FIT 1043 ASSIGNMENT 3 REPORT

Name: Chew Yang Xuan

Student ID: 33520496

Introduction

This report mainly explains the use the BASH Shell and the R programming language to work on a larger dataset provided. The dataset provided is a pre-processed data that already filtered by keywords related to COVID-19. The dataset contains a variety of information including user id, location, followers and other related information. In this report, the relevant codes and outputs will be shown, followed by the answer to each question and explanation to the code provided. The report outline are as follow:

1. A1: Inspecting the Data
2. A2: Investigating the information from Data
3. A3: Data aggregation
4. A4: Small Challenge
5. Conclusion
6. References

A1: Inspecting the Data

Question 1

Shell Command:

```
$ ls -lh corona_tweets.csv.gz
```

Output:

```
-rw-r--r-- 1 user 197121 118M May 15 16:28 corona_tweets.csv.gz
```

Answer:

Based on the output above, the file size is 118 Megabytes.

Question 2

Shell Command:

```
$ cat corona_tweets.csv.gz | gunzip | head -1 | tr "\t" "\n"
```

Output:

```
Created
Tweet_ID
Text
User_ID
User
User_Location
Followers_Count
Friends_Count
Geo
Place_Type
Place_Name
Place_Country
Language
```

Answer:

Every column header is separated in each line to provide a better visualisation to the existing headers. There are 13 headers in total which are: Created, Tweet_ID, Text, User_ID, User, User_Location, Followers_Count, Friends_Count, Geo, Place_Type, Place_Name, Place_Country and Language.

Question 3

Shell Command:

```
$ cat corona_tweets.csv.gz | gunzip | wc -l
```

Output:

```
1143559
```

Answer:

There are 1143559 lines in the dataset provided.

A2: Investigating the information from Data

Question 1

As the question referring to the number of unique twitter users in the dataset, we should be referring to the User_ID column.

Shell Command:

```
$ cat corona_tweets.csv.gz | gunzip | awk -F '\t' '{print $4}' | grep -v
'User_ID' | sort | uniq | wc -l
```

Output:

```
641975
```

Answer:

- The ‘awk’process the content by extracting the User_ID column.
- ‘grep’ is used to exclude the column header which is the User_ID or else it will include the header as one of the unique ID resulting in wrong answer.
- ‘sort’ is used to sort the User_ID accordingly. It is important because ‘uniq’ only removes adjacent duplicate lines.
- ‘uniq’ filters adjacent matching lines, which the data filtered right now only consists of all the unique user IDs.
- ‘wc -l’ will count the number of the lines in the output from the ‘uniq’.
- There is a total of 641975 number of the unique twitter users.

Question 2

a) Shell Command:

```
$ cat corona_tweets.csv.gz | gunzip | cut -f 3 | grep -i "vaccine" | wc -l
```

Output:

```
19403
```

Answer:

- The ‘cut’ is used to extract the third column of the dataset, which is Text.
- The ‘grep’ is used to search ‘vaccine’ that word in the particular column.
- ‘-i’ flag is used to ignore the case distinction in the ‘vaccine’ key word.
- ‘wc -l’ is used to show how many lines there are after the data is being filtered.
- The output shows there are 19403 tweets mentioned the word ‘vaccine’.

b) Shell Command:

```
$ cat corona_tweets.csv.gz | gunzip | cut -f 3 | grep -i "vaccine" | grep -v -E
'vaccine|Vaccine' | wc -l
```

Output:

```
370
```

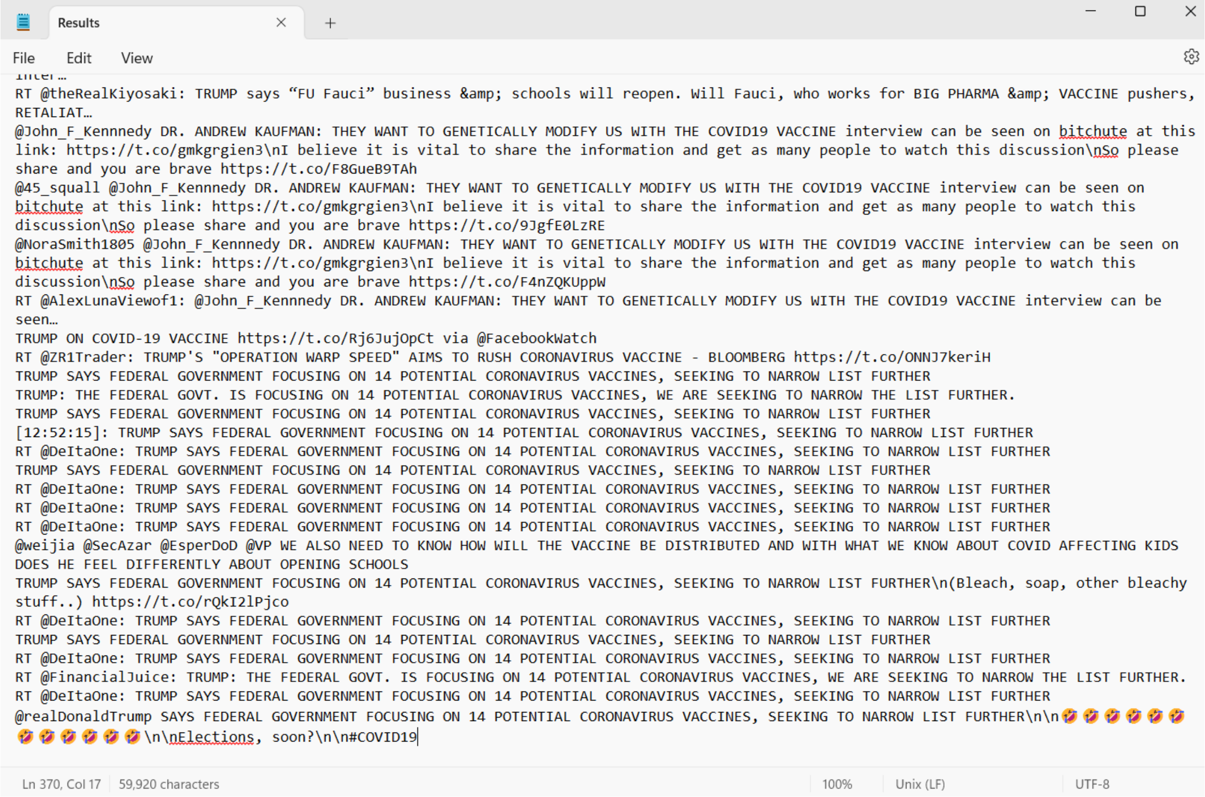
Answer:

- Based on the command in 2a, we add in an additional grep command.
- '-v' flag is used to filter unwanted words from any combination of capital and small letter of the word 'vaccine'.
- '-E' flag is used to make sure the program filtered out and exclude the two different words ("vaccine" and "Vaccine" are excluded in)
- The output shows there are 370 tweets mentioning the word 'vaccine' in other combination of uppercase and lowercase but not spelt exactly "vaccine" or "Vaccine".

c) Shell Command:

```
$ cat corona_tweets.csv.gz | gunzip | cut -f 3 | grep -i "vaccine" | grep -v -E
'vaccine|Vaccine' > Results.txt
```

The overview of Results.txt file



```
inter…
RT @theRealKiyosaki: TRUMP says "FU Fauci" business &amp; schools will reopen. Will Fauci, who works for BIG PHARMA &amp; VACCINE pushers,
RETALIAT…
@John_F_Kennnedy DR. ANDREW KAUFMAN: THEY WANT TO GENETICALLY MODIFY US WITH THE COVID19 VACCINE interview can be seen on bitchute at this
link: https://t.co/gmkgrgien3\nI believe it is vital to share the information and get as many people to watch this discussion\nSo please
share and you are brave https://t.co/F8GueB9TAh
@45_squall @John_F_Kennnedy DR. ANDREW KAUFMAN: THEY WANT TO GENETICALLY MODIFY US WITH THE COVID19 VACCINE interview can be seen on
bitchute at this link: https://t.co/gmkgrgien3\nI believe it is vital to share the information and get as many people to watch this
discussion\nSo please share and you are brave https://t.co/9JgfE0LzRE
@NoraSmith1805 @John_F_Kennnedy DR. ANDREW KAUFMAN: THEY WANT TO GENETICALLY MODIFY US WITH THE COVID19 VACCINE interview can be seen on
bitchute at this link: https://t.co/gmkgrgien3\nI believe it is vital to share the information and get as many people to watch this
discussion\nSo please share and you are brave https://t.co/F4nZQKUppW
RT @AlexLunaViewof1: @John_F_Kennnedy DR. ANDREW KAUFMAN: THEY WANT TO GENETICALLY MODIFY US WITH THE COVID19 VACCINE interview can be
seen…
TRUMP ON COVID-19 VACCINE https://t.co/Rj6JujOpCt via @FacebookWatch
RT @ZR1Trader: TRUMP'S "OPERATION WARP SPEED" AIMS TO RUSH CORONAVIRUS VACCINE - BLOOMBERG https://t.co/ONNJ7keriH
TRUMP SAYS FEDERAL GOVERNMENT FOCUSING ON 14 POTENTIAL CORONAVIRUS VACCINES, SEEKING TO NARROW LIST FURTHER
TRUMP: THE FEDERAL GOVT. IS FOCUSING ON 14 POTENTIAL CORONAVIRUS VACCINES, WE ARE SEEKING TO NARROW THE LIST FURTHER.
TRUMP SAYS FEDERAL GOVERNMENT FOCUSING ON 14 POTENTIAL CORONAVIRUS VACCINES, SEEKING TO NARROW LIST FURTHER
[12:52:15]: TRUMP SAYS FEDERAL GOVERNMENT FOCUSING ON 14 POTENTIAL CORONAVIRUS VACCINES, SEEKING TO NARROW LIST FURTHER
RT @DeItaOne: TRUMP SAYS FEDERAL GOVERNMENT FOCUSING ON 14 POTENTIAL CORONAVIRUS VACCINES, SEEKING TO NARROW LIST FURTHER
TRUMP SAYS FEDERAL GOVERNMENT FOCUSING ON 14 POTENTIAL CORONAVIRUS VACCINES, SEEKING TO NARROW LIST FURTHER
RT @DeItaOne: TRUMP SAYS FEDERAL GOVERNMENT FOCUSING ON 14 POTENTIAL CORONAVIRUS VACCINES, SEEKING TO NARROW LIST FURTHER
RT @DeItaOne: TRUMP SAYS FEDERAL GOVERNMENT FOCUSING ON 14 POTENTIAL CORONAVIRUS VACCINES, SEEKING TO NARROW LIST FURTHER
RT @DeItaOne: TRUMP SAYS FEDERAL GOVERNMENT FOCUSING ON 14 POTENTIAL CORONAVIRUS VACCINES, SEEKING TO NARROW LIST FURTHER
@weijia @SecAzar @EsperDoD @VP WE ALSO NEED TO KNOW HOW WILL THE VACCINE BE DISTRIBUTED AND WITH WHAT WE KNOW ABOUT COVID AFFECTING KIDS
DOES HE FEEL DIFFERENTLY ABOUT OPENING SCHOOLS
TRUMP SAYS FEDERAL GOVERNMENT FOCUSING ON 14 POTENTIAL CORONAVIRUS VACCINES, SEEKING TO NARROW LIST FURTHER\n(Bleach, soap, other bleachy
stuff..) https://t.co/rQkI2lPjco
RT @DeItaOne: TRUMP SAYS FEDERAL GOVERNMENT FOCUSING ON 14 POTENTIAL CORONAVIRUS VACCINES, SEEKING TO NARROW LIST FURTHER
TRUMP SAYS FEDERAL GOVERNMENT FOCUSING ON 14 POTENTIAL CORONAVIRUS VACCINES, SEEKING TO NARROW LIST FURTHER
RT @DeItaOne: TRUMP SAYS FEDERAL GOVERNMENT FOCUSING ON 14 POTENTIAL CORONAVIRUS VACCINES, SEEKING TO NARROW LIST FURTHER
RT @FinancialJuice: TRUMP: THE FEDERAL GOVT. IS FOCUSING ON 14 POTENTIAL CORONAVIRUS VACCINES, WE ARE SEEKING TO NARROW THE LIST FURTHER.
RT @DeItaOne: TRUMP SAYS FEDERAL GOVERNMENT FOCUSING ON 14 POTENTIAL CORONAVIRUS VACCINES, SEEKING TO NARROW LIST FURTHER
@realDonaldTrump SAYS FEDERAL GOVERNMENT FOCUSING ON 14 POTENTIAL CORONAVIRUS VACCINES, SEEKING TO NARROW LIST FURTHER\n\n🤣🤣🤣🤣🤣🤣
🤣🤣🤣🤣🤣🤣\n\nElections, soon?\n\n#COVID19
```

Ln 370, Col 17 | 59,920 characters | 100% | Unix (LF) | UTF-8

A3: Data aggregation

Question 1

As we are referring to the twitter users and the number of followers each user has. We should inspect the data between both columns ‘User_ID’ (Column 4) and ‘Followers_Count’ (Column 7).

Shell Command:

```
$ cat corona_tweets.csv.gz | gunzip | awk -F'\t' '{print $4,$7}' | sort -u | less
```

- The ‘`awk`’process the content by extracting the User_ID column.
- The ‘`-f`’ flag used to tell awk that the columns are separated by tab.
- The instruction ‘`{print $4,$7}`’ tells it to print the value in columns 4 and 7 for each line of input it sees.
- ‘`sort`’ is used to sort the ‘User_ID’ accordingly.
- The ‘`-u`’ flag is used to output only the first of the equal run (unique lines).
- The ‘`less`’ command is used to visualise the dataset.

By visualising the content of the output, we can see that there are some repeating users exist in between the sorted data. Example is provided as follows:

```
100032683 2768

100032683 2769
```

This happens might be due to the user has tweeted a few times a day, resulting in the multiple records. During the time they posted the tweets, they might have gained or lost followers, causing the follower count to be different. Therefore, we must choose the latest data with the latest followers count and remove the repeating data. By taking the user id above as an example, we can reach this by using the following code:

Shell command:

```
$ cat corona_tweets.csv.gz | gunzip | awk -F'\t' '{print $4,$7}' | sort -r -u |
awk '!seen[$1]++' | grep '100032683'
```

- Based on the previous command, I added in a few more commands to remove the duplicates.
- ‘`-r`’ flag is used to sort the number of followers reversely when the user id is the same. This allows the latest record will exist at the top.
- ‘`awk ‘!seen[$1]++`’ is used to only take the first or topmost data based on the user id that are repeated.

Output:

```
100032683 2769
```

We can see that the particular user id only left with the data with the latest follower’s count. Hence, we can use this shell command as a base to group them accordingly.

a) Shell command:

```
$ cat corona_tweets.csv.gz | gunzip | awk -F'\t' '{print $4,$7}' | sort -r -u |
awk '!seen[$1]++' | awk -F' ' '$2<=1500 {print $1,$2}' | wc -l
```

Output:

```
498446
```

b) Shell command:

```
$ cat corona_tweets.csv.gz | gunzip | awk -F'\t' '{print $4,$7}' | sort -r -u |
awk '!seen[$1]++' | awk -F' ' '$2>=1501 && $2<=2500 {print $1,$2}' | wc -l
```

Output:

```
43883
```

c) Shell command:

```
$ cat corona_tweets.csv.gz | gunzip | awk -F'\t' '{print $4,$7}' | sort -r -u |
awk '!seen[$1]++' | awk -F' ' '$2>=2501 && $2<=3500 {print $1,$2}' | wc -l
```

Output:

```
23607
```

d) Shell command:

```
$ cat corona_tweets.csv.gz | gunzip | awk -F'\t' '{print $4,$7}' | sort -r -u |
awk '!seen[$1]++' | awk -F' ' '$2>=3501 && $2<=4500 {print $1,$2}' | wc -l
```

Output:

```
15152
```

e) Shell command:

```
$ cat corona_tweets.csv.gz | gunzip | awk -F'\t' '{print $4,$7}' | sort -r -u |
awk '!seen[$1]++' | awk -F' ' '$2>=4501 && $2<=5500 {print $1,$2}' | wc -l
```

Output:

```
9286
```

f) Shell command:

```
$ cat corona_tweets.csv.gz | gunzip | awk -F'\t' '{print $4,$7}' | sort -r -u |
awk '!seen[$1]++' | awk -F' ' '$2>=5501 && $2<=6500 {print $1,$2}' | wc -l
```

Output:

```
6838
```

g) Shell command:

```
$ cat corona_tweets.csv.gz | gunzip | awk -F'\t' '{print $4,$7}' | sort -r -u |
awk '!seen[$1]++' | awk -F' ' '$2>=6501 && $2<=7500 {print $1,$2}' | wc -l
```

Output:

```
5067
```

h) Shell command:

```
$ cat corona_tweets.csv.gz | gunzip | awk -F'\t' '{print $4,$7}' | sort -r -u |
awk '!seen[$1]++' | awk -F' ' '$2>=7501 && $2<=8500 {print $1,$2}' | wc -
```

Output:

```
3854
```

i) Shell command:

```
$ cat corona_tweets.csv.gz | gunzip | awk -F'\t' '{print $4,$7}' | sort -r -u |
awk '!seen[$1]++' | awk -F' ' '$2>=8501 && $2<=9500 {print $1,$2}' | wc -l
```

Output:

```
3071
```

j) Shell command:

```
$ cat corona_tweets.csv.gz | gunzip | awk -F'\t' '{print $4,$7}' | sort -r -u |
awk '!seen[$1]++' | awk -F' ' '$2>9500 {print $1,$2}' | grep -v 'User_ID' | wc -l
```

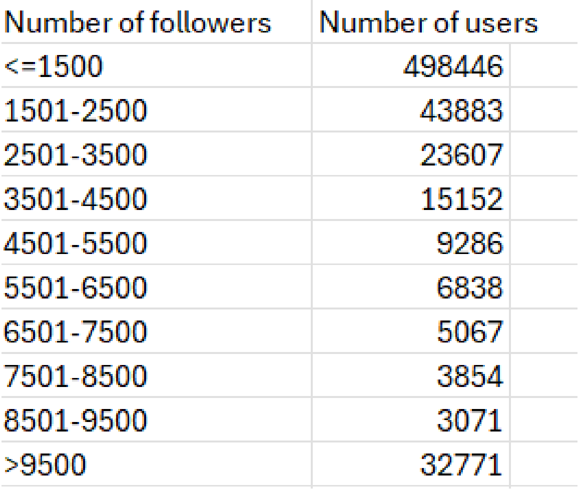
Output:

```
32771
```

Explanation:

- In all the code, there is an extra ‘awk’ command which is used to filter the number of followers accordingly.
- In the last category, which the number of followers falls more than 9500 included the headings as one of the data.
- ‘`grep -v`’ function is used to invert the match, whereby it doesn’t include the headers when calculating the number of data.

2) A CSV file to tabulate the data obtained.



Number of followers	Number of users
<=1500	498446
1501-2500	43883
2501-3500	23607
3501-4500	15152
4501-5500	9286
5501-6500	6838
6501-7500	5067
7501-8500	3854
8501-9500	3071
>9500	32771

3) Upload the csv file created to the R studio.

R Code:

```
>  data <- read.table(("A3_Q2_Output.csv"),header=TRUE,sep=",")
>   data
```

Output:

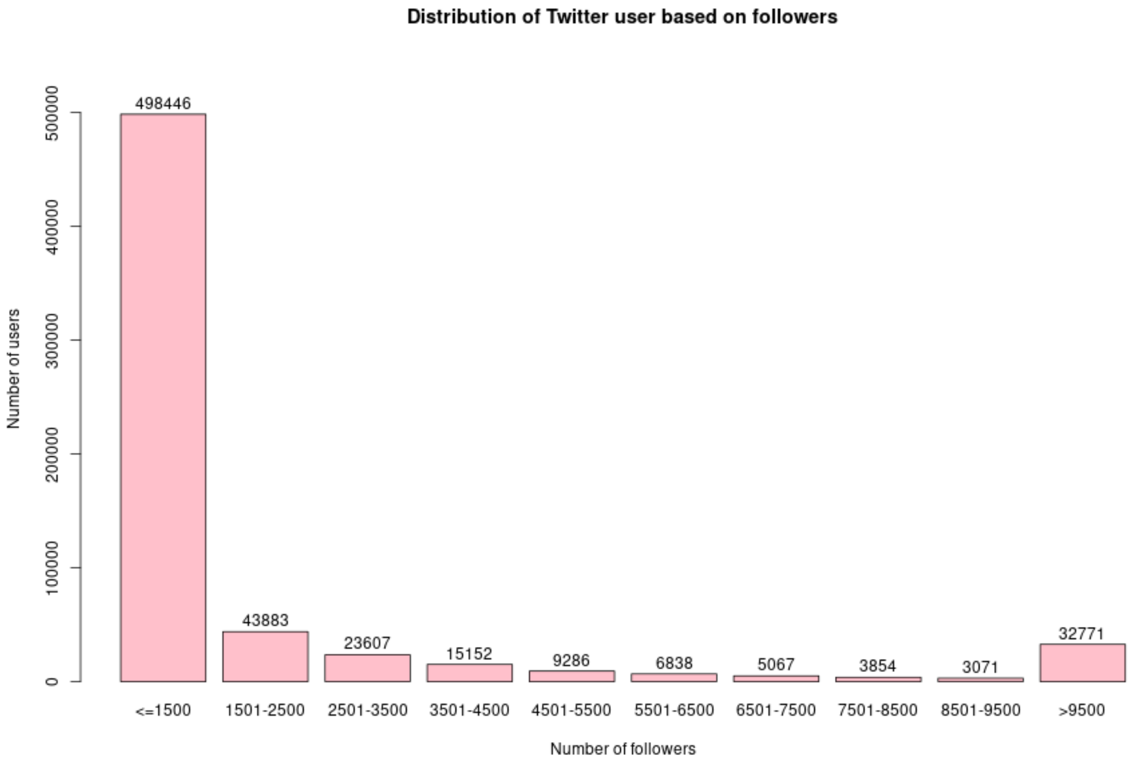
```
   Number.of.followers         Number.of.users
1                 <=1500                498446
2                 1501-2500             43883
3                 2501-3500             23607
4                 3501-4500             15152
5                 4501-5500             9286
6                 5501-6500             6838
7                 6501-7500             5067
8                 7501-8500             3854
9                 8501-9500             3071
10                 >9500                32771
```

4) Bar chart is plotted to visualise the data.

R Code:

```
>  png('bargraph.png',width= 900,600)
>  graph = barplot(data$Number.of.users, ylim = c(0,550000),
          main = "Distribution of Twitter user based on followers",
          names.arg = data$Number.of.followers,
          xlab = "Number of followers", ylab = "Number of users",
          col = "pink")
>  text(x = graph,
   y = data$Number.of.users + 10000,
   label=data$Number.of.users)
dev.off()
```

Output:



A4: Small challenge

As for the A3, we are removing the duplicating data. To ensure the consistency in making comparison between retweet data and non-retweet data between all the unique users, we need to remove the duplicating data in this case also.

1) Shell command:

```
$ cat corona_tweets.csv.gz | gunzip | cut -f 3,4,7 | grep -v 'RT @' | awk -F'\t'
'{print $2,$3}' | sort -r -u | awk '!seen[$1]++' | gzip > non_retweet.csv.gz
```

- `‘cut -f’` extracts the third, fourth and seventh column from the filtered data.
- `‘grep -v’` function is used to invert the match, whereby it doesn’t include the `‘RT @’` which will be found in the text column.
- `‘sort -r -u’` is used to sort the data reversely based on the number of follower’s count. The topmost data will show the latest record.
- `‘awk ‘!seen[$1]++’’` is used to only take the first or topmost data based on the user id that are repeated.
- `‘gzip’` function is used to compress a file. In this case, the output of the results is compress into a gz file called non_retweet.csv.

2)

a) Shell command:

```
$ cat non_retweet.csv.gz | gunzip | awk -F' ' '$2<=1500 {print $1,$2}' | wc -l
```

Output:

```
156968
```

b) Shell command:

```
$ cat non_retweet.csv.gz | gunzip | awk -F' ' '$2>=1501 && $2<=2500 {print
$1,$2}' | wc -l
```

Output:

```
16058
```

c) Shell command:

```
$ cat non_retweet.csv.gz | gunzip | awk -F' ' '$2>=2501 && $2<=3500 {print
$1,$2}' | wc -l
```

Output:

```
9011
```

d) Shell command:

```
$ cat non_retweet.csv.gz | gunzip | awk -F' ' '$2>=3501 && $2<=4500 {print
$1,$2}' | wc -l
```

Output:

```
6068
```

e) Shell command:

```
$ cat non_retweet.csv.gz | gunzip | awk -F' ' '$2>=4501 && $2<=5500 {print
$1,$2}' | wc -l
```

Output:

```
3867
```

f) Shell command:

```
$ cat non_retweet.csv.gz | gunzip | awk -F' ' '$2>=5501 && $2<=6500 {print
$1,$2}' | wc -l
```

Output:

```
2962
```

g) Shell command:

```
$ cat non_retweet.csv.gz | gunzip | awk -F' ' '$2>=6501 && $2<=7500 {print
$1,$2}' | wc -l
```

Output:

```
2183
```

h) Shell command:

```
$ cat non_retweet.csv.gz | gunzip | awk -F' ' '$2>=7501 && $2<=8500 {print
$1,$2}' | wc -l
```

Output:

```
1726
```

i) Shell command:

```
$ cat non_retweet.csv.gz | gunzip | awk -F' ' '$2>=8501 && $2<=9500 {print
$1,$2}' | wc -l
```

Output:

```
1424
```

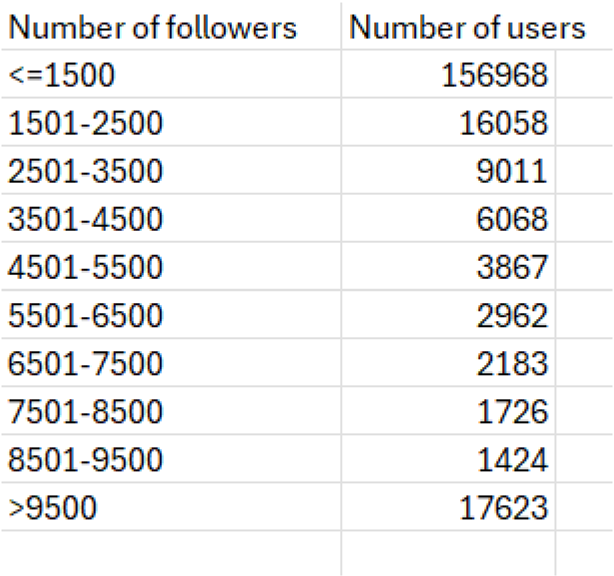
j) Shell command:

```
$ cat non_retweet.csv.gz | gunzip | awk -F' ' '$2>9500 {print $1,$2}' | grep -v
'User_ID' | wc -l
```

Output:

```
17623
```

Create a CSV file to tabulate the data obtained.



Number of followers	Number of users
<=1500	156968
1501-2500	16058
2501-3500	9011
3501-4500	6068
4501-5500	3867
5501-6500	2962
6501-7500	2183
7501-8500	1726
8501-9500	1424
>9500	17623

3) R Code:

```
> non_rt_data <- read.table(("A4_Q2.csv"),header=TRUE,sep=",")
> non_rt_data
```

Output:

```
  Number.of.followers  Number.of.users
1         <=1500           156968
2       1501-2500          16058
3       2501-3500          9011
4       3501-4500          6068
5       4501-5500          3867
6       5501-6500          2962
7       6501-7500          2183
8       7501-8500          1726
9       8501-9500          1424
10        >9500            17623
```

4) R Code:

Combine both csv file together as a data frame and add in a new column to represent retweet and non-retweet data.

```
> df1 = data.frame(data)
> df2 = data.frame(non_rt_data)
> final_df = rbind(df1, df2)
> final_df$Types_of_Tweets = c(rep("Include RT", 10), rep("Exclude RT", 10))
> final_df
```

Output:

```
  Number.of.followers   Number.of.users  Types_of_Tweets
1          <=1500            498446            Include RT
2        1501-2500           43883             Include RT
3        2501-3500           23607             Include RT
4        3501-4500           15152             Include RT
5        4501-5500           9286              Include RT
6        5501-6500           6838              Include RT
7        6501-7500           5067              Include RT
8        7501-8500           3854              Include RT
9        8501-9500           3071              Include RT
10            >9500          32771             Include RT
11          <=1500           156968            Exclude RT
12       1501-2500           16058             Exclude RT
13       2501-3500            9011             Exclude RT
14       3501-4500            6068             Exclude RT
15       4501-5500           3867              Exclude RT
16       5501-6500            2962             Exclude RT
17       6501-7500           2183              Exclude RT
18       7501-8500           1726              Exclude RT
19       8501-9500           1424              Exclude RT
20            >9500          17623             Exclude RT
```

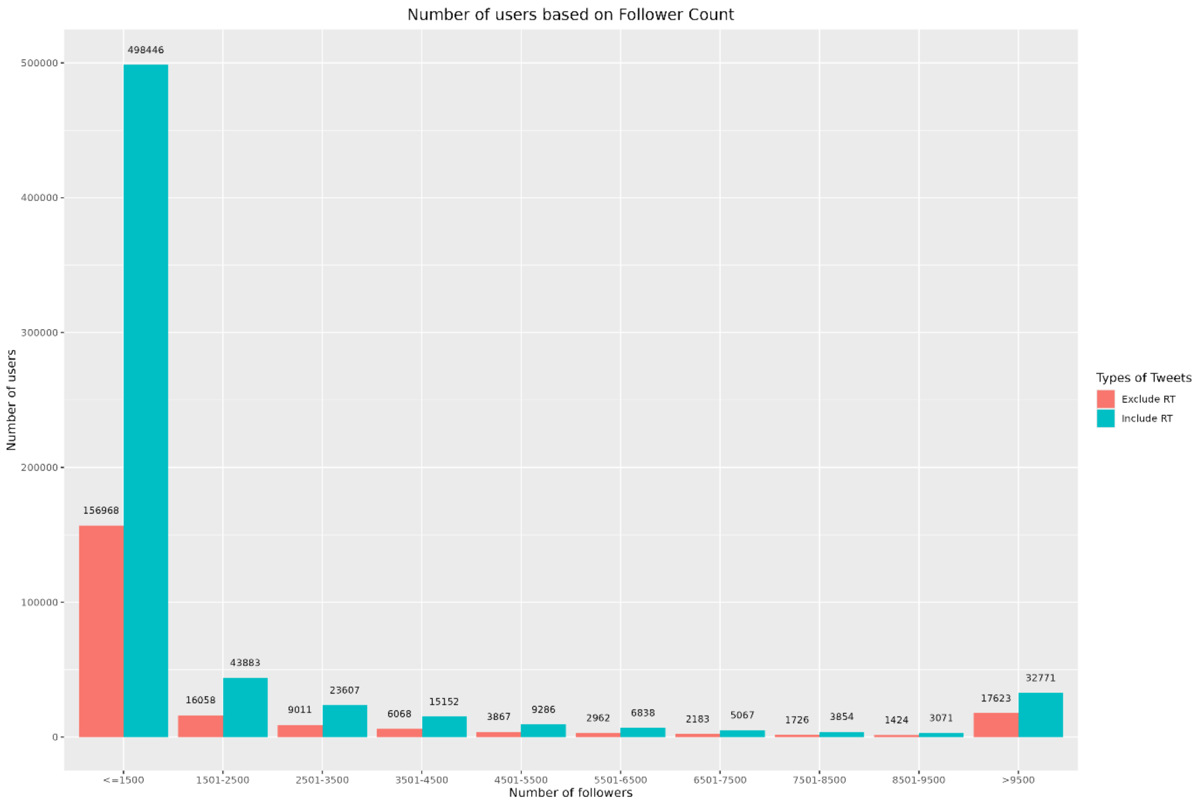
Plot the side by side bar chart by using ggplot.

```
#converts number of followers column to factor
#manually specify the order of the levels to suit the x column property
> final_df$Number.of.followers <-
factor(final_df$Number.of.followers,levels=unique(final_df$Number.of.followers ))

#use ggplot to plot the side-by-side bar chart
>  ggplot(final_df,   aes(x=final_df$Number.of.followers, y=final_df$Number.of.users,
fill=final_df$Types_of_Tweets),width=50) + coord_cartesian(ylim = c(0,500000)) +
  geom_col(position = 'dodge') +
  ggtitle('Number of users based on Follower Count') +
  xlab("Number of followers") + ylab("Number of users") +
  theme(plot.title = element_text(size = 14, hjust = 0.5)) +
  guides(fill=guide_legend(title="Types of Tweets")) +
  geom_text(aes(label=final_df$Number.of.users, y=final_df$Number.of.users + 10000),
           position=position_dodge(width=0.9),vjust=0.25,size = 3)

#save the plotted graph to a png with specific dimension.
> ggsave("grouped_bar.png", width = 15, height = 10, units = 'in' ,limitsize = FALSE)
```

Explanation of using ggplot:

- Passed in the data frame formed (‘final_df’)
- ‘aes’ refers to aesthetic mappings which maps variables to a visual property. ‘x’ refers to the variable on the x-axis, y refers to the variable on y-axis, fill used to map the variable to the categories (Include or exclude RT data).
- ‘coord_cartesian’ used to limit the y plot axes.
- ‘geom_col’ creates bar plots, and position ‘dodge’ refers to creating the grouped bar chart.
- ‘ggtitle’ add the title to the plot, ‘xlab’ and ‘ylab’ add the label to both axis.
- ‘guides’ used to change the title of the legend.
- ‘geom_text’ add values to each bar plotted.

Output:



5) The bar chart plotted above represents the number of users based on the follower's count. Each category of the number of followers shows the number of unique users that tweets and not retweets. By observing the graph, we can clearly see that the dataset collected mainly consist of the users with followers <= 1500. Next, we can observe a significant decrease starting from the users with number of followers between 1501 to 9500. The decline supports the social networks where a smaller number of users are able to achieve high number of followers.

In every number of followers' category, the number of including non-RT users consistently exceeds the number of excluding non-RT users. Hence, resulting both categories possess same trends of data.

Users with less followers tend to tweet more than retweet other data might be because they are more willing to share their thoughts and content. Due to fewer followers, it poses a lower risk of exposure from being conform or judged by others, eventually encouraging more original posts. In other words, they are less concerned about how their tweets are perceived. As users gain more followers, they are more conscious about their online presence to the followers and the comments they tweeted. As more followers increase the visibility of their tweets towards the society. Hence, this influences their tendency to retweet rather than posting the original content. In short, the analysis of the bar chart shows trends in user behaviour towards retweet among the users with different follower counts.

Conclusion

The large dataset is successfully be processed using BASH Shell Scripts. The data has been aggregated and filtered in order to perform analysis towards user behaviour. With the help of R programming language, we are able to read the data and generate good visualisation graphs such as bar charts to present our findings.

References

Kai Yuan. (2024, March 18). *Remove Duplicate Lines from a File Without Sorting*. Baeldung. https://www.baeldung.com/linux/remove-duplicate-lines-no-sorting#:~:text=We've%20learned%20we%20can,%240%20means%20the%20whole%20line

Prajwal CN. (2022, August 4). *The rbind() function in R - Binding Rows Made Easy | DigitalOcean*. Www.digitalocean.com. https://www.digitalocean.com/community/tutorials/rbind-function-r

Rech, R. (2021, June 16). *Barplot for Two Factors in R – Step-by-Step Tutorial*. Rosane Rech. https://statdoe.com/barplot-for-two-factors-in-r/#:~:text=The%20default%20barplot%20from%20ggplot

The handbook team. (2023). 30 ggplot basics | The Epidemiologist R Handbook. In *epirhandbook.com*. https://epirhandbook.com/en/ggplot-basics.html