IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

IntelliQuant: Regime-Aware Stock Market Trend Analysis Using Hidden Markov Models

Arindam Jaiman[1], Ananya Jaiswal[2], S. Poornima[3]

Department of CSE (Emerging Technologies), SRM Institute of Science and Technology, Vadapalani Campus

Chennai, India[1-3]

Abstract: Stock prices are noisy and non-stationary, and they move between calm, trending and stressed conditions that call for different interpretation, so a single model fitted to a whole price history can average across behaviours it should distinguish. This paper presents IntelliQuant, a market-intelligence platform for Indian equities that organises trend analysis around latent market regimes inferred by a Gaussian hidden Markov model (HMM). The design separates observation, inference, decision and risk. Daily OHLCV bars are validated and converted into 18 trailing technical features; the HMM, estimated with Baum–Welch, supplies filtered regime probabilities that condition rather than replace directional signals; and the directional layer combines a two-layer LSTM and an XGBoost classifier through accuracy-weighted voting over 1-, 5- and 10-day horizons, using volatility-scaled labels and FinBERT–VADER news sentiment. The implemented platform comprises a FastAPI backend with 30 REST endpoints and live WebSocket streaming and a Next.js terminal with fifteen configurable panels; the HMM regime engine is specified but not yet implemented. On a single chronological 80/20 split, LSTM models for 11 NIFTY 50 stocks reach a mean validation accuracy of 70.5% (59.7–87.0%) and XGBoost 57.4% on two stocks. However, in 32 of the 33 LSTM cases the reported precision, recall and F1 identify a model that predicts HOLD for every day, so its accuracy equals the majority-class rate and its balanced accuracy is at chance. The paper therefore specifies a leakage-safe walk-forward protocol with class-balanced metrics and reports a code audit showing that feature scalers were fitted before the chronological split.

Keywords: hidden Markov models, market regime detection, stock trend analysis, LSTM, XGBoost, financial sentiment analysis, walk-forward validation

I. INTRODUCTION

A stock market is not governed by one stable pattern. Prices react to earnings, policy, liquidity and investor behaviour, so a price series alternates between calm drifts, sustained trends and short bursts of high volatility; NIFTY 50 constituents move through broad rallies, sector rotations and sharp sell-offs, often within one year. A technical rule or classifier fitted to the whole history averages across these conditions, which is one reason why indicators that work in a trending market fail in a volatile one, and why reported accuracies often do not survive a change in conditions.

Hidden Markov models (HMMs) [1], [2] offer a principled way to describe this behaviour. An HMM assumes that an unobserved state evolves as a Markov chain and that each state emits observations, such as daily returns and volatility, from its own distribution. Since Hamilton's regime-switching model [3] and the analysis of state-dependent stock-market risk by Turner, Startz and Nelson [4], latent-state models have been used to separate bull-like, bear-like and stable periods [5]. Their appeal is interpretability: each fitted state can be described by its mean return, volatility and persistence, and the posterior probability of each state gives a running, uncertainty-aware summary of market conditions.

IntelliQuant was built as a market-intelligence terminal for NSE- and BSE-listed stocks. Rather than a single predictor, it treats trend analysis as a pipeline in which data quality, feature construction, prediction, signal filtering, risk control and evaluation are separate stages, and in which a model score is never itself a trade. The main contributions of this paper are:

- a layered architecture that separates market data, analytics, decision and presentation, with a Gaussian HMM regime engine as the core analytical model and a monitoring feedback loop;
- a directional signal layer, implemented in the platform, that combines an LSTM and an XGBoost classifier through accuracy-weighted voting over 1-, 5- and 10-day horizons, with volatility-scaled labels and FinBERT–VADER news sentiment;
- a working full-stack implementation: a FastAPI service, a Next.js terminal with fifteen configurable panels and per-stock model training for NIFTY 50 stocks;
- an evaluation of the directional layer on 11 stocks showing that its headline accuracy equals the majority-class rate, together with a leakage-safe walk-forward protocol and a code audit of the validation path.

Section II reviews related work, Section III describes the data and problem formulation, Section IV presents the method, Section V the implementation, Section VI the results, and Section VII concludes.

 This work is licensed under a Creative Commons Attribution 4.0 International License

II. RELATED WORK

A. Hidden Markov Models and Regime Switching

Baum and Petrie [2] established statistical inference for probabilistic functions of finite-state Markov chains, and the Baum–Welch expectation–maximisation procedure that followed remains the standard way to estimate HMM parameters. Rabiner's tutorial [1] organised HMM practice around three problems: evaluating the likelihood of an observation sequence with the forward algorithm, decoding the most likely state path with the Viterbi algorithm, and estimating the parameters. Hamilton [3] brought latent states into econometrics with a Markov-switching model whose parameters change with an unobserved business-cycle state, and Turner et al. [4] modelled state-dependent stock-market risk. Dias et al. [5] used an extended HMM to cluster financial time series across markets and over time, recovering regimes interpretable as bull-like, bear-like or stable. Textbook treatments are given by Bishop [6] and Tsay [7]; both stress that the number and meaning of states are modelling choices, not facts about the market.

B. Technical Indicators and Classical Forecasting

Technical indicators summarise price and volume behaviour with simple, transparent formulas. Wilder [8] introduced the relative strength index (RSI) and the average directional index (ADX), and Brock et al. [9] found that simple moving-average and trading-range rules had measurable predictive content on historical US data. ARIMA models [10] capture linear temporal structure but assume a single stationary process after differencing, which makes abrupt regime changes difficult to represent. Indicators remain useful as inputs and transparent baselines, but their thresholds are heuristic and they do not model transitions between conditions.

C. Machine Learning and Sentiment for Market Prediction

Random forests [11] and gradient-boosted trees such as XGBoost [12] have been widely applied to direction prediction, and Krauss et al. [13] compared deep networks, boosted trees and random forests for statistical arbitrage on the S&P 500. Long short-term memory (LSTM) networks [14] were shown by Fischer and Krauss [15] to outperform memory-free classifiers on the same task. These studies also show how sensitive results are to validation design: random train–test shuffling, preprocessing fitted on the full sample and the omission of transaction costs can all inflate reported performance [16]. On the text side, FinBERT [17] adapts BERT to financial sentiment and VADER [18] provides a fast lexicon-based alternative. IntelliQuant combines these strands, using the HMM for context and the supervised models and sentiment for directional evidence.

Table I contrasts IntelliQuant with representative existing work. Prior studies typically address one layer in isolation, either regime modelling, direction prediction or text sentiment, and evaluate it offline; IntelliQuant combines all three for Indian equities and exposes them in an interactive terminal.

TABLE I. COMPARISON OF INTELLIQUANT WITH EXISTING WORK

Work	Market / data	Regime model	Direction model	Sentiment	UI
Brock et al. [9]	DJIA index	No	MA and range-break rules	No	No
Hamilton [3]	US real GNP	Markov switching	No	No	No
Dias et al. [5]	Stock market indices	Extended HMM	No	No	No
Krauss et al. [13]	S&P 500 stocks	No	DNN, GBT, RF ensemble	No	No
Fischer and Krauss [15]	S&P 500 stocks	No	LSTM	No	No
Araci [17]	Financial news text	No	No	FinBERT	No
IntelliQuant (ours)	NSE/BSE equities	Gaussian HMM (proposed)	LSTM + XGBoost, 1/5/10 days	FinBERT + VADER	Web terminal, LLM

 This work is licensed under a Creative Commons Attribution 4.0 International License

III. DATASET AND PROBLEM FORMULATION

IntelliQuant works with daily open, high, low, close and volume (OHLCV) bars for NSE- and BSE-listed equities, retrieved through the yfinance interface to Yahoo Finance and cached locally in a SQLite database. Table II summarises the data configuration used by the platform's training pipeline. Each series is stored with its ticker, interval, adjustment policy and retrieval date, because vendor data can be revised or incomplete.

TABLE II. SUMMARY OF THE DATA CONFIGURATION

Component	Description
Data source	Yahoo Finance via yfinance; local SQLite cache
Universe	NIFTY 50 constituents; 11 with trained LSTM models
Frequency	Daily OHLCV bars, NSE/BSE sessions
History	2 years cached on demand; 5 years in Colab pipeline
Features	18 trailing technical indicators per trading day
Direction targets	SELL / HOLD / BUY at 1-, 5- and 10-day horizons
Latent states	K hidden regimes inferred by the HMM

Two related problems are formulated. In the regime problem, let $x_t \in \mathbb{R}^d$ be the feature vector observed on trading day t and $z_t \in \{1, \ldots, K\}$ an unobserved market state. The aim is to estimate the filtered probability $P(z_t = k \mid x_1, \ldots, x_t)$ of each state using only information available at the close of day t, and to describe each state by its fitted return, volatility and persistence. In the direction problem, the aim is to estimate, for horizons $h \in \{1, 5, 10\}$ days, the probability that the forward return falls below, within or above a volatility-scaled band, giving SELL, HOLD or BUY. The two outputs are kept separate: regime probabilities describe conditions, direction probabilities describe expected movement, and any trading decision is taken downstream under explicit risk constraints.

IV. PROPOSED METHOD

Fig. 1 shows the layered design. A data layer acquires and validates market data, an analytics layer computes features and infers regimes, a decision layer turns model outputs into candidate signals under risk constraints, and a presentation layer reports results with their uncertainty. A monitoring loop triggers re-validation and retraining when data or state distributions drift.

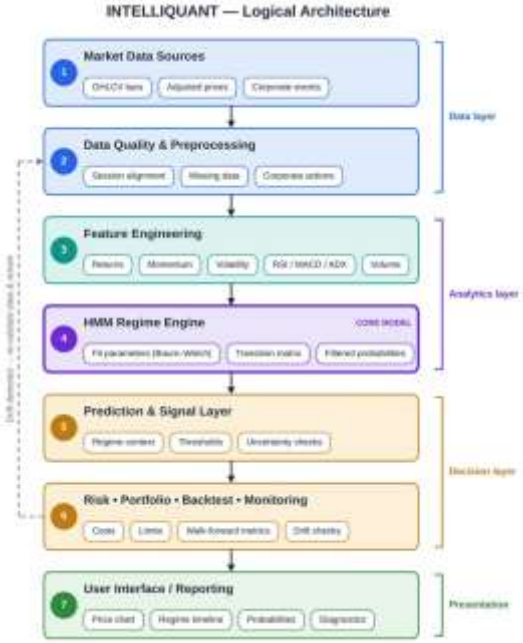


Fig. 1. Layered architecture of IntelliQuant, from market data to the user interface, with the HMM regime engine as the core model and a monitoring feedback loop.

 This work is licensed under a Creative Commons Attribution 4.0 International License

A. Data Validation and Feature Engineering

Raw bars are sorted chronologically, de-duplicated and checked for impossible values: the high must be at least the maximum of open and close, the low at most their minimum, and prices must be positive. Missing bars, stale prices and zero-volume days are flagged before any imputation, because forward-filling across a closure creates artificial zero returns and winsorising extreme returns removes the very stress periods a regime model should detect.

The platform computes 18 features from trailing windows only: RSI(14); MACD(12, 26) with its 9-day signal line and histogram; Bollinger bands (20, 2σ) and their relative width as a volatility measure; 20- and 50-day simple and 12- and 26-day exponential moving averages; 1- and 5-day price change; volume relative to its 20-day mean; average true range (ATR); ADX [8]; and the close relative to a 20-day volume-weighted average price. Rows whose features are undefined because the lookback is incomplete are dropped rather than back-filled. For the HMM, a compact subset of the log return $r_t = \ln(P_t / P_{t-1})$, rolling volatility and momentum is used, since a long, correlated feature vector makes Gaussian emission covariances unstable.

B. Gaussian Hidden Markov Model

The HMM is parameterised by $\lambda = (\pi, A, \theta)$, where $\pi_i = P(z_1 = i)$ is the initial distribution, $A_{ij} = P(z_{t+1} = j \mid z_t = i)$ is the transition matrix, and $\theta = \{\mu_k, \Sigma_k\}$ are the parameters of state-specific Gaussian emissions:

$$x_t \mid z_t = k \sim \mathcal{N}(\mu_k, \Sigma_k) \tag{1}$$

Filtered regime probabilities are obtained with the forward recursion, which uses observations up to day t only:

$$\alpha_t(j) = b_j(x_t) \sum_i \alpha_{t-1}(i) A_{ij} \tag{2}$$

where $b_j(x_t)$ is the Gaussian density of state j, and $P(z_t = j \mid x_{1:t}) = \alpha_t(j) / \sum_k \alpha_t(k)$. The diagonal of A determines persistence: the expected duration of state k is $1/(1 - A_{kk})$ trading days, so a model whose states switch almost every day is over-fitting rather than describing meaningful regimes.

C. Parameter Estimation, Decoding and State Selection

Parameters are estimated with Baum–Welch [1], [2]. The E-step uses the forward–backward algorithm to compute the posterior probability of each state and transition, and the M-step re-estimates π, A, μ_k and Σ_k from those responsibilities, iterating until the log-likelihood improvement falls below a tolerance or 300 iterations are reached. The Viterbi algorithm gives the most likely historical state path for charts, but online use relies on the filtered probabilities of (2), because smoothed probabilities use future observations. The number of states is chosen from $K \in \{2, 3, 4\}$ using the Bayesian information criterion:

$$\mathrm{BIC} = -2 \ln \hat{L} + p \ln T \tag{3}$$

where $\hat{L}$ is the maximised likelihood, p the number of free parameters and T the number of observations, together with state occupancy, parameter stability across rolling windows and out-of-sample usefulness. Labels such as calm or stress-like are assigned only after inspecting fitted means and volatilities.

D. Direction Labels and Signal Ensemble

For horizon h, the forward return $R_t^{(h)} = P_{t+h} / P_t - 1$ is compared with a volatility-scaled threshold:

$$\tau_t^{(h)} = \mathrm{clip}(\sqrt{h} \cdot \sigma_t, 0.005, 0.03\sqrt{h}) \tag{4}$$

where σ_t is the 20-day standard deviation of daily returns. The label is BUY if $R > \tau$, SELL if $R < -\tau$ and HOLD otherwise, so a 1% move counts as a signal for a quiet stock but as noise for a volatile one. Two classifiers are trained per stock and horizon. The LSTM receives 60-day sequences of the 18 scaled features through two LSTM layers of 64 and 32 units, a 32-unit dense layer and a three-way softmax, with dropout 0.2, the Adam optimiser and early stopping on validation loss; an XGBoost classifier receives the same features without the sequence dimension. Their class probabilities are combined with weights proportional to each model's latest recorded accuracy a:

$$p_{ens} = w_L\, p_{LSTM} + w_X\, p_{XGB}, \qquad w_L = a_L / (a_L + a_X) \tag{5}$$

with equal weights when no accuracy has been recorded. News headlines are scored by FinBERT and VADER, combined in a 0.6 : 0.4 ratio into a compound score, and the resulting sentiment signal is blended with the technical signal in a 0.3 : 0.7 ratio; a final score above 0.2 is read as BUY and below −0.2 as SELL.

 This work is licensed under a Creative Commons Attribution 4.0 International License

E. Regime-Aware Signal Gating

In the proposed design, the HMM conditions how directional signals are used rather than replacing them. If the filtered probability of the state with the highest fitted volatility exceeds a threshold chosen on validation data, BUY signals are downgraded to HOLD and position limits are tightened; when no state dominates because the posterior is diffuse, the interface shows a low-confidence flag instead of a signal. The effect of the regime model on every decision can therefore be traced, and its contribution can be measured directly by comparing gated and ungated signals under identical conditions.

F. Validation Protocol and Metrics

Financial observations are ordered in time, so random shuffling would let a model learn from future market conditions. Evaluation therefore uses the walk-forward protocol of Fig. 2: rolling windows of training, validation and test periods, with the most recent contiguous period reserved as a final hold-out that is evaluated once. Within each fold the scaler, HMM parameters and classifiers are fitted on the training window only, thresholds and K are chosen on the validation window, and the frozen configuration is scored on the test window. Table III lists the metrics.

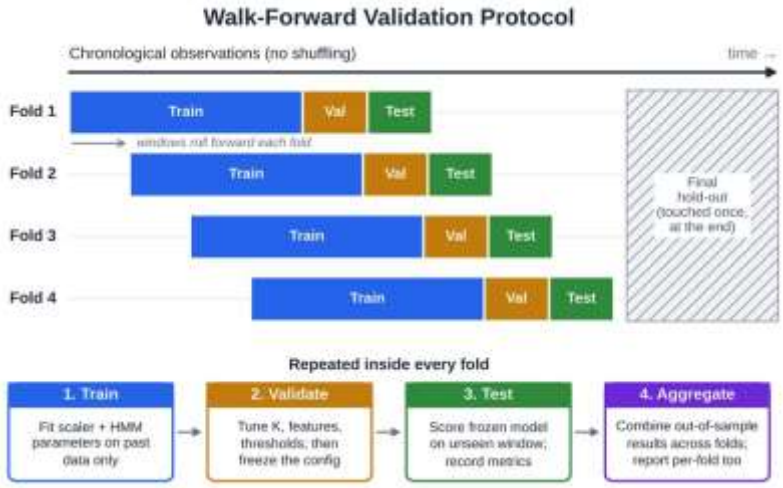


Fig. 2. Walk-forward validation: rolling train, validation and test windows, with a final hold-out period touched only once.

TABLE III. EVALUATION METRICS

Metric	Question answered	Caution
Log-likelihood, BIC	Fit relative to model complexity	In-sample fit is not forecasting skill
Regime duration, transitions	Are states persistent and interpretable?	State order can change between fits
Balanced accuracy, macro-F1	Is direction identified across classes?	HOLD dominates; report class balance
Sharpe ratio, max drawdown	Risk-adjusted outcome of a rule	Valid only net of costs, out of sample
Turnover, exposure	How often positions change	High turnover can erase gross edge

V. SYSTEM IMPLEMENTATION

IntelliQuant is implemented as a two-tier web application [21]. The backend is a Python FastAPI service of about 6,400 lines that exposes 30 REST endpoints and a WebSocket stream for live prices for market data, news, indicators, sentiment, training, prediction, backtesting and performance tracking. Data and performance records are kept in SQLite through SQLAlchemy. The models are built with PyTorch [19], TensorFlow, XGBoost, scikit-learn [20] and the Hugging Face FinBERT checkpoint, and a Groq-hosted large language model (Llama 3.1 8B for quick responses, Qwen3 32B for chat) explains signals in plain language without generating them. The deployed engine holds saved PyTorch LSTM weights for 11 stocks (44 weight files) and trains XGBoost on the fly; a Google Colab notebook that trains the three-horizon LSTMs for all 50 NIFTY 50 stocks on a T4 GPU has been written but not yet run.

 This work is licensed under a Creative Commons Attribution 4.0 International License

The complete source code is publicly available at https://github.com/ArindamJaiman/IntelliQuant.git [21].
The frontend is a Next.js 16 and React 19 application of about 11,000 lines of TypeScript, designed as a trading-terminal workspace (Fig. 3). Users arrange fifteen panels (chart, watchlist, news, AI assistant, heatmap, sentiment, comparison, risk, portfolio, paper trading and others) in one-, two- or three-column grids. Charts use TradingView Lightweight Charts with optional moving-average and support–resistance overlays, and each chart carries a badge with the current signal and its confidence. The news feed separates verified from unverified sources, and the risk panel reports volatility, beta, Sharpe ratio, maximum drawdown, 95% value at risk and an India VIX proxy.

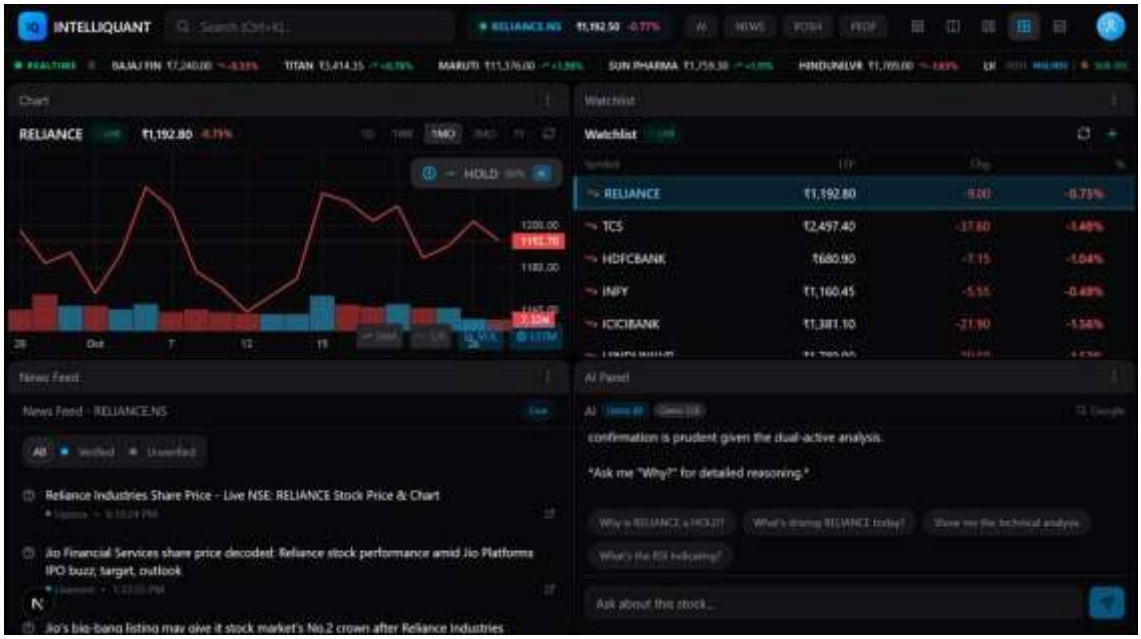


Fig. 3. IntelliQuant terminal for RELIANCE.NS: live price chart with volume, watchlist, verified and unverified news feed and AI panel; the badge shows the ensemble signal (HOLD, 66%).

VI. RESULTS AND DISCUSSION

The results come from the platform's current evaluation, run between 21 and 25 September 2026, in which each model was trained per stock and scored on a single chronological hold-out: the last 20% of the series, giving 76–79 validation days per LSTM model and about 100 test days per XGBoost model. This is the 80/20 split implemented in the code, not the walk-forward protocol of Section IV-F, and the HMM regime engine is not yet part of the pipeline, so the numbers describe the directional layer only.

A. Model Accuracy

Fig. 4 shows the validation accuracy of the LSTM models for the 11 stocks with saved weights. Mean accuracy is 71.9% at 1 day, 69.8% at 5 days and 69.9% at 10 days, ranging from 59.7% to 87.0%, with the highest value for ADANIENT.NS at 10 days. XGBoost metrics have been recorded for only two stocks (Table IV), with mean accuracy of 63.0%, 49.0% and 60.2% at the three horizons.

 This work is licensed under a Creative Commons Attribution 4.0 International License

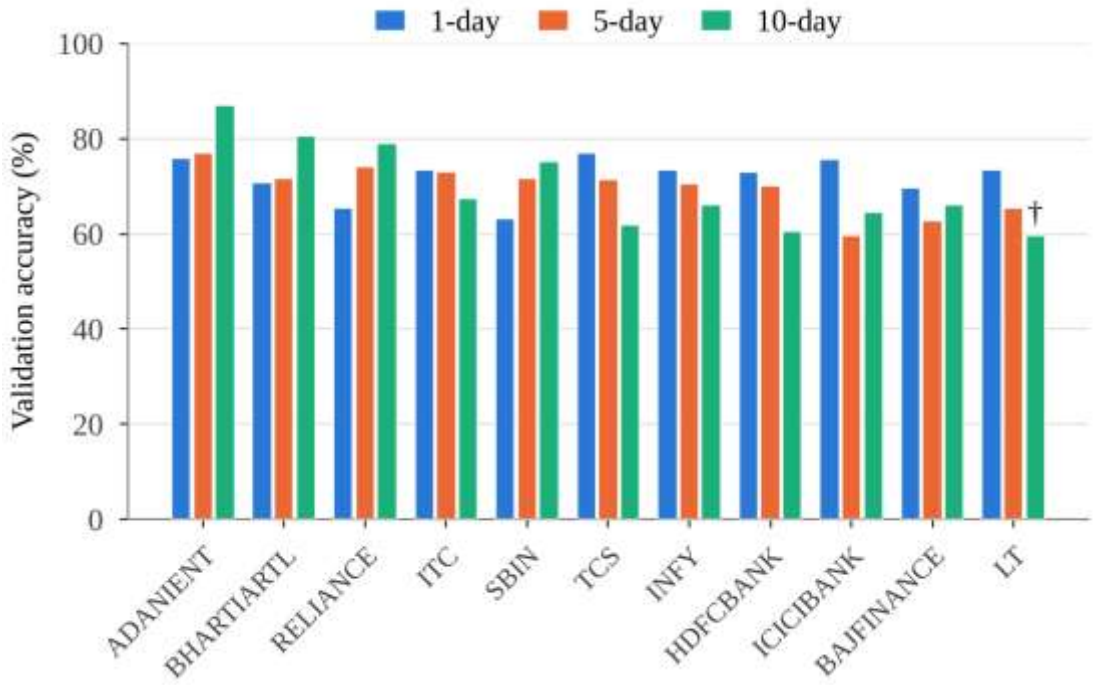


Fig. 4. LSTM validation accuracy for the 11 stocks with saved weights, by horizon. In every case except LT.NS at 10 days (†) the model predicts HOLD for all validation days, so the bar equals the HOLD share of the window.

B. What the Accuracy Measures

The headline numbers do not show predictive skill. For a classifier that outputs the same class for every sample, weighted precision equals the accuracy squared, weighted recall equals the accuracy, and weighted F1 equals $a \cdot 2a/(1 + a)$, where a is the accuracy. The reported LSTM metrics satisfy all three identities, to rounding, in 32 of the 33 stock–horizon cases; for ADANIENT.NS at 10 days, $0.870^2 = 0.757$ and $0.870 \times 1.740/1.870 = 0.810$, exactly the reported precision and F1. In these cases the LSTM predicts a single class, HOLD, the majority class under the volatility-scaled labels, for every validation day. Its accuracy is therefore the HOLD share of that window and its balanced accuracy is 33.3%, the chance level for three classes. Only LT.NS at 10 days produces more than one class. Differences between stocks and horizons, such as ADANIENT.NS rising from 76.0% at 1 day to 87.0% at 10 days, reflect how often each window's returns stayed inside the HOLD band, not differences in learning.

C. LSTM Versus XGBoost

XGBoost does produce mixed predictions: its precision does not follow the constant-class identity. On the two stocks where both models were evaluated (Table IV), the LSTM figure is higher in five of six cases, but because that figure is the majority-class rate, the comparison shows that XGBoost's mixed predictions are less accurate than always predicting HOLD, not that the LSTM is the better forecaster. The two models were also scored on slightly different windows, and both used scalers fitted before the split, so neither number is an unbiased estimate.

TABLE IV. LSTM AND XGBOOST ON THE TWO JOINTLY EVALUATED STOCKS

Stock	Horizon	LSTM acc. (= HOLD share)	XGBoost acc.	XGBoost F1
TCS.NS	1 day	76.9%	60.0%	0.593
TCS.NS	5 days	71.4%	46.5%	0.497
TCS.NS	10 days	61.8%	57.1%	0.544
RELIANCE.NS	1 day	65.4%	66.0%	0.568
RELIANCE.NS	5 days	74.0%	51.5%	0.516
RELIANCE.NS	10 days	79.0%	63.3%	0.600

 This work is licensed under a Creative Commons Attribution 4.0 International License

Training was also unstable. An earlier TCS.NS LSTM run on 22 September 2026 scored 12.5%, 11.5% and 15.1% at the three horizons, far below the three-class chance level, which is consistent with a model that predicts a minority class almost everywhere; the retrain that replaced it collapsed to the majority class instead.

D. Implementation Status and Validation-Path Audit

Table V summarises component status. Reviewing the training code against the protocol of Section IV-F found two issues. First, both trainers hold out the last 20% of each series chronologically, with shuffling disabled, which is correct; but the feature scaler is fitted on the full series before that split, so it sees the validation period's value range, a small look-ahead leak. Second, the backtest endpoint evaluates a fixed RSI(14) rule (buy below 30, sell above 70) rather than the model ensemble and applies no costs, which is why the LICI.NS analysis panel reported zero trades over three months.

TABLE V. IMPLEMENTATION STATUS OF INTELLIQUANT COMPONENTS

Component	Status	Evidence
Data ingestion and SQLite cache	Implemented	Repository code
18-feature engine	Implemented	Repository code
LSTM models (11 stocks)	Evaluated	Fig. 4
XGBoost models (2 stocks)	Evaluated	Table IV
FinBERT–VADER sentiment fusion	Implemented	Repository code
Colab pipeline (50 stocks)	Written, not run	Notebook
Terminal interface (15 panels)	Implemented	Fig. 3
Gaussian HMM regime engine	Specified	Sec. IV
Walk-forward evaluation	Specified	Table III, Fig. 2
Net-of-cost strategy backtest	Pending	RSI rule only

E. Discussion

Three lessons emerge. First, a headline accuracy of about 70% on a three-class problem looked strong but was the base rate of the majority class; reporting balanced accuracy, macro-F1 and the class distribution, as Table III requires, exposes this immediately. Second, the collapse to HOLD is a class-imbalance problem: with volatility-scaled thresholds most days fall inside the band, and an unweighted cross-entropy loss is minimised by predicting that class, so class-weighted or focal loss and rebalanced thresholds are the direct remedies. Third, separating context from prediction still matters, but before an HMM layer can be judged, the directional signals it conditions must first beat the majority-class baseline under the walk-forward protocol.

The main limitations are that the HMM regime engine is not yet implemented; the reported accuracies come from a single 80/20 split with 76–79 validation days per model and a scaler fitted before the split, and do not exceed the majority-class baseline; XGBoost has been evaluated on only two stocks; Gaussian emissions may misrepresent heavy-tailed returns; free vendor data may contain gaps and survivorship bias; and the system is a decision-support and learning tool that must not be used for real-money trading.

VII. CONCLUSION AND FUTURE WORK

IntelliQuant is a working market-intelligence terminal for Indian equities that separates observation, inference, decision and risk, and produces multi-horizon direction probabilities from an accuracy-weighted LSTM–XGBoost ensemble fused with news sentiment. This paper has positioned a Gaussian HMM as the regime layer of that design, with filtered posteriors that condition signals instead of replacing them. The current evaluation shows why a rigorous protocol is needed: LSTM models for 11 stocks reach 59.7–87.0% validation accuracy, but in 32 of 33 cases they predict HOLD for every day, so the accuracy equals the majority-class rate. Future work will train with class-weighted loss and report balanced accuracy and macro-F1; fit all preprocessing within training windows; implement the HMM engine and select the number of states by BIC and stability; extend XGBoost to all stocks; run the walk-forward comparison of buy-and-hold, moving-average, ungated-ensemble and regime-gated strategies net of costs; and add a regime timeline with posterior probabilities to the interface.

 This work is licensed under a Creative Commons Attribution 4.0 International License

ACKNOWLEDGMENT

The authors thank Dr. P. Chitra, Professor and Head, Department of Computer Science and Engineering (Emerging Technologies), SRM Institute of Science and Technology, Vadapalani Campus, for her guidance and support.

REFERENCES

[1]. L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," Proc. IEEE, vol. 77, no. 2, pp. 257–286, 1989.
[2]. L. E. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state Markov chains," Ann. Math. Statist., vol. 37, no. 6, pp. 1554–1563, 1966.
[3]. J. D. Hamilton, "A new approach to the economic analysis of nonstationary time series and the business cycle," Econometrica, vol. 57, no. 2, pp. 357–384, 1989.
[4]. C. M. Turner, R. Startz, and C. R. Nelson, "A Markov model of heteroskedasticity, risk, and learning in the stock market," J. Financ. Econ., vol. 25, no. 1, pp. 3–22, 1989.
[5]. J. G. Dias, J. K. Vermunt, and S. Ramos, "Clustering financial time series: New insights from an extended hidden Markov model," Eur. J. Oper. Res., vol. 243, no. 3, pp. 852–864, 2015.
[6]. C. M. Bishop, Pattern Recognition and Machine Learning. New York, NY, USA: Springer, 2006.
[7]. R. S. Tsay, Analysis of Financial Time Series, 3rd ed. Hoboken, NJ, USA: Wiley, 2010.
[8]. J. W. Wilder Jr., New Concepts in Technical Trading Systems. Greensboro, NC, USA: Trend Research, 1978.
[9]. W. Brock, J. Lakonishok, and B. LeBaron, "Simple technical trading rules and the stochastic properties of stock returns," J. Finance, vol. 47, no. 5, pp. 1731–1764, 1992.
[10]. G. E. P. Box and G. M. Jenkins, Time Series Analysis: Forecasting and Control. San Francisco, CA, USA: Holden-Day, 1970.
[11]. L. Breiman, "Random forests," Mach. Learn., vol. 45, no. 1, pp. 5–32, 2001.
[12]. T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., 2016, pp. 785–794.
[13]. C. Krauss, X. A. Do, and N. Huck, "Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500," Eur. J. Oper. Res., vol. 259, no. 2, pp. 689–702, 2017.
[14]. S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Comput., vol. 9, no. 8, pp. 1735–1780, 1997.
[15]. T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," Eur. J. Oper. Res., vol. 270, no. 2, pp. 654–669, 2018.
[16]. M. López de Prado, Advances in Financial Machine Learning. Hoboken, NJ, USA: Wiley, 2018.
[17]. D. Araci, "FinBERT: Financial sentiment analysis with pre-trained language models," 2019, arXiv:1908.10063.
[18]. C. J. Hutto and E. Gilbert, "VADER: A parsimonious rule-based model for sentiment analysis of social media text," in Proc. Int. AAAI Conf. Web Social Media (ICWSM), vol. 8, 2014, pp. 216–225.
[19]. A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 32, 2019.
[20]. F. Pedregosa et al., "Scikit-learn: Machine learning in Python," J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011.
[21]. A. Jaiman, "IntelliQuant: AI-powered financial terminal for NSE/BSE," GitHub repository, 2026. [Online]. Available: https://github.com/ArindamJaiman/IntelliQuant.git

 This work is licensed under a Creative Commons Attribution 4.0 International License