NLP Individual Assignment 1

Mehdi Greefhorst

October 2024

Contents

1 Data exploration and creation

1.1 Data creation

In this project, we focused on creating a relevant corpus of news articles related to Nvidia (NVDA) stock and labeling them based on the stock's price movements. The process involved several key steps:

1. **Defining the target variable**: We created a binary classification label based on the stock's daily price movement. If the closing price was higher than the opening price, we labeled it as an increase (1); otherwise, it was labeled as a decrease (0).

2. **Creating a date-to-label dictionary**: We mapped each trading date to its corresponding label (increase or decrease) based on the stock's price movement.

3. **Labeling news articles**: We added these labels to our news articles dataset by matching the article's publication date with the corresponding label in our dictionary.

4. **Handling non-trading days**: For articles published on non-trading days (e.g., weekends or holidays), we implemented a fallback mechanism. In these cases, we searched for the most recent previous trading day with an available label.

5. **Improving NVDA ticker identification**: To overcome the limitation of single-ticker labeling in our original dataset, we created an "improved NVDA ticker" indicator. This indicator is set to true if any of the following conditions are met:

- The article's original ticker is NVDA
- The ticker symbol "NVDA" appears in the article's title or content
- The company name "NVIDIA" appears in the article's title or content
-

6. **Document filtering**: We filtered our corpus to include only documents that matched our improved NVDA ticker criteria, ensuring a more comprehensive and relevant dataset for our analysis.

This approach allowed us to create a more robust and relevant corpus for our experiments, addressing potential gaps in the original dataset and ensuring that our analysis captures a wider range of Nvidia-related news articles. The improved ticker identification helps to mitigate the risk of missing relevant articles due to limitations in the original ticker labeling.

There were more than 200.000 articles in the dataset. After filtering there were 3.467 articles left.

1.2 Preprocessing Data - defining variables

Our preprocessing pipeline was designed to transform the raw text data into a more structured and meaningful format suitable for analysis. The input data consisted of article titles and content in raw string format. We implemented the following preprocessing steps:

1. **Tokenization**: We used regular expressions to separate the text into a list of individual tokens (words and symbols). This step was applied to both the title and content of each article.

2. **Lowercasing**: All tokens were converted to lowercase to ensure consistency and simplify subsequent processing steps.

3. **Numerical and Single-Letter Token Removal**: We filtered out tokens based on specific criteria:

- Pure numerical strings (e.g., "123") were removed as they typically don't carry meaningful textual information.
- However, alphanumeric tokens like "Q4" were retained due to their potential significance in financial contexts.
- Single-letter tokens were eliminated as they often represent stop words or artifacts from the original text formatting.

4. **Stop Word Removal**: We utilized the NLTK package's English language stop word list to remove common words that typically don't contribute significant meaning to the text.

5. **Lemmatization**: We applied the Porter-Stemmer algorithm to lemmatize the remaining tokens. This step reduces words to their base or dictionary form, helping to decrease vocabulary size and consolidate word meanings.

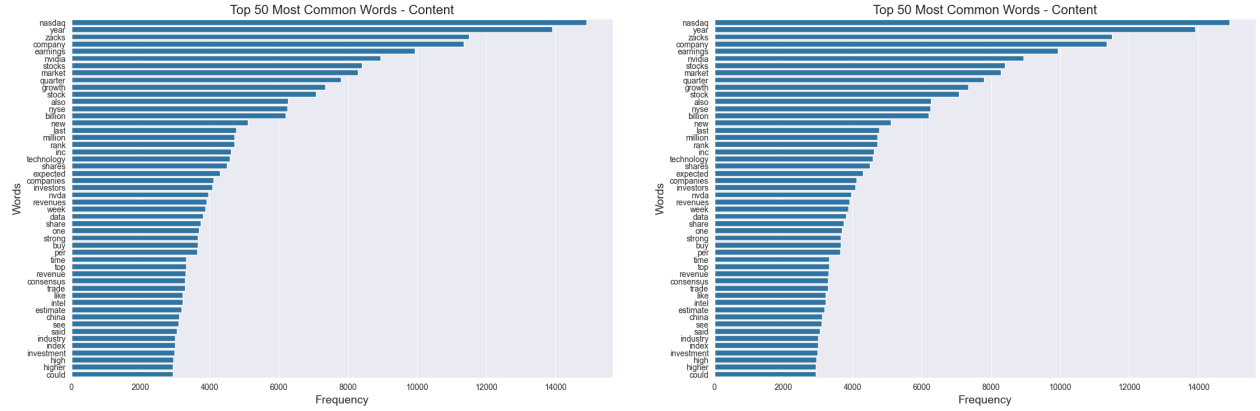
It's worth noting that the input data had already undergone some preprocessing before we received it. Specifically, the text of the title and content have been stripped from punctuation marks and have been replaced with spaces, which occasionally led to ambiguities (e.g., "U.S." becoming "U S"). Our preprocessing pipeline was designed to address these issues and further refine the text data.

The provided text in the data was already stripped from punctuation and were replaced by spaces, leading to abundance of single letter tokens.

The result of this preprocessing pipeline was two separate lists of tokens for each article: one for the title and one for the content. These processed token lists formed the foundation for our subsequent analysis and modeling steps.

1.3 Analysis of processed data

Our analysis of the processed data revealed interesting patterns in the vocabulary used in article titles versus content, as well as words indicative of stock price movements. We present these findings through several visualizations and statistical analyses.

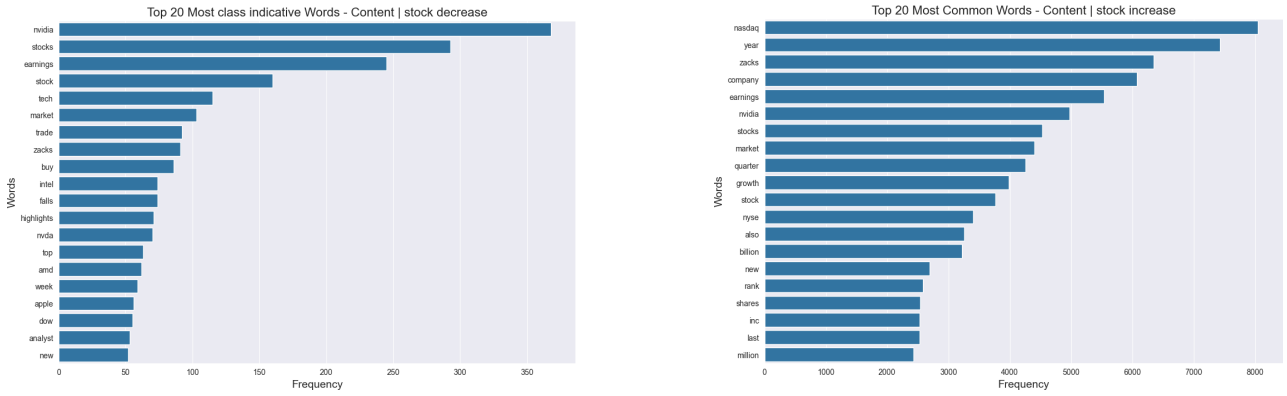


(a) 50 Most Common Words in Titles

(b) 50 Most Common Words in Content

Figure 1: Comparison of Most Common Words in Titles and Content

Figure 1 presents the 50 most common words in article titles (Fig. 1a) and content (Fig. 1b). Interestingly, we observe a notable difference between these two sets. The titles frequently mention "NVIDIA", suggesting a direct focus on the company. However, the content appears to have a more diverse vocabulary, potentially covering broader market trends or related technologies.



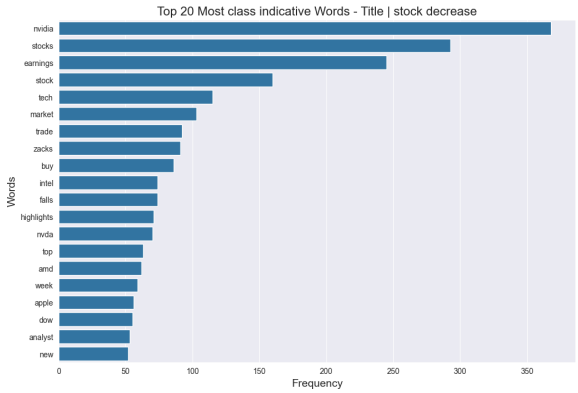
(a) Top 20 Most Class Indicative Words - Content (Stock Decrease)

(b) Top 20 Most Class Indicative Words - Content (Stock Increase)

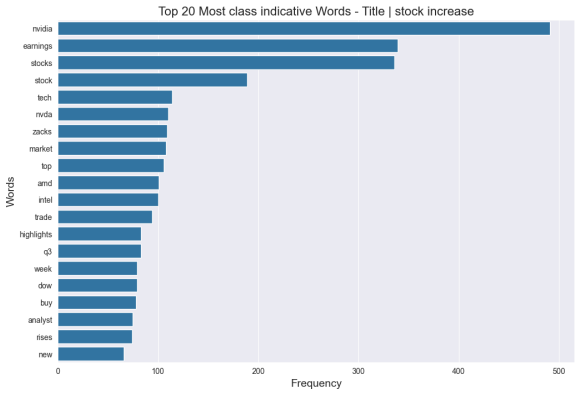
Figure 2: Top 20 Most Class Indicative Words for Content, showing words for stock increase and decrease.

Figure 3 displays the 20 most indicative words for each class (stock price increase or decrease) in titles. Figure 2 displays the 20 most indicative words for each class (stock price increase and decrease.

Table 1 presents key statistics about our corpus, including the number of unique words, total words, and lexical variation for both titles and content. The stark difference in lexical variation between titles (0.13) and content (0.02) is particularly noteworthy. This suggests that there are established patterns or "rules of thumb" in writing article titles for NVIDIA-related news. The higher lexical variation in titles indicates a more diverse vocabulary, likely chosen to catch readers' attention and encourage them to click on the article.



(a) Top 20 Most Class Indicative Words - Title (Stock Decrease)



(b) Top 20 Most Common Words - Title (Stock Increase)

Figure 3: Top 20 Most Class Indicative and Common Words for Title, showing words for stock increase and decrease.

Figure 1 illustrates the lexical variation of the entire corpus. Interestingly, when considering the corpus as a whole, the lexical variation for both title and content converges to 0.02. This suggests that while titles have more varied vocabulary when considered independently, the overall corpus exhibits similar lexical diversity across both titles and content.

The distinct patterns observed in titles versus content vocabulary usage suggest that treating these as separate features could potentially improve our model's predictive power. While this approach wasn't implemented in the current study, it presents an interesting avenue for future work, particularly for ensemble methods like Random Forest that can effectively leverage multiple feature sets.

Measure	**Title**	**Content**
Unique Tokens	4,481	36,680
Lexical Richness	0.138	0.015

Table 1: Comparison of Unique Tokens and Lexical Richness between Title and Content

2 Modeling

2.1 Numerical representation

What were your findings when using TF-IDF for document representation? Also, what are the pros and cons of training your own Word2Vec representation vs. downloading a pretrained one? Which Word2Vec algorithm did you use and why (CBOW vs Skip-Gram)? How do you select your hyperparameters? Do you train the model on the whole corpus or only on the selected articles? How do you convert your word embeddings into a document embedding?

Our approach to numerical representation involved several key steps and decisions, balancing between accuracy, computational efficiency, and practical constraints.

2.1.1 TF-IDF and Document Similarity

We initially used TF-IDF (Term Frequency-Inverse Document Frequency) to represent our documents and computed cosine similarity between all document pairs. This process, while informative, had a time complexity of $O(n\hat{2}), which proved to be computationally expensive for our dataset. In retrospect, this could have been optimized using matrix calculations, highlighting an area for future improvement in our methodology.

The similarity analysis yielded an interesting finding: some documents showed extremely high similarity scores. Upon closer inspection, these turned out to be template-based articles where only numerical values differed.

Figure 2 shows an example of two such highly similar documents. Our preprocessing steps, which removed purely numerical tokens, resulted in these documents appearing nearly identical in their processed form. This observation underscores the importance of carefully considering preprocessing steps and their potential impact on downstream analysis.

Article	Article 2963	Article 2965
Similarity	1.0	
Content	*Investing.com: NVIDIA (NASDAQ: NVDA) stock rose by 3.01% to trade at $247.42 by 13:53 (18:53 GMT) on Tuesday on the NASDAQ exchange. The volume of NVIDIA shares traded since the start of the session was 4.97M. NVIDIA has traded in a range of $240.72 to $247.42 on the day. The stock has traded at $259.37 at its highest and $232.71 at its lowest during the past seven days.*	*Investing.com: NVIDIA (NASDAQ: NVDA) stock rose by 3.03% to trade at $247.61 by 11:32 (16:32 GMT) on Tuesday on the NASDAQ exchange. The volume of NVIDIA shares traded since the start of the session was 3.15M. NVIDIA has traded in a range of $244.05 to $247.75 on the day. The stock has traded at $249.25 at its highest and $232.71 at its lowest during the past seven days.*

Table 2: Comparison of Article 2963 and Article 2965, with a similarity score of 1.0. Both articles describe similar price movements and stock ranges for NVIDIA on the NASDAQ exchange.

2.1.2 Word2Vec Representation

For our word embedding approach, we chose to train our own Word2Vec model rather than using a pre-trained one. This decision was motivated by several factors:

1. Project constraints: Using pre-trained models was not permitted for this project.
2. Memory efficiency: A custom-trained model on our specific corpus (focused on NVIDIA) requires significantly less memory than a comprehensive pre-trained model.
3. Domain specificity: A model trained on our NVIDIA-focused corpus captures domain-specific semantic relationships more accurately [1, 2].

We implemented the Continuous Bag of Words (CBOW) variant of Word2Vec, primarily due to its superior training speed compared to the Skip-gram model [3]. This choice was driven by our self-imposed requirement for computational efficiency, particularly considering the potential for scaling the system to handle a much larger corpus of news articles from various sources.

Hyperparameters for our Word2Vec model were selected through grid search, with the final configuration aligning well with parameters used in similar financial news prediction tasks [4, 5]. The specific hyperparameters used were:

Vector Size	Window	Min Count	SG (Skip-Gram)	Epochs
100	5	5	0 (CBOW)	5

Table 3: Best Parameters for the Word2Vec Model

The model was trained exclusively on articles related to NVIDIA, identified either by the NVIDIA ticker or explicit mentions of the company name. While this approach ensured relevance, future iterations could potentially benefit from including articles about competitors and related products to provide broader context.

2.1.3 Document Embedding

To convert our word embeddings into document embeddings, we used a simple averaging approach. Each document was represented by the average of its constituent word vectors. This method was chosen for its simplicity and computational efficiency, which were key considerations for this project [4, 5]. While more sophisticated approaches exist, this method provided a good balance between performance and complexity.

This numerical representation approach, combining TF-IDF for similarity analysis and custom-trained Word2Vec embeddings for document representation, provided us with a solid foundation for our subsequent modeling steps. The choices made in this phase were heavily influenced by the need for computational efficiency, highlighting the practical constraints often encountered in real-world machine learning projects.

2.2 Training procedure

How do you train your model, i.e. obtain the optimal parameters? What is the experimental design that you use for training and evaluating the model? Explain the procedure you followed during training.

3 Predictions and evaluation

In this section, we discuss the performance of our models and the metrics used for evaluation. We implemented two different models: a Naive Bayes classifier and a Random Forest classifier. Here, we focus on the overall evaluation approach and results, with a more detailed discussion of the Random Forest model in a subsequent section.

3.1 Evaluation Metrics

We chose two primary metrics to evaluate our models:

1. **Accuracy**: This metric provides an overall measure of correct predictions (both positive and negative) out of all predictions made.
2. **F1 Score**: This is the harmonic mean of precision and recall, providing a balanced measure of the model's performance, especially useful when dealing with imbalanced datasets.

These metrics were calculated using the scikit-learn library, ensuring standardized and reliable evaluation.

The choice of these metrics was driven by the nature of our problem: predicting stock price movements. In the context of stock trading, accurately predicting both price increases (allowing for long positions) and decreases (allowing for short positions) is equally valuable. Therefore, we needed metrics that give equal importance to true positives and true negatives.

3.2 Model Performance

3.2.1 Naive Bayes Classifier

Our initial model, the Naive Bayes classifier, achieved the an accuracy of 0.49, which might seem low at first glance, it's important to contextualize this result. Predicting stock price movements is a notoriously challenging task, and even slight improvements over random guessing (which would yield an accuracy of 0.5) can be significant in financial applications.

3.2.2 Random Forest Classifier

We also implemented a Random Forest classifier, which showed improved performance over the Naive Bayes model:

Model	Accuracy	F1 Score
Naive Bayes	0.490	0.514
Random Forest	0.562	0.636

Table 4: Comparison of Accuracy and F1 Scores between Naive Bayes and Random Forest Models

The Random Forest model's superior performance can be attributed to its ability to capture more complex relationships in the data and its robustness to overfitting.

3.3 Discussion of Results

The performance of both models, while not achieving high accuracy, provides valuable insights:

1. **Baseline Establishment**: The Naive Bayes model serves as a useful baseline, demonstrating the challenges inherent in predicting stock movements from news articles alone.
2. **Improvement with Ensemble Methods**: The better performance of the Random Forest model suggests that ensemble methods may be more suitable for this complex prediction task.
3. **Potential for Further Improvement**: Given that both models perform better than random guessing, there's potential for further improvement. This could involve feature engineering, incorporating additional data sources, or exploring more advanced machine learning techniques.
4. **Practical Implications**: Even modest improvements in prediction accuracy can be valuable in financial markets, where small edges can translate to significant advantages when applied systematically.
5. **Limitations**: The performance of both models underscores the difficulty of predicting stock movements solely based on news articles. Factors such as market sentiment, broader economic indicators, and company financials, which are not captured in our current approach, likely play significant roles in stock price movements.

3.4 Limitations of the Current Approach

Our study on predicting stock price movements using news articles has revealed several critical limitations and areas for improvement. This section discusses these limitations and proposes potential avenues for future research.

1. **Historical vs. Future Prediction**: Our model attempts to predict stock price movements over a 25-year period, which is fundamentally flawed for practical applications. In real-world scenarios, we're interested in predicting future price movements, not historical ones. This approach doesn't account for the evolving nature of markets and companies over such a long period.

2. **Binary Classification Limitation**: Our current binary classification (increase/decrease) is overly simplistic. It treats a one-cent increase the same as a significant price jump, which doesn't reflect the nuanced reality of stock market gains and losses. This approach fails to capture the magnitude of price movements, which is crucial for practical trading strategies.

3. **Low Accuracy**: The model's accuracy of 0.49 is barely better than random guessing. This performance level suggests that our current approach doesn't provide meaningful predictive power for stock price movements.

4. **Time Sensitivity**: Stock markets react to information extremely quickly, often within seconds. Our model doesn't account for the critical factor of when an article was published relative to market movements. In today's high-frequency trading environment, this timing is crucial.

5. **Competitive Disadvantage**: The stock market is highly competitive, with sophisticated players using advanced technologies and vast resources. Our simple system is unlikely to compete effectively in this environment. Large companies with superior resources can act on information much faster than our model could.

6. **Quality of News Articles**: News articles, especially in the age of AI-generated content, may not provide the depth of information needed for accurate predictions. They often lack the insider knowledge or technical understanding required for meaningful market insights.

3.5 Proposed Improvements and Future Directions

1. **Alternative Data Sources**: Rather than relying solely on news articles, future research could explore more informative data sources. For example:

    - Expert interviews and commentaries from industry insiders
    - Social media sentiment analysis
    - Company financial reports and earnings calls transcripts
    - Patent filings and technological developments

2. **Focus on Valuation**: Instead of predicting short-term price movements, a more valuable approach might be to develop models that assess whether a stock is currently under or overvalued. This aligns better with fundamental analysis principles and could provide more actionable insights.

3. **Improved Target Variable**: If continuing with a price prediction model, using a more nuanced target variable would be beneficial. For instance, predicting percentage price changes over various time horizons (e.g., 1-day, 1-week, 1-month) could provide more meaningful and actionable predictions.

4. **Incorporation of Time Factors**: Future models should account for the timing of information release relative to market movements. This could involve using time-series analysis techniques or incorporating features that capture the recency of information.

5. **Advanced NLP Techniques**: Employing more sophisticated NLP methods could help in extracting more meaningful information from textual data. This might include sentiment analysis, named entity recognition, or topic modeling to capture deeper insights from the text.

6. **Multi-modal Approach**: Combining textual analysis with other data types, such as financial metrics, market indicators, and macroeconomic data, could provide a more comprehensive view for prediction.

7. **Ethical Considerations**: Future research should also consider the ethical implications of algorithmic trading and its potential impact on market stability and fairness.

3.5.1 Fake news domain considerations

How do we make sure that we detect if there if fake news? If most articles within a time-frame say one thing, then when a low number of articles says something completely different, it might be fake. We must read the articles to find out. Also we can do this separately for each provider, it is probable that when a news provider has some fake news, it might also result in more fake news when more articles from the same provider.

- Providers that provide fake news, are more likely to have more fake news
- Providers that do not do rigorous research about a topic and have some fake news, are more likely to have other fake news as well.
- The country of origin of a provider might also change the perspective of a news writer. for example in china, where people are not free to say what they want to say. Are more likely to provide content that is in the same line of reasoning as the politics in the country. resulting in more fake news. However fake news is still news and it influences people's buying decision, thus impact the stock price.

In conclusion, while our current approach has significant limitations, it has provided valuable insights into the challenges of stock market prediction using NLP techniques. The limitations identified here serve as a roadmap for future research, pointing towards more sophisticated, multi-faceted approaches that could yield more practical and actionable results in the complex world of financial markets.

References

4 Appendix

Put your appendix figures and tables here