**Week 14 Assignment**

**PART 2: Corpus/Term Document Matrix & Identifying KEIs**

**Company used for analysis-**

Coca-cola - Beverages

**'Q&A with chairman and CEO'** section of coca-cola company is analyzed as it provides some insights of company's growth and other events/activities which is helpful information for shareowners. Also, we have analyzed **Q&A section on proxy materials** to get more information about annual meetings.

**Below code is to analyze 'Q&A with chairman and CEO' -**

Below are the pre-requisites for execution of document term matrix code :

Installed below required packages and assigned libraries

#Installation of packages

install.packages("tm")

install.packages("wordcloud")

install.packages("ggplot2")

#Assign library

library(RCurl)

library(XML)

library(tm)

library(magrittr)

library(ggplot2)

**Creation of corpus**

#Store all file paths in the list

urls <- c("E:/Sonali\_MS/ITMD529/data\_mining/cocacola/Cocacola\_2016.htm",

"E:/Sonali\_MS/ITMD529/data\_mining/cocacola/Cocacola\_2015.html",

"E:/Sonali\_MS/ITMD529/data\_mining/cocacola/Cocacola\_2014.html",

"E:/Sonali\_MS/ITMD529/data\_mining/cocacola/Cocacola\_2013.html",

"E:/Sonali\_MS/ITMD529/data\_mining/cocacola/Cocacola\_2012.html",

"E:/Sonali\_MS/ITMD529/data\_mining/cocacola/Cocacola\_2011.html")

)

plainTxt <- list()

#Remove all xml tags and parse html

for(i in urls){

html.files <- readLines(urls[i])

html = htmlTreeParse(html.files, useInternal=TRUE)

plainTxt[i] <- toString(xpathApply(html, "//body//text()

[not(ancestor::script)][not(ancestor::style)]", xmlValue))

}

#Copy file into temp file

cat(plainTxt[[1]], file="E:/Sonali\_MS/ITMD529/data\_mining/cocacola/temp.txt", sep="n", append = FALSE)

cocacola.files <-readLines("E:/Sonali\_MS/ITMD529/data\_mining/cocacola/temp.txt")

#Get the start and end of QA sections for all files and crop it

#----------------------------------------------------------

qa\_sentence\_start <- "Q&A WITH OUR CHAIRMAN"

grep(qa\_sentence\_start, cocacola.files, ignore.case = TRUE)

qa\_sentence\_end <- "ANNUAL MEETING OF SHAREOWNERS"

grep(qa\_sentence\_end, cocacola.files, ignore.case = TRUE)

QA.Section\_2016 <- cocacola.files[404:602]

qa\_sentence\_start <- "Q&A FROM OUR CHAIRMAN"

grep(qa\_sentence\_start, cocacola.files, ignore.case = TRUE)

qa\_sentence\_end <- "VOTING INFORMATION"

grep(qa\_sentence\_end, cocacola.files, ignore.case = TRUE)

QA.Section\_2015 <- cocacola.files[471:653]

qa\_sentence\_start <- "Q&A WITH OUR CHAIRMAN"

grep(qa\_sentence\_start, cocacola.files, ignore.case = TRUE)

qa\_sentence\_end <- "ANNUAL MEETING OF SHAREOWNERS"

grep(qa\_sentence\_end, cocacola.files, ignore.case = TRUE)

QA.Section\_2014 <- cocacola.files[366:602]

#---------------------------------------------------------

#Store data into text files

cat(QA.Section\_2016, file="E:/Sonali\_MS/ITMD529/data\_mining/cocacola//qa\_section/qa\_2016.txt", sep="n", append = FALSE)

cat(QA.Section\_2015, file="E:/Sonali\_MS/ITMD529/data\_mining/cocacola//qa\_section/qa\_2015.txt", sep="n", append = FALSE)

cat(QA.Section\_2014, file="E:/Sonali\_MS/ITMD529/data\_mining/cocacola//qa\_section/qa\_2014.txt", sep="n", append = FALSE)

cat(QA.Section\_2013, file="E:/Sonali\_MS/ITMD529/data\_mining/cocacola//qa\_section/qa\_2013.txt", sep="n", append = FALSE)

cat(QA.Section\_2012, file="E:/Sonali\_MS/ITMD529/data\_mining/cocacola//qa\_section/qa\_2012.txt", sep="n", append = FALSE)

cat(QA.Section\_2011, file="E:/Sonali\_MS/ITMD529/data\_mining/cocacola//qa\_section/qa\_2011.txt", sep="n", append = FALSE)

#Create corpus of files

cname <- "E:/Sonali\_MS/ITMD529/data\_mining/cocacola/qa\_section"

qa\_sections<-Corpus(DirSource(cname))

summary(qa\_sections)

#View doc using function

viewDocs <- function(d,n) {d %>% extract2(n) %>% as.character() %>% writeLines()}

viewDocs(qa\_sections, 3)

**Creation of document term matrix**

#Apply required functions

#To lower case:

qa\_sections <- tm\_map(qa\_sections, content\_transformer(tolower))

#Remove Numbers:

qa\_sections <- tm\_map(qa\_sections, removeNumbers)

#Remove Stop Words:

qa\_sections <- tm\_map(qa\_sections, removeWords, stopwords("english"))

viewDocs(qa\_sections, 1)

#Check the stopwords

length(stopwords("english"))

stopwords("en")

#Replacing certain expressions with spaces:

toSpace <- content\_transformer(function(x, pattern) gsub(pattern, " ",x))

qa\_sections <- tm\_map(qa\_sections, toSpace, "/|<|>|”|=|@|\\|:|;|-|--\"")

viewDocs(qa\_sections, 1)

#Strip white spaces:

qa\_sections <- tm\_map(qa\_sections, stripWhitespace)

viewDocs(qa\_sections, 1)qa\_sentence\_start <- "Q&A WITH OUR CHAIRMAN"

grep(qa\_sentence\_start, cocacola.files, ignore.case = TRUE)

qa\_sentence\_end <- "ANNUAL MEETING OF SHAREOWNERS"

grep(qa\_sentence\_end, cocacola.files, ignore.case = TRUE)

QA.Section\_2014 <- cocacola.files[366:602]

#---------------------------------------------------------

#Store data into text files

cat(QA.Section\_2016, file="E:/Sonali\_MS/ITMD529/data\_mining/cocacola//qa\_section/qa\_2016.txt", sep="n", append = FALSE)

cat(QA.Section\_2015, file="E:/Sonali\_MS/ITMD529/data\_mining/cocacola//qa\_section/qa\_2015.txt", sep="n", append = FALSE)

cat(QA.Section\_2014, file="E:/Sonali\_MS/ITMD529/data\_mining/cocacola//qa\_section/qa\_2014.txt", sep="n", append = FALSE)

#Create corpus of files

cname <- "E:/Sonali\_MS/ITMD529/data\_mining/cocacola/qa\_section"

qa\_sections<-Corpus(DirSource(cname))

summary(qa\_sections)

#View doc using function

viewDocs <- function(d,n) {d %>% extract2(n) %>% as.character() %>% writeLines()}

viewDocs(qa\_sections, 3)

#Remove Punctuations:

qa\_sections <- tm\_map(qa\_sections, removePunctuation)

#Punctuation characters: ! " # $ % & ' ( ) \* + , - . / : ; < = > ? @ [ \ ] ^ \_ ` { | } ~.

#Strip white spaces:

qa\_sections <- tm\_map(qa\_sections, stripWhitespace)

viewDocs(qa\_sections, 1)

#Remove non-required words

qa\_sections <- tm\_map(qa\_sections, removeWords, c("n", "â€", "a", "i", "â", "nn","nj","k","the","address","variety","questions",

"can", "also", "e", "mail", "via","indicate","named","page","going","section", "currently","place","degree", "said","way","every","many","various", "with","following","either","using","will","ivs", "comments","commonly","across","includes",

"if", "may", "help", "us", "will","well","date","provided","many","around","begins", "please", "unless", "visit","need","close", "also","preferences","persons","thinking","nnn", "address","pm","eastern","time","see","plus","still","ahead","topics","account",

"break","always","active","actions","achieve","adopt","added","actionsnwe","accessnthere","accessnas","little", "lot","made", "make","making","managed","makes","looking","look","line","like","light","momentum","much","received","recently","restore","including","back","contents","addition","additional","combined", "plain","english","read","year","value","one","companyâ€™s","serve","believe","ensure"))

#Create term doc matrix

qa\_dtm <- DocumentTermMatrix(qa\_sections)

qa\_dtm

inspect(qa\_dtm[1:6, 650:690])

#Store sum of columns

freq <- colSums((as.matrix(qa\_dtm)))

length(freq)

#Sort the frequency table

ord <- order(freq)

#Check the lower and higher freqency words

freq[head(ord)]

freq[tail(ord,25)]

#Remove sparse terms

#Keep value as0.6 so if term appears in 2/3 of docs then it will keep the terms

dtms <- removeSparseTerms(qa\_dtm, 0.6)

dim(dtms)

inspect(dtms)

freq <-colSums(as.matrix(dtms))

#Sort the frequency table

ord <- order(freq)

#Again check the lower and higher frequency words after sparse term removal

freq[head(ord)]

freq[tail(ord,25)]

#Check higher and lower frequency terms after sparse term removal

#set the threshold looking at term matrix

findFreqTerms(dtms, lowfreq=7)

findFreqTerms(dtms, highfreq=2)

#find association with coca word and store result in table

assoc\_word<-findAssocs(dtms, "coca", corlimit = .9)

associated.terms <- data.frame(word=assoc\_word)

class(associated.terms)

associated.terms.matrix<-data.frame(word=rownames(associated.terms),value=associated.terms)

#Sort matrix of terms

freq <- sort(colSums(as.matrix(dtms)), decreasing = TRUE)

#Get the high frequent words

head(freq, 30)

wf <- data.frame(word=names(freq), freq=freq)

head(wf)

**Graph plotting**

#Plot the graph of high frequency words

library(dplyr)

subset(wf, freq>6) %>% ggplot(aes(word, freq)) + geom\_bar(stat="identity") + theme(axis.text.x=element\_text(angle=45, hjust=1))

# Generate a Word Cloud

install.packages("wordcloud")

library(wordcloud)

set.seed(123)

wordcloud(names(freq), freq, min.freq=7,random.color=TRUE,colors=rainbow(7),random.order = FALSE)

**Below code is to analyze Q&A section on proxy materials' -**

**Document term matrix with top 30 high frequency words**

*Script 1/2: HTML to text*

# Sourced from Weblink https://github.com/tonybreyal/Blog-Reference-Functions/blob/master/R/htmlToText/htmlToText.R

# Author: Tony Breyal

# Date: 2011-11-18

# Modified: 2011-11-18

# Description: Extracts all text from a webpage (aims to extract only the text you would see in a web browser)

# Packages Used: RCurl, XML

htmlToText <- function(input, ...) {

###---PACKAGES ---###

require(RCurl)

require(XML)

###--- LOCAL FUNCTIONS ---###

# Determine how to grab html for a single input element

evaluate\_input <- function(input) {

# if input is a .html file

if(file.exists(input)) {

char.vec <- readLines(input, warn = FALSE)

return(paste(char.vec, collapse = ""))

}

# if input is html text

if(grepl("</html>", input, fixed = TRUE)) return(input)

# if input is a URL, probably should use a regex here instead?

if(!grepl(" ", input)) {

# downolad SSL certificate in case of https problem

if(!file.exists("cacert.perm")) download.file(url="http://curl.haxx.se/ca/cacert.pem", destfile="cacert.perm")

return(getURL(input, followlocation = TRUE, cainfo = "cacert.perm"))

}

# return NULL if none of the conditions above apply

return(NULL)

}

# convert HTML to plain text

convert\_html\_to\_text <- function(html) {

doc <- htmlParse(html, asText = TRUE)

text <- xpathSApply(doc, "//text()[not(ancestor::script)][not(ancestor::style)][not(ancestor::noscript)][not(ancestor::form)]", xmlValue)

return(text)

}

# format text vector into one character string

collapse\_text <- function(txt) {

return(paste(txt, collapse = " "))

}

###--- MAIN ---###

# STEP 1: Evaluate input

html.list <- lapply(input, evaluate\_input)

# STEP 2: Extract text from HTML

text.list <- lapply(html.list, convert\_html\_to\_text)

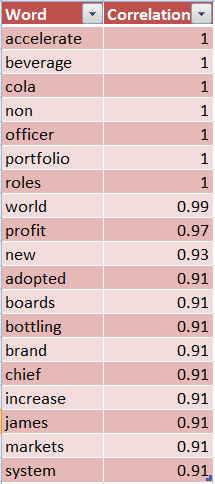
# STEP 3: Return text

text.vector <- sapply(text.list, collapse\_text)

return(text.vector)

}

**Associated words with "coca" term**

****

**Graph plotting**

*Script 2/2: Final Project*

library(RCurl)

library(XML)

library(tm)

library(NLP)

library(SnowballC)

library(magrittr)

library(ggplot2)

library(dplyr)

library(wordcloud)

# save CocaCola htm files

CocaCola2016 <- "https://www.sec.gov/Archives/edgar/data/21344/000130817916000226/lcocacola2016\_def14a.htm"

CocaCola2015 <- "https://www.sec.gov/Archives/edgar/data/21344/000130817915000041/lcocacola2015\_def14a.htm"

CocaCola2014 <- "https://www.sec.gov/Archives/edgar/data/21344/000130817914000049/lcocacola2014\_def14a.htm"

CocaCola2013 <- "https://www.sec.gov/Archives/edgar/data/21344/000130817913000057/lcocacola2013\_def14a.htm"

CocaCola2012 <- "http://www.sec.gov/Archives/edgar/data/21344/000130817912000055/lcocacola\_def14a.htm"

CocaCola2016Txt <- htmlToText(CocaCola2016)

CocaCola2015Txt <- htmlToText(CocaCola2015)

CocaCola2014Txt <- htmlToText(CocaCola2014)

CocaCola2013Txt <- htmlToText(CocaCola2013)

CocaCola2012Txt <- htmlToText(CocaCola2012)

# save the cursors for start qa section and end qa section

qa\_sentence\_start <- "What is included in the proxy materials"

qa\_sentence\_end <- "What are my voting"

pattern <- "</?\\w+((\\s+\\w+(\\s\*=\\s\*(?:\".\*?\"|'.\*?'|[^'\">\\s]+))?)+\\s\*|\\s\*)/?>"

plain.text <- gsub(pattern, "\\1", txt)

file2016 <- "C:/Users/tanmaygole/Desktop/CocaCola/CocaCola2016.txt"

write(CocaCola2016Txt, file = file2016)

file2015 <- "C:/Users/tanmaygole/Desktop/CocaCola/CocaCola2015.txt"

write(CocaCola2015Txt, file = file2015)

file2014 <- "C:/Users/tanmaygole/Desktop/CocaCola/CocaCola2014.txt"

write(CocaCola2014Txt, file = file2014)

file2013 <- "C:/Users/tanmaygole/Desktop/CocaCola/CocaCola2013.txt"

write(CocaCola2013Txt, file = file2013)

file2012 <- "C:/Users/tanmaygole/Desktop/CocaCola/CocaCola2012.txt"

write(CocaCola2012Txt, file = file2012)

# process 2016 file

# read the data from the file

Coke2016 = readLines(file2016)

# find the start and end section

grep(qa\_sentence\_start,Coke2016, ignore.case = TRUE)

grep(qa\_sentence\_end,Coke2016, ignore.case = TRUE)

# show the content of qa section

Coke2016[11697:11769]

# save the qa section

qa\_Coke2016 <- Coke2016[11697:11769]

# save the data into a new text file

cat(qa\_Coke2016, file="C:/Users/tanmaygole/Desktop/CocaCola/QA\_Coke2016.txt", sep="n", append = TRUE)

# process 2015 file

# read the data from the file

Coke2015 = readLines(file2015)

# find the start and end section

grep(qa\_sentence\_start,Coke2015, ignore.case = TRUE)

grep(qa\_sentence\_end,Coke2015, ignore.case = TRUE)

# show the content of qa section

Coke2015[13087:13184]

# save the qa section

qa\_Coke2015 <- Coke2015[13087:13184]

# save the data into a new text file

cat(qa\_Coke2015, file="C:/Users/tanmaygole/Desktop/CocaCola/QA\_Coke2015.txt", sep="n", append = TRUE)

# process 2014 file

# read the data from the file

Coke2014 = readLines(file2014)

# find the start and end section

grep(qa\_sentence\_start,Coke2014, ignore.case = TRUE)

grep(qa\_sentence\_end,Coke2014, ignore.case = TRUE)

# show the content of qa section

Coke2014[11585:11674]

# save the qa section

qa\_Coke2014 <- Coke2014[11585:11674]

# save the data into a new text file

cat(qa\_Coke2014, file="C:/Users/tanmaygole/Desktop/CocaCola/QA\_Coke2014.txt", sep="n", append = TRUE)

# process 2013 file

# read the data from the file

Coke2013 = readLines(file2013)

# find the start and end section

grep(qa\_sentence\_start,Coke2013, ignore.case = TRUE)

grep(qa\_sentence\_end,Coke2013, ignore.case = TRUE)

# show the content of qa section

Coke2013[13087:13184]

# save the qa section

qa\_Coke2013 <- Coke2013[13087:13184]

# save the data into a new text file

cat(qa\_Coke2013, file="C:/Users/tanmaygole/Desktop/CocaCola/QA\_Coke2013.txt", sep="n", append = TRUE)

# process 2012 file

# read the data from the file

Coke2012 = readLines(file2012)

# find the start and end section

grep(qa\_sentence\_start,Coke2012, ignore.case = TRUE)

grep(qa\_sentence\_end,Coke2012, ignore.case = TRUE)

# show the content of qa section

Coke2012[13087:13184]

# save the qa section

qa\_Coke2012 <- Coke2012[13087:13184]

# save the data into a new text file

cat(qa\_Coke2012, file="C:/Users/tanmaygole/Desktop/CocaCola/QA\_Coke2012.txt", sep="n", append = TRUE)

# save the directory for corpus

cname <- "C:/Users/tanmaygole/Desktop/CocaCola/MyCorpus"

# save the corpus

qa\_sections <- Corpus(DirSource("C:/Users/tanmaygole/Desktop/DAFinalProject - Backup/newdataset/MyCorpus"),readerControl = list(reader=readPlain))

# show details for the corpus

inspect(qa\_sections[1])

dir(cname)

summary(qa\_sections)

# view the file

viewDocs <- function(d,n) {d %>% extract2(n) %>% as.character() %>% writeLines()}

viewDocs(qa\_sections, 1)

# convert to lower case

qa\_sections <- tm\_map(qa\_sections, content\_transformer(tolower))

# convert to lower case

qa\_sections <- tm\_map(qa\_sections, content\_transformer(tolower))

# remove numbers

qa\_sections <- tm\_map(qa\_sections, removeNumbers)

# remove stop words

qa\_sections <- tm\_map(qa\_sections, removeWords, stopwords("english"))

# Replacing certin expressions with spaces:

toSpace <- content\_transformer(function(x, pattern) gsub(pattern, " ",x))

qa\_sections <- tm\_map(qa\_sections, toSpace, "/|<|>|â|=|@|\\|:|;|-|\"")

# Remove Punctuations

qa\_sections <- tm\_map(qa\_sections, removePunctuation)

# Strip white spaces:

qa\_sections <- tm\_map(qa\_sections, stripWhitespace)

# Remove known often words:

qa\_sections <- tm\_map(qa\_sections, removeWords, c("right","color","inherit","vertical","decoration","background","b", "br", "q", "a", "i", "e", "div", "style","n","trim","size","font", "can", "also", "e", "mail", "via", "td", "align","border", "familytimes", "roman", "p", "tr", "nbsp", "with", "table", "cellspacing", "valign", "cellpadding", "width", "top", "left", "sizepx", "telephone", "if", "may", "help", "us", "will", "please", "unless", "visit", "thnbsp","toppx","bottompx", "nnn", "address", "nonbsp", "new", "bottom", "em", "arial", "serif"))

# Specific Transformation:

toString <- content\_transformer(function(x, from, to) gsub(from, to, x))

qa\_sections <- tm\_map(qa\_sections, toString, "broker bank", "bb")

# Stemming:

#install.packages("SnowballC")

qa\_sections <-tm\_map(qa\_sections, stemDocument)

# create a term document matrix

qa\_sections <-tm\_map(qa\_sections, PlainTextDocument)

qa\_dtm <- DocumentTermMatrix(qa\_sections)

qa\_dtm

# inpect the term document matrix

inspect(qa\_dtm[1:3, 1:203])

freq <- colSums((as.matrix(qa\_dtm)))

length(freq)

#freq

ord <- order(freq)

freq[head(ord)]

freq[tail(ord)]

head(table(freq), 15)

tail(table(freq), 15)

# remove sparse terms

dtms <- removeSparseTerms(qa\_dtm, 0.1)

dim(dtms)

inspect(dtms)

freq <-colSums(as.matrix(dtms))

freq

# frequencies of words

table(freq)

# Below we specify that we want terms / words

# which were used 50 or more times (in all documents / paragraphs).

findFreqTerms(dtms, lowfreq=50)

# save this frequencies to CVS file

freq50 <- findFreqTerms(dtms, lowfreq=50)

write.csv(freq50,file= "C:/Users/tanmaygole/Desktop/CocaCola/frequencies50.csv")

# Below we specify that we want terms / words

# which were used 100 or more times (in all documents / paragraphs).

findFreqTerms(dtms, lowfreq=100)

# save this frequencies to CVS file

freq100 <- findFreqTerms(dtms, lowfreq=100)

write.csv(freq100,file= "C:/Users/tanmaygole/Desktop/CocaCola/frequencies100.csv")

# Finding words which 'associate' together. Here, we are specifying

# the Term Document Matrix to use, the term we want to find associates

# for, and the lowest acceptable correlation limit with that term. This

# returns a vector of terms which are associated with 'CocaCola' at

# 0.50 or more (correlation) -- and reports each association in

# decending order.

findAssocs(dtms, "cocacola", corlimit = .5)

# Plotting Frequent Words

freq <- sort(colSums(as.matrix(dtms)), decreasing = TRUE)

head(freq, 14)

wf <- data.frame(word=names(freq), freq=freq)

head(wf)

subset(wf, freq>200) %>% ggplot(aes(word, freq))

subset(wf, freq>200)

subset(wf, freq>200) %>% ggplot(aes(word, freq)) + geom\_bar(stat="identity") + theme(axis.text.x=element\_text(angle=45, hjust=1))

#Dplyr

install.packages("dplyr")

library(dplyr)

subset(wf, freq>200) %>% ggplot(aes(word, freq)) + geom\_bar(stat="identity") + theme(axis.text.x=element\_text(angle=45, hjust=1))

# Generating a Word Cloud

set.seed(123)

wordcloud(names(freq), freq, min.freq=100,random.color=TRUE,colors=rainbow(7),random.order = FALSE)