



## Article

# A Comparative Analysis of Machine Learning and Deep Learning Techniques for Accurate Market Price Forecasting

Olamilekan Shobayo <sup>1,\*</sup> , Sidikat Adeyemi-Longe <sup>1</sup>, Olusogo Popoola <sup>1</sup> and Obinna Okoyeigbo <sup>2</sup> 

<sup>1</sup> School of Computing and Digital Technologies, Sheffield Hallam University, Sheffield S1 2NU, UK; sidikat.a.adeyemi-longe@student.shu.ac.uk (S.A.-L.)

<sup>2</sup> Department of Engineering, Edge Hill University, Ormskirk L39 4QP, UK

\* Correspondence: o.shobayo@shu.ac.uk

**Abstract:** This study compares three machine learning and deep learning models—Support Vector Regression (SVR), Recurrent Neural Networks (RNN), and Long Short-Term Memory (LSTM)—for predicting market prices using the NGX All-Share Index dataset. The models were evaluated using multiple error metrics, including Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Square Error (RMSE), Mean Percentage Error (MPE), and R-squared. RNN and LSTM were tested with both 30 and 60-day windows, with performance compared to SVR. LSTM delivered better R-squared values, with a 60-day LSTM achieving the best accuracy (R-squared = 0.993) when using a combination of endogenous market data and technical indicators. SVR showed reliable results in certain scenarios but struggled in fold 2 with a sudden spike that shows a high probability of not capturing the entire underlying NGX pattern in the dataset correctly, as witnessed by the high validation loss during the period. Additionally, RNN faced the vanishing gradient problem that limits its long-term performance. Despite challenges, LSTM's ability to handle temporal dependencies, especially with the inclusion of On-Balance Volume, led to significant improvements in prediction accuracy. The use of the Optuna optimisation framework further enhanced model training and hyperparameter tuning, contributing to the performance of the LSTM model.



Academic Editor: Domenico Ursino

Received: 12 September 2024

Revised: 8 January 2025

Accepted: 23 January 2025

Published: 11 February 2025

**Citation:** Shobayo, O.; Adeyemi-Longe, S.; Popoola, O.; Okoyeigbo, O. A Comparative Analysis of Machine Learning and Deep Learning Techniques for Accurate Market Price Forecasting. *Analytics* **2025**, *4*, 5. <https://doi.org/10.3390/analytics4010005>

**Copyright:** © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

**Keywords:** deep learning; hybrid model; LSTM; logistic regression; market price forecasting; Recurrent Neural Network; Support Vector Regression; FinBERT

## 1. Introduction

The prediction of stock market prices has been a long-standing challenge for financial analysts and investors. The intrinsic volatility and complexity of financial markets make it difficult to accurately forecast future trends. Traditional models such as linear regression and time-series models such as ARIMA are often limited in their ability to capture non-linear and sequential dependencies within the data [1]. This has led to the growing use of machine learning (ML) and deep learning (DL) techniques for stock market forecasting. The NGX All-Share Index, a key indicator of stock market performance in Nigeria, provides a robust dataset for evaluating the effectiveness of different forecasting models. In this study, we compare three models: Support Vector Regression (SVR), Recurrent Neural Networks (RNN), and Long Short-Term Memory (LSTM), to identify the most effective model for predicting market prices. Each model is uniquely suited to different types of data and prediction tasks, and understanding these differences is crucial for financial time-series forecasting. SVR is known for its robustness in small datasets and simple tasks, while RNN and LSTM excel in handling sequential data due to their inherent architecture. SVR

operates by mapping input features into two-dimensional spaces using kernel functions, allowing it to capture market relationships. However, it can struggle with time-series data where past information strongly influences future trends. RNN is a type of deep learning model that is designed to capture sequential data, but it suffers from the vanishing gradient problem, which limits its ability to retain long-term dependencies. LSTM, an extension of RNN, solves this issue by introducing memory cells that store relevant information over longer periods [2]. This makes it ideal for time-series forecasting.

According to Fama's Efficient Market Hypothesis (EMH), asset prices completely represent all available information at any given time in financial markets, which are said to be informationally efficient [3]. This paradigm suggests that price changes follow a "random walk", which implies that utilising historical data to predict future prices is virtually futile. The market's capacity to swiftly respond to new data negates any advantage obtained from analysing previous patterns since new information is absorbed into pricing fast and unpredictably. The EMH would posit for this study that using historical data to predict future stock prices with advanced algorithms such as Support Vector Regression (SVR), Recurrent Neural Networks (RNN), or Long Short-Term Memory (LSTM) would ultimately be unsuccessful. This theoretical stance poses significant challenges for time-series forecasting in financial markets. However, the aim of this study was not to challenge the validity of EMH but rather to examine how these artificial intelligence models perform in real-world scenarios, particularly in the context of recent financial theories such as the Adaptive Markets Hypothesis. The conventional theory of markets as perfectly efficient is challenged by Andrew Lo's Adaptive Markets Hypothesis (AMH), which presents a more dynamic viewpoint in which market efficiency fluctuates over time [4]. AMH claims that rather than being constantly totally efficient, markets adjust and evolve as a result of investors' reactions to the assimilation of new information. This could be perceived as a blend of principles from classical economics (recognition of market efficiency most of the time) and behavioural finance that recognise the irrational acts of humans. This framework recognises the possibility of using historical patterns, such as trends and anomalies, for short periods of time, especially when there are market inefficiencies. AMH suggests that market participant's behaviour can change depending on a variety of circumstances, including experience, the state of the market, and the availability of information. Therefore, machine learning approaches such as SVR, RNN, or LSTM that are included in price prediction models may be able to capture these inefficiencies at certain times. As a result, AMH offers a theoretical justification for this study methodology. This implies that, while financial markets are generally efficient, there are instances in which prediction models outperform others, most especially during market transitions or when specific market inefficiencies emerge. This is essential for understanding why artificial intelligence models succeed in forecasting prices under certain conditions, such as volatile markets or during periods of economic uncertainty as seen in this study. The behaviour of market participants deviates from rational expectations, which creates opportunities for artificial intelligence algorithms to capture patterns that would otherwise be hidden in more stable, efficient markets. The outperformance of LSTM, compared to SVR and RNN, in its ability to capture long-term dependencies and market complexities may be particularly useful during periods of market adaptation. This superior performance of LSTM suggests that the model is more adept at identifying and leveraging market inefficiencies, especially when compared to SVR or RNN, which may not handle temporal dependencies or complex patterns as effectively. The positioning of this study within the AMH framework shows the relevance and applicability of these models in real-world financial forecasting.

In this research, we used the NGX dataset, which contains historical price data along with technical indicators such as moving averages and On-Balance Volume. These indica-

tors are essential in understanding market behaviour and enhancing predictive accuracy. Additionally, the performance of these models was evaluated using a variety of error metrics, including Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Percentage Error (MPE), and R-squared. Among these, R-squared is a crucial metric as it measures how well a model explains the variance in the data, providing an overall indication of each model's performance. In this study, we employed Optuna, an optimisation framework, to fine-tune the hyperparameters of the models for better accuracy. The training process involved time-series cross-validation, which ensures that models are trained and tested on temporally consistent data. This respects the chronological order of the data and prevents data leakage between training and testing sets. Python and its libraries were used for implementing and training the models. This provided flexibility and efficiency in managing the computational requirements. The results reveal that LSTM outperforms SVR and RNN in both 30-day and 60-day time windows, particularly when exogenous factors such as On-Balance Volume are included. The 60-day LSTM model achieved the best R-squared value of 0.993, indicating its superior ability to predict future stock prices. Despite RNN's good performance, the vanishing gradient issue limited its effectiveness for longer time windows. SVR delivers good results for smaller datasets but showed sudden drops in performance in the second fold during cross-validation. This highlights SVR's limitation in handling larger and more complex datasets.

### 1.1. Research Contributions

The specific contributions of this research include:

- **Comparative Analysis of SVR, RNN, and LSTM:** This study provides a detailed comparison of three widely used models for time-series forecasting in financial markets, highlighting their strengths and limitations.
- **Demonstration of LSTM Superiority:** This research demonstrates that LSTM, with its ability to handle long-term dependencies, outperforms both SVR and RNN in longer time windows and also confirms its suitability for financial forecasting.
- **Inclusion of Exogenous Variables:** The incorporation of On-Balance Volume (OBV) as an exogenous variable in this study shows how the inclusion of technical indicators can significantly enhance predictive accuracy in stock price forecasting.
- **Application of Advanced Hyperparameter Tuning:** The use of Optuna for hyperparameter tuning illustrates the importance of optimisation in improving the model's performance. This framework enables a more systematic and efficient search for the best hyperparameters compared to manual tuning.
- **Evaluation with Time-Series Cross-Validation (TimeSeriesSplit):** This study emphasises the importance of time-series cross-validation in ensuring robust model evaluation, as it respects the temporal structure of financial data and prevents information leakage, leading to more realistic performance estimates.
- **Comprehensive Error Metrics:** A broad range of evaluation metrics, including MAE, MSE, RMSE, MPE, and R-squared, are used to provide a thorough assessment of each model's performance. The inclusion of R-squared highlights the models' ability to explain variance in the data, which is particularly relevant for financial prediction tasks.
- **Diagnostic Evaluation:** Further diagnostic evaluation tools such as scatter plot, Loss over Epoch plot, and residual plot are used to provide a comprehensive analysis of how artificial intelligence models behave and assist in diagnosing issues, fine-tuning the model, and ensuring generalisation.

## 1.2. Significance of the Study

The practical significance of this study is evident in its

- **Improved Financial Forecasting:** This study's identification of LSTM as the most effective model for the time-series forecasting of stock prices enables traders and investors to make more informed decisions about buying or selling stocks, and other financial assets.
- **Enhanced Algorithmic Trading Strategies:** The findings support the use of LSTM models in algorithmic trading, where predictive models are used to automate trades based on real-time data. This can lead to more profitable trading strategies, allowing algorithms to respond to market movements faster and more accurately.
- **Risk Management and Financial Planning:** This study contributes to better risk management as financial institutions can use the findings to develop more accurate models that anticipate market downturns or periods of high volatility.
- **Application of Technical Indicators:** The incorporation of technical indicators such as On-Balance Volume (OBV) in machine learning models provides traders and analysts with a competitive edge in identifying market opportunities.
- **Supporting Quantitative Finance and Machine Learning in Finance:** The findings of this study provide a valuable benchmark for model selection in financial forecasting and have practical significance for quantitative finance professionals who develop machine learning models to predict market trends.
- **Educational Value for Financial Data Science:** This research provides a clear and practical example of how machine learning models can be applied to real-world financial data and offers educational value for students, researchers, and professionals learning about financial data science and its practical applications. It also contributes to the growing body of knowledge on applying machine learning to financial markets, thereby providing a case study that can be expanded upon by future researchers or practitioners.

This could lead to better tools and methods for predictive analytics in financial institutions, improving the overall accuracy of forecasting systems used in asset management, risk assessment, and trading.

Risk management and quantitative finance. By demonstrating the effectiveness of LSTM models and optimising predictive models with technical indicators and hyperparameter tuning, this research equips financial professionals with tools to improve their forecasting accuracy, automate trading strategies, and manage risk more effectively. These insights are highly valuable to both academic researchers and industry practitioners, offering real-life benefits in the field of finance.

To provide a clearer understanding of the current research and position of this study within the academic field, a detailed Literature Review Table (Table 1) was created. This table presents the various data analysis techniques previously used in stock price prediction research, highlighting their pros and cons. Furthermore, the performance of the three models will be compared with these established methods, showing their ability to capture market trends effectively and improve prediction accuracy.

**Table 1.** Summary of previous work.

| Author                           | Algorithms Used                            | Study Procedure   | Outcome   | Study Limitation  |
|----------------------------------|--|---|---|---|
| Abuein, QQ. et al. [2]           | LSTM and Support Vector Regression (SVR)   | LSTM showed lower MSE, RMSE, and MAE values compared to SVR.                      | LSTM outperformed and captured complex trends while SVR struggled with capturing the underlying complex trend.                        | Limited to two comparative models (LSTM and SVR) without comparison to RNN.   |
| Zulfike, MS. et al. [5]          | LSTM, SVR, and Vector Autoregression (VAR) | LSTM had higher prediction accuracy and a better R-squared score than SVR.        | Our study includes technical indicators such as moving averages and volatility, as well as capturing non-linear relationships better. | Limited to the addition of a linear statistical model without technical indicators or comparison to RNN.                |
| Lakshminarayanan, JP. et al. [6] | SVM and LSTM                               | LSTM achieved better MAPE and R-squared scores than SVM.                          | LSTM outperformed due to the handling of temporal dependencies.   | Limited to SVM that classifies and cannot predict continuous values.  |
| Pashankar, SS. et al. [7]        | Linear regression, Random Forest, SVR      | SVR demonstrated good short-term predictions but struggled with long-term trends. | LSTM had better results in long-term prediction accuracy, especially with large datasets.   | Limited to short-term predictions with traditional models. Lack of focus on long-term dependencies and non-linear data. |
| Chhajer P, et al. [8]            | LSTM, SVM, and ANN                         | LSTM outperformed SVM and ANN for non-linear time-series data.                    | LSTM provided superior long-term performance compared to SVM and ANN when trained on multiple input features.                         | Models lacked diverse input features as variables.  |
| Shangshang J. [9]                | LSTM, ARIMA, SVR, GRU                      | LSTM significantly outperformed ARIMA, GRU, and SVR in long-term predictions.     | LSTM captured both long-term trends and non-linear relationships better than RNN.   | Limited focus on long-term trends, with ARIMA struggling in non-linear tasks.   |

Several studies have compared the effectiveness of SVR, RNN, and LSTM for time-series forecasting. The authors in [5] investigated the application of Long Short-Term Memory (LSTM) and Support Vector Regression (SVR) models in time-series forecasting. Their study focused on comparing the effectiveness of both models, using error metrics such as MSE, RMSE, and MAE. They found that the LSTM model consistently outperformed SVR with lower error rates, making it more suitable for capturing complex patterns in stock data [2]. The study was limited by its scope, comparing only two models without including RNN. However, the findings were significant in highlighting LSTM's strength in handling time-series forecasting when dealing with large datasets. Zulfike, MS et al., extended this comparison to include Vector Autoregression (VAR) in predicting stock prices but did not include RNN and technical indicators such as moving averages or volatility measures as model input features, which are key variables in our study. By comparing these models, they found that LSTM achieved better prediction accuracy and a higher R-squared score than SVR. However, the study primarily relied on statistical models such as VAR, which are based on linear assumptions, and limited the assessment of advanced AI models that could better capture non-linear relationships. Our study addresses this by including a wider range of variables as model input features and showing that LSTM performs better in capturing non-linear relationships in stock data than SVR and VAR. Lakshminarayanan, JP et al. [6] conducted a comparative study between Support Vector Machine (SVM) and LSTM models for stock price prediction. They demonstrated that LSTM consistently outperformed SVM in terms of the Mean Absolute Percentage Error (MAPE) and R-squared

scores and in handling continuous prediction tasks better than SVM. However, their work did not include technical indicators as variables, which our study does, further supporting LSTM's advantages in stock forecasting. Pashankar, SS. et al. [7] focused on evaluating the effectiveness of linear regression, Random Forest, and Support Vector Regression (SVR) for predicting stock prices. While their findings indicated that SVR performed well for short-term predictions, SVR struggled with long-term dependencies. Our research builds on this by demonstrating that LSTM achieves better long-term prediction accuracy, especially with larger datasets and more complex technical indicators. Chhajer P, et al. [8] explored LSTM, ARIMA, SVR, and GRU models. The study revealed that LSTM significantly outperformed ARIMA, GRU, and SVR in long-term predictions. However, the research was limited by its focus on long-term trends, with ARIMA struggling to manage non-linear tasks [9]. Our study proved the effectiveness of LSTM superiority by capturing both long-term trends and non-linear relationships, outperforming other models in these aspects.

There is evidence of a gap in the literature regarding the superiority of the approach, particularly when considering the variables used in this study. While previous studies have compared SVR and LSTM, there is still a gap when it comes to the inclusion of multiple technical indicators as model input features, as used in this study, especially when comparing these models on non-linear, complex financial data. LSTM's advantage over both SVR and RNN should be emphasised in handling large datasets with long-term dependencies. Many studies have highlighted RNN's ability to handle sequential data better than SVR, but RNN still falls short compared to LSTM. This is because RNN struggles with long-term dependencies, while LSTM, with its internal gating mechanisms, can manage long-term temporal dependencies far more effectively. Furthermore, no study has thoroughly examined the use of LSTM, RNN, and SVR combined with historical price data, return, price, and volume technical indicators as input features, which makes this study a novel contribution to the field.

## 2. Materials and Methods

This study employs historical price data from the NGX All-Share Index (NGX) to examine the performance and trajectory of the Nigerian stock market. The NGX is a comprehensive benchmark index that reflects the total market capitalization of all equities listed on the NGX.

$$\text{Market Capitalization} = \text{Current Share Price} \times \text{Total Number of Outstanding Share} \quad (1)$$

The NGX Index value is given by the formula:

$$(\text{Current Market Value}) / (\text{Base Market Value}) \times 100 = \frac{\sum_{i=1}^n PaQa}{\sum_{i=1}^n PbQb} \times 100 \quad (2)$$

The dataset spans from 4 January 2010 to 7 June 2024 and consists of 3573 observations. Each observation represents a daily record of stock prices and volume. Technical indicators, such as returns, moving averages, On-Balance Volume (OBV), etc, which are crucial for predicting market trends, were calculated from the initial observation. The NGX dataset pre-processing involved eliminating leading spaces, converting and sorting dates, excising commas, discarding NaN values, converting data to float, and normalising closing price values. The Min–Max scaling technique from Scikit-learn was employed to normalise the NGX dataset. This considers its specific attributes as capital market data. The mathematical formula of Min–Max scaler is shown below:

$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (3)$$

where  $X$  = Original value,  $X_{min}$  = Feature minimum value,  $X_{max}$  = Feature maximum value, and  $X_{scaled}$  = Scaled value.

The NGX dataset was divided into an 85% dataset for training and validation purposes (i.e., from 4 January 2010 to approximately mid-2023) and a 15% dataset (i.e., from mid-2023 to 7 June 2024), the latter of which was reserved for testing the model’s performance on unseen data. Two approaches were employed for running the algorithm: i.e., the classic approach (with 8 input features) and the OBV inclusion approach (which has 9 input features). The splitting ratio for the training and test data is shown in Table 2:

**Table 2.** NGX data training and test data.

| Dataset Split               | Split% Per Feature | Dataset/Observations Per Feature |
|-----------------------------|--------------------|----------------------------------|
| Training and Validation Set | 85% of 3573        | 3037                             |
| Testing Set                 | 15% of 3573        | 536                              |

The division was performed chronologically to respect the temporal structure of the data. This ensured that no future data were used to train the model. It mirrors real-world conditions where future data are unknown at the time of prediction. The model was trained and validated using TimeSeriesSplit. TimeSeriesSplit, a specific Scikit-learn (version 1.6.0) time-series cross-validation (TSCV), was used on the chronologically sorted NGX dataset with five folds ( $n = 5$ ). In this method, the dataset is iteratively split into training and validation sets, where each fold uses a progressively larger portion of the dataset for training, and the subsequent fold acts as the validation set. The cross-validation sequence is provided in Supplementary S1.

### 2.1. Feature Re-Engineering

The NGX dataset was re-engineered to generate technical indicators using cleaned NGX market data in order to improve predicted accuracy and gain insight into market dynamics. The multi-dimensional approach of including technical indicators to assess price fluctuations, patterns, and trading signals improves the ability of models to detect market swings and provide accurate forecasts [10]. The technical indicators that were added to the historical price and return series for market prediction included the Simple Moving Average (SMA-15 and SMA-45), Relative Strength Index (RSI), Moving Average Convergence Divergence (MACD), Bollinger Band (Middle, High, and Low), and OBV. Bollinger Bands represent regions of high or low prices, while RSI confirms whether the market is overbought or oversold. Systematic Moving Averages (SMAs) show the direction of market trends, MACD validates momentum, and OBV uses volume flow to forecast fluctuations in stock prices.

Return series is a classic input feature for predicting stock prices. It is the financial gain or loss derived from an investment or stock portfolio within a certain timeframe. Rates of return are determined by calculating the percentage change in market price between two time periods [11]. The equation is shown below:

$$Return = \frac{Current\ Price - Previous\ Price}{Previous\ Price} \times 100 \tag{4}$$

The SMA-15 and SMA-45 are statistical moving average measures employed to detect patterns in market data. These figures are derived by computing the average closing prices across time periods of 15 and 45. The combination of the indications can identify crossings, which are useful for making selections into purchasing or selling. The equation is shown as follows:

For a 15-period SMA:

$$\frac{P1 + P2 + \dots P15}{15} \quad (5)$$

For a 45-period SMA:

$$\frac{P1 + P2 + \dots P45}{45} \quad (6)$$

where “ $P$ ” is the closing price, and the rolling window period ( $n$ ) is 45 and 15.

The Relative Strength Index (RSI) is a technical indicator used to measure the speed and extent of price changes. It is often used to identify situations when a market is showing either excessive buying or selling [12]. In this study, the RSI values were computed using a window duration of 14 days, as shown below.

$$\text{Average gain} = \frac{\sum_{i=1}^n \text{Gain}}{n} \text{ and } \text{Average loss} = \frac{\sum_{i=1}^n \text{Loss}}{n} \quad (7)$$

Relative Strength (RS) is given by:

$$\frac{\text{Average Gain}}{\text{Average Loss}} \quad (8)$$

$$\text{RSI} = 100 - (100 / (1 + \text{RS})) \quad (9)$$

where

Gain = Positive price change between the closing prices of two consecutive periods (only positive changes are considered).

Loss = The absolute value of the negative price change (only negative changes are considered).

$n$  = Number of periods (commonly 14).

Average Gain: The sum of all gains (positive price changes) divided by the number of periods being considered (often 14 periods).

Average Loss: The sum of all losses (negative price changes) divided by the number of periods.

The MACD is a momentum indicator applied to detect trends and monitor the correlation between two moving averages of stock prices. The computation of MACD entails the utilization of exponential moving averages (EMAs), which are often shown with a signal line, as illustrated below:

$$\text{EMA}_{12} \text{ (Short term)} = \text{Market Price}_{\text{Current}} \times \left( \frac{2}{12+1} \right) + \text{Market Price}_{\text{Previous}} \times \left( 1 - \frac{2}{12+1} \right) \quad (10)$$

$$\text{EMA}_{26} \text{ (Long term)} = \text{Market Price}_{\text{Current}} \times \left( \frac{2}{26+1} \right) + \text{Market Price}_{\text{Previous}} \times \left( 1 - \frac{2}{26+1} \right) \quad (11)$$

$$\text{MACD} = \text{EMA}_{12} - \text{EMA}_{26} \quad (12)$$

$$\text{Signal Line} = \text{MACD}_{\text{Current}} \times \left( \frac{2}{9+1} \right) + \text{Signal Line}_{\text{Previous}} \times \left( 1 - \frac{2}{9+1} \right) \quad (13)$$

$$\text{MACD Histogram} = \text{MACD} - \text{Signal Line} \quad (14)$$

Bollinger Bands provide a visual representation of market volatility. This indicates a period of instability characterised by substantial price fluctuations and identifies instances of excessive stock purchasing or selling. The gradient between the upper and lower bands varies in response to market volatility. The Bollinger Mid, Upper, and Lower variables



utilised in this study indicate the mean, maximum, and minimum levels of volatility in technical analysis. They are computed using the following formula:

$$\text{Mid Band} = SMA_{20} \quad (15)$$

$$\text{Standard Deviation (SD)} = \sqrt{\frac{1}{n} \sum_{i=1}^n (P - SMA)^2} \quad (16)$$

$$\text{Upper Band} = \text{Mid Band} + (\text{SD} \times k) \quad (17)$$

$$\text{Lower Band} = \text{Mid Band} - (\text{SD} \times k) \quad (18)$$

where  $k = 2$  (Bollinger Band standard).

The use of OBV in this analysis provides a unique approach to using the cumulative total volume, which modulates the volume in response to price fluctuations by either adding or deleting it. This method is employed to validate pricing trends and provide insights into the magnitude of price swings by analysing variations in volume [13]. The algorithm for OBV is shown below:

$$\text{Price Difference Calculation } (\Delta P) = P_{\text{close}(t)} - P_{\text{close}(t-1)} \quad (19)$$

where

$\Delta P$  = Price difference

$P_{\text{close}(t)}$  = Current closing price

$P_{\text{close}(t-1)}$  = Previous closing price.

Volume Adjustment Based on Price Movement:

- Increase in price ( $P_{\text{close}(t)} > P_{\text{close}(t-1)}$ ):

- Add the current volume  $V_{(t)}$  to the previous OBV:

$$OBV_{(t)} = OBV_{(t-1)} + V_{(t)} \quad (20)$$

- Decrease in price ( $P_{\text{close}(t)} < P_{\text{close}(t-1)}$ ):

- Subtract the current volume  $V_{(t)}$  from the previous OBV:

$$OBV_{(t)} = OBV_{(t-1)} - V_{(t)} \quad (21)$$

- Unchanged price ( $P_{\text{close}(t)} = P_{\text{close}(t-1)}$ ):

- Subtract the current volume  $V_{(t)}$  from the previous OBV:

$$OBV_{(t)} = OBV_{(t-1)} \quad (22)$$

- Cumulative Calculation: OBV starts with an initial value, set to zero, and the volume is then added or subtracted based on price changes at each time step, which leads to the cumulative OBV over time.

The dataset was used in two different approaches to define the input features for the model, i.e., the classic and OBV inclusion approaches. The classic approach involves the usage of return, historical price, and price technical indicators as input features, while the OBV inclusion approach includes all the features of the classic approach with the addition of the OBV indicator as input feature integration into the model.

## 2.2. Algorithm Selection for NGX Market Data

The selection of a specific algorithm type depends on the computational requirements necessary to generate a desired outcome [14]. Various machine and deep learning models are commonly used for regression analysis, but this comparative study used the SVR, RNN, and LSTM models due to their unique adaptability, strength, and high performance on different data types in several research contexts [15,16]. An evaluation of the RNN and LSTM showed their effectiveness in capturing intricate temporal relationships in comparison to other machine learning methods [17].

### 2.2.1. Model Architecture, Development and Training

SVR is a variant of the Support Vector Machine (SVM) technique that is specifically employed for regression tasks [18]. The architectural components of SVR are kernel function, regularisation parameter (C), epsilon ( $\epsilon$ ) and support vectors. The model separates higher-dimensional data points using the kernel function. In this study, we used the linear kernel for final model training. This is simple, quick, and effective when the input characteristics and target variables are linearly related. The model balances a wide class margin and low error rates with the regularization parameter (C). The “epsilon” ( $\epsilon$ ) specifies the area where the model does not penalise errors. This makes the model less sensitive to small data changes and improves performance. Support vectors are the margin-boundary data points that determine the decision boundary (the line or curve separating the data points).

For the SVR model used in our work, the objective function was created and hyperparameter ranges were selected, as shown in Table 3.

**Table 3.** Hyperparameters selected for SVR.

| Hyperparameter | Value Range                 |
|----------------|-----------------------------|
| C              | 0.001, 1000                 |
| Epsilon        | 0.001–10                    |
| Kernel         | Linear, Poly, RBF, Sigmoid  |
| Degree         | 2, 5 (for kernel = Poly), 3 |

The recurrent neural network (RNN) model was developed and trained using TensorFlow and Keras API. The model architecture comprises two recurrent neural network (RNN) layers with alternating dropout layers, as well as a dense layer with a single unit specifically designed for the regression task [19,20]. The dense layer of the feed-forward neural network (RNN) generates prediction output, while the dropout layer introduces noise during training to reduce overfitting. The input layer shape rotates between time steps of 30 and 60. The hyperparameters selected for the training of the RNN network are provided in Table 4 below.

**Table 4.** Hyperparameters selected for RNN.

| Hyperparameter | Value Range        |
|----------------|--------------------|
| Units          | 32, 128            |
| Dropout rates  | 0.1 to 0.5         |
| Batch size     | 32, 64, 128        |
| Learning rate  | 0.00001 to 0.01    |
| Optimiser      | Adam               |
| Loss function  | Mean Squared Error |

The Long Short-Term Memory (LSTM) model used in our study is composed of two input layers, a dropout layer, and a dense output layer. The decision to use two input layers instead of the classical one-input LSTM layer was made to effectively handle long-term sequence dependencies, address the complexity of market data, and improve the performance of the model [21]. Using a dropout layer as a regularization technique enhances the generalisation of the model. The adaptability and non-linearity of the dense output layer, achieved via activation functions, effectively capture intricate data patterns and connections. These features are utilised to produce continuous values in regression tasks. The dense layer reduces the dimensionality of information, makes predictions, integrates, and translates learning features to facilitate decision formation. The LSTM model was trained using the same hyperparameters as the RNN model described in Table 4.

The hyperparameters were evaluated using fold-specific Mean Squared Error. The algorithm finds optimal hyperparameters using Optuna and TimeriesSplit. This improves model performance on unknown test data. The training data and target variable were used to fit the SVR, RNN, and LSTM models with the optimal hyperparameters during final training. Regression metrics were used on a test dataset not used during training to evaluate model prediction accuracy and dependability. The model development can be accessed in the following GitHub repository: <https://github.com/Sidikat123/Model-Data-Analytics-/tree/main> (accessed on 13 December 2024).

The performance of the machine learning and deep learning models was evaluated using various regression metrics, including Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), and R-squared ( $R^2$ ), after the training process. These metrics were selected because of their simplicity, wide acceptance, and ability to provide robust evaluations [22].

#### 2.2.2. Hardware and Computational Resources:

A premium Google Colab, a cloud-based computational resource, with Python 3 run type, was employed for running the machine and deep learning tasks [23]. The laptop hardware specifications used for uploading data to the Google Colab include: Device Name: LENOVO-ALSA01, Processor: Intel(R) Core(TM) i5-7200U, CPU @ 2.50 GHz, 2.71 GHz, Installed RAM: 8.00 GB (7.79 GB usable), System Type: 64-bit operating system, x64-based processor. Furthermore, a premium NVIDIA T4 GPU hardware accelerator on Google Colab was employed for running the RNN and LSTM due to its suitability for deep learning tasks. The SVR model was initially run on CPU on Google Colab but this was changed to NVIDIA T4 GPU. However, it was observed that there was no significant benefit in the optimization and training time of using T4 GPU on SVR in the cloud environment. This could be attributed to sequential processing and time complexity ( $O(n^2)$  to  $O(n^3)$ ) of SVR, where  $n$  is the number of data points. The optimization and training time for the SVR classic approach with linear kernel was around 90 min, while it took longer (around 105 min) for the OBV inclusion approach with rbf (radial basis function) kernel. The inference time for SVR was within one minute for the test set. RNNs tend to optimise and train faster than LSTMs with around 55 min per run. The inference time was much faster than the training time and it took within a minute to process the entire dataset. LSTM optimisation and training time took around 70 min, while the inference time was within a minute.

### 3. Results and Discussion

The evaluations of the predictive models are discussed below. The SVR, RNN, and LSTM models were trained and tested using historical data from the Nigerian Stock Exchange, with a particular focus on forecasting the NGX All-Share Index.

### 3.1. Support Vector Machines for Regression (SVR)

The NGX dataset's features were converted into a two-dimensional matrix format for an SVR model. Table 5 below shows the evaluation result of the SVR using regression metrics.

**Table 5.** Evaluation result of SVR.

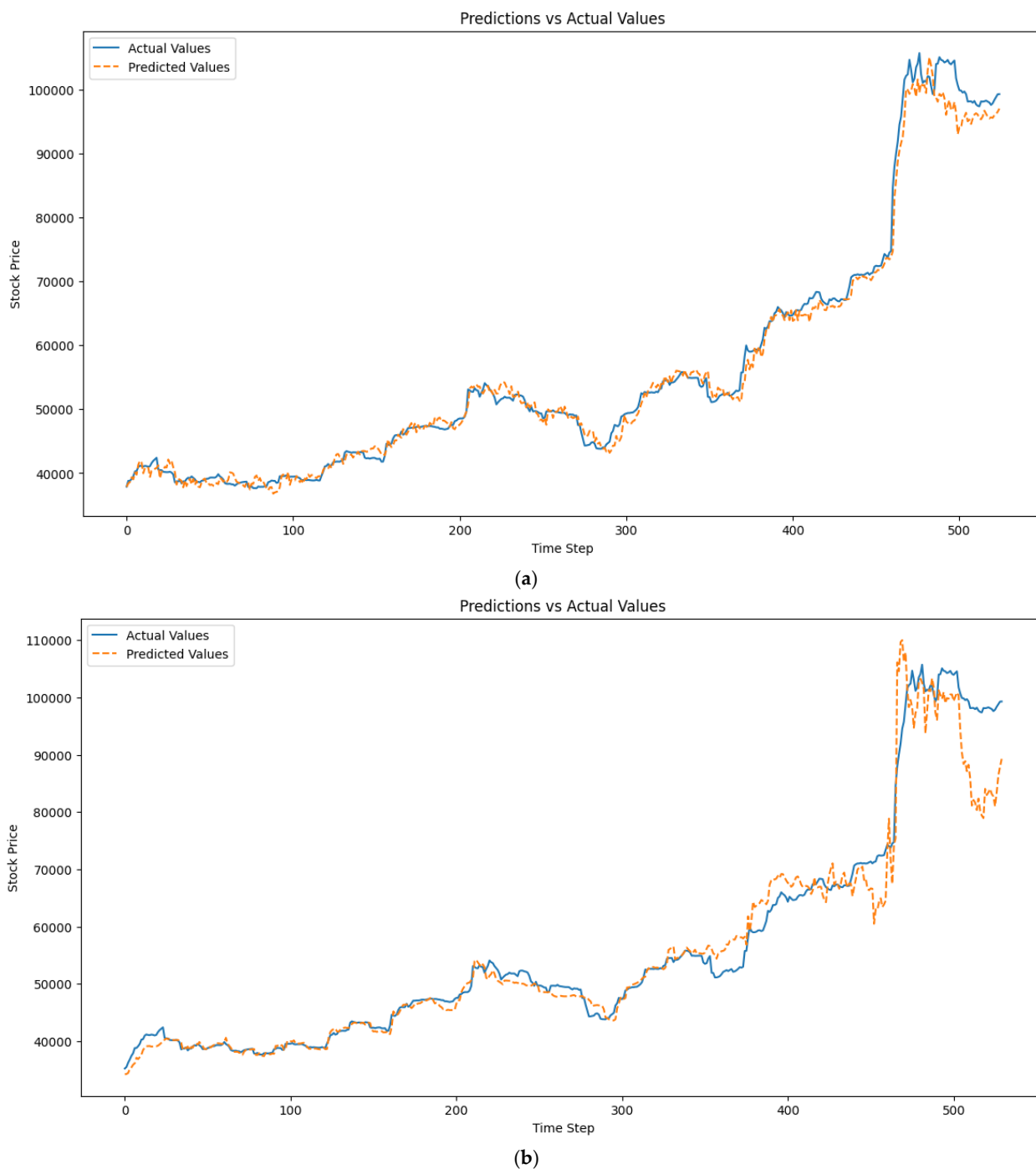
|                                | Best Hyperparameters (Without OBV)<br>{'C': 0.2520357450941616, 'Epsilon':<br>0.027026985756138284, 'Kernel':<br>'Linear'} | Best Hyperparameters (with OBV)<br>{'C': 1.1625895249237326, 'Epsilon':<br>0.005801399787450704, 'Kernel':<br>'Rbf'} |
|--------------------------------|--|--|
| Cross-Validation Training Loss | 0.0011   | 0.00133  |
| Average Final Validation Loss  | 0.00197  | 0.0136   |
| Test MAE                       | 0.0143 (1231.61)   | 0.027 (2318.66)  |
| MSE                            | 0.000459 (3,396,945.36)  | 0.00242 (17,878,018.81)  |
| RMSE                           | 0.0214 (1843.08)   | 0.0491 (4228.24)   |
| MAPE (%)                       | 2.03   | 3.4  |
| R-Squared                      | 0.99   | 0.95   |

The results in Table 5 reveal that the classic approach (training loss: 0.00110) outperforms the OBV inclusion method (training loss: 0.00133), with a lower validation loss (0.00197 vs. 0.0136), indicating better generalisation to unseen data. The training and validation plots are provided in Supplementary S2.

The classic approach also demonstrates superior predictive accuracy with lower MAE (0.0143 vs. 0.027), MSE (0.000459 vs. 0.00242), RMSE (0.0214 vs. 0.0491), and MAPE (2.03% vs. 5.36%), alongside a higher R-squared value (99% vs. 95%). These metrics suggest the classic approach achieves more reliable, accurate predictions with fewer errors compared to the OBV inclusion method. The inclusion of OBV added complexity without improving model performance, potentially leading to overfitting. While incorporating additional features such as OBV can enhance predictions in some cases, it may hinder non-linear models such as SVR if the added information is not effectively utilised. These findings align with prior research and additional diagnostics conducted on NGX stock data [24]. The predicted and actual values plot for the classical approach with and without OBV is shown in Figure 1a,b.

As observed in Figure 1a,b, the exclusion of OBV closely aligns with predicted and actual NGX Index values, effectively capturing market trends, particularly in the early and mid-segments. However, it diverges slightly during peak periods. However, when OBV is included, there is greater divergence. This shows SVR struggles with high volatility, reducing predictive accuracy. It is also worth noting that removing outliers from NGX time-series data is challenging due to temporal dependencies, potential data loss, and its impact on model performance. Unlike non-temporal data, perceived outliers in time-series data may represent significant events, such as market crashes, which are critical for analysis. Traditional outlier detection methods, such as Z-scores, often fail due to the dynamic nature of stock market trends and seasonal patterns. Investigating a spike in validation loss in fold 2 revealed high market volatility and non-linearity that the Radial Basis Function (RBF) kernel of SVR struggled to capture [25,26]. Stock prices, influenced by diverse factors such as economic events and investor sentiment, are highly volatile and non-stationary. Research highlights that financial time-series data can exhibit unique characteristics, such as volatility clustering and regime transitions, which can lead to increased validation errors. Despite using regularisation and smoothing techniques, such as moving averages, SVR exhibited higher validation loss (see Supplementary S2) during periods of market shocks and trend reversals. However, the actual vs. predicted scatter plot demonstrates a strong

linear relationship, indicating that the SVR model is generally accurate in predicting NGX price indices, with only a few extreme outliers affecting performance [27]. These findings align with prior research on model instability in non-stationary financial time series [28].



**Figure 1.** (a) Predicted vs. actual value (without OBV); (b) Predicted vs. actual value (with OBV).

### 3.2. Recurrent Neural Network

For the RNN model, we compared six different variations of the model, as shown in Table 6. We compared the NGX unnormalised and normalised datasets, both with an optimised RNN model (based on the default values from Keras), and an optimised RNN model, based on the folds and cross-validation as selected by Optuna. We also compared the 30-day and 60-day time steps for the optimised model, but only the 30-day time steps for the unoptimised model due to the unsatisfactory shape of the 60-day time step.

**Table 6.** Evaluation metrics for the RNN model variations.

| Evaluation Metrics | RNN Model Variations  |  |  |   |  |   |
|--------------------|---|--|--|---|--|---|
|                    | 30-Day Step-Time (Unnormalised Value) Unoptimised RNN Model | Transformed Original Value (Unoptimised RNN Model) | Optimised RNN (30-Day Time Step Without OBV) * | Optimised RNN (60-Day Time Step Without OBV) ** | Optimised RNN (30-Day Time-Stamp with OBV) *** | Optimised RNN (60-Day Time-Stamp with OBV) **** |
| MAE                | 0.07483   | 6434.68  | 0.011321 (973.52)                              | 0.012490 (1074.05)                              | 0.019642 (1689.04)                             | 0.017465 (1501.84)                              |
| MSE                | 0.01043   | 77,130,110.92                                      | 0.000388 (2,870,386.08)                        | 0.000613 (4,533,240.36)                         | 0.000872 (6,447,080.45)                        | 0.000982 (7,264,937.75)                         |
| RMSE               | 0.10213   | 8782.37  | 0.019702 (1694.22)                             | 0.024760 (2129.14)                              | 0.029528 (2539.11)                             | 0.031345 (2695.35)                              |
| MAPE (%)           | 11.27   | 11.27  | 1.57   | 1.65  | 2.72   | 2.17  |
| R-Squared          | 0.78613   | ~0.79  | 0.992  | 0.987   | 0.982  | 0.980   |

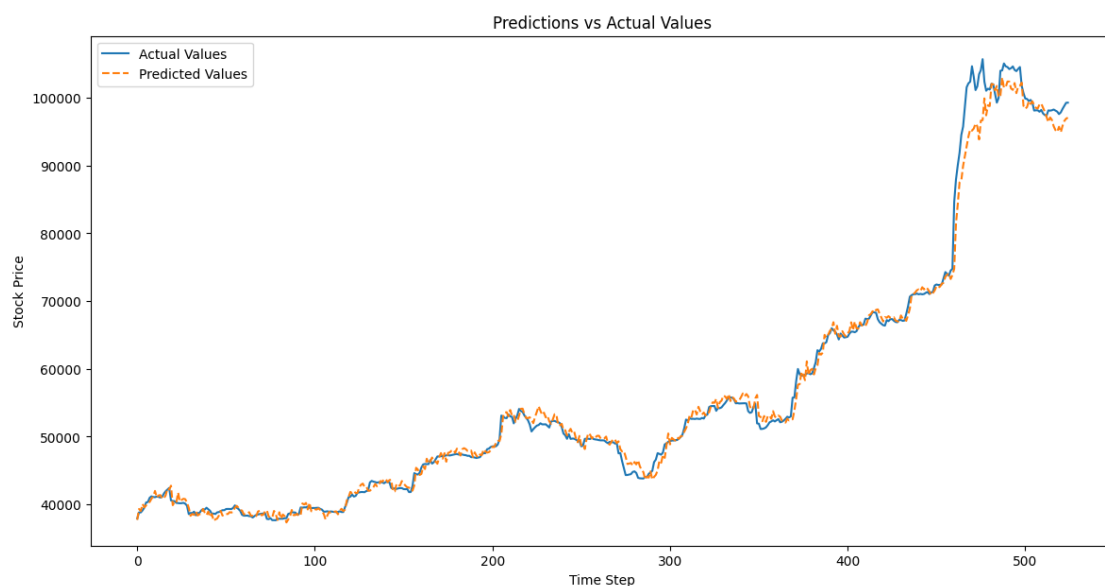
\* Based on best hyperparameters where {units = 122, Dropout rate = 0.21057927576994717, Batch size = 32, Learning rate = 0.00045995847308281305}. \*\* Based on best hyperparameters where {units = 105, Dropout rate = 0.15860460902099602, Batch size = 64, Learning rate = 0.004964007144265714}. \*\*\* Based on best hyperparameters where {units = 94, Dropout rate = 0.39808512912645366, Batch size = 32, Learning rate = 0.0033264056797172536}. \*\*\*\* Based on best hyperparameters where {units = 79, Dropout rate = 0.28896636206794524, Batch size = 64, Learning rate = 0.009618218785212735}.

The unnormalised and transformed original value shows high training and validation loss (see Supplementary S2), and high error of the NGX test datasets using evaluation metrics such as MAE (0.0748), MSE (0.010), MRSE (0.10), MAPE (11%), and the suboptimal result of 0.79 of R-squared. However, MSE is a differentiable function unlike MAE, but the high error of other metrics shows the inaccuracy of the regression model. MAPE is a percentage error between the model-fitted values and the observed data values. However, the acceptability depends on the industry, application, and specific context [29]. For the NGX stock market, a MAPE above 10% is considered inaccurate because of the financial implications and loss on the investment.

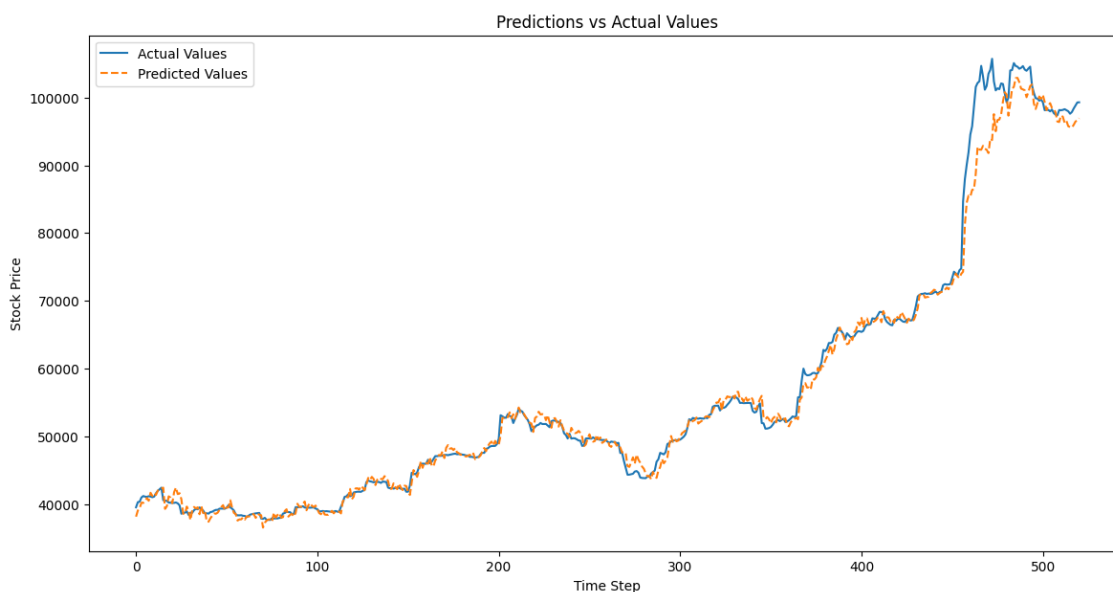
For the optimised model, the prediction was performed using the classic and OBV inclusion approaches. The financial market has several endogenous variables, such as volatility, dividend yield, bid-ask spread, and market sentiment, but the main endogenous variables are price, volume, and returns [30]. Training loss for the classic method increased with longer time steps, highlighting the vanishing gradient problem without optimising the RNN model [31–33]. However, adding OBV reduced training loss for the medium to long term, potentially mitigating this limitation. Validation loss decreased in both methods as the time steps increased, demonstrating robust model performance without overfitting. MAE for the classic approach increased slightly, while OBV inclusion showed reduced absolute errors over longer periods, despite starting higher. RMSE increased with more features and time steps, reflecting difficulties in modelling specific data segments. R-squared values remained high, between 0.98 and 0.99, across methods. The predicted and actual value plots for the optimised RNN variations are shown Figure 2a–d.

As seen from Figure 2a–d, the inclusion of OBV in the optimised RNN models enhances its sensitivity to price movements, improving its responsiveness to market volatility. This enables the model to dynamically react to significant price changes, potentially increasing prediction accuracy during volatile periods. However, it also introduces noise during stable periods, resulting in slightly higher deviations toward the end of predictions. The comparison with the classic approach reveals that OBV integration adds depth and adaptability to the model, making it more effective for volatile markets. The training loss (Supplementary S2) of the classic 30-day time step model shows a rapid decline, reaching a minimum loss of 0.002479, indicating quick learning and good generalisation without overfitting. The 60-day time step model follows a similar pattern but faces challenges with longer sequences, achieving a slightly lower minimum loss of 0.002 and a stable validation loss of 0.00008. The OBV inclusion for 30-day time steps starts with higher initial complexity

but quickly reduces to a minimum training loss of 0.0055 and a validation loss of 0.000138, demonstrating the ability to capture more complex patterns. Similarly, the OBV 60-day time step model achieves a minimum training loss of 0.0036 and validation loss of 0.0001, indicating robustness and good generalisation [34]. Scatter plots (Supplementary S2) reveal a strong linear relationship between actual and predicted values, with the classic models tightly aligned along the diagonal. OBV inclusion slightly improves predictive accuracy, particularly in the 60-day model, complementing price data with volume information. The results suggest robust RNN performance across configurations, validating their use in financial forecasting. OBV adds depth to predictions, aligning with the findings in [35]. The residual plots are also shown in Supplementary S2, with values centred around zero, and a positive skew [36] also validates the inclusion of OBV for the RNN model.

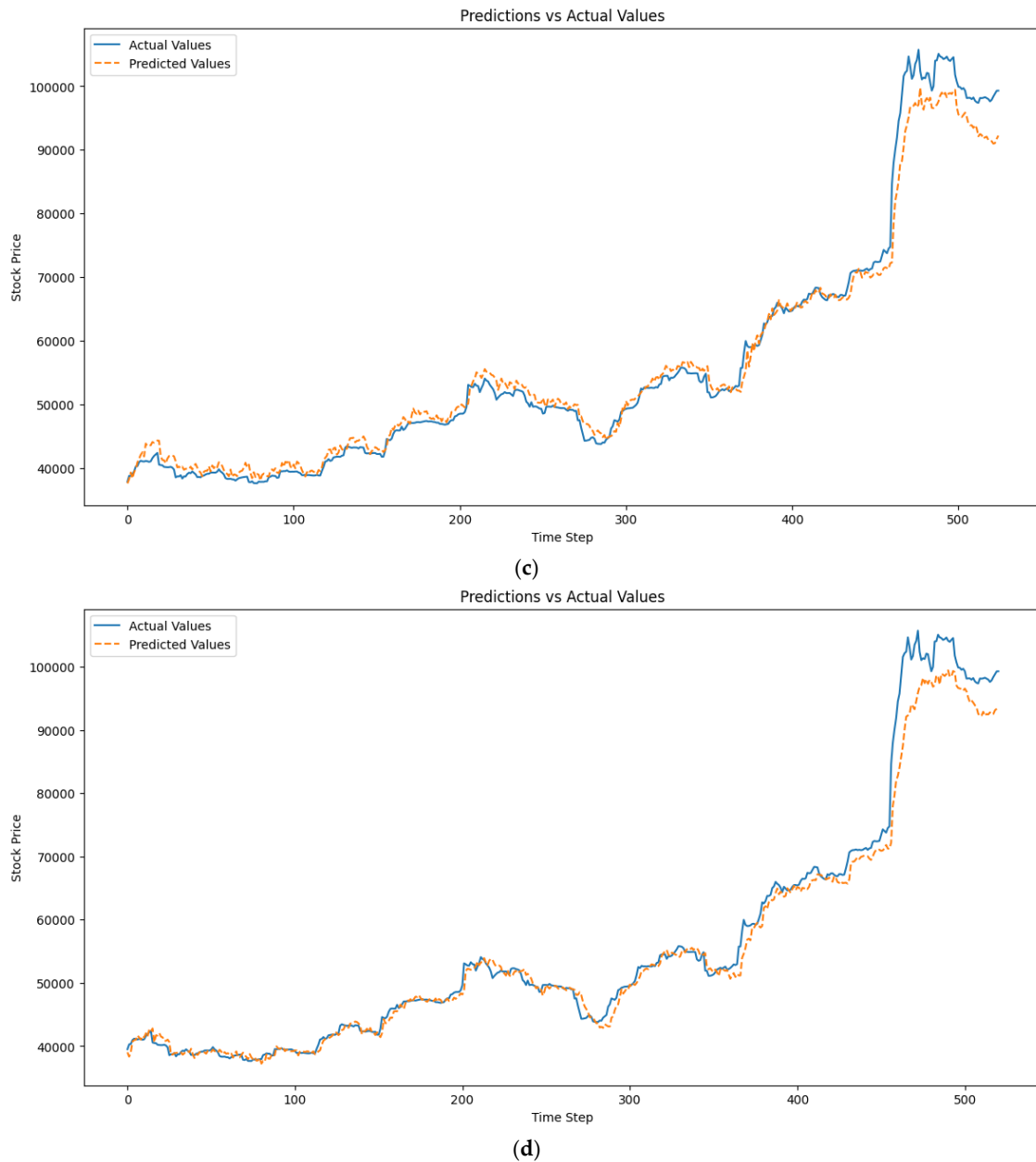


(a)



(b)

Figure 2. Cont.



**Figure 2.** (a) RNN 30-day time step without OBV (classic). (b) RNN 60-day time step without OBV (classic). (c) RNN 60-day time step with OBV. (d) RNN 60-day time step with OBV.

### 3.3. Long Short-Term Memory Network

LSTM followed a similar procedure to the RNN architecture, and the result obtained is shown in Table 7.

The results in Table 7 show that LSTM can predict accurately even without optimisation of the model or normalisation of the dataset. However, we experimented with the normalised and optimised variants and obtained a better R-squared performance [37,38]. This is due to the sequential nature of the time-series data. The evaluation metrics show improved performance of the LSTM model with increasing time steps and the inclusion of On-Balance Volume (OBV), as confirmed by studies [39,40]. In the classic approach, MAE decreases from 0.015 to 0.012 and MSE decreases from 0.000745 to 0.000485 as time steps increase. Including OBV further lowers MAE to 0.009662, indicating enhanced predictive accuracy. RMSE and MAPE also decrease, with MAPE reducing from 1.96% to 1.61% in the classic approach and to 1.33% with OBV inclusion. The R-squared value increases to



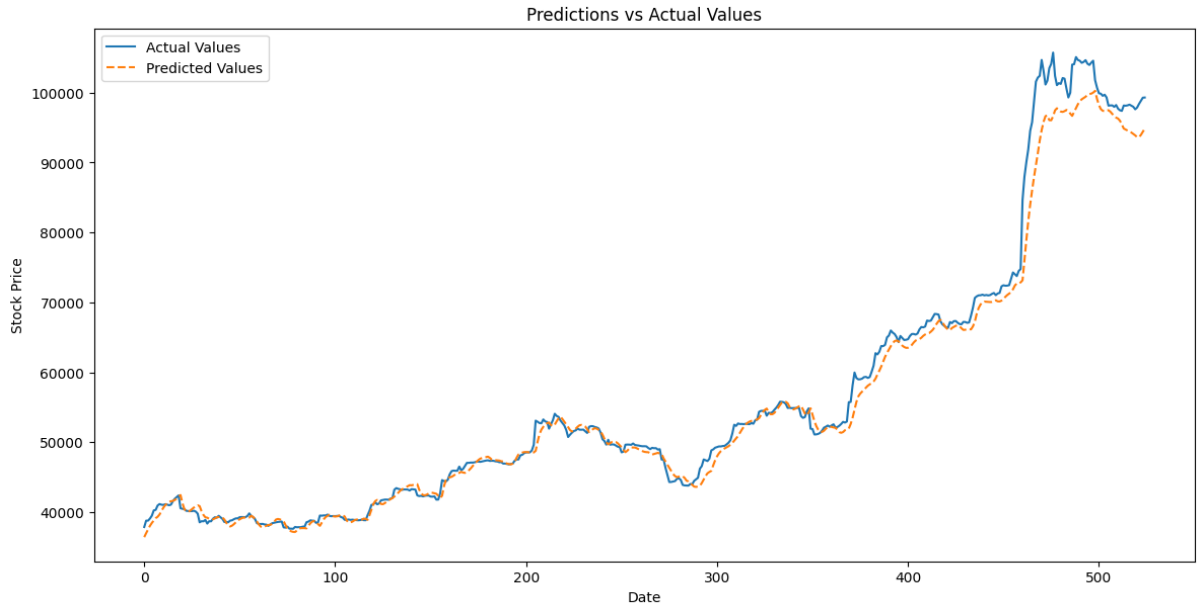
0.993, demonstrating strong alignment between predictions and actual data and confirming the model’s robustness. Longer time steps improve the LSTM model’s ability to capture trends, while OBV enhances accuracy by adding volume-related insights, a key factor in stock price movements [41]. The predicted vs. actual value plot is shown in Figure 3a–d.

**Table 7.** Evaluation metrics for the LSTM model variations.

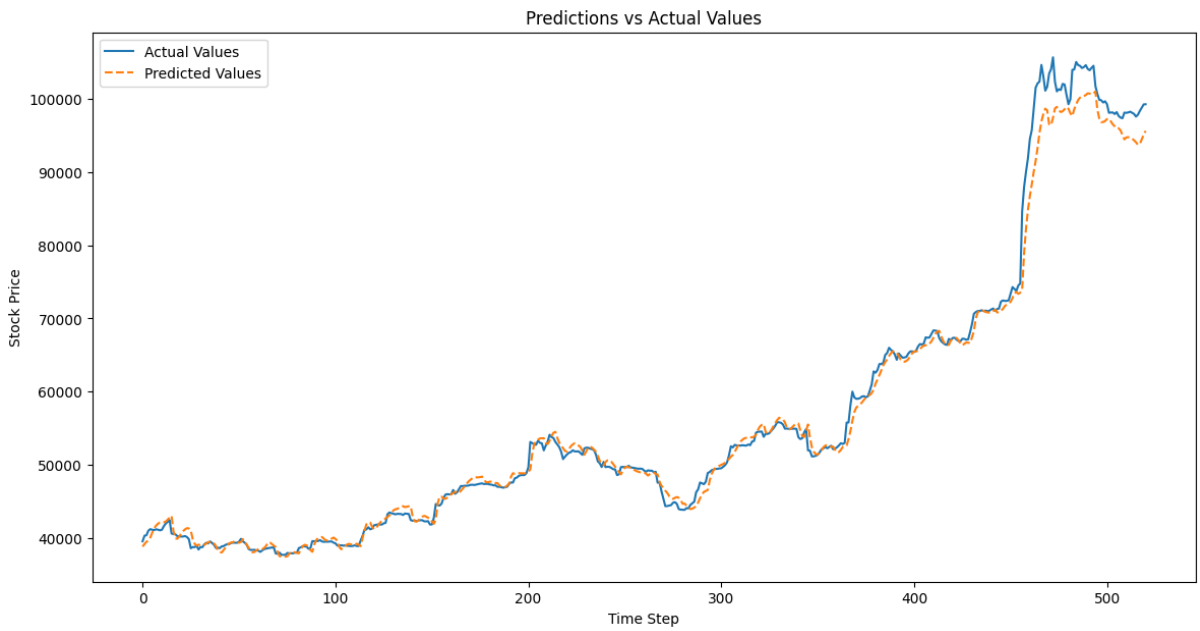
| Evaluation Metrics | LSTM Model Variations  |   |   |  |  |   |
|--------------------|--|---|---|--|--|---|
|                    | 60-Day Time Step (Unnormalised Value) Unoptimised LSTM Model | Transformed Original Value (Unoptimised LSTM Model) | Optimised LSTM (30-Day Time Step Without OBV) * | Optimised LSTM (60-Day Time Step Without OBV) ** | Optimised LSTM (30-Day Time Step with OBV) *** | Optimised LSTM (60-Day Time Step with OBV) **** |
| MAE                | 0.015733   | 1352.93   | 0.015346 (1319.58)                              | 0.012397 (1066.05)                               | 0.014333 (1232.53)                             | 0.009662 (830.84)                               |
| MSE                | 0.000734   | 5,425,600.48  | 0.000745 (5,512,396.17)                         | 0.000485 (3,586,624.63)                          | 0.000756 (5,593,662.96)                        | 0.000343 (2,536,732.36)                         |
| RMSE               | 0.027088   | 2329.29   | 0.027303 (2347.85)                              | 0.022024 (1893.84)                               | 0.027504 (2365.09)                             | 0.018522 (1592.71)                              |
| MAPE (%)           | 2.02   | 2.02  | 1.96  | 1.61   | 1.96   | 1.33  |
| R-squared          | 0.985  | 0.980   | 0.984   | 0.990  | 0.984  | 0.993   |

\* Based on best hyperparameters where {units = 122, Dropout rate = 0.10131318703976958, Batch size = 128, Learning rate = 0.0038833897789281994}. \*\* Based on best hyperparameters where {units = 65, Dropout rate = 0.20445711460250626, Batch size = 32, Learning rate = 0.007529066043550584}. \*\*\* Based on best hyperparameters where {units = 122, Dropout rate = 0.27017135632721034, Batch size = 128, Learning rate = 0.0033264056797172536}. \*\*\*\* Based on best hyperparameters where {units = 104, Dropout rate = 0.2657214090579303, Batch size = 32, Learning rate = 0.0036850279954446555}.

The plots in Figure 3a–d show actual versus predicted NGX price indices using LSTM models with classic and OBV inclusion approaches over 30- and 60-day time steps. The classic models closely track actual values but lag during rapid price changes, with the 60-day model performing slightly better by capturing upward trends more accurately. Including OBV improves predictions, especially during significant price changes, as volume trends help capture market volatility. While all the models exhibit some lag due to LSTM’s reliance on past data, OBV inclusion reduces this discrepancy. The LSTM model’s learning curve, scatter plots, and residual values are provided in Supplementary S2. Analysis of the graphs provides further evidence of the usefulness of the OBV features. The classic LSTM models show rapid initial loss reduction, with the 30-day model reaching a minimum loss of 0.002163 and the 60-day model achieving 0.001993. Longer time steps improve generalisation and capture underlying trends. Including OBV increases initial loss due to added complexity but improves predictive accuracy as the models adapt. The LSTM 60-day OBV model demonstrates effective handling of complexity, achieving a validation loss of 0.000076. This confirms that longer sequences help LSTM utilise OBV effectively [42–44]. Analysis of the scatter plots also shows a strong linear relationship between actual and predicted NGX prices using LSTM models. The LSTM 60-day classic and OBV models exhibit tighter clustering around the diagonal, indicating improved accuracy and consistency compared to the 30-day models. The inclusion of OBV and longer time steps enhances predictions by capturing more temporal patterns and market dynamics effectively [45].

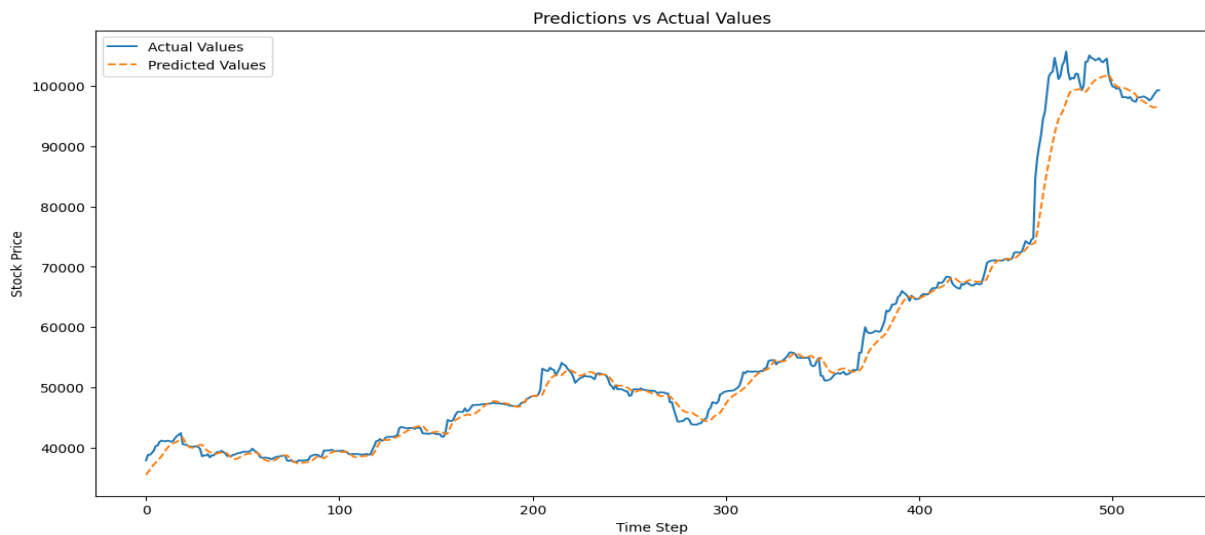


(a)

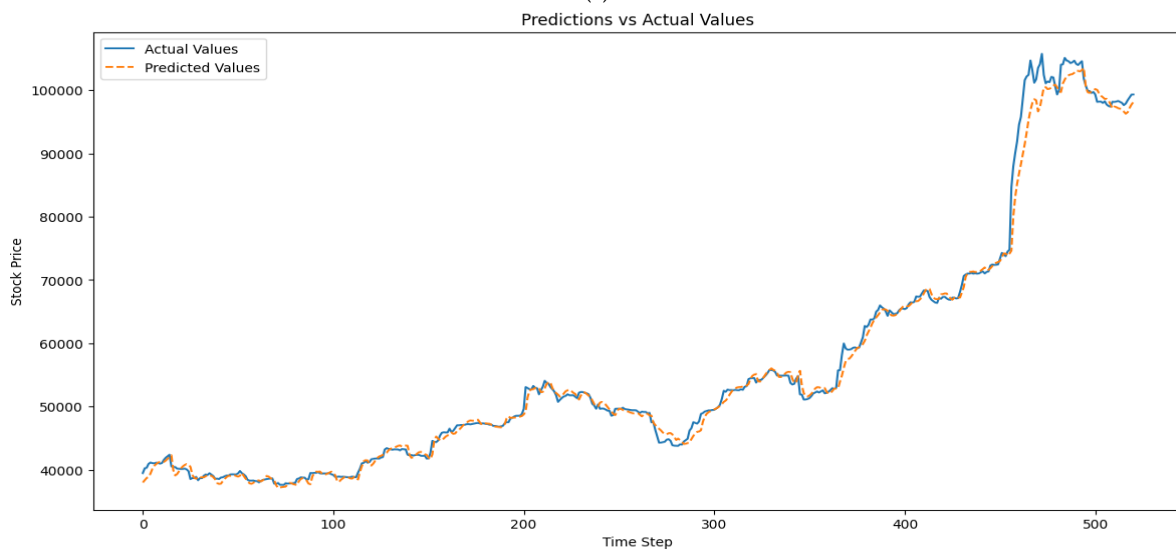


(b)

Figure 3. Cont.



(c)



(d)

**Figure 3.** (a) LSTM 30-day time step without OBV (classic). (b) LSTM 60-day time step without OBV (classic). (c) LSTM 30-day time step with OBV. (d) LSTM 60-day time step with OBV.

#### 4. Conclusions

This study compared the performance of SVR, RNN, and LSTM models in predicting NGX market prices. SVR demonstrated strong performance, with an MSE value of 0.00459, RMSE value of 0.0214, and R-squared value of 0.99 with smaller datasets (without OBV values) and simpler market conditions, although it struggled to capture the complex temporal dependencies in stock price data. RNN also performed well with smaller datasets without the OBV variables, with the best performance obtained for a 30-day time step, with an MSE value of 0.000388, RMSE value of 0.019702, and R-squared value of 0.992. However, it showed limitations due to the vanishing gradient problem, resulting in lower predictive accuracy over long periods. LSTM emerged as the most effective model for stock price prediction in this study, with an MSE value of 0.000343, RMSE value of 0.018522, and R-squared value of 0.993 for the optimised model over a 60-day time step. Its ability to retain information over long time sequences allowed it to outperform both SVR and RNN, especially when dealing with volatile and unpredictable market trends. The results demonstrated that LSTM achieved the lowest error results and the highest R-squared values, highlighting its superior ability to capture both short-term and long-term dependencies in

stock market data. The research findings suggest that while SVR can be a useful tool for simple financial forecasting, LSTM provides far more accurate predictions for time-series data, particularly in complex and volatile markets. The integration of exogenous data, such as financial news sentiment, could further enhance the accuracy of deep learning models, offering valuable insights for investors and financial analysts.

**Supplementary Materials:** The following supporting information can be downloaded at: <https://www.mdpi.com/article/10.3390/analytics4010005/s1>.

**Author Contributions:** Conceptualization, S.A.-L. and O.S.; methodology, S.A.-L. and O.S.; software, S.A.-L. and O.S.; validation, O.S., O.P. and O.O.; formal analysis, S.A.-L. and O.S.; investigation, S.A.-L.; resources, O.S.; data curation, S.A.-L.; writing—original draft preparation, S.A.-L.; writing—review and editing, O.S., O.P. and O.O.; supervision, O.S.; project administration, O.P., O.O. and O.S. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The data presented in this study are available on request from the corresponding author. The data are not publicly available due to privacy.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

- Han, H.; Liu, Z.; Barrios Barrios, M.; Li, J.; Zeng, Z.; Sarhan, N.; Awwad, E.M. Time series forecasting model for non-stationary series pattern extraction using deep learning and GARCH modeling. *J. Cloud Comput.* **2024**, *13*, 2. [CrossRef]
- Abuein, Q.Q.; Shatnawi, M.Q.; Aljawarneh, E.Y.; Manasrah, A. Time Series Forecasting Model for the Stock Market using LSTM and SVR. *Int. J. Adv. Soft Comput. Appl.* **2024**, *16*, 169–185. [CrossRef]
- Fama, E.F. The behavior of stock-market prices. *J. Bus.* **1965**, *38*, 34–105. [CrossRef]
- Lo, A.W. The adaptive markets hypothesis: Market efficiency from an evolutionary perspective. *J. Portf. Manag. Forthcom.* 2004. Available online: [https://papers.ssrn.com/sol3/papers.cfm?abstract\\_id=602222](https://papers.ssrn.com/sol3/papers.cfm?abstract_id=602222) (accessed on 6 May 2024).
- Zulfiker, M.S.; Basak, S. Stock Market Prediction: A Time Series Analysis. ResearchGate. 2021. Available online: <https://www.researchgate.net/publication/354362969> (accessed on 6 May 2024).
- Lakshminarayanan, S.K.; McCrae, J.P. A Comparative Study of SVM and LSTM Deep Learning Algorithms for Stock Market Prediction. CEUR Workshop Proceedings. 2019. Available online: [https://ceur-ws.org/Vol-2563/aics\\_41.pdf](https://ceur-ws.org/Vol-2563/aics_41.pdf) (accessed on 5 June 2024).
- Pashankar, S.S.; Shendage, J.D.; Pawar, J. Machine Learning Techniques for Stock Price Prediction—A Comparative Analysis of Linear Regression, Random Forest, And SVR. *J. Adv. Zool.* **2024**, *45*, 118–127. [CrossRef]
- Chhajer, P.; Shah, M.; Kshirsagar, A. The applications of artificial neural networks, support vector machines, and long-short term memory for stock market prediction. *Decis. Anal. J.* **2022**, *2*, 100015. [CrossRef]
- Shangshang, J. A Comparative Analysis of Traditional and Machine Learning Methods in Forecasting the Stock Markets of China and the US. *Int. J. Adv. Comput. Sci. Appl.* **2024**, *15*, 1–8. Available online: [https://thesai.org/Downloads/Volume15No4/Paper\\_1-A\\_Comparative\\_Analysis\\_of\\_Traditional\\_and\\_Machine\\_Learning.pdf](https://thesai.org/Downloads/Volume15No4/Paper_1-A_Comparative_Analysis_of_Traditional_and_Machine_Learning.pdf) (accessed on 5 August 2024).
- Li, X.; Liang, C.; Ma, F. Forecasting Stock Market Volatility with a Large Number of Predictors: New evidence from the MS-MIDAS-LASSO model. *Ann. Oper. Res.* **2022**, 1–40. [CrossRef]
- Dong, X.; Li, Y.; Rapach, D.E.; Zhou, G. Anomalies and the expected market return. *J. Financ.* **2022**, *77*, 639–681. [CrossRef]
- Sarainmaa, O. Swing Trading the S&P500 Index with Technical Analysis and Machine Learning Methods with Responsible Way. Master's Thesis, Abo Akademi University, Turku, Finland, 2024. Available online: <https://www.doria.fi/handle/10024/189661> (accessed on 6 August 2024).
- Badhe, T.; Borde, J.; Thakur, V.; Waghmare, B.; Chaudhari, A. Comparison of different machine learning methods to detect fake news. In *Innovations in Bio-Inspired Computing and Applications*; Abraham, A., Ed.; IBICA 2021. Lecture Notes in Networks and Systems; Springer: Cham, Switzerland, 2022; p. 419. [CrossRef]
- Cormen, T.H.; Leiserson, C.E.; Rivest, R.L.; Stein, C. *Introduction to Algorithms*; MIT Press: Cambridge, MA, USA, 2022.
- Edward, S. Tesla Stock Close Price Prediction Using KNNR, DTR, SVR, and RFR. *J. Hum. Earth Future* **2022**, *3*, 403–422. [CrossRef]

16. Roy, D.K.; Sarkar, T.K.; Kamar, S.S.A.; Goswami, T.; Muktadir, M.A.; Al-Ghobari, H.M.; Alataway, A.; Dewidar, A.Z.; El-Shafei, A.A.; Mattar, M.A. Daily Prediction and Multi-Step Forward Forecasting of Reference Evapotranspiration Using LSTM and Bi-LSTM Models. *Agronomy* **2022**, *12*, 594. [CrossRef]
17. Vikas, K.; Kotgire, K.; Reddy, B.; Teja, A.; Reddy, H.; Salvadi, S. An Integrated Approach Towards Stock Price Prediction using LSTM Algorithm. In Proceedings of the 2022 International Conference on Edge Computing and Applications (ICECAA), Tamilnadu, India, 13–15 October 2022; pp. 1696–1699. [CrossRef]
18. Akinjole, A.; Shobayo, O.; Popoola, J.; Okoyeigbo, O.; Ogunleye, B. Ensemble-Based Machine Learning Algorithm for Loan Default Risk Prediction. *Mathematics* **2024**, *12*, 3423. [CrossRef]
19. Benidis, K.; Rangapuram, S.S.; Flunkert, V.; Wang, Y.; Maddix, D.; Turkmen, C.; Januschowski, T. Deep learning for time series forecasting: Tutorial and literature survey. *ACM Comput. Surv.* **2022**, *55*, 1–36. [CrossRef]
20. Mehmood, F.; Ahmad, S.; Whangbo, T.K. An Efficient Optimization Technique for Training Deep Neural Networks. *Mathematics* **2023**, *11*, 1360. [CrossRef]
21. Ghadimi, N.; Akbarimajd, A.; Shayeghi, H.; Abedinia, O. Improving time series forecasting using LSTM and attention models. *J. Ambient. Intell. Humaniz. Comput.* **2022**, *13*, 673–691. [CrossRef]
22. Naidu, G.; Zuva, T.; Sibanda, E.M. A review of evaluation metrics in machine learning algorithms. In *Artificial Intelligence Application in Networks and Systems*; Silhavy, R., Silhavy, P., Eds.; CSOC 2023. Lecture Notes in Networks and Systems; Springer: Cham, Switzerland, 2023; Volume 724, pp. 19–32. [CrossRef]
23. Shobayo, O.; Adeyemi-Longe, S.; Popoola, O.; Ogunleye, B. Innovative Sentiment Analysis and Prediction of Stock Price Using FinBERT, GPT-4 and Logistic Regression: A Data-Driven Approach. *Big Data Cogn. Comput.* **2024**, *8*, 143. [CrossRef]
24. Espiritu Pera, J.A.; Ibañez Diaz, A.O.; García López, Y.J.; Taquíá Gutiérrez, J.A. Prediction of Peruvian companies' stock prices using machine learning. In Proceedings of the First Australian International Conference on Industrial Engineering and Operations Management, Sydney, Australia, 20–22 December 2022; IEOM Society International: Southfield, Michigan, 2022; pp. 20–21.
25. Tawakuli, A.; Havers, B.; Gulisano, V.; Kaiser, D. Survey: Time-series data preprocessing: A survey and an empirical analysis. *J. Eng. Res.* **2024**. [CrossRef]
26. Ibrahim, K.S.M.H.; Huang, Y.F.; Ahmed, A.N.; Koo, C.H.; El-Shafie, A. A review of the hybrid artificial intelligence and optimization modelling of hydrological streamflow forecasting. *Alex. Eng. J.* **2022**, *61*, 279–303. [CrossRef]
27. López-González, J.; Peña, D.; Zamar, R. Detecting and handling outliers in financial time series: Methods and applications. *J. Financ. Econom.* **2023**, *21*, 345–369.
28. Zhu, X.; Zhang, Y.; Li, X. The impact of non-stationarity on machine learning models in financial time series. *Quant. Financ.* **2024**, *24*, 95–110.
29. Lewis, C.D. *Industrial and Business Forecasting Methods: A Practical Guide to Exponential Smoothing and Curve Fitting*; Butterworth Scientific: London, UK; Boston, MA, USA, 1982.
30. Song, R.; Shu, M.; Zhu, W. The 2020 global stock market crash: Endogenous or exogenous? *Phys. A Stat. Mech. Appl.* **2022**, *585*, 126425. [CrossRef]
31. Salem, F.M.; Salem, F.M. Recurrent neural networks (RNN): From simple to gated architectures. In *Recurrent Neural Networks*; Springer: Cham, Switzerland, 2022; pp. 43–67.
32. Moodi, F.; Jahangard-Rafsanjani, A.; Zarifzadeh, S. Feature selection and regression methods for stock price prediction using technical indicators. *arXiv* **2023**, arXiv:2310.09903.
33. Gupta, A.; Kapil, D.; Jain, A.; Negi, H.S. Using neural network for financial forecasting. In Proceedings of the 2024 5th International Conference for Emerging Technology (INCET), Belgaum, India, 24–26 May 2024; pp. 1–6.
34. Swamy, S.R.; Rajgoli, S.R.; Hegde, T. Stock market prediction with machine learning: A comprehensive review. *Indiana J. Multidiscip. Res.* **2024**, *4*, 265–271. [CrossRef]
35. Zhang, Y.; Zhong, W.; Li, Y.; Wen, L. A deep learning prediction model of DenseNet-LSTM for concrete gravity dam deformation based on feature selection. *Eng. Struct.* **2023**, *295*, 116827. [CrossRef]
36. Mba, J.C. Assessing portfolio vulnerability to systemic risk: A vine copula and APARCH-DCC approach. *Financ. Innov.* **2024**, *10*, 20. [CrossRef]
37. Janardhan, N.; Kumares, N. Enhancing the early prediction of depression among adolescent students using dynamic ensemble selection of classifiers approach based on speech recordings. *Int. J. Early Child. Spec. Educ.* **2022**, *14*, 1–21. [CrossRef]
38. Lønning, K.; Caan, M.W.; Nowee, M.E.; Sonke, J.J. Dynamic recurrent inference machines for accelerated MRI-guided radiotherapy of the liver. *Comput. Med. Imaging Graph.* **2024**, *113*, 102348. [CrossRef]
39. Muralidhar, K.S.V. Demystifying R-Squared and Adjusted R-Squared. 2023. Available online: <https://builtin.com/data-science/adjusted-r-squared> (accessed on 7 June 2024).
40. Liu, Y. Stock prediction using LSTM and GRU. In Proceedings of the 2022 6th Annual International Conference on Data Science and Business Analytics (ICDSBA), Changsha, China, 14–18 October 2022; pp. 1–6. [CrossRef]

41. Gheisari, M.; Ebrahimzadeh, F.; Rahimi, M.; Moazzamigodarzi, M.; Liu, Y.; Dutta Pramanik, P.K.; Kosari, S. Deep learning: Applications, architectures, models, tools, and frameworks: A comprehensive survey. *CAAI Trans. Intell. Technol.* **2023**, *8*, 581–606. [[CrossRef](#)]
42. Mohr, F.; van Rijn, J.N. Learning curves for decision making in supervised machine learning: A survey. *arXiv* **2022**, arXiv:2201.12150. [[CrossRef](#)]
43. Li, C.; Song, Y. Applying LSTM model to predict the Japanese stock market with multivariate data. *J. Comput.* **2024**, *35*, 27–38.
44. Dhafer, A.H.; Mat Nor, F.; Alkaws, G.; Al-Othmani, A.Z.; Ridzwan Shah, N.; Alshanbari, H.M.; Baashar, Y. Empirical analysis for stock price prediction using NARX model with exogenous technical indicators. *Comput. Intell. Neurosci.* **2022**, *2022*, 9208640. [[CrossRef](#)]
45. Poernamawatie, F.; Susipta, I.N.; Winarno, D. Sharia Bank of Indonesia stock price prediction using long short-term memory. *J. Econ. Financ. Manag. Sci. (JEFMS)* **2024**, *7*, 4777. [[CrossRef](#)]

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.