Text Mining in Data Science

A tutorial of Text Mining in R Using TM Package

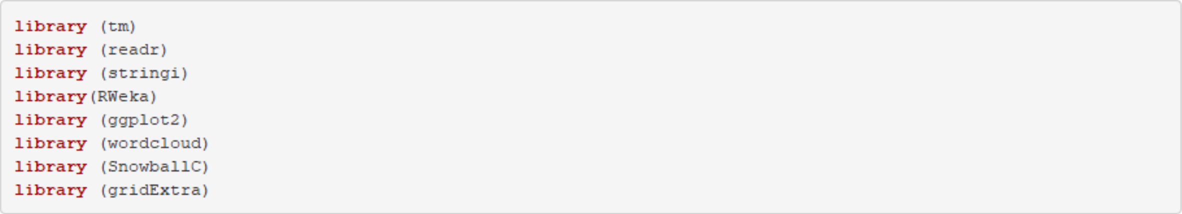
Among all things for the people working on Data Analytics, one thing they will surely come across is Data Mining. Data Mining is all about examining huge to extremely huge amount of structured and unstructured data to form actionable insights.

This article is your guide to get started with Text Mining in R using TM package. It explains enormous power that R and its packages have to offer on Text Mining. A person with elementary R knowledge can use this article to get started with Text Mining. It guides user till exploratory data analysis and N-Grams generation.

Important Terms:
Before we dig dip into Text Mining, let's understand some of the important concepts related to Text Mining.

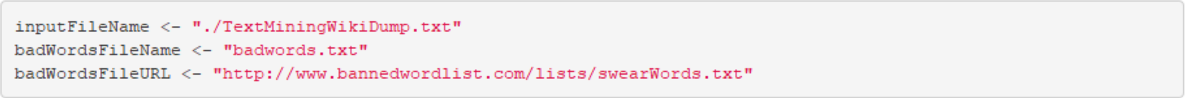
a. **TM package**: R package for Text Mining
b. **Corpus & Corpora**: Corpus is a large collection of text. It is a body of written or spoken material upon which a linguistic analysis is based. Plural form of Corpus is Corpora which essentially is collections of documents containing natural language text.
c. **Document Term Matrix (DTM)**: A Document Term Matrix is a mathematical matrix that describes the frequency of terms that occur in a collection of documents. It has documents in rows and word frequencies in columns.
d. **Stemming**: Stemming is the process of converting words into their basis form making it easier for analysis e.g. Words like win, winning and winner are converted and counted to their basic form i.e. win.
e. **Stop Words**: These are most common words in a language that get repeated. However, they add little value to text mining e.g. I, our, they'll, etc. There are 174 stop words in English.
f. **Bad Words**: These are offensive words which need to be removed before we start data mining.

Step 1: Install & load necessary libraries. Out of these, TM is R's text mining package. Other packages are supplementary packages that are used for reading lines from file, plotting, preparing word clouds, N-Gram generation, etc.



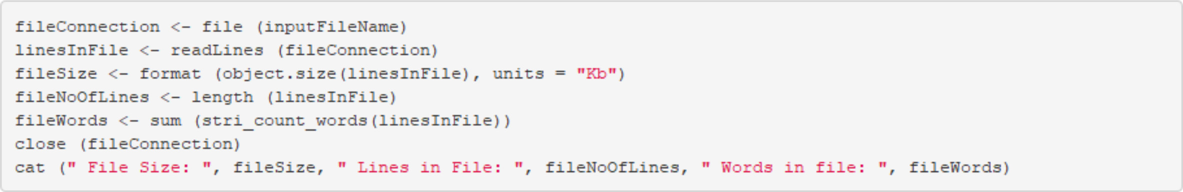
```r
library (tm)
library (readr)
library (stringi)
library(RWeka)
library (ggplot2)
library (wordcloud)
library (SnowballC)
library (gridExtra)
```

Set constants to be used multiple times:

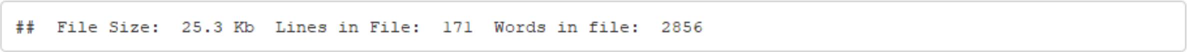


```r
inputFileName <- "./TextMiningWikiDump.txt"
badWordsFileName <- "badwords.txt"
badWordsFileURL <- "http://www.bannedwordlist.com/lists/swearWords.txt"
```

Step 2: Read text file contents. Optional - Gather and display basic file attributes viz. file size, number of lines in file, number of words in file.

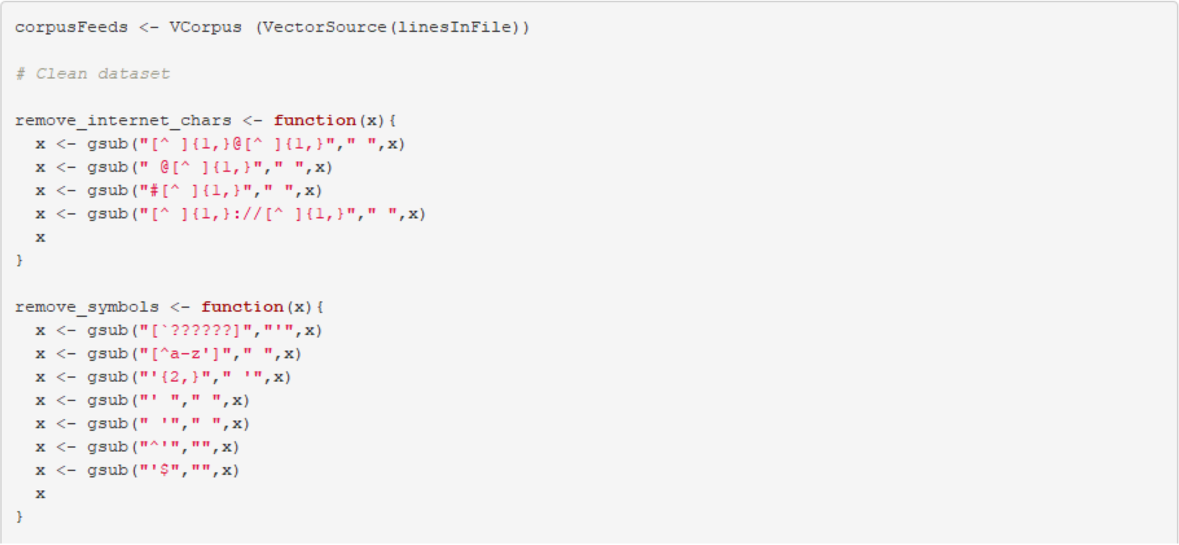


```
fileConnection <- file (inputFileName)
linesInFile <- readLines (fileConnection)
fileSize <- format (object.size(linesInFile), units = "Kb")
fileNoOfLines <- length (linesInFile)
fileWords <- sum (stri_count_words(linesInFile))
close (fileConnection)
cat (" File Size: ", fileSize, " Lines in File: ", fileNoOfLines, " Words in file: ", fileWords)
```

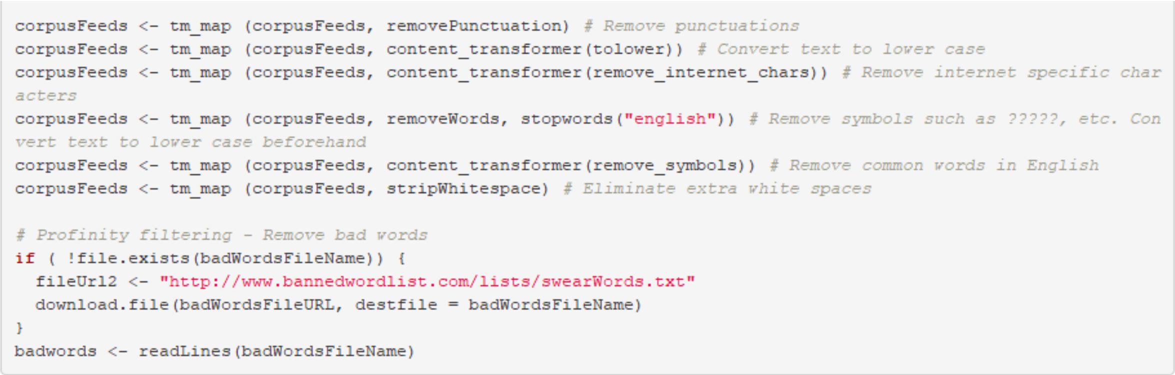


```
##  File Size:  25.3 Kb  Lines in File:  171  Words in file:  2856
```

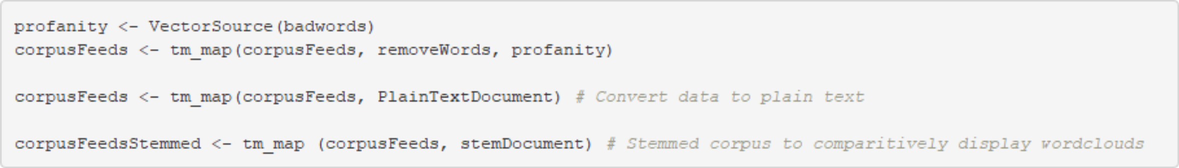
Step 3: Create file corpus, clean the corpus



```
corpusFeeds <- VCorpus (VectorSource(linesInFile))

# Clean dataset

remove_internet_chars <- function(x){
  x <- gsub("[^ ]{1,}@[^ ]{1,}"," ",x)
  x <- gsub(" @[^ ]{1,}"," ",x)
  x <- gsub("#[^ ]{1,}"," ",x)
  x <- gsub("[^ ]{1,}://[^ ]{1,}"," ",x)
  x
}

remove_symbols <- function(x){
  x <- gsub("[`??????]","'",x)
  x <- gsub("[^a-z']"," ",x)
  x <- gsub("'{2,}"," '",x)
  x <- gsub("' "," ",x)
  x <- gsub(" '"," ",x)
  x <- gsub("^'","",x)
  x <- gsub("'$","",x)
  x
}
```

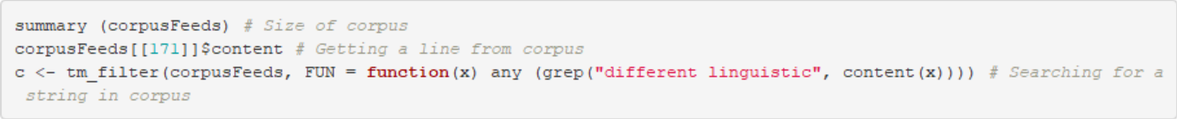


```
corpusFeeds <- tm_map (corpusFeeds, removePunctuation) # Remove punctuations
corpusFeeds <- tm_map (corpusFeeds, content_transformer(tolower)) # Convert text to lower case
corpusFeeds <- tm_map (corpusFeeds, content_transformer(remove_internet_chars)) # Remove internet specific char
acters
corpusFeeds <- tm_map (corpusFeeds, removeWords, stopwords("english")) # Remove symbols such as ?????, etc. Con
vert text to lower case beforehand
corpusFeeds <- tm_map (corpusFeeds, content_transformer(remove_symbols)) # Remove common words in English
corpusFeeds <- tm_map (corpusFeeds, stripWhitespace) # Eliminate extra white spaces

# Profinity filtering - Remove bad words
if ( !file.exists(badWordsFileName)) {
  fileUrl2 <- "http://www.bannedwordlist.com/lists/swearWords.txt"
  download.file(badWordsFileURL, destfile = badWordsFileName)
}
badwords <- readLines(badWordsFileName)
```



```
profanity <- VectorSource(badwords)
corpusFeeds <- tm_map(corpusFeeds, removeWords, profanity)

corpusFeeds <- tm_map(corpusFeeds, PlainTextDocument) # Convert data to plain text

corpusFeedsStemmed <- tm_map (corpusFeeds, stemDocument) # Stemmed corpus to comparitively display wordclouds
```

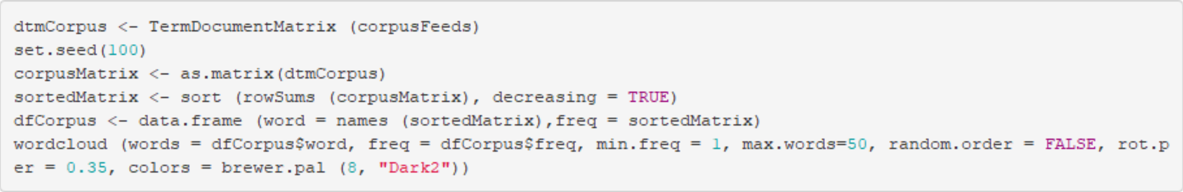
Step 4: This step illustrates few basic exploratory data analysis steps that can act as reference for detailed exploratory data analysis.



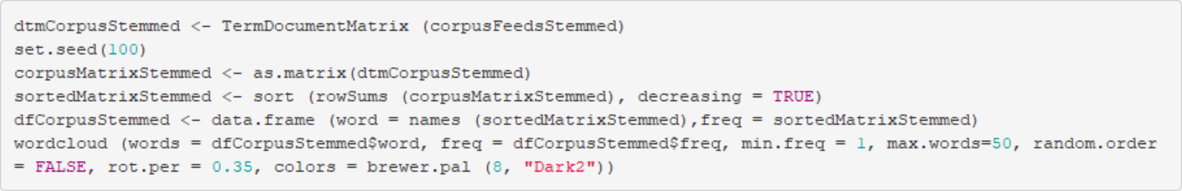
```
summary (corpusFeeds) # Size of corpus
corpusFeeds[[171]]$content # Getting a line from corpus
c <- tm_filter(corpusFeeds, FUN = function(x) any (grep("different linguistic", content(x)))) # Searching for a
 string in corpus
```

Output is not shown.

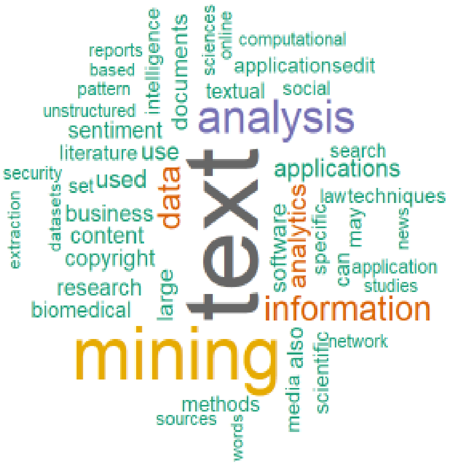
Step 5: Visualize frequency of words occurring in text file by using word clouds. Following code snippet generates two word clouds to show un-stemmed and stemmed corpus word clouds:

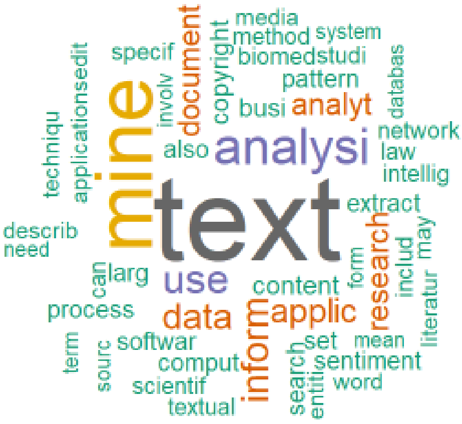


```
dtmCorpus <- TermDocumentMatrix (corpusFeeds)
set.seed(100)
corpusMatrix <- as.matrix(dtmCorpus)
sortedMatrix <- sort (rowSums (corpusMatrix), decreasing = TRUE)
dfCorpus <- data.frame (word = names (sortedMatrix),freq = sortedMatrix)
wordcloud (words = dfCorpus$word, freq = dfCorpus$freq, min.freq = 1, max.words=50, random.order = FALSE, rot.p
er = 0.35, colors = brewer.pal (8, "Dark2"))
```



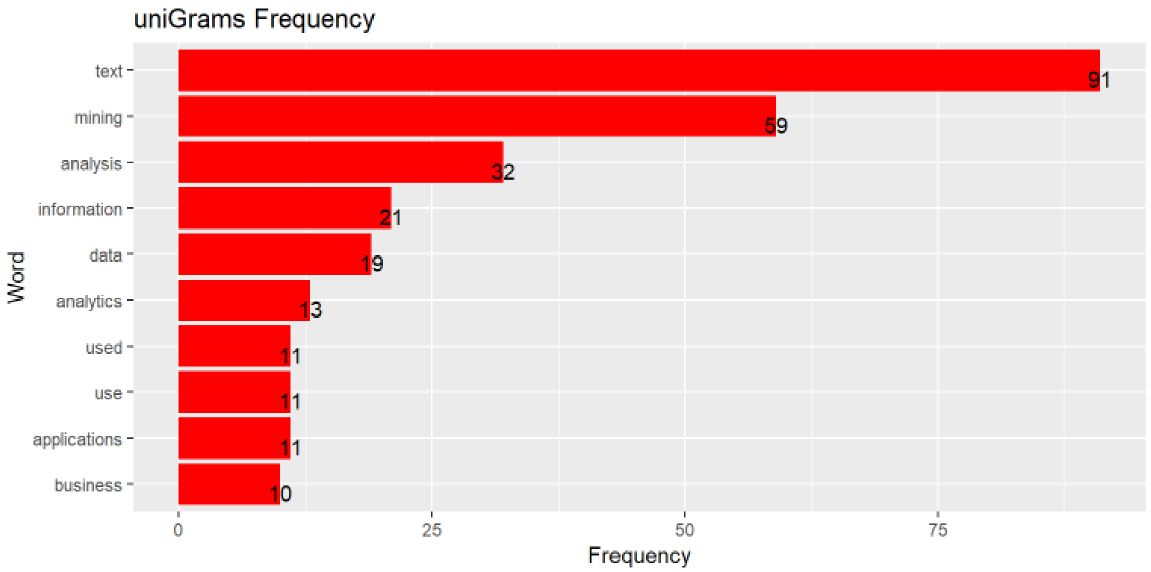
```
dtmCorpusStemmed <- TermDocumentMatrix (corpusFeedsStemmed)
set.seed(100)
corpusMatrixStemmed <- as.matrix(dtmCorpusStemmed)
sortedMatrixStemmed <- sort (rowSums (corpusMatrixStemmed), decreasing = TRUE)
dfCorpusStemmed <- data.frame (word = names (sortedMatrixStemmed),freq = sortedMatrixStemmed)
wordcloud (words = dfCorpusStemmed$word, freq = dfCorpusStemmed$freq, min.freq = 1, max.words=50, random.order
= FALSE, rot.per = 0.35, colors = brewer.pal (8, "Dark2"))
```

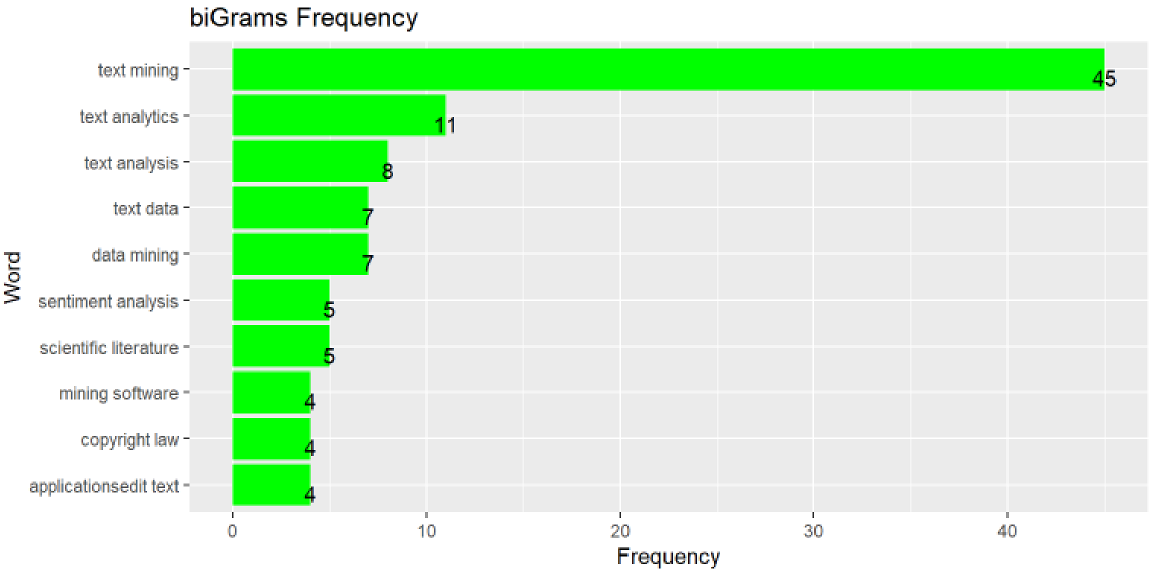


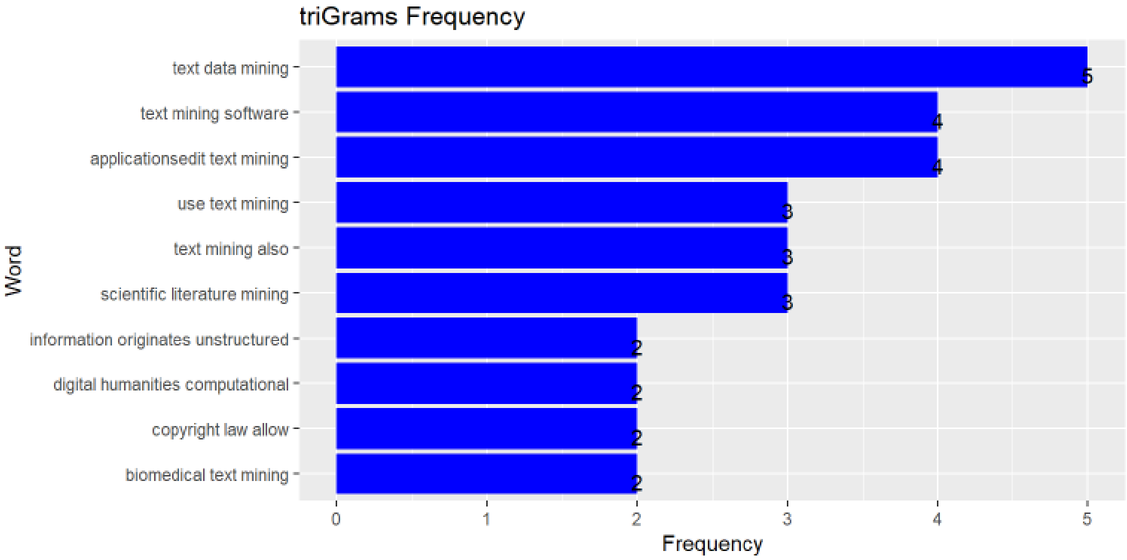


Step 6: Last step of this guide is to generate N-Grams (uni, bi and tri grams) and plot histograms of top 10 occurring N-Grams.

```
# N-Grams and Histograms

dataFrameForNGrams <- data.frame (text = sapply (corpusFeeds, as.character), stringsAsFactors = FALSE)
uniGramToken <- NGramTokenizer (dataFrameForNGrams, Weka_control (min=1, max=1))
biGramToken <- NGramTokenizer (dataFrameForNGrams, Weka_control(min=2, max=2))
triGramToken <- NGramTokenizer (dataFrameForNGrams, Weka_control(min=3, max=3))

uniGrams <- data.frame(table(uniGramToken))
biGrams <- data.frame(table(biGramToken))
triGrams <- data.frame(table(triGramToken))

uniGrams <- uniGrams[order(uniGrams$Freq, decreasing=TRUE),]
colnames(uniGrams) <- c ("Word", "Frequency")
biGrams <- biGrams[order(biGrams$Freq, decreasing=TRUE),]
colnames(biGrams) <- c ("Word", "Frequency")
triGrams <- triGrams[order(triGrams$Freq, decreasing=TRUE),]
colnames(triGrams) <- c ("Word", "Frequency")

uniGrams_s <- uniGrams [1:10,]
biGrams_s <- biGrams [1:10,]
triGrams_s <- triGrams [1:10,]

plotUniGram = ggplot (uniGrams_s, aes (x = reorder (Word, Frequency), y = Frequency)) + geom_bar(stat="identity
", fill="red") +
  geom_text (aes(y = Frequency, label = Frequency), vjust = 1) + coord_flip() + labs (x = "Word", y = "Frequenc
y", title = "uniGrams Frequency")

plotBiGram = ggplot (biGrams_s, aes (x = reorder (Word, Frequency), y = Frequency)) + geom_bar(stat="identity",
 fill="green") +
  geom_text (aes(y = Frequency, label = Frequency), vjust = 1) + coord_flip() + labs(x = "Word", y = "Frequency
", title = "biGrams Frequency")

plotTriGram = ggplot (triGrams_s, aes (x = reorder (Word, Frequency), y=Frequency)) + geom_bar(stat="identity",
 fill="blue") +
  geom_text (aes(y=Frequency, label = Frequency), vjust=1) + coord_flip() + labs (x = "Word", y = "Frequency",
title = "triGrams Frequency")


grid.arrange (plotUniGram, plotBiGram, plotTriGram, nrow = 3)
```







Further steps could be use above generated N-Grams text mining activities like word predictions, etc.

References:

a. TM package - https://cran.r-project.org/web/packages/tm/tm.pdf
b. Corpus & Corpora - http://language.worldofcomputing.net/linguistics/introduction/what-is-corpus.html
c. Text used in this guide is text of following WIKI page - https://en.wikipedia.org/wiki/Text_mining