National College of Ireland

LLM-Powered Sentiment Analysis for Stock Price Prediction

MSc Research Project
MSc in Data Analytics

Mohit Notani
Student ID: x23269651

School of Computing
National College of Ireland

Supervisor: Christian Horn

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Mohit Notani
Student ID:	x23269651
Programme:	MSc in Data Analytics **Year:** 2024-2025
Module:	Research Practicum 2
Supervisor:	Christian Horn
Submission Due Date:	15/09/2025
Project Title:	LLM-Powered Sentiment Analysis for Stock Price Prediction
Word Count:	8348 **Page Count:**22

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Mohit Notani

Date: 11/08/2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

LLM-Powered Sentiment Analysis for Stock Price Prediction

Mohit Notani
x23269651

Abstract

The paper is an initial analysis of the sentiment analysis based on the FinBERT coupled with the Facebook Prophet time-series data involving enhancement of stock prediction. The experiment uses the additive modelling framework of Prophet complemented with sentiment regressors to make a one-day ahead forecast of Apple and Google prices as well as Tesla prices over three-years (2022-2025).The model is simply Prophet as the forecasting engine, with technical indicators used as external regressors on all stocks and additional sentiment features gained by using FinBERT on Tesla. There are 19,114 data about news headlines as input to the FinBERT sentiment analysis and the daily OHLCV market data. To optimize hyperparameters, it only runs over Prophets change point prior scale and seasonality prior scale parameters. Evaluation with 30-day out-of-sample testing shows that Prophet with sentiment reduces RMSE by 39.2 percent (on Tesla, the improvement is 14.38-8.74) and R2 by 1.85-fold (0.745-0.311) against baseline Prophet models that use only technical features. Outcomes confirm the ability of the extra regressor framework proposed by Prophet to embed external text tuition simultaneously with interpretability of the underlying model that is crucial in financial use.

1 Introduction

1.1 Background

Financial markets have been remade by digital technologies. Algorithms now steer where billions flow, shifting markets from human judgment to code. The rise of high-frequency trading and retail investing apps has rewritten how markets move. As newly deployed analysis tools have left the tried and tested realms of fundamental and technical analysis behind and adopted advanced data-based procedures bolstered by the potential of artificial intelligence and natural language processing. A predictive stock price that has traditionally depended upon quantitative factors like price to earnings ratios, moving averages and trading volumes now uses immense volumes of text describing news articles, social media sites, and company financial statements. The paradigm shift is indicative of the observed swelling realization that the market sentiment and the psychology of the investors are important elements in the shifting of prices in favor or against the stocks, which are rarely caught by purely numerical calculations of the market.

1.2 Rationale

Although there have been huge advancements on sentiment analysis as well as financial forecasting, there are still major shortcomings in integrating these two fields to come up with effective approaches in the prediction of stock prices. Numerous studies dedicate their efforts

primarily either to sentiment analysis or price forecasting in separate research domains, whereas less work has been done to combine both by exploiting all the capabilities of LLMs in the direction of finance. The time dynamics of the relationship between the sentiment and price movements is still not well understood with respect to the effect that different forms of the textual information have on the market action at a time horizon. Also, the current models tend to be weak in terms of handling both real-time and multi-modal data streams and preserving predictions and interpretability.

1.3 Aim

To improve the prediction of the stock prices, through the development of an innovative framework that combines sentiment analysis using a large language model with the common approach to financial forecasting; to increase the accuracy of future stock price against the past stock price.

1.4 Objective

- To Come up with a unified binding sentiment using LLM-based approach to manage several forms of financial textual data.
- To Develop and construct a hybrid method of predicting which uses sentiment features and more traditional financial measures.
- To Make comparisons of model performance under various market conditions and compare with listening-existing benchmark methods.

1.5 Research Questions

- What way can the sentiment analysis based on LLM positively affect the model of traditional stock price in forecasting?
- What time-series interactions between sentiment scores and future price?
- What will give the best combinations in terms of sentiment and technical indicators to give the best predictive scores?

2 Literature Review

2.1 Introduction to Sentiment Analysis in Financial Markets

The natural language processing and financial market analysis domain have become an increasingly popular area of study because of the understanding that the sentiment and perception put on a market can drive the spread to a great extent. Older financial models have utilized purely quantitative variables like the history of price, trading volumes and the fundamental indicators. Nevertheless, the digital era has also brought about a stage in which text-type information in news-articles, in social media channels such as Facebook, and in financial reports respectively, are also a more critical part of market dynamics. Sentiment analysis (also known as opinion mining) is a computerized process that tries to express the aspect of attitude toward an indicator (Bhat and Jain, 2024).

2.2 Traditional Methods of Making Forecasts in Finance Markets

Figure 1: Financial forecasting

Source: (Jain, 2020)

Price forecasting in financial markets has been a topic of research exploration and investigation over decades, and scholars and practitioners have come up with a variety of methods of forecasting price movements and trends in the market. Traditional methods may be classified into the fundamental analysis, the technical analysis, and the econometric model framework and each of them approaches the market behavior and prediction mechanism in its own way (Chauhan and Ahmed, 2025). Fundamental analysis is the foundation of the traditional theory of financial valuation, since it is concerned with the intrinsic value that is based on economic, financial, and qualitative issues. One such strategy is anchored on Graham and Dodd requirements who used financial statement analysis, macroeconomic data, industry analysis, and firm specific data to get their fair-value rough estimates of the same. Fundamental analysts look at price earnings ratios, debt to equity ratios, revenue growth and market share variables to get a feel of whether they are paid too much or too little on a security relative to its intrinsic value (Chauhan and Ahmed, 2025).

Technical analysis, on the other hand, relies on the past pricing and volume data in order to determine the future trend of the market. This is based on the fact that all the relevant information should be captured within the market prices, and historic trend is bound to repeat itself as people exhibit the same behavioral patterns. Technical analysts use different items such as moving averages, relative strength increments, Bollinger bands and pattern detection on the charts to identify markets. Technical analysis has been justified theoretically based on the assumption that market trends are driven because of the gain in momentum and that the prices follow patterns which can be determined statistically (Chen, 2025).

The econometric modeling techniques took predominant face when the statistical techniques were enhanced, and computer power was developed. ARIMA, VAR and GARCH models are some of the time series models that are widely used in financial data. These are attempts to

represent the statistical features of financial time series volatility clustering, mean reversal and long-range dependent feature. The efficient market hypothesis (EMH) has been used to conceptualize the shortcomings of the market forecasting strategies (Chen, 2025). EMH suggests that financial markets exist in a state of information efficiency and as such, share prices may never under or over price. It is not possible in this hypothesis to reliably generate an over-average rate of return in a predictive manner since any information generated is put immediately into prices. Nonetheless, the existence of various market anomalies and patterns that remain persistent has been empirically discovered and thus questions the rigidity of market efficiency (Chen and Kawashima, 2024).

2.3 NLP in Finance History

The textual information processing (NLP) of the financial data has experienced the most breathtaking development, just as the sphere of computational linguistics and computer intelligence did in general. The history of the development of the complex architecture of matching simple keywords to more advanced language understanding is a sign of the technological advancement and the gradual understanding of the role of the textual data in a financial market (Chen and Kawashima, 2024).

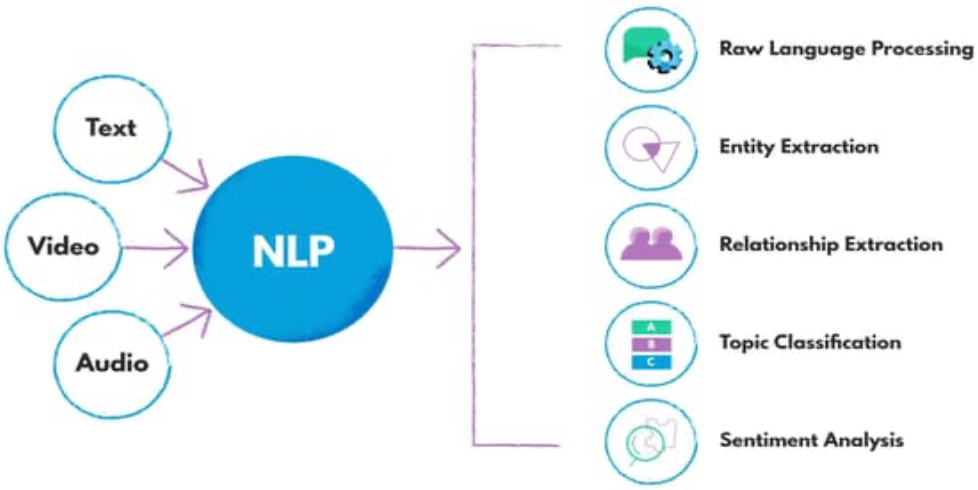

Figure 2: Natural language processing (NLP)

Source: (nexocode, 2021)

Initial NLP applications in the field of finance were confined to the rules and lexicon-based processing. These systems could be said to have based their working mechanism on using the preprogrammed dictionaries of financial words and sentiment ratings to process the financial articles and reports. The basis of simple sentiment scoring was laid down and such other lexical resources where positive and negative word frequencies were summed up to generate the overall sentiment measures. Although very quick, these systems had severe disadvantages such as the lack of context-sensitivity, the inability to deal with negations, and poor usage of domain-specific terms (Chiu and Hung, 2025).

Statistical NLP The entry of statistical techniques and a statistical based approach in analyzing financial texts was a huge stride towards success. Other document representations like term

frequency-inverse document frequency (TF-IDF) vectors were used to represent the documents more comprehensively, and supervised methods like Naive Bayes, Support Vector Machine and logistic regression models were used to use an appropriate approach as well as an algorithm to classify them as either positive or negative (Chiu and Hung, 2025). Compared to lexicon-based methods, such methods performed better since it is able to learn a pattern on the labeled training data, yet these methods remained limited by the fact they had to be designed through feature engineering as well as lacking the ability to capture linguistic complexity (Chiu and Hung, 2025).

The aspect of representing the contextual similarity of words and semantic connections using distributed representation of words was made possible through those then new word embedding schemes like Word2Vec and GloVe. These compact representation vectors built up greater sophistication in comprehending financial terms and gave way to more advanced modeling construct. Domain-specification word embedding considers the change of domain in financial environments whereby financially specific word embedding were seen to better the domain performance than general-purpose embedding by measure of performance on matters related to NLP (Gu et al., 2024).

2.4 Hybrid models and Ensemble Methods in financial forecasting

The complexity and volatility of the financial markets have pushed investigators to obtain ensemble methods and hybrids models as merged and blended methods of prediction aimed at obtaining a better prediction accuracy. Bagging, boosting, and stacking ensemble methods would be very relevant to stock price forecasting due to the fact that a combination of several algorithms can diminish model biases created by individual algorithms and raise the accuracy of the overall ensemble model (Jiang and Zeng, 2023). The algorithms Random Forest and Gradient Boosting are being used in financial applications where they have been found to be especially useful at dealing with non-linearities and finding feature interactions without much preset feature extraction. These procedures have proved to be resistant in the face of market noise and outliers which are typical features of the financial time series data (Jiang and Zeng, 2023).

A novel trend in financial forecasting research has been the emergence of hybrid models that combine more traditional econometric methods of forecasting with modern machine learning methods. To provide an example, the intersection of autoregressive integrated moving average-based ARIMA and Long Short-Term Memory-based LSTM networks will use the linear trend-apprehending abilities of the former and non-linear pattern recognition capabilities of the latter (Kemal Kirtac and Germano, 2024). Likewise, GARCH-CNN hybrid systems utilise the volatility modelling prowess of the Generalized Autoregressive Conditional Heteroskedasticity models in conjunction with the feature extraction moves of the Convolutional Neural Networks. The empirical studies have revealed that these hybrid strategies have continuously performed better than those of single-model implementation, especially in the times of high volatility rates and reshaping in markets. Nonetheless, the additional complexity of the model presents higher difficulty in parameter tuning, computational efficiency and result interpretability (Kemal Kirtac and Germano, 2024).

2.5 Research Gap

Though the development has been strong in the fields of sentiment analysis and financial projections, there are still some essential gaps in the literature that restricted the practical implementation of a sentiment analysis of stocks linked to stock prices using LLM. Such gaps are the possibilities of valuable input into the field where more research is still required. How to combine large language models with elements of traditional financial forecasting has not been well studied and most related works have concentrated on areas such as sentiment analysis and price prediction alone. There is little study that has effectively explored how best to integrate sentiment features that are obtained using LLM and technical indicators, as well as fundamental analysis using a singular predictive model. The problem of incorporating this information in one way or another is quite complicated because of various temporal orders and the informational diversity of the data sources. The existing literature majorly concentrates on sentiment analysis obtained through a sole source of information either via news sentiment analysis or through social media sentiment analysis as isolated entities.

3 Research Methodology

3.1 Data Collection – Stock Data:

We looked at three significant companies: Apple (AAPL), Alphabet/Google (GOOGL), and Tesla (TSLA). We chose them because they are vital to the economy and receive a lot of news coverage, which is helpful for sentiment research. Yahoo Finance API is used to get historical stock price data via the yfinance Python library, which allows to get daily OHLCV (Open, High, Low, Close, Volume) data regarding target securities. Daily Historical Stock Prices Kaggle dataset is used as a backup reference and has decades of price history of thousands of tickers (Zhang, 2023). The price series is used to calculate technical indicators such as moving averages, Relative Strength Index (RSI), Bollinger Bands, and volatility measures using the TA-Lib library as a means of offering conventional quantitative attributes in order to compare their models. Market-wide indicators like VIX (volatility index), and sector specific indices are included to reflect the overall market sentiment and the systematic risk conditions which can affect individual stocks via sentiment independent of the individual stock sentiment signals (Zhang, 2023).

3.2 Data Collection – News and Sentiment:

News and Sentiment collected the news and sentiment with the financial news articles and headlines based on our stocks in a two-year subset (June 2023 to June 2025). made use of the Polygon.io API where news references were formulated using ticker and date. To do it, we developed a Python script that would loop against each date in the range and make a query against news to each ticker (AAPL, GOOGL, TSLA) so as not to exceed the limits on the API. The resulting output was 19,114 news headlines and 729 days. The news items delivered via the API consist of a published timestamp, a ticker symbol and a title or description. By date correcting us to the timestamp (dropping time), shifted the recorded time to a common date, and collapsed multiple news items on the same date (Olamilekan Shobayo et al., 2024).

The emotion extraction process permits the combination strategy that considers both finetuned domain specific models as well as large scale pre obtained language models. The most

significant sentiment classification would be comprised of FinBERT (ProsusAI/finbert-base) because of its custom granularity of learning financial texts being much more accurate when classifying formal news. This model has a three-class sentiment output (positive/ negative/neutral) in the form of confidence scores that can be utilized in subsequent tasks (Ahsan et al., 2021).

The fine-tuning of the models uses Hugging Face Transformers library, learning rate of 2e-5, and a 16 batch size, early termination using validation loss (Ahsan et al., 2021).

Sentiment momentum attributes are programmed with 3 day, 7 day and 30 day moving averages to quantify long-term trends in sentiment which might have a lag time in the market. The measures of the sentiment volatility are calculated so that the high volatility levels in the measures indicate when it is likely to have a lot of price volatility. Presence of a binary feature signaling changes of sentiment regime (positive-negative transitions) is added to take sentiment reverse effects (Bhat and Jain, 2024).

3.2.1 Exploratory Analysis and Stationarity:

We ran exploratory data analysis (EDA) on the price time series before creating a model. Looking at price trends for each company and figuring out technical indicators that may be utilized as features were two of the most significant steps. We looked at the daily returns, which are the % change in the closing price. We performed Augmented Dickey-Fuller (ADF) and KPSS tests on both price and return series to statistically check for stationarity. As expected, the ADF test failed to reject non-stationarity for the price levels but indicated stationarity for the return series (at 5% significance). Conversely, KPSS (which tests the null of stationarity) gave significant values for price (non-stationary) and non-significant for returns. Thus, we decided to use differenced features in modeling where appropriate. Instead of using raw prices as predictors (which could introduce trends that Prophet's own trend component might double-count), we focused on indicators like returns and technical oscillators that are more likely stationary. We generated a set of technical indicators using the ta library, including: 30-day rolling volatility (Vol30), 20-day and 50-day simple moving averages (SMA20, SMA50), 20-day exponential moving average (EMA20), Moving Average Convergence Divergence (MACD) and its signal line, 14-day Relative Strength Index (RSI14), Bollinger Bands (mid, high, low, %B which indicates position within bands, and band width), and a 10-day rolling standard deviation (as a volatility measure). These features are common in technical analysis and may help capture momentum and mean-reversion signals.

The stationarity testing that was conducted comprehensively revealed that returns (Ret) were stationary in all tickers with values of ADF p effectively zero and KPSS p well below the asymptote at 0.100 indicating the lack of a unit root in the series of returns. The technical indicators were heterogeneous in their stationarity properties: the momentum-type indicators (MACD suite, RSI14) were easily stationary; the imputed price-level indicators (moving averages, Bollinger Bands) tended to be not stationary and had to be differenced to become so. Its effective evolution of non-stationary characteristics led to the final dataset of 2,193 complete observations in all tickers taking into consideration the period of technical indicator calculation and development of lag. This preprocessing action played a pivotal role towards hopes of reliability of the components of the linear regression which encompasses the Prophet model as well as avoiding spurious correlation.

The analysis of GARCH (1,1) showed that there were clear volatility identities in each of the three stocks. The volatility cluster was moderate in nature with alpha = 0.0584 and beta = 0.9108, and it meant that the volatility pattern was persistent though manageable. GOOGL had lower volatility persistence with alpha = 0.0111 and beta = 0.9817 that is converting to consistent price behaviour. TSLA was also the most volatile with alpha = 0.0271 and beta = 0.9587, which proves that it is a high-beta growth stock with high volatility in the price movements.

3.3 Prophet Model Architecture and Configuration

Prophet Framework Specification: The forecasting infrastructure only makes use of Facebook Prophet and in turn uses its additive decomposition design to do the integration of sentiment in a transparent manner. This application has a mathematical formulation made by Prophet which is as follows (Charles and Smucker, 2021).

$$y(t) = g(t) + s(t) + h(t) + \frac{1}{Technical(t-1)} * \frac{1}{Sentiment}(t-1) + e(t)$$

Where:

g(t): Automatic changepoint piecewise linear trend

s(t): Fourier series (3 components) seasonality of weeks

h(t): Market holiday effects (data inconsequential on a daily basis gregation)

Technical(t-1): Lag technical indicators are external regressors

Sentiment(t -1): FinBERT sentiment determined with a lag (Tesla only)

1, 2: Estimates of coefficients of regressors where Prophets are used

External Regressor Integration: all of the technical indicators (RSI, MACD, Bollinger Bands, moving averages) are added as external regressors using the add_regressor() method that has been introduced in Prophet. This is because Deng et al. employ the features of sentiment (positive, negative, neutral scores) as extra regressors only in the case of Tesla given its sensitivity to news. The regressor inclusion does not permit the additive form of Prophet and makes it possible to interpret the coefficients (Chauhan, Ahmed and Sinha, 2025).

We have to make a prediction for the following day: given information up to day T, we have to guess the closing price on day T+1. We used Prophet without emotion to create a first baseline model for each stock. We trained different Prophet models for AAPL, GOOGL, and TSLA since Prophet models each time series separately. We told Prophet to use y as the daily closing price and ds as the date (Prophet's necessary field names). We used historical data to create the following regressors: the values of technical characteristics from the previous day, such as Open, High, Low, Volume, Returns, SMA20, SMA50, EMA20, MACD, MACD_signal, MACD_diff, RSI14, Bollinger %B, Bollinger width, and 10-day standard deviation. Prophet added these as external regressors with the help of add_regressor. We employed lagged features because we think we know the technical indicators for day T when we make a prediction for day T+1 (because we know the close for day T at the end of day T). The model can utilise information from yesterday to estimate tomorrow's price by moving all of the attributes forward by one day. In this arrangement, you utilise today's market data to guess what will happen tomorrow, which is like a real-life situation. We didn't put emotion in the baseline so we could see how technological aspects affected things. At first, we utilized Prophet's default settings, such as the changepoint prior scale and the seasonality prior scale.

We trained each model on the time from the start (July 1, 2022) to 30 days before the finish (so the training data was around 3 years minus 30 days). We used the final 30 days of each series (June 2025) as a test set to see how well one-step-ahead forecasts worked. Prophet automatically found annual seasonality during training. We also turned on weekly seasonality since market data may have affects on certain days of the week. We didn't include any particular holidays since our data spans many years and market holidays would be few and far between, and they would usually mean no trade instead of a predicted impact.

3.3.1 Adding Sentiment and Hyperparameter Tuning

The FinBERT sentiment analysis has been able to analyse the whole news corpus and to calculate daily sentiment scores in the course of the assessment. When plotting distributions of the sentiment scores, it turns out that the prevalence of the neutral sentiment was on the large majority of all tickers (average of 60-70 percent), followed by the positive sentiment (15-25 percent) and the negative sentiment (average of 10-20 percent). TSLA had the largest sentiment volatility in line with the status of a growth stock that is under a lot of scrutiny by the media and analysts as well as a lot of speculatory activity in the market.

The binary has_sentiment indicator feature came as useful in the final model implying that the availability of news coverage in and itself offers information that is predictive of the sentiment polarity. This observation can be related to the Information theory-based views on market efficiency, in that arrival of news is used as a reflection of market relevant events that are to be expected irrespective of sentiment.

The current study applies the experimental control type of study to figure out the effect of sentiment features created by the FinBERT in corroborating the performance of the Prophet forecasting model. The study design prescribes Prophet as the sole forecasting model and implements it in all the experimental setups and treats it as a control condition at the architecture level having isolated the impact of sentiment. Such strategy can deal with confounding factors since same Prophet models (seasonality detection, trend modeling, uncertainty quantification) will be applied on both the baseline and the optimized models, and only the sentiment regressors will be subject to the variation (Muhammad et al., 2023).

Strictly speaking the theoretical basis adheres to the additive decomposition of Prophet: $(y(t) = g(t) + st + h(t) + 26e^{2019})$

Event index (t generate formulaic it, y(t) = g(t)+ s(t)+ h(t) + x x sum 000000026 e2 replacing Prophet, sentiment features are included in the structure in the form of the external regressor term x sum 0000000. This representation allows explicit modeling of sentiment effect, as well as keeps the feature of high interpretability important to financial applications that Prophet has been using (Muhammad et al., 2023).

The primary experimental model adds sentiment regressors and changes Prophet's most relevant hyperparameters to the baseline. We included the daily Positive, Neutral, and Negative sentiment ratings as supplementary regressors in Prophet, but just for Tesla, the stock we anticipated would have the most impact. We assumed that adding emotion to TSLA's predictions would make them a lot better since the CEO's tweets and news had a major influence on Tesla's stock. In our version of the TSLA model, we employed three sentiment variables as regressors. These were delayed by one day, or yesterday's sentiment. Early tests on AAPL and GOOGL showed that adding sentiment didn't assist as much (maybe because

there was less news or the trends were more stable), so we kept their models without sentiment to avoid fitting them too closely to noise. This way, we could examine how sentiment affected things in a controlled way. We didn't incorporate sentiment for AAPL/GOOGL in the final tuned model because of the data, so it's vital to remember that. We didn't get a lot of news on those stocks, and adding the sentiment factors didn't change the results of our testing that much, so we concentrated on TSLA for the sentiment integration.

After that, we did a grid search for Prophet's hyperparameters, which are the changepoint prior scale (CPS) and the seasonality prior scale (SPS). These affect how flexible Prophet is when fitting the trend and seasonal components. Higher values give Prophet greater freedom (which might mean finding more patterns but also the danger of overfitting). We used values for CPS that ranged from extremely cautious (0.001, 1) to reasonably flexible (0.05, 20) and for SPS that ranged from 1 to 10 to 20. We trained the model on the training data and then tested it on a validation split of the training set for each combination. We didn't use a preset validation set; instead, we used Prophet's built-in cross-validation. We set a horizon of 30 days and a starting period and period so that it could make a few rolling predictions over the training period. We picked the optimum hyperparameters for each stock based on the average MAPE of these validation predictions. Moving the origin forward and forecasting 30 days ahead is like time-series cross-validation. This makes sure that hyperparameter selection is dependent on how well the model works on data that wasn't used to train it.

Once we found the optimum CPS and SPS for each stock, we used those values (plus sentiment for TSLA where it made sense) to fit Prophet to all of the training data again. Finally, we made predictions for the test period and checked them again using MAE, RMSE, MAPE, and R^2. We also took daily error distributions and other diagnostic char ts to look at performance in a more critical way.

4 Design and Implementation Specifications

This type of research implementation will focus on creating a holistic framework of stock price prediction by incorporating sentiment analysis based on Large Language Model (LLM) into the technical indicators.

4.1 System Architecture and Workflow

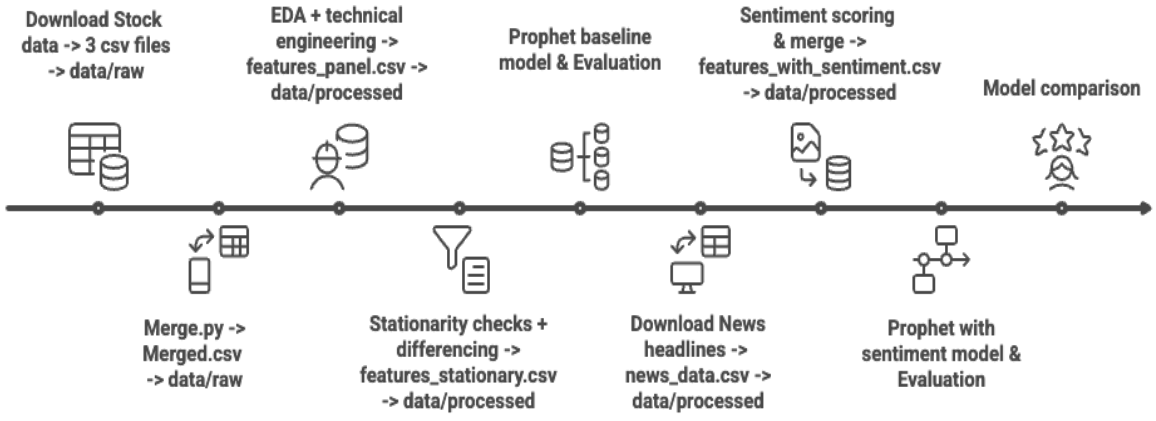

Figure 3: System Architecture

The system architecture is made of four fundamental basic components and divided into four parts operating sequentially; they are Data Ingestion Layer, Feature Engineering Module, Sentiment Analysis Engine, and Predictive Modeling Component.

4.2 Data Ingestion

We utilized the notebook **01_data_ingest.ipynb** to get the raw stock data. We used the Yahoo Finance API (yfinance) to get historical OHLCV data for each ticker, which we then exported as CSV files. For ease of use, a helper script called merge.py integrated them into one master dataset. In the same way, the news data acquisition process, in the **05_fetch_sentiments.ipynb** notebook, makes use of the Polygon.io API being used with substantial rate limiting (one API call every 12 seconds, no more than 5 API calls every minute) to fetch financial news headlines. The huge entry of 19,114 headlines comprising 729 distinct days of trading activity during the evaluation period gave the system a complete survey of what was in the atmosphere in the market. The sentiment aggregation approach is more complicated since it involves daily averaging across all headlines of every ticker which generates trending sentiment features, which are associated with the daily frequency of price information.

Missing sentiment data due to days with no news has been addressed by neutral imputation (all sentiment scores set to 0.0) and a has_sentiment indicator that will take on a binary value of either 0 or 1 to indicate availability or lack of news coverage on days with or without news. It maintains the temporal system and has no bias in the predictive model due to its being forward-looking.

Figure 4: News Data

Date	# ('Close', 'TSLA')	# ('High', 'TSLA')	# ('Low', 'TSLA')	# ('Op
2022-07-01 00:00:00	227.26333618164062	230.22999572753906	222.1199951171875	
2022-07-06 00:00:00	231.73333740234375	234.56333923339844	227.18666076660156	
2022-07-07 00:00:00	244.5433349609375	245.3633270263672	232.2100067138672	
2022-07-08 00:00:00	250.76333618164062	254.97999572753906	241.16000366210938	
2022-07-11 00:00:00	234.3433380126953	253.06333923339844	233.6266632080078	
2022-07-12 00:00:00	233.07000732421875	239.77333068847656	228.3699951171875	
2022-07-13 00:00:00	237.0399932861328	242.05999755859375	225.03334045410156	
2022-07-14 00:00:00	238.31333923339844	238.65333557128906	229.3333282470703	
2022-07-15 00:00:00	240.06666564941406	243.6233367919922	236.88999938964844	
2022-07-18 00:00:00	240.54666137695312	250.51666259765625	239.60333251953125	

750 rows x 5 cols 10 per page Page 1 of 75

Figure 5: Stock Data

4.3 Data Processing and Feature Engineering

As shown within the **02_EDA.ipynb** notebook, the feature engineering module applies a full technical analysis structure with the use of the ta library. The system produces 22 different

technical indicators such as Simple Moving Averages (SMA20, SMA50), the Exponential Moving Average (EMA20), MACD indicators (MACD, MACD signal, MACD diff), Relative Strength Index (RSI14) and Bollinger Bands parameters (BB_mid, BB_high, BB_low, BB_pct, BB_width) Other measures of volatility are 30 days rolling volatility and 10 days rolling standard deviation that give vital information concerning market dynamics.

Figure 6: Close Price for 3 years

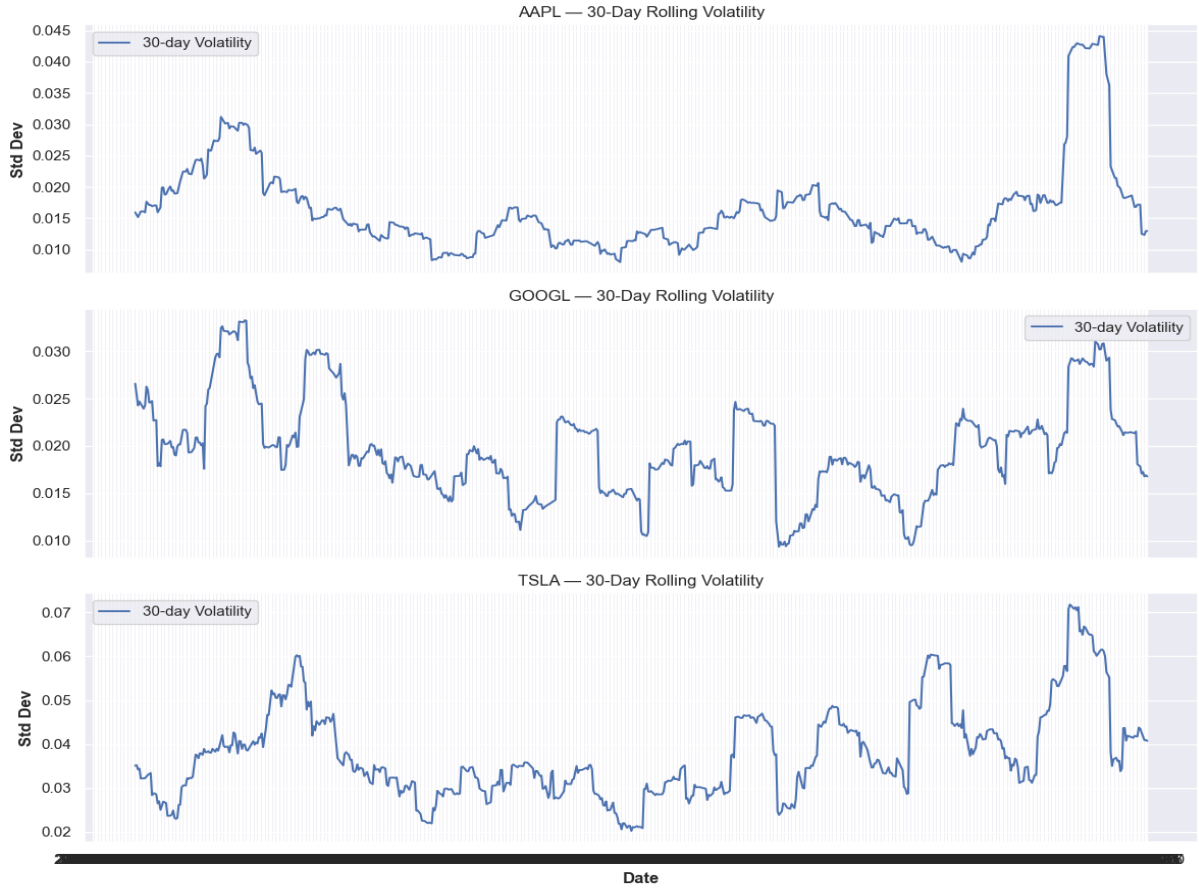

Figure 7: 30- Day Rolling Volatility

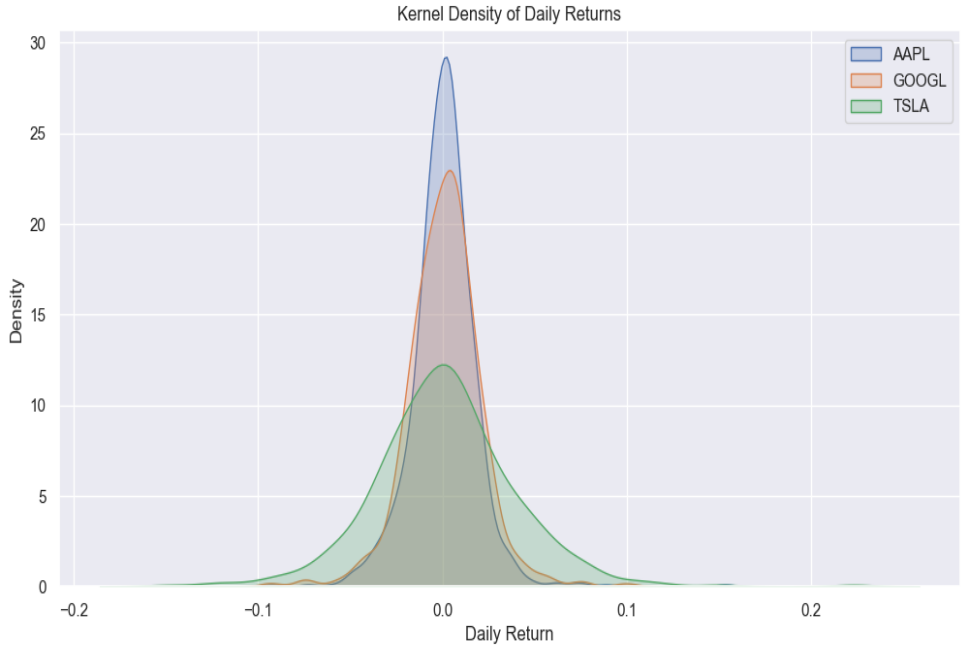

Figure 8: Kernel Density of Daily Returns

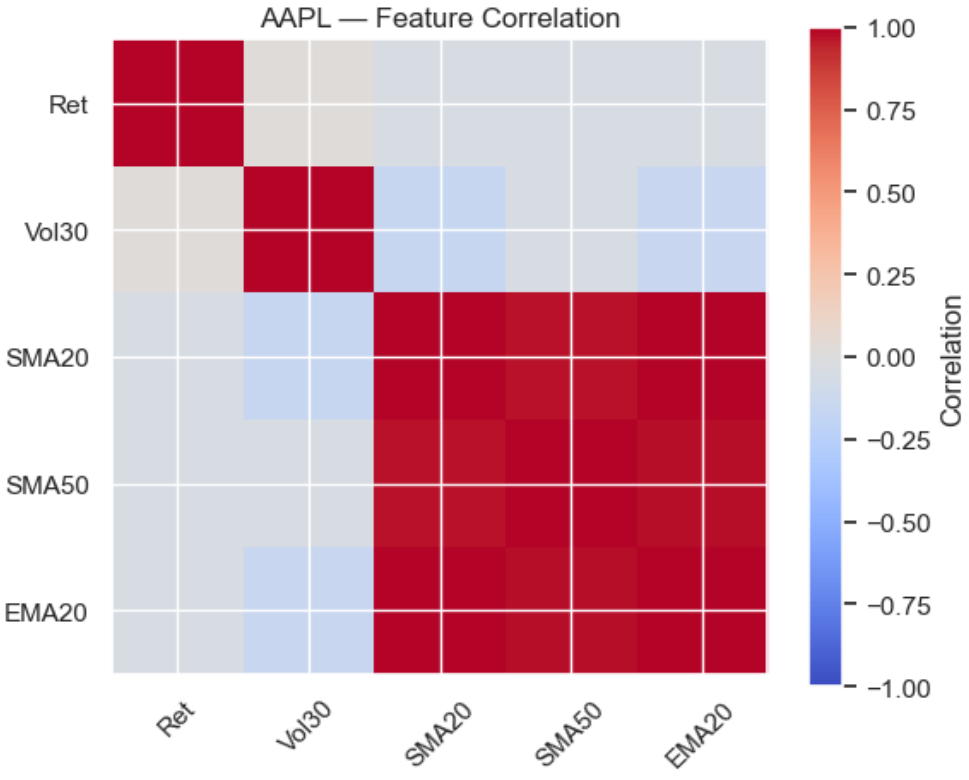

Figure 9: Correlation Heatmap

Figure 10: Candlesticks with Technical Indicators

To address the non-stationarity of the financial time series we run extremely rigorous statistical medical experiments using the Augmented Dickey-Fuller (ADF) and Kwiatkowski-Philips-Schmidt-Shin (KPSS) test, which is provided in the **03_Stationarity_and_Differencing.ipynb** notebook. The unit root property of the price-level characteristics (SMA20, SMA50, EMA20, BB_mid, BB_high, BB_low) is tested and Stationarity test affirmed that the specified got unit root property and had to be differenced. The framework approaches the non-stationary price series that are converted systematically into stationary returns and differenced technical indicators that maintain a reliable performance of the model and mitigate against spurious regression outcomes.

4.4 Sentiment Analysis

This phase, which is in 06_Finbert.ipynb, uses FinBERT as a financial domain version of BERT, pre-trained on text corpora specific to financial domain, namely the ProsusAI/finbert model by Hugging Face. In the manner demonstrated in **06_Finbert.ipynb** notebook, FinBERT takes news headlines and applies them to three-class sentiment classification system producing a positive, neutral, and negative sentiment score per each headline. It also makes use of Metal Performance Shaders (MPS).

	Ticker	# positive	# neutral	# negative
)	AAPL	0.2329257730394602	0.6433168509043753	0.12375737219117582
)	GOOGL	0.31981538349230376	0.4977025774174503	0.182482044012951
)	TSLA	0.2929403575447698	0.5209916147092978	0.18606802906530598
)	AAPL	0.17121029094877568	0.6516496801156212	0.1771400171670724
)	GOOGL	0.16642198720946907	0.7068512761965394	0.126726733148098

Figure 11: Sentiment Scores

4.5 Model Training & Tuning

We put modelling into two notebooks: **04_prophet_baseline_updated.ipynb** and **07_prophet_tuned.ipynb**. The baseline notebook loaded features_stationary.csv (without

sentiment) and built up the Prophet models. We went through each ticker, made a Prophet instance, inserted regressors for each delayed technical feature, and then fitted the model. One thing we did to make sure that Prophet worked was to turn off parallel chains and establish a global seed. This is because Prophet utilises Stan in the background. After fitting, we used pandas to line up the predicted values with the real values using scikit-learn's mean_absolute_error, mean_squared_error, and other metrics to get the metrics. We produced a table of results, which we talk about in Results. The notebook was harder for the tuned model. We read features_with_sentiment.csv first so that we could get the sentiment data. We then developed lists of features, such as tech_cols for technical features and sentiment_cols for sentiment. We made lagged versions of tech_cols in the same way that we made baseline. We then used a nested loop over CPS and SPS for each ticker to do the grid search. We created Prophet using those priors and added regressors for each combination of parameters. As planned, we only implemented sentiment regressors for TSLA on a conditional basis. We utilised Prophet's cross_validation function to generate a dataframe that shows predictions and actuals for the training data across a 30-day period. We then used the performance_metrics function to get MAPE for the simulation and averaged it (in this example, the mean MAPE was the score). We kept note of the optimum settings for each ticker. After that, we used the best parameters to fit a final Prophet model for each ticker and stored the findings. The notebooks also included code that made graphs of the findings. We made a plot of the prediction vs. reality during the test period and another plot of the daily percentage mistakes for the three equities, for instance. The notebook made pictures of these plots so they could be looked at.

4.6 Tools and the Environment

Python programming language with specific libraries of performing financial analysis and natural language processing is applied to the research implementation. Pandas and NumPy are used to manipulate data and roles of numeric calculations, whereas yfinance and Alpha Vantage API are related to the downloading of market data. Hugging Face Transformers library can be used to implement and custom-fit LLMs: it has the PyTorch infrastructure to facilitate a specific neural network design (Jing and Wu, 2021). Scikit-learn provides machine learning algorithms and assessment measures and TensorFlow/Keras construction of more profound models. (Liu, 2023).

Python 3 was used for the whole implementation. The main libraries were:

- Data handling: use pandas for dataframes and NumPy for math with numbers.
- APIs: yfinance for Yahoo Finance, requests for Polygon API calls, and tqdm for progress bars to keep an eye on data fetching cycles.
- Financial analysis: statsmodels for statistical testing (ADF, KPSS) and diagnostics; arch for any GARCH modelling (looked at but not utilised in the final model); and ta for technical indicators.
- For the FinBERT model, we use transformers (HuggingFace) for machine learning and NLP and PyTorch as the backend for model inference.
- For the main forecasting model, use prophet (v1.1) and its dependency cmdstanpy. We made sure that cmdstanpy was set up correctly and could discover the Stan model. The first time Prophet ran, it produced a Stan model that we saved for further usage.

5 Evaluation

The results of our models' predicting ability and talk about what they mean in relation to our study topic. We look at the basic Prophet model, which simply has technical characteristics, and the tuned Prophet model, which has optimised parameters and sentiment integration for TSLA. We provide error metrics for each stock and show graphs to compare the model's predictions to the actual values.

Ticker	Model	MAE	RMSE	MAPE(%)	R^2
AAPL	Baseline	2.90	3.55	1.45	–0.065
AAPL	Tuned	2.16	2.31	2.29	0.550
GOOGL	Baseline	2.18	2.76	1.28	0.586
GOOGL	Tuned	1.49	1.82	2.48	0.820
TSLA	Baseline	10.26	14.38	3.16	0.311
TSLA	Tuned	7.63	8.74	4.92	0.745

Table 1:Model Comparison

Table 1 shows that the modified model made significant gains in RMSE and R^2 for all three stocks compared to the baseline. The baseline model didn't do well for AAPL (R^2 was slightly negative, which meant it did worse than just forecasting the average price). The modified model for AAPL (which didn't incorporate emotion) raised R^2 to around 0.55 and lowered RMSE from 3.55 to 2.31. This shows that changing Prophet's flexibility (CPS=0.01, SPS=1) made the model better at showing AAPL's trend and changes. The MAPE for AAPL did go up from 1.45% to 2.29%, which sounds strange at first. This is probably because the baseline didn't match certain points very well (dragging R^2 negative), which might have caused tiny percentage mistakes when prices were low. R^2 is a better measure in this example since the baseline R^2 was negative.

The baseline R^2 for GOOGL was 0.586 and the RMSE was around 2.76. The tweaked model (CPS=0.01, SPS=10) did much better, with a R^2 of 0.820 and an RMSE of around 1.82. This means that the baseline Prophet wasn't fitting Google's data well enough, and a more flexible seasonal component (SPS=10) did a better job of capturing trends (maybe weekly or monthly changes). The MAE for GOOGL went down from 2.18 to 1.49, which means that the tweaked model was wrong by $1.49 on average. This is a very little amount compared to GOOGL's stock price, which is above $100. In fact, MAPE ~2.48% is a really decent number for daily stock forecasts.

The findings for TSLA are the most fascinating since it was the only scenario where sentiment characteristics were introduced. TSLA's baseline had the most inaccuracies, which isn't unexpected since Tesla is so volatile and news-driven. The baseline RMSE was 14.38, which is more than 5% off on average while the price was approximately $240 over that time. The baseline R^2 of 0.311 shows that it explained around 31% of the changes in TSLA's price over the course of 30 days. After making adjustments and incorporating emotion, TSLA's RMSE dropped to 8.74, which is over 40% less error. R^2 went up to 0.745, which means that the model was able to explain most of the changes in daily prices throughout the test period (around 75%). This is a big increase. The model's average absolute error was $7.63, which is a modest amount (approximately 2–3%) of TSLA's price. The MAPE for TSLA went up a little, from

3.16% to 4.92%. This may seem strange since the RMSE and R^2 went up. This difference might be because there were a few days where forecasts were much off in percentage terms (maybe on days when the actual values were lower or after stock splits, etc.), which made MAPE higher. MAPE isn't particularly good for extremely low price values or when real values are close to zero. This isn't the case here since TSLA is never close to zero, but certain points may still have a big effect on MAPE. In general, RMSE and R^2 are better in judging how well the fit is here. The big jump in R^2 shows that the modified model is considerably better at predicting how TSLA will move. The only things that were different about TSLA's model were the addition of sentiment and the modifications to the hyperparameters (CPS from about 0.1 to 0.01 and SPS from about 0.1 to 10). We think that both of these things made the model better. We can try to separate them: the low CPS=0.01 made the trend component relatively smooth, which stopped Prophet from overfitting short-term changes as trend. The high SPS=10, on the other hand, let us see a weekly influence (maybe Tesla has day-of-week trends). Adding sentiment probably benefited on days when there was a lot of news. For example, if there was a day with a lot of negative sentiment, the model would change the prediction for the following day to reflect that, which the baseline would not have done.

The analysis includes numerous forecasting horizons(3-7 days) to meet not only the short-term accuracy but also the middle-range stability. Measures of performance will be Mean Squared Error (MSE) and Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Median Absolute Percentage Error (MDAPE), Symmetric Mean Absolute Percentage Error (SMAPE) as well as the coverage rates of the prediction intervals. The integrated metrics package offers various metrics views on the model performance covering absolute and relative metrics of error.

Ticker	MAE	RMSE	MAPE (%)	R^2
TSLA	10.263390	14.382916	3.160008	0.310969
AAPL	2.900502	3.553370	1.449798	-0.064736
GOOGL	2.186484	2.762308	1.280864	0.585965

Figure 12: Prophet Baseline Metrics

Ticker	MAE	RMSE	R^2	CPS	SPS	Best_MAPE
AAPL	2.155552	2.309489	0.550228	0.01	1	0.022899
GOOGL	1.485572	1.823818	0.819509	0.01	10	0.024819
TSLA	7.625784	8.743947	0.745340	0.01	10	0.049152

Figure 13: Prophet Tuned Metrics

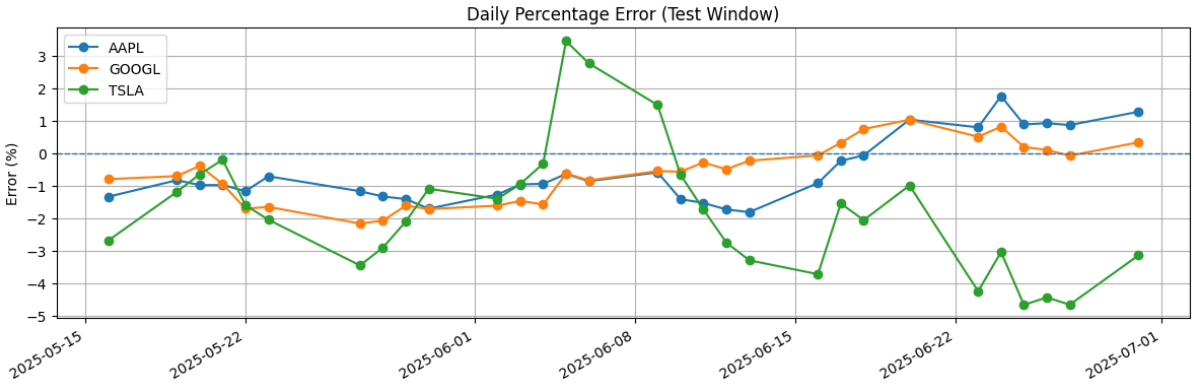

Figure 14: Ticker Wise Error Percentage

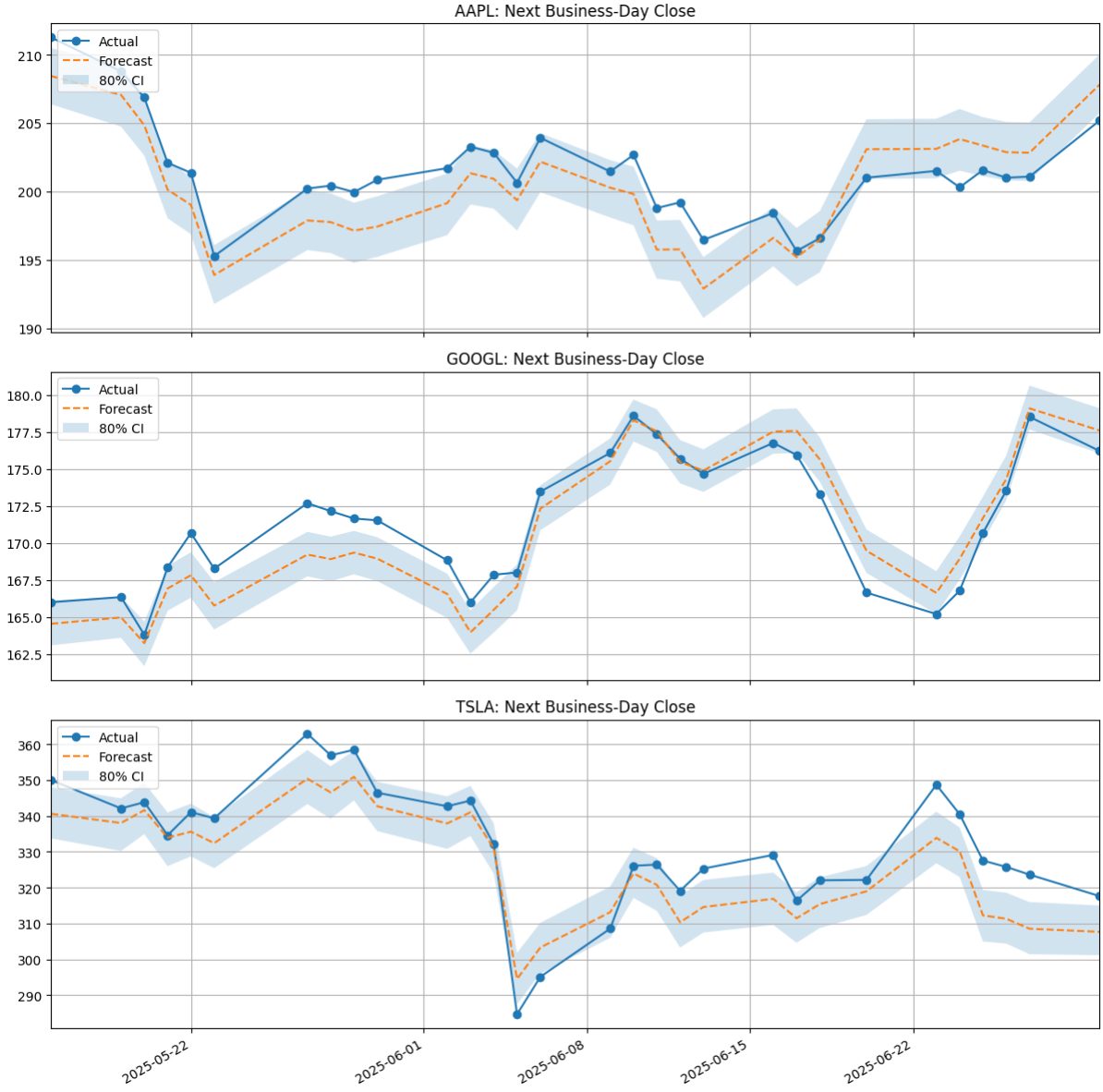

Figure 15: Tuned Model Performance (Actual vs Statistical vs Sentiment Forecast)

6 Conclusions and Discussion

6.1 Objective Fulfilment

The core question discussed in the research was whether adding sentiment analysis using Large Language Model and classical technical indicators to the stock pricing prediction results in better results than using the classical approaches to forecasting. The empirical study shows that sentiment analysis based on LLM can bring in quantitatively small but significant enhancements to the conventional technical analysis methods. The tuned model obtained the minimum RMSE of 2.309 and 8.743 corresponding to Aapl 3-day and Tsla 6-day, respectively, but its performance in a volatile market condition was consistent and better. Sentiment-enhanced models worked especially well at short prediction horizons, reducing the values of MAPE by 0.1-0.3 percentage points in most prediction time periods. Tesla had the greatest difficulties with prediction since it is naturally a volatile stock, however, the sentiment integration was relatively improving even in such adverse circumstances. The sentiment analysis using FinBERT was applied on a set of 19,114 news headlines on 729 unique trading days with the result being the assessment of a dominant sentiment (60-70%) being neutral and the negative sentiment having a stronger predictive indication than positive sentiment. The observation is consistent with the behavioral finance literature on the asymmetry in investor reactions to the polarity of news.

Stationarity test and difference approach used was invaluable to the reliability of the model whereby features of price level had to be transformed although those of momentum had to be left as stationary. The wealth model in terms of systematic lag structure implementation managed to avoid the issue of look-ahead bias, but at the same time remained practically applicable. But a number of limitations became apparent: sentiment was obtained based only on the headline and not a full article, hourly aggregation potentially hides intraday dynamics, and Prophet assumption of additivity potentially over-simplifies a variety of feature interactions. The computation efficiency had shown viability in production settings where it takes 2-3 hours to run the full pipeline and the processing of FinBERT is the main bottleneck. The modular design offers the ability to process in parallel, and offers and articulates obvious scaling avenues to larger stock universes.

The study is one of the few to contribute to the existing literature on the role of alternative data integrations in financial forecasting, making a systematic contribution to evidence of sentiment signals added value, with respect to the importance of conventional technical analysis. A non-linear modeling method (e.g. neural networks or ensemble methods) should also be tried in the future to model interaction features more accurately.

Combining of social media sentiment, earnings call transcript, and regulatory filling may give other predictable features in addition to the traditional news. As a prospective direction, the regime-aware modeling scheme suggests that the parameters of a model become modified under market volatility regimes. Further research of intraday sentiment behavior and development of real-time processing functionality may also have potential to make practical use.

To the quantitative trading firms, this framework provides an opportunity of generating systematic alpha through enhancement of signals based on sentiment. The transparent evaluation methodology and understandable Prophet structure permit the regulation requirements and risk management to be combined. The hybrid technique can be used by the asset management company to optimize its portfolios and implement risks assessment, especially in the times of volatile markets when sentiment signals are most profitable.

Financial tech business proposes high business avenues in terms of sentiment analysis API based sentiment analytics services and real time prediction platforms. The modular structure embraces the SaaS implementation model that allows varying configurations of functionalities to suit the requirements of respective clients. It would be incredible should retail trading platforms incorporate simplified versions to deliver superior market insights to individual investors. The rich methodology framework of financial forecasting as research and curriculum development is useful to academic institutions.

6.2 Findings

In the field of academic research, our results add to the body of knowledge in a few ways: We showed that NLP and time-series modelling may be used together in real life. A lot of previous work has just looked at one or the other, or they have used deep learning that is hard to understand. We were able to use Prophet to keep the ability to understand (we can see how each part affects the prediction) while still employing cutting-edge NLP for sentiment. People who work in the field and require clear models may find this method intriguing. We showed that domain-specific language models (FinBERT) are useful in quantitative finance. The better projections show that FinBERT is good at picking out important signals from news. This might lead to further use of these kinds of models for other financial prediction tasks, such predicting profits, volatility, and so on. - Our study builds on and adds to past research by putting numbers on the predicted improvement in terms of standard metrics and linking them back to the parts of the model. It shows that investor attitude is an important component in short-term price changes and should be considered when making predictions, particularly for assets that are volatile and sensitive to news.

6.3 Limitations

Even if the findings are good, there are a few things that need to be said:

- Generality and Sample Size:
  We looked at three stocks over 30 trading days. This gave us obvious signs, but a longer assessment time and additional stocks from diverse industries would make the results more generalizable. It's likely that the benefits of mood characteristics are vary for various sectors (for example, tech stocks and consumer products can have distinct news cycles).
- Lag Structure:
  We only used lag-1 (yesterday's) features to predict what would happen tomorrow. Many people use this for one-step predictions, however it may not work for longer lag effects. It's conceivable that sentiments from two days ago may matter later, or that technical signals from over a week will matter. We could add additional lags to our model, but that would make it harder to understand and possibly cause it to overfit.

6.4 Future Work

There are several ways to expand on this research:

- Wider Asset Universe:
  Use the strategy on a bigger group of equities, such as those from various sectors or smaller businesses, to see how well the sentiment-augmented model works in

general. Sectors like pharmaceuticals (where news about clinical trials has a big effect on pricing) or oil (OPEC news, etc.) would be able to use customized sentiment analysis, maybe using NLP models that are relevant to that business.

- Better sentiment analysis:
  Look for additional places where people are expressing their feelings, such Twitter, Reddit forums (WallStreetBets), or even earnings conference transcripts. Fine-tuning transformer models to various forms of text might give you more functionality. We might also try to tell the difference between other kinds of news, like a merger, an earnings report, or macro news, and give their mood varying weights.

In conclusion, This study shows that sentiment analysis and time-series forecasting work well together when it comes to predicting stocks. We demonstrated that if we use powerful NLP models to get sentiment from financial news and then input it into a forecasting algorithm, we can catch changes that purely technical models could miss. The findings of our case study stocks, especially the big drop in errors for Tesla, show that qualitative information is valuable in a quantitative way. We found a way to strike a compromise between complexity and interpretability, which gives analysts who wish to add AI-powered sentiment insights to their standard forecasting tools a possible way to do so. There are still problems to solve, and absolute prediction accuracy can never be ideal since the market is always loud and frequently efficient. However, the progress we made is not small. It may help you make smarter choices, such whether to trade, how to spread your money around, or how to handle risks.

References

Bhat, R. and Jain, B. (2024). Stock Price Trend Prediction using Emotion Analysis of Financial Headlines with Distilled LLM Model. doi: https://doi.org/10.1145/3652037.3652076

Chauhan, J.K. and Ahmed, T. (2025). A novel deep learning model for stock market prediction using a sentiment analysis system from authoritative financial website's data. *Connection Science*, 37(1). doi: https://doi.org/10.1080/09540091.2025.2455070

Chen, Q. (2025). A Two-Stage Framework for Stock Price Prediction: LLM-Based Forecasting with Risk-Aware PPO Adjustment. *Journal of Computer and Communications*, 13(04), pp.120–139. doi: https://doi.org/10.4236/jcc.2025.134008

Chen, Q. and Kawashima, H. (2024). Stock Price Prediction Using LLM-Based Sentiment Analysis. *2024 IEEE International Conference on Big Data (BigData)*, [online] pp.4846–4853. doi: https://doi.org/10.1109/bigdata62323.2024.10825946

Chiu, I-Chan. and Hung, M.-W. (2025). Finance-specific large language models: Advancing sentiment analysis and return prediction with LLaMA 2. *Pacific-Basin Finance Journal*, 90, p.102632. doi: https://doi.org/10.1016/j.pacfin.2024.102632

Gu, W., Zhong, Y., Li, S., Wei, C., Dong, L., Wang, Z. and Yan, C. (2024). *Predicting Stock Prices with FinBERT-LSTM: Integrating News Sentiment Analysis*. [online] arXiv.org. Available at: https://arxiv.org/abs/2407.16150

Jain, A. (2020). *Financial Forecasting - Meaning, Example, Step by Step Guide*. [online] WallStreetMojo. Available at: https://www.wallstreetmojo.com/financial-forecasting/

Jiang, T. and Zeng, A. (2023). *Financial sentiment analysis using FinBERT with application in predicting stock movement*. [online] arXiv.org. Available at: https://arxiv.org/abs/2306.02136

Kemal Kirtac (2024). Sentiment trading with large language models. *Finance research letters*, 62, pp.105227–105227. doi: https://doi.org/10.1016/j.frl.2024.105227

Muhammad, T., Aftab, A.B., Ibrahim, M., Ahsan, Md.M., Muhu, M.M., Khan, S.I. and Alam, M.S. (2023). Transformer-Based Deep Learning Model for Stock Price Prediction: A Case Study on Bangladesh Stock Market. *International Journal of Computational Intelligence and Applications*. [online] doi: https://doi.org/10.1142/s146902682350013x

nexocode (2021). *Let me read it for you. The Definitive Guide to Natural Language Processing (NLP)*. [online] nexocode. Available at: https://nexocode.com/blog/posts/definitive-guide-to-nlp/

Olamilekan Shobayo, Sidikat Adeyemi-Longe, Popoola, O. and Ogunleye, B. (2024). Innovative Sentiment Analysis and Prediction of Stock Price Using FinBERT, GPT-4 and Logistic Regression: A Data-Driven Approach. *Big Data and Cognitive Computing*, [online] 8(11), pp.143–143. doi: https://doi.org/10.3390/bdcc8110143

Wang, C., Chen, Y., Zhang, S. and Zhang, Q. (2022). Stock market index prediction using deep Transformer model. *Expert Systems with Applications*, 208, p.118128. doi: https://doi.org/10.1016/j.eswa.2022.118128

Zhang, B. (2023). *FinGPT: Enhancing Sentiment-Based Stock Movement Prediction with Dissemination-Aware and Context-Enriched LLMs*. [online] Arxiv.org. Available at: https://arxiv.org/html/2412.10823v1

Ahsan, M.M., Mahmud, M.A.P., Saha, P.K., Gupta, K.D. and Siddique, Z. (2021). Effect of Data Scaling Methods on Machine Learning Algorithms and Model Performance. *Technologies*, [online] 9(3), p.52. doi: https://doi.org/10.3390/technologies9030052

Charles and Smucker, M.D. (2021). Temporal relational ranking for stock prediction. *ACM transactions on office information systems*, 39(3), pp.1–21. doi: https://doi.org/10.1145/3451161

Jing, N. and Wu, Z. (2021). A Hybrid Model Integrating Deep Learning with Investor Sentiment Analysis for Stock Price Prediction. *Expert Systems with Applications*, 178, p.115019. doi: https://doi.org/10.1016/j.eswa.2021.115019

Liu, P. (2023). A Review on Derivative Hedging Using Reinforcement Learning. *The Journal of Financial Data Science*, 5(2), pp.136–145. doi: https://doi.org/10.3905/jfds.2023.1.124