

Implementation and Evaluation of Stock Sense: An AI-Driven Stock Market Analysis and Advisory Platform

Asst. Prof. Nitisha Rajgure , Atharva Jawalkar¹, Sarthak Kapse², Adwait Dhumal³

Asst. Prof Nitisha Rajgure , Computer Department ,ZCOER ,Pune

¹Atharva Jawalkar Computer Department ,ZCOER ,Pune

²Sarthak Kapse Computer Department ,ZCOER ,Pune

³Adwait Dhumal Computer Department ,ZCOER ,Pune

Abstract - This paper presents the implementation and evaluation of Stock Sense, a full-stack web application that integrates machine learning forecasting models with a locally hosted large language model to provide intelligent stock market analysis, sentiment-driven advisory, and portfolio management capabilities. The system employs a decoupled architecture comprising a Next.js 15 frontend, a Django REST Framework backend, and a hybrid database layer combining SQLite for relational transaction data with MongoDB for document-oriented financial data. Stock price forecasting is achieved through an ensemble of models—ARIMA, Exponential Moving Average (EMA), XGBoost Regression, and Long Short-Term Memory (LSTM) networks—while financial news sentiment classification utilizes a Bidirectional LSTM neural network and an XGBoost classifier. Natural language interaction is powered by the DeepSeek R1:1.5b model served locally through Ollama, enabling contextual financial advisory without external API dependency. Experimental evaluation on 163 processed news articles yields an XGBoost classification test accuracy of 87.88% for news nature prediction, while stock price forecasting across 30 NIFTY stocks demonstrates model-specific performance characteristics, including a negative R^2 score for short-horizon XGBoost price regression.

Key Words: Stock Market Prediction, Machine Learning, Sentiment Analysis, LSTM, XGBoost, Large Language Model, DeepSeek, Django, Next.js, MongoDB.

1.INTRODUCTION

The intersection of artificial intelligence and financial technology has produced a growing body of systems aimed at democratizing access to market analytics [1]. Stock market prediction remains challenging due to the stochastic nature of equity prices and macroeconomic factors [2]. Contemporary approaches combine time-series forecasting with natural language processing (NLP) of financial news to produce robust predictive signals [3]. Prior systems that predict stock price direction via deep learning (e.g., LSTMs [4]) or public sentiment analysis (e.g., Twitter mood [7], financial lexicons [5], or pre-trained transformers like FinBERT [6]) typically operate in isolation or rely on costly cloud-based large language model (LLM) APIs.

Stock Sense differentiates itself by implementing a complete end-to-end, privacy-preserving system that integrates multiple forecasting models, neural sentiment analysis, and a locally

hosted LLM within a production-grade web application. Operating the DeepSeek R1:1.5b model locally via Ollama eliminates external API dependencies and ensures data privacy.

2. Body of Paper

The body of the paper consists of numbered sections that present the main findings. These sections should be organized to best present the material.

It is often important to refer back (or forward) to specific sections. Such references are made by indicating the section number, for example, “In Sec. 2 we showed...” or “Section 2.1 contained a description...” If the word Section, Reference, Equation, or Figure starts a sentence, it is spelled out. When occurring in the middle of a sentence, these words are abbreviated Sec., Ref., Eq., and Fig.

At the first occurrence of an acronym, spell it out followed by the acronym in parentheses, e.g., charge-coupled diode (CCD).

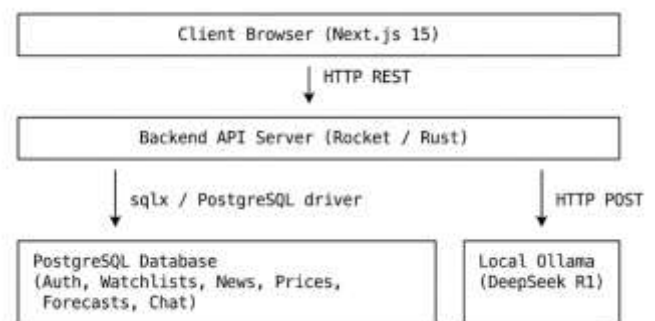


Fig 1: Architectural diagram of the StockSense system.

Table -1: Technology Stack Summary

Layer	Technology	Version	Purpose
Frontend	Next.js / React	15.5.4 / 19.x	App Router, Server-side rendering, API routes
UI styling	Tailwind CSS / shadcn	3.4.17	Responsive layouts and accessible primitive UI elements

Layer	Technology	Version	Purpose	Collection	Key Fields	Purpose
Charting	Recharts	Latest	SVG-based responsive data visualization	forecasts	ticker, arima, ema, xgboost, date	prices for 30 stocks
Backend	Django / DRF	5.0.2 / 3.14	RESTAPI, authentication, ORM business logic	transactions	userId, symbol, type (buy/sell), shares, price	Consolidated model predictions
Auth	simplejwt	5.3.0	JWT authentication with 50-minute access token lifetime	user_alerts	user_id, stock, message, is_read, created_at	User ledger records
Database	SQLite / MongoDB	3.x / 4.x	Relational (auth, profiles) and document-based data layers	chat_conversations	userId, title, messages[]	Sentiment-triggered security alerts
ML / DL	scikit-learn / Keras	1.3.2 / 2.15	XGBoost training, preprocessing, BiLSTM networks			Persistent user-AI chat threads
Time Series	statsmodels	Latest	ARIMA model fitting and forecasting			
LLM Server	Ollama (DeepSeek R1)	1.5b model	Local natural language inference and advisory chat			
Data Scraping	BeautifulSoup4 / yfinance	4.12.2 / Latest	Automated retrieval of financial news and stock prices			

3.2 Frontend and Backend API Integration

The Next.js frontend interacts with the Django backend using API routes under `app/api/` as a Backend-for-Frontend (BFF) proxy, which attaches JWT tokens to outgoing requests. Authentication uses token rotation: a custom Next.js hook automatically refreshes the access token every 10 minutes. The Django backend exposes nine REST endpoints, shown in Table 3.

Table 3: Core REST API Endpoints

Endpoint	Method	Auth	Purpose
<code>/api/register/</code>	POST	None	Creates Django user, profile, and initial balance
<code>/api/login/</code>	POST	None	Authenticates user and returns JWT tokens
<code>/api/user/</code>	POST	None	Retrieves user details from JWT token claims
<code>/model/</code>	POST	JWT	Interacts with the local LLM & triggers prediction queries
<code>/api/user/portfolio/</code>	GET/POST	JWT	Reconstructs and retrieves portfolio from transaction logs
<code>/api/user/watchlist/</code>	GET/POST	JWT	Manages watchlisted stocks with duplicate detection
<code>/api/user/balance/</code>	GET/PUT	JWT	Retrieves and updates user wallet balance
<code>/api/user/alerts/</code>	GET/POST	JWT	Fetches unread alerts and marks them as read

3.3 Automated Data pipeline

3. Subsystem Implementation & Database Design

3.1 Relational and Document Databases

SQLite (via Django ORM) handles data requiring transaction guarantees (UserProfile with wallet balance, Portfolio asset mapping, and Watchlist symbols). MongoDB (trada_db) manages high-volume financial document schemas, detailed in Table 2.

Table 2: MongoDB Collections

Collection	Key Fields	Purpose
raw_news	title, content, source, date, url	Raw scraped news articles
processed_news	All raw fields + Sentiment Score, Volatility, Nature	Enriched news with sentiment predictions
stock_prices	symbol, current_price, volume, timestamp	Daily historical

Automated jobs are orchestrated in Django using django-apscheduler. The scheduler coordinates three primary pipeline tasks:

1. **News Scraper (scrape_news):** Scrapes articles from *Business Standard* and *Economic Times* every 15 minutes. It cleans HTML content using regex filters and ensures deduplication via database queries.
2. **Price Scraper (scrape_stock_prices):** Fetches 60-day daily historical price series from Yahoo Finance for 30 major NIFTY stocks every 10 minutes.
3. **Portfolio Alert Analyzer (analyze_portfolio_alerts):** Runs every 2 hours, scanning recent news sentiment for user holdings and generating alerts in user_alerts if critical negative shifts or high volatility are predicted.

4. AI and Machine Learning Pipelines

StockSense implements a dual NLP and quantitative forecasting pipeline, combining a local LLM with predictive classifiers and regression models.

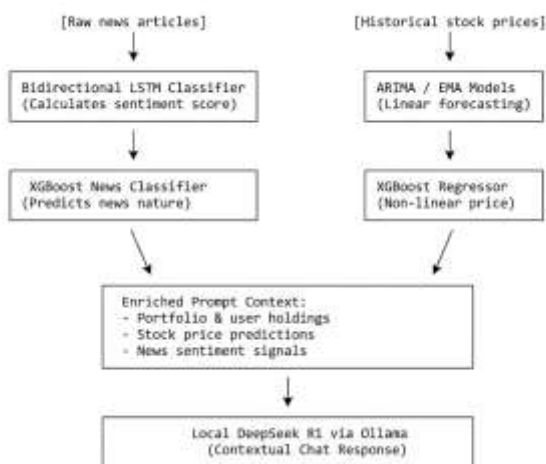


Fig 2: Flow diagram of the prediction and advisory pipelines.

4.1 Sentiment Analysis and News Classification

News sentiment is analyzed through a two-stage pipeline: - **Bidirectional LSTM Classifier:** This model processes clean tokenized text (lemmatized via NLTK) to output a sentiment label (Positive, Negative, Neutral) and a confidence score. The model architecture comprises a 5,000-word embedding layer, two Bidirectional LSTM layers (64 and 32 units) with dropout (0.3), and a Softmax output layer. - **XGBoost Classifier:** Enriches the sentiment label by predicting the overall “Nature of News.” It is trained on four features: the BiLSTM sentiment score, the LLM-derived relevance score (1–10), the impact level, and the volatility indicator.

4.2 Stock Price Forecasting Ensemble

For each of the 30 NIFTY stocks, StockSense maintains a predictive ensemble: 1. **ARIMA(5,1,0):** Captures stationary linear trend patterns in daily prices. 2. **EMA (20-period):** Computes a smoothed moving average, emphasizing recent price changes. 3. **XGBoost Regressor:** Models non-linear short-term price variations. It uses a sliding window of 10 daily closing prices as input features to predict the next close.

4.3 AI Chat and Context Integration

Natural language financial advisory is handled by a local DeepSeek R1:1.5b model. When a user interacts with the chat interface (AiChat.tsx), the request flows to the backend /model/ endpoint, where:

1. **Query Classification:** DeepSeek R1 classifies the user’s intent into five categories (GENERAL_INVESTMENT, PORTFOLIO_SPECIFIC, MARKET_ANALYSIS, FINANCIAL_EDUCATION, PRICE_QUERY).
2. **Context Enrichment:** The backend dynamically fetches user portfolio holdings, recent transaction history, sentiment averages, price forecasts, and current stock prices to construct an enriched prompt.
3. **Response & Visualization:** The LLM output is parsed. If Python matplotlib code is generated, the backend executes it within a local script, saves the visualization to /media/, and returns it alongside the text response. Reasoning traces (<think>...</think>) are filtered out.

5. Experimental Results

The machine learning models were trained and evaluated on data gathered from the pipeline (timestamp: 2025-11-13).

5.1 News Nature Classification Performance

The XGBoost classifier was trained and evaluated on 163 manually annotated and AI-enriched news articles. - 3-Fold Cross-Validation Accuracy: 79.84% (standard deviation of $\pm 11.84\%$) - Hold-out Test Accuracy (80/20 split): 87.88% The classifier demonstrates robust generalization for nature classification despite the modest dataset size.

5.2 Sentiment Trend Models

To evaluate trend forecasting on sentiment sequences, both LSTM and ARIMA models were fitted on 153 consecutive daily news sentiment scores (Table 4).

Table 4: Sentiment Trend Model Comparison

Model	Mean Square Error (MSE)	Root Squared (RMSE)	Mean Error	Mean Absolute Error (MAE)
LSTM	1.0329	1.0163		0.4979
ARIMA (5,1,0)	1.1698	1.0816		0.4839

The high RMSE values relative to the $[-1, 1]$ sentiment scale suggest that predicting raw sentiment trajectories contains high noise, though the LSTM neural network marginally outperforms the linear ARIMA model.

5.3 Stock Price Forecasting Performance

The XGBoost regression model was evaluated across the 30 NIFTY stocks using 10-day sliding price sequences. The averaged metrics are presented in Table 5.

Table 5: XGBoost Price Forecasting Performance

Metric	Averaged Value
Mean Absolute Error (MAE)	₹17.56
Root Mean Squared Error (RMSE)	₹23.33
R ² Score	-0.8885
Number of Stocks	30

The negative R² score (-0.8885) indicates that the regression model performs worse than a simple historical mean predictor. This result is analyzed in Section 6.

6. Discussion, Future Work & Conclusion

6.1 Analysis of Forecasting Limitations

The negative R^2 score highlights the limitations of using a restricted dataset (60-day price history) and a basic autoregressive feature set (only closing prices). Stock price trends are highly non-stationary, and modeling them purely as a sliding window of historical prices fails to capture underlying drivers. To address this, future iterations will expand feature engineering to include volume indicators, technical markers (RSI, MACD, Bollinger Bands), and cross-stock sector correlations.

6.2 Architectural Strengths and Weaknesses

- **Strengths:** Successfully implements a modular, containerized, local AI/ML deployment. Serving DeepSeek R1 locally through Ollama ensures 100% data privacy and zero API query costs. Proactive, sentiment-triggered alerting acts as a practical risk mitigation tool for portfolios.
- **Limitations:** The synchronous execution of matplotlib code generated by the LLM poses security risks (potential arbitrary code execution) and requires containerized sandboxing. SQLite's write-locking constraints restrict simultaneous user database writes. Storing JWTs in local storage remains vulnerable to XSS.

6.3 Future Work

Key technical milestones for the next iteration of StockSense include:

1. **Transformer Upgrades:** Replacing the BiLSTM sentiment text analyzer with a fine-tuned FinBERT transformer model to capture nuanced financial semantics.
2. **Infrastructure Scaling:** Migrating SQLite to PostgreSQL to enable production-grade connection pooling, and serving ML models using TensorFlow Serving or ONNX Runtime to eliminate per-request model-loading latencies.
3. **Streaming & Security:** Integrating Django Channels for real-time portfolio updates via WebSockets, storing JWT tokens in HTTP-only secure cookies, and implementing Docker-sandboxed runtimes for dynamic chart generation scripts.

7. Conclusion

StockSense demonstrates the feasibility of combining multiple traditional forecasting models, custom neural network NLP classifiers, and locally hosted large language models to construct a comprehensive financial advisory platform. Although the quantitative price forecasting models highlight the classic difficulty of short-term market forecasting, the integrated sentiment-driven alerting, data scraping pipelines, and privacy-preserving local LLM advisory establish a robust architectural blueprint for modern intelligent trading dashboards.

REFERENCE

- [1]. Lohar, K., Pahwa, P., Asthana, N. and Gautam, D., 2024, February. Stock Market Analysis Using Time Series with ARIMA Model. In 2024 2nd International Conference on Computer, Communication and Control (IC4) (pp. 1-5). IEEE.
- [2]. Zhang, M., 2021, September. Design of simulated stock forecasting trading system based on time series. In 2021 IEEE 4th International Conference on Information Systems and Computer Aided Education (ICISCAE) (pp. 382-385). IEEE.
- [3]. Banerjee, P. and Nayak, R., 2024. Recommendations on Financial Models for Stock Price Prediction. SN Computer Science, 5(1), p.178.
- [4]. Dwivedi, S.A., Attry, A., Parekh, D. and Singla, K., 2021, February. Analysis and forecasting of Time- Series data using SARIMA, CNN and LSTM. In 2021 international conference on computing, communication, and intelligent systems (icccis) (pp. 131-136). IEEE.
- [5]. Rahman, M.H., Nahid, S.I., Al Fahad, I.H., Nahid, F.M. and Khan, M.M., 2021, October. Price prediction using LSTM based machine learning models. In 2021 IEEE 12th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON) (pp. 0453- 0459). IEEE
- [6]. Kumar, P. and Mallieswari, R., 2022, February. Predicting stock market price movement using machine learning technique: Evidence from india. In 2022 Interdisciplinary Research in Technology and Management (IRTM) (pp. 1-7). IEEE.
- [7]. Jakhar, Y.K., Sharma, P. and Ahmed, B., 2024, July. Stock Price Prediction by Using Machine Learning Techniques: A Study of TCS Ltd. In 2024 2nd International Conference on Sustainable Computing and Smart Systems (ICSCSS) (pp. 1256-1260). IEEE.
- [8]. Manasa, N., Praveenraj, D.W. and SR, L., 2023, December. Predictive Analytics for Stock Market Trends using Machine Learning. In 2023 4th International Conference on Computation, Automation and Knowledge Management (ICCAKM) (pp. 1-8). IEEE.
- [9]. Varaprasad, B.N., Kanth, C.K., Jeevan, G. and Chakravarti, Y.K., 2022, March. Stock Price Prediction using Machine Learning. In 2022 International Conference on Electronics and Renewable Systems (ICEARS) (pp. 1309-1313). IEEE.
- [10]. Vakayil, S., Juliet, D.S. and Vakayil, S., 2024, April. RAG-Based LLM Chatbot Using Deepseek-2. In 2024 7th International Conference on Devices, Circuits and Systems (ICDCS) (pp. 1-5). IEEE.
- [11]. Kumari, S., Mahajan, J., Jain, P., Panikulangara, L., Kulkarni, S. and Saxena, A., 2024, March. Charting the Future of Fintech: Unveiling Finoracle through an In-depth Comparison of DEEPSEEK 2, FLAN, and GPT-3.5. In 2024 International Conference on Trends in Quantum Computing and Emerging Business Technologies (pp. 1-5). IEEE.
- [12]. Selvi, A., Mounika, V., Rubika, V. and Uvadhananee, B., 2024, January. COLLEGEBOT: Virtual Assistant System for Enquiry Using Natural Language Processing. In 2024 2nd International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT) (pp. 1407-1414). IEEE.
- [13]. Lin, D., Wen, Y., Wang, W. and Su, Y., 2024. Enhanced Sentiment Intensity Regression through LoRA Fine-Tuning on Deepseek