# ArticleAlpha – Sentiment Analysis for Stock Predictions

Cavin Gada, Yashvardhan Sharma, Ioannis Ypsilantis

Project Repository

October 9, 2023

## 1. Introduction

One of the hardest, everchanging, and open-ended problems is predicting the stock market. There are many quantitative and qualitative solutions to try and model the market, make predictions, and attempt to make money off of those predictions. A common task for stock brokers and traders is to read the news in order to gain insights into current events that will cause the stock markets to move.

Using machine learning and natural language processing to analyze stock performance is a booming field of research. In the past, several researchers and companies have invested in building sentiment models to predict stock movement. One example of said research comes from two students at Stanford who used data from Twitter tweets to predict the public mood of a stock (Mittal and Goel). They were able to predict with 75% accuracy using Self Organizing Fuzzy Neural Networks (SOFFN). Their research built upon that of a previous team (Bollen et al) who achieved an accuracy of 87%. This is just one example of many papers that have explored the relationship between stock news sentiment and price movements.

One main challenge in this line of work is market volatility. There are many factors that play into the price movements of a stock, sentiment analysis alone is not indicative enough. For example, large hedge funds, banks, and institutions move money quickly and on large scales, making their impact on the market extremely instrumental. Institutional investors and most authors for articles on stock news don't have insight into the decisions of these banks. Thus, sentiment analysis alone is sometimes not enough to predict the movement. Other factors include macroeconomic conditions such as federal interest rates, international politics, etc.

With the influx and spread of machine learning, techniques to garner insight from textual information introduce an interesting new avenue for stock predictions. Rather than having a person read the news to make predictions, can we have a machine learning model automatically read the news to make trades in the stock market? The goal of this project is to design, train, and test an ML model that takes in news articles and makes predictions on stock movement in the market.

In our project we have randomly selected 68935 samples from our selected dataset to begin preprocessing for training. The main challenge was extracting the target for our data; for our focus, we wanted to extract the stock movement due to the news. This can happen on the same day or the next day, depending on what time of day the article was published and whether the market was open. Additionally, we needed to bear in mind that we can only retrieve daily prices. Our dataset cannot capture intraday market shocks which reverted within the same day.

## 2. Problem formulation

While there are many methods and algorithms for automatic trading, in order to learn from news articles, we require a Machine learning model, specifically an NLP model. The model will take in news article titles and provide a sentiment for the given article.

## 3. Methods

For this task, we are interested in comparing BERT to finBERT. We need to consider how we will train BERT and how we will get our results for analysis. To best compare, we opted to have BERT's output match that of finBERT. To do so, we needed a three class, multi-class output.

Due to the many different factors that can play into the movement of the price of a stock, especially in a single day timeframe, we opted to focus specifically on the three-class classification task. Our three classes consisted of positive return (greater than 1% change in price), negative return (less than -1% change in price), and neutral (between -1% and 1% change in price). Setting this distinction for the class labels allowed for an equally weighted distribution of samples for each class, and thus protected the models from developing bias due to class imbalance.

In order to tune BERT for our task, we opted to add no additional layers to the model, instead, retrieving the model where the number of output labels is three. We opted to update all weights in the BERT model during training. When developing the code for training the BERT model, we referenced the CS589 HW4 stub from the Spring 2023 semester. This included the data loading and general training framework.

## 4. Datasets and Experiments

Our dataset requires news articles' titles as well as the corresponding daily return after the publishing of the article. To fulfill this requirement, we use news articles from the 'Daily Financial News' kaggle dataset, consisting of over 6000 stocks and four million articles between 2016 and 2020 (Aenlle). Specifically, we use the 'analyst_ratings_processed.csv,' which includes the relevant pertinent information (article headline, date and time of publication, and stock ticker). Of the data in this CSV, we sample a total of 67,321 rows.

Most of the data was preprocessed by the model tokenizers, however, we still needed to retrieve our returns to determine sentiment. First, we parsed the date strings into Datetime objects. Next, we acquired the corresponding sentiment values for each row in the dataset (article, ticker, and date) by retrieving financial data from yahoo finance. In this step, a key assumption is made: the movement in the price of the stock within 1 day reflects the sentiment of the news article. Bellintani and Junior in their Student Economic Review Paper concluded that "positive, negative or neutral news has a… statistically significant effect on the price of a stock at an intra-day level… [and] has almost nil effect on the volatility of the stock in the 10 days following publishing" (Bellintani and Junior). Thus, we decided to look at close prices that are a single trading day apart. Additionally, because of the described intraday assumption and that markets are closed on weekends, we look at weekday news only. The time window for this movement in the price of the stock is defined as follows:

I. If the article is published before 4:00 pm EST on date *d*, the 'before' price is the adjusted close price of the stock on the date *d*. The 'after' price is the adjusted close price of the stock on the date of the most recent trading day *d-n,* where *n >= 1* and refers to the number of days prior to *d* until a valid trading day is reached. *Note: a valid trading day is a non-holiday weekday.

II. If the article is published after 4:00 pm EST on date *d,* the ‘before’ price is the adjusted close price of the stock on the most recent trading day, *d-k,* where *k >= 1* and refers to the number of days prior to *d* until a valid trading day is reached. The ‘after’ price is the adjusted close price of the stock on the day, *d,* the article is published.

Once we have the before and after price for each stock given a date, ticker, and title, we can compute the return. The stock return is calculated by the following formula:

$$(after\ -\ before) \div before$$

The following graphs depict the distributions of returns from our train, validation, and test sets:

Figure 1.1

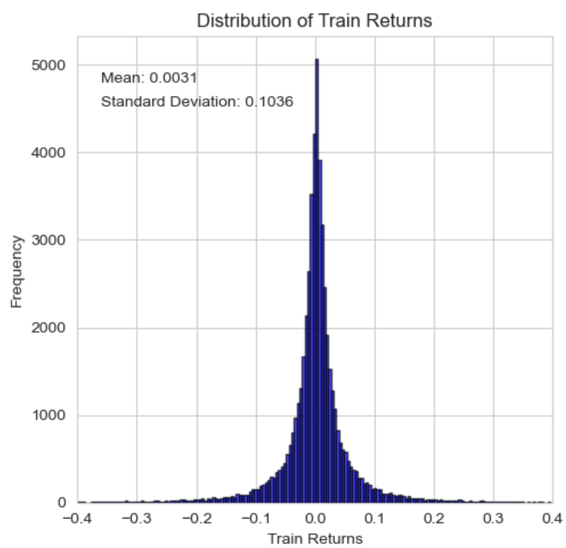



Figure 1.2

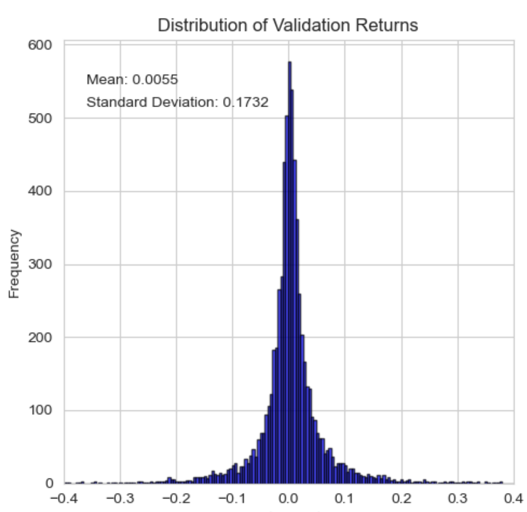
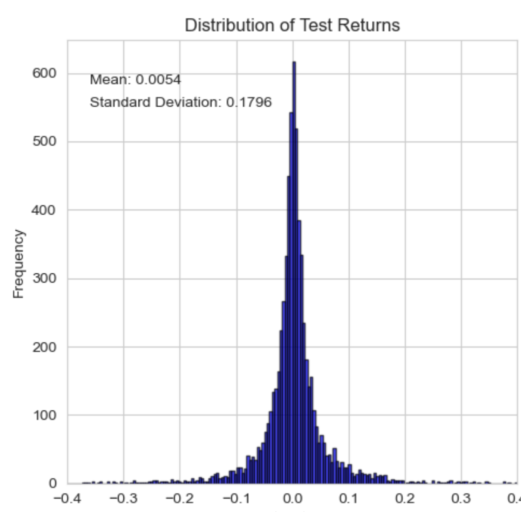



Figure 1.3



As shown in figures 1.1, 1.2, and 1.3, the three datasets' returns are very similarly distributed. More specifically, they follow a normal distribution with low variances of (.1036, .1732, and .1796). The mean return of each distribution is also very close to 0. We opted to leave the data as is and not normalize it out of concern that normalization would alter the predicted stock return. This is a moot point for the binary classification as the returned will be converted regardless.

Figure 2.0

| Number of Samples | Number of Samples | Number of Samples with Meaningful Gain | Number of Samples with Small movement | Number of Samples with Meaningful loss | Minimum Number of Tokens | Maximum Number of Tokens | Mean Number of Tokens |
|---|---|---|---|---|---|---|---|
| Train | 55148 | 19313 | 18786 | 17049 | 3 | 458 | 74.55 |
| Test | 6893 | 2432 | 2312 | 2149 | 13 | 390 | 74.48 |
| Validation | 6894 | 2419 | 2353 | 2122 | 9 | 401 | 73.31 |

As shown in Figure 2.0, for each dataset (train, test, and validation), the counts of positives and negatives are roughly even, with slightly more positive returns than negative returns. It is possible that from 2008 to 2020 there was, in general, slightly more positive growth in the stock market due to rising inflation and macroeconomic conditions.

Lastly, we analyzed the number of tokens we processed for each model. The shortest sequence had three tokens. The longests sequence had 458 tokens. The samples had an average token length of 74 tokens.

When training/testing our models, we used a train/val/test split of 80%/10%/10%. In both the BERT and finBERT testing we used the same testing set. For training BERT we used 10 epochs, a learning rate of 1e-6, for our optimizer we chose Adam, for our loss function we used Binary Cross Entropy with logits loss. We used a batch size of 32.

Upon running our models, we obtained a test accuracy of 41.24% for finBERT and a test accuracy of 49.10% for BERT. We also check the F1, Precision and Recall scores by class, pictured below:

## FinBERT Results

| | Negative | Positive | Neutral |
|---|---|---|---|
| F1 Score | 40.86% | 38.16% | 43.78% |
| Precision | 47.22% | 44.61% | 36.61% |
| Recall | 36.02% | 33.35% | 54.41% |

## BERT results

| | Negative | Positive | Neutral |
|---|---|---|---|
| F1 Score | 44.47% | 50.69% | 50.99% |
| Precision | 50.56% | 48.43% | 48.74% |
| Recall | 39.69% | 53.17% | 53.46% |

Overall, the trained BERT model outperformed the finBERT model. For the finBERT model, we see that the recall score on the neutral case is considerably higher than the recall scores of the other cases. This, paired with the fact that the precision is lower in the neutral case suggests that the finBERT model preferred to stay neutral in many cases, even when it should have picked  a side by our evaluation of sentiment. The BERT model, having been trained on our data, was a lot more balanced and was even more optimistic than the finBERT model. This fact can easily be seen by the better F1 and recall scores than the Positive and Neutral classes got in the BERT when compared to its negative results. The big takeaway here is that BERT, when tuned to take on the sentiment classification task for financial headlines, can actually perform on par or better than finBERT, a pre-trained model that was specifically designed with this task in mind. This highlights the importance of fine-tuning, even when utilizing a pre-trained model. It can aid the model in grasping the specific task and performing as best as it possibly can.

## 5. Conclusions

In this project, we have successfully compared fine-tuning a BERT model against the finBERT pre-trained model. In order to accomplish this task, we needed to figure out how to classify the sentiment for each news article headline. For that, we found using the stock return as a result of the news article to be the best indication of sentiment. By training the BERT model and comparing against the finBERT model, we found that the fine-tuned BERT model outperformed the pre-trained finBERT model. Particularly, the BERT model was more optimistic and willing to try and predict outside of the neutral case, whereas the finBERT model preferred to stay neutral and only acted on very decidable news headlines. Through our experimentation, we found the importance of fine-tuning a model to the specific requirements of the said model.

While the results were good for comparison, our models performed poorly compared to the implementation by Mittal and Goel using a Self Organized Fuzzy Neural Network (SOFNN) which achieved a 75% accuracy, or compared to the improved model developed by Bollen et al who achieved an 87% accuracy. Our poor accuracy results comparatively can be attested to only feed the article titles into our models. Having just the headline of an article is not enough to strongly determine the sentiment for a stock. Quantitative data needs to be considered in tandem with any articles.

An important consideration of our experiments is the individuality of each stock. Each stock behaves and reacts differently to market events. Many ML models on market data are fine-tuned to a specific stock rather than a holistic model for any stock. In the future, we are considering training a model using a specific stock or a basket of stocks, to see how this affects performance.

Additionally, we would be interested in implementing sentiment analysis for live data. In this feature, we could pull from both the Bloomberg terminal as well as https://finviz.com (or any other resource we can find). This website provides news articles as they come out giving us the ability to easily put articles live and make predictions. This data would be used strictly for live testing of the models. The best metric for a finance model is how much return it gains in the market.

## 6. Project Management

For this project, we are working in a team of 3. Cavin Gada, Yashvardhan Sharma, Ioannis Ypsilantis. The roles and responsibilities of each team member are as follows:

I. Cavin Gada - Dataset and Result Analysis
II. Yashvardhan Sharma - finBERT - Multi-Class (3) Testing
III. Ioannis Ypsilantis - BERT - Multi-Class (3) Training and Testing
IV. Collectively - Dataset Procurement and Preprocessing

Due to the natural overlap between these different models, everyone will be responsible for validating and ensuring the correctness of others' work.

Project timeline:

Week 1 - Dataset procurement, exploration
Week 2 - Dataset labeling
Week 3 - Dataset preprocessing, merging, and consolidation
Week 4 - BERT training
Week 5 - finBERT/BERT testing
Week 6 - Fine-tuning/(if time permits: live-testing)
Week 7 - Results Aggregation/Analysis
Week 8 - Presentation
Final Week - Final Report/Conclusion

## References


a. Mittal, Anshul, and Goel, Arpit. "Stock prediction using twitter sentiment analysis." *Stanford University, CS229 (2011 http://cs229. stanford. edu/proj2011/GoelMittal-StockMarketPredictionUsingTwitterSentimentAnalysis. pdf)* 15 (2012): 2352.

b. *Stock Market News & Blogs*, finviz.com/news.ashx. Accessed 9 Oct. 2023.

c. *[Python Project] Sentiment Analysis and Visualization of Stock News*, TheCodex, 21 July 2020, https://www.youtube.com/watch?v=o-zM8onpQZY. Accessed 9 Oct. 2023.

d. Aenlle, Miguel. "Daily Financial News for 6000+ Stocks." *Kaggle*, 4 July 2020, www.kaggle.com/datasets/miguelaenlle/massive-stock-news-analysis-db-for-nlpbacktests/data?select=analyst_ratings_processed.csv.

e. Bellintani, Luc, and Sophistor, Junior. "The Effect of News on Intra-Day Stock Prices & their Volatility." *Student Economic Review*, XXXIII, 2019, pp. 111–121.