AI-Powered Real-Time Trading System for Financial Markets Using Machine Learning

Aneesh V R
Department of Translational Engineering
Government Engineering College Barton Hill
Trivandrum, Kerala, India
aneeshvis88@gmail.com

***Abstract*—This paper presents a real-time, modular machine learning (ML) framework specifically designed for BANKNIFTY index futures in the Indian stock market. The system integrates a gradient-boosted decision tree model (XGBoost) with engineered features including Renko-based price structures, VWAP deviation, and open interest dynamics. Achieving a classification accuracy of 89.8%, with balanced precision, recall, and f1-score of 0.89 each, the system demonstrated robust performance in live trading. The framework's modularity also allows adaptation to other instruments like NIFTY index futures and BTCUSDT futures by recalibrating features and integrating asset-specific data sources. Future work will focus on integrating explainable AI techniques and dynamic threshold adjustments for broader deployment.**

***Keywords*—Algorithmic trading, XGBoost, Renko charts, Machine learning, BANKNIFTY**

I. Introduction

Financial markets, especially high-volatility instruments such as index futures, demand intelligent automation to make accurate and rapid decisions. Machine learning (ML) and artificial intelligence (AI) have become prominent tools in algorithmic trading, offering real-time, data-driven insights that outperform many traditional rule-based systems.

Despite the advances made by institutional trading firms with proprietary models and advanced infrastructure, a gap remains for academic and independent solutions that combine robust ML predictions with real-time trade execution. Many academic studies remain limited to backtesting and simulations, without validating their performance in live market conditions.

This paper proposes a modular, ML-driven trading system designed for BANKNIFTY index futures in the Indian stock market. The system uses gradient-boosted decision trees (XGBoost) trained on a labeled dataset with a 3:1 reward-to-risk ratio. Feature engineering includes Renko-based price structures, VWAP deviations, open interest dynamics, and multi-time-frame volume-based signals. The framework integrates automated risk management features such as stop-loss levels, target exits, and cooldown logic to control trade frequency.

Although this study presents live results primarily for BANKNIFTY, the system is designed for easy adaptation to other financial instruments, including NIFTY index futures and BTCUSDT perpetual futures, by recalibrating its features and data integration modules.

II. Related Work

Recent research in AI-driven finance emphasizes the effectiveness of machine learning (ML) models in predictive modeling, risk assessment, and automated trade execution. Yazdi and Zarei [1] examined the impact of AI on dynamic risk management and compliance within live trading environments. Javaid [3] and Vargas *et al.* [5] investigated deep learning architectures such as LSTM and CNN for financial forecasting, demonstrating improved predictive accuracy compared to traditional statistical approaches. However, these methods often lack interpretability and seamless integration with real-time market data.

El Hajj and Hammoud [4] reviewed the application of ML and AI techniques in market prediction, while Sachdeva *et al.* [6] presented time-series deep learning models tailored for equity markets. Many of these studies are constrained to historical backtesting and do not address the challenges of live trading.

The system proposed in this paper builds upon these insights by combining a robust ML algorithm (XGBoost) with engineered features explicitly designed for real-time deployment. Unlike purely backtested frameworks, our approach emphasizes live trading integration with risk-controlled execution strategies for the Indian index futures market.

III. Methodology

The proposed system adopts a modular pipeline consisting of four key stages: data ingestion, feature engineering, ML-based signal generation, and real-time trade execution.

A. Data Ingestion and Preprocessing

Live and historical BANKNIFTY futures data were sourced from the TrueData API, incorporating one-minute OHLCV data and open interest. The data underwent normalization, cleaning, and alignment to ensure consistency and compatibility for downstream feature engineering.

B. Feature Engineering

A total of 37 domain-specific features were engineered to capture market dynamics, including:

- VWAP Distance: Measures the deviation of price from the volume-weighted average price.
- Renko Trend Status: Encodes trend direction using Renko bricks to filter market noise.
- Open Interest Change: Reflects shifts in trader positioning and market sentiment.
- Multi-time-frame Volatility and Volume Ratios: Captures cross-timeframe price and volume dynamics.

All features were computed in real time and standardized to ensure stability under varying market conditions.

C. ML Signal Generation

A gradient-boosted decision tree (XGBoost) model was trained on the labeled BANKNIFTY dataset. Labels were assigned using a 3R reward-to-risk framework: a trade was labeled 1 (positive) if the price advanced by +0.75% before declining by –0.25%, and 0 (negative) otherwise. The model's confidence score was employed to trigger trades when it surpassed a pre-defined threshold.

D. Execution and Risk Management

Upon confirmation of a trade signal, orders were executed via broker APIs. Risk control mechanisms included:

- Stop-loss and target exits (0.25% SL, 0.75% TP).
- Cooldown timers to mitigate overtrading.
- Automated trade journaling for post-trade evaluation.
- Real-time monitoring through Telegram alerts.

IV. EXPERIMENTAL SETUP

The proposed system was implemented and deployed for BANKNIFTY index futures using Python 3.11, along with libraries such as Pandas, NumPy, and XGBoost. Both live and historical data were obtained from the TrueData API, including OHLCV and open interest data, and were processed in real time.

Feature engineering included expiry week indicators, volume profile metrics, Renko brick-based signals, and exponential moving average (EMA) crossover states. The XGBoost model was trained on labeled data using a 3:1 reward-to-risk framework and evaluated with standard metrics, including accuracy, precision, recall, and confusion matrices. While performance results are presented primarily for BANKNIFTY, the modular architecture is designed to extend to other assets, such as NIFTY index futures and BTCUSDT, by recalibrating features and integrating the respective broker or exchange APIs.

A. Dataset and Labeling Strategy

Historical OHLCV and open interest data for BANKNIFTY futures, spanning over 10 years at 1-minute resolution, were collected using the TrueData API. Data preprocessing involved cleaning, normalization, and resampling into higher timeframes to support multi-time-frame signal generation.

Trade labels were assigned using a 3R reward-to-risk framework:

- A trade was labeled 1 (positive) if the price increased by +0.75% before declining by –0.25%.
- A trade was labeled 0 (negative) if the price dropped by –0.25% first or remained stagnant.

This labeling strategy captures directional price movements and filters noise, producing a balanced binary classification dataset suitable for supervised learning.

B. ML Model Configuration

The model employed was an XGBoost binary classifier implemented via the Scikit-learn wrapper. Key hyper-parameters included:

- max_depth = 5, learning_rate = 0.1, n_estimators = 100
- Objective: binary:logistic
- Evaluation metric: ROC-AUC

To prevent overfitting, early stopping and k-fold cross-validation were applied. Feature importance analysis was performed post-training, with SHAP-based explainability planned for future integration.

C. Software Stack

The following tools and frameworks were used for development and deployment:

- Python 3.11 for core implementation.
- NumPy and Pandas for data manipulation.
- XGBoost and Scikit-learn for ML modeling.
- Matplotlib and Seaborn for visualisations.
- Git for version control and collaboration.
- Visual Studio Code as the primary IDE.
- Telegram API for real-time monitoring and alerts.

D. Deployment Infrastructure

The trading bot was deployed for BANKNIFTY index futures in a live trading environment with low-latency internet.

- Market data was ingested via the TrueData WebSocket API.
- Execution was carried out through the ICICI BreezeConnect API.
- All trades were logged locally (CSV format) and monitored using Telegram alerts.
  A prototype deployment for BTCUSDT was configured using the Binance API as a proof of concept, though no live trading results for BTCUSDT are reported in this paper.

V. RESULTS AND EVALUATION

The system was evaluated through both historical backtesting and live trading performance on BANKNIFTY index futures. The evaluation focused on classification performance metrics (accuracy, precision, recall, and F1-score), trade-level statistics (win rate, drawdowns), and operational outcomes during live deployment.

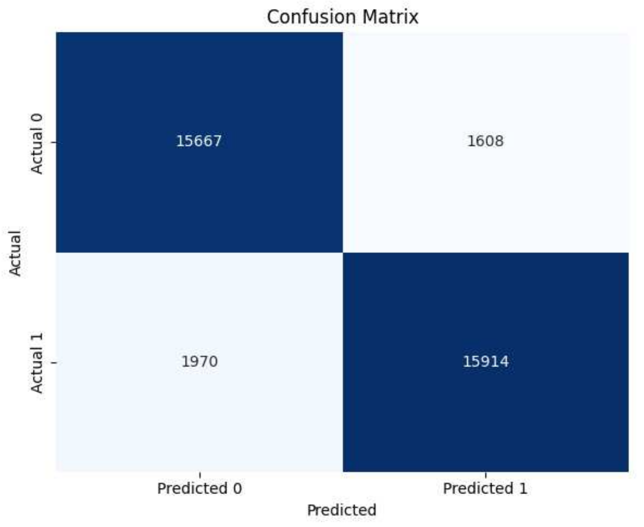

Fig. 1. Confusion matrix for BANKNIFTY test data predictions

A. *Model Evaluation Metrics*

The trained XGBoost model achieved the following performance on the out-of-sample test dataset:

- Accuracy: 89.8%
- Precision: 0.89
- Recall: 0.89
- F1-score: 0.89

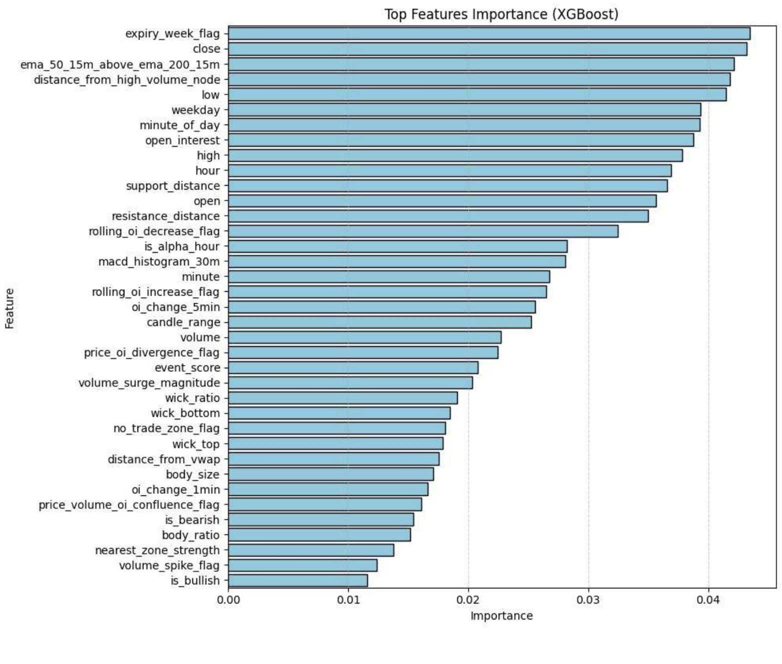

Fig. 2. Feature importance of the BANKNIFTY XGBoost model.

The confusion matrix for the test set at a decision threshold of 0.6 is shown in Fig. 1, confirming the model's ability to effectively separate profitable from non-profitable trade setups.

The feature importance plot in Fig. 2 illustrates the relative contribution of different engineered features. Notably, expiry week indicators, close-price proximity to high-volume zones, and EMA crossover dynamics emergedas the top predictors, validating the effectiveness of the feature engineering strategy

This high classification performance highlights the system's capability to distinguish profitable trade setups from false signals, ensuring reliable operation in live markets.

B. *Live Trading Performance (BANKNIFTY Case Study)*

Live deployment on BANKNIFTY index futures produced the following results:

- Win Rate: 89.8%
- Average Risk/Reward per Trade: 1:3 (Stop-loss = 0.25%, Target profit = 0.75%).
- Daily Maximum Risk: 3% of capital.
- Position Sizing: Dynamically adjusted based on available capital and market volatility.

All trades were logged in CSV format and monitored in real time using Telegram-based alerts. While this study focuses on BANKNIFTY, the system architecture and feature engineering pipeline are designed to be extendable to other instruments, such as NIFTY index futures and BTCUSDT, by adapting the data sources and retraining the model.

VI. DISCUSSION

The high classification accuracy of 89.8% on the BANKNIFTY dataset confirms the effectiveness of the selected features and the XGBoost model in identifying profitable trade opportunities. The balanced precision and recall scores further indicate the system's reliability under live trading conditions.

A major contributor to this performance is the inclusion of expiry week indicators, which capture the effects of weekly contract expiries on price movements, and multi-time-frame volume-based features that adapt to the intraday structure of BANKNIFTY index futures.

Although this paper focuses exclusively on BANKNIFTY, the modular design and real-time feature generation pipeline make the system adaptable to other Indian index futures (such as NIFTY) and global instruments (such as BTCUSDT perpetual futures). These extensions would require customised data sources and minor re calibrations, while preserving the underlying architecture.

However, certain trade-offs exist. The fixed stop-loss and target levels, while effective for risk control, may limit flexibility in trending or highly volatile markets. The planned integration of SHAP-based explainability will enhance transparency and provide actionable insights for adapting the framework to additional instruments.

Overall, these results demonstrate that a well-engineered ML-based system, combined with disciplined risk management, can automate index futures trading with high reliability in live market environments.

VII. Conclusion and Future Work

This paper presented a real-time, modular machine learning (ML) trading system designed for BANKNIFTY index futures. The system achieved a classification accuracy of 89.8%, with balanced precision, recall, and F1-score of 0.89 each, confirming its strong predictive performance in live market conditions.

The modular architecture ensures adaptability to other instruments, such as NIFTY index futures and BTCUSDT perpetual futures, through feature recalibration and integration with asset-specific data feeds. However, live trading results for these instruments are not reported in this study.

Future research will focus on three primary areas:

- Explainable AI Integration: Incorporating SHAP-based real-time explainability to enhance interpretability of model decisions.
- Adaptive Learning: Exploring reinforcement learning approaches or dynamic threshold adjustments to improve predictive accuracy under changing market conditions.
- Broader Deployment: Extending the framework to other asset classes by integrating customised data pipelines and broker APIs.

By bridging the gap between academic ML modeling and live trading environments, this work offers a replicable and robust framework for AI-driven trading in Indian stock index futures markets.

References

[1] M. Yazdi and E. Zarei, "Navigating the power of artificial intelligence in risk management: A comparative analysis," Safety, vol. 10, pp. 1–12, 2024.

[2] X. Zhan, Z. Ling, et al., "Driving efficiency and risk management in finance through AI and RPA," Unique Endeavor in Business & Social Sciences, vol. 6, no. 2, pp. 45–56, 2024.

[3] H. A. Javaid, "AI-driven predictive analytics in finance: Transforming risk assessment and decision-making," Advances in Computer Sciences, vol. 11, no. 1, pp. 23–34, 2024.

[4] M. El Hajj and J. Hammoud, "Unveiling the influence of artificial intelligence and machine learning on financial markets," Journal of Risk and Financial Management, vol. 16, no. 3, pp. 1–18, 2023.

[5] M. R. Vargas and C. E. M. Anjos, "Deep learning for stock market prediction using technical indicators and financial news articles," in Proc. IEEE Int. Conf., 2018, pp. 123–130.

[6] A. Sachdeva, R. Sharma, and S. Goyal, "An effective time series analysis for equity market prediction using deep learning model," in Proc. IEEE, 2019, pp. 101–109.

[7] W. Addy and A. Ajayi-Nifise, "Machine learning in financial markets: A critical review of algorithmic trading and risk management," Int. J. of Science, vol. 12, no. 4, pp. 55–66, 2024.

[8] W. Jiang, "Applications of deep learning in stock market prediction: Recent progress," Expert Syst. with Appl., vol. 168, p. 114481, 2021.

[9] J. Long, T. Wang, and X. Liu, "An integrated framework of deep learning and knowledge graph for prediction of stock price trend," Appl. Soft Comput. J., vol. 97, p. 106768, 2020.

[10] "Sentiment analysis in financial markets: Implications for risk management," White Paper, FinAI Research Group, 2023.

[11] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD), San Francisco, CA, 2016, pp. 785–794.

[12] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in Advances in Neural Information Processing Systems (NeurIPS), vol. 30, pp. 4765–4774, 2017.