Proceedings of the 4th International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/49/20241299

Machine learning and deep learning models for stock price prediction, case study: Google Company

Xiaowei Wang

Lally School of Management, Rensselaer Polytechnic Institute, 110 Eighth Street, Troy 12180, Troy City, United States

wangx56@rpi.edu

Abstract. Stock prediction is crucial for investors to manage risk, diversify portfolios, and plan for long-term financial goals. Companies use predictions to allocate capital wisely, attract investors, and make strategic decisions. Stock prediction is to analyze competitive stock (e.g. Google, Apple and Microsoft), involving parameters of revenue, profit, PE ratio, growth rate, and to predict the future price of the stock. Based on these assumptions, this project objective is to forecast the trend in Google stock prices with four predictive models: linear regression, XGBoost, Long Short-Term Memory Network (LSTM) and Recurrent neural network (RNN). The dataset is Google stock prediction dataset, which involves 17,598 stock information. Among all four models, LSTM method obtains the best final result with 0.0012 of MSE and 0.95115 of R square. LSTM has the lowest MSE, and its R square is similar to that of other models. For future plan, this project may expand more on feature engineering, model techniques and fine-tuning methods.

Keywords: Machine learning, deep learning, stock price forecast, neural network.

1. Introduction

For Stock prediction plays a vital role in the financial market. in the financial market. Its value is evident from three perspectives: the company, customers, and the market [1]. For companies, accurate stock prediction provides decision support, aids in attracting capital, and supports financial planning. For investors and customers, stock prediction forms the basis for investment decisions, wealth management, and retirement planning, helping achieve financial goals. At the market level, stock prediction serves as an economic indicator, aids in risk assessment, and facilitates investor sentiment analysis. In summary, stock prediction plays an indispensable role in shaping company strategies, guiding investment decisions, and understanding market trends and risks, holding significant importance for both the financial market and its participants.

In a quest to gain deeper insights into the intricate world of stock market dynamics, diligent researchers have harnessed the power of contemporary modeling techniques. They embarked on a journey through the vast landscape of financial data, meticulously employing a diverse array of models, including the venerable AR(AutoRegressive) model [2], the time-tested linear regression model [3], the robust tree model [4], and the cutting-edge deep learning model [5][6]. Contrasting the four models for stock predictions reveals their distinctive strengths and characteristics. The AR model, a time series model, relies on past stock prices to predict future values, making it suitable for short-term forecasting.

Linear regression, while simple, assumes a linear relationship between variables and may not capture complex market dynamics effectively. Tree models, such as decision trees and random forests, offer interpretability and handle non-linearity but might struggle with overfitting. Deep learning models, like neural networks, excel in capturing intricate patterns but require extensive data and computing power, making them ideal for long-term predictions. Choosing the right model depends on the specific forecasting horizon and data available. This formidable arsenal of modeling tools enabled them to navigate through oceans of stock datasets, meticulously dissecting and scrutinizing the performance and behavior of numerous companies. Through their tireless efforts, a treasure trove of valuable insights emerged, shedding light on the complex interplay of factors that govern the financial markets.

This article utilizes the dataset of Google Stock Prediction. The dataset contains 14 columns and 1257 Rows. The 14 columns include symbol: - Name of the company (in this case Google), date:- year and date, close:- closing of stock value, high:- highest value of stock at that day, low:- lowest value of stock at that day, open:- opening value of stock at that day, etc. To forecast the trend in Google stock prices, this article chooses four machine learning and deep learning models: linear regression, XGBoost, LSTM and RNN.

The article is structured as follows: In Section 2, this article presents previous relevant research and demonstrates different methods and historical changes in predicting stocks. In Section 3, this article provides a detailed introduction to the four models selected, including the reasons for choosing these methods, and introduces the theoretical foundations of statistics and mathematics. Subsequently, in Section 4, this article presents experimental results for comparative analysis and provides conclusions. Finally, in Section 5, this article provides all cited references.

2. Related works

The Stock forecasting is influenced by various factors, including historical data, fundamentals, macroeconomic, technical indicators, and emotional factors. The historical development of machine learning and deep learning has provided new tools and methods to enhance the precision and efficiency of stock prediction, enabling investors to better understand and predict the dynamics of the stock market. Early stock prediction methods mainly relied on traditional statistical techniques such as linear regression [3] and time series analysis [2]. However, these methods perform poorly in handling complex nonlinear relationships. With the improvement of computing power, machine learning methods have begun to be applied to stock prediction. For example, algorithms such as tree models [4] are used to construct prediction models.

In addition, deep learning uses neural networks to simulate complex nonlinear relationships. The application of deep learning in stock prediction is relatively new, but it has attracted widespread attention. Deep learning models, such as RNN [5] and LSTM [6], can handle large-scale datasets and capture complex patterns in time series data. These models can automatically extract features, improving the accuracy of prediction.

Utilizing linear regression for stock price prediction is foundational in financial analysis. Known for its simplicity and ease of interpretation, Linear regression provides a solid starting point for examining relationships between stock prices and potential influencing factors. It offers clarity in its assumptions and results, making it accessible for those new to financial modeling and analysis. Despite its simplicity, it can provide significant insights into stock price trends and aid in the forecasting of future prices. Linear regression can handle different types of variables, offering flexibility in analysis. It also serves as a basis for more complex modeling and machine learning applications, aiding in the understanding and exploration of more complex relationships and trends in stock market data. Its widespread use and understanding make Linear regression an essential tool for initial exploratory analysis in the financial field, assisting investors, analysts, and financial institutions in making informed decisions. For example, this analytical approach calculates the coefficients of a linear equation, incorporating one or multiple independent variables, to optimally forecast the dependent variable. It entails fitting a linear model that minimizes disparities between predicted and observed output values.

RNN exhibits robust capabilities in processing time series data, allowing for multi-step input of interconnected time series through hidden layers. This interconnectivity among time series ensures the memory function of RNN. Subsequent advancements, such as LSTM and Gated Recurrent Unit (GRU), have effectively addressed challenges like gradient explosion, gradient vanishing, and RNN convergence problems. Due to the time series characteristics of financial data, an increasing number of scholars are dedicating their research efforts to leverage RNNs for addressing challenges in stock prediction [5]. For instance, Sun Ruiqi (2020) conducted an initial analysis and comparison of the traditional BP network, RNN, and LSTM in terms of their feature extraction capabilities for time series problems, revealing superior performance by LSTM [5]. Also, to enhance prediction accuracy, researchers have fused different types of data such as stock prices, financial indicators, and social media sentiment analysis. The fusion of multimodal data can provide more comprehensive information and help to improve prediction models.

LSTM has become increasingly popular in the domain of stock price prediction and time series forecasting. In stock price prediction, historical prices often exhibit complex and sequential, and LSTMs stand out in this area due to their special capacity to use their memory cells to preserve data from earlier stock prices, effectively addressing the sequential nature of stock market data [7-9]. Unlike RNN, LSTMs are designed to navigate the challenges of vanishing or exploding gradients, ensuring consistent learning across long sequences. However, despite LSTMs' capabilities, predicting stock prices is inherently complex, swayed by numerous unpredictable factors. Also, Stock price movements often depend on previous prices and trends, which can involve time lags. LSTMs, with their memory cells, can effectively model and account for these lags, making them suitable for capturing temporal relationships in financial data.

Utilizing XGBoost for stock price prediction is advantageous in financial analysis. Known for its exceptional predictive accuracy, XGBoost efficiently handles large datasets, making it ideal for the complexity of stock market data. It identifies key influencing factors, aiding model refinement and result interpretation. XGBoost handles missing data effectively, reducing preprocessing needs. It combats overfitting through built-in regularization, enhancing generalization. Parallel processing accelerates training by leveraging multiple processor cores. Its flexibility extends beyond stock prediction to classification and ranking tasks. Supported by a robust open-source community, XGBoost remains a crucial tool for accurate and insightful stock market predictions, benefiting investors, analysts, and financial institutions in navigating financial complexities.

This article selects models based on deep learning and machine learning. The advantage of choosing two types of models is that at first researchers don't know which one is better, so researchers can conduct experiments on different models to compare their applicability. The reason why the LSTM model performs well in stock prediction is mainly due to its excellent time series processing ability. LSTM is a neural network suitable for time related data, which has the following advantages: firstly, LSTM exhibit the capability to effectively capture prolonged temporal dependencies inherent in time series data., which is crucial for understanding trends and patterns in the stock market. Secondly, LSTM has built-in memory units that can remember past information and help the model better adapt to market changes. Thirdly, LSTM can handle complex nonlinear relationships, enabling it to better capture the volatility and nonlinear characteristics of the stock market. In addition, the LSTM model has strong flexibility, can handle data at different time scales, and can automatically extract important features related to stock prices, reducing the workload of feature engineering.

3. Method

In this study, we first conducted exploratory data analysis on the dataset and preprocessed the stock data. Then, we constructed and trained machine learning and deep learning models, including linear regression model, XGBoost model, RNN model, and LSTM model, and conducted in-depth analysis of the corresponding results.

3.1. Exploratory Data Analysis

Firstly, this article conducts exploratory data analysis to ensure a foundation for subsequent data analysis of the dataset. Exploratory data analysis includes data distribution, feature analysis, variable correlation, etc. Please refer to Section 4 for detailed information.

Before establishing and training a stock prediction model, this article conducted data preprocessing. The dataset used has no missing values or outliers that must be cleaned up. Therefore, this article did not conduct too many data cleaning programs. Besides data scaling, this article utilizes the following train-test split: training set (80%) and test set (20%).

3.2. Model selection and construction

This study utilizes comparative experiments with linear regression, XGBoost, RNN, and LSTM for stock prediction. Comparing different stock prediction models can improve the quality and reliability of investment decisions, reduce risks, and help investors better understand market dynamics.

- Linear regression

Linear regression analysis is employed to forecast one variable's value based on another. The former, termed the dependent variable, is predicted using the latter, known as the independent variable. This analytical method involves estimating the coefficients of a linear equation that includes one or more independent variables, in order to improve the prediction of the dependent variable. Linear regression seeks to establish a linear model that minimizes disparities between anticipated and actual output values, often represented as a straight line or surface.

Here is a representation of the fundamental equation for linear regression, where y stands for the dependent variable attempting to predict, $x1$ to xn for the independent variables using to do so, β0 for the constant term, β1 to βn for the independent variable coefficients, and a ϵ (residual) term.

$$y = \beta 0 + \beta 1x1 + \beta 2x2 + \cdots + \beta nxn + \in \quad (1)$$

- RNN

Exhibit strong capabilities in processing time-series data, allowing for multi-step input of such data with interconnected hidden layers [5]. Given the inherent time-series characteristics of financial data, RNNs prove to be highly beneficial in addressing stock prediction challenges.:

First, RNN is an excellent tool for modeling time-series data, effectively capturing temporal correlations in stock market data. This attribute is of paramount significance when it comes to forecasting future price fluctuations., as stock prices are often influenced by temporal patterns such as seasonality, cyclicality, and trend. Second, the recursive structure of RNNs enables them to deal with long-term dependencies, and price movements in the stock market are often influenced by multiple factors over an extensive historical time frame. This helps to better understand and model the dynamics of stock prices and improve the accuracy of predictions. Moreover, RNNs have the capability of automatic feature learning, which allows them to automatically learn and extract features related to stock prices without having to manually select features. This means that RNNs can discover useful information and patterns hidden in the data, improving the performance of the model. Additionally, RNN adapts to data of different frequencies. Stock data can be day-to-day data, minute-to-minute data, or higher frequency data, and RNN can be modeled according to different data frequencies to meet the requirements of different investment strategies and needs. Furthermore, stock prices often exhibit nonlinear relationships, and traditional linear models may not be able to capture these complexities. RNNs can effectively model nonlinear relationships, improving the accuracy of predictions. Finally, RNNs can also be used for real-time decision support, which helps investors make timely decisions and has significant application potential especially for intraday and high-frequency trading strategies.

RNN is a specialized neural network architecture engineered to effectively process sequential data., with the primary purpose of capturing and utilizing information from previous time steps when processing the current input. The fundamental concept behind an RNN is the maintenance of an internal state (hidden state) that gets updated at each time step. This internal state incorporates information from the current input ($x(t)$) and the preceding internal state (h(t-1)), allowing the network to retain a form of memory throughout the sequence.

Here's a mathematical representation of a basic RNN:

Internal State Update:

$$h(t) = f(W_{hh}h(t-1) + W_{xh}x(t) + b_h) \tag{2}$$

The weight matrices$W_{hh} and\ W_{xh}$ are utilized to update the internal state by taking into account the previous state and the current input, respectively. The term b_h is a bias term, and f typically represents a non-linear activation function such as tanh or ReLU.

Output Computation:

$$y(t) = W_{hy}h(t) + b_y \tag{3}$$

W_{hy} is the weight matrix used to calculate the output, and b_y is the bias term for the output.

An essential feature of RNNs is that they use the same set of weight parameters (W_{hh},W_{xh},W_{hy}) at each time step, enabling them to share information and maintain state across the sequence. However, traditional RNNs suffer from issues like vanishing gradients and exploding gradients, making them less effective in handling long sequences. To address these problems, improved RNN structures, such as LSTM and GRU, have been developed. These advanced models incorporate gating mechanisms, enabling them to capture long-term dependencies with greater efficiency. Consequently, they are highly suitable for tasks that involve sequential data processing.

- LSTM

LSTM, initially proposed by Hochreiter and Schmidhuber in 1997 and later refined by Alex Graves, has emerged as a powerful and widely adopted neural network architecture. LSTM effectively addresses the challenge of long-term dependencies in sequential data by design, enabling it to retain and utilize information over extended time spans effortlessly. Unlike conventional RNNs, LSTM's core innovation lies in its cell state and gate structure. The cell state serves as the conduit for information flow throughout the sequence, facilitating the transmission of information from earlier time steps to later ones. This circumvents the limitations of short-term memory. The crucial aspect of LSTM's operation is the "gate" mechanism, which learns, during training, what information to retain and what to discard. This strategic gating process enhances its ability to capture and utilize critical temporal information, making LSTM a robust choice for a wide range of applications in sequential data analysis. LSTM networks represent a significant leap in the ability of models to recognize patterns over extended sequences, which is the case of stock price. Their unique architecture, which introduces memory cells and gating mechanisms, allows them to overcome challenges faced by traditional RNNs and make them a go-to choice for many sequence-based problems in deep learning.

LSTM units are a type of RNN architecture. RNN faces the gradient vanishing and exploding problems and it's more sensitive to short term input which is not the case of stock price. LSTMs are specifically designed to handle long-term dependencies, which means they can capture and remember patterns over long sequences more effectively than traditional RNNs. Given the sequential nature of stock prices, LSTMs can be especially useful in predicting stock market movements [10].

LSTMs address those by introducing a series of gate units to control the flow of information to be reminded or discarded [11]. At the heart of LSTM networks lies the pivotal concept of the cell state, complemented by an array of gates that regulate the inflow and outflow of information to and from this state. Here, we introduce the basic principle of LSTM cells.

Here's a breakdown of the main components:

Forget Gate (*f*): Determines which information from the cell state should be discarded or retained.

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{4}$$

Input Gate (i): Decides the selection of new information to be stored within the cell state.

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{5}$$

Cell State Update ($\tilde{C}$): Creates a candidate vector that might be added to the cell state

$$\tilde{C_t} = tanh(W_C[h_{t-1}, x_t] + b_C) \tag{6}$$

New Cell State (C): Combines the effects of the forget gate, input gate, and the cell state update to produce a new cell state.

$$C_t = f_t \times C_{t-1} + i_t \times \tilde{C_t} \tag{7}$$

Output Gate (o): Determines which portion of the cell state contributes to the output.

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{8}$$

$$h_t = o_t \times tanh(C_t) \tag{9}$$

where σ is the sigmoid activation function, $[h_{t-1}, x_t]$ concatenates the previous hidden state and the current input. W and b represent weight matrices and bias vectors for each gate, respectively. C_t represent the cell state of each LSTM cell unit.

- XGBoost

Using XGBoost for stock price prediction offers numerous advantages in the realm of financial analysis [4]. XGBoost, short for Extreme Gradient Boosting, is renowned for its exceptional predictive accuracy, making it well-suited to the inherently complex and non-linear dynamics of stock markets. Its benefits span various dimensions:

Firstly, XGBoost handles vast datasets efficiently, a crucial feature when dealing with the extensive time-series data associated with stock market analysis. Its capacity to process large volumes of data without compromising performance is invaluable in capturing meaningful patterns. Secondly, XGBoost provides insights into feature importance, aiding in the identification of factors that wield the most influence over stock price variations. This helps refine models and enhances the interpretability of predictions. Moreover, XGBoost is adept at managing missing data, a common issue in financial datasets, reducing the need for extensive data preprocessing. Additionally, it incorporates built-in regularization techniques to curb overfitting, improving the model's ability to generalize from historical data. Furthermore, XGBoost supports parallel processing, expediting the training process by capitalizing on multiple processor cores. Its flexibility extends beyond stock price prediction; XGBoost can be applied to classification, ranking, and other financial analysis tasks, rendering it versatile in various financial contexts. Lastly, XGBoost benefits from a robust open-source community, ensuring continuous development, support, and readily available documentation. XGBoost's outstanding accuracy, scalability, feature selection capabilities, and flexibility make it an indispensable tool for accurate and insightful stock market predictions, offering substantial advantages for investors, analysts, and financial institutions in navigating the complex world of finance.

XGBoost is the meaning of extreme gradient enhancement, which is for Improvement of Distributed Gradient Enhancement Decision Tree (GBDT). The principle of XGBoost is gradient boosting, which is an extension of boosting.

The basic principle of Boosting is to integrate the results of weak models to generate a powerful composite model. The GBDT model uses residuals from the previous tree model to fit the next model in each iteration, and the final result is the weighted sum of the ensemble results from all trees. XGBoost enhances the performance of machine learning models and improves computational speed, typically more accurate than GBDT [7].

Given a dataset $D = \{(x_1, y_1), (x_2, y_2), \cdots, (x_m, y_m)\}$, where $x_i = (x_{i1}, x_{i2}, \cdots, x_{id})^T, y_i \in R$, a tree ensemble method makes use of additive functions to forecast.

$$\hat{y}_i = \varphi(x_i) = \sum_{k=1}^{K} f_k(x_i), \ f_k \in F \tag{10}$$

where F is the space of regression trees.

The regularized objective function is

$$L(\varphi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \tag{11}$$

l is a differentiable convex loss function for difference of the prediction $\hat{y}_i$ and target y_i. Ω represents the model complexity. Since conventional optimization methods in Euclidean space are invalid, XGBoost trains the equation in an additive way. XGBoost also offers relative importance of each parameter after training [8].

3.3. Evaluation Metrics

The metrics used for model evaluation are MSE and R^2.

- Mean Squared Error (MSE)

$$\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{12}$$

where y_i is the actual value, $\hat{y}_i$ is the predicted value and N is the number of predictions.

MSE quantifies data variability by computing the average of squared errors between predicted and original data points. A reduced MSE indicates improved accuracy in characterizing experimental data using the prediction model.

- R^2

The coefficient of determination, or R^2 is defined by

$$R^2 = \frac{\sum_{i=1}^{N}(\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2} = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2} \tag{13}$$

where $\sum_{i=1}^{N}(y_i - \bar{y})^2$, $\sum_{i=1}^{N}(\hat{y}_i - \bar{y})^2$, $\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$ represent the total sum of squares (TSS), the explained sum of squares (ESS) and residual sum of squares (RSS).

$R^2$is the proportion of the predicted value that explains the proportion of *y* variable, measuring the degree to how well the predicted values align with the actual values.

4. Experimental setup and results

4.1. Dataset Overview

This paper utilizes the Google Stock Dataset from Kaggle: https://www.kaggle.com/datasets /shreenidhihipparagi/google-stock-prediction. The dataset comprises 14 columns, each corresponding to a specific attribute, and consists of 1257 rows, with each row representing the attribute values. Attributes and descriptions for the 14 columns are shown in Table 1.

Table 1. Description of Attributes in the Dataset

Attribute	**Description**
Symbol	Google Company
Date	Year and date
Close	Closing value of stock
High	Highest value of stock
Low	Low value of stock
Open	Opening value of stock
Volume	Volume of stock
AdjClose	Adjusted close value of stock
AdjHigh	Adjusted high value of stock
AdjLow	Adjusted low value of stock
AdjOpen	Adjusted open value of stock
AdjVolume	Adjusted volume of stock

Table 1. (continued).

DivCash	Dividend or cash
SplitFactor	Split factor of stock

The Figure 1 shows that the values of the open, high, low, and close columns broadly range from zero to one hundred fifty, and they all share a similar distribution. From 2004 to 2020, Google Company's open figures, high figures, low figures and close figures all go up; from 2020 and after, the company's figures go down.

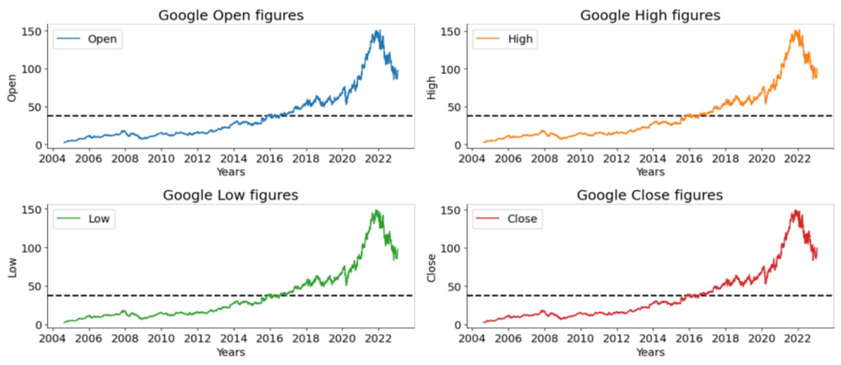

Figure 1. Distribution of the dataset

The boxplot is the most commonly used graph for data analysis, from which information like the minimum, maximum, median,etc. can be demonstrated vividly. Above all, the graph shows that the distributions of all the features are similar. Additionally, although there are several outliers, the number of them is so small that they can be ignored properly as presented in Figure 2.

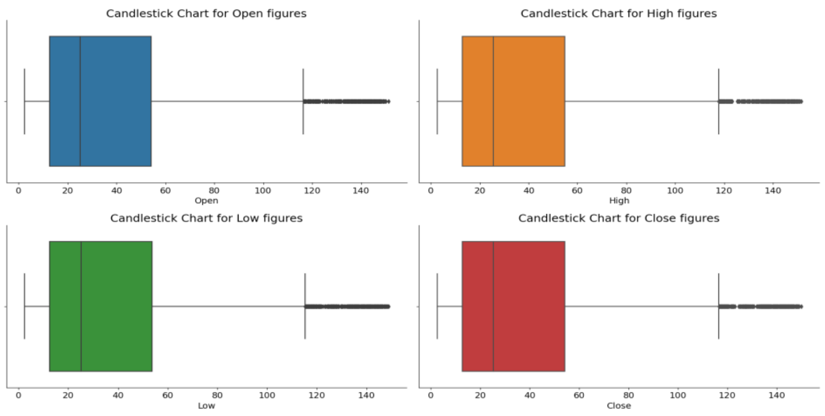

Figure 2. Candlestick Charts of the dataset

As indicated in Figure 3, The normalized dataset is divided into two subsets: a training set and a test set, following an 80-20 split ratio.The split is done sequentially instead of randomly. This split ensures a great balance between training and testing data, which simulate the real world unseen data.

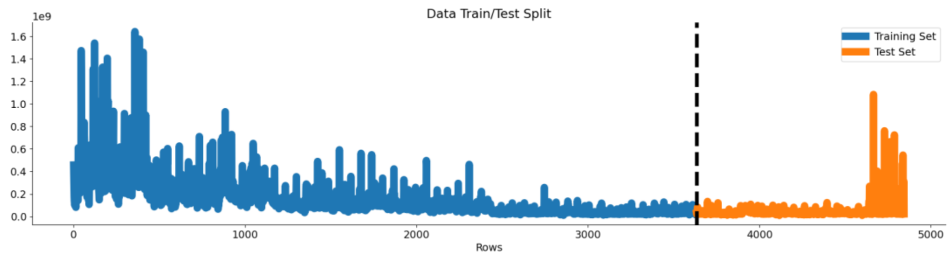

Figure 1. Train/Test Split of the dataset

4.2. Experimental Settings

In this study, all models were implemented in the Python 3.10 environment, including Pandas, Scikit Learn, Tensorflow, and XGBoost packages.The specific parameter settings for each model we used are as follows:

- Linear regression

This article utilizes ordinary least square linear regression with no adjustments.

- RNN

A RNN model consists of three layers in this experiment: two GRU layers for the input and the hidden layer, one Dense for the output. The activation function is tanh and the optimizer is Adam.

- LSTM

In this experiment, a LSTM model with three layers is created as shown in graph 2. The first layer is a LSTM input layer with 40 units. The second layer is also a LSTM layer but with 50 units. And the output layer is a dense fully connected layer with 5 units.

- XGBoost

The booster for this research in the XGBoost Regression model is gbtree, and a total of 200 gradient boost trees are utilized. The objective function is mean squared error.

4.3. Model Evaluation

These models are evaluated using MSE and R^2 metrics. The results are demonstrated in the following Table 2.

Table 2. Model Evaluation Results

Model	MSE	R^2
Linear regression	0.1090	0.998
XGBoost	0.0600	0.950
RNN	0.1812	0.981
LSTM	0.0012	0.951

LSTM has the lowest MSE of 0.0012, which is much lower than that of other models. Besides, LSTM’s R square of 0.951 is similar to that of the other three models. Therefore, LSTM provides the best performance for this task of stock prediction.

The reasons why LSTM performs best in stock prediction are as follows:

1. Ability to process temporal data: LSTM is a variant of RNN suitable for processing temporal data. Stock prices and trading data often have a significant time dependence, and LSTM has the capability to capture both long-term and short-term patterns in this time series data, making it a powerful tool for stock prediction.

2. Learning long-term dependencies: LSTM utilizes a gating mechanism that allows it to capture long-term dependencies in sequential data. This enables LSTM to recognize and retain intricate patterns and trends in the stock market, without being constrained by specific time periods.

3. Multilayer structure: LSTM can stack multiple layers to construct deep neural networks, which helps better learn and represent abstract features of data. This allows LSTM to adapt to market patterns at different levels, from short-term fluctuations to long-term trends.

4. Adjustable parameters: LSTM has many adjustable parameters that can be adjusted to adapt to different data and problems. This flexibility enables LSTM to adapt to various stock market conditions.

5. Long term memory: The memory units within the LSTM network can store and retrieve previous information, enabling them to better understand historical data and predict future price changes.

6. Adapting to nonlinear relationships: The stock market often involves complex nonlinear relationships, and LSTM, as a deep learning model, can better capture these nonlinear relationships, which is more advantageous compared to traditional linear models.

As shown in Figure 4, LSTM fits well in both train and test sets for prediction with high accuracy.

However, it should be noted that although LSTM performs well in stock forecasting, the stock market is very complex and affected by a myriad of elements, encompassing economic and political aspects, social, and natural factors. Therefore, even if LSTM and other models perform well, it is impossible to make perfect predictions. In practical applications, it is still necessary to use caution and combine other data and methods to make more accurate decisions.

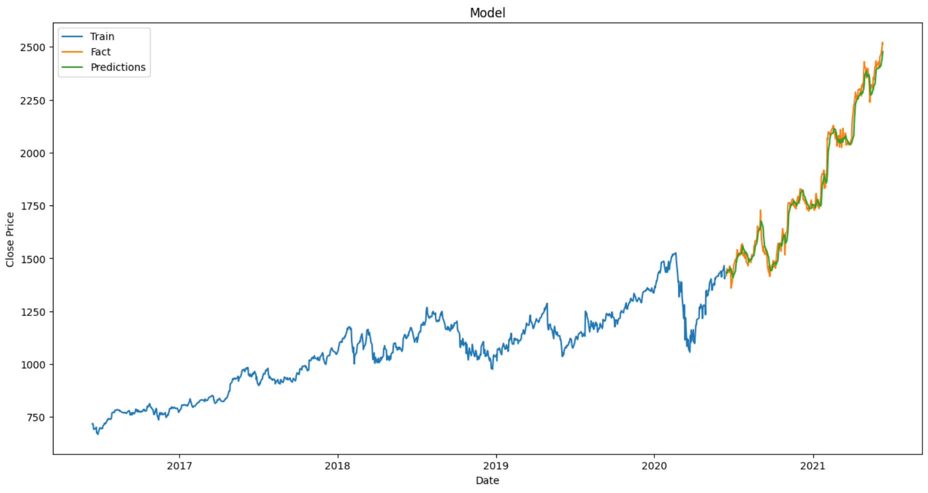

Figure 4. LSTM performance

5. Conclusion

In summary, this article includes machine learning and deep learning to find a suitable method to predict Google's stock price. These models include linear regression, XGBoost, RNN, and LSTM. Among all these models, LSTM performs the best in predicting stock prices with an R square of 0.951 and MSE of 0.0012. RNN have the highest MSE of 0.1812 and R square of 0.981; Linear regression have the second highest MSE of 0.1090 and R square of 0.998; XGBoost MSE of 0.0600 and R square of 0.950. Since all models have similar R squares, LSTM with the lowest MSE is the best model in this prediction task.

LSTM is proficient at capturing patterns of both extended and brief durations within this time series dataset, making it a powerful tool for stock prediction. In addition, LSTM has a gating mechanism that can capture long-term dependencies in sequence data. Finally, LSTM can stack multiple levels to construct deep neural networks, which helps to better learn and represent abstract features of data. It's important to acknowledge that in real-world applications, it is still necessary to use predictive models with caution and combine objective data and scientific methods to make more rigorous decisions.

For future work, this article will focus on three perspectives: Feature Engineering: additional relevant features, feature transformation, and feature selection; Modeling Techniques: ensemble methods, regularization and non-linear models; Fine-Tuning Methods: hyperparameter optimization, cross-validation and regular monitoring and updating.

References

[1] R. Patel, V. Choudhary, D. Saxena and A. K. Singh, " Review of stock prediction using machine learning techniques," 2021 Conference on Trends in Electronics and Informatics, 2021, pp. 840-846.

[2] Tang, H., Chiu, K. C., & Xu, L. (2003). Finite mixture of ARMA-GARCH model for stock price prediction. Computational Intelligence in Economics and Finance, pp. 1112-1119.

[3] Y. E. Cakra and B. Distiawan Trisedya, "Stock price prediction using linear regression based on sentiment analysis," 2015 International Conference on Advanced Computer Science and Information Systems, 2015, pp. 147-154.

[4] L. Jidong and Z. Ran, "Dynamic Weighting Multi Factor Stock Selection Strategy Based on XGboost Machine Learning Algorithm," 2018 Conference of Safety Produce Informatization, 2018, pp. 868-872.

[5] Zhao, J., Zeng, D., Liang, S. et al. Prediction model for stock price trend based on recurrent neural network. J Ambient Intell Human Comput 12, 745–753 (2021)

[6] Roondiwala, M., Patel, H., & Varma, S. (2017). Predicting stock prices using LSTM. International Journal of Science and Research, 6(4), 1754-1756.

[7] NVIDIA., “What is XGBoost?” NVIDIA Data Science Glossary, (2022). https://www.nvidia.com/en-us/glossary/data-science/xgboost/

[8] Chen, T., & Guestrin, C., "Xgboost: A scalable tree boosting system." Acm sigkdd international conference on knowledge discovery and data mining, 785-794 (2016).

[9] Lipton, Zachary C., John Berkowitz, and Charles Elkan. "A critical review of recurrent neural networks for sequence learning." arXiv preprint arXiv:1506.00019 (2015).

[10] Roondiwala, Murtaza and Harshal Patel "Predicting stock prices using LSTM." International Journal of Science and Research 6.4 (2017): 1754-1756.

[11] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," in Neural Computation, vol. 9, no. 8, pp. 1735-1780, 1997.