STA141B Final Assignment

Sitong Qian

12/13/2020

For this report,my job is to create a interactive time series plot for covid-19 cases in California county. The data used is extracted from new york time, for the first step, I used professor's code directly to extract data.

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

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

## The following object is masked from 'package:httr':
##
##     config

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

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

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select

## The following object is masked from 'package:plotly':
##
##     select

## -- Attaching packages --------------------------------------------------------------------- tidyvers

## v tibble  3.0.4     v purrr   0.3.3
## v tidyr   1.1.2     v forcats 0.5.0
## v readr   1.3.1
## -- Conflicts ------------------------------------------------------------------------ tidyverse_confl
## x tidyr::complete() masks RCurl::complete()
## x plotly::config()  masks httr::config()
## x dplyr::filter()   masks plotly::filter(), stats::filter()
## x dplyr::lag()      masks stats::lag()
## x MASS::select()    masks dplyr::select(), plotly::select()
```

```
## [1] 3337
```

Now, I have the covid-19 dataset, my next step is to investigate the information presented in the dataset. I found out the information has a total of 3337 columns since I am aware that the data here is far more than the number of states. I used the sapply function to call out the region type and found out it contains, country-level,state-level, county-level, and territory. Then, I was wondering how to find out the county related to California states, being said, I first google it online, and found out geoid is the unique key here, and for the county in California, all of them start with 06 followed by 3 digits. Thus, I extracted all entries with 06 by a regular expression and I stored the index value as CA_county_index.

Then, with the CA county index, I now convert and processed the information I needed in the final plot. First, I fixed the date, since the data is stored by date, I convert the range to date value. Then, I extracted county name and case and death For pro-rated cases and deaths, I divided them by population. Also, I processed the accumulated cases and deaths to present more meaningful information.

```r
## date
date=seq(as.Date("2020-03-01"), as.Date("2020-12-13"), by="days")

## county_name

county_nm = sapply(1:length(us$data),function(i){
  county_name = us$data[[i]][["display_name"]]
  return(county_name)
  }
)

CA_cty = county_nm[CA_county_index]


## case and death
case = sapply(1:length(us$data),function(i){
  case = us$data[[i]][["cases"]]
  return(case)
  }
)
death = sapply(1:length(us$data),function(i){
  death = us$data[[i]][["deaths"]]
  return(death)
  }
)

CA_cases = case[CA_county_index]
CA_deathes = death[CA_county_index]

## Pro-rated rate for cases and deathes
population = sapply(1:length(us$data),function(i){
  population = us$data[[i]][["population"]]
  return(population)
  }
)
CA_population = population[CA_county_index]
Pro_rated_cases = mapply(FUN = `/`, CA_cases, CA_population, SIMPLIFY = FALSE)
Pro_rated_deathes = mapply(FUN = `/`, CA_deathes, CA_population, SIMPLIFY = FALSE)

## Accumulate cases for each date
acc_case = sapply(1:length(CA_cases),function(i){
  case_daily = CA_cases[[i]]
  acc_case <- case_daily %>% accumulate(`+`)
  return(acc_case)
  }
)



## Accumulate deathes for each date
acc_death = sapply(1:length(CA_deathes),function(i){
  death_daily = CA_deathes[[i]]
  acc_death <- death_daily %>% accumulate(`+`)
  return(acc_death)
  }
)
```

Since the information I got is a list of list, which can't be used for graphing, I now convert them to list.

```r
case_final= sapply(1:length(CA_cases),function(i){
  case_fl = CA_cases[[i]]
  return(case_fl)
  }
)

death_final=sapply(1:length(CA_deathes),function(i){
  death_fl = CA_deathes[[i]]
  return(death_fl)
  }
)

pro_case_final=sapply(1:length(Pro_rated_cases),function(i){
  pro_case_fl = Pro_rated_cases[[i]]
  return(pro_case_fl)
  }
)

pro_death_final=sapply(1:length(Pro_rated_deathes),function(i){
  pro_death_fl = Pro_rated_deathes[[i]]
  return(pro_death_fl)
  }
)
```

Here, I convert everything into a dataframe for interactive plot for later used.

```r
### convert everything into a dataframe for interactive plot
dt_case=as.vector(case_final)
dt_death=as.vector(death_final)
dt_p_case=as.vector(pro_case_final)
```

```r
dt_p_death=as.vector(pro_death_final)
dt_acc_case=as.vector(acc_case)
dt_acc_death=as.vector(acc_death)
dt_date=rep(date, 58)
dt_county=rep(CA_cty, each = 288)
```

```r
final_covid_data=as.data.frame(as.vector(as.character(dt_date)))
final_covid_data$case <- dt_case
final_covid_data$death <- dt_death
final_covid_data$pro_case <- dt_p_case
final_covid_data$pro_death <- dt_p_death
final_covid_data$acc_case <- dt_acc_case
final_covid_data$acc_death <- dt_acc_death
final_covid_data$county <- dt_county
colnames(final_covid_data)[1] <- "date"
```

Here, I first graphed it in ggplot with everything listed above and group them by county. Then, I introduce plotly for creating interactive plot here, with highlight feature and sizingpolicy to make sure that the text I added later will not be covered by the plot.

Next, I added the button by using professor's code, while fixed the function with HideLines.js and hideLines() to make the function works.

```r
p1 <- final_covid_data %>%
  highlight_key(~county) %>%
  ggplot(aes(x = date, y = case, color = county, group = county,
             text = paste( "death:", death,
                           "\n case:",case,
                           "\n pro-rated case:",pro_case,
                           "\n pro-rated death:",pro_death,
                           "\n cumulative case:",acc_case,
                           "\n cumulative death:",acc_death))) +
    geom_line() +
    geom_point()


ply1 <- ggplotly(p1, tooltip = c("x", "group", "y", "text"),width = 1000,height = 700) %>%
  highlight(on = "plotly_hover", off = "plotly_doubleclick")






library(htmltools)

h = HTML("<h2>California County Cases</h2><p>
This plot shows the COVID cases for each county in California by date.
<button onclick=\"hideLines()\">Click me</button>")
h2 = HTML("<ol>
<li>Click on the button to hide all of the lines</li>
<li>Hover/mouse over a point to see the tooltip and more details about that day's data for that county<
<li>Click on a time series to highlight it and grey out the others</li>
<li>Click on a county name in the legend to toggle the corresponding time series's visibility on
plot</li>
</ol>")
```

```r
sc = tags$script(paste(readLines("HideLines.js"), collapse = "\n"))
sc = tags$script(src = "HideLines.js")
ply2 = prependContent(ply1, sc, h, h2)
saveWidget(ply2, "combined.html", selfcontained = FALSE)
```

Then, I planed to work on the bonus part, which meant by creating a interactive plot for CA map, to do this, I planned to first search on if R has packages included county-level california map, and then combined such graph with the total's data accumulate cases. But because of time, I have tried numerous way of combining the data into the map, but I failed, I first try add the county in the dataframe, to meet up the requirements for built in map functions. But I falied, next I tried join two dataframe with latitude, but I failed again.

```r
library(usmap)
```

```
## Warning: package 'usmap' was built under R version 3.6.2
```

```r
library(ggmap)
```

```
## Google's Terms of Service: https://cloud.google.com/maps-platform/terms/.

## Please cite ggmap if you use it! See citation("ggmap") for details.

##
## Attaching package: 'ggmap'

## The following object is masked from 'package:plotly':
##
##     wind
```

```r
library(rgdal)
```

```
## Warning: package 'rgdal' was built under R version 3.6.2

## Loading required package: sp

## rgdal: version: 1.5-18, (SVN revision 1082)
## Geospatial Data Abstraction Library extensions to R successfully loaded
## Loaded GDAL runtime: GDAL 2.4.2, released 2019/06/28
## Path to GDAL shared files: /Library/Frameworks/R.framework/Versions/3.6/Resources/library/rgdal/gdal
## GDAL binary built with GEOS: FALSE
## Loaded PROJ runtime: Rel. 5.2.0, September 15th, 2018, [PJ_VERSION: 520]
## Path to PROJ shared files: /Library/Frameworks/R.framework/Versions/3.6/Resources/library/rgdal/proj
## Linking to sp version:1.4-4
## Overwritten PROJ_LIB was /Library/Frameworks/R.framework/Versions/3.6/Resources/library/rgdal/proj
```

```r
acc_case_total <- acc_case[290,]
acc_death_total <- acc_death[290,]
dt_acc_case_total <- as.vector(acc_case_total)
dt_acc_death_total <- as.vector(acc_death_total)
dt_county_bonus <- (CA_cty)
bonus_covid_data=as.data.frame(as.vector(acc_case_total))
bonus_covid_data$case_total <- acc_case_total
bonus_covid_data$death_total <- acc_death_total
bonus_covid_data$fips <- dt_county_bonus
ca_base <- plot_usmap('counties',include = c('CA'))
ca_base
```