Week 4:

**Self learning R and R Shiny. There are some useful websites:**

1. <https://www.w3schools.com/r/>
2. <https://www.udemy.com/?utm_source=aff-campaign&utm_medium=udemyads&LSNPUBID=yNfEamYSgXk&ranMID=47901&ranEAID=yNfEamYSgXk&ranSiteID=yNfEamYSgXk-2z687JcYZK10rwH7Svufsw&gclid=EAIaIQobChMIzNvG-bTegwMVi-EWBR0cxAseEAAYAiAAEgKbNfD_BwE>
3. <https://shiny.posit.co/r/getstarted/shiny-basics/lesson1/index.html>

**What is R?**

R provides a wide variety of statistical (linear and nonlinear modelling, classical statistical tests, time-series analysis, classification, clustering, …) and graphical techniques, and is highly extensible. The S language is often the vehicle of choice for research in statistical methodology, and R provides an Open Source route to participation in that activity.

One of R’s strengths is the ease with which well-designed publication-quality plots can be produced, including mathematical symbols and formulae where needed. Great care has been taken over the defaults for the minor design choices in graphics, but the user retains full control.

R is available as Free Software under the terms of the [Free Software Foundation](http://www.gnu.org/)’s [GNU General Public License](https://www.r-project.org/COPYING) in source code form. It compiles and runs on a wide variety of UNIX platforms and similar systems (including FreeBSD and Linux), Windows and MacOS.

**What is R Shiny?**

Shiny is an open source R package that provides an elegant and powerful web framework for building web applications using R. Shiny helps you turn your analyses into interactive web applications without requiring HTML, CSS, or JavaScript knowledge.

Week 5:

Self learning Tableau. There are some useful website:

## **Controlling Reaction Example:**

### **ui.R**

```r

**library**(ggplot2)

**data**("mtcars")

xAxisChoices <- **colnames**(mtcars)

yAxisChoices <- **colnames**(mtcars)

cylinderChoices <- **unique**(mtcars$cyl)

fluidPage(

  selectInput(inputId = "xSelector", label = "Select the x axis", choices = xAxisChoices),

  selectInput(inputId = "ySelector", label = "Select the y axis", choices = yAxisChoices),

  selectInput(inputId = "cylSelector", label = "Select the cylinder", choices = cylinderChoices),

  actionButton(inputId = "refreshPlot", label = "refresh"),

  plotOutput("p1")

)

```

### **server.R**

```r

function(input, output) {

  filterData <- reactive({

    filteredData <- mtcars[mtcars$cyl == input$cylSelector,]

    return(filteredData)

  })

  plot1 <- eventReactive(input$refreshPlot, {

    ggplot(data = filterData(), aes\_string(x = input$xSelector, y = input$ySelector)) + geom\_point()

  })

  output$p1 <- renderPlot(plot1())

}

```

![Alt text](image-11.png)

## **3. Single Page Layout Example:**

### **ui.R**

```r

**library**(shiny)

airquality

**data**("airquality")

monthChoice <- **unique**(airquality$Month)

fluidPage(

  sidebarLayout(

    sidebarPanel(

      selectInput(inputId = "monthSelector", label = "Select a month", choices = monthChoice),

      selectInput(inputId = "colSelector", label = "Select a column to plot", choices = **c**("Temp", "Wind"))

    ),

    mainPanel(

      fluidRow(

        column(width = 6,

          DTOutput(outputId = "dt1")

        ),

        column(width = 6,

          plotOutput("plot1")

        )

      )

    )

  )

Findings: The result is the same as the results generated by the previous intern. The small proportion of the data extracted does not affect the accuracy.

Week 6:

Testing: Compare results between the patient’s data and the data generated by the previous intern.

library(ggplot2)

library(DT)

library(shiny)

# Define server logic required to draw a histogram

function(input, output) {

output$plot1 <- renderPlot(

ggplot(data = conditionData, aes\_string(x = input$rowSelector, y = input$colSelector))

+ geom\_point() + geom\_abline()

)

output$dt1 <- renderDT(

datatable(conditionData, options = list(pageLength = 5, lengthMenu = c(5, 10, 15)))

)

}

library(shiny)

conditionData <- read.csv("C:/Users/luyua/Desktop/conditions.csv")

colChoice <- colnames(conditionData)

rowChoice <- colnames(conditionData)

# Define UI for application that draws a histogram

fluidPage(

# Application title

titlePanel("Conditions"),

# Sidebar with a slider input for number of bins

sidebarLayout(

sidebarPanel(

selectInput(inputId = "colSelector", label = "Select a column", choices = colChoice),

selectInput(inputId = "rowSelector", label = "Select a row", choices = rowChoice)

),

# Show a plot of the generated distribution

mainPanel(

plotOutput("plot1"),

DTOutput("dt1")

)

)

)