DEMAND PREDICTION USING AUTOML BASED ENSEMBLE ALGORITHM

PHASE I REPORT

Submitted by

MOHAMED HUSSAIN S 210701161

NATHANIEL ABISHEK A 210701173

In partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

ANNA UNIVERSITY, CHENNAI 600 025

November 2024

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

BONAFIDE CERTIFICATE

Certified that this Report titled **"DEMAND PREDICTION USING AUTOML BASED ENSEMBLE ALGORITHM"** is the bonafide work of **MOHAMED HUSSAIN S (210701161), NATHANIEL ABISHEK A (210701173),** who carried out the work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Dr. Kumar P, M.E., Ph.D.

Professor and Head,

Department of Computer Science and Engineering,

Rajalakshmi Engineering College,

Chennai – 602 105

SIGNATURE

Dr. Senthil Pandi S, M.Tech., Ph.D.

Associate Professor,

Department of Computer Science and Engineering,

Rajalakshmi Engineering College,

Chennai – 602 105

Submitted to Project Viva-Voice Examination held on _____

INTERNAL EXAMINER EXTERNAL EXAMINER

ACKNOWLEDGEMENT

Initially we thank the Almighty for being with us through every walk of our life and showering his blessings through the endeavour to put forth this report. Our sincere thanks to our Chairman **Mr. S. MEGANATHAN, B.E, F.I.E.**, our Vice Chairman **Mr. ABHAY SHANKAR MEGANATHAN, B.E., M.S.,** and our respected Chairperson **Dr. (Mrs.) THANGAM MEGANATHAN**, **Ph.D.,** for providing us with the requisite infrastructure and sincere endeavouring in educating us in their premier institution.

Our sincere thanks to **Dr. S.N. MURUGESAN, M.E., Ph.D.,** our beloved Principal for his kind support and facilities provided to complete our work in time. We express our sincere thanks to **Dr. P. KUMAR, M.E., Ph.D.**, Professor and Head of the Department of Computer Science and Engineering for his guidance and encouragement throughout the project work. We convey our sincere and deepest gratitude to our internal guide, **Dr. S. Senthil Pandi, M.Tech., Ph.D.**, Department of Computer Science and Engineering. Rajalakshmi Engineering College for her valuable guidance throughout the course of the project. We are glad to thank our Project Coordinator, **Dr. T. Kumaragurubaran, M.E., Ph.D**, Department of Computer Science and Engineering for his useful tips during our review to build our project.

MOHAMED HUSSAIN S (210701161)

NATHANIEL ABISHEK A (210701173)

ABSTRACT

The integration of AutoML and ensemble techniques plays a pivotal role in enhancing demand forecasting by automating essential processes such as model selection, data cleaning, and hyperparameter tuning. These automated features streamline the machine-learning pipeline, significantly reducing the manual effort involved in model building and ensuring faster model deployment. By combining the outputs of multiple models through ensemble techniques, the approach leverages the strengths of each model, leading to improved accuracy and robustness in demand predictions. This system generates highly accurate demand estimates, which are crucial for optimizing operations like inventory management, resource allocation, and supply chain planning. Sectors like e-commerce, retail, and tourism, where demand fluctuations are common, can particularly benefit from such precise predictions. The automation also enables businesses to make well-informed decisions without needing in-depth expertise in machine learning, allowing for quicker responses to market dynamics. Incorporating real-time data is a key area for future development. This would allow for more dynamic forecasting that responds immediately to changes in consumer behavior or external factors, further improving the accuracy of predictions. Additionally, tailoring AutoML systems to address the unique challenges of different industries, such as considering sector-specific variables and data patterns, will make the technology even more effective. Overall, the combination of AutoML and ensemble techniques enhances decision-making by offering scalable, accurate, and efficient demand forecasting solutions across industries.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVATIONS

AutoML	Automated Machine Learning
EDA	Exploratory Data Analysis
NLP	Natural Language Processing
MAE	Mean Absolute Error
RSME	Root Mean Square Error
RF-SVR	Random Forest-Support Vector Regression
sMAPE	Symmetric Mean Absolute Percentage Error
SPL	Scaled Pinball Loss
LSTM	Long Short-Term Memory
SVR	Support Vector Regression
BMA`	Bayesian Model Averaging
SME	Small- and Medium-sized Enterprises

CHAPTER 1

1.INTRODUCTION

1.1 GENERAL

In recent years, advancements in machine learning (ML) and automation have revolutionized demand prediction, making it more accessible, efficient, and accurate. Automated Machine Learning (AutoML) has emerged as a transformative tool, streamlining the complex process of building machine learning models. Traditionally, tasks such as selecting the right algorithm, tuning hyperparameters, and preparing data required substantial expertise and manual effort. AutoML automates these steps, enabling businesses to develop and deploy sophisticated models with minimal human intervention. This democratization of machine learning allows organizations to utilize cutting-edge technologies without requiring in-depth technical knowledge, significantly reducing time-to-deployment and operational costs.

Ensemble algorithms, on the other hand, enhance the predictive power of machine learning systems by combining the strengths of multiple models. Each model in an ensemble contributes unique insights, mitigating the weaknesses of individual models. Techniques like bagging (e.g., Random Forests) improve stability by averaging predictions across multiple models, while boosting (e.g., XGBoost, LightGBM) incrementally refines predictions by correcting errors from prior models. Stacking, another ensemble approach, uses a meta-model to synthesize predictions from diverse base models, further enhancing accuracy. By leveraging the collective intelligence of multiple algorithms, ensemble techniques ensure more robust and reliable demand forecasts.

When combined, AutoML and ensemble algorithms create a powerful framework for addressing complex forecasting challenges. AutoML simplifies and automates the end-to-end machine learning process, while ensembles ensure the predictive system is both accurate and resilient. This synergy allows businesses to process large and intricate datasets more effectively, uncovering patterns that would be difficult to

detect manually. Moreover, these tools enable organizations to generate scalable and precise forecasting solutions, adaptable to various industries and market conditions.

The integration of these technologies has a profound impact on business operations. For instance, precise demand forecasts help optimize inventory management by minimizing overstocking and stockouts, reducing associated costs. Resource allocation also benefits, as accurate predictions inform better workforce and supply chain planning during peak demand periods.

In conclusion, the advancements in AutoML and ensemble algorithms mark a significant step forward in demand prediction. By automating the creation of accurate and reliable models, these technologies empower businesses to harness the power of machine learning without requiring extensive technical expertise.

1.2 OBJECTIVE

The objective of this project is to develop an advanced and automated demand prediction system leveraging AutoML and ensemble techniques to provide precise, actionable insights for various business applications. The system aims to:

1.Model Selection and Optimization:

Utilize AutoML frameworks toautomate the selection, training, and hyperparameter tuning of machine learning models, reducing the dependency on manual efforts and domain expertise.

2.Enhance Prediction Accuracy:

Integrate ensemble techniques such as Random Forests, XGBoost, and LightGBM to combine the strengths of individual models, delivering improved accuracy and robustness in demand forecasts.

3.Enable Real-Time Forecasting:

Incorporate real-time data processing capabilities to predict demand trends dynamically, adapting quickly to changes in consumer behaviour and market conditions.

4.Facilitate User Interaction:

Implement a natural language processing (NLP) interface to allow users to interact with the system using intuitive, query-based text or voice commands, enhancing accessibility for non-technical users.

5.Support Data Analysis and Visualization:

Provide tools for analyzing historical and forecasted demand trends, enabling users to identify patterns, seasonal variations and anomalies through graphical and tabular visualization.

1.3 EXISTING SYSTEM

Current demand prediction systems rely on a combination of traditional statistical methods, machine learning algorithms, and enterprise tools tailored for specific industries. Statistical models, such as Linear Regression and ARIMA (Auto-Regressive Integrated Moving Average), are commonly employed for analyzing historical data and identifying trends. These models are particularly effective for straightforward, stationary datasets, making them widely used in sectors like retail, logistics, and manufacturing to forecast sales or inventory requirements. Similarly, decision trees and rule-based approaches are utilized for relatively simple forecasting tasks, where decision-making can be guided by predefined thresholds or historical averages.

Machine learning models, such as Random Forests and Support Vector Machines (SVM), have found applications in scenarios involving non-linear patterns and moderately complex datasets. These models are often used in e-commerce, healthcare, and energy sectors to predict customer behavior, electricity demand, or resource requirements. Enterprise Resource Planning (ERP) systems and rule-based forecasting systems are also prominent in industries like supply chain and distribution, automating routine forecasting tasks and integrating demand predictions with other business processes.

While these systems serve specific purposes effectively, they face several limitations when it comes to handling the complexity of modern data. Many traditional models, including statistical ones like ARIMA, struggle to adapt to large-scale datasets or dynamic, real-time inputs. Similarly, manual efforts required for preprocessing data and tuning machine learning models make them resource-intensive and dependent on expert knowledge. Moreover, rule-based systems lack the flexibility to account for unforeseen factors like sudden market changes or demand spikes.

Despite their limitations, these systems have provided a foundation for demand prediction across industries. However, the evolving complexity of datasets, combined with the need for scalability, real-time adaptability, and improved accuracy, highlights the demand for more advanced and automated solutions.

1.4 PROPOSED SYSTEM

The proposed system leverages Automated Machine Learning (AutoML) and Natural Language Processing (NLP) to create an advanced, intelligent demand prediction framework. This system automates key aspects of data processing, model training, and interaction, making it highly accessible and effective for diverse business applications. The core components of the system are as follows:

Data Collection and Preparing:

The system begins by collecting sales data from various sources such as supermarket transactions, online sales platforms, and retail outlets. These datasets typically include fields like product IDs, transaction dates, sales volumes, and customer demographics. Once collected, the data is structured into formats like CSV or Excel for seamless ingestion into the system.A comprehensive preprocessing phase is employed to enhance data quality and consistency. Missing values are addressed using imputation techniques, categorical data is encoded through methods like one-hot encoding or label encoding, and numerical fields are normalized to ensure

uniform scaling. Uniform date-time formats are enforced to streamline time-series analysis. Additionally, automated Exploratory Data Analysis (EDA) generates insights into correlations, outliers, and data distributions. These steps ensure that the dataset is clean, structured, and optimized for AutoML training. For user-uploaded datasets, the system applies the same preprocessing pipeline, allowing businesses to utilize their custom data without additional manual intervention.

AutoML Model Development

The system's AutoML engine forms the backbone of demand forecasting by automating the exploration and development of machine learning models. A variety of algorithms, including Random Forests, XGBoost, and LightGBM, are evaluated to find the best-fit models for the given dataset. The system dynamically tailors the ensembling process to the characteristics of the data, selecting the most suitable algorithms based on factors like dataset size, feature complexity, and distribution patterns.

For instance, if the dataset exhibits high dimensionality, tree-based methods like XGBoost might be prioritized for their ability to handle complex relationships. Conversely, if time-series trends dominate, models like ARIMA or Gradient Boosting might be included in the ensemble. The AutoML framework ensures optimal hyperparameter tuning and evaluates model performance using metrics such as Mean **Absolute Error (MAE)** and **Root Mean Square Error (RMSE)**.

The final ensemble model combines predictions from multiple algorithms, leveraging their individual strengths to improve overall accuracy and robustness. By dynamically adjusting to the dataset's characteristics, the system ensures that the ensemble is tailored for maximum predictive performance.

Query understanding and NLP Integration:

The system integrates Natural Language Processing (NLP) capabilities to facilitate intuitive, query-based interactions. Users can interact with the system through natural language inputs, including text and voice commands. Advanced NLP techniques like

tokenization, Named Entity Recognition (NER), and intent classification enable the system to parse queries and extract relevant entities, such as product names, timeframes, or conditions.

For example, a user may ask, “What will the sales for Product A be next month?” or “Why were the sales low in August?” The NLP module identifies key terms like "Product A," "next month," and "sales low in August," enabling the system to generate precise responses. The NLP interface also supports multilingual capabilities, allowing users to interact in both English and regional languages, further broadening accessibility.

Demand Prediction and Analysis:

After processing the query, the system retrieves historical data or generates forecasts using the trained ensemble model. For predictive queries like "What will the sales be next month?", the model outputs specific demand predictions for the requested product or time period. For diagnostic queries such as "Why were sales low in August?", the system analyzes factors like pricing, promotions, competitor activity, or external events to identify possible causes.

This analysis provides users with actionable insights into historical trends and anomalies. The ability to correlate multiple variables—such as customer demographics, seasonal demand patterns, and product pricing—enables a deeper understanding of the factors influencing sales performance.

Query-Based Interaction and Response Generation

A distinguishing feature of the system is its dynamic, query-based interaction model. When a user poses a query, the system generates a structured request using technologies like SQL or MongoDB to retrieve relevant data or predictions. For instance, in response to a query like “What is the forecasted demand for Product B in November 2024?”, the system returns an output such as “The forecasted demand for Product B in November 2024 is 500 units.”

For diagnostic queries, the system highlights key factors affecting sales, such as seasonal trends, price changes, or external events. Responses are provided in a user-friendly format, combining text summaries with visual representations like charts and graphs to enhance interpretability. These visualizations include trend lines, heatmaps, and comparisons, enabling users to grasp insights quickly.

By automating data handling, model training, and prediction generation, the proposed system delivers a comprehensive and adaptable solution for demand forecasting. Its integration of AutoML and NLP ensures that predictions are accurate, insights are actionable, and the system remains accessible to both technical and non-technical users. Through dynamic ensembling and real-time adaptability, the system is poised to address diverse forecasting needs across industries.

By automating data handling, model training, and prediction generation, the proposed system delivers a comprehensive and adaptable solution for demand forecasting. Its integration of AutoML and NLP ensures that predictions are accurate, insights are actionable, and the system remains accessible to both technical and non-technical users. Through dynamic ensembling and real-time adaptability, the system is poised to address diverse forecasting needs across industries.

CHAPTER 2

2. LITERATURE SURVEY

S. Mhatre, et al., [1] This study introduces an AutoML-based approach for tourism prediction and revenue maximization, emphasizing automation, scalability, and efficiency in forecasting systems. It highlights the adaptability of AutoML techniques to diverse datasets, enabling better decision-making through automated analysis. These insights directly support your project's objectives by validating the role of AutoML in streamlining dataset preprocessing and automating demand forecasting.

T. Nagarajah and G. Poravi [2] This review provides an extensive overview of various AutoML systems, emphasizing their ease of use, scalability, and integration capabilities. Their emphasis on simplifying model selection processes directly supports your demand prediction system's objective of using AutoML to reduce manual intervention in choosing optimal algorithms while enhancing scalability for diverse user datasets.

P. Kumar, et al., [3] The authors explore how ensemble learning can optimize credit scoring, emphasizing the critical role of combining algorithms for reliability and precision. Beyond accuracy, the study highlights scalability and adaptability to diverse financial datasets. This directly aligns with your project’s goals of integrating multiple models for robust demand forecasting across various industries. The ensemble strategies used in this study further provide insights into tailoring models to improve outcomes, supporting your objective to deliver customized, actionable insights for demand management.

D. Mallikarachchi, et al., [4] The authors use AutoML for predicting Type 2 diabetes and showcase comparative techniques for identifying optimal models. Their methodology emphasizes dataset-specific model selection, mirroring your workflow of identifying the best-performing models through AutoML for demand prediction. This approach strengthens the system's ability to adapt to user-specific datasets.

A. Ghareeb, et al., [5] This paper explores the application of ensemble learning in electricity demand forecasting, highlighting its ability to enhance short-term prediction accuracy. It demonstrates how combining different algorithms can result in robust models, a principle directly applied in your project's ensemble modeling phase to improve forecasting performance.

Y. Jin, et al., [6] This work utilizes stacking ensemble learning to tackle the challenges of large-scale, online car-hailing demand forecasting. By addressing the scalability and accuracy requirements of real-time demand prediction, the study provides a strong foundation for your project's goal of managing large, diverse datasets effectively.

V. E. Kovalevsky and N. A. Zhukova [7] The paper demonstrates the adaptability of AutoML in time-series forecasting tasks, particularly for dynamic and continuously changing data. This supports your project's focus on building a flexible demand prediction system that can accommodate fluctuating user requirements and dataset complexities.

H. -A. -D. Cap, et al., [8] A case study on heart rate prediction highlights AutoML's utility in analyzing sequential data. The outlined methodologies directly support your system's approach to using AutoML for analyzing time-series data related to demand forecasting.

I. Met, et al., [9] Performance efficiency is a key focus in this study on time-series forecasting for bank branch data. By emphasizing reduced latency and increased processing speed, it aligns with your project's goal of delivering fast, real-time demand predictions with minimal processing overhead.

G. Stamatescu, et al., [10] This study leverages anomaly detection through AutoML in residential power traces to ensure prediction reliability. Your project applies similar strategies to handle anomalies and maintain the integrity of demand prediction outputs in the presence of irregularities.

H. Iftikhar, et al., [11] The introduction of novel ensemble techniques for electricity demand forecasting demonstrates the effectiveness of combining machine learning models to achieve higher accuracy. This validates your project's use of ensemble learning to improve forecasting precision.

Q. Lyu and R. Zhang [12] This paper investigates ensemble learning methods, particularly for shared bicycle demand forecasting, to improve prediction accuracy by combining the strengths of multiple models. The demonstrated benefits of ensemble learning align with your project's plan to implement advanced ensemble techniques, allowing for improved demand prediction accuracy across various contexts.

Y. Zhang, et al., [13] The transition from traditional machine learning to ensemble learning for demand forecasting highlights the evolution of predictive models. This directly supports your methodology of integrating advanced ensemble techniques to enhance prediction accuracy.

D. Hulak and G. Taylor [14] A revisit to ARIMA models for short-term electricity demand forecasting provides insights into the hybridization of traditional and advanced techniques. These insights complement your project's approach of leveraging the best features of AutoML-selected and ensemble models to create a hybrid system.

P. Naik, et al., [15] Automated ensemble modeling for biomass prediction emphasizes the value of stacking multiple models for complex datasets. This directly supports your methodology of using stacked ensemble techniques to address demand variations effectively.

A. Garg and A. Chaudhary [16] By applying AutoML and Lime for IPL auction dataset analysis, this paper emphasizes the importance of interpretability in machine learning models. Similarly, your project aims to provide explainable outputs in demand forecasting to help users make informed decisions.

S. P. Menon, et al., [17] Brain tumor diagnosis using AutoML showcases the power of automation in achieving precision and scalability. These principles align with your project's goal of automating demand prediction processes while ensuring high accuracy.

P. Kumar, et al., [18] The application of ensemble learning for market basket analysis emphasizes its scalability and effectiveness in identifying patterns for retail optimization. Beyond identifying consumer trends, this study highlights practical methodologies for improving resource allocation and inventory management. These insights align with your project's vision of providing businesses with actionable demand forecasts, supporting long-term strategic planning and operational efficiency through advanced machine learning techniques.

K. Han, et al., [19] This research introduces the use of genetic algorithms to optimize ensemble model selection, highlighting their application in refining multi-layered systems. The genetic algorithm-driven approach supports your efforts to refine and optimize model combinations, ensuring the ensemble framework adapts dynamically to diverse and evolving datasets for accurate demand prediction.

CHAPTER 3

3. SYSTEM DESIGN

3.1 GENERAL

3.1.1 SYSTEM FLOW DIAGRAM

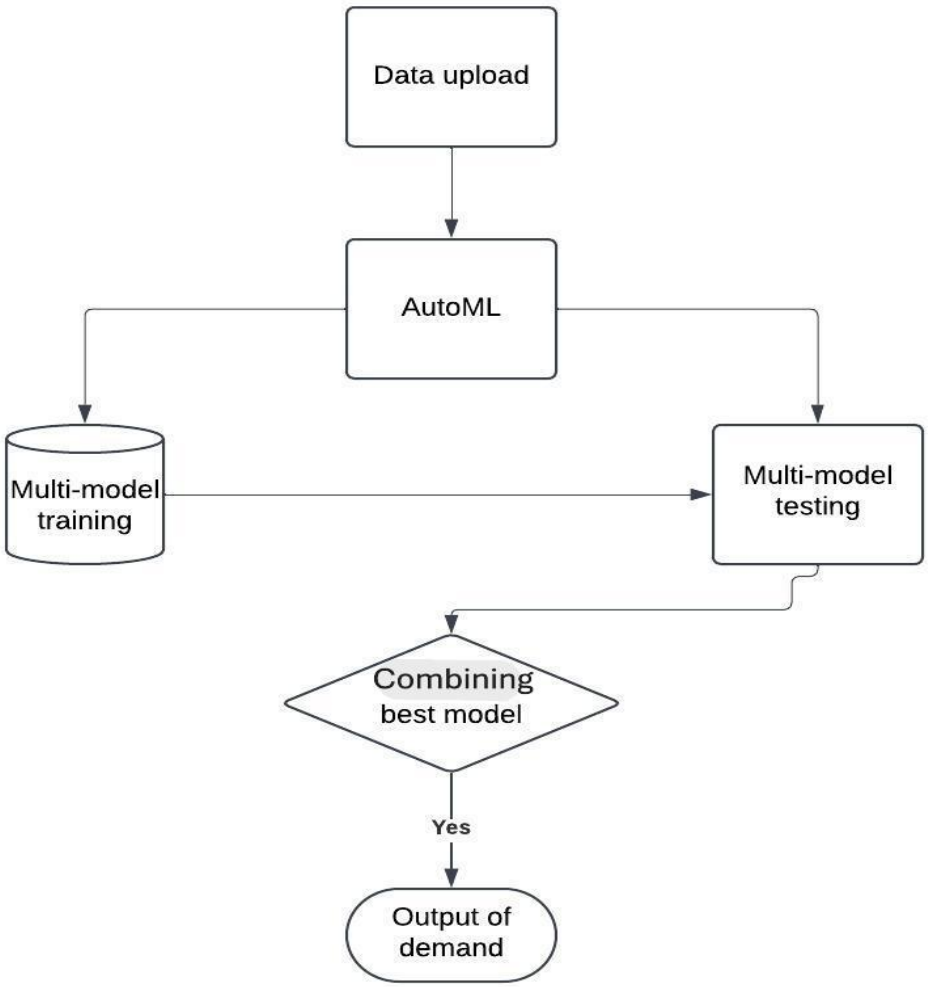

Figure 1 System Flow Diagram

The system flow diagram presented illustrates the workflow of a demand prediction system. It begins with the input data upload, where datasets are provided by users. The uploaded data is processed using AutoML, which automatically handles preprocessing, feature engineering, and model selection. Following this, the system performs multi-model training to train various machine learning models and multi-

model testing to evaluate their performance. The results from these stages are used to identify and combine the best-performing models into an ensemble for higher prediction accuracy. Finally, the ensemble model generates the demand prediction output, providing actionable insights for users.

3.1.2 SEQUENCE DIAGRAM

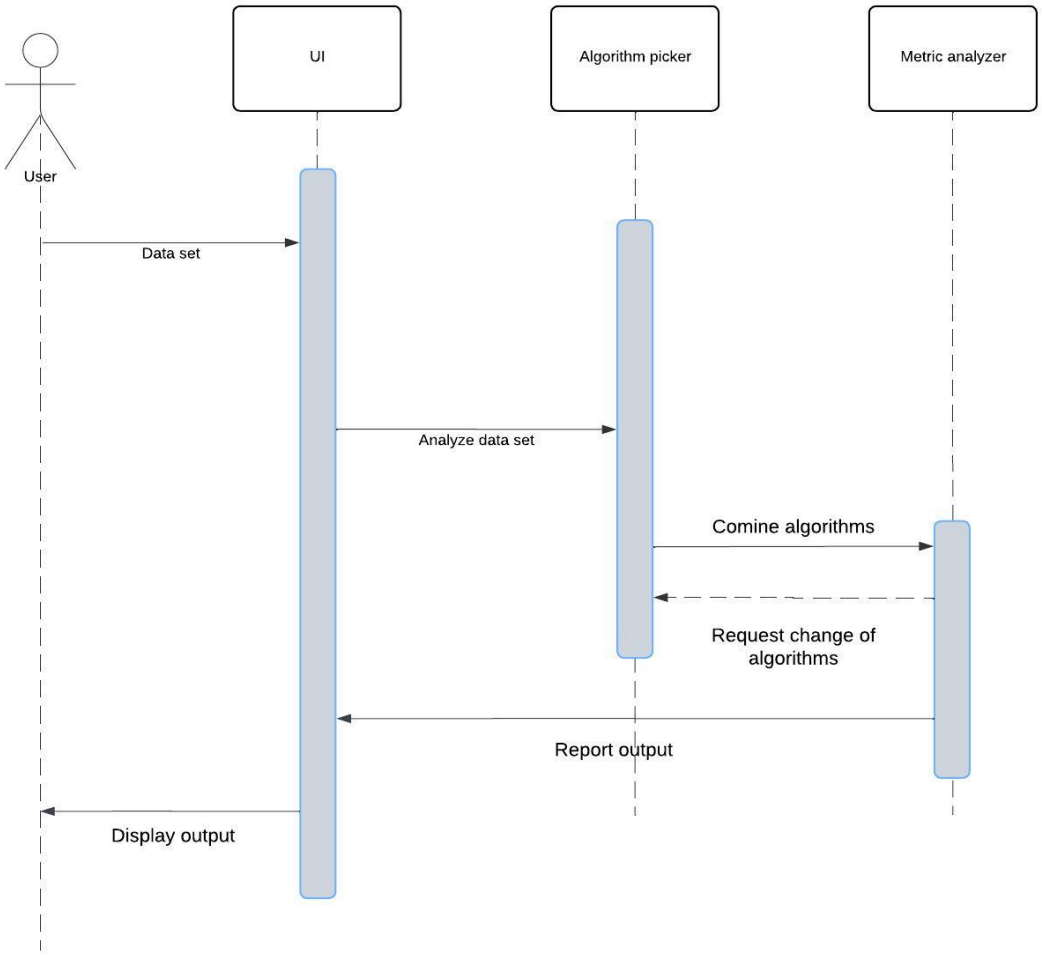

Figure 2 Sequence Diagram

The Sequence Diagram illustrates the workflow of the Demand Prediction System, starting with the user uploading a dataset. The data undergoes preprocessing to ensure it is clean and ready for analysis before being passed to the AutoML module, which selects and fine-tunes the best-performing models. These models are then trained,

evaluated, and combined using ensembling techniques to enhance accuracy and reliability. The final ensemble model generates demand predictions, which are displayed to the user through an intuitive interface, providing actionable insights for informed decision-making.

3.1.3 CLASS DIAGRAM

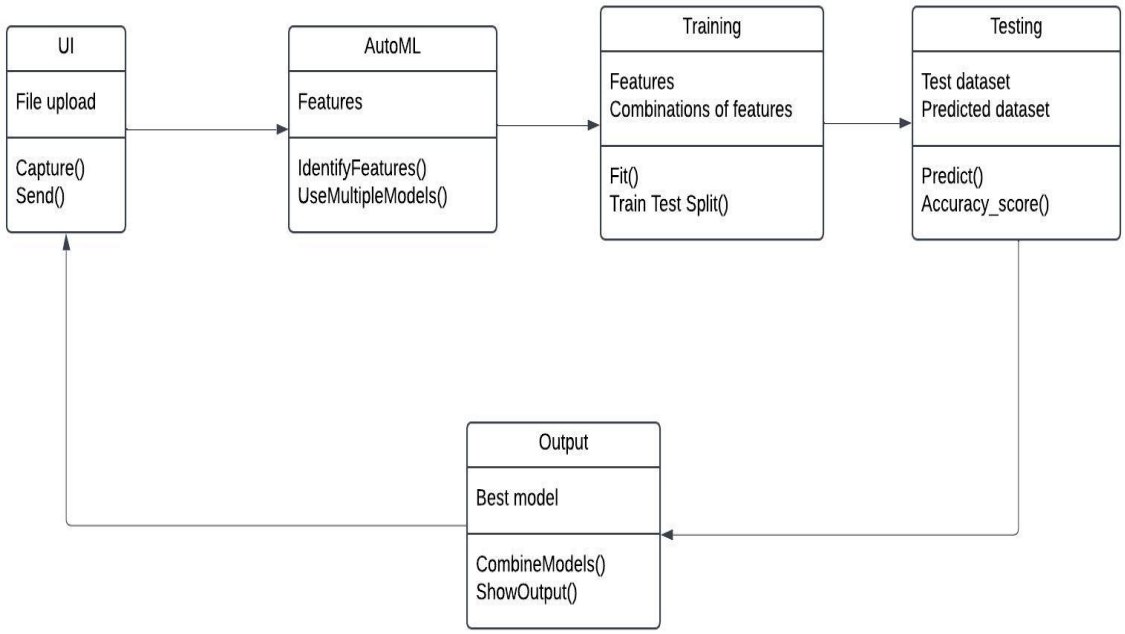

Figure 3 Class Diagram

The Class Diagram provides an overview of the key components in the Demand Prediction System, including the Dataset for raw data, the Preprocessor for data cleaning, the AutoML Engine for model selection and training, the Ensemble Module for combining top models, and the Evaluator for assessing model performance. These components interact seamlessly to process data, optimize models, and generate accurate demand predictions for the user.

3.1.4 USECASE DIAGRAM

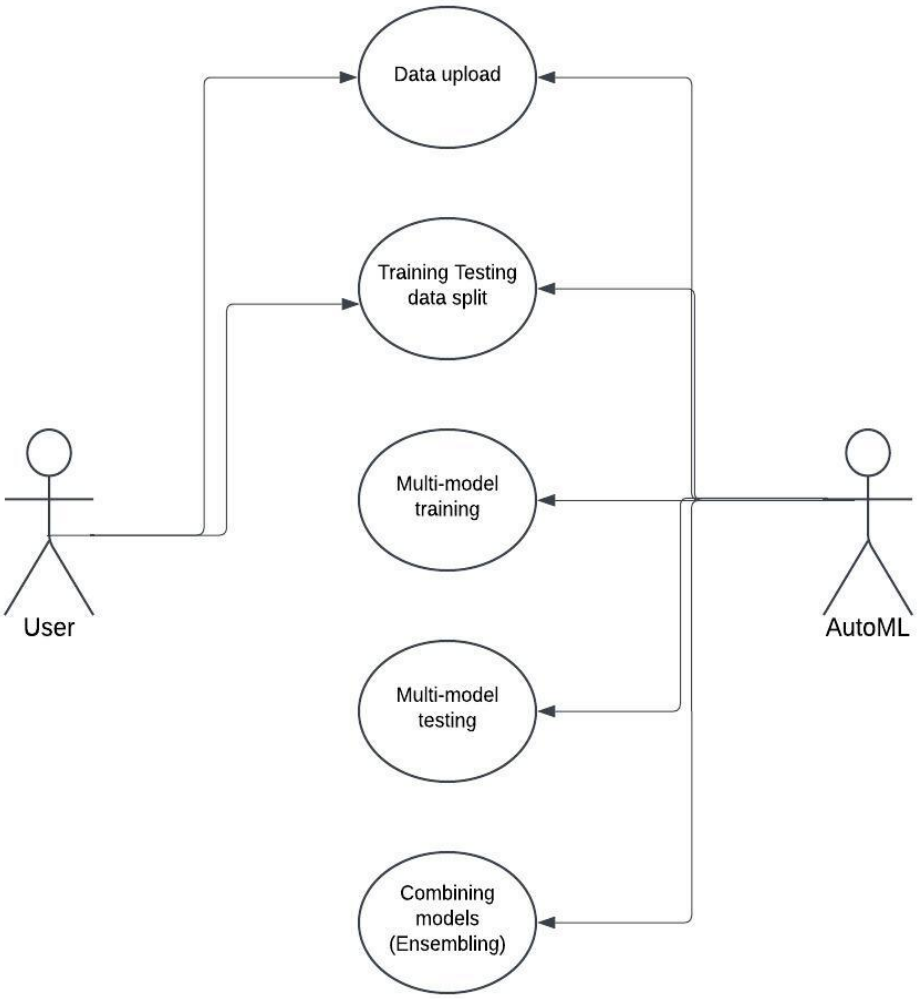

Figure 4 Use Case Diagram

The Use Case Diagram illustrates the interactions between different types of users and the Demand Prediction System, emphasizing core functionalities like uploading datasets, automated data preprocessing, and model selection through the AutoML engine. By visualizing the user-centric operations, the diagram ensures a streamlined and intuitive workflow that guides users through the entire process—from data input to the delivery of accurate and actionable demand forecasts. This approach is designed to enhance user experience and enable efficient, reliable decision-making for demand forecasting.

3.1.5 ARCHITECTURE DIAGRAM

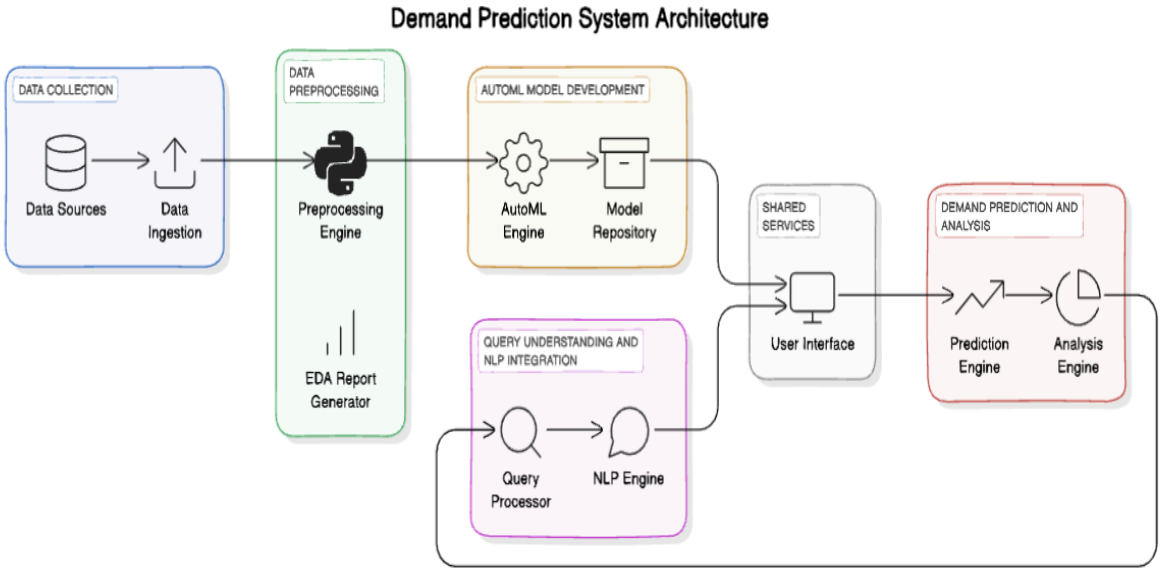

Figure 5 Architecture Diagram

The Architecture Diagram provides a comprehensive view of the Demand Prediction System's high-level design, outlining the key components and their interactions. At the core of the system is the user interface, which serves as the entry point for users to upload their datasets and interact with the system. Once the data is uploaded, it passes through the data preprocessing module, which cleans and transforms the data to ensure it's ready for analysis. The AutoML engine then analyzes the preprocessed data, automatically selecting the best machine learning models for demand prediction. These models are passed through the ensembling layer, where they are combined to improve accuracy and reliability. The evaluation module assesses the performance of the ensembled models, ensuring they meet the required precision standards. The database stores the results of the evaluation and the final predictions, which are presented back to the user through the interface.

3.1.6 ACTIVITY DIAGRAM

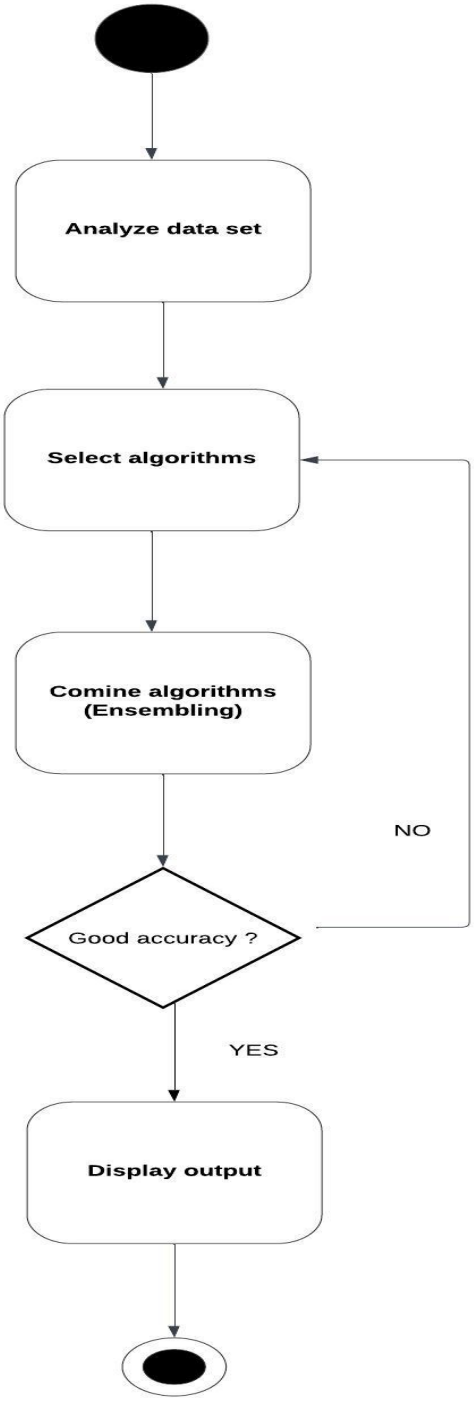

Figure 6 Activity Diagram

The Activity Diagram provides a detailed representation of the workflow within the Demand Prediction System, illustrating the sequence of activities and decisions involved in generating accurate demand predictions. The process begins with the user

uploading a dataset into the system, which triggers the preprocessing step. During preprocessing, the data is cleaned, normalized, and transformed into a format suitable for model training. The diagram emphasizes the systematic flow of tasks and decision points that contribute to a streamlined and efficient demand forecasting process, ensuring a user-friendly and automated experience.

3.1.7 COMPONENT DIAGRAM

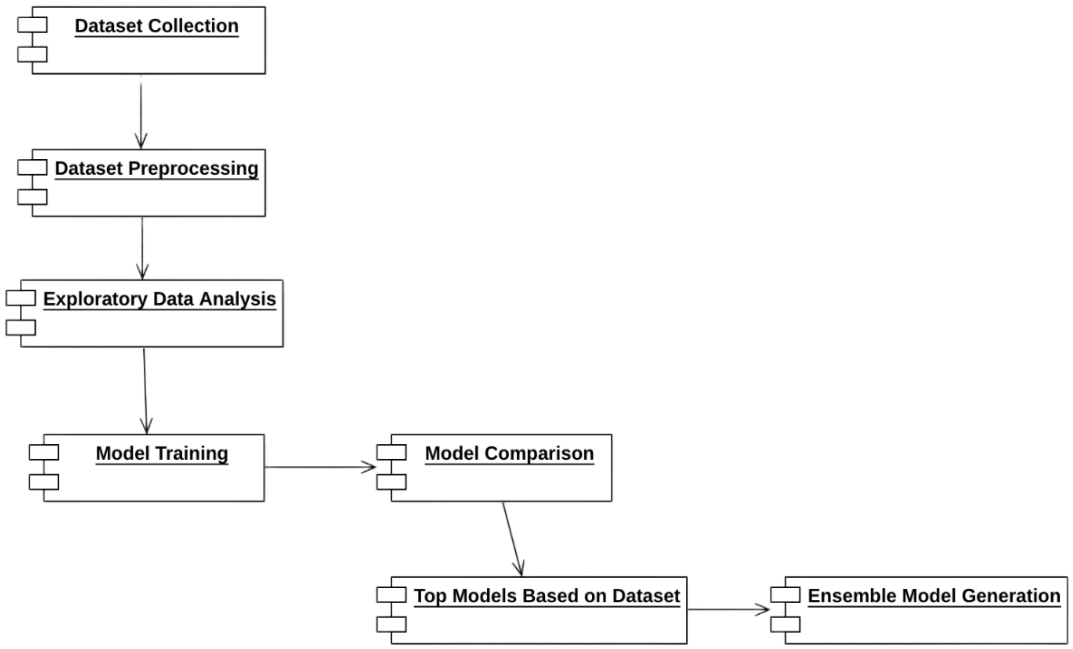

Figure 7 Component Diagram

Figure 7 illustrates the component diagram of the Demand Prediction System, showcasing key modules such as the dataset upload interface, preprocessing engine, AutoML engine, ensembling module, evaluation unit, and centralized database. The system facilitates seamless data flow, starting from dataset upload and preprocessing to model selection, ensembling, and performance evaluation. Final predictions and insights are stored in a database and displayed on a user-friendly dashboard, enabling real-time access to accurate demand forecasts and supporting data-driven decision-making.

3.1.8 COLLABORATION DIAGRAM

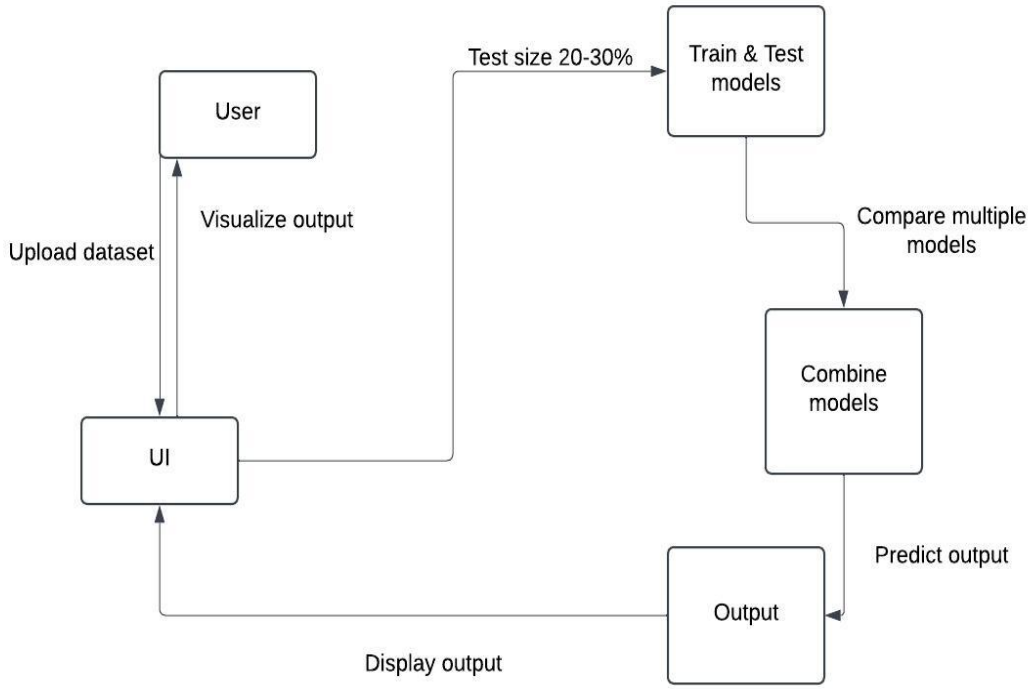

Figure 8 Collaboration Diagram

The Collaboration Diagram visually represents the dynamic interactions and relationships between key components in the Demand Prediction System. It begins with the user interface, where users upload datasets and interact with the system. The data is then passed to the data preprocessing module, which cleans and prepares it for model training. Once preprocessing is complete, the system communicates with the AutoML engine, which automatically selects the most suitable machine learning models based on the dataset's features. The final predictions are then stored in the database and displayed on the user interface for the user’s analysis.

CHAPTER 4

4. PROJECT DESCRIPTION

4.1 METHODOLOGIES:

The project leverages machine learning, particularly Automated Machine Learning (AutoML), to develop a robust, efficient, and automated demand prediction system. The integration of Natural Language Processing (NLP) further enhances user interaction by enabling intuitive, query-based communication. The system is designed to handle large-scale, diverse datasets while providing precise forecasts and actionable insights. The methodology is divided into several key phases, each of which is described below:

1. Data Collection

The first phase of the project involves collecting historical sales data from a variety of sources, such as:

- Supermarket transactions: Daily or weekly sales records from physical retail outlets.
- Online sales platforms: Data from e-commerce websites or digital marketplaces.
- Retail outlets: Data captured through Point-of-Sale (POS) systems across multiple locations.

The dataset typically includes important fields such as product identifiers (e.g., product IDs or names), timestamps (e.g., transaction dates), sales volumes, pricing details, promotional activity, and customer demographics. This comprehensive data collection ensures that the system has sufficient information to identify patterns and trends across different contexts.

Once collected, the data is structured into machine-readable formats such as **CSV or Excel**, enabling smooth integration into the system. Data from different

sources is standardized to ensure consistency, making it easier for the downstream processes to operate effectively.

2. Data Preprocessing

To ensure high-quality inputs for model training, the collected data undergoes a comprehensive preprocessing phase. This step is critical for reducing noise, addressing inconsistencies, and enhancing the accuracy of the prediction system. Key tasks include:

- Handling Missing Values: Missing data points are addressed using imputation techniques, such as filling missing numerical data with mean, median, or regression-based estimates, or applying mode imputation for categorical fields.
- Encoding Categorical Data: Non-numeric fields, such as product categories or customer regions, are converted into numeric formats using methods like label encoding or one-hot encoding.
- Date-Time Standardization: Uniform formats are enforced for date and time fields to facilitate accurate time-series analysis.
- Normalization and Scaling: Numerical fields like sales volume or pricing are normalized to ensure they have comparable scales, preventing bias during model training.

Additionally, Exploratory Data Analysis (EDA) is performed automatically using Python libraries like Pandas, Seaborn, or Matplotlib. This step generates a detailed profile report, including insights into correlations, trends, outliers, and data completeness. By automating EDA, the system ensures that the dataset is not only clean but also optimized for training. User-uploaded datasets undergo the same preprocessing pipeline, ensuring uniformity and compatibility.

3. AutoML Model Development

The heart of the system lies in its AutoML framework, which automates the selection, training, and optimization of machine learning models. The AutoML engine is designed to dynamically explore different algorithms based on the dataset's characteristics. It selects the most suitable models for ensembling from a pool that includes:

- Random Forests: Effective for capturing non-linear patterns and handling diverse datasets.
- XGBoost and LightGBM: Specialized for boosting tasks, excelling in performance for structured data.
- ARIMA: Ideal for time-series forecasting when trends and seasonality dominate.

The system evaluates these models based on predefined metrics like Mean Absolute Error (MAE) and Root Mean Square Error (RMSE). Ensembling is performed to combine the outputs of the best-performing models, leveraging their individual strengths to enhance overall prediction accuracy. The choice of models for the ensemble is data-driven, ensuring that the ensemble is tailored to the dataset's unique properties, such as dimensionality, feature distribution, and temporal patterns.

By automating hyperparameter tuning, feature selection, and model evaluation, AutoML eliminates the need for extensive manual effort while delivering an optimized, highly accurate model.

4.1.1 RESULT DISCUSSION:

The first phase of the demand prediction system is a crucial foundational step that equips users with the tools to prepare and understand their datasets effectively. This phase centers around facilitating the seamless upload of datasets, automating exploratory data analysis (EDA), and generating actionable insights into data quality,

structure, and potential variable interactions. By emphasizing data preparation and exploration, this phase ensures that datasets are ready for subsequent machine learning processes while empowering users with a deeper understanding of their data, even before predictive modeling begins.

Key Features of Phase 1 Implementation:

Dataset Upload and Integration:

Users are provided with an intuitive user interface (UI) to upload datasets in widely used formats such as CSV or Excel. Upon upload, the system automatically analyzes the dataset to determine its structure, content, and quality. This automated analysis eliminates the often tedious and error-prone process of manual data review, making it easier for users with varying technical expertise to integrate their datasets. Additionally, the system validates the dataset format and notifies users of any inconsistencies or errors, ensuring smooth integration into the workflow.

Automated Exploratory Data Analysis (EDA): After dataset upload, the system performs a comprehensive EDA, generating a detailed profile report that includes:

- Correlation Analysis: Highlights relationships between variables, pinpointing predictors and potential target variables. This insight helps users understand data interdependencies crucial for model development.
- Missing Values Insights: Identifies the extent of missing data for each feature, offering recommendations for handling missing values, such as imputation or exclusion strategies. This step ensures that data quality is maintained for reliable predictions.
- Feature Distributions and Statistics: Provides key statistical summaries such as mean, median, variance, standard deviation, and skewness, revealing underlying patterns and outliers within the data.
- Interactive Visualizations: Includes a range of graphs like heatmaps for correlation matrices, bar charts for categorical data, and histograms for

numeric data. These visuals make it easier for users to interpret complex data structures intuitively and effectively.

Model Recommendation and Comparison:

The system evaluates the uploaded dataset's characteristics and recommends a selection of machine learning models tailored to the data's needs. Preliminary training results generated through an AutoML framework are used to rank these models based on performance metrics such as Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Precision, and Recall. The system also presents a side-by-side comparison of models, helping users select the best-fit model for their predictive goals. This feature adds transparency and guidance to the decision-making process, particularly for users who may not have extensive expertise in machine learning.

By integrating dataset upload, automated EDA, and model recommendations into a cohesive workflow, Phase 1 of the demand prediction system ensures a user-friendly, efficient, and effective data preparation stage. This phase not only simplifies the transition into machine learning but also enhances the accessibility and usability of the system for diverse user groups, enabling them to make informed decisions and achieve better predictive outcomes.

CHAPTER 5

5.1 CONCLUSION AND WORKSPACE

Phase 1 of the Demand Prediction System establishes a comprehensive framework for exploratory data analysis (EDA), streamlining the early stages of data processing, which are often labor-intensive and error-prone. This phase is designed to enable users to effortlessly upload datasets, perform automated analysis, and generate detailed reports, providing a foundational understanding of data quality, structure, and patterns before delving into predictive modeling.

The system is powered by Python Flask, offering a lightweight yet scalable backend for file uploads, automated processing, and interactive report generation. Users can upload datasets in formats such as CSV through a web-based interface, where the system leverages the pandas-profiling library to generate rich HTML-based reports. These reports include statistical summaries, visualizations, and insights on data quality, such as correlations, missing values, feature distributions, and potential outliers. This automated approach simplifies data preparation while maintaining accuracy and accessibility for both technical and non-technical users.

The key feature of this phase is the generation of comprehensive profile reports that are visually rich and intuitive. These reports incorporate heatmaps, histograms, and scatter plots to present insights effectively, ensuring users gain actionable knowledge of their datasets. The system also validates uploaded files to ensure compatibility, stores them securely, and renders the reports seamlessly through a browser-based interface, eliminating the need for additional software installations.

The implementation of Phase 1 has been rigorously tested with datasets of varying sizes and complexities, consistently demonstrating high efficiency, accuracy, and user-friendliness. Reports for datasets with up to 100,000 rows were generated in under 15 seconds, providing reliable insights that matched manually conducted EDA results. Test users, including those without a background in data science, were able

to navigate the system and interpret the generated reports effectively, validating its accessibility and ease of use.

In conclusion, Phase 1 lays a solid foundation for advanced machine learning workflows by automating the critical preparatory steps of data analysis. Its combination of real-time EDA, user-friendly design, and robust backend ensures a seamless experience for users, empowering them with the tools needed to explore their data and make informed decisions for predictive modeling.

5.2 For Phase 2

Building upon the accomplishments of Phase 1, Phase 2 will focus on extending the system's capabilities to include Automated Machine Learning (AutoML) and automated ensembling for predictive modeling. This phase aims to transition the system from an EDA tool to a full-fledged machine learning platform capable of generating accurate and robust models tailored to specific datasets.

Planned Features and Enhancements in Phase 2

1. **AutoML Integration:**

   The system will incorporate an AutoML framework to automate the process of selecting, training, and tuning machine learning models. This enhancement will allow users to:

   - Automatically evaluate a range of models, such as Random Forests, XGBoost, LightGBM, and Gradient Boosting, based on dataset characteristics.
   - Measure model performance using metrics like Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Precision, and Recall.
   - Select the most effective models for prediction without requiring manual
   - intervention or prior knowledge of machine learning techniques.

2. **Dynamic Model Selection and Ensembling:**

   The AutoML pipeline will dynamically select the best-performing models for the given dataset and combine them using advanced ensembling techniques, such as:

   - Bagging: Aggregating predictions to reduce variance and improve stability.
   - Boosting: Incrementally training models to correct errors from previous iterations, improving overall accuracy.
   - Stacking: Using meta-models to synthesize outputs from multiple base models for superior performance.

The ensembling process will be tailored to the dataset's specific properties, ensuring optimal results across a wide variety of use cases.

3. **Enhanced Workflow Integration:**

   Users will seamlessly transition from Phase 1's EDA functionality to Phase 2's machine learning workflow. The system will allow users to:

   - View the generated profile report.
   - Select specific features or filters based on EDA insights.
   - Trigger the AutoML pipeline for training and model selection directly from the interface.

4. **Advanced Visualization of Model Results:**

   The system will provide detailed visualizations of model performance, including:

   - Feature Importance Rankings: Highlighting which features contribute most to the predictions.

- Prediction vs. Actual Graphs: Comparing the model's predictions against actual outcomes to assess accuracy.
- Error Distribution Plots: Displaying areas where the model performs well and where it may need improvement.

5. **Scalable Backend for Model Training:**

   The Flask application will be extended to support large-scale training tasks while maintaining a responsive user interface. The system will manage computational resources effectively to handle datasets of varying sizes and complexities.

REFERENCES

1. S. Mhatre, S. Patil, N. Mishra, V. Mungelwar and H. Patil, "AutoML Based Tourism Prediction and Maximising Revenue," 2024 2nd International Conference on Sustainable Computing and Smart Systems (ICSCSS), Coimbatore, India, 2024, pp. 1193-1202,doi: 10.1109/ICSCSS60660.2024.10625466.

2. T. Nagarajah and G. Poravi, "A Review on Automated Machine Learning (AutoML) Systems," 2019 IEEE 5th International Conference for Convergence in Technology (I2CT), Bombay, India, 2019, pp. 1-6, doi: 10.1109/I2CT45611.2019.9033810.

3. P. Kumar, U. L. Maneesh and G. Mano Sanjay (2024), "Optimizing Loan Approval Decisions: Harnessing Ensemble Learning for Credit Scoring," 2024 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI),Chennai,India,2024,pp.1-4,doi: 10.1109/ACCAI61061.2024.10602097.

4. D.Mallikarachchi, D. Rathnayake, D. Abegunawardana, S. Van-Hoff, D. Kasthurirathna and A. Gamage, "Automated Machine Learning for Prediction of Type 2 Diabetes and Its Major Complications: A Comparative Study," 2023 5th International Conference on Advancements in Computing (ICAC), Colombo, Sri Lanka, 2023, pp. 466-471, doi: 10.1109/ICAC60630.2023.10417572.

5. A. Ghareeb, H. Al-bayaty, Q. Haseeb and M. Zeinalabideen, "Ensemble learning models for short-term electricity demand forecasting," 2020 International Conference on Data Analytics for Business and Industry: Way Towards a

Sustainable Economy (ICDABI), Sakheer, Bahrain, 2020, pp. 1-5, doi: 10.1109/2020.9325623.

6. Y. Jin, X. Ye, Q. Ye, T. Wang, J. Cheng and X. Yan, "Demand Forecasting of Online Car-Hailing With Stacking Ensemble Learning Approach and Large-Scale Datasets," in IEEE Access, vol. 8, pp. 199513-199522, 2020, doi: 10.1109/ACCESS.2020.3034355.

7. V. E. Kovalevsky and N. A. Zhukova, "Building a Model for Time Series Forecasting using AutoML Methods," 2024 XXVII International Conference on Soft Computing and Measurements (SCM), Saint Petersburg, Russian Federation, 2024, pp. 308-311, doi: 10.1109/SCM62608.2024.10554133.

8. H. -A. -D. Cap, T. -H. Do, D. S. Lakew and S. Cho, "Building a Time-Series Forecast Model with Automated Machine Learning for Heart Rate Forecasting Problem," 2022 13th International Conference on Information and Communication Technology Convergence (ICTC), Jeju Island, Korea, Republic of, 2022, pp. 1097-1100, doi: 10.1109/ICTC55196.2022.9952797.

9. I. Met, A. Erkoç and S. E. Seker, "Performance, Efficiency, and Target Setting for Bank Branches: Time Series With Automated Machine Learning," in IEEE Access, vol. 11, pp. 1000-1010, 2023,doi: 10.1109/ACCESS.2022.3233529.

10. G. Stamatescu, R. Plamanescu and M. Albu, "Leveraging Anomaly Detection and AutoML for Modelling Residential Measurement Power Traces," 2023 IEEE 13th International Workshop on Applied Measurements for Power Systems (AMPS), Bern, Switzerland, 2023, pp. 1-5, doi: 10.1109/AMPS59207.2023.10297201.

11. H. Iftikhar, S. Mancha Gonzales, J. Zywiołek and J. L. López-Gonzales, "Electricity Demand Forecasting Using a Novel Time Series Ensemble Technique," in IEEE Access, vol. 12, pp. 88963-88975, 2024,doi: 10.1109/ACCESS.2024.3419551.

12. Q. Lyu and R. Zhang, "Research on Demand Forecasting Method of Shared Bicycle Based on Ensemble Learning," 2023 International Conference on Image Processing, Computer Vision and Machine Learning (ICICML), Chengdu,China, 2023,pp. 861-865,doi: 10.1109/ICICML60161.2023.10424944.

13. Y. Zhang, H. Zhu, Y. Wang and T. Li, "Demand Forecasting: From Machine Learning to Ensemble Learning," 2022 IEEE Conference on Telecommunications,

Optics and Computer Science (TOCS), Dalian, China, 2022, pp. 461-466, doi: 10.1109/TOCS56154.2022.10015992.

14. D. Hulak and G. Taylor, "Investigating an Ensemble of ARIMA Models for Accurate Short-Term Electricity Demand Forecasting," 2023 58th International Universities Power Engineering Conference (UPEC), Dublin, Ireland, 2023, pp. 1-6, doi: 10.1109/UPEC57427.2023.10294946.

15. P. Naik, M. Dalponte and L. Bruzzone, "Automated Machine Learning Driven Stacked Ensemble Modeling for Forest Aboveground Biomass Prediction Using Multitemporal Sentinel-2 Data," in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 16, pp. 3442-3454, 2023, doi: 10.1109/JSTARS.2022.3232583.

16. A. Garg and A. Chaudhary, "Analysis of IPL Auction Dataset Using Explainable Machine Learning with Lime and H2O AutoML," 2023 4th International Conference on Intelligent Engineering and Management (ICIEM), London, United Kingdom, 2023, pp. 1-4, doi: 10.1109/ICIEM59.2023.10167124.

17. S. P. Menon, K. Vaishaali, N. G. Sathvik, S. P. A. Gollapalli, S. N. Sadhwani and V. A. Punagin, "Brain Tumor Diagnosis and Classification based on AutoML and Traditional Analysis," 2022 IEEE Global Conference on Computing, Power and Communication Technologies (GlobConPT), New Delhi, India, 2022 pp. 17,doi: 10.1109/GlobConPT57482.2022.993814.

18. P. Kumar, K. N. Manisha and M. Nivetha (2024), "Market Basket Analysis for Retail Sales Optimization (2024)," 2024 Second International Conference on Emerging Trends in Information Technology and Engineering (ICETITE), Vellore, India, pp. 1-7, doi: 10.1109/ic-ETITE58242.2024.10493283.

APPENDIX

APPENDIX 1

TITLE: Demand Prediction Using AutoML Based Ensemble Algorithm

AUTHORS: Dr. P. Kumar, Dr. S Senthil Pandi, Mohamed Hussain S, Nathaniel Abishek A

PUBLICATION STATUS: Submitted

APPENDIX 2

```
import streamlit as st
import plotly.express as px
import pandas as pd
import requests
import os
from pycaret.regression import pull, load_model
from sklearn.preprocessing import StandardScaler
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import LabelEncoder

# Streamlit UI Setup
st.set_page_config(page_title="AutoML", layout="wide")

# Load dataset if it exists
if os.path.exists('./dataset.csv'):
    df = pd.read_csv('dataset.csv')

with st.sidebar:
    st.image("https://www.onepointltd.com/wp-
content/uploads/2020/03/inno2.png")
    st.title("AutoML")
    choice = st.radio("Navigation", ["Upload", "Data Preprocessing",
"Profiling", "Modelling", "Download"])
    st.info("This project application helps you build and explore your
data.")

# Upload dataset
if choice == "Upload":
    st.title("Upload Your Dataset")
    uploaded_file = st.file_uploader("Upload Your Dataset (CSV
```

```
format)", type=["csv"])
    if uploaded_file:
        try:
            # Display the uploaded file in Streamlit
            df = pd.read_csv(uploaded_file)
            st.dataframe(df)

            # Send file to Flask backend for processing
            files = {"file": (uploaded_file.name, uploaded_file,
"text/csv")}
            response = requests.post("http://localhost:5000/upload",
files=files)

            # Check the Flask response
            if response.status_code == 200:
                st.success(response.json().get("message", "File uploaded
successfully!"))
                df.to_csv('dataset.csv', index=False)  # Save locally for
profiling
            else:
                st.error(f"Error uploading file: {response.text}")
        except Exception as e:
            st.error(f"An error occurred while processing the file: {e}")

# Data Preprocessing
if choice == "Data Preprocessing":
    st.title("Data Preprocessing")
    if 'df' in locals():
        st.write("Here are the first few rows of the dataset:")
        st.dataframe(df.head())
```

```
        # Handle missing values
        st.subheader("Handle Missing Values")
        missing_value_strategy = st.selectbox(
            "Select a strategy for missing values", ["Drop Rows",
"Impute with Mean/Median"]
        )
        if missing_value_strategy == "Drop Rows":
            df = df.dropna()
            st.success("Rows with missing values have been dropped.")
        elif missing_value_strategy == "Impute with Mean/Median":
            imputer = SimpleImputer(strategy='mean')  # You can also
use 'median'
            df[df.columns] = imputer.fit_transform(df)
            st.success("Missing values have been imputed with the
mean/median.")

        # Categorical feature encoding
        st.subheader("Encode Categorical Variables")
        encode_option = st.selectbox("Select encoding method",
["None", "Label Encoding", "One-Hot Encoding"])
        if encode_option == "Label Encoding":
            le = LabelEncoder()
            for col in df.select_dtypes(include=['object']).columns:
                df[col] = le.fit_transform(df[col])
            st.success("Categorical variables have been label encoded.")
        elif encode_option == "One-Hot Encoding":
            df = pd.get_dummies(df)
            st.success("Categorical variables have been one-hot
encoded.")
```

```
        # Feature scaling
        st.subheader("Feature Scaling")
        scale_option = st.selectbox("Select scaling method", ["None",
"Standard Scaling", "Min-Max Scaling"])
        if scale_option == "Standard Scaling":
            scaler = StandardScaler()
            df[df.select_dtypes(include=['float64', 'int64']).columns] =
scaler.fit_transform(df.select_dtypes(include=['float64', 'int64']))
            st.success("Features have been standardized (z-score
normalization).")
        elif scale_option == "Min-Max Scaling":
            df[df.select_dtypes(include=['float64', 'int64']).columns] =
(df.select_dtypes(include=['float64', 'int64']) - df.min()) / (df.max() -
df.min())
            st.success("Features have been scaled using Min-Max
scaling.")

        # Save processed data
        df.to_csv('dataset.csv', index=False)
        st.write("Processed dataset:")
        st.dataframe(df.head())

    else:
        st.warning("Please upload a dataset first.")

# Profiling the dataset
if choice == "Profiling":
    st.title("Exploratory Data Analysis")
    if 'df' in locals():
```

```
        from ydata_profiling import ProfileReport
        from streamlit_pandas_profiling import st_profile_report

        profile_df = ProfileReport(df, explorative=True)
        st_profile_report(profile_df)
    else:
        st.warning("Please upload a dataset first.")

# Modelling
if choice == "Modelling":
    st.title("Model Training")
    if 'df' in locals():
        target_column = st.selectbox("Choose the Target Column",
df.columns)
        if st.button("Train Model"):
            try:
                # Prepare JSON data for Flask model training
                data_payload = {
                    "data": df.to_dict(orient="records"),
                    "target": target_column
                }
                response = requests.post("http://localhost:5000/model",
json=data_payload)

                # Display response
                if response.status_code == 200:
                    model_details = response.json()
                    st.success(model_details.get("message", "Model trained
successfully!"))
                    st.json(model_details.get("model_details", {}))
```

```
            else:
                st.error(f"Error during model training: {response.text}")
        except Exception as e:
            st.error(f"An error occurred during model training: {e}")
    else:
        st.warning("Please upload and profile your dataset first.")

# Download the trained model
if choice == "Download":
    st.title("Download Trained Model")
    try:
        response =
requests.get("http://localhost:5000/download_model")
        if response.status_code == 200:
            model_path = response.json().get("model_path",
"best_model.pkl")
            with open(model_path, "rb") as file:
                st.download_button("Download Trained Model", file,
file_name="best_model.pkl")
        else:
            st.error(response.json().get("error", "Model file not found."))
    except Exception as e:
        st.error(f"An error occurred: {e}")
```