Class 06 R Functions

Eduardo Modolo

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 this problem. Grad 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
```

Use fn f1 to get help with `min()` function and see that there is a `which.min()` function that returns the location of extreme values

```
student1
```

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

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

```
[1] 8
```

The lowest value is in the 8th element of the vector

```
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
```

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

```
[1] 100
```

Now what about student2

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

```
[1] NA
```

Nope :(

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

```
[1] 8
```

```
  student2[-8]
```

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

```
  mean(student2)
```

```
[1] NA
```

```
  mean( c(5,5,5,NA))
```

```
[1] NA
```

after checking fn F1 mean() we found: na.rm
a logical value indicating whether NA values in x should be stripped before the computation proceeds

```
  mean( c(5,5,5,NA), na.rm = TRUE)
```

```
[1] 5
```

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

```
[1] 92.83333
```

Hmmmm... okay what about student 3

```
  student3
```

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

```
mean(student3[-which.min(student3)], na.rm = TRUE)
```

```
[1] NaN
```

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

```
[1] 90
```

This student did NOT do that good in the class. We gotta find a better way that doesn't inflate the grade.

After a quick Google search, it said to use a function called `is.na()`, how 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 the logical vector to index another vector, "access the *TRUE* values of another vector"

Use it to make the NA = 0

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

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

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

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

Combine new method of changing NA with mean

```
x <- student1
x[is.na(x)] <-0
mean(x[-which.min(x)])
```

```
[1] 100
```

It WORKS!

We have our working snippet of code! we can now use this in the body of our function

All functions 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 out the function!

```
grade(student1)
```

```
[1] 100
```

```
grade(student2)
```

```
[1] 91
```

```
grade(student3)
```

```
[1] 12.85714
```

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"

```
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
```

How to use `apply()` function, which is super useful but a bit more complicated, to use the `grade()` function to the whole class gradebook.

apply(gradebook "input for the function", margin = 1 "means applying the function over rows, 2 would be for the columns", grade "is the function we want to apply over the rows for each row")

```
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
```

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

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
```

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

```
  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
```

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

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

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

```
hw2
  2
```

Homework 2 was the toughest

If you want to use the mean approach, You will need to mask the NA (missing homeworks) to 0

```
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
```

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

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

```
mask$hw4
```

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

Im going to use the `cor()` function "google how to find correlation in R"

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

```
[1] 0.3810884
```

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

```
[1] 0.6325982
```

apply(gradebook "input for the function", margin = 1 "means applying the function over rows, 2 would be for the columns", grade "is the function we want to apply over the rows for each row")

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

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

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

```
hw5
  5
```

Homework 5 had the highest correlation!