FIT3152 Assignment 1 Report

Student Full Name: Zhiyue Li
Student Email: zlii0010@student.monash.edu
Student ID: 28280016

Introduction

In social science, an online community fills the roles of activity, language use, and social interactions. Metadata and linguistic summary are investigated in the real online forum. Linguistic Inquiry and Word Count (LIWC) was used for the linguistic analysis, which assessed the prevalence of specific thoughts, feelings, and motivations by calculating the proportion of key words used in communication [1]. Furthermore, 20,000 posts are used in the analysis, which are chosen at random from the original dataset, and it will then become the main focus of the analysis.

There are three major areas to be examined. To begin, the forum's activity and language are tracked over time. Second, the language used is examined by various groups. Finally, online social networks are examined.

Preliminary Analyse

Data cleaning is an important step before conducting data analysis. It's odd to notice that there are -1 for authorID in the table. As a result, such rows in the sample data must be removed. Aside from that, there is no more dirty data inside after the sample data has been examined.

Each post has 14 LIWC attributes that must be evaluated. Identifying key characteristics could influence how one author communicates with the others and save computation over analysis. Also, correlation can be used to determine which attributes have a strong relationship with others in order to determine the most important attributes.

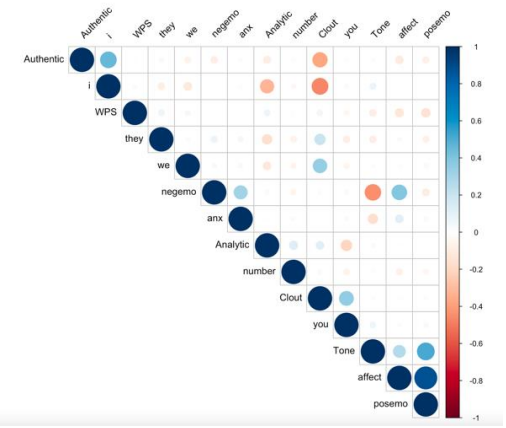


Figure 1: Correlation map

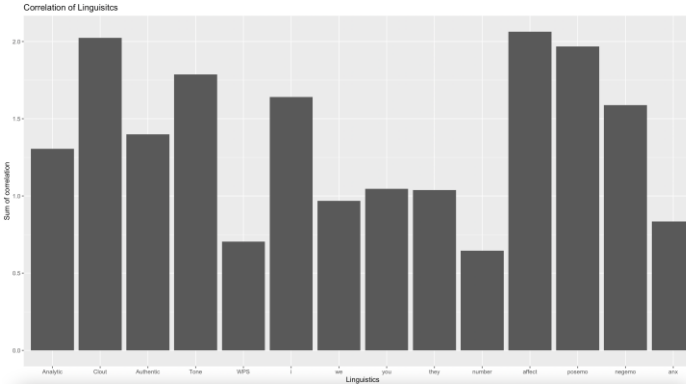
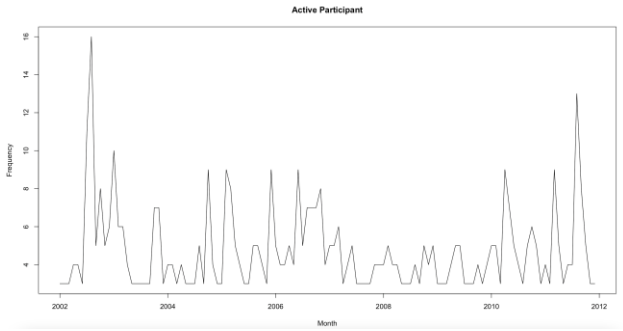


Figure 2: Correlation Bar

Figure 1 depicts the correlation coefficients between all linguistic attributes. The stronger the relationship between two linguistic attributes, the darker blue the circle in the map. The colour on the right bar represents the how close relationship between the two attributes. The graph in Figure 2 depicts the sum of the correlation coefficients for one linguistic attribute with others. The higher the bar for each attribute, the more important the attribute is in comparison to all other linguistic attributes. Clout, posemo, negemo, tone, and affect are the top five linguistic attributes according to both figures. As a result, the analysis will primarily focused on these five attributes.

Activity and Language analysis over time

The best method for analysing the characteristics of time and attributes is to use a time series plot. To determine how active the participants are, data will be grouped by date and post occurrences will be plotted. The first figure used to represent information about a dataset is always a summary statistic [2]. If the number of posts on that day is less than the first quantile, this row of data will be omitted. Consequently, only posts with a frequency greater than 2 are considered for analysis.



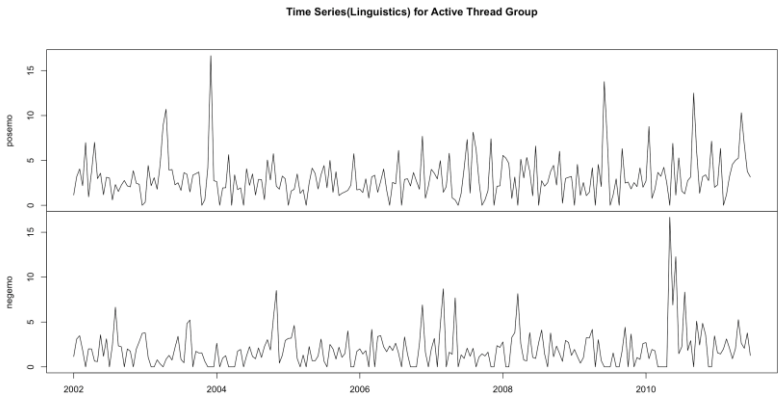
Date		Frequency	
2002-01-01:	1	Min.	: 1.000
2002-01-02:	1	1st Qu.:	2.000
2002-01-04:	1	Median :	5.000
2002-01-06:	1	Mean	: 6.024
2002-01-07:	1	3rd Qu.:	8.000
2002-01-08:	1	Max.	:68.000
(Other)	:3186		

Table 1: Summary for Frequency Post on Date

Figure 3: Active Participant over time

The Augmented Dickey Fuller Test (adf.test) is used to determine whether the plot is stationary. Because the p value is less than 0.05, it proves that the signal is stationary. In other words, when every two years are used to compare, the mean and variance are constant and independent of time. Furthermore, on average, for every two years, the number of posts on each month will increase at the start of each two year, then reach a peak, and then decrease. When it comes to the end of each two year, it might increase again.

Looking at the linguistic variables, the five main attributes mentioned above are chosen. Positive emotions, negative emotions, and affect are all useful when analysed together.



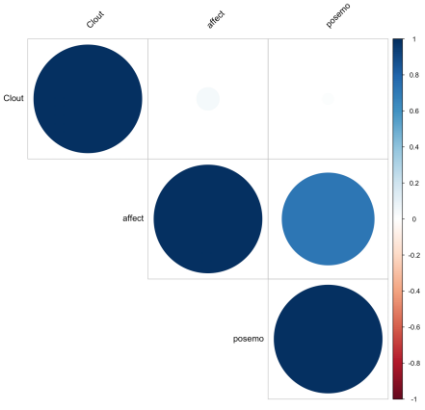


Figure 4: Posemo and Negemo in time series

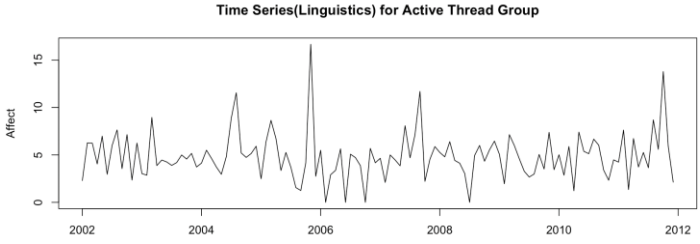


Figure 5: Affect in time series.

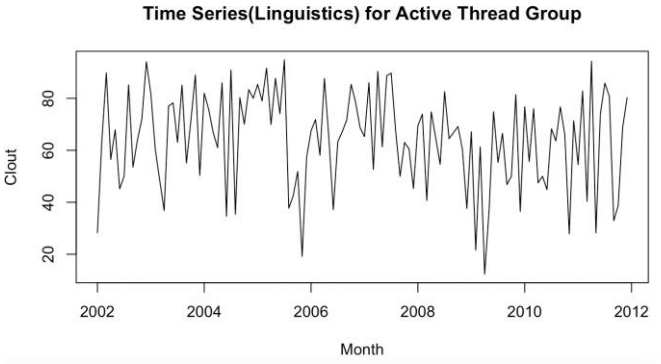


Figure 6: Clout in time series

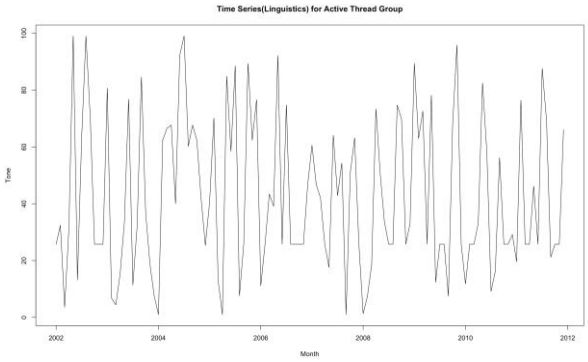


Figure 7: Tone in time series

Figure 1 clearly shows a strong correlation between positive emotions and affect, as well as negative emotions and affect. When comparing positive and negative emotions from Figure 4, both signals have an inverse trend. For example, if the positive emotions increase on a specific date, the negative emotions will decrease. There is yearly seasonality in Clout and Tone.

The Augmented Dickey Fuller Test is used to detect obvious tread among these five linguistic variables over time. The null hypothesis is not rejected if the p-value is greater than 0.05. The null hypothesis asserts that the data are not stationary. Only the negative emotions attribute (null hypothesis) in table 1 is stationary, whereas the other four attributes are more alternative hypothesis (based on the R adf.test function suggestion). To see more details on the trend of other 4 linguistic variables, mean and variance are applied for every year.

	Clout	Posemo	Analytic	Affect
2002-2003	64.25	2.55	66.75	4.97
2004- 2005	60.69	2.53	65.01	4.78
2006-2007	58.97	2.44	65.42	4.77
2008-2009	60.38	2.52	63.25	4.70
2010 - 2012	63	2.37	65.94	4.76

Table 2: Mean Value for each Linguistic Variable (2002 – 2012)

From table 2, it is interesting to notice that mean value for Clout has decreased from 64.25 to 58.97 and then increase from 580.97 to 63 ranging from 2006 to 2012. In terms of Posemo, it also has decreased from 2.55 to 2.44 then rise from 2.52 to 2.37. Thirdly, in terms of analytic, it has keeping decreasing from 66.75 ranging from 2002 to 2003 to 63.25 in 2008 to 2009. Finally, speaking of Affect, it also performs the similar tread as Clouts, decreasing from 4.97 to 4.70. And then it has risen to 4.76. In summary, Clout, Posemo, Affect has similar trend where they will decrease in the first 6 years and then it will increase again in the next four years.

In conclusion, linguistics variables like Tone, Analytic, Affect, Posemo have similar trends decrease in the first few years and increase latter. In addition, participants are always quite active over time due to constant mean and variance.

Language analysis via different Group

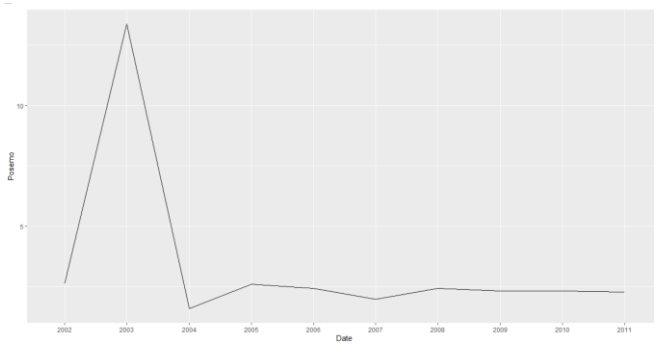
To see linguistic variable variance within or between threads, sample data are divided into groups based on threadID, authorID, or both. Furthermore, after categorising data by threadID, authorID, or both, if there are only a few occurrences of each threadID or one author, these rows of data will be omitted.

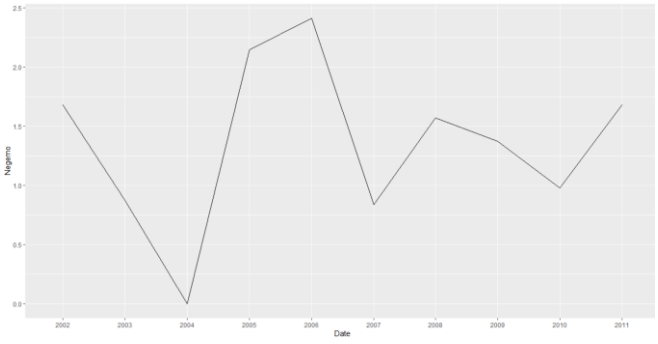
Firstly, when analysing data that has been classified by threadID, a scatterplot is a good way to see how the values of linguistic variables are distributed across the median value when the x axes are thread ID. The scatterplots are included in the Appendix. Furthermore, information about how different authors express their emotions and share their analytical thinking can be revealed on the same thread.

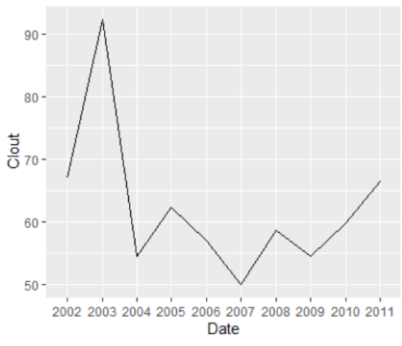
If you look at two scatterplots (posemo and negemo), one of them will be the domain for each threadID. Furthermore, the majority of posemo values are below the median threshold, while the majority of negemo values are above the corresponding threshold value, indicating that the majority of threads have a negative emotion pattern. When looking at the tone scatterplot, most points are lower than the corresponding median value, and 25 is the most frequent value occurring in the plot.

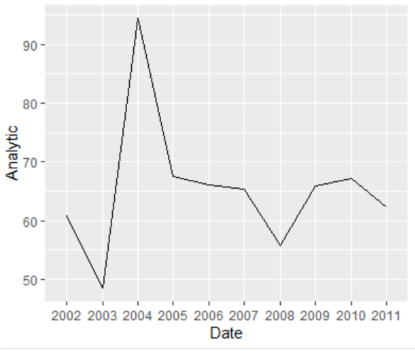
Secondly, when analysing data categorised by authorID, a scatterplot is a useful tool for observing how each author expressed their emotions and tone. The scatterplots for tone, anxiety, and analytics are included in the Appendix. When looking at a scatterplot for anxiety, most people have anxiety values that are lower than the median sample value. In terms of tone in authorID, 25 is still the most common value occurring in the diagram. Most analytical values are greater than the median sample value.

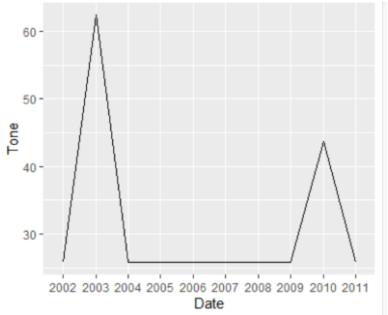
Thirdly, language used between threads are analysed over time. Another 100 threads are randomly selected among sample data. It is still good to view characteristics of languages used between threads change over time by creating time series plot.











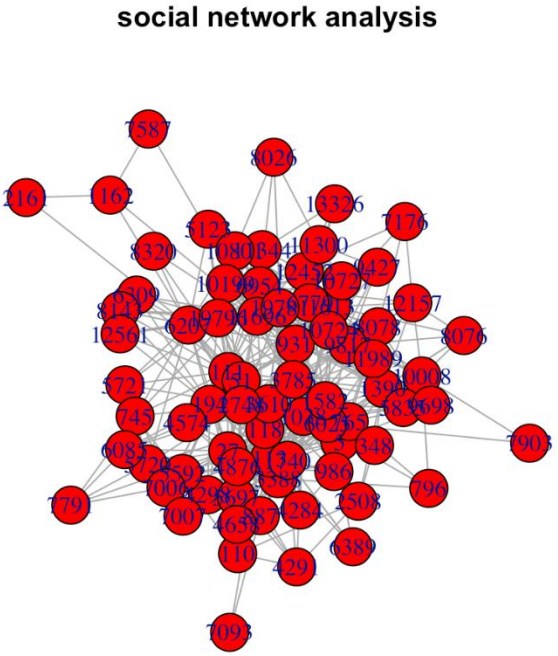
The five plots above illustrate top 5 main linguistic variables changes for different threads over time. Posemo and negemo have quite opposite emotions as one is increasing and the other is decreasing over the time. While the Clout and Tone have similar trend, both start increasing and decreasing from 2002 to 2004. In the end, they remain stable for the rest of period. While speaking of Analytic, it starts decrease and increase dramatically since 2003. When it comes to 2004, it starts to decrease until 2008.

When it comes to analyze the language used change within threads, only 5 threads are picked due to their highest occurrence compared with other thread (472752,252620,1271115). All these thread plots are attached in the Appendix 1.C. If check the stationarity on each time series plot by calculating the mean and variance, then both of attributes are constant. In conclusion, for each specific thread, there are not too much difference for language used as the same topic.

In summary, for each thread, there is slight change on the language used within threads because of constant mean and variance for each period. For different threads, the difference in the language used exists in different groups. In addition, either positive emotion or negative emotion will be the choice for the thread. Most positive emotions lie below the mean value while the negative emotions stay higher than mean value. The Analytic and Affect are distributed densely around the mean value. The Tone value has 25 as the most frequent occurrence. Between different threads, there have been an increasing trend initially and then it will drop on average.

Social Networks Online

Participants posting to the same thread at similar times are considered a social network in social network analysis. Furthermore, when the author begins to post to another thread during this time, the network expands as well. Participants who only posted once in such a period will be filtered out in the middle of the analysis. Furthermore, due to the greatest number of posts, only the period from 2002 to 2003 is chosen. The network diagram is attached below.



Several network characteristics are listed to be analysed. Firstly, diameter and average path length are direct ways to determine how the network is robust or connected. Among the sample data from 2002 to 2003, the average path length is 2.05 and diameter is 4. The short average path indicates that the concept of a small world where everyone is connected to everyone else through a short path. The shortest distance between two authors is 4.

To figure out the importance of a vertex, there are two factors: Degree and Centrality of vertex within the specific network. Degree means the total number of connections linked to one vertex or popularity of each thread or author. To check on the centrality of vertex, three parameters are needed to compare: betweenness, closeness and eigenvector. The rest three images are separately the top 5 rows for highest betweenness, closeness and eigenvector for the author.

	degree	betwness	closeness	eig
1038	36	309	0.0087	1.00
118	34	270	0.0086	1.00
11696	30	255	0.0082	0.73
6207	26	227	0.0079	0.66
931	30	214	0.0083	0.81
3785	29	142	0.0080	0.86

Table 3: Top 5 rows for Degree

	degree	betwness	closeness	eig
1038	36	309	0.0087	1.00
118	34	270	0.0086	1.00
931	30	214	0.0083	0.81
11696	30	255	0.0082	0.73
3785	29	142	0.0080	0.86
27	27	113	0.0079	0.84

Table 4: Top 5 rows for betweenness

	degree	betwness	closeness	eig
118	34	270	0.0086	1.00
1038	36	309	0.0087	1.00
3785	29	142	0.0080	0.86
27	27	113	0.0079	0.84
931	30	214	0.0083	0.81
1740	25	87	0.0079	0.78

Table 5: Top 5 rows for closeness

From the table above, it is clear to tell that authorIDs with 1038, 118 and 931 have the highest degree, betweenness, closeness and eigenvector generally. It proves that these 3 authors have posted to a few threads and they are popular. Moreover, these 3 authorID are more likely to become the main “bridge” between threadID and authorID in this network because of high betweenness. Thirdly, high closeness and eigenvector centrality indicates these 3 authors are more likely to influence all thread’s linguistics variable which they post. In conclusion, authorID 1038, 118,931 are most active author and quite popular in the network and they have much more influence on each thread in terms of linguistics attributes values.

In conclusion, authorID with 1038,118 and 931 are the most important nodes in the network as they have more influence on thread. In addition, every node is connected closely due to low diameter and appropriate average path length, which indicates a robust network.

Conclusion

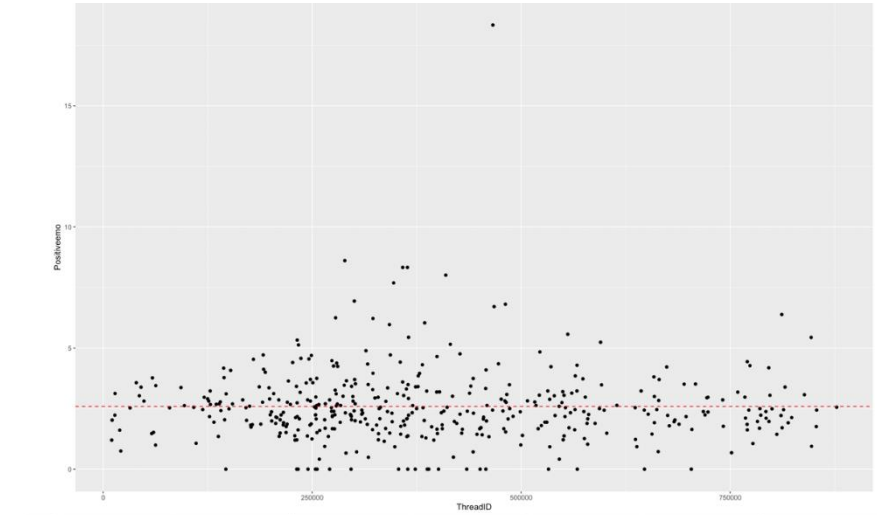
In summary, social network is heavily connected and robust. AuthorID with 1038, 118 and 931 have more influence as they have higher degree and betweenness. In addition, for each thread, there is not too much language used variance. Over the period, most linguistics variables perform decrease trend first and increase later due to varying mean and variance.

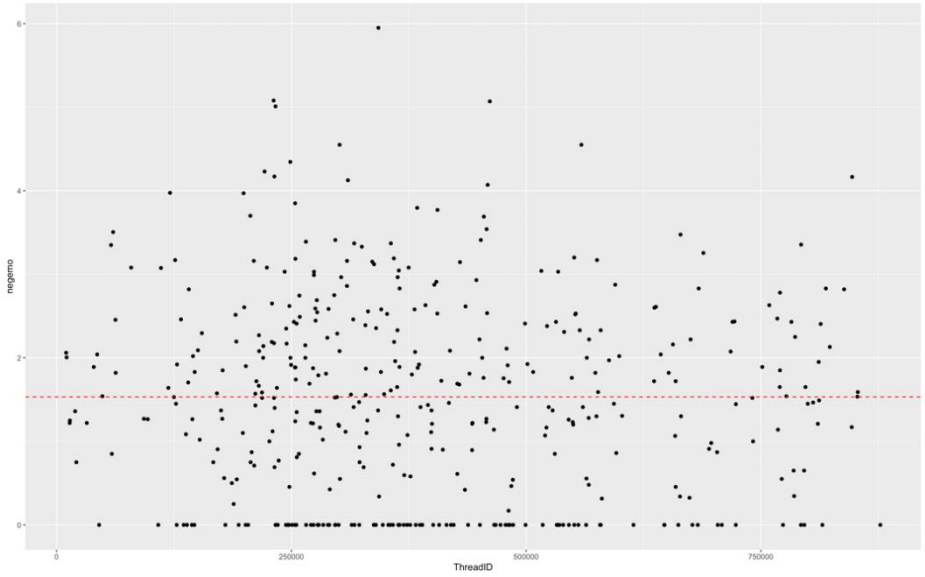
Reference

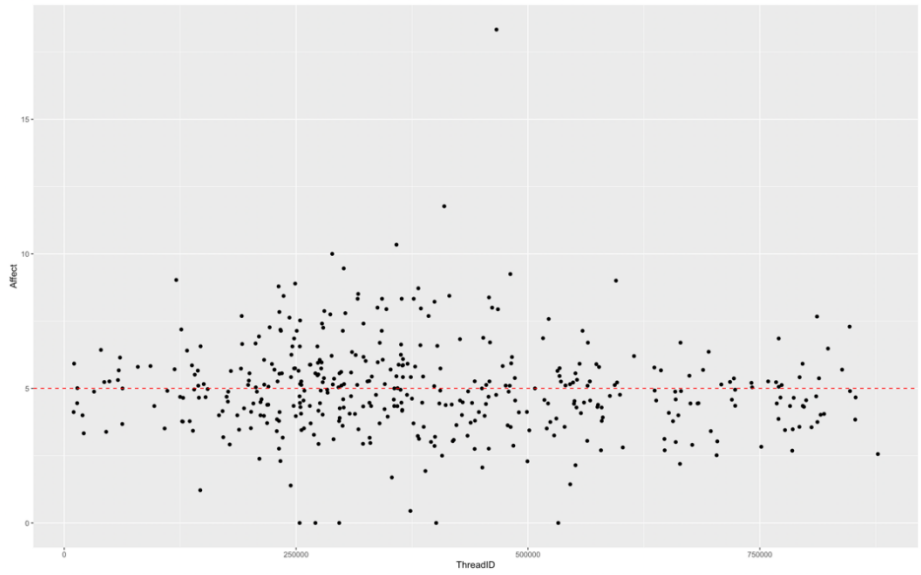
1. James W. Pennebaker, Ryan L. Boyd, Kayla Jordan, and Kate Blackburn, "Development and Psychometric Properties of 2015", The University of Texas at Austin, 2015.
2. Applied Time Series, "Dickey-Fuller and Augmented Dickey-Fuller tests", https://nwfsc-timeseries.github.io/atsa-labs/sec-boxjenkins-aug-dickey-fuller.html (Assessed: *2021-03-01)*

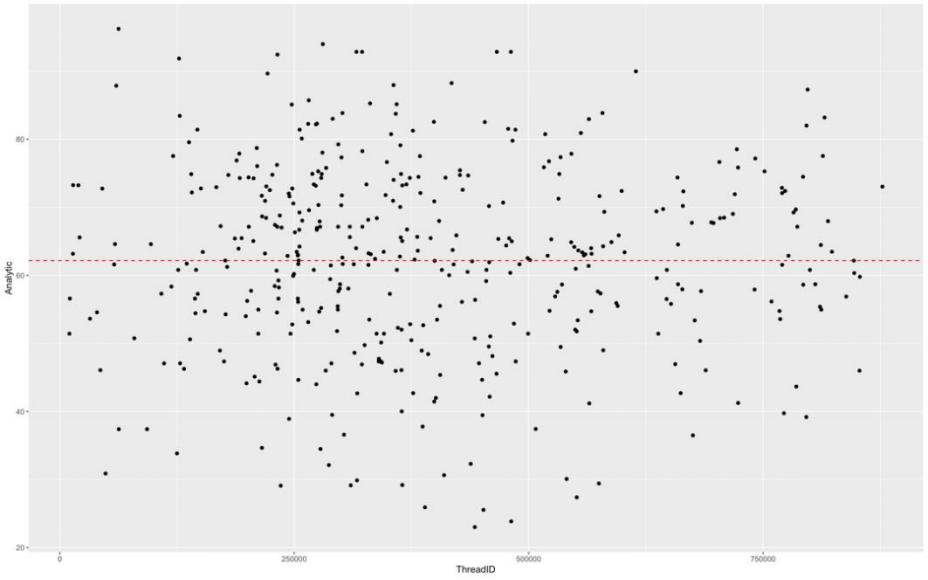
Appendix

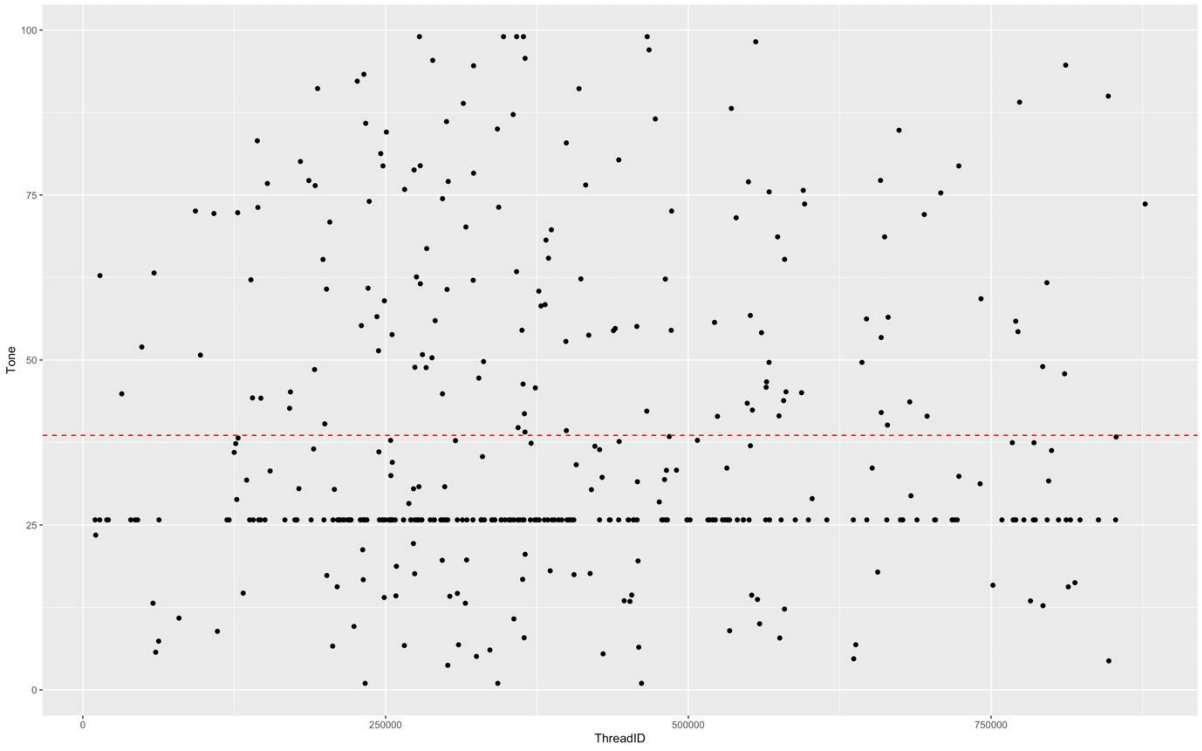
1. Language analysis scatterplot in different threadID



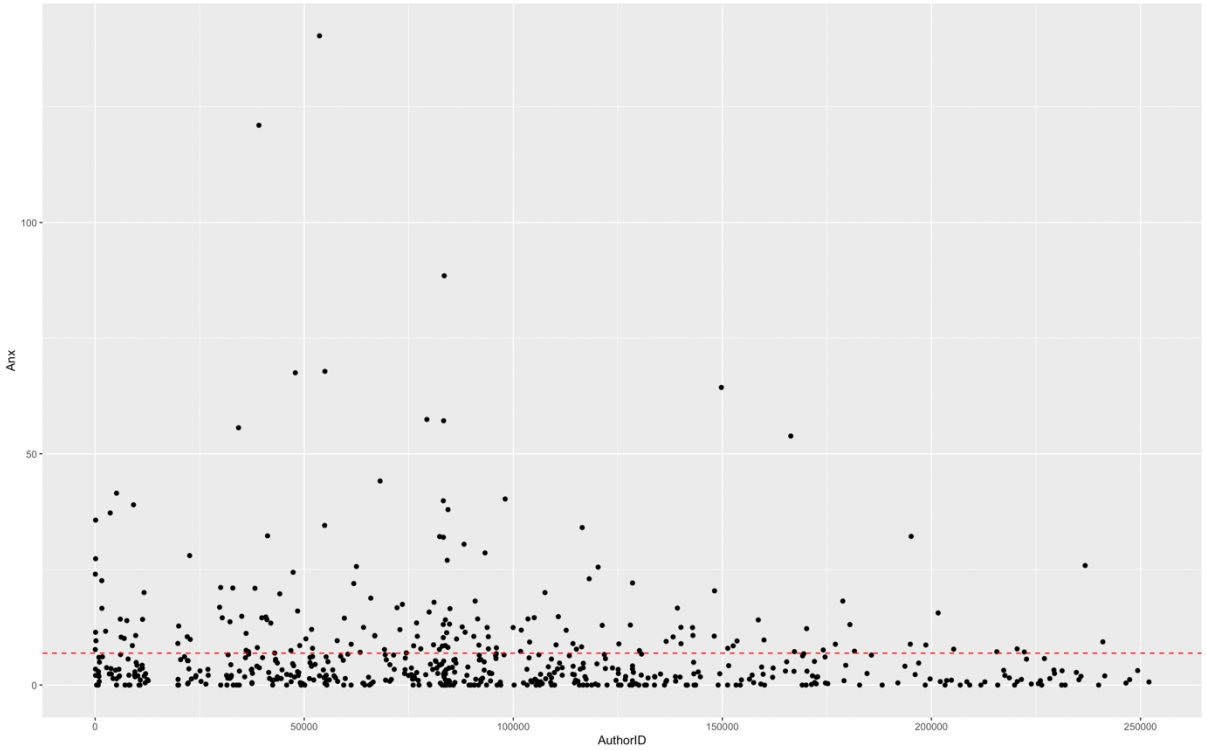


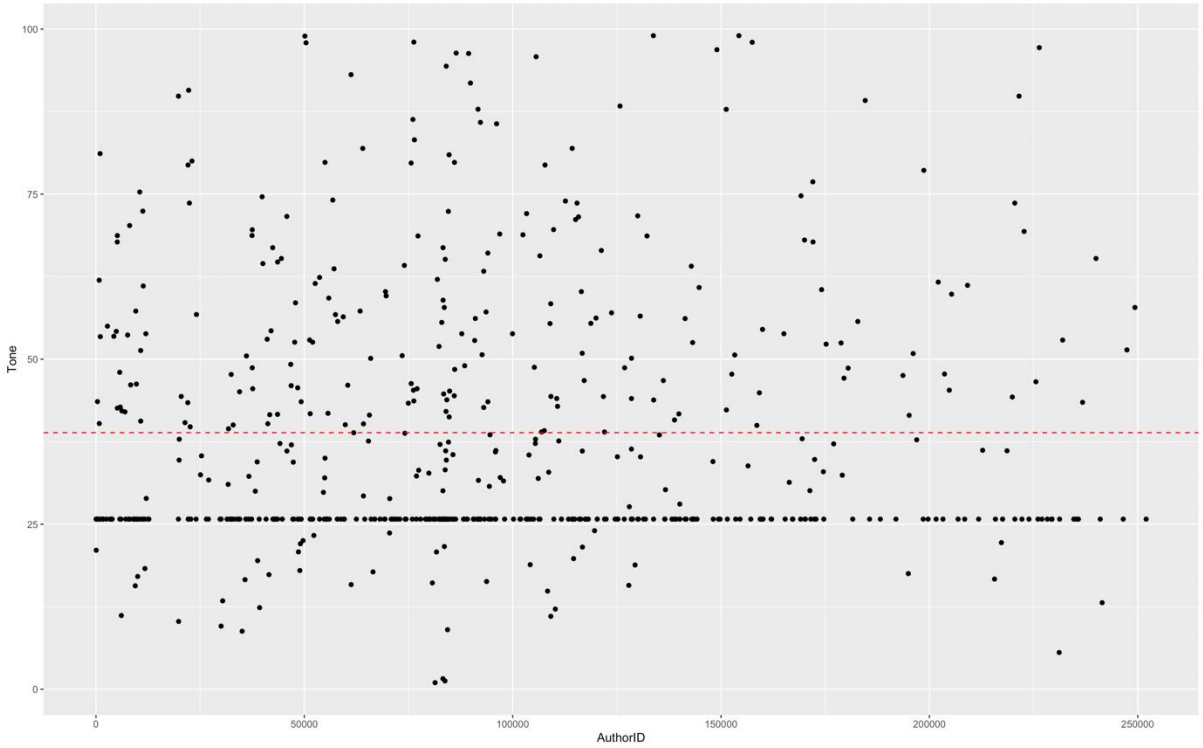


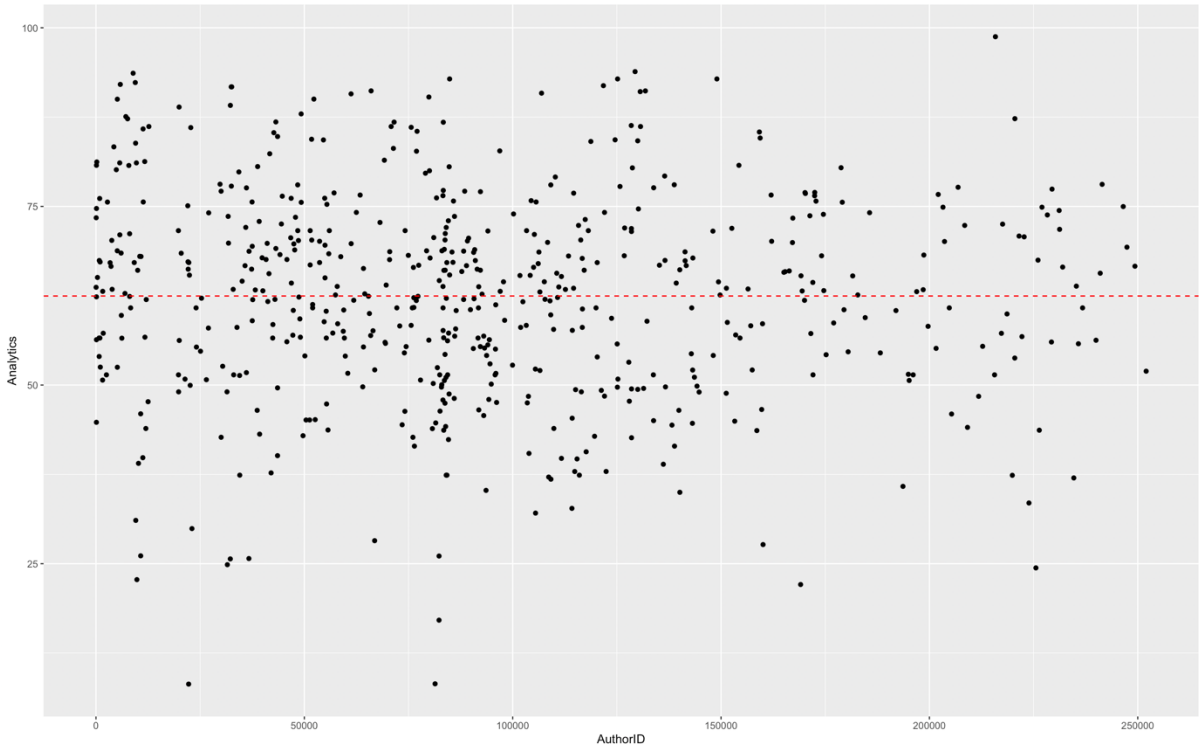

Analytic
80
60
40
20
0
250000
500000
750000
ThreadID


Tone
100
75
50
25
0
0
250000
500000
750000
ThreadID

2. Language analysis scatterplot in different authorID

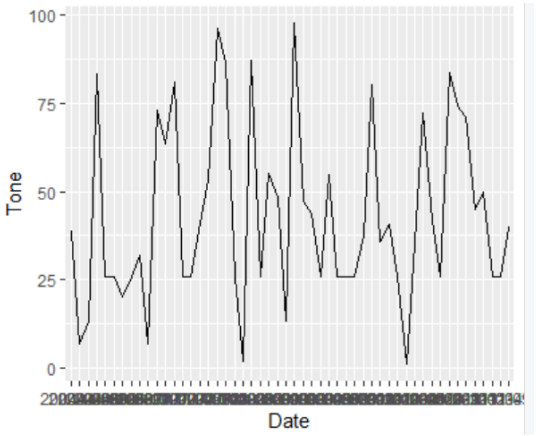


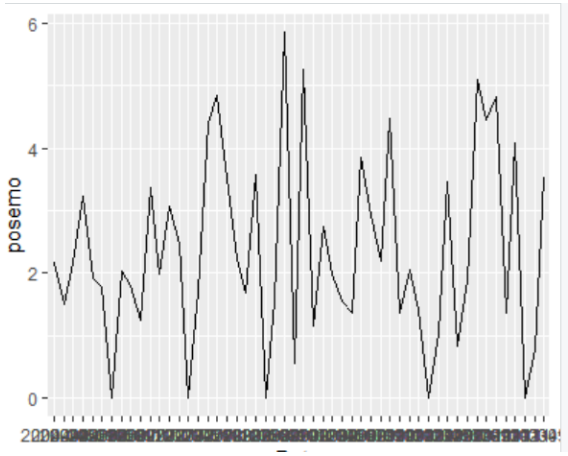

Tone
100
75
50
25
0
0
50000
100000
150000
200000
250000
AuthorID

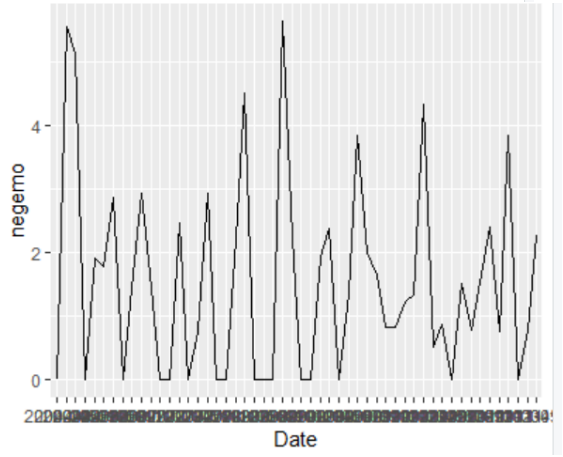

Analytics
100
75
50
25
0
50000
100000
150000
200000
250000
AuthorID

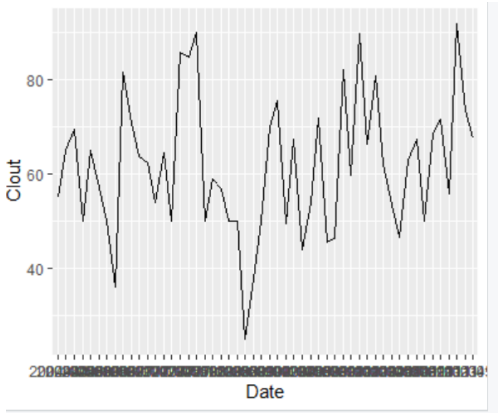
Time Series Plot within Threads.

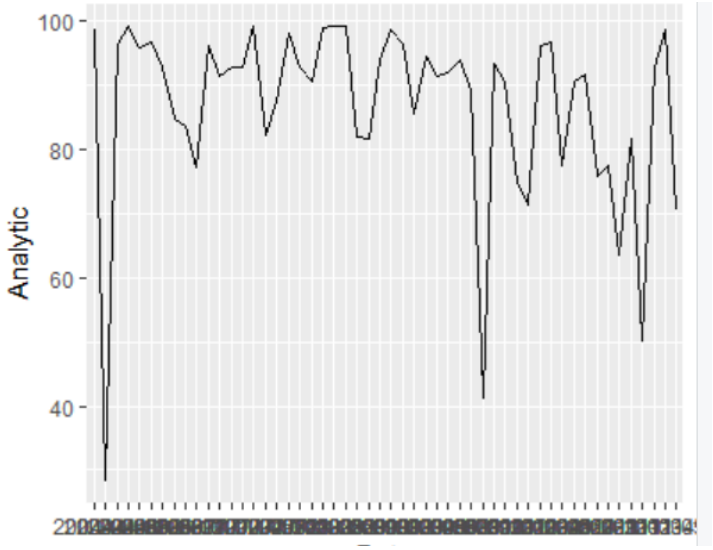
ThreadID = 127115



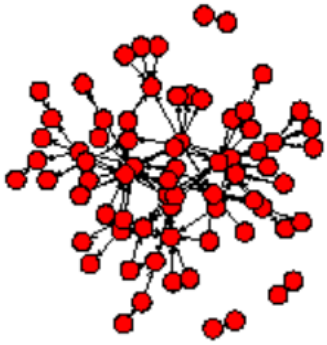








Network Group Diagram



R code

```
## FIT3152 Assignment 1
## Written by: Zhiyue Li
## Last Modified date: 23th April 2021

# Reference used to finish the assignment
# https://stackoverflow.com/questions/53561299/rmin-and-max-of-a-date-column-in-a-dataframe/53561477
# https://www.datanovia.com/en/lessons/identify-and-remove-duplicate-data-in-r/
# https://www.kaggle.com/jmmvutu/eda-online-c2c-fashion-store-user-behaviour
# https://stackoverflow.com/questions/12976542/how-to-convert-in-both-directions-between-year-month-day-and-dates-in-r

## clear the environment
rm(list=ls())
set.seed(28280016)    # my student ID = 28280016
# Random seed to make subset reproducible

# Clear the console output
cat("\014")

## Install and load some necessary libraries if not dowloaded
#install.packages("ggplot2")
#install.packages("dplyr")
#install.packages("reshape2")
#install.packages("ggcorrplot")
#install.packages("lubridate")
#install.packages(c("igraph","igraphdata))
#install.packages("sand")
#install.packages("date")
#install.packages("mapplots")
#install.packages("ggraph")

library(igraph)
library(igraphdata)
```

```r
library(ggplot2)
library(dplyr)
library(plyr)
library(reshape2)
library(stringr)
library(ggcorrplot)
library(lubridate)
library(sand)
library(date)
library(mapplots)
library(network)
library(tidyverse)
library(ggraph)
#library("TTR")



# Step 1: Import the data
#setwd("~/desktop/FIT3152/Assignment 1")
webforum <- read.csv("webforum.csv",stringsAsFactors =FALSE)
webforum <- webforum[sample(nrow(webforum),20000), ] # 20000 rows
attach(webforum)
head(webforum)

## Preliminary Analysis
str(webforum)     # Overview of webforum

summary(webforum)
min(webforum$WC)
which(is.na(webforum$Date))    # No missing date value in Date column


# Notice there are few -1 for AuthorID,find out how many of them among
dataset
AuthorID_1 <- with(webforum,length(AuthorID[AuthorID == -1]))
AuthorID_1

# Notice if there are some zero WC length
WC_zero <- with(webforum,length((WC[WC == 0])))


# Notice if there are duplicate rows
duplicateRows <- webforum[duplicated(webforum),]    # yes, there are
duplicate rows

# Notice the domain of Date
range(webforum$Date, na.rm = TRUE)

# Extract linguistic variables
linguistic_var <- webforum[ ,6:19]

# Plot correlation between linguistics in the map
# http://www.sthda.com/english/wiki/correlation-matrix-a-quick-start-guide-
to-analyze-format-and-visualize-a-correlation-matrix-using-r-software#use-
heatmap
# Use corrplot() function: Draw a correlogram
#install.packages("corrplot")
library("corrplot")
res <- cor(linguistic_var)
#install.packages("Hmisc")
library("Hmisc")
```

```r
res2 <- rcorr(as.matrix(linguistic_var))
res2
# Extract the correlation coefficients
res2$r
# Extract p-values
res2$P
corrplot(res, type = "upper", order = "hclust",
         tl.col = "black", tl.srt = 45)

# Plot correlation between linguistics in the bar
linguistic_cor <- as.data.frame(as.table(cor(linguistic_var)))
linguistic_cor <- linguistic_cor[!(linguistic_cor$Var1 ==
linguistic_cor$Var2),]
linguistic_cor$Freq <- abs(linguistic_cor$Freq)
cdf <- as.data.frame(as.table(by(linguistic_cor, linguistic_cor$Var2,
function(df) sum(df$Freq))))
cdf <- cdf[order(-cdf$Freq),]
plot(cdf[1:4,])
xaxes <- cdf$linguistic_cor.Var2
yaxes <- cdf$Freq
graph <- ggplot(data = cdf,aes(x = xaxes,y = yaxes))
graph <- graph + geom_col() +  labs(title = "Correlation of Linguisitcs", x
= "Linguistics",y="Sum of correlation")
plot(graph)

# Based on this plot, top 5 linguistic plot,Affect, Positive emotions,
Negative emotions, Clout and Tone are chosen for analysis
## Step 2: Tidying (Cleaning)
# Assign new data set
web <- webforum
attach(web)

# Remove the row with -1 AuthorID
web <- web[web$AuthorID != -1,]

# Remove the WC with 0 count
web <- web[web$WC != 0,]

# Filter out data based on date
web <- web %>%
  filter(Date >= as.Date("2002-01-01") & Date <= as.Date("2011-12-31"))

## Step 3: Transform, Visualize and Model ( Data Analysis )

# Part a: Analyze activity and language on the forum over time.
# Participants active or not.
# Thread divided into two different parts: active or inactive
thread_activity <- ddply(web,.(ThreadID), function(df)
length(unique(df$Date)))
#thread_activity
colnames(thread_activity)[2] <- "Thread_Frequency"
summary(thread_activity)
plot(thread_activity)
threshold_active <- ceiling(mean(thread_activity$`Thread_Frequency`))

## Find out the active participants flow diagram
melbourne <- as.data.frame(table(web$Date))
colnames(melbourne)[1] <- "Date"
colnames(melbourne)[2] <- "Frequency"
active_threshold <- median(melbourne$Frequency)
summary(melbourne)
```

```
active_melbourne <- melbourne[melbourne$Frequency>2,]
p <- ggplot(active_melbourne, aes(x=Date, y=Frequency,group = 1)) +
            geom_line(aes(y = Frequency))

active_participant <- ts(active_melbourne$Frequency,frequency = 12, start =
c(2002,1), end = c(2011,12))
ts.plot(active_participant, main = "Active Participant",ylab =
"Frequency",xlab = "Month")
summary(active_participant)

adf.test(active_participant) # p-value < 0.05 indicates the TS is
stationary

# Calculate  the mean of 5 linguistic variables of active thread group
ordered by month
ts_month_active_threads <- web[web$ThreadID > threshold_active,] %>%
                                dplyr::group_by(Date) %>%
                                dplyr::summarise(
                                  affect = median(affect),
                                  Clout = median(Clout),
                                  Tone = median(Tone),
                                  posemo = median(posemo),
                                  negemo = median(negemo),
                                  Analytic = median(Analytic)
                                )


dta.sum <- aggregate(x = ts_month_active_threads[c("Date")],
                     FUN = sum,
                     by = list(Group.date = ts_month_active_threads$Date))

fit <- lm(Clout~Tone + affect,data=ts_month_active_threads)
summary(fit)

fit1 <- lm(affect~posemo+negemo,data=ts_month_active_threads)
summary(fit1)

active_linguistic_posemo <- ts(ts_month_active_threads$posemo,frequency =
12, start = c(2002,1), end = c(2011,12))
ts.plot(active_linguistic, main = "Time Series(Linguistics) for Active
Thread Group",ylab = "Positive emotions",xlab = "Month")

active_linguistic_negemo <- ts(ts_month_active_threads$negemo,frequency =
12, start = c(2002,1), end = c(2011,12))
ts.plot(active_linguistic_negemo, main = "Time Series(Linguistics) for
Active Thread Group",ylab = "Negative emotions",xlab = "Month")

active_linguistic_affect <- ts(ts_month_active_threads$affect,frequency =
12, start = c(2002,1), end = c(2011,12))
ts.plot(active_linguistic_affect, main = "Time Series(Linguistics) for
Active Thread Group",ylab = "Affect",xlab = "Month")

active_linguistic_analytic <- ts(ts_month_active_threads$Analytic,frequency
= 12, start = c(2002,1), end = c(2011,12))
ts.plot(active_linguistic_analytic, main = "Time Series(Linguistics) for
Active Thread Group",ylab = "Analytic",xlab = "Month")

active_linguistic_clout <- ts(ts_month_active_threads$Clout,frequency = 12,
start = c(2002,1), end = c(2011,12))
ts.plot(active_linguistic_clout, main = "Time Series(Linguistics) for
Active Thread Group",ylab = "Clout",xlab = "Month")
```

```r
active_linguistic_tone <- ts(ts_month_active_threads$Tone,frequency = 12,
start = c(2002,1), end = c(2011,12))
ts.plot(active_linguistic_tone, main = "Time Series(Linguistics) for Active
Thread Group",ylab = "Tone",xlab = "Month")

acf(active_linguistic_tone, lag.max = 36)


ts_month_active_threads$Date <- substring(ts_month_active_threads$Date,1,4)

ts_posemo_mean <- aggregate(ts_month_active_threads$posemo,by
=list(Category = ts_month_active_threads$Date),FUN = median)
ts_clout_mean <- aggregate(ts_month_active_threads$Clout,by =list(Category
= ts_month_active_threads$Date),FUN = median)
ts_analytic_mean <- aggregate(ts_month_active_threads$Analytic,by
=list(Category = ts_month_active_threads$Date),FUN = median)
ts_tone_mean <- aggregate(ts_month_active_threads$Tone,by =list(Category =
ts_month_active_threads$Date),FUN = median)
ts_affect_mean <- aggregate(ts_month_active_threads$affect,by
=list(Category = ts_month_active_threads$Date),FUN = median)


#install.packages("tseries")
library(tseries)
adf.test(active_linguistic_posemo) # p-value < 0.05 indicates the TS is
stationary
adf.test(active_linguistic_negemo) # p-value < 0.05 indicates the TS is
stationary
adf.test(active_linguistic_clout) # p-value < 0.05 indicates the TS is
stationary
adf.test(active_linguistic_analytic) # p-value < 0.05 indicates the TS is
stationary
adf.test(active_linguistic_affect) # p-value < 0.05 indicates the TS is
stationary
adf.test(active_linguistic_tone) # p-value < 0.05 indicates the TS is
stationary


Box.test(active_linguistic_posemo, lag=15, type="Ljung-Box") # test
stationary signal


# Part b: Analyze the language used by groups
# New data used for analysis
# Group data by thread
web_task2<- data.frame()
web_task2 <- by(web,web$ThreadID,function(df) df[order(df$Date),])
head(web_task2)

newdata <- data.frame()
# Find ThreadID index for threadID
for (threadID in unique(web$ThreadID))
{
  temp <- web[web$ThreadID == threadID,]
  frequency <- nrow(temp)
  temp_clout <- median(temp$Clout)
  temp_affect <- median(temp$affect)
  temp_posemo <- median(temp$posemo)
  temp_negemo <- median(temp$negemo)
  temp_analytic <- median(temp$Analytic)
```

```
  temp_tone <- median(temp$Tone)

  if (frequency >9)
  {
    newdata <- rbind(newdata,
c(threadID,temp_clout,temp_affect,temp_posemo,temp_negemo,frequency,temp_an
alytic,temp_tone))
  }
}
colnames(newdata)[1] <-"ThreadID"
colnames(newdata)[2] <- "Clout"
colnames(newdata)[3] <- "affect"
colnames(newdata)[4] <- "posemo"
colnames(newdata)[5] <- "negemo"
colnames(newdata)[6] <- "Frequency"
colnames(newdata)[7] <- "Analytic"
colnames(newdata)[8] <- "Tone"



# Group by the ThreadID and analyse some linguistic variables( or all )
newdata1 <- aggregate(newdata$posemo,by =list(Category =
newdata$ThreadID),FUN = sum)
colnames(newdata1)[1] <- "ThreadID"
colnames(newdata1)[2] <- "Positiveemo"
threshold_postive_emo <- mean(newdata1$Positiveemo)
ggplot(data=newdata1,aes(x=ThreadID,y=Positiveemo)) + geom_point()
+geom_hline(yintercept =
threshold_postive_emo,linetype="dashed",color="red")

newdata2 <- aggregate(newdata$negemo,by =list(Category =
newdata$ThreadID),FUN = sum)
colnames(newdata2)[1] <- "ThreadID"
colnames(newdata2)[2] <- "negemo"
threshold_negemo<- mean(newdata2$negemo)
ggplot(data=newdata2,aes(x=ThreadID,y=negemo)) + geom_point()
+geom_hline(yintercept = threshold_negemo,linetype="dashed",color="red")

newdata3 <- aggregate(newdata$affect,by =list(Category =
newdata$ThreadID),FUN = sum)
colnames(newdata3)[1] <- "ThreadID"
colnames(newdata3)[2] <- "Affect"
threshold_affect <- mean(newdata3$Affect)
ggplot(data=newdata3,aes(x=ThreadID,y=Affect)) + geom_point()
+geom_hline(yintercept = threshold_affect,linetype="dashed",color="red")


newdata4 <- aggregate(newdata$Clout,by =list(Category =
newdata$ThreadID),FUN = sum)
colnames(newdata4)[1] <- "ThreadID"
colnames(newdata4)[2] <- "Clout"
threshold_clout <- mean(newdata4$Clout)
clout_plot <- ggplot(data=newdata4,aes(x=ThreadID,y=Clout))
plot(clout_plot)

newdata5 <- aggregate(newdata$Analytic,by =list(Category =
newdata$ThreadID),FUN = sum)
colnames(newdata5)[1] <- "ThreadID"
colnames(newdata5)[2] <- "Analytic"
threshold_analytic <- mean(newdata5$Analytic)
```

```
ggplot(data=newdata5,aes(x=ThreadID,y=Analytic)) +
  geom_point() +
  geom_hline(yintercept = threshold_analytic,linetype="dashed",color="red")

newdata6 <- aggregate(newdata$Tone,by =list(Category =
newdata$ThreadID),FUN = sum)
colnames(newdata6)[1] <- "ThreadID"
colnames(newdata6)[2] <- "Tone"
threshold_tone <- mean(newdata6$Tone)
ggplot(data=newdata6,aes(x=ThreadID,y=Tone)) +
  geom_point() +
  geom_hline(yintercept = threshold_tone,linetype="dashed",color="red")


## Language used within threads (or between threads ) change over time
newdata_top_20 <- newdata[sample(nrow(newdata), 100), ] # randomly chose 10
threadID to see language change over time


sample_data <- web[web$ThreadID %in% newdata_top_20$ThreadID,]
sample_data$Date <- substring(sample_data$Date,1,4)

sample_group_date <- aggregate(sample_data$Tone,by = list(Category =
sample_data$Date), FUN = median)
colnames(sample_group_date)[1] <- "Date"
colnames(sample_group_date)[2] <- "Tone"
head(sample_group_date)

ggplot(data = sample_group_date, aes(x =Date, y= Tone,group = 1)) +
geom_line()


## Analysise the Language used within threads
sample_with_thread <- web[web$ThreadID == 252620, ]

sample_with_thread$Date <- substring(sample_with_thread$Date,1,7)


sample_group_date <- aggregate(sample_with_thread$Tone,by = list(Category =
sample_with_thread$Date), FUN = median)
colnames(sample_group_date)[1] <- "Date"
colnames(sample_group_date)[2] <- "Tone"
ggplot(data = sample_group_date, aes(x =Date, y= Tone,group = 1)) +
geom_line()

sample_group_date <- aggregate(sample_with_thread$posemo,by = list(Category
= sample_with_thread$Date), FUN = median)
colnames(sample_group_date)[1] <- "Date"
colnames(sample_group_date)[2] <- "posemo"
ggplot(data = sample_group_date, aes(x =Date, y= posemo,group = 1)) +
geom_line()

sample_group_date <- aggregate(sample_with_thread$negemo,by = list(Category
= sample_with_thread$Date), FUN = median)
colnames(sample_group_date)[1] <- "Date"
colnames(sample_group_date)[2] <- "negemo"
ggplot(data = sample_group_date, aes(x =Date, y= negemo,group = 1)) +
geom_line()

sample_group_date <- aggregate(sample_with_thread$Clout,by = list(Category
= sample_with_thread$Date), FUN = median)
```

```
colnames(sample_group_date)[1] <- "Date"
colnames(sample_group_date)[2] <- "Clout"
ggplot(data = sample_group_date, aes(x =Date, y= Clout,group = 1)) +
geom_line()

sample_group_date <- aggregate(sample_with_thread$Analytic,by =
list(Category = sample_with_thread$Date), FUN = median)
colnames(sample_group_date)[1] <- "Date"
colnames(sample_group_date)[2] <- "Analytic"
ggplot(data = sample_group_date, aes(x =Date, y= Analytic,group = 1)) +
geom_line()



## Group by AuthorID and Analyse linguistic variables (Analytic,
Clout,anx,Tone, Authentic)
newdata_author <- data.frame()
for (authorID in unique(web$AuthorID))
{
  temp <- web[web$AuthorID == authorID,]
  frequency <- nrow(temp)
  temp_analytic <- median(temp$Analytic)
  temp_clout <- median(temp$Clout)
  temp_tone <- median(temp$Tone)
  temp_authentic <- max(temp$Authentic)
  temp_anx <- sum(temp$anx)

  if (frequency >5)
  {
    newdata_author <- rbind(newdata_author,
c(authorID,temp_analytic,temp_clout,temp_tone,temp_authentic,temp_anx,frequ
ency))
  }
}
colnames(newdata_author)[1] <-"AuthorID"
colnames(newdata_author)[2] <- "Analytics"
colnames(newdata_author)[3] <- "Clout"
colnames(newdata_author)[4] <- "Tone"
colnames(newdata_author)[5] <- "Authentic"
colnames(newdata_author)[6] <- "anx"
colnames(newdata_author)[7] <- "Frequency"
summary(newdata_author)

newdata_author1 <- aggregate(newdata_author$anx,by =list(Category =
newdata_author$AuthorID),FUN = sum)
colnames(newdata_author1)[1] <- "AuthorID"
colnames(newdata_author1)[2] <- "Anx"
threshold_anx <- mean(newdata_author1$Anx)
ggplot(data=newdata_author1,aes(x=AuthorID,y=Anx)) + geom_point()
+geom_hline(yintercept = threshold_anx,linetype="dashed",color="red")


newdata_author2 <- aggregate(newdata_author$Tone,by =list(Category =
newdata_author$AuthorID),FUN = sum)
colnames(newdata_author2)[1] <- "AuthorID"
colnames(newdata_author2)[2] <- "Tone"
threshold_tone <- mean(newdata_author2$Tone)
ggplot(data=newdata_author2,aes(x=AuthorID,y=Tone)) + geom_point()
+geom_hline(yintercept = threshold_tone,linetype="dashed",color="red")
```

```r
newdata_author3 <- aggregate(newdata_author$Analytics,by =list(Category =
newdata_author$AuthorID),FUN = sum)
colnames(newdata_author3)[1] <- "AuthorID"
colnames(newdata_author3)[2] <- "Analytics"
threshold_analytic <- mean(newdata_author3$Analytics)
ggplot(data=newdata_author3,aes(x=AuthorID,y=Analytics)) + geom_point()
+geom_hline(yintercept = threshold_analytic,linetype="dashed",color="red")



## Part c : Social network online (Network Analysis), trees are chosen
n_vertices <- length(unique(web$ThreadID))
n_dates <- length(unique(web$Date))

#https://www.jessesadler.com/post/network-analysis-with-r/

# Extract dates based on the range of date
temp1<-web %>%
  filter(Date >= as.Date("2002-01-01") & Date <= as.Date("2002-01-31"))

threadID_list <- unique(temp1$ThreadID)
authorID_list <- unique(temp1$AuthorID)
dots <- data.frame()
dots$ThreadID <- threadID_list
dots$AuthorID <- authorID_list

# Edge information
temp1<-aggregate(.~ThreadID+AuthorID, temp1, count)    # per route, count
the number of occurrence
temp2 <- temp1 %>% select(c(1,2,6))

temp2$Post <- temp2$Analytic[ ,2]
temp2$Weight <- lengths(temp2$Post)

# Delete columns
temp2$Analytic <- NULL
temp2$Post <- NULL
colnames(temp2)[1] <- "source"
colnames(temp2[2]) <- "destination"

threadID_list <- unique(temp2$ThreadID)
authorID_list <- unique(temp2$AuthorID)
nodes <- data.frame(c(threadID_list,authorID_list))  %>%
rowid_to_column("id")
colnames(nodes)[1] <- "id"
colnames(nodes)[2] <- "label"

edges <- temp2 %>%
  left_join(nodes, by = c("source" = "label")) %>%
  rename(from = id)

temp2$ThreadID <- as.character(temp2$ThreadID)
temp2$AuthorID <- as.character(temp2$AuthorID)
class(temp2$AuthorID)    # char
class(temp2$ThreadID)     # char

# network objects
thread_network = network(temp2,vertex.attr = nodes, matrix.type =
"edgelist")
thread_network
```

```r
betweeness(thread_network)
plot(thread_network,vertex.cex = 3)


web <- web %>%
  filter(Date >= as.Date("2002-01-01") & Date <= as.Date("2011-12-31"))


author_post_frequency = ddply(web, .(AuthorID), function(df)
length(unique(df$ThreadID)))
colnames(author_post_frequency)[2] = c("PostNum")
threshold <- ceiling(median(author_post_frequency$PostNum))
summary(author_post_frequency$PostNum)
author_post_frequency = with(author_post_frequency,
author_post_frequency[PostNum > 1,])
summary(author_post_frequency)
nrow(author_post_frequency)

author_post_frequency_authorID = author_post_frequency$AuthorID



## Extract information matching authorID post more than median
author_list_by_threads = web[web$AuthorID %in%
author_post_frequency_authorID,] %>%
                      dplyr::group_by(ThreadID) %>%
                      dplyr::summarise(
                        AuthorID = list(unique(AuthorID))
                      )
#install.packages("gtools")
library("gtools")

## my_rel_comb function and creating table functions was cited from the
kz_sher
## Here is the link:https://github.com/kz-
sher/fit3152_a1/blob/master/FIT3152_A1.R
### Create a function that takes a list of values and generates all the
combinations of connection among them
my_rel_comb = function(list){
  temp = length(list)
  if (temp >2)
  {
      return(combinations(n=temp, r=2, v=list))
  }
}

##### Apply combination function to each list of authors and combine them
by rows
connection_table = do.call(rbind, lapply(author_list_by_threads$AuthorID,
my_rel_comb))
connection_table = unique(connection_table) # Remove those duplicate
connections between two authors

#### Convert table (data frame) into graph
g1 <- graph_from_data_frame(connection_table,directed=F)
plot(g1,vertex.color = "red",main = "social network analysis")

#degree <- as.table(degree(g1))
#betwness <- as.table(betweenness(g1))
#closeness <- as.table(closeness(g1))
#eig <- as.table(evcent(g1)$vector)
```

```r
degree <- centralization.degree(g1)$res
betwness <- centralization.betweenness(g1)$res
closeness <- centralization.closeness(g1)$res
eig <- centralization.evcent(g1)$res


tabularised <- as.data.frame(rbind(degree,betwness,closeness,eig))
tabularised <- t(tabularised)

#install.packages("sna")
# Density
library("sna")
degree(connection_table)

averagePath <- average.path.length(g1)
diameter <- diameter(g1)
cat("\nAverage Path Length: ", averagePath, "\n\n")
cat("\nDiameter: ", diameter, "\n\n")

print(tabularised,digits = 3)

# Order by degrees
print(head(tabularised[order(-degree), ]), digits = 2)
cat("\nDegree: ", degree, "\n\n")

# Order by Betweenness
print(head(tabularised[order(-betwness), ]), digits = 2)
cat("\nBetweenness: ", betwness, "\n\n")


# Order by Eigenvector Centrality
print(head(tabularised[order(-eig), ]), digits = 2)


threadID_frequency = ddply(web, .(ThreadID), function(df)
length(unique(df$AuthorID)))
colnames(threadID_frequency)[2] = c("PostNum")
threshold <- ceiling(median(threadID_frequency$PostNum))
summary(threadID_frequency$PostNum)
threadID_frequency = with(threadID_frequency, threadID_frequency[PostNum >
1,])
summary(threadID_frequency)
nrow(threadID_frequency)

threadID_post_frequency_authorID = threadID_frequency$ThreadID


## Extract information matching authorID post more than median
thread_list_by_author = web[web$ThreadID %in%
threadID_post_frequency_authorID,] %>%
                            dplyr::group_by(AuthorID) %>%
                            dplyr::summarise(
                              AuthorID = list(unique(ThreadID))
                            )


##### Apply combination function to each list of authors and combine them
by rows
connection_table_threadID = do.call(rbind,
lapply(thread_list_by_author$AuthorID, my_rel_comb))
```

```
connection_table_threadID = unique(connection_table_threadID) # Remove
those duplicate connections between two authors

#### Convert table (data frame) into graph
g2 <- graph_from_data_frame(connection_table_threadID,directed=F)
plot(g2,vertex.color = "red",main = "social network analysis")

degree <- centralization.degree(g2)$res
betwness <- centralization.betweenness(g2)$res
closeness <- centralization.closeness(g2)$res
eig <- centralization.evcent(g2)$res

tabularised_threadID <- as.data.frame(rbind(degree,betwness,closeness,eig))
tabularised_threadID <- t(tabularised_threadID)

#install.packages("sna")
# Density
library("sna")
degree(connection_table_threadID)

averagePath <- average.path.length(g2)
diameter <- diameter(g2)
cat("\nAverage Path Length: ", averagePath, "\n\n")
cat("\nDiameter: ", diameter, "\n\n")

print(tabularised,digits = 3)

# Order by degrees
print(head(tabularised[order(-degree), ]), digits = 2)
cat("\nDegree: ", degree, "\n\n")

# Order by Betweenness
print(head(tabularised[order(-betwness), ]), digits = 2)
cat("\nBetweenness: ", betwness, "\n\n")
```