CS 4120: Natural Language Processing
Eric Shi, Ayush Rana, Rachel Chen
Group 140

Final Project Report

Title
Financial News Sentiment Analyzer

Abstract
The Financial News Sentiment Analyzer project leverages natural language processing models to analyze sentiments in financial news articles. Utilizing web-scraped data from Market Insiders, a leading financial news provider, the project captured content from articles published over a 30-day period, focusing on a selected group of high-profile stocks known as the "Magnificent Seven" (MSFT, APPL, TSLA, NVDA, GOOG, META, AMZN), along with additional randomly selected stocks. The project's core components involve two models: a Logistic Regression model for classifying text within the articles as relevant content or publisher's metadata (achieving 98% accuracy), and a sentiment analysis model built on FinBERT, a BERT model pre-trained on financial texts. The sentiment model evaluated the collected articles, identifying predominant sentiments among the stocks discussed, with results showing a majority of negative sentiments. Future work will focus on improving the logistic model's sensitivity to non-keyword-based relevancy and exploring methods to assess the impact of news on stock relevance directly. The project has demonstrated the potential of targeted NLP techniques in extracting and analyzing nuanced sentiment in financial news, offering valuable insights for financial analytics and decision-making.

Introduction
There is an overwhelming amount of volume and velocity of financial news available today, yet this information is crucial for market participants who rely on timely and accurate data to make informed trading decisions. This deluge of data poses a challenge: how to efficiently sift through vast amounts of news and extract actionable insights. The Financial News Sentiment Analyzer project addresses this challenge by applying advanced natural language processing (NLP) techniques to analyze the sentiment of financial news articles and assess its impact on stock movements.

We based our product on the key assumption that financial markets are sensitive to news and public sentiment, where the tone and content of news can influence stock prices significantly. This is an assumption that is well-backed by research (*The effect of news and public mood on stock movements*, Qing Li et. al., 2014). Manually analyzing the sentiment of financial news is impractical due to the sheer amount of content generated daily. Additionally, the subjective

interpretation of news can lead to inconsistencies in analysis that can cost stakeholders money. It is clear that there is a need for an automated tool that can consistently and accurately analyze the sentiment of financial news to aid investors, analysts, and financial institutions in their decision-making processes.

Previous studies and tools have demonstrated the influence of news sentiment on market dynamics. Many approaches utilize basic NLP techniques for sentiment analysis, such as polarity scores from predefined lexicons. However, these methods often fail to capture the nuances of financial language and can misinterpret the context. More recent developments have employed machine learning models, including deep learning techniques, which offer improvements by learning from large datasets but still struggle with the specificity and jargon of financial news.

Our Financial News Sentiment Analyzer projects builds on these foundations by using two natural language models. First, the Logistic Regression Model which classifies paragraphs within financial articles as relevant content or as part of the publisher's non-essential metadata with a high degree of accuracy. Then, the FinBERT Sentiment Analysis Model. The FinBERT model is a variant of the BERT model pre-trained specifically on financial texts. This allows us to apply the BERT model to process financial content and best analyze the sentiment of the content deemed relevant by the logistic model. This approach allows for a nuanced understanding of financial texts, recognizing complex expressions of sentiments which are typical in economic articles.

We applied these models to a dataset consisting of web scrape articles about the "Magnificent Seven" stocks over a 30-day period. The result revealed a predominance of negative sentiments within the "Magnificent Seven." This insight aligns with current market trends observed as the tech industry is experiencing a period of high volatility and negative press. The findings underscore the capability of our NLP models to not only parse large volumes of data but also provide insights that are in concordance with market conditions.

Methods

I. Data Collection and Processing

The project utilized web-scraping techniques to gather financial news data from Market Insiders. Of the five available publishers, four were accessed that did not require account creation, ensuring a smooth data extraction process. We collected data spanning a 30-day period, focusing on stocks from the "Magnificent Seven" as well as additional stocks selected randomly. This effort resulted in two primary datasets:

1. paragraph-1.csv: This dataset contains unique paragraphs from various news articles over 15 days, detailing stocks such as CRM, NFLX, BA, and DIS.

2. stockNews-1.csv: This dataset stocked details about the stock news, including their stock ticker, news title, publication date, link to the full article, the complete news text, and a list of paragraphs split from the full text.

II. Model Development

The analysis incorporated two main NLP models to process and analyze the text data:

1. Logistic Regression Model for Text Classification: This model was developed to discern the relevancy of paragraphs within the news articles. It classifies each paragraph as either relevant news content or non-essential elements like advertisements or publisher's metadata.
2. FinBERT Sentiment Analysis Model: Utilizing the FinBERT model, a specialized BERT model pre-trained on financial texts, this model assesses the sentiment of the paragraphs deemed relevant by the logistic regression model. The sentiment analysis results are categorized into negative, positive, and neutral sentiments, allowing for a nuanced understanding of market sentiment as reflected in news coverage.

Data

From our web-scraping softwares, we ended up utilizing data from Market Insiders, a common financial news provider. From Market Insiders, there were five unique publishers available. Of these five, there were four sources that allowed us to access the news article without creating an account. Hence, these four sources were scraped. Market Insiders allows for news dating 13 years prior, so for the instance of our project, we only gathered 30 days worth of articles for the Magnificent Seven stocks for testing. Using our web scraper, we extracted information into paragraph-1.csv, a csv file with a header consisting of 'ticker' (stock ticker), 'link' (link to the article), 'paragraph' (paragraphs of text within a news article)', and 'relevant' (whether the paragraph belongs to the news article or is part of an ad/closing within the same page). This csv consists of 5,071 rows of unique paragraphs mined from the past 15 days worth of news articles from each of CRM, NFLX, BA, DIS (we took random stocks). We also mined a similar csv file named stockNews-1.csv, which contains the headers: 'Unnamed: 0' (row index) , 'ticker'(stock ticker), 'title' (news title), 'date' (news publish date), 'link' (link to article), 'articleInfo' (news text), 'paragraphList' (news text split by paragraphs, captured within a list). This csv consists of 1,780 rows of unique news articles mined from the past 30 days worth of news articles regarding the Magnificent Seven.

Results

The first part of our project that requires NLP techniques is building out a Logistic Regression model for the sake of text classification. The model takes in different paragraphs within a news article and identifies whether the paragraph is a part of the news article or a part of the publisher's signature/ad/closing lines. The accuracy of this model was around 98 percent since every paragraph that is identified to not exist as part of the article exhibits some sort of similar pattern. The other part of the project was centered around the creation of a sentiment analysis model utilizing FinBERT, a BERT model pre-trained on past financial news data. After applying our model on the stockNews-1.csv file, we were left with an observed majority of positive sentiment at 800 articles, 470 negative articles, and 294 neutral articles. Using the current state of the market as a benchmark, we should be expecting a good number of articles starting to mention that in face of economic downturn, we as investors should continue holding and buying these popular companies.

Discussion

At the stage of our project presentation, we were fixated on fine-tuning our FinBERT model. There was an overwhelming number of negative sentiment at that stage and it did not seem reasonable. By removing parts of articles that are not relevant to the text and could give way to varying sentiment values, we were able to adjust and tune our sentiment analysis to be more reasonable and possibly more accurate. However, concerning our Logistic Regression model, we had to manually assign relevancy scores to all the paragraphs based on patterned key words. This model is then capable of capturing most instances where the key words exist, but incapable of finding other instances where the keywords do not exist but the paragraph is not part of the news. Our next step will be training this model to recognize more instances of non-news paragraphs. Additionally, we would like to determine relevancy of a certain stock within an article. This was something we had targeted as next-steps during our presentation, but it is difficult to find a way that does so while accounting for relevant terminology (ex: if we want to find the relevancy of Apple within an article, we also have to find all the terms such as iPad, iPhone, etc. that are relevant to Apple and then find an aggregated relevance score involving all these terms and how often they were mentioned.

Appendix

- External dependencies: sklearn, torch, numpy, pandas, nltk, imblearn, ast, ProsusAI/finbert

- Functions:
  - **sliding_window**: create chunks of text
  - **process_chunks**: take list of chunks and return sentiment scores and probabilities
  - **aggregate_results**: finds average sentiment and probability
  - **full_window**: pipeline process of sliding_window, process_chunks, and aggregate_results
  - **preprocess**: preprocess a string of text via common NLP techniques
  - **verify_logreg**: predict relevancy of paragraphs of text

- Code Structure
  - webScrape.ipynb
    - This web scrapes the magnificent 7 stocks and creates a csv where each row is a ticker symbol, single article related to symbol, and date of article. The csv is called **stockNew-1.csv**
    - It also creates csv called **paragraph-train.csv** which is list of paragraphs and true or false saying if the paragraph is relevant to the stock and article
  - Logregmodel.ipynb
    - This is where we create our logistic regression model to identify whether a paragraph is relevant to a stock or not by training it on the data from paragraph-train.csv
    - We then use this model to filter **stockNew-1.csv** of irrelevant paragraphs to come up with **FinalStockNews.csv**
  - bertModeling.ipynb
    - Here we import the finbert model and use it to come up with sentiment scores for each article in **FinalStockNews.csv**, then we output **dfSentiment.csv** which includes the ticker, article, article info, and sentiment score
  - demo.py
    - This is our demo file, here we import **dfSentiment.csv** and a user inputs, ticker, startDate, endDate and the user receives a sentiment score between 0 and 1 for the ticker between those dates
  - Logregmodel_tests.py
    - Unit tests for the functions involving the logistic regression model
  - Sentiment_tests.py
    - Unit tests for the functions involving the sentiment analysis function

- Inputs and Outputs
    - In demo.py the user imports a ticker, startDate, endDate
    - The ticker must be one of ‘AMZN','TSLA','META','MSFT','GOOG','NVDA','AAPL’, since that is what we scraped for
    - The dates must be between 2024-03-15 and 2024-04-15 in that string format since that is the data we collected
    - The output is a score between 0-1 representing the sentiment score for the ticker between those dates
    - 0 is the negative end of the spectrum and 1 is the positive end of the spectrum

- User Manual
    - To run the code, go to demo.py call getSentimentScore method, with inputs of ticker, startDate, endDate, and output is the String printed showing the sentiment score of the stock in the given timeframe
    - If you want to rescrape and re train the models to update them for current news or change tickers
        - First run all cells in webScrape.ipynb to completion, then run all cells in logregmodel.ipynb to completion, then run all cells sentiment.ipynb to completion
        - Then go to demo.py file and input ticker, startDate, and endDate like described above
    - If you want to run the unit tests for our functions, you can go to the [file_name]_tests.py and execute `python [file_name]_tests.py`

References

- https://www.youtube.com/watch?v=HR2amKqWU-8&ab_channel=BrandonHarding
- https://www.youtube.com/watch?v=uj4hm7Lr2Wo&t=74s&ab_channel=RitheshSreenivasan
- https://huggingface.co/ProsusAI/finbert
- https://markets.businessinsider.com/
- https://www.sciencedirect.com/science/article/pii/S0020025514003879