Subsetting

Neeraj Jain

27-01-2020

Lab 4:

Subsetting can be done in three ways

1. **By Index**

```
rm(list = ls())

a <- c(1,2,3,4)

b <- a[1] #getting first element

c <- a[c(1,2)]

a[5] #returns NA as first element is not known.
```

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

```
a[c(1, 1, 1)] #first element 3 times.
```

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

```
a[c(1, 2, NA, 4)] #NA returns NA
```

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

```
a[-2] #remove 2nd element
```

```
## [1] 1 3 4
```

```
a[c(-3, -4)]
```

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

```
#another Example
a <- c(23, 24, 31, 31)
a[1]
```

```
## [1] 23
```

```
a[2]
```

```
## [1] 24
```

```
a <- a[-2] #remove 2nd element and overewrite a
a
```

```
## [1] 23 31 31
```

```
a[c(1,2)] <- c(34, 101) #replacing first 2 element of a
a
```

```
## [1]  34 101  31
```

```r
#character Vector
BB <- c("Vedansh", "Asim", "Sehnaz", "Paras", "Abhijit", "Shukla")

BB <- BB[-c(1, 5)] #removing 1st and 5th element from BB
BB
```

```
## [1] "Asim"   "Sehnaz" "Paras"  "Shukla"
```

```r
#"Aarti", "Maraya"
BB[c(5, 6)] <- c("aarti", "marya") #adding two names to BB
BB <- c(BB, "aarti", "marya") #alternative of last line #it is better too.
BB
```

```
## [1] "Asim"   "Sehnaz" "Paras"  "Shukla" "aarti"  "marya"  "aarti"  "marya"
```

```r
length(BB)
```

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

```r
BB[5] <- "Tarang"

BB <- c(BB[1:5], "Katrina Kaif", BB[6:10]) #Adding another name between the character.
BB
```

```
##  [1] "Asim"         "Sehnaz"       "Paras"        "Shukla"
##  [5] "Tarang"       "Katrina Kaif" "marya"        "aarti"
##  [9] "marya"        NA             NA
```

```r
#Another Example
SG <- c("Kukku", "Bunty", "Sartaz", "Gaitonde")

SG <- c(SG[1:2], "Gurteg", SG[3:4]) #Adding another element between the SG
SG
```

```
## [1] "Kukku"    "Bunty"    "Gurteg"   "Sartaz"   "Gaitonde"
```

```r
#Subsetting For Matrix
Mat2 <- matrix(1:16, ncol = 4)

Mat2
```

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

```r
Mat2[5]
```

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

```r
Mat2[2,3] #, menas providing index for 2nd dimension (ie column)
```

```
## [1] 10
```

```r
a <- 1:4

a[1,2] #it wonot work. Since, a has only one dimension.
```

```
## Error in a[1, 2]: incorrect number of dimensions
```

```r
Mat2[c(2,3) , ]
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    2    6   10   14
## [2,]    3    7   11   15
```

```r
Mat2[ , c(2,3)]
```

```
##      [,1] [,2]
## [1,]    5    9
## [2,]    6   10
## [3,]    7   11
## [4,]    8   12
```

```r
Mat2[c(2,3), c(3,4)]
```

```
##      [,1] [,2]
## [1,]   10   14
## [2,]   11   15
```

```r
#to add column and rows in matrix use these two function
rbind
```

```
## function (..., deparse.level = 1)
## .Internal(rbind(deparse.level, ...))
## <bytecode: 0x7fa76be3dce8>
## <environment: namespace:base>
```

```r
cbind
```

```
## function (..., deparse.level = 1)
## .Internal(cbind(deparse.level, ...))
## <bytecode: 0x7fa76ba85240>
## <environment: namespace:base>
```

```r
Mat2
```

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

```r
Paridhi <- rbind(Mat2, c(1,2,3,4))
cbind(Mat2, c(1,2,3,4))
```

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

```r
#Subsetting for list
ls <- list(a = 1:4, b = BB, c = Mat2)
ls
```

```
## $a
## [1] 1 2 3 4
##
## $b
```

```
##  [1] "Asim"         "Sehnaz"       "Paras"        "Shukla"
##  [5] "Tarang"       "Katrina Kaif" "marya"        "aarti"
##  [9] "marya"        NA             NA
##
## $c
##      [,1] [,2] [,3] [,4]
## [1,]    1    5    9   13
## [2,]    2    6   10   14
## [3,]    3    7   11   15
## [4,]    4    8   12   16
```

```
ls[c(2, 3)]
```

```
## $b
##  [1] "Asim"         "Sehnaz"       "Paras"        "Shukla"
##  [5] "Tarang"       "Katrina Kaif" "marya"        "aarti"
##  [9] "marya"        NA             NA
##
## $c
##      [,1] [,2] [,3] [,4]
## [1,]    1    5    9   13
## [2,]    2    6   10   14
## [3,]    3    7   11   15
## [4,]    4    8   12   16
```

```
str(ls[1]) #return list
```

```
## List of 1
##  $ a: int [1:4] 1 2 3 4
```

```
ls2 <- ls[c(2,3)]
str(ls2)
```

```
## List of 2
##  $ b: chr [1:11] "Asim" "Sehnaz" "Paras" "Shukla" ...
##  $ c: int [1:4, 1:4] 1 2 3 4 5 6 7 8 9 10 ...
```

```
ls3 <- ls[2]
str(ls3)
```

```
## List of 1
##  $ b: chr [1:11] "Asim" "Sehnaz" "Paras" "Shukla" ...
```

```
length(ls3)
```

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

```
ls4 <- ls[[2]] #double closed bracked to get element of list
str(ls4)
```

```
##  chr [1:11] "Asim" "Sehnaz" "Paras" "Shukla" "Tarang" "Katrina Kaif" ...
```

```
length(ls4)
```

```
## [1] 11
```

```
#Always use single index in double closed bracket.
ls[[2, 3]] #unpredictable resutls. Mostly returns error.
```

```
## Error in ls[[2, 3]]: incorrect number of subscripts
```

```r
#alternative of [[ is $
ls$b
```

```
##  [1] "Asim"         "Sehnaz"       "Paras"        "Shukla"
##  [5] "Tarang"       "Katrina Kaif" "marya"        "aarti"
##  [9] "marya"        NA             NA
```

```r
ls$c
```

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

```r
ls$c[3, c(1,2)]
```

```
## [1] 3 7
```

```r
ls$c[3, ][c(1,2)] #subsetting in subsetting result.
```

```
## [1] 3 7
```

```r
#Subsetting in Dataframe
set.seed(123)
Df <- data.frame(a = rnorm(10), b = rnorm(10))
Df
```

```
##              a          b
## 1  -0.56047565  1.2240818
## 2  -0.23017749  0.3598138
## 3   1.55870831  0.4007715
## 4   0.07050839  0.1106827
## 5   0.12928774 -0.5558411
## 6   1.71506499  1.7869131
## 7   0.46091621  0.4978505
## 8  -1.26506123 -1.9666172
## 9  -0.68685285  0.7013559
## 10 -0.44566197 -0.4727914
```

```r
#like matrix (when use ,)
Df[1,2]
```

```
## [1] 1.224082
```

```r
Df[1, ]
```

```
##           a        b
## 1 -0.5604756 1.224082
```

```r
Df[, 1]
```

```
##  [1] -0.56047565 -0.23017749  1.55870831  0.07050839  0.12928774
##  [6]  1.71506499  0.46091621 -1.26506123 -0.68685285 -0.44566197
```

```r
Df[c(7,8), 1]
```

```
## [1]  0.4609162 -1.2650612
```

```r
#like list
Df$a[c(7,8)]
```

```
## [1]  0.4609162 -1.2650612
```

```
str(Df[1])
```

```
## 'data.frame':    10 obs. of  1 variable:
##  $ a: num  -0.5605 -0.2302 1.5587 0.0705 0.1293 ...
```

```
str(Df[[1]])
```

```
##  num [1:10] -0.5605 -0.2302 1.5587 0.0705 0.1293 ...
```

Exercise

```
#Create following Df
set.seed(007)
Df2 <- as.data.frame(matrix(rnorm(16), ncol = 4))

#1. Add new row as c(1,2,3, 4)
#2. First get 2nd column using subsetting, then calculate sum, var of this.
#3. Get 2nd column of your dataframe using 3 differetn methods.
#4. Get sum of 3rd row.
#5. Replace first column with 1:5
```

---

2. **Subsetting by Logical value**

```
rm(list = ls())

#logical Operators in R
a <- c(2, 4, 6, 10)

a > 3
```

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

```
a < 3
```

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

```
a >= 3
```

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

```
a <= 3
```

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

```
a == 3
```

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

```
a != 3
```

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

```
all #alternative of excel and
```

```
## function (..., na.rm = FALSE)  .Primitive("all")
```

```
any #alternative of excel or
```

```
## function (..., na.rm = FALSE)  .Primitive("any")
```

```r
all(TRUE,TRUE, TRUE, FALSE)
```

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

```r
any(TRUE, FALSE, FALSE)
```

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

```r
b <- c("a", "a ", "a")
b[c(1,2,3)] == "a"
```

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

```r
all(b == "a")
```

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

```r
d <- c(rep(1, 100), 2, 2, rep(1, 200))
all(d == 1)
```

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

```r
which(c(TRUE, FALSE, FALSE, TRUE, TRUE)) #which returns index of true value
```

```
## [1] 1 4 5
```

```r
d[which(d != 1)] #getting values from index
```

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

```r
e <- c(TRUE, FALSE,TRUE)
!e  #! reverse the T and F
```

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

```r
which(!(d == 1))  #do not use !d == 1
```

```
## [1] 101 102
```

```r
g <- c(2, 5, 7, 10, 6)
g[c(TRUE, FALSE, TRUE, FALSE, TRUE)]
```

```
## [1] 2 7 6
```

```r
#all values greater than 5
g > 5
```

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

```r
g[g > 5]
```

```
## [1]  7 10  6
```

```r
#Small Example:
#a. values <= 2
#b.  values not equal to 7
#c.  how many's values are greater than 3

g[g <= 2] #a.
```

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

```r
g[g != 7] #b.
```

```
## [1]  2  5 10  6
```

```
sum(g > 3) #c.
```

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

```
#logical operators.
#    &, |

F & T & T  # & is and
```

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

```
c(T, F, T, F) | c(F, T, T, F)  #| is or
```

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

Class Exercise

```
#Question
set.seed(123)
marks <- rnorm(82, mean = 65, sd = 20)
```

1. Validate your data to ensure that al values are between 0-100. If any value greter than 100, replace by 100, and in case of less than 0, replace by 0.
2. Percentage of students who have scored more than 60 marks.
3. Get index of students who scored more than 90 marks, and also vector of marks.
4. Get mean of top 10 and bottom 10 students marks. [Hint: order]
5. Percentage of stuents who have scored marks between 75 - 90.
6. If gender of the students is given as follows:

```
Gender <- sample(x = c("F", "G"), size = 82, replace = TRUE)
```

7. Find number of male students, and female students.
8. Find mean, sd, min, and max of males and female marks respectively.

---

3. **By elements name**