Comparison of Different Machine Learning Approaches for Forecasting Stock Prices

Jingyao Li *

School of Electronic and Information Engineering, Tongji University, Shanghai, 201800, China

* Corresponding Author Email: 2253228@tongji.edu.cn

Abstract. Predicting stock prices is a crucial task that has significant implications for investment decisions, business strategies, and financial market stability. Accurate predictions can help investors make informed decisions, capture opportunities, and minimize risks. Understanding financial markets, economic data, company-specific elements, and a variety of statistical and analytical approaches are all necessary for making accurate stock price predictions. This paper employs three machine learning methods to forecast stock price data from a three-year time series dataset. Stock prediction plays a fundamental role in investment decisions, and accurate predictions are crucial. While various methods for predicting stock trends exist, different stock datasets may require different prediction models. Using three years of stock data, including gold and bitcoin, from the 2022 MCM Problem C, this study applies Support Vector Machines (SVM), Long Short-Term Memory (LSTM), and Convolutional Neural Networks (CNN) to make predictions. Bayesian hyperparameter optimization is employed, and a comparative analysis is conducted. Ultimately, the results indicate that CNN outperforms the other methods, exhibiting relatively low error metrics and a high R-squared value, with no apparent signs of overfitting.

Keywords: Machine learning; stock price prediction; deep learning.

1. Introduction

In financial world, people always have an expectation on stock price. Predicting stock trends is one of the most studied and difficult issues facing researchers and investors [1,2]. The stock market's tendencies are frequently unpredictable, and there are differences in the risk and reward associated with buying different combinations of stocks. Numerous methods have been developed over the years to forecast stock trends. Originally, stock trends were predicted using traditional regression techniques. Non-stationary time series data, such as stock data, can be analyzed using non-linear machine learning algorithms. The most popular machine learning techniques for stock and stock price index movement prediction are Artificial Neural Networks (ANN) and Support Vector Machines (SVM) [3, 4]. The inherent noise and volatility in daily stock price movement, however, makes it difficult to foresee these trends. By giving the network weights a probabilistic character, the Bayesian regularized network enables the network to automatically and optimally penalize overly complex models [5]. Decision algorithms are based on accurate, well-fitted predictive models. The more accurate the predictions, the lower the corresponding risks and the higher the returns. In this paper, the focus of the prediction primarily revolves around the medium to long-term forecasting of stocks, encompassing a time span of three years. This duration allows for the inclusion of a broader spectrum of economic and industry trend information, albeit alongside an increased presence of noise and instability within the dataset. Faced with such data, it becomes imperative to explore which forecasting approach is better suited for this type of time series prediction.

In this work, the dataset comes from the attachments provided for Problem C of the 2022 Mathematical Contest in Modeling (MCM) [6]. The dataset includes the daily trends of Bitcoin and gold from September 10, 2016, to September 10, 2021. The initial capital on September 10, 2019, was1, 000USA. Aler applying different prediction models to stocks and using a decision model, calculate its value on September 10, 2022. Using different prediction models and comparing their advantages and disadvantages. By adjusting parameters continuously, optimize the most prominent prediction model's performance to achieve higher accuracy.

2. Methods

2.1. SVM

Support Vector Machine (SVM) is a powerful machine learning algorithm with a core principle centered on finding the optimal hyperplane to separate datasets [7]. This goal can be formulated as an optimization problem, minimizing $||w||^2/2$ while adhering to the constraint:

$$y_i(w^T x_i + b) \geq 1 \tag{1}$$

In this equation, y_i represents the data label, x_i is the data vector, w is a normal vector of the hyperplane, and b is an intercept. However, real-world data is often more complex. Some data points deviate from the majority of points with the same data label, so slack variables,ξ_i are introduced, which allow for deviations from the ideal hyperplane. The optimization problem is then modified as follows: minimize$\frac{\|w\|^2}{2} + C\sum_{i=1}^{n}\xi_i$ subject to:

$$\begin{cases} y_i(w^T x_i + b) \geq 1 - \xi_i \\ \xi_i \geq 0 \end{cases} \quad i = 1, 2, \ldots, n \tag{2}$$

In previous equation, C represents the penalty factor. A larger C indicates less tolerance for errors, encouraging a stricter fit to the data. SVM introduces a kernel function to transform data into a higher-dimensional space for datasets that cannot be separated linearly. The definition of this kernel function is:

$$K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j) \tag{3}$$

In a high-dimensional feature space, it enables SVM to function without the need to explicitly compute the transformation $\varphi(x_i)$ this powerful technique is at the heart of SVM's ability to handle complex, non-linear data.

To solve the optimization problem effectively, SVM applies the strong duality theorem from optimization theory, transforming it into a dual problem. The dual problem aims to maximize $\theta(\alpha)$ as follows:

$$\theta(\alpha) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j K(x_i, x_j) \tag{4}$$

So that

$$\begin{cases} 0 \leq \alpha_i \leq C \\ \sum_{i=1}^{n}\alpha_i y_i \geq 0 \end{cases} \quad i = 1, 2, \ldots, n \tag{5}$$

Here, α_i represents Lagrange multipliers. Solving this convex optimization problem allows SVM to find the optimal hyperplane in the transformed feature space. This transformed space provides the necessary separation for even the most challenging datasets.

In practice, training an SVM involves inputting training samples, solving the optimization problem based on these equations, and determining the parameter b with the help of the Karush-Kuhn-Tucker (KKT) conditions. Once the training is complete, the SVM model can be used to classify new data points, making it a versatile tool for a wide range of applications.

2.2. LSTM

The Long Short-Term Memory (LSTM) network, as an improved version of recurrent neural networks (RNNs), addresses the issues of gradient explosion and vanishing as well as the problem of handling long-term dependencies in traditional RNNs [8,9]. The LSTM network leverages the concept of "gates" to achieve memory functionality over time. This not only mitigates the problem of gradient vanishing in typical RNNs, but also successfully addresses the barrier of RNNs failing to capture long-term dependencies in time series data.

The forget gate, input gate, and output gate are the main gate mechanisms used by the LSTM network to represent memory units. These gates have specific functions related to information control, information retention, adding new data, and deciding what the network should output.

Firstly, the Forget Gate (also known as F_t): The forget gate determines whether or not information from the previous time step should be remembered by evaluating the relevance of the prior cell state, c_{t-1} with the present input X_t, To produce values between 0 and 1, it uses a sigmoid activation function. A value of 0 indicates that the network should forget all of its contents, while a value of 1 indicates that no information should be forgotten. The forget gate has the following mathematical expression:

$$F_t = \sigma\left(X_t W_{xf} + H_{t-1} W_{hf} + b_f\right) \tag{6}$$

Secondly, Input Gate (Input Gate, denoted as I_t): The input gate attempts to learn important information from the input and updates the cell state. It involves two steps: It first determines which parts of the data need to be updated using the sigmoid activation function. Then, it runs the input data through the tanh activation function to create a new candidate vector. The forget gate output multiplied by the previous cell state plus the input gate output multiplied by the candidate vector equals the final cell state update. The following are the mathematical expressions:

$$I_t = \sigma(X_t W_{xi} + H_{t-1} W_{hi} + b_i) \tag{7}$$

$$\widetilde{C_t} = tanh(X_t W_{xc} + H_{t-1} W_{hc} + b_c) \tag{8}$$

$$C_t = F_t \odot C_{t-1} + I_t \odot \widetilde{C_t} \tag{9}$$

Finally, Output Gate (Output Gate, denoted asO_t): The output gate determines what should be output and transmits the modified data from the previous time step to the current time step. Like the other gates, it first utilizes a sigmoid activation function for controlling the data's range and then applies an tanh activation function to the current cell state to regulate the output range. The mathematical expressions are as follows:

$$O_t = \sigma(X_t W_{xo} + H_{t-1} W_{ho} + b_o) \tag{10}$$

$$H_t = O_t \odot tanh(C_t) \tag{11}$$

In summary, the LSTM network's use of gate mechanisms, such as the forget, input, and output gates, allows it to effectively manage and control information flow, solve the challenges of capturing long-term dependencies, and mitigate gradient vanishing problems encountered in traditional RNNs. These gate mechanisms, governed by sigmoid and tanh activation functions, provide a robust framework for sequential data processing and information retention.

2.3. CNN

The application of Convolutional Neural Networks (CNNs) to time series data involves a systematic approach to capturing temporal features and patterns within the data [10]. Initially, it is imperative to set a random seed to ensure the reproducibility of experiments. This precaution is essential for maintaining consistency in training outcomes across different runs. Once the seed is established, the model architecture is defined. A Sequential model, which is essentially a linear stack of layers, is employed for this purpose.

Within this model, several key components are integrated. Notably, Convolutional Layers (Conv1D) are employed to extract pertinent features from the time series data. In this context, these layers are configured with a Rectified Linear Unit (ReLU) activation function and 12 convolutional kernels of size 3. These convolutional layers are pivotal for identifying patterns within the temporal data. Additionally, a Dropout layer is included with a 20% neuron dropout rate, serving as a regularization technique to mitigate overfitting.

Following the convolutional layers, a Flatten layer is used to reshape the output, making it compatible for connection to the subsequent Fully Connected Layer (Dense). This fully connected

layer is responsible for producing the final predictions and typically involves regression tasks. In the specified model, a single fully connected layer is employed, generating an output with a dimension of 1.

Regarding optimization, the Adam optimizer is selected as the optimization algorithm. The optimizer is a critical component responsible for iteratively adjusting model parameters to minimize the chosen loss function, in this case, the Mean Squared Error (MSE). To facilitate the optimization process, the learning rate is fixed at 0.01.With the model architecture defined and the optimization strategy in place, the final step is model compilation. During this stage, the chosen loss function, MSE, and an evaluation metric, Mean Absolute Error (MAE), are specified. The MSE serves as the model's objective function to minimize during training, while MAE is monitored to gauge the model's performance.

3. Data Analysis

The dataset comes from the attachments provided for Problem C of the 2022 MCM (Mathematical Contest in Modeling). The dataset includes the daily trends of Bitcoin and gold demonstrated in Table 1 and Table 2, from September 10, 2016, to September 10, 2021.

Table 1. Demonstration of bitcoin prices.

Date	Value
2009/11/16	621.65
2009/12/16	609.67
2010/01/16	610.92
2010/02/16	608.82
2010/03/16	610.38
2010/04/16	609.11
…	…

Table 2. Demonstration of gold prices.

Date	Value
2009/12/16	1324.6
2010/01/16	1323.65
2010/02/16	1321.75
2010/03/16	1310.8
2010/04/16	1308.35
2010/05/16	1314.85
…	…

The data preprocessing phase involved an initial inspection for missing values, followed by the imputation of missing data points using the previous day's values. To gain a deeper understanding of the data, a visualization was conducted using line graphs to depict the price trends of Bitcoin and gold over the course of the three years.

4. Result

4.1. Performance of SVM

When comparing the predicted data generated by the SVM method with the original data, it becomes evident that the overall trends of the predicted data are generally consistent with the actual data, as displayed in Table 3 and Fig 1 respectively. However, there are discrepancies in the specific data details. Notably, in the latter half of the year, the accuracy of the predicted data is lower, and it

tends to underestimate the actual values. Additionally, the predicted data exhibits smoother fluctuations compared to the actual data, with less pronounced peak values.

Table 3. Performance of SVM.

	Training Set Evaluation Metrics	Testing Set Evaluation Metrics
MSE (Mean Squared Error)	1188826.4537	32907428.4950
MAE (Mean Absolute Error)	915.7628	32907428.4950
RMSE (Root Mean Squared Error)	1090.3332	5736.4997
R-squared (R^2)	0.9108	0.8628

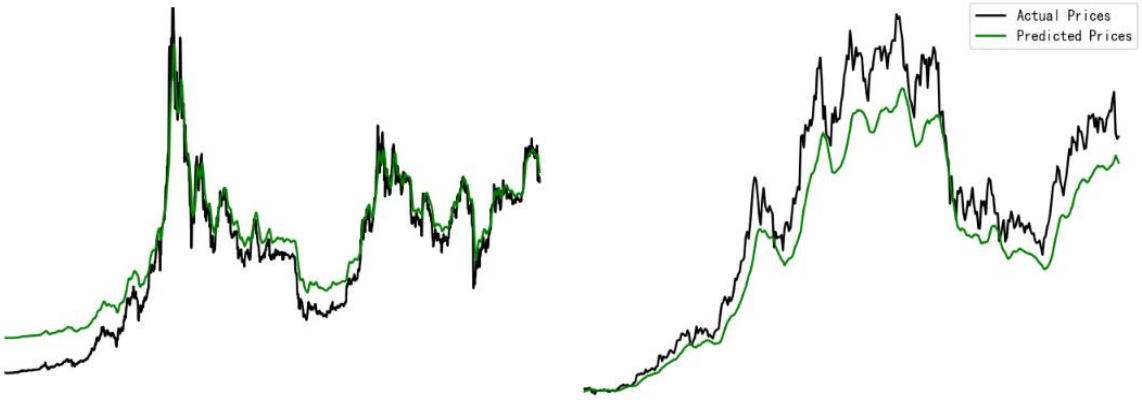

Fig 1. Comparison of bitcoin price predicted by SVM (Figure Credits: Original).

4.2. Performance of LSTM

The overall accuracy of the LSTM model is higher, especially in the first half of the data where predictions are particularly accurate, as displayed in Table 4 and Fig 2. Notably, the model accurately captures peak values without showing a reduction in their magnitude. In the latter half of the data, there are still instances where the predicted values are lower than the actual data, indicating some discrepancies. However, despite these deviations, the predictive results are quite satisfactory and acceptable.

Table 4. Performance of LSTM.

	Training Set Evaluation Metrics	Testing Set Evaluation Metrics
MSE (Mean Squared Error)	211228.5647	24949443.6389
MAE (Mean Absolute Error)	325.5053	3819.0926
RMSE (Root Mean Squared Error)	459.5961	4994.9418
R-squared (R^2)	0.9842	0.8960

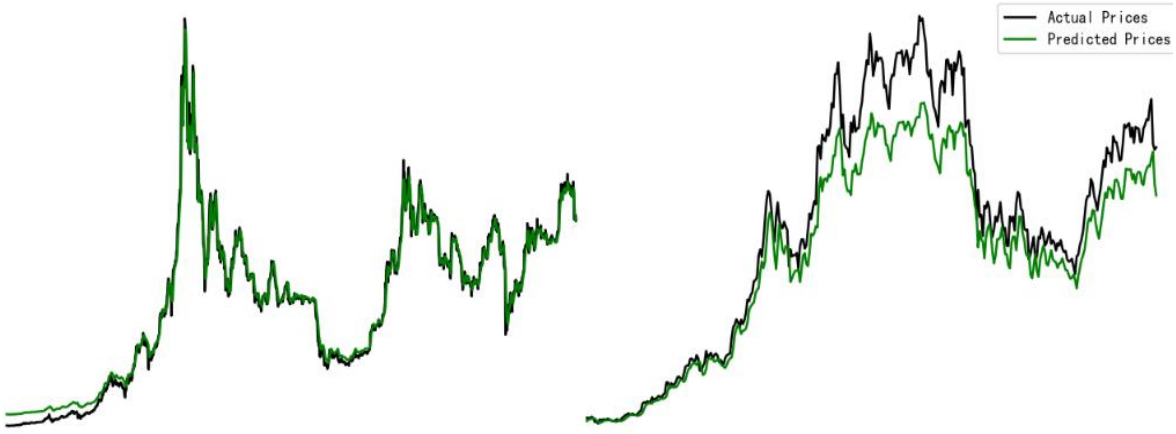

Fig 2. Comparison of bitcoin price predicted by LSTM (Figure Credits: Original).

4.3. Performance of CNN

The CNN model demonstrates outstanding performance on the training set with low MSE and MAE values, as displayed in Table 5 and Fig 3 respectively. This shows that the training data's underlying patterns and trends are successfully captured by the model. Its high accuracy, precision, and consistency make it the best choice among the three models for this particular prediction task.

Table 5. Performance of CNN.

	Training Set Evaluation Metrics	Testing Set Evaluation Metrics
MSE (Mean Squared Error)	301458.7754	2929695.9956
MAE (Mean Absolute Error)	408.1356	1222.6130
RMSE (Root Mean Squared Error)	549.0526	1711.6355
R-squared (R^2)	0.9774	0.9878

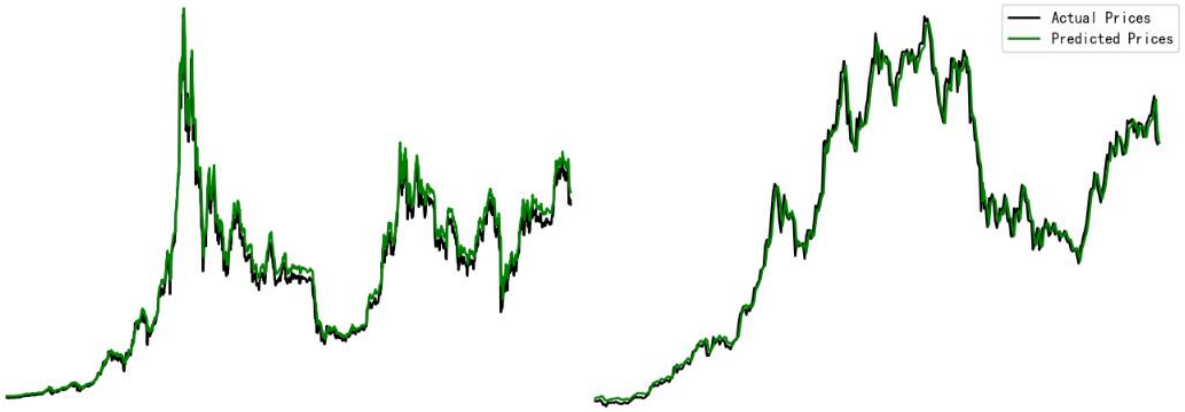

Fig 3. Comparison of bitcoin price predicted by CNN (Figure Credits: Original).

5. Discussion

Three models are compared and analyzed based on the provided evaluation metrics as shown in the following paragraps.

In SVM, good performance on the training set is indicated by the training set assessment measures, which also show strong R-squared and relatively low MSE and MAE values. The testing set exhibits relatively high MSE and MAE, along with a high RMSE and a lower R-squared value, suggesting poorer performance on the testing set compared to the training set. This difference may indicate that the SVM model is overfitting to the training set on the testing set.

As for LSTM, the training set evaluation metrics display low MSE and MAE values, as well as a high R-squared value, showcasing excellent performance on the training set. The testing set has relatively high MSE and MAE, a high RMSE, and a lower R-squared value, indicating comparatively worse performance on the testing set, but still better than SVM. Overfitting may still be present on the testing set, although the LSTM model performs more accurately than SVM.

In CNN, the training set evaluation metrics indicate low MSE and MAE values, as well as a high R-squared value, showcasing strong performance on the training set. The testing set exhibits low MSE and MAE, along with a low RMSE and a high R-squared value, indicating excellent performance on the testing set, outperforming both SVM and LSTM. There appear to be fewer signs of overfitting, as the testing set performance closely aligns with the training set.

In summary, the CNN model appears to be the best choice in this context, as it performs the best on the testing set, showing relatively low error metrics and a high R-squared value, with no apparent signs of significant overfitting. The LSTM model performs well on the training set but slightly lags behind the CNN on the testing set, while the SVM model performs the worst, potentially indicating overfitting.

6. Conclusion

This study analyzed and forecasted three years of Bitcoin and gold prices using machine learning methods. The dataset contained a significant amount of noise and instability, particularly in the case of Bitcoin, which experienced periods of high demand and price volatility. Bayesian hyperparameter optimization was employed to identify the best parameters, and the prediction results indicated that CNN outperformed other models, with LSTM following closely behind and SVM performing the least effectively. Therefore, for medium to long-term stock data, CNN may be a preferable choice.

References

[1] Chen, Wei, Manrui Jiang, Wei-Guo Zhang, and Zhensong Chen. A novel graph convolutional feature based convolutional neural network for stock trend prediction. Information Sciences, 2021, 556: 67-94.

[2] Gandhmal, Dattatray P., and K. Kumar. Systematic analysis and review of stock market prediction techniques. Computer Science Review, 2019, 34: 100190.

[3] Patel, Jigar, Sahil Shah, Priyank Thakkar, and Ketan Kotecha. Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. Expert systems with applications, 2015, 42(1): 259-268.

[4] Soni, Payal, Yogya Tewari, and Deepa Krishnan. Machine Learning approaches in stock price prediction: A systematic review. In Journal of Physics: Conference Series, 2022, 2161(1): 012065.

[5] Ticknor, Jonathan L. A Bayesian regularized artificial neural network for stock market forecasting. Expert systems with applications, 2013, 40(14): 5501-5506.

[6] Mathematical Contest in Modeling, 2022, URL: https://www.contest.comap.com/undergraduate/contests/mcm/contests/2022/results/. Access Time: 2023/11/12.

[7] Noble, William S. What is a support vector machine? Nature biotechnology, 2006, 24(12): 1565-1567.

[8] Yu, Yong, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. A review of recurrent neural networks: LSTM cells and network architectures. Neural computation, 2019, 31(7): 1235-1270.

[9] Sherstinsky, Alex. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. Physica D: Nonlinear Phenomena, 2020, 303: 132306.

[10] Gu, Jiuxiang, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu et al. Recent advances in convolutional neural networks. Pattern recognition, 2018, 77: 354-377.