Quote with ChatGPT

Julien LEBRANCHU

4/2/24

First test

Write a dashboard shiny which present the iris dataset with table and charts

```
library(shiny)
library(ggplot2)
library(dplyr)
library(plotly)

# Load the iris dataset
data(iris)

# Define UI
ui <- fluidPage(
  titlePanel("Iris Dataset"),
  sidebarLayout(
    sidebarPanel(
      selectInput("xcol", "X-Axis Variable", names(iris)),
      selectInput("ycol", "Y-Axis Variable", names(iris),
                  selected = names(iris)[[2]]),
      radioButtons("plottype", "Plot Type",
                   c("Scatter Plot" = "scatter",
                     "Box Plot" = "box"))
    ),
    mainPanel(
      tabsetPanel(
        tabPanel("Table", dataTableOutput("table")),
        tabPanel("Plot", plotlyOutput("plot"))
      )
    )
```

```
  )
)

# Define server logic
server <- function(input, output) {

  # Create a reactive subset of the iris dataset based on user inputs
  iris_subset <- reactive({
    iris %>%
      select(input$xcol, input$ycol)
  })

  # Render the table
  output$table <- renderDataTable({
    iris_subset()
  })

  # Render the plot
  output$plot <- renderPlotly({
    if (input$plottype == "scatter") {
      p <- ggplot(iris_subset(), aes_string(x = input$xcol, y = input$ycol)) +
        geom_point()
    } else {
      p <- ggplot(iris_subset(), aes_string(x = input$xcol, y = input$ycol)) +
        geom_boxplot()
    }
    ggplotly(p)
  })

}

# Run the app
shinyApp(ui = ui, server = server)
```

Second test

Write a dashboard shiny with multiple tab pane. One the first tab, it present the iris dataset with table and charts. The second tab present the storms dataset with the table, a map and chart for 2015

```r
library(shiny)
library(ggplot2)
library(leaflet)

# Load the iris dataset
data(iris)

# Load the storms dataset
data(storms)

# Define UI
ui <- fluidPage(
  titlePanel("Iris and Storms Data Dashboard"),
  sidebarLayout(
    sidebarPanel(
      # Tab Selector
      tabsetPanel(
        tabPanel("Iris Dataset",
                 h4("Iris Data"),
                 dataTableOutput("iris_table"),
                 h4("Iris Sepal Length"),
                 plotOutput("iris_plot1"),
                 h4("Iris Sepal Width"),
                 plotOutput("iris_plot2")
                 ),
        tabPanel("Storms Dataset",
                 h4("Storms Data"),
                 dataTableOutput("storms_table"),
                 h4("Storms Map"),
                 leafletOutput("storms_map"),
                 h4("Storms Events in 2015"),
                 plotOutput("storms_plot")
                 )
      )
    ),
    mainPanel()
  )
)

# Define Server
server <- function(input, output) {
```

```r
  # Iris Dataset
  output$iris_table <- renderDataTable({
    iris
  })

  output$iris_plot1 <- renderPlot({
    ggplot(iris, aes(x = Sepal.Length)) +
      geom_histogram(fill = "blue", bins = 30) +
      labs(title = "Distribution of Sepal Length")
  })

  output$iris_plot2 <- renderPlot({
    ggplot(iris, aes(x = Sepal.Width)) +
      geom_histogram(fill = "red", bins = 30) +
      labs(title = "Distribution of Sepal Width")
  })

  # Storms Dataset
  output$storms_table <- renderDataTable({
    storms
  })

  output$storms_map <- renderLeaflet({
    leaflet(storms) %>%
      addTiles() %>%
      addMarkers(~longitude, ~latitude)
  })

  output$storms_plot <- renderPlot({
    storms_2015 <- storms[year(storms$date) == 2015, ]
    ggplot(storms_2015, aes(x = factor(status))) +
      geom_bar(fill = "green") +
      labs(title = "Storms Events in 2015")
  })

}

# Run the app
shinyApp(ui, server)
```

Third test

Write a dashboard shiny with multiple tab pane. One the first tab, it present the iris dataset with table and charts. The second tab present the storms dataset with the table, a map and chart for 2015. To create the map and chart, you need to use the plotly library

```
library(shiny)
library(ggplot2)
library(plotly)

# Load datasets
data(iris)
data(storms)

# Define UI
ui <- fluidPage(
  titlePanel("Multi-tab dashboard"),
  sidebarLayout(
    sidebarPanel(
      h3("Choose a dataset:"),
      radioButtons("dataset", "Dataset:",
                   choices = c("Iris", "Storms"), selected = "Iris")
    ),
    mainPanel(
      tabsetPanel(
        tabPanel("Dataset summary",
          tableOutput("summary"),
          plotOutput("histogram")
        ),
        tabPanel("Storms dataset",
          tabsetPanel(
            tabPanel("2015",
              tableOutput("storm_table_2015"),
              plotOutput("storm_map_2015"),
              plotlyOutput("storm_chart_2015")
            )
          )
        )
      )
    )
  )
```

```
)

# Define server
server <- function(input, output) {

  # Create summary table and histogram for iris dataset
  output$summary <- renderTable({
    if (input$dataset == "Iris") {
      summary(iris)
    } else {
      summary(storms)
    }
  })

  output$histogram <- renderPlot({
    if (input$dataset == "Iris") {
      ggplot(iris, aes(x = Sepal.Length)) + geom_histogram()
    } else {
      ggplot(storms, aes(x = DATE)) + geom_histogram() + xlab("Date")
    }
  })

  # Create table, map, and chart for storms dataset for 2015
  output$storm_table_2015 <- renderTable({
    subset(storms, YEAR == 2015)
  })

  output$storm_map_2015 <- renderPlot({
    # Create map using plotly
    if (!is.null(input$dataset) && input$dataset == "Storms") {
      p <- ggplot(storms, aes(x = -LONGITUDE, y = LATITUDE)) +
        geom_point(aes(color = CATEGORY)) +
        scale_color_gradient(low = "green", high = "red") +
        labs(x = "Longitude", y = "Latitude", title = "Storms in 2015")
      ggplotly(p)
    }
  })

  output$storm_chart_2015 <- renderPlotly({
    # Create chart using plotly
    if (!is.null(input$dataset) && input$dataset == "Storms") {
```

```
      storms_2015 <- subset(storms, YEAR == 2015)
      p <- ggplot(storms_2015, aes(x = as.factor(MONTH_NAME), fill = as.factor(CATEGORY)))
        geom_bar(position = "dodge") +
        labs(x = "Month", y = "Count", title = "Storms in 2015 by category")
      plotly_build(p)
    }
  })
}

# Run app
shinyApp(ui, server)
```

Fouth test

I start a discussion with ChatGPT to improve the code.