Class06

Lucy Wang

R Functions

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

We will start with a simplified version of the problem. Grade some vectors of student scores. We want to drop the lowest score and get the average.

```
# 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 can use the `mean` function to get the average.

```
mean(student1)
```

```
[1] 98.75
```

We can find the smallest value with the `min()` function.

```
min(student1)
```

```
[1] 90
```

*There is also the `which.min()` function. Let's see if this help:

```
student1
```

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

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

```
[1] 8
```

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

```
[1] 90
```

```
x <- 1:5
x
```

```
[1] 1 2 3 4 5
```

```
x[4]
```

```
[1] 4
```

```
x[-4]
```

```
[1] 1 2 3 5
```

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

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

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

```
[1] 100
```

Now what about student2 ?

```
student2
```

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

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

```
[1] NA
```

```
  which.min(student2)
```

```
[1] 8
```

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

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

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

```
[1] 91
```

But `student3` gets more tricky…(*don't miss HW! makes life harder!*)

```
  student3
```

```
[1] 90 NA NA NA NA NA NA NA
```

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

```
[1] 90
```

SUCKS! It **inflates** grades as it drops all the NAs before determining the mean…

Genius uses GOOGLE, tells me `is.na()`. Does it work?

```
  is.na(student3)
```

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

```
student2
```

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

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

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

I can use a logical vector to index another vector.

```
x <- 1:5
x[x>3]
```

```
[1] 4 5
```

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

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

```
x <- student3
is.na(x)
```

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

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

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

```
mean(x)
```

```
[1] 11.25
```

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

```
[1] 12.85714
```

We have our working snippet of code! This is now going to be the body of our function.

All function in R have at least 3 things:

- A name (we pick)
- input arguments
- a body (the code that does the work)

```
grade <- function(x){
  #mask NA to zero
  x[is.na(x)] <- 0
  #drop lowest value and get mean
  mean(x[-which.min(x)])
}
```

TRY

```
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]

```
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
```

I can use the `apply()` function to *use* our existing `grade()` function on the gradebook.

How does `apply()` work?

```
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? [3pts]

```
which.max(results)
```

```
student-18
        18
```

Q3. From your analysis of the gradebook, which homework was toughest on students (i.e. obtained the lowest scores overall? [2pts]

```
# A good way
hw <- apply(gradebook, 2, sum, na.rm = TRUE)
hw
```

```
 hw1  hw2  hw3  hw4  hw5
1780 1456 1616 1703 1585
```

```
which.min(hw)
```

```
hw2
  2
```

```
#Not a good way
which.min( apply(gradebook, 2, mean, na.rm = TRUE))
```

```
hw3
  3
```

If I want to use the mean approach I will need to mask the `NA` (missing homeworks) to zero

```
mask <- gradebook
mask[is.na(mask)] <- 0
which.min ( apply(mask, 2, mean, na.rm = TRUE))
```

```
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)? [100pt]

Here we are going to look at the correlation of each homework results (i.e. columns in the gradebook) with the overall grade of students from the course (in the `results` object obtained from using our `grade()` function)

```
mask$hw4
```

```
 [1]  88  89 100 100  86  89  87  86  88   0  84  92 100  89  89  89  86  87  86
[20]  88
```

I am going to use `cor()` function

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

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

Q5. Make sure you save your Quarto document and can click the “Render” (or Rmark- down”Knit”) button to generate a PDF foramt report without errors. Finally, submit your PDF to gradescope. [1pt]

OK!