Sentiment Analysis Techniques Comparison for Stock Prediction

Abstract

This document compares four sentiment analysis techniques—FinBERT, TextBlob, VADER, and Flair applied to a financial dataset for stock prediction. The study evaluates the models based on performance metrics, computational resources, interpretability, and generalization to provide insights for choosing the most suitable technique.

Dataset Description

A dataset of financial news was collected for several stocks/companies in various sectors to evaluate the role of sentiment analysis in predicting the stock market. the stocks are: 'PG', 'META', 'AMD', 'NFLX', 'TSM', 'AAPL', 'GOOGL', 'MSFT', 'AMZN', 'TSLA', 'SPY', 'QQQ', 'JPM', 'BAC', 'GS', and 'C'.

The dataset contains more than 20000 news entries for all the companies. A news entry can be either a news article or a news video to make sure more news is covered.

The dataset has 8 columns as follows:

**ticker**: indicates the stock symbol e.g. AAPL for Apple.

**date**: the exact date the news was published.

**title**: the title of the news article/video.

**text**: description of the news story.

**source**: what channel or news agency published the story.

**url**: news URL.

**type**: whether the news entry is an article or a video.

**sentiment**: the sentiment label of that news story i.e. Positive, Neutral, or Negative.

3. Methodology

­­The process for applying the four techniques was identical to be able to evaluate their performance against each other and pick one model for our research.

All four implementations followed this structure:

Data preprocessing:

* Data Cleaning:
  + Handle Missing Values:
    - Identify and handle any missing values in the dataset.
  + Remove Duplicates:
    - Check for and remove any duplicate records in the dataset.
* Text Cleaning:
  + Lowercasing:
    - Convert all text to lowercase to ensure consistency.
  + Remove Punctuation:
    - Eliminate unnecessary punctuation from the text data.
  + Tokenization:
    - Tokenize the text into individual words or phrases.
  + Remove Stop Words:
    - Exclude common stop words (e.g., "the," "and," "is") that may not contribute significantly to sentiment analysis.
  + Lemmatization:
    - Lemmatizing each token by returning each word/verb to its root word.
* Concatenating two columns:
  + To have more text to analyze the sentiment, we concatenated the “text” column with the “title” column to have more words, thus, getting a more accurate classification of the sentiment.

3.1 FinBERT

* Sentiment Analysis Function:
  + The methodology involves the utilization of a pre-trained FinBERT model. The model is loaded, and a sentiment analysis function is introduced. This function leverages FinBERT to predict sentiment labels for each processed text.
* Applying FinBERT Sentiment Analysis to the Dataset:
  + The sentiment prediction function is applied to each pre-processed text in the dataset. This results in a new column named finbert\_label containing the predicted sentiment labels.

3.2 TextBlob

The analysis begins by importing the TextBlob library.

* Defining Sentiment Analysis Functions:
  + get\_polarity(): Measures the positivity or negativity of the text with a score between [-1,1]
  + get\_subjectivity(): measures the subjectiveness of the text.
* Applying Sentiment Analysis to the Dataset:
  + These functions are then applied to the pre-processed text in the dataset, generating numerical values for polarity and subjectivity.
* Defining Labeling Function to simplify interpretation, a function named get\_label is introduced. This function categorizes the sentiment based on the calculated polarity:
  + Positive for values greater than 0.
  + Negative for values less than 0.
  + Neutral for values equal to 0.

3.3 VADER

* VADER Sentiment Analysis Function:
  + This function uses the SentimentIntensityAnalyzer from the NLTK library. It assesses the sentiment of each processed text, generating a compound score that indicates the positivity and negativity of the text sentiment.
* Labelling Function:
  + This function assigns sentiment labels ("Positive," "Negative," or "Neutral") based on the compound score.
* Creating Sentiment Labels:
  + The labelling function is then applied to categorize the sentiment of each text based on the computed compound scores. The resulting sentiment labels are stored in a new column named VADER\_label in the DataFrame.

3.4 Flair

* Defining Sentiment Analysis Functions:
  + get\_score(text): a function designed to retrieve sentiment scores. For each processed text, a Flair Sentence is created, and a pre-trained sentiment classifier is utilized to predict the sentiment score.
  + get\_label(text): it utilizes the Flair Sentence and sentiment classifier to predict the sentiment label associated with each processed text.
* Applying Flair Sentiment Analysis to the Dataset:
  + The get\_label() function is applied to each pre-processed text in the dataset. This results in a new column named flair\_label containing the predicted sentiment labels.

4. Evaluation Metrics

The evaluation metrics for comparing the models will be:

accuracy: the ratio of correctly predicted instances to the total instances.

precision: the ratio of correctly predicted positive instances to the total predicted positive.

recall: the ratio of correctly predicted positives to the total observations in the class.

F1-score: the weighted average of precision and recall.

Confusion Matrix: a table illustrating the counts of true positives, true negatives, false positives, and false negatives.

5. Model Performance

5.1 FinBERT

On the full dataset

A screenshot of a computer

Description automatically generatedA comparison of a bar chart

Description automatically generated

On the small dataset

A screenshot of a computer

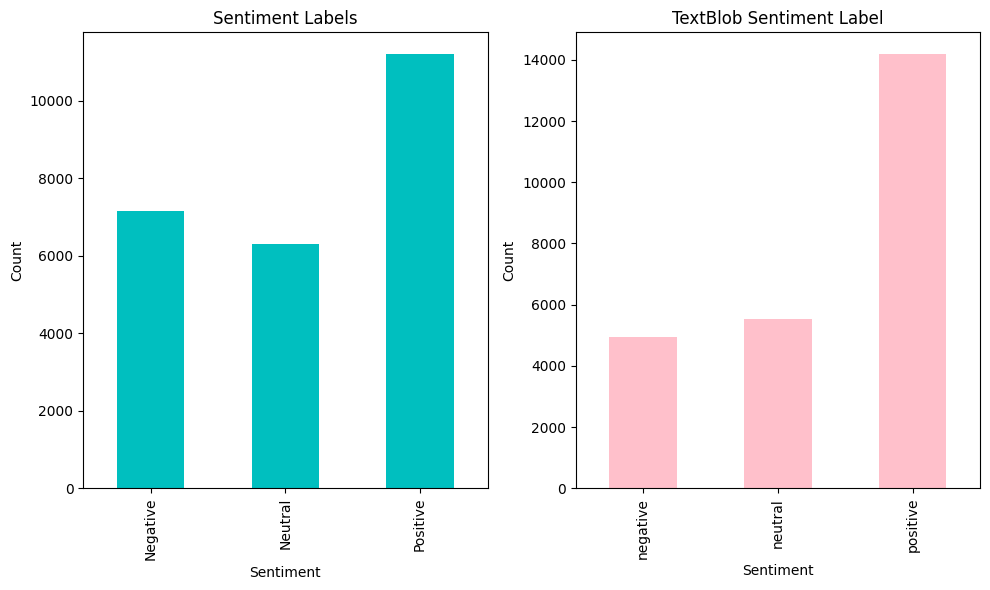
Description automatically generatedA comparison of a bar chart

Description automatically generated with medium confidence

5.2 TextBlob

On the full dataset

A screenshot of a computer

Description automatically generated

On the small dataset

A screenshot of a computer

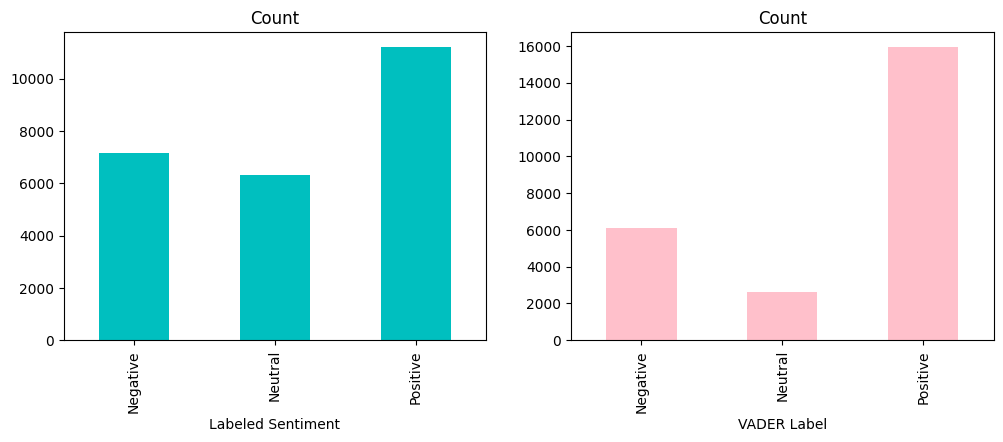
Description automatically generated A comparison of a bar chart

Description automatically generated with medium confidence

5.3 VADER

On the full dataset

A screenshot of a computer

Description automatically generated

On the small dataset

A screenshot of a computer screen

Description automatically generatedA close-up of a graph

Description automatically generated

5.4 Flair

Took more than 300 mins and didn’t finish running

6. Analysis and Conclusion

On full dataset

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Accuracy | Precision | Recall | F1-Score |
| FinBERT | 0.12 | 0.20 | 0.12 | 0.15 |
| TextBlob | 0.45 | 0.44 | 0.44 | 0.44 |
| VADER | 0.54 | 0.51 | 0.54 | 0.51 |

based on each mode’s performance on our full dataset of financial news, we can see that VADER performed the best overall.

On the small dataset

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Accuracy | Precision | Recall | F1-Score |
| FinBERT | 0.67 | 0.64 | 0.67 | 0.63 |
| TextBlob | 0.47 | 0.49 | 0.47 | 0.48 |
| VADER | 0.55 | 0.61 | 0.55 | 0.55 |

based on each mode’s performance on our small dataset of financial news, we can see that FinBERT performed the best overall.

Time taken for each model to analyse the entire dataset.

|  |  |
| --- | --- |
| Model | Time Taken |
| FinBERT | 16m40s |
| TextBlob | 5.2s |
| VADER | 3.2s |

Based on the results of comparing both the performance and the computational time, we will choose **VADER** for the purpose of this research.

7. References