class06

Sawyer (PID: A16335277)

2024-01-25

R Functions

Functions are how we get stuff done. We call functions to do everything useful in R.

One cool thing about R is that it makes writing your own functions comparatively easy.

All functions in R have at least three things:

- A **name** (we get to pick this)
- One or more **input arguments** (the input to our function)
- The **body** (lines of code that do the work)

```r
funname <- function(input1, input2) {
  # The body with R code
}
```

Let's write a silly first function to add two numbers:

```r
x <- 5
y <- 1
x + y
```

```
[1] 6
```

```r
addme <- function(x, y=1) {
  x + y
}
```

```r
addme(100, 100)
```

```
[1] 200
```

```r
addme(10)
```

```
[1] 11
```

Lab for today

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"

```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)
```

```r
# we can set null values to 0
student3[is.na(student3)] <- 0
student3
```

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

```r
# we can find the index of the min value
which.min(student1)
```

```
[1] 8
```

```r
grade <- function(x) {
  # set null values to 0
  x[is.na(x)] <- 0

  # eliminate min value and take mean
  mean(x[-which.min(x)])
}
```

```r
grade(student1)
```

```
[1] 100
```

```r
grade(student2)
```

```
[1] 91
```

```r
grade(student3)
```

```
[1] 12.85714
```

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

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
```

```r
# apply function across all rows of df
results <- apply(gradebook, 1, grade)
results
```

```
 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?

```r
# find max score and index
max(results)
```

```
[1] 94.5
```

```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?

```r
# find hw with lowest total sum of scores, excluding na
which.min(apply(gradebook, 2, sum, na.rm=T))
```

```
hw2
  2
```

Based on the sums, it looks like hw2 had the lowest scores overall.

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)?

```r
# Make all (or mask) NA to zero
mask <- gradebook
mask[is.na(mask)] <- 0
# mask
```

We can use the `cor()` function for correlation analysis.

```r
cor(mask$hw5, results)
```

```
[1] 0.6325982
```

```r
cor(mask$hw3, results)
```

```
[1] 0.3042561
```

I need to use the `apply()` function to run this analysis over the whole course (i.e. masked gradebook)

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

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

Hw5 seems to have the highest correlation with overall score (results).