A COMPARATIVE ANALYSIS OF STOCK SERIES PREDICTION OF APPLE AND GOOGLE DATASETS USING DEEP LEARNING

Supported by tetfund

Oladunjoye John Abiodun[1] and Johnson Miracle Omoware[2]
[1]Computer Science Department, Federal University Wukari, Taraba State, Nigeria.
[2]Computer Science Department, Federal University Wukari, Taraba State, Nigeria.
E-mail: oladunjoye.abbey@yahoo.com[1] and miracleojohnson@gmail.com[2]

Received: June 18, 2023 **Accepted:** August 13, 2023

Abstract

"A Comparative Analysis of Stock Series Prediction using Deep Learning" is a research journal that focuses on the application of deep learning techniques to predict stock market trends. The work presents a comparative study of several deep learning algorithms, including Long Short-Term Memory (LSTM), Support Vector Machines (SVM), Random Forests (RF), Convolutional Neural Networks (CNN), Deep Neural Networks (DAN), Artificial Neural Network (ANN), and Recurrent Neural Network (RNN) to analyze their effectiveness in predicting stock prices. This research provides insights into the potential of deep learning techniques for predicting stock market trends and helps investors and financial analysts make informed decisions. The experimental results indicated that the RNN model outperformed other models based on its strong predictive capabilities and suitability for predicting Google stock, with an accuracy of 87.32% and the top-performing models for Apple stock was the hybridization of Convolutional Neural Network and Long Short-Term Memory (C-LSTM) with an outstanding accuracy of 99.73%.

Keywords: Time series forecasting, Stock price, Apple Stock price, Google Stock price, prediction, genetic algorithm, LSTM, RNN, SVM, MLP, GA, ANN, neural network, machine learning

Introduction

Background of the Study

In the 1600s the Dutch East India Company employed hundreds of ships to trade gold, spices, and silks around the globe, but running this massive operation became difficult and costly (MacLeod, 1986). To fund this expensive budget, the company turned to private citizens, individuals who could invest money to support the trip in exchange for the company shares of the ship's profits. This practice allows the company to afford great sales and increase profits for both themselves and their serving investors. Selling these shares in coffee houses and shipping ports across the continent, the Dutch East India Company unknowingly invented the world's first stock market (Liu, 2007).

The modern stock market is significantly more complicated than its original incarnation. As one of the most fascinating inventions, the financial markets have had a major impact on the country's economy (Hiransha, Gopalakrishnan, Menon, and Soman, 2018). Stock markets are seen as remote places where shares and stocks of a firm are traded not minding the physical location of an individual. The stock market continues to have a highly fluctuating and non-linear time data series due to the pressure of demand and supply by both traders and investors (Hiransha *et al.*, 2018).

The stock exchange has been an issue of concern to both investors and stock exchangers at large, as prices tend to change and vary consistently. The buyer tries to buy the stock at the lowest possible price, while the seller tries to sell it at the highest possible price (Kimberly, 2021).

The supply and demand in stock markets are influenced by many factors. Companies are under the unavoidable influence of market forces such as the fluctuating price of materials, changes in production technology, and the shifting cost of labour. Investors might be worried about changes in leadership, changes in government, and even bad publicity or larger factors like new laws and trade policies. And of course, plenty of investors are simply ready to sell valuable stocks and pursuit personal interest. All these variables cause day-to-day noise in the market which can make the company appear more or less successful. And in the stock market appearing to lose value, often leads to losing investors and in return losing actual value (Pedersen, 2019). Furthermore, recent market volatility has raised significant issues for the prediction of economic and financial time series (Siami-Namini, S., Tavakoli, N., & Namin, A. S, 2018). Therefore, while utilizing different types of prediction methods, and more specifically prediction using regression analysis as they have certain limits in applications, it is vital to evaluate the accuracy of forecasts (Siami-Namini *et al.*, 2018).

Stock market prediction is the act of trying to determine the future value of company stock or other financial instruments traded on a financial exchange. The successful prediction of a stock's future price will maximize investors' gains (Hegazy, Osman & Soliman, Omar S. & Abdul Salam, Mustafa., 2013). In light of the increasing volatility of the financial markets and the globalization of capital flows, greater prediction accuracy is the essential component for better financial decision-making (Guresen *et al.*, 2011).

However, experts are constantly building tools to increase their chances of success in this highly unpredictable system (Siami-Namini *et al.*, 2018). But the stock market is not just for the rich and powerful. With free access to the internet, everyday investors can buy stocks in many of the same ways as large investors would, and as more people educate themselves about this complex system, they too can trade stocks, support the businesses they believe in, and pursue their financial goals.

This paper compares different algorithms and models that are employed in the determination of stock series price prediction and also tends to provide the best model with respect to their performance in reducing error rates.

Literature Review

The main purpose of the stock market prediction is to predict the price of stocks with respect to time either downward or upward for profit maximization. Below are some literature reviews that discuss various techniques that have been proposed for predicting stock market trends.

Siami-Namini *et al.,* (2018) conduct a comparison of ARIMA and LSTM in forecasting time series. The study aimed at determining which method produced the most accurate predictions with the lowest forecast errors. While traditional techniques like AR, MA, SES, and ARIMA have been used for forecasting, the authors noted that deep learning methods like LSTM can identify patterns and structures in complex, nonlinear time series data. The authors collected monthly financial time series data from Yahoo Finance from Jan 1985 to Aug 2018, which included N225, IXIC, HIS, GSPC, and DJ indices. The two techniques were applied to the financial data and the results showed that LSTM outperformed ARIMA, improving predictions by an average of 85%. The study also found that changing the number of epochs did not improve the results.

Sezer, O. B., Ozbayoglu, M., & Dogdu, E. (2017) introduced a novel approach for stock trading that utilizes a deep neural network and evolutionary optimized technical analysis parameters. The objective of their study was to propose a method that incorporates a genetic algorithm and a deep Multilayer Perceptron (MLP) to make buy-sell-hold predictions. To determine the buy and sell points for stocks, Sezer *et al.,* (2017) employed RSI values of the stock prices. Additionally, they utilized Apache Spark and Spark MLlib library for big data analytics. The model was validated using Dow 30 stocks, with each Dow stock being trained separately using daily close prices between 1996-2016 and tested between 2007-2016. According to the study, the proposed trading system produced comparable or better results compared to Buy & Hold and other trading systems for a wide range of stocks, even for relatively longer periods.

Hegazy *et al.,* (2013) developed a machine-learning model for stock market prediction by combining the PSO and LS-SVM models. The LS-SVM model's performance depends on the values of free parameters such as C, ϵ, and γ. The authors used PSO to optimize the LS-SVM model by finding the best parameter combination. They tested the proposed algorithm on the S&P 500 stock dataset from Jan 2009 to Jan 2012 and compared it with the LM algorithm. The PSO-LS-SVM model outperformed the single LS-SVM and achieved the lowest error value compared to the ANN-BP algorithm. The PSO algorithm helped to overcome over-fitting problems and local minima issues and improved prediction accuracy. The proposed model is easily tunable and has the potential in optimizing LS-SVM for stock market prediction, especially during fluctuations in the stock sector.

Nabipour, M., Nayyeri, P., Jabani, H., Shahab, S., & Mousavi, A. (2020) conducted a comparative analysis of machine learning and deep learning algorithms for predicting stock market trends using continuous and binary data. They aimed to make short-term predictions for the emerging Iranian stock market using data from November 2009 to November 2019 (10 years) for four stock market groups: Diversified Financials, Petroleum, Non-metallic minerals, and Basic metals. The study focused on predicting the future values of these groups, which are essential for investors. The authors employed tree-based models (Decision Tree, Bagging, Random Forest, AdaBoost, Gradient Boosting, and XGBoost) and neural networks (ANN, RNN, and LSTM) as regression problems to forecast the values of the four stock market groups for 1, 2, 5, 10, 15, 20, and 30 days ahead. Historical records of the groups for ten years were used for data collection. The results were evaluated using four metrics for each technique. The LSTM algorithm outperformed other techniques and showed the highest model fitting ability, indicating more accurate results. In conclusion, both tree-based and deep learning algorithms have significant potential in regression problems to predict future values of the Tehran stock exchange.

Hiransha *et al.,* (2018) proposed NSE stock market prediction using deep-learning models. Their study involved using four types of deep learning architectures, namely Multilayer Perceptron (MLP), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), and Convolutional Neural Network (CNN), to forecast the stock prices of different companies based on historical prices. Specifically, they used these models to predict the stock prices of MARUTI, HCL, and AXIS BANK from the NSE stock market, as well as the stock prices of BANK OF AMERICA (BAC) and CHESAPEAKE ENERGY (CHK) from the NYSE. The authors gathered their training dataset from the period of 1 JAN 1996 TO 2015 June 30, containing the closing price of 4,861 days. Results from their research indicate that the models were able to identify patterns existing in both stock markets. Additionally, the study showed that CNN outperformed the other models and that the neural networks performed better than the existing linear model (ARIMA).

Nayak, A., Pai, M. M., & Pai, R. M. (2016) put forward prediction models for the Indian stock market intending to utilize historical data and social media data to build models that could forecast stock trend movements, whether they would go up or down. The study produced two models that relied on supervised machine-learning algorithms. The first model focused on daily predictions by considering both sentiment and historical data. The supervised machine learning algorithms were used to achieve up to 70% accuracy on this model. The second model was the monthly prediction model, which aimed to assess the similarity between any two months' trends. The findings revealed that there was little correlation between the trends of different months. The Decision Boosted Tree algorithm was found to perform better than Support Vector Machine and Logistic Regression algorithms on the considered dataset.

Xiao, D., & Su, J. (2022) conducted a study titled "Research on Stock Price Time Series Prediction Based on Deep Learning and Autoregressive Integrated Moving Average". The authors attempted to predict time-series data of stocks by applying traditional and machine learning models to forecast linear and non-linear problems, respectively. They collected stock samples from the New York Stock Exchange that occurred between 2010 and 2019. The study utilized the ARIMA (autoregressive integrated moving average) model and LSTM (long short-term memory) neural network model to train and predict the stock price and stock price sub-correlation. Several indicators were used to evaluate the proposed model, and the experiment results showed that

the ARIMA and LSTM models accurately predicted the stock price and stock price correlation. The LSTM model outperformed the ARIMA model in prediction. Additionally, the ensemble model of ARIMA and LSTM significantly outperformed other benchmark methods. Consequently, the proposed method can provide theoretical support and reference for investors interested in stock trading in the Chinese stock market.

Shen, S., Jiang, H., & Zhang, T. (2012) conducted research using machine learning algorithms to forecast stock market trends. They developed a new prediction algorithm that takes into account the temporal correlation among global stock markets and various financial products and used Support Vector Machines (SVM) to predict the next-day stock trend. The dataset used in the study included NASDAQ, DJIA, S&P 500, Nikkei 225, Hang Seng index, FTSE100, DAX, and ASX. The results of the study showed that the algorithm had a prediction accuracy of 74.4% in NASDAQ, 76% in S&P500, and 77.6% in DJIA. The researchers also tested the algorithm with different regression algorithms to track the actual increment in the markets. Finally, the researchers established a simple trading model to evaluate the performance of the proposed prediction algorithm against other benchmarks.

Moghar & Hamiche, M. (2020) conducted a study on stock market prediction using a Long-Short Term Memory (LSTM) model in combination with Recurrent Neural Networks (RNN). They aimed to develop a model that could accurately predict future stock market values and determine how many epochs were necessary to improve the model's precision. To achieve this, they gathered data from Yahoo Finance, specifically the daily opening prices of two NYSE stocks (GOOGL and NKE) during specific periods. The training process involved using mean squared error optimization and varying the number of epochs (12, 25, 50, and 100) for the training data. The testing results indicated that the adopted model successfully traced the evolution of opening prices for both assets.

Khedmati, M., Seifi, F., & Azizi, M. J. (2020) worked on time series forecasting of bitcoin price based on autoregressive integrated moving average and machine learning approaches. The authors presented several forecasting models based on ARIMA and ML methods, including Kriging, Artificial Neural Networks (ANNs), Bayesian model, Support Vector Machines (SVMs), and Random Forest (RF). Some of the models were univariate, while others were multivariate and incorporated the maximum, minimum, and opening daily prices of Bitcoin. The authors tested their proposed models on Bitcoin price data from December 18, 2019, to March 1, 2020, and evaluated their performance using RMSE and MAPE measures, which were compared using the Diebold-Mariano statistical test. The results showed that SVM had the best performance among all the models, according to both RMSE and MAPE measures. Moreover, the study found that ARIMA and Bayesian approaches outperformed other univariate models, providing smaller values for RMSE and MAPE.

Guresen *et al.*, (2011) investigated the use of artificial neural network (ANN) models for predicting stock market indices. Their study focused on evaluating the performance of the most recent ANN models in forecasting time series used in market values and assessing the effectiveness of dynamic and efficient neural network models in stock market prediction. The authors analyzed three different models: multi-layer perceptron (MLP), dynamic artificial neural network (DAN2), and hybrid neural networks that used generalized autoregressive conditional heteroscedasticity (GARCH) to extract new input variables. They compared each model's performance from two viewpoints: Mean Square Error (MSE) and Mean Absolute Deviation (MAD) using real exchange daily rate values of the NASDAQ Stock Exchange index. The results showed that the classical ANN model MLP outperformed DAN2 and GARCH-MLP models by a small margin.

Models and Methodology

The paper tends to compare different algorithms and models that are employed in the determination of both Apple and Google stock series price prediction. The below methodology comprises three steps which involve data preprocessing, application of the model, and evaluation metrics.

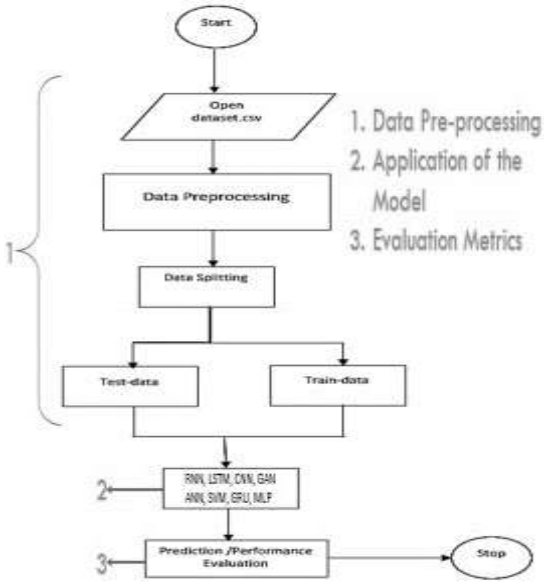


Figure 3.1: Methodology

Data Preprocessing

Data preprocessing involves cleaning and transforming the raw data to make it suitable for analysis and modeling. The following steps are typically involved in data preprocessing for stock series prediction:

a) Data Cleaning: This involves handling missing values, outliers, and inconsistencies in the data. Missing values can be filled using various techniques like interpolation or mean imputation. Outliers may be removed or adjusted based on domain knowledge or statistical methods.

b) Feature Selection/Extraction: Identifying relevant features or variables from the dataset that can contribute to the prediction task is essential. Techniques like correlation analysis, principal component analysis (PCA), or feature importance methods can help select the most informative features.

c) Data Normalization/Scaling: Stock data often have different scales and ranges. Normalizing or scaling the data helps bring all the variables to a common scale,

enabling better modeling and analysis. Common normalization techniques include min-max scaling or standardization.

d) Handling Time-Series Data: Stock data is time-series data, where the temporal order is important. Time-series techniques, such as lagging or differencing, can be applied to capture patterns and trends in the data.

Application of the Model

Once the data preprocessing is completed, various predictive modeling techniques can be applied to forecast stock prices or trends. Some commonly used methods include:

a) Statistical Models: Statistical models like autoregressive integrated moving average (ARIMA), autoregressive conditional heteroscedasticity (ARCH), or generalized autoregressive conditional heteroscedasticity (GARCH) are widely used for stock series prediction. These models leverage historical price patterns and statistical properties to forecast future prices.

b) Machine Learning Models: Machine learning algorithms, such as linear regression, support vector machines (SVM), random forests, or neural networks, can be applied to predict stock prices. These models can capture complex patterns and relationships in the data, considering both historical prices and relevant features.

c) Deep Learning Models: Deep learning architectures like recurrent neural networks (RNNs) or long short-term memory (LSTM) networks are specifically designed for sequential data and have shown promising results in stock series prediction. These models can capture long-term dependencies and temporal patterns in the data.

Evaluation Metrics

To assess the performance of the stock series prediction models, various evaluation metrics can be used. The commonly used metrics include Accuracy, Precision, Mean Absolute Percentage Error (MAPE), Root Mean Squared Error (RMSE), Relative Absolute Error (RAE), Root Relative Squared Error (RRSE), and Mean Squared Error (MSE) which are later discussed in this paper. These evaluation metrics help assess the performance of the stock series prediction models and compare different approaches to identify the most accurate and reliable model for forecasting stock prices.

Algorithms Description

This segment mainly concentrated on the build models and how predictions are made.

1) Recurrent Neural Network (RNN): An RNN, or recurrent neural network, is a specific type of neural network designed to predict the next step in a sequence of observations based on previous observations in the sequence. The core concept behind RNNs is to utilize sequential data and use past observations to make predictions about future trends. Hidden layers within an RNN function as internal storage, capturing information from earlier stages of sequential data. RNNs are referred to as "recurrent" because they perform the same task for each element in the sequence, utilizing information from previous stages to predict future, unseen sequential data (Siami-Namini *et al.*, 2022).

"The Recurrent Neural Network (RNN) consists of states, where each state is represented by a vector denoted as h_t. These vectors encode the hidden units at a specific time step, as shown in Figure 3.2. The sequence of states in an RNN is influenced by both the previous states and the input at each time step, creating a dependency between them. Mathematically, the recurrent neural network can be represented as:

$$h_t = \mathcal{F}(h_{t-1}, X_t, \theta) \dots\dots\dots\dots\dots\dots\dots\dots (3.1)$$

When the state of the system has been inferred, the output at any given time step can be determined as a function of that inferred state. The output is then mathematically represented as: $O_t = g(h_t;\ \theta_*,) \dots\dots\dots\dots\dots\dots\dots (3.2)$

where, θ_* is a different set of parameters, specifically trained for the output variable. Note that the functions $\mathcal{F}$ and g are applied at each time step. Collectively, these functions, applied at each time step, are known as an **RNN cell**, a specialized repeatable unit in the network.

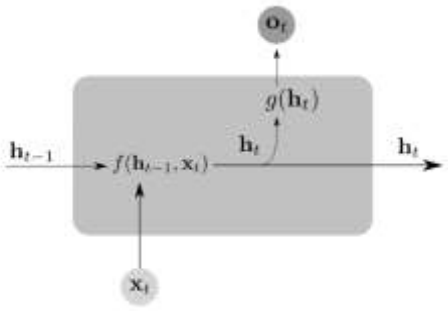


Figure 3.2: Recurrent function of the recurrent network

Table 3.1 RNN-Algorithm

Recurrent Neural Network Algorithm
The value of x defines the training instance *y defines an output instance* *P_t is the target value at time step t:* *Repeat till the stopping criterion is met:* *Set all h to zero.* *Repeat for t = 0 to n-x* *Forward propagate the network over the unfolded network for x time steps to compute all h and y.* *Compute the error as: $e = y_{t+x} - P_{t+x}$* *Backpropagate the error across the unfolded network and update the weights.*

2) Long Short-Term Memory (LSTM): LSTM is a type of recurrent neural network that has additional capabilities to remember the sequence of data. This is achieved through the use of gates and a memory line within each LSTM cell. The LSTM is composed of a set of cells, which act as system modules that capture and store the data streams (Dua *et al.*, 2020). These cells are connected in a transport line, allowing data to be passed from past cells to present ones. The use of gates in each cell allows for data to be filtered, disposed of, or added to the next cells (Ludwig, 2019). Each LSTM cell contains three types of gates that control the state of the cell:

- Forget Gate outputs a number between 0 and 1, where 1 shows "completely keep this"; whereas, 0 implies "completely ignore this."

- Memory Gate chooses which new data need to be stored in the cell. First, a sigmoid layer, called the "input door layer" chooses which values will be modified. Next, a tanh layer makes a vector of new candidate values that could be added to the state.
- Output Gate decides what will be yielded out of each cell. The yielded value will be based on the cell state along with the filtered and newly added data.

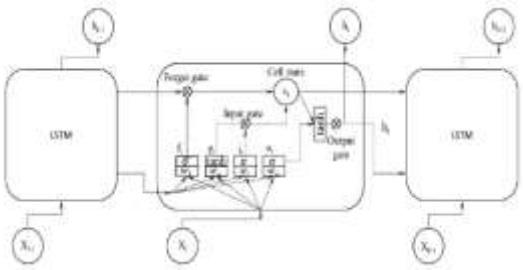


Figure 3.3: LSTM model

By observing the LSTM diagram presented above, one can discern the forward propagation process. The forward propagation involves four distinct networks, each of which is activated by either the sigmoid or the tanh function. These networks have their own unique set of parameters and are commonly referred to as gates. Their primary purpose is to transform the cell state from one time step to the next. The LSTM algorithm can be shown by the following equations:

$$i_t = \sigma_s(w_{hi}\ h_{t-1} + w_i X_t + b_i) \dots\dots\dots (3.3)$$

$$f_t = \sigma_s\left(w_{hf}\ h_{t-1} + w_f X_t + b_f\right) \dots\dots\dots (3.4)$$

$$o_t = \sigma_s(w_{ho}\ h_{t-1} + w_o X_t + b_o) \dots\dots\dots (3.5)$$

$$g_t = \tanh\left(w_{hg}\ h_{t-1} + w_g X_t + b_g\right) \dots\dots\dots (3.6)$$

Cell State:

$$c_t = f_t^o\ c_{t-1} + \ i_t^o g_t \ \dots\dots\dots\dots\dots\dots\dots (3.7)$$

Output vector:

$$h_t = Y_t = o_t^o \tanh(c^t) \dots\dots\dots\dots\dots\dots (3.8)$$

From the equations 3.3 – 3.8, h_t can be considered as the short-term state whereas c_t as the long-term state. The variables i, f, o, g denote the input, forget, output, and the new candidate gate layers. Mathematically, the LSTM recurrent networks first use the previous hidden state for gaining the fixed-dimensional proxy of the sequential input to estimate the conditional probability, and then define the probability of its corresponding output sequence with the initial state, which is set to the representation of the input sequence (Siami-Namini *et al.*, 2022).

Table 3.2 The developed rolling LSTM algorithm.

Rolling LSTM
Inputs: Time series Outputs: RMSE of the forecasted data # Split data into: # 70\% training and 30\% testing data 1. size ← length(series) * 0.70 2. train ← series[0...size] 3. test ← series[size...length(size)] 4. set random.seed(7) # Fit an LSTM model to training data Procedure fit_lstm(train, epoch, neurons) 5. X ← train 6. y ← train – X 7. model = Sequential() 8. model.add(LSTM(neurons), stateful=True)) 9. model.compile(loss='mean_squared_error', optimizer='adam') 10. for each i in range(epoch) do 11. model.fit(X, y, epochs=1, shuffle=False) 12. model.reset_states() 13.end for return model # Make a one-step forecast Procedure forecast_lstm(model, X) 14. yhat ← model.predict(X) return yhat 15. epoch ← 1 16. neurons ← 4 17. predictions ← empty 18. lstm_model = fit_lstm(train,epoch,neurons) 19. lstm_model.predict(train) 20. for each i in range(length(test)) do 21. # make one-step forecast 22. X ← test[i] 23. yhat ← forecast_lstm(lstm_model, X) 24. # record forecast 25. predictions.append(yhat) 26. expected ← test[i] 27. end for 28. MSE ← mean_squared_error(expected, predictions) 29. Return (RMSE ← sqrt(MSE))

3) Artificial Neural Networks

Artificial Neural Networks (ANNs) are neural networks that can be either single-layered or multi-layered, with full connectivity between the layers. The figure below illustrates an example of an ANN comprising an input layer, an output layer, and two hidden layers. Each node within a layer is connected to every other node in the subsequent layer. By adding more hidden layers, the network can be deepened, allowing for increased complexity and representation capabilities.

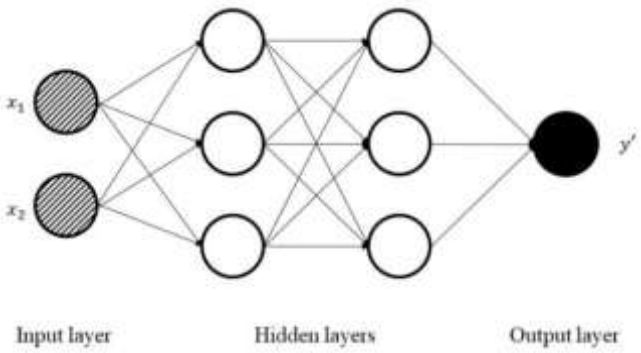


Figure 3.4. Schematic illustration of artificial neural networks

In figure 3.4, each hidden or output node is depicted. To compute the value of a node, it takes the weighted sum of its inputs, which is then added to a bias value. This sum is then passed through an activation function, typically a non-linear function. The resulting output of the node

becomes an input for the subsequent layer's nodes (Ashtiani *et al.*, 2022). The procedure progresses from the input to the output layer, and the final output is determined by performing this process for all nodes in the network. During the learning process, the weights and biases associated with all nodes are adjusted and updated to train the neural network (Lee & Song, 2019).

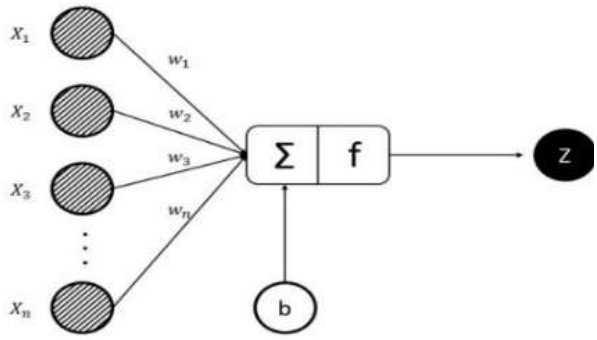


Figure 3.5 Illustrate the relationship between inputs and output for ANN

Figure 3.5 shows the relationship between nodes, weights, and biases. The weighted sum of inputs for a layer passed through a non-linear activation function to another node in the next layer. It can be interpreted as a vector, where X_1, X_2, : : : , and X_n are inputs, w_1, w_2,: : : , and wn are weights respectively, n is the number of the input for the final node, f is activation function and z is the output.

$$Z = f\,(x.w + b) = f\left(\sum_{i=1}^{n} x_i^T w_i + b\right).$$

……..…...(3.9)

4) Multi-Layer Perceptron (MLP)

A Multi-Layer Perceptron (MLP), also known as Feed-Forward Network, exemplifies a straightforward neural network architecture (Meshram *et al.*, 2022). In this type of network, the input neurons are connected to the subsequently hidden layer neurons via a weighted matrix w_{ki}. A Network has three sections of layers input, hidden and output layers Artificial neurons are found in the hidden and output layers of a neural network. Each of these neurons receives inputs from the preceding layer. It is important to note that neurons within the same layer are not connected, but they are connected to neurons in the subsequent layer (Mariet & Sra, 2015). The equation for activation function an i^{th} hidden neuron is given by;

$$h_i = f(u_i) = f\left(\sum_{k=0}^{K} w_{ki} x_k\right)$$

……..….(3.10)

h_i: i^{th} hidden neuron, f(ui) : link function which provides non-linearity between input and hidden layer, *wki* : weight in the (k, i)th entry in a (K X N) weight matrix, xK : K^{th} input value.

$$y_j = f(u'_j) = f\left(\sum_{i=1}^{N} w'_{ij} h_i\right)$$

…….(3.11)

$y\,j$: j^{th} output value.

5) Random Forest

Decision tree learning is a highly popular technique for classification (Sharma & Kumar, 2016). It offers a remarkable balance between efficiency and classification accuracy, rivaling other classification methods. The resulting classification model, often referred to as a decision tree, effectively represents the knowledge acquired from this technique. Additionally, decision trees can also be applied to regression tasks, where they demonstrate exceptional efficiency and yield low error values. Random Forest belongs to the category of ensemble learning algorithms. The decision tree is used by it as the base learner of the ensemble. The fundamental concept of ensemble learning stems from the recognition that a single regression tree may lack sufficient accuracy in predicting the value of a dependent variable (Rashidi *et al.*, 2019). This is primarily due to the inability of a regressor, trained solely on sample data, to effectively distinguish between noise and underlying patterns. To address this limitation, ensemble learning employs a technique called sampling with replacement. By generating multiple data set samples, where each sample contains different instances of the original data, a set of n trees can be learned based on these samples (Nikam, 2015).

Algorithm 1. implementation of Random Forest

Input: training set D, number of trees in the ensemble k

Output: a composite model M^*

1: **for** $i = 1$ to n **do**
2: Create bootstrap sample D_i by sampling D with replacement.
3: Select 3 features randomly.
4: Use D_i and randomly selected three features to derive a regression tree M_i.
5: **end for**
6: return M^*.

6) Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNNs) are a class of deep learning algorithms that are particularly effective in analyzing visual data. CNNs are widely used for tasks such as image classification, object detection, and image recognition. The key characteristic of CNNs is their ability to automatically learn and extract hierarchical patterns or features from input data (Lin *et al.*, 2017). This is achieved through the application of convolutional layers, pooling layers, and fully connected layers. In a CNN, the convolutional layers perform local receptive field operations, where filters (also known as kernels) are convolved with the input data. This operation captures local patterns and spatial relationships in the input, allowing the network to learn features at different levels of abstraction (Wang *et al.*, 2020). The pooling layers

reduce the spatial dimensionality of the features, extracting the most salient information. Common pooling operations include max pooling and average pooling (Lee *et al.*, 2016).

Typically, a CNN accepts a 3-dimensional tensor as input, such as an image with H rows, W columns, and 3 channels representing the RGB colors. However, CNNs can also similarly handle higher-order tensor inputs. The input data undergoes a sequential process, with each step referred to as a layer. These layers can include convolutional layers, pooling layers, normalization layers, fully connected layers, loss layers, and more. Below is an abstract description of the CNN structure.

$$x^1 \longrightarrow \boxed{w^1} \longrightarrow x^2 \longrightarrow \cdots \longrightarrow x^{L-1} \longrightarrow \boxed{w^{L-1}} -$$

……..(3.12)

The equation presented as Equation 5 demonstrates the sequential execution of a CNN through its layers during a forward pass. The input, denoted as x^1, typically represents an image (an order 3 tensor). It undergoes processing in the first layer, represented by the initial box. The parameters associated with the first layer's processing are collectively denoted as a tensor w^1. The resulting output of the first layer is x^2, which serves as the input for the subsequent layer's processing.

7) Support Vector Machines (SVMs)

Support Vector Machines (SVMs) encompass a collection of supervised learning methods used for classification and regression tasks. The classifier variant is known as SVC. The primary objective of SVMs is to determine a decision boundary, typically in the form of vectors, that effectively separates two classes (Bi *et al.*, 2019). This boundary is deliberately positioned to be distant from any point within the dataset, and the support vectors are identified as observation coordinates that reside within a margin, creating a gap. SVMs strive to establish the most optimal boundary, employing a line or hyperplane, to effectively separate the two classes (Ghosh *et al.*, 2019).

$$f(x) = \text{sgn}(\sum_{i=1}^{n} \alpha_i y_i \cdot K(x, x_i) + l$$

……..…(3.13)

Dataset Description

a) Apple Stock Dataset Description

The "Apple Stock Dataset" contains historical financial data of Apple Inc.'s publicly traded shares. It includes date and time stamps, stock prices (closing, opening, high, low), and trading volume, allowing analysis of stock performance (Aasi *et al.*, 2021). Market indices, dividends, and stock splits are included for comparison and context. In addition to stock prices, the dataset likely incorporates information on market indices, such as the S&P 500 or Nasdaq Composite, allowing investors and analysts to compare Apple's stock performance with broader market trends (Shah *et al.*, 2019).

b) Google Stock Dataset Description:

The "Google Stock Dataset" consists of historical financial data of Alphabet Inc.'s publicly traded shares (Jhin *et al.*, 2021). It shares similarities with Apple's dataset, featuring date-time stamps, stock prices, market indices, dividends, and stock splits.

Historical prices

Historical prices are obtained from Yahoo Finance and related daily stock data from the Standard & Poor's 500 (S&P 500) index in Yahoo Finance, and the financial news headline from AAPL stock data (Liu *et al.*, 2018). Each transaction date consists of the open price, close price, low price, high price, adjusted close price, and volume traded on that day. Adjusted close price and close price depicts the close price of the stock on a particular day (Nayak *et al.*, 2016). The sentiment of the tweets and the sentiment of the news are integrated daily. Both tweets and news has been collected for both datasets and the sentiment analysis algorithm is applied to the same.

Performance Evaluation Metrics

Various metrics were used to assess the performance of stock series predictions which includes;

i). Accuracy: is measured as the percentage of the number of correctly predicted instances to the total number of instances present.

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \ldots\ldots (3.14)$$

Precision: This measures the classifier's accuracy. It is the percentage of the number of correctly predicted positive instances divided by the total number of positive instances present.

$$Precision = \frac{TP}{TP + FP} \ldots\ldots\ldots\ldots\ldots\ldots (3.15)$$

F-Score: This defines the harmonic mean of precision and recall. It combines recall and precision metrics to obtain a score.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \ldots\ldots\ldots (3.16)$$

The Root-Mean-Square Error (RMSE): This a measure frequently used for assessing the accuracy of prediction obtained by a model. It measures the differences or residuals between actual and predicted values.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i^f - y_i^{ob})^2}{n}} \ldots\ldots\ldots\ldots (3.17)$$

MAPE (Mean Absolute Percentage Error): define the average of the percentage errors. In essence, MAPE indicates the average difference between the predicted value and the original value.

$$MAPE = \frac{1}{n} \sum \frac{|o_t - p_t|}{|o_t|} * 100 \ldots\ldots\ldots\ldots (3.18)$$

Mean absolute error (MAE) is a measure of the difference between two values. MAE is an average of the difference between the prediction and the actual values.

$$MAE = \frac{1}{n}\sum_{t=1}^{n} * |A_t - F_t| \ldots\ldots (3.19)$$

The Mean Squared Error (MSE) measures the quality of a predictor and its value is always non-negative (values closer to zero are better). The MSE is the second moment of the error (about the origin), and incorporates both the variance of the prediction model and its bias.

$$MSE = \frac{1}{n}\sum_{t=1}^{n} * (A_t - F_t)^2 \ldots\ldots\ldots (3.20)$$

Relative Root Mean Square Error: Root Mean Square Error (RMSE) is the standard deviation of the prediction errors in regression work. RMSE is the square root of the average of squared differences between predictions and actual observations.

$$\text{RRMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}\left(\frac{A_t - F_t}{A_t}\right)^2}, \quad \ldots\ldots\ldots (3.21)$$

Experimental Results

Table 4.1 and 4.2 below shows the comparative outcome of the various techniques that were adopted by different researchers in the prediction of both the Apple and Google stock dataset.

Table 4.1 Comparative table for APPLE Dataset

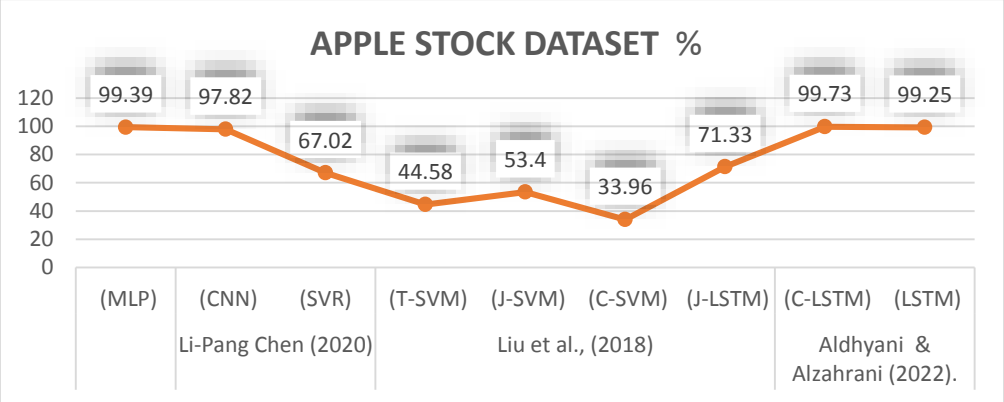
AUTHOR & YEAR	MAPE	RMSE	MSE	RAE	RRSE	F1_Score%
Torres *et al.,* (2019)	0.453 (MLP)	0.606 (MLP)	-	6.33 (MLP)	250 (MLP)	99.39
Li-Pang Chen (2020)	13.75 (LSTM) 2.18 (CNN) 0.67 (SVR)	-	-	-	-	86.25 97.82 67.02
Liu *et al.,* (2018)	-	-	-	-	-	44.58 (T-SVM) 53.40 (J-SVM) 33.96 (C-SVM) 63.04(C-LSTM) 71.33(J-LSTM)
Aldhyani, & Alzahrani (2022).	-	0.00756 (CNN-LSTM) 0.01246 (LSTM)	0.000057(CNN-LSTM) 0.000155(LSTM)	-	-	99.73 99.25

KEY: Mean Absolute Percentage Error (MAPE), Root Mean Squared Error (RMSE), Relative Absolute Error (RAE), Root Relative Squared Error (RRSE), Mean Squared Error (MSE).

Table 4.1.1 Summary of Apple Stock Dataset

AUTHOR & YEAR	MODELS	Accuracy %
Torres *et al.,* (2019)	(MLP)	99.39
Li-Pang Chen (2020)	(CNN)	97.82
	(SVR)	67.02
Liu *et al.,* (2018)	(T-SVM)	44.58
	(J-SVM)	53.4
	(C-SVM)	33.96
	(J-LSTM)	71.33
Aldhyani & Alzahrani (2022).	(C-LSTM)	99.73
	(LSTM)	99.25

KEY: Random Forest (RF), Multi-Layer Perception (MLP), Long Short-Term Memory (LSTM), Convolutional Neural Networks (CNN), Support Vector Regression (SVR), Gated Recurrent Unit (GRU), T-SVM (Input of Tf-idf algorithm and Support Vector Machine), J-SVM (Input of Joint learning and Support Vector Machine), C-SVM (Input of CNN and Support Vector Machine) C-LSTM (Input of CNN and LSTM) J-LSTM (Input of Joint learning and LSTM)



The table and graph above represent the accuracy percentages of different models used for the prediction of Apple stock data in various research papers. Based on the table, the authors have employed various machine-learning techniques to predict Apple stock data. The accuracy percentages allow us to compare the performance of these models and also show that some models achieve very high accuracy. The experimental result indicated that the top-performing model was the hybridization of Convolutional Neural Network and Long Short-Term Memory (C-LSTM) from Aldhyani & Alzahrani, (2022) with an accuracy of 99.73%, followed by the Multi-Layer Perception (MLP) model from Torres *et al.*, 2018 with an accuracy of 99.39%. The CNN model from Li-Pang Chen also shows promising performance, though not as high as the top performers. On the other hand, the hybridization of Tf-idf algorithm and Support Vector Machine (T-SVM), Convolutional Neural Network, and Support Vector Machine (C-SVM), from Liu *et al.*, (2018) shows a lower performance accuracy of 33.96% and 44.58% respectively.

Table 4.2 Comparative table for GOOLE Dataset

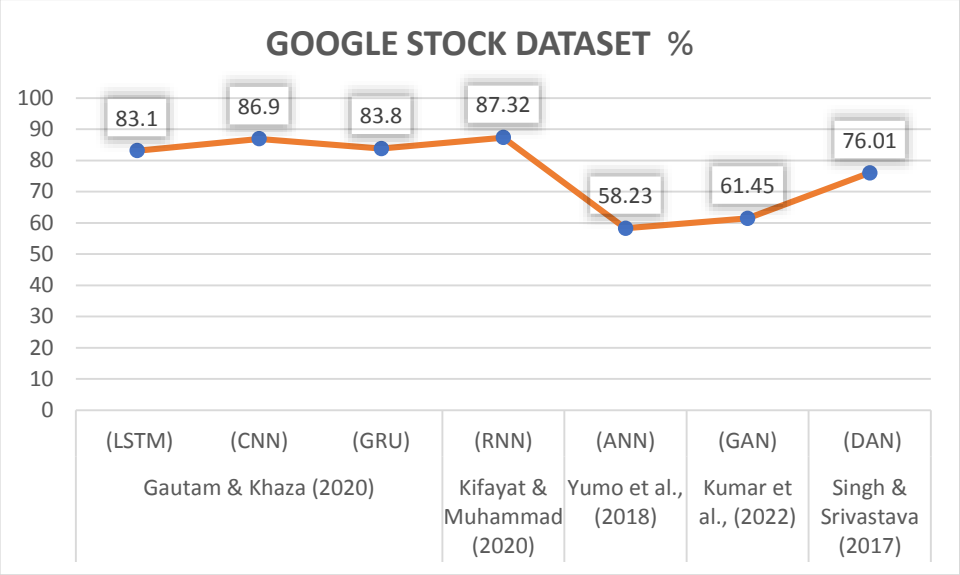
AUTHOR & YEAR	MAPE	RMSE	MSE	RAE	RRSE	F1_Score %
Gautam and Khaza (2020)	-	16.9 (LSTM) 13.1 (CNN) 16.2 (GRU)	-	-	-	83.10 86.90 83.80
Kifayat & Muhammad (2020)	-	12.68 (RNN)	-	-	-	87.32
Yumo *et al.*, (2018)	-	-	-	-	-	58.23 (ANN)
Kumar *et al.*, (2022)	-	-	-	-	-	61.45 (GAN)
Singh & Srivastava (2017)	-	-	-	-	-	76.0 (DAN)

KEY: Mean Absolute Percentage Error (MAPE), Root Mean Squared Error (RMSE), Relative Absolute Error (RAE), Root Relative Squared Error (RRSE), Mean Squared Error (MSE)

Table 4.2.1 Summary of Google Stock Dataset

AUTHOR & YEAR	MODELS	Accuracy %
Gautam & Khaza (2020)	(LSTM)	83.1
	(CNN)	86.9
	(GRU)	83.8
Kifayat & Muhammad (2020)	(RNN)	87.32
Yumo *et al.*, (2018)	(ANN)	58.23
Kumar *et al.*, (2022)	(GAN)	61.45
Singh & Srivastava (2017)	(DAN)	76.01

KEY: Random Forest (RF), Multi-Layer Perception (MLP), Long Short-Term Memory (LSTM), Convolutional Neural Networks (CNN), Support Vector Regression (SVR), Gated Recurrent Unit (GRU), Deep Neural Network (DNN), Recurrent Neural Network (RNN).



According to the above table and graph, the best-performing model in predicting Google stock data are the Recurrent Neural Network (RNN) model by Kifayat & Muhammad (2020) with an impressive accuracy of 87.32% making it the top performer among the listed models followed by the Convolutional Neural Network (CNN) model proposed by Gautam & Khaza (2020) having 86.90% accuracy, indicating its strong predictive capabilities for Google stock data. On the other hand, the Artificial Neural Network (ANN) and Generative Adversarial Network (GAN) show a lower performance accuracy of 58.23% and 61.45% respectively.

Conclusions

The stock market's behavior is inherently unpredictable and influenced by a multitude of factors. These variables encompass global economic conditions, geopolitical events, individual company performance, investor sentiments, and financial disclosures. Among these factors, a crucial element in evaluating stock price fluctuations is a company's profitability. However, despite its significance, accurately forecasting market movements remains challenging for investors. The comparative analysis of various machine learning models used for predicting Apple and Google stock provides valuable insights into their performance and predictive capabilities. For predicting Apple stock, two standout models have achieved exceptionally high accuracy percentages. The hybridization of the Convolutional Neural Network and Long Short-Term Memory (C-LSTM) model, as proposed by Aldhyani & Alzahrani (2022), demonstrates an impressive accuracy of 99.73% followed by the Multi-Layer Perception (MLP) model from Torres *et al.*, (2018) with an accuracy of 99.39%. while in Google stock prediction, the analysis highlights the effectiveness of the Recurrent Neural Network (RNN) model by Kifayat & Muhammad (2020) which stands out as the top performer, achieving an impressive accuracy of 87.32%. This RNN model showcases its strong predictive capabilities and suitability for predicting Google stock. In conclusion, the hybrid C-LSTM and MLP models demonstrate exceptional accuracy in predicting Apple stock data, while the RNN and CNN models prove remarkably effective for Google stock prediction. However, the hybrid T-SVM and C-SVM models, along with the ANN and GAN models, exhibit lower accuracy levels for both Apple and Google stock prediction. Researchers and practitioners interested in stock data prediction can consider the top-performing models mentioned in the analysis to achieve more accurate and reliable predictions.

Conflict of Interest Statement

On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

Aasi, B., Imtiaz, S. A., Qadeer, H. A., Singarajah, M., & Kashef, R. (2021, April). Stock Price Prediction Using a Multivariate Multistep LSTM: A Sentiment and Public Engagement Analysis Model. In *2021 IEEE International IOT, Electronics, and Mechatronics Conference (IEMTRONICS)* (pp. 1-8). IEEE.

Aldhyani, T. H., & Alzahrani, A. (2022). Framework for predicting and modeling stock market prices based on deep learning algorithms. *Electronics*, *11*(19), 3149.

Ashtiani, F., Geers, A. J., & Aflatouni, F. (2022). An on-chip photonic deep neural network for image classification. *Nature*, *606*(7914), 501-506.

Bi, Q., Goodman, K. E., Kaminsky, J., & Lessler, J. (2019). What is machine learning? A primer for the epidemiologist. *American Journal of Epidemiology*, *188*(12), 2222-2239.

Dua, M., Yadav, R., Mamgai, D., & Brodiya, S. (2020). An improved RNN-LSTM-based novel approach for sheet music generation. *Procedia Computer Science*, *171*, 465-474.

Ghosh, S., Dasgupta, A., & Swetapadma, A. (2019, February). A study on support vector machine-based linear and non-linear pattern classification. In *2019 International Conference on Intelligent Sustainable Systems (ICISS)* (pp. 24-28). IEEE.

Guresen, E., Kayakutlu, G., & Daim, T. U. (2011). Using artificial neural network models in stock market index prediction. *Expert Systems with Applications*, *38*(8), 10389-10397.

Hegazy, Osman & Soliman, Omar S. & Abdul Salam, Mustafa. (2013). A Machine Learning Model

for Stock Market Prediction. *International Journal of Computer Science and Telecommunications*. 4. 17-23.

Hiransha, M., Gopalakrishnan, E. A., Menon, V. K., & Soman, K. P. (2018). NSE stock market prediction using deep-learning models. *Procedia computer science*, *132*, 1351-1362.

Jhin, S. Y., Shin, H., Hong, S., Jo, M., Park, S., Park, N., ... & Jeon, S. (2021, December). Attentive neural controlled differential equations for time-series classification and forecasting. In *2021 IEEE International Conference on Data Mining (ICDM)* (pp. 250-259). IEEE.

Khedmati, M., Seifi, F., & Azizi, M. J. (2020). Time series forecasting of bitcoin price based on autoregressive integrated moving average and machine learning approaches. *International Journal of Engineering*, *33*(7), 1293-1303.

Lee, C. Y., Gallagher, P. W., & Tu, Z. (2016, May). Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree. In *Artificial intelligence and statistics* (pp. 464-472). PMLR.

Lee, H., & Song, J. (2019). Introduction to a convolutional neural network using Keras; an understanding from a statistician. *Communications for Statistical Applications and Methods*, *26*(6), 591-610.

Lin, Y. Z., Nie, Z. H., & Ma, H. W. (2017). Structural damage detection with automatic feature extraction through deep learning. *Computer-Aided Civil and Infrastructure Engineering*, *32*(12), 1025-1046.

Liu, Y. (2007). *The Dutch East India Company's Tea Trade with China: 1757-1781* (Vol. 6). Brill.

Liu, Y., Zeng, Q., Yang, H., & Carrio, A. (2018). Stock price movement prediction from financial news with deep learning and knowledge graph embedding. In *Knowledge Management and Acquisition for Intelligent Systems: 15th Pacific Rim Knowledge Acquisition Workshop, PKAW 2018, Nanjing, China, August 28-29, 2018, Proceedings 15* (pp. 102-113). Springer International Publishing.

Ludwig, S. A. (2019, June). Comparison of time series approaches applied to greenhouse gas analysis: ANFIS, RNN, and LSTM. In *2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)* (pp. 1-6). IEEE.

MacLeod, C. (1986). The 1690s patents boom: invention or stock-jobbing? *Economic History Review*, 549-571.

Mariet, Z., & Sra, S. (2015). Diversity networks: Neural network compression using determinantal point processes. *arXiv preprint arXiv:1511.05077*.

Meshram, S. G., Meshram, C., Pourhosseini, F. A., Hasan, M. A., & Islam, S. (2022). A multi-layer perceptron (MLP)-Firefly algorithm (FFA)-based model for sediment prediction. *Soft Computing*, 1-10.

Moghar, A., & Hamiche, M. (2020). Stock market prediction using LSTM recurrent neural network. *Procedia Computer Science*, *170*, 1168-1173.

Nabipour, M., Nayyeri, P., Jabani, H., Shahab, S., & Mosavi, A. (2020). Predicting stock market trends using machine learning and deep learning algorithms via continuous and binary data; a comparative analysis. *IEEE Access*, *8*, 150199-150212.

Nayak, A., Pai, M. M., & Pai, R. M. (2016). Prediction models for the Indian stock market. *Procedia Computer Science*, *89*,

Nikam, S. S. (2015). A comparative study of classification techniques in data mining algorithms. *Oriental Journal of Computer Science and Technology*, *8*(1), 13-19.

Pedersen, L. H. (2019). *Efficiently inefficient: how smart money invests and market prices are determined*. Princeton University Press.

Rashidi, H. H., Tran, N. K., Betts, E. V., Howell, L. P., & Green, R. (2019). Artificial intelligence and machine learning in pathology: the present landscape of supervised methods. *Academic pathology*, *6*, 2374289519873088.

Sezer, O. B., Ozbayoglu, M., & Dogdu, E. (2017). A deep neural-network-based stock trading system based on evolutionary optimized technical analysis parameters. *Procedia computer science*, *114*, 473-480.

Shah, D., Isah, H., & Zulkernine, F. (2019). Stock market analysis: A review and taxonomy of prediction techniques. *International Journal of Financial Studies*, *7*(2), 26.

Sharma, H., & Kumar, S. (2016). A survey on decision tree algorithms of classification in data mining. *International Journal of Science and Research (IJSR)*, *5*(4), 2094-2097.

Shen, S., Jiang, H., & Zhang, T. (2012). Stock market forecasting using machine learning algorithms. *Department of Electrical Engineering, Stanford University, Stanford, CA*, 1-5.

Siami-Namini, S., Tavakoli, N., & Namin, A. S. (2018, December). A comparison of ARIMA and LSTM in forecasting time series. In *2018 17th IEEE international conference on machine learning and applications (ICMLA)* (pp. 1394-1401). IEEE.

Torres P, Edgar P., et al. "Stock market data prediction using machine learning techniques." *Information Technology and Systems: Proceedings of ICITS 2019*. Springer International Publishing, 2019.

Wang, W., Hu, Y., Zou, T., Liu, H., Wang, J., & Wang, X. (2020). A new image classification approach via improved MobileNet models with local receptive field expansion in shallow layers. *Computational Intelligence and Neuroscience*, *2020*.

Xiao, D., & Su, J. (2022). Research on Stock Price Time Series Prediction Based on Deep Learning and Autoregressive Integrated Moving Average. *Scientific Programming*, *2022*.