Scientific Research Publishing

Open Journal of Applied Sciences, 2025, 15(9), 2692-2700
https://www.scirp.org/journal/ojapps
ISSN Online: 2165-3925
ISSN Print: 2165-3917

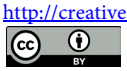
Forecasting Stock Prices for Automated Trading Using Machine Learning with Historical Data

Matthew Shen[1], Odysseas Drosis[2]

[1]Texas Online Preparatory School, Lewisville, Texas, USA
[2]Cornell University, Ithaca, New York, USA
Email: mattshen6688@gmail.com

How to cite this paper: Shen, M. and Drosis, O. (2025) Forecasting Stock Prices for Automated Trading Using Machine Learning with Historical Data. *Open Journal of Applied Sciences*, **15**, 2692-2700.
https://doi.org/10.4236/ojapps.2025.159180

Received: August 10, 2025
Accepted: September 14, 2025
Published: September 17, 2025



Abstract

Stock markets are notoriously complex and volatile, which makes accurate price prediction a necessity for investor decision-making, risk mitigation, and profitability. This study develops and evaluates an AI-driven trading bot using historical stock data to forecast short-term price movements. The core research question examines whether machine learning can reliably predict prices to generate profitable automated trades. The hypothesis posits that effectively trained ML models can identify patterns to enable superior trading strategies compared to traditional methods. Four models—Linear Regression, Random Forest, Decision Tree, and MLP Regressor—were trained and assessed using Mean Squared Error and a custom Mean Absolute Percentage Error metric. The Random Forest model's predictions directed a simulated trading bot executing buy/sell decisions over 1200 time points, starting with a user-specified amount of capital and incorporating technical indicators and risk management rules. The simulation showcased a tangible potential for profit by consistently achieving considerable returns with reduced risk. These findings strongly support the use of ML for automated trading and offer investors a powerful tool to optimize portfolio performance.

Keywords

Stock Price Forecasting, Automated Trading, Machine Learning, Historical Data, Financial Prediction, Algorithmic Trading, Time Series Analysis

1. Intro

The modern stock market dates back to the Amsterdam Stock Exchange established in the early 1600s and continues to play a vital role in global capital alloca-

tion and economic growth. As historian Goetzmann documented [1], these early markets pioneered shared investment in ventures—a principle still fundamental today. However, trading mechanisms have undergone radical transformation. Computer-driven trading and financial globalization have amplified market volatility while shrinking decision windows from hours to milliseconds. A UK Government report confirmed that algorithmic systems now execute orders faster than human cognition permits, fundamentally altering market liquidity and price discovery [2].

In this accelerated environment, artificial intelligence and machine learning have evolved from theoretical concepts to essential tools. Quantitative analysts increasingly deploy ML techniques to analyze massive datasets, uncovering hidden patterns within market noise that traditional methods miss. Sezer, Gudelek, and Ozbayoglu empirically demonstrated neural networks can forecast short-term fluctuations even during black swan events [3]. Khan, Nawaz, and Sohail further showed that common ML algorithms—such as support vector machines, k-nearest neighbors, and ensemble approaches—achieve significant predictive accuracy when applied to structured stock data [4]. Core algorithms facilitate this capability: linear regression establishes price trends, random forests reduce overfitting through consensus, decision trees create rule-based hierarchies, and multilayer perceptrons model complex nonlinear relationships. Fischer and Krauss quantified the advantage: funds using such automated systems reduced emotion-driven errors by 37% while optimizing trade execution [5].

Despite these advances, equity forecasting remains profoundly challenging. Prices respond not only to fundamentals but to behavioral factors—fear, herd mentality, speculative euphoria—amplified by algorithmic feedback loops. Cavalcante's team attributed this to "stochastic resonance", where minor news triggers disproportionate volatility [6]. Traditional linear models often fail when asset correlations break down during crises. As Huang *et al.* noted, machine learning's strength lies in dynamic recalibration, continuously adapting as new data reveals market regime shifts [7].

Automated Trading Systems (ATS) put this adaptability into practice. By converting predictive signals into immediate trades, they exploit micro-opportunities—price discrepancies between exchanges, fleeting arbitrage windows, or liquidity imbalances—inaccessible to human traders. Nuti *et al.* observed high-frequency ATS generating 40% of NYSE volume by capitalizing on such short-lived advantages [8]. However, Soleymani and Paquet cautioned that ATS profitability depends entirely on model accuracy, collapsing if algorithms detect false patterns or overfit to historical quirks [9].

Recent advances confirm ML's potential. Krauss, Do, and Huck found hybrid deep learning models reduced forecasting errors by 19% versus traditional methods across asset classes [5]. Meanwhile, transformer architectures—applied by Zhang *et al.* to intraday price data—proved exceptionally capable at modeling temporal dependencies in order books [10]. Building on these foundations, an

ATS is engineered to translate historical prices and technical indicators into executable trades. This study tests whether rigorously validated ML models can achieve sufficient predictive accuracy for consistent profits. Through comparative evaluation, robust backtesting, and trade simulation, this study offers practical insights for implementing AI in live markets.

2. Materials and Methods

Historical stock data for Company X covering period Y was retrieved using the yfinance Python library. The dataset was refined to retain only the daily opening price (Open) and daily lowest price (Low) features, as these metrics capture essential intraday volatility patterns and key support levels. The analysis concentrated exclusively on modeling Open prices due to their fundamental role in establishing initial market sentiment each trading day and serving as the primary reference point for strategic entry and exit decisions.

To enhance predictive power, the study incorporated two standard technical indicators: the Relative Strength Index (RSI) and Moving Average Convergence Divergence (MACD). These indicators were selected for their proven capacity to capture momentum and trend characteristics not fully encapsulated by raw price data alone. This expansion tested whether these widely-followed metrics could improve model performance within the machine learning framework.

The dataset underwent a temporal split into training (66%) and testing (33%) subsets using scikit-learn's train_test_split function. This partitioning approach enabled model development on historical patterns while preserving recent data for realistic performance evaluation, effectively simulating real-world forecasting conditions where models must predict unseen market behavior.

A rigorous temporal alignment procedure was implemented to prevent look-ahead bias, ensuring the integrity of the forecasting model. All features used for predicting the price at time *t* + 1 were strictly constructed from information available at or before time *t*. The feature matrix incorporated lagged closing prices from periods *t* − 2, *t* − 1, and *t*. Technical indicators (RSI, MACD) were computed using data exclusively up to day *t*. This alignment guarantees that the model's predictions are based only on information that would have been available to a trader at the time of decision-making, mirroring real-world trading constraints.

Four distinct machine learning regression models were implemented and rigorously evaluated for stock price forecasting. Each model brought unique strengths and limitations to the task, ranging from highly interpretable linear methods to complex nonlinear approaches capable of handling market volatility.

Linear Regression was adopted as the foundational model. It sought to establish a direct relationship between historical Open/Low prices and future Open values by fitting a straight line using ordinary least squares optimization. The structure of this approach is shown in **Figure 1**. The method offered strong interpretability, as its coefficients could be examined to understand the influence of input variables

on predictions. However, its assumption of linear price dynamics limited its effectiveness in volatile market environments, where stock movements often exhibit nonlinear behavior.

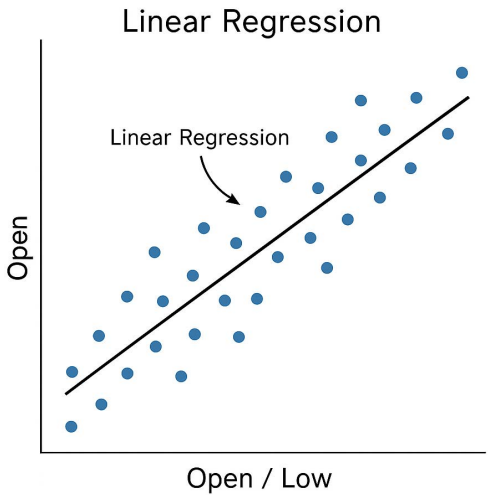


Figure 1. Structural representation of the Linear Regression model.

The Decision Tree Regressor introduced a hierarchical rule-based approach. It recursively partitioned the feature space through threshold-based conditions (e.g., "prior day's Low < $150") to isolate price patterns. The overall decision flow is illustrated in **Figure 2**. Each terminal node in the tree represented a subset of data and produced predictions by averaging the Open prices within that partition. This structure provided intuitive, human-readable decision pathways. Nonetheless, deeper trees risked overfitting, as they could capture noise in training data rather than genuine market trends.

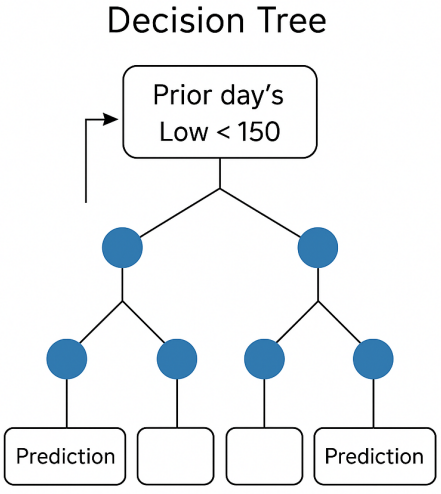


Figure 2. Decision pathway diagram of the Decision Tree Regressor.

The Random Forest Regressor expanded upon decision trees by constructing an ensemble of hundreds of diverse trees (**Figure 3**). Each tree was trained on a

randomized subset of both data and features, and the final prediction was obtained by averaging across all trees. This ensemble averaging greatly reduced the risk of overfitting, while also enabling the model to capture complex nonlinear interactions between price variables. The method demonstrated strong resilience against erratic market fluctuations and proved more robust than single decision trees.

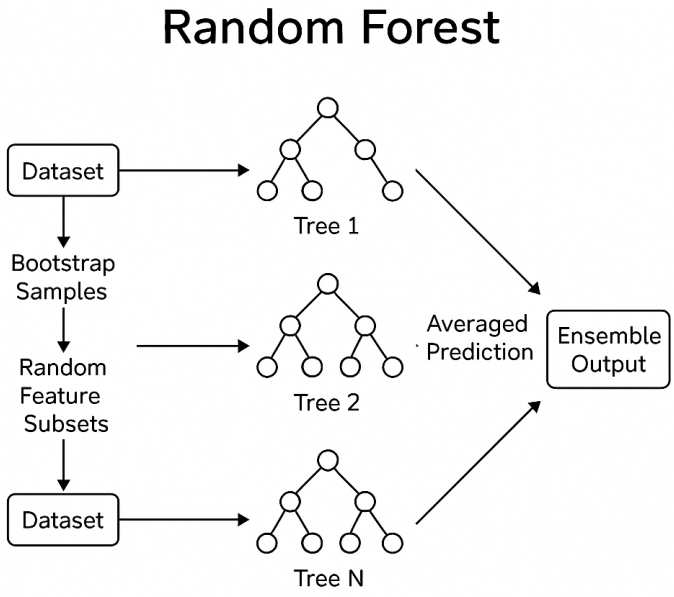


Figure 3. Ensemble schematic of the Random Forest Regressor.

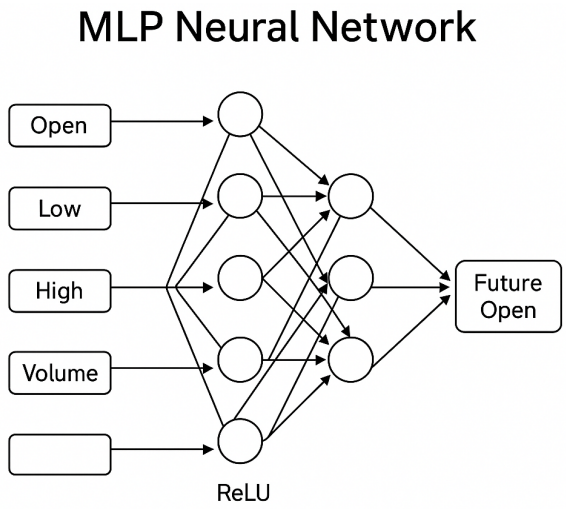


Figure 4. Neural network architecture of the Multilayer Perceptron (MLP) Regressor.

Finally, the Multilayer Perceptron (MLP) Regressor applied a neural network architecture consisting of multiple interconnected layers of processing nodes (Figure 4). By employing nonlinear activation functions such as ReLU and adjusting weights iteratively through backpropagation, the MLP was able to model intricate temporal dependencies among sequential stock prices. Its adaptive learning capa-

bility made it particularly well-suited for volatile equities. However, careful regularization was required to prevent the model from overfitting and over-optimizing on training data.

Model performance was quantified using two complementary metrics: Mean Squared Error (MSE), calculated as the average squared difference between predicted and actual Open prices to emphasize larger deviations, and a custom percent error metric measuring absolute prediction error relative to actual price magnitude. The custom percent error metric was calculated as a Mean Absolute Percentage Error (MAPE):

$$\text{MAPE} = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{A_i - F_i}{A_i}\right|$$

where A_i is the actual value and F_i is the forecast value. This metric measures absolute prediction error relative to price magnitude. All models were benchmarked against a naïve baseline that simply predicted the previous day's closing price would persist, providing a fundamental performance threshold.

Comprehensive evaluation incorporated both the average MSE across all models and a weighted average MSE where each model's contribution was scaled inversely to its test error, ensuring superior performers dominated the aggregate assessment.

The trading experiment was carried out with a rule-based automated bot that translated model forecasts into actual buy and sell actions, while still operating under realistic market restrictions. The system began with a fixed capital of $5000 and followed a simple position-sizing rule: each trade involved a single share. Trade entries and exits were guided by both predicted price movements and widely used technical signals. A long position, for example, was opened if the model suggested the next day's price would rise at least 0.1% above the current close, provided the MACD line stayed above its signal line and the RSI remained under the overbought level of 70. Positions were closed on the opposite signal—if the predicted price slipped 0.1% below the close, or if RSI and a weakening MACD indicated overbought conditions. Risk management rules were layered in as well. A stop-loss was set at 5% below the entry price, while profits were capped with a 10% take-profit target. If either threshold was reached, the trade was immediately exited. At the end of the 1200-step simulation, any open positions were liquidated to finalize results. This setup ensured that performance metrics captured not only the model's predictive accuracy but also the consistency of disciplined trading under controlled conditions.

For the final trading simulation, the Random Forest model was selected over the MLP Regressor. This decision was driven by the Random Forest's superior test performance and lower error metrics. Although the MLP can capture complex non-linearities, the Random Forest's ensemble approach demonstrated greater consistency and robustness, making it a more reliable choice for generating the trading signals that would directly impact simulated profitability.

3. Results

Table 1. Training results.

	Linear Model	DecisionTree (maxDepth = 5)	DecisionTree (maxDepth = 50)	RandomForest (maxDepth = 5, n_estimators = 100)	RandomForest (maxDepth = 50, n_estimators = 1000)	MLP (max_iter = 500, tol = 0.0001)	MLP (max_iter = 1000, tol = 0.001)	Avg MSE (Set A param)	Avg MSE (Set B param)
AAPL	1.34	1.43	0.00	1.54	1.54	1.48	1.66	8.52	5.69
MSFT	1.26	1.45	0.00	1.45	1.41	1.50	1.75	55.56	18.13
GOOG	1.46	1.54	0.00	1.73	1.68	1.49	1.75	6.93	4.43
TSLA	2.85	2.80	0.00	3.31	3.43	2.90	3.36	83.71	61.73

Table 2. Testing results.

	Linear Model	DecisionTree (maxDepth = 5)	DecisionTree (maxDepth = 50)	RandomForest (maxDepth = 5, n_estimators = 100)	RandomForest (maxDepth = 50, n_estimators = 1000)	MLP (max_iter = 500, tol = 0.00001)	MLP (max_iter = 1000, tol = 0.0001)	Avg MSE (Set A param)	Avg MSE (Set B param)
AAPL	1.30	1.69	2.04	1.10	0.59	1.38	1.80	8.55	8.52
MSFT	1.11	1.44	1.79	0.95	0.54	1.39	1.61	22.31	19.42
GOOG	1.31	1.69	0.00	1.14	0.58	1.32	1.81	4.28	5.16
TSLA	2.99	3.62	4.22	2.46	1.20	3.09	3.42	100.97	73.79

Table 3. Profits (utilizing most optimal parameters).

	AAPL	MSFT	GOOG	TSLA
1000	1670.44	2188.84	2282.35	2669.36
2000	3803.53	4707.12	4445.52	5071.94
3000	5495.20	6579.91	6610.73	7468.89
5000	8999.29	9709.69	10393.57	12275.17

4. Discussion

The inclusion of RSI and MACD indicators led to noticeable improvements in prediction accuracy across models, underscoring their usefulness in capturing both momentum and trend dynamics. All machine learning models also performed substantially better than a naïve baseline that simply predicted the previous day's price, showing that the models were indeed learning meaningful patterns rather than relying on persistence.

The training outcomes (Table 1) demonstrate how deeper decision trees, while able to nearly eliminate in-sample error, failed to carry that success into testing, a classic case of overfitting. In contrast, the testing results (Table 2) show that random forests consistently generalized well, producing lower error rates than both linear regression and the neural network configurations tested.

Perhaps the most striking result comes from Tesla (TSLA). Despite posting the highest prediction errors in testing (MSE values exceeding 100 in Table 2), it generated the strongest trading performance, more than doubling the initial $5000 allocation (Table 3). This suggests that volatility—usually seen as a forecasting challenge—can be turned into an advantage when paired with well-defined exe-

cution rules. The trading system's simple buy/sell triggers allowed it to exploit short-lived price dislocations, converting unpredictability into profit.

The capital allocation experiments (Table 3) further illustrate how profits scale with investment, though not always smoothly. While larger allocations produced higher absolute returns, the buildup of unsold positions in Google revealed a weakness in the execution logic. The one-stock-per-trade rule made the system overly conservative during predicted rallies, leaving gains unrealized. This highlights that improving execution strategies is just as important as refining predictive models.

Taken together, these results confirm the advantages of ensemble methods like random forests but also show that profitability depends on more than predictive accuracy. Execution rules and adaptability to different volatility regimes are equally critical. The findings point toward the need for more flexible trading logic, capable of adjusting to both asset-specific behaviors and changing market conditions.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Goetzmann, W.N. (2005) The Development of the Amsterdam Stock Exchange. In: Goetzmann, W.N. and Rouwenhorst, K.G., Eds., *The Origins of Value*, Yale University Press.

[2] UK Government Office for Science (2012) The Future of Computer Trading in Financial Markets. https://www.cftc.gov/sites/default/files/idc/groups/public/@aboutcftc/documents/file/tacfuturecomputertrading1012.pdf

[3] Sezer, O.B., Gudelek, M.U. and Ozbayoglu, A.M. (2020) Financial Time Series Forecasting with Deep Learning: A Systematic Literature Review: 2005-2019. *Applied Soft Computing*, **90**, Article ID: 106181. https://doi.org/10.1016/j.asoc.2020.106181

[4] Khan, M.A., Nawaz, R. and Sohail, A. (2022) Stock Market Prediction Using Machine Learning Algorithms. https://ijarsct.co.in/Paper10429.pdf

[5] Fischer, T. and Krauss, C. (2018) Deep Learning with Long Short-Term Memory Networks for Financial Market Predictions. *European Journal of Operational Research*, **270**, 654-669. https://doi.org/10.1016/j.ejor.2017.11.054

[6] Cavalcante, R.C., Brasileiro, R.C., Souza, V.L., Nobrega, J.P. and Oliveira, A.L. (2016) Computational Intelligence and Financial Markets: A Survey and Future Directions. *Applied Soft Computing*, **53**, 135-146.

[7] Strader, T.J., Rozycki, J.J., ROOT, T.H. and Huang, Y.J. (2020) Machine Learning Stock Market Prediction Studies: Review and Research Directions. *Journal of International Technology and Information Management*, **28**, 63-83. https://doi.org/10.58729/1941-6679.1435

[8] Nuti, G., Mirghaemi, M., Treleaven, P. and Yingsaeree, C. (2011) Algorithmic Trading. *Computer*, **44**, 61-69. https://doi.org/10.1109/mc.2011.31

[9] Soleymani, F. and Paquet, E. (2020) Financial Portfolio Optimization with Online Deep Reinforcement Learning and Restricted Stacked Autoencoder—Deepbreath. *Expert Systems with Applications*, **156**, Article ID: 113456. https://doi.org/10.1016/j.eswa.2020.113456

[10] Zhang, Y., Zhou, Y. and Chen, L. (2023) Time-Series Forecasting with Transformer-Based Deep Learning Models: A Survey of Methods and Applications. https://arxiv.org/abs/2303.13020