Volume XXIX 2026
MBNA Publishing House Constanta 2026

Scientific Bulletin of Naval Academy

SBNA PAPER • OPEN ACCESS

Forecasting of Stock Market Trends Using LSTM Neural Networks

To cite this article: Alexandru-Ionel Toma, Scientific Bulletin of Naval Academy, Vol. XXIX 2026, pg. 102-109.

Submitted: 30.06.2026
Revised: 31.07.2026
Accepted: 04.08.2026

Available online at www.anmb.ro

ISSN: 2392-8956; ISSN-L: 1454-864X

doi: 10.21279/1454-864X-26-I1-010

Forecasting of Stock Market Trends Using LSTM Neural Networks

Alexandru-Ionel Toma

National University of Science and Technology POLITEHNICA Bucharest, Faculty of Automation and Computers, Department of Automation and Industrial Informatics, Bucharest, Romania
e-mail: toma.alexandru87@yahoo.com

Abstract. One could argue that electronic trading in the financial world was well on its way to development long before the Internet came into being. J. Christopher Westland and Theodore H. Clark note in their work, Global Electronic Commerce: Theory and Case Studies, that the U.S. security markets of the 19th century were the first to see this kind of modernization, the New York Stock Exchange among them. When it comes to making sense of today's complicated financial markets, the statistical frameworks that underpin traditional analysis are not always up to the task. That is why there has been a move toward high-tech solutions like neural networks and machine learning; their popularity is only growing by the day. In fact, as Ethem Alpaydin points out in Machine Learning: The New AI, these methods are now common in fields as diverse as healthcare and the auto industry. Not to mention finance, where they are routinely used to forecast the future from historical data. The research will show how effective and accurate these tools are at gauging price movements or spotting market trends.

Keywords: Machine Learning, Stock Market, Neural Networks, Financial Forecasting, Algorithmic Trading, Predictive Analytics.

1. Introduction

The stock market's inherent unpredictability makes forecasting its prices a focus of financial research. When investors can form an accurate picture of where a stock is headed, they are better positioned to make trading calls, keep risk in check, and put their portfolios in optimal shape. While older statistical approaches have their place, they tend to fall short in capturing the nuances of stock price behavior. As a result, there has been a move toward machine learning (ML) and deep learning (DL) to get better predictive results. Yahoo Finance was chosen as the primary source for this project. It is a go-to for anyone doing financial analysis or building ML models for prediction because of the breadth of its data, from trading volume and trends to the prices themselves. The application is designed to be flexible; as long as the stock price data is publicly available on Yahoo Finance, it works. That means the application can handle listings from any number of global exchanges, such as the NYSE or NASDAQ in the U.S., the LSE and Euronext in Europe, or the TSE, SSE, and HKEX in Asia.

2. Materials and methods

In putting together the technology and platform for our stock market forecasting system, several considerations needed to be balanced. The aim was to build something that is not only accurate and

efficient but also easy to use, all while leaving room for down-the-line improvements. To that end, the stack put in place is a mix of modern machine learning frameworks, proven software development tools, and a dependable source for financial data, underpinning the system's ability to forecast, evaluate, and visualize.

Python is the language of choice. It has become ubiquitous in scientific computing and financial analysis, and with its straightforward syntax and vast AI ecosystem, it is hard to beat for rapid prototyping and research work.

The analyzed stock symbols were BAC (Bank of America Corporation); META (Meta Platforms); AAPL (Apple Inc.); AMZN (Amazon.com Inc.); TSLA (Tesla Inc.).

The forecasting task focused on predicting the next trading day closing price using historical financial information.

2.1. Software Environment

The forecasting system was implemented using Python due to its extensive support for machine learning, scientific computing, and financial analysis (Alpaydin, 2016). The development process relied on several specialized software libraries and frameworks.

TensorFlow and Keras were used for implementing and training the Long Short-Term Memory (LSTM) neural network. These frameworks provide efficient support for deep learning architectures and time-series forecasting applications (Schmidhuber, 2015).

Pandas and NumPy were used for financial data preprocessing and manipulation, while Matplotlib was used to generate graphical representations of stock market trends and prediction results.

The yfinance library was used to retrieve historical stock market data directly from Yahoo Finance (Yahoo Finance, 2026). The retrieved datasets included daily stock prices, historical trends, and trading information.

Scikit-learn was used for preprocessing operations and evaluation metrics. In addition, the MinMaxScaler normalization method was applied to improve neural network convergence during training.

2.2. Forecasting Models

Several forecasting approaches were analyzed before selecting the final prediction model. Traditional machine learning algorithms such as Random Forest and XGBoost are commonly used for structured financial data analysis and stock market prediction tasks (Breiman, 2001; Chen & Guestrin, 2016).

Time-series forecasting models such as Facebook Prophet are also frequently applied for trend and seasonality analysis in financial datasets (Taylor & Letham, 2018). However, these methods are generally less effective in modeling long-term sequential dependencies and highly nonlinear financial patterns. A comparison between the five analyzed models for this study is presented in Table 1.

Criteria	Facebook Prophet	XGBoost	Random Forest (RF)	LSTM	CNN + LSTM Hybrid
Type	Time-series forecasting	Ensemble Learning	Ensemble Learning	Deep Learning (RNN-based)	Deep Learning (Hybrid)
Captures Trends & Seasonality?	Yes	No	No	Yes	Yes
Handles Short-Term Movements?	No	Yes	Yes	Yes	Yes

Handles Long-Term Trends?	Yes	No	No	Yes	Yes
Computational Cost	Medium	Low	Low	High	Very High
Prediction Speed	Fast	Fast	Fast	Medium	Slow
Overfitting Risk	Low	High	High	Medium	High
Works Well with Small Data?	Yes	Yes	Yes	No	No
Works Well with Large Data?	Medium	Yes	Yes	Yes	Yes
Best for Algorithmic Trading?	No	Yes	Yes	Yes	Yes
Handles Market Volatility?	No	Yes	Yes	Yes	Yes
Explainability	High	Medium	Medium	Low	Low
Accuracy in Real Market Conditions	Medium	Medium	Medium	High	Very High

Table 1. Comparison between the five models

The proposed forecasting system was based on Long Short-Term Memory (LSTM) neural networks, originally introduced for learning long-term dependencies in sequential data (Hochreiter & Schmidhuber, 1997). Recent studies demonstrated that LSTM-based deep learning architecture achieves strong performance in time-series forecasting applications (Hua et al., 2019).

LSTM models are particularly suitable for stock market forecasting because they can retain historical information through memory cells and gating mechanisms, allowing the network to capture temporal dependencies in financial time-series data. The internal structure of the LSTM architecture is presented in Figure 1.

The implemented architecture consisted of sequential input layers; multiple LSTM layers; dense neural network layers; output prediction layers.

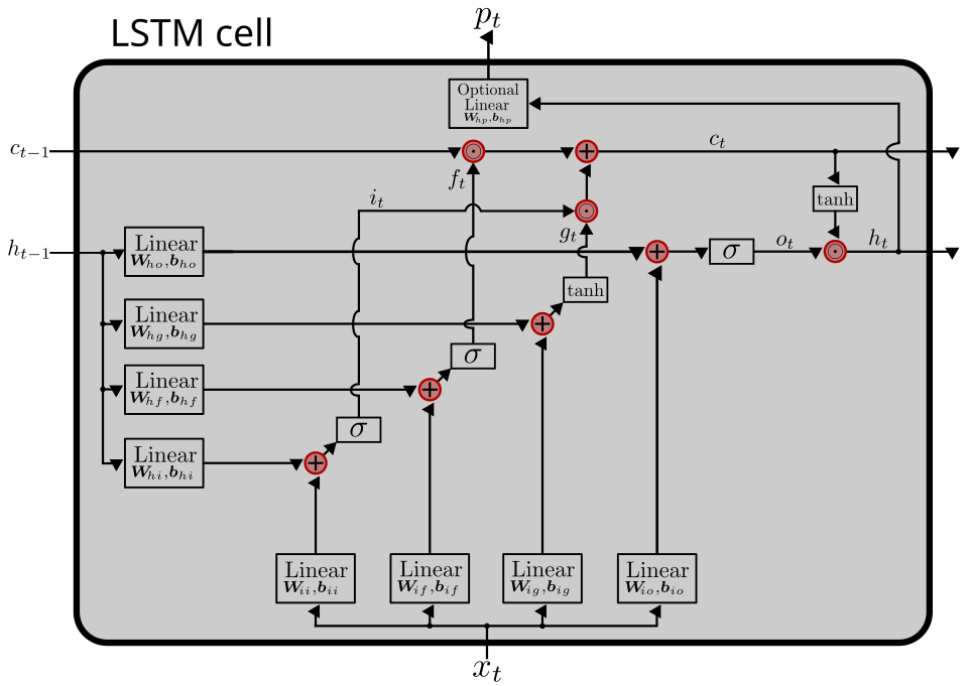

Figure 1. Representation of LSTM. Source: Wikipedia (2026).

2.3. Data Collection

Historical stock market data was collected directly from Yahoo Finance using the yfinance Python library (Yahoo Finance, 2026). The dataset included: daily opening prices; daily closing prices; stock volume information; historical price movements.

The experiments used historical closing prices as the primary feature for prediction generation. The retrieved datasets were dynamically updated during runtime, ensuring that the forecasting process used recent market information.

2.4. Data Preprocessing

The collected financial datasets were preprocessed before training the forecasting model. Data preprocessing is a critical stage in machine learning systems because neural networks are highly sensitive to noisy or inconsistent input data (Alpaydin, 2016).

The preprocessing pipeline included: validation of stock symbols; removal of missing values; extraction of closing price data; normalization of numerical values; generation of sequential datasets.

Stock prices were normalized using the MinMaxScaler method from Scikit-learn to scale the numerical values into the interval [0,1]. This normalization process improved neural network convergence and reduced instability during training.

The sequential dataset was generated using rolling windows containing the previous 60 trading days. Each sequence was associated with the next trading day's closing price, representing the prediction target.

2.5. Model Training

The forecasting model was trained individually for each company using historical stock price sequences. The neural network processed sequential financial information and learned temporal dependencies through the internal memory structure of the LSTM architecture (Hochreiter & Schmidhuber, 1997).

The training process included: feeding sequential input data into the network; optimizing neural network weights; minimizing forecasting error through backpropagation.

Deep learning architecture has shown significant improvements in predictive accuracy for complex data analysis problems compared to traditional machine learning techniques (Schmidhuber, 2015).
The trained model generated predictions for the next trading day closing price. The normalized outputs were transformed back into real stock price values before evaluation.

2.6. Performance Evaluation

The forecasting performance of the proposed system was evaluated using several statistical indicators commonly applied in financial forecasting and machine learning research. The evaluation metrics included: Mean Absolute Error (MAE); Mean Squared Error (MSE); Root Mean Squared Error (RMSE); Coefficient of Determination (R^2); Explained Variance Score (EVS); Mean Absolute Percentage Error (MAPE); Symmetric Mean Absolute Percentage Error (sMAPE).
The predictions generated were compared against actual stock prices to evaluate forecasting accuracy, robustness, and generalization capability across different market sectors.
The execution time required for model training and prediction generation was also measured to evaluate the computational efficiency of the forecasting system.

2.7. Experimental Workflow

The experimental workflow of the proposed forecasting system consisted of the following stages: Retrieve historical stock market data from Yahoo Finance; Validate and preprocess financial datasets; Normalize stock price values; Generate sequential input windows; Train the LSTM neural network; Generate next-day stock price predictions; Evaluate forecast performance using statistical metrics.
The proposed methodology allowed the forecasting model to adapt dynamically to different stock market conditions while maintaining strong predictive performance across multiple companies and financial sectors. The justification summary is presented in Table 2.

Component	**Technology**	**Reason**
Programming Language	Python	Simplicity, large AI ecosystem, research-friendly
ML Framework	TensorFlow + Keras	LSTM support, GPU acceleration, ease of use
Data Source	yfinance	Real-time data, easy access
Data Processing	Pandas, NumPy	Efficient time-series manipulation
Visualization	Matplotlib/Plotly	Clear, customizable visualizations
UI	Tkinter	Rapid development, minimal configuration

Table 2. Justification summary

3. Results and discussions

The proposed LSTM forecasting model was tested on multiple publicly traded companies from different economic sectors, including banking, technology, e-commerce, and automotive industries. Historical stock market data was retrieved from Yahoo Finance and processed using Python-based data analysis libraries.
The experiments evaluated both prediction accuracy and computational performance. The forecasting performance was measured using several statistical indicators.

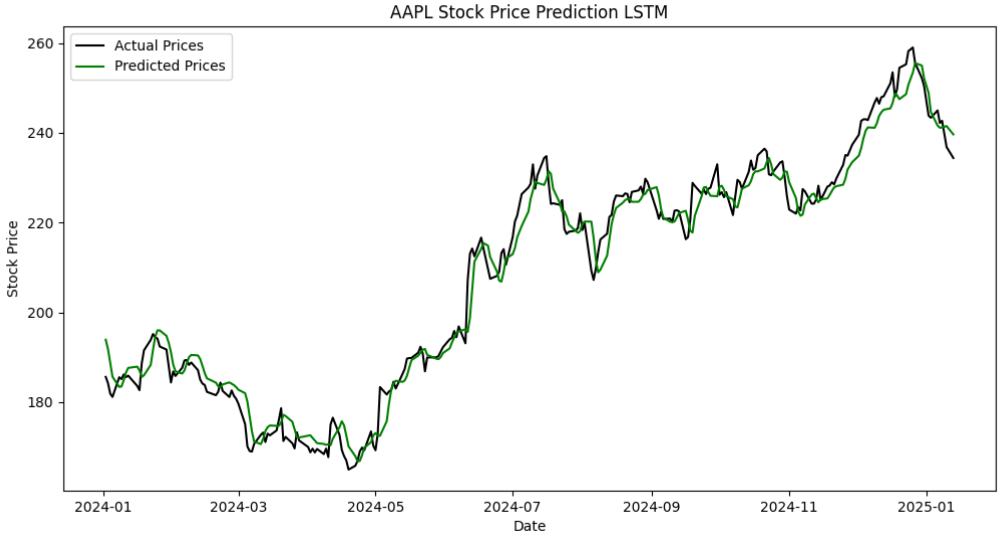

Figure 2. Actual AAPL share price compared to predicted share price

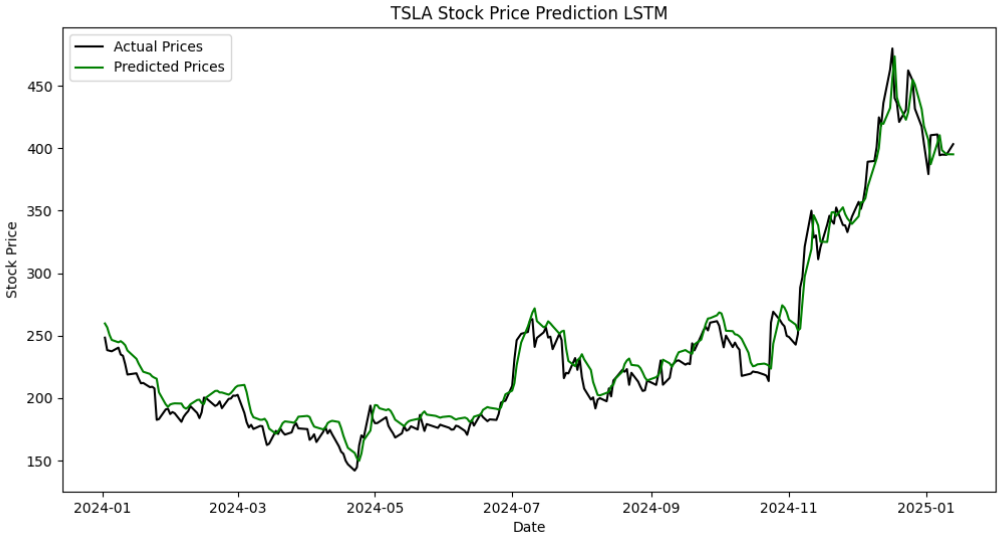

Figure 3. Actual TSLA share price compared to predicted share price

The experimental results demonstrated that the LSTM model achieved strong predictive performance across all tested companies. The predictions generated closely followed the actual stock price trends, indicating that the model successfully captured temporal dependencies in financial time-series data. Table 3 presents the forecasting performance obtained for each tested company.

Metric	BAC	META	AAPL	AMZN	TSLA
Execution Time (s)	148.12	149.18	155.25	153.83	164.03
MAE ($)	0.52	11.11	3.16	3.06	10.49
MSE ($)	0.49	221.39	15.75	17.11	167.68

RMSE ($)	0.70	14.88	3.97	4.14	12.95
R^2	0.97	0.94	0.98	0.95	0.97
EVS	0.97	0.95	0.98	0.96	0.98
MAPE (%)	1.33%	2.12%	1.53%	1.60%	4.67%
sMAPE (%)	1.33%	2.14%	1.53%	1.61%	4.56%

Table 3. Forecasting performance metrics obtained using the LSTM model

The results obtained indicate that the proposed forecasting model achieved high accuracy for most tested companies. The average R^2 score reached 0.962, while the average MAPE remained below 3%, demonstrating low prediction error and good model generalization capabilities. Table 4 presents the average performance metrics.

Metric	Average
R^2	0.962
MAPE (%)	2.250
sMAPE (%)	2.234
Time (s)	154.482

Table 4. The average performance metrics

The closeness of MAPE and sMAPE suggests the model does not suffer from directional bias (it doesn't consistently overpredict or underpredict).
The model completed training in about 2.5 minutes on average, which is efficient considering the 3-layer LSTM architecture. This makes the approach practical for iterative development and experimentation.
The experimental results confirm that LSTM neural networks are effective for stock market forecasting tasks due to their ability to capture long-term dependencies and nonlinear patterns in financial data. The model performed particularly well on relatively stable stocks such as BAC and AAPL, while higher volatility in TSLA resulted in slightly larger prediction errors.
There is no denying that the current version of the application does its job and yields honest out-of-sample results, but it is by no means without its share of room for improvement. Markets are not moved by price alone; news, earnings reports, interest rates, inflation, and the general mood of investors are at play, too. A later iteration might well incorporate fundamental and sentiment data from social media or financial news to give the model context that a straight look at price history can't provide.
Then there is the matter of forecasting. At present, the model will only tell you tomorrow's closing price. Most users would prefer something more in the way of a week or a month out, or perhaps a range of likely prices rather than a single number. And while the application now provides an exact prediction, it doesn't say how confident it is in it. If the system included confidence intervals or uncertainty estimates, the user would know when the model is fairly certain and when it is merely guessing, resulting in a prediction the user can trust.

4. Conclusions

The purpose of this article was to put together a market-prediction application that can forecast the next day's price for a publicly traded asset with the aid of deep learning, yet in a way that an everyday user could run it without needing to understand machine learning.

The intent was not to rely on an off-the-shelf model for the forecasting. For instance, the model does not go after the absolute price but rather the next day's return, the change in size and direction, before converting it back to a price. This is more numerically stable and prevents the model from simply echoing the current price.

In the end, the work bears out what is already known about the hard task of short-term price forecasting. A low percentage error is not in itself evidence of a useful model; more often than not, it is simply because prices do not move much from one day to the next. The measurements confirm this.

There are limitations to the application, but they also tell us where to go from here. The system does not factor in fundamental or sentiment data, let alone the wider market; the model is confined to an asset's own price and volume history. It was run on one machine and over thirty days, and the numbers will tell that nailing down daily direction is no easy matter.

All told, the project has done what it set out to do: it offers an honest read on the capabilities of such a model and a firm base for anyone who wants to do further research. In so doing, it was shown how deep learning can be applied to financial forecasting using a tool that is both complete and usable.

Funding: research received no external funding.

References

1. Alpaydin, E. (2016). Machine Learning: The New AI. MIT Press.
2. Breiman, L. (2001). Random Forests. Machine Learning, vol. 45(1), pp. 5-32.
3. Chen, T. and Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. San Francisco, USA, pp. 785-794.
4. Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation, vol. 9(8), pp. 1735-1780.
5. Hua, Y., Zhao, Z., Li, R., Chen, X., Liu, Z. and Zhang, H. (2019). Deep Learning with Long Short-Term Memory for Time Series Prediction. IEEE Communications Magazine, vol. 57(6), pp. 114-119.
6. Schmidhuber, J. (2015). Deep Learning in Neural Networks: An Overview. Neural Networks, vol. 61, pp. 85-117.
7. Taylor, J. and Letham, B. (2018). Forecasting at Scale. The American Statistician, vol. 72(1), pp. 37-45.
8. Yahoo Finance. (2026). Historical Stock Market Data. [Online]. Available: https://finance.yahoo.com/
9. Wikipedia. (2026). Long Short-Term Memory. [Online]. Available: https://en.wikipedia.org/wiki/Long_short-term_memory