Class 06

Dennis Kim

R Functions

In this class we will work on the process of developing our own function for calculating the 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 can help:

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

```
[1] 8
```

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

```
[1] 90
```

example

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

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

```r
x[-4]
```

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

back to the main topic, average score of student 1 without the lowest score

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

```
[1] 100
```

What about student 2?

```r
student2
```

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

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

```
[1] NA
```

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

```
[1] 8
```

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

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

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

```
[1] NA
```

Can I use this `na.rm=TRUE` argument here?

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

```
[1] 92.83333
```

Well, what about student 3?

```r
student3
```

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

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

```
[1] 90
```

This is not what we want, as it removes all of the NA's and does not help us here.

With the power of the friendship and the internet, we have found the `is.na()` function, but what is it and how does it work?

```r
student3
```

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

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

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

`is.na()` will identify if there are NA values, and you can convert those NA values into zeros but setting them with `"groupname"[is.na("groupname")] = 0)`

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

```
[1] 12.85714
```

Whoo it works and this is now going to be the body of our function.

All functions in R have at least 3 things:

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

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

Lets try it out (make sure to run the code, if there is an error where it cannot find the function, just run the code first (to introduce it to r) and then try again)

```r
grade(student1)
```

```
[1] 100
```

```r
grade(student2)
```

```
[1] 91
```

```r
grade(student3)
```

```
[1] 12.85714
```

1) Lets use the grade function to determine the overall grade of assignments while dropping the lowest score

Lets try it on the given data set

```
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
```

our `grade()` function does not work, we need to observe with something else. I can use the super useful but a bit more complicated `apply()` function to use our existing `grade()` function on the whole class gradebook.

How does this `apply()` function work? `apply()` goes in this order, `apply(input, margin (= 1 for rows, = 2 for columns), function)`, or in our case ‘apply(gradebook (the data set), 1 (we want to apply this to our rows), grade (the function we want to apply))

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

2) To sort by score

```
results[order(results, decreasing = TRUE)]
```

```
student-18  student-7  student-8 student-13  student-1 student-12 student-16
     94.50      94.00      93.75      92.25      91.75      91.75      89.50
 student-6  student-5 student-17  student-9 student-14 student-11  student-3
     89.00      88.25      88.00      87.75      87.75      86.00      84.25
 student-4 student-19 student-20  student-2 student-10 student-15
     84.25      82.75      82.75      82.50      79.00      78.75
```

```r
# call back the results using the order function
```

From this we can see that the top scoring student is student 18 with an average score of 94.5.

3) What assignment was the toughest on the students?

To sort by assignment, use `apply()` but change the margin to 2

```r
tough.assignment <- apply(gradebook, 2, sum, na.rm=TRUE)
tough.assignment
```

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

From these results, we can see that hw2 was the toughest on students.

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

```
hw3 
  3 
```

From this it shows homework 3 is the worst assignment, however this is incorrect as it removes the NA scores altogether.

If I want to use the mean approach, I will need to mask the NA (missing assignments)

```r
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
```

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

```
hw2
  2
```

```
mean(results)
```

```
[1] 87.425
```

4) From the gradebook analysis, which was the most predicative of the overall score

Look at the correlation between the average homework scores and student's average scores

I will use the `cor()` function

```
noNAgrades <- gradebook
noNAgrades[is.na(noNAgrades)]=0
correlation <- cor(noNAgrades, results)
correlation
```

```
         [,1]
hw1 0.4250204
hw2 0.1767780
```

```
hw3 0.3042561
hw4 0.3810884
hw5 0.6325982
```

From the results, homework 5 has the highest correlation to the grades of the students and as a result is the most predictive of the overall score

I want to use the `apply()` function as well (mask and noNAgrades are the same, just have different names)

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

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