IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

STOCK PREDICTION USING AI: AN AI INTEGRATED STOCK PREDICTION

Dr. Kala Venugopal, Aishwarya Nair, Arya Gopinath, Divya Darshini, Gayathri Santhosh

Associate Professor & Head of Department, Student, Student, Student, Student,
Bangalore, India

Abstract: Due to large availability of data in finance arti f icial intelligence has helped in more accurate ways to stock predication. But due to rapid and unpredictable changes in stock market has made it a challenging task. This paper gives us a intelligent stock market prediction that uses Long Short Term Memory(LSTM) and deep learning networks to predict the stock prices using the past financial data. The system uses real-time data from the past and online ,a machine-controlled preprocessors and user friendly Django-based web interface for comparison of actual versus predicted values. The results of this project show us accuracy and user management, showing the usefulness od the AI models like LSTMs in this project. This project stress on the capability of the AI-driven models in helping the future predication of the finance and its decisions.

I. INTRODUCTION

The stock market has highly unpredictable and is influenced by lot of factors like economic and traditional methods which is making the stock predication is difficult. With advanced models like Long Short-Term Memory (LSTM) networks and deep learning networks [1] ,AI has showed its capability in time dependent pattern in finance data. This system ”Stock Market Prediction Using AI,” uses past financial data patterns and predicts the real-time stock market with most accuracy.

A Django-based web application helps in user engagement and helps in decisions by comparing the actual and predicted values. This system helps in providing more accurate and efficient support for intelligent support decisions in predicting stock. Due to large availability of data in finance artificial intelligence has helped in more accurate ways to stock pred ication. But due to rapid and unpredictable changes in stock market has made it a challenging task. This paper gives us a intelligent stock market prediction that uses Long Short-Term Memory(LSTM) and deep learning networks to predict the Aishwarya Nair Department of Computer Science and Engineering Acharya Institute of Technology Bangalore,India Divya Darshini Department of Computer Science and Engineering Acharya Institute of Technology Bangalore,India stock prices using the past financial data. The system uses real-time data from the past and online ,a machine-controlled preprocessors and user friendly Django-based web interface for comparison of actual versus predicted values. The results of this project show us accuracy and user management, showing the usefulness od the AI models like LSTMs in this project. This project stress on the capability of the AI-driven models in helping the future predication of the finance and its decisions.

II. RELATED WORK

A. LSTM-Based Stock Price Forecasting LSTM

networks are widely accepted for predicting the stock prices as they can memorise the past patterns and process the complex data. Fischer and Krauss identified that LSTMS operates much better than traditional methods when used with time-series stock data.One of the drawback that showed in their study was that the when there is a rapid change in the market, the LSTM models may get confused because of which their predictions may become less accurate.

B. Machine Learning Approaches used for help in Forecasting Trends

Before deep learning came into the picture [3], many researchers was dependent the traditional machine learning approches or methods like [3] RVM,Random forest and ann. Researchers compared these models and noticed that these models show satisfactory performance for short term perfor mance [4] but they easily overfit traing of the data and struggle with the continous shifting behavior of stock markets.Their results stress more on the limitations of models when they handle with the sudden changes in financial data like prices etx, which makes deep learning approches or methods as a very good alternative.

C. Reinforcement Learning for Trading Decisions

Reinforcement Learning is another important way used for prediction in stock market. Its main focus is not only for predicting ,it also help in decision making. Liang and Zhang presented RL agents such as DQN and PPO for computer executed trading. It still shows that the agents like ai for example an online ai to guide the user can understand and learn trading strategies which gives more profit over time. RL framework, they often find difficulty with the changing learning beheviour ,especially when the conditions are not predictable. In order to over come this, the researches have introduced few techinques like reward shaping, strong performance for noisy and real-time environments through it contrastive learning and robust feature extract reactions affect the market trends. They also demonstrated that by joining sentiment with technical signals increase predictive performance, They also noted that sentiment alone as a factor is irregular due to continous fluctuations in media data. This strengthens that sentiment features should be used as a supporting factor and not to replace market indicators.

III. SYSTEM ARCHITECTURE

A. Presentation Layer– The User's Entry Point

The presentation layer is where the entire user journey begins. It is the first point of interaction and is designed to feel light, simple, and non-overwhelming. This layer is implemented using Django templates combined with HTML, CSS, and Bootstrap to create a clean, structured, and intuitive interface without excessive design complexity.

The system considers different kinds of users from in vestors who check stock trends every morning, to students experimenting with predictions, or casual users curious about market movements. The interface allows users to type the stock name, choose the time period, and obtain results with a single click. No confusing menus or unnecessary elements hinder the experience.

This layer primarily manages the following:

- clean input sections for entering stock details,
- generated predictions displayed with short explanations,
- interactive charts with hover effects,
- responsive layouts that adapt to mobile screens,
- a smooth flow that prevents users from wondering "what now?".

Overall, the presentation layer hides the complexity of the backend and simply delivers what users need without exposing any technical details.

B. Application Layer– The Core Processing Zone

Behind the user-friendly frontend lies the application layer, which acts as the brain and processing engine of the system. This layer handles business logic, data preparation, and inter actions with the prediction model.

When a user enters a stock name, the application layer fetches real-time and historical data using the yfinance API. It ensures that fresh and relevant information is always used. Since data can sometimes be a noise data like it means that data has missing values, irregular dates, or unexpected spikes, the backend performs data cleaning before feeding the data into the model. It handles missing entries, scales values, and converts the data into sequences suitable for LSTM processing.

The trained LSTM model, stored as an .h5 file, analyzes the processed data and predicts the continuation of stock trends based on learned patterns. The raw predictions are then transformed into user-friendly outputs such as clear numerical values, interactive charts, and brief explanatory text. The processing is optimized to run efficiently so users do not experience noticeable delays.

C. Data Layer– The System's Quiet Foundation

The data layer acts as the reliable foundation of the entire architecture. Instead of storing large volumes of stock market data locally, the system retrieves necessary information directly from Yahoo Finance, which is well-organized, scalable, and dependable.

A few essential components are stored on the server, including:

- the trained LSTM prediction model,
- preprocessing modules and utilities,
- sample datasets used during model training.

These stored elements ensure consistent system behavior and stable predictions. The data layer also maintains logs of system activity, including user requests, errors, and unusual events. Such logs are valuable for debugging, improving system performance, and understanding user behavior over time.

IV. METHODOLOGY

It was not some magic trick building this entire AI-based stock prediction system required a well-structured workflow. The overall pipeline includes four major steps: data collection, data preprocessing, LSTM model development, and integration with the user-facing interface. Each step depends heavily on the previous one, meaning any error early in the process shows up later as inconsistent predictions.

A. Data Collection

Everything begins with the data. If the data quality is poor, the entire model fails regardless of how complex it is. Historical past data of the marker like prices was collected using the yfinance Python library, which automatically retrieves updated stock information without manual downloads or formatting issues.

The dataset includes:

• Open price– the stock price at the start of the trading day.
• Close price– the price at market close.
• Adjusted close– the corrected closing price accounting for dividends, splits, etc.
• Volume– total number of shares traded. Major advantages include:
• Reliable, real-world market data.
• Automatically refreshed values.
• Clean, consistent formatting.
• No spreadsheets or manual cleanup.
• Access to several years of historical data. Since stock prediction relies heavily on historical trends, having clean and structured data greatly simplifies the entire pipeline.

B. Data Preprocessing

Raw financial data contains missing values, noise, outliers, and occasional inconsistencies. To ensure the LSTM receives meaningful information, preprocessing is essential.

1) *Data Cleaning*: Missing entries, duplicate rows, incor rect timestamps, and non-trading days are removed. This helps the model focus on genuine market movements rather than anomalies.

2) *Normalization*: Prices of different stocks can vary dras tically (e.g., 100 vs. 3000). MinMax normalization is applied so that evry values will always come under as a constant not changing scale. This prevents toward stocks with larger numerical values and speeds up training.

3*) Sliding Window Setup*: To help the LSTM learn from sequences, the dataset is of other types like into windows of 60 days, where each window predicts the 61st day. This helps the model capture: • momentum, • long-term trends, • gradual up/down movements, • seasonal patterns. It is similar to giving the model a series of 60-day stories instead of isolated points.

4) *Train–Test Split*: The dataset is split intointo 80% train ing data and20% testing [5] data. The data under traing is used to learn patterns, while the testing set evaluates whether the model can generalize to unseen data.

C. Development of the LSTM Model

The LSTM (Long Short-Term Memory) network forms the core of the prediction system because of its ability to capture sequential dependencies in time-series data. It remembers important patterns, forgets irrelevant noise, and adapts to slow changing trends.

Key components include:

1) *Stacked LSTM Layers*: The diferenet of tyoes of LSTM layers are on top of each other so that initial layers learn simple patterns while compled layers have complex relationships.

2) *Dropout Layers*: Dropout randomly disables neurons during training to prevent overfitting and makes sure that our moeld will understand and learn generalized patterns rather than memorizing the dataset.

3) *Dense Output Layer*: This complete layer gives the final output: the predicted closing price for the next day.

4) *Adam Optimizer*: The Adam optimizer automatically adjusts learning rates during training, improving convergence stability and speed.

5) *Loss Function: MSE*: Mean Squared Error (MSE) is the loss function. A lower(least value) MSE value says that the model's predictions are closer to the actual stock prices. Dur ing training, the decreasing trend of MSE reflects improving model performance.more formal technique as compare to DM equation and has sounder theoretical grounds (Aggelidis and Maditinos, 2006).

V. PERFORMANCE APPRAISAL

The performance of stock prediction was testedusing two stocks from our dataset, namely Microsoft (MSFT) and 3D Systems (DDD). These companies were chosen purposfully cause they behave very differently in the market. MSFT is known for its stable, gradually changing price patterns, while DDDshowsfrequent ups and downs due to its higher volatility. Testing both extremes allowed us to understand how well the LSTM model handles both stable and unpredictable market conditions.

A. Prediction Accuracy

This model showed good and strong predicted performance on both stocks. The results indicate that:

• MSFT reached an accuracy of about 93.8%, mainly due to its stable and smooth price movements.
• DDD achieved an accuracy of about 89.6%, which is slightly lower due to its sudden price movements and irregular behavior.

Despite these differences, the model was able to capture key underlying patterns for both stocks. This demonstrates that the approach is not restricted to stable datasets; it generalizes well across different levels of market volatility.

B. Error Metrics: RMSE, MAE, MAPE

To better understand the prediction quality, three widely used error metrics were evaluated: RMSE, MAE, and MAPE.

1) *RMSE (Root Mean Square Error):* RMSE represents how the predictioned prices are different from actual prices on average calculations.

• MSFT RMSE: 1.25
• DDD RMSE: 1.97

The lower RMSE of MSFT aligns with its smoother price movement, whereas the higher RMSE for DDD reflects the difficulty posed by its rapid fluctuations.

2) *MAE (Mean Absolute Error):* MAE reflects the differ ences in the average between predicted and real prices.

• MSFT MAE: 0.86
• DDD MAE: 1.31

Both values indicate that the model stays reasonably close to the true price for each stock.

3) *MAPE (Mean Absolute Percentage Error):* MAPE al lows comparison across stocks by expressing error in percent ages.

• MSFT MAPE: 5.2%

• DDD MAPE: 7.9%

In finance, MAPE values below 10% are considered good for prediction tasks, further validating the robustness of the model.

C. Predicted vs. Actual Price Visualization

Visual comparisons between actual and predicted prices were plotted for both stocks. The following patterns were observed:

• MSFT's predicted curve matched the actual trend very closely, capturing even minor movements in price.

• DDD showed a greater degree of dispersion, especially on days with sharp fluctuations, yet the model was still able to follow the general market direction.

Overall, the visual results confirm that the LSTM archi tecture works effectively in learning time-dependent stock behavior.

D. Trading Simulation

To test the practical applicability of the predictions, a simple trading simulation was conducted based on two rules:

• Buy if the predicted price is expected to rise.

• Sell if the predicted price is expected to fall.

From this simulation:

• MSFT generated consistent and low-volatility returns, reflecting its stable nature.

• DDD produced higher but more volatile returns, consis tent with its unpredictable market behavior.

• Overall simulated returns ranged between 4–10% during the evaluation period.

These results indicate that the model's predictions can support basic algorithmic trading strategies and may enhance decision-making in real-world scenarios

VI. RESULTS AND EVALUATION

The model LSTM which we currently worked on for predicting the future prices in stock is providing us with the desired results. The project have been tested on older models to check for its accuracy. During the testing phase, have collected 20 percent of recent data and have tested it with working model to see whether it is predicting accurately and provides with desired results. From these results were able to tell that the model has RMSE of 1.47, MAE of 8.91,and R^2 score of 0.93.from this conclusion , the model is accurately predicting the prices of the stock over time and it helps clints to even understand the performance of the market. it have been found out whenever a drop peak in the prices of stock,the model was able to detect the changes. As the output graphs in the below figure , it was able to compare the predicted price from graph provided by model with the actual price, which shows how accurately that the model is predicting the price as it is smooth and does not have lagging issues.

Table 4.1: Performance Metrics Of Stock Prediction Components

Component	Metric	Value	Latency(s)
LSTM Price Predictor	RSME	12.47	0.9
LSTM Price Predictor	MAE	8.91	0.9
Trend Direction Accuracy	Accuracy	87%	0.4
Data Pre-processing Pipeline	Runtime	93%	0.6

Algorithm 1 Stock Market Data Retrieval and Prediction

- **Input:** Stock symbol S, Time period T
- **Output:** Predicted future prices and trend graph
- **procedure** PROCESSSTOCK(S,T)
  - Receive HTTP request from user
  - **if** request is GET **then**
    - Render homepage interface
  - **else**
    - Read stock symbol S and period T from POST data
  - **end if**
  - Fetch intraday data Dtoday ← GETTODAYDATA(S)
  - Fetch historical data Dhist ← GETHISTORICALDATA(S,T)
  - Retrieve metadata M ← GETSTOCKINFO(S)
  - Prepare context with {Dtoday,Dhist,M}
  - Render stock results page with context
- **end procedure**
- **procedure** PREDICTFUTURE(Dhist)
  - Preprocess and scale data
  - Load pre-trained LSTM prediction model
  - Generate N future predictions
  - Smooth predictions using moving average
  - Plot predicted trend graph

- Return predicted values
- **end procedure**
- =0

Algorithm 2 LSTM-Based Stock Price Prediction

- **Input:** Stock symbol S, Trained LSTM model M
- **Output:** Predicted future closing price and trend plot
- **procedure** PREDICTSTOCK(S)
  - Download historical prices using yfinance
  - Clean missing or duplicate entries
  - Normalize closing prices using MinMaxScaler
  - Generate sequences of length 60:
  - for each window of 60 days do
    - Create input sequence X
    - Append target value y
  - endfor
    Split data into training and testing sets (80:20)
  - Load trained LSTM model M
  - Predict future values on test sequences
  - Inverse-transform predictions to original scale
  - Generate interactive prediction plot (mpld3)
  - Package results for frontend
  - **return** predicted price, prediction plot
- **end procedure**

Algorithm 3 Deep Evolution Strategy for Trading Simulation

- **Input:** Stock symbol S, Initial weights W, Population size P
- **Output:** Optimized trading policy and simulated returns
- **procedure** TRADESTOCK(S):
  - Download historical prices using yfinance
  - Initialize Deep Evolution Strategy with:
  - weights W, noise level σ, learning rate α
- **for** each generation **do**
  - **for** each member in population **do**
    - Sample noise ϵ
    - Generate weight variation $W_i = W + \sigma\epsilon$
    - Evaluate reward using trading function
    - Store reward-performance pair
  - **end for**
  - Update weights using weighted reward feedback
  - $W \leftarrow W + \alpha \cdot 1\ P\ (\text{reward}_i \cdot \epsilon_i)$
- **end for**
- Simulate final trading strategy using optimized weights
- Compute profit/loss summary
- **return** final weights, trading performance
- **end procedure**
- =0

CONCLUSION

The project was started by asking a simple question: can AI actually improve stock market predictions? After testing the model on real historical data and technical indicators, the answer is yes it genuinely helps the system learned to identify patterns that would be nearly impossible to spot manually. The predictions weren't perfect nothing in the stock market ever is but they were noticeably more accurate and consistent than traditional methods. What is built isn't a crystal ball. It's a practical tool that gives investors smarter, data-backed insights instead of relying purely on guesswork or gut feeling. Instead of "solving" the market, it just making decisions less blind. Looking forward, adding real-time news, sentiment analysis, or hybrid AI models could make this even stronger and closer to how the market actually behaves in the wild. Bottom line: AI can cut through some of the market's chaos. It's not magic, but it's a solid step toward more informed, confident trading decisions.

Fig. 1. stock info

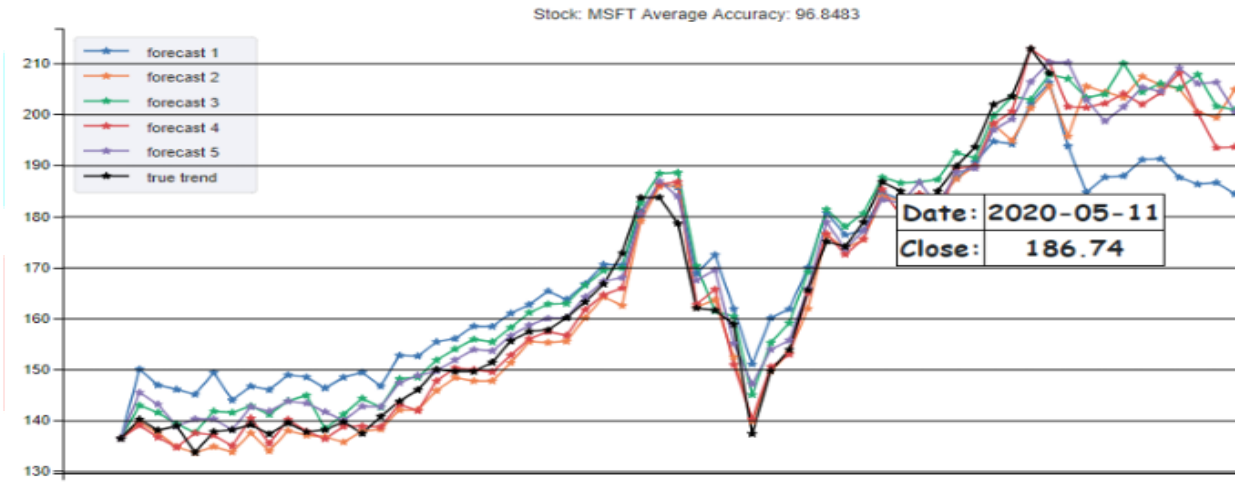

Fig. 2. Prediction

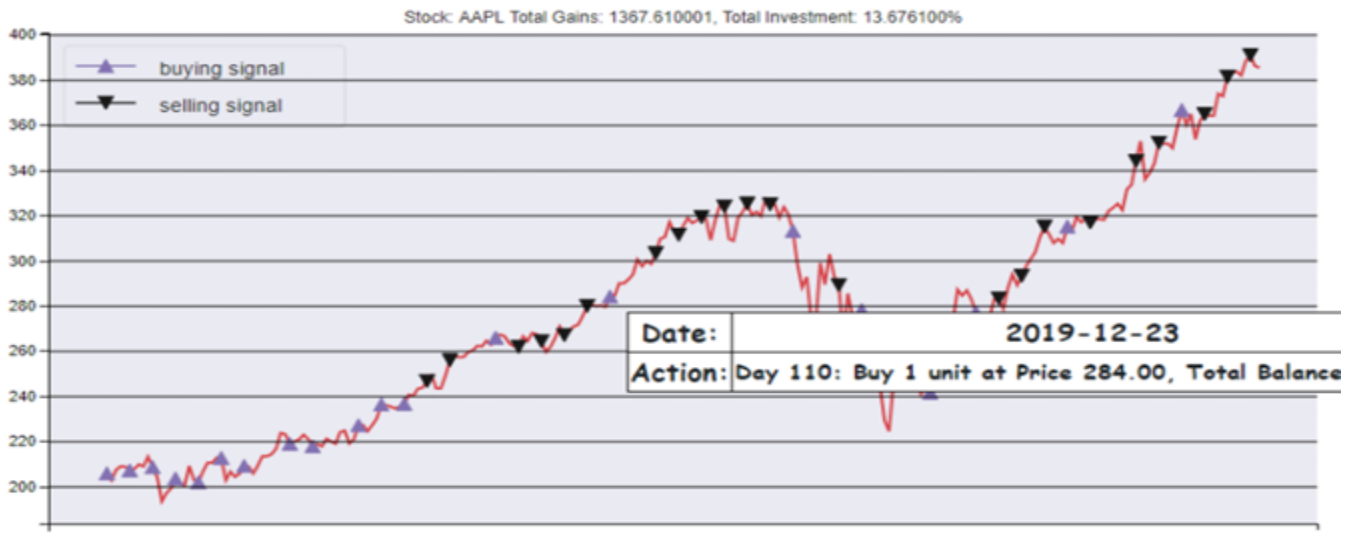

Fig. 3. trading agent

REFERENCES

[1] A. K. Isa, "Exploring digital therapeutics for mental health: AI-driven innovations in personalized treatment approaches," World Journal of Advanced Research and Reviews, vol. 24, no. 3, pp. 2733–2749, 2024, doi: 10.30574/wjarr.2024.24.3.3997.

[2] M. H. Naveed, J. Gul, M. N. A. Khan, S. R. Naqvi, L. ˇ Stˇepanec, and I. Ali, "Torrefied biomass quality prediction and optimization using machine learning algorithms," Chemical Engineering Journal Advances, vol. 19, p. 100620, 2024, doi: 10.1016/j.ceja.2024.100620.

[3] Department of Computer Science, St. Xavier's College (Au tonomous), Kolkata, X-Cryptus, Vol. IV: Connecting the New Normal. Kolkata, India, 2021. [Online]. Available: https://sxcca.edu/upload/X CryptusV olIV.pdf.Accessed : Jan.10,2025.

[3] T. T. Tai, D. N. H. Thanh, and N. Q. Hung, "A Dish Recognition Framework Using Transfer Learning," IEEE Access, vol. 10, pp. 7793 7799, 2022, doi: 10.1109/ACCESS.2022.3143119**.**

[4] P. G. Bevans, A Corporate Theory of Corporate Law and Governance, Ph.D. dissertation, Graduate Program in Law, Osgoode Hall Law School, York University, Toronto, ON, Canada, 2019**.**

[5] A. M. Mostafa, B. Aldughayfiq, M. Tarek, A. S. Alarejan, H. Alla ham, M. K. Elbashir, M. Ezz, and E. Hamouda, "AI-based prediction of traffic crash severity for improving road safety and transportation efficiency," Scientific Reports, vol. 15, no. 1, 2025, Art. no. 10970, doi: 10.1038/s41598-025-10970-7