Class 6 R functions

Wenxi Tang

Functions in R

In this class we will work through the process of developing your own function for calculating average grades for fictional students in a fictional class.

We are going to start with a simplified version of the problem where I know what the answer should be.

```
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)
```

To get the average we can use the function `mean()`

```
mean(student1)
```

```
[1] 98.75
```

The `min()` functions will return the lowest value:

```
min(student1)
```

```
[1] 90
```

The `which.min()` will tell the index fo the lowest value:

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

```
[1] 8
```

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

```
[1] 90
```

A - before the indexes of the vector will remove that index from the vector.

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

```
[1] 100
```

Student 2:

```
student2
```

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

```
mean(student2[-which.min(student2)])
```

```
[1] NA
```

```
mean(student2[-which.min(student2)], na.rm = TRUE)
```

```
[1] 92.83333
```

We can maybe use the `is.na()` function to help here but how does it work?

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

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

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

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

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

```
[1] 91
```

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

```
[1] 12.85714
```

We now have our working code snapet that can become the body of our function.

Recall that all fucntions in R have at least 3 things:

- name (we pick this)
- arguments (input to the function)
- body (where the work gets done)

```
grade <- function(x) {
  #Assign NA the value of 0
  x[is.na(x)] <- 0
  #Drop lowest value from the vector and calculate the mean
  mean( x[-which.min(x)])
}
```

Let's use this new function `grade()`

```
grade(student1)
```

```
[1] 100
```

```
grade(student2)
```

```
[1] 91
```

```
grade(student3)
```

```
[1] 12.85714
```

> Q1. Write a function grade() to determine an overall grade from a vector of student homework assignment scores dropping the lowest single score. If a student misses a homework (i.e. has an NA value) this can be used as a score to be potentially dropped. Your final function should be adquately explained with code comments and be able to work on an example class gradebook such as this one in CSV format: "https://tinyurl.com/gradeinput" [3pts]

To read this CSV file we are going to use `read.csv()`

```
# read the csv the assign the 1st column as row names.
gradebook <- read.csv("https://tinyurl.com/gradeinput", 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
```

We can use the `apply()` function to grade all the students in the gradebook. The `apply()` function will apply any function over the row (MARGIN=1) or columns (MARGIN=2) of any data.frame/matrix etc.

```
results <- apply(gradebook, 1 ,grade)
results
```

```
 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?

```
results[which.max(results)]
```

```
student-18
      94.5
```

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

We could calculate the mean for the homeworks (i.e. the columns in the gradebook)

```
#apply the mean function to each column (each homework) and find the min.
which.min(apply(gradebook, 2, mean, na.rm = TRUE))
```

```
hw3
  3
```

We could just take the sum of the cols.

```
which.min(apply(gradebook, 2, sum, na.rm=T))
```

```
hw2
  2
```

I guess we need to mask those NA values to zero.

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

```
           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   0  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   0  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   0
student-16  92 100  74  89  77
student-17  88  63 100  86  78
student-18  91   0 100  87 100
student-19  91  68  75  86  79
student-20  91  68  76  88  76
```

We can use `mask` instead of `gradebook` as we will not have NA values to mess us up.

```
which.min(apply(mask, 2, mean))
```

```
hw2
  2
```

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

```
cor(mask$hw5, results)
```

```
[1] 0.6325982
```

```
cor(mask$hw2, results)
```

```
[1] 0.176778
```

```
cor(mask$hw1, results)
```

```
[1] 0.4250204
```

```
cor(mask$hw4, results)
```

```
[1] 0.3810884
```

```
cor(mask$hw3, results)
```

```
[1] 0.3042561
```

Can we use the `apply()` function to do this all for us?

```
cor <- apply(mask, 2, cor, y = results)
cor
```

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

```
which.max(cor)
```

```
hw5
  5
```

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.