Interactive visualization:

R Shiny

Weiai Xu (Wayne), PhD
Assistant Professor
Department of Communication, UMass-Amherst
Email: weiaixu@umass.edu
curiositybits.cc
Examples of R Shiny apps

Introducing Engagement Indicators

Here we quantify various forms of Twitter engagement practice used by the candidates. We also track the effectiveness of their engagement. The analysis below is based on the candidates’ original tweets (i.e., excluding their retweets of other accounts) since October 1, 2018. See the About page for the information about how each indicator is calculated.

https://curiositybits.shinyapps.io/PH_Tracker_dashboard/
Components in a R Shiny app

Sidebar Panel: contains one or multiple control widgets

Allow users to make selection or enter values
Components in a R Shiny app

Basic widgets available in R Shiny

Components in a R Shiny app

**Main Panel:** contains visual outputs

Outputs change in response to user selection
How does a R Shiny app work

1. When a user makes a selection in the Sidebar Panel, it creates an input value.

2. The input value is used to select cases for visualization.
What files does a R Shiny app consist of

Each app has a stand-alone folder

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app.R</td>
<td>app.R is the app script</td>
</tr>
<tr>
<td>geocodes.csv</td>
<td>Data files</td>
</tr>
<tr>
<td>tweets.csv</td>
<td>Data files</td>
</tr>
<tr>
<td>workflow.R</td>
<td>Backend code for pre-processing data</td>
</tr>
</tbody>
</table>
Create an example

Before you create an app, make sure the shiny library is installed and loaded
Test example files

Example files are available on Moodle

Create a folder in your laptop and move app.R and associated data files to the folder. Then open app.R in RStudio.

Make sure the example app works on your laptop
What’s in app.R?

The **ui** part and the **server** part

```r
library(shiny)
library(rCharts)
library(lubridate)
library(highcharter)

# UI
ui <- fluidPage(
  # Application title
titlePanel("Your app title"),

  p(
    class = "text-muted",
    paste("Demo: you can enter text here."
  ),

  sidebarLayout(
    sidebarPanel(
      helpText(h5("based on sentiment type"), style = "font-family: 'arial'; font-size:12pt"),
      checkboxGroupInput("type"),

      # SERVER
      server <- function(input, output) {
        senti_data <- read.csv("senti_aggregated.csv", header = TRUE)
        geocodes <- read.csv("geocodes.csv", header = TRUE)
        senti_data$day <- as.Date(senti_data$day)
        output$chart1 <- renderHighchart({
          highchart() %>%
          hc_add_series(data = senti_data[senti_data$variable %in% input$type], "line", hcaes(x =
          hc_xAxis(type = "datetime")
        })
        output$mymap <- renderLeaflet({
          usericon <- makeIcon()
          iconUrl = geocodes$profile_image_url,
          iconWidth = 15, iconHeight = 15
        })
      }]
```
What’s in app.R?

The **ui** part

```r
library(shiny)
library(rCharts)
library(lubridate)
library(highcharter)

# UI
ui <- fluidPage(

  titlePanel("Your app title")

  p(
    class = "text-muted",
    paste("Demo: you can enter text here.",

  )

  sidebarLayout(
    sidebarPanel(
      helpText(h5("based on sentiment type"), style = "font-family: 'arial'; font-size:12pt"),
      checkboxGroupInput("type",

```

**Q:** What should I do if I want to change the app name to *Class Demo* and add an introduction?
What’s in app.R?

The **ui** part

```r
sidebarLayout(
  sidebarPanel(
    helpText(h5("based on sentiment type"), style = "font-family: 'arial'; font-size: 12pt")),
    checkboxGroupInput("type",
      label = (helpText(h5("sentiment type"))),
      choices = list("anger" = "anger",
                      "anticipation" = "anticipation",
                      "disgust" = "disgust",
                      "fear" = "fear",
                      "joy" = "joy",
                      "sadness" = "sadness",
                      "surprise" = "surprise",
                      "trust" = "trust"),
      selected = "trust")
  ),
)
```

A list of choices available to users

When a user selects anger, the selection creates an input value. The value is the string **anger**, the input value is stored as `input$type`
What’s in app.R?

The **ui** part

```r
36 sliderInput("slider1", h3("min. retweet count"),
37   min = 0, max = 100, value = 1),
38
39 p(
40   class = "text-muted",
41   paste("Demo: you can enter text here.
42   
43   
44   
45   
46   
```

Every time a user sets on a number, it creates an input value. The input value is stored as `input$slider1`
What's in app.R?

The **ui** part

```r
mainPanel(
    highchartOutput("chart1", height = "500px"),

    leafletOutput("mymap"),

    p(class = "text-muted", paste("Demo: you can enter text here.")))
What’s in app.R?

The **server** part

Do you remember the two input values created from user selection?

`input$type`

`input$slider1`
What’s in app.R?

The server part: import data, process data, and create visualization

```r
# SERVER
server <- function(input, output) {

senti_data <- read.csv("senti_aggregated.csv", header = TRUE)
geocodes <- read.csv("geocodes.csv", header = TRUE)

senti_data$day <- as.Date(senti_data$day)

output$chart1 <- renderHighchart({
  highchart() %>%
  hc_add_series(data = senti_data[senti_data$variable %in% input$type], "line", hcaes(x = day, y = value, group=variable)) %>%
  hc_xAxis(type = "datetime")
})
```

Import the two data files from your app folder.

Clean and standardize the data if necessary.

The part where you create a highchart showing sentiment trends.
What's in app.R?

```R
function(input, output) {
  data <- read.csv("senti_aggregated.csv", header = TRUE)
  codes <- read.csv("geocodes.csv", header = TRUE)
  data$day <- as.Date(senti_data$day)
  output$chart1 <- renderHighchart({
    chart() %>%
      add_series(data = senti_data[senti_data$variable %in% input$type], "line", hcaes(x = day, y = value, group=variable, 
      xAxis(type = "datetime")

  Use input$type as the filtering criteria to select cases from senti_data
```
What’s in app.R?

The server part: import data, process data, and create visualization

```r
output$mymap <- renderLeaflet({

  usericon <- makeIcon(
    iconUrl = geocodes$profile_image_url,
    iconWidth = 15, iconHeight = 15
  )

  leaflet(data = geocodes[geocodes$retweet_count >= input$slider1,]) %>%
    addTiles() %>%
    setView(lng = -98.35, lat = 39.50, zoom = 2) %>%
    addMarkers(lng = ~lng, lat = ~lat, popup = ~ as.character(text), icon = usericon) %>%
    addProviderTiles("Stamen.TonerLite") %>% # more layers: http://leaflet-extras.github.io/leaflet-
    addCircleMarkers(
      stroke = FALSE, fillOpacity = 0.5
    )
})
```

Use `input$slider1` as the filtering criteria to select cases from `geocodes`
In-class practice

Make sure the example app works on your laptop
See if you can add a new slider bar with the input value named “slider3” and the help text “min. Favorite count.”
Publish a R Shiny app

https://shiny.rstudio.com/articles/shinyapps.html
More resources

https://shiny.rstudio.com/articles/basics.html
Other visualization options

Using R Markdown: [https://rmarkdown.rstudio.com/](https://rmarkdown.rstudio.com/)