MTH208: Worksheet 14

Building R Shiny Apps

We will finally make some Shiny Apps! The idea of a Shiny App is to make an interactive environment for analyzing a dataset or certain concepts.

A Shiny application is divided into two parts: the **User Interface (UI)** and the **Server**. The UI contains the way the app looks and its presentation. The server is responsible for the logical flow of the app, and the core functions. The Server also controls the data that will be displayed.

To start a new R Shiny App, choose "Shiny Web App" in the upper left corner. This creates a default folder with a single file called `app.R`. There are two aspects: `ui` and `server()`:

```r
library(shiny)

# Define UI for application that draws a histogram
ui <- fluidPage(

    # Application title
    titlePanel("Old Faithful Geyser Data"),

    # Sidebar with a slider input for number of bins
    sidebarLayout(
        sidebarPanel(
            sliderInput("bins",
                        "Number of bins:",
                        min = 1,
                        max = 50,
                        value = 30)
        ),

        # Show a plot of the generated distribution
        mainPanel(
           plotOutput("distPlot")
        )
    )
)
```

`ui` saves the characteristics of the visual of the shiny app.

- The style being used is `fluidPage()` which turns the page into column-panels and rows.

- `titlePanel()` sets the main title of the ShinyApp.

- `sidebarLayout()` starts how the sidebar looks.

  – In that, there is a `sidebarPanel()` which has a `sliderInput()`.

  – The slider is tagged as `bins`, has the name "Number of bins". The `min` value is 1, the `max` value is 50 and the default value is 30.

- The `mainPanel()` describes what the main panel will look like

  – It has a `plotOutput` that has the plot `distPlot`, this `distPlot` is plotted using the `server()` function.

The `server()` function looks like the following:

```
# Define server logic required to draw a histogram
server <- function(input, output) {

    output$distPlot <- renderPlot({
        # generate bins based on input$bins from ui.R
        x    <- faithful[, 2]
        bins <- seq(min(x), max(x), length.out = input$bins + 1)

        # draw the histogram with the specified number of bins
        hist(x, breaks = bins, col = 'darkgray', border = 'white',
             xlab = 'Waiting time to next eruption (in mins)',
             main = 'Histogram of waiting times')
    })
}
```

- `input` contains all the input parameters gathered from the UI interface.

- `output` has the output components that will go into the `mainPanel()`.

Go ahead and click on `Run App` to see how the page looks.

Questions

1. In the GitHub repository, there is another folder: `DataApp` folder, which has another `app.R`. Run that app and study the code carefully.

2. To the previous code, add a fourth dataset to the choice: "faithful".

3. Add a third row panel to the above app which makes a scatterplot using `plot()` for the chosen dataset.

4. Replace the above plot with a plot made with `ggplot2`.