Homework for Week 5 Friday Class

Mirte Ciz Marieke Kuijpers

04/02/2022

Introduction

The topic of this class is the creation of functions. The assignment is to create functions to help with pulling statistics from a gradebook of student grades for homework assignments.

Question 1

Q1 requires writing a function, grade(), to obtain an overall grade (i.e. average grade) for each student. It provides the example file student_homework.csv, which the function must be able to use. It also stipulates that the lowest grade a student receives should not be included in the calculation of the average.

Below is a function I wrote during class, before the final solution was revealed. While it is functional, it would not be efficient for larger inputs, thus the class solution to question 1 will also be given.

```r
##create the function grade using the function function(args){code to carry out}
grade.M <- function(File, Header=TRUE, Print=TRUE){
  #Print a few notes for this function, such that people are aware of important information.
  print("WARNING: this function assumes that the students name/number or other identifier is in the first column of the input data. If this is not the case, this function will give an error.")
  print("Note: this function presumes your input file has a header, unless the second argument (Header) is set to FALSE.")
  print("Secondary Note: this function will print rather than return results unless Print (the third arguement) is set to FALSE.")
  #read in the .csv input, using input arguments
  dat <- read.csv(File, header = Header)

  #turn NA into 0, this way students are penalized for not having turned in an assignment
  dat[is.na(dat)] = 0

  #create a data frame to store grades
  scores <- data.frame(Student=character(), Average.Grades=integer(), Letter.grade=character())
  #iteratively grade students using a for loop
  for(i in 1:length(dat[,1])){
    #find the minimum value which will be removed from grades such that the
lowest grade does not contribute to the student's average
    mv <- which.min(as.numeric(dat[i, -1]))
    #print(paste("The index of the minimum score for student", i, "is", mv,
"."))

    #create input for data frame: student name/number and average grade
    stu <- paste("Student", i)
    scrs <- dat[i, -1]
    scr <- mean(as.numeric(scrs[-mv]))

    #feed average grades to the data frame
    scores <- rbind(scores, c(stu, scr))

    #give a letter grade to each student
    if(scores[i,2] > 90){
      scores[i,3] <- "A+"
    } else if(scores[i,2] > 80){
      scores[i,3] <- "A"
    }else if(scores[i,2] > 70){
      scores[i,3] <- "B"
    }else if(scores[i,2] > 60){
      scores[i,3] <- "C"
    }else if(scores[i,2] > 50){
      scores[i,3] <- "D"
    }else{
      scores[i,3] <- "Fail"
    }

  }

  #Set the column names back to what they should be
  colnames(scores) <- c("Student", "Average Grade", "Letter Grade")

  #return or print scores
  if(Print){
    print(scores)
  }else{
    return(scores)
  }

}
```

A test of the function below shows that it does work.

```
#test the function
grade.M("student_homework.csv")
```

```
## [1] "WARNING: this function assumes that the students name/number or other
identifier is in the first column of the input data. If this is not the case,
this function will give an error."
## [1] "Note: this function presumes your input file has a header, unless the
second argument (Header) is set to FALSE."
## [1] "Secondary Note: this function will print rather than return results
unless Print (the third arguement) is set to FALSE."
##       Student Average Grade Letter Grade
## 1   Student 1         91.75           A+
## 2   Student 2          82.5            A
## 3   Student 3         84.25            A
## 4   Student 4         84.25            A
## 5   Student 5         88.25            A
## 6   Student 6            89            A
## 7   Student 7            94           A+
## 8   Student 8         93.75           A+
## 9   Student 9         87.75            A
## 10 Student 10            79            B
## 11 Student 11            86            A
## 12 Student 12         91.75           A+
## 13 Student 13         92.25           A+
## 14 Student 14         87.75            A
## 15 Student 15         78.75            B
## 16 Student 16          89.5            A
## 17 Student 17            88            A
## 18 Student 18          94.5           A+
## 19 Student 19         82.75            A
## 20 Student 20         82.75            A
```

However, as discussed below, the above function (grade.M()), is not the most efficient way to complete this question. Thus, the class solution is given below. First the data is read in and assigned to an r object.

```
#read in data
gradebook <- read.csv("student_homework.csv", header= TRUE, row.names = 1) #
row.names = 1 sets the first column of the data read in to the row names of
the r object
#check the data is as expected
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
```

Then the function is written and applied to the data.

```
#function for a vector
grade <- function(x){
  x[is.na(x)] = 0
  mean(x[-(which.min(x))])
}

#apply function for a vector across the data frame gradebook
scores <- apply(gradebook, MARGIN = 1, FUN = grade)

#print scores
scores

##  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

For question 2 we need to find the top scoring student overall, this is simple using the which.max() function which provides the index of the first highest score in a vector.

```
which.max(scores)

## student-18 
##         18
```

Question 3

For question 3 the aim is to ascertain which homework assignment was found most difficult by the students i.e. which assignment had the lowest score overall? This can be done using the which.min() function which is identical to which.max() except that it returns the index of the first minimum score.

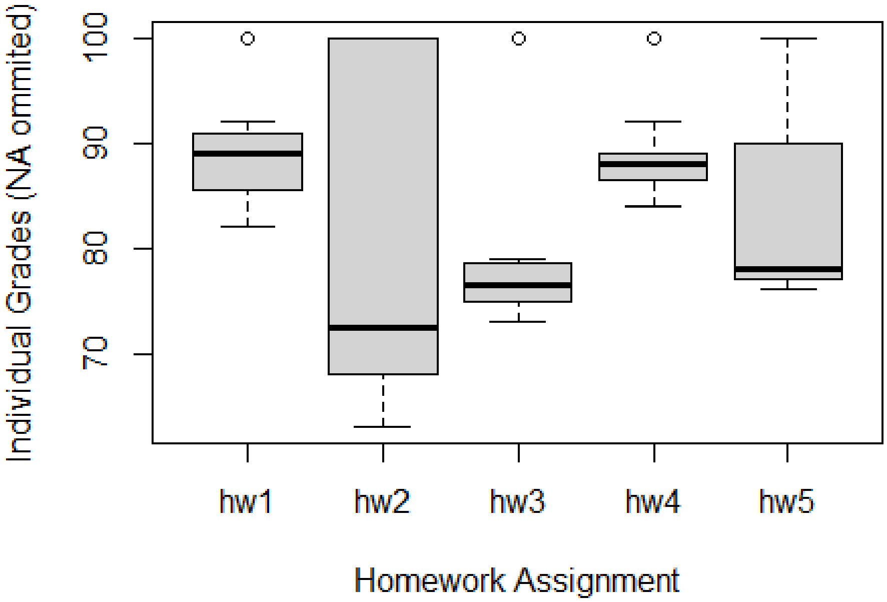
```
which.min(apply(gradebook, MARGIN = 2, FUN = mean, na.rm=TRUE))

## hw3 
##   3
```

However, the above does not take account of NA values, and it is possible that those assignments not turned in are the hardest (such that students cannot even complete them).

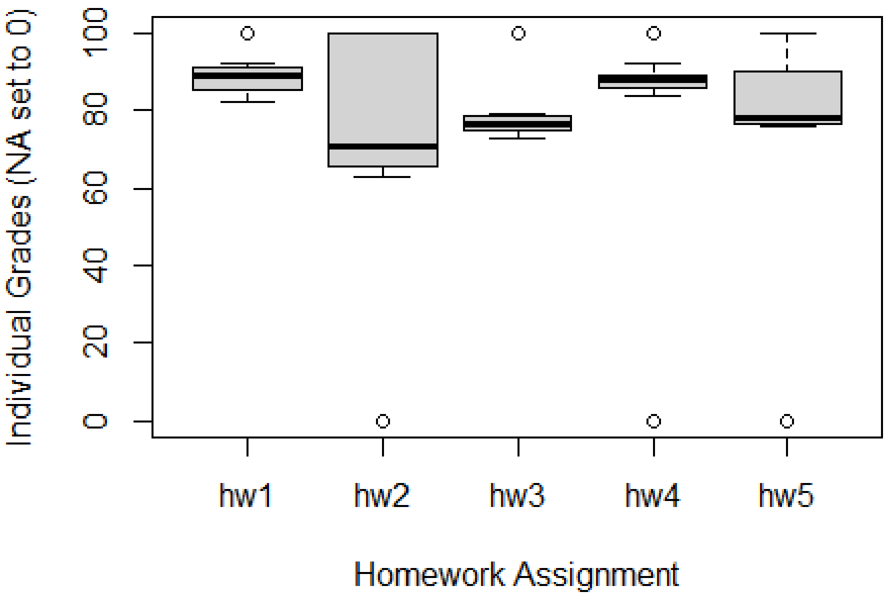
```
#replace NA with 0 in gradebook and assign this to grade0
grade0 <- gradebook
grade0[is.na(grade0)] <- 0

#use which.min to ascertain which assignment was hardest if NA are set to a
score of 0
which.min(apply(grade0, MARGIN = 2, FUN = mean))

## hw2
##   2
```

Plot the data to make sure that these results make sense.

```
boxplot(gradebook, ylab="Individual Grades (NA ommited)", xlab="Homework
Assignment")
```



```
boxplot(grade0, ylab="Individual Grades (NA set to 0)", xlab="Homework
Assignment")
```



Question 4

To answer the question of which homework assignment correlates the best with average grades will require the use of two previously created r objects. Namely: scores, which gives the average score for each student, and grade0, which is the gradebook with all NA values converted to 0. Additionally, the cor function will be useful here, as explained in the help pages, this function computes the correlations between two inputs.

```
#first a simpler example, considering a particular column of group0
cor(grade0$hw1, scores)

## [1] 0.4250204

#the help page indicates that cor can accommodate matrices, thus the whole of
grade0 can be considered at once
apply(grade0, 2, cor, scores) #note the 2 indicates that cor should be
applied to the columns of grade0

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

An alternative method might be to calculate the differences of each individual assignment grade from the average, and then find which has minimal difference, as below. However, this is considerably more messy than the above solution.

```
#initialize a matrix for use in the calculation
dif <- matrix(ncol=4, nrow=0)

#calculate the difference from the average of each individual assignment
grade
for(i in 1:length(scores)){
  dif <- rbind(dif, abs(grade0[i,] - scores[i]))
}

#find which homework assignment correlates best with each student's average
grade
corre <- apply(dif, 1, which.min)
```

Question 5

Knit this document to a pdf!