezza del vettore da sezionare.

```r
vector1[c(TRUE, FALSE, FALSE, FALSE, FALSE, TRUE)]
```

```
## [1] 34 89
```

```r
vector1 > 3
```

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

```r
vector1[vector1 > 3]
```

```
## [1] 34  8 13 89
```

```r
vector1[vector1 > 3 & vector1 < 40]
```

```
## [1] 34  8 13
```

```r
vector2[vector2 == "casa" & vector1 == "cane"]
```

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

```r
vector2[vector2 == "casa" | vector2 == "cane"]
```

```
## [1] "cane" "casa"
```

```r
vector2 %in% c("casa", "cane")
```

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

```r
vector2[vector2 %in% c("casa", "cane")]
```

```
## [1] "cane" "casa"
```

Analisi esplorativa con R

Valori mancanti

Su R i valori mancanti sono indicati con NA

```r
rooms <- c(2, 1, 1, NA, 4)
mean(rooms)
```

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

```r
max(rooms)
```

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

```
?mean

mean(rooms, na.rm = TRUE)
```

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

```
is.na(rooms)
```

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

```
!is.na(rooms)
```

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

```
rooms[!is.na(rooms)]
```

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

```
rooms <- rooms[!is.na(rooms)]
```

Carichiamo un file esterno

```
data <- read.csv("../data/ecommerce.csv")
summary(data)
```

```
##     orderid             status         customerid
##  573585 :  1114   cancelled:  9288   Min.   :12346
##  581219 :   749   shipped  :532621   1st Qu.:13953
##  581492 :   731                      Median :15152
##  580729 :   721                      Mean   :15288
##  558475 :   705                      3rd Qu.:16791
##  579777 :   687                      Max.   :18287
##  (Other):537202                      NA's   :135080
##            country            year          month             day
##  United Kingdom:495478   Min.   :2010   Min.   : 1.000   Min.   : 1.00
##  Germany       :  9495   1st Qu.:2011   1st Qu.: 5.000   1st Qu.: 7.00
##  France        :  8557   Median :2011   Median : 8.000   Median :15.00
##  EIRE          :  8196   Mean   :2011   Mean   : 7.553   Mean   :15.02
##  Spain         :  2533   3rd Qu.:2011   3rd Qu.:11.000   3rd Qu.:22.00
##  Netherlands   :  2371   Max.   :2011   Max.   :12.000   Max.   :31.00
##  (Other)       : 15279
##       hour           minute         stockid
##  Min.   : 6.00   Min.   : 0.00   85123A :  2313
##  1st Qu.:11.00   1st Qu.:16.00   22423  :  2203
##  Median :13.00   Median :30.00   85099B :  2159
##  Mean   :13.08   Mean   :30.01   47566  :  1727
##  3rd Qu.:15.00   3rd Qu.:44.00   20725  :  1639
##  Max.   :20.00   Max.   :59.00   84879  :  1502
##                                  (Other):530366
##                              description       unitprice
##  WHITE HANGING HEART T-LIGHT HOLDER:  2369   Min.   :-11062.06
##  REGENCY CAKESTAND 3 TIER          :  2200   1st Qu.:     1.25
##  JUMBO BAG RED RETROSPOT           :  2159   Median :     2.08
##  PARTY BUNTING                     :  1727   Mean   :     4.61
##  LUNCH BAG RED RETROSPOT           :  1638   3rd Qu.:     4.13
##  (Other)                           :530362   Max.   : 38970.00
```

```
##  NA's                           :  1454
##     quantity
##  Min.   :-80995.00
##  1st Qu.:     1.00
##  Median :     3.00
##  Mean   :     9.55
##  3rd Qu.:    10.00
##  Max.   : 80995.00
##
```

```
head(data)
```

```
##   orderid  status customerid        country year month day hour minute
## 1  536365 shipped      17850 United Kingdom 2010    12   1    8     26
## 2  536365 shipped      17850 United Kingdom 2010    12   1    8     26
## 3  536365 shipped      17850 United Kingdom 2010    12   1    8     26
## 4  536365 shipped      17850 United Kingdom 2010    12   1    8     26
## 5  536365 shipped      17850 United Kingdom 2010    12   1    8     26
## 6  536365 shipped      17850 United Kingdom 2010    12   1    8     26
##   stockid                          description unitprice quantity
## 1  85123A  WHITE HANGING HEART T-LIGHT HOLDER      2.55        6
## 2   71053                 WHITE METAL LANTERN      3.39        6
## 3  84406B      CREAM CUPID HEARTS COAT HANGER      2.75        8
## 4  84029G KNITTED UNION FLAG HOT WATER BOTTLE      3.39        6
## 5  84029E      RED WOOLLY HOTTIE WHITE HEART.      3.39        6
## 6   22752        SET 7 BABUSHKA NESTING BOXES      7.65        2
```

```
nrow(data)
```

```
## [1] 541909
```

```
ncol(data)
```

```
## [1] 13
```

Calcoliamo una nuova variabile.

```
data$price <- data$unitprice * data$quantity

min(data$unitprice)
```

```
## [1] -11062.06
```

```
max(data$unitprice)
```

```
## [1] 38970
```

```
range (data$unitprice)
```

```
## [1] -11062.06  38970.00
```

```
quantile(data$unitprice)
```

```
##        0%       25%       50%       75%      100%
## -11062.06      1.25      2.08      4.13  38970.00
```

```
IQR (data$unitprice)
```

```
## [1] 2.88
```

```r
quantile(data$unitprice, 0.05)
```

```
##   5%
## 0.42
```

```r
quantile(data$unitprice, 0.95)
```

```
##  95%
## 9.95
```

```r
mean(data$unitprice)
```

```
## [1] 4.611114
```

```r
median(data$unitprice)
```

```
## [1] 2.08
```

```r
var(data$unitprice)
```

```
## [1] 9362.469
```

```r
sd(data$unitprice)
```

```
## [1] 96.75985
```

Installiamo il primo pacchetto

```r
install.packages("moments")
```

ed usiamo le funzioni appena caricate per calcolare curtosi e simmetria.

```r
library(moments)
kurtosis(data$unitprice)
```

```
## [1] 59008.17
```

```r
skewness(data$unitprice)
```

```
## [1] 186.5065
```