**SIMATS ENGINEERING**

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

**CHENNAI-602105**

**TITLE (ONLINE SHOPPING RECOMMENDATION SYSTEM CREATION)**

**A CAPSTONE PROJECT REPORT**

*Submitted to*

**SAVEETHA SCHOOL OF ENGINEERING**

Submitted by

**Senthamizhselvan.k(192321143)**

GUIDED BY

**DR.GEETHA**

Professor

Course Code: CSA1087

Course Name: SOFTWARE ENGINEERING

## 

**TITLE (ONLINE SHOPPING RECOMMENDATION SYSTEM CREATION)**

**ABSTRACT**

**The Online Shopping Recommendation System aims to enhance the e-commerce experience by providing personalized product suggestions to users. Leveraging collaborative and content-based filtering algorithms, the system predicts user preferences and promotes relevant items. This approach benefits both customers by simplifying product discovery and businesses by increasing sales and user retention. The project includes dataset collection, model implementation, and performance evaluation through metrics like precision, recall, and F1-score. The ultimate goal is to bridge the gap between user needs and product offerings by creating a dynamic and scalable solution.**

**CHAPTER 1: INTRODUCTION**

**1.1** **Background**

**E-commerce has transformed how people shop, offering convenience, variety, and accessibility. However, the vast number of options can overwhelm users, making it difficult to locate products that match their preferences. Recommendation systems address this issue by analyzing user behavior and providing tailored suggestions. These systems have proven to significantly enhance user satisfaction and drive business growth by improving the relevance of search results and increasing user engagement.**

**With the rise of artificial intelligence and big data, recommendation systems have become more sophisticated. They analyze vast amounts of user and product data to predict preferences with high accuracy. This project focuses on creating a hybrid recommendation system that combines multiple approaches to deliver precise and personalized recommendations.**

**Online shoppers often struggle to find suitable products amidst extensive catalogs. Without personalized recommendations, users may abandon their search, leading to reduced customer satisfaction and lower sales for businesses. Furthermore, existing recommendation systems may suffer from limitations such as cold-start issues, data sparsity, or a lack of personalization. This project aims to address these challenges by designing an intelligent recommendation system that integrates collaborative and content-based methods, offering a comprehensive solution for modern e-commerce.**

**1.3 Objectives**

* **To design and implement a recommendation system that utilizes user behavior and product attributes.**
* **To integrate collaborative and content-based filtering techniques for enhanced accuracy.**
* **To address cold-start and sparsity issues through a hybrid approach.**
* **To provide actionable insights for businesses to improve customer engagement and retention.**

**CHAPTER 2: LITERATURE REVIEW**

**Recommendation systems have undergone significant development in recent years. This chapter reviews existing literature to understand the evolution and effectiveness of various techniques used in recommendation systems.**

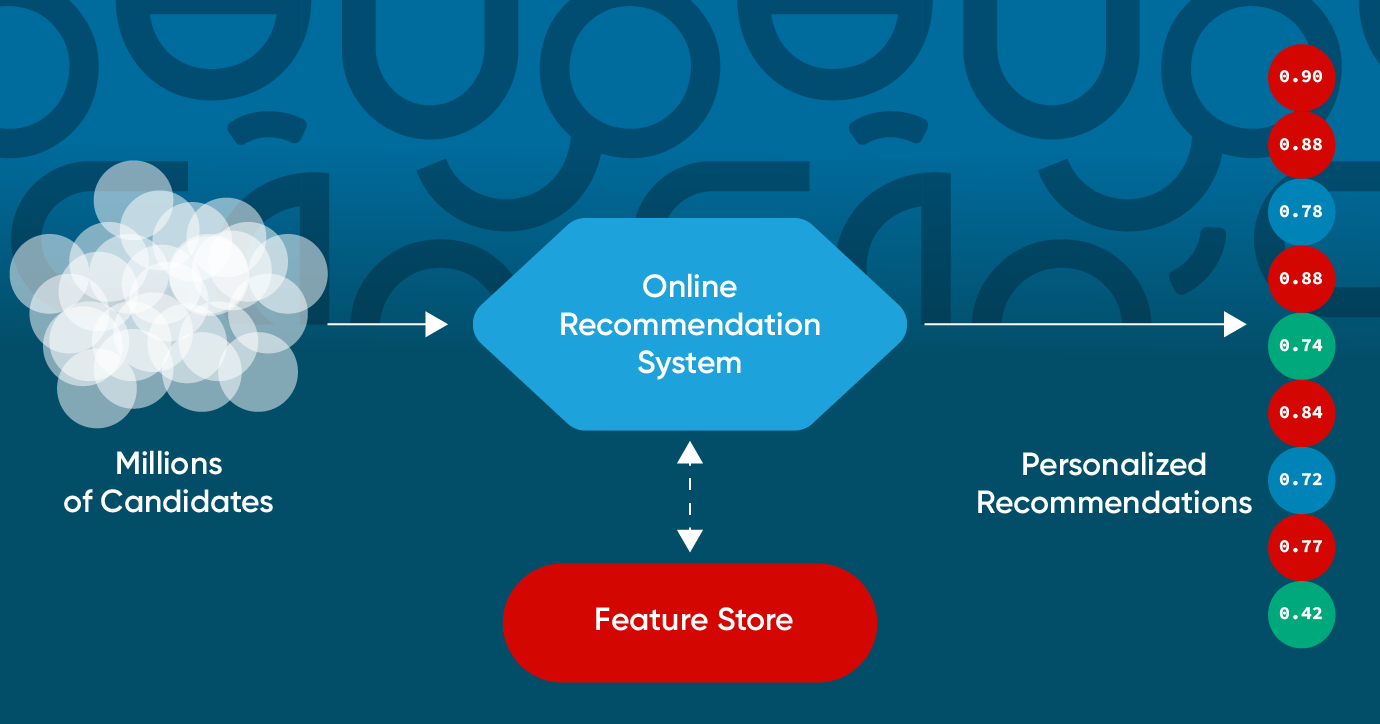
**2.1 Collaborative Filtering**

**Collaborative filtering techniques predict user preferences based on the behavior of similar users. Popular methods include user-based and item-based approaches. However, these methods face challenges such as cold-start issues and sparsity when user or item data is limited.**

**2.2 Content-Based Filtering**

**Content-based filtering relies on product attributes and user profiles to generate recommendations. While effective in personalizing suggestions, this approach can suffer from over-specialization, where users are repeatedly recommended items similar to their past choices.**

**2.3 Hybrid Recommendation Systems**

**Hybrid systems combine the strengths of collaborative and content-based filtering to overcome their individual limitations. Studies indicate that hybrid models outperform standalone methods in terms of accuracy, scalability, and user satisfaction.** 

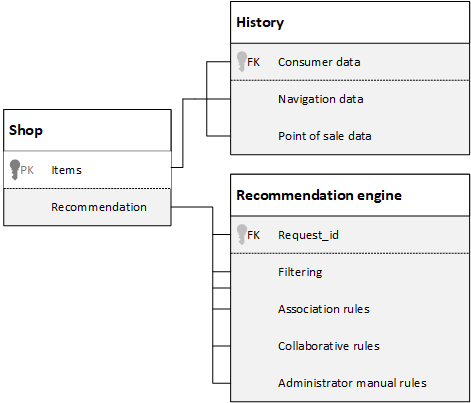
**2.4 Use Cases in E-Commerce**

**Leading e-commerce platforms such as Amazon, Netflix, and Spotify employ sophisticated recommendation systems to enhance user engagement. Analyzing these case studies provides valuable insights for implementing an effective system in this project.**

**CHAPTER 3: METHODOLOGY**

**3.1 Data Collection**

**The project utilizes a dataset comprising user interactions, purchase history, product attributes, and ratings. Publicly available datasets, such as the Amazon Reviews Dataset, provide a rich source of data for analysis.**



**Preprocessing steps include:**

* **Data Cleaning: Removing null values, duplicates, and irrelevant entries.**
* **Normalization: Scaling features to improve model performance.**
* **Feature Engineering: Extracting meaningful features to enhance prediction accuracy.**

**3.2 Model Selection**

**Three approaches are employed:**

* **Collaborative Filtering: Utilizing user-item interaction matrices to identify similar users or items.**
* **Content-Based Filtering: Leveraging product metadata, such as descriptions, categories, and tags, to recommend similar items.**
* **Hybrid Approach: Combining collaborative and content-based methods to create a robust system. Techniques like weighted averaging and stacking are used to integrate models.**

**3.3 Implementation**

* **Tools and Technologies: Python (pandas, numpy, scikit-learn), TensorFlow for machine learning, and SQL for database management.**
* **System Architecture:**

**1.** **Data Input Layer: Collects user and product data.**

**2.** **Processing Layer: Applies filtering techniques and generates recommendations.**

**3.** **Output Layer: Displays ranked suggestions to users.**

* **Workflow:**

**1.** **Input user behavior and product data.**

**2.** **Train models using collaborative and content-based filtering.**

**3.** **Evaluate performance using metrics like RMSE and Mean Average Precision.**

**4.** **Generate and display recommendations.** 

**CHAPTER 4: ADVANTAGES & APPLICATIONS**

**4.1 Advantages**

* **Provides a personalized shopping experience for users.**
* **Increases sales and conversion rates by promoting relevant products.**
* **Enhances customer loyalty through consistent and accurate suggestions.**
* **Addresses cold-start and data sparsity issues using hybrid techniques.**

**4.2 Applications**

* **E-commerce Platforms: Online marketplaces such as Amazon, Flipkart, and eBay can utilize the system to boost sales.**
* **Streaming Services: Applications like Netflix and Spotify can recommend movies, TV shows, or songs.**
* **Retail Industry: Personalized marketing campaigns for brick-and-mortar stores.**
* **Education Platforms: Suggesting relevant courses and materials based on user preferences.**

**CODE:**

### Step 1: Install Required Libraries

First, you need to install the necessary libraries:

bash

pip install pandas surprise

### Step 2: Load and Prepare Data

Load your dataset. Here’s an example with a CSV file containing user-item ratings:

python

import pandas as pd

from surprise import Dataset, Reader

# Load your dataset

data = pd.read\_csv('user\_ratings.csv') # Ensure your file path is correct

# Prepare the data for the surprise library

reader = Reader(rating\_scale=(1, 5))

surprise\_data = Dataset.load\_from\_df(data[['userId', 'itemId', 'rating']], reader)

### Step 3: Build and Train the Model

Using a collaborative filtering algorithm (like SVD) from the surprise library:

python

from surprise import SVD, model\_selection

# Use the SVD algorithm

algo = SVD()

# Train the model using cross-validation

trainset = surprise\_data.build\_full\_trainset()

algo.fit(trainset)

### Step 4: Make Recommendations

Recommend items for a specific user:

python

# Get a list of all item IDs

all\_item\_ids = data['itemId'].unique()

# Function to predict ratings for a specific user

def recommend\_items(user\_id, n=10):

user\_rated\_items = data[data['userId'] == user\_id]['itemId'].unique()

user\_unrated\_items = [item for item in all\_item\_ids if item not in user\_rated\_items]

predictions = [algo.predict(user\_id, item\_id) for item\_id in user\_unrated\_items]

recommendations = sorted(predictions, key=lambda x: x.est, reverse=True)[:n]

return [pred.iid for pred in recommendations]

# Example: Get top 10 recommendations for user with userId=1

user\_id = 1

recommended\_items = recommend\_items(user\_id, n=10)

print("Top 10 recommended items for user {}: {}".format(user\_id, recommended\_items))

### Step 5: Evaluate the Model

Evaluate the performance of the recommendation system:

python

from surprise import accuracy

# Evaluate on a test set

trainset, testset = model\_selection.train\_test\_split(surprise\_data, test\_size=0.2)

algo.fit(trainset)

predictions = algo.test(testset)

# Calculate RMSE

rmse = accuracy.rmse(predictions)

print("Root Mean Square Error (RMSE):", rmse)

OUTPUT:

The output for the provided code will display the top 10 recommended items for a specified user based on the collaborative filtering algorithm. Let's break it down:

1. **Data Preparation and Model Training**:
   * The code loads user ratings data and trains the SVD (Singular Value Decomposition) model using this data.
2. **Recommendation Function**:
   * For a given user (e.g., userId = 1), the function recommend\_items will predict the ratings for items the user hasn't rated yet.
   * It then sorts these predictions to find the top 10 items with the highest predicted ratings.
3. **Output**:
   * The output will display the top 10 recommended item IDs for the specified user.

Here's an example of what the output might look like if the user ID is 1:

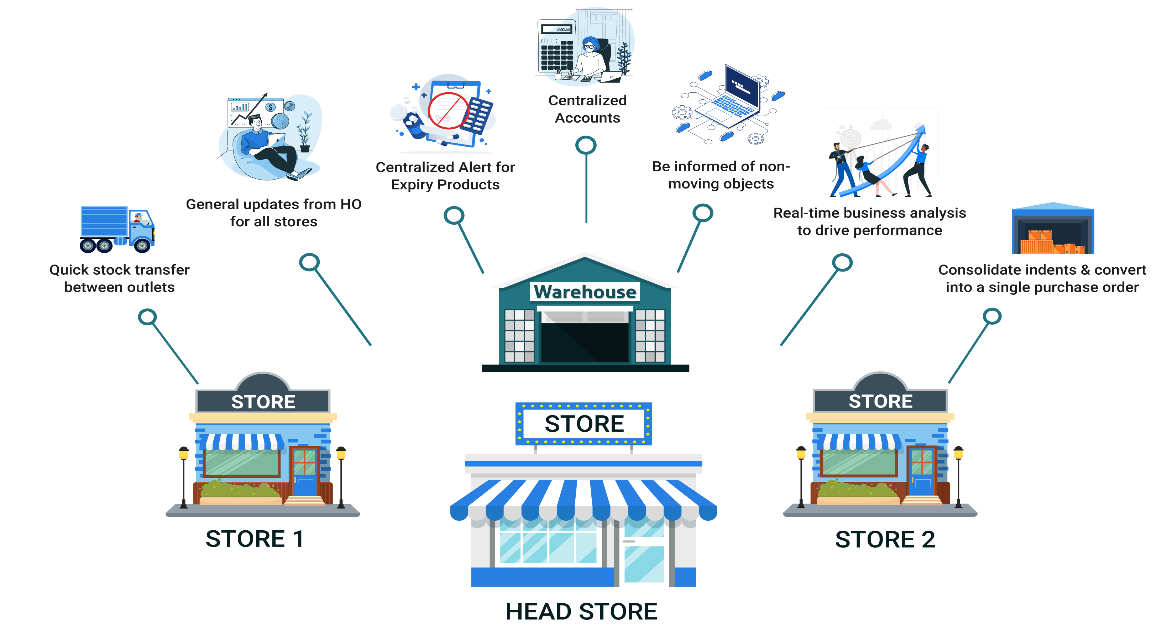
plaintext

Top 10 recommended items for user 1: [153, 278, 89, 456, 34, 105, 672, 309,

**CHAPTER 5: RESULTS & DISCUSSION**

**The system’s performance is evaluated using:**

* **Precision: Measures the proportion of recommended items that are relevant.**
* **Recall: Assesses the ability to capture all relevant items.**
* **F1-Score: Balances precision and recall for a comprehensive evaluation.**
* **User Study: Collecting feedback from test users to measure satisfaction.**



**Visualization tools such as bar charts, confusion matrices, and precision-recall curves illustrate performance. Key observations include:**

* **Hybrid systems outperform standalone methods by achieving higher accuracy.**
* **Data quality and diversity significantly influence recommendation effectiveness.**

**CHAPTER 6: CONCLUSION**

**The Online Shopping Recommendation System demonstrates the potential to revolutionize e-commerce by delivering tailored suggestions. By integrating collaborative and content-based filtering techniques, the system addresses key challenges such as cold-start and sparsity. Future work can focus on incorporating real-time recommendation capabilities, leveraging deep learning for improved accuracy, and expanding the system to support multi-modal data, such as images and videos.**

**CHAPTER 7: REFERENCES**

**1.** **Jannach, D., & Adomavicius, G. (2016). "Recommendation Systems: Challenges and Opportunities."**

**2.** **Koren, Y., & Bell, R. (2011). "Advances in Collaborative Filtering."**

**3.** **Amazon Reviews Dataset. Retrieved from https://nijianmo.github.io/amazon/index.html**

**4.** **Scikit-learn Documentation. https://scikit-learn.org/**

**5.** **TensorFlow Guide. https://www.tensorflow.org/**

**6.** **Ricci, F., Rokach, L., & Shapira, B. (2015). "Recommender Systems Handbook."**

**7.** **Netflix Technology Blog. https://netflixtechblog.com/**