Class 06: R Functions

Claire Chapman

10/15/2021

Question 1

> 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"

Simple example vectors

```
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)
```

Mean without dropping the lowest score

```
mean(student1)
```

```
## [1] 98.75
```

Mean with dropping the lowest score

which.min() returns the position of the minimum value

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

```
## [1] 8
```

This will drop the lowest score from our vector

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

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

Take the mean with the lowest score dropped from the vector

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

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

Dealing with NAs

Simply removing NAs wouldn't be fair in the final grades. We want the NAs to be treated as 0s Use **is.na()** to find and replace NAs with 0

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

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

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

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

Combine into one function

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

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

```
x <- student2
x[is.na(x)] <- 0
mean(x[-which.min(x)])
```

```
## [1] 91
```

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

```
## [1] 12.85714
```

Time to make the function!

Go to Code > Extract function on working snippet

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

```
grade(student1)
```

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

```
grade(student2)
```

```
## [1] 91
```

```
grade(student3)
```

```
## [1] 12.85714
```

Add code comments

For better understanding of **why** in the future

```
grade <- function(x) {
  # make missing homework scores equal to zero
  x[is.na(x)] <- 0
  # drop the lowest homework score
  mean(x[-which.min(x)])
}
```

Code > Insert Roxygen Skeleton to make your own "Help" page for your new function

```
#' Calculate average score for a vector of homework scores while dropping the lowest score. Missing val
#'
#' @param x Numeric vector of homework scores
#'
#' @return Average score
#' @export
#'
#' @examples
#'  student <- c(100, NA, 90, 80)
#'  grade(student)
grade <- function(x) {
  # make missing homework scores equal to zero
  x[is.na(x)] <- 0
  # drop the lowest homework score
  mean(x[-which.min(x)])
}
```

Apply function to entire gradebook

Read in the spreadsheet, replace the quotes that were in the wrong font

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

Change first column of students to row names

```
gradebook <- read.csv(url, row.names = 1)
```

Use **apply()** to apply the **grade()** function to the whole gradebook "1" denotes going by 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
```

Question 2

> Using your grade() function and the supplied gradebook, Who is the top scoring student overall in the gradebook?

Use **sort** to pick out the top score from the results

```
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
```

Another way to find the top score

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

```
## student-18
##         18
```

Student 18 has the top average score

Question 3

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

Find the median (less affected by outliers) of each column of the gradebook to see which assignment had the lowest performance

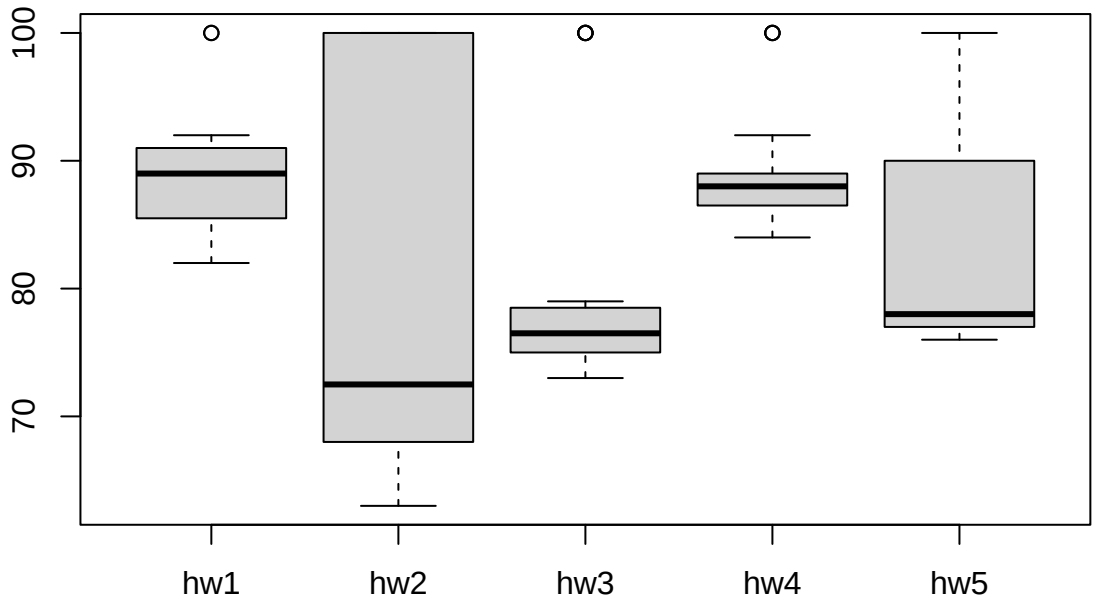
```
toughest_hw <- apply(gradebook, 2, median, na.rm=TRUE)
which.min(toughest_hw)
```

```
## hw2
##   2
```

Homework 2 was the toughest on students when using the median

Always a good idea to check with a plot of your data

```
boxplot(gradebook)
```



Question 4

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

Use cor() to find the best correlation between the results and the gradebook

```
gradebook[is.na(gradebook)] <- 0
correlations <- apply(gradebook, 2, cor, x = results)
```

```
which.max(correlations)
```

```
## hw5
##   5
```

Homework 5 scores were the most correlated with overall performance by students