Scientific Research Publishing

Journal of Computer and Communications, 2025, 13(4), 180-196
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

A Comparative Analysis of the Performance of Machine Learning and Deep Learning Techniques in Predicting Stock Prices

Shimaa Ouf[1], Mona El Hawary[1*], Amal Aboutabl[2], Sherif Adel[3]

[1]Information System Department, Faculty of Commerce and Business Administration Helwan University, Cairo, Egypt
[2]Computer Science Department, Faculty of Computer and Artificial Intelligence, Helwan University, Cairo, Egypt
[3]Administration Department, Faculty of Commerce and Business, Helwan University, Cairo, Egypt
Email: *mona.medhat.pbis@commerce.helwan.edu.eg

How to cite this paper: Ouf, S., El Hawary, M., Aboutabl, A. and Adel, S. (2025) A Comparative Analysis of the Performance of Machine Learning and Deep Learning Techniques in Predicting Stock Prices. *Journal of Computer and Communications*, **13**, 180-196.
https://doi.org/10.4236/jcc.2025.134012

Received: March 12, 2025
Accepted: April 25, 2025
Published: April 28, 2025

Abstract

The Efficient Market Hypothesis postulates that stock prices are unpredictable and complex, so they are challenging to forecast. However, this study demonstrates that it is possible to predict stock prices with reasonable accuracy using machine learning (ML) and deep learning (DL) models with optimized parameters. This compares ML models, such as Random Forest (RF) and XGBoost, against deep learning models, such as Long Short-Term Memory (LSTM), in terms of the accuracy of their market forecasting over different time horizons. The above models are used to predict the Apple stock market prices captured from Yahoo Finance from 2015 to 2020. The primary purpose of this paper is to enhance the prediction accuracy by tuning hyperparameters to choose the best optimization parameters that fit every predictive model. The experimental part of this paper uses fixed value (default) parameters for each model compared to the use of tuned hyperparameters; it tries combinations of hyperparameters and evaluates their performance on a validation set. This is done to determine the extent to which the hyperparameters enhanced the accuracy of the predictions and their impact on the results. The LSTM model achieved higher accuracy and recorded the first rank. They lowered it from 5.22 to 3.82; XGBoost had the second-best reduction of RMSE from 0.79 to 0.65, and Random Forest had a low-rate reduction of RMSE from 28.12 to 27.39. This means that effect-tuning hyperparameters can be used to improve the model's prediction accuracy and lower the error rate.

Keywords

Stock Price Prediction, LSTM, XGBoost, Random Forest Regressor, Machin Learning

1. Introduction

The stock market is a publicly accessible platform where investors and companies trade firm stocks. Online stock trading has been gaining popularity over the years. It has a significant role in stimulating economic growth and increasing economic efficiency. Stock trading is a crucial operation in the financial industry. It facilitates the purchasing and selling of stocks representing ownership in a company. If stock prices increase, investors will own a fraction of it. Conversely, if they decline, they suffer financial losses alongside the company [1]. Hence, predicting the future value of a stock or financial asset, also known as forecasting, is significant.

In stock market trading, stockbrokers utilize forecasting, mathematics, and learning technologies [2]. They use time series and technical analysis to predict stock market prices. They employ programming languages such as Python to forecast the outcome. However, [3] and [4] Investors often find difficulties in developing financial predictions due to stagnant and noisy data. Over 90% of personal investors are expected to leave the stock market within a year due to trading-related challenges such as stock price predictions. Therefore, companies and individual investors need assistance in accurately predicting stock market prices to maximize their gains. Predicting stock market patterns is essential for making sound investment decisions and generating profitable returns. Therefore, identifying the factors influencing the performance of machine learning (ML) models is crucial. In the literature, various review papers have focused on different machine learning, statistical, and deep learning techniques used for predicting stock market trends. However, there has been a lack of research into feature selection and extraction strategies specifically for stock market forecasting [5].

In predicting the stock market, machine learning and deep learning have an essential role in analyzing many datasets, identifying patterns, and determining whether the stock trend is up or down based on firstly, training the model on a parameter that is selected, next, evaluating the model to measure its performance lastly. Tune suitable hyperparameters to aid the model in achieving the best performance.

2. Literature Review

Historically, stock market forecasting depended on fundamental and technical analysis; however, recent years have ushered in a transformational period characterized by the use of advanced machine learning and deep learning methodologies [6] [7].

The paper examines Bitcoin price prediction through the lens of speculative stocks, emphasizing the efficacy of the LSTM-BTC model and its limitations for future data and applicability to other cryptocurrencies [7]. Comparative analyses using random forest regression and LSTM provide insights for enhancing Bitcoin price forecasting [8].

The capacity to accurately forecast financial trade information can be achieved through various methodologies utilizing specific acceptable data and diverse strat-

egies, including sentiment analysis, to indicate a significant correlation between individual emotions and their influence on sentiment for stocks. One of the increasingly significant aspects of the proposed method was to extract major events from online news to assess their influence on stock prices. Trade prediction safeguard: using historical data analysis. The particle swarm optimization approach is used to improve the support vector machine's parameters, enabling robust stock value predictions. This work enhances the support vector machine approach; nonetheless, the particle swarm optimization methodology necessitates an extended computation time. LSTM was combined with the naïve Bayesian method to gather market sentiment elements, improving prediction accuracy [9].

This paper does a comparative examination of several machine learning algorithms used for stock price prediction. The research used stock data from American Airlines for training objectives. This study uses the Python programming language. This research employs Machine Learning (ML) models, namely Decision Tree (DT), Support Vector Regression (SVR), Random Forest (RF), and Artificial Neural Network (ANN). The dataset was divided into a 70% training subset and a 30% testing subset. This dataset comprises stock information spanning five years. The simulation results indicate that the Random Forest model outperforms the other models. Consequently, real-time implementation is advised [10].

The authors in this study showed that the Stock Market is a highly researched area, and predicting its behavior is an essential requirement nowadays. However, predicting the Stock Market is a challenging task that requires a thorough study of data patterns. To achieve this, specific statistical models and artificially intelligent algorithms are needed. Various machine learning and deep learning algorithms can be used to make accurate predictions with minimal margin of error. The Artificial Neural Network (ANN) Deep Feedforward Neural Network, and the Convolutional Neural Network (CNN) are the two network models that have been extensively used to predict stock market prices. These models predict upcoming data values based on the last few days' data values. This recursive process continues until the dataset is valid. To optimize this prediction, deep learning has been used, and the results are significant. The ANN model achieved an accuracy of 97.66%, while the CNN model achieved an accuracy of 98.92%. The CNN model used 2-D histograms generated from the quantified dataset within a specific time frame to make predictions. To test the model, it was applied to the recent COVID-19 pandemic, which caused a sudden downfall of the stock market [11].

This research sought to determine the most efficacious stock price prediction model for each industry. The research used data from Infosys, ICICI, and SUN PHARMA spanning from January 2004 to December 2019. The research used Holt-Winters Exponential Smoothing, ARIMA, Random Forest, MARS, basic RNN, and LSTM models. The MARS model outperformed other machine learning models; however, the LSTM model had the greatest performance among deep learning models. Nonetheless, the MARS model has shown enhanced efficacy in sales forecasting across all three sectors: IT (utilizing data from Infosys), banking

(employing data from ICICI), and healthcare (leveraging data from SUN PHARMA) [12].

Forecasting stock prices is a prevalent subject in finance. The intrinsic volatility of the stock market predicts stock prices as a challenging endeavor. This issue pertains to a time series, and forecasting stock prices in the financial market is difficult, owing to the absence of defined protocols. A variety of approaches exist for forecasting stock values, including the Logistic Regression Model, Support Vector Machine (SVM), ARCH model, Recurrent Neural Network (RNN), Convolutional Neural Network (CNN), Backpropagation, Naïve Bayes, and ARIMA model, among others. Of these approaches, the Long Short-Term Memory (LSTM) algorithm is considered the most suitable for time series issues. The primary objective is to forecast prevailing market trends and precisely estimate stock valuations. Employing Long Short-Term Memory (LSTM) recurrent neural networks to predict market prices. Our results demonstrate that the forecast accuracy rate surpasses 93% [13].

The author of this research developed a machine learning technique using LSTM recurrent neural networks to forecast stock prices based on market trends. Stocks are classified into two categories, one of which is intraday trading, often referred to as day trading. Intraday trading entails liquidating all holdings before market closure to avoid ownership transfer at the day's conclusion. LSTM models are crucial in sequence prediction tasks because of their ability to efficiently store and utilize previous information. In stock prediction, it is essential to retain and evaluate previous stock data to precisely forecast future stock values [14].

This study [15] Used a multi-series stock prediction approach to predict stock prices in the future implementing the LSTM Model. They found that the accuracy of the weekly data exceeded the daily data.

This study examines the efficacy of a machine learning algorithm in forecasting future prices and identifying the ideal number of epochs necessary to enhance model accuracy. The research employs a recurrent neural network (RNN) with long short-term memory (LSTM) to forecast future values of GOOGL and NKE assets. The model's findings are encouraging, since it effectively tracks the progression of the starting values for both assets. In subsequent projects, the researchers will ascertain the optimal configurations for data length and the number of training epochs that align with available resources to enhance prediction accuracy [16].

This research employs both LSTM and ARIMA models to forecast stock prices. Various statistical metrics, including mean squared error (MSE), mean absolute error (MAE), and root mean squared error (RMSE), have been used to evaluate and compare predicting accuracy. The models were developed from historical closing price data from Yahoo Finance, Inc. The prediction results demonstrate that the LSTM methodology exhibits reduced values for MSE, MAE, and RMSE in comparison to the ARIMA method. The results of this research may aid market participants in forecasting future prices [17].

This study used ensemble learning to forecast stock returns and mitigate risk in

stock market investments. The model employs many technical indicators, such as the Relative Strength Index (RSI), stochastic oscillator, and more pertinent metrics. It comprises many decision trees and has shown enhanced efficacy relative to other algorithms outlined in the current literature. The research also assessed the validity of out-of-bag (OOB) error estimations [18].

The previous study focused on predicting stock prices using machine learning and deep learning models. However, no study tackled a comparison between machine learning and deep learning to enhance the accuracy of stock prediction and improve the model performance. This study aims to compare machine learning and deep learning to enhance model performance by adding and aiding some hyperparameters. This increases the model's efficiency by following some techniques, such as training the model on a large amount of data and splitting the data into training, evaluation, and test sets. The evaluation set process enhances the model's efficacy by monitoring the model during the training and then tuning the suitable hyperparameters to enhance the accuracy of stock prediction.

2.1. Long Short-Term Memory (LSTM)

LSTMs are an updated type of RNN (networks designed for processing sequential data by maintaining a hidden state that receives information about previous elements in the sequence) that aim to circumvent the limitations of older types of RNN by adding a more complex unit called an LSTM cell. LSTM cells consist of four gates (input, output, forget, and cell state) that control the flow of essential information, making the network retain information for long periods [19].

The Forget Gate determines whether to preserve or forget the cell's previous information. The Input Gate determines whether to add or ignore new information based on whether the new information adds new value. The Cell State is a memory that saves relevant information through time steps. The Output Gate determines whether to output or hide the information [20].

The implementation of LSTM using sequence data follows a set of sequential rules. The implementation of LSTM using sequence data follows a set of sequential procedures. LSTM Network Initialization is a process that puts in the values of the weights and biases for them. Furthermore, set the value of the cell state ($C0$) and hidden state ($h0$) to zero. The weights determine which features are influential in the dataset. The biases help improve the flexibility of machine learning training. Both weights and biases enhance the accuracy of predictions of the prices of a given future day based on the Sequence Data from the three previous sequential days. Sequence Data is a sequence of elements (they could be events or states) in which the order is significant, and no fixed time interval exists between the elements [20] [21].

2.1.1. Input Sequence

Let's examine an input sequence of length T, where we represent each element as a vector. Each vector is represented in sequence as {x1, x2, ..., xT}.

2.2.2. Processing Each Time Step

At each time step t in the sequence, the LSTM cell executes the following operations. Table 1(A) presents LSTM operational gates, while Table 1(B) presents the LSTM cell executing the following operations. Table 2 shows how LSTM can deal with three sequential time phases

Table 1. (A) LSTM operational gates; (B) Operational steps to execute the LSTM Cell.

(A)	
$f_t = \sigma\left(W_f \cdot \left[h_{t-1}, x_1\right] + b_f\right)$	Forget Gate (1)
$i_t = \sigma\left(W_i \cdot \left[h_{t-1}, x_1\right] + b_i\right)$ $\tilde{C}_t = \tanh\left(W_C \cdot \left[h_{t-1}, x_1\right] + b_C\right)$	Input Gate (2)
$C_t = f_t \odot C_{t-1}, i_1 \odot \tilde{C}_t$	Cell State Update (4)
$o_t = \sigma\left(W_o \cdot \left[h_{t-1}, x_1\right] + b_o\right)$ $h_t = o_t \tanh \odot \left(C_t\right)$	Output Gate (2)
(B)	
Steps	Explanation
Step 1	Suppose Time step 1, which (t) = 1: the first step = set x1 ad input and define (hidden and cell states).
Step 2	The Forget Gate determines whether to preserve or forget the previous information of the cell.
Step 3	The Input Gate determines whether to add or ignore new information based on whether the new information adds new value or not.
Step 4	Choose the candidate cell state based on the previous step.
Step 5	Modify the cell state; Compute the output Gate.
Step 6	Compute the hidden state.

Table 2. Three sequential time phases.

Time step 1: t = 1	Time step 2: t = 2	Time step 3: t = 3
$f_t = \sigma\left(W_f \cdot \left[h_{t-1}, x_1\right] + b_f\right)$	$f_2 = \sigma\left(W_f \cdot \left[h_1, x_2\right] + b_f\right)$	$f_3 = \sigma\left(W_f \cdot \left[h_2, x_3\right] + b_f\right)$
$i_t = \sigma\left(W_i \cdot \left[h_{t-1}, x_1\right] + b_i\right)$	$i_2 = \sigma\left(W_i \cdot \left[h_1, x_2\right] + b_i\right)$	$i_3 = \sigma\left(W_i \cdot \left[h_2, x_3\right] + b_i\right)$
$\tilde{C}_t = \tanh\left(W_C \cdot \left[h_{t-1}, x_1\right] + b_C\right)$	$\tilde{C}_2 = \tanh\left(W_C \cdot \left[h_1, x_2\right] + b_C\right)$	$\tilde{C}_3 \tanh\left(W_C \cdot \left[h_2, x_3\right] + b_C\right)$
$C_t = f_t \odot C_{t-1}, i_1 \odot \tilde{C}_t$	$C_2 = f_t \odot C_1, i_2 \odot \tilde{C}_2$	$C_3 = f_t \odot C_2, i_3 \odot \tilde{C}_3$
$o_t = \sigma\left(W_o \cdot \left[h_{t-1}, x_1\right] + b_o\right)$	$o_2 = \sigma\left(W_o \cdot \left[h_1, x_2\right] + b_o\right)$	$o_3 = \sigma\left(W_o \cdot \left[h_2, x_3\right] + b_o\right)$
$h_t = o_t \tanh \odot \left(C_t\right)$	$h_2 = o_2 \tanh \odot \left(C_2\right)$	$h_3 = o_3 \tanh \odot \left(C_3\right)$

2.2. XGBoost (Extreme Gradient Boosting) Model

The XGBoost algorithm employed in machine learning for regression and classification consists of a series of weak learners (usually decision trees) constructed sequentially. Each weak learner's purpose is to correct mistakes made by the preceding members. These weak learners (decision trees) are models that aim to eliminate the residuals (the differences between actual and predicted values). This whole process is called the loss function [23].

XGBoost uses regularization techniques such as L1 (Lasso) and L2 (Ridge) regularization to minimize overfitting and improve the generalization powers of the model. XGBoost uses parallel processing, which makes it computationally efficient and perfect for big datasets [24].

The optimization of an objective function in XGBoost combines a loss function with a regularization technique. The loss function measures the model's residuals (difference between predicted and actual values), while the regularization technique eliminates overly complex models to minimize overfitting [25].

An objective function in regression machine learning represents the goal the algorithm aims to minimize during its training process. The optimization of an objective function in XGBoost is the aggregate of a loss function and a regularization technique. The loss function measures the model's residuals (the difference between predicted and actual values), while the regularization technique eliminates overly complex models to minimize overfitting [26].

The objective function decreases by adjusting model parameters using gradient descent optimization (an iterative optimization algorithm used in machine learning to minimize the loss function by tuning the parameters gradually to reduce most of the loss). During the optimization, the decision trees aim to eliminate the residuals (the differences between actual and predicted values). This process continues until the algorithm reaches a minimum point where further tuning does not significantly reduce the loss.

$$\mathcal{L}(\theta) = \sum_{i=1}^{n} \iota(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{1}$$

Where $\mathcal{L}(\theta)$ the objective function (overall), $\iota(y_i, \hat{y}_i)$ –(evaluation metrics), and loss function were measured by (mean absolute error), $\Omega(f_k)$ Regularization term to minimize overfitting.

$$\hat{y}_i^{(t)} = \sum_{k=1}^{t} f_k(x_i) \tag{2}$$

Prediction: At each stage, the forecast $\hat{y}_i$, is calculated by summing the predictions made by all (t) the trees, and f_k : k-the tree. The objective function decreases by adjusting model parameters using gradient descent optimization (an iterative optimization algorithm used in machine learning to minimize the loss function by tuning the parameters gradually to reduce most of the loss) [27]. During the optimization, the decision trees aim to eliminate the residuals (the differences between actual and predicted values). This process continues until the algorithm reaches a minimum point where further tuning does not significantly reduce the loss.

2.3. Random Forest Model Regressor

Random Forest is a method of ensemble learning that involves averaging many decision trees. Within the forest, every decision tree is independently trained by utilizing a randomly chosen subset of the training data and its corresponding attributes [26].

The ultimate forecast is determined by taking the average of the predictions

from all trees in the case of regression or by utilizing a majority vote in the case of classification [28]. The mathematical formulation for predicting the output of a regression random forest model is as follows:

$$\hat{y} = \frac{1}{N}\sum_{i=0}^{n} f_i(x) \tag{3}$$

The anticipated output, represented as $\hat{y}$, represents the closing price. N refers to the number of trees in the forest, whereas $f_i(X)$ represents the prediction made by the *i*- the decision tree for the input attributes *X*. The fundamental notion in this context is that each decision tree plays a crucial part in the final prediction. Random Forest models, by aggregating the predictions of multiple trees, generally produce more robust and accurate predictions compared to a single decision tree. The Random Forest algorithm is a powerful machine learning technique often used for regression problems, such as predicting stock prices [29]. By adjusting parameters and hyperparameters, it is feasible to enhance the model's performance and make predictions by leveraging the combined knowledge of several decision trees [30] [31].

3. The Proposed Framework

Figure 1 outlines a comparative approach for predicting stock prices using DL and ML methods. The steps include gathering historical data from Yahoo Finance

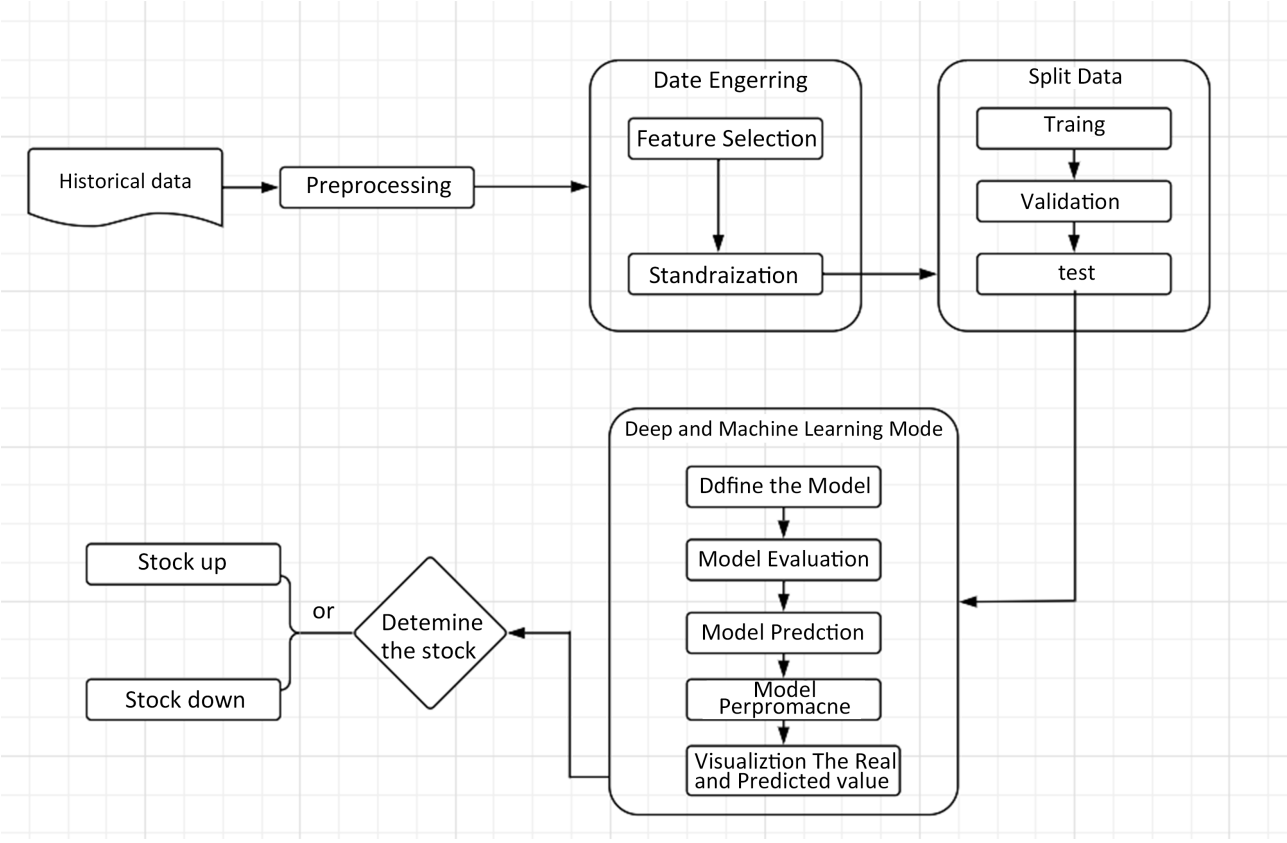


Figure 1. Framework of predicting Stock Prices using machine learning models.

for Apple, removing duplicates and handling missing values, and normalizing data between zero and one. Identifying key features for model training is crucial for effective prediction. Some techniques are often essential for ML and DL algorithms to perform efficiently, such as technical indicators, fundamental indicators, and normalizing.

Split the data set into training, validation, and test sets. In the training set, we train the model. In validation, the model performance is evaluated to train the data, and in the test set, the model is examined to predict unseen data.

Some steps applied to the model are selecting and defining the ML model (XGBoost, RF) regressor and DL (LSTM) to be used on these.

- Model Evaluation—Aids the model in training and understanding data performance and tuning.
- Model predictions are the model's ability to forecast the stock price.
- Model performance. We measure the model's performance to predict values by evaluating matrices such as MSE, R-square, or RMSE.
- Visualizing: Visualizing the difference between actual value and predicted value.
- Depending on the model forecast, the vendor will decide whether the stock price is up or down.
- Stock Up: Determine if the stock or market price will increase.
- Stock Down: Determine if the stock or market price will decrease.

Dataset Description

Dataset of Apple stock price (financial stock) from January 2015 and January 2020 from Yahoo Finance This dataset contains (Open-Close-Adj_Close-Low-High-Volume) prices that are shown in Table 3.

4. The Experimental Approach

This section discusses the experiments conducted to predict the future price of Apple. Detailed steps are described below through three main subsections. The first subsection illustrates the data preprocessing. The second subsection applies three different models to predict the stock price in the future.

Table 3. Parameter used in three models.

Parameter	Meaning
Date	Date of stock price
Open	The opening price of the stock at the start of the day.
High/low prices	The higher and lower price of stock on a day
Close price	The closing price of the stock at the end of the day.
Volume	How many exchanges and investments are made at all marketplaces in a certain time?
Adjusted close price	Its true value after dividend distribution and commonly considered the real price of the stock

4.1. Data Preprocessing

- Missing Values: Handle missing values by imputation about mean.
- Drop the duplicated row.
- Handling Date and Time: Convert date columns to Datetime and set it to index.
- Get the correlation between the column and the other to choose the column that has a strong correlation

4.2. Experimental for Stock Price Prediction

Once the data cleaning is done, this section uses three different models to predict future stock prices. Hence, examine the default hyperparameter compared to hyperparameters tuned to determine its effect on results and to what extent it improves model prediction and reduces error.

4.2.1. RF Regressor to Predict the Stock Price in the Future

To check how well the RF model works, data is gathered and preprocessed by getting rid of duplicates, filling in missing values, changing date columns to Datetime type, and picking out important features based on how well they match up (open, close, low, high, and volume prices). The chosen features undergo normalization to accelerate training. then divide the data set into a training set (70%), a validation set (15%), and a testing set (15%). train the RF model on the training data and tune its hyperparameters for optimal accuracy (100 DTs, random state 42). to determine the model's performance by calculating the root mean square error (RMSE) between the actual closing price and the predicted future price. Table 4 shows these predictions visually to help you understand the hyperparameter before and after tuning.

Table 4. Demonstrates hyperparameters before and after tuning using Random forest regression.

Hyperparameter	Before Tuning	After Tuning	Impact	Function
	Baseline Model	using GridSearchCV	Examine various combinations of hyperparameters to get the best solution and enhance the predictive model based on cross-validation.	
n_estimators	100	200	improve the model fit by reducing the error rate, including MSE and RMSE	the forest's total number of trees
max_depth	-	10	overcome complexity in the model and overfitting, which are controlled	Maximum depth of the trees.
min_samples_split	2	2	It's crucial to comprehend the impact of adding a new node to the tree and how many are required.	A minimum total number of samples is required to divide an internal node.
min_samples_leaf	1	4	Limiting the number of samples at the leaf node helps control the complexity of the model.	

4.2.2. XGBoost Regressor Is Predicting Stock Prices in the Future

To assess the efficiency of the XGBoost regressor model, this study follows the same steps described in the RF Model with a difference in tuning the hyperparam-

eters using techniques such as Grid Search, Random Search, Maximum depth of the trees (Max_Depth), Estimators (Number of trees in the ensemble), Sub-Sample (Fraction of samples used for training each tree), and lambda (L2 Regularization term). Table 5 shows these predictions visually to help you understand the hyperparameter before and after tuning.

Table 5. Demonstrates hyperparameters before and after tuning using the XGBoost Model.

Hyperparameter	Before Tuning	After Tuning	Importance	Impact on Model
learning_rate	0.1	0.1	Each iteration affects the step size.	To obtain results, more conservative updates use smaller values.
max_depth	3	5	Determine the tree's depth.	Deeper trees may be susceptible to overfitting when generating complex patterns.
min_child_weight	1	3	Determine the weight required for the node to split	Prevent overfitting through big values.
subsample	1.0	0.7	Training trees use a fraction of the samples.	More minor sample data reduces overfitting.
colsample_bytree	1.0	0.6	Training each tree using feature fractions	Sampling features Reduce overfitting
n_estimators	100	300	the total number of trees in the ensemble	Despite achieving higher performance by increasing the number of trees in the contract, the training time has increased.
gamma	0	0.5	Regularization parameter for tree complexity.	Regularization of tree structure controls

4.2.3. Using the STM in Predicting the Stock Price in the Future

Data preprocessing follows the same steps as the previous models, except for feature selection, which is set to the close price only. Next, we divide the dataset into three sets: 86% for model training, 9% for model validation, and 5% for model testing. In model training, create a function that feeds the model's input sequences of 60 days in a row of data to predict the next day's step for a close price. This process iterates for the whole training dataset in the future (update). Create a function to take input sequences for LSTM. Each input sequence considers the output for the next step. The close price (target) was set as the preceding step to predict the next step to the Close price. The regression method was applied to the close price of Apple stock. Once all the sequence data has been computed, it is put into a table with input features (X) and output targets (Y). The model training process will use this table to tune some hyperparameters, like the number of units being 50, the number of epochs being 100, and the batch size being 32. Split the data into training, validation, and testing sets. Use all data training except the last 60 records used in the test. Define the architecture of the LSTM model. Use two LSTM layers: The first LSTM layer obtains essential time series features, including short-term price fluctuations. In contrast, the second LSTM layer gains more complex sequential dependencies and movements over a longer duration than followed by the Dense layer. Compile the model with the Adam optimizer and mean squared error loss function. Visualize the actual and predicted

values. GAssess performance using evaluation metrics that include RMSE and mean absolute error (MSE). The RMSE is the root of MSE; its formula is as follows:

$$\text{Square Error} = \left(y_i + \hat{y}_i\right)^2 \tag{4}$$

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right| \tag{5}$$

$$\text{RMSE} = \frac{1}{\sqrt{n}}\sum_{i=1}^{n}\left(y_i + \hat{y}_i\right)^2 \tag{6}$$

$$\text{R}^2 = \sum_{n=1}^{\infty}\left(\frac{\text{RSS}}{\text{TSS}} = \frac{\sum\left(y_i + \hat{y}_i\right)^2}{\sum\left(y_i + \bar{y}_i\right)^2}\right) \tag{7}$$

In the sequence data (time step), we rely on the closing price as both a training value and a prediction value. This implies that we consider each predicted output, Y(t), as a new input, X (t-3), to indicate the next value based on the data points. Assume that the sequence data is 3, the time step is 3, and Table 6 presents predictions in sequence data utilizing previous Close prices in the future.

Table 6. Presents predictions in sequence data utilizing previous close prices in the future.

Close Price feature										
Days	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Close	**2**	**6**	**5**	**7**	**6**	**10**	**11**	**12**	**10**	**15**
Sequence Len = 3										
Days	**Previous (3) time step feature**			**Predict future (1) time steps output**						
	X(t-3)	**X (t-2)**	**X(t-1)**	**Y(t)**						
Day 4	**2**	**6**	**5**	**7**						
Day 5	**7**	**6**	**10**	**11**						
Day 6	**11**	**12**	**10**	**15**						

For example,

- On day four, the predicted output (7) is the first new input with two previous values
- (day 3 and 2) to predict the next value for day 5 (11).
- The predicted output (11) is the first new input with two previous values on day five.
- Days 3 and 4 are used to predict the next value for Day 6 (15), and so on. Then, follow the same steps with all training data. Once you have the training data, proceed to train, validate, and test it. It plays an important role in tuning hyperparameters to enhance the performance of machine learning models like the LSTM model.
- Finding the best values for hyperparameters like optimizer, number of units, number of layers, batch size, epochs, and dropout rate can help predict how well the model will work and how well it will work in general. It can also help train the model faster. The table shows these predictions visually to help you understand the hyperparameter before and after tuning, as shown in Table 7.

Table 7. Demonstrates hyperparameters before and after tuning Using the LSTM Model.

Hyperparameter	Without Tuning	Tuning (Keras Tuner)	Function	Importance of the tuning
Number of LSTM Units	50	Varies (50 - 200)	The LSTM layer determines the capacity to learn the patterns.	It helps in overcoming underfitting or overfitting issues.
Dropout Rate	0.3	Varies (0.2 - 0.5)	used to prevent overfitting randomly during training	Get a suitable balance between the complexity of the model and generalization ability.
Optimizer	Adam	(Adam, RMSprop)	weights for the model must be improved	Finding the best optimizer has significant improvement in the model performance
Epochs	20	20	(number of training epochs) its iterations (20) time on all training data. Determine how often the whole training data set comes across the network. Note that the model can achieve the best training while using more epochs. However, sometimes, it leads to overfitting, so choosing the best number of epochs is important.	
Batch Size	32	32	The size of each batch of input data is crucial. It's important for training efficiency to determine the number of input samples propagated at every network.	
Activation Functions	Relu	Relu	The model incorporates functions that introduce non-linearity. Allow the model to recognize and learn complex patterns. Tanh and sigmoid are widely used internally in the LSTM layer, while ReLU is commonly used in the dense layer because it is the most suitable for regression tasks.	
Loss Function	MSE, MAE, RMSE	MSE, MAE, RMSE	A function is used to measure the difference between the predicted and actual values.	

5. The Experimental Results

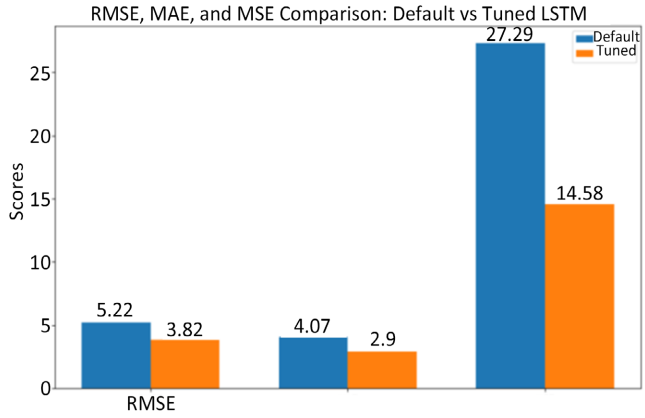
This section demonstrates the results obtained from three experimental models. To predict the stock price in the future and determine the effect of tuning hyperparameters before comparing to the result after tuning parameters to improve and enhance the model performance table, the finding result was conducted as demonstrated in Table 8.

Table 8. Shows the result after tuning the hyperparameter compared to the result after turning.

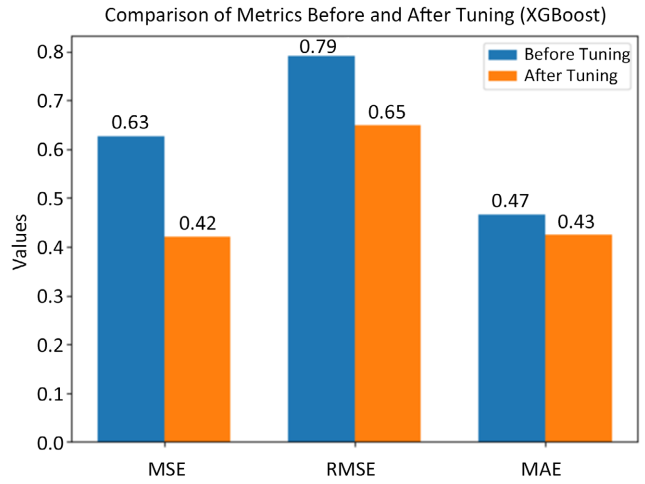
model	Before tuning hyperparameter			After tuning hyperparameter		
Evaluation performance	RMSE	MAE	MSE	RMSE	MAE	MSE
LSTM	5.22	4.04	27.29	3.82	2.9	14.58
XGBoost	0.79	0.47	0.63	0.65	0.43	0.42
RF	28.12	23.53	790.8	27.39	23.01	750.37

This section visualizes the performance using evaluation metrics such as RMSE, MSE, and MAE before tuning and hyperparameters compared to after tuning in Figure 2(A). demonstrates the results of the LSTM model before and after turning on the hyperparameter, while Figure 2(B). demonstrates the results of the XGBoost model before and after turning on the hyperparameter, and Figure 2(C). shows the outcome of the Random Forest model both before and after the hy-

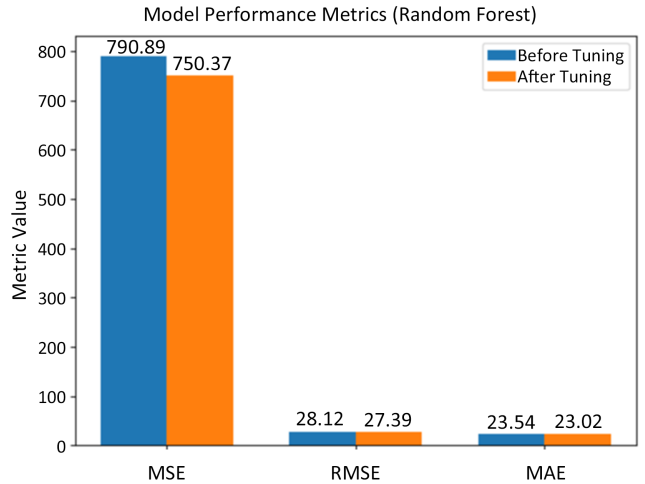
perparameter was changed.



(A)



(B)



(C)

Figure 2. (A) Demonstrates the results of the LSTM model before and after turning on the hyperparameter; (B) Demonstrates the results of the XGBoost model before and after; turning the hyperparameter. (C) Shows the outcome of the Random Forest model both before and after the hyperparameter was changed.

6. Study Limitations

There are several limitations seen in the study, even with the promising findings it has presented. All the models were trained and tested with only Apple's stock price data, which could perform poorly when implemented on other stocks or financial markets with different characteristics. The models mainly used historical stock price data and failed to consider other factors such as market sentiment, indices, and news that affect stock prices. Also, the hyperparameters were adjusted based on the present knowledge, meaning there could be a better set of hyperparameters if a more complex search or optimization algorithm was used. Besides, the short time horizons of forecasts restrict the usage of the models in the context of long-term investment planning. Future work should overcome these limitations by employing more extensive data, including external variables, tuning hyperparameters more comprehensively, and considering realistic trading constraints.

7. Conclusion

The study showed that our three models—LSTM, RF, and XGBoost—worked better than others by using sequence data and fine-tuning the hyperparameters to get better predictions than other studies. The regression models exhibited higher accuracy, lower prediction errors, and greater consistency in predicting stock prices across various periods and market conditions. These regression models used historical data to predict the stock price. Among these models, the LSTM model best predicted the stock price in the future, reducing the RMSE value from 5.22 to 3.82. Similarly, XGBoost reduced prediction errors from 0.79 to 0.65 in RMSE, and by recording the last rand, the RF model reduced an RMSE value from 28.12 to 27.39 in predicting the stock price for Apple. Although his study achieved excellent performance for all three models by tuning their hyperparameters and analyzing their features, increasing the number of forecasted days will positively affect the results and performance.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Julian, T., Devrison, T., Anora, V. and Suryaningrum, K.M. (2023) Stock Price Prediction Model Using Deep Learning Optimization Based on Technical Analysis Indicators. *Procedia Computer Science*, **227**, 939-947. https://doi.org/10.1016/j.procs.2023.10.601

[2] Yu, P. and Yan, X. (2019) Stock Price Prediction Based on Deep Neural Networks. *Neural Computing and Applications*, **32**, 1609-1628. https://doi.org/10.1007/s00521-019-04212-x

[3] Wang, Z., Ho, S. and Lin, Z. (2018). Stock Market Prediction Analysis by Incorporating Social and News Opinion and Sentiment. 2018 *IEEE International Conference on Data Mining Workshops* (*ICDMW*), Singapore, 17-20 November 2018, 1375-1380.

https://doi.org/10.1109/icdmw.2018.00195

[4] Duong, H. and Nguyen-Thi, T. (2021) A Review: Preprocessing Techniques and Data Augmentation for Sentiment Analysis. *Computational Social Networks*, **8**, Article No. 1. https://doi.org/10.1186/s40649-020-00080-x

[5] Hamed, S., Ezzat, M. and Hefny, H. (2020) A Review of Sentiment Analysis Techniques. *International Journal of Computer Applications*, **176**, 20-24. https://doi.org/10.5120/ijca2020920480

[6] Sonkavde, G., *et al.* (2023) Forecasting Stock Market Prices Using Machine Learning and Deep Learning Models: A Systematic Review, Performance Analysis and Discussion of Implications. *International Journal of Financial Studies*, **11**, Article 94. https://doi.org/10.3390/ijfs11030094

[7] Ateeq, K., Abdelrahim, A., Zarooni, A., Rehman, A. and Khan, A. (2023) A Mechanism for Bitcoin Price Forecasting Using Deep Learning. *International Journal of Advanced Computer Science and Applications*, **14**, 441-448. https://www.ijacsa.thesai.org

[8] Chen, J. (2023) Analysis of Bitcoin Price Prediction Using Machine Learning. *Journal of Risk and Financial Management*, **16**, Article 51. https://doi.org/10.3390/jrfm16010051

[9] Li, J., Bu, H. and Wu, J. (2017) Sentiment-Aware Stock Market Prediction: A Deep Learning Method. 2017 *International Conference on Service Systems and Service Management*, Dalian, 16-18 June 2017, 1-6.

[10] Vijh, M., Chandola, D., Tikkiwal, V.A. and Kumar, A. (2020) Stock Closing Price Prediction Using Machine Learning Techniques. *Procedia Computer Science*, **167**, 599-606. https://doi.org/10.1016/j.procs.2020.03.326

[11] Mukherjee, S., Sadhukhan, B., Sarkar, N., Roy, D. and De, S. (2021) Stock Market Prediction Using Deep Learning Algorithms. *CAAI Transactions on Intelligence Technology*, **8**, 82-94. https://doi.org/10.1049/cit2.12059

[12] Chatterjee, A., Bhowmick, H. and Sen, J. (2021) Stock Price Prediction Using Time Series, Econometric, Machine Learning, and Deep Learning Models. 2021 *IEEE Mysore Sub Section International Conference* (*MysuruCon*), Hassan, 24-25 October 2021, 289-296. https://doi.org/10.1109/mysurucon52639.2021.9641610

[13] Chen, S. (2022) Stock Price Prediction Using LSTM Model. *CAIBDA* 2022—2*nd International Conference on Artificial Intelligence*, Big Data and Algorithms, Nanjing, 17-19 June 2022, 217-220.

[14] Zhang, R. (2022) LSTM-Based Stock Prediction Modeling and Analysis. *Advances in Economics, Business and Management Research*, **211**, 2537-2542. https://doi.org/10.2991/aebmr.k.220307.414

[15] Shao, G. (2023) Prediction of Stock Prices Based on the LSTM Model. Atlantis Press International BV. https://doi.org/10.2991/978-94-6463-142-5_42

[16] Zhang, Y. (2022) Stock Price Prediction Method Based on Xgboost Algorithm. Atlantis Press International BV. https://doi.org/10.2991/978-94-6463-030-5_60

[17] Qian, H. (2022) Stock Predicting Based on LSTM and Arima. *Proceedings of the* 2022 2*nd International Conference on Economic Development and Business Culture* (*ICEDBC* 2022), Dali, 24-26 June 2022, 485-490. https://doi.org/10.2991/978-94-6463-036-7_72

[18] Khaidem, L., Saha, S. and Dey, S.R. (2016) Predicting the Direction of Stock Market Prices Using Random Forest. arXiv: 1605.00003.

[19] Nirob, F.A. and Hasan, M.M. (2023) Predicting Stock Price from Historical Data Using LSTM Technique. *Journal of Artificial Intelligence and Data Science* (*JAIDA*), **3**,

36-49.

[20] Wu, S., Liu, Y., Zou, Z. and Weng, T. (2021) S_I_LSTM: Stock Price Prediction Based on Multiple Data Sources and Sentiment Analysis. *Connection Science*, **34**, 44-62. https://doi.org/10.1080/09540091.2021.1940101

[21] Subasi, A., Amir, F., Bagedo, K., Shams, A. and Sarirete, A. (2021) Stock Market Prediction Using Machine Learning. *Procedia Computer Science*, **194**, 173-179. https://doi.org/10.1016/j.procs.2021.10.071

[22] Moghar, A. and Hamiche, M. (2020) Stock Market Prediction Using LSTM Recurrent Neural Network. *Procedia Computer Science*, **170**, 1168-1173. https://doi.org/10.1016/j.procs.2020.03.049

[23] Shahani, N.M., Zheng, X., Liu, C., Hassan, F.U. and Li, P. (2021) Developing an Xgboost Regression Model for Predicting Young's Modulus of Intact Sedimentary Rocks for the Stability of Surface and Subsurface Structures. *Frontiers in Earth Science*, **9**, Article 761990. https://doi.org/10.3389/feart.2021.761990

[24] Chen, T. (2019) Release 0.81 XGBoost Developers.

[25] Bentéjac, C., Csörgő, A. and Martínez-Muñoz, G. (2019) A Comparative Analysis of XGBoost. arXiv: 1911.01914.

[26] Wu, Y. (2023) Stock Price Prediction Based on Simple Decision Tree Random Forest and XGBoost.

[27] Yang, Y., Wu, Y., Wang, P. and Jiali, X. (2021) Stock Price Prediction Based on XGBoost and LightGBM. *E3S Web of Conferences*, **275**, Article ID: 01040. https://doi.org/10.1051/e3sconf/202127501040

[28] Kirasich, K., Smith, T. and Sadler, B. (2018) Random Forest vs Logistic Regression: Binary Classification for Heterogeneous Datasets. *SMU Data Science Review*, **1**, Article 9.

[29] Illa, P.K., Parvathala, B. and Sharma, A.K. (2022) Stock Price Prediction Methodology Using Random Forest Algorithm and Support Vector Machine. *Materials Today: Proceedings*, **56**, 1776-1782. https://doi.org/10.1016/j.matpr.2021.10.460

[30] Kurdi, F.T. (2021) Random Forest Machine Learning Technique for Automatic Vegetation Detection and Modelling in Lidar Data. *International Journal of Environmental Sciences & Natural Resources*, **28**, Article ID: 556234. https://doi.org/10.19080/ijesnr.2021.28.556234

[31] Alam, O. and Qiao, X. (2020) An In-Depth Review on Municipal Solid Waste Management, Treatment and Disposal in Bangladesh. *Sustainable Cities and Society*, **52**, Article ID: 101775. https://doi.org/10.1016/j.scs.2019.101775