Prometheus. Аналіз даних та статистичне виведення на мові R. Конспект по R

Анастасія Корнілова

жовтень, 2016

Встановлення R:

- перейдіть за посиланням https://cloud.r-project.org/
- оберіть вашу операційну систему
- завантажте відповідний пакунок
- інсталюйте його

Встановлення RStudio:

- перейдіть за посиланням https://www.rstudio.com/products/rstudio/download/
- оберіть вашу операційну систему
- завантажте відповідний пакунок
- інсталюйте його

Створити новий файл в RStduio:

- в меню обрати пункт меню `File`
- далі `New File`
- обрати пункт `RScript`

R має модульну структуру. Ви встановлюєте базовий функціонал і розширяєте його потрібними вам бібліотеками. Зараз у репозиторії CRAN наличується ~ 7000 бібліотек. Для встановлення бібліотеки використовується команда `install.packages`

Скопіюйте у R файл ці команди:

```
install.packages('dplyr', dependencies = TRUE)
install.packages('ggplot2', dependencies = TRUE)
```

Щоб виконати код, виділіть рядки та натисніть піктограму Run з зеленою стрілкою або комбінацію клавіш `CTRL + ENTER` або `COMMAND + ENTER`. Також код можна набирати в консолі(нижня ліва панель в RStudio).

Для завантаження в робоче середовище команда `library`. Далі завантажте ці бібліотеки до вашого робочого середовища. Це можна зробити з допомогою функції `library`. Зауважте, що ми встановлюємо бібліотеку один раз, але завантажувати її потрібно щоразу, як ви перезапускаєте RStudio. Тобто при наступному запуску RStudio команди інсталяції не будуть потрібні і їх можна буде закоментувати використовуючи символ #:

R як калькулятор:

```
2+3-8 # додавання та віднімання
```

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

```
7*5/2 # множення та ділення
```

```
## [1] 17.5
```

```
pi # константа пі
```

```
## [1] 3.141593
```

```
sqrt(4) # корінь квадратний
```

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

```
2^3 # піднесення до степеня
```

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

Символ присвоєння: `<-`

```
# x присвоїти значення 2
x <- 2

# вивести значення x
x
```

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

`x` та `X` - різні змінні

```
X <- 3

X
```

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

```
x
```

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

Типи даних в R:

- Логічні - `TRUE`, `FALSE`
- Стрічкові - `character`
- Числові - `numeric`, `integer`, `double`, `complex`

На відміну від мов Java чи C в R не обов'язково декларувати тип змінної. Дізнатися тип має змінна можна з допомогою функції `class`

```
v1 <- TRUE
class(v1)
```

```
## [1] "logical"
```

```
v1 <- -10.6
class(v1)
```

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

```
v1 <- 3L # L вказує що це ціле число
class(v1)
```

```
## [1] "integer"
```

```
v1 <- 3+2i
class(v1)
```

```
## [1] "complex"
```

```
v1 <- "stats"
class(v1)
```

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

Типи R обєктів

- Вектор
- Матриця
- Список(list)
- Фактор
- Таблиця даних(data frame)

Вектор - набір значень одного типу. Утворюється з допомогою функції `c` (скорочення від `concatenate`).

```
numeric_vector <- c(1, 10, 49)

boolean_vector <- c(TRUE, FALSE, TRUE)

character_vector<- c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday")
```

Операції з векторами:

```
x <- c(10, 2, 3, 7, 4)
y <- c(2, -1, 3, 2, 6)

# додавання/віднімання
# додаються/віднімаються поелементно
x + y
```

```
## [1] 12  1  6  9 10
```

```
x - y
```

```
## [1]  8  3  0  5 -2
```

```r
# множення на скаляр
2*x
```

```
## [1] 20  4  6 14  8
```

```r
# застосування функції до кожного елемента
sqrt(x)
```

```
## [1] 3.162278 1.414214 1.732051 2.645751 2.000000
```

```r
# сума елементів
sum(x)
```

```
## [1] 26
```

```r
# довжина вектора
length(x)
```

```
## [1] 5
```

```r
# об'єданання векторів
z <- c(x, y)
z
```

```
##  [1] 10  2  3  7  4  2 -1  3  2  6
```

Доступ до елементів вектора.

```r
x <- 1:20 # інший спосіб задання вектора, вказуємо послідовність від 1 до 20
x
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
```

```r
# п'ятий елемент(відлік починається з одиниці)
x[5]
```

```
## [1] 5
```

```r
# елементи з 6 по 12
x[6:12]
```

```
## [1]  6  7  8  9 10 11 12
```

```r
# елементи 6, 10, 13
x[c(6, 10, 13)]
```

```
## [1]  6 10 13
```

```r
# елементи  за винятком 6 та 13
x[-c(6, 13)]
```

```
##  [1]  1  2  3  4  5  7  8  9 10 11 12 14 15 16 17 18 19 20
```

```r
# елементи, які більше 5
x[x > 5]
```

```
##  [1]  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
```

```
# елементи, які більше 5 і менші 15
x[x > 5 & x < 15]
```

```
## [1]  6  7  8  9 10 11 12 13 14
```

```
# елементи, які менші 5 або більші 15
x[x < 5 | x > 15]
```

```
## [1]  1  2  3  4 16 17 18 19 20
```

Відсутні значення (аналог `null`) позначаються як `NA` (Not avaliable). Впливають на результат обчислень.

```
x <- c(10, 20, NA, 4, NA, 2)
sum(x)
```

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

```
sum(x, na.rm = TRUE)
```

```
## [1] 36
```

Матриця - по суті двовимірний вектор

```
mat <- matrix(data=c(9,2,3,4,5,6),ncol=3)
mat
```

```
##      [,1] [,2] [,3]
## [1,]    9    3    5
## [2,]    2    4    6
```

Список (list). Якщо елементи вектора мають бути одного типу, то елементи списку можуть мати різні типи.

```
a <- list(p_name="Joe", 4, foo=c(3,8,9))
print(a)
```

```
## $p_name
## [1] "Joe"
##
## [[2]]
## [1] 4
##
## $foo
## [1] 3 8 9
```

Доступ до елементів з використанням символа `[[]]` або `$` та імені(якщо елемент має ім'я)

```
a[[3]]
```

```
## [1] 3 8 9
```

```
a[[1]]

## [1] "Joe"

a$p_name

## [1] "Joe"
```

Фактор - вектор для збереження категоріальних даних. Може містити як категоріальні впорядковані, так і категоріальні невпорядковані дані.

Нехай маємо звичайний вектор:

```
mons <- c("March","April","January","November","January","September","October
","September","November","August", "January","November","November","February"
,"May","August","July","December","August","August","September", "November","
February","April")

class(mons)

## [1] "character"
```

Створимо впорядкований фактор, в параметрі level задамо та задамо порядок:

```
mons <- factor(mons,levels=c("January","February","March","April","May","June
","July","August","September", "October","November","December"),ordered=TRUE)

class(mons)

## [1] "ordered" "factor"
```

Тепер можемо визначити, чи March < April

```
mons[1]

## [1] March
## 12 Levels: January < February < March < April < May < June < ... < Decembe
r

mons[2]

## [1] April
## 12 Levels: January < February < March < April < May < June < ... < Decembe
r

mons[1] < mons[2]

## [1] TRUE
```

Дата фрейм (data frame) використовується для роботи з таблицями.

Є три способи створити data frame.

1. Об'єднати вектори однакової довжини, використовуючи команду `data.frame`

```
cause <- c('pilot error', 'mechanical', 'weather', 'sabotage', 'other')
amount <- c(640, 195, 63, 95, 111)
plane_crash <- data.frame(cause, amount)
```

2. Використовувати вбудовані набори даних

```
head(airquality)
```

```
##   Ozone Solar.R Wind Temp Month Day
## 1    41     190  7.4   67     5   1
## 2    36     118  8.0   72     5   2
## 3    12     149 12.6   74     5   3
## 4    18     313 11.5   62     5   4
## 5    NA      NA 14.3   56     5   5
## 6    28      NA 14.9   66     5   6
```

3. Зчитати з файла (розглянемо в лабораторній)

Базові операції

```
# кількість стовпців
ncol(airquality)
```

```
## [1] 6
```

```
# кількість рядків
nrow(airquality)
```

```
## [1] 153
```

```
# назви колонок
colnames(airquality)
```

```
## [1] "Ozone"   "Solar.R" "Wind"    "Temp"    "Month"   "Day"
```

```
# всі дані для 5 місяця
airquality2 <- airquality[airquality$Month == 5, ]
head(airquality2)
```

```
##   Ozone Solar.R Wind Temp Month Day
## 1    41     190  7.4   67     5   1
## 2    36     118  8.0   72     5   2
## 3    12     149 12.6   74     5   3
## 4    18     313 11.5   62     5   4
## 5    NA      NA 14.3   56     5   5
## 6    28      NA 14.9   66     5   6
```

```
# температура для  5 місяця
airquality$Temp[airquality$Month == 5]
```

```
##  [1] 67 72 74 62 56 66 65 59 61 69 74 69 66 68 58 64 66 57 68 62 59 73 61
## [24] 61 57 58 57 67 81 79 76
```