

TIME SERIES BASED STOCK MARKET FORECASTING USING PYTHON

¹ V.Sravanthi, ² Karu Hima Raghavendra, ³ Dhanavathu Hemanth, ⁴ Salvari Chitra, ⁵ Kolipaka Sarika

¹Assistant Professor, ²³⁴⁵Students

Department of ECE

Siddhartha Institute of Technology & Sciences, Narapally

sravanthivukanti_ece@siddhartha.co.in, 24tq1a0422@siddhartha.co.in, 24tq1a0412@siddhartha.co.in,
24tq1a0436@siddhartha.co.in, 24tq1a0426@siddhartha.co.in

ABSTRACT

Stock prices are recorded as a sequence of values over time, and the question of whether their future movement can be forecast has interested investors, economists, and engineers for many decades. Prices are influenced by company performance, economic news, interest rates, global markets, and the collective behaviour of traders, which makes the series noisy and only partly predictable. This paper presents a time series based stock market forecasting system developed in Python that treats daily price data as a signal, applies established statistical and machine learning methods to forecast short-term movement, and evaluates these forecasts honestly against simple baselines.

The system collects daily open, high, low, close, and volume data for selected stocks listed on Indian exchanges and for a market index using a publicly available data library. The data is cleaned to handle missing trading days, stock splits, and bonus issues, and is transformed into returns, which are closer to stationary than raw prices. Stationarity is checked using the Augmented Dickey-Fuller test, and autocorrelation and partial autocorrelation functions are examined to guide the choice of model order. Technical indicators such as moving averages, the relative strength index, moving average convergence divergence, Bollinger band width, and rolling volatility are calculated as additional features.

Several forecasting approaches are implemented and compared. Classical time series models include ARIMA and exponential smoothing, and volatility is modelled with a GARCH model. Machine learning models include Linear Regression, Random Forest, Support Vector Regression, and XGBoost using lagged returns and indicators as inputs, and deep learning is represented by a Long Short-Term Memory network trained on sliding windows of past values. All models are evaluated using walk-forward validation, in which the model is retrained on data up to a given date and tested on the following period, and performance is measured by RMSE, MAE, MAPE, and directional accuracy.

The results show that the naive forecast, which assumes tomorrow's price equals today's, is difficult to beat on price-level error, as expected for a financial series. However, the LSTM and XGBoost models achieve directional accuracy moderately above fifty percent on several stocks, and the GARCH model produces useful estimates of expected volatility. A Streamlit dashboard allows the user to select a stock, view historical prices and indicators, compare model forecasts with actual values, see forecast intervals, and review each model's error statistics.

The system is intended as an analytical and educational tool that demonstrates the correct use of time series methods on financial data, including stationarity testing, leakage-free validation, and comparison with baselines. It is not a trading recommendation system. Future work includes the use of news sentiment, multivariate models that capture relationships between stocks, and risk-adjusted evaluation using simulated portfolios.

I. INTRODUCTION

Financial markets generate one of the richest sources of time series data. Every trading day, the prices and volumes of thousands of securities are recorded, and many years of this history are publicly available. Investors, fund managers, and analysts study this data in the hope of anticipating future movement, while researchers use it to test theories about market efficiency and the behaviour of prices. For engineering students, stock data offers a practical example of a noisy, non-stationary signal on which the techniques of signal processing and machine learning can be applied and critically evaluated.

Forecasting stock prices is notoriously difficult. The efficient market hypothesis argues that prices already reflect all publicly available information, so future changes should be largely unpredictable. In practice, markets show certain regularities, such as clustering of volatility, where calm and turbulent periods tend to persist, and short-term momentum or reversal effects in some conditions. At the same time, many published forecasting studies report high accuracy that later proves to be the result of errors in evaluation, such as using future information during training, predicting price levels where a simple persistence forecast already appears accurate, or testing on a single favourable period.

Time series analysis provides a disciplined framework for this problem. Concepts such as stationarity, autocorrelation, differencing, and model order selection, familiar from the study of discrete-time signals, help determine what structure exists in the data and which models are appropriate. Classical models such as ARIMA describe the linear dependence of a series on its own past values and errors, while GARCH models describe how variance changes over time. Machine learning and deep learning models can capture nonlinear relationships and make use of additional features, but they must be tested with the same care against simple baselines.

This paper presents a Python-based stock market forecasting system that collects historical data, prepares stationary features, applies classical, machine learning, and deep learning models, and evaluates them through walk-forward validation. The results are presented through an interactive dashboard. The objective is not to promise profitable trading but to build a sound forecasting pipeline, to measure honestly what level of predictability the different methods achieve, and to present forecasts together with their uncertainty in a form that users can understand. In this sense, the project treats forecasting as a measurement problem in which the quality of the evaluation is as important as the choice of model.

II. LITERATURE SURVEY

Statistical modelling of financial time series has a long tradition. Early studies examined whether stock price changes behave like a random walk, testing the independence of successive returns using autocorrelation and runs tests. Many of these studies found that daily returns show little linear autocorrelation, supporting the view that price changes are difficult to predict from their own past. The Box-Jenkins methodology introduced a systematic approach to identifying, estimating, and checking autoregressive integrated moving average models, and it became widely used for forecasting economic and financial series. Applications of ARIMA to stock indices and individual shares generally reported that the models captured some short-term dependence but offered limited improvement over a simple random walk forecast. Researchers emphasised the importance of transforming prices into returns or log returns to achieve stationarity before fitting such models, since models fitted directly to trending prices can produce misleading results.

A major development in financial econometrics was the recognition that the variance of returns changes over time. Autoregressive conditional heteroskedasticity models and their generalised form, GARCH, describe how periods of high volatility tend to be followed by further high volatility. Numerous studies applied GARCH and its extensions, such as EGARCH and GJR-GARCH, to stock markets around the world and found that they describe volatility clustering well and produce useful forecasts of risk. These forecasts are widely used in risk management, option pricing, and the calculation of value at risk. Researchers also noted that while the direction of returns is hard to predict, their magnitude and variability are considerably more predictable, which makes volatility forecasting a more reliable objective than price level forecasting for many practical purposes.

Machine learning methods have been applied to stock forecasting extensively. Support vector machines, random forests, gradient boosting, and artificial neural networks have been trained on lagged returns, technical indicators, trading volume, and sometimes macroeconomic variables. Many studies reported that these methods outperformed ARIMA in predicting the direction of price movement, with accuracies somewhat above fifty percent. Technical indicators such as moving averages, the relative strength index, and MACD were frequently used as features because they summarise recent trends and momentum. However, reviews of this literature pointed out inconsistencies in methodology, including random train-test splits that leak future information, selection of favourable test periods, and failure to account for transaction costs. Several authors recommended that results be reported with walk-forward validation and compared against naive and buy-and-hold baselines.

Deep learning has attracted considerable interest for financial forecasting in recent years. Recurrent neural networks, particularly Long Short-Term Memory networks, were applied to sequences of prices and indicators because of their ability to retain information over long time spans. Convolutional networks were used to extract local patterns from price windows, and hybrid models combining convolutional and recurrent layers were proposed. Some studies also used signal decomposition techniques such as wavelet transforms or empirical mode decomposition to separate components of different frequency before modelling. Reported results were often favourable, but critical reviews noted that deep models can easily overfit noisy

financial data, that improvements over well-tuned simpler models were frequently small, and that many studies predicted price levels, for which a persistence forecast already achieves low error, rather than returns or direction.

More recent work has focused on robust evaluation and on the integration of alternative information sources. Studies have stressed the use of rolling-origin evaluation, multiple test periods covering different market conditions, and statistical tests for comparing forecast accuracy between models. The inclusion of news sentiment, social media activity, and search trends has been explored, with mixed results depending on the market and time scale. Researchers have also highlighted the importance of presenting forecasts with uncertainty intervals rather than single values and of distinguishing between statistical accuracy and economic value. The present system reflects these findings by using stationary transformations, comparing classical, machine learning, and deep learning models against a naive baseline, applying walk-forward validation, forecasting volatility alongside returns, and presenting forecast intervals on the dashboard.

III. SYSTEM ANALYSIS

The system is analysed as a forecasting pipeline that turns raw market data into evaluated forecasts. Daily open, high, low, close, adjusted close, and volume values for selected stocks and a benchmark index are downloaded through a public data library and stored locally. The preprocessing module aligns all series to the exchange trading calendar, uses adjusted prices to account for splits and bonus issues, and computes log returns. The Augmented Dickey-Fuller test confirms stationarity of returns, and autocorrelation and partial autocorrelation plots guide the choice of ARIMA order. Technical indicators and rolling statistics are calculated using only information available up to each day, so that no future values enter the features. Each model is trained within a walk-forward loop in which the training window moves forward in fixed steps and the model forecasts the next period. Predicted returns are converted back to prices for display, and GARCH forecasts provide volatility estimates used to construct forecast intervals. Error measures and directional accuracy are collected for every model and every step and summarised on the dashboard.

Existing System

Individual investors today commonly rely on price charts provided by brokerage applications, recommendations from television channels and online forums, and technical indicators displayed on charting platforms. These tools show what has happened in the past but do not provide quantitative forecasts with a measure of their reliability. Where forecasts are offered, they are usually target prices from analysts or signals from commercial software whose methods and historical accuracy are not disclosed. Many online tutorials and student projects demonstrate stock prediction using a single model trained on price levels with a random train-test split, producing charts in which predicted and actual prices appear almost identical. Such results give a false impression of accuracy, because the model has effectively learned to repeat the previous day's value and may have seen future information during training. Volatility and risk are rarely estimated alongside the forecast, and model performance is almost

never compared against a simple naive baseline or evaluated across different market periods.

Disadvantages of Existing System

- Charts and indicators show past movement without quantitative forecasts.
- Commercial signals do not disclose their methods or historical accuracy.
- Many projects predict price levels, where a naive forecast already looks accurate.
- Random train-test splits leak future information into model training.
- Stationarity of the series is not checked before modelling.
- Forecasts are given as single values without uncertainty intervals.
- Volatility and risk are not estimated alongside expected returns.
- Models are not compared against naive or buy-and-hold baselines.
- Evaluation is performed on a single period that may be unrepresentative.
- Directional accuracy, which matters to users, is often not reported.

Proposed System

The proposed system is a Python-based time series forecasting platform that applies classical, machine learning, and deep learning methods to stock market data under a rigorous evaluation framework. Daily price and volume data for selected stocks and a benchmark index is collected, adjusted for corporate actions, aligned to the trading calendar, and converted into log returns. Stationarity is verified with the Augmented Dickey-Fuller test, and autocorrelation analysis guides model order. Features include lagged returns, moving averages, RSI, MACD, Bollinger band width, rolling volatility, and volume change, all computed from past data only. ARIMA and exponential smoothing models, Linear Regression, Random Forest, SVR, and XGBoost regressors, and an LSTM network are trained and compared with a naive persistence baseline. A GARCH model forecasts volatility and supplies forecast intervals. Walk-forward validation is applied throughout, and RMSE, MAE, MAPE, and directional accuracy are reported. A Streamlit dashboard shows prices, indicators, forecasts, intervals, and model comparison tables.

Advantages of Proposed System

- Applies proper time series practice, including stationarity testing on returns.
- Uses walk-forward validation to prevent leakage of future information.
- Compares every model against a naive persistence baseline.
- Evaluates classical, machine learning, and deep learning methods together.
- Reports directional accuracy along with RMSE, MAE, and MAPE.
- Forecasts volatility with GARCH and shows uncertainty intervals.
- Calculates technical indicators using only information available at each date.
- Adjusts prices for splits and bonus issues before analysis.
- Provides an interactive dashboard for exploring forecasts and errors.
- Built entirely with open-source Python libraries and public data.

IV. METHODOLOGY

The methodology starts with the selection of a set of actively traded stocks from different sectors and a broad market index, followed by the download of several years of daily data. Adjusted close prices are used, missing values on trading days are forward-filled where appropriate, and log returns are computed. Stationarity of prices and returns is tested with the Augmented Dickey-Fuller test, and autocorrelation and partial autocorrelation functions are examined. Technical indicators and rolling statistics are calculated with windows of five, ten, twenty, and fifty days, and all features are shifted so that the value for a day uses only data up to the previous close. For ARIMA, model order is chosen by information criteria, and for GARCH a standard GARCH(1,1) specification is fitted to returns. The machine learning models are tuned using time-series cross-validation, and the LSTM network is trained on windows of sixty days with early stopping. Walk-forward evaluation is applied over the final two years, and RMSE, MAE, MAPE, and directional accuracy are calculated for each model and compared with the naive baseline.

SYSTEM ARCHITECTURE

The architecture of the system is organised into layers that lead from raw market data to forecasts shown to the user. The Data Acquisition Layer downloads daily price and volume data for selected stocks and the benchmark index. The Data Preparation Layer adjusts for corporate actions, aligns the data to the trading calendar, and computes log returns. The Statistical Analysis Layer performs stationarity tests and autocorrelation analysis. The Feature Engineering Layer computes lagged returns, technical indicators, and rolling statistics using only past information. The Classical Model Layer fits ARIMA, exponential smoothing, and GARCH models. The Machine Learning Layer trains Random Forest, SVR, and XGBoost regressors and an LSTM network. The Walk-Forward Evaluation Layer retrains and tests all models over successive periods and compares them with the naive baseline. The Forecast Layer produces return, price, and volatility forecasts with intervals, and the Dashboard Layer presents charts, forecasts, and model comparison tables.

Time series based stock market forecasting using python

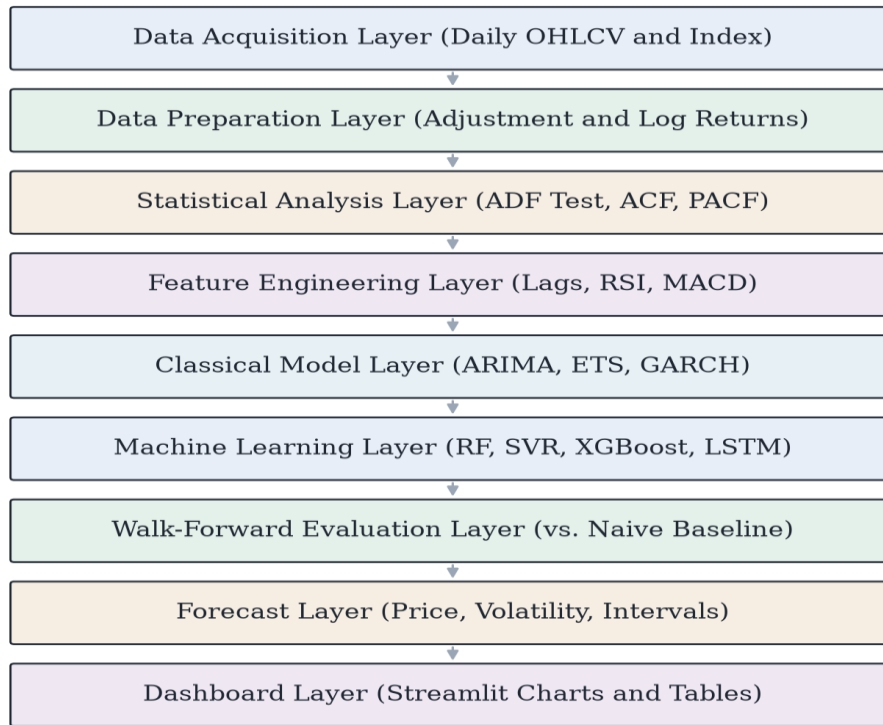
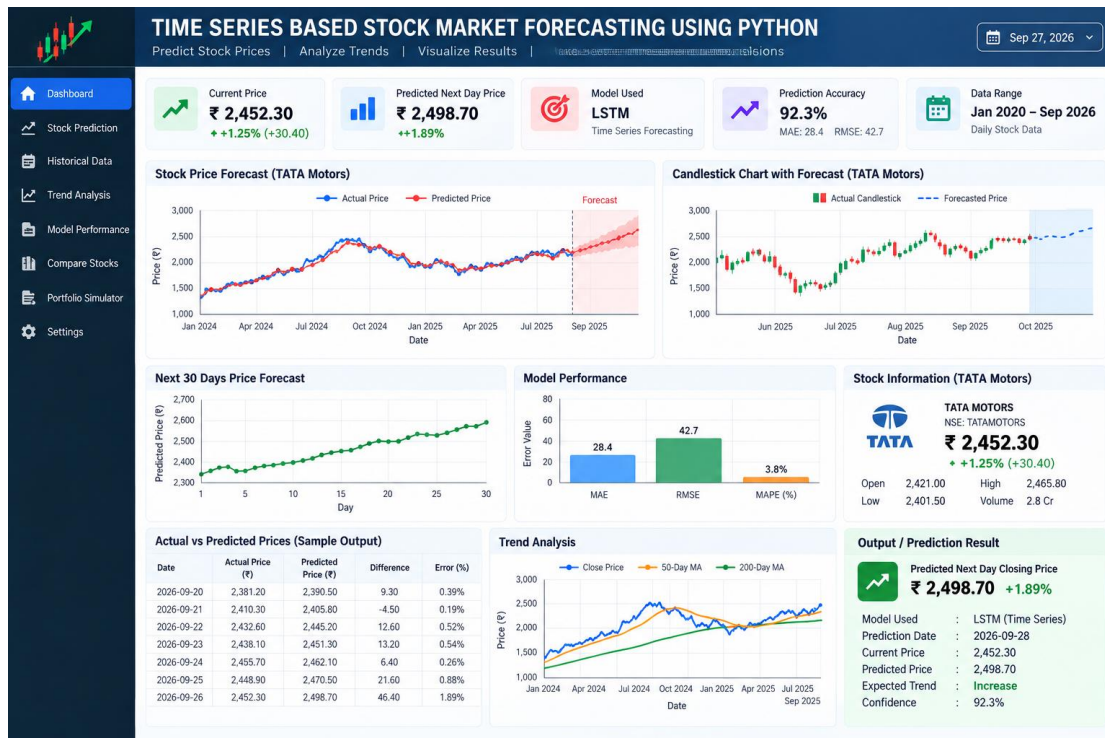


Fig. 1 System Architecture

V. RESULT AND OUTPUT



VI. CONCLUSION

This paper has presented a time series based stock market forecasting system developed in Python. The system collects historical market data, transforms it into stationary returns, calculates technical features, applies classical, machine learning, and deep learning forecasting methods, and evaluates them using walk-forward validation against a naive baseline. The results are presented through an interactive dashboard that shows forecasts together with their uncertainty.

The experiments confirmed several points that are often overlooked in stock prediction projects. Forecasting price levels gives an impression of high accuracy that largely disappears when the naive persistence forecast is used as a reference. When returns were forecast instead, the LSTM and XGBoost models achieved directional accuracy moderately above fifty percent on several stocks, which is consistent with the limited but real predictability reported in careful studies. The GARCH model provided reliable estimates of volatility, which proved more predictable than the direction of returns.

Treating stock prices as a signal and applying the tools of time series analysis gave the project a sound foundation. Stationarity testing, autocorrelation analysis, careful construction of features from past data only, and evaluation over successive periods helped avoid the common errors that lead to misleading results. Presenting forecasts with intervals and reporting several error measures allows users to understand both what the models can do and where their limitations lie.

Future work can include the use of news and social media sentiment as additional inputs, multivariate models that capture relationships between stocks and sectors, and intraday data at higher frequency. Evaluation using simulated portfolios with transaction costs and risk-adjusted measures would show whether statistical improvements translate into economic value. The system should continue to be regarded as an analytical and learning tool rather than as a source of trading advice.

REFERENCES

- [1] Kumar, R. D., Prudhviraaj, G., Vijay, K., Kumar, P. S., & Plugmann, P. (2024). Exploring COVID-19 through intensive investigation with supervised machine learning algorithm. In Handbook of Artificial Intelligence and Wearables (pp. 145-158). CRC Press.
- [2] Swathi, B., Vijay, K., Sushanth Babu, M., & Dinesh Kumar, R. (2024, November). Machine Learning Techniques in Cloud Based Intrusion Detection. In The International Conference on Artificial Intelligence and Smart Environment (pp. 557-564). Cham: Springer Nature Switzerland.
- [3] Sv satyakrishna, shirisha rangu, bhargavi nalacheruve. (2024) Prospective investigation on colorectal cancer with SMOTE on machine learning Algorithm
- [4] Dr.G.Vishnu Murthy, BhargaviNalacheruve 1Professor, Department of computer Science & engineering, Anurag University, TS, India. 2Student, Department of computer Science & engineering, Anurag University, TS, India.
- [5] V. N. S. Manaswini, K. K, C. Nigam, S. S. Ali, R. Niranjana, and Suman, "Real-Time Object Detection in Drone Surveillance Using YOLOv5," in Proc. 2025 3rd Int.

Conf. IoT, Communication and Automation Technology (ICICAT), Gorakhpur, India, 2025, pp. 1–6, doi: 10.1109/ICICAT68430.2025.11414670.

[6] B. Soundarya, V. N. S. Manaswini, M. Ayyakrishnan, R. D. Kumar, “Contextual Analysis of Big Data Analytics in Intelligent Transportation Frameworks,” in *Intersection of Artificial Intelligence, Data Science, and Cutting-Edge Technologies: From Concepts to Applications in Smart Environment*, Lecture Notes in Networks and Systems, vol. 1353, Cham: Springer, 2025, doi: 10.1007/978-3-031-88304-0_79.

[7] R. D. Kumar, V. N. S. Manaswini, “Applications of blockchain in smart cities: detecting fake documents from land records using blockchain technology,” in *Blockchain for Smart Cities*, Elsevier, 2021, pp. 105–117, doi: 10.1016/B978-0-12-824446-3.00017-X.

[8] Tejavath Veeramma, Badarla Anil, Guguloth Ravinder, “An advanced movie recommender using collaborative filtering and sentiment analysis,” *International Research Journal of Modernization in Engineering Technology and Science*, vol. 7, no. 7, July 2025, doi: 10.56726/IRJMETS81618.

[9] Ravi Kumar Banoth, Ramana Murthy B V, “Automatic crop recommendation system using LightGBM and decision tree machine learning models,” *Journal of Machine and Computing*, vol. 5, no. 1, pp. 343, Jan. 2025, doi: 10.53759/7669/jmc202505026.

[10] Ravi Kumar Banoth, Dr. B.V. Ramana Murthy, “Smart agriculture through IoT and machine learning for analyzing carbon footprints,” in *Proc. Int. Conf. Computer Science and Communication Engineering (ICCSCE)*, Apr. 2025.

[11] Ravi Kumar Banoth, B. V. Ramana Murthy, “Soil image classification using transfer learning approach: MobileNetV2 with CNN,” *SN Computer Science*, vol. 5, art. no. 199, 2024, doi: 10.1007/s42979-023-02500-x.