Text Analytics for Corporate Social Responsibility

Overview

In this report, I will utilize text analytics on the annual 2019 Corporate Social Responsibility (CSR) reports submitted by Apple, Google, and Microsoft. The CSR report represents information that explains how companies are giving back to the community, pushing towards sustainability and other related practices or goals they look to achieve in the direction of being a better company for the people, society, and environment. Each of these companies dominates the technology industry, with their ability to collect data on their customers. However, the tech industry is under scrutiny with data breaches (Google & Facebook) and battery performance scandals (Apple) in recent years.

Tokenization

Tokenizing each companies report gives us a better understanding of the main topic of focus. It is important to note that Microsoft's CSR report is half the length compared to Apple and Google. Below are the top 3 most frequently used words in their respective CSR reports.

Apple	**Google**	**Microsoft**
Energy (341)	Energy (332)	Learn (94)
Renewable (197)	Data (165)	AI (63)
Data (176)	Carbon (146)	Human (52)

Apple and Google have very similar top occurring words – Energy and Data. Apple and Google both are large enterprises that have to store and collect user data every minute. The focus topics of *energy* and *data* signals that they are trying to reassure that they are taking strong energy-saving procedures to manage their data intake. With databases and data centers scattered all over the world, it is reassuring to see that they are taking the necessary steps to improve energy consumption. In comparison, we look at Microsoft's top 3 words, and they are more focused on the actual usability of their technology as it relates to *learn, AI ,and human*. These points position Microsoft's report in finding ways to utilize their technology advancements for ethical practices.

Sentiment Analysis

Understanding the tone of the language is critical in determining what direction the company is trying to relate to their audience. Below is a breakdown of the sentiment used in the reports using R's nrc library.

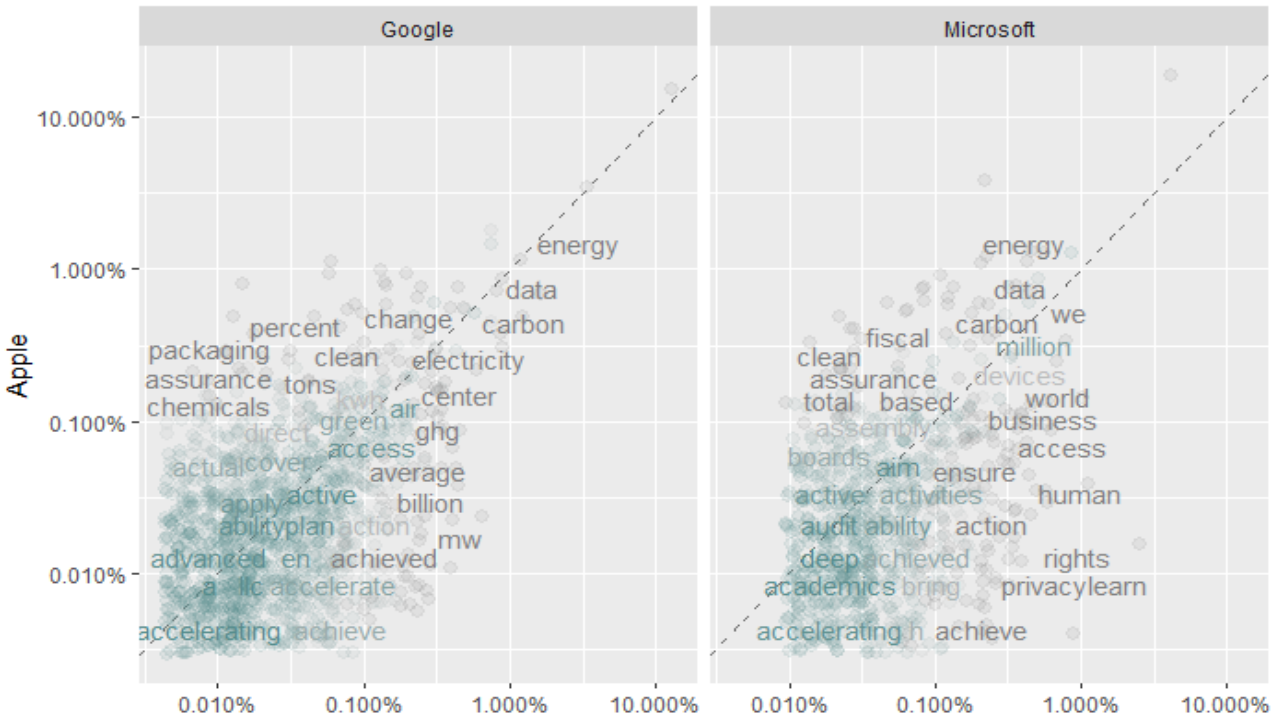
NRC Library	**Apple**	**Google**	**Microsoft**
Anger	86	65	54
Anticipation	321	367	245
Disgust	129	76	24
Fear	256	103	94
Joy	411	263	226
Negative	295	182	113
Positive	**1730**	**1197**	**945**
Sadness	87	46	43
Surprise	64	76	67
Trust	**960**	**707**	**481**

Historically CSR reports, have communicated positivity and optimism since they are focused on sustainability practices. However, it is interesting to see that the second most popular sentiment is trust. All three of these companies are leaders in the technology industry, so it seems like they are trying to convince their audience they are authentic and can be reliable. In 2019, the California Data Privacy Law act was passed, which enhanced the privacy rights and consumer protection laws for California residents - this could relate to past data breaches from Google in 2019, and when Apple had to release a statement informing customers of malpractices for battery performance issues in 2018. These companies are using their CSR report to regain trust and stabilize any doubt investors and customers may have felt.

Combining this sentiment analysis with the tokenization, Apple and Google are ensuring they are building a strong foundation around data and trust policies, whereas Microsoft is aiming to build trust around the ethical use of AI and technology opportunities.

Correlogram and Correlation

Below we can see a correlogram that compares Apple against Google and Microsoft. This information will help us understand if there are any trends in the technology industry.



Correlation	Google	Microsoft
Apple	**0.97**	0.66

Microsoft has already made it clear the focus of their report is technology, so it understandable there isn't a strong relationship with Apple. However, looking at this correlogram and the correlation coefficients it is clear that Apple and Google have extremely high correlations 0.97, which means that they are nearly perfectly correlated and have very consistent use of similar language. Since Apple and Google have a high correlation, this provides us with business insights that highlight industry trends. These results mean other high performing tech companies such as Facebook or Twitter could have similar language in their CSR reports, with their focus to also warrant language focused around data privacy and trust amongst their stakeholders.

Word Cloud, NRC Library

Data privacy was a big topic in 2019, and it is important to see if this correlation remains consistent in other leading tech companies. The main company which came under excessive scrutiny for their data privacy laws in 2019 was Facebook. Facebook does not have a CSR report, so I extracted text from its sustainability page on their website. Below is a nrc library word cloud analyzing Facebook's website.

Facebook – Word Cloud, NRC Library

Similarly, we can see Facebook's main topics are trust and positivity. Understandably, Facebook is in the process of rebuilding its public relations after its data breach affected 267 million users. They are trying to make amends, and are following Apple and Google in the route of reassuring their customers that data and trust, have a positive direction in the company. Nevertheless, it is clear Facebook is hoping to regain the trust from their users – like most tech companies data trust is again the main discussion in their public reports.

Conclusion

From this analysis, it is clear that major companies in the tech industry are looking to rebuild trust with their customers, partners, and investors. With several data management scandals arising in recent years, these giant companies are laying the foundations to make sure they are not met with challenges again by acknowledging that trust and a positive rebuild of the relationship are key to corporate social responsibility. Through text analytics, these results help apply the tokenization, correlogram, and sentiment analysis frameworks, which has given a clear picture of the topic of focus as well as the tone the companies are trying to convey.

Code

```
library(textreadr)
library(tidytext)
library(stopwords)
library(dplyr)
library(tidyverse)
library(tidytext)
library(stringr)
library(ggplot2)
library(tidyr)
library(scales)
library(readr)
library(reshape2)
library(wordcloud)
library(igraph)
library(ggraph)


#TXT FILE FORMAT AS GROUP

setwd("C:/Users/jason/Dropbox/Hult Business School/Module B/Text Mining/Business Insights Report/txt")
nm <- list.files(path="C:/Users/jason/Dropbox/Hult Business School/Module B/Text Mining/Business Insights Report/txt")

#using read document to import the data:
my_data <- read_document(file=nm[1]) #This comes out as a vector
my_data_together <- paste(my_data, collapse = " ") # This will give us a concatenated vector


#merge texts files into on and apply function to make it structured
my_txt_text <- do.call(rbind, lapply(nm, function(x) paste(read_document(file=x), collapse = " ")))


my_txt_text <- as.data.frame(my_txt_text)
colnames(my_txt_text) <- c("text")
class(my_txt_text$text)

my_txt_text <- data.frame(lapply(my_txt_text, as.character), stringsAsFactors=FALSE)


tidy_txt <- my_txt_text %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words)  %>%
  anti_join(cust_stop) %>%
  count(word, sort = T)



##############################################################################################
################################### READ FILES INDIVIDUALLY ##################################

apple <- read_document(file="C:/Users/jason/Dropbox/Hult Business School/Module B/Text Mining/Business Insights Report/txt/apple.txt")
google <- read_document(file="C:/Users/jason/Dropbox/Hult Business School/Module B/Text Mining/Business Insights Report/txt/google.txt")
microsoft <- read_document(file="C:/Users/jason/Dropbox/Hult Business School/Module B/Text Mining/Business Insights Report/txt/microsoft.txt")
```

```
twitter <- read_document(file="C:/Users/jason/Dropbox/Hult Business School/Module B/Text Mining/Business Insights Report/txt/twitter.txt")
facebook <- read_document(file="C:/Users/jason/Dropbox/Hult Business School/Module B/Text Mining/Business Insights Report/txt/facebook.txt")

########## SET UP DATAFRAME APPLE ##########
ap <- as.data.frame(apple)
colnames(ap) <- c("text") #rename column to text
ap_df <- data.frame(lapply(ap, as.character), stringsAsFactors=FALSE) #convert column type to character

########## SET UP DATAFRAME GOOGLE ##########
gg <- as.data.frame(google)
colnames(gg) <- c("text")
gg_df <- data.frame(lapply(gg, as.character), stringsAsFactors=FALSE)

########## SET UP DATAFRAME MICROSOFT ##########
mc <- as.data.frame(microsoft)
colnames(mc) <- c("text")
mc_df <- data.frame(lapply(mc, as.character), stringsAsFactors=FALSE)

########## SET UP DATAFRAME TWITTER ##########
tw <- as.data.frame(twitter)
colnames(tw) <- c("text")
tw_df <- data.frame(lapply(tw, as.character), stringsAsFactors=FALSE)

########## SET UP DATAFRAME FACEBOOK ##########
fb <- as.data.frame(facebook)
colnames(fb) <- c("text")
fb_df <- data.frame(lapply(fb, as.character), stringsAsFactors=FALSE)

#########################################################################
###################### CREATE STOP WORDS ############################
cust_stop <- data_frame(
  word = c("page", "2018", "2019", "â", "apple", "google", "microsoft", "weâ", "twitter", "facebook"), #words we want to remove
  lexicon = rep("custom", each = 10)
)

#########################################################################
######################### COUNT FREQUENCY ###############################

tidy_ap <- ap_df %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  anti_join(cust_stop) %>%
  count(word, sort = T)

tidy_gg <- gg_df %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  anti_join(cust_stop) %>%
  count(word, sort = T)

tidy_mc <- mc_df %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  anti_join(cust_stop) %>%
  count(word, sort = T)
```

```
tidy_tw <- tw_df %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  anti_join(cust_stop) %>%
  count(word, sort = T)

tidy_fb <- fb_df %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  anti_join(cust_stop) %>%
  count(word, sort = T)

#####################################################################
######################### PLOT FREQUENCY ############################
#####################################################################

#### Apple PLOT ######
tidy_ap <- ap_df %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  anti_join(cust_stop)

tidy_ap %>%
  count(word, sort=TRUE) %>%
  filter(n > 100) %>%
  mutate(word = reorder(word,n )) %>%
  ggplot(aes(word, n))+
  geom_col(fill= 'blue')+
  labs(y="Word Count", x=NULL, title = "Apple")+
  coord_flip()

##### GOOGLE PLOT #######
tidy_gg <- gg_df %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  anti_join(cust_stop)

tidy_gg %>%
  count(word, sort=TRUE) %>%
  filter(n > 60) %>%
  mutate(word = reorder(word,n )) %>%
  ggplot(aes(word, n))+
  geom_col(fill= 'red')+
  labs(y="Word Count", x=NULL, title = "Google")+
  coord_flip()

##### Microsoft PLOT #######
tidy_mc <- mc_df %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  anti_join(cust_stop)

tidy_mc %>%
  count(word, sort=TRUE) %>%
  filter(n > 30) %>%
  mutate(word = reorder(word,n )) %>%
  ggplot(aes(word, n))+
  geom_col(fill= 'lightgreen')+
  labs(y="Word Count", x=NULL, title = "Microsoft")+
```

```
  coord_flip()

##############################################################################
################## CHECK SENTIMENT VALUE FOR AFINN LIBRARY ###################
##############################################################################

afinn_ap <- ap_df %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  anti_join(cust_stop)%>%
  inner_join(get_sentiments("afinn")) %>%
  summarise(mean(value))

affin_gg <- gg_df %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  anti_join(cust_stop) %>%
  inner_join(get_sentiments("afinn")) %>%
  summarise(mean(value))

affin_mc <- mc_df %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  anti_join(cust_stop) %>%
  inner_join(get_sentiments("afinn")) %>%
  summarise(mean(value))

##############################################################################
################## CHECK SENTIMENT VALUE FOR BING LIBRARY ####################
##############################################################################

bing_ap <- ap_df %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  anti_join(cust_stop)%>%
  inner_join(get_sentiments("bing")) %>%
  count(sentiment)

bing_gg <- gg_df %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  anti_join(cust_stop) %>%
  inner_join(get_sentiments("bing")) %>%
  count(sentiment)

bing_mc <- mc_df %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  anti_join(cust_stop) %>%
  inner_join(get_sentiments("bing")) %>%
  count(sentiment)

##############################################################################
################## CHECK SENTIMENT VALUE FOR NRC LIBRARY #####################
##############################################################################

nrc_ap <- ap_df %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
```

```
  anti_join(cust_stop)%>%
  inner_join(get_sentiments("nrc")) %>%
  count(sentiment)

nrc_gg <- gg_df %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  anti_join(cust_stop) %>%
  inner_join(get_sentiments("nrc")) %>%
  count(sentiment)

nrc_mc <- mc_df %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  anti_join(cust_stop) %>%
  inner_join(get_sentiments("nrc")) %>%
  count(sentiment)

################################################################################
####################### ALL DATAFRAMES SENTIMENT GRAPHS ########################
######### CHANGE THE DATAFRAM TYPE TO GET GRAPHS OF DIFFERENT FILES #############
################################################################################

# BING
bing_text <- my_txt_text %>%
  unnest_tokens(word, text) %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort=T) %>%
  ungroup()

# BING GRAPH
bing_text %>%
  group_by(sentiment) %>%
  top_n(10) %>%
  ungroup() %>%
  mutate(word=reorder(word, n)) %>%
  ggplot(aes(word, n, fill=sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y")+
  labs(y="Contribution to sentiment", x=NULL, title = "Combined Dataframes")+
  coord_flip()

# BING WORDCLOUD
bing_text %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort=TRUE) %>%
  acast(word ~sentiment, value.var="n", fill=0) %>%
  comparison.cloud(colors = c("grey20", "gray80"),
                   max.words=100,
                   scale = c(0.5,0.5),
                   fixed.asp = TRUE,
                   title.size = 1)

# NRC WORDCLOUD
nrc_text <- my_txt_text %>%
  unnest_tokens(word, text) %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort=T) %>%
  ungroup()
```

```
nrc_text %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort=TRUE) %>%
  acast(word ~sentiment, value.var="n", fill=0) %>%
  comparison.cloud(colors = c("grey20", "gray80"),
                   max.words=100,
                   scale = c(0.5,0.5),
                   fixed.asp = TRUE,
                   title.size = 1)

# AFINN BAR CHART
afinn_text <- my_txt_text %>%
  unnest_tokens(word, text) %>%
  inner_join(get_sentiments("afinn")) %>%
  count(word, value, sort=T) %>%
  ungroup()

afinn_text %>%
  group_by(value) %>%
  top_n(10) %>%
  ungroup() %>%
  mutate(word=reorder(word, n)) %>%
  ggplot(aes(word, n, fill=value)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~value, scales = "free_y")+
  labs(y="Contribution to sentiment", x=NULL)+
  coord_flip()

################################################################################
#################### APPLE GRAPHS - SENTIMENT ##################################
################################################################################

# BING
bing_ap <-ap_df %>%
  unnest_tokens(word, text) %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort=T) %>%
  ungroup()

# BING GRAPH
bing_ap %>%
  group_by(sentiment) %>%
  top_n(10) %>%
  ungroup() %>%
  mutate(word=reorder(word, n)) %>%
  ggplot(aes(word, n, fill=sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y")+
  labs(y="Contribution to sentiment", x=NULL, title = "Apple - Bing Sentiments")+
  coord_flip()

# BING WORDCLOUD
bing_ap %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort=TRUE) %>%
  acast(word ~sentiment, value.var="n", fill=0) %>%
  comparison.cloud(colors = c("grey20", "gray80"),
                   max.words=100,
```

```
                scale = c(0.5,0.5),
                fixed.asp = TRUE,
                title.size = 1)

# NRC WORDCLOUD
nrc_ap <-ap_df %>%
  unnest_tokens(word, text) %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort=T) %>%
  ungroup()

nrc_ap %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort=TRUE) %>%
  acast(word ~sentiment, value.var="n", fill=0) %>%
  comparison.cloud(colors = c("grey20", "gray80"),
                max.words=100,
                scale = c(0.5,0.5),
                fixed.asp = TRUE,
                title.size = 1)

# AFINN BAR CHART
afinn_ap <-ap_df %>%
  unnest_tokens(word, text) %>%
  inner_join(get_sentiments("afinn")) %>%
  count(word, value, sort=T) %>%
  ungroup()

afinn_ap %>%
  group_by(value) %>%
  top_n(10) %>%
  ungroup() %>%
  mutate(word=reorder(word, n)) %>%
  ggplot(aes(word, n, fill=value)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~value, scales = "free_y")+
  labs(y="Contribution to sentiment", x=NULL, title = "Apple - NRC Sentiment")+
  coord_flip()

afinn_ap %>%
  inner_join(get_sentiments("afinn")) %>%
  count(word, value, sort=TRUE) %>%
  acast(word ~value, value.var="n", fill=0) %>%
  comparison.cloud(colors = c("grey20", "gray80"),
                max.words=100,
                scale = c(0.5,0.5),
                fixed.asp = TRUE,
                title.size = 1)

################################################################################
################### GOOGLE GRAPHS - SENTIMENT ##############################
################################################################################

# BING
bing_gg <-gg_df %>%
  unnest_tokens(word, text) %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort=T) %>%
  ungroup()
```

```
# BING GRAPH
bing_gg %>%
  group_by(sentiment) %>%
  top_n(10) %>%
  ungroup() %>%
  mutate(word=reorder(word, n)) %>%
  ggplot(aes(word, n, fill=sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y")+
  labs(y="Contribution to sentiment", x=NULL, title = "Google - Bing Sentiments")+
  coord_flip()

# BING WORDCLOUD
bing_gg %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort=TRUE) %>%
  acast(word ~sentiment, value.var="n", fill=0) %>%
  comparison.cloud(colors = c("grey20", "gray80"),
                   max.words=100,
                   scale = c(0.5,0.5),
                   fixed.asp = TRUE,
                   title.size = 1)

# NRC WORDCLOUD
nrc_gg <-gg_df %>%
  unnest_tokens(word, text) %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort=T) %>%
  ungroup()

nrc_gg %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort=TRUE) %>%
  acast(word ~sentiment, value.var="n", fill=0) %>%
  comparison.cloud(colors = c("grey20", "gray80"),
                   max.words=100,
                   scale = c(0.5,0.5),
                   fixed.asp = TRUE,
                   title.size = 1)

# AFINN BAR CHART
afinn_gg <-gg_df %>%
  unnest_tokens(word, text) %>%
  inner_join(get_sentiments("afinn")) %>%
  count(word, value, sort=T) %>%
  ungroup()

afinn_gg %>%
  group_by(value) %>%
  top_n(10) %>%
  ungroup() %>%
  mutate(word=reorder(word, n)) %>%
  ggplot(aes(word, n, fill=value)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~value, scales = "free_y")+
  labs(y="Contribution to sentiment", x=NULL, title = "Google - NRC Sentiment")+
  coord_flip()
```

```
afinn_gg %>%
  inner_join(get_sentiments("afinn")) %>%
  count(word, value, sort=TRUE) %>%
  acast(word ~value, value.var="n", fill=0) %>%
  comparison.cloud(colors = c("grey20", "gray80"),
                   max.words=100,
                   scale = c(0.5,0.5),
                   fixed.asp = TRUE,
                   title.size = 1)


################################################################################
################## MICROSOFT GRAPHS - SENTIMENT ##############################
################################################################################

# BING
bing_mc <-mc_df %>%
  unnest_tokens(word, text) %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort=T) %>%
  ungroup()

# BING GRAPH
bing_mc %>%
  group_by(sentiment) %>%
  top_n(10) %>%
  ungroup() %>%
  mutate(word=reorder(word, n)) %>%
  ggplot(aes(word, n, fill=sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y")+
  labs(y="Contribution to sentiment", x=NULL, title = "Microsoft - Bing Sentiments")+
  coord_flip()

# BING WORDCLOUD
bing_mc %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort=TRUE) %>%
  acast(word ~sentiment, value.var="n", fill=0) %>%
  comparison.cloud(colors = c("grey20", "gray80"),
                   max.words=100,
                   scale = c(0.5,0.5),
                   fixed.asp = TRUE,
                   title.size = 1)

# NRC WORDCLOUD
nrc_mc <-mc_df %>%
  unnest_tokens(word, text) %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort=T) %>%
  ungroup()

nrc_mc %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort=TRUE) %>%
  acast(word ~sentiment, value.var="n", fill=0) %>%
  comparison.cloud(colors = c("grey20", "gray80"),
                   max.words=100,
                   scale = c(0.5,0.5),
```

```
                fixed.asp = TRUE,
                title.size = 1)

# AFINN BAR CHART
afinn_mc <- mc_df %>%
  unnest_tokens(word, text) %>%
  inner_join(get_sentiments("afinn")) %>%
  count(word, value, sort=T) %>%
  ungroup()

afinn_mc %>%
  group_by(value) %>%
  top_n(10) %>%
  ungroup() %>%
  mutate(word=reorder(word, n)) %>%
  ggplot(aes(word, n, fill=value)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~value, scales = "free_y")+
  labs(y="Contribution to sentiment", x=NULL, title = "Google - NRC Sentiment")+
  coord_flip()

afinn_mc %>%
  inner_join(get_sentiments("afinn")) %>%
  count(word, value, sort=TRUE) %>%
  acast(word ~value, value.var="n", fill=0) %>%
  comparison.cloud(colors = c("grey20", "gray80"),
                max.words=100,
                scale = c(0.5,0.5),
                fixed.asp = TRUE,
                title.size = 1)

################################################################################
################## FACEBOOK GRAPHS - SENTIMENT ##############################
################################################################################

# BING
bing_fb <-fb_df %>%
  unnest_tokens(word, text) %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort=T) %>%
  ungroup()

# BING GRAPH
bing_fb %>%
  group_by(sentiment) %>%
  top_n(10) %>%
  ungroup() %>%
  mutate(word=reorder(word, n)) %>%
  ggplot(aes(word, n, fill=sentiment)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~sentiment, scales = "free_y")+
  labs(y="Contribution to sentiment", x=NULL, title = "Microsoft - Bing Sentiments")+
  coord_flip()

# BING WORDCLOUD
bing_fb %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort=TRUE) %>%
  acast(word ~sentiment, value.var="n", fill=0) %>%
```

```
  comparison.cloud(colors = c("grey20", "gray80"),
                   max.words=100,
                   scale = c(0.5,0.5),
                   fixed.asp = TRUE,
                   title.size = 1)

# NRC WORDCLOUD
nrc_fb <-fb_df %>%
  unnest_tokens(word, text) %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort=T) %>%
  ungroup()

nrc_fb %>%
  inner_join(get_sentiments("nrc")) %>%
  count(word, sentiment, sort=TRUE) %>%
  acast(word ~sentiment, value.var="n", fill=0) %>%
  comparison.cloud(colors = c("grey20", "gray80"),
                   max.words=100,
                   scale = c(0.5,0.5),
                   fixed.asp = TRUE,
                   title.size = 1)

# AFINN BAR CHART
afinn_fb <- fb_df %>%
  unnest_tokens(word, text) %>%
  inner_join(get_sentiments("afinn")) %>%
  count(word, value, sort=T) %>%
  ungroup()

afinn_fb %>%
  group_by(value) %>%
  top_n(10) %>%
  ungroup() %>%
  mutate(word=reorder(word, n)) %>%
  ggplot(aes(word, n, fill=value)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~value, scales = "free_y")+
  labs(y="Contribution to sentiment", x=NULL, title = "Google - NRC Sentiment")+
  coord_flip()

afinn_fb %>%
  inner_join(get_sentiments("afinn")) %>%
  count(word, value, sort=TRUE) %>%
  acast(word ~value, value.var="n", fill=0) %>%
  comparison.cloud(colors = c("grey20", "gray80"),
                   max.words=100,
                   scale = c(0.5,0.5),
                   fixed.asp = TRUE,
                   title.size = 1)


################################################################################
##################### APPLE vs Google and Microsoft ############################
################################################################################


frequency <- bind_rows(mutate(tidy_ap, author="Apple"),
                       mutate(tidy_gg, author= "Google"),
```

```
                mutate(tidy_mc, author="Microsoft")
) %>%
  mutate(word=str_extract(word, "[a-z']+")) %>%
  count(author, word) %>%
  group_by(author) %>%
  mutate(proportion = n/sum(n))%>%
  select(-n) %>%
  spread(author, proportion) %>%
  gather(author, proportion, `Google`, `Microsoft`)


#############################################################################
########################## PLOT CORRELOGRAM ##################################

ggplot(frequency, aes(x=proportion, y=`Apple`,
                color = abs(`Apple`- proportion)))+
  geom_abline(color="grey40", lty=2)+
  geom_jitter(alpha=.1, size=2.5, width=0.3, height=0.3)+
  geom_text(aes(label=word), check_overlap = TRUE, vjust=1.5) +
  scale_x_log10(labels = percent_format())+
  scale_y_log10(labels= percent_format())+
  scale_color_gradient(limits = c(0,0.001), low = "darkslategray4", high = "gray75")+
  facet_wrap(~author, ncol=2)+
  theme(legend.position = "none")+
  labs(y= "Apple", x=NULL)


cor.test(data=frequency[frequency$author == "Google",],
       ~proportion + `Apple`)

cor.test(data=frequency[frequency$author == "Microsoft",],
       ~proportion + `Apple`)

#############################################################################
###################### Apple vs Facebook and Twitter ########################
#############################################################################

frequency_2 <- bind_rows(mutate(tidy_ap, author="Apple"),
                mutate(tidy_fb, author= "Facebook"),
                mutate(tidy_tw, author="Twitter")
) %>%
  mutate(word=str_extract(word, "[a-z']+")) %>%
  count(author, word) %>%
  group_by(author) %>%
  mutate(proportion = n/sum(n))%>%
  select(-n) %>%
  spread(author, proportion) %>%
  gather(author, proportion, `Facebook`, `Twitter`)


#############################################################################
########################## PLOT CORRELOGRAM AGAINST TWITTER ##################

ggplot(frequency_2, aes(x=proportion, y=`Apple`,
                color = abs(`Apple`- proportion)))+
  geom_abline(color="grey40", lty=2)+
  geom_jitter(alpha=.1, size=2.5, width=0.3, height=0.3)+
  geom_text(aes(label=word), check_overlap = TRUE, vjust=1.5) +
  scale_x_log10(labels = percent_format())+
```

```
  scale_y_log10(labels= percent_format())+
  scale_color_gradient(limits = c(0,0.001), low = "darkslategray4", high = "gray75")+
  facet_wrap(~author, ncol=2)+
  theme(legend.position = "none")+
  labs(y= "Apple", x=NULL)


cor.test(data=frequency[frequency$author == "Facebook",],
        ~proportion + `Apple`)

cor.test(data=frequency[frequency$author == "Twitter",],
        ~proportion + `Apple`)

##############################################################################
#################### COMBINE ALL TIDY QUESTIONS INTO ONE ######################
##############################################################################

combined_reports <- bind_rows(
  mutate(tidy_ap, location = "one"),
  mutate(tidy_gg, location = "two"),
  mutate(tidy_mc, location = "three"),
)


######## WE ARE MORE INTERESTED IN THE WORDS WHICH ARE LESS FREQUENT ############

combined_reports <- combined_reports %>%
  bind_tf_idf(word,location, n) %>%
  arrange(desc(tf_idf))

# graph
combined_reports %>%
  arrange(desc(tf_idf)) %>%
  mutate(word=factor(word, levels=rev(unique(word)))) %>%
  group_by(location) %>%
  top_n(15) %>%
  ungroup %>%
  ggplot(aes(word, tf_idf, fill=location))+
  geom_col(show.legend=FALSE)+
  labs(x=NULL, y="tf-idf")+
  facet_wrap(~location, ncol=2, scales="free")+
  coord_flip()


#############################################################################
######################### BI GRAMS ##########################################
#############################################################################

txt_bigrams <- my_txt_text %>%
  unnest_tokens(bigram, text, token = "ngrams", n=2) %>%
  count(bigram, sort = TRUE)

bigrams_separated <- txt_bigrams %>%
  separate(bigram, c("word1", "word2"), sep = " ")

# remove stop words from each variable
bigrams_filtered <- bigrams_separated %>%
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word)
```

```
#creating the new bigram, "no-stop-words":
bigram_counts <- bigrams_filtered %>%
  count(word1, word2, sort = TRUE)

#want to see the new bigrams
bigram_counts

# united bigrams
bigram_united <- bigrams_filtered %>%
  unite(bigram, word1, word2, sep=" ")

bigram_tf_idf <- bigram_united %>%
  count(location, bigram) %>%
  bind_tf_idf(bigram, location, n) %>%
  arrange(desc(tf_idf))

negation_tokens <- c("no", "not")#what negation tokens do you want to use?

negated_words <- bigrams_separated %>%
  filter(word1 %in% negation_tokens) %>%
  inner_join(get_sentiments("afinn"), by=c(word2="word")) %>%
  count(word1, word2, value, sort=TRUE) %>%
  ungroup()

negated_words

##################################################
####### function to plot the negations ##########
##################################################
negated_words_plot <- function(x){
  negated_words %>%
    filter(word1 == x) %>%
    mutate(contribution = n* value) %>%
    arrange(desc(abs(contribution))) %>%
    head(20) %>%
    mutate(word2 = reorder(word2, contribution)) %>%
    ggplot(aes(word2, n*value, fill = n*value >0))+
    geom_col(show.legend = FALSE)+
    xlab(paste("Words preceded by", x))+
    ylab("Sentiment score* number of occurences")+
    coord_flip()
}#closing the negated_words_plot function

negated_words_plot(x="not") #this is your first negation word
negated_words_plot(x="xxxxxxx") #this is your second negation word
negated_words_plot(x="xxxxxxx") #this is your third negation word

#########################################################
####### VISUALISING A BIGRAM NETWORK ################
#########################################################
bigram_graph <- bigram_counts %>%
  filter(n>20) %>%
  graph_from_data_frame()

ggraph(bigram_graph, layout = "fr") +
  geom_edge_link()+
  geom_node_point()+
  geom_node_text(aes(label=name), vjust =1, hjust=1)
```

CODE OUTPUT

Popular Words (Top 3)

Apple	Google	Microsoft
Energy (341)	Energy (332)	Learn (94)
Renewable (197)	Data (165)	AI (63)
Data (176)	Carbon (146)	Human(52)

Sentiment Analysis

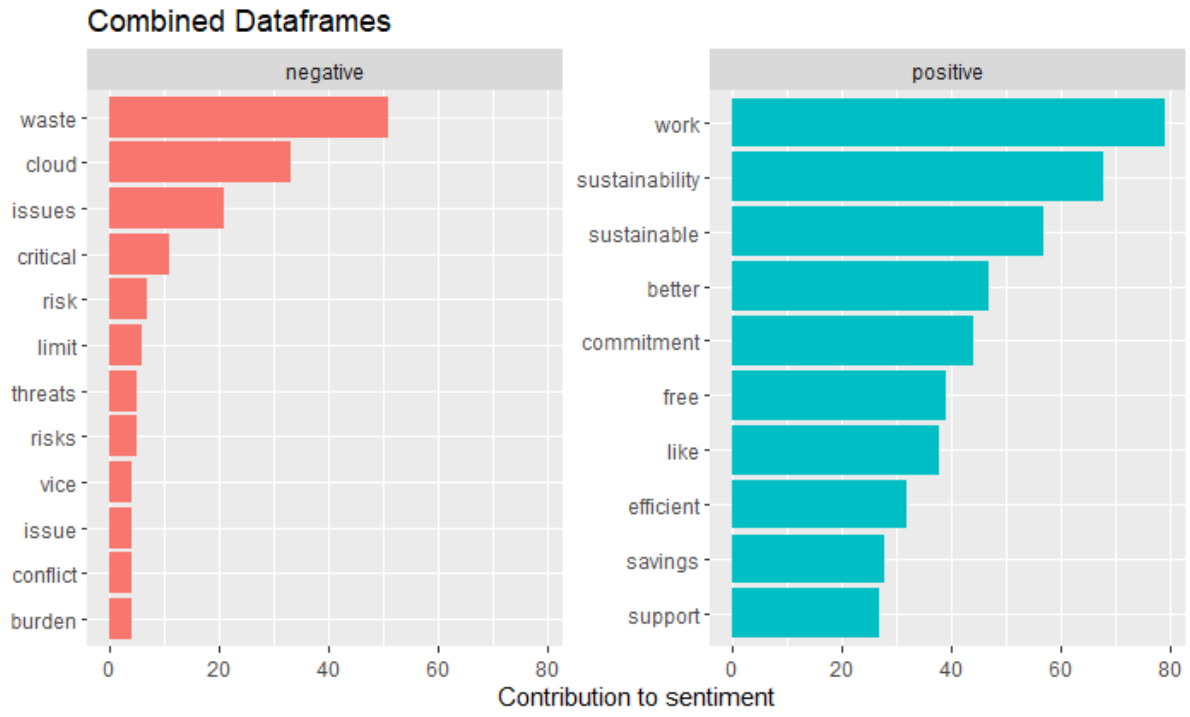
AFINN Library	Apple	Google	Microsoft
	1.09	0.89	1.20

BING Library	Apple	Google	Microsoft
Positive	828	515	440
Negative	213	130	101

NRC Library	Apple	Google	Microsoft
Anger	86	65	54
Anticipation	321	367	245
Disgust	129	76	24
Fear	256	103	94
Joy	411	263	226
Negative	295	182	113
Positive	1730	1197	945
Sadness	87	46	43
Surprise	64	76	67
Trust	960	707	481

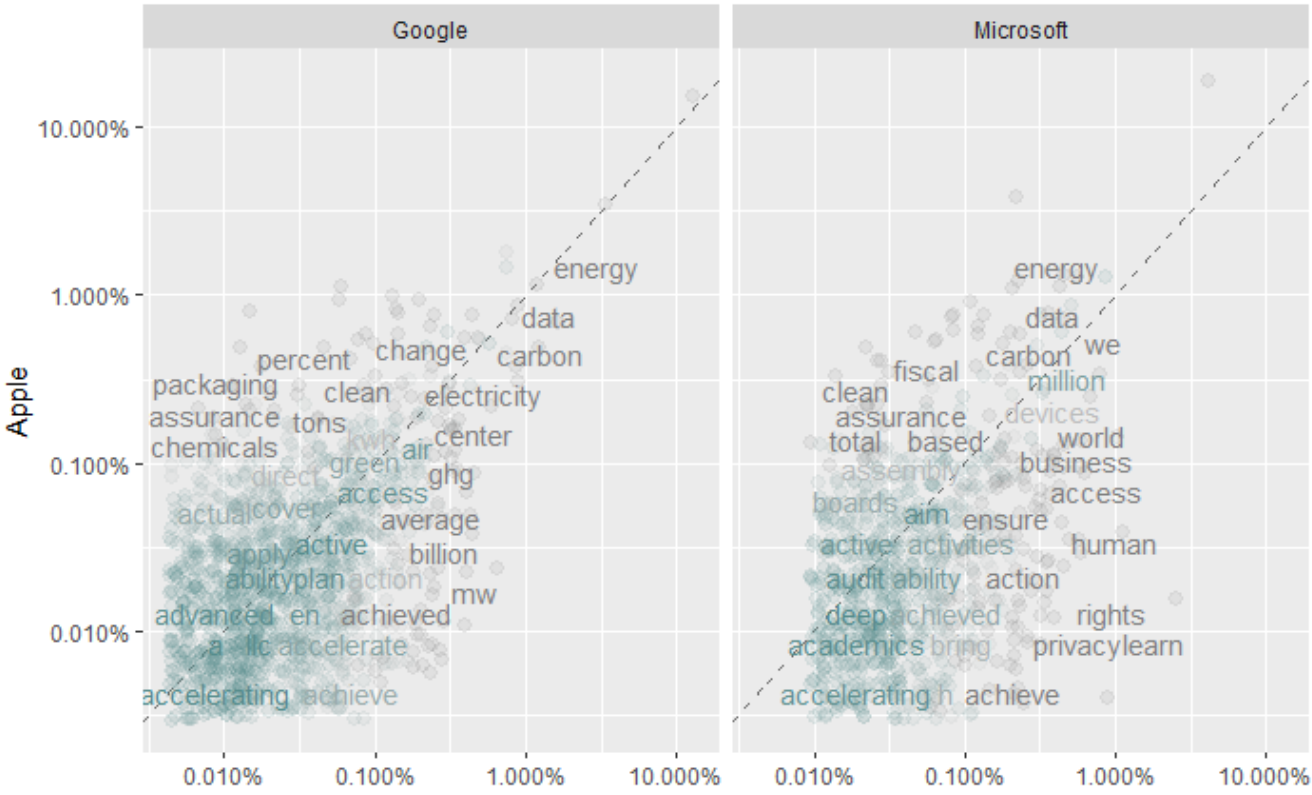
Combined Datasets (Apple, Google, Microsoft)

Bing Library



NRC Word Cloud

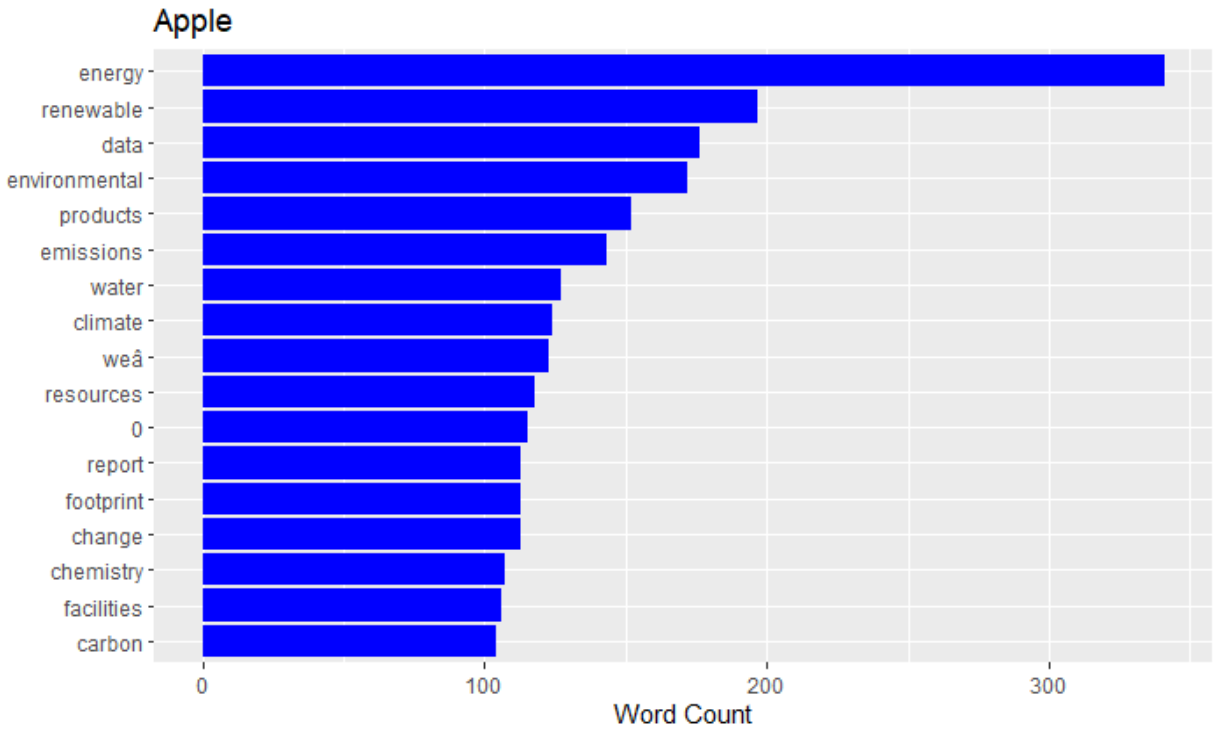
Correlogram – Comparing both to Apple



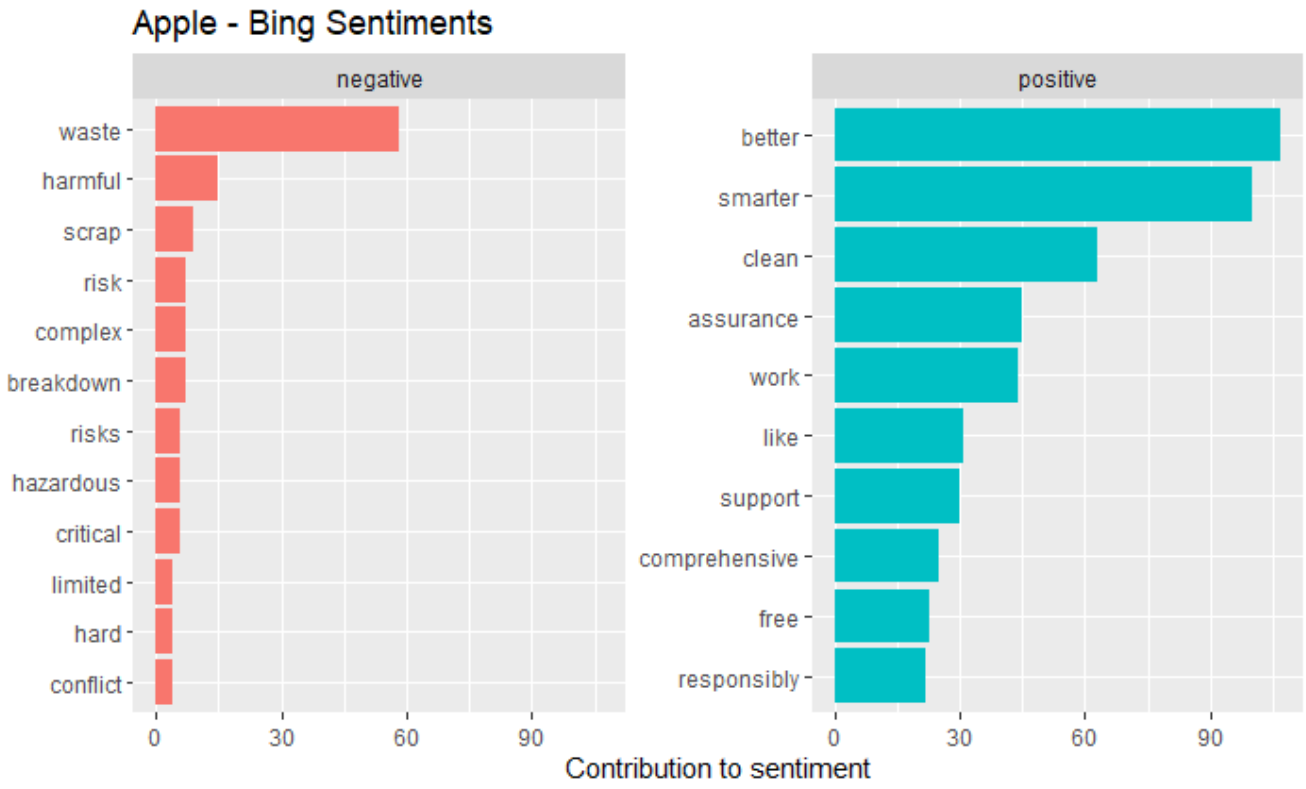
Correlation	Google	Microsoft
Apple	0.97	0.66

Apple Dataframe Only

Word Count

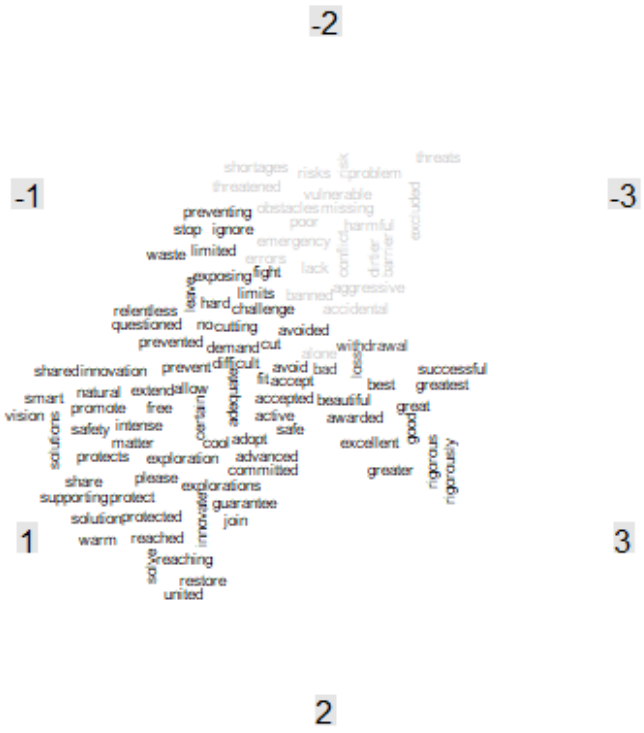


Bing Library



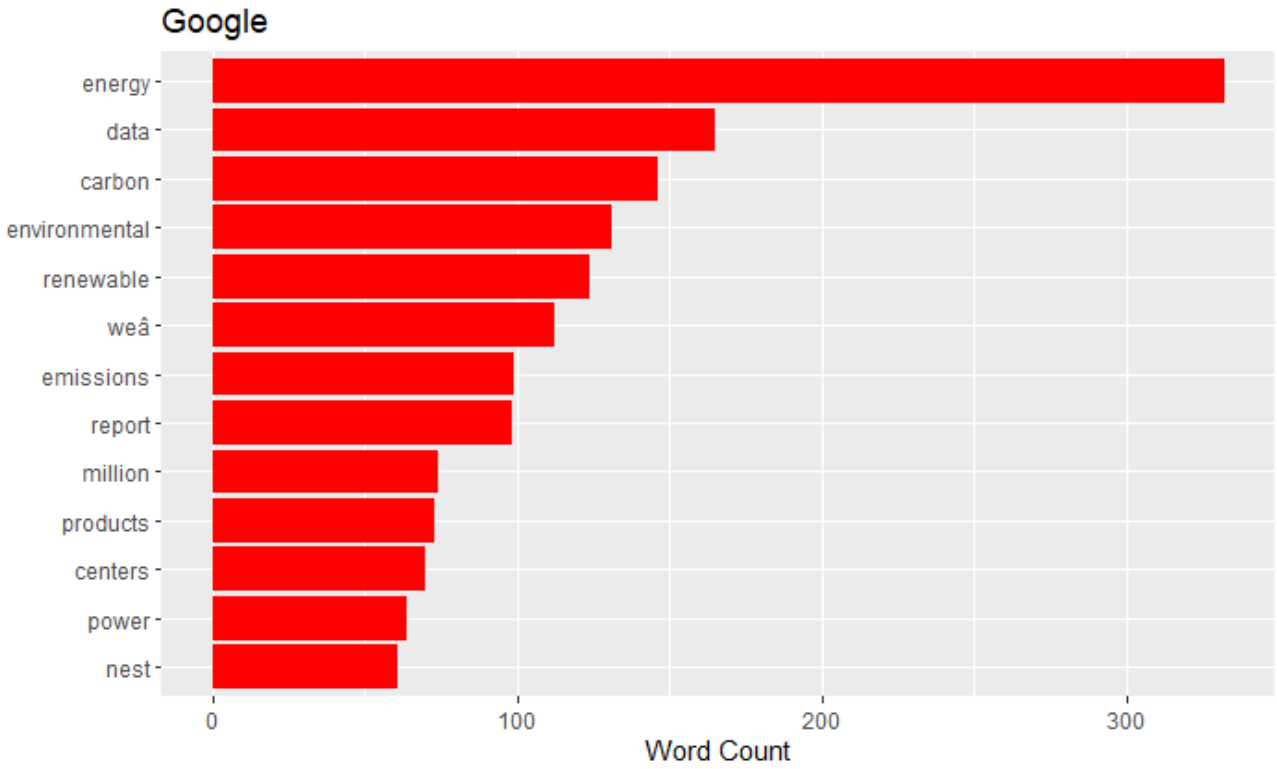
NRC Word Cloud

Afinn Word Cloud

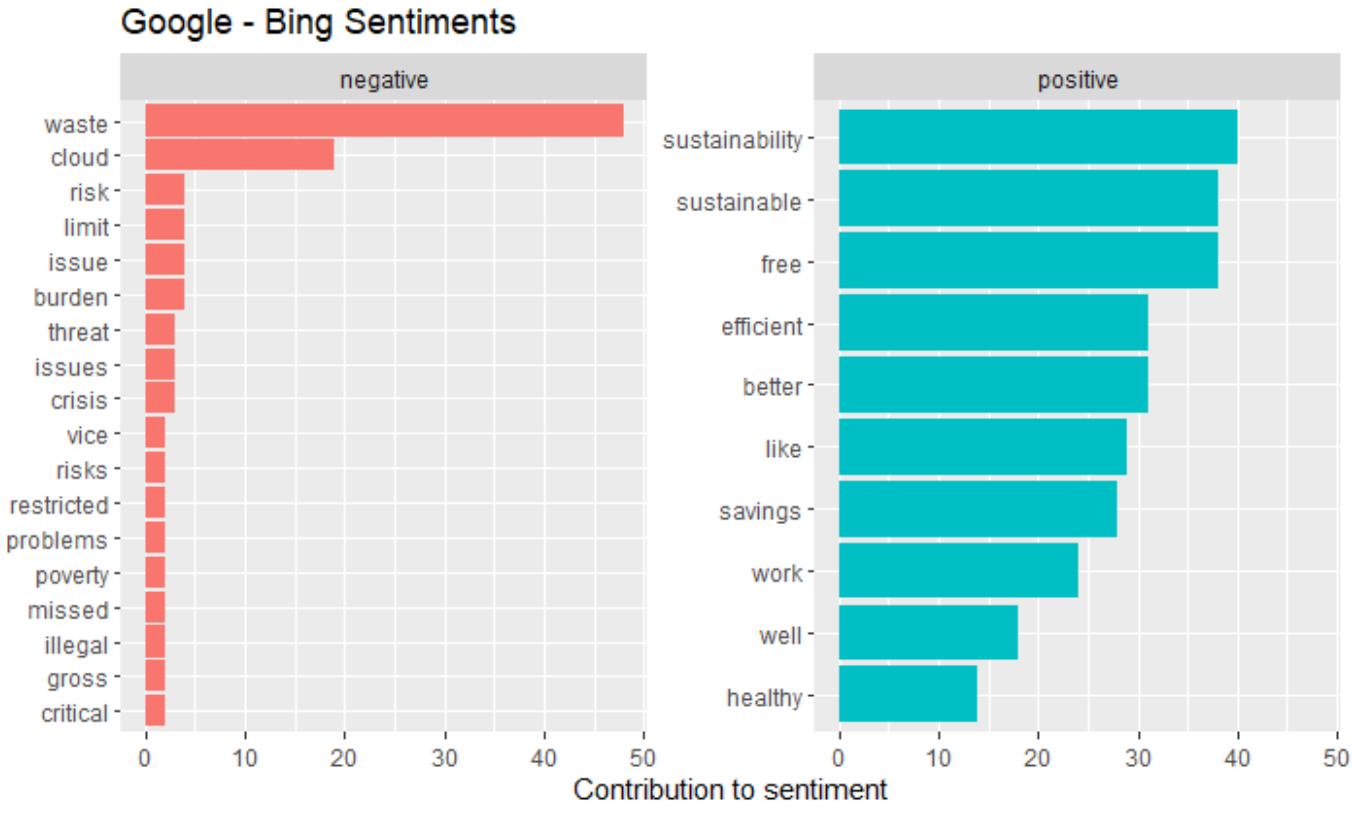


Google Dataframe Only

Frequency Word Count

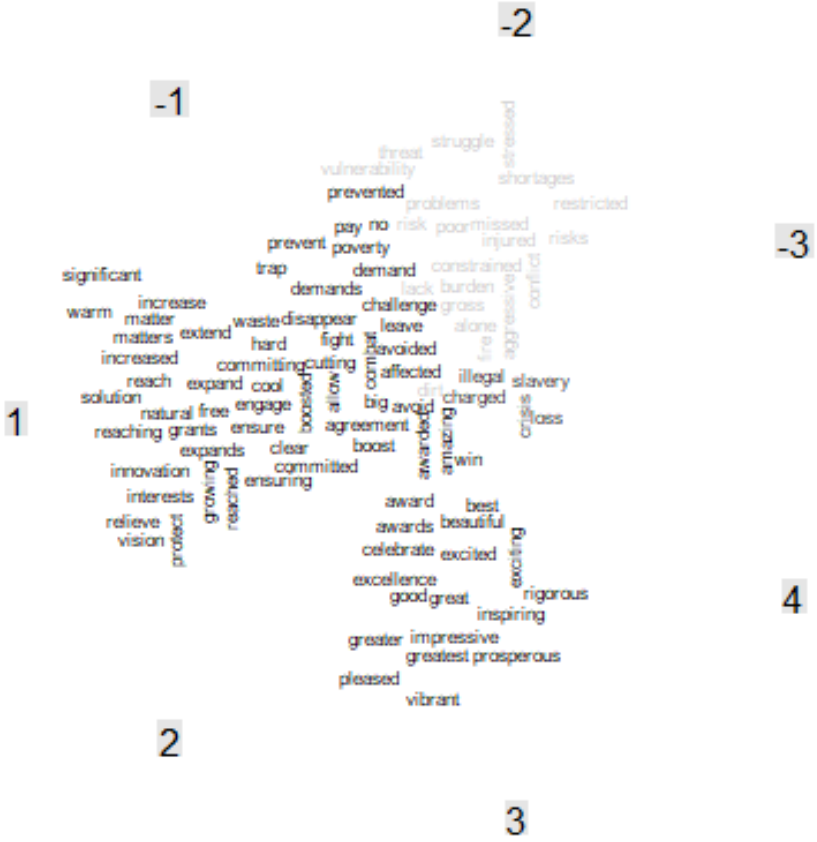


Bing Library



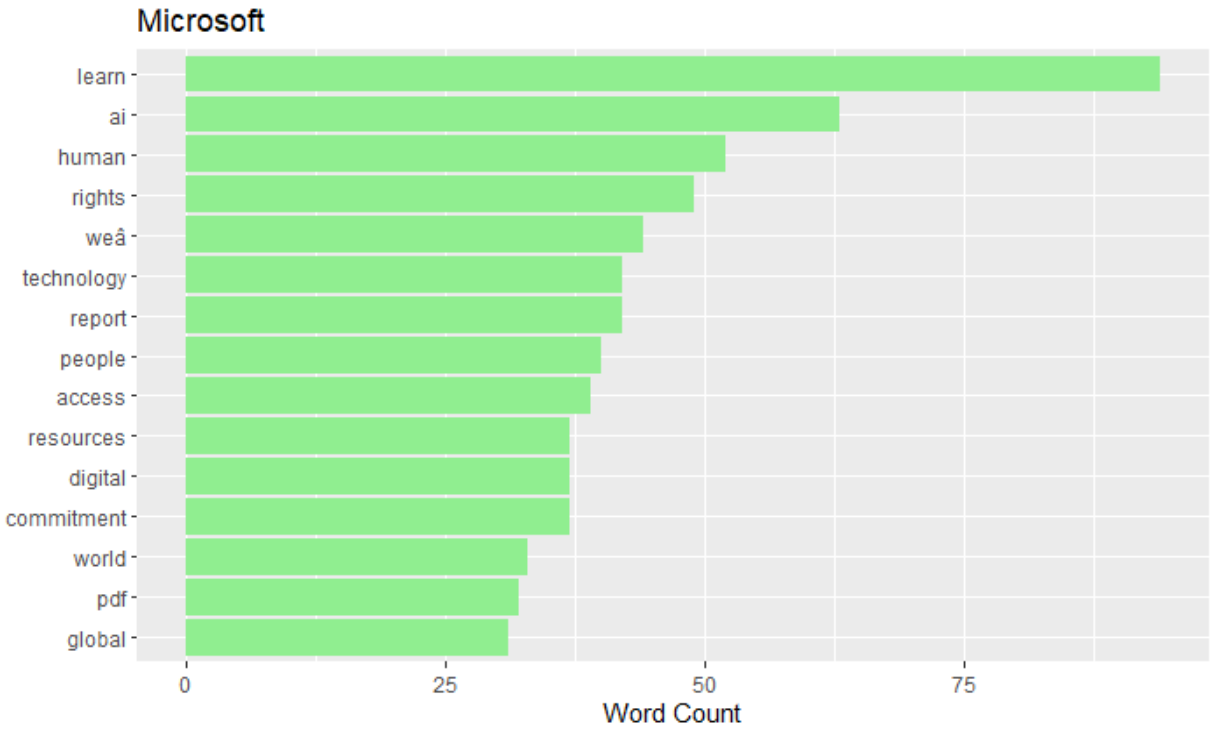
NRC Word Cloud

Afinn Word

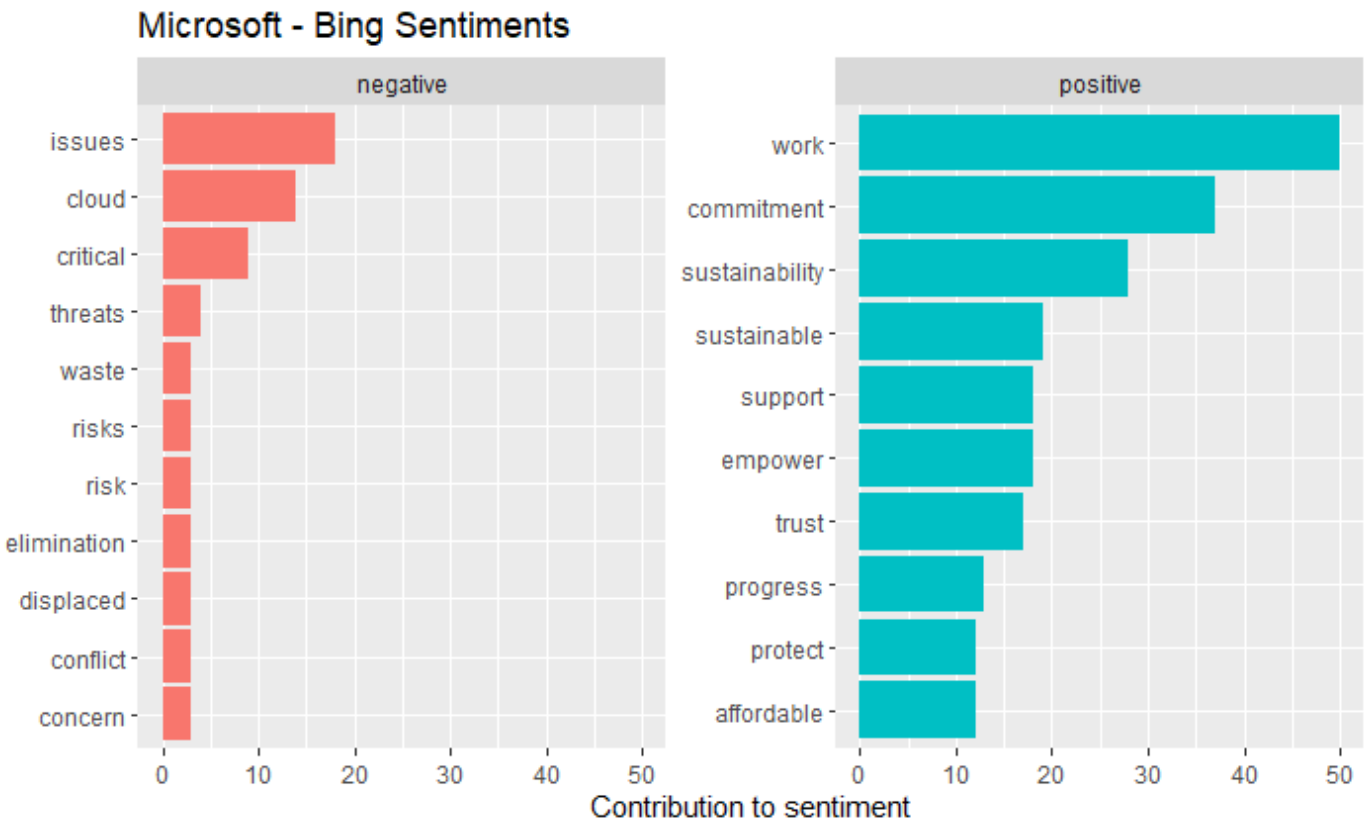


Microsoft Dataframe Only

Frequency Word Count



Bing Library



NRC Word Cloud

Afinn Word Cloud