STAT 663 Project 1

Harinath Reddy

2022-10-26

```
library(readxl)
setwd("C:\\Users\\harin\\OneDrive\\Documents\\Statistical_graphs_and_exploration")

pfizer <- read_excel("C:\\Users\\harin\\OneDrive\\Documents\\Statistical_graphs_and_exploration\\pfizer

moderna <- read_excel("C:\\Users\\harin\\OneDrive\\Documents\\Statistical_graphs_and_exploration\\moder

janssen <- read_excel("C:\\Users\\harin\\OneDrive\\Documents\\Statistical_graphs_and_exploration\\janss
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.5
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(dplyr)
```

```
library(devtools)
```

```
## Loading required package: usethis
```

```
devtools::install_github('FIRST-Data-Lab/IDDA', force = TRUE)
```

```
## Downloading GitHub repo FIRST-Data-Lab/IDDA@HEAD
```

```
##          checking for file 'C:\Users\harin\AppData\Local\Temp\RtmpOk9TPk\remotes24f0c5ecca\FIRST-Data
##       -  preparing 'IDDA':
##    checking DESCRIPTION meta-information ...  v  checking DESCRIPTION meta-information
##       -  checking for LF line-endings in source and make files and shell scripts
##   -  checking for empty or unneeded directories
##      NB: this package now depends on R (>=        NB: this package now depends on R (>= 3.5.0)
##        WARNING: Added dependency on R >= 3.5.0 because serialized objects in
##      serialize/load version 3 cannot be read in older versions of R.
##      File(s) containing such objects:
##        'IDDA/data/CA.county.ts.rda' 'IDDA/data/D.county.rda'
##        'IDDA/data/D.state.rda' 'IDDA/data/I.county.rda'
##        'IDDA/data/I.state.rda' 'IDDA/data/PosTest.state.rda'
##        'IDDA/data/Test.state.rda' 'IDDA/data/counties1.rda'
##        'IDDA/data/counties2.rda' 'IDDA/data/county.top10.long.rda'
##        'IDDA/data/county.top10.rda' 'IDDA/data/features.county.rda'
```

```
##        'IDDA/data/features.state.rda' 'IDDA/data/fore.rda'
##        'IDDA/data/policy.county.rda' 'IDDA/data/pop.county.rda'
##        'IDDA/data/pop.state.rda' 'IDDA/data/state.long.rda'
##        'IDDA/data/state.ts.rda' 'IDDA/data/states1.rda'
##   -  building 'IDDA_1.0.0.tar.gz'
##
##
```

```
## Installing package into 'C:/Users/harin/AppData/Local/R/win-library/4.2'
## (as 'lib' is unspecified)
```

creating variables from Datasets

```
State <- pfizer$Jurisdiction
Date <- pfizer$Week.of.Allocations
Pfizer.1st.Dose.Allocations <- pfizer$X1st.Dose.Allocations
Pfizer.2nd.Dose.Allocations <- pfizer$X2nd.Dose.Allocations
Moderna.1st.Dose.Allocations <- moderna$X1st.Dose.Allocations
Moderna.2nd.Dose.Allocations <- moderna$X2nd.Dose.Allocations
Janssen.1st.Dose.Allocations <- janssen$X1st.Dose.Allocations
All.Dose.Allocations <- 0.5 *(Pfizer.2nd.Dose.Allocations+Pfizer.1st.Dose.Allocations+Moderna.1st.Dose.A
```

```
## Warning in Pfizer.2nd.Dose.Allocations + Pfizer.1st.Dose.Allocations +
## Moderna.1st.Dose.Allocations: longer object length is not a multiple of shorter
## object length
```

```
## Warning in Pfizer.2nd.Dose.Allocations + Pfizer.1st.Dose.Allocations +
## Moderna.1st.Dose.Allocations + : longer object length is not a multiple of
## shorter object length
```

```
## Warning in 0.5 * (Pfizer.2nd.Dose.Allocations + Pfizer.1st.Dose.Allocations + :
## longer object length is not a multiple of shorter object length
```

```
Cum.Allocation <-cumsum(All.Dose.Allocations)
```

#1.Data Exploration. #setting the variables to equal legth

```
length(State) <-length(pfizer$Jurisdiction)
length(Date) <- length(pfizer$Jurisdiction)
length(Pfizer.1st.Dose.Allocations) <- length(pfizer$Jurisdiction)
length(Pfizer.2nd.Dose.Allocations) <- length(pfizer$Jurisdiction)
length(Moderna.1st.Dose.Allocations) <- length(pfizer$Jurisdiction)
length(Moderna.2nd.Dose.Allocations) <- length(pfizer$Jurisdiction)
length(Janssen.1st.Dose.Allocations) <- length(pfizer$Jurisdiction)
length(All.Dose.Allocations) <- length(pfizer$Jurisdiction)
```

#creating a dummy dataframe

```
df <- data.frame(State, Date, Pfizer.1st.Dose.Allocations, Pfizer.2nd.Dose.Allocations, Moderna.1st.Dos
```

```
population_data <- IDDA::pop.state
```

```
df<- left_join(df,population_data, by="State")
```

#creating Doseperpop Variable using Mutate

```
library(dplyr)
df <- mutate(df, Doseperpop=Cum.Allocation/df$population)
```

#Making sure to drop all the NA values

```
week.allo.state <- df[!(df$State=="American Samoa" | df$State=="Palau" | df$State == "Guam" | df$State =
```

2.Visualization

```
library(plotly)
```

```
##
## Attaching package: 'plotly'
```

```
## The following object is masked from 'package:ggplot2':
##
##     last_plot
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following object is masked from 'package:graphics':
##
##     layout
```

```
plot1 <- plot_ly() %>%

# Add Cook County's time series using mode: lines+markers
  add_trace(data = week.allo.state %>%
              filter(State == 'Virginia'),x = ~Date , y = ~Janssen.1st.Dose.Allocations, type = 'scatte

   add_trace(data = week.allo.state %>%

               filter(State == 'Virginia'),x = ~Date , y = ~Pfizer.2nd.Dose.Allocations, type = 'scatte

  add_trace(data = week.allo.state %>%
               filter(State == 'Virginia'),x = ~Date , y = ~Pfizer.1st.Dose.Allocations, type = 'scatte

  add_trace(data = week.allo.state %>%
               filter(State == 'Virginia'),x = ~Date , y = ~Moderna.2nd.Dose.Allocations, type = 'scatt

  add_trace(data = week.allo.state %>%
               filter(State == 'Virginia'),x = ~Date , y = ~Moderna.1st.Dose.Allocations, type = 'scatt

library(htmlwidgets)
saveWidget(plot1, "Figure1.html", selfcontained = F)
```

Figure 1

(b)

```
plot2 <- plot_ly() %>%

add_trace(data = week.allo.state %>%
filter(State == "Virginia"),
x = ~Date, y = ~Doseperpop, type = 'scatter', mode = 'lines+markers',
showlegend = TRUE, name = 'doseperpopulation')
```

```
saveWidget(plot2, "Figure2.html", selfcontained = F)
```

Figure 2

#(c)

```
library(plotly)

plot3 <- plot_ly() %>%

add_trace(data = week.allo.state %>%
group_by(State)                          ,
x = ~Date, y = ~Doseperpop, type = 'scatter', mode = 'lines+markers',
showlegend = TRUE, name = 'doseperpopulation')

saveWidget(plot3, "Figure3.html", selfcontained = F)
```

Figure3

#map(d)

```
#install.packages("sp")
#install.packages("sf")
#install.packages("leaflet")
#install.packages("geojsonio")
#devtools::install_github("rstudio/leaflet")
```

1

```
library(geojsonio); library(leaflet); library(dplyr); library(sp); library(sf)
```

```
## Registered S3 method overwritten by 'geojsonsf':
##   method        from
##   print.geojson geojson
```

```
##
## Attaching package: 'geojsonio'
```

```
## The following object is masked from 'package:devtools':
##
##     lint
```

```
## The following object is masked from 'package:base':
##
##     pretty
```

```
## Linking to GEOS 3.9.1, GDAL 3.4.3, PROJ 7.2.1; sf_use_s2() is TRUE
```

```
library(sp)
library(sf)
library(rgeos)
```

```
## rgeos version: 0.5-9, (SVN revision 684)
##  GEOS runtime version: 3.9.1-CAPI-1.14.2
##  Please note that rgeos will be retired by the end of 2023,
## plan transition to sf functions using GEOS at your earliest convenience.
##  GEOS using OverlayNG
##  Linking to sp version: 1.5-0
##  Polygon checking: TRUE
```

```
library(geojsonio); library(leaflet); library(dplyr)
```

```
df2<- data.frame(week.allo.state)
df_date <- df2[757:1070,]
```

```
pal.state.factor <- colorFactor(palette = "YlOrRd", domain = All.Dose.Allocations)
```

```
library(geojsonio)
states0 <- geojson_read(
x = "https://raw.githubusercontent.com/PublicaMundi/MappingAPI/master/data/geojson/us-states.json"
, what = "sp"
)
class(states0)
```

```
## [1] "SpatialPolygonsDataFrame"
## attr(,"package")
## [1] "sp"
```

```
head(states0@data)
```

```
##   id       name density
## 1 01    Alabama  94.650
## 2 02     Alaska   1.264
## 3 04    Arizona  57.050
## 4 05   Arkansas  56.430
## 5 06 California 241.700
## 6 08   Colorado  49.330
```

```
states1 <- states0
states1@data <- states0@data %>%
  mutate(name_ns = sapply(name, gsub, pattern = " ",
replacement = ""))

states1@data <- left_join(states1@data, week.allo.state%>%


                          filter(Date == as.Date('2021-03-01')),
by = c('name_ns' = 'State'))

states1@data <- states1@data%>% mutate(new_bin_dosepop = cut(Doseperpop, breaks = 6))
```

```
labels_cases <- sprintf("<strong>%s</strong><br/>Population: %g M<br>
  Dosage per population: %g<br>
  All dose allocation: %g",
  states1$name_ns, round(states1$population / (1e6), 2),
  states1$Doseperpop, states1$All.Dose.Allocations) %>%
 lapply(htmltools::HTML)
labels_cases[[1]]
```

```
pal.state.factor <- colorFactor(
  palette = "YlOrRd",
  domain = states1@data$Doseperpop
)
dmap <- leaflet(states1) %>%
  setView(-96, 37.8, zoom = 4) %>%
  addPolygons(fillColor = ~pal.state.factor(Doseperpop),
```

```
    weight = 1, opacity = 0.5,
    color = "white", dashArray = "3",
    fillOpacity = 0.9, layerId = ~name_ns,
    highlight = highlightOptions(
    weight = 5, color = "#666",
    dashArray = NULL, fillOpacity = 0.9,
    bringToFront = TRUE),
    label = labels_cases,
    labelOptions = labelOptions(
    style = list("font-weight" = "normal", padding = "3px 8px"),
    textsize = "15px", direction = "auto"))
```

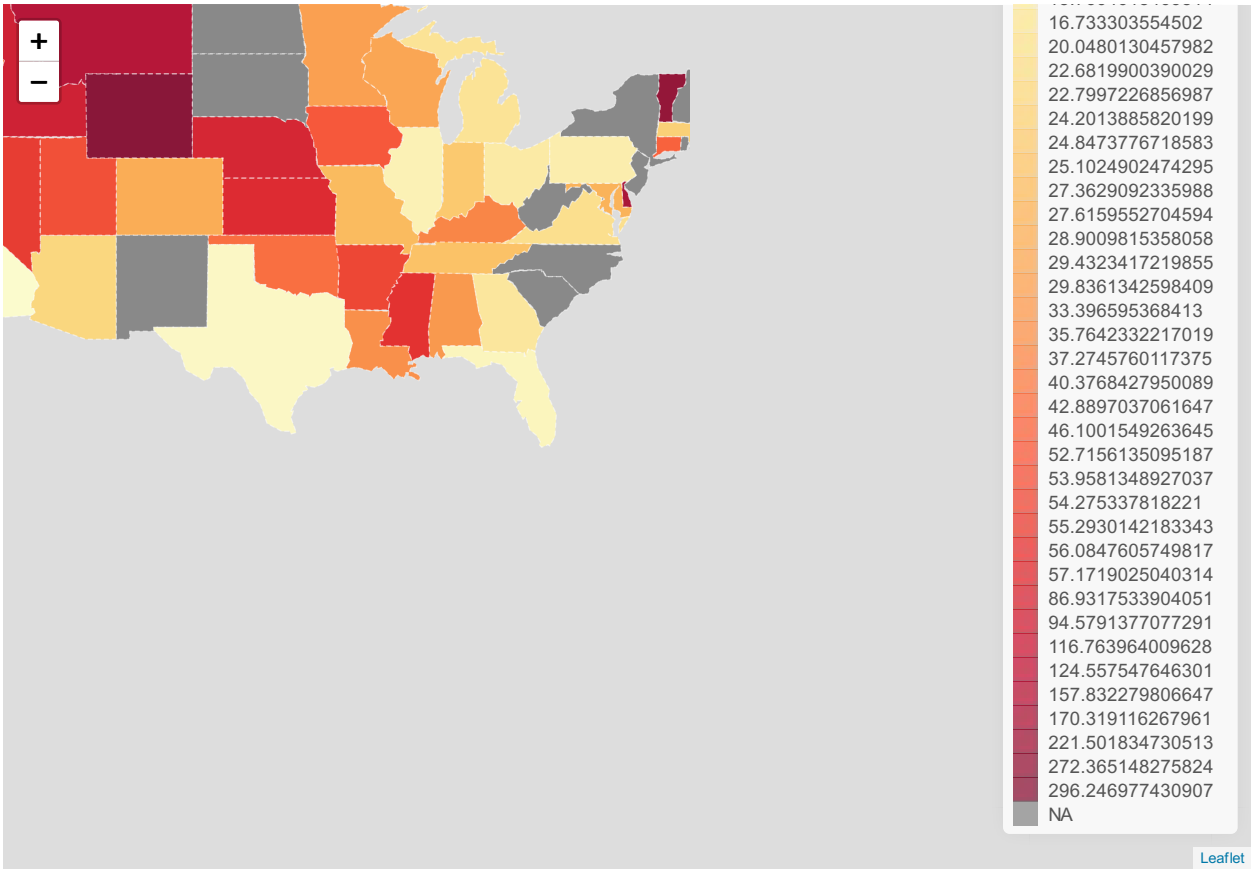
```
## Warning in RColorBrewer::brewer.pal(max(3, n), palette): n too large, allowed maximum for palette Yl
## Returning the palette you asked for with that many colors
```

```
dmap <- dmap %>% addLegend(pal = pal.state.factor, values = ~Doseperpop,
            opacity = 0.7, title = "Dosage per population",
            position = "bottomright")
```

```
## Warning in RColorBrewer::brewer.pal(max(3, n), palette): n too large, allowed maximum for palette Yl
## Returning the palette you asked for with that many colors
```

```
dmap
```

```
## Input to asJSON(keep_vec_names=TRUE) is a named vector. In a future version of jsonlite, this option
```




```
saveWidget(dmap, "Figure4.html", selfcontained = F)
```

```
## Input to asJSON(keep_vec_names=TRUE) is a named vector. In a future version of jsonlite, this option
```

Figure 4