Integrated Computational System for Share Market Forecasting: Comparative Evaluation of Linear Regression, Random Forest, and Long Short-Term Memory Architectures

Vansh Vashist
Department of Artificial Intelligence and Data Science
HMR Institute of Technology and Management
Guru Gobind Singh Indraprastha University
New Delhi, India
Enrollment No. 00913311924

Mayur Narang
Department of Artificial Intelligence and Machine Learning
HMR Institute of Technology and Management
Guru Gobind Singh Indraprastha University
New Delhi, India
Enrollment No. 01313311624

Mr. Karmbir
HMR Institute of Technology and Management
Guru Gobind Singh Indraprastha University
New Delhi, India
karmbir@hmritm.ac.in

Abstract— Share market forecasting remains a difficult computational problem because price series are noisy, non-stationary, and highly sensitive to short-term shocks. This paper presents an AI-based academic prototype that converts historical open, high, low, close, and volume records into structured forecasts and decision-support signals through a disciplined time-series pipeline. The study compares three model families—Linear Regression, Random Forest, and Long Short-Term Memory networks—under a common workflow consisting of chronological data collection, cleaning, technical indicator engineering, scaling, time-aware train-validation-test splitting, and comparative evaluation. In addition to raw market fields, the prototype uses engineered features such as returns, moving averages, relative strength index, moving average convergence divergence, volatility measures, and lag variables. The discussion shows that Linear Regression is valuable as a transparent baseline, Random Forest is effective for nonlinear interactions in feature-rich tabular inputs, and LSTM is particularly suited to sequential dependency learning. The paper emphasizes practical safeguards against leakage, overfitting, and concept drift, and it interprets forecasting as a tool for academic analysis and decision support rather than a guarantee of profit. The resulting framework is reproducible, extensible, and appropriate for student-level experimentation in financial analytics.

Keywords— artificial intelligence, share market prediction, time-series forecasting, technical indicators, random forest, LSTM.

I. INTRODUCTION

Financial markets generate large volumes of sequential data whose interpretation is difficult to perform manually in real time. Daily open, high, low, close, and volume records reflect the combined effect of investor behavior, macroeconomic developments, sector-level changes, and abrupt sentiment shifts. Because these factors interact in a nonlinear and time-dependent manner, forecasting future movement is not a trivial statistical exercise. The project material supplied for this study frames the forecasting task as an academic decision-support problem in which AI transforms raw market history into structured, explainable signals for students, beginner investors, and analysts in training.

Traditional market analysis is commonly divided into fundamental analysis and technical analysis. Fundamental analysis studies intrinsic value through earnings, balance sheets, macroeconomic conditions, and firm-level disclosures, whereas technical analysis studies patterns embedded in historical price and volume movements. In practice, both approaches become difficult when the analyst must track many securities over long horizons while reacting to rapidly changing conditions. Human interpretation is also vulnerable to cognitive overload, emotional bias, and inconsistency. These practical limitations make the forecasting domain a suitable setting for machine learning and deep learning methods that can identify hidden structure in high-dimensional financial data.

The present paper studies an integrated computational system for share market forecasting using three widely recognized model families. Linear Regression is used as the simplest benchmark for understanding linear dependency. Random Forest extends the modeling space by capturing nonlinear interactions among engineered indicators. Long Short-Term Memory networks are included because they are designed to learn temporal relationships and long-range dependencies in sequential data. The aim is not to claim guaranteed profitability, but to demonstrate a reproducible workflow for model comparison, disciplined preprocessing, and interpretable presentation of outputs.

The principal contribution of the paper is therefore methodological. It consolidates data collection, preprocessing, feature construction, model training, evaluation, and presentation into a single academic prototype. The structure follows the project workflow described in the uploaded materials and selectively incorporates additional conceptual detail from the supplementary notes provided by the user, especially with respect to literature review, technical indicator formulation, and discussion of modeling challenges.

II. RELATED WORK AND LITERATURE REVIEW

Research on stock market prediction has evolved from classical econometric approaches to machine learning and deep learning systems. Early forecasting studies often relied

on autoregressive and variance-based methods such as ARIMA and GARCH because they offered mathematically grounded baselines for temporal modeling. Although useful in certain stationary settings, these methods are often less effective when market series contain nonlinear interactions, structural breaks, abrupt volatility shifts, and changing regimes. Contemporary forecasting research therefore increasingly combines statistical reasoning with richer data-driven methods.

Within machine learning, ensemble models and kernel methods have been widely used for financial classification and regression. Random Forest is notable for its use of bootstrap aggregation and randomized feature selection, which together reduce variance and improve robustness relative to a single decision tree. This makes it attractive for feature-rich tabular representations built from technical indicators, lagged returns, and rolling-window summaries. Linear models remain useful as transparent baselines because they make the contribution of explanatory variables easier to interpret and help reveal whether the dominant structure in a dataset is approximately linear.

Deep learning brought a further shift in the literature by enabling models to learn directly from sequential patterns instead of relying only on hand-crafted statistical assumptions. Recurrent Neural Networks process data step by step through time, but standard variants may struggle with vanishing gradients when long dependencies must be retained. LSTM networks address this difficulty through gating mechanisms that regulate how information is stored, forgotten, and propagated. As a result, they are frequently reported as effective for forecasting tasks in which trend persistence, delayed reactions, and longer memory effects matter.

Recent literature also highlights hybrid directions that combine numerical price histories with other information sources such as financial news, social media sentiment, macroeconomic indicators, and cross-asset relationships. These systems are promising, yet they also reveal an important caution: strong reported benchmark values often depend on the dataset, the forecast horizon, the evaluation window, and the rigor of leakage control. Consequently, model quality should be judged not only by headline accuracy but also by reproducibility, fairness of comparison, and the realism of the experimental design.

III. PROBLEM FORMULATION AND SYSTEM OBJECTIVES

The forecasting problem addressed in this study arises from three practical barriers: market volatility, information overload, and the difficulty of manual pattern recognition. Financial price series contain noise, sudden spikes, and regime shifts that can obscure usable trend information. At the same time, analysts are expected to process large amounts of heterogeneous market information in a timesensitive environment. Without automated support, manual forecasting becomes slow, error-prone, and inconsistent.

In this context, the system is formulated as a supervised time-series prediction pipeline. Given a chronological sequence of historical OHLCV records and engineered technical indicators, the model estimates either the next-day closing value or the direction of the next move. The project objectives are fivefold: first, to collect and organize historical share market data in time order; second, to transform raw data into predictive features such as returns, moving averages, RSI, MACD, volatility statistics, and lags; third, to train and compare Linear Regression, Random Forest, and LSTM under a common framework; fourth, to present predictions in an interpretable dashboardstyle format; and fifth, to document practical limitations such as overfitting, leakage, non-stationarity, and data drift.

The project scope is intentionally bounded. It focuses on historical data analysis, daily prediction, model comparison, and explanatory outputs. It does not claim to provide guaranteed returns, direct brokerage execution, or financial advice. This limitation is important because forecasting systems can support reasoning and experimentation even when they are not suitable for autonomous trading.

IV. DATASET DESIGN AND METHODOLOGY A. Data Source and Input Representation

The prototype uses historical market records characterized by open, high, low, close, and volume variables. These fields form the raw representation of daily market activity and provide the basis for engineered features. The uploaded project slides explicitly describe the input pipeline as a flow from source data to cleaning, feature generation, and model input, with chronological splitting into training, validation, and test periods. Each row is treated not merely as an isolated observation but as part of a temporal sequence from which the forecasting system learns consecutive relationships.

B. Preprocessing and Leakage Control

Preprocessing is critical because financial data can contain duplicate rows, missing values, inconsistent timestamps, and scale differences across variables. The workflow therefore sorts data chronologically, removes duplicates, and fills or flags missing values. Feature scaling is applied where necessary, particularly for models that are sensitive to feature magnitude. A strict time-aware split is preserved so that future records never enter the training features for earlier days. This chronological discipline is one of the most important safeguards in market forecasting because random shuffling can produce unrealistically optimistic results by leaking information across time.

The project material specifies a typical split of 70 percent for training, 15 percent for validation, and 15 percent for testing. Hyperparameter decisions are taken on the validation subset, while the test subset is reserved for final comparison. This design makes the evaluation more realistic and supports fair comparison across models trained on the same feature set.

C. Technical Indicator Formulation

To improve the information content of the raw OHLCV fields, the prototype derives momentum, trend, and smoothing indicators that summarize behavior over rolling windows. Four standard indicators are particularly important in this study.

$$SMA\_N(t) = (1/N)\ \Sigma\ C(t-i),\ \ i = 0 \ldots N-1 \quad (1)$$

The simple moving average smooths short-term fluctuations by averaging the recent closing prices over a window of length N. A more responsive smoothing mechanism is the exponential moving average, defined recursively as follows.

$$EMA(t) = \alpha C(t) + (1 - \alpha)EMA(t-1) \quad (2)$$

Momentum is further represented through the Relative Strength Index, which compares recent gains and losses and is widely used to indicate overbought or oversold conditions.

$$RSI = 100 - 100 / (1 + RS) \quad (3)$$

Trend crossover information is captured through the Moving Average Convergence Divergence indicator, which measures the difference between short-horizon and longhorizon exponential averages.

$$MACD(t) = EMA\_12(t) - EMA\_26(t) \quad (4)$$

In addition to these indicators, the system uses returns, volatility estimates, and lagged values. Together these engineered variables create a richer feature space than raw prices alone and improve the interpretability of the learning process.

D. Model Training Strategy

After preprocessing and feature engineering, the pipeline trains three model families. Linear Regression provides the simplest baseline and establishes whether the principal signal can be captured through a weighted linear combination of explanatory variables. Random Forest is then trained on the same feature representation to identify nonlinear interactions without assuming a specific parametric form. Finally, LSTM is trained on ordered sequences constructed from look-back windows so that the model can retain temporal memory across successive trading days.

V. MODEL ARCHITECTURES A. Linear Regression

Linear Regression is computationally efficient, easy to interpret, and useful for baseline comparison. In the forecasting setting, it estimates the contribution of each engineered feature to the output under an additive linear assumption. Its main advantage is transparency, but its primary weakness is limited capacity to model abrupt nonlinear transitions and interactions between indicators during volatile periods.

B. Random Forest

Random Forest builds an ensemble of decision trees trained on bootstrapped samples and randomized subsets of features. This architecture is well suited to tabular feature engineering because it can capture threshold effects, interaction patterns, and moderate nonlinearities without strong parametric assumptions. It is also more robust to noise and outliers than a single decision tree. However, because the model treats each training row as a structured input rather than an evolving sequence, its direct temporal memory is limited compared with recurrent networks.

C. Long Short-Term Memory Network

LSTM is a recurrent architecture specifically designed for sequential data. Through input, forget, and output gates, it controls how new information is written into memory, how old information is retained or discarded, and how internal state is exposed to later time steps. This makes LSTM attractive for share market forecasting, where recent prices may interact with patterns established over longer windows. The main trade-off is that recurrent models require more tuning, more compute, and stricter sequence preparation than baseline machine learning models.

VI. PROTOTYPE ANALYSIS AND DISCUSSION A. Evaluation Criteria

The project evaluates models using both price-error and trend-direction criteria. Mean Absolute Error measures the average magnitude of the gap between predicted and actual price without overly emphasizing a few large mistakes. Root Mean Squared Error penalizes large errors more strongly, which is useful when occasional forecast failures are costly. Directional accuracy measures how frequently the model correctly predicts whether the next movement is upward or downward. Using this combination is appropriate because a model can predict price levels imperfectly while still being informative about direction.

B. Interpretation of Prototype Behavior The uploaded project presentation states that the prototype compares Linear Regression, Random Forest, and LSTM and presents outputs such as predicted close value, trend label, confidence estimate, and a demonstration-oriented buy, hold, or sell cue. It also clearly notes that the dashboard values and comparison bars shown in the slides are illustrative rather than final experimental benchmarks. Therefore, the discussion in this paper interprets the prototype qualitatively rather than treating those visual values as validated performance claims.

Within that qualitative frame, the comparative roles of the three architectures are clear. Linear Regression is strongest as an interpretability anchor and helps determine whether the feature space contains a simple linear signal. Random Forest is attractive for engineered indicator sets because it captures nonlinear interactions and generally handles noisy tabular inputs well. LSTM is the most appropriate for genuine sequence learning because it models temporal continuity directly rather than indirectly through lag variables alone. For this reason, it is usually the most conceptually aligned model for next-day movement forecasting, provided the training data volume and tuning discipline are adequate.

The prototype also demonstrates value beyond raw prediction. By separating data ingestion, cleaning, feature computation, model training, prediction, and dashboard presentation into distinct stages, the system becomes easier to inspect, explain, and extend. This modularity is important in academic settings because it allows students to trace how modeling decisions affect downstream outputs.

C. Challenges and Limitations

Three practical issues remain central throughout the study: noise, overfitting, and concept drift. Market noise makes very short-term movement unstable and can tempt complex models to memorize random fluctuations. Overfitting becomes especially problematic when recurrent networks or large ensembles are trained without careful validation. Concept drift arises because the statistical properties of market behavior change over time; a model that performs well during one regime may degrade during another. These risks are mitigated, though not eliminated, through chronological splitting, validation-based tuning, regularization, disciplined feature computation, and periodic retraining on recent windows.

A second limitation is the absence of fully reported experimental logs in the provided project deck. Because the source slides explicitly describe their displayed values as examples, the present paper avoids inventing benchmark tables. This choice is methodologically important: a careful research paper should distinguish between demonstrated workflow capability and verified quantitative superiority. The system is therefore best interpreted as a solid prototype whose evaluation framework is ready for final datasetspecific experimentation.

VII. CONCLUSION AND FUTURE SCOPE

This paper presented an integrated computational system for share market forecasting based on historical OHLCV data, technical indicator engineering, and comparative model evaluation. The study shows that a disciplined pipeline—rather than algorithmic complexity alone—is fundamental to meaningful financial forecasting. Linear Regression serves as a transparent baseline, Random Forest provides strong nonlinear modeling for structured indicators, and LSTM offers the best conceptual fit for sequential dependency learning. Together, these models support an academic prototype that transforms raw market records into interpretable outputs for analysis and learning.

Future work can extend the prototype in several directions. First, sentiment-aware forecasting can be developed by integrating financial news and social-media signals with numerical price histories. Second, hybrid architectures can combine transformer or graph-based components with recurrent sequence models. Third, the system can be extended from single-stock analysis to portfolio-level monitoring and comparative screening. Finally, automated hyperparameter optimization, rolling retraining, and explainability modules can improve the adaptability and trustworthiness of the forecasting workflow in changing market regimes.

ACKNOWLEDGMENT

The author expresses sincere gratitude to Mr. Karmbir for guidance, academic support, and encouragement during the development of this research work. Appreciation is also extended to the Department of Artificial Intelligence and Machine Learning, HMR Institute of Technology and Management, and Guru Gobind Singh Indraprastha University for providing the institutional environment required for the completion of the project.

REFERENCES

[1] M. Saberironaghi, J. Ren, and A. Saberironaghi, “Stock market prediction using machine learning and deep learning techniques: A review,” AppliedMath, vol. 5, no. 3, Art. 76, 2025.
[2] L. N. Mintarya and S. H. F. Zulfikar, “Machine learning approaches in stock market prediction: A systematic literature review,” Procedia Computer Science, 2023.
[3] pandas development team, “Time series / date functionality,” pandas documentation.
[4] scikit-learn developers, “LinearRegression,” scikit-learn documentation.
[5] scikit-learn developers, “RandomForestRegressor,” scikitlearn documentation.
[6] TensorFlow, “tf.keras.layers.LSTM,” TensorFlow API documentation.
[7] TensorFlow, “Time series forecasting,” TensorFlow Core tutorials.
[8] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Computation, vol. 9, no. 8, pp. 1735-1780, 1997.
[9] L. Breiman, “Random forests,” Machine Learning, vol. 45, no. 1, pp. 5-32, 2001.