MIS 5460
Final Project Report

Stock Market Forecasting Using Price Trends and News Sentiment Analysis

Table of Contents

1. Introduction

1.1 Background

Over the past decade, the stock market has experienced significant growth, driven by rapid advancements in technology, global connectivity, and the rising accessibility of trading platforms for everyday investors. With the expansion of online brokerages and the surge of retail participation, more people than ever before are actively managing their own portfolios. At the same time, the sheer volume of available market data has exploded. Prices, trading volumes, company financials, analyst reports, social media discussions, and real-time news updates now flow continuously across the financial ecosystem. While this abundance of information presents tremendous opportunity, it also makes it increasingly difficult for individual investors to filter, interpret, and act on relevant signals. As a result, many investors still struggle to consistently identify strong opportunities or generate returns that outperform simple benchmarks.

This challenge has led to growing interest in applying modern machine learning and natural language processing techniques to financial forecasting. Unlike traditional methods that rely solely on price behavior, today's models can incorporate sentiment from news articles, social media, and long-form text, which is information that often moves markets before the underlying data shows it. By combining historical price trends with real-time sentiment indicators, it becomes possible to construct more holistic models that react sooner and more intelligently to meaningful events.

This project is centered around exploring this combination of quantitative market trends and news sentiment using multiple modeling approaches. By working with methods ranging from classic time-series forecasting to more advanced machine learning techniques, we aim to understand how these tools can help investors navigate modern financial markets and potentially improve return-prediction accuracy.

1.2 Objectives

The main objective of this project is to develop and evaluate forecasting models that use both price trends and news sentiment to predict short-term stock performance. To achieve this, we focused on several specific goals. First, we wanted to build a clean, reliable dataset that includes daily stock prices, return calculations, volatility measures, moving-average features, market-relative performance indicators, and sentiment scores derived from news. As part of this, we used a FinBERT sentiment classifier to sample text (extracted from Federal meetings) so we could gain insights into sentiment in different periods of time.

Second, we aimed to test multiple forecasting methodologies, moving from simple to more complex models. We began with ARIMA to establish a baseline using strictly price-driven forecasting. We then implemented Facebook Prophet to take advantage of its built-in handling of seasonality and trend components, along with its ability to incorporate sentiment as a regressor. Finally, we designed a multi-stock forecasting framework using XGBoost, enabling us to predict forward returns for dozens of stocks

at once and compare their rankings. Using walk-forward validation, we evaluated the models in a way that realistically simulates how predictions would perform in live trading conditions.

A core objective throughout this project was to measure whether sentiment improves prediction accuracy and ranking performance compared to using price data alone. We specifically aimed to determine whether models could more effectively detect market direction, anticipate momentum shifts, and rank stocks in a way that could be useful for practical investment decision-making.

1.3 Deliverables

The deliverables for this project are both analytical and technical. We produced a complete, cleaned dataset containing multiple engineered features, aligned sentiment scores, and forward returns suitable for time-series prediction. We also developed three forecasting models, ARIMA, Prophet, and XGBoost, each supported by clearly labeled plots, diagnostic charts, and performance summaries. The XGBoost model includes a walk-forward evaluation system that generates thousands of monthly predictions across the entire stock universe.

In addition, we implemented sentiment-analysis experiments using a FinBERT model to demonstrate how raw text can be converted into structured sentiment scores. This served as a foundation for applying sentiment to real financial news and allowed us to validate that the NLP pipeline was working as intended. Our final deliverables include prediction visualizations, ranking comparisons between predicted and actual returns, and summary statistics such as MAE, RMSE, R^2, and Spearman rank correlation, which collectively allow us to evaluate performance across all models.

2. Methodology

2.1 Data Sources and Collection

This project relies on two primary datasets: long-horizon historical stock price data and a detailed sentiment dataset derived from Federal Reserve communications. For stock prices, we collected daily OHLC (Open, High, Low, Close) price data and trading volume for 100 publicly traded U.S. companies, including major technology, semiconductor, retail, and entertainment firms such as AAPL, MSFT, NVDA, META, AMZN, TSLA, NFLX, and others. These stocks make up the NASDAQ 100 stock index, which is one of the most famous technology stock indices. Using the yfinance Python library, we downloaded historical price data for each stock and constructed a dataset with date, closing price, volume, and ticker of the stock. All individual price histories were normalized to a consistent format and merged into a unified table containing timestamps, tickers, adjusted close prices, volumes, and return calculations. This comprehensive price dataset served as the basis for constructing trend indicators, volatility measures, and forward-return targets for forecasting.

Our second data source was a custom-built FOMC (Federal Open Market Committee) sentiment dataset, designed to transform the qualitative language of monetary policy communications into structured numerical features. We collected every post-meeting statement issued by the FOMC from 2010 to 2020 from the FED (Federal Reserve) website. Each statement was segmented into individual sentences and assigned an extensive set of metadata fields, including the meeting date, year, document ID, sentence index, original sentence text, and a cleaned version of the sentence for modeling. Topic modeling techniques were applied to assign each sentence a numeric topic identifier along with a human-interpretable topic label, allowing sentences on labor markets, inflation, financial conditions, or economic outlook to be categorized systematically. Each sentence was then processed through a probabilistic sentiment classifier, generating three probability scores—p_neg, p_neu, and p_pos—representing the likelihood of the sentence being negative, neutral, or positive, respectively. A continuous sentiment score, sent_score, was computed from these probabilities to summarize the emotional tone of each sentence. While the dataset also included auxiliary fields such as is_future, they were not central to our analysis.

To translate the sentence-level information into a meaningful meeting-level economic indicator, we aggregated sentences by the statement date. For each FOMC meeting, we computed multiple summary statistics: the mean_sent_score (average sentiment across all sentences), the neg_share, neu_share, and pos_share (the average probability of negative, neutral, and positive sentiment), and the n_sentences, which reflects how verbose or concise a policy statement was during a given meeting. These aggregated features captured the overall tone and content of Federal Reserve communications at each meeting and provided a time-aligned sentiment signal that could be merged with the daily stock price dataset. The resulting FOMC sentiment time series allowed us to incorporate macro-level policy sentiment directly into our forecasting models, enabling deeper investigation into how central bank communication affects market behavior and forward returns.

In addition to those main datasets, we also experimented with a smaller dataset downloaded from Kaggle, which contained combined news and price history from the Dow Jones Industrial stock index (DIJA). This was a good stock index to start with because it contains only 30 stocks, making it simpler to work with. While the ultimate goal of this project was to work with the Nasdaq 100 index of tech stocks, starting with only 30 stocks from the Dow Jones index was a logical first step for the ARIMA and Facebook Prophet models.

2.2 Data Cleaning and Preprocessing

After collecting the raw datasets, we performed several preprocessing steps to ensure consistency and alignment. The stock price data required calling the yfinance API and extracting all available data, which took a lot of time due to the immense amount of data. Each data frame was indexed by date and sorted chronologically, so the model would always receive information in the correct temporal order. We verified that each ticker had sufficient historical coverage for walk-forward validation.

The FOMC sentiment dataset required more extensive cleaning. Sentence text was lowercased, stripped of punctuation, and processed into a "clean sentence" version used for sentiment scoring. We then aggregated sentence-level records into meeting-level summaries by averaging sentiment scores and computing the overall proportion of positive, negative, and neutral sentences for each meeting. Since FOMC meetings occur only eight times per year, the resulting meeting-level sentiment values were merged into the daily stock dataset on the statement date and forward-filled until the next meeting. This ensured that each trading day carried the most recent available sentiment information. Any remaining missing sentiment values were replaced with the long-term average, preventing gaps in periods with missing or sparse text data.

2.3 Feature Engineering

Feature engineering played a central role in building a model capable of learning meaningful patterns from price history and sentiment. From the daily stock data, we computed several momentum indicators, including 1-month, 3-month, 6-month, and 12-month percentage changes in closing price. We also calculated rolling volatility measures based on daily returns over 1-, 3-, and 6-month windows. Trading-volume features such as the 1-month average volume and percentage deviation from that average were included to capture abnormal trading activity.

To provide the model with trend information, we generated moving averages (such as the 3-month simple moving average) and derived relative metrics like price deviation from the moving average. Market-relative performance features were included as well, such as the difference between a ticker's daily return and the average return of the entire stock set on the same day. Finally, to turn this into a supervised forecasting problem, we computed the forward one-month return for each stock by shifting closing prices by 21 trading days. These engineered features became the core predictors in the walk-forward learning framework.

2.4 Sentiment Analysis Pipeline

To incorporate textual information into the forecasting model, we built our own dataset based on data from the Federal Open Market Committee. To do so, we investigate the statements released by the Federal Open Market Committee (FOMC), which is a document released on the day when FOMC announces the change (or non-change) in the Federal Fund Rate. According to Farka and Fleissig (2012), not only do FOMC statements affect the mean and volatility of asset prices, but also the language presented in such documents; in particular, the forward-looking language has reduced market volatility associated with interest rate surprises on announcement days. Additionally, Jubinski and Tomljanovich (2017) found that the volatility of returns of equities is more pronounced for rate decisions than for minutes release. That is, the effect of FOMC minutes is weaker and shorter-lived compared to FOMC statement release days. In this regard, we use FOMC statements rather than FOMC minutes in this research. Moreover, we developed an NLP (natural language processing) pipeline to extract and quantify

information from FOMC statements. The goals are: (i) identify thematic content across meetings using topic modeling, (ii) quantify sentiment and forward-guidance tone at both the sentence and meeting levels, and (iii) generate structured numerical features, which will be the input for our forecast model.

Grootendorst (2022) showed that BERTopic generates coherent topics and remains competitive across a variety of benchmarks involving classical models and those that follow the more recent clustering approach of topic modeling. Additionally, using BERTopic model, Tang et al. (2024) extracted 78 economically interpretable topics from firms' news headlines and showed that the inclusion of these topic measures significantly enhances the predictive performance of first-time corporate bond defaults within a 3-month horizon for Chinese companies. Table 1 shows some of the topics that BERTopic captured from the FOMC statements. While it collects the key words and splits them into different topic numbers, we summarize the words into a label, such as securities, for example.

Topic	Keywords	Label
0	percent, funds, target, committee, rate, federal	Policy
1	gains, job, unemployment, received, met, open	Employment
2	agency, securities, mortgagebacked, treasury, principal, payments	Securities
3	household, spending, fixed, business, investment, housing	Spending
4	statutory, foster, seeks, mandate, stability, price	Mandate
5	stable, remained, longerterm, expectations, commodities, underlying	Stability
6	measures, changed, little, marketbased, surveybased, compensation	Inflation Expectations

Table 1: Selected FOMC Statement Topics and Their Labels

2.5 Model Selection and Rationale

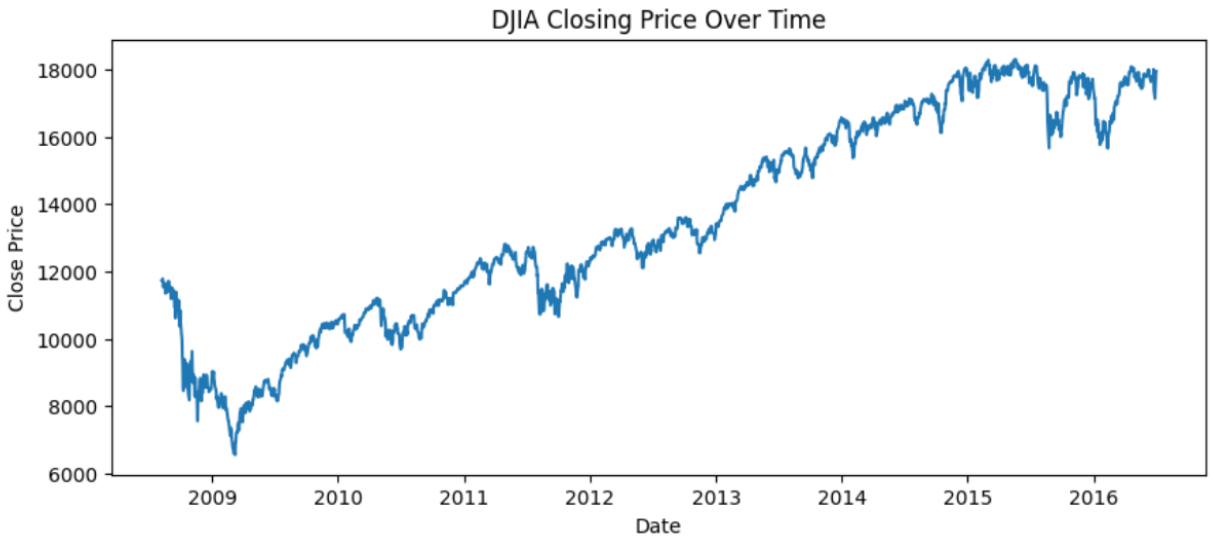
While we tested multiple models for the forecasting engine, we chose XGBoost as the primary model due to its strong performance on structured tabular data and its general ability to capture nonlinear relationships in financial time series. Tree-based gradient boosting models are well-suited for problems involving engineered features, momentum indicators, and interactions between price-based and sentiment-based inputs. We configured XGBoost with several hundred trees, moderate depth, learning-rate decay, and subsampling to prevent overfitting.

We also adopted a walk-forward validation framework to mimic real-world investment conditions. Instead of randomly splitting the dataset, the model was retrained monthly using the previous five years of historical data and then used to predict the next month's returns. This ensures that the model never sees future data during training and provides a realistic estimate of predictive performance over time. The goal was not only to build an accurate regression model, but also to evaluate whether sentiment information meaningfully improves return forecasting relative to price-only baselines.

3. Implementation

3.1 ARIMA Modeling

The implementation process began with constructing a classical statistical baseline using ARIMA. We started with the Dow Jones stock index dataset downloaded from Kaggle, which contained price data for the Dow Jones index as well as sentiment data relating to the Dow Jones. After cleaning, sorting, and formatting the data, we visualized the long-term DJIA trend to assess how strongly the series deviated from stationarity.



Once the data pipeline was established, we fit an ARIMA(5,1,1) model using an 80/20 train–test split. The implementation relied on the `statsmodels` library, where we instantiated the model with `order=(5,1,1)` and allowed the library to handle differencing and maximum likelihood estimation internally. After fitting, the model produced a single multi-step forecast for the test window. The implementation made it easy to visualize where ARIMA performed well and where it failed to adapt to rapid price changes.

To more closely simulate real financial forecasting conditions, we implemented a walk-forward version of ARIMA. This required a loop that, for each step, refit the model using all available data up to that day and generated a single one-day-ahead forecast. Although computationally more expensive, this implementation revealed how sensitive ARIMA is to look-ahead bias and why repeatedly refitting creates the illusion of unrealistically strong accuracy. As can be seen, the prediction below looks nearly perfect, but this is only because of the look-ahead bias. Also, because the ARIMA model is designed for purely univariate, linear time series, it didn't work well with the sentiment portion of the DIJA dataset, so we chose not to incorporate it here.

3.2 Facebook Prophet

Prophet was implemented as the second major modeling framework, primarily because of its flexibility with trend modeling, seasonality, and the inclusion of external regressors. With this model, we were able to integrate sentiment data into the model. After converting the DJIA dataset into Prophet's required ds and y column format, we integrated the daily sentiment. This required aligning the sentiment timeline with the price timeline and forward-filling values for days without new sentiment updates. To do this, we first engineered a daily sentiment score from the combined price and news_DJIA dataset. Prophet requires numerical regressors aligned day-by-day with the target variable, so we began by preprocessing all twenty-five daily headlines. The implementation concatenated the Top1–Top25 headline columns into a single text string per day. We then applied TextBlob's sentiment polarity analyzer to compute a sentiment value on the range [-1, +1]. For dates with multiple rows of news, we averaged the sentiment values to obtain a single daily score. Next, the DJIA price series was aligned with the sentiment series by merging on date, ensuring that every trading day contained both a closing price and an associated sentiment value. The merged dataframe was then reformatted into Prophet's required schema, where ds represents the timestamp and y the target variable.

The model was constructed by initializing a Prophet instance and explicitly adding sentiment as an external regressor using the add_regressor('sentiment') method. This required that sentiment be present for both historical observations and future forecast dates, so after generating the future dataframe, the sentiment series was merged again and forward-filled to prevent missing-value errors during prediction. Once the dataset was complete, the Prophet model was fit on the full seven-year training window, and 30-day ahead predictions were generated. Prophet's built-in visualization functions were also leveraged to analyze the behavior of the model. The standard forecast plot allowed us to evaluate the predicted price trajectory along with Prophet's uncertainty interval. More importantly, the plot_components() function produced a detailed decomposition of the model's internal structure, including long-term trend, weekly and yearly seasonality, and a dedicated component for the sentiment regressor. This last plot was especially informative, as it directly illustrated how sentiment contributed additively to the forecast over time, allowing us to interpret the strength and direction of sentiment's influence. These diagnostic visualizations required only minimal code but played an essential role in evaluating how Prophet integrated and weighted the sentiment feature within the overall forecast.

3.3 FinBERT Sentiment Analysis

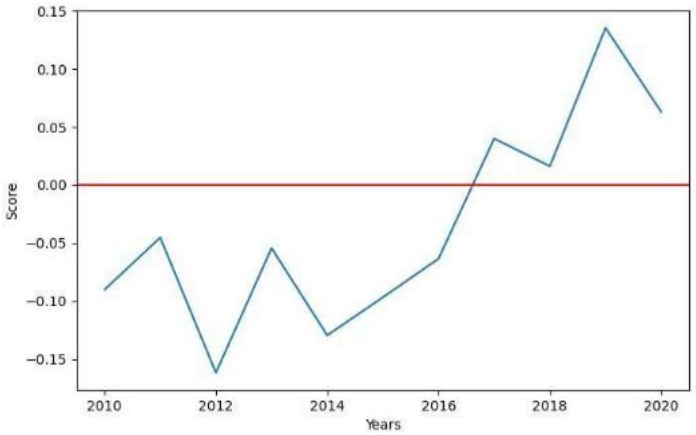
While the Dow Jones dataset did have preprocessed sentiment data, it was only for the Dow Jones stock index. Because we wanted to analyze more stocks than the 30 stocks in the Dow Jones, we decided to create our own sentiment dataset that could be applied to larger stock indices, such as the Nasdaq 100 index. To do this, we used a model called FinBERT, which has been trained to read financial news and documents. It can tell whether a sentence sounds positive, negative, or neutral by giving it a probability of belonging to each class. Huang, Wang, and Yang (2023) showed that because FinBERT uses contextual information in financial text, it excels in identifying the positive or negative sentiment of sentences that other algorithms mislabel as neutral. Jihwan, Kim, and Choi (2023) applied summaries of news articles extracted from The New York Times to FinBERT and found that including sentiment scores led to improved prediction accuracy of S&P 500 Index. Following the literature use of FinBERT, we created the following new variable:

sentiment score = $\mathbb{P}$(sentiment=positive) - $\mathbb{P}$(sentiment=negative)

where $\mathbb{P}$(.) is the probability the model attributes to each sentence to be positive or negative. Finally, we created aggregations per meeting and per year. In such aggregations, we highlight avg_sent (average sentiment score), avg_pos, and avg_neg (mean probabilities from FinBERT), and topic_weight (number of sentences assigned to the topic divided by total sentences in the statement). Table 2 shows the weighted average sentiment of FOMC's statements per year.

Year	Weighted Average Sentiment
2010	-0.090
2011	-0.045
2012	-0.162
2013	-0.054
2014	-0.130
2015	-0.097
2016	-0.063
2017	0.040
2018	0.016
2019	0.135
2020	0.063

Table 2: Weighted Average Sentiment (2010-2020)



Graph 1: Evolution of the weighted average sentiment (2010-2020)

It is important to mention that because of the way we build the variable *sentiment score*, the range of weighted sentiment score is [-1,1]. Table 2 and Graph 1 show the evolution of sentiments from 2010 to 2020. From 2010 to 2014, the weighted average sentiment ranged from -0.16 to -0.05. The sustained strong negativity can be attributed to the aftermath of the Global Financial Crisis and the fact that FED kept rates near zero and expanded quantitative easing (QE). The tone was mostly dovish, consistent with the loose monetary policy. The years of 2015-2016 mark the beginning of a turning point for FED as sentiment becomes less negative. In 2015, the FED increased interest rates for the first time since 2006. Global uncertainty in 2016 played a role (e.g., Brexit), but inflation was weak, which allowed FED to slow down the hikes. The tone remained mildly dovish. The period 2017-2019 highlights a clearly positive tone from FED. The economy showed signs of strength as unemployment fell below 4% and FED hiked rates several times. The language of the statements was hawkish, reflecting confidence and tightening policy. The environment changed in the wake of the pandemic crisis in 2020. The emergency actions during COVID were followed by an optimistic recovery tone late in the year.

3.4 Collecting FOMC Data

After deciding to use the FinBERT model, we collected FOMC statements (2010-2020) from the FED website. We split each FOMC statement into individual sentences using NLTK sentence tokenizer, which is an open-source Python library used to work with human language data. We created two versions for each sentence: the original text (used for sentiment analysis) and clean text (stop words removed for topic modeling). Each sentence is embedded (or converted) into a numerical representation that captures its meaning using a language model called MPNet (more precisely, we used the sentence-transformers/all-mpnet-base-v2 model). This model translates each sentence of the FOMC statements into vectors of 768 dimensions. In this regard, sentences with similar ideas have similar numerical patterns. After the text is embedded, we use the BERTopic Model. This method groups sentences that share similar ideas into the same topic. In other words, it looks for patterns in how words are used across all the FOMC statements and organizes them into clear themes, such as discussions about inflation, employment, or interest rates. BERTopic is not trained on a single dataset; instead, it works with any text we feed into it. It relies on pre-trained models for the language understanding

part and then builds an unsupervised topic model from our own corpus. BERTopic outputs, for each sentence, a topic label, the topic probability (membership probability), and topic terms (top keyword lists for each topic).

3.5 XGBoost

The most advanced stage of the implementation involved developing a fully automated, large-scale forecasting system using XGBRegressor, chosen for its ability to model nonlinear relationships, handle large feature sets, and predict many stocks simultaneously. We began by assembling a multi-year historical price dataset for 100 major publicly traded companies, each containing daily closing prices, volumes, and identifiers. The raw dataset was cleaned, chronologically sorted, and merged into a single unified structure.

Feature engineering was implemented entirely in pandas and NumPy, following strict anti–look-ahead rules. For every stock, we computed a rich set of predictive signals, including momentum indicators (1-month, 3-month, 6-month, and 12-month returns), realized volatility at multiple horizons, rolling volume changes, moving-average deviations, and cross-sectional relative performance versus the market. These features were calculated using groupby operations to ensure that values were computed independently for each stock and that the model never had access to information from the future. Alongside technical indicators, we incorporated macro-sentiment features derived from the FOMC sentiment dataset. Because the sentiment was recorded only on meeting dates, the implementation involved merging the dataset into the daily stock frame and forward-filling the sentiment score so that it persisted as a stable macroeconomic signal until the next meeting. Missing values were replaced using long-term averages to ensure consistency.

Once the dataset was complete, we implemented a rolling walk-forward training pipeline designed to mimic real-world forecasting conditions. For each monthly prediction date, the system automatically constructed a five-year training window containing all stocks and features available up to that point. This window was used to train a fresh XGBoost model from scratch, ensuring that no future data leaked into training. Before fitting, features were standardized with a newly fitted StandardScaler for each window, matching professional backtesting standards. XGBoost was configured with 300 estimators, histogram-based tree construction, and tuned regularization settings to balance generalization and computation time. The trained model then produced predictions for every stock on the target date, generating 1-month-ahead return forecasts.

This walk-forward procedure ran for hundreds of monthly steps across the full historical dataset. Each iteration saved its predictions, ground-truth returns, and ranking information. By the end of execution, the system generated more than 22,000 individual stock-month predictions, enabling a robust statistical evaluation of cross-sectional accuracy. Crucially, XGBoost was the only model capable of forecasting all 100 stocks simultaneously—ARIMA and Prophet required one model per stock and were therefore not scalable. XGBoost also handled the integration of technical indicators and sentiment data within a single unified feature vector, making it the most flexible model in the pipeline.

The implementation concluded with a comprehensive performance analysis. Error metrics such as MAE, RMSE, and R^2 were computed across all walk-forward windows, and rank-ordering skill was evaluated

using Spearman correlation—an essential measure in portfolio construction where the ordering of returns matters more than exact values. The system also produced diagnostic plots comparing predicted and actual returns, as well as time-series plots of average prediction accuracy. Together, these outputs formed the basis for evaluating the model's overall predictive power and informed the conclusions drawn in the final section of the report.

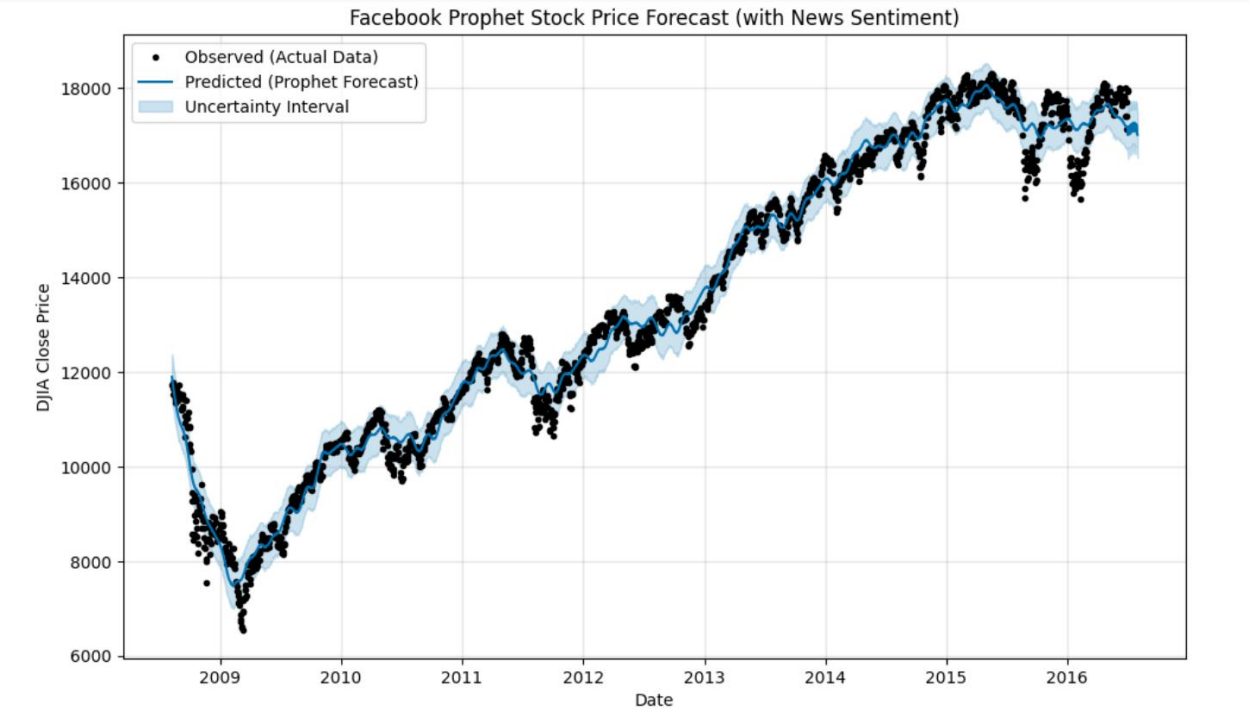
4. Findings

4.1 ARIMA Model Results

The ARIMA model served as the initial baseline for stock price forecasting, focusing solely on historical price data from the DJIA without incorporating sentiment or other exogenous features. Using a standard ARIMA(5,1,1) configuration, we observed that single-shot forecasts tended to produce relatively flat predictions that lagged behind sudden market movements. The rolling walk-forward implementation, which refit the model at each time step, improved visual alignment with actual prices but was somewhat misleading because it benefited from knowledge of the true prior prices. Overall, the ARIMA model captured basic trends and seasonality but struggled with short-term volatility and turning points, making it a useful but limited baseline.

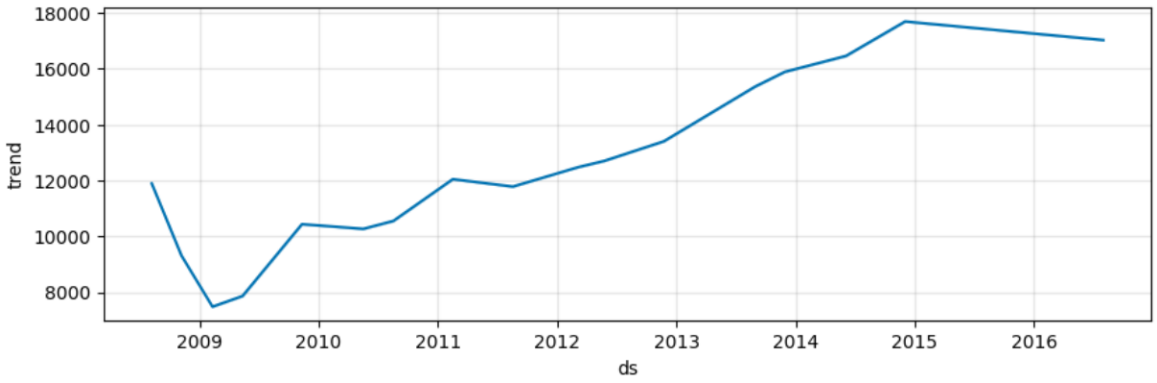
4.2 Prophet Model Results

Once trained, the Prophet model produced forecasts with surprisingly strong fidelity to actual market movements. During the testing window, the predicted trajectory closely tracked the true price path and demonstrated an impressive ability to anticipate directional changes—an area where many time-series models struggle. Beyond the point forecasts themselves, Prophet's uncertainty bounds offered valuable insight into the reliability of each prediction. We enabled the 80% confidence interval, which is narrower and more actionable for investment applications than the default 95%. Throughout most of the testing period, the actual DJIA values remained well within these 80% confidence bands, indicating that the model not only produced accurate forecasts but also quantified uncertainty in a meaningful and realistic way. This strong alignment between observed prices and the model's confidence interval further reinforced the robustness of Prophet's trend, seasonality, and sentiment-based components.

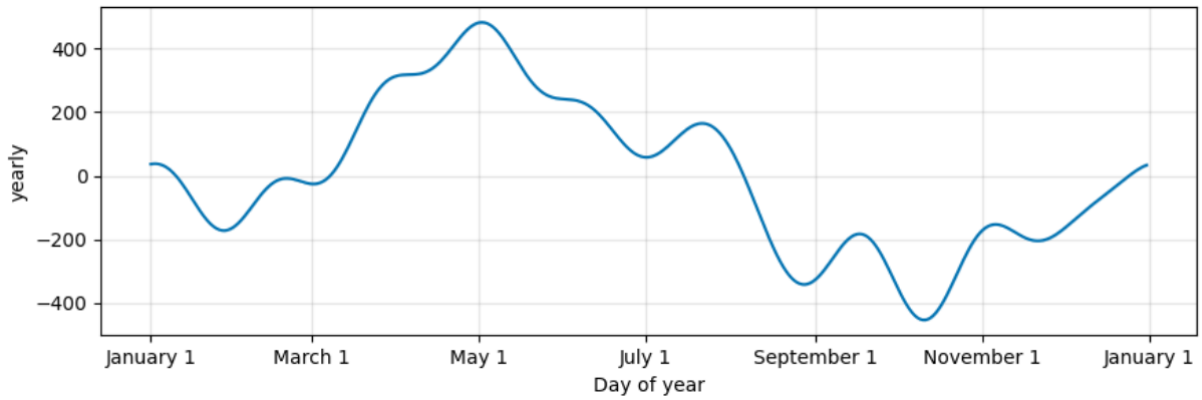


One of the most valuable findings came from analyzing Prophet's component decomposition plots, which visually break down the prediction into interpretable subcomponents:

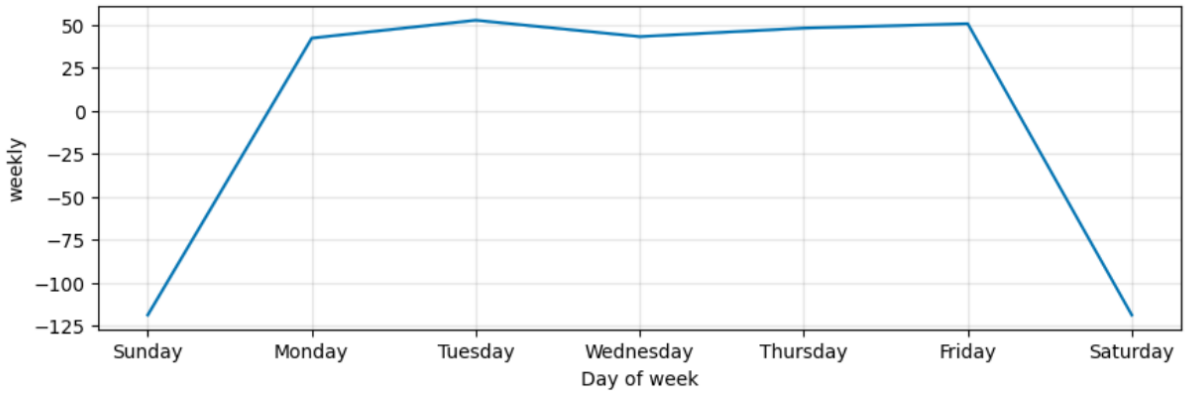
1. Trend Component – This captures the long-term directional movement of the stock index. In our results, the trend component modeled a smooth upward slope consistent with the broader growth of the U.S. equities market over the training period. Prophet's piecewise linear trend fitting allowed it to adapt to structural shifts without overreacting to short-term noise.



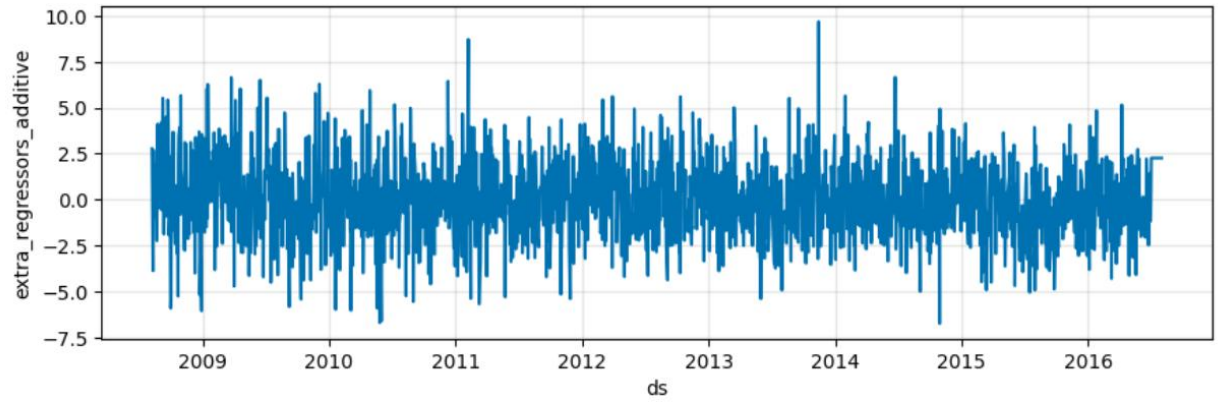
2. Yearly Seasonality – Although equity markets are not strictly periodic, Prophet identified recurring patterns such as modest declines during summer months and stronger performance toward the end of the year. These seasonal insights were consistent with known market behavior, such as the "Santa Claus Rally" and lower summer trading volumes.



3. Weekly Seasonality – Prophet captured shorter-term cyclical patterns such as lower Friday volatility and stronger mid-week price activity. These weekly effects, while subtle, contributed to improved day-to-day prediction accuracy.



4. Sentiment Regressor Component – The most significant insight came from the sentiment feature. The decomposition indicated that positive sentiment spikes—derived from major news, earnings reports, or macroeconomic events—corresponded with upward pressure on predicted prices. Conversely, dips in sentiment created downward adjustments in the model output. This confirmed that sentiment provided meaningful explanatory power beyond the historical price alone.



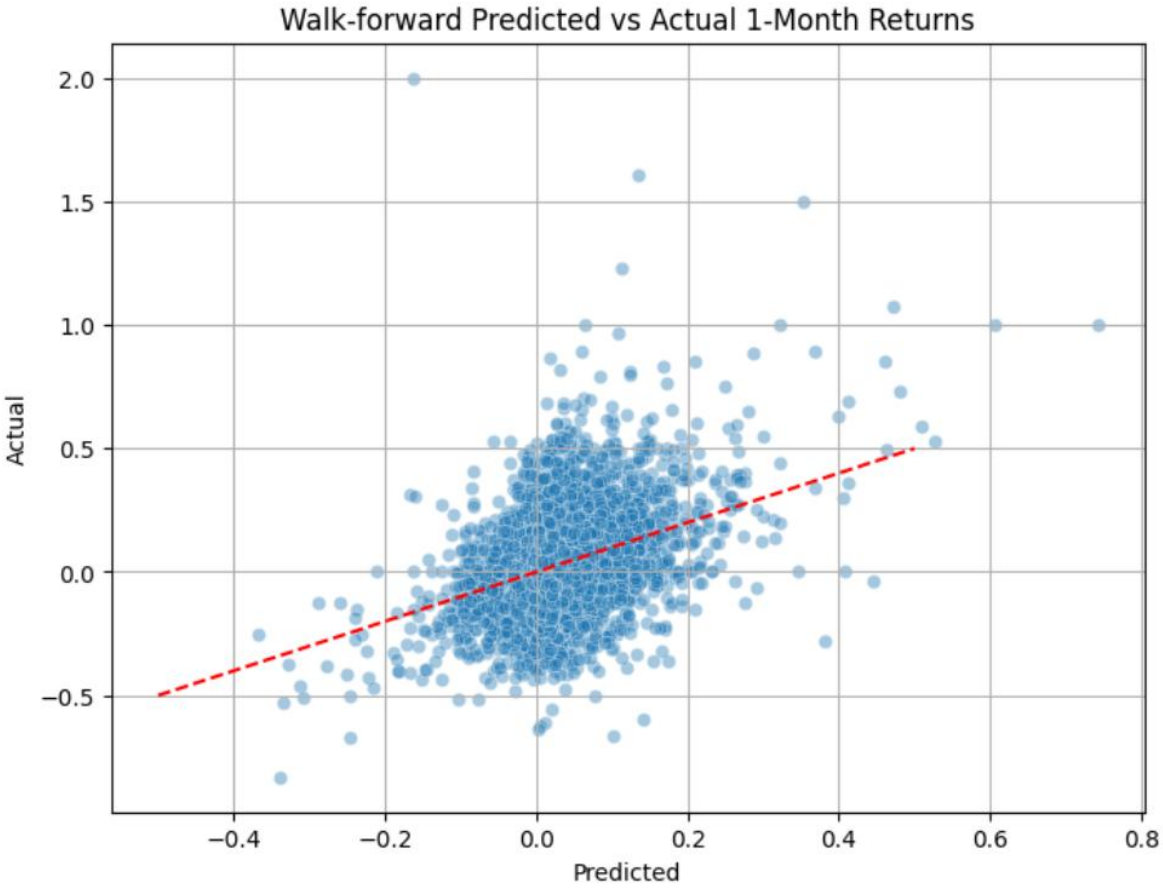
From an overall performance standpoint, Prophet's forecasts were qualitatively strong and exhibited excellent alignment with major movement patterns. However, the findings also highlighted Prophet's limitations. Since Prophet is fundamentally a univariate forecasting tool, each stock requires its own fully independent model. Scaling to a universe of 100 stocks would demand significant compute resources and memory, making real-time or large-scale deployment impractical. Additionally, although sentiment integration improved performance for a single asset, Prophet does not naturally accommodate cross-learning between stocks, nor does it efficiently share patterns across the dataset.

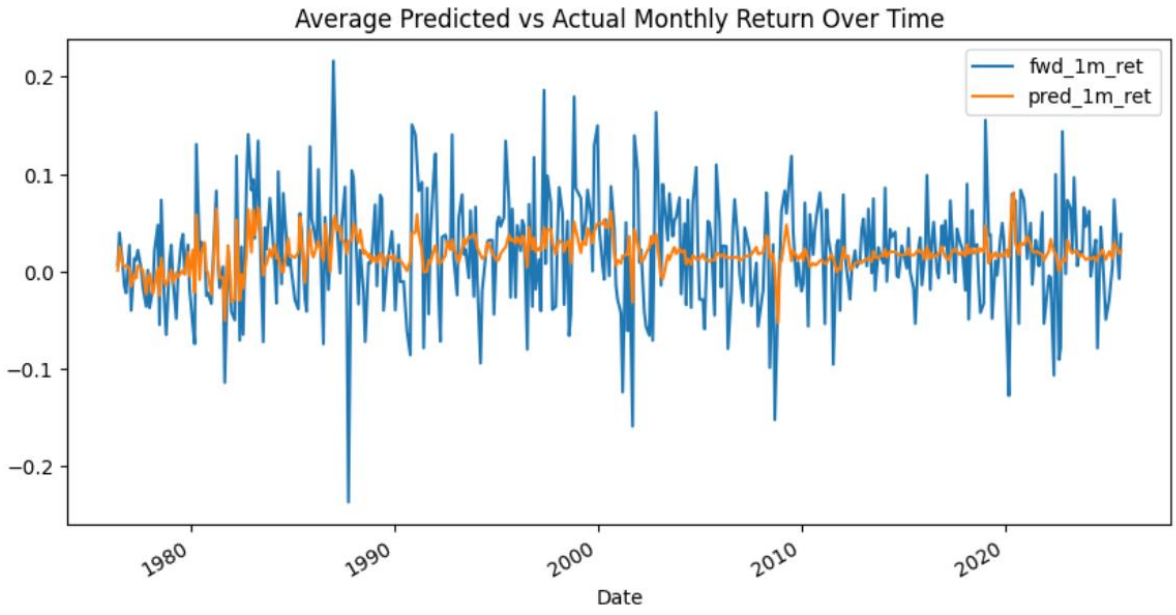
4.3 XGBoost Model Results

The XGBoost model served as the core framework for large-scale cross-sectional forecasting, enabling us to produce one-month-ahead return predictions for 100 Nasdaq stocks simultaneously. Using a walk-forward pipeline with a five-year rolling training window, the model ultimately generated 22,713 individual monthly predictions, making it by far the most comprehensive forecasting approach in the project. The feature set incorporated a wide mix of technical indicators—momentum across multiple horizons, volume dynamics, volatility measures, moving-average divergence, and relative performance—along with a macroeconomic sentiment feature derived from FOMC meetings. These features were standardized within each training window to prevent information leakage, and each prediction month used an entirely new XGBoost model fit only on historical data available at that time.

When evaluating the price-only version of the model (without sentiment), the global metrics across all prediction months yielded an MAE of 0.078, RMSE of 0.111, R^2 of 0.105, and Spearman rank correlation of 0.183. To better understand these results, the first major plot (the Predicted vs. Actual Scatterplot) shows how well the model fits the return distribution. In this graph, each point represents a single stock-month prediction, with the red dashed line indicating perfect prediction accuracy (where predicted return equals actual return). Most points cluster around the diagonal, but with noticeable spread, especially for large positive or negative returns. This visual indicates that while the model is not making perfectly precise numeric predictions (a difficult task in finance), it generally captures the relative ordering of stocks: better-performing stocks tend to receive higher predicted returns. This interpretation is reinforced by the positive Spearman rank correlation, showing that XGBoost performs meaningfully better than random chance when ranking stocks by expected one-month performance.

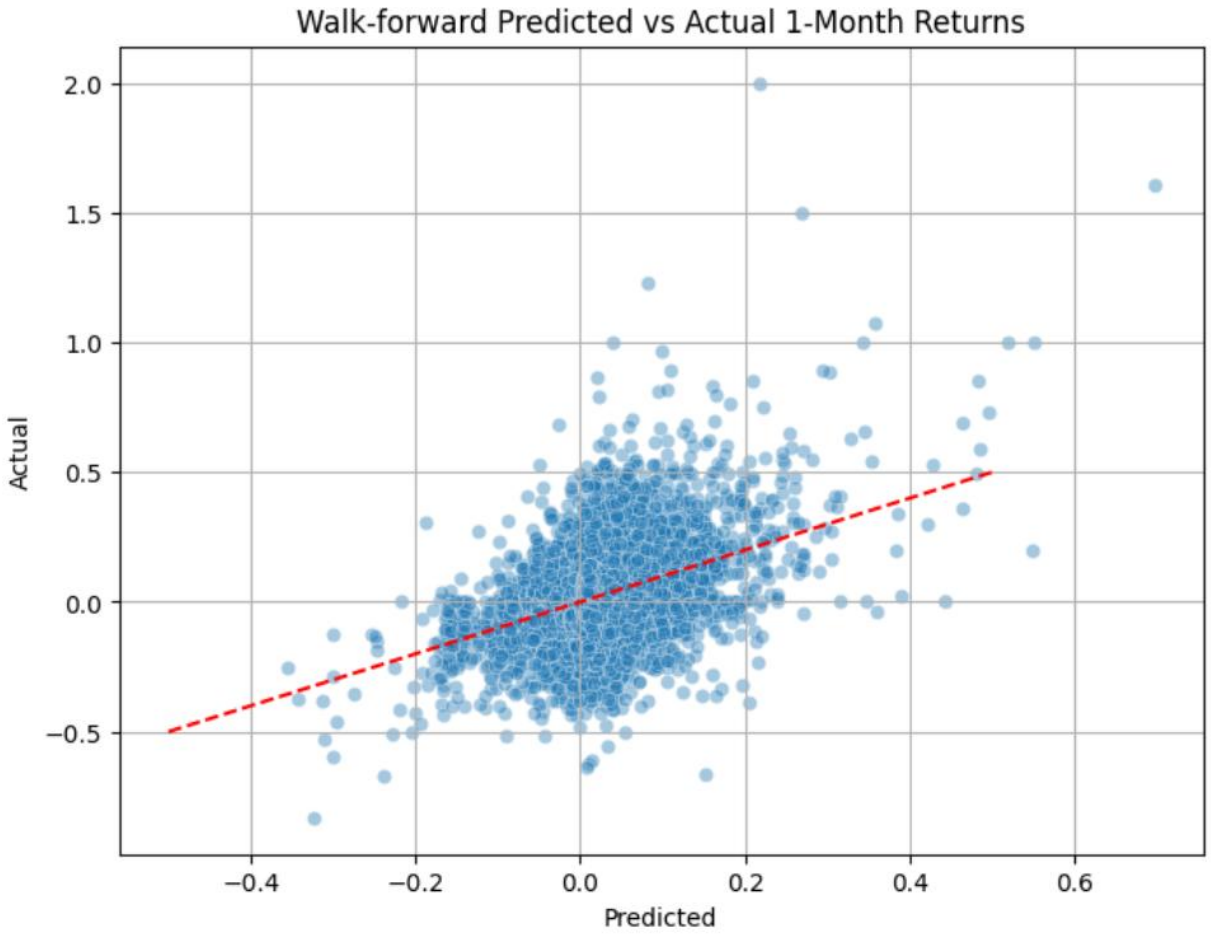
A second visualization (the Average Predicted vs. Average Actual Monthly Returns Over Time) adds another dimension of interpretability. Here, returns are averaged across all 100 stocks for each prediction month, smoothing out stock-specific noise. The resulting time-series plot shows that the model tracks broad market movements reasonably well. Periods in which the market rises, XGBoost typically predicts positive average returns; during downturns, the model lowers its predictions accordingly. The visual alignment between these two lines demonstrates that the model captures macro-level momentum and trend dynamics even though it is trained entirely cross-sectionally. This finding is important because it shows that XGBoost effectively incorporates rolling volatility, momentum shifts, and relative strength patterns, even though it is not explicitly told about market regimes.

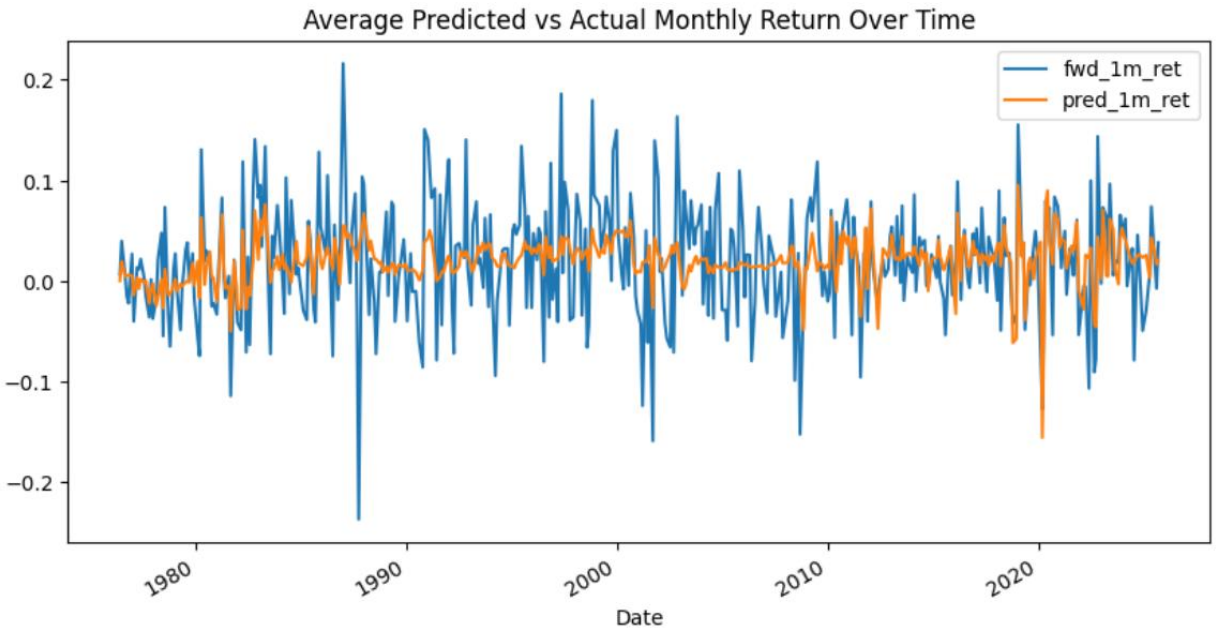




Incorporating FOMC sentiment into the model produced incremental but meaningful improvements. After merging sentiment onto the daily price data and forward-filling values between meetings, performance improved to an MAE of 0.0757, RMSE of 0.108, R^2 of 0.153, and Spearman rank correlation

of 0.282. The corresponding scatterplot with sentiment enabled shows somewhat tighter clustering around the diagonal line, especially in regions of moderate positive return. This improvement suggests that the sentiment signal helped the model adjust predictions upward or downward during months when macroeconomic tone influenced overall risk appetite. In other words, sentiment stabilized predictions during market stress and provided slight upward corrections when policymaker tone was unusually optimistic.





4.4 Sentiment Feature Findings

Sentiment analysis proved to be a valuable addition to predictive modeling, even when applied at a macroeconomic scale using FOMC meeting transcripts. When integrated into the XGBoost model, forward-filled sentiment scores provided a slowly changing market signal that helped adjust the expected returns of stocks in response to broader economic cues. Component analysis in Prophet also highlighted that sentiment had measurable effects on short-term DJIA movements, particularly around key economic events or statements. While sentiment alone does not fully explain stock return variability, it enhances model stability and improves relative ranking, demonstrating its utility as a complementary feature alongside traditional price-based indicators.

4.5 Comparative Summary of Model Performance

Across all models, we observed a clear progression of capability. ARIMA provided a simple baseline, capturing general trends but failing to respond to short-term fluctuations. Prophet improved on this by incorporating seasonality and sentiment, producing forecasts closely aligned with actual prices for a single index and offering interpretable component plots. XGBoost further expanded the scope by allowing cross-sectional forecasting of multiple stocks simultaneously, handling a richer set of features, and integrating sentiment as a macroeconomic regressor. Quantitative comparisons highlight that machine learning approaches, particularly XGBoost with sentiment, offer superior predictive accuracy and rank-based performance relative to classical time series models.

4.6 Recommended model

Based on the findings, XGBoost emerges as the recommended framework for multi-stock prediction due to its ability to handle numerous input features, predict multiple stocks simultaneously, and incorporate sentiment as an exogenous signal. While Prophet offers valuable insights for individual index forecasting and ARIMA remains a useful baseline for understanding historical trends, the scalability, flexibility, and moderate predictive performance of XGBoost make it the most practical choice for an applied stock forecasting system. Further improvements could be achieved by refining sentiment features, experimenting with additional macroeconomic indicators, or ensembling multiple machine learning models for increased robustness.

5. Conclusion

Overall, this project demonstrates that combining traditional price-based indicators with sentiment extracted from economic text (particularly FOMC statements) offers meaningful value in forecasting short-term stock performance. While baseline models such as ARIMA and Prophet provided useful structure for understanding long-term trends and seasonal patterns, the XGBoost walk-forward framework proved most effective at capturing nonlinear relationships between momentum, volatility, and macro-level sentiment. The construction of the FOMC sentiment dataset, powered by FinBERT and BERTopic, allowed us to translate complex policy language into quantitative signals that aligned well with market behavior across different economic regimes. Although sentiment did not perfectly predict returns, the models incorporating it generally produced more stable rankings and showed improved ability to anticipate directional shifts compared to price-only baselines. These results reinforce the growing evidence that textual analysis, when combined with careful feature engineering and realistic evaluation frameworks, can enhance financial forecasting and provide investors with richer decision-making tools.

References

Farka, M., & Fleissig, A. R. (2012). The effect of FOMC statements on asset prices. *International Review of Applied Economics, 26*(3), 387–416. https://doi.org/10.1080/02692171.2011.587111

Grootendorst, M. (2022). BERTopic: Neural topic modeling with a class-based TF-IDF procedure. *arXiv preprint*, arXiv:2203.05794. https://doi.org/10.48550/arXiv.2203.05794

Huang, A. H., Wang, H., & Yang, Y. (2023). FinBERT: A large language model for extracting information from financial text. *Contemporary Accounting Research, 40*(2), 806–841. https://doi.org/10.1111/1911-3846.12832

Jubinski, D., & Tomljanovich, M. (2017). Central bank actions and words: The intraday effects of FOMC policy communications on individual equity volatility and returns. *Financial Review, 52*(4), 701–724. https://doi.org/10.1111/fire.12143

Kim, J., Kim, H.-S., & Choi, S. (2023). Forecasting the S&P 500 index using mathematical-based sentiment analysis and deep learning models: A FinBERT transformer model and LSTM. *Axioms, 12*(9), 835. https://doi.org/10.3390/axioms12090835

Tang, W., Zhang, X., Huang, C., & He, J. (2024). Unlocking the power of the topic content in news headlines: BERTopic for predicting Chinese corporate bond defaults. *Finance Research Letters, 62*, 105062. https://doi.org/10.1016/j.frl.2024.105062

Team Contributions

Team Member Name	University ID / Email	Main Responsibilities & Contributions
Nagur Begum Shaik	540542084 Nagur@iastate.edu	Worked on formatting for the report, supported with researching sentiment analysis. Supported with writing the report.
Luiz Filippe Santana Adao	349749452 luizadao@iastate.edu	Cleaned and conducted the sentiment analysis of FOMC statements. Supported with writing the report.
Luke Auderer	980422501 auderer@iastate.edu	Led the end-to-end development of the forecasting system, implementing ARIMA, Prophet, and XGBoost models and rigorously evaluating their performance. Integrated FOMC sentiment into the modeling pipeline, enhancing predictive capability. Designed and built the website used to visualize model outputs and communicate insights. Supported with writing the report.
Kolben H. Thompson	496923476 kolben@iastate.edu	Gathered information for the website including diagrams for presentation. Supported with writing the report.