SmartShelf: AI-Powered Unsold Inventory Management with Geotagged Demand and Flash Sale Optimization

Prasad Kolte[1], Shreya Raghoji[2], Mayur Pawar[3], Sagar Zujam[4]
[1,2,3,4]*Department of Computer Science and Engineering, PCET Nutan College of Engineering and Research, Talegaon, Pune, India*

***Abstract*—Efficient inventory management is essential for maintaining profitability in retail and supply chain operations. Unsold or slow-moving products increase storage costs, block working capital, and reduce revenue. This paper presents SmartShelf, an AI-powered Unsold Inventory Management System that identifies stagnant inventory, forecasts demand using geotagged sales data, and generates region-specific flash-sale recommendations to optimize stock movement. SmartShelf integrates machine learning models (XGBoost for demand forecasting, SVM for inventory classification, and Linear Regression for interpretability) with a vendor-facing dashboard and an API bridge to an AI core. Experimental results on prototype datasets demonstrate improved inventory turnover and practical benefits for small and medium enterprises (SMEs).**

***Index Terms*—AI in Retail, Inventory Optimization, Unsold Inventory Management, Demand Forecasting, Geotagged Data Analytics, Flash Sale Automation.**

I. INTRODUCTION

Inventory is a critical asset for retail and supply chain businesses; imbalances between supply and demand can cause stockouts or excess inventory. Unsold and slow-moving stock increases holding costs, heightens obsolescence risk, and locks capital, particularly harming SMEs with tight margins. Traditional methods rely on manual analysis and heuristics, which are often reactive and fail to capture complex patterns such as seasonality and regional preferences.

This work proposes SmartShelf, an AI-driven system designed to detect unsold inventory early, forecast demand using geotagged data, and automate flash sale execution to accelerate inventory clearance and improve revenue. The system combines forecasting and classification models with a web-based dashboard and a Flask-based API bridge to the AI core, enabling practical deployment for vendors. (Report and project content used: :contentReference[oaicite:2]index=2 :contentReference[oaicite:3]index=3.)

II. RELATED WORK

AI-based demand forecasting and inventory optimization have become well-established with successful industry and academic applications. Ensemble models (Gradient Boosting, Random Forest), deep learning (RNN/LSTM, Transformers), and hybrid approaches are commonly used for forecasting and replenishment [1], [2]. Geospatial analytics and location intelligence (geo-demand heatmaps) are emerging as vital components for region-specific promotions and distribution efficiency [3], [4].

SmartShelf differentiates itself by focusing on SMEs and integrating geotagged forecasting, SVM-based classification of inventory health, and an automated flash-sale recommendation engine into a single vendor-friendly platform (source material: project report). See the references for background literature and online resources.

III. SYSTEM ARCHITECTURE

SmartShelf adopts a modular microservices-style architecture with the following main components:

IV. SYSTEM ARCHITECTURE

The SmartShelf system is designed using a modular, scalable, and service-oriented architecture that enables seamless interaction between vendors, customers, and the AI core. The architecture ensures efficient data

processing, real-time analytics, and intelligent decision-making for inventory optimization.

A. Core Components of the System

- Vendor Web Portal (Node.js / Express.js / HTML / CSS / JavaScript): Provides an interactive dashboard for vendors to manage inventory, upload product details, track sales performance, and view AI-driven insights. The portal supports secure authentication, role-based access control, and real-time updates using RESTful APIs.
- Customer Interface (Web/Mobile): Enables customers to browse products, receive personalized flash-sale notifications, view discounts, and place orders. The interface dynamically updates offers based on regional demand and inventory availability.
- AI Core (Python Microservice): Implements machine learning models such as XGBoost for demand forecasting, Support Vector Machine (SVM) for inventory classification, and Linear Regression for interpretability. The AI core also performs geospatial analytics to identify region-specific demand patterns and generates flash sale recommendations.
- Database Layer (MySQL): Stores structured data including product catalogs, inventory levels, historical sales transactions, vendor profiles, promotions, and notification logs. The relational schema ensures data consistency and efficient query performance.
- REST API Bridge (Flask + Express.js): Facilitates JSON-based communication between the Node.js backend and the Python AI core. This layer supports both synchronous requests (on-demand predictions) and asynchronous processing (batch analysis and scheduled model updates).
- Notification Service (Email / Push / SMS): Sends automated alerts to vendors and customers regarding flash sales, low stock warnings, demand spikes, and promotional events.

B. Workflow Overview

The operational workflow of the system is as follows:

1) Vendors upload inventory and sales data through the Node.js-based portal.
2) Data is stored in the MySQL database and forwarded to the AI core for preprocessing.
3) Machine learning models analyze demand trends and classify inventory risk.
4) Flash sale recommendations and pricing strategies are generated.
5) Results are pushed back to the vendor dashboard and customer interface in real time.

This architecture ensures scalability, fault tolerance, and efficient integration of AI-driven intelligence into real-world retail workflows.

C. Data pipeline

The SmartShelf data pipeline follows:

1) Data ingestion from vendor uploads and POS integrations.
2) Preprocessing and feature engineering (include geotags, shelf-age, discount history).
3) Model inference: forecasting and classification.
4) Action generation: flash sale proposals, markdown suggestions, notifications.
5) Monitoring and feedback: track performance of promotions and refine models.

V. MACHINE LEARNING MODELS

SmartShelf employs complementary models to solve forecasting and classification problems:

A. XGBoost for demand forecasting

XGBoost is used as the primary forecasting model due to its speed and strong predictive performance. Features include historical daily/weekly sales, product attributes (category, price, discounts), temporal indicators (season, holidays), and geotag features (store or region id).

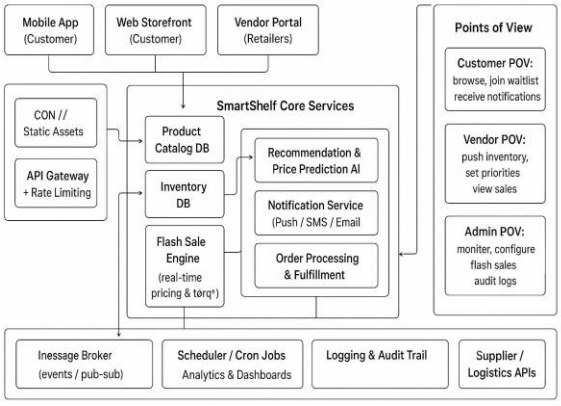

Fig. 1: Conceptual block diagram of SmartShelf (architecture, data flow and major components).

B. SVM for inventory classification

SVM classifies SKUs into fast-moving, moderately-moving, and slow-moving based on sales velocity, stock age, discount history, and regional demand indices. Items flagged as slowmoving are prioritized for clearance actions such as targeted flash sales.

C. Linear Regression for interpretability

Linear Regression is used as a lightweight baseline to interpret the effect of individual features (e.g., price or discount) on demand. This interpretability helps vendor decision-makers to understand the magnitude of change expected from pricing adjustments.

VI. IMPLEMENTATION DETAILS

A. Frontend and Dashboard

The SmartShelf frontend is implemented using modern web technologies including HTML, CSS, JavaScript, and is powered by a Node.js backend using the Express.js framework. The vendor dashboard provides an interactive interface to manage products, monitor inventory health, and visualize AIdriven insights. Key features include real-time stock status, demand forecasting charts, and geotagged demand heatmaps that highlight regional sales patterns.

Vendors can review AI recommendations for flash sales, configure discount parameters, and approve promotional campaigns directly through the dashboard. The frontend communicates with backend services using RESTful APIs and dynamically updates charts and alerts to support data-driven decisionmaking.

B. AI Core and API Layer

The AI Core is developed as a Python-based microservice responsible for demand forecasting, inventory classification, and recommendation generation. Machine learning models such as XGBoost, Support Vector Machine (SVM), and Linear Regression are trained on historical sales data and geotagged features to predict demand trends and identify slow-moving products.

The AI Core exposes RESTful endpoints using the Flask framework. These endpoints are consumed by the Node.js backend for real-time predictions and batch processing. Trained models are serialized and loaded into memory to ensure lowlatency inference, enabling timely responses for flash sale recommendations and inventory risk alerts.

C. Backend Services and API Integration

The backend layer is built using Node.js with Express.js, acting as the central orchestrator between the frontend, AI Core, and database services. It handles authentication, role-based access control (vendor, admin, customer), request validation, and business logic processing.

The Node.js backend communicates with the Python AI microservice through JSON-based REST APIs. This separation of concerns ensures scalability, maintainability, and ease of deployment. Asynchronous processing is supported to handle high-volume prediction requests without blocking user interactions.

D. Data Storage and Messaging

A MySQL database is used to store structured data including product catalogs, inventory levels, sales transactions, promotions, and notification logs. Proper indexing and relational constraints ensure data consistency and efficient query performance.

For real-time event handling such as flash sale activation, inventory threshold alerts, and customer notifications, an eventdriven messaging system can be integrated. Technologies such as Redis Pub/Sub or RabbitMQ enable asynchronous communication between services, improving system responsiveness and scalability.

Overall, the integration of Node.js, Express.js, Python-based

AI services, and a relational database enables SmartShelf to operate as a robust, scalable, and intelligent inventory management platform suitable for modern retail environments.

VII. EXPERIMENTAL OVERVIEW AND RESULTS

A prototype of SmartShelf was evaluated on sample retail datasets (historical sales store metadata). After preprocessing and splitting, models were trained and evaluated:

- XGBoost produced lower RMSE compared to baseline Linear Regression on short-horizon forecasts.

- SVM classified slow-moving items with high precision, effectively surfacing items for targeted promotions.
- Geotagged heatmaps helped identify regions with latent demand and increased conversion rates after targeted flash sales.

These preliminary resultsvalidatethe feasibility of SmartShelf and its usefulness to SMEs for reducing inventory wastage and improving turnover (summary derived from the project report). See Section VII for limitations and future expansions.

VIII. DISCUSSION AND FUTURE WORK

SmartShelf demonstrates the value of combining forecasting, classification and geospatial analytics to manage unsold inventory. Key future enhancements include:

- Dynamic price optimization: Reinforcement learning or optimization frameworks for price adjustments during flash sales.
- Supplier integration: Automated PO generation and coordination with supplier lead times.
- IoT integration: Smart-shelf sensors (weight, RFID) for automated stock updates.
- Richer external signals: Integrate social data, weather and events for better demand context.
- Scalability: Cloud-native deployment for multi-warehouse and enterprise usage.

IX. CONCLUSION

We presented SmartShelf, an AI-driven system to detect unsold inventory, forecast demand with geotagged data, and automate flash-sale recommendations to optimize stock movement. The system's combination of XGBoost, SVM, and explainable Linear Regression provides a practical, vendor-friendly solution for SMEs. Prototype evaluations show promising improvements in forecasting accuracy and inventory prioritization, and the proposed future work aims to make SmartShelf a scalable, production-ready platform.

(Report sources: :contentReference[oaicite:4]index=4 :contentReference[oaicite:5]index=5.)

REFERENCES

[1] C. Sekhar, "Optimizing Retail Inventory Management with AI: A Predictive Approach to Demand Forecasting, Stock Optimization, and Automated Reordering," 2022.

[2] "Complete Guide to Machine Learning in Retail Demand Forecasting," RELEX Solutions, 2024.

[3] "Using Location Intelligence for Retail Demand Forecasting," Kalibrate, accessed Nov. 2025. [Online]. Available: https://kalibrate.com

[4] "Supercharging Retail Sales Through Geospatial Analytics," McKinsey & Company, accessed Nov. 2025.

[5] "AI in Inventory Management: Top Use Cases You Need To Know," SmartDev, accessed Nov. 2025.