# STOCK PRICE PREDICTION

AI&ML COURSE PROJECT REPORT –MA60274

**Under the supervision of**
**prof.Buddhananda Banerjee**
**& prof.Bodhayan Roy**
**Department of Mathematics**

**INDIAN INSTITUTE OF TECHONOLOGY, KHARAGPUR**
**West Bengal, India- 721302**
**April 4th,2025**

Group Members:
**KRISHITHA DARA - 21MA25019**
**MAYANK KUMAR - 24MA60R10**
**PRADEEP AGRAHARI - 24MA60R05**
**ANUJ SINGH - 24MA60R18**
**SHUBHJAY KUMAR- 24MA60R29**
**AMAR KUMAR PANDEY - 24MA60R30**

# Contents

# 1. ABSTRACT

The valuation of a company's stock, reflecting its financial health and market position, is crucial for investors and stakeholders m. It is challenging for customers or stockholding companies to predict the future value of a single stock due to the volatility of stock prices. This uncertainty often leads to substantial financial losses for investors. To overcome this issue, we designed a project that focuses on developing a stock price prediction web application (for now, it’s just a flask app accessible using systems’ IP address) utilizing two prominent machine learning algorithms: Linear Regression and Long ShortTerm Memory (LSTM) networks. Using the historical stock data, we aim to harness the strengths of both models to enhance prediction accuracy. The main objective is to evaluate and compare the effectiveness of Linear Regression's statistical approach with the advanced capabilities of LSTM, which is designed to capture long-term dependencies in time-series data. This research seeks to establish a robust framework for stock price forecasting that can be beneficial for investors seeking informed decisions in a volatile market environment.

**Keywords:** Machine Learning, Linear Regression, LSTM, Python, Yahoo Finance.

## 2. INTRODUCTION

The stock market is a place, where investors buy and sell shares of publicly traded companies. It serves as a platform for companies to raise capital and for investors to acquire ownership stakes. Various factors, such as market mood, company performance, geopolitical events, and economic data, affect stock prices. Accurately predicting stock prices is essential for investors and financial analysts, as it can significantly impact investment decisions and financial returns. Traditional statistical methods frequently fall short due to the inherent volatility and complexity of stock price fluctuations, creating a need for advanced techniques such as machine learning.

Linear Regression and Long Short-Term Memory (LSTM) networks are supervised machine learning that are commonly used in predictive analysis. The continuous values of mathematical variables are largely consistent with the linear regression model, which produces linear correlations between independent and dependent variables. Based on the instructions supplied in the training data, the algorithm produces the predictions.

LSTM networks bring a different strength to the table. As a special type of recurrent neural network (RNN), they are designed to capture long-term dependencies in sequential data, which is crucial when we look at time-series forecasting. This means LSTMs can remember information over longer periods, allowing them to recognize complex patterns in stock price movements that might be overlooked by simpler models.

The Linear Regression gives us a clear and interpretable approach to understanding data relationships, LSTM dives deeper into capturing the nuances of historical trends. Together, these methods form a solid framework for predicting stock prices, providing investors with valuable insights to make more informed decisions.

# 3. MACHINE LEARNING MODELS

## 3.1. LINEAR REGRESSION :

Linear regression is a type of supervised machine learning algorithm that computes the linear relationship between the dependent variable and one or more independent features by fitting a linear equation to observed data. When there is only one independent feature, it is known as Simple Linear Regression, and when there are more than one feature, it is known as Multiple Linear Regression. It is a type of machine learning where the algorithm learns from labelled data. Labelled data means the dataset whose respective target value is already known. It determines the line of least resistance or the best fit line, this line can be utilised to create stock predictions.

- The equation for simple linear regression is :

  $$Y = \beta_0 + \beta_1 X + \varepsilon$$

  Where :

  Y is the dependent variable

  X is the independent variable

  $\beta_0$ is the intercept

  $\beta_1$ is the slope

  ε is the error term (captures the unexplained variation in Y)

### 3.1.1. Multiple Linear Regression :

- The equation for multiple linear regression is :

  $$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n + \varepsilon$$

  Where :

  Y is the dependent variable

  $X_0$,$X_1$, $X_2$,... ,$X_n$ are the independent variables

  $\beta_0$ is the intercept

  $\beta_0$,$\beta_1$, $\beta_2$, ... , $\beta_n$ are the slopes(here we call them as Regression-coefficients)

  ε is the error term

### 3.1.2. How Linear Regression works :

Our primary objective while using linear regression is to locate the best-fit line, which implies that the error between the predicted and actual values should be kept to a minimum. There will be the least error in the best-fit line.

The best Fit Line equation provides a straight line that represents the relationship between the dependent and independent variables. The slope of the line indicates how much the dependent variable changes for a unit change in the independent variable(s).

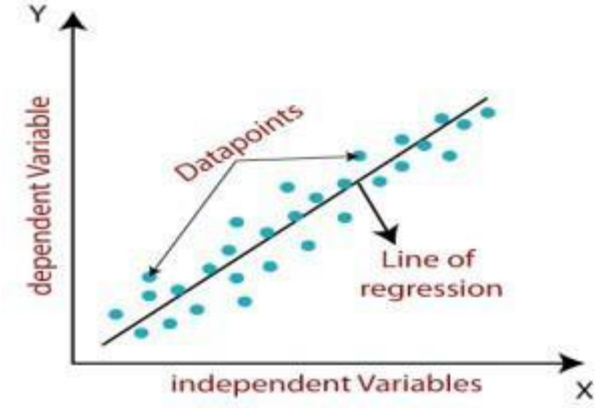


**Fig. 1**. Representation of best fit line of regression

To achieve the best-fit regression line, the model aims to predict the target value $\hat{Y}$ such that the error difference between the predicted value $\hat{Y}$ and the true value Y is minimum. So, it is very important to update the $\beta_0$ and $\beta_i$ values(i=1,2,…n), to reach the best value that minimizes the error between the predicted y value ($\hat{Y}$) and the true y value (Y).

## 3.2. LSTM – a special type of RNN :

### 3.2.1. What is RNN :

A Recurrent Neural Network (RNN) is a class of neural networks specifically designed to process sequential data. Unlike traditional feedforward neural networks, RNNs have connections that form directed cycles, allowing information to persist across steps. This makes RNNs powerful for tasks involving sequences, such as time series prediction, speech recognition, and natural language processing.

A traditional RNN has a single hidden state that is passed through time, which can make it difficult for the network to learn long-term dependencies. LSTMs model address this problem by introducing a memory cell, which is a container that can hold information for an extended period.

### 3.2.2. Long Short-Term Memory (LSTM) Networks :

Long Short-Term Memory (LSTM) is an improved version of RNN designed to capture both short- and long-term dependencies in sequential data. It was introduced to address the vanishing and exploding gradient problems that standard RNNs face.

### LSTM architecture :

The LSTM architectures involves the memory cell which is controlled by three gates: the input gate, the forget gate, and the output gate. These gates decide what information to add to, remove from, and output from the memory cell.

- Input gate : Input gate consists of the input.
- Cell State : Runs through the entire network and has the ability to add or remove information with the help of gates.
- Forget gate layer: Decides the fraction of the information to be allowed.
- Output gate : It consists of the output generated by the LSTM.
- Sigmoid layer generates numbers between zero and one, describing how much of each component should be let through.
- Tanh layer generates a new vector, which will be added to the state.

This allows LSTM networks to selectively retain or discard information as it flows through the network, which allows them to learn long-term dependencies.

The LSTM maintains a hidden state, which acts as the short-term memory of the network. The hidden state is updated based on the input, the previous hidden state, and the memory cell’s current state.

### 3.2.3. How LSTM works :

LSTM architecture has a chain structure that contains four neural networks and different memory blocks called cells.

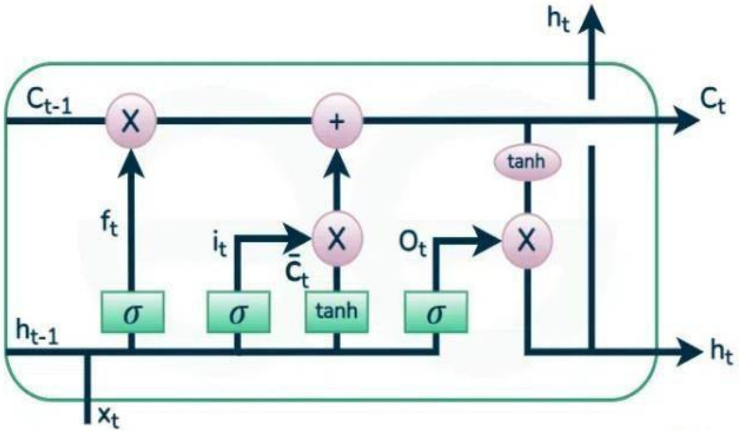


**Fig. 2**. Representation of LSTM Cell

Information is retained by the cells and the memory manipulations are done by the gates**.** The LSTM cell includes three gates to regulate the flow of information :

1. **Forget Gate** ($\boldsymbol{f_t}$): It decides how much of the previous cell state should be "forgotten" or retained. This is a sigmoid layer that outputs a value between 0 and 1 (0 means "completely forget" and 1 means "completely retain").

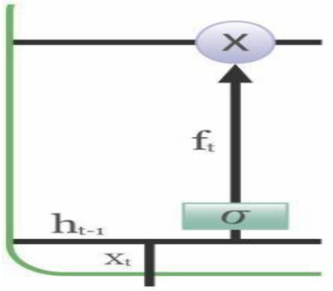


**Fig. 3**. Representation of Forget gate

$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f)$

Where :

$f_t$ : Forget gate output (a value between 0 and 1 indicating how much information to forget). σ : Sigmoid activation function. $W_f$ : Weight matrix for the forget gate. $h_{t-1}$ : Previous hidden state (from the previous time step).

$x_t$ : Input at the current time step.

$b_f$ : Bias for the forget gate.

2. **Input Gate ($\boldsymbol{i_t}$):** It controls how much new information from the current input should be written to the cell state. It has a sigmoid layer and a candidate memory update component.

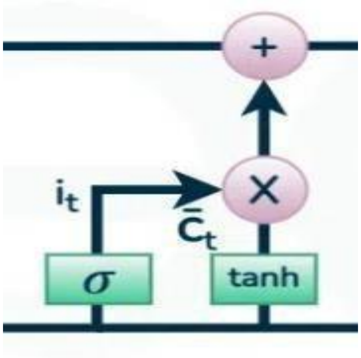


**Fig. 4**. Representation of Input gate

$$i_t \quad = \quad \sigma(W_i.[h_{t-1}, \quad x_t] \quad + \quad b_i$$

Where :

$i_t$ : Input gate output (a value between 0 and 1 indicating how much new information to allow in).

$W_i$ : Weight matrix for the input gate. $b_i$

: Bias for the input gate.

2.**a) Cell State ($\boldsymbol{C_t}$)**: This represents the memory of the network and carries information throughout the sequence. The cell state can be modified via input and forget gates.

$$C_t' = tanh(W_C.\,[h_{t-1},x_t] + b_C)$$

$$C_t \quad = \quad f_t \quad .C_{t-1} \quad + \quad i_t \quad . \quad C_t$$

Where :

$C_t'$ : Candidate memory content (new information to be added to the cell state).

$W_C$ : Weight matrix for candidate memory update.

$b_C$ : Bias for the candidate memory update. $C_t$ : Updated cell state (current memory of the network).

$f_t$ : Forget gate output (decides what to retain from the previous cell state).

$C_{t-1}$ : Previous cell state.

$i_t$ : Input gate output (controls how much new information to add).

3. **Output Gate** (***$o_t$***The task of extracting useful information from the current cell state to be presented as output is done by the output gate. The hidden state influences the output of the LSTM at the current time step.

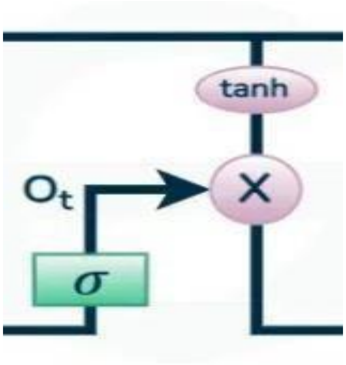


**Fig. 5**. Representation of Output gate

$$o_t = \sigma(W_o.[h_{t-1},\ x_t] + b_o)$$

$$h_t = o_t \ . \ \tanh \ (C_t)$$

Where :

$o_t$ : Output gate output (a value between 0 and 1 indicating how much of the cell state should influence the hidden state).

$W_o$ : Weight matrix for the output gate.

$b_i$ : Bias for the output gate.

$h_t$ : Updated hidden state (the output of the LSTM

$C_t$ : Updated cell state at the current time step). tanh

: Hyperbolic tangent activation function (produces values between -1 and 1).

# 4. **METHODOLOGY:**

The methodology adopted for this project involves several steps to ensure efficient and accurate prediction of stock prices using machine learning models. This section outlines the step-by-step process followed in building the stock price prediction system.

## 4.1. Data Collection

The first step involved collecting historical stock price data. Data was sourced from Yahoo Finance using the yfinance Python library. The dataset included daily stock prices for a period of 8 years, with the following key features:

- Open Price
- High Price
- Low Price
- Close Price
- Volume

These features were chosen because they represent crucial aspects of stock price movements that may influence future trends.

## 4.2. Data Preprocessing

Before training the models, the dataset was preprocessed to make it suitable for analysis :

- **Handling Missing Data**: Any missing values in the dataset were handled appropriately, either by imputing or removing them.
- **Data Normalization**: The stock price data was normalized using MinMaxScaler from the sklearn library to scale the values between 0 and 1. This ensured that features with larger numerical ranges did not dominate the learning process in the models.

## 4.3. Model Selection

For this project, two models were chosen: **Linear Regression** and **Long ShortTerm Memory (LSTM)** neural networks.

- **Linear Regression**: This is a basic and interpretable statistical model that assumes a linear relationship between the input features and the target variable (stock prices). The regression equation is of the form:

  $$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n$$

  Where $\beta_0, \beta_1, \beta_2, \ldots, \beta_n$ are the model parameters.

- **LSTM (**Long Short-Term Memory**)**: LSTM is a type of Recurrent Neural Network (RNN) that is particularly well-suited for sequential data like stock prices. It captures both short-term and long-term dependencies using a combination of gates (forget gate, input gate, and output gate) and maintains a memory cell to store relevant information across time steps.

## 4.4. Training the Models

Training dataset used is APPLE data which is from Technology sector. The training dataset is from the period of 31 JULY 2016 TO 2024 October 01 and it contains the closing price of 2984 days. The training data ranges between 39.31 and 226.21. The extracted data was then subjected to normalization to unify the data range within 0 and 1. Normalization of data is done to bring all stock data into a common range. Since we are using stock data from different market, we need the data to be under a common range

- **Training Linear Regression**: The dataset was split into training and testing sets, with majority of the data used for training (around 80%). The Linear Regression model was trained on historical stock price features like closing price, moving averages, and volume. The model was evaluated using metrics such as Mean Squared Error (MSE), Root mean square error (RMSE), Mean Absolute Error (MAE), and $R^2$ score.
- **Training LSTM**: The LSTM model was implemented using TensorFlow and Keras libraries. The sequential nature of the stock price data was captured by providing the model with a window of past stock prices as input. The model was trained using 100 epochs and evaluated using similar

performance metrics. The Adam optimizer and the MSE loss function were employed for training.

## 4.5. Model Evaluation

The performance of both models was evaluated using the following metrics:

- **Mean Squared Error (MSE)**: Measures the average squared difference between actual and predicted values.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - Y_i')2$$

- **Mean Absolute Error (MAE)**: Measures the average magnitude of errors in predictions.

$$MAE= \frac{1}{}\sum{}^{n}{}_{i=1} | Y_i - Y_i'|$$

- $R^2$ **Score**: Indicates how well the model explains the variance in the target variable.

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \overline{y})^2}$$

Where $SS_{res}$ is the sum of squared residuals and

$SS_{tot}$ is the total sum of squares, given by

$SS_{res} = \sum_{i=1}^{n}(Y_i - Y_i')^2$ and

$SS_{tot} = \sum_{i=1}^{n}(Y_i - \bar{Y})^2$

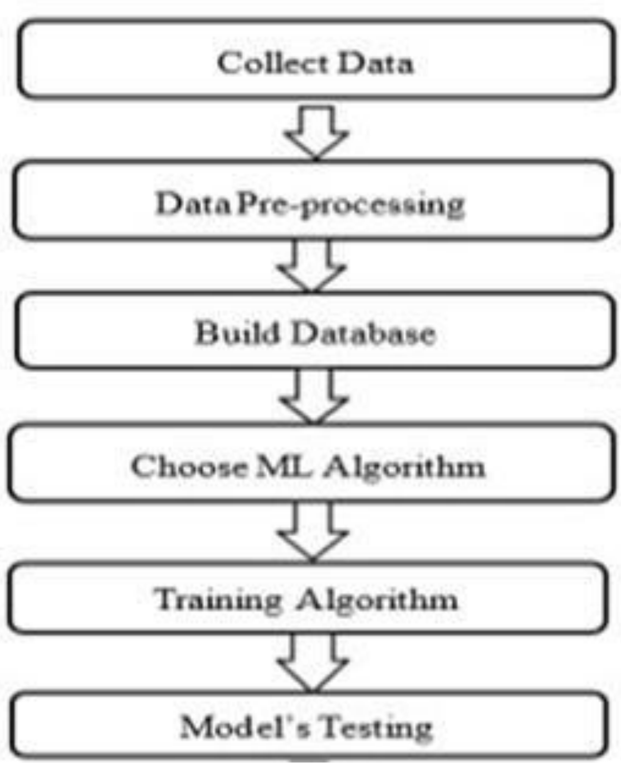
Here : $Y_i$ is the actual value of the dependent variable

$Y_i'$ is the predicted value from the model

$Y$ is the mean of the actual values

## 5. FLOW OF STOCK PRICE PREDICTION

Collecting data is the initial step. The data is fetched from the dependable source site Yahoo- Finance. Another crucial stage is data preprocessing, which involves cleaning up inaccurate or inconsistent data and transforming it into a format that is understandable to the average user using the Python pandas module. For the backends ’multiple operation execution, NumPy, tensorflow, and matplotlib are employed. The prediction part is performed using linear regression and LSTM techniques of machine learning.



**Flow chart**

## 6. RESULTS AND DISCUSSIONS

we implemented a Linear Regression and LSTM models to predict stock prices and developed a user-friendly website to provide these predictions to users. The models were trained on historical stock opening price data. The accuracy of the predictions was evaluated using various metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), RMSE to Mean Ratio, $R^2$ (Coefficient of Determination)**.**

6.1. **Model Accuracy**- Both the models demonstrated promising accuracy in predicting stock prices. The MSE, RMSE, MAE, and $R^2$ values, calculated on the test dataset (APPLE), were found to be within an acceptable range, indicating that the models were able to capture the underlying patterns in the data effectively.

| **Model** | MSE | RMSE | MAE | $R^2$ |
|---|---|---|---|---|
| Linear Regression | 105.32 | 10.26 | 9.24 | -4.76 |
| LSTM | 31.16 | 5.58 | 4.42 | -0.61 |

Table 1. Comparison of the accuracy measuring metrics between Linear regression and LSTM for Short-term (Jan-Mar 2024).

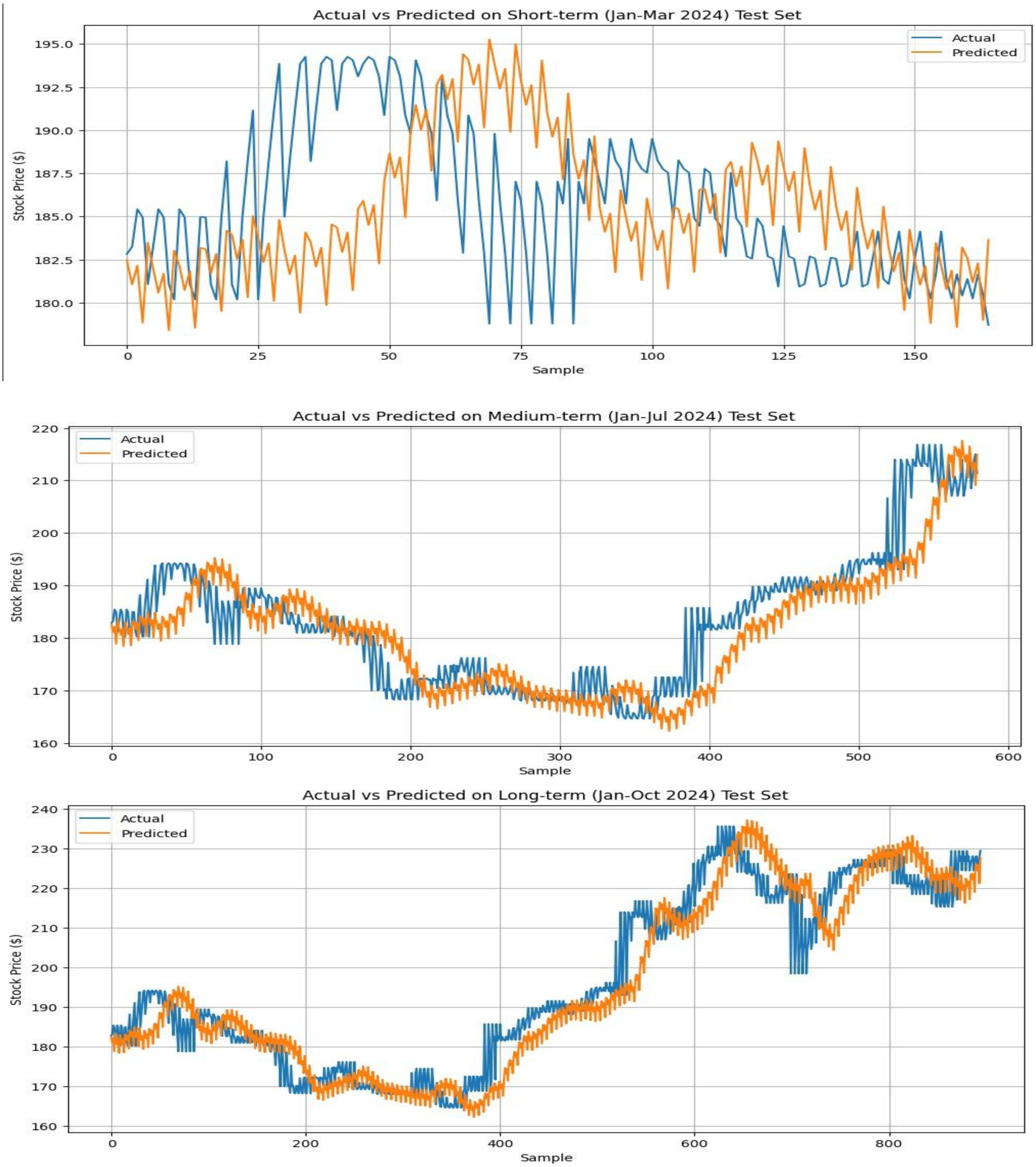
| **Model** | MSE | RMSE | MAE | $R^2$ |
|---|---|---|---|---|
| Linear Regression | 327.66 | 18.10 | 15.74 | -0.98 |
| LSTM | 40.21 | 6.34 | 4.74 | 0.75 |

Table 2. Comparison of the accuracy measuring metrics between Linear regression and LSTM for Medium-term (Jan-Jul 2024).

| **Model** | MSE | RMSE | MAE | $R^2$ |
|---|---|---|---|---|
| Linear Regression | 359.78 | 18.96 | 17.03 | 0.23 |
| LSTM | 49.59 | 7.04 | 5.40 | 0.89 |

Table 3. Comparison of the accuracy measuring metrics between Linear regression and LSTM for Long-term (Jan-Oct 2024).

The graph plots between actual and predicted values for both Linear regression and LSTM is shown below.



Actual vs Predicted on Short-term (Jan-Mar 2024) Test Set

Actual vs Predicted on Medium-term (Jan-Jul 2024) Test Set

Actual vs Predicted on Long-term (Jan-Oct 2024) Test Set

**Graph plots for LSTM**

### Analysis of Results of both models

The Linear Regression model produced satisfactory results. The model demonstrated its ability to capture the general trend of the stock prices but struggled with significant fluctuations or outliers.

The LSTM model outperformed the Linear Regression model. This model effectively captures the temporal dependencies in the stock price data due to its architecture, making it more suitable for time-series predictions. The LSTM's ability to learn from previous time steps contributed to more accurate predictions, particularly during volatile market conditions

6.2. **Prediction Performance**- The model's performance were evaluated for different stocks and time periods. It was observed that the accuracy of predictions varied based on the volatility and historical patterns of the specific stocks. Stocks with stable historical prices showed more accurate predictions compared to those with frequent fluctuations.

## 7. FUTURE SCOPE

- Information extraction utilizing natural language processing (NLP) from news articles and other text sources. NLP can be used to learn about a company's financial performance, competitive landscape, and other factors that could affect its stock price.

- The use of machine learning to predict the impact of non-financial events on stock prices. Political scandals or natural disasters are two examples of nonfinancial events that can significantly affect stock prices. The impact of these occurrences on stock prices can be predicted using machine learning, allowing investors to lower their risk exposure

- Overall, there is a lot of promise in using machine learning to predict stock prices. Accuracy of machine learning algorithms is increasing, and there is more data accessible to train them.

- Furthermore, fresh machine learning algorithms are always being developed. These factors suggest that future stock price forecasts will be more accurate.

## 8. CONCLUSION

In this work we used two machine learning models for the stock price prediction. Here we trained Linear Regression and LSTM models with the stock price of Apple. The models are capable of identifying the patterns existing in the stock markets. According to a survey of academic articles, selecting the right dataset is essential for accurate stock market prediction using linear regression. In the proposed work, LSTM is outperforming Linear Regression model, as it able to understand the non-linear structures present in the dataset and also it can better identify the underlying dynamics within various time series.

## 9. REFERENCES

[1] Vedashree Bhat(Oct-2023) – “Stock prediction using Linear Regression – A machine learning algorithm”. International journal of progressive research in engineering management and science (IJPREMS)

https://www.ijprems.com/uploadedfiles/paper/issue_10_october_2023/32103/final/fin_ijprems1699080511.pdf

[2] Hiransha M., Gopalakrishnan E.A., Vijay Krishna Menon, Soman K.P. (2018) – “NSE stock market prediction using Deep learning models”. International conference on computational intelligence and data science (ICCIDS)

https://www.sciencedirect.com/science/article/pii/S1877050918307828

[3] M. Umer, M. Awais, M. Muzammul (2019)- “Stock market prediction using machine learning (ML) algorithms”. Advances in Distributed Computing and Artificial Intelligence Journal (ADCAIJ).

https://www.researchgate.net/publication/342225624_Stock_Market_Prediction_Using_Machine_LearningMLAlgorithms

[4] https://www.geeksforgeeks.org/ml-linear-regression/

[5] https://www.geeksforgeeks.org/deep-learning-introduction-to-long-shortterm-memory/