Social Media Analytics for Canadian Banks (knitR Report)

Chris Tan, Student ID 303428

26 August 2019

Abstract This is for the fulfillment of the York University's Advanced Analytics Course Capstone Project. The aim of this project is to uncover insights from the social media space through programmatic means.

Project Scope

Here are the boundaries of the project:

1. Social media channel: Twitter (to include Facebook if time permits)
2. Social media scope: Major Canadian Financial Institutions (FI) like BMO, CIBC, RBC, Scotiabank, TD (to include digital banks like Simplii, Tangerine, EQ if time permits)
3. Comparison of the following insights across the above FIs: Sentiment Analysis (polarity and categorical); Word Cloud (conversation drivers); Key-word dendrogram (blend of sentiment and conversation drivers); Network Analysis (demographics and product segmentation). Paraphrases of these insights are given in the "Research Questions" section below

Research Questions

Here are the research questions for this project:

1. Which bank has the most favourable / unfavourable trending opinion?

2. What are the current financial products being discussed?

3. What are the current emotions (anger, fear, anticipation, trust, surprise, sadness, joy, and disgust) towards each bank?

4. What are the current sentiments towards trending financial product segments / categories (and the general network of terms being tweeted)?

Research Questions (revised)

Based on the exploratory data analysis (EDA) from Sprint #1, the research questions are revised as follows:

1. Which bank has the most favourable / unfavourable trending opinion?
Comments: About 1,623 tweets have been collected since July 7th, 2019, with close 5,500 terms. The collection will increase in the next several weeks. It should be feasible to answer this research question. The main drawback is for the low count of CIBC tweets (40 tweets) versus that of Scotia Bank (661 tweets). The wide difference will skew the analysis, especially that of CIBC's

2. What are the current financial products being discussed?
Comments: The EDA shows that frequent terms related to banking products are generic ones, for example, stock, charges, account. Unless we have a much more collection of tweets, it will be difficult to objectively address this research question

3. What are the current emotions (anger, fear, anticipation, trust, surprise, sadness, joy, and disgust) towards each bank?
Comments: Not shown in this Sprint#1 report as the codes are still experimental, the author managed to "see" these emotional terms at the AllBanks level. Again, due to the low tweet count for CIBC, it may be difficult to pin down the sentiments, especially for these 8 sentiment categories.

4. What are the current sentiments towards trending financial product segments / categories (and the general network of terms being tweeted)?
Comments: As stated above, frequent terms related to banking products are generic ones, hence it will be difficult to assess sentiments towards product segments. Network of terms is certainly a possibility

Introduction and overall approach

Here is the general approach I adopted for this project:

<u>Sprint #1</u>:

1. Data Preparation
   - 1.1 1.1 Preliminaries - Load libraries and set seed / working directory
   - 1.2 Data Access
   - 1.3 Data extraction using twitteR
   - 1.4 Data Storage

2. Exploratory Data Analysis
   - 2.1 Create the term-document matrix
   - 2.2 Remove terms which have at least 99% of sparse elements
   - 2.3 Visualise term counts (or frequency)
   - 2.4 Data processing and normalisation
     - 2.4.1 Remove stop words
     - 2.4.2 Remove terms which have at least 99% of sparse elements
     - 2.4.3 Visualise term counts (or frequency)
   - 2.5 Additional data processing and normalisation
   - 2.6 Explore clustering of terms/tweets using k-means method
   - 2.7 Plot the hierarchical clusters
   - 2.8 Find the pair of terms that appears frequently together
     - 2.8.1 Pair of terms that appears frequently together
     - 2.8.2 Find the network of terms
   - 2.9 Wordcloud
   - 2.10 Sentiment analysis
   - 2.11 Observations

<u>Sprint #2</u>:

3. Social Media Analytics for Canadian Banks
   - 3.1 Create and compact the term-document matrix (TDM) for each bank
   - 3.2 Further compact the term-document matrix
   - 3.3 Clustering of terms/tweets k-means method
   - 3.4 Plot the hierarchical clusters
   - 3.5 Find the pair of terms that appears frequently together
     - 3.5.1 Find the network of terms
     - 3.5.2 Find optimal term count for optimal (neither too busy nor sparse) visualisation of network diagram

Steps employed in Social Media (Twitter) Analytics using R

1. Data Preparation

1.1 Preliminaries - Load libraries and set seed / working directory

```
```{r}
options(warn=-1) # Suppress warnings to make output more readable
# This is tweets data extraction and other utilities specific to Twitter
suppressMessages(library(twitteR))

# Core data analytics packages like ggplot2, dplyr, tidyr, readr, purrr, tibble, stringr,
forcats
suppressMessages(library(tidyverse))

# Primary text mining package
suppressMessages(library(tm))

suppressMessages(library(wordcloud))

suppressMessages(library(syuzhet)) # Extracts Sentiment and Sentiment-Derived Plot Arcs
from Text
suppressMessages(library(lubridate)) # For ease of analysis on date fields

# Graphical scales map data to aesthetics, and provide methods for automatically
determining breaks and labels for axes and legends
suppressMessages(library(scales))

suppressMessages(library(reshape2)) # Flexibly restructure and aggregate data using just
two functions: melt and 'dcast'

suppressMessages(library(igraph)) # Network Analysis and Visualization

set.seed(123)

setwd("/Users/sgchr/Documents/CSDA1050/Data/")
```
```

1.2 Data Access

Because the four keys and tokens below are confidential, I have commented the code. Uncomment the codes, enter your own tokens and keys before running the codes

```
```{r}
### set the credentials
#CONSUMER_SECRET <- 'Your CONSUMER_SECRET'
#CONSUMER_KEY <- 'Your CONSUMER_KEY'
#ACCESS_TOKEN <- 'Your ACCESS_TOKEN'
#ACCESS_TOKEN_SECRET <- 'Your ACCESS_TOKEN_SECRET'

### Connect to twitter app. Select 2 in the console
#setup_twitter_oauth(CONSUMER_KEY, CONSUMER_SECRET, ACCESS_TOKEN, ACCESS_TOKEN_SECRET)
```
```

1.3 Data extraction using twitteR

Like above, uncomment codes below before running them
```{r}
# Get tweets separately for each bank for two reasons (1) Twitter's free developer account has limits to search terms (2) Easier to do analysis by individual banks later

### Get Cibc tweets
#Cibc <- searchTwitter("CIBC", n=4000, lang="en", since='2019-07-07', until='2019-08-10')

### Get Rbc tweets
#Rbc <- searchTwitter("rbc", n=4000, lang="en", since='2019-07-07', until='2019-08-10')

### Get Td tweets
#Td <- searchTwitter("TD bank", n=4000, lang="en", since='2019-07-07', until='2019-08-10')

### Get Bmo tweets
#Bmo <- searchTwitter("bmo", n=4000, lang="en", since='2019-07-07', until='2019-08-10')

### Get Bns tweets
#Bns <- searchTwitter("scotiabank", n=4000, lang="en", since='2019-07-07', until='2019-08-10')

### Check remaining rate limits to avoid penalty from Twitter
#RL <- getCurRateLimitInfo()
```
Notes:

1. I found that searchTwitter using multiple terms does not seem to work, even when the syntax is correct (see example below)

```
terms <- c("cibc","Canadian Imperial Bank of Commerce", "CanadianImperialBankofCommerce", "CIBCForTheFans")
terms_search <- paste(terms, collapse = " OR ") # Insert "OR between each term
Cibc <- searchTwitter(terms_search, n=4000, since='2019-07-07')
```

2. The above only gives a small handful of tweets, and mainly discussion on stocks

3. But using one search term gives many more tweets. It is the same situation the other four banks. Hence, I restrict the search terms to just one

4. Multiple search terms used for other banks (and method discarded) are:

```
terms <- c("rbc", "royal bank of canada", "royalbankofcanada")
terms <- c("toronto dominion bank", "td bank", "TD bank", "TD Bank", "#tdbank", "#TDBank")
terms <- c("bmo", "bank of montreal", "bankofmontreal", "bmoharris")
terms <- c("scotiabank", "scotia bank")
```

5. Must remember to run code below periodically to make sure you don't go over the rate limit

```
RL <- getCurRateLimitInfo()
```

1.4 Data storage

Uncomment below before running the codes. For first time run, use col.names=T. The collected csv files are uploaded to github.com/chrissgtan/CSDA-1050F18S1
```{r}
### Convert into dataframe for easier analysis later
```

```
#Cibc_df <- twListToDF(Cibc)
#Rbc_df <- twListToDF(Rbc)
#Td_df <- twListToDF(Td)
#Bmo_df <- twListToDF(Bmo)
#Bns_df <- twListToDF(Bns)

### Store the dataframed tweets
#write.table(Cibc_df,"/Users/sgchr/Documents/CSDA1050/Data/Cibc.csv", append=T,
row.names=F, col.names=F,  sep=",")
#write.table(Cibc_df,"/Users/sgchr/Documents/CSDA1050/Data/AllBanks.csv", append=T,
row.names=F, col.names=F,  sep=",")

#write.table(Rbc_df,"/Users/sgchr/Documents/CSDA1050/Data/Rbc.csv", append=T,
row.names=F, col.names=F,  sep=",")
#write.table(Rbc_df,"/Users/sgchr/Documents/CSDA1050/Data/AllBanks.csv", append=T,
row.names=F, col.names=F,  sep=",")

#write.table(Td_df,"/Users/sgchr/Documents/CSDA1050/Data/Td.csv", append=T, row.names=F,
col.names=F,  sep=",")
#write.table(Td_df,"/Users/sgchr/Documents/CSDA1050/Data/AllBanks.csv", append=T,
row.names=F, col.names=F,  sep=",")

#write.table(Bmo_df,"/Users/sgchr/Documents/CSDA1050/Data/Bmo.csv", append=T,
row.names=F, col.names=F,  sep=",")
#write.table(Bmo_df,"/Users/sgchr/Documents/CSDA1050/Data/AllBanks.csv", append=T,
row.names=F, col.names=F,  sep=",")

#write.table(Bns_df,"/Users/sgchr/Documents/CSDA1050/Data/Bns.csv", append=T,
row.names=F, col.names=F,  sep=",")
#write.table(Bns_df,"/Users/sgchr/Documents/CSDA1050/Data/AllBanks.csv", append=T,
row.names=F, col.names=F,  sep=",")
```

2. Exploratory Data Analysis

Unlike spreasheets and typical databases, tweet contents are unstructured data. To aid in the analysis of unstructured data, I employ a technique to transform the tweets into structured data. In Data Science jargon, this is called the term-document matrix (or document-term matrix if we want the document to be displayed in rows). "Documents" are the tweets; and "terms" are the words in the tweets. Each element in the matrix represents the number of times a particular term appears in a particular document (the tweets).

2.1 Create the term-document matrix

```{r}
# Load the archived tweets
AllBanks_csv <- read.csv("/Users/sgchr/Documents/CSDA1050/Data/AllBanks.csv", header =
TRUE)

# Build the term-document matrix Corpus
AllBankstext <- iconv(AllBanks_csv$text, to = 'UTF-8')
corp <- Corpus(VectorSource(AllBankstext))

# Create term document matrix. Inf or infinity means to ingest everything
tdmat <- TermDocumentMatrix(corp, control = list(minWordLength=c(1,Inf)))
inspect(tdmat)
```

```
<<TermDocumentMatrix (terms: 46775, documents: 29310)>>
Non-/sparse entries: 357528/1370617722
Sparsity           : 100%
Maximal term length: 439
Weighting          : term frequency (tf)
Sample             :
                          Docs
Terms                       2079 2139 2152 344 483 5247 540 6328 6508 6512
  @arilennox:                  0    0    0   0   0    0   0    0    0    0
  <u+0001f60d>                 0    0    0   0   0    0   0    0    0    0
  bank                         0    0    0   0   0    0   0    0    0    0
  bmo                          0    0    0   1   1    1   1    1    0    1
  https://t.co/ao4jqjtgwp      0    0    0   0   0    0   0    0    0    0
  https://t.co/lfcs1xlaqn      0    0    0   0   0    0   0    0    0    0
  now!!!                       0    0    0   0   0    0   0    0    0    0
  out                          0    0    0   1   1    0   0    0    0    0
  the                          0    2    0   3   3    2   4    0    0    3
  video                        0    1    0   0   0    0   0    0    0    0
```

There are 46,775 terms in 29,310 documents (tweets). 100% sparsity means there are lots of terms occuring zero times in a document.

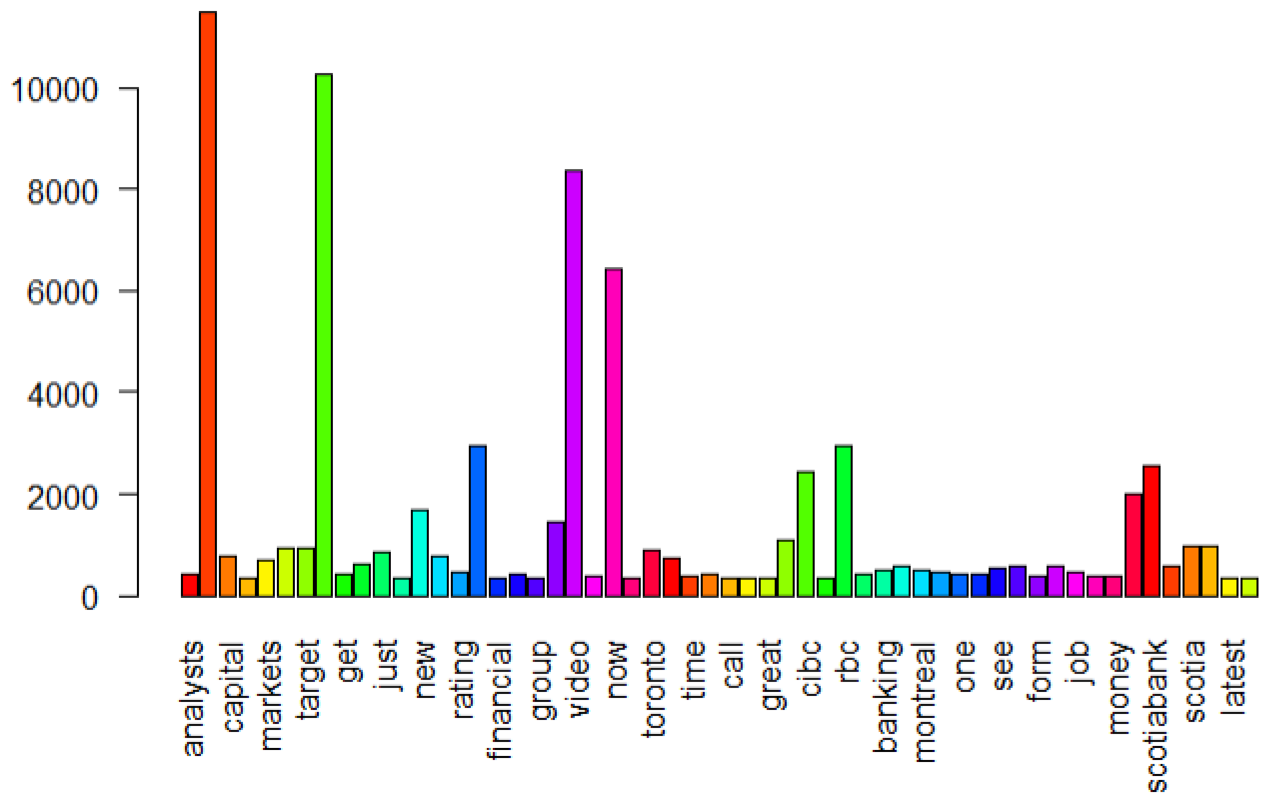
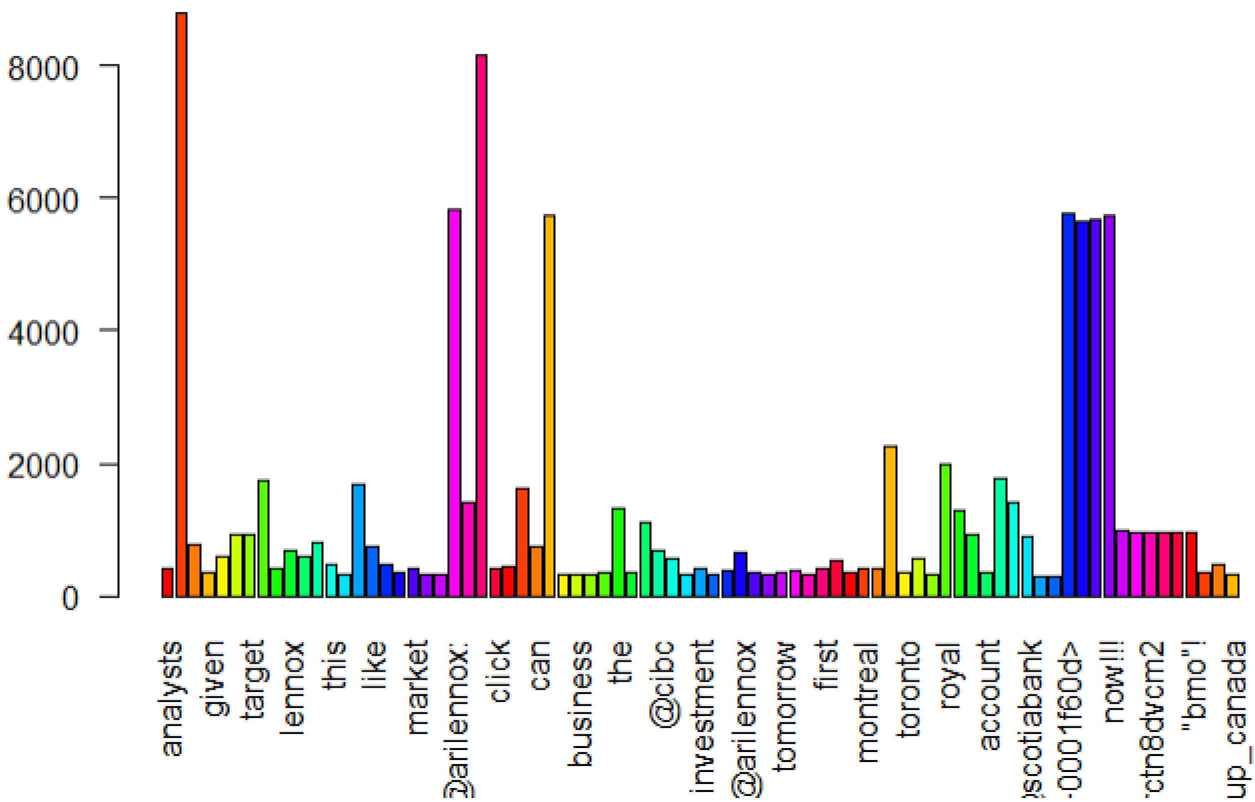
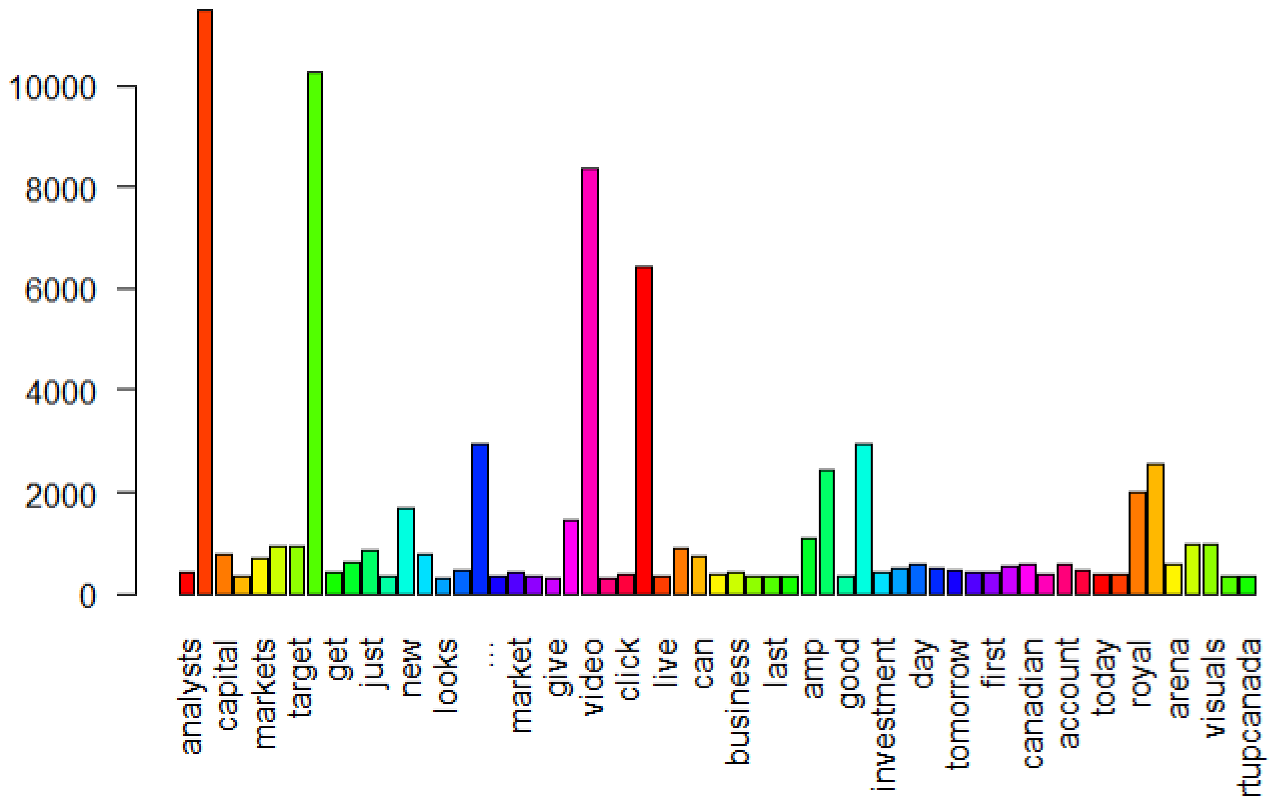
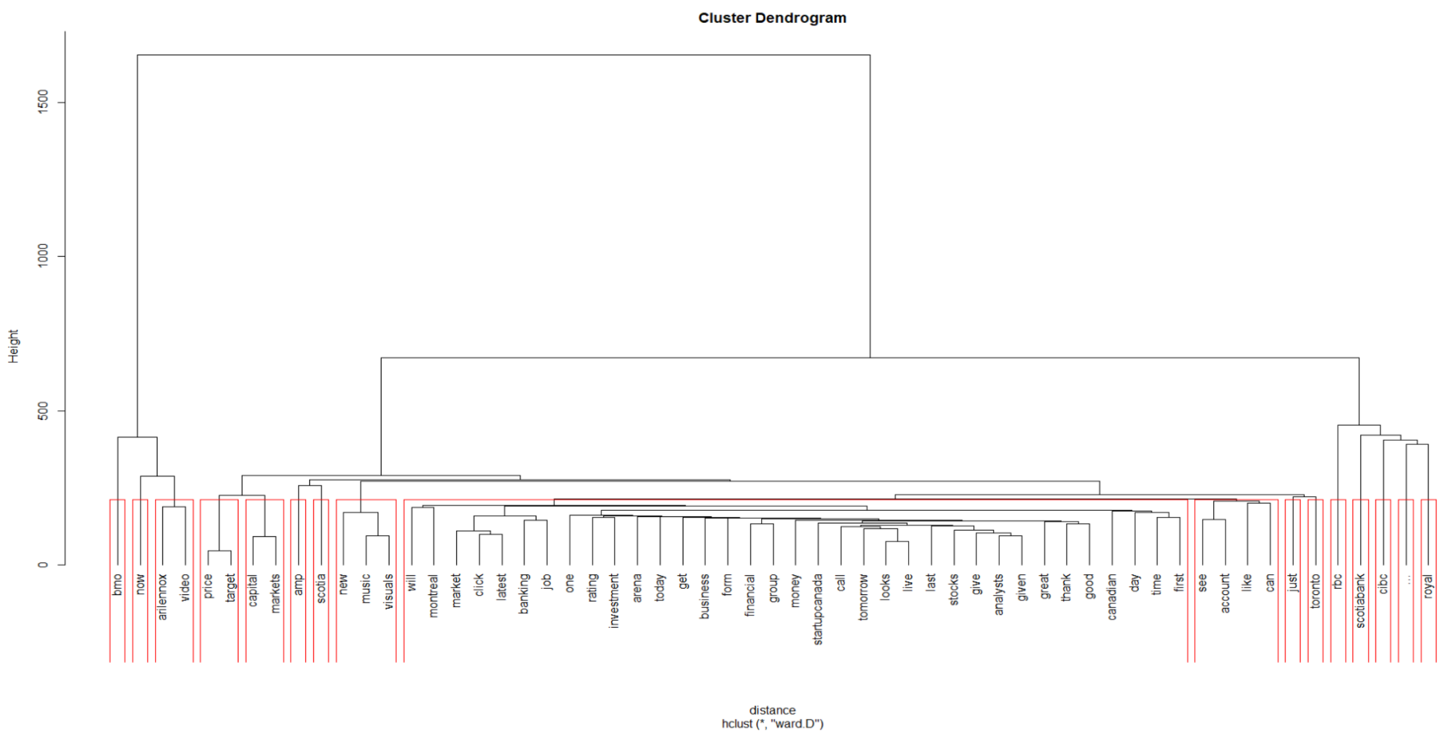
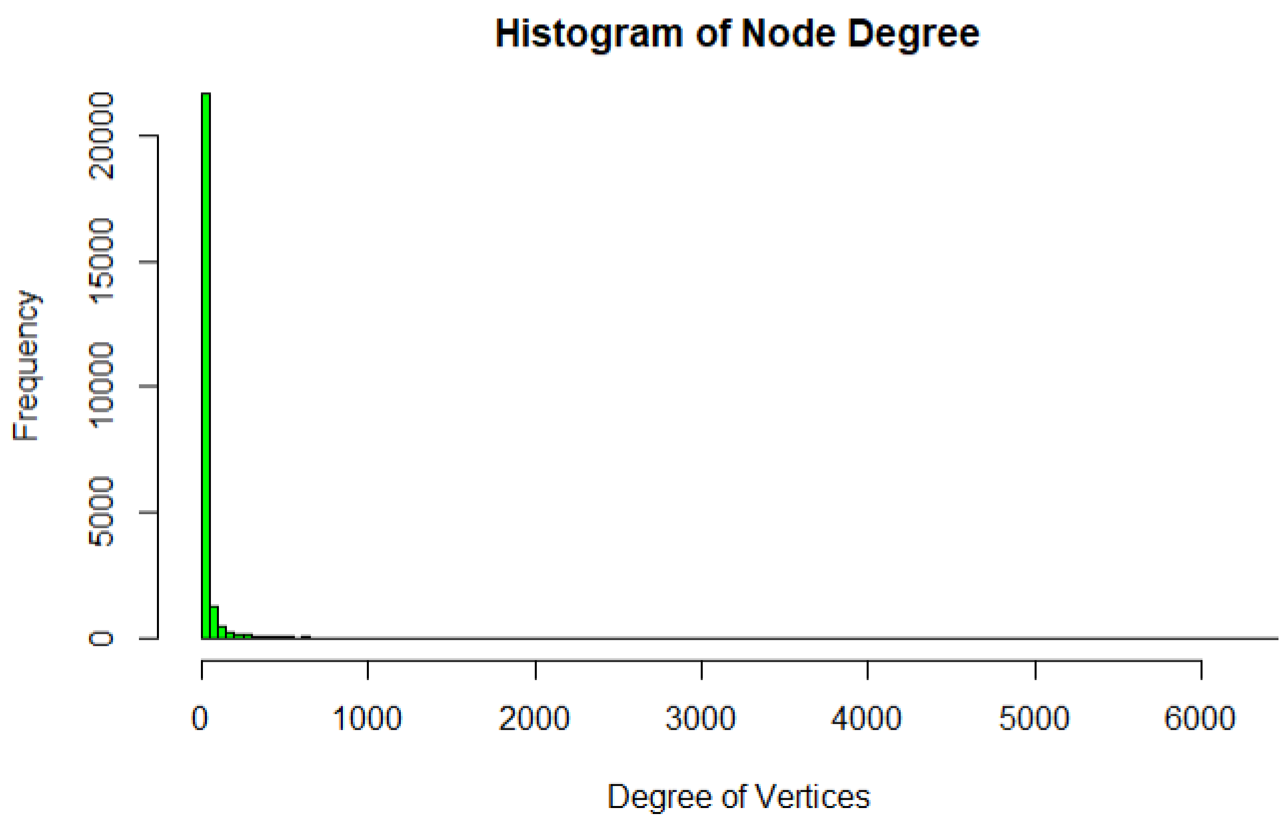
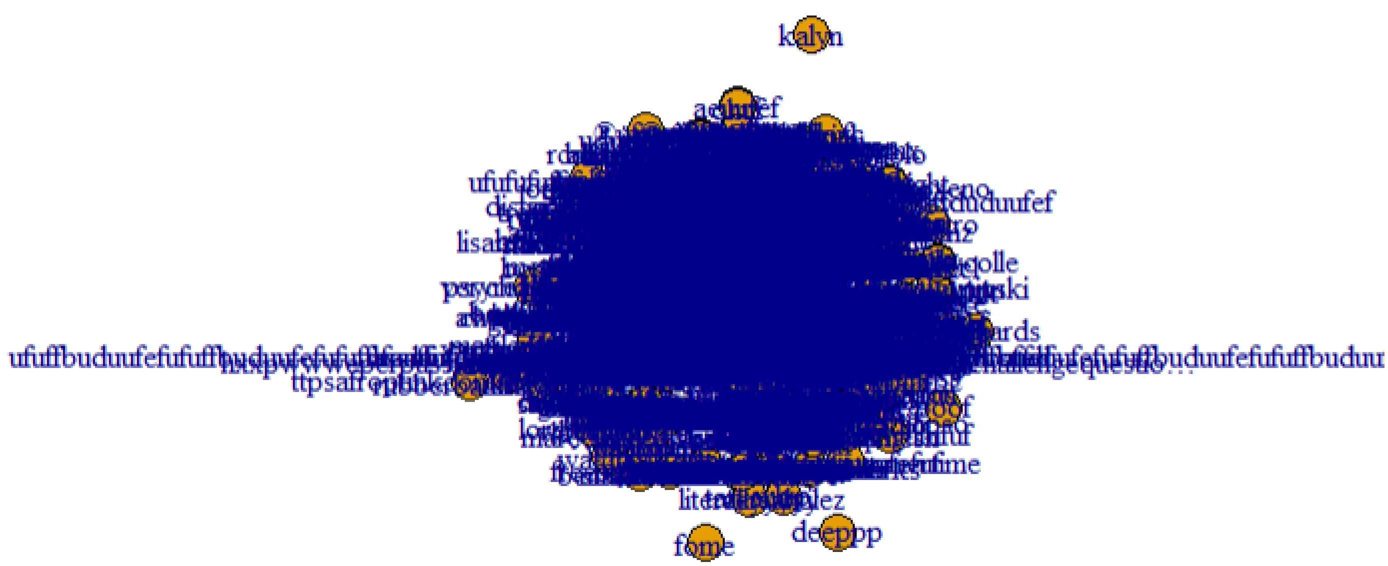
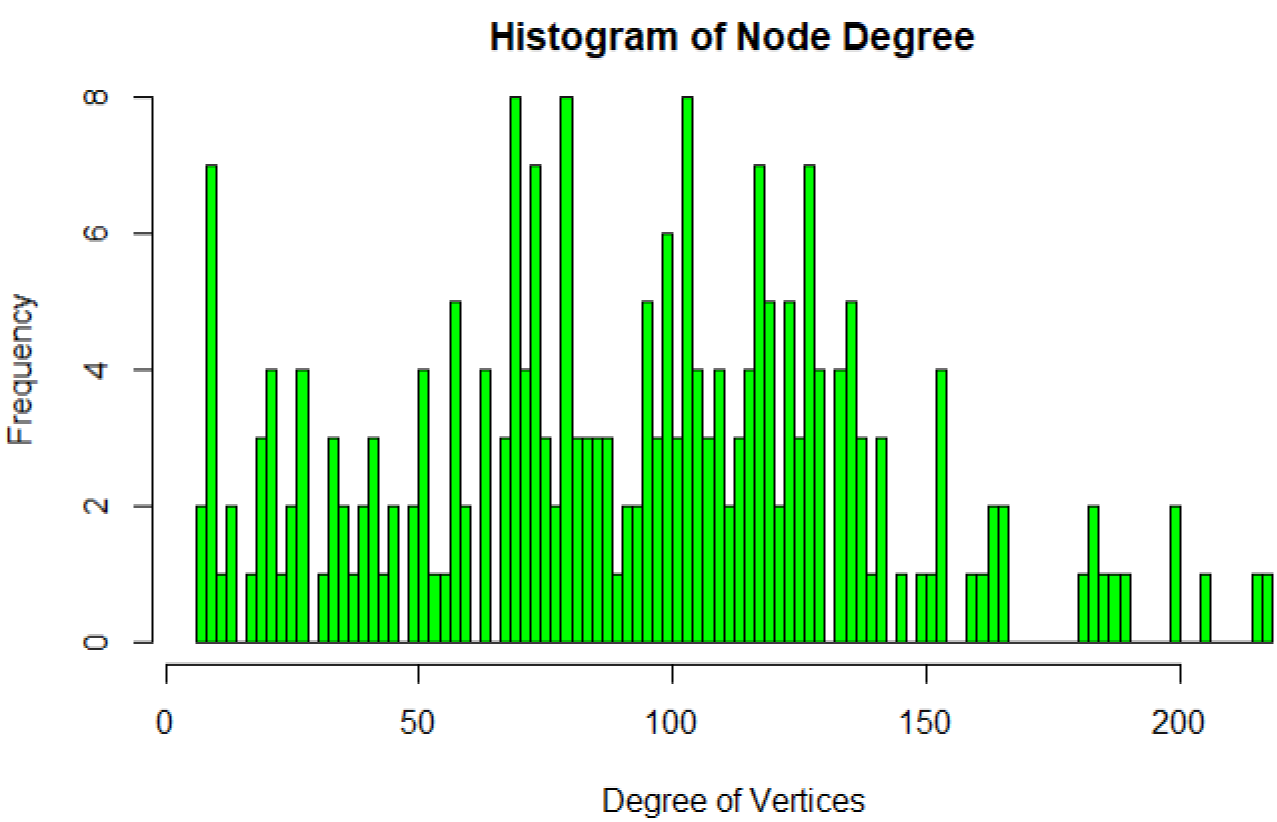
2.2 Remove terms which have at least 99% of sparse elements

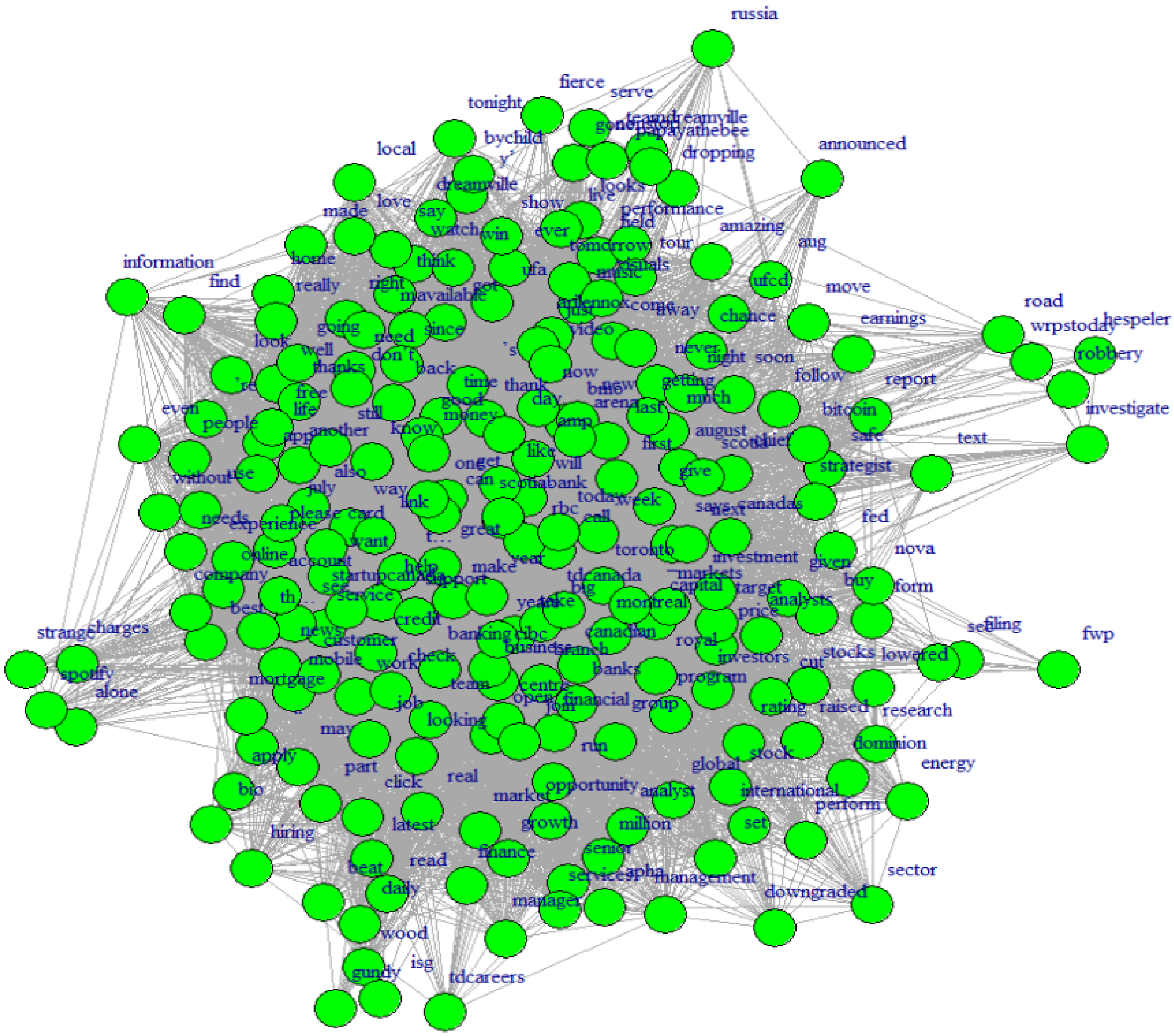
````
```{r}
tdm <- removeSparseTerms(tdmat, sparse=0.99)

# Check sparsity
inspect(tdm)
```
````

```
<<TermDocumentMatrix (terms: 107, documents: 29310)>>
Non-/sparse entries: 139832/2996338
Sparsity           : 96%
Maximal term length: 23
Weighting          : term frequency (tf)
Sample             :
                          Docs
Terms                       19577 23669 24237 344 3902 4223 483 487 556 869
  @arilennox:                   0     0     0   0    0    0   0   0   0   0
  <u+0001f60d>                  0     0     0   0    0    0   0   0   0   0
  bank                          0     1     0   0    0    0   0   0   0   0
  bmo                           0     0     0   1    1    1   1   1   1   1
  https://t.co/ao4jqjtgwp       0     0     0   0    0    0   0   0   0   0
  https://t.co/lfcs1xlaqn       0     0     0   0    0    0   0   0   0   0
  now!!!                        0     0     0   0    0    0   0   0   0   0
  out                           0     0     0   1    0    1   1   0   0   0
  the                           1     0     2   3    3    3   3   1   1   1
  video                         0     0     0   0    1    2   0   0   0   0
```

Number of terms has dropped to 107 and sparsity has dropped to 96%. We can experiment with sparse=0.xx values to make sure we have sufficient term counts for analysis. sparse=0.98 reduced the term count to 51 with 92% sparsity. To make sure terms are not unduly deleted, and there is enough terms for analysis, I opted for sparse=0.99. Not surprisingly, the term "bank" appeared at least twice in majority of the document sample

2.3 Visualise term counts (or frequency)

````
```{r}
# Convert into matrix for further analysis
mat <- as.matrix(tdm)

# Plot frequent terms
````
````

```
freq <- rowSums(mat) # Count number of times each of the 107 terms appears

# It will be a very busy chart if we plot all 107 terms, so we restrict any terms that
appears more > 320 times. You can experiment with freq > ?? to make sure you can see all
the text in the bar plot. This is so that you could determine what "junk" terms (eg.,
"&amp:") you could eliminate
freq <- subset(freq, freq>320)

# Visualise
barplot(freq,
        las=2, # list all words in rows and display text vertically
        col = rainbow(25))
```



There are many stop words that can be removed

2.4 Data processing and normalisation

2.4.1 Remove stop words

```{r}
corp <- tm_map(corp, removeWords, stopwords(kind="en"))
tdmat <- TermDocumentMatrix(corp, control = list(minWordLength=c(1,Inf)))
inspect(tdmat)
```

```
transformation drops documents<<TermDocumentMatrix (terms: 46316, documents: 29310)>>
Non-/sparse entries: 308769/1357213191
Sparsity           : 100%
Maximal term length: 436
Weighting          : term frequency (tf)
Sample             :
                         Docs
Terms                     11461 15479 15509 15516 16749 2079 2152 6328 6344 6508
  @arilennox:                 0     0     0     0     0    0    0    0    0    0
  <u+0001f60d>                0     0     0     0     0    0    0    0    0    0
  bank                        0     0     0     0     0    0    0    0    0    0
  bmo                         0     0     0     0     0    0    0    1    0    0
  canada                      0     0     0     0     0    0    0    0    0    0
  https://t.co/ao4jqjtgwp     0     0     0     0     0    0    0    0    0    0
  https://t.co/lfcs1xlaqn     0     0     0     0     0    0    0    0    0    0
  now!!!                      0     0     0     0     0    0    0    0    0    0
  royal                       0     0     0     0     0    0    0    0    0    0
  video                       0     0     0     0     0    0    0    0    0    0
```

Term count reduced from 46,775 to 46,316 (459 stop words have been removed). But sparsity is still 100%

2.4.2 Remove terms which have at least 99% of sparse elements

```{r}
tdm <- removeSparseTerms(tdmat, sparse=0.99)

# Check sparsity
inspect(tdm)
```

```
<<TermDocumentMatrix (terms: 79, documents: 29310)>>
Non-/sparse entries: 98971/2216519
Sparsity           : 96%
Maximal term length: 23
Weighting          : term frequency (tf)
Sample             :
                         Docs
Terms                     2365 2373 2375 2377 2383 2394 2424 2431 2444 6328
  @arilennox:                0    0    0    0    0    0    0    0    0    0
  <u+0001f60d>               0    0    0    0    0    0    0    0    0    0
  bank                       0    0    0    0    0    0    0    0    0    0
  bmo                        0    0    0    0    0    0    0    0    0    1
  canada                     0    0    0    0    0    0    0    0    0    0
  https://t.co/ao4jqjtgwp    0    0    0    0    0    0    0    0    0    0
  https://t.co/lfcs1xlaqn    0    0    0    0    0    0    0    0    0    0
  now!!!                     0    0    0    0    0    0    0    0    0    0
  royal                      0    0    0    0    0    0    0    0    0    0
  video                      1    1    1    1    1    1    1    1    1    0
```

Term count has reduced to 79 (and with stop words also removed via the preceding code)

2.4.3 Visualise term counts (or frequency)

```{r}
# Convert into matrix for further analysis
mat <- as.matrix(tdm)

# Plot frequent terms
freq <- rowSums(mat) # Count number of times each of the 79 terms appears

# It will be a very busy chart if we plot all 79 terms, so we restrict any terms that
appears more > 70 times. Can experiment with freq>?? to make sure you can see all the
```

~~~
text in the bar plot. This is so that you could determine what "junk" terms (eg.,
"@amp:") you could eliminate
freq <- subset(freq, freq>70)

# Visualise
barplot(freq,
        las=2, # list all words vertically
        col = rainbow(25))
```
~~~



It is evident that more text cleaning are still needed. We will iterate steps 2.5 and 2.6 to buid the stop and replacement words

2.5 Additional data processing and normalisation

~~~
```{r}
corp <- tm_map(corp, tolower) # Not crucial for text analytics but good practice to do so
corp <- tm_map(corp, removePunctuation) # Remove punctuations
corp <- tm_map(corp, removeNumbers) # Remove numbers

# Remove URL
removeURL <- function(x) gsub('http[[:alnum:]]*', '', x)
corp <- tm_map(corp, content_transformer(removeURL))

# Remove words. Since we are analysing "banks" in "Canada", words like "bank" and
"canada" do not add value to our analysis
MyStopwords <- c(stopwords(kind="en"), "bank", "canada", "…", "’s", "ufd", "via",
"mefeater")
corp <- tm_map(corp, removeWords, MyStopwords)

# Replace words
corp <- tm_map(corp, gsub, pattern = 'lennox', replacement = 'arilennox')
corp <- tm_map(corp, gsub, pattern = '“bmo”', replacement = 'bmo')
corp <- tm_map(corp, gsub, pattern = 'lennox’s', replacement = 'arilennox')
corp <- tm_map(corp, gsub, pattern = 'ariarilennox', replacement = 'arilennox')
corp <- tm_map(corp, gsub, pattern = 'ari', replacement = 'arilennox')
~~~

```
corp <- tm_map(corp, gsub, pattern = 'arilennoxlennox', replacement = 'arilennox')

corp <- tm_map(corp, stripWhitespace) # remove leftover from the preceding removal

# Repeat the preceding codes
tdmat <- TermDocumentMatrix(corp, control = list(minWordLength=c(1,Inf)))
tdm <- removeSparseTerms(tdmat, sparse=0.99)
mat <- as.matrix(tdm)
freq <- rowSums(mat)
freq <- subset(freq, freq>=60)
barplot(freq,
        las=2, # list all words vertically
        col = rainbow(25))
```



The text is mostly cleaned up, and the barplot is evidently cleaner

2.6 Explore clustering of terms/tweets using k-means method
This analysis will uncover the following two insights:

1. Within cluster sum of squares by cluster: We want this to be low, which means the elements within each cluster are close to each other

2. Between_SS / total_SS: We want this to be high, that is, distances between clusters are further apart

I experimented with k = 9 to 18 to find best mix of distance within and between clusters and deemed the best is k=17

In section 2.7, we will use k=17 to visualise the term-clusters using a cluster dendrogram plot

```{r}
set.seed(123)
```

```
m1 <- t(mat) # Transpose
k <- 17 # I experimented with k = 9 to 18 to find best mix of distance within and between clusters and deemed the best is k=17
kc <- kmeans(m1, k)
kc
```

```
K-means clustering with 17 clusters of sizes 6290, 2243, 6523, 898, 236, 1630, 1121, 440, 401, 2015,
235, 490, 2048, 1636, 2530, 238, 336

Cluster means:
     analysts         bmo      capital        given      markets        price       target
arilennox         get         will         just        stocks          new          like        looks
rating          …
1  0.000000000 0.9993640700 0.0000000000 0.0000000000 0.000000000 0.0000000000 0.0000000000
1.000317965 0.000317965 0.0001589825 0.006995231 0.0000000000 0.000317965 0.0004769475 0.003338633
0.0000000000 0.016375199
2  0.008916630 0.0584039233 0.0414623272 0.0008916630 0.023183237 0.0013374944 0.0013374944
0.000000000 0.010254124 0.0022291574 0.020062416 0.0040124833 0.027641551 0.0133749443 0.001337494
0.0022291574 0.007133304
3  0.002146252 0.0000000000 0.0045991108 0.0024528591 0.002912770 0.0027594665 0.0003066074
0.002912770 0.022229036 0.0000000000 0.038019316 0.0056722367 0.027134754 0.0406254791 0.002146252
0.0016863406 0.000000000
4  0.357461024 0.3184855234 0.3229398664 0.3151447661 0.308463252 1.0077951002 1.0044543430
0.000000000 0.000000000 0.0000000000 0.001113586 0.1714922049 0.065701559 0.0000000000 0.002227171
0.0022271715 0.007795100
5  0.000000000 0.0042372881 0.0000000000 0.0000000000 0.000000000 0.0000000000 0.0000000000
0.000000000 0.008474576 0.0000000000 0.008474576 0.0000000000 0.004237288 0.5805084746 0.000000000
0.0000000000 0.021186441
6  0.007975460 0.0055214724 0.0263803681 0.0030674847 0.022699387 0.0042944785 0.0042944785
0.000000000 0.031901840 0.0276073620 0.022699387 0.0134969325 0.021472393 0.0263803681 0.003067485
0.0693251534 0.010429448
7  0.018733274 0.0000000000 0.0017841213 0.0223015165 0.008920607 0.0044603033 0.0000000000
0.000000000 0.010704728 0.0000000000 0.005352364 0.0713648528 0.016057092 0.0062444246 0.000000000
0.2256913470 0.003568243


 [ reached getOption("max.print") -- omitted 28310 entries ]

Within cluster sum of squares by cluster:
 [1]  813.0590 2207.6416 5773.9703 1817.4276  402.7754 1600.9620  821.6753  682.0909  449.0324
2589.6754  216.1277  783.1878 3185.4604 1364.4756 3335.7700  370.5924  552.9702
 (between_SS / total_SS =  61.4 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"    "size"
"iter"         "ifault"
```

Observations:

1. There are 17 clusters of sizes 6290, 2243, 6523, etc

2. "Cluster means" shows the average of each term being analysed. High average means that word has appeared in the particular cluster with higher frequency.

3. "Clustering vector" shows which cluster each term has gone to.

4. "Within cluster sum of squares by cluster" shows the distance between terms within each cluster. We want this to be low.

5. "between_SS / total_SS" shows the distance between clusters. We want this to be high

I have experimented with k = 9 to 18 and deemed the best mix of distance within and between clusters is k=17

```
k   Within clusters   Between clusters
```

```
=   ==============     =================
9   3,844              50.5
10  3,590              48.6
11  2,997              52.8
12  2,802              51.9
13  2,529              53.0
14  2,390              52.1
15  2,127              54.3
16  1,991              54.4
17  1,586 <-           61.4 <-
18  1,791              53.9
```

2.7 Plot the hierarchical clusters

```{r}
# Utilise dendrogram for hierarchical clustering of terms in tweets
# Find the distance, use scale to normalise the matrix
distance <- dist(scale(mat))

# Print the terms, and calculate the distance between the words in each document (tweets)
print(distance, digits = 2)

hc <- hclust(distance, method = "ward.D")
plot(hc, hang=-1)
rect.hclust(hc, k=17) # 17 clusters per findings in step 2.6
```



Observations:

1. Notice the banks are clustered together, which is logical. But there is a single cluster "scotia". This can be easily corrected by replacing "scotia" with "scotiabank"

2. Price and target are clustered together due to discussions about stock prices

2.8 Find the pair of terms that appears frequently together

```{r}
# Term document matrix to convert unstructure text into structured for easier analysis
tdmat <- TermDocumentMatrix(corp)
```

```
tdmat <- as.matrix(tdmat)
tdmat[1:30,1:30] # See the first 30 terms in the first 30 documents (tweets)
```

```
                Docs
Terms            1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
  analysts       1 0 0 0 0 0 0 1 0  1  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  1  1  1
  arcc           1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  ares           1 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  bmo            1 1 1 1 1 1 1 1 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
  capital        2 0 0 1 0 1 1 1 0  1  0  0  0  1  1  0  1  1  1  0  1  0  0  1  1  1  1  1  1  1
  given          1 0 0 0 0 1 1 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1
  markets        1 0 0 1 0 2 1 1 0  1  0  0  0  0  1  0  1  1  1  0  1  0  0  1  1  1  1  1  1  1
  price          1 0 0 1 0 1 1 0 0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  1  0  0  1  1  1
  target         1 0 0 1 0 1 1 0 0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  1  0  0  1  1  1
  arilennox      0 2 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  get            0 1 0 0 0 0 0 0 1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  hype           0 1 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  never          0 1 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  will           0 1 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  amiller        0 0 1 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  another        0 0 1 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  city           0 0 2 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  fan            0 0 1 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  field          0 0 1 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  just           0 0 1 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  manofbird      0 0 1 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  owned          0 0 1 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  share          0 0 1 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  tfc            0 0 1 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  toro…          0 0 1 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  upset          0 0 1 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  aks            0 0 0 1 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  reiterates     0 0 0 1 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0
  steel          0 0 0 1 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  bygoneboatmen  0 0 0 0 1 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

Above show how many times each term appears in each tweet. Example "city" appears twice in tweet 3

```{r}
# Network of terms
tdmat[tdmat>1] <- 1 # Convert matrix into binary, that is whether a term appears (1) or not (0)

tdmat[1:25,1:25]
```

2.8.1 Pair of terms that appears frequently together

```{r}
# Create term-term matrix
termM <- tdmat %*% t(tdmat) # Multiply tdmat and transpose of tdmat
termM[1:25,1:25]
```

Terms	analysts	arcc	ares	bmo	capital	given	markets	price	target	arilennox	get	hype	never	will	amiller	another	city	fan	field	just	manofbird	owned	share	tfc	toro…
analysts	425	3	3	133	117	124	121	322	322	0	3	0	1	2	0	0	0	0	0	1	0	0	0	0	0
arcc	3	5	5	5	5	2	5	4	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ares	3	5	5	5	5	2	5	4	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
bmo	133	5	5	11369	555	127	512	286	287	8428	62	3	19	48	3	24	9	16	158	387	4	2	7	6	1
capital	117	5	5	555	773	113	588	283	287	0	0	0	0	6	0	3	0	0	0	4	0	0	2	0	0
given	124	2	2	127	113	353	117	283	283	8	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
markets	121	5	5	512	588	117	699	273	275	0	0	0	0	26	0	2	0	0	0	3	0	0	1	0	0
price	322	4	4	286	283	283	273	940	898	4	2	0	0	0	0	0	0	0	0	1	0	0	5	0	0
target	322	4	4	287	287	283	275	898	927	0	0	0	0	2	0	0	0	0	0	3	0	0	2	0	0
arilennox	0	0	0	8428	0	8	0	4	0	8544	9	3	4	2	0	0	0	4	0	253	0	0	0	0	0
get	3	0	0	62	0	0	0	2	0	9	412	1	5	20	1	2	1	1	6	16	12	1	0	2	0
hype	0	0	0	3	0	0	0	0	0	3	1	6	1	1	0	0	0	0	0	0	0	0	0	0	0
never	1	0	0	19	0	0	0	0	0	4	5	1	218	17	0	0	1	0	1	8	0	0	0	1	0
will	2	0	0	48	6	0	26	0	2	2	20	1	17	572	1	9	11	1	5	6	1	0	24	1	0
amiller	0	0	0	3	0	0	0	0	0	0	1	0	0	1	4	1	3	1	3	2	4	2	1	2	1
another	0	0	0	24	3	0	2	0	0	0	2	0	0	9	1	166	1	1	1	5	1	1	1	1	1
city	0	0	0	9	0	0	0	0	0	0	1	0	1	11	3	1	73	1	5	2	3	2	2	2	1
fan	0	0	0	16	0	0	0	0	0	4	1	0	0	1	1	1	1	77	3	4	1	1	1	1	1
field	0	0	0	158	0	0	0	0	0	0	6	0	1	5	3	1	5	3	182	62	3	2	1	2	1
just	1	0	0	387	4	1	3	1	3	253	16	0	8	6	2	5	2	4	62	860	2	1	1	1	1
manofbird	0	0	0	4	0	0	0	0	0	0	12	0	0	1	4	1	3	1	3	2	17	2	1	2	1
owned	0	0	0	2	0	0	0	0	0	0	1	0	0	0	2	1	2	1	2	1	2	5	1	2	1
share	0	0	0	7	2	0	1	5	2	0	0	0	0	24	1	1	2	1	1	1	1	1	78	1	1
tfc	0	0	0	6	0	0	0	0	0	0	2	0	1	1	2	1	2	1	2	1	2	2	1	12	1
toro…	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	7

The above shows pair of terms that appears frequently together. For example: tfc and bmo appear together in 6 tweets. This is not surprising, since bmo is a major sponsor of the Toronto Football Club (tfc)

2.8.2 Find the network of terms

````
```{r}
g <- graph.adjacency(termM, weighted = T, mode = 'undirected')
g
```
````

```
IGRAPH 9f0c209 UNW- 24277 411973 --
+ attr: name (v/c), weight (e/n)
+ edges from 9f0c209 (vertex names):
 [1] analysts--analysts    analysts--arcc        analysts--ares        analysts--bmo         analysts--capital     analysts--given       analysts--markets     analysts--price
 [9] analysts--target      analysts--get         analysts--never       analysts--will        analysts--just        analysts--aks         analysts--steel       analysts--well
[17] analysts--primo       analysts--prmw        analysts--water       analysts--stocks      analysts--coverage    analysts--new         analysts--okta        analysts--receives
[25] analysts--earnings    analysts--level       analysts--right       analysts--like        analysts--fuel        analysts--growth      analysts--rating      analysts--realty
[33] analysts--stock       analysts--trust       analysts--…           analysts--financial   analysts--initiated   analysts--zscaler     analysts--t…          analysts--can't
[41] analysts--market      analysts--mastercard  analysts--gild        analysts--gilead      analysts--sciences    analysts--set         analysts--group       analysts--international
[49] analysts--connections analysts--give        analysts--waste       analysts--wcn         analysts--now         analysts--increased   analysts--mcd         analysts--mcdonald
[57] analysts--amgen       analysts--amgn        analysts--street      analysts--know        analysts--near        analysts--won't       analysts--actually    analysts--national
+ ... omitted several edges
```

````
```{r}
g <- simplify(g) # To prevent looping of same terms, example, pairing of same terms like "analysts--analysts"
V(g)$label <- V(g)$name # Labels for the terms
V(g)$degree <- degree(g) # How often each term appears
```
````

````
```{r}
# Histogram of node degree
hist(V(g)$degree,
     breaks = 100, # how many bars
     col = 'green',
     main = 'Histogram of Node Degree',
     ylab = 'Frequency',
     xlab = 'Degree of Vertices')
```
````



Above is a right skewed histogram and most terms appears less than 400 times

````
```{r}
# Network diagram
plot(g)
```
````



Above graph is too busy. One method is to reduce the size of the matrix

````
```{r}
tdmat <- tdmat[rowSums(tdmat)>120,] # rowSums counts the total frequency; that is, keep terms that appear > 120 times

# Re-run earlier code
````
````

```
tdmat[tdmat>1] <- 1
termM <- tdmat %*% t(tdmat)
termM[1:10,1:10]
g <- graph.adjacency(termM, weighted = T, mode = 'undirected')
g
g <- simplify(g) # To prevent looping of same terms
V(g)$label <- V(g)$name
V(g)$degree <- degree(g)
```

```
           Terms
Terms       analysts   bmo capital given markets price target arilennox get never
  analysts       425   133     117   124     121   322    322         0   3     1
  bmo            133 11369     555   127     512   286    287      8428  62    19
  capital        117   555     773   113     588   283    287         0   0     0
  given          124   127     113   353     117   283    283         8   0     0
  markets        121   512     588   117     699   273    275         0   0     0
  price          322   286     283   283     273   940    898         4   2     0
  target         322   287     287   283     275   898    927         0   0     0
  arilennox        0  8428       0     8       0     4      0      8544   9     4
  get              3    62       0     0       0     2      0         9 412     5
  never            1    19       0     0       0     0      0         4   5   218
IGRAPH ab08a1d UNW- 231 10884 --
+ attr: name (v/c), weight (e/n)
+ edges from ab08a1d (vertex names):
 [1] analysts--analysts     analysts--bmo          analysts--capital       analysts--given        analysts--markets      analysts--price        analysts--target       analysts--get
 [9] analysts--never        analysts--will         analysts--just          analysts--well         analysts--stocks       analysts--new          analysts--earnings     analysts--right
[17] analysts--like         analysts--growth       analysts--rating        analysts--stock        analysts--…            analysts--financial    analysts--t…           analysts--market
[25] analysts--set          analysts--group        analysts--international analysts--give         analysts--now          analysts--know         analysts--week         analysts--last
[33] analysts--great        analysts--watch        analysts--amp           analysts--banks        analysts--best         analysts--cibc         analysts--rbc          analysts--report
[41] analysts--investment   analysts--banking      analysts--apha          analysts--montreal     analysts---            analysts--way          analysts--perform      analysts--'m
[49] analysts--don't        analysts--life         analysts--still         analysts--cut          analysts--much         analysts--buy          analysts--global       analysts--first
[57] analysts--say          analysts--see          analysts--research      analysts--canadian     analysts--downgraded   analysts--th…          analysts--million      analysts--energy
+ ... omitted several edges
```

```{r}
# Histogram of node degree
hist(V(g)$degree,
     breaks = 100, # how many bars
     col = 'green',
     main = 'Histogram of Node Degree',
     ylab = 'Frequency',
     xlab = 'Degree of Vertices')
```



Note the histogram is less busy and more evenly spreadout

```{r}
```

````
# Network diagram
plot(g)
plot(g,
     vertex.color='green',
     vertex.size = 8, # can experiment with this
     vertex.label.dist = 1.5)
```
````



Much less busy than earlier. There are obvious discussions around branch robbery, russia, CIBC Wood Gundy and careers at TD Bank

2.9 Word cloud

````
```{r}
w <- sort(rowSums(tdmat), decreasing = TRUE)
wordcloud(words = names(w),
          freq = w,
          max.words = 150,
          random.order = F,
          min.freq = 5,
          colors = brewer.pal(8, 'Dark2'),
          scale = c(5, 0.3),
````

```
          rot.per = 0.7)
```

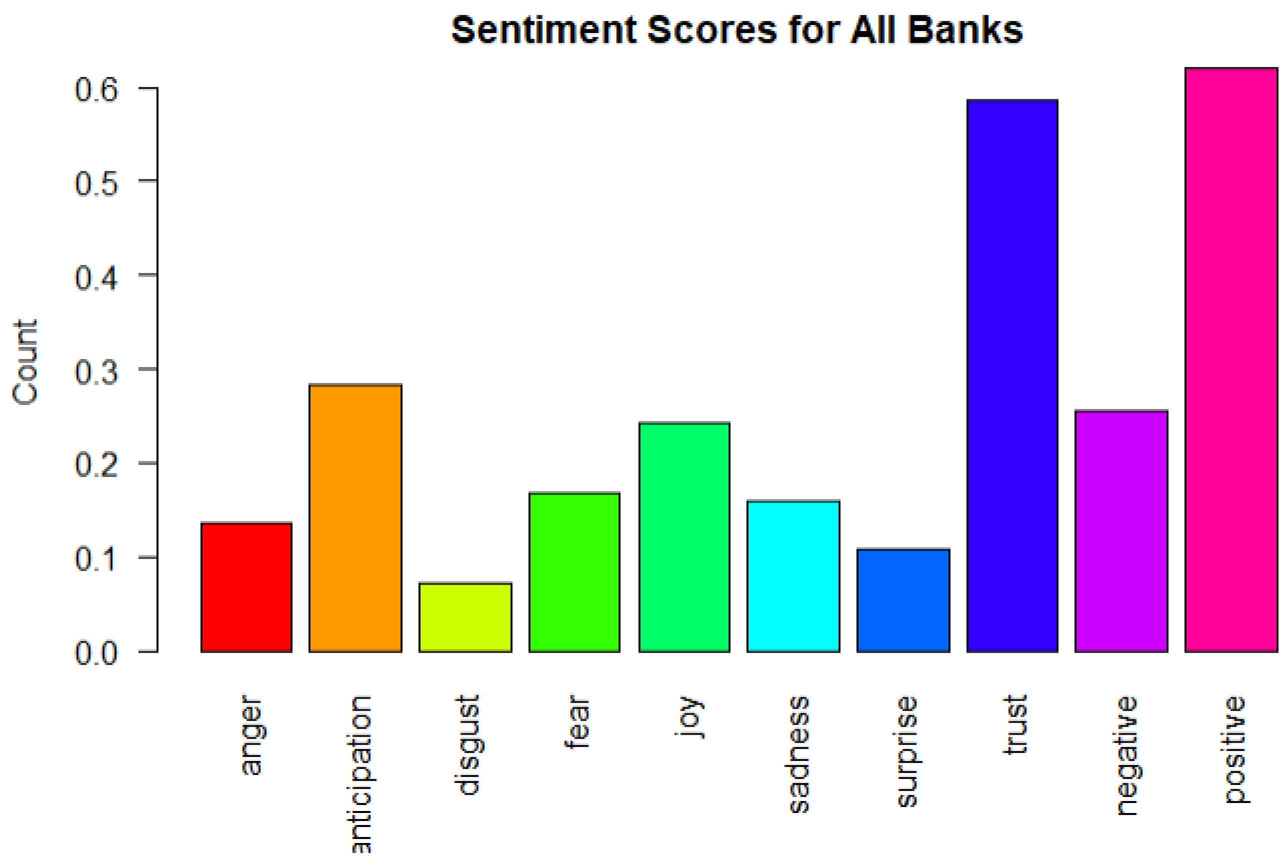
If Ari Lennox indeed is the professional singer and songwriter, there is an odd discussion about this artiste in banking tweets. Maybe this artiste performed at the Scotiabank arena

2.10 Sentiment analysis

```{r}
sen <- get_nrc_sentiment(AllBankstext)

# Archive the sentiment scores for further analysis later
Allbanks_sen_df <- t(as.data.frame(colMeans(sen)))
write.table(Allbanks_sen_df,"/Users/sgchr/Documents/CSDA1050/Data/Allbanks_sen_df.csv", 
append=T, row.names=F, col.names=T,  sep=",")

# So that we could compare with the individual banks, we will normalise the data by 
applying the statistical mean
barplot(colMeans(sen),
        las = 2,
        col = rainbow(10),
        ylab = 'Count',
        main = 'Sentiment Scores for All Banks')
```



Observations:

1. The sentiment is generally positive for all banks

2. There is also a lot of trust in the banks

3. "Disgust" is the lowest sentiment

3. Social Media Analytics for Canadian Banks

Based on the exploratory data analysis for all banks in the preceding sections, we are now ready to do the social media analytics for each of the five Canadian Banks. The outline is given below

3.1 Create and compact the term-document matrix (TDM) for each bank
3.2 Further compact the term-document matrix
3.3 Clustering of terms/tweets k-means method
3.4 Plot the hierarchical clusters
3.5 Find the pair of terms that appears frequently together
  3.5.1 Find the network of terms
  3.5.2 Find optimal term count for optimal (neither too busy nor sparse) visualisation of network diagram
  3.5.3 Plot the network of terms diagram
3.6 Word cloud
3.7 Sentiment analysis

CIBC

3.1a Create and compact the term-document matrix (TDM) for each bank

````
```{r}
# Load the archived tweets
CIBC_csv <- read.csv("/Users/sgchr/Documents/CSDA1050/Data/Cibc.csv", header = TRUE)

# Build the initial term-document matrix Corpus
CIBCtext <- iconv(CIBC_csv$text, to = 'UTF-8')
CIBCcorp <- Corpus(VectorSource(CIBCtext))

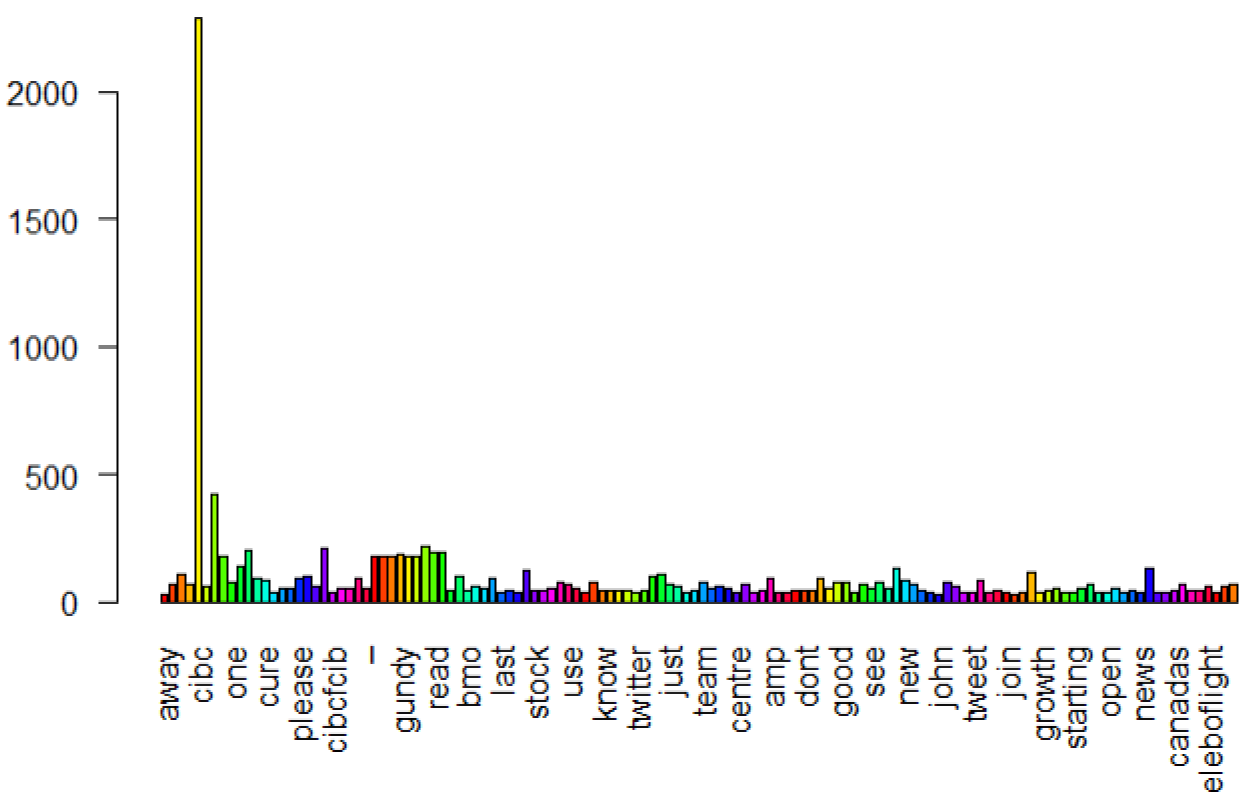
# Compact the Corpus
CIBCcorp <- tm_map(CIBCcorp, tolower) # Not crucial for text analytics but good practice to do so
CIBCcorp <- tm_map(CIBCcorp, removePunctuation) # Remove punctuations
CIBCcorp <- tm_map(CIBCcorp, removeNumbers) # Remove numbers

# Remove URL
removeURL <- function(x) gsub('http[[:alnum:]]*', '', x)
CIBCcorp <- tm_map(CIBCcorp, content_transformer(removeURL))

CIBCcorp <- tm_map(CIBCcorp, removeWords, stopwords(kind="en")) # Remove "standard" stop words

CIBCcorp <- tm_map(CIBCcorp, stripWhitespace) # remove leftover from the preceding removal

# Create the term document matrix
CIBCtdmat <- TermDocumentMatrix(CIBCcorp, control = list(minWordLength=c(1,Inf)))
CIBCtdm <- removeSparseTerms(CIBCtdmat, sparse=0.99)
CIBCmat <- as.matrix(CIBCtdm)
CIBCfreq <- rowSums(CIBCmat)
CIBCfreq <- subset(CIBCfreq, CIBCfreq>=30) # Experiment with CIBCfreq=? to find the optimal of of terms in the barplot
barplot(CIBCfreq,
        las=2, # list all words vertically
        col = rainbow(25))
```
````



The above plot shows we need remove and replace some terms. For example, since we are analysing banks, therefore term = "bank" is redundant. Likewise, this analysis is speacifically about CIBC, therefore "cibc" is redundant. Also, otherwise, "cibc" will unduly skew the wordcloud, network of terms analysis

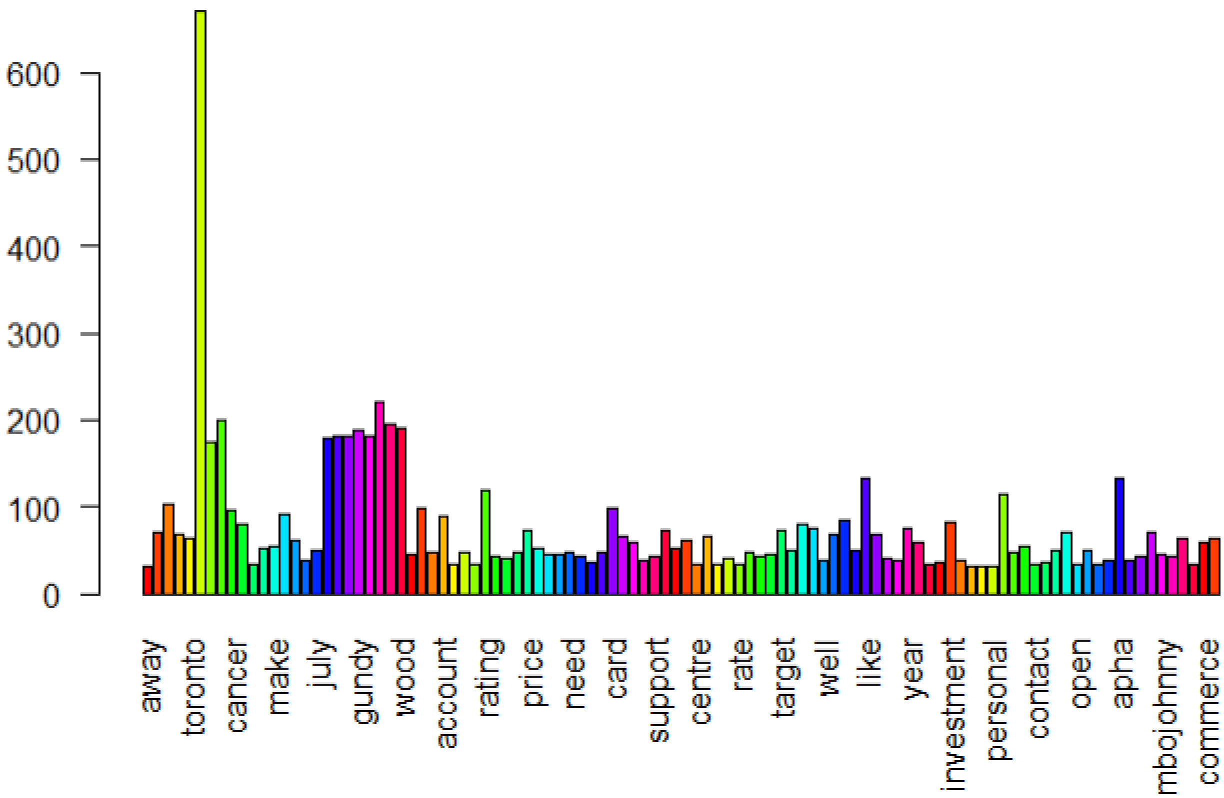
3.2a Further compact the term-document matrix

````
```{r}
MyStopwords <- c("’s", "...", "bank", "b…", "cibc", "cibcs")
CIBCcorp <- tm_map(CIBCcorp, removeWords, MyStopwords)

CIBCcorp <- tm_map(CIBCcorp, gsub, pattern = 'aphria', replacement = 'aphriainc') #
Replace words

CIBCcorp <- tm_map(CIBCcorp, stripWhitespace) # remove leftover from the preceding
removal

# Repeat the preceding codes
CIBCtdmat <- TermDocumentMatrix(CIBCcorp, control = list(minWordLength=c(1,Inf)))
CIBCtdm <- removeSparseTerms(CIBCtdmat, sparse=0.99)
CIBCmat <- as.matrix(CIBCtdm)
CIBCfreq <- rowSums(CIBCmat)
CIBCfreq <- subset(CIBCfreq, CIBCfreq>=30) # Experiment with CIBCfreq=? to find the
optimal of of terms in the barplot
barplot(CIBCfreq,
        las=2, # list all words vertically
        col = rainbow(25))
```
````



The text is mostly cleaned up

3.3a Clustering of terms/tweets k-means method

This analysis will uncover the following two insights:

1. **Within cluster sum of squares by cluster**: We want this to be low, which means the elements within each cluster are close to each other

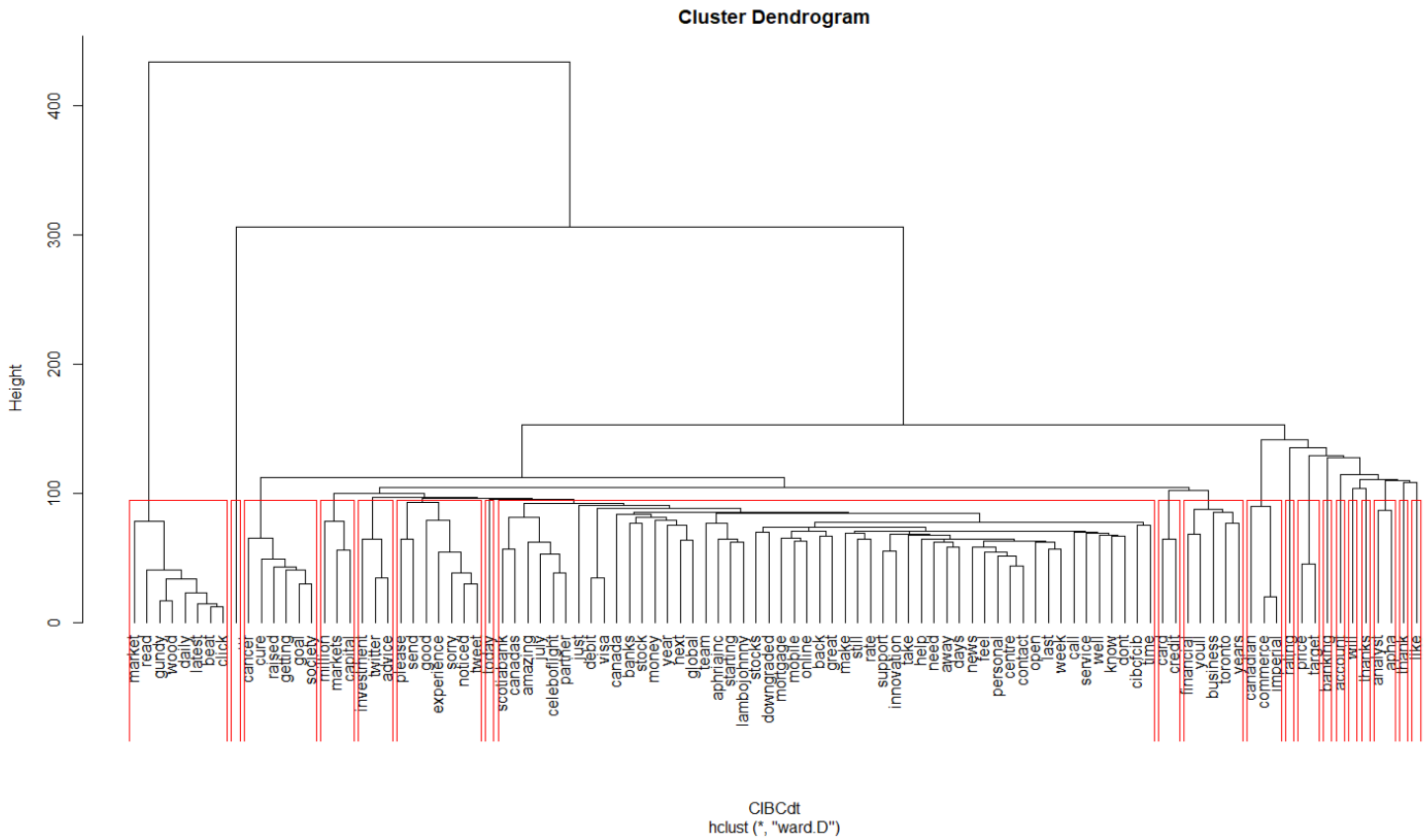
2. **Between_SS / total_SS**: We want this to be high, that is, distances between clusters are further apart

I experimented with k = 10-22 to find best mix of distance within and between clusters and deemed the best is k=20

In section 3.4a, we will use k=20 to visualise the term-clusters using a cluster dendrogram plot

```
```{r}
set.seed(123)

CIBCm1 <- t(CIBCmat) # Transpose CIBCmat
CIBCk <- 20
CIBCkc <- kmeans(CIBCm1, CIBCk)
CIBCkc
```
```

```
Within cluster sum of squares by cluster:
 [1]   59.69231  721.75636   71.96429  233.39806  108.00000  114.61111  121.79592   53.18519
20.76923   38.60000  121.78082   91.01587   77.13043    0.00000  104.51282   60.44444 2148.51050
32.88889  104.68000
[20]   38.43478
 (between_SS / total_SS =  44.3 %)
```

Observations:

Experimentation for k = 10 to 22 implied the best mix of distance within and between clusters for the current CIBC dataset is k=21

```
k   Within clusters   Between clusters
=   ===============   ================
10  488               21.6
11  558               20.0
12  507               23.9
13  366               24.4
14  324               40.3
15  377               24.6
16  285               25.6
17  340               25.7
18  240               27.6
19  229               43.6
20  209               44.3 <-
21  195               29.2
22  190               30.7
```

3.4a Plot the hierarchical clusters

```
```{r}
# Find the distance, use scale to normalise the matrix
CIBCdt <- dist(scale(CIBCmat))

CIBChc <- hclust(CIBCdt, method = "ward.D")
plot(CIBChc, hang=-1)
rect.hclust(CIBChc, k=20) # 20 clusters based on analysis in section 3.3a
```
```



Not surprisingly, the terms "cancer", "cure", "raised", "goal" are clustered together since CIBC is a majore sponsor for "Run For The Cure" charity

3.5a Find the pair of terms that appears frequently together

```{r}
# Term document matrix to convert unstructure text into structured for easier analysis
CIBCtdmat <- TermDocumentMatrix(CIBCcorp)
CIBCtdmat <- as.matrix(CIBCtdmat)

# Convert matrix into binary, that is whether a term appears (1) or not (0)
CIBCtdmat[CIBCtdmat>1] <- 1

# Create term-term matrix
CIBCttm <- CIBCtdmat %*% t(CIBCtdmat) # Multiply CIBCtdmat and transpose of CIBCtdmat
```

3.5.1a Find the network of terms

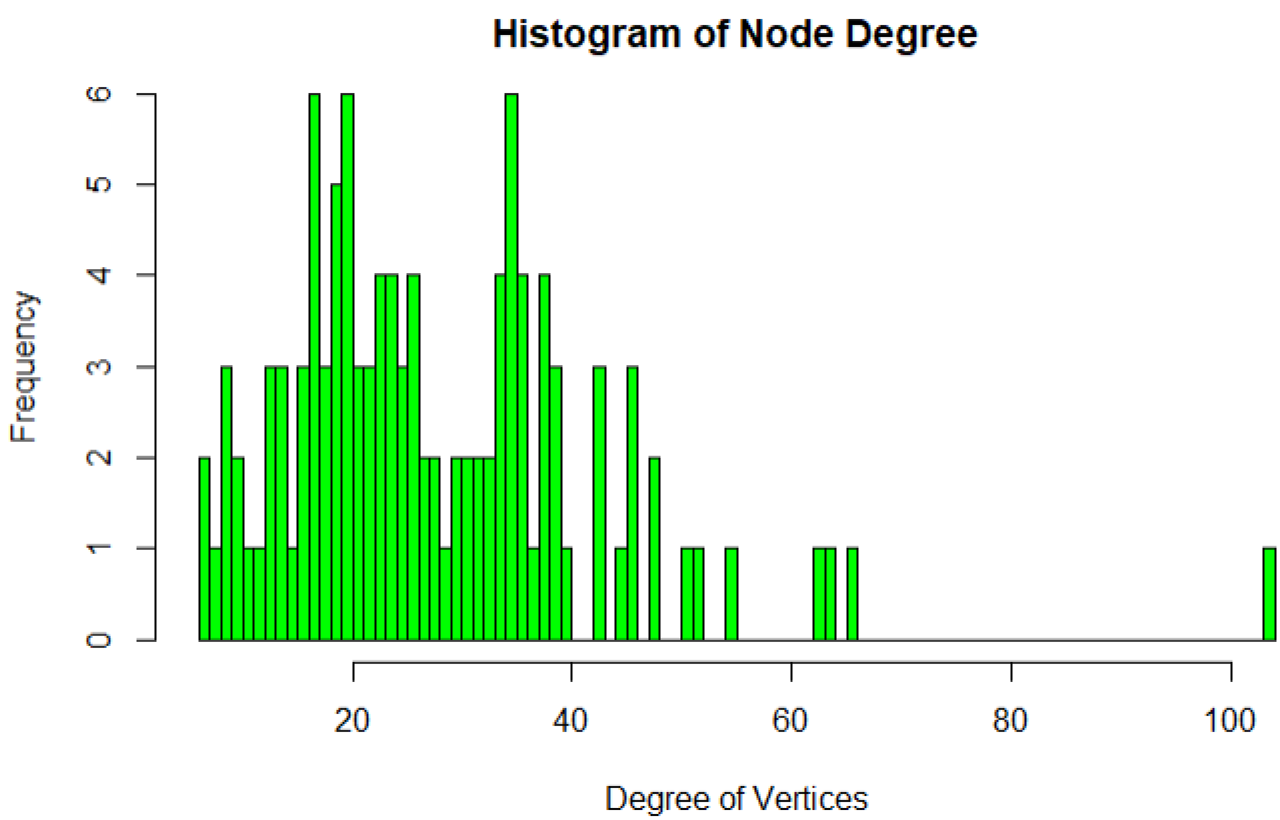
```{r}
CIBCg <- graph.adjacency(CIBCttm, weighted = T, mode = 'undirected')
CIBCg <- simplify(CIBCg) # To prevent looping of same terms
V(CIBCg)$label <- V(CIBCg)$name # Labels for the terms
V(CIBCg)$degree <- degree(CIBCg) # How often each term appears, or number of connections
between terms
CIBCg
```

```
IGRAPH ded7853 UNW- 5749 56458 --
+ attr: name (v/c), label (v/c), degree (v/n), weight (e/n)
+ edges from ded7853 (vertex names):
 [1] awardwinning--away          awardwinning--broadway      awardwinning--broadwaychicago awardwinning--b…            awardwinning--chicago       awardwinning--director      awardwinning--honor
 [8] awardwinning--passed        awardwinning--producer      awardwinning--today         awardwinning--tony          awardwinning--will          awardwinning--today…        away        --broadway
[15] away        --broadwaychicago away        --b…          away        --chicago     away        --director    away        --honor       away        --passed      away        --producer
[22] away        --today       away        --tony        away        --will        away        --food        away        --today…      away        --gundy       away        --wood
[29] away        --thing       away        --goes        away        --thank       away        --house       away        --difficult   away        --need        away        --canada
[36] away        --card        away        --just        away        --rewarding   away        --stopped     away        --times       away        --working…    away        --month
[43] away        --right       away        --much        away        --builders    away        --cant        away        --create…     away        --enough      away        --helped
[50] away        --sponsors    away        --thecx       away        --money       away        --youre       away        --like        away        --bootsotground away        --days
+ ... omitted several edges
```

3.5.2a Find optimal term count for optimal (neither too busy nor sparse) visualisation of network diagram

```{r}
```

```
# Reset CIBCtdmat if needed to experiment with optimal rowSums(CIBCtdmat)>?
# The larger ? is, the more visible the network of terms is, but less granular
CIBCtdmat <- TermDocumentMatrix(CIBCcorp)
CIBCtdmat <- as.matrix(CIBCtdmat)

# rowSums counts the total frequency; that is, keep terms that appear > 30 times
CIBCtdmat <- CIBCtdmat[rowSums(CIBCtdmat)>30,]

# Re-run earlier code
CIBCtdmat[CIBCtdmat>1] <- 1 # Whenvevr CIBCtdmat is > 1, assign value of 1
CIBCtermM <- CIBCtdmat %*% t(CIBCtdmat)
#CIBCtermM[1:10,1:10]
CIBCg <- graph.adjacency(CIBCtermM, weighted = T, mode = 'undirected')
CIBCg <- simplify(CIBCg) # To prevent looping of same terms
V(CIBCg)$label <- V(CIBCg)$name
V(CIBCg)$degree <- degree(CIBCg)

# Histogram of node degree
hist(V(CIBCg)$degree,
     breaks = 100, # how many bars
     col = 'green',
     main = 'Histogram of Node Degree',
     ylab = 'Frequency',
     xlab = 'Degree of Vertices')
```

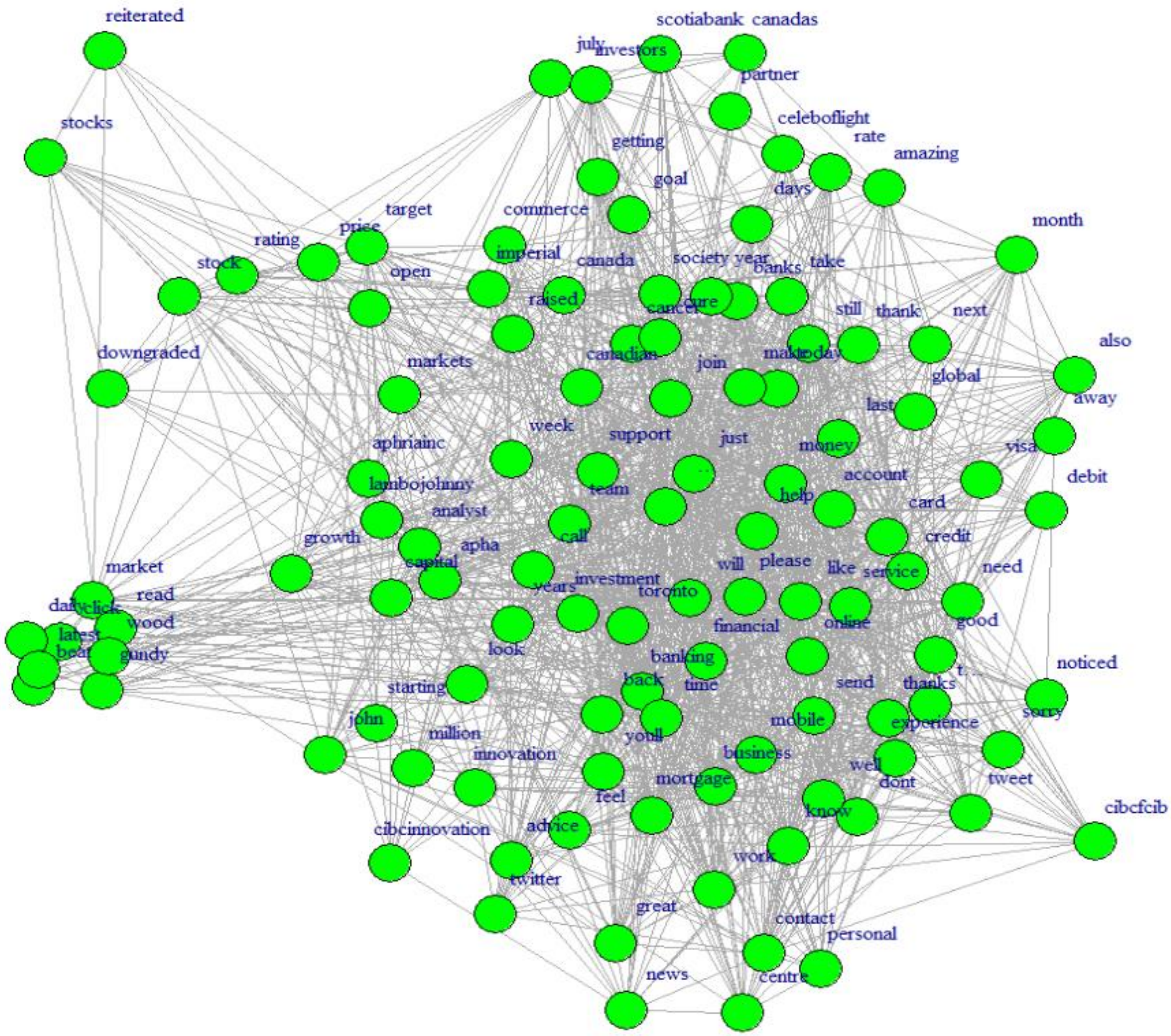


Most terms appears less than 70 times

3.5.3a Plot the network of terms diagram

```{r}
# To experiment with rowSums(CIBCtdmat)>? in 3.5.2 if there is no apparent clusters or if
diagram is too busy / sparse
plot(CIBCg)
plot(CIBCg,
```

```
     vertex.color='green',
     vertex.size = 8, # can experiment with this
     vertex.label.dist = 1.5)
```



An apparent network appears surrounding "Wood Gundy", "Growth" and "Innovation"

3.6a Word cloud

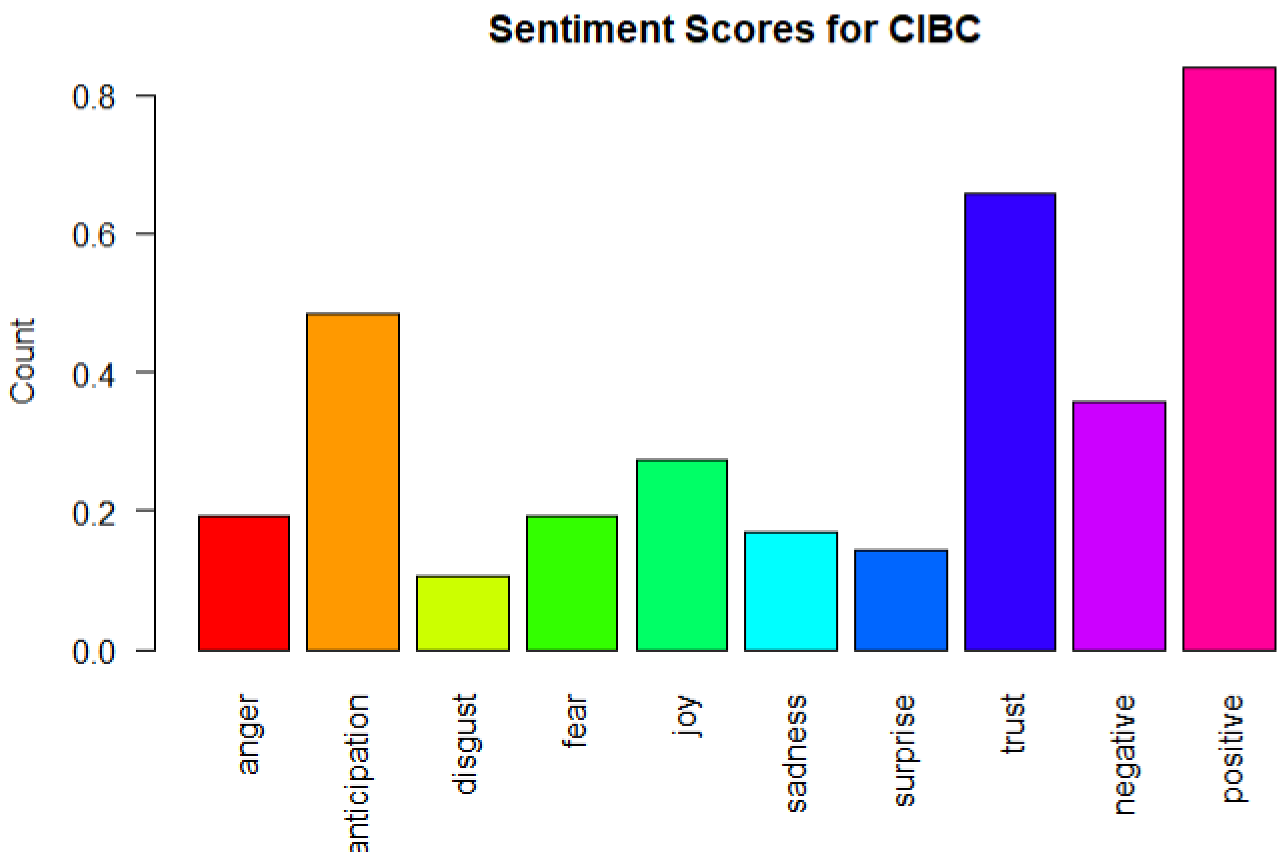
```{r}
CIBCtdmat <- TermDocumentMatrix(CIBCcorp)
CIBCtdmat <- as.matrix(CIBCtdmat)

CIBCw <- sort(rowSums(CIBCtdmat), decreasing = TRUE)
wordcloud(words = names(CIBCw),
          freq = CIBCw,
          max.words = 150,
          random.order = F,
          min.freq = 5,
          colors = brewer.pal(8, 'Dark2'),
          scale = c(5, 0.3),
```

```
        rot.per = 0.7)
```

Consistent with previous analysis, lots of discussion about stock investments, some postive words are obvious: "like", "thank"

3.7a Sentiment analysis

```{r}
CIBCsen <- get_nrc_sentiment(CIBCtext)

# Archive the sentiment scores for further analysis later
CIBC_sen_df <- t(as.data.frame(colMeans(CIBCsen)))
write.table(CIBC_sen_df,"/Users/sgchr/Documents/CSDA1050/Data/Allbanks_sen_df.csv",
append=T, row.names=F, col.names=F,  sep=",")

# So that we could compare with the individual banks, we will normalise the data by
applying the statistical mean
barplot(colMeans(CIBCsen),
        las = 2,
        col = rainbow(10),
        ylab = 'Count',
        main = 'Sentiment Scores for CIBC')
```

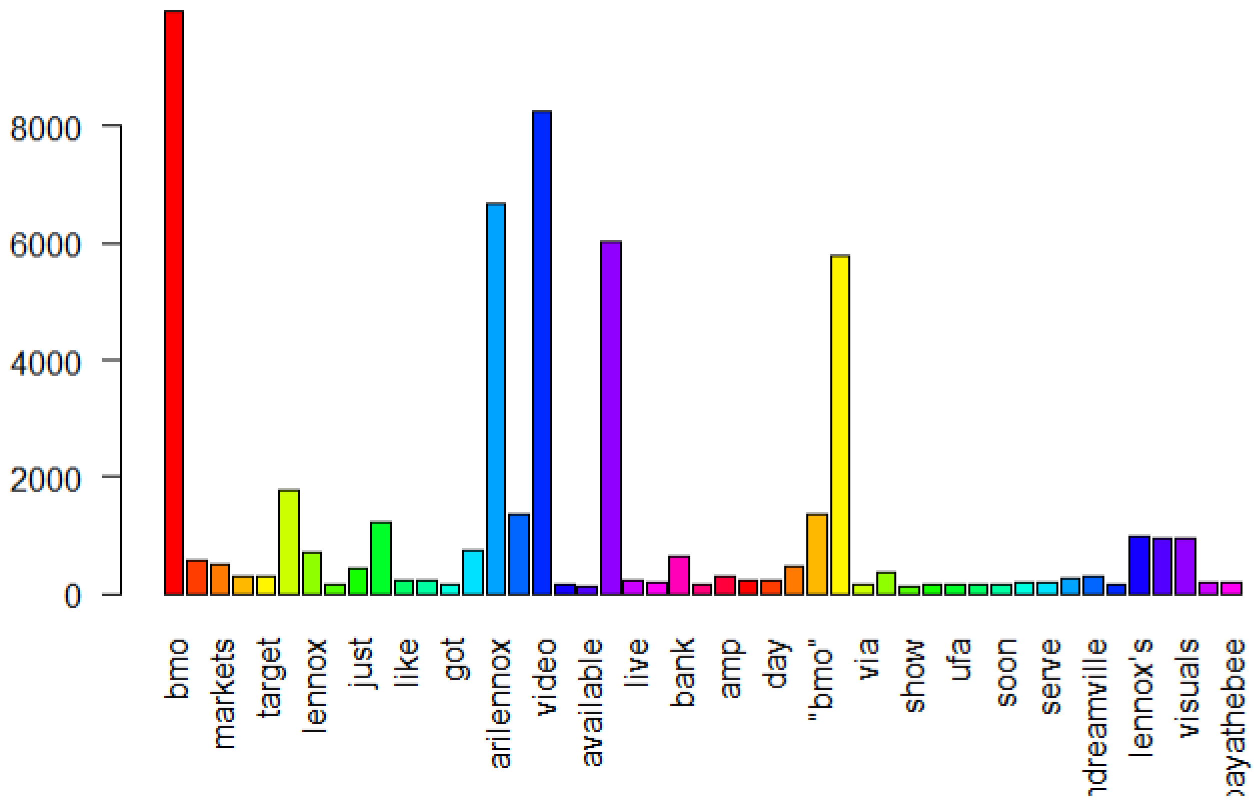


BMO

3.1b Create and compact the term-document matrix (TDM) for each bank

```{r}
# Load the archived tweets
BMO_csv <- read.csv("/Users/sgchr/Documents/CSDA1050/Data/Bmo.csv", header = TRUE)

# Build the initial term-document matrix Corpus
BMOtext <- iconv(BMO_csv$text, to = 'UTF-8')
BMOcorp <- Corpus(VectorSource(BMOtext))

# Compact the Corpus
BMOcorp <- tm_map(BMOcorp, tolower) # Not crucial for text analytics but good practice to do so
BMOcorp <- tm_map(BMOcorp, removePunctuation) # Remove punctuations
BMOcorp <- tm_map(BMOcorp, removeNumbers) # Remove numbers

# Remove URL
removeURL <- function(x) gsub('http[[:alnum:]]*', '', x)
BMOcorp <- tm_map(BMOcorp, content_transformer(removeURL))

BMOcorp <- tm_map(BMOcorp, removeWords, stopwords(kind="en")) # Remove "standard" stop words

BMOcorp <- tm_map(BMOcorp, stripWhitespace) # remove leftover from the preceding removal

# Create the term document matrix
BMOtdmat <- TermDocumentMatrix(BMOcorp, control = list(minWordLength=c(1,Inf)))
BMOtdm <- removeSparseTerms(BMOtdmat, sparse=0.99)
BMOmat <- as.matrix(BMOtdm)
BMOfreq <- rowSums(BMOmat)
```

```
BMOfreq <- subset(BMOfreq, BMOfreq>=50) # Experiment with BMOfreq=? to find the optimal
of of terms in the barplot
barplot(BMOfreq,
        las=2, # list all words vertically
        col = rainbow(25))
```

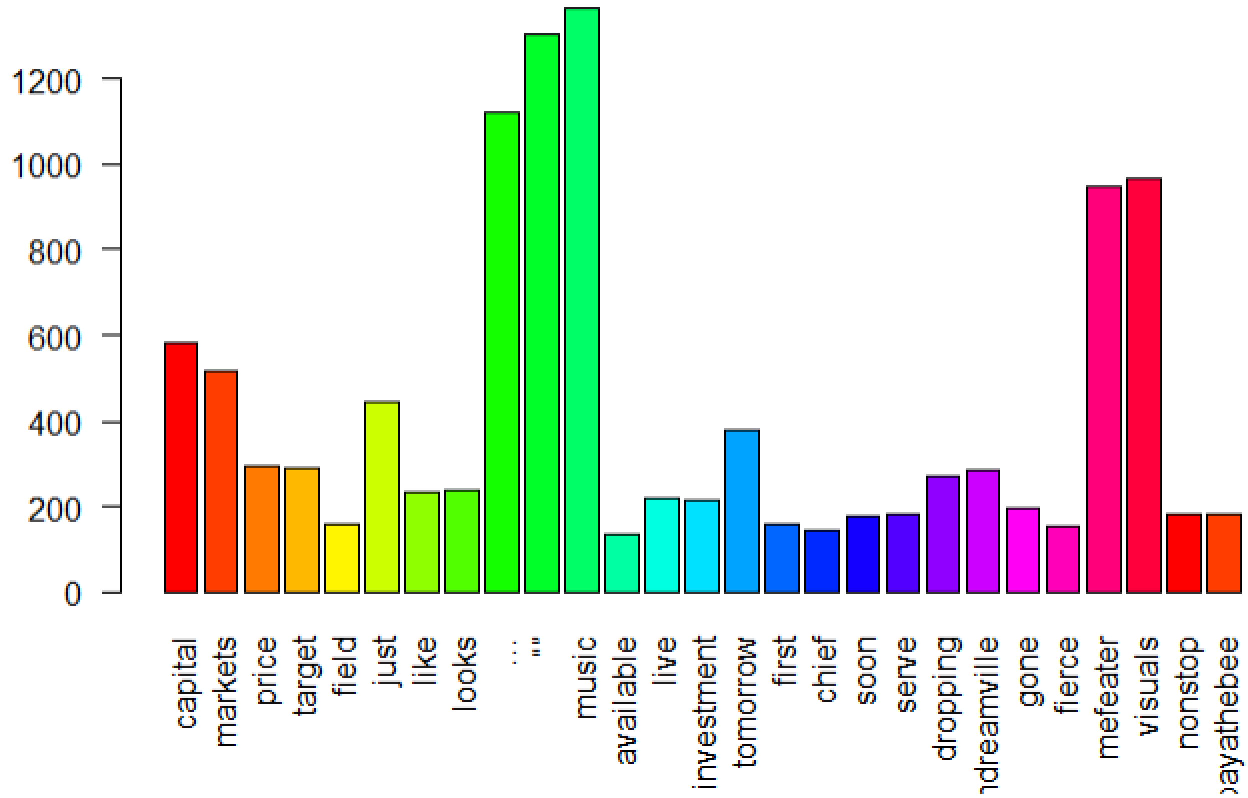


The above plot shows we need remove and replace some terms. For example, since we are analysing banks, therefore term = "bank" is redundant. Likewise, this analysis is speacifically about BMO, therefore "bmo" is redundant. Also, otherwise, "bmo" will unduly skew the wordcloud, network of terms analysis

3.2b Further compact the term-document matrix

```{r}
MyStopwords <- c("...", "bank", "bmo", "href", "montreal", "“bmo”", "arilennox", "ari",
"lennox", "video", "lennox's", "“”")
BMOcorp <- tm_map(BMOcorp, removeWords, MyStopwords)

BMOcorp <- tm_map(BMOcorp, gsub, pattern = 'wages”', replacement = 'wages') # Replace
words
BMOcorp <- tm_map(BMOcorp, gsub, pattern = '“perhaps', replacement = 'perhaps')
BMOcorp <- tm_map(BMOcorp, gsub, pattern = '"hold”', replacement = 'hold')

BMOcorp <- tm_map(BMOcorp, stripWhitespace) # remove leftover from the preceding removal

# Repeat the preceding codes
BMOtdmat <- TermDocumentMatrix(BMOcorp, control = list(minWordLength=c(1,Inf)))
BMOtdm <- removeSparseTerms(BMOtdmat, sparse=0.99)
BMOmat <- as.matrix(BMOtdm)
BMOfreq <- rowSums(BMOmat)
BMOfreq <- subset(BMOfreq, BMOfreq>=30) # Experiment with BMOfreq=? to find the optimal
of of terms in the barplot
barplot(BMOfreq,
        las=2, # list all words vertically
        col = rainbow(25))
```

```
```



The text is mostly cleaned up

3.3b Clustering of terms/tweets k-means method

This analysis will uncover the following two insights:

1. Within cluster sum of squares by cluster: We want this to be low, which means the elements within each cluster are close to each other

2. Between_SS / total_SS: We want this to be high, that is, distances between clusters are further apart

I experimented with k = 10-12 to find best mix of distance within and between clusters and deemed the best is k=11

In section 3.4b, we will use k=11 to visualise the term-clusters using a cluster dendrogram plot

```{r}
set.seed(123)

BMOm1 <- t(BMOmat) # Transpose BMOmat
BMOk <- 11
BMOkc <- kmeans(BMOm1, BMOk)
BMOkc
```

```
Within cluster sum of squares by cluster:
 [1]   46.918919    0.000000    7.989474   13.984375    0.000000   28.411765  318.951983  559.523005   68.058394    2.980916 1082.091026
 (between_SS / total_SS =  80.7 %)
```

Observations:

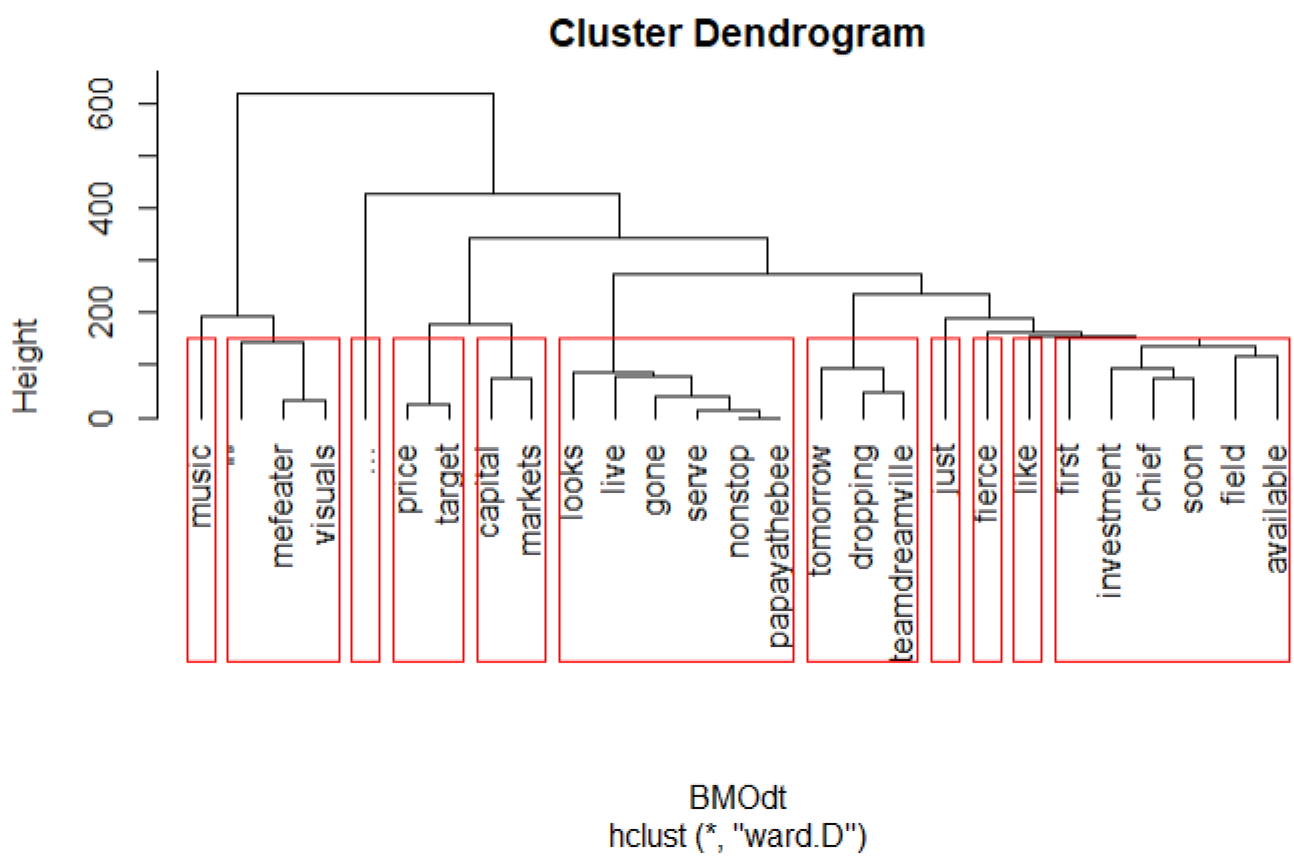
Experimentation for k = 10 to 12 implied the best mix of distance within and between clusters for the current CIBC dataset is k=11

```
k   Within clusters   Between clusters
=   ===============   ================
10  627               67.4
11  424               80.7 <-
12  711               55.6
```

3.4b Plot the hierarchical clusters

~~~
```{r}
# Find the distance, use scale to normalise the matrix
BMOdt <- dist(scale(BMOmat))

BMOhc <- hclust(BMOdt, method = "ward.D")
plot(BMOhc, hang=-1)
rect.hclust(BMOhc, k=11) # 11 clusters
```
~~~



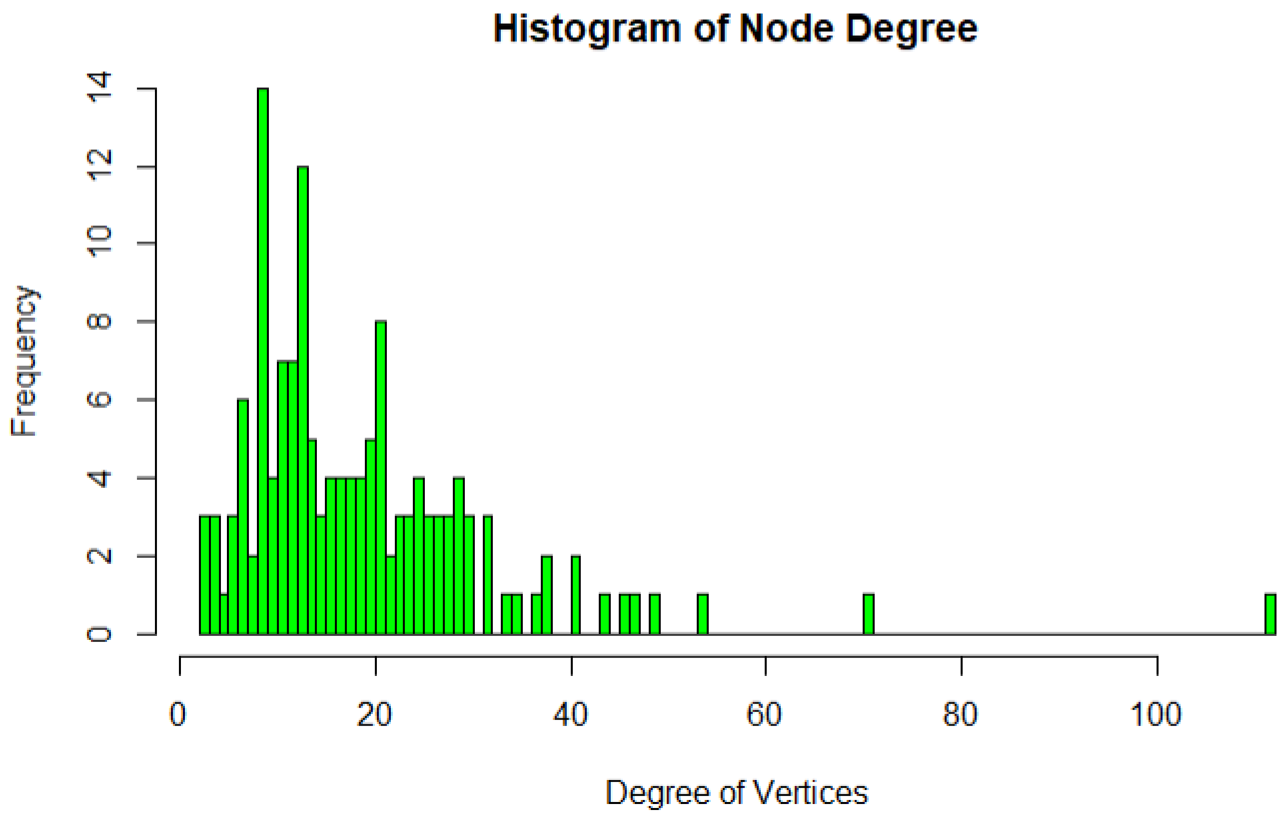
3.5b Find the pair of terms that appears frequently together

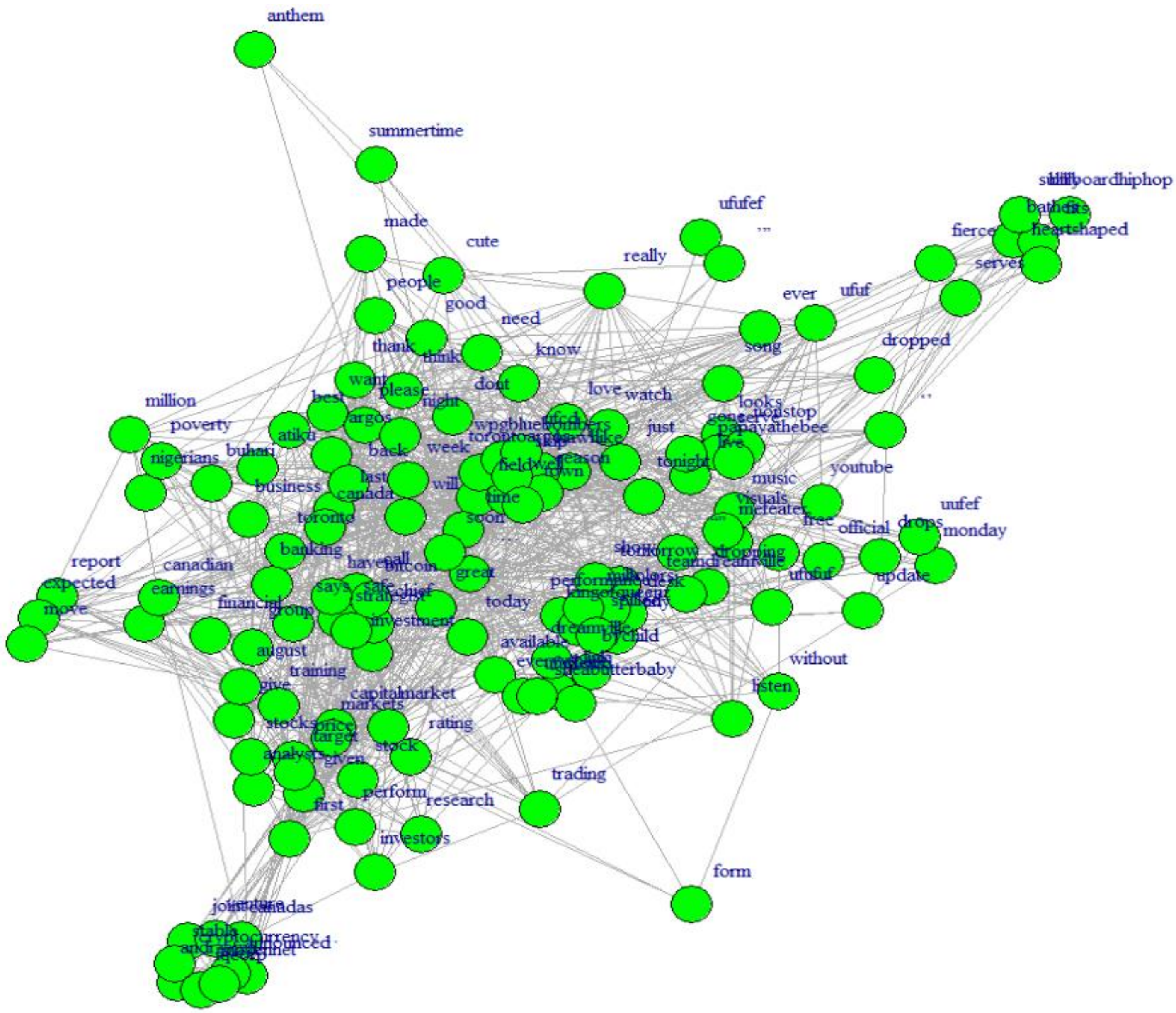
~~~
```{r}
# Term document matrix to convert unstructure text into structured for easier analysis
BMOtdmat <- TermDocumentMatrix(BMOcorp)
BMOtdmat <- as.matrix(BMOtdmat)

# Convert matrix into binary, that is whether a term appears (1) or not (0)
BMOtdmat[BMOtdmat>1] <- 1

# Create term-term matrix
BMOttm <- BMOtdmat %*% t(BMOtdmat) # Multiply BMOtdmat and transpose of BMOtdmat
```
~~~

3.5.1b Find the network of terms

```
```{r}
BMOg <- graph.adjacency(BMOttm, weighted = T, mode = 'undirected')
BMOg <- simplify(BMOg) # To prevent looping of same terms
V(BMOg)$label <- V(BMOg)$name # Labels for the terms
V(BMOg)$degree <- degree(BMOg) # How often each term appears, or number of connections between terms
BMOg
```
```

```
IGRAPH 62b8aab UNW- 9131 76282 --
+ attr: name (v/c), label (v/c), degree (v/n), weight (e/n)
+ edges from 62b8aab (vertex names):
 [1] analysts--arcc       analysts--ares       analysts--capital       analysts--given       analysts--markets     analysts--price      analysts--target     analysts--steel
 [9] analysts--primo      analysts--prmw       analysts--water         analysts--stocks      analysts--coverage    analysts--okta       analysts--receives   analysts--earnings
[17] analysts--rating     analysts--trust      analysts--initiated     analysts--zscaler     analysts--market      analysts--mastercard analysts--gild       analysts--gilead
[25] analysts--sciences   analysts--group      analysts--international analysts--connections analysts--give        analysts--waste      analysts--increased  analysts--mcdonald
[33] analysts--amgen      analysts--amgn       analysts--national      analysts--dean        analysts--investment  analysts--terex      analysts--perform    analysts--hudbay
[41] analysts--minerals   analysts--average    analysts--earns         analysts--“hold”      analysts--foods       analysts--energy     analysts--brands     analysts--management
[49] analysts--onemain    analysts--nutrien    analysts--medical       analysts--match       analysts--covered     analysts--equities   analysts--consensus  analysts--brewing
[57] analysts--coors      analysts--molson     analysts--offer         analysts--services    analysts--mednax      analysts--nexa       analysts--resources  analysts--restaurant
+ ... omitted several edges
```

3.5.2b Find optimal term count for optimal (neither too busy nor sparse) visualisation of network diagram

```
```{r}
# Reset BMOtdmat if needed to experiment with optimal rowSums(BMOtdmat)>?
# The larger ? is, the more visible the network of terms is, but less granular
BMOtdmat <- TermDocumentMatrix(BMOcorp)
BMOtdmat <- as.matrix(BMOtdmat)

# rowSums counts the total frequency; that is, keep terms that appear > 25 times
BMOtdmat <- BMOtdmat[rowSums(BMOtdmat)>50,]

# Re-run earlier code
BMOtdmat[BMOtdmat>1] <- 1 # Whenvevr BMOtdmat is > 1, assign value of 1
BMOtermM <- BMOtdmat %*% t(BMOtdmat)
BMOg <- graph.adjacency(BMOtermM, weighted = T, mode = 'undirected')
BMOg <- simplify(BMOg) # To prevent looping of same terms
V(BMOg)$label <- V(BMOg)$name
V(BMOg)$degree <- degree(BMOg)

# Histogram of node degree
hist(V(BMOg)$degree,
     breaks = 100, # how many bars
     col = 'green',
     main = 'Histogram of Node Degree',
     ylab = 'Frequency',
     xlab = 'Degree of Vertices')
```
```



3.5.3b Plot the network of terms diagram

```
```{r}
# Network diagram
plot(BMOg)
plot(BMOg,
     vertex.color='green',
     vertex.size = 8, # can experiment with this
     vertex.label.dist = 1.5)
```
```

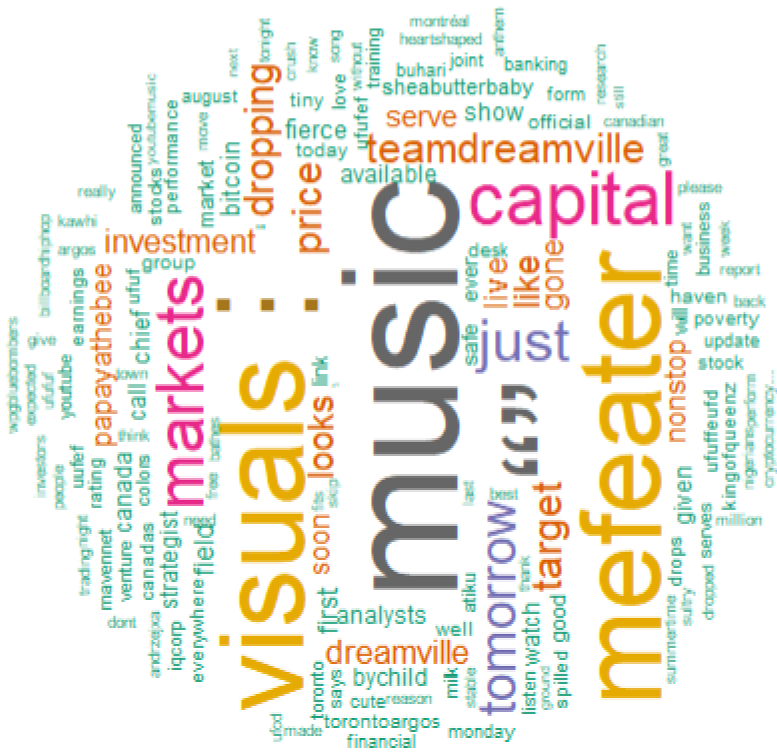


Obvious cluster of discussion on cryptocurrency for bmo

3.6b Word cloud

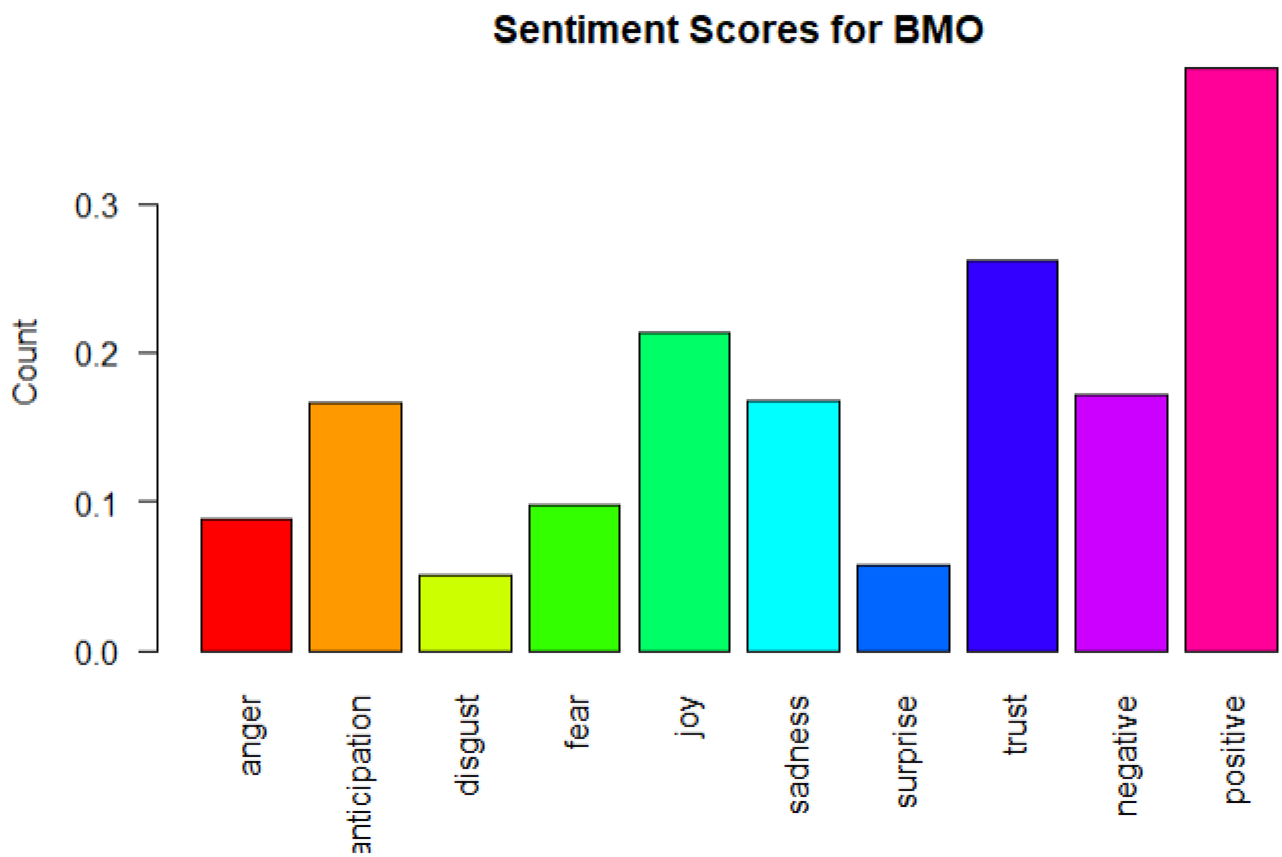
```{r}
BMOtdmat <- TermDocumentMatrix(BMOcorp)
BMOtdmat <- as.matrix(BMOtdmat)

BMOw <- sort(rowSums(BMOtdmat), decreasing = TRUE)
wordcloud(words = names(BMOw),
          freq = BMOw,
          max.words = 150,
          random.order = F,
          min.freq = 5,
          colors = brewer.pal(8, 'Dark2'),
          scale = c(5, 0.3),
          rot.per = 0.7)
```



Lots of discussion about (possibly the artiste) Ari Lennox, instead of banking stuff. Can include this into the stopword distionary. Afternote: "Ari Lennox" was included in stopword dictionary but "music" was not. Apparently there is a Ari Lennox music video about BMO, hence the enormous volume of tweets about this subject

3.7b Sentiment analysis

```
```{r}
BMOsen <- get_nrc_sentiment(BMOtext)

BMO_sen_df <- t(as.data.frame(colMeans(BMOsen)))
write.table(BMO_sen_df,"/Users/sgchr/Documents/CSDA1050/Data/Allbanks_sen_df.csv",
append=T, row.names=F, col.names=F,  sep=",")

barplot(colMeans(BMOsen),
        las = 2,
        col = rainbow(10),
        ylab = 'Count',
        main = 'Sentiment Scores for BMO')
```
```



BNS

3.1c Create and compact the term-document matrix (TDM) for each bank

```
```{r}
# Load the archived tweets
BNS_csv <- read.csv("/Users/sgchr/Documents/CSDA1050/Data/Bns.csv", header = TRUE)

# Build the initial term-document matrix Corpus
BNStext <- iconv(BNS_csv$text, to = 'UTF-8')
BNScorp <- Corpus(VectorSource(BNStext))

# Compact the Corpus
BNScorp <- tm_map(BNScorp, tolower) # Not crucial for text analytics but good practice to
do so
BNScorp <- tm_map(BNScorp, removePunctuation) # Remove punctuations
BNScorp <- tm_map(BNScorp, removeNumbers) # Remove numbers

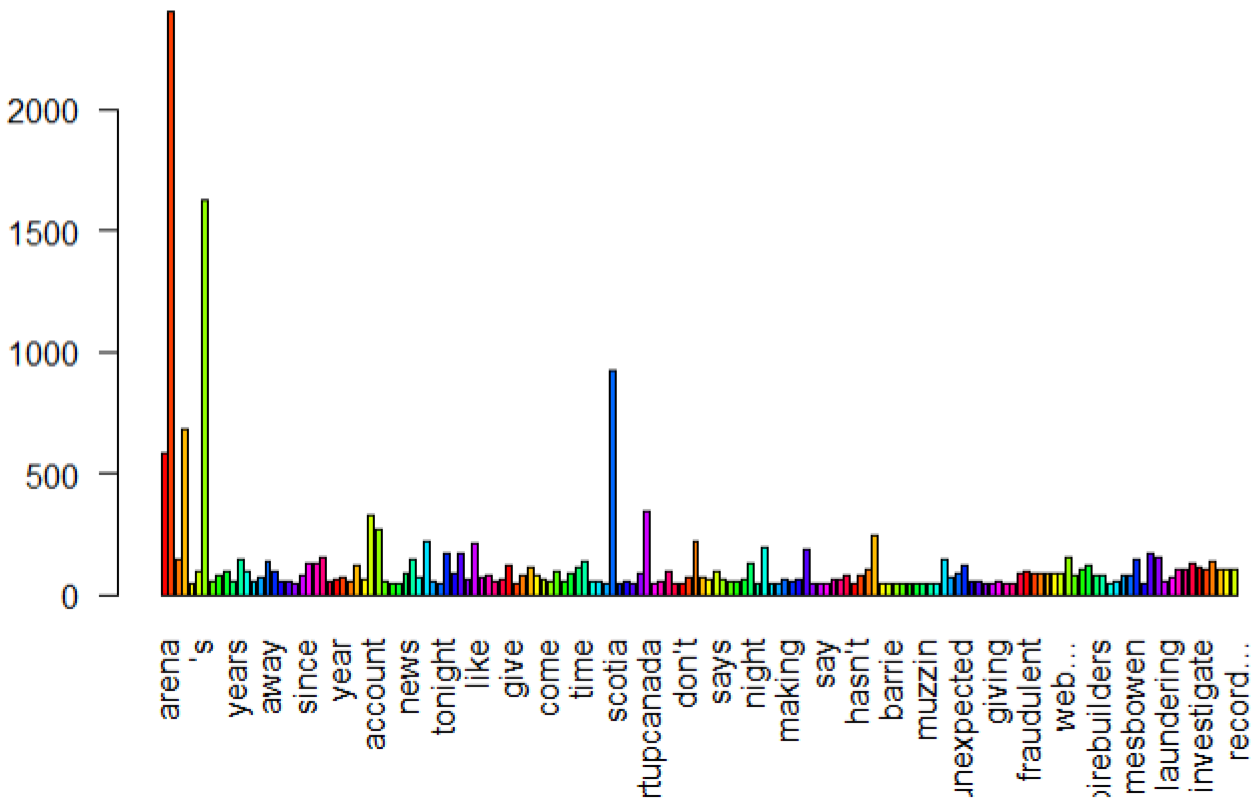
# Remove URL
removeURL <- function(x) gsub('http[[:alnum:]]*', '', x)
BNScorp <- tm_map(BNScorp, content_transformer(removeURL))

BNScorp <- tm_map(BNScorp, removeWords, stopwords(kind="en")) # Remove "standard" stop
words

BNScorp <- tm_map(BNScorp, stripWhitespace) # remove leftover from the preceding removal

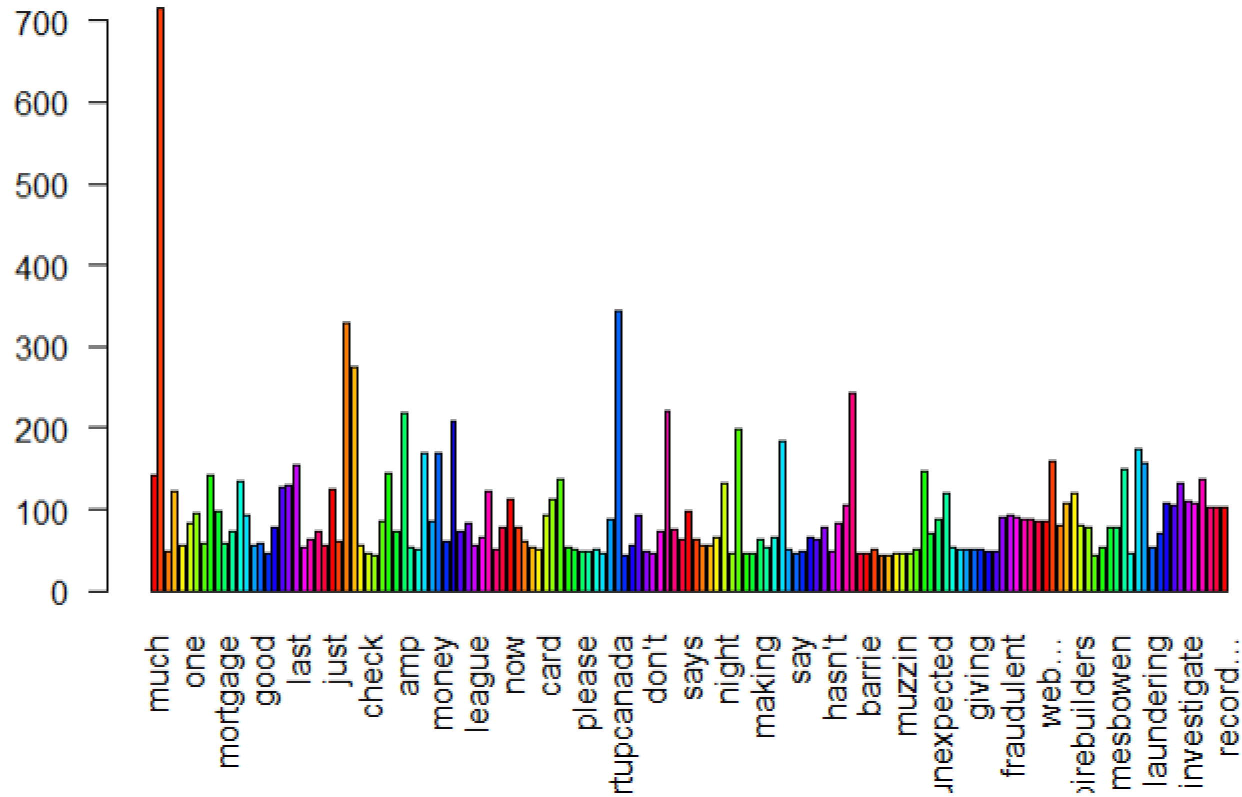
# Create the term document matrix
BNStdmat <- TermDocumentMatrix(BNScorp, control = list(minWordLength=c(1,Inf)))
BNStdm <- removeSparseTerms(BNStdmat, sparse=0.99)
BNSmat <- as.matrix(BNStdm)
BNSfreq <- rowSums(BNSmat)
```
```

```
BNSfreq <- subset(BNSfreq, BNSfreq>=30) # Experiment with BNSfreq=? to find the optimal
of of terms in the barplot
barplot(BNSfreq,
        las=2, # list all words vertically
        col = rainbow(25))
```



3.2c Further compact the term-document matrix

```{r}
MyStopwords <- c("scotiabank", "scotia", "bank", "bns", "arena", "’m", "…","t…")
BNScorp <- tm_map(BNScorp, removeWords, MyStopwords)

BNScorp <- tm_map(BNScorp, gsub, pattern = 'aphria', replacement = 'aphriainc') # Replace
words

BNScorp <- tm_map(BNScorp, stripWhitespace) # remove leftover from the preceding removal

# Repeat the preceding codes
BNStdmat <- TermDocumentMatrix(BNScorp, control = list(minWordLength=c(1,Inf)))
BNStdm <- removeSparseTerms(BNStdmat, sparse=0.99)
BNSmat <- as.matrix(BNStdm)
BNSfreq <- rowSums(BNSmat)
BNSfreq <- subset(BNSfreq, BNSfreq>=30) # Experiment with BNSfreq=? to find the optimal
of of terms in the barplot
barplot(BNSfreq,
        las=2, # list all words vertically
        col = rainbow(25))
```



3.3c Clustering of terms/tweets k-means method

```{r}
set.seed(123)

BNSm1 <- t(BNSmat) # Transpose BNSmat
BNSk <- 21
BNSkc <- kmeans(BNSm1, BNSk)
BNSkc
```

```
 within cluster sum of squares by cluster:
  [1]  132.573643  107.938462  340.560510   25.818182  186.347826   77.450980  104.285714   73.859155
140.275862  181.200000 4121.691330  131.000000  274.682927    3.954023  349.466667   56.122449
13.695238
 [18]   98.604651  750.947531   94.000000    2.937500    3.911111
  (between_SS / total_SS =  49.1 %)
```

Observations:

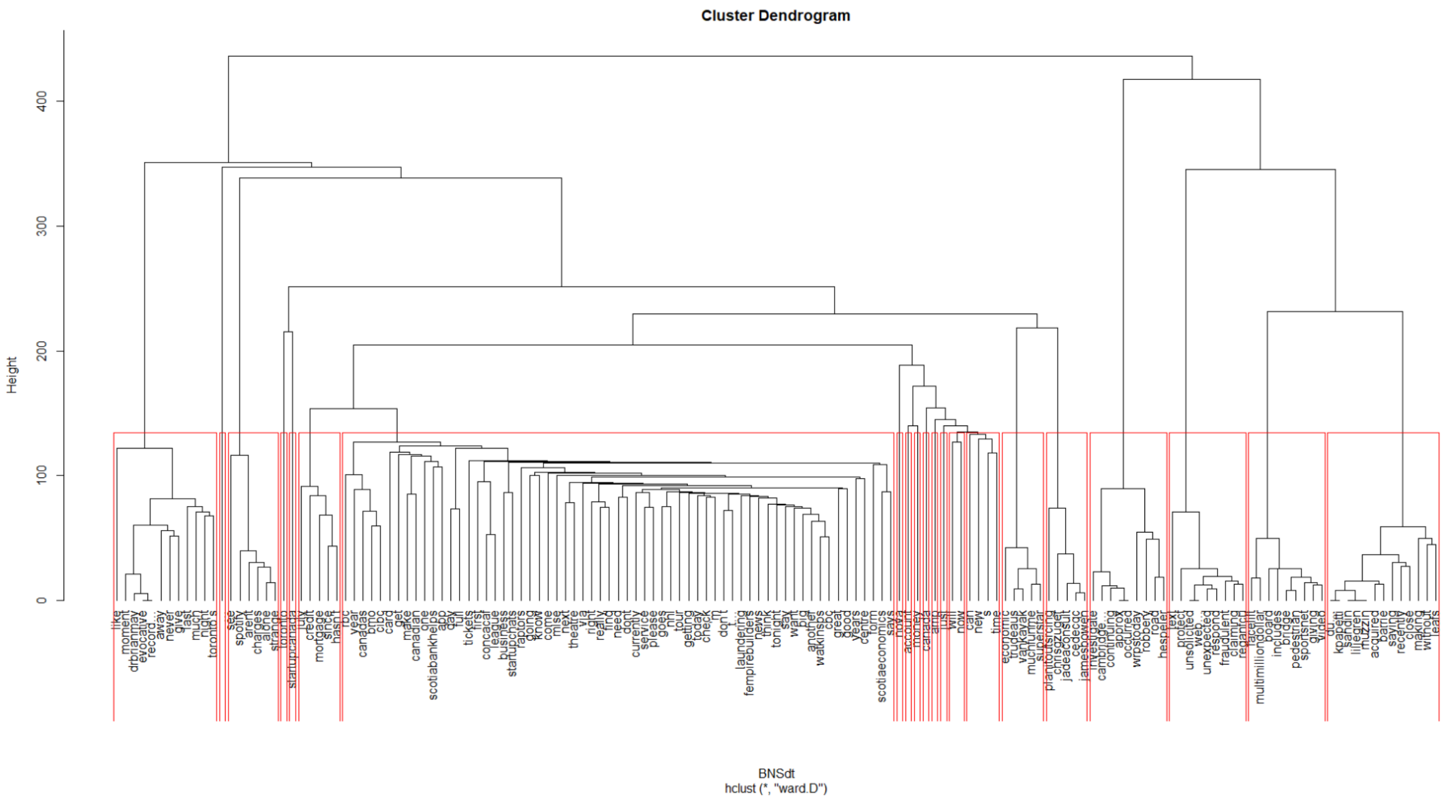
Experimentation for k = 10 to 22 implied the best mix of distance within and between clusters for the current CIBC dataset is k=21

```
k   Within clusters   Between clusters
=   ===============   ================
10  1066              25.4
11  958               26.2
12  781               34.4
13  852               22.4
14  646               36.7
15  619               35.0
16  585               34.4
17  484               42.5
18  535               32.7
19  502               33.2
20  521               27.0
```

```
21   311              49.1 <-
22   403              37.8
```

3.4c Plot the hierarchical clusters

```{r}
# Find the distance, use scale to normalise the matrix
BNSdt <- dist(scale(BNSmat))

BNShc <- hclust(BNSdt, method = "ward.D")
plot(BNShc, hang=-1)
rect.hclust(BNShc, k=21) # 21 clusters
```



3.5c Find the pair of terms that appears frequently together

```{r}
# Term document matrix to convert unstructure text into structured for easier analysis
BNStdmat <- TermDocumentMatrix(BNScorp)
BNStdmat <- as.matrix(BNStdmat)

# Convert matrix into binary, that is whether a term appears (1) or not (0)
BNStdmat[BNStdmat>1] <- 1

# Create term-term matrix
BNSttm <- BNStdmat %*% t(BNStdmat) # Multiply BNStdmat and transpose of BNStdmat
```

3.5.1c Find the network of terms

```{r}
BNSg <- graph.adjacency(BNSttm, weighted = T, mode = 'undirected')
BNSg <- simplify(BNSg) # To prevent looping of same terms
V(BNSg)$label <- V(BNSg)$name # Labels for the terms
V(BNSg)$degree <- degree(BNSg) # How often each term appears, or number of connections
between terms
BNSg
```

```
IGRAPH 5fa5a5d UNW- 6705 73864 --
+ attr: name (v/c), label (v/c), degree (v/n), weight (e/n)
+ edges from 5fa5a5d (vertex names):
 [1] chrestontalks--reno           chrestontalks--underway       reno          --underway
underway      --…                underway      --year              underway      --toronto
 [7] underway      --raptors          underway      --mapleleafs        underway      --old
underway      --parts            underway      --play…             underway      --renovations
[13] underway      --scotiabankarena underway      --urbantoronto      underway      --venue
underway      --mapleleafs…      underway      --via               underway      --designed
[19] underway      --torontos         underway      --pictures          underway      --dollar
underway      --multimillion     underway      --renovation        underway      --cre
[25] underway      --urban            bloody        --enough            bloody        --fees
bloody        --hell             bloody        --jumping           bloody        --mean
[31] bloody        --much             bloody        --ridiculous        bloody        --work
bloody        --…                enough        --fees              enough        --hell
[37] enough        --jumping          enough        --mean              enough        --much
enough        --ridiculous       enough        --work              enough        --…
[43] enough        --good             enough        --never             enough        --going
enough        --take             enough        --team              enough        --big
+ ... omitted several edges
```

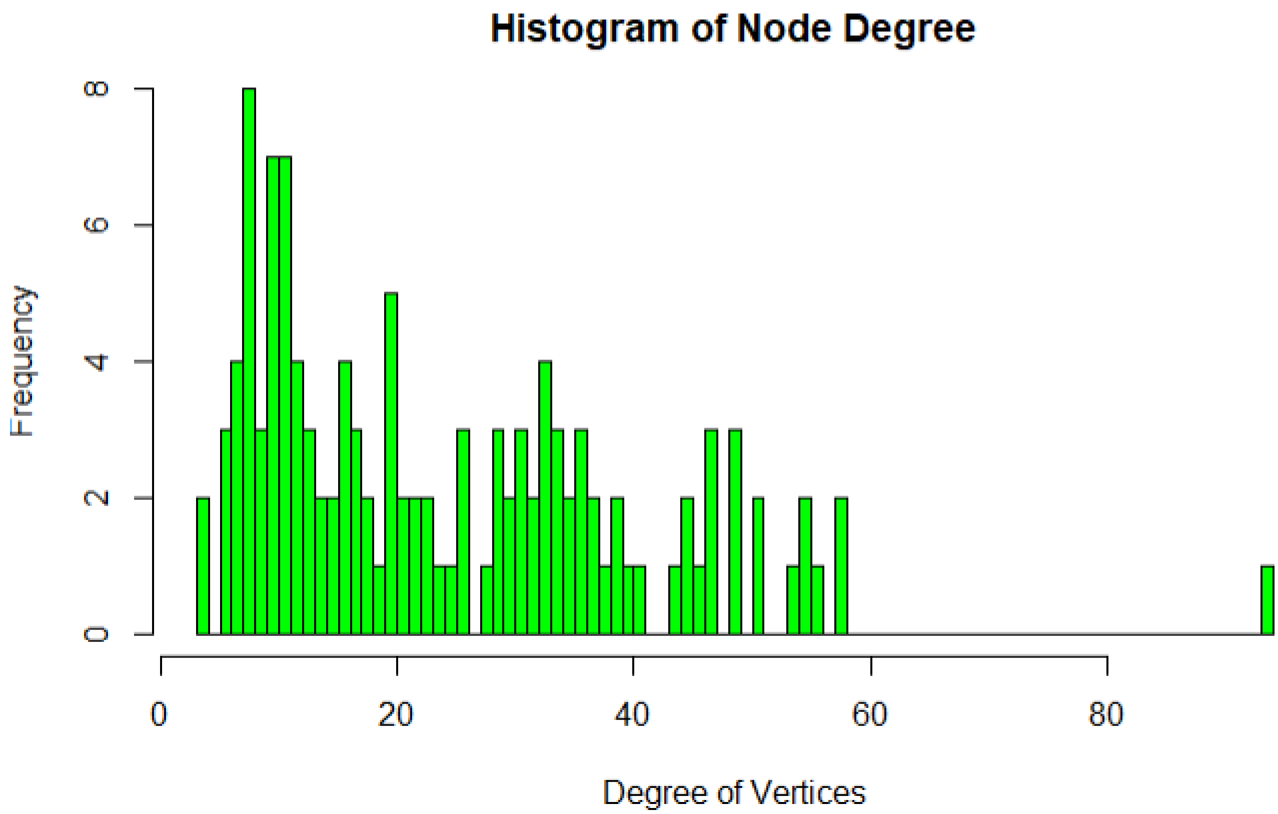
3.5.2c Find optimal term count for optimal (neither too busy nor sparse) visualisation of network diagram

````
```{r}
# Reset BNStdmat if needed to experiment with optimal rowSums(BNStdmat)>?
# The larger ? is, the more visible the network of terms is, but less granular
BNStdmat <- TermDocumentMatrix(BNScorp)
BNStdmat <- as.matrix(BNStdmat)

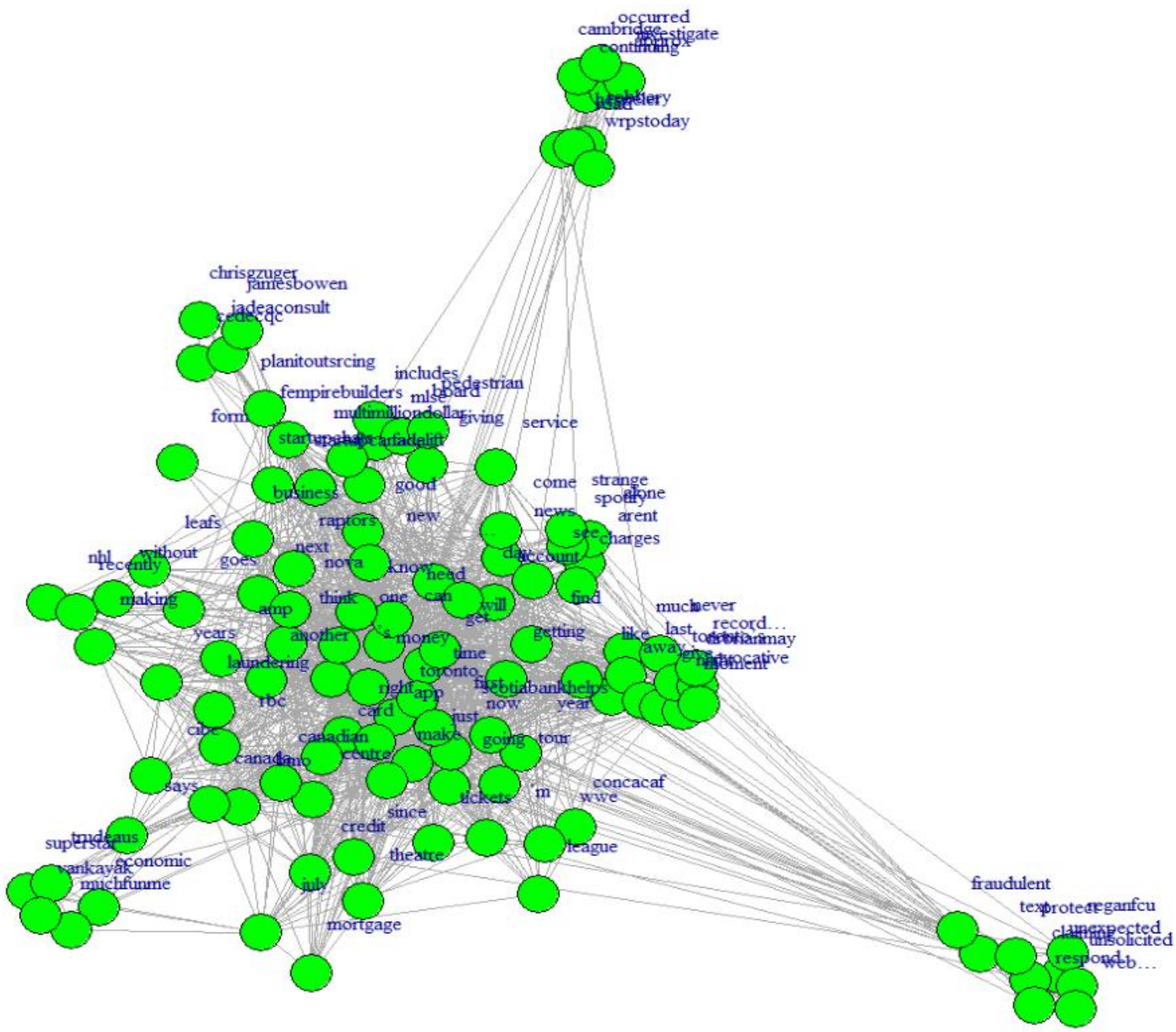
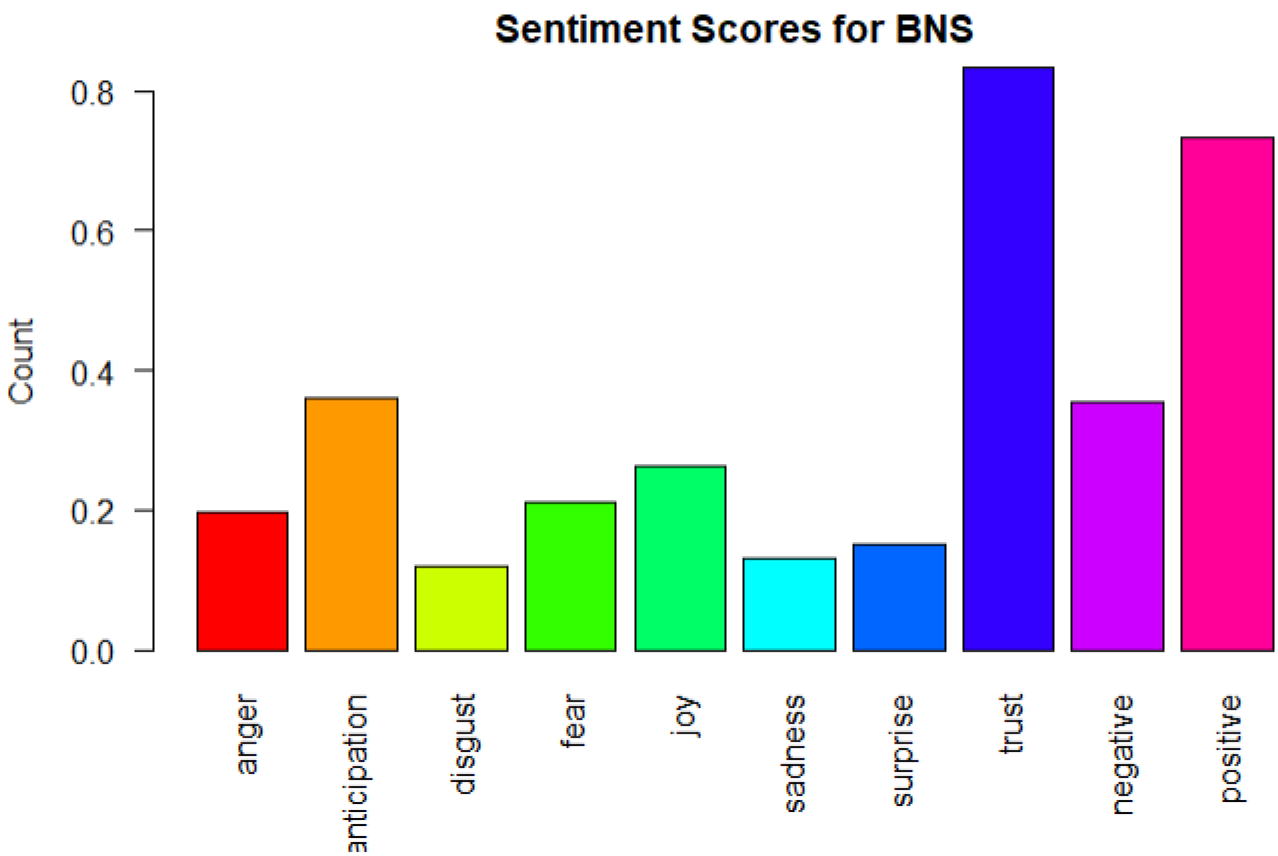
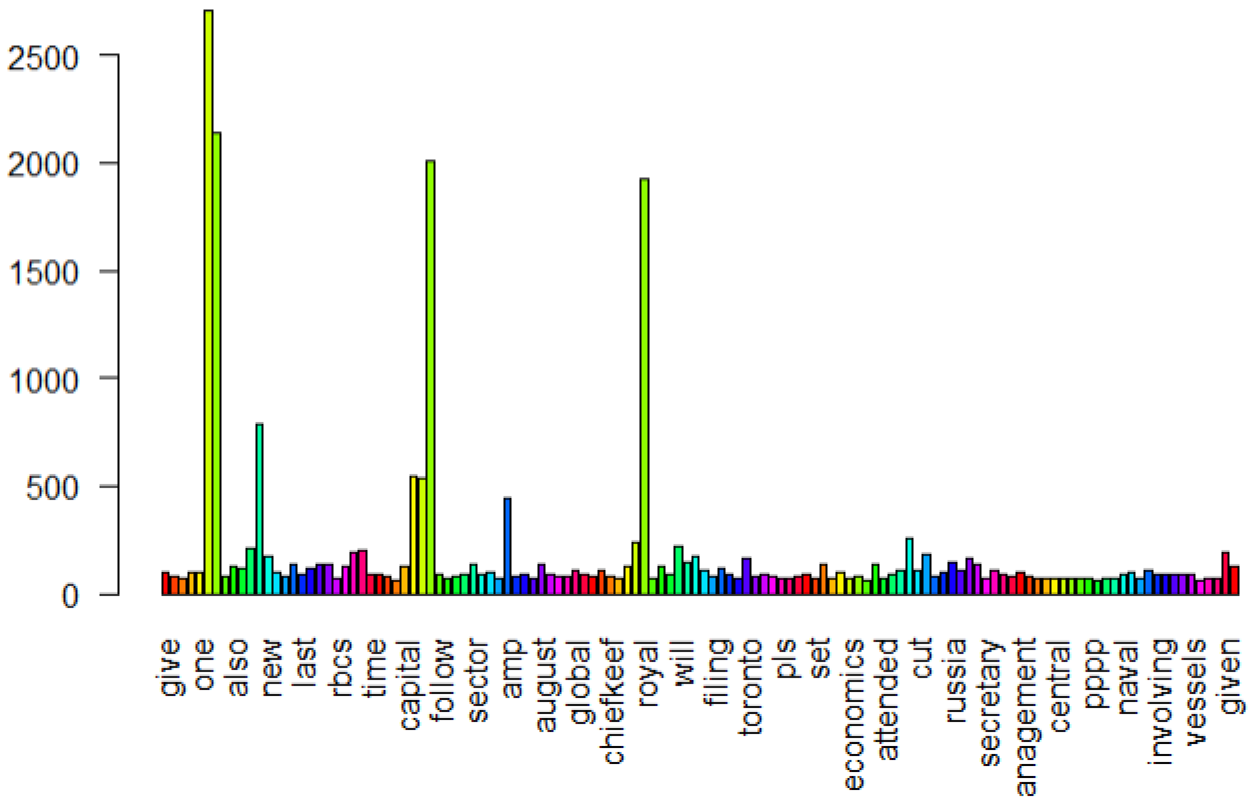
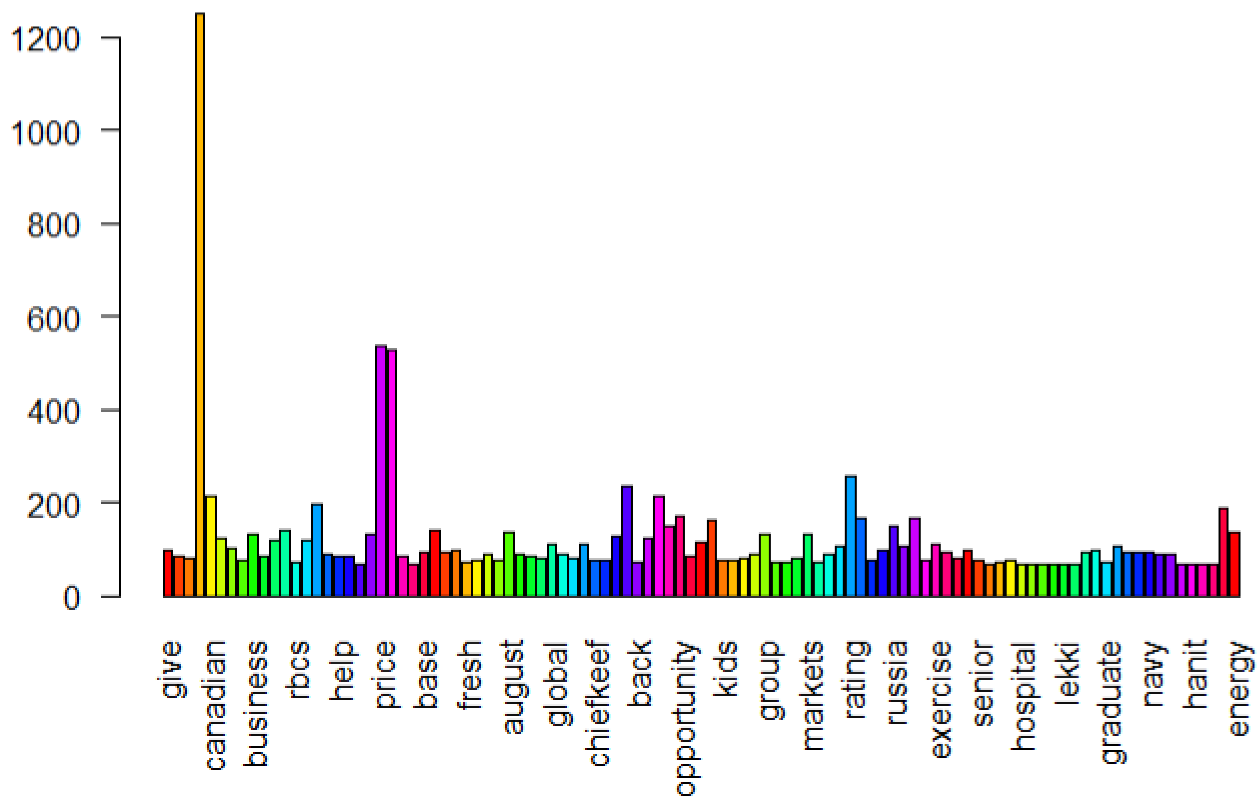
# rowSums counts the total frequency; that is, keep terms that appear > 25 times
BNStdmat <- BNStdmat[rowSums(BNStdmat)>50,]

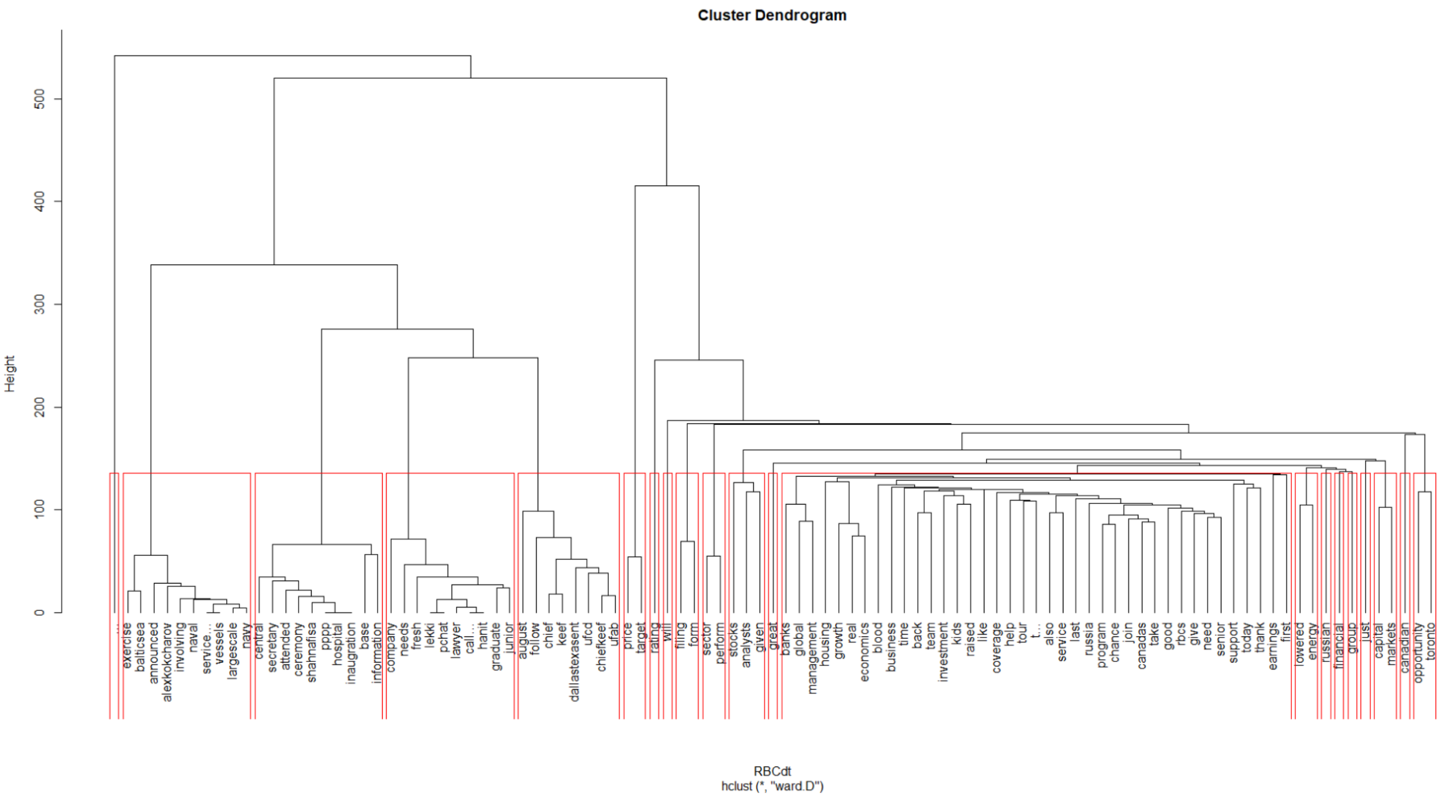
# Re-run earlier code
BNStdmat[BNStdmat>1] <- 1 # Whenvevr BNStdmat is > 1, assign value of 1
BNStermM <- BNStdmat %*% t(BNStdmat)
BNSg <- graph.adjacency(BNStermM, weighted = T, mode = 'undirected')
BNSg <- simplify(BNSg) # To prevent looping of same terms
V(BNSg)$label <- V(BNSg)$name
V(BNSg)$degree <- degree(BNSg)

# Histogram of node degree
hist(V(BNSg)$degree,
     breaks = 100, # how many bars
     col = 'green',
     main = 'Histogram of Node Degree',
     ylab = 'Frequency',
     xlab = 'Degree of Vertices')
```
````



3.5.3c Plot the network of terms diagram

```
```{r}
# Network diagram
plot(BNSg)
plot(BNSg,
     vertex.color='green',
     vertex.size = 8, # can experiment with this
     vertex.label.dist = 2)
```
```



Discussions about "trudeaus"-economic"; "fraudulent-web-response"; "occured-cambridge-investigate-continuing"

3.6c Word cloud

````
```{r}
BNStdmat <- TermDocumentMatrix(BNScorp)
BNStdmat <- as.matrix(BNStdmat)

BNSw <- sort(rowSums(BNStdmat), decreasing = TRUE)
wordcloud(words = names(BNSw),
          freq = BNSw,
          max.words = 50,
          random.order = F,
          min.freq = 5,
          colors = brewer.pal(8, 'Dark2'),
          scale = c(5, 0.3),
          rot.per = 0.7)
```
````

3.7c Sentiment analysis

```
```{r}
BNSsen <- get_nrc_sentiment(BNStext)

BNS_sen_df <- t(as.data.frame(colMeans(BNSsen)))
write.table(BNS_sen_df,"/Users/sgchr/Documents/CSDA1050/Data/Allbanks_sen_df.csv",
append=T, row.names=F, col.names=F,  sep=",")

barplot(colMeans(BNSsen),
        las = 2,
        col = rainbow(10),
        ylab = 'Count',
        main = 'Sentiment Scores for BNS')
```
```



RBC

3.1d Create and compact the term-document matrix (TDM) for each bank

```
```{r}
# Load the archived tweets
RBC_csv <- read.csv("/Users/sgchr/Documents/CSDA1050/Data/Rbc.csv", header = TRUE)

# Build the initial term-document matrix Corpus
RBCtext <- iconv(RBC_csv$text, to = 'UTF-8')
RBCcorp <- Corpus(VectorSource(RBCtext))

# Compact the Corpus
RBCcorp <- tm_map(RBCcorp, tolower) # Not crucial for text analytics but good practice to
do so
RBCcorp <- tm_map(RBCcorp, removePunctuation) # Remove punctuations
RBCcorp <- tm_map(RBCcorp, removeNumbers) # Remove numbers

# Remove URL
removeURL <- function(x) gsub('http[[:alnum:]]*', '', x)
RBCcorp <- tm_map(RBCcorp, content_transformer(removeURL))

RBCcorp <- tm_map(RBCcorp, removeWords, stopwords(kind="en")) # Remove "standard" stop
words

RBCcorp <- tm_map(RBCcorp, stripWhitespace) # remove leftover from the preceding removal

# Create the term document matrix
RBCtdmat <- TermDocumentMatrix(RBCcorp, control = list(minWordLength=c(1,Inf)))
RBCtdm <- removeSparseTerms(RBCtdmat, sparse=0.99)
RBCmat <- as.matrix(RBCtdm)
RBCfreq <- rowSums(RBCmat)
RBCfreq <- subset(RBCfreq, RBCfreq>=40) # Experiment with RBCfreq=? to find the optimal
of of terms in the barplot
```
```

```
barplot(RBCfreq,
        las=2, # list all words vertically
        col = rainbow(25))
```



3.2d Further compact the term-document matrix

```{r}
MyStopwords <- c("...", "bank", "royal", "canada", "jbgasajohn", "“”")
RBCcorp <- tm_map(RBCcorp, removeWords, MyStopwords)

RBCcorp <- tm_map(RBCcorp, gsub, pattern = '"buy"', replacement = 'buy') # Replace words
RBCcorp <- tm_map(RBCcorp, gsub, pattern = '"hold"', replacement = 'hold') # Replace
words

RBCcorp <- tm_map(RBCcorp, stripWhitespace) # remove leftover from the preceding removal

# Repeat the preceding codes
RBCtdmat <- TermDocumentMatrix(RBCcorp, control = list(minWordLength=c(1,Inf)))
RBCtdm <- removeSparseTerms(RBCtdmat, sparse=0.99)
RBCmat <- as.matrix(RBCtdm)
RBCfreq <- rowSums(RBCmat)
RBCfreq <- subset(RBCfreq, RBCfreq>=30) # Experiment with RBCfreq=? to find the optimal
of of terms in the barplot
barplot(RBCfreq,
        las=2, # list all words vertically
        col = rainbow(25))
```



3.3d Clustering of terms/tweets k-means method

```
```{r}
set.seed(123)

RBCm1 <- t(RBCmat) # Transpose RBCmat
RBCk <- 21
RBCkc <- kmeans(RBCm1, RBCk)
RBCkc
```
```

```
Within cluster sum of squares by cluster:
  [1]  127.167883   10.428571 2535.446373    0.000000    2.967391   154.930233   108.835366    76.591837
 134.075949   42.958333  173.795620  275.739583 1406.043887  341.412088  175.295775   92.701031
 1.970149
 [18]   82.509804   82.028169  209.623037   22.952381
  (between_SS / total_SS =  51.5 %)
```

```
k   Within clusters   Between clusters
=   ===============   ================
16  441               43.4
17  368               49.9
18  379               45.4
19  347               47.2
20  320               48.7
21  284               51.5 <-
22  277               51.1
```

3.4d Plot the hierarchical clusters

```
```{r}
# Find the distance, use scale to normalise the matrix
RBCdt <- dist(scale(RBCmat))

RBChc <- hclust(RBCdt, method = "ward.D")
plot(RBChc, hang=-1)
rect.hclust(RBChc, k=21) # 21 clusters
```
```

````
```
````



3.5d Find the pair of terms that appears frequently together

```{r}
# Term document matrix to convert unstructure text into structured for easier analysis
RBCtdmat <- TermDocumentMatrix(RBCcorp)
RBCtdmat <- as.matrix(RBCtdmat)

# Convert matrix into binary, that is whether a term appears (1) or not (0)
RBCtdmat[RBCtdmat>1] <- 1

# Create term-term matrix
RBCttm <- RBCtdmat %*% t(RBCtdmat) # Multiply RBCtdmat and transpose of RBCtdmat
```

3.5.1d Find the network of terms

```{r}
RBCg <- graph.adjacency(RBCttm, weighted = T, mode = 'undirected')
RBCg <- simplify(RBCg) # To prevent looping of same terms
V(RBCg)$label <- V(RBCg)$name # Labels for the terms
V(RBCg)$degree <- degree(RBCg) # How often each term appears, or number of connections between terms
RBCg
```
```

```
IGRAPH 4aa6568 UNW- 9873 90923 --
+ attr: name (v/c), label (v/c), degree (v/n), weight (e/n)
+ edges from 4aa6568 (vertex names):
 [1] consider--ever          consider--firstwork     consider--give          consider--industry
consider--like          consider--look          consider--many          consider--unique
 [9] consider--windsor       consider--work          consider--year          consider--youth
consider--yout…         consider--ythaspire     consider--investment    consider--september
[17] consider--taking        consider--m…            consider--canadians     consider--think
consider--russia        consider--russians      consider--miracle       consider--forget
[25] consider--adding        consider--airlines      consider--allenthird    consider--american
consider--otherwise     consider--section       consider--'stuc…        consider--switch
[33] consider--ottawa        consider--kyruer        consider--laws          consider--prosecuti…
consider--protect       consider--spurious      consider--unsafe        consider--cardiac
[41] consider--cardiologists consider--drugholy      consider--grail         consider--manetteness
consider--pedsicu       consider--surgeons      consider--transfusions  consider--ucufef
[49] ever     --firstwork    ever     --give          ever     --industry     ever     --like
ever     --look          ever     --many          ever     --unique       ever     --windsor
[57] ever     --work         ever     --year          ever     --youth        ever     --yout…
ever     --ythaspire     ever     --…             ever     --canadian     ever     --read
+ ... omitted several edges
```

3.5.2d Find optimal term count for optimal (neither too busy nor sparse) visualisation of network diagram

```{r}
# Reset RBCtdmat if needed to experiment with optimal rowSums(RBCtdmat)>?
# The larger ? is, the more visible the network of terms is, but less granular
RBCtdmat <- TermDocumentMatrix(RBCcorp)
RBCtdmat <- as.matrix(RBCtdmat)

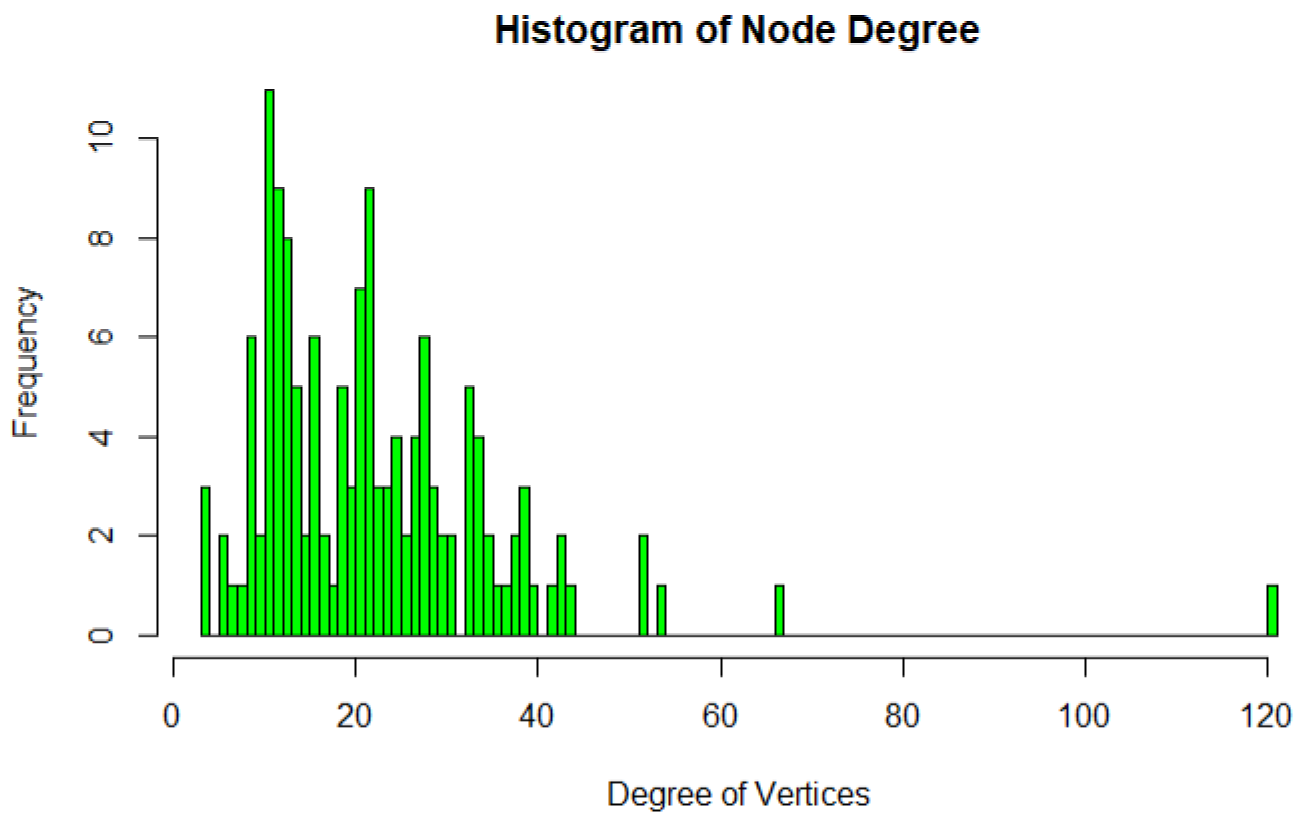
# rowSums counts the total frequency; that is, keep terms that appear > 25 times
RBCtdmat <- RBCtdmat[rowSums(RBCtdmat)>50,]

# Re-run earlier code
RBCtdmat[RBCtdmat>1] <- 1 # Whenvevr RBCtdmat is > 1, assign value of 1
RBCtermM <- RBCtdmat %*% t(RBCtdmat)

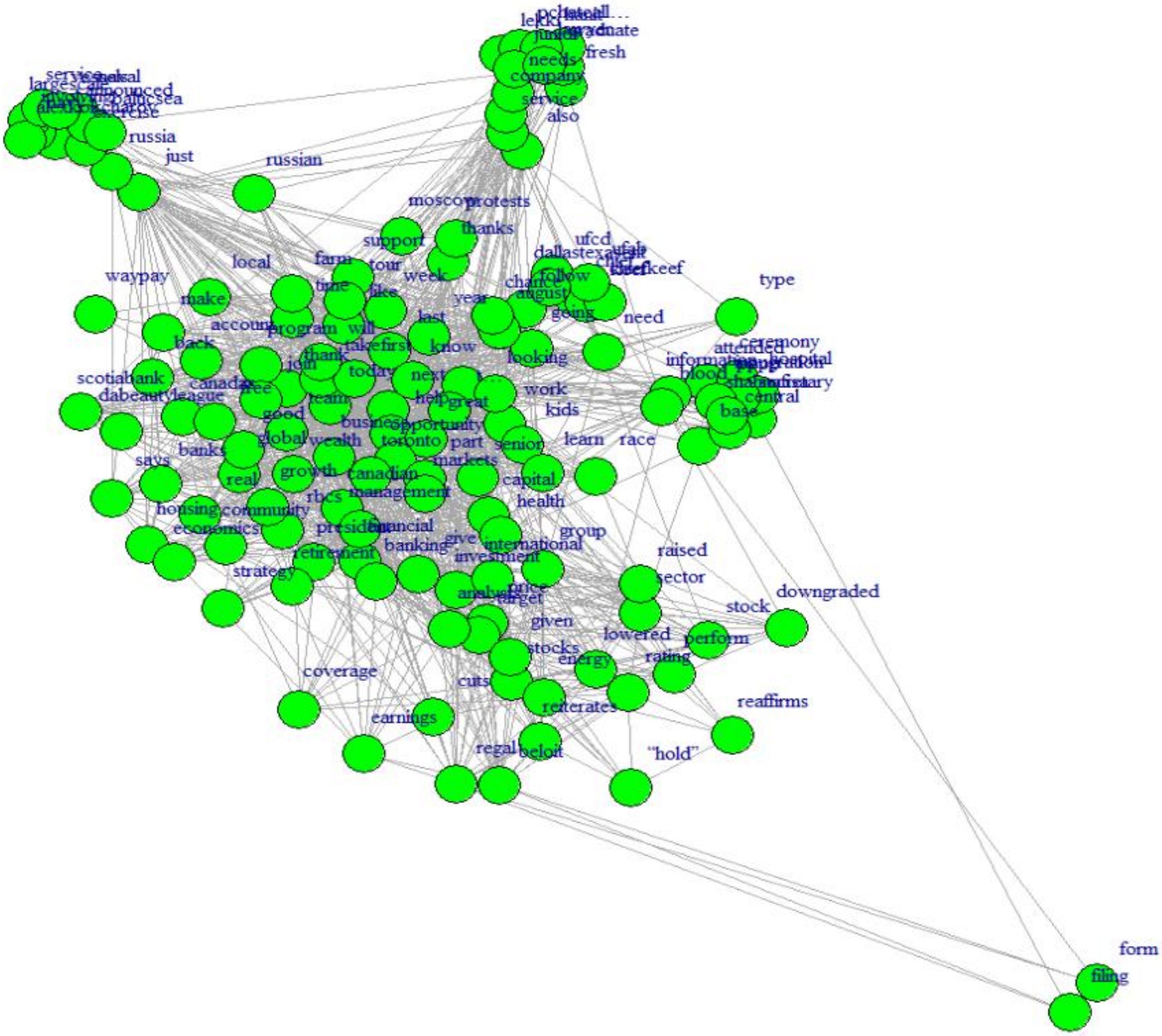
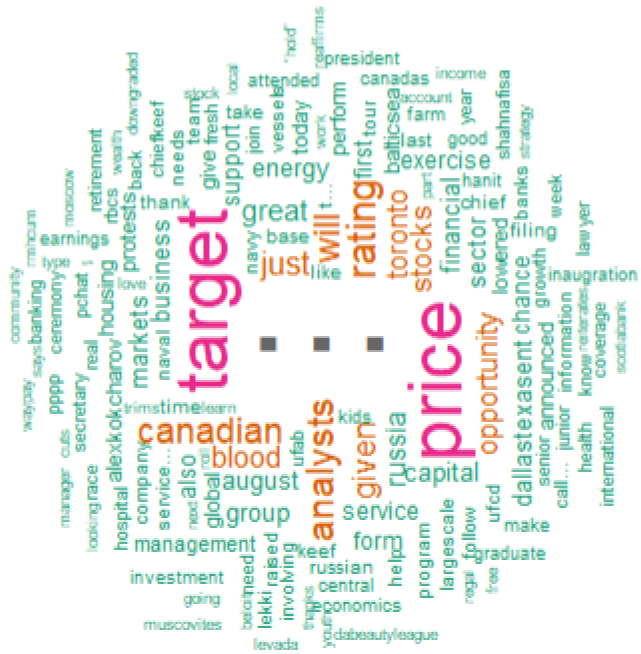
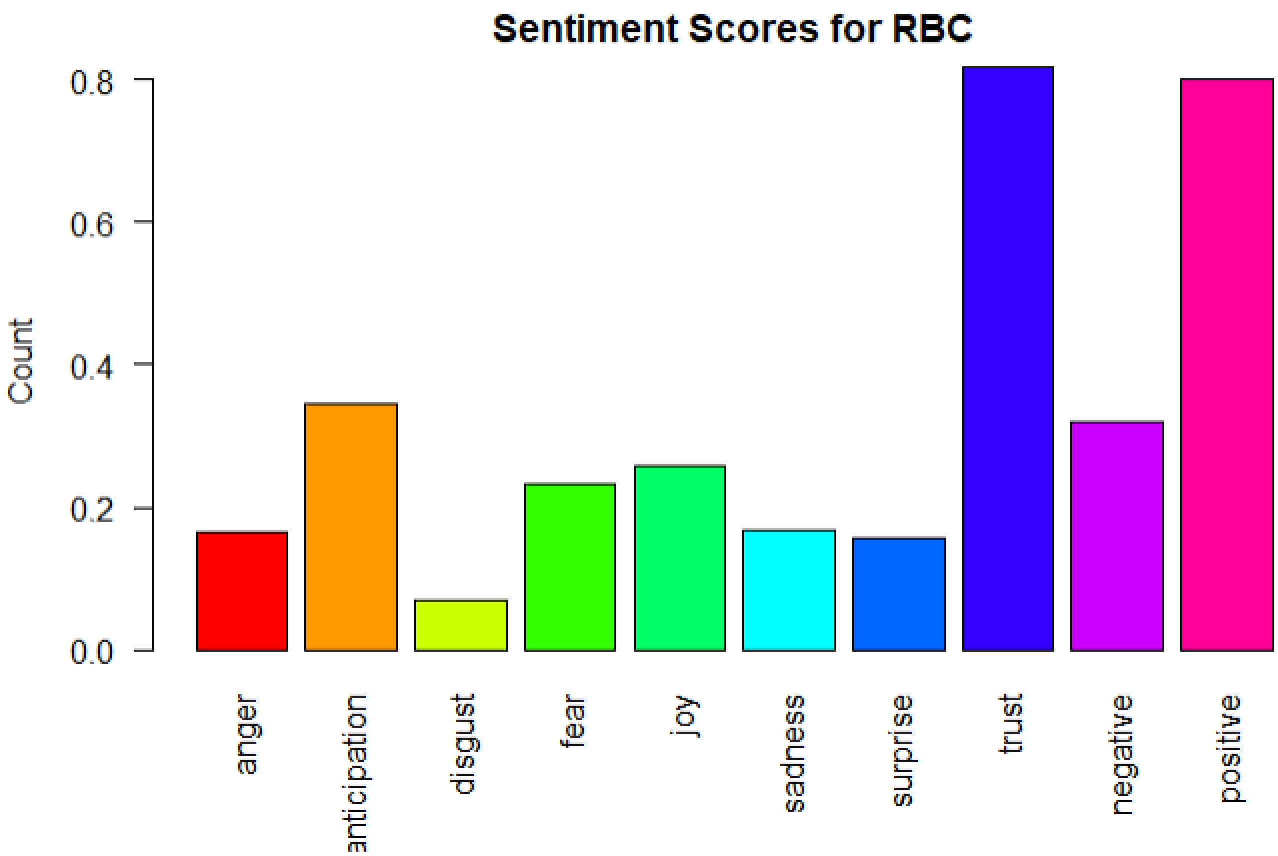
RBCg <- graph.adjacency(RBCtermM, weighted = T, mode = 'undirected')
RBCg <- simplify(RBCg) # To prevent looping of same terms

V(RBCg)$label <- V(RBCg)$name
V(RBCg)$degree <- degree(RBCg)

# Histogram of node degree
hist(V(RBCg)$degree,
     breaks = 100, # how many bars
     col = 'green',
     main = 'Histogram of Node Degree',
     ylab = 'Frequency',
     xlab = 'Degree of Vertices')
```



3.5.3d Plot the network of terms diagram

```
```{r}
# Network diagram
plot(RBCg)
plot(RBCg,
     vertex.color='green',
     vertex.size = 8, # can experiment with this
     vertex.label.dist = 1.5)
```
```



Some discussions about russia and protest; "largest-service-announced"; "central-base-information"

3.6d Word cloud

```
```{r}
RBCtdmat <- TermDocumentMatrix(RBCcorp)
RBCtdmat <- as.matrix(RBCtdmat)

RBCw <- sort(rowSums(RBCtdmat), decreasing = TRUE)
wordcloud(words = names(RBCw),
          freq = RBCw,
          max.words = 150,
          random.order = F,
          min.freq = 5,
          colors = brewer.pal(8, 'Dark2'),
          scale = c(5, 0.3),
          rot.per = 0.7)
```
```



3.7d Sentiment analysis

```
```{r}
RBCsen <- get_nrc_sentiment(RBCtext)

RBC_sen_df <- t(as.data.frame(colMeans(RBCsen)))
write.table(RBC_sen_df,"/Users/sgchr/Documents/CSDA1050/Data/Allbanks_sen_df.csv",
append=T, row.names=F, col.names=F,  sep=",")

barplot(colMeans(RBCsen),
        las = 2,
        col = rainbow(10),
        ylab = 'Count',
        main = 'Sentiment Scores for RBC')
```
```



TD

3.1e Create and compact the term-document matrix (TDM) for each bank

```
```{r}
# Load the archived tweets
TD_csv <- read.csv("/Users/sgchr/Documents/CSDA1050/Data/Td.csv", header = TRUE)

# Build the initial term-document matrix Corpus
TDtext <- iconv(TD_csv$text, to = 'UTF-8')
TDcorp <- Corpus(VectorSource(TDtext))

# Compact the Corpus
TDcorp <- tm_map(TDcorp, tolower) # Not crucial for text analytics but good practice to
do so
TDcorp <- tm_map(TDcorp, removePunctuation) # Remove punctuations
TDcorp <- tm_map(TDcorp, removeNumbers) # Remove numbers

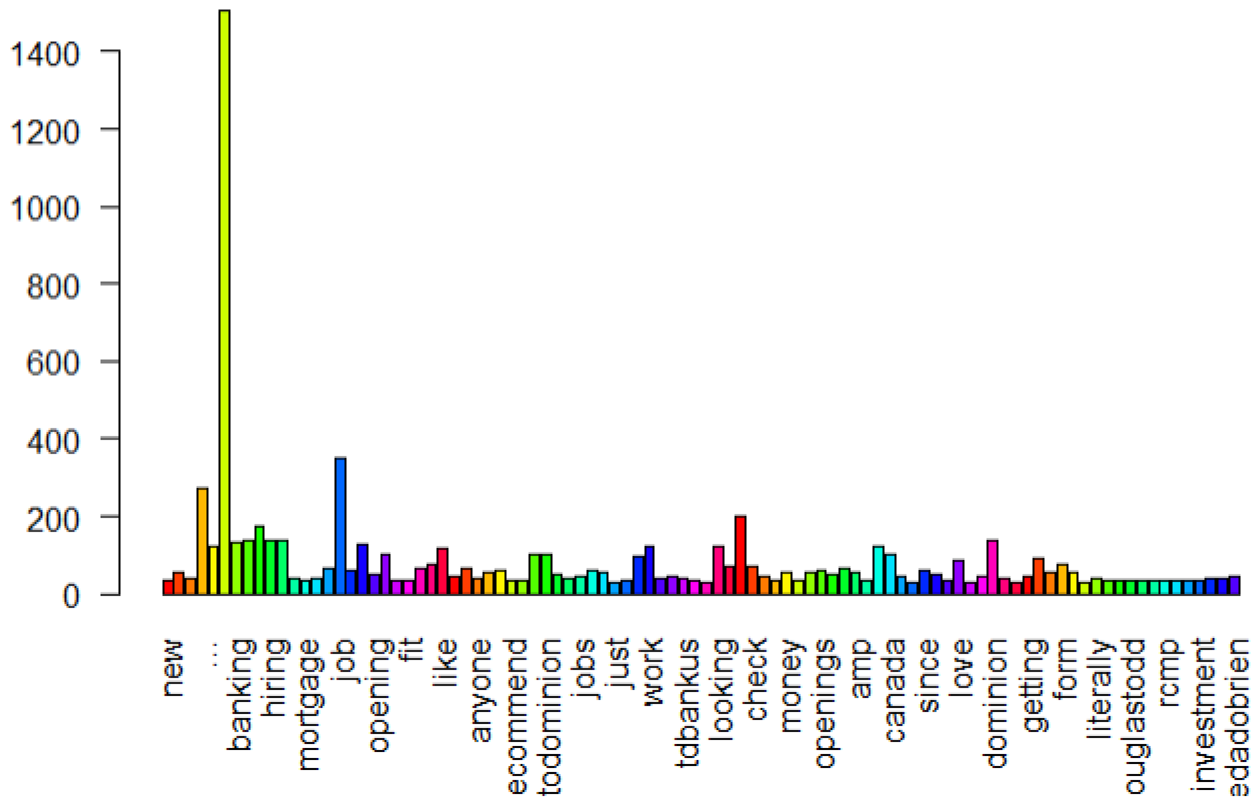
# Remove URL
removeURL <- function(x) gsub('http[[:alnum:]]*', '', x)
TDcorp <- tm_map(TDcorp, content_transformer(removeURL))

TDcorp <- tm_map(TDcorp, removeWords, stopwords(kind="en")) # Remove "standard" stop
words

TDcorp <- tm_map(TDcorp, stripWhitespace) # remove leftover from the preceding removal

# Create the term document matrix
TDtdmat <- TermDocumentMatrix(TDcorp, control = list(minWordLength=c(1,Inf)))
TDtdm <- removeSparseTerms(TDtdmat, sparse=0.99)
TDmat <- as.matrix(TDtdm)
TDfreq <- rowSums(TDmat)
```
```

```
TDfreq <- subset(TDfreq, TDfreq>=30) # Experiment with TDfreq=? to find the optimal of of
terms in the barplot
barplot(TDfreq,
        las=2, # list all words vertically
        col = rainbow(25))
```



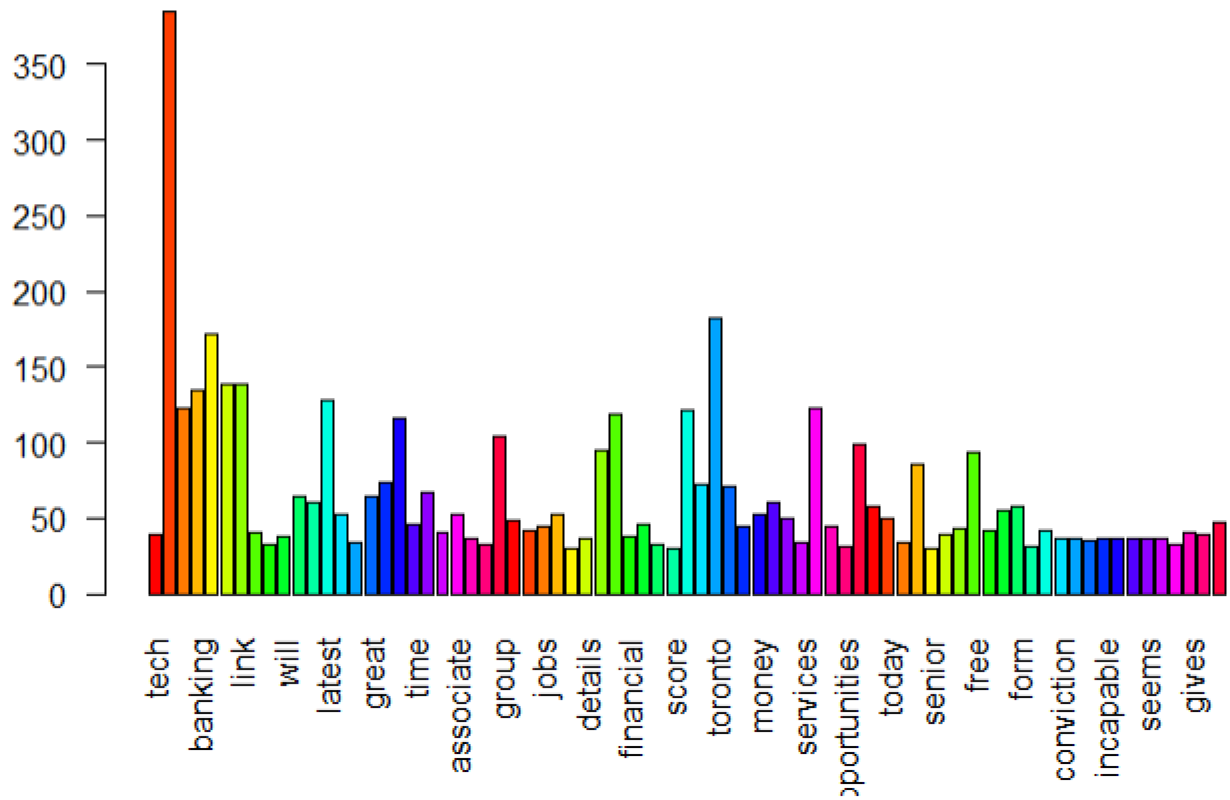
3.2e Further compact the term-document matrix

```{r}
MyStopwords <- c("tdcanada", "...", "bank", "todominion", "tdbankus", "banks",
"dominion", "torontodominion", "p…", "n…", "ugordonhoekstrau")
TDcorp <- tm_map(TDcorp, removeWords, MyStopwords)

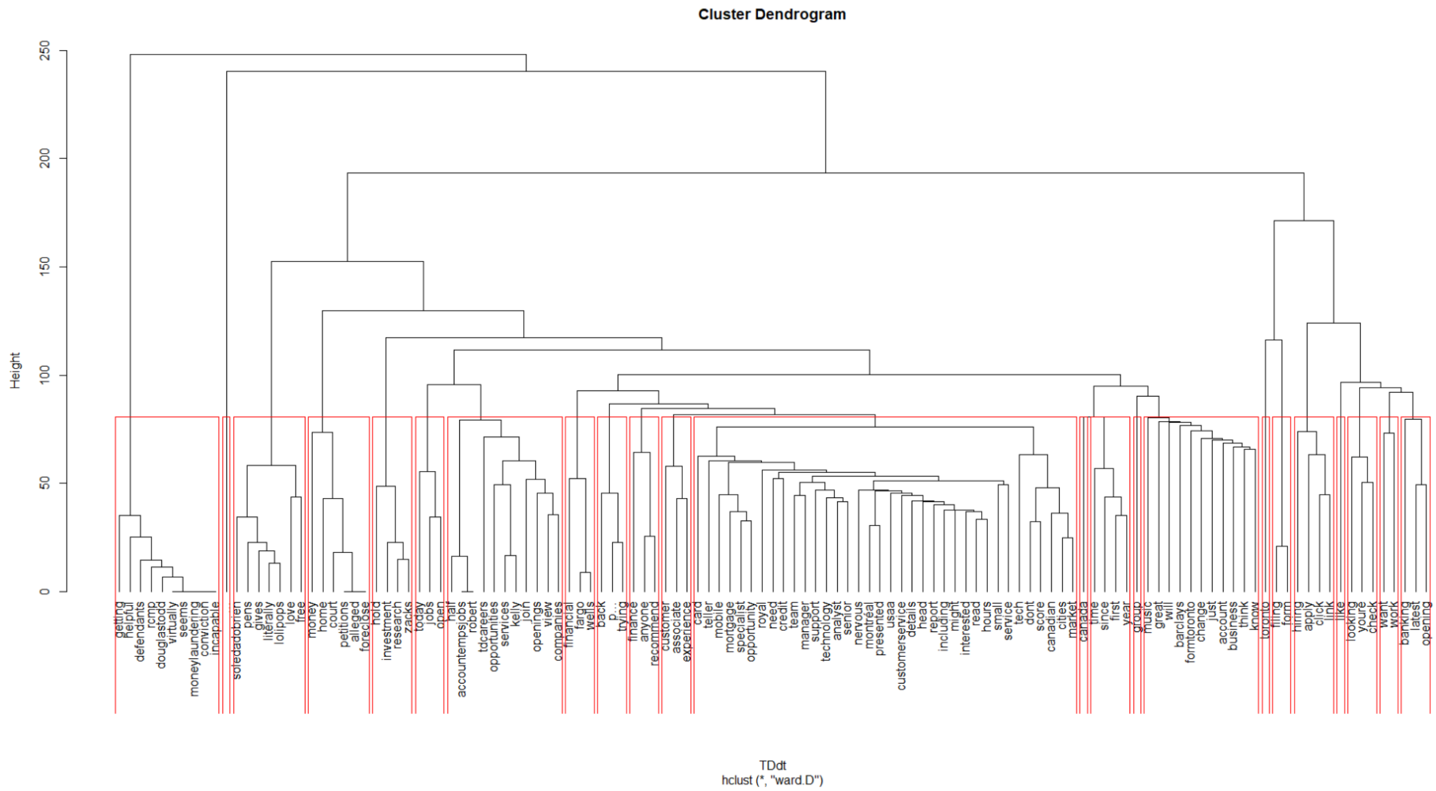
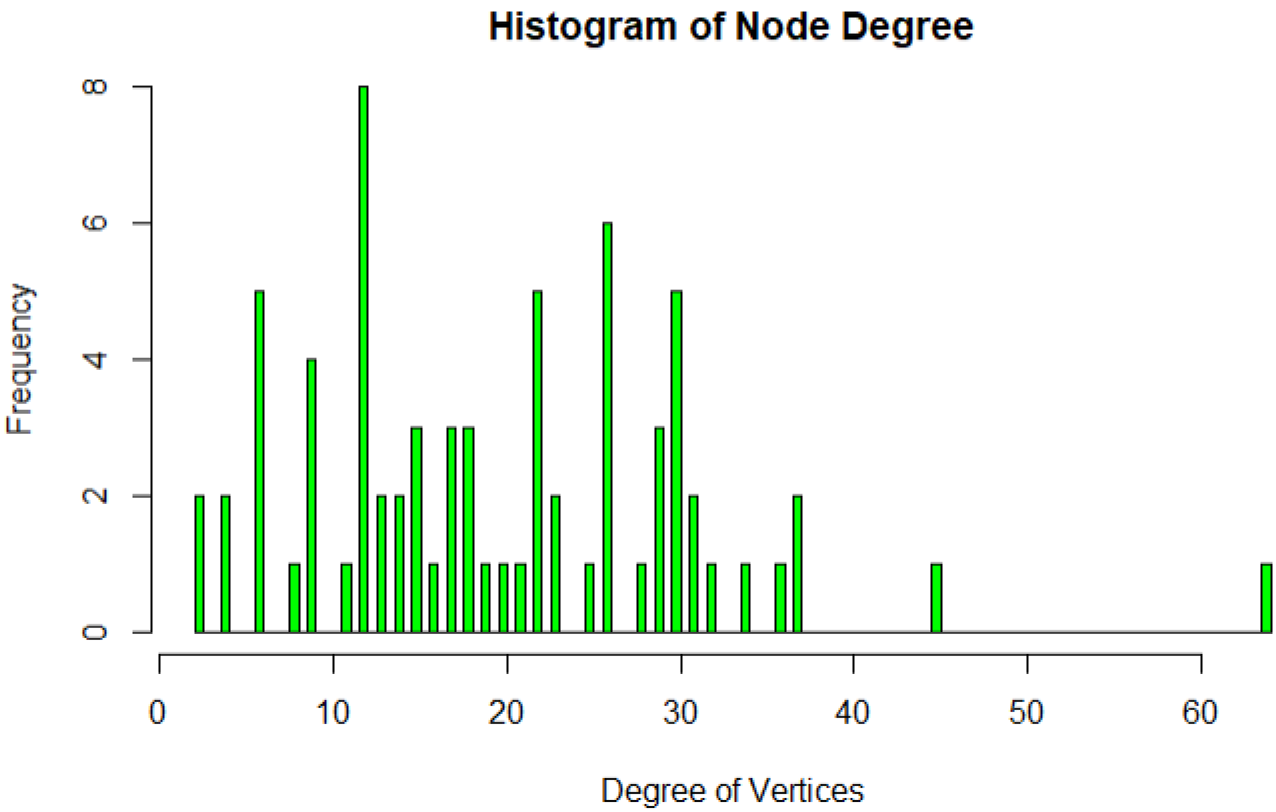
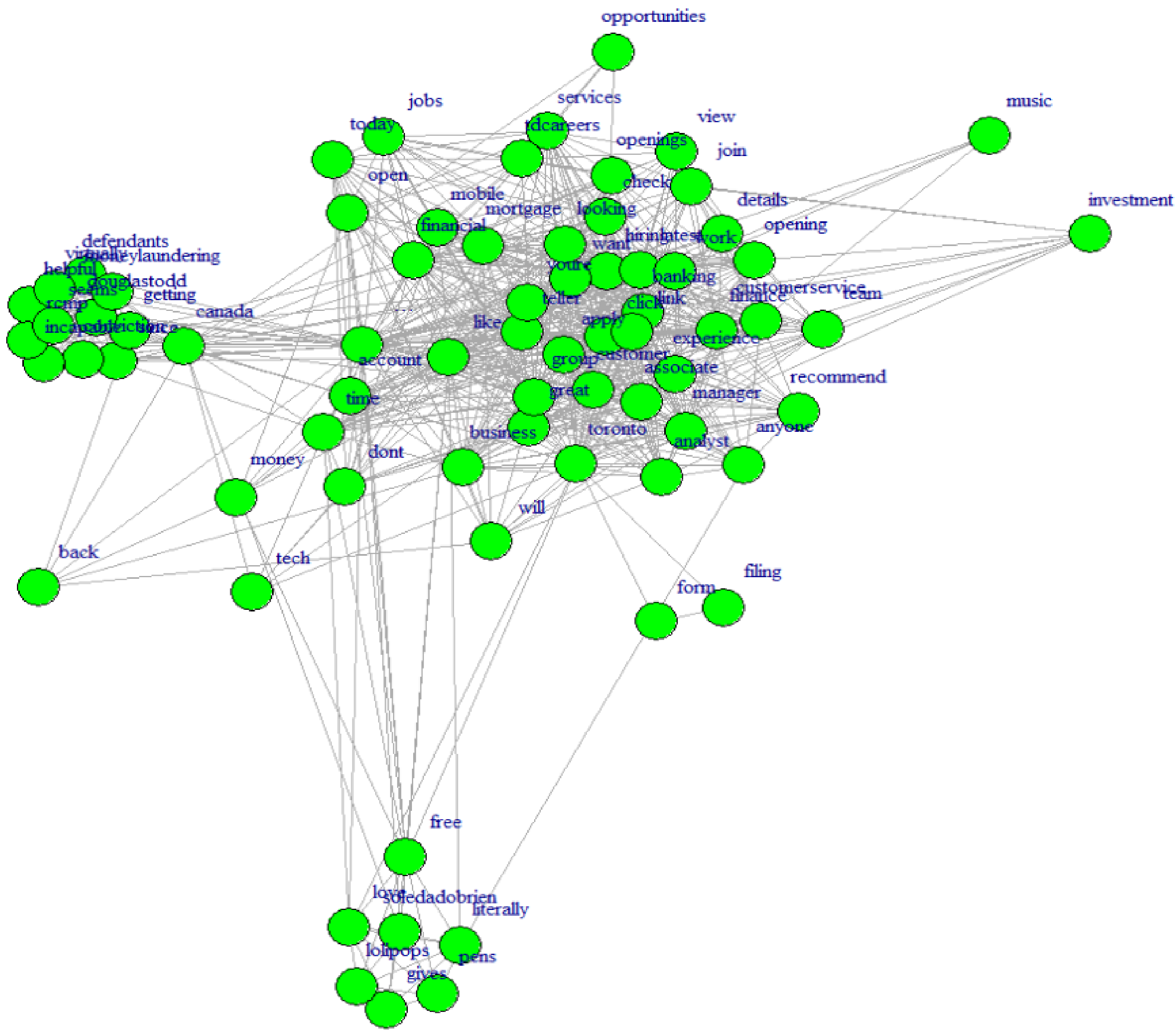
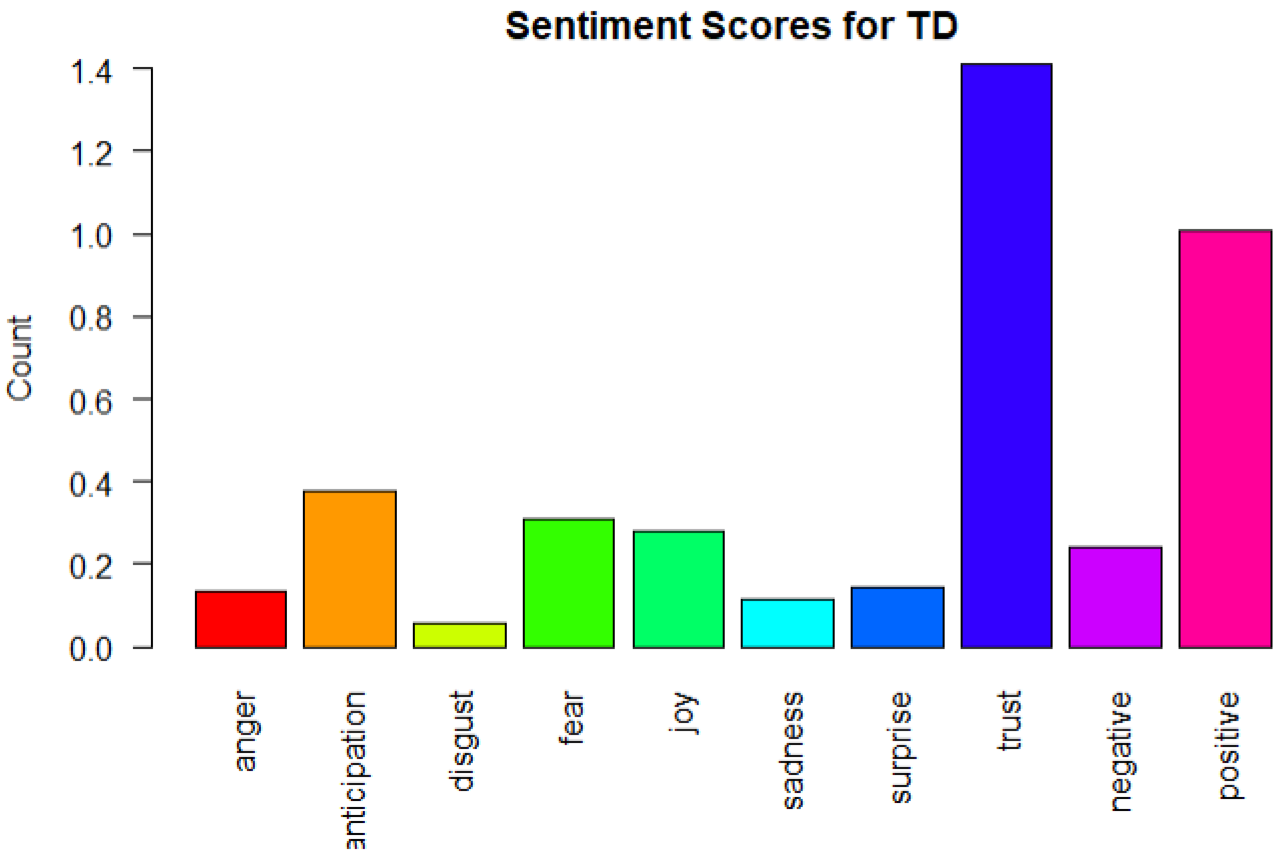
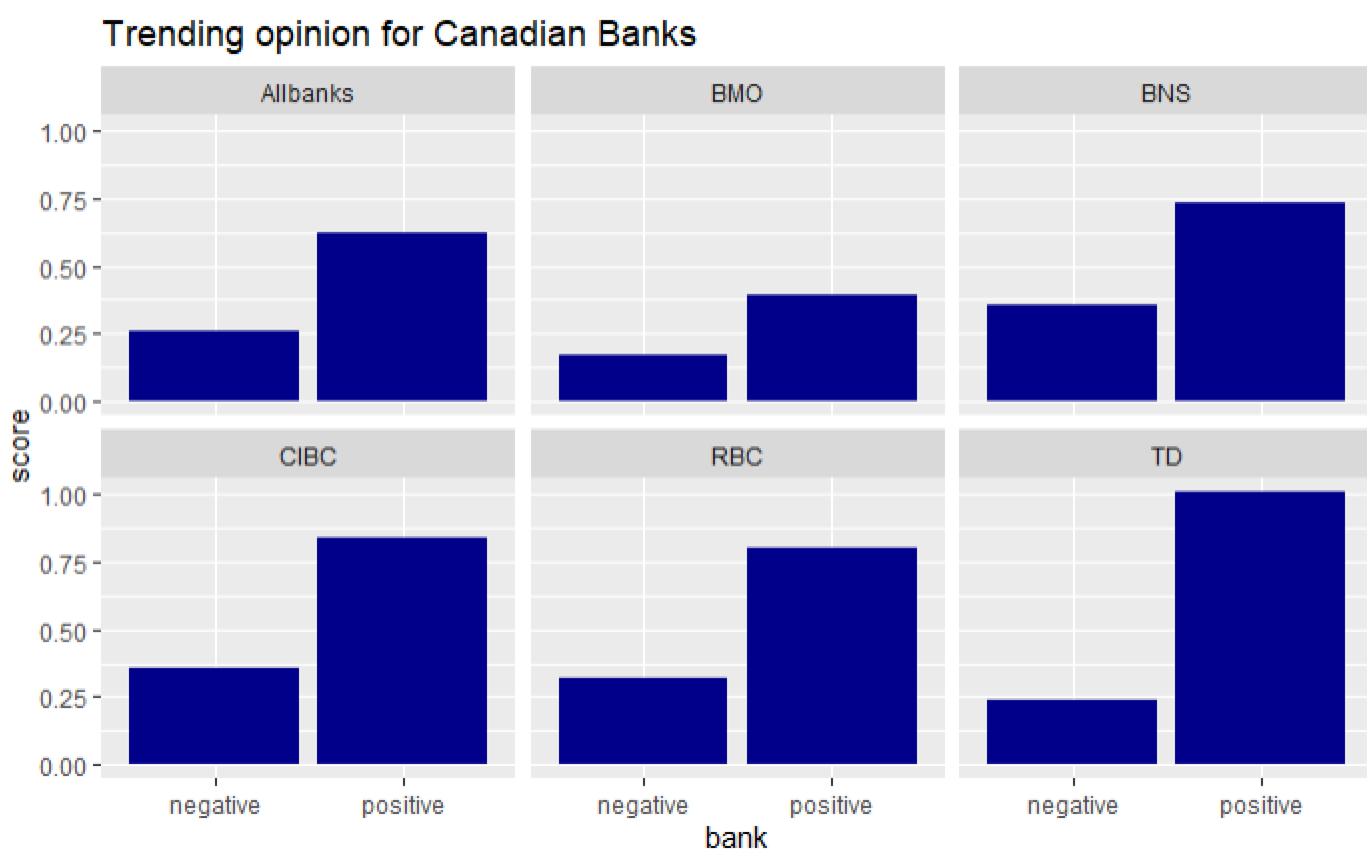
#TDcorp <- tm_map(TDcorp, gsub, pattern = 'aphria', replacement = 'aphriainc') # Replace
words

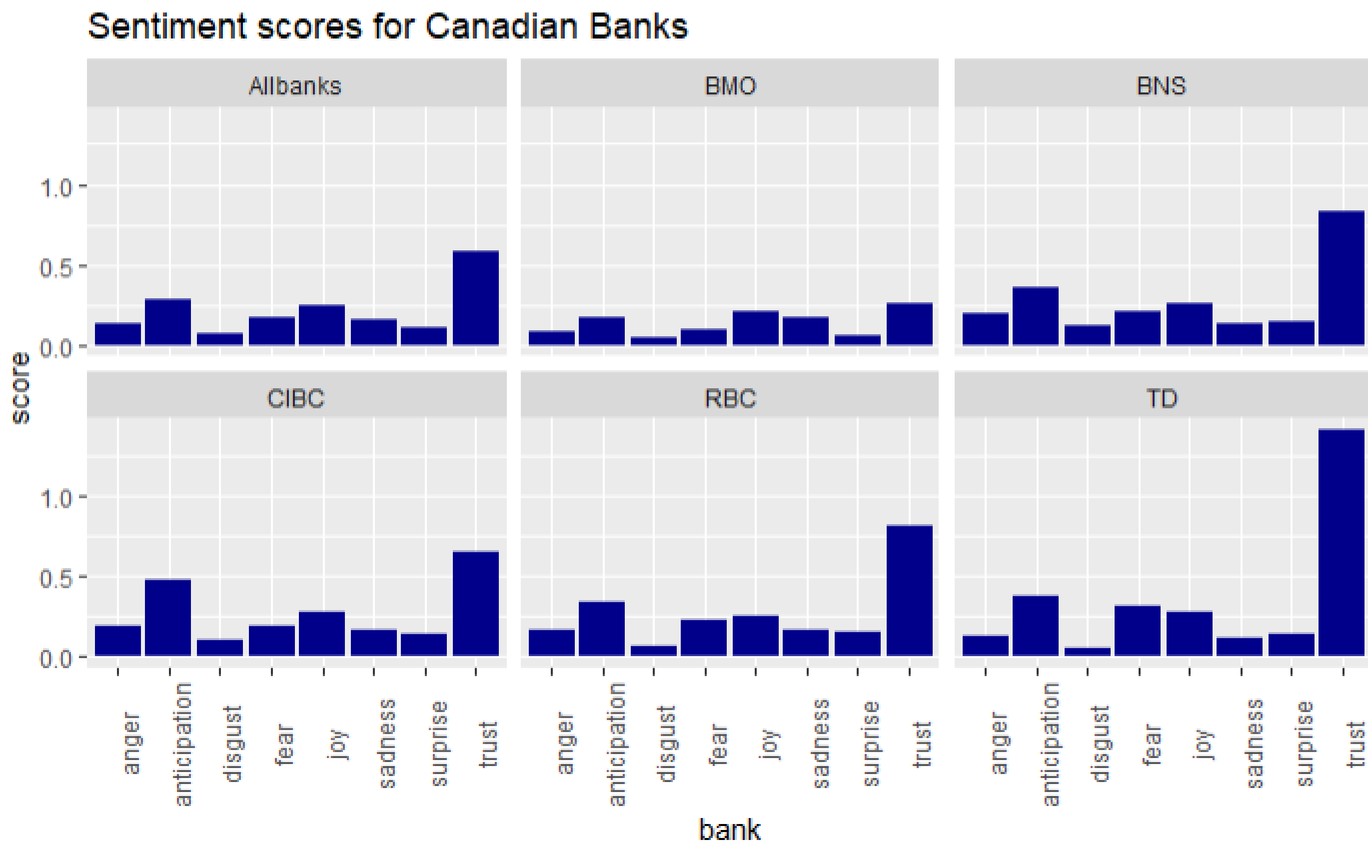
TDcorp <- tm_map(TDcorp, stripWhitespace) # remove leftover from the preceding removal

# Repeat the preceding codes
TDtdmat <- TermDocumentMatrix(TDcorp, control = list(minWordLength=c(1,Inf)))
TDtdm <- removeSparseTerms(TDtdmat, sparse=0.99)
TDmat <- as.matrix(TDtdm)
TDfreq <- rowSums(TDmat)
TDfreq <- subset(TDfreq, TDfreq>=30) # Experiment with TDfreq=? to find the optimal of of
terms in the barplot
barplot(TDfreq,
        las=2, # list all words vertically
        col = rainbow(25))
```



3.3e Clustering of terms/tweets k-means method

```
```{r}
set.seed(123)

TDm1 <- t(TDmat) # Transpose TDmat
TDk <- 23
TDkc <- kmeans(TDm1, TDk)
TDkc
```
```

```
 Within cluster sum of squares by cluster:
  [1]   21.076923 1090.292560   44.411765   39.379310  178.861538   88.727273   51.772727  609.794393
 152.854167    0.974359  161.716667   97.181818  133.980000   53.379310  107.380000   66.114286
 26.689655
 [18]   73.500000   37.500000  123.152542   58.714286  106.300000   23.750000
  (between_SS / total_SS =  44.7 %)
```

```
k   Within clusters  Between clusters
=   ===============  ================
19  200              37.3
20  194              35.7
21  166              42.5
22  176              36.1
23  138              44.7 <-
24  143              43.4
```

3.4e Plot the hierarchical clusters

```
```{r}
# Find the distance, use scale to normalise the matrix
TDdt <- dist(scale(TDmat))

TDhc <- hclust(TDdt, method = "ward.D")
plot(TDhc, hang=-1)
rect.hclust(TDhc, k=23) # 23 clusters
```
```



3.5e Find the pair of terms that appears frequently together

````
```{r}
# Term document matrix to convert unstructure text into structured for easier analysis
TDtdmat <- TermDocumentMatrix(TDcorp)
TDtdmat <- as.matrix(TDtdmat)

# Convert matrix into binary, that is whether a term appears (1) or not (0)
TDtdmat[TDtdmat>1] <- 1

# Create term-term matrix
TDttm <- TDtdmat %*% t(TDtdmat) # Multiply TDtdmat and transpose of TDtdmat
```
````

3.5.1e Find the network of terms

````
```{r}
TDg <- graph.adjacency(TDttm, weighted = T, mode = 'undirected')
TDg <- simplify(TDg) # To prevent looping of same terms
V(TDg)$label <- V(TDg)$name # Labels for the terms
V(TDg)$degree <- degree(TDg) # How often each term appears, or number of connections
between terms
TDg
```
````

```
IGRAPH 8f5dfc4 UNW- 3526 28101 --
+ attr: name (v/c), label (v/c), degree (v/n), weight (e/n)
+ edges from 8f5dfc4 (vertex names):
 [1] calls--city           calls--four            calls--reasons         calls--report          calls-
-rising          calls--superstar       calls--tech            calls--waterloo        calls--
waterlooedc
[10] calls--…              calls--help            calls--call            calls--like            calls-
-dont            calls--today           calls--number          calls--friday          calls--
havenreceived
[19] calls--'made          calls--received        calls--careful         calls--ignore          calls-
-stephkleid      calls--users–          calls--usually         calls--'…              calls--concerned
[28] city --four           city --reasons         city --report          city --rising          city -
-superstar       city --tech            city --waterloo        city --waterlooedc     city --…
[37] city --banking        city --join            city --latest          city --great           city -
-like            city --teller          city --jobs            city --open            city --asking
[46] city --dont           city --referrals       city --score           city --looking         city -
-check           city --challenge       city --drops           city --growth          city --real
[55] city --sector         city --wages           city --warns           city --windsor         city -
-windsorpoli     city --never           city --openings        city --tdcareers       city --worst
[64] city --today          city --accountempsjobs city --half          city --robert          city -
-million         city --national        city --atlantic        city --cvshealthjobs   city --health
+ ... omitted several edges
```

3.5.2e Find optimal term count for optimal (neither too busy nor sparse) visualisation of network diagram

````
```{r}
# Reset TDtdmat if needed to experiment with optimal rowSums(TDtdmat)>?
# The larger ? is, the more visible the network of terms is, but less granular
TDtdmat <- TermDocumentMatrix(TDcorp)
TDtdmat <- as.matrix(TDtdmat)

# rowSums counts the total frequency; that is, keep terms that appear > 30 times
TDtdmat <- TDtdmat[rowSums(TDtdmat)>30,]

# Re-run earlier code
TDtdmat[TDtdmat>1] <- 1 # Whenevvr TDtdmat is > 1, assign value of 1
TDtermM <- TDtdmat %*% t(TDtdmat)

TDg <- graph.adjacency(TDtermM, weighted = T, mode = 'undirected')
TDg <- simplify(TDg) # To prevent looping of same terms

V(TDg)$label <- V(TDg)$name
V(TDg)$degree <- degree(TDg)

# Histogram of node degree
hist(V(TDg)$degree,
     breaks = 100, # how many bars
     col = 'green',
     main = 'Histogram of Node Degree',
     ylab = 'Frequency',
     xlab = 'Degree of Vertices')
```
````



3.5.3e Plot the network of terms diagram

```
```{r}
# Network diagram
plot(TDg)
plot(TDg,
     vertex.color='green',
     vertex.size = 8, # can experiment with this
     vertex.label.dist = 1.5)
```
```



Discussions about "defendants-laundering-rcmp-douglastodd"; "lollipos-gives-pens-literally-free" (maybe there is a promotional event going on)

3.6e Word cloud

```{r}
TDtdmat <- TermDocumentMatrix(TDcorp)
TDtdmat <- as.matrix(TDtdmat)

TDw <- sort(rowSums(TDtdmat), decreasing = TRUE)
wordcloud(words = names(TDw),
          freq = TDw,
          max.words = 150,
          random.order = F,
          min.freq = 5,
          colors = brewer.pal(8, 'Dark2'),
          scale = c(5, 0.3),
          rot.per = 0.7)
```

3.7e Sentiment analysis

```{r}
TDsen <- get_nrc_sentiment(TDtext)

TD_sen_df <- t(as.data.frame(colMeans(TDsen)))
write.table(TD_sen_df,"/Users/sgchr/Documents/CSDA1050/Data/Allbanks_sen_df.csv",
append=T, row.names=F, col.names=F,  sep=",")

barplot(colMeans(TDsen),
        las = 2,
        col = rainbow(10),
        ylab = 'Count',
        main = 'Sentiment Scores for TD')
```



4. Addressing the Research Questions

To recap, here are the revised research questions after Sprint #1 exploratory data analysis

1. Which bank has the most favourable / unfavourable trending opinion?

Comments entered in Sprint #1: About 1,623 tweets have been collected since July 7th, 2019, with close 5,500 terms. The collection will increase in the next several weeks. It should be feasible to answer this research question. The main drawback is for the low count of CIBC tweets (40 tweets) versus that of Scotia Bank (661 tweets). The wide difference will skew the analysis, especially that of CIBC's

2. What are the current financial products being discussed?

Comments entered in Sprint #1: The EDA shows that frequent terms related to banking products are generic ones, for example, stock, charges, account. Unless we have a much more collection of tweets, it will be difficult to objectively address this research question

3. What are the current emotions (anger, fear, anticipation, trust, surprise, sadness, joy, and disgust) towards each bank?

Comments entered in Sprint #1: Not shown in this Sprint#1 report as the codes are still experimental, the author managed to "see" these emotional terms at the AllBanks level. Again, due to the low tweet count for CIBC, it may be difficult to pin down the sentiments, especially for these 8 sentiment categories.

4. What are the current sentiments towards trending financial product segments / categories (and the general network of terms being tweeted)?

Comments entered in Sprint #1: As stated above, frequent terms related to banking products are generic ones, hence it will be difficult to assess sentiments towards product segments. Network of terms is certainly a possibility

4.1 Data preparation

````
```{r}
# Load the archived sentiment and favourable (positive) and unfavourable (negative) scores
SEN_csv = read.csv("/Users/sgchr/Documents/CSDA1050/Data/Allbanks_sen_df.csv", header = TRUE)

# Create a vector containing bank names in sequence of their respective sections (3 to 3e). For example, we started with "Allbanks" and ended with "TD"
banknames = c("Allbanks", "CIBC", "BMO", "BNS", "RBC", "TD")
SEN_csv$Bank = banknames

# Split the table into "sentiment" score table and "positive/negative" score table
sent = SEN_csv[,-9:-10]
pos_neg = SEN_csv[,-1:-8]

# "sent" and "pos_neg" are wide-format, which is difficult for R to process
# Convert into long-format
sent_score = gather(sent, sentiment, score,1:8)
pos_neg_score = gather(pos_neg, pos_neg, score,1:2)
```
````

4.2 Which bank has the most favourable / unfavourable trending opinion

````
```{r}
# Visualise
ggplot(pos_neg_score, aes(pos_neg, score)) + geom_bar(stat = "identity", fill= "darkblue") + scale_x_discrete("bank") + labs(title = "Trending opinion for Canadian Banks") + facet_wrap( ~ Bank)
```
````



It is apparent that TD has the most favourable trending opinion. Inspection of the most frequent terms used for TD in sections 3.1e and 3.2e suggest TD's favourable opinion is helped by the following frequent terms: hiring, fit, like, money, love

The most unfavourable trending opinions are towards CIBC and Scotia Bank (BNS). Inspection of the most frequent terms used for CIBC and BNS in sections 3.1a and 3.1c suggest the impact of the following negative terms:

For CIBC: away, last, dont, sorry
For BNS: robbery, hasnt, strange

The mitigating factor is that the postive opinion of both BNS and CIBC are higher than the average for Allbanks. Even though, four of the five nabks have higher than average positive opinion, both CIBC and BNS have higher than average opninion - it is because of the enormous BMO tweet count (more than 13,400, about 45% of total tweets of all banks), and BMO has a low positive rating

4.3 What are the current financial products being discussed?

For all banks (sections 2.5 and 2.9), the top financial products are stocks and investments. "Capital" is also frequently discussed term which could be attributed to business products

For CIBC (sections 3.2a and 3.6a), they are gundy, stocks, mortgage, capital (maybe related to business product), debit, credit

For BMO (sections 3.2b and 3.6b), they are capital and investments. BMO has many terms relating to music and video, hence these may cloud the financial products discussion

For BNS (sections 3.2c and 3.6c), they are credit and business. Like BMO, who sponsors the BMO Field, BNS sponsors the Arena; hence the events at the Arena may cloud the terms relating to financial products

For RBC (sections 3.2d and 3.6d), capital, investments, stocks. There is an interest term "energy" in the Word cloud. Perhaps, RBC is growing interest in the "energy" sector

For TD (sections 3.2e and 3.6e), mortgage. There are many job postings in TD tweets, hence, these may cloud the related financial product terms

4.4 What are the current emotions (anger, fear, anticipation, trust, surprise, sadness, joy, and disgust) towards each bank?

````
```{r}
# Visualise
ggplot(sent_score, aes(sentiment, score)) + geom_bar(stat = "identity", fill= "darkblue")
+ scale_x_discrete("bank") + labs(title = "Sentiment scores for Canadian Banks") +
facet_wrap( ~ Bank) + theme(axis.text.x=element_text(angle=90))
```
````



1. Anger: Most Anger is towards BNS and least is towards BMO

2. Anticipation: Most Anticipation is towards CIBC and least is towards BNS. Anticpation is typically related to earnings release

3. Disgust: Most is BNS and the least is BMO

4. Fear: Most is TD, and the least is BMO

5. Joy: Most is TD, and the least is BMO. Base on section 3.5.3e, where existence network-of-terms "lollipos-gives-pens-literally-free" suggest there maybe a joyful promotional event going on

6. Sadness: Most is CIBC, and the least is TD. For CIBC, it may be attributed to sad discussion of impact of cancer (CIBC is a major sponsor of the Run For The Cure for cancer). Not usrprising, more joyful tweets in TD (see preceding note) will suppress the sadness

7. Surprise: Most is RBC, least is BMO

8. Trust: Most is TD, least is BMO. The initial hunch is that people are tweeting TD Canada "Trust". But inspection of the most frequent term used for TD in section 3.1e showed "trust" is not in the frequent term list. So this results does not appear to be fluke.

4.5 What are the current sentiments towards trending financial product segments / categories (and the general network of terms being tweeted)?
I used the cluster dendrogram and network of terms to do this. For all banks, they are found in sections 2.7 and 2.8 respectively. For the individual banks, they are found in sections 3.4 and 3.5.3 respectively

1. All banks
Cluster dendrogram - Nothing significant

Network of terms - Nothing significant
Nothing much could be seen at "all banks", perhaps due to the enormous non-financial products related tweets for BMO (who sponsored the BMO field) and BNS (who sponsored the Arena), and therefore lots of tweets relating to events there

2. CIBC
Cluster dendrogram
- cluster of good-experience (although there is a sub cluster of sorry-noticed-tweet)
- cluster of mortgage-mobile-online-back-great

Network of terms diagram (nothing significant)

CIBC has the least tweets for all the five banks. It does sponsor the Run For The Cure (RFTC) for cancer, but the RFTC tweets does significantly cloud the analysis

3. BMO
Cluster dendrogram (nothing significant)
Network of terms diagram (nothing significant)

4. BNS
Cluster dendrogram - cluster of july-credit-mortgage-since-hasn't

Network of terms diagram (nothing significant)

5. RBC
Cluster dendrogram - cluster of back-team-investment-kids-raised-like

Network of terms diagram (nothing significant)

RBC has many tweets about job opportunities that would have clouded this analysis

6. TD
Cluster dendrogram (nothing significant)

Network of terms diagram (nothing significant)

Like RBC, TD has many tweets about job opportunities that would have clouded this analysis