Lab 6: R Functions

Taylor Darby

10/19/2021

Here we will write a function to grade some student homework.

We will start with a more simple input example - a vector of student homework scores.

```
# 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)
```

> 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**]

```
student1
```

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

The regular average will be returned by the 'mean()' function.

```
mean(student1)
```

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

To find the position of the "min" value in our vector we can use 'which.min()'

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

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

```
student1
```

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

```
student1[-8]
```

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

To return the value in the position specified by 'which.min()' we can use 'student1[which.min(student1)]'. This gets us the min value.

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

```
## [1] 90
```

To get everything but the min value use the same code but us a minus symbol

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

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

Then we take the mean of the remaining scores

```
# first option to solve Q1
mean(student1[which.min(student1)])
```

```
## [1] 90
```

```
student2
```

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

Is this a good idea?

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

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

```
mean(student2[-2])
```

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

```
student3
```

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

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

```
## [1] 90
```

This is a bad idea of using the 'na.rm=TRUE' argument and will be unfair.

So lets map/change the *NA* values to zero. How do I indentify *NA* values? Use the 'is.na()' function.

```
student2
```

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

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

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

```r
x <- student2
x
```

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

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

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

```r
mean(x)
```

```
## [1] 79.625
```

Combine our working snippets to find the average score for student3

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

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

Now we can make our function

We will take our working snippet and make it a function

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

Now use it

```r
grade(student1)
```

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

```r
grade(student2)
```

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

```r
grade(student3)
```

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

> Reminder of Q1 requirements: 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"

Now we can take the gradebook and **grade the whole class** of multiple students.

```
#' Calculate average score for a vector of homework scores
#' dropping the lowest single score. Missing values will be treated as zero
#' score.
#'
#' @param x Numeric vector of homework scores
#'
#' @return Average score
#' @export
#'
#' @examples
#' student <- c(100, NA, 90, 80)
#' grade(student)
#'
grade <- function(x) {
  # Map missing homework (NA) homework values to zero
  # Missing homework scores zero
  x[is.na(x)] <- 0
  # We exclude lowest score homework before calculating grade
  mean(x[-which.min(x)])
}
```

Now we can take the gradebook and **grade the whole class** of multiple students.

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

```
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**]

```
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
```

```
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**]

Here we want to calculate a summary stat for each column of the gradebook. Which stat should we use?

```
# Let's try average
hw.ave <- apply(gradebook, 2, mean, na.rm=TRUE)
which.min(hw.ave)
```

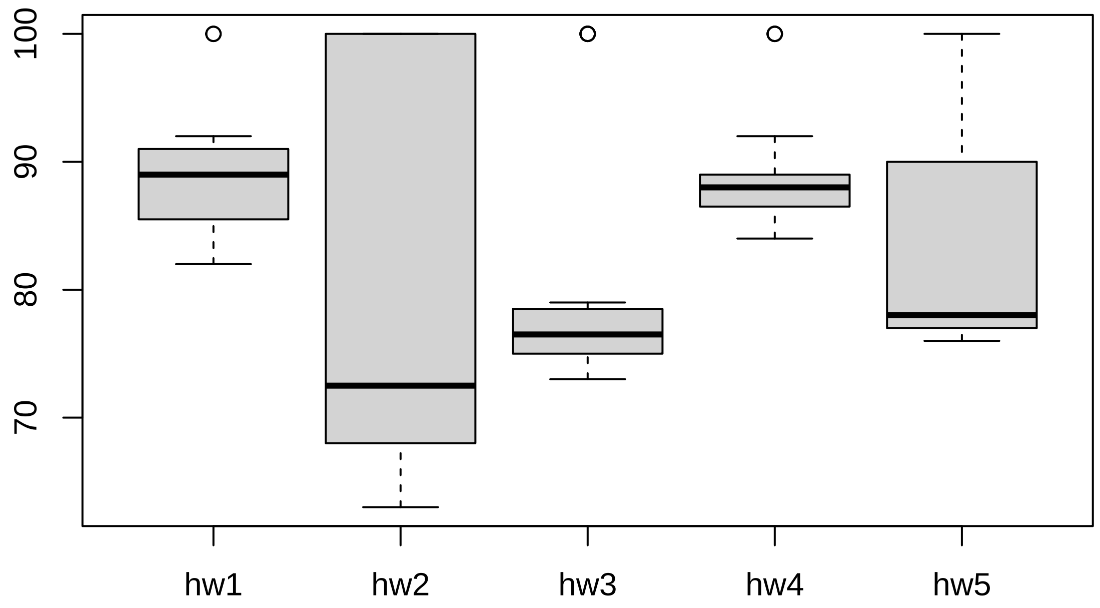
```
## hw3
##   3
```

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

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

There is a different test when using mean and median. Good idea to plot the data and see.

```
boxplot(gradebook)
```



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**]

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

Let's make a PDF report