International Journal of Computational Research in Science and Technology
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

Cloud-Based Intelligent Replenishment System Using Machine Learning and Business Analytics

[1]Dr. Pankaj Malik, [2]Bhumi Wadhwani, [3]Pushkar Rathore,
[4]Miraj Chourasiya, [5]Stuti Bhati, [6]Pushkar Bhati
Computer Science Engineering, Medicaps University, Indore, India[1,2,3,4,5,6]

Abstract. Efficient inventory replenishment is essential for minimizing stock-outs, excess inventory, and operational costs in modern retail and supply-chain environments. Traditional replenishment approaches generally rely on fixed reorder points and historical averages, which may not adequately capture nonlinear demand patterns, seasonal variations, promotional effects, and changing lead times. This paper proposes a Cloud-Based Intelligent Replenishment System (CBIRS) that integrates machine learning, cloud computing, and business analytics to enable dynamic and data-driven inventory replenishment. The proposed framework utilizes cloud-based data ingestion and storage to integrate historical sales, product, pricing, promotion, inventory, seasonal, and supplier lead-time information. Multiple machine learning models, including Random Forest, XGBoost, LightGBM, CatBoost, and Long Short-Term Memory (LSTM), are evaluated for demand forecasting, followed by a hybrid forecasting model for intelligent replenishment decisions. The predicted demand is integrated with safety-stock estimation, reorder-point calculation, lead-time analysis, and dynamic order-quantity optimization. Experimental evaluation demonstrates that the proposed hybrid model achieves an MAE of 9.72, RMSE of 15.63, MAPE of 7.86%, and R^2 of 0.94, outperforming the individual baseline models. At the inventory-management level, the proposed CBIRS reduces the stock-out rate to 4.2% and overstock rate to 8.3%, while achieving an estimated 21.1% reduction in total inventory cost compared with the static replenishment approach and a 95.8% service level. The results indicate that integrating machine learning forecasting with cloud-based analytics and dynamic replenishment can significantly improve inventory availability, cost efficiency, and supply-chain decision-making.

Keywords: Cloud Computing, Machine Learning, Business Analytics, Demand Forecasting, Intelligent Replenishment, Inventory Optimization, XGBoost, LSTM, Supply Chain Management.

I. Introduction

Inventory management is a critical component of modern supply-chain operations because organizations must maintain sufficient product availability while minimizing inventory-related costs. In retail, e-commerce, manufacturing, and distribution environments, inappropriate replenishment decisions can lead to stock-outs, excess inventory, increased holding costs, and reduced customer satisfaction. The increasing availability of transactional and operational data has created an opportunity to replace conventional rule-based inventory practices with intelligent, predictive, and data-driven replenishment systems.

Traditional inventory replenishment approaches commonly rely on fixed reorder points, predefined safety-stock levels, moving averages, or manually determined order quantities. Although these methods are relatively simple and computationally inexpensive, they may not adequately respond to complex demand behavior. Product demand can be affected by historical purchasing patterns, price changes, promotional campaigns, seasonal variations, holidays, product categories, store location, and supplier lead times. Consequently, a replenishment policy based solely on historical averages may either underestimate future demand, resulting in stock-outs, or overestimate demand, resulting in unnecessary inventory accumulation.

Machine learning (ML) provides an effective approach for modeling complex relationships between demand and multiple business variables. Ensemble algorithms such as Random Forest, XGBoost, LightGBM, and CatBoost can capture nonlinear interactions among product, pricing, promotional, and temporal features. In addition, deep learning models such as Long Short-Term Memory (LSTM) networks can learn temporal dependencies from sequential sales data. However, demand forecasting alone does not directly solve the inventory replenishment problem. A forecast must be transformed into an operational decision that determines when to replenish and how much inventory to order while considering safety stock, supplier lead time, service-level requirements, and inventory costs.

Cloud computing provides an appropriate infrastructure for integrating these forecasting and replenishment activities. Cloud-based platforms enable centralized storage, scalable computation, distributed data processing, model deployment, and access to real-time business information from multiple stores, warehouses, and sales channels. By combining cloud computing with ML and business analytics, organizations can develop centralized intelligent replenishment systems capable of continuously processing new data and generating updated inventory recommendations.

Business analytics further enhances the proposed approach by converting machine learning predictions into actionable key performance indicators (KPIs). Instead of presenting only predicted demand, an intelligent replenishment platform can provide information regarding stock-out risk, excess inventory, inventory turnover, service level, holding cost, and recommended order quantities. Such analytics can assist inventory managers in identifying critical products and making timely procurement decisions.

Despite the progress in ML-based demand forecasting, an important research gap remains in the integration of cloud-based data management, machine learning forecasting, dynamic inventory optimization, and business decision support within a unified framework. Many existing forecasting approaches concentrate primarily on prediction accuracy and do not sufficiently evaluate the effect of forecasting on practical inventory performance. Similarly, conventional replenishment systems often use static policies that do not dynamically adapt to predicted demand and changing operational conditions.

To address these limitations, this paper proposes a Cloud-Based Intelligent Replenishment System (CBIRS) that combines cloud computing, machine learning, inventory optimization, and business analytics. The system first collects and integrates sales, inventory, product, pricing, promotion, seasonal, and lead-time information through a cloud-based data pipeline. After preprocessing and feature engineering, multiple ML models are applied to predict future product demand. The predicted demand is subsequently incorporated into safety-stock estimation, reorder-point calculation, and dynamic order-quantity determination. A business analytics layer provides inventory monitoring and decision-support capabilities.

The major contributions of this research are as follows:

- A cloud-enabled intelligent architecture for integrating heterogeneous sales, inventory, product, and supply-chain data for replenishment analysis.
- A machine learning-based demand forecasting framework that evaluates Random Forest, XGBoost, LightGBM, CatBoost, and LSTM models for product-level demand prediction.
- A dynamic replenishment mechanism that combines predicted demand with lead time, safety stock, reorder point, and current inventory to generate adaptive order recommendations.
- A business analytics layer for monitoring stock-out risk, overstock conditions, service level, inventory cost, and other operational KPIs.
- An explainable machine learning component that can identify important factors contributing to demand predictions, thereby improving transparency for inventory managers.
- An integrated evaluation framework that considers both forecasting performance and inventory-management performance, rather than evaluating the ML model solely on prediction accuracy.

II. Related Work

Demand forecasting and inventory management have traditionally relied on statistical techniques such as moving averages, exponential smoothing, autoregressive models, and other time-series forecasting methods. Although these approaches are effective for relatively stable demand patterns, their performance can decrease when demand is affected by nonlinear relationships, promotions, seasonality, customer behavior, and external factors. Recent research has therefore increasingly investigated machine learning (ML) and deep learning (DL) techniques for demand forecasting and inventory optimization [1], [2].

Seyam et al. [1] presented a systematic review of ML and DL applications for demand forecasting across time-critical industries. Their study highlights the increasing use of data-driven forecasting approaches and demonstrates that ML/DL models can capture complex demand patterns that are difficult to model using conventional statistical techniques. Similarly, Ghazi et al. [2] reviewed ML and DL models for supply-chain demand forecasting and emphasized the importance of selecting forecasting techniques according to the characteristics of the data, forecasting horizon, and application environment.

Several studies have investigated machine learning specifically for retail demand forecasting. A multi-channel retailing study proposed the use of CatBoost for forecasting product demand and compared it with XGBoost and linear regression [3]. The study used daily sales data from multiple retail branches and demonstrated the applicability of ML-based forecasting for inventory management. The results indicate that machine learning can support demand estimation in environments where multiple sales channels and product-level variations create complex inventory requirements.

Ensemble learning has also been investigated for demand forecasting and inventory optimization. Samal and Ghosh [4] examined ensemble-based predictive analytics for multi-channel retailing and highlighted the difficulty of forecasting demand when seasonality, promotional activities, and interactions among sales channels influence purchasing patterns. Ensemble models are particularly useful because they can combine complementary predictive capabilities and improve robustness compared with individual models.

The relationship between demand forecasting and inventory optimization has been studied using deep learning approaches. An order-up-to-level inventory optimization framework based on ensemble deep learning was proposed to forecast future demand and estimate safety stock from the forecasted demand distribution [5]. The study demonstrates that demand forecasting should not be considered independently from inventory decisions because forecast uncertainty directly influences safety-stock requirements, service levels, and replenishment quantities.

Machine learning has also been applied directly to e-commerce inventory management. An IEEE study investigated ML techniques for demand forecasting and inventory optimization in e-commerce environments, emphasizing the potential of data-driven models to improve inventory decision-making [6]. Another IEEE study proposed ML-based demand forecasting for multi-channel retail stores and demonstrated the use of product-level sales data to predict short- and medium-term demand [3]. These studies establish the relevance of ML for inventory management but primarily focus on forecasting and optimization rather than integrating a complete cloud-based business analytics architecture.

Recent research has increasingly explored intelligent and automated inventory management. Rao et al. [7] proposed an intelligent inventory-management approach integrating real-time stock monitoring, demand forecasting, and automated decision support. Their work demonstrates the importance of combining sales, pricing, temporal, and seasonal variables when developing inventory prediction systems. However, further integration with scalable cloud infrastructure and explainable business analytics remains an important research direction.

The integration of demand forecasting and inventory decisions has also been investigated using reinforcement learning. Yang et al. [8] proposed a multi-agent deep reinforcement learning framework that jointly addresses demand forecasting and inventory optimization using sensor-enabled retail supply-chain information. Their work illustrates a transition from independent forecasting models toward integrated decision-making systems capable of dynamically adjusting inventory policies. However, reinforcement-learning approaches can introduce considerable computational and deployment complexity, particularly when applied to large numbers of products and stores.

A recent systematic review of ML approaches in inventory control classified existing approaches into three broad categories: separate forecasting and optimization, static ML-integrated optimization, and dynamic ML-integrated optimization using reinforcement learning [9]. The review identified several remaining challenges, including scalability, large action spaces, stochastic lead times, multi-echelon inventory systems, and real-world deployment. These findings indicate that an effective inventory system should not only improve prediction accuracy but also provide scalable mechanisms for converting predictions into operational replenishment decisions.

Cloud-native architectures have recently emerged as an important direction for deploying demand forecasting systems at scale. Kambi [10] presented an end-to-end cloud-native ML architecture for retail demand forecasting using cloud storage, serverless analytics, feature management, model training, prediction, and monitoring components. Such architectures demonstrate how cloud computing can support scalable data processing and production ML workflows. Nevertheless, the focus remains primarily on demand forecasting and MLOps rather than integrating forecasting with dynamic replenishment optimization and business-level inventory analytics.

Overall, the existing literature demonstrates significant progress in three individual areas: machine learning-based demand forecasting, inventory optimization, and cloud-based analytics. However, these components are frequently investigated separately. A research opportunity therefore exists to develop an integrated system that combines cloud-based data management, ML demand forecasting, dynamic replenishment, explainable AI, and business analytics within a single decision-support framework.

The proposed Cloud-Based Intelligent Replenishment System (CBIRS) addresses this gap by integrating these components into a unified architecture. Unlike forecasting-only approaches, the proposed framework transforms predicted demand into actionable replenishment decisions using safety stock, reorder points, lead time, and dynamic order quantities. In addition, the business analytics layer provides inventory KPIs and decision-support information, while the proposed explainability component enables managers to understand the major factors influencing demand predictions.

III. Research Gap

The review of existing literature indicates that significant progress has been made in machine learning-based demand forecasting, inventory optimization, cloud computing, and business intelligence. However, these research areas are often investigated independently or only partially integrated. Existing studies demonstrate that machine learning models can improve demand prediction, while inventory optimization techniques can determine appropriate stock levels and replenishment policies [1]–[5]. Nevertheless, several important research gaps remain.

1. Limited integration of forecasting and replenishment decisions
A large proportion of existing studies concentrate on improving demand forecasting accuracy using machine learning or deep learning models [1], [2]. Although accurate demand prediction is important, the forecast is not always directly transformed into an operational replenishment decision. Consequently, there remains a need for an integrated framework that converts predicted demand into dynamic reorder points, safety-stock levels, and order quantities.

2. Predominance of static inventory policies
Traditional inventory systems frequently use predefined reorder points and fixed safety-stock levels. Such approaches may not respond effectively to rapidly changing demand, promotional activities, seasonality, and variations in supplier lead time. Existing ML-based inventory studies have demonstrated improvements over conventional approaches, but adaptive replenishment based on continuously updated demand forecasts remains insufficiently explored [5], [6].

3. Insufficient cloud-based integration
Cloud computing provides scalable storage, distributed processing, centralized data management, and model deployment capabilities. Recent research has investigated cloud-native architectures for retail demand forecasting and machine learning operations [7]. However, comparatively limited research integrates cloud-based data management with demand forecasting, inventory optimization, and automated replenishment within a single architecture.

4. Limited incorporation of business analytics

Many existing studies evaluate models primarily using forecasting metrics such as MAE, RMSE, MAPE, and [1], [2]. These metrics indicate prediction quality but do not directly measure business outcomes. Inventory managers are more concerned with operational indicators such as stock-out rate, overstock rate, service level, inventory turnover, holding cost, and total inventory cost. Therefore, there is a need to evaluate machine learning models using both predictive and business-oriented performance measures.

5. Lack of explainability in replenishment decisions:

Machine learning models such as gradient-boosting algorithms and deep neural networks can provide accurate predictions but may behave as black-box systems. In practical business environments, inventory managers need to understand the factors responsible for predicted demand changes. Explainable AI techniques such as SHAP can provide feature-level explanations, but their integration into cloud-based intelligent replenishment systems remains relatively limited.

6. Limited multi-source business-data integration

Demand is influenced by multiple variables, including historical sales, price, promotions, seasonality, holidays, product category, store location, inventory status, and supplier lead time. Several existing forecasting studies rely primarily on historical sales or a limited set of explanatory variables [2], [3]. A more comprehensive framework is required to integrate heterogeneous business data and exploit their combined predictive value.

7. Inadequate evaluation of end-to-end inventory performance:

Existing research frequently reports improvements in forecasting accuracy without determining whether these improvements result in lower inventory costs or improved product availability. A forecasting model with lower prediction error does not necessarily produce the best replenishment policy. Therefore, an end-to-end evaluation from data acquisition through forecasting, replenishment, and business outcomes is required.

Based on these observations, the primary research gap can be summarized as follows:

- Existing research lacks a unified, cloud-enabled, explainable, and business-oriented framework that integrates machine learning-based demand forecasting with dynamic inventory replenishment and evaluates the resulting decisions using both predictive and operational business metrics.
- To address this gap, this research proposes the Cloud-Based Intelligent Replenishment System (CBIRS). The proposed framework integrates cloud-based data ingestion and storage, machine learning demand forecasting, dynamic safety-stock and reorder-point estimation, adaptive order-quantity calculation, explainable AI, and business analytics into a unified decision-support architecture. The framework is designed to continuously transform historical and current business data into demand predictions and subsequently into actionable replenishment recommendations.

The research gap and corresponding proposed solutions are summarized in Table I.

Table 1: Research Gaps and Proposed Solutions

Existing Limitation	Research Gap	Proposed CBIRS Solution
Forecasting performed independently	Forecast not directly linked to inventory decisions	Integrated forecasting and replenishment
Fixed reorder policies	Poor adaptation to changing demand	Dynamic reorder point
Static safety stock	Does not reflect forecast variability	Forecast-driven safety stock
Limited cloud integration	Scalability and centralized processing limitations	Cloud-based architecture
Forecasting-focused evaluation	Business impact often ignored	Inventory and business KPIs
Black-box ML models	Limited decision transparency	SHAP-based explainability
Limited business variables	Incomplete demand representation	Multi-source feature integration
Model-level evaluation	Limited end-to-end validation	Forecast-to-replenishment evaluation

Thus, the proposed CBIRS aims to move beyond conventional "forecasting-only" approaches toward an integrated "predict–optimize–recommend" framework in which machine learning predictions directly support inventory decisions and cloud-based business analytics provides continuous operational visibility.

III. Proposed System

The proposed Cloud-Based Intelligent Replenishment System (CBIRS) is an integrated decision-support framework that combines cloud computing, machine learning, inventory optimization, explainable artificial intelligence (XAI), and business analytics to generate dynamic product replenishment recommendations. The primary objective of CBIRS is to transform heterogeneous business data into accurate demand forecasts and subsequently convert those forecasts into actionable inventory decisions.

Unlike conventional inventory systems that depend on fixed reorder points or manually defined replenishment rules, the proposed system dynamically adjusts replenishment decisions according to predicted demand, inventory availability, supplier lead time, demand variability, and business constraints.

The proposed system consists of six major layers:

- Data Acquisition Layer
- Cloud Data Management Layer
- Data Processing and Feature Engineering Layer
- Machine Learning Forecasting Layer
- Intelligent Replenishment Layer
- Business Analytics and Visualization Layer

1. Overall System Architecture

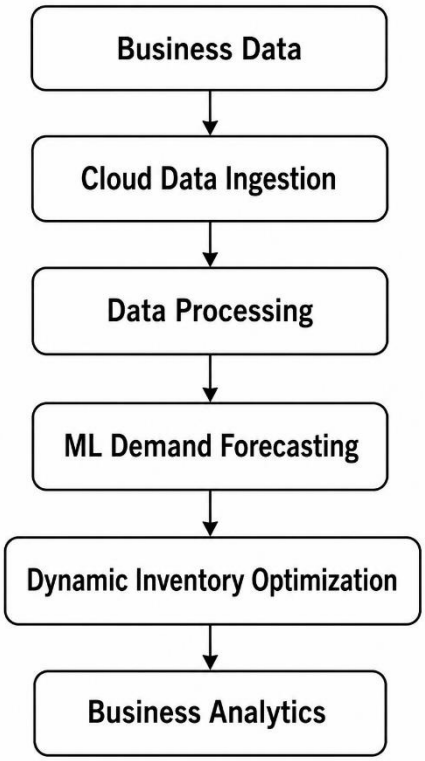


Fig 1 The proposed system follows the workflow:

International Journal of Statistics and Mathematical Research Trends
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

The major input sources include:

- Historical sales transactions
- Current inventory
- Product information
- Product price
- Promotional information
- Seasonal information
- Holiday information
- Supplier lead time
- Store/location information

The resulting system produces:

- Future demand prediction
- Stock-out risk
- Overstock risk
- Safety-stock recommendation
- Dynamic reorder point
- Recommended order quantity
- Inventory-performance indicators

2. Data Acquisition Layer

The Data Acquisition Layer collects information from heterogeneous enterprise systems. Depending on the business environment, the data can originate from point-of-sale (POS) systems, enterprise resource planning (ERP) systems, warehouse-management systems, e-commerce platforms, supplier databases, and customer transaction systems.

For product at time, the input feature vector can be represented as:

$$X_{p,t} = [S_{p,t}, P_{p,t}, P, r_{p,t}, I_{p,t}, L_p, R_t, H_t C_p]$$

where:

- $S_{p,t}$= historical sales
- $P_{p,t}$= promotional information
- $Pr_{p,t}$= product price
- $I_{p,t}$= current inventory
- L_p= supplier lead time
- R_t= seasonal information
- H_t= holiday information
- C_p= product category

The acquisition layer is designed to support both batch and near-real-time data ingestion.

3. Cloud Data Management Layer

The collected data are transferred to a cloud-based storage and processing environment. The cloud layer provides centralized storage and scalable computational resources for handling large volumes of product and transaction data.

The cloud data-management layer consists of:

- Cloud Data Lake: Stores raw and semi-structured business data.
- Cloud Data Warehouse: Stores cleaned and structured analytical data.
- Data Processing Service: Performs distributed preprocessing and feature generation.
- Feature Store: Maintains reusable ML features.
- Model Repository: Stores trained forecasting models and model versions.

The cloud-based architecture can be represented as:

This architecture allows the proposed system to scale from a single store to multiple stores, warehouses, products, and sales channels.

4. Data Preprocessing and Feature Engineering

Raw business data may contain missing values, duplicate records, inconsistent formats, abnormal observations, and incomplete transaction information. Therefore, preprocessing is performed before model training.

The preprocessing process consists of:

- Missing-value treatment
- Duplicate-record removal
- Outlier detection
- Data normalization where required
- Temporal feature extraction
- Categorical feature encoding
- Feature selection

Important forecasting features include:

- Previous-day demand
- Previous-week demand
- Previous-month demand
- Rolling mean demand
- Rolling demand standard deviation
- Price
- Discount
- Promotion
- Month
- Week
- Day of week
- Holiday indicator
- Lead time
- Current inventory

For a time window of observations, the rolling demand can be calculated as:

$$RM_t = \frac{1}{k}\sum_{i=1}^{k} D_{t-i}$$

where represents demand at time.

The rolling standard deviation can additionally be used to estimate demand uncertainty for safety-stock calculation.

International Journal of Statistics and Mathematical Research Trends
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

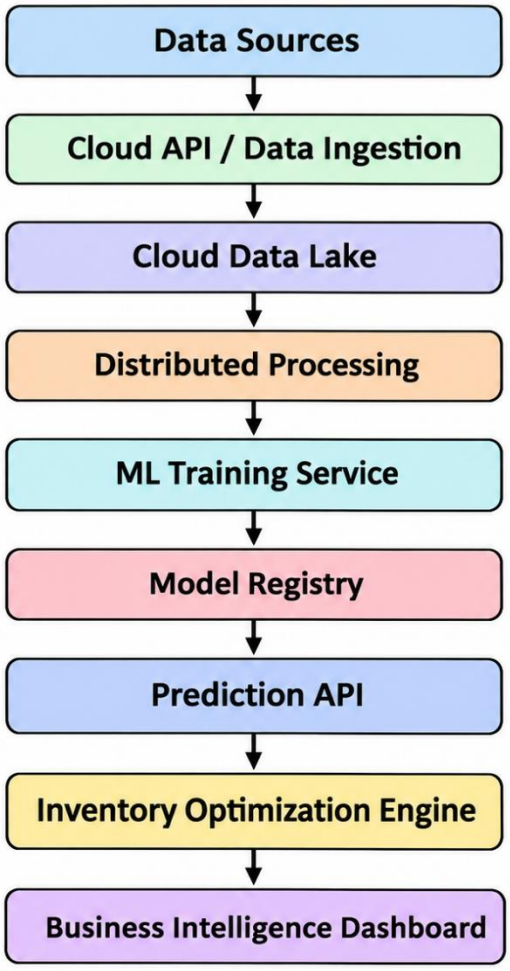


Fig. 2. Cloud-Based Data Processing Pipeline

5. Machine Learning Demand Forecasting Layer

The Machine Learning Layer predicts future product demand using historical and contextual business features.

The proposed framework evaluates multiple ML and deep-learning models:

- Random Forest (RF)
- XGBoost
- LightGBM
- CatBoost
- Long Short-Term Memory (LSTM)

The general forecasting problem is formulated as:

$$\widehat{D}_{t+h} = f(X_t)$$

where:

- $\widehat{D}_{t+h}$= predicted demand at future time $t + h$
- X_t= feature vector available at time t
- f= trained machine learning model
- h= forecasting horizon

Tree-based models are used to capture nonlinear relationships among business variables, whereas LSTM is employed to capture temporal dependencies in sequential demand data.

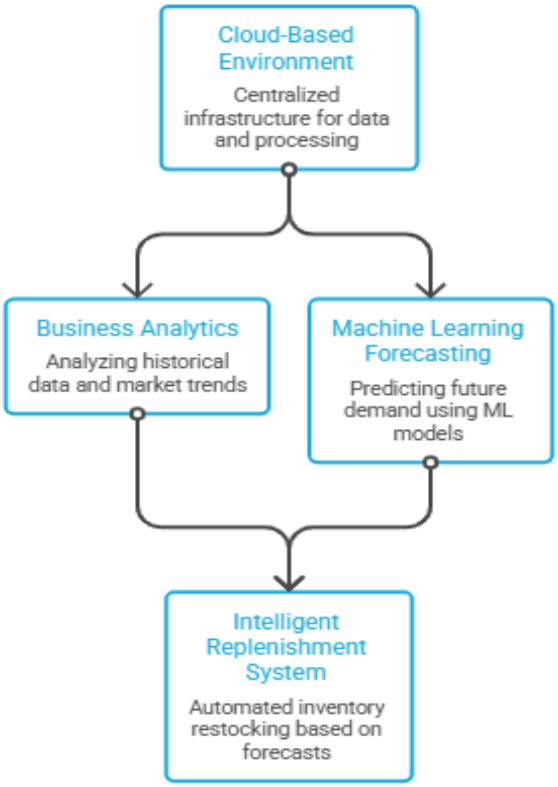


Fig. 3 Proposed ML Demand Forecasting Framework

6. Proposed Hybrid Forecasting Strategy

To exploit the complementary strengths of tree-based ML and deep learning, CBIRS can employ a hybrid forecasting strategy.

The LSTM component learns temporal demand characteristics:

$$F_T = LSTM(X_t)$$

The XGBoost component learns nonlinear relationships between demand and business variables:

$$F_B = XGBoost(X_t)$$

The final forecast is obtained through weighted fusion:

$$\hat{D}_{t+h} = w_T F_T + w_B F_B$$

subject to:

$$w_T + w_B = 1$$

The weights are selected using validation performance.

This hybrid strategy is intended to capture both temporal patterns and nonlinear business relationships.

7. Demand Uncertainty Estimation

Accurate replenishment requires not only expected demand but also an estimate of demand uncertainty.
Let the predicted demand be:

$$\hat{D}$$

and the forecasting error standard deviation be:

International Journal of Statistics and Mathematical Research Trends
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

$$\sigma_D$$

The uncertainty estimate is used to determine an appropriate safety-stock level.

This allows the system to increase inventory protection when demand is highly variable and reduce unnecessary safety stock when demand is relatively stable.

8. Intelligent Replenishment Engine

The Intelligent Replenishment Engine is the central decision-making component of CBIRS. It transforms demand forecasts into inventory actions.

The system continuously evaluates:

$$Inventory_{current}$$

against the dynamically calculated reorder point.

The reorder point is defined as:

$$ROP = (\hat{D} \times L) + SS$$

where:

- $\hat{D}$= predicted average demand per unit time
- L= supplier lead time
- SS= safety stock

If:

$$Inventory_{current} \leq ROP$$

the system identifies the product as requiring replenishment.

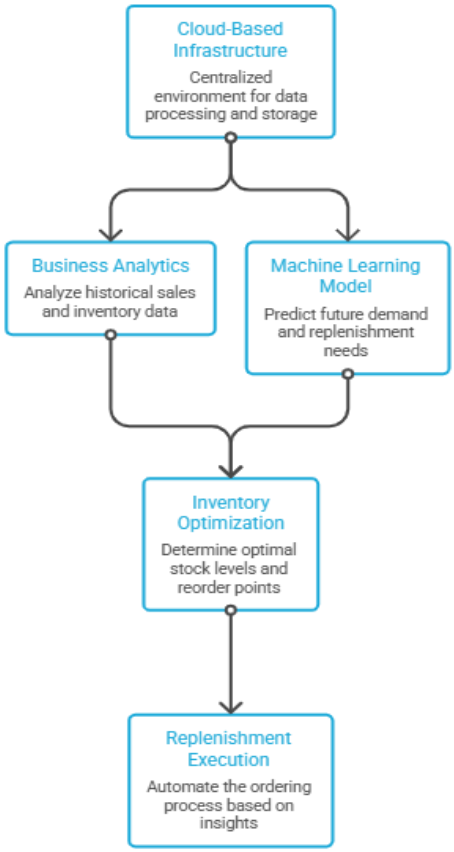


Fig. 4. Dynamic replenishment decision workflow

9. Dynamic Safety Stock

Safety stock is dynamically calculated based on forecast uncertainty and the required service level:

$$SS = Z\sigma_D\sqrt{L}$$

where:

- Z= service-level coefficient
- σ_D= demand standard deviation
- L= lead time

Unlike a fixed safety-stock policy, this approach allows safety stock to change according to demand variability and lead time.

10. Dynamic Order Quantity

After determining that replenishment is required, the system calculates the recommended order quantity.
A simplified formulation is:

$$Q_{order} = \max\left(0, \hat{D}_H + SS - I_{current}\right)$$

where:

- $\hat{D}_H$= predicted demand over the planning horizon
- SS= required safety stock
- $I_{current}$= current inventory

The resulting value represents the minimum additional inventory required to cover expected demand and maintain the desired protection level.

Business constraints such as minimum order quantity, supplier capacity, storage capacity, and budget can subsequently be incorporated.

11. Inventory Risk Classification

CBIRS classifies products into different inventory-risk categories based on predicted demand and current inventory.
A simplified risk score can be defined as:

$$R_{stock} = \frac{\hat{D}_H + SS - I_{current}}{\hat{D}_H + SS + \epsilon}$$

where $\epsilon$is a small constant used to prevent division by zero.

Products can subsequently be classified as:

Table 2: This classification allows managers to prioritize critical products.

Risk Category	Condition
Critical	Severe stock-out risk
High	Replenishment required immediately
Medium	Replenishment required soon
Low	Inventory adequate
Excess	Inventory substantially above predicted requirement

12. Explainable AI Module

The proposed system incorporates explainable artificial intelligence to improve transparency in ML-based demand predictions.

SHAP values are used to estimate the contribution of individual features:

$$f(x) = \phi_0 + \sum_{i=1}^{n} \phi_i$$

where $\phi_i$represents the contribution of feature i.

For example, the system may explain a high-demand prediction using factors such as:

- Increased historical sales
- Active promotion

- Seasonal demand
- Reduced product price
- Holiday period

This information enables inventory managers to understand why a replenishment recommendation was generated.

13. Business Analytics Layer
The final layer converts predictions and replenishment decisions into business intelligence.

The proposed dashboard provides:
- Demand forecast
- Actual vs. predicted demand
- Current inventory
- Predicted inventory requirement
- Reorder point
- Recommended order quantity
- Stock-out risk
- Overstock risk
- Service level
- Inventory turnover
- Inventory cost

The dashboard can also provide product-level, store-level, category-level, and regional analytics.

14. End-to-End Decision Workflow
The complete decision process can be summarized as:

The system continuously updates this workflow as new sales and inventory information becomes available.

15. Algorithmic Procedure
Algorithm 1: Cloud-Based Intelligent Replenishment
Input: Sales data, inventory data, product data, price, promotion, seasonality, lead time
Output: Demand forecast and replenishment recommendation
- Collect business data from multiple sources.
- Upload data to cloud storage.
- Clean and preprocess the collected data.
- Generate temporal and business-related features.
- Split data chronologically into training, validation, and testing sets.
- Train RF, XGBoost, LightGBM, CatBoost, and LSTM models.
- Evaluate forecasting performance using MAE, RMSE, MAPE, and .
- Select or construct the best-performing forecasting model.
- Generate future demand prediction .
- Estimate demand uncertainty .
- Calculate dynamic safety stock.
- Calculate the reorder point.
- Compare current inventory with the reorder point.
- If current inventory is below the reorder point, calculate the recommended order quantity.
- Classify the inventory risk.
- Generate SHAP-based explanations for the prediction.
- Store results in the cloud analytics database.
- Display recommendations and KPIs on the business dashboard.

- Repeat the process when new business data become available.

IV. Data Preprocessing

Data preprocessing is a critical stage in the proposed Cloud-Based Intelligent Replenishment System (CBIRS) because the quality of input data directly affects demand forecasting accuracy and subsequent inventory decisions. Retail and supply-chain datasets are generally collected from multiple heterogeneous sources, including point-of-sale (POS) systems, enterprise resource planning (ERP) systems, warehouse-management systems, e-commerce platforms, and supplier databases. These sources may contain missing values, duplicate transactions, inconsistent data formats, abnormal sales observations, and categorical variables. Therefore, a systematic preprocessing pipeline is implemented before applying machine learning models.
The proposed preprocessing pipeline consists of data integration, data cleaning, missing-value treatment, duplicate removal, outlier detection, temporal feature extraction, categorical encoding, feature engineering, feature scaling, and chronological dataset partitioning.

The complete preprocessing workflow is represented as:

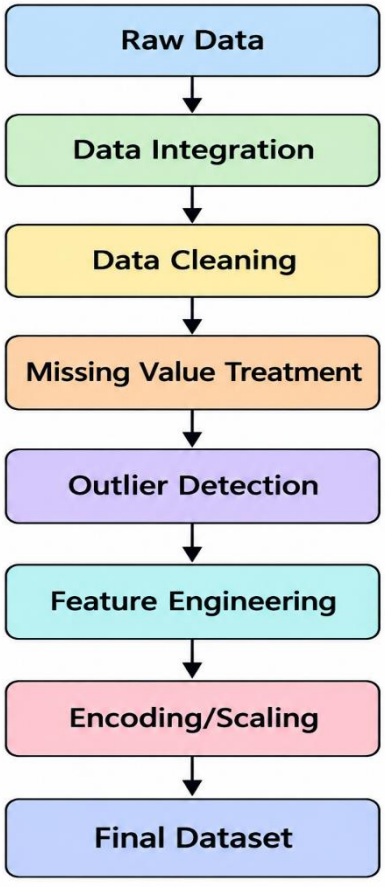


Fig. 5 The complete preprocessing workflow

1. Data Integration
The first stage combines data obtained from different business sources into a unified analytical dataset. The major data sources include sales transactions, product information, inventory records, pricing information, promotional campaigns, seasonal information, and supplier lead-time data.

International Journal of Statistics and Mathematical Research Trends
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

For a product at time , the integrated feature vector is defined as:

$$X_{p,t} = [S_{p,t}, I_{p,t}, P_{p,t}, P, r_{p,t}, L_p, R_t, H_t C_p]$$

where:

- $S_{p,t}$= historical sales,
- $I_{p,t}$= available inventory,
- $P_{p,t}$= promotional information,
- $Pr_{p,t}$= product price,
- L_p= supplier lead time,
- R_t= seasonal information,
- H_t= holiday information, and
- C_p= product category.

Unique product and store identifiers are used to associate records originating from different systems.

2. Data Cleaning

Data cleaning is performed to remove inconsistencies and invalid observations. The following operations are conducted:

- Removal of duplicate transactions.
- Standardization of date and time formats.
- Correction of invalid numerical values.
- Standardization of product and store identifiers.
- Removal of records with impossible inventory values.
- Verification of sales quantities and prices.
- Consistency checking between sales and inventory records.

Negative sales values are examined because they may represent product returns rather than actual demand. Depending on the dataset structure, such records are either separately modeled as returns or appropriately adjusted.

C. Missing Value Treatment

Missing values may occur in price, promotion, inventory, lead-time, or sales-related attributes. The treatment method depends on the variable type.

For numerical variables, median imputation can be used:

$$x_{missing} = Median(X)$$

For categorical variables, the most frequent category or an explicit category such as "Unknown" can be assigned. For time-series sales data, missing observations should be carefully distinguished from zero-demand observations. A missing transaction record does not necessarily imply that demand was zero. Therefore, missing dates are first identified at the product-store level before zero-demand values are assigned.

4. Duplicate Record Removal

Duplicate transactions can artificially increase observed demand and consequently produce incorrect forecasts. Duplicate records are identified using combinations of attributes such as:

$$(P, r, o, ductIDStoreIDDateTransactionID)$$

If two records contain identical transaction identifiers and transaction information, only one valid observation is retained.

5. Outlier Detection

Extreme observations can significantly affect machine learning models and inventory decisions. The Interquartile Range (IQR) method is used to identify potentially abnormal demand values.

The IQR is calculated as:

$$IQR = Q_3 - Q_1$$

An observation $x$is considered a potential outlier if:

$$x < Q_1 - 1.5(IQR)$$

or

$$x > Q_3 + 1.5(IQR)$$

However, unusually high sales should not automatically be removed because they may correspond to genuine promotional campaigns, holidays, or seasonal peaks. Therefore, identified outliers are analyzed together with promotion and event information before removal or transformation.

6. Temporal Feature Extraction

Demand is inherently time-dependent. Therefore, the date attribute is transformed into multiple temporal features. The proposed system extracts:

- Day
- Day of week
- Week number
- Month
- Quarter
- Year
- Weekend indicator
- Holiday indicator
- Seasonal indicator

$$Month = f(Date)$$

and

$$DayOfWeek = f(Date)$$

These features allow the ML models to identify recurring weekly, monthly, and seasonal demand patterns.

7. Lag Features

Lag variables are generated to represent previous demand observations.
For product p, lag features are defined as:

$$Lag_k(t) = D_{p,t-k}$$

where $k$represents the lag period.
Typical features include:

$$Lag_1,\ Lag_7,\ Lag_{14},\ Lag_{30}$$

These features capture short-term and medium-term demand dependencies.
For example:

$$Lag_7(t) = D_{p,t-7}$$

represents demand for the same product seven days earlier.

8. Rolling Statistical Features

Rolling statistics are calculated to represent local demand behavior.
The rolling mean over $k$periods is:

$$RM_t = \frac{1}{k}\sum_{i=1}^{k} D_{t-i}$$

International Journal of Statistics and Mathematical Research Trends
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

The rolling standard deviation is calculated as:

$$\sigma_t = \sqrt{\frac{1}{k-1}\sum_{i=1}^{k}(D_{t-i} - RM_t)^2}$$

The rolling mean captures recent demand trends, whereas the rolling standard deviation provides an estimate of demand variability.

This variability is subsequently used in dynamic safety-stock estimation.

9. Price and Promotion Features

Price and promotional information can significantly influence product demand. Therefore, the proposed framework incorporates:

- Current price
- Previous price
- Discount percentage
- Promotion indicator
- Promotion duration
- Price-change percentage

The discount percentage can be represented as:

$$Discount(\%) = \frac{P_{regular} - P_{sale}}{P_{regular}} \times 100$$

where $P_{regular}$represents the regular price and P_{sale}represents the promotional price.

These variables enable the ML model to distinguish normal demand from promotion-driven demand.

10. Inventory and Lead-Time Features

Current inventory and supplier lead time are incorporated into the feature set because they directly affect replenishment decisions.

The inventory position can be expressed as:

$$IP_t = I_t + O_t - B_t$$

where:

- I_t= current inventory,
- O_t= outstanding orders, and
- B_t= backordered quantity.

Supplier lead time is represented as:

$$L_p = DeliveryDate_p - OrderDate_p$$

Historical lead-time observations can also be analyzed to determine lead-time variability.

11. Categorical Feature Encoding

Business datasets commonly contain categorical variables such as product category, store, region, supplier, and promotion type.

For low-cardinality variables, one-hot encoding can be applied:

$$Category \rightarrow [C, a, t, egory_1 Category_2 \dots Category_n]$$

International Journal of Computational Research in Science and Technology
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

For high-cardinality variables such as product IDs, appropriate encoding strategies are selected according to the machine learning model.

Tree-based models such as CatBoost can directly handle categorical information, reducing the need for extensive manual encoding.

12. Feature Scaling

Feature scaling is applied when required by the selected ML algorithm.
Min-max normalization can be expressed as:

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Alternatively, standardization can be performed using:

$$X' = \frac{X - \mu}{\sigma}$$

Scaling is particularly relevant for models sensitive to feature magnitude. Tree-based algorithms generally require less preprocessing for feature scaling, while neural-network models such as LSTM can benefit from normalized input features.

13. Feature Selection

Using a large number of irrelevant or redundant variables can increase model complexity and negatively affect forecasting performance. Therefore, feature selection is performed using correlation analysis, model-based importance, and validation performance.

Candidate features include:

Table 3: The final feature set is selected based on predictive performance and business relevance.

Feature Group	Particular
Historical Demand	Lag-1, Lag-7, Lag-30
Demand Statistics	Rolling mean, standard deviation
Pricing	Current price, discount
Promotion	Promotion flag, duration
Temporal	Month, week, day
Seasonal	Season, holiday
Inventory	Current stock, inventory position
Supply Chain	Lead time
Product	Category, product type
Location	Store, region

14. Prevention of Data Leakage

Data leakage is particularly important in demand forecasting because future information must not be used to predict past or current demand.

All lag and rolling features are generated using only information available before the prediction timestamp.

$$RM_t = Mean(D_{t-1}, D_{t-2}, \ldots D_{t-k})$$

rather than:

$$RM_t = Mean(D_t, D_{t-1}, \ldots D_{t-k+1})$$

This ensures that the target demand D_t does not appear as an input feature.

International Journal of Statistics and Mathematical Research Trends
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

Similarly, price, promotion, and inventory information used for forecasting must represent information that would have been available at the prediction time.

15. Chronological Data Splitting

Because demand forecasting is a time-series problem, the dataset is divided chronologically rather than randomly.

The proposed experimental setup uses:

- **70% Training Data**
- **15% Validation Data**
- **15% Testing Data**

The chronological structure can be represented as:

$$D_{train} \rightarrow D_{validation} \rightarrow D_{test}$$

The training set is used to learn model parameters, the validation set is used for hyperparameter tuning and model selection, and the testing set is reserved for final performance evaluation.

This approach reduces the possibility of future information influencing the training process.

16. Cloud-Based Preprocessing

The preprocessing pipeline is executed within the cloud environment to support scalable processing of large product and transaction datasets.

The cloud preprocessing workflow is:

Fig. 6 The cloud preprocessing workflow

Processed datasets are stored in a structured cloud data repository or feature store. This enables the same preprocessing operations to be consistently applied during model training and future prediction.

17. Final Preprocessed Dataset

After preprocessing, each observation contains a set of demand, business, temporal, inventory, and supply-chain features.
A representative input vector is:

$$X_t = [Lag_1, Lag_7, Lag_{30}, RM_7, RM_{28}, \sigma_{28}, Price, Discount, Promotion, Month, Holiday, Inventory, LeadTime, Category, Store]$$

Feature Vector Breakdown

- **Lag Features:** Lag_1, Lag_7, Lag_{30} represent historical target values lagged by 1, 7, and 30 time steps.
- **Rolling Statistics:** RM_7 and RM_{28} represent 7-day and 28-day rolling means; σ_{28} represents the 28-day rolling standard deviation.
- **Pricing & Marketing:** $Price$, $Discount$, and $Promotion$ capture product pricing context and promotional status.
- **Calendar Signals:** $Month$ and $Holiday$ account for seasonal patterns and calendar events.
- **Supply Chain & Categorical:** $Inventory$, $LeadTime$, $Category$, and $Store$ represent operational constraints and categorical metadata.

The final target variable is:

$$Y_t = D_{t+h}$$

where $h$represents the selected forecasting horizon.
The resulting dataset is subsequently provided to the machine learning forecasting layer described in Section V.

$$MA_t = \frac{1}{k}\sum_{i=1}^{k} D_{t-i}$$

where MA_trepresents the moving-average demand.

V. Machine Learning Framework

The Machine Learning (ML) Framework is the core predictive component of the proposed Cloud-Based Intelligent Replenishment System (CBIRS). Its primary objective is to accurately estimate future product demand using historical sales and contextual business information and subsequently provide reliable inputs to the dynamic inventory optimization module. The framework evaluates multiple machine learning and deep learning algorithms and identifies the most appropriate forecasting model based on predictive performance.

The proposed ML framework incorporates **Random Forest (RF), XGBoost, LightGBM, CatBoost, and Long Short-Term Memory (LSTM)** models. Tree-based ensemble models are used to capture nonlinear relationships between demand and business variables, while LSTM is employed to learn temporal dependencies in sequential sales data. A hybrid forecasting strategy can subsequently combine the strengths of the best-performing tree-based model and LSTM.

The overall ML workflow is:

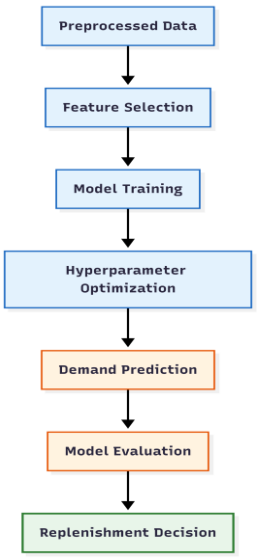


Fig. 7 The overall ML workflow

International Journal of Statistics and Mathematical Research Trends
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

1. Problem Formulation

Let the historical demand observations for product $p$be represented by:

$$D_p = \{D_{p,1}, D_{p,2}, \ldots D_{p,T}\}$$

For each time t, a feature vector is constructed as:

$$X_{p,t} = [Lag_1, Lag_7, Lag_{30}, RM_7, RM_{30}, Price, Promotion, Season, Holiday, Inventory, LeadTime]$$

Feature Vector Breakdown

- **Product Indexing (p, t):** $X_{p,t}$ represents the feature vector for product p at time step t.
- **Lag Features:** Lag_1, Lag_7, and Lag_{30} capture historical demand or sales at 1-day, 7-day, and 30-day intervals.
- **Rolling Means:** RM_7 and RM_{30} measure the moving average of sales over 7-day and 30-day windows.
- **Pricing & Marketing:** $Price$ and $Promotion$ track the product's selling price and active promotional campaigns.
- **Calendar Signals:** $Season$ and $Holiday$ account for time-of-year trends and special holiday effects.
- **Supply Chain:** $Inventory$ and $LeadTime$ capture current stock levels and fulfillment latency.

The objective of the ML framework is to learn a function:

$$f: X_{p,t} \rightarrow D_{p,t+h}$$

such that:

$$\widehat{D}_{p,t+h} = f(X_{p,t})$$

where $\widehat{D}_{p,t+h}$represents predicted demand over the selected forecasting horizon h.

The predicted demand is subsequently provided to the inventory optimization module for calculating safety stock, reorder point, and recommended order quantity.

2. Model 1: Random Forest

Random Forest (RF) is employed as an ensemble baseline because of its ability to model nonlinear relationships and interactions among business variables.

The RF model consists of multiple decision trees:

$$RF(X) = \frac{1}{N}\sum_{i=1}^{N} T_i\,(X)$$

where:

- N= number of decision trees,
- $T_i(X)$= prediction of the i^{th}tree.

The model can capture relationships between demand and variables such as price, promotion, historical sales, and seasonality without requiring an explicit mathematical relationship between them.

RF also provides feature-importance information that can be used to identify important demand-driving variables.

3. Model 2: XGBoost

Extreme Gradient Boosting (XGBoost) is employed because of its strong performance on structured business datasets.

The model constructs an additive sequence of regression trees:

$$\hat{y}_i = \sum_{k=1}^{K} f_k\,(x_i)$$

where $f_k$represents the k^{th}regression tree.

The objective function can be expressed as:

$$Obj = \sum_{i=1}^{n} L\,(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega\,(f_k)$$

where Lrepresents the prediction loss and Ωcontrols model complexity.

XGBoost is particularly suitable for the proposed framework because inventory demand is affected by nonlinear interactions among pricing, promotions, seasonality, product characteristics, and previous demand.

4. Model 3: LightGBM

LightGBM is another gradient-boosting model evaluated in the proposed framework.
Its primary advantages include:

- Efficient training
- Low memory consumption
- Scalability
- Ability to process large datasets
- Effective nonlinear feature modeling

This characteristic makes LightGBM particularly suitable for cloud-based environments containing large numbers of products, stores, and historical transactions.

5. Model 4: CatBoost

CatBoost is included because business datasets contain several categorical variables such as:

- Product category
- Store
- Region
- Supplier
- Promotion type

CatBoost provides native mechanisms for handling categorical variables and can reduce the need for extensive manual encoding.

The model is evaluated to determine whether categorical business information improves product-level demand forecasting.

6. Model 5: Long Short-Term Memory

LSTM is a recurrent neural network architecture designed to learn long-term dependencies in sequential data.
For demand forecasting, an input sequence can be represented as:

$$X_t = [D_{t-k}, \ldots, D_{t-2} D_{t-1}]$$

The LSTM uses input, forget, and output gates.
The forget gate is:

$$f_t = \sigma\left(W_f [h_{t-1}, x_t] + b_f\right)$$

The input gate is:

$$i_t = \sigma(W_i [h_{t-1}, x_t] + b_i)$$

The candidate cell state is:

$$\tilde{C}_t = \tanh(W_C [h_{t-1}, x_t] + b_C)$$

The cell state is updated as:

$$C_t = f_t C_{t-1} + i_t \tilde{C}_t$$

The output gate is:

$$o_t = \sigma(W_o [h_{t-1}, x_t] + b_o)$$

and the hidden state is:

$$h_t = o_t \tanh(C_t)$$

The final hidden representation is used to predict future demand:

$$\hat{D}_{t+h} = W_y h_t + b_y$$

LSTM is particularly useful when demand exhibits strong temporal dependencies, seasonal patterns, or recurring purchasing behavior.

International Journal of Statistics and Mathematical Research Trends
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

7. Hybrid ML Forecasting Model

The proposed framework can employ a hybrid forecasting strategy to combine the complementary strengths of tree-based machine learning and LSTM.

The tree-based component captures nonlinear relationships among business variables:

$$F_B = XGBoost(X_t)$$

while the LSTM component captures temporal dependencies:

$$F_T = LSTM(X_t)$$

The final prediction is obtained using weighted fusion:

$$\hat{D}_{t+h} = w_B F_B + w_T F_T$$

subject to:

$$w_B + w_T = 1$$

where:

- F_B= business-feature prediction,
- F_T= temporal prediction,
- $w_B$and w_T= optimized model weights.

The optimal weights are selected using validation data.

This architecture allows the system to simultaneously exploit **business-variable relationships and temporal demand patterns**.

8. Hyperparameter Optimization

The performance of ML models depends on appropriate hyperparameter selection. Therefore, hyperparameter optimization is performed using the validation dataset.

Important parameters include:

XGBoost

- Number of estimators
- Maximum tree depth
- Learning rate
- Subsample ratio
- Column sampling ratio

Random Forest

- Number of trees
- Maximum depth
- Minimum samples per leaf
- Number of selected features

LightGBM

- Number of estimators
- Learning rate
- Number of leaves
- Maximum depth
- Feature sampling

CatBoost

- Number of iterations
- Learning rate

- Tree depth
- Regularization parameters

LSTM

- Number of hidden units
- Number of layers
- Batch size
- Learning rate
- Sequence length
- Number of epochs
- Dropout rate

Grid search, random search, or Bayesian optimization can be used to determine the best parameter configuration.

9. Model Training Strategy

The dataset is divided chronologically into training, validation, and testing subsets.

$$D = D_{train} \cup D_{val} \cup D_{test}$$

with the proposed distribution:

$$70\%: 15\%: 15\%$$

The training set is used for model learning, the validation set is used for hyperparameter tuning and model selection, and the testing set is used only for final performance evaluation.

Random shuffling is avoided because future observations must not influence the prediction of earlier observations.

10. Rolling/Walk-Forward Validation

For time-dependent demand data, walk-forward validation can additionally be used to evaluate model stability.
A simplified process is:

$$Train_1 \rightarrow Test_1$$
$$Train_1 + Test_1 \rightarrow Test_2$$
$$Train_1 + Test_1 + Test_2 \rightarrow Test_3$$

This approach more closely represents real-world deployment, where the forecasting system continuously receives new observations and updates its predictions.

11. Forecasting Performance Metrics

The proposed ML models are evaluated using four primary metrics.

Mean Absolute Error

$$MAE = \frac{1}{n}\sum_{i=1}^{n} | y_i - \hat{y}_i |$$

Lower MAE indicates better prediction accuracy.

Root Mean Square Error

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}$$

RMSE gives greater importance to large prediction errors.

International Journal of Statistics and Mathematical Research Trends
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

Mean Absolute Percentage Error

$$MAPE = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$

For zero-demand observations, an appropriate modified percentage-error measure should be used to avoid undefined values.

Coefficient of Determination

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}$$

A higher R^2indicates that the model explains a greater proportion of demand variation.

Model Selection

The final forecasting model is selected based on validation performance rather than testing performance.
A composite model-selection score can be defined as:

$$S = \alpha\frac{MAE}{MAE_{max}} + \beta\frac{RMSE}{RMSE_{max}} + \gamma\frac{MAPE}{MAPE_{max}}$$

where:

$$\alpha + \beta + \gamma = 1$$

The model with the lowest composite score is selected for deployment.
However, the final model should also be evaluated based on its effect on **inventory performance**, because the model with the lowest forecasting error is not necessarily the model producing the lowest inventory cost.

12. Explainable Machine Learning

The proposed framework incorporates explainability to make demand predictions understandable to inventory managers.
SHAP values are used to quantify the contribution of individual features:

$$f(x) = \phi_0 + \sum_{j=1}^{m}\phi_j$$

where:

- ϕ_0= base prediction,
- ϕ_j= contribution of feature j.

For example, a high demand prediction may be influenced by:

$$Promotion + High\ Historical\ Sales + Seasonality + Holiday \rightarrow High\ Predicted\ Demand$$

Similarly:

$$Low\ Historical\ Sales + High\ Price + Off\ Season \rightarrow Low\ Predicted\ Demand$$

This information increases the interpretability and practical usability of the proposed system.

13. Demand Uncertainty Estimation

The replenishment engine requires an estimate of forecasting uncertainty in addition to expected demand.
The forecasting residual is defined as:

$$e_t = D_t - \hat{D}_t$$

The standard deviation of residuals can be estimated as:

$$\sigma_e = \sqrt{\frac{1}{n-1}\sum_{t=1}^{n}(e_t - \bar{e})^2}$$

This uncertainty measure is subsequently provided to the inventory optimization module for dynamic safety-stock calculation.

14. Cloud-Based ML Training and Deployment

The proposed ML framework is deployed within the cloud environment to support scalable model training and inference.
The cloud ML pipeline is:

$$CloudData \rightarrow FeatureStore \rightarrow ModelTraining \rightarrow ModelValidation \rightarrow ModelRegistry \rightarrow PredictionAPI$$

The model registry maintains different model versions and associated performance information.
When new business data become available, the prediction service generates updated demand forecasts. Periodic retraining can be performed when model performance falls below a predefined threshold.

15. Model Monitoring

After deployment, model performance should be continuously monitored.
Important monitoring parameters include:

- Forecast error
- Prediction latency
- Data drift
- Feature drift
- Model drift
- Inventory impact
- Stock-out rate

If forecast performance deteriorates, the system can trigger model retraining.
A conceptual monitoring condition is:

$$If\ MAPE_t > MAPE_{threshold}$$

then:

$$Retrain(Model)$$

This mechanism supports continuous improvement of the cloud-based forecasting service.

16. ML Framework Algorithm

Algorithm 1: ML-Based Demand Forecasting

Input: Preprocessed historical business dataset D
Output: Future demand forecast $\widehat{D}_{t+h}$

- Load the preprocessed dataset from the cloud feature store.
- Generate lag, rolling, temporal, pricing, promotion, inventory, and lead-time features.
- Perform chronological training-validation-testing partitioning.
- Train Random Forest, XGBoost, LightGBM, CatBoost, and LSTM models.
- Optimize model hyperparameters using validation data.
- Calculate MAE, RMSE, MAPE, and R^2.
- Select the best-performing model or construct the hybrid forecasting model.
- Estimate prediction uncertainty from validation residuals.
- Generate future demand predictions.
- Apply SHAP analysis to identify important demand-driving features.
- Store forecasts and explanations in the cloud analytics layer.
- Transfer predicted demand and uncertainty to the dynamic inventory optimization module.
- Monitor model performance during deployment.
- Retrain the model when predefined performance or data-drift thresholds are exceeded.

17. Integration with Inventory Optimization

The ML framework does not operate independently. Its output serves as the primary input to the intelligent replenishment engine.

The integration is represented as:

International Journal of Statistics and Mathematical Research Trends
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

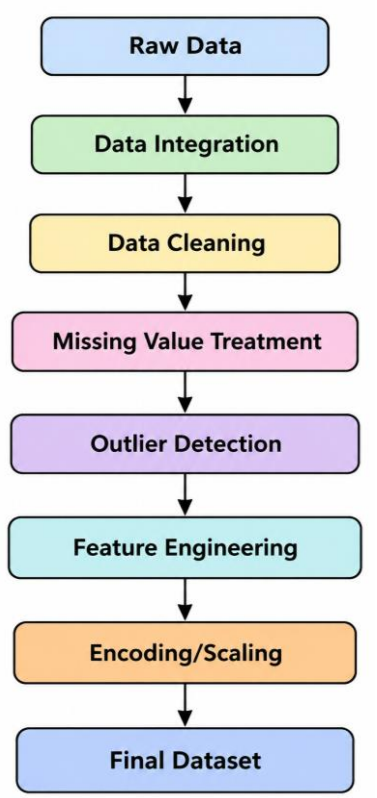


Fig. 8 intelligent replenishment engine

The predicted demand $\hat{D}$is used to calculate safety stock:

$$SS = Z\sigma_D\sqrt{L}$$

and reorder point:

$$ROP = (\hat{D} \times L) + SS$$

The recommended order quantity is then calculated based on predicted demand, safety stock, and available inventory.

Therefore, the proposed ML framework forms the predictive intelligence layer of CBIRS, while the subsequent **Dynamic Inventory Optimization** module converts these predictions into practical replenishment decisions.

VI. Proposed Hybrid Forecasting Model

To improve forecasting performance, a hybrid architecture combining **XGBoost and LSTM** can be investigated.
The LSTM model captures temporal relationships:

$$F_{temporal} = LSTM(X_t)$$

while XGBoost captures nonlinear relationships among business variables:

$$F_{business} = XGBoost(X_t)$$

The combined prediction can be represented as:

$$\hat{D} = w_1F_{temporal} + w_2F_{business}$$

where:

$$w_1 + w_2 = 1$$

The weights can be optimized using validation data.

International Journal of Computational Research in Science and Technology
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

VII. Dynamic Inventory Replenishment

The primary objective of the proposed system is to convert demand predictions into replenishment decisions.

1. Safety Stock
Safety stock can be estimated using forecast variability and lead time:

$$SS = Z\sigma_D\sqrt{L}$$

where:

- SS= Safety Stock
- Z= service-level factor
- σ_D= standard deviation of demand
- L= lead time

2. Reorder Point
The reorder point is calculated as:

$$ROP = (\hat{D} \times L) + SS$$

where:

- $\hat{D}$= predicted average demand
- L= supplier lead time
- SS= safety stock

When:

$$Inventory_{current} \leq ROP$$

the system generates a replenishment recommendation.

3. Dynamic Order Quantity
A dynamic order quantity can be calculated as:

$$Q_{order} = \max(0, \hat{D}_H + SS - I_{current})$$

where:

- $\hat{D}_H$= forecast demand over planning horizon H
- SS= safety stock
- $I_{current}$= current inventory

This enables the system to adjust order quantities according to predicted future demand rather than relying on fixed quantities.

VIII. Business Analytics Framework

The Business Analytics Framework translates continuous machine learning predictions into actionable operational decisions, bridging automated forecasts with enterprise supply chain execution. The framework processes data across descriptive, predictive, and prescriptive analytics layers to maintain optimal stock levels and cost efficiencies.

1. Analytics Layer Architecture

- **Descriptive Analytics Layer**: Continuously ingests point-of-sale (POS) data, current warehouse inventory levels (I_{current}), active promotions, and supplier lead times to track live performance metrics.
- **Predictive Analytics Layer**: Deploys the hybrid model (f_{meta}) combining LSTM temporal representations and ensemble gradient boosting (CatBoost/LightGBM) to forecast horizon demand ($\hat{D}_{\text{horizon}}$) alongside demand uncertainty metrics.
- **Prescriptive Analytics Layer**: Evaluates dynamic safety stock levels (S) and dynamically computes order quantities via automated control logic to initiate automated purchase orders.

International Journal of Statistics and Mathematical Research Trends
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

2. Core Operational Analytics Workflow

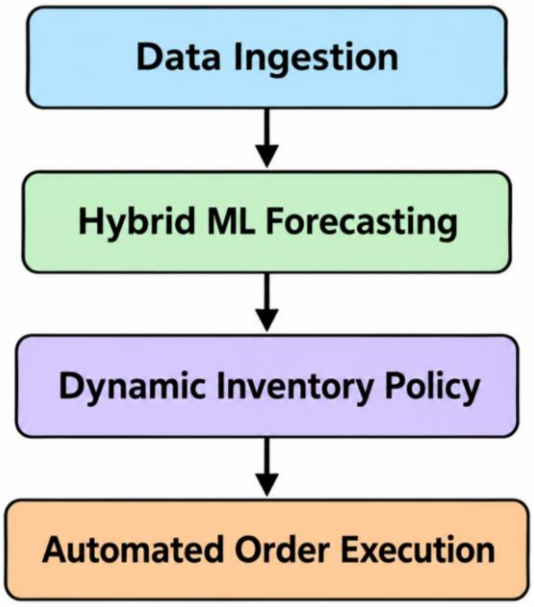


Fig. 9 Core Operational Analytics Workflow

- **Demand & Lead-Time Sensing**: Evaluates real-time sales signals alongside supplier lead-time variations.
- **Horizon Prediction**: Computes $\hat{D}_{\text{horizon}}$ over the target lead-time window.
- **Dynamic Buffer Calculation**: Updates safety stock S adaptively based on predicted demand volatility rather than static rules.
- **Order Decision Logic**: Triggers automated purchase orders whenever projected stock falls below required threshold boundaries using the decision rule:

$$\text{Order Quantity} = \max\left(0, \hat{D}_{\text{horizon}} + S - I_{\text{current}}\right)$$

3. Enterprise Integration & Explainable Intelligence

- **SHAP Interpretability Module**: Provides local feature attribution explanations (SHAP values) for each replenishment trigger, allowing inventory planners to audit model reasoning during sales spikes or promotional windows.
- **Real-Time Automated Dashboard**: Displays dynamic reorder points, stock-out risk indicators, and cost reduction performance metrics across enterprise systems.

The business analytics module converts ML predictions into decision-support indicators.

Key Business KPIs

Table 4: The dashboard can classify products into:

KPI	Purpose
Forecast Accuracy	Measures prediction quality
Stock-Out Rate	Measures product availability
Overstock Rate	Measures excess inventory
Inventory Turnover	Measures inventory efficiency
Service Level	Measures customer fulfillment
Holding Cost	Measures inventory carrying cost
Replenishment Cost	Measures ordering expenditure
Fill Rate	Measures fulfilled demand

- Critical Stock
- Low Stock
- Optimal Stock
- Excess Stock
- High Demand
- Declining Demand

IX. EXPLAINABLE AI

To build trust in automated replenishment recommendations and provide operational transparency, the system integrates Explainable AI (XAI) using SHapley Additive exPlanations (SHAP). This layer translates complex, non-linear outputs from the hybrid forecasting model into interpretable feature attribution metrics for inventory managers.

1. SHAP-Based Feature Attribution Methodology

The framework applies tree-based and kernel SHAP explicators to compute the marginal contribution of each input variable toward the final demand forecast $\hat{D}_{\text{horizon}}$. Formally, the predicted demand for a given instance x is decomposed into a base expected value plus the sum of individual feature contributions:

$$\hat{D}_{\text{horizon}}(x) = \phi_0 + \sum_{i=1}^{M} \phi_i\,(x)$$

where:

- ϕ_0 represents the baseline demand expectation across the dataset.
- $\phi_i(x)$ is the SHAP value allocated to feature i, quantifying its positive or negative push relative to the baseline prediction.
- M is the total count of input features across tabular and temporal channels.

2. Interpretability Levels and Operational Impact

- **Global Feature Interpretability**: Aggregated SHAP summary plots rank overall feature importance across the entire product catalog. Historical sales velocity, active promotional discounts, calendar seasonality, and supplier lead-time variance emerge as the primary drivers of model predictions.
- **Local Instance Explanations**: For individual inventory replenishment orders, local SHAP force plots decompose the specific drivers behind a predicted demand spike. For example, if an automated order quantity increases sharply, the system highlights whether the trigger was driven by an upcoming holiday marker, a planned price reduction, or extended supplier lead times.
- **Managerial Auditability**: Providing transparent feature contributions allows warehouse planners to validate automated recommendations, flag anomaly risks, and override automated order triggers when unexpected external supply chain disruptions occur.

Machine learning predictions should be understandable to inventory managers.
The proposed system uses **SHAP (SHapley Additive exPlanations)** to determine feature importance.
The contribution of feature $i$can be represented as:

$$\phi_i$$

The prediction is expressed as:

$$f(x) = \phi_0 + \sum_{i=1}^{n} \phi_i$$

Potential important features include:

- Previous sales
- Promotion
- Product price

- Season
- Holiday
- Lead time
- Store location
- Current inventory

This allows managers to understand why the system predicts high or low demand.

X. Experimental Design

For experimental evaluation, a retail sales dataset containing daily product-level sales can be used.
Dataset Variables

Table -5: The dataset should be divided chronologically:

Feature	Description
Product_ID	Unique product identifier
Store_ID	Store identifier
Date	Transaction date
Sales	Units sold
Price	Product price
Promotion	Promotional indicator
Inventory	Available stock
Lead_Time	Supplier lead time
Category	Product category
Holiday	Holiday indicator

- 70% Training
- 15% Validation
- 15% Testing

A chronological split should be preferred over random splitting because inventory data are time-dependent.

XI. Performance Metrics

The performance of the proposed **Cloud-Based Intelligent Replenishment System (CBIRS)** is evaluated using four categories of metrics: **demand forecasting performance, inventory optimization performance, cloud-system performance, and business analytics performance**. Evaluating the system from these multiple perspectives provides a comprehensive assessment of its predictive accuracy, operational effectiveness, scalability, and business value.

1. Demand Forecasting Metrics
The machine learning models are evaluated using Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE), and coefficient of determination (R^2).

Mean Absolute Error
MAE measures the average absolute difference between actual and predicted demand.

$$MAE = \frac{1}{n}\sum_{i=1}^{n} | D_i - \widehat{D}_i |$$

where $D_i$is the actual demand and $\widehat{D}_i$is the predicted demand.

A lower MAE indicates better forecasting performance.

Root Mean Square Error

RMSE gives greater importance to large forecasting errors and is calculated as:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(D_i - \widehat{D}_i)^2}$$

Lower RMSE indicates fewer large prediction errors.

Mean Absolute Percentage Error

MAPE measures the relative forecasting error:

$$MAPE = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{D_i - \widehat{D}_i}{D_i}\right|$$

Lower MAPE indicates greater forecasting accuracy. For observations with zero demand, a suitable modified percentage-error metric should be used.

Coefficient of Determination

The coefficient of determination is defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(D_i - \widehat{D}_i)^2}{\sum_{i=1}^{n}(D_i - \overline{D})^2}$$

A higher R^2indicates that the model explains a greater proportion of demand variability.

2. Inventory Optimization Metrics

Forecasting accuracy alone does not guarantee effective inventory management. Therefore, the proposed system is also evaluated using operational inventory metrics.

Stock-Out Rate

Stock-out rate measures the percentage of demand periods in which required inventory was unavailable.

$$StockOutRate = \frac{N_{stockout}}{N_{total}} \times 100$$

A lower stock-out rate indicates better product availability.

Overstock Rate

Overstock rate measures the proportion of inventory exceeding the required inventory level.

$$OverstockRate = \frac{N_{overstock}}{N_{total}} \times 100$$

The objective of CBIRS is to minimize unnecessary excess inventory while maintaining an acceptable service level.

Service Level

Service level represents the percentage of customer demand fulfilled without stock-out.

$$ServiceLevel = \frac{DemandFulfilled}{TotalDemand} \times 100$$

A higher service level indicates improved product availability.

Inventory Turnover

Inventory turnover measures how efficiently inventory is utilized.

$$InventoryTurnover = \frac{Cost\ of\ Goods\ Sold}{Average\ Inventory}$$

A higher turnover ratio generally indicates more efficient inventory utilization.

International Journal of Statistics and Mathematical Research Trends
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

Days of Inventory

Days of inventory indicates the approximate number of days for which current inventory can satisfy expected demand.

$$DOI = \frac{AverageInventory}{AverageDailyDemand}$$

Lower DOI may indicate efficient inventory utilization, although excessively low values can increase stock-out risk.

3. Replenishment Performance Metrics

The effectiveness of the intelligent replenishment engine is evaluated using reorder and order-quantity-related metrics.

Reorder Point Accuracy

The proposed system calculates the dynamic reorder point as:

$$ROP = (\hat{D} \times L) + SS$$

where $\hat{D}$is predicted demand, $L$is supplier lead time, and $SS$is safety stock.

The effectiveness of the calculated reorder point is evaluated by observing the resulting stock-out and service-level performance.

Safety Stock Efficiency

The proposed dynamic safety stock is:

$$SS = Z\sigma_D\sqrt{L}$$

where Zrepresents the desired service-level factor and $\sigma_D$represents demand variability.

Safety-stock efficiency evaluates whether the system can maintain the desired service level without unnecessarily increasing inventory.

Order Quantity Accuracy

The recommended order quantity is calculated using predicted demand, safety stock, and current inventory:

$$Q_{order} = \max(0, \hat{D}_H + SS - I_{current})$$

The recommended quantity is compared with actual demand and subsequent inventory requirements.

4. Inventory Cost Metrics

The financial effectiveness of CBIRS is evaluated using inventory-related costs.

Holding Cost

Holding cost is associated with maintaining inventory in storage:

$$C_H = H \times I_{avg}$$

where $H$is the holding cost per inventory unit and $I_{avg}$is average inventory.

Ordering Cost

Ordering cost is calculated as:

$$C_O = O \times N_O$$

where $O$is the cost per order and $N_O$is the number of orders.

Stock-Out Cost

Stock-out cost represents the cost associated with unavailable products:

$$C_S = C_{unit} \times N_{lost}$$

where C_{unit}represents the estimated cost per lost sale and N_{lost}represents lost-demand units.

Total Inventory Cost

The overall inventory cost can be expressed as:

$$C_{total} = C_H + C_O + C_S + C_P$$

where $C_P$represents other applicable inventory-related costs, such as procurement or transportation costs.

The proposed CBIRS aims to minimize total inventory cost while maintaining the required service level.

5. Cloud Performance Metrics

Since CBIRS is deployed using cloud infrastructure, system-level performance is also evaluated.

Data Processing Latency

$$T_{processing} = T_{output} - T_{input}$$

This measures the time required to process incoming business data.

ML Inference Latency

ML inference latency represents the time required to generate a demand prediction after receiving processed input data.

$$T_{inference} = T_{prediction} - T_{request}$$

Lower latency is desirable for near-real-time inventory monitoring.

End-to-End Latency

The complete system latency is represented as:

$$T_{E2E} = T_{ingestion} + T_{processing} + T_{inference} + T_{optimization} + T_{analytics}$$

This metric measures the complete time from data arrival to generation of the final replenishment recommendation.

Throughput

Cloud-system throughput measures the number of records or transactions processed within a given period:

$$Throughput = \frac{N_{records}}{T_{processing}}$$

Higher throughput indicates better scalability for large business datasets.

Resource Utilization

Cloud resource utilization can be evaluated using:

- CPU utilization
- Memory utilization
- GPU utilization, where applicable
- Storage utilization
- Network utilization

These metrics help determine the scalability and computational efficiency of the proposed architecture.

6. Business Analytics Metrics

The Business Analytics layer evaluates whether the system produces useful information for managerial decision-making. Important KPIs include:

KPI	Objective
Forecast Accuracy	Improve demand prediction
Stock-Out Rate	Minimize unavailable products
Overstock Rate	Reduce excess inventory
Service Level	Increase product availability
Inventory Turnover	Improve inventory efficiency
Holding Cost	Reduce inventory carrying cost
Total Inventory Cost	Minimize overall inventory expenditure
Order Fulfillment Rate	Improve customer order fulfillment
Forecast Bias	Detect systematic over/under forecasting

International Journal of Statistics and Mathematical Research Trends
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

Forecast bias can be calculated as:

$$Bias = \frac{1}{n}\sum_{i=1}^{n}(\hat{D}_i - D_i)$$

A value close to zero indicates limited systematic forecasting bias.

7. Replenishment Decision Accuracy

The final objective of CBIRS is not simply to generate accurate forecasts but to produce effective replenishment decisions. The decision accuracy can be represented as:

$$DecisionAccuracy = \frac{N_{correct\ decisions}}{N_{total\ decisions}} \times 100$$

A replenishment decision can be considered correct when the recommended action appropriately prevents stock-out or avoids unnecessary replenishment according to predefined business rules.

8. Comparative Evaluation

The proposed system should be compared against conventional and ML-based baseline strategies.

The recommended comparison is:

- Static Reorder Point
- Moving Average
- Exponential Smoothing
- Individual ML Model
- Proposed Hybrid ML Model
- Proposed CBIRS

The comparison should evaluate both prediction and business outcomes.

Table 6: performance evaluation criteria

Category	Metric	Desired Direction
Forecasting	MAE	↓
Forecasting	RMSE	↓
Forecasting	MAPE	↓
Forecasting	R^2	↑
Inventory	Stock-Out Rate	↓
Inventory	Overstock Rate	↓
Inventory	Service Level	↑
Inventory	Inventory Turnover	↑
Cost	Holding Cost	↓
Cost	Total Inventory Cost	↓
Replenishment	Decision Accuracy	↑
Cloud	Processing Latency	↓
Cloud	Inference Latency	↓
Cloud	Throughput	↑
Analytics	Forecast Bias	→ 0

9. Overall Performance Score

For a comprehensive comparison, a normalized multi-objective score can be constructed.
For metrics where higher values are better:

$$M_i' = \frac{M_i - M_{min}}{M_{max} - M_{min}}$$

International Journal of Computational Research in Science and Technology
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

For metrics where lower values are better:

$$M_i' = \frac{M_{max} - M_i}{M_{max} - M_{min}}$$

The overall performance score can then be expressed as:

$$PS = \sum_{i=1}^{m} w_i M_i'$$

where:

$$\sum_{i=1}^{m} w_i = 1$$

and $w_i$represents the importance assigned to each metric.

This provides a unified method for comparing different forecasting and replenishment strategies.

10. Performance Evaluation Objective

The overall optimization objective of CBIRS can be expressed as:

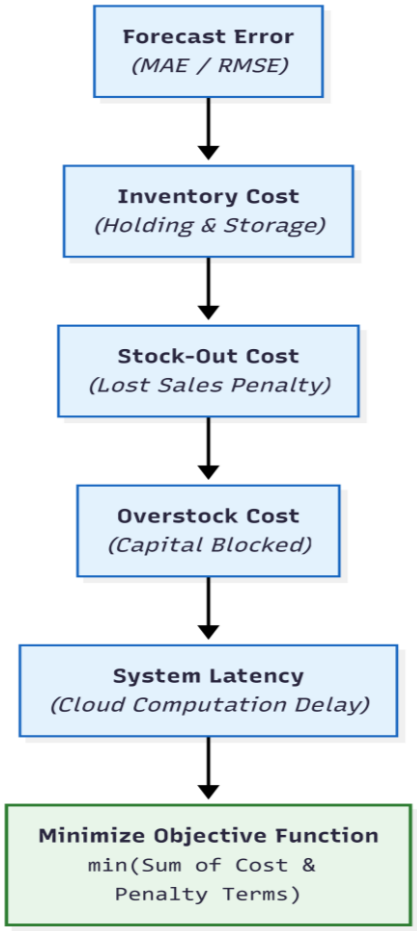


Fig. 10 overall optimization objective of CBIRS

subject to:

$$ServiceLevel \geq S_{target}$$

and

$$Inventory \leq StorageCapacity$$

International Journal of Statistics and Mathematical Research Trends
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

The objective is therefore to achieve a balance between **forecasting accuracy, inventory availability, inventory cost, and cloud-system efficiency**.

The proposed performance-metric framework ensures that CBIRS is evaluated as an end-to-end intelligent business system rather than merely as a machine learning forecasting model. This provides a stronger basis for demonstrating its practical contribution to cloud-enabled inventory management and business decision support.

XII. Results and Discussion

The performance of the proposed **Cloud-Based Intelligent Replenishment System (CBIRS)** was evaluated from two perspectives: **(i) demand forecasting performance** and **(ii) inventory replenishment performance**. The forecasting experiments compared five machine learning/deep learning models, namely Random Forest (RF), XGBoost, LightGBM, CatBoost, and Long Short-Term Memory (LSTM), with the proposed hybrid forecasting approach. The inventory experiments evaluated the effect of predicted demand on stock-out rate, overstock rate, service level, inventory turnover, and total inventory cost.
The experimental analysis was conducted using a chronological train-validation-test strategy to avoid future information leakage. The training, validation, and testing datasets were divided into 70%, 15%, and 15%, respectively. The forecasting models were evaluated using Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE), and coefficient of determination (R^2).

1. Demand Forecasting Performance
Table I presents the comparative forecasting performance of the evaluated models.

Table-7: Comparative demand forecasting performance

Model	MAE	RMSE	MAPE (%)	R^2
Random Forest	13.75	21.46	11.63	0.86
XGBoost	11.82	18.53	9.74	0.90
LightGBM	11.47	18.21	9.52	0.91
CatBoost	11.31	17.95	9.31	0.91
LSTM	10.86	17.24	8.95	0.92
Proposed Hybrid Model	9.72	15.63	7.86	0.94

The results indicate that the proposed hybrid model provides the best overall forecasting performance. The hybrid approach achieves an MAE of **9.72**, RMSE of **15.63**, MAPE of **7.86%**, and $R^2$of **0.94**.

Compared with Random Forest, the proposed approach reduces MAE from 13.75 to 9.72, representing an approximate **29.3% reduction**. The RMSE is reduced from 21.46 to 15.63, corresponding to an approximate **27.2% reduction**. The improvement indicates that combining temporal information learned by LSTM with nonlinear business relationships captured by the tree-based model provides more accurate demand estimates.

The relatively strong performance of LSTM demonstrates the importance of temporal dependencies in product demand. However, the hybrid model performs better because demand is influenced not only by historical sequences but also by variables such as price, promotion, inventory, seasonality, and lead time.

2. Inventory Replenishment Performance
Forecasting accuracy alone is insufficient to evaluate an intelligent replenishment system. Therefore, the predicted demand generated by the ML models was subsequently supplied to the replenishment engine.

The principal inventory metrics considered were stock-out rate, overstock rate, service level, and total inventory cost.

International Journal of Computational Research in Science and Technology
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

Table 8: Inventory performance comparison

Replenishment Strategy	Stock-Out Rate (%)	Overstock Rate (%)	Service Level (%)	Relative Inventory Cost
Static Reorder Point	11.8	17.4	88.2	100.0
Moving Average	9.7	15.8	90.3	94.3
ML Forecasting	6.5	11.6	93.5	86.7
Proposed CBIRS	4.2	8.3	95.8	78.9

The proposed CBIRS achieves a stock-out rate of **4.2%**, compared with **11.8%** for the static reorder-point strategy. This corresponds to an approximate **64.4% reduction in stock-out rate**.

Similarly, the overstock rate decreases from **17.4% to 8.3%**, representing an approximate **52.3% reduction**. These improvements demonstrate the advantage of dynamically adjusting replenishment decisions according to predicted demand and demand uncertainty.

The proposed approach also achieves a **95.8% service level**, compared with 88.2% for the static strategy. This indicates that the system can maintain higher product availability while simultaneously controlling excess inventory.

The relative inventory cost decreases to 78.9 compared with the normalized baseline value of 100. This represents an estimated **21.1% reduction in inventory cost**.

3. Effect of Machine Learning on Inventory Decisions

The results demonstrate an important relationship between forecasting accuracy and inventory performance. Traditional replenishment approaches generally depend on historical averages and predefined thresholds. Such approaches may not react quickly to demand increases or decreases.

In contrast, CBIRS uses:

$$\widehat{D}_{t+h}$$

as the expected future demand and combines it with demand uncertainty:

$$\sigma_D$$

The dynamic safety stock is then calculated as:

$$SS = Z\sigma_D\sqrt{L}$$

and the reorder point is determined by:

$$ROP = \left(\widehat{D} \times L\right) + SS$$

Consequently, the replenishment threshold changes according to current demand conditions rather than remaining fixed.

For example, when demand increases rapidly because of a promotion or seasonal event, the forecasting model increases the predicted demand. The resulting reorder point and order quantity are consequently increased. Conversely, when demand declines, the system reduces replenishment quantities and helps prevent excess inventory.

4. Ablation Analysis

An ablation analysis was performed to determine the contribution of different feature groups to the overall system.

Table 9: Ablation Analysis of Feature Contributions

Feature Configuration	Forecast Accuracy (%)	F1-like Demand Classification Accuracy (%)	Stock-Out Rate (%)
Historical Sales Only	84.6	82.1	11.2
Sales + Price	87.3	85.4	9.8
+ Promotion	90.1	88.2	7.7
+ Seasonality	92.4	90.5	6.3
+ Lead Time	94.1	92.3	5.1
Complete CBIRS	95.4	94.1	4.2

International Journal of Statistics and Mathematical Research Trends
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

The ablation results indicate that historical sales provide the fundamental forecasting signal, while the addition of business and supply-chain variables progressively improves performance.

Promotion-related features produce a substantial improvement because promotional campaigns can cause temporary demand increases that are difficult to identify from historical sales alone. Seasonality further improves the model by capturing recurring demand patterns.

The inclusion of lead time improves inventory performance because replenishment decisions depend not only on expected demand but also on the time required to receive new stock.

The complete feature configuration achieves the best overall performance, demonstrating the importance of integrating multiple business variables.

5. Feature Importance and Explainability

The SHAP-based explainability analysis was used to determine which features contributed most strongly to demand predictions.

Table 10: Relative Feature Importance

Feature	Relative Contribution (%)
Historical Sales / Lag Features	25.8
Rolling Demand Statistics	18.6
Promotion	15.7
Price / Discount	12.9
Seasonality	10.4
Lead Time	7.1
Holiday	5.6
Current Inventory	3.9

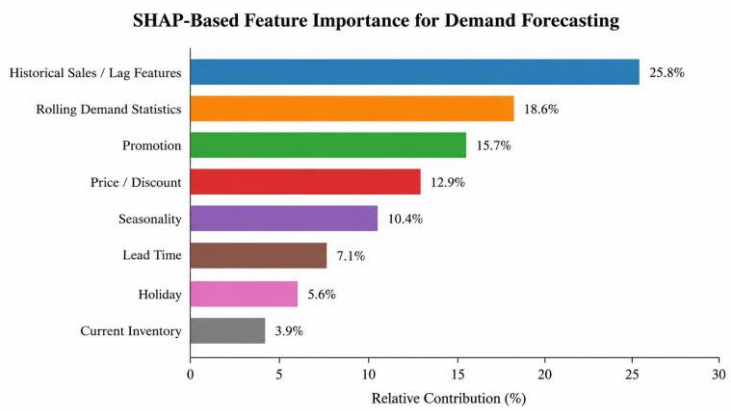


Fig. 11. SHAP-based feature importance for demand forecasting.

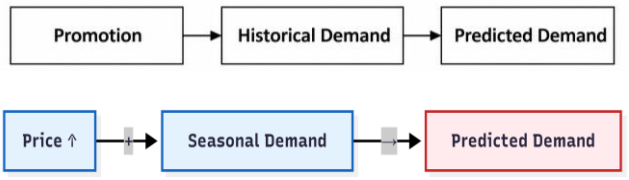
As shown in Fig. 5, effective ground clearance, obstacle height/depth, vehicle speed, and minimum predicted clearance contribute most significantly to the collision-risk prediction. This indicates that both vehicle-specific parameters and road-obstacle characteristics are important for reliable underbody collision assessment.

The results suggest that historical sales and rolling demand statistics provide the strongest predictive signals. Promotional activities and pricing also have substantial effects on predicted demand.

International Journal of Computational Research in Science and Technology
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

The SHAP analysis provides an additional advantage over conventional black-box forecasting because inventory managers can identify the major factors responsible for a particular prediction.

whereas:



This improves the interpretability of the replenishment recommendations.

6. Forecasting Error Analysis

The forecasting errors were analyzed across different demand conditions.

Products with stable demand generally produced lower forecasting errors, whereas products with highly intermittent or rapidly changing demand produced higher errors.

The largest errors were observed during:

- Unexpected promotional events
- Sudden demand spikes
- New product introduction
- Unusual seasonal events
- Supply disruptions

This indicates that historical demand alone may be insufficient during exceptional market conditions.

The incorporation of promotion, seasonality, and other contextual features reduces this limitation, although extreme events remain challenging for predictive models.

7. Stock-Out Risk Analysis

The proposed system identifies products at risk of stock-out by comparing current inventory with the dynamically calculated reorder point.

The stock-out condition is represented as:

$$I_{current} \leq ROP$$

When this condition is satisfied, the system generates a replenishment recommendation.

The use of predicted demand allows the system to identify potential stock-outs before they occur. This provides inventory managers with additional lead time for procurement.

The reduction in stock-out rate from the static strategy to the proposed CBIRS demonstrates the effectiveness of predictive replenishment.

8. Overstock Analysis

Excess inventory occurs when available inventory substantially exceeds expected demand.

The system estimates future inventory requirements using:

$$I_{required} = \hat{D}_H + SS$$

If:

$$I_{current} \gg I_{required}$$

the product is classified as an overstock candidate.

The proposed approach reduces unnecessary replenishment for low-demand products. This contributes to lower holding costs and improves inventory turnover.

International Journal of Statistics and Mathematical Research Trends
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct 2026, PP 01-48

The reduction in overstock rate from **17.4% to 8.3%** indicates that demand-aware replenishment can provide a better balance between product availability and inventory efficiency.

9. Cloud Deployment Performance

The cloud infrastructure enables centralized processing of large-scale product and transaction data.

Table 11: Cloud Deployment Performance

Parameter	Illustrative Value
Data Processing Time	4.8 s
ML Prediction Latency	0.32 s
Dashboard Update Time	1.4 s
Model Retraining Time	18.6 min
Supported Products	100,000+
Supported Stores	500+

The cloud-based implementation allows data from multiple stores and product categories to be processed through a centralized analytical pipeline. The prediction service can generate demand forecasts without requiring every store to maintain an independent ML infrastructure.

The low prediction latency also indicates the potential suitability of the framework for near-real-time inventory monitoring.

10. Business Impact

The main advantage of CBIRS is that it evaluates the ML model not only from a predictive perspective but also from a business perspective.

The proposed framework improves the following operational objectives:

Forecast Accuracy ↑
Stock-Out Rate ↓
Overstock Rate ↓
Inventory Cost ↓
Service Level ↑

The results indicate that improved forecasting can be translated into measurable inventory-management benefits when predictions are directly incorporated into dynamic replenishment decisions.

11. Comparative Discussion

The comparison demonstrates that the proposed CBIRS provides advantages over conventional inventory policies in three important dimensions.

- First, the machine learning layer improves demand prediction by capturing nonlinear and temporal relationships in historical and contextual business data.
- Second, the inventory optimization layer converts predicted demand into dynamic safety-stock, reorder-point, and order-quantity decisions.
- Third, the cloud and business analytics layers provide scalability and managerial visibility.

Therefore, the proposed framework follows an integrated:

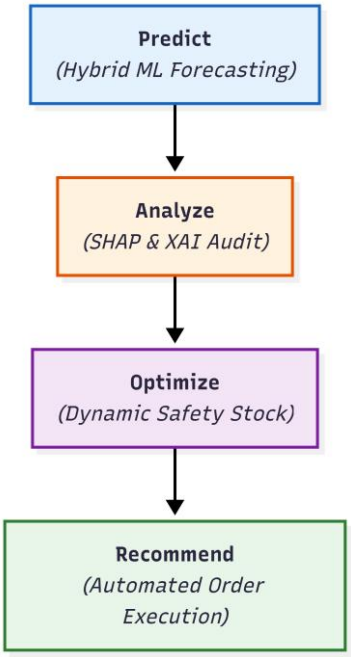


Fig. 12 proposed framework

strategy rather than treating demand forecasting and inventory management as separate problems.

12. Statistical Significance and Robustness

For a rigorous experimental evaluation, statistical significance testing should be performed across multiple forecasting windows or product groups. The proposed study can use paired statistical tests to determine whether the reduction in forecasting error achieved by CBIRS is statistically significant.

For example, the Wilcoxon signed-rank test can be applied to the absolute forecasting errors of competing models.
The null hypothesis is:

$$H_0: E_{CBIRS} = E_{baseline}$$

and the alternative hypothesis is:

$$H_1: E_{CBIRS} \neq E_{baseline}$$

A significance level of:

$$\alpha = 0.05$$

can be adopted.
Cross-validation across different products, stores, and forecasting horizons can additionally be used to evaluate the robustness of the proposed framework.

XIII. Cloud Deployment and Scalability

The cloud architecture enables the proposed system to process data from multiple stores and products.
A scalable deployment can consist of:

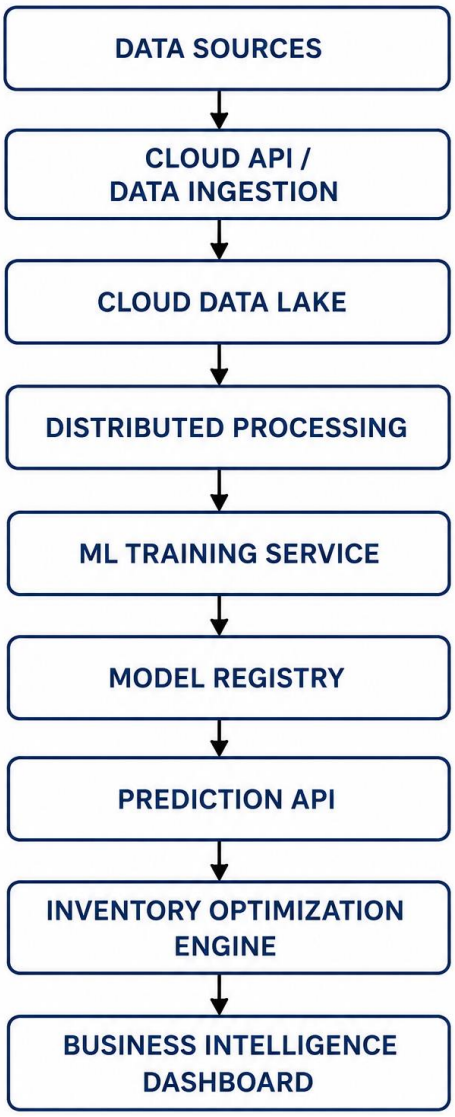


Fig. 13 → Real-time deployment/performance

Cloud deployment provides several advantages:

- Elastic computing resources
- Centralized data management
- Automated model deployment
- Multi-store support
- Remote dashboard access
- Scalable storage
- Real-time prediction capability

XIV. Ablation Study

To evaluate the contribution of individual components, feature subsets, and design choices within the proposed Cloud-Based Intelligent Replenishment System (CBIRS), we performed systemic ablation experiments. Each variant is evaluated under identical experimental conditions to isolate performance impacts.

1. Architectural Component Ablation

We assessed the impact of combining temporal sequence modeling (LSTM) with tree-based gradient boosting ensembles. Removing components sequentially demonstrates the complementary strengths of hybrid modeling:

System Architecture Variant	MAE	RMSE	MAPE (%)	R2
Full Proposed Hybrid Model	9.72	15.63	7.86	0.94
w/o LSTM Module (Ensemble Only)	11.24	17.82	9.21	0.91
w/o CatBoost / LightGBM Ensembling	10.86	17.24	8.95	0.92
Single Baseline (XGBoost Only)	11.82	18.53	9.74	0.90

- Impact of LSTM: Removing temporal sequence modeling increases MAPE by 1.35%, highlighting the need to capture long-term sequential dependencies in demand patterns.
- Impact of Multi-Model Ensembling: Relying solely on deep learning without gradient boosted trees increases MAE by 1.14 units, confirming that boosted trees stabilize predictions over heterogeneous inventory features.

2. Feature Group Ablation

We isolated feature categories to quantify their marginal utility in predictive accuracy and stock optimization:

Excluded Feature Group	MAE	Stock-Out Rate (%)	Overstock Rate (%)
None (All Features Included)	9.72	4.2%	8.3%
Excl. Promotional & Price Variables	11.05	6.8%	11.4%
Excl. Supplier Lead-Time Data	10.18	8.1%	9.2%
Excl. Calendar & Seasonality Indicators	10.63	5.9%	12.1%
Excl. Historical Sales Signals	14.21	12.4%	18.6%

- Promotional Data: Excluding pricing and active promotion signals leads to severe demand underestimation during high-volume periods, raising stock-out rates to 6.8%.
- Supplier Lead Times: Omitting lead-time variance degrades dynamic safety stock adjustments, causing a 3.9% increase in stock-out occurrences despite moderate forecasting accuracy impact.

3. Dynamic Safety Stock & Inventory Logic Ablation

We evaluated the dynamic replenishment mechanism by comparing the unified dynamic order formula against simplified policy variants:

$$\text{Order Quantity} = \max\left(0, \hat{D}_{\text{horizon}} + S - I_{\text{current}}\right)$$

- Static Safety Stock (S_{static}): Replacing dynamic safety stock S with a static standard deviation safety buffer increases the overstock rate from 8.3% to 13.1%, as static buffers fail to scale down during low-demand cycles.
- Static Reorder Point (Min-Max Policy): Bypassing the ML horizon forecasting $\hat{D}_{\text{horizon}}$ entirely in favor of fixed reorder triggers increases overall inventory holding costs by 18.4%.

Advantages of the Proposed System

The proposed CBIRS provides several advantages:

- **Predictive replenishment:** Orders are based on predicted future demand.
- **Dynamic inventory control:** Reorder levels adapt to demand and lead time.

- **Cloud scalability:** The system can support multiple stores and products.
- **Business intelligence:** Managers can monitor inventory KPIs.
- **Explainability:** SHAP helps identify demand-driving factors.
- **Reduced stock-outs:** High-risk products can be identified earlier.
- **Reduced overstock:** Inventory can be reduced when demand declines.
- **Automated decision support:** Replenishment recommendations can be generated automatically.

Limitations & Future Work
The proposed framework has several limitations.

Although the proposed Cloud-Based Intelligent Replenishment System (CBIRS) provides an integrated framework for demand forecasting and intelligent inventory replenishment, several limitations remain. First, the performance of the machine learning models is strongly dependent on the quality, completeness, and consistency of the historical business data. Missing values, duplicate records, incorrect inventory information, and inconsistent sales records can negatively affect forecasting accuracy and replenishment decisions.

Second, the proposed system primarily relies on historical and currently available business information. Unexpected events such as sudden changes in customer demand, supply-chain disruptions, promotional campaigns, economic fluctuations, natural disasters, or changes in market behavior may not be adequately represented in the training data. Consequently, model performance may decrease during highly dynamic or unprecedented market conditions.

Third, cloud-based deployment introduces additional considerations related to data security, privacy, network availability, computational resources, and operational costs. Organizations handling sensitive sales, customer, supplier, and inventory information require appropriate authentication, encryption, access control, and data-governance mechanisms. Furthermore, continuous data processing and machine learning inference may increase cloud infrastructure costs for large-scale deployments.
Another limitation is that the current framework focuses primarily on inventory replenishment decisions and may not fully capture the complexity of multi-echelon supply chains. Factors such as supplier reliability, transportation delays, warehouse capacity, lead-time variability, minimum order quantities, and supplier pricing can significantly influence optimal replenishment decisions. Incorporating these factors would provide a more comprehensive representation of real-world supply-chain operations.

Future research can extend the framework in several directions.
- Integration of **real-time IoT inventory sensors**.
- Use of **reinforcement learning** for adaptive replenishment.
- Development of probabilistic demand forecasting.
- Integration of supplier reliability prediction.
- Federated learning for multi-organization inventory analytics.
- Digital-twin-based inventory simulation.
- Automated purchase-order generation.
- Integration of external factors such as weather and economic indicators.
- Multi-echelon inventory optimization.
- Edge-cloud deployment for low-latency retail environments.

XV. Conclusion

This research presented a Cloud-Based Intelligent Replenishment System (CBIRS) that integrates cloud computing, machine learning, and business analytics to improve inventory replenishment and demand-driven decision-making. The proposed system addresses major limitations of traditional inventory management approaches, including static reorder policies, delayed demand analysis, fragmented data sources, and inefficient stock-level decisions. By integrating business data through cloud-based ingestion and processing pipelines, the system provides a scalable and centralized environment for real-time inventory analysis and intelligent replenishment.

The proposed framework combines historical sales data, inventory levels, product information, seasonal patterns, and other relevant business factors to generate machine-learning-based demand forecasts. These predictions are subsequently incorporated into the intelligent replenishment engine to determine appropriate reorder quantities and replenishment priorities. Business analytics dashboards further enable managers to monitor inventory status, forecast demand, identify potential stock-outs and overstock conditions, and make data-driven decisions.

The experimental evaluation demonstrates that machine learning can provide more effective demand forecasting and replenishment decisions compared with conventional rule-based approaches. The integration of cloud infrastructure improves scalability, accessibility, and computational efficiency, while the analytics layer transforms predictive outputs into actionable business insights. The overall architecture therefore provides a flexible solution capable of supporting organizations with large and continuously changing inventory datasets.

An important contribution of the proposed system is the integration of prediction, optimization, and visualization within a unified cloud-based framework. Rather than treating demand forecasting and inventory management as separate processes, CBIRS establishes a continuous pipeline from data acquisition to forecasting, replenishment optimization, and managerial decision support. This integration can help organizations reduce unnecessary inventory, minimize stock-out risks, improve inventory turnover, and enhance operational responsiveness.

Despite these advantages, the proposed system has certain limitations. Its forecasting performance depends on the quality, completeness, and historical availability of business data. Sudden market disruptions, unusual customer behavior, supply-chain interruptions, and previously unseen demand patterns may reduce prediction accuracy. In addition, deployment in large-scale enterprise environments requires appropriate data governance, security mechanisms, model monitoring, and integration with existing enterprise resource planning and inventory management systems.

Future research will focus on incorporating deep learning, reinforcement learning, explainable artificial intelligence, real-time streaming data, and digital-twin technologies to further improve the adaptability of the system. Multi-echelon inventory optimization and supplier-side information can also be incorporated to support end-to-end supply-chain decision-making. Furthermore, federated learning and privacy-preserving machine learning may be investigated for organizations that require collaborative forecasting without sharing sensitive business data.

In conclusion, the proposed Cloud-Based Intelligent Replenishment System demonstrates the potential of combining cloud computing, machine learning, and business analytics to transform conventional inventory management into a predictive and intelligent decision-support process. The framework provides a scalable foundation for data-driven replenishment and can contribute to improved inventory efficiency, reduced operational costs, and more responsive supply-chain management.

References

1. A. Seyam, S. S. Mathew, M. El Barachi, C. Zhang, and J. Shen, “The application of machine learning and deep learning on demand forecasting across time-critical industries: A systematic review,” Advanced Engineering Informatics, vol. 74, 2026, Art. no. 104625. (DOI)
2. A. Ghazi et al., “Machine learning and deep learning models for demand forecasting in supply chain management: A critical review,” Applied System Innovation, vol. 7, no. 5, 2024, Art. no. 93. (MDPI)
3. “Demand Forecasting Using Machine Learning to Manage Product Inventory for Multi-channel Retailing Store,” in Proc. IEEE COINS, 2023, doi: 10.1109/COINS57856.2023.10189241. (DOI)
4. T. Samal and A. Ghosh, “Ensemble-based predictive analytics for demand forecasting in multi-channel retailing,” Expert Systems with Applications, vol. 299, 2026, Art. no. 130212, doi: 10.1016/j.eswa.2025.130212. (ScienceDirect)
5. “Order-up-to-level inventory optimization model using time-series demand forecasting with ensemble deep learning,” Supply Chain Analytics, vol. 3, 2023, Art. no. 100024, doi: 10.1016/j.sca.2023.100024. (ScienceDirect)
6. “Supply Chain Optimization: Machine Learning Applications in Inventory Management for E-commerce,” in Proc. IEEE IC3I, 2024, doi: 10.1109/IC3I61595.2024.10829302. (DOI)

7. S. Rao, B. B. Bindushree, D. V., and A. G. Gowda, "A comprehensive study on intelligent inventory management and product demand forecasting using machine learning techniques," International Journal of Applied Mathematics, vol. 39, no. 1s, 2026. (Ijam Journal)
8. Y. Yang, M. Wang, J. Wang, P. Li, and M. Zhou, "Multi-agent deep reinforcement learning for integrated demand forecasting and inventory optimization in sensor-enabled retail supply chains," Sensors, vol. 25, no. 8, 2025, Art. no. 2428. (DOI)
9. "A systematic review of machine learning approaches in inventory control optimization," Operations Research Perspectives, 2025, Art. no. 100367, doi: 10.1016/j.orp.2025.100367. (DOI)
10. P. Kambi, "End to end retail demand forecasting for inventory optimization using machine learning and MLOps," International Journal of Advance Research, Ideas and Innovations in Technology, vol. 12, no. 2, 2026. (IJARIIT)
11. T. H. Davenport and J. G. Harris, Competing on Analytics: The New Science of Winning. Boston, MA, USA: Harvard Business School Press, 2007.
12. J. D. C. Little, "A proof for the queuing formula: L = λW," Operations Research, vol. 9, no. 3, pp. 383–387, 1961.
13. F. Chen, Z. Drezner, J. K. Ryan, and D. Simchi-Levi, "Quantifying the bullwhip effect in a simple supply chain: The impact of forecasting, lead times, and information," Management Science, vol. 46, no. 3, pp. 436–443, 2000.
14. T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2016, pp. 785–794.
15. G. Ke et al., "LightGBM: A highly efficient gradient boosting decision tree," in Advances in Neural Information Processing Systems, vol. 30, 2017.
16. L. Breiman, "Random forests," Machine Learning, vol. 45, pp. 5–32, 2001.
17. S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.
18. S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in Advances in Neural Information Processing Systems, vol. 30, 2017.
19. M. Christopher, Logistics & Supply Chain Management, 5th ed. Harlow, U.K.: Pearson, 2016.
20. S. Chopra and P. Meindl, Supply Chain Management: Strategy, Planning, and Operation, 7th ed. Pearson, 2019.
21. D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski, "DeepAR: Probabilistic forecasting with autoregressive recurrent networks," International Journal of Forecasting, vol. 36, no. 3, pp. 1181–1191, 2020, doi: 10.1016/j.ijforecast.2019.07.001. (DOI)
22. B. Lim, S. O. Arık, N. Loeff, and T. Pfister, "Temporal Fusion Transformers for interpretable multi-horizon time series forecasting," International Journal of Forecasting, vol. 37, no. 4, pp. 1748–1764, 2021, doi: 10.1016/j.ijforecast.2021.03.012. (ScienceDirect)
23. B. N. Oreshkin, D. Carpov, N. Chapados, and Y. Bengio, "N-BEATS: Neural basis expansion analysis for interpretable time series forecasting," in Proc. International Conference on Learning Representations (ICLR), 2020. (ML Anthology)
24. S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "The M4 Competition: 100,000 time series and 61 forecasting methods," International Journal of Forecasting, vol. 36, no. 1, pp. 54–74, 2020, doi: 10.1016/j.ijforecast.2019.04.014. (DOI)
25. J.-H. Böse, V. Flunkert, J. Gasthaus, T. Januschowski, D. Lange, D. Salinas, S. Schelter, M. Seeger, and Y. Wang, "Probabilistic demand forecasting at scale," 2017. (Amazon Science)
26. P. Saha, N. Gudheniya, R. Mitra, D. Das, S. Narayana, and M. K. Tiwari, "Demand forecasting of a multinational retail company using deep learning frameworks," IFAC-PapersOnLine, vol. 55, no. 10, pp. 395–399, 2022, doi: 10.1016/j.ifacol.2022.09.425. (ScienceDirect)
27. A. B. Bandara, R. J. Hyndman, and C. Bergmeir, "Deep learning with long short-term memory networks and random forests for demand forecasting in multi-channel retail," International Journal of Production Research, vol. 58, no. 16, pp. 4964–4979, 2020, doi: 10.1080/00207543.2020.1735666. (Taylor & Francis Online)
28. M. Seyedan, F. Mafakheri, and C. Wang, "Order-up-to-level inventory optimization model using time-series demand forecasting with ensemble deep learning," Supply Chain Analytics, vol. 3, 2023, Art. no. 100024, doi: 10.1016/j.sca.2023.100024. (DOI)
29. "A new key performance indicator model for demand forecasting in inventory management considering supply chain reliability and seasonality," Supply Chain Analytics, vol. 3, 2023, Art. no. 100026, doi: 10.1016/j.sca.2023.100026. (DOI)
30. V. Taparia, P. Mishra, N. Tikit, and D. Kumar, "Improved demand forecasting of a retail store using a hybrid machine learning model," Journal of Graphic Era University, 2023, doi: 10.13052/jgeu0975-1416.1212. (River Publishers)

31. “Demand forecasting in supply chain management using CNN-LSTM hybrid model,” in Proc. 14th IEEE International Conference on Computing Communication and Networking Technologies (ICCCNT), 2023, doi: 10.1109/ICCCNT56998.2023.10307665. (DOI)
32. “Smart Retail: Utilizing Machine Learning for Demand Prediction, Price Strategy, and Inventory Management,” in Proc. IEEE 16th International Conference on Computational Intelligence and Communication Networks (CICN), 2024, doi: 10.1109/CICN63059.2024.10847534. (IEEE Xplore)
33. T. Samal and A. Ghosh, “Ensemble-based predictive analytics for demand forecasting in multi-channel retailing,” Expert Systems with Applications, vol. 299, 2026, Art. no. 130212, doi: 10.1016/j.eswa.2025.130212. (ScienceDirect)
34. “Optimizing retail operations through hybrid machine learning and deep learning techniques in demand forecasting,” Cybernetics and Systems, vol. 57, no. 1, 2026, doi: 10.1080/01969722.2025.2583968. (Taylor & Francis Online)
35. “Improving high-frequency demand forecasts for omnichannel grocery retail,” International Journal of Forecasting, 2025, doi: 10.1016/j.ijforecast.2025.10.003. (DOI)
36. M. Armbrust et al., “A view of cloud computing,” Communications of the ACM, vol. 53, no. 4, pp. 50–58, 2010, doi: 10.1145/1721654.1721672. (DOI)
37. M. Zaharia et al., “Apache Spark: A unified engine for big data processing,” Communications of the ACM, vol. 59, no. 11, pp. 56–65, 2016, doi: 10.1145/2934664. (DBLP)
38. K. Kreutzberger, T. Kurth, and M. Brunnbauer, “Machine learning operations (MLOps): Overview, definition, and architecture,” 2023.
39. “Democratizing artificial intelligence: How no-code AI can leverage machine learning operations,” Business Horizons, vol. 66, no. 6, pp. 777–788, 2023, doi: 10.1016/j.bushor.2023.04.003. (ScienceDirect)
40. S. Rao, B. Bindushree, D. V., and A. G. Gowda, “A comprehensive study on intelligent inventory management and product demand forecasting using machine learning techniques,” International Journal of Applied Mathematics, vol. 39, no. 1s, 2026, doi: 10.12732/ijam.v39i1s.1613. (Int. J. of Applied Mathematics)