3 Functions and Macros

Andrew Marshall

6/12/2018

# General instructions.

Exercise 1 is required and must be solved using both R and SAS. This exercise is worth 20 points, 10 points each for R and SAS. The R portion is included in this document and includes a unit test that checks the results of your calculations.

Exercises 2 is also required, but you may choose to use either R or SAS. This exercise is worth 10 points.

Note that Exercise 1 and 2 both use the same unit tests as Homework 2.

Exercises 3-6 are optional; choose two and choose one language for each, as previously.

A couple exercises ask you to refer to the fisher.lsd function from lecture, I’ve included this function above.

# Exercise 1. (SAS and R, required)

1. Write a function in R to return Cohen’s . Apply this function to calories per recipe, calories per serving and servings per recipe, for years 1936 and 2006. Compare the results to Homework 2.

Use the fisher.lsd function from lecture as a template. Your function should accept two means and two standard deviations as parameters.

1. Similarly, write a SAS macro or IML function to compute Cohen’s and repeat a.

d\_12 <- 0  
#Variables for "calories per recipe" (CPR) to be input into function  
 m\_1 <- 2123.8 #Mean CPR 1936  
 m\_2 <- 3051.9 #Mean CPR 2006  
 s\_1 <- 1050.0 #SD CPR 1936  
 s\_2 <- 1496.2 #SD CPR 2006  
  
#Function created using Cohen's $d$  
d\_12CPR <- function(m\_1,m\_2,s\_1,s\_2) {  
 d\_12\_var <- (abs(m\_1-m\_2))/(sqrt((s\_1^2 + s\_2^2)/2))  
 return(d\_12\_var)  
}  
d\_12 <- d\_12CPR(m\_1,m\_2,s\_1,s\_2)  
d\_12

## [1] 0.718065555553

# Variables for "calories per serving" (CPS) to be input into function  
m\_1 <- 268.1 #Mean CPS 1936  
m\_2 <- 285.6 #Mean CPs 2006  
s\_1 <- 124.8 #SD CPS 1936  
s\_2 <- 118.3 #SD CPS 2006  
  
#Function created using Cohen's $d$ for CPS  
 d\_12CPS <- function(m\_1,m\_2,s\_1,s\_2) {  
 d\_12\_var <- (abs(m\_1-m\_2))/(sqrt((s\_1^2 + s\_2^2)/2))  
 return(d\_12\_var)  
}   
d\_12 <- d\_12CPR(m\_1,m\_2,s\_1,s\_2)  
d\_12

## [1] 0.143922236212

# Variables for "servings per recipe" (SPR) to be input into function  
m\_1 <- 12.9 #Mean SPR 1936  
m\_2 <- 12.4 #Mean SPR 2006  
s\_1 <- 13.3 #SD SPR 1936  
s\_2 <- 14.3 #SD SPR 2006  
  
# Assigning formula to variable d\_12 (SPR).  
 d\_12SPR <- function(m\_1,m\_2,s\_1,s\_2) {  
 d\_12\_var <- (abs(m\_1-m\_2))/(sqrt((s\_1^2 + s\_2^2)/2))  
 return(d\_12\_var)  
 }  
 d\_12 <- d\_12SPR(m\_1,m\_2,s\_1,s\_2)  
 d\_12

## [1] 0.0362081257545

## d is not assigned the correct value

# Exercise 2. (SAS or R, required)

Using the formula for log-likelihood from Homework 2, Exercise 2, write a function that accepts x,mu () and sigma () as parameters and returns the corresponding likelihood value. Make mu and sigma optional parameters with values mu=0 and sigma=1. As with Homework 2, call your function with values of to pass the unit tests.

# Assigning variables values for use with function calcuations  
mu <- 0  
sigma <- 1  
  
# function using x = 0.1   
l\_1 <- 0  
x <- 0.1  
l\_1\_func <- function(x,mu,sigma) {   
l\_2\_var <- (1/(sigma \* sqrt(2 \* pi))) \* (exp(-((x-mu)^2)/(2\*sigma ^ 2)))   
return(l\_2\_var)  
}   
l\_1 <- l\_1\_func(x,mu,sigma)  
l\_1

## [1] 0.396952547477

# function using x = 0.2  
l\_2 <- 0  
x <- 0.2  
l\_2\_func <- function(x,mu,sigma) {   
l\_2\_var <-(1/(sigma \* sqrt(2 \* pi))) \* (exp(-((x-mu)^2)/(2\*sigma ^ 2)))  
return(l\_2\_var)  
}   
l\_2 <- l\_2\_func(x,mu,sigma)  
l\_2

## [1] 0.391042693975

### Unit Test (R)

## [1] 4

## [1] 8

# Exercise 3 (R or SAS)

Write a function/macro that accepts two means, two standard devations, two n and an optional . Inside this function, call both the fisher.lsd function and your Cohen’s function, and return the results as a list. Also include in your list the difference between the means and if the differences are significant. You can use the code for lsd.test from the course outline as a template.

Call your function with Calories per Recipe values from 1936 and 2006, assign the results to res3 and print the results as below. The return value from your function should match the list object.

#Variables for "calories per recipe" (CPR) to be input into function  
 m\_1 <- 2123.8 #Mean CPR 1936  
 m\_2 <- 3051.9 #Mean CPR 2006  
 s\_1 <- 1050.0 #SD CPR 1936  
 s\_2 <- 1496.2 #SD CPR 2006  
 n\_1 <- 18 # Total number of recipes  
 n\_2 <- 18 # Total number of recipes to keep recipe range at 18  
   
 #CohenD function written for use with res3.func function below  
 CohenD <- function(m\_1,m\_2,s\_1,s\_2) {  
 local.d\_12 <- (abs(m\_1-m\_2))/(sqrt((s\_1^2 + s\_2^2)/2))  
 return(local.d\_12)  
 }  
 # CohenD(m\_1,m\_2,s\_1,s\_2)  
   
 #Function written to produce desired result based on requested CPR data  
 res3.func <- function(m\_i,m\_j,s\_i,s\_j,n\_i,n\_j, alpha=0.05){  
 local.LSD <- fisher.lsd(s\_i,n\_i,s\_j,n\_j)  
 local.CohenD <- CohenD(m\_i,m\_j,s\_i,s\_j)  
 local.Difference <- abs(m\_i-m\_j)  
 return (list(Difference=local.Difference,LSD = local.LSD,CohenD = local.CohenD,  
 Significant = local.Difference>=local.LSD))  
 }  
 # res3.func(m\_1,m\_2,s\_1,s\_2,n\_1,n\_2,alpha=0.05)  
   
 res3 <- res3.func(m\_1,m\_2,s\_1,s\_2,n\_1,n\_2,alpha=0.05)  
 # list(Difference=0,LSD=0,CohenD=0,Significant=0)   
 print(res3)

## $Difference  
## [1] 928.1  
##   
## $LSD  
## [1] 875.558930292  
##   
## $CohenD  
## [1] 0.718065555553  
##   
## $Significant  
## [1] TRUE

If you do this in SAS, package the results in a data table. Use the code for lsd\_test in Stat700Outline.sas as a template.

# Exercise 4 (R or SAS)

For the R portion, write a function/macro to compute upper and lower confidence intervals for a count mean , assuming the Poisson distribution, given (respectively) by

(Hint - homework 1 use the -distribution and the qt function. You’ll need to find the corresponding function for . In R, this function has parameters p ( or ) and df ( and ).

Let your function accept the required parameter x and an optional parameter alpha with a default value of 0.05. Return a list with values for Lower and Upper.

Name your function chisq.ci. Call your function with x=12 and x=13, and check your results against

# Function for chi^2 calcuations  
  
chisq.ci <- function(x,alpha=0.05) {  
p\_lower <- alpha/2  
p\_upper <- (1-alpha)/2  
df\_lower <- 2\*x  
df\_upper <- 2\*(x+1)  
local.chisqlower <- qchisq(p\_lower, df\_lower, ncp = 0, lower.tail = TRUE, log.p = FALSE)/2  
local.chisqupper <- qchisq(p\_upper, df\_upper, ncp = 0, lower.tail = TRUE, log.p = FALSE)  
return(list(chisqlower = local.chisqlower,chisqupper = local.chisqupper))  
}  
  
# Function with requested values  
chisq.ci(12,alpha = 0.05)

## $chisqlower  
## [1] 6.20057510872  
##   
## $chisqupper  
## [1] 24.8936780286

chisq.ci(13,alpha = 0.05)

## $chisqlower  
## [1] 6.921952491  
##   
## $chisqupper  
## [1] 26.8761310222

poisson.test(12,conf.level=0.95)

##   
## Exact Poisson test  
##   
## data: 12 time base: 1  
## number of events = 12, time base = 1, p-value = 8.31610743e-10  
## alternative hypothesis: true event rate is not equal to 1  
## 95 percent confidence interval:  
## 6.20057510872 20.96158504818  
## sample estimates:  
## event rate   
## 12

poisson.test(13,conf.level=0.95)

##   
## Exact Poisson test  
##   
## data: 13 time base: 1  
## number of events = 13, time base = 1, p-value = 6.35977733e-11  
## alternative hypothesis: true event rate is not equal to 1  
## 95 percent confidence interval:  
## 6.9219524910 22.2303959182  
## sample estimates:  
## event rate   
## 13

(This will not be unit tested, we will check visually)

If you choose SAS, you will need to write a macro that can be invoked in both PROC IML and in the DATA step. See the SAS template for instructions. You will still compare your results to the R code above.

# Exercise 5 (R or SAS)

Using the fisher.lsd function/macro as a template, write a function compute Tukey’s HSD. You will be expected to research this formula. Expect to make three changes 1. An extra parameter () for the number of means to be compared. Let this parameter default to 2. 2. A different distribution function (not qt) 3. A modification to the error term (but still use the pooled variance s2)

Document your changes relative to the LSD function, preferably in literate code (that is, typeset the formula, etc). Cite your references. You can, if you wish, compare your result with an example from your reference, as long as you document the source.

Calculate HSD using and values for the years 1936 and 2006, also using values for and ; compare this to Fisher LSD for the same years.

Name this function tukey.hsd.

# Exercise 6 (R or SAS)

Prepare a brief (~ 3-5 min.) video, literate document or user manual to guide a user through the steps required build or install an R library or SAS Macro library, using methods not covered in lecture. Some topcs include:

### a. Create a SAS Compiled Stored Macro (mstored) library.

### Use R CMD BUILD and INSTALL

I demonstrated in lecture video how to build and install a simple library using RStudio. Demonstrate how to do this on your system, without using RStudio; instead using R from the command line (i.e. ’R CMD BUILD JoyOfCooking`)

### c. devtools

For example, explain how this install command works, and what you need to use it.

devtools::install\_github("Rdatatable/data.table")

Upload video, Word or PDF document to Dropbox. Expect that these submissions will be made available to all students, so provide a brief title for your topic. You may choose a topic not on this list; it should be related to building or installing R or SAS libraries. Post a comment in the Functions and Macros Discussion giving a brief outline if you choose a different topic. You may also choose to do this in a group, but each member will be expected to provide at least 3 minutes of content. Provide an outline on the Discussion forum for this option.

Grading criteria will be similar to the criteria described for Discussions, listed in the course syllabus. If your initial submission does not score 9-10, I’ll give you feed back and allow you to make changes before posting to the rest of the class.

# Total points from Unit Tests

unit.test.points

## [1] 8