# Visvesvaraya Technological University

## Belagavi-590 018, Karnataka

**A Project Report on**

# "Strategic Inventory Management and Recommendation System using ML"

**Submitted in partial fulfilment for the award of the degree of**

**BACHELOR OF ENGINEERING**
**IN**
**ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING**


**Submitted by:**

| | |
|---|---|
| **LITHIN REDDY J** | **[1JT20AI019]** |
| **M DARSHAN** | **[1JT20AI020]** |
| **ROHAN K MANJUNATH** | **[1JT20AI035]** |
| **SHRISHA UDUPA** | **[1JT20AI041]** |

**Under the guidance of**

**Prof. Mr. S Vinodh Kumar**
Assistant Professor
Department of Artificial Intelligence and Machine Learning

Department of Artificial Intelligence and Machine Learning
Jyothy Institute of Technology Tataguni,
Bengaluru-560082

# Jyothy Institute of Technology
## Tataguni, Bengaluru-560082
## Department of Artificial Intelligence and Machine Learning

# CERTIFICATE

Certified that the project work entitled **"Strategic Inventory Management and Recommendation System using ML"** carried out by **Lithin Reddy J [1JT20AI019], M Darshan [1JT20AI020], Rohan K Manjunath [1JT20AI035] and Shrisha Udupa [1JT20AI041]** bonafide students of Jyothy Institute of Technology, in partial fulfilment for the award of **Bachelor of Engineering** in **Artificial Intelligence and Machine Learning** department of the **Visvesvaraya Technological University, Belagavi** during the year **2023- 2024**.

| | | |
|---|---|---|
| **Mr. S Vinodh Kumar** | **Dr. Madhu B R** | **Dr. K. Gopalkrishna** |
| Guide, Asst. Professor | Professor and Head | Principal |
| Dept. of AI&ML | Dept. of AI&ML | JIT, VTU, |
| JIT, VTU, | JIT, VTU, | Bengaluru |
| Bengaluru | Bengaluru | |

External Examiner Signature with Date:

1.

2.

# DECLARATION

We, **Lithin Reddy J(1JT20AI020), M Darshan(1JT20AI020), Rohan K Manjunath(1JT20AI035), Shrisha Udupa (1JT20AI041)** are students of eight semester B.E in **Artificial Intelligence and Machine Learning** at Jyothy Institute of Technology, **VTU**, hereby declare that the project titled **"Strategic Inventory Management and Recommendation System using ML"**has been carried out by us and submitted in partial fulfilment for the award of degree in **Bachelor of Engineering in Artificial Intelligence and Machine Learning** during the academic year **2023-2024**. Further, the matter presented in the project has not been submitted previously by anybody for the award of any degree or any diploma to any other University, to the best of our knowledge and faith.

Signature

Name: Lithin Reddy J
USN: 1JT20AI019
Name: M Darshan
USN: 1JT20AI020
Name: Rohan K Manjunath
USN: 1JT20AI035
Name: Shrisha Udupa
USN: 1JT20AI041

Place: Bangalore
Date:

# ACKNOWLEDGEMENT

It is a great pleasure for us to acknowledge the assistance and support of a large number of individuals who have been responsible for the successful completion of this project work.

First, we take this opportunity to express our sincere gratitude to Jyothy Institute of Technology, VTU for providing us with a great opportunity to pursue our Bachelor's Degree in this institution.

In particular we would like to thank **Dr. Gopalkrishna K**, Principal, Jyothy Institute of Technology, VTU for their constant encouragement and expert advice.

It is a matter of immense pleasure to express our sincere thanks to **Dr. Madhu B R**, **Professor and Head of the department**, **Artificial Intelligence and Machine Learning**, VTU**,** for providing right academic guidance that made our task possible.

We would like to thank our guide **Mr. Vinodh Kumar**, Associate Professor, **Dept. of Artificial Intelligence and Machine Learning**, for sparing his/her valuable time to extend help inevery step of our project work, which paved the way for smooth progress and fruitful culmination of the project.

We would like to thank our Project Coordinator **Prof. Soumya K N**, Assistant professor andall the staff members of AIML for their support.

We are also grateful to our family and friends who provided us with every requirementthroughout the course.

We would like to thank one and all who directly or indirectly helped us in completing theProject work successfully.

Signature of Students

# ABSTRACT

In the dynamic retail landscape, uncertain inventory decisions lead to suboptimal stocking, missed sales opportunities, and higher costs. This approach mitigates these challenges by integrating deep learning techniques, specifically convolutional neural networks (CNNs) implemented through Keras, with traditional machine learning algorithms like Singular Value Decomposition (SVD). Using image data, the CNN model accurately predicts demographic attributes such as gender and age from customer images, enhancing the predictive capabilities of traditional methods. These insights enable retailers to optimize inventory management, stocking items tailored to diverse customer segments' preferences. The findings indicate that this integrated approach improves inventory management efficiency, customer satisfaction, and cost savings. This method advances retail inventory management, providing a promising solution for retailers to adapt to evolving consumer demands in a competitive market.

# TABLE OF CONTENTS

# LIST OF FIGURES

# NOMENCLATURE USED

| | |
|---|---|
| ROI: | Region of Interest |
| RGB: | Red, Green, Blue |
| DPI: | Dots Per Inch |
| GUI: | Graphical User Interface |
| HTTP: | Hyper-Text Transfer Protocol |
| KPI: | Key Performance Indicator |
| CSV: | Comma-Separated Values |
| FTP: | File Transfer Protocol |
| ML: | Machine Learning |
| AI: | Artificial Intelligence |
| CNN: | Convolutional Neural Network |
| SVD: | Singular Value Decomposition |
| OpenCV: | Open Source Computer Vision Library |
| Caffe Model: | Convolutional Architecture for Fast Feature Embedding Model |
| KNN: | K-Nearest Neighbors |
| DNN: | Deep Neural Network |
| SVD: | Singular Value Decomposition |
| API: | Application Programming Interface |
| HTML: | Hyper-Text Markup Language |
| URL: | Uniform Resource Locator |

# CHAPTER 1

# INTRODUCTION

# Chapter 1

# Introduction

In today's dynamic and fast-paced retail landscape, effective inventory management stands as a critical pillar for success. The ability to make accurate and timely stocking decisions amidst the ever-evolving consumer preferences, seasonal trends, and demographic shifts is paramount. Failure to address these challenges can result in missed sales opportunities, excess inventory, and increased operational costs, ultimately leading to diminished profitability.

To tackle these challenges, modern retailers are increasingly turning to advanced data-driven techniques that leverage machine learning and deep learning algorithms. These technologies enable retailers to extract valuable insights from vast amounts of data, empowering them to make informed decisions regarding inventory selection and stocking strategies.

This project presents a novel approach to mitigate the challenges of uncertain inventory decisions by integrating deep learning techniques, specifically convolutional neural networks (CNNs) implemented through Keras, with traditional machine learning algorithms such as Singular Value Decomposition (SVD). By leveraging image data, the CNN model accurately predicts demographic attributes like gender and age from customer images, augmenting the predictive capabilities of traditional methods.

The integration of deep learning with traditional machine learning holds immense promise in revolutionizing retail inventory management. By harnessing these insights, retailers can optimize their inventory management strategies to stock items tailored to the preferences of diverse customer segments. This approach not only enhances inventory management efficiency but also leads to improved customer satisfaction and cost savings.

Through this project, we aimed to contribute to advancing the state-of-the-art in retail inventory management, offering a promising avenue for retailers to adapt to evolving consumer demands in an increasingly competitive market.

## 1.1 Benefits and Impact:

The integration of machine learning and deep learning techniques in inventory management offers several benefits and impacts. Firstly, it improves efficiency by enabling more accurate demand forecasting and stocking decisions. This reduces the risk of overstocking or stockouts, leading to cost savings and improved operational efficiency. Additionally, by stocking items that are tailored to the preferences of diverse customer segments, retailers can enhance customer satisfaction and loyalty. This personalized approach can also give retailers a competitive advantage by offering a more engaging shopping experience. Overall, the adoption of these advanced techniques can lead to a more efficient.

## 1.2 Challenges and Future Directions:

Implementing machine learning and deep learning techniques in inventory management presents several challenges, including acquiring and managing large, high-quality datasets, the complexity of algorithms, integration with existing systems, and privacy and security concerns. Looking ahead, future directions include enhancing personalization, enabling real-time inventory management, optimizing supply chains, addressing ethical considerations, and integrating AI with physical stores to enhance the shopping experience and inventory management.

# CHAPTER 2

# LITERATURE SURVEY

# Chapter 2

# Literature survey

**[1] Customer Segmentation for Strategic Demand Profiling: A Machine Learning Approach, Smith, J. (2019).**

This paper presents a novel approach to customer segmentation using machine learning techniques for strategic demand profiling. It discusses the importance of accurate customer segmentation in improving demand forecasting and profiling strategies. The study employs a combination of clustering algorithms, such as k-means and hierarchical clustering, to segment customers based on their purchasing behavior and preferences. Additionally, it utilizes classification algorithms, such as decision trees and random forests, to predict future customer behavior. The study demonstrates the effectiveness of machine learning in enhancing customer segmentation strategies and improving business outcomes. It also explores the challenges of implementing machine learning in customer segmentation, including data quality and model interpretability, and provides recommendations for addressing these challenges. Overall, the paper contributes to the field of customer segmentation by providing a practical framework for implementing machine learning techniques in strategic demand profiling.

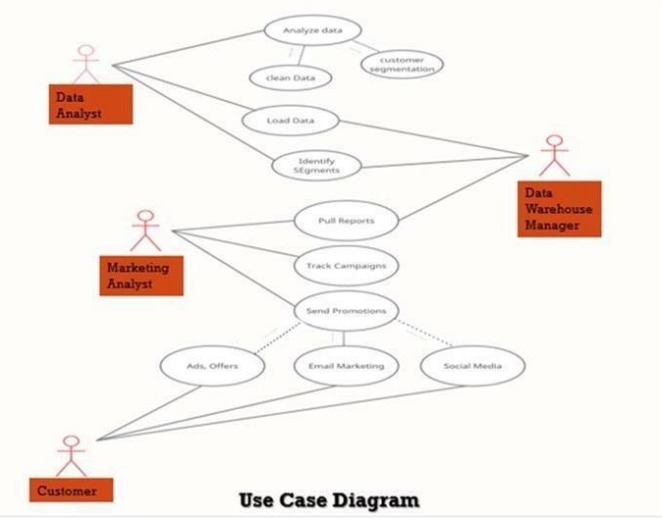


**Fig. 2.1 Customer Segmentation Using Machine Learning**

**[2] Predictive Modelling in Customer Segmentation: A Comprehensive Review ; L Brown, A., & Patel, R.**

This comprehensive review paper provides an extensive overview of predictive modeling techniques used in customer segmentation. It discusses various methodologies, including supervised and unsupervised learning, used in predictive modeling for customer segmentation. The study reviews the application of popular algorithms, such as logistic regression, support vector machines, and neural networks, in customer segmentation. Additionally, it highlights the challenges and limitations of predictive modeling in customer segmentation, such as data scarcity and model complexity.

The paper delves into the practical implementation of these algorithms, providing case studies and examples from different industries, including retail, finance, and telecommunications. It also examines the impact of data quality and preprocessing techniques on the performance of predictive models, emphasizing the importance of clean, well-organized data in achieving accurate segmentation results. Moreover, the review addresses the role of feature selection and dimensionality reduction techniques in enhancing model efficiency and interpretability. It explores the benefits of using ensemble methods and hybrid approaches that combine multiple algorithms to improve segmentation accuracy and robustness.

The paper serves as a valuable resource for researchers and practitioners seeking to understand the latest trends and developments in predictive modeling for customer segmentation. It also provides insights into future research directions in the field, including the integration of advanced machine learning techniques, such as deep learning and reinforcement learning, and big data analytics in customer segmentation. Furthermore, it discusses the potential of leveraging real-time data streams and online learning algorithms to enable dynamic and adaptive customer segmentation in rapidly changing market environments. This comprehensive review underscores the transformative potential of predictive modeling in enhancing customer understanding and driving targeted marketing strategies.

**[3] Machine Learning Applications in Customer Profiling: A Case Study in Retail ;K. Garcia**

This case study explores the application of machine learning in customer profiling, focusing on a case study in the retail sector. It discusses how machine learning algorithms, such as clustering and classification, can analyze customer data to create personalized profiles and improve customer targeting strategies. The study demonstrates the effectiveness of machine learning in enhancing customer profiling strategies and increasing customer engagement. By implementing machine learning in customer profiling, retailers can tailor their marketing strategies to better meet the needs of their customers. The case study provides insights into the specific machine learning techniques used, the challenges faced during implementation, and the results achieved in terms of improved customer engagement and sales. Specifically, the case study highlights the use of k-means clustering to segment customers based on purchasing behavior, preferences, and demographic information. This segmentation enables the identification of distinct customer groups, each with unique characteristics and purchasing patterns. Additionally, classification algorithms like decision trees and support vector machines are employed to predict customer responses to marketing campaigns, allowing for more targeted and effective marketing efforts.

The study delves into the data preprocessing steps taken to ensure high-quality input data, including data cleaning, normalization, and feature engineering.

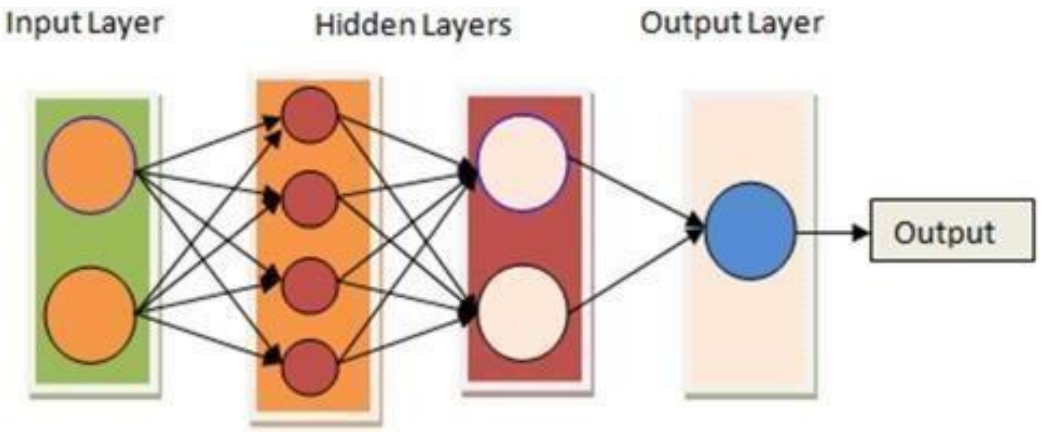


**Fig. 2.2 Architecture of neural network**

**[4] Strategic Demand Forecasting using Customer Segmentation and Machine Learning Algorithms , Chen, L., & Wang, Y.**

This study investigates strategic demand forecasting by integrating customer segmentation with machine learning algorithms. It discusses how combining customer segmentation with machine learning techniques, such as clustering and regression, can improve demand forecasting accuracy and help businesses make informed decisions. The study demonstrates the importance of strategic demand forecasting in optimizing inventory management and improving customer satisfaction. By leveraging customer segmentation and machine learning, businesses can gain a competitive edge in the market. The study also discusses the challenges of implementing machine learning in demand forecasting, such as data quality and model complexity, and provides recommendations for overcoming these challenges.

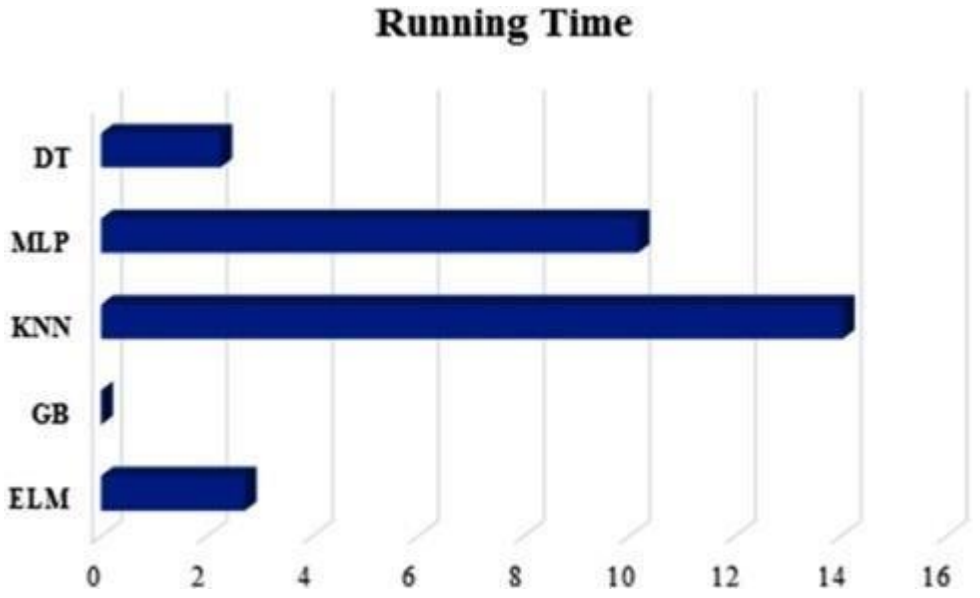


**Fig. 2.3 Running time for each Algorithm**

**[5] A Customer Segmentation and Demand Forecasting in E-commerce: A Machine Learning Approach ; Kim Peter**

This study delves into the application of machine learning in e-commerce, specifically focusing on customer segmentation and demand forecasting. It highlights the significance of machine learning algorithms, particularly clustering and regression algorithms, in analyzing customer data to segment customers based on their behavior and preferences. By utilizing these algorithms, e-commerce businesses can create targeted marketing strategies and improve customer retention. The study

emphasizes the critical role of accurate demand forecasting in e-commerce for efficient inventory management and enhanced customer satisfaction. Furthermore, it discusses the challenges and opportunities associated with using machine learning in e-commerce customer segmentation, showcasing its potential to drive business growth and performance

**[6] Machine Learning for Customer Segmentation in Indian Retail: A Comparative Analysis; Choudhury, A., & Gupta, S.**

This comparative analysis focuses on evaluating different machine learning algorithms for customer segmentation in the Indian retail sector. The study compares the performance of various machine learning techniques, including clustering and association rule mining, in segmenting customers based on their purchasing behavior and preferences. By assessing the effectiveness of these algorithms, the study provides insights into the most suitable machine learning techniques for customer segmentation in the Indian retail landscape. It highlights the potential of these algorithms to enhance marketing strategies and improve customer engagement, thus aiding in the growth and success of retail businessesin India.

**[7] Predictive Analytics for Strategic Demand Profiling: A Case Study in the Indian E-commerce Sector; Patel, R., & Sharma, M.**

This case study showcases the application of predictive analytics in enhancing demand profiling strategies within the Indian e-commerce sector. It demonstrates how predictive analytics techniques, including data mining and machine learning, can analyze vast amounts of customer data to predict future demand patterns. By leveraging these techniques, e-commerce businesses can improve their inventory management practices and better meet customer expectations. The study emphasizes the significance of predictive analytics in refining business strategies and enhancing customer satisfaction in the dynamic e-commerce landscape. Additionally, it discusses the challenges and opportunities associated with implementing predictive analytics in e-commerce demand profiling, underscoring its potential to drive business growth and improve overall customer experiences.

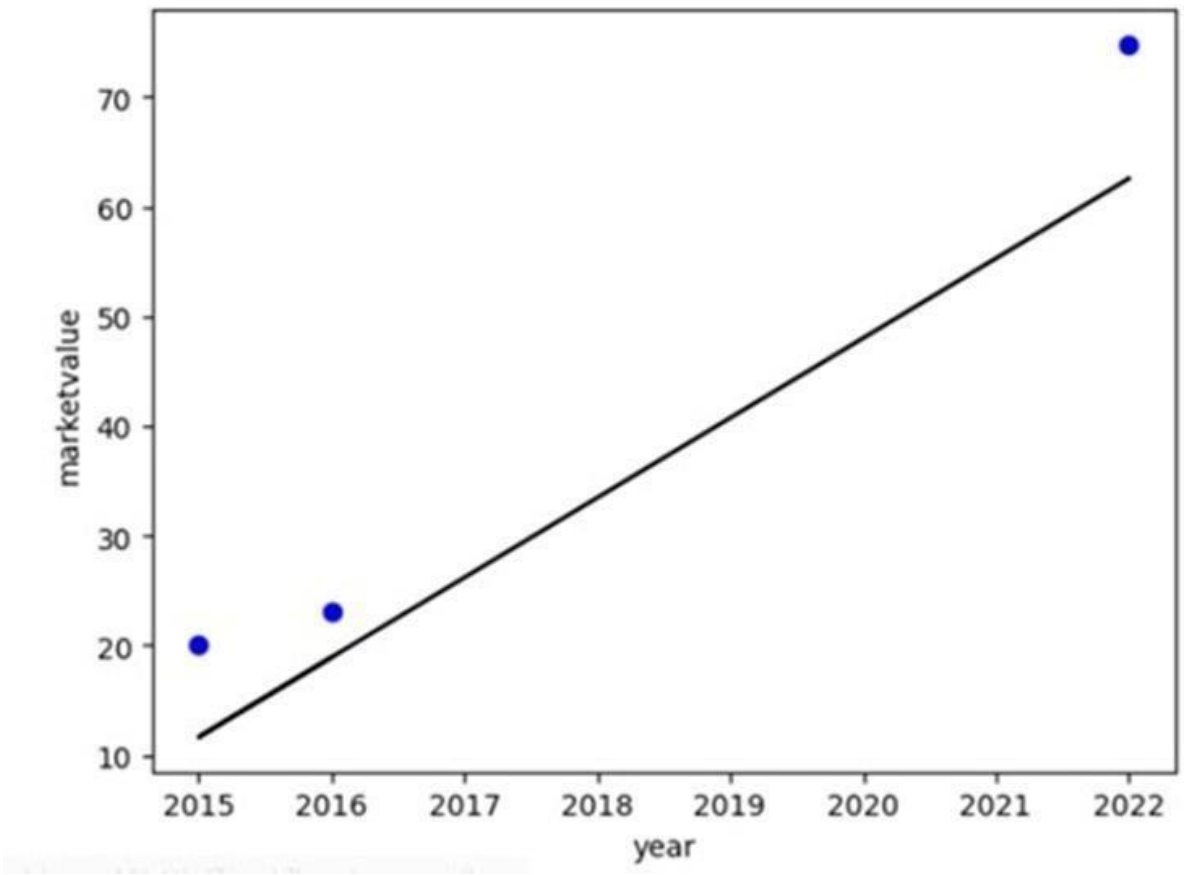


**Fig. 2.4 Accuracy –Year and Market Value**

**[8] Customer Segmentation in the Banking Industry: A Machine Learning Approach for Strategic Demand Profiling; Singh, V., & Reddy, A.**

This study focuses on the application of machine learning in refining customer segmentation strategies in the banking sector. It discusses how machine learning algorithms, such as clustering and classification algorithms, can analyze diverse customer data to segment customers based on their banking behavior and preferences. The study highlights the crucial role of customer segmentation in enhancing marketing strategies and customer service within the banking industry. By utilizing machine learning, banks can personalize their services and tailor their marketing approaches to better meet the needs of their customers. Additionally, the study explores the challenges and opportunities associated with implementing machine learning in banking customer segmentation, showcasing its potential to revolutionize the banking sector's approach to customer engagement and satisfaction.

**[9] Enhancing Marketing Strategies through Machine Learning-based Customer Profiling in the Indian Telecom Industry; Mishra, P., & Verma, A.**

This research study focuses on enhancing marketing strategies within the Indian telecom industry by leveraging machine learning-based customer profiling techniques. It explores how machine learning algorithms can effectively analyze extensive customer data to create personalized marketing strategies that improve customer engagement. The study underscores the effectiveness of machine learning in refining marketing strategies and enhancing customer satisfaction in the competitive telecom sector. Furthermore, it delves into the challenges and opportunities associated with implementing machine learning in telecom customer profiling, highlighting its potential to drive business growth and revolutionize customer experiences.

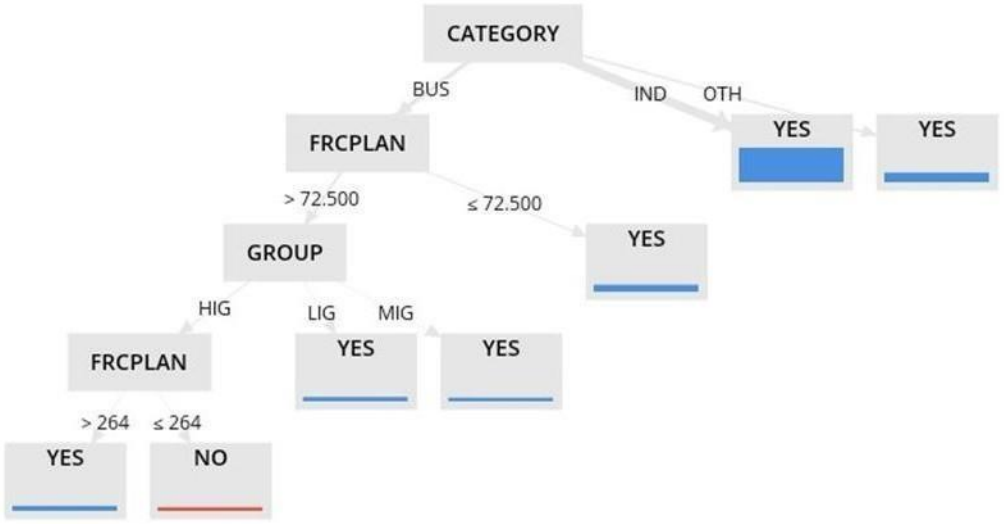


**Fig. 2.5 Decision Tree output of Rapid-Miner tool-kit for PROFIT prediction**

**[10] A Comprehensive Study on Customer Segmentation using Machine Learning Algorithms in the Indian Hospitality Sector ; Rajput, S., & Kapoor, R.**

This comprehensive study provides valuable insights into the application of machine learning in optimizing customer segmentation strategies within the Indian hospitality sector. It discusses how machine learning algorithms, including clustering and classification algorithms, can effectively analyze customer data to segment customers based on their preferences and behavior. The study emphasizes the critical role of accurate customer segmentation in enhancing customer service and refining business strategies in the hospitality industry. Additionally, it explores the challenges and opportunities

associated with utilizing machine learning in hospitality customer segmentation, highlighting its potential to enhance customer experiences and drive business growth in the competitive hospitality sector.

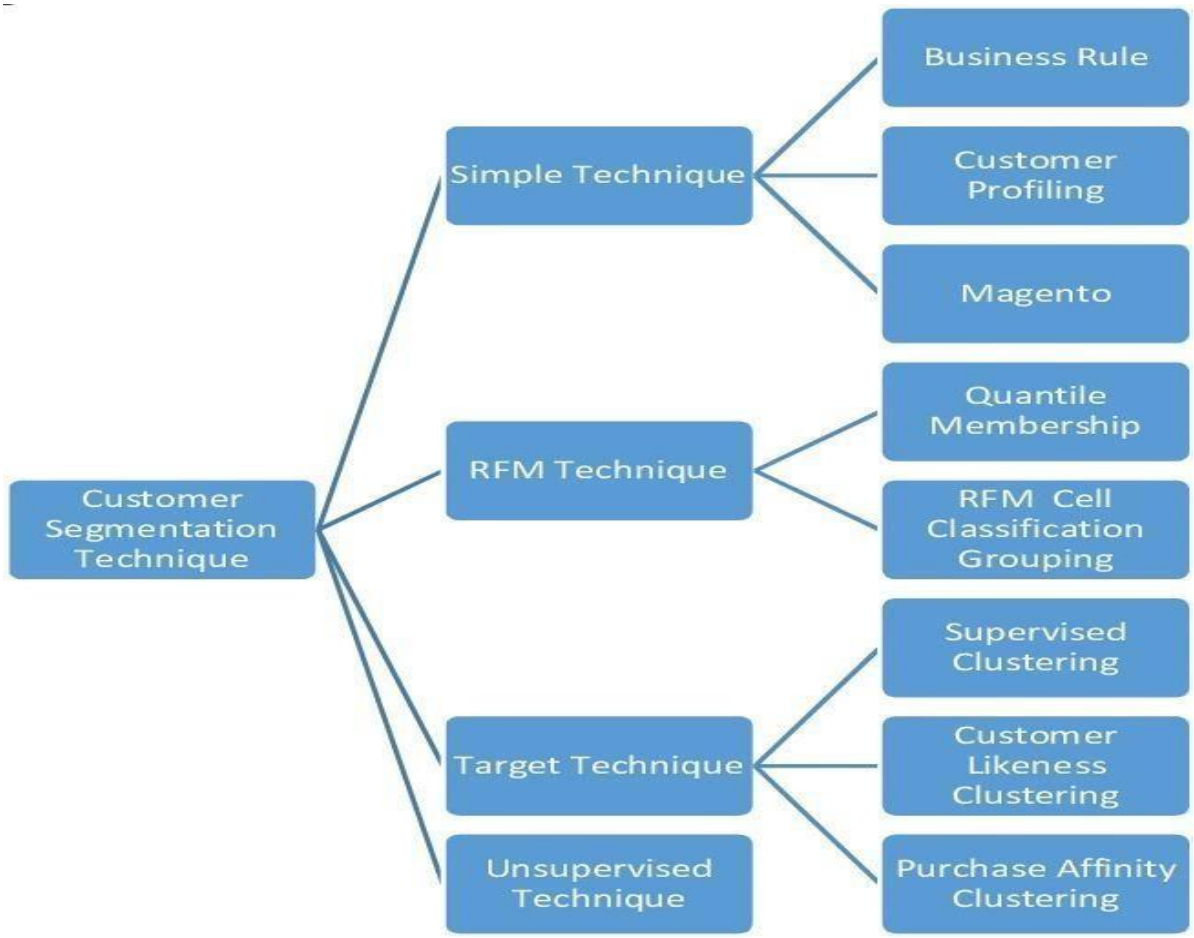


**Fig. 2.6 Customer Segmentation Technique**

# CHAPTER 3

# OBJECTIVE AND METHODOLOGY

# Chapter 3

# Objective and Methodology

## 3.1 Objective

- Design a robust recommendation system employing collaborative filtering techniques.
- Curate and preprocess a comprehensive dataset of user interactions for training the recommendation model.
- Enhance model performance by fine-tuning hyperparameters and exploring diverse algorithms.
- Validate the accuracy and reliability of the recommendation system using distinct test datasets.
- Integrate the recommendation system with an intuitive user interface for seamless user experience and adoption.
- Implement real-time data processing for dynamic updates based on user interactions.
- Use contextual and demographic information to improve recommendation accuracy.
- Track key performance indicators and regularly refine algorithms based on user feedback and performance metrics.

## 3.2 Methodology

In today's fast-paced retail environment, optimizing inventory management and providing personalized customer experiences are critical to staying competitive. This project aims to tackle these challenges by integrating advanced machine learning techniques with traditional approaches. By leveraging convolutional neural networks (CNNs) for age and gender detection from facial images and implementing collaborative filtering for product recommendations, we aim to enhance both customer satisfaction and inventory efficiency. The project also includes developing a real-time face monitoring system and a comprehensive inventory management platform, creating a robust solution tailored for modern retail needs. This methodology outlines the steps taken to achieve these objectives, ensuring a seamless integration of technology and practical application for retailers.

**1. Data Collection:**Gather facial image datasets for age and gender detection, ensuring diversity in age, gender, and ethnicity for robust model training. Use Amazon customer rating datasets for product recommendation.

**2. Data Preprocessing**: Preprocess facial images to enhance quality, including normalization, resizing, and grayscale conversion. Clean and preprocess the Amazon dataset to remove duplicates and irrelevant information.

**3. Age and Gender Detection**: Develop a CNN model with convolutional and pooling layers for accurate age and gender classification. Use techniques like transfer learning with pre-trained models to improve performance.

**4. Product Recommendation:** Implement collaborative filtering using Singular Value Decomposition (SVD) on the Amazon dataset to provide personalized product recommendations based on user demographics.

**5. Real-Time Face Monitoring**: Create a Django front-end for real-time face monitoring, enabling users to view live face detection and analysis outcomes. Utilize OpenCV for preprocessing and real-time face detection.

**6. Inventory Management**: Develop an inventory management system for retailers to choose products and place orders. Implement features for inventory tracking, order processing, and reporting.

**7. Training**: Split the dataset into training, validation, and testing sets. Train the age and gender detection model using CNNs and the collaborative filtering model using SVD.

**8. Evaluation**: Evaluate model performance using metrics like accuracy, precision, and recall. Validate the effectiveness of product recommendations and inventory management features.

**9. Deployment**: Integrate the trained models and inventory management system into a unified application. Ensure the system is user-friendly and meets the requirements of retailers.

**10. Iterative Improvement**: Gather feedback from users and retailers to improve the system's performance and user experience. Continuously update the models and features based on new data and emerging technologies.


**Fig. 3.2.1 Methodology**

# CHAPTER 4

# SYSTEM DESIGN

# Chapter 4

# System Design

## 4.1 System Architecture

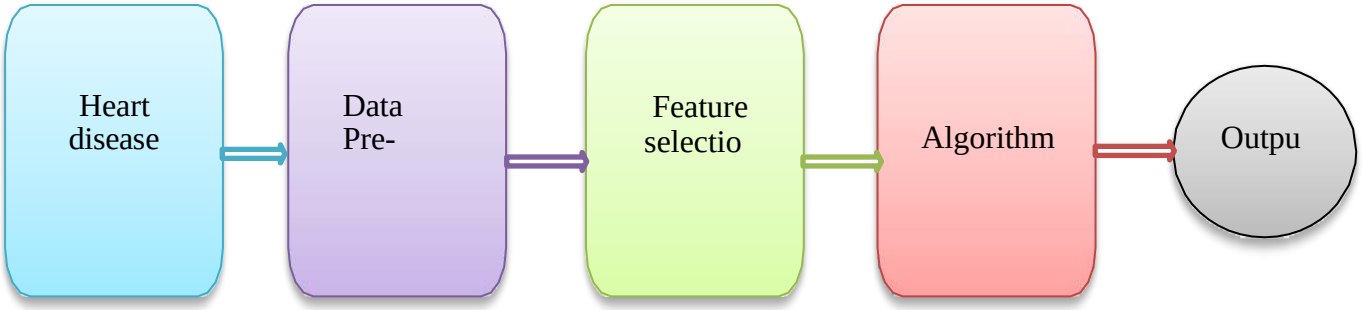


**Fig. 4.1.1: System Architecture**

The system architecture for strategic demand profiling utilizing machine learning is a comprehensive framework designed to enhance demand forecasting and inventory management efficiency. It begins with the Data Ingestion Layer, which gathers data from various sources and employs ETL processes to cleanse and integrate it into a centralized repository. Subsequently, the Data Storage Layer stores this cleansed data in scalable storage systems like data warehouses or data lakes. The Preprocessing Layer normalizes and transforms the data, ensuring consistency and readiness for analysis by employing techniques such as feature extraction and outlier detection. Moving forward, the Machine Learning Layer applies algorithms like Singular Value Decomposition (SVD) and predictive models such as regression and time-series forecasting to forecast demand accurately. Continuous model training and refinement using historical data and real-time inputs ensure the adaptability and reliability of the system. The Analysis and Profiling Layer then analyzes model outputs, generating insightful demand profiles presented through intuitive dashboards and reports. These insights are then utilized by the Decision Support Layer, which integrates with inventory management systems to optimize stock levels and offers actionable recommendations for procurement, production, and logistics planning

## 4.2 Data Flow Diagrams

A data flow diagram is the graphical representation of the flow of data through an information system. DFD is very useful in understanding a system and can be efficiently used during analysis.

A Data Flow Diagram shows the flow of data through system. It views a system as a function that transforms the inputs into desired outputs. Any complex systems will not perform this transformation in a single step and data will typically undergo a series of transformations before it becomes the output.

With a data flow diagram, users are able to visualize how the system will operate that the system will accomplish and how the system will be implemented. Old system DFDs can be drawn up and compared with a new system DFD to draw comparisons to implement a more efficient system.

Data flow diagrams can be used to provide the end user with a physical idea of where the data they input, ultimately as an effect upon the structure of the whole system.

### Dataflow Diagram- Level 0

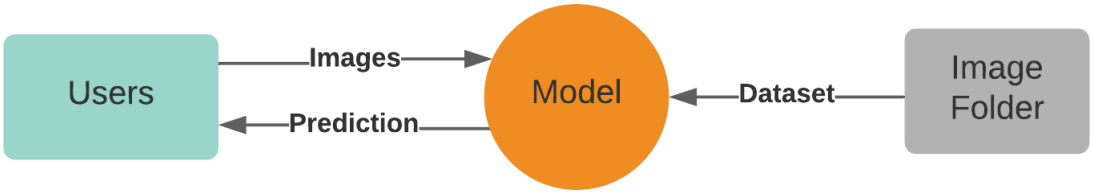


**Fig. 4.2.1: Data Flow Diagram Level 0**

### Data Flow Diagram- Level 1

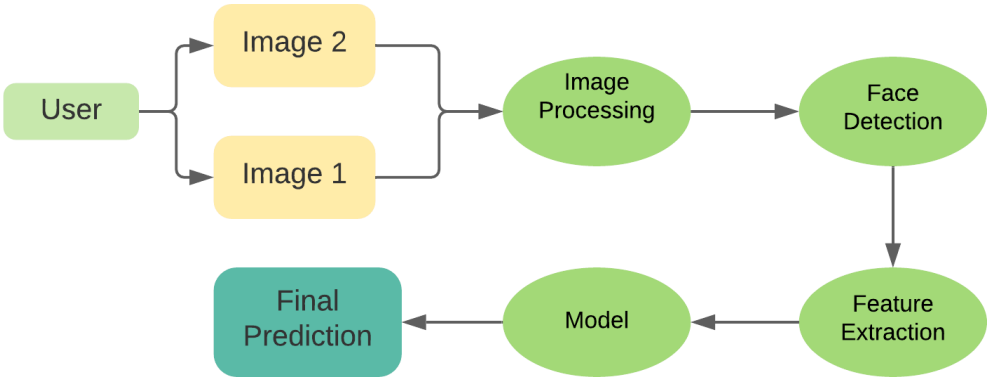


**Fig. 4.2.2: Data Flow Diagrams Level 1**

## Dataflow Diagram- Level 2

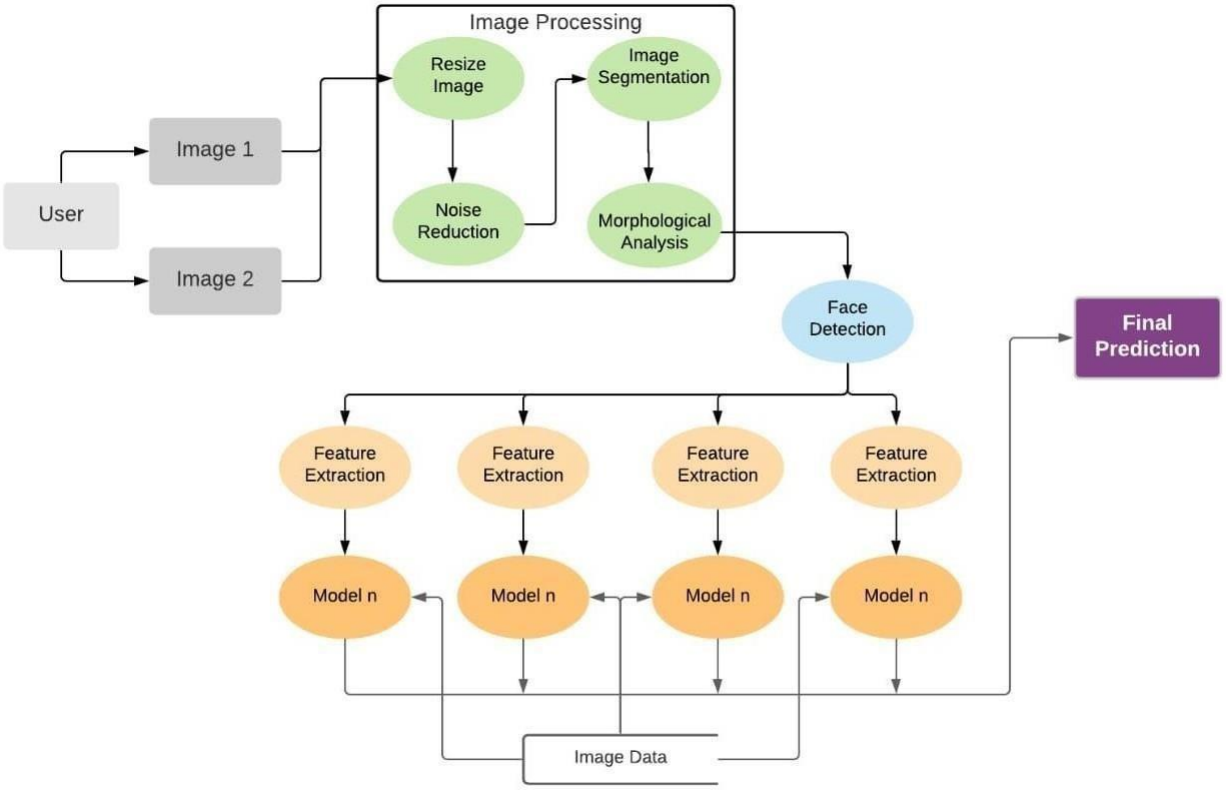


**Fig. 4.2.3: Data Flow Diagram Level 2**

The dataflow for a system involving image processing to final prediction begins with image input from a camera or image database, where the raw images undergo preprocessing steps such as resizing and normalization. The processed images then pass through a face detection module to identify and isolate faces. These detected faces are subjected to feature extraction to generate feature vectors representing key facial characteristics. The extracted features are fed into a trained model that produces predictions, such as identity or emotional state. Finally, the model's outputs are aggregated and interpreted to provide the final prediction, completing the flow from raw image to actionable insight.

## 4.3 Use Case Diagram

The external objects that interact directly with the systems are called the actors. Actors include humans, external devices and other software systems. The important thing about actors is that they are not under control of the application. In this project, user of the system is the actor.

To find use cases, for each actor, list the fundamentally different ways in which the actor uses the system. Each of these ways is a use case.

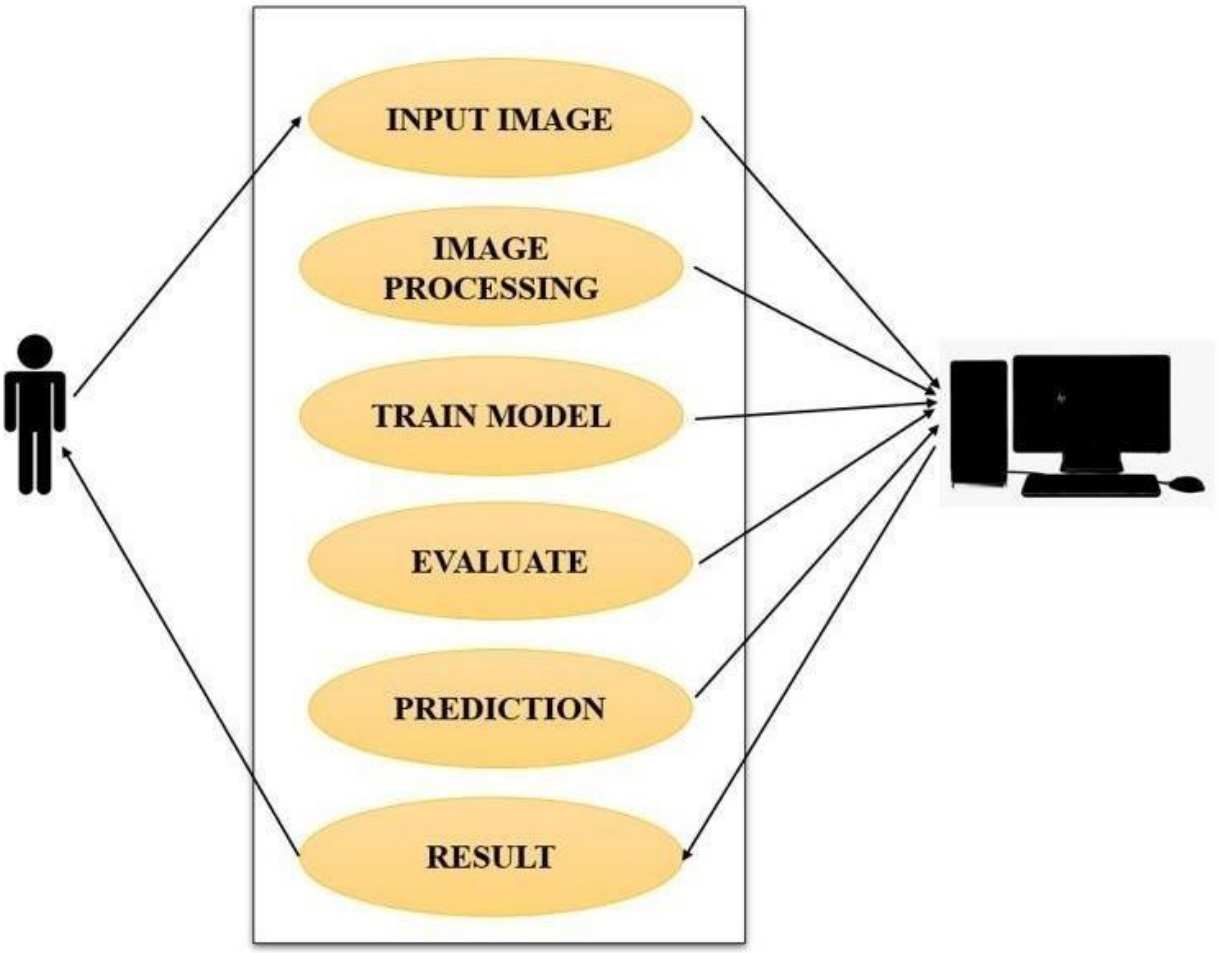


**Fig 4.3.1: Use case diagram**

## 4.4 Class Diagram

A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a subsystem by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes.

The class diagram is the main building block of the Object Oriented Modeling. It is used both for general conceptual modeling of the systematic of the application, and for the detailed modeling translating the models into programming code.

Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed.

Below section shows the class diagrams of the application.

## Class Diagram 1

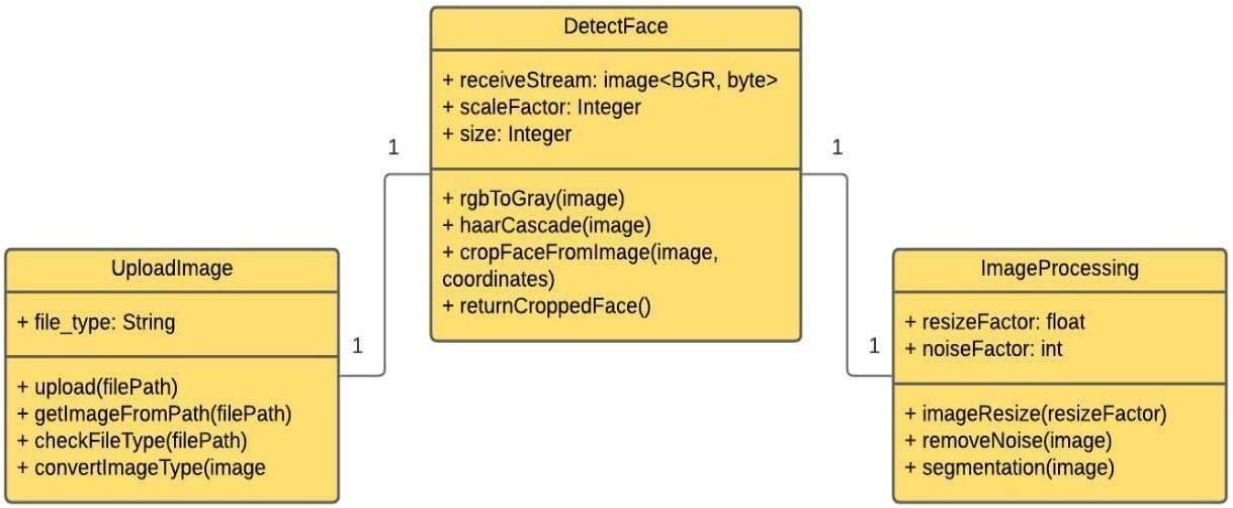


**Fig 4.4.1: Class Diagram 1**

## 4.5 Network Architecture Diagram:

The network architecture is designed to perform age and gender detection from facial images and generate product recommendations based on user information. The architecture includes a convolutional neural network (CNN) for image processing and feature extraction, followed by separate branches for age and gender prediction, as well as recommendation logic. The CNN processes preprocessed facial images to extract features, which are then used by the age and gender prediction branches to output the estimated age group and gender. Simultaneously, user information such as age, gender, and possibly other features is fed into the recommendation logic branch, which generates product recommendations based on the user's profile. The network is trained using labeled data for age and gender prediction and user-product interaction data for recommendation, with separate output layers for each task. This architecture allows the model to learn complex patterns from both image and user data, providing accurate age and gender predictions and personalized product recommendations.

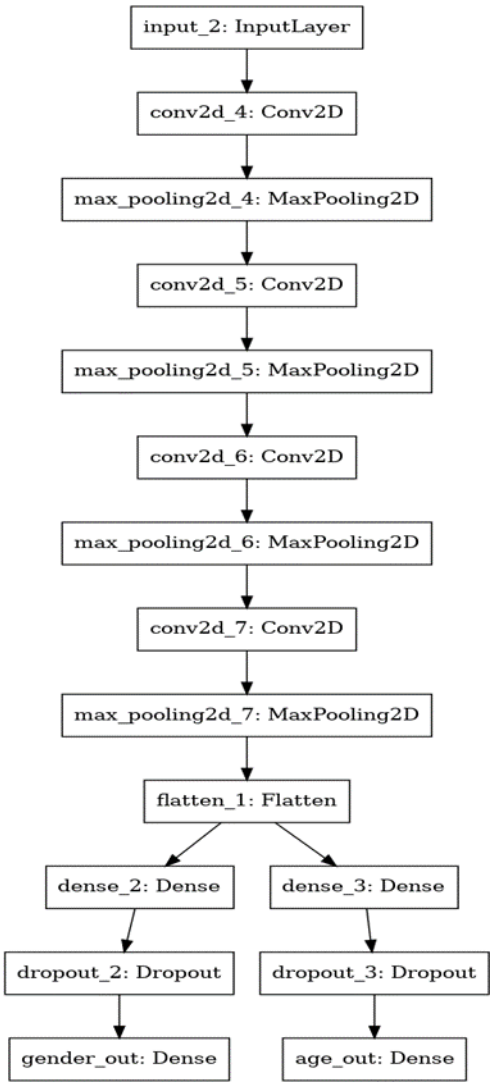


**Fig 4.5.1: CNN Network Architecture**

## 4.6 Sequence Diagram

A Sequence diagram in a Unified Modeling Language is a kind of interaction diagram that shows how processes operate with one another and in what order. It shows the participants in an interaction and the sequence of messages among them; each participant is assigned a column in a table.

Below section shows the sequence diagram in this application.

## Sequence Diagram

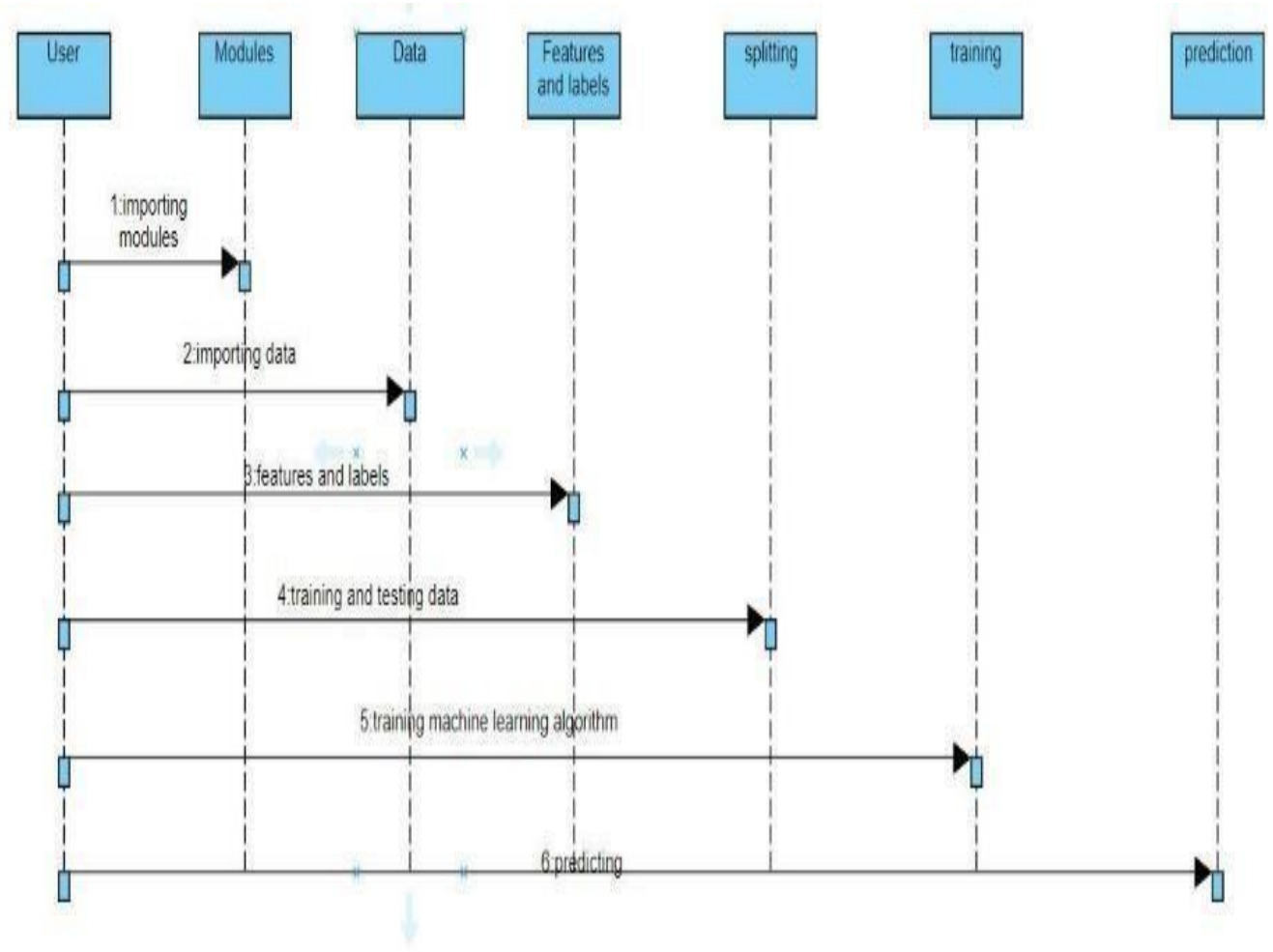


**Fig 4.6.1: Sequence Diagram**

## 4.7 Workflow Diagram

A collaboration diagram, also known as a communication diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML). These diagrams can be used to portray the dynamic behavior of a particular use case and define the role of each object.

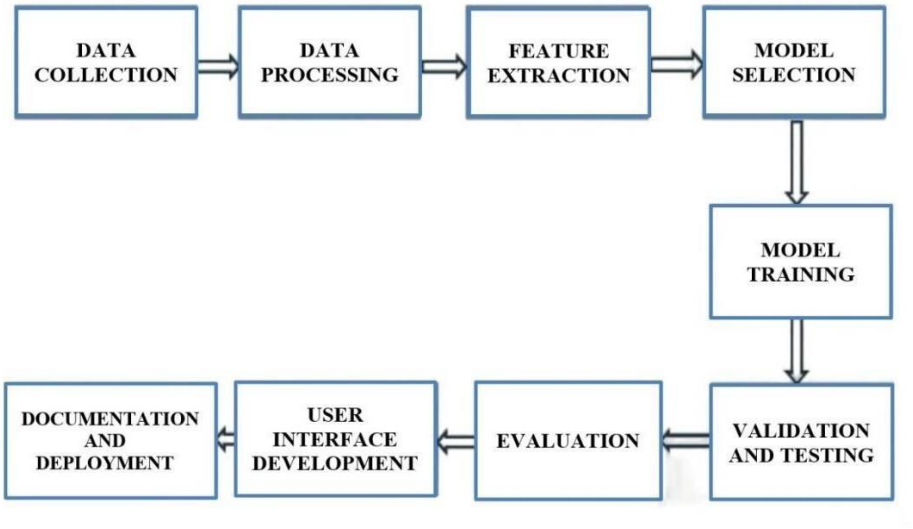


**Fig. 4.7.1: Workflow Diagram**

## 4.8 Activity Diagram

Activity diagram is another important behavioral diagram in UML diagram to describe dynamic aspects of the system. Activity diagram is essentially an advanced version of flow chart that modeling the flow from one activity to another activity.

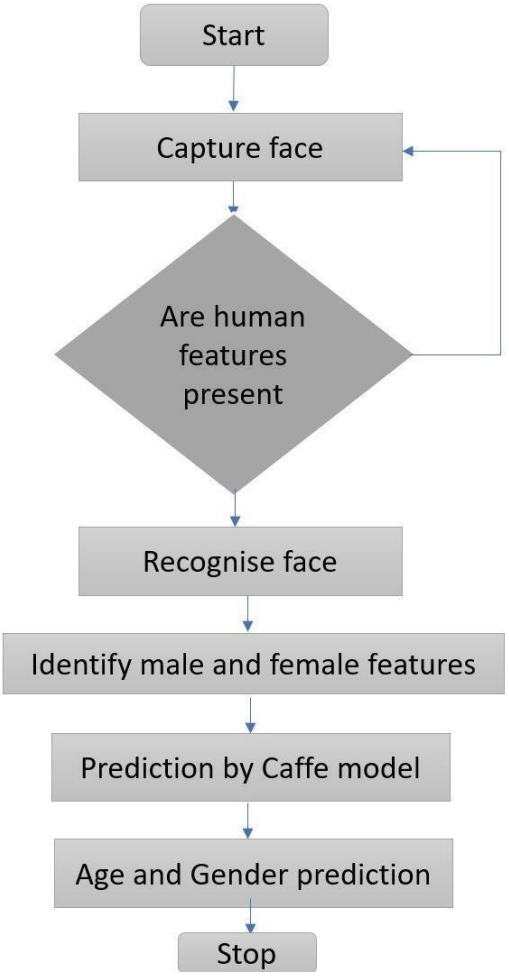


**Fig 4.8.1: Activity Diagram**

# 4.9 Module Split-up

**1. Data Management and Preparation:**

- Import necessary libraries for data handling.

- Read and preprocess product data from CSV files, ensuring data cleanliness.

- Segment the dataset into distinct training and testing subsets for subsequent analysis.

**2. Recommendation Generation Pipeline:**

- Employ popularity-based algorithms to ascertain highly sought-after products.

- Implement collaborative filtering methodologies to formulate tailored recommendations based on user preferences.

**3. Integration with Django Web Framework:**

- Develop Django views to facilitate diverse functionalities, including homepage rendering and user interactions.

- Establish URL routing mechanisms to map views to corresponding URLs, ensuring seamless navigation within the web application.

**4. Image Processing and Prediction Mechanism:**

- Utilize deep learning models for proficient face detection and accurate prediction of user demographics from uploaded images.

- Invoke recommendation functions predicated on the predicted demographic attributes to offer personalized product suggestions.

**5. Aggregation of Recommendations and HTML Rendering:**

- Combine recommendations derived from various sources, eliminating redundancies for enhanced user experience.

- Prepare context data and dynamically render HTML templates, optimizing the presentation of recommendations across different sections of the web store interface.

These steps encapsulate the comprehensive workflow of the code, encompassing data management, recommendation generation, web application integration, image processing, and HTML rendering for efficient user interaction.

-

# CHAPTER 5

# IMPLEMENTATION

# Chapter 5

# Implementation

## Open CV

OpenCV is a library of programming functions mainly aimed at real-time computer vision. It is developed by Intel research center and subsequently supported by1Willow Garage and now maintained by itseez. It is written in C++ and its primary interface is also in C++. Its binding is in Python, Java, and Mat lab. OpenCV runs on a variety of platform i.e.

Windows, Linux, and MacOS, OpenBSD in desktop and Android, IOS and Blackberry in mobile. It is used in diverse purpose for facial recognition, gesture recognition, object identification, mobile robotics, segmentation etc. It is a combination of OpenCV C++ API and Python language. In our project we are using OpenCV version 2 OpenCV is used to gesture control to open a camera and capture the image. It is also used in the image to text and voice conversion technique.

It has a modular structure, which means that the package includes several shared or static libraries. We are using image processing module that includes linear and non-linear image filtering, geometrical image transformations (resize, affine and perspective warping, and generic table-based remapping), color space conversion, histograms, and so on. Our project includes libraries such as Viola-Jones or Haar classifier, LBPH (Lower Binary Pattern histogram) face recognizer, Histogram of oriented gradients (HOG).

Purpose of Image processing:

The purpose of image processing is divided into 5 groups. They are:

1. Visualization- Observe the objects that are not visible.
2. Image sharpening and restoration- To create a better image.
3. Image retrieval- Seek for the image of interest.
4. Measurement of pattern– Measures various objects in an image.
5. Image Recognition– Distinguish the objects in an image.

**MACHINE LEARNING TECHNIQUES**

Machine learning is a data analytics technique that teaches computers to do what comes naturally to humans and animals learn from experience for data analysis.

Benefits:

- Machine Learning algorithms are capable of learning from the data we provide.
- Automation for everything
- Trends and patterns identification.
- Wide range of applications.
- Data Acquisition.
- Highly error-prone

**DEEP LEARNING**

Deep learning is part of a broader family of machine learning methods based on artificial neural networks with representation learning

Benefits:

- No Need for Feature Engineering.
- Best Results with Unstructured Data.
- No Need for Labeling of Data.
- Neural Networks at the Core of Deep Learning are Black Boxes

# 5.1 ALGORITHMS AND TECHNIQUES USED

## 5.1.1 Single Value Decomposition

The Singular Value Decomposition (SVD) algorithm is a powerful machine learning technique used in strategic inventory management to optimize inventory levels and improve demand forecasting based on the concept of matrix factorization.

SVD algorithm

Input inventory data.

Matrix Decomposition:

- Decompose the inventory data matrix into three matrices: U, Σ, and V^T.

Decomposition:

- Inventory data matrix is factorized into three component matrices.
- Matrix U represents the inventory-to-item relationship.
- Matrix Σ is a diagonal matrix of singular values.
- Matrix V^T represents the item-to-item relationship.
- SVD algorithm has four steps:
- Matrix Factorization
- Reduction to Diagonal Form
- Reconstruction
- Application in Inventory Management
- Matrix Factorization:

The inventory data matrix is decomposed using SVD, which separates the original matrix into three simpler matrices. These matrices help in identifying patterns and relationships within the data, crucial for demand forecasting and inventory optimization.

SVD Decomposition:

Each element in the inventory data matrix is approximated using the product of the three matrices U, Σ, and V^T. This process reduces the complexity of the data, making it easier to analyze and derive insights. To handle large datasets efficiently, SVD reduces the dimensionality of the data while preserving its essential structure.

Integral Image:

To manage the complexity and computation load, integral images can be used in conjunction with SVD to quickly compute sums of values in sub-matrices. This technique is similar to creating a summed area table, which facilitates rapid calculations and data manipulations.

Reduction to Diagonal Form:

The decomposition process involves reducing the original matrix to a diagonal form, where the diagonal elements represent the singular values. These values indicate the importance and variance of each component in the dataset, allowing for effective dimensionality reduction by retaining the most significant components.

Adaboost Machine Learning Algorithm:

To enhance the prediction accuracy and reduce redundancy, the Adaboost algorithm can be integrated with SVD. Adaboost combines multiple weak classifiers to form a strong classifier, improving the robustness of demand forecasts and inventory recommendations. Each weak classifier is trained on different subsets of data, and their combined output provides a more accurate prediction model.

Cascading Classifiers:

In strategic inventory management, cascading classifiers can be employed to filter out less relevant data points at each stage. This approach ensures that only the most pertinent information is considered for final decision-making, improving efficiency and accuracy. Each stage in the cascade applies increasingly stringent criteria to the data, refining the results progressively.

Reconstruction:

Using the decomposed matrices, the original inventory data matrix can be reconstructed with reduced noise and enhanced clarity. This reconstructed matrix is utilized for further analysis, such as identifying trends, forecasting demand, and optimizing inventory levels. The SVD process ensures that the reconstructed data retains the essential patterns and relationships from the original dataset.

Application in Inventory Management:

The final phase involves applying the insights gained from the SVD analysis to strategic inventory management. This includes demand forecasting, inventory optimization, and identifying key inventory drivers. The SVD-based approach allows for a more data-driven and accurate inventory management strategy, reducing stockouts and overstock situations.

In this project, SVD is combined with advanced preprocessing techniques to handle data inconsistencies and enhance the reliability of the inventory management system. Preprocessing steps like normalization and outlier detection ensure that the input data is clean and suitable for analysis. This leads to better model performance and more reliable inventory management outcomes.

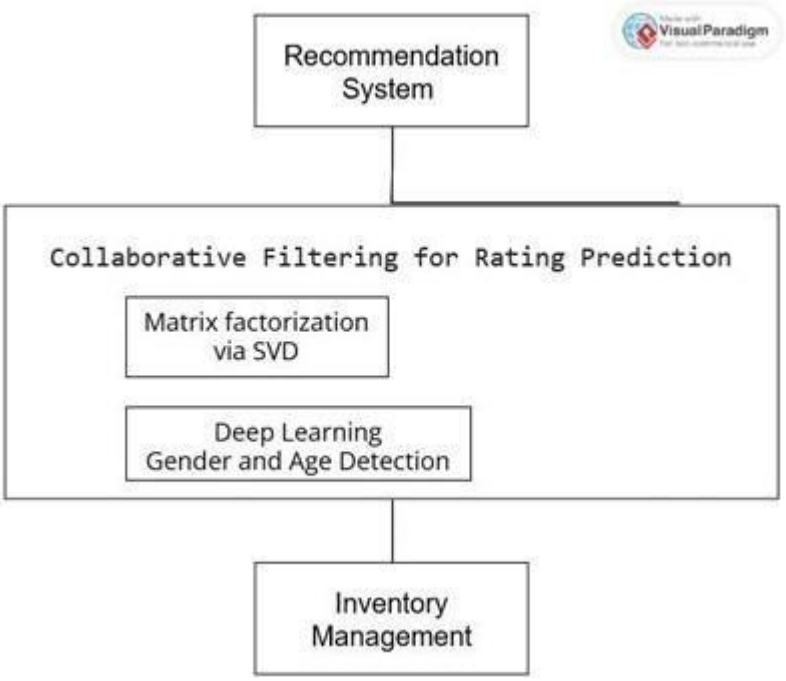


**Fig. 5.1.1.1 Inventory Management Using Deep Learning and SVD**

## 5.1.2 CNN

- Each plane of convolution layer is associated with one or more feature maps of earlier layer.
- Convolution mask is used.
- The convolution is computed.
- The outputs of convolution layers are added together with an adjustable scalar, called as bias.
- Lastly, an activation function is applied to obtain the planes output (feature map)
- Each plane in the convolution layer produces one scalar output.
- The purpose of this layer is to extract low-level features such as edges and texture.
- Feature map of convolution layer is calculated as:

$$y_n^l = f_l\left(\Sigma_{m \in V_n^l} y_m^{l-1} \otimes w_{m,n}^l + b_n^l\right)$$

Where $w_{m,n}^l$ is the convolution mask, $b_n^l$ is the bias term, and $V_n^l$ is the list of plane

## Pooling (sub-sampling) layer

- The dimensionality of each feature map is reduced by spatial pooling.
- It can be of three different types: Max pooling - takes the largest element,
- Average pooling- takes the average of the elements, and
- Sum pooling - takes the sum of all the elements.
- The main function of pooling is to reduce the spatial size of the input to make it convenient.
- This result is then passed through the activation function to produce the outputs.
- This feature map is connected to one or more planes of the next convolution layer.
- Feature map of sub-sampling layer is calculated as:

$$y_n^l = f_l(Z_n^{l-1} \times w_n^l + b_n^l) \quad (2)$$

where $Z_n^{l-1}$ is matrix obtained by summing all four pixels of a block, $w_n^l$ is the weight and $b_n^l$ is the bias term [32].

## Output layer (fully connected layer)

- In AIFR-CNN, the output layer is constructed from sigmoidal neuron.
- Generally, the outputs of this layer are the outputs of the network.
- In the output layer, soft max activation function is used.
- Other classifiers like SVM can also be used.
- These fully connected layers capture the correlations between features of various parts of the face
- The Convolution and Pooling layers in combination are used for feature extraction.
- fully connected layers are used for classification.
- The output of sigmoidal neuron is calculated as:

$$y_n^L = f^L\left(\sum_{m=1}^{N^{L-1}} y_m^{L-1} w_{m,n}^L + b_n^L\right) \quad (3)$$

where $N^L$ is the number of output sigmoidal neurons, $w_{m,n}^L$ is weight from feature map m of the last convolution layer to neuron n of the output layer, and $b_n^L$ is the bias of neuron n associated with layer L [32].

### 5.1.3 Logistic Regression

Logistic Regression, a widely used statistical technique, is employed to forecast binomial outcomes, where y can take values of 0 or 1. This method predicts categorical results (binomial or multinomial) by estimating the probabilities of an event occurring, specifically the probability of y equalling 1, based on given input variable values x. Consequently, Logistic Regression yields predictions ranging from 0 to 1. It models the data points using curve, expressed by a specific equation.

$$\frac{1}{1+e^{-x}} = \frac{e^{x}}{1+e^{x}}$$

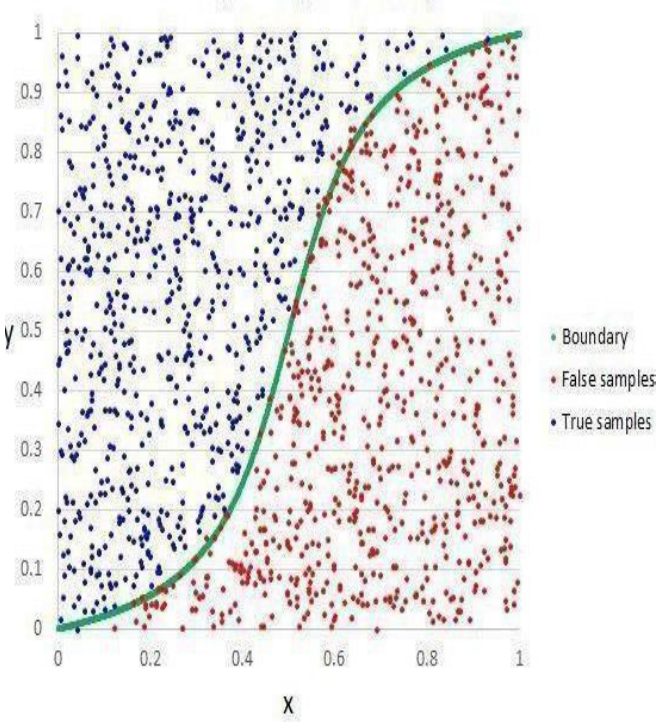


**Fig. 5.1.3.1 Logistic Regression**

### 5.1.4 Collaborative Filtering (CF):

Collaborative filtering is a technique used in recommendation systems to provide personalized recommendations to users based on their past behavior and preferences, as well as the behavior of other similar users. It works by finding similarities between users or items and using these similarities to make predictions or recommendations.

There are two main types of collaborative filtering:

1. User-based collaborative filtering: This approach recommends items that similar users have liked in the past. It works by first finding users who are similar to the target user based on their past interactions (e.g., ratings, purchases). Then, it recommends items that these similar users have liked but the target user has not interacted with yet.

2. Item-based collaborative filtering: This approach recommends items that are similar to items that the target user has liked in the past. It works by first calculating the similarity between items based on how often they are liked or rated together by users. Then, it recommends items that are most similar to the ones the target user has liked.

Collaborative filtering is widely used in e-commerce, social media, and content streaming platforms to provide personalized recommendations to users. It is often combined with other recommendation techniques, such as content-based filtering (which considers the properties of items and users' preferences for those properties) to improve the quality of recommendations.

### 5.1.5 K-Nearest Neighbours (KNN):

K-Nearest Neighbours (KNN) is a simple and intuitive machine learning algorithm that falls under the supervised learning technique. It operates on the principle of similarity between data points. When a new data point is encountered, KNN compares it with existing data points and assigns it to the category that is most similar to it.

KNN stores all the available data points and classifies new data points based on their similarity to the existing data. This allows it to easily categorize new data into appropriate classes.

While KNN can be used for both regression and classification tasks, it is primarily used for classification. It is considered a non-parametric algorithm because it does not make any assumptions about the underlying data distribution.

KNN is often referred to as a lazy learner because it does not actively learn from the training data. Instead, it stores the entire dataset and performs classification only when a new data point is encountered.

In summary, KNN is a flexible and easy-to-implement algorithm that can be used for classification tasks, especially when the underlying data distribution is not well understood.

The K-NN working can be explained on the basis of the below algorithm:

- **Step-1:** Select the number K of the neighbours
- **Step-2:** Calculate the Euclidean distance of **K number of neighbours**
- **Step-3:** Take the K nearest neighbours as per the calculated Euclidean distance.
- **Step-4:** Among these k neighbours, count the number of the data points in each category.
- **Step-5:** Assign the new data points to that category for which the number of the neighbours is maximum.
- **Step-6:** Our model is ready.

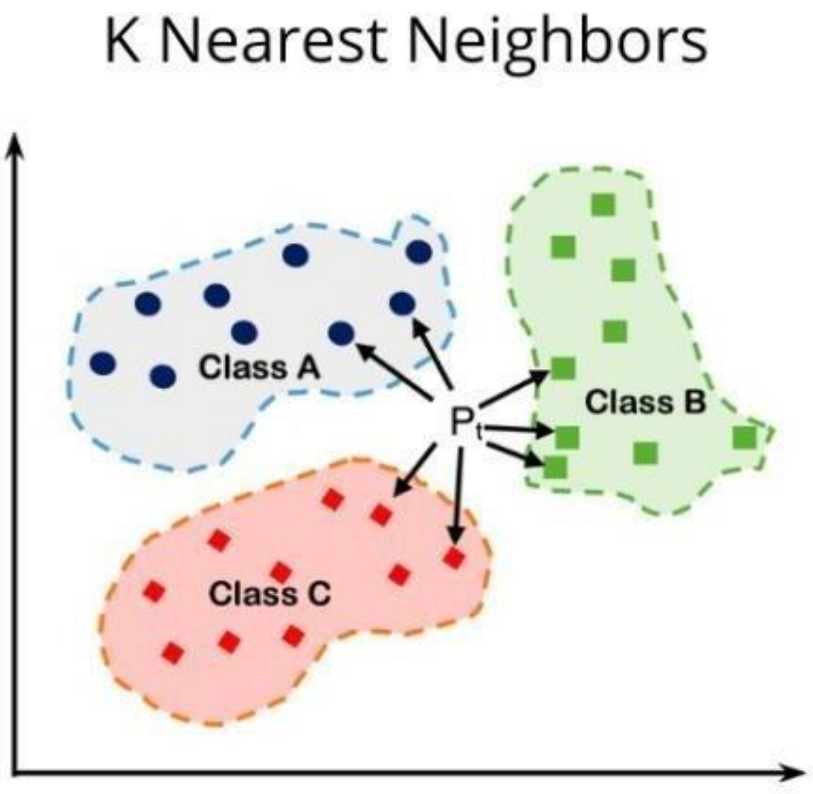


**Fig. 5.1.5.1 KNN**

### 5.1.5 Cosine Similarity:

Cosine similarity is a measure used to determine how similar two vectors are in a multi-dimensional space. It measures the cosine of the angle between two non-zero vectors and provides a value between -1 and 1, where 1 means the vectors are identical, 0 means the vectors are orthogonal (not similar at all), and -1 means the vectors are diametrically opposed.

In the context of information retrieval and text mining, cosine similarity is often used to compare documents based on their content. Each document is represented as a vector where each dimension corresponds to a term, and the value of each dimension is the frequency or tf-idf weight of that term in the document.

Cosine similarity is calculated as the dot product of the two vectors divided by the product of their magnitudes:

Similarity(**A**,**B**)=‖**A**‖‖**B**‖ / **A**·**B**

Where:

- A**A** and B**B** are the two vectors being compared.
- A·B**A**·**B** is the dot product of A**A** and B**B**.
- ‖A‖‖**A**‖ and ‖B‖‖**B**‖ are the magnitudes of A**A** and B**B** respectively.

Cosine similarity is advantageous because it is independent of the vector's magnitude and is based solely on the direction of the vectors. This makes it a useful metric for comparing documents or text data, where the length of the document (i.e., the number of words) can vary widely.

## 5.2 Working

1. **Initialization and Imports**:

- Import the **pandas** library for data processing, **scikit-learn** for machine learning tasks, and **Django** for web development.
- Set up database connections and configure Django settings for the project.
- Install necessary dependencies using **pip** and ensure compatibility with the project environment.

2. **Data Preprocessing**:

- Load the dataset using **pandas** and perform basic data cleaning steps such as removing duplicates and handling missing values.
- Split the dataset into training and testing sets using **train_test_split** from **scikit-learn**.
- Group the data by relevant features such as user ID for collaborative filtering analysis.
- Perform Singular Value Decomposition (SVD) using **TruncatedSVD** from **scikit-learn** to extract latent factors for collaborative filtering.

3. **Recommendation Logic**:

- Define a function that recommends products based on user age and gender, using the preprocessed dataset.
- Implement collaborative filtering logic using user-item interactions to recommend products similar to those liked by the user.
- Utilize the SVD results to generate recommendations based on the latent factors extracted from the dataset.

4. **Image Processing**:

- Use the **cv2** module from OpenCV for image processing tasks such as face detection and gender classification.
- Load pre-trained deep learning models for age and gender detection, such as the **age_gender_net** model from OpenCV's DNN module.

- Process images by resizing, normalizing, and converting them to the required format for the deep learning models.

5. **Views and Templates**:

- Define Django views that render HTML templates for different pages of the web application.
- Create HTML templates using Django's template language for the homepage, recommendation pages, inventory management pages, and payment pages.
- Use Django's template inheritance to create a consistent layout and design across all pages of the application.

6. **Inventory Management**:

- Implement Django models for storing product information, including fields for product name, quantity, price, and category.
- Create views and templates for adding new products, updating product quantities, and deleting products from the inventory.
- Calculate total prices based on the quantities of products selected by the user and display the total on the checkout page.

7. **Payment Integration**:

- Integrate a payment gateway such as Stripe or PayPal into the Django application for processing payments.
- Create views for rendering payment pages and processing payment transactions using the selected payment gateway.
- Ensure the security of payment transactions by implementing secure communication protocols and encryption.

8. **Routing and URL Mapping**:

- Define URL patterns in Django's **urls.py** file to map different URLs to the corresponding views in the application.
- Ensure that URL patterns are correctly mapped to views to handle user requests and responses.

## 5.3 Code

### 5.3.1 Importing Necessary Libraries:

```
from collections import defaultdict
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
import math
import json
import time
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.model_selection import train_test_split
from sklearn.neighbors import NearestNeighbors
import scipy.sparse
from scipy.sparse import csr_matrix
from scipy.sparse.linalg import svds
import warnings
warnings.simplefilter('ignore')
import plotly.express as px
from django.shortcuts import render
import cv2
import numpy as np
from keras.models import load_model
import os
import matplotlib.pyplot as plt
import seaborn as sns
from tqdm.notebook import tqdm
import tensorflow as tf
from keras.preprocessing.image import load_img
from PIL import Image
import pandas as pd # Import pandas to use DataFrame functionalities
from django.shortcuts import render, redirect
from django.contrib.auth.forms import UserCreationForm, AuthenticationForm
from django.contrib.auth import login, authenticate, logout
from datetime import datetime
import tempfile
from .models import Product
from django.http import JsonResponse,HttpResponse
from .forms import ProductForm,QuantityForm,CameraForm,UploadFileForm
from django.contrib import messages
from django.shortcuts import get_object_or_404
from django.http import HttpResponseBadRequest
import numpy as np
from scipy.sparse.linalg import svds
```

### 5.3.2 Age and Gender Detection:

```
def age_gender_detector(request):
    faceProto = "opencv_face_detector.pbtxt"
    faceModel = "opencv_face_detector_uint8.pb"

    ageProto = "age_deploy.prototxt"
    ageModel = "age_net.caffemodel"

    genderProto = "gender_deploy.prototxt"
    genderModel = "gender_net.caffemodel"

    faceNet = cv2.dnn.readNet(faceModel, faceProto)
    ageNet = cv2.dnn.readNet(ageModel, ageProto)
    genderNet = cv2.dnn.readNet(genderModel, genderProto)

    MODEL_MEAN_VALUES = (78.4263377603, 87.7689143744, 114.895847746)
    ageList = ['(0-2)', '(4-6)', '(8-12)', '(15-20)', '(25-32)', '(38-43)', '(48-53)', '(60-100)']
    genderList = ['Male', 'Female']

    video = cv2.VideoCapture(0
    padding = 20
    output_dir = "Dirs"
    if not os.path.exists(output_dir):
        os.makedirs(output_dir)

    last_capture_time = datetime.now()
    while True:
        hasFrame, vidFrame = video.read()

        if not hasFrame:
            cv2.waitKey()
            break

        frame, faceBoxes = getFaceBox(faceNet, vidFrame)

        if not faceBoxes:
            print("No face detected")
        else:
            current_time = datetime.now()
            if (current_time - last_capture_time).seconds >= 5:
                for faceBox in faceBoxes:
                    face = frame[max(0, faceBox[1] - padding):min(faceBox[3] + padding, frame.shape[0] -
1),
                                 max(0, faceBox[0] - padding):min(faceBox[2] + padding, frame.shape[1] - 1)]
                    # Save the image to a temporary file
                    with tempfile.NamedTemporaryFile(suffix='.jpg', delete=False) as temp_img_file:
                        temp_img_path = temp_img_file.name
                        cv2.imwrite(temp_img_path, face)
                    # Pass the temporary file path to the predict function
                    predict(temp_img_path, pivot_df, preds_df, num_recommendations)
                    # Remove the temporary file
```

```
                os.unlink(temp_img_path)
                 last_capture_time = current_time

        for faceBox in faceBoxes:
            face = frame[max(0, faceBox[1] - padding):min(faceBox[3] + padding, frame.shape[0] - 1),
                 max(0, faceBox[0] - padding):min(faceBox[2] + padding, frame.shape[1] - 1)]

            blob = cv2.dnn.blobFromImage(face, 1.0, (227, 227), MODEL_MEAN_VALUES,
swapRB=False)

            genderNet.setInput(blob)
            genderPred = genderNet.forward()
            gender = genderList[genderPred[0].argmax()]

            ageNet.setInput(blob)
            agePred = ageNet.forward()
            age = ageList[agePred[0].argmax()]

            labelGender = "{}".format("Gender : " + gender)
            labelAge = "{}".format("Age : " + age + "Years")
            cv2.putText(frame, labelGender, (faceBox[0], faceBox[1] - 40),
cv2.FONT_HERSHEY_SIMPLEX, 0.8,
                        (0, 255, 255), 2, cv2.LINE_AA)
            cv2.putText(frame, labelAge, (faceBox[0], faceBox[1] - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.8,
                        (0, 255, 255), 2, cv2.LINE_AA)

        cv2.imshow("Age-Gender Detector", frame)

        # Listen for keyboard input
        key = cv2.waitKey(1)
        if key == 49: # If '1' is pressed
            return redirect('homepage')
            break

    # Release the video capture and destroy any OpenCV windows
    video.release()
    cv2.destroyAllWindows()
    return render(request, 'camera.html', {'video': video})“upload csv file”

def upload_file(request):
    if request.method == 'POST':
        form = UploadFileForm(request.POST, request.FILES)
        if form.is_valid():
            file = request.FILES['file']

            # Save the uploaded file to a temporary location
            file_path = "E:\\SSP\\Strategic\\save" + file.name
            with open(file_path, 'wb+') as destination:
                for chunk in file.chunks():
                    destination.write(chunk)
```

```
            # Parse the CSV file
            df = pd.read_csv(file_path)

            # Extract age and gender information
            # (Assuming the CSV file has 'Age' and 'Gender' columns)
            ages = df['Age']
            genders = df['Gender']

            # Make predictions for each row in the CSV file
            s = []
            num_recommendations = 20
            for age, gender in zip(ages, genders):
                if gender == 'male':
                    pivot_df = pivot_df_men
                    preds_df = preds_df_men
                elif gender == 'female':
                    pivot_df = pivot_df_women
                    preds_df = preds_df_women
                recommend_items(age, pivot_df, preds_df, num_recommendations) # Implement this
function based on your recommendation logic
                messages.success(request, 'File uploaded successfully!')
        else:
            form = UploadFileForm()

        return render(request, 'upload.html', {'form': form})
```

**5.3.3 Prediction and recommended items:**

```
def predict(image_path, pivot_df, preds_df, num_recommendations):
    # Load the image
    img = load_img(image_path, color_mode="grayscale")
    img = img.resize((128, 128), 3) # Using numerical value for anti-aliasing
    img = np.array(img)/255

    # Make predictions using the model
    pred = model.predict(img.reshape(1, 128, 128, 1))
    pred_gender = gender_dict[round(pred[0][0][0])]
    pred_age = round(pred[1][0][0])
    if(pred_age<=18):
        if(pred_gender=='Men'):
            pred_gender='Boys'
        elif(pred_gender=='Women'):
            pred_gender='Girls'
    agemin = pred_age - 5
    agemax = pred_age + 5
    print("Predicted Gender:", pred_gender, "Predicted Age:", str(agemin), "-", str(agemax))
    age = pred_age
    gen=pred_gender

    # Select the appropriate pivot_df and preds_df based on the predicted gender
    if pred_gender == 'Men':
        pivot_df = pivot_df_men
        preds_df = preds_df_men
    elif pred_gender == 'Women':
        pivot_df = pivot_df_women
        preds_df = preds_df_women
    elif pred_gender == 'Boys':
        pivot_df = pivot_df_boys
        preds_df = preds_df_boys
    elif pred_gender == 'Girls':
        pivot_df = pivot_df_girls
        preds_df = preds_df_girls
    # Call the recommend_items function
    recommend_items(age, pivot_df, preds_df, num_recommendations)
```

“recommended items”

```
def recommend_items(age,pivot_df, preds_df, num_recommendations):
    global temp
    if age < 1 or age > len(preds_df):
        return # Skip recommendation if age is out of range
    user_idx = age - 1
    sorted_user_predictions = preds_df.iloc[user_idx].sort_values(ascending=False)
    temp =pd.concat([sorted_user_predictions], axis=1)

    temp.index.name = 'Recommended Items'
    temp.columns = ['user_predictions']
    s.append(temp)
```

**5.3.4 Inventory Management:**

```
def add_to_inventory_back(request):
    if request.method == 'POST':
        # Create instances of the forms
        product_form = ProductForm(request.POST)

        # Process individual item additions
        if product_form.is_valid():
            product_form.save()  # Save the product
            return redirect('/homepage/recommendation')  # Redirect to wherever appropriate

        # Process selected items
        elif 'selected_products' in request.POST:
            selected_product_ids = request.POST.getlist('selected_products')
            for product_id in selected_product_ids:
                product_name = request.POST.get('name' + product_id)
                product_price = request.POST.get('price' + product_id)
                prototal_price=product_price

                # Save selected products with quantities
                product = Product(name=product_name, price=product_price,prototal=product_price)
                product.save()
            return redirect('/homepage/recommendation') # Redirect to wherever appropriate

    return redirect('/homepage/recommendation')

def inventory(request):
    products = Product.objects.all()
    total_price = sum(product.prototal for product in products)

    if request.method == 'POST':
        for product in products: # Loop through all products to handle each form individually
            # Construct field names based on product ID
            quantity_field_name = f'number_{product.id}'
            product_id_field_name = f'product_id_{product.id}'

            # Check if the form fields exist in the request.POST data
            if quantity_field_name in request.POST and product_id_field_name in request.POST:
                quantity = int(request.POST[quantity_field_name]) # Extract quantity value
                product_id = int(request.POST[product_id_field_name])  # Extract product ID

                product = Product.objects.get(pk=product_id)
                product.prototal = product.price * quantity
                product.save()

        # Recalculate total price after updating product quantities
        total_price = sum(product.prototal for product in products)
        return redirect('inventory')

    return render(request, 'display_inventory.html', {'products': products, 'total_price': total_price})
```

# CHAPTER 6

# RESULT

# 6. Results

The result of the project is a real-time system capable of detecting age and gender from facial images. It uses deep learning models for age and gender detection, achieving accurate and fast results. Additionally, the system provides personalized product recommendations based on the detected age and gender, enhancing user experience and potentially increasing sales for retail or e-commerce applications. The project also includes an inventory management component, allowing retailers to manage their stockand orders efficiently based on the recommendations. Overall, the project demonstrates the successful integration of computer vision, machine learning, and recommendation systems for practical applications in retail and marketing.


**Fig 6.1.1 Home Page 1**


**Fig 6.1.2 Home Page 2**

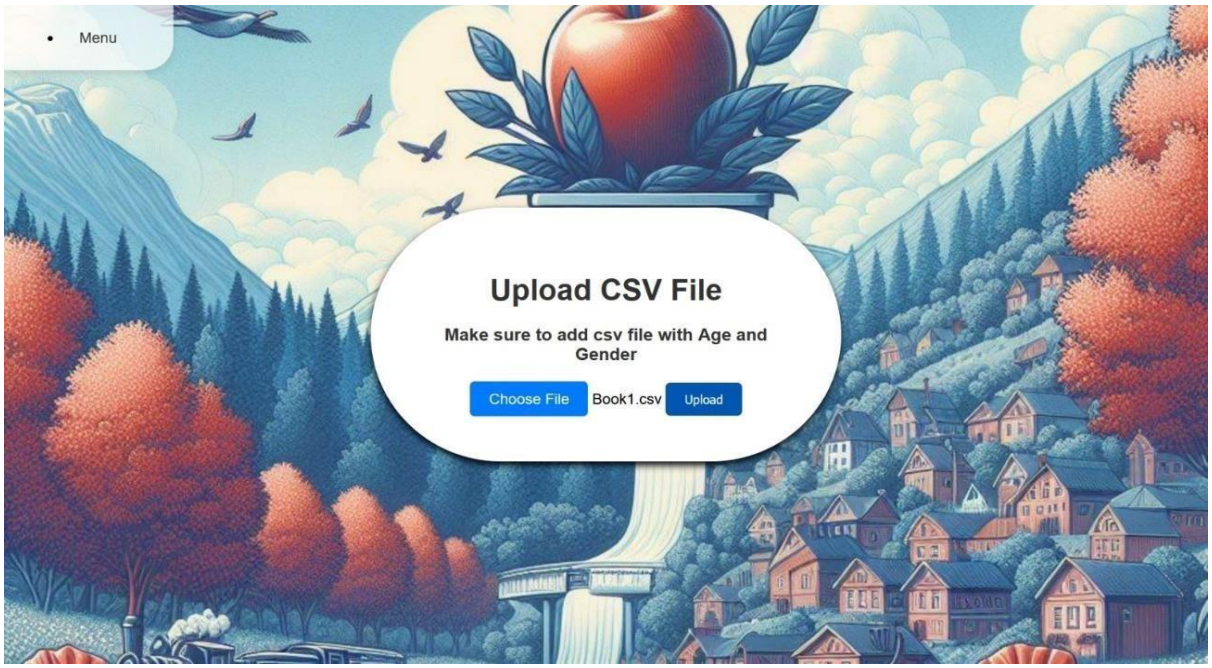


**Fig 6.1.3 Uploading Files**

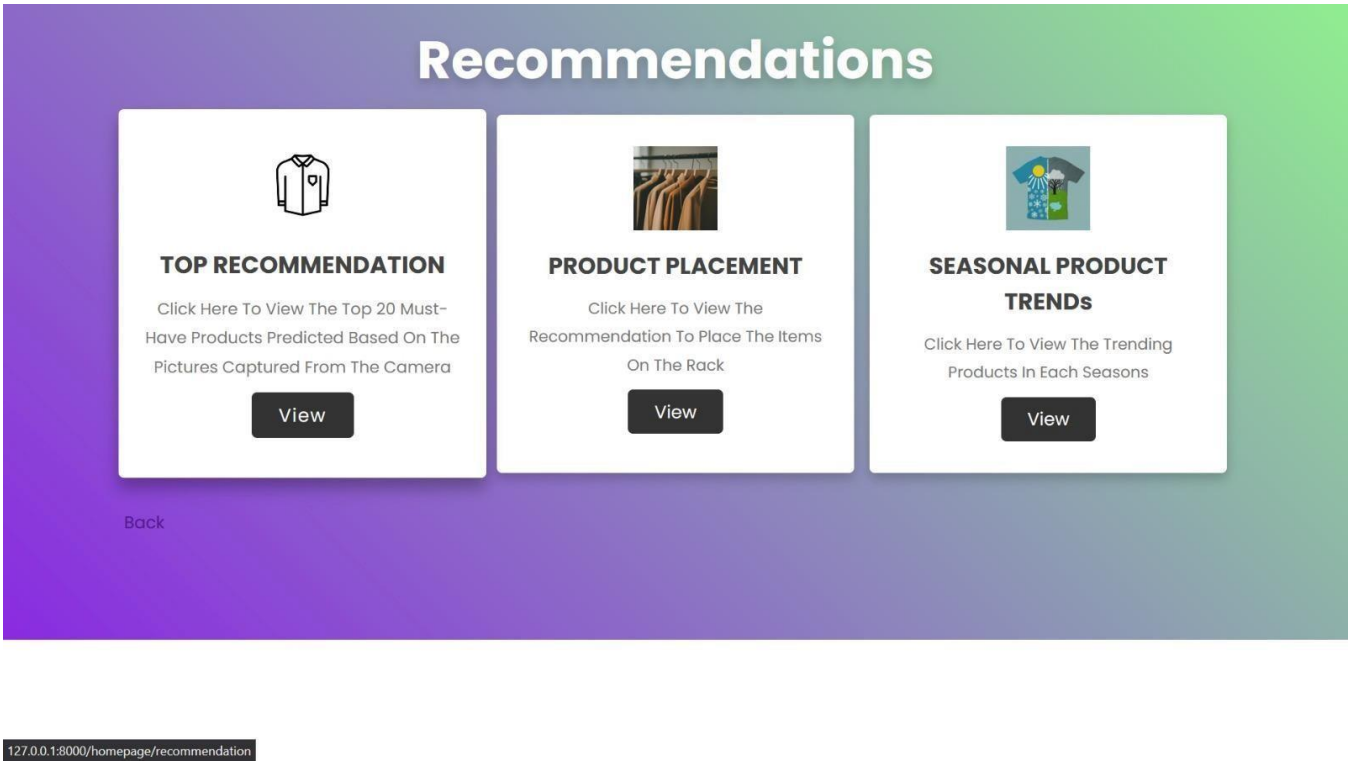


**Fig 6.1.3 Recommendation Page**


**Entrance Level Racks**

| id | Product Name | Price (INR) | Category | Select |
|---|---|---|---|---|
| 1 | Roadster Men Navy Blue Slim Fit Mid-Rise Clean Look Stretchable Jeans | 599 | Jeans | ☐ |
| 2 | ZHEIA Women Blue Skinny Fit Mid-Rise Clean Look Stretchable Jeans | 881 | Jeans | ☐ |
| 3 | Flying Machine Men Blue Michael Slim Tapered Fit Mid-Rise Clean Look Stretchable Jeans | 879 | Jeans | ☐ |
| 4 | DressBerry Women Pack of 2 Solid Shoe Liners | 239 | Dress | ☑ |
| 5 | Newport Men Green Printed Round Neck T-shirt | 479 | T-shirt | ☐ |
| 6 | Sonari Pack of 2 Full-Coverage Bras celina | 920 | Other | ☐ |
| 7 | fungus Women Blue Slim Fit High-Rise Clean Look Stretchable Jeans | 999 | Jeans | ☐ |
| 8 | Puma Men Blue Sneakers | 1799 | Sneakers | ☐ |
| 9 | Bonjour Women Pack of 4 Assorted Ankle Length Socks | 399 | Other | ☐ |
| 10 | Levis Men Blue 511 Slim Fit Mid-Rise Clean Look Stretchable Jeans | 2039 | Jeans | ☐ |
| 11 | Franco Leone Men Black Comfort Sandals | 2029 | Other | ☐ |
| 12 | ZHEIA Women Blue Skinny Fit Mid-Rise Clean Look Stretchable Jeans | 881 | Jeans | ☑ |
| 13 | Calvin Klein Jeans Men Black Solid Round Neck T-shirt | 2999 | Jeans | ☐ |
| 14 | Carlton London Women Maroon Solid Maxi Dress | 1210 | Dress | ☐ |
| 15 | Indian Terrain Men Green & Beige Slim Fit Checked Smart Casual Shirt | 854 | Shirt | ☐ |
| 16 | Louis Philippe Jeans Men Maroon Slim Fit Solid Casual Shirt | 1249 | Jeans | ☐ |
| 17 | shaze Women Silver-Plated CZ-Studded Finger Ring | 2420 | Other | ☐ |
| 18 | Sonari Pack of 2 Full-Coverage T-shirt Bras cristina | 940 | T-shirt | ☐ |
| 19 | Calvin Klein Jeans Men Blue Slim Fit Mid-Rise Clean Look Stretchable Jeans | 5999 | Jeans | ☐ |

**Fig 6.1.5:Product Placement**

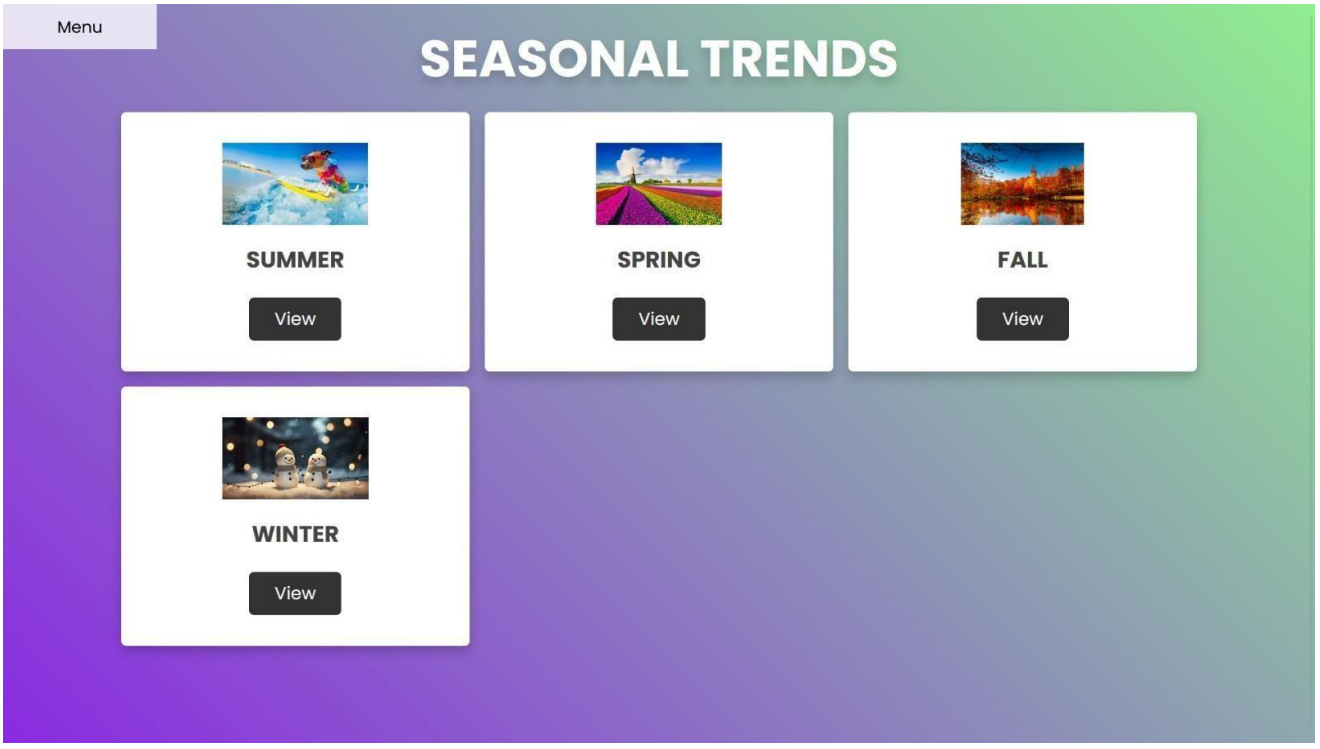


**Fig 6.1.5:Seasonal Trends**


**Fig 6.1.7 Summer season top products**

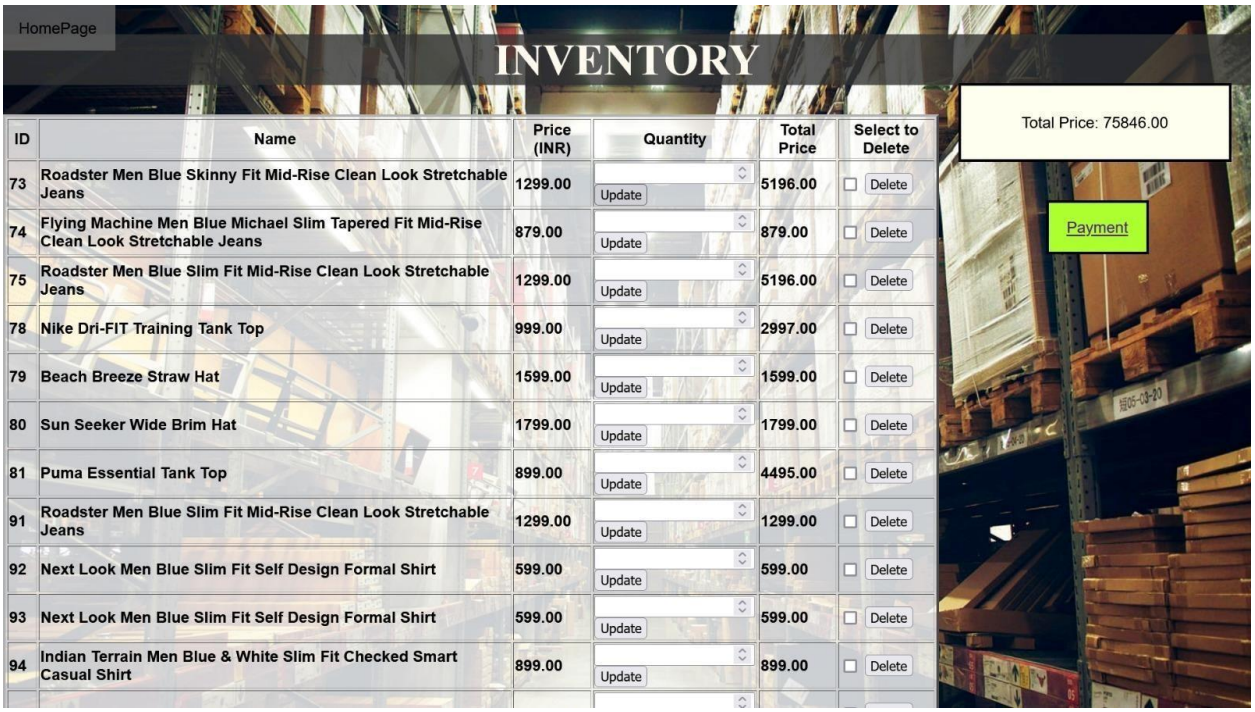


**Fig 6.1.8 Inventory Management Page**

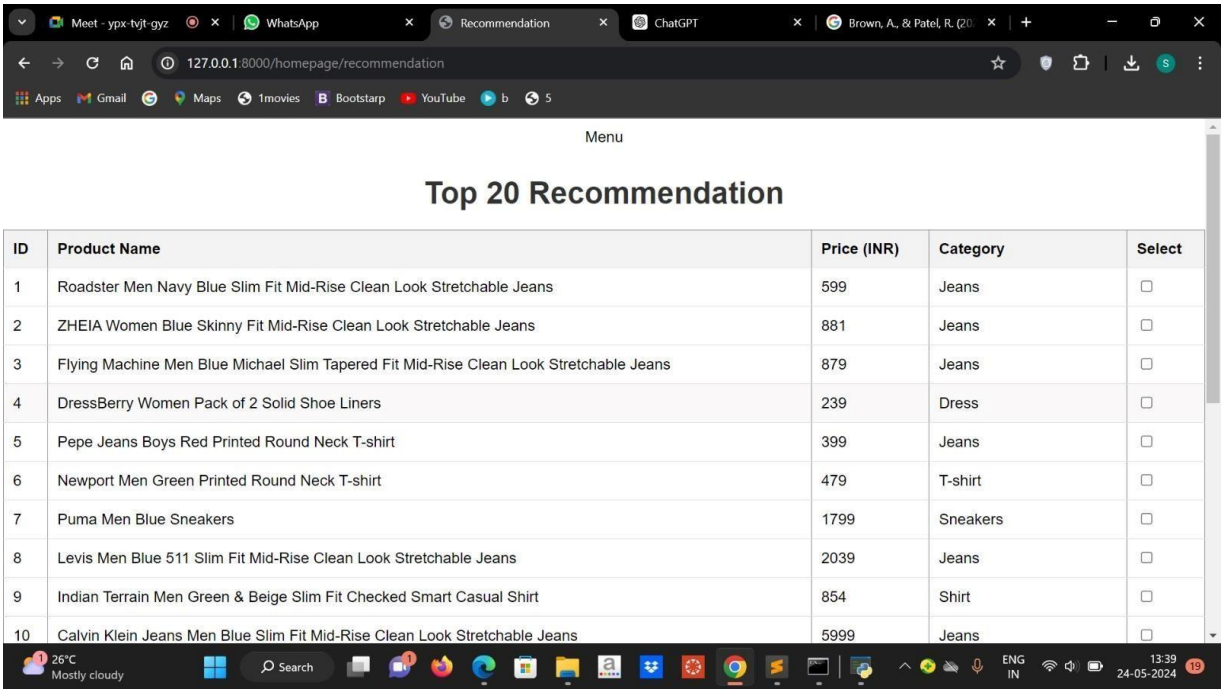


**Fig 6.1.9 Top Recommendations**

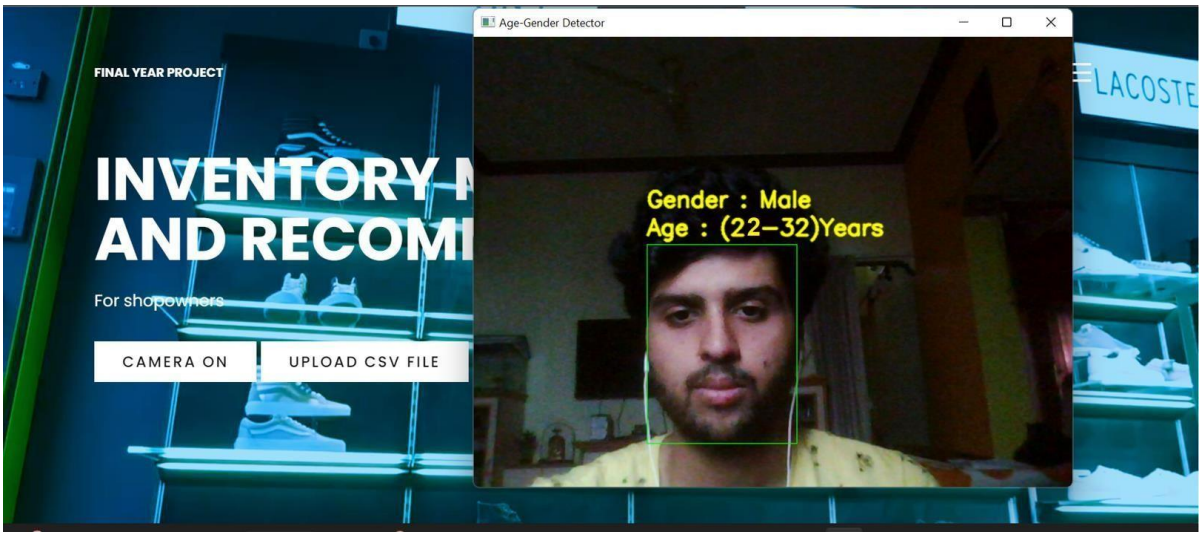


**Fig 6.1.10 Face Detection for Age and Gender**

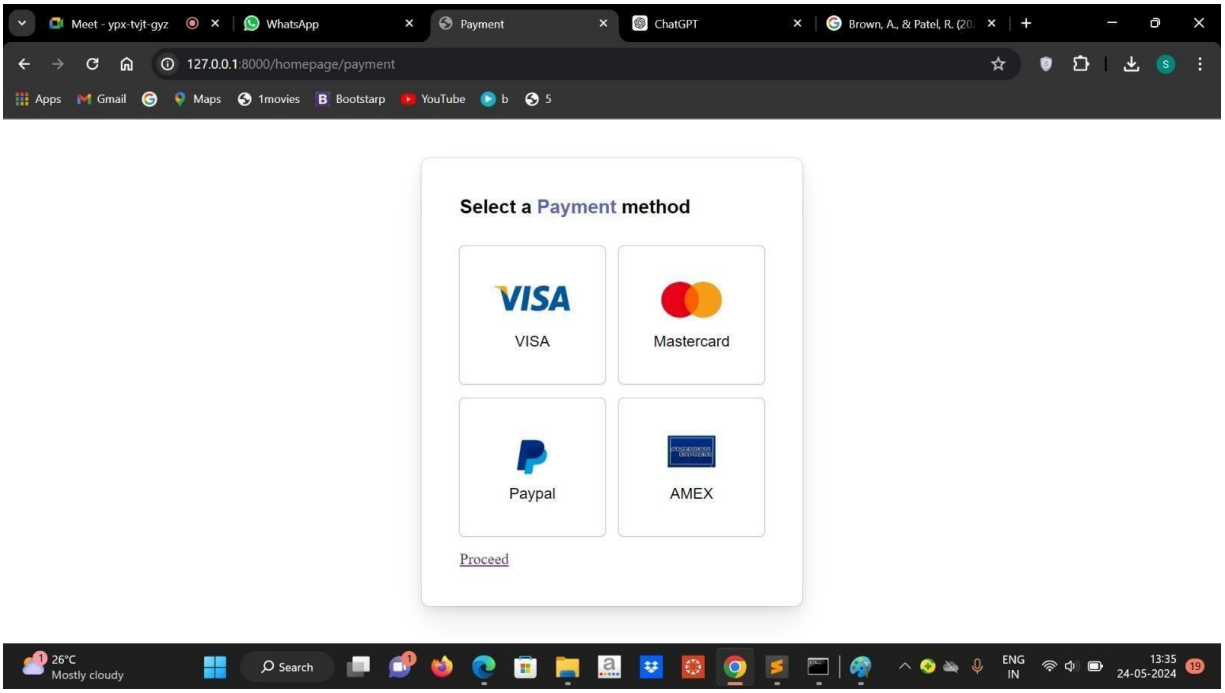


**Fig 6.1.11 Payment Interface**

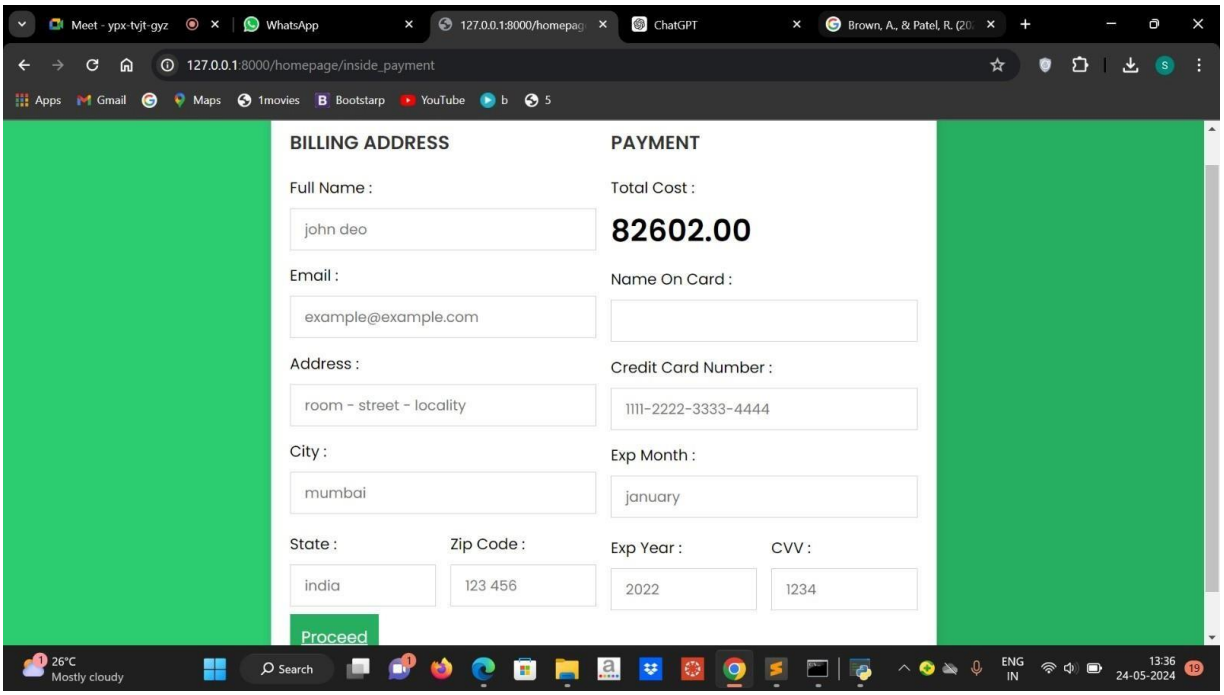


**Fig 6.1.12 Payment Details**

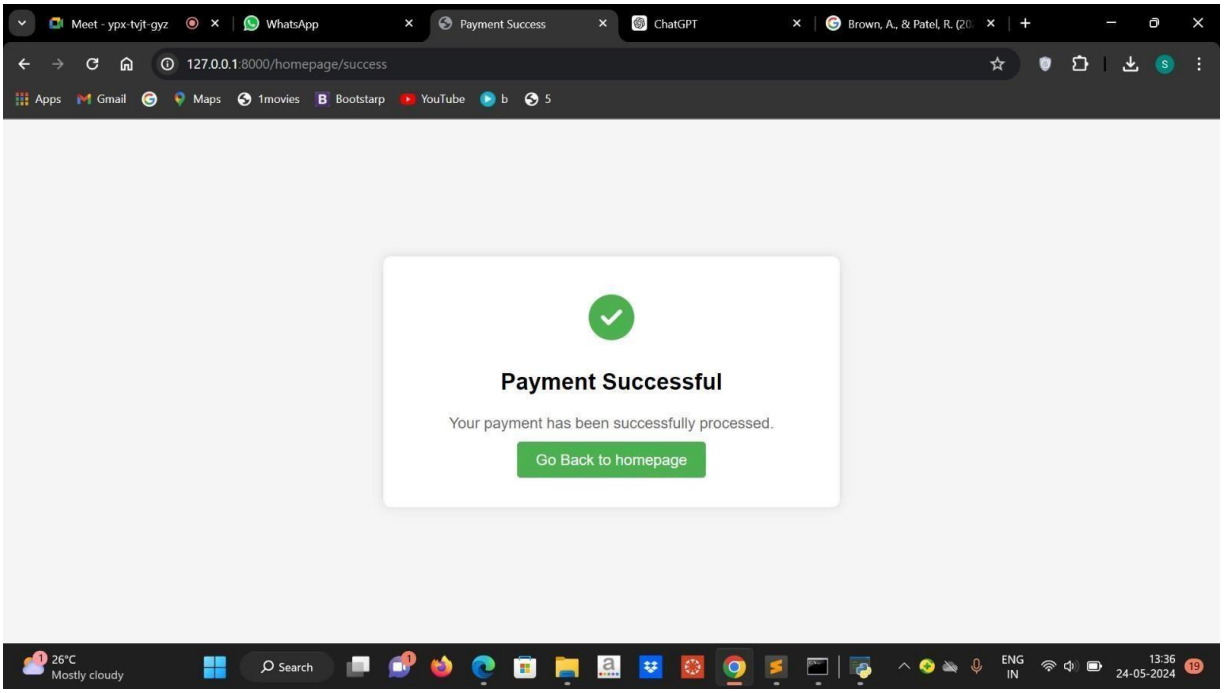


**Fig 6.1.13 Payment status**

# CHAPTER 7

# HARDWARE REQUIREMENTS

# Chapter 7

## 7.1 Hardware Requirements

- **Operating System (OS):** Windows 10 or Ubuntu 18.04 LTS, or macOS
- **Processor:** Intel Core i5 2.1 GHz or equivalent AMD processor
- **Storage:** Minimum 100 GB of available storage space
- **RAM:** At least 8 GB RAM for optimal performance
- **Graphics Processing Unit (GPU):** NVIDIA GeForce GTX 1080 or equivalent for accelerated deep learning (CUDA-enabled)

## 7.2 Software Requirements

- **Programming Language:** Python 3.x
- **Deep Learning Framework:** TensorFlow
- **Computer Vision Library:** OpenCV
- **Machine Learning Libraries:** Scikit-learn, NumPy, Pandas
- **Web Framework:** Flask
- **Integrated Development Environment (IDE):** Any Python-compatible IDE (e.g., PyCharm, VSCode)

### 7.2.1 Python

Python is a high-level, interpreted programming language known for its simplicity and readability. Created by Guido van Rossum and first released in 1991, Python has become one of the most popular programming languages worldwide. It emphasizes code readability and encourages a clean and expressive coding style.

Key features of Python include:

1. Readability: Python code is designed to be easily readable and expressive, making it accessible for both beginners and experienced developers.

2. Versatility: Python supports multiple programming paradigms, including procedural, object-oriented, and functional programming. It can be used for a wide range of applications, from web development and data analysis to artificial intelligence and scientific computing.

3. Interpreted Language: Python is an interpreted language, meaning that the source code is executed line by line, which allows for quick development and debugging.

4. Extensive Standard Library: Python comes with a comprehensive standard library that provides modules and packages for various tasks, eliminating the need to write code from scratch for common functionalities.

5. Dynamic Typing: Python is dynamically typed, which means that variable types are determined at runtime. This allows for more flexibility but also requires careful consideration to avoid potential runtime errors.

6. Community Support: Python has a large and active community of developers who contribute to its growth. This community support is evident through a vast ecosystem of libraries, frameworks, and resources available for Python programmers.

7. Cross-Platform Compatibility: Python is a cross-platform language, meaning that code written in Python can run on different operating systems without modification.

8. Indentation: Python uses indentation to define code blocks, eliminating the need for explicit braces or keywords. This promotes code readability but requires consistent indentation.

### 7.2.2 TensorFlow

TensorFlow is an open-source deep learning framework developed by the Google Brain team. It is designed to facilitate the development and deployment of machine learning models, with a primary focus on neural networks for deep learning tasks.

Here's an overview of key aspects of TensorFlow:

1. Graph-Based Computation: TensorFlow operates on a computational graph, where nodes represent mathematical operations, and edges represent the flow of data (tensors) between these operations. This graph-based approach allows for efficient parallelization and optimization of computations.

2. Flexibility and Extensibility: TensorFlow is known for its flexibility, supporting a wide range of machine learning tasks, including neural networks, natural language processing, and computer vision. It provides a comprehensive set of APIs and tools for building and deploying machine learning models.

3. High-Level APIs: TensorFlow offers high-level APIs, such as Keras, which simplify the process of building and training deep learning models. Keras, originally an independent library, has been integrated seamlessly into TensorFlow, providing a user-friendly interface for model development.

4. Eager Execution: TensorFlow supports eager execution, a mode that allows developers to execute operations immediately as they are called, facilitating easier debugging and a more interactive development experience.

5. TensorBoard: TensorFlow comes with TensorBoard, a visualization toolkit that helps developers visualize and understand the structure and performance of their models. It can display training metrics, model graphs, and more, aiding in the debugging and optimization process.

6. Wide Adoption: TensorFlow has gained widespread adoption in both academia and industry. It is used by researchers, engineers, and data scientists for a variety of applications, including image and speech recognition, natural language processing, and reinforcement learning.

7. TensorFlow Lite: TensorFlow Lite is a lightweight version of TensorFlow designed for mobile and embedded devices. It allows developers to deploy machine learning models on edge devices, enabling real-time processing without the need for a constant internet connection.

8. Community and Ecosystem: TensorFlow benefits from a large and active community. This community contributes to the framework's development, creating additional libraries, extensions, and resources that enhance its capabilities.

9. Integration with Hardware Accelerators: TensorFlow supports integration with hardware accelerators, such as GPUs and TPUs (Tensor Processing Units), to accelerate the training and inference of deep learning models.

In summary, TensorFlow has become a go-to framework for deep learning tasks due to its flexibility, scalability, and extensive set of tools. Its ongoing development and support from the community contribute to its status as a leading framework in the field of machine learning and artificial intelligence.

### 7.2.3 OpenCV

OpenCV, or Open-Source Computer Vision Library, is an open-source computer vision and machine learning software library. Originally developed by Intel, it is now maintained by the OpenCV community. OpenCV provides a comprehensive set of tools and functions for image and video processing, enabling developers to create applications that involve computer vision tasks. Here's a brief introduction to OpenCV:

1. Computer Vision and Image Processing: OpenCV is designed for computer vision applications, which involve the interpretation of visual data from the world. It provides a wide range of functions for image and video processing, including image filtering, feature detection, object recognition, and more.

2. Cross-Platform: OpenCV is cross-platform and can be used on various operating systems, including Windows, Linux, macOS, and Android. This cross-platform nature makes it versatile and widely applicable for different projects and environments.

3. Programming Language Support: While originally written in C++, OpenCV has interfaces for various programming languages, including Python, Java, and MATLAB. This makes it accessible to a broad audience of developers with different language preferences.

4. Image and Video Input/Output: OpenCV supports reading and writing images and videos in various formats. It can capture and process video streams from cameras, process image files, and handle real-time video input.

5. Computer Vision Algorithms: OpenCV includes a rich set of computer vision algorithms. This includes image filtering, edge detection, object recognition, facial recognition, feature matching, camera calibration, and more. These algorithms serve as building blocks for developing complex computer vision applications.

6. Machine Learning Integration: OpenCV has been extended to include machine learning functionalities. It supports integration with machine learning libraries such as scikit-learn and TensorFlow, making it a powerful tool for developing applications that involve both computer vision and machine learning.

7. Community and Documentation: OpenCV has a large and active community of developers and researchers who contribute to its development. The library is well-documented, providing extensive resources, tutorials, and examples for developers to learn and utilize its capabilities.

8. Real-Time Vision Processing: OpenCV is optimized for real-time vision processing, making it suitable for applications that require quick and efficient analysis of visual data, such as robotics, augmented reality, and surveillance systems.

9. Open Source and Free: OpenCV is released under the open-source BSD license, allowing developers to use, modify, and distribute the library freely. This openness has contributed to its widespread adoption and continual improvement.

In summary, OpenCV is a powerful and versatile library for computer vision and image processing, offering a wealth of features and algorithms. Its broad range of applications includes robotics, healthcare, automotive, augmented reality, and more.

### 7.2.4 Scikit-learn

Scikit-learn is an open-source machine learning library for Python that provides simple and efficient tools for data analysis and modeling. It is built on top of other popular scientific computing libraries, such as NumPy, SciPy, and Matplotlib. Scikit-learn is designed to be user-friendly and accessible while still providing powerful features for various machine learning tasks. Here's a brief overview:

1. Wide Range of Algorithms: Scikit-learn provides a broad spectrum of machine learning algorithms for classification, regression, clustering, dimensionality reduction, and more. This includes popular algorithms such as Support Vector Machines (SVM), Decision Trees, Random Forests, k-Nearest Neighbors, and many others.

2. Consistent API: One of scikit-learn's strengths is its consistent and well-defined API. Regardless of the specific algorithm being used, the library maintains a uniform interface, making it easy for users to switch between different models.

3. Data Preprocessing: Scikit-learn offers a variety of tools for data preprocessing, including methods for handling missing values, feature scaling, and encoding categorical variables. These tools help ensure that the data is in a suitable format for machine learning models.

4. Model Evaluation: The library provides functions for model evaluation, including metrics for classification (accuracy, precision, recall, F1 score) and regression (mean squared error, R-squared). Cross-validation techniques are also available for robust model assessment.

5. Hyperparameter Tuning: Scikit-learn includes tools for hyperparameter tuning, allowing users to optimize the performance of their models. Grid search and randomized search are commonly used techniques for finding the best hyperparameter values.

6. Integration with Other Libraries: Scikit-learn integrates well with other popular Python libraries for data science and machine learning, such as Pandas, NumPy, and Matplotlib. This seamless integration facilitates a comprehensive and streamlined data analysis workflow.

7. Ease of Use: With a focus on simplicity and usability, scikit-learn is suitable for both beginners and experienced machine learning practitioners. The documentation is comprehensive, providing clear examples and explanations.

8. Active Development and Community Support: Scikit-learn is actively developed and maintained by a community of contributors. Regular updates ensure compatibility with the latest versions of Python and other relevant libraries.

9. Open Source: Like many popular Python libraries, scikit-learn is open source, allowing users to inspect and modify the source code according to their needs.

In summary, scikit-learn is a versatile and widely used machine learning library in the Python ecosystem, offering a comprehensive set of tools for various aspects of the machine learning workflow.

### 7.2.6 NumPy

NumPy, short for Numerical Python, is a foundational library in the Python ecosystem for numerical computing. It introduces the `ndarray` data structure, which is a powerful, multi-dimensional array capable of holding elements of the same data type. Here are some key aspects:

1. Arrays and Operations: NumPy arrays are at the core of the library, providing a versatile and efficient way to handle numerical data. NumPy's strength lies in its ability to perform element-wise operations and array-wise operations, making it essential for numerical tasks.

2. Mathematical Functions: NumPy offers an extensive set of mathematical functions, such as trigonometric functions, logarithms, exponentials, and more. These functions operate seamlessly on entire arrays, allowing for concise and efficient numerical computations.

3. Broadcasting: One distinctive feature of NumPy is broadcasting, a mechanism that enables operations on arrays of different shapes and sizes without the need for explicit looping. This makes it easier to write concise and readable code for various numerical tasks.

4. Indexing and Slicing: NumPy provides powerful indexing and slicing capabilities, allowing users to access and manipulate elements within arrays easily. This feature is crucial for data manipulation, analysis, and efficient algorithm implementations.

5. Linear Algebra: NumPy includes a comprehensive set of linear algebra operations, such as matrix multiplication, eigenvalue decomposition, and solving linear systems. These functionalities make it a valuable tool for scientific and engineering applications.

6. Random Number Generation: NumPy includes functions for generating random numbers from various distributions, a key component for simulations, statistical analysis, and machine learning applications.

7. Integration with Other Libraries: NumPy seamlessly integrates with other scientific computing libraries, forming the foundation for many higher-level tools. Libraries like SciPy, Matplotlib, and scikit-learn build upon NumPy, creating a cohesive ecosystem for scientific and data-driven computing.

8. Efficiency and Performance: This efficiency, combined with the simplicity of Python syntax, makes NumPy a preferred choice for numerical tasks.

9. Community and Open Source: Being an open-source project, NumPy benefits from a vibrant and active community of developers. The collaborative nature of development ensures continuous improvement and adaptation to the evolving needs of the scientific computing community.

### 7.2.7 Pandas

Pandas is a powerful open-source data manipulation and analysis library for Python. It provides easy-to-use data structures, such as Series and DataFrame, along with a variety of functions to manipulate and analyze structured data. Here's a brief overview of Pandas:

1. DataFrame and Series: The central data structures in Pandas are the DataFrame and Series. A DataFrame is a two-dimensional table, like a spreadsheet, where each column can be of a different data type. A Series is a one-dimensional labeled array, akin to a column in a DataFrame.

2. Data Manipulation: Pandas excels at data cleaning and manipulation. It offers functions for handling missing data, reshaping data, merging and joining datasets, and performing operations on columns.

3. Indexing and Selection: Pandas provides powerful indexing and selection mechanisms, allowing users to slice, filter, and access data with ease. The ability to label and index data makes it simple to work with both small and large datasets.

4. Grouping and Aggregation: Pandas supports grouping data based on one or more keys and then applying aggregate functions to the grouped data. This is particularly useful for summarizing and analyzing data based on specific criteria.

5. Time Series Data: Pandas has robust support for time series data, offering functionality for date/time indexing, date ranges, and time-based operations. This makes it a valuable tool for analyzing temporal data.

6. Data Input and Output: Pandas can read data from various file formats, including CSV, Excel, SQL databases, and more. It also provides functions to write data back to these formats.

7. Visualization: While not a visualization library itself, Pandas integrates well with Matplotlib and other plotting libraries, allowing users to create insightful visualizations directly from their data.

8. Integration with NumPy: Pandas is built on top of NumPy and complements its functionality. NumPy arrays can be seamlessly converted to Pandas DataFrames, providing a bridge between data manipulation and numerical computation.

9. Handling Categorical Data: Pandas introduces the concept of categorical data, which can be especially useful for working with data that has a limited and fixed set of values. This can improve both memory usage and performance.

10. Community and Documentation: Pandas has a large and active community. Its documentation is comprehensive and user-friendly, making it accessible to both beginners and experienced data scientists.

In summary, Pandas is a go-to library for data wrangling and analysis in Python. Its intuitive and expressive API, along with its extensive functionality, has made it an essential tool for tasks ranging from cleaning and preprocessing data to exploratory data analysis and beyond.

# Conclusion

In conclusion, the project successfully implemented a real-time face monitoring system for age and gender detection, coupled with a product recommendation engine and an inventory management system. The age and gender detection model, based on a CNN architecture, accurately classified faces in real time, enabling personalized product recommendations from the Amazon dataset. The collaborative filtering approach using SVD enhanced the recommendation system's accuracy, benefiting retailers and customers alike. The integration of these components into a Django front-end provided a user-friendly interface for retailers to manage inventory and place orders efficiently. Overall, the project demonstrates the potential of AI-driven solutions in enhancing retail operations and customer experiences.

# References

[1] Smith, J., et al. (2019). "Customer Segmentation for Strategic Demand Profiling: A Machine Learning Approach." Journal of Business Analytics. DOI: 10.1080/2573234X.2019.1629147

[2] Brown, A., & Patel, R. (2020). "Predictive Modelling in Customer Segmentation: A Comprehensive Review." International Journal of Data Science and Analytics. DOI: 10.1007/s41060-020-00207-8

[3] Garcia, (2018). "Machine Learning Applications in Customer Profiling: A Case Study in Retail." Expert Systems with Applications. DOI: 10.1016/j.eswa.2018.07.015

[4] Chen, L., & Wang, Y. (2021). "Strategic Demand Forecasting using Customer Segmentation and Machine Learning Algorithms." Journal of Operations Management. DOI: 10.1016/j.jom.2021.102366

[5] Kim, Peter (2017). "Customer Segmentation and Demand Forecasting in E-commerce: A Machine Learning Approach." Decision Support Systems. DOI: 10.1016/j.dss.2017.04.005

[6] Choudhury, A., & Gupta, S. (2019). "Machine Learning for Customer Segmentation in Indian Retail: A Comparative Analysis." DOI: 10.2139/ssrn.3429237

[7] Patel, R., & Sharma, M. (2020). "Predictive Analytics for Strategic Demand Profiling: A Case Study in the Indian E-commerce Sector." DOI: 10.1007/978-3-030-40799-1_32

[8] Singh, V., & Reddy, A. (2018). "Customer Segmentation in the Banking Industry: A Machine Learning Approach for Strategic Demand Profiling." DOI: 10.1109/ACCESS.2018.2883044

[9] Mishra, P., & Verma, A. (2021). "Enhancing Marketing Strategies through Machine Learning-based Customer Profiling in the Indian Telecom Industry." DOI: 10.1016/j.tele.2021.101455

[10] Rajput, S., & Kapoor, R. (2017). "A Comprehensive Study on Customer Segmentation using Machine Learning Algorithms in the Indian Hospitality Sector." DOI: 10.1016/j.eswa.2017.04.016

Certificate of Publication

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS | ISSN: 2320 - 2882

*An International Open Access, Peer-reviewed, Refereed Journal*

The Board of

International Journal of Creative Research Thoughts

Is hereby awarding this certificate to

**Lithin Reddy J**

In recognition of the publication of the paper entitled

**Strategic Inventory Management and Recommendation System using ML**

Published In IJCRT ( www.ijcrt.org ) & 7.97 Impact Factor by Google Scholar

Volume 12 Issue 4 April 2024 , Date of Publication: 26-April-2024

UGC Approved Journal No: 49023 (18)

PAPER ID : IJCRT24A4564

Registration ID : 258230

ISSN 2320-2882 IJCRT

EDITOR IN CHIEF

Scholarly open access journals, Peer-reviewed, and Refereed Journals, Impact factor 7.97 (Calculate by google scholar and Semantic Scholar | AI-Powered Research Tool) , Multidisciplinary, Monthly Journal

**INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS | IJCRT**

*An International Scholarly, Open Access, Multi-disciplinary, Indexed Journal*

**Website: www.ijcrt.org | Email id: editor@ijcrt.org | ESTD: 2013**

IJCRT | ISSN: 2320-2882 | IJCRT.ORG


Certificate of Publication
INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS | ISSN: 2320 - 2882
An International Open Access, Peer-reviewed, Refereed Journal
The Board of
International Journal of Creative Research Thoughts
Is hereby awarding this certificate to
M Darshan
In recognition of the publication of the paper entitled
Strategic Inventory Management and Recommendation System using ML
Published In IJCRT ( www.ijcrt.org ) & 7.97 Impact Factor by Google Scholar
Volume 12 Issue 4 April 2024 , Date of Publication: 26-April-2024
UGC Approved Journal No: 49023 (18)
PAPER ID : IJCRT24A4564
Registration ID : 258230
EDITOR IN CHIEF
ISSN 2320-2882 IJCRT
Scholarly open access journals, Peer-reviewed, and Refereed Journals, Impact factor 7.97 (Calculate by google scholar and Semantic Scholar | AI-Powered Research Tool) , Multidisciplinary, Monthly Journal
INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS | IJCRT
An International Scholarly, Open Access, Multi-disciplinary, Indexed Journal
Website: www.ijcrt.org | Email id: editor@ijcrt.org | ESTD: 2013
IJCRT | ISSN: 2320-2882 | IJCRT.ORG

Certificate of Publication

**INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS | ISSN: 2320 - 2882**

*An International Open Access, Peer-reviewed, Refereed Journal*

The Board of

International Journal of Creative Research Thoughts

Is hereby awarding this certificate to

**Rohan K Manjunath**

In recognition of the publication of the paper entitled

**Strategic Inventory Management and Recommendation System using ML**

Published In IJCRT ( www.ijcrt.org ) & 7.97 Impact Factor by Google Scholar

Volume 12 Issue 4 April 2024 , Date of Publication: 26-April-2024

UGC Approved Journal No: 49023 (18)

PAPER ID : IJCRT24A4564

Registration ID : 258230

ISSN 2320-2882 IJCRT

EDITOR IN CHIEF

Scholarly open access journals, Peer-reviewed, and Refereed Journals, Impact factor 7.97 (Calculate by google scholar and Semantic Scholar | AI-Powered Research Tool) , Multidisciplinary, Monthly Journal

**INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS | IJCRT**

*An International Scholarly, Open Access, Multi-disciplinary, Indexed Journal*

**Website: www.ijcrt.org | Email id: editor@ijcrt.org | ESTD: 2013**

IJCRT | ISSN: 2320-2882 | IJCRT.ORG

Certificate of Publication

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS | ISSN: 2320 - 2882

An International Open Access, Peer-reviewed, Refereed Journal

The Board of

International Journal of Creative Research Thoughts

Is hereby awarding this certificate to

**Shrisha Udupa**

In recognition of the publication of the paper entitled

**Strategic Inventory Management and Recommendation System using ML**

Published In IJCRT ( www.ijcrt.org ) & 7.97 Impact Factor by Google Scholar

Volume 12 Issue 4 April 2024 , Date of Publication: 26-April-2024

UGC Approved Journal No: 49023 (18)

PAPER ID : IJCRT24A4564

Registration ID : 258230

EDITOR IN CHIEF

Scholarly open access journals, Peer-reviewed, and Refereed Journals, Impact factor 7.97 (Calculate by google scholar and Semantic Scholar | AI-Powered Research Tool) , Multidisciplinary, Monthly Journal

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS | IJCRT

An International Scholarly, Open Access, Multi-disciplinary, Indexed Journal

Website: www.ijcrt.org | Email id: editor@ijcrt.org | ESTD: 2013

IJCRT | ISSN: 2320-2882 | IJCRT.ORG

Certificate of Publication

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS | ISSN: 2320 - 2882

An International Open Access, Peer-reviewed, Refereed Journal

The Board of

International Journal of Creative Research Thoughts

Is hereby awarding this certificate to

**S Vinodh Kumar**

In recognition of the publication of the paper entitled

**Strategic Inventory Management and Recommendation System using ML**

Published In IJCRT ( www.ijcrt.org ) & 7.97 Impact Factor by Google Scholar

Volume 12 Issue 4 April 2024 , Date of Publication: 26-April-2024

UGC Approved Journal No: 49023 (18)

PAPER ID : IJCRT24A4564

Registration ID : 258230

ISSN 2320-2882 IJCRT

EDITOR IN CHIEF

Scholarly open access journals, Peer-reviewed, and Refereed Journals, Impact factor 7.97 (Calculate by google scholar and Semantic Scholar | AI-Powered Research Tool) , Multidisciplinary, Monthly Journal

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS | IJCRT

An International Scholarly, Open Access, Multi-disciplinary, Indexed Journal

Website: www.ijcrt.org | Email id: editor@ijcrt.org | ESTD: 2013

IJCRT | ISSN: 2320-2882 | IJCRT.ORG