Transactions on Banking, Finance, and Leadership Informatics

"Volume 1, Issue 2, Year 2025"

website: https://www.c5k.com

Research Article

Forecasting Stock Prices: A Machine Learning-Based Approach for Predictive Analytics Through a Case Study

Mohammad Shahidullah[1,*], Fahad Ahmed[2], Mst Shurovi Akter[1]

[1]*Department of Business Administration, International American University, Los Angeles, CA 90010, USA;*
[2]*Department of Science in Engineering Management, Trine University, Indiana, USA;*

**Corresponding Author: shahidbd2004@gmail.com*

ARTICLE INFO

Article history:
6 Sep 2025 (Received)
17 Oct 2025 (Accepted)
25 Oct 2025 (Published Online)

Keywords:
Machine Learning, Deep Learning, SMA, EMA, RSI, MACD, Bollinger Bands, RFE, Random Forest Regressor, Multivariate Analysis, LSTM.

ABSTRACT

Stock price prediction has always been a challenging task, requiring careful observation of trends and dynamics of the market because of the volatile and complex nature of financial markets. Various factors affect market behavior all the time. Even some unquantifiable factors like emotions of the masses, social and political dynamics, etc., also play a great role. So perfect prediction of stock prices is next to impossible. But taking other quantifiable factors and their behaviors into consideration is crucial for better prediction of the ups and downs of prices. Various machine learning and deep learning models have been proposed to tackle the challenges by capturing and interpreting complex patterns and relationships in historical price data. Technical features are important for understanding market trends and thus improving the accuracy of stock price predictions. In this paper, we calculate key technical indicators such as Simple Moving Average (SMA), Exponential Moving Average (EMA), Relative Strength Index (RSI), Moving Average Convergence Divergence (MACD), Bollinger Bands, and others. We then focus on selecting the most relevant indicators by employing feature selection methods from these to enhance the extraction of meaningful features reflecting underlying market behavior and increase the probability of more precise prediction. Here, Recursive Feature Elimination (RFE) and Random Forest Regressor-based importance ranking methods have been applied for the feature selection task. To get a better forecast of market price, it is important to capture long-term dependencies and patterns over time. Long Short-Term Memory (LSTM) networks are well-suited for modeling and predicting sequential data like stock prices. By leveraging an LSTM model and taking the selected features, we do a multivariate analysis to forecast stock price based on historical data, identifying the trends fairly accurately with some lags here and there.

Transactions on Banking, Finance, and Leadership Informatics (TBFLI), C5K Research Publication

1. Introduction

The prediction of stock prices has long been a subject of interest and importance for investors, financial analysts, and economists. Accurate forecasting can lead to significant financial gains, while inaccurate predictions may result in substantial losses. In recent years, advancements in machine learning (ML) techniques have opened new avenues for stock price prediction, particularly through models designed to handle time series data. One of the most promising approaches in this domain is Long Short-Term Memory (LSTM) networks, a type of recurrent neural network that has shown exceptional potential in capturing patterns in time-dependent data, such as stock prices. Stock prices are dependent on numerous factors and thus make their prediction rather complex. Unlike univariate time series analysis, which considers only a single dependent variable, multivariate analysis accounts for several interconnected features. The multivariate approach is particularly effective in stock price predictions, as the other variables like opening, highest, and lowest prices, etc., influence the stock closing price significantly. In this context, technical indicators are often used as features for multivariate models. As these indicators are derived from historical price and volume data, they give clear insights into trends and price movement. As a result, incorporating these indicators helps the model to capture additional patterns that can lead to more accurate predictions and capture of trends in the future, which may not be evident from the closing price data only, which is used as the only feature in univariate models.

Several studies have demonstrated the advantages of LSTM models in stock market analysis, emphasizing their ability to capture the complex relationships and volatility inherent in stock prices. For instance, (Orsel & Cain, 2022) found that LSTM models outperform simpler algorithms, such as Kalman filters, especially for high-volatility stocks like Tesla (TSLA),

*Corresponding author: shahidbd2004@gmail.com (Mohammad Shahidullah)

Cite: Mohammad Shahidullah, Fahad Ahmed, Mst Shurovi Akter (2025). Forecasting Stock Prices: A Machine Learning-Based Approach for Predictive Analytics Through Case Study. *Transactions on Banking, Finance, and Leadership Informatics,* 1(2), pp. 1-XY.

highlighting their robustness in handling intricate time series data. Similarly, (Liu et al., 2022) showed that LSTM models achieved over 90% accuracy in predicting stock prices, further solidifying their reliability in real-world scenarios. Given the complexity of stock market behavior, numerous researchers have proposed hybrid models that combine LSTM with other techniques, such as convolutional autoencoders (CAE) and principal component analysis (PCA), to improve prediction accuracy and generalization. These methods enhance the model's ability to extract valuable features from high-dimensional data, leading to more accurate forecasts. By leveraging these advanced techniques, machine learning-based models, particularly LSTM, are becoming indispensable tools in financial forecasting and automated portfolio management.

In this paper, we explore the application of LSTM network for stock price prediction. We make multivariate analysis by taking multiple features as the input of the LSTM architecture while keeping the model as simple as possible. Multivariate LSTM models are challenging to develop due to the involvement of several parameters in high-dimensional space. Also feature selection is necessary to deduce which ones are to keep based on their influence on the analysis. Here, we compare the model performance with traditional methods such as Naive Forecast. Through a comprehensive analysis of historical stock prices, we aim to evaluate the efficacy of this model in capturing the underlying trends in the market.

2. Dataset Description

The dataset comprises stock price data for Microsoft (MSFT) which is downloaded using ‘yfinance’. The data ranges from January 4, 2010, to December 29, 2023, encompassing over 3500 trading days. It includes Date which is the index of the data table and ‘Open’, ‘High’, ‘Low’, ‘Close’, ‘Adj Close’ and ‘Volume’ columns. This dataset gives a detailed description of MSFT’s historical stock performance and with this dataset stock price forecasting and analyzing trends over a 14-year period is done.

3. Methodology

In this paper, stock price prediction is done using multivariate analysis and an LSTM-based model to track market trends based on Microsoft (MSFT) stock. We divide the methodology into multiple key steps: data collection, feature engineering, feature selection, model training, and evaluation.

3.1. Data Collection

As we have given the dataset description before, the stock data for Microsoft (MSFT) collected from Yahoo Finance covers the period from January 2010 to December 2023. The dataset includes the following columns: Open, High, Low, Close, Adjusted Close prices, and Volume.

3.2. Data Visualization

We perform several visualizations to better understand the dataset:

- Bar Plot for Trading Volume vs Date

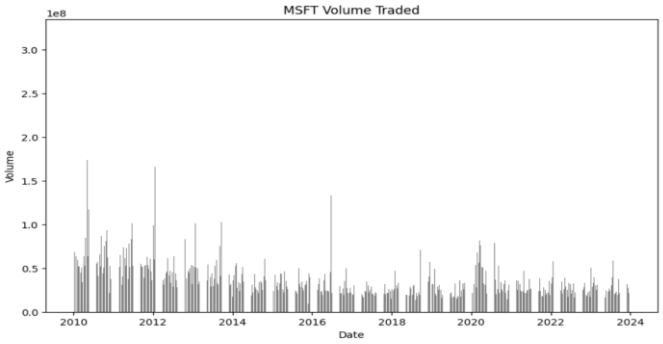

Fig.1. MSFT Volume Traded over Time

- Line Plots for 'Open', 'High', 'Low', 'Close', and 'Adjusted Close' prices vs Date.

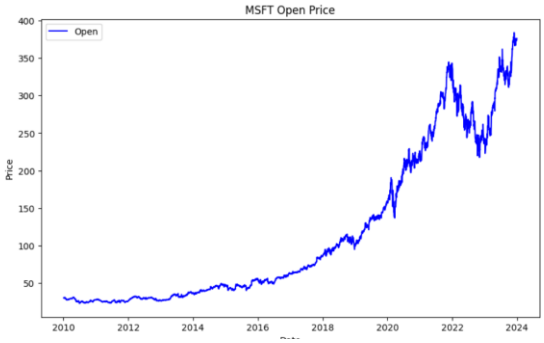

Fig.2. MSFT Open Price over Time

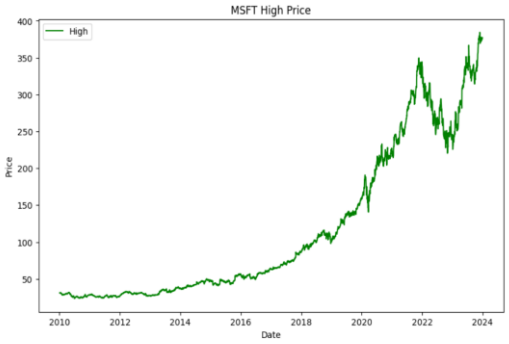

Fig. 3. MSFT High Price over Time

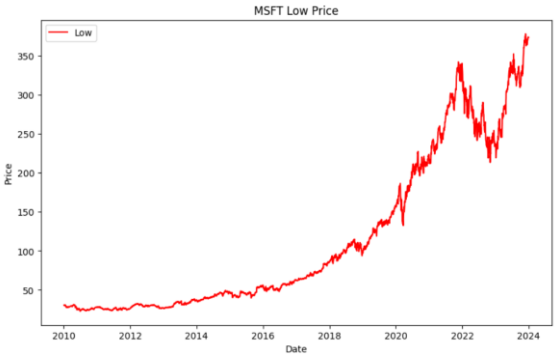

Fig. 4. MSFT Low Price over Time

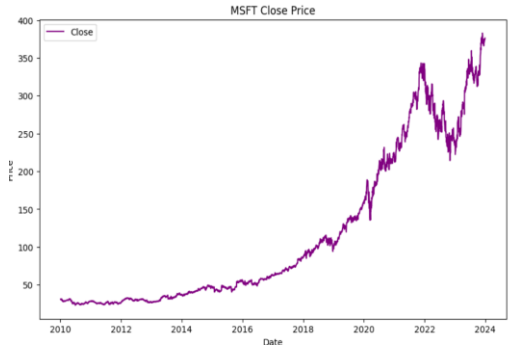

Fig. 5. MSFT Close Price over Time

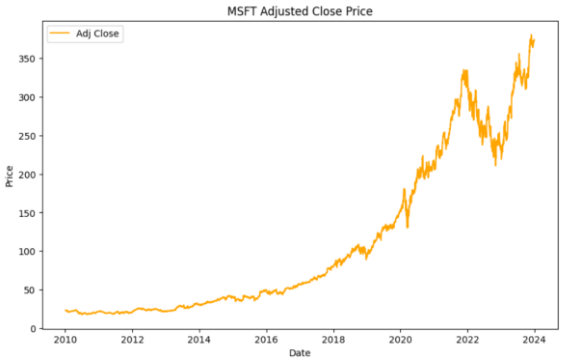

Fig. 6. MSFT Adjusted Close Price over Time

3.3. Technical Indicators

We calculate different technical indicators to incorporate these indicators in the analysis. These indicators can represent different aspects of market behavior:

1. Simple Moving Average (SMA):

Simple Moving Average (SMA) refers to a stock's average closing price over a specified period. The reason the average is called "moving" is that the stock price constantly changes, so the moving average changes accordingly. Smoothing out the short-term fluctuation, it can provide a clear view of the trend.

$$SMA_t = \frac{1}{n} \sum_{i=0}^{n-1} P_{t-i} \qquad (1)$$

Where $\mathbf{SMA_t}$ is the simple moving average at time t. $\mathbf{P_t}$ is the closing price at time **t,** $\mathbf{P_{t-i}}$

is the closing price at time t-i, and **n** is the window length.

SMA is computed using a window of 20-days, that means n = 20 here. So, for each

day, the average closing price of the previous 20 days is calculated.

2. Exponential Moving Average (EMA):

Putting more emphasis on the recent data points like the latest prices, EMA can be more responsive to new information than SMA. It can show how price changes over a certain period of time.

$$EMA_t = EMA_{t-1} + \alpha \times (p_t - EMA_{t-1}) \qquad (2)$$

Where $\alpha = \frac{2}{n+1}$

Here, $\mathbf{EMA_t}$ is the exponential moving average at time t. $\mathbf{P_t}$ is the closing price at time t,

$\mathbf{EMA_{t-1}}$ is the EMA of the previous day, $\boldsymbol{\alpha}$ is the smoothing factor, and **n** is the window

length. In this case, a 20-day window is used to calculate EMA.

3. Relative Strength Index (RSI):

RSI is a momentum oscillator that measures the speed and change in price movements which can range from 0 to 100. It can detect oversold or overbought conditions and help identify potential reversal points.

$$RSI = 100 - \left(\frac{100}{1 + \frac{Average\ Gain}{Average\ Loss}}\right) \qquad (3)$$

Here, the RSI is calculated using a 14-day window. The RSI is determining whether a stock has been overbought (RSI > 70) or oversold (RSI < 30).

4. Moving Average Convergence Divergence (MACD):

MACD gives the difference between the 12-day and 26-day EMAs, along with a 9-day EMA signal line. MACD is mainly a trend-following momentum indicator and

captures trends in stocks.

$$MACD = EMA_{12} - EMA_{26} \qquad (4)$$

Here, we calculate the two EMAs using n = 12 and n = 26 and then differentiating them we get the MACD for our data where n is the window length to evaluate EMA.

5. Bollinger Bands:

Consisting of a middle band (20-day SMA) and two outer bands, Bollinger bands can provide insight of market price levels and volatility.

$$\text{Upper Band} = \text{Middle Band} + 2 \times \text{Standard Deviation} \qquad (5)$$

$$\text{Lower Band} = \text{Middle Band} - 2 \times \text{Standard Deviation} \qquad (6)$$

6. Stochastic Oscillator (%K and %D):

This oscillator is used to compare the closing price to its range over a specified period of time. This one can also identify oversold or overbought phases. The primary calculation is done with %K, and if a smoother version is needed, then %D is calculated.

$$\%K = \frac{(P - L_{14})}{H_{14} - L_{14}} \times 100 \qquad (7)$$

Where **P** is the closing price, $\mathbf{L_{14}}$ is the lowest price over the past 14 days, and $\mathbf{H_{14}}$ is the highest price over the past 14 days.

$\%D = SMA_3(\%K)$ (8)

This implies that %D is the average of %K values over the past three days of time.

7. True Range (TR) & Average True Range (ATR):

Considering the most significant price difference between the current high, current low, and the previous day's close prices, True Range can measure market volatility over a certain period of time.

$TR = \max(H_t - L_t, |H_t - C_{t-1}|, |L_t - C_{t-1}|)$ (9)

Where $\mathbf{H_t}$ is the current high, $\mathbf{L_t}$ is the current low, and $\mathbf{C_{t-1}}$ is the previous close.

The Average True Range (ATR) is the smoothed moving average of the True Range (TR) of a 14-period window typically.

$$ATR = \frac{1}{n}\sum_{i=0}^{n-1} TR_{t-i} \quad (10)$$

Where **n** is the window length (typically 14 days), and $\mathbf{TR_{t-i}}$ is the True Range for the i-th day. ATR also captures trends and market volatility like TR.

8. On-Balance Volume (OBV):

OBV collects trading volume based on price shifts, thus giving information about buying and selling pressure.

$$OBV = \text{Previous OBV} + \begin{cases} V_t \; if \; P_t > P_{t-1} \\ -V_t \; if \; P_t < P_{t-1} \\ 0 \; if \; P_t = P_{t-1} \end{cases} \quad (11)$$

where $\mathbf{V_t}$ is the trading volume at time t, and $\mathbf{P_t}$ is the price at time t.

3.4. Feature Selection

As we calculate the technical indicators such as SMA, EMA, RSI, MACD, Bollinger Bands, Stochastic Oscillators, TR, ATR, and OBV, we add them to the database as new feature columns. Now, the dataset has 21 columns which include the previous columns that were present from the beginning and the newly added columns. However, using all the features is not feasible because it is not time efficient and will not yield good results as not all the features have the same influence on the closing price of the stocks. So, we need to find the features that are best suited to make the stock prediction using our LSTM model.

In this paper, we perform feature selection using two methods:

1. Recursive Feature Elimination (RFE):

Recursive Feature Elimination is a process that eliminates the least significant features recursively based on the performance of the specified model. Here, we utilize RFE with a Random Forest Regressor as the estimator. The features selected for RFE are 'SMA', 'EMA', 'RSI', 'MACD', 'Signal Line', 'Low', 'High', 'Volume', 'Adj Close', 'Open', 'Close', 'Middle Band', 'Upper Band', 'Lower Band', '%K', '%D', 'ATR', and 'OBV'. Then with 'StandardScaler', the features are standardized to keep them on the same scale. We initialize the RFE with the Random Forest Regressor and select the number of features to be 6. Then, fitting the RFE selector to the scaled feature data, we retain the most influential features, which are 'Low', 'High', 'Adj Close', 'Open', 'Close', and 'Middle Band'. To evaluate the model's performance, we employ mean squared error (MSE) as the scoring metric.

2. Feature Importance with Random Forest Regressor:

Feature importance evaluation focuses on the contribution of each feature to the prediction. We use the Random Forest Regressor to analyze the importance scores given to each feature. Similar to RFE, we use the same feature set and standardize in the same way using 'Standard Scaler' to ensure uniformity in feature scales. We train the regressor on the standardized feature data with 340 estimators to compute the significance of each feature. Now the feature importance extraction is done, and we get 'Adj Close', 'High', 'Low', 'Open', 'Close', 'SMA', 'EMA', and 'Middle Band', which are the most influential features.

From these, we select 'SMA', 'Adj Close', 'High', 'Low', 'Open', 'Close', and 'Middle Band' as the input features for our LSTM model to obtain better predictions.

3.5. Train, Test, and Prediction Data Split and Visualization

First, we scale the dataset with the selected features with 'MinMaxScaler' to normalize the dataset to a range of 0 to 1. Then the total dataset is divided into 3 parts. The first 80% of the scaled data is for training, the next 10% is for testing and the remaining 10% is for prediction. The dataset is then reshaped into the expected input format for our LSTM model where the input dimension is structured as (samples x timesteps x features). We take the look-back period 30 days, and the prediction horizon 1 day ahead. The reshaped data dimensions are as follows:

X_{train} = (n_samples, 30, n_features), y_{train} = (n_samples, 1). The X_{test} and y_{test} also follow the same structure. $X_{predict}$ is reshaped in the same way. Here, n_features = 7 as we have taken 7 features for the forecasting task.

We are showing the plots for 'SMA' and 'Middle Band' only here. Rest is divided in the same way.

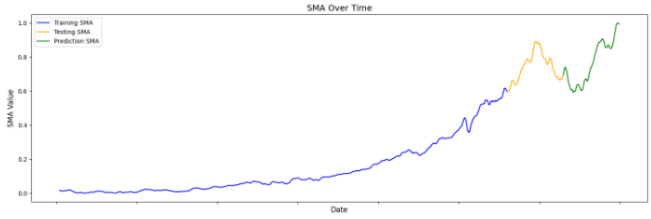

Fig. 7. SMA over Time with data split

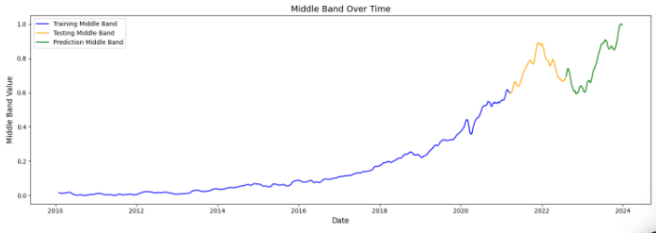

Fig. 8. Middle Band over Time with data split

3.6. Model Architecture

We design a simple LSTM model to predict the stock prices based on historical data. It has a sequential architecture with the following elements:

LSTM Layer:

- Units: 220 neurons
- Activation Function: ReLU to introduce non-linearity and capture complex patterns of the data
- Input Shape: The input data shape is (30, n_features), where 30 refers to the look-back period.
- Return Sequence: Set to False, to make sure LSTM layer's output is a single vector for each sequence.

Dropout Layer:

- Rate: Set to 0.5 to prevent overfitting.

Optimizer:

- Adam Optimizer with a learning rate of 0.0001.

3.7. Training

The model is trained for 50 epochs with a batch size of 16. 20% of the training data is used for validation to evaluate the model performance during training. We implement Early Stopping with a patience of 15 epochs to stop training once validation loss stops improving so the model doesn't overfit.

4. Model Evaluation Metric

Error measures are computed to assess the model's feasibility. We employ MSE, RMSE, MAE, and MAPE metrics to evaluate the model on the test data. Here, y_i represents the actual value, and $\hat{y}_i$ represents the predicted value of the stock price at the i-th observation.

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

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \qquad (12)$$

Root Mean Squared Error (RMSE): By calculating the square root of the MSE, RMSE can be obtained. It can help with the direct interpretation of the error in the same units as the target variable.

$$\text{RMSE} = \sqrt{MSE} \qquad (13)$$

Mean Absolute Error (MAE): MAE assesses the average magnitude of errors; thus, it can provide a more intuitive idea of the overall deviation from the actual values.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \qquad (14)$$

Mean Absolute Percentage Error (MAPE): MAPE calculates the error percentage with respect to the actual values, so it can help with the relative prediction accuracy.

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| x100 \qquad (15)$$

Here **n** is the number of samples to observe.

Symmetric Mean Absolute Percentage Error(SMAPE): SMAPE is a common metric for evaluating forecast accuracy. It measures the percentage difference between the predicted and actual values, but unlike MAPE, it accounts for symmetry. This means it gives equal penalty to both over- and under-forecasts. The formula for SMAPE is:

$$\text{SMAPE} = \frac{100\%}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{(|y_i| + |\hat{y}_i|)/2}$$

Where:

- y_i is the actual value.
- $\hat{y}_i$ is the predicted value.
- n is the total number of data points.

(Write this down equation)

R² (R-Squared or Coefficient of Determination): R² measures the proportion of variance in the actual values that is explained by the predicted values. It quantifies how well the predictions match the real data. A value of 1 indicates a perfect fit, and 0 indicates no correlation between the model and the actual data.

Formula:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Where:

- y_i = actual values
- $\hat{y}_i$ = predicted values
- $\bar{y}$ = mean of the actual values
- n = number of data points

MBD (Mean Bias Deviation): MBD measures the average bias in the forecast, indicating whether the model tends to overestimate or underestimate. A positive value indicates overestimation, while a negative value indicates underestimation.

Formula:

$$MBD = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{y_i}\right) \times 100$$

Where:

- y_i = actual values
- $\hat{y}_i$ = predicted values
- n = number of data points

Hitting Rate (HR): Hitting Rate measures the proportion of predictions that fall within a specified tolerance range of the actual values. It reflects how often the forecast hits the target within an acceptable margin of error.

Formula:

$$HR = \frac{\text{Number of hits}}{n}$$

Where:

- A hit refers to a prediction that falls within a certain tolerance range of the actual value.
- n = total number of data points

Thiel's U Statistic: Thiel's U Statistic compares the forecasting model to a naïve benchmark model. If U is less than 1, the model outperforms the naïve approach. If U is greater than 1, the model performs worse than a simple forecast.

Formula:

$$U = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n} y_i^2} + \sqrt{\frac{1}{n}\sum_{i=1}^{n} \hat{y}_i^2}}$$

Where:

- y_i = actual values
- $\hat{y}_i$ = predicted values
- n = number of data points

These metrics were carefully selected to ensure that the model is robust in both large and small scales.

5. Result and Analysis

The evaluation of time series data has to be robust and consistent over both small scale and large scales as there could be instances where the overall performance of the model is good but within a specific timeframe the model might perform poorly which is a big concern that stock forecasting is usually done over a relatively short period thus it needs to be reliable in long terms to ensure that the model will be able to keep with the market trend unless a an unanticipated change occurs and also in short terms to ensure that the model will provide predictions with an acceptable reage of error.

The predictions made by the model for two time spans are shown in the following graph in comparison to both the actual value and the Naïve Forecast.

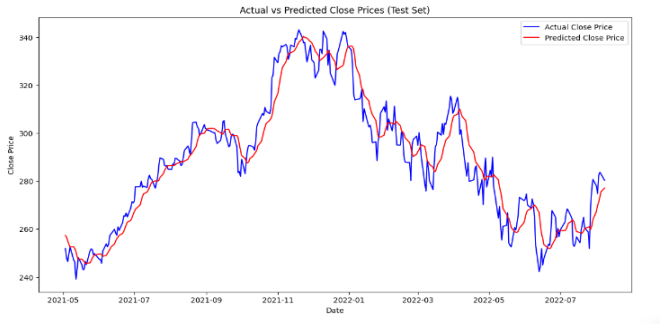

Fig. 9. Actual Vs Predicted Close Prices (Test Set) over Time

From Figure 9, it is seen that the model closely follows the trend of the actual price of the stock with reasonable accuracy and deviation. To ensure the validity of the model, it is also plotted against the Naïve Forecast, which acts as the benchmark for prediction.

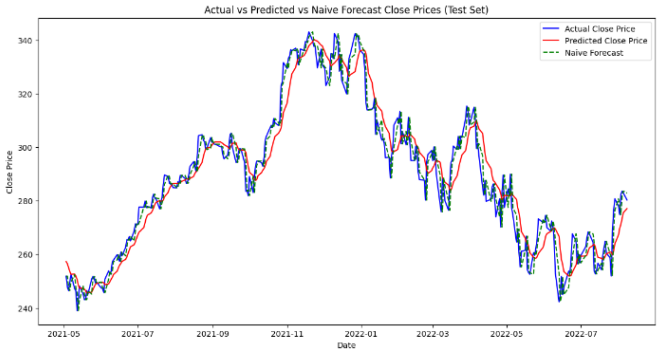

Fig. 10. Actual Vs Predicted Vs Naïve Forecast Close Prices (Test Set) over Time

It is clear from Figure 10, the prediction made by the model is able to find the overall trend and properly follow it when it is plotted for the range of 30 days with some lag.

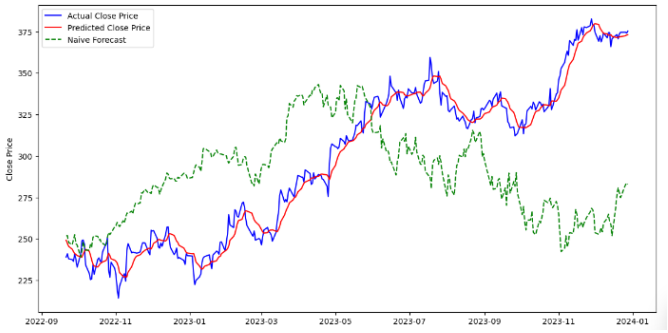

Fig. 11. Actual Vs Predicted Vs Naïve Forecast Close Prices (Predict Set) over Time

Figure 11 shows that for long-term prediction, the model correctly keeps up with the actual data, whereas Naïve Forecast fails as it follows the previous data, thus it veers off rapidly. This plot shows that the model predicts based on the data it has learned and does not follow the data blindly.

To ensure the robustness of the model, it is now subject to K-fold cross-validation for K = 30. The model is subject to such cross-validation for determining the performance of the model as well as the validity of the predictions it makes.

Table 1. Performance of the model's performance using multivariate analysis

Fold	MSE	RMSE	MAE	SMAPE(%)
01	0.000052	0.007177	0.004826	3.970769
02	0.000034	0.005859	0.003611	2.957225
03	0.000062	0.007850	0.004547	4.380218
04	0.000064	0.007975	0.005162	4.275031
05	0.000041	0.006423	0.004454	4.976192
06	0.000063	0.007930	0.005214	4.616836
07	0.000074	0.008608	0.004528	3.894674
08	0.000029	0.005356	0.003635	4.169000
09	0.000100	0.009988	0.005211	4.047574
10	0.000035	0.005895	0.003776	4.108608
11	0.000042	0.006445	0.003890	2.963958
12	0.000046	0.006785	0.004015	3.895295
13	0.000058	0.007607	0.004316	3.279109
14	0.000079	0.008872	0.004858	3.931708
15	0.000053	0.007299	0.004467	4.841501
16	0.000032	0.005672	0.003934	3.552614
17	0.000032	0.005653	0.003899	3.766811
18	0.000018	0.004292	0.003273	4.285424
19	0.000121	0.010978	0.005444	3.490777
20	0.000062	0.007882	0.004500	5.116213
21	0.000056	0.007464	0.004693	5.544284
22	0.000062	0.007869	0.004531	4.912676
23	0.000034	0.005820	0.003567	4.632146
24	0.000033	0.005731	0.003703	4.273023
25	0.000080	0.008957	0.005134	4.166605
26	0.000061	0.007802	0.004869	7.306278
27	0.000050	0.007101	0.004531	3.651303
28	0.000056	0.007510	0.004805	3.877002
29	0.000046	0.006801	0.003984	3.175038
30	0.000080	0.008960	0.005528	4.804610

Average MSE: 0.0001

Average RMSE: 0.0073

Average MAE: 0.0044

Average SMAPE: 4.23%

From the 30-fold cross-validation, it is seen that the errors tend to stay quite low for all instances. The interpretations of the data are as follows:

1. Mean Squared Error (MSE)

- **MSE** measures the average squared difference between predicted and actual values. It is sensitive to larger errors since it squares the differences. In your case, the **average MSE is 0.0001**, indicating that the squared errors are very small, which is good, especially for stock price prediction where precision is critical.

2. Root Mean Squared Error (RMSE)

- **RMSE** is the square root of the MSE, making it easier to interpret because it's in the same units as the target variable (stock price). The **average RMSE is 0.0073**, meaning that the model's predictions are off by approximately 0.73% of the stock price on average across all folds, which suggests the model is performing well in terms of error.

3. Mean Absolute Error (MAE)

- **MAE** gives the average of the absolute differences between predicted and actual values. Unlike MSE, it doesn't heavily penalize larger errors, making it a more direct measure of typical prediction accuracy. The **average MAE is 0.0044**, which indicates that the model's average prediction error is about 0.44% of the stock price. This also reflects high accuracy.

4. Symmetric Mean Absolute Percentage Error (SMAPE)

- **SMAPE** measures the relative accuracy between the predictions and actual values, expressed as a percentage. Your **average SMAPE is 4.23%**, which indicates that, on average, the model's predictions are off by 4.23% from the true values. In stock price forecasting, this is considered a good result, as small percentage errors are desirable due to the volatile nature of stock prices.

Summary of Results:

- **Low MSE and RMSE values** indicate that the model is producing highly accurate predictions, with very small deviations from the actual stock prices.
- **MAE** is similarly low, confirming that the average error is quite small.
- **SMAPE** being just over 4% further reinforces that the model is performing well, maintaining small percentage differences between actual and predicted values.

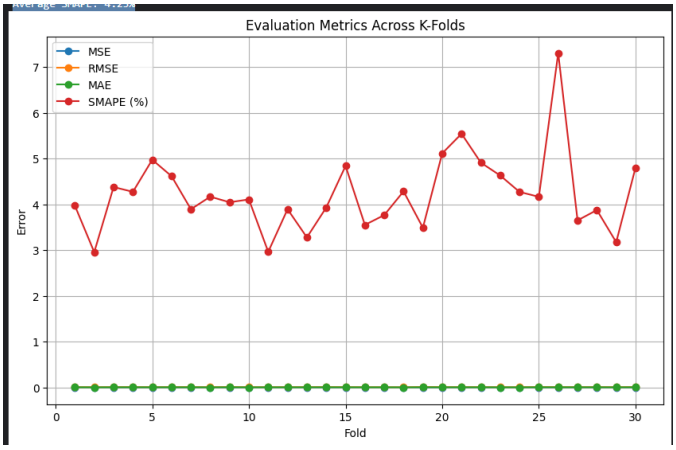

Fig. 12. Evaluation of the model for 30 folds.

Figure-12 shows the performance and it is seen that the model performs well and is robust enough for reliable stock prediction.

However since the evaluation is on a time series model it has to be also evaluated on the basis of nature of it's predictions as well since a model can be quite accurate overall and yet the

predictions can be unreliable. Table -1 illustrates the overall model's performance thus the analysis of the nature of the predictions is also shown in the following table.

Table 2. Performance of the model's performance using multivariate analysis

Fold	R^2	Explained Variance	SMAPE (%)	MBD	Hitting Rat	Thiel's U
01	0.998249	0.998249	3.552516	-0.000115	1.0	0.014521
02	0.996066	0.996957	4.411900	0.003939	1.0	0.020450
03	0.996651	0.996662	7.053538	0.000467	1.0	0.020715
04	0.995402	0.995933	4.196369	0.002922	1.0	0.022176
05	0.991376	0.993448	5.709523	-0.005325	1.0	0.031322
06	0.997588	0.997589	5.881562	0.000173	1.0	0.017559
07	0.995657	0.995813	4.768399	-0.001718	1.0	0.023134
08	0.996119	0.996187	5.384907	-0.001002	1.0	0.020082
09	0.994111	0.994821	5.000924	-0.004167	1.0	0.027086
10	0.995291	0.995417	4.898681	-0.001069	1.0	0.021394
11	0.996988	0.997319	4.000654	-0.002404	1.0	0.018478
12	0.996360	0.996593	4.743986	-0.002086	1.0	0.020828
13	0.997251	0.997263	5.337274	0.000601	1.0	0.018043
14	0.995882	0.995888	4.724390	-0.000350	1.0	0.021155
15	0.996546	0.996622	4.443983	-0.001291	1.0	0.021086
16	0.995815	0.996728	4.740691	-0.004243	1.0	0.021224
17	0.997024	0.997174	4.104781	0.001478	1.0	0.016760
18	0.998198	0.998199	5.625567	0.000057	1.0	0.014231
19	0.995341	0.995348	3.947382	-0.000419	1.0	0.023010
20	0.996642	0.996870	4.630232	0.002117	1.0	0.020205
21	0.997123	0.997638	8.167897	0.003286	1.0	0.018116
22	0.997177	0.997178	5.220905	-0.000143	1.0	0.018629
23	0.995711	0.995786	6.335526	-0.001068	1.0	0.023588
24	0.997331	0.997366	4.040515	0.000645	1.0	0.016238
25	0.990694	0.992609	7.377931	-0.007769	1.0	0.035200
26	0.995994	0.996008	7.203927	-0.000569	1.0	0.022850
27	0.997030	0.997684	4.060413	0.004558	1.0	0.019447
28	0.995459	0.996169	4.961718	-0.004235	1.0	0.023086
29	0.997717	0.997773	3.322415	-0.001151	1.0	0.016478
30	0.987442	0.988796	4.296245	-0.004363	1.0	0.037262

Table-2 shows the results of the nature of the predictions the model makes and the interpretation of them is as follows:

1. R^2 (R-Squared):

- R^2 values are consistently close to 1, ranging from 0.987 to 0.998, indicating that the model explains almost all the variance in the data for each fold. This suggests excellent model fit in terms of how well the model's predictions align with the actual values.

2. Explained Variance Score:

- Similar to R^2, this score measures the proportion of variance explained by the model. Values near 1 (between 0.988 and 0.998) confirm that the model performs well across all folds, consistently capturing most of the variance in the closing price predictions.

3. SMAPE (Symmetric Mean Absolute Percentage Error):

- SMAPE values range from **3.32% to 8.17%**, which is generally considered good to very good for time series data, especially in stock price forecasting where even small errors can have a significant impact. This low percentage indicates that the model's predictions are relatively close to the true values.

4. MBD (Mean Bias Deviation):

- MBD values fluctuate between -0.0077 and 0.0045, which are close to zero, indicating that the model has minimal bias across folds. The model does not systematically over- or under-predict the stock prices, which is ideal in forecasting tasks.

5. Hitting Rate:

- The hitting rate is **1.0** across all folds, suggesting that the model's predictions consistently fall within the chosen threshold (0.09). However, given that this threshold may be too large for stock price forecasting, this metric may need further refinement with a tighter or more dynamic threshold for more insightful results.

6. Thiel's U Statistic:

- Thiel's U statistic values are low (ranging from **0.014 to 0.037**). A value less than 1 suggests that the model is performing better than a naïve forecast (where future values would simply replicate previous values). The lower the Thiel's U value, the better the forecast; hence, these low values indicate that your model is making highly accurate forecasts compared to a naïve model.

Overall Analysis:

- The model is performing very well in terms of predictive accuracy, as reflected by the high R^2 and explained variance scores, low SMAPE, minimal bias (MBD), and low Thiel's U statistic. However, the hitting rate metric might need further tuning, especially with a different threshold that is more relevant for stock prices, where even small differences matter.

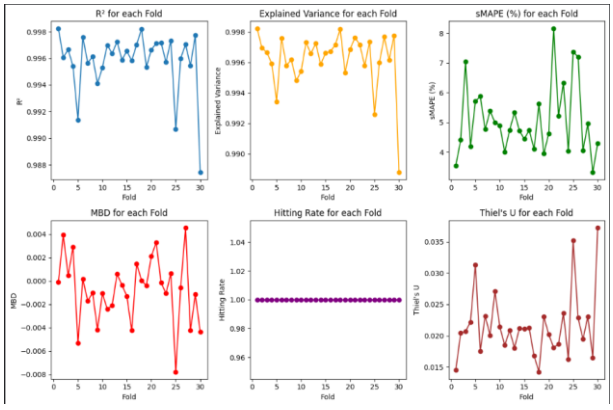

Fig. 13. Evaluation of the model's predictions for 30 folds.

Figure 13 illustrates the overall appropriateness of the model's predictions, that the model can indeed reliably forecast the price with good accuracy and without bias, and that it doesn't underpredict or overpredict the value but rather follows the trend and predicts correctly on all evaluation metrics. Which makes the model robust and accurate.

6. Conclusion

In this paper, we explored a multivariate LSTM model with multiple features as its input. Here, we selected the features from the feature selection methods using RFE and Random Forest Regressor-based ranking importance models. When compared to the naïve forecast, which gave better performance on the test data, the LSTM model we have proposed demonstrated greater efficiency in handling unseen data. Even though the naïve forecast performed well in the test set, it failed to predict properly for the future unseen data. But the LSTM model gave us more accurate predictions. So a multivariate LSTM model is a good choice for forecasting stock prices. With more robust feature selection models, more relatable features can be selected, which can optimize the model's performance even more. (Write the conclusion based on the result and analysis)

7. Remark

I would like to convey my utmost gratitude to Shuvankar Biswas and S.M. Tahmeed Reza for helping me with valuable suggestions and insights.

References

Liu, H., Qi, L., & Sun, M. (2022). Short-Term Stock Price Prediction Based on CAE-LSTM Method. *Wireless Communications and Mobile Computing*, *2022*(1), 4809632. https://doi.org/https://doi.org/10.1155/2022/4809632

Orsel, O., & Cain, S. (2022). *Comparative Study of Machine Learning Models for Stock Price Prediction*. https://doi.org/10.48550/arXiv.2202.03156