Challenge-8

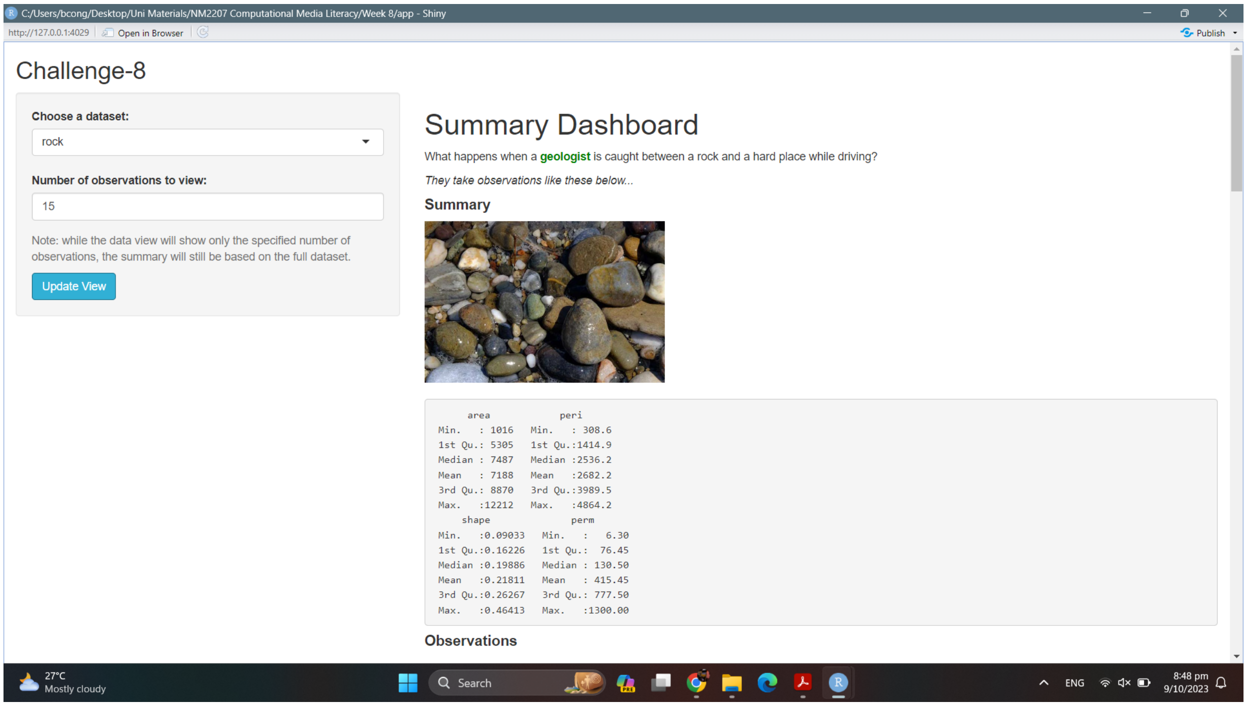
Lew Bo Cong

2023-10-09

The Challenge

Explain my customisations

I included a title ("Summary Dashboard") and a simple background information regarding the datasets by giving it a "storyline". I also made the "Summary" and "Observations" headings bold, and changed the colour of the "Update View" button to make it more obvious that the user needs to click on it to update the view. I set the default observation view number to 15.



C:/Users/bcong/Desktop/Uni Materials/NM2207 Computational Media Literacy/Week 8/app - Shiny

http://127.0.0.1:4029

Challenge-8

Choose a dataset: rock

Number of observations to view: 15

Note: while the data view will show only the specified number of observations, the summary will still be based on the full dataset.

Summary Dashboard

What happens when a **geologist** is caught between a rock and a hard place while driving?

They take observations like these below...

Summary

```
     area            peri
 Min.   : 1016   Min.   : 308.6
 1st Qu.: 5305   1st Qu.:1414.9
 Median : 7487   Median :2536.2
 Mean   : 7188   Mean   :2682.2
 3rd Qu.: 8870   3rd Qu.:3989.5
 Max.   :12212   Max.   :4864.2
     shape              perm
 Min.   :0.09033   Min.   :   6.30
 1st Qu.:0.16226   1st Qu.:  76.45
 Median :0.19886   Median : 130.50
 Mean   :0.21811   Mean   : 415.45
 3rd Qu.:0.26267   3rd Qu.: 777.50
 Max.   :0.46413   Max.   :1300.00
```

Observations

Figure 1: Screenshot of my Shiny App

```r
library(shiny)

# Define UI for dataset viewer app ----
ui <- fluidPage(

  # App title ----
  titlePanel("Challenge-8"),
# Sidebar layout with input and output definitions ----
sidebarLayout(

  # Sidebar panel for inputs ----
  sidebarPanel(

    # Input: Select a dataset ----
    selectInput("dataset", "Choose a dataset:",
                choices = c("rock", "pressure", "cars")),

    # Input: Specify the number of observations to view ----
    numericInput("obs", "Number of observations to view:", 15),

    # Include clarifying text ----
    helpText("Note: while the data view will show only the specified",
             "number of observations, the summary will still be based",
             "on the full dataset."),

    # Input: actionButton() to defer the rendering of output ----
    # until the user explicitly clicks the button (rather than
    # doing it immediately when inputs change). This is useful if
    # the computations required to render output are inordinately
    # time-consuming.
    actionButton("update", "Update View", class = "btn-info")

  ),

  # Main panel for displaying outputs ----
  mainPanel(

    h1("Summary Dashboard"),
    p("What happens when a",
      strong(span("geologist", style = "color: green")),
      "is caught between a rock and a hard place while driving?"),
    em("They take observations like these below..."),
    br(),

    # img(src = "imagerock.jpg", height = 200),
    # img(src = "imagepressure.jpg", height = 200),
    # img(src = "imagecars.jpg", height = 200),

    # Output: Header + summary of distribution ----
    h4(strong("Summary")),
    imageOutput("images", height = 200),
    br(),
    verbatimTextOutput("summary"),

    # Output: Header + table of distribution ----
    h4(strong("Observations")),
    tableOutput("view")
  )

)
```

```r
)

# Define server logic to summarize and view selected dataset ----
server <- function(input, output) {

  # Return the requested dataset ----
  # Note that we use eventReactive() here, which depends on
  # input$update (the action button), so that the output is only
  # updated when the user clicks the button
  datasetInput <- eventReactive(input$update, {
    switch(input$dataset,
           "rock" = rock,
           "pressure" = pressure,
           "cars" = cars)
  }, ignoreNULL = FALSE)

  # Generate a summary of the dataset ----
  output$summary <- renderPrint({
    dataset <- datasetInput()
    summary(dataset)
  })

  output$images <- renderImage({
    filename <- normalizePath(file.path('./www', paste('image', input$dataset, '.jpg', sep='')))
    list(src = filename, height = 200)
  }, deleteFile = FALSE)

  # Show the first "n" observations ----
  # The use of isolate() is necessary because we don't want the table
  # to update whenever input$obs changes (only when the user clicks
  # the action button)
  output$view <- renderTable({
    head(datasetInput(), n = isolate(input$obs))
  })

}

# Create Shiny app ----
shinyApp(ui, server)
```