FIT1043 Intro to Data Science

Assignment 3

Naveed Hassan

32799047

Table of Contents

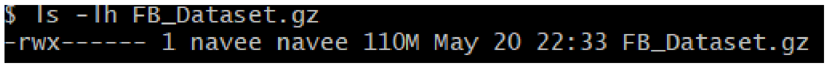
Task A1: 3
Task A2: 4
Task A3: 5
Task A4: 6
Task A5: 7
Task A6: 8
Task A7: 9
Task A8: 9
Task A9: 10
Task A10: 11
Task B1: 14
Task B2: 15
i).. 15
ii)... 17
iii).. 18

Task A1:

To find the original size of the file, we use the command:

```
ls -lh FB_Dataset.gz
```

Output:



```
$ ls -lh FB_Dataset.gz
-rwx------ 1 navee navee 110M May 20 22:33 FB_Dataset.gz
```

Image 1.1: Output of initial size of the file

Hence we can conclude that the initial size of the file is 110 MB

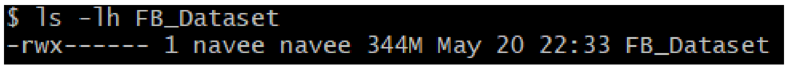
To decompress the file:

```
gunzip FB_Dataset.gz
```

Check the size of the decompressed file:

```
ls -lh FB_Dataset
```

Output:



```
$ ls -lh FB_Dataset
-rwx------ 1 navee navee 344M May 20 22:33 FB_Dataset
```

Image 1.2: Output of size of the decompressed file

Conclusion:

The original size of the file is 110 MB, After decompression, the size of the file is 344 MB.

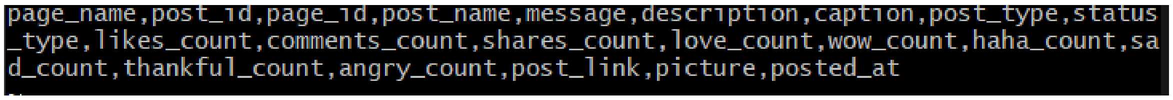
Task A2:

We know that the 1st line is the header

Command used:

```
head -1 FB_Dataset | less
```

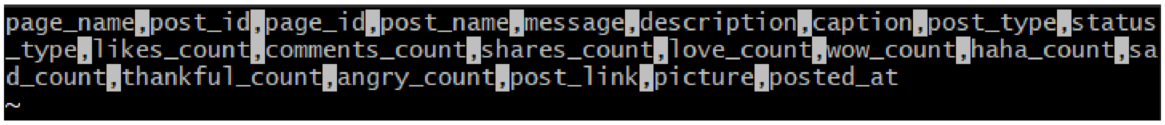
Output:



```
page_name,post_id,page_id,post_name,message,description,caption,post_type,status
_type,likes_count,comments_count,shares_count,love_count,wow_count,haha_count,sa
d_count,thankful_count,angry_count,post_link,picture,posted_at
```

Image 2.1: Output of the 1st line of the file.

Then we use /, to confirm where commas are



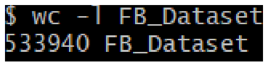
```
page_name,post_id,page_id,post_name,message,description,caption,post_type,status
_type,likes_count,comments_count,shares_count,love_count,wow_count,haha_count,sa
d_count,thankful_count,angry_count,post_link,picture,posted_at
~
```

Image 2.2: Output showing the delimiters.

Command Used:

```
wc -l FB_Dataset
```

Output:



```
$ wc -l FB_Dataset
533940 FB_Dataset
```

Image 2.3: Output showing the number of lines in the file

Conclusion:

The delimiter used in this case is comma. The number of rows is 533,940.

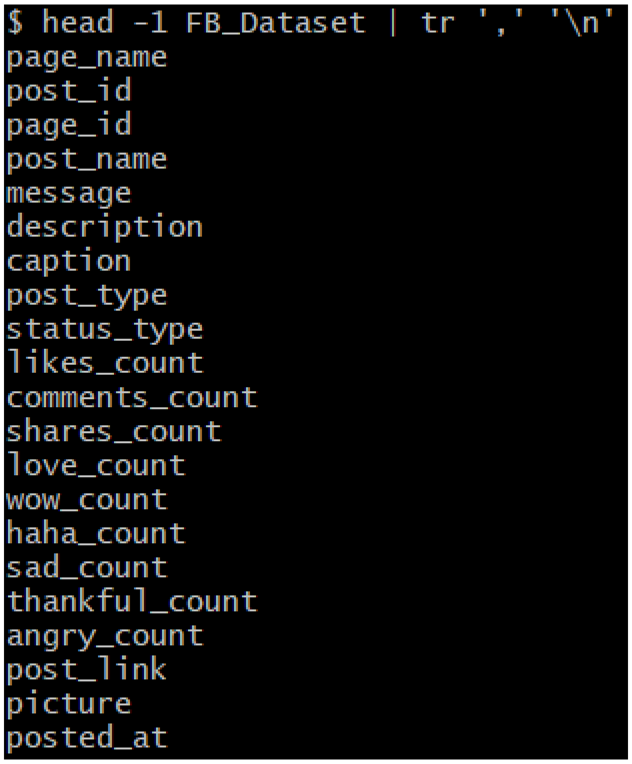
Task A3:

To identify the columns we output the first line and replace the comma with a new line by using the tr command.

Command used:

```
head -1 FB_Dataset | tr ',' '\n'
```

Output:



```
$ head -1 FB_Dataset | tr ',' '\n'
page_name
post_id
page_id
post_name
message
description
caption
post_type
status_type
likes_count
comments_count
shares_count
love_count
wow_count
haha_count
sad_count
thankful_count
angry_count
post_link
picture
posted_at
```

Image 3.1: Output of the column names of the file

Conclusion:

The file contains 21 columns.

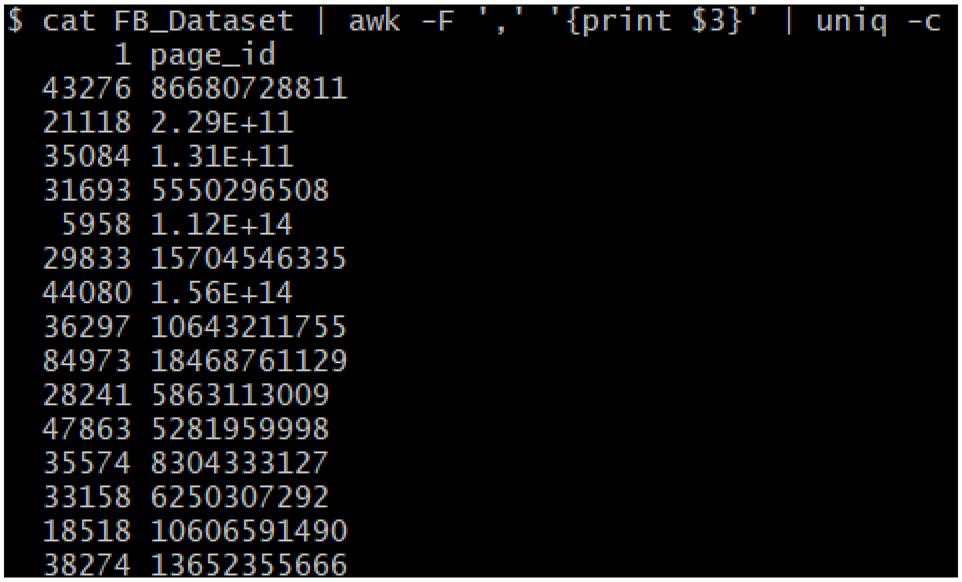
Task A4:

Number of Unique pages can be thought of as the number of pages with different page_id

Command used:

```
cat FB_Dataset | awk -F ',' '{print $3}' | uniq -c
```

Output:



```
$ cat FB_Dataset | awk -F ',' '{print $3}' | uniq -c
      1 page_id
  43276 86680728811
  21118 2.29E+11
  35084 1.31E+11
  31693 5550296508
   5958 1.12E+14
  29833 15704546335
  44080 1.56E+14
  36297 10643211755
  84973 18468761129
  28241 5863113009
  47863 5281959998
  35574 8304333127
  33158 6250307292
  18518 10606591490
  38274 13652355666
```

Image 4.1: Showing the unique pages in the dataset

Conclusion:

There are a total of 15 unique pages in this dataset.

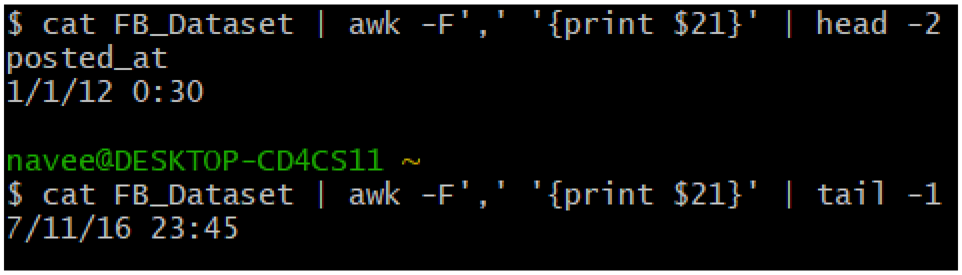
Task A5:

Since we can assume that the data is arranged in chronological order, we can take the posted_at of the 1st row of data and the last row of the data to determine the date range for the Facebook posts.

Commands Used:

```
cat FB_Dataset | awk -F',' '{print $21}' | head -2
cat FB_Dataset | awk -F',' '{print $21}' | tail -1
```

Output:



```
$ cat FB_Dataset | awk -F',' '{print $21}' | head -2
posted_at
1/1/12 0:30

navee@DESKTOP-CD4CS11 ~
$ cat FB_Dataset | awk -F',' '{print $21}' | tail -1
7/11/16 23:45
```

Image 5.1: Output of the 1st and last rows of data

Conclusion:

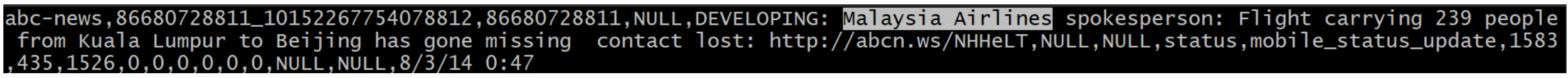
The date range for the Facebook posts given in this dataset is 1/1/12 – 7/11/16.

Task A6:

Commands Used:

```
Less FB_Dataset
/Malaysia Airlines
```

Output:



```
abc-news,86680728811_10152267754078812,86680728811,NULL,DEVELOPING: Malaysia Airlines spokesperson: Flight carrying 239 people
 from Kuala Lumpur to Beijing has gone missing  contact lost: http://abcn.ws/NHHeLT,NULL,NULL,status,mobile_status_update,1583
,435,1526,0,0,0,0,0,0,NULL,NULL,8/3/14 0:47
```

Image 6.1: Finding the first instance of Malaysia Airlines

Conclusion:

Date: 8/3/14 0:47

Message: Malaysia Airlines spokesperson: Flight carrying 239 people from Kuala Lumpur to Beijing has gone missing contact lost: http://abcn.ws/NHHeLT

Media Source: ABC News

Source:

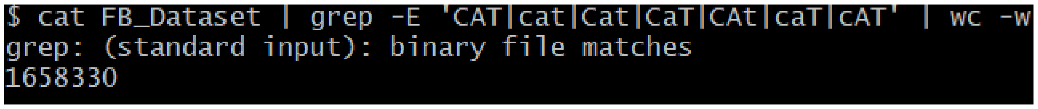
ABC News, 2014. *Malaysia Airlines Flight Vanishes, Three Americans on Board*. [online] ABC News. Available at: https://abcnews.go.com/International/malaysia-airlines-flight-missing-en-route-china/story?id=22827892

Task A7:

Command Used:

```
cat FB_Dataset | grep -E 'CAT|cat|Cat|CaT|CAt|caT|cAT' | wc -w
```

Output:



```
$ cat FB_Dataset | grep -E 'CAT|cat|Cat|CaT|CAt|caT|cAT' | wc -w
grep: (standard input): binary file matches
1658330
```

Image 7.1: The number of times Cat appears in the message column of the dataset

Conclusion:

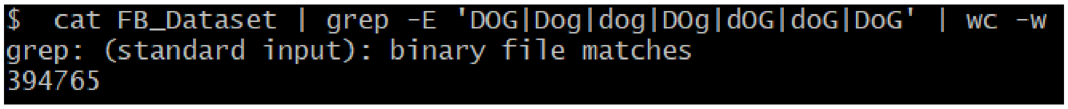
Cat is mentioned 1,658,330 times in the messages column without ignoring all upper and lower case possibilities.

Task A8:

Command Used:

```
cat FB_Dataset | grep -E 'DOG|Dog|dog|DOg|dOG|doG|DoG' | wc -w
```

Output:



```
$  cat FB_Dataset | grep -E 'DOG|Dog|dog|DOg|dOG|doG|DoG' | wc -w
grep: (standard input): binary file matches
394765
```

Image 8.1: The number of times Dog appears in the message column of the dataset

Conclusion:

Dog is mentioned 394,765 times in the messages column without ignoring all upper and lower case possibilities.

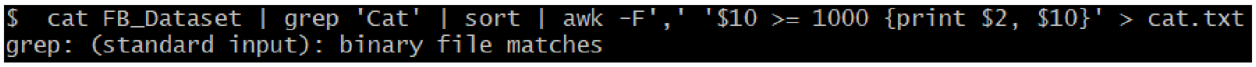
Hence we can conclude that Cats are more popular than dogs, considering the definition of popularity as the quality of being recognized by many people.

Task A9:

Commands Used:

```
cat FB_Dataset | grep 'Cat' | sort | awk -F',' '$10 >= 1000 {print $2, $10}' > cat.txt
```

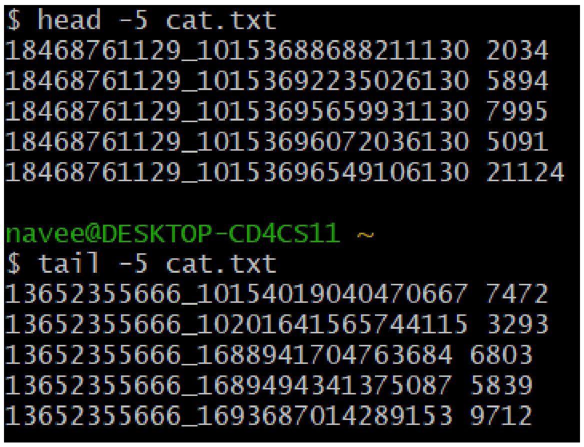
Output:



```
$  cat FB_Dataset | grep 'Cat' | sort | awk -F',' '$10 >= 1000 {print $2, $10}' > cat.txt
grep: (standard input): binary file matches
```

Image 9.1: The text file created from the messages containing Cat

Output the first and last 5 rows of data:



```
$ head -5 cat.txt
18468761129_10153688688211130 2034
18468761129_10153692235026130 5894
18468761129_10153695659931130 7995
18468761129_10153696072036130 5091
18468761129_10153696549106130 21124

navee@DESKTOP-CD4CS11 ~
$ tail -5 cat.txt
13652355666_10154019040470667 7472
13652355666_10201641565744115 3293
13652355666_1688941704763684 6803
13652355666_1689494341375087 5839
13652355666_1693687014289153 9712
```

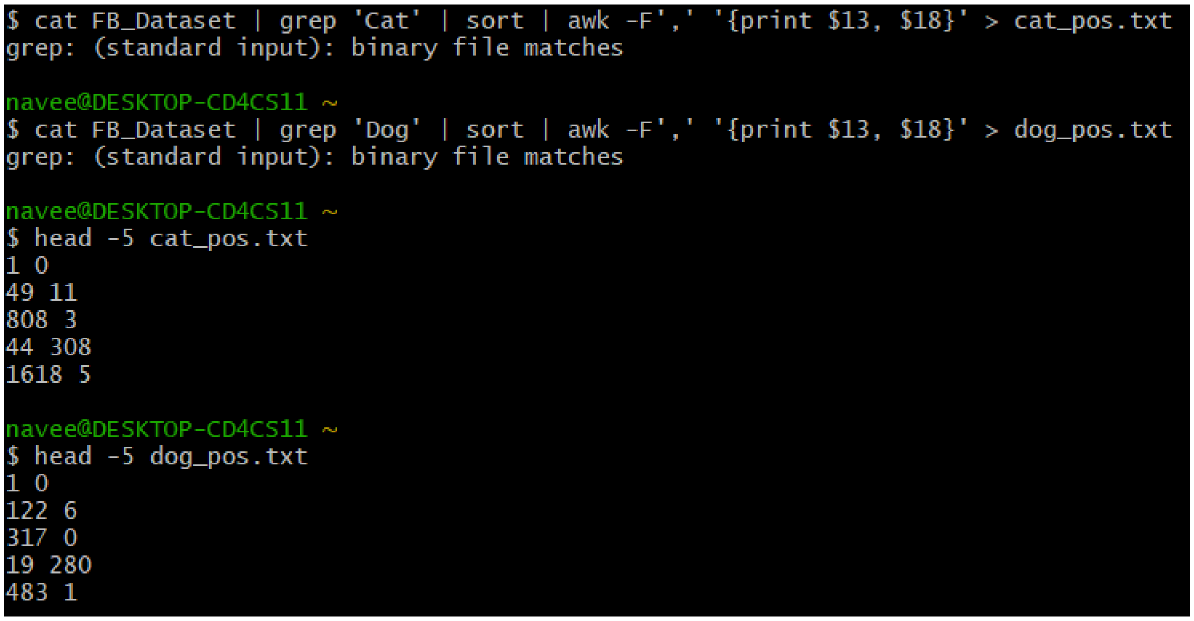
Image 9.2: The text file created from the messages containing Cat

Task A10:

Command Used:

```
cat FB_Dataset | grep 'Cat' | sort | awk -F',' '{print $13, $18}' > cat_pos.txt
cat FB_Dataset | grep 'Dog' | sort | awk -F',' '{print $13, $18}' > dog_pos.txt
```

Output:



```
$ cat FB_Dataset | grep 'Cat' | sort | awk -F',' '{print $13, $18}' > cat_pos.txt
grep: (standard input): binary file matches

navee@DESKTOP-CD4CS11 ~
$ cat FB_Dataset | grep 'Dog' | sort | awk -F',' '{print $13, $18}' > dog_pos.txt
grep: (standard input): binary file matches

navee@DESKTOP-CD4CS11 ~
$ head -5 cat_pos.txt
1 0
49 11
808 3
44 308
1618 5

navee@DESKTOP-CD4CS11 ~
$ head -5 dog_pos.txt
1 0
122 6
317 0
19 280
483 1
```

Image 10.1: The text file created from the messages containing Cat and Dog

Then we copy the files to a different directory to work on it in R.

```
cp cat_pos.txt /cygdrive/c/Users/navee/OneDrive/Desktop/A3
cp dog_pos.txt /cygdrive/c/Users/navee/OneDrive/Desktop/A3
```

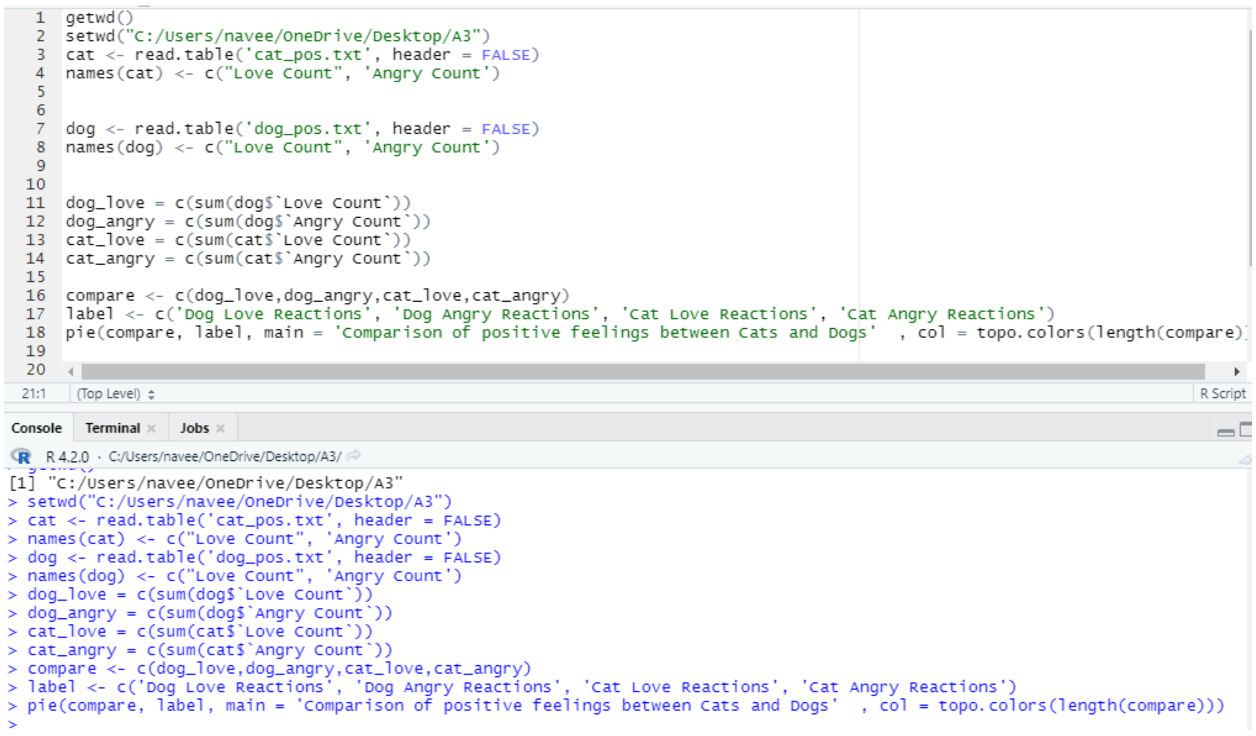
R Part:

Code:

```r
getwd()

setwd("C:/Users/navee/OneDrive/Desktop/A3")

cat <- read.table('cat_pos.txt', header = FALSE)

names(cat) <- c("Love Count", 'Angry Count')



dog <- read.table('dog_pos.txt', header = FALSE)

names(dog) <- c("Love Count", 'Angry Count')



dog_love = c(sum(dog$`Love Count`))

dog_angry = c(sum(dog$`Angry Count`))

cat_love = c(sum(cat$`Love Count`))

cat_angry = c(sum(cat$`Angry Count`))


compare <- c(dog_love,dog_angry,cat_love,cat_angry)

label <- c('Dog Love Reactions', 'Dog Angry Reactions', 'Cat Love Reactions', 'Cat
Angry Reactions')

pie(compare, label, main = 'Comparison of positive feelings between Cats and Dogs'  ,
col = topo.colors(length(compare)))
```

We plot a pie chart to compare the reactions to the posts about cats and dogs.

Below shows the R Code used to produce the pie chart.



```
getwd()
setwd("C:/Users/navee/OneDrive/Desktop/A3")
cat <- read.table('cat_pos.txt', header = FALSE)
names(cat) <- c("Love Count", 'Angry Count')


dog <- read.table('dog_pos.txt', header = FALSE)
names(dog) <- c("Love Count", 'Angry Count')


dog_love = c(sum(dog$`Love Count`))
dog_angry = c(sum(dog$`Angry Count`))
cat_love = c(sum(cat$`Love Count`))
cat_angry = c(sum(cat$`Angry Count`))

compare <- c(dog_love,dog_angry,cat_love,cat_angry)
label <- c('Dog Love Reactions', 'Dog Angry Reactions', 'Cat Love Reactions', 'Cat Angry Reactions')
pie(compare, label, main = 'Comparison of positive feelings between Cats and Dogs'  , col = topo.colors(length(compare)
```

```
[1] "C:/Users/navee/OneDrive/Desktop/A3"
> setwd("C:/Users/navee/OneDrive/Desktop/A3")
> cat <- read.table('cat_pos.txt', header = FALSE)
> names(cat) <- c("Love Count", 'Angry Count')
> dog <- read.table('dog_pos.txt', header = FALSE)
> names(dog) <- c("Love Count", 'Angry Count')
> dog_love = c(sum(dog$`Love Count`))
> dog_angry = c(sum(dog$`Angry Count`))
> cat_love = c(sum(cat$`Love Count`))
> cat_angry = c(sum(cat$`Angry Count`))
> compare <- c(dog_love,dog_angry,cat_love,cat_angry)
> label <- c('Dog Love Reactions', 'Dog Angry Reactions', 'Cat Love Reactions', 'Cat Angry Reactions')
> pie(compare, label, main = 'Comparison of positive feelings between Cats and Dogs'  , col = topo.colors(length(compare)))
>
```

Pie Chart Output:

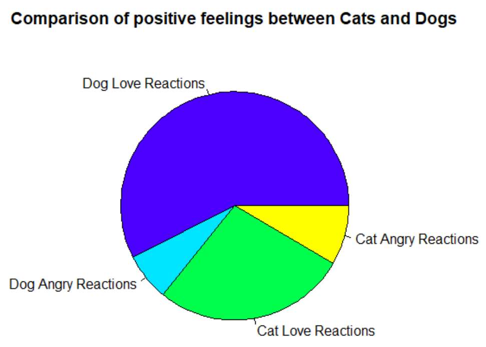


Figure 10.1: Pie Chart comparing positive feelings between Cats And Dogs

Conclusion:

Using the simple visualization of a pie chart, it can be easily seen that Dogs have the majority of love reactions and minority of Angry reactions as compared to that of Cats. Hence using these ideas, we can conclude that Dogs produce more positive feeling as opposed to Cats.

Task B1:

Commands used:

```
cat FB_Dataset | grep 'Dog' | sort | awk -F',' '{print $21}' > dog_time.txt
```

Then copy into R working directory:

```
cp dog_time.txt /cygdrive/c/Users/navee/OneDrive/Desktop/A3
```

R Code:

```
dogtime <- read.table('dog_time.txt', header = FALSE)

date <- data.frame(paste(dogtime$V1, dogtime$V2))

names(date) <- c("Date")

date$Date <- strptime(date$Date, format='%D/%M/%y %H:%M')

class(date$Date)

install.packages('plotly')

library(plotly)



ggplot(date, aes(x=Date)) +
  geom_histogram() +
  labs(x="Date", y="Frequency") +
  scale_x_date(date_breaks = "months", date_labels = "%d/%m/%Y %H:%M",
               limits=c(as.Date("6/1/12"), as.Date("7/11/16")))
```

Sources:

RStudio Community. 2021. *How to properly plot a histogram with dates using ggplot?*. [online] Available at: https://community.rstudio.com/t/how-to-properly-plot-a-histogram-with-dates-using-ggplot/97444

Understanding dates and plotting a histogram with ggplot2 in R. [online] Stack Overflow. Available at: https://stackoverflow.com/questions/10770698/understanding-dates-and-plotting-a-histogram-with-ggplot2-in-r

Task B2:

i)

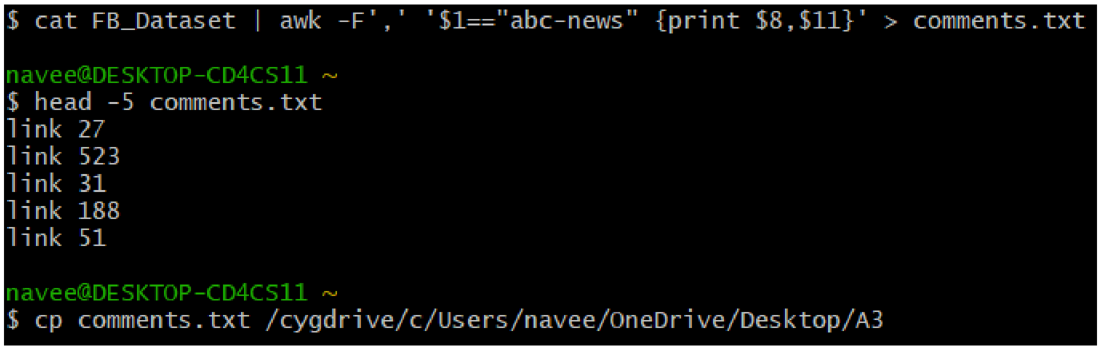
First we create the relevant text file and check if its created correctly by looking at its header.

Then we copy the file to the R Studio working directory.

Commands used:

```
cat FB_Dataset | awk -F',' '$1=="abc-news" {print $8,$11}' > comments.txt
head -5 comments.txt
cp comments.txt /cygdrive/c/Users/navee/OneDrive/Desktop/A3
```

Output:



```
$ cat FB_Dataset | awk -F',' '$1=="abc-news" {print $8,$11}' > comments.txt

navee@DESKTOP-CD4CS11 ~
$ head -5 comments.txt
link 27
link 523
link 31
link 188
link 51

navee@DESKTOP-CD4CS11 ~
$ cp comments.txt /cygdrive/c/Users/navee/OneDrive/Desktop/A3
```

Image 12.1: The text file created for comparison of commenst

R Code:

```
comments <- read.table('comments.txt', header = FALSE)
names(comments) <- c("Post Type", "No. of Comments")
library(dplyr)
boxplot <- ggplot(comments, aes(x = `Post Type`, y = `No. of Comments`)) +
geom_boxplot()
boxplot
```

Boxplot Output:

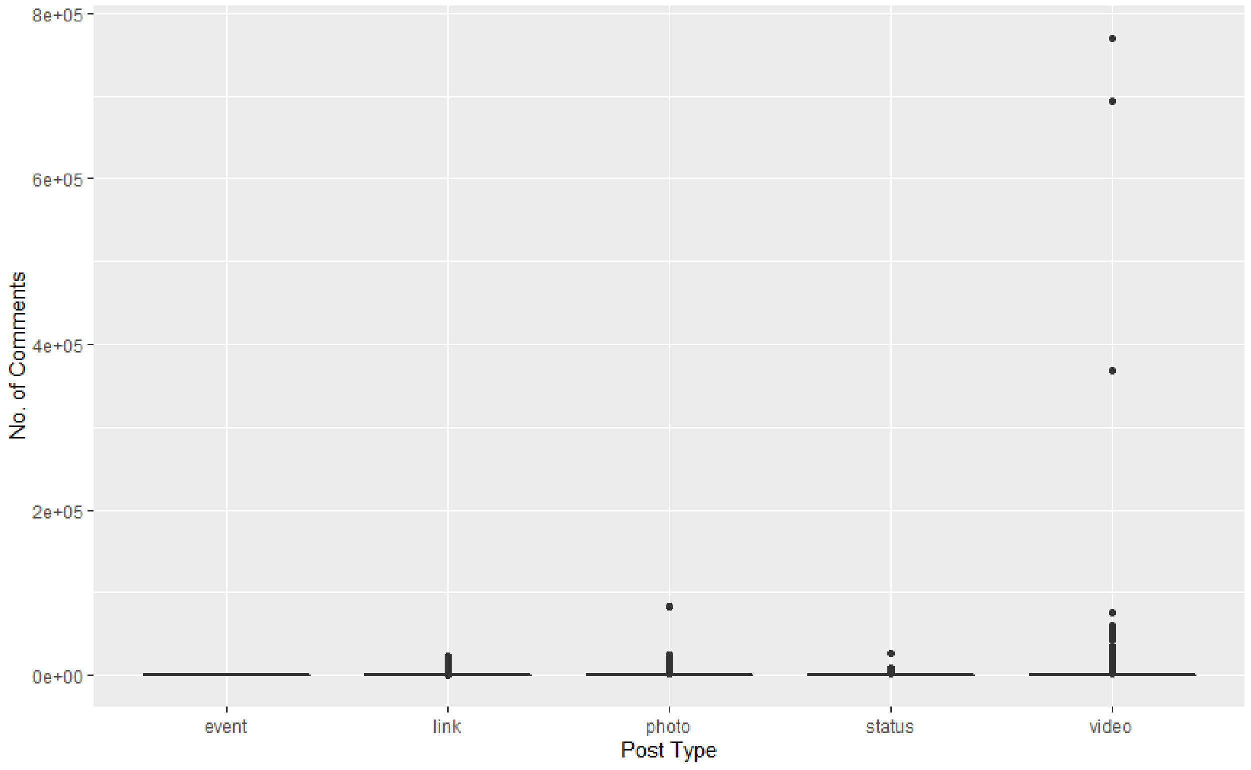


Figure 12.1: Boxplot of the number of comments for each post type

Conclusion:

Although the boxplot is difficult to read, we can conclude from the readable values that videos and photos produce the most interaction from users through comments in comparison to other post types.

ii)

We use the dplyr library and filter out the values of comments that are greater than 1000

R Code:

```
comments <- read.table('comments.txt', header = FALSE)
names(comments) <- c("Post Type", "No. of Comments")
library(dplyr)
fltr <- filter(comments, comments$'No. of Comments' < 1000)
boxplot <- ggplot(fltr, aes(x = `Post Type`, y = `No. of Comments`)) + geom_boxplot()
boxplot
```

Boxplot Output:

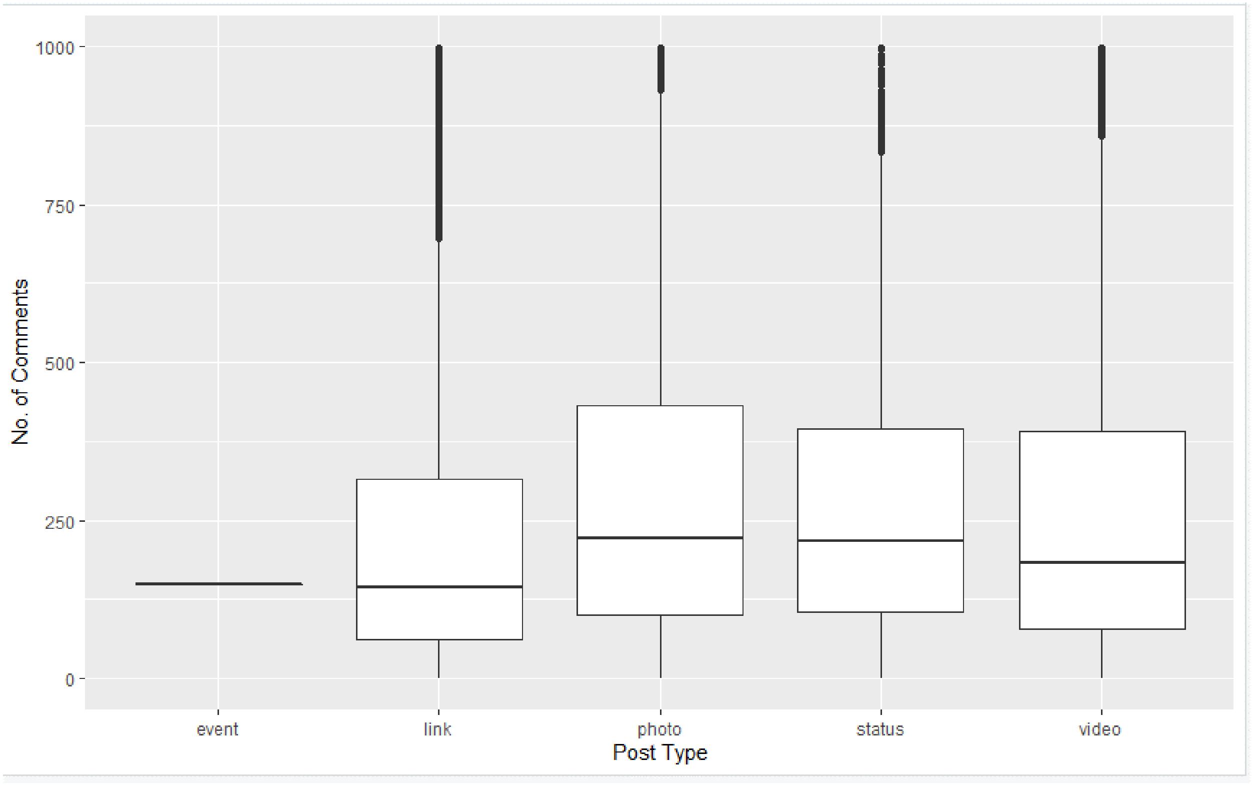


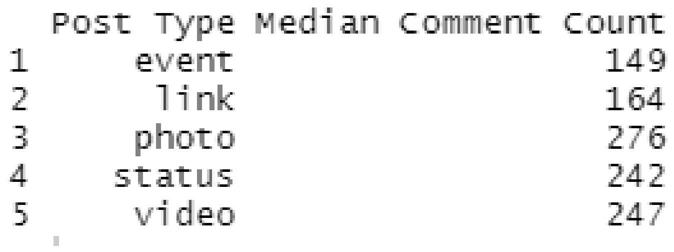
Figure 12.1: Boxplot of the number of comments for each post type

iii)

R Code:

```
comments <- read.table('comments.txt', header = FALSE)

names(comments) <- c("Post Type", "No. of Comments")

most_effective <- aggregate(comments$`No. of Comments`, list(comments$`Post Type`),
FUN=median)

names(most_effective) <- c("Post Type", "Median Comment Count")

most_effective
```

Output:



```
  Post Type Median Comment Count
1     event                  149
2      link                  164
3     photo                  276
4    status                  242
5     video                  247
```

Image 12.3: Table comparing the median value between Post Types

Conclusion:

Using this visualization, we can conclude that the type of post that has been most effective for abc-news on average is photos. This can be seen as the median comment count for photos is the highest as compared to the other post types.