

Text Analytics for Stock Market Trading

Group Assignment

Group 1

206036P	MSV Gimhan
206001F	M.M Aabidh
206066G	M.Lakshitha
206144U	W.A.A.M.S. Wijesinghe
206064A	K.A.R.P.Kumarasingha

Table of Content

Introduction	2
Data Extraction	2
Data Preprocessing	3
Data Exploration	4
Text Preprocessing	7
Data Types Conversion	7
Sentiment Analysis	11
Model Building and Selection	11
Classification Models	11
Classification Models Results Summary	12
Regression Models	13
Regression Models Results Summary	15
Conclusion	15

Introduction

The public's sentiment is very important to the stock market, and stock price movements can be greatly influenced by news stories and social media conversations. A viable method of forecasting stock market movements is to use text analytics to examine these attitudes. The goal of this research is to predict changes in stock prices by utilizing sentiment analysis on news articles and social media data. To improve trading methods and get insights into market behavior, we aim to create prediction models by combining sentiment scores obtained from textual sources with historical stock data.

The project consists of multiple crucial phases, the first of which is the selection and extraction of pertinent datasets spanning more than ten years. This includes sentiment on social media, historical stock prices, and news data from multiple sources. To guarantee accuracy and consistency, the data is painstakingly preprocessed, enabling sentiment analysis. Depending on the sentiment analysis, a variety of machine learning models will be used to forecast stock prices. The efficacy of these models will be assessed depending on how well they perform. The project's ultimate goal is to provide traders and investors with a useful tool by showcasing the potential profitability and resilience of trading techniques based on sentiment analysis.

Data Extraction

The data extraction process involves several key steps to acquire and prepare the datasets necessary for sentiment analysis and predictive modeling. The datasets used span different periods and sources, including news articles, social media posts, and historical stock prices. These datasets are sourced from Kaggle and Hugging Face for a diverse collection of textual data and Yahoo Finance for stock data.

Apart from the pandas library for data manipulation, the datasets library is installed to facilitate the downloading and handling of large datasets from Hugging Face. Next, Kaggle API was configured which is essential for downloading datasets directly from the Kaggle platform.



The first dataset consists of news headlines from Kaggle: Stock Market News and spans the years 2010 through 2016. For consistency, this data is filtered, cleaned, and reformatted. A second dataset from Kaggle: Massive Stock News Analysis, which covers the years 2012 to 2020, is similarly analyzed and turned into a pivot table that arranges headlines according to date. CNN news headlines from 2011 to 2022 are likewise arranged and matched with the other datasets. Hugging Face and Kaggle are the sources of social media sentiment data, which offers insightful information on public opinion. The data comprises tweets from 2022 to 2023 regarding Nvidia and stock market tweets from 2016 to 2019.

Yahoo Finance provides historical stock price data spanning from 2010 to 2023, which completes the extensive timescale required to link sentiment analysis with stock market performance. All datasets are cleaned and altered during the extraction process to make them suitable for predictive modeling. This thorough planning enables efficient sentiment analysis and predictive modeling by enabling a thorough examination of the impact of news and social media sentiment on stock prices. (The Data Exploration notebook will be attached in the Submission for further details).

Data Preprocessing

The shape and the structure of the dataset were first determined to analyze to obtain its fundamental properties under the data exploration phase. The dataset contained 3700 rows with 34 columns; "Index", "Date", "Close", "News1", "News2", ... "News30". This dataset had specifically 30 News columns and the "Close" column here refers to the Nvidia stock prices for the particular date.

Afterwards, the imported dataset was examined for null values and identified the presence of null values in the "Close" column and in several News columns. Initially, measures were taken to address the 403 null values in the "Close" column. To address this, the null values were filled by averaging the previous and the next day's closing process, guaranteeing the continuity of the data for subsequent analysis. This step avoided gaps that could skew further analysis and maintained the integrity of the data. It identified 30 separate news columns for each row in the original dataset and found out that having 30 distinct news columns could be challenging for the subsequent analysis. Hence, the data frame was adjusted by combining all these 30 separate columns into a single cell called "News_Text "

separating each news under a particular date with commas (,) and dropping the unnecessary column "Index", permitting for more meaningful and efficient analysis. Subsequently, the adjusted dataframe was checked for null values and found out the presence of 700 null values in the "News_Text " column, which consists of textual data. Given the difficulty of meaningfully filling the identified null values, it was opted to drop the brown with null values in "News_Text " column to continue the quality of the analysis and integrity. The below depicts the adjusted dataframe.

```
#get the adjusted dataframe
melted_df.head(50)
```

	Date	Close	News_Text
0	2010-01-04	4.6225	b'New airport scanners break child porn laws',...
1	2010-01-05	4.6900	b'These images depict the untouched stomach co...
2	2010-01-06	4.7200	b'Three Americans go to Uganda and teach thous...
3	2010-01-07	4.6275	b'23-year-old British woman on holiday in Duba...
4	2010-01-08	4.6375	b'Top Imams affiliated with the Islamic Suprem...
5	2010-01-11	4.5725	b'Why would a former Guantanamo Bay prison gua...
6	2010-01-12	4.4175	b'Dear World, The Chinese government stole int...
7	2010-01-13	4.4775	b'Please go to www.redcross.org and donate at ...
8	2010-01-14	4.4075	b'"It never ceases to amaze me that in times o...
9	2010-01-15	4.2775	b"France Calls for Cancellation of Haiti's Deb...
10	2010-01-19	4.3575	b'Haiti Struck By Major Earthquake for 2nd Tim...

Data Exploration

Under the EDA phase, it was determined to analyze the shape and the structure of the adjusted dataframe, resulting in 3000 rows with 3 columns; "Date", "Close", and "News_Text".

The descriptive summary of the dataset was then generated to uncover significant statistical measures associated with its numerical variables. Using the "describe" function, the statistics including count, mean, standard deviation, minimum, maximum, and

	Date	Close	News_Text
count	3000	3000.000000	3000
unique	NaN	NaN	3000

quartiles were computed for the “Close” variable. It could be seen that “Close” as a floating-point, “Date” as datetime and “News_Text” to be object type.

```
Date          datetime64[ns]
Close          float64
News_Text      object
dtype: object
```

Following that, the dataset was examined for null values to eliminate the potential for errors and inaccuracies during the analysis. The non-appearance of null values made sure the dataset’s reliability and integrity. Similarly, the dataset was then tested and discovered that there are no duplicate records in the dataset.

Several text preprocessing steps were conducted to identify any discrepancies or defects in the textual data. Regular expressions were then applied to investigate special characters, punctuations, emojis, URLs, and HTML tags. This step ensured the existence of the mentioned elements in the “News_Text” column. The significance of data preprocessing to address these discrepancies and to improve the quality of the subsequent analyses was revealed as a result of this step.

In addition, it was determined to carry out N-gram analyses for the “News_Text” column. Under that, Unigrams, Bigrams, and Trigrams were utilized to ensure frequent word clusters. This resulted in identification of common stopwords and special characters, requiring further preprocessing for analysis. Also, an effort was undertaken to figure out the frequently used parts of speech in reviews. It was regarded as a vital step to ensure a more precise and effective examination process.

Relevant preprocessing techniques and analysis methods can be chosen by utilizing the financial and analytical insights gained in the data exploration stage and by detecting the errors and characteristics of the dataset. It is important to note that the precise results could not be obtained from these interpretations as data exploration is conducted before the text preprocessing phase.

For this reason, it is important to ensure that the raw data is refined and structured in a conducive manner to meaningful analysis. The text preprocessing phase plays a vital role in the data analysis. It is determined to employ several techniques to clean and transform the dataset under this section with the aim of enhancing its quality and allowing for more deeper analysis.

Text Preprocessing

Data Types Conversion

Under this subsection, steps are first taken to convert the “News_Text” column in the dataset into string format to ensure the uniformity of the dataset. This step was regarded to be vital to maintain the consistency and avoid any problems with data types in the latter preprocessing steps.

Removal of Contractions

Here, the measures were taken to create a dictionary mapping the contractions to their expanded forms with the aim of addressing the issue of contractions and standardizing the content of the text. Since contractions can generate ambiguity and limit the accuracy of subsequent analyses specifically in tasks like sentiment analysis, this step was found to be very critical. A methodical technique was employed to swap each contraction in the “News_Text” column with its corresponding expanded version.

Removal of Special Characters

It has been recognized that the dataset’s special characters as potential causes of noise may reduce the accuracy of the analysis. Non-alphanumeric and non-whitespace characters are thus excluded methodically from the text utilizing a function that uses regular expressions as a means of addressing this issue.

Removal of Emojis

A function “remove_emoji”, was defined to remove emojis from the textual data. A regular expression pattern consisting of emoticons, symbols, and flags was introduced and thus eliminated from the reviews.

Removal of URLs

The existence of URLs in the textual data is one source of the noises that might hinder significant insights. Utilizing regular expressions, the strategy was taken to replace URLs with empty strings in an orderly manner, thus discarding them from the dataset. This step is also found to be very significant in enhancing the consistency of the text and clarity as it makes sure that analyses focus only on the important details revealed through the reviews.

Removal of HTML Tags

A function was created under the name of “remove_html_tags” to remove HTML tags from the movie reviews dataset. Using a regular expression pattern, HTML tags reflecting the formatting and structure of the web pages were eliminated.

Tokenization

A fundamental preprocessing technique named tokenization is applied to distinguish the text into independent words or terms. “punkt” package from the NLTK library is used to tokenize the “News_Text” column, creating a structured representation of the textual data. Each news is broken down into constituent tokens allowing for deeper examination and making further processing simpler. A new column called “News_Text_tokens” was formed to store the tokenized reviews, which serves as the basis for deeper text analysis and investigation.

Lowercasing

It is regarded that lowercasing the text to an appropriate level is important in continuing consistency and mitigating the complexity of further analyses. Variations in capitalization are neutralized by lowercasing the tokens in the “Text_News_tokens” column permitting for more accurate comparisons and computations.

Removal of Stopwords

Phrases such as “the”, “and”, “is”, etc. are some of the stopwords that are generally used in natural language and frequently have no deeper importance when analyzed. . Stopwords were removed from the dataset after each token was evaluated through NLTK library’s stopwords list. This process facilitates in lowering the dataset’s dimensionality and improves the effectiveness of the performance of subsequent analyses by focusing on meaningful content.

Removal of Rare Words

Identification and removal of rare words is crucial for mitigating noise and improving the quality of the dataset. The frequency of each token was calculated and the infrequently appearing words were regarded to be rare and measures were taken for removal of them. Although several attempts were made to remove those identified rare words, the need of high processing power and longer execution times restricted the completion of that process.

Stemming

The method “Stemming” which reduces the words to their base or root was employed to further condense their textual material, Porter stemming method was used to convert the words in the “News_Text_tokens” column to their matching root forms.

Lemmatization

The technique, Lemmatization is used combined with stemming to further improve the textual data by extracting its lemma, or basic form from the words. WordNetLemmatizer from the NLTK package was employed in lemmatizing the words in the “News_Text_tokens” column assuring the consistency and coherence throughout the dataset. Morphological variances and grammatical

nuances are considered by ensuring that this step produces more precise representation of textual content.

Spelling Correction

Considerable efforts were made to correct the spelling mistakes with the TextBlob library, however, limitations in the presence of resources, processing power and execution time restricted the completion of this. The focus on enhancing data accuracy and quality in the preprocessing phase was intended to accomplish by addressing the spelling mistakes in the text.

In summary, text preprocessing is regarded to be the procedure of carefully structuring and enhancing the raw data in the preparation for analysis. It is ensured that the dataset is cleansed and standardized utilizing the techniques; data type conversion, contraction removal, tokenization, etc., confirming integrity of the data and improving interpretability. Actionable insights and well-informed decision-making are made possible by these steps

3 Sentiment Analysis

This section includes sentiment analysis of the textual data using two pre-trained sentiment analysis models. The goal is to derive positive, negative, or neutral sentiment scores.

3.1 Utilizing Pre-trained Sentiment Analysis Models

1. VADER (Valence Aware Dictionary and Entailment Reasoner)

The VADER sentiment analysis tool is particularly effective for social media text. analyzer provides a normalized score between “-1”, the most extreme negative, and “1” the most extreme positive. VADER is a pre-trained model that is specifically designed for sentiment analysis on social media texts.

	News_Text_Final	VADER_Sentiment
0	new airport scanner break child porn law india...	-0.9871
1	imag depict untouch stomach content babi bird ...	-0.9964
2	three american go uganda teach thousand includ...	-0.9976
3	23yearold british woman holiday dubai told pol...	-0.9846
4	top imam affili islam suprem council canada is...	-0.9978
5	would former guantanamo bay prison guard track...	-0.9953
6	dear world chine govern stole intellectu prope...	-0.9840
7	plea go wwwredcrossorg donat least 10 help peo...	-0.9964
8	never ceas amaz time amaz human suffer somebod...	-0.9657
9	franc call cancel haiti debt photoshop disast ...	-0.9861

2. Using Text Blob

Text Blob is another tool for processing textual data. It provides a simple API for performing common natural language processing (NLP) tasks including sentiment analysis. Text-Blob returns scores range from -1 (negative) to 1 (positive). Textblob sentiment analysis considers the entire sentence structure and context to compute the negativity of the text.

	News_Text_Final	TextBlob_Sentiment
0	new airport scanner break child porn law india...	0.006238
1	imag depict untouch stomach content babi bird ...	0.044781
2	three american go uganda teach thousand includ...	0.028833
3	23yearold british woman holiday dubai told pol...	0.115128
4	top imam affili islam suprem council canada is...	-0.032591
5	would former guantanamo bay prison guard track...	-0.011828
6	dear world chine govern stole intellectu prope...	0.031211
7	plea go wwwredcrossorg donat least 10 help peo...	0.062336
8	never ceas amaz time amaz human suffer somebod...	-0.046164
9	franc call cancel haiti debt photoshop disast ...	0.052782

The results suggest that VADER and Text-Blob are superior to the custom-built model. Because they show their perfect accuracy on the test dataset of VEDER. These models come pre-trained and ready to use, requiring minimal setup. Because of that, they perform well even with small or no labeled training data.

Model Building and Selection

In order to find the best models for the stock price predictions both classification and regression models used and evaluated.

Classification Models

For the classification task, we began by preparing our dataset to ensure it was suitable for predictive modeling. We calculated the 'Tomorrow_Price' by shifting the 'Close' price column by one day, allowing us to determine the stock's trend by comparing today's closing price with tomorrow's. The trend was categorized into 'Buy' if the 'Tomorrow_Price' was higher than today's 'Close' price and 'Sell' if otherwise. To focus on relevant features, we removed the 'Tomorrow_Price' and 'News_Text' columns. Text data was normalized, tokenized, and transformed using TF-IDF vectorization, limiting the features to the top 10,000 most significant words. This resulted in a comprehensive feature set that integrated textual sentiment and stock prices.

Models and Justification

We employed a variety of machine learning models for our classification task, including Logistic Regression, Support Vector Machine (SVM), Gradient Boosting Classifier, K-Nearest Neighbors (KNN), and a Long Short-Term Memory (LSTM) network. Logistic Regression was chosen for its simplicity and interpretability, serving as a baseline. SVM was selected for its effectiveness in handling high-dimensional data, suitable for our large feature set from TF-IDF. Gradient Boosting was included for its ability to handle complex relationships and provide robust predictions through ensemble learning. KNN was considered due to its simplicity and intuitive approach of leveraging nearby data points. Finally, we experimented with LSTM, a deep learning model, to capture any temporal dependencies in the data, given its potential to understand sequential patterns in stock prices and sentiment.


Results and Evaluation

The models were evaluated based on accuracy, precision, recall, and F1-score. Logistic Regression yielded an accuracy of 50.79%, indicating modest performance. SVM performed better with an accuracy of 57.54%, highlighting its strength in handling complex feature spaces. Gradient Boosting, however, underperformed with an accuracy of 48.41%. KNN achieved a similar result to Logistic Regression with an accuracy of 48.81%. The LSTM model, despite its potential, delivered an accuracy of 47.22%, suggesting challenges in capturing temporal dependencies effectively within the given data constraints. SVM emerged as the best-performing model for classification, with a precision of 56.41% and recall of 57.54%, indicating a balanced performance in predicting stock trends based on textual sentiment.

Classification Models Results Summary

Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression	0.5079	0.5191	0.5079	0.5096
Support Vector Machine (SVM)	0.5754	0.5641	0.5754	0.5569
Gradient Boosting	0.4841	0.4737	0.4841	0.4765
K-Nearest Neighbors (KNN)	0.4881	0.4976	0.4881	0.4901
Long Short-Term Memory (LSTM)	0.4722	0.4938	0.4722	0.4673

Regression Models



For the regression task, the feature engineering process began similarly by preparing the dataset and removing the last row to avoid future data leakage. We calculated 'Tomorrow_Price' to be used as our target variable. The 'News_Text' column was also removed to focus on numeric predictors. The text data underwent TF-IDF vectorization, generating a matrix of significant features (limited to 1,000 to prevent overfitting). This vectorized text was then concatenated with the original numerical features, resulting in a comprehensive dataset ready for regression analysis.

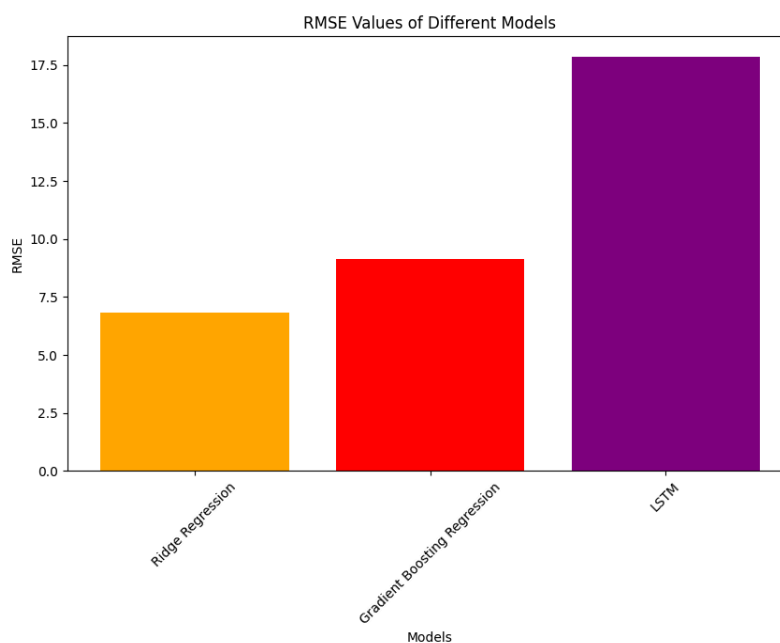
Models and Justification

We tested several regression models: Linear Regression, Ridge Regression, Gradient Boosting Regressor, and an LSTM network. Linear Regression was selected for its straightforward approach to understanding the relationship between input features and the target variable. Ridge Regression, a regularized version of linear regression, was chosen to handle potential multicollinearity and prevent overfitting. Gradient Boosting Regressor was included for its ability to build an ensemble of weak learners to provide robust predictions. Lastly, we utilized an LSTM network to capture any sequential patterns in stock price movements, leveraging its capability to learn long-term dependencies.

Results and Evaluation

Model performance was evaluated using Root Mean Squared Error (RMSE) to measure prediction accuracy. Linear Regression surprisingly resulted in an impractically high RMSE of 16.23 billion, indicating it failed to capture the stock price dynamics effectively. Ridge Regression performed significantly better, with an RMSE of 6.82, suggesting it effectively handled the feature set. Gradient Boosting Regressor yielded an RMSE of 9.14, demonstrating moderate performance but underperforming compared to Ridge Regression. The LSTM model, despite its advanced architecture, resulted in the highest RMSE of 17.85, indicating it struggled with the dataset's complexity and temporal aspects. Ridge Regression stood out as the best-performing model, indicating its effectiveness in handling the provided features while minimizing prediction errors.

Below are the top 3 models RMSE values.



Regression Models Results Summary

Model	RMSE
Linear Regression	16226987755.8131
Ridge Regression	6.8229
Gradient Boosting Regressor	9.1362
Long Short-Term Memory (LSTM)	17.8481

Conclusion

Based on the results from both classification and regression models, the Ridge Regression model demonstrated the most robust performance with the lowest RMSE, suggesting it captured the relationship between features and stock prices more effectively than other models. Although the SVM model showed promise in the classification task, predicting whether to buy or sell, its precision and recall were only moderately better than other models. Given the objective of predicting stock price movements with higher accuracy, the Ridge Regression model is recommended for further development and potential deployment. This model's ability to handle a broad range of features and regularize to prevent overfitting makes it the most reliable choice for achieving accurate and actionable stock price predictions.