Class 6: R functions

Max Gruber

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 a 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 first 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:

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

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

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

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

Let's try to generalize it to student2:

```
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
```

This 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
```

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 the NA values with zeroes.

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

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

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

```
[1] 2
```

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

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

If I use the same for student 3

```
student3[which(is.na(student3))] <- 0
student3
```

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

```
mean(student3)
```

```
[1] 11.25
```

This is going to be our final workign 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
```

Let's build a function now:

```
x <- c(100,75,50,NA)

x[is.na(x)] <- 0
x_drop_lowest = x[-which.min(x)]
```

```
  mean(x_drop_lowest)
```

```
[1] 75
```

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 zeroes
#'
#' @param x A numberic vector of scores
#'
#' @return The average value of homework scores
#' @export
#'
#' @examples
#'
#' student <- c('100', '50', NA)
#' grade(student)
#'
grade <- function(x){
  # Masking NA values with 0
  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)
```

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?

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

```
student-18
        18
```

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

```
[1] 94.5
```

The maximum score is 94.5, which was earned by student 18.

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

First, we are going to mask NA values with zeros.

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

Now, 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 zeroes for missing homework is too struct and is not a good representation of the homework difficultly.

One thing we can do is remove the missing values.

```
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
```

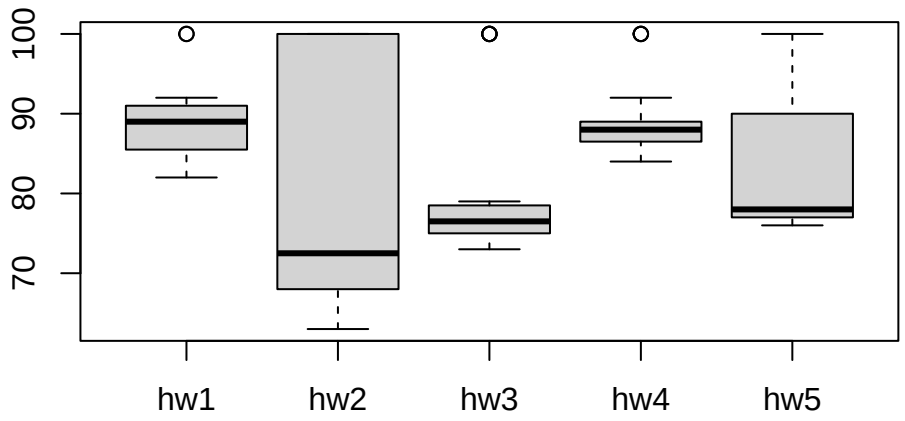
Instead of assigning zeros to missing values, if we directly don't consider missing values, the toughest homework will be hw2 (according to the mean).

If we use the median instead of the mean as a measure of overall score…

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

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

```
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)?

```
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
```

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

```
[1] 0.4250204
```

```r
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 maximum value is...

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

```
hw5
  5
```

Thus, homework 5 was most predictive of overall score.