Packages used:

- Corrgram
- tidyverse
- lubridate
- extrafont
- igraph
- fmsb
- Hmisc
- psych
- car

```
---
        title: "FIT3152 assignment 1 code"
        author: "Jason Ching Yuen Siu"
        output:
        word_document: default
  html_document: default
subtitle: Lingustic analysis using LIWC
        ---

```{r setup}
        knitr::opts_chunk$set(echo = TRUE,include=T, message =FALSE)
```
        Import needed library
```{r import-lib, message = FALSE, warning = FALSE}
        library(tidyverse)
        library(lubridate)
        library(extrafont) # for changing the font stlye of the graph
```
        Read the data
```{r read data}
        rm(list = ls())
        set.seed(31084222)
        data <-
read.csv("C:/Users/sjsa3/Desktop/Shared_with_Mac/year2_sem1/FIT3152/Assignment_FIT3152_202
1/webforum.csv")

        data <- data[sample(nrow(data),20000),] #20000 rows
```
        Clean the data
```{r min_max_normalisation }
        #define Min-Max normalisation method
        min_max_normalisation <- function(x) {
        (x - min(x)) / (max(x) - min(x))
        }
```
        Change the font style
```{r}
        changeFont <-  function(){
        theme_classic() + theme(text=element_text(family="Times New Roman", face="bold",
size=12)) #Times New Roman, 12pt, Bold
        }
```


```{r data modification}
        data$Date <- as.Date(data$Date)


        #check if there is any missing values
```
```

````
        sum(is.na(data))

        data_tidy <- data %>%
        mutate(month = month(Date,  label = TRUE, abbr = TRUE),
        wday = wday(Date, label = TRUE, abbr = TRUE, week_start = 1),
        year = year(Date),
        day = day(Date),
        hour =  hour(hm(data$Time)))
        data_tidy1 <- data_tidy
        #create a function for normalisation
        normalise_data_tidy <- function(x){
        #apply Min-Max normalisation to all numeric columns
        data_tidy_norm <- as.data.frame(lapply(x[,5:19], min_max_normalisation))
        return(data_tidy_norm)
        }
        data_tidy_norm <-  normalise_data_tidy(data_tidy)
```




==========================================================================================
====================



        - Q1

        a. How active are participants, are there periods where this increases or
decreases?
```{r}
        over_df = data_tidy %>% group_by(Date) %>% summarise(count =n())
        ggplot(over_df,
        aes(x = Date,
        y = count)) +
        geom_line() +
        stat_smooth()+labs(
        title = "The number of active authors over the years",
        subtitle = "(2002-2011)",
        x = "Year",
        y = "Activity volume"
        )+changeFont()
```

```{r hour}
        #Week-Day
        library(lubridate)
        hour_df = data_tidy %>% group_by(hour) %>% summarise(count=n())
        ggplot(hour_df,
        aes(x = hour,
        y = count))+labs(
        title = "The number of active authors over the hours",
        subtitle = "(00:00 - 24:00)",
        x = "Hour",
        y = "Activity volume"
        ) +
        geom_col()+theme_classic()+geom_smooth()+changeFont()
```
        ------------------------------------------------------------------------------------------
----------
        b Looking at the linguistic variables,
        1 do these change over time?
````
````

````
```{r grp-variable-by-yr}

        grp_yr = data.frame(data_tidy_norm ) %>% cbind(year = data_tidy$year)

        grp_yr = grp_yr %>% group_by(year) %>%
        summarise(count =n(),Tone = mean(Tone, na.rm = TRUE),
        WC = mean(WC,na.rm = TRUE),
        Analytic = mean(Analytic,na.rm = TRUE),
        Clout = mean(Clout,na.rm = TRUE),
        Authentic = mean(Authentic,na.rm = TRUE),
        WP = mean(WPS,na.rm = TRUE),
        i = mean(i,na.rm = TRUE),
        we = mean(we,na.rm = TRUE),
        you = mean(you,na.rm = TRUE),
        they = mean(they,na.rm = TRUE),
        number = mean(number,na.rm = TRUE),
        affect = mean(affect,na.rm = TRUE),
        posemo = mean(posemo,na.rm = TRUE),
        negemo = mean(negemo,na.rm = TRUE),
        anx = mean(anx,na.rm = TRUE))  %>%
        arrange(desc(count))
```


```{r vis-by-yr}

        ggplot(data = grp_yr)+geom_line(aes(year,Tone,colour  = "Tone"))+

        geom_line(aes(year,Authentic,colour  = "Authentic"))+

        geom_line(aes(year,Clout, colour  = "Clout"))+

        geom_line(aes(year,Analytic,colour ="Analytic"))+
        scale_colour_manual("",values =
        c( "Tone"="blue",
        "Authentic"="green",
        "Clout" = "black",
        "Analytic" = "red"))+
        ylim(0.36,.64)+labs(
        title = "The trend of summary variables over the years",
        subtitle = "(2002-2011)",
        x = "Year",
        y = "Active Authors"
        )+theme_minimal()+changeFont()
```


        Since Clout's change has been most turbulent, it is worth know which variables
define this variable.
        We will use linear regression model to know it
        ---------------------------------------------------------------------------------
----------
        c Is there a relationship between variables?
        ---------------------------------------------------------------------------------
----------



```{r}
        library(corrgram)
````
````

```
        corrgram(data_tidy_norm,upper.panel=panel.cor, main= "The correlation between
linguistic variables")

```

```
        Regression model

```{r lib-for-MLR, message=F}
        library(Hmisc)
        library(psych)
        library(car)
```

```{r view-summary}
        fit <- lm(affect ~ posemo +negemo, data=data_tidy)
        summary(fit)
```

```{r elim-xtreme-1}
        # crPlots(fit)

        # Eliminate extreme values
        cutoff <- 4/((nrow(data_tidy)-length(fit$coefficients)-2)) # Cook's D plot, cutoff
as 4/(n-k-1)
        plot(fit, which=4, cook.levels=cutoff)                        # identify D
values > cutoff

        data_tidy <- data_tidy[-which(rownames(data_tidy)    # Row names discovered in 2
rounds
        %in% c("19998", "5379", "14068")),]
```
        After refiting the model 1
```{r refit1}
        fit <- lm(affect ~ posemo+negemo, data=data_tidy)
        summary(fit)
```
```{r elim-xtreme-2}
        # crPlots(fit)

        # Eliminate extreme values
        cutoff <- 4/((nrow(data_tidy)-length(fit$coefficients)-2)) # Cook's D plot, cutoff
as 4/(n-k-1)
        plot(fit, which=4, cook.levels=cutoff)                        # identify D
values > cutoff

        data_tidy <- data_tidy[-which(rownames(data_tidy)    # Row names discovered in 2
rounds
        %in% c("19390", "16438", "5739")),]
```
        After refiting the model 2
```{r refit2}
        fit <- lm(affect ~ posemo+negemo+anx, data=data_tidy)
        summary(fit)$adj.r.squared # R2=81%, F=139.5
```
```{r elim-xtreme-3}
        # crPlots(fit)

        # Eliminate extreme values
        cutoff <- 4/((nrow(data_tidy)-length(fit$coefficients)-2)) # Cook's D plot, cutoff
```
```

```
as 4/(n-k-1)
        plot(fit, which=4, cook.levels=cutoff)                       # identify D
values > cutoff

        data_tidy <- data_tidy[-which(rownames(data_tidy)     # Row names discovered in 2
rounds
        %in% c("4561", "1755", "15379")),]
```

        After refiting the model 3

```{r refit3}
        fit <- lm(affect ~ posemo+negemo, data=data_tidy)
        summary(fit)$adj.r.squared
```

```{r av-plot}
        avPlots(fit, main = "The partial regression on affect given posemo and negemo",col
= carPalette()[7])

```

==
====================

- Q2

Analyse the language used by groups. Some starting points:

a Threads indicate groups of participants communicating on the same topic.
Describe the
threads present in your data.


```{r}
        df_tone <- data_tidy %>% group_by(ThreadID) %>% summarise(Tone = median(Tone))
        summary(df_tone$Tone)
```

```{r t-test-for-tone}
        t.test(data_tidy1$negemo, data_tidy1$posemo, conf.level = .99)
```

```{r day}
        df_for_donut_chart<- data_tidy %>%
        group_by(ThreadID) %>%
        summarise(Tone = median(Tone, na.rm = TRUE))
        df_for_donut_chart= df_for_donut_chart%>% mutate(Tone = ifelse(Tone >50 ,
"Positive", "Negative"))%>%
        group_by(Tone) %>% summarise(count =n())

        df_for_donut_chart <- df_for_donut_chart


        # Compute percentages
        df_for_donut_chart<-df_for_donut_chart%>%  mutate( fraction = count / sum(count))
```

```
        ggplot(df_for_donut_chart, aes(x=2,y=fraction,fill=Tone)) +
        geom_col()+
        coord_polar(theta="y",start = 1) +
        geom_text(aes(label= paste0(round(fraction*100),"%")),
        position = position_stack(vjust = .5))+
        theme(panel.background = element_blank(),
        axis.line = element_blank(),
        axis.text = element_blank(),
        axis.ticks = element_blank(),
        axis.title = element_blank(),
        plot.title = element_text(hjust = .5,size = 18)
        )+
        labs(title = "The proportion of Tone to all threads") +
        scale_fill_brewer(palette="BrBG") +
        xlim(0.5,2.5)+
        theme(text=element_text(family="Times New Roman", face="bold", size=12))
```

        1. Sentiments : are most the thread +ve ?
        2. Pronoun : What are the most used pronoun?
        3. Structure : What are the mean of WC and WPS?

        b By analysing the linguistic variables for all or some of the threads, is it
possible to see a
        difference in the language used by different groups?

        Find out the languages used between the most postivie and negative threads.
        1. find out the 10 most active threads

```{r top-10-Active}
        data_tidy_norm <- data_tidy_norm %>% cbind(ThreadID = data$ThreadID)
        df_active_10 <-  data_tidy %>% group_by(ThreadID)%>% summarise(count = n()) %>%
arrange(desc(count))
        df_active_10 <- head(df_active_10,10)
        df_active_10

        df <-  data_tidy1 %>% filter(data$ThreadID %in% df_active_10$ThreadID  ) %>%
arrange(ThreadID)
```

        2. In the most active threads, find out the most postivie and negative threads.
The data set is called data_nega_pose

```{r which-+ve}
        data_nega_pose <-  df %>% group_by(ThreadID) %>% summarise(mean = mean(Tone))
        data_nega_pose1 <- data_nega_pose
        data_nega_pose = data_nega_pose %>% mutate(emo = ifelse(mean >50 , "Positive",
"Negative"))%>%
        group_by(emo) %>% arrange(desc(mean))


        #It is noticed that Thread 472752 is the most Positive and 309286 is the most
Negative
```

````
        #most positve
        data.mostPose <- data_tidy %>% filter(ThreadID =="472752" )
        data.mostPose <- data.mostPose[5:19]
        data.mostPose1 <- data.mostPose

        #most negative
        data.mostNega <- data_tidy %>% filter(ThreadID =="309286" )
        data.mostNega <- data.mostNega[5:19]
        data.mostNega1 <- data.mostNega
```
````

Based on the data collected from the most negative and positive, we can make a Radarchart to visualise the difference of language used

````
```{r radar-chart-all-new }
        library(fmsb)
        data.mostPose <- data.mostPose %>% select(Analytic,Clout,Authentic,WC,WPS,affect)


        data.mostPose <- as.data.frame(lapply(data.mostPose[,], min_max_normalisation))
        data.mostPose <- data.mostPose %>% summarise(Analytic_m = mean(Analytic)*100,
        Clout_m = mean(Clout)*100,
        Authentic_m = mean(Authentic)*100,
        WC_m = mean(WC)*100,
        WPS_m = mean(WPS)*100,
        affect_m = mean(affect)*100
        )

        # negative
        data.mostNega <- data.mostNega %>% select(Analytic,Clout,Authentic,WC,WPS,affect)


        data.mostNega <- as.data.frame(lapply(data.mostNega[,], min_max_normalisation))
        data.mostNega <- data.mostNega %>% summarise(Analytic_m = mean(Analytic)*100,
        Clout_m = mean(Clout)*100,
        Authentic_m = mean(Authentic)*100,
        WC_m = mean(WC)*100,
        WPS_m = mean(WPS)*100,
        affect_m = mean(affect)*100
        )

        radar_data_Pose <- data.mostNega %>% rbind(data.mostPose )



        radar_data_Pose <-data.frame( Analytic = c(75, 0 ,
data.mostPose[1,1],data.mostNega[1,1] ),
        Clout = c(75, 0 , data.mostPose[1,2],data.mostNega[1,2]  ),
        Authentic = c(75, 0 , data.mostPose[1,3],data.mostNega[1,3]  ),
        WC = c(75, 0 , data.mostPose[1,4],data.mostNega[1,4]  ),
        WPS = c(75, 0 , data.mostPose[1,5],data.mostNega[1,5]  ),
        affect = c(75, 0 , data.mostPose[1,6],data.mostNega[1,6]  ),
        row.names = c("max","min","Positive","Negative")
        )
        #defien the colors filled
        colors_fill <- c(scales::alpha("yellow", 0.3),scales::alpha("black", 0.5))
        #define the line colors
        colors_line <- c(scales::alpha("black", 0.5),scales::alpha("darkgrey", 0.5))
````

````
        radarchart(radar_data_Pose, axistype = 1,
        seg = 2,
        # Customize the polygon
        pfcol =colors_fill, plwd = 2, plty = 1,
        # Customize the grid
        cglcol = "grey", cglty = 1, cglwd = 0.8,
        pcol = colors_line,
        # Customize the axis
        axislabcol = "grey",
        caxislabels = c(25, 50, 75),
        palcex = 1.5)

        title(main = "The linguistic variables comparsion between the most negative \nand
positive threads",
        cex.main = 1.1,
        font.main= 1,
        cex.sub = 0.75, font.sub = 1, col.sub = "green",
        col.lab ="darkblue")


        legend( x=1.3,y=1.3,legend = row.names( radar_data_Pose[3:4,] ) ,
        bty = "n",
        pch = 20,
        col=colors_fill,
        cex=1,
        pt.cex=3)

```
        The comparison of pronouns used between the most positive and negative
```{r radar-chart-all-pronoun }
        library(fmsb)
        data.mostPose1 <- data.mostPose1 %>% select(i ,we,they, you)
        data.mostPose1 <- as.data.frame(lapply(data.mostPose1[,], min_max_normalisation))
        data.mostPose1 <- data.mostPose1 %>% summarise(i_m = mean(i)*100,
        we_m = mean(we)*100,
        you_m = mean(you)*100,
        they_m = mean(they)*100
        )

        # negative
        data.mostNega1 <- data.mostNega1 %>% select(i ,we,they, you)
        data.mostNega1 <- as.data.frame(lapply(data.mostNega1[,], min_max_normalisation))
        data.mostNega1 <- data.mostNega1 %>% summarise(i_m = mean(i)*100,
        we_m = mean(we)*100,
        you_m = mean(you)*100,
        they_m = mean(they)*100
        )


        radar_data_Pose <- data.mostNega1 %>% rbind(data.mostPose1 )
        radar_data_Pose <-data.frame( i = c(30, 0 ,
data.mostPose1[1,1],data.mostNega1[1,1] ),
        we = c(30, 0 , data.mostPose1[1,2],data.mostNega1[1,2]  ),
        you = c(30, 0 , data.mostPose1[1,3],data.mostNega1[1,3]  ),
        they= c(30, 0 , data.mostPose1[1,4],data.mostNega1[1,4]  ),
        row.names = c("max","min","Positive","Negative")
        )

        radarchart(radar_data_Pose, axistype = 1,
````

~~~~
        seg = 2,
        # Customise the polygon
        pfcol =colors_fill, plwd = 2, plty = 1,
        # Customise the grid
        cglcol = "grey", cglty = 1, cglwd = 0.8,
        pcol = colors_line,
        # Customise the axis
        axislabcol = "grey",
        caxislabels = c(10, 20, 30))

        title(main = "The pronoun variables comparsion between \nthe most negative and
positive threads",
        cex.main = 1.1,
        font.main= 1,
        cex.sub = 0.75, font.sub = 1, col.sub = "green",
        col.lab ="darkblue")


        legend( x=1.3,y=1.3,legend = row.names( radar_data_Pose[3:4,] ) ,
        bty = "n",
        pch = 20,
        col=colors_fill,
        cex=1,
        pt.cex=3)
```



        c Does the language used within threads (or between threads) change over time? How
        consistent or variable is the language used within threads?
        -----------------------------------------------------------------------------------
----------
        The following graphs are to view the change and consistency
        **Language structure (LangStructure) = WPS + WC**
        The more the LangStructure, the more complex the structure is
```{r vis-lan-complexity-by-yr}

        data_tidy_norm <-  normalise_data_tidy(data_tidy1)
        data_tidy_norm <- data_tidy_norm %>% cbind(ThreadID = data_tidy1$ThreadID, Date
=data_tidy1$Date )



        data_tidy_norm <- data_tidy_norm %>%
select(Analytic,Clout,Tone,Authentic,WC,WPS,affect,ThreadID,Date, i ,we,they, you )%>%
        mutate(LangStructure = data_tidy_norm[,1] + data_tidy_norm[,6] )

        grp_yr = data.frame(data_tidy_norm ) %>% cbind(year = data_tidy1$year,
month=data_tidy1$month )

        grp_yr <- grp_yr %>% group_by(ThreadID,year,month) %>% summarise(count
=n(),LangStructure = mean(LangStructure,na.rm = TRUE),)%>%
        arrange(desc(year))

        # visualise it in a heatmap
        mycol <- c("navy", "blue", "cyan", "lightcyan", "yellow", "red", "red4")
        ggplot(grp_yr, aes(x= year , y=month, color =LangStructure )) +
        geom_tile (aes(fill=LangStructure),colour = "white" )+labs(
        title = "Time-Series Calendar Heatmap of the language struture",
        x = "Year",
        y = "Month"
~~~~

```
        )+
        scale_fill_gradientn(colours = mycol)+theme_bw()+
        changeFont()+
        theme(axis.text.x = element_text(, color="BLACK", angle=90))
```
```

```{r vis-pronoun-used-by-yr}

        data_tidy_norm <- data_tidy_norm %>% select(ThreadID,Date, i ,we,they, you )


        grp_yr <- data.frame(data_tidy_norm ) %>% cbind(year = data_tidy1$year,
month=data_tidy1$month )

        grp_yr <- grp_yr %>% group_by(ThreadID,year,month) %>% summarise(count
=n(),i=median(i),
        we=median(we),
        you=median(you),
        they = median(they))%>%
        arrange(desc(year))

        grp_yr <-grp_yr %>%   pivot_longer(cols = c(`i`,`we`,`you`,`they`), names_to =
"Pronoun", values_to= "Values")

        mycol <- c("navy", "blue", "cyan", "lightcyan", "yellow", "red", "red4")
        ggplot(grp_yr, aes(x= year , y=month, color = Values)) +
        geom_tile (aes(fill=Values),colour = "white" ) +
        scale_fill_gradientn(colours = mycol)+
        facet_grid(~Pronoun)+
        theme_linedraw()+
        theme(axis.text.x = element_text(, color="BLACK", angle=90))+labs(
        title = "Time-Series Calendar Heatmap: Pronoun of I, They, We, You",
        x = "Year",
        y = "Month"
        )  +theme(text=element_text(family="Times New Roman", face="bold", size=12))

```
```

==
====================

Q3 Challenge: Social networks online

```
```{r filter-data}
        #filter data for social network analysis
        filter_dt <- function(yr,month1){
        df <-  data_tidy1 %>% filter(year == yr, month == month1)
        df <- df %>% select(ThreadID, AuthorID)
        df <- inner_join(df, df, by = "ThreadID")
        df3 <- apply(df, 2, as.character) #AuthorID as character will become vertex ID
        df3 <- as.tibble(df3)
        df3 <- df3 %>% rename(sources = AuthorID.x, destinations= AuthorID.y)
        df3 <- df3 %>% filter(sources != destinations)
        return(df3)
        }
```
```

This is to create tabulate_dt() for tabulising the adjacency of nodes

```
```{r tabulate_dt}
        library(igraph)
```
```

~~~~
        #set a class of table so that I can return multiple
        #objects for the function of tabulate_dt
        setClass(Class = "Table",representation (edges ="list",
        nodes ="list"))

        tabulate_dt <-  function(df3){

        sources <- df3  %>%
        distinct(sources) %>%
        rename(label = sources)

        ## take destination from letters and make it as "destinations" and renamed as
"label"
        destinations <- df3 %>%
        distinct(destinations) %>%
        rename(label = destinations)


        ##To create a single dataframe with a column with the unique locations
        nodes <- full_join(sources, destinations, by = "label")
        nodes <- nodes %>% rowid_to_column("id")

        per_route <- df3 %>%
        group_by(sources, destinations) %>%
        summarise(weight = n()) %>%
        ungroup()


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

        edges <- edges %>%
        left_join(nodes, by = c("destinations" = "label")) %>%
        rename(to = id)

        edges <- select (edges, from, to, weight)

        #this will return an instance of this class -- Table
        return(new("Table",edges=edges,nodes=nodes))
        }
```


        create function (network_centrality) for centrality summary of node within the
network
```{r network-summary}

        network_centrality <- function(routes_igraph){
        degree_table <-  as.table( degree(routes_igraph))
        betweenness_table <-  as.table( betweenness(routes_igraph))
        closeness_table <-  as.table( closeness(routes_igraph))
        eigenvector_table <-  as.table(    evcent(routes_igraph)$vector)

        #merge table
        bt_degree <- merge(degree_table, betweenness_table,by= "Var1" )
        bt_degree <-  bt_degree %>% rename(Betweenness = Freq.x, Degree=Freq.y , id=Var1)

        cls_eig <- merge(closeness_table,eigenvector_table, by= "Var1" )
        cls_eig <-  cls_eig %>% rename(Closeness = Freq.x, Eigenvector=Freq.y,id=Var1)
~~~~

````
        network_summary <-  merge(bt_degree,cls_eig,by= "id")
        network_summary <-  merge(nodes,network_summary,by="id") %>% select(-id)

        #round all numbers in 2 digits
        network_summary <- network_summary %>%
        mutate(Degree = round(Degree,2),
        Closeness =round( Closeness,2),
        Eigenvector = round(Eigenvector,2))

        network_summary <-  network_summary %>%
arrange(desc(Betweenness,Degree,Eigenvector,Closeness) )
        return(network_summary)
        }
```


        Lets have a look at the network of 2002 feb
```{r summary-feb}
        df3 <- filter_dt(2002,"Feb")
        feb_table <- tabulate_dt(df3)
        # access your information of feb_table and make them as dataframe
        edges <- as.data.frame(feb_table@edges)
        nodes <- as.data.frame(feb_table@nodes)

        routes_igraph <- graph_from_data_frame(d = edges,  vertices=nodes, directed = F)

        network_centrality(routes_igraph)
        #count the vertexes and nodes
        vcount(routes_igraph)
        ecount(routes_igraph)
```
```{r visualise-2002-feb}

        plot(routes_igraph,
        vertex.shape="none",
        edge.curved=TRUE)


        plot(routes_igraph, layout = layout_in_circle(routes_igraph),
        vertex.shape="none",
        edge.curved=TRUE)
        # change the degree size
        # plot(routes_igraph, layout = layout_with_graphopt,
        #      edge.arrow.size = 0.2,
        #      vertex.size=deg*3,
        #      vertex.color=rgb(0.1,0.7,0.8,0.5) )
```


        network for 2002 march
```{r summary-march}
        df3 <- filter_dt(2002,"Mar")
        mar_table <- tabulate_dt(df3)
        # access your information of feb_table and make them as dataframe
        edges <- as.data.frame(mar_table@edges)
        nodes <- as.data.frame(mar_table@nodes)

        routes_igraph <- graph_from_data_frame(d = edges, vertices = nodes, directed = F)
        network_centrality(routes_igraph)
        #count the vertexes and nodes
````

```
       vcount(routes_igraph)
       ecount(routes_igraph)
```

```{r visualise-2002-mar}
       plot(routes_igraph,
       vertex.shape="none",
       edge.curved=TRUE)


       plot(routes_igraph, layout = layout_in_circle(routes_igraph),
       vertex.shape="none",
       edge.curved=TRUE)

```