class06

Kyle Alvarez

Table of contents

Make three student vectors that have the same length, but have different values and print them out to see if the vectors contain the specified values

```
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)

student1
```

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

```
student2
```

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

```
student3
```

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

Function basics

All functions in R consit of at least 3 things: - A **name** (we can pick this but it must start with a character) - input **arguments** (there can be multiple comma separated inputs - The **body** (where the work actually happens)

Can start by using the `mean()` function to calculate an average

```r
mean(student1)
```

```
[1] 98.75
```

Can find the minimum value of a vector using the `min()` function Note: Can use F1 to as shortcut to see what a function does

```r
min(student1)
```

```
[1] 90
```

To find the index at which the minimum exists, can use `which.min()`

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

```
[1] 8
```

I can get the same vector without the 8th element with the minus index trick…

```r
student1[-8]
```

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

So I will combine the output of `which.min()` with the minus index trick to get the student scores without the lowest value

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

```
[1] 100
```

For student 2 and student 3 this gives NA

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

```
[1] NA
```

```
mean(student3[-which.min(student3)])
```

```
[1] NA
```

Can replace all NA (missing values) with zero.

```
student3 [ is.na(student3) ] <- 0
student3
```

```
[1] 90  0  0  0  0  0  0  0
```

```
mean( (student3[ -which.min(student3)]) )
```

```
[1] 12.85714
```

Copy pasting is silly and dangerous - time to write a function Class function:

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

```
[1] 12.85714
```

^ Working snippet of code that can be simplified to work with any student `x`.

My Grade Function

Now turn into a function:

```
grade <- function(x){
  x[ is.na(x)] <- 0 # assigns 0 to all NA occurrences
  mean( x[ -which.min(x)]  ) # removes the lowest grade, then takes the mean of the remain
}
```

```
grade(student1)
```

```
[1] 100
```

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

Have a look at the first 6 rows

```
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
```

Time to learn about the `apply()` function. 1 for rows, or 2 for columns

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

Q2

> Q2 : Which student did the best overall?

```
results[ which.max(results) ]
```

```
student-18
      94.5
```

Q3

> Q3 : Which homework was toughest on the students (o.e. obtained the lowest scores overall)?

```
which.min( apply(gradebook, 2, sum, na.rm=TRUE) )
```

```
hw2
  2
```

```
lowestScoreOverall <- apply(gradebook, 2, grade)
lowestScoreOverall[which.min(lowestScoreOverall) ]
```

```
     hw2
76.63158
```

Q4

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

```
mask <- gradebook
mask[ is.na(mask) ] <- 0

cor(mask$hw5, results)
```

```
[1] 0.6325982
```

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

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

My function:

```
grade2 <- function(arg1) {
  arg1[is.na(arg1)] <- 0 # changes all the NA values to 0
  newVector <- mean(arg1[-which.min(arg1)])
}
```