Journal of Economics and Public Finance
ISSN 2377-1038 (Print) ISSN 2377-1046 (Online)
Vol. 10, No. 4, 2024
www.scholink.org/ojs/index.php/jepf

Original Paper

Stock Price Prediction System LSTM Based on Deep Learning

Xizi Pan
Chengdu No.7 High tech Campus International Department, Chengdu Sichuan, China

Received: October 21, 2024 Accepted: November 23, 2024 Online Published: December 2, 2024
doi:10.22158/jepf.v10n4p235 URL: http://dx.doi.org/10.22158/jepf.v10n4p235

Abstract
Stock price prediction is a highly complex task due to the inherent volatility and non-linear patterns of financial markets. Recent advancements in deep learning, particularly Long Short-Term Memory (LSTM) networks, have yielded promising results in time series forecasting. This paper provides an in-depth analysis of LSTM models applied to the stock price prediction of major technology companies, including Apple (AAPL), Tesla (TSLA), Microsoft (MSFT), and Nvidia (NVDA). The study utilizes historical stock data to train and evaluate LSTM models, comparing their performance against various other regression models such as Linear Regression, Support Vector Regression (SVR), Decision Tree Regression, and Gradient Boosting Decision Trees (GBDT).
The results demonstrate that the LSTM model outperforms traditional models, particularly in its ability to capture long-term dependencies and manage complex, volatile stock data. Among the companies analyzed, the LSTM model achieved the most accurate predictions for Apple, as indicated by the lowest Mean Squared Error (MSE). Tesla's predictions, while less accurate due to high volatility, still showed improvement compared to other models. In conclusion, the LSTM network proved to be the most effective method for stock price prediction in this study, particularly excelling in forecasting long-term trends and reducing prediction errors for volatile stocks.

1. Introduction

With the development of artificial intelligence and technology, the field of computing has become increasingly advanced. Methods for stock prediction through big data analysis have become more robust and practical (Li, Xu, & Zhang, 2022). Many researchers have already provided answers for specific methods, categorizing stock prediction models into linear and nonlinear prediction models. Among them, linear prediction models are relatively simple in form and thus easier to model. These models predict by learning functions that are linear combinations of certain attributes. The basic principle is to use the information contained in time series data to predict future stock trends. Linear models include, but are not limited to, the AutoRegressive Integrated Moving Average (ARIMA) model, the Conditional Autoregressive Heteroskedasticity (CARAH) model, and Linear Discriminant Analysis (LDA) (Yang & Wang, 2019). These linear models also lay the foundation for nonlinear models, as many powerful nonlinear models are derived from linear models by introducing hierarchical structures or high-level mappings. However, due to the uncertainty of financial time series and the dynamic changes over time between dependent and independent variables, linear prediction models face limitations in accuracy. Therefore, nonlinear prediction models have gradually developed linear prediction models, as they have the advantage of learning complex features from data.

Currently, nonlinear prediction models include, but are not limited to, Support Vector Regression (SVR), Artificial Neural Networks (ANN), Recurrent Neural Networks (RNN), and Long Short-Term Memory (LSTM) networks. Among them, LSTM networks have certain advantages in financial forecasting (Greff, Srivastava, Koutn k, Steunebrink, & Schmidhuber, 2017). LSTM is a special type of RNN that has been further optimized based on traditional RNNs, addressing the vanishing and exploding gradient problems when handling long sequences. LSTM networks can capture long-term dependencies in time series data and flexibly obtain long-term relationships, thereby improving prediction accuracy.

First, the article will introduce some basic information about deep learning. Around the 1950s, Alan Turing's invention—Neural Turing Machines—marked the beginning of deep learning. With the rapid development of AI, in the early 21st century, AlphaGo's victory over Lee Sedol successfully brought deep learning to a whole new level. Now, with the emergence of OpenAI, deep learning is developing at an unprecedented pace, bringing unprecedented power. As shown in the figure below, deep learning is closely related to machine learning and artificial intelligence.

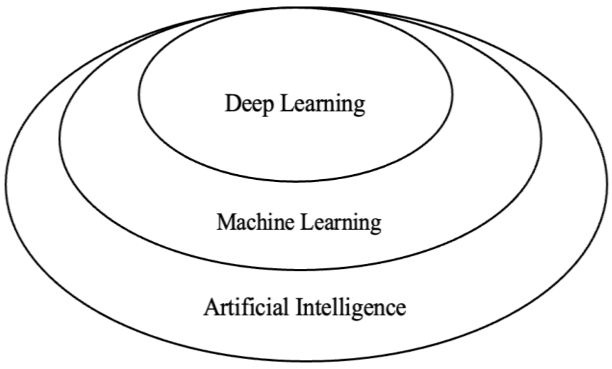


Figure 1. Three Related Types

Machine learning methods include decision trees, support vector machines, neural networks, and others (Md. Arif Istiake Sunny, Mirza Mohd Shahriar Maswood, Abdullah G. Alharbi, n.d.). Among these, neural networks constitute the most important component of deep learning. Compared to LSTM networks, Recurrent Neural Networks (RNN) encounter significant difficulties when dealing with nodes that are far apart in time series data. This is because calculating the connections between distant nodes involves multiple multiplications of the Jacobian matrix, which can lead to the vanishing gradient or exploding gradient problem. To overcome this issue, researchers have continuously developed new techniques, leading to the creation of LSTM networks.

LSTM, which stands for Long Short-Term Memory Neural Network, has its core component as the "cell state," which transmits information at each time step in the network. LSTM controls the flow of information through a structure known as "gates," specifically including the forget gate, input gate, and output gate. Each gate is composed of one or more neurons that are combined through weighted connections.

2. Model Principle

LSTM, which stands for Long Short-Term Memory Neural Network, has its core component as the "cell state," which transmits information at each time step within the network. LSTM cleverly controls the flow of information through a structure called "gates." Each gate is composed of one or more neurons combined through weighted connections. By incorporating input gates, forget gates, and output gates, the self-loop weights can be modified. Therefore, under fixed model parameters, the integration scale at different time steps can dynamically change, thereby avoiding the vanishing gradient or exploding gradient problem. Each gate is composed of one or more neurons combined through weighted connections.

The roles of the three gates are as follows:

1. Input Gate: Determines the entry of new information. It contains a sigmoid layer and a tanh layer. The sigmoid layer controls which information will be updated, while the tanh layer generates new candidate values for updating.
2. Forget Gate: Controls which information from the previous state is retained for the current state. A sigmoid function determines which information will be forgotten and which will be retained.
3. Output Gate: Determines the output at the current time step. It first compresses the hidden state through a tanh layer and then uses a sigmoid layer to decide the final part of the output.

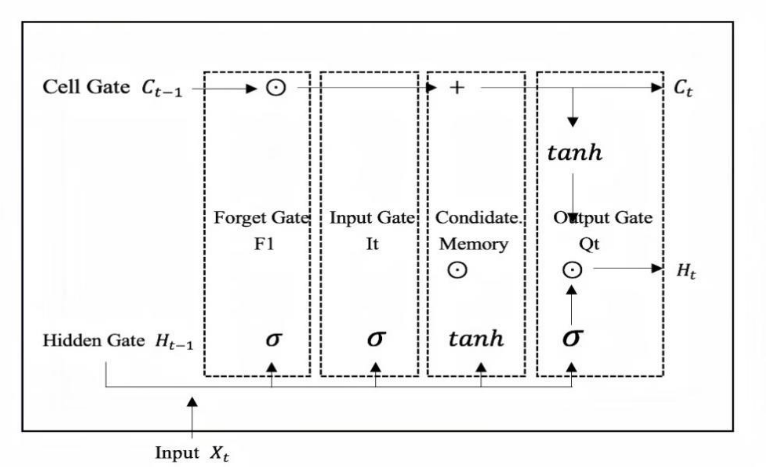


Figure 2. LSTM Structure

This diagram represents the architecture of a Long Short-Term Memory (LSTM) cell, a type of recurrent neural network (RNN) unit used in deep learning for sequence modeling. Here is an explanation of the symbols and components in the diagram:

Table 1. Explanation

Category	Element	Description
Components	Input (X_t)	The input at the current time step t
	Hidden State (H_{t-1})	The hidden state from the previous time step $t-1$, containing learned information from the prior sequence.
	Cell State (C_{t-1})	The cell state from the previous time step $t-1$, representing the long-term memory of the LSTM.
Gates	Forget Gate (F_t)	Determines which part of the cell state should be forgotten. It applies a sigmoid activation to decide what information to discard.
	Input Gate (I_t)	Controls how much new information will be added to the cell state. Combines a sigmoid activation to filter the input and a tanh function to scale it.
	Candidate Memory ($\widetilde{C}_t$)	The potential new cell state value created by the tanh activation function. This is the information added to the cell state after being gated.
	Output Gate (O_t)	Decides what information from the current cell state is passed as output. It compresses the hidden state using tanh and gates the result using sigmoid.
Operations	Element-wise Multiplication ($\odot$)	Used to apply the forget gate and input gate to the previous cell state and candidate memory respectively.
	Addition ($+$)	Combines the outputs of the forget gate and input gate to update the cell state.
	Cell State (C_t)	The updated cell state incorporating both the past memory and new information after gate operations.
	Hidden State (H_t)	The final output, which is a combination of the current cell state processed through tanh and the output gate.
Activation Functions	Sigmoid (σ)	Outputs values between 0 and 1. Used to decide the degree of information flow in gates like forget, input, and output gates.
	Tanh ($tanh$)	Outputs values between -1 and 1. Used for candidate memory and final hidden state activation to scale input values.

3. Basic Content of Experiment

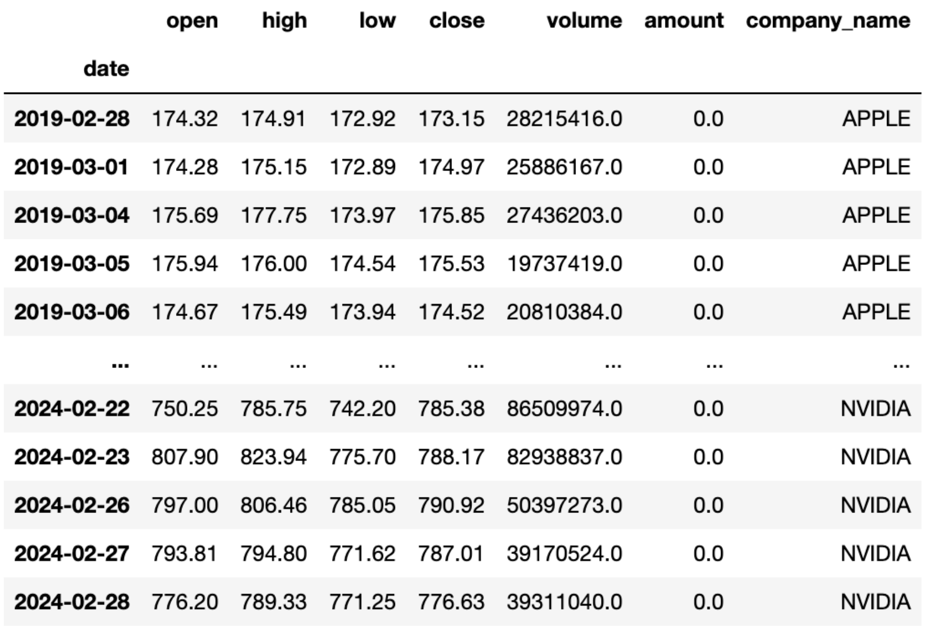
Evidence-Based Design

Using LSTM to achieve time series prediction for stocks involves the following steps: reading data, exploring the dataset, data preprocessing, dividing the training data, implementing the LSTM model, and visualizing the results. This experiment uses Python programming to apply the LSTM neural network for stock training and prediction. Deep learning is performed using TensorFlow and Keras to build and train the LSTM model, followed by graphical fitting and error evaluation.

Considering the rationality, stability, and accuracy of the experiment, this article selects different stocks from the same country's stock market as the research objects to ensure the uniformity of many political factors and fiscal policies. To reduce the possibility of individual stocks being manipulated by major players and to mitigate the influence of different markets, the experiment decides to select stocks with larger market capitalization listed on a single market within the U.S. stock market.

Data Preparation

Data Acquisition: Historical stock price data is obtained and read from an Wind detabase, including opening price, closing price, highest price, lowest price, and trading volume.



	open	high	low	close	volume	amount	company_name
date							
2019-02-28	174.32	174.91	172.92	173.15	28215416.0	0.0	APPLE
2019-03-01	174.28	175.15	172.89	174.97	25886167.0	0.0	APPLE
2019-03-04	175.69	177.75	173.97	175.85	27436203.0	0.0	APPLE
2019-03-05	175.94	176.00	174.54	175.53	19737419.0	0.0	APPLE
2019-03-06	174.67	175.49	173.94	174.52	20810384.0	0.0	APPLE
...	...	...	...	...	...	...	...
2024-02-22	750.25	785.75	742.20	785.38	86509974.0	0.0	NVIDIA
2024-02-23	807.90	823.94	775.70	788.17	82938837.0	0.0	NVIDIA
2024-02-26	797.00	806.46	785.05	790.92	50397273.0	0.0	NVIDIA
2024-02-27	793.81	794.80	771.62	787.01	39170524.0	0.0	NVIDIA
2024-02-28	776.20	789.33	771.25	776.63	39311040.0	0.0	NVIDIA

Figure 3. Data

Data Preprocessing Steps

1. Data Collection and Cleaning

Data Source Selection: Historical stock price data for Apple (AAPL), Tesla (TSLA), Microsoft (MSFT), and Nvidia (NVDA) was obtained from a financial database such as Wind, covering the period from February 28, 2019, to February 28, 2024. The dataset includes key attributes such as opening price, closing price, highest price, lowest price, and trading volume.

Data Cleaning: It’s crucial to check for missing or anomalous data points. If missing values are detected, the following approaches are used:

If the number of missing values is small, fill them using interpolation or the mean of adjacent data points.

If a significant portion of data is missing, those data entries may need to be removed to maintain the dataset's integrity.

2. Data Standardization/Normalization

To ensure that all features are comparable, the data must be standardized or normalized. This step is necessary to prevent features with larger numerical ranges from disproportionately affecting the training process. Common methods include:

Normalization (Min-Max Scaling): Scales data to a range between 0 and 1, using the following formula:

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{1}$$

Standardization (Z-score Normalization): Transforms data to have a mean of 0 and a standard deviation of 1, using the formula:

$$X' = \frac{X - \mu}{\sigma} \tag{2}$$

In this study, normalization is used to scale the data between [0, 1], which helps mitigate issues like exploding or vanishing gradients during model training.

3. Sliding Window for Time Series Data

LSTM models require sequential data, so the data must be structured in a format that captures the time series aspect. A sliding window approach is used, where data from the previous \(n\) days is used to predict the stock price on day \(n+1\). Steps include:

Set the sliding window size to n, using the stock prices from the previous 10 days (e.g., opening price, closing price) as input features, and predict the closing price on the 11th day.

This process creates rolling training and testing samples, such that data from day $(t-10)$ to day $(t-1)$ is used to predict the price on day t.

4. Train-Test Split

To evaluate the model’s performance, the dataset is split into training and testing sets. A common split is 80% training and 20% testing. It’s important to ensure that the test data is kept separate from the training data, as this will be used for performance evaluation.

For this study, data from February 28, 2019, to December 2023 is used for training, and data from January 2024 to February 2024 is reserved for testing.

5. Time Step Transformation

Convert the dates into time steps or timestamps, as LSTM requires a clear time order to process the sequence data. Typically, date formats (e.g., 2020-01-01) are converted to Unix timestamps (a continuous time format based on seconds since 1970) to ensure the temporal continuity is captured effectively.

6. Feature Selection

Five features are selected for the study: opening price, closing price, highest price, lowest price, and trading volume. These features are crucial for capturing the main movements in stock prices and provide sufficient input to predict future price changes.

Depending on the specific prediction task, more features (such as technical indicators like moving averages or volatility) could be added to improve the model's performance.

4. Model Structure

Moving averages are calculated over 10, 20, and 50 days.

Adding LSTM Layers: The LSTM layers form the core part of the model. One or more LSTM layers can be used depending on the complexity of the prediction task. Each LSTM unit will learn the time dependencies in the data.

Adding Dense Layers: The final layer is usually a fully connected layer, which converts the output of the LSTM layers into the desired prediction output.

Loss Function: Mean Squared Error (MSE) is selected as the loss function, as stock prediction is a regression problem.

In this experiment, five independent variables (parameters) are used: opening price, highest price, lowest price, closing price, and trading volume. The layers influence each other sequentially from front to back and provide feedback from back to front, forming a self-loop within the entire hidden layer.

5. Data Acquisition and Processing

Historical stock price data is obtained and read from an Excel file, including the opening price, closing price, highest price, lowest price, and trading volume. This experiment primarily studies the daily K-line data of Apple (AAPL), Tesla (TSLA), Microsoft (MSFT), and Nvidia (NVDA) from February 28, 2019, to February 28, 2024. These four stocks are selected because these four companies are representatives of the technology industry and are very attractive to investors. Each of these four companies is a leader in its industry: Apple is the world's leading consumer electronics company, and Microsoft is one of the world's largest software companies and a giant in cloud computing, office software and enterprise solutions. Nvidia is a leader in graphics processing units, and Tesla is a pioneer in the electric vehicle industry and has an innovative edge in renewable energy, autonomous driving and other fields. These companies have large market capitalization, deep influence on the U.S. stock market, and are highly transparent, making them easy to predict and analyze. The daily K-line data is downloaded using an Excel plugin, with stock attributes including the date, opening price, closing price, highest price, lowest price, and daily trading volume.

6. Analysis of Experimental Results

Analysis of four different methods' training results of the model:

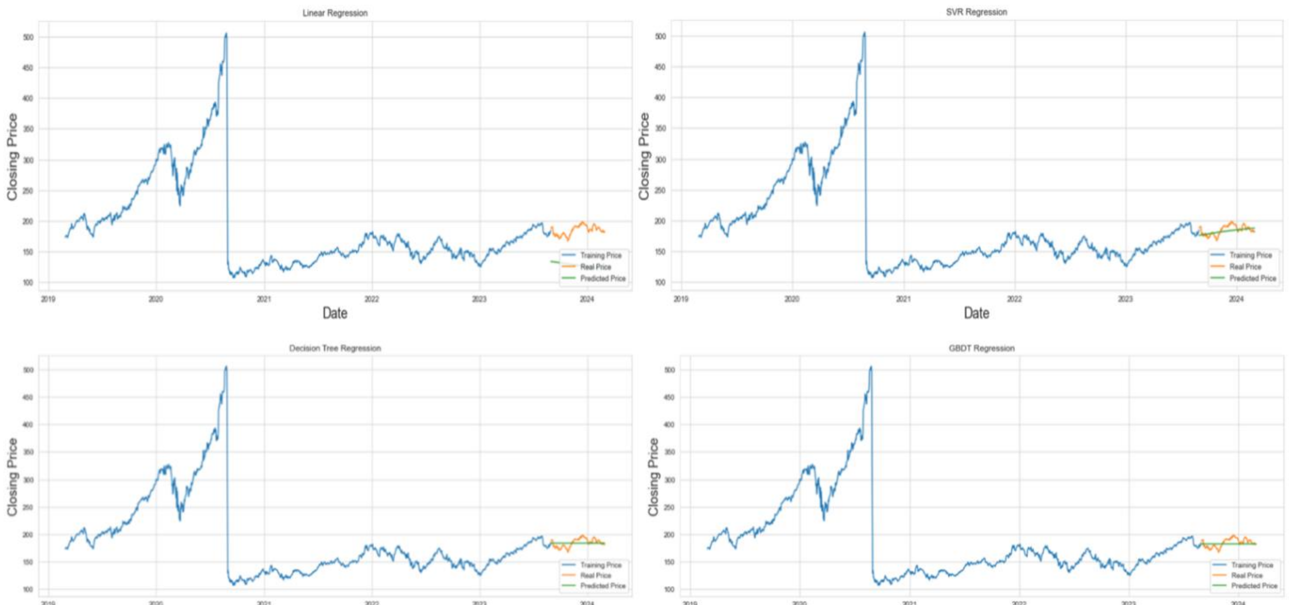


Figure 4. Training Results for AAPL

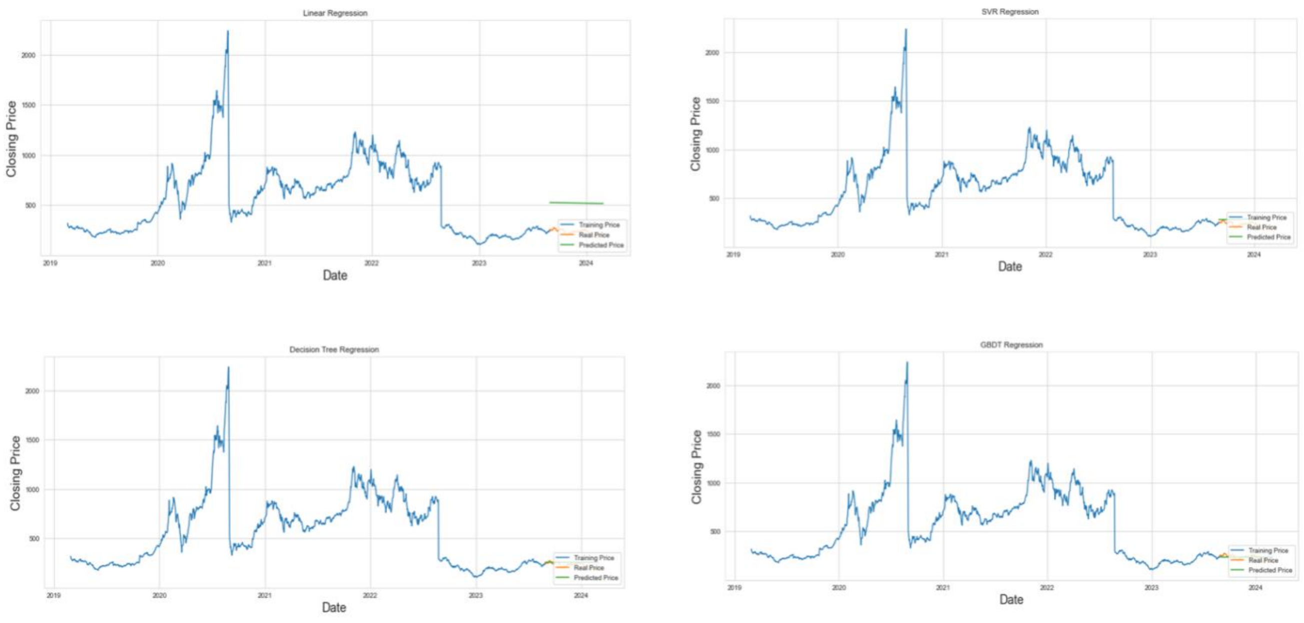


Figure 5. Training Results for TSLA

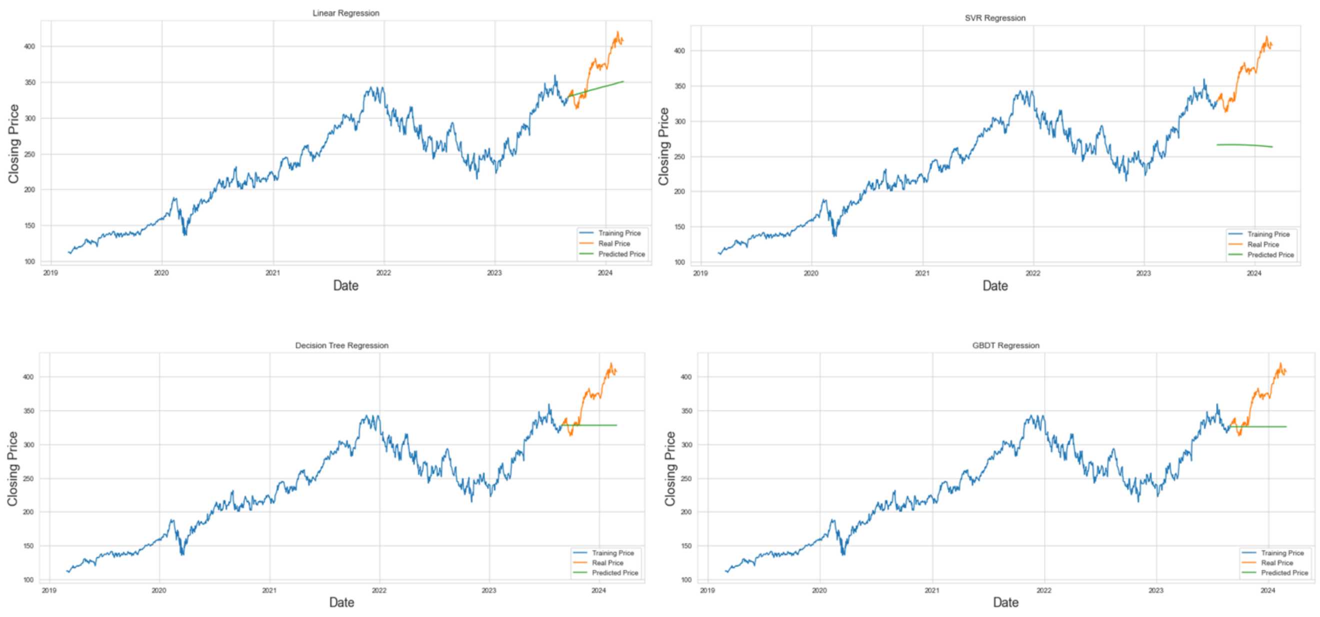


Figure 6. Training Results for MSFT

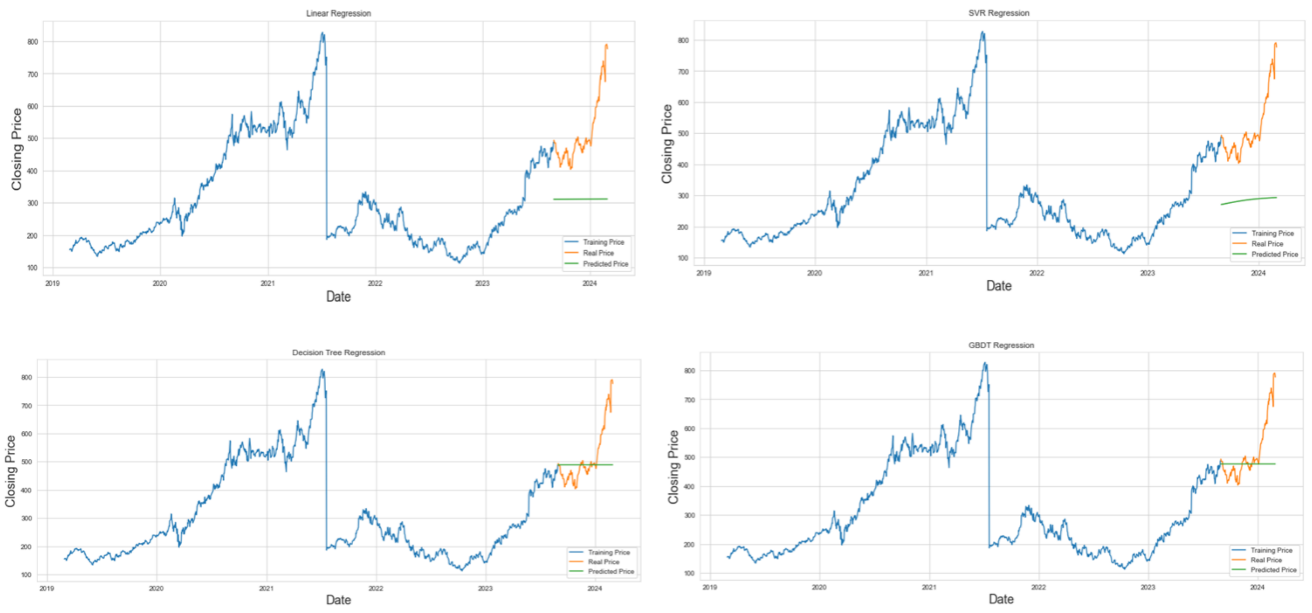


Figure 7. Training Results for NVDA

The model's prediction results are fitted into the following graph (as shown in the legend: the blue curve represents the training values, the red curve represents the actual values, and the green curve represents the predicted values. The horizontal axis represents time, and the vertical axis represents stock prices): The above chart presents the results of predicting stock closing prices using four different regression models: Linear Regression, Support Vector Regression (SVR), Decision Tree Regression, and Gradient Boosting Regression (GBDT).

Comparative Analysis of Predicted Results with Graphical Fitting

Table 2. MSE for AAPL

Method	MSE
Linear Regression	57.415683
SVR Regression	8.683609
Decision Tree Regression	7.749206
GBDT Regression	7.944308

Table 3. MSE for TSLA

Method	MSE
Linear Regression	291.929937
SVR Regression	65.335531
Decision Tree Regression	36.950371
GBDT Regression	27.555739

Table 4. MSE for MSFT

Method	MSE
Linear Regression	35.187934
SVR Regression	103.859370
Decision Tree Regression	47.541762
GBDT Regression	49.284474

Table 5. MSE for NVDA

Method	MSE
Linear Regression	228.997886
SVR Regression	253.300584
Decision Tree Regression	105.227391
GBDT Regression	109.177858

$$MSE = \frac{1}{n} \times \sum_{i=1}^{n} (y_i - \hat{y_i})^2 \tag{3}$$

Above is the formula for calculating the mean square error loss. These table presents the Mean Squared Error (MSE) of four stocks (AAPL, TSLA, MSFT, NVDA) in four different methods for predictions. MSE is a metric used to measure the difference between the predicted values and the actual values, where a smaller value indicates smaller errors and more accurate predictions.

Below is a brief analysis of each model:

Linear Regression:

The curve of the predicted prices shows a significant deviation from the real prices, especially where the prices change sharply. Linear regression models struggle to capture complex nonlinear relationships, which may lead to rough predictions.

Support Vector Regression (SVR):

The SVR model's predicted curve closely follows the real prices, particularly in the later time period. However, it also exhibits some deviations, especially when the prices fluctuate dramatically.

Decision Tree Regression:

The predicted curve of the decision tree model shows significant differences from the real prices, especially in areas with high data volatility. Decision tree models are prone to overfitting, particularly when dealing with highly volatile data, which may contribute to inaccurate predictions.

Gradient Boosting Regression (GBDT):

The GBDT model yields the best prediction performance, with the predicted curve closely matching the real price curve. The gradient boosting model improves prediction performance by integrating multiple weak learners (such as decision trees), enabling it to better capture complex patterns in the data.

Overall, the GBDT regression performs the best on this dataset, while the linear regression model performs the worst. Other models (SVR and decision tree regression) perform well in certain time periods but have limitations when dealing with sharp price changes.

Analysis of LSTM training results of the model

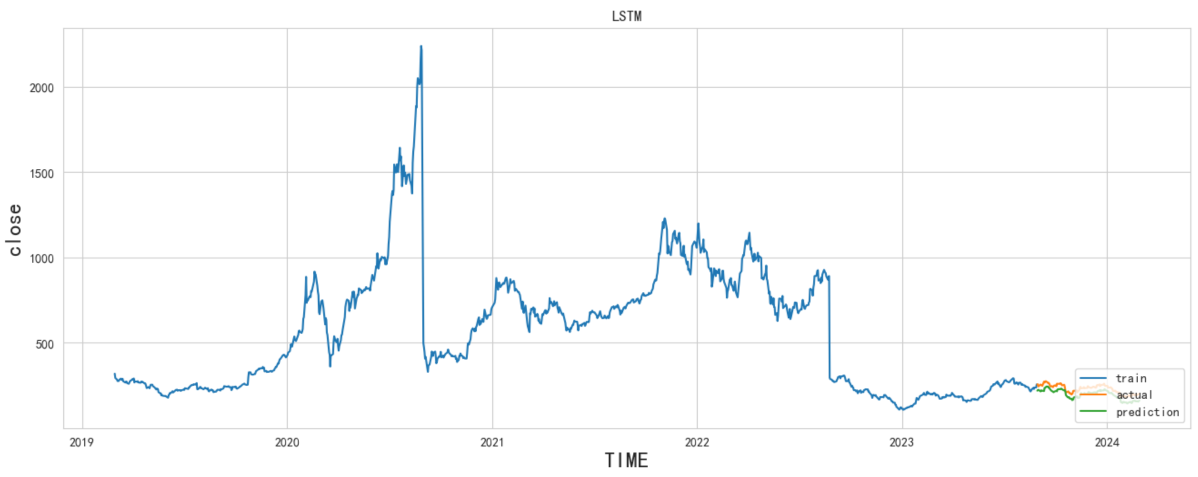


Figure 8. Training Result for AAPL

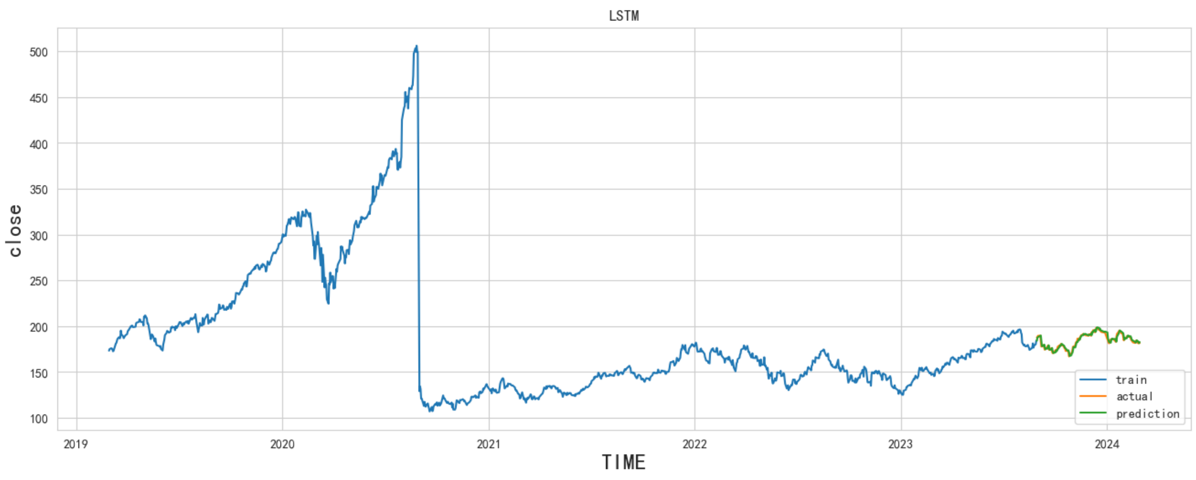


Figure 9. Training Result for TSLA

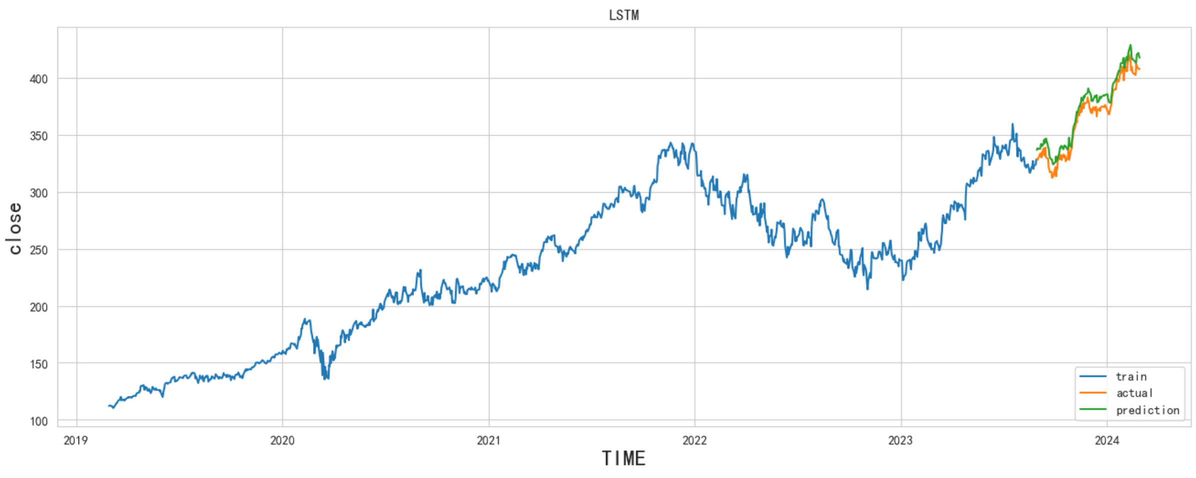


Figure 10. Training Result for MSFT

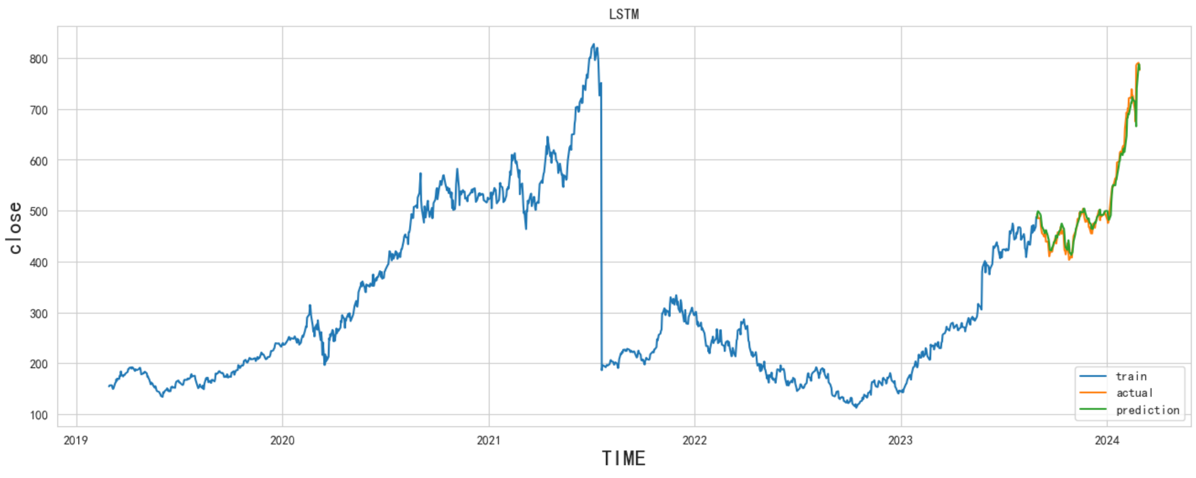


Figure 11. Training Result for NVDA

The model's prediction results are fitted into the four graphs (as shown in the legend: the blue curve represents the training values, the red curve represents the actual values, and the green curve represents the predicted values. The horizontal axis represents time, and the vertical axis represents stock prices).

Table 6. MSE for LSTM Method

STOCK	MSE
AAPL.O	2.201569
TSLA.O	33.62209
MSFT.O	9.597979
NVDA.O	17.5258

The above is the mean square error obtained by using LSTM model to forecast stocks. From the table, it can be seen that the LSTM model produced the most accurate predictions for AAPL, while the predictions for TSLA had the largest error. However, for AAPL, TSLA, MSFT and NVDA, the mean square error is significantly smaller than the first four methods (Linear Regression, SVR Regression, GBDT Regression, and Decision Tree Regression), so we can conclude that:

Compared with the first four forecasting methods, LSTM model can predict stocks more accurately.

7. Summary

This study utilizes deep learning, specifically Long Short-Term Memory (LSTM) networks, to predict the historical stock prices of Apple, Tesla, Microsoft, and Nvidia. Python, along with the TensorFlow and Keras frameworks, was used to build and train the LSTM model, with the results analyzed through graphical fitting and error evaluation. The "gated" structure of LSTM effectively overcomes vanishing or exploding gradients, allowing it to capture long-term dependencies in time series data. The findings indicate that LSTM models excel in handling time series data, improving prediction accuracy.

By comparing the LSTM model with four other models to predict the same four stocks, the value of the LSTM model was assessed. Using the Mean Squared Error (MSE) as a metric, we found that the LSTM model significantly outperformed the other four approaches, as the mean squared error between the LSTM model's predictions and the actual prices for the four stocks was the lowest, indicating highly accurate predictions. In graphical fitting analyses comparing actual stock prices with model predictions, the LSTM model demonstrated relative accuracy across different stock datasets, especially excelling in forecasting long-term trends. However, for highly volatile stocks like Tesla, further research and the development of more advanced models or algorithms may be needed to enhance prediction stability and accuracy.

In conclusion, the LSTM model shows great potential in stock price prediction, particularly in forecasting long-term trends and handling complex, fluctuating data. However, for markets with high uncertainty and volatility, further research and model optimization are still required.

References

Greff, K., Srivastava, R. K., Koutn k, J., Steunebrink, B. R., & Schmidhuber, J. (2017). LSTM:A Search Space Odyssey. *IEEE transactions on neural networks and learning systems*, *10*(10), 2222-2232.

Li Guicheng, Xu Li, & Zhang Li. (2022). *Stock price forecast analysis based on LSTM: 2095-2163*, 05-0123-06.

Md. Arif Istiake Sunny, Mirza Mohd Shahriar Maswood, & Abdullah G. Alharbi. (n.d.). *Deep Learning-Based Stock Price Prediction Using LSTM and Bi-Directional LSTM Model*.

Yang Qing, & Wang Chenwei. (2019). *A Study on Forecast of Global Stock Indices Based on Deep LSTM Neural Network: 1002-4565*, 03-0065-13.