WARWICK BUSINESS SCHOOL

Masters Programmes
Individual Work Assignment Cover Sheet

Submitted by: U2087013

Date: 30/06/2021
Module Title: Text Analytics

Module Code: IB9CW0
Date/Year of Module: 2020-2021
Submission Deadline: 20:00 on 30th June 2021

Word Count: 2400
Number of Pages: 42

I declare that this work is being submitted by myself, in accordance with the University's Regulation 11 and the WBS guidelines on plagiarism and collusion. All external references and stheces are clearly acknowledged and identified within the contents. No substantial part(s) of the work submitted here has also been submitted in other assessments for accredited ctheses of study and if this has been done it may result in us being reported for self- plagiarism and an appropriate reduction in marks may be made when marking this piece of work.

Table of Contents

Executive summary

The objective of this report is to analyze and understand the annual financial report management discussion of 10-k filing of 30 companies over last decade (2010-2020) which is publicly accessible to public using edgar package or sec.gov website, by leveraging sentiment analysis from different dictionaries, text mining to clean and merge data, the final data frame has 92 percent less observations than the raw data.

Introduction

```
# Set current file directory
setwd("~/Movies/Text Analytics/Individual")
# reading SP500 data file containing 30 companies data
sp500 <- read.csv("sp500.csv", stringsAsFactors = TRUE)
```

Acquiring Data

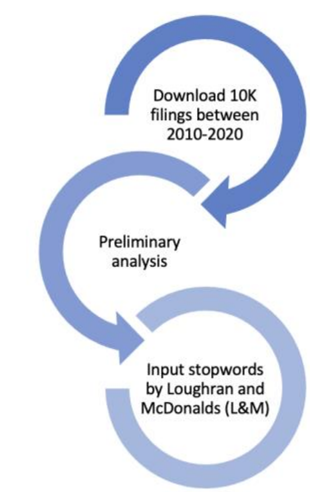

Figure 1:Acquring Data

Acquiring data before any analysis is most crucial part, and to make process efficient S&P 500 CSV consists of 30 companies was loaded to get master index meta data and management discussion of 10k filing of those companies of last decade, since 10k filings are identical in all companies, author found out by visually exploration among these data files key components are accession number, company name, and big chunk of management discussion containing valuable information about its current state and strategic direction. The content of an annual report can also change dramatically from year to year which results in affecting financial market, hence 10K's MD&A content is the most important to any company and to the analyst

by using Edgar package, it's very easy to download the 10-k filings of each company over the period of 10 years.

```
# getting master index meta data from edgar package
edgar::getMasterIndex(filing.year = c(2010:2020))
# getting management discussion of every cik from edgar package
 for(cik in sp500$cik){
   edgar::getMgmtDisc(cik.no=cik, filing.year=c(2010:2020))
 }
```

The package will create a folder called "Master Indexes" and will aggregate **all the report metadata filed** for the that is specified, consists of important variable like, date of filing, accession_number, edgar link and quater If instead of an atomic value a vector is provided then the command will execute in a loop.

Inspect the master index

The folder Master Indexes contains the metadata

```
indexesMaster_list <- list.files("Master Indexes/",pattern="Rda")
indexMaster_df <- data.frame()

for(indexMaster in indexesMaster_list){
  load(paste0("Master Indexes/",indexMaster))

  processedIndex_df <- year.master %>%
    filter(cik %in% sp500$cik, form.type %in% '10-K') %>%
    mutate(date.filed = as.Date(date.filed)) %>%
    mutate(year_filed = year(date.filed)) %>%
    mutate(accession.number = gsub(".*/","",edgar.link))  %>%
    mutate(accession.number = gsub('.txt','',accession.number))

  colnames(processedIndex_df) <- gsub("\\.","_", colnames(processedIndex_df))
  indexMaster_df <- rbind(indexMaster_df, processedIndex_df)

}

sp500$cik <- as.character(sp500$cik)

indexMaster_df <- indexMaster_df %>%
  left_join(sp500, by="cik")
```

from preliminary analysis, certain stop words has been chosen like company, financial, footnote. total of 181 stop words has been added to the list which will be used for futher analysis

```r
stopwords <- stopwords(language = "en")

stopwordsfinal_col <- c(stopwords, c("company","disclaimer","financial","repo
rt","figure","footnote", "fiscal","asset") )
```

1. Part A

a. Exploratory data analysis

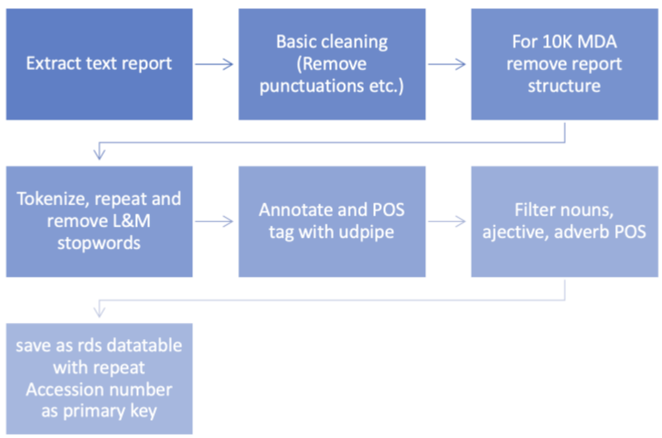

Figure 2:EDA

The raw data files are treated for further analysis where necessary information was taken out by cleaning the text using regex functions and annotating important words

```r
## data exploration

plainText_df <- data.frame()
split_size <- 50
ud_model <- udpipe_download_model(language="english", overwrite=F)
ud_model <- udpipe_load_model(ud_model$file_model)

index <- split(indexMaster_df$accession_number, ceiling(seq_along(indexMaster
_df$accession_number)/split_size))

for(i in index){
  #input data
  listed_files <- list.files('MD&A section text', pattern = paste0(paste0(i,
'.txt'), collapse = "|"))
  file_path <- paste0('MD&A section text/', listed_files)
```

```r
  for(i in 1:length(file_path)){
    text_file <- read_lines (file_path[i])
    processed_text <- tibble(company_name = tolower(gsub('Company Name: ','',
text_file[2])),
                               accession_number = gsub('Accession Number: ','
', text_file[5]),
                               managementDiscussion = tolower(text_file[8]) %
>% # Convert all words to lowercase:
                                 removeNumbers() %>% # remove numbers
                                 stripWhitespace()) # remove extra space

    # text cleaning by removing punctuation and repeated words
    company_name <- unlist(str_split(processed_text$company_name, " ", n=ncha
r(processed_text$company_name)))[1]
    sub_this <- c("item","management","managements","discussion and analysis"
,"financial condition", "results of operations", company_name)
    processed_text$text <- gsub(paste0(sub_this, collapse = '|'),"",processed
_text$managementDiscussion)
    processed_text$text <- gsub('[[:punct:][:blank:]]+',' ',processed_text$te
xt)


    # tokenize files and removal of stop words
    processedTokens_df <- processed_text %>%
      select(accession_number, text) %>%
      unnest_tokens(word, text) %>%
      group_by(accession_number, word) %>%
      filter(!word %in% stopwordsfinal_col)

   # part of speech tagging using udpipe
    local_df <- udpipe_annotate(processedTokens_df$word,
                               doc_id = processedTokens_df$accession_number,
                               object = ud_model,
                               parallel.cores = 4,
                               Trace = T) %>% as.data.frame()

    # filtering Part of speech tagging
     annotatedText_df <- local_df %>%
      filter(upos %in% c("AUX","NOUN","ADJ","ADV","PART")) %>%
      select(doc_id, lemma) %>%
      group_by(doc_id) %>%
      summarise(plain_text = paste(lemma, collapse = " ")) %>%
      rename(accession_number = doc_id)

     # binding for every other filing, for further analysis
     plainText_df <- rbind(plainText_df, annotatedText_df)

  }
}
```

```r
saveRDS(plainText_df,"plainText_df.rds")
```

plain clean text has been joined with master index to get one consolidated data set for further analysis

```r
## for further analysis
plainText_df <- readRDS("plainText_df.rds")
sample_reports  <- plainText_df %>%
       left_join(indexMaster_df, by="accession_number")
```

b. Data reduction

Figure 3:Data Reduction process

by analyzing the corpus using boxplot for representing most dominant words and rare used words which can be seen as outliers can be added as stop words, moreover by removing stop words from the corpus will help in accurate analysis, on the other hand, with the help of outlier detection, term frequency cut off value percentage (threshold of elimination) was calculated by using zipf's law distribution.

```r
cikTF_IDF_df <- sample_reports %>%
  unnest_tokens(word, plain_text) %>%
  count(year_filed, word, sort=TRUE) %>%
  ungroup() %>%
  bind_tf_idf(word, year_filed, n)

cik_tf_idf_summarised <- cikTF_IDF_df %>%
  group_by(word) %>%
  summarise(avg_tf_idf = mean(tf_idf)) %>%
  arrange(desc(avg_tf_idf))

# plotting tf-idf Distribution and try with boxplot
ggplot(cik_tf_idf_summarised, aes(x=avg_tf_idf)) +
  geom_histogram(color="black", fill="black", bins=100)+
  scale_y_log10()+
  labs(title = "TF-IDF Distribution on log-scales")

## Warning: Transformation introduced infinite values in continuous y-axis
```

```
## Warning: Removed 56 rows containing missing values (geom_bar).
```

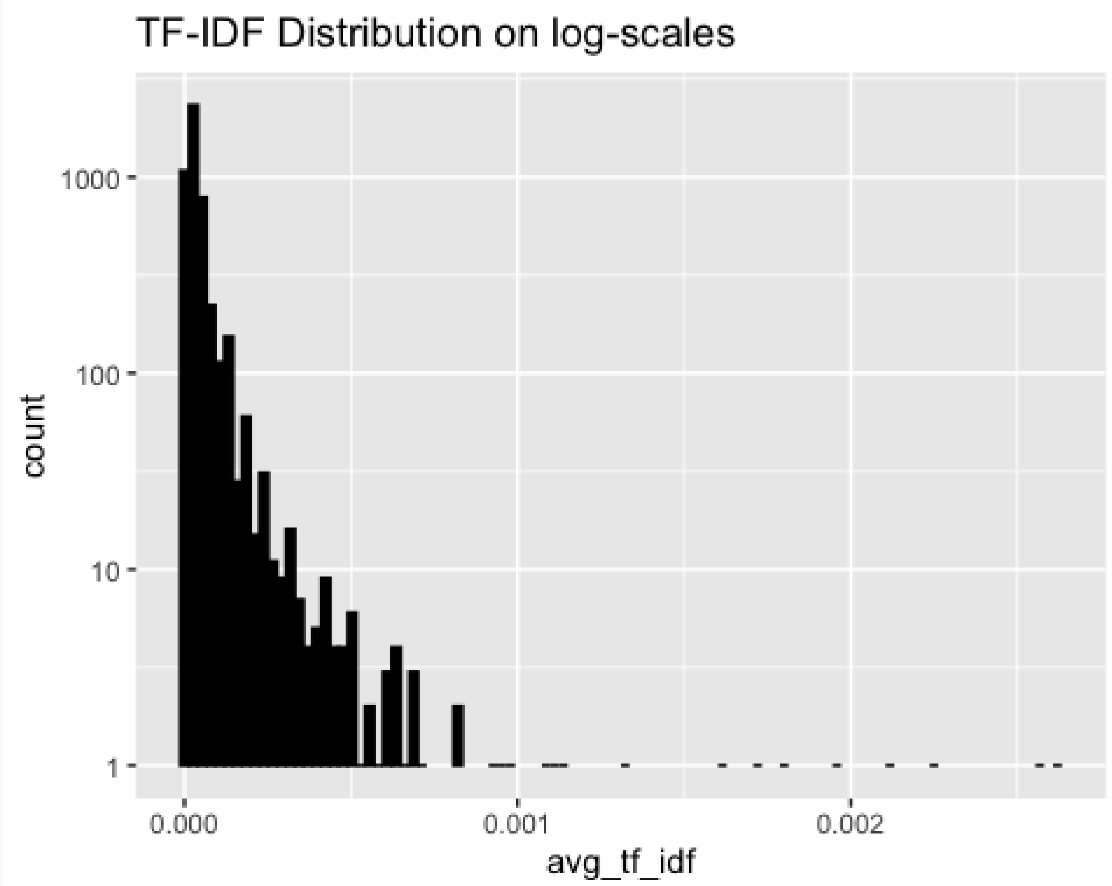

Figure 4:Distribution on log scale

```
hist(cikTF_IDF_df$tf_idf,
breaks = 100,
main = "Ideal range of TF-IDF",
xlab = "TF-IDF of Discussion tokens")
```

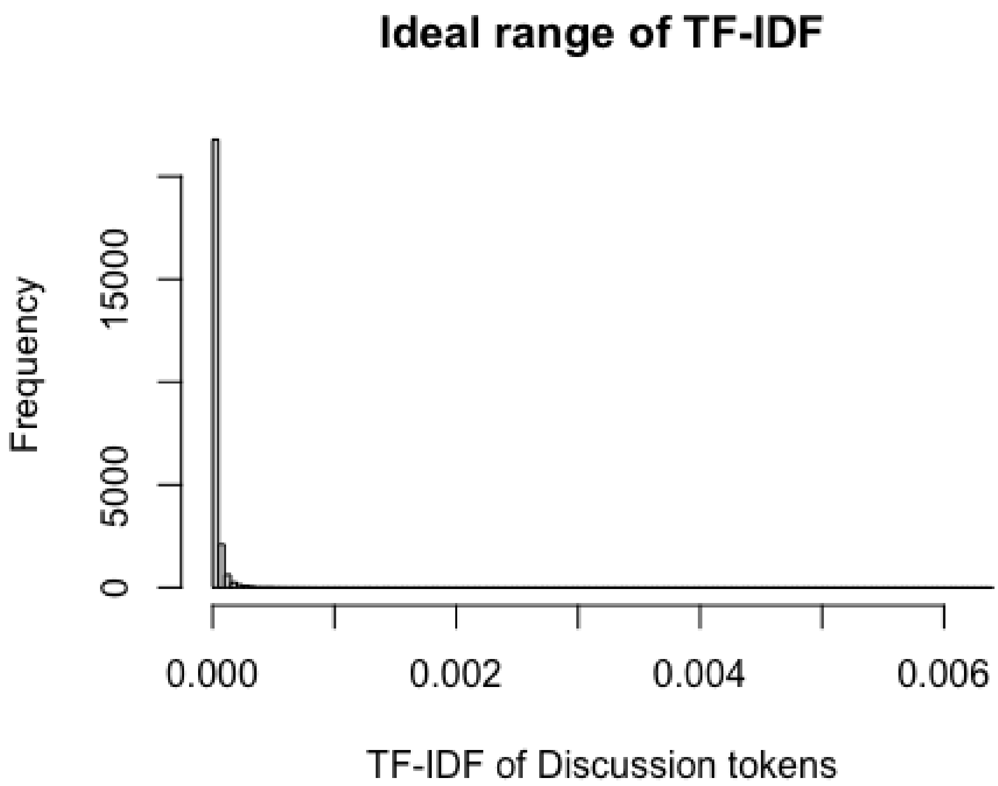

Figure 5: Frequency of Tokens

```
cikTF_IDF_df<-cikTF_IDF_df%>%
  filter(tf_idf>0.0015)

cikTF_IDF_df<-cikTF_IDF_df%>%
  filter(tf_idf<0.05)


# Adding buttom 10% words with the lowest TF-IDF as stop words
CIK_90P <- cik_tf_idf_summarised %>%
  top_frac(0.90) %>%
  arrange(desc(avg_tf_idf))

## Selecting by avg_tf_idf

CIK_10P <- cik_tf_idf_summarised %>%
  anti_join(CIK_90P) %>%
  arrange(desc(avg_tf_idf))

## Joining, by = c("word", "avg_tf_idf")

stopw_tfidf <- CIK_10P$word
stopwordsfinal_col <- c(stopwordsfinal_col, stopw_tfidf)
```

TF_IDF based on sub industry

```
tf_idf_samples_sub_industry <- sample_reports %>%
  unnest_tokens(word, plain_text) %>%
  count(GICS.Sub.Industry, word, sort=TRUE) %>%
  ungroup() %>%
  bind_tf_idf(GICS.Sub.Industry, word, n)

#Makes so much sense for the venue
tf_idf_samples_sub_industry.2<-sample_reports%>%
  unnest_tokens(word, plain_text, token = "ngrams", n=2)%>%
  group_by(word)%>%
  summarise(Count=n())%>%
  arrange(desc(Count))%>%
  top_n(20)

## Selecting by Count

tf_idf_samples_sub_industry.3<-sample_reports%>%
  unnest_tokens(word, plain_text, token = "ngrams", n=3)%>%
  group_by(word)%>%
  summarise(Count=n())%>%
  arrange(desc(Count))%>%
  top_n(10)

## Selecting by Count

summarised_tf_idf_sub_industry <- tf_idf_samples_sub_industry %>%
  group_by(word) %>%
  summarise(avg_tf_idf = mean(tf_idf)) %>%
  arrange(desc(avg_tf_idf))

# plotting tf-idf Distribution and try with boxplot
ggplot(summarised_tf_idf_sub_industry, aes(x=avg_tf_idf)) +
  geom_histogram(color="black", fill="black", bins=200)+
  scale_y_log10()+
  labs(title = "TF-IDF Distribution on log-scales for sub industry")

## Warning: Transformation introduced infinite values in continuous y-axis

## Warning: Removed 107 rows containing missing values (geom_bar).
```

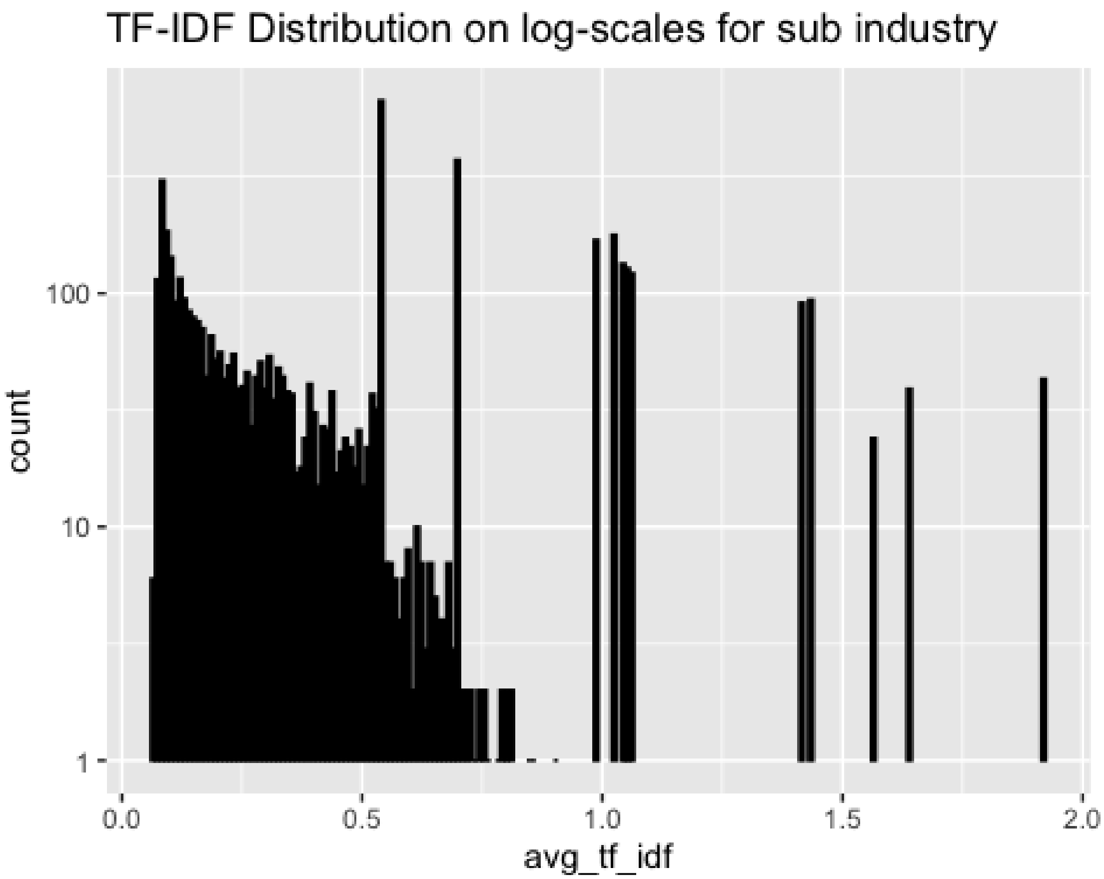

Figure 6: Distribution on log scales for Sub Industry

```
hist(tf_idf_samples_sub_industry$tf_idf,
breaks = 100,
main = "Ideal range of TF-IDF",
xlab = "TF-IDF of Discussion tokens")
```

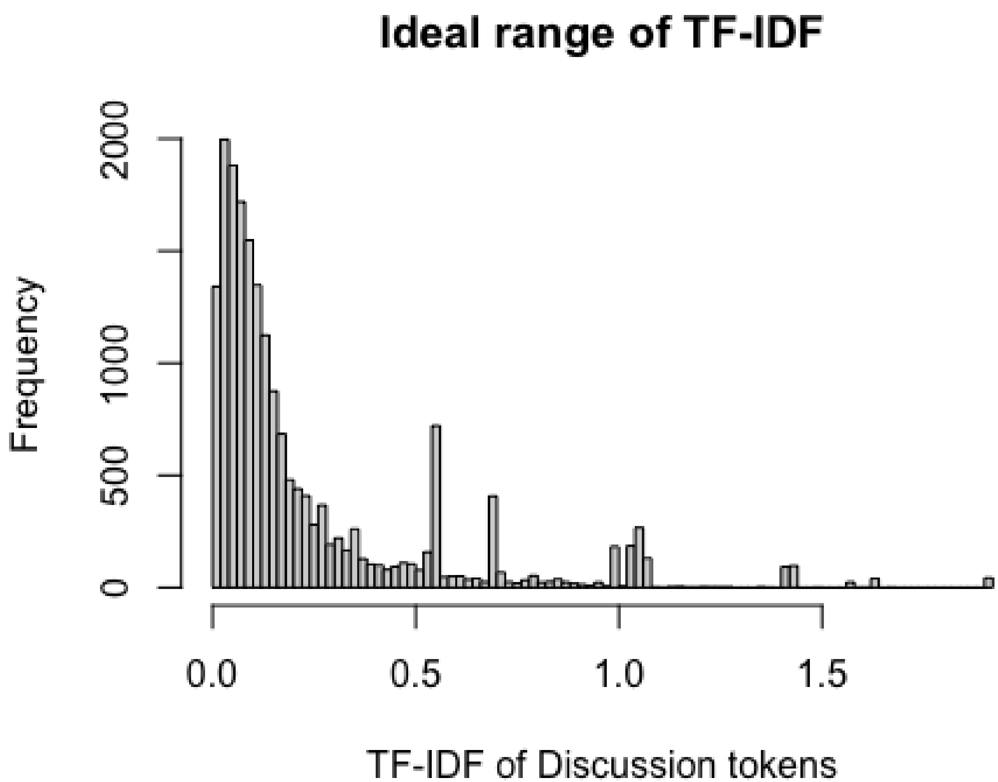

Figure 7: Frequency of tokens for sub industry

```
tf_idf_samples_sub_industry<-tf_idf_samples_sub_industry%>%
  filter(tf_idf>0.0075)

tf_idf_samples_sub_industry<-tf_idf_samples_sub_industry%>%
  filter(tf_idf<0.05)

# Adding buttom 10% words with the lowest TF-IDF as stop words
top_90_percent_sub_industry <- summarised_tf_idf_sub_industry %>%
  top_frac(0.90) %>%
  arrange(desc(avg_tf_idf))

## Selecting by avg_tf_idf

buttom_10_percent_sub_industry <- summarised_tf_idf_sub_industry %>%
  anti_join(top_90_percent_sub_industry) %>%
  arrange(desc(avg_tf_idf))

## Joining, by = c("word", "avg_tf_idf")

stopw_tfidf <- buttom_10_percent_sub_industry$word
stopwordsfinal_col <- c(stopwordsfinal_col, stopw_tfidf)
```

c. Stop words

stop words like “payment”, “authority”, “design”,“solution”,“credit”,“distribution” and 670 others, which are common and less meaningful, hence author took the right decision by removing these stop words

```
# 10k MDA Text Cleaning

 for(i in 1:nrow(sample_reports)){
   tryCatch({
    x<-sample_reports[i,]
    companyname <- strsplit(tolower(x[1])," ")[[1]]

    sample_reports$plain_text[i] <- x %>%
      unnest_tokens(word, plain_text) %>%
      filter(!word %in%  companyname) %>%
      filter(!word %in%  stopwordsfinal_col) %>%
      summarise(plain_text= paste(word, collapse=" "))

   }, error=function(e){cat("ERROR:", conditionMessage(e),"\n")})
 }
```

d. TF-IDF GICS sub industry level

By looking at the top terms at Sub industry level, it can be seen how TF-IDF plays important role in indentifying and beautifully displays important words through out. for instance

Technology, hardware, software and peripheral - Iphone, Ipad, mac Application software - online, subscription, autocad Communication equipments - distributor, configuration, lan, switch, router Electronic components - lcd, fiber, glass, display semiconductor equipments - solar, display, semiconductors IT Consulting and other services - indian, ruppee, clients, consult

no wonder why campanies outsource IT services from india, and indian consultant because maybe paygrade is low in india?.

```
# Tf idf GICS Sub Industry Level

sample_tokens <- sample_reports %>%
  unnest_tokens(word, plain_text) %>%
  count(GICS.Sub.Industry, word, sort=TRUE) %>%
  ungroup() %>%
  bind_tf_idf(word, GICS.Sub.Industry,n)  %>%
  mutate(token.length=nchar(as.character(word) ))%>%
  filter(token.length>2)

sample_tokens$token.length <- NULL
```

```
sample_tokens %>%
  arrange(desc(tf_idf)) %>%
  mutate(word=factor(word, levels=rev(unique(word)))) %>%
  group_by(GICS.Sub.Industry) %>%
  top_n(20) %>%
  ungroup %>%
  ggplot(aes(word,tf_idf,fill=GICS.Sub.Industry)) + geom_col(show.legend=FALS
E) +
  scale_x_reordered() +
  labs(x=NULL, y="tf-idf", title="Important Terms as per GICS industry for S&
P 500") +
  facet_wrap(~GICS.Sub.Industry, ncol=4, scales="free") + coord_flip()
```

e. Selecting by tf_idf

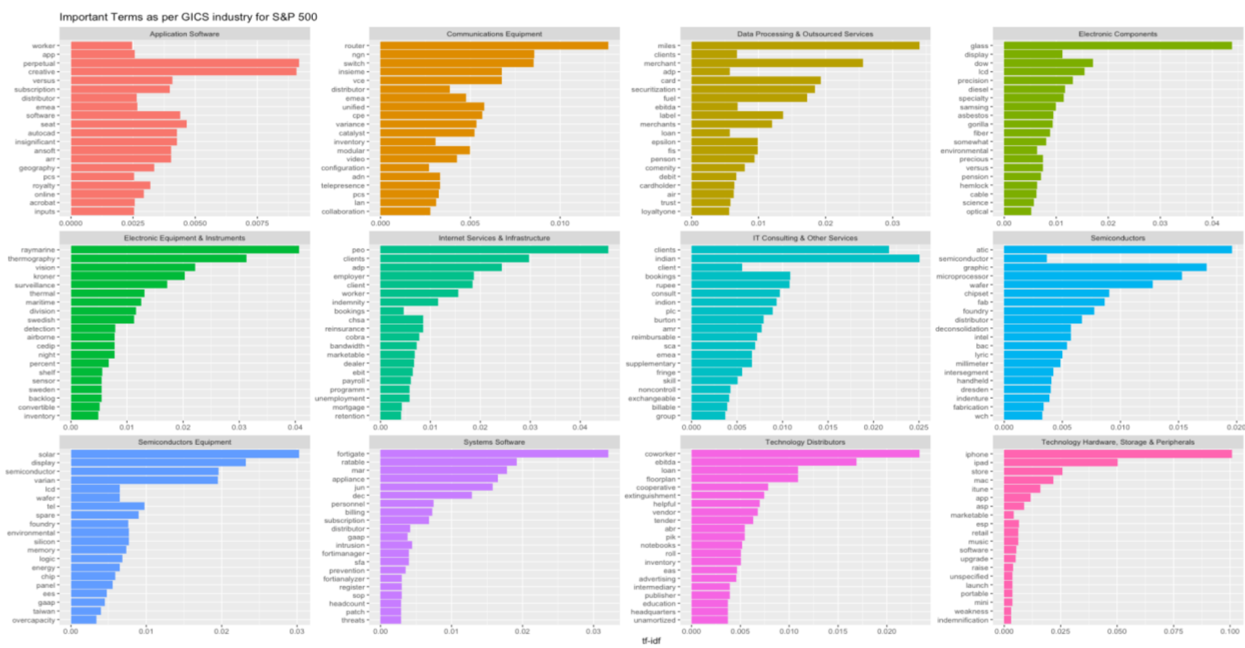

Figure 8: Important terms as per Sub Industry

TF-IDF year filed level

so, it can inferred from the below graph that iphone, ipad were pretty common terms in late 2013- 2015, maybe because it revolutinsed the whole cell phone and everybody at the management level was talking about it and captured, also in 2018 Cobra was the popular word, The Consolidated Omnibus Budget Reconciliation Act (COBRA) is a landmark federal law, passed in 1985, that provides for continuing group health insurance coverage for some employees and their families after a job loss or other qualifying event.

(2017-2019) - cobra (federal law for insurance coverage) (2017- 2019) - ebit (earnings before interest and taxes)

was most common words/terminologies in the brexit era(2016 - 2019)

```r
# Tf idf Yearlevel

sample_tokens <- sample_reports %>%
  unnest_tokens(word, plain_text) %>%
  count(year_filed, word, sort=TRUE) %>%
  ungroup() %>%
  bind_tf_idf(word, year_filed,n)  %>%
  mutate(token.length=nchar(as.character(word) ))%>%
  filter(token.length>2)

sample_tokens$token.length <- NULL

sample_tokens %>%
  arrange(desc(tf_idf)) %>%
  mutate(word=factor(word, levels=rev(unique(word)))) %>%
  group_by(year_filed) %>%
  top_n(20) %>%
  ungroup %>%
  ggplot(aes(word,tf_idf,fill=year_filed)) + geom_col(show.legend=FALSE) +
  theme_minimal() +
  scale_x_reordered() +
  scale_y_continuous(expand = c(0,0)) +
  labs(x=NULL, y="tf-idf", title="Important Terms per year for S&P 500") +
  facet_wrap(~year_filed, ncol=4, scales="free") + coord_flip()
```

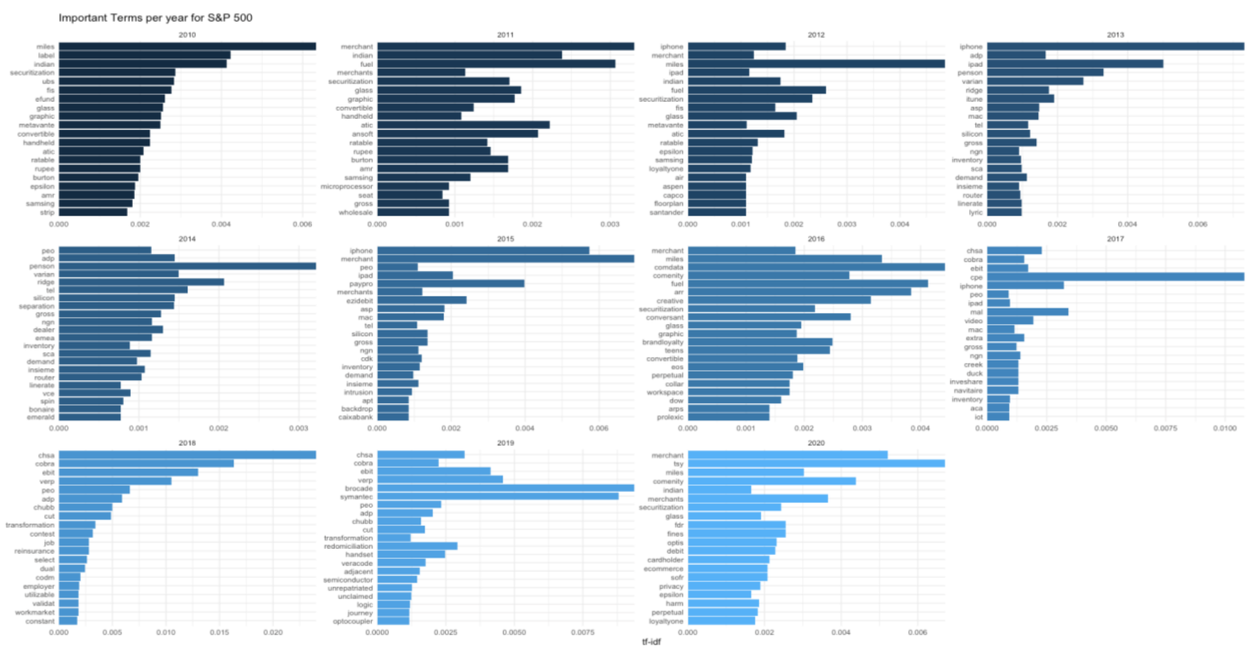

Figure 9: Important term per year

2. Part B Sentiment Analysis

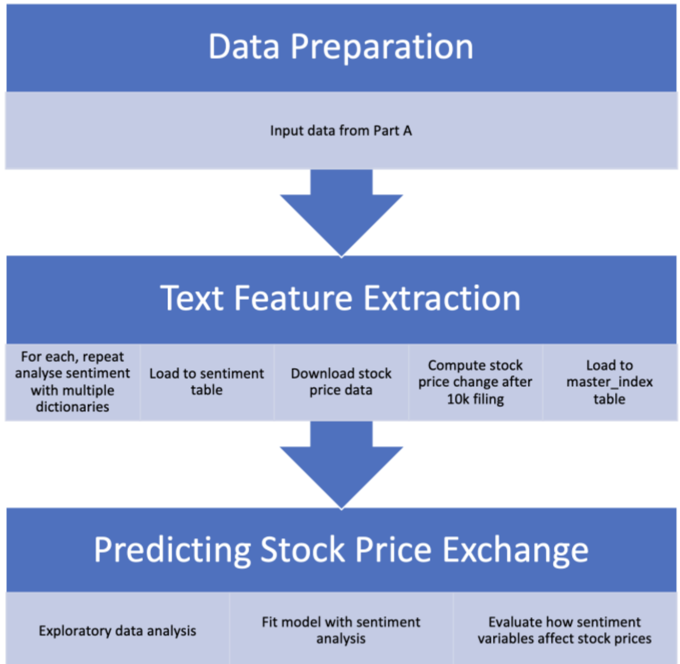

Figure 10: Pipeline for Part B

a. Data extraction

As the data was consolidated in above sections, same data will be used to get the sentiments of the filings, this section will conclude how the stock price reacts based on how sentimental the report is. for that author downloaded different dictionaries to analyze sentiments- like Loughran, nrc, bing, afinn. where Loughran-Mcdonald sentiments plays important role as it is built specifically for textual analysis related to finance

The sentiment categories are Constraining, Litigious, Positive, Negative, Superfluous, and Uncertainty etcetra.

```
   sentiment <- data.frame()
for(accessionnumber in sample_reports$accession_number){

  # selecting filing as per accession_number
   filing_df <- sample_reports %>%
    select(accession_number, plain_text, cik, year_filed, form_type) %>%
   filter(sample_reports$accession_number == accessionnumber) %>%
     mutate(plain_text= as.character(plain_text))


  processedTokens_df <- filing_df %>%
    unnest_tokens(word, plain_text)
```

```r
  words_df <- processedTokens_df %>%
    group_by(accession_number) %>%
    count(accession_number)

  ncomplex_df <- processedTokens_df %>%
    group_by(word, accession_number) %>%
    mutate(complexity=nchar(gsub("[^X]","",gsub("[aeiouy]+", "x",tolower(word
))))) %>%
    filter(complexity >=3) %>%
    group_by(accession_number) %>%
    count(accession_number)

  complexity <- tibble(accession_number=accessionnumber,
                       complexity = ncomplex_df$n / words_df$n)

  # LM sentiment dictionary
  LM_tokens <- processedTokens_df %>%
    inner_join(get_sentiments("loughran"))


  LM_wordCount <- LM_tokens %>%
    group_by(accession_number) %>%
    summarise(LM_totalWords= n())


  LM_sentiment <- LM_tokens %>%
    group_by(accession_number, sentiment) %>%
    summarise(total_sentiment= n())%>%
    spread(sentiment, total_sentiment, fill=0) %>%
    left_join(LM_wordCount) %>%
    mutate(sentiment.score=(positive-negative)/LM_totalWords)%>%
    mutate(polarity=(positive-negative)/(positive+negative))%>%
    mutate(polarity=ifelse(is.nan(polarity), 0, polarity))%>%
    mutate(polarity=ifelse(is.na(polarity), 0, polarity))%>%
    mutate(sentiment.score=ifelse(is.nan(sentiment.score), 0, sentiment.score
))

  LM_sentiment <- LM_sentiment %>%   mutate(litigious=ifelse("litigious" %in%
colnames(LM_sentiment), litigious, 0)) %>%
     mutate(constraining=ifelse("constraining" %in% colnames(LM_sentiment), c
onstraining, 0)) %>%
     mutate(litigious=ifelse("uncertainty" %in% colnames(LM_sentiment), uncer
tainty, 0))

   LM_sentiment <- LM_sentiment %>%  mutate(LM_sent = positive- negative,
           LM_positive = positive/LM_totalWords,
           LM_negative = negative/ LM_totalWords,
           LM_uncertainty = uncertainty/LM_totalWords,
```

```
           LM_litigious=litigious/LM_totalWords,
           LM_constraining=constraining/LM_totalWords) %>%
    select(-c(positive,negative, uncertainty,litigious))

  # sentiment r
  sentimentr <- tibble (accession_number = accessionnumber,
                        sentimentr = as.data.frame(sentiment_by(get_sentences
(filing_df$plain_text)))$ave_sentiment)

  # Afinn sentiment dictionary
  AFINN_sentiment <- processedTokens_df %>%
    inner_join(get_sentiments("afinn")) %>%
    group_by(accession_number) %>%
    summarise(afinn_sent = sum(value))

  # Bing sentiment dictionary
  BING_tokens <- processedTokens_df %>%
    inner_join(get_sentiments("bing"))

  BING_wordCount<- BING_tokens %>%
    group_by(accession_number) %>% summarise(BING_totalwords = n())

  BING_sentiment <- BING_tokens %>%
    group_by(accession_number, sentiment) %>%
    summarise(total_sentiment =n()) %>%
    spread(sentiment, total_sentiment, fill=0) %>%
    left_join(BING_wordCount) %>%
    mutate(bing_sent=positive-negative,
           bing_positve= positive/BING_totalwords,
           bing_negative = negative/BING_totalwords) %>%
    select(-c(positive,negative))

  #  NRC sentiment dictionary
  NRC_tokens <- processedTokens_df %>%
    inner_join(get_sentiments("nrc"))

  NRC_wordCount <- NRC_tokens %>% group_by(accession_number) %>% summarise(NR
C_totalWords=n())

  NRC_sentiment <- NRC_tokens %>%
    group_by(accession_number, sentiment) %>%
    summarise(total_sentiment =n()) %>%
    spread(sentiment, total_sentiment, fill=0) %>%
    left_join(NRC_wordCount) %>%
    mutate(nrc_sent = positive-negative,
           nrc_positive = positive/NRC_totalWords,
           nrc_negative = negative/NRC_totalWords,
           nrc_anger = anger/NRC_totalWords,
           nrc_fear = fear/NRC_totalWords,
```

```
            nrc_trust = trust/NRC_totalWords,
            nrc_sadness = sadness/NRC_totalWords,
            nrc_surprise = surprise/NRC_totalWords,
            nrc_disgust = disgust/NRC_totalWords,
            nrc_joy = joy/NRC_totalWords,
            nrc_anticipation= anticipation/NRC_totalWords) %>%
    select(-c(positive,negative, anger, trust, sadness,surprise, disgust, joy
, anticipation,fear))

  # syuzhet_vader sentiment dictionary
  SV_sentiment <- textfeatures( filing_df$plain_text , normalize = FALSE, wor
d_dims=FALSE, sentiment = TRUE) %>% select(sent_syuzhet, sent_vader) %>% muta
te(accession_number= accessionnumber)

  # merging sentiment features
  sentiment_df <- LM_sentiment %>%
    left_join(sentimentr, by = "accession_number") %>%
    left_join(AFINN_sentiment, by = "accession_number") %>%
    left_join(BING_sentiment, by = "accession_number") %>%
    left_join(NRC_sentiment, by = "accession_number") %>%
    left_join(SV_sentiment, by = "accession_number")

  # insert into sentiment table
  sentiment <- rbind(sentiment, sentiment_df)

}

saveRDS(sentiment,"sentiment.rds")
```

b. Download stock price

as soon as the 10k report is released to the public, the stock market will come into affect in 2-3 days so, it'll be beneficial to compare stock price of 7 days prior to filing and 3 days later to the filing date. based on that comparison assumption can be verified whether sentiment plays important role or not.

```
sentiment <- readRDS("sentiment.rds")

  sample_reports$ret_adjusted_price <- NA

  for (i in 1:nrow(sample_reports)){
    tryCatch({

      # get stock price information for every compnay symbol
      stock_data <- BatchGetSymbols(tickers = sample_reports$Symbol[i],
                                    first.date = sample_reports$date_filed[i]
-7,
                                    last.date = sample_reports$date_filed[i]
```

```r
+3,
                                               type.return = "log")
      # filter the 2nd day and the last day
      stockData_list <- stock_data[[2]] %>%
        filter(ref.date == max(ref.date)| row_number()==2) %>%
        arrange(desc(ref.date))

      # calculate stock price change on log scale
      ret_adjusted_price <-stockData_list$ret.closing.prices[1] - stockData_l
ist$ret.closing.prices[2] # return difference

      # Extract stock price
          sample_reports$ret_adjusted_price[i] <- ret_adjusted_price


    }, error = function(e){cat("error:",conditionMessage(e), "\n")})
  }

saveRDS(sample_reports,"sample_reports.rds")
```

c. Download Sentiments

```r
sentiment <- readRDS("sentiment.rds")
sample_reports <- readRDS("sample_reports.rds")
# Exploratory Data Analysis
sentiment_data <- sentiment %>%
  inner_join(sample_reports, by="accession_number") %>%
  mutate(year_filed=as.factor(year_filed),
         company_name = as.factor(company_name),
         cik = as.factor(cik),
         GICS.Sub.Industry = as.factor(GICS.Sub.Industry),
         accession_number = as.factor(accession_number))

sentiment_data <- sentiment_data %>%
  filter(LM_totalWords >10 , BING_totalwords>10, NRC_totalWords >10) %>%
  filter(ret_adjusted_price != !is.na(ret_adjusted_price)) %>%
  mutate_if(is.numeric, funs(ifelse(is.na(.),0,.)))

plot_missing(sentiment_data)

plot_histogram(sentiment_data)
```

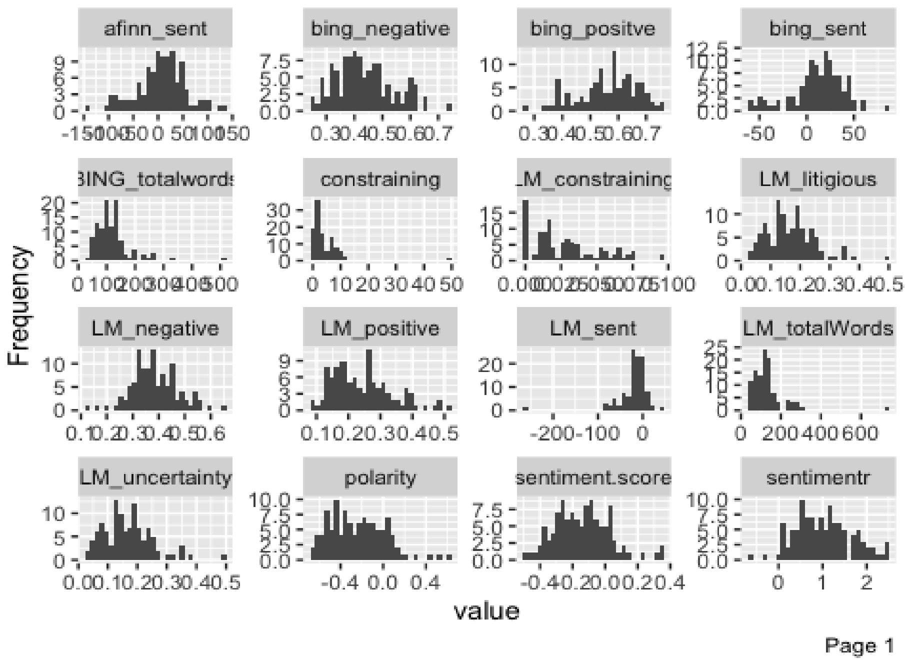

Figure 11: Frequency per sentiment

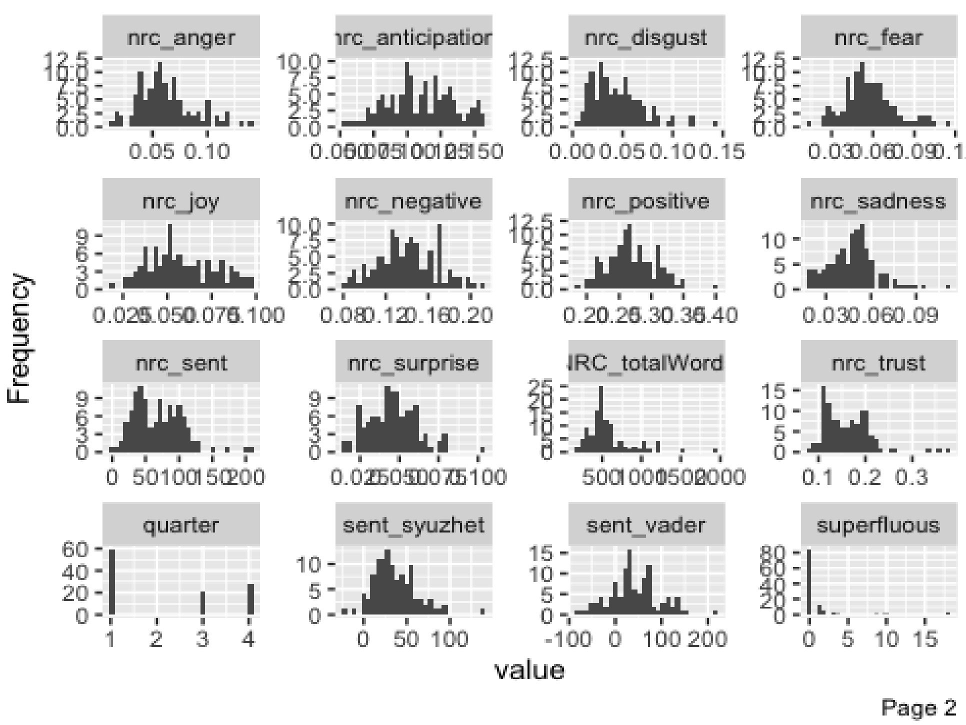

Figure 12: frequecny per sentiment

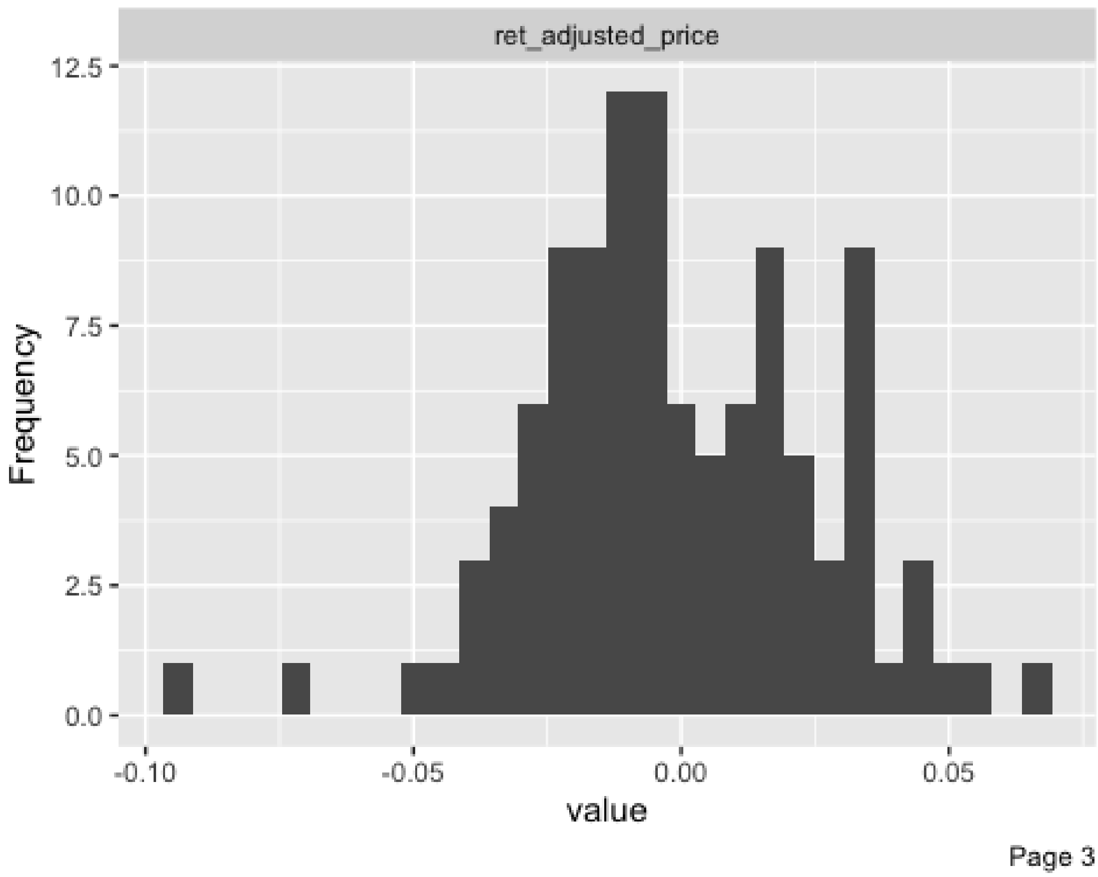

Figure 13: Frequency of Return adjusted price

```
# plot average return
return_avg <- sentiment_data %>%
  group_by (GICS.Sub.Industry, year_filed, form_type) %>%
  summarise(ret_adjusted_price = mean(ret_adjusted_price))

## `summarise()` has grouped output by 'GICS.Sub.Industry', 'year_filed'. You
can override using the `.groups` argument.

# plot average stock price change
price_avg <- sentiment_data %>%
  group_by(GICS.Sub.Industry, year_filed, form_type) %>%
  summarise(ret_adjusted_price=mean(ret_adjusted_price))

## `summarise()` has grouped output by 'GICS.Sub.Industry', 'year_filed'. You
can override using the `.groups` argument.

# Individual dictionaries

sentiment_regaression_data <- sentiment_data %>% na.omit() %>%
  select(-c(form_type, year_filed, cik, company_name, GICS.Sub.Industry, acce
ssion_number ))
```

d. How sentiment analysis affects stock price change

```
## Adding missing grouping variables: `accession_number`

LM_reg10k <- lm(ret_adjusted_price ~ LM_totalWords + LM_sent + LM_positive +
LM_negative + LM_uncertainty + LM_litigious + LM_constraining  , data = senti
ment_regaression_data)

BING_reg10k <- lm(ret_adjusted_price ~ BING_totalwords +bing_sent + bing_posi
tve + bing_negative, data = sentiment_regaression_data)

AFINN_reg10k <- lm(ret_adjusted_price ~ afinn_sent , data = sentiment_regares
sion_data)

NRC_reg10k <- lm(ret_adjusted_price ~ nrc_sent + NRC_totalWords + nrc_positiv
e +nrc_negative + nrc_anger + nrc_fear + nrc_trust + nrc_sadness + nrc_surpri
se + nrc_disgust + nrc_joy + nrc_anticipation, data = sentiment_regaression_d
ata)

SYUZHET_reg10k <- lm(ret_adjusted_price ~ sent_syuzhet, data = sentiment_rega
ression_data)

VADER_reg10k <- lm(ret_adjusted_price ~ sent_vader, data = sentiment_regaress
ion_data)


library(stargazer)
```

```
------------------------------------------------------------------------------------------------------------
Observations             109              109              109              109              109
R2                      0.037            0.013            0.014            0.006            0.0003
Adjusted R2             0.020           -0.015            0.005           -0.004           -0.009
Residual Std. Error  0.030 (df = 102)  0.030 (df = 105)  0.029 (df = 107)  0.029 (df = 107)  0.029 (df = 107)
F Statistic        0.650 (df = 6; 102) 0.452 (df = 3; 105) 1.514 (df = 1; 107) 0.620 (df = 1; 107) 0.027 (df = 1; 107)
============================================================================================================
Note:                                                                        *p<0.1; **p<0.05; ***p<0.01
```

Figure 14:Model Evaluation

Conclusion

According to the result shown above, out of every dictionary Loughran and McDonald's has significance of approx 0.04 (r squared) (40% variance) and p-value is significant in predicting the stock price, as Loughran and Mcdonald's is based on financial data, also different dictionary plays different role in analyzing sentiment, it proves our assumption that for financial companies are better predicted by Loughran and McDonald's sentiment. Next steps suggestions would be to increase the sample size of the dataset by including more companies and also, increasing the time span from previous annual reports.

3. Part C Topic Modeling

Part C Process Workflow

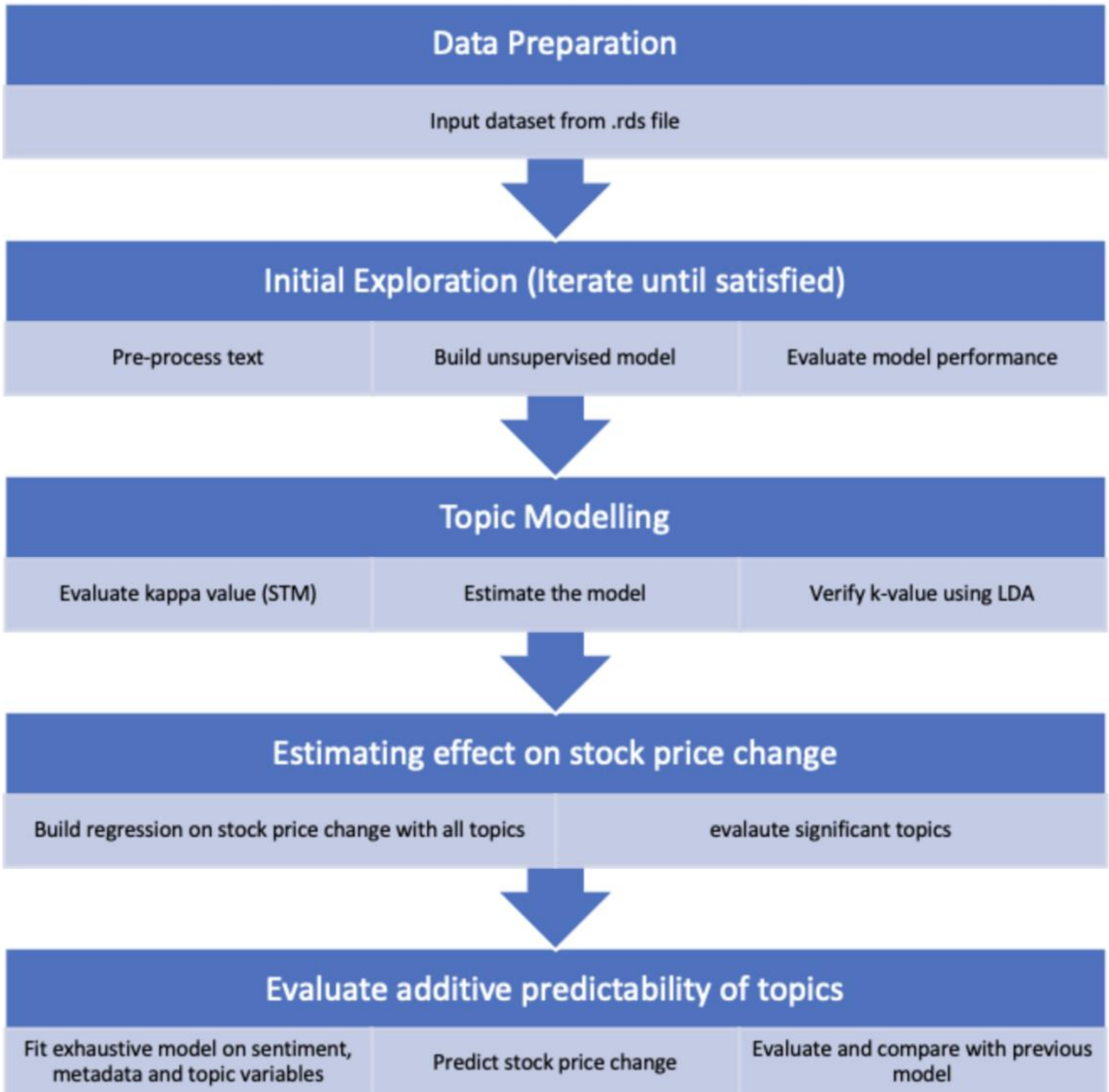

Figure 15: Part C Process Workflow

a. Data Exploration

continuing with the above consolidated and cleaned data from part b, topic modeling is the last step of text mining, where topics should explain the whole corpus based on the variable, author decided to go with unsupervised topic modeling first and then based on suggested kappa, will approximate number of topics for supervised kappa as second iteration of udpipe annotation is required to further clean the text and take only part of speech which are noun, adjective, and adverb to visualize the words with in the topic more carefully

```
sample10k_df <- sample_reports  %>%
  mutate(date_filed = as.Date(date_filed, origin="1970-01-01")) %>% mutate(ci
k = as.factor(cik), company_name= as.factor(company_name), GICS.Sub.Industry=
as.factor(GICS.Sub.Industry), form_type=as.factor(form_type), year_filed= as.
factor(year_filed), accession_number = as.factor(accession_number)) %>% na.om
it()

set.seed(199)

#Part of speech tagging
pos.tagged<-udpipe_annotate(as.character(sample10k_df$plain_text),
                            doc_id = sample10k_df$accession_number,
                            object = ud_model) %>% as.data.frame()

saveRDS(pos.tagged, "pos.tagged.rds")

pos.tagged <- readRDS("pos.tagged.rds")


#Pos tagged report
pos.tagged<-pos.tagged%>%
  filter(upos %in% c("NOUN","ADJ","ADV")) %>%
  group_by(doc_id)%>%
  summarise(plain_text=paste0(token, collapse = " "))%>%
  select(plain_text, accession_number=doc_id)

# cleaned text
sample10k_df$plain_text <- as.character(sample10k_df$plain_text)

sample_10k<-sample10k_df%>%
  left_join(pos.tagged)

## Joining, by = c("accession_number", "plain_text")

# corpus preperation
processed<- textProcessor(sample_10k$plain_text,
                          metadata = sample_10k,
                          customstopwords = c("net","product","service","marg
in","volume","revenue"), stem=FALSE)

threshold <- round(1/100* length(processed$documents),0)
sample10k_out <- prepDocuments(processed$documents,
                         processed$vocab,
                         processed$meta,
                         lower.thresh = threshold)

## Removing 1145 of 4310 terms (1145 of 47668 tokens) due to frequency
## Your corpus now has 109 documents, 3165 terms and 46523 tokens.
```

```r
# stm model fitting
unsupervised10k_stm <- stm(documents = sample10k_out$documents,
                           vocab = sample10k_out$vocab,
                           K=0,
                           prevalence = NULL,
                           max.em.its = 150,
                           data=sample10k_out$meta,
                           reportevery =5,
                           sigma.prior = 0.7,
                           init.type = 'Spectral')

saveRDS(unsupervised10k_stm, "unsupervised10k_stm.rds")

unsupervised10k_stm <- readRDS("unsupervised10k_stm.rds")

unsupervised10k_stm_theta <- as.data.frame(unsupervised10k_stm$theta)

colnames(unsupervised10k_stm_theta) <- paste0("topic", 1:ncol(unsupervised10k
_stm_theta))

unsupervised10k_stm_theta$best.estimator <- unlist(lapply(1:nrow(unsupervised
10k_stm_theta), function(i){
  which(unsupervised10k_stm_theta[i,] == max(unsupervised10k_stm_theta[i,]))
}))

unsupervised10k_stm_theta %>%
  group_by(best.estimator) %>%
  summarise(Count=n()) %>%
  arrange(desc(Count))

# review performance

unsupervisedTopic_summary <- summary(unsupervised10k_stm)

## A topic model with 83 topics, 109 documents and a 3165 word dictionary.

# plot the topic model
plot(unsupervised10k_stm)
```

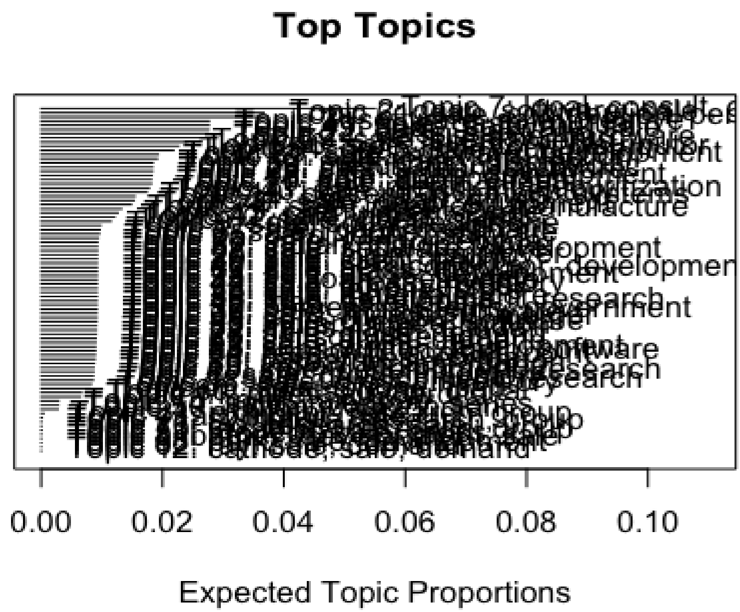

Figure 16: Top Topics proportion

```
# review topic semantic coherence
topicQuality(unsupervised10k_stm, documents = sample10k_out$documents)
```

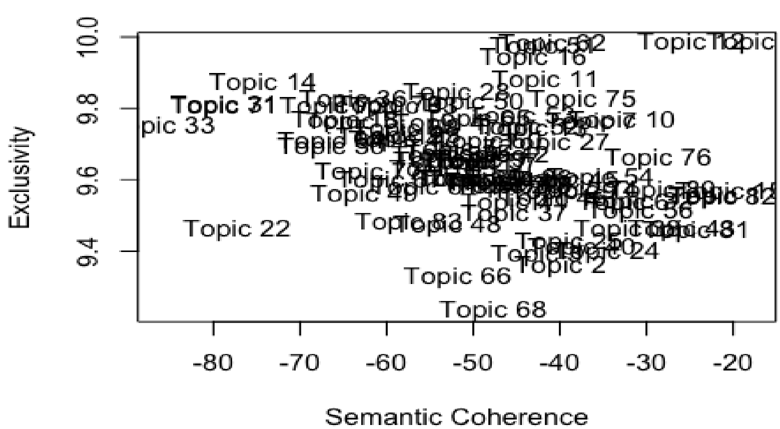

Figure 17: Topic Semantic coherence

```
# review word frequency to identify potential stopwords

kappa_unsp <- length(unsupervisedTopic_summary$topicnums)
topWords_topic <- c()
for( i in 1:kappa_unsp){
  topWords_topic <- c(topWords_topic, unsupervisedTopic_summary$prob[i,])
}
```

b. Decide on kappa

as the Unsupervised model suggested 90 number of topics from initial exploration, by using searhk, we can check the neighborhood around it, by giving the range of kappa (+10, -10), irrespective of different aggregation, kappa (k) value should lie in this given range

```
# deciding on k number of topics
# kappa_unsp is 90
expected_k <- c(kappa_unsp-10, kappa_unsp-6, kappa_unsp-2,kappa_unsp, kappa_u
nsp+2, kappa_unsp+6, kappa_unsp+10 )

searchK_result <- searchK(sample10k_out$documents, sample10k_out$vocab, expec
ted_k)

saveRDS(searchK_result, "sk_result.rds")
```

c. Optimal kappa

kappa is based on high held-out likelihood, low residuals and high semantic coherence, hence by seeing the graph it is easy to set the value of kappa as for the supervised topic modeling as 80

```
searchK_result <- readRDS("sk_result.rds")

plot(searchK_result)
```

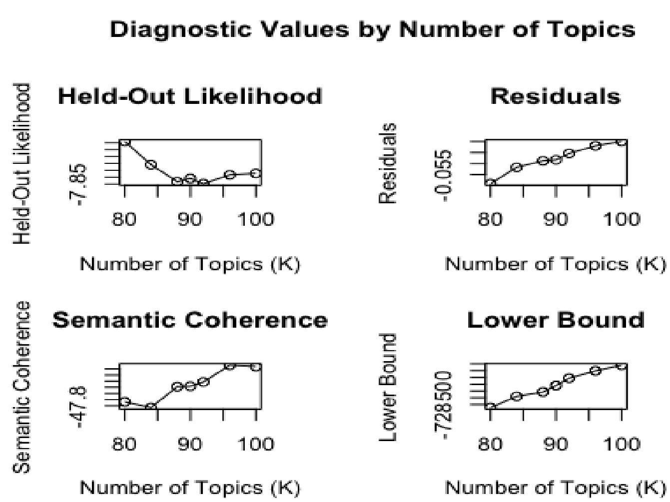

Figure 18: Diagnostic values by number of topics

```
set.seed(199)
# corpus preparation
processed <- textProcessor(sample10k_df$plain_text,
                           metadata = sample10k_df,
```

```
                           customstopwords = c("net","product","service","mar
gin","volume","revenue","inventory"), stem=FALSE)

threshold <- round(1/100* length(processed$documents),0)
sample10k_out <- prepDocuments(processed$documents,
                         processed$vocab,
                         processed$meta,
                         lower.thresh = threshold)

## Removing 1145 of 4309 terms (1145 of 47603 tokens) due to frequency
## Your corpus now has 109 documents, 3164 terms and 46458 tokens.

# stm model fitting
supervised10k_stm <- stm(documents = sample10k_out$documents,
                            vocab = sample10k_out$vocab,
                            K=80,
                            prevalence = ~factor(GICS.Sub.Industry) + factor(
year_filed),
                            max.em.its = 150,
                            data=sample10k_out$meta,
                            reportevery =5,
                            sigma.prior = 0.7,
                            init.type = 'Spectral')

saveRDS(supervised10k_stm, "supervised10k_stm.rds")

supervised10k_stm <- readRDS("supervised10k_stm.rds")

supervisedTopic_summary <- summary(supervised10k_stm)

# evaluate supervied modelperformance
# plot the topic model
plot(supervised10k_stm)
```

d. review topic semantic coherance

```
topicQuality(supervised10k_stm, documents = sample10k_out$documents)
```

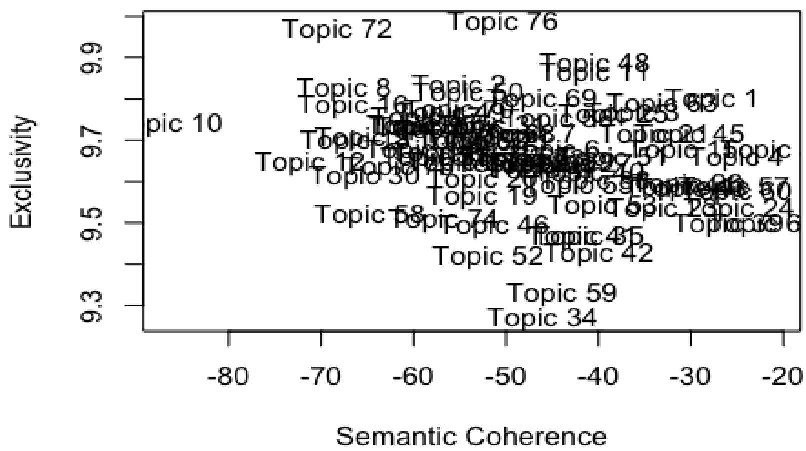

Figure 19:Topic quality semantic coherence

```
# review word frequency to identify potential stopwords_dict
topWords_topic <- c()
kappa_sp <- length(supervisedTopic_summary$topicnums)

for(i in 1:kappa_sp){
  topWords_topic <- c(topWords_topic, supervisedTopic_summary$prob[i,])

}

data.frame(word=topWords_topic) %>%
  group_by(word) %>%
  summarise(count=n()) %>%
  arrange(desc(count)) %>%
  top_n(50) %>%
  mutate(word = factor(word, word)) %>%
  ggplot(aes(x=reorder(word, count), y=count)) + geom_bar(stat="identity") +
coord_flip() +labs(title='occerrance of highest probable words across topics'
, subtitle = paste('a result of supervised stm with', kappa_sp,'number of top
ics'), x='Word')

## Selecting by count
```

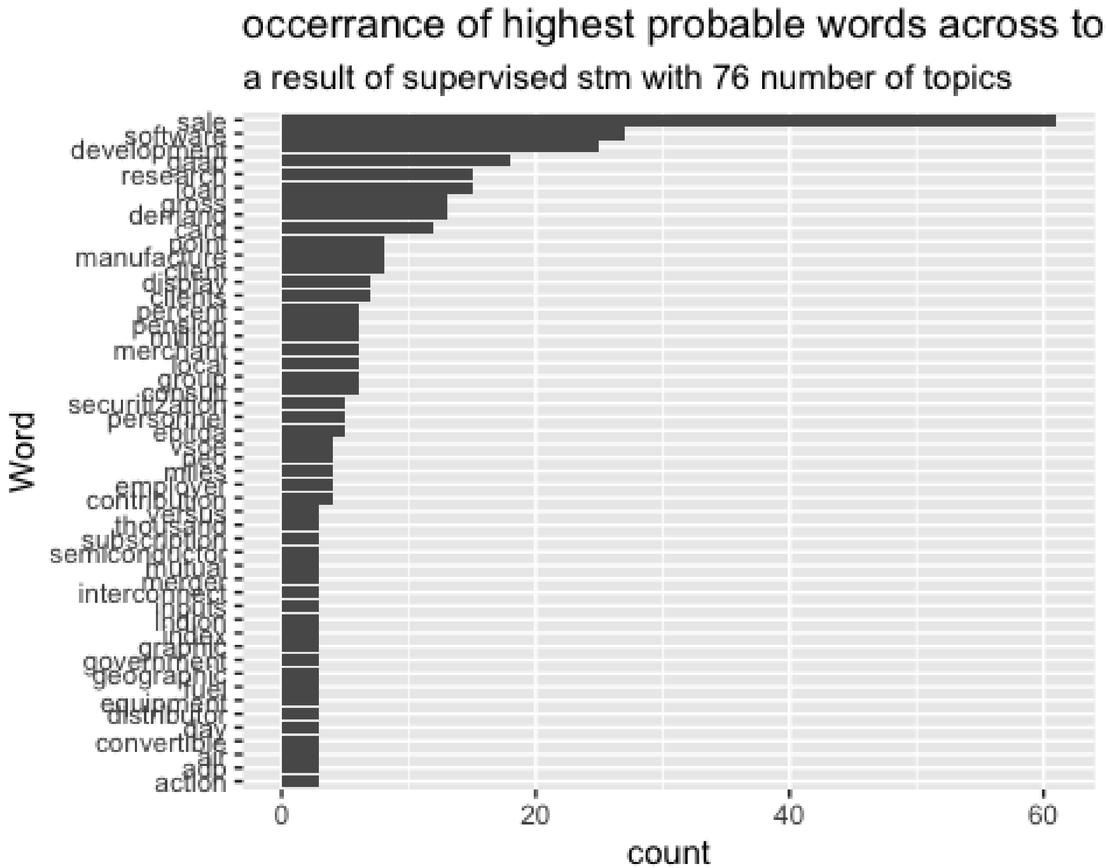

Figure 20:occurrence of highest possible word

```
# review FREX words for every topic
topic_proportions <- colMeans(supervised10k_stm$theta)
frexWords_Supervised <- data.frame()

for(i in 1:length(supervisedTopic_summary$topicnums)){

  row_here <- tibble(topicnum = supervisedTopic_summary$topicnums[i],
                     proportion = 100*round(topic_proportions[i],4),
                     frex_words= paste(supervisedTopic_summary$frex[i,1:7],
                                       collapse=","))
  frexWords_Supervised <- rbind(row_here, frexWords_Supervised)

}
rm(row_here)

frexWords_Supervised %>%
  arrange(desc(proportion)) %>%
  filter(topicnum ==16)
```

in comparison to unsupervised topic modeling, here are some of the topics which has high proportion ,

topic 16 – sale, gaap, regulatory, corporation

topic 6 – research, equipment, manufacture

topic1 – Iphone, ipad, retail, store

topic 73 – Indian, rupee, expansion, demand

these all above topics corelate to the corpus, which define certain industries like – technology hardware, financial terms, outsourcing services

```
stm::cloud(supervised10k_stm, topic=16)
```

Figure 21:Topic 16 word cloud

```
stm::cloud(supervised10k_stm, topic=6)
```

Figure 22:topic 6 word cloud

```
stm::cloud(supervised10k_stm, topic=1)
```

Figure 23:Topic 1 Word cloud

```
stm::cloud(supervised10k_stm, topic=73)
```

Figure 24:Topic 73 Word cloud

```
 #Effects estimation
supervised10k_stm.effects <- estimateEffect(~year_filed+cik,
 stmobj = supervised10k_stm, metadata = sample10k_out$meta)

 plot(supervised10k_stm.effects, covariate = "year_filed",
     topics = c(1:6),
     model = supervised10k_stm,
     method = "difference",
     cov.value1 = "2020",
     cov.value2 = "2010",
     xlab = "Low Rating ... High Rating",

     main = "Effects - year aggregation",
     labeltype = "custom")
```

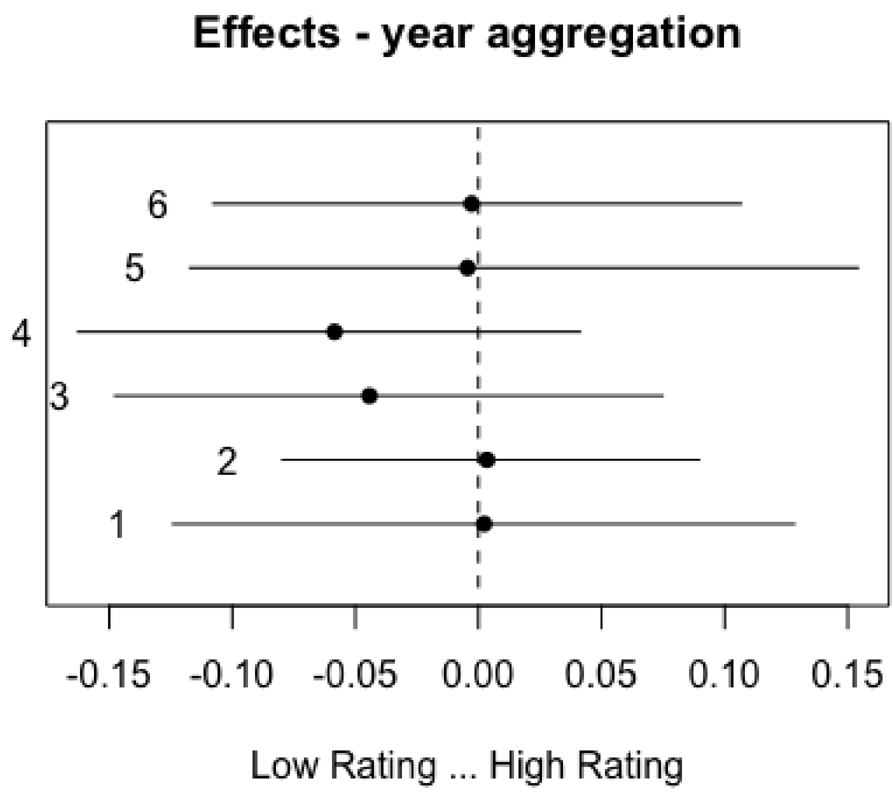

Figure 25:Year aggregation

The topics are aggregated with return adjustment price in order to determine the amount of variance, as this data is very granular and 6 topics are predicting the same price wil cause loss of information

```
# all topic effect

convergence <- as.data.frame(supervised10k_stm$theta)
colnames(convergence)<- paste0("topic", 1:76)

regression_data <- cbind(sample10k_out$meta, convergence) %>% na.omit() %>%
select(-c(plain_text, accession_number, date_filed, cik, company_name, form_t
ype, GICS.Sector, edgar_link))


str(regression_data)

# topic effect singular

library(stargazer)

reg_topic16 <- lm(ret_adjusted_price ~ topic16, data = regression_data)
```

```r
reg_topic73 <- lm(ret_adjusted_price ~ topic73, data = regression_data)

reg_topic1 <- lm(ret_adjusted_price ~ topic1, data = regression_data)
stargazer::stargazer(reg_topic16,reg_topic73,reg_topic1, type="text")

# plot most significant topics of 16, 73, 1 in cloud words
cloud(supervised10k_stm, topic=16, type=c("model"))
```

Figure 26:Topic 16 Word Cloud

```r
cloud(supervised10k_stm, topic=73, type=c("model"))
```

Figure 27:Topic 73 Word Cloud

```
cloud(supervised10k_stm, topic=1, type=c("model"))
```

Figure 28:Topic 1 Word Cloud

Topic 1 topic 73, topic 73 are significantly affecting thee stock price change, and from the regression model topic 1 significantly accounts for 2% change

```
# estimate effects : topics proportions over time
sample10k_out$meta$year_filed <- as.numeric(sample10k_out$meta$year_filed )

effects_10k <- estimateEffect(~factor(GICS.Sub.Industry)+s(year_filed),
                              stmobj= supervised10k_stm,
                              metadata = sample10k_out$meta)
```

```
## Warning in estimateEffect(~factor(GICS.Sub.Industry) + s(year_filed), stmo
bj = supervised10k_stm, : Covariate matrix is singular.  See the details of ?
estimateEffect() for some common causes.
##              Adding a small prior 1e-5 for numerical stability.
```

```
convergence <- as.data.frame(supervised10k_stm$theta)
colnames(convergence)<- paste0("topic", 1:76)

meaningful_topic <- c(16, 73, 1)
meaningful_topiclabel <- c("financial terms","outsourcing services","Technolo
gy & hardware")

for(t in 1:length(meaningful_topic)){

  plot(effects_10k, covariate = "year_filed",
       topics=meaningful_topic[t],
       model = supervised10k_stm, method="continuous",
       xaxt='n',
       xlab="year filed",
       main= paste('topic',meaningful_topic[t],':',meaningful_topiclabel[t]),
       printlegend = FALSE,
       linecol="black",
       labeltype="none")

  axis(1,at=seq(from=1, to=length(unique(sample10k_out$meta$year_filed)),
                by=1), labels=c("2010","2011","2012","2013","2014","2015","20
16","2017","2018","2019","2020"))
}
```

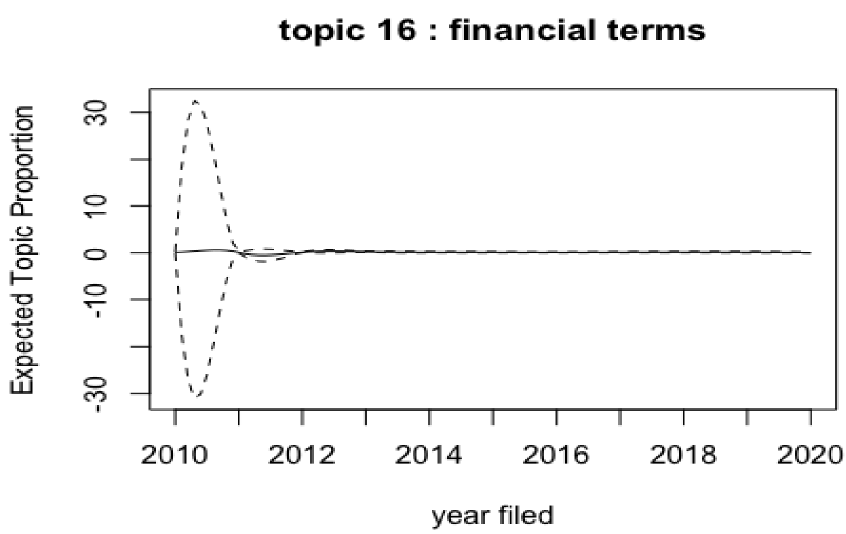

Figure 29: effect of topic 16 per year

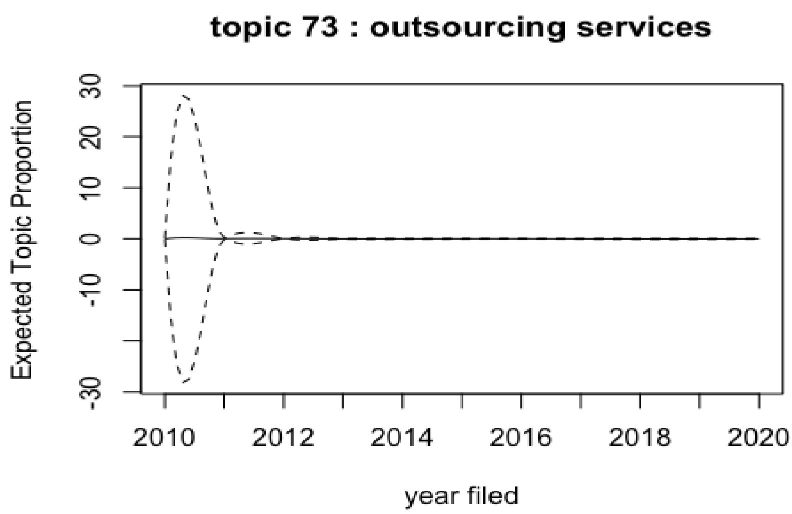

Figure 30:Effect of topic 73 per year

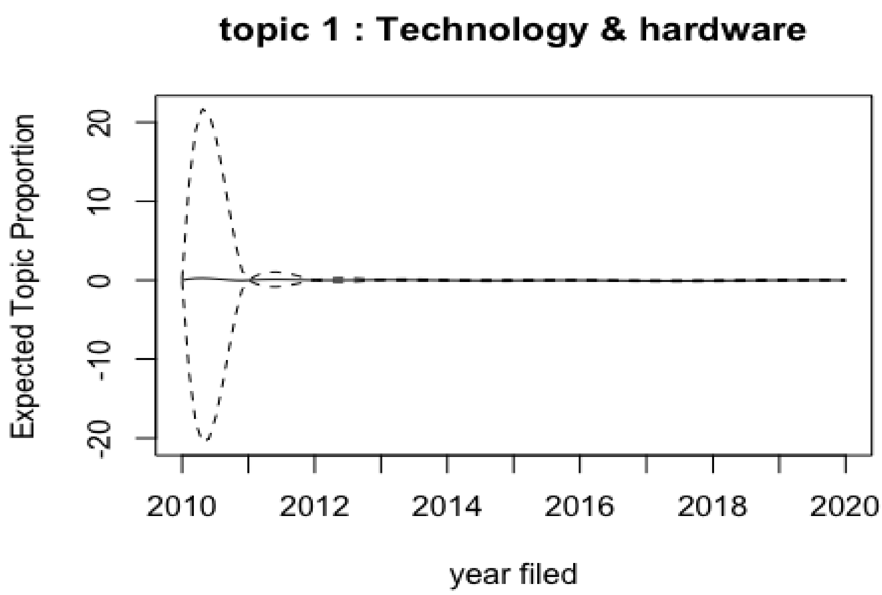

Figure 31:effect of topic 1 per year

```
sample10k_out$meta$year_filed <- as.factor(sample10k_out$meta$year_filed)
# model fitting as addition
```

e. Addictive Predictabily on estimating stock prices

After estimating return adjusted score when regressed with topics, the variance for topic 1 was significant and p value came out to be ***(p<0.01) or it can be predicted that higher level of aggregation with topic wont provide good results

```
full_regData <- cbind(sample10k_out$meta, convergence) %>%
  select(cik, company_name, accession_number, GICS.Sub.Industry, year_filed,
topic16, topic73, topic1) %>%  na.omit() %>% left_join(sample_reports %>% sel
ect(-c(year_filed,GICS.Sub.Industry, company_name, plain_text, GICS.Sector, f
orm_type,edgar_link, date_filed)), by='accession_number') %>% ungroup() %>% d
rop_na()
str(full_regData)

model_10k_with_topic <- lm(ret_adjusted_price ~., data=full_regData, na.actio
n = na.exclude)

summary(model_10k_with_topic)

save(model_10k_with_topic, file="model_10k_with_topic.rds")
save(full_regData, file="full_regData.rds")
```

Citation

```
citation(pdftools)
citation(textshape)
citation(edgar)
citation(lubridate)
citation(rvest)
citation(readr)
citation(tidyverse)
citation(data.table)
citation(dplyr)
citation(textreadr)
citation(stringr)
citation(cld3)
citation(tidytext)
citation(textcat)
citation(doParallel)
citation(udpipe)
citation(stm)
citation(wordcloud)
citation(foreach)
citation(qdap)
citation(tm)
citation(rlist)
```

```
citation(tidyr)
citation(keyholder)
citation(qdapDictionaries)
citation(textstem)
citation(textclean)
citation(ggplot2)
citation(reshape2)
citation(SentimentAnalysis)
citation(stm)
citation(topicmodels)
citation(wordcloud)
citation(textdata)
citation(emmeans)
citation(textfeatures)
citation(sentimentr)
citation(BatchGetSymbols)
citation(DataExplorer)
```