Rshiny

John Brandt

3/15/2018

Packages & Data

```
library(shiny)
library(ggvis)
library(tibble)
library(magittr)

data <- mtcars
colnames(data) <- c("MPG", "Number_cylinders",
      "Displacement", "Horsepower",
      "Rear_axle_ratio", "Weight_1000_lb",
      "Quarter_m_time", "Engine_shape", "Transmission",
      "Forward_gears", "Carburetors")
```

Process data

```
data$Transmission[data$Transmission == 0 ] <- "Automatic"
data$Transmission[data$Transmission == 1] <- "Manual"
data$Transmission <- factor(data$Transmission)

data$Engine_shape[data$Engine_shape == 0] <- "V-shape"
data$Engine_shape[data$Engine_shape == 1] <- "S-shape"
data$Engine_shape <- factor(data$Engine_shape)

data$Forward_gears <- factor(data$Forward_gears)
data$Carburetors <- factor(data$Carburetors)

data$Number_cylinders <- factor(data$Number_cylinders)
```

Subset data for varible selections

```
axis_vars <- c("MPG", "Displacement", "Horsepower",
               "Rear_axle_ratio", "Weight_1000_lb",
               "Quarter_m_time")

factor_vars <- c("Number_cylinders", "Transmission",
                 "Forward_gears", "Carburetors",
                 "Engine_shape")

data <- tibble::rownames_to_column(data, var="Model")
```

Rshiny basics

- UI & Server
- fluidRow - rows that expand to monitor/screen size
- Columns, relative sizing
- https://shiny.rstudio.com/gallery/widget-gallery.html
  - Many widgets
  - selectInput
  - sliderInput

selectInput

```
selectInput("reference.name", "label", choices,
            selected = "Default")
```

Now, in the server, `input$reference.name` will interactively update to the input

Create UI

```
ui <- fluidPage(
  titlePanel("Yale NUS"),

  fluidRow(
    column(3,
           ## Next Slide
    ),
```

Input

```
helpText("Explore relationships between
    fuel efficiency and vehicle properties"),
selectInput("yvar", "Y-axis variable", axis_vars,
    selected = "Horsepower"),
selectInput("xvar", "X-axis variable", axis_vars,
    selected = "MPG"),
selectInput("lmodel", "Linear model variable",
            factor_vars,
    selected = "Number_cylinders"),
sliderInput("range", label= "Horsepower:",min = 0,
            max = 350, value = c(0,350))
```

Column (75% of space) with visualization

```
    column(9,
           ggvisOutput("plot1")
    )
  )
)
```

Create server

Server must have input, output

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

Create interactive tooltip

```
car_tooltip <- function(x) {
  if (is.null(x)) return(NULL)
  # Pick out the car with this model
  column <- yvar$value
  row <- xvar$value
  if(is.null(x[[column]])) return(NULL)
  car <- data[data[[column]] == x[[column]] &
                data[[row]] == x[[row]],]
  paste0("<b>", as.character(car$Model), "</b><br>")
}
```

Add reactivity to visualization

```
vis <- reactive({
  # Labels for axes
  xvar_name <- names(axis_vars)[axis_vars == input$xvar]
  yvar_name <- names(axis_vars)[axis_vars == input$yvar]
  xvar <<- prop("x", as.symbol(input$xvar))
  yvar <<- prop("y", as.symbol(input$yvar))
  factorvar <<- prop("factor", as.symbol(input$lmodel))
```

Update the dataframe used & create ggvis

```
    data <- data[data$Horsepower >= input$range[1],]
    data <- data[data$Horsepower <= input$range[2],]

    data %>%
      group_by(data[[factorvar$value]]) %>%
      ggvis(xvar, yvar) %>%
      layer_model_predictions(stroke=
                                factorvar$value,
                              model="lm") %>%
      layer_points(fill = factorvar$value) %>%
      add_tooltip(car_tooltip, on="hover") %>%
      add_axis("x", title = xvar_name) %>%
      add_axis("y", title = yvar_name) %>%
      set_options(width = 1000, height = 700) %>%
      add_legend("stroke")
  })
```

Add the visualization

```
  vis %>% bind_shiny("plot1")
}
```

Run the app

```
# Run the app ----
shinyApp(ui = ui, server = server, options =
           list(height= 700))
```