While we wait, please take the poll & download or clone the repo

Poll: https://forms.gle/Tm6mvEkgtWCEXV4y5

Exercises:

https://github.com/anastasia-lucas/shinydash-rladies-dc

- ○ Exercises in exercises/
- ○ Solutions in solutions/
- ○ Make sure packages in utils.R are installed!
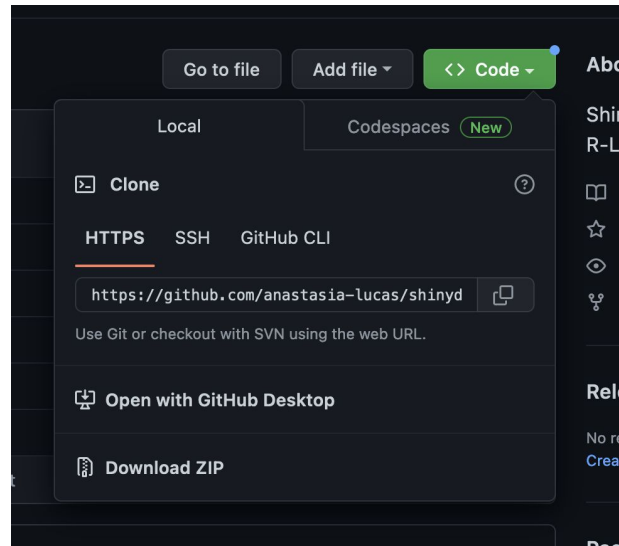
Quick about me
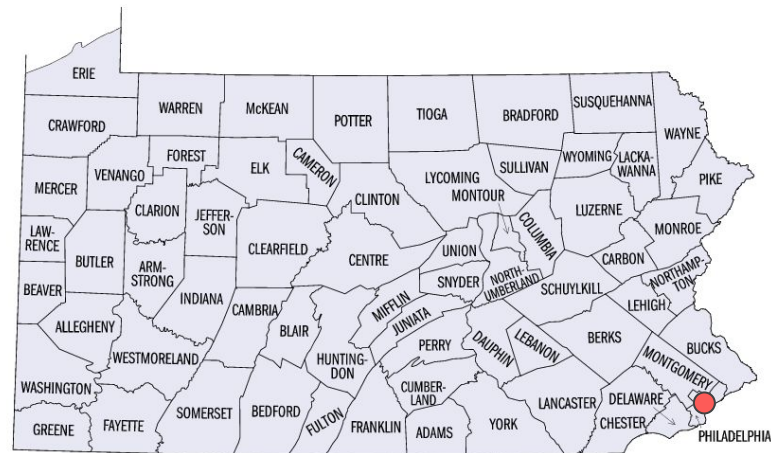
- B.S. in Biostatistics

- Genetics data analysis

- EHR & genetics data integration

- EHR & genetics data integration

- Immunotherapy clinical & multi-omics data integration

- 1st year PhD student in Genomics & Computational Biology

Age Distribution

Self-Reported Ancestry

Cohort Selection

Select entire Penn Medicine BioBank or patients with genetic data

PMBB ▾

Quick

er by ICD-9 Code

codes were mapped to ICD-9 using CMS General Equivalence Mappings
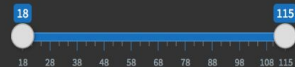
ll ▾

mber of Patients per Code

☰   Penn Medicine BioBank COVID-19 Dashboard

COVID-19 Positive Cohort

Penn Medicine EHR ▾

☐ Include Only Emergency Admissions

☐ Include Only Patients With Residuals

Age at COVID-19 Test

18   115

18   28   38   48   58   68   78   88   98   108 115

Reported Gender

☑ Female

☑ Male

☑ X

Reported Race

Black   White   Asian   Other   Unknown

American Indian   East Indian

Pacific Island

Pre-existing Condition

No filter ▾

Information About the Data Displayed   —

This dashboard was last updated on 2021-05-13.

Total RT-PCR Tested Patients include the total number of patients tested in the COVID-19 registry with positive or negative COVID-19 status.

Cases include the total number of patients with a positive RT-PCR test in the COVID-19 registry or with any instance of ICD-10 code U07.1 in their health record.

Hospitalizations include patients who were listed as having a positive RT-PCR test result within 21 days of their hospital admission date, having either an instance of the ICD-10 code U07.1 or a respiratory code determined by a clinician as having a high probability of being COVID-19 related, and having an inpatient status. In brief, these are patients hospitalized for, not with, COVID-19.

ICU Stays include the total of patients determined to have been a COVID-19 hospitalization and had an indication of ICU stay during their admission, with or without vent and/or oxygen use.

Deaths include the number of patients who were determined to have been a COVID-19 hospitalization and had a death dated listed during their admission.

This dashboard was developed by Anastasia Lucas and Anurag Verma. For additional information please contact anastasia.lucas@pennmedicine.upenn.edu or anurag.verma@pennmedicine.upenn.edu

PMBB | Penn Medicine BioBank

nical Labs

rrently data on 37 labs are available

1C ▾
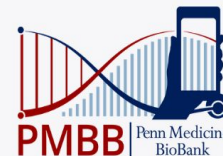
Total RT-PCR Tested Patients in COVID-19 Registry

Cases

Hospitalizations

ICU Stays

Hospital Deaths

By Testing Site   By County of Residence

RT-PCR Confirmed COVID-19 Positives (Overall COVID-19 Registry)

Number of Patients Per COVID-19 Outcome

CUMBER- LAND   LANCASTER   DELAWARE

ADAMS   YORK   CHESTER   PHILADELPHIA

Before we get to Shiny dashboards, what is a Shiny app?

https://shiny.rstudio.com/

- interactive web app
- built on HTML

Before we get to Shiny dashboards, what is a Shiny app?

https://shiny.rstudio.com/

interactive web app

slider user can interact with

Before we get to Shiny dashboards, what is a Shiny app?

https://shiny.rstudio.com/
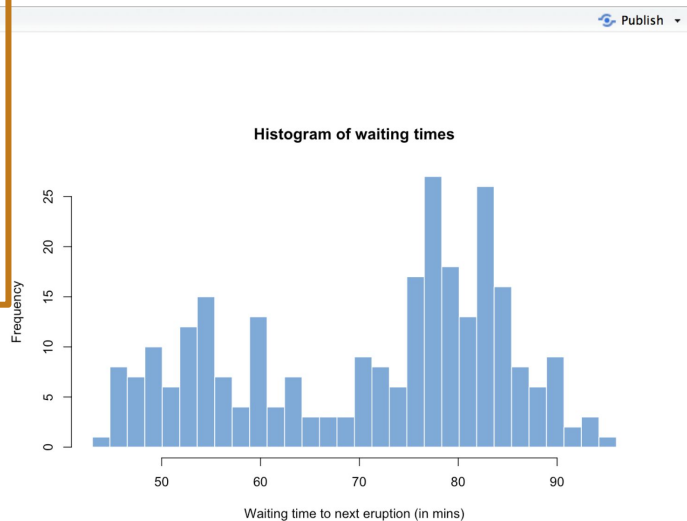
interactive web app

histogram which changes based on slider value
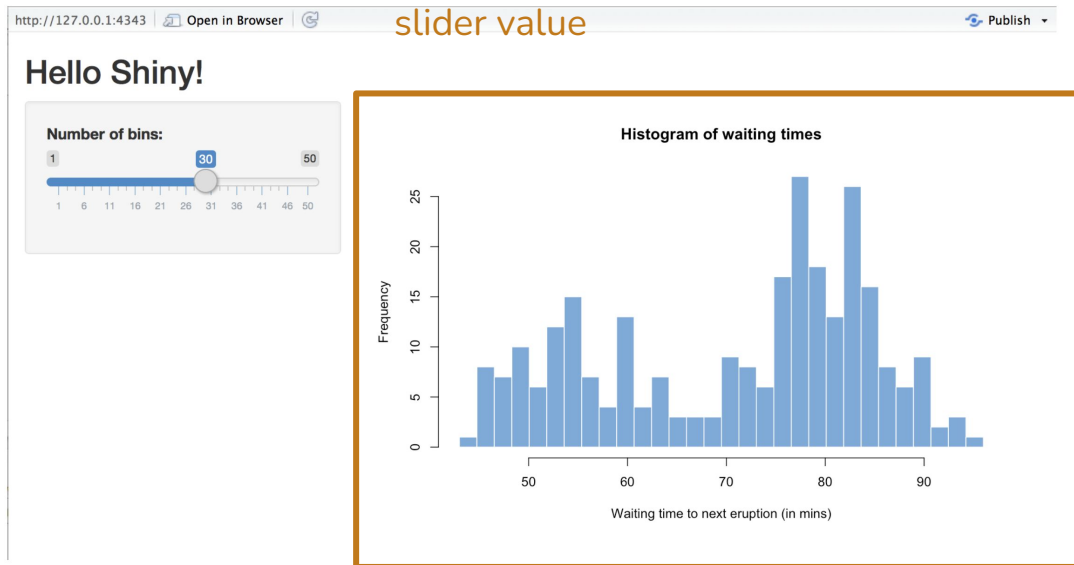
Before we get to Shiny dashboards, what is a Shiny app?

runs on a server

https://shiny.rstudio.com/

interactive web app

Shiny apps allow for interactivity and reactivity

- Interactivity: an element reacts to user actions
  - hover text, on-click actions, highlight on hover
  - {plotly}, {ggiraph}, {Shiny}

- Reactivity: data is updated from the server without refreshing the site
  - changing binwidth, filtering, changing parameters in models
  - {Shiny}, {shinydashboard}, {flexdashboard}

How would we make a Shiny app?

Shiny apps have three main components:

How would we make a Shiny app?

Shiny apps have three main components:

1. a user interface object

```r
library(shiny)

# Define UI for app that draws a histogram
ui <- fluidPage(
  # App title
  titlePanel("Hello Shiny!"),
  # Sidebar layout with input and output definitions
  sidebarLayout(
    # Sidebar panel for inputs
    sidebarPanel(
      # Input: Slider for the number of bins
      sliderInput(inputId = "bins",
                  label = "Number of bins:",
                  min = 1,
                  max = 50,
                  value = 30)

    ),
    # Main panel for displaying outputs
    mainPanel(
      # Output: Histogram
      plotOutput(outputId = "distPlot")

    )
  )
)
```

https://shiny.rstudio.com/articles/basics.html

How would we make a Shiny app?

Shiny apps have three main components:

1. a user interface object

```r
1  library(shiny)
2
3  # Define UI for app that draws a histogram
4  ui <- fluidPage(
5    # App title
6    titlePanel("Hello Shiny!"),
7    # Sidebar layout with input and output definitions
8    sidebarLayout(
9      # Sidebar panel for inputs
10     sidebarPanel(
11       # Input: Slider for the number of bins
12       sliderInput(inputId = "bins",
13                   label = "Number of bins:",
14                   min = 1,
15                   max = 50,
16                   value = 30)
17
18     ),
19     # Main panel for displaying outputs
20     mainPanel(
21       # Output: Histogram
22       plotOutput(outputId = "distPlot")
23
24     )
25   )
26 )
```

https://shiny.rstudio.com/articles/basics.html

How would we make a Shiny app?

Basic widgets

Buttons

Action

Submit

Single checkbox

☑ Choice A

Checkbox group

☑ Choice 1
☐ Choice 2
☐ Choice 3

Date input

2014-01-01

Date range

2017-06-21 to 2017-06-21

File input

Browse... No file selected

Help text

Note: help text isn't a true widget, but it provides an easy way to add text to accompany other widgets.

Numeric input

1

Radio buttons

◉ Choice 1
○ Choice 2
○ Choice 3

Select box

Choice 1 ▾

Sliders

0   50   100

0 10 20 30 40 50 60 70 80 90 100

0   25   75   100

0 10 20 30 40 50 60 70 80 90 100

Text input

Enter text...

How would we make a Shiny app?

Basic widgets

Buttons

Action

Submit

Single checkbox

☑ Choice A

Checkbox group

☑ Choice 1
☐ Choice 2
☐ Choice 3

Date input

2014-01-01

Date range

2017-06-21 to 2017-06-21

File input

Browse... No file selected

Help text

Note: help text isn't a true widget, but it provides an easy way to add text to accompany other widgets.

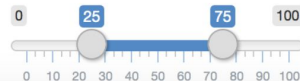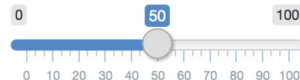### Numeric input

1

Radio buttons
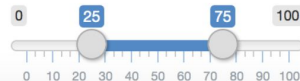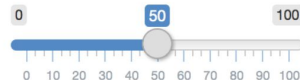
◉ Choice 1
◯ Choice 2
◯ Choice 3

Select box

Choice 1 ▼

Sliders

0   50   100

0 10 20 30 40 50 60 70 80 90 100

0   25   75   100

0 10 20 30 40 50 60 70 80 90 100

Text input

Enter text...

How would we make a Shiny app?

Shiny apps have three main components:

1. a user interface object
2. a server function

```
1   # Define server logic required to draw a histogram
2 ▾ server <- function(input, output) {
3
4     # Histogram of the Old Faithful Geyser Data
5     # with requested number of bins
6     # This expression that generates a histogram is wrapped in a call
7     # to renderPlot to indicate that:
8     #
9     # 1. It is "reactive" and therefore should be automatically
10    #    re-executed when inputs (input$bins) change
11    # 2. Its output type is a plot
12 ▾  output$distPlot <- renderPlot({
13
14      x    <- faithful$waiting
15      bins <- seq(min(x), max(x), length.out = input$bins + 1)
16
17      hist(x, breaks = bins, col = "#75AADB", border = "white",
18           xlab = "Waiting time to next eruption (in mins)",
19           main = "Histogram of waiting times")
20
21    })
22
23  }
```

https://shiny.rstudio.com/articles/basics.html

How would we make a Shiny app?

Shiny apps have three main components:

1. a user interface object
2. a server function

```
1   # Define server logic required to draw a histogram
2   server <- function(input, output) {
3
4      # Histogram of the Old Faithful Geyser Data
5      # with requested number of bins
6      # This expression that generates a histogram is wrapped in a call
7      # to renderPlot to indicate that:
8      #
9      # 1. It is "reactive" and therefore should be automatically
10     #    re-executed when inputs (input$bins) change
11     # 2. Its output type is a plot
12     output$distPlot <- renderPlot({
13
14        x    <- faithful$waiting
15        bins <- seq(min(x), max(x), length.out = input$bins + 1)
16
17        hist(x, breaks = bins, col = "#75AADB", border = "white",
18             xlab = "Waiting time to next eruption (in mins)",
19             main = "Histogram of waiting times")
20
21     })
22
23   }
```

https://shiny.rstudio.com/articles/basics.html

How would we make a Shiny app?

Shiny apps have three main components:

1. a user interface object
2. a server function

```
1   # Define server logic required to draw a histogram
2   server <- function(input, output) {
3
4     # Histogram of the Old Faithful Geyser Data
5     # with requested number of bins
6     # This expression that generates a histogram is wrapped in a call
7     # to renderPlot to indicate that:
8     #
9     # 1. It is "reactive" and therefore should be automatically
10    #    re-executed when inputs (input$bins) change
11    # 2. Its output type is a plot
12    output$distPlot <- renderPlot({
13
14      x    <- faithful$waiting
15      bins <- seq(min(x), max(x), length.out = input$bins + 1)
16
17      hist(x, breaks = bins, col = "#75AADB", border = "white",
18           xlab = "Waiting time to next eruption (in mins)",
19           main = "Histogram of waiting times")
20
21    })
22
23  }
```

https://shiny.rstudio.com/articles/basics.html

How would we make a Shiny app?

Shiny apps have three main components:

1. a user interface object
2. a server function
3. a call to `ShinyApp()`

```
1  ui <- {
2      ### UI
3      }
4
5  server <- {
6      ### Server
7      }
8
9  shinyApp(ui = ui, server = server)
10
11 runApp(shinyApp(ui = ui, server = server))
```

So what is a Shiny dashboard?

- Simply put, a collection of Shiny apps
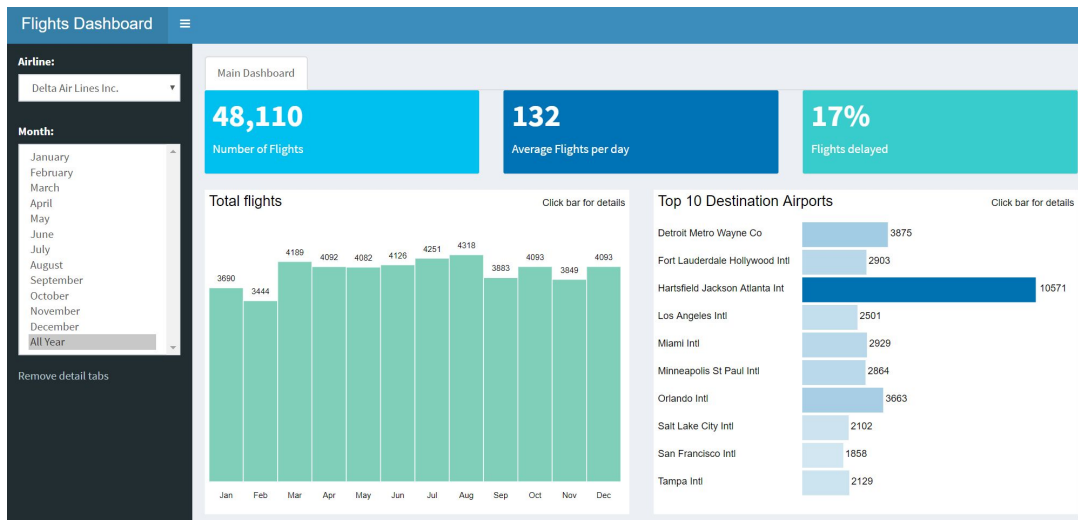- More generally, a graphical interface for users to quickly visualize important metrics

Image: https://db.rstudio.com/best-practices/dashboards/dashboard.png

Reasons to use Shiny dashboard

- Nice framework for making a professional looking product without wrangling with layouts

- You can do a lot without needing to know HTML and CSS

- If you already know flexdashboard, this may not provide a ton of advantages
  - shinydashboardPlus that provides even more features and is built off of shinydashboard

- Run analyses on the fly

Things to think about before we start

Before we start coding, I like to ask myself a few questions

      1. Who will my users be?

      2. What insights do we want our users to be able to gain?

      3. Is this the best way to present the data given #1 & #2?

and one more...

How will users access the dashboard?

R provides a couple of hosting options:

1. Deploy to the cloud using Shinyapps.io
    a. Different pricing levels offer different features, limits on active users, etc.
    b. Free tier does not include authentication
2. Host on Shiny server
    a. Open source
3. Deploy with RStudio Connect
    a. Commercial software
    b. Not free

* We needed authentication, so we set up the free Shiny server and app on an AWS server, then set up our website which had authentication to be a proxy for the server.

Ways to make your dashboard easier to deploy

- Have separate scripts for data preprocessing

- Use a utils script that installs all needed packages

- Only load libraries that you use a lot of functions from, e.g. shiny, dplyr, ggplot2
  Otherwise use package::function() notation

- Use Conventional Commits & tag repos

Now let's do some exercises!

https://github.com/anastasia-lucas/shinydash-rladies-dc