Lab06

BominXie

Questions to answer:

> 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]

```r
# 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)
```

1. A working snippet of code that solve a simple problem:

```r
student1 <- c(100, 100, 100, 100, 100, 100, 100, 90)

which.min(student1) # Find the element of the vector is the lowest
```

```
[1] 8
```

Drop the lowest score from the mean() calculation

```r
student1[-8] # Return everything but the eighth element of the vector
```

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

Replace the manual input of minima with the one identified by R:

```
mean(student1[-which.min(student1)]) # The working snippet
```

```
[1] 100
```

2. Generalize the working snippet to all dataset.

Use of na.rm=TRUE argument is not a good approach.

```
mean(student2, na.rm=TRUE) # Drop the NA value in the vector, which is ideal
```

```
[1] 91
```

```
mean(student3, na.rm=TRUE) # But, it is not applicable for student 3, since they only did
```

```
[1] 90
```

Replace all NA value with zeros might work. We need to identify the NA values first.

```
is.na(student2) # Identify the value is NA or not
```

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

```
which(is.na(student2)) # Locate where the NA value is at
```

```
[1] 2
```

After identification of NA values, we would like to "mask" them, and calculate the value.

```
x <- student2 # Assign student2 to vector x
x[is.na(x)]<-0 # Assign all NA value with 0
mean(x[-which.min(x)]) # Combine the working snippet with the above code
```

```
[1] 91
```

Similarly, for student3:

```r
student3 <- c(90, NA, NA, NA, NA, NA, NA, NA)
y <- student3 # Assign student2 to vector x
y[is.na(y)]<-0 # Assign all NA value with 0
mean(y[-which.min(y)]) # Combine the working snippet with the above code
```

```
[1] 12.85714
```

Now the function is done. We would convert the snippet and turn it into a function.

```r
#' Calculate the average score of student scores, while dropping the lowest value.
#' Missing values wouldbe treated as zeros.
#'
#' @param x A numeric vector, from the student gradebook (homework scores).
#'
#' @return Average score of the student, after dropping the lowest grade.
#' @export
#'
#' @examples
#'  students <- c(100, NA, 90, 68)
#'  grade(student)
#'
#'
grade <- function(x){
  x[is.na(x)]<-0 # Assign all NA value with 0
  mean(x[-which.min(x)]) # Calculate mean score, and exclude the lowest score for calculat
}
```

Test the function:

```r
grade(student1)
```

```
[1] 100
```

```r
grade(student2)
```

```
[1] 91
```

```r
grade(student3)
```

```
[1] 12.85714
```

Applying the function to the whole class data:

```r
url <- "https://tinyurl.com/gradeinput"
gradebook <- read.csv(url, row.names = 1)
```

```r
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 
```

> Q2:Using your grade() function and the supplied gradebook, Who is the top scoring student overall in the gradebook? [3pts]

Based on the results, the apply() function could help us find the top scoring student:

```r
results <- apply(gradebook, 1, grade)
sort(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
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]

To find the lowest score overall, we could see the mean value of each column:

```r
mean_scores <- apply(gradebook, 2, mean, na.rm=TRUE)
which.min(mean_scores)
```

```
hw3
  3
```

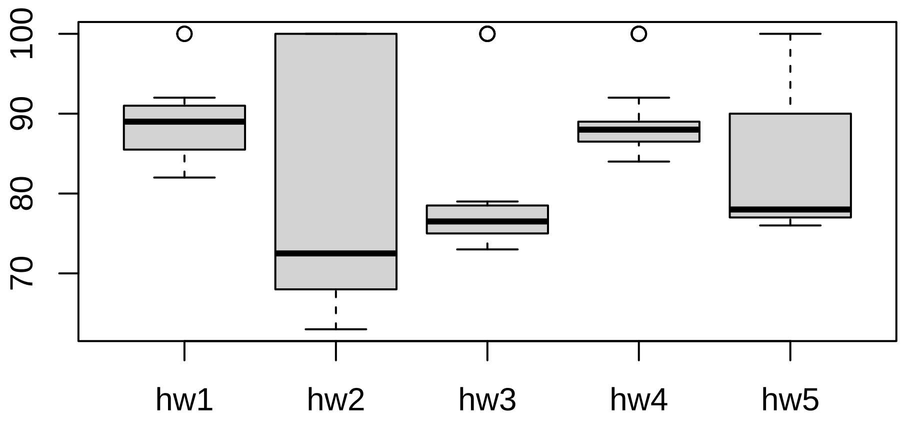
Or, we could use medium value:

```r
median_scores <- apply(gradebook, 2, median, na.rm=TRUE)
which.min(median_scores)
```

```
hw2
  2
```

It seems like mean and median are contradict with each other, we could look at the boxplot of those homeworks:

```r
boxplot(gradebook)
```



Homework 2 has a HUGE spread comparing to homework 3, which cause the median and mean inconsistencies.

> Q4. From your analysis of the gradebook, which homework was most predictive of overall score (i.e. highest correlation with average grade score)? [1pt]

We could use Cor() function to analyze the correlation betweeen two different factors, x and y:

```r
masked.gradebook <- gradebook
masked.gradebook[is.na(masked.gradebook)] <-0
correlation.results <- cor(results, masked.gradebook)
which.max(correlation.results)
```

```
[1] 5
```

Therefore, homework 5 is most predictive of overall score.

> Q5. 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. [1pt]