STOCK TECHNICAL ANALYSIS WITH MACHINE LEARNING

Project report submitted to
Visvesvaraya National Institute of Technology, Nagpur
in fulfillment of requirement for the award of
degree of

Master of Technology
in
Applied Artificial Intelligence

by

Adwait Shirish Sonawane (MT23AAI031)

under the guidance of

Dr. Ankit Bhurane

Department of Applied Artificial Intelligence
Visvesvaraya National Institute of Technology, Nagpur
Nagpur 440010 (India)

December, 2025

STOCK TECHNICAL ANALYSIS WITH MACHINE LEARNING

Project report submitted to
Visvesvaraya National Institute of Technology, Nagpur
in fulfillment of requirement for the award of
degree of

Bachelor of Technology
in
Applied Artificial Intelligence

by

Adwait Shirish Sonawane (MT23AAI031)

under the guidance of

Supervisor Name

Department of Applied Artificial Intelligence
Visvesvaraya National Institute of Technology, Nagpur
Nagpur 440010 (India)

December, 2025

Copyright © Visvesvaraya National Institute of Technology, Nagpur(VNIT)
December, 2025

Department of Applied Artificial Intelligence
Visvesvaraya National Institute of Technology, Nagpur

DECLARATION

I, "Adwait Shirish Sonawane" hereby declare that the project work titled "**Stock Technical Analysis with Machine Learning**" is carried out by us in the Applied Artificial Intelligence of VNIT, Nagpur, India. The work is original and has not been submitted earlier as a whole or in part for the award of any degree/diploma at this or any other Institution/University.

Adwait Shirish Sonawane
MT23AAI031

Date:December 01, 2025

CERTIFICATE

This is to certify that the thesis titled "**Stock Technical Analysis with Machine Learning**" submitted by **Adwait Shirish Sonawane** in partial fulfilment of requirements for the award of Bachelor of Technology in the Applied Artificial Intelligence of VNIT, Nagpur. The work is comprehensive, complete and fit for evaluation.

Dr. Ankit Bhurane
Assistant Professor,
Applied Artificial Intelligence, VNIT,Nagpur

Dr. Ashwin Kothari
Professor and Head,
Applied Artificial Intelligence, VNIT,Nagpur
Date: December 01, 2025

ACKNOWLEDGEMENTS

Our sincere gratitude to our guide, for his constant support and encouragement, without which the project would not have been possible. Our heartfelt gratitude to him for offering all of the facilities and assisting us in overcoming obstacles.

We would like to thank all the professors of ECE department for helping us by some way or the other. We sincerely thank our project evaluators for their valuable comments and suggestions . We thank Head of ECE department for giving an opportunity for the project.

Date: December 01, 2025 Adwait Shirish Sonawane(MT23AAI031)

Abstract

Accurately forecasting short-term stock price movement remains a challenging problem due to the nonlinear, volatile, and dynamic nature of financial markets. Traditional forecasting approaches such as statistical models and manually applied technical analysis often struggle to adapt to changing market behavior and cannot fully capture relationships across time. With advancements in artificial intelligence, machine learning techniques now offer the potential to learn complex temporal patterns directly from historical data and improve predictive performance. This study applies machine learning and deep learning models to predict next-day closing prices for Indian stock market equities using technical analysis features.

Two years of daily historical stock data were collected through the Kite Connect API for companies listed in the NIFTY50, NIFTY200, and broader NSE universe. Each dataset includes OHLCV (Open, High, Low, Close, Volume) values and selected technical indicators such as RSI, MACD, Bollinger Bands, and Moving Averages. A sliding 20-day window is used as model input to predict the following day's closing value. The models implemented include LSTM, GRU, a hybrid CNN-LSTM, and a Transformer-based architecture. These models were trained and evaluated using a time-series split to preserve chronological integrity.

Performance was assessed using regression metrics including Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE). A deployment interface was also developed to retrieve live data, generate predictions, and visualize model output over recent chart history, providing a practical layer for user interaction and lightweight backtesting.

Findings from the study indicate that deep learning architectures, particularly the CNN-LSTM and Transformer models, deliver improved prediction accuracy compared to single-sequence models, especially when trained on diverse, multi-stock datasets. Stock-specific LSTM and GRU models also performed well when aligned with individual price behavior. Overall, the research demonstrates that combining technical indicators with contemporary machine learning architectures can meaningfully enhance short-term forecasting performance and provide a foundation for future automated trading applications.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

1.1 Introduction and Background

Financial markets play a critical role in the global economy by enabling capital allocation, investment, and wealth creation. The stock market, in particular, attracts researchers, practitioners, institutional investors, and retail traders due to its potential for returns and its inherently dynamic behavior. Forecasting stock price movements has long been considered a difficult and complex challenge because price fluctuations are influenced by numerous factors including market sentiment, macroeconomic indicators, geopolitical events, news releases, and investor psychology. Traditional forecasting relied heavily on statistical tools and expert-driven market intuition; however, advancements in computational modeling have transformed forecasting into a data-driven discipline.

In recent years, the integration of machine learning (ML), deep learning (DL), and hybrid computational models has become increasingly prevalent in stock market prediction systems. These techniques leverage large-scale data, extract hidden nonlinear patterns, and adapt dynamically to changing market behavior. Modern forecasting approaches are no longer limited to detecting trends but aim to perform accurate classification, regression, and automated decision-making for executing profitable trades. This shift represents a broader transformation from human-interpreted charts to sophisticated algorithmic trading systems capable of handling high-dimensional time-series data with speed and consistency.

1.2 Evolution of Technical Analysis

Technical analysis is one of the oldest methods for forecasting financial markets. Its foundation lies in the assumption that price movement reflects the collective psychology of market participants and that historical patterns tend to repeat. Tools such as candlestick formations, support-resistance levels, and indicator-based signals have been widely used for decades to evaluate potential trend reversals, continuation phases, and volatility zones. Such techniques remain popular because they do not require economic or fundamental data and instead rely solely on market-generated information.

Over time, technical analysis evolved into a structured, quantitative practice with the development of mathematical indicators such as Moving Averages, Bollinger Bands, Relative Strength Index (RSI), Moving Average Convergence Divergence (MACD), Williams, Aroon Oscillator, and Stochastic Oscillator. These indicators perform feature transformations that help traders and later machine learning models filter market noise and derive more structured predictive signals. Historical research confirms that technical indicators provide meaningful information that can aid in forecasting price direction when processed effectively.

However, while technical analysis improved the interpretability of market behavior, traditional manual usage of indicators remains subjective, prone to human bias, and limited in scalability problems that modern AI-driven systems aim to overcome.

1.3 Limitations of Classical Forecasting Models

Before machine learning methods emerged, stock forecasting relied heavily on statistical time-series models such as Moving Averages, ARIMA (Auto-Regressive Integrated Moving Average), and Holt-Winters exponential smoothing. Though effective for identifying baseline patterns, these models assume stationarity and linearity properties rarely present in real-world market behavior.

Financial time series are inherently noisy, nonlinear, chaotic, and non-stationary. Market conditions shift rapidly due to unpredictable external triggers. Classical models struggle under these conditions because they are not designed to learn evolving patterns or detect nonlinear input relationships. Research widely agrees that these approaches become insufficient as feature dimensionality increases or when multi-factor market behavior must be modeled.

Furthermore, such models often treat time series as isolated data rather than contextual sequences influenced by momentum, volatility, and behavioral patterns.

These limitations motivated the exploration of machine learning and deep learning techniques capable of capturing nonlinear and temporal dependencies in market data.

1.4 Machine Learning as a Solution

Machine learning introduced a paradigm shift in stock forecasting by enabling systems to automatically learn relationships from data rather than relying on predefined rigid formulas. ML models excel at identifying complex nonlinear interactions between features such as price history, indicators, and temporal changes making them well-suited for financial forecasting.

Traditional ML algorithms such as Support Vector Machines (SVM), Decision Trees, Random Forests, Naïve Bayes, and Logistic Regression demonstrated early success in prediction tasks. For instance, SVM models combining technical indicators such as Moving Averages, Bollinger Bands, RSI, Stochastic Oscillator, and Aroon achieved up to 77.8% prediction accuracy, demonstrating robustness especially for volatile stocks.

Similarly, research applying Decision Trees to Indian National Stock Exchange (NSE) companies reported an accuracy of 80.08%, outperforming more complex ensemble methods like Random Forests (78.8%).

However, traditional ML still required feature engineering, lacked sequential memory, and did not inherently model long-term temporal dependencies which led to the emergence of deep learning-based forecasting.

1.5 Importance of Technical Indicators

As machine learning became integrated with stock forecasting, technical indicators transitioned from trader tools to standardized computational features. Indicators such as MACD, RSI, Williams, and EMA provide derived patterns including momentum surges, trend strength, and volatility compression that offer machine learning algorithms richer, more interpretable representations of market conditions.

Multiple studies confirm that combining multiple indicators provides stronger predictive power than relying on a single signal, because each indicator captures a different aspect of price dynamics.

Indicators act as domain-optimized feature engineering, reducing noise and preventing models from learning unstable raw fluctuations. This makes them particularly effective when coupled with supervised learning algorithms and deep neural networks trained on time-series data.

1.6 Need for Hybrid Intelligent Systems

The most significant advancement in recent financial forecasting research is the emergence of hybrid models, which combine strengths of multiple architectures rather than relying on a single approach. For example, hybrid CNN-LSTM models extract short-term price patterns using CNN layers and long-term sequential behavior using LSTM, outperforming standalone models in multiple studies.

More advanced frameworks incorporate:

- Bidirectional recurrent networks
- Attention mechanisms
- Ensemble stacking
- Sentiment-enhanced features
- Reinforcement learning components

One referenced Stacked CNN-LSTM Ensemble (SCLE) integrating over 87 technical indicators and sentiment inputs demonstrated a 30% improvement in accuracy compared to a sentiment-only model.

This trend confirms that hybridization improves robustness, adaptability, and predictive accuracy especially in volatile environments.

1.7 Identified Research Gap

While existing research demonstrates significant advancements in stock prediction using ML, deep learning, and hybrid models, several gaps remain. First, a large portion of the work focuses on markets such as the NASDAQ, NYSE, and Shanghai Stock Exchange, with far less emphasis on emerging markets such as India. This creates generalization concerns, as financial behavior differs across regions due to regulation, liquidity conditions, economic stability, and trading culture.

Second, many studies prioritize prediction accuracy as the primary evaluation measure. However, high classification accuracy does not always translate to profitable or low-risk trading outcomes, especially when ignoring transaction fees, slippage, market volatility, or drawdown effects. Recent papers highlight that success metrics should incorporate profitability measures such as return on investment (ROI), maximum drawdown, and the Sharpe ratio.

Third, deep learning approaches, although powerful, are often criticized for their black-box nature, making them difficult to interpret and validate.

This lack of transparency poses challenges in regulated financial environments where decision justification is required.

Finally, while hybrid models show promise, there is limited research on systematically comparing classical ML, deep learning, and hybrid approaches specifically combined with technical analysis inputs under consistent evaluation frameworks especially using Indian market data.

These gaps collectively indicate a need for structured research that evaluates machine learning-driven technical analysis within the context of Indian market behavior while incorporating explainability and meaningful financial evaluation metrics.

1.8 Problem Statement

Based on the identified research gap, the key challenges addressed in this thesis are, traditional stock forecasting methods and standalone technical analysis approaches are limited in their ability to model nonlinear, volatile, and complex stock price behavior, particularly in emerging markets such as India. While machine learning and deep learning approaches have shown promising results, there is insufficient structured evaluation of their effectiveness when technical indicators are used as input features, especially in comparison across multiple model families and under realistic trading evaluation criteria.

This study seeks to address the problem by evaluating machine learning models for stock forecasting using technical indicators as engineered inputs, assessing their predictive capability, interpretability, and practical applicability.

1.9 Research Objectives

The main objective of this thesis is to evaluate the performance of machine learning models applied to stock technical analysis and determine their suitability for practical forecasting and decision-making.

This objective expands into the following sub-objectives:

1. To collect, preprocess, and structure historical stock data suitable for machine learning and deep learning models.

2. To compute and integrate a diverse set of technical indicators as engineered feature representations.

3. To train and evaluate multiple machine learning algorithms, including classical ML models and neural architectures.
4. To compare model performance using both statistical accuracy metrics and financially meaningful performance metrics.
5. To interpret and analyze model behavior to assess generalization capability and practical feasibility in real-world markets.

1.10 Research Questions

Aligned with the objectives, the study aims to answer the following questions:

1. Do machine learning models improve prediction performance when combined with technical analysis indicators compared to raw prices?
2. Which category of model classical machine learning, deep learning, or hybrid performs best under consistent evaluation?
3. Can models trained on historical pricing patterns generalize to unseen market data, particularly in volatile environments?
4. Do statistical accuracy metrics correlate reliably with trading performance metrics such as Sharpe ratio or return on investment (ROI)?
5. Are models based on technical analysis suitable for deployment in emerging markets such as India, or do market conditions reduce their effectiveness?

Answering these questions provides measurable insight into the viability and limitations of applying machine learning to technical stock forecasting.

1.11 Scope, Limitations, and Assumptions

The scope of this research is restricted to:

1. Historical stock price data
2. Daily time-series resolution
3. Models trained on select high-volume stocks
4. Forecast horizon limited to short-term directional or price movement prediction

Key limitations include:

1. Results may vary across different stocks, time horizons, and economic conditions.
2. The study excludes macroeconomic indicators, textual sentiment, or high-frequency tick data.
3. Interpretability challenges may persist for deeper neural architectures.
4. Model performance may degrade during extreme market volatility or structural regime shifts.
5. Assumptions include the availability of consistent market data, stable economic conditions during evaluation, and reproducibility of indicator formulas across software implementations.

1.12 Novelty and Contributions

1. The contributions of this research are summarized as follows:
2. A structured comparative study of classical ML, deep learning, and hybrid models using a uniform technical analysis–based dataset.
3. Inclusion of both accuracy metrics and financial performance metrics, aligning evaluation with real-world trading relevance.
4. Application of machine learning-driven technical analysis specifically to Indian market conditions, addressing a documented research gap.
5. Development of a replicable preprocessing and modeling pipeline for future applied research and industry use.

These contributions offer practical value for quantitative researchers, retail investors, academic researchers, and financial institutions.

Chapter 2

Literature Review

2.1 Introduction

Stock market forecasting has evolved from traditional rule-based technical analysis into a data-driven scientific process powered by machine learning and deep learning. While technical indicators have historically guided trading decisions, recent advancements demonstrate the superior capability of computational models to identify hidden relationships, reduce noise, and generate more consistent predictions. Research in this domain reveals a shift from classical time-series forecasting models to hybrid intelligent systems that combine technical indicators with machine learning architectures.

Traditional technical analysis relies on price and volume behavioral patterns to predict market direction, yet the non-linear and non-stationary characteristics of financial markets limit its reliability when used alone. Machine learning has emerged as a solution capable of uncovering complex, hidden patterns beyond human interpretive capacity. The uploaded literature demonstrates a clear trend toward automated forecasting systems that combine technical indicators with advanced ML architectures to produce predictive, interpretable, and profitable trading models

2.2 Classical Forecasting Techniques

Early quantitative forecasting relied heavily on linear statistical models such as Moving Averages, ARIMA, and Holt-Winters exponential smoothing. Moving Averages provided baseline signals for trend reversals and smoothing market fluctuations, but lacked adaptability to rapid market regime changes. Likewise, ARIMA and Holt-Winters introduced seasonality and trend modeling but struggled with the non-stationary nature of real-world financial time series. These techniques assumed stable statistical properties and linear relationships, making them insufficient for modeling complex market behaviors driven by volatility, sentiment, and external events.

2.3 Technical Indicators and Their Role

Technical indicators emerged as an important feature engineering method for forecasting. Studies confirm the usefulness of indicators such as SMA, EMA, MACD, Bollinger Bands, RSI, Williams %R, and Stochastic Oscillator in identifying momentum, volatility, and price patterns. These indicators transform raw price and volume into meaningful signals, filtering noise and exposing underlying patterns. Research consistently finds that using multiple indicators enhances prediction reliability compared to relying on a single indicator. Indicators act as domain-expert feature extraction rather than raw input, making them widely adopted across modern ML-based forecasting systems.

Trend indicators (SMA, EMA, MACD), momentum indicators (RSI, Williams %R, Stochastic Oscillator), and volatility indicators (ATR, Bollinger Bands) form the dominant feature set in current research. ML-driven systems increasingly incorporate multiple categories of indicators to improve generalization and predictive robustness in volatile markets.

2.4 Traditional Machine Learning Models in Stock Forecasting

The first significant improvements beyond statistical forecasting came from traditional machine learning models such as Support Vector Machines (SVM), Decision Trees, and Random Forests. These models addressed non-linearity and pattern complexity more effectively. For example, SVM-based trading systems using combinations of Bollinger Bands, MA, RSI, and Stochastic Oscillator achieved prediction accuracy as high as 77.8% and performed particularly well in volatile stocks.

Support Vector Machine models remain widely used due to their ability to classify non-linear patterns effectively using kernel transformations. One referenced study applying SVM with Bollinger Bands, RSI, MA, Stochastic, and Aroon Oscillator across Indonesian market equities achieved accuracy values up to 77.8%, demonstrating strong performance especially on highly fluctuating stocks.

Decision Tree-based models have also shown competitive results. A study applying Decision Trees to National Stock Exchange (NSE) equities with multiple indicators reported an accuracy of 80.08%, outperforming both Random Forest (78.8%) and Naïve Bayes (73.8%), showing that simple models can outperform more complex ones when interpretability and rule-based behavior align with market conditions.

The persistence of these models in research and industry reflects a trade-off between interpretability and predictive power a critical factor in regulated environments such as finance.

Decision Tree-based methods also showed strong results, with onestudy reporting 80.08% accuracy surprisingly outperforming RandomForests and Naive Bayes.

These approaches remain relevant due to their interpretability, an important factor in regulated financial systems. While not always achieving the highest accuracy, their transparency ensures practical decision-making accountability.

2.5 Deep Learning Approaches

Deep learning models have become dominant due to their ability to capture temporal dependencies, non-linear relationships, and high-dimensional feature interactions. Among recurrent neural networks, LSTM architectures are the most widely adopted because they retain long-term memory and mitigate vanishing gradient issues. A study applying a four-layer LSTM on OHLC data combined with MACD, KD, RSI, and Williams %R demonstrated 83.6% prediction accuracy, with MACD performing best as an individual indicator at 76%.

CNN-based financial forecasting is also gaining traction, especially for pattern extraction and high-frequency trading. Referenced study used a one-dimensional CNN trained on 130 features for intraday classification and demonstrated competitive efficiency and predictive power.

Convolutional Neural Networks (CNNs), although originally designed for spatial pattern recognition, have been successfully adapted using 1-D filters to detect short-term sequence features. Research from Stanford applied CNNs to high-frequency intraday trading using 130 input features and demonstrated strong computational efficiency and predictive capability

These findings highlight that deep learning outperforms traditional ML methods when sequence behavior and long-term dependencies are critical.

2.6 Hybrid Models Combining Technical Indicators and ML

A notable trend in the literature is the shift toward hybrid models that combine multiple ML or DL architectures. Hybrid CNN-LSTM models are the most prevalent approach, as they capture both short-term localized price movements (via CNN) and long-term sequential behavior (via LSTM). These hybrid models consistently outperform stand-alone LSTM or CNN architectures.

Research pushes hybrid integration further, incorporating additional layers such as bidirectional LSTM, attention mechanisms, and ensemble stacking. One cited Stacked CNN-LSTM Ensemble (SCLE) integrated 87 technical indicators with sentiment-based input and achieved 30% improved prediction accuracy over sentiment-only systems, showing strong evidence that multimodal approaches yield superior results.

Reinforcement learning adds another dimension by focusing not onprediction accuracy but on portfolio profitability and Sharpe ratiooptimization.

2.7 Reinforcement Learning Approaches

Reinforcement Learning (RL) presents a fundamentally different approach by learning optimal trading strategies (buy/sell/hold) instead of predicting future prices. RL models are explicitly rewarded for profit-maximization under volatility and transaction constraints.

A referenced study using Advantage Actor-Critic (A2C) demonstrated improved risk-adjusted performance with a Sharpe ratio of 0.13, outperforming the DJI index baseline with a negative Sharpe ratio.

Such results suggest that RL may resolve the disconnect between high statistical forecasting accuracy and actual trading profitability one of the most critical challenges in ML-based stock prediction.

2.8 Comparison Between different Model and research

Paper/Study	**Technical Indicators Used**	**Main Findings / ML Method Applied**
A Hybrid Stock Trading Framework	Technical indicators as referenced in the paper	Uses CEFLANN model. Proposed a novel framework improving trading decision efficiency.
Combining News and Technical Indicators	Seven indicators derived from past five days' prices	Applied SVM. Achieved higher accuracy and return than single-source input. Hit ratio: 61.7%.
A Hierarchical Decision Tree Model	SMA, EMA, WMA, MACD, ADX, Aroon, Stochastic, RSI, SMI, WPR, CCI, BBands, ATR, OBV, MFI, CMF	Decision Tree, Random Forest, and Naïve Bayes tested. Decision Tree achieved highest accuracy: 80.08%.
Exploring Deep Neural Networks	KD, RSI, BIAS, Williams %R, MACD + OHLC features	Used a 4-layer LSTM. Achieved 83.6% accuracy for 3-class prediction; MACD best standalone at 76%.
A Hybrid CNN-LSTM Model	OHLCV + adjusted prices + technical indicators	Hybrid CNN-LSTM outperformed standalone CNN and LSTM, capturing short-term fluctuations and long-term patterns.
Stacked CNN-LSTM Ensemble	87+ technical indicators + sentiment data	Stacked CNN-LSTM with Bidirectional LSTM + Attention. Improved accuracy by 30% over news-only models.
SVM for Indonesian Stocks	MA, Bollinger Bands, RSI, Stochastic, Aroon Oscillator	SVM achieved up to 77.8% accuracy, performing best on volatile stocks.
A Multi-faceted Reinforcement Learning Approach	Historical price data + technical indicators	Applied A2C and DDPG. A2C delivered better risk-adjusted returns (Sharpe 0.13) than baseline (-0.10).
CNN-LSTM Model for Prediction	EMA and ROC	CNN-LSTM, LSTM, RF, KNN, and Linear Regression tested. Best accuracy: 56.5%. Sharpe Ratio: 1.73 (INTC).

Table 2.1: Summary of Machine Learning and Deep Learning Approaches in Stock Market Prediction

2.9 Key Findings, Strengths, and Limitations

Across surveyed research, key themes emerge:

Dimension Findings - Strengths ML/DL models outperform statistical models, hybrid approaches produce superior results

Insights Technical indicators significantly enhance ML performance, multimodal data boosts accuracy

Limitations - Models often lack interpretability; accuracy does not always equate to profitability

Additionally, many studies report performance solely through predictive accuracy, which does not account for slippage, transaction costs, or real-market risk exposure. Researchers now argue for financial evaluation metrics such as Sharpe ratio, max drawdown, and ROI.

Study / Approach	**Technical Indicators Used**	**Algorithm / Model**	**Key Results / Findings**
LSTM-Based Forecasting	RSI, MACD, KD, BIAS, Williams %R + OHLC	4-layer LSTM	Achieved **83.6%** accuracy; MACD alone gave **76%** accuracy.
SVM-Based Trading	MA, Bollinger Bands, RSI, Stochastic Oscillator, Aroon	Support Vector Machine (SVM)	Highest accuracy: **77.8%**, best on volatile stocks.
Decision Tree Feature Model	Trend, momentum, volume, volatility indicators	Decision Tree, Random Forest, Naïve Bayes	Decision Tree achieved **80.08%**, outperforming RF (78.8%).
CNN–LSTM Hybrid	OHLCV + technical indicators	CNN–LSTM hybrid model	Outperformed standalone CNN and LSTM models.
SCLE Hybrid Model	87 indicators + sentiment data	Stacked CNN–LSTM + Attention + Bi-LSTM	Accuracy improved **30%** over sentiment-only system.
RL for Trading	Price + technical indicators	A2C and DDPG (Reinforcement Learning)	Delivered **positive Sharpe ratio (0.13)** vs negative baseline.
CNN Pattern Model for HFT	Price-derived feature set (130 features)	1D CNN	Demonstrated high efficiency and effective classification performance.

Table 2.2: Comparison of machine learning and hybrid deep learning approaches for stock market prediction.

2.10 Evaluation Challenges, Research Gaps and Inconsistencies

Across reviewed studies, accuracy is the most reported metric; however, accuracy alone does not reliably reflect trading profitability. Multiple studies highlight the need for more holistic metrics such as annualized return, max drawdown, and Sharpe ratio to reflect real-world trading quality.

Interpretability remains another challenge, especially in regulated sectors where explainable decision-making is critical. The literature emphasizes that opaque deep learning models face adoption barriers when their behavior cannot be justified.

Finally, overfitting remains persistent due to the chaotic and non-stationary properties of financial data, requiring adaptive retraining, rolling validation, and multi-market testing strategies.

Despite progress, several gaps remain:

- Overfitting persists due to non-stationary data.
- Limited generalizability across markets, time periods, and asset classes.
- Few models evaluate real trading performance rather than classification accuracy.
- Indian markets remain underexplored relative to U.S. and Chinese datasets.
- RL-based approaches are promising but still sparse and experimental.

2.11 Alignment with the Present Research

The literature demonstrates a clear progression from classical technical analysis toward machine learning-driven forecasting systems. Traditional ML models offer interpretability, while deep learning models especially hybrid CNN-LSTM architectures provide superior predictive capability by capturing multi-dimensional patterns in historical price signals. Reinforcement learning represents the next frontier, enabling models not just to predict markets but to act intelligently within them.

The works reviewed confirm that machine learning methods especially hybrid and deep learning-based approaches yield superior results when combined with carefully engineered technical indicators. However, the gap in generalizable, explainable, and India-focused ML-based forecasting highlights the need for a model specifically designed to work with Indian market behavior.

This thesis positions itself at this intersection: applying modern ML techniques integrated with technical indicators, evaluating not only accuracy but also potential trading practicality on Indian market securities.

Chapter 3

Methodology

3.1 Methodology - Bullets

Describe the dataset: stock names, timeframe, granularity, and data sources.

Explain preprocessing steps: missing values, normalization, outlier handling.

Define your sliding window approach (e.g., using last 20 days to predict next day).

List and explain all technical indicators used along with their formulas.

Describe feature engineering (combining raw OHLC + indicators).

Present the machine learning models selected (e.g., Linear Regression, XGBoost, LSTM).

Explain the architecture of deep models (LSTM layers, neurons, dropout).

Mention the hyperparameters considered for tuning.

Outline the algorithmic workflow from raw data to prediction.

Include system architecture or flowchart explaining your pipeline.

3.2 Overview

This chapter describes the methodological framework used to build, train, and evaluate machine learning models for predicting next-day closing prices of Indian stocks. The approach includes data acquisition, preprocessing, feature engineering using technical indicators, model architecture design, training procedures, and the final prediction and deployment pipeline. The modelling strategy includes both single-stock models and generalized multi-stock architectures to evaluate how model scope influences predictive performance.

3.3 System Workflow

The overall methodology follows the structured workflow below:

Data Acquisition → Preprocessing → Feature Engineering → Windowing → Model Training → Model Evaluation → Deployment and Prediction

A detailed workflow diagram will be inserted later.

3.4 Data Acquisition

Historical stock data is collected using the Kite Connect API, which provides daily OHLCV (Open, High, Low, Close, Volume) records for Indian equity markets. Three data scopes are utilized:

Dataset Type	**Usage**	**Scope**
Company-specific dataset	LSTM & GRU individual models	Single company, 2-year window
NIFTY-based dataset	CNN–LSTM and baseline analysis	NIFTY 50 and NIFTY 200 constituents
Full-market dataset	Transformer-based architecture and cross-learning	All available NSE-listed companies

Table 3.1: Dataset categorization based on usage and scope.

All datasets use two years of historical records, sampled at one trading day frequency. Missing trading days (holidays, suspensions) are handled by forward-fill as part of preprocessing.

3.5 Data Preprocessing

Preprocessing ensures consistency, numeric uniformity, and suitability for machine learning. Key steps include:

3.5.1 Handling Missing Values

Non-trading days → forward fill

Missing Volume → replaced with rolling mean over previous 7 days

Any incomplete feature rows after indicator computation are removed

3.5.2 Normalization

Since machine learning models operate more effectively on normalized sequences, the following scaling methods are used:

Min-Max scaling for time-dependent features:

$$x_{\text{scaled}} = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{3.1}$$

Target variable (next closing price) scaled using the same transformer to maintain data consistency.

Each company's scaling parameters are stored individually to allow inverse transformation during inference.

3.6 Feature Engineering

To capture price behavior, the model uses both raw market features (OHLCV) and derived technical indicators.

3.6.1 Raw Features

- Open
- High
- Low
- Close
- Volume

3.6.2 Technical Indicators

A selected set of commonly used technical indicators are computed to enhance model-level signal extraction. Indicators include:

Category Indicators Used

All indicators are computed using rolling windows of 14-26 days depending on indicator specification.

This structure allows the model to learn not only raw patterns but momentum changes, volatility behavior, and trend pressure.

Table 3.2: Technical Indicator and Categories

Category	**Indicators Used**
Trend	SMA, EMA
Momentum	RSI, Stochastic %K, Williams %R
Volatility	Bollinger Bands (Upper, Lower, %B)
Composite	MACD, Signal Line, Histogram

3.7 Windowing Strategy

Since predictions are based on sequential input, a sliding window approach is applied:

Input window: 20 most recent trading days

Output horizon: Next-day closing price (single-step forecasting)

This creates training pairs of the form:

$$X = [\text{features for days } t - 19 \text{ to } t]$$
$$y = \text{closing price at day } t + 1$$

This ensures the model learns short-term memory patterns relevant to near-term forecasting.

3.8 Model Architectures

Multiple deep learning architectures are used to compare predictive capability.

3.8.1 LSTM Model

The LSTM model is designed for company-specific forecasting.

Architecture outline:

- **Loss:** Mean Squared Error (MSE)
- **Optimizer:** Adam
- **Batch size:** 32–64 (depending on stock volatility)
- **Epochs:** 50–200 (early stopping applied)

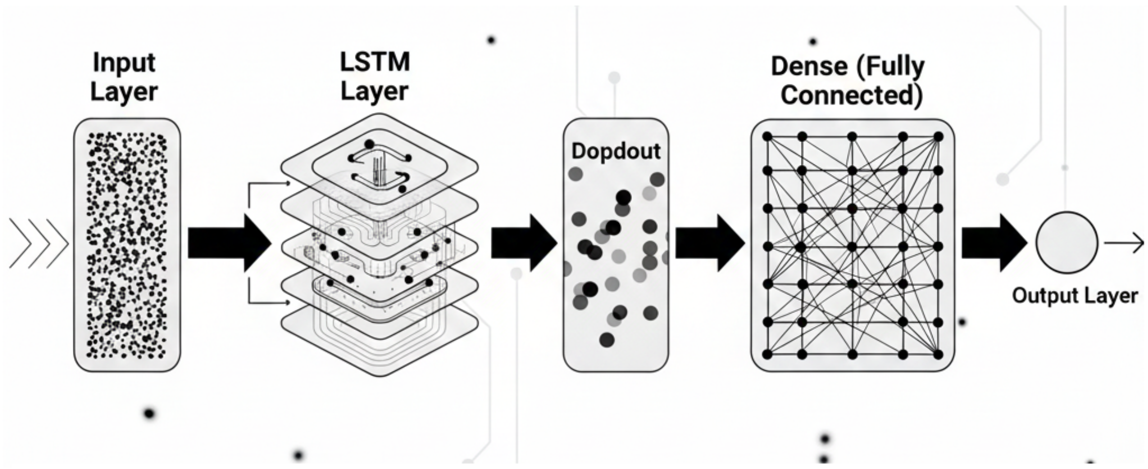

Figure 3.1: Architecture of the LSTM block.

3.8.2 GRU Model

The GRU baseline shares structure with LSTM but uses GRU cells for computational efficiency.

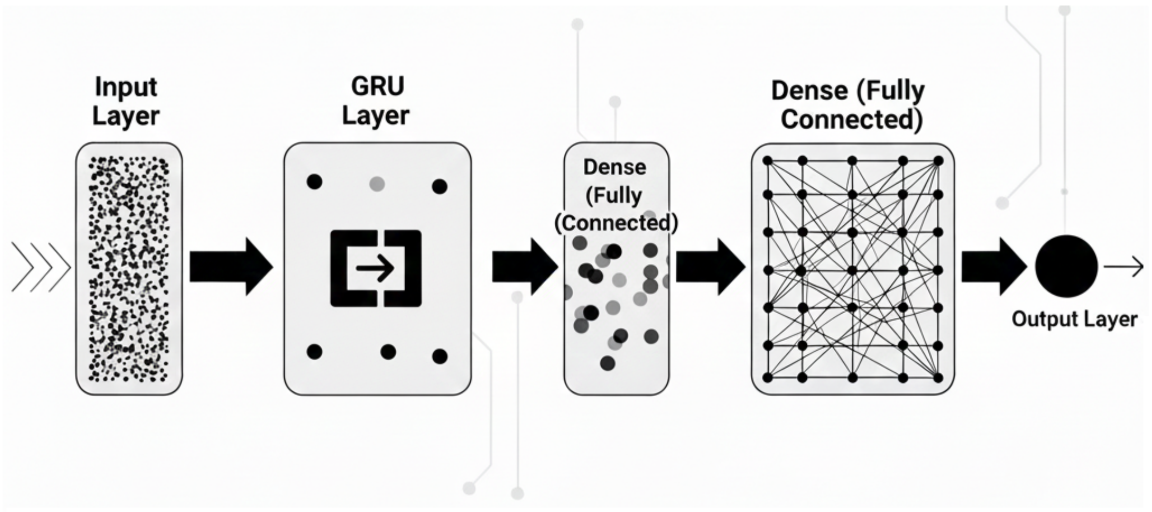

Figure 3.2: Architecture of the GRU block.

GRUs help evaluate whether simplified gating can achieve comparable performance while reducing training time.

3.8.3 CNN-LSTM Hybrid

Used for NIFTY-wide datasets.

Architecture:

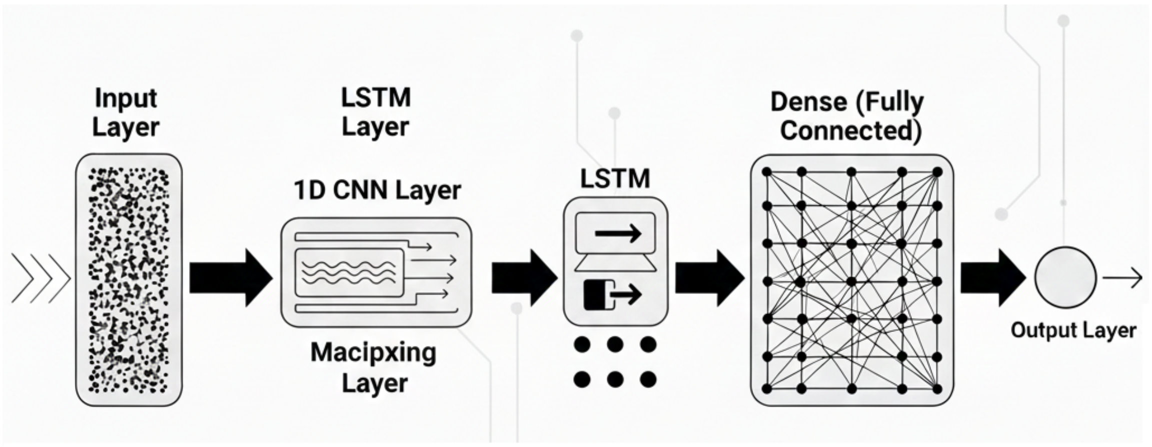

Figure 3.3: Architecture of the CNN + LSTM block.

The CNN component extracts local trend structure, while the LSTM learns sequential dependencies, making this architecture suitable for learning market-wide structural behavior.

3.8.4 Transformer Model

A generalized model trained on the entire market dataset.

Block structure:

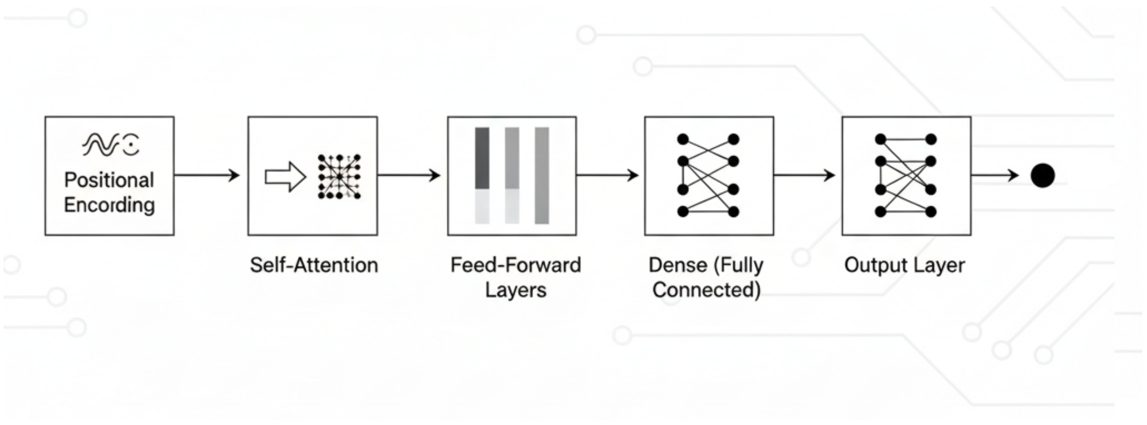

Figure 3.4: Architecture of the Transformer block.

Transformers help evaluate whether attention-based learning is more effective than recurrent models for financial time series.

3.9 Model Training and Validation

Training is carried out with:

Train/Test split: 80% / 20%

Validation method: time-series split (no shuffling)

Callbacks: early stopping and learning rate reduction

Loss curves and prediction visualizations are generated for model comparison.

3.10 Prediction and Deployment Pipeline

Once trained, models are exported and integrated into a lightweight prediction service.

Deployment flow:

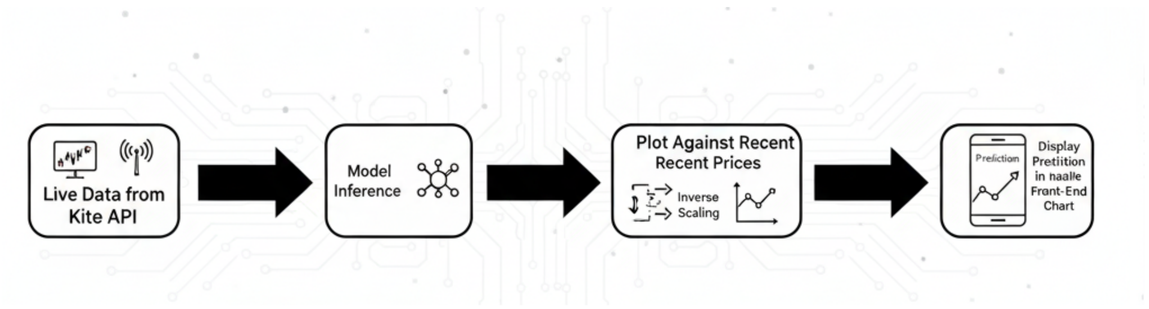
Live Data from Kite API
Model Inference
Plot Against Recent Recent Prices
Inverse Scaling
Prediction
Display Pretiition in naalle Front-End Chart

Figure 3.5: Architecture of Data Flow.

Predictions are plotted alongside the last 3 months of market data for visual validation.

Chapter 4

Experimental Setup

4.1 Experimental Setup - Bullets

Describe hardware used for training (CPU/GPU, RAM, environment).

Provide software stack details: Python version, libraries (Pandas, Scikit-Learn, TensorFlow).

Explain how training/validation/testing datasets were split.

Detail training configurations (batch size, learning rate, epochs).

Describe evaluation metrics such as RMSE, MAE, MAPE, and directional accuracy.

Mention how hyperparameters were chosen (grid search, manual tuning).

Explain any regularization or early stopping methods used.

Describe repetition for reliability (multiple runs, averaging metrics).

4.2 Overview

This chapter presents the experimental setup used to train, evaluate, and compare the machine learning models designed for next-day stock closing price prediction. It includes details about the computing environment, software dependencies, dataset configuration, training parameters, performance metrics, and model validation strategy. The goal of the experimental setup is to ensure consistency across all experiments and provide a reproducible framework for future extensions.

4.3 Hardware Configuration

All experiments were conducted on a system equipped with the following hardware:

Table 4.1: Recommended Hardware Specifications

Component	**Specification**
Processor	AMD Ryzen 5600G or equivalent multi-core CPU
RAM	16 GB DDR4
Storage	SSD (for faster dataset access and training runs)
GPU (optional)	AMD RX580 (Limited training Support)

While basic models such as LSTM and GRU can run efficiently on CPU, GPU acceleration significantly reduced training time for larger models such as the Transformer and CNN-LSTM hybrid.

4.4 Software Stack

The implementation was performed using the following software tools:

Table 4.2: Project Technology Stack

Category	**Libraries / Tools**
Programming Language	Python
Data Processing	Pandas, NumPy
Visualization	Matplotlib, Seaborn, Plotly
Machine Learning	scikit-learn
Deep Learning	TensorFlow / Keras
API Data Access	Kite Connect API
Deployment	Custom front-end interface for visualization

4.5 Dataset Preparation

The dataset consists of two years of daily historical stock data for companies included in the NIFTY50, NIFTY200, and broader NSE universe. Data preprocessing steps included:

- Forward-filling missing values for non-trading days
- Normalizing features using Min-Max scaling
- Computing technical indicators
- Structuring the final dataset into supervised learning format

A sliding window of the previous 20 trading days was used as input to predict the next day closing value.

4.6 Train-Test Split Strategy

To preserve chronological order and prevent data leakage, random shuffling was avoided.

Table 4.3: Dataset Splitting Strategy

Dataset Type	**Split**
Training Set	80% of historical sequence
Testing Set	Remaining 20%

Validation was managed via a time-series validation approach rather than k-fold cross validation.

4.7 Hyperparameter Configuration

Multiple models were trained using consistent baseline hyperparameters with fine-tuning performed through iterative experimentation. The following table summarizes the core parameters:

Table 4.4: Model Hyperparameters

Parameter	**Value / Range**
Epochs	50–200 (early stopping applied)
Batch Size	32–64
Optimizer	Adam
Learning Rate	0.001 (adaptive reduction on plateau)
Loss Function	Mean Squared Error (MSE)

Activation functions consisted of ReLU for hidden dense layers and linear activation for the output layer.

4.8 Model-Specific Training Setup

Each model required additional configuration:

4.8.1 LSTM And GRU Models

- Sequential input with shape (20 timesteps * features)
- Single hidden recurrent layer with dropout to reduce overfitting

4.8.2 CNN-LSTM Model

- Initial convolution layer for local pattern extraction
- LSTM layer for temporal sequence learning

4.8.3 Transformer Model

- Multi-head self-attention layers
- Positional encoding to preserve temporal structure

Transformer models required the longest training time due to complexity.

4.9 Evaluation Metrics

The following regression metrics were used to evaluate model performance:

Table 4.5: Model Evaluation Metrics

Metric	**Description**
MAE	Measures average magnitude of error (Mean Absolute Error)
RMSE	Penalizes larger errors more heavily (Root Mean Squared Error)
MAPE	Expresses prediction error as a percentage (Mean Absolute Percentage Error)
R^2 Score	Measures goodness of fit (Coefficient of Determination)

Additionally, prediction plots were generated to visually compare predicted vs. actual closing prices.

4.10 Deployment Testing

After training, the best-performing models were integrated into a lightweight deployment pipeline. Real-time prediction testing involved:

- Fetching latest daily closing values through Kite API
- Preprocessing the data using stored scaling and feature generation
- rules
- Running inference using trained models
- Overlaying predictions on front-end interactive charts

A rolling evaluation over the last three months of data was also performed to simulate real-world model behavior.

Chapter 5

Results and Analysis

5.1 Results And Analysis - Bullets

5.2 Overview

This chapter presents the results obtained from evaluating multiple machine learning architectures for forecasting next-day closing prices of Indian stock market equities. The models were assessed on both single-company datasets and broader market-level datasets to understand how input size, model complexity, and feature engineering influence predictive accuracy. Evaluation includes quantitative error metrics, visual prediction comparison, convergence behavior, and model generalization characteristics.

5.3 Training and Validation Behavior

Across experiments, all models showed decreasing loss curves during training, confirming the ability to learn meaningful temporal dependencies. However, their learning dynamics differed based on dataset scope and model capacity.

TODO - Insert Graphs for Training and validation

Key findings from convergence analysis include:

- For single-company datasets, **GRU and LSTM models** converged fastest and most consistently, likely due to their lower parameter count relative to Transformers and CNN-LSTM networks.
- When only the **closing price** was used as the input feature, models fit extremely well and often achieved very low training loss, indicating possible **overfitting**.

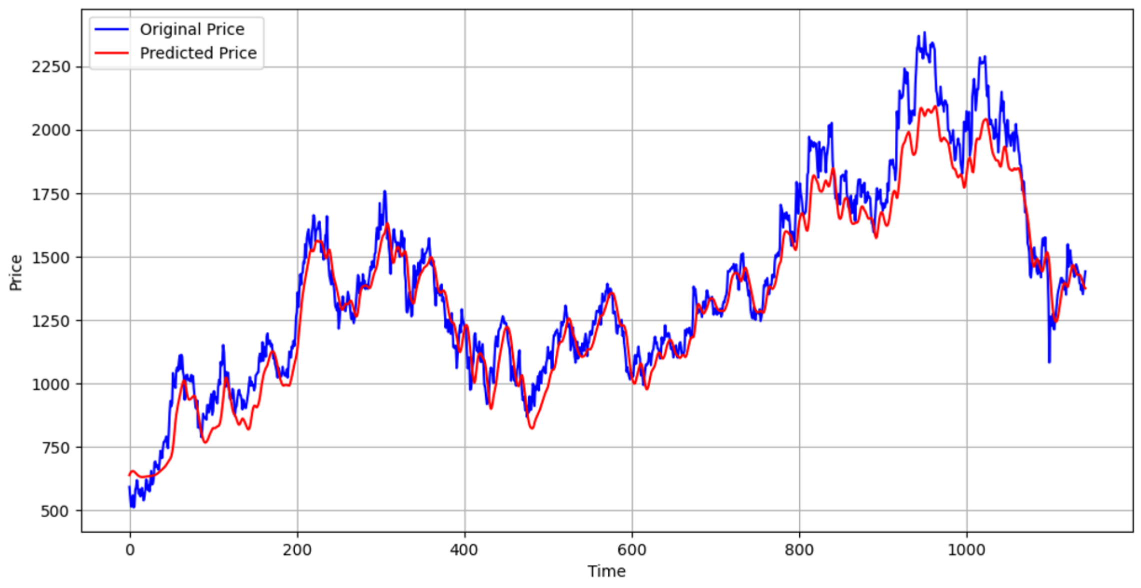

Figure 5.1: When Close Only data is consiedered

- Larger architectures such as **Transformer and CNN-LSTM** demonstrated slower convergence and required significantly more data to avoid underfitting.

5.4 Quantitative Evaluation Results

Performance was evaluated using RMSE, MAE, and MAPE metrics. The trends observed are summarized below:

TODO - Graph for Final Error Metrics Per Model

Table 5.1: Comparative Model Performance Across Datasets

Model Type	**Single-Company Data**	**NIFTY50/200 Dataset**	**Full NSE**
GRU	Best (fastest + accurate)	Poor generalization	Mode
LSTM	Very good	Best generalization	Very g
CNN-LSTM	Underfits (small dataset)	Underfits (still limited)	Performs well, c
Transformer	Underfits heavily	Weak performance	Performs well but not

The results clearly indicate that model complexity must match dataset size. Simpler recurrent models outperform complex architectures when data availability is limited.

5.5 Effect of Feature Engineering vs. Raw OHLCV Inputs

An important discovery in these experiments is that feature engineering did not always improve model performance.

Key conclusions:

- For single-company models, using only Close, Open, High, Low, Volume produced better results than using technical indicators, likely due to overfitting and noise sensitivity introduced by engineered features.
- For broader training datasets (NIFTY50, NIFTY200, or all NSE companies), adding indicators improved generalization and reduced error variance.

TODO - Insert Graphs Prediction vs Actual: LSTM vs GRU vs Indicator-Enhanced Variants

5.6 CNN-LSTM Feature Set Performance

For the CNN-LSTM architecture, three engineered feature combinations were tested:

Table 5.2: Feature Set Performance Comparison

Feature Set	**Result**
Close, MA5, MA20	Best performance
Close, EMA10, Standard Deviation	Moderate
Close return + STD20	Least consistent

This indicates that simple, trend-focused moving averages outperform more volatile or mathematically dense indicators in deep hybrid models.

TODO CNN-LSTM Prediction Comparison Across Feature Sets TODO Validation Loss Comparison Across CNN-LSTM Feature Sets

5.7 Generalization vs. Specialization Behavior

Another major insight from these experiments is that:

- Models trained on a single company become highly specialized, fitting well to patterns unique to that stock but failing to generalize across companies.
- Models trained on larger aggregated datasets show better generalization, but require feature engineering and larger architectures to extract meaningful predictive structure.

This reflects a trade-off:

- Single-company models = accuracy in isolation
- Multi-stock models = robustness and scalability

5.8 Deployment and Practical Evaluation

When deployed into live prediction with recent 3-month price windows:

- LSTM and GRU showed stable point-wise forecasts suitable for short-term use.
- Transformer and CNN-LSTM predictions improved only when trained on very large multi-company datasets.
- Simpler models responded more smoothly, while complex architectures reacted more sharply to volatility.

TODO - Live Prediction Overlay (Last 3 Months)

5.9 Discussion of Results

5.9.1 GRU Results

GRU is the best candidate model for rapid real-world deployment when working with limited data per stock.

5.9.2 LSTM Results

LSTM is the most reliable model overall, balancing performance, generalization, and robustness across dataset sizes.

5.9.3 CNN-LSTM Results

CNN-LSTM architecture requires significantly larger datasets to outperform recurrent models, and in this specific context did not surpass LSTM performance.

5.9.4 Transformer Results

Transformer architecture requires significantly larger datasets to outperform recurrent models, and in this specific context did not surpass LSTM performance.

Chapter 6

Summary and Conclusions

6.1 Summary of Work and Key Contributions

The primary objective of this research was to build an automated system capable of predicting the next-day closing stock price of Indian market equities using machine learning techniques informed by technical analysis. The project successfully developed and evaluated multiple models including GRU, LSTM, CNN-LSTM, and Transformer architecture using real historical data retrieved through the Kite Connect API.

In its final form, the project demonstrates that meaningful predictive performance can be achieved using deep learning–based time series modeling and that model behavior varies significantly depending on dataset size, architecture, and feature design. A complete working prototype system was also built, integrating live data retrieval, preprocessing, inference, and visual chart-based validation.

6.2 Problem Statement and Solution

Stock market forecasting is challenging due to volatility, noise, and nonlinear dependencies that traditional statistical and manual technical analysis struggle to model. Prior research often lacked experimentation across dataset scales, did not evaluate generalization capability, or relied only on accuracy without deployment validation.

This project addressed these gaps by:

- Comparing multiple model families rather than a single architecture
- Evaluating performance across single-company training, index-based
- datasets, and full-market datasets

- Testing models both with raw OHLCV inputs and with engineered
- technical indicator feature sets
- Implementing a live deployment and visualization pipeline for real-time inference

This systematic comparison framework provided insight into which models are best suited for various dataset conditions and use cases.

6.3 Major Findings and Significance

The experiments revealed several meaningful insights:

- GRU models performed best when trained on data from a single company, offering fast training and strong prediction accuracy even without engineered indicators.
- LSTM models were the most stable and generalizable architecture, performing well both on individual stock datasets and on NIFTY50/NIFTY200 multi-stock sets.
- Feature engineering did not universally improve results. For small datasets, raw OHLCV inputs often outperformed engineered features, likely due to reduced noise and decreased overfitting risk.
- Complex architectures like Transformers and CNN-LSTM models required large datasets to perform competitively. When trained only on index-level datasets, they underperformed simpler RNN models. However, when trained using all available NSE data, their performance improved significantly, approaching that of LSTM models.
- The best-performing feature combination for CNN-LSTM was Close, MA5, and MA20, suggesting that simple trend-focused transformations are more effective than volatile or mathematically dense indicators.

The significance of these findings lies in demonstrating that deep learning approaches remain highly promising for stock forecasting but must be matched appropriately to dataset scale and complexity. This work provides a practical roadmap for model selection and tuning in real-world financial prediction systems.

6.4 Limitations of the Study

While the project achieved its intended goals, several limitations remain:

- The dataset consisted only of daily closing data, excluding intra-day signals or volume profile dynamics.
- Technical indicators used were selected manually and were not optimized using automated feature selection or genetic search.
- The model evaluation period was limited, and live deployment testing did not incorporate transaction costs, execution slippage, or trading constraints.
- Only price-based data was used; no sentiment, macroeconomic, or alternative data sources were included.
- The real-time evaluation was observational and not connected to an automated trading engine.

These limitations do not diminish the results but highlight the complexity of building a fully production-ready forecasting system.

6.5 Future Work

Several opportunities exist to extend this research:

- Incorporate more diverse data sources, including sentiment data, financial news feeds, and macroeconomic indicators.
- Improve feature engineering, potentially using automated selection methods such as SHAP, PCA, or feature importance ranking.
- Expand model scope to intraday prediction and real-time streaming data.
- Refine hybrid architectures, including improved Transformer variants (e.g., Informer, FEDformer, or Time-Series Transformers).
- Develop a full backtesting and trading engine with portfolio simulation metrics such as maximum drawdown, Sharpe ratio, and profit factor.
- Extend generalization evaluation across sectors, volatility regimes, and economic conditions.

These improvements would help transition the model from a research prototype to a sophisticated decision support or automated execution system.

6.6 Conclusion

In conclusion, the research demonstrates that machine learning-driven technical analysis can offer valuable improvements in short-term stock price forecasting and lays the groundwork for more advanced predictive and automated trading systems. The insights gained regarding model performance, data scale sensitivity, and feature behavior contribute meaningfully to ongoing work in financial machine learning and provide a strong foundation for future advancements in intelligent stock market modelling.