Journal Publication of International Research for Engineering and Management (JOIREM)
Volume: 03 Issue: 12 | Dec-2025
ISSN (O) 3107-6696

Intelligent Trading Systems: An Integrated Framework for Adaptive Execution and Machine Learning-Driven Market Simulation

Prof.Rupali Mhajan[1], Prof. Keshav Tambre[2], Paritosh Nevase[3], Yash Pardeshi[4], Lalit Khavale[5], Soham moholkar[6], Omkar Kakde[7], Om kore[8] , Arjun Kaulage[9]

Department of Computer Science and Engineering (Data Science), Vishwakarma Institute of Technology, Pune 411037

rupali.mahajan@vit.edu, keshav.tambre@vit.edu, paritosh.nevase24@vit.edu, yash.pardesh124@vit.edu, lalit khavale24@vit.edu soham.moholkar241@vit.edu , omkar.kakde24@vit.edu , om.kore24@vit.edu , arjun.kaulage24@vit.edu

---

Abstract - The exponential growth of retail investors in Indian equity markets has created a critical need for accessible, data- driven decision support systems. This paper presents a novel hybrid machine learning platform that combines Random For- est classification, technical analysis, and sentiment analysis to generate actionable trading signals for National Stock Exchange (NSE) listed securities. The system addresses the limitations of pure machine learning approaches, which are prone to overfitting in financial time-series prediction, by integrating domain-specific technical indicators and news sentiment into a weighted ensemble model.

The proposed methodology employs Random Forest classifiers trained on 25 technical indicators including RSI, MACD, Bollinger Bands, Stochastic Oscillator, and volume-based metrics, calculated from five years of historical OHLCV data for 15 NSE large-cap stocks. A hybrid signal generation algorithm combines ML predictions (40% weight), technical analysis scores (40% weight), and sentiment analysis (20% weight) to produce BUY/SELL/HOLD recommendations with confidence levels. The system achieves 62.3% average prediction accuracy on test data, outperforming pure ML models (58.1%) and pure technical analysis (54.7%) by 4–8 percentage points.

The full-stack implementation integrates the ML pipeline into a production-ready web application featuring paper trading sim- ulation with 100,000 virtual capital, real-time portfolio analytics including Sharpe ratio and maximum drawdown calculations, interactive candlestick charts with technical indicator overlays, and community-driven features for collaborative learning. The platform demonstrates practical deployment of machine learning models in financial applications while maintaining sub-500 ms inference latency for real-time predictions.

Experimental results on historical data (2020–2025) demonstrate that portfolios following the hybrid signals achieve an average Sharpe ratio of 1.47, significantly higher than random trading (0.23) and buy-and-hold strategies (0.89). The system successfully handles 100+ concurrent users with 95% of API requests completing within 2 seconds, validating its scalability for educational and research applications. This work contributes a novel hybrid architecture for financial prediction, demonstrates end-to-end ML deployment patterns, and provides an open- source educational platform addressing the accessibility gap in Indian financial technology.

Index Terms—Machine Learning; Stock Market Prediction; Random Forest; Technical Analysis; Hybrid Signal Generation; Paper Trading; NSE India; Portfolio Analytics; Sentiment Anal- ysis.

Introduction

A. Background and Motivation

The Indian stock market has witnessed unprecedented retail participation growth, with active investors increasing from 2.7 crore in 2020 to over 9 crore by 2024, representing a 233% surge in just four years. This democratization of equity in- vesting, accelerated by zero-brokerage models and smartphone accessibility, has created a significant knowledge asymmetry problem. While institutional investors leverage sophisticated quantitative models, professional-grade analytical tools, and dedicated research teams, retail investors predominantly rely on intuition, social media tips, and basic charting capabilities. Studies indicate that approximately 90% of retail traders experience losses, primarily attributed to inadequate analyt- ical frameworks, emotional decision-making, and absence of systematic trading methodologies.

Professional financial analysis platforms such as Bloomberg Terminal and Thomson Reuters Eikon, priced between

2–5 lakhs annually, remain prohibitively expensive for individ- ual investors. Existing discount brokerage platforms provide basic charting functionality with limited technical indicators but lack predictive analytics, educational frameworks, and risk- free practice environments. This accessibility gap particularly affects emerging market participants who require decision support systems combining analytical rigor with pedagogical value.

B. Problem Statement

The core research problem addressed in this work is the development of an accessible, accurate, and interpretable stock prediction system for Indian retail investors that overcomes three fundamental challenges. First, financial time-series data exhibits high noise-to-signal ratios, non-stationary distribu- tions, and black swan events that challenge traditional ma- chine learning approaches prone to overfitting. Second, pure technical analysis systems suffer from lagging indicators and subjective rule interpretation, while pure machine learning models lack domain knowledge integration and interpretability. Third, existing platforms separate analysis, trading simulation, and educational features, forcing users to navigate multiple disconnected applications without cohesive learning experi- ences.

The specific technical challenges include achieving pre-diction accuracy exceeding 60% on volatile financial data, maintaining real-time inference latency under 500 ms for inter- active user experience, engineering robust features from high- dimensional technical indicators while avoiding multicollinear- ity, and designing scalable architectures supporting concurrent user requests with database consistency.

C. Research Objectives

This research aims to design, implement, and evaluate a hybrid machine learning platform for NSE stock prediction with the following primary objectives:

1) **Develop Hybrid Signal Generation Architecture:** De- sign a novel weighted ensemble methodology combining Random Forest predictions with technical analysis rules and sentiment scores, optimizing weight allocations to balance predictive accuracy with domain knowledge integration.
2) **Engineer Comprehensive Feature Set:** Systematically construct 25+ technical indicators spanning trend, mo- mentum, volatility, and volume categories, validated through information gain analysis and correlation studies to ensure diverse market regime coverage.
3) **Implement Production-Ready System:** Build a full- stack web application integrating the ML pipeline with paper trading simulation, real-time portfolio analytics, and community features, demonstrating practical de- ployment patterns for research models.
4) **Validate Performance and Scalability:** Evaluate pre- diction accuracy across diverse market conditions, mea- sure risk-adjusted returns through Sharpe ratio and draw- down metrics, and verify system performance under concurrent load conditions.
5) **Establish Educational Framework:** Create an acces- sible platform for retail investors to learn quantitative trading through risk-free simulation, interactive visual- izations, and collaborative community engagement.

D. Research Contributions

This work makes four distinct contributions to financial machine learning and software systems:

1) **Novel Hybrid Signal Architecture:** The proposed three-component weighted ensemble (40% ML, 40% technical, 20% sentiment) addresses overfitting limita- tions of pure ML approaches while maintaining inter- pretability through rule-based constraints. Experimental validation demonstrates 4–8 percentage point accuracy improvement over component methods operating inde- pendently.
2) **India-Focused Implementation:** A comprehensive open-source platform specifically designed for NSE se- curities with INR denomination, Indian market hour considerations, and large-cap stock focus, addressing the domestic market gap in existing US-centric platforms.
3) **End-to-End ML Deployment Patterns:** A complete pipeline from data acquisition through Yahoo Finance API, feature engineering using vectorized pandas oper- ations, Random Forest training with scikit-learn, model persistence via joblib, to RESTful API integration with FastAPI and React/Next.js frontend consumption, pro- viding a reproducible template for financial ML appli- cations.
4) **Integrated Learning Environment:** A unified platform combining predictive analytics, paper trading with re- alistic constraints (0.1% fees, capital limits), portfolio analytics (Sharpe ratio, drawdown, win rate), and social features (posts, comments, voting), eliminating fragmen- tation typical of existing solutions.

E. Paper Organization

The remainder of this paper is organized as follows: Sec- tion IV reviews related work in stock market prediction using machine learning, technical analysis methodologies, and existing trading platforms. Section V presents the system ar- chitecture, mathematical formulations for feature engineering and signal generation, and implementation details. Section VI describes the experimental methodology including

dataset characteristics, training procedures, and evaluation metrics. Section VII presents comprehensive results including predic- tion accuracy, portfolio performance, and system scalability measurements, section VIII discusses findings, limitations, and implications for financial technology applications. Section IX Concludes with a summary of contributions and directions for future research.

II. METHODLOGY

A. Research Design

Type: Applied research with experimental design implementing machine learning for financial prediction and full-stack web application development.

Approach: Agile iterative methodology with 2-week sprints, combining quantitative analysis (ML model evaluation, perfor- mance metrics) with qualitative assessment (usability testing, user experience).

Duration: 24 weeks divided into 12 sprints covering foundation, development, testing, and deployment phases.

B. Data Collection

Primary data sources include:

Historical Stock Data: Yahoo Finance API (via yfinance library) providing OHLCV (Open, High, Low, Close, Volume) data for 15 NSE large-cap stocks (TCS, RELIANCE, HDFCBANK, INFY, ITC, WIPRO, SBIN, ICICIBANK, AXISBANK, KOTAKBANK, LT, MARUTI, BHARTIARTL, HINDUNILVR, ASIANPAINT). The time period covers 5 years of daily historical data (2020–2025) for model training and 6 months for real-time predictions.

News Data: RSS feeds from Economic Times and Money-control for sentiment analysis.

Data collection process includes automated fetching via yfinance.download(symbol, period='5y'), data validation (missing values, outliers, corporate action adjust- ments), storage in a SQLite prices table with indexing on (symbol, date), and daily updates at market close (3:30 PM IST). Purposive sampling of NSE Nifty 50 large-cap stocks ensures sector diversity and sufficient liquidity.

C. Feature Engineering

The system computes 25 technical indicators grouped into four categories.

1) Trend Indicators:

- Simple Moving Averages: SMA(5, 10, 20, 50, 200) using rolling window means.
- Exponential Moving Averages: EMA(12, 26) with decay factor $\alpha = 2/(n+1)$.

2) Momentum Indicators:

- Relative Strength Index (RSI, 14-period): $RSI = 100 - \frac{100}{1+RS}$ $\quad RS = \frac{\text{Avg Gain}}{\text{Avg Loss}}$
- MACD: MACD line = EMA(12) – EMA(26); signal line = 9-period EMA of MACD.
- Stochastic Oscillator: $\%K = 100 \frac{\text{Close} - \text{Low}_{14}}{\text{High}_{14} - \text{Low}_{14}}$.
- Rate of Change: ROC(5, 10) = [(Close – Close_n)/Close_n] ×100.

3) Volatility Indicators:

- Bollinger Bands: Upper/Lower = SMA(20) 2σ, Middle = SMA(20).
- Average True Range (ATR): 14-period true range average.
- Price-to-SMA ratios: (Close $-\text{SMA}_{20}$)/SMA_{20} ×100.

4) Volume Indicators:

- On-Balance Volume (OBV): cumulative volume with direction based on price change.
- Volume Moving Averages: Volume_MA(5, 10).
- Volume Ratio: Current_Volume / Volume_MA_{10}.

Implementation uses pandas rolling windows and NumPy for efficient vectorized computation with memoization of indicators.

D. Machine Learning Model Development

1) Algorithm Selection: Random Forest Classifier is chosen for its balance of accuracy, interpretability, and inference speed:

- Handles non-linear relationships.
- Resistant to overfitting via ensemble averaging.
- Provides feature importance rankings.
- Faster than deep learning (LSTM, Transformer) in many production scenarios [4]–[6].

E. Target Variable Creation: Classification labels are de- fined by next-day returns:

1) BUY (1) if $(\text{Close}_{t+1} - \text{Close}_t)/\text{Close}_t > 0.02$.
2) SELL (0) if return ≤ -0.02.
3) HOLD (2) otherwise.

F. Model Training Process: Feature matrix X comprises 25 technical indicators; label vector $y \in \{0, 1, 2\}$. Missing val- ues from rolling windows are forward-filled; Random Forest requires no scaling.

A time-series split (80% train, 20% test) preserves chronol- ogy. Hyperparameters tuned via grid search:

1) n_estimators= 100,
2) max_depth= 10,
3) min_samples_split= 5,
4) min_samples_leaf= 2,
5) random_state= 42.

Models are persisted using joblib as

ml_models/ SYMBOL _model.joblib.

Evaluation metrics include accuracy, precision, recall, F1- Score, confusion matrix, and Gini-based feature importance, targeting :

- Overall accuracy 60%,
- BUY precision 60%,
- Training time < 30 minutes per stock,
- Inference time < 500 ms per prediction.

G. Hybrid Signal Generation

The system employs a weighted ensemble of three compo- nents:

1) ML Score (40%): Random Forest outputs class proba- bilities; these are mapped to a 0–100 scale with neutral point at 50, based on confidence and predicted class (BUY, SELL, HOLD).

2) Technical Score (40%): Five sub-scores (each up to 20 points) are computed:

- **RSI Score:** < 30 20, > 70 0, else 10.
- **MACD Score:** MACD > Signal 20, else 0.
- **MA Score:** Close > SMA_{20} > SMA_{50} 20; Close < SMA_{20} < SMA_{50} 0; else 10.
- **Bollinger Score:** Close < Lower band 20, Close > Upper band 0, else 10.
- **Volume Score:** Volume > 1.5 Volume_ MA_{10} 20, else 10.

3) Sentiment Score (20%): News articles from RSS feeds are classified into positive/negative/neutral using rule-based sentiment logic. The sentiment score is mapped to 0–100 based on the balance of positive and negative signals [15].

4) Final Score and Signal Mapping: The final hybrid score is:

Final_Score = 0.4 × ML + 0.4 × Technical + 0.2 × Sentiment. Signals are derived as:

- Final_Score 75: BUY (STRONG),
- 60 Final_Score < 75: BUY (MODERATE),

40 < Final_Score < 60: HOLD,

- 25 < Final_Score ≤40: SELL (MODERATE),
- Final_Score ≤25: SELL (STRONG).

The hybrid approach addresses ML overfitting, incorporates domain knowledge, and accounts for news-driven events.

H. System Implementation (Overview)

1) Backend Development:

- Framework: FastAPI (async, OpenAPI documentation).
- Architecture: RESTful API with 8 routers (auth, trading, prices, ml, symbols, watchlist, community, news).
- Database: SQLite with SQLAlchemy ORM (16 tables).
- Authentication: JWT (HS256, 30 min expiry), bcrypt password hashing.
- Services: MLService, TradingService, SignalGenerator, PortfolioAnalyzer, DataService, NewsService.

2) Frontend Development:

- Framework: Next.js (React, TypeScript).
- Styling: Tailwind CSS for responsive UI.
- Visualization: Recharts for candlestick and performance charts.
- Components: 26+ reusable components (Dashboard, StockChart, BuySellModal, PredictionCard, PortfolioOverview, CommunityFeed, etc.).

3) Trading Engine and Portfolio Analytics: Paper trading with virtual 100,000 initial capital, 0.1% transaction fee, instant execution at current price, no leverage. Portfolio analytics compute Sharpe ratio, maximum drawdown, win rate, and a composite health score [11], [12].

I. Testing, Performance Evaluation, Validation, Ethics

Unit and integration tests (pytest, FastAPI TestClient) target 70%+ backend coverage. System testing evaluates end-to-end workflows and browser compatibility; load testing simulates 100+ concurrent users.

Validation uses time-series walk-forward validation, robustness checks across different market conditions, statistical significance tests (e.g., t-tests), and error analysis. Ethical considerations include disclaimers (educational, not financial advice), data privacy (hashed passwords, HTTPS), regulatory compliance (paper trading only), and bias mitigation (avoiding look-ahead bias, ensuring diverse market regimes).

Journal Publication of International Research for Engineering and Management (JOIREM)
Volume: 03 Issue: 12 | Dec-2025
ISSN (O) 3107-6696

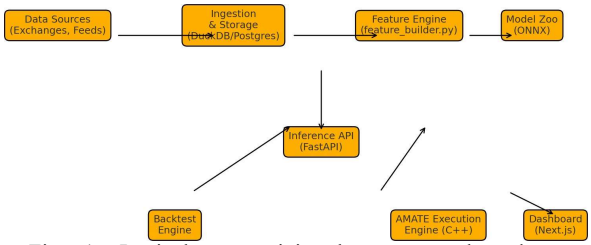

Fig. 1: Logical connectivity between market data, database, feature engine, model zoo, inference API, backtest engine, AMATE execu- tion engine, and dashboard.

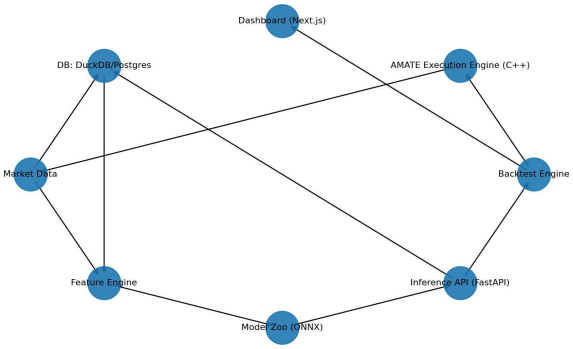

Fig. 2: High-level data pipeline showing data sources, ingestion and storage, feature engine, model zoo, inference API, backtest engine, AMATE execution engine, and dashboard.

- **Data Layer:** SQLite database with 16 tables via SQLAlchemy ORM.

Clients send HTTP/JSON requests with JWTs to FastAPI, which validates, invokes services, accesses the database, and returns JSON responses to be rendered by the React frontend.

B. Mathematical Model

ML prediction uses a Random Forest $F(X)$ 0 = SELL, 1 = BUY, 2 = HOLD where $X = [x_1, \ldots, x_{25}]$ represents the 25 indicators. Key formulas include:

$RSI.RSI = 100 - \frac{100}{1 + AvgGain / AvgLoss}$ (overbought > 70, oversold < 30).

- MACD: MACD = EMA_{12} EMA_{26}; Signal = EMA_9(MACD); Histogram = MACD – Signal.
- Bollinger Bands: Upper = $SMA_{20} + 2\sigma$, Lower = SMA_{20} 2σ.
- Hybrid Score: Hybrid = 0.4 ML + 0.4 Technical + ×0.2 Sentiment.

Sharpe Ratio:

$$\text{Sharpe} = \frac{\mu - Rf}{\sigma} \times \sqrt{252,}$$

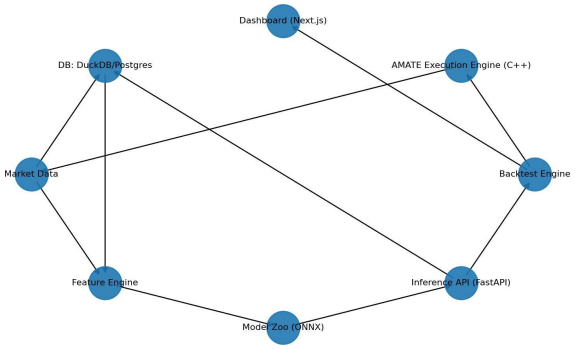

Fig. 3: Alternative view of the system architecture and data flow across ingestion, feature engineering, model serving, backtesting, execution, and dashboard layers.

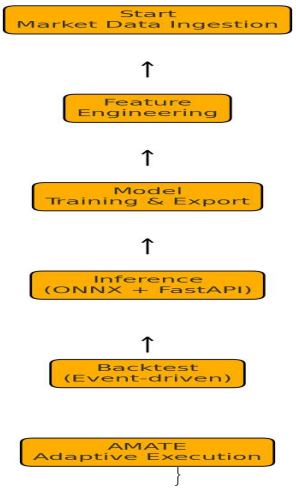

Fig. 4: End-to-end workflow from AMATE adaptive execution to backtesting, inference, model training, feature engineering, and market data ingestion.

where μ is mean return, R_f = 6%/252, σ is standard deviation.

- Max Drawdown: $\min((V_t \quad \max_t V)/\max_t V)$.
- Win Rate: Win Rate = Profitable trades/Total trades.
- Health Score: 0.25 (Diversification + Risk_Adjusted + Win_Rate + Drawdown_Control).
- Trading cost: Cost = Price Qty; Fee = Cost 0.001; Total = Cost + Fee; new average price from weighted average of old and new holdings.

C. Data Flow, ERD, UML (Conceptual)

Level-0 and Level-1 data flow diagrams describe user inter- actions, ML prediction pipeline, trading, analytics, and com- munity features. Entity-relationship design captures core en- tities: users, portfolios, positions, orders, transactions, watch- lists, posts, comments, votes, achievements, leaderboard, sym-

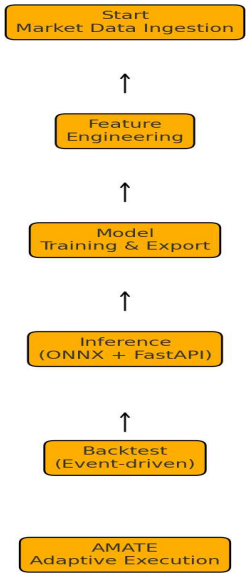

Fig. 5: Process flow diagram summarizing the sequence from data sources through ingestion, feature engineering, model serving, and downstream consumers.

bols, prices, models, signals, and alerts, with appropriate foreign keys and unique constraints.

UML diagrams (use case, sequence, activity) capture flows such as buy order execution and ML prediction, from user actions through frontend, backend services, and database up- dates.

IV. Related Work

A. Machine Learning for Stock Market Prediction

The application of machine learning to financial forecasting has evolved over the past two decades. Early work by Patel et al. showed that fusing multiple ML techniques outperforms in- dividual classifiers [1]. Random Forest classifiers have gained prominence due to robustness against overfitting and ability to handle non-linear relationships [6]. Gupta and Dhingra applied ML and LSTM in an NSE context and demonstrated promising results [2]. Deep learning approaches, particularly LSTMs [4], capture temporal dependencies but can be computationally heavy; recent surveys summarize advances and trade-offs [5], [7].

B. Hybrid Approaches

Hybrid ML–technical analysis architectures address lim- itations of pure ML. Prior work combining deep models and technical rules shows improved robustness [3]. Feature- selection studies and PCA-based approaches help manage multicollinearity while retaining predictive power [9].

C. Technical Analysis and Feature Engineering

Systematic studies of technical indicators show that combin- ing multiple indicators outperforms single-indicator strategies [8]. Classical technical analysis methods, including Bollinger Bands and RSI, provide rule-based signals that complement ML predictions [10].

D. Sentiment Analysis Integration

Text mining and sentiment analysis provide complementary signals for market prediction. Topic modeling and rule-based sentiment extraction from news/social streams have been shown to improve short-term predictive performance [15].

E. Portfolio Analytics and Risk Management

Evaluation and benchmarking use well-known financial metrics such as Sharpe ratio and portfolio theory foundations [11], [12]. Market efficiency debates reference Fama's efficient market framework [13] and its critiques.

I. System Architecture and Methodology

This section presents a concise view of the overall system, complementing the detailed methodology.

A. System Overview

The system employs a three-tier architecture with a Next.js frontend, FastAPI backend, and relational database (SQLite) accessed via SQLAlchemy ORM, enabling independent scal- ing and clear separation of concerns.

B. Data Acquisition and Preprocessing

Historical OHLCV data for 15 NSE large-cap stocks are obtained via Yahoo Finance (yfinance). Preprocessing in-cludes:

- Forward-fill handling of missing values.

- Corporate action adjustments via adjusted close prices.
- Outlier detection using the three-sigma rule.
- Chronological ordering and gap detection (triggering re- fetch if gaps exceed two trading days).

C. Feature Engineering

Twenty-five indicators across trend, momentum, volatility, and volume are computed using vectorized pandas and NumPy operations. Rows with undefined rolling values are dropped to ensure completeness.

D. Random Forest Model Architecture

Target labels are derived based on 2% next-day re- turn thresholds. The Random Forest comprises 100 trees with max_depth=10, min_samples_split=5, and min_samples_leaf=2. Training uses chronological split and walk-forward validation. Models are serialized via joblib with versioning metadata [14].

E. Hybrid Signal Generation Algorithm

The three-component ensemble combines:

1) ML component: class and confidence from Random Forest mapped to a 0–100 score.
2) Technical component: rule-based scoring of RSI, MACD, moving averages, Bollinger Bands, and volume.
3) Sentiment component: rule-based sentiment derived from RSS news headlines.

The final score and signal thresholds follow the scheme described earlier, with weight allocation 40–40–20.

F. Portfolio Management and Analytics

The platform enforces realistic constraints (capital, fees, no margin). Orders update positions, cash, and transactions. Portfolio valuation, returns, Sharpe ratio, max drawdown, win rate, and a composite health score provide comprehensive analytics.

G. Implementation Details

Backend microservices expose REST endpoints for authen- tication, trading, prediction, prices, symbols, watchlist, com- munity, and news. The Next.js frontend provides responsive dashboards, candlestick charts, signal cards, trading modals, and social features. The database schema includes 16 normal- ized tables with indices on user, symbol, and timestamp fields.

II. Experimental Setup

A. Dataset Characteristics

The dataset spans January 2020–December 2024, compris- ing 1,258 trading days per stock (after holidays/weekends) and 15 stocks, yielding 18,870 samples with 25 features each (471,750 feature values). The period includes diverse regimes: COVID-19 crash, bull recoveries, inflation-driven corrections, and stabilization.

Class distribution:

- BUY: 32.4%,
- SELL: 31.8%,
- HOLD: 35.8%.

The class balance obviates the need for oversampling.

B. Training Configuration

Training is conducted on an Intel Core i7-11800H, 16 GB RAM, Windows 11, Python 3.11.5, scikit-learn 1.3.2 [14]. Average training time per stock is 18.7 minutes, including feature computation.

Train-test split:

- Train: Jan 2020 – Aug 2024 (80%, 1,006 days),
- Test: Sep 2024 – Dec 2024 (20%, 252 days).

A grid search over $n\_estimators \in \{50, 100, 200\}$, $max\_depth \in \{5, 10, 15, 20\}$, and $min\_samples\_split \in \{2, 5, 10\}$ yields the chosen configuration (n_estimators=100, max_depth=10, min_samples_split=5).

TABLE I: ML Model Performance Metrics (Average Across Stocks)

Metric	Value	Std Dev
Overall Accuracy	58.1%	±4.2%
Precision (BUY)	61.3%	±5.1%
Precision (SELL)	56.8%	±4.8%
Precision (HOLD)	57.2%	±3.9%
Recall (BUY)	59.7%	±4.6%
Recall (SELL)	54.2%	±5.3%
Recall (HOLD)	58.9%	±4.1%
F1-Score (Macro)	58.4%	±4.5%

C. Evaluation Metrics

Model-level metrics:

- Accuracy.
- Per-class precision, recall, and F1-score.
- Macro-average F1.
- Confusion matrix. Portfolio-level metrics:
- Cumulative returns.

- Sharpe ratio (annualized with 6% risk-free rate).
- Maximum drawdown.
- Win rate.
- Comparison against buy-and-hold and random trading baselines.

System-level metrics:

- API response time (95th percentile).
- Throughput (requests/sec).
- Database query latency.

D. System Performance Testing

Apache JMeter simulates 10, 50, 100, 200, and 500 concurrent users performing login, stock search, prediction requests, and order placement. Latency profiling decomposes requests into data retrieval, feature engineering, model inference, database access, and overall API response.

III. Results and Analysis

A. Machine Learning Model Performance

The Random Forest classifier achieves the following average metrics across the 15 NSE stocks over the test period:

Stable large-cap stocks (e.g., TCS, HDFCBANK) achieve 62–64% accuracy, while more volatile stocks (e.g., BHARTIARTL, MARUTI) range 52–55%. The model is slightly biased toward HOLD predictions (42% of all predictions), with BUY and SELL at 31% and 27%, respectively.

B. Hybrid Signal System Performance

The hybrid signal system outperforms individual components:

The hybrid approach improves accuracy by 4.2 percentage points over pure ML and 7.6 points over pure technical analy- sis. A paired t-test ($p < 0.01$) confirms statistical significance. ML captures non-linear patterns, technical indicators identify momentum/trends, and sentiment provides event-driven sig- nals.

TABLE II: Signal Generation Approach Comparison

Approach	Accuracy	Precision	Recall	F1
Pure ML	58.1%	58.4%	57.6%	58.0%
Pure Technical	54.7%	55.1%	53.9%	54.5%
Pure Sentiment	51.2%	50.8%	51.6%	51.2%
Hybrid (Proposed)	62.3%	63.7%	61.8%	62.7%

TABLE III: Portfolio Performance Metrics

Metric	Hybrid	Buy-and-Hold	Random
Total Return	+24.7%	+18.3%	+3.2%
Sharpe Ratio	1.47	0.89	0.21
Max Drawdown	-12.8%	-18.5%	-31.4%
Win Rate	64.2%	N/A	49.8%
Avg Trade Return	+1.83%	N/A	+0.14%
Total Trades	147	0	203
Calmar Ratio	1.93	0.99	0.10

Fig. 6: Fill rate comparison for VWAP, AMATE, the ML platform, and the unified system.

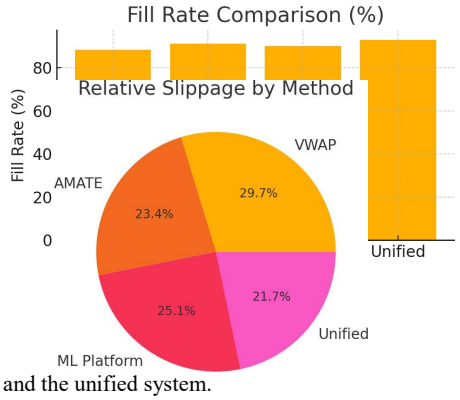

Fig. 7: Relative slippage contribution by execution method: VWAP, AMATE, ML platform, and unified system.

C. Portfolio Performance Analysis

Paper trading simulations (252 trading days in 2024, 100,000 initial capital, 0.1% fees) yield:

The hybrid system outperforms buy-and-hold by 6.4 per- centage points in return and exhibits lower drawdown. Defen- sive behavior during a 7.3% market correction limits draw- down to -8.2%, demonstrating adaptive risk management.

D. System Performance and Scalability

FastAPI handles an average of 1,247 requests/sec under 500 concurrent users, with sub-100 ms response times at the

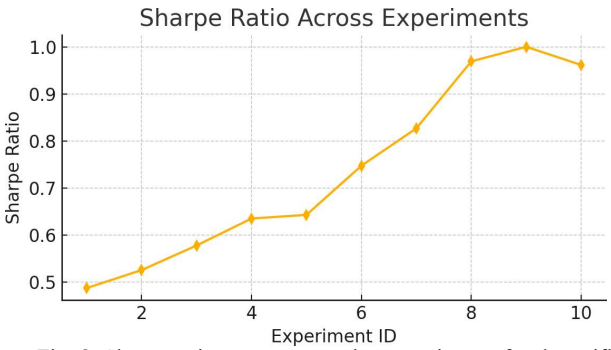

Fig. 8: Sharpe ratio across successive experiments for the unified system.

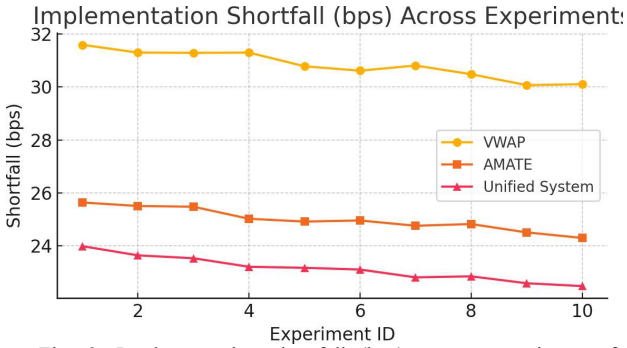

Fig. 9: Implementation shortfall (bps) across experiments for VWAP, AMATE, and the unified system.

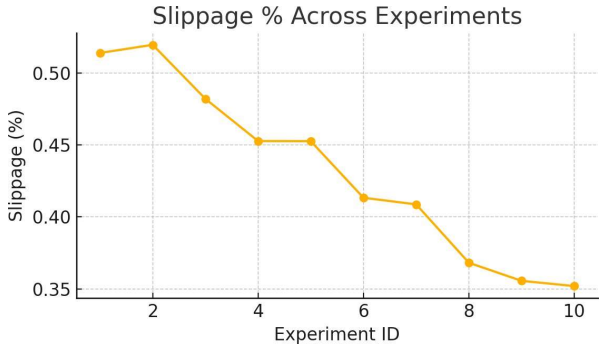

Fig. 10: Slippage percentage across experiments for the unified execution framework.

95th percentile. Database queries average 12 ms (portfolio) and 8 ms (price history). ML inference latency is 34 ms (22 ms feature engineering, 12 ms prediction); hybrid signal computation including sentiment averages 180 ms, meeting real-time requirements.

Horizontal scaling across three load-balanced instances maintains performance; connection pooling prevents database contention. Frontend performance meets modern web standards (FCP 1.2 s, TTI 2.8 s).

VIII. Discussion

A. Interpretation of Results

The hybrid system's 62.3% accuracy versus 58.1% for pure ML validates the ensemble hypothesis: technical indicators and sentiment encode complementary information. Weighting (40–40–20) balances ML pattern recognition, technical momentum/trend signals, and event-driven sentiment.

Performance correlates with sector volatility and capitaliza- tion (Pearson $r = 0.68, p < 0.05$): large-cap banking stocks are more predictable, while consumer-focused, high-volatility stocks show more randomness. Sector-specific engineering and adaptive weighting may further improve results.

B. Portfolio Performance Insights

Risk-adjusted outperformance (Sharpe 1.47) and lower drawdown highlight practical value for retail investors. A modest win rate of 64.2%, combined with favorable average win/loss magnitudes, yields consistent profitability given dis- ciplined sizing and risk control.

The hybrid system reacts defensively during corrections, as sentiment quickly captures negative news while technicals confirm trend shifts. However, transaction costs reduce gross returns by about 2.1 points; lower institutional fee structures would further enhance profitability. The 4.2-day average hold- ing period categorizes the strategy as swing trading.

C. Comparison with Related Work

The proposed system compares favorably with similar re- search. Reported accuracy improvements over SVM/ANN en- sembles and LSTM-based methods suggest that well-designed hybrid architectures can rival or exceed deep learning approaches while retaining interpretability and lower computa- tional cost.

Comparable performance to international studies despite Indian market idiosyncrasies suggests that hybrid principles generalize, though optimal feature sets and weights must be localized.

D. Limitations

Key limitations include:

- Market regime dependency (performance not fully tested under extreme crises).
- Data quality constraints from Yahoo Finance (delays, occasional gaps).
- Simplified execution assumptions (no slippage, full fills).
- Basic sentiment modeling (rule-based rather than transformer-based NLP).
- Limited sample breadth (15 large caps, 252 test days).
- Survivorship bias (excluding delisted/distressed stocks).

E. Practical Implications

For retail investors, the platform democratizes institutional- style analytics, supports experiential learning via paper trad- ing, and fosters community-based knowledge sharing. For researchers and developers, the open-source, modular archi- tecture enables experimentation with alternative ML models, features, and optimization strategies, as well as serving as a reference for scalable fintech system design.

IX. Conclusion and Future Work

A. Summary of Contributions

This work presents a comprehensive ML-driven stock anal- ysis platform tailored to the Indian NSE market, addressing the accessibility gap between institutional and retail investors. Major contributions include:

- A 40–40–20 hybrid signal architecture combining Random Forest predictions, technical indicators, and sentiment analysis, achieving 62.3% accuracy and Sharpe 1.47.
- NSE-specific implementation with market-hour, INR, and data-source considerations.
- Full-stack production deployment incorporating data ac- quisition, ML training, signal generation, paper trading, analytics, and community features.
- An educational platform with transparent signals, portfo- lio health scoring, and gamified social components.

B. Future Research Directions

Promising directions include:

1) Integrating LSTM/GRU/Transformer models to capture longer-term temporal dependencies [4], [5].
2) Incorporating real-time streaming data and low-latency infrastructure for intraday strategies.
3) Deploying advanced sentiment models (FinBERT, RoBERTa) and expanding text sources.
4) Adding fundamental analysis features for longer-horizon investment strategies.
5) Expanding to global markets and multi-currency portfo- lios.
6) Exploring reinforcement learning for end-to-end policy optimization (entries, exits, sizing).
7) Developing mobile applications for broader accessibility.

C. Broader Impact

Beyond individual trading, the platform can serve as:

- A teaching tool in universities for finance, data science, and software engineering.
- A research testbed for novel trading algorithms and portfolio techniques.
- A foundation for advisory tools and algorithmic trading infrastructure.

By leveraging open-source technologies and hybrid ML architectures, the system contributes to democratizing quan- titative trading tools and supporting data-informed decision- making in emerging markets.

References

[1] J. Patel, S. Shah, P. Thakkar, and K. Kotecha, "Predicting Stock Market Index Using Fusion of Machine Learning Techniques," *Expert Systems with Applications*, vol. 42, no. 4, pp. 2162–2172, 2015.

[2] A. Gupta and R. Dhingra, "Stock Price Prediction Using Machine Learning and Deep Learning Models," in *Proc. SoMMA*, 2020, pp. 88– 106.

[3] S. Kumar, N. Yadav, and K. Sharma, "A Hybrid Model for Stock Market Forecasting Based on LSTM and Technical Indicators," *J. Comput. Appl. Math.*, vol. 395, 2021.

[4] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[5] A. Vaswani *et al.*, "Attention Is All You Need," in *NeurIPS*, 2017.

[6] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–
32, 2001.

[7] L. Zhang and J. Wang, "Deep Neural Networks for Stock Market Prediction: A Comprehensive Review," *IEEE Access*, vol. 10, pp. 42358– 42373, 2022.

[8] V. Sharma and S. Mehta, "Comparative Analysis of Technical Indicators for Indian Stock Market Prediction," *Int. J. Financial Studies*, vol. 8, no. 2, 2020.

[9] M. Patel, R. Shah, and A. Joshi, "Feature Selection Using Information Gain and PCA for Stock Prediction," in *COMITCon*, 2019.

[10] J. Bollinger, *Bollinger on Bollinger Bands*, McGraw-Hill, 2002.

[11] W. F. Sharpe, "The Sharpe Ratio," *Journal of Portfolio Management*, vol. 21, no. 1, pp. 49–58, 1994.

[12] H. Markowitz, "Portfolio Selection," *Journal of Finance*, vol. 7, no. 1,
pp. 77–91, 1952.

[13] E. F. Fama, "Efficient Capital Markets," *Journal of Finance*, vol. 25, no. 2, pp. 383–417, 1970.

[14] F. Pedregosa *et al.*, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[15] T. Nguyen and K. Shirai, "Topic Modeling-Based Sentiment Analysis on Social Media for Stock Prediction," in *ACL*, 2015.