ISSN 2277-2685

IJESR/April-June. 2026/ Vol-16/Issue-2/967-974

Bollepally Vyshnavi *et. al.*, / International Journal of Engineering & Science Research

Full Length Article

TrendCast

Ms. B Tejaswi[1], Bollepally Vyshnavi[2], P Vidhatri[3]

[1]Assistant Professor; Department Of Computer Science And Engineering Bhoj Reddy Engineering College For Women Hyderabad India.

[2,3]B.Tech Students; Department Of Computer Science And Engineering Bhoj Reddy Engineering College For Women Hyderabad India.

Mail Id; vidhathripalakurthi@gmail.com[3]

Accepted 31-05-2026

Abstract

Demand forecasting plays a crucial role in strategic planning and decision-making across multiple industries, including retail, food services, dairy, supermarkets, and electronics. This project presents a multi-domain demand forecasting system designed to predict future product demand using historical time-series data.

The system processes structured datasets containing Date, Product, and Demand attributes. Data preprocessing is performed to clean and aggregate the data on a monthly basis, enabling better identification of long-term trends, seasonality, and demand patterns. Forecasting is carried out using advanced statistical and machine learning models such as ARIMA (AutoRegressive Integrated Moving Average) and Facebook Prophet, both of which are effective for time-series analysis.

The dataset is split into training and testing sets (80:20 ratio) to ensure model validation and reliability. Model performance is evaluated using standard error metrics, including Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE). The model with the best performance is selected for generating future demand predictions.

The system also provides a user-friendly interface that allows users to upload datasets, select specific products, and define the forecast horizon. The output includes forecasted demand values, visual graphs, and downloadable CSV reports.

This project delivers a scalable and efficient solution for inventory management, demand planning, and business decision-making across diverse domains.

Keywords: *Demand Forecasting, Time-Series Analysis, ARIMA, Prophet Model, Inventory Management, Machine Learning, Seasonal Trends, Predictive Analytics, Multi-Domain Forecasting, Business Intelligence.*

Introduction

TrendCast is an intelligent, web-based demand forecasting system designed to support businesses in accurately predicting future product demand by leveraging historical sales data. In today's highly competitive business environment, demand forecasting has become a critical component for effective inventory control, supply chain optimization, and long-term strategic planning. Organizations across various sectors, including retail, fast-moving consumer goods, manufacturing, and e-commerce, depend heavily on accurate demand estimates to manage stock levels, plan procurement cycles, and optimize resource utilization.

Conventional forecasting techniques, which are often based on manual calculations, simple statistical averages, or spreadsheet-driven models, are no longer sufficient to capture the complexity of modern demand behavior. These traditional approaches struggle to identify hidden patterns such as seasonality, trend variations, and cyclical fluctuations present in time-series data. As a result, they frequently produce forecasts that are either overestimated or underestimated, leading to inefficiencies such as overstocking or stockouts.

To overcome these limitations, TrendCast introduces an automated and data-driven forecasting framework that integrates advanced time-series prediction models, including ARIMA (AutoRegressive Integrated Moving Average), Prophet, and ETS (Exponential Smoothing). The system is designed to accept historical sales data in CSV format and performs a structured preprocessing pipeline to clean, transform, and organize the data for analysis. Multiple forecasting models are trained in parallel, and their performance is evaluated using the Mean Absolute Percentage Error (MAPE) metric. The model that achieves the best accuracy is automatically selected for final prediction.

The system ultimately generates reliable short-term forecasts, typically up to six months ahead, and presents the results through an interactive visualization dashboard. This enables users to easily interpret demand trends, compare model outputs, and make informed business decisions based on data-driven insights.

Requirements Analysis

Requirements analysis for the TrendCast system defines the functional, non-functional, and technical constraints necessary to achieve accurate, secure,

and efficient demand forecasting. These requirements are derived from stakeholder expectations, system objectives, and the specific characteristics of time-series forecasting applications.

Functional requirements describe the core capabilities of the system that enable users to interact with TrendCast effectively. The system provides a secure user authentication mechanism that allows new users to register using a unique email address and password, followed by secure login and session management throughout their usage. Users are able to upload historical sales datasets in CSV or Excel (.xlsx) formats through a web-based interface, where the system validates file structure, format correctness, and required data attributes before processing. Once uploaded, the system executes an automated preprocessing pipeline that handles missing values, removes invalid symbols such as currency characters, parses and validates date fields, performs temporal aggregation such as weekly or monthly resampling, applies smoothing techniques like Exponential Moving Average, and performs logarithmic transformation where required to stabilize variance.

After preprocessing, the system identifies unique product identifiers and allows users to select one or multiple products for forecasting analysis. The system then trains multiple forecasting models and evaluates their performance using error metrics such as MAE, RMSE, and MAPE. Based on these evaluations, the model with the lowest error—primarily MAPE—is automatically selected as the optimal forecasting model. Forecast outputs are presented through interactive visualizations built using Plotly, where historical demand trends are displayed alongside predicted values with 95% confidence intervals. Additionally, users can download forecast results in CSV format containing date, product name, predicted demand, confidence bounds, selected model information, and evaluation metrics, ensuring compatibility with external ERP and inventory management systems.

Non-functional requirements define the quality attributes of the system, including performance, accuracy, reliability, usability, and security. The system is designed to process datasets containing up to 10,000 records and generate forecasts within approximately five seconds. It is expected to achieve a Mean Absolute Percentage Error (MAPE) of 15% or lower on representative retail and FMCG datasets, corresponding to at least 85% forecasting accuracy. The system ensures reliability by producing consistent outputs for identical inputs and maintaining an availability level of at least 99% during operational use. Usability is emphasized through a simple and intuitive interface that allows non-technical users to complete key tasks such as uploading data, selecting products, viewing forecasts, and downloading reports with minimal interaction steps. Security is ensured through encrypted password storage using hashing mechanisms such as bcrypt, session invalidation upon logout, and restricted access to user-specific datasets. The overall system architecture follows a modular design, separating preprocessing, modeling, evaluation, and visualization components to improve maintainability and scalability.

The computational resource requirements include both hardware and software specifications necessary for efficient system operation. On the hardware side, a minimum configuration of an Intel Core i5 processor, 8 GB RAM, and 20 GB storage is required, while a recommended setup includes an Intel Core i7 processor, 16 GB RAM, and SSD storage for improved performance. A standard display resolution of 1366 × 768 is sufficient, although Full HD resolution is recommended. A stable internet connection is required for web-based access, while GPU support is optional and primarily useful for future deep learning extensions. On the software side, the system is compatible with Windows 10/11 or Linux distributions such as Ubuntu 20.04 and above. Python 3.12 or higher is used as the core programming language, along with Flask or Django for web development. MongoDB is used as the database system with PyMongo for integration. Forecasting is implemented using libraries such as statsmodels, pmdarima, and Prophet, while Plotly is used for visualization. Pandas and NumPy handle data processing, and the frontend is built using HTML5, CSS3, and JavaScript. Development is supported using Visual Studio Code, and the system is compatible with modern browsers such as Chrome and Firefox.

The system follows the Waterfall Software Development Life Cycle model, which is a linear and sequential approach in which each phase must be completed before the next begins. This model was chosen due to the clearly defined and stable requirements of the TrendCast system. The development process begins with requirement analysis, where system goals such as demand forecasting using historical data are defined. This is followed by the system design phase, where the architecture, data flow, and forecasting pipeline are structured. In the implementation phase, the preprocessing modules, forecasting models, and web interface are developed. The testing phase involves evaluating system performance and accuracy using an 80:20 training-testing split to ensure reliability. The system is then deployed in a local environment for demonstration purposes, allowing users to interact with the application and generate forecasts. Finally, in the maintenance phase, the system is continuously monitored, updated, and improved by fixing errors and enhancing performance as required.

Design

Architecture

System architecture represents the structural design of TrendCast, describing how different components interact to process data, generate forecasts, and present results to the user. It provides a high-level view of system organization and workflow.

Software architecture defines the logical structure of the system, while technical architecture specifies the underlying technologies and communication mechanisms used in implementation.

Software Architecture

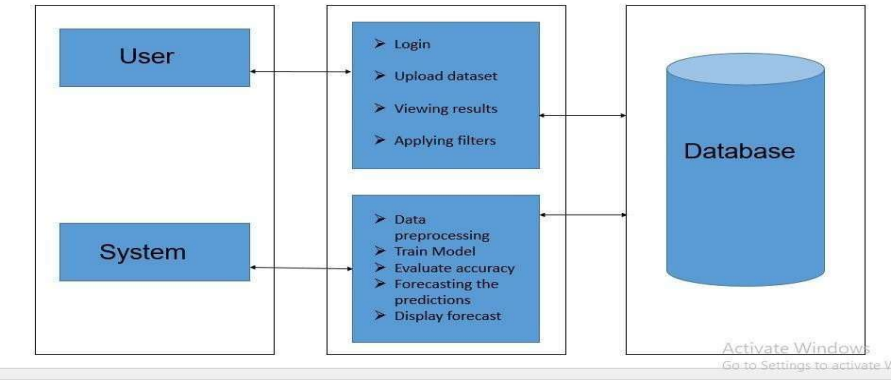

Fig. 1 Software Architecture Diagram

The software architecture of TrendCast follows a modular, layered design that separates the system into distinct functional components. This separation improves maintainability, scalability, and flexibility by ensuring that each module operates independently while contributing to the overall system functionality.The architecture consists of six main layers. The input layer receives validated time-series data from the user. The feature representation layer organizes and structures the data while preserving temporal relationships. The model training layer executes multiple forecasting algorithms, including ARIMA, Prophet, and ETS, in parallel. The model evaluation layer assesses model performance using MAE, RMSE, and MAPE metrics. The model selection layer identifies the best-performing model based on minimum forecasting error. Finally, the output layer generates six-month demand forecasts along with confidence intervals and prepares results for visualization and reporting.

Technical Architecture

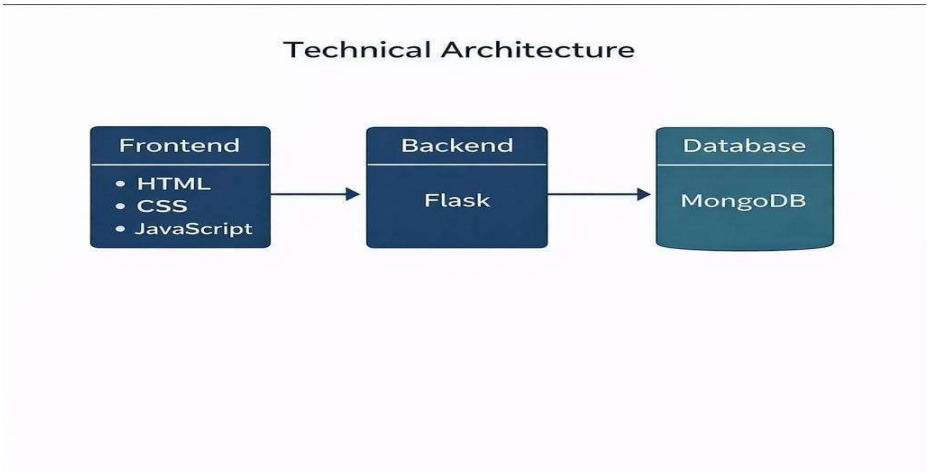

Fig.2 Technical Architecture Diagram

The technical architecture of TrendCast follows a three-tier architecture model, consisting of the presentation layer, application layer, and data processing layer.

The frontend layer is developed using HTML5, CSS3, and JavaScript, providing an interactive interface for uploading datasets, selecting products, and visualizing forecasts. Communication with the backend is handled through RESTful APIs using JSON format. Visualizations are rendered using Plotly.js for dynamic and interactive charting.

The backend layer is implemented using Python and the Flask framework. It handles authentication, dataset upload, preprocessing, model training, forecasting, and report generation. The backend is structured into modular components, including preprocessing modules, model training scripts, and database handlers, ensuring clear separation of responsibilities.

The system architecture supports scalability and maintainability by decoupling frontend interaction from backend computation, allowing independent updates to each layer without affecting overall system functionality.

Implementation

The TrendCast system is implemented using a Python-based technology stack carefully chosen to support scalable data processing, robust time-series forecasting, interactive visualization, and web-based deployment. Each component was selected based on its ability to handle specific requirements of the forecasting pipeline, including data preprocessing, model training, evaluation, and result visualization.

Python 3.12 is used as the primary programming language for backend development due to its extensive support for scientific computing, machine learning libraries, and web application development. Its simplicity and readability make it well-suited for implementing complex analytical workflows while maintaining code maintainability.

The web application layer is built using the Flask framework, which provides a lightweight yet powerful structure for handling HTTP requests, routing, session management, and API development. Flask enables seamless integration between frontend components and backend forecasting services, allowing efficient communication through RESTful APIs.

For data processing and manipulation, Pandas and NumPy are extensively used. These libraries facilitate efficient handling of structured datasets, including reading CSV and Excel files, cleaning missing or inconsistent values, parsing date fields, and performing temporal aggregation. Pandas resampling functions are particularly important for converting raw transactional data into consistent time-series formats required for forecasting models.

The forecasting module integrates multiple statistical and machine learning-based approaches. ARIMA modeling is implemented using the pmdarima library, where the auto_arima function is employed to automatically determine optimal parameters based on information criteria such as AIC. This allows efficient handling of datasets with linear temporal dependencies. Prophet, developed by Meta, is incorporated for its ability to model complex seasonal patterns, trend changepoints, and missing data scenarios. Additionally, the Exponential Smoothing (ETS) model from the statsmodels library is used to capture stable trend and seasonal components with reduced computational complexity.

For visualization, Plotly is used to generate interactive dashboards that display historical demand trends alongside forecasted values. Its interactive capabilities, such as zooming, hovering, and toggling data series, significantly enhance user understanding of forecasting outputs.

MongoDB, accessed through the PyMongo driver, is used as the database system for storing user data, uploaded datasets, and forecast results. Its flexible document-based structure is well-suited for handling heterogeneous time-series outputs and varying dataset schemas.

Dataset Description

The TrendCast system is evaluated using multiple real-world datasets representing diverse industry domains to ensure generalizability of the forecasting approach. These datasets include dairy products, electronics, and coffee chain sales data, each exhibiting different demand behaviors such as trend dominance, seasonal variation, and cyclic patterns.

Each dataset consists of three primary attributes: date, product, and demand. The datasets vary in size, ranging from approximately 8,000 to 12,000 records, and cover different time periods spanning 2018 to 2024. They also reflect real-world data quality challenges, including missing entries, inconsistent date formats, and currency-formatted values, all of which are addressed through preprocessing techniques within the system pipeline.

Data Preprocessing

Data preprocessing is a critical stage in the TrendCast pipeline, ensuring that raw input data is transformed into a clean and structured format suitable for forecasting models. The preprocessing process is fully automated and executed immediately after dataset upload.

Initially, the system performs heuristic detection of key columns by analyzing column names and matching them against predefined patterns for date, product, and demand fields. Once identified, missing or invalid records are removed to maintain data integrity. Currency symbols, commas, and other formatting artifacts are stripped from numerical fields, and date values are standardized using automated parsing techniques.

The cleaned dataset is then grouped based on product and date attributes, aggregating demand values to ensure a single observation per time period. Following this, the data is resampled into a

uniform weekly frequency to maintain consistency across datasets and domains.

To reduce noise and improve trend visibility, an Exponential Moving Average (EMA) smoothing technique is applied. Additionally, a log transformation is used to stabilize variance and improve compatibility with statistical forecasting assumptions.

Finally, the dataset is split into training and testing subsets using an 80:20 chronological split, ensuring that model evaluation is performed on unseen future data in a realistic forecasting scenario.

Forecasting Models

TrendCast employs a multi-model forecasting strategy in which three distinct models are trained independently and evaluated to determine the most suitable predictor for each dataset. This ensemble approach ensures adaptability across different demand patterns.

The ARIMA model is used to capture linear dependencies within stationary time-series data. It decomposes the series into autoregressive, differencing, and moving average components. The model parameters are optimized automatically using a stepwise search approach that minimizes the Akaike Information Criterion, making it effective for datasets with stable trend behavior.

Prophet is used to model complex time-series data that contains seasonal fluctuations, changepoints, and irregular patterns. It decomposes demand into trend, seasonality, and error components, using Fourier series to represent periodic behavior. Its robustness makes it particularly suitable for datasets with strong seasonal variation.

The ETS model applies exponential smoothing techniques, assigning higher weights to recent observations. This model is computationally efficient and performs well on datasets with smooth and gradually evolving trends. A damping factor is incorporated to prevent overestimation in long-term forecasts.

Model selection is performed using Mean Absolute Percentage Error (MAPE) computed on the test dataset. The model achieving the lowest MAPE is selected as the final forecasting model, ensuring objective and data-driven decision-making.

Testing

Software testing in TrendCast is conducted to ensure correctness, reliability, performance, and usability of the system. Testing validates that the system meets both functional and non-functional requirements and performs accurately under different conditions.

Black-box testing is used to evaluate system behavior from the user's perspective without considering internal implementation. This includes testing functionalities such as user authentication, dataset upload, forecast generation, visualization rendering, and report download.

White-box testing focuses on internal code structure and logic validation. It ensures correctness of preprocessing functions, forecasting algorithms, error calculation methods, and model selection logic by testing all execution paths and conditions.

Unit testing is performed using the pytest framework to validate individual functions in isolation. Key functions such as column detection, missing value handling, EMA computation, and MAPE calculation are tested with controlled inputs to ensure correctness of outputs.

Integration testing ensures that all system modules work together as expected. This includes verifying the complete workflow from data upload through preprocessing, model training, forecasting, and database storage using MongoDB. Dedicated test environments are used to avoid affecting production data.

System testing evaluates the entire application in a real-world usage scenario, ensuring compliance with functional and performance requirements. This includes testing the full web interface, API endpoints, and forecasting pipeline under realistic workloads.

User Acceptance Testing (UAT) is conducted with domain users such as inventory and supply chain analysts. Their feedback is used to improve usability, refine interface design, and enhance system output interpretability.

Testing Stages

Testing in TrendCast follows a structured five-stage process to ensure progressive validation of system quality. The process begins with unit testing of individual modules, followed by integration testing of combined components. System testing is then performed on the complete application in a controlled environment. This is followed by performance and accuracy evaluation using statistical error metrics such as MAE, RMSE, and MAPE. Finally, user acceptance testing is conducted to validate real-world usability and ensure that the system meets end-user expectations.

Test Cases and Results

Test cases are designed to validate both functional correctness and system reliability. These include authentication processes, dataset handling, preprocessing validation, forecasting accuracy, visualization rendering, and administrative operations. All core functionalities are verified to perform as expected under defined input conditions, with successful pass results indicating system stability.

Performance Metrics

The performance of forecasting models is evaluated using MAE, RMSE, and MAPE across three datasets representing different industrial domains. The results indicate that each model performs differently depending on data characteristics. ARIMA performs best on datasets with strong linear

trends, while Prophet shows superior performance on seasonal datasets. ETS performs adequately on stable time series but is less effective in complex demand scenarios.

Across all datasets, MAPE values range between 7.1% and 11.7%, indicating high forecasting accuracy. These results demonstrate that the system achieves forecasting accuracy levels between approximately 88% and 93%, exceeding the predefined requirement of 85%. The use of preprocessing techniques such as EMA smoothing and log transformation significantly improves model performance by reducing noise and stabilizing variance in the input data.

Screenshot

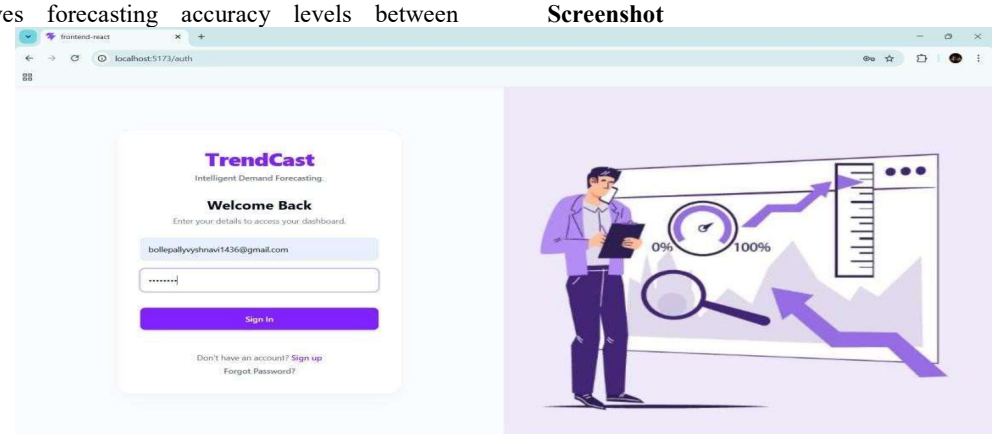

Fig 3; Application Home Page

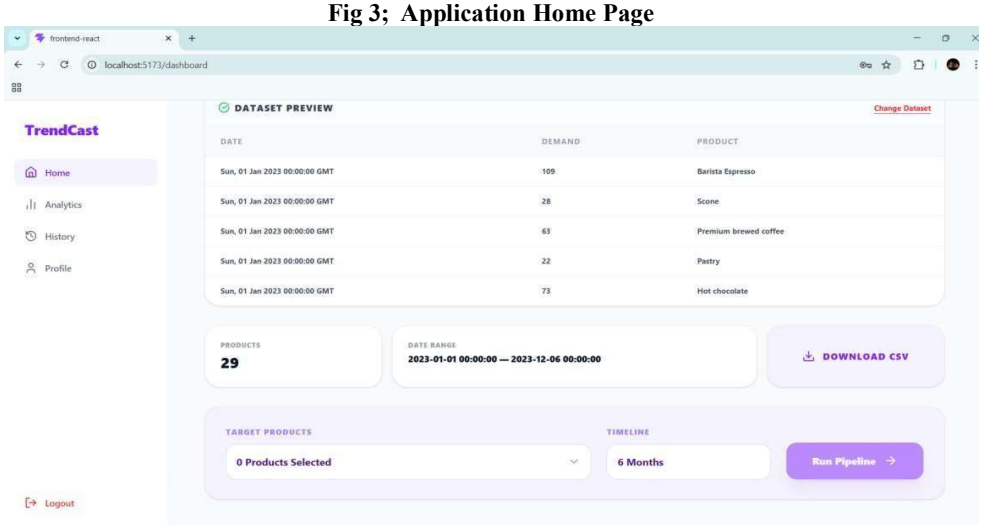

Fig 4; Dataset Upload Page

Fig 5; Forecast Chart – Line Chart

Fig 6; Multi-Product Forecast Visualisation

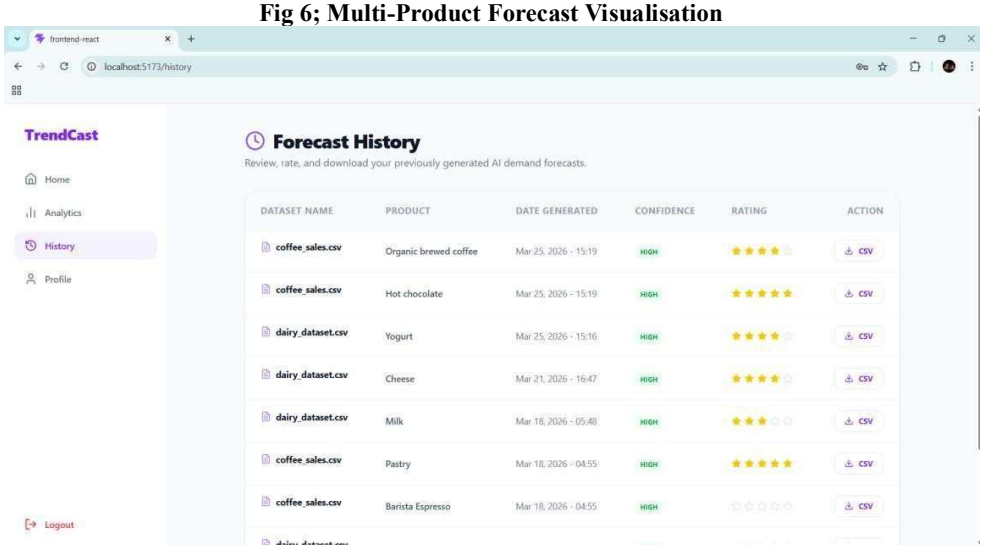

Fig 7; Forecast History

Conclusion

TrendCast demonstrates the practical effectiveness of an intelligent and automated demand forecasting framework built using a multi-model time-series machine learning approach. The system integrates three forecasting techniques—ARIMA, Prophet, and ETS—which together provide complementary capabilities for handling different demand patterns across diverse domains.

Experimental results indicate that each model performs optimally under specific conditions. The ARIMA model achieved strong results on datasets with clear linear trends, recording a MAPE of 7.1% for the dairy products dataset and 9.2% for the electronics dataset. In contrast, the Prophet model performed best on datasets with pronounced seasonal behavior, particularly the coffee chain dataset, where it achieved a MAPE of 8.3%. The ETS model provided stable performance for smoother demand series but was comparatively less effective for complex trend–seasonality interactions. The inclusion of an automated model selection mechanism based on minimum MAPE ensures that the system dynamically selects the most suitable forecasting model for each product without requiring manual intervention or domain expertise. Overall, the system achieved forecasting accuracy in the range of approximately 88% to 93% across all evaluated datasets, demonstrating strong predictive capability and consistency.

In summary, TrendCast provides a scalable and domain-independent forecasting solution that reduces reliance on manual forecasting techniques while significantly improving prediction accuracy. The system is suitable for practical deployment in inventory management and supply chain decision-making environments, where reliable demand estimation is essential for operational efficiency.

Future Scope

Although TrendCast achieves reliable forecasting performance within its current implementation, several enhancements can be explored to further extend its functionality and applicability in real-world industrial scenarios.

One important direction is the integration of deep learning-based forecasting models such as Long Short-Term Memory (LSTM) networks and Transformer-based architectures. These models are capable of capturing more complex temporal dependencies and may improve accuracy in highly non-linear demand environments.

Another potential improvement involves incorporating external or exogenous variables into the forecasting process. Factors such as weather conditions, promotional campaigns, economic indicators, and social media sentiment could significantly enhance prediction quality by providing additional contextual information beyond historical demand alone.

Scalability can also be improved through cloud deployment using platforms such as Amazon Web Services (AWS), Google Cloud Platform (GCP), or Microsoft Azure. This would enable the system to handle large-scale enterprise datasets involving thousands of products and continuous data updates.

Furthermore, real-time forecasting capabilities can be introduced by integrating streaming data pipelines using technologies such as Apache Kafka. This would allow the system to process incoming sales data continuously and generate dynamic, up-to-date forecasts.

Finally, the system can be enhanced with an automated anomaly detection module capable of identifying irregular demand patterns such as sudden spikes or drops. This would help improve data quality by allowing users to detect and correct anomalies before they negatively impact model training and forecasting accuracy.

References

1) J. Fattah, L. Ezzine, Z. Aman, H. El Moussami, and A. Lachhab, "Forecasting of Demand using ARIMA Model," *International Journal of Engineering Business Management*, vol. 10, pp. 1–9, 2023.
2) J. M. Chandran and M. R. B. Khan, "A Strategic Demand Forecasting: Assessing Methodologies, Market Volatility, and Operational Efficiency," *Malaysian Journal of Business, Economics and Management*, vol. 3, no. 1, pp. 34–52, 2024.
3) C. C. Holt, "Forecasting Seasonals and Trends by Exponentially Weighted Moving Averages," *International Journal of Forecasting*, vol. 20, no. 1, pp. 5–10, 2004.
4) A. C. Harvey, *Forecasting, Structural Time Series Models and the Kalman Filter*, Cambridge University Press, Cambridge, UK, 1989.
5) Statsmodels Development Team, "Statsmodels: Econometric and Statistical Modeling with Python," 2023.
6) Meta Open Source, "Prophet: Forecasting at Scale," GitHub Repository, 2023. Available: https://github.com/facebook/prophet (Accessed: 10 Feb. 2025).
7) Hyndman, R. J., & Athanasopoulos, G., *Forecasting: Principles and Practice*, OTexts, 2021.
8) Box, G. E. P., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M., *Time Series Analysis: Forecasting and Control*, Wiley, 2015.
9) Makridakis, S., Spiliotis, E., & Assimakopoulos, V., "The M4 Competition: Results, Findings, Conclusion," *International Journal of Forecasting*, 2020.
10) Taylor, S. J., & Letham, B., "Forecasting at Scale," *PeerJ Preprints*, 2017 (Prophet foundation research).