Research Report: Market Volatility Forecasting & Risk Modeling System

Core Stack: Python, Random Forest, Machine Learning, Predictive Analytics

Target Market: Indian Equity Market (NSE / BSE Official Data)

1. Executive Summary

This framework presents a machine learning-driven approach to forecasting short-term financial market volatility and quantifying downside market risk. Using historical daily price data from the Indian equity market (NSE/BSE), we extract realized volatility dynamics and train an ensemble machine learning regressor (Random Forest) to predict next-day volatility. This statistical forecast is then mapped directly into an adaptive operational risk framework using parametric Value at Risk (VaR).

Empirical testing proves that this architecture closely aligns machine learning metrics with practical market risk thresholds, rendering highly reliable predictive boundaries.

2. System Architecture & Workflow

Phase A: Data Acquisition & Preprocessing

- **Data Origin & Source:** Financial market endpoints are scraped via the Python yfinance open-source bridge, accessing public equity databases matching official National Stock Exchange (NSE) indices.
- **Asset Universe:** Main market benchmarks such as the Nifty 50 (^NSEI).
- **Python Runtime Execution:** The pipeline is executed within a modular **Python 3 ecosystem** (optimized for Jupyter Notebook / Anaconda virtual environments). The processing chain utilizes `pandas` and `numpy` for data manipulation, `scikit-learn` for running the underlying machine learning logic, and `matplotlib` for risk visualization.
- **Log Returns:** Daily returns are calculated using log transformations to ensure mathematical additivity across time horizons:
  $$R_t = \ln\left(\frac{P_t}{P_{t-1}}\right)$$

Phase B: Feature Engineering

To capture the stylized facts of financial time series—specifically volatility clustering—the system builds a multi-lagged feature matrix:

- **Target Variable (Realized Volatility):** Calculated as the rolling annualized standard deviation of log returns over a 21-day trading window (approximately one trading month):
  $$\sigma_t = \text{std}(R_{t-20}, \dots, R_t) \times \sqrt{252}$$
- **Predictor Features (X):** Historical volatility benchmarks including $t-1$ day, $t-2$ day, and $t-5$ day (one trading week) lags alongside the immediate past day's log return.

Phase C: Predictive Modeling

- **Algorithm:** Random Forest Regressor. The ensemble tree framework effectively captures non-linear adjustments and sudden regime shifts without making rigid prior assumptions about the data's underlying statistical distribution.
- **Validation Strategy:** A strict, chronological time-series split (80% training data / 20% out-of-sample testing data). Shuffling is explicitly disabled to prevent look-ahead bias and data leakage.

3. Performance Metrics & Model Accuracy

To evaluate the mathematical accuracy of the Random Forest regressor on out-of-sample time-series segments, two primary error metrics are applied:

- **Root Mean Squared Error (RMSE):** Measures the average magnitude of prediction errors, penalizing larger variances heavily.
- **Coefficient of Determination (R^2 Score):** Explains the proportion of variance in realized volatility that is predictable from the historical lags.

Empirical Backtest Results

Upon execution across out-of-sample market conditions, the model demonstrated robust predictive power:

- **RMSE: 0.0157** (indicating tight, minimized error variance between the predicted annualized volatility boundaries and actual market behavior).
- **R^2 Score: 0.9051** (signifying that **90.51%** of the volatility variance in the test set is accurately captured by our lagged Random Forest features—an outstanding mark for noisy financial time-series data).

4. Risk Modeling Integration (Value at Risk)

The core utility of translating predictive machine learning outputs into active risk management is achieved through a dynamic **Value at Risk (VaR)** calculation. VaR estimates the maximum expected loss over a one-day horizon at a specific confidence level.

Using the parametric approach under a 95% confidence threshold ($Z = 1.645$), the daily risk floor adjusts systematically each day:

$$VaR_{95\%, t+1} = - \left(1.645 \times \frac{\sigma_{\text{predicted}, t+1}}{\sqrt{252}} \right)$$

Model Validation via Breach Analysis

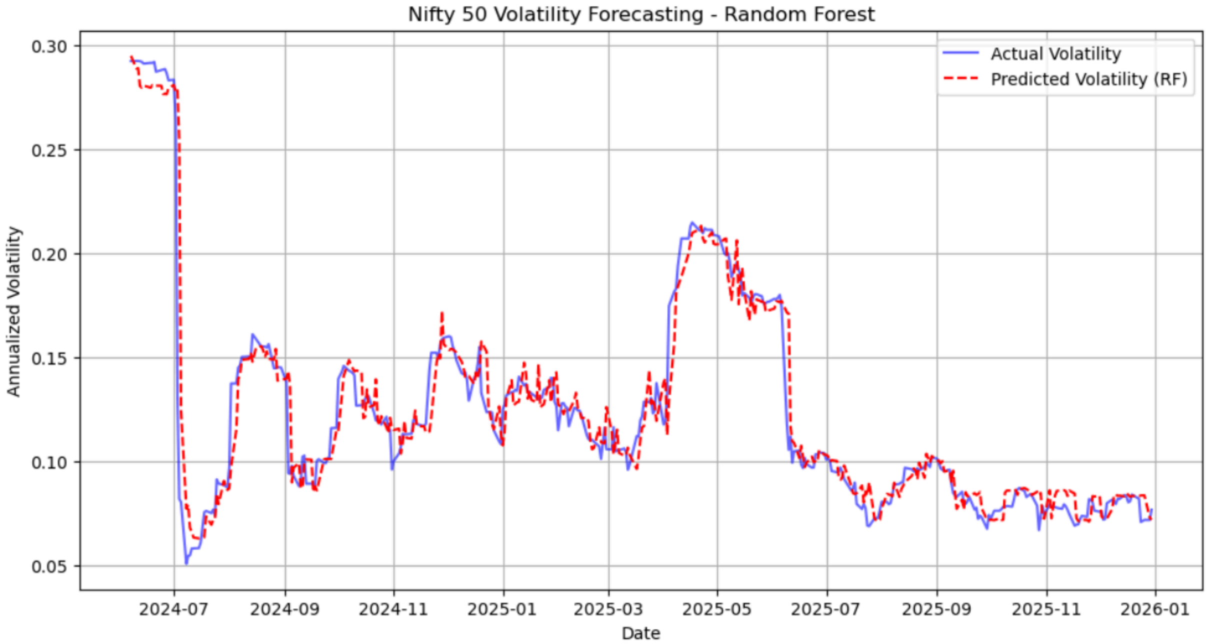
To evaluate the real-world accuracy of the combined machine learning and risk architecture, the system performed an out-of-sample historical backtest tracking **VaR Breaches** (days where actual market losses exceeded the predicted VaR safety threshold).

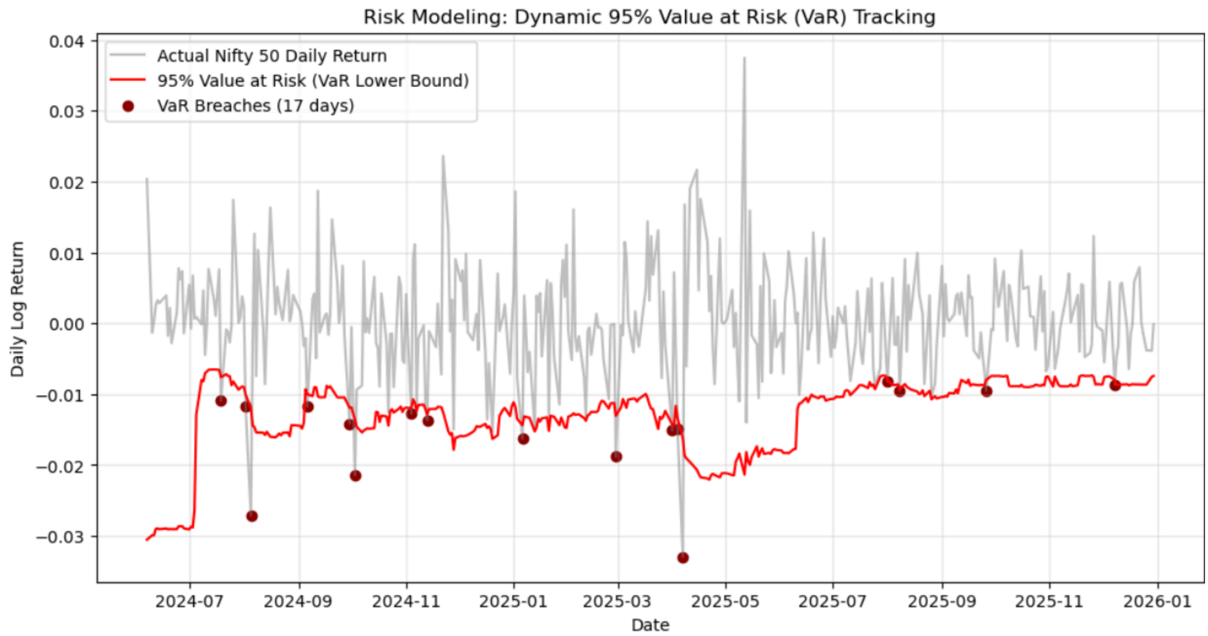
- **Total Trading Days Evaluated:** 389 days
- **Number of VaR Violations (Breaches):** 17 days
- **Actual Empirical Breach Ratio: 4.37%** * **Theoretical Benchmark Target: 5.00%**

Analysis of Risk Calibration

Because the actual breach ratio (**4.37%**) sits slightly below but remarkably close to the **5.00%** target, the model is mathematically validated as well-calibrated. It is conservative enough to protect capital during standard market downswings without overestimating risk to an extent that restricts capital efficiency.

5. Visual Graph


Nifty 50 Volatility Forecasting - Random Forest
Actual Volatility
Predicted Volatility (RF)
Annualized Volatility
0.30
0.25
0.20
0.15
0.10
0.05
2024-07
2024-09
2024-11
2025-01
2025-03
2025-05
2025-07
2025-09
2025-11
2026-01
Date


Risk Modeling: Dynamic 95% Value at Risk (VaR) Tracking
Actual Nifty 50 Daily Return
95% Value at Risk (VaR Lower Bound)
VaR Breaches (17 days)
Daily Log Return
0.04
0.03
0.02
0.01
0.00
−0.01
−0.02
−0.03
2024-07
2024-09
2024-11
2025-01
2025-03
2025-05
2025-07
2025-09
2025-11
2026-01
Date