Class 6: R Functions

Alyssa Hayashi

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 us that it makes writing your own function comparitively easy.

All functions in R have at least 3 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)

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

Let's Write silly first function to add two numbers:

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

```
[1] 6
```

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

```
addme(1,1)
```

```
[1] 2
```

```r
addme(100)
```

```
[1] 101
```

Lab for today

```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
grade<- function(x){
  mean(x, na.rm=TRUE)
}
```

Let's just find the average

```r
grade(student1)
```

```
[1] 98.75
```

```r
grade(student2)
```

```
[1] 91
```

```r
grade(student3)
```

```
[1] 90
```

This is not fair grading Come back to this NA problem

We want to drop the lowest score before getting the mean()

```r
student1
```

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

```r
min(student1)
```

```
[1] 90
```

I found the 'which.min' function

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

```
[1] 8
```

Cool it's the 8th element of the vector that has the lowest score. Can I remove that?

```r
#find the lowest score in the vector
student1[ which.min(student1) ]
```

```
[1] 90
```

```r
#find and remove lowest score from the vector 'vector[-which.min()]'
student1[ -which.min(student1) ]
```

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

```r
x<-1:5
x[-3]
```

```
[1] 1 2 4 5
```

Now put these bits of knowledge together to make some code that identifies and drops the lowest score

```r
#find the lowest score and remove it from the mean calculation
grade<- function(x){
  mean(x[ -which.min(x) ], na.rm=TRUE)
}
```

```r
grade(student1)
```

```
[1] 100
```

```r
grade(student2)
```

```
[1] 92.83333
```

We still have the problem of missing values

Replace NA values with 0

```r
y<-1:5
y[y==3]<-1000
y
```

```
[1]    1    2 1000    4    5
```

```r
y<- c(1,1,2,NA,4,5)
y==NA
```

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

```r
is.na(y)
```

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

How can I remove NA elements from the vector?

```r
#y[is.na(y)]
c( FALSE,FALSE,FALSE)
```

```
[1] FALSE FALSE FALSE
```

```r
!c(F,F,F)
```

```
[1] TRUE TRUE TRUE
```

```r
y[!is.na(y)]
```

```
[1] 1 1 2 4 5
```

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

```
[1] 1 1 2 0 4 5
```

Put it together

> 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]

```r
grade<- function(x){
  #Changes NA values to zero
  x[is.na(x)]<-0
  #Finds and removes the min value and gets the mean
  mean(x[ -which.min(x) ])
}

grade(student1)
```

```
[1] 100
```

```r
grade(student2)
```

```
[1] 91
```

```r
grade(student3)
```

```
[1] 12.85714
```

Now read the online grade book

```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(x, MARGIN , FUN, )
#Margin: 1=columns, 2=rows
#apply(gradebook, 1, grade())

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

```r
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? [3pts]

```r
max(results)
```

```
[1] 94.5
```

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

```
student-18 
        18 
```

```r
results[18]
```

```
student-18 
      94.5 
```

Q3. From your analysis of the gradebook, which homework was toughest on students (i.e. obtained the lowest scores overall? [2pts]

```r
results<-apply(gradebook, 2, mean, na.rm=T)
results
```

```
     hw1      hw2      hw3      hw4      hw5 
89.00000 80.88889 80.80000 89.63158 83.42105 
```

```r
min(results)
```

```
[1] 80.8
```

```r
which.min(results)
```

```
hw3 
  3 
```

```r
results<-apply(gradebook, 2, sum, na.rm=T)
results
```

```
 hw1  hw2  hw3  hw4  hw5 
1780 1456 1616 1703 1585 
```

```r
min(results)
```

```
[1] 1456
```

```r
which.min(results)
```

```
hw2 
  2 
```

> 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]

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

#mask
```

We can use the 'core()' function for correlation analysis.

```r
results<-apply(gradebook, 1, grade)
cor(mask$hw1, results)
```

```
[1] 0.4250204
```

```r
cor(mask$hw2, results)
```

```
[1] 0.176778
```

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

```
[1] 0.3042561
```

```r
cor(mask$hw4, results)
```

```
[1] 0.3810884
```

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

```
[1] 0.6325982
```

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

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