Class06: R Functions

Mahsa Naeimi

4/21/23

In this class we will develop our own R functions to calculate average grades in a fictional class

we will start with simplified version of the problem, just calculating the average grade of one student.

Simplified version

```
# Example input vectors to start with
student1 <- c(100, 100, 100, 100, 100, 100, 100, 90)
student2 <- c(100, NA, 90, 90, 90, 90, 97, 80)
student3 <- c(90, NA, NA, NA, NA, NA, NA, NA)
```

WE are going to start by calculating the average score of the homeworks.

```
mean(student1)
```

```
[1] 98.75
```

To get the minimum score we can use which.min.

```
student1
```

```
[1] 100 100 100 100 100 100 100  90
```

```
which.min(student1)
```

```
[1] 8
```

I can do the average of the fisrt 7 homework scores:

```
mean(student1[1:7])
```

```
[1] 100
```

Another way to select the first 7 homeworks:

```
student1[1:7]
```

```
[1] 100 100 100 100 100 100 100
```

```
student1[-8]
```

```
[1] 100 100 100 100 100 100 100
```

Another way to drop the lowest score:

we can assign student1 to a variable to get the mean.

```
student1_drop_lowest = student1 [-which.min(student1)]
student1_drop_lowest
```

```
[1] 100 100 100 100 100 100 100
```

I can get the mean of the homework scores after dropping the lowest score by doing:

```
mean(student1_drop_lowest)
```

```
[1] 100
```

we have our first working snippet of code!

Student2:

Let's generalize student2:

```
  student2
```

```
[1] 100  NA  90  90  90  90  97  80
```

```
  student2 <- c(100, NA, 90, 90, 90, 90, 97, 80)
  student2_drop_lowest = student2 [-which.min(student2)]
  student2_drop_lowest
```

```
[1] 100  NA  90  90  90  90  97
```

There is a way to calculate the mean dropping missing values (or NA)

```
  student2 <- c(100, NA, 90, 90, 90, 90, 97, 80)
  mean(student2, na.rm = TRUE)
```

```
[1] 91
```

the looks good for student2. however for student3...

```
  student3 <- c(90, NA, NA, NA, NA, NA, NA, NA)
  mean(student3, na.rm = TRUE)
```

```
[1] 90
```

we want to know the position of the NAs. so for student2 we can use the following:

```
  student2 <- c(100, NA, 90, 90, 90, 90, 97, 80)
  which(is.na(student2))
```

```
[1] 2
```

position 2 is NA in student2.

for student3:

```
  student3 <- c(90, NA, NA, NA, NA, NA, NA, NA)
  which(is.na(student3))
```

```
[1] 2 3 4 5 6 7 8
```

For considering missing values we can mask the NA values with zeros.

```
which(is.na(student2))
```

```
[1] 2
```

```
 student2[which(is.na(student2))]
```

```
[1] NA
```

now we want to change NA to zero

```
  student2[which(is.na(student2))] <- 0
  student2
```

```
[1] 100   0  90  90  90  90  97  80
```

For student3:

```
student3 <- c(90, NA, NA, NA, NA, NA, NA, NA)
 student3[which(is.na(student3))] <- 0
 student3
```

```
[1] 90  0  0  0  0  0  0  0
```

```
student3 <- c(90, NA, NA, NA, NA, NA, NA, NA)
 mean(student3)
```

```
[1] NA
```

This is going to be our final working snippet of code for all students (with and without NA values)

```
student3 <- c(90, NA, NA, NA, NA, NA, NA, NA)
student3[is.na(student3)] <-0
student3_drop_lowest <- student3[-which.min(student3)]
  mean(student3_drop_lowest)
```

```
[1] 12.85714
```

```
x <- c(100,75,50,NA)
x[is.na(x)] <-0
x_drop_lowest <- x[-which.min(x)]
  mean(x_drop_lowest)
```

```
[1] 75
```

Q1. Function Grade

we can write it as a function:

```
#'  Calculate the average score for a vector of homework scores,
#'  dropping the lowest score,
#'  and considering NA values as zeros.
#'
#' @param x A numeric vextor of homwwork scores
#'
#' @return The average value of homework scores
#' @export
#'
#' @examples
#' student1 <- c('100', '50', NA)
#' grade(student)
#'
grade <- function(x){
  # mask NA values with zero
  x[is.na(x)] <- 0
  # dropping the lowest score
  x_drop_lowest <- x[-which.min(x)]
  mean(x_drop_lowest)
  }
```

Let's apply the function

```
student1 <- c(100, 100, 100, 100, 100, 100, 100, 90)
student2 <- c(100, NA, 90, 90, 90, 90, 97, 80)
student3 <- c(90, NA, NA, NA, NA, NA, NA, NA)

grade(student1)
```

```
[1] 100
```

```
grade(student2)
```

```
[1] 91
```

```
grade(student3)
```

```
[1] 12.85714
```

let's apply our function to a gradebook from this URL:

"https://tinyurl.com/gradeinput"

```
 URL <- "https://tinyurl.com/gradeinput"
gradebook <- read.csv(URL, row.names = 1)
head(gradebook)
```

```
          hw1 hw2 hw3 hw4 hw5
student-1 100  73 100  88  79
student-2  85  64  78  89  78
student-3  83  69  77 100  77
student-4  88  NA  73 100  76
student-5  88 100  75  86  79
student-6  89  78 100  89  77
```

let's apply my function `grade` to the gradebook using `apply` and running it by rows using `MARGIN=1`

```
apply(gradebook, 1, grade)
```

```
 student-1  student-2  student-3  student-4  student-5  student-6  student-7
     91.75      82.50      84.25      84.25      88.25      89.00      94.00
 student-8  student-9 student-10 student-11 student-12 student-13 student-14
     93.75      87.75      79.00      86.00      91.75      92.25      87.75
student-15 student-16 student-17 student-18 student-19 student-20
     78.75      89.50      88.00      94.50      82.75      82.75
```

Q2. Using your `grade()` function and the supplied gradebook, Who is the top scoring student overall in the gradebook?

```
max(apply(gradebook,1, grade))
```

```
[1] 94.5
```

the max score is 94.5

```
which.max(apply(gradebook,1, grade))
```

```
student-18
        18
```

the student getting max score was student 18.

Q3. From your analysis of the gradebook, which homework was toughest on students (i.e. obtained the lowest scores overall?

FIrst we aer going to mask NA values with zeros

```
is.na(gradebook)
```

```
            hw1   hw2   hw3   hw4   hw5
student-1  FALSE FALSE FALSE FALSE FALSE
student-2  FALSE FALSE FALSE FALSE FALSE
student-3  FALSE FALSE FALSE FALSE FALSE
student-4  FALSE  TRUE FALSE FALSE FALSE
student-5  FALSE FALSE FALSE FALSE FALSE
student-6  FALSE FALSE FALSE FALSE FALSE
student-7  FALSE FALSE FALSE FALSE FALSE
```

```
student-8  FALSE FALSE FALSE FALSE FALSE
student-9  FALSE FALSE FALSE FALSE FALSE
student-10 FALSE FALSE FALSE  TRUE FALSE
student-11 FALSE FALSE FALSE FALSE FALSE
student-12 FALSE FALSE FALSE FALSE FALSE
student-13 FALSE FALSE FALSE FALSE FALSE
student-14 FALSE FALSE FALSE FALSE FALSE
student-15 FALSE FALSE FALSE FALSE  TRUE
student-16 FALSE FALSE FALSE FALSE FALSE
student-17 FALSE FALSE FALSE FALSE FALSE
student-18 FALSE  TRUE FALSE FALSE FALSE
student-19 FALSE FALSE FALSE FALSE FALSE
student-20 FALSE FALSE FALSE FALSE FALSE
```

```
gradebook[is.na(gradebook)] <-0
```

Then, we apply the mean function to the gradebook/

```
apply(gradebook, 2, mean)
```

```
  hw1   hw2   hw3   hw4   hw5
89.00 72.80 80.80 85.15 79.25
```

The toughest homework will be hw2 considering the mean, and considering missing homework as 0.

Maybe having zeros for missing homework is too strict and is not a good representation of the homework difficulty.

One thing we can do is remove the missing value.

```
gradebook <- read.csv(URL, row.names = 1)
apply(gradebook, 2, mean, na.rm =TRUE)
```

```
     hw1      hw2      hw3      hw4      hw5
89.00000 80.88889 80.80000 89.63158 83.42105
```

Instead of assigning zeros to missing values if we directly don't consider missing value the toughest homework will be hw3 (according to the mean).

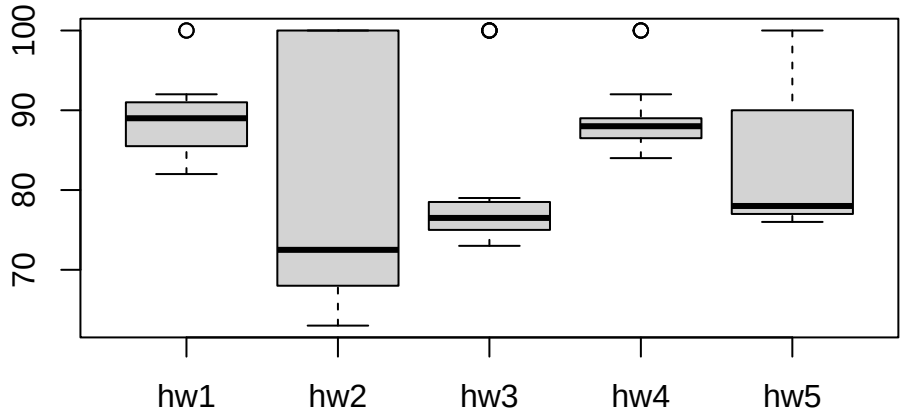
If we use the median of the mean as a measure of overal score....

```r
apply(gradebook, 2, median, na.rm = TRUE)
```

```
 hw1  hw2  hw3  hw4  hw5
89.0 72.5 76.5 88.0 78.0
```

If we use some plot...

```r
boxplot(gradebook)
```



Q4. From your analysis of the gradebook, which homework was most predictive of overall score (i.e. highest correlation with average grade score)?

```r
overall_grades = apply(gradebook, 1, grade)
overall_grades
```

```
student-1  student-2  student-3  student-4  student-5  student-6  student-7
    91.75      82.50      84.25      84.25      88.25      89.00      94.00
student-8  student-9 student-10 student-11 student-12 student-13 student-14
    93.75      87.75      79.00      86.00      91.75      92.25      87.75
```

```
student-15 student-16 student-17 student-18 student-19 student-20
     78.75      89.50      88.00      94.50      82.75      82.75
```

```
  # for column use $
  gradebook$hw1
```

```
 [1] 100  85  83  88  88  89  89  89  86  89  82 100  89  85  85  92  88  91  91
[20]  91
```

calculate correlation:

```
   cor(gradebook$hw1, overall_grades)
```

```
[1] 0.4250204
```

```
  apply(gradebook, 2, cor, y = overall_grades)
```

```
      hw1       hw2       hw3       hw4       hw5
0.4250204        NA 0.3042561        NA        NA
```

Let's change NA for zeros:

```
   gradebook [is.na(gradebook)] <- 0
   apply(gradebook, 2, cor, y = overall_grades)
```

```
      hw1       hw2       hw3       hw4       hw5
0.4250204 0.1767780 0.3042561 0.3810884 0.6325982
```

The max value is...

```
   which.max(apply(gradebook, 2, cor, y = overall_grades))
```

```
hw5
  5
```

Q5. Make sure you save your Quarto document and can click the "Render" (or Rmarkdown"Knit") button to generate a PDF format report without errors. Finally, submit your PDF to gradescope.