FIT 1043

ASSIGNMENT 3

Full Name: George Tan Juan Sheng

Student ID: 30884128

Class Number: FIT 1043

Introduction

For this assignment, we will be focusing on using BASH shell scripts to process and extract data from datasets and visualize the summarized data using plots generated by the R programming language. This assignment is also done by using the server constructed by Dr Ian.

Part A: Inspecting the data

1) **Answer**:
   The size of the corona_tweets.csv.gz file is 118 MegaBytes.

   Shell command:
   ls -lh corona_tweets.csv.gz

2) **Answer**:
   Created, Tweet_ID, Text, User_ID, User, User_Location, Followers_Count, Friends_Count, Geo, Place_Type, Place_Name, Place_Country, Language

   Shell command:
   cat corona_tweets.csv.gz |gunzip|head -1|less

3) **Answer**:
   There are 1143559 lines in the dataset.

   Shell command:
   cat corona_tweets.csv.gz |gunzip|wc -l

Part B: Information from data

1) Answer:
   There are 641976 unique twitter users in the dataset.

Shell command:
cat corona_tweets.csv.gz |gunzip|awk -F'\t' '{print $4}' | sort|uniq| wc -l

Explanation:
For this question, we had used cat, gunzip,awk,sort, uniq and wc to get the answer. By using these, we did not decompress the file into the server and was still able to ge the answer. Uniq was also sufficient to answer this question if the output was sorted before, this is because uniq works only if the lines are adjacent, hence as after sorting the output becomes adjacent to each other, the uniq command will be able to remove the user id that had appeared more than twice and output the remaining user id. The command wc is then used to get the number of user id and this would be our answer.

2)

a) **Answer**:

There are 50911 tweets mentioned the word ‘death’ in any combination of uppercase or lowercase letters.

Shell command:
cat corona_tweets.csv.gz |gunzip| awk -F'\t' '{print $3}' |grep -i death|wc -l

Explanation:
In order to grep the letter in any combination of uppercase or lowercase letters, we have to use grep -i ‘death’ instead of grep ‘death’ alone as grep -i ‘death’ gets the lines that contains the any combination of uppercase or lowercase letter of the letter ‘death’, and this fulfils the question’s requirement.

b) **Answer**:

There are 450 of those that are not spelt exactly “death”, “deaths”, “Death” or “Deaths” but in other combination of uppercase and lowercase.

Shell command:

cat corona_tweets.csv.gz |gunzip|awk -F'\t' '{print $3}'| grep -i 'death'|grep -v 'death\|deaths\|Death\|Deaths' | wc -l

Explanation:
For this question, we had used grep -i ‘death’ in order to get all the lines that contains any combination of uppercase or lowercase letter of the letter ‘death’, we then pipe it to grep the lines that do not contain the word ‘death’, ‘deaths’, ‘Death’ or ‘Deaths’, in the end we pipe it to a wc command and we will get the number tweets.

c) **Answer**:

cat corona_tweets.csv.gz |gunzip|awk -F'\t' '{print $3}'| grep -i 'death'|grep -v 'death\|deaths\|Death\|Deaths' >myText.txt

Explanation:
In order to output the lines to a new file, ‘>’ symbol is used to output the output of the command into a file. For this case, our file will be called myText.txt, hence after the command to get the lines of sub-part 2(b), we will output it to a file by adding a >myText.txt at the back of the command.

Part C: Data aggregation

1) a) **Answer**:

cat corona_tweets.csv.gz |gunzip|awk -F'\t' '$7<=1000 {print $4}'|sort -u

b) **Answer**:
cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7>=1001 && $7<=2000 {print $4}' |sort -u

c) **Answer**:
cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7>=2001 && $7<=3000 {print $4}' |sort -u

d) **Answer**:

cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7>=3001 && $7<=4000 {print $4}' |sort -u

e) **Answer**:

cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7>=4001 && $7<=5000 {print $4}' |sort -u

f) **Answer**:

cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7>=5001 && $7<=6000 {print $4}' |sort -u

g) **Answer**:

cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7>=6001 && $7<=7000 {print $4}' |sort -u

h) **Answer**:

cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7>=7001 && $7<=8000 {print $4}' |sort -u

i) **Answer**:

cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7>=8001 && $7<=9000 {print $4}' |sort -u

j) **Answer**:

cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7>=9001 && $7<=10000 {print $4}' |sort -u

k) **Answer**:

cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7>10000 {print $4}’|sort -u

2)

Answer:

```
#!/bin/bash
echo '<=1k',$(cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7<=1000 {print $4}' |sort -u|wc -l)>>assignment2.csv

echo 1k-2k,$(cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7>=1001 && $7<=2000 {print $4}' |sort -u|wc -l)>>assignment2.csv
```

```
echo 2k-3k,$(cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7>=2001 && $7<=3000 {print $4}' |sort -u|wc -l)>>assignment2.csv

echo 3k-4k,$(cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7>=3001 && $7<=4000 {print $4}' |sort -u|wc -l)>>assignment2.csv

echo 4k-5k,$(cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7>=4001 && $7<=5000 {print $4}' |sort -u|wc -l)>>assignment2.csv

echo 5k-6k,$(cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7>=5001 && $7<=6000 {print $4}' |sort -u|wc -l)>>assignment2.csv

echo 6k-7k,$(cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7>=6001 && $7<=7000 {print $4}' |sort -u|wc -l)>>assignment2.csv

echo 7k-8k,$(cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7>=7001 && $7<=8000 {print $4}' |sort -u|wc -l)>>assignment2.csv

echo 8k-9k,$(cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7>=8001 && $7<=9000 {print $4}' |sort -u|wc -l)>>assignment2.csv

echo 9k-10k,$(cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7>=9001 && $7<=10000 {print $4}' |sort -u|wc -l)>>assignment2.csv

echo '>10k',$(cat corona_tweets.csv.gz|gunzip|awk -F'\t' '$7>10000 {print $4}' |sort -u|wc -l)>>assignment2.csv
```

The CSV file would look like this:

	A	B	C
1	<=1k	455758	
2	1k-2k	68574	
3	2k-3k	31281	
4	3k-4k	18803	
5	4k-5k	11699	
6	5k-6k	7985	
7	6k-7k	5875	
8	7k-8k	4368	
9	8k-9k	3525	
10	9k-10k	2738	
11	>10k	31473	

3,4)

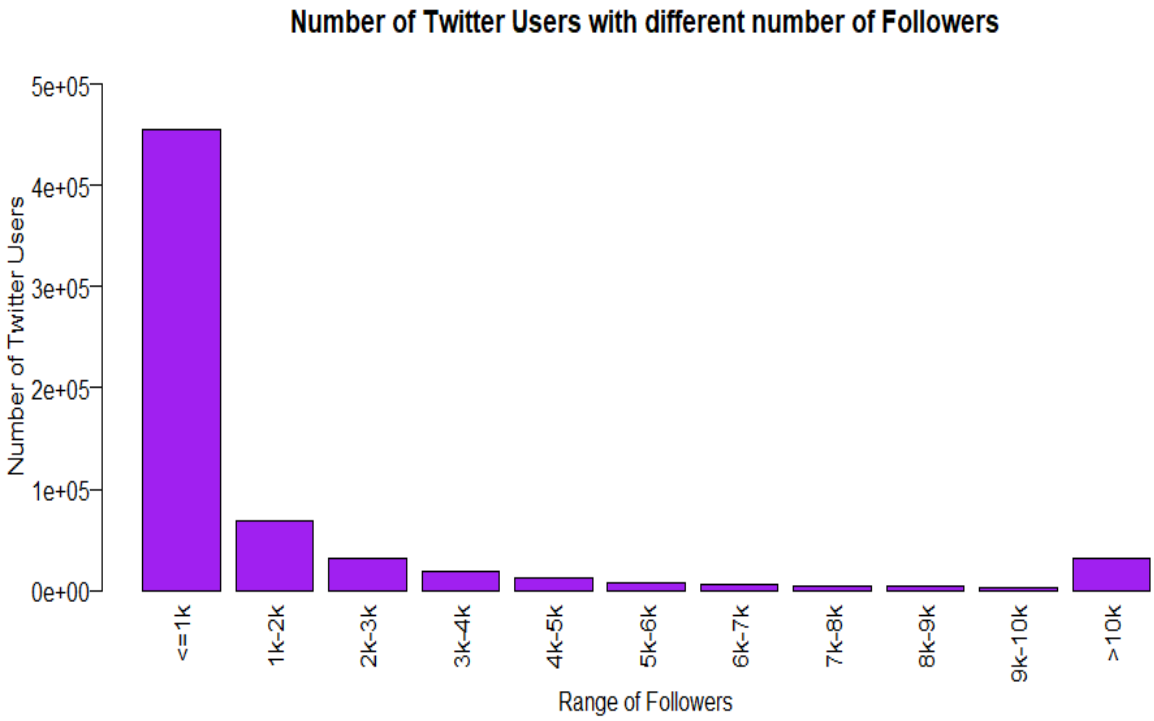
R Code:

```
setwd('C:/Users/georg/Desktop')

assignmentpartC <- read.table('assignment2.csv', header=FALSE, sep = ',' )

barplot(assignmentpartC$V2,names.arg = assignmentpartC$V1,las =2, mgp = c(3,0.5,0),ylim=c(0,500000),col = 'purple',main = 'Number of Twitter Users with different number of Followers',xlab='Range of Followers',ylab='Number of Twitter Users')
```

PNG Image of Bar chart created:



Part D: Small Challenge

1) **Shell command**:

   cat corona_tweets.csv.gz | gunzip | grep -v 'RT @' |gzip > assignment2partD1.csv.gz

2) **Bash Code**:

(Shell command in order to group the twitter users according to their number of followers)

```
cat assignment2partD1.csv.gz |gunzip|awk -F'\t' '$7<=1000 {print $4}'|sort -u
```

```
cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7>=1001 && $7<=2000
{print $4}' |sort -u

cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7>=2001 && $7<=3000
{print $4}' |sort -u
cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7>=3001 && $7<=4000
{print $4}' |sort -u

cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7>=4001 && $7<=5000
{print $4}' |sort -u

cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7>=5001 && $7<=6000
{print $4}' |sort -u

cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7>=6001 && $7<=7000
{print $4}' |sort -u

cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7>=7001 && $7<=8000
{print $4}' |sort -u

cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7>=8001 && $7<=9000
{print $4}' |sort -u

cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7>=9001 && $7<=10000
{print $4}' |sort -u

cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7>10000 {print $4}'|sort -u
```

(Codes in .sh file)

```
#!/bin/bash
echo '<=1k',$(cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7<=1000
{print $4}' |sort -u|wc -l)>>assignment2partDoutput.csv

echo 1k-2k,$(cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7>=1001 &&
$7<=2000 {print $4}' |sort -u|wc -l)>>assignment2partDoutput.csv
```

```
echo 2k-3k,$(cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7>=2001 &&
$7<=3000 {print $4}' |sort -u|wc -l)>>assignment2partDoutput.csv

echo 3k-4k,$(cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7>=3001 &&
$7<=4000 {print $4}' |sort -u|wc -l)>>assignment2partDoutput.csv

echo 4k-5k,$(cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7>=4001 &&
$7<=5000 {print $4}' |sort -u|wc -l)>>assignment2partDoutput.csv

echo 5k-6k,$(cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7>=5001 &&
$7<=6000 {print $4}' |sort -u|wc -l)>>assignment2partDoutput.csv

echo 6k-7k,$(cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7>=6001 &&
$7<=7000 {print $4}' |sort -u|wc -l)>>assignment2partDoutput.csv

echo 7k-8k,$(cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7>=7001 &&
$7<=8000 {print $4}' |sort -u|wc -l)>>assignment2partDoutput.csv

echo 8k-9k,$(cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7>=8001 &&
$7<=9000 {print $4}' |sort -u|wc -l)>>assignment2partDoutput.csv

echo 9k-10k,$(cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7>=9001
&& $7<=10000 {print $4}' |sort -u|wc -l)>>assignment2partDoutput.csv

echo '>10k',$(cat assignment2partD1.csv.gz|gunzip|awk -F'\t' '$7>10000
{print $4}' |sort -u|wc -l)>>assignment2partDoutput.csv
```

The CSV file from the command would look like this:

	A	B	C
1	<=1k	142249	
2	1k-2k	24039	
3	2k-3k	11771	
4	3k-4k	7348	
5	4k-5k	4726	
6	5k-6k	3462	
7	6k-7k	2498	
8	7k-8k	1908	
9	8k-9k	1611	
10	9k-10k	1279	
11	>10k	17017	

3)

R Code:

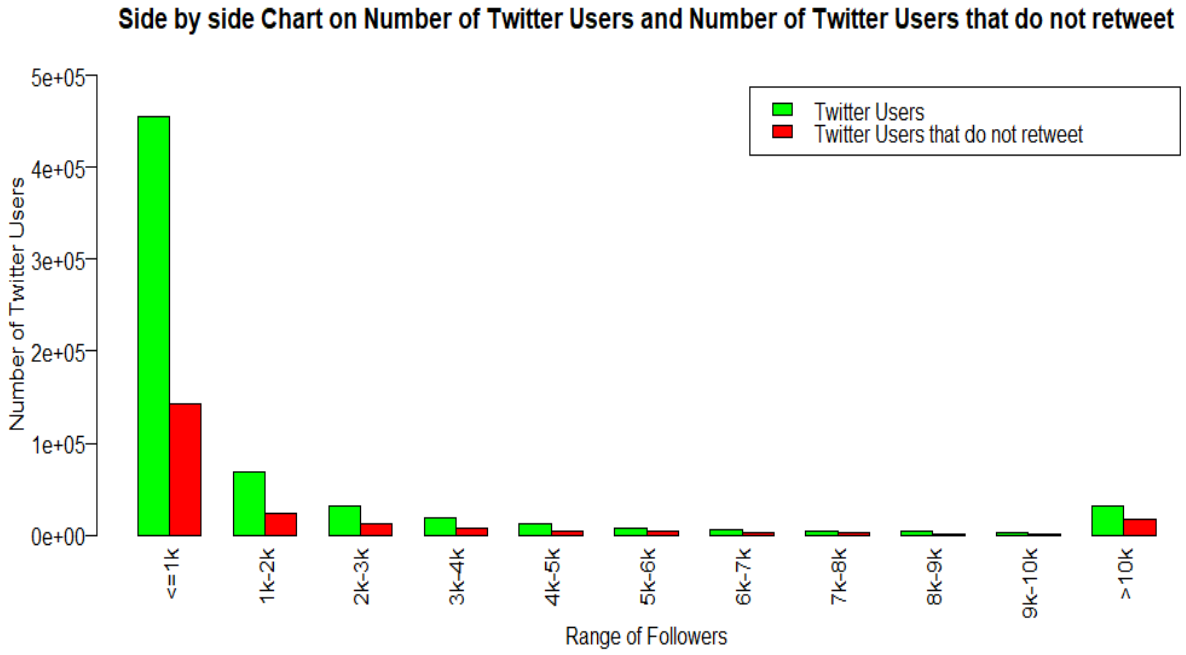
```
setwd('C:/Users/georg/Desktop')

assignmentpartC <- read.table('assignment2.csv', header=FALSE, sep = ',' )

assignmentpartD <- read.table('assignment2partDoutput.csv',header=FALSE, sep = ',')

output <- rbind (assignmentpartC$V2, assignmentpartD$V2)

barplot(output,beside=TRUE, names.arg = assignmentpartC$V1,las =2, mgp = c(3,0.5,0),ylim=c(0,500000),col =c('green','red'), legend = c('Twitter Users','Twitter Users that do not retweet'), main = 'Side by side Chart on Number of Twitter Users and Number of Twitter Users that do not retweet',xlab='Range of Followers',ylab='Number of Twitter Users')
```



From the side by side bar chart above, we can clearly see that the number of Twitter Users are more than then number of Twitter Users that do not retweet in all range of followers.

Conclusion

Conclusion, BASH scripting was confusing at first however after this assignment I was able to familiarize myself with BASH scripts. R visualizations were also a new experience as visualizations were done with Python before this. However, when it comes to visualization, R is preferable than Python. All in all, I would like to say that the objective of this assignment was achieved, and I am looking forward to learn more about Data Science. I would also like to thank Dr Ian for all the guidance provided.