Text mining and sentiment analysis for banking

Gautam

31 March 2020

# Check the r version  
#R.version  
#tinytex::install\_tinytex()

#####Read Twitter Data

# Set directory and read data  
  
setwd("E:/GLIM/EMPIRICAL/Sectors")  
tweets.df <- read.csv("Banking.csv")  
  
# Convert char date to correct date format  
#tweets.df$created <- as.Date(tweets.df$created, format= "%d-%m-%y")  
tweets.df$text <- as.character(tweets.df$text)  
#str(tweets.df)

#####Cleaning the text data by removing links, tags and delimiters.  
#####Build a Corpus, and specify the location to be the character Vectors

# Remove character string between < >  
#tweets.df$text <- genX(tweets.df$text, " <", ">")  
  
# Create document corpus with tweet text  
myCorpus<- Corpus(VectorSource(tweets.df$text))

#####convert to Lowercase

myCorpus <- tm\_map(myCorpus, content\_transformer(stri\_trans\_tolower))

## Warning in tm\_map.SimpleCorpus(myCorpus,  
## content\_transformer(stri\_trans\_tolower)): transformation drops documents

#####Remove the links (URLs)

removeURL <- function(x) gsub("http[^[:space:]]\*", "", x)   
myCorpus <- tm\_map(myCorpus, content\_transformer(removeURL))

## Warning in tm\_map.SimpleCorpus(myCorpus, content\_transformer(removeURL)):  
## transformation drops documents

#####Remove the @ (usernames)

removeUsername <- function(x) gsub("@[^[:space:]]\*", "", x)   
myCorpus <- tm\_map(myCorpus, content\_transformer(removeUsername))

## Warning in tm\_map.SimpleCorpus(myCorpus, content\_transformer(removeUsername)):  
## transformation drops documents

#####Remove anything except the english language and space

removeNumPunct <- function(x) gsub("[^[:alpha:][:space:]]\*", "", x)   
myCorpus <- tm\_map(myCorpus, content\_transformer(removeNumPunct))

## Warning in tm\_map.SimpleCorpus(myCorpus, content\_transformer(removeNumPunct)):  
## transformation drops documents

#####Remove Stopwords

myStopWords<- c((stopwords('english')),c("also"))  
myCorpus<- tm\_map(myCorpus,removeWords , myStopWords)

## Warning in tm\_map.SimpleCorpus(myCorpus, removeWords, myStopWords):  
## transformation drops documents

#####Remove Single letter words

removeSingle <- function(x) gsub(" . ", " ", x)   
myCorpus <- tm\_map(myCorpus, content\_transformer(removeSingle))

## Warning in tm\_map.SimpleCorpus(myCorpus, content\_transformer(removeSingle)):  
## transformation drops documents

#####Remove Extra Whitespaces

myCorpus<- tm\_map(myCorpus, stripWhitespace)

## Warning in tm\_map.SimpleCorpus(myCorpus, stripWhitespace): transformation drops  
## documents

#####keep a copy of “myCorpus” for stem completion later

myCorpusCopy<- myCorpus

#####removing similar words and replacing with single word

myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("bank", "banks", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("transaction", "transactions", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("securities", "security", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("blockchains", "blockchain", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("systems", "system", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("systemss", "system", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("bankss", "banks", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("transactionss", "transactions", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("banksing", "banking", x)))

#####Creating a term document matrix

#myCorpus <- Corpus(VectorSource(myCorpus))  
tdm<- TermDocumentMatrix(myCorpus, control= list(wordLengths= c(1, Inf)))  
tdm

## <<TermDocumentMatrix (terms: 1006, documents: 75)>>  
## Non-/sparse entries: 1860/73590  
## Sparsity : 98%  
## Maximal term length: 20  
## Weighting : term frequency (tf)

#####Find the terms used most frequently

(freq.terms <- findFreqTerms(tdm, lowfreq = 25))

## [1] "blockchain" "technology" "banks" "transactions"

term.freq <- rowSums(as.matrix(tdm))  
term.freq <- subset(term.freq, term.freq > 25)  
df <- data.frame(term = names(term.freq), freq= term.freq)

#####Frequency analysis

(freq.terms <- findFreqTerms(tdm, lowfreq = 10))

## [1] "banking" "blockchain" "india" "technology" "banks"   
## [6] "trade" "use" "will" "payments" "yes"   
## [11] "data" "financial" "transactions" "vendor" "security"   
## [16] "solution" "system" "can"

term.freq <- rowSums(as.matrix(tdm))  
term.freq <- subset(term.freq, term.freq > 10)  
df1 <- data.frame(term = names(term.freq), freq= term.freq)  
  
(freq.terms <- findFreqTerms(tdm, lowfreq = 55))

## [1] "blockchain" "banks"

term.freq <- rowSums(as.matrix(tdm))  
term.freq <- subset(term.freq, term.freq > 55)  
df2 <- data.frame(term = names(term.freq), freq= term.freq)  
  
(freq.terms <- findFreqTerms(tdm, lowfreq = 85))

## [1] "blockchain"

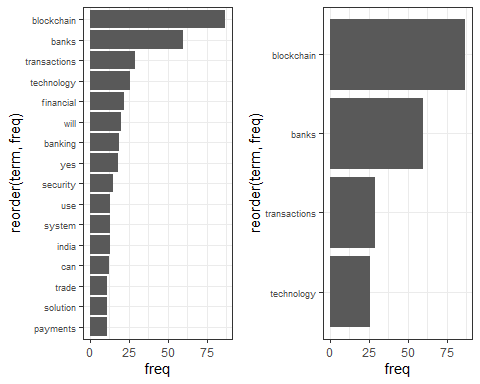
term.freq <- rowSums(as.matrix(tdm))  
term.freq <- subset(term.freq, term.freq > 85)  
df3 <- data.frame(term = names(term.freq), freq= term.freq)

#####plotting the graph of frequent terms

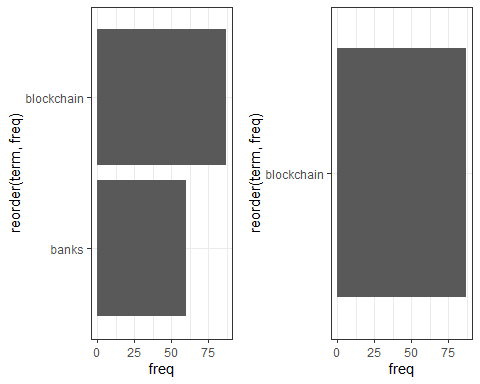
p1=ggplot(df1, aes(reorder(term, freq),freq)) + theme\_bw() + geom\_bar(stat = "identity") + coord\_flip() +labs(list(title="@10", x="Terms", y="Term Counts")) + theme(axis.text.y = element\_text(size=7))  
  
  
p2=ggplot(df, aes(reorder(term, freq),freq)) + theme\_bw() + geom\_bar(stat = "identity") + coord\_flip() +labs(list(title="@25", x="Terms", y="Term Counts"))+  
 theme(axis.text.y = element\_text(size=7))  
  
  
p3=ggplot(df2, aes(reorder(term, freq),freq)) + theme\_bw() + geom\_bar(stat = "identity") + coord\_flip() +labs(list(title="@55", x="Terms", y="Term Counts"))  
  
p4=ggplot(df3, aes(reorder(term, freq),freq)) + theme\_bw() + geom\_bar(stat = "identity") + coord\_flip() +labs(list(title="@85", x="Terms", y="Term Counts"))

#####plotting the graph of frequent terms

grid.arrange(p1,p2,ncol=2)

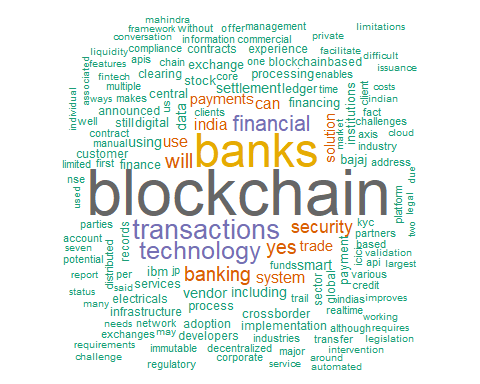


grid.arrange(p3,p4,ncol=2)



#####calculate the frequency of words and sort it by frequency and setting up the Wordcloud

# Creating the wordcloud  
  
word.freq <-sort(rowSums(as.matrix(tdm)), decreasing= F)  
pal<- brewer.pal(8, "Dark2")  
wordcloud(words = names(word.freq), freq = word.freq, min.freq = 2, random.order = F, colors = pal, max.words = 150)

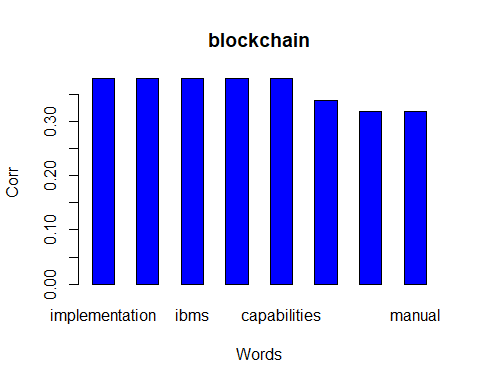


##### Correlations between the keywords

list1<- findAssocs(tdm, "blockchain", 0.3)  
corrdf1 <- t(data.frame(t(sapply(list1,c))))  
corrdf1

## blockchain  
## implementation 0.38  
## endtoend 0.38  
## ibms 0.38  
## offer 0.38  
## capabilities 0.38  
## technology 0.34  
## management 0.32  
## manual 0.32

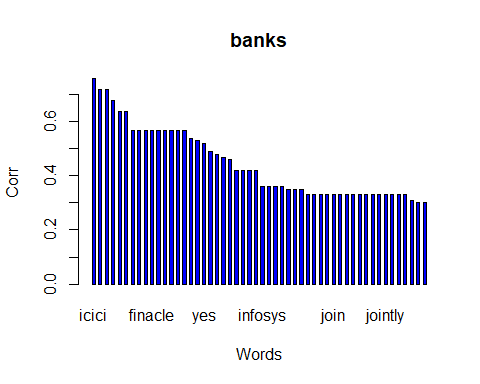
barplot(t(as.matrix(corrdf1)), beside=TRUE,xlab = "Words",ylab = "Corr",col = "blue",main = "blockchain",border = "black")



list1<- findAssocs(tdm, "banks", 0.3)  
corrdf1 <- t(data.frame(t(sapply(list1,c))))  
corrdf1

## banks  
## icici 0.76  
## kotak 0.72  
## mahindra 0.72  
## axis 0.68  
## indusind 0.64  
## rbl 0.64  
## additionally 0.57  
## connect 0.57  
## digitizes 0.57  
## finacle 0.57  
## invoice 0.57  
## joined 0.57  
## led 0.57  
## letters 0.57  
## south 0.57  
## trade 0.54  
## finance 0.53  
## yes 0.52  
## consortium 0.49  
## requirements 0.48  
## idfc 0.47  
## financing 0.46  
## bill 0.42  
## offers 0.42  
## february 0.42  
## collection 0.42  
## infosys 0.36  
## functionality 0.36  
## processes 0.36  
## powered 0.36  
## india 0.35  
## indian 0.35  
## business 0.35  
## border 0.33  
## cross 0.33  
## federal 0.33  
## interbanks 0.33  
## join 0.33  
## movement 0.33  
## vastly 0.33  
## carried 0.33  
## pilot 0.33  
## runs 0.33  
## conducted 0.33  
## elemential 0.33  
## jointly 0.33  
## know 0.33  
## recently 0.33  
## trial 0.33  
## address 0.31  
## hdfc 0.30  
## startup 0.30

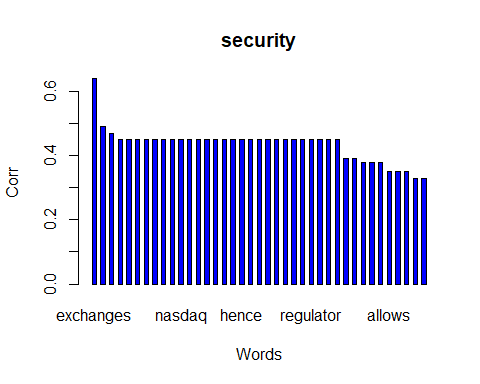
barplot(t(as.matrix(corrdf1)), beside=TRUE,xlab = "Words",ylab = "Corr",col = "blue",main = "banks",border = "black")



list1<- findAssocs(tdm, "security", 0.3)  
corrdf1 <- t(data.frame(t(sapply(list1,c))))  
corrdf1

## security  
## exchanges 0.64  
## including 0.49  
## new 0.47  
## begin 0.45  
## company 0.45  
## enterprisewide 0.45  
## fullyelectronic 0.45  
## initiative 0.45  
## learning 0.45  
## machine 0.45  
## nasdaq 0.45  
## noncurrency 0.45  
## world 0.45  
## amendments 0.45  
## brought 0.45  
## continue 0.45  
## disrupt 0.45  
## hence 0.45  
## least 0.45  
## legal 0.45  
## model 0.45  
## near 0.45  
## noted 0.45  
## placed 0.45  
## regulated 0.45  
## regulator 0.45  
## seems 0.45  
## term 0.45  
## viz 0.45  
## announced 0.39  
## parties 0.39  
## leverage 0.38  
## framework 0.38  
## issuance 0.38  
## allows 0.35  
## trust 0.35  
## individual 0.35  
## first 0.33  
## one 0.33

barplot(t(as.matrix(corrdf1)), beside=TRUE,xlab = "Words",ylab = "Corr",col = "blue",main = "security",border = "black")



##### Topic Modelling to identify latent/hidden topics using LDA technique

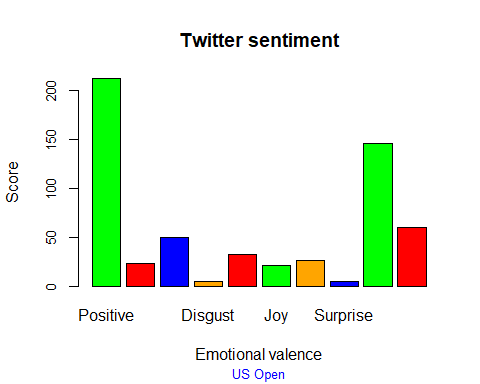
dtm <- as.DocumentTermMatrix(tdm)  
  
rowTotals <- apply(dtm , 1, sum)  
  
NullDocs <- dtm[rowTotals==0, ]  
dtm <- dtm[rowTotals> 0, ]  
  
if (length(NullDocs$dimnames$Docs) > 0) {  
tweets.df <- tweets.df[-as.numeric(NullDocs$dimnames$Docs),]  
}  
  
lda <- LDA(dtm, k = 5) # find 5 topic  
term <- terms(lda, 10) # first 7 terms of every topic  
(term <- apply(term, MARGIN = 2, paste, collapse = ", "))

## Topic 1   
## "blockchain, transactions, banks, system, processing, will, banking, yes, partners, developers"   
## Topic 2   
## "blockchain, banks, technology, security, including, financial, use, india, transactions, first"   
## Topic 3   
## "transactions, blockchain, will, banks, payments, crossborder, payment, settlement, technology, smart"   
## Topic 4   
## "banks, blockchain, yes, technology, financing, vendor, exchange, trade, solution, stock"   
## Topic 5   
## "blockchain, financial, technology, banking, adoption, sector, services, banks, institutions, finance"

topics<- topics(lda)  
#topics<- data.frame(date=(tweets.df$created), topic = topics)  
#qplot (date, ..count.., data=topics, geom ="density", fill= term[topic], position="stack")

#####Sentiment Analysis: understanding emotional valence in tweets using syuzhet

mysentiment<-get\_nrc\_sentiment((tweets.df$text))  
  
# Get the sentiment score for each emotion  
mysentiment.positive =sum(mysentiment$positive)  
mysentiment.anger =sum(mysentiment$anger)  
mysentiment.anticipation =sum(mysentiment$anticipation)  
mysentiment.disgust =sum(mysentiment$disgust)  
mysentiment.fear =sum(mysentiment$fear)  
mysentiment.joy =sum(mysentiment$joy)  
mysentiment.sadness =sum(mysentiment$sadness)  
mysentiment.surprise =sum(mysentiment$surprise)  
mysentiment.trust =sum(mysentiment$trust)  
mysentiment.negative =sum(mysentiment$negative)  
  
# Create the bar chart  
yAxis <- c(mysentiment.positive,  
 + mysentiment.anger,  
 + mysentiment.anticipation,  
 + mysentiment.disgust,  
 + mysentiment.fear,  
 + mysentiment.joy,  
 + mysentiment.sadness,  
 + mysentiment.surprise,  
 + mysentiment.trust,  
 + mysentiment.negative)  
  
xAxis <- c("Positive","Anger","Anticipation","Disgust","Fear","Joy","Sadness",  
 "Surprise","Trust","Negative")  
colors <- c("green","red","blue","orange","red","green","orange","blue","green","red")  
yRange <- range(0,yAxis)  
barplot(yAxis, names.arg = xAxis,   
 xlab = "Emotional valence", ylab = "Score", main = "Twitter sentiment",   
 sub = "US Open", col = colors, border = "black", xpd = F, ylim = yRange,  
 axisnames = T, cex.axis = 0.8, cex.sub = 0.8, col.sub = "blue")



Text mining and sentiment analysis for public sector

Shubhi

31 March 2020

Use Twitter data on Flipkart (e-commerce giant in India) to demonstrate text mining and visualization techniques including text cleanup, word cloud, frequent terms, topic modelling and sentiment analysis

# Check the r version  
#R.version  
#tinytex::install\_tinytex()

#####Read Twitter Data

# Set directory and read data  
  
setwd("E:/Empirical Study/Final")  
tweets.df <- read.csv("Public.csv")  
  
# Convert char date to correct date format  
#tweets.df$created <- as.Date(tweets.df$created, format= "%d-%m-%y")  
tweets.df$text <- as.character(tweets.df$text)  
#str(tweets.df)

#####Cleaning the text data by removing links, tags and delimiters.  
#####Build a Corpus, and specify the location to be the character Vectors

# Remove character string between < >  
#tweets.df$text <- genX(tweets.df$text, " <", ">")  
  
# Create document corpus with tweet text  
myCorpus<- Corpus(VectorSource(tweets.df$text))

#####convert to Lowercase

myCorpus <- tm\_map(myCorpus, content\_transformer(stri\_trans\_tolower))

## Warning in tm\_map.SimpleCorpus(myCorpus,  
## content\_transformer(stri\_trans\_tolower)): transformation drops documents

#####Remove the links (URLs)

removeURL <- function(x) gsub("http[^[:space:]]\*", "", x)   
myCorpus <- tm\_map(myCorpus, content\_transformer(removeURL))

## Warning in tm\_map.SimpleCorpus(myCorpus, content\_transformer(removeURL)):  
## transformation drops documents

#####Remove the @ (usernames)

removeUsername <- function(x) gsub("@[^[:space:]]\*", "", x)   
myCorpus <- tm\_map(myCorpus, content\_transformer(removeUsername))

## Warning in tm\_map.SimpleCorpus(myCorpus, content\_transformer(removeUsername)):  
## transformation drops documents

#####Remove anything except the english language and space

removeNumPunct <- function(x) gsub("[^[:alpha:][:space:]]\*", "", x)   
myCorpus <- tm\_map(myCorpus, content\_transformer(removeNumPunct))

## Warning in tm\_map.SimpleCorpus(myCorpus, content\_transformer(removeNumPunct)):  
## transformation drops documents

#####Remove Stopwords

myStopWords<- c((stopwords('english')),c("also"))  
myCorpus<- tm\_map(myCorpus,removeWords , myStopWords)

## Warning in tm\_map.SimpleCorpus(myCorpus, removeWords, myStopWords):  
## transformation drops documents

#####Remove Single letter words

removeSingle <- function(x) gsub(" . ", " ", x)   
myCorpus <- tm\_map(myCorpus, content\_transformer(removeSingle))

## Warning in tm\_map.SimpleCorpus(myCorpus, content\_transformer(removeSingle)):  
## transformation drops documents

#####Remove Extra Whitespaces

myCorpus<- tm\_map(myCorpus, stripWhitespace)

## Warning in tm\_map.SimpleCorpus(myCorpus, stripWhitespace): transformation drops  
## documents

#####keep a copy of “myCorpus” for stem completion later

myCorpusCopy<- myCorpus

#####removing similar words and replacing with single word

myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("bank", "banks", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("transaction", "transactions", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("securities", "security", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("blockchains", "blockchain", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("systems", "system", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("systemss", "system", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("bankss", "banks", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("transactionss", "transactions", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("banksing", "banking", x)))

#####Creating a term document matrix

#myCorpus <- Corpus(VectorSource(myCorpus))  
tdm<- TermDocumentMatrix(myCorpus, control= list(wordLengths= c(1, Inf)))  
tdm

## <<TermDocumentMatrix (terms: 777, documents: 80)>>  
## Non-/sparse entries: 1338/60822  
## Sparsity : 98%  
## Maximal term length: 24  
## Weighting : term frequency (tf)

#####Find the terms used most frequently

(freq.terms <- findFreqTerms(tdm, lowfreq = 25))

## [1] "blockchain" "can"

term.freq <- rowSums(as.matrix(tdm))  
term.freq <- subset(term.freq, term.freq > 25)  
df <- data.frame(term = names(term.freq), freq= term.freq)

#####Frequency analysis

(freq.terms <- findFreqTerms(tdm, lowfreq = 10))

## [1] "blockchain" "government" "indiachain" "will" "india"   
## [6] "technology" "can" "use" "coffee" "farmers"   
## [11] "platform"

term.freq <- rowSums(as.matrix(tdm))  
term.freq <- subset(term.freq, term.freq > 10)  
df1 <- data.frame(term = names(term.freq), freq= term.freq)  
  
(freq.terms <- findFreqTerms(tdm, lowfreq = 55))

## [1] "blockchain"

term.freq <- rowSums(as.matrix(tdm))  
term.freq <- subset(term.freq, term.freq > 55)  
df2 <- data.frame(term = names(term.freq), freq= term.freq)  
  
(freq.terms <- findFreqTerms(tdm, lowfreq = 85))

## character(0)

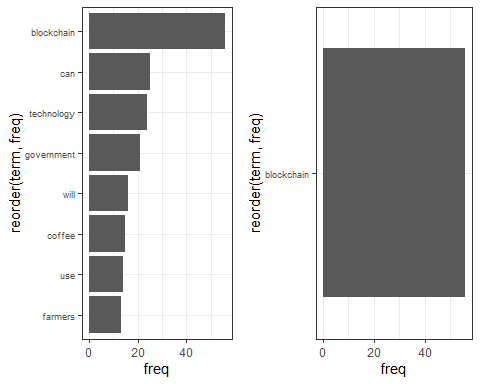
term.freq <- rowSums(as.matrix(tdm))  
term.freq <- subset(term.freq, term.freq > 85)  
df3 <- data.frame(term = names(term.freq), freq= term.freq)

#####plotting the graph of frequent terms

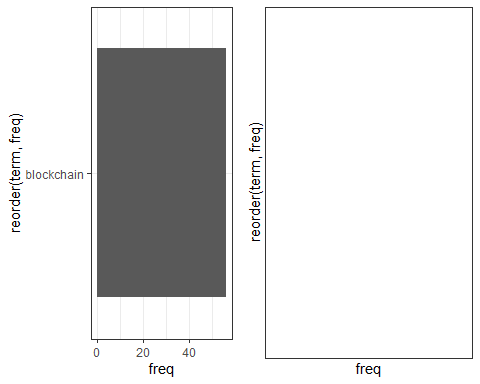
p1=ggplot(df1, aes(reorder(term, freq),freq)) + theme\_bw() + geom\_bar(stat = "identity") + coord\_flip() +labs(list(title="@10", x="Terms", y="Term Counts")) + theme(axis.text.y = element\_text(size=7))  
  
  
p2=ggplot(df, aes(reorder(term, freq),freq)) + theme\_bw() + geom\_bar(stat = "identity") + coord\_flip() +labs(list(title="@25", x="Terms", y="Term Counts"))+  
 theme(axis.text.y = element\_text(size=7))  
  
  
p3=ggplot(df2, aes(reorder(term, freq),freq)) + theme\_bw() + geom\_bar(stat = "identity") + coord\_flip() +labs(list(title="@55", x="Terms", y="Term Counts"))  
  
p4=ggplot(df3, aes(reorder(term, freq),freq)) + theme\_bw() + geom\_bar(stat = "identity") + coord\_flip() +labs(list(title="@85", x="Terms", y="Term Counts"))

#####plotting the graph of frequent terms

grid.arrange(p1,p2,ncol=2)

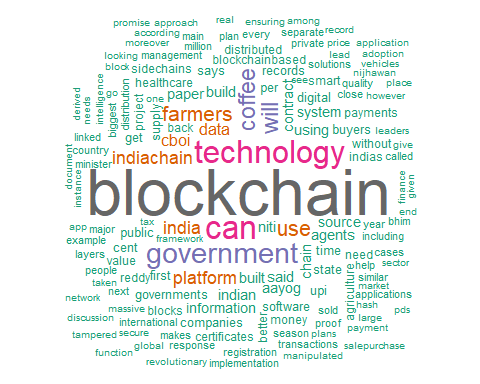


grid.arrange(p3,p4,ncol=2)



#####calculate the frequency of words and sort it by frequency and setting up the Wordcloud

# Creating the wordcloud  
  
word.freq <-sort(rowSums(as.matrix(tdm)), decreasing= F)  
pal<- brewer.pal(8, "Dark2")  
wordcloud(words = names(word.freq), freq = word.freq, min.freq = 2, random.order = F, colors = pal, max.words = 150)

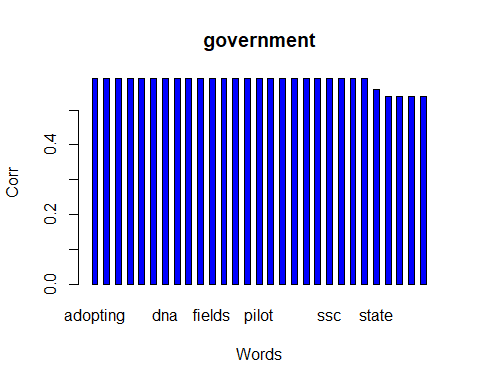


##### Correlations between the keywords

list1<- findAssocs(tdm, "government", 0.5)  
corrdf1 <- t(data.frame(t(sapply(list1,c))))  
corrdf1

## government  
## adopting 0.59  
## common 0.59  
## completed 0.59  
## cutting 0.59  
## departments 0.59  
## detailed 0.59  
## dna 0.59  
## edge 0.59  
## enhance 0.59  
## extensive 0.59  
## fields 0.59  
## five 0.59  
## governance 0.59  
## health 0.59  
## pilot 0.59  
## prepared 0.59  
## principal 0.59  
## regulatory 0.59  
## sandbox 0.59  
## secretary 0.59  
## ssc 0.59  
## svr 0.59  
## told 0.59  
## various 0.59  
## state 0.56  
## department 0.54  
## months 0.54  
## report 0.54  
## srinivas 0.54

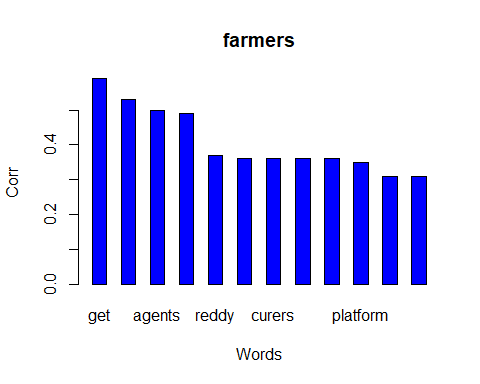
barplot(t(as.matrix(corrdf1)), beside=TRUE,xlab = "Words",ylab = "Corr",col = "blue",main = "government",border = "black")



list1<- findAssocs(tdm, "farmers", 0.3)  
corrdf1 <- t(data.frame(t(sapply(list1,c))))  
corrdf1

## farmers  
## get 0.59  
## cboi 0.53  
## agents 0.50  
## coffee 0.49  
## reddy 0.37  
## produce 0.36  
## curers 0.36  
## register 0.36  
## salepurchase 0.36  
## platform 0.35  
## says 0.31  
## buyers 0.31

barplot(t(as.matrix(corrdf1)), beside=TRUE,xlab = "Words",ylab = "Corr",col = "blue",main = "farmers",border = "black")



##### Topic Modelling to identify latent/hidden topics using LDA technique

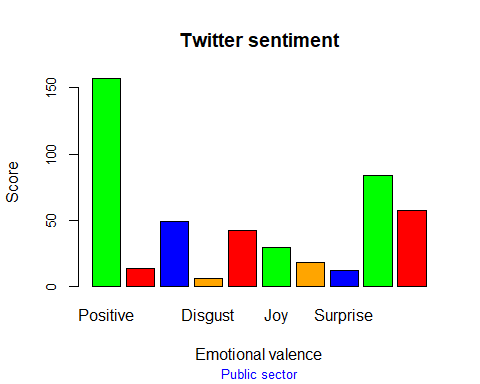
dtm <- as.DocumentTermMatrix(tdm)  
  
rowTotals <- apply(dtm , 1, sum)  
  
NullDocs <- dtm[rowTotals==0, ]  
dtm <- dtm[rowTotals> 0, ]  
  
if (length(NullDocs$dimnames$Docs) > 0) {  
tweets.df <- tweets.df[-as.numeric(NullDocs$dimnames$Docs),]  
}  
  
lda <- LDA(dtm, k = 5) # find 5 topic  
term <- terms(lda, 10) # first 10 terms of every topic  
(term <- apply(term, MARGIN = 2, paste, collapse = ", "))

## Topic 1   
## "contract, blockchain, coffee, data, can, platform, smart, indiachain, digital, use"   
## Topic 2   
## "can, blockchain, technology, government, agents, farmers, information, indiachain, using, data"   
## Topic 3   
## "blockchain, technology, chain, government, use, supply, blocks, management, indian, minister"   
## Topic 4   
## "blockchain, india, aayog, niti, technology, government, will, healthcare, governments, indian"   
## Topic 5   
## "farmers, will, coffee, cboi, blockchain, can, platform, built, government, get"

topics<- topics(lda)  
#topics<- data.frame(date=(tweets.df$created), topic = topics)  
#qplot (date, ..count.., data=topics, geom ="density", fill= term[topic], position="stack")

#####Sentiment Analysis: understanding emotional valence in tweets using syuzhet

mysentiment<-get\_nrc\_sentiment((tweets.df$text))  
  
# Get the sentiment score for each emotion  
mysentiment.positive =sum(mysentiment$positive)  
mysentiment.anger =sum(mysentiment$anger)  
mysentiment.anticipation =sum(mysentiment$anticipation)  
mysentiment.disgust =sum(mysentiment$disgust)  
mysentiment.fear =sum(mysentiment$fear)  
mysentiment.joy =sum(mysentiment$joy)  
mysentiment.sadness =sum(mysentiment$sadness)  
mysentiment.surprise =sum(mysentiment$surprise)  
mysentiment.trust =sum(mysentiment$trust)  
mysentiment.negative =sum(mysentiment$negative)  
  
# Create the bar chart  
yAxis <- c(mysentiment.positive,  
 + mysentiment.anger,  
 + mysentiment.anticipation,  
 + mysentiment.disgust,  
 + mysentiment.fear,  
 + mysentiment.joy,  
 + mysentiment.sadness,  
 + mysentiment.surprise,  
 + mysentiment.trust,  
 + mysentiment.negative)  
  
xAxis <- c("Positive","Anger","Anticipation","Disgust","Fear","Joy","Sadness",  
 "Surprise","Trust","Negative")  
colors <- c("green","red","blue","orange","red","green","orange","blue","green","red")  
yRange <- range(0,yAxis)  
barplot(yAxis, names.arg = xAxis,   
 xlab = "Emotional valence", ylab = "Score", main = "Twitter sentiment",   
 sub = "Public sector", col = colors, border = "black", xpd = F, ylim = yRange,  
 axisnames = T, cex.axis = 0.8, cex.sub = 0.8, col.sub = "blue")



Text mining and sentiment analysis for SC sector

Shubhi

31 March 2020

Use Twitter data on Flipkart (e-commerce giant in India) to demonstrate text mining and visualization techniques including text cleanup, word cloud, frequent terms, topic modelling and sentiment analysis

# Check the r version  
#R.version  
#tinytex::install\_tinytex()

#####Read Twitter Data

# Set directory and read data  
  
setwd("E:/Empirical Study/Final")  
tweets.df <- read.csv("SupplyChain.csv")  
  
# Convert char date to correct date format  
#tweets.df$created <- as.Date(tweets.df$created, format= "%d-%m-%y")  
tweets.df$text <- as.character(tweets.df$text)  
#str(tweets.df)

#####Cleaning the text data by removing links, tags and delimiters.  
#####Build a Corpus, and specify the location to be the character Vectors

# Remove character string between < >  
#tweets.df$text <- genX(tweets.df$text, " <", ">")  
  
# Create document corpus with tweet text  
myCorpus<- Corpus(VectorSource(tweets.df$text))

#####convert to Lowercase

myCorpus <- tm\_map(myCorpus, content\_transformer(stri\_trans\_tolower))

## Warning in tm\_map.SimpleCorpus(myCorpus,  
## content\_transformer(stri\_trans\_tolower)): transformation drops documents

#####Remove the links (URLs)

removeURL <- function(x) gsub("http[^[:space:]]\*", "", x)   
myCorpus <- tm\_map(myCorpus, content\_transformer(removeURL))

## Warning in tm\_map.SimpleCorpus(myCorpus, content\_transformer(removeURL)):  
## transformation drops documents

#####Remove the @ (usernames)

removeUsername <- function(x) gsub("@[^[:space:]]\*", "", x)   
myCorpus <- tm\_map(myCorpus, content\_transformer(removeUsername))

## Warning in tm\_map.SimpleCorpus(myCorpus, content\_transformer(removeUsername)):  
## transformation drops documents

#####Remove anything except the english language and space

removeNumPunct <- function(x) gsub("[^[:alpha:][:space:]]\*", "", x)   
myCorpus <- tm\_map(myCorpus, content\_transformer(removeNumPunct))

## Warning in tm\_map.SimpleCorpus(myCorpus, content\_transformer(removeNumPunct)):  
## transformation drops documents

#####Remove Stopwords

myStopWords<- c((stopwords('english')),c("also"))  
myCorpus<- tm\_map(myCorpus,removeWords , myStopWords)

## Warning in tm\_map.SimpleCorpus(myCorpus, removeWords, myStopWords):  
## transformation drops documents

#####Remove Single letter words

removeSingle <- function(x) gsub(" . ", " ", x)   
myCorpus <- tm\_map(myCorpus, content\_transformer(removeSingle))

## Warning in tm\_map.SimpleCorpus(myCorpus, content\_transformer(removeSingle)):  
## transformation drops documents

#####Remove Extra Whitespaces

myCorpus<- tm\_map(myCorpus, stripWhitespace)

## Warning in tm\_map.SimpleCorpus(myCorpus, stripWhitespace): transformation drops  
## documents

#####keep a copy of “myCorpus” for stem completion later

myCorpusCopy<- myCorpus

#####removing similar words and replacing with single word

myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("bank", "banks", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("transaction", "transactions", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("securities", "security", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("blockchains", "blockchain", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("systems", "system", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("systemss", "system", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("bankss", "banks", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("transactionss", "transactions", x)))  
myCorpus <-tm\_map(myCorpus, content\_transformer( function(x) gsub("banksing", "banking", x)))

#####Creating a term document matrix

#myCorpus <- Corpus(VectorSource(myCorpus))  
tdm<- TermDocumentMatrix(myCorpus, control= list(wordLengths= c(1, Inf)))  
tdm

## <<TermDocumentMatrix (terms: 910, documents: 63)>>  
## Non-/sparse entries: 1531/55799  
## Sparsity : 97%  
## Maximal term length: 26  
## Weighting : term frequency (tf)

#####Find the terms used most frequently

(freq.terms <- findFreqTerms(tdm, lowfreq = 25))

## [1] "blockchain" "supply" "chain"

term.freq <- rowSums(as.matrix(tdm))  
term.freq <- subset(term.freq, term.freq > 25)  
df <- data.frame(term = names(term.freq), freq= term.freq)

#####Frequency analysis

(freq.terms <- findFreqTerms(tdm, lowfreq = 10))

## [1] "blockchain" "logistics" "technology" "supply" "can"   
## [6] "chain" "help" "will" "shrimp" "walmart"

term.freq <- rowSums(as.matrix(tdm))  
term.freq <- subset(term.freq, term.freq > 10)  
df1 <- data.frame(term = names(term.freq), freq= term.freq)  
  
(freq.terms <- findFreqTerms(tdm, lowfreq = 55))

## [1] "blockchain"

term.freq <- rowSums(as.matrix(tdm))  
term.freq <- subset(term.freq, term.freq > 55)  
df2 <- data.frame(term = names(term.freq), freq= term.freq)  
  
(freq.terms <- findFreqTerms(tdm, lowfreq = 85))

## character(0)

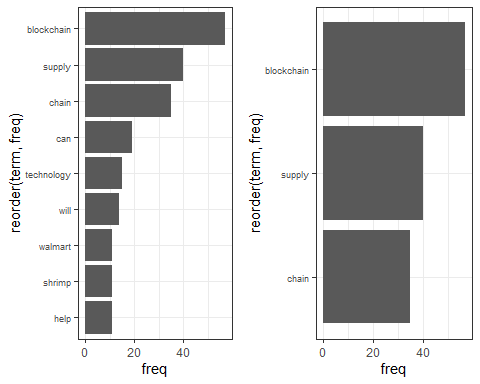
term.freq <- rowSums(as.matrix(tdm))  
term.freq <- subset(term.freq, term.freq > 85)  
df3 <- data.frame(term = names(term.freq), freq= term.freq)

#####plotting the graph of frequent terms

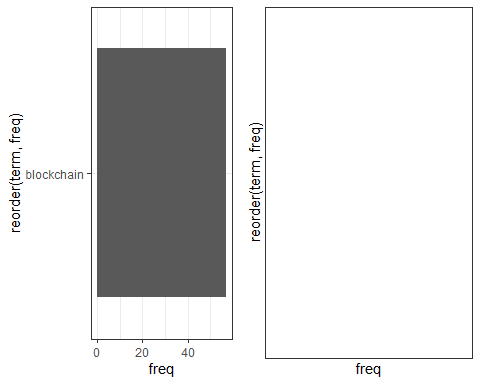
p1=ggplot(df1, aes(reorder(term, freq),freq)) + theme\_bw() + geom\_bar(stat = "identity") + coord\_flip() +labs(list(title="@10", x="Terms", y="Term Counts")) + theme(axis.text.y = element\_text(size=7))  
  
  
p2=ggplot(df, aes(reorder(term, freq),freq)) + theme\_bw() + geom\_bar(stat = "identity") + coord\_flip() +labs(list(title="@25", x="Terms", y="Term Counts"))+  
 theme(axis.text.y = element\_text(size=7))  
  
  
p3=ggplot(df2, aes(reorder(term, freq),freq)) + theme\_bw() + geom\_bar(stat = "identity") + coord\_flip() +labs(list(title="@55", x="Terms", y="Term Counts"))  
  
p4=ggplot(df3, aes(reorder(term, freq),freq)) + theme\_bw() + geom\_bar(stat = "identity") + coord\_flip() +labs(list(title="@85", x="Terms", y="Term Counts"))

#####plotting the graph of frequent terms

grid.arrange(p1,p2,ncol=2)

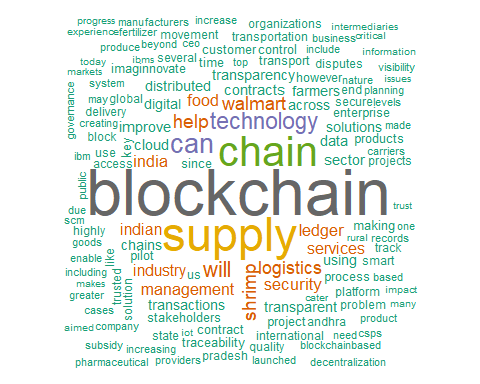


grid.arrange(p3,p4,ncol=2)



#####calculate the frequency of words and sort it by frequency and setting up the Wordcloud

# Creating the wordcloud  
  
word.freq <-sort(rowSums(as.matrix(tdm)), decreasing= F)  
pal<- brewer.pal(8, "Dark2")  
wordcloud(words = names(word.freq), freq = word.freq, min.freq = 2, random.order = F, colors = pal, max.words = 150)

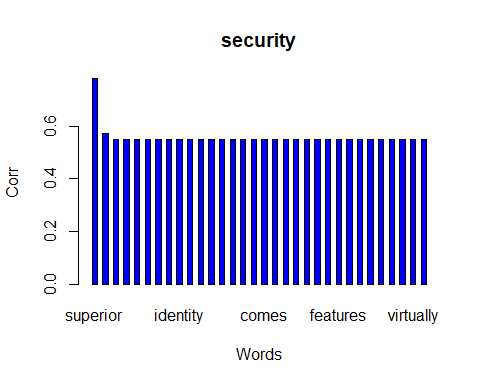


##### Correlations between the keywords

list1<- findAssocs(tdm, "security", 0.5)  
corrdf1 <- t(data.frame(t(sapply(list1,c))))  
corrdf1

## security  
## superior 0.78  
## without 0.57  
## another 0.55  
## area 0.55  
## automating 0.55  
## blockchainenabled 0.55  
## concern 0.55  
## consolidate 0.55  
## identity 0.55  
## infrastructure 0.55  
## oversight 0.55  
## significantly 0.55  
## strengthened 0.55  
## validation 0.55  
## acknowledged 0.55  
## characteristics 0.55  
## comes 0.55  
## consensus 0.55  
## cryptographic 0.55  
## defining 0.55  
## enabling 0.55  
## environment 0.55  
## evidence 0.55  
## features 0.55  
## leaving 0.55  
## mandates 0.55  
## mechanism 0.55  
## modified 0.55  
## protection 0.55  
## validated 0.55  
## virtually 0.55  
## way 0.55

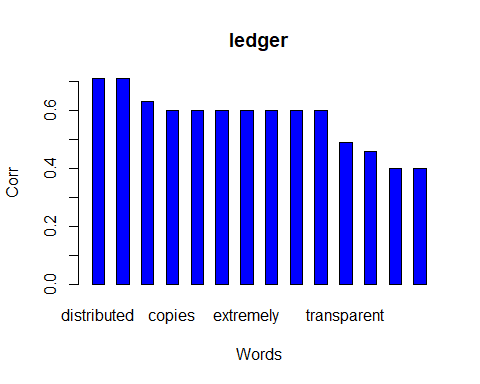
barplot(t(as.matrix(corrdf1)), beside=TRUE,xlab = "Words",ylab = "Corr",col = "blue",main = "security",border = "black")



list1<- findAssocs(tdm, "ledger", 0.3)  
corrdf1 <- t(data.frame(t(sapply(list1,c))))  
corrdf1

## ledger  
## distributed 0.71  
## transactions 0.71  
## makes 0.63  
## copies 0.60  
## efficient 0.60  
## every 0.60  
## extremely 0.60  
## function. 0.60  
## recording 0.60  
## scalable 0.60  
## transparent 0.49  
## highly 0.46  
## multiple 0.40  
## facilitates 0.40

barplot(t(as.matrix(corrdf1)), beside=TRUE,xlab = "Words",ylab = "Corr",col = "blue",main = "ledger",border = "black")



##### Topic Modelling to identify latent/hidden topics using LDA technique

dtm <- as.DocumentTermMatrix(tdm)  
  
rowTotals <- apply(dtm , 1, sum)  
  
NullDocs <- dtm[rowTotals==0, ]  
dtm <- dtm[rowTotals> 0, ]  
  
if (length(NullDocs$dimnames$Docs) > 0) {  
tweets.df <- tweets.df[-as.numeric(NullDocs$dimnames$Docs),]  
}  
  
lda <- LDA(dtm, k = 5) # find 5 topic  
term <- terms(lda, 10) # first 10 terms of every topic  
(term <- apply(term, MARGIN = 2, paste, collapse = ", "))

## Topic 1   
## "blockchain, management, imaginnovate, will, transportation, stakeholders, like, security, logistics, technology"   
## Topic 2   
## "blockchain, supply, chain, shrimp, walmart, improve, traceability, logistics, food, can"   
## Topic 3   
## "blockchain, supply, technology, can, chain, block, logistics, secure, chains, highly"   
## Topic 4   
## "blockchain, services, supply, can, chain, cloud, ledger, transactions, customer, security"   
## Topic 5   
## "chain, supply, blockchain, can, help, technology, process, across, transparent, pilot"

topics<- topics(lda)  
#topics<- data.frame(date=(tweets.df$created), topic = topics)  
#qplot (date, ..count.., data=topics, geom ="density", fill= term[topic], position="stack")

#####Sentiment Analysis: understanding emotional valence in tweets using syuzhet

mysentiment<-get\_nrc\_sentiment((tweets.df$text))  
  
# Get the sentiment score for each emotion  
mysentiment.positive =sum(mysentiment$positive)  
mysentiment.anger =sum(mysentiment$anger)  
mysentiment.anticipation =sum(mysentiment$anticipation)  
mysentiment.disgust =sum(mysentiment$disgust)  
mysentiment.fear =sum(mysentiment$fear)  
mysentiment.joy =sum(mysentiment$joy)  
mysentiment.sadness =sum(mysentiment$sadness)  
mysentiment.surprise =sum(mysentiment$surprise)  
mysentiment.trust =sum(mysentiment$trust)  
mysentiment.negative =sum(mysentiment$negative)  
  
# Create the bar chart  
yAxis <- c(mysentiment.positive,  
 + mysentiment.anger,  
 + mysentiment.anticipation,  
 + mysentiment.disgust,  
 + mysentiment.fear,  
 + mysentiment.joy,  
 + mysentiment.sadness,  
 + mysentiment.surprise,  
 + mysentiment.trust,  
 + mysentiment.negative)  
  
xAxis <- c("Positive","Anger","Anticipation","Disgust","Fear","Joy","Sadness",  
 "Surprise","Trust","Negative")  
colors <- c("green","red","blue","orange","red","green","orange","blue","green","red")  
yRange <- range(0,yAxis)  
barplot(yAxis, names.arg = xAxis,   
 xlab = "Emotional valence", ylab = "Score", main = "Twitter sentiment",   
 sub = "Supply Chain", col = colors, border = "black", xpd = F, ylim = yRange,  
 axisnames = T, cex.axis = 0.8, cex.sub = 0.8, col.sub = "blue")

