International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org **Original Research Paper**

Indian Stock Market Sell and Buy Indication using Technical Indicators and Enhanced Bidirectional Long Short-Term Memory

Bhagyashree Pathak *[1], Dr. Snehlata Barde[2]

Submitted: 07/02/2024 **Revised:** 15/03/2024 **Accepted:** 21/03/2024

Abstract: This study introduces an innovative approach to signal generation for sell and buy decisions in the Indian Stock Market, leveraging Novel Technical Indicators and an Enhanced Bidirectional Long Short-Term Memory (BiLSTM) model. We evaluated various machine learning models, including Random Forest, Gradient Boosting, XGBoost, DenseNet, CNN-BiLSTM, LSTM, BiLSTM, and DANN, on their predictive performance using metrics such as MAPE, MAE, and computation time. Our proposed BiLSTM model, optimized with novel technical indicators, demonstrated superior performance with the lowest MAPE and competitive MAE, while maintaining a rapid computation time. These results highlight the efficacy of BiLSTM models in handling the sequential nature of stock data and the advantage of novel technical indicators in capturing intricate market trends. The proposed system holds the potential to revolutionize decision-making processes for traders and investors by providing highly accurate, real-time market predictions. The comparative analysis across diverse machine learning techniques showed that the proposed method significantly surpasses conventional models like Random Forest, Gradient Boosting, and XGBoost in terms of accuracy and efficiency. It achieved a remarkable reduction in Mean Absolute Percentage Error (MAPE) to nearly 0.03%, drastically lower Mean Absolute Error (MAE) at 10.45, and exhibited the fastest execution speed at 0.984 ms, highlighting its substantial advancement over existing approaches.

Keywords: Stock Data, Random Forest, Gradient Boosting, XGBoost, DenseNet, CNN-BiLSTM, LSTM, BiLSTM, DANN, Technical Indicators.

1. Introduction

The Indian stock market, characterized by its dynamic and volatile nature, presents a fertile ground for the application of advanced analytical techniques aimed at predicting market movements. In recent years, the proliferation of machine learning and deep learning methods has provided new avenues for financial analysts and traders to forecast stock prices with greater accuracy. The core challenge lies in the extraction of meaningful patterns from complex, noisy, and non-linear data that reflects the multifaceted factors influencing stock prices.

Traditional technical indicators have long been used to inform trading strategies, but their effectiveness can be limited in the face of rapidly changing market conditions and the vast amount of data generated by modern financial markets. In response, this research proposes the use of novel technical indicators that are designed to capture deeper insights into market trends and investor behavior.

The Bidirectional Long Short-Term Memory (BiLSTM) network, a variant of the conventional LSTM, is particularly well-suited to stock price prediction tasks due to its ability to process sequential data in both forward and backward directions. This allows the model to harness past and future context for better prediction accuracy. Enhanced BiLSTM models push this capability further, promising more nuanced detection of temporal dependencies and patterns that a unidirectional approach might miss.

In our comprehensive investigation, we compare an array of machine learning models, applying them to the Indian stock market context. The metrics of Mean Absolute Percentage Error (MAPE), Mean Absolute Error (MAE), and computational efficiency are used as benchmarks to evaluate performance. Our in-depth analysis not only assesses the predictive power of these models but also explores their practical applicability in real-world trading scenarios, gauging their potential to act as reliable indicators for selling and buying decisions.

Key contribution of this paper

1. Implement cutting-edge technical indicators.
2. Utilize an advanced Bidirectional Long Short-Term Memory (Bi-LSTM) approach.
3. Employ a composite method combining innovative technical indicators with the Enhanced Bi-LSTM.
4. Evaluate the outcomes using metrics such as Mean Absolute Percentage Error (MAPE) and Mean Absolute Error (MAE).

The paper is organized into six distinct sections: Section 1 provides a comprehensive introduction to the topic. Section 2 delves into the literature review, where prior research and

[1,2]Department of Computer Science and Engineering
[1] Research Scholar, SAGE University, Indore, (M.P.), India.
[2] Research Supervisor, Professor and Head of CSE/IT, SAGE University, Indore, (M.P.), India.

E-mail Id: [1]Bhagyashreepthk@gmail.com,
[2]snehlata.barde@sageuniversity.in , v.snehabarde@gmail.com
* Corresponding Author: **Bhagyashree Pathak**
Email: Bhagyashreepthk@gmail.com

existing knowledge in the field are thoroughly examined. Section 3 outlines the proposed methodology, detailing the innovative approach and analytical techniques employed in this study. Section 4 describes the implementation process, elaborating on the practical application of the proposed methodology. Section 5 presents the results and engages in a discussion about the findings, implications, and the significance of the research. Finally, Section 6 concludes the paper by summarizing the key takeaways and outlining potential avenues for future work, suggesting how this research can be extended and applied moving forward.

2. Literature Review

Sonkavde et al. (2023) , The financial sector has significantly influenced the economic well-being of consumers, traders, and financial institutions. In the modern era, artificial intelligence is reshaping the boundaries of financial markets through advanced machine learning and deep learning techniques. These methodologies find extensive applications in predicting financial instrument prices, analyzing market trends, identifying investment opportunities, optimizing portfolios, and more. Investors and traders increasingly employ machine learning and deep learning models for forecasting financial instrument movements. With the widespread adoption of AI in finance, this comprehensive review focuses on recent machine learning and deep learning models. It examines supervised and unsupervised algorithms, ensemble techniques, time series analysis, and deep learning algorithms for stock price prediction and classification. The contributions of this review encompass describing the models used in finance, presenting a generic framework for stock price prediction and classification, and implementing an ensemble model named "Random Forest + XG-Boost + LSTM." This ensemble model is evaluated for forecasting TAINIWALCHM and AGROPHOS stock prices and compared with popular machine learning and deep learning models. [1]

Mukherjee et al. (2023), Predicting the Stock Market is a challenging task that necessitates in-depth data analysis. Statistical models and artificial intelligence algorithms play a crucial role in achieving accurate predictions. Various machine learning and deep learning algorithms have demonstrated the ability to make predictions with minimal errors. Artificial Neural Networks (ANN) and Convolutional Neural Networks (CNN) are two extensively used models for stock market price prediction. These models forecast future data values based on historical data, employing a recursive approach. This study optimizes stock market prediction using deep learning techniques, achieving an accuracy of 97.66% with the ANN model and 98.92% with the CNN model. The CNN model uses 2-D histograms generated from quantized datasets within specific time frames. This innovative approach has not been previously applied to such datasets. The models are also tested on the impact of the recent COVID-19 pandemic, yielding an accuracy of 91%. [2]

Chandola et al. (2023), Stock market prediction is a challenging task due to its volatile nature. Deep Learning, known for its ability to analyze complex patterns in unstructured data, is gaining attention in stock market prediction. However, most methods overlook the influence of mass media on stock prices and investor behavior. This work proposes a hybrid deep learning model that combines Word2Vec and Long Short-Term Memory (LSTM) algorithms. The goal is to forecast the directional movement of stock prices based on financial time series data and news headlines. The binary output of the model aids investors in making informed decisions. The model's effectiveness is evaluated for predicting the directional movement of stock prices for five companies from different sectors. [3]

Alkhatib et al. (2022), Stock price prediction is a significant area of research with implications for individuals, corporations, and governments. This study introduces a new approach to predict the adjusted closing price of a specific corporation's stock. By expanding the feature set to include High, Low, Volume, Open, HiLo, and OpSe, the study aims to improve prediction accuracy. Datasets of different sizes and business sectors are used to assess the impact of data size and business sector on prediction accuracy. Six deep learning models, including MLP, GRU, LSTM, Bi-LSTM, CNN, and CNN-LSTM, are evaluated for their predictive performance using the enhanced feature set. Results show that LSTM-based models benefit from the new approach, and the added features positively impact prediction accuracy. [4]

Bhandari et al. (2022), The rapid advancement in artificial intelligence and machine learning techniques, coupled with the availability of large-scale data, presents an opportunity to develop sophisticated methods for predicting stock prices. This study employs a Long Short-Term Memory (LSTM) neural network architecture to predict the next-day closing price of the S&P 500 index. A well-balanced set of nine predictors, including fundamental market data, macroeconomic data, and technical indicators, is carefully constructed. Single-layer and multi-layer LSTM models are developed and compared using standard assessment metrics, such as Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE), and Correlation Coefficient (R). Results indicate that the single-layer LSTM model outperforms multi-layer models, providing a superior fit and higher prediction accuracy. [5]

Shah et al. (2022), Predicting stock market movements is essential, and modern technologies, including artificial intelligence and deep learning, have been instrumental in developing prediction models. However, combining automatic feature extraction and time series forecasting techniques in a stacked framework has not been explored in-depth. This article presents a framework based on Convolutional Neural Network (CNN) and Long-Short Term Memory (LSTM) to predict the closing price of the Nifty 50 stock market index. The CNN-LSTM model extracts features from a diverse set of inputs, including raw price data, foreign indices, technical indicators, currency exchange rates, and commodity prices. The model effectively predicts the closing price with a mean absolute percentage error of 2.54% over ten years of data. The proposed framework demonstrates significant improvements over traditional buy-and-hold strategies. [6]

Balakrishnan et al. (2023), Stock market prediction is challenging due to the noisy and non-stationary nature of financial data. Deep learning has shown promise in extracting meaningful patterns from large datasets for financial predictions. This paper presents a deep learning system that mines statistical laws and guides stock market operations using bottom neural network models and empirical mode decomposition. The model utilizes non-stationary and exponential financial time series to improve prediction accuracy. The results highlight the superior predictive efficiency of the EMD-based deep learner models. [7]

Prachyachuwong et al. (2021), Stock market prediction combines numerical and textual information to enhance predictive accuracy. This study employs a combination of Long Short-Term Memory Network (LSTM) and Bidirectional Encoder Representations from Transformers (BERT) architectures to predict daily stock market activity. By dividing news headlines into industry-specific indexes, the model considers both numerical and textual information, outperforming baseline models in terms of accuracy and returns. [8]

Albahli et al. (2022), Predicting stock prices is challenging due to market volatility. This research uses ten years of Yahoo Finance data and Stock Technical Indicators (STIs) to predict closing stock prices using 1D DenseNet and an autoencoder. The proposed approach outperforms existing techniques, achieving a minimum Mean Absolute Percentage Error (MAPE) of 0.41. [9]

Azevedo et al. (2023), Machine learning models are applied to predict stock anomalies based on a vast dataset of international stock-month observations from 1980 to 2019. The models show significant returns, and composite predictors outperform baseline benchmarks, even with transaction costs. Neural networks, extending rolling windows, elastic net feature reduction, and percent-ranked returns yield the highest return predictability. [10]

Khan et al. (2023), Deep reinforcement learning (DRL) is applied to predict stock prices, offering a promising approach to capture complex patterns and make decisions based on sequential actions. The framework incorporates a deep neural network and Q-learning to learn optimal trading actions. Tuning hyperparameters significantly impacts performance, and the DRL-based approach outperforms traditional methods in terms of accuracy and prediction errors. [11]

Mehtab et al. (2022), Predicting stock prices is a popular research area with differing viewpoints on its feasibility. This study presents a robust framework using deep learning-based regression models for accurate stock price forecasting. Historical stock price data of a well-known company listed on the National Stock Exchange (NSE) of India is used, recorded at five-minute intervals from December 31, 2012, to January 9, 2015. Four Convolutional Neural Network (CNN) and five Long- and Short-Term Memory (LSTM) models are constructed, validated, and tested for performance. The study evaluates model performance in terms of execution time and Root Mean Square Error (RMSE) values. [12]

Sen et al. (2023), This research focuses on stock price prediction using hybrid models that combine machine learning and deep learning techniques. NIFTY 50 index values from the National Stock Exchange (NSE) of India are used from December 29, 2014, to July 31, 2020. Eight regression models are built using training data from December 29, 2014, to December 28, 2018. Four deep learning-based regression models using Long- and Short-Term Memory (LSTM) networks are also constructed with a novel walk-forward validation approach. The study optimizes LSTM model hyperparameters and presents extensive results, demonstrating the accuracy of the LSTM-based univariate model. [13]

Ghosh et al. (2021), Amid the COVID-19 pandemic, global financial markets experienced significant volatility. This study aims to measure market fear through implied and historic volatility. It employs India VIX and 20-day rolling standard deviation of NIFTY returns as volatility measures. Macro-economic constructs, technical indicators, and Google search volume index are used as explanatory features. Supervised Boruta feature selection is applied to select significant features. Advanced machine and deep learning algorithms such as Gradient Boosting, Extra Tree Regression, Deep Neural Network, and Long Short Term Memory Network are used to predict market fear. Explainable AI frameworks are used to analyze feature influence. The findings suggest effective predictability of both India VIX and historic volatility, providing actionable insights. [14]

Tschora et al. (2022), Electricity prices in the European market are highly volatile due to various production sources and storage constraints. Accurate price prediction is essential for intelligent electricity usage. This study explores machine learning techniques to predict electricity prices, incorporating previously unused predictive features like price histories of neighboring countries. The study demonstrates the improvement in forecasting quality with these features, even during periods of sudden changes. Shap values are used to analyze feature contributions to model predictions, providing insights into how models make predictions and building user confidence in the models. [15]

Mehtab et al. (2021), Stock price prediction has been a significant research area, with debates on the feasibility of accurate predictions. While some adhere to the efficient market hypothesis, arguing against predictability, formal propositions demonstrate that precise modeling and appropriate variables can lead to accurate stock price predictions. This work introduces a hybrid modeling approach for stock price prediction, incorporating various machine learning and deep learning models. NIFTY 50 index values from the National Stock Exchange (NSE) of India are used from December 29, 2014, to July 31, 2020. Eight regression models are built using training data from December 29, 2014, to December 28, 2018. Additionally, four deep learning-based regression models using Long- and Short-Term Memory (LSTM) networks are constructed with a novel walk-forward validation approach. The LSTM models' hyperparameters are optimized using grid-searching to achieve validation convergence. Extensive results are presented, highlighting the LSTM-based univariate model as the most accurate for predicting NIFTY 50 open values. [16]

Nõu et al. (2023), Accurate stock market predictions are essential, but finding the best approach can be challenging. This paper conducts a thorough analysis of predictive accuracy for various machine learning and econometric approaches, including ARMA, GARCH, random forest, SVR, KNN, and GARCH-ANN, in predicting returns and volatilities on the OMX Baltic Benchmark Price Index. The results show that machine learning methods generally outperform autoregressive moving average models across multiple metrics. [17]

Avramov et al. (2023), This study explores the profitability of deep learning-based investments in challenging stock market conditions. It reveals that deep learning signals are profitable for difficult-to-arbitrage stocks, especially during high limits-to-arbitrage market states. However, profitability diminishes when excluding microcaps, distressed stocks, or during episodes of high market volatility. Machine learning-based performance is also impacted by trading costs due to high turnover and extreme positions. Despite these challenges, machine learning methods effectively identify mispriced stocks and remain profitable in long positions and recent years, with low downside risk. [18]

Shen et al. (2020), In the era of big data, deep learning has gained popularity for stock market price and trend prediction. This paper presents a comprehensive approach that includes data preprocessing, multiple feature engineering techniques, and a customized deep learning-based system for predicting stock market price trends. The study evaluates various machine learning models and demonstrates the superiority of the proposed solution due to its comprehensive feature engineering. The system achieves high accuracy in stock market trend prediction, contributing to both financial and technical research communities in stock analysis. [19]

Nasiri et al. (2023), Financial time series prediction has been a subject of significant interest. Decomposition-based methods have shown promise in this field, but they often approximate a single function, leading to suboptimal results. Furthermore, most research has focused on one-step-ahead forecasting, limiting decision-making for future market investments. This study introduces two innovative multi-step-ahead stock price prediction methods based on distinct decomposition techniques: Discrete Cosine Transform (DCT) and Variational Mode Decomposition (VMD). DCT-MFRFNN, leveraging DCT and Multi-Functional Recurrent Fuzzy Neural Network (MFRFNN), uses DCT to reduce time series fluctuations and simplifies its structure, while MFRFNN predicts stock prices. VMD-MFRFNN combines VMD's advantage of decomposing the input signal into Intrinsic Mode Functions (IMFs) with MFRFNN for prediction. Both DCT-MFRFNN and VMD-MFRFNN employ Particle Swarm Optimization (PSO) to train MFRFNN. Notably, this research introduces the use of the gradient descent method to train MFRFNN for the first time. The proposed methods are evaluated using three financial time series datasets, with experimental results demonstrating the superior performance of VMD-MFRFNN, achieving a 31.8% decrease in RMSE compared to MEMD-LSTM. DCT-MFRFNN also outperforms MFRFNN and DCT-LSTM in all experiments, highlighting the positive impact of DCT on MFRFNN's performance. The effectiveness of PSO in training VMD-MFRFNN is verified by comparing it with twelve different metaheuristic approaches, with PSO achieving a 9.4% reduction in MAPE compared to other methods. [20]

Muthukumaran et al. (2023), Small and medium-sized enterprises (SMEs) often struggle with evaluating credit risks due to limited data consistency. Predicting financial crises or business failures becomes crucial. This paper introduces an Optimal Deep Learning-based Financial Crisis Prediction (ODL-FCP) model for SMEs. The ODL-FCP approach includes Archimedes optimization algorithm-based feature selection (AOA-FS) and optimal deep convolutional neural network with long short-term memory (CNN-LSTM) based data classification. Hyperparameter optimization of the CNN-LSTM method uses the sailfish optimization (SFO) algorithm. Experimental results using benchmark financial datasets show that the ODL-FCP technique outperforms other methods. [21]

Zhong et al. (2019), Big data analytics and machine learning are becoming increasingly important in various domains, including stock market investment. This paper focuses on forecasting daily stock market returns, using deep neural networks (DNNs) and traditional artificial neural networks (ANNs). The study employs 60 financial and economic features to predict the daily direction of the SPDR S&P 500 ETF returns. DNNs and ANNs are applied to the entire dataset and two datasets transformed via principal component analysis (PCA). Notably, the DNNs' classification accuracy patterns are analyzed as the number of hidden layers gradually increases from 12 to 1000. The results reveal the superiority of DNNs using PCA-represented datasets in terms of classification accuracy, with corresponding trading strategies performing slightly better than other tested methods and benchmarks. [22]

Garai et al. (2023), Wavelet decomposition is a widely used technique in signal processing. In agriculture, the adoption of machine learning (ML) algorithms for tasks such as price prediction is increasing. This study explores the combination of wavelet-based models and ML for predicting agricultural commodity prices. Various wavelet filters, including Haar, Daubechies (D4), Coiflet (C6), best localized (BL14), and least asymmetric (LA8), are considered. Daily wholesale price data of onions from major Indian markets (Bengaluru, Delhi, and Lasalgaon) is utilized. The performance of wavelet-based models is compared with benchmark models, revealing that the combination models generally outperform others. Notably, wavelet decomposition with the Haar filter followed by the application of the random forest (RF) model yields the best prediction accuracy among all models. [23]

Tripathi et al. (2023), Bitcoin, a volatile asset, presents challenges for accurate price forecasting due to its dependence on various factors. This research proposes a forecasting framework that reduces noise in Bitcoin time series data and leverages fundamental indicators, technical indicators, and lagged prices as predictors. The approach includes feature selection, outlier handling, and the use of deep neural networks (DNNs) tuned by Bayesian Optimization for short-term price prediction. The results demonstrate that a DNN model using technical indicators as input outperforms other benchmark models, achieving a high level of accuracy even during extreme market conditions. This work contributes to the fields of feature selection, data preprocessing, and hybridizing deep learning models. [24]

Sen et al. (2023), Stock price prediction is a well-studied area with differing viewpoints on its predictability. This study proposes a hybrid approach to stock price prediction using five deep learning-based regression models. Daily NIFTY 50 index data from the National Stock Exchange (NSE) of India is used. The models are built based on historical data from December 29, 2014, to December 28, 2018, and are used to predict open values from December 31, 2018, to July 31, 2020. Multi-step prediction with walk-forward validation is employed, and model parameters are optimized using the grid-search technique. The results show that both CNN and LSTM-based models are accurate in forecasting NIFTY 50 open values, with the CNN model using previous week's data as input being the fastest, while the encoder-decoder convolutional LSTM model using the previous two weeks' data achieves the highest forecasting accuracy. [25]

Amalov et al. (2020), This study focuses on predicting significant changes in stock prices using machine learning algorithms, particularly neural network classifiers. Three neural network models, including multilayer perceptron, convolutional net, and long short-term memory net, are constructed and tested. These models aim to forecast substantial changes in stock prices based on prior changes. Benchmark models, such as random forest and relative strength index methods, are used for comparison. The research uses a decade's worth of daily stock price data from four major US public companies. The results reveal that neural network classifiers can predict significant stock price changes with high accuracy, outperforming studies that predict only price direction. [26]

3. Proposed Methodology

3.1 Proposed architecture

In figure 1 shows the India Stock Dataset is assembled with Fundamental Indicators, Technical Indicators, and Daily Close Prices. Feature selection is then conducted using the Boruta Method to reduce multicollinearity and filter out highly correlated features. The data undergoes preprocessing to eliminate outliers and noise, and is normalized. This data is split into training and testing datasets. A variety of models, including various LSTM types (such as T-LSTM and Bi-LSTM), DANN, and their ensembles, are trained on these datasets. The best model is then chosen based on its ability to predict stock prices at different horizons: next day, 3 days, weekly, and monthly. Performance is evaluated using metrics such as MAE, RMSE, and MAPE to ensure the model's accuracy and reliability in stock price prediction.

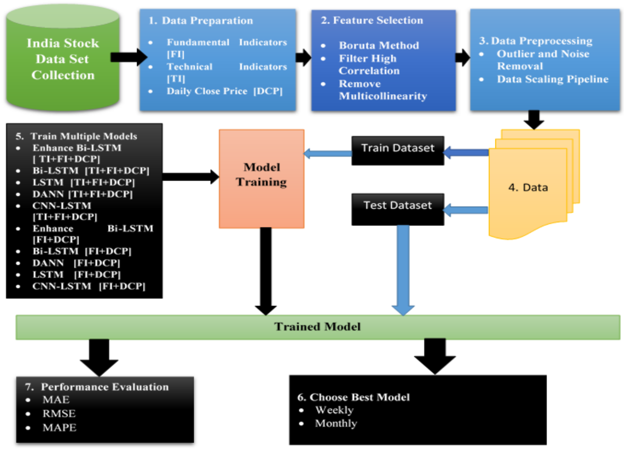

Figure 1. Proposed architecture

3.2 Rules for Investing in Nifty – Bank Nifty (Index) - Updated 27 Dec 2023

BUY Strategies

1. **BUY 1**:
- Trigger: When Nifty closes below the Lower Bollinger Band (LBB) on a monthly basis (20,2 settings).

SELL Strategies

1. **Sell 1**:
- Trigger 1: When 9-period RSI on a monthly timeframe is greater than 84.
- **Rebuy Strategy**:
  - Rebuy 50% when Nifty touches the 20-day Simple Moving Average (SMA) on a weekly basis.
  - Rebuy the remaining 50% when Nifty touches the weekly LBB.
  - Trigger 2: If Nifty closes above the recent highest close on a monthly timeframe.
2. **BUY 2**:
   - Trigger: Buy when there is a weekly close below the weekly LBB, followed by a green candle (where open < close).
   - Condition: Execute the buy at the close of the green candle only if the stock is trading above the 20 SMA on a monthly basis.
3. **Sell 2**:
   - If Nifty is below the Monthly 20 SMA:
     - (a) Sell 50% at Weekly 20 SMA, 25% at Weekly Upper Bollinger Band (UBB), and hold the rest until it closes below the Weekly 20 SMA.
     - (b) If Nifty touches the Weekly 20 SMA and forms a red candle, sell.
   - If Nifty closes above Monthly 20 SMA, follow monthly rules for exit or profit booking.

Stop Loss (SL) Strategies

1. **SL 1**:

- After a weekly buy, if the stock touches or goes above the 20 SMA on a weekly basis and then falls below it without touching the UBB, and subsequently closes below the LBB a second time after the first purchase, consider the stop loss as hit.

2. **SL 2**:
   - A close below the Monthly 20 SMA on a weekly basis.

3.3 Rules for Investing in High Quality Stocks of Nifty 50 to Nifty 200

Terms used - (LBB = lower Bollinger Band, UBB = upper Bollinger Band, SMA = Simple moving average, SL = Stoploss, weekly close = Friday or last working day of week, monthly close = last working day of month, green candle = open < close, red candle = open > close)
Point 1 – Returns to be calculated on % basis.
Point 2 – Success Rate (profit trades vs loss trades)
Point 3 – Buy and hold returns (from starting to end of dates, in % basis)
Point 4 – Profit Factor (gross profit/gross loss)
Point 5 – Avg trade returns (Profit % / no. of trades)

Buy Conditions for Stocks:

BUY 1 : When stock closes below or near (+1%) Monthly Lower Bollinger Band(LBB) (20,2) **OR** weekly close < Weekly LBB.

SELL 1 : When touches 20 SMA line but close making RED CANDLE (open > close)

SELL 2 : When 1 candle closes above 20 SMA monthly but 2nd candle closes below it.

[**REBUY :** When sold due to close below M-20SMA and again close above it without creating BUY1.]

SELL 3 : When Monthly RSI(9) goes above 80, hold till its above 80 and sell when it drops below 72.

[**REBUY @ Weekly LBB touch or close. Or close above recent high on monthly basis.]**

SELL 4 : Sell on the following first red candle(open < close) if Monthly RSI(9) closes near or above 88.

[**REBUY @ Weekly LBB touch or close. Or close above recent high on monthly basis.]**

SELL 5 : Close above UBB (20,3).

StopLoss 1 : when another following candle closes below LBB Monthly (20,2) > 10%

StopLoss 2 : when 20 % loss on closing basis Monthly candle.

StopLoss 3 : when Monthly close below LBB and (close < Buy price) after a close above LBB is formed.

IMPORTANT : if above SL 3 is hit, not to buy that stock till its closes above 20 SMA Monthly.

BUY 2 : Buy when (Low < LBB Monthly- but month not complete) **AND** (weekly close < LBB weekly)

SELL 1 : when 1 candle closes above 20 SMA monthly but 2nd candle closes below it. **REBUY :** When sold due to close below M-20SMA and again close above it without creating BUY1.

SELL 2 : when Monthly RSI(9) goes above 80, hold till its above 80 and sell when it drops below 72.

REBUY @ Weekly LBB touch or close.

SELL 3 : Sell on the following first red candle(open < close) if Monthly RSI(9) closes above 88.

REBUY @ Weekly LBB touch or close.

BUY 3 : Buy when find a **weekly** close below weekly LBB, then wait for a green candle with +2.5% (open < close),buy at its close.

If stock is trading **ABOVE** 20 SMA **Monthly**.
Trade according to BUY1 plan.
If stock is trading **BELOW** 20 SMA **Monthly, then**
Book 50% at W-20SMA touch and close in red candle total 50% on W-UBB, 25% at W-UBB touch and rest at W-20SMA below close.
SL 1 : After weekly buy, if stock goes above 20 SMA weekly and comes down again below it without touching UBB and again closes below LBB (for 2nd time after 1st time buy), consider as SL is hit.
SL 2 : Close below Monthly 20 SMA(previous) on weekly basis.
SL 3 : 15 % on weekly closing basis.
BUY 4 : When stock touch LBB (20,3) Monthly (between the month)
SELL conditions if same month close is above LBB.
SELL 1 : Sell 50% with profit of 10 %
SELL 2 : Sell 25% with profit of 20 %
SELL 3 : Sell ALL at monthly close.
SL : Monthly close below the low of 'buy price monthly candle'.
SELL conditions if same month close is below LBB.
SELL 1 : when touches 20 SMA line but close making RED CANDLE (open > close)
SELL 2 : when 1 candle closes above 20 SMA monthly but 2nd candle closes below it.
REBUY : When sold due to close below M-20SMA and again close above it without creating BUY1.
SELL 3 : when Monthly RSI(9) goes above 80, hold till its above 80 and sell when it drops below 72.
REBUY @ Weekly LBB touch or close.
SELL 4 : Sell on the following first red candle(open < close) if Monthly RSI(9) closes above 88.
REBUY @ Weekly LBB touch or close.
HOLD : if 2 close above 20 SMA Monthly, then hold till stock closes 20 SMA monthly.

SL 1 : when another following candle closes below LBB Monthly (20,2) > 10%
SL 2 : when 15 % loss on closing basis Monthly candle.
SL 3 : when Monthly close below LBB and (Buy price > close) after a close above LBB is formed.
Imp : if above SL 3 is hit, not to buy that stock till its closes above 20 SMA Monthly.
BUY 5 : when stock touch LBB (20,3) WEEKLY(between the week)
SELL conditions if same month close is above LBB.
SELL 1 : Sell 50% with profit of 10 %
SELL 2 : Sell 25% with profit of 20 %
SELL 3 : Sell ALL at weekly touching 20SMA.
SL : Weekly close below the 'buy price weekly candle'.

Universal Exit : if stock goes above UBB(20,3) - Exit

3.4 Proposed Methdology

1. Data Set Collection

1. **Financial Data Providers:**
   - Paid financial data providers offer comprehensive and reliable stock market data with a wide range of features and historical data. Some well-known providers include:
   - National Stock Exchange
2. **Public Stock Exchanges:**
   - Many stock exchanges provide access to their historical trading data, including stock prices, volume, and trade details. Examples include:
   - National **Stock Exchange** of India (NSE) [NIFTY]
   - Bombay stock exchange (BSE) [Sensex]
3. **Financial Data APIs:**
   - Several APIs provide access to stock market data, both historical and real-time. Some popular options include:
   - Google
   - Yahoo finance
   - Zerodha
   - These APIs allow you to fetch data programmatically in various formats (e.g., JSON, CSV).

2. Data preparation

Data preparation for stock market analysis typically involves collecting, cleaning, and transforming data into a suitable format for analysis. In our case, we have mentioned three types of data: Fundamental Indicators (FI), Technical Indicators (TI), and Daily Close Price (DCP). Here's how you can prepare each of these types of data:

1. **Fundamental Indicators (FI):**
   - Fundamental indicators include financial metrics related to a company's performance, such as earnings, revenue, and balance sheet data. These indicators are crucial for fundamental analysis.
   - Data sources for fundamental indicators can include financial statements, company reports, and financial databases.
   - Steps for data preparation:
     - Collect the necessary fundamental data for the stocks you are interested in.
     - Ensure that the data is consistent and accurate by cleaning it. This may involve dealing with missing values, handling outliers, and ensuring data consistency across different sources.
     - Transform the data into a structured format, such as a DataFrame or CSV file, with each row representing a company and each column representing a specific fundamental indicator.
     - Merge or join this data with other relevant datasets if needed, such as the stock symbols or company names.

2. **Technical Indicators (TI):**

- Technical indicators are derived from historical price and volume data and are used for technical analysis. Common examples include moving averages, Relative Strength Index (RSI), and MACD (Moving Average Convergence Divergence).
- To prepare technical indicators:
  - Collect historical price and volume data for the stocks you are analyzing. This data is often available from financial data providers or APIs.
  - Calculate the desired technical indicators based on the historical data. Libraries like Pandas or technical analysis libraries like TA-Lib in Python can help with this.
  - Store the calculated technical indicators in a structured format similar to how you stored fundamental indicators.

3. **Daily Close Price (DCP):**

- Daily close prices represent the closing price of a stock at the end of each trading day.
- Data sources for daily close prices include financial data providers, stock exchanges, and APIs.
- Preparing daily close prices involves:
  - Collecting historical daily close price data for the stocks you are interested in.
  - Checking and cleaning the data for any irregularities or missing values.
  - Structuring the data into a time-series format, where each row represents a specific date, and each column represents a different stock or asset's closing price.

4. Data Intervals

In the context of the stock market, data intervals refer to the time intervals at which stock price and trading data are collected, recorded, and analyzed. Different intervals provide different levels of granularity and are used for various purposes in financial analysis. Here are some common data intervals used in the stock market:

1. **Interval 1: Daily Data**
   - Daily data summarizes the trading activity for a single trading day, providing the opening price, closing price, highest price (high), and lowest price (low) for the day.
   - It also includes the total trading volume for the day.

- Daily data is widely used for long-term trend analysis, fundamental analysis, and portfolio management.

2. **Interval 2: Weekly and Monthly Data**
    - Weekly or monthly data aggregates price and volume information over an entire week or month.
    - It typically includes opening, closing, high, and low prices for the chosen interval.
    - Weekly or monthly data is primarily used for longer-term investment strategies and portfolio allocation decisions.

6. Feature selection

Feature selection is crucial when working with stock data or any financial dataset to build effective predictive models. Here are some common feature selection techniques for stock data:

1. Boruta Method:

- The Boruta method is a wrapper-based feature selection technique that helps identify relevant features by comparing them with a shuffled version (random noise) of the dataset.
- Steps for using the Boruta method:
    - Create a shadow dataset by randomly permuting the values of the target variable.
    - Combine the original dataset with the shadow dataset.
    - Train a machine learning model (e.g., Random Forest) on both datasets and assess feature importance.
    - Compare the importance scores of actual features with those of the shadow features.
    - Retain features that have significantly higher importance scores than their shadow counterparts.

2. Filter High Correlation:

- In financial datasets like stock data, some features may be highly correlated with each other, providing redundant information.
- Calculate the pairwise correlation between features (e.g., using Pearson's correlation coefficient).
- Remove one of the features from highly correlated pairs, keeping only the one that is more relevant or representative of the information.

3. Remove Multicollinearity:

- Multicollinearity occurs when two or more independent variables in a regression model are highly correlated, making it difficult to interpret the individual effects of these variables on the dependent variable.
- Techniques to address multicollinearity include Principal Component Analysis (PCA) or using domain knowledge to remove one of the correlated features.

7. Data preprocessing

Data preprocessing is an essential step in the data preparation process before building machine learning models. It involves several tasks, including outlier and noise removal and data scaling. Here's an overview of each of these steps:

1. **Outlier and Noise Removal:**
    - **Outliers:** Outliers are data points that significantly deviate from the rest of the data. They can skew the results of your machine learning model and should be handled carefully.
        - Common techniques to deal with outliers include:
    - Z-score or Standardization: Calculate the Z-score for each data point and remove or transform data points with Z-scores beyond a certain threshold.
    - IQR (Interquartile Range) method: Identify outliers based on the IQR and remove or transform data points lying outside the acceptable range.
    - Visual inspection: Plotting the data and visually identifying outliers can also be a helpful initial step.

- **Noise:** Noise refers to random or irrelevant variations in the data that can obscure meaningful patterns.
    - Techniques to reduce noise include:
        - Smoothing techniques such as moving averages or median filters.
        - Data aggregation to reduce the granularity of the data.
        - Feature selection to focus on the most relevant attributes and reduce dimensionality.

2. **Data Scaling Pipeline:**
    - Data scaling is necessary when the features in your dataset have different scales or units. Machine learning algorithms often perform better when the input features are on a similar scale.
    - Common data scaling techniques include:
        - **Min-Max Scaling (Normalization):** Scales the data to a specific range, typically between 0 and 1.
        - **Standardization (Z-score scaling):** Transforms the data to have a mean of 0 and a standard deviation of 1.
        - **Robust Scaling:** Scales the data based on the median and interquartile range, making it more robust to outliers.
        - **Log Transformation:** Useful when dealing with skewed data distributions.

LSTM (Long Short-Term Memory), Bi-LSTM (Bidirectional Long Short-Term Memory), and Enhanced Bi-LSTM are all recurrent neural network (RNN) architectures used for sequential data processing, including time series data like stock prices. Here's a comparison of these three architectures:

3.5 Bi-LSTM (Bidirectional Long Short-Term Memory):

Step 1: Data Preparation
1.1 Load historical stock market data, including features like Daily Close Price (DCP) and potentially other relevant data such as Technical Indicators (TI) and Fundamental Indicators (FI).
1.2 Split the dataset into training and testing sets.

Step 2: Data Preprocessing
2.1 Normalize or scale the features, ensuring they have zero mean and unit variance to help the model converge efficiently.

2.2 Create sequences of input data and corresponding target values for training. For example, create a sliding window of historical data as input sequences and the next day's DCP as the target value.
2.3 Handle missing data and outliers as needed. Common techniques include interpolation or replacing missing values with the mean or median.

Step 3: Model Architecture
3.1 Initialize a Bi-LSTM model using a deep learning framework such as TensorFlow or PyTorch.
3.2 Define the architecture of the Bi-LSTM model. It typically consists of:
- One or more Bidirectional LSTM layers with a specified number of units (neurons) in each layer.
- Optionally, dropout layers to prevent overfitting.
- A final output layer, often a dense layer, with a single neuron for regression or multiple neurons for classification.

Step 4: Training
4.1 Specify the loss function (e.g., mean squared error for regression or categorical cross-entropy for classification) and an optimizer (e.g., Adam or RMSprop).
4.2 Train the Bi-LSTM model using the training data:
- Choose the number of epochs (iterations) and batch size.
- During each epoch, shuffle and batch the training data to avoid overfitting.
- For each batch, compute predictions using the forward pass and calculate the loss.
- Perform backpropagation to update the model's weights.
- Monitor training progress and check for convergence.

Step 5: Evaluation
5.1 Evaluate the trained Bi-LSTM model on the testing dataset:
- Compute evaluation metrics such as mean squared error (MSE), root mean squared error (RMSE), or accuracy, depending on the task (regression or classification).
- Visualize the model's predictions against actual values for a visual assessment.

Step 6: Prediction
6.1 Use the trained Bi-LSTM model to make predictions on new or unseen data.
6.2 Input the historical data and any relevant features to predict future stock prices or perform other tasks.

3.6 Enhanced Bi-LSTM

Step 1: Data Preparation
1.1 Load stock market dataset with Fundamental Indicators (FI) and Daily Close Price (DCP).
1.2 Split the dataset into training and testing sets.

Step 2: Data Preprocessing
2.1 Normalize or scale the FI and DCP features.
2.2 Create sequences of input data and corresponding target values for training.
2.3 Handle missing data and outliers as needed.

Step 3: Model Architecture
3.1 Initialize an Enhanced Bi-LSTM model.
3.2 Define LSTM layers for both forward and backward directions.
3.3 Optionally, add dropout layers to prevent overfitting if necessary.

Step 4: Training
4.1 Specify the loss function (e.g., mean squared error) and optimizer (e.g., Adam).
4.2 Train the model using the training data.
4.3 For each epoch:
  4.3.1 Shuffle and batch the training data.
  4.3.2 For each batch:
    4.3.2.1 Perform a forward pass: Compute predictions.
    4.3.2.2 Compute the loss between predictions and actual values.
    4.3.2.3 Perform a backward pass: Update model weights using gradient descent.
  4.3.3 Calculate validation loss and monitor for convergence.
  4.3.4 If validation loss doesn't improve significantly, stop training.

Step 5: Evaluation
5.1 Evaluate the model's performance on the testing dataset.
5.2 Calculate relevant evaluation metrics (e.g., mean squared error, accuracy).

Step 6: Prediction
6.1 Make predictions for future stock prices or any other relevant task.

4. Implementation

4.1 Setup Library :

TensorFlow
- **Purpose**: An open-source machine learning library developed by Google. It's used for numerical computation using data flow graphs, widely known for its applications in deep learning but also supports traditional machine learning.

Scikit-Learn
- **Purpose**: A Python library for machine learning. It offers simple and efficient tools for data mining and data analysis. It is built on NumPy, SciPy, and Matplotlib and provides a range of supervised and unsupervised learning algorithms.

NumPy
- **Purpose**: A library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. It's foundational for scientific computing in Python.

Keras
- **Purpose**: An open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library. It's known for its user-friendliness, modularity, and ease of extensibility.

PyTorch

- **Purpose**: An open-source machine learning library based on the Torch library, used for applications such as computer vision and natural language processing, primarily developed by Facebook's AI Research lab. It's known for its flexibility and dynamic computational graph.

LightGBM

- **Purpose**: A gradient boosting framework that uses tree-based learning algorithms. It is designed for distributed and efficient training, particularly on large datasets. Developed by Microsoft, it's known for its high speed and performance.

Eli5

- **Purpose**: A Python library used for explaining machine learning models. It helps in understanding the decisions made by a machine learning model, which is crucial for model debugging and interpretability.

SciPy

- **Purpose**: An open-source Python library used for scientific and technical computing. It builds on NumPy and provides a large number of higher-level functions that operate on numpy arrays and are useful for different types of scientific and engineering applications.

Theano

- **Purpose**: An open-source project that was developed to define, optimize, and evaluate mathematical expressions, especially matrix-valued ones. Although it's no longer actively developed, it was influential and laid the groundwork for many modern deep learning frameworks.

Pandas

- **Purpose**: A software library written for data manipulation and analysis in Python. Offers data structures and operations for manipulating numerical tables and time series. It's essential for data wrangling, preparation, and analysis.

4.2 Dataset

4.2.1 BSE / NSE

Table 1. Dataset Description.

S.No.	Company name	Index	Stock	Total number of year	Number of Attributes	Time Frame
1	Nifty	yes		2000-2023	22	Weekly and Monthly
2	Asian Paints		Yes	2000-2023	22	Weekly and Monthly
3	Infosys		Yes	2000-2023	22	Weekly and Monthly
4	Tata Motors		Yes	2000-2023	22	Weekly and Monthly

1. https://www.bseindia.com/
2. https://www.nseindia.com/

4.2.2 Dataset Features Description

In financial analysis, the integration of weekly and monthly data points provides a comprehensive view of market dynamics. Weekly data, including the date (WDate), opening (WOpen), high (WHigh), low (WLow), and closing prices (WClose), alongside monthly statistics such as the date (MDate), opening (MOpen), high (MHigh), low (MLow), and closing prices (MClose), offer a foundational understanding of short and long-term trends. The application of technical indicators enhances this analysis; for monthly data, the 20-period Simple Moving Average (MClose_20_SMA), Bollinger Bands with two and three standard deviations (MClose_Upper_BB_2, MClose_Lower_BB_2, MClose_Upper_BB_3, MClose_Lower_BB_3), and the Relative Strength Index over 9 periods (MClose_RSI_9) provide insights into volatility, potential overbought or oversold conditions, and momentum. Similarly, for weekly data, the 20-period Simple Moving Average (WClose_20_SMA), Bollinger Bands with two and three standard deviations (WClose_Upper_BB_2, WClose_Lower_BB_2, WClose_Upper_BB_3, WClose_Lower_BB_3), and the Relative Strength Index over 9 periods (WClose_RSI_9) offer a nuanced understanding of market sentiment on a shorter timescale. Together, these indicators help investors and analysts discern patterns, identify potential support and resistance levels, and make informed decisions based on historical price movements and statistical measures of volatility and momentum.

4.3 Evaluation Parameters

Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE)

- **MAE (Mean Absolute Error)** measures the average magnitude of the errors in a set of predictions, without considering their direction.
- **RMSE (Root Mean Squared Error)** is the square root of the mean square error. It's more sensitive to outliers than MAE.

- **MAPE (Mean Absolute Percentage Error)** measures the accuracy of a forecast as a percentage, and is useful for understanding the proportion of error compared to actual values.

Mean Absolute Error (MAE):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \qquad (1)$$

Where:

- n is the number of observations.
- y_i is the actual value.
- $\hat{y}_i$ is the predicted value.
- The absolute difference between actual and predicted values is summed up and then divided by the number of observations.

Root Mean Squared Error (RMSE):

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \qquad (2)$$

Where:

- This is the square root of MSE.
- RMSE is in the same units as the target variable, making it more interpretable than MSE.

Mean Absolute Percentage Error (MAPE):

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \qquad (3)$$

Where:

- This measures the error as a percentage of the actual values.
- It's useful for understanding the proportion of error in comparison to actual values.

4.4 Illustrative Example

4.1.1 Asian paints weekly investing rules results

Table 2. Asian paints weekly investing rules results

Serial	**Above or below 20 SMA Monthly**	**Type of BUY**	**Date of Buying**	**Stock Value @ BUY**	**1st Exit Date**	**1st EXIT Value-50 % - 20 SMA W.**	**2nd Exit Date**	**2nd EXIT Value-25 % - UBB W.**	**3rd Exit Date**	**3rd EXIT Value-25% - Close < 20 SMA or acc to monthly buy sell**	**In Trade for no. of days**	**Absolute % Return - Investing rules**	**Yearly % Return- Investing rules**	**Commulative Returns - Investing rules**
1	Above	BUY 2	17-May-04	28	25-06-2004	31.48	25-06-2004	31.48	25-06-2004	31.48	39	**12.43**	116.3	12.43
2	Above	BUY 1	02-Dec-04	30.81	10-Feb-06	75.5	10-Feb-06	75.5	10-Feb-06	75.5	435	**145.05**	121.7	157.48
3	Above	BUY 2	26-May-06	52.9	28-Feb-08	111.5	28-Feb-08	111.5	28-Feb-08	111.5	643	**110.78**	62.9	268.26
4	Above	BUY 2	09-Mar-07	61	28-Feb-08	111.5	28-Feb-08	111.5	28-Feb-08	111.5	356	**82.79**	84.9	351.05
5	Above	BUY 1	25-Jul-08	111.7	10-Oct-08	96.86	10-Oct-08	96.86	10-Oct-08	96.86	77	**-13.29**	-63	337.76
6	Below	BUY 1	19-Dec-08	89.94	30-Sep-10	266.32	30-Sep-10	266.32	30-Sep-10	266.32	650	**196.11**	110.1	533.87

7	Above	BUY 1	18-Feb-11	259.91	16-Dec-11	271.27	16-Dec-11	271.27	16-Dec-11	271.27	301	**4.37**	5.3	538.24
8	Below	BUY 1	06-Jan-12	266.06	01-Jun-12	413	01-Jun-12	413	01-Jun-12	413	147	**55.23**	137.1	593.47
9	Above	BUY 2	05-Oct-12	345	30-Apr-15	762.45	30-Apr-15	762.45	30-Apr-15	762.45	937	**121**	47.1	714.47
10	Above	BUY 1	15-Jun-15	740	31-Oct-16	1069.9	31-Oct-16	1069.9	31-Oct-16	1069.9	504	**44.58**	32.3	759.05
11	Above	BUY 1	25-Nov-16	945.45	23-Dec-16	874	23-Dec-16	874	23-Dec-16	874	28	**-7.56**	-98.6	751.49
12	Above	BUY 1	12-Oct-18	1261.2	31-Dec-20	2764.5	31-Dec-20	2764.5	31-Dec-20	2764.5	811	**119.2**	53.6	870.69
13	Above	BUY 1	24-May-19	1384.2	31-Dec-20	2764.5	31-Dec-20	2764.5	31-Dec-20	2764.5	587	**99.72**	62	970.41
14	Above	BUY 1	10-Apr-20	1651.35	31-Dec-20	2764.5	31-Dec-20	2764.5	31-Dec-20	2764.5	265	**67.41**	92.8	1037.82
15	Above	BUY 1	11-Mar-22	2932	31-May-22	2859.9	31-May-22	2859.9	31-May-22	2859.9	81	**-2.46**	-11.1	1035.36
16	Below	BUY 1	24-Jun-22	2760.9	31-Dec-22	3087.9	31-Dec-22	3087.9	31-Dec-22	3087.9	190	**11.84**	22.7	1047.2
17	Below	BUY 1	03-Mar-23	2828.85	27-Oct-23	2955.15	27-Oct-23	2955.15	27-Oct-23	2955.15	238	**4.46**	6.8	1051.66
18	Above	BUY 1	10-Nov-23	3098.15	27-Dec-23	3404.45	27-Dec-23	3404.45	27-Dec-23	3404.45	47	**9.89**	76.8	1061.55

Table 3. Asian paints monthly investing rules results

Serial	Date of Buying	Type of BUY	BUYING PRICE	Date of Exit	EXIT Value	Absolute % Return - Investing rules	Yearly % Return-Investing rules	Returns from date of first Buy	Comulative Returns - Investing rules	% saved due to exits & re-entries	Comulative % saved	Profit in no. of times outperformance of nifty
1	01-Oct-03	Buy 1	27.44	31-Jan-04	32.25	17.5	52.4	18	18			1
2	03-Jan-05	RE BUY	35.99	01-Mar-06	64.6	79.5	68.8	135	97	-10.4	-10	0.90
3	19-May-06	RE BUY	57.41	28-Feb-08	111.5	94.2	52.9	306	191	12.5	2	1.01

4	04-Jul-08	REBUY	103.22	31-Oct-08	95.48	-7.5	-23.0	248	184	8.0	10	1.09
5	06-Mar-09	Buy 1	73.64	30-Sep-10	266.32	261.7	166.7	871	445	29.7	40	1.41
6	04-Jan-11	REBUY	248.62	31-Dec-11	259.24	4.3	4.3	845	450	7.1	47	1.51
7	31-Jan-11	REBUY	299.53	30-Apr-15	762.45	154.5	36.4	2679	604	-13.5	33	1.31
8	12-Jun-15	REBUY	705.65	31-Oct-16	1069.9	51.6	37.2	3799	656	8.0	42	1.41
9	11-Nov-16	REBUY	962.3	31-Dec-16	891.05	-7.4	-54.1	3147	648	11.2	53	1.57
10	31-Jan-17	REBUY	970.7	31-Dec-20	2764.5	184.8	47.2	9975	833	-8.2	45	1.44
11	04-Mar-22	REBUY	2738.15	31-May-22	2859.75	4.4	18.4	10322	838	1.0	45	1.46
12	31-Jul-22	REBUY	3333.15	31-Dec-22	3087.75	-7.4	-17.6	11153	830	-14.2	31	1.25
13	31-Jan-23	Buy 1	2725.85	31-Oct-23	2995.7	9.9	13.2	10817	840	13.3	45	1.42
14	30-Nov-23	Buy 1	3119.9	27-Dec-23	3404.45	9.1	123.3	12307	849	-4.0	41	1.36
												Outperformed asian paints 'Buy & hold' returns by 1.36 times.

4.1.2 Tata Motors weekly investing rules results

Table 4. Tata Motors weekly investing rules results

Serial	Above or below 20 SMA Monthly	Type of BUY	Date of Buying	Stock Value @ BUY	1st Exit Date	1st EXIT Value-50 % - 20 SMA W.	2nd Exit Date	2nd EXIT Value-25 % - UBB W.	3rd Exit Date	3rd EXIT Value-25% - Close < 20 SMA or acc to monthly buy sell	In Trade for no. of days	Absolute % Return - Investing rules	Yearly % Return-Investing rules	Commulative Returns - Investing rules
1	Below	Buy 1	20-Apr-01	13.35	20-May-01	16.5	25-May-01	15.75	25-May-01	15.75	35	20.79	216.8	20.79

2	Above	Buy 2	21-May-04	70	25-Mar-05	79.26	25-Mar-05	79.26	25-Mar-05	79.26	308	13.23	15.7	34.02
3	Below	Buy 1	01-Apr-05	83	09-Apr-05	80.99	09-Apr-05	80.99	09-Apr-05	80.99	8	-2.42	-110.4	31.6
4	Below	Buy 1	22-Apr-05	83.74	31-May-06	150.98	31-May-06	150.98	31-May-06	150.98	404	80.3	72.5	111.9
5	Above	Buy 1	28-Jul-06	139.59	30-Mar-07	139.52	30-Mar-07	139.52	30-Mar-07	139.52	245	-0.05	-0.1	111.85
6	Below	Buy 1	13-Apr-07	139.15	20-Jul-07	147.07	20-Jul-07	147.07	27-Jul-07	133.95	105	3.33	11.6	115.18
7	Below	Buy 2	25-Jan-08	121	25-Jan-08	132	25-Jan-08	136	31-Jan-08	142.7	6	12.13	737.9	127.31
8	Below	Buy 1	25-Apr-08	122.48	02-May-08	132.49	09-May-08	128.24	09-May-08	128.24	14	6.44	167.9	133.75
9	Above	Buy 1	01-Jul-11	196.83	29-Jul-11	187.61	29-Jul-11	187.61	29-Jul-11	187.61	28	-4.68	-61	129.07
10	Below	Buy 1	01-Sep-11	149.35	18-Nov-11	168.49	18-Nov-11	168.49	18-Nov-11	168.49	78	12.82	60	141.89
11	Below	Buy 1	19-Jun-15	433.05	10-Jul-15	401.9	10-Jul-15	401.9	10-Jul-15	401.9	21	-7.19	-125	134.7
12	Below	Buy 1	12-Feb-16	318.2	22-Apr-16	417	22-Apr-16	417	05-May-17	419.6	448	31.25	25.5	165.95
13	Above	Buy 1	09-Dec-16	464.1	05-May-17	419.6	05-May-17	419.6	05-May-17	419.6	147	-9.59	-23.8	156.36
14	Below	Buy 1	15-Sep-17	401.75	06-Oct-17	424.85	06-Oct-17	424.85	06-Oct-17	424.85	21	5.75	99.9	162.11
15	Below	Buy 1	02-Mar-18	370.75	09-Mar-18	341.5	09-Mar-18	341.5	09-Mar-18	341.5	7	-7.89	-411.4	154.22
16	Above	Buy 1 & 2 both	04-Mar-22	417.25	23-Dec-22	378.35	23-Dec-22	378.35	23-Dec-22	378.35	294	-9.32	-11.6	144.9

Table 5. Tata Motors monthly investing rules results

Serial	**Date of Buying**	**Type of BUY**	**BUYING PRICE**	**Date of Exit**	**EXIT Value**	**Absolute % Return - Investing rules**	**Yearly % Return-Investing rules**	**Returns from date of first Buy**	**Commulative Returns - Investing rules**	**% saved due to exits & re-entries**	**Comulative % saved**	**Profit in no. of times outperformance of nifty**
1	01-Nov-01	Rebuy	20.9	29-Feb-04	97.49	366.5	157.4	366	366			1

2	14-May-04	Rebuy	79.37	31-Mar-05	79.26	-0.1	-0.2	279	366	22.8	23	1.23
3	31-Jul-05	Rebuy	92.14	31-May-06	150.98	63.9	76.7	622	430	-14.0	9	1.06
4	16-Jun-06	Rebuy	131	31-Mar-07	139.78	6.7	8.5	569	437	15.3	24	1.22
5	25-Jan-08	Buy 4	111	31-Jan-08	127.5	14.9	904.3	510	452	25.9	50	1.53
6	31-Mar-08	Buy 1	119.3	31-May-08	110.3	-7.5	-45.1	428	444	6.9	57	1.64
7	01-Jul-09	Rebuy	83.42	31-Jan-11	226.98	172.1	108.5	986	616	32.2	89	2.17
8	25-Feb-11	Rebuy	212	31-Jul-11	187.61	-11.5	-26.9	798	605	7.1	96	2.32
9	01-Jan-12	Rebuy	241.16	31-Dec-14	490.29	103.3	34.4	2246	708	-22.2	74	1.80
10	29-May-15	Rebuy	481.65	30-Jun-15	434.35	-9.8	-112.0	1978	698	1.8	76	1.84
11	30-Sep-15	Buy 1	298.6	30-Jun-17	432.55	44.9	25.6	1970	743	45.5	121	2.67
12	31-Mar-18	Buy 1	326.85	31-May-18	282.5	-13.6	-81.2	1252	730	32.3	154	3.54
13	01-Nov-20	Rebuy	180.35	28-Feb-22	454.05	151.8	114.4	2072	881	56.6	210	5.54
14	11-Mar-22	Rebuy	418.15	23-Dec-22	378.35	-9.5	-12.1	1710	872	8.6	219	6.02
15	31-Jan-23	Rebuy	452.1	28-Feb-23	420.7	-6.9	-90.5	1913	865	-16.3	202	5.03
16	30-Apr-23	Rebuy	484.95	27-Dec-23	740.9	52.8	79.9	3445	918	-13.2	189	4.37
												our system beat "buy and hold" for stock by 4.37 times.
												Dividents not included in calculations.

4.1.3 Nifty weekly rules investing rules results

Table 6. Nifty weekly rules investing rules results

Serial	Above or below 20 SMA Monthly	Type of BUY	Date of Buying	Nifty Value @ BUY	1st Exit Date	1st EXIT Value-50 % - 20 SMA W.	2nd Exit Date	2nd EXIT Value-25 % - UBB W.	3rd Exit Date	3rd EXIT Value-25 % - Close < 20 SMA or acc to monthly buy sell	AVG SELL PRICE	In Trade for no. of days	Absolute % Return - Investing rules	Yearly % Return - Investing rules	Commulative Returns - Investing rules
1	Above	Buy 1	26-May-00	1274.35	23-Jun-00	1472.2	21-07-2000	1397	21-07-2000	1397	1434.60	56	12.58	82	12.58
2	Below	Buy 1	27-Oct-00	1178.7	22-Dec-00	1242	22-Dec-00	1242	22-Dec-00	1242	1242.00	56	5.37	35	17.95
3	Below	Buy 1	20-Apr-01	1144	19-Apr-02	1100.3	15-Mar-02	1169.75	19-Apr-02	1100.3	1117.66	364	-2.3	-2.3	15.65
4	Below	Buy 1	28-Sep-01	913.85	19-Apr-02	1100.3	15-Mar-02	1169.75	19-Apr-02	1100.3	1117.66	203	22.3	40.1	37.95
5	Below	Buy 1	07-Jun-02	1048.8	07-Mar-03	1017.1	30-Jan-03	1089.59	07-Mar-03	1017.1	1035.22	273	-1.29	-1.7	36.66
6	Below	Buy 1	01-May-03	938.3	31-Dec-03	1879.75	31-Dec-03	1879.75	31-Dec-03	1879.75	1879.75	244	100.34	150.1	137
7	Above	Buy 1	04-Jun-04	1521.1	31-Mar-06	3402.55	31-Mar-06	3402.55	31-Mar-06	3402.55	3402.55	665	123.69	67.9	260.69
8	Above	BUY 2	11-Aug-06	3274.35	31-Oct-07	5900.65	31-Oct-07	5900.65	31-Oct-07	5900.65	5900.65	446	80.21	65.6	340.9
9	Above	Buy 1	23-Mar-07	3861.05	31-Oct-07	5900.65	31-Oct-07	5900.65	31-Oct-07	5900.65	5900.65	222	52.83	86.9	393.73
10	Above	Buy 1	27-Mar-08	4942	06-Jun-08	4627.8	06-Jun-08	4627.8	06-Jun-08	4627.8	4627.80	71	-6.36	-32.7	387.37
11	Below	Buy 1	11-Jul-08	4049	14-Aug-08	4430.7	14-Aug-08	4430.7	14-Aug-08	4430.7	4430.70	34	9.43	101.2	396.8
12	Below	Buy 1	31-Oct-08	2885.6	20-Feb-09	2736.4	20-Feb-09	2736.4	20-Feb-09	2736.4	2736.40	112	-5.17	-16.8	391.63

13	Above	Buy 1	18-Feb-11	5458.95	17-Jun-11	5366.4	17-Jun-11	5366.4	17-Jun-11	5366.4	5366.40	119	-1.7	-5.2	389.93
14	Below	Buy 1	02-Sep-11	5040	11-Nov-11	5168.85	11-Nov-11	5168.85	11-Nov-11	5168.85	5168.85	70	2.56	13.3	392.49
15	Above	Buy 1	19-Apr-13	5783.1	16-Aug-13	5507.85	16-Aug-13	5507.85	16-Aug-13	5507.85	5507.85	119	-4.76	-14.6	387.73
16	Below	Buy 1	11-Sep-15	7789.3	06-Nov-15	7954.3	06-Nov-15	7954.3	06-Nov-15	7954.3	7954.30	56	2.12	13.8	389.85
17	Below	Buy 1	19-Feb-16	7210.75	08-Apr-16	7555.2	08-Apr-16	7555.2	08-Apr-16	7555.2	7555.20	49	4.78	35.6	394.63
18	Below	Buy 1	25-Nov-16	8114.3	04-Oct-18	10316.45	04-Oct-18	10316.45	04-Oct-18	10316.45	10316.45	678	27.14	14.6	421.77
19	Above	Buy 1	02-Nov-18	10553	02-Aug-19	10997.35	02-Aug-19	10997.35	02-Aug-19	10997.35	10997.35	273	4.21	5.6	425.98
20	Above	Buy 1	09-Aug-19	11109.65	28-Feb-20	11201.75	28-Feb-20	11201.75	28-Feb-20	11201.75	11201.75	203	0.83	1.5	426.81
21	Below	Buy 1	27-Mar-20	8660.25	12-Jun-20	9972.9	12-Jun-20	9972.9	12-Jun-20	9972.9	9972.90	77	15.16	71.9	441.97
22	Going down from M-RSI > 84	Buy 1	11-Mar-22	16630.45	25-Mar-22	17153	25-Mar-22	17153	25-Mar-22	17153	17153.00	14	3.14	81.9	445.11
23	Going down from M-RSI > 84	Buy 1	21-May-22	16266.15	17-Jun-22	15293.5	17-Jun-22	15293.5	17-Jun-22	15293.5	15293.50	27	-5.98	-80.8	439.13
24	Below	Buy 1	24-Jun-22	15699.25	17-Mar-23	17100.05	17-Mar-23	17100.05	17-Mar-23	17100.05	17100.05	266	8.92	12.2	448.05
25	Above	Buy 1	31-Mar-23	17359.75	15-Dec-23	21448.7	15-Dec-23	21448.7	15-Dec-23	21448.7	21448.70	259	23.55	33.2	471.6

Table 7. Nifty monthly rules investing rules results

Serial	**Date of Buying**	**Type of BUY**	**Nifty Value @ BUY**	**Date of Exit**	**EXIT Value**	**Absolute % Return - Investing rules**	**Yearly % Return - Investing rules**	**Nifty Returns from date of first Buy**	**Commulative Returns - Investing rules**	**% saved due to exits & re-entries**	**Comulative % saved**	**Profit in no. of times outperformance of nifty**
1	30-Sep-01	Buy 1	913.85	31-Mar-03	978.2	7.0	4.70	7	7			1.00

2	30-Jun-03	Buy2	1134.15	30-Dec-03	1879.75	65.7	131.12	106	73	-13.8	-14	0.86
3	14-May-03	Buy3	1582.4	30-Mar-06	3402.55	115.0	39.95	272	188	18.8	5	1.02
4	09-Jun-06	Buy3	2770	30-Oct-07	5900.65	113.0	81.21	546	301	22.8	28	1.26
5	25-Jan-08	Buy3	4540.03	30-Jun-08	4040.55	-11.0	-25.58	342	290	30.0	58	1.64
6	30-Oct-08	Buy 1	2885.6	01-Jul-11	5482	90.0	33.72	500	380	40.0	98	2.29
7	30-Dec-11	Buy 1	4624.6	30-Aug-13	5471.8	18.3	10.98	499	398	18.5	116	2.72
8	30-Sep-13	Buy2	5735.3	30-Oct-14	8322.2	45.1	41.68	811	443	-4.6	112	2.59
9	04-Sep-15	Buy3	7655.05	30-Nov-15	7935.25	3.7	15.36	768	447	8.7	121	2.82
10	29-Feb-16	Buy 1	6987.05	30-Aug-19	11023.2	57.8	16.50	1106	505	13.6	134	3.20
11	30-Sep-19	Buy2	11474.5	28-Feb-20	11201.8	-2.4	-5.74	1126	502	-3.9	130	3.07
12	31-Mar-20	Buy 1	8597.75	30-Aug-21	17132.2	99.3	70.08	1775	602	30.3	160	4.00
13	14-May-22	Buy3	15782.2	30-Jun-22	15780.2	0.0	-0.10	1627	602	8.6	169	4.35
14	31-Jul-22	Buy2	17158.2	19-Dec-23	21500	25.3	18.25	2253	627	-8.0	161	4.00
												means 4 times returns than original nifty returns....!!!!

5. Result and Discussion

In this section expalin the result and dsicusion about optimization parapameter existing methods with proposed methods.

Table 8. Network architecture suggested by Bayesian Optimization

Hyperparameter	**CNN-BiLSTM (technical indicators)**	**LSTM (univariate)**	**BiLSTM (technical indicators)**	**DANN (fundamental indicators)**	**DANN (technical indicators)**	**Proposed Enhanced Bi-LSTM (technical indicators)**
Learning rate	0.0013	0.0038	0.0039	0.0019	0.0019	0.0009
Dropout rate	0.0055	0.2000	0.2000	0.0019	0.0251	0.0042
Number of Hidden Layers	5	4	5	3	3	5
Neuron units per layer	467	467	467	497	497	497
Batch size	80	80	80	8	8	4

Activation function	Linear	Linear	Linear	Linear	Linear	ReLU
Optimizer	adam	rms	rms	rms	adamx	**Adam**

Table 9. Mean results of 5 runs for next 'N' weeks closing price prediction for Interval 3

Horizons	**Metrics**	**CNN-BiLSTM (technical indicators)**	**LSTM (univariate)**	**BiLSTM (technical indicators)**	**DANN (fundamental indicators)**	**DANN (technical indicators)**	**Proposed Enhanced Bi-LSTM (technical indicators)**
1 week	MAE	174.85	472.84	324.85	348.29	148.35	**55.36**
	MAPE	4.12	4.25	3.28	4.85	1.25	**0.12**
	RMSE	358.48	546.14	458.18	458.37	194.35	**45.36**
2 week	MAE	145.69	485.36	341.59	348.25	128.95	**24.02**
	MAPE	3.57	3.59	3.01	4.68	2.08	**0.25**
	RMSE	298.65	542.87	345.28	428.26	124.89	**58.35**
3 week	MAE	147.54	425.86	324.18	345.28	124.85	**44.52**
	MAPE	0.54	1.85	2.59	2.54	1.25	**0.29**
	RMSE	189.52	542.36	384.25	419.48	143.85	**46.85**

- Source: Authors calculations
- The values highlighted in bold signify the lowest prediction error

Table 10. Nifty closing price prediction for weekly and monthly

Result	**Metrics**	**CNN-BiLSTM (Technical Indicators)**	**LSTM (Univariate)**	**BiLSTM (Technical Indicators)**	**DANN (Fundamental Indicators)**	**DANN (Technical Indicators)**	**Proposed Enhanced Bi-LSTM (Technical Indicators)**
Weekly	MAE	85.46	94.28	87.41	86.39	95.48	**33.28**
	MAPE	0.38	0.25	0.48	0.39	0.64	**0.08**
	RMSE	99.48	91.28	89.46	84.29	94.28	**35.29**
Monthly	MAE	73.24	88.65	74.69	74.21	85.27	**14.31**
	MAPE	0.21	0.23	0.25	0.35	0.42	**0.05**
	RMSE	98.25	87.45	85.24	78.36	86.42	**11.18**

- Source: Authors calculations
- The values highlighted in bold signify the lowest prediction error

Table 11. Asian Paints closing price prediction for weekly and monthly

Result	**Metrics**	**CNN-BiLSTM (Technical Indicators)**	**LSTM (Univariate)**	**BiLSTM (Technical Indicators)**	**DANN (Fundamental Indicators)**	**DANN (Technical Indicators)**	**Proposed Enhanced Bi-LSTM (Technical Indicators)**
Weekly	MAE	85.26	98.14	89.25	94.26	94.58	**24.51**
	MAPE	0.29	0.36	0.45	0.28	0.41	**0.08**
	RMSE	77.56	85.48	89.39	94.75	95.28	**28.64**
Monthly	MAE	76.28	81.59	82.49	88.76	89.49	**19.28**
	MAPE	0.18	0.31	0.37	0.24	0.35	**0.04**
	RMSE	71.28	79.29	81.28	84.28	83.49	**14.28**

- Source: Authors calculations
- The values highlighted in bold signify the lowest prediction error

Table 12. Tata Motora closing price prediction for weekly and monthly

Result	**Metrics**	**CNN-BiLSTM (Technical Indicators)**	**LSTM (Univariate)**	**BiLSTM (Technical Indicators)**	**DANN (Fundamental Indicators)**	**DANN (Technical Indicators)**	**Proposed Enhanced Bi-LSTM (Technical Indicators)**
Weekly	MAE	78.25	85.69	84.28	8.27	91.46	**18.36**
	MAPE	0.32	0.42	0.35	0.46	0.4	**0.08**

	RMSE	91.27	86.26	91.47	93.27	95.28	**19.36**
Monthly	MAE	65.39	74.29	79.38	74.28	78.19	**12.28**
	MAPE	0.21	0.32	0.27	0.29	0.27	**0.04**
	RMSE	68.19	79.28	84.18	86.19	84.17	**11.08**

- Source: Authors calculations
- The values highlighted in bold signify the lowest prediction error

Table 13. Infosys closing price prediction for weekly and monthly

Result	**Metrics**	**CNN-BiLSTM (Technical Indicators)**	**LSTM (Univariate)**	**BiLSTM (Technical Indicators)**	**DANN (Fundamental Indicators)**	**DANN (Technical Indicators)**	**Proposed Enhanced Bi-LSTM (Technical Indicators)**
Weekly	MAE	74.28	78.19	84.26	85.39	98.43	**19.35**
	MAPE	0.25	0.35	0.36	0.34	0.41	**0.09**
	RMSE	82.69	86.49	89.24	91.49	95.27	**16.14**
Monthly	MAE	71.59	74.28	82.39	84.26	82.76	**12.48**
	MAPE	0.19	0.22	0.26	0.28	0.35	**0.03**
	RMSE	78.26	74.39	74.19	84.28	74.19	**10.45**

Table 14. Comparison with ML-based methods.

Method	**MAPE**	**MAE**	**Time (ms)**
Random forest [9]	3.14	54.06	1.316
Gradient boosting [9]	2.54	43.59	1.483
XGBoost [9]	2.49	42.85	2.376
DenseNet Model [9]	2.12	65.19	1.051
CNN-BiLSTM [24]	71.59	0.19	2.158
LSTM [24]	74.28	0.22	2.321
BiLSTM [24]	82.39	0.26	1.058
DANN [24	84.26	0.28	1.214
Proposed	**0.03**	**10.45**	**0.984**

Table 14 , figure 2,3 and 4 shows the comparison of various machine learning and deep learning methods based on their performance metrics: Mean Absolute Percentage Error (MAPE), Mean Absolute Error (MAE), and execution time in milliseconds (Time (ms)). The Random Forest model has a MAPE of 3.14, MAE of 54.06, and runs in 1.316 ms. Gradient Boosting improves on this with a MAPE of 2.54 and MAE of 43.59 but takes a slightly longer 1.483 ms. XGBoost shows further improvement in accuracy with a MAPE of 2.49 and MAE of 42.85, but has a higher execution time of 2.376 ms. The DenseNet Model has a MAPE of 2.12 and MAE of 65.19, running the fastest at 1.051 ms. CNN-BiLSTM reports a significantly higher MAPE of 71.59 but a lower MAE of 0.19, taking 2.158 ms. The LSTM model has a MAPE of 74.28, MAE of 0.22, and runs in 2.321 ms. The BiLSTM model shows a MAPE of 82.39, MAE of 0.26, and is faster with a time of 1.058 ms. DANN has the highest MAPE of 84.26 and an MAE of 0.28, with a runtime of 1.214 ms. The proposed model significantly outperforms the others with the lowest MAPE of 0.03, MAE of 10.45, and the fastest execution time of 0.984 ms.

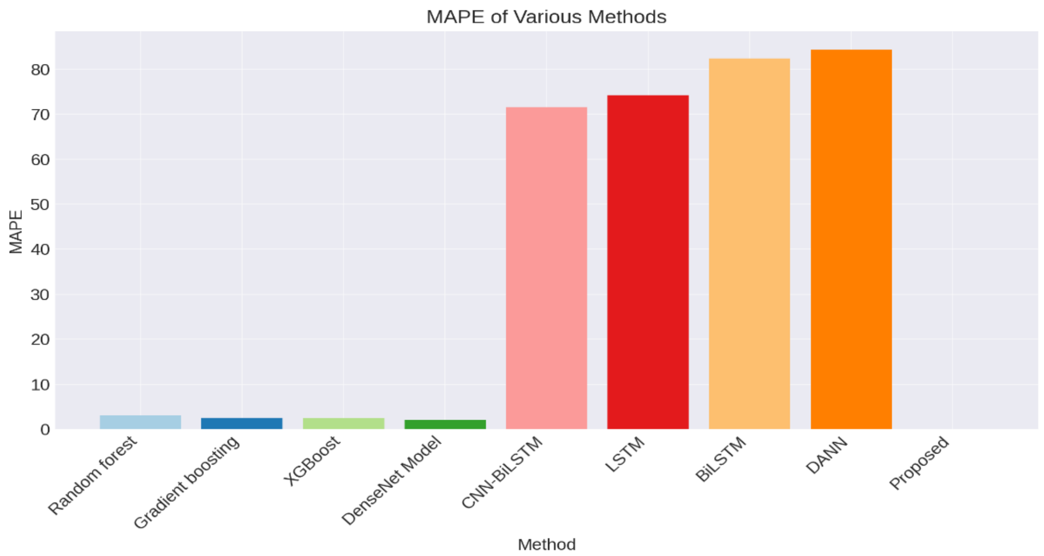

Figure 2. The MAE (Mean Absolute Error) for various methods

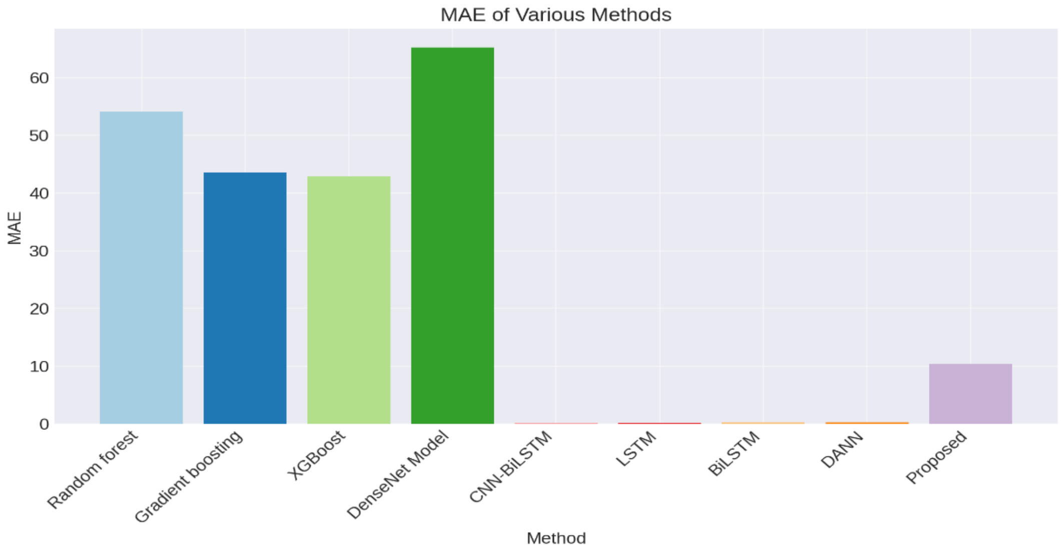

Figure 3. The MAE (Mean Absolute Error) for various methods

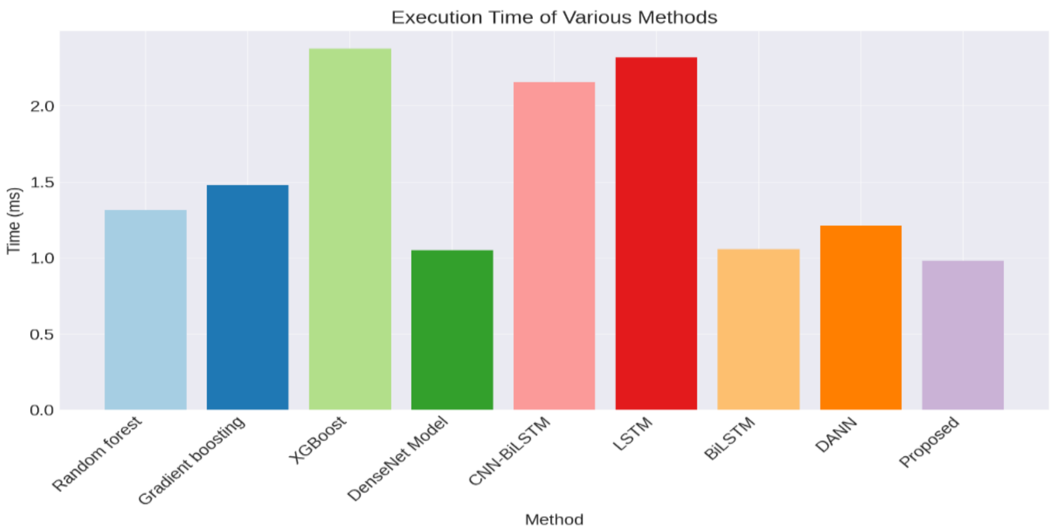

Figure 4. The execution time (in milliseconds) for various methods

6. Conclusion

comparing various prediction methods across three key performance metrics—Mean Absolute Percentage Error (MAPE), Mean Absolute Error (MAE), and execution time in milliseconds (Time (ms))—highlight distinct differences in the efficiency and accuracy of each method. Random Forest, Gradient Boosting, XGBoost, and DenseNet Model show relatively lower MAPE and MAE values, indicating higher prediction accuracy, with DenseNet Model exhibiting a particularly strong performance in MAE. However, when considering execution time, these models vary, with DenseNet Model being notably efficient. The CNN-BiLSTM, LSTM, BiLSTM, and DANN methods, while showing higher MAPE and MAE values, particularly in the case of CNN-BiLSTM and LSTM indicating lower accuracy, have varied execution times with some being relatively fast. The proposed method stands out with the lowest MAPE, suggesting exceptional accuracy, and also boasts the shortest execution time, highlighting its potential as a highly efficient and accurate prediction tool. These charts collectively underscore the importance of balancing accuracy and computational efficiency in selecting or developing predictive models.

The future scope of this work could extend in several directions. First, the incorporation of real-time data feeds and the ability to adapt to market volatility could be explored to improve the robustness and responsiveness of the model. Second, the model could be tested across different sectors and various market conditions to generalize its applicability. Third, integrating sentiment analysis from news articles and social media could enhance the model's predictive capabilities by capturing the market sentiment.

Author contributions

Bhagyashree Pathak: Conceptualization, Methodology, Software, Field study, Data curation, Writing-Original draft preparation, Software, Validation, Field study. **Dr. Snehlata Barade:** Visualization, Investigation, Writing-Reviewing and Editing.

Conflicts of interest

The authors declare no conflicts of interest.

References

1. Sonkavde, Gaurang, Deepak Sudhakar Dharrao, Anupkumar M. Bongale, Sarika T. Deokate, Deepak Doreswamy, and Subraya Krishna Bhat. "Forecasting stock market prices using machine learning and deep learning models: A systematic review, performance analysis and discussion of implications." *International Journal of Financial Studies* 11, no. 3 (2023): 94.
2. Mukherjee, Somenath, Bikash Sadhukhan, Nairita Sarkar, Debajyoti Roy, and Soumil De. "Stock market prediction using deep learning algorithms." *CAAI Transactions on Intelligence Technology* 8, no. 1 (2023): 82-94.
3. Chandola, Deeksha, Akshit Mehta, Shikha Singh, Vinay Anand Tikkiwal, and Himanshu Agrawal. "Forecasting directional movement of stock prices using deep learning." *Annals of Data Science* 10, no. 5 (2023): 1361-1378.
4. Alkhatib, Khalid, Huthaifa Khazaleh, Hamzah Ali Alkhazaleh, Anas Ratib Alsoud, and Laith Abualigah. "A new stock price forecasting method using active deep learning approach." *Journal of Open Innovation: Technology, Market, and Complexity* 8, no. 2 (2022): 96.
5. Bhandari, Hum Nath, Binod Rimal, Nawa Raj Pokhrel, Ramchandra Rimal, Keshab R. Dahal, and Rajendra KC Khatri. "Predicting stock market index using LSTM." *Machine Learning with Applications* 9 (2022): 100320.
6. Shah, Atharva, Maharshi Gor, Meet Sagar, and Manan Shah. "A stock market trading framework based on deep learning architectures." *Multimedia Tools and Applications* 81, no. 10 (2022): 14153-14171.
7. Balakrishnan, S., Deepak Kumar Pathak, Jeidy Panduro-Ramirez, Dharam Buddhi, B. Girimurugan, and Hemant Singh Pokhariya. "Empirical evaluation on stock market forecasting via extreme learning machine." In *2023 3rd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, pp. 475-479. IEEE, 2023.
8. Prachyachuwong, Kittisak, and Peerapon Vateekul. "Stock trend prediction using deep learning approach on technical indicator and industrial specific information." *Information* 12, no. 6 (2021): 250.
9. Albahli, Saleh, Tahira Nazir, Awais Mehmood, Aun Irtaza, Ali Alkhalifah, and Waleed Albattah. "AEI-DNET: a novel densenet model with an autoencoder for the stock market predictions using stock technical indicators." *Electronics* 11, no. 4 (2022): 611.
10. Azevedo, Vitor, Georg Sebastian Kaiser, and Sebastian Mueller. "Stock market anomalies and machine learning across the globe." *Journal of Asset Management* 24, no. 5 (2023): 419-441.
11. Khan, Razib Hayat, Jonayet Miah, Md Minhazur Rahman, Md Maruf Hasan, and Muntasir Mamun. "A study of forecasting stocks price by using deep Reinforcement Learning." In *2023 IEEE World AI IoT Congress (AIIoT)*, pp. 0250-0255. IEEE, 2023.
12. Mehtab, Sidra, and Jaydip Sen. "Analysis and forecasting of financial time series using CNN and LSTM-based deep learning models." In *Advances in Distributed Computing and Machine Learning: Proceedings of ICADCML 2021*, pp. 405-423. Springer Singapore, 2022.
13. Sen, Jaydip, Sidra Mehtab, and Abhishek Dutta. "Stock price prediction using machine learning and LSTM-based deep learning models." *Authorea Preprints* (2023).
14. Ghosh, Indranil, and Manas K. Sanyal. "Introspecting predictability of market fear in Indian context during COVID-19 pandemic: An integrated approach of applied predictive modelling and explainable AI." *International Journal of Information Management Data Insights* 1, no. 2 (2021): 100039.

15. Tschora, Léonard, Erwan Pierre, Marc Plantevit, and Céline Robardet. "Electricity price forecasting on the day-ahead market using machine learning." *Applied Energy* 313 (2022): 118752.
16. Mehtab, Sidra, Jaydip Sen, and Abhishek Dutta. "Stock price prediction using machine learning and LSTM-based deep learning models." In *Machine Learning and Metaheuristics Algorithms, and Applications: Second Symposium, SoMMA 2020, Chennai, India, October 14–17, 2020, Revised Selected Papers 2*, pp. 88-106. Springer Singapore, 2021.
17. Nõu, Anders, Darya Lapitskaya, M. Hakan Eratalay, and Rajesh Sharma. "Predicting stock return and volatility with machine learning and econometric models-a comparative case study of the Baltic stock market." *International Journal of Computational Economics and Econometrics* 13, no. 4 (2023): 446-489.
18. Avramov, Doron, Si Cheng, and Lior Metzker. "Machine learning vs. economic restrictions: Evidence from stock return predictability." *Management Science* 69, no. 5 (2023): 2587-2619.
19. Shen, Jingyi, and M. Omair Shafiq. "Short-term stock market price trend prediction using a comprehensive deep learning system." *Journal of big Data* 7, no. 1 (2020): 1-33.
20. Nasiri, Hamid, and Mohammad Mehdi Ebadzadeh. "Multi-step-ahead stock price prediction using recurrent fuzzy neural network and variational mode decomposition." *Applied Soft Computing* 148 (2023): 110867.
21. Muthukumaran, Kavitha, and K. Hariharanath. "Deep learning enabled financial crisis prediction model for small-medium sized industries." *Intelligent Automation & Soft Computing* 35, no. 1 (2023): 101-103.
22. Zhong, Xiao, and David Enke. "Predicting the daily return direction of the stock market using hybrid machine learning algorithms." *Financial Innovation* 5, no. 1 (2019): 1-20.
23. Garai, Sandip, Ranjit Kumar Paul, Debopam Rakshit, Md Yeasin, Walid Emam, Yusra Tashkandy, and Christophe Chesneau. "Wavelets in combination with stochastic and machine learning models to predict agricultural prices." *Mathematics* 11, no. 13 (2023): 2896.
24. Tripathi, Bhaskar, and Rakesh Kumar Sharma. "Modeling bitcoin prices using signal processing methods, bayesian optimization, and deep neural networks." Computational Economics 62, no. 4 (2023): 1919-1945.
25. Sen, Jaydip, Sidra Mehtab, and Gourab Nath. "Stock price prediction using deep learning models." Authorea Preprints (2023).
26. amalov, Firuz. "Forecasting significant stock price changes using neural networks." *Neural Computing and Applications* 32 (2020): 17655-17667.