

**Gildart Haase School of Computer Sciences and Engineering**

**Metropolitan Campus**

**CSCI 6885.81 Big Data Analytics with Hadoop and R**

**Dr. Wenyun Dai**

**Coronavirus Data analysis**

Shunqi Zheng

Junior

2020/5/10

**Table of Contents**

[Configuration 3](#_Toc40024363)

[Using the tool and problem met 4](#_Toc40024364)

[An so amazing idea when I Googling something(highlighted) 4](#_Toc40024365)

[Preprocessing and analyzing the data: 5](#_Toc40024366)

[Apply function 7](#_Toc40024367)

[# plot the data 7](#_Toc40024368)

[# Build a linear regression 12](#_Toc40024369)

[# # explore an idea 14](#_Toc40024370)

# Configuration

knitr::opts\_chunk$set(error = TRUE)

the packages I installed:

library(openxlsx)

## Warning: package 'openxlsx' was built under R version 3.6.3

library(xlsx)

##   
## Attaching package: 'xlsx'

## The following objects are masked from 'package:openxlsx':  
##   
## createWorkbook, loadWorkbook, read.xlsx, saveWorkbook, write.xlsx

library(tidyverse)

## Warning: package 'tidyverse' was built under R version 3.6.3

## -- Attaching packages ------------------------------------- tidyverse 1.3.0 --

## √ ggplot2 3.3.0 √ purrr 0.3.3  
## √ tibble 2.1.3 √ dplyr 0.8.4  
## √ tidyr 1.0.2 √ stringr 1.4.0  
## √ readr 1.3.1 √ forcats 0.4.0

## Warning: package 'ggplot2' was built under R version 3.6.3

## -- Conflicts ---------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(ggplot2)  
library(dplyr)  
library(factoextra)

## Warning: package 'factoextra' was built under R version 3.6.3

## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa

library(cluster)

All the data are sorted by date and grouped by location (country) build a linear model

# Using the tool and problem met

I recognized that using a normal way to finish a final project is a waste of time, for example, implement the code and make a report with text and images by copying and pasting.

So, I tried to google how to simplify this process. And I found there is a tool that can achieve my thought: *Jupyter*. I put tons of time into configuring its installation, especially the IDE, because every time I try to run the code in Python or R, they didn't work since the External reasons. So, I gave up this tool.

So, I decided to find the alternative: *Rmarkdown*. Then I can do images, words, and code at the same workplace. It was so easy that I never thought I could simply work in Rstudio as usual with some additional notation to add the text and images, just like Word document.

# An so amazing idea when I Googling something(highlighted)

I was stuck on what functions I can apply for this coronavirus data, then I googled it, I found many functions that they almost all have the pipe operators, "%>%, then I wonder why people always like using it in their codes.

Here is what an amazing thought I came across:

This is how my codes look like before in R: plot(diff(log(sample(rnorm(10000, mean = 10, sd = 1), size = 100, replace = FALSE))), col = “red”, type = “l”)

I think this was the right case that I wrote the code, until the pipe operators, %>%, someone using it.

let data = [|1..100|] |> Array.filter (fun i -> i*i <= 50) |> Array.map (fun i -> i+i*i) |> Array.sum

Yes, it is much clearer than the last same codes as they said.

When I tried to compare why my code is worse than this new one, I suddenly recalled that I learned *Lambda calculus reduction* in math class. That was so painful that I looking for which functions corresponding to which parameters. I was like why I have to learn how the old codes look like in modern era.

Now until I learned how %>% parameter works, I realized that the problem has already solved by %>%. This is so cool operator simplifies the code, which makes the reader read someone else's codes.

A great idea to solve parameters pass problem.

This is really what I found when doing this project. I must note it down since it is an so amazing moment!

# Preprocessing and analyzing the data:

First, let's read the data:

country\_data <-read.csv("country.csv")  
world\_data <- read.xlsx("world.xlsx", sheetIndex = 1)

Then let’s see the summary of the dataset:

summary(country\_data)

## date location new\_cases new\_deaths   
## 4/6/2020: 203 Australia : 100 Min. : -9.0 Min. : 0.000   
## 4/7/2020: 203 Austria : 100 1st Qu.: 0.0 1st Qu.: 0.000   
## 4/8/2020: 203 Belgium : 100 Median : 0.0 Median : 0.000   
## 4/4/2020: 202 Canada : 100 Mean : 146.3 Mean : 8.565   
## 4/3/2020: 201 China : 100 3rd Qu.: 13.0 3rd Qu.: 0.000   
## 4/5/2020: 201 Czech Republic: 100 Max. :34272.0 Max. :2004.000   
## (Other) :8300 (Other) :8913   
## total\_cases total\_deaths   
## Min. : 0 Min. : 0.00   
## 1st Qu.: 0 1st Qu.: 0.00   
## Median : 8 Median : 0.00   
## Mean : 1971 Mean : 92.38   
## 3rd Qu.: 121 3rd Qu.: 1.00   
## Max. :398809 Max. :17129.00   
##

summary(world\_data)

## date location new\_cases new\_deaths   
## Min. :2019-12-31 World:100 Min. : 0.0 Min. : 0.0   
## 1st Qu.:2020-01-24 1st Qu.: 406.2 1st Qu.: 12.5   
## Median :2020-02-18 Median : 2141.0 Median : 85.0   
## Mean :2020-02-18 Mean :13918.9 Mean : 814.8   
## 3rd Qu.:2020-03-14 3rd Qu.:11821.2 3rd Qu.: 375.0   
## Max. :2020-04-08 Max. :86713.0 Max. :7412.0   
## total\_cases total\_deaths   
## Min. : 27 Min. : 0.00   
## 1st Qu.: 1237 1st Qu.: 37.25   
## Median : 74259 Median : 1942.50   
## Mean : 187773 Mean : 8790.75   
## 3rd Qu.: 146114 3rd Qu.: 5495.50   
## Max. :1391890 Max. :81478.00

Although we know the date is from 2019-12-31 to 2020-04-08, however, As I found that the date format is not in the same format

So, I tried str() function

str(country\_data)

## 'data.frame': 9513 obs. of 6 variables:  
## $ date : Factor w/ 100 levels "1/1/2020","1/10/2020",..: 32 1 12 23 26 27 28 29 30 31 ...  
## $ location : Factor w/ 204 levels "Afghanistan",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ new\_cases : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ new\_deaths : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ total\_cases : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ total\_deaths: int 0 0 0 0 0 0 0 0 0 0 ...

str(world\_data)

## 'data.frame': 100 obs. of 6 variables:  
## $ date : Date, format: "2019-12-31" "2020-01-01" ...  
## $ location : Factor w/ 1 level "World": 1 1 1 1 1 1 1 1 1 1 ...  
## $ new\_cases : num 27 0 0 17 0 15 0 0 0 0 ...  
## $ new\_deaths : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ total\_cases : num 27 27 27 44 44 59 59 59 59 59 ...  
## $ total\_deaths: num 0 0 0 0 0 0 0 0 0 0 ...

I found that the date format in country\_data is "Factor," whereas in world\_data is "Date."

And it is clearer when I use class() function to see the format of dates in two different dataset.

class(country\_data$date)

## [1] "factor"

class(world\_data$date)

## [1] "Date"

Then I used below function:

country\_data$date<- as.Date(country\_data$date,format = "%m/%d/%Y")

Now, check the current format of it:

class(country\_data$date)

## [1] "Date"

Finally, combine these two datasets into a whole one.

virusData <- rbind(country\_data,world\_data)

Check the head and tail Data:

head(virusData)

## date location new\_cases new\_deaths total\_cases total\_deaths  
## 1 2019-12-31 Afghanistan 0 0 0 0  
## 2 2020-01-01 Afghanistan 0 0 0 0  
## 3 2020-01-02 Afghanistan 0 0 0 0  
## 4 2020-01-03 Afghanistan 0 0 0 0  
## 5 2020-01-04 Afghanistan 0 0 0 0  
## 6 2020-01-05 Afghanistan 0 0 0 0

tail(virusData)

## date location new\_cases new\_deaths total\_cases total\_deaths  
## 9608 2020-04-03 World 77447 4689 1005937 51573  
## 9609 2020-04-04 World 81714 6586 1087651 58159  
## 9610 2020-04-05 World 86713 6115 1174364 64274  
## 9611 2020-04-06 World 71232 4655 1245596 68929  
## 9612 2020-04-07 World 71392 5137 1316988 74066  
## 9613 2020-04-08 World 74902 7412 1391890 81478

Then, let’s check if the data has missing value:

any(is.na(virusData))

## [1] FALSE

Luckily, there is no missing value within it, so I can start applying the function now.

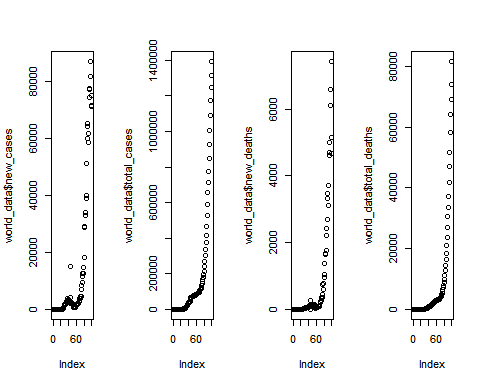
# Apply function

# # plot the data

I use a tidyverse package, which includes many packages used for plotting the data.

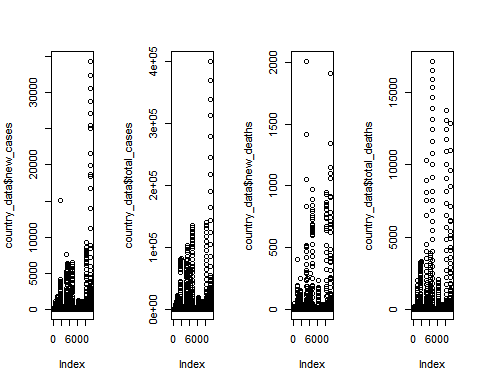
# install.packages("tidyverse")  
# library(tidyverse)

par(mfrow=c(1,4))  
plot(world\_data$new\_cases)  
plot(world\_data$total\_cases)  
plot(world\_data$new\_deaths)  
plot(world\_data$total\_deaths)

 We found that the world data points shown in different variables are all exponentially grows.

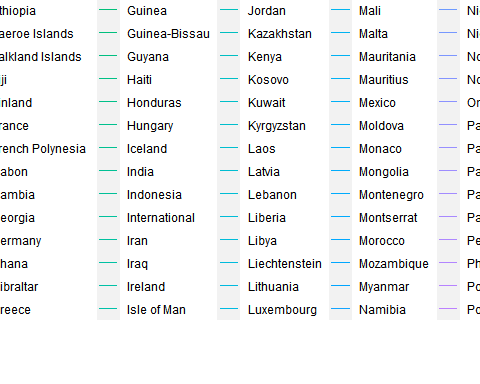
Let’see the country data:

par(mfrow=c(1,4))  
plot(country\_data$new\_cases)  
plot(country\_data$total\_cases)  
plot(country\_data$new\_deaths)  
plot(country\_data$total\_deaths)

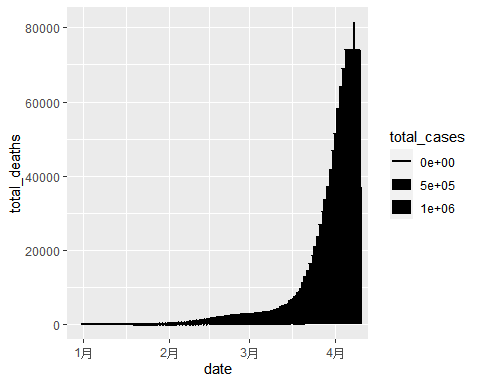
 As we can see that these points are So messy, nothing special.

Now, let's try to scatter point.

# library(ggplot2)  
ggplot(virusData, aes(x=date, y=total\_cases, size=total\_deaths,color=location))+geom\_line()

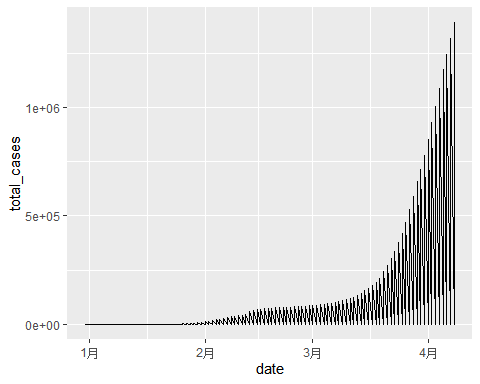
 it’s too much to get the whole figure since the number of locations are too many.

ggplot(virusData, aes(x=date, y=total\_deaths, size=total\_cases))+geom\_line()

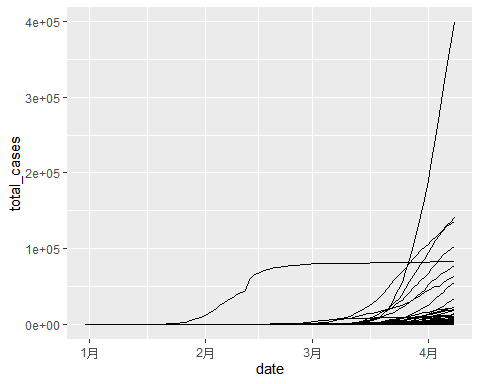


And it’s much clear when I remove the function size() and color().

library(dplyr)  
virusData %>% ggplot(aes(date, total\_cases))+geom\_line()



country\_data %>%  
 ggplot(aes(date, total\_cases, group = location))+geom\_line()

 we can find a special curve that differs the others, but still not idea about which country it represents.

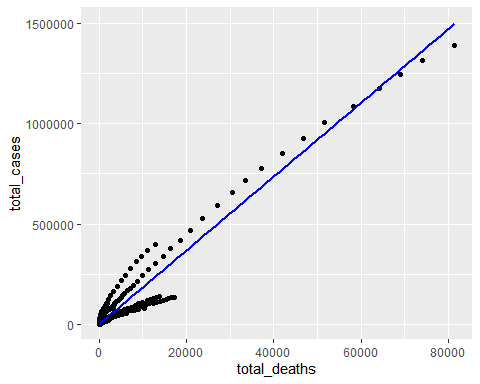
# # Build a linear regression

linearmodel <- lm(total\_cases ~total\_deaths, data=virusData)  
summary(linearmodel)

##   
## Call:  
## lm(formula = total\_cases ~ total\_deaths, data = virusData)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -179678 -544 -541 -474 165746   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 543.80131 95.79378 5.677 1.41e-08 \*\*\*  
## total\_deaths 18.37352 0.04497 408.579 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 9358 on 9611 degrees of freedom  
## Multiple R-squared: 0.9456, Adjusted R-squared: 0.9456   
## F-statistic: 1.669e+05 on 1 and 9611 DF, p-value: < 2.2e-16

ggplot(virusData,aes(x = total\_deaths,y= total\_cases))+geom\_point()+stat\_smooth(method = "lm", col="blue")

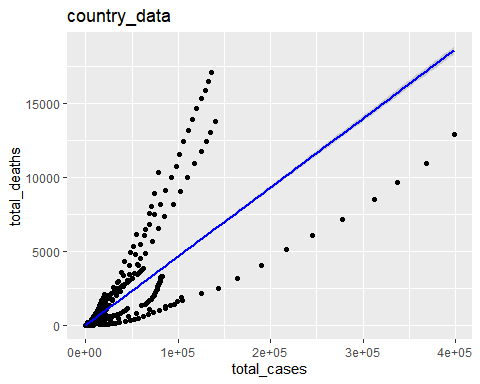
## `geom\_smooth()` using formula 'y ~ x'

 Here, we found that the line almost fit the data going further, but still, many other points are not.

let’s just find out the feature of country and world data:

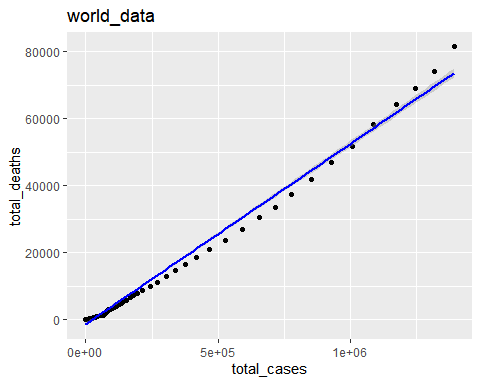
par(mfrow=c(1,2))  
# still make a linear model  
linearmodel1 <- lm(total\_cases ~total\_deaths, data=country\_data)  
# plot the model  
ggplot(country\_data,aes(x = total\_cases,y=total\_deaths))+ggtitle("country\_data") +geom\_point()+stat\_smooth(method = "lm", col="blue")

## `geom\_smooth()` using formula 'y ~ x'



# for world data doing the same steps  
linearmodel2 <- lm(total\_cases ~total\_deaths, data=world\_data)  
ggplot(world\_data,aes(x = total\_cases,y=total\_deaths)) +geom\_point()+stat\_smooth(method = "lm", col="blue")+ggtitle("world\_data")

## `geom\_smooth()` using formula 'y ~ x'

 So we found that the line is actually fit the world\_data, not for the country data, I thought that the reason is different country should have their own linear model featursasdade.

# # # explore an idea

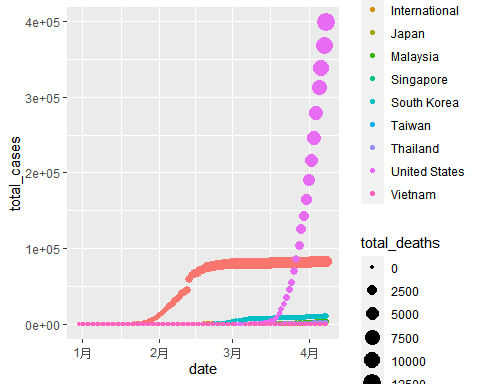
let’s find out the top 10 countries with most cases within these months. I was trying to select the top 10 countries in one specific date, like on 2020-2-15.

subcountry\_data <- filter(country\_data,date == "2020-02-15")%>%  
 arrange(.,total\_cases)%>% tail(.,10)  
subcountry\_data

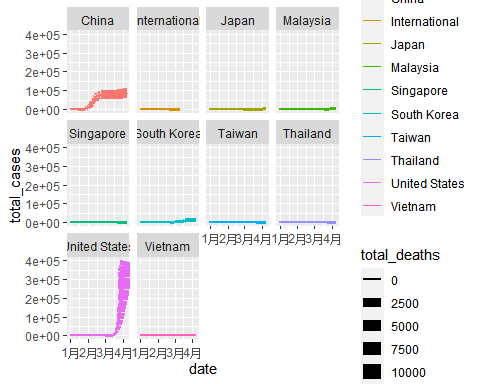
## date location new\_cases new\_deaths total\_cases total\_deaths  
## 58 2020-02-15 United States 0 0 15 0  
## 59 2020-02-15 Vietnam 0 0 16 0  
## 60 2020-02-15 Taiwan 0 0 18 0  
## 61 2020-02-15 Malaysia 2 0 21 0  
## 62 2020-02-15 South Korea 0 0 28 0  
## 63 2020-02-15 Thailand 1 0 34 0  
## 64 2020-02-15 Japan 8 0 38 1  
## 65 2020-02-15 Singapore 9 0 67 0  
## 66 2020-02-15 International 0 0 221 0  
## 67 2020-02-15 China 2538 143 66559 1524

Ok, so let's do some investigation about these ten countries.

# Making a subset of selected top 10 countries:  
top10countries <- country\_data[ country\_data$location %in% c("United States","Vietnam","Taiwan","Malaysia","South Korea", "Thailand","Japan","Singapore","International","China"),]  
  
# ggplot the subset with labels of locations  
ggplot(top10countries,aes(x=date, y= total\_cases, color = location,size=total\_deaths))+geom\_point()

 We found that China and United States are the two countries that have more cases than the others. And China is the one with this different line we saw before. the number of cases in United states explosively raised in the middle of March, whereas in China, it is tending to be flattening after the middle of March. They both have most cases during this period by the way.

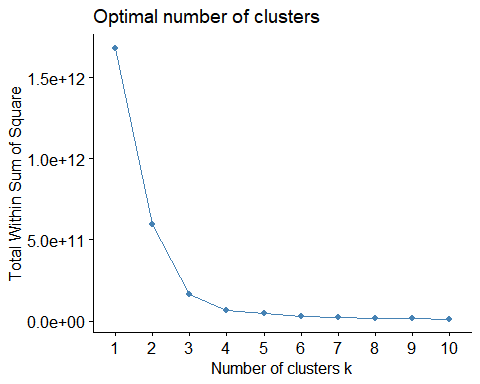
ggplot(top10countries,aes(x=date,y=total\_cases,group = location,color = location, size = total\_deaths))+geom\_line()+facet\_wrap(~location)

 This is more clear.

library(factoextra)  
library(cluster)  
  
par(mfrow=c(1,2))  
fviz\_nbclust(top10countries,FUN=hcut, method = "wss")

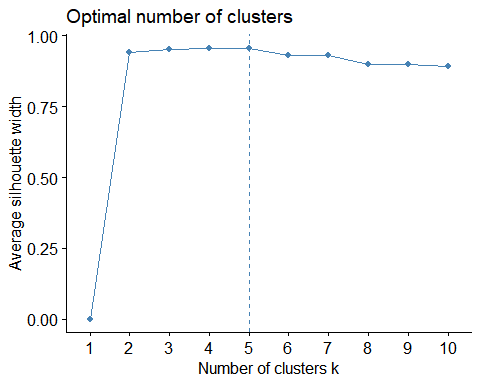
## Warning in stats::dist(x): 强制改变过程中产生了NA

## Warning in stats::dist(x, method = method, ...): 强制改变过程中产生了NA  
  
## Warning in stats::dist(x, method = method, ...): 强制改变过程中产生了NA  
  
## Warning in stats::dist(x, method = method, ...): 强制改变过程中产生了NA  
  
## Warning in stats::dist(x, method = method, ...): 强制改变过程中产生了NA  
  
## Warning in stats::dist(x, method = method, ...): 强制改变过程中产生了NA  
  
## Warning in stats::dist(x, method = method, ...): 强制改变过程中产生了NA  
  
## Warning in stats::dist(x, method = method, ...): 强制改变过程中产生了NA  
  
## Warning in stats::dist(x, method = method, ...): 强制改变过程中产生了NA  
  
## Warning in stats::dist(x, method = method, ...): 强制改变过程中产生了NA  
  
## Warning in stats::dist(x, method = method, ...): 强制改变过程中产生了NA



fviz\_nbclust(top10countries,FUN=hcut, method = "silhouette")

## Warning in stats::dist(x): 强制改变过程中产生了NA  
  
## Warning in stats::dist(x): 强制改变过程中产生了NA  
  
## Warning in stats::dist(x): 强制改变过程中产生了NA  
  
## Warning in stats::dist(x): 强制改变过程中产生了NA  
  
## Warning in stats::dist(x): 强制改变过程中产生了NA  
  
## Warning in stats::dist(x): 强制改变过程中产生了NA  
  
## Warning in stats::dist(x): 强制改变过程中产生了NA  
  
## Warning in stats::dist(x): 强制改变过程中产生了NA  
  
## Warning in stats::dist(x): 强制改变过程中产生了NA  
  
## Warning in stats::dist(x): 强制改变过程中产生了NA



gap\_stat<- clusGap(top10countries$total\_cases, FUN=hcut,nstart = 25, K.max = 4, B=50)

## Error in clusGap(top10countries$total\_cases, FUN = hcut, nstart = 25, : length(dim(x)) == 2 is not TRUE

top10countries$total\_cases %>% na.omit() %>% scale()

k1<- kmeans(top10countries$total\_cases, centers = 2, nstart = 25)  
fviz\_cluster(k1, data=top10countries)

## Error in colMeans(x, na.rm = TRUE): 'x'必需为数值

I failed in this step. And I still don't know why a data frame with many vectors can be put in this cluster function, how the cluster function to plot the clusters to segregate data based on too many vectors in a data frame.

# # Conclusion

I learned a lot from this project; to be honest, I learned how to set up and configure the environment of Rstudio, Rmarkdown, and Jupyter. Although one of them did not work when I did my project. But I still solved issues and fixed them by googling and self-teaching.

This dataset is similar to the one we did in homework, but we must look up the format of different vectors by transforming the data type. This transformation is one of the processes of data analysis. Also, by doing this project, I learned that why preprocessing the data is so essential and how painful it is. It spent a lot of time until I can get the figure by applying some functions along with it. Finally, some plot functions may not work because of the model of data, or the type of dataset has not matched that method. So we have to try many functions several times.

Finally, with the output of running these bunch of codes, we can get the information from them, and that's how data analysts usually do in these years.