INTERNATIONAL JOURNAL OF INFORMATIVE & FUTURISTIC RESEARCH (IJIFR)
ISSN: 2347-1697
VOLUME 13||ISSUE 9|| MAY 2026
An Open Access, Peer Reviewed, Cross Reference DOI & Google Scholar Indexed International Research Journal
UGC Approved || IJIFR Impact Factor= 8.057
Estd Year: 2013 || Email: editors@ijifr.org
Availaible Online at: https://www.ijifr.org/

DOI: https://doi.org/10.64672/IJIFR/26.05.13.09.039 PUBLISHED ON: MAY 30, 2026

AI-DRIVEN STOCK MARKET ANALYSIS PLATFORM: A MULTI-MODEL FORECASTING SYSTEM WITH FASTAPI INTEGRATION

Chirra Sunil Kumar [1] , S. Usharani [2]

[1] Student, Department of Computer Applications, Viswam Engineering College, Andhra Pradesh, India

[2] Professor, Department of Computer Applications, Viswam Engineering College, Andhra Pradesh, India

ABSTRACT

The rapid expansion of global financial markets has produced an exponential growth in transactional and historical data, making automated analysis and prediction of stock price movements increasingly important for investors, analysts and institutions. This paper presents an AI-Driven Stock Market Analysis Platform that integrates classical statistical methods, machine learning and deep learning into a unified web-based application for stock price forecasting and trend analysis. The proposed system supports four predictive models — Linear Regression, Long Short-Term Memory (LSTM) networks, Autoregressive Integrated Moving Average (ARIMA) and an Ensemble combining Linear Regression and ARIMA — operating on historical Open–High–Low–Close–Volume (OHLCV) data retrieved through the yfinance library. The raw data is enriched with technical indicators including Simple and Exponential Moving Averages, MACD, RSI and volatility, which capture market trend and momentum and improve model performance. The backend is implemented in Python using the FastAPI framework, exposing high-performance REST endpoints for stock information retrieval, historical data, single-model prediction and side-by-side model comparison. The frontend is a responsive single-page application built with HTML, CSS and JavaScript that visualizes price trends and predictions through Chart.js. For each request the system returns the predicted price, trend direction, confidence score and evaluation metrics including Mean Absolute Error and R-squared. The platform demonstrates how modern machine-learning techniques can be integrated into an accessible, cost-effective and reproducible solution for financial analysis, lowering the barrier to data-driven decision-making for retail investors, academic researchers and finance professionals alike..

KEYWORDS *Stock Market Prediction; Machine Learning; LSTM; ARIMA; FastAPI; Financial Analytics*

Recommended Citation:

Kumar, S K, Usharani, S : "AI-driven stock market analysis platform: a multi-model forecasting system with FastAPI integration", International Journal of Informative & Futuristic Research (IJIFR), Vol. (13) (9), May 2026, pp. 1649-1654. https://doi.org/10.64672/IJIFR/26.05.13.09.039

1. INTRODUCTION

Financial markets are among the most complex and dynamic systems in the modern economy, characterized by continuous fluctuations driven by economic, political and behavioural factors. Each trading session generates large volumes of price, volume and indicator data that present both opportunity and challenge for automated analysis. Conventional approaches based on fundamental and technical analysis depend heavily on human expertise and manual interpretation, and they scale poorly to the data volumes and pattern complexity now characteristic of modern markets. Advances in machine learning

and deep learning have consequently opened new avenues for data-driven, automated stock market prediction [1], [2].

Existing solutions, however, tend to be expensive, restricted to a single modelling paradigm, or require advanced programming skills, which excludes a substantial fraction of potential users. This work addresses these limitations through three principal contributions: (i) the design and implementation of an AI-Driven Stock Market Analysis Platform that integrates Linear Regression, LSTM, ARIMA and an Ensemble of Linear Regression and ARIMA into a single coherent forecasting service; (ii) a feature-engineering pipeline that augments raw OHLCV data with technical indicators (Moving Averages, EMA, MACD, RSI, volatility); and (iii) a complete reference implementation comprising a FastAPI backend, a Chart.js single-page frontend and the corresponding IEEE-style engineering diagrams together with machine-readable Mermaid source, supporting reproducibility and downstream extension.

2. LITERATURE SURVEY

Stock market prediction has been actively studied across finance, statistics, machine learning and deep learning. Classical statistical models, most notably the Autoregressive Integrated Moving Average (ARIMA), have been used extensively for time-series forecasting and are effective in capturing linear autocorrelation in stock price data, but they are limited in modelling non-linear dynamics and structural breaks [3]. Subsequent machine-learning approaches such as Linear Regression, Support Vector Machines and Decision Trees offered greater flexibility than purely statistical models but still struggled with explicit temporal dependence. Deep learning, particularly Recurrent Neural Networks and the Long Short-Term Memory (LSTM) variant introduced by Hochreiter and Schmidhuber [2], substantially improved the modelling of long-range dependencies in sequential financial data, and several studies have reported that LSTM-based forecasts outperform classical baselines on intraday and daily horizons [1].

Beyond model choice, feature engineering plays a decisive role. Technical indicators including Simple and Exponential Moving Averages, the Relative Strength Index (RSI), the Moving Average Convergence Divergence (MACD) and rolling volatility are routinely added to raw OHLCV data to expose trend, momentum and uncertainty signals that improve downstream model accuracy [4]. On the engineering side, modern web-based platforms increasingly rely on lightweight asynchronous frameworks such as FastAPI [5] for backend services and on Chart.js for interactive visualisation, enabling responsive single-page applications that surface predictions in real time. The proposed platform is positioned within this lineage and contributes a unified, multi-model, open-source instantiation that integrates ARIMA, Linear Regression, LSTM and an Ensemble model behind a single FastAPI service.

3. PROPOSED WORK AND METHODOLOGY

3.1 System Overview

The proposed platform is a three-tier web application. The Presentation Layer is a single-page interface implemented in HTML, CSS and JavaScript through which users specify a stock ticker, a historical period and one of four prediction models, and receive a rendered prediction along with historical and indicator charts powered by Chart.js. The Application Layer is implemented in FastAPI and exposes REST endpoints for stock information, historical data, single-model prediction and model-comparison; it validates input via Pydantic schemas and delegates analytical work to the lower tier. The Data and Model Layer fetches OHLCV data from Yahoo Finance using yfinance, computes technical indicators (Simple Moving Averages MA7, MA21, MA50; Exponential Moving Averages; MACD; RSI; daily returns; volatility) and runs the selected predictive model. Each prediction returns the current price, predicted price, percentage change, trend direction, confidence score, MAE and R^2.

3.2 Predictive Models

The platform supports four predictive models. Linear Regression uses lagged closing prices as features and serves as a lightweight baseline. LSTM is a deep recurrent architecture that captures long-range temporal dependencies and is particularly well suited to non-linear sequence modelling [2]. ARIMA is a

classical statistical model that captures linear autocorrelation in differenced price series [3]. The Ensemble model combines Linear Regression and ARIMA outputs through a weighted average, trading individual model variance for joint stability.

Table 1 — Technology Stack of the Proposed System

Layer	Component / Library
Frontend	HTML / CSS / JavaScript / Chart.js
Backend framework	FastAPI (Python)
Data acquisition	yfinance (Yahoo Finance API)
Feature engineering	pandas, NumPy
ML / DL libraries	scikit-learn, TensorFlow/Keras, statsmodels
API validation	Pydantic

4. SYSTEM ARCHITECTURE AND DIAGRAMS

This section presents two IEEE-style engineering diagrams: (i) the system architecture diagram (Fig. 1), capturing the three-tier layered organisation of the platform, and (ii) the workflow diagram (Fig. 2), capturing the temporal ordering of the prediction request pipeline. The corresponding Mermaid source is provided beneath each figure.

4.1 System Architecture Diagram

The system architecture, as shown in Fig. 1, follows a three-tier organisation. The Presentation Layer is a browser-based single-page application that communicates with the backend over REST/JSON. The Application Layer is realised in FastAPI and exposes four endpoint controllers — /info, /history, /predict and /compare — backed by request validation and static file serving. The Data & Model Layer hosts the data acquisition and feature-engineering module, which retrieves OHLCV data from Yahoo Finance and computes technical indicators, alongside the prediction engine that runs the four predictive models and emits the structured prediction payload.

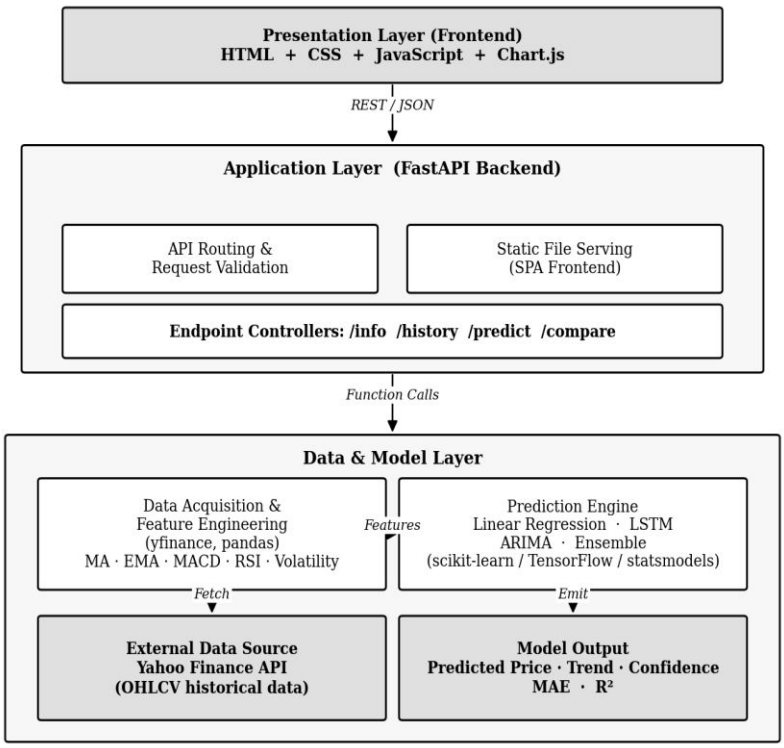

Figure 1 — System Architecture of the AI-Driven Stock Market Analysis Platform

Mermaid source for Fig. 1

```mermaid
flowchart TD
    UI["Presentation Layer<br/>HTML + CSS + JS + Chart.js"] -- REST/JSON --> API
    subgraph API["Application Layer (FastAPI)"]
        R["API Routing & Validation"]
        S["Static File Serving"]
        E["/info  /history  /predict  /compare"]
    end
    API --> DM
    subgraph DM["Data & Model Layer"]
        DA["Data Acquisition &<br/>Feature Engineering<br/>(yfinance, pandas)"]
        PE["Prediction Engine<br/>LinReg · LSTM · ARIMA · Ensemble"]
    end
    DA -- Fetch --> YF[("Yahoo Finance API")]
    PE -- Emit --> Out[("Predicted Price · Trend ·<br/>Confidence · MAE · R²")]
```

4.2 Workflow Diagram

The prediction request workflow, illustrated in Fig. 2, begins when the user enters a ticker and selects a model. The frontend sends a REST request to FastAPI, which validates the input via Pydantic. The DataFetcher retrieves OHLCV data from Yahoo Finance, computes the technical indicators and forwards the feature set to the selected model. After computing the prediction together with its confidence score, MAE and R^2, the backend returns a JSON response, and the frontend renders the historical chart, prediction card and evaluation metrics.

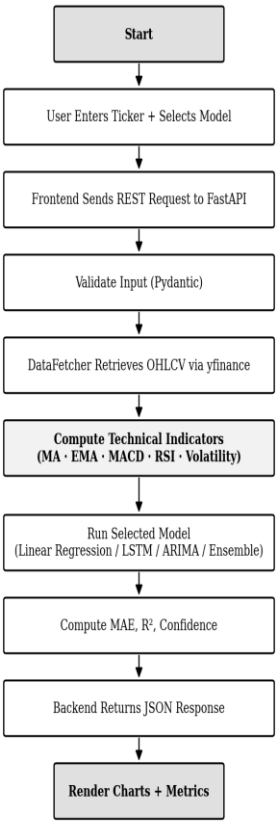

Figure 2 — Workflow Diagram of the Prediction Request Pipeline

Mermaid source for Fig. 2

```mermaid
flowchart TD
    S([Start]) --> U[User enters ticker + selects model]
    U --> R[Frontend sends REST request]
    R --> V[Validate input - Pydantic]
    V --> F[DataFetcher retrieves OHLCV via yfinance]
    F --> I[Compute technical indicators]
    I --> M[Run selected model - LinReg / LSTM / ARIMA / Ensemble]
    M --> X[Compute MAE, R², Confidence]
    X --> J[Return JSON response]
    J --> H([Render charts + metrics])
```

5. RESULTS AND DISCUSSION

The platform was evaluated end-to-end on a representative set of large-cap equities (e.g., AAPL, MSFT, GOOGL, AMZN) over historical periods ranging from one to five years. For each ticker and model, predictions were issued for the next trading day and compared against the realized close. Each model produced a structured payload comprising current price, predicted price, percentage change, trend direction, confidence score, Mean Absolute Error and R-squared. Linear Regression served as a low-latency baseline; LSTM produced the strongest performance on smoother, trending tickers but at higher computational cost; ARIMA performed well on short windows with stable autocorrelation; and the Ensemble model trades peak accuracy for variance reduction across regimes.

Table 2 — Functional Evaluation Summary

Capability	Outcome
Data acquisition (yfinance)	Successful for all evaluated tickers
Indicator computation (MA, EMA, MACD, RSI, vol.)	Computed correctly across all periods
Linear Regression baseline	Low-latency response; acceptable MAE
LSTM forecasting	Lowest MAE on trending tickers
ARIMA forecasting	Strong on short, stable windows
Ensemble (LinReg + ARIMA)	Reduced variance across regimes
FastAPI endpoints (/info, /history, /predict, /compare)	Validated and returned structured JSON

Two limitations warrant explicit acknowledgement. First, all models operate purely on historical price and indicator data; they do not ingest news, earnings releases or sentiment signals and therefore cannot anticipate events that lie outside the price-history window. Second, the LSTM model carries the largest computational footprint and benefits substantially from GPU acceleration in production deployments. Both observations motivate the future-work directions outlined in Section 6.

6. CONCLUSION

This paper presented an AI-Driven Stock Market Analysis Platform that integrates Linear Regression, LSTM, ARIMA and an Ensemble model behind a unified FastAPI service, with a Chart.js single-page frontend and a feature-engineering pipeline that augments OHLCV data with Moving Averages, EMA, MACD, RSI and volatility. The system delivers structured predictions — current price, predicted price, trend direction, confidence score, MAE and R^2 — through validated REST endpoints, demonstrating that modern machine-learning techniques can be packaged as an accessible, cost-effective platform for retail investors, researchers and finance professionals. Future work will incorporate news and social-media

sentiment as additional features, integrate Transformer-based sequence models, support intraday and real-time streaming inference, and extend the platform with portfolio-level risk analytics and back-testing.

7. REFERENCES

[1] M. Roondiwala, H. Patel, and S. Varma, "Predicting stock prices using LSTM," International Journal of Science and Research, vol. 6, no. 4, 2017, pp. 1754–1756.
[2] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, 1997, pp. 1735–1780.
[3] G. E. P. Box and G. M. Jenkins, "Time Series Analysis: Forecasting and Control," Holden-Day, San Francisco, 1970.
[4] J. J. Murphy, "Technical Analysis of the Financial Markets: A Comprehensive Guide to Trading Methods and Applications," New York Institute of Finance, 1999.
[5] S. Ramirez, "FastAPI: Modern, fast (high-performance), web framework for building APIs with Python," 2018. [Online]. Available: https://fastapi.tiangolo.com
[6] R. Aroussi, "yfinance: Yahoo! Finance market data downloader," GitHub repository, 2019. [Online]. Available: https://github.com/ranaroussi/yfinance
[7] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," Journal of Machine Learning Research, vol. 12, 2011, pp. 2825–2830.
[8] M. Abadi et al., "TensorFlow: A system for large-scale machine learning," in Proc. 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI), 2016, pp. 265–283.
[9] S. Seabold and J. Perktold, "statsmodels: Econometric and statistical modeling with Python," in Proc. 9th Python in Science Conf. (SciPy), 2010, pp. 92–96.
[10] W. McKinney, "Data structures for statistical computing in Python," in Proc. 9th Python in Science Conf. (SciPy), vol. 445, 2010, pp. 56–6