Market Pulse: Stock Trend Prediction Using Multivariate LSTM and Sentiment Analysis

Rishith Gupta(23B1234)
WIDS 4-Week Project

February 12, 2025

Contents

1 Introduction

Stock market prediction has been a long-standing challenge in financial markets, requiring the use of advanced techniques to understand trends and forecast future movements. Traditional statistical methods have given way to machine learning models, which leverage large datasets to extract meaningful patterns. Among these, deep learning models like Long Short-Term Memory (LSTM) networks have shown significant promise due to their ability to capture temporal dependencies in sequential data.

Moreover, financial markets are influenced not only by numerical indicators but also by public sentiment. News articles, social media discussions, and analyst opinions can heavily sway stock prices. Sentiment analysis, a Natural Language Processing (NLP) technique, helps quantify market sentiment and integrate it with stock price predictions. By combining multivariate LSTM models with sentiment analysis, we aim to improve the predictive accuracy of stock trends, providing valuable insights to investors and traders.

This project explores how multivariate LSTM models, enhanced with sentiment analysis, can predict stock market trends more effectively. The following sections will outline the theoretical background, methodology, implementation, and results of this approach.

2 Week 0 : Introduction to time series and Python

The project kicked off with learning Python, the language we would use to implement our final model. Alongside this, an introduction to time series analysis was essential to grasp the fundamentals of stock price prediction.

The assignment for this week focused on identifying trends and patterns in stock price and trading volume. This required exploring the yfinance library, which connects to Yahoo Finance and provides essential stock market data, including opening and closing prices, volume, adjusted closing prices, and daily highs and lows. For my analysis, I chose NASDAQ and examined five years of historical data.

Before diving into analysis, the data needed preprocessing to ensure the correct format, type, and structure. This involved handling missing values, verifying data types, and computing basic statistics. I then created various visualizations, including trading volume and closing price trends over time, and compared them with the 30-day moving average. This exploratory data analysis (EDA) helped uncover key patterns and insights.

Over the past five years, the closing prices have shown a steady upward trend, climbing from 6000 to 16,000. While the overall trajectory is positive, short-lived price spikes often lead to brief corrections before the stock resumes its upward movement. The presence of minor fluctuations, or "noise," can be smoothed out using rolling averages to reveal broader market patterns. Interestingly, no clear seasonal trends were observed in the data.

Trading volume initially remained stable but saw a sharp rise in the last two years, maintaining a higher average level. This surge in activity coincided with a significant price increase, suggesting a potential relationship between trading volume and market movements. Compared to stock prices, volume data appeared noisier, making it more challenging to interpret without applying smoothing techniques.

Figure 1: Closing price vs time

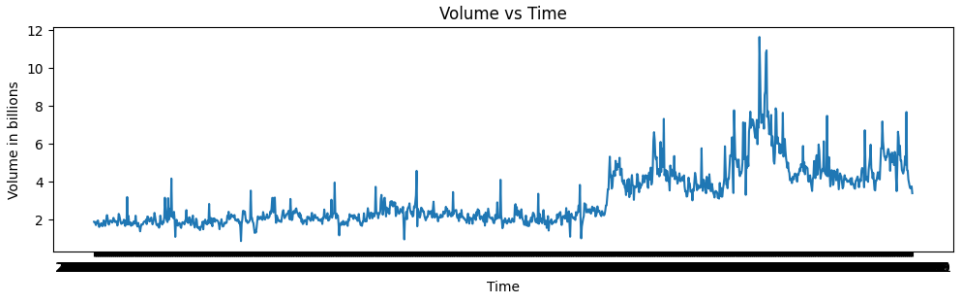

Figure 2: Volume vs time

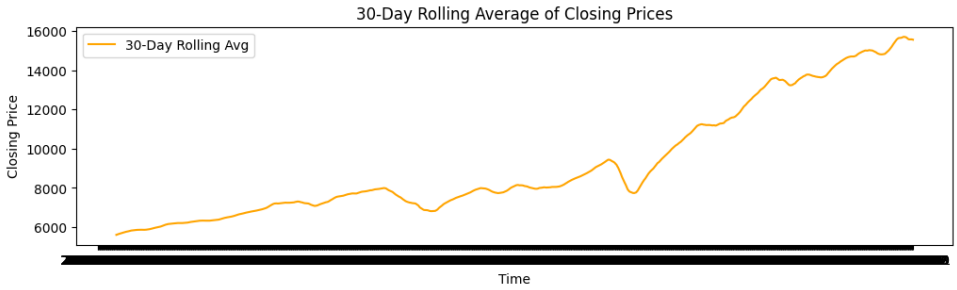

Figure 3: 30 day rolling average of price vs time

3 Week 1 : Sentiment Analysis on text data

This week focused on preparing stock data, extracting features, and understanding the role of sentiment analysis in financial forecasting. This required learning the basics of Natural Language Processing (NLP) and using TextBlob, a popular library for sentiment analysis.

TextBlob is a powerful yet easy-to-use NLP library built on top of NLTK and Pattern. It simplifies various text-processing tasks, making it widely used in sentiment analysis, text classification, and language translation. Key features include tokenization, noun phrase extraction, part-of-speech tagging, word inflection, and lemmatization. These functions help break down text into structured components, making analysis more effective.

One of the most valuable features of TextBlob is sentiment analysis, which quantifies the sentiment expressed in text. It provides two key metrics: polarity, which ranges from -1 (negative) to 1 (positive), and subjectivity, which indicates whether a statement is more factual or opinion-based. This makes it a useful tool for financial analysis, where

market sentiment derived from news and social media can influence stock trends.

Additionally, TextBlob offers built-in spelling correction, making it useful for processing noisy datasets like social media posts. It also supports language detection and translation, allowing seamless text conversion across multiple languages. Another key functionality is text classification using a Naive Bayes classifier, which helps categorize text into predefined labels, such as positive or negative sentiment. The library also supports word frequency analysis and n-gram extraction, making it a versatile tool for NLP-based financial modeling.

This week's assignment involved cleaning the IMDb dataset of top Netflix movies and TV shows in preparation for sentiment analysis. The preprocessing steps included removing rows with missing values and special characters. Since the dataset was already well-structured, minimal additional cleaning was required.

For sentiment analysis, the text data underwent further transformation, including removing stopwords, punctuation, escape sequences, special characters, HTML tags, URLs, and numbers. After preprocessing, lemmatization was applied, followed by sentiment analysis. The output consisted of two key values: polarity, which indicates whether the sentiment is positive or negative, and subjectivity, which measures how opinion-based the text is. These steps laid the foundation for integrating sentiment analysis into stock market forecasting.

	review	sentiment	Label	polarity	subjectivity	Sentiment_feeling	Sentiment_label
35730	great actor an oscar nominee actress stunning ...	positive	1	0.364040	0.564116	POSITIVE	1.0
25028	I m a sucker for mobgangland movie so I rent t...	negative	-1	-0.092857	0.469048	NEUTRAL	NaN
35010	Chris Smiths American Movie be an insightful e...	positive	1	0.237391	0.366594	POSITIVE	1.0
10250	s comedy especially one with John Cusak be awe...	positive	1	0.060606	0.591667	NEUTRAL	1.0
13326	I first see this movie on MSTK and although I ...	negative	-1	0.089773	0.487626	NEUTRAL	NaN
...	...	...	...	...	...	...	...
8378	to some of us director Ernst Lubitsch adore fo...	negative	-1	0.086979	0.412500	NEUTRAL	NaN
15564	its really just terrible Quaid overact more th...	negative	-1	-0.074444	0.507222	NEUTRAL	NaN
38789	the violent and rebel twentyfive year old sail...	positive	1	0.160238	0.583425	POSITIVE	1.0
3480	as one who frequently go to the movie I have t...	positive	1	0.327473	0.522802	POSITIVE	1.0
22804	I just see this movie last night at a midnight...	negative	-1	-0.037665	0.533763	NEUTRAL	NaN

20000 rows × 7 columns

Figure 4: sentiment Analysis results

4 Week 2: Reinforcement Learning

This week provided an introduction to the mathematical foundations of Reinforcement Learning. While this served as a stepping stone, the main focus was on ARIMA (Autoregressive Integrated Moving Average)—a widely used time series forecasting method for predicting future trends.

The first task was to implement a grid search to identify the optimal (p, d, q) parameters for ARIMA by comparing their Akaike Information Criterion (AIC) values. Once the best configuration was found, I proceeded to train the model on Microsoft's stock data, sourced from yfinance. Using a two-year training period, I forecasted the closing prices for the next 30 days.

To evaluate performance, I calculated Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) for the forecasted period. The results were underwhelming, indicating that the model struggled to capture underlying

patterns in the data. However, this experiment provided valuable insights and marked another step forward toward the final project.

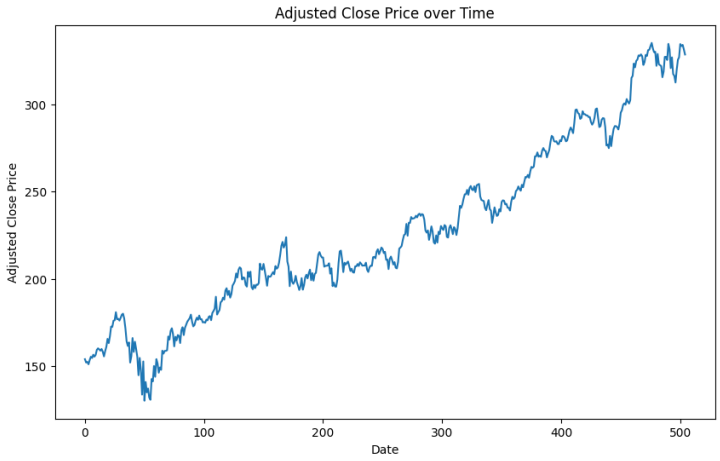

Figure 5: Adjusted Closing price vs time

Figure 6: Differenced closing price vs time for stationality

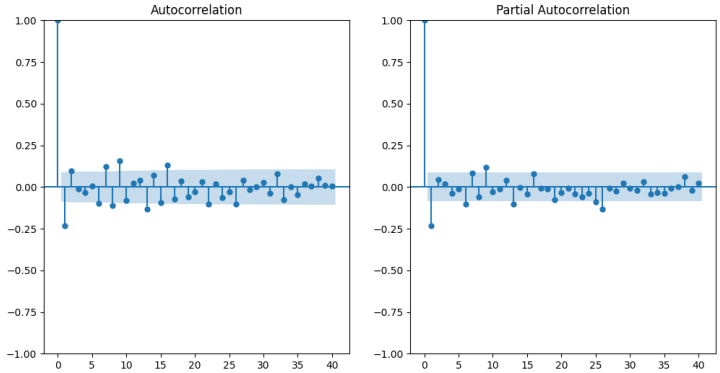

Figure 7: Correlation Graph for ARIMA

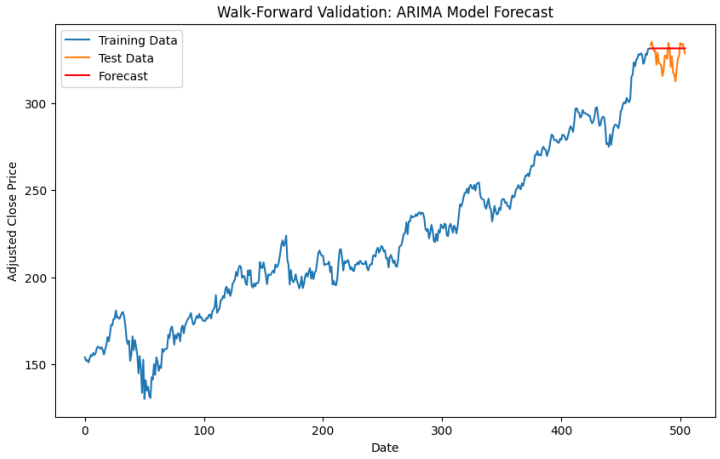

Figure 8: Final Results of ARIMA

5 Week 3: Univariate LSTM model and its comparison with ARIMA

This week focused on exploring LSTM (Long Short-Term Memory) networks and leveraging their power for time series analysis. I began by diving into their architecture, history, and underlying mathematics. After gaining a conceptual understanding, I applied LSTMs to visualize and analyze time series data.

The assignment built upon the previous two weeks by training both an LSTM model and an ARIMA model, comparing their results, and evaluating their strengths and weaknesses. This hands-on approach helped me truly appreciate the capabilities of LSTMs in capturing complex temporal patterns.

While implementing LSTMs using libraries is relatively straightforward, achieving optimal results requires extensive hyperparameter tuning. After fine-tuning, I compared the predictions of ARIMA and LSTM, visualizing the results through various plots. Although the performance wasn't exceptional, this experiment significantly deepened my understanding and prepared me for the final project.

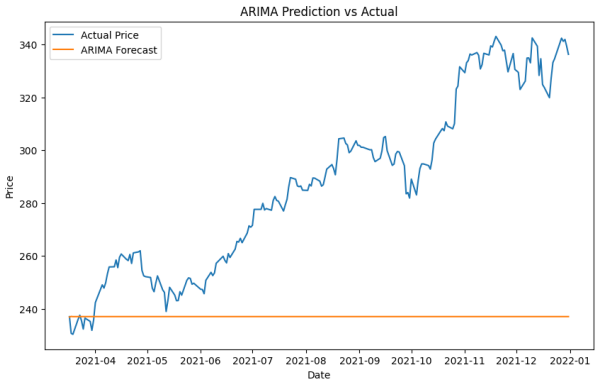

Figure 9: ARIMA prediction output

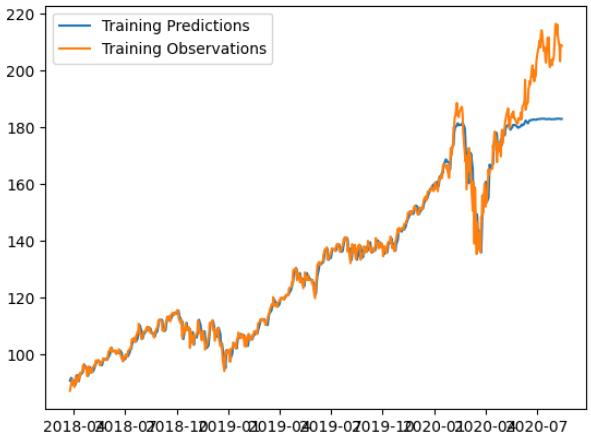

Figure 10: LSTM Training output

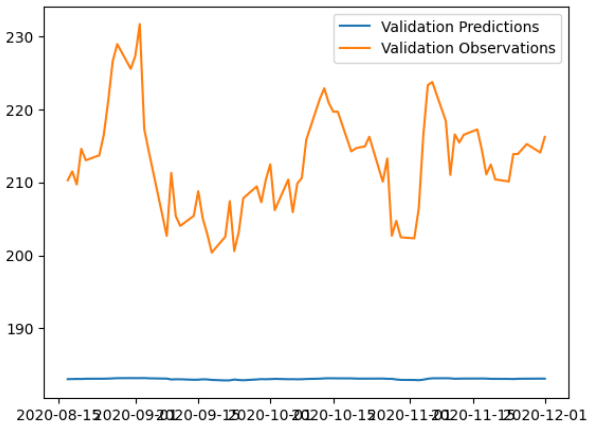

Figure 11: LSTM Validation output

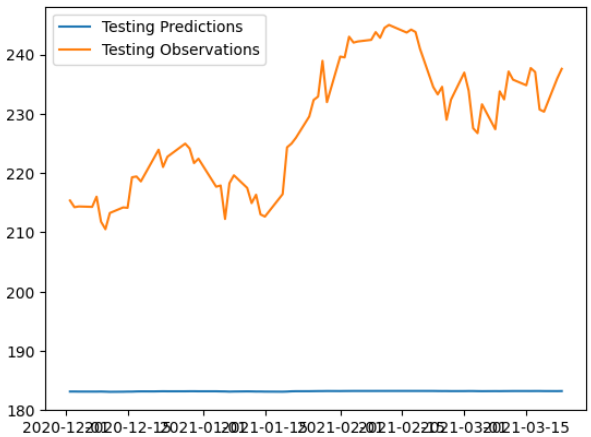

Figure 12: LSTM Testing output

6 Week 4: Final Project

The goal of the final project was to integrate market analysis and sentiment analysis to predict stock trends, reinforcing my ability to combine multiple analytical techniques into a unified model. For this, I chose Tesla stock as my target and sourced news data from

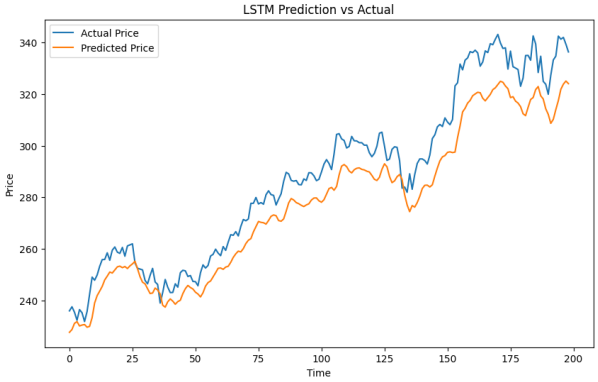

Figure 13: LSTM Overall Prediction output

a Kaggle dataset. After collecting the news, I appended it to the stock price dataset and preprocessed it for sentiment analysis, following the same steps as in Week 2.

For the time series modeling, I used only the polarity score from the sentiment analysis results, excluding subjectivity. The next step was to train an LSTM model using both stock prices and polarity scores as inputs. Given the complexity of the task, training an LSTM wasn't straightforward, but it provided a more powerful approach than ARIMA, making it the preferred choice for this experiment. Once trained, the model was used to generate predictions based on the provided data.

The final results were not highly accurate, but this project was a valuable learning experience. It gave me deeper insights into the challenges of stock market prediction, sentiment analysis, and LSTM networks. More importantly, it pushed me to code independently, without relying on AI assistance, and made me appreciate the difficulty of forecasting market trends.

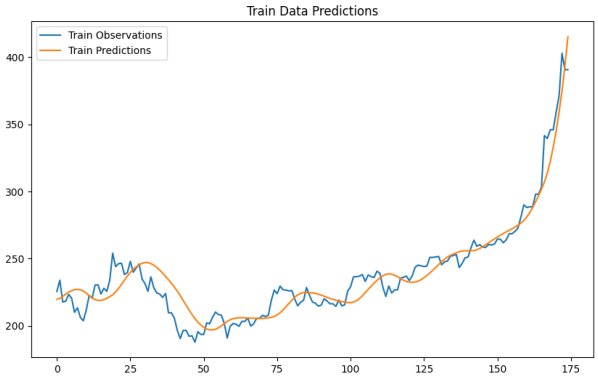

Figure 14: Train Result of Final Project

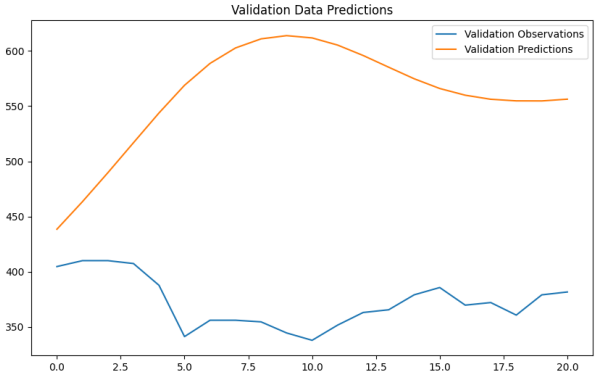

Figure 15: Validation Result of Final Project

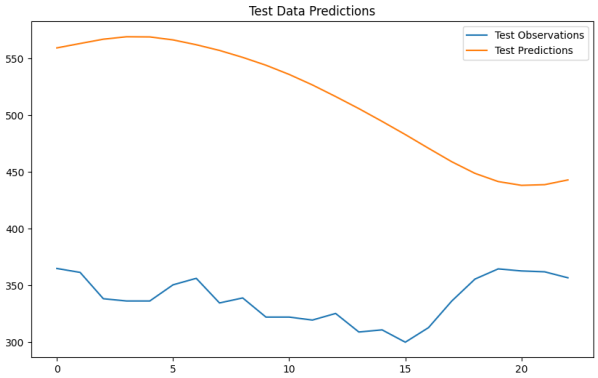

Figure 16: Test Result of Final Project

7 Conclusions

In conclusion, combining multivariate LSTMs with sentiment analysis enhances stock trend prediction by capturing both numerical patterns and market sentiment. LSTMs improve forecasting accuracy by handling sequential dependencies, while sentiment analysis extracts insights from news and social media. Despite challenges such as data noise and market volatility, the model demonstrated promising potential.

One key limitation was the infrequency of news updates, which provided only a few instances for the model to learn how sentiment influences stock prices. Additionally, while the LSTM effectively captured trend patterns, its predictions consistently underestimated actual prices by a certain margin, raising questions about whether the model required more training data or better tuning.

Potential improvements include using transformer-based models like BERT for more accurate sentiment analysis, incorporating real-time news feeds and social media trends for richer market signals, optimizing LSTM architectures with attention mechanisms, and experimenting with ensemble models to enhance predictive performance. These advancements could significantly improve the robustness and reliability of stock trend prediction.