Final Report: News Headline Informed Stock Trading

Marcs Au Peter Wang

Abstract

In this project, we investigate the predictive power of news headlines on next-day stock price movements. Our objective was to determine whether machine learning models can infer directional and percentage change in stock prices based solely on textual news data. We benchmarked a baseline linear regression model using VADER sentiment scores against more complex models, including an MLP using TF-IDF features, XGBoost, and a Longformer transformer. The MLP model consistently outperformed all others, achieving 70–74% directional accuracy compared to 55% from the baseline, 60% from XGBoost, and 58% accuracy from transformer based models.

1. Introduction

Many retail traders, day to day people who don't have the advanced algorithms large financial companies have, make their stock decisions based on their knowledge, intuition, and the news. However, much of the media can be very misleading and vague, providing unobvious hints to the stock movements, whether and how much they rise and fall.

In this project, we aim to predict how much a stock price will change after the announcement of a company-related news. Using various methods, we model the relationship between the inputs (news articles), and the output (directional change or percentage change in stock price from the time of the news release to the end of the trading day).

The baseline uses a linear regression model after condensing each article to a sentiment score. In addition to the baseline linear regression method (using VADER sentiment scores) that achieved roughly 55% directional accuracy, we have leveraged an MLP-based approach. This method converts article text into TF-IDF feature vectors and feeds them into a shallow neural network with two hidden layers. Early experiments show that the updated MLP approach achieves directional accuracies in the 70–74% range with lower mean squared errors, indicating a significant improvement. We also explored XGBoost, a gradient-boosted decision tree approach, which achieved about 60% directional accuracy. Finally, we used transformers to model the relationship and context between intraday articles, achieving 58% directional accuracy with this approach.

2. Related Work

There are several papers we've found that investigate the relationship between news headlines and change in company stock value.

The first, "Predicting stock movements based on financial news with segmentation," leveraged information from the target company and from companies in its homogeneous group, to then predict stock price movements using kernel learning techniques on text from news and headlines (Seung & Nam, 2020). Segmentation was performed by K-means clustering using financial metrics, and Multiple Kernel Learning was applied for text analysis.

Another paper, "Stock Trend Prediction Using News Headlines," uses text from news headlines to estimate market sentiment and predict whether a sector will grow, or whether a sector will cause trends in another sector. This work employs natural language processing to convert text into sentiment measures (subjectivity, objectivity, polarity) and tests various machine learning algorithms such as random forest, AdaBoost, gradient boosting, and multilayer perceptron (Kameshwari et al., 2021).

In comparison to the first paper, our project does not rely on company groupings or domain knowledge to pre-segment data. Instead, we train a generalized model across top companies using TF-IDF and learn patterns from text alone.

Like the second paper, we included sentiment-based models (baseline with VADER) and used an MLP; however, we extended this by using full TF-IDF vectors instead of collapsing text to a single sentiment score. Our results suggest that richer text representations significantly outperform sentiment-only approaches. We also went further by evaluating transformer models, something not explored in their study.

3. Datasets and Evaluation

3.1. Data Sources

For company stock time series, we use Yahoo! Finance's python library (`yfinance`), which provides bulk historical market data, including corporate actions and earnings data. For company news, we use the Finnhub API, obtaining the past year's historical news data via the Company News endpoint, which links each article to its corresponding company.

3.2. Dataset Splits and Statistics

Data was collected from April 2024 to December 2024 for the top ten companies by revenue (tickers AAPL, AMZN, COR, GOOG, JPM, MCK, MSFT, UNH, WMT, XOM). Finnhub API provided data up to a year back, and we collected up to December of 2024 to avoid large macroeconomic affects related to the new presidency. A time-aware train/dev/test split is employed to avoid look-ahead bias. In cases with limited data, cross-validation is used for hyperparameter tuning. Detailed statistics—such as the number of articles per company in each split—are maintained to ensure robust evaluation.

3.3. Evaluation Metrics

We evaluate our models using:

- **Mean Squared Error (RMSE):** Measures the average squared difference between predicted and actual percentage changes in stock price.
- **Directional Accuracy:** Measures the fraction of instances where the model correctly predicts the direction (up or down) of the stock price change. This metric is crucial because even if the magnitude is off, correctly predicting the direction can still be beneficial for traders.
- R^2 **Score:**: measures the proportion of variance in the dependent variable that can be explained by the independent variable in a regression model.

4. Methods

4.1. Baseline

The baseline model is a linear regression model. Each datapoint (a given news article) is represented by one numerical feature and one numerical target. The feature is derived by processing the article's headline and summary using VADER (Valence Aware Dictionary and sEntiment Reasoner), which outputs a sentiment score between -1 and 1. The target (Y) is defined as the percentage change in the stock price between the closing price on the publish date and the closing price at the end of the previous trading day.

Iteratively, a linear regression model would follow the procedure:

$x(0) \leftarrow 0 \in R^d$
for $t = 1, \ldots, T$ **do**
  $x(t) \leftarrow x^{(t-1)} - \eta \nabla_x F(x^{(t-1)})$
end for
return $x^{(T)}$

However, we used the analytical solution provided by Scikit-Learn:

$$w = (X^T X)^{-1} X^T y$$

This baseline, applied on data from April 2024 to December 2024 for the top ten companies (AAPL, AMZN, COR, GOOG, JPM, MCK, MSFT, UNH, WMT, XOM), achieved approximately 55% directional accuracy overall.

4.2. MLP with TF-IDF Features

The updated method transforms article text into a predicted stock price change using a shallow feedforward neural network (MLP) with TF-IDF features. The pipeline is as follows:

1. **Text Preprocessing:** Combine the `headline` and `summary` fields from each article. If one field is missing, the available field is used.
2. **Feature Extraction:** Convert the combined text into numerical features using TF-IDF (Term Frequency-Inverse Document Frequency). TF-IDF weighs each word's frequency in the article relative to its overall importance in the dataset.
3. **MLP Architecture:**
   - **Input Layer:** Receives the TF-IDF feature vector.
   - **Hidden Layers:** Two hidden layers with 64 and 32 neurons respectively, using ReLU activation functions.
   - **Output Layer:** A single neuron outputs a continuous value representing the predicted percentage change in stock price.
4. **Training:** The network is trained using backpropagation with hyperparameters (hidden layer sizes, learning rate, max iterations) tuned via cross-validation on the development set.

This approach enables the model to capture complex relationships between word importance and stock movement, yielding improved predictive performance.

4.3. XGBoost

XGBoost was another classical machine learning method we selected. XGBoost (Extreme Gradient Boosting) is a gradient-boosted decision tree ensemble model, and is widely used to analyze tabular data. In our pipeline, XGBoost is used to predict short-term stock price movements based on TF-IDF features extracted from financial news headlines and summaries.

The algorithm works by sequentially building a series of decision trees, where each tree learns to correct the residual errors of the previous ones. Specifically, we use the XGBRegressor from the XGBoost library with squared error loss (reg:squarederror) to regress future stock returns based on TF-IDF-transformed text inputs.

XGBoost, like MLPs, are good for modeling non-linear relationships, but has some unique advantages such as having fewer hyperparameters, whereas MLPs often require careful tuning. Additionally, XGBoost natively handles sparse input efficiently (like TF-IDF matrices), without the need for dense embeddings or additional preprocessing layers.

4.4. Transformers

Here, we perform a binary classification of the directional movement of stock prices. We used the Longformer transformer model due to its ability to process long input sequences. We strongly hypothesize that a Longformer-based approach will outperform our TF-IDF + MLP pipeline for several reasons.

First, the transformer produces deep contextualized embeddings that capture both syntactic structure and semantic nuances, enabling it to disambiguate word senses and model long-range dependencies across entire headlines and summaries.

Second, its self-attention mechanism can dynamically focus on the most informative phrases—such as earnings announcements, management commentary, or regulatory updates—while attenuating noisy or filler content, a flexibility that fixed TF-IDF vectors and shallow feedforward layers cannot match.

Third, by fine-tuning a pretrained language model on our financial news corpus, we inherit representations learned from massive text corpora, including knowledge of domain-specific terminology, entity relationships, and idiomatic expressions that are difficult to encode manually.

Finally, because the Longformer can ingest concatenated same-day articles as a single sequence, it has the capacity to integrate signals from multiple related news items—potentially capturing intra-day momentum or cross-article interactions—whereas the MLP treats each article in isolation. Together, these strengths suggest a transformer-based model will offer richer feature extraction and stronger generalization than our existing MLP baseline.

We propose the following pipeline:

1. **Input Construction:** We took a bit of a different approach this time, considering the fact that the same articles published in a single day may have a group effect on the change in price the following day. To allow the transformer to take advantage of the context different intraday articles provided, we concatenate the `headline` and `summary` of all such articles for a given ticker. To preserve structure, articles are separated by the special `[SEP]` token.

2. **Tokenization:** We used the allenai/longformer-base-4096 model with a classification head (LongformerForSequenceClassification) configured for binary output (num labels=2). Input texts were tokenized using LongformerTokenizer with padding and truncation up to a maximum of 2048 tokens.

3. **Training Procedure** The data was split into training and test sets with an 80/20 split. We trained the model using HuggingFace's Trainer API with the following settings:

   - Learning rate: 2e-5
   - Epochs: 5
   - Evaluation strategy: per epoch
   - Metric: Accuracy (best model saved based on highest accuracy)

5. Experiments

5.1. Baseline Experiments

The baseline linear regression model using VADER sentiment scores achieved an average directional accuracy of approximately 55% and an average MSE of around 0.000406 across the top ten companies. Additionally, the average R^2 value was -0.0701. The negative R^2 means that the model performed worse than if it had simply predicted the average y value each time. Detailed results for individual companies are presented in Table 1.

5.2. MLP Experiments

Our updated MLP model using TF-IDF features was evaluated on the same dataset. Table 4 summarizes the performance for each company.

The MLP model significantly outperforms the baseline in directional accuracy (improving from approximately 55% to about 70%) and shows lower average MSE value of 0.000178. Additionally, MLP also achieved 0.5 R^2 value,

Table 1. RMSE and accuracy of the linear regression models for the different companies (Baseline)

TICKER	TEST MSE	TEST ACC
WMT	0.000287	58.24%
AAPL	0.000241	56.88%
XOM	0.000189	44.74%
GOOG	0.000363	54.85%
JPM	0.000463	54.69%
UNH	0.000877	55.41%
COR	0.000242	42.11%
MAC	0.000692	64.10%
AMZ	0.000508	50.52%
MSFT	0.000202	55.89%

Table 2. MLP Model Results for Different Companies

TICKER	MSE	DIRECTIONAL ACCURACY
AAPL	0.000218	74.01%
GOOG	0.000233	72.12%
MSFT	0.000240	70.92%
AMZN	0.000236	72.60%
WMT	0.000240	71.83%
UNH	0.000260	70.52%
JPM	0.000249	70.23%
XOM	0.000242	71.43%
MCK	0.000289	69.22%
COR	0.000256	63.95%

a moderately good value which would basically mean the model explains about half of the variance in the stock price changes.

5.3. XGBoost

XGBoost performed slightly better than the baseline, getting about 60% average accuracy, 0.000355 average MSE, and 0.1656 R^2. The breakdown for each company is shown below.

Table 3. XGBoost Model Results for Different Companies

TICKER	MSE	DIRECTIONAL ACCURACY
AAPL	0.000233	61.97%
GOOG	0.000308	63.91%
MSFT	0.000186	57.01%
AMZN	0.000385	57.47%
WMT	0.000227	62.54%
UNH	0.000665	63.71%
JPM	0.000348	59.59%
XOM	0.000165	54.40%
MCK	0.000739	60.62%
COR	0.000290	55.56%

The accuracy was significantly lower than MLP and also had higher MSE and lower R^2.

5.4. Transformers Expiriments

Transformers that considered all articles in a day together achieved 0.58% accuracy after 5 epochs, after which the growth plateaued.

Table 4. Transformers Model Results for Different Companies

EPOCH	DIRECTIONAL ACCURACY
1	0.452229 %
2	0.547771 %
3	0.547771 %
4	0.566879 %
5	0.585987 %

6. Discussion

6.1. Baseline

The baseline model, which uses a single sentiment score from VADER and linear regression, provided a simple and intuitive method for predicting stock price changes. However, its performance (approximately 55% directional accuracy and negative R^2) suggests that sentiment alone does not capture the complete nuances of the text.

Results indicate the compound sentiment score doesn't appear to have much power for next-day stock returns. Market movements are noisy and influenced by many factors that cannot be captured well by a single number summarizing the overall tone of a news article.

6.2. MLP

Our MLP model, using TF-IDF features, showed the best performance with the accuracy, MSE, and R^2 metrics. We suggest the following future improvements:

1. Incorporate contextual embeddings (e.g., BERT) to capture richer semantic details.

2. Fuse additional features (such as sentiment scores or named entity recognition outputs) with the TF-IDF features to enhance the model's robustness.

6.3. XGBoost

Although XGBoost improved over the linear baseline by capturing some non-linear patterns in the TF-IDF space, it still trailed our MLP by a wide margin (60% vs. 70–74% directional accuracy). We believe this under-performance stems from several code-level and data-regime factors:

- **Limited feature interactions.** With 100 trees and max_depth=6, XGBoost's splits handle each TF-IDF

dimension almost independently and only learn shallow pairwise interactions, whereas the MLP's dense hidden layers can discover richer, higher-order combinations of words.

- **Overfitting to sparse noise.** Our dataset has only a few thousand examples per ticker and 500 TF-IDF features. The ensemble of small trees easily fits spurious token patterns (zeros in sparse vectors), which do not generalize, whereas the MLP benefited from weight decay and early-stopping during cross-validation.
- **Hyperparameter fragility.** We used default XGBRegressor settings (learning_rate=0.1, reg:squarederror). In practice, slight adjustments to tree depth or learning rate caused large performance swings, indicating a narrow optimal region—whereas our MLP training proved more stable under the same CV schedule.

6.4. Transformers

Our Longformer-based transformer reached only 58% directional accuracy after five epochs—well below the MLP's performance—due to several intertwined issues revealed by our code choices:

- **Truncation and input noise.** Concatenating all same-day headlines (with `[SEP]` tokens) into 2048-token inputs meant that later articles were often cut off, and mixing multiple voices introduced irrelevant content that diluted key signals.
- **Binary head vs. regression.** We fine-tuned a LongformerForSequenceClassification for up/down labels, which simplified the pipeline but threw away continuous magnitude information that our regression-oriented MLP naturally captures.
- **Data hunger and overfitting.** Fine-tuning a half-billion-parameter model on only a few thousand examples led to rapid overfitting and plateaued gains. We lacked adapter-based tuning or gradient checkpointing to enable deeper, more robust training.

To better understand its failure modes, we manually inspected misclassified examples and identified consistent patterns in the articles that proved most challenging:

- **Mixed or neutral sentiment headlines:** Articles that combined positive and negative phrases (e.g. "Revenue up but guidance cut in Q3") often led the model to hedge its prediction, resulting in incorrect up/down calls.
- **Numeric-heavy summaries:** Pieces dense with financial tables, percentages, or technical KPIs ("EPS of $1.23 vs $1.05 expected") provided little natural language context for the transformer's attention mechanism to latch onto.
- **Macro-level news:** Broad economic or sector-wide reports (e.g. Fed announcements, commodity price shifts) affected multiple tickers simultaneously, making it hard for a single-ticker sequence to isolate company-specific signals.
- **Short or headline-only entries:** Very brief headlines without a supporting summary often lacked sufficient tokens for the model to form a reliable representation, especially after the [SEP]-based concatenation with other articles.
- **Out-of-domain jargon:** Regulatory filings, merger legalese, or niche technical terms appeared in only a handful of examples, so the transformer's pretrained vocabulary and self-attention weights could not generalize to these rare contexts.

These patterns suggest that the transformer's capacity for deep contextual encoding was undermined by input noise and label ambiguity. Addressing these issues—through cleaner per-article inputs, domain-specific tokenization, or auxiliary numeric features—may help the model focus on truly predictive content rather than being misled by format or topic drift.

6.5. Additional Observations

In our testing with transformers, we considered the fact that articles are often be published in the same day. Given a short enough timeframe this would mean the market would react to these headlines together; and so as mentioned earlier, we joined with such articles together with [SEP] tokens to create single input sequences that are fed into the tokenizer and model.

However, there may also be relationships between current headlines and those published in previous days or further back. Exploring this direction while balancing with introducing unhelpful context could be another direction to potentially explore.

7. Conclusion

7.1. Baseline Conclusion

Our baseline method, using VADER sentiment scores and linear regression, served as an initial attempt and achieved roughly 55% directional accuracy. While simple and interpretable, it did not capture the full complexity of the text.

7.2. MLP Conclusion

Building on the baseline, our updated MLP model using TF-IDF features has significantly improved performance, achieving directional accuracies between 70% and 74% and lower mean squared errors. These promising results indicate that using a richer textual representation helps the model capture complex linguistic cues that influence stock price changes. In future work, we plan to explore advanced text representations and feature fusion techniques to further enhance model performance, alongside deeper error analysis and more extensive hyperparameter tuning on larger datasets.

7.3. XGBoost Conclusion

XGBoost, despite its popularity for tabular and sparse data, could not match the MLP's ability to extract nuanced sentiment signals from news text. Its reliance on shallow tree ensembles over high-dimensional TF-IDF features led to overfitting and limited interaction modeling, and its default hyperparameters proved fragile in our low-data regime. Without richer feature engineering or a substantially larger dataset, tree-based boosters will likely remain less effective than neural approaches in article-level stock return prediction.

7.4. Transformers Conclusion

While transformers promise deep contextual understanding, our Longformer experiments demonstrated that large pretrained models demand far more data and careful framing. Binary classification heads discarded valuable magnitude cues, input concatenation led to truncation noise, and fine-tuning on a small corpus quickly overfit. To leverage transformers fully, future work should adopt regression-style heads, hierarchical or sliding-window encoding for long contexts, and adapter-oriented or low-rank tuning on larger labeled sets.

7.5. Lessons Learned

We realized that "more" is not always "better" when it comes to model complexity. The modest MLP architecture delivered consistently strong results by capturing key word interactions without overfitting, whereas XGBoost's ensemble of shallow trees often chased random noise in the sparse TF-IDF space, and our transformer fine-tuning overfit rapidly on a limited article corpus. This taught us a valuable lesson: advanced architectures, such as large language models, or inappropriate architectures, such as gradient boosters, must be carefully matched to both the quantity and quality of available data, as well as to the precise problem formulation. In many real-world scenarios, a simpler, well-regularized model can outperform a heavyweight alternative simply because it is better suited to the data at hand.

References

Kameshwari, S., Kaniskaa, S., Kaushika, S., et al. Stock trend prediction using news headlines. In *IEEE Conference*, 2021.

Seung, N. and Nam, K. Predicting stock movements based on financial news with segmentation. *Expert Systems with Applications*, 2020.

A. Appendix

Github repository for code:

https://github.com/pwang1092/CSCI467_NewsHeadlineStockTrading

Instructions and details about code:

1. Data - all the news article data is organized in the finnhub news subfolder, which was fetched in chunks by the fetchNewsData.py script due to API limits.

2. Baseline, MLP, and XGBoost - can be run as is in VSCode or other IDE.

   First run command:
   "pip install -r requirements.txt"
   Then run any of the commands:
   "python [filename].py"

3. Transformers - due to local memory contraints, we ran these in Google Colab on a GPU. It is expected to take an hour or so.