Class 06: R Functions

Arturo Agramont

4/21/23

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

We well start with a simplified version of the problem. Just calculating the average grade for one student.

Simplified version

```
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
```

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

```
[1] 8
```

The way in which to obtain the average of the homework scores excluding the lowest score

```
mean(student1[-which.min(student1)])
```

```
[1] 100
```

I can get the mean of homeworks after dropping by doing

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

```
[1] 100
```

Trying to generalize to student 2

```
student2_drop_lowest = student2[-which.min(student2)]
student2_drop_lowest
```

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

Finding a way to remove NA from the sequence

```
na.omit(student2)
```

```
[1] 100  90  90  90  90  97  80
attr(,"na.action")
[1] 2
attr(,"class")
[1] "omit"
```

Using to na.omit to find the mean

```
mean(na.omit(student2))
```

```
[1] 91
```

Finding mean of student 2

```
mean(student2, na.rm = T)
```

```
[1] 91
```

This does not work for student 3

```
mean(student3, na.rm = T)
```

```
[1] 90
```

We want to know positions of NAs so we can use

```
student2
```

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

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

```
[1] 2
```

For student 3

```
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 NA with zeros

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

```
[1] 2
```

```
student2[is.na(student2)] = 0
```

If used for student 3

```
student3[is.na(student3)] = 0
mean(student3)
```

```
[1] 11.25
```

Removing the lowest score

```
mean(student3[-which.min(student3)])
```

```
[1] 12.85714
```

This is going to be final working snippet for all students

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

```
[1] 12.85714
```

Let’s build a function now

```
#x[is.na(x)] = 0
#mean(x[-which.min(x)])
```

Function: grade()

Q1

We can write it as a function

```
#' Calculate the average score for a sector of homework scores, dropping the lowest score,
#'
#' @param x numeric vector of homework scores
#'
#' @return average value of homework scores
#' @export
#'
#' @examples
#'
#' student = c(100,75,50,0)
#' grade(student)
#'
grade = function(x){
  #we are masking values, changing NA to 0
  x[is.na(x)] = 0
  #finding the average grade while removing the lowest score
```

```r
  mean(x[-which.min(x)])
}
```

Let's apply the function

```r
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
```

```r
grade(student2)
```

```
[1] 91
```

```r
grade(student3)
```

```
[1] 12.85714
```

Let's apply our function to the grade book from the URL "https://tinyurl.com/gradeinput"

```r
URL = "https://tinyurl.com/gradeinput"
gradebook = read.csv(URL, row.names = 1)
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
student-7  89 100  74  87 100
student-8  89 100  76  86 100
```

```
student-9   86 100  77  88  77
student-10  89  72  79  NA  76
student-11  82  66  78  84 100
student-12 100  70  75  92 100
student-13  89 100  76 100  80
student-14  85 100  77  89  76
student-15  85  65  76  89  NA
student-16  92 100  74  89  77
student-17  88  63 100  86  78
student-18  91  NA 100  87 100
student-19  91  68  75  86  79
student-20  91  68  76  88  76
```

Applying the function using apply, with function grade on gradebookm using margin = 1

```
apply(gradebook, MARGIN = 1, FUN = 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?

```
which.max(apply(gradebook, MARGIN = 1, FUN = grade))
```

```
student-18 
        18 
```

Student 18 is the highest scoring student overall.

```
max(apply(gradebook, MARGIN = 1, FUN = grade))
```

```
[1] 94.5
```

Maximum score is 94.5

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

First we mask all of the NAs into 0, then we can take the average of each homework assignment.

```
gradebook[is.na(gradebook)] = 0
apply(gradebook, MARGIN = 2, FUN = mean)
```

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

Homework 2 was the toughest on the students, it had the lowest average of the 5 homework assignments. Having the missing homework may be too strict and not a good representation of the difficulty of the homework.

```
gradebook = read.csv(URL, row.names = 1)
apply(gradebook, MARGIN = 2, FUN = mean, na.rm = T)
```

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

In this case Homework 3 is the most difficult.

If we use the median instead of the mean…

```
apply(gradebook, MARGIN = 2, FUN = median, na.rm = T)
```

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

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

```
overall_grades = apply(gradebook, MARGIN = 1, FUN = 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 
```

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

```
[1] 0.4250204
```

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

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

The maximum is…

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

```
hw5 
  5 
```

HW 5 had the greatest correlation between the overall grade.