Subsetting and Functions

Neeraj Jain

03-02-2020

1. Subsetting by logical values

```
rm(list = ls())
a <- c(1, 4 , 5, 10)
a[c(T, F, T, F)]  #values return for TRUE
```

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

```
a[a > 2] #values greater than 2
```

```
## [1]  4  5 10
```

```
a[c(T, F)] #logical values rotates in subsetting.
```

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

```
#Creating sequence of odd number
seq(1, 100,2)  #using `seq` function
```

```
##  [1]  1  3  5  7  9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45
## [24] 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91
## [47] 93 95 97 99
```

```
(1:100)[c(T, F)] #using subsetting
```

```
##  [1]  1  3  5  7  9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45
## [24] 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91
## [47] 93 95 97 99
```

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

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

```
#Subsetting only NA value
a[NA] #since, NA is logical by default, therefore it rotates and return 4 NA's.
```

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

```
#logical functions and operator
#  >  < >= <= ! & | all any
```

2. Subsetting by element name

```
#Subsetting by element name
BBE <- c(abhijit = 10, harkaran = 9.2, Guntas = 2.1, Dhawal = 20, Risabh = 20)
#element names are meta data

sum(BBE)
```

```
## [1] 61.3
```

```
#Getting marks of abhijit and guntas
BBE[c("abhijit", "Guntas")]  #highly case sensitive
```

```
## abhijit  Guntas
##    10.0     2.1
```

```
BBE["Risabh"] #marks of risabh only
```

```
## Risabh
##     20
```

```
#name of student with highest marks
max(BBE)  #maximum marks
```

```
## [1] 20
```

```
names(BBE) #names of all elements
```

```
## [1] "abhijit"  "harkaran" "Guntas"   "Dhawal"   "Risabh"
```

```
BBE == max(BBE) #testing which elements marks are equal to maximum marks.
```

```
##  abhijit harkaran   Guntas   Dhawal   Risabh
##    FALSE    FALSE    FALSE     TRUE     TRUE
```

```
names(BBE[BBE == max(BBE)]) #required result: method 1
```

```
## [1] "Dhawal" "Risabh"
```

```
names(BBE)[BBE == max(BBE)] #required result: method 2
```

```
## [1] "Dhawal" "Risabh"
```

1. Subsetting on dataframe: More complicated case:

```
#Generating some random data
set.seed(0809)
Gender <- sample(c("M", "F"), size = 20, replace = T)
Course <- sample(c("BBE", "BCOM"), size = 20, replace = T)
Accounting <- runif(n = 20, min = 10, max = 20)
Eco <- runif(n = 20, min = 10, max = 20)
df <- data.frame(Gender, Course, Accounting, Eco)


##Students cases
#1 Data for marks of BBE students
df[df$Course == "BBE", ]
```

```
##    Gender Course Accounting      Eco
## 1       M    BBE   19.73451 12.50231
## 3       M    BBE   18.48234 10.44414
## 4       M    BBE   12.92757 12.53903
## 5       M    BBE   10.18981 12.92337
## 7       F    BBE   12.22001 11.19310
## 10      M    BBE   11.77382 18.42128
## 11      M    BBE   16.41943 19.86492
## 12      M    BBE   15.81613 15.25941
```

```
## 13      M    BBE   11.82744 19.68167
## 14      M    BBE   12.39075 17.30314
## 16      M    BBE   16.79499 10.46661
## 17      F    BBE   10.24201 12.78516
## 18      F    BBE   16.61114 10.99383
## 19      F    BBE   12.20747 14.77012
```

```
df$Course
```

```
##  [1] BBE  BCOM BBE  BBE  BBE  BCOM BBE  BCOM BCOM BBE  BBE  BBE  BBE  BBE
## [15] BCOM BBE  BBE  BBE  BBE  BCOM
## Levels: BBE BCOM
```

```
#2: Data for BCOM and Female
df[df$Course == "BCOM" & df$Gender == "F", ]
```

```
##    Gender Course Accounting      Eco
## 9       F   BCOM   10.60826 12.05293
## 15      F   BCOM   18.64371 15.55129
## 20      F   BCOM   16.10270 10.95361
```

```
#3. Maixmum marks of accounting for BBE males.
a_bbe_m <- df[df$Course == "BBE"& df$Gender == "M", ]
max(a_bbe_m$Accounting)
```

```
## [1] 19.73451
```

```
#4. BCOM : Difference of ECo and accounting  marks
df2 <- df[df$Course == "BCOM", ]
df2$Diff <- df2$Accounting - df2$Eco #saving the variable to existing data frame.

df$status <- ifelse(df$Eco < 12, "Fail", "Pass") #assigning pass and fail.
df
```

```
##    Gender Course Accounting      Eco status
## 1       M    BBE   19.73451 12.50231   Pass
## 2       M   BCOM   13.08880 11.13340   Fail
## 3       M    BBE   18.48234 10.44414   Fail
## 4       M    BBE   12.92757 12.53903   Pass
## 5       M    BBE   10.18981 12.92337   Pass
## 6       M   BCOM   19.32187 13.24689   Pass
## 7       F    BBE   12.22001 11.19310   Fail
## 8       M   BCOM   19.52473 17.28952   Pass
## 9       F   BCOM   10.60826 12.05293   Pass
## 10      M    BBE   11.77382 18.42128   Pass
## 11      M    BBE   16.41943 19.86492   Pass
## 12      M    BBE   15.81613 15.25941   Pass
## 13      M    BBE   11.82744 19.68167   Pass
## 14      M    BBE   12.39075 17.30314   Pass
## 15      F   BCOM   18.64371 15.55129   Pass
## 16      M    BBE   16.79499 10.46661   Fail
## 17      F    BBE   10.24201 12.78516   Pass
## 18      F    BBE   16.61114 10.99383   Fail
## 19      F    BBE   12.20747 14.77012   Pass
## 20      F   BCOM   16.10270 10.95361   Fail
```

```
## Practice
#1. Get mean marks of BBE, and BCOM
```

```r
##a. First method
mean(df[df$Course == "BBE", ]$Accounting)
```

```
## [1] 14.11696
```

```r
##b. 2nd method
mean(df$Accounting[df$Course == "BBE"])
```

```
## [1] 14.11696
```

```r
#2. Mean marks of males and females.
mean(df$Accounting[df$Gender == "M"])
```

```
## [1] 15.25325
```

```r
mean(df$Accounting[df$Gender == "F"])
```

```
## [1] 13.80504
```

```r
#3. Cross tabulation for course and gender
table(df$Gender, df$Course) #return frequency distribution
```

```
##
##     BBE BCOM
##   F   4    3
##   M  10    3
```

```r
#4. max marks gender wise and course wise: For accounting only
max( df$Accounting[df$Course == "BBE" & df$Gender == "M"] ) #method 1
```

```
## [1] 19.73451
```

```r
max( df[df$Course == "BBE" & df$Gender == "M", ]$Accounting ) #method 2
```

```
## [1] 19.73451
```

```r
#5. create following object
a <- c(1, 5, 10, NA, 20)

#a. Remove missing values
a == NA #not right method
```

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

```r
a[!is.na(a)]
```

```
## [1]  1  5 10 20
```

```r
#b. remove NA with mean values.
mean(a, na.rm = T) #to get mean removing NA
```

```
## [1] 9
```

```r
a[is.na(a)] <-  mean(a, na.rm = T) #replacing NA value with mean
a
```

```
## [1]  1  5 10  9 20
```

3. Creating our own function

```
#1. Function for square
sqr <- function(x) {
  x^2
}

#example
sqr(23)
```

```
## [1] 529
```

```
sqr(c(1, 4, 67))
```

```
## [1]    1   16 4489
```

```
#2. power function
powerf <- function(x, power ){
  x^power
}

#example
powerf(25, 2.2)
```

```
## [1] 1189.784
```

```
powerf(x = 25, power = 2.2)
```

```
## [1] 1189.784
```

```
powerf(x = 26)
```

```
## Error in powerf(x = 26): argument "power" is missing, with no default
```

```
#3. Setting default value for arguments
powerf <- function(x, power = 1){
  x ^ power
}

#example
powerf(24)
```

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

```
#4. Skewness function
skew <- function(x) {
  Mean <- mean(x)
  L <- length(x)
  s <- sum((x - Mean)^3) / L
  s/sd(x)^3
}

#example
Data <- c(2, 5, 7, 9, 10)
skew(Data)
```

```
## [1] -0.2918646
```

```r
#5. Kurtosis
kurt <- function(x) {
  Mean <- mean(x)
  L <- length(x)
  s <- sum((x - Mean)^4) / L
  s/sd(x)^4
}

#exampole
kurt(Data)
```

```
## [1] 1.170961
```

```r
#6. Moments
moment <- function(x, order = 1) {
  sum((x - mean(x))^order) / length(x)
}


#7. Summary Statistics
ss <- function(x) {
  Mean <- mean(x)
  Sd <- sd(x)
  Skew <- skew(x)
  c(Mean = Mean, std  = Sd, Skewness = Skew, Min = min(x), Max = max(x), nobs = length(x))
}

#example
ss(Data)
```

```
##       Mean        std   Skewness        Min        Max       nobs
##  6.6000000  3.2093613 -0.2918646  2.0000000 10.0000000  5.0000000
```