Dean&Francis
ISSN 2959-6130

# An Empirical Study Based on Machine Learning and LSTM in Stock Prediction

**Zhongyu Wang**

King's College London, London, United Kingdom
ariahu@gmail.com

**Abstract:**

This research aims to make a comprehensive comparative analysis between traditional machine learning methods and Long Short-Term Memory networks (LSTMs) towards stock return predictions. By using the full Kaggle market data set (the Stock Price Prediction Challenge), we managed to generate an integrated forecasting pipeline for stock prediction. In this data set, we used 45 stocks and three major indexes to engineer extensive features and a strong model validation. After considering all factors, Gradient Boosting, outperforming both traditional methods and LSTM, achieves the greatest training performance: Mean Squared Error (MSE) of 0.000135, R2 of 0.027195, and mean absolute percentage error (MAPE) of 155.85%. Contrary to the initial assumption, all models exhibited severe overfitting. A significant performance drop on the validation set suggests a major challenge in practical prediction use. The findings indicate that while the models do not provide practically useful, accurate return predictions based solely on price information, they do provide strong comparison benchmarks and methodological suggestions in future studies involving other data and stronger regularisation approaches.

**Keywords:** LSTM; stock prediction; Machine Learning

## 1. Introduction

Profit and better forecasting ability are always what a researcher or a fund manager aims for during their career. Predicting stock prices remains a significant challenge in financial analytics, primarily due to the volatility, noise, and non-stationarity of financial data [1]. Traditional statistical methods have been used for decades to evaluate time-series data and make future predictions. Most traditional statistical methods would not be considered valuable forecasters of time series data in the face of complex, non-linear relationships that are commonly present in financial markets. However, due to advancements in computing power, data mining, and the implementation of machine learning (ML/AI) algorithms, ML methods offer viable and competitive options to more traditional techniques based on the ability to detect patterns or identify relationships from historical data [2, 3]. More recently, deep learning models, particularly Long Short-Term Memory (LSTM) networks, have established working performance standards with many sequence prediction problems primarily due to their capability to model long-range temporal depen-

ISSN 2959-6130

dencies [4].

This research employs the Kaggle: Stock Price Prediction Challenge dataset, which consists of three significant indices (the Dow Jones, NASDAQ, and S&P 500) as well as 45 other stocks (AAPL and CSCO), to analyze and compare stock-price predictions using traditional machine learning models and LSTM networks. After conducting EDA, feature engineering, and training the features on multiple training protocols, we found that the LSTM outperformed traditional models with a training MSE of 0.000120 and validation MSE of 0.000106; these results suggest that the LSTM's focus on prediction of returns resulted in a superior ability to capture temporal dependencies.

Classical time-series models (ARIMA and GARCH) capture linear dependencies and volatility clustering of stock return time-series, but struggle with modeling nonlinearity, nonstationarity, and fail to capture dynamic dependencies of financial data. Ensemble machine learning approaches (Random Forest, Gradient Boosting) often do provide an efficient way to model non-linear patterns of data, but generally don't appropriately account for the temporal ordering of those features. Advanced deep learning techniques, especially LTSM, are a powerful method as they attempt to capture dynamic dependencies through gating and memory cells that learn long-range dependencies, and have been shown in the last decade or so to be a leading methodology in predicting financial time series. There is also support in the empirical financial literature in favor of using LSTM architecture models, such as Fischer & Krauss, who showed that LSTM prediction performance improved by 6–8% when compared to Random Forest or GBM on 112 S&P 500 stocks [5]. Qiu et al. in their hybrid CNN——LSTM architecture suggested that such complex models can enhance the reduction in prediction error of highly volatile stocks [6].

The advanced evolution of deep learning has led to the proliferation of recurrent architectures, notably long short-term memory (LSTM) architectures, as a tool for time-series forecasting, since these architectures have gates and memory cells that can adequately capture long-range dependencies [3]. Quantified empirical evidence supports their superiority in predictive performance: Fischer & Krauss (2018) found that LSTMs produced around 6–8% better predictive performance than Random Forests and GBM across an empirical cross-section of 112 S&P 500 stocks. Hybrid designs that incorporate convolutional layers alongside LSTMs have been shown to lower error even further—for instance, they reported reductions in mean squared error on volatile assets in the range of about 10–15% depending on the dataset and related experimentation (i.e., 0.0021 to 0.0018, ≈14%) [6]. This study extends on those existing findings by employing a larger multi-index (and multi-stock) dataset to provide further tests on LSTM robustness and generalizability.

Based on these previous studies, this project will utilize a vast dataset that includes financial data of major indices and a wide selection of individual stocks. We hope this research will not only continue to help validate and expand prior research in this area, but will also provide more data on the strengths of the LSTM specifically for predictions of time-series data from financial markets.

## 2. Methodology

### 2.1 Data Collection and Pre-processing

The dataset was obtained from Kaggle's Stock Price Prediction Challenge, which included historical daily price data for 3 market indices (Dow Jones Industrial Average, NASDAQ Composite, and S&P 500) and 45 individual stocks (such as AAPL, CSCO, and MSFT) [7]. Each record contained opening, high, low, and closing prices, trading volume, and the calculated daily returns spanning ~10 years of daily trading data. Data preprocessing included a number of steps to ensure the quality and consistency of the data. In order to create temporal consistency, all date fields had to be converted to datetime objects. No missing data is detected.

### 2.2 Exploratory Data Analysis and Visualization

To comprehensively explore the data, we also conduct the descriptive analysis (mean, standard deviation, max, and min values) for key financial metrics across all stocks and indexes. And the distribution analysis is used for statistical properties of daily returns, revealing characteristic leptokurtic distributions with fat tails typical of financial time series.

We used price trend plots to compare individual stock performance against three indexes, histograms and density plots of return distributions to capture volatility, and correlation matrices to identify relationships among momentum, various price metrics, and volume data. Lastly, the implementation of Matplotlib and Seaborn libraries is to visualize the analysis results.

### 2.3 Feature Engineering

This study adopts a comprehensive feature pipeline to take the raw data into the prediction result. The feature set consisted of four categories:

Price-based features included daily price change (Close - Open), high-low spread, and close-open ratio. Technical indicators were calculated, such as moving averages (5, 10, 20, and 50-day averages) and prices to moving average ratios. Volatility measures included the rolling standard deviations of returns in multiple windows (5, 10,

20 days). Market-based features were combined by amalgamating index data (closing prices, returns, and volumes) and individual stock data, and generalizing index-based moving averages.

For the training data, the target variables were the multi-day forward returns (1-5 days). The final feature set consisted of 48 features for the training data and 42 features for the test data, while having a set of common features paired to ensure consistency between training and prediction.

### 2.4 Model Architecture and Training

The study implemented and compared two categories of predictive models:

Traditional machine-learning baselines and configuration: This study assessed a number of traditional regression and ensemble methods for benchmarking. These include Ordinary Least Squares (Linear Regression), Ridge Regression (L2 regularization, $\alpha = 1.0$), Lasso Regression (L1 regularization, $\alpha = 0.01$), Random Forest (n of estimators = 100, max of depth = 10), and Gradient Boosting (n of estimators = 100, max of depth = 5). The linear, Ridge, and Lasso predictors are interpretable linear models, with Ridge and Lasso discouraging overfitting through L2 or L1 penalties, respectively, and Lasso also producing large coefficients that are sparse or composed of "zero" values. Random Forests will reduce variance by aggregating and bootstrapping decision trees and identifying nonlinear feature interactions, while Gradient Boosting minimizes residual error sequentially to capture complex nonlinear behaviours, increasing sensitivity to hyperparameters.

Deep learning architecture and rationale:

The deep model is a multi-step horizon forecasting long short-term memory (LSTM) stacked network. The input sequences consisted of ten previous consecutive trading days, each with 41 features at each time step. The architecture consisted of a LSTM layer with 64 units with (return of sequences=True), followed by Dropout (rate=0.20) and Batch Normalization, a second LSTM layer with 32 units, a second Dropout (rate=0.20) and Batch Normalization, a fully connected hidden layer with 16 units with ReLU activation, and a linear output layer with 5 units corresponding to the 1-5 day-ahead predictions. LSTMs utilize gated memory cells, which allow for learning of long-range dependencies and reducing vanishing/exploding gradients that recurrent networks are known for; dropout and batch normalization as regularization techniques help with generalizing the model and stabilizing training.

Training protocol and preprocessing: The models underwent assessment via time-series cross-validation with three temporal splits that maintained temporal dependence. For the traditional models, the input features were standardized via StandardScaler. The LSTM was trained on sequences of 10 continuous trading days, which were normalized (mean=0; standard deviation=1) before training. The training utilized the Adam optimizer, batch size of 64, and up to 20 epochs with early stopping (patience = 5) based on validation loss to prevent overfitting the model.

### 2.5 Evaluation Metrics and Validation

The model performance leverages three standard metrics: Mean Squared Error (MSE), with no primary loss function; Mean Absolute Error (MAE), to interpret the average prediction error; and the $R^2$ coefficient to explain the explanatory power of the prediction. The predictions were evaluated on a held-out validation set containing the most recent 20% of temporal data.

The final analyses included full visualizations of the results involving: (1) Time-series plots of the actual and predicted returns; (2) Scatter plots of the predicted values and actual values with the ideal fitted line; and (3) Distribution of prediction errors for various time horizons. These visualizations provided finer detail to the model's performance in addition to the cumulative metrics.

### 2.6 Prediction Generation

The best-performing model created predictions for five test sets using the following process: feature extraction and standardization (using the same scaler that was fitted on the training data), sequence preparation of the data for the LSTM models, batch prediction, and organization of results into submission-ready CSV files with dates, in this instance, 1-5 day return predictions. After interpreting how the Implementation performed, it was clear that the Implementation could produce reproducible results, since every random number used in the implementation was fixed, and all models, scalers, and results were systematically stored.

## 3 Results

### 3.1 Overall Model Performance Comparison

To assess five different models' prediction abilities, this study examined five traditional machine learning models and one LSTM. The performance comparison across all models is shown in Table 1, presenting significant variation in accuracy and validity. We evaluated five traditional machine learning models and one LSTM neural network using multiple performance metrics to comprehensively assess their predictive capabilities. Table 1 presents the detailed performance comparison across all models on the training dataset, revealing significant variations in predictive accuracy and reliability.

ISSN 2959-6130

**Table 1. Model Performance Comparison (Training Set)**

| Model | MSE | R2 | MAPE |
|---|---|---|---|
| Linear Regression | 0.000138 | -0.001407 | 144.74% |
| Ridge Regression | 0.000138 | -0.001133 | 145.28% |
| Lasso Regression | 0.000138 | -0.000031 | 132.26% |
| Random Forest | 0.000138 | 0.002233 | 130.65% |
| Gradient Boosting | 0.000135 | 0.027195 | 155.85% |
| LSTM | 0.000120 | -0.164276 | 147.84% |

In contrast to expectations, the Gradient Boosting achieved the best predictive performance with the lowest MSE (0.000135), highest R2 (0.027195), and MAPE (155.85%). It showed a consistent marginal improvement among traditional machine learning models, even though the improvement may seem modest. In addition, the LSTM model conversely underperformed with a worse predictive ability (MSE of 0,000138 and a negative R2 of -0.164276), indicating a worse predictive ability than the simple mean predictor.

## 3.2 Validation Performance and Model Generalization

Even though the Gradient Boosting achieved the best performance among all models, it still reflects an overfitting problem on the validation set (Table 2):

**Table 2. Validation Set Performance (Gradient Boosting)**

| Metric | Value | Interpretation |
|---|---|---|
| MSE | 0.000336 | 2.5× training error |
| MAE | 0.012647 | Substantial absolute error |
| R2 | -2.231488 | Severe underperformance vs the mean predictor |
| MAPE | 680.51% | Extremely high percentage error |

On the validation set, the MSE increases by 150% over training performance. In addition, there is a strongly negative $R^2$ score (-2.23) and extremely high MAPE of 680.51%, showing the concerning generalization problem in capturing real new incoming data through the modeling process, and provides predictions. Further studies revolving financial data rather than pure historical return and price data should be considered.

## 3.3 Comparative Analysis of Model Characteristics

According to the evaluation metrics, we found that the traditional models exhibited homogeneous performance on MSE and a nuanced difference in predictive accuracy across several approaches. The range of MSE is from 0.000138 to 0.000139, compared to $R^2$ values with more variation. $R^2$ of Gradient Boosting(0.027) exhibited a substantially positive value, while Random Forest is 0.002233, and other models are all negative. This means this model is better at variance explanation than others. The MAPE ranges from 130.65% (Random Forest) to 155.85% (Gradient Boosting), suggesting the difficulty of stock prediction accuracy.

Contrary to initial expectations, LSTM performed poorly, which was particularly noteworthy. A strong negative $R^2$(-0.164) indicates that a theoretically better fit for time series data may not be meaningful in a specific application. Lacking sufficient training points, the stock volatility nature, and noise may weaken the performance of deep learning in practical cases.

## 3.4 Error Analysis and Practical Implications

The significant drop of performance from training to validation set in MSE (0.000135 to 0.000336), R2(from positive to negative), and MAPE (155.85% to 680.51%) emphasizes the significant value of emphasizing validation in financial forecasting cases. There is a clear pattern of overfitting in the models, indicating that while the models can draw certain insights into observed empirical patterns in the training data, these patterns may not replicate out of sample, as indicated by the efficient market hypothesis.

Nevertheless, there remains value in the relative performance across the timeline, and despite limitations in the architecture, tree-based ensemble models (Gradient Boosting and Random Forest) do indicate an incremental improvement in predictability against those of the linear models, as well of to the LSTM employed in this scenario, for this dataset and prediction task.

## 4 Conclusion

This research has realistically evaluated machine learning methods for predicting stock returns. While originally expecting traditional models and LSTM networks to perform better, in fact, gradient boosting outperformed both methods, yielding the best results on training data (MSE = 0.000135, $R^2 = 0.027$).

More importantly, all models demonstrated dramatic overfitting, which is a major finding in this research. Model performance results were dramatically worse than in the training phase, with MSE increasing by 150% and MAPE increasing to 680.51%. The $R^2$ of -2.23 demonstrates that, despite being trained for predictive purposes, each model performed worse than using a mean value of the training data to predict stock returns. These results demonstrate that price history alone is not sufficient to predict because the models would easily learn noise rather than real, meaningful algorithms and relationships regardless of the model selection. Meanwhile, a proper validation and conservative methodological expectation for machine learning in financial prediction should be prioritized before practical implementation.

Further studies should consider other categorizations of data and ensemble methods to offset the overfitting effect. Although precise stock return prediction was not achieved, this study will provide benchmarks and methodological directions for ongoing work in this difficult area of research.

## References

[1] Bollerslev T. Generalized autoregressive conditional heteroskedasticity. Journal of Econometrics, 1986, 31(3): 307–327.

[2] Breiman L. Random forests. Machine Learning, 2001, 45(1): 5–32.

[3] Friedman J H. Greedy function approximation: A gradient boosting machine. Annals of Statistics, 2001, 29(5): 1189–1232.

[4] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Computation, 1997, 9(8): 1735–1780.

[5] Fischer T, Krauss C. Deep learning with long short-term memory networks for financial market predictions. European Journal of Operational Research, 2018, 270(2): 654–669.

[6] Qiu J, Wang B, Zhou C. Forecasting stock prices with long-short term memory neural network based on attention mechanism. PLOS ONE, 2016, 15(1): e0227222.

[7] Manaenkov N. Stock Price Prediction Challenge. Kaggle, 2025. Kaggle. https://kaggle.com/competitions/stock-price-prediction-challenge