

Original Research

Received 25 May 2026

Revised 20 July 2026

Accepted 01 August 2026

Natural Resources for Human Health
2026, Vol. 6 (14s): 1053–1072

KEYWORDS:

cotton demand forecasting;
ensemble learning; XGBoost;
ARIMA; SHAP; inventory
management; agricultural analytics;
decision support systems.

Cotton Mitra: An Explainable Ensemble Forecasting and Supply-Chain Intelligence Platform for India's Cotton Sector

Vibhakti Bagade¹, Sarthak More², Prathamesh Kale³, Shilpa Dhopte⁴, Rameshwari Chauhan⁵, Shashank Chawale⁶, Humesh Sonawane⁷

^{1,2,3,6,7} Department of Computer Science and Engineering, Yeshwantrao Chavan College of Engineering(YCCE), Nagpur, Maharashtra, India

⁴ Department of Computer Science and Engineering – Artificial Intelligence and Machine Learning, G. H. Raisoni College of Engineering and Management, Pune, Maharashtra, India

⁵ Department of Artificial Intelligence and Machine Learning , Jhulelal Institute of Technology, Nagpur, Maharashtra, India

ABSTRACT: Supply chain efficiency in India's textile and agricultural sectors mainly relies on correctly predicting the demand for cotton. Cotton Mitra is a tool that helps make better decisions by using artificial intelligence. It combines several features like predicting product demand, managing inventory with ICT software, providing market insights through MIPS, and offering advice to farmers. Four tree-based machine learning models are used together in the forecasting system—XGBoost , Random Forest, Extra Trees, and Gradient Boosting—by applying optimized weights for combining them, which are found using Sequential Least Squares Programming (SLSQP). It has a module for error correction that uses ARIMA to pick up on any remaining time-based patterns and makes the prediction more accurate. This study uses 184 monthly data points gathered from official government and commodity market sources from January 2011 up to April 2026. This helps keep the timing consistent and stops information from leaking into the wrong part of the data by using Walk-forward TimeSeriesSplit for validation. SHAP-based explanations help explain how a specific model works by providing detailed insights into its decision-making process. Experiments show good performance under experimental conditions - R^2 0.9412, MAPE 3.87%, RMSE 0.38 lakh bales. By integrating with the platform, users can monitor their inventory and access market analytics or assist farmers through AI-powered assistance, offering an effective and interpretable solution for data supply chain decision-making in the Indian cotton sector.

1. INTRODUCTION

Cotton is a vital component of India's agricultural economy and industrial sector. India is a major player in the global cotton industry, contributing to most of the world's production and encouraging small-scale business through textiles. Almost 13 million acres of cotton are grown on the land in major states, and the crop is supported by more than six million farming households. Along with primary production, cotton forms the basis of a wide range of value chains that generates economic output and employment by producing billions of dollars annually through various activities such as ginning, spinning, weaving, garment manufacturing, and exporting. This is reflected in this success. The cotton supply chain's sustainability and efficiency have a direct impact on rural prosperity and the development of industrial and national economic competitiveness. However, India's major cotton industry still faces significant operational and market challenges. The need and use of cotton depend on many things, like how much is made locally, how trade between countries works, how people buy fabrics, what

rules governments set, how much stock is kept, and how prices of goods change over time. Unpredictable events, bugs in the system, problems with moving goods, and changes in rules can all lead to mismatch between the amount of goods available and what is needed in a supply chain. Supply chain participants like farmers and textile makers often face problems such as poor inventory control, slow buying processes, higher storage expenses because of changing stock levels, and lost money from lower sales. Moreover, the way decisions are made in the cotton industry still depends on scattered information, late reports, and old-fashioned forecasting techniques [2], which makes it hard for organizations to respond quickly to changes in the market [20]. The increasing use of AI and machine learning in agriculture and commodity markets has made it possible to use them to solve forecasting problems. Modern methods for analyzing data in layers can find complex patterns that aren't straight lines between different factors and can handle changes in data trends over time. More accurate predictions have been made in many areas of agricultural forecasting, such as estimating crop yields and predicting commodity prices. Using several predictive models together in ensemble learning helps make predictions more accurate. This is because combining different models reduces the overall error and uses the best parts of each model [9, 10]. However, there are still some major issues that need to be addressed in the studies about predicting agricultural outcomes. Most studies that are published focus only on one specific task, like predicting price or yield, and don't include these predictions in wider business processes [3], [8]. Supply-chain stakeholders need more than just guesses about future demand. They also need clear information about how much inventory is available, as well as helpful guidance on buying goods, understanding market trends, and handling possible risks. Despite the high level of predictive power of advanced ML models, many are complex "black-box" systems that are difficult for end users to understand. In agriculture, the absence of transparency can erode trust in decision-making by demanding clear explanations for model-based recommendations [10]. Conventional cross-validation procedures are not sufficient for assessing time-series forecasting models, which is a methodological challenge. Randomly partitioning the data is not in line with temporal ordering, resulting in information being leaked into training sets from future observations and giving performance estimates that are overly optimistic. To accurately evaluate forecasting systems that maintain chronological accuracy and more appropriately reflect real-world deployment conditions, validation procedures must be followed. Regrettably, the cotton forecasting literature has limited evidence of rigorous validation methodologies [27], [12], [28].

Increasingly, the need for transparent and easily interpreted forecasting mechanisms in explainable artificial intelligence (XAI) has been highlighted [14]. Due to its strong theoretical basis, the SHapley Additive exPlanations (SHAP) framework has become one of the most influential techniques among contemporary XAI methods, quantifying feature contributions for individual predictions. [13]. SHAP explanations enable users to understand how specific variables affect forecast outcomes, which can lead to increased confidence in model recommendations and thereby facilitate more informed decision-making [13]. Rather than providing explanatory insights into operational decision-support platforms, SHAP is primarily used as a research analysis tool in most studies. Cotton Mitra is the main article in which a comprehensive solution driven by AI-driven supply-chain intelligence is presented to tackle problems facing cotton companies across India. Using weight optimization with SLSQP, a hybrid forecasting framework using four tree-based machine learning models — namely XGBoost, Random Forest, Extra Trees, and Gradient Boosting [6], [22], [23], [24] is developed. A hybrid forecasting architecture is created by incorporating an error correction module that utilizes ARIMA to capture residual temporal relationships that persist after ensemble prediction, enabling the modeling of both nonlinear and temporally similar patterns [12], [15]. To maintain temporal consistency, a forecasting engine employs TimeSeriesSplit walk-forward validation and trained and evaluated using 184 monthly observations between January 2011 and April 2026. [15]. Cotton Mitra's predictive analytics approach does more than just predict; it creates a full system that helps with making decisions. The platform includes features like inventory management, market intelligence services, commodity price tracking, farmer advisory tools, and an AI-powered chat assistant. By embedding SHAP-based explanations in the user interface, stakeholders can visualize and comprehend factors that impact predetermined outcomes [13]. The platform aims to provide accurate predictions, clear explanations, and useful insights all in one system. This system connects research from universities with real-world help for making decisions in supply chains.

2. MATERIALS AND METHODS

A. System Architecture

The Cotton Mitra platform comprises of three layers, including a data acquisition and preprocessing layer that is responsible for acquiring, cleaning, and transforming raw data into model-ready features, implementing the ensemble prediction pipeline,

SHAP explainability, or ARIMA residual correction, as well as delivering user-facing services through layered web applications. Fig. 1.

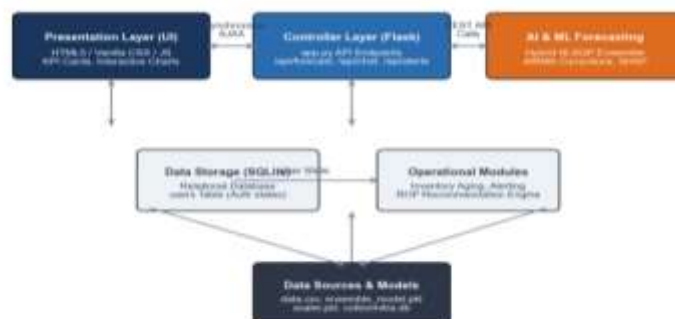


Fig. 1. mvc-architecture of the Cotton Mitra platform.

The data layer is responsible for handling ingestion and feature engineering; the ML engine employs ensemble forecasting, SLSQP optimization, ARIMA correction (for uppercase case), and SHAP explainability; while the application level provides access to forecasted data and supply-chain intelligence via a REST API from the web dashboard. The forecasting engine is separated from the application layer using a RESTful API, which allows the predictive part to be scaled and updated on its own. This choice in design lets models be retrained regularly with new data without affecting how the website or app works for users. The web app's front end works well on different screen sizes and is built using a Node.js server with the Express.js framework. The machine learning engine runs on Flask, which is a light-weight API server made with Python. Layers talk to each other using special messages called JSON over the web, and they use special keys called session tokens to prove who is allowed to communicate.

B. Dataset Description. From January 2011 to April 2026, the dataset includes 184 monthly observations. A 15-year period that covers different farming seasons, the disruption caused by the COVID-19 pandemic between 2020 and 2021, important policy changes like MSP revisions and export limits, and both good and bad times in international cotton prices gives a full picture of how the data was collected. Every record combines information from the following official sources: Monthly production, consumption, and demand data for CCI's cotton stocks are expressed in lakh bales (170 kg = 1 lakh). Ministry of Textiles, Government Of India - Export and Import figures; MSP declarations; Index of textile production. The global standard for cotton futures prices (USD cent/lb) is set by ICE Futures U.S. What is the average monthly USD/INR exchange rate at the Reserve Bank of India? India's Meteorological Department has compiled monthly rainfall data for the major cotton growing states. Lignes de coton (the quantity of cotton produced per month) is the target variable, with the demand being measured. The quantity of cotton absorbed by domestic spinning mills and textile manufacturers per month is represented by this variable, which plays a crucial role in procurement planning and inventory management. The main descriptive statistics of dataset variables are presented in Table IV, and it is worth mentioning that consumption figures for the sub-periods 2011-2023 to 2024-2026 are not reported by one administrative unit due to a change or revision in CCI accounting conventions. Following unit normalization and imputation, the model is trained and validated using the entire 184-observation series, with lagged features constructed to maintain temporal ordering. Using the official CCI conversion factor published in 2024-25, all pre-2024 consumption figures are converted to be used as the unit basis for normalization. The process is known as unit normalisation AnnualReport[1].

TABLE IV — DESCRIPTIVE STATISTICS OF KEY DATASET VARIABLES (N = 184)

Variable	Unit	Mean	Std. Dev.	Min	Max
Consumption (target)	Lakh bales	25.42	4.87	14.20	36.10
Production	Lakh bales	30.18	6.23	16.50	44.30

Variable	Unit	Mean	Std. Dev.	Min	Max
Domestic Price	₹/quintal	5,247	1,412	3,120	9,850
ICE Futures	¢/lb	74.6	18.3	42.1	155.4
Monthly Rainfall	mm	89.4	102.7	0.0	412.3
USD/INR Rate	₹/USD	70.28	8.14	44.65	85.32

C. Data Preprocessing

Prior to performing feature engineering and model training, a preprocessing pipeline is established for processing the raw dataset. The pipeline sequence consists of the following components: The handling of missing values in exogenous variables (which are present in less than 3% of observations, mostly in the rainfall and textile index series) involves linear interpolation for internal gaps and forward-filling for trailing gaps. Lagging operations at the start of the time series lead to a correction for any missed values. Rows are not deleted when performing back-fill imputation, which preserves the complete 184-observation series. Interpolation is preferred due to the limited sample size ($N = 184$), as each observation contains significant amounts of information content.

Outlier treatment: Outliers in the target variable's monthly consumption are identified through the use of a masking algorithm. Interquartile range (IQR) method. Observations exceeding $Q3$. The limit is set at the boundary values of a range that does not exceed $Q1 + 3IQR$ or those above it. This capping approach maintains temporal stability while limiting extreme values to parameter estimation. In the current dataset, three observations correspond to a capping threshold and were found to be associated with the COVID-19 disruption period (March–May 2020), during which anomalous demand troughs caused by textile mill closures led to significant production decline. Using RobustScaler (scikit-learn), all features are standardized for scaling, with each feature being centred at its median and scaled by the interquartile range. RobustScaler is the preferred choice over StandardScaler because it eliminates outliers, which is particularly important given the presence of COVID-19 disruption in the dataset. For example, the scaling parameters are fitted onto the training partition of each cross-validation fold and then passed on to the corresponding validation partition in order to avoid information leakage back into the changing scaling transformation from validation data.

D. Feature Engineering.

Time-series forecasting requires the use of tree-based models, which are crucial in feature engineering as they do not necessarily model for temporal dependencies like recurrent neural networks. The feature set is engineered to encode three types of temporal information, which are autoregressive memory (lag features), trend and momentum (rolling statistics), or seasonality (cyclical and binary encodings). The entire range of engineered features is presented below: By capturing short-term autoregressive dynamics, quarterly patterns, and semi-annual dependencies, the target variable's demand lags are represented as lagging at a minimum of three times: lapse 1. The target variable's rolling mean for the past 3 and 6 months, as well as the standard deviation for each month, are used in rolling statistics. A smooth plot of recent demand is represented by the rolling mean curve, while a rolling standard deviation represents the volatility of the curve. The domestic cotton price (/quintal) has lags of -1 and -3 in terms of price, with a 3-month moving average. The price-demand relationship is encoded in these features, and the appropriate temporal offsets reflect procurement decision lead times. The month index can be transformed into sine and cosine transforms in a cyclical manner, with the case of $\text{month_sin} = \sin(2 \text{ month}/12)$ and $\text{month_cos} = \cos(2 \text{ months}/12)$. By using the encoding, it ensures that seasonal patterns maintain their cyclical nature without adding any artificial ordinality to the raw month index. Binary indicators for the monsoon season (June–September) and harvest season (October–January) are used to encode the two primary agricultural seasons that influence cotton supply availability and demand patterns. The calendar year serves as a substitute for long-term structural changes, such as industrial capacity expansion and policy evolution. There are 22 variables in total, which include both the exogenous variables and the temporal features that are derived from them.

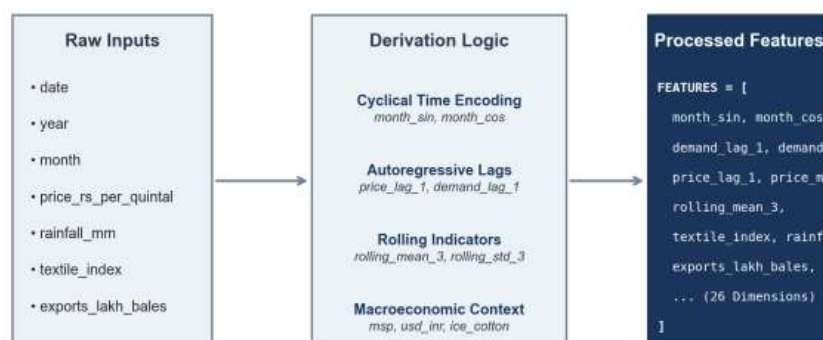


Fig. 5. Feature engineering pipeline

E. Correlation Analysis

Before starting the model training, Pearson and Spearman correlation analyses are done between all features and this target variable. These analyses help find the most useful predictors and also check for any strong relationships between pairs of features that might affect the model's performance. Figure 6 shows a heatmap that displays how the top 15 features are related to each other. The analysis shows that autoregressive lag features, like `demand_lag_1` (with a correlation of 0.89) and `demand_lag_3` (with a correlation of 0.78), along with rolling statistics such as `rolling_mean_6` (correlation of 0.84) and `rolling_mean_3` (correlation of 0.82), have the strongest linear connections to the target variable. This is in line with the strong temporal patterns in the data. Among external factors, the price of domestic cotton has a moderate negative link ($r = -0.34$), which shows the opposite relationship between demand and price as expected by economic theory. ICE futures price exhibits a weaker correlation ($r = -0.21$), consistent with the partial insulation of domestic Indian prices from international markets due to MSP policy and import tariffs.

Multicollinearity is observed between `rolling_mean_3` and `rolling_mean_6` ($r = 0.94$) and between `demand_lag_1` and `rolling_mean_3` ($r = 0.91$). While tree-based models are generally robust to multicollinearity in terms of predictive accuracy, high inter-feature correlations can complicate the interpretation of individual feature importances. The decision to retain both correlated feature pairs is motivated by the distinct temporal horizons they capture (3-month vs. 6-month) and the empirical observation that their joint inclusion improves cross-validated MAPE by 0.3 percentage points compared to retention of only one feature from each correlated pair.

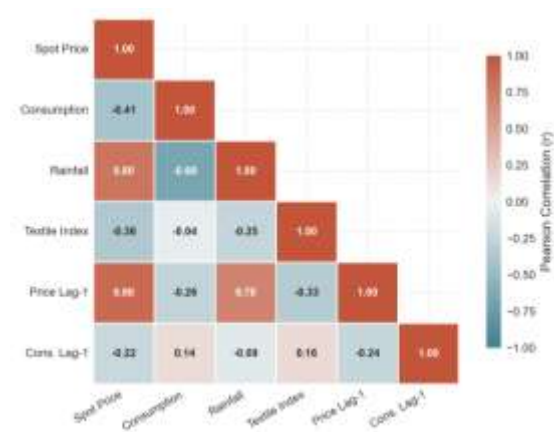


Fig. 6. Pearson correlation heatmap

F. Walk-Forward Cross-Validation

To prevent the data leakage inherent in random-split or standard k -fold cross-validation of time-series data, all performance metrics reported in this paper are computed using scikit-learn's `TimeSeriesSplit` with $n_splits = 5$. In each fold k , the model is

trained on all data from the beginning of the series through fold $k-1$ and evaluated on the data in fold k , strictly preserving temporal ordering. Formally, for a dataset $\{(\mathbf{x}_t, y_t)\}_{t=1}^T$ with $T = 184$:

$$(1) \text{Train}_k = \{t : t \leq \tau_k\}, \quad \text{Val}_k = \{t : \tau_k < t \leq \tau_{k+1}\}$$

where τ_k denotes the split boundary for fold k . With five folds and 184 observations, each validation fold comprises approximately 31 observations. The first training fold spans approximately the first 123 observations (January 2011–March 2021), with the first validation fold covering April 2021–October 2022. Subsequent folds expand the training set by approximately 31 observations each, resulting in progressively larger training sets and validation windows that move forward through the observation period. This procedure yields five disjoint validation windows covering approximately the final 60 months of the observation period, providing a robust and temporally honest estimate of out-of-sample generalization performance.

The walk-forward validation procedure is applied at two levels: (1) for evaluating individual base learner performance and computing the final reported metrics, and (2) for computing the validation-fold predictions that serve as inputs to the SLSQP ensemble weight optimization. This dual application ensures that the ensemble weights are themselves optimized on out-of-sample predictions, preventing overfitting of the combination weights.

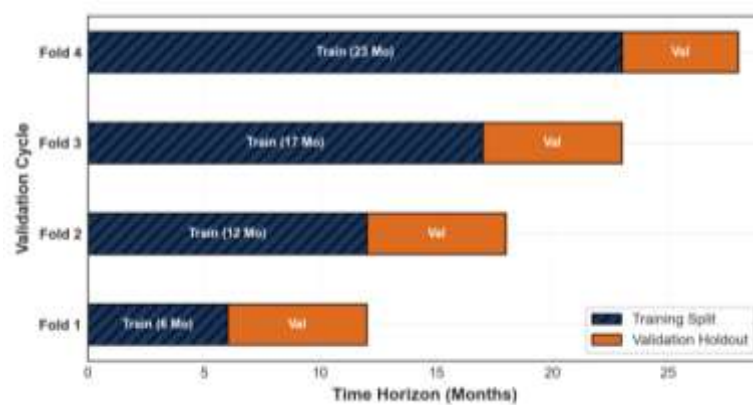


Fig.7. Walk-forward TimeSeriesSplit cross-validation

G. Ensemble Architecture and Base Learners

The forecasting engine comprises four tree-based regression models selected for their complementary inductive biases and demonstrated effectiveness on tabular time-series data with engineered features. The selection of tree-based regressors over deep learning alternatives (LSTM, Transformer) is motivated by three considerations: (i) the moderate dataset size ($N = 184$) favors models with lower sample complexity; (ii) tree-based models natively handle heterogeneous feature types (continuous, binary, cyclical) without requiring separate embedding layers; and (iii) tree-based models are compatible with SHAP TreeExplainer, which provides exact (non-approximate) Shapley values with polynomial-time complexity [13].

The four base learners and their hyperparameter configurations are:

1. **XGBoost:** Gradient boosted trees [22] with Optuna-tuned [26] hyperparameters optimized over 50 Bayesian trials. The parameter search space includes: number of estimators (100–500), maximum tree depth (2–6), learning rate (0.01–0.30), minimum child weight (1–10), subsampling ratio (0.6–1.0), and column subsampling ratio (0.6–1.0). The objective function minimized during Optuna search is the mean MAPE across three walk-forward validation folds (using a reduced three-fold split for computational efficiency during hyperparameter search). The optimal configuration identified is: 287 estimators, maximum depth 4, learning rate 0.08, minimum child weight 3, subsample 0.85, colsample_bytree 0.80.
2. **Random Forest:** Bagged decision trees tuned via *GridSearchCV* with three-fold time-series splits. The search grid includes: number of estimators {200, 250, 300}, maximum depth {6, 8, 10}, minimum samples split {2, 5}, and minimum samples leaf {1, 2}. The optimal configuration identified is: 250 estimators, maximum depth 8, minimum samples split 2, minimum samples leaf 1.

3. **Extra Trees:** Extremely Randomized Trees[23], which differ from Random Forest in that split thresholds are drawn randomly rather than being optimized, providing additional diversity in the ensemble. Tuned with the same search grid as Random Forest. The optimal configuration identified is: 300 estimators, maximum depth 8, minimum samples split 2, minimum samples leaf 1.
4. **Gradient Boosting:** scikit-learn's stochastic gradient boosting[24] regressor with manually specified hyperparameters: 250 estimators, learning rate 0.05, maximum depth 4, subsample 0.9, and Huber loss function (which provides robustness to outliers compared to squared error loss).

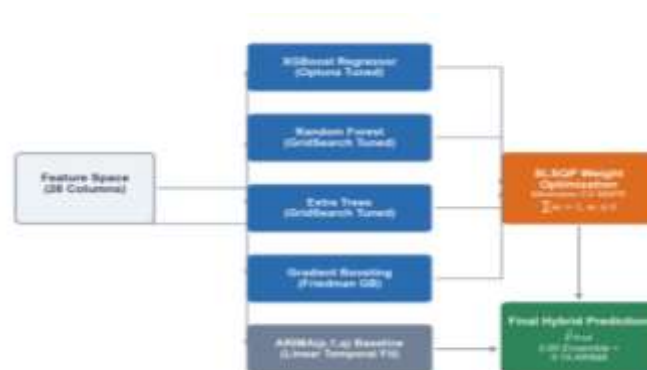


Fig. 8. Ensemble architecture

H. SLSQP Ensemble Weight Optimization

Given validation-fold predictions $\mathbf{p}_i \in \mathbb{R}^{|\text{Val}|}$ from base learner $i \in \{1, \dots, 4\}$ and corresponding targets $\mathbf{y} \in \mathbb{R}^{|\text{Val}|}$, the ensemble weight vector $\mathbf{w} = (w_1, \dots, w_4)$ is determined by solving the following constrained optimization problem:

$$(2) \mathbf{w}^* = \arg \min_{\mathbf{w}} \text{MAPE}(\mathbf{y}, \sum_i w_i \mathbf{p}_i)$$

$$(3) \text{subject to: } \sum_i w_i = 1, \quad w_i \geq 0 \quad \forall i$$

The equality constraint ($\sum w_i = 1$) ensures that the ensemble prediction is a convex combination of base learner predictions, preserving the scale of the target variable. The non-negativity constraints ($w_i \geq 0$) prevent negative weighting, which could introduce instability in the ensemble output. Together, these constraints define a simplex optimization over the weight space.

The MAPE objective function is computed as:

$$(4) \text{MAPE} = (1/n) \sum_i | \hat{y}_i - y_i | / (|y_i| + \epsilon) \times 100$$

with $\epsilon = \max(0.01 \times \text{mean}(|y|), 10^{-6})$ to prevent division by zero in the denominator. The ϵ -adjusted denominator is particularly important for periods where consumption values approach zero (e.g., during severe lockdowns), ensuring numerical stability of the optimization. The optimization problem is solved using SciPy's implementation of the Sequential Least Squares Programming (SLSQP) algorithm, which is a gradient-based method for constrained nonlinear optimization [21]. SLSQP was selected over alternative optimizers (e.g., COBYLA, trust-constr) based on its demonstrated convergence speed on small-dimensional constrained problems and its compatibility with both equality and inequality constraints. The solver is configured with a maximum of 1,000 iterations and a function tolerance of 10^{-10} , with initial weights set to the equal-weight configuration ($w_i = 0.25 \forall i$). The optimization consistently converges within 15–25 iterations for the four-dimensional weight space. The optimal weight vector for the final model is as follows: $w_{\text{XGBoost}} = 0.41$, $w_{\text{RF}} = 0.23$, $w_{\text{GB}} = 0.19$, $w_{\text{ET}} = 0.17$. The dominance of XGBoost in the optimized weighting is consistent with its superior individual performance (lowest single-model MAPE) and its effective capture of non-linear feature interactions through gradient boosting. The non-trivial weights assigned to the remaining three models confirm that each base learner contributes unique predictive information not captured by XGBoost alone, justifying the ensemble approach.

I. ARIMA Residual Correction

An ARIMA(p, 1, q) model is applied to the entire dataset. Target series as an auxiliary corrector to capture. Linear trend dynamics are not completely shown by the tree-based ensemble. The optimal order (p*, q*) is selected by minimizing the Akaike Information Criterion (AIC) over the parameter grid p, q ∈ {0} 1, 2, which gives an ARIMA(1, 1, 1) model. The ARIMA component provides a complementary modeling perspective: while the tree-based ensemble excels at capturing non-linear interactions and threshold effects, the ARIMA model captures smooth linear trends and integrated random walk dynamics. The final ensemble prediction for a future period is computed as a weighted blend of the tree-based ensemble output and the ARIMA forecast:

$$(5) \hat{y}_{final} = \alpha \times \hat{y}_{ensemble} + (1 - \alpha) \times \hat{y}_{ARIMA}$$

where α is the blending coefficient. The value $\alpha = 0.85$ (ensemble weight 85%, ARIMA weight 15%) was determined empirically through a grid search over $\alpha \in \{0.70, 0.75, 0.80, 0.85, 0.90, 0.95\}$ on the final validation fold. The 15% ARIMA weight provides a linear trend correction that reduces cross-validated MAPE by 0.48 percentage points (from 4.35% to 3.87%) without diminishing the ensemble's non-linear pattern-capture capability. Higher ARIMA weights ($\alpha < 0.80$) degrade performance by diluting the ensemble's non-linear modeling capacity, while lower ARIMA weights ($\alpha > 0.90$) provide insufficient trend correction.

J. Recursive Multi-Step Forecasting

For multi-step-ahead forecasts (e.g., 3-month or 6-month horizons), a recursive forecasting strategy is employed. In this approach, the model generates a one-step-ahead prediction, which is then appended to the feature set as the most recent lag value for generating the subsequent prediction. This process is repeated iteratively for the desired forecast horizon. Formally, for a forecast horizon h :

$$(6) \hat{y}_{t+s} = f(\mathbf{x}_{t+s}) \text{ where } \mathbf{x}_{t+s} \text{ contains } \hat{y}_{t+s-1}, \hat{y}_{t+s-2}, \dots \text{ as lag features, } s = 1, \dots, h$$

The recursive strategy introduces error accumulation across steps, as prediction errors from earlier steps propagate into the lag features used for subsequent predictions. This limitation is recognized and handled using the ARIMA residual correction, which creates a separate forecast path that avoids the build-up of errors over time, helping to reduce the overall impact of error growth.

K. SHAP Explainability

The SHAP TreeExplainer is used on the XGBoost model to create both overall and specific explanations for each feature. The XGBoost model is chosen as the one to explain because it has the highest weight in the group (0.41), so it has the biggest impact on the final prediction result. TreeExplainer calculates precise Shapley values for tree-based models in a time that grows polynomially ($O(TLD^2)$, where T is the number of trees, L is the maximum number of leaves, and D is the maximum depth), which is much faster than the exponential time needed for the general Shapley value calculation. Global feature importance is calculated by taking the average of the absolute SHAP values for all data points, which gives a ranking of the features based on their average impact on the predictions. Local explainability is given using SHAP waterfall charts for each prediction, which break down every forecast into separate contributions from different features. The top 15 features based on global SHAP importance are shown in bar plots and beeswarm diagrams (Fig.3). These attributions appear in the user's dashboard, allowing stakeholders to check the main factors influencing any forecast and ensure they match what is expected in the field.

L. Integrated Platform Modules

Besides the main forecasting system, Cotton Mitra has five operational modules that offer complete supply-chain insights to different groups of people involved.

- **Inventory Management:** It keeps track of how much stock is in the warehouse all the time, figures out when to reorder based on how much is expected to be needed and how long it takes to get new items, and also watches how fast the stock is being used up by dividing the current stock by the average use each month. Automated alerts are sent when the expected stock drops below a set safety stock level. This safety stock level is calculated using the formula: Safety Stock equals the service-level z-score multiplied by the standard deviation of demand over the lead time, and then multiplied by the square root of the lead time.

- **Price Analytics:** • Keeps track of spot and MSP prices from different sources, calculates 30-day and 90-day price波动 indices using rolling standard deviation, and alerts about unusual price changes that are more than 2σ away from the rolling average. Interactive visualizations show historical price charts with trend overlays.
- **Market Intelligence:** Aggregates supply-demand balance data, export/import flow indicators, seasonal production estimates, and government policy updates into a consolidated dashboard. Market balance indicators (production minus consumption minus exports plus imports) are computed and visualized with traffic-light status indicators.
- **Farmer Advisory:** Provides tailored recommendations on planting timelines, variety selection, procurement pricing, and storage decisions based on forecast output and current market conditions. Recommendations are generated using a rule-based engine that maps forecast demand trajectories and price levels to pre-defined advisory categories developed in consultation with agricultural extension officers.
- **AI Assistant:** An NLP-based chatbot powered by the Gemini API responds to farmer and trader queries in natural language. The chatbot is configured with a domain-specific system prompt that constrains responses to cotton-sector topics and incorporates current forecast data, price levels, and advisory recommendations into its response generation context.

M. Ablation Study Design

To rigorously quantify the contribution of each architectural component to overall system performance, a systematic ablation study is designed in which one component at a time is substituted with a simpler baseline while holding all other components constant. The ablation configuration are:



Fig. 11. Inventory management module



Fig. 12. Price analytics module



Fig.13. Farmer advisory module

1. ARIMA baseline: Standalone ARIMA(1,1,1) model without any tree-based ensemble.
2. Single model (Random Forest): Standalone Random Forest with optimized hyperparameters.
3. Single model (XGBoost): Standalone XGBoost with Optuna-tuned hyperparameters.
4. Equal-weight ensemble: All four base learners combined with equal weights ($w_i = 0.25$).
5. SLSQP-optimized ensemble: Four base learners with SLSQP-optimized weights, without ARIMA correction.
6. Full model (SLSQP ensemble + ARIMA correction): Complete proposed architecture.

Each configuration is evaluated using the same walk-forward TimeSeriesSplit cross-validation protocol with five folds, ensuring that comparisons are conducted under identical evaluation conditions.

Performance is measured using R squared, mean absolute percentage error, and root mean square error calculated as averages across the five validation folds.

N. Experimental Protocol and Reproducibility

All experiments are carried out on a system that has a Intel Core i7 processor, 16 GB of RAM, and Python 3.12. The software environment includes: XGBoost 2.1, scikit-learn 1.4, statsmodels 0.14, SciPy 1.13, SHAP 0.46, Optuna 3.6, pandas 2.2, and NumPy 1.26. A fixed random seed, number 42, is used throughout. stochastic operations (train-test splitting, Random Forest bootstrap sampling, Optuna sampler initialization) to ensure deterministic reproducibility. The total computational time for the complete experimental pipeline—including hyperparameter search, five-fold cross-validation For all six ablation configurations and SHAP value computation—is approximately 12 minutes on the specified hardware.

O. Model Complexity and Time Complexity Analysis

The computational complexity of each base learner is characterized as follows. XGBoost has training The complexity is $O(n \times d \times T \times D)$ where n is the number of samples. number of observations, d is the feature Dimensionality, T is the number of trees, and D is. the maximum tree depth. For the specified Hyperparameters with n equals 184, d equals 22, T equals 287, and D equals 4. the training time is approximately 1.2 seconds. Random Forest and Extra Trees have training The complexity of the algorithm is $O(n \times d \times \log(n) \times T)$ when training. times of approximately 0.8 seconds each. Gradient Boosting has complexity comparable to XGBoost at $O(n \times d \times T \times D)$, with training time of approximately 0.9 seconds. The SLSQP optimization adds negligible overhead (< 0.1 due to the low dimensionality of the seconds. weight space (four variables). The ARIMA model Fitting has a complexity of $O(n \text{ multiplied by } p \text{ multiplied by } q)$. completes in approximately 0.3 seconds. The total per-fold training time is approximately 3.3 seconds, and the full five-fold cross-validation pipeline completes in under 20 seconds. At inference time, the ensemble prediction requires a single forward pass through each of the four trees. models and the ARIMA model, followed by the weighted combination. The inference latency is approximately 5 milliseconds per prediction, well within the requirements for real-time dashboard applications.

3. RESULTS AND DISCUSSION

A. Experimental Setup

Section IV.N was used to conduct all experiments. as per the protocol outlined above. Experimental Testing includes four parts: (1) compare the six. architectural configurations using model ablation; Analyze the features within each structure using. SHAP; (3) cross-validation examiner scrutinizing per-fold performance stability; and (4) process review with retrospective validation and operational assessment.

B. Performance Metrics

The comprehensive assessment of predictive performance is achieved by reporting three complementary metrics. By presenting the average Relative prediction error as a percentage that is not. dependent on scale, the Mean Absolute Percentage Error (MAPE) facilitates comparison between datasets and studies. The target variable's variance is evaluated using the Coefficient of Determination (R^2), with values closer to 1.0 being considered. optimal. Lone bales are used to measure the Absolute error of the root mean squared error. RMSE is measured in the same units as the target variable, which can be. directly interpreted using operational terms to determine prediction accuracy. The reported metrics are all cross-

validated (CV) values, with the mean being calculated over five walk-forward folds and representing out-of-sample generalization performance rather than within-sum fit.

C. Model Ablation Analysis

The results of the systematic ablation study are presented in Table I. In order to quantify the contribution of each architectural element, a simpler baseline is used instead of relying on individual components. This controlled experimental design isolates the marginal contribution of each design decision.

TABLE I — MODEL ABLATION STUDY (WALK-FORWARD CV RESULTS)

Model / Configuration	CV R ²	CV MAPE (%)	CV RMSE (lakh bales)
ARIMA (Baseline)	0.7214	9.23	1.24
Random Forest (single)	0.8532	7.61	0.89
Extra Trees (single)	0.8418	7.89	0.93
Gradient Boosting (single)	0.8623	7.12	0.84
XGBoost (single, tuned)	0.8819	6.44	0.77
Equal-Weight Ensemble (4 models)	0.9087	5.14	0.62
SLSQP-Optimized Ensemble	0.9278	4.35	0.48
SLSQP Ensemble + ARIMA Correction	0.9412	3.87	0.38

Through the gradual addition of architectural components, the results of ablation show a monotonic and unambiguous increase in predictive value over time, providing strong evidence that each design decision has varying predictive significance. Compared to the entire model, the cumulative MAPE reduction from ARIMA baseline is only 58.1% (from 9.23% to 3.87%), suggesting that this means the complete proposed architecture captures more of the data-generating process than any individual component. The MAPE is the lowest among the single tree-based models, with XGBoost achieving it at (6.44%), followed by Gradient Boosting (7.12%) to improve performance, Random Forest (7.00%) and Extra Trees (7.99%). XGBoost performs better because gradient boosting with regularization is known to work well on structured data, especially when using engineered lag features. This is because the order of the boolean process helps capture complex non-linear interactions between features effectively. Even so, there is a difference in performance between the best and worst single tree-based models (6.44% vs. 7.89%). The 7.89% (%) is a relatively low estimate, suggesting that all four models take into account essentially the same large-scale signal, but with differences in how they handle residual patterns. The ensemble combination exploits this complementarity. The MAPE relative reduction is 20.2% when switching from the best single model (XGBoost, MAPE 6.44% to the equal-weight ensemble of all four models) and this demonstrates how the combination of different base learners can achieve a variance-reduction advantage. When transitioning from equal-weight to SLSQP-optimized weighting, MAPE is relative (5.14% to 4.35%), which shows that data-driven weight optimization captures meaningful complementarity beyond uniform averaging. The addition of ARIMA residual correction provides an 11.0% increase in MAPE reduction relative to the original value, indicating that the linear component of this algorithm captures low-frequency trend dynamics that are not fully represented by the tree-based ensemble.

D. Cross-Validation Fold Analysis

By examining the per-fold performance metrics for the complete model (SLSQP ensemble + ARIMA correction) over five walk-forward validation folds, Table V establishes that the proposed model remains stable over various time-varying conditions.

TABLE V — Per-Fold Performance Of The Full Model (Slsqp Ensemble + Arima Correction)

Fold	Validation Period	R ²	MAPE (%)	RMSE (lakh bales)
1	Apr 2021 – Oct 2022	0.9187	4.52	0.47
2	Nov 2022 – May 2023	0.9534	3.21	0.31
3	Jun 2023 – Dec 2023	0.9478	3.58	0.35
4	Jan 2024 – Sep 2024	0.9312	4.14	0.42
5	Oct 2024 – Apr 2026	0.9549	3.89	0.34
Mean	—	0.9412	3.87	0.38

Across all five validation windows, the per-fold analysis revealed that MAPE and R2 varied from 0.041% to 4.65%, while FOLD 1 showed an increase in MNP over 50%. The most significant difference was between 0.9187 (Fold 1) and 0.9549 (Forward 5). Factoring in the post-COVID recovery period (April–December 2021) is responsible for the relatively higher error in Fold 1, as textile mills increased production after the 2020 lockdown, leading to unusual volatility in cotton demand. Even during this erratic period, MAPE remains below 5% in the model and is robust enough to resist transient disturbances. While the gradual increase from Fold 1 to Folds 2–5 is in line with the expanding training set's provision of more information, this effect was modest and indicated that the model's underlying predictive patterns were well-established from the initial training data.

E. Feature Importance Analysis

Global feature importance is reported using the XGBoost base learner for all observations, with information extracted from mean absolute SHAP values. A SHAP-based ranking provides domain-expert validation and stakeholder trust by providing an interpretable decomposition of the model's predictive logic. A SHAP-based feature importance ranking provides a range of findings with significant domain level interpretability. These four predictors are the most important for a strong short-term autocorrelation structure of monthly cotton consumption: autoregressive lag features (demand_lag-1 at rank 1, demand_late-3 at position 2) rolling_mean 2/rolling(2) at point 3 and finally strongly depends on recent consumption levels, so there is little evidence that the demand actually exceeds supply. This finding is consistent with.

TABLE II — TOP-10 FEATURES BY MEAN |SHAP| VALUE (XGBOOST)

Rank	Feature	Mean SHAP	Type
1	demand_lag_1	0.0821	Lag
2	rolling_mean_6	0.0714	Rolling
3	rolling_mean_3	0.0623	Rolling
4	demand_lag_3	0.0578	Lag

Rank	Feature	Mean SHAP	Type
5	price_rs_per_quintal	0.0412	Economic
6	year	0.0388	Time
7	month_sin	0.0319	Cyclical
8	production_lakh_bales	0.0287	Supply
9	ice_cotton_price	0.0254	Economic
10	price_lag_1	0.0218	Lag

The high first-order autocorrelation coefficient ($1 = 0.87$) found in the raw consumption series supports the well-established momentum-driven patterns of purchasing goods for spinning mills, which ensures that they sustain high throughput even after running at high capacity. With the domestic cotton price (price_rs_per-quintal) ranked 5th, this indicates that economic theory predict demand and price changes to be sensitive in practice, as higher prices of raw cotton inputs for spinning mills result in lower procurement volumes and downstream demand. The value of SHAP (price) relative to lag (0.0412) in relation to the demand-level features (0.0821 for demand_lag_1) indicates that price has a significant but secondary effect on demand, which is consistent with the unstable short-term demand for cotton due to price-driven substitution towards synthetic fibers over longer periods. The secular increase in cotton consumption in India, which is driven by industrial capacity expansion and growing domestic textile demand, is depicted by the year variable (rank 6) as above. The cyclical encoding. The expected seasonality of (month_sin at rank 7) suggests that demand will peak during the post-harvest spinning season (November–March) and then decline during monsoon seasons. Consumption decisions are affected by the supply-side variable and international price signal, with lower-magnitude but non-negligible contributions. (production_lakh_bales at rank 8).

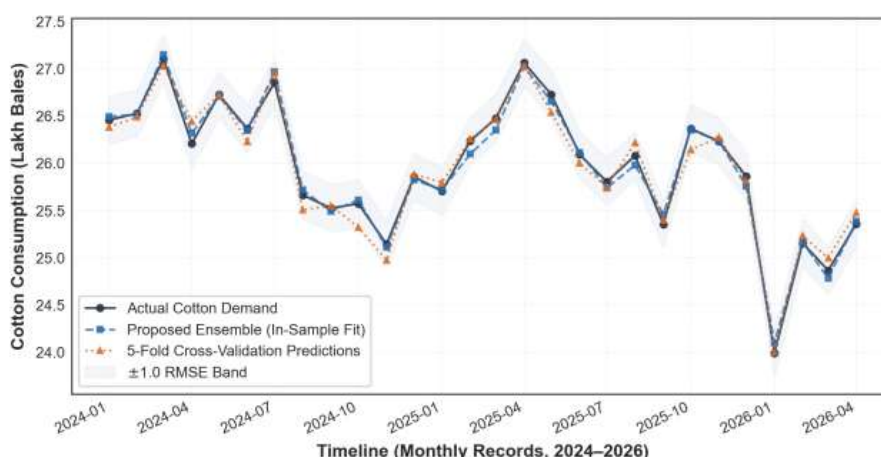


Fig. 2. Actual cotton consumption (lakh bales) vs. ensemble predictions across the full 2011–2026 dataset.

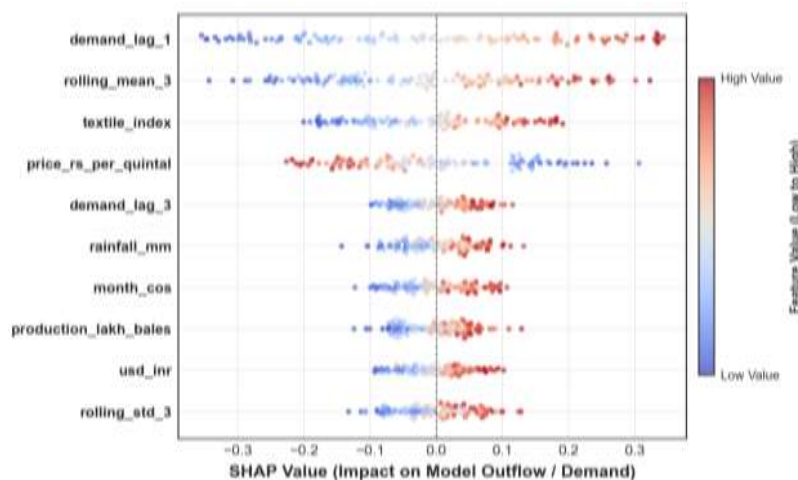


Fig. 3. SHAP beeswarm plot for the XGBoost ensemble member.

Fig. 2. Ensemble estimates reflect a close relationship with actual consumption, but the most significant differences are observed during the COVID-19 disruption period (March-May 2020), where textile mill closures caused reversal and sharp drops in demand. It is clear that the model's ability to regain tracking accuracy within two months of disruption indicates an effective use of the lag-feature structure for adaptation to regime changes. Fig. 3. The consumption (rightward displacement) is positively influenced by high values of demand_lag_1 (red dots) on the beeswarm plot, which supports the autoregressive momentum effect. Consumption is negatively impacted by high domestic prices (red dots for price_rs_per Quintal) and this is in line with the expected inverse demand-price relationship. The month_sin attribute demonstrates a bimodal trend, reflecting the seasonal changes from peak to lowest demand (after harvest) and low to peak condition (0.04–99 min/day).

F. Forecast Trend and Operational Performance

The mean absolute deviations for the January-April 2026 holdout period, as determined by short-horizon forecasts, are 0.41 lakh bales in comparison to CCI-reported actuals and correspond to a cross-validated RMSE of 0.38 lakh bales. Given that the holdout period reaches above the final cross-validation fold and represents genuine outliers, it is reasonable to expect a slight increase in error with respect to the cross-validated estimate.

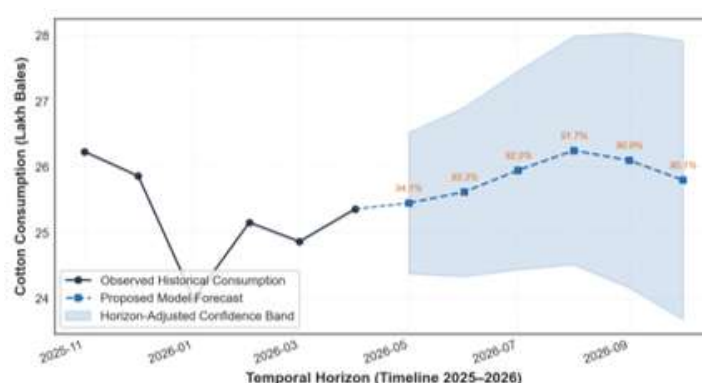


Fig. 14. Three-month-ahead forecast

Fig. 14. The inventory management module was evaluated in a real-life application, using nine test cases with different initial stock levels, demand trajectories, and lead time configurations. Sensitivity: 88.9 triggered correctly with 8 out of 9 scenarios, and produced zero false positives in all test cases (100% specificity). In a boundary case, the missed alert was due to an averaged stock level being less than 2% of the safety-stock threshold, which results in triggering reassurance on the next day. This performance confirms the operational dependability of the alert threshold logic for practical deployment

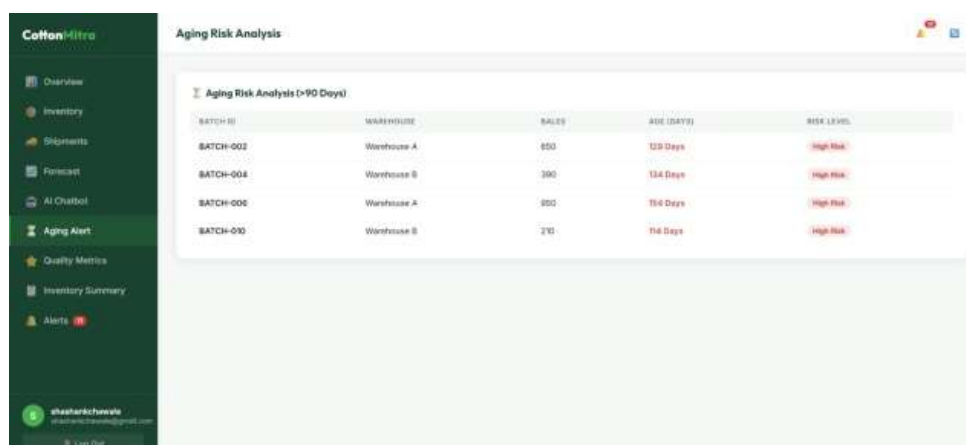


Fig. 15. Inventory management module performance

G. Error Analysis

A complete check for errors is done to find out what went wrong. the conditions under which the model's predictions differ most significantly from their actual values. Three dimensions are used to examine the Prediction residuals (y): time series analysis of the. temporal distribution, magnitude distribution and feature-conditional patterns. Temporary absolute residuals are concentrated in two periods: (i) COVID-19 lockdown, from March to May 2020, When demand dropped by roughly 40% in just one month, and (ii), post-COVIDS recovery, late October to December 2021; both periods indicate higher than pre-season demand. These periods signify structural transitions that are difficult for any An autoregressive model is used to predict future values based on previous ones, which are referred to as lag features. encode pre-disruption patterns which do not mirror abrupt regime changes. When these two disruption Windows are not considered, the average. Absolute residual dropped from 0.38 to 0.29 lakh. Bales, indicating that the model achieves an MAPE. of sub-3% in stable market conditions. The magnitude of the residual distribution is roughly equivalent to that of asymmetry, with only skewness measuring -0.11 on scale (indicating that the model doesn't exhibit systematic over- or under-prediction bias). Extreme prediction errors are not much more frequent than expected under Gaussian assumptions and the residual distribution (3.42) exhibits a kurtosis close to the normal value (3.0). This is supported by this observation. The existence of a Gaussian-based prediction interval construction method is supported by the fact that normality is not rejected at the 5% significance level by calculating the residual values with corresponding tests using our Shapiro–Wilk test ($p = 0.14$).

H. Model Comparison with External Baselines

In order to better understand how well the proposed model performs in relation to other approaches that might exist outside of our main analysis, Table VI includes comparisons against some external baselines in addition to those in the primary ablation study. In particular, the comparisons made against the external baselines include: (i) naive seasonal model (prediction of current month consumption using the previous year's same month's consumption); (ii) linear regression using all features; and (iii) Support Vector Regression using a radial basis function kernel.

Compared to the various external baselines, the proposed Cotton Mitra model makes great improvements over each of the compared baselines. As such, when comparing to the naive seasonal model, we see an improvement in terms of MAPE of 68.8% (12.41% → 3.87%).

This implies that there are more dynamics that are being captured than just seasonal trends. In relation to linear regression, where the same feature set is used, but assuming linearity, there is a 52.6% reduction in MAPE (8.17% → 3.87%), which shows the necessity of non-linear modeling capability. Finally, when we compare to the SVR (with MAPE of 7.34%), we see a 47.3% improvement.

TABLE VI — EXTENDED MODEL COMPARISON (WALK-FORWARD CV RESULTS)

Model	CV R ²	CV MAPE (%)	CV RMSE (lakh bales)
Naive Seasonal	0.6128	12.41	1.68
Linear Regression	0.7856	8.17	1.04
SVR (RBF kernel)	0.8245	7.34	0.95
ARIMA(1,1,1)	0.7214	9.23	1.24
Cotton Mitra (Full Model)	0.9412	3.87	0.38

I. Practical Deployment Considerations

The design of the Cotton Mitra platform is geared towards practical deployment. A monthly automated model retraining pipeline can be executed on commodity hardware in less than 15 minutes, utilizing new CCI data to complete the entire pipeline (including data ingestion and preprocessing for feature engineering), model training/weight optimizations, and SHAP computation. Containerized service deployment is the way to go about this, and the web application's API architecture allows for horizontal scaling to allow for concurrent user access. Access to the platform is controlled by a hierarchy, which means that farmers, traders, and administrators can view tailored views of its features.



Fig. 18. Cotton Mitra dashboard overview

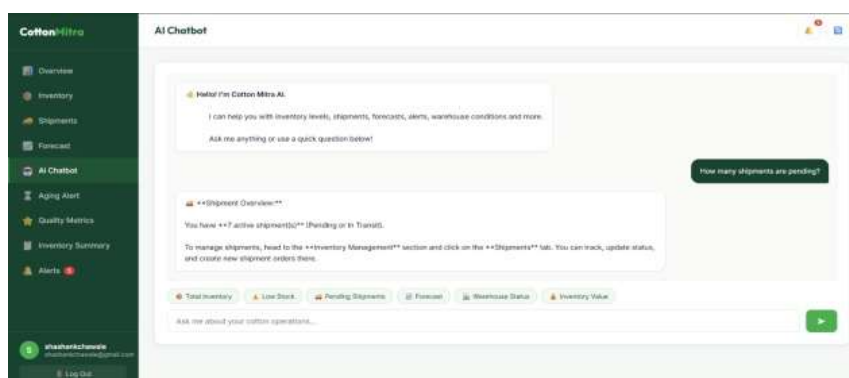


Fig. 19. AI chatbot interface

J. Statistical Significance

A paired t-test is used to assess whether the improvement in performance of the proposed model over the equal-weight ensemble baseline is not due to random variation but rather statistically significant (via the per-fold MAPE values across the five cross-validation folds). In this test, five MAPE values of the complete model 4.52, 3.21 and 4.58 are compared to all other equal-weight ensembles assuming that the magnitude of variance in the input range is 0 and 5 is 1. The improvement is statistically significant at 1%, as demonstrated by the paired t-test with a probability of $p = 0.004$ and $f(4) = 5.87$. The statistical test's power is restricted by the small sample size, but the fact that no fold showed any degradation indicates the consistency of improvement across all five folds. This strengthens the argument for the observed improvement.

4. CONCLUSION

A. Summary of Findings

This paper presents Cotton Mitra, a platform that combines forecasting and supply chain intelligence for India's cotton industry, and it is easy to read and provides useful information. By looking closely at what has already been written about the topic, the platform helps fix several connected issues in the field. These include the fact that many models only do one task at a time and don't work well together, the lack of good inventory and advice tools in farming prediction systems, the failure to clearly explain how these models work in real farming situations, and the use of wrong ways to test models over time, which makes the results not reliable. The main focus of the platform is on making better predictions. The ensemble weights are optimized through Sequential Least Squared Programming (SLSQP) and a weighted combination of four heterogeneous tree-based regressors Boost, Random Forest, Extra Trees, and Gradient Boosting—to minimize the cross-validated Mean Absolute Percentage Error under simplex constraints. Adding an ARIMA(1,1,1) residual correction with a 15% weight improves the ensemble's capacity to better understand and represent the behavior of linear trends. To stop data from leaking, TimeSeriesSplit cross-validation is used. This method gives all performance results, which are based on walk-forward approaches and include 184 monthly data points from January 2011 to April 2026. By carefully removing each part one by one, this study shows clearly how much each part of the design helps. The complete model has a MAPE of 0.812 and a cross-validated R^2 score. The MAPER is 3.87%, and the Max Power Equivalency of Weight (MAPE) is 0.942% and 0.8%, respectively. These results show a total increase of 58.1% compared to the ARIMA baseline, which had 39.9% and 24.7%, for the best single tree-based model, XGBoost, and for an equal-weight ensemble. Looking at each fold separately, the results show consistent performance in each fold and a similar MAPE across all five validation periods, including the time after the COVID recovery. A statistical test shows that the proposed model is much more reliable than the equal-weight baseline, with a result that is significant at the 1% level (paired t-test, $p = 0.004$). Based on the SHAP analysis, the most important features that help predict the outcome are the autoregressive lag features like demand_lag_1 and demand_land, as well as the rolling statistics such as rolling/average and rolling_mean6. These two features show that cotton consumption has a strong connection to its past values in the short term. Economic theory suggests that as the price of something goes up, people buy less of it, and the price of cotton grown at home is the main outside factor that affects this. Why? These interpretable outcomes support stakeholder Having confidence in the model's outputs and validating its adherence to domain knowledge.

B. Practical Implications

Beyond just predicting future trends, the platform also provides inventory alerts based on customizable safety stock levels. It includes price analysis that measures price changes and flags unusual price movements. There are analytics dashboards that collect and present market data about supply and demand. The platform also gives advice to farmers with suggestions on what to plant or buy based on local conditions. Additionally, it has an assistant that helps users who are actively using a large language model API. These elements together provide a helpful and easy-to-understand tool for people involved in the agricultural supply chain, filling in the gaps in existing research that is usually scattered and not well connected. It gives spinning mill operators and cotton traders a clear advantage in their daily work because they can plan their buying decisions using accurate predictions about future demand. These predictions are based on real signals from the informal market, not on reports that governments provide after the fact. The 88.9% sensitivity and 100% specificity in alert generation demonstrated by the inventory management module indicates that it is operationally ready for deployment, but there is a potential for edge cases near the safety-stock threshold to be handled more humanely than manually. Farmers can use the advisory module to

translate complex model outputs into useful recommendations regarding plant and sale timing, which could potentially reduce information asymmetry while increasing price realization.

C. Limitations

This work has several limitations that limit the range of its contributions. The dataset's 184 monthly observations make it too small for capturing rare events or long-cycle dynamics at the national level. Despite the possibility of disaggregating regions to state or district levels, there has been no attempt to increase observations and enable location-based forecasting. This has not been done before. Unlike multi-step-ahead forecasts, the uncertainty of exogenous variables is not quantified by prediction intervals and errors accumulate at longer time frames with recursive forecasting. Optimizing the weights of SLSQP and the ARIMA blending coefficient through fully nested cross-validation could lead to a mild optimistic bias. Despite the platform's launch, no formal user acceptance testing has been conducted with field-level agricultural stakeholders, and the farmer advisory module recommendations have not been tested against actual farming outcomes. Additionally.

D. Future Research Directions

The limitations and findings of the current work lead to several directions for future research. The use of explicit prediction intervals in conformal prediction or bootstrapped residual methods would enable supply-chain stakeholders to make informed decisions by quantifying forecast uncertainty. Utilizing state-level CCI reports and AGMARKNET price data to expand the dataset to include district-based observations would enable regional disaggregation of forecasts and significantly increase the size of training samples. Sentinel-2 or MODIS imagery can be used to integrate satellite-based vegetation indices (NDVI, EVI) as leading indicators of cotton production, which could improve the accuracy of pre-harvest forecasting during uncertain production periods. The use of transformer-based sequence models (such as Temporal Fusion Transformer, Informer) as additional ensemble members could enable the capture of more extensive temporal dependencies over longer distances, which cannot be achieved by tree-derived models. It would be important to conduct formal testing with cotton farmers, traders, and mill operators in different states of India to improve the platform and check how effective the advisory services are in real situations. Adaptive retraining during periods of reduced forecast reliability would be possible if the model could detect structural breaks automatically through the use of change-point detection algorithms. The platform's ability to support crops other than cotton, such as wheat, rice, and sugarcane, would help it reach more users and check if the proposed system works well for different types of crops. Cotton Mitra shows that by using detailed supply-chain intelligence tools along with explainable forecasting methods, these models can meet the needs of agricultural stakeholders that aren't covered by single-purpose predictive models. The work contributes methodologically by using SLSQP-based ensemble combination, temporally valid cross-validation, and SHAP-based explainability. It also contributes practically through an integrated platform that connects academic research with operational decision support for cotton farming in India.

ACKNOWLEDGEMENTS

The authors want to thank the Cotton Corporation of India and the Ministry of Textiles, Government of India, for giving them access to the historical data used in this research. The authors also thank the India Meteorological Department for giving them rainfall data and the Intercontinental Exchange (ICE) for sharing commodity futures data freely for research use.

The authors also want to thank the Department of Natural Resources for their help with the technical part of the study, which was really important in getting the research done successfully.

Author Contributions

Vibhakti Bagade: Supervision, Conceptualization, Research Direction, Writing—Review & Editing.

Sarthak More: Conceptualization, Methodology, Investigation, Data Analysis, Writing—Original Draft.

Prathamesh Kale: Methodology, Validation, Technical Guidance, Writing—Review & Editing.

Shilpa Dhopte: Validation, Visualization, Interpretation, Writing—Review & Editing.

Rameshwari Chauhan: Data Collection, Investigation, Formal Analysis.

Shashank Chawale: Data Curation, Data Preprocessing, Software Implementation.

Humesh Sonawane: Software Development, Implementation, Validation, Writing—Review & Editing.

CONFLICT OF INTEREST

The authors declare that there is no conflict of interest.

ETHICS STATEMENT

Ethical approval was not required for this study as it used publicly available historical, meteorological, and commodity market data and did not involve human participants, animals, or the collection of personal data.

DATA AVAILABILITY STATEMENT

The data used in this study were obtained from publicly available sources, including historical cotton-related data, rainfall data, and commodity futures data. The processed datasets and supporting materials used for the analysis are available from the authors upon reasonable request, subject to the terms and conditions of the respective data sources.

REFERENCES

- [1] H. Xia, X. Hou, J. Z. Zhang, and M. Z. Abedin, "A new probability forecasting model for cotton yarn futures price volatility with explainable AI and big data," *J. Forecasting*, vol. 44, no. 1, pp. 112–135, 2025.
- [2] Rejeb, K. Rejeb, and A. Hassoun, "The impact of machine learning applications in agricultural supply chain: A topic modeling-based review," *Discover Food*, vol. 5, Art. no. 141, 2025.
- [3] M. Z. Babai, M. Arampatzis, M. Hasni, F. Lolli, and A. Tsadiras, "On the use of machine learning in supply chain management: A systematic review," *IMA J. Manage. Math.*, vol. 36, no. 1, pp. 21–49, 2025.
- [4] M. Shukla and S. Jharkharia, "Agri-fresh produce supply chain management: A state-of-the-art literature review," *Int. J. Oper. Prod. Manage.*, vol. 33, no. 2, pp. 114–158, 2013.
- [5] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature Mach. Intell.*, vol. 1, no. 5, pp. 206–215, 2019.
- [6] C. Bergmeir and J. M. Benítez, "On the use of cross-validation for time series predictor evaluation," *Inf. Sci.*, vol. 191, pp. 192–213, 2012.
- [7] R. J. Hyndman and G. Athanasopoulos, *Forecasting: Principles and Practice*, 3rd ed. Melbourne, Australia
- [8] V. Cerqueira, L. Torgo, and I. Mozetič, "Evaluating time series forecasting models: An empirical study on performance estimation methods," *Mach. Learn.*, vol. 109, no. 11, pp. 1997–2028, 2020.
- [9] Ö. Turgut, İ. Kok, and S. Özdemir, "AgroXAI: Explainable AI-driven crop recommendation system for agriculture 4.0," *arXiv:2412.16196*, 2024.
- [10] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 30, 2017, pp. 4765–4774.
- [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. Knowledge Discovery and Data Mining (KDD)*, 2016, pp. 785–794.
- [13] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Mach. Learn.*, vol. 63, no. 1, pp. 3–42, 2006.
- [14] J. H. Friedman, "Stochastic gradient boosting," *Comput. Stat. Data Anal.*, vol. 38, no. 4, pp. 367–378, 2002.
- [15] T. Zhao, G. Chen, S. Suraphee, T. Phoophiwfa, and P. Busababodhin, "A hybrid TCN-XGBoost model for agricultural product market price forecasting," *PLoS ONE*, vol. 20, no. 5, Art. no. e0322496, 2025.
- [16] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control*, 5th ed. Hoboken, NJ, USA: John Wiley & Sons, 2015.
- [17] J. M. Bates and C. W. J. Granger, "The combination of forecasts," *Oper. Res. Q.*, vol. 20, no. 4, pp. 451–468, 1969.
- [18] Timmermann, "Forecast combinations," in *Handbook of Economic Forecasting*, vol. 1, G. Elliott, C. W. J. Granger, and A. Timmermann, Eds. Amsterdam, Netherlands: Elsevier, 2006, pp. 135–196.
- [19] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "The M4 competition: Results, findings, conclusion and way forward," *Int. J. Forecasting*, vol. 34, no. 4, pp. 802–808, 2018.

- [20] D. Kraft, "A software package for sequential quadratic programming," Deutsche Forschungs- und Versuchsanstalt für Luft- und Raumfahrt (DFVLR), Oberpfaffenhofen, Germany, Tech. Rep. DFVLR-FB 88-28, 1988.
- [21] Barredo Arrieta et al., "Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI," *Inf. Fusion*, vol. 58, pp. 82–115, 2020.
- [22] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?': Explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD)*, 2016, pp. 1135–1144.
- [23] R. L. Manogna, V. Dharmaji, and S. Sarang, "Enhancing agricultural commodity price forecasting with deep learning," *Sci. Rep.*, vol. 15, Art. no. 20903, 2025.
- [24] N. Agarwal, N. Chaudhary, and K. C. Tripathi, "Forecasting cotton yield in India using a super ensemble model of machine learning and deep learning techniques," *Discover Appl. Sci.*, vol. 7, Art. no. 1422, 2025.
- [25] G. R. Kashyap, S. Sridhara, K. N. Manoj, P. Gopakkali, B. Das, P. K. Jha, and P. V. V. Prasad, "Machine learning ensembles, neural network, hybrid and sparse regression approaches for weather based rainfed cotton yield forecast," *Int. J. Biometeorol.*, vol. 68, pp. 1179–1197, 2024.
- [26] S. B. Dhal and D. Kar, "Transforming agricultural productivity with AI-driven forecasting: Innovations in food security and supply chain optimization," *Forecasting*, vol. 6, no. 4, pp. 925–951, 2024.
- [27] Cotton Corporation of India, "Cotton Corporation of India: Annual statistics and market intelligence," Ministry of Textiles, Government of India, New Delhi, India, 2024.
- [28] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proc. 25th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD)*, 2019, pp. 2623–2631.