Data Science Capstone Project - Week 2 Assignment

The goal of this project is just to display that you've gotten used to working with the data and that you are on track to create your prediction algorithm. Please submit a report on R Pubs (http://rpubs.com/) that explains your exploratory analysis and your goals for the eventual app and algorithm. This document should be concise and explain only the major features of the data you have identified and briefly summarize your plans for creating the prediction algorithm and Shiny app in a way that would be understandable to a non-data scientist manager. You should make use of tables and plots to illustrate important summaries of the data set.

The motivation for this project is to:

1. Demonstrate that you've downloaded the data and have successfully loaded it in.
2. Create a basic report of summary statistics about the data sets.
3. Report any interesting findings that you amassed so far.
4. Get feedback on your plans for creating a prediction algorithm and Shiny app.

Data set Downloading and Loading into R

Download the data from following link and unzip the files in the current working directory - https://d396qusza40orc.cloudfront.net/dsscapstone/dataset/Coursera-SwiftKey.zip

Loading the Data into R

For the demonstation perpose we are using the English Language text data.

```
    setwd("./Coursera-SwiftKey/final/en_US")


    con <- file("en_US.news.txt", open="r")
    En_US_NEWS_text <- readLines(con); close(con)


    con <- file("en_US.blogs.txt", open="r")
    En_US_blogs_text <- readLines(con); close(con)


    con <- file("en_US.twitter.txt", open="r")
    En_Twit_text <- readLines(con); close(con)
```

Data set summary statistics details

Extracting the following text files summary

```
- en_US.news.txt
- en_US.blogs.txt
- en_US.twitter.txt
```

```
file_stat<- function(text_file, lines) {
    f_size <- file.info(text_file)[1]/1024^2
    nchars <- lapply(lines, nchar)
    maxchars <- which.max(nchars)
    word_count <- sum(sapply(strsplit(lines, "\\s+"), length))
    return(c(text_file, format(round(as.double(f_size), 2), nsmall=2), length(lines),maxchars, word_count))
}


    En_US_news_stat<- file_stat("en_US.news.txt", En_US_NEWS_text)
    En_US_blogs_stat <- file_stat("en_US.blogs.txt", En_US_blogs_text)
    En_Twit_text_stat<- file_stat("en_US.twitter.txt", En_Twit_text)


    test_summary <- c(En_US_news_stat, En_US_blogs_stat,En_Twit_text_stat)


    df <- data.frame(matrix(unlist(test_summary), nrow=3, byrow=T))
    colnames(df) <- c("Text_file", "Size(MB)", "Line_Count", "Max Line Length", "Words_Count")
    print(df)
```

```
##           Text_file Size(MB) Line_Count Max Line Length Words_Count
## 1    en_US.news.txt   196.28      77259           14556     2643972
## 2   en_US.blogs.txt   200.42     899288          483415    37334441
## 3 en_US.twitter.txt   159.36    2360148         1484357    30373792
```

Exploratory data analysis

Here I am writing a functions to make the test data Corpus, Clean the corpus, and capture the hight frquency words

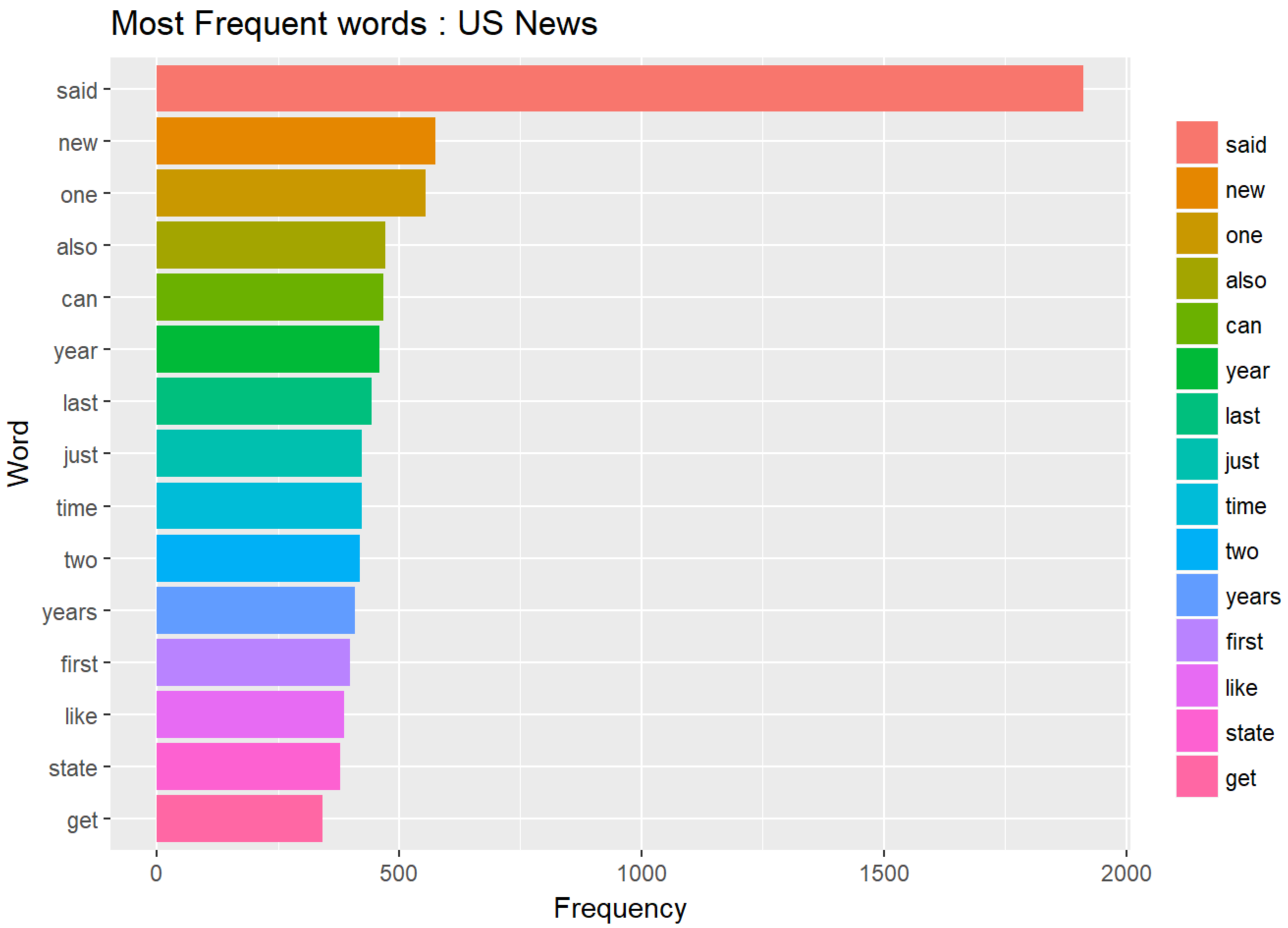
```
make_Corpus<- function(test_file) {
    gen_corp<- paste(test_file, collapse=" ")
    gen_corp <- VectorSource(gen_corp)
    gen_corp <- Corpus(gen_corp)
}


clean_corp <- function(corp_data) {


    corp_data <- tm_map(corp_data, removeNumbers)
    corp_data <- tm_map(corp_data, content_transformer(tolower))
    corp_data <- tm_map(corp_data, removeWords, stopwords("english"))
    corp_data <- tm_map(corp_data, removePunctuation)
    corp_data <- tm_map(corp_data, stripWhitespace)
    return (corp_data)
}
```

```
high_freq_words <- function (corp_data) {
    term_sparse <- DocumentTermMatrix(corp_data)
    term_matrix <- as.matrix(term_sparse)   ## convert our term-document-matrix into a normal matrix
    freq_words <- colSums(term_matrix)
    freq_words <- as.data.frame(sort(freq_words, decreasing=TRUE))
    freq_words$word <- rownames(freq_words)
    colnames(freq_words) <- c("Frequency","word")
    return (freq_words)
}
```

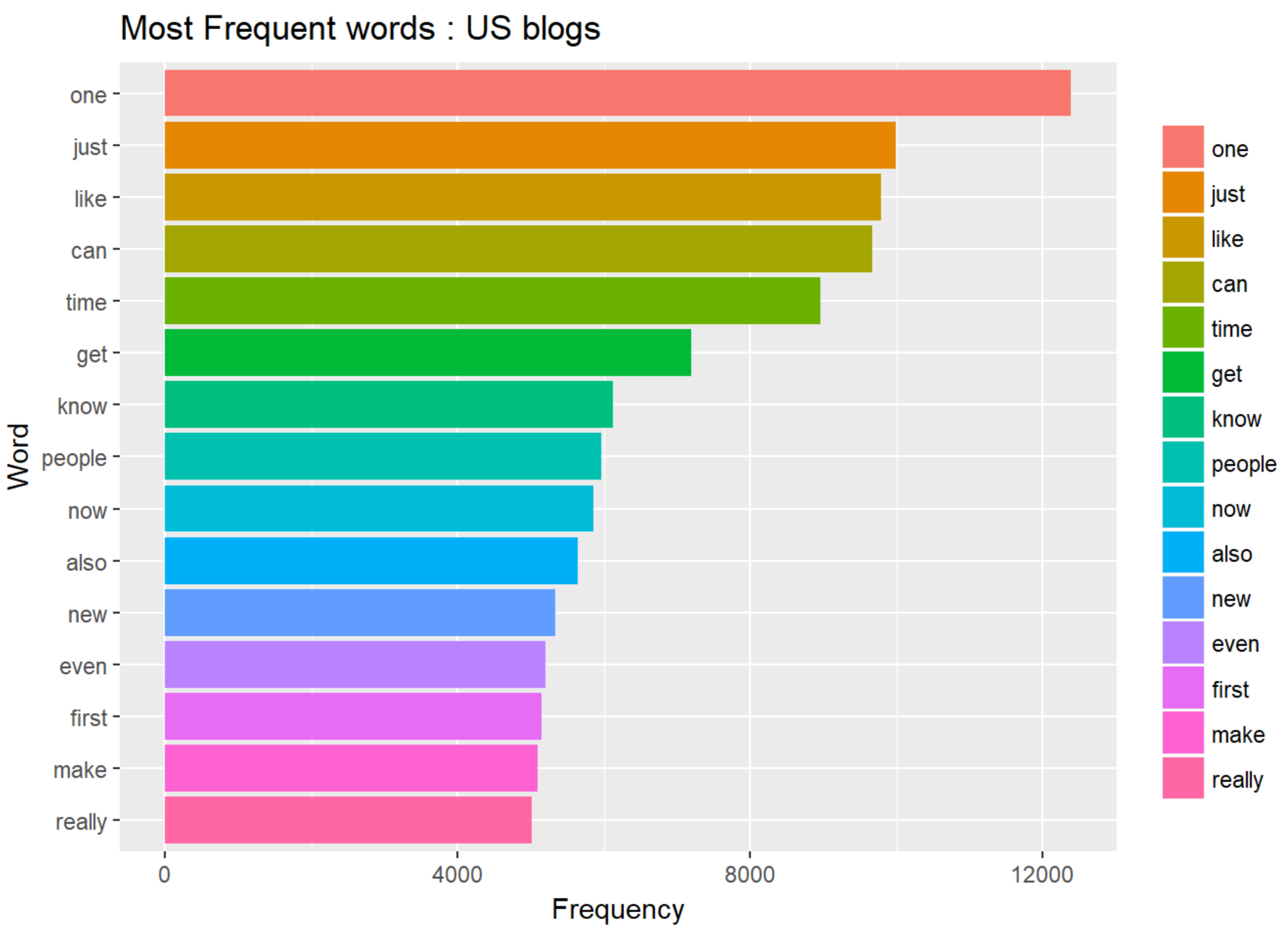
Bar Chart of High frequency words

This section is explore the different text mining commads and extract the high frequency words

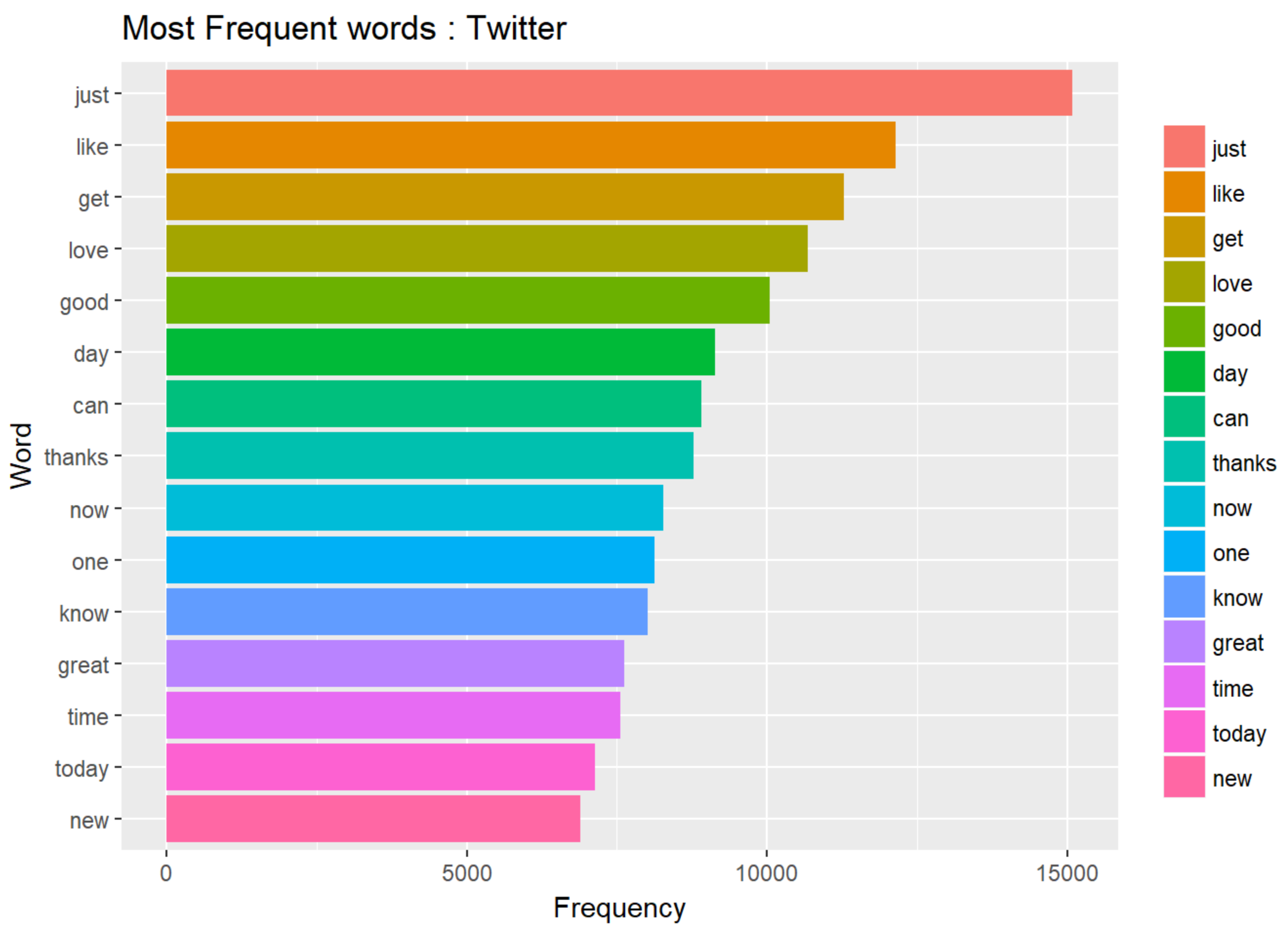
```
## en_US.news.txt High frequency words
    En_US_NEWS_text1<-sample(En_US_NEWS_text, round(0.1*length(En_US_NEWS_text)), replace = F)
    US_news_corpus <- make_Corpus(En_US_NEWS_text1)
    US_news_corpus <- clean_corp(US_news_corpus)
    US_news_most_used_word <- high_freq_words(US_news_corpus)
    US_news_most_used_word1<- US_news_most_used_word[1:15,]

    p<-ggplot(data=US_news_most_used_word1, aes(x=reorder(word,Frequency), y=Frequency,
                    fill=factor(reorder(word,-Frequency))))+ geom_bar(stat="identity")
    p + xlab("Word") +labs(title = "Most Frequent words : US News") +theme(legend.title=element_blank()) + coord_flip()
```



```
## en_US.blogs.txt High frequency words

    En_US_blogs_text1<-sample(En_US_blogs_text, round(0.1*length(En_US_blogs_text)), repl
ace = F)

    US_blogs_corpus <- make_Corpus(En_US_blogs_text1)

    US_blogs_corpus <- clean_corp(US_blogs_corpus)

    US_blogs_most_used_word <- high_freq_words(US_blogs_corpus)

    US_blogs_most_used_word1<- US_blogs_most_used_word[1:15,]


    p<-ggplot(data=US_blogs_most_used_word1, aes(x=reorder(word,Frequency), y=Frequency,

                    fill=factor(reorder(word,-Frequency))))+ geom_bar(stat="identity")

    p + xlab("Word") +labs(title = "Most Frequent words : US blogs") +theme(legend.title=
element_blank()) + coord_flip()
```



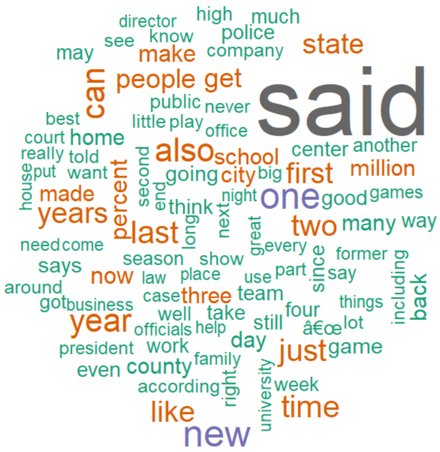
```
## en_US.twitter.txt High frequency words
    En_Twit_text1<-sample(En_Twit_text, round(0.1*length(En_Twit_text)), replace = F)
    twitter_corpus <- make_Corpus(En_Twit_text1)
    twitter_corpus <- clean_corp(twitter_corpus)
    twitter_most_used_word <- high_freq_words(twitter_corpus)
    twitter_most_used_word1<- twitter_most_used_word[1:15,]

    p<-ggplot(data=twitter_most_used_word1, aes(x=reorder(word,Frequency), y=Frequency,
                    fill=factor(reorder(word,-Frequency))))+ geom_bar(stat="identity")
    p + xlab("Word") +labs(title = "Most Frequent words : Twitter") +theme(legend.title=e
lement_blank()) + coord_flip()
```



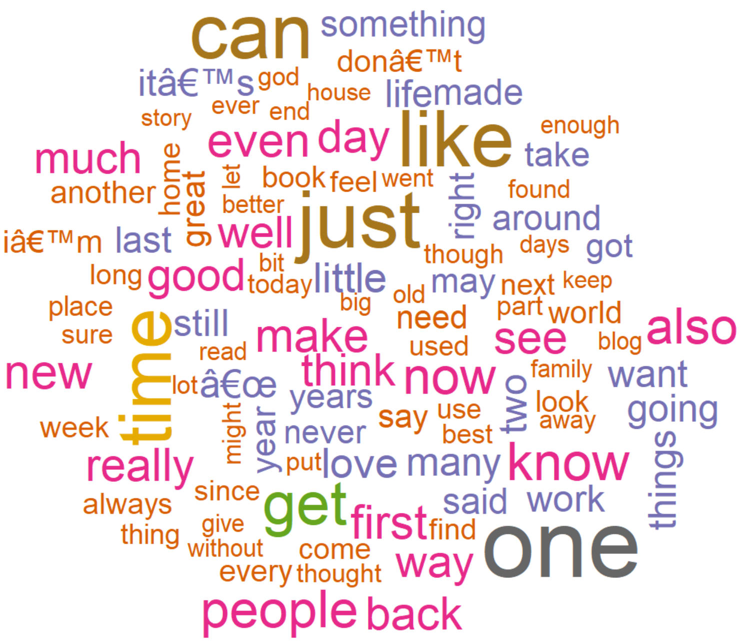
Generating the Word Cloud

Word Cloud is Cool representation of the Word display based on the Frequencies.

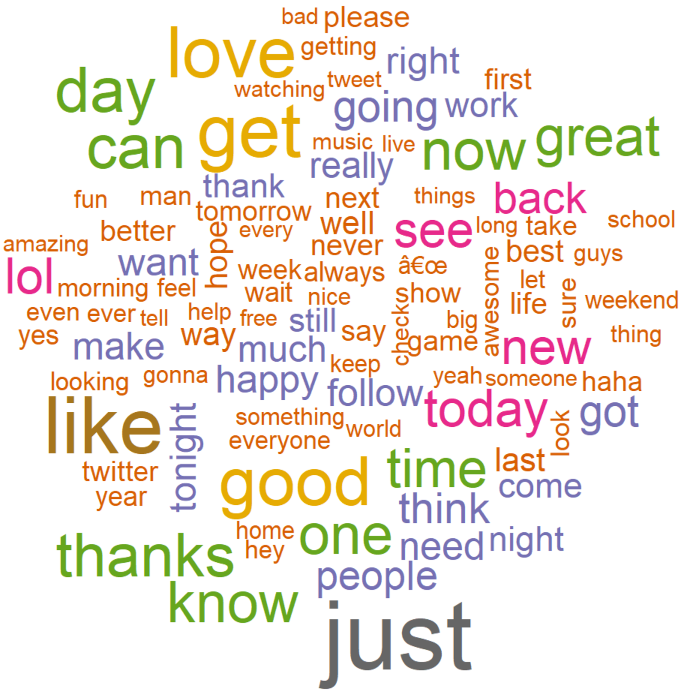
```
## US News Word Cloud
    wordcloud(US_news_most_used_word$word[1:100], US_news_most_used_word$Frequency[1:100]
,
              colors=brewer.pal(8, "Dark2"))
```



```
## US News Word Cloud

    wordcloud(US_blogs_most_used_word$word[1:100], US_blogs_most_used_word$Frequency[1:10
0],

              colors=brewer.pal(8, "Dark2"))
```



```
## US Twitter Word Cloud

    wordcloud(twitter_most_used_word$word[1:100], twitter_most_used_word$Frequency[1:100]
,
              colors=brewer.pal(8, "Dark2"))
```



Word Analysis

For the Data analysis of text document we need to create a bag of word matrices with Unigram, Bigram, Trigrams. These Ngram model set improve the predictabily of the data analysis.

```
## en_US.news.txt High frequency words

    En_US_NEWS_text1<-sample(En_US_NEWS_text, round(0.01*length(En_US_NEWS_text)), replac
e = F)

    US_News_tokens<- tokens(En_US_NEWS_text1,what ="word", remove_numbers = TRUE,
                            remove_punct = TRUE, remove_separators = TRUE, remove_symbols
=TRUE )
    US_News_tokens <- tokens_tolower(US_News_tokens)
    US_News_tokens <- tokens_select(US_News_tokens, stopwords(),selection ="remove")

    US_News_unigram <- tokens_ngrams(US_News_tokens, n=1)  ## unigram
    US_News_unigram.dfm <- dfm(US_News_unigram, tolower =TRUE, remove = stopwords("englis
h"),
                              remove_punct = TRUE)

    US_News_bigram <- tokens_ngrams(US_News_tokens, n=2)  ## bigram
    US_News_bigram.dfm <- dfm(US_News_bigram, tolower =TRUE, remove = stopwords("english"
),
                              remove_punct = TRUE)

    US_News_trigram <- tokens_ngrams(US_News_tokens, n=3)  ## trigram
    US_News_trigram.dfm <- dfm(US_News_trigram, tolower =TRUE, remove = stopwords("englis
h"),
                              remove_punct = TRUE)
    topfeatures(US_News_unigram.dfm, 20)  # 20 top US News Unigram words
```

```
##    said      â    new      s   also   time   just    one    two  state
##     191    132     62     58     51     46     45     44     43     41
##     can  years  first   like    now   many   year people   city   even
##      39     39     39     38     35     35     33     30     29     29
```

```
    topfeatures(US_News_bigram.dfm, 20)  # 20 top US News Bigram words
```

```
##          said_â       new_york        a.m_p.m           â_□i       st_louis
##              10              9              9              8              8
##     los_angeles         â_said     new_jersey      years_ago           kd_â
##               8              8              7              7              6
##      petre_kick         canâ_t            â_â fourth_quarter        kick_kd
##               6              5              5              5              5
##            iâ_m  united_states     four_times vice_president         â_□itâ
##               4              4              4              4              4
```

```
    topfeatures(US_News_trigram.dfm, 20)  # 20 top US News Trigram words
```

```
##                      petre_kick_kd                         kick_kd_â
##                                  5                                 5
##                           â_□itâ_s                          â_said_â
##                                  4                                 3
##                            â_□iâ_m                    run_petre_kick
##                                  3                                 3
##                     said_â_□thatâ                        â_□thatâ_s
##                                 2                                 2
##                  smoke_cedar_bark republican_presidential_candidate
##                                 2                                 2
##         housing_urban_development                    women_st_louis
##                                 2                                 2
##                         said_â_□i                  east_los_angeles
##                                 2                                 2
##           just_like_shakespeare's       members_congressional_black
##                                 2                                 2
##        congressional_black_caucus            president_barack_obama
##                                 2                                 2
##          cable_networks_broadcast             networks_broadcast_tv
##                                 2                                 2
```

```
## en_US.blog.txt High frequency words
    En_US_blogs_text1<-sample(En_US_blogs_text, round(0.02*length(En_US_blogs_text)), replace = F)
    US_blogs_tokens<- tokens(En_US_blogs_text1,what ="word", remove_numbers = TRUE,
                            remove_punct = TRUE, remove_separators = TRUE, remove_symbols =TRUE )
    US_blogs_tokens <- tokens_tolower(US_blogs_tokens)
    US_blogs_tokens <- tokens_select(US_blogs_tokens, stopwords(),selection ="remove")

    US_blogs_unigram <- tokens_ngrams(US_blogs_tokens, n=1)  ## unigram
    US_blogs_unigram.dfm <- dfm(US_blogs_unigram, tolower =TRUE, remove = stopwords("english"),
                              remove_punct = TRUE)

    US_blogs_bigram <- tokens_ngrams(US_blogs_tokens, n=2)  ## bigram
    US_blogs_bigram.dfm <- dfm(US_blogs_bigram, tolower =TRUE, remove = stopwords("english"),
                              remove_punct = TRUE)

    US_blogs_trigram <- tokens_ngrams(US_blogs_tokens, n=3)  ## tiigram
    US_blogs_trigram.dfm <- dfm(US_blogs_trigram, tolower =TRUE, remove = stopwords("english"),
                              remove_punct = TRUE)
    topfeatures(US_blogs_unigram.dfm, 20)  # 20 top US blogs Unigram words
```

```
##      â      s    one   just   like    can      t   time    get   know
##   4614   3374   2494   1992   1971   1958   1858   1781   1336   1208
##    now    new     iâ people   also   back  first   even   make  think
##   1189   1102   1086   1086   1074   1055   1022   1022   1006    985
```

```
    topfeatures(US_blogs_bigram.dfm, 20)  # 20 top US blogs Bigram words
```

```
##    itâ_s     iâ_m   donâ_t    iâ_ve  didnâ_t  thatâ_s   canâ_t doesnâ_t
##      716      554      545      282      247      221      180      164
##  youâ_re thereâ_s      â_â     iâ_d    iâ_ll   isnâ_t  wasnâ_t   â_□the
##      160      135      126      123      122      121      118      114
##   t_know     â_□i    heâ_s   said_â
##      114      107      107      100
```

```
    topfeatures(US_blogs_trigram.dfm, 20)  # 20 top US blogs Trigram words
```

```
##           donâ_t_know             iâ_m_sure          donâ_t_think
##                    89                    45                    45
##            iâ_m_going                 ã_ã_ã           donâ_t_want
##                    40                    37                    36
##              â_□itâ_s                 â_â_â            donâ_t_get
##                    27                    26                    24
##           think_itâ_s            itâ_s_like accounting_jobs_italy
##                    21                    20                    20
##            itâ_s_hard          didnâ_t_know           donâ_t_need
##                    19                    19                    19
##         new_york_city               â_□iâ_m            itâ_s_just
##                    18                    18                    18
##              â_said_â             know_iâ_m
##                    16                    15
```

```
## en_US.twitter.txt Ngram words
    En_Twit_text1<-sample(En_Twit_text, round(0.02*length(En_Twit_text)), replace = F)
    twitter_tokens<- tokens(En_Twit_text1,what ="word", remove_numbers = TRUE,
                            remove_punct = TRUE, remove_separators = TRUE, remove_symbols
=TRUE )
    twitter_tokens <- tokens_tolower(twitter_tokens)
    twitter_tokens <- tokens_select(twitter_tokens, stopwords(),selection ="remove")


    twitter_unigram <- tokens_ngrams(twitter_tokens, n=1)  ## unigram
    twitter_unigram.dfm <- dfm(twitter_unigram, tolower =TRUE, remove = stopwords("englis
h"),
                              remove_punct = TRUE)


    twitter_bigram <- tokens_ngrams(twitter_tokens, n=2)  ## bigram
    twitter_bigram.dfm <- dfm(twitter_bigram, tolower =TRUE, remove = stopwords("english"
),
                              remove_punct = TRUE)


    twitter_trigram <- tokens_ngrams(twitter_tokens, n=3)  ## trigram
    twitter_trigram.dfm <- dfm(twitter_trigram, tolower =TRUE, remove = stopwords("englis
h"),
                              remove_punct = TRUE)
    topfeatures(twitter_unigram.dfm, 20)  # 20 top Unigram words
```

```
##   just   like    get   love   good    day    can     rt thanks    one
##   3079   2460   2205   2144   2021   1860   1807   1766   1695   1653
##    now   know      u  great   time  today    lol     go    new    see
##   1641   1640   1555   1520   1513   1478   1430   1419   1380   1290
```

```
    topfeatures(twitter_bigram.dfm, 20)  # 20 top Bigram words
```

```
##             â_□       right_now  happy_birthday      last_night
##             423             353             217             202
##    good_morning looking_forward       feel_like      looks_like
##             165             164             143             142
##       good_luck   thanks_follow        let_know        just_got
##             132             131             131             126
##     follow_back       next_week         can_get           ðÿ_ðÿ
##             119             114             107              90
##       make_sure       great_day       look_like       thanks_rt
##              86              85              81              80
```

```
    topfeatures(twitter_trigram.dfm, 20)  # 20 top  Trigram words
```

```
##            let_us_know      happy_mothers_day     happy_mother's_day
##                     54                     37                     32
##         happy_new_year               ðÿ_ðÿ_ðÿ                 rt_â_□
##                     30                     28                     26
##          cinco_de_mayo  good_morning_everyone                  î_î_î
##                     18                     17                     16
##                  â_â_â looking_forward_seeing               la_la_la
##                     14                     14                     14
##    thanks_following_us     follow_back_please         add_boston_add
##                     13                     12                     12
##      boston_add_boston   please_please_please         love_love_love
##                     12                     12                     11
##          just_got_home      sounds_like_great
##                     11                     11
```

Interesting findings that you amassed so far

I have gone through the multiple literatures and youtube vidios on Text mining and “quanteda” library With Small text data sets problems how the data set will get exploded with different ngrams and Bag of words. I found quanteda library is very useful in generating the text analytics. Which very fast compare to TM library. This project motivated me to work on small samples sets to establish the my code.

Get feedback on your plans for creating a prediction algorithm and Shiny app.

Plan of Approach:

```
- Tockenization and bag of words with multiple Ngrams.

- Since We need build the shiny app which have limitation on resources. I will use the sm
all sample (~ 1        to 5%).

- I will explore the options for data compression. Run the Machine Learning programs to d
evelop the         predictive model.

- Explore the options to improve the accuracy and speed of execution.
```