### t test code

library(shiny)

library(shinythemes)

# shinyUI(fluidPage(

ui <- fluidPage(

theme = shinytheme("united"),

titlePanel("Hypothesis Test (t.test : one, two)"),

br(),

br(),

br(),

sidebarPanel(

# SlidebarPanel for t-test

## 선택된 화면이 about 이면 의미, input.tabselected 와 같음

conditionalPanel(condition = "$('li.active a').first().html()==='About T-test'",

h2("About One Sample T-test:"),

p("A t-test is any statistical hypothesis test ~"),

p("it is most commonly applied ~"),

p("When the scaling term is unknown a~ "),

br(),

tags$a(href = "http://en.wikipedia.org/wiki/Student's\_t-test", "More Detail About t-test."),

br(),

br(),

sliderInput('range',

'The range you feel interest:',

min = -50,

max = 50,

value = c(-10,10)),

sliderInput('df',

'Degree of Freedom:',

min = 1,

max = 50,

value = 1)

),

# SlidebarPanel for file upload tab

conditionalPanel(condition = "$('li.active a').first().html()==='Data View'",

fileInput('file1', 'Choose CSV File',

accept=c('text/csv',

'text/comma-separated-values,text/plain',

'.csv')),

tags$hr(),

checkboxInput('header', 'Header', TRUE),

radioButtons('sep', 'Separator',

c(Comma=',',

Semicolon=';',

Tab='\t'),

','),

radioButtons('quote', 'Quote',

c(None='',

'Double Quote'='"',

'Single Quote'="'"),

'"')

),

# SliderbarPanel for t-test tab

conditionalPanel(condition = "$('li.active a').first().html()==='T-test'",

sliderInput("bins",

"Numer of bins:",

min = 1,

max = 50,

value = 2

),

radioButtons("sample",

"Please choose one sample t test or two sample t test:",

choices = c("One sample" = "oneSamp",

"Two sample" = "twoSamp")),

selectInput("var1",

label = "Please Select a Numerical Variable",

""

),

conditionalPanel(condition = "input.sample == 'twoSamp'",

selectInput("var2",

label = "Please Select a Numerical Variable",

""

),

radioButtons("varequal",

"Are the two samples have equal variance:",

choices = c("Yes" = "y",

"No" = "n"))

),

selectInput("tail",

label = "Please Select a relationship you want to test:",

choices = c("Equal" = "two.sided",

"Less" = "less",

"Greater" = "greater")),

conditionalPanel(condition = "input.sample == 'oneSamp'",

numericInput("test",

"Mean value You Want to Test",

value = 0

)

),

numericInput("conf",

label = "Please Select a confidence level:",

value = 0.95,

min = 0.8,

max = 0.99),

helpText("Note: Please assign a number between 0 and 1 in the numeric Input")

)

),

##

mainPanel(

tabsetPanel(

tabPanel('About T-test',

plotOutput('tplot')),

tabPanel('Data View',

fluidRow(column(10, offset = 1,

h2("Data Summary"),

verbatimTextOutput('disc'))),

fluidRow(column(10, offset = 1,

h2("Data Structure"),

verbatimTextOutput('str'))),

fluidRow(column(10, offset = 1,

h2("Data Table"),

tableOutput('contents')))

),

tabPanel('T-test',

fluidRow(column(10, offset = 1,

plotOutput('graph'))),

fluidRow(column(8, offset = 1,

h2("Key summary statistics"),

p("The observed sample statistics were:"),

tableOutput('parametric'),

h2("Hypothesis of the t-test"),

p("We are testing the null hypothesis that the mean of population equals to the value you set"),

p("The observed t test statistic :"),

textOutput('tvalue'),

p("A low P value suggests that your sample provides enough evidence that you can reject the null hypothesis for the entire population."),

textOutput('pvalue')))

)

)

)

)

###

# shinyServer(function(input, output, session) {

server <- function(input, output, session){

data <- reactive({

inFile <- input$file1

if (is.null(inFile)){return(NULL)}

read.csv(inFile$datapath, header=input$header, sep=input$sep,

quote=input$quote)

})

# Updata value user could select

observe({

updateSelectInput(

session,

"var1",

choices=names(data()))

})

# Updata value user could select

observe({

updateSelectInput(

session,

"var2",

choices=names(data()))

})

# Output a t distribution plot

output$tplot <- renderPlot({

# Display the Student's t distributions with various

# degrees of freedom and compare to the normal distribution

x <- seq(input$range[1], input$range[2], length=100)

hx <- dnorm(x)

labels <- c('t distribution', 'normal distribution')

plot(x, hx, type="l", lty=2, xlab="x value", col = "black",

ylab="Density", main="t Distributions")

lines(x, dt(x,input$df), lwd=2, col="red")

legend("topright", inset=.05, title="Distributions",

labels, lwd=2, lty=c(1, 2), col=c("red", "black"))

})

# Output a data table for the upload tab page

output$contents <- renderTable({

inFile <- input$file1

if (is.null(inFile))

return(NULL)

read.csv(inFile$datapath, header=input$header, sep=input$sep, quote=input$quote)

})

# Output a histogram for the variables user chose

output$graph <- renderPlot({

var1 <- data()[,input$var1]

var2 <- data()[,input$var2]

if (is.null(var1)){return(NULL)}

if (is.null(var2)){return(NULL)}

graph2 <- ifelse(input$sample == 'oneSamp', FALSE, TRUE)

p1 <- hist(var1, breaks = input$bins)

p2 <- hist(var2, breaks = input$bins)

plot(p1, col=rgb(0,0,1,1/4))

if(input$sample == 'twoSamp')

plot(p2, col=rgb(1,0,0,1/4),add = graph2)

})

# Output of discriptive summary of this variable

output$disc <- renderPrint({

Data <- data()

if (is.null(Data)){return(NULL)}

summary(Data)

})

# Output of the data structure

output$str <- renderPrint({

Data <- data()

if (is.null(Data)){return(NULL)}

str(Data)

})

# Create a one sample and two sample t-test reactive function

ttestout <- reactive({

var1 <- data()[,input$var1]

conf <- input$conf

if (is.null(var1)){return(NULL)}

t1 <- t.test(var1, alternative = input$tail, mu = input$test, conf.level = conf)

var2 <- data()[,input$var2]

if (is.null(var2)){return(NULL)}

ve <- ifelse(input$varequal == 'y', TRUE, FALSE)

t2 <- t.test(var1, var2, alternative = input$tail, var.equal = ve, conf.level = conf)

if(input$sample == "oneSamp") {return(t1)}

if(input$sample == "twoSamp") {return(t2)}

})

# Output of one sample t value of t-test

output$tvalue <- renderPrint({

vals <- ttestout()

if (is.null(vals)){return(NULL)}

vals$statistic

})

# Output of p value

output$pvalue <- renderPrint({

vals <- ttestout()

if (is.null(vals)){return(NULL)}

vals$p.value

})

# Output of key statistical parametric

output$parametric <- renderTable({

var1 <- data()[,input$var1]

if (is.null(var)){return(NULL)}

var2 <- data()[,input$var2]

if (is.null(var)){return(NULL)}

mean1 <- mean(var1)

mean2 <- mean(var2)

standard\_deviation1 <- sd(var1)

standard\_deviation2 <- sd(var2)

standard\_error1 <- sd(var1)/sqrt(length(var1))

standard\_error2 <- sd(var2)/sqrt(length(var2))

parametric1 <- data.frame(mean = mean1,

standard\_deviation=standard\_deviation1,

standard\_error=standard\_error1)

rownames(parametric1) <- input$var1

parametric2 <- data.frame(mean = mean2,

standard\_deviation=standard\_deviation2,

standard\_error=standard\_error2)

rownames(parametric2) <- input$var2

if(input$sample == "oneSamp") {return(parametric1)}

if(input$sample == "twoSamp") {return(rbind(parametric1,parametric2))}

})

}

## shinyApp

shinyApp(ui, server)