UDC 004.8:336.761

DESIGN AND IMPLEMENTATION OF AN INTELLIGENT WEB-BASED SYSTEM FOR STOCK MARKET FORECASTING USING ARTIFICIAL INTELLIGENCE

Erkinov Farkhodjon Gulomjon Ugli

Samarkand State University named after Sharof Rashidov

Abstract. The practical deployment of artificial intelligence in financial forecasting requires an integrated environment for data acquisition, preprocessing, model training, evaluation, storage, visualization, and repeated forecasting. This article presents the design and implementation of an intelligent web-based stock market forecasting system developed as a modular Python/Streamlit application. The architecture consists of user interface, application service, machine-learning pipeline, storage, and reporting/visualization layers. The platform supports online and local OHLCV data loading, validation, feature engineering, chronological train-validation-test splitting, train-only scaling, Adaptive CatBoost training, metric computation, model registration, and forecast generation with previously trained models. A model registry stores configuration files, scalers, metrics, predictions, figures, and experiment metadata, improving reproducibility and model reuse. The prototype demonstrates how an AI forecasting methodology can be transformed from an experimental script into a practical research and educational environment.

Keywords: *artificial intelligence, stock market forecasting, web application, financial time series, CatBoost, Streamlit, model registry, reproducibility.*

Annotatsiya. Moliyaviy prognozlashda sun’iy intellektdan amaliy foydalanish ma’lumotlarni yuklash, qayta ishlash, modelni o‘qitish, baholash, saqlash, vizuallashtirish va takroriy prognozlashni yagona muhitda birlashtirishni talab qiladi. Ushbu maqolada fond bozori dinamikasini prognozlash uchun ishlab

chiqilgan intellektual veb-tizimning arxitekturasi va dasturiy amalga oshirilishi yoritilgan. Tizim Python va Streamlit asosida modulli tarzda ishlab chiqilgan bo‘lib, foydalanuvchi interfeysi, amaliy xizmatlar, mashinaviy o‘qitish konveyeri, saqlash hamda hisobot-vizualizatsiya qatlamlaridan tashkil topgan. Platforma OHLCV ma’lumotlarini onlayn va lokal yuklash, validatsiya, feature engineering, xronologik bo‘lish, faqat training ma’lumotlari asosida masshtablash, Adaptive CatBoost modelini o‘qitish, metrikalarni hisoblash, model registri va saqlangan model orqali prognoz yaratishni qo‘llab-quvvatlaydi.

Kalit so‘zlar*: sun’iy intellekt, fond bozori, veb-ilova, moliyaviy vaqt qatori, CatBoost, Streamlit, model registri, takrorlanuvchanlik.*

Аннотация. Практическое применение искусственного интеллекта для финансового прогнозирования требует единой программной среды, объединяющей загрузку данных, предварительную обработку, обучение модели, оценивание, хранение результатов, визуализацию и повторное прогнозирование. В статье представлена архитектура и программная реализация интеллектуальной веб-системы прогнозирования динамики фондового рынка. Система разработана как модульное приложение на Python/Streamlit и включает пользовательский интерфейс, сервисный слой, конвейер машинного обучения, слой хранения данных и модуль отчетности и визуализации. Платформа поддерживает загрузку OHLCV-данных, их проверку, формирование признаков, хронологическое разделение выборки, обучение Adaptive CatBoost, расчет метрик, регистрацию моделей и формирование прогнозов с использованием ранее обученных моделей.

Ключевые слова*: искусственный интеллект, фондовый рынок, веб-приложение, финансовые временные ряды, CatBoost, Streamlit, реестр моделей, воспроизводимость.*

1. Introduction

The rapid growth of digital financial data and the development of artificial intelligence have expanded the possibilities of automated stock market analysis.

Financial forecasting is nevertheless difficult because market series are noisy, nonlinear, non-stationary and sensitive to changing volatility regimes. Machine-learning and deep-learning methods can capture complex relationships, but the practical value of a forecasting method depends on more than predictive accuracy [1–4].

In many studies, a forecasting algorithm remains an isolated experimental script. Such an implementation can be difficult to reproduce, compare, maintain and reuse. A complete research environment must support the full workflow: data acquisition, validation, preprocessing, feature engineering, model training, performance evaluation, visualization, storage of artifacts and subsequent forecasting. The master's dissertation used as the basis of this article addresses this gap by implementing the Adaptive CatBoost forecasting methodology as a reusable web application.

The aim of this study is to design and implement an intelligent web-based system that converts an artificial-intelligence forecasting methodology into an integrated analytical environment. The system is intended for academic experimentation, educational demonstration and financial time-series analysis. Its software contribution is the integration of a leakage-free forecasting pipeline, experiment tracking and model reuse in a single application.

2. System Architecture and Design Principles

The application is implemented as a modular Python-based web system with a Streamlit user interface. The architecture follows the principle of separation of responsibilities and is organized into five layers: user interface, application services, machine-learning pipeline, storage, and reporting/visualization. This separation reduces coupling between interface logic and forecasting algorithms and simplifies future extension with additional models.

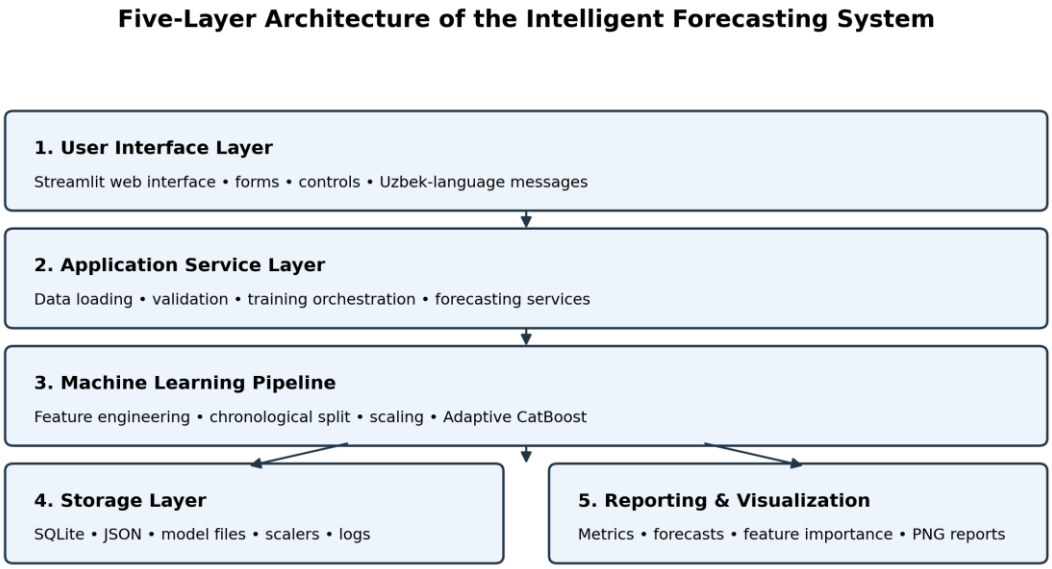


Figure 1. Five-layer architecture of the intelligent forecasting system.

Table 1. Functional responsibilities of the system layers.

Layer	Core functions	Main outputs
User interface	Forms, controls, notifications, pages	User requests and displayed results
Application services	Workflow coordination and validation	Prepared jobs and service responses
Machine-learning pipeline	Feature engineering, splitting, scaling, training	Predictions, metrics, feature importance
Storage	SQLite, JSON, model/scaler files, logs	Reusable experiment artifacts
Reporting & visualization	Tables, plots and forecast summaries	Human-readable analytical reports

The architecture ensures that trained models, datasets, metric tables, forecast results and visualizations can be stored and retrieved later. This is especially important when experiments are repeated across different assets such as AAPL, MSFT, GOOGL, SPY, NVDA and TSLA.

3. Functional Organization of the Web Application

The interface contains seven main pages. “Bosh sahifa” presents the system purpose and the selected forecasting methodology. “Dataset yuklash” supports online and local data acquisition. “Modelni o‘qitish” provides model configuration and training controls. “Baholash natijalari” displays numerical and graphical evaluation outputs. “Prognozlash” applies a saved model to recent or user-supplied data. “Saqlangan modellar” acts as the model registry, while “Loglar / run tarixi” preserves experiment history and execution records.

Table 2. Main pages of the application and their functions.

Page	Function
Bosh sahifa	General project, model and metric information
Dataset yuklash	Online retrieval or CSV/XLSX/XLS upload
Modelni o‘qitish	Training configuration and execution
Baholash natijalari	Metrics, charts and prediction tables
Prognozlash	Forecast generation using a saved model
Saqlangan modellar	Model registry and artifact management
Loglar / run tarixi	Experiment history and execution logs

4. Data Acquisition, Validation and Feature Engineering

The system supports two acquisition modes. In online mode, historical data are downloaded for a ticker and date range. In local mode, CSV, XLSX or XLS files can be uploaded. The application validates the presence of the OHLCV variables (Open, High, Low, Close and Volume), checks chronological consistency and reports missing fields before model training.

After basic validation, raw observations are transformed into a structured forecasting dataset. The feature engineering stage constructs return-based variables, lagged prices and returns, rolling statistics, technical indicators, volatility descriptors, momentum indicators, regime-sensitive variables and

calendar features. Thus, the model uses a richer representation of market state than raw prices alone [5–7].

A methodological requirement is prevention of data leakage. The full series is divided chronologically into training, validation and test subsets before scaling or model fitting. Scaling parameters are estimated only from the training subset and then reused for the validation, test and forecasting stages. This procedure preserves the natural temporal direction of the experiment.

$$D = D_train \cup D_validation \cup D_test, \quad D_train < D_validation < D_test \text{ in time}$$

5. Machine-Learning and Forecasting Pipeline

The predictive core of the application is the Adaptive CatBoost framework developed in the dissertation. It combines a CatBoostRegressor for future log-return magnitude estimation and a CatBoostClassifier for movement direction analysis. The system therefore separates two related forecasting questions: how large the future movement may be and in which direction it may occur.

$$r(t,h) = ln[C(t+h) / C(t)]$$

$$\hat{C}(t+h) = C(t) \cdot exp(\hat{r}(t,h))$$

The training page exposes the ticker, observation interval, forecasting horizon, data split ratios and model configuration. During training, the regression and classification branches are fitted using the same engineered features but different targets. The validation set is used for model monitoring and selection, while the test set remains untouched until the final evaluation stage.

6. Model Registry and Reusable Forecasting

A central practical component of the system is its model registry. After training, the application creates a unique experiment directory and saves the model, scaler, feature list, configuration, performance metrics, prediction results, feature-importance values, generated figures and execution metadata. These artifacts link every forecast to the exact model and preprocessing state that produced it.

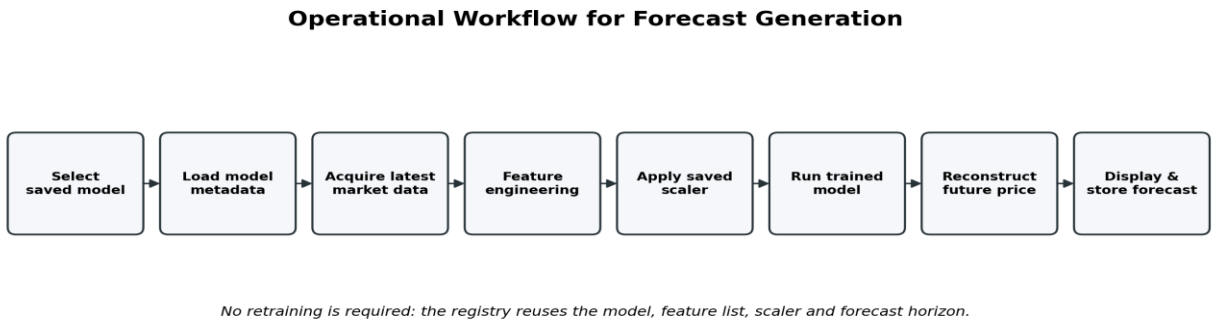


Figure 2. Operational workflow for forecasting with a previously trained model.

The forecasting page does not require the original training dataset to be selected again. The user chooses a saved model; the application reads its metadata, obtains the latest market data for the corresponding ticker, recreates the required features, applies the stored scaler and generates a new forecast. A separate mode accepts a local file for external testing. The registry therefore converts model training from a one-time experiment into a reusable forecasting service.

7. Evaluation, Visualization and Experimental Results

The evaluation module calculates MAE, RMSE, MAPE, R^2 and Directional Accuracy. These measures describe complementary properties of forecasting quality: absolute error, sensitivity to large deviations, relative percentage error, explained price variance, and correctness of the predicted movement direction. The application also generates actual-versus-predicted plots, error distributions, feature-importance charts and prediction tables. Metrics are stored in SQLite and JSON, while figures are saved as PNG artifacts.

Table 3. Forecasting performance on the GOOGL dataset.

Model	MAE	RMSE	MAPE, %	R^2	Directional Accuracy, %
Adaptive CatBoost	6.2940	8.4677	3.1877	0.9792	56.54

CNN-LSTM Hybrid	6.3831	8.7386	3.2493	0.9778	58.05
LSTM	6.6520	9.0734	3.3961	0.9761	59.73
Multivariate CNN	8.0840	11.5942	4.0310	0.9610	50.67
ARIMA	11.2634	14.7638	5.5447	0.9367	46.64

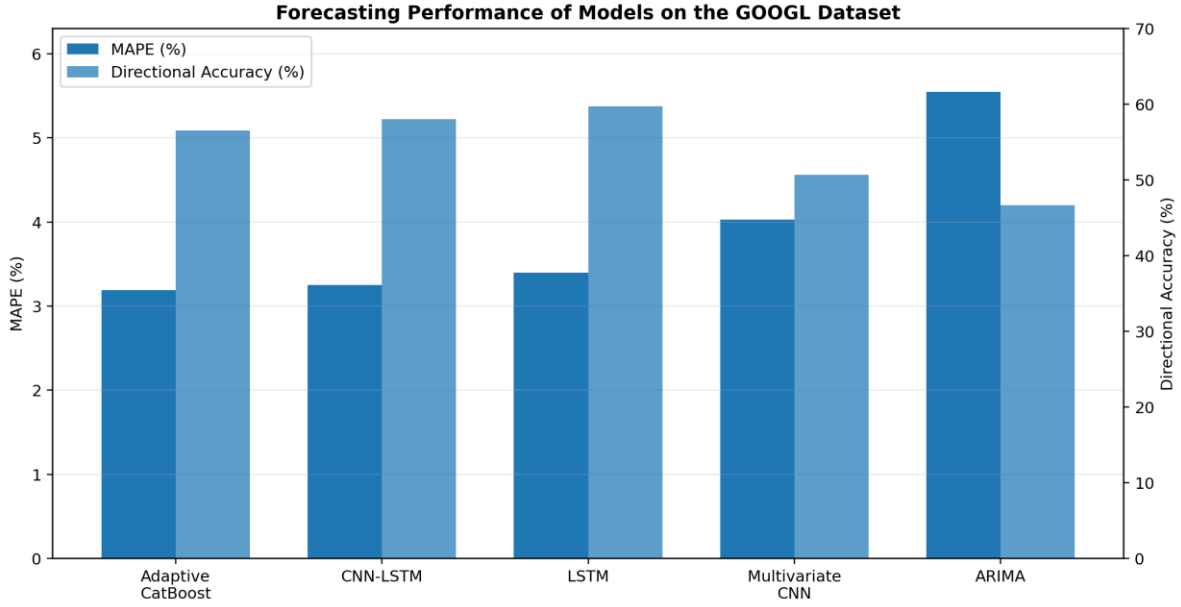


Figure 3. Comparison of MAPE and Directional Accuracy on the GOOGL dataset.

The integrated Adaptive CatBoost model achieved the lowest MAPE (3.1877%) and the highest R^2 (0.9792) in the principal GOOGL experiment. LSTM obtained the highest Directional Accuracy (59.73%). The result is important for software interpretation: price-level accuracy and directional forecasting are related but not identical objectives, so the interface reports both rather than presenting a single aggregate score.

8. Practical Significance, Limitations and Future Development

The developed platform provides a complete workflow for financial time-series experimentation. Its main practical advantages are flexible data acquisition, consistent preprocessing, leakage-free chronological training, automatic

evaluation, reusable model storage and separation of training from forecasting. The same pipeline can be applied to multiple instruments when the required OHLCV structure is available.

The current prototype also has limitations. It relies mainly on historical OHLCV and engineered technical features and does not yet fully integrate financial news, investor sentiment, macroeconomic indicators or company reports. It primarily provides point forecasts rather than probabilistic prediction intervals. Furthermore, it is an analytical forecasting tool rather than a complete trading system: transaction costs, portfolio optimization, position sizing, risk-adjusted backtesting and automated execution are outside its present scope.

Future development can include three-class direction labeling (up, neutral, down), sentiment analysis based on transformer models, macroeconomic and sector variables, and additional forecasting algorithms such as XGBoost, LightGBM, Temporal Convolutional Networks and Transformer-based time-series models. From the software-engineering perspective, the prototype can evolve toward a client-server architecture with user authentication, role-based access, scheduled data updates, background jobs, cloud deployment and automatic retraining.

Conclusion

This study presented the design and implementation of an intelligent web-based system for stock market forecasting using artificial intelligence. The developed platform transforms an AI forecasting methodology into an integrated environment that supports data acquisition, validation, feature engineering, model training, evaluation, visualization, experiment storage and future forecasting.

The five-layer architecture separates interface, service, machine-learning, storage and reporting responsibilities. The model registry is particularly important because it stores the trained model together with preprocessing parameters, configuration, metrics, predictions and figures, allowing experiments to be reproduced and previously trained models to be reused without retraining.

The prototype demonstrates that practical AI-based financial forecasting is both a modeling and a software-engineering problem. Reliable deployment requires consistent preprocessing, temporal validation, transparent evaluation, artifact management and mechanisms for operational reuse. The developed system provides a foundation for further academic research and for future extension toward a more scalable financial analytics platform.

References

[1] M. M. Kumbure, C. Lohrmann, P. Luukka, and J. Porras, "Machine learning techniques and data for stock market forecasting: A literature review," Expert Systems with Applications, vol. 197, 116659, 2022. doi: 10.1016/j.eswa.2022.116659.

[2] O. B. Sezer, M. U. Gudelek, and A. M. Ozbayoglu, "Financial time series forecasting with deep learning: A systematic literature review: 2005–2019," Applied Soft Computing, vol. 90, 106181, 2020. doi: 10.1016/j.asoc.2020.106181.

[3] G. Sonkavde et al., "Forecasting stock market prices using machine learning and deep learning models: A systematic review, performance analysis and discussion of implications," International Journal of Financial Studies, vol. 11, no. 3, 94, 2023. doi: 10.3390/ijfs11030094.

[4] M. Saberironaghi, J. Ren, and A. Saberironaghi, "Stock market prediction using machine learning and deep learning techniques: A review," AppliedMath, vol. 5, no. 3, 76, 2025. doi: 10.3390/appliedmath5030076.

[5] H. H. Htun, M. Biehl, and N. Petkov, "Survey of feature selection and extraction techniques for stock market prediction," Financial Innovation, vol. 9, 26, 2023. doi: 10.1186/s40854-022-00441-7.

[6] M. Nabipour, P. Nayyeri, H. Jabani, and A. Mosavi, "Deep learning for stock market prediction," Entropy, vol. 22, no. 8, 840, 2020. doi: 10.3390/e22080840.

[7] K. Sako, B. N. Mpinda, and P. C. Rodrigues, "Neural networks for financial time series forecasting," Entropy, vol. 24, no. 5, 657, 2022. doi: 10.3390/e24050657.

[8] S. Mehtab, J. Sen, and A. Dutta, “Stock price prediction using machine learning and LSTM-based deep learning models,” in Machine Learning and Metaheuristics Algorithms, and Applications, Springer, 2021, pp. 88–106.

[9] S. Mehtab and J. Sen, “Stock price prediction using convolutional neural network on a multivariate time series,” Proc. 3rd National Conference on Machine Learning and Artificial Intelligence, 2020.

[10] M. R. Sahib, H. Elkina, and T. Zaki, “From technical indicators to trading decisions: A deep learning model combining CNN and LSTM,” International Journal of Advanced Computer Science and Applications, vol. 15, no. 6, 2024.

[11] M. Patsiarikas, G. Papageorgiou, and C. Tjortjis, “Using machine learning on macroeconomic, technical, and sentiment indicators for stock market forecasting,” Information, vol. 16, no. 7, 584, 2025. doi: 10.3390/info16070584.

[12] D. N. Kalange, “Prediction of stock prices using LSTM-ARIMA hybrid deep learning model,” Asian Journal of Probability and Statistics, vol. 27, no. 8, pp. 28–39, 2025. doi: 10.9734/ajpas/2025/v27i8791.