Data Visualization

Module 5: Assignment 5: Static Visuals to Dashboards

Jesse Conlon

2/23/20

Assignment 5: Static Visuals to Dashboards

There were several goals to this assignment:

- Create a shiny dashboard that shows your first 3 graphs from last week
    - All three graphs were displayed from Assignment 4, a few small changes were made such as using a universal data set to keep the filtering consistent.
- Allow the user to filter by price to change all the graphs.
    - A sliding price filter was added to the graph that allows the user to filter out prices below the selected value.
- Use an additional input type to modify your graph. Either filter the data again, or change the graph itself
    - For the purpose of practicing and gaining familiarity I added a data table, a radio button option, checkboxes, and a slider.
- Extra credit: Put graphs on separate tabs but keep the inputs consistent (5 Points)
    - I was able to look up how to place the graphs on separate tabs and fulfilled this.
- Please deploy via: shinyapps.io/
    - https://jesse-conlon.shinyapps.io/Assignment_05/
    - Here is the link that displays my app, posted below is my code for the app.

```
## UI
ui <- fluidPage(

  titlePanel("King County House Prices"),
  sidebarLayout(
    sidebarPanel(
      radioButtons(inputId = "waterfront",
                   label = "Waterfront Property",
                   choiceNames = c("Yes", "No"),
                   choiceValues = c(1, 0),
                   selected = 1),
      sliderInput("price_range2", "Price Range",
                  value = c(83000, 7060000), min = 83000, max = 7060000, pre = "$"),
      checkboxGroupInput(inputId = "decade",
                         label ="Decade:",
                         choiceNames = c("1900", "1910", "1920", "1930", "1940", "1950",
                                         "1960", "1970", "1980", "1990", "2000", "2010"),
                         choiceValues = c("1900", "1910", "1920", "1930", "1940", "1950",
                                          "1960", "1970", "1980", "1990", "2000", "2010"),
                         selected = c("1900", "1910", "1920", "1930", "1940", "1950",
                                      "1960", "1970", "1980", "1990", "2000", "2010")),
      dataTableOutput("pricetable")
    ),
    mainPanel(
      tabsetPanel(
        tabPanel("Median Price Based on Year Built",
                 plotOutput("pricePlot1")),
        tabPanel("Zipcode Price Median Values",
                 plotOutput("pricePlot2")),
```

```
            tabPanel("Median Price Based on Year Built and Location",
                     plotOutput("pricePlot3"))
          )
        ),


    )
)
```

```
## Server

library(shiny)
library(tidyverse)
library(plotly)
library(DT)
library(maps)
library(sf)
library(scales)
library(purrr)
library(rsconnect)

## Data Sets
king <- read_csv("KING COUNTY House Data.csv")
king_mod <- king %>%
   group_by(yr_built, decade) %>%
   summarize(price = median(price))


geo_price <- king %>%
   group_by(zipcode, decade, yr_built, waterfront) %>%
   summarize(price = median(price), lat = median(lat), long = median(long))

countries <- st_as_sf(maps::map('world', plot = FALSE, fill = TRUE))

states <- st_as_sf(map("state", plot = FALSE, fill = TRUE))

counties <- st_as_sf(map("county", plot = FALSE, fill = TRUE))

counties_wa <-counties %>%
   filter(str_detect(ID, 'washington,'))

server <- function(input, output) {

   plot_table <- reactive({
      king_mods <- king %>%
         select(yr_built, price)
   })

   plot_data <- reactive({
      geo_price  %>%
         filter(decade %in% input$decade) %>%
         filter(price >= input$price_range2[1] & price <= input$price_range2[2]) %>%
         filter(waterfront %in% input$waterfront)
   })

   plot_data2 <- reactive({
      geo_price  %>%
```

```
    filter(decade %in% input$decade) %>%
    filter(price >= input$price_range2[1] & price <= input$price_range2[2]) %>%
    filter(waterfront %in% input$waterfront)
})

plot_data3 <- reactive({
  geo_price  %>%
    filter(decade %in% input$decade) %>%
    filter(price >= input$price_range2[1] & price <= input$price_range2[2]) %>%
    filter(waterfront %in% input$waterfront)
})

## Visual 1

output$pricePlot1 <- renderPlot({
  ggplot(data = plot_data(), aes_string(x = "yr_built", y = "price", color = "decade"), alpha = 0.5) +
    scale_colour_viridis_c("Decade") +
    labs(x = "Year Built",
         y = "Median Price of House",
         title = "Price of House Based on Year Built",
         subtitle = "Created for Data Vis Class",
         caption = "Source: King County House Data") +
    scale_x_continuous(breaks = seq(1900,2015, by = 10)) +
    scale_y_continuous(labels = dollar) +
    geom_jitter(size = 2) + geom_smooth()

})

## Visual 2

output$pricePlot2 <- renderPlot({

  theme_set(theme_minimal())
  counties_wa <-counties %>%
    filter(str_detect(ID, 'washington,'))

  counties_wa %>%
    filter(str_detect(ID, "king|kitsap"))%>%
    ggplot() +
    geom_sf() +
    geom_point(data = plot_data2(), aes(x = long, y = lat, color = decade, size = price), alpha = 0.5) +
    scale_colour_viridis_c("Decade") +
    scale_size(name = "Sale Price", labels = dollar) +
    labs(x = "Longitude",
         y = "Latitude",
         title = "Price of House Based on Zipcode",
         subtitle = "Created for Data Vis Class",
         caption = "Source: King County House Data")
```

```
  })

  ## Visual 3
  output$pricePlot3 <- renderPlot({

    counties_wa %>%
      filter(str_detect(ID, "king|kitsap")) %>%
      ggplot() +
      geom_sf() +
      geom_point(data = plot_data3(), aes(x = long, y = lat, color = price)) +
      scale_colour_viridis_c("Sale Price", labels = dollar) +
      facet_wrap(~decade) +
      labs(x = "Longitude",
          y = "Latitude",
          title = "Price of House Based on Time and Space",
          subtitle = "Created for Data Vis Class",
          caption = "Source: King County House Data") +
      theme(axis.text.x = element_text(angle=90))
  })

  ## Interactive Table

  output$pricetable <- renderDataTable({
    plot_table()
  })
}
```