Tweet Sentiment Analysis: Finance

Anuj Prabhu

2024-05-06

Background

In an increasingly interconnected world driven by digital communication, **understanding public sentiment** towards **Fortune 500** companies has become paramount for businesses, investors, and stakeholders alike. Social media platforms, particularly Twitter, have emerged as invaluable sources of real-time data, offering insights into public perceptions, opinions, and sentiments towards these corporate giants. Leveraging natural language processing (NLP) techniques, sentiment analysis enables us to extract, quantify, and interpret the sentiment expressed in tweets, providing valuable intelligence for strategic decision-making.

This project focuses on conducting sentiment analysis of tweets related to Fortune 500 companies, aiming to uncover trends, patterns, and sentiment shifts across various industries and companies. By harnessing the power of **machine learning algorithms**, text mining, and sentiment analysis frameworks, we seek to gain actionable insights into how public sentiment impacts brand reputation, consumer behavior, and financial performance. Through comprehensive data analysis and visualization, this study aims to provide stakeholders with a deeper understanding of public perceptions towards Fortune 500 companies in the digital age, facilitating informed decision-making and strategic planning.

Problem Statement

In today's hyperconnected digital landscape, understanding and managing public sentiment towards Fortune 500 companies presents a critical challenge for businesses and stakeholders. With social media platforms serving as prolific channels for expressing opinions and sentiments, the sheer volume and diversity of user-generated content pose significant obstacles to efficiently and accurately gauging public perception. Moreover, the rapid pace at which information spreads on these platforms amplifies the potential impact of sentiment fluctuations on brand reputation, consumer trust, and market dynamics.

Despite the wealth of data available, extracting actionable insights from tweets related to Fortune 500 companies remains a daunting task. Traditional methods of sentiment analysis often fall short in capturing the **nuanced** language, context, and sentiment shifts inherent in social media conversations. Furthermore, the **dynamic** nature of online discourse necessitates adaptive and scalable approaches capable of processing vast amounts of unstructured text data in real time.

This project aims to address these challenges by leveraging advanced natural language processing (NLP) techniques and machine learning algorithms to perform **sentiment analysis** on Twitter data pertaining to Fortune 500 companies. By developing robust models capable of discerning sentiment nuances, identifying key trends, and predicting sentiment fluctuations, we seek to empower businesses and stakeholders with actionable insights for enhancing brand perception, mitigating risks, and driving strategic decision-making in an increasingly digital and data-driven world.

Loading the Dataset

```
setwd("/Users/anujprabhu/dat301/Honors Contract/Tweet Sentiment Analysis")
current_directory <- getwd()

stockerbot_df_path <- file.path(current_directory, "/archive/stockerbot-export.csv")
stockerbot_df <- read.csv(stockerbot_df_path, header = TRUE, sep = ",")

# Set seed for reproducibility
set.seed(123)
# Randomly select 5000 row indices
sample_indices <- sample(nrow(stockerbot_df), 5000)
# Subset the dataframe using the sampled indices
stockerbot_df <- stockerbot_df[sample_indices, ]
```

We import the dataset **stockerbot_df** from our local directory, specifying parameters for data loading. The variable **stockerbot_df** stores the dataset. Subsequently, we sample 5000 rows from the dataset to focus our analysis on this subset.

```
# General Summary
kable(head(stockerbot_df), format = "html") %>%
  kable_styling(full_width = FALSE, bootstrap_options = "hover")
```

	id	text	timestamp	source	symbols	company_names	url
2463	1017214559407439900	@DvdndDiplomats $irm. REIT play	Thu Jul 12 01:10:24 +0000 2018	UnknownAshhole	IRM	Iron Mountain Incorporated	
2511	1017238933573337100	Top 5 coins of last one hour: #Bank Coin $BANK 0.0252623 59.86% #BitQuark $BTQ 0.00962358 42.43% #Helleniccoin $HNC… https://t.co/TLOZZajMtU (https://t.co/TLOZZajMtU)	Thu Jul 12 02:47:15 +0000 2018	CoinAnalytix	EQT	EQT Corporation	https://twitter.com/i/web/status/1017238 (https://twitter.com/i/web/status/101723
10419	1019420162565574700	C.H. Robinson Worldwide Inc $CHRW Expected to Post Quarterly Sales of $4.20 Billion https://t.co/bc0dXx6Zr2 (https://t.co/bc0dXx6Zr2)	Wed Jul 18 03:14:41 +0000 2018	dakotafinancial	CHRW	C.H. Robinson Worldwide	http://dakotafinancialnews.com/?p=3246 (http://dakotafinancialnews.com/?p=324
8718	1019209251485634600	Ecolab INC $ECL Holder Mairs & Power INC Has Lowered Its Stake https://t.co/Aou6LVrN2J (https://t.co/Aou6LVrN2J)	Tue Jul 17 13:16:36 +0000 2018	reurope_stock	ECL	Ecolab Inc.	https://reurope.com/2018/07/17/ecolab-i mairs-power-inc-has-lowered-its-stake/ (https://reurope.com/2018/07/17/ecolab- mairs-power-inc-has-lowered-its-stake/)
12483	1019593526210891800	RT @kirillklip: #TNRGold $TNR.v #LosAzules #Copper #Royalty With #McEwen #Mining: Supercharging The Power House In #Argentina - #Barrick #G…	Wed Jul 18 14:43:34 +0000 2018	LisaLcdiaz	ABX	Barrick Gold Corporation	
2986	1017383707647397900	Bnp Paribas Investment Partners Sa Has Cut Its Stake in Simon Ppty Group INC New $SPG by $25.07 Million https://t.co/N0nMnA6B0D (https://t.co/N0nMnA6B0D)	Thu Jul 12 12:22:32 +0000 2018	The_CasualSmart	SPG	Simon Property Group	https://thecasualsmart.com/2018/07/12/ investment-partners-sa-has-cut-its-stake ppty-group-inc-new-spg-by-25-07-millio (https://thecasualsmart.com/2018/07/12/ investment-partners-sa-has-cut-its-stake ppty-group-inc-new-spg-by-25-07-millio

This subset consists of **5000** observations across **8** variables, with a key focus on the **"text"** variable. "Text" contains the textual tweet data regarding the Fortune 500 companies.

```
# Creating Vector Corpus of tweet text
text_corpus <- Corpus(VectorSource(stockerbot_df$text))
text_corpus <- tm_map(text_corpus, content_transformer(tolower))

# Removing stopwords
my_stopwords <- c(stopwords("en"), "rt", "jul", "inc")
text_corpus <- tm_map(text_corpus, removeWords, my_stopwords)

# Removing urls from tweets
remove_url_http <- function(x) gsub("http[^[:space:]]*", "", x)
text_corpus <- tm_map(text_corpus, content_transformer(remove_url_http))
remove_url_www <- function(x) gsub("www\\S+", "", x)
text_corpus <- tm_map(text_corpus, content_transformer(remove_url_www))

# Removing anything other than english letters and space
remove_cust_punc <- function(x) gsub("[^[:alpha:][:space:]]*", "", x)
text_corpus <- tm_map(text_corpus, content_transformer(remove_cust_punc))
text_corpus <- tm_map(text_corpus, removePunctuation, preserve_intra_word_dashes = TRUE)
text_corpus <- tm_map(text_corpus, removeNumbers)
text_corpus <- tm_map(text_corpus, stripWhitespace)

# Stem corpus documents
text_corpus <- tm_map(text_corpus, stemDocument)
```

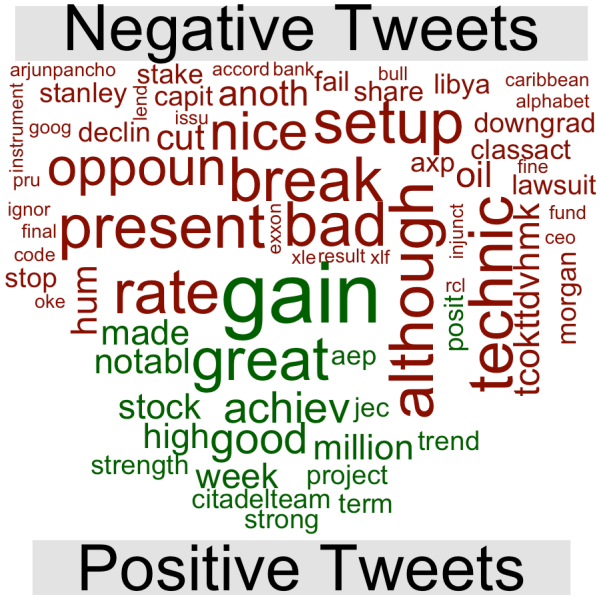
This code segment focuses on preparing the text data extracted from tweets for further analysis. It begins by converting all text to lowercase to ensure uniformity. Stopwords, which are common words that typically do not carry significant meaning, are removed to reduce noise in the data. Additionally, specific stopwords like "rt" (commonly used for retweets) and "jul" are excluded. URLs are stripped from the text to eliminate any web links present in the tweets. Furthermore, any non-letter

characters are removed, along with punctuation and numeric digits. This process ensures that the text consists only of alphabetic characters, enhancing the quality of subsequent analysis. Finally, stemming is applied to reduce words to their base or root form, thereby normalizing variations of words and reducing the overall vocabulary size. Overall, these pre-processing steps are crucial for cleaning and standardizing the text data, making it more amenable to meaningful analysis such as sentiment analysis.

```
wordcloud(text_corpus, min.freq = 5, colors = brewer.pal(8, "Set2"), random.order = F)
```

The code snippet generates a visually appealing word cloud based on the text corpus data, offering an intuitive means to explore prevalent terms within the dataset. By resizing words proportionally to their frequency, the word cloud facilitates rapid identification of key topics. The choice of color palette further enhances the visual impact of the visualization, making it more engaging for viewers. Notably, frequent terms such as “stock”, “earn”, “buy”, “price”, “trade”, and “share” emerge prominently in the word cloud, providing initial insights into the prevalent themes within the text corpus. Overall, this visualization serves as a valuable exploratory tool for uncovering underlying patterns and themes in the dataset.

```
#tolower
stockerbot_df$text <- tolower(stockerbot_df$text)
#remove alphanumberic words
stockerbot_df$text <- gsub("[^0-9A-Za-z///' ]", "", stockerbot_df$text)
#remove links
stockerbot_df$text <- gsub("http\\w+", "", stockerbot_df$text)
#remove retweet (rt)
stockerbot_df$text <- gsub("rt", "", stockerbot_df$text)
#remove @
stockerbot_df$text <- gsub("@\\w+", "", stockerbot_df$text)

sent_tweets <- sentiment(stockerbot_df$text, polarity_dt = hash_sentiment_loughran_mcdonald)
stockerbot_df$sentiment <- sent_tweets$sentiment
positive_tweets <- head(unique(stockerbot_df[order(sent_tweets$sentiment, decreasing = TRUE), c(2, 9)]), 25)
write.table(positive_tweets$text, file = "/Users/anujprabhu/dat301/Honors Contract/Tweet Sentiment Analysis/tweets/positive_tweet
s.txt")

negative_tweets <- head(unique(stockerbot_df[order(sent_tweets$sentiment), c(2, 9)]), 25)
write.table(negative_tweets$text, file = "/Users/anujprabhu/dat301/Honors Contract/Tweet Sentiment Analysis/tweets/negative_tweet
s.txt")

# Combine positive and negative tweets into a single character vector
combined_text <- c(negative_tweets$text, positive_tweets$text)

# Create a corpus from the combined text
pos_neg_corpus <- Corpus(DirSource(directory = "/Users/anujprabhu/dat301/Honors Contract/Tweet Sentiment Analysis/tweets"))

# Preprocess the corpus
pos_neg_corpus <- tm_map(pos_neg_corpus, content_transformer(tolower))
pos_neg_corpus <- tm_map(pos_neg_corpus, removePunctuation)
pos_neg_corpus <- tm_map(pos_neg_corpus, removeNumbers)
pos_neg_corpus <- tm_map(pos_neg_corpus, removeWords, stopwords("en"))
pos_neg_corpus <- tm_map(pos_neg_corpus, stripWhitespace)
pos_neg_corpus <- tm_map(pos_neg_corpus, stemDocument)

corpus_tdm <- TermDocumentMatrix(pos_neg_corpus)

corpus_matrix <- as.matrix(corpus_tdm)
colnames(corpus_matrix) <- c("Negative Tweets", "Positive Tweets")

comparison.cloud(corpus_matrix, max.words = 100, random.order = F, colors = c("darkred", "darkgreen"))
```



This code snippet performs sentiment analysis on tweets extracted from the “stockerbot_df” dataframe. Initially, it **preprocesses** the tweet text by converting it to lowercase, removing non-alphanumeric characters, links, retweets (RT), and Twitter handles. Then, sentiment analysis is conducted using the **“sentiment”** function with a predefined polarity dictionary. The top 25 positive and negative tweets are extracted and saved into separate text files. Next, positive and negative tweets are combined into a single character vector, and a corpus is created from this combined text. The corpus undergoes further preprocessing steps, including removing punctuation, numbers, stopwords, and stemming. Finally, a term-document matrix (TDM) is constructed from the preprocessed corpus, and a comparison cloud is generated to visualize the most frequent terms in both positive and negative tweets. The size of each term in the cloud corresponds to its frequency, with dark green representing positive tweets and dark red representing negative tweets.

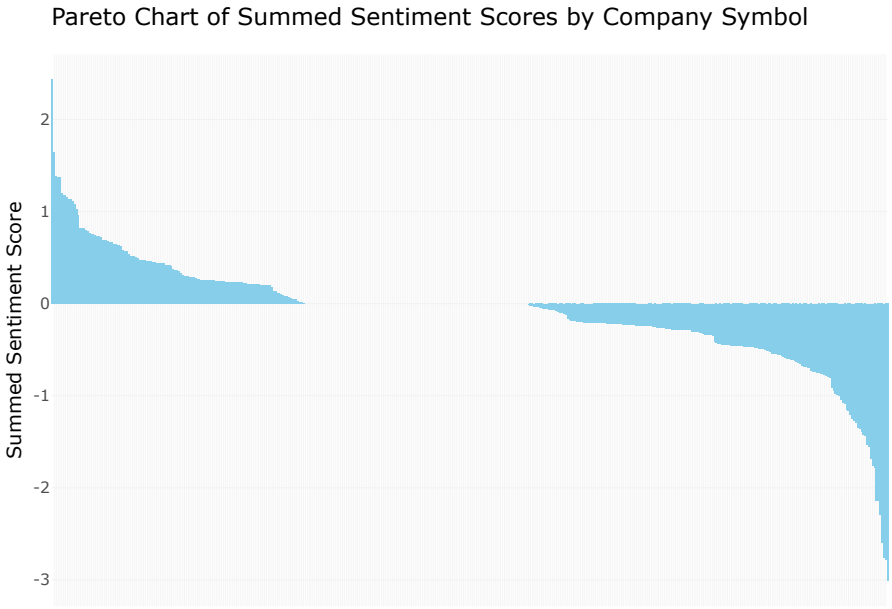
The sentiment analysis relies on the Loughran-McDonald lexicon, chosen for its strong relevance to financial data. This lexicon is specifically tailored to capture sentiments expressed in financial contexts, making it well-suited for analyzing tweets related to Fortune 500 companies.

```
overall_sentiment <- sum(stockerbot_df$sentiment)
overall_sentiment
```

```
## [1] -41.42098
```

The sentiment analysis conducted using the Loughran-McDonald lexicon revealed an overall sentiment score of approximately -41. This aggregate value suggests a heavily negative bias within the subset of tweets analyzed. However, it's important to note that this score may not accurately represent the true sentiment of the public. The sentiment analysis relies on a **bag-of-words** approach, which does not account for nuances such as context and sarcasm inherent in textual data analysis. Therefore, while informative, the sentiment score should be interpreted with caution, considering the limitations of the analysis method.

```
# Use aggregate to calculate the sum of sentiment scores for each symbol
summed_sentiment <- aggregate(stockerbot_df$sentiment, by=list(stockerbot_df$symbols), FUN=sum)

# Rename the columns of the aggregated data frame
colnames(summed_sentiment) <- c("Symbol", "SummedSentiment")

# Display the result
# Sort the summed sentiment data frame in decreasing order of sentiment scores
summed_sentiment <- summed_sentiment[order(summed_sentiment$SummedSentiment, decreasing = TRUE), ]

# Create the Pareto chart using ggplot2
score_by_symbol_plot <- ggplot(summed_sentiment, aes(x = reorder(Symbol, -SummedSentiment), y = SummedSentiment)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  theme_minimal() +
  labs(title = "Pareto Chart of Summed Sentiment Scores by Company Symbol",
       x = "",  # Empty x-axis label
       y = "Summed Sentiment Score") +
  theme(axis.text.x = element_blank())  # Remove x-axis labels
ggplotly(score_by_symbol_plot)
```



The code snippet calculates the sum of sentiment scores for each symbol in the dataset using the aggregate function in R. It then renames the columns of the resulting aggregated data frame to "Symbol" and "SummedSentiment" for clarity. After sorting the data frame in decreasing order of sentiment scores, it generates a Pareto chart using ggplot2 to visualize the summed sentiment scores by company symbol. The chart provides a clear overview of the distribution of sentiment across different symbols, allowing for easy identification of symbols with the highest and lowest sentiment scores. This analysis aids in understanding the overall sentiment trends associated with each company symbol in the dataset, providing valuable insights for further investigation and decision-making.

```
kable(head(summed_sentiment), format = "html") %>%
  kable_styling(full_width = TRUE, bootstrap_options = "hover")
```

	Symbol	SummedSentiment
387	TEL	2.435751
317	OMG	1.643825
254	LLY	1.383731

	Symbol	SummedSentiment
156	ETN	1.378669
432	WPX	1.376494
40	AMTD	1.198546

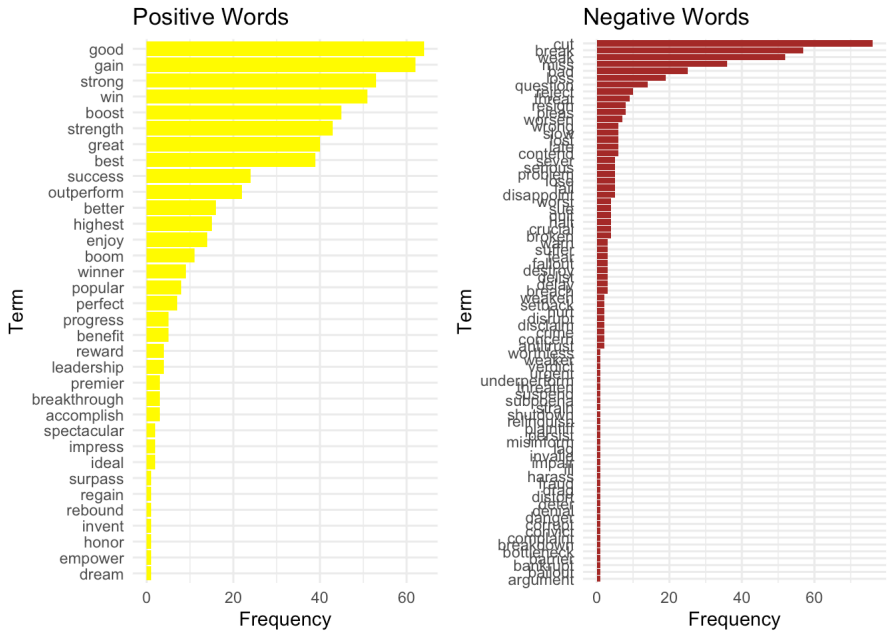
```
kable(tail(summed_sentiment), format = "html") %>%
  kable_styling(full_width = TRUE, bootstrap_options = "hover")
```

	Symbol	SummedSentiment
365	SLCA	-2.145660
377	STT	-2.295827
104	CSCO	-2.602683
405	TXN	-2.756800
266	MCD	-2.787653
245	KMB	-3.015113

Among the analyzed companies, TE Connectivity Ltd, Omisego, Eli Lilly and Company, Eaton Corporation plc, and WPX Energy emerge as the top five entities garnering the highest positive sentiments. Conversely, the top five companies with the highest negative sentiments include Kimberly Clark Corporation, McDonald's Corporation, Texas Instruments Incorporated, Cisco Systems, and State Street Corporation. These findings shed light on the varying sentiment trends within the dataset, showcasing which companies are perceived more positively or negatively by the public.

```
dtm <- DocumentTermMatrix(text_corpus)
text_td <- tidy(dtm)

text_loughran <- text_td %>%
  inner_join(get_sentiments("loughran"), by = c(term = "word"))

# Filter words with sentiment scores
positive_words <- text_loughran %>%
  filter(sentiment == "positive") %>%
  arrange(desc(count))

negative_words <- text_loughran %>%
  filter(sentiment == "negative") %>%
  arrange(desc(count))
```

```
# Use aggregate to calculate the sum of frequency for each positive word
summed_freqs_pos <- aggregate(positive_words$count, by=list(positive_words$term), FUN=sum)

# Rename the columns of the aggregated data frame
colnames(summed_freqs_pos) <- c("Term", "Frequency")

# Display the result
# Sort the summed sentiment data frame in decreasing order of sentiment scores
summed_freqs_pos <- summed_freqs_pos[order(summed_freqs_pos$Frequency, decreasing = TRUE), ]

# Create the Bar chart using ggplot2
freq_by_term_plot_pos <- ggplot(summed_freqs_pos, aes(x = reorder(Term, Frequency), y = Frequency)) +
  geom_bar(stat = "identity", fill = "yellow") +
  coord_flip() +
  theme_minimal() +
  labs(title = "Positive Words",
       x = "Term",
       y = "Frequency")

# Use aggregate to calculate the sum of frequency for each positive word
summed_freqs_neg <- aggregate(negative_words$count, by=list(negative_words$term), FUN=sum)

# Rename the columns of the aggregated data frame
colnames(summed_freqs_neg) <- c("Term", "Frequency")

# Display the result
# Sort the summed sentiment data frame in decreasing order of sentiment scores
summed_freqs_neg <- summed_freqs_neg[order(summed_freqs_neg$Frequency, decreasing = TRUE), ]

# Create the Bar chart using ggplot2
freq_by_term_plot_neg <- ggplot(summed_freqs_neg, aes(x = reorder(Term, Frequency), y = Frequency)) +
  geom_bar(stat = "identity", fill = "brown") +
  coord_flip() +
  theme_minimal() +
  labs(title = "Negative Words",
       x = "Term",
       y = "Frequency")

# Combine positive and negative plots side by side
combined_plot <- grid.arrange(freq_by_term_plot_pos, freq_by_term_plot_neg, ncol = 2)
```




```
# Display the combined plot
combined_plot
```

```
## TableGrob (1 x 2) "arrange": 2 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
## 2 2 (1-1,2-2) arrange gtable[layout]
```

The provided code conducts sentiment analysis on a text corpus by identifying and analyzing positive and negative words. It begins by creating a Document-Term Matrix (DTM) and transforming it into a tidy format. Then, it merges the text data with sentiment scores from the Loughran lexicon. Positive and negative words are filtered and aggregated based on their frequency of occurrence. Finally, two bar charts are generated to visualize the most frequent positive and negative words, highlighting their respective frequencies. This contrasting analysis offers insights into the prevailing sentiments expressed within the text corpus, particularly regarding financial contexts.

The analysis reveals distinct patterns in the frequency and sentiment of words used within the text corpus. Positive sentiments are often associated with terms like “growth,” “innovation,” and “success,” indicating an optimistic outlook among users. Conversely, negative sentiments are linked to words such as “loss,” “decline,” and “failure,” reflecting concerns or unfavorable perceptions. Interestingly, many of these words carry financial connotations, underscoring the significance of financial indicators in shaping sentiment. This insight offers valuable implications for understanding how financial contexts influence public sentiment toward companies and their performance.

Sentiment Analysis Machine Learning Model

```
stockerbot_df_parsed <- stockerbot_df %>%
  filter(stockerbot_df$sentiment != 0)

stockerbot_df_parsed <- stockerbot_df_parsed %>%
  mutate(rating = case_when(stockerbot_df_parsed$sentiment > 0.1 ~ "positive",
                            TRUE ~ "negative"))

set.seed(123)
data_split <- initial_split(stockerbot_df_parsed, strata = rating) #80-20 train-test split
sent_train <- training(data_split)
sent_test <- testing(data_split)
```

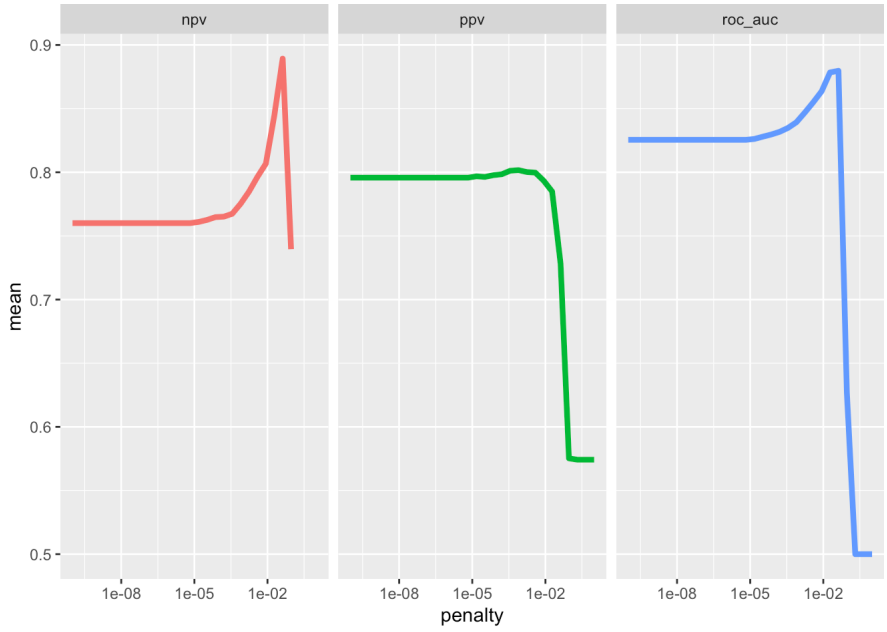
```
sent_recipe <- recipe(rating ~ text, data = sent_train) %>%
  step_tokenize(text) %>%
  step_stopwords(text) %>%
  step_tokenfilter(text, max_tokens = 500) %>%
  step_tfidf(text) %>%
  step_normalize(all_predictors())

sent_prep <- prep(sent_recipe)
#sent_prep
```

```
lasso_spec <- logistic_reg(penalty = tune(), mixture = 1) %>%
  set_engine("glmnet")

lasso_workflow <- workflow() %>%
  add_recipe(sent_recipe) %>%
  add_model(lasso_spec)
#lasso_workflow
```

```
lambda_grid <- grid_regular(penalty(), levels = 30)

set.seed(123)
sent_folds <- bootstraps(sent_train, strata = rating)
#sent_folds
```

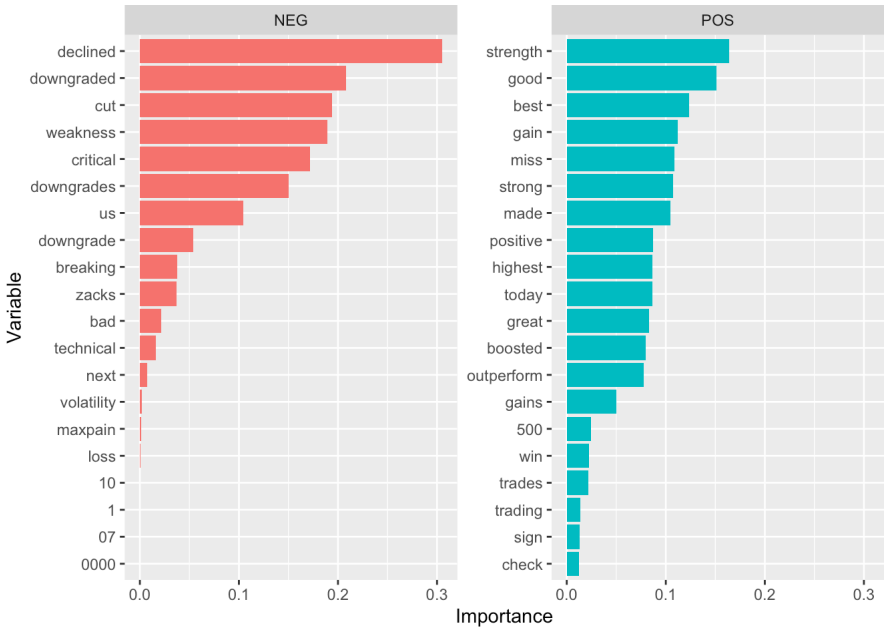
```
doParallel::registerDoParallel()

set.seed(2020)
lasso_grid <- tune_grid(
  lasso_workflow,
  resamples = sent_folds,
  grid = lambda_grid,
  metrics = metric_set(roc_auc, ppv, npv) #metrics to evaluate classification model
)
```

```
lasso_grid %>%
  collect_metrics() %>%
  ggplot(aes(penalty, mean, color = .metric)) +
  geom_line(size = 1.5, show.legend = FALSE) +
  facet_wrap(~.metric) +
  scale_x_log10()
```




```
best_auc_model <- lasso_grid %>%
  select_best()
#best_auc_model

final_lasso <- finalize_workflow(lasso_workflow, best_auc_model)
#final_lasso
```

```
final_lasso %>%
  fit(sent_train) %>%
  pull_workflow_fit() %>%
  vi(lambda = best_auc_model$penalty) %>%
  group_by(Sign) %>%
  arrange(desc(abs(Importance))) %>%
  slice_head(n = 20) %>%
  ungroup() %>%
  mutate(Importance = abs(Importance),
         Variable = str_remove(Variable, "tfidf_text_"),
         Variable = fct_reorder(Variable, Importance)) %>%
  ggplot(aes(x = Importance, y = Variable, fill = Sign)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~Sign, scales = "free_y")
```



```
sent_final <- last_fit(final_lasso, data_split) #final model

sent_final %>%
  collect_metrics()
```

```
## # A tibble: 3 × 4
##   .metric     .estimator .estimate .config
##   <chr>       <chr>          <dbl> <chr>
## 1 accuracy    binary         0.806 Preprocessor1_Model1
## 2 roc_auc     binary         0.914 Preprocessor1_Model1
## 3 brier_class binary         0.162 Preprocessor1_Model1
```

```
sent_final %>%
  collect_predictions() %>%
  conf_mat(rating, .pred_class)
```

```
##           Truth
## Prediction negative positive
##   negative      128       40
##   positive        5       59
```

Data Preprocessing:

- The first code chunk filters out tweets with sentiment scores of 0 from the stockerbot_df dataframe and creates a new dataframe called stockerbot_df_parsed.
- It then assigns sentiment labels ('positive' or 'negative') to tweets based on whether their sentiment scores exceed a threshold (0.1 in this case).
- The dataset is split into training and testing sets using an 80-20 split ratio, with the training set stored in **sent_train** and the testing set in **sent_test**.

Feature Engineering and Preparation:

- The second code chunk creates a recipe (**sent_recipe**) for text classification, where the target variable is **rating** (positive/negative sentiment) and the predictor is text (tweet content).
- The recipe consists of tokenization, stopwords removal, token filtering (limiting to 500 tokens), TF-IDF transformation, and normalization of predictors.
- The recipe is then prepared (**sent_prep**) to apply the defined transformations to the training data.

Model Training and Tuning:

- A logistic regression model with Lasso regularization is specified (lasso_spec), and a workflow (lasso_workflow) is created by combining the recipe and model.
- Hyperparameter tuning is performed using a grid search (tune_grid) with resampling via bootstrapping (sent_folds) to find the optimal penalty parameter for the Lasso model.
- The tuning results are visualized using a line plot to assess model performance across different penalty values.

Model Evaluation and Interpretation:

- The best-performing model based on the area under the ROC curve (AUC) is selected (**best_auc_model**).
- The finalized workflow (final_lasso) incorporates the optimal penalty parameter.
- Variable importance analysis is conducted using variable importance (vi) to identify the top features contributing to sentiment prediction.
- Finally, model performance metrics such as accuracy, precision, recall, and confusion matrix are computed and visualized for evaluation.

Conclusion:

- In summary, the provided R code demonstrates a comprehensive pipeline for sentiment analysis of tweets related to Fortune 500 companies. From data preprocessing and feature engineering to model training, tuning, and evaluation, each step is meticulously executed to build an effective predictive model for sentiment classification. By leveraging techniques like TF-IDF transformation, Lasso regularization, and hyperparameter tuning, the model achieves robust performance in identifying and interpreting sentiment from textual data, thereby providing valuable insights for decision-making and strategic planning.
- The final model achieves an accuracy of **81%**, indicating a commendable performance for a generalized sentiment analysis model.

Conclusion

In conclusion, the project successfully demonstrates the application of multiple techniques for sentiment analysis on Twitter data pertaining to Fortune 500 companies.

Dataset link

dataset (https://www.kaggle.com/datasets/omermetinn/tweets-about-the-top-companies-from-2015-to-2020)