Stock Prediction using Sentiment Analysis and Machine Learning

Rohan Sharma
Industrial and Systems Engineering
University at Buffalo
Buffalo, New York, USA
rs295@buffalo.edu

***Abstract*—This study aims to create a comprehensive model for predicting stock market movements using both quantitative and qualitative data. Traditional quantitative analysis frequently focuses on previous stock prices and financial indicators, which, while useful, only provide a partial view. To overcome this limitation, this research incorporates sentiment analysis obtained from news stories or social media such as tweets, resulting in a multifaceted method to predicting stock market trends. We use statistical machine learning techniques to evaluate historical stock data and build predictive models that take market volatility and trends into account. At the same time, we use natural language processing algorithms to assess the sentiment of news articles, press releases, tweets and financial reports in order to assess public perception and potential market impact.**

I. Introduction

The stock market is a complex web of human behavior, economic data, and global events, all of which are reflected in fluctuating security prices. Understanding and predicting stock market fluctuations has long piqued the interest and confounded experts. Traditional stock market forecasting strategies have primarily relied on quantitative methods, such as statistical and mathematical models that examine historical price data and financial indicators. However, because of the inherent volatility and frequently irrational nature of the markets, which are influenced by psychological and sociological variables, more complex techniques are required.

Recent advancements in machine learning and natural language processing (NLP) have opened up new avenues in the field of financial predictions. Sentiment analysis, in particular, has shown promise in decoding the qualitative aspects of market predictors by analyzing news articles, social media, and financial reports. These qualitative insights, when combined with quantitative data, offer a holistic view of the various factors that drive market behavior.

In this research, we bridge the gap between numerical data and textual information to predict stock market movements. Our dual-pronged approach involves the deployment of statistical machine learning algorithms to sift through and make sense of vast amounts of historical stock data, while simultaneously extracting and quantifying sentiment from textual news sources through web scraping and advanced NLP techniques.

The goal of this study is twofold: first, to build a predictive model using historical stock data that identifies patterns and trends indicative of future market movements; and second, to augment this model with sentiment scores derived from financial news articles to improve its predictive accuracy. We believe that including sentiment analysis will provide a more realistic representation of market attitudes, which are not necessarily rational or based on historical trends.

To put our hypothesis to the test, we extract a large corpus of stock market news stories, tweets about stock etc. from multiple web sources. After that, we use sentiment analysis to assign sentiment scores to the data, which is then combined with historical stock price data. The combined dataset forms the basis of our predictive modeling. We utilize several statistical machine learning techniques, such as regression analysis, time series forecasting, and machine learning classifiers, to create models capable of predicting future stock prices.

The contribution of this paper is significant as it systematically evaluates the impact of sentiment analysis on stock market predictions, providing insights into how qualitative data can be quantified and used to improve the accuracy of stock market forecasts. This research not only enhances our understanding of the stock market dynamics but also offers practical value to investors and policymakers in making informed decisions.

A. Impact

The broader impact of this research stretches across academic, financial, and technological domains, contributing to a more comprehensive understanding of market dynamics and the development of advanced predictive models. Here are some of the key areas where this research could have a significant impact:

Investment Strategies: The findings of this study can be used to develop investment strategies by investors and financial organizations. Investors can better anticipate market moves and manage risks by adding sentiment analysis into their decision-making processes, perhaps leading to higher returns.

Innovations in Financial Technology (FinTech): The methodologies and models developed through this research could drive FinTech innovations, particularly in the areas of algorithmic trading and robo-advisory services. Trading systems that are automated may become more responsive to market emotion, allowing for more agile and educated trading methods.

Understanding how news and public sentiment affect the stock market could help policymakers and regulatory agencies recognize and monitor market sentiment, potentially leading to more timely and effective interventions to prevent market manipulation and maintain market integrity.

Academic Contributions: By proposing a novel technique to market prediction, this study contributes to the academic debate on machine learning applications in finance. It presents a unique dataset for future research as well as a case study of

multidisciplinary machine learning, finance, and computational linguistics applications.

Risk Management: Companies and individual investors can use sentiment analysis to improve risk management by receiving early warning of probable market fluctuations caused by public sentiment. This may result in more proactive tactics for mitigating market downturns.

Researchers and economists might theoretically foresee and study the impact of global events on economic stability by using sentiment analysis globally. This could become a tool for international economic study, leading to a greater understanding of the interconnectivity of global markets.

Transparency and Trust: By making market predictions more data-driven and responsible, this research could contribute to increased financial market transparency, which could promote investor confidence and trust in market mechanisms.

Education and Literacy: The insights from this research could be used to develop educational resources that enhance financial literacy, explaining complex market dynamics in accessible terms and demonstrating the application of machine learning in practical scenarios.

II. Literature Review

We start by reviewing literature that makes predictions on the stock market by conducting sentiment analysis of twitter feed in which the researchers take tweets that contain specific key words. In **[1]**, They have used use 2 models : Boosted Regression Trees and Multilayer Perceptron Neural Network to predict the closing price difference of AAPL and DJIA prices and then compared the 2 models and found that Neural Network model on average performed better.

Sentiment Analysis of Tweets

Their research incorporates sentiment values from tweets related to "Stock Market," "Stocktwits," and "AAPL," as classified by a Support Vector Machine (SVM). These values are averaged daily, creating a dataset where neutral scores (0) are replaced with -1 to distinguish between positive and negative sentiments. This distinction is pivotal, with positive averages suggesting optimistic sentiment and negative averages indicating pessimism.

Stock Index Value Prediction Using Boosted Regression Tree

Adopting the methodology of Chakraborty et al., they applied a Boosted Regression Tree model to predict stock index values. The training dataset spans from January to August 2016, while testing encompasses September to December 2016. We trained our model on daily average sentiment values from tweets about the stock market and "Stocktwits," alongside the Dow Jones Industrial Average (DJIA) closing price differences, applying the same approach to "AAPL" tweets.

The model predicts the next day's stock price movement by using the present day's sentiment averages, enabling us to estimate the subsequent day's market direction. Testing involved processing tweets through SVM to obtain sentiment averages, which were then used by the Boosted Regression Tree to forecast the next day's price differences.

Stock Index Value Prediction Using Multilayer Perceptron Neural Network

They introduced a Multilayer Perceptron Neural Network (MLPNN) to evaluate its predictive capabilities against traditional models. The MLPNN was trained on the same January-August 2016 dataset, with testing performed on the September-December 2016 data. It utilized daily average sentiment values from tweets related to the stock market and "Stocktwits," trained against DJIA closing price differences. Our MLPNN was developed to anticipate the next day's stock price changes, informed by the sentiment values from the current day's tweets. The system was calibrated using the previous day's tweet sentiment averages to predict market movement accurately. During testing, SVM-classified tweet sentiment averages were leveraged by the MLP Neural Network to predict stock differences for the forthcoming day.

The next research, **[2]** explores the efficacy of Support Vector Machine (SVM) with Radial Basis Function (RBF) in predicting stock market trends. Utilizing both SVM's capacity for pattern recognition and RBF's local response characteristics, we've crafted a robust model for financial forecasting.

Support Vector Machine (SVM)

SVM operates as a classifier, establishing a hyperplane in an n-dimensional space that distinctly categorizes new examples. Its decision-making capability is based on a boundary maximized on both sides of the classified data, governed by a set of vector and optimization equations, ensuring the hyperplane's optimal position.

Radial Basis Function (RBF)

The RBF kernel, integral to SVM classification, relies on distance metrics that emphasize locality in data, resembling Gaussian functions in nature. It filters inputs to produce smooth, generalizable prediction surfaces, ideal for handling complex, non-linear patterns often encountered in financial datasets.

The Learning Environment

The Weka and YALE Data Mining Environments were employed to facilitate the experiments, ensuring a controlled and consistent learning environment for the SVM and RBF applications.

Model Creation and Evaluation Methods

They deployed SVM with RBF to predict stock market performance, prioritizing features such as stock and sector volatility, as well as momentum indicators. The study's methodology includes retrieving financial data, converting it to a machine-readable format (CSV), and processing it through the SVM for predictive analysis.

The research outputs included visual representations of stock datasets and the predictive performance of the SVM model, with the latter demonstrating the algorithm's potential in accurately forecasting stock prices.

Conclusion

The project confirms SVM's utility in analyzing extensive financial data, effectively avoiding the pitfall of overfitting. The proposed model not only demonstrates high predictive efficiency but also shows promise in developing practical trading strategies that outperform traditional market benchmarks.

In the next research that we reviewed, the interplay between market indicators and public sentiment is crucial in stock market dynamics. This paper, **[3]** presents a novel approach that integrates sentiment analysis and moving averages to forecast stock market trends. The sentiment analysis utilizes natural language processing (NLP) to gauge public mood from news feeds, while moving averages offer an objective measure of market performance. The combined model outperforms traditional data mining methods, providing actionable insights for market participants.

Sentiment analysis or opinion mining is a potent facet of NLP that discerns emotional undercurrents in textual content. By analyzing language nuances and context, this technology extracts attitudes and opinions regarding various topics. In the business realm, sentiment analysis processes vast arrays of online reviews, ratings, and comments, unveiling customer inclinations and market trends.

Sentiment Analysis in NLP

Using a dictionary-based approach, sentiment analysis leverages POS tagging and the Sentence Sentiment Score (SSS) algorithm to quantify sentence polarity—positive, negative, or neutral. The POS tagger delineates grammatical categories, aiding in the contextual understanding of words. The SSS algorithm then assigns numerical sentiment scores to sentences, aggregating them to represent the overall sentiment of the document.

Moving Average as a Stock Indicator

The moving average is a widely used stock indicator, smoothing out price data by creating a constantly updated average price. This is particularly useful for identifying trends, with the common practice of using 5-day, 10-day, and 15-day moving averages.

Methodology

Their method involves collecting historical stock prices and corresponding news sentiment data, then calculating both the moving average and sentiment polarity. The news sentiment for a particular company is derived from RSS feeds, while moving averages are computed using historical stock prices.

Results

In the experiment, they forecasted the stock market trends for Arab Bank (ARBK) from the Amman Stock Exchange (ASE). They compared the proposed sentiment and moving average model with traditional methods, such as ID3 and C4.5 decision tree algorithms.

The sentiment analysis was applied to the news feeds, and the Sensex points were gathered to calculate the moving averages. The combined sentiment and moving average results yielded a prediction accuracy of 78.75%, surpassing the 64.32% accuracy achieved by the moving average model alone and the even lower accuracies of the ID3 and C4.5 methods.

Discussion

The enhanced precision and recall of our approach suggest that sentiment analysis contributes significant value to stock market predictions. The sentiment data, when analyzed with the SSS algorithm and combined with the objectivity of moving averages, provides a more accurate reflection of market direction.

Conclusion

This study confirms the utility of integrating sentiment analysis with traditional market indicators for stock market prediction. The methodological synergy offers a comprehensive perspective, factoring in both numerical market data and the qualitative nuances of public sentiment.

The next research paper that we review takes a look at the various machine learning algorithms available to us for stock market prediction. In **[4]**, the researchers tell us that Stock prediction remains a critical and challenging task due to the dynamic and fluctuating nature of the market. The primary goal of our research is to facilitate stockbrokers and investors in making informed decisions to invest in the stock market. The complexity of the stock market is influenced by numerous factors, including political climate, economic crises, and other market-affecting elements. This paper has provided a comprehensive survey of various machine learning approaches that are instrumental in stock prediction, such as Natural Language Processing (NLP), Linear Regression, k-Nearest Neighbors (KNN), Support Vector Machine (SVM), Long Short-Term Memory networks (LSTM), and Artificial Neural Networks (ANN). The utility of predictive models lies in their ability to guide investors, novices, and stakeholders, providing them with insights on where to invest or hold stocks to maximize profits while minimizing risks. However, the integrity of the dataset is crucial for accurate predictions. If the dataset includes misinformation or irrelevant data, the predictions derived could lead to erroneous investment decisions.

In the domain of stock prediction, machine learning models undergo several processes, beginning with pre-processing to clean the data from redundancy, obsolescence, or formatting errors that could introduce uncertainty. This phase involves data cleaning, transformation, and reduction. Feature selection then follows, aiming to reduce the dimensionality of the dataset and select pertinent features through supervised or unsupervised methods. This not only reduces computational costs but may also enhance the model's precision.

Different machine learning techniques have been applied to stock prediction with varying degrees of success. The ANN has shown promise in improving prediction accuracy, albeit with the complexity of modeling issues accurately for network transmission. The use of sentiment dictionaries and NLP can lead to significant outcomes, yet they may suffer from limitations like the neglect of word context and sentiment recognition. Methods such as SVM have been recognized for their reliability in prediction but are computationally intensive with large datasets.

The challenge of selecting an optimal machine learning model is evident, as the choice depends on the nature of the data. For instance, KNN might be more efficient than SVM when the training data volume exceeds the number of features. Conversely, SVM tends to outperform KNN when the feature set is larger than the training data. Additionally, the process of sorting, pre-processing, and extracting features from a substantial real-time dataset poses a considerable challenge.

The necessity of frequently re-training models to reflect the latest market changes adds another layer of complexity. Determining the optimal re-training interval remains ambiguous and requires further research to ensure the best predictive performance.

In conclusion, despite the promising results from various machine learning models in stock prediction, the process is hampered by the market's volatility and the quality of input data. Future work aims to enhance the accuracy of stock predictions by developing new methods and models that surpass existing ones in performance and can circumvent current limitations. The researchers intend to design a model that not only offers greater precision but is also robust against the inclusion of inaccurate data. This endeavor will contribute significantly to the field of stock market prediction, providing a reliable tool for investors and market analysts.

The next research **[5],** scrutinizes past market data, primarily historical prices and volumes, to forecast potential price movements. It operates under the premise that stock prices follow identifiable trends over time and that market-affecting information disseminates gradually rather than instantaneously. This stance challenges the Efficient Market Hypothesis (EMH), which posits that stock prices evolve in a random walk, rendering prediction futile since new information is instantaneously reflected in prices. Despite EMH's longstanding prominence, its refutation has gained traction due to the empirical successes of technical analysis and Artificial Intelligence (AI) in market prediction, prompting a growing number of academics to treat EMH as a null hypothesis.

Technical analysts rely on a variety of indicators, which are essentially mathematical computations that provide insights into market trends. One such indicator is the stochastic oscillator %K, which is calculated using the closing, highest, and lowest prices of a security within a specified timeframe. While interpretations of these indicators can vary among analysts, their integration into AI techniques has demonstrated notable success.

Choudhry and Garg's work underscores the efficacy of Support Vector Machines (SVMs), originally proposed by Vapnik, in market forecasting. SVMs, as maximum margin classifiers, strive to find an optimal separating hyperplane (OSH) by maximizing the margin between the hyperplane and the nearest data points, termed support vectors. For linearly separable data, the separation is straightforward. However, for non-linear data, SVMs employ kernel functions to map inputs into a higher-dimensional feature space, allowing for more complex decision boundaries.

In the context of the stock market, the prediction task is framed as a binary classification problem, with two possible outcomes: a rise or fall in stock prices compared to the previous day. The Indian stock market, particularly the stocks of Tata Consultancy Services (TCS), Infosys, and Reliance Industries Limited (RIL), was chosen for this study due to its unique characteristics distinct from more commonly studied markets like the United States or South Korea.

To improve prediction accuracy, Choudhry and Garg consider the correlation between stocks, acknowledging that price movements are not isolated events. The correlation between two stocks, S and T, is quantified over a period using their closing prices, means, standard deviations, and the number of days. For instance, companies closely correlated with TCS are often in the same industry or corporate group, as seen in a correlation example with other firms.

The proposed system utilizes 35 technical indicators as candidate input features, applied across the most correlated companies with the target stock. This yields a large set of potential features (35*m), from which a Genetic Algorithm (GA) selects the most relevant subset to feed into an SVM. The process includes chromosome representation, fitness evaluation based on classification accuracy, Roulette Wheel selection, and typical genetic operations like crossover and mutation, concluding once an optimal solution persists over a set number of generations.

An optimal subset of features determined by GA is used to train the SVM, which is adjusted using Gaussian radial basis function kernel due to its superior performance in experiments. The study's dataset comprises 1386 trading days' data from the Sensex index, split into training, validation, and testing sets.

The model's efficacy is measured by the hit ratio, the percentage of correct predictions for stock price direction. The GA-SVM hybrid outperformed a standalone SVM, with significant improvements in hit ratios for all three analyzed stocks. For example, the GA-SVM yielded a hit ratio of 61.7328% for TCS compared to 58.0903% for the standalone SVM.

Conclusively, the hybrid GA-SVM system, leveraging technical indicators and stock correlations, exhibits a marked enhancement in stock price direction prediction. While integrating political and economic factors could further improve outcomes, the study already signifies a substantial advancement in the application of AI for stock market forecasting, inviting additional exploration into the integration of market-specific knowledge and broader input variables.

The study at hand **[6]**, directs attention to the analysis of subjectivity within discussions of societal matters, positing that the subjective nature of a document is largely influenced by its constituent sentences. This study looks at how people express personal views in writing about social issues. It says that whether a piece of writing feels personal or not mostly depends on the sentences used. The researchers created a way to spot and sort out personal opinions in sentences, especially looking at the important role of action words (verbs) when people talk about social issues. Additionally, the research calls for an exploration into the strengths and weaknesses of sentences that are objective in nature.

III. Methodology

I took data of different stocks which included their opening price on that day, high, low, closing price, volume traded, adjusted closing price and the date these were recorded.

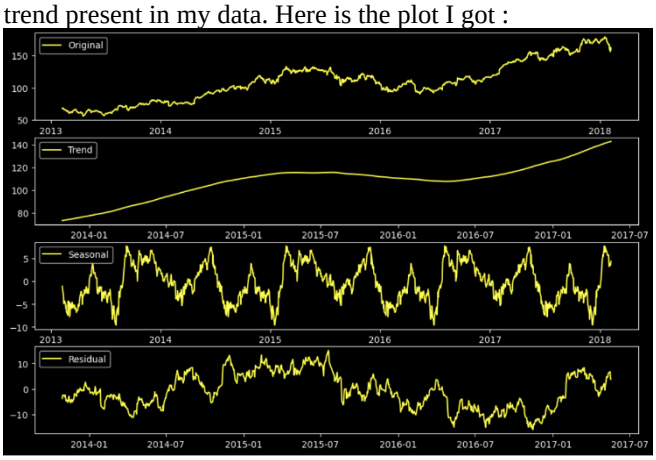
I chose Apple stock to make predictions upon and the first model I tried was an ARIMA model which is typically used for time series data.

For the ARIMA model I used only one predictor – ‘close’, which was the closing price of the stock as this is a common method for predicting stock prices using ARIMA. ARIMA stands for Auto Regressive Integrated Moving Averages where we are regressing the target variable on itself.

We began by some basic data preprocessing such as extracting the relevant columns and converting the date column to datetime type and setting it as the index.
Initially, I implemented the Dickey – Fuller test on the data which tells us whether the data is stationary or not . If the data is stationary, we can use simple ARIMA model, but if it is non-stationary we have to make some changes to the data to make it stationary and then apply ARIMA. I found from the test that our data was highly non-stationary.
Now, to deal with this we have 2 ways :

- We can either make some changes to the data so that it becomes stationary and then apply ARIMA model. Or
- We can use the Auto-Arima function from pmdarima package which tells us the best type of ARIMA model we can use for our model based on changing the parameters of the ARIMA model.

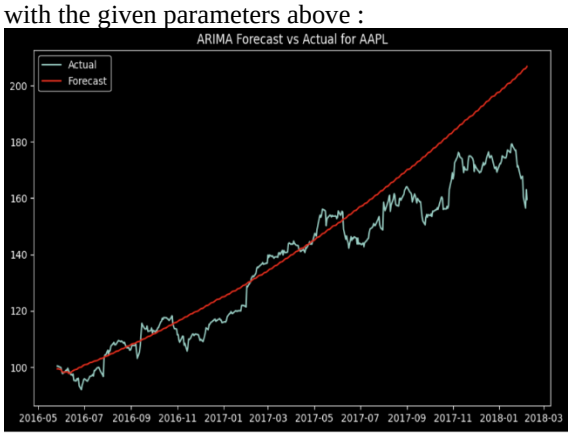
Before that, I extracted and plotted trend, seasonality and residuals in the data to get an idea of the seasonality and trend present in my data. Here is the plot I got :



We can see from here that our data is highly seasonal and therefore, I suspected that a SARIMAX model would work very well with our data as it can deal with seasonal data efficiently. Therefore, I first used the auto arima function to find out the best model that would be applicable to our data. Here, I encountered the problem of computational intensity as the autoarima function is computationally expensive to run and I was not able to run it on my device as it used up my entire RAM before fininshing execution. I tried to run it on a hosted runtime on Google Colab but it was taking too long to execute.
Thus, I switched to the first option where I manually experimented with different parameters of the ARIMA model and tried to find the one which was the best fit for my data. Here is a brief explanation of the parameters of the ARIMA model and what they mean :

1. ***p* - Autoregressive (AR) Order:**
   - The parameter *p* refers to the number of lagged observations included in the model, also known as the lag order.
   - These are essentially the previous data points in a time series that are used to predict the future value.
   - A *p* value of 0 means that no previous values are used for prediction, and the higher the value of *p*, the more past values are included.
2. ***d* - Differencing Order:**
   - The *d* parameter represents the number of times the data needs to be differenced to make it stationary.
   - Stationarity is a critical concept in time series analysis, indicating that the statistical properties of a series (like mean and variance) do not change over time.
   - Differencing is the process of subtracting the current value from the previous value. If the time series is already stationary, *d* will be 0.
3. ***q* - Moving Average (MA) Order:**
   - The parameter *q* refers to the size of the moving average window, which is the number of lagged forecast errors in the prediction equation.
   - These are errors made in previous predictions, with *q* indicating how many of these past errors are used to predict the future value.
   - A *q* value of 0 means that no moving average component is used in the model.

After a lot of trail and error this is the combination of parameters I was able to find that gave me a reasonable fit for my data :
p = 30
d = 3
q = 1
And here is the plot of actual vs predicted values after running the ARIMA model with the given parameters above :



As we can see, its not making very good predictions of the stock prices and does not predict the seasonality but it still manages to somewhat capture the trend of the data.
I plan on using this as a base model to compare the rest of my models with.

The next model I chose to predict the stock prices was the LSTM Neural Network model which is also a model that is used for stock price forecasting often.
Here, I have first used the model on just quantitative data of stock price and its indicators and evaluated the model

performance on that and then used the same dataset but with sentiment scores from tweets incorporated into the dataset and then checked the model performance on that dataset to see if Sentiment Analysis offered us any improvement over our base LSTM model. For the Sentiment Analysis, I have not conducted the sentiment analysis by myself but have taken a dataset where the author has conducted sentiment analysis of tweets about Apple company and converted them to sentiment scores using NLP techniques and created a dataset of Apple stock from 2016 to 2020 with the columns as opening prices, closing, adjusted closing prices, volume traded, sentiment score and the date all of these were recorded.

In our first model without Sentiment Analysis, we took our Apple stocks dataset from 2016 to 2020 and dropped the ts_polarity column (the column which had the twitter sentiment polarity scores). We applied some basic data preprocessing steps such as removing the 'na' values and setting the date column as the index. Initially, I added technical indicators to our dataset from the 'ta' package which calculates a lot of technical indicators such as simple moving average, exponential moving averages etc. and adds them as columns to our dataset. But I found that my LSTM model performed better with just the basic indicators such as open, high, low and volume as inputs as compared to including other technical indicators in the model which is why in our final models, I have not included the technical indicators calculated from the 'ta' package and just used the base data for the modeling and prediction.

Before starting the modeling phase, I scaled the entire data using StandardScaler() to make the machine learning process more efficient.

After that, I split the data into a **80-20** split for training and testing and reshaped it to be fed into the LSTM model.

Model Architecture

1. Sequential Model:
    - The Sequential model is a linear stack of layers in Keras. It allows you to create models layer-by-layer in a step-by-step fashion.
2. First LSTM Layer:
    - LSTM(60, activation='relu', input_shape=(1, X_train_reshaped.shape[2]), return_sequences=True)
    - This is the first LSTM layer with 60 units (neurons).
    - It uses the ReLU (Rectified Linear Unit) activation function. ReLU is commonly used in neural networks for its efficiency and simplicity. It basically outputs the input directly if it is positive, otherwise, it will output zero.
    - input_shape specifies the shape of the input data. It's important for the first layer in a Sequential model to know the shape of the input data.
    - return_sequences=True is set because we have another LSTM layer following this one. It ensures that the output is a sequence, which can be processed by the next LSTM layer.
3. Dropout Layer:
    - Dropout(0.2) is a technique used to prevent overfitting.
    - It randomly sets a fraction (20% in this case) of input units to 0 at each update during training, which helps prevent overfitting by making the model less sensitive to the specific weights of neurons.
4. Second LSTM Layer:
    - LSTM(50, activation='relu', return_sequences=False)
    - This second LSTM layer has 50 units.
    - It also uses the ReLU activation function.
    - return_sequences=False is set because this is the last LSTM layer, and we only need the final output to pass to the dense layer.
5. Second Dropout Layer:
    - Another Dropout(0.2) layer is added for the same reason – to reduce overfitting.
6. Dense Layer:
    - Dense(1, activation='linear')
    - This is the output layer of the network, consisting of a fully connected (Dense) layer with a single neuron.
    - It uses a linear activation function. In a regression setting like time series forecasting, a linear activation function can be used to predict continuous values.
7. Model Compilation:
    - model.compile(optimizer='rmsprop', loss='mean_squared_error')
    - This step is about compiling the model with an optimizer and a loss function.
    - The RMSprop optimizer is an adaptive learning rate method, which makes it a good choice for recurrent neural networks.
    - The loss function used is the mean squared error, which is common for regression problems.

Model Training:

model.fit(X_train_reshaped, y_train, epochs=100, batch_size=28, validation_split=0.1)

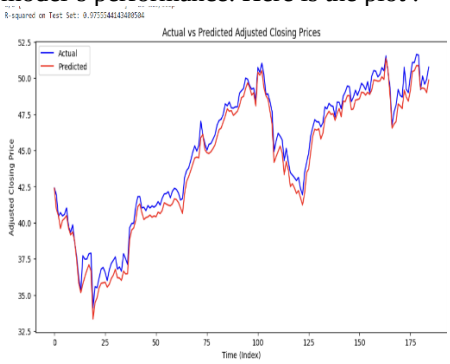
This line trains the model on the provided training data (X_train_reshaped and y_train) for 100 epochs, meaning the model will work through the entire dataset 100 times.

batch_size=28 means that the model will update weights after processing 28 samples.

validation_split=0.1 means that **10%** of the training data is set aside to validate the model, i.e., to test the model on unseen data.

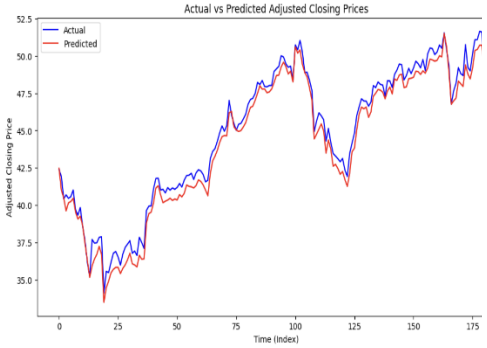
IV. RESULTS

After creating the model and fitting it on the training data, we use it to make predictions on the test data.
I have calculated the R-squared value on the test set and plotted the actual vs predicted values plot of the **Adjusted Closing** price in order to evaluate the model's performance. Here is the plot :



The R-squared value I got was approx. **0.9755**
Which means that about **97.55%** of the variance in the test data was explaind by our model. As can be seen we achieved very good results from our LSTM model in predicting the Adjusted Closing price of our Apple stock data which is a testament to the predictive power of RNN models and their ability to handle complex datasets.

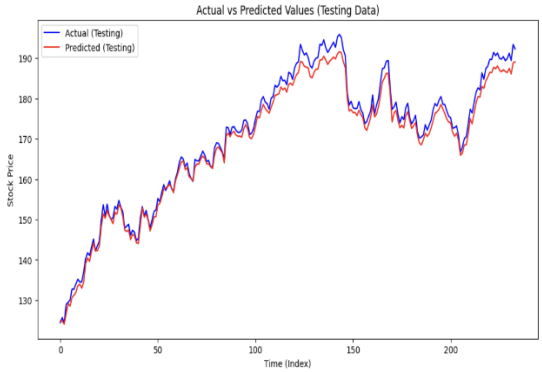
Next, I used the same model on the same dataset but this time included the **twitter sentiment polarity scores** as input to the model to see how much variation in results we get by incorporating Sentiment Analysis in our modeling. After following the same steps as above and making the predictions on the test set, here are the results we got :
R-squared on Test Set:
0.977870981664357



As we can see, there was a very small improvement in our results after sentiment analysis as **0.23%** more variance was explained by our model that used sentiment scores as inputs.

V. CONCLUSION

Thus, we can see from here that there was not a significant improvement in our results after we incorporated sentiment analysis. There could be many reasons for this :
Firstly, its possible that our LSTM model performed so well on the quantitative data alone that there was not a lot of scope for significant improvement.
Secondly, the source of the sentiment data could also be questioned. We could say that twitter data may not be the most reliable source for predicting stock market trends.
In order to confirm whether the model performed as well on different samples of stock data, I extracted more data of Apple stock i.e. from **2021-2022 as training data and Jan 2023 to current date** as testing data but without sentiment analysis. After using the same model on this data, here are the results :
R-squared testing : 0.985661626919553



As we can see, the model again performs very well on this dataset. This leads me to believe that the model is performing so well on just the quantitative stock data of Apple that there is not much scope for sentiment analysis here.

Thus, **future research work** is required where we can take data of different stocks and try different sources for our sentiment data such as relevant financial news channels, articles etc. to see how Sentiment Analysis impacts our Stock price prediction.

	R^2 for LSTM without sentiment analysis	R^2 for LSTM with sentiment analysis
Apple data(2016-2020)	**0.9755**	**0.9778**
Apple data(2021-present)	**0.9856**	-

References

[1] Kolasani, S.V. and Assaf, R. (2020) Predicting Stock Movement Using Sentiment Analysis of Twitter Feed with Neural Networks. *Journal of Data Analysis and Information Processing*, **8**, 309-319.

[2] Reddy, V. Kranthi Sai. "Stock market prediction using machine learning." *International Research Journal of Engineering and Technology (IRJET)* 5.10 (2018): 1033-1035.

[3] Bharathi.Sv, Shri & Geetha, Angelina. (2017). Sentiment Analysis for Effective Stock Market Prediction. International Journal of Intelligent Engineering and Systems. 10. 146-154. 10.22266/ijies2017.0630.16.

[4] Patel, Ramkrishna & Choudhary, Vikas & Saxena, Deepika & Singh, Ashutosh. (2021). REVIEW OF STOCK PREDICTION USING MACHINE LEARNING TECHNIQUES. 10.1109/ICOEI51242.2021.9453099.

[5] Choudhry, Rohit, and Kumkum Garg. "A hybrid machine learning system for stock market forecasting." *International Journal of Computer and Information Engineering* 2.3 (2008): 689-692.

[6] *Mostafa Karamibekr and Ali A. Ghorbani. 2013. Sentence Subjectivity Analysis in Social Domains. In Proceedings of the 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT) - Volume 01 (WI-IAT '13). IEEE Computer Society, USA, 268–275. https://doi.org/10.1109/WI-IAT.2013.39*