Class06: Writing Functions

Natalie Ogg A91030809

#Functions in R

We can make functions to read data, compute things, plot things, etc. Always start by getting a working snippet of code before you tackle the function.

Gradebook Project

Goal: Create a function to calculate grades for a whole class. First, we want to create a snippet of code that doesn what we want.

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)
```

Calculate average for student1 using `mean` fxn

```
mean(student1)
```

```
[1] 98.75
```

Find the lowest grade using `min` fxn

```
min(student1)
```

```
[1] 90
```

Locate that min value using `which.min()`

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

```
[1] 8
```

Minus sign is used to exclude the designated element

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

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

Now, you can get the mean with the lowest grde dropped

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

```
[1] 100
```

But for student2 with an NA…

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

```
[1] NA
```

Doesn't work!

Use `na.rm = TRUE` to exclude the NA elements

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

```
[1] 91
```

But student3 has many NAs

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

```
[1] 90
```

However, this is only using one grade for the mean, which isn't fair because all other assignments were missing.

We want to drop only 1 NA value and assign 0 to the others.

To make this faster, we can assign a shorter variable to the student, ie `x`

```
x <- student2
```

Assigning a value of 0 to all elements of x where the value is NA: 1. `is.na(x)` tells you whether `x` elements are NA 2. `x[is.na(x)]` calls all elements where`is.na(x)` is TRUE 3. Assign those NA elements 0

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

x
```

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

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

```
[1] 91
```

Applying this to student3: pt1

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

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

pt2

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

```
[1] 12.85714
```

Making a function:

```
grade <- function(x) {
  x[is.na(x)] <- 0
  mean(x[-which.min(x)])
}
```

Testing it out:

```
grade(student1)
```

```
[1] 100
```

Final results for grades:

```
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]

My function:

```
grade <- function(x) {
  x[is.na(x)] <- 0
  mean(x[-which.min(x)])
}
```

Opening gradebook:

```
gradebook <- read.csv("https://tinyurl.com/gradeinput",
                      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
```

Using `apply` to input the data (gradebook matrix), margins (1 = apply by rows), function (`grade`):

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

```
 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(scores)
```

```
student-18
        18
```

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

Use mask to make copy of gradebook for eliminating NAs.

```
  mask <- gradebook

  mask[is.na(mask)] <- 0
  hw.avg <- apply(mask, 2, mean)
  hw.avg
```

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

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

```
hw2
  2
```

below is INCORRECT: using grade drops the lowest, and we want all data

```
  hw_grades <- apply(gradebook, 2, grade)

  hw_grades
```

```
     hw1      hw2      hw3      hw4      hw5
89.36842 76.63158 81.21053 89.63158 83.42105
```

```
  which.min(hw_grades)
```

```
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)? [1pt]

Using apply:

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

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

```
which.max(apply(mask, 2, cor, y=scores))
```

```
hw5
  5
```

```
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
```

Alternate method:

```r
correlation <- cor(scores, mask, use = "complete.obs")

correlation
```

```
          hw1      hw2       hw3       hw4       hw5
[1,] 0.4250204 0.176778 0.3042561 0.3810884 0.6325982
```

```r
which.max(correlation)
```

```
[1] 5
```

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