Predicting U.S. Stock Market Movements: Does LSTM Outperform Classical Classification Algorithms?

Wang Ruitao 1155190411   Ni Zhichen 1155232779

1. Introduction

Stock market prediction is always a meaningful topic to investors, and it has long been characterized by its dynamic, complicated, and non-stationary nature (Fama, 1965). Previous scholars have used a wide variety of different approaches to pursue better performances in stock market predictions.

Schumaker and Chen (2010) have demonstrated that SVM is a machine learning algorithm that can classify a future stock price direction (rise or drop). Das and Padhy (2012) used backpropagation (BP) and SVM to predict future prices in the Indian stock market. The performance of these techniques is compared and it is observed that SVM provides better results when compared with the results from the BP technique. However, Huang, Yeh, and Lee (2011) address problems when using support vector regression to forecast stock market values when dealing with kernel function hyperparameters. The study by Patel, Shah, Thakkar, and Kotecha (2015) compares four Indian stock market prediction models from 2003-2012, including ANN, SVM, Random Forest, and Naive-Bayes, which finds that random forest outperforms the other three prediction models. Basak (Basak et al, 2019) and other scholars (Jiao, 2017) have used random forest and gradient-boosted decision trees, and have all reported a high accuracy for predicting up and down movements for various companies in the U.S. stock market.

Apart from traditional classification approaches, scholars also developed different deep learning models to capture the complex dynamic stock market movement trends. LSTM, as an extension of recurrent neural networks (RNNs), is mainly introduced to handle time series forecasting, as the latter fails to store information for a longer period of time and is incapable of handling such "long-term dependencies". Given that the complex nature of the structure for LSTM is designed to capture time series data, the tradeoff between time efficiency and the extent of increase in accuracy is worth considering.

These varied models for prediction arouse our attention to which model can give the best performance while balancing the tradeoff of the time taken for training the model. This paper will compare the performance of different models in stock market prediction and investigate whether it is worth sacrificing efficiency by using LSTM.

2. Related Work and Literature Reviews

2.1 Target Selection

Studies (Thawornwong & Enke, 2004) have shown that when developing forecasting and trading systems, direction forecasts (up or down) perform better in a trading simulation than particular level forecasts (an exact value prediction of the stock or index one period forward). To ensure the accuracy of predictions, this paper primarily focused on the predictions of the direction of the market. It is decided to use data from the S&P 500 index, which consists of 500 companies in the U.S. Stock Exchange market, as the indicator of the overall U.S. stock market performance.

2.2 Features Selection

Public opinions and investors' confidence are highly heterogeneous and structured, and can be positive, negative, or neutral to the stock market (Lakshmi et al. 2017). Schumaker and Chen (2009) conducted textual analysis in conjunction with the SVM approach to observe the impact of news articles on stock prices. They investigated a large sample of stock quotes and financial articles during five weeks in 2005, showing that models containing both article terms and the stock price at the time of article release indicate the closest estimation in terms of showing the same direction of price movement compared with the future price. Khan and Malik (2020) tried to fetch data from Twitter and conducted textual analysis to investigate the effect of sentiment and political situations on stock prices but found that the impact of public sentiment on a particular company is minimal. They concluded that there are two possible reasons for this: either sentiment has no impact or the sentiment analysis techniques that currently exist cannot provide much information about the correlation between sentiment and a company's stock. Therefore, this paper directly extracted an existing sentiment indicator, rather than conducting an independent Natural Language Processing (NLP) investigation to detect market sentiment, for the sake of accuracy. To choose the suitable sentiment indicators to include in the study, we referred to the study by G Campisi, S Muzzioli, and B De Baets (Campisi et al., 2024), which discussed the effect of market sentiment on predicting the up and down trend of the U.S. stock market by using the CBOE Volatility Index, which also known as VIX index. This index measures the market's expected volatility on the S&P 500 Index, calculated and published by the Chicago Board Options Exchange (CBOE). It is calculated by selecting a range of call and put options on the S&P 500 Index with strike prices near the current market price. The implied volatilities are weighted and averaged to produce a single value that represents the market's expected volatility over the next 30 days, and a higher index indicates higher expected volatility and uncertainty.

Chavan and Patil (2013) reviewed nine published articles that used the ANN approach to predict stock prices to attempt to find the most important input parameters that produce better model prediction accuracy. Based on their investigation, they find that most machine learning techniques make use of technical variables instead of fundamental variables for a particular stock price prediction, while microeconomic variables are mostly used to predict stock market index values. In addition, hybridized parameters produce better results when compared with the use of only a single input variable type. Therefore, this paper included several common macroeconomic indicators but did not include any technical variables such as RSI or MACD.

We also added the U.S. interest rate, the Federal Funds Rate, as one of the macroeconomic features. The Federal Funds Rate is announced by the Federal Open Market Committee (FOMC), which is the target interest rate set by around only once per two months. Therefore, to seek an alternative rate that can match the frequency of stock data (which is daily), we refer to Bhandari's research who conduct LSTM analysis to predict stock market trends (Bhandari et al, 2022). In their study, they used the effective federal funds rate (EFFR) as the measure of interest rate, which is a dynamic rate at which banks lend reserves to each other. The EFFR is calculated as a volume-weighted median of overnight federal funds transactions reported in the FR 2420 Report of Selected Money Market Rates, influenced by market demand and supply so it is a dynamic rate changed daily.

In addition to the macroeconomic features, the paper used the ICE U.S. Dollar Index as an indicator of the exchange rate, as referred to in Johansson's study (2024). The index is a measure of the value of the United States dollar relative to a basket of foreign currencies. It is a weighted geometric mean of the dollar's value compared to six major world currencies, including Euro (EUR), Japanese Yen (JPY), British Pound (GBP), Canadian Dollar (CAD), Swedish Krona (SEK) and Swiss Franc (CHF).

The overview of all selected variables in this paper is shown in Table 1 in the Appendix. We used the Phi-k correlation matrix to show the correlation between variables since all variables are converted into binary form. Details can be found in Table 2 in the Appendix.

2.3 Feature Engineering

In the paper, all coding was conducted using Python.

1. We retrieved the daily close price of all variables by using the Yahoo Finance library from Python (yfinance), with the only exception of interest rate. The

interest rate data was collected from the official website of the Federal Reserve Bank of New York through data scraping.

2. The data of all variables are then combined and merged into a single data frame, removing rows with N.A. values.
3. Rather than directly calculating daily differences to convert into binary variables, we used 3-day and 6-day moving averages to smooth out short-term fluctuations and highlight longer-term trends, whereas this method is referred from Hui and Yu's research (2022). On a particular day, if the 3-day moving average is larger than the 6-day, it indicates faster growth in the short term, then it will be considered as an upward trend (coded as 1 for this day), and vice versa (coded as 0).

$$moving\ average\ of\ N-day = \frac{1}{N}\sum_{i=1}^{N} P_i$$

2.4 Research Scope Design

By merging all data and conducting all necessary preprocessing as mentioned above, we left with daily data from March 1st 2016 to Oct 25th 2024, in total 2177 rows of observations.

3. Methodology

3.1 Models Selection

This study utilizes seven models to predict the movement of the S&P 500 index. We used logistic regression as the baseline and used 5 classification algorithms including Naive Bayes, Linear Discriminant Analysis (LDA), Support Vector Machines (SVM), Decision Tree, K-Nearest Neighbors (KNN), and compared with the deep learning algorithm Long Short-Term Memory (LSTM). The diversity of these models allows us to evaluate a range of methodologies, from simple statistical techniques to advanced neural network-based approaches, ensuring robust analysis.

3.1.1 Logistic Regression (Logit)

Logistic regression serves as a baseline model for binary classification, predicting the probability of upward or downward movement in the S&P 500 index. It was chosen for its simplicity and interpretability, enabling comparisons against more complex models. The hyperparameter C (regularization strength) was tuned to balance bias and variance.

$$\Pr(Y_i = y|X_i) = \frac{e^{\beta X_{i*y}}}{1 + e^{\beta X_i}}$$

3.1.2 Naive Bayes

Naive Bayes assumes feature independence, making it computationally efficient for classification tasks. Despite its simplicity, the algorithm has shown reliability in handling noisy datasets, such as financial time series. In this paper, we used the Bernoulli Naive Bayes that assumes that the features are binary and follow a Bernoulli distribution, which is a more appropriate assumption for binary data.

$$P(y|x_1, \dots, x_n) = \frac{P(x_1|y) \dots P(x_n|y)P(y)}{P(x_1)P(x_2) \dots P(x_n)}$$

Linear Discriminant Analysis (LDA)

LDA is a linear classification model that seeks to maximize the separation between classes by projecting data onto a lower-dimensional space. It was included to test the effectiveness of a dimension-reduction approach in predicting S&P 500 movements, especially given the multi-dimensionality of our input features.

$$P(x|y = k) = \frac{1}{(2\pi)^{p/2}|\sum k|^{1/2}} \mathrm{e}^{(-\frac{1}{2}(x-\mu_k)^t \sum k^{-1}(x-\mu_k))}$$

Support Vector Machines (SVM)

As a popular choice for stock market prediction, SVM was chosen for its robustness in high-dimensional data and ability to handle non-linear relationships through kernel functions. An RBF kernel was employed, with hyperparameters C (regularization strength) and γ (kernel coefficient) fine-tuned via grid search to optimize performance. SVM is particularly well-suited to financial markets, where relationships between variables are often complex.

$$min\ L = \frac{1}{2}||w||^2 - \sum_{i=1}^{l} a_i y_i (x_i \cdot w + b) + \sum_{i=1}^{l} a_i$$

Decision Tree

Decision Trees provide a straightforward approach to classification, splitting data into branches based on feature thresholds. Their interpretability is a significant advantage, as they offer insights into the most influential features driving predictions. The depth of the tree and the minimum number of samples per split were carefully optimized to avoid overfitting.

$Gain(S, A) = Entropy(S) - \sum_v^A \frac{|S_v|}{|S|} \cdot Entropy(S_v)$,
where entropy is:

$$E(T, X) = \sum_{c \in x} P(c)E(c)$$

K-Nearest Neighbors (KNN)

KNN classifies data points based on their proximity to the K nearest neighbors. It is particularly useful for exploring local patterns in the data, such as recent market movements. The number of neighbors and the distance metric were optimized to improve the algorithm's ability to generalize beyond the training data. A typical algorithm, Minkowski Distance, is calculated by:

$$d(x, y) = (\sum_{i=1}^{n} (x_i - p_i)^p)^{\frac{1}{p}}$$

Long Short-Term Memory (LSTM)

As a special type of recurrent neural network that is designed for time series data analysis, LSTM excels in capturing temporal dependencies in sequential data. Its ability to retain information over longer periods makes it ideal for financial time series predictions, where past movements significantly influence future trends. It involves four different gates, Forget Gate, Input Gate, Input Modulation Gate, and Output Gate. Key hyperparameters included a learning rate, dropout rate, and hidden layer size.

3.2 Cross-Validation

To ensure the robustness and generalizability of our models, we employed a walk-forward cross-validation approach, a method well-suited for time-series data. This technique sequentially trains and tests the model on subsets of the data multiple times, respecting the chronological order to prevent data leakage and better mimic real-world conditions.

The dataset was first split into three parts: training, validation, and testing. The specific proportion for each set is decided to be 80:10:10, as concluded by Shad (2018) that this is the optimal result across different setup scenarios in his research. During training and validation, a sliding window mechanism was used, where a fixed training window of 50 days was employed, followed by testing on the next 50 days (i.e., train the model on days 1-50, evaluate on days 51-100, then train on days 1-100 and evaluate on days 101-150, and so on). This window was then shifted forward step-by-step until the entire training set was utilized, ensuring there was no data leakage for the 10% testing set throughout the process. For each fold:

1. Trained the model on a specific window (i.e., 50) of the training data.
2. Using GridSearchCV to perform hyperparameter tuning within each fold of the walk-forward validation. Each hyperparameter configuration was evaluated on the validation set of the fold within the sliding window.
3. Evaluated the performance for each split by error function. Since all variables are binary, we used the binary cross entropy (log loss function), which lowers the error function better the model is.
4. Retrained the model on the entire training dataset using the best hyperparameter settings found during the fine-tuning process.
5. Conduct the final evaluation by comparing the predictions to the actual dataset (testing set).

This iterative process ensured that the model's performance was evaluated across multiple time windows, reducing the risk of overfitting and ensuring that the models could generalize well to unseen data. Additionally, this method accounts for the temporal dependencies in financial data, a critical factor for time-series prediction.

4. Result and Conclusion

We printed out the performance matrix of each model based on accuracy, precision, recall, F1-score, and Area Under Curve (AUC). Details can be found in Table 3 in the Appendix.

$$Precision = \frac{True\ Positves}{Total\ Positives}$$

$$Recall = \frac{TruePositves}{TruePositives + FalseNegative}$$

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

Based on the result, the results of accuracy, precision, recall, and F1-score do not vary significantly across different models. Overall, the Naïve Bayes performed the best in all models. In terms of accuracy, it shows the highest performance, with 0.74, while also presenting the highest F1-Score for 0.75. It is tied with the logit regression at 0.8195 in the criterion of AUC. Surprisingly, the LSTM that we expected to outperform other models, is only slightly better than a few models such as KNN and SVM. It is also noticed that the KNN model and the SVM model indicate almost the same performance, that the AUC only varied by 0.0003. Even more surprisingly, the performance for the model that we used as baseline, logit regression, performed very well, with 0.71 accuracy and 0.8195 AUC.

5. Discussion and Limitations

This paper has shown that in terms of predicting a simple binary variable, such as the upward and downward trend of the stock market, traditional classification models are already complex enough to make high-performed prediction analyses. In addition, it is found that models with simpler structures can perform better than those with complex structured models when predicting simple data. Although LSTM has a complex structure in nature that is designed to make time series analysis, its performance can be even worse than Naïve Bayes and logit regression when dealing with simple data.

Therefore, it is concluded that, if the nature of data is direct and simple, it is not worth using complex models, such as LSTM, to capture the underlying patterns in the data. Instead, it is rather to use more simple models for the sake of efficiency of training.

Similar conclusions were found by other scholars previously as well. Ou and Wang (2009) used 10 classification techniques to capture the daily Hang Seng Index return trend, and the results of accuracy only ranged from 0.7960-0.8640. Qian and Rasheed (2007) also used 3 classification models to predict daily returns of the Dow Jones Industrial Average (DJIA) index, and the accuracy of prediction fluctuated within 2% across three models.

We believe that the results of different models will be more distinguishable if we increase the complexity of data, for example, by creating more categorical variables in the feature engineering process rather than binary variables. This can be done by further dividing the variable, such as upward trend “1”, into categories such as “highly increased” or “slightly increased”. Furthermore, using other stationary variables such as return rate can solve this issue directly, since we will be having continuous variables rather than manually creating discrete variables.

Moreover, more advanced classification methods, such as a hybrid model can be used for further testing. A two-stage fusion approach involving Support Vector Regression (SVR) in the first stage and Artificial Neural Network (ANN), Random Forest (RF), and SVR in the second stage is proposed by Chavan and Patil (2013) research when making stock market predictions, and their results show that two-stage hybrid models perform better than that of the single stage prediction models.

Appendix

Table 1: Overview of all selected variables

Type	Name	Source	Frequency
Target Variable	S&P 500 Index	Yahoo Finance	Daily
Features: Sentiment	CBOE Volatility Index (VIX index)	Yahoo Finance	Daily
Features: Macroeconomic Indicator	Gold price	Yahoo Finance	Daily
	Oil Price	Yahoo Finance	Daily
	ICE U.S. Dollar Index	Yahoo Finance	Daily
	Effective Federal Fund rate	Federal Reserve Bank of New York	Daily

Table 2: Correlation between each variable

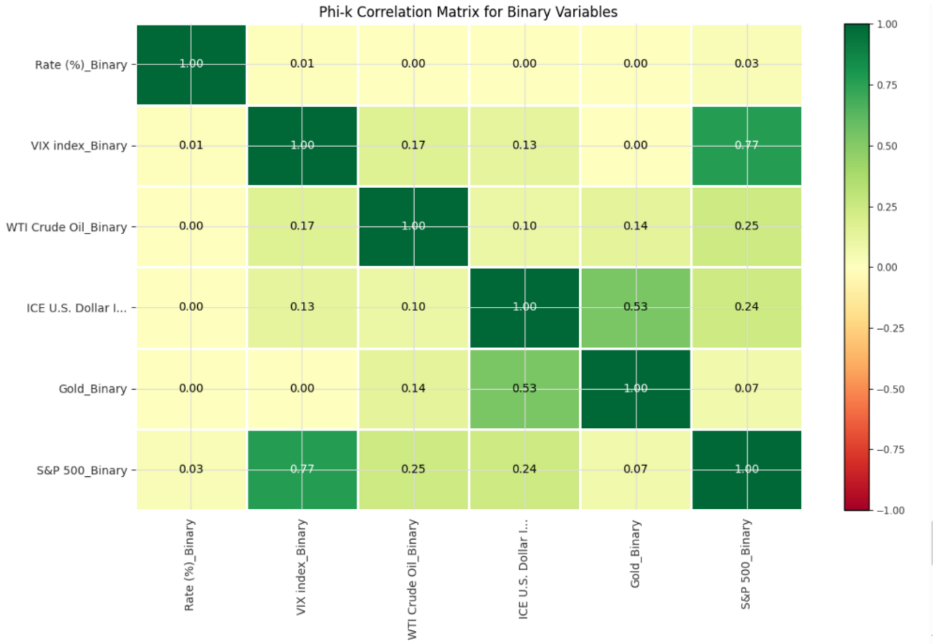

Table 3: Final result of each model

Models	Parameter Settings	Accuracy	Precision	Recall	F1-Score	AUC
Logit Regression	{'C': 1}	0.71	0.77	0.71	0.72	0.8195
Naive Bayes	{'alpha': 2.5}	0.74	0.76	0.74	0.75	0.8195
LDA	{'shrinkage': 'auto', 'solver': 'eigen'}	0.71	0.77	0.71	0.72	0.8081
SVM	{'C': 0.1, 'degree': 2, 'gamma': 'auto', 'kernel': 'sigmoid'}	0.70	0.76	0.70	0.71	0.7153
Decision Tree	{'max_depth': 10, 'min_samples_leaf': 20, 'min_samples_split': 10}	0.71	0.77	0.71	0.72	0.7962
K-Nearest Neighbors	{'algorithm': 'auto', 'leaf_size': 1, 'metric': 'hamming', 'n_neighbors': 24, 'p': 1, 'weights': 'uniform'}	0.70	0.76	0.70	0.71	0.7150
LSTM	{'batch_size': 32, 'hidden_dim': 64, 'dropout_rate': 0.05, 'activation': 'tanh', 'optimizer': 'adam', 'learning_rate': 0.001}	0.72	0.77	0.72	0.73	0.7960

References

Basak, S., Kar, S., Saha, S., Khaidem, L., & Dey, S. R. (2019). Predicting the direction of stock market prices using tree-based classifiers. The North American Journal of Economics and Finance, 47, 552-567.

Bhandari, H. N., Rimal, B., Pokhrel, N. R., Rimal, R., Dahal, K. R., & Khatri, R. K. (2022). Predicting stock market index using LSTM. Machine Learning with Applications, 9, 100320.

Breinlich, H., Leromain, E., Novy, D., Sampson, T., & Usman, A. (2018). The Economic Effects of Brexit – Evidence from the Stock Market. Centre for Economic Performance. https://cep.lse.ac.uk/pubs/download/dp1570.pdf

Bu, Q., & Forrest, J. (2021). Comparing sentiment measures in mutual fund performance. International Journal of Managerial Finance, 17(3), 478-493.

Campisi, G., Muzzioli, S., & De Baets, B. (2024). A comparison of machine learning methods for predicting the direction of the us stock market on the basis of volatility indices. International Journal of Forecasting, 40(3), 869-880.

Chatigny, P., Wang, S., Patenaude, J. M., & Oreshkin, B. N. (2021). Neural forecasting at scale. arXiv preprint arXiv:2109.09705.

Chavan, P. S., & Patil, S. T. (2013). Parameters for stock market prediction. International Journal of Computer Technology and Applications, 4(2), 337.

Das, S. P., & Padhy, S. (2012). Support vector machines for prediction of futures prices in Indian stock market. International Journal of Computer Applications, 41(3).

E. F. Fama, “The behavior of stock-market prices,” The journalofBusiness,vol.38,no.1,pp.34–105,1965.

Jiao, Y., & Jakubowicz, J. (2017, December). Predicting stock movement direction with machine learning: An extensive study on S&P 500 stocks. In 2017 IEEE International Conference on Big Data (Big Data) (pp. 4705-4713). IEEE.

Jigar Patel, Sahil Shah, Priyank Thakkar, K Kotecha. Predicting stock market index using fusion of machine learning techniques. Expert Systems with Applications. Volume 42. Issue 4. 2015. Pages 2162-2172. ISSN 0957-4174.

Johansson, W. (2024). Markets vs. Machines: An investigation of prediction market forecasting accuracy of economic variables compared to time series forecasting methodologies.

Khan, W., Malik, U., Ghazanfar, M.A. et al. Predicting stock market trends using machine learning algorithms via public sentiment and political situation analysis. Soft Comput 24, 11019–11043 (2020).

Lakshmi, V., Harika, K., Bavishya, H., & Harsha, C. S. (2017). Sentiment analysis of twitter data. Int Res J Eng Technol, 4(2), 2224-2227.

Maknickienė, N., & Vaškevičiūtė, A. (2017). Comparison of sentiments data extraction and prediction. Innovative Infotechnologies for Science, Business and Education, 1(22), 14-20.

Ou, P., & Wang, H. (2009). Prediction of stock market index movement by ten data mining techniques. Modern Applied Science, 3(12), 28-42.

Patel, J., Shah, S., Thakkar, P., & Kotecha, K. (2015). Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. Expert Systems with Applications, 42(1), 259-268.

Qian, B., & Rasheed, K. (2007). Stock market prediction with multiple classifiers. Applied Intelligence, 26, 25-33.

R. P. Schumaker and H. Chen, "A Discrete Stock Price Prediction Engine Based on Financial News," in Computer, vol. 43, no. 1, pp. 51-56, Jan. 2010, doi: 10.1109/MC.2010.2.

Schumaker, R. P., & Chen, H. (2009). Textual analysis of stock market prediction using breaking financial news: The AZFin text system. ACM Transactions on Information Systems (TOIS), 27(2), 12.

Shah, D., Campbell, W., & Zulkernine, F. H. (2018, December). A comparative study of LSTM and DNN for stock market forecasting. In 2018 IEEE international conference on big data (big data) (pp. 4148-4155). IEEE.

Thawornwong S, Enke D (2004) The adaptive selection of financial and economic variables for use with artificial neural networks. Neurocomputing 56:205–232

Yeh, C. Y., Huang, C. W., & Lee, S. J. (2011). A multiple-kernel support vector regression approach for stock market price forecasting. Expert Systems with Applications, 38(3), 2177-2186.