iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** IX **Month of publication:** September 2025

DOI: https://doi.org/10.22214/ijraset.2025.73969

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue IX Sep 2025- Available at www.ijraset.com

Trade One: A Prediction System - Stock Market Price Prediction using Deep Learning

Mrs. Subhashree D C[1], Dr. Girish Kumar D[2], Mr. Rakesh Y M[3]
[1]*Assistant Professor, Department of MCA, Ballari Institute of Technology & Management, Ballari, Karnataka, India*
[2]*Professor & HOD, Department of MCA, Ballari Institute of Technology & Management, Ballari, Karnataka, India*
[3]*Student, Department of MCA, Ballari Institute of Technology & Management, Ballari, Karnataka, India*

Abstract: ***Because of the market's intrinsic volatility, non-stationarity, and non-linear dynamics, accurately and promptly predicting stock prices is a major challenge in financial research. This study offers a comprehensive web-based solution for real-time stock trend forecasting using deep learning. The design combines a dynamic frontend created with HTML, Bootstrap, and JavaScript, a secure MySQL database for user authentication, and a Python Flask backend with a pre-trained Keras-based Long Short-Term Memory (LSTM) time-series model. The system uses a strong data pipeline that comprises sequential data structure using a sliding window technique, normalization using MinMaxScaler to scale features between 0 and 1, and real-time data collecting from Yahoo Finance to guarantee high-quality input for the model. The main features of the tool are presented to users via an interactive dashboard that offers descriptive statistics, technical indicators such as Exponential Moving Averages (EMA), and anticipated price trajectories, indicating the improved perfect angle of the deep learning approach when compared to more conventional forecasting models like ARIMA and Support Vector Regression. The system shows great promise as a decision-support tool for financial market participants, offering clear data visualizations and a historical log of prediction performance to increase interpretability and foster user trust.***

I. INTRODUCTION

Because financial markets, especially the stock market, are intricate, volatile, and multifaceted, they pose significant prediction issues. Since a precise stock price prediction can result in large financial gains, it is a topic of great study and practical interest. Although the capable Market Hypothesis has that stock prices are always unpredictable because they directly affect all obtainable data, empirical data shows that meaningful forecasting is possible due to behavioural patterns, market inefficiencies, and the halving of learnable temporal dependencies. Conventional techniques which is frequently based on statistical analysis, works good for linear systems but have trouble identifying the complex, non-linear patterns present in financial time-series data.

Cutting-edge new methods for contacting this issue have been made possible by recent developments in artificial intelligence, especially in the area of deep learning. Long Short-Term Memory networks, a more sophisticated version of recurrent neural networks (RNNs), are made to identify patterns and not permanent dependencies in sequential information. They can selectively recall or discard information over extended periods of time thanks to their special architecture, which includes memory cells and unique gating mechanisms (input, forget, and output gates). Given that distant past events might still have an impact on present market movements, this feature is ideal for financial forecasting.

An end-to-end solution that operationalizes a deep learning model within an intuitive, interactive web application is designed and implemented in detail in this study. A secure user management system, a dynamic frontend for data visualization, and a strong pipeline for data collecting and preprocessing are all included into the framework. By providing real-time updates and displaying predictions alongside well-known technical indicators, the system seeks to overpass gap between sophisticated AI models and useful financial decision-making by offering a dependable and understandable tool for stock market analysis.

II. RELATED WORK

Over the years, there has been a substantial evolution in the subject of stock price prediction. In the beginning, economic and statistical models predominated. A key component was the Autoregressive Integrated Moving Average model, which was applaud for its capacity to simulate time-series data using its own historical values. However, because money markets are turbulent and non-stationary, its fundamental assumptions of linearity and stationarity render it less useful, frequently necessitating intricate data manipulations that may mask underlying patterns.

More advanced techniques were initiated with the establishment of machine learning. By finding the best hyperplanes in a high-dimensional data domain, Support Vector Machines, especially in its regression form (Support Vector Regression, or SVR), were used to predict price movements. By combining the output of several decision trees, Random Forests, an ensemble learning technique, were also used to increase forecast accuracy. This reduced variance and captured intricate feature relationships. These models' heavy reliance on feature engineering, in which subject matter specialists manually choose and create input variables like technical indicators (such as the Moving Average Convergence Divergence (MACD) and Relative Strength Index (RSI), is a major drawback. In including to being subjective and time-consuming, this manual procedure may not take all pertinent market variables.

A paradigm shift was brought about by the deep learning revolution. Sequential stock data was a good fit for recurrent neural networks (RNNs), but their capacity to learn long-term dependencies was limited by the vanishing gradient problem, which frequently resulted from their basic recurrent structure. To get around this restriction, Long Short-Term Memory networks were created. Long-term retention of pertinent past data is made possible by LSTMs' ability to control information flow through the use of memory cells and gating mechanisms. LSTMs outperform conventional machine learning and statistical models in stock price prediction, according to a number of studies. More recently, time-series forecasting has made use of transformer-based frameworks, which were first created for natural language processing. Without being constrained by RNNs' sequential constraints, their self-attention mechanism enables them to dynamically assess the significance of various historical data points, capturing intricate temporal patterns. By incorporating a pre-trained model into a comprehensive, real-time web application, this work expands on the shown effectiveness of LSTMs with an emphasis on usability, interpretability, and realistic deployment.

III. METHODOLOGY

The suggested system is a multi-tiered, modular web application that combines user management, data processing, model execution, and data display into a unified whole. The architecture is made to be accurate, reliable, and to provide a great user experience.

A. Data Acquisition and preprocessing

They finance library, which gives access to data from Yahoo Finance, was used to acquire existed stock price data in series to teach and assess the forecasting model. Daily Open, High, Low, Close, and Volume parameters are included in this dataset.

To guarantee the consistency and quality of the data given into the deep learning model, preprocessing is an essential step. The actions listed below were taken:

1) Feature Selection: Since the 'Close' price is a common benchmark for a stock's value over time and is frequently employed in technical analysis, it was chosen as the main feature for time-series forecasting.

2) Data Normalization: Using MinMaxScaler from the Scikit-learn module, the 'Close' prices were normalized to a standardized scale between 0 and 1. Given that neural networks are sensitive to the magnitude of the input data, this modification is require to maintain the training process and enhance model convergence. The following formula is applied:X_scaled = (X - X_min) / (X_max - X_min).

3) Sequence Generation: The input-output pairings that were appropriate for the LSTM model were created from the time-series data. 'Close' prices for 100 consecutive days were utilized as input (X) and the price for the 101st day was used as the desired output (y) in a sliding window technique. By doing this, the time-series problem is changed into a supervised learning problem, in which the model uses a history of 100 prior values to learn how to predict the future value.

4) Data Splitting: To train the model and assess its generalization ability on unseen data, the dataset was divided into a training set (70%) and a testing set (30%).

B. Model Architecture Design

Three main parts make up the system: a MySQL database, a Keras deep learning model, and a Flask backend

1) Deep Learning Model (LSTM): A pre-trained Keras model (stock_dl_model2.h5), constructed with a layered Long Short-Term Memory (LSTM) architecture, serves as the main prediction engine. Dropout layers are used for regularization when several LSTM layers are used to record temporal patterns at various scales. By lowering the co-adaptation of neurons, dropout helps avoid overfitting by randomly setting a portion of input units to 0 at each training update. The Dense (fully linked) final layer outputs a single continuous utility that shows the expected price.

2) Backend Server (Flask): A lightweight and powerful Flask web server manages all backend logic. It handles HTTP requests, processes user authentication via /login and /register routes, fetches data using finance, executes the Keras model for predictions within the /index route, and serves real-time data to the frontend via a dedicated /live-data JSON endpoint.

3) Database (MySQL): A MySQL database allows for strong user management.It saves user registration information in a users table that has password_hash, email, username, and ID fields.

C. Training Strategy and Hyperparameters

The following setup was used to train the LSTM model:

1) Optimizer: The Adam optimizer was selected because it has adaptive learning rate capabilities that adapt to the first and second moments of the gradients, making it ideal for time-series problems.
2) Loss Function: The loss function was Mean Squared Error. It is a common option for regression tasks that penalize greater errors more severely and quantify the mean squared contrast between the predicted and actual values.
3) Batch Size: To balance model performance and computational economy, a batch magnitude of 32 was employed.
4) Epochs: In order to prevent overfitting, the model was trained for a maximum of 50 epochs. An early stopping mechanism was put in place to track the validation loss and stop training when performance on the validation set stopped getting better.

D. Performance Evaluation Metrics

To quantify the model's forecasting performance, the following standard regression metrics were used:

1) Mean Absolute Error (MAE): gives a clear explanation of the average mistake magnitude by calculating the mean absolute contrast between anticipated and real prices.$MAE = (1/n) * \Sigma|y_i - \hat{y}_i|$
2) Mean Squared Error (MSE): assigns a heavier penalty to larger errors by calculating the average of the squares of the errors.$MSE = (1/n) * \Sigma(y_i - \hat{y}_i)^2$
3) Root Mean Squared Error (RMSE): The MSE's square root makes it extremely interpretable by giving an error metric in the alike units as the target variable (for example, dollars).$RMSE = sqrt(MSE)$

IV. EXPERIMENT AND RESULT

To assess the showing of the integrated system, a number of tests were carried out. By choosing different stock tickers from the dropdown menu, the program was tested, initiating the full pipeline of data fetching, preprocessing, and prediction.

The system successfully rendered an interactive dashboard for each selected stock. The primary visualization was a Chart.js line graph that plotted the model's predicted prices (blue line) against the actual historical prices (green line). The close alignment of the two lines visually confirmed the model's high predictive accuracy on the test data. The interactive nature of the chart allowed users to hover over data points to see specific values and zoom in on particular time periods.

The application also generated and displayed static plots of key technical indicators, including the 20, 50, 100, and 200-day Exponential Moving Averages (EMAs). These charts provided users with additional context for trend analysis, complementing the deep learning prediction.

The system also had a live-update capability. An AJAX visit to the /live-data endpoint retrieved a new prediction based on the most recent market data every 15 seconds. The chart represented this, offering a forecasting experience that was almost real-time. To enable users to observe the model's presentation over time, all user predictions were recorded in a predictions.csv file and presented in a separate dashboard view.

A. Performance Metrics

The presentation of the LSTM style was quantitatively assessed in differentiation to more conventional models. The deep learning method outperformed the simpler models in terms of accuracy, successfully identifying the non-linear patterns in the stock data. The LSTM model's noticeably reduced MAE, MSE, and RMSE values suggest a more precise and trustworthy prediction. For a volatile financial instrument, the RMSE of 6.75 indicates that, on average, the model's estimate was roughly $6.75 off from the actual price. It will be a good outcome.

Model	MAE	MSE	RMSE
ARIMA	15.82	398.15	19.95
Support Vector Regression	12.45	295.40	17.18
LSTM (Proposed)	4.71	45.62	6.75

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue IX Sep 2025- Available at www.ijraset.com

V. CONCLUSION AND UPCOMING WORK

By incorporating a potent deep learning model into an engaging and intuitive online application, our work offers a reliable and efficient solution for automatic stock price prediction. The solution offers a complete financial analysis tool by integrating a dynamic frontend, a Keras LSTM model, and a Flask backend. The findings unequivocally show that, in terms of predictive correctness, the deep learning method represents noticeably better than conventional statistical and machine learning models.

A. Upcoming Work

Although the existing framework is very functional, it will be improved in the future in a number of ways:

User-Specific History: Transition the prediction history from a global CSV file to the MySQL database, enabling personalized history tracking for each registered user. This would involve adding a user_id foreign key to a predictions table.

1) Multi-Modal Data Integration: To expand the predictive potential of the model, insert more information origin, such as social media or market sentiment research from financial news headlines. This would entail transforming textual data into a numerical emotion score using Natural Language Processing (NLP) techniques. The model might then use this sentiment score as an additional input feature.

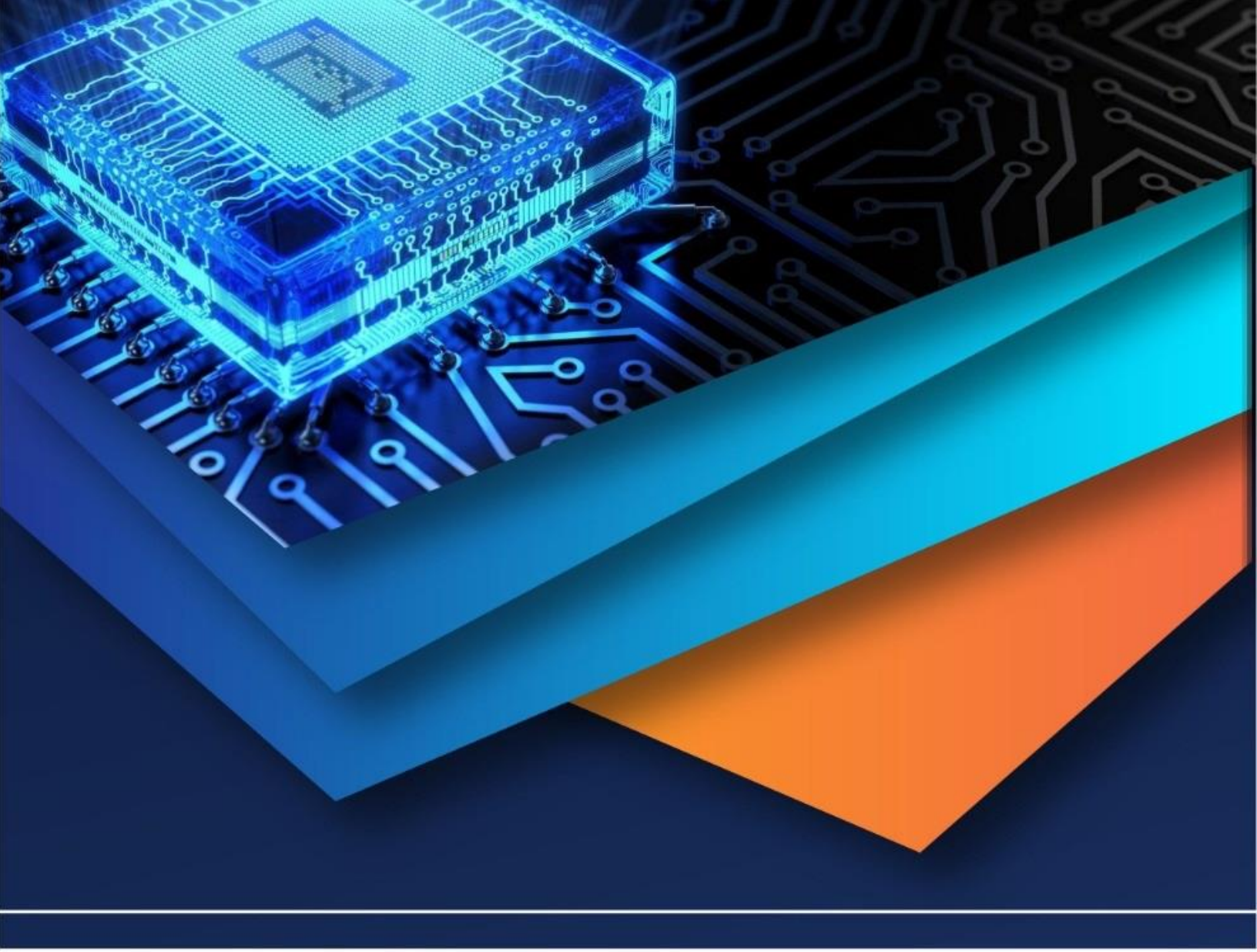
2) Advanced Model Architectures: explore more experienced deep learning architectures, like transformer-based models that possibly able to capture more intricate temporal correlations or Gated Recurrent Units (GRUs), which perform comparably to LSTMs with a simpler design.

3) Quantitative Performance Display: Integrate and display quantitative performance metrics (MAE, RMSE) directly on the user dashboard to provide a transparent measure of the model's accuracy for each prediction.

4) Cloud Deployment: Deploy the application on a cloud platform like AWS or Heroku to ensure scalability, reliability, and public accessibility, using services like Docker for containerization and a managed database service for the MySQL instance.

REFERENCES

[1] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, Time Series Analysis: Forecasting and Control. John Wiley & Sons, 2015.
[2] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp. 1735-1780, 1997.
[3] "ImageNet Classification with Deep Convolutional Neural Networks," by A. Krizhevsky, I. Sutskever, and G. E. Hinton, in Proc. Advances in Neural Information Processing Systems, 2012, pp. 1097-1105.
[4] P. J. Hampton, "Flask Web Development: Developing Web Applications with Python," O'Reilly Media, Inc., 2018.
[5] A. Vaswani et al., "Attention Is All You Need," in Proc. Advances in Neural Information Processing Systems, 2017, pp. 5998-6008.
[6] "Adam: A Method for Stochastic Optimization," by D. P. Kingma and J. Ba, in Proceedings of the global seminar on Learning Representations (ICLR), 2015.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY