Final Report: NLP-Driven Financial Sentiment Analysis and Risk Scoring

NDR Team #4: Brian Adams, Hilung Huang, Luis Tupac
NED Davis Research – Practicum, GT - Spring 2025

April 8, 2025

Abstract

This report presents a comprehensive study on predicting market volatility using Natural Language Processing (NLP) techniques applied to financial headlines and SEC filings. Conducted in collaboration with Ned Davis Research (NDR), our objective was to develop robust "Risk Scores" for the S&P 500 index and its constituents that anticipate significant stock price movements. We investigated multiple modeling strategies, including token-level sentiment scoring, VIX-based volatility modeling, and topic modeling using BERTopic. Our findings highlight the predictive power of combining token-level features with sentiment and high-risk language metrics, particularly when sufficient article volume is available. Notably, a single engineered token-focused risk feature demonstrated superior performance and stability over more complex multi-feature models. We conclude by proposing future enhancements such as segmentation-based modeling, ensemble learning, and reinforcement learning to further improve accuracy and scalability.

1 Introduction

1.1 Project Aim

This project, conducted in collaboration with **Ned Davis Research (NDR)**, explores the predictive power of aggregated sentiment indicators derived from financial data and SEC filings to identify potential market inflection points. By leveraging **Natural Language Processing (NLP)** and financial analytics, our objective is to create a comprehensive "Risk Score" for the S&P 500 index and its historical constituents.

1.2 Background and Motivation

Investment sentiment plays a crucial role in market trends. News articles and SEC filings impact market movements by emphasizing risk-laden keywords such as "bankrupt" and "lawsuit." The goal is to quantify the relationship between sentiment trends and financial health, thereby better predicting market volatility.

1.3 Project Objectives

- **Data Collection:** Collect data from the sponsor.
- **High-Risk Dictionary:** Aggregate high-risk words from news headlines and SEC filings (with emphasis on 10-K reports, particularly Items 1 and 1A).
- **Risk Score Development:** Create a Risk Score to help investors anticipate market drops.

2 Literature Review

2.1 FinBERT: Financial Sentiment Analysis with BERT

FinBERT is a language model tailored for financial text analysis, built by fine-tuning BERT on a financial corpus [?]. It leverages transformers to enable bi-directional processing of financial texts. The model is trained to understand how to talk like a trader by using a financial corpus and labeled data for financial sentiment classification. The model outperformed other models like LSTM, LSTM with ELMo, ULMFit, and BERT, with a 97% accuracy on the subset. While the model improves sentiment prediction for complex financial texts, it is not able to capture all forms of "trader speak," which is a form of manipulating neutral or negative statements to sound better than normal.
Example Statement:
This implementation is very important to the operator, since it is about to launch its Fixed to Mobile convergence service in Brazil

- **True value**: Neutral
- **Predicted**: Positive

Even with its limitations, it represents a significant advancement over general-purpose sentiment models in financial applications. This paper inspired us to combine FinBERT sentiment and embeddings into our models in an effort to extract as much information as possible from the provided articles.

2.2 Automatic Text Summarization Using Gensim Word2Vec and K-Means

Haider et al. [?] proposed an interesting method for automatic text summarization by combining Word2Vec embeddings from the Gensim library with K-Means clustering. Their approach transforms sentences into vector representations and groups semantically similar sentences to extract meaningful summaries. The authors used clustering on the embedded sentences to identify representative sentences from each cluster to summarize the input document. This method works well for single-document extractive summarization and highlights the potential of word embeddings and clustering for structuring unlabeled text.

Their work inspired our decision to use Word2Vec for financial news embeddings, though we adapted the clustering strategy to DBSCAN and HDBSCAN for better noise handling and density based grouping in our financial articles.

2.3 Investor Sentiment and Market Trends

Tetlock's research shows that negative sentiment in news articles (e.g., from the Wall Street Journal) can predict lower future stock returns and heightened market volatility. This insight is reflected in our incorporation of sentiment polarity in the Risk Score computation.

2.4 Event-Based Financial Modeling

Research by Hu and colleagues demonstrates that detecting significant financial events through NLP improves stock return predictions. This approach motivates our keyword-based method for tracking price swings by identifying events such as earnings reports, legal issues, and downgrades.

2.5 Bollinger Bands

Developed by John Bollinger, Bollinger Bands® are volatility bands placed above and below a moving average. Volatility is based on the standard deviation, which changes as volatility increases and decreases. Bollinger Bands consist of three lines: a simple moving average (SMA) line, an up-per band, and a lower band. The SMA line is the average price over a specified period of time, in our case 5 days for strategy learner and 14 days for manual learner. The upper band is two standard deviations above the SMA line, and the lower band is two standard deviations below the SMA line. [?]

2.6 Relative Strength Index (RSI)

Developed by J. Welles Wilder, the Relative Strength Index (RSI) is a momentum oscillator that measures the speed and change of price movements. RSI oscillates between zero and 100. RSI is considered overbought when above 70 and oversold when below 30. [?]

2.7 Stochastic Oscillator

Developed by George C. Lane in the late 1950s, the Stochastic Oscillator is a mo-mentum indicator that shows the location of the close relative to the high-low range over a set number of periods. [?]

3 Exploratory Data Analysis (EDA)

We created a database following the Medallion architecture pattern which consists of Bronze, Silver, and Gold data layers. All of the raw data landed as is into the bronze layer, where it was then curated into the silver layer, and finally had aggregations and joins performed to it that allowed us to efficiently use it for our downstream experimentation.

The Medallion architecture consists of three distinct data layers:

- **Bronze:** Raw, unprocessed data ingested from source systems.
- **Silver:** Cleaned, structured, and joined data that is ready for analytics.
- **Gold:** Aggregated, curated datasets optimized for modeling and experimentation.

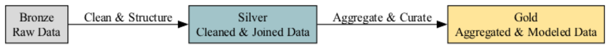


Figure 1: Medallion Architecture Diagram.

We followed this pattern to structure the data efficiently and make it analyzable data. This database consists of 3 schemas:

1. Headline Schema
2. SP500 Schema
3. Test Schema

3.1 Headline Schema

This schema held our most important data and housed key data and transformations such as:

- Ticker article data
  - These were raw XML files which we had to parse, sanitize, and link to tickers.

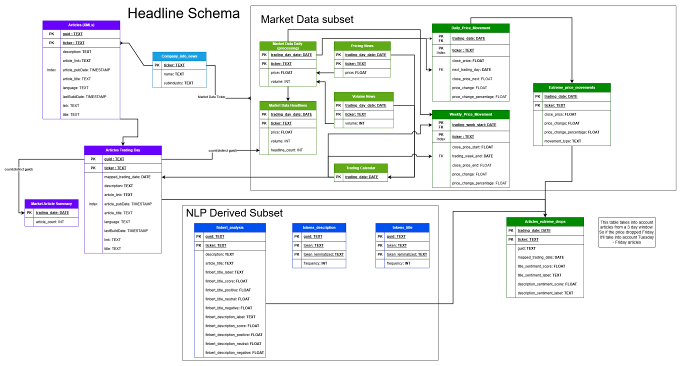


Figure 2: Headline Schema.

- Some challenges from this dataset included parsing the special characters, handling encoding errors, and aligning articles published after market close to the next available trading day. Aligning the articles was a key part of our analysis since it gave us the ability to see the impact an article had on price.

- Financial market data
  - This consisted of daily price and volume information for all the tickers provided to us. This was specially tricky since the data was stored in a matrix format (tickers as columns, dates as rows) which is visually pleasing when viewing the data, but not helpful when performing aggregations. We transformed this data into a long format structure, allowing us to perform critical operations and allowing us to infer additional data easier, like inferring the market calendar for the data we were given without relying on external datasets.

- FinBERT sentiments
  - We built a custom FinBERT pipeline to extract ***all*** sentiment scores and labels (positive, neutral, negative) instead of relying on the default return of top label and score from default pipeline.
  - This custom pipeline consisted of:
    1. Creating a distributed network with Dask (since we did not have access to GPUs).
    2. Run FinBERT in a custom inference loop.
    3. Extract logits and using Softmax to convert them into probability distributions.
  - This custom loop enriched our downstream modeling by providing us with additional information on article sentiment we didn't have before.

- Extreme price movements
  - We tracked all price movements and flagged the tickers that had a ±5% price change from previous close. This system captured the data from the past three trading days to identify factors for this kind of movement.

This schema combined with the seamless integration we received from DuckDB allowed us to quickly prototype by minimizing the time spent on data cleaning and data integration used in our notebooks.

3.2 SP500 Schema

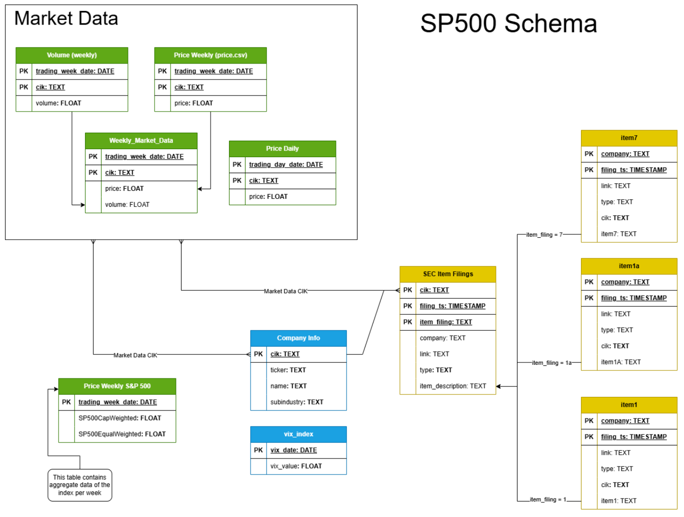


Figure 3: SP500 Schema.

This schema consisted of financial data for the S&P500 companies and SEC item filings. This schema was not the core focus of the project, it mainly served as a valuable extension to our Headline data.

The main aggregations are similar to the Headline aggregation with us:

- Flattening Volume and Price data.
- Flattening SEC filings.
- Performing FinBERT sentiment analysis to the SEC filings.

A key feature we used from this schema was the VIX index, which became an important input to many of our models.

3.3 Test Schema

The Test schema was modeled after the Headline schema and was used to hold newly provided data from our sponsor. We ran this data through the same curation pipelines as the Headline schema. This provided us with identical table structures, with a few small adjustments like removing unused tables and renaming tables.

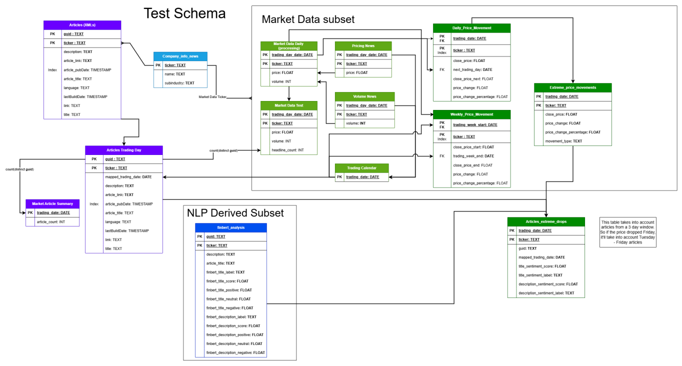


Figure 4: Test Schema.

Although this schema may look simple, having the test data available in table format made a huge difference. By loading it directly into DuckDB we were able to use its integration with Python to rapidly iterate on our models and evaluate performance on this new data with minimal adjustments to our existing training pipelines.

4 Model Survey and Methodology

Our project has evolved significantly through extensive experimentation with diverse modeling strategies. We surveyed several approaches, ranging from traditional predictive methods to sophisticated NLP-driven techniques, aiming to derive insightful risk metrics that anticipate stock price volatility. This section introduces these modeling approaches, highlighting their theoretical foundations and practical implementation considerations, leading logically into our detailed experiments and results.

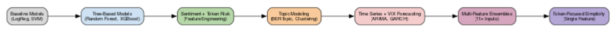

Baseline Models (LogReg, SVM)
Tree-Based Models (Random Forest, XGBoost)
Sentiment + Token Risk (Feature Engineering)
Topic Modeling (BERTopic, Clustering)
Time Series + VIX Forecasting (ARIMA, GARCH)
Multi-Feature Ensembles (11+ Inputs)
Token-Focused Simplicity (Single Feature)

Figure 5: Evolution of Modeling Complexity.

4.1 Baseline and Initial Modeling Approaches

We began our analysis with standard statistical and machine learning models to benchmark performance:

- **Logistic Regression:** Employed initially for binary classification of price movements.
- **Support Vector Machines (SVM):** Selected for effectiveness in handling text-based feature spaces.
- **Random Forest and XGBoost:** Tested for capturing complex, nonlinear interactions among features.

These models established performance baselines and guided subsequent methodological refinements.

4.2 Advanced Sentiment and Token-Level Analytics

We significantly enhanced our modeling by incorporating sentiment analysis and detailed token-level analytics, leveraging FinBERT's financial-specific transformer model:

- **Sentiment Analysis Integration (FinBERT):** Extracted nuanced sentiment scores beyond simple positive-negative labels to gain deeper predictive signals from news headlines and SEC filings.
- **Token-Level Risk Scores:** Developed features such as sentiment volatility, high-risk word variability, and token-score metrics. Experimentation identified a comprehensive "Token-Focused" risk metric that notably improved correlation with market volatility.
- **Weighted Risk Score Experiments:** Evaluated multiple weighting schemes—equal, sentiment-focused, high-risk-focused, and token-focused—to maximize predictive accuracy.

4.3 NLP: Named Entity Recognition (NER)

We also incorporate NER from spaCy library to increase our prediction accuracy. We removed all NER categories to achieve:

- **Reducing Noise in Text Classification:** Named entities (like names, dates, or locations) may add irrelevant variation and reduce our model performance and accuracy. Removing them can help the model focus on core linguistic or semantic patterns.
- **Improving Generalization in Models:** Names and specific references can overfit models to particular examples. Removing or replacing entities (e.g., with placeholders like PERSON or ORG) makes the model more robust.
- **Enhancing Keyword or Topic Extraction:** Entities like names and dates can crowd out meaningful keywords. For instance, we use weighted average in our token score/risk score. In this case, "Recession" should have a high negative value or weighted average. However, by allowing less meaningful words such as Saturday, three... We can actually reduce the score of "Recession".

4.4 Time-Series Modeling with VIX

Recognizing the importance of broader market sentiment captured by the VIX volatility index, we:

- Conducted exploratory correlations between VIX and future price movements.

- Built predictive models (ARIMA, ARMA-GARCH) to forecast VIX values, employing them as additional features for predicting stock price volatility.

These experiments validated our hypothesis regarding the delayed market impact of volatility signals, particularly under moderate VIX conditions.

4.5 Topic Modeling Approaches (BERTopic)

To uncover latent thematic drivers of market sentiment, we applied sophisticated topic modeling:

- Extracted meaningful themes from headline data using BERTopic and sentence embeddings.
- Engineered topic-based predictive features, such as historical topic sensitivity and sentiment impact, to inform our regression models.

Topic modeling proved effective, particularly for stocks with a higher volume of news articles, indicating its viability as a complementary feature engineering strategy.

4.6 Technical Indicators Analysis

To explore whether technical indicators can enhance our model's performance and predictive accuracy, we incorporated them as additional input features.

- Technical Indicators Used:
  - Bollinger Bands – captures price volatility and potential overbought/oversold conditions.
  - Relative Strength Index (RSI) – measures momentum to identify possible trend reversals.
  - Stochastic Oscillator – compares closing price to a price range over a given period to gauge momentum.
- While various methods such as rule-based systems, machine learning, and reinforcement learning are commonly employed by investors to predict stock price movement, our study focuses specifically on machine learning. This decision is due to the greater complexity and resource demands of reinforcement learning, making it less practical for our current scope

4.7 Multi-Feature and Ensemble Approaches

Finally, we explored combinations of features and ensemble techniques:

- Evaluated multi-feature models integrating diverse signals—sentiment, article volume, and topic-based metrics—to maximize predictive power.
- Tested single-feature, token-focused models for robustness and simplicity, demonstrating stable and reliable predictive performance.

These experiments underscored the delicate balance between feature richness and model complexity, emphasizing the importance of targeted feature selection and aggregation.

Our modeling survey highlights how each approach explored contributes uniquely to predictive performance. Detailed experimental findings and insights derived from each strategy are discussed in further detail in subsequent sections, systematically presenting our analytical progression.

5 Preliminary Results and Key Observations

5.1 Observations

- A surge in articles often precedes market volatility.
- Keywords such as "downgrade," "earnings," and "acquisition" frequently coincide with significant price swings.
- News clustering across multiple sources can amplify market reactions.
- Evening articles sometimes lead to notable price changes the following day.

5.2 Preliminary Modeling Results

- **Initial Modeling:** An XGBoost model using only sentiment scores on selected tech stocks resulted in a negative R-squared, possibly due to the exaggeration in headline news.
- **Improved Modeling:** By incorporating Risk Score, sentiment features (VADER and FinBERT), and article volume, models such as Random Forest and XGBoost performed notably better ($R^2 \sim 0.67$). Linear models (e.g., Ridge) underperformed, indicating the need for capturing complex interactions.

6 VIX Time Series Modeling

The goal of this approach was to explore the predictive value of the VIX index when combined with sentiment analysis for predicting price movement. Our hypothesis was that elevated VIX values could amplify the effect of negative news articles on stock prices. This approach made a key assumption which was sentiment, specially negative sentiment, had an impact in price. This is loosely shown to be true in table 1. The flow is as follows:

1. Find correlation between VIX and price movements.

2. Fit the VIX time series and test for stationarity.

3. Train ARIMA and ARMA-GARCH models on the data and compare performance.

4. Perform rolling forecasts on test data.

5. Use predicted VIX values alongside FinBERT sentiment scores to classify price movements.

6.1 Exploratory Correlation: VIX vs. Price Movements

Before modeling the VIX index, we first explored whether VIX values had a meaningful relationship with stock price volatility. Using our **extreme_price_movements** table, we were able to extract price movements and compute the correlation metrics between historical VIX values and extreme price changes over different future windows.

As seen in table 1, we ran the correlation experiment with various future time windows, VIX thresholds, and even included the sentiments in the experiments. This table contains the more promising results.

Window	**VIX >**	**Sentiment Filter**	**Spearman Value**	**p-value**
7	25	None	0.0039	0.6431
7	25	Negative	-0.0989	0.1383
7	15	Negative	0.1692	0.1692
14	25	None	0.0513	3.47e-09
14	15	None	0.0234	1.24e-05
14	15	Negative	0.0993	9.58e-09
14	25	Negative	-0.4430	2.77e-04

Table 1: Spearman Correlation Between VIX and Future Price Movements.

Key insights:

- Longer lag windows (14-day) showed consistent correlations even without a sentiment filter, suggesting delayed market responses.

- Filtering by **negative sentiment** increased the strength of the correlation.

- The strongest correlation was seen with VIX > 15 and a 7-day future window.

- Surprisingly, extremely high VIX values (> 25) had lower or negative correlations. We believe this means that fear from negative news may already be priced into the market.

To visualize this delayed effect, Figure 6 shows the count of extreme price surges and drops after VIX spikes with a value between $15 - 25$. These will be called Moderate VIX values moving forward.

These observations showed promising results, so we decided to move forward with VIX modeling and use these future VIX values as a feature of the overall risk score.

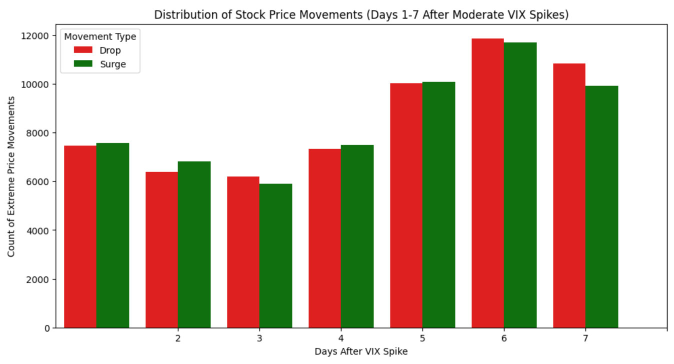


Figure 6: Distribution of Stock Price Movements (Days 1–7 After Moderate VIX Spikes).

6.2 Time Series Modeling

Data Preparation and Aggregation

We split the VIX dataset into 70% training, 15% validation, and 15% test and modeled the following VIX aggregations:

- Daily VIX values
- Weekly Mean (average of the week)
- Weekly Close (last trading day of the week)

The VIX dataset was daily from the start so we decided to use it as is. We were concerned from the extra noise we'd get by using the data at such a granular level, so we added 2 additional aggregations, Weekly Mean and Weekly Close. With this we began the time series modeling.

Stationarity Testing

Before we're able to model we have to ensure the data is stationary. This involved visual inspection of the time series and ACF/PACF plots as well as performing the Augmented Dickey-Fuller (ADF) test for stationarity.

All of the datasets show a downward trend and seasonality so we differenced the data and ensured they passed the ADF test for stationarity.

ARIMA Modeling Results

Once the data was stationary, we moved on towards capturing the model with ARIMA. We iterated over different p, d, and q values, choosing the ones that minimized AICc. We eventually landed on the values seen in table 2.

When it came to predictions, we received underwhelming results form this. Table 3 has the results.

The Weekly Mean performed best on all the metrics except for the Precision Error (PM). We believe this is because we smoothed out the data with the Weekly Mean aggregation. Performing an LJung-Box Squared

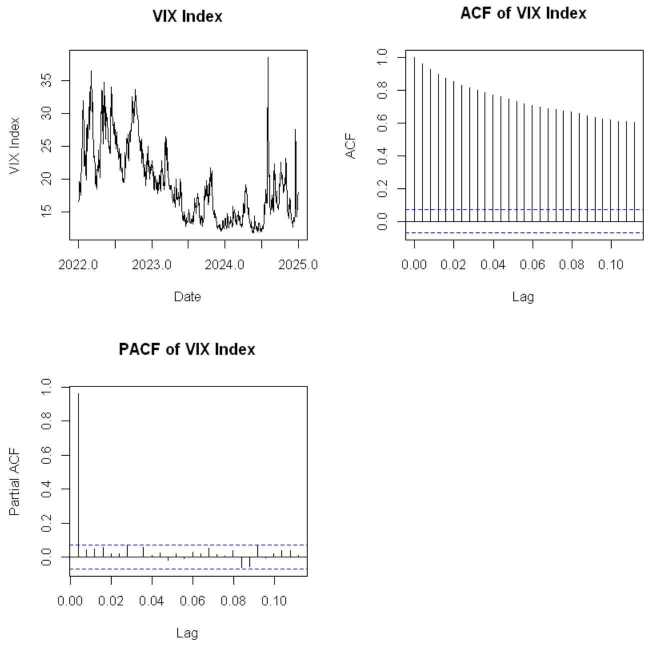


Figure 7: VIX Daily Close graphs.

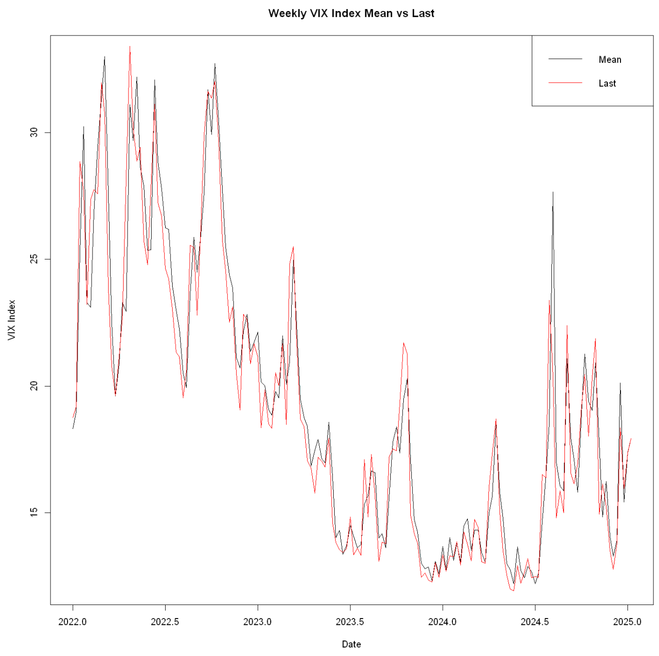


Figure 8: VIX Weekly Mean and Weekly Close (shown as Last here) graphs overlaid.

Dataset	**p**	**q**	**d**
Weekly Mean	3	5	1
Weekly Close	4	3	1
Daily VIX	5	4	1

Table 2: Best p, q, and d values for ARIMA(p,d,q).

Model	**MAPE**	**PM**	**RMSE**	**MAE**
Weekly Mean	8.59	1.05	1.49	1.23
Weekly Close	9.45	0.99	1.76	1.38
Daily VIX	9.35	2.30	3.55	2.81

Table 3: ARIMA prediction metrics.

test showed that there was volatility in our data, so using a GARCH model to capture it would be helpful.

ARMA-GARCH Modeling

To find the best model efficiently and without large amounts of nested iterations, we used a heuristic approach to find the best p, q, m, and n for ARMA(p,q) GARCH(m,n). The approach is as follows:

1. Optimize p, q for ARMA.
2. Tune m, n for GARCH using fixed ARMA.
3. Re-tune ARMA based on best GARCH.
4. Select best model by lowest BIC.

Best model configurations are shown in table 4. Interestingly enough, the "best" GARCH model for the Daily VIX value didn't involve any GARCH values. Similarly, the best GARCH model for Weekly Mean was a GARCH only model. The only model that used both ARMA and GARCH was the Weekly Close. This is interesting because we assumed that if any of the models wouldn't have a GARCH value, it'd be the smoothened out Weekly Mean model.

Dataset	**p**	**q**	**m**	**n**
Weekly Mean	0	0	1	1
Weekly Close	2	2	0	1
Daily VIX	5	4	0	0

Table 4: GARCH configurations.

The resulting performance is seen in table 5. Both Weekly Close and Daily VIX performed much better than before, while Weekly Mean performed much much worse than its ARIMA counterpart.

Model	**MAPE**	**PM**	**RMSE**	**MAE**
Weekly Mean	16.50	4.38	3.04	2.33
Weekly Close	8.41	0.75	1.54	1.20
Daily VIX	7.82	1.96	3.28	2.37

Table 5: GARCH metrics.

6.3 Classifying Price Movements

Following our VIX time series modeling, we used the predicted VIX values $(t + 7)$ to classify future price movements into multiple categories. We experimented with three approaches using XGBoost classifiers, each with different threshold definitions for price fluctuations.

3-Class Model

Our initial approach was a 3-class model that classified the data into the following 3 targets:

- -1: Drop (price decreased by 2.5% or more)
- 0: No significant change (between -2.5% and 2.5%)
- 1: Surge (price increased by 2.5% or more)

Since we observed a delayed effect from the VIX, we used the lagged price and volume features over a 7-day window. The features used were:
`ticker, subindustry, VIX_t, VIX_t-7, VIX_t+7**, positive/neutral/negative sentiment scores, top sentiment label, lagged price, price % change, lagged volume, and volume % change`

**We trained the model using observed VIX at $t+7$ during training, but during prediction, we substituted this with our predicted VIX value.

The model performed *a little too well* with an accuracy of 100%.

label	precision	recall	f1-score
-1	0.98	0.99	0.99
0	1.00	1.00	1.00
1	0.98	1.00	0.99

Table 6: 3 class classifier metrics.

We noticed that there was a huge class imbalance. The distribution is shown in table 7.

label	distribution
0	169,018
1	16,569
-1	15,480

Table 7: 3 class classifier distribution.

Despite the strong metrics, the model overwhelmingly predicted class 0. To address this, we introduced more granular classes.

5-Class Model

We adjusted our thresholds to define 5 distinct classes:

This balanced the class distribution some more but didn't get rid of the giant cluster at "No Change" label 0.

To compensate for this imbalance, we added class weights to the XGBoost model. This improved the performance and resulted in predictions that were better spread across classes as seen in table 10. Our accuracy dropped a bit from the previous model to 90%.

Class	% price movement	Description
-2	$\leq -5.0\%$	Huge Drop
-1	$> -5.0\%$ and $\leq -2.5\%$	Moderate Drop
0	$> -2.5\%$ and $\leq 2.5\%$	No Change
1	$> 2.5\%$ and $\leq 5.0\%$	Moderate Surge
2	$> 5.0\%$	Major Surge

Table 8: 5 class classifier labels.

label	distribution
0	165,338
1	12,500
-1	11,285
-2	4,066
2	3,837

Table 9: 5 class classifier labels.

label	precision	recall	f1-score
-2	0.89	0.48	0.63
-1	0.55	0.09	0.15
0	0.91	0.99	0.95
1	0.74	0.31	0.44
2	0.82	0.34	0.48

Table 10: 5 class classifier metrics. ***Note:*** While it is not shown, we shifted all labels by +2 to fit XGBoost's label constraints (i.e., no negative class labels).

The most important feature based on feature importance was the predicted VIX at $t+7$, further reinforcing its predictive value.

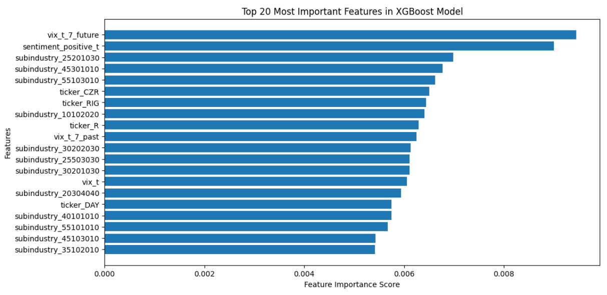


Figure 9: Top 20 features ordered by importance in XGBoost.

7-Class Model

To further split the dominant "no change" bucket, we tried a 7-class setup:

This produced a more balanced distribution as seen in table 12

Performance:

While the overall accuracy dropped to 46%, we were able to capture more price movement variation and break up the large "No Change" cluster. While we retained a high precision and recall for the extreme cases (labels -2 and 4), the model struggled more with subtler price

Class	% price movement	Description
-2	$\leq -5.0\%$	Huge Drop
-1	$> -5.0\%$ to $\leq -2.5\%$	Moderate Drop
0	$> -2.5\%$ to $\leq -0.5\%$	Minor Drop
1	$> -0.5\%$ to $\leq 0.5\%$	No Change
2	$> 0.5\%$ to $\leq 2.5\%$	Minor Surge
3	$> 2.5\%$ to $\leq 5.0\%$	Moderate Surge
4	$> 5.0\%$	Major Surge

Table 11: 7 class classifier labels. No change is now label 1 in this setup.

label	distribution
1	69,672
2	52,050
0	43,616
3	12,500
-1	11,285
-2	4,066
4	3,837

Table 12: 7 class classifier labels.

label	precision	recall	f1-score
-2	0.87	0.52	0.65
-1	0.67	0.20	0.30
0	0.56	0.22	0.32
1	0.41	0.69	0.51
2	0.48	0.46	0.47
3	0.72	0.40	0.51
4	0.83	0.36	0.50

Table 13: 7 class classifier metrics. As before we applied a +2 shift to the labels which is not shown in the above table.

movements as seen in the lower precision and recall scores for those intermediate labels.

This highlights the benefits of class weighting with underrepresented classes. It was less effective for the middle categories where they had more data but where still overshadowed by the "No Change" cluster.

Conclusion

Across all models, class imbalance proved to be the primary challenge. The 3-class model had high scores but was overly biased toward the dominant "No Change" class. The 5-class model struck a better balance and showed promise, especially with class weighting. The 7-class model, while less accurate, offered more realistic and nuanced results by capturing subtler movements.

Unfortunately, none of the aforementioned models brought much confidence due to the data imbalance that we had. They'd be great at predicting "No Change" signals, which isn't the goal of this project.

We also experimented with logistic regression and decision trees, but none outperformed XGBoost across these multi-class setups. Based on feature importance, the predicted VIX value at $t + 7$ was consistently the most significant input in all models.

7 Thematic Clustering

To explore alternative approaches to labeling sentiment in financial news, we developed an unsupervised clustering pipeline using Word2Vec embeddings and the Loughran-McDonald (LM) financial lexicon. This approach was inspired by the text summarization approach done by Haider et al. [?] Due to the short text information from the articles and complex relationships within the market, we decided to modify their approach to use density based clustering instead of K-means. Additionally we used a combination of metrics to find the best Word2Vec paramaters that maximized the vector information. This allowed us to rely on unseen relationships between the articles to label the thematic category of article clusters using the LM lexicon.

Embedding Financial News Articles

We first cleaned and tokenized article titles and descriptions by removing punctuation, stopwords, and irrelevant tokens. A custom Word2Vec Skip-Gram model was then trained on the cleaned corpus. To validate the quality of our embeddings, we compared them against the WordSim-353 benchmark using Spearman correlation, as discussed by Kliegr and Zamazal [?], and achieved a coefficient of 0.2568 with:

- `vector_size` = 400
- `window` = 4

We then averaged the word embeddings in each article to obtain a document vector representing its semantic content.

Clustering with DBSCAN and HDBSCAN

We applied two density-based clustering methods: DBSCAN and HDBSCAN, both of which are well-suited for discovering non-spherical clusters and identifying noise.

DBSCAN: We tuned DBSCAN's eps and min samples parameters using Silhouette Score, Davies-Bouldin Score, cluster count, and noise ratio. The best configuration was:

- `eps` = 1.2
- `min_samples` = 50
- Result: 19 clusters (including noise)

Unfortunately, we observed a large class imbalance, with one dominant cluster as seen in figure 11a. We applied sub-clustering to the largest group using eps = 1.0 and min samples = 37, which resulted in 46 additional subclusters as seen in figure 11b. However, this recursive

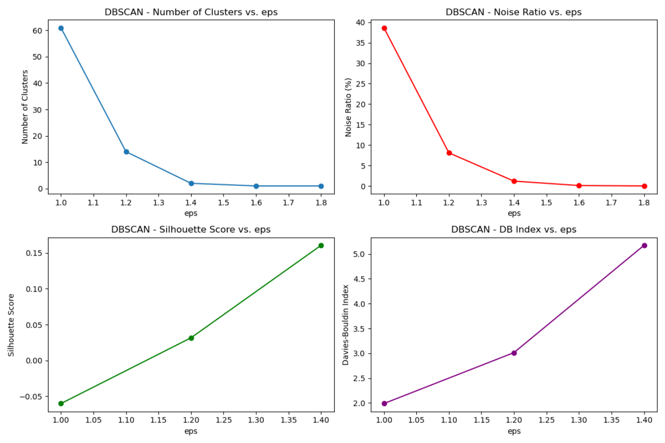


Figure 10: DBSCAN tuning metrics.

process still failed to fully resolve imbalance. To prevent doing this recursively until we finally break up the dominant cluster, we decided to see what information we can get from just one level of clustering.

HDBSCAN: We faced many challenges trying to implement this clustering algorithm due to its heavy CPU limitations. HDBSCAN was initially abandoned but was later revisited using a GPU-accelerated version via NVIDIA's cuML library[**?**].

This approach yielded 4 main clusters (including noise). Similar to DBSCAN, we achieved one extremely dominant cluster and several smaller ones. Further sub-clustering produced 9 total clusters but did not fully break up the dominant cluster as seen in figure 11c.

Labeling Clusters with Risk Sentiment

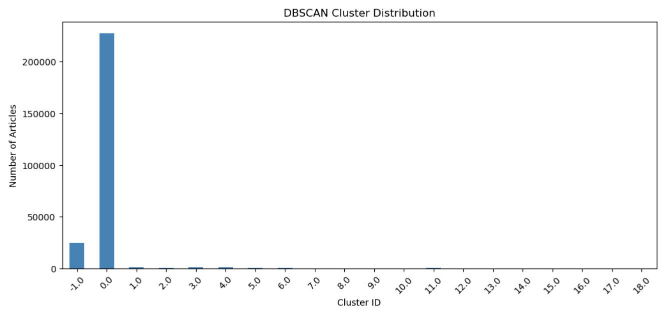
To assign a thematic sentiment to each cluster, we used the LM financial lexicon to tag clusters with sentiment categories:

- Positive
- Negative
- Litigious
- Uncertain
- Mixed Sentiment (default when no category dominated)

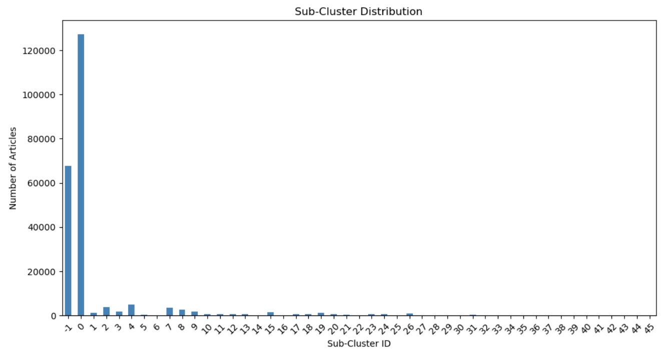
Each cluster was labeled based on the proportion of its vocabulary overlapping with lexicon categories. We used a multi-labeling approach to capture mixed risk types. Below are examples of labeled clusters from each algorithm:

DBSCAN Cluster Tags (excerpt):

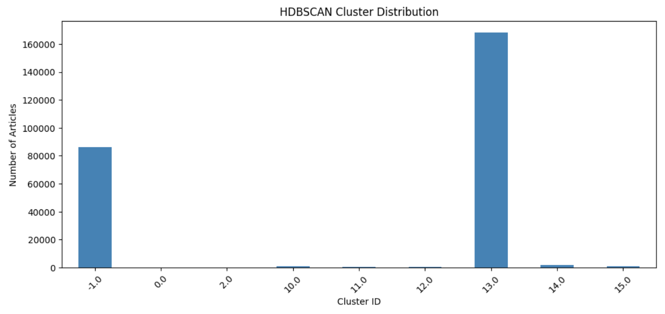
- Cluster 0.0: [Negative, Positive]
- Cluster 4.0: [Negative, Positive, Uncertainty]
- Cluster 6.0: [Negative, Uncertainty]



(a) DBSCAN after clustering.



(b) DBSCAN after sub-clustering.



(c) HDBSCAN after sub-clustering.

Figure 11: DBSCAN and HDBSCAN cluster distribution. -1 denotes the noise cluster.

- Cluster 12.0: [Negative, Uncertainty]
- Cluster 15.0: [Positive]

HDBSCAN Cluster Tags (excerpt):

- Cluster -1.0: [Negative, Positive]
- Cluster 2.0: [Negative, Uncertainty]
- Cluster 13.0: [Negative, Positive]
- Cluster 15.0: [Negative, Positive, Uncertainty]

The results were interesting, as some clusters had both negative and positive sentiment. This could mean that the articles in these clusters had both positive and negative news. We know articles could talk about 1 or many tickers at once, so there could be a mix where a ticker has good news but another has negative news within the same article.

Comparing DBSCAN vs. HDBSCAN Labeling

Although clustering was not usable for predictive modeling due to the severe imbalance, we decided to compare the results from both of these attempts.

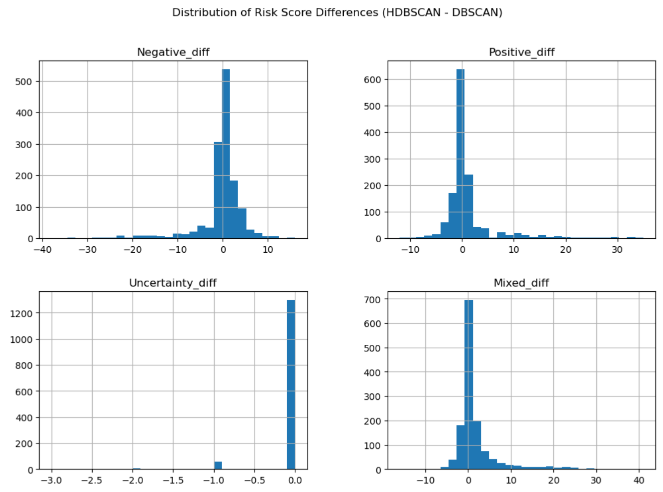


Figure 12: HDBSCAN vs DBSCAN.

Key Observations:

- **Negative Sentiment:** DBSCAN assigned more distinct negative clusters shown by the long negative tail. HDBSCAN was more conservative.
- **Uncertainty:** Both methods had nearly identical Uncertainty scores, suggesting consistent identification of uncertain content.
- **Positive Sentiment:** DBSCAN assigned positive tags to more tickers overall, while HDBSCAN had fewer but more distinct positive clusters shown by the long positive tail.
- **Mixed Sentiment:** Both models agreed on mixed sentiment tagging for ambiguous clusters, with HDBSCAN being slightly more liberal in assigning this category.

7.0.1 Clustering with FinBERT Embeddings

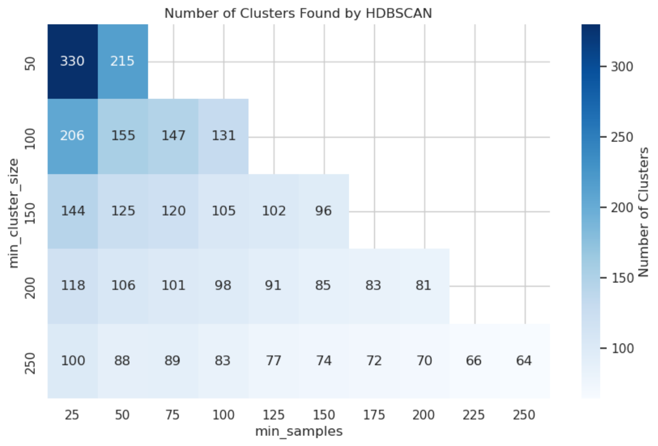
Since the results from our Word2Vec-based clustering (via DBSCAN and HDBSCAN) were somewhat underwhelming, we decided to explore a transformer-based embedding model, FinBERT. We hoped this switch would allow us to capture more contextualized representations of the financial text since this model is pre-trained on a financial corpus.

Compared to Word2Vec, the FinBERT approach was more hands-off. The model handled tokenization, sanitation, and embedding internally, making the pipeline much simpler. For clustering, we used HDBSCAN, which we found to be more effective than DBSCAN for handling variable-density clusters and separating noise.[**?**]

We performed a parameter grid search and identified the best configuration as:

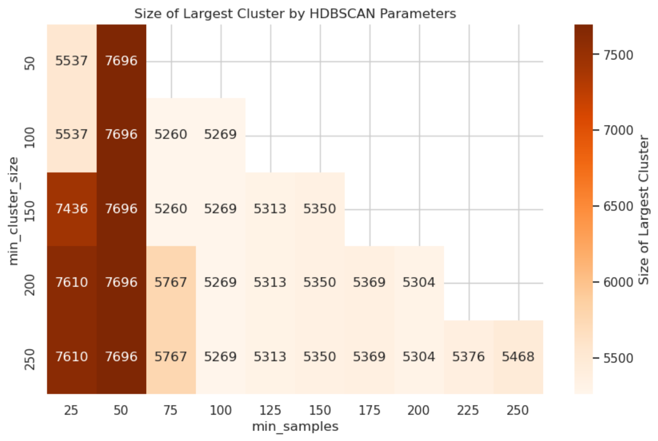
- `min_cluster_size` = 150
- `min_samples` = 100

This configuration minimized cluster size while maximizing cluster count, as shown in figure 13.



min_cluster_size \ min_samples	25	50	75	100	125	150	175	200	225	250
50	330	215								
100	206	155	147	131						
150	144	125	120	105	102	96				
200	118	106	101	98	91	85	83	81		
250	100	88	89	83	77	74	72	70	66	64

(a) Number of FinBERT clusters using HDBSCAN.



min_cluster_size \ min_samples	25	50	75	100	125	150	175	200	225	250
50	5537	7696								
100	5537	7696	5260	5269						
150	7436	7696	5260	5269	5313	5350				
200	7610	7696	5767	5269	5313	5350	5369	5304		
250	7610	7696	5767	5269	5313	5350	5369	5304	5376	5468

(b) Size of largest FinBERT clusters using HDBSCAN.

Figure 13: HDBSCAN results from grid search hyper parameter tuning.

We then ran the resulting clusters through the same multi-label risk tagging pipeline using the LM lexicon. Surprisingly, the results were the same across all clusters, with the same two labels: [**Negative, Positive**].

FinBERT HDBSCAN Cluster Tags (excerpt):

- Cluster 0: [Negative, Positive]
- Cluster 1: [Negative, Positive]
- Cluster 2: [Negative, Positive]
- Cluster 9: [Negative, Positive]

This result was somewhat ironic, given that FinBERT itself is designed to classify text as either negative, positive, or neutral. We're not sure if this outcome is a byproduct of the transformer architecture or a result of the clustering process. Ultimately this approach did not yield the insights we hoped for because the clusters did not offer meaningful differentiation in terms of risk sentiment.

Conclusion

This clustering pipeline helped uncover thematic risk patterns and offered a new way to use article data past the positive, negative, and neutral tagging done by FinBERT. Despite the class imbalance challenges, the labeling approach provided insight into common themes shared by the articles, proving this could be a reasonable approach at exploring hidden relationships in the complex landscape of financial texts. Unfortunately this was not stable enough to use for the overall risk score, but it was a fun way to explore sentiment analysis.

8 Token Word Score and Sentiment Analysis Approach

This approach leverages token-level analysis to compute a detailed risk metric from financial news articles. Our focus on sentiment—especially negative tone—and its relationship to volatility is aligned with prior findings in the literature showing that negative language in financial news can forecast lower future returns and increased market volatility [?].

8.0.1 Risk Score Computations

Three distinct risk scores were calculated for each ticker:

1. **Basic Sentiment Risk:** The volatility (standard deviation) of sentiment scores from FinBERT's title and description analyses.
2. **Enhanced Sentiment Risk:** Incorporates additional sentiment features, including positive, neutral, and negative scores from both titles and descriptions.
3. **Comprehensive Risk:** Combines sentiment volatility, high-risk word count variability, and token score variability into a single measure.

Correlation analysis (Figure 14) and regression results (Figure 15) highlight that the Comprehensive Risk measure has the strongest relationship with stock volatility (correlation: 0.58, R^2: 0.33), significantly outperforming the Basic (R^2: 0.06) and Enhanced Sentiment Risk (R^2: 0.02) measures. This indicates that combining sentiment, high-risk words, and token-level variability offers greater predictive insight into volatility.

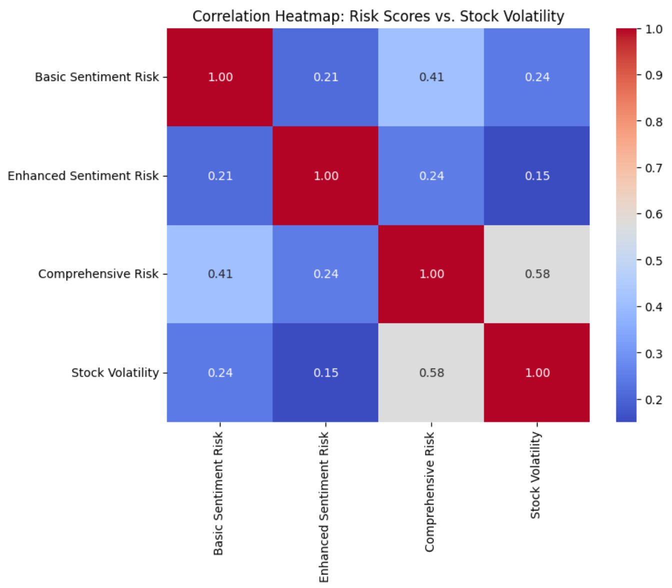


Figure 14: Correlation Heatmap of Risk Scores and Stock Volatility.

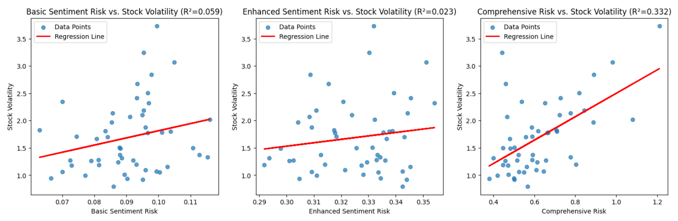


Figure 15: Regression Analysis of Risk Scores versus Stock Volatility.

8.0.2 Weighted Risk Score Experiments

To identify the most effective weighting strategy, we experimented with multiple schemes:

- **Equal Weights:** Equal contribution of sentiment volatility, high-risk words, and token variability.
- **Sentiment-Focused:** Greater emphasis on sentiment volatility (60%).
- **High-Risk-Focused:** Greater emphasis on high-risk word variability (60%).
- **Token-Focused:** Greater emphasis on token-level variability (60%).

The correlation heatmap (Figure 16) indicates that the Token-Focused Risk strongly correlates with stock volatility (correlation: 0.65), outperforming other weighting schemes. This suggests that token-level granularity captures substantial volatility signals.

8.0.3 Key Insights

The analysis underscores several important findings:

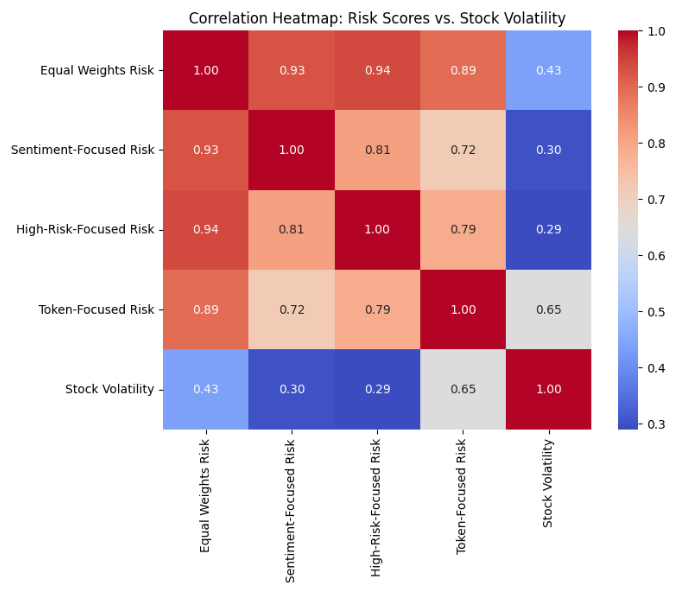


Figure 16: Correlation Heatmap of Weighted Risk Scores and Stock Volatility.

- Combining multiple sentiment and token-based features significantly improves volatility prediction compared to using sentiment alone.
- Token-focused approaches show the greatest promise, emphasizing the critical role of detailed textual analysis at the token level.
- Weighted combinations of risk measures provide additional insights, offering opportunities for customized volatility modeling.

8.1 Multi-Feature vs. Token-Focused Models

8.1.1 Multi-Feature Model (11 Features Including `article_count`)

This model utilized a comprehensive set of features, including sentiment scores (from FinBERT), token-based indicators (e.g., token-level sentiment scoring and high-risk word counts), and metadata such as the number of articles per ticker. While the inclusion of diverse features aimed to improve generalization, the model's performance varied widely across tickers.

Important Note on Interpretation: While the R^2 metric is theoretically bounded above by 1.0, our analysis observed several ticker-specific models producing R^2 values exceeding this limit. These inflated values stem from a combination of factors:

- Extremely small test set sizes for certain tickers, leading to unstable variance estimates.
- Outliers in price change percentages skewing performance metrics.
- Model overfitting on sparse or noisy feature combinations.
- **Median R^2 Scores (across 50 tickers):**
  - **XGBoost:** 1.06
  - **Linear Regression:** 1.07
  - **Random Forest and Neural Network:** Generally lower, often negative or highly variable.

Additional Observations:

- The mean R^2 scores were highly skewed by outlier values, reaffirming the use of the **median** as a more stable indicator of overall model performance.
- Despite the richness of the feature set, models often overfit—especially when article volume was low or sentiment features lacked variability.

These findings motivated us to streamline our modeling pipeline. In the following section, we explore a simplified, token-focused feature—derived from sentiment scores, high-risk word counts, and token scoring—that achieved more stable and generalizable results with lower variance in both R^2 and MAE.

8.1.2 Token-Focused Model (Single Feature: `token_focused_risk`)

This model was streamlined, employing a single engineered feature that aggregated sentiment scores, high-risk word counts, and token-level risk metrics:

- **Median R^2 Scores:**
  - **XGBoost:** 0.75
  - **Linear Regression:** 0.70
  - **Neural Network and Random Forest:** Also showed consistent and stable performance.
- **Observations:**
  - Demonstrated greater stability and fewer outliers.
  - Proved that a well-designed single feature can deliver robust predictive performance, reducing the risk of overfitting.

8.2 Impact of Article Count on Model Performance

Implementing a minimum article threshold (100 articles per ticker) significantly reduced noise and enhanced reliability of performance metrics:

- Including `article_count` explicitly as a feature further improved explanatory power.
- These findings were consistent with prior topic-modeling experiments, emphasizing the critical role of adequate article volume.

8.3 Key Insights and Recommendations

1. **Simplicity enhances model robustness:** The token-focused risk feature alone effectively captured significant predictive insights.

2. **Median metrics provide more stable information:** Median values were more indicative of true model performance compared to mean values due to outlier influence.

3. **Article volume is essential:** Providing enough data per ticker improved both the accuracy and reliability of the model.

4. **Model sensitivity to noise:** Random Forests and Neural Networks exhibited high sensitivity to noisy data without hyperparameter tuning, indicating a need for cautious selection or further optimization.

9 Alternative Risk Score Method with Named Entity Recognition and Technical Analysis

This method focuses on a weighted score with NER reduced token words. We also studied the impact on the performance of the model by adding additional features such as VIX and technical indicators.

9.1 Risk Score Calculation Steps:

1. **Tokenization:** Articles are cleaned, tokenized, and lemmatized. Stopwords and non-informative tokens are removed.

2. **NER removal:** Run spaCy's named entity recognition function to remove all NER from tokenized words.

3. **Final Set:** find set of all tokenized words(NER removed) to remove duplicate words.

4. **Calculate risk score for each tokenized word:**

   (a) **Iterate thru each trading day and add weighted score to each word in the final set.**

   (b) **The weighted score for each trading day for a particular word = total appearance of that word/total word count * tomorrow's stock increase percentage.**

   (c) **risk score for particular word = total weighted score sum / total trading days.**

We ran two kinds of prediction: regression and classification and we use multiple metrics to measure and compare model performance:

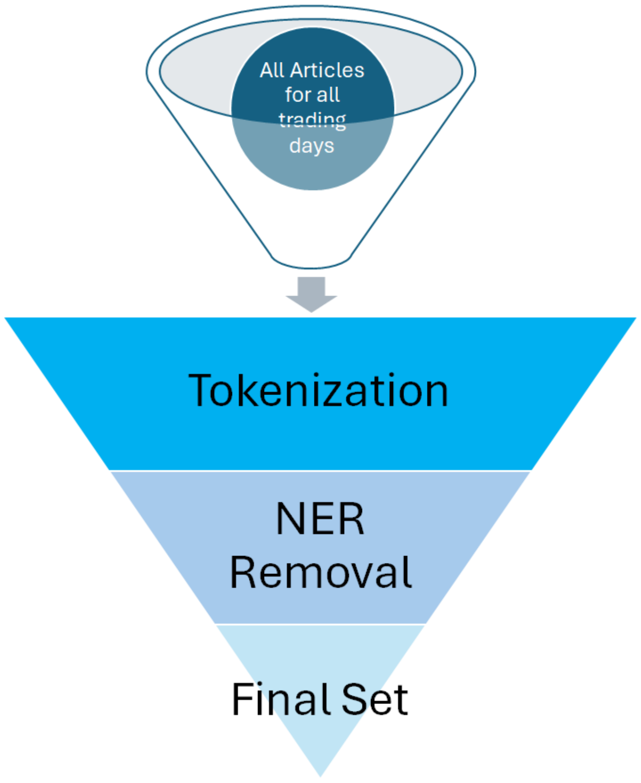


Figure 17: Tokenization and Reduction.

1. **Regression:** Measure - mean absolute error and R-square.

2. **Classification:** Measure - Precision, Recall and F1 score.

We chose these classification measures because the positive identification of the stock price drop is important to us.

Because we have done comparison of different regression methods in other part of this report, we are only using XGboost to simplify the comparison process.

9.2 Regression Analysis and Comparison:

We started our initial analysis with multiple individual features, most of which provide a negative or low R-square value. Risk Score feature is the only one that shows a good R-square result. We then try adding other features to see if the performance can improve. We found a combination of Vix and Risk Score significantly improves R-square value. This is clearly illustrated in the Vix and Risk Score Interaction table. This improved performance could be due to the non-linear effects that emerge when both are present. In our case, Vix adds context to Risk Score. Keep in mind, our XGboost method automatically captures interaction and does not require additional interaction feature.

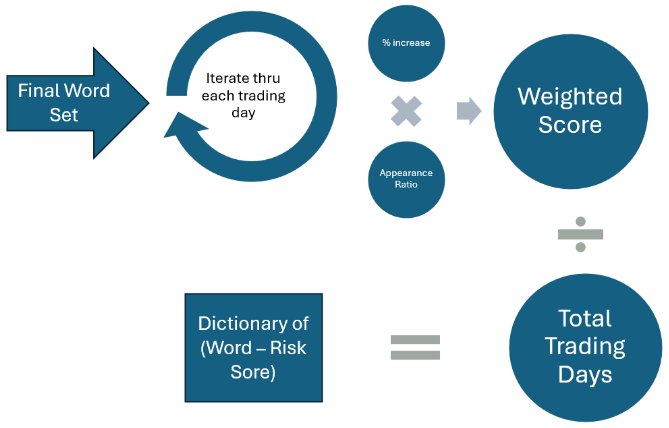
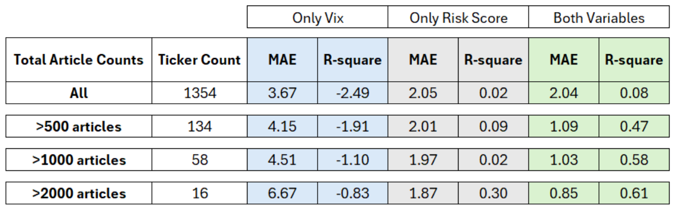


Figure 18: Alternative Risk Score Calculation.

Similarly to the previous risk score method, we also noticed the impact of the count of articles. As the total number of articles increased, the performance of our model increased dramatically. As we observed on the Vix and Risk Score fig 19, with both variables features, MAE decreased from 2.04 to 0.85 as we increase the number of article. In addition, the R-square value increased from 0.08 to 0.61. This also makes sense, as more data generally means:

1. Less noise.
2. Variance is more stable.
3. Model can detect patterns more reliably.
4. Reduction in overfitting effects. A small sample might not capture the true variability.



		Only Vix		Only Risk Score		Both Variables	
Total Article Counts	**Ticker Count**	**MAE**	**R-square**	**MAE**	**R-square**	**MAE**	**R-square**
All	1354	3.67	-2.49	2.05	0.02	2.04	0.08
>500 articles	134	4.15	-1.91	2.01	0.09	1.09	0.47
>1000 articles	58	4.51	-1.10	1.97	0.02	1.03	0.58
>2000 articles	16	6.67	-0.83	1.87	0.30	0.85	0.61

Figure 19: Vix and Risk Score Interaction.

9.3 Classification Categorization:

While regression prediction is important for our analysis, classification/categorization is more meaningful for investors. Because investors can trigger actions easier based on different scenario/categorization. For our analysis, we break the daily percentage change into 6 categories/groups/scenarios per table 14.

It is common for different stocks to have different volatility. For instance, it is not typical for Apple stock to move up and down 5 percentage within a day. However, that is not uncommon for highly volatile stocks such as SMCI. This is perfectly illustrated in fig 20. where it lists the average daily stock price change percentage (in absolute value) So the definition of big gain or big drop will really depend on stock volatility.

Group	Label
1	Big Gain
2	Moderate Gain
3	Small Gain
4	Small Drop
5	Moderate Drop
6	Big Drop

Table 14: Daily Stock Percentage Change Group.

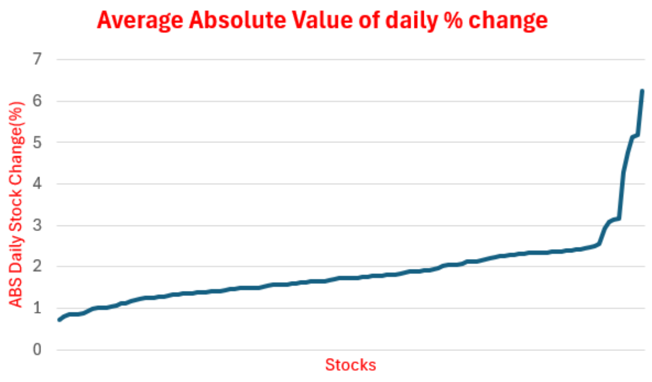
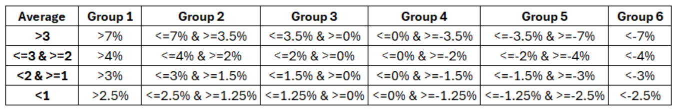


Figure 20: Average Absolute Value of Daily Stock Price Change.

To reflect this volatility, we introduced 4 different sets of groupings based on their average daily stock price change. The exact breakdown is shown in fig 21.

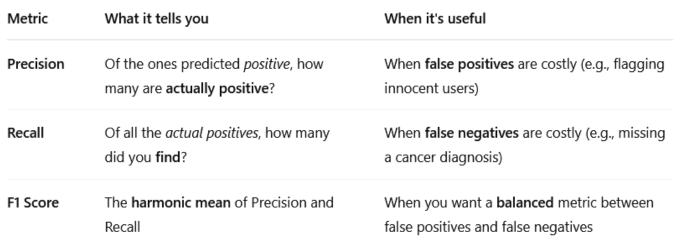


Average	Group 1	Group 2	Group 3	Group 4	Group 5	Group 6
>3	>7%	<=7% & >=3.5%	<=3.5% & >=0%	<=0% & >=-3.5%	<=-3.5% & >=-7%	<-7%
<=3 & >=2	>4%	<=4% & >=2%	<=2% & >=0%	<=0% & >=-2%	<=-2% & >=-4%	<-4%
<2 & >=1	>3%	<=3% & >=1.5%	<=1.5% & >=0%	<=0% & >=-1.5%	<=-1.5% & >=-3%	<-3%
<1	>2.5%	<=2.5% & >=1.25%	<=1.25% & >=0%	<=0% & >=-1.25%	<=-1.25% & >=-2.5%	<-2.5%

Figure 21: Daily Stock Price Change Grouping Sets.

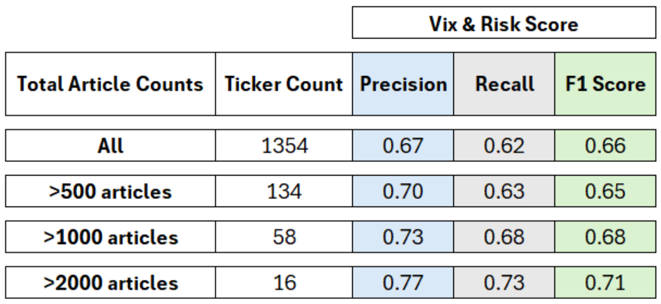
One of our primary goals is to warn investor about a potential stock price drop. In our modeling, we combine groups 5 and 6 (Moderate Drop and Big Drop) into one group and rest to another group. We then see how accurate we are at predicting correctly groups 5 and 6. There are many measures we could use to see our classification accuracy. If false positive is more costly we should use precision. On the other hand, F1 score is a balanced metric between the two.

Our classification result fig 23, shows similar observation vs article count. Just as we saw in regression analysis, higher article count corresponds to higher performance. This is caused by same reasons stated previously.



Metric	What it tells you	When it's useful
Precision	Of the ones predicted *positive*, how many are **actually positive**?	When **false positives** are costly (e.g., flagging innocent users)
Recall	Of all the *actual positives*, how many did you **find**?	When **false negatives** are costly (e.g., missing a cancer diagnosis)
F1 Score	The **harmonic mean** of Precision and Recall	When you want a **balanced** metric between false positives and false negatives

Figure 22: Classification Metric Comparison.



		Vix & Risk Score		
Total Article Counts	**Ticker Count**	**Precision**	**Recall**	**F1 Score**
All	1354	0.67	0.62	0.66
>500 articles	134	0.70	0.63	0.65
>1000 articles	58	0.73	0.68	0.68
>2000 articles	16	0.77	0.73	0.71

Figure 23: Classification Result.

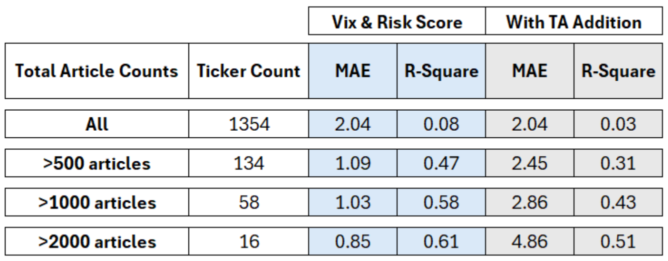
9.4 Technical Analysis:

NLP/Machine Learning is our primary focus for stock price prediction. There are two other major approaches:

1. **Technical Analysis:** Uses historical price and volume data to identify trends, patterns, and signals. In our case, we use:
   (a) Bollinger Band.
   (b) Relative Strength Index (RSI).
   (c) Stochastic Oscillator.
2. **Fundamental Analysis:** Focuses on company financial, industry health, and macroeconomic factors (e.g. P/E ratio, revenue growth, earnings reports).

We want to incorporate one additional method and see if it will improve our model performance. Fundamental analysis will require very comprehensive financial information of each company and it will be difficult to obtain and incorporate into our model. Technical analysis, on the other hand, is relatively easy as we already have price data. We choose Bollinger Band, RSI and Stochastic Oscillator as those are relatively easy to calculate and are based on information we already have. Unfortunately, we did not see a performance improvement by adding these information. In contrast, we actually saw a performance degradation fig 24 . We believe this degradation is due to additional noise introduced by 6 more TA features. Reinforcement learning (RL) might be a better method for the following reasons:

1. More strategic instead of static.
2. Has temporal awareness. RL models sequence or long term dependencies.



		Vix & Risk Score		With TA Addition	
Total Article Counts	**Ticker Count**	**MAE**	**R-Square**	**MAE**	**R-Square**
All	1354	2.04	0.08	2.04	0.03
>500 articles	134	1.09	0.47	2.45	0.31
>1000 articles	58	1.03	0.58	2.86	0.43
>2000 articles	16	0.85	0.61	4.86	0.51

Figure 24: Model Performance Comparison with Addition of Technical Analysis.

Due to time constraints and the complexity of the RL method, we did not incorporate RL into our analysis. That could be something we recommend as a future action.

9.5 Summary

1. **VIX and Risk Score interaction improve model performance:** Each Vix is a weak feature individually and Risk Score performs much better with Vix added.
2. **Article count has big impact on performance:** Higher article count leads to higher performance for both regression and classification analysis.
3. This comparison reaffirms our earlier conclusion that token-based sentiment metrics, rather than traditional technical indicators, offer greater predictive power in this context.

10 Topic Modeling Approach

Our topic modeling approach leverages **BERTopic** to extract latent themes from financial news articles. By converting unstructured text into topic-based features, we can capture underlying market sentiments and risk signals that improve stock price movement predictions.

10.1 Methodology and Model Parameters

We follow a multi-step pipeline:

1. **Text Preprocessing:** Articles are cleaned, tokenized, and lemmatized. Stopwords and non-informative tokens are removed.
2. **Embedding Generation:** We compute sentence embeddings using the `SentenceTransformer` model `"paraphrase-MiniLM-L3-v2"`.

3. **Dimensionality Reduction:** UMAP is applied with parameters: `n_neighbors = 5`, `min_dist = 0.5`, and `n_components = 5`.

4. **Topic Extraction:** BERTopic is then used to cluster the embeddings into topics. We experimented with different numbers of topics (e.g., $K = 5$, 10, 15) and found that $K = 10$ offers the best balance between topic granularity and coherence (average coherence ≈ 0.42).

10.2 Feature Engineering

Once each article is assigned a distribution over topics, we aggregate topic information at the ticker-day level by computing several features:

- **`topic_avg_movement`:** The historical average return for each topic.
- **`topic_sensitivity`:** The standard deviation of returns associated with each topic.
- **`sentiment_impact`:** Derived from FinBERT analysis as the difference between positive and negative sentiment scores.
- **`market_volatility`:** A 30-day rolling standard deviation of the price change percentage.

This process helps identify clusters of event-driven themes—such as legal, earnings, or M&A news—that have been shown to carry predictive value in prior event-based financial modeling research [**?**].

These engineered features serve as inputs to our per-ticker predictive models.

10.3 Per-Ticker Modeling and Quantitative Results

To evaluate the predictive power of topic-based features, we trained separate XGBoost regressors for each ticker with:

- **Target:** Daily price change percentage.
- **Metrics:** Mean Absolute Error (MAE) and R^2 score.
- **Feature Correlations:** Pearson correlations between each feature and the price change target. For example, correlations (in absolute value) range from about 0.05 to 0.62, indicating that the features are moderately predictive.

Table 15 (values provided here are illustrative) summarizes these metrics:

Feature	**Correlation (r)**
topic_avg_movement	0.25
topic_sensitivity	0.09
sentiment_impact	0.07
market_volatility	0.08
risk_score_topic	0.11

Table 15: Summary of feature correlations with the price change target.

10.4 Impact of Article Volume on Model Performance

We further investigated how the number of articles per ticker affects model performance:

- Tickers with fewer than 100 articles generally exhibit poor performance.
- Tickers with more than 250 articles show a significant improvement in R^2 scores (e.g., a median R^2 around 0.57).

Figure 25 displays a scatter plot (with the number of articles on a logarithmic scale) versus R^2 scores, overlaid with a regression line. In Figure 26, a LOWESS smoothed curve further highlights the positive trend between article volume and model performance.

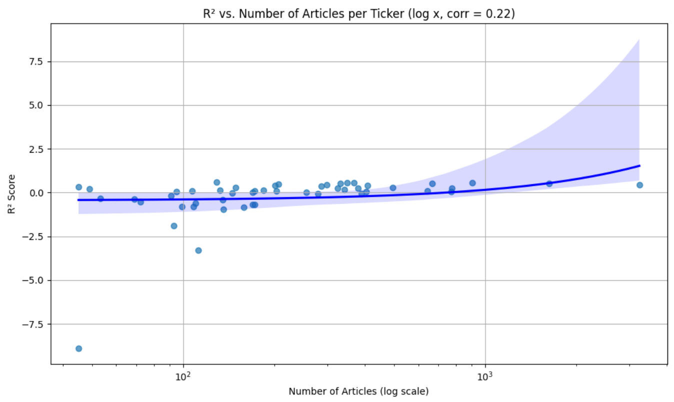


Figure 25: Scatter plot of number of articles per ticker vs. R^2 score with a regression line.

10.5 Summary

This topic modeling framework not only captures latent themes in financial news using BERTopic but also translates them into meaningful features that enhance per-ticker stock price movement prediction. The integration of topic-based features with sentiment analysis and article volume metrics has led to improved predictive performance—especially for tickers with a higher volume of news. This method showed particular strength for tickers with a large volume of articles, suggesting that topic-based features are especially valuable in high-coverage environments.

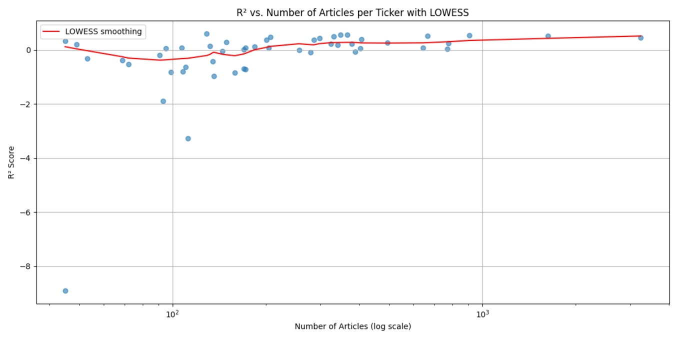


Figure 26: LOWESS smoothed trend of ticker article volume (log scale) vs. R^2 score.

11 Short-Term vs. Long-Term Opportunities

This project has laid foundational groundwork on modeling stock volatility using sentiment and token-based risk features. Should this work be extended, there are several promising approaches to deepen the analysis and enhance predictive performance.

Not all enhancements have equal feasibility or time requirements. We identify a few that are viable in the short term versus others that require more R&D investment:

- **Short-Term:** Segmentation, adaptive ensembles, multi-feature fine-tuning.
- **Long-Term:** Reinforcement learning, mixture-of-experts, and hierarchical multitask models.

11.1 Segmentation or Clustering

Concept: Group stocks based on similar characteristics such as market capitalization, article volume, or model performance metrics (e.g., MAE and R^2), aiming for more homogeneous data clusters.

Implementation Ideas:

- **Unsupervised Clustering:** Apply algorithms like k-means, hierarchical clustering, or DBSCAN on relevant features to discover inherent groupings, e.g., high-cap versus low-cap stocks.
- **Domain-Based Segmentation:** Explicitly segment stocks based on predefined criteria (small, mid, large-cap) and create tailored predictive models for each segment.

Pros and Cons:

- *Pros:* Specialized models for distinct groups, better handling of data sparsity.
- *Cons:* Potential complexity in cluster boundary definitions and feature normalization.

11.2 Mixture-of-Experts Models

Concept: Develop multiple specialized predictive models ("experts"), each suited to particular stock subsets or market conditions. A gating model dynamically assigns weights to each expert's prediction.

Implementation Ideas:

- **Expert Networks:** Train neural networks on distinct subsets or conditions.
- **Gating Mechanism:** Utilize a secondary model to dynamically determine expert weighting based on input features.
- **Dynamic Adaptation:** Continuously update the gating model to respond to evolving market dynamics.

Pros and Cons:

- *Pros:* Flexibility, dynamic responsiveness to market conditions.
- *Cons:* Increased complexity, risk of overfitting, higher computational demands.

11.3 Multi-Task or Hierarchical Modeling

Concept: View each stock prediction as a distinct but related task, leveraging shared information to inform predictions across tasks.

Implementation Ideas:

- **Multi-Task Neural Networks:** Employ shared layers for general trends and task-specific layers for individual stock nuances.
- **Hierarchical Bayesian Models:** Assume parameters for each stock come from a common distribution, sharing strength between data-rich and sparse stocks.

Pros and Cons:

- *Pros:* Better predictions for stocks with limited data, captures both shared and individual patterns.
- *Cons:* Complex model structures, careful balancing required to retain stock-specific details.

11.4 Adaptive Weighting in Ensemble Methods

Concept: Employ ensembles of models, adaptively weighting contributions based on historical performance under specific conditions.

Implementation Ideas:

- **Stacking and Blending:** Combine predictions from multiple models using a meta-model informed by past accuracy.
- **Performance-Based Weighting:** Dynamically adjust ensemble weights based on recent prediction accuracy.
- **Adaptive Algorithms:** Use boosting methods to iteratively focus on correcting previous model errors.

Pros and Cons:

- *Pros:* Enhanced overall predictive accuracy, adaptability.
- *Cons:* Increased complexity, potential for overfitting.

11.5 Incorporate Reinforcement Learning

Concept: Enhance technical Analysis and NLP based analysis by using Reinforcement Learning to training an agent to make decisions by interacting with an environment to maximize rewards over time.

Technical Analysis Implementation Steps:

- **Define the Environment:** Use gym-anytrading or FinRL for financial RL environment. Define states, action space and rewards.
- **Choose an RL Algorithm:** Choose either Deep Q-Learning (DQN), PPO, A2C or DDPG.
- **Build the Agent:** Use neural network architecture that maps state $\rightarrow$ action values or probabilities). Train using the chosen RL algorithm.
- **Train the Agent:** Choose neural network architecture that maps state $\rightarrow$ action values or probabilities. Use backpropagation and experience replay (for DQN) or advantage estimations (for PPO).
- **Back test and Evaluate:** Test the trained agent on unseen data. Compare to benchmarks.

NLP Based Analysis:

- **Extract Features with NLP:** Apply NLP models to get article embeddings or sentiment scores.
- **Create State Representations:** Combine NLP features with market indicators.
- **Define the RL Trading Environment:** Define state, actin space, reward and episode.
- **Choose and Train the RL Agent:** Select RL algorithm, similar algorithms listed for technical analysis. Train the agent by simulating trading over historical news and price data.
- **Back test and Evaluate:** Compare to baseline strategies. Evaluate metrics.

11.6 Final Thoughts

Selecting an optimal approach—or a combination thereof—depends on practical constraints, data availability, and computational resources. Often, hybrid strategies, such as segmenting data and using mixture-of-experts or multi-task frameworks enhanced by additional data streams, provide the most robust outcomes. Future work could involve iterative prototyping, rigorous validation against historical data, and careful evaluation of practical constraints.

12 Workload Distribution

- **Luis Tupac:** In charge of database creation and maintenance, data pipelines, and data transformation; FinBERT sentiment classification on financial reports, headline news, and new testing data. Worked on VIX Time Series Modeling and Thematic Clustering approaches.
- **Brian Adams:** Exploratory data analysis, experimenting with the calculation of risk scores, and exploring different modeling approaches (topic modeling, token score with sentiment).
- **Hilung Huang:** Overall project approach, initial modeling, alternative risk score method, Named Entity Recognition enhancement, technical analysis addition, and coordination of tasks.

13 Conclusion

This project has extensively demonstrated the potential of NLP-driven approaches to predict market volatility through detailed analysis of financial sentiment indicators. We systematically explored and compared various modeling strategies, including token-level sentiment analytics, volatility forecasting using VIX, topic modeling, and multi-feature ensembles. Our results clearly indicate that models integrating comprehensive token-based risk scores significantly outperform simpler sentiment-only models, validating our initial hypotheses regarding the importance of nuanced textual insights.

We also found that predictive accuracy greatly benefits from adequate data volume and carefully engineered and aggregated features. Our experiments underscore the importance of balancing model complexity with robustness, highlighting that streamlined, thoughtfully constructed features often provide the most stable predictions.

Future efforts extending this work should focus on advanced hybrid approaches, such as segmentation-based clustering, hierarchical multitask modeling, dynamic ensemble methods, and reinforcement learning as outlined in our recommended next steps. Ultimately, by continuing to refine these methods and rigorously validating them against diverse market conditions, the proposed sentiment-based Risk Scores can evolve into powerful, actionable tools for investors aiming to anticipate and respond effectively to market inflection points.

References

[1] ChartSchool. Bollinger bands. `https://school.stockcharts.com/doku.php?id=technical_indicators:bollinger_bands`, 2025. Accessed: April 4, 2025.

[2] ChartSchool. Relative strength index (rsi). `https://school.stockcharts.com/doku.php?id=technical_indicators:relative_strength_index_rsi`, 2025. Accessed: April 4, 2025.

[3] ChartSchool. Stochastic oscillator. `https://school.stockcharts.com/doku.php?id=technical_indicators:stochastic_oscillator_fast_slow_and_full`, 2025. Accessed: April 4, 2025.

[4] Zeynep Genc. Finbert: Financial sentiment analysis with bert. `https://medium.com/prosus-ai-tech-blog/finbert-financial-sentiment-analysis-with-bert-b277a3607101`, July 2020.

[5] Mofiz Mojib Haider, Md. Arman Hossin, Hasibur Rashid Mahi, and Hossain Arif. Automatic text summarization using gensim word2vec and k-means clustering algorithm. In *2020 IEEE Region 10 Symposium (TENSYMP)*, pages 283–286, 2020.

[6] Zhiyong Hu, Indranil Bose, Nancy S Koh, and Ling Liu. Listening to the crowd: Automated analysis of financial news and social media to detect and predict abnormal returns. In *Proceedings of the 51st Hawaii International Conference on System Sciences*, 2018.

[7] Tomáš Kliegr and Ondřej Zamazal. Antonyms are similar: Towards paradigmatic association approach to rating similarity in simlex-999 and wordsim-353. *Data Knowledge Engineering*, 115:174–193, 2018.

[8] Alok Pratap. Exploring the benefits of dbscan and hdbscan in remote sensing technology. `https://medium.com/@alokguy2004/exploring-the-benefits-of-dbscan-and-hdbscan-in-remote-sensing-technology-3c8e37985167`, February 2023.

[9] Shubham Sharma, Nick Becker, Bryan Tepera, and Daniel Gama Dessavre. Nvidia cuml brings zero code change acceleration to scikit-learn, March 31 2025. NVIDIA Technical Blog.

[10] Paul C Tetlock. Giving content to investor sentiment: The role of media in the stock market. *The Journal of Finance*, 62(3):1139–1168, 2007.