3 Functions and Macros Homework

James Young

6/12/2019

# General instructions.

There are five exercises below. You are required to provide solutions for at least four of the five. You are required to solve at least one exercise in R, and at least one in SAS. You are required to provide five solutions, each solution will be worth 10 points. Thus, you may choose to provide both R and SAS solutions for a single exercise, or you may solve all five problems, mixing the languages as you wish. The first three exercises refer back to Homework 2 and you should produce the same values for this set of exercises.

CookingTooMuch.dat <- data.frame(  
 Year=c(1936, 1946, 1951, 1963, 1975, 1997, 2006),  
 CaloriesPerRecipeMean = c(2123.8, 2122.3, 2089.9, 2250.0, 2234.2, 2249.6, 3051.9),  
 CaloriesPerRecipeSD = c(1050.0, 1002.3, 1009.6, 1078.6, 1089.2, 1094.8, 1496.2),  
 CaloriesPerServingMean = c(268.1, 271.1, 280.9, 294.7, 285.6, 288.6, 384.4),  
 CaloriesPerServingSD = c(124.8, 124.2, 116.2, 117.7, 118.3, 122.0, 168.3),  
 ServingsPerRecipeMean = c(12.9, 12.9, 13.0, 12.7, 12.4, 12.4, 12.7),  
 ServingsPerRecipeSD = c(13.3, 13.3, 14.5, 14.6, 14.3, 14.3, 13.0))

#### Experimental

Again, you will be allowed to provide one solution using Python. Elaborate on the similarities and differences between Ptyhon function definitions and R or IML or Macro language.

# Exercise 1

### Please Grade This

Implement Cohen’s as a function of

where is a pooled standard deviation. Use the formula . You may implement pooled standard deviation as a function as well.

Calculate the effect size for the differences among calories per serving, 1936 versus 2006, 1936 vs 1997 and 1997 vs 2006. Use the values from Wansink, Table 1 as given in Homework 1 or in the course outline. Name this function cohen.d (or similar if using SAS)

## Answer

Define your function(s) in the code chunk below, then call the function with appropriate arguments in the following sections

# function definition  
  
cohen.d <- function(m1, s1, m2, s2) {  
 sdpooled <- ((((s1\*\*2 + s2\*\*2))/2)^(1/2))  
 (abs(m1 - m2))/sdpooled  
   
}

#### 1936 versus 2006

cohen.d(268.1, 124.8, 384.4, 168.3)

## [1] 0.784987603959

#### 1936 versus 1997

cohen.d(268.1, 124.8, 288.6, 122.0)

## [1] 0.166115727787

#### 1997 versus 2006

cohen.d(288.6, 122.0, 384.4, 168.3)

## [1] 0.651769377713

Check your work by comparing with the previous homework.

# Exercise 2.

### Please Grade This

Implement the required replicates calculation as a function of , , and as required parameters, and and as optional parameters. Let alpha=0.05 and beta=0.2, so you’ll need to compute quantiles for 1-alpha/2 and 1-beta.

Your function should return an integer , such that

where and .

You may use the pooled standarad deviation function from Ex. 1 (if you defined such a function).

Name this function required.replicates (or similar if using SAS)

## Answer

Define your function(s) in the code chunk below, then call the function with appropriate arguments in the following sections

# function definition  
  
required.replicates <- function (m1, s1, m2, s2) {  
 CV <- (((s1\*\*2+s2\*\*2)/2)^(1/2))/((m1+m2)/2)  
 Diff <- (m1-m2)/((m1+m2)/2)  
 sdpooled <- ((((s1\*\*2 + s2\*\*2))/2)^(1/2))  
 2 \* ((CV/Diff)\*\*2) \* (((1-(0.05/2))+(1-0.2)))\*\*2  
   
   
   
}

#### 1936 versus 2006

required.replicates(268.1, 124.8, 384.4, 168.3)

## [1] 10.2258899144

#### 1936 versus 1997

required.replicates(268.1, 124.8, 288.6, 122.0)

## [1] 228.352202023

#### 1997 versus 2006

required.replicates(288.6, 122.0, 384.4, 168.3)

## [1] 14.8333350465

Check your work by comparing with the previous homework.

# Exercise 3

### Please Grade This

Implement the likelihood formula as a function or macro.

Define and as optional parameters, taking values mu=0 and sigma=1. Name this function norm.pdf

## Answer

Define your function(s) in the code chunk below, then call the function with appropriate arguments in the following sections

# function definition  
norm.pdf <- function(x) {  
 (1/((2\*pi)^(1/2))) \* exp(-((x-0)^2)/((2\*1)^2))  
   
}

#### 

norm.pdf(-0.1)

## [1] 0.397946170357

#### 

norm.pdf(0.0)

## [1] 0.398942280401

#### 

norm.pdf(0.1)

## [1] 0.397946170357

Check your work by comparing with the previous homework.

# Exercise 4

### Don’t GRADE

The probability mass function for value from Poisson data with a mean and variance is given by

Write a function pois.pmf that accepts two parameters, x and lambda. Use the built in factorial function for . Note that should be an integer value, so call a rounding function inside your function. Test your function with at

## Answer

Define your function(s) in the code chunk below, then call the function with appropriate arguments in the following sections

# function definition

#### 

#### 

#### 

You can check your work against the built in Poisson distribution functions.

Something to ponder. Note that there are two formula given. Can you implement both forms in R/IML/Macro language? Would there be a difference in computational speed or efficiency?

# Exercise 5

### Please Grade This

Write a function, stat.power that combines calculations from Exercises 1 and 2. This function should accept and as required parameters, and and as optional parameters. This function must return a list with named elements CV, PercentDiff, EffectSize and RequiredReplicates.

If you choose to do this exercise in SAS, you will need to write a subroutine that accepts the same parameters as the R function, but also accepts output parameters CV, PercentDiff, EffectSize and RequiredReplicates. See <https://blogs.sas.com/content/iml/2012/08/20/how-to-return-multiple-values-from-a-sasiml-function.html>.

Another option for SAS is to package the calculations in a macro and create a data table, using the code from Course Outline SAS Source (under Course Outline > Outline Source and Output Files), about line 320.

## Answer

Define your function(s) in the code chunk below, the call the function with appropriate parameters in the following sections

# function definition  
  
stat.power <- function(m1, s1, m2, s2) {  
   
 CV <- (((s1\*\*2+s2\*\*2)/2)^(1/2))/((m1+m2)/2)  
 Diff <- (abs(m1-m2))/((m1+m2)/2)\*100  
 sdpooled <- ((((s1\*\*2 + s2\*\*2))/2)^(1/2))  
 replicates <- 2 \* ((CV/(Diff/100))\*\*2) \* (((1-(0.05/2))+(1-0.2)))\*\*2  
 sdpooled <- ((((s1\*\*2 + s2\*\*2))/2)^(1/2))  
 effectsize <- (abs(m1 - m2))/sdpooled  
   
 print(CV)  
 print(Diff)  
 print(effectsize)  
 print(replicates)  
   
   
}

If you define the class of the list returned by your function as stat.power, this function should work automatically; you shouln’t need to call the function explicity.

print.stat.power <- function(value) {  
 cat(paste("Coefficient of Variation :",value$CV\*100,"\n"))  
 cat(paste("Percent Difference :",value$PercentDiff\*100,"\n"))  
 cat(paste("Effect Size :",value$EffectSize,"\n"))  
 cat(paste("Required Replicates :",value$RequiredReplicates,"\n"))  
}

#### 1936 versus 2006

stat.power(268.1, 124.8, 384.4, 168.3)

## [1] 0.454115573275  
## [1] 35.6475095785  
## [1] 0.784987603959  
## [1] 10.2258899144

#In descending order CV, PercentDiff, Effect Size, Replicates

#### 1936 versus 1997

stat.power(268.1, 124.8, 288.6, 122.0)

## [1] 0.443355277161  
## [1] 7.36482845339  
## [1] 0.166115727787  
## [1] 228.352202023

#In descending order - CV, PercentDiff, Effect Size, Replicates

#### 1997 versus 2006

stat.power(288.6, 122.0, 384.4, 168.3)

## [1] 0.436803881088  
## [1] 28.4695393759  
## [1] 0.651769377713  
## [1] 14.8333350465

#In descending order - CV, PercentDiff, Effect Size, Replicates