UNIVERSITY OF MAURITIUS

FACULTY OF INFORMATION, COMMUNICATION AND DIGITAL TECHNOLOGIES

DEPARTMENT OF SOFTWARE AND INFORMATION SYSTEMS

AND

UNIVERSITY OF PARIS-SEINE

DIVING INTO DATA SCIENCE: A NOVEL APPROACH TO PERSONALISED GROCERY RECOMMENDATION SYSTEMS IN MAURITIUS

RONNISH YAANSH
RAJANAH
(2012054)

A THESIS SUBMITTED AS PART OF FULFILMENT
OF THE DUAL DEGREE PROGRAMME
BSC(HONS) DATA SCIENCE/COMPUTER SCIENCE

PROJECT SUPERVISOR: MR. SOMVEER KISHNAH

SUBMITTED ON: 28 JULY 2023

Contents

List of figures

List of tables

Acknowledgement

This achievement would not have been possible without the incredible support and contributions from many amazing individuals. I am deeply grateful for their unwavering belief in me and their invaluable assistance throughout this project.

First and foremost, my heartfelt appreciation goes to my parents and my sister for their support throughout this project. I thank my project supervisor, Mr. Somveer Kishnah, for his support, guidance, and advice for the duration of this project.

I extend my sincere gratitude to my friends, for their encouragement, stimulating discussions, and constructive feedback, which have played a significant role in refining this work. Furthermore, I want to express my gratitude to everyone who was willing to provide the necessary data to make this project possible.

A special thanks to the academic community and all the researchers whose groundbreaking word has laid the foundation for this study, inspiring me to explore new avenues of knowledge.

Declaration

On submission of my dissertation to the UoM, I solemnly declare that:

a) I have read and understood the sections on "Plagiarism and Fabrication and Falsification of Results" found in the University's Regulations Handbook (2020/2021) and certify that the dissertation embodies the results of my own work.

b) I have submitted a soft copy of my dissertation through the Turnitin Platform.

c) I have adhered to the "Harvard system of referencing" or a system acceptable as per "The University of Mauritius Referencing Guide" for referencing, quotations and citations in my dissertation. Each contribution to, and quotation and citations in my dissertation. Each contribution to, and quotation in my dissertation from the work of other people has been attributed, and has cited and referenced.

d) I have not allowed and will not allow anyone to copy my work with the intention of passing it off as his/her own work.

e) I am aware that I may have to forfeit the certificate/diploma/degree in the event that plagiarism has been detected after the award.

Notwithstanding the supervision provided to me by the University of Mauritius, I warrant that any alleged act(s) of plagiarism during my stay as a registered student of the University of Mauritius is entirely my own responsibility and the University of Mauritius and/or its employees shall under no circumstances whatsoever be under any liability of any kind in respect of the aforesaid act(s) of plagiarism.

Ronnish Yaansh Rajanah
28/07/2023

Abstract

This research presents the development and implementation of a novel grocery recommendation system aimed at improving the grocery shopping experience while providing grocery enterprises with an opportunity to upscale their business strategies.

The study begins with an overview of the problem, the objectives, and the scope. It delves into consumer buying patterns, the types of grocery shopping, and explores advanced recommendation algorithms like NMF, SVD, and NCF.

The recommendation system is trained and tested on a diverse dataset, consisting of both real-world data and fictitious data, to comprehensively evaluate its performance and effectiveness. The use of real-world data ensures the system's applicability in practical scenarios, while the inclusion of fictitious data serves as a controlled testing environment to assess its behaviour under various simulated conditions.

The system exhibits robust performance with real data and holds promise when dealing with larger datasets. The novel approach also showcased the ability to significantly enhance the machine learning algorithm's efficiency. Our investigation also revealed the distinct consumer buying behaviour patterns in the real-world dataset as the model displays vulnerability to noise. However, the presence of data sparsity adversely affects precision and sensitivity. The research also established that using implicit ratings, rather than explicit ratings, contributes to improved prediction accuracy.

List of Abbreviations

1.	SVD	Singular Value Decomposition
2.	NMF	Non-Negative Matrix Factorisation
3.	NCF	Neural Collaborative Filtering
4.	DNN	Deep Neural Network
5.	GMF	Generalised Matrix Factorisation
6.	MLP	Multi-Layer Perceptron
7.	NeuMF	Neural Matrix Factorisation
8.	CRISP-DM	Cross-Industry Standard Process for Data Mining
9.	OCR	Optical Character Recognition
10.	RFM	Recency, Frequency, Monetary
11.	CF	Collaborative Filtering
12.	RMSE	Root Mean Square Error
13.	MAE	Mean Absolute Error
14.	TP	True Positive
15.	TN	True Negative
16.	FP	False Positive
17.	FN	False Negative

Chapter breakdown

Chapter 1: Introduction

- This chapter presents an overview of the task ahead, the problem definition, the aims and objectives, as well as the scope of the project.

Chapter 2: Background Study

- This chapter dives into consumer buying patterns as well as types of grocery shopping. The chapter also covers several aspects of recommendation systems. Advanced algorithms are also discussed in the project, such as NMF, SVD, and NCF.

Chapter 3: Data Understanding, Collection, and Preparation

- The chapter talks about the data collection process, its preparation, and the validity of its application within the project.

Chapter 4: Analysis

- This segment conducts the analysis to gain insight about the data as well as tackle the cold-start problem. The requirements are defined, and comparisons between existing e-commerce platforms are made along with an examination of Mauritius top grocery franchises. Finally, a plan is devised to create a new and unique approach to building a recommendation system.

Chapter 5: Design

- This portion of the project provides a visual representation of the devised and proposed solution to implement the recommendation system.

Chapter 6: Implementation

- This chapter dives into the coding aspect of creating the recommendation system and provides explanations.

Chapter 7: Results, Evaluation, and Discussions

- This chapter presents the results and findings from implementing the recommendation system, together with an explanation.

Chapter 8: Testing

- This segment provides a visual depiction of when the recommender system is implemented in a simulated web application.

Chapter 9: Conclusion

- This chapter talks about the achievements accomplished throughout the project as well as the challenges, limitations, and proposals for future works.

Chapter 1: Introduction

1.1 Overview

Nowadays, businesses must adapt to remain competitive and satisfy the demands of their consumers. As more consumers turn to the comfort and ease of buying their groceries online, traditional grocery stores are struggling to match customers' expectations. Furthermore, the resulting impact of COVID-19 saw a shift in people making their grocery purchases online with the ease of reaching for their phones or laptops (Tyrväinen and Karjaluoto, 2022).

The increasing popularity of e-commerce websites and online purchasing facilities provides a much greater need for recommendation systems to assist customers (Isinkaye et al., 2015). Countries such as the United Kingdom, France, and the Netherlands that are leading in online grocery are still expected to grow. ("The next S-curve of growth: Online grocery to 2030 | McKinsey," 2022). The retail sector, being data-rich, provides a huge business opportunity. Consumer data such as purchase history and the customer's interests and preferences can be used to heavily influence the market.

Recommendation systems can be used to promote new or less popular items while creating long-term connections with clients and increasing corporate profitability. (Lü et al., 2012). Personalised recommendations account for more than 35% of orders on Amazon, the world's largest online retailer. Another notable example is in the world of entertainment streaming services. According to current statistics, tailored recommendations account for about 75% of what individuals watch on Netflix. ("How retailers can keep up with consumers | McKinsey," 2013). This demonstrates the efficacy of recommendation systems to influence consumer behaviour and purchase decisions. With their ability to properly propose material based on individual interests, recommendation algorithms have not only improved user satisfaction but also contributed to the phenomenal success of streaming services.

The application of recommendation systems in the grocery industry in Mauritius is at its nadir. The Mauritian grocery enterprises tend to focus more on the traditional aspect of grocery shopping, limiting their growth and customer comfort. The collection and analysis of a huge volume of data, as well as its variability and complexity, are some of the challenges present when implementing a recommendation system. Moreover, the recommendations need to be real-time and present a balance of interests for both the customers and the retailers. Other obstacles faced include the customers' privacy concerns, the development of the algorithm and its optimisation, the cold-start problem (lack of data), and the integration with the existing systems. (Lakshmi and Lakshmi, 2014).

1.2 Problem Statement

Companies must adapt to ever-changing consumer demands and trends to remain competitive and in business. In Mauritius, most grocery franchises rely mainly on conventional means, such as in-store promotions. However, these methods are restricted in their capacity to meet increasing consumer satisfaction.

To address this issue and allow businesses to remain competitive, a unique recommendation approach that utilises machine learning techniques is created to deliver individualised suggestions to clients with the aim of enhancing the customer experience and satisfaction while also improving the company's revenue. However, possible challenges would include:

- Since grocery enterprises are not very keen on sharing data with third parties.
- Surveys must be conducted.
- Transferring questionnaire or receipt data to Excel.

1.3 Aims and Objectives

The aim of this project is to create a novel recommendation system for a grocery store using machine learning algorithms to make personalised recommendations.

The following goals will be pursued to accomplish this aim:

1. It is important to review the current literature on recommendation systems to identify the various types available, their efficacy and implementation, as well as potential issues.
2. To make a recommendation system tailored to an industry, it is important to collect, preprocess, and learn about the data.
3. When designing and implementing a personalised grocery recommendation system, it is imperative to select the best-performing algorithms.
4. The proposed recommendation system must be evaluated in terms of RMSE, MAE, Accuracy, Precision, Sensitivity, and other relevant metrics.
5. It is important to recognize the full potential of the system as well as suggest future improvements and enhancements to it.

1.4 Scope of the Project

The scope of the project is limited to creating and implementing a grocery recommendation system for a Mauritian company. Personalised recommendations are made for customers based on their previous purchase habits and other important information. The project will centre on the implementation of machine learning algorithms and improving the approach. It will also include data gathering and analysis. The project, however, does not involve making a complete e-commerce platform.

1.5 Gantt Chart

	JAN	FEB	MAR	APR	MAY	JUN	JUL
Introduction	x						
Background Study		x	x	x			
Data Understanding, Collection, and Preparation	x	x	x	x			
Analysis				x	x		
Design				x	x		
Implementation					x	x	x
Results, Evaluation, and Discussion						x	x
Testing						x	x
Conclusion							x

Figure 1: Gantt Chart

Chapter 2: Background Study

2.1 Consumer Buying Pattern

While decision-making appears standardised, individuals vary in their approach. Analysing shopper trends enables consumer industry companies to stay updated with marketing and advertising strategies. By studying consistent customer data, organisations foster customer relationships and leverage this knowledge to fuel business growth.

In the US, online grocery sales are expected to grow at a steady pace of 6.5% from 2023 to 2027, as per Mercatus. Due to COVID-19 and inflation, there will be a continuous disagreement between cost and convenience that will impact customer behaviour. This would likely push retailers to adapt and innovate their businesses by renovating their customer experience and leveraging personalised promotions. ("US eGrocery Sales Predictions for Business Growth in 2023 and Beyond," 2023)

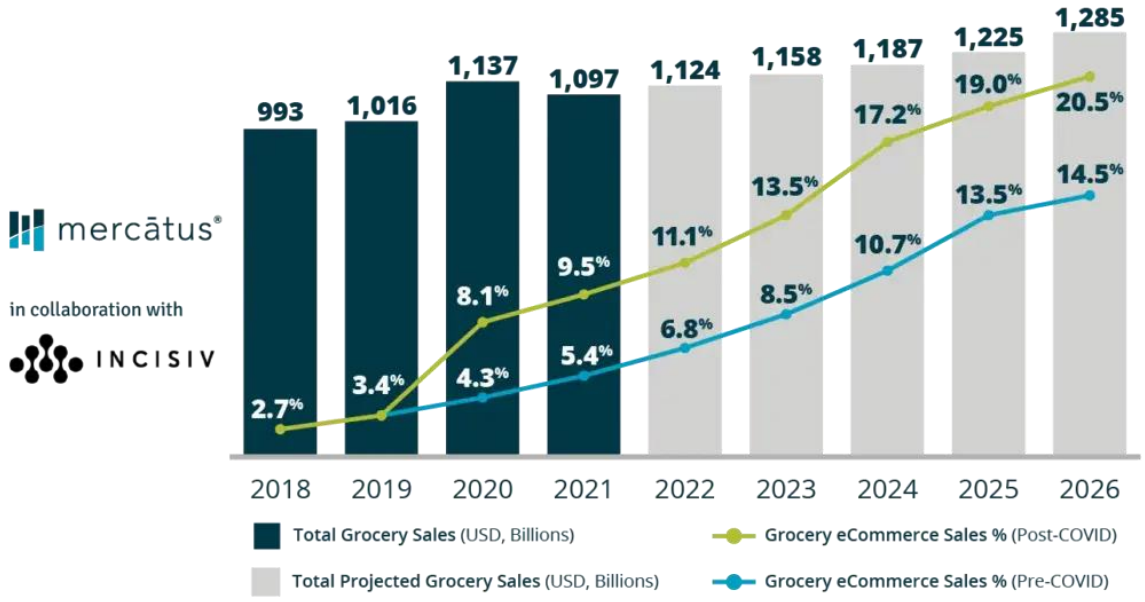


Figure 2: Mercatus Forecast

Consumer grocery shopping behaviour is influenced by various factors categorised as internal, external, and situational. These factors interact to influence consumer decisions. Understanding them is critical for supermarkets to generate successful marketing strategies. The figure below depicts the elements that influence customer product and service evaluations and purchase choices.

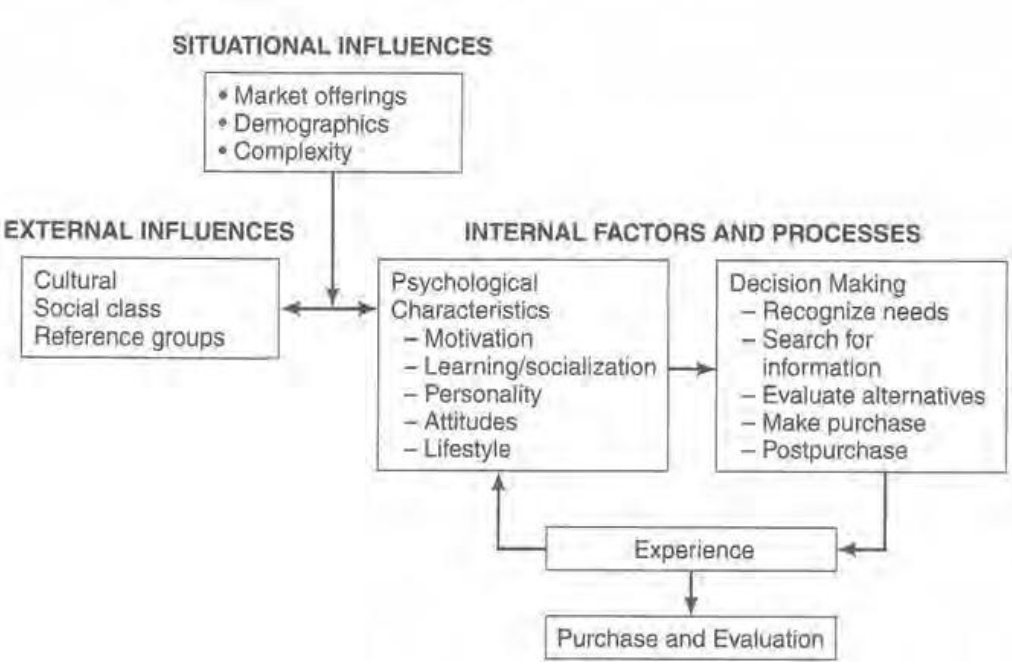


Figure 3: Consumer Behaviour Factors

2.1.1 Internal Factors

Examining internal variables influencing human decision-making is critical for understanding the consumer purchasing process. Personality: This research utilised a Pearson correlation analysis to examine the relationship between personal values and shopping styles. The results revealed a significant and positive correlation, indicating that consumers' shopping styles are influenced by their personal values. These findings seem to be consistent with previous studies (Helmi, 2016). Attitude: The study reveals that offering personalised products while maintaining price and quality has the potential to disrupt at least 50% of brand loyalty (Aichner and Coletti, 2013). Lifestyle: The analysis conducted by the authors revealed that utilitarian shopping motivation impacts the shopping habits of e-commerce shoppers. This suggests that online consumers who prioritise functional benefits are more inclined to adopt healthy shopping habits (Adaji et al., 2018).

2.1.2 External Factors

External factors have a significant influence on consumer purchasing decisions, impacting attitudes and behaviours. The following examples explore how some of these external factors affect consumer decision-making processes and choices in product or service purchases. Social Influence: Customers who engage with their peers in a shopping group are more likely to buy a greater quantity of products (Zhang et al., 2014a)**.** Economic conditions: The experiment's results regarding the selection of a primary store indicate that the most prevalent factor influencing this decision is price, accounting for 34.2% of the respondents' choices (Shier et al., 2022)**.**

2.1.3 Situational Factors

Situational factors refer to the specific conditions or circumstances that influence a person's decision-making process. For example, Budget constraints: As per the author's hypotheses, budget-constrained customers tend to create a spending safety margin to abstain from exceeding their budget, and the size of the safety margin positively correlates with a shopper's risk aversion regarding excess spending (Pennings et al., n.d.). Purchase Environment: According to this research, store layout is a significant factor that influences the shoppers sense of control over their shopping experiences in the stores, as stated by the interviewers: "I can go around very, very quickly. The things are set out very well too (Helen)" and "Sometimes if I want to get something I will telephone them and find out rather than have a look […] I let my fingers do the walking […] (Susan)". (Woodruffe-Burton and Wakenshaw, 2011). (Seasonal factors) The study conducted on ten women revealed that consumption occasions and situational factors intensify the challenges of grocery shopping, as one of the participants disclosed that during Christmas, she thinks of additional products that she would not normally purchase. (Woodruffe-Burton and Wakenshaw, 2011)

2.2 Approaches to Grocery Shopping

A survey in the US showed that 62% of 1,100 customers prefer to make in-store purchases, while 18% use delivery services. Additionally, 13% opt for pickup options, and the remaining 7% employ other methods for their purchases. ("Grocery Store Statistics: Where, When, & How Much People Grocery Shop," 2023)

2.3 Comparison of Online Grocery Shopping and Traditional Grocery Shopping

Traditional grocery shopping is mostly preferred by the older age groups, while the younger age groups tend to prefer online grocery shopping. There was a huge difference in terms of age, household types, and education level when comparing online and offline subgroups. (George Adamides et al., 2006). This reflects the fact that as the younger population matures, online grocery shoppers will become more significant.

According to Mintel's 2019 report, the online grocery sector is experiencing rapid growth in the UK grocery market, going from 6.1% in 2016 to 7% in 2018. Due to the expected and continuous increase in popularity of online grocery shopping, the revenue generated is anticipated to exceed £13.6 billion. It is also believed that in the next five years, online grocery shopping will amount to 10% of the overall market, with a 60% increase to £19.8 billion by 2023. ("Brits spent £12.3 billion on online groceries in 2018," 2019)

Table 4 and 5 presented below depict the advantages and disadvantages associated with online grocery shopping and in-store grocery shopping.

Advantages	Disadvantages
Less likely to overspend	Hidden charges such as delivery
Able to add items to one's shopping list throughout the week	Unable to handpick items
Able to shop 24/7	Possibility of fraud
Time-saving	
Less stressful than dealing with busy grocery stores at peak hours	
Able to compare prices with other platforms	
Less money spent on gas money	
Easier to reorder items and make adjustments	

Figure 4: Pros and Cons of Online Grocery Shopping

Advantages	Disadvantages
No Security Issue	Does not operate 24/7
Immediacy	Time consuming and long lines
Able to handpick items	Commute issue
No hidden charges	Limitation and restrictions
Interaction	Tiring and exhausting

Figure 5: Pros and Cons of In-Store Grocery Shopping

According to the American Heart Association, online grocery shopping experienced a surge in popularity during the COVID-19 pandemic as consumers sought a safe way to obtain groceries. Presently, online grocery shopping remains highly popular due to several advantages it offers. These include saving time and money, enjoying convenience, minimising impulse buying of unhealthy items, sticking to a budget by comparing prices and utilising electronic coupons, planning meals in advance, avoiding duplicate purchases, being aware of product availability and selecting substitutions, using the virtual cart as a modern-day shopping list, and choosing between free store pick-up or home delivery services. ("Online grocery shopping offers convenience, health benefits," 2022)

2.4 Recommendation Systems

A recommendation system is a system that tries to predict the ratings or preferences of a user for an item. It takes into account a total number of variables, such as previous purchases, search history, demographics, and more, to predict and provide recommendations. Companies like Amazon, Netflix, and YouTube are built on these systems.

A classic recommendation system has a four-stage process. First, data collection occurs. Data can either be considered as explicit ratings, i.e., ratings on a scale of 1 to 5, or as implicit ratings, i.e., the number of clicks. The second stage involves data storage, which is determined by the nature of the collected data and the intended recommendations. In the analysis stage, algorithms are employed to identify patterns and similarities within user engagement data. In the last stage, the relevant information is retrieved and suggestions are made.

Machine learning has found extensive applications in addressing diverse challenges, with product recommendation being a prominent example. There are several factors behind suggested item recommendations, as depicted in the figure below.

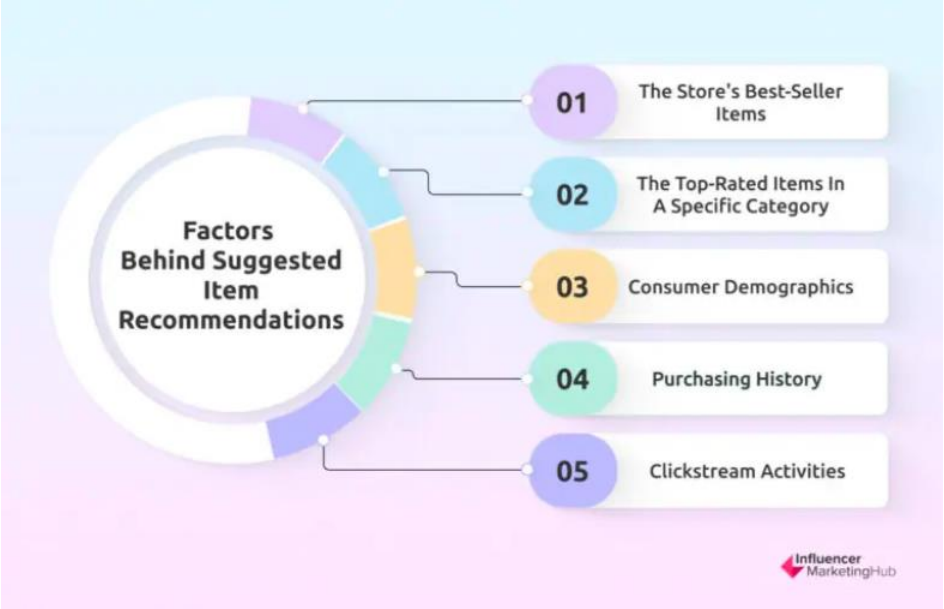


Figure 6: Factors Affecting Recommendations

There are different techniques for recommendation systems. This includes collaborative filtering, content-based filtering, and an integration of both known as hybrid systems. Collaborative filtering involves analysing the behaviour of similar users or items, while content-based filtering uses the characteristics of products to make recommendations. The hybrid systems combine these two approaches to provide more accurate and diverse recommendations. (Kumar and Thakur, 2018).

The figure below represents an overview of Netflix's recommendation system.

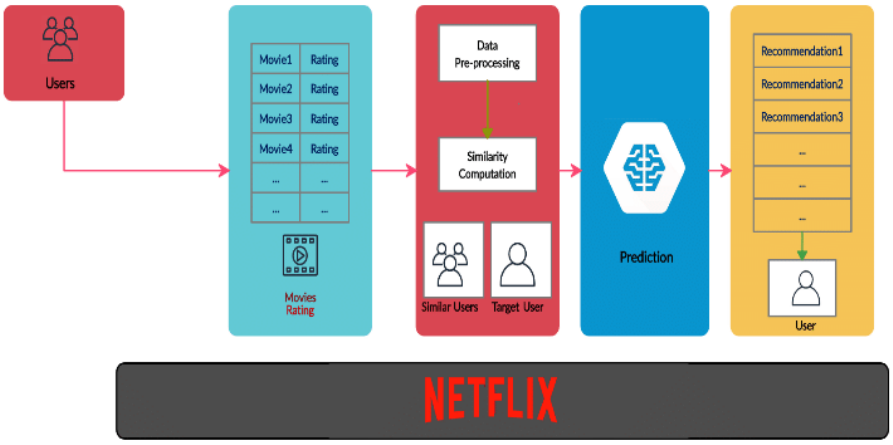


Figure 7: Netflix Recommendation ("Has the Future Started?," 2022)

The figure below portrays the generic architecture of the IALBR (Interest Aware Location-Based Recommender) system, which is another type of recommendation system.

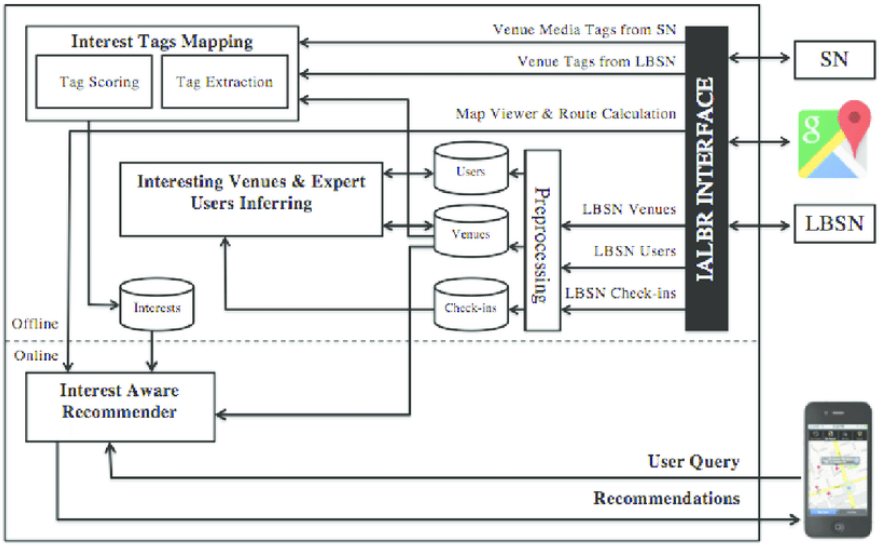


Figure 8: IALB recommender System ("Figure 3. Interest-Aware Location-Based Recommender System (IALBR)...," 2016, p. 3)

The figure shown below is a recommender system that uses a large set of collected data from various sources.

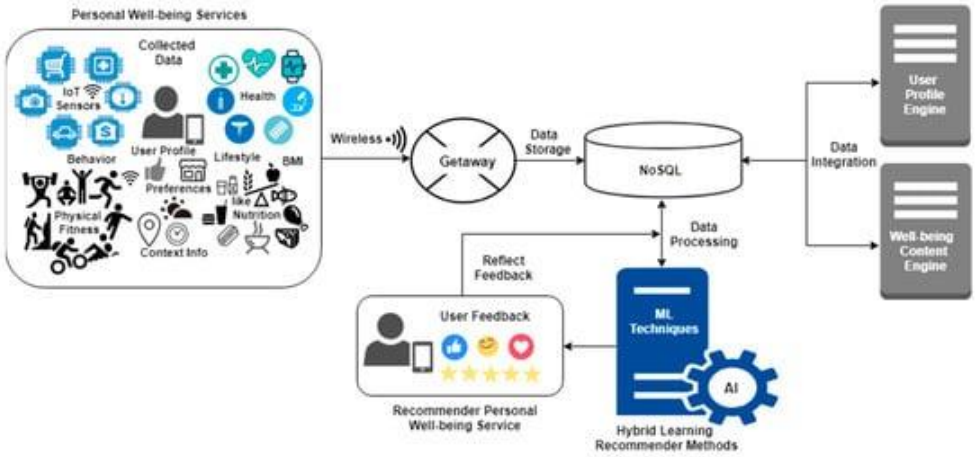


Figure 9: Recommender Based on Hybrid Models (Nouh et al., 2019)

Recommendation systems demonstrate their versatility by being widely adopted across various industries and adapting to specific needs and requirements. Their effectiveness in delivering personalised recommendations and enhancing user experiences is clearly evident, highlighting their significance in different domains.

2.5 Predictions vs. Recommendations

While predictions and recommendations might appear similar, they are distinct concepts in the realm of machine learning. Predictions involve forecasting future outcomes, while recommendations focus on suggesting specific items or actions to individuals.

For example, predictions are estimates of how much a person would like an item. This would often scale to some rating scale or be tied to searching for or browsing specific products. Recommendations are suggestions for items a person might like or that might fit what a person is doing. These are often presented in the form of "top-N lists". Usually, recommendations involve items that the person did not interact with.

2.6 Use Cases and Applications

2.6.1 E-Commerce Recommendations

E-Commerce is by far the most common and frequently encountered use case for recommendation systems in action. Amazon was a pioneer in introducing this change back in 2012 by making use of item-item collaborative filtering to recommend products to buyers. To this date, Amazon continues to remain a market leader by virtue of its helpful and user-friendly recommendation engine. According to

a 2023 report from the Statista Research Department, the net revenue of the multinational e-commerce company in 2022 reached almost 514 billion U.S. dollars. This signifies a growth of approximately 9% compared to the previous year, 2021. These figures suggest that recommendation systems have been highly effective in contributing to the company's success. (“Amazon annual net sales 2022,” 2023)

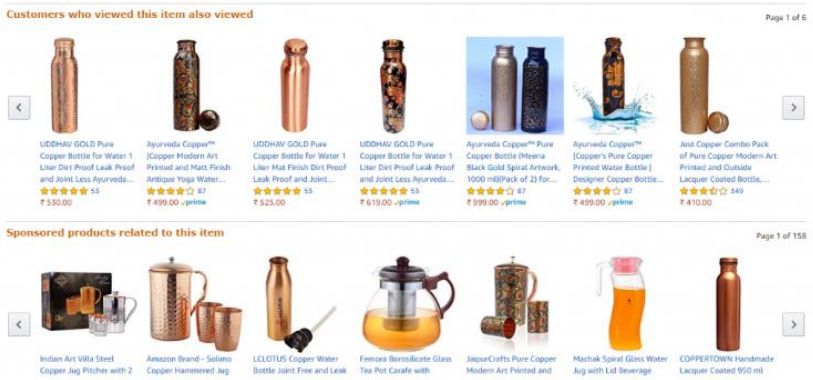


Figure 10: E-commerce Recommendations

2.6.2 Media Recommendations

Recommendation systems are one of the building blocks that help companies such as Netflix, Spotify, Prime Video, YouTube, and Disney+ attain their current success in media and entertainment. These media streaming providers leverage a relational understanding of their user base's content consumption patterns. Furthermore, the self-learning and self-training capabilities of AI within recommendation engines enhance the relevance of recommendations, ensuring high levels of user engagement while mitigating customer churn.

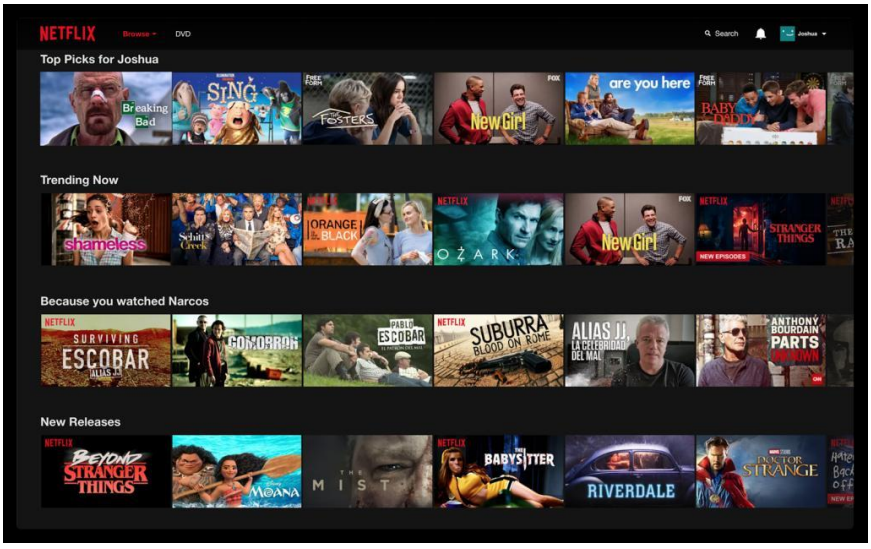


Figure 11: Media Recommendations

2.6.3 Video Games and Store Recommendations

Gaming platforms like Steam, the Xbox Games Store, and the PlayStation Store have excellent recommender engines that personalise game recommendations based on players' histories and preferences.

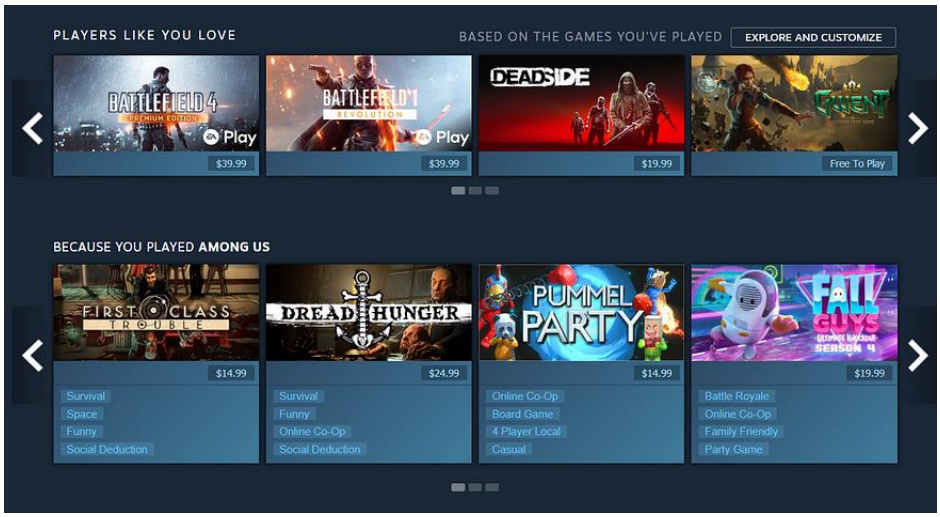


Figure 12: Gaming Platform Recommendations

2.6.4 Location-Based Recommendations

Geographic location, as a demographic factor, serves as a bridge between the online and offline customer experience, enhancing marketing, advertising, and sales effectiveness for improved profitability. Businesses have been implementing reliable location-based recommender systems (LBRS) or location-aware recommender systems (LARS). For example, Sephora can trigger notifications based on people's location to advertise ongoing promotions when they are located near a physical store, prompting people to enter their establishment.

Figure 13: Location-Based Recommendations

2.6.5 Health and Fitness Recommendations

Health and fitness recommender systems use user inputs like dietary preferences, activity levels, and fitness goals to suggest customised plans and routines for achieving desired outcomes.

Figure 14: Health and Fitness Recommendations

2.7 Impact of recommendation systems

Recommendation systems have revolutionised online discovery and consumption by utilising intelligent algorithms and vast data to suggest personalised items, services, and content. As mentioned, recommendation systems can have a very powerful impact on a business and its revenue.

Companies invest in recommendation systems primarily to drive sales and increase revenue. These systems also enhance consumer engagement on their websites and prolong session durations. Based on the authors' analysis, recommendations have a positive effect on the sales of a recommended item. Another benefit of a recommendation system is that it increases the engagement and satisfaction of customers by providing them with a myriad array of personalised products (Pathak et al., 2010). The author run a multiple linear regression of customer retention rates on purchase quantity and category diversity, and the results supported the statement that category diversity more strongly influences customer retention. (Park and Han, 2013)

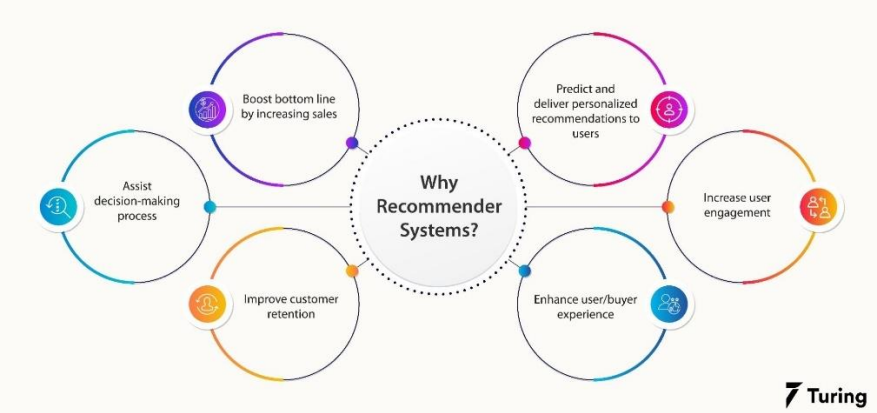


Figure 15: Impacts of Recommendation Systems ("How Collaborative Filtering Works in Recommender Systems," n.d.)

According to a study conducted by Mordor Intelligence, a company that provides reports on detailed market coverage and analyst insight into various markets, the recommendation engine market size is expected to grow from USD 5172.41 million in 2023 to USD 21574 million by 2028 ("Recommendation Engine Market Size & Share Analysis - Industry Research Report - Growth Trends," 2023). As per Forrester, a company that has been doing research for more than 35 years to provide a clear vision to its global consumer business and technology leaders, 89% of digital businesses are investing in personalisation, such as Coca-Cola, Fabletics, and so on (Witcher, 2018). Research conducted by Epsilon in 2018 showed that 80% of consumers are more likely to make a purchase when brands offer personalised experiences ("New Epsilon research indicates 80% of consumers are more likely to make a purchase when brands offer personalised experiences," 2018). According to McKinsey, non-personalised communications pose a business risk in a low-loyalty environment while putting emphasis on the fact that 71% of consumers expect personalization while 76% get frustrated when they don't find it ("The value of getting personalization right—or wrong—is multiplying | McKinsey," 2021). In a report from Invesp, 49% of consumers said they have purchased a product that they did not initially intend to buy after receiving a personalised recommendation; 54% of retailers reported product recommendation as the key driver of the average order value in the customer purchase; and 75% of customers are more likely to buy based on personalised recommendations (Ross, 2019). Likewise, research from Salesforce mentioned that personalised product recommendations account for only 7% of e-commerce traffic but make up 24% of orders and 26% of revenue, and 52% of consumers are likely to switch brands if a company does not personalise communications to them ("How Artificial Intelligence Powers Personalized Shopping," 2018). A study conducted by Monetate found that online retail browsers who engaged with a recommended product had a 70% higher conversion rate during that session, and shoppers that did interact with the product recommendation but did not buy anything were 20% more likely to return to the site later ("The Impact of Product Recommendations," 2018). As per another report from McKinsey, personalised product recommendations are estimated to account for more than 35% of purchases on Amazon ("How retailers can keep up with consumers | McKinsey," 2013).

2.8 Machine Learning and Data Mining Techniques

The machine learning techniques mentioned below were chosen after a thorough analysis of our dataset, and the following methods were identified as the most effective.

2.8.1 Clustering

Clustering is a technique used to identify distinct groups or clusters within a data set where observations within each cluster are similar to each other while those in different clusters are dissimilar (Sohil et al., 2022). Clustering is frequently used in unsupervised learning when a dataset lacks a predefined outcome variable. It has become widely popular in statistical data analysis as it can uncover valuable patterns in intricate datasets. The application of clustering spans various domains such as business, finance, healthcare, and the social sciences.

Data analysis is essential across various scientific disciplines, including communication science, computer science, and biology. Clustering plays a crucial role as a fundamental part of data analysis (Xu and Tian, 2015).

2.8.1.1 K-Means Clustering Algorithm

The K-means algorithm is a popular iterative clustering method ideal for quantitative variables using squared Euclidean distance as the dissimilarity measure.

The K-Means Pseudocode is as follows:

- Input: Data points D, Number of clusters k
- Step 1: Initialize k centroids randomly.
- Step 2: Associate each data point in D with the nearest centroid. This will divide the data points into k clusters.
- Step 3: Recalculate the position of centroids.
- Repeat steps 2 and 3 until there are no more changes in the membership of the data points
- Output: Data points with cluster memberships

Figure 16: K-Means Pseudocode (Singh and Reddy, 2015)

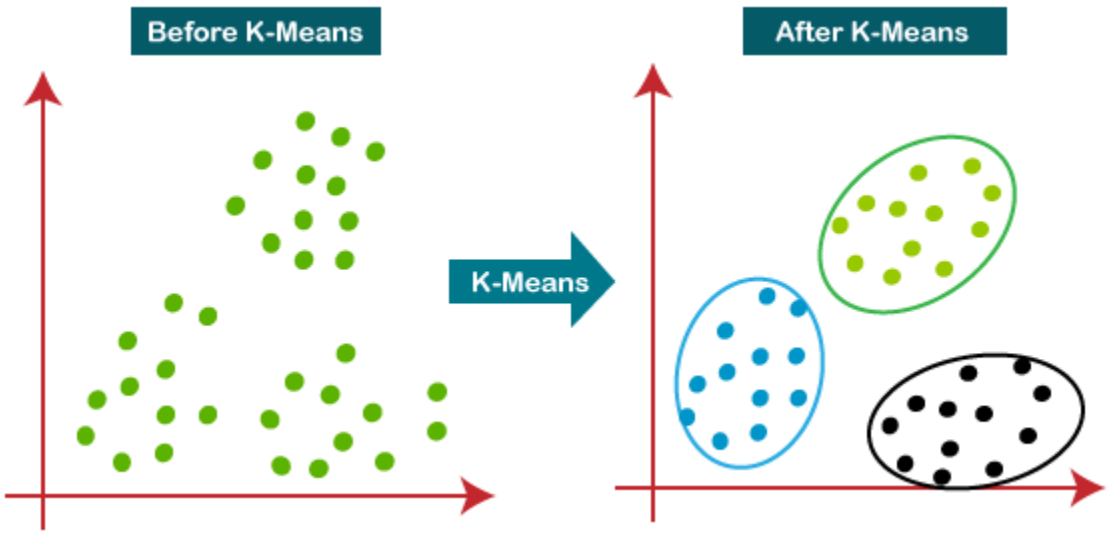


Figure 17: K-Means Clustering

2.8.1.2 Hierarchical Clustering Algorithm

Hierarchical clustering is a flexible alternative to K-Means clustering as it does not require a predetermined number of clusters (K). It offers the advantage of producing a dendrogram, a visually appealing tree-based representation of the observations (Sohil et al., 2022).

The Hierarchical Pseudocode is as follows:

- Turn each input element into a singleton, i.e. into a cluster of a single element,
- For each pair of clusters c1, c2, calculate their distance d(c1, c2).
- Merge the pair of clusters that take the smallest distance.
- Continue the step 2, until the termination criterion is satisfied.
- The termination criterion most commonly used in a threshold of the distance value.

Figure 18: Hierarchical Pseudocode (Markowska-Kaczmar et al., 2010)

Calculating the closest distance between clusters is crucial when dealing with multiple observations within a cluster. This is done using linkage methods, which determine how the distance between clusters is measured. The four most common types of linkage are complete, single, average, and centroid.

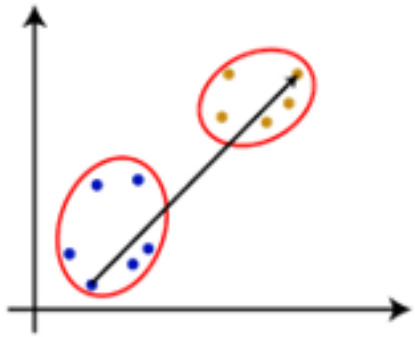
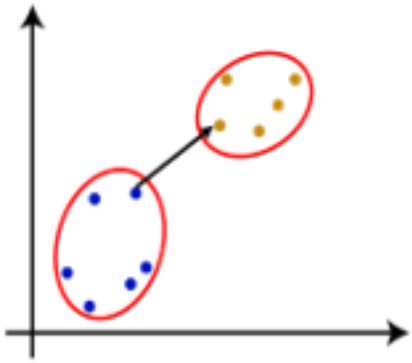
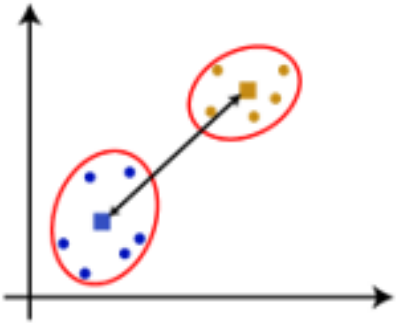
Linkage		Description
Complete		The farthest distance between the two points of two different clusters.
Single		The shortest distance between the closest points of two different clusters.
Average		The distance between each pair of datasets is added up and then divided by the total number of datasets to calculate the average distance between two clusters.
Centroid		The distance between the centroid of two different clusters.

Figure 19: Linkages

The figure below illustrates cluster creation in agglomerative clustering on the left and the resulting dendrogram on the right.

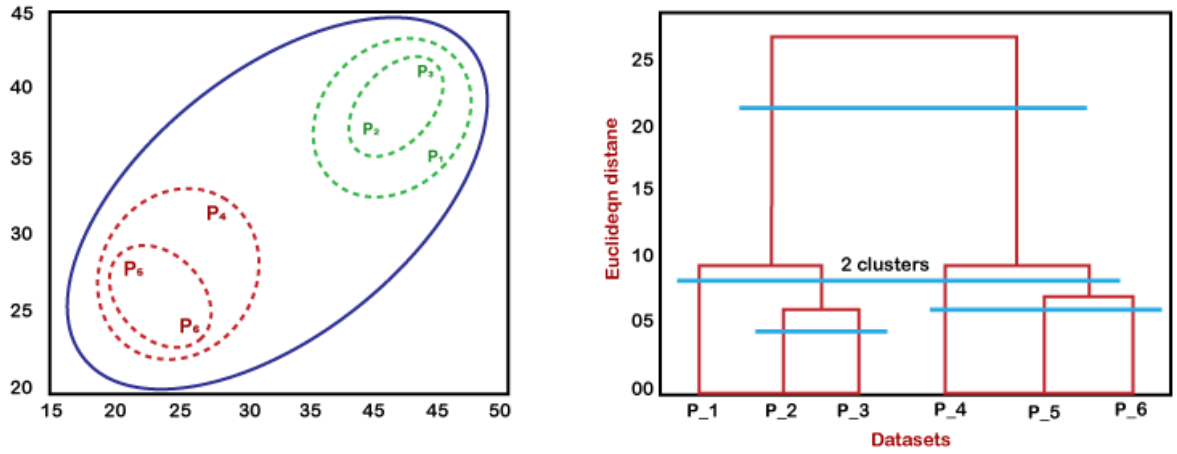


Figure 20: Hierarchical Clustering

2.8.1.3 Spectral Clustering Algorithm

Spectral clustering uses spectral graph theory to group data based on their similarity. Spectral clustering is valuable for complex data structures and applicable to various data types, providing an effective approach to identifying clusters in datasets.

Suppose our data consists of n points $x_1, x_2, \dots, x_n$, which can be arbitrary objects. Their pairwise similarities $s_{ij} = (x_i, x_j)$ can be measured by some similarity function that is symmetric and non-negative, and the similarity matrix is represented as $S = (s_{ij})\ i, j = 1, \dots, n$.

The steps in spectral clustering algorithm are as follows:

- Input: Similarity matrix $S \in \mathbb{R}^{n \times n}$, number k of clusters to construct
- Construct a similarity graph. Let W be its weighted adjacency matrix
- Compute the unnormalized Laplacian L
- Compute the first k eigenvectors $u_1, \dots, u_k$ of L
- Let x $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors $u_1, \dots, u_k$ as columns
- For $i = 1, \dots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i^{th} row of U
- Cluster the points $(y_i)_{i=1,\dots,n}$ in $\mathbb{R}^k$ with the k-means algorithm into clusters $C_1, \dots, C_k$.
- Output: Clusters $A_1, \dots, A_k$ with $A_i = \{j \mid y_j \in C_i\}$

Figure 21: Spectral Pseudocode (Von Luxburg, 2007)

2.8.2 Collaborative Filtering

Collaborative filtering leverages relationships between products and users' interests for accurate recommendations. It has two approaches: user-based and item-based. User-based filtering relies on user similarity or neighbourhood, while item-based filtering focuses on item similarity. These techniques help identify relevant product recommendations for users.

In user-based collaborative filtering, the recommendation is aimed at an active user. The collaborative filtering engine identifies users who have similar rating patterns to the active user based on factors such as history, preferences, and choices. This similarity is used to generate personalised recommendations. Collaborative filtering is widely employed by various websites to build effective recommendation systems.

The first step in user-based collaborative filtering is to find the similarity of users to the active user. This can be done using the cosine similarity or Pearson correlation formula:

$$similarity(\vec{A}, \vec{B}) = \frac{\vec{A}.\vec{B}}{\| \vec{A} \| * \| \vec{B} \|}$$

Where:

- $\vec{A}, \vec{B}$: user vectors

Figure 22: Cosine Similarity

$$Sim(a,b) = \frac{\sum_p (r_{ap} - \bar{r}_a)(r_{ab} - \bar{r}_b)}{\sqrt{\sum (r_{ap} - \bar{r}_a)^2} \sqrt{\sum (r_{bp} - \bar{r}_b)^2}}$$

Where:

- r_{up}: rating of user u against item p
- p : items

Figure 23: Pearson Correlation

Ratings given by more similar users are given a higher weight than those given by less similar users. This is achieved by multiplying each user's rating by a similarity factor calculated using the aforementioned formula. The missing rating can then be calculated using this weighted average approach:

$$r_{up} = \bar{r}_u + \frac{\sum_{i \in users} sim(u,i) * r_{ip}}{\sum_{i \in users} |sim(u,i)|}$$

Figure 24: Calculate User-Based Missing Rating

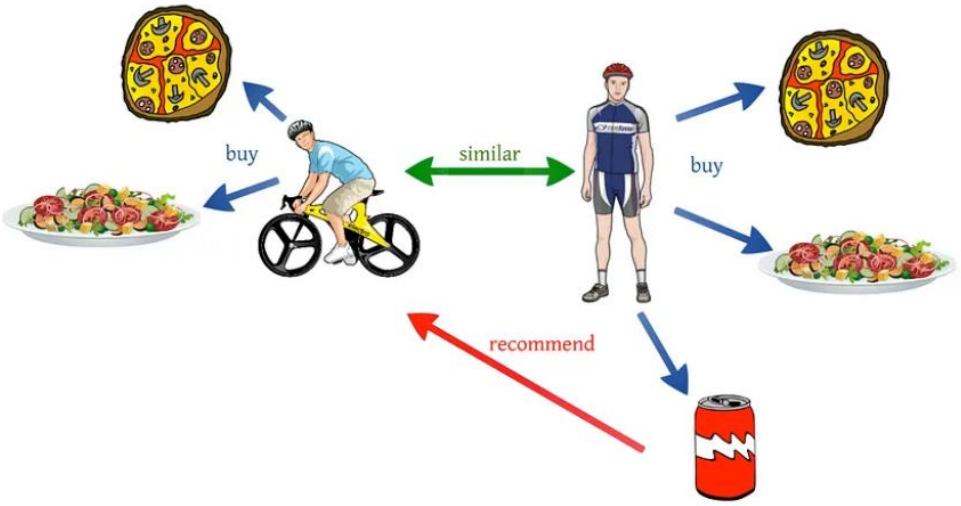


Figure 25: User-Based Recommendations

Item-based collaborative filtering focuses on exploring the relationships between pairs of items. Rather than matching the user to similar customers, item-to-item collaborative filtering matches each of the user's purchased and rated items to similar items, then combines those similar items into a recommendation list (Linden et al., 2003). To calculate the similarity between items in item-to-item collaborative filtering, formulas like cosine similarity or Pearson correlation are used. The prediction of ratings is done by taking the weighted sum of the ratings of other similar items using the following formula:

$$rating(U, I_i) = \frac{\sum_j rating(U, I_j) * s_{ij}}{\sum_j s_{ij}}$$

Figure 26: Calculate Item-Based Missing Rating

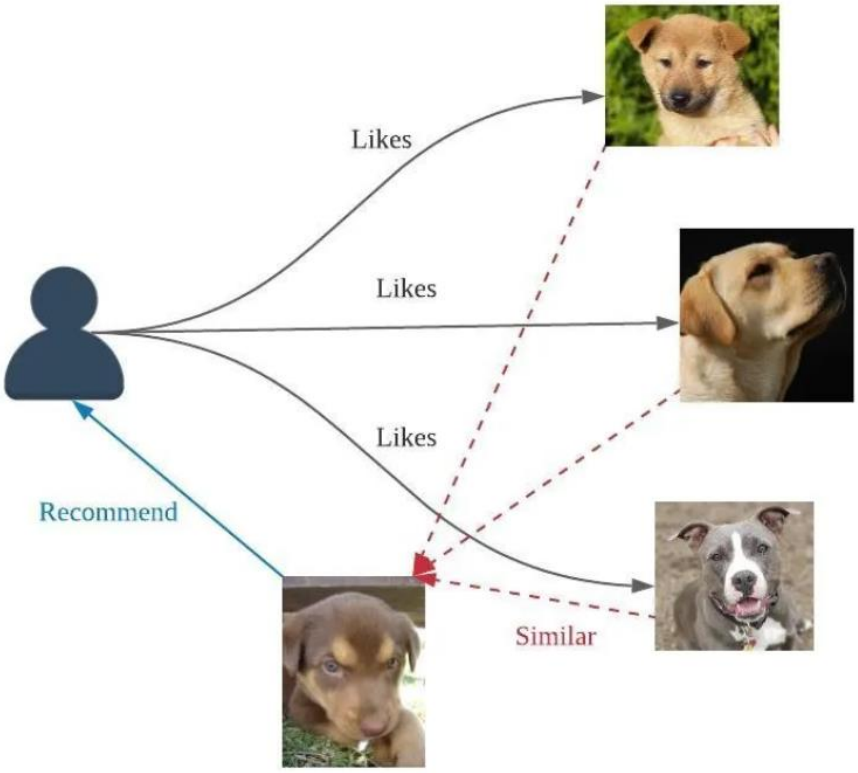


Figure 27: Item-Based Recommendations

To create a recommendation system for groceries, it is crucial to consider both the users and the items. Advanced collaborative methods like Singular Value Decomposition (SVD), Non-Negative Matrix Factorization (NMF), and Neural Collaborative Filtering (NCF) can be employed to leverage information from both users and items. These techniques allow for a more comprehensive analysis and generate more accurate recommendations in the grocery domain.

2.8.2.1 Singular Value Decomposition (SVD)

Collaborative Filtering encompasses two popular approaches: latent factor models and neighbourhood models. Latent factor models, such as Singular Value Decomposition (SVD), extract features from user and item matrices. SVD is a popular method in linear algebra for dimensionality reduction and matrix factorization in machine learning. It reduces the dimensionality of the user-item matrix from N-dimension to k-dimension (k<N). SVD constructs a matrix using users' ratings, with users as rows and items as columns. By decomposing the matrix into three other matrices, SVD extracts the factors representing features and correlations from the high-level user-item-rating matrix (Eckart and Young, 1936).

To summarise, SVD captures underlying patterns and similarities between users and items by decomposing the user-item interaction matrix. It enables dimensionality reduction and matrix factorization to extract meaningful information.

$$A = USV^T$$

Where:

- Matrix U is the singular matrix of user-latent factors
- Matrix S is a diagonal matrix showing the strength of each latent factors
- Matrix V is a singular matrix of item-latent factors.

Figure 28: SVD Matrix Decomposition

Matrix $A = USV^T$ uncovers the relationship between users and items by mapping them into a latent space with r-dimensionality. The rating given by a user on an item, $\hat{r}_{ui}$, is determined as follows:

$$\hat{r}_{ui} = x_i^T . y_u$$

Where:

- x_i is a vector considering each item
- y_u is a vector representing each user

Figure 29: SVD Calculate Predicted Ratings

To ensure the accuracy of the predicted ratings, the goal is to minimize the loss between the actual rating, r_{ui}, and the predicted rating, $\hat{r}_{ui}$, which serves as a measure of accuracy in capturing user-item interactions. This is achieved through the objective function:

$$Min(x, y) \sum_{(u,i) \in K} (r_{ui} - x_i^T . y_u)^2$$

Figure 30: SVD Minimising Loss Function

Regularisation is crucial to avoid overfitting, generalise the dataset, and balance the model's performance by including a penalty term. The selection of λ depends on the specific problem and the

characteristics of the dataset. Generally, λ is obtained through hyperparameter tuning such as cross-validation. Multiple values of λ are evaluated to find the optimal balance between bias and variance in the model. (Koren et al., 2009).

To improve the model's performance by reducing the error between actual and predicted values, a bias term was added. This bias term contains the average rating of all items (μ), the average rating of each item $i - \mu(b_i)$, and the average rating given by the user $u - \mu(b_u)$ (Koren et al., 2009). The complete objective function incorporating the bias term to refine predictions and account for user and item-specific bias is:

$$Min(x, y, b_i, b_u) \sum_{(u,i) \in K} (r_{ui} - x_i^T . y_u - \mu - b_i - b_u)^2 + \lambda(\|x_i\|^2 + \|y_u\|^2 + {b_i}^2 + {b_u}^2)$$

Where:

- (u, i): user-item pair
- μ: the average rating of all items
- b_i: average rating of item i minus μ
- b_u: the average rating given by user u minus μ

Figure 31: SVD Minimising loss function with Bias

2.8.2.2 Non-Negative Matrix Factorization (NMF)

Non-Negative Matrix Factorization (NMF) is another method for matrix factorization (Gillis, n.d.). NMF differs from SVD because NMF decomposes the non-negative utility matrix R into the product of matrices W and H. The component-wise non-negativity is a substantial difference from SVD. By decomposing the matrix R into multiple latent factors, NMF uncovers hidden patterns and relationships, allowing for more accurate recommendations.

$$R \approx W \times H^T; \; W \geq 0, H \geq 0$$

Where:

- W represents the user's latent factors
- H represents the item's latent factors

Figure 32: NMF Matrix Decomposition

To predict the rating of an item for a user, the dot product is calculated for matrices P and Q:

$$\hat{r}_{ij} = w_i h_j^T = \sum_{k=1}^{K} w_{ik} h_{kj}$$

Figure 33: NMF Calculate Predicted Ratings

With W and H, an approximation of matrix R can be calculated, which would estimate the values for missing inputs. Gradient descent, an optimization algorithm, is often used to obtain matrices P and Q.

The algorithm proceed as follows:

- Initialize matrices P and Q with random numbers.
- For each iteration, calculate the product of P and Q and compare with matrix R
- If the product is close to R, the iteration stop; otherwise, update the values of P and Q to minimize the difference.

Figure 34: NMF Algorithm Procedure

This iterative procedure continues until the minimum estimated error between the approximated and the real matrix is reached. The squared error for each user-item pair can be calculated as:

$$e_{ij}^2 = (r_{ij} - \hat{r}_{ij})^2 = \left(r_{ij} - \sum_{k=1}^{K} p_{ik} q_{kj} \right)^2$$

Figure 35: NMF Squared Error Function

To minimise the error, the squared error equation is differentiated with respect to both p_{ik} and q_{kj}:

$$\frac{\partial e_{ij}^2}{\partial p_{ik}} = -2\left(r_{ij} - \hat{r}_{ij}\right)\left(q_{kj}\right) = -2e_{ij}q_{kj}$$

$$\frac{\partial e_{ij}^2}{\partial q_{kj}} = -2\left(r_{ij} - \hat{r}_{ij}\right)\left(p_{ik}\right) = -2e_{ij}p_{ik}$$

Figure 36: NMF Minimising Error

The values of p_{ik} and q_{kj} can then be updated using the obtained gradient:

$$p'_{ik} = p_{ik} + \alpha \frac{\partial e_{ij}^2}{\partial p_{ik}} = p_{ik} + 2\alpha e_{ij} q_{kj}$$

$$q'_{kj} = q_{kj} + \alpha \frac{\partial e_{ij}^2}{\partial q_{kj}} = q_{kj} + 2\alpha e_{ij} p_{ik}$$

Where:

- α is the learning rate which determined the rate of approaching to the minimum.

Figure 37: NMF updating matrices P and Q

Consequently, the learning rate, α, depict the step size at which model parameters are updated during the optimization process. It denotes the rate of convergence towards the optimal solution. A higher learning rate could lead to faster convergence but at the risk of overshooting the optimal values, while a lower learning rate would ensure precise updates at the cost of slower convergence.

2.8.2.3 Neural Collaborative Filtering (NCF)

To address this limitation of matrix factorization, (He et al., 2017), mentioned in Appendix 2, NCF leverages Deep Neural Networks (DNNs) in its architecture. The NCF architecture consists of several layers that process the input data. The input layer binarizes a sparse vector to indicate user-item interaction, while the embedding layer converts the sparse representation into dense vectors that capture the underlying characteristics of users and items.

The Neural CF layers employ a multi-layered neural architecture to map the latent user and item vectors to predict scores. Non-linear transformations and activations are applied to capture complex relationships between users and items. Finally, the output layer returns the predicted score by minimising the ranking loss, ensuring the item ranking and recommendations are accurate.

The NCF's predictive model can be formulated as follows:

$$\hat{y}_{ui} = f\left(P^T v_u^U Q^T v_i^I \middle| P, Q, \theta_f\right)$$

Where:

- P is the latent factor matrix for users, $P \in \mathbb{R}^{M \times K}$
- Q is the latent factor matrix for items, $Q \in \mathbb{R}^{M \times K}$
- θ_f denotes model parameters

Figure 38: NCF Calculate Predicted Ratings

Since f is defined as a MLP, it can be formulated as acting as the scoring function:

$$f(P^T v_u^U Q^T v_i^I) = \phi_{out}\left(\phi_X\left(\dots \phi_2\left(\phi_1(P^T v_u^U Q^T v_i^I)\right)\dots\right)\right).$$

Where:

- ϕ_{out} is the mapping function for the output layer.
- ϕ_X is the mapping function for the output layer and the x[th] NCF layer.

Figure 39: Scoring Function

To learn model parameters, the following pointwise squared loss equation is used:

$$L_{sqr} = \sum_{(u,i)\in y\cup y^-} w_{ui}(y_{ui} - \hat{y}_{ui})^2$$

Where:

- y denotes the set of observed interaction in Y
- y^- denotes the set of unobserved interactions
- w_{ui} is the weight of training instance

Figure 40: Pointwise Squared Loss Equation

It is important that $\hat{y}_{ui}$ should return a score between 0 and 1 to represent the likelihood of the given user-item interaction. This can be achieved by using a probabilistic function such as Logistic or Probit function as the activation function for the output layer ϕ_{out}. Hence, the likelihood function can be defined as:

$$p(\mathrm{y}, y^-|P, Q, \theta_f) = \prod_{(u,i)\in y} \hat{y}_{ui} \prod_{(u,i)\in y^-} (1 - \hat{y}_{ui})$$

Figure 41: Likelihood Function

By taking the negative logarithm of the likelihood:

$$L = -\sum_{(u,i)\in y} \log \hat{y}_{ui} - \sum_{(u,i)\in y^-} \log(1 - \hat{y}_{ui}) = -\sum_{(u,i)\in y\cup y^-} y_{ui} \log \hat{y}_{ui} + (1 - y_{ui}) \log (1 - \hat{y}_{ui})$$

Figure 42: Log Loss Function

Which is the cross-entropy loss/ log loss. By employing this probabilistic treatment, NCF transforms the recommendation into a binary classification problem.

The NCF design combines two instantiations: Generalized Matrix Factorization (GMF) and Multi-Layer Perceptron (MLP). GMF uses sigmoid as the activation function and learns the edge weights from the data with log loss. The predicted output can be expressed as:

$$\hat{y}_{ui} = a_{out}(h^T(p_u \odot q_i))$$

Where:

- a_{out} is the activation function
- h is the edge weights of the output layer

Figure 43: GMF Predicted Output

Next, the NCF tries to model the user-item interaction using Multi-Layer Perceptron (MLP). The NCF uses ReLU as an activation function. The MLP model under the NCF framework is thus defined as:

$$z_1 = \emptyset(p_u, q_i) = \begin{bmatrix} p_u \\ q_i \end{bmatrix},$$
$$\emptyset_2(z_1) = a_2(W_2^T z_1 + b_2),$$
$$\dots\dots\dots$$
$$\emptyset_L(z_L - 1) = a_L(W_L^T z_{L-1} + b_L),$$
$$\hat{y}_{ui} = \sigma\left(h^T \emptyset_L(z_L - 1)\right),$$

Where:

- W_x is the weight matrix
- b_x is the bias vector
- a_x is the activation function for the xth layer's perceptron
- p is the latent vector for the user
- q is the latent vector for an item

Figure 44: MLP Predicted Output

Now that NCF has two instantiations, namely, GMF and MLP, NCF combines these models together to superimpose their desirable characteristics and concatenates the outputs of GMF and MLP before feeding them into the NeuMF layer.

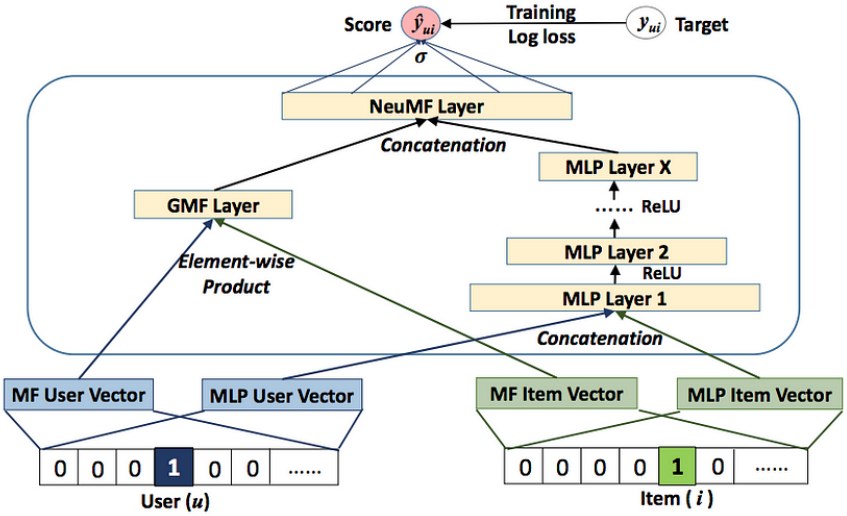


Figure 45: NCF Overview

The score function in the NCF can be modelled as:

$$\emptyset^{GMF} = p_u^G \odot q_i^G,$$

$$\emptyset^{MLP} = a_L\left(W_L^T\left(a_{L-1}\left(\ldots a_2\left(W_2^T\begin{bmatrix}p_u^M\\ q_i^M\end{bmatrix} + b_2\right)\ldots\right)\right) + b_L\right),$$

Finally, the predicted score is obtained by:

$$\hat{y}_{ui} = \sigma\left(h^T\begin{bmatrix}\emptyset^{GMF}\\ \emptyset^{MLP}\end{bmatrix}\right),$$

Where:

- σ is the activation function of the output layer in NCF

Figure 46: MLP and GMF combination to predict user-item interactions

The NeuMF layer models the latent structures of user-item interactions in NCF by combining the linearity of matrix factorization with the non-linearity of DNNs, which leads to improved recommendations.

2.9 Literature Review

This section contains some of the important research papers that had a direct effect on building the proposed recommendation system.

Table 1: Literature Review 1

Paper Title	Author	Research Description	Conclusion
Research and Implementation of SVD in Machine Learning	(YongchangWang and Zhu, 2017)	The author focuses on data reduction techniques such as PCA and SVD. In an experiment, they worked with a row of words and their appearance in the titles of multiple documents.	Dimension reduction for big data mining and machine learning. SVD can get the same results as PCA, but it is considered to be better and more stable.
Context-aware QoS Prediction with Neural Collaborative Filtering for Internet-of-Things Service	(Gao et al., 2020)	A comprehensive framework was created by the author to tackle quality-of-service (QoS) prediction in an IoT setting. The proposed QoS prediction method was Context-aware Neural Collaborative Filtering (CNCF).	When evaluated with a series of training sets and compared, the results showed that the CNCF performed better than User-Based Probabilistic Collaborative Filtering, Item-Based Probabilistic Collaborative Filtering, NCF, and so on.
Prediction of Customer Behaviour Analysis Using Classification Algorithms	(Raju and Dhandayudam, 2018)	The authors used three classification models to develop business strategies for analysing customer data. The models include Naïve Bayes and J48 decision-trees.	After repeating the experiments with 10 iterations on each model on a bank dataset, summary tables containing the values for TP, TN, TP, TN, Accuracy, Sensitivity and Specificity showed that the J48 classifier was performing better.
Foundations of consumer behaviour analysis	(Foxall, 2001)	It is a summary-like paper that reviews the behavioural basis of consumer choice, the behavioural economics of consumption, and so on.	Although marketing is very successful, it has its own weaknesses as researchers generally ignore recent changes in behaviour. It is a serious disadvantage to interpret complex human behaviour from laboratory experiments as reality.

Table 2: Literature Review 2

Paper Title	Author	Research Description	Conclusion
Performing Customer Behaviour Analysis using Big Data Analytics.	(Khade, 2016)	The author proposed a MapReduce implementation of the C4.5 decision tree. The author provided a detailed overview of Apache Hadoop and the proposed methodology.	A proposed system for distributed implementation of the C4.5 algorithm while using MapReduce framework.
Data Mining Approach for Intelligent Customer Behaviour Analysis for a Retail Store.	(Abirami and Pattabiraman, 2016)	The author tried a new approach to customer classification based on RFM (Mode) to analyse and predict customer behaviour using k-mean clustering and association rule mining.	The new approach seems to be effective in segmenting customers. Depending on the dataset, RFM (Mean) can also be used.
An Efficient CRM-Data Mining Framework for the Prediction of Customer Behaviour.	(Bahari and Elayidom, 2015)	The author proposed a CRM-data mining framework based on classification models, namely Naïve Bayes and MLP Neural Networks.	The MLP Neural Network showed better classification accuracy based on True Positive Rate, False Positive Rate, and ROC Area.
Design and Implementation of Online Shopping System Based on B/S Model.	(Wei and Zhang, 2018)	The author provided an overview of how to implement an online shopping system, along with defining the system requirements, database design, and the implementation of the modules.	
Python and MySQL based Smart Digital Retail Management System	(Shah et al., 2021)	The author built a retail management system that would keep track of products expiration dates, and customers total expenditures. The system was designed to be linked to other businesses.	

Table 3: Literature Review 3

Paper Title	Author	Research Description	Conclusion
Survey on Collaborative Filtering, Content-Based Filtering and Hybrid Recommendation System	(B.Thorat et al., 2015)	The authors presented a detailed overview of Collaborative Filtering, Content-Based Filtering, and Hybrid Recommendation systems.	Working with large datasets for example, commercial datasets, will lead to data sparsity. Traditional Collaborative Filtering algorithms will suffer from scalability. Recommendation system with a diverse range of products will lead to lower accuracy. Recommendation systems on an e-commerce platform are easy target to attacks.
An Examination of Social Influence on Shopper Behaviour Using Video Tracking Data	(Zhang et al., 2014)	The authors used videos to track customers' paths and activities during a store visit. The authors wanted to know the customers' product interactions and purchase likelihood. The paper presents models of shopper touch frequency and purchase implemented in a hierarchical Bayes framework within zones.	- The paper presents multiple evaluations and results. For example, the conclusion that the author presented was that shoppers are less likely to buy items when a store is crowded. Likewise, shoppers tend to buy items on clearance. An interesting observation is that customers who walk slowly tend to have a higher purchase rate. When people interact with their peers, their purchase likelihood increases. A salesperson also increases a shopper's purchase likelihood.
Detect Resource Related Events in a Cloud-Edge Infrastructure using Knowledge Graph Embeddings and Machine Learning	(Mitropoulou et al., 2022)	The paper is about detecting anomalous events by making use of Isolation Forest and Cluster-based Local Outlier Factor.	- The author used evaluation metrics such as TP, TN, FP, FN, Accuracy, Precision and much more on an unsupervised model.

Table 4: Literature Review 4

Paper Title	Author	Research Description	Conclusion
The Why and How of Nonnegative Matrix Factorization	(Gillis, 2014)	The author provided a huge overview of NMF and its application in image processing, the methods of decomposition, and its ability to extract sparse and easily interpretable factors.	NMF can be easily interpretable and relatively easy to understand. It has many applications across different fields.
A Tutorial in Spectral Clustering	(Von Luxburg, 2007)	The paper provides a detailed description of how spectral clustering works.	
AN ENHANCED RECOMMENDATION SCHEME FOR ONLINE GROCERY SHOPING	(Wu and Teng, 2011)	The authors developed a recommendation scheme for online grocery shopping that takes into account product replenishment and product promotion. The authors performed Item-Based CF using explicit ratings.	A well-detailed plan was proposed by the authors, as was a user interface. The methods employed cater to individual interests, product replenishment, and product promotion.
Recommendation system for grocery store considering data sparsity	(Sano et al., 2015)	The authors proposed two recommendation systems based on POS data. The recommendation methods include User-Based VF, SVD reconstruction, SVD similarity, CF and SVD, and NL-PCA. The dataset contained 8674 items and 6997 users. The authors estimated a user's evaluation value of a product by using an explicit scale of 0 to 5.	The models were evaluated using precision and recall. The precision value was about 0.0035, recall was 0.003, and the F1 score was 0.006. They also performed category recommendations as well as item recommendations by category.

Chapter 3: Data Understanding, Collection, and Preparation

According to the Cross Industry Standard Process for Data Mining (CRISP-DM), a data science project has a life cycle consisting of six phases, illustrated as:

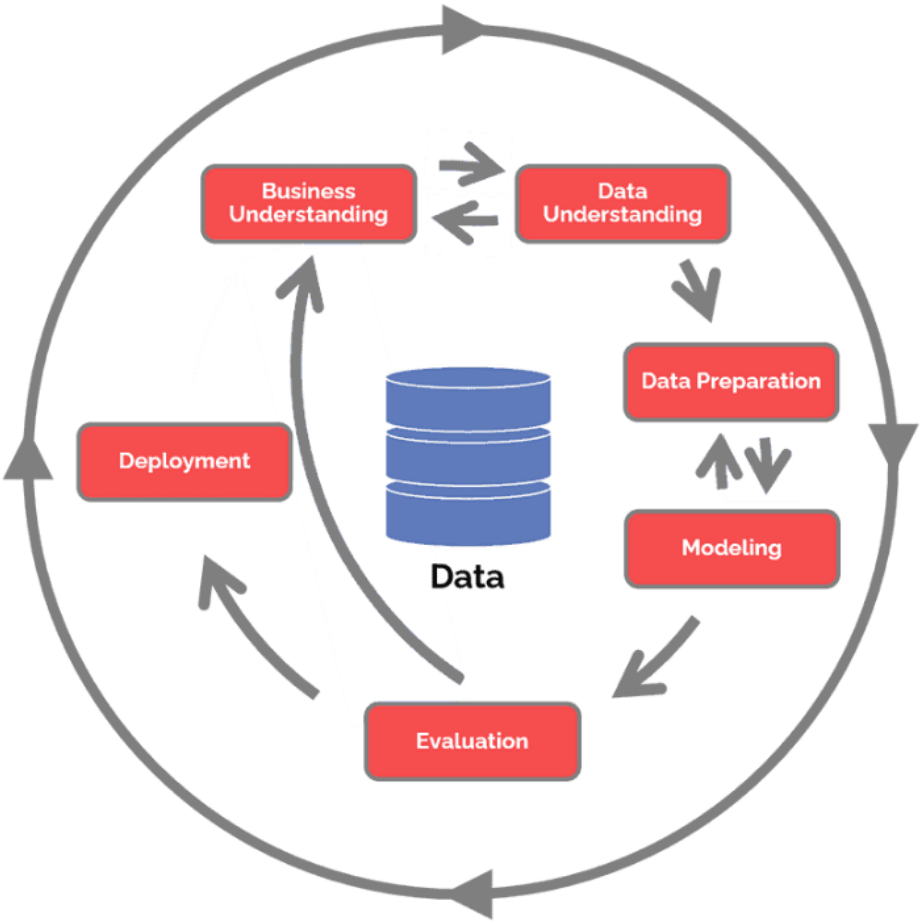


Figure 47: CRISP-DM Life Cycle (Hotz, 2018)

3.1 Business/Research Understanding

In Chapter 2, it has been established that a consumer's purchasing behaviour is driven by underlying motivations. While there may be consistent factors that drive consumer behaviour, the grocery shopping patterns of people from country to country differ. Consequently, data requirement gathering is important to assess and understand Mauritian consumer behaviour. Once the data is obtained and understood, datasets can be created. It is from these datasets that the appropriate models for prediction and recommendations can be selected. To make the most of the project, the models can be compared among each other, and the best-performing model is chosen for deployment.

3.2 Data Understanding

3.2.1 Data Requirements Gathering

Data requirement gathering is the process of identifying the specific data needs and expectations of the project. As the project scope involves building a recommendation system for groceries, the main data source was the different supermarket enterprises. The stakeholders involved in this project would be the supermarket and its customers. After engaging in a series of iterative and extensive dialogues with a senior executive representing a prominent supermarket brand, it was determined that procuring their proprietary data was implausible due to inherent restrictions and constraints.

Due to the aforementioned data unavailability, my alternative approach to gathering data for the project necessitated the implementation of surveys, which was also ineffective. Predominantly, the survey's lengthy nature discouraged respondents from granting their time towards its completion. Moreover, even among those who did respond, a substantial number of questions remained unanswered, thereby withholding crucial data required for the successful execution of the project.

Considering the previous challenges, the final possibility was conducting on-field data gathering, specifically acquiring supermarket receipts from customers within the supermarket premises.

3.2.2 Data Collection

As mentioned, to gather data, supermarket receipts were acquired within the supermarket premises. Collecting receipts from customers was a demanding process that relied on actively searching for customers. The Mauritian population seems to be more sympathetic to giving their receipts rather than spending a few minutes filling out a survey form. Another way to obtain the receipts was to retrieve them from dustbins or counters, as people tend to discard them as soon as they make their purchase. It was guaranteed that there would be people who would feel uncomfortable sharing their receipts, and no attempt was made to persuade them with respect to their choice. The receipts obtained were from three major supermarket enterprises, namely: Winners, Intermart, and Super U.

3.2.3 Data Extraction

The data obtained from the physical receipts presented a challenge as it required digitization for effective analysis and storage. To address this issue, two approaches were considered for digitization. The first involved using computer vision and optical character recognition (OCR) technology to automatically extract data from scanned receipts. However, this method proved to be ineffective due to difficulties in recognizing characters caused by faded ink and creases on the majority of the receipts. As an alternative, the data was manually inserted into an Excel file. This approach was time-consuming and required multiple checks to minimise input errors.

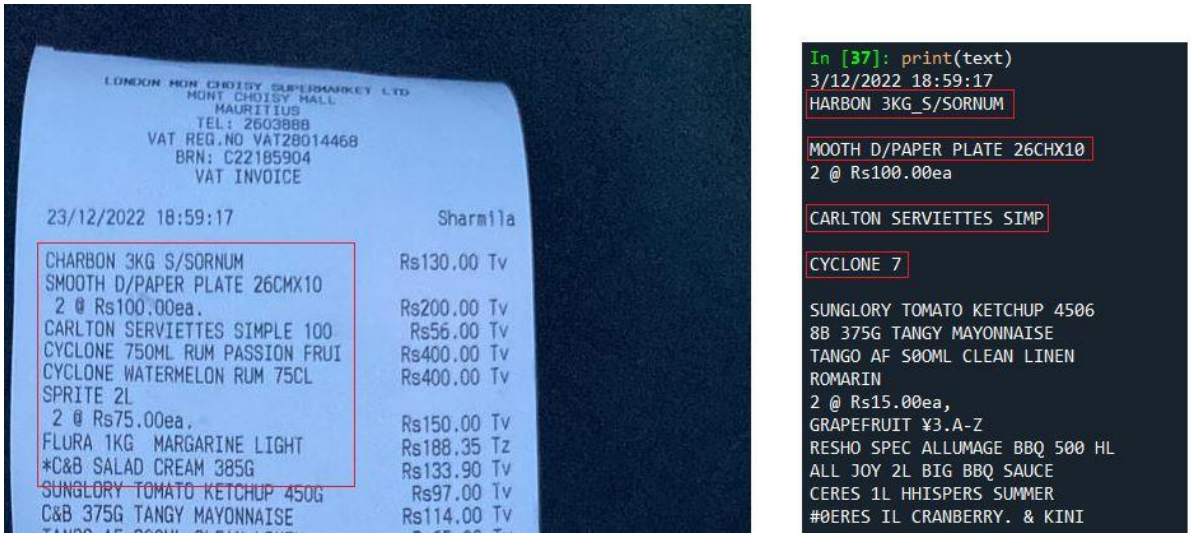


Figure 48: Data Extraction with OCR

Attribute extraction plays a vital role in the advancement of grocery item recommendation systems. It encompasses the identification and extraction of essential features or attributes from supermarket receipts. These attributes encompass crucial details such as products, quantity, price, and other pertinent information that contribute to making precise recommendations. Through the extraction and organisation of these attributes, the system becomes adept at recognizing similarities and patterns within the data, thereby enhancing the accuracy of recommendations. This process empowers the recommendation system to effectively analyse and leverage the extracted attributes, paving the way for more reliable and personalised grocery item suggestions. A major setback during the data extraction phase was realising that each company has a different naming convention for the same product. As a result, the dataset was built with only receipts from Winners.

The figure below provides a visual representation of the data collected from supermarket receipts, especially from the chosen supermarket franchise of winners.

winners

WINNERS VICTORIA URBAN TERMINAL
M1, PORT LOUIS
PICK AND BUY VICTORIA LTD
VAT:VAT27244473 / BRN:C13119825
Tel:214-6100 Fax:214-6196

------> DUPLICATA <-------
LOYALTY ID : 15823487

2>	LKS WHITE FLOUR 2KG	58.95 P
1>	NANDOS EX.EXT HT PE.	209.95 P
2>	BLUEBAND MARG.500G	92.85
2>	TR TUNA FL OIL 170G	33.95 P
2>	TR TUNA FL OIL 170G	33.95 P
2>	SELVA SEL 500G	18.95 P
1>	JULIES B.CRACK 250G	99.50
1>	TBRU FILA PECH180G	160.00
2>	PANINI PAT.GOLDEN	32.00
2>	PANINI PAT.GOLDEN	32.00
2>	M.GAJAK S.ROLL POISS	60.95 P
2>	M.GAJACK SAM.POISSON	60.95 P
1>	VAR LONGLIFE PO 4920	158.00
2>	CHIKO ROU VOL HONGRO	58.32
1>	NAVIGABLE ONION CRAC	49.00
2>	S.SHOP CATEAU TR.TRA	235.25
1>	TORCH LED	55.95 P
1>	MIKO MAG MN ALMOND 6	194.95 P
1>	MUNCHY LEXUS CH 150G	99.95 P
1>	WS WHOLE PEEL.TOMATO	36.25
2>	LUCKY STAR 155G PILC	37.55
1>	WS WHOLE PEEL.TOMATO	36.25
2>	TR TUNA FL OIL 170G	33.95 P
2>	HERITAGE PREMIUM BAS	349.95 P
2>	TWIN COW FCMP 1KG(FO	254.95 P
2>	KRAFT 250G CHEDDAR	81.95 P
2>	EGGSMORE LGE X 12	98.67

TOTAL 2674.94 Rs

27 Article(s)

MCB 2674.94 Rs

TVA incluse:

2	VATZRO	1575.14	0.00
1	VAT15%	1099.80	143.46

Ope	Mag.	Date	Heure	TPV	Ticket
109	28	19.02.23	12:43	9	90339

Join www.wiivrewards.com & earn rewards!

MERCI DE VOTRE VISITE
AU PLAISIR DE VOUS REVOIR

On est tous WINNERS

www.winners.mu

6600280090090339202302191241O1

Figure 49: Winner's receipt

During the attribute extraction process, various relevant pieces of information were identified and extracted from the raw data sources. These attributes play a crucial role in the recommendation system as they provide insights into the characteristics of each recommended item. The following Figure shows the specific attributes that have been extracted from the grocery receipts:

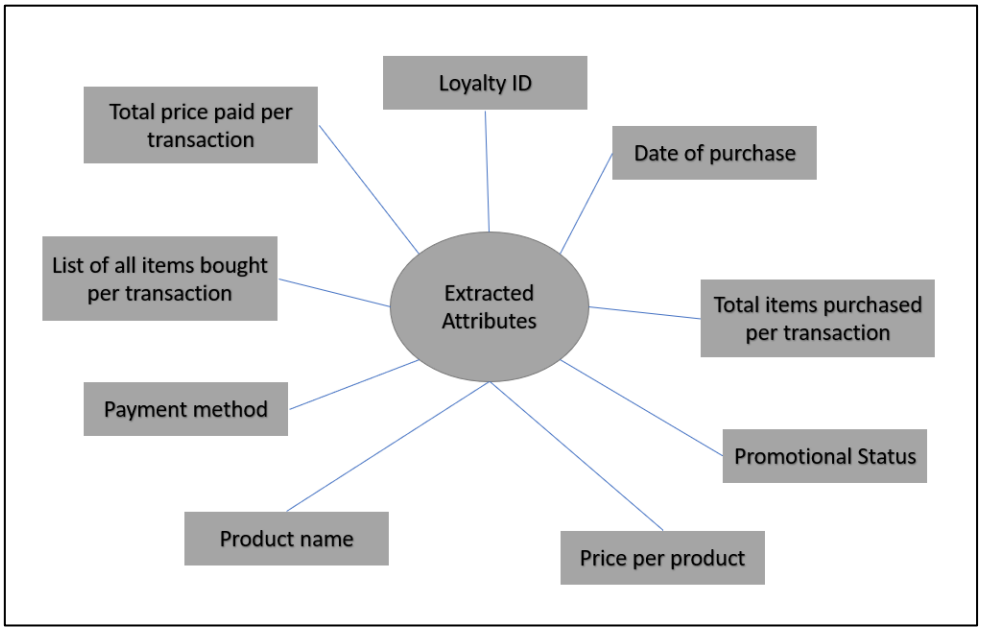


Figure 50: Attribute Extraction

1. Customer ID/Loyalty ID: A unique identifier assigned to each customer of the supermarket. It is used to track individual customers' purchase histories.

Figure 51: Customer ID

2. Date of purchase: The date when the transaction took place.

Figure 52: Transaction Date

3. Total items purchased per transaction: The number of items bought by a customer in a single transaction

Figure 53: Total items purchased

4. Total price paid per transaction: The total sum of money the customer paid in a single transaction.

Figure 54: Total price paid

5. List of all items bought per transaction: The list of all the products a customer purchased in a single transaction.

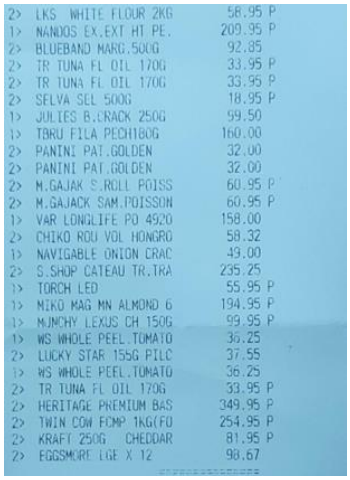


Figure 55: List of Items

6. Payment method: The mode of payment used by the customer to pay for their purchases.

Figure 56: Payment method

7. Product name: The name of the purchased product.

Figure 57: Product name

8. Price per product: The cost of a particular product.

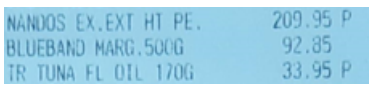


Figure 58: Product price

9. Promotional status: The promotional status of the product

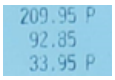


Figure 59: Promotional status

3.2.4 Data Description and Dataset

It is important to identify and describe the key variables to understand our dataset.

Variable	Description	Type	Example Values
Customer ID/Loyalty ID	Unique identifier assigned to each customer	Categorical	C18752486, C15887873, C18058183
Date of purchase	Date when the transaction took place	Categorical	18/02/2023, 19/02/2023, 12/03/2023
Total items purchased per transaction	Number of items bought by a customer in a single transaction	Numerical	194, 22, 36
Total price paid per transaction	Total sum of money paid by the customer in a single transaction	Numerical	1014.95, 498.84, 11403.91
List of all items bought per transaction	List of products purchased by the customer in a single transaction	Categorical	ESKO GAUFRET VAN 75G x 1 x 16.50EA, TIAS GRANOLA RAISIN x 1 x 238.50EA
Payment method	Mode of payment used by the customer	Categorical	SBM CARD, MCB CARD, CASH
Product name	Name of products purchased by the customer	Categorical	TWISTIES 20G CHEESE, MIRINDA 1.5L VANILLE, APOLLO CURRY 85G
Price per product	Individual price assigned to each item in the product list	Numerical	8.75, 53.00, 195.25
Promotional status	Indicates whether a product was purchased at a regular price or a discounted price	Categorical	(P) (indicating promotional price), Regular

Figure 60: Data Description

Transaction ID	Date	Customer ID	Items	Total Items	Total	Payment Method
69	19/02/2023		SUPER CROIX 3L BORA x 1 x 229.95EA(P), CANDIA UHT	56	9741.96	SBM CARD
70	18/02/2023		MONSTER ULTRA WHT 50 x 1 x 68.00EA, HARRYS AME	48	3467.26	SBM CARD
71	18/02/2023		PRUNE SACHET 500G SU x 1 x 109.95EA, KNORR SP C./	38	2734.72	CASH
72	18/02/2023		WS SUCRE TOP BL 2KG x 1 x 91.95EA(P), KING BRAND	57	4180.94	CASH
73	18/02/2023		BEBECALIN WIPES ECO x 1 x 72.95EA(P), YPT NAT.SUC	41	2990.51	MCB CARD
74	18/02/2023		BAGUETTE PARISIENNE x 2 x 9.90EA, CRYSTAL 1.5L x 6	34	2459.61	MCB CARD
75	18/02/2023		PANZANI 500G MACAR x 2 x 65.08EA, WS VEGETABLE	54	3105.71	MCB CARD
76	18/02/2023		SEARA CHICKEN FRANKS x 2 x 31.95EA(P), APOLLO CU	49	2512.06	CASH
77	18/02/2023		CIABATTA X 3 x 1 x 38.00EA, FRICO MILD EDAM BALL >	29	2753.85	MCB CARD
78	19/02/2023	C18752486	PHOENIX BEER CAN 50C x 2 x 77.00EA, SUPPLEMENT (	26	2239.81	MCB CARD
79	19/02/2023	C15887873	OIGNONS BLANC LOCAL x 1 x 33.90EA, GFOOD LAITUE	39	3897.62	MCB CARD
80	18/02/2023		CRYSTAL 1.5L x 6 x 24.00EA, SAC PAPIER CAISSE x 1 x 3	74	5856.94	MCB CARD
81	18/02/2023	C18058183	MIRINDA 1.5L ORANGE x 1 x 54.95EA(P), BAGUETTE P./	48	1896	CASH
82	18/02/2023	C16577462	SAC BIO CAISSE x 7 x 4.00EA, HOT DOG ROLLS 200G G	41	2899.17	MCB CARD
83	18/02/2023	C19866748	WHITE TOAST BREAD- S x 1 x 49.00EA, PANZANI 1KG N	77	5275.02	MCB CARD
84	18/02/2023	C14801419	WHITE TOAST BREAD- S x 1 x 33.00EA, C.COLA LESS SU	50	3183.33	MCB CARD
85	18/02/2023	C18547765	WS MIXED VEGETABLES x 1 x 49.95EA(P), 5 ALIVE PUL	60	3900.86	SBM CARD
86	18/02/2023		AQUAF BAD FLEX MED T x 1 x 177.00EA, SENSOD DEN	30	4918.5	CASH
87	18/02/2023		KILLTOX INSECT 300ML x 1 x 105.00EA, XTRA LESS LIQ	26	4033.6	CASH
88	19/02/2023		BOIS CHERI EXTR 500G x 1 x 210.00EA, TAMTAM CRAE	32	3432.34	MCB CARD
89	18/02/2023	C18603982	BAGUETTE PARISIENNE x 2 x 9.90EA, WATTIES MIXED	120	11612.95	SBM CARD
90	18/02/2023		XTRA LESS LIQ 3L TOT x 1 x 320.00EA, ROYAL BLCK ED	44	5444.29	CASH
91	18/02/2023		JEWEL NET MEN 1 25L x 1 x 123.95EA(P), DORITOS CH	34	2544.7	CASH
92	18/02/2023		FARMLAND FCMP 1KG x 2 x 261.95EA(P), SELVA SUCR	26	2225.62	CASH
93	18/02/2023		TWIN COW FCMP 1KG(FO x 1 x 254.95EA(P), RANI VEC	37	2977.05	CASH
94	19/02/2023	C18649657	COTE DOR NOIR NOISE x 3 X 125.95EA(P), BALA WHOI	47	4585.96	SBM CARD
95	18/02/2023	C14739580	VITAL 50CL x 1 x 14.00EA, BE DELI FRENCH G.HB x 2 x	50	2754.4	MCB CARD
96	18/02/2023		M.BURGERS 30GX10 x 1 x 49.00EA, OEUDOR MATINE	35	1856.46	SBM CARD
97	18/02/2023		YPL L.CAIL NAT 1L x 2 x 85.95EA(P), RANI VEGETABLE (	37	2834.7	CASH
98	18/02/2023		ROSALINDA 400G PEELE x 2 x 36.50EA, MIRINDA 2LTV	30	1990.05	CASH
99	18/02/2023	C18463858	PEPSI 1.5L x 1 x 54.95EA(P), FUZE TEA 50CL-PECHE x 2	41	2393.15	MCB CARD
100	18/02/2023	C15578186	BAGUETTE PARISIENNE x 2 x 9.90EA, SAMY.H.CHIC R.C	59	4057.7	MCB CARD
101	18/02/2023	C18740053	ORIENT LENTILLES ROU x 1 x 31.95EA(P), RED COW FA	37	2944.17	CASH
102	11/02/2023		LA TROP POT GL CAFÉ x 1 x 43.00EA, LA TROP SORBET	58	5412.3	MCB CARD

Figure 61: Raw Data

3.2.5 Data Pre-processing

After inserting each transaction into Excel, a cross-check is necessary to identify potential mistakes or typos. This tedious process ensures that the accuracy and integrity of the data are untarnished. All columns that contained blank cells were removed except for the Customer ID column. Eventually, the columns were converted into their respective data types.

3.2.6 Validity of the data.

Understanding the data at hand is key to making this project possible, and it is important to provide a description of why the data being used is legitimate. We've talked about consumer buying patterns in Chapter 2 and the factors that influence a person's decision to make a purchase. Since we are dealing with transactions, we are positively confident that the items on the list already suit the consumer behaviour factors pertaining to their respective customers. For example, a consumer who favours buying promotional items will have their interaction matrix consist exclusively of the promotional items without including any other items.

1	TRANSACTION ID	DATE PURCHASE	CUSTOMER ID	PRODUCT	QUANTITY	PRICE	PROMOTIONAL_STATUS
4512	207	12/03/2023	C15823487	POMME DE TERRE LOCAL	1	40.90	
4513	207	12/03/2023	C15823487	VALSPRING 1.5LT X 6	1	99.00	(P)
4514	207	12/03/2023	C15823487	FARMSTEAD SAUC.PLT M	1	74.00	
4515	207	12/03/2023	C15823487	MOROIL 1LT PURE SUN	2	97.00	
4516	207	12/03/2023	C15823487	PRIMA GARLIC SAUCE 2	2	30.50	
4517	208	12/03/2023		POMME ROUGE H.PACK 6	1	69.90	
4518	208	12/03/2023		MANDARINE SWEET C FI	1	169.90	
4519	208	12/03/2023		LUX SOAP 175G WAKE M	1	52.00	
4520	208	12/03/2023		CIDF BLC JAMB.BOUIL	1	99.80	
4521	208	12/03/2023		BANANES	1	61.29	
4522	208	12/03/2023		MIRINDA 1.5L VANILLE	2	54.95	(P)
4523	208	12/03/2023		TR TUNA FL OIL 170G	2	33.95	(P)
4524	208	12/03/2023		BF OIGNON ROUGE FILL	1	50.00	
4525	208	12/03/2023		MAYIL PRE MAIS 500G	1	27.95	(P)
4526	208	12/03/2023		CADBURY COCOA JAR 20	1	119.95	(P)
4527	208	12/03/2023		LMLC FARINE BLANCHE	1	57.95	(P)
4528	208	12/03/2023		I&J CHICKEN BURGERS	1	131.95	(P)
4529	208	12/03/2023		POMME D'AMOUR LOCAL	1	39.74	
4530	208	12/03/2023		BAGUETTE PARISIENNE	2	9.90	
4531	208	12/03/2023		SMIRNOFF ICE VODKA C	5	70.00	
4532	208	12/03/2023		TASTY SOAN PAPDI ELA	1	89.70	
4533	208	12/03/2023		BF OIGNON BLANC FILL	1	50.00	
4534	208	12/03/2023		POIVRON ROUGES	1	38.54	
4535	208	12/03/2023		MEADOW LEA ORIGINAL	2	100.88	
4536	209	12/03/2023	C18321842	HARIBO GOLDBAREN 80G	1	45.00	
4537	209	12/03/2023	C18321842	RED FEATHER BUTTER S	2	108.95	(P)
4538	209	12/03/2023	C18321842	PIMENT ECRASE 90G	1	34.90	
4539	209	12/03/2023	C18321842	OVALTINE JAR 200G	1	125.00	
4540	209	12/03/2023	C18321842	FR.LAIT 900G LAIT C	1	460.77	
4541	209	12/03/2023	C18321842	WATTIES SWEET CORN 4	1	110.95	(P)
4542	209	12/03/2023	C18321842	ALL IN ASSORTED BISC	1	70.00	
4543	209	12/03/2023	C18321842	PHOENIX BEER CAN 50C	2	77.00	

Figure 62: Pre-processed Data

3.2.7 Datasets

Apart from our collected data (Dataset 1), two additional datasets were used. Dataset 2 is generated from our collected data by randomly inserting rows and user-item transactions. The third dataset is a licensed dataset that contains a total of 1,048,575 transactions ("groceryData - dataset by rit-17," n.d.). The dataset used to make predictions is important. Selecting the appropriate data with a respectable time frame is necessary, as seasonal factors are factors that influence a customer's shopping behaviour.

Chapter 4: Analysis

The Analysis chapter delves into a comprehensive analysis of the collected data and the findings throughout the research process.

4.1 Data Analysis

4.1.1 Quantifying the Sales Volume for Promotional Items and Non-Promotional Items

After segregating promotional and non-promotional items from the purchase history, the data is grouped by products, and the total quantity purchased for each product is calculated. Ranks are then assigned to each product based on the quantity sold, with higher ranks indicating higher sales. This helps identify the most in-demand products while tackling cold-start issues.

Table 5 below shows the top 20 promotional products:

Table 5: Top 20 Promotional Products

PRODUCT	QUANTITY	RANK
APOLLO CURRY 85G	232	1.0
KRAFT 250G CHEDDAR	164	2.0
APOLLO CHICKEN 85G	110	3.0
TR TUNA FL OIL 170G	97	4.0
ROSSA W.PEELED TOMAT	71	5.0
CANDIA UHT D.EC 1LX6	61	6.0
FARMLAND FCMP 1KG	55	7.0
WS TUNA FLAKES IN OI	39	8.0
APOLLO VEGETABLE 85G	37	9.5
CANDIA UHT ENT 1LX6	37	9.5
LMLC FARINE BLANCHE	36	11.0
AQUA SPRING 1.5LT	33	12.0
WS VEGETABLE OIL BTL	31	13.0
PERETTE AMANDE 250ML	30	14.0
LP GALETTES BRETONNE	26	15.0
TRPICAL TUNA SLID IN	21	16.0
PERETTE VANILL 250ML	20	17.5
PEPSI TWIST 1.5LT	20	17.5
YPLAIT NATURE 125ML	19	19.5
YPT NAT.SUCRE 125ML	19	19.5

Table 6 below shows the top 20 non-promotional products.

Table 6: Top 20 Non-Promotional Products

PRODUCT	QUANTITY	RANK
SAC BIO CAISSE	103	1.0
BAGUETTE PARISIENNE	74	2.0
CRYSTAL 1.5L	65	3.0
SUPPLEMENT GLACE	58	4.0
PHOENIX BEER 650ML	45	5.0
BELINDA WH PEELED TO	43	6.0
PAIN BURGER X 4 PAT	34	7.0
QUEUE OIGNONBOT	33	8.0
PHOENIX BEER CAN 50C	32	9.0
FLUTE BLANC 100G	29	11.0
TWISTIES 20G CHEESE	29	11.0
WS WHOLE PEEL.TOMATO	29	11.0
RANI VEGETABLE OIL B	26	13.0
SMIRNOFF ICE VODKA C	24	15.0
PHOENIX BEER FRESH 3	24	15.0
JUTE SHOPPING BAG 45	24	15.0
PHOENIX BEER CAN 330	21	17.0
GLENRYCK 425G MACKER	20	18.0
YPLAIT PRUNEAU 125ML	19	19.5
LEADER SUNFL. OIL 1L	19	19.5

4.2 System Requirements

4.2.1 Functional Requirements

Table 7: Functional Requirements

Accuracy and Performance	FR1	Cluster Algorithm selection based on Silhouette Score, Calinski-Harabasz Index, and Davies-Bouldin Index.
	FR2	Assess the accuracy and performance of advanced collaborative filtering algorithms.
Recommendation Quality	FR3	Evaluation of recommendations model using Accuracy, Precision, Specificity, Sensitivity, and F1 Score.
Real-time processing	FR4	Real-Time recommendation system integration with updated database.
User Management	FR5	User Account Recreation.
	FR6	User Authentication.
Purchase Management	FR7	Shopping Cart Management.
	FR8	Update Database after Successful Purchase.

4.2.2 Non-functional Requirements

Table 8: Non-Functional Requirements

Usability and User Experience	NFR1	Intuitive Design and Responsive Interface
	NFR2	Clear Representation of Recommended Products
	NFR3	Seamless Integration with the Shopping Process
Privacy and Security	NFR4	User Account Security and Privacy
Scalability and Resource Requirements	NFR5	Model Computational Efficiency
Interpretability and Explainability	NFR6	Model design should be interpretable and explainable
Extensibility and Maintainability	NFR7	Model should be modular and easily extensible, allowing straightforward addition of new features

4.3 Comparison between existing E-commerce systems

Recommender systems have restructured users' approaches to online platforms. By providing personalised recommendations, these systems have amplified user satisfaction and engagement. In various industries, they play a pivotal role in guiding users through an expansive amount of content, making informed decisions, and finding new and relevant items tailored to their preferences. Consequently, companies do not tend to disclose their systems to gain a competitive advantage.

Table 9: Amazon vs. eBay

Criteria	Amazon	eBay
Data Analysis	- Collects general data about products and users. - Considers relationships and dependencies between products.	- Utilises user-clicked items and browsing activity
Recommendation Algorithms	Content-based filtering: - Recommends similar products based on user preferences. Collaborative filtering: - Uses experiences of other users to generate recommendations.	Deep learning and NLP methods: - Generates embeddings based on item and user entities. KNN search: - Finds relevant items recommendations.
Architecture and Processing	- Offline calculations for recommendation generation, with cached results for fast access.	- Phase 1: Offline processing and caching. - Phase 2: Near real-time KNN service. - Phase 3: Real-time processing using streaming technology and deep learning model inference.
Latency	- Offline processing introducing delay between user activity and recommendation generation.	- Phase 2 and Phase 3 approaches aim for near real-time and real-time processing, reducing latency.
Scalability	- Handles large-scale operations with millions of users and products.	- Handle high-volume traffic with millions of users and billion live listings.
Innovation	- Constantly explores new methods, including bandit-based algorithms and causal interference algorithms.	- Continuously working on developing and refining recommendation algorithms.
User Engagement	- Strives for personalisation and high-quality recommendations to enhance user experience.	- Prioritises relevant and diverse recommendations to align with user preferences and increase engagement.

4.4 Comparison between Winners, Super U, and Inter-Mart Websites

In Table 10 below, it is evident that online grocery shopping platforms in Mauritius are not well developed. This observation also reveals a huge opportunity for development and improvement, and this project is intended to address this gap and contribute to the progress of online grocery shopping in Mauritius.

Table 10: Comparing Mauritian Grocery Enterprises' Websites

	Winners	Intermart	Super U
Product Catalog and Diversity	- Products thoughtfully Categorized - Not all items are available	- Only brochure is available - No product representation on the website	- Products thoughtfully Categorized - Only Brand U products
User Interface (UI) and User Experience (UX)	- Needs improvement	- Poor	- Good
Recommendation Capabilities	- No personalized recommendations - No popular items recommendations - Recommendations on latest promotions	- None	- No personalized recommendations - No popular items recommendations - Recommendations on latest promotions
Recommendation Algorithms	- None	- None	- None
Data Collection and Privacy	- None	- None	- None
Accuracy and Relevance of Recommendations	- None	- None	- None
Personalization Options	- None	- None	- None
Feedback and User Interaction	- None	- None	- None
Integration with Shopping Process	- None	- None	- None
Machine Learning Model Explainability	- None	- None	-None

4.5 Proposed Solution

4.5.1 Research Questions and Solution Goals

The solution for the grocery recommendation system is driven by following these research questions and goals.

Research Questions:

1. How can we develop a grocery recommender system using the collected dataset that provides accurate and personalised recommendations for users in the Mauritian market?
2. How is the collaborative filtering algorithm implemented in the system?

3. What strategies can be employed to enhance the diversity and novelty of recommendations in the system?
4. How was the cold-start problem addressed for new users?

Solution Goals:

1. Generate personalised recommendations based on the user's item purchases.
2. The algorithm is implemented after using RFM analysis and clustering for enhanced accuracy and personalization of recommendations.
3. Enhance the diversity and novelty through popular item-based recommendations, item similarity, and generating recommendations by segmenting item categories.
4. New users are recommended to buy most promotional and non-promotional products.

4.5.2 Proposed system compared to existing E-commerce system

Table 11: Proposed System Architecture

Criteria	Proposed Solution
Data Analysis	- Utilises customer's purchase history
Recommendation Algorithms	Collaborative filtering: - Uses purchase of other customers to generate recommendations in both user and item perspective
Architecture and Processing	- Customers are segmented based on their monetary expenditure and purchase frequency and recency. - Algorithm is applied to customers within the same cluster.
Latency	- Real-time processing
Scalability	- Can handle new users and transactions
Innovation	- Able to expand the algorithm with new attributes such as product category
User Engagement	- Prioritise relevant recommendations aligned with users' purchase history to improve user experience

4.5.3 Proposed Solution Explanation

4.5.3.1 User based

In Chapter 3, we acquired our dataset, which consists of supermarket receipts. By meticulously analysing the data at hand, there were not so many options for making a recommendation system. In Chapter 2, we have studied and understood the principles of collaborative filtering as well as algorithms such as NMF, NCF, and SVD.

The proposed solution starts with grouping all the customers' transactions. For each customer, their total expenditure is calculated. Their total number of transactions is also accounted for, as is the recency of their latest purchase. This would provide an understanding of the customers buying patterns. Using the total expenditure, the total number of transactions, and the recency score of their latest purchase, the customers can be clustered into three different classes.

Once the customers have already been clustered, the next step is to predict the item purchases for a particular customer. To do so, the transactions for the target customer and his or her similar counterparts in the cluster are retrieved. This incorporates the concept of homophily. The term homophily is used to refer to people who associate with similar others (McPherson et al., 2001). By comparing customer transaction histories, we followed the logic of "buying what people like you buy" (Lavelle-Hill et al., 2020). By making use of one-hot encoding, the items are then converted into classes. The underlying goal is to create a user-item matrix that would indicate what item each customer purchased.

After obtaining the binary user-item interaction matrix, it is split into training and testing sets. To predict the purchases for the target user, it is imperative that the target user's transactions be put into the testing set. The advanced algorithm is then applied to the training set, which would create the user' latent factors and the item latent factors. A transformation is then applied to the testing set to retrieve only the user's latent factors. A dot product is then performed with the user latent factors from the testing set and the item latent factors from the training set to create a prediction matrix. When the prediction matrix is obtained, the highest k-rated columns (items) for each row (customer) can be used as recommendations.

4.5.3.2 Item-based

Making recommendations based on items is fairly straightforward. Only the transactions for a specific product are selected. The logic is to make predictions based on items that are similar to the ones a customer is showing interest in. Once the transactions are sorted, the same principle of collaborative filtering is applied.

4.6 Programming Language Analysis

Python	R	Java
Pros: • Extensive Libraries • Readability and Ease of Use • Integration with Web Frameworks Cons: • Performance • Memory Consumption	Pros: • Powerful Statistical Packages • Data Visualisation Cons: • Learning Curve • Performance • Versatility	Pros: • Performance • Platform Independence • Scalability Cons: • Longer development period • Learning Curve • Few Specialised Libraries
Best Choice: Python		

4.7 IDE Analysis

Visual Studio Code	Google Colab	IntelliJ IDEA	RStudio
• Wide range of languages • Highly Customisable • Lightweight • Responsive Nature	• Cloud-Based • Easy to learn • Pre-installed Libraries • Sharing and Collaboration • Integration with Google Drive	• Powerful refactoring tools • Extensive set of features • Smart coding assistance	• Data analysis • Visualisation
Best Choice: Google Colab (Training and Testing) & Visual Studio Code (Web Application)			

4.8 Use Case

4.8.1 Use Case Description

Name:	Personalized Grocery Recommendations
Participating Actor:	User
Pre-condition:	1. The user is a registered user of the system. 2. The user has already logged into the system successfully in the past. 3. The system has the user's purchase history. 4. The system has already classify the user into a specific cluster.
Flow of events:	1. The user try to log into the system. 2. The user inserts his/her email address and password. 3. The system validates the users credentials. 4. The system welcomes the user in the homepage. 5. The user can view all the products available in the system. 6. The user can look up the details of a particular item by clicking on that item. 7. The system takes the user to the item page. 8. The system retrieves all the transactions that contain this particular item. 9. The system applies the Collaborative Filtering algorithm to generate recommendations based on item similarity. 10. The user can add to cart or proceed to his account page. 11. The system takes the user to his account page. 12. The system retrieves the user's customer id from the database. 13. The system applies the Collaborative Filtering algorithm to generate recommendations based on user similarity. 14. The system presents a list of personalized grocery recommendations to the user on his account page. 15. The user explores the recommended grocery items, and adds the desired items to their cart. 16. The user proceeds to checkout, makes the necessary payments, and completes the purchase. 17. The use case ends.

Figure 63: Use Case Description 1

Alternative flows:	A. Invalid Credentials
	If in step 3, the system determines that the user's credentials are incorrect, then
	1. The system ask the user to try again.
	2. The use case resumes at step 2.
	B. Un-registered user
	If in step 2, the user does not have an account, then
	1. The system will ask the user to create an account.
	2. The user provides his/her first name, last name, username, email address, phone number, password.
	3. The system will allocate a customer id to the user
	4. The system will allocate the user to the lowest cluster.
	5. The system will ask the user to log in.
	6. The use case resumes at step 1.

Post-conditions:	The user receives personalized grocery recommendations based on their purchase history and similarities with other users in the same clusters.

Figure 64: Use Case Description 2

4.8.2 Use Case Diagram

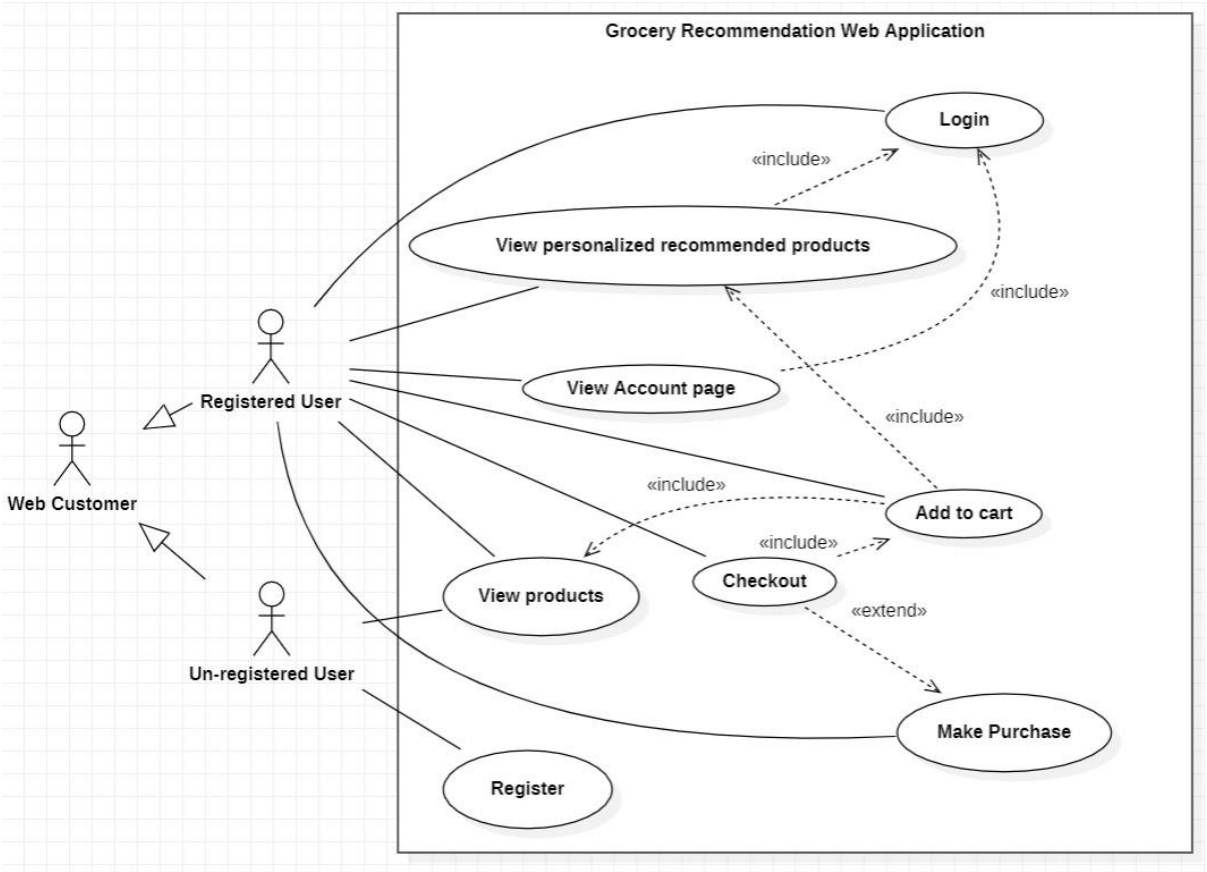


Figure 65: Use Case Diagram

Chapter 5: Design

The proposed solution generated in the Analysis chapter is used to produce a comprehensive software design. The design phase focuses on the creation of the recommender system and the web application.

5.1 Recommender System: User-Based Activity Diagram

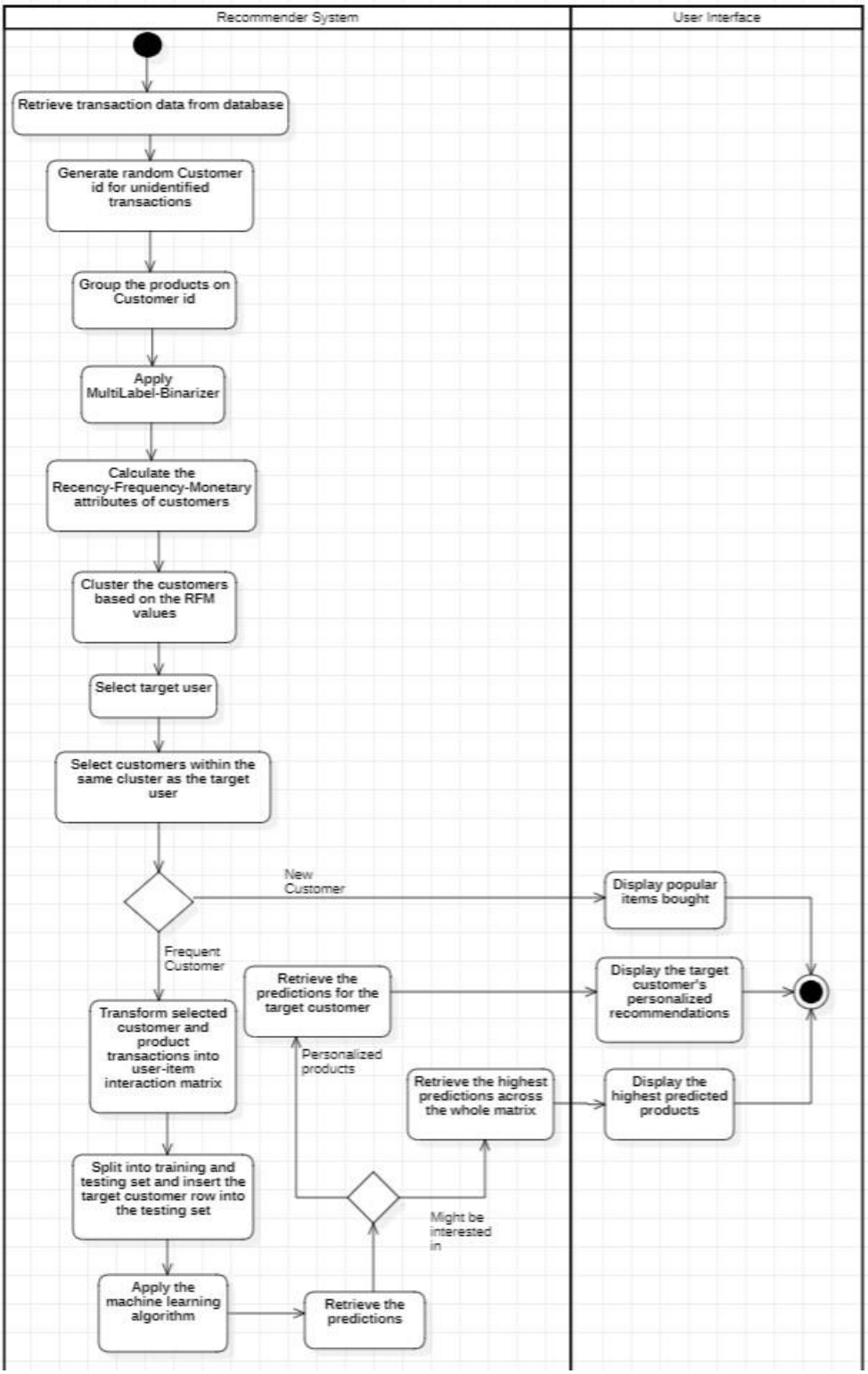


Figure 66: User-Based Recommendation Activity Diagram

5.2 Recommender System: Item-Based Activity Diagram

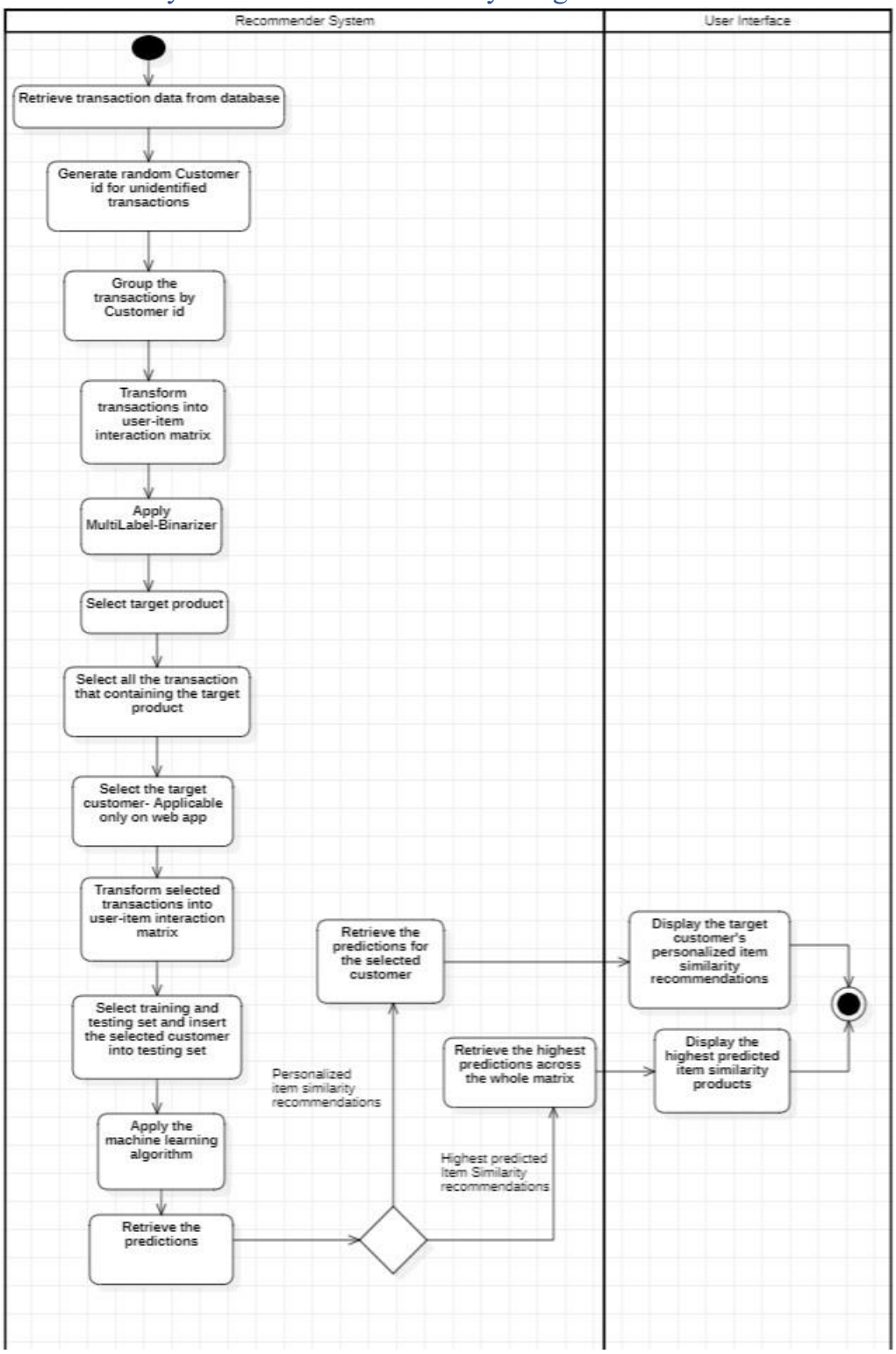


Figure 67: Item-Based Recommendation Activity Diagram

5.3 User-Based Recommendations Data Flow Diagram

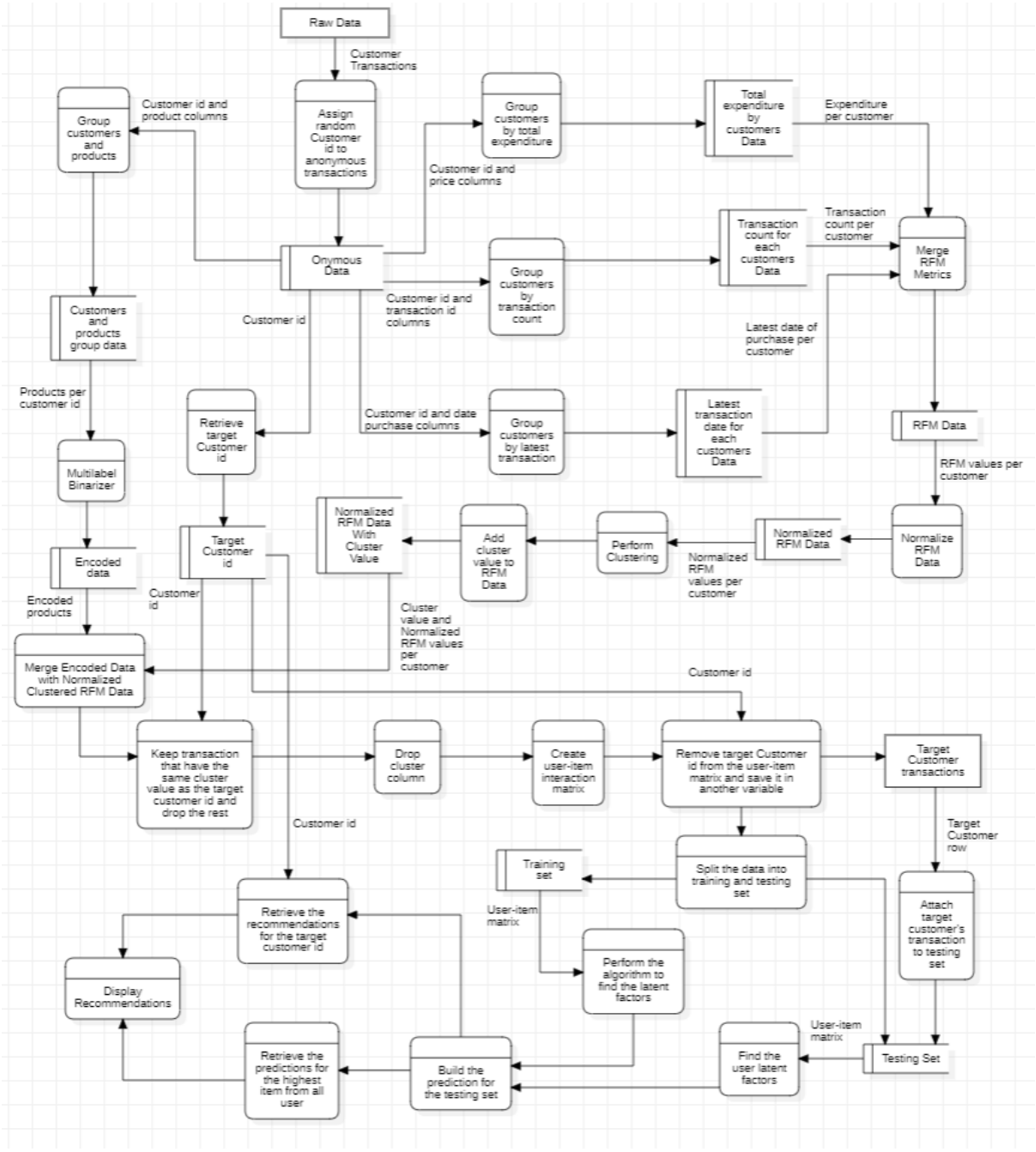


Figure 68: User-Based Recommendations Data Flow Diagram

5.4 Item-Based Recommendations Data Flow Diagram

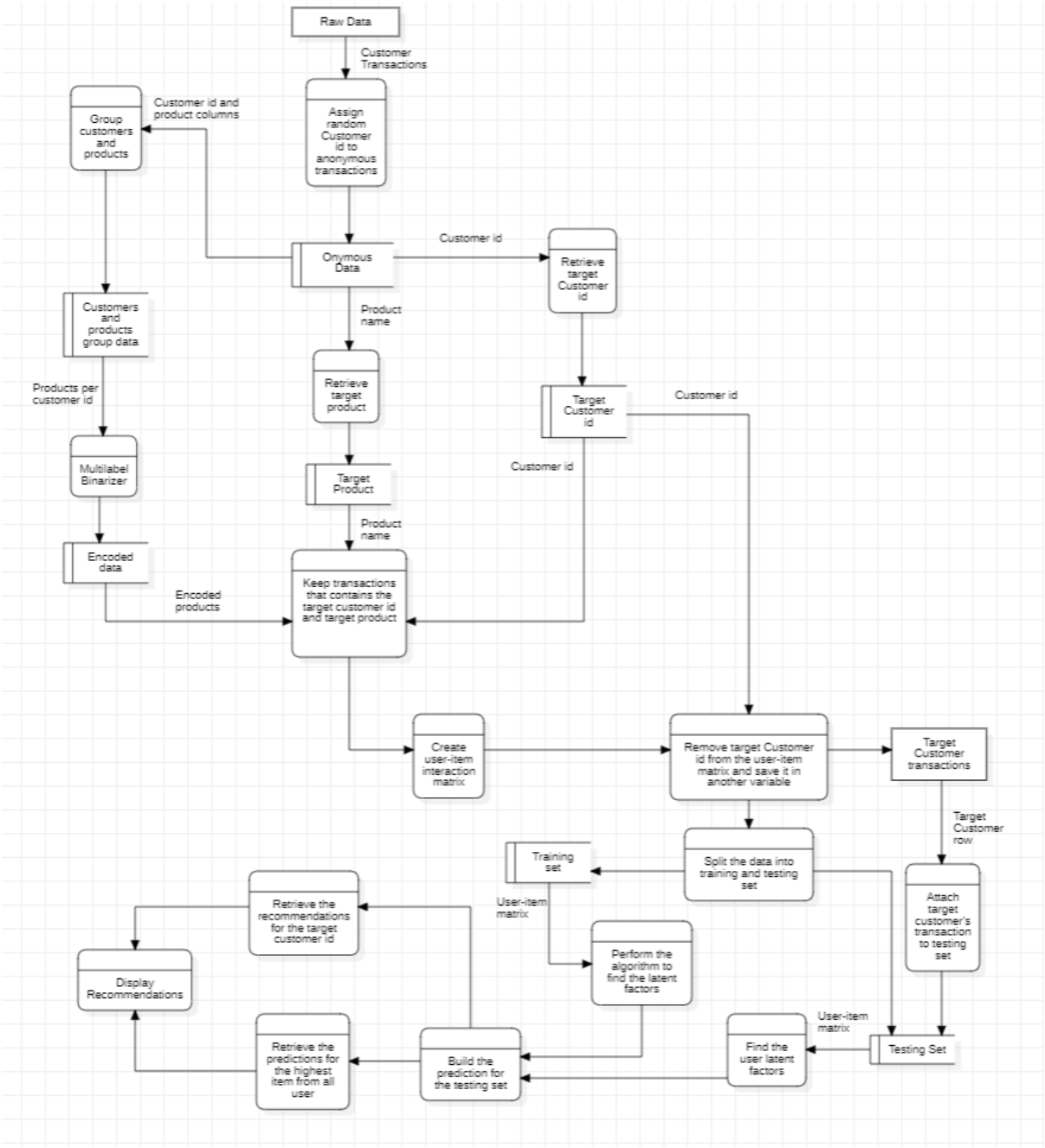


Figure 69: Item-Based Recommendation Data Flow Diagram

5.5 Web App Architecture Diagram

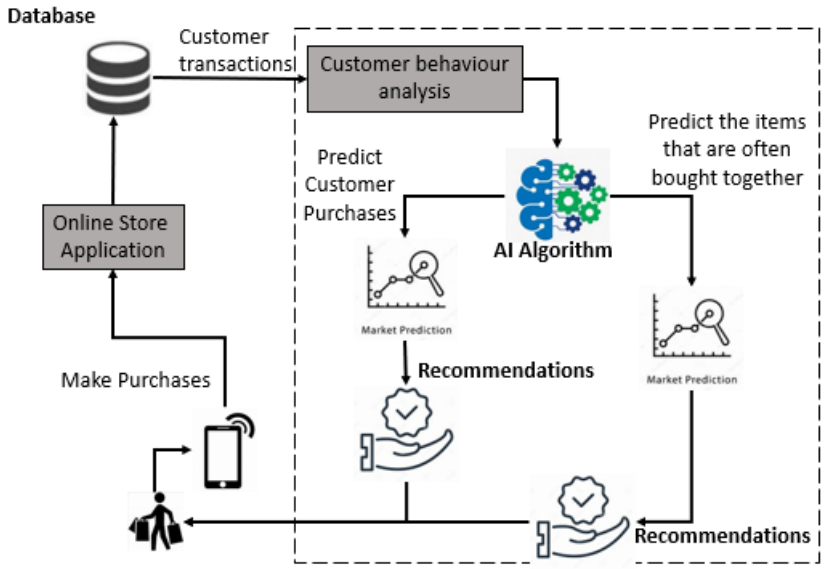


Figure 70: Recommender System Architecture Diagram

5.6 Web Application Recommender System Integration Flowchart

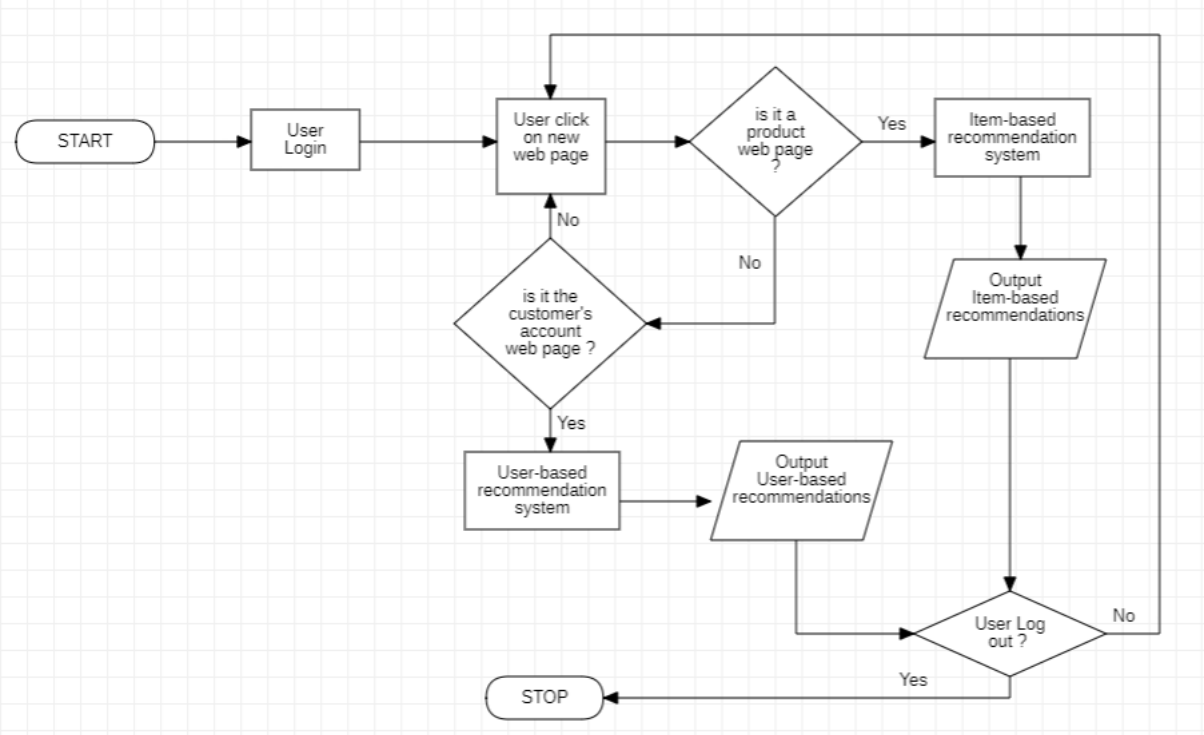


Figure 71: Recommender System Flowchart

5.7 Sequence Diagram

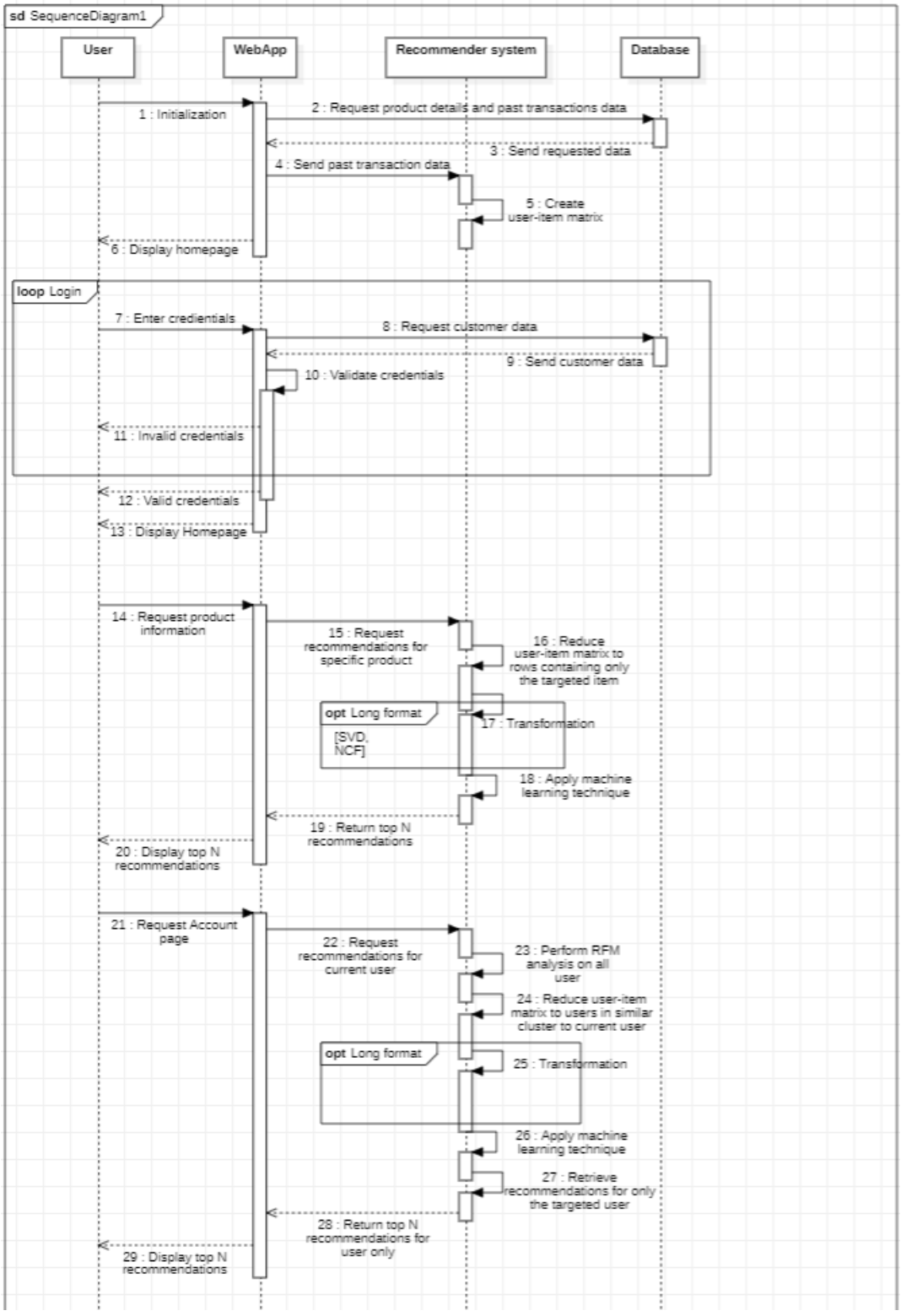


Figure 72: Recommender System Sequence Diagram

5.8 Interface Design

5.8.1 User Registration

The new user fills out a form to register.

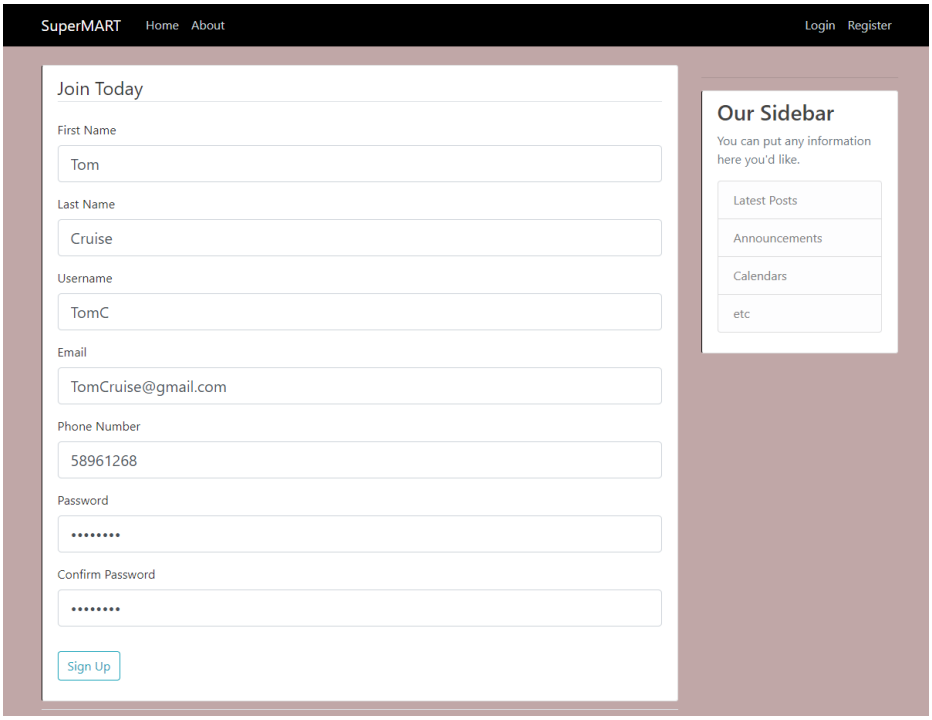


Figure 73: Registration form

The newly registered user is shown a welcome message and is directed to the login page.

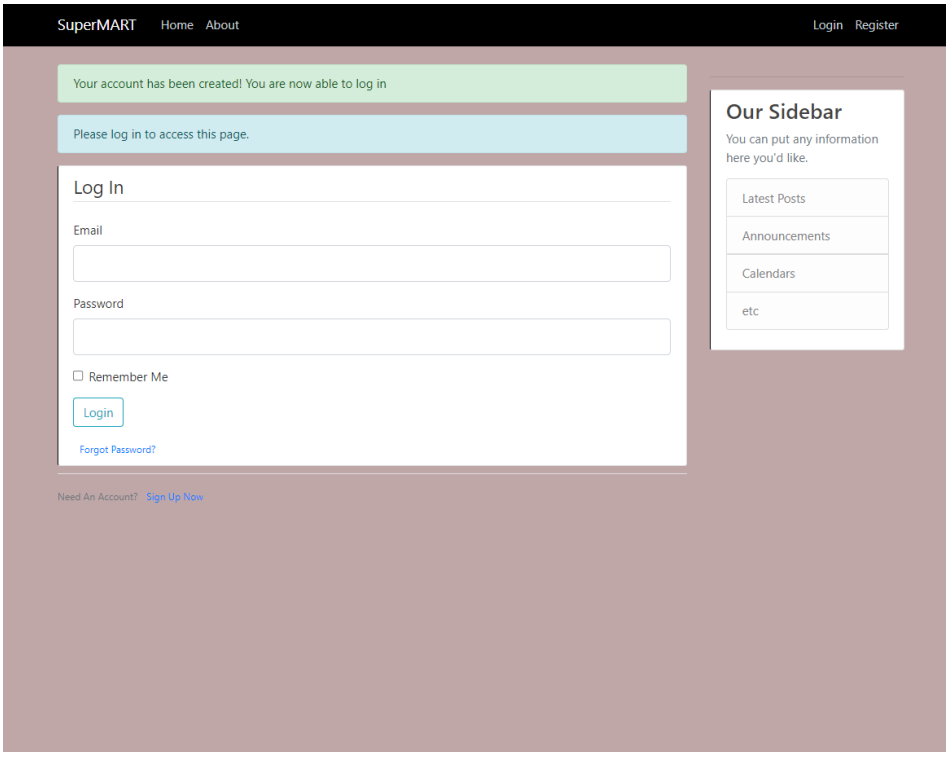


Figure 74: Successful Registration

If the credentials are already taken by another user, a warning is sent.

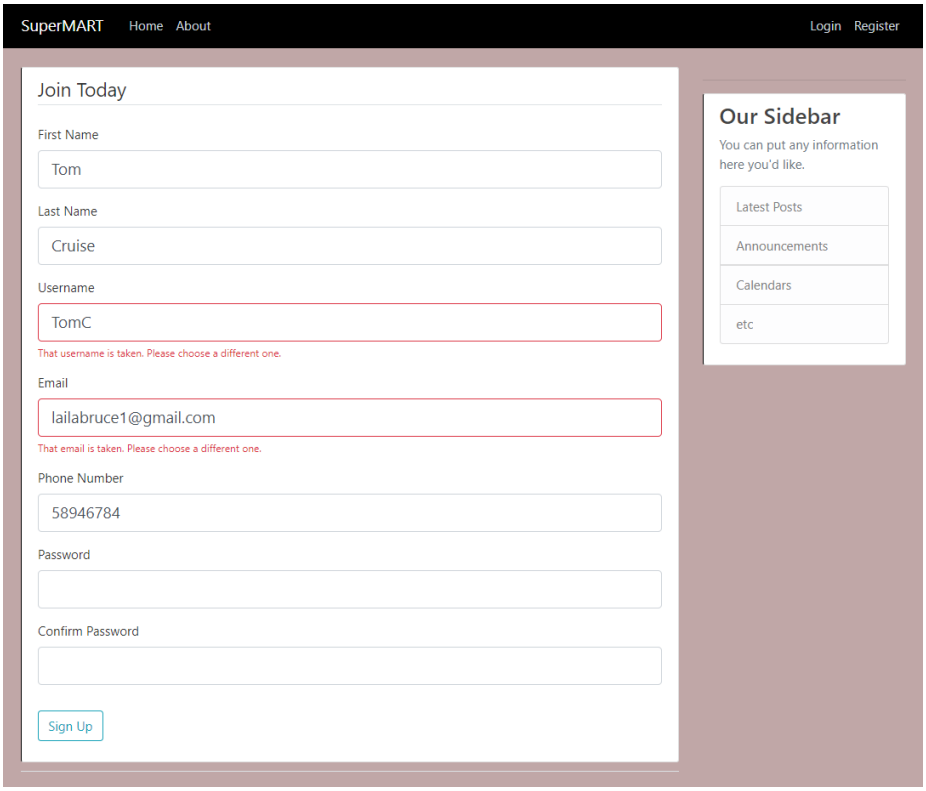


Figure 75: Unsuccessful Registration

5.8.2 User Login

To login, only the email address and password are required.

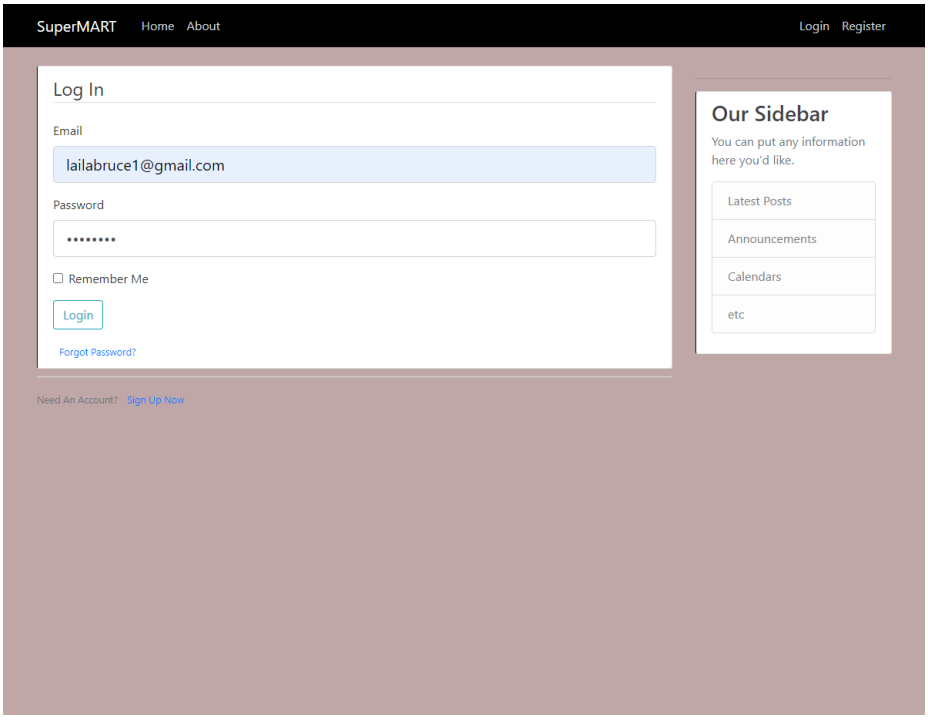


Figure 76: Login Form

Chapter 6: Implementation

This section provides an overview of the implementation steps for the proposed recommender system.

The following data presents the flow of the implementation process.

6.1 Environment Setup

The project was executed on a window-based system. The specifications used are as follows:

Hardware Specification
• ASUS VivaBook • Intel(R) Core(TM) i5-1035G1 CPU @ 1.00GHz 1.19Gz • 8.00 GB RAM
Software Specification
• Windows 11 Home 64-bit • Google Colab • Visual Studio Code • Wampserver64 • Excel

Figure 77: Environments

Libraries Used
• Pandas • Numpy • Scikit-learn • Datetime • Surprise • Keras • TensorFlow

Figure 78: Libraries

6.2 Dataset pre-processing and preparation

In Chapter 2, it has been established that collaborative filtering is separated into user-based collaborative filtering and item-based collaborative filtering. Consequently, these approaches require separate treatment. User-based collaborative filtering utilises RFM analysis and clustering to group users with similar preferences, while item-based collaborative filtering requires a less subtle approach. When the desired number of recommendations is specified, there are two ways to provide the customer with recommendations. The first method involves obtaining the top-K recommended items for each user, and the second method focuses on identifying the top items with the highest predicted ratings.

```
# Loading the dataset on Colab
df = pd.read_csv("transaction.csv", encoding="ISO-8859-1")

# Remove leading/trailing spaces from the "CUSTOMER_ID" and "PRODUCT"
column
df["CUSTOMER_ID"] = df["CUSTOMER_ID"].str.strip()
df["PRODUCT"] = df["PRODUCT"].str.strip()
```

Figure 79: Loading Dataset

```
      ï»¿INDEX_ID  TRANSACTION_ID DATE_PURCHASE CUSTOMER_ID  \
0               1               6    16/02/2023         NaN
1               2               6    16/02/2023         NaN
2               3              14    19/01/2023         NaN
3               4              14    19/01/2023         NaN
4               5              15    19/01/2023         NaN
...           ...             ...           ...         ...
5251         5252             253    22/03/2023         NaN
5252         5253             253    22/03/2023         NaN
5253         5254             253    22/03/2023         NaN
5254         5255             253    22/03/2023         NaN
5255         5256             253    22/03/2023         NaN

                  PRODUCT  QUANTITY   PRICE PROMOTIONAL_STATUS
0     ESKO GAUFRET VAN 75G         1   16.50                NaN
1      TIAS GRANOLA RAISIN         1  238.50                NaN
2          KIT KAT 4F DARK         2   31.00                NaN
3      ERASERS X6PCS 8 ASS         1   45.00                NaN
4     BREAD CRUMBS CHAPELU         2   29.00                NaN
...                    ...       ...     ...                ...
5251  COLG MW PL ICE 500ML         1  215.00                NaN
5252  MIEL OR POLYFLORAL S         1  219.95                (P)
5253   DODO FISH BALLS 1KG         1  261.00                NaN
5254   AVON CITRON PLUS 1K         1  168.95                (P)
5255  CTECLER POULET ENTIE         1  207.79                (P)

[5256 rows x 8 columns]
```

Figure 80: df DataFrame

The code loads a dataset from a CSV format into a Pandas DataFrame using the ISO-8859-1 encoding to interpret the file correctly. It then removes leading and trailing spaces from the “CUSTOMER_ID” and “PRODUCT” columns, ensuring data consistency and eliminating potential issues caused by white characters.

```
# Generate random values
random_values = ['C' + str(np.random.randint(100, 999999)) for _ in
range(df['TRANSACTION_ID'].nunique())]

# Assign random values to NaN customer IDs based on TRANSACTION_ID
df.loc[df['CUSTOMER_ID'].isnull(), 'CUSTOMER_ID'] =
df.loc[df['CUSTOMER_ID'].isnull(),
'TRANSACTION_ID'].map(dict(zip(df['TRANSACTION_ID'].unique(),
random_values)))
```

Figure 81: Assigning customer IDs to anonymous transactions

Random values, which will have “C” followed by a random number, are generated. These values are then assigned to the “CUSTOMER_ID” column, where values were previously missing based on the corresponding “TRANSACTION_ID” values. This approach enables us to utilise the transactions that were originally anonymous.

	ï»¿INDEX_ID	TRANSACTION_ID	DATE_PURCHASE	CUSTOMER_ID	\
0	1	6	16/02/2023	C692030	
1	2	6	16/02/2023	C692030	
2	3	14	19/01/2023	C224110	
3	4	14	19/01/2023	C224110	
4	5	15	19/01/2023	C487418	
...	...	...	...	...	
5251	5252	253	22/03/2023	C285157	
5252	5253	253	22/03/2023	C285157	
5253	5254	253	22/03/2023	C285157	
5254	5255	253	22/03/2023	C285157	
5255	5256	253	22/03/2023	C285157	

	PRODUCT	QUANTITY	PRICE	PROMOTIONAL_STATUS
0	ESKO GAUFRET VAN 75G	1	16.50	NaN
1	TIAS GRANOLA RAISIN	1	238.50	NaN
2	KIT KAT 4F DARK	2	31.00	NaN
3	ERASERS X6PCS 8 ASS	1	45.00	NaN
4	BREAD CRUMBS CHAPELU	2	29.00	NaN
...	...	...	...	...
5251	COLG MW PL ICE 500ML	1	215.00	NaN
5252	MIEL OR POLYFLORAL S	1	219.95	(P)
5253	DODO FISH BALLS 1KG	1	261.00	NaN
5254	AVON CITRON PLUS 1K	1	168.95	(P)
5255	CTECLER POULET ENTIE	1	207.79	(P)

[5256 rows x 8 columns]

Figure 82: Onymous df DataFrame

6.3 MultiLabelBinarizer

```
grouped_df =
df.groupby('CUSTOMER_ID')['PRODUCT'].agg(list).reset_index()

# Create an instance of the MultiLabelBinarizer
mlb = MultiLabelBinarizer()

# Apply one-hot encoding on the "PRODUCT" column
encoded_products = mlb.fit_transform(grouped_df['PRODUCT'])

# Create a new DataFrame with the encoded products
encoded_df = pd.DataFrame(encoded_products, columns=mlb.classes_)

# Concatenate the "CUSTOMER_ID" column with the encoded DataFrame
result_df = pd.concat([grouped_df['CUSTOMER_ID'], encoded_df], axis=1)
```

Figure 83: MultiLabel Binarizer

Once the DataFrame is grouped by 'CUSTOMER_ID' and 'PRODUCTS', the instance of the 'MultiLabelBinarizer' class is created. One-hot encoding is applied to the 'PRODUCT' column using the 'fit_transform()' method. The result is stored in the 'encoded_products' variable and assigned as column names for the 'encoded_df' DataFrame. By concatenating the 'CUSTOMER_ID' with 'encoded_df' DataFrame, a new 'result_df' DataFrame is created.

	CUSTOMER_ID	FUMAKILLA VAPE MOSQU	MALTA GUINNESS CAN 3	\
0	C104942	0	0	
1	C11471187	0	0	
2	C11486781	0	0	
3	C11844454	0	0	
4	C119515	0	0	
..	...	...	...	
188	C976176	0	0	
189	C980299	0	0	
190	C989419	0	0	
191	C991892	0	0	
192	C997932	0	0	

Figure 84: result df DataFrame

6.4 User-based data pre-processing and preparation

To address User-Based CF, it is imperative to have a designated target user. With regards to the web application, the designated target user will correspond to the individual who logs into their respective account.

6.4.1 RFM Analysis

```
# Calculate the sum of 'PRICE' column for each 'CUSTOMER_ID'
monetary = df.groupby('CUSTOMER_ID')['PRICE'].sum()
monetary = monetary.reset_index()

# Group the DataFrame by 'TRANSACTION_ID' and 'CUSTOMER_ID', and count
the occurrences of 'PRODUCT'
x_df = df.groupby(['TRANSACTION_ID', 'CUSTOMER_ID']).agg({'PRODUCT':
'count'}).reset_index()

# Calculate the count of 'TRANSACTION_ID' for each 'CUSTOMER_ID'
frequency = x_df.groupby('CUSTOMER_ID')['TRANSACTION_ID'].count()
frequency = frequency.reset_index()

# Convert 'DATE_PURCHASE' column to datetime format
df['DATE_PURCHASE'] = pd.to_datetime(df['DATE_PURCHASE'],
format="%d/%m/%Y")

# Calculate the difference between the maximum date and 'DATE_PURCHASE'
for each 'CUSTOMER_ID'
df['Diff'] = max(df['DATE_PURCHASE']) - df['DATE_PURCHASE']

# Calculate the minimum difference in days (recency) for each
'CUSTOMER_ID'
recency = df.groupby('CUSTOMER_ID')['Diff'].min()
recency = recency.reset_index()
recency['Diff'] = -recency['Diff'].dt.days

# Merge the 'recency', 'frequency', and 'monetary' DataFrames based on
'CUSTOMER_ID'
rfm = pd.merge(recency, frequency, on='CUSTOMER_ID', how='inner')
rfm = pd.merge(rfm, monetary, on='CUSTOMER_ID', how='inner')

# Rename the columns in the 'rfm' DataFrame
rfm.columns = ['CUSTOMER_ID', 'Recency', 'Frequency', 'Monetary']
```

Figure 85: RFM Analysis

	CUSTOMER_ID	PRICE
0	C11471187	4024.82
1	C11486781	4580.44
2	C11844454	5009.01
3	C12126090	4075.80
4	C12197918	5122.86
..	...	...
188	C953184	2428.80
189	C958658	2178.85
190	C965263	1057.84
191	C976443	1344.60
192	C984339	1685.90

Figure 86: Monetary DataFrame

	CUSTOMER_ID	TRANSACTION_ID
0	C11471187	2
1	C11486781	2
2	C11844454	2
3	C12126090	2
4	C12197918	2
..	...	...
188	C953184	1
189	C958658	1
190	C965263	1
191	C976443	1
192	C984339	1

Figure 87: Frequency DataFrame

	CUSTOMER_ID	Diff
0	C108600	-32
1	C11471187	-10
2	C11486781	-5
3	C11844454	-10
4	C12126090	-10
..	...	...
188	C984149	-16
189	C984194	-50
190	C992778	-32
191	C994362	-23
192	C999251	-23

Figure 88: Recency DataFrame

The monetary section calculates the corresponding total sum of prices for each 'CUSTOMER_ID', and the frequency section groups 'TRANSACTION_ID' on 'CUSTOMER_ID', counting the transaction occurrences. The recency section calculates the latest date of purchase for a customer by converting the 'DATE_PURCHASE' column to datetime format. Finally, the DataFrames 'recency', 'frequency', and 'monetary' are combined on the 'CUSTOMER_ID' into a single 'rfm' DataFrame.

	CUSTOMER_ID	Recency	Frequency	Monetary
0	C108600	-32	1	1837.83
1	C11471187	-10	2	4024.82
2	C11486781	-5	2	4580.44
3	C11844454	-10	2	5009.01
4	C12126090	-10	2	4075.80
..	...	...	...	...
188	C984149	-16	1	527.85
189	C984194	-50	1	1330.30
190	C992778	-32	1	7587.31
191	C994362	-23	1	845.65
192	C999251	-23	1	553.60

Figure 89: rfm DataFrame

```
# Initialize a StandardScaler instance for normalization
scaler = StandardScaler()

# Select the columns to be normalized and transform them
rfm_normalized = rfm[['Recency', 'Frequency', 'Monetary']]
rfm_normalized = scaler.fit_transform(rfm_normalized)

# Convert the normalized array back to a DataFrame
rfm_normalized = pd.DataFrame(rfm_normalized)
rfm_normalized.columns = ['Recency', 'Frequency', 'Monetary']
```

Figure 90: Normalisation

Using the 'StandardScaler' class from the 'sklearn.preprocessing' module, feature normalisation (feature scaling) is performed. This transforms the variables from the "rfm" DataFrame to have a similar distribution by removing the mean and scaling to unit variance. Feature normalisation is mostly applied to avoid bias in the model, improve convergence, equalise feature importance, interpret feature importance, and handle numerical instability. The new 'rfm_normalized' DataFrame then contains the normalised values for 'monetary', 'recency', and 'frequency'.

	Recency	Frequency	Monetary
0	1.284458	-0.511310	-0.400832
1	-1.494297	-0.511310	-0.313517
2	-1.494297	-0.511310	-0.941325
3	0.870601	1.955761	0.781091
4	1.166213	1.955761	1.051876
..	...	...	...
188	1.284458	-0.511310	-0.516165
189	0.456744	-0.511310	-1.056936
190	-0.370970	-0.511310	1.626134
191	-0.962195	-0.511310	-0.446809
192	-0.370970	-0.511310	0.295528

Figure 91: rfm-normalized DataFrame

6.4.2 Customer Segmentation

Customer segmentation involves dividing the customer base into distinct groups based on shared characteristics and behaviours. Three clustering techniques were evaluated using the intrinsic evaluation measures mentioned in Appendix 3, we identified the best-performing algorithm for our dataset. The choice of having three clusters was based on classifying customers into three distinct categories: low-, medium-, and high-value customers. The evaluation values obtained are thoroughly analysed and discussed in Chapter 7.

6.4.2.1 Spectral Clustering

```
# Spectral Clustering
spectral = SpectralClustering(n_clusters=3)
spectral.fit(rfm_normalized)
```

Figure 92: Spectral Clustering

```
#Evaluation Metrics
silhouette_spectral = silhouette_score(rfm_normalized,
spectral.labels_)
calinski_spectral = calinski_harabasz_score(rfm_normalized,
spectral.labels_)
davies_bouldin_spectral = davies_bouldin_score(rfm_normalized,
spectral.labels_)
```

Figure 93: Spectral Evaluation

```
Spectral Clustering:
Silhouette Score: 0.5123836771939313
Calinski-Harabasz Index: 214.625126991695
Davies-Bouldin Index 0.7091682963197171
```

Figure 94: Spectral Example Output

An instance of the Spectral clustering algorithm with three desired clusters is created. The clustering is then applied to the "rfm_normalized" DataFrame which performs the clustering process and assigns cluster labels to each data point. The silhouette score, Calinski-Harabasz index, and Davies-Bouldin index are calculated to assess the quality of the clustering obtained.

6.4.2.2 Hierarchical Clustering

```
# Hierarchical Clustering
hierarchical = AgglomerativeClustering(n_clusters=3,
linkage='average').fit(rfm_normalized)
```

Figure 95: Hierarchical Clustering

```
#Evaluation Metrics
silhouette_hierarchical = silhouette_score(rfm_normalized,
hierarchical.labels_)
calinski_hierarchical = calinski_harabasz_score(rfm_normalized,
hierarchical.labels_)
davies_bouldin_hierarchical = davies_bouldin_score(rfm_normalized,
hierarchical.labels_)
```

Figure 96: Hierarchical Evaluation

```
Hierarchical Clustering:
Silhouette Score: 0.5123836771939313
Calinski-Harabasz Index: 214.625126991695
Davies-Bouldin Index 0.7091682963197171
```

Figure 97: Hierarchical Example Output

Likewise, the agglomerative clustering instance with three desired clusters is created. It is then fitted onto the "rfm_normalized" DataFrame which performs the hierarchical clustering process and assigns cluster labels to each data point. The linkage represents different approaches to defining the distance between clusters. The silhouette score, Calinski-Harabasz index, and Davies-Bouldin index for each linkage method are calculated to assess the quality of the clustering obtained.

6.4.2.3 K-Means Clustering

```
# K-means Clustering
kmeans = KMeans(n_clusters=3)
kmeans.fit(rfm_normalized)
```

Figure 98: K-Means Clustering

```
#Evaluation Metrics
silhouette_kmeans = silhouette_score(rfm_normalized, kmeans.labels_)
calinski_kmeans = calinski_harabasz_score(rfm_normalized,
kmeans.labels_)
davies_bouldin_kmeans = davies_bouldin_score(rfm_normalized,
kmeans.labels_)
```

Figure 99: K-Means Evaluation

```
K-means Clustering:
Silhouette Score: 0.5138357760452381
Calinski-Harabasz Index: 216.68529204405453
Davies-Bouldin Index 0.7052952736313335
```

Figure 100: K-Means Example Output

An instance of the K-Means algorithm with three desired clusters is created, which is then fitted to the "rfm_normalized" which performs the K-Means clustering process and assigns cluster labels to each data point. The silhouette score, Calinski-Harabasz index, and Davies-Bouldin index are calculated to assess the quality of the clustering obtained.

6.4.3 Linking Customers to Clusters

Following the identification of K-Means as the most effective algorithm for customer segmentation (Chapter 7), the next objective is to associate each customer with their corresponding cluster.

```
# Linking Customer ID to cluster value
rfm_normalized.loc[:,'CUSTOMER_ID'] = rfm['CUSTOMER_ID']
rfm_normalized['cluster'] = kmeans.labels_
data = rfm_normalized[['CUSTOMER_ID', 'cluster']]
data = pd.merge(data, result_df, on='CUSTOMER_ID', how='inner')
```

Figure 101: Linking customers to their respective clusters

	CUSTOMER_ID	cluster	FUMAKILLA VAPE MOSQU	MALTA GUINNESS CAN 3	\
0	C10404	1	0	0	
1	C11471187	0	0	0	
2	C11486781	0	0	0	
3	C115634	1	0	0	
4	C11844454	0	0	0	
..	...	...	...	...	
188	C967800	1	0	0	
189	C968130	2	0	0	
190	C982061	1	0	0	
191	C982288	1	0	0	
192	C988090	2	0	0	

Figure 102: Data DataFrame

The "CUSTOMER_ID" column from the "rfm" DataFrame is assigned to the new column in the "rfm_normalized" DataFrame as are cluster values for each customer. A new DataFrame, "data", is created that consists of only the "CUSTOMER_ID" and "cluster". This DataFrame is then merged with the "result_df" DataFrame which contains the "CUSTOMER_ID" and "PRODUCT" columns.

```
customer_id = "C11486781"
matching_rows = data[data["CUSTOMER_ID"] == customer_id]
```

Figure 103: Target user selection

```
if not matching_rows.empty:
    # Selecting customers from the same cluster as the target user
    cluster_value = matching_rows["cluster"].values[0]
    rows_with_same_cluster = data[data["cluster"] == cluster_value]
    result_df = pd.DataFrame(rows_with_same_cluster)

    # Drop the "cluster" column
    result_df = result_df.drop("cluster", axis=1)

else:
    print(f"No rows found with customer ID: {customer_id}")
```

Figure 104: Customer selection and cluster column removal

After selecting the target user, "C18547765", we retrieve the rows for the target user. The "if not matching_row.empty" is just a precautionary step to check if the matching rows were found. Then, the DataFrame "result_df" is updated, containing all the rows that match our target user's cluster value. The cluster column is then removed.

6.5 Item-based data pre-processing and preparation

For Item-Based CF, a target product is chosen. Only transactions that involve the specified target product will be used to make recommendations. The logic is to make predictions and recommendations based on items that are bought with the target product. Regarding the web application, the designated target item will be the item that captures the customer's attention, prompting them to delve deeper and inspect it further.

6.5.1 Target Product Selection

```
# Name of the column you want to filter on
column_name = 'APOLLO CHICKEN 85G'

# Remove leading and trailing whitespaces from the column names
result_df.columns = result_df.columns.str.strip()

# Retrieve rows where the specified column has a value of 1
result_df = result_df[result_df[column_name] == 1]
```

Figure 105: Selecting the target product

The target product is assigned to the 'column_name' variable, and the rows of the 'result_df' DataFrame are filtered based on the condition that the target product column is 1. This means that only the rows containing the target product will be used to make predictions.

	CUSTOMER_ID	FUMAKILLA VAPE MOSQU	MALTA GUINNESS CAN 3	\
2	C11471187	0	0	
8	C12593957	0	0	
11	C13086215	0	0	
35	C17515238	0	0	
41	C18111945	0	0	
44	C18463858	0	0	
45	C18475504	0	0	
50	C18649657	0	0	
51	C18740053	0	0	
54	C19030007	0	0	
55	C190944	0	0	
64	C236403	0	0	
70	C238832	0	0	
89	C363144	0	0	
98	C406069	0	0	
103	C45658744	0	0	
106	C487057	0	0	
109	C495625	0	0	
131	C68236	0	0	
135	C689238	0	0	
151	C798747	0	0	
155	C823578	0	0	

Figure 106: Customer transactions that contain the target product

6.6 Advanced Collaborative Filtering

Both User-Based and Item-Based approaches use the following principles to make predictions and recommendations:

6.6.1 Non-Negative Matrix Factorization Collaborative Filtering

```
# Create user-item matrix
ratings_matrix = result_df.set_index('CUSTOMER_ID').iloc[:,
1:].fillna(0)
```

Figure 107: User-item matrix creation

The user-item interaction matrix is created, where the rows represent customers and the columns represent the products. The missing values are filled with zeros.

```
# Extract the row corresponding to the customer_id
customer_ratings = ratings_matrix.loc[[customer_id]]

# Drop the row corresponding to the customer_id from ratings_matrix to
create train_data
train_data = ratings_matrix.drop(index=customer_id)

# Perform the 70:30 train-test split on the remaining data (train_data)
train_data, test_data = train_test_split(train_data, test_size=0.3,
random_state=42)

# Concatenate the customer_ratings with test_data
test_data = pd.concat([test_data, customer_ratings])
```

Figure 108: Train-Test Split for NMF

The corresponding row for the target customer is removed from the interaction matrix and kept in "customer_ratings". The data is then split into training and testing sets, with 70% of the data used for training and 30% for testing. To make predictions for the target customer, "customer_ratings" is added to the testing set.

```
# Define the NMF algorithm and train the model
nmf_model = NMF(n_components=25, init='nndsvdar', random_state=25,
solver='mu')
nmf_model.fit(train_data)
```

Figure 109: NMF Training

The NMF model is defined with hyperparameters such as the number of components, initialization method, and solver for further optimization. The model is then trained on the training set.

```
# Get predicted ratings for the test data
predicted_ratings = np.dot(nmf_model.transform(test_data),
nmf_model.components_)
```

Figure 110: Test set prediction for NMF

The testing set is transformed to obtain the user's latent space factors. A dot product is performed on the testing set's user latent factors and the training set's item latent factors. The predicted ratings are essentially generated by combining the testing set's user behaviour with the items latent factors learned from training.

```
# Flatten the predicted and actual rating matrices
predicted_ratings_flat = predicted_ratings.flatten()
actual_ratings_flat = test_data.values.flatten()
```

Figure 111: Flattening predicted and actual rating matrices to a 1-D array NMF

The predicted ratings and actual ratings matrices are flattened into a 1-dimensional array for easier comparison and evaluation.

```
# Calculate RMSE and MAE
rmse = np.sqrt(mean_squared_error(actual_ratings_flat,
predicted_ratings_flat))
mae = mean_absolute_error(actual_ratings_flat, predicted_ratings_flat)
```

Figure 112: Calculate RMSE and MAE for NMF

The RMSE and MAE are calculated by comparing the actual ratings with the predicted ratings.

```
# Calculate TP, TN, FP, FN
threshold = 0.15
predicted_positives = predicted_ratings_flat >= threshold
actual_positives = actual_ratings_flat > 0

tp = np.sum(predicted_positives & actual_positives)
tn = np.sum((~predicted_positives) & (~actual_positives))
fp = np.sum(predicted_positives & (~actual_positives))
fn = np.sum((~predicted_positives) & actual_positives)
```

Figure 113: Calculate TP, TN, FP, and FN for NMF

The TP, TN, FP, and FN are calculated based on a threshold value of 0.15. The threshold value is used to convert the predicted matrix into a binary matrix.

```
# Initialize a new recommendation matrix with zeros
recommendation_matrix = np.zeros_like(predicted_ratings)

# Compare the predicted_ratings matrix with the actual test_data matrix
for i in range(predicted_ratings.shape[0]):
    for j in range(predicted_ratings.shape[1]):
        # Check if the predicted value is greater than zero and the
actual value is zero
        if predicted_ratings[i, j] > 0 and test_data.iloc[i, j] == 0:
            # Update the recommendation_matrix with the predicted value
            recommendation_matrix[i, j] = predicted_ratings[i, j]
```

Figure 114: NMF recommendation matrix

To provide recommendations, a recommendation matrix is created following the same shape as the predicted ratings matrix. When comparing the recommendation matrix with the predicted ratings matrix, if the predicted ratings are greater than zero, the predicted ratings are inserted into the recommendation matrix. Otherwise, the value remains zero. This process is used to recommend items to customers that they have not purchased.

```
# Number of recommendations
k = 5

# Get recommended items with predicted ratings for each user
for user_id in test_data.index:
    # Check if user_id exists in test_data
    if user_id in test_data.index:
        # Get the index position of the user_id
        user_index = test_data.index.get_loc(user_id)

        # Get the predicted ratings for the user
        user_ratings = recommendation_matrix[user_index]

        # Get the indices of the top k items
        top_items_indices = np.argsort(-user_ratings)[:k]

        # Get the actual product names corresponding to the top item
indices
        recommended_items = ratings_matrix.columns[top_items_indices]

        # Get the predicted ratings for the top recommended items
        predicted_ratings_top = user_ratings[top_items_indices]
```

Figure 115: NMF recommendation for each customer

```
        print(f"Recommended items with predicted ratings for user
{user_id} :")
        for item, rating in zip(recommended_items.tolist(),
predicted_ratings_top.tolist()):
            print(f"Item: {item}, Predicted Rating: {rating:.2f}")
        print()
    else:
        print("User", user_id, "does not exist in the test data.")
```

Figure 116: NMF printing recommendation for each customer

For recommendation method 1, the code retrieves the index position for each user and gets the predicted ratings for that user from the "predicted_ratings" array. The top k items are identified, and the corresponding product names are retrieved.

```
Recommended items with predicted ratings for user C12935659 :
Item: APOLLO CURRY 85G , Predicted Rating: 0.13
Item: KRAFT 250G CHEDDAR , Predicted Rating: 0.12
Item: JUTE SHOPPING BAG 45 , Predicted Rating: 0.09
Item: SAC BIO CAISSE , Predicted Rating: 0.09
Item: WS WHOLE PEEL.TOMATO , Predicted Rating: 0.09

Recommended items with predicted ratings for user C18321842 :
Item: WILLARDS 30G CHEESE , Predicted Rating: 0.18
Item: SAC BIO CAISSE , Predicted Rating: 0.16
Item: APOLLO CURRY 85G , Predicted Rating: 0.15
Item: APOLLO CHICKEN 85G , Predicted Rating: 0.14
Item: PAIN CHOCO X5PC , Predicted Rating: 0.13

Recommended items with predicted ratings for user C13960407 :
Item: KRAFT 250G CHEDDAR , Predicted Rating: 0.31
Item: BROWN TOAST BREAD-LA , Predicted Rating: 0.17
Item: PEPSI TWIST 500ML , Predicted Rating: 0.17
Item: WILLARDS 30G CHUTNEY , Predicted Rating: 0.16
Item: CURLIES 40G-CHICKEN , Predicted Rating: 0.15

Recommended items with predicted ratings for user C11486781 :
Item: APOLLO CURRY 85G , Predicted Rating: 0.31
Item: WS WHOLE PEEL.TOMATO , Predicted Rating: 0.28
Item: WS TUNA FLAKES IN OI , Predicted Rating: 0.25
Item: TWISTIES 20G CHEESE , Predicted Rating: 0.23
Item: BROWN TOAST BREAD-LA , Predicted Rating: 0.22
```

Figure 117: Recommendation for each customer's output

```python
# Find the indices of the k columns with the highest maximum values
top_k_indices = np.argsort(-np.max(recommendation_matrix, axis=0))[:k]

# Get the corresponding column names and their respective ratings
top_k_items = ratings_matrix.columns[top_k_indices]
top_k_ratings = np.max(recommendation_matrix, axis=0)[top_k_indices]

print("Top", k, "items with highest predicted ratings:")

# Print the top k items with their ratings
for item, rating in zip(top_k_items, top_k_ratings):
    print("-", item, "with rating:", rating)
```

Figure 118: NMF's highest-rated item recommendation

For recommendation method 2, the 'np.argsort()' and 'np.max()' functions identify the indices of the k items with the highest maximum predicted ratings across all users. Once the column names corresponding to these indices are obtained, the top k items can be retrieved.

```
Top 5 items with highest predicted ratings:
- KRAFT 250G CHEDDAR  with rating: 0.4104108545437398
- APOLLO CURRY 85G  with rating: 0.3130957848767512
- BAGUETTE PARISIENNE  with rating: 0.290082566064483
- WS WHOLE PEEL.TOMATO  with rating: 0.27953116782967075
- BROWN TOAST BREAD-LA  with rating: 0.27627462765169736
```

Figure 119: NMF's highest-rated item output

6.6.2 Singular Value Decomposition Collaborative Filtering

```
melted_df = result_df.melt(id_vars=['CUSTOMER_ID'], var_name='Product',
value_name='Rating')
melted_df = melted_df.rename(columns={'CUSTOMER_ID': 'customer_id',
'Product': 'product_id'})
```

Figure 120: SVD conversion to long format

The 'melt()' function transforms the 'result_df' DataFrame from a wide format to a long format, and the columns are renamed.

```
# Extract the row corresponding to the customer_id
customer_ratings = melted_df.loc[melted_df['customer_id'] ==
customer_id]

# Drop the rows corresponding to the customer_id from melted_df to
create trainset
trainset = melted_df.drop(melted_df[melted_df['customer_id'] ==
customer_id].index)

# Convert melted_df to Surprise Dataset
reader = Reader(rating_scale=(0, 1))
data = Dataset.load_from_df(trainset, reader)

# Split data into train and test sets
trainset, testset = train_test_split(data, test_size=0.3,
random_state=42)

# Add customer_ratings to the testset
testset_with_customer = testset + [(customer_id, row['product_id'],
row['Rating']) for _, row in customer_ratings.iterrows()]

# Train the model
model = SVDpp(n_factors=100, n_epochs=10, lr_all=0.01, reg_all=0.1,
random_state=25, verbose=True)
model.fit(trainset)
```

Figure 121: SVD Train-Test Split and Training

Similarly to NMF, the target customer's row is removed and stored temporarily in another variable. The data is then split into 70% training and 30% testing. The target customer's information is then added to the testing to make the predictions. The SVD++ model is defined with hyperparameters such as the number of factors, the number of iterations, the learning rate, and the regularisation term for all parameters. Finally, the model is trained on the training set.

```
# Predict ratings for test set
predictions = model.test(testset)

# Compute RMSE and MAE
rmse = accuracy.rmse(predictions)
mae = accuracy.mae(predictions)
```

Figure 122: SVD Test set predictions and RMSE and MAE calculations

To generate the predictions for the testing set, the testing set's user-latent factors are multiplied by the item-latent factors from the training set. The RMSE and MAE are then calculated.

```
# Set the threshold
threshold = 0.15

# Convert predictions to flat arrays
predicted_ratings_flat = np.array([pred.est for pred in predictions])
actual_ratings_flat = np.array([pred.r_ui for pred in predictions])


# Calculate TP, TN, FP, FN
predicted_positives = predicted_ratings_flat >= threshold
actual_positives = actual_ratings_flat > 0

tp = np.sum(predicted_positives & actual_positives)
tn = np.sum((~predicted_positives) & (~actual_positives))
fp = np.sum(predicted_positives & (~actual_positives))
fn = np.sum((~predicted_positives) & actual_positives)
```

Figure 123: Calculate TP, TN, FP, and FN for SVD

By extracting the estimated ratings 'pred.est' and actual ratings 'pred.r_ui' from each prediction, the respective resulting list of ratings is converted into an array for both predicted ratings and actual ratings. Using the threshold value, the predicted ratings are converted to 1 and 0s, and TP, TN, FP, and FN are calculated.

```
# Convert predictions to a DataFrame for easy indexing
predictions_df = pd.DataFrame(predictions, columns=['customer_id',
'product_id', 'actual_rating', 'predicted_rating', 'details'])
predictions_df = predictions_df.set_index(['customer_id',
'product_id'])

# Create a mapping between user_id and integer index
user_id_to_index = {user_id: index for index, user_id in
enumerate(melted_df['customer_id'].unique())}

# Create a mapping between product_id and integer index
product_id_to_index = {product_id: index for index, product_id in
enumerate(melted_df['product_id'].unique())}

# Initialize a new recommendation matrix with zeros
num_users = len(user_id_to_index)
num_products = len(product_id_to_index)
recommendation_matrix = np.zeros((num_users, num_products))

# Fill the recommendation matrix with predicted ratings for appropriate
user-item pairs
for user_id in melted_df['customer_id'].unique():
    user_index = user_id_to_index[user_id]
    for product_id in melted_df['product_id'].unique():
        product_index = product_id_to_index[product_id]
        if (user_id, product_id) in predictions_df.index:
            predicted_rating = predictions_df.loc[(user_id,
product_id), 'predicted_rating']
            actual_rating = predictions_df.loc[(user_id, product_id),
'actual_rating']
            if predicted_rating > 0 and actual_rating == 0:
                recommendation_matrix[user_index, product_index] =
predicted_rating
```

Figure 124: SVD recommendation matrix

The predictions generated are converted into a format for proper indexing. After mapping "CUSTOMER_ID" and "PRODUCT" to unique values, the recommendation matrix is created and filled with zeros. During the comparison between the predicted and actual ratings, if the predicted ratings are greater than zero and the actual is zero, the predicted ratings are then inserted into the recommendation matrix.

```
# Make recommendations for a given user
def get top n recommendations(recommendation matrix, user id, n):
    # Get all unique products
    products = melted_df['product_id'].unique()

    # Get the integer index corresponding to the given user ID
    user_index = user_id_to_index[user_id]

    # Get the predicted ratings for the user from the
recommendation matrix
    user ratings = recommendation matrix[user index]

    # Sort the user_ratings in descending order and get the indices of
the top N items
    top_n_indices = np.argsort(-user_ratings)[:n]

    # Get the product IDs corresponding to the top N items
    top_n_items = products[top_n_indices]

    # Get the predicted ratings for the top N items
    top_n_ratings = user_ratings[top_n_indices]

    # Return the list of top N recommendations as tuples of (product
ID, predicted rating)
    return list(zip(top_n_items, top_n_ratings))
```

Figure 125: SVD recommendation for a given customer

The function takes the recommendation matrix, the target user ID and the number of desired recommendations and outputs the items for recommendations.

```
# Number of recommendations
k = 5

recommendations = {}
for user id in melted df['customer id'].unique():
    user_recommendations =
get_top_n_recommendations(recommendation_matrix, user_id, k)
    recommendations[user id] = user recommendations

for user_id, user_recommendations in recommendations.items():
    print("Recommended items for user", user_id, ":")
    for item id, rating in user recommendations:
        print("-", item_id, "(predicted rating:", rating, ")")
    print()
```

Figure 126: SVD printing recommendations for each customer

This first method makes recommendations for each user. It retrieves all products, calculates the predicted ratings for each user, and sorts them to select the top k recommendations.

```
Recommended items for user C11471187 :
- BELINDA WH PEELED TO  (predicted rating: 0.24025938175298028 )
- LEADER SUNFL. OIL 1L  (predicted rating: 0.18597408670733712 )
- SUPPLEMENT GLACE  (predicted rating: 0.17954165610561154 )
- TR TUNA FL OIL 170G  (predicted rating: 0.16627799120198322 )
- OREO VANILLA CR 176G  (predicted rating: 0.1654614639663213 )

Recommended items for user C11486781 :
- APOLLO CURRY 85G  (predicted rating: 0.40007390510980234 )
- FARMLAND FCMP 1KG  (predicted rating: 0.23837520494281697 )
- JUTE SHOPPING BAG 45  (predicted rating: 0.23629526241557122 )
- APOLLO CHICKEN 85G  (predicted rating: 0.21006272716902688 )
- BF OIGNON ROUGE FILL  (predicted rating: 0.16887384441080877 )

Recommended items for user C11844454 :
- APOLLO CURRY 85G  (predicted rating: 0.36855790359456897 )
- WS TUNA FLAKES IN OI  (predicted rating: 0.1804277177721073 )
- TR TUNA FL OIL 170G  (predicted rating: 0.15896006009289904 )
- ORIENT GROS POIS 500  (predicted rating: 0.15839636295740359 )
- BF OIGNON ROUGE FILL  (predicted rating: 0.13579217759311987 )

Recommended items for user C12126090 :
- WS WHOLE PEEL.TOMATO  (predicted rating: 0.2654454748625784 )
- FARMLAND FCMP 1KG  (predicted rating: 0.21938381652560382 )
- JUTE SHOPPING BAG 45  (predicted rating: 0.2160477882815688 )
- APOLLO CHICKEN 85G  (predicted rating: 0.19028181442717282 )
- LEADER SUNFL. OIL 1L  (predicted rating: 0.1807666444548301 )
```

Figure 127: SVD recommendation for each customer output

```python
# Get the top k highest-rated items
top_k_items = []
top_k_ratings = []

def add_item(item_id, rating):
    if item_id not in top_k_items:
        if len(top_k_items) < k:
            top_k_items.append(item_id)
            top_k_ratings.append(rating)
        else:
            min_rating_index = top_k_ratings.index(min(top_k_ratings))
            if rating > top_k_ratings[min_rating_index]:
                top_k_items[min_rating_index] = item_id
                top_k_ratings[min_rating_index] = rating
```

Figure 128: SVD's highest-rated item recommendation

```
# Loop through the recommendations and add items to the top_k lists
for user_recommendations in recommendations.values():
    for item_id, rating in user_recommendations:
        add_item(item_id, rating)

# Print the top k highest-rated items
print("Top", k, "highest-rated items:")
for item, rating in zip(top_k_items, top_k_ratings):
    print(item, "(predicted rating:", rating, ")")
```

Figure 129: SVD printing's highest-rated item recommendation

The second recommendation method identifies the top k highest-rated items regardless of the user. These highest-rated items are then given as recommendations.

```
Top 5 highest-rated items:
BAGUETTE PARISIENNE  (predicted rating: 0.31740854721451967 )
WS WHOLE PEEL.TOMATO  (predicted rating: 0.2654454748625784 )
KRAFT 250G CHEDDAR  (predicted rating: 0.4383641863787792 )
SAC BIO CAISSE  (predicted rating: 0.37533522822488347 )
APOLLO CURRY 85G  (predicted rating: 0.40007390510980234 )
```

Figure 130: SVD's highest-rated item recommendation output

6.6.3 Neural Collaborative Filtering

```
# Define the NCF model
class NCF(tf.keras.Model):
    def __init__(self, num_users, num_items, mf_dim=8, mlp_dim=[32, 16,
8], dropout=0.2):
        super(NCF, self).__init__()

        # User embedding
        self.user_embedding = tf.keras.layers.Embedding(num_users,
mf_dim)

        # Item embedding
        self.item_embedding = tf.keras.layers.Embedding(num_items,
mf_dim)

        # Generalized Matrix Factorization (GMF)
        self.gmf_user_embedding = tf.keras.layers.Embedding(num_users,
mf_dim)
        self.gmf_item_embedding = tf.keras.layers.Embedding(num_items,
mf_dim)
        self.gmf_output = tf.keras.layers.Dense(1,
activation='sigmoid')

        # Multi-Layer Perceptron (MLP)
        mlp_layers = []
        for units in mlp_dim:
            mlp_layers.append(tf.keras.layers.Dense(units,
activation='relu'))
            mlp_layers.append(tf.keras.layers.Dropout(dropout))
        self.mlp_layers = mlp_layers
        self.mlp_output = tf.keras.layers.Dense(1,
activation='sigmoid')

        # Fusion network
        self.fusion_output = tf.keras.layers.Dense(1,
activation='sigmoid')
```

Figure 131: NCF class constructor

```
    def call(self, inputs):
        user_id, item_id = inputs

        # User embedding
        user_embedding = self.user_embedding(user_id)
        item_embedding = self.item_embedding(item_id)

        # GMF
        gmf_user_embedding = self.gmf_user_embedding(user_id)
        gmf_item_embedding = self.gmf_item_embedding(item_id)
        gmf_output = self.gmf_output(gmf_user_embedding *
gmf_item_embedding)

        # MLP
        mlp_user_embedding = tf.keras.layers.Flatten()(user_embedding)
        mlp_item_embedding = tf.keras.layers.Flatten()(item_embedding)
        mlp_input = tf.concat([mlp_user_embedding, mlp_item_embedding],
axis=-1)
        for layer in self.mlp_layers:
            mlp_input = layer(mlp_input)
        mlp_output = self.mlp_output(mlp_input)

        # Fusion network
        reshaped_gmf_output = tf.keras.layers.Reshape((-
1,))(gmf_output)
        fusion_input =
tf.keras.layers.Concatenate()([reshaped_gmf_output, mlp_output])
        fusion_output = self.fusion_output(fusion_input)

        return fusion_output
```

Figure 132: NCF forward pass method

The 'NCF' class inherits from 'tf.keras.model' and defines the structure of the NCF model. The constructor method '__init__' initialises the model's parameters and layers and takes as inputs: the number of users, the number of items, the dimension of the matrix factorization, the dimensions of the MLP layers, and the dropout rate. The model consists of several layers for embeddings: GMF, MLP, and the fusion network. The 'user_embeddings' and 'item_embeddings' create embeddings for users and items, respectively. The GMF component calculates the interaction between user and item embeddings using element-wise multiplication, while the MLP component applies a series of dense layers with dropout to capture the complex patterns in the user-item interactions. The fusion network combines the outputs of the GMF and MLP components, and the model outputs the final prediction using a sigmoid activation.

```
# Map CUSTOMER_ID and PRODUCT to unique integers
user_mapping = {user_id: i for i, user_id in
enumerate(df['CUSTOMER_ID'].unique())}
item_mapping = {item_id: i for i, item_id in
enumerate(df['PRODUCT'].unique())}

df['user_id'] = df['CUSTOMER_ID'].map(user_mapping)
df['item_id'] = df['PRODUCT'].map(item_mapping)

num_users = len(user_mapping)
num_items = len(item_mapping)
```

Figure 133: Mapping Customer ID and Product to Unique Integers

The code maps unique 'CUSTOMER_ID' and 'PRODUCT' to unique integers. The 'user_mapping' dictionary maps the unique 'CUSTOMER_ID', while the 'item_mapping' dictionary maps the unique 'PRODUCT'. The 'map' method replaces the 'CUSTOMER_ID' and 'PRODUCT' columns with their unique integer values, resulting in new 'user_id' and 'item_id' columns. The 'num_users' and 'num_items' represent the total count of unique users and items, respectively.

```
# Extract the row corresponding to the customer_id
customer_ratings = df[df['CUSTOMER_ID'] == customer_id]

# Remove the rows corresponding to the customer_id from df to create
train_df
train_df = df[df['CUSTOMER_ID'] != customer_id]

# Split the data into training and testing sets
train_df, test_df = train_test_split(df, test_size=0.3,
random_state=25)

# Now, we concatenate customer_ratings to test_df
test_df = pd.concat([test_df, customer_ratings])
```

Figure 134: NCF Train-Test Split

Similar to NMF and SVD, the target customer's information is removed and saved in another variable. The data is then split into 70% training and 30% testing. Once the splitting is complete, the saved target customer's data is added to the testing data.

```
# Define the embedding size
embedding_size = 16

# Define the user input
user_input = Input(shape=(1,))
user_embedding = Embedding(num_users, embedding_size)(user_input)
user_flatten = Flatten()(user_embedding)

# Define the item input
item_input = Input(shape=(1,))
item_embedding = Embedding(num_items, embedding_size)(item_input)
item_flatten = Flatten()(item_embedding)
```

Figure 135: NCF Embeddings

The 'embedding_size' specifies the dimensionality of the user and item embeddings. The user and item input layers are defined by using the 'Input' function from TensorFlow, and the input shape is set to '(1,)', indicating that each input is a single integer representing a user or an item. The number of users or items is used as the input dimension for creating the user embedding layer. By using the 'Flatten' layer, the multi-dimensional user or item embeddings are converted into a one-dimensional vector.

```
# Define the NCF model
NCF_input = [user_input, item_input]
NCF_output = NCF(num_users, num_items)(NCF_input)

# Compile the NCF model
NCF_model = Model(inputs=[user_input, item_input], outputs=NCF_output)
NCF_model.compile(optimizer=Adam(learning_rate=0.001),
loss='binary_crossentropy', metrics=['accuracy'])

# Train the NCF model
NCF_model.fit(
    [train_df['user_id'], train_df['item_id']],
    train_df['RATING'],
    batch_size=64,
    epochs=10,
    validation_data=([test_df['user_id'], test_df['item_id']],
test_df['RATING'])
)
```

Figure 136: NCF Definition, Compilation, and Training

The NCF is defined, compiled, and trained. The user and item inputs are combined, and the NCF model's output is obtained. The model is compiled using the Adam optimizer and the binary cross-

entropy loss function. The model learns the embeddings for both users and items based on the training set.

```
# Generate predictions for all user-item pairs
all_predictions = NCF_model.predict([np.array(df['user_id']).reshape(-
1, 1), np.array(df['item_id']).reshape(-1, 1)])


# Create a DataFrame to store the predictions
predictions_df = pd.DataFrame({
    'CUSTOMER ID': df['CUSTOMER ID'],
    'PRODUCT': df['PRODUCT'],
    'PREDICTED_RATING': all_predictions.flatten()
})

# Group the predictions by CUSTOMER_ID
grouped_predictions = predictions_df.groupby('CUSTOMER_ID')
```

Figure 137: Generate NCF predictions

Using the 'NCF_model.predict' method, the predictions for all user-item pairs are generated. For each user or item in the testing set, their corresponding embedding is extracted. The dot product of the user and item embeddings is calculated to predict the user-item matrix.

```
sorted_predictions = predictions_df.sort_values(by='PREDICTED_RATING',
ascending=False)

# Assuming threshold of 0.5
threshold = 0.15

# Classify predicted ratings as positive (1) or negative (0)
predictions_df['PREDICTED_CLASS'] =
predictions_df['PREDICTED_RATING'].apply(lambda x: 1 if x >= threshold
else 0)

# Merge actual ratings and predicted classes
merged_df = pd.merge(test_df, predictions_df, on=['CUSTOMER_ID',
'PRODUCT'], how='inner')

# Calculate TP, TN, FP, FN
TP = merged_df[(merged_df['RATING'] == 1) &
(merged_df['PREDICTED_CLASS'] == 1)].shape[0]
TN = merged_df[(merged_df['RATING'] == 0) &
(merged_df['PREDICTED_CLASS'] == 0)].shape[0]
FP = merged_df[(merged_df['RATING'] == 0) &
(merged_df['PREDICTED_CLASS'] == 1)].shape[0]
FN = merged_df[(merged_df['RATING'] == 1) &
(merged_df['PREDICTED_CLASS'] == 0)].shape[0]
```

Figure 138: Calculate TP, TN, FP, and FN for NCF

The threshold value converts the predicted ratings into 1s and 0s. The TP, TN, FP, and FN are then calculated.

```
# Initialize a new recommendation matrix with zeros
recommendation_matrix = np.zeros((num_users, num_items))

# Iterate through the predictions and fill the recommendation matrix
for _, row in predictions_df.iterrows():
    user_id = row['CUSTOMER_ID']
    item_id = row['PRODUCT']
    predicted_rating = row['PREDICTED_RATING']

    # Check if the predicted rating is greater than zero and the actual
rating is zero
    if predicted_rating > 0 and
recommendation_matrix[user_mapping[user_id], item_mapping[item_id]] ==
0:
        recommendation_matrix[user_mapping[user_id],
item_mapping[item_id]] = predicted_rating
```

Figure 139: NCF recommendation matrix

The recommendation matrix is created using the number of unique users and items, and it is filled with zeros. During the iteration, if the predicted rating is greater than zero and the actual rating is zero, the predicted rating is inserted into the recommendation matrix.

```
# Number of recommendations
k = 5

# Function to get the top-K recommendations for each user
def get_top_k_recommendations(group, k):
    # Sort the predictions in descending order
    sorted_predictions = group.sort_values('PREDICTED_RATING',
ascending=False)
    # Get the top-K recommendations
    top_k_recommendations = sorted_predictions.head(k)
    # Return the recommended items
    return top_k_recommendations['PRODUCT'].tolist()

# Get the top-K recommendations for each user
recommended_items = grouped_predictions.apply(lambda x:
get_top_k_recommendations(x, k)).reset_index()

# Print the recommended items
print(recommended_items)
```

Figure 140: Printing recommendations for all customers

	CUSTOMER_ID	0
0	C11471187	[KRAFT 250G CHEDDAR , JUTE SHOPPING BAG 45 , W...
1	C11486781	[SAC BIO CAISSE , KRAFT 250G CHEDDAR , BAGUETT...
2	C11844454	[KRAFT 250G CHEDDAR , SAC BIO CAISSE , WS VEGE...
3	C12126090	[JUTE SHOPPING BAG 45 , COCA COLA 1.5L , APOLL...
4	C12197918	[KRAFT 250G CHEDDAR , BAGUETTE PARISIENNE , WS...
5	C12208033	[KRAFT 250G CHEDDAR , WS VEGETABLE OIL BTL , S...
6	C12593957	[SAC BIO CAISSE , BAGUETTE PARISIENNE , WEETAB...
7	C12935659	[BAGUETTE PARISIENNE , JUTE SHOPPING BAG 45 , ...
8	C13086215	[KRAFT 250G CHEDDAR , WATTIES SWEET CORN 4 , R...
9	C13131761	[KRAFT 250G CHEDDAR , WS VEGETABLE OIL BTL , B...
10	C13960407	[KRAFT 250G CHEDDAR , LEADER SUNFL. OIL 1L , F...
11	C14447774	[KRAFT 250G CHEDDAR , LKS WHITE FLOUR 2KG , PH...
12	C14739580	[KRAFT 250G CHEDDAR , SAC BIO CAISSE , FARMLAN...
13	C14801419	[KRAFT 250G CHEDDAR , LEADER SUNFL. OIL 1L , S...
14	C15320509	[BAGUETTE PARISIENNE , LKS WHITE FLOUR 2KG , K...
15	C15578186	[KRAFT 250G CHEDDAR , BELINDA WH PEELED TO , L...
16	C15823487	[JUTE SHOPPING BAG 45 , WS TUNA FLAKES IN OI ,...

Figure 141: Recommendations for all customer output

```
# Filter the recommended items for the specific customer ID
customer_recommendations =
recommended items[recommended items['CUSTOMER ID'] == customer id]

# Get the recommended items for the customer
recommended_items_list = customer_recommendations.iloc[0, 1]

# Get the corresponding predicted ratings for the recommended items
predicted_ratings_list = []
for item in recommended_items_list:
    predicted rating = predictions df[(predictions df['CUSTOMER ID'] ==
customer_id) & (predictions_df['PRODUCT'] ==
item)]['PREDICTED_RATING'].iloc[0]
    predicted_ratings_list.append(predicted_rating)

# Print the recommended items and their corresponding predicted ratings
print("Recommendations for Customer", customer_id, ":")
for item, rating in zip(recommended_items_list,
predicted ratings list):
    print(item, "(predicted rating:", rating, ")")
```

Figure 142: Printing recommendations for a given customer

```
Recommendations for Customer C13131761 :
KRAFT 250G CHEDDAR  (predicted rating: 0.69048786 )
WS VEGETABLE OIL BTL  (predicted rating: 0.6872397 )
BAGUETTE PARISIENNE  (predicted rating: 0.68196607 )
LEADER SUNFL. OIL 1L  (predicted rating: 0.67599404 )
BELINDA WH PEELED TO  (predicted rating: 0.64838666 )
```

Figure 143: Recommendations for a given customer output

In the first method to get the recommendations, all the predicted items for the target user are retrieved. If the item is not purchased by the target user, the predicted item is added to a list. This list contains the recommendations for the target user.

```python
# Sort the predictions in descending order by 'PREDICTED_RATING'
sorted_predictions = predictions_df.sort_values('PREDICTED_RATING',
ascending=False)

# Get the top-K unique recommendations
top_k_recommendations =
sorted_predictions.drop_duplicates(subset='PRODUCT').head(k)

# Print the recommended items and their ratings
for index, row in top_k_recommendations.iterrows():
    print("Item:", row['PRODUCT'])
    print("Rating:", row['PREDICTED_RATING'])
    print()
```

Figure 144: NCF's highest-rated item recommendation

```
Item: KRAFT 250G CHEDDAR
Rating: 0.6926701664924622

Item: BAGUETTE PARISIENNE
Rating: 0.6926632523536682

Item: SAC BIO CAISSE
Rating: 0.6925541162490845

Item: SELVA SUCRE BLC 1KG
Rating: 0.6923627853393555

Item: BELINDA WH PEELED TO
Rating: 0.6922738552093506
```

Figure 145: NCF's highest-rated item recommendation output

In the second method of providing recommendations, the predicted ratings are sorted in descending order. Duplicates are removed so that recommendations are made only for unique items. Recommendations are made based on the top-k rated items.

Chapter 7: Results, Evaluation, and Discussion

This chapter evaluates the performance and effectiveness of the algorithms considered in building the recommendation system. Through this evaluation, we gain valuable insights into the strengths and limitations of the algorithms, enabling us to make informed decisions about optimizing the recommendation system. By analysing their performance and effectiveness, we ensure that our system meets the desired objectives. More details about the evaluation metrics used can be found in Appendix 3.

7.1. Clustering

In Chapter 6, we explored three clustering algorithms to identify the most suitable approach for our dataset in the context of the proposed recommender system. To ensure optimal performance, hyperparameter tuning was performed for the algorithms that required it. This comprehensive evaluation allows us to confidently determine the best-performing clustering algorithm.

Table 12: Clustering Evaluation

Clustering Technique	Silhouette Score	Calinski-Harabasz Index	Davies-Bouldin Index
Spectral	0.512384	214.625127	0.709168
K-Means	0.513836	216.685292	0.705295
Hierarchical (complete linkage)	0.41174	90.513865	1.183582
Hierarchical (Single linkage)	0.550538	105.117544	0.49885
Hierarchical (Average linkage)	0.550538	105.117544	0.49885
Hierarchical (Ward linkage)	0.512384	214.625127	0.709168

To determine the best-performing algorithm based on Table 12, we need to consider the following:

1. A higher silhouette score indicates better-defined and well-separated clusters.
2. A higher Calinski-Harabasz index means better clustering performance with dense and well-separated clusters.
3. A lower Davies-Bouldin index indicates better-defined clusters with minimal overlap.

We can observe that the K-Means clustering technique exhibits superior performance across these metrics compared to the other techniques.

7.2 Hyperparameter tuning

Hyperparameters are crucial for optimising machine learning models. It improves the performance metrics by achieving better generalization, ensuring model stability, and ensuring efficient resource utilization. By using a better combination of hyperparameters that align with the dataset, this leads to a more accurate and effective model.

7.2.1 Non-Negative Matrix Factorization

There are several hyperparameters in NMF that can significantly impact the performance and quality of the factorization results. Table 13 presents a summary of the various options and their corresponding impacts on the performance of the algorithm.

Table 13: NMF Hyperparameter Tuning

NMF Variations	Number of components	Initialization method	Numerical Solver
A	5	random	mu
B	10	random	mu
C	25	random	mu
D	25	nndsvd	mu
E	25	nndsvda	mu
F	25	nndsvdar	mu
G	25	random	cd

Table 14 shows the result of performing the proposed model and using NMF as the advanced collaborative filtering algorithm on identical customer IDs.

Table 14: NMF Hyperparameter Tuning Results

	NMF A	NMF B	NMF C	NMF D	NMF E	NMF F	NMF G
RMSE	0.2082	0.2075	0.2059	0.2063	0.2060	0.2061	0.2060
MAE	0.0635	0.0644	0.0644	0.0648	0.0644	0.0645	0.0645
TRUE POSITIVE	56	73	109	96	107	101	109
TRUE NEGATIVE	16656	16610	16481	16505	16487	16474	16504
FALSE POSITIVE	119	165	294	270	288	301	271
FALSE NEGATIVE	732	715	679	692	681	687	679
ACCURACY	0.9515	0.9499	0.9446	0.9452	0.9448	0.9437	0.9459
PRECISION	0.3200	0.3067	0.2705	0.2623	0.2709	0.2512	0.2868
SPECIFICITY	0.9929	0.9902	0.9825	0.9839	0.9828	0.9821	0.9838
SENSITIVITY	0.0711	0.0926	0.1383	0.1218	0.1358	0.1282	0.1383
F1 SCORE	0.1163	0.1423	0.1830	0.1664	0.1809	0.1698	0.1866

Table 15: NMF Hyperparameter Tuning Observations

Observations	
RMSE and MAE	The RMSE values range from 0.2059 to 0.2082 and the MAE values range from 0.0635 to 0.0648. NMF C has the lowest RMSE value of 0.2059 which shows it has the smallest average prediction error while NMF A has the lowest MAE value of 0.0635 which indicates it has the smallest average absolute prediction error.
True Positive, True Negative, False Positive, and False Negative	True Positive, representing the number of positive samples correctly classified, ranges from 56 to 109, while True Negative, which represents the number of negative samples correctly classified, ranges from 16474 to 16656. NMF G indicates to have to most positive classifications of 109 and NMF A shows to have the most negative classifications at 16656. False Positive representing the negative samples incorrectly classified as negative ranges from 119 to 301. False Negative, which is the number of positive samples incorrectly classified as negative ranges from 679 to 732. NMF A has the lowest False Positive value of 119 while NMF C and NMF G has the lowest False Negative value of 679.
Accuracy, Precision, Specificity, Sensitivity, and F1 Score	The model with the highest accuracy of 0.9515 is NMF A (95.15%). NMF A also shows to have the highest precision of 0.3200 (32.00%) and the highest specificity value at 0.9929 (99.29%). The model with the highest sensitivity is NMF C and NMF G at 0.1383 (13.83%). Lastly, NMF G shows to have the highest F1 Score, indicating the best balance between precision and sensitivity at 0.1866.
Best Model	As per our observations, it seems that NMF A is the best-performing model with number of components, 5, initialisation method, random, and numerical solver, mu.

Conclusion: NMF A is chosen as the best-performing model for further analysis.

7.2.2 Singular Value Decomposition

Table 16 provides a summary for SVD on various of its hyperparameter options and their corresponding impacts on the algorithm's performance.

Table 16: SVD Hyperparameter Tuning

SVD Variations	Number of factors	Number of iterations	Learning rate	Regularization term
SVD A	100	10	0.001	0.01
SVD B	10	10	0.001	0.01
SVD C	50	10	0.001	0.01
SVD D	100	5	0.001	0.01
SVD E	100	20	0.001	0.01
SVD F	100	10	0.01	0.01
SVD G	100	10	0.1	0.01
SVD H	100	10	0.01	0.1
SVD I	100	10	0.01	0.5
SVD J	100	10	0.01	0.001

Table 17 shows the result of performing the proposed model and using SVD as the advanced collaborative filtering algorithm on identical customer IDs.

Table 17: SVD Hyperparameter Tuning Results

	SVD A	SVD B	SVD C	SVD D	SVD E	SVD F	SVD G	SVD H	SVD I	SVD J
RMSE	0.2455	0.2361	0.2410	0.2368	0.2436	0.2452	0.2526	0.2245	0.2289	0.2426
MAE	0.1179	0.1073	0.1118	0.1210	0.1160	0.1095	0.1195	0.0961	0.1004	0.1209
True Positive	21	0	11	31	15	16	31	9	7	23
True Negative	2379	2729	2581	2212	2494	2588	2259	2704	2689	2329
False Positive	356	7	153	544	239	139	490	53	60	413
False Negative	151	171	162	120	159	164	127	141	151	142
Accuracy	0.8256	0.9388	0.8916	0.7716	0.8631	0.8958	0.7878	0.9333	0.9274	0.8091
Precision	0.0557	NAN	0.0671	0.0539	0.0591	0.1032	0.0595	0.1452	0.1045	0.0528
Specificity	0.8698	0.9974	0.9440	0.8026	0.9126	0.9490	0.8218	0.9808	0.9782	0.8494
Sensitivity	0.1221	NAN	0.0636	0.2053	0.0862	0.0889	0.1962	0.0600	0.0443	0.1394
F1 Score	0.0765	NAN	0.0653	0.0854	0.0701	0.0955	0.0913	0.0849	0.0622	0.0765

From Table 18, we can observe that SVD B is the worst-performing model. By its very nature, SVD B is not being considered for the following observations:

Table 18: SVD Hyperparameter Tuning Observations

Observations	
RMSE and MAE	The RMSE values range from 0.2245 to 0.2526 and the MAE values range from 0.0961 to 0.1210. SVD H has both the lowest RMSE value of 0.2245 and the lowest MAE value of 0.0961.
True Positive, True Negative, False Positive, and False Negative	True Positive, representing the number of positive samples correctly classified, ranges from 7 to 31, while True Negative, which represents the number of negative samples correctly classified, ranges from 2212 to 2704. SVD G indicates to have to most positive classifications of 31 and SVD H shows to have the most negative classifications at 2704. False Positive representing the negative samples incorrectly classified as negative ranges from 53 to 544. False Negative, which is the number of positive samples incorrectly classified as negative ranges from 120 to 171. SVD H has the lowest False Positive value of 53 while SVD D has the lowest False Negative value of 120.
Accuracy, Precision, Specificity, Sensitivity, and F1 Score	The model with the highest accuracy of 0.9333 is SVD H (93.33%). SVD H shows to have the highest precision of 0. 1452 (14.52%) while and the highest specificity value at 0.9808 (98.08%). The model with the highest sensitivity is SVD D at 0.2053 (20.53%). Lastly, SVD F shows to have the highest F1 Score, indicating the best balance between precision and sensitivity at 0.0955.
Best Model	As per our observations, it seems that SVD H is the best-performing model with number of factors of 100, number of iterations of 10, a learning rate of 0.01, and a regularization term of 0.1.

Conclusion: SVD H is chosen as the best-performing model for further analysis.

7.2.3 Neural Collaborative Filtering

To illustrate the influence of hyperparameters on the performance of NCF, Table 19 provides a summary of the various options for the hyperparameters and their impacts.

Table 19: NCF Hyperparameter Tuning

NCF Variations	Mf_dim	Mlp_dim	dropout	Learning rate
NCF A	8	[32, 16, 8]	0.2	0.001
NCF B	16	[32, 16, 8]	0.2	0.001
NCF C	32	[32, 16, 8]	0.2	0.001
NCF D	32	[64, 32, 16, 8]	0.2	0.001
NCF E	16	[16, 8]	0.2	0.001
NCF F	16	[8]	0.2	0.001
NCF G	16	[16, 8]	0.4	0.001
NCF H	16	[16, 8]	0.1	0.001
NCF I	16	[16, 8]	0.1	0.01

Table 20 shows the result of performing the proposed model and using NCF as the advanced collaborative filtering algorithm on identical customer IDs.

Table 20: NCF Hyperparameter Tuning Results

	NCF A	NCF B	NCF C	NCF D	NCF E	NCF F	NCF G	NCF H	NCF I
RMSE	0.2488	0.2571	0.2458	0.2524	0.2436	0.2452	0.2407	0.2550	0.2451
MAE	0.1742	0.1969	0.1570	0.1840	0.1063	0.1168	0.1036	0.1912	0.0906
True Positive	18	43	21	25	24	27	13	29	15
True Negative	2457	2051	2453	2390	2533	2450	2658	2265	2577
False Positive	273	679	277	340	197	280	72	465	153
False Negative	159	134	156	152	153	150	164	148	162
Accuracy	0.8514	0.7203	0.8510	0.8308	0.8796	0.8521	0.9188	0.7891	0.8916
Precision	0.0619	0.0596	0.0705	0.0685	0.1086	0.0879	0.1529	0.0587	0.0893
Specificity	0.9000	0.7513	0.8985	0.8755	0.9278	0.8974	0.9736	0.8297	0.9440
Sensitivity	0.1017	0.2429	0.1186	0.1412	0.1356	0.1525	0.0734	0.1638	0.0847
F1 Score	0.0769	0.0957	0.0884	0.0923	0.1206	0.1116	0.0992	0.0864	0.0870

Table 21: NCF Hyperparameter Tuning Observations

Observations	
RMSE and MAE	The RMSE values range from 0.2407 to 0.2571 and the MAE values range from 0.0906 to 0.1969. NCF G has the lowest RMSE value of 0.2407 while NCF I has the lowest MAE value of 0.0906.
True Positive, True Negative, False Positive, and False Negative	True Positive, representing the number of positive samples correctly classified, ranges from 13 to 43, while True Negative, which represents the number of negative samples correctly classified, ranges from 2051 to 2658. NCF B indicates to have to most positive classifications of 43 and NCF G shows to have the most negative classifications at 2658. False Positive representing the negative samples incorrectly classified as negative ranges from 72 to 679. False Negative, which is the number of positive samples incorrectly classified as negative ranges from 134 to 164. NCF G has the lowest False Positive value of 72 while NCF B has the lowest False Negative value of 134.
Accuracy, Precision, Specificity, Sensitivity, and F1 Score	The model with the highest accuracy of 0.9188 is NCF G (91.88%). NCF G also shows to have the highest precision of 0.1525 (15.25%) and the highest specificity value at 0.9736 (97.36%). The model with the highest sensitivity is NCF B at 0.2429 (24.29%). Lastly, NCF E shows to have the highest F1 Score, indicating the best balance between precision and sensitivity at 0.1206.
Best Model	As per our observations, it seems that NCF G is the most performing model with Matrix Factorization dimension of 16, Multi-Layer Perceptron dimension of [16, 8], a dropout of 0.4, and a learning rate of 0.001.

Conclusion: NCF G is chosen as the best-performing model for further analysis.

7.3 Model Evaluation for User-Based Recommendations

7.3.1 Model Evaluation on a Real Dataset

After having meticulously fine-tuned our hyperparameters to optimise the configuration of our models, the following evaluation will enable us to distinguish the best-performing algorithm with respect to our proposed methodology. All the algorithms were employed using the same clustered data and customer IDs.

Table 22: Algorithm Evaluation for User-Based

	NMF	NCF	SVD
RMSE	0.2544	0.2336	0.2303
MAE	0.0762	0.1587	0.1104
True Positive	27	19	10
True Negative	3797	2506	2647
False Positive	76	248	101
False Negative	249	134	149
Accuracy	0.9217	0.8686	0.9140
Precision	0.2621	0.0712	0.0901
Specificity	0.9804	0.9099	0.9632
Sensitivity	0.0978	0.1242	0.0629
F1 Score	0.1425	0.0905	0.0741

Table 23: Algorithm Evaluation for User-Based Observation

Observations	
RMSE and MAE	The SVD algorithm has the lowest RMSE value of 0.2303, which shows it has the smallest average prediction error. On the other hand, the NMF algorithm has the lowest MAE value of 0.0762, indicating the smallest absolute prediction error.
True Positive, True Negative, False Positive, and False Negative	The NMF algorithm appears to have the highest count of True Positive and True Negative at a value of 27 and 3727 compared to NCF at 19,and 2506 and SVD at 10 and 2647 respectively. This indicates that the NMF algorithm performs better at classifying correct positive and negative samples, compared to NCF and SVD. The algorithm with the lowest False Positive value of 76 is the NMF algorithm, while the NCF algorithm obtained the lowest False Negative value of 134. Hence, we can observe that the NMF algorithm has a stronger performance in classifying correct negative samples. Similarly, NCF algorithm shows a better performance in correctly classifying positive samples.
Accuracy, Precision, Specificity, Sensitivity, and F1 Score	The algorithm with the highest overall correctness is the NMF algorithm with an accuracy value of 0.9217 (92.17%). The NMF algorithm also possesses the highest precision value of 0.2621 (26.21%) and the highest specificity value of 0.9804 (98.04%). In terms of sensitivity, we can observe that the NCF algorithm has the highest sensitivity value of 0.1242 (12.42%). Finally, the algorithm with the best balance between precision and sensitivity is the NMF algorithm with an F1 Score of 0.1425.
Best Model	As per our observation, it seems that the NMF is the best-performing algorithm.

Conclusion: NMF is chosen as the best-performing model for our proposed approach to user-based recommendations. This is because NMF is known to work well with data sparsity (Gillis, 2014).

7.3.2 Model Evaluation with and without RFM Analysis and Clustering

The following evaluation aims to assess the efficacy and effectiveness of our proposed solution.

Table 24: Proposed Approach Effect on NMF

	NMF without RFM Analysis and Clustering	NMF with RFM Analysis and Clustering
RMSE	0.2084	0.1812
MAE	0.0640	0.0489
True Positive	50	42
True Negative	17024	13262
False Positive	146	95
False Negative	759	431
Accuracy	0.9497	0.9620
Precision	0.2551	0.3066
Specificity	0.9915	0.9929
Sensitivity	0.0618	0.0888
F1 Score	0.0995	0.1377

Table 25: Proposed Approach Effect on NCF

	NCF without RFM Analysis and Clustering	NCF with RFM Analysis and Clustering
RMSE	0.1991	0.1707
MAE	0.0735	0.0624
True Positive	41	14
True Negative	16531	13254
False Positive	178	29
False Negative	689	405
Accuracy	0.9503	0.9683
Precision	0.1872	0.3256
Specificity	0.9893	0.9978
Sensitivity	0.0562	0.0334
F1 Score	0.0864	0.0606

Table 26: Proposed Approach Effect on SVD

	SVD without RFM Analysis and Clustering	SVD with RFM Analysis and Clustering
RMSE	0.1920	0.1773
MAE	0.0725	0.0648
True Positive	52	42
True Negative	16519	13073
False Positive	260	188
False Negative	608	399
Accuracy	0.9502	0.9572
Precision	0.1667	0.1826
Specificity	0.9845	0.9858
Sensitivity	0.0788	0.0952
F1 Score	0.1070	0.1252

Table 27: Proposed Approach Effect Observations

Observations	
RMSE and MAE	For all NMF, NCF, and SVD, the RMSE and MAE value decreases when using RFM Analysis and Clustering. This indicates that the prediction error improves.
Accuracy, Precision, Specificity, Sensitivity, and F1 Score	For all algorithms, we can observe that the Accuracy, Precision, Specificity, Sensitivity increases which shows that the model performs better when RFM Analysis and Clustering is applied. F1 Score for both NMF and SVD increases which indicates a better balance between precision and sensitivity.

Conclusion: We can conclude that our proposed solution enhances the predictive capabilities of all the algorithms.

7.3.3 Model Evaluation on an Augmented dataset

Table 28 below presents the results for the augmented dataset, where 1,000 fictitious rows were added to the original user-item matrix.

Table 28: Approach Evaluation on 1000 Augmented Rows

	NMF	NCF	SVD
RMSE	0.4995	0.5023	0.4939
MAE	0.4985	0.4837	0.4848
True Positive	101586	94666	96376
True Negative	117567	124175	122508
False Positive	94665	95112	96410
False Negative	109045	108277	106936
Accuracy	0.5183	0.5183	0.5184
Precision	0.5176	0.4988	0.4999
Specificity	0.5540	0.5663	0.5596
Sensitivity	0.4823	0.4665	0.4740
F1 Score	0.4993	0.4821	0.4866

Table 29 below presents the results for the augmented dataset, where 2500 fictitious rows were added to the original user-item matrix.

Table 29: Approach Evaluation on 2500 Augmented Rows

	NMF	NCF	SVD
RMSE	0.4992	0.5034	0.4989
MAE	0.4984	0.4935	0.4938
True Positive	272105	257968	507338
True Negative	264608	265764	14970
False Positive	250239	257825	507736
False Negative	243861	249874	586
Accuracy	0.5207	0.5078	0.5068
Precision	0.5209	0.5001	0.4998
Specificity	0.5140	0.5076	0.0286
Sensitivity	0.5274	0.5080	0.9988
F1 Score	0.5241	0.5040	0.6662

Conclusion: We know that data augmentation is important because it increases data diversity, improves model generalisation, reduces model bias, and enhances robustness, leading to improved model performance. However, in our specific case, when comparing Table 28 and Table 29 with Table 22, data augmentation does not appear to improve the performance of the models and seems to have a negative impact. The addition of noise to the data will not accurately reflect the underlying customer behaviour. This eventually leads to a decrease in prediction accuracy and an increase in RMSE and MAE values.

7.3.4 Model Evaluation on Larger and Real Datasets

Table 30 showcases the results from Ritika Verma's grocery dataset.

Table 30: Approach Evaluation on Ritika Verma's Dataset

	NMF	NCF	SVD
RMSE	0.1807	0.1100	0.1645
MAE	0.0386	0.0300	0.0747
True Positive	2004	2196	1580
True Negative	17496	17728	11369
False Positive	23	134	510
False Negative	640	160	20
Accuracy	0.9671	0.9855	0.9607
Precision	0.9887	0.9425	0.7560
Specificity	0.9987	0.9925	0.9571
Sensitivity	0.7579	0.9321	0.9875
F1 Score	0.8352	0.9373	0.8563

Conclusion: When the recommender system is implemented on a larger dataset that has an underlying consumer pattern, we can observe notable improvements in the performance metrics. The RMSE and MAE values have decreased, which indicates a higher level of accuracy in predicting user-item interactions. Moreover, the results show an increase in accuracy, precision, specificity, sensitivity, and F1 Score. This is due to Ritika Verma's dataset being more diverse in terms of observations, enabling the algorithms to better capture the underlying patterns and preferences of the customers. The improved performance can be attributed to underlying buying patterns, which enable the algorithms to make more reliable and accurate predictions and recommendations. With this dataset, the best-performing algorithm is NCF.

We have encountered in Chapter 3 that the majority of our data collected was without some customer IDs. To augment our dataset, we made use of this anonymous data by assigning each transaction a unique Customer ID. Tables 31, 32, and 33 below show the impact of using a larger dataset on the models.

Table 31: Proposed Approach Evaluation on Larger Data Using NMF

	NMF (Without anonymity)	NMF (with anonymity)
RMSE	0.1495	0.2148
MAE	0.0336	0.0586
True Positive	44	4
True Negative	30287	3942
False Positive	90	10
False Negative	675	193
Accuracy	0.9754	0.9511
Precision	0.3284	0.2857
Specificity	0.9970	0.9975
Sensitivity	0.0612	0.0203
F1 Score	0.1032	0.0379

Table 32: Proposed Approach Evaluation on Larger Data Using NCF

	NCF (Without anonymous data)	NCF (with anonymous data)
RMSE	0.1480	0.2242
MAE	0.0402	0.0955
True Positive	73	8
True Negative	28360	2704
False Positive	406	50
False Negative	595	145
Accuracy	0.9660	0.9329
Precision	0.1524	0.1379
Specificity	0.9859	0.9818
Sensitivity	0.1093	0.0523
F1 Score	0.1273	0.0758

Table 33: Proposed Approach Evaluation on Larger Data Using SVD

	SVD (Without anonymous data)	SVD (with anonymous data)
RMSE	0.1476	0.2357
MAE	0.0455	0.1101
True Positive	57	9
True Negative	27807	2287
False Positive	268	63
False Negative	584	133
Accuracy	0.9703	0.9213
Precision	0.1754	0.1250
Specificity	0.9905	0.9732
Sensitivity	0.0889	0.0634
F1 Score	0.1180	0.0841

Table 34: Proposed Approach Evaluation on Larger Data Observations

Observations	
RMSE and MAE	When removing the anonymity in the dataset, the RMSE and MAE values decreases indicating a better prediction error for all algorithms.
Accuracy, Precision, Specificity, Sensitivity, and F1 Score	For NMF, we can observe that Accuracy, Precision, Sensitivity and F1 Score increases while Specificity decreases slightly. For NCF and SVD, we can see that all the metrics increase in value.

Conclusion: We can conclude that leveraging these anonymous transactions enhances the predictive capabilities of all the algorithms. This proves that just adding random noise to the dataset will severely hinder its capabilities, but increasing the dataset with genuine transactions will provide better predictions.

7.4 Model evaluation for item-based recommendations

All the algorithms were employed using the same target item.

Table 35: Algorithm Evaluation for Item-Based

	NMF	NCF	SVD
RMSE	0.1326	0.1384	0.1330
MAE	0.0239	0.0367	0.0367
True Positive	38	31	26
True Negative	16361	15542	15396
False Positive	65	86	110
False Negative	280	288	262
Accuracy	0.9794	0.9765	0.9764
Precision	0.3689	0.2650	0.1912
Specificity	0.9960	0.9945	0.9929
Sensitivity	0.1195	0.0972	0.0903
F1 Score	0.1805	0.1422	0.1226

Table 36: Algorithm Evaluation for Item-Based Observations

Observations	
RMSE and MAE	The NMF algorithm has the lowest RMSE value of 0.2303, and lowest MAE value of 0.0762, indicating better prediction errors.
True Positive, True Negative, False Positive, and False Negative	NMF shows to have to most true positive classifications of 38 and the most true negative classifications at 16361. NMF has the lowest False Positive value of 65 while SVD has the lowest False Negative value of 262.
Accuracy, Precision, Specificity, Sensitivity, and F1 Score	The algorithm with the highest overall correctness is the NMF algorithm with an accuracy value of 0.9794 (97.94%). The NMF also possesses the highest precision value of 0.3689 (36.89%) and the highest specificity value of 0.9960 (99.60%). In terms of sensitivity, we can observe that the NMF algorithm has also the highest sensitivity value of 0.1195 (11.95%). Finally, the algorithm with the best balance between precision and sensitivity is the NMF algorithm with an F1 Score of 0.1805.
Best Model	As per our observation, it seems that the NMF is the best-performing algorithm for item-based recommendations.

Conclusion: NMF is chosen as the best-performing model for item-based recommendations.

7.5 Overall Findings

Based on the overall results, we are satisfied with the validity of both proposed approaches. It works well with real data and offers promising results with larger datasets. The integration of RFM analysis and clustering into the data indeed improves the performance of the machine learning algorithm.

We've also shown the existence of consumer buying behaviour in our dataset, as the model performs poorly with noise. However, the presence of data sparsity is the cause of low precision and sensitivity. On a positive note, we demonstrate that using implicit ratings rather than explicit ratings improves the predictions when compared to (Sano et al., 2015).

Chapter 8: Testing

This chapter will showcase the implementation of the recommendation system in a web application. The recommendation system aims to enhance the user experience by providing tailored product recommendations based on user and item preferences. The key components, such as item browsing and the user's personal profile, will also be presented.

8.1 Database

The Database contains all the transactions from the collected receipts.

	index_id	transaction_id	date_purchase	customer_id	product	quantity	price	promotional_status
Edit Copy Delete	5268	0	2023-06-27 13:00:25	C11471187	G.FIELD PPACK B/VEAL	1	75	NULL
Edit Copy Delete	5267	0	2023-06-27 13:00:25	C11471187	.SAVANE POCKET CHOCO	2	16.5	NULL
Edit Copy Delete	5266	0	2023-06-27 13:00:25	C11471187	1000 PATES MINI FEUI	2	238.5	NULL
Edit Copy Delete	5265	0	2023-06-27 13:00:25	C11471187	C.CHOICE VEAL LEG IN	2	170	NULL
Edit Copy Delete	5264	2	2023-06-26 18:34:59	C11471187	7 SEAS DRY CANE SPIR	2	19.3	NULL
Edit Copy Delete	5263	0	2023-06-26 18:34:07	C11471187	2-ZERO CHOCO SANDWI	1	45	NULL
Edit Copy Delete	5262	0	2023-06-26 18:34:07	C11471187	KNR SP CHIK NDLE 52G	1	330	NULL
Edit Copy Delete	5261	0	2023-06-26 18:34:07	C11471187	KNORR ECO TOM VE 67G	1	32.5	NULL
Edit Copy Delete	5260	9	2023-06-26 18:33:24	C11471187	15 PCS BALLOON BLACK	2	31	NULL
Edit Copy Delete	5259	9	2023-06-26 18:33:24	C11471187	3 DAMES THE VAN 50G	1	2.6	NULL
Edit Copy Delete	5258	9	2023-06-26 18:29:44	C11471187	1000 PATES MINI FEUI	1	238.5	NULL
Edit Copy Delete	5257	9	2023-06-26 18:29:44	C11471187	.SAVANE POCKET CHOCO	1	16.5	NULL
Edit Copy Delete	1	6	2023-02-16 00:00:00		ESKO GAUFRET VAN 75G	1	16.5	
Edit Copy Delete	2	6	2023-02-16 00:00:00		TIAS GRANOLA RAISIN	1	238.5	
Edit Copy Delete	3	14	2023-01-19 00:00:00		KIT KAT 4F DARK	2	31	
Edit Copy Delete	4	14	2023-01-19 00:00:00		ERASERS X6PCS 8 ASS	1	45	
Edit Copy Delete	5	15	2023-01-19 00:00:00		BREAD CRUMBS CHAPELU	2	29	
Edit Copy Delete	6	15	2023-01-19 00:00:00		PAIN MAISON 100G	1	2.6	
Edit Copy Delete	7	15	2023-01-19 00:00:00		SAC BIO CAISSE	3	4	
Edit Copy Delete	8	15	2023-01-19 00:00:00		KRAFT 250G CHEDDAR	20	80.95	(P)
Edit Copy Delete	9	15	2023-01-19 00:00:00		TR TUNA FL OIL 170G	6	28.95	(P)
Edit Copy Delete	10	16	2023-01-19 00:00:00		ETI PAYKEK 180G FR	1	49.95	(P)
Edit Copy Delete	11	16	2023-01-19 00:00:00		ORANGE FILLET 1KGPRO	1	59.9	(P)
Edit Copy Delete	12	16	2023-01-19 00:00:00		COLG DENT DETOX 75ML	1	62.95	(P)
Edit Copy Delete	13	16	2023-01-19 00:00:00		YOPLAIT YAOURT SAV P	2	12.95	(P)

Figure 146: Transaction Database

8.2 Catalogue Browsing

Any user can have access to the items and their corresponding prices.

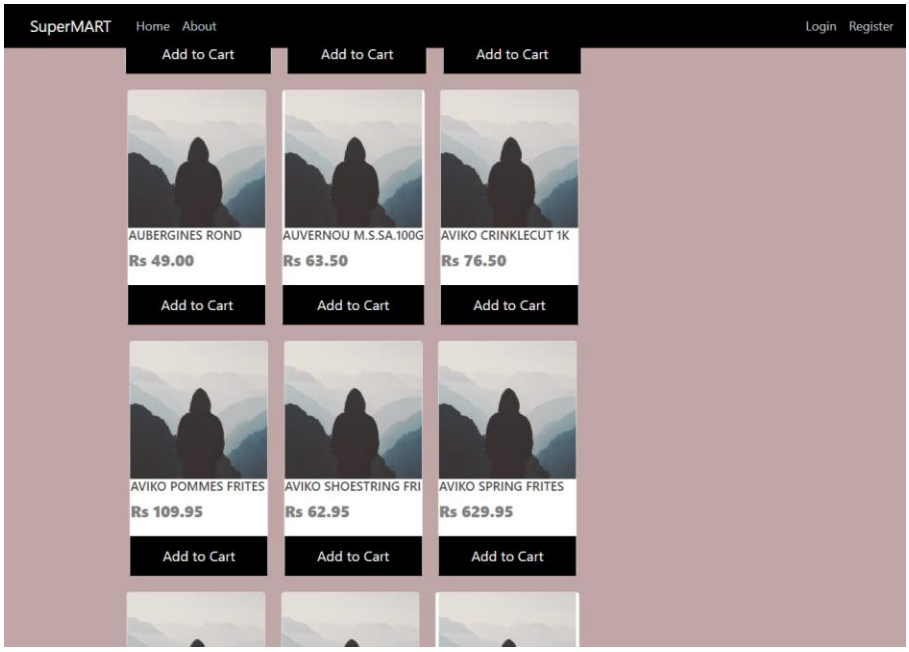


Figure 147: Browsing products

8.3 Item Browsing

Only Registered users can have access to a detailed overview of any selected item.

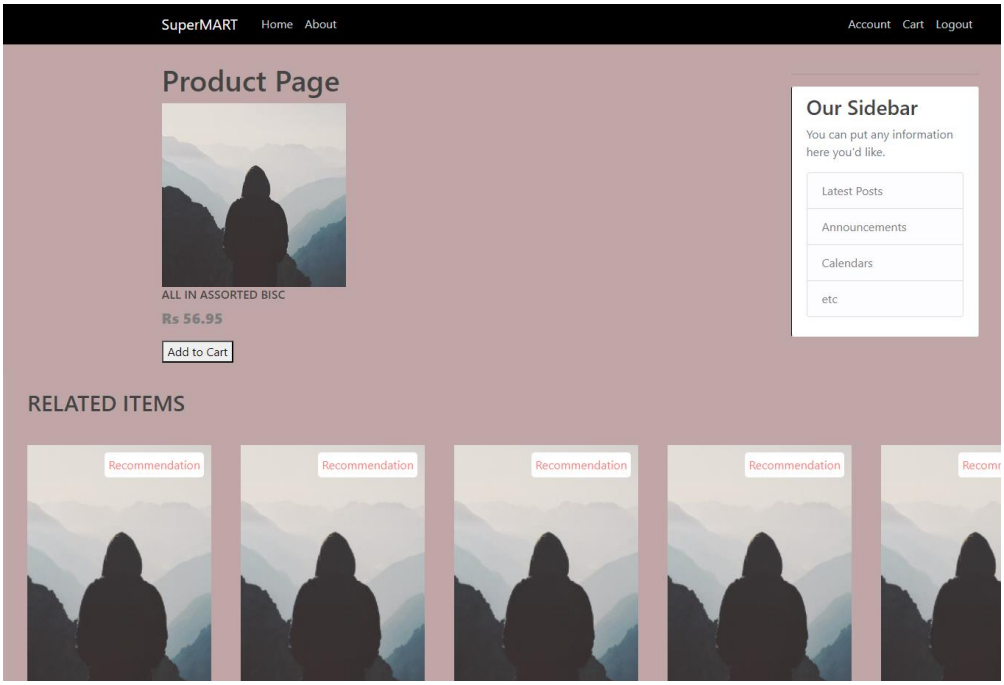


Figure 148: Inspecting a Target Item

8.4 Item-based Recommendations

The recommendations are generated based on the transactions that contain the selected item.

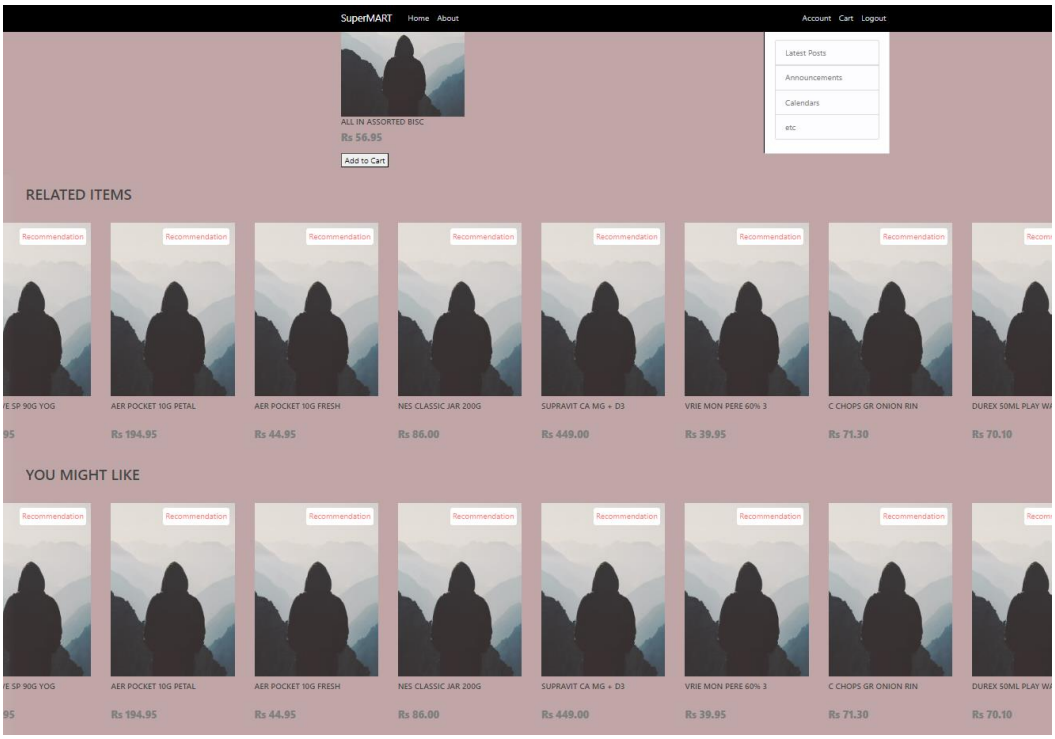


Figure 149: Recommendations based on Target Item

8.5 User Personal Profile

Registered users can have access to their personal information.

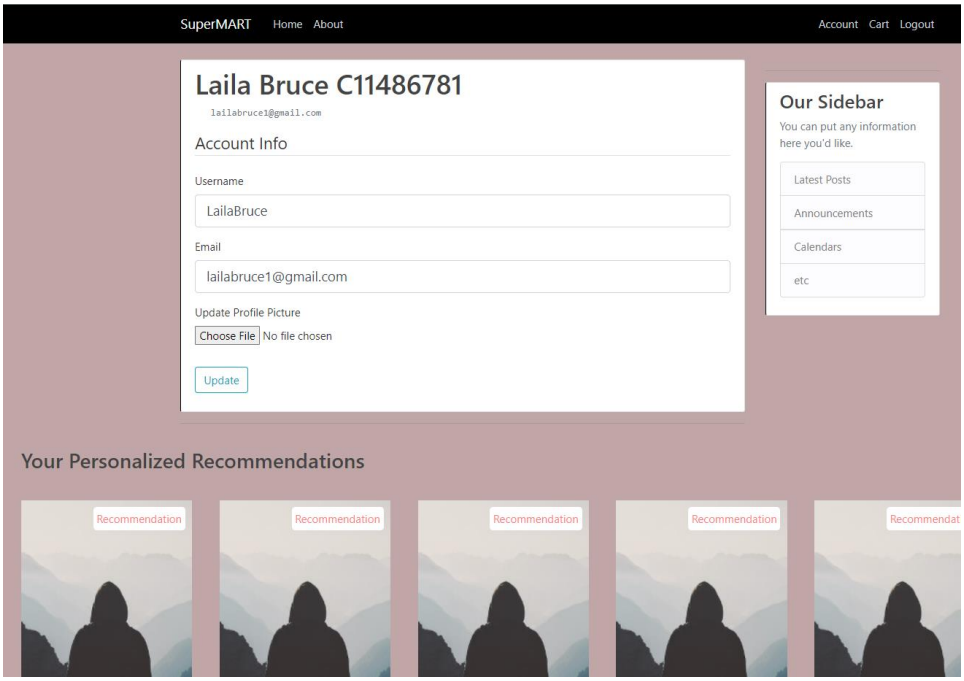


Figure 150: User profile

8.6 User-based Recommendations

The recommendations are generated based on the transactions of the current user and customers with similar buying behaviour.

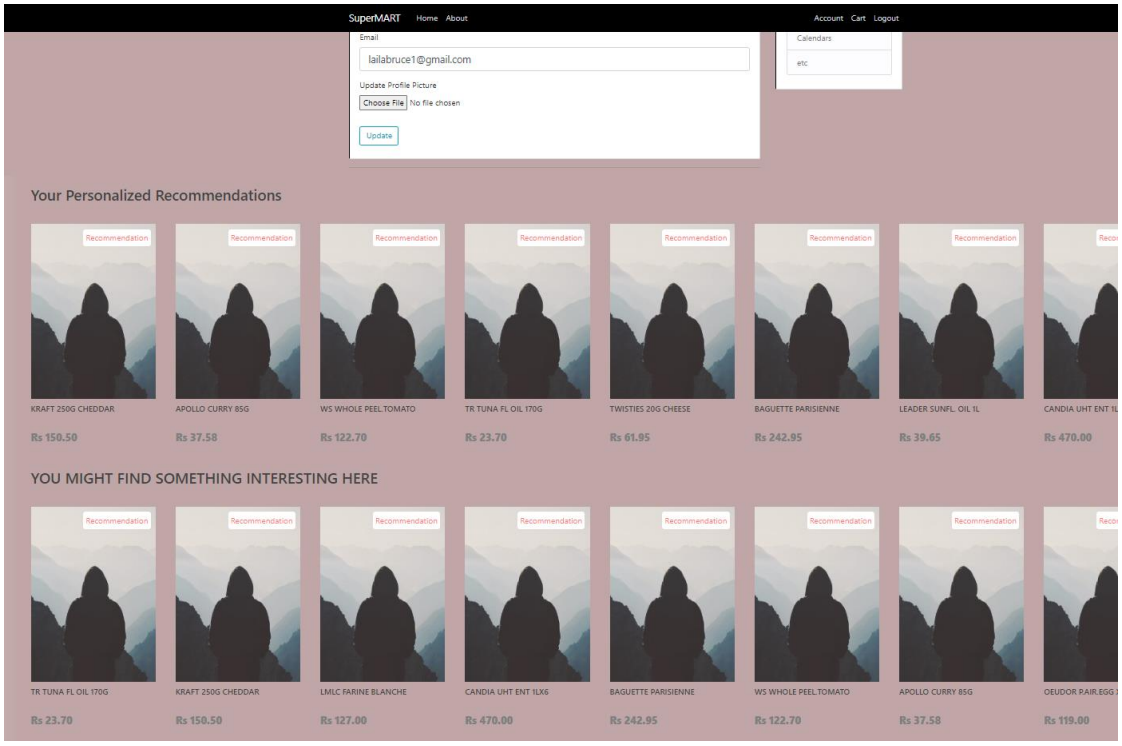


Figure 151: Recommendations based on Target User

8.7 Cart

The Cart allows the user to select the quantity of the items they intend to buy.

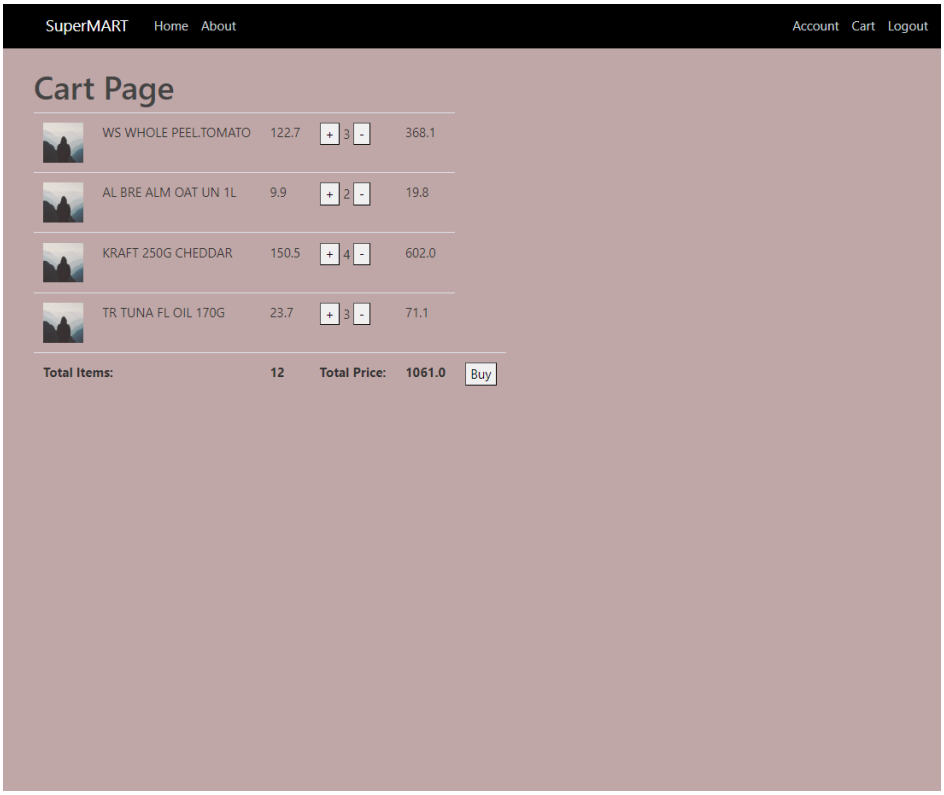


Figure 152: Adding Items to the Cart

8.8 Purchase

Making the purchase from the cart.

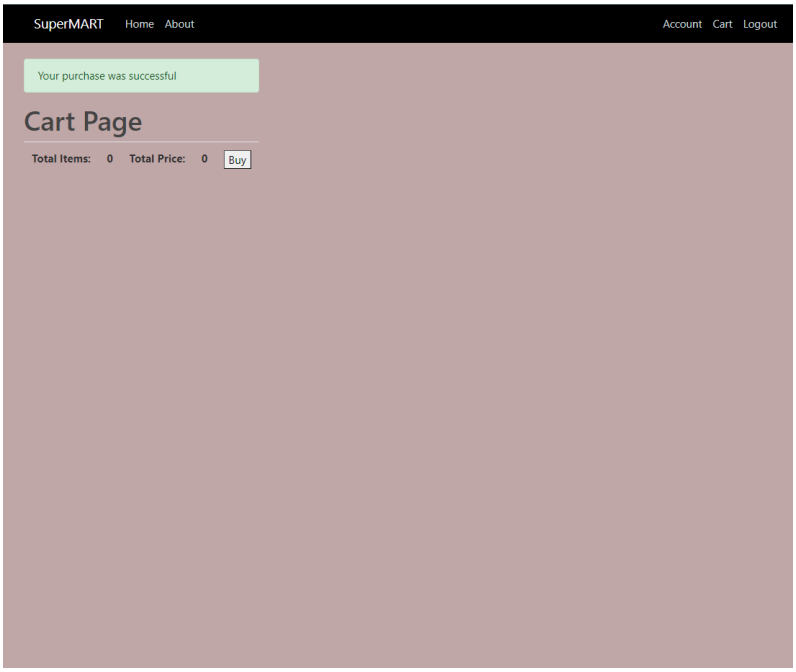


Figure 153: Making Purchases

Once the purchase is made, the transaction table is updated with the new purchases.

	index_id	transaction_id	date_purchase	customer_id	product	quantity	price	promotional_status
Edit Copy Delete	5271	0	2023-07-23 16:37:12	C11486781	KRAFT 250G CHEDDAR	4	150.5	NULL
Edit Copy Delete	5270	0	2023-07-23 16:37:12	C11486781	AL BRE ALM OAT UN 1L	2	9.9	NULL
Edit Copy Delete	5272	0	2023-07-23 16:37:12	C11486781	TR TUNA FL OIL 170G	3	23.7	NULL
Edit Copy Delete	5269	0	2023-07-23 16:37:12	C11486781	WS WHOLE PEEL TOMATO	3	122.7	NULL
Edit Copy Delete	5268	0	2023-06-27 13:00:25	C11471187	G.FIELD PPACK B/VEAL	1	75	NULL
Edit Copy Delete	5267	0	2023-06-27 13:00:25	C11471187	.SAVANE POCKET CHOCO	2	16.5	NULL
Edit Copy Delete	5266	0	2023-06-27 13:00:25	C11471187	1000 PATES MINI FEUI	2	238.5	NULL
Edit Copy Delete	5265	0	2023-06-27 13:00:25	C11471187	C.CHOICE VEAL LEG IN	2	170	NULL
Edit Copy Delete	5264	2	2023-06-26 18:34:59	C11471187	7 SEAS DRY CANE SPIR	2	19.3	NULL
Edit Copy Delete	5263	0	2023-06-26 18:34:07	C11471187	2-ZERO CHOCO SANDWI	1	45	NULL
Edit Copy Delete	5262	0	2023-06-26 18:34:07	C11471187	KNR SP CHIK NDLE 52G	1	330	NULL
Edit Copy Delete	5261	0	2023-06-26 18:34:07	C11471187	KNORR ECO TOM VE 67G	1	32.5	NULL
Edit Copy Delete	5260	9	2023-06-26 18:33:24	C11471187	15 PCS BALLOON BLACK	2	31	NULL
Edit Copy Delete	5259	9	2023-06-26 18:33:24	C11471187	3 DAMES THE VAN 50G	1	2.6	NULL
Edit Copy Delete	5258	9	2023-06-26 18:29:44	C11471187	1000 PATES MINI FEUI	1	238.5	NULL
Edit Copy Delete	5257	9	2023-06-26 18:29:44	C11471187	.SAVANE POCKET CHOCO	1	16.5	NULL
Edit Copy Delete	5184	250	2023-03-22 00:00:00	C23642631	SAVON CITRON PLUS 1K	1	168.95	(P)
Edit Copy Delete	5193	250	2023-03-22 00:00:00	C23642631	G.FIELD PP BUFFALO	1	159.45	(P)
Edit Copy Delete	5192	250	2023-03-22 00:00:00	C23642631	WHITE TOAST BREAD	1	49	
Edit Copy Delete	5190	250	2023-03-22 00:00:00	C23642631	ZESS CRACKER SWICH C	1	69	
Edit Copy Delete	5189	250	2023-03-22 00:00:00	C23642631	GLENRYCK 215G PILCHA	1	59.63	
Edit Copy Delete	5196	250	2023-03-22 00:00:00	C23642631	HS SHP 2EN1 ITCHYSCA	5	249	
Edit Copy Delete	5188	250	2023-03-22 00:00:00	C23642631	SPRITE 1.5L	1	63	
Edit Copy Delete	5187	250	2023-03-22 00:00:00	C23642631	CANDIA UHT D.EC 1LX6	1	51.95	(P)
Edit Copy Delete	5194	250	2023-03-22 00:00:00	C23642631	3 HORSE DARK MALTA 3	1	55	

Figure 154: Updated Transaction Database

8.9 Cold-Start Issue for User-Based Recommendations

Information about new users is not available, and therefore, the algorithm cannot make any predictions or recommendations.

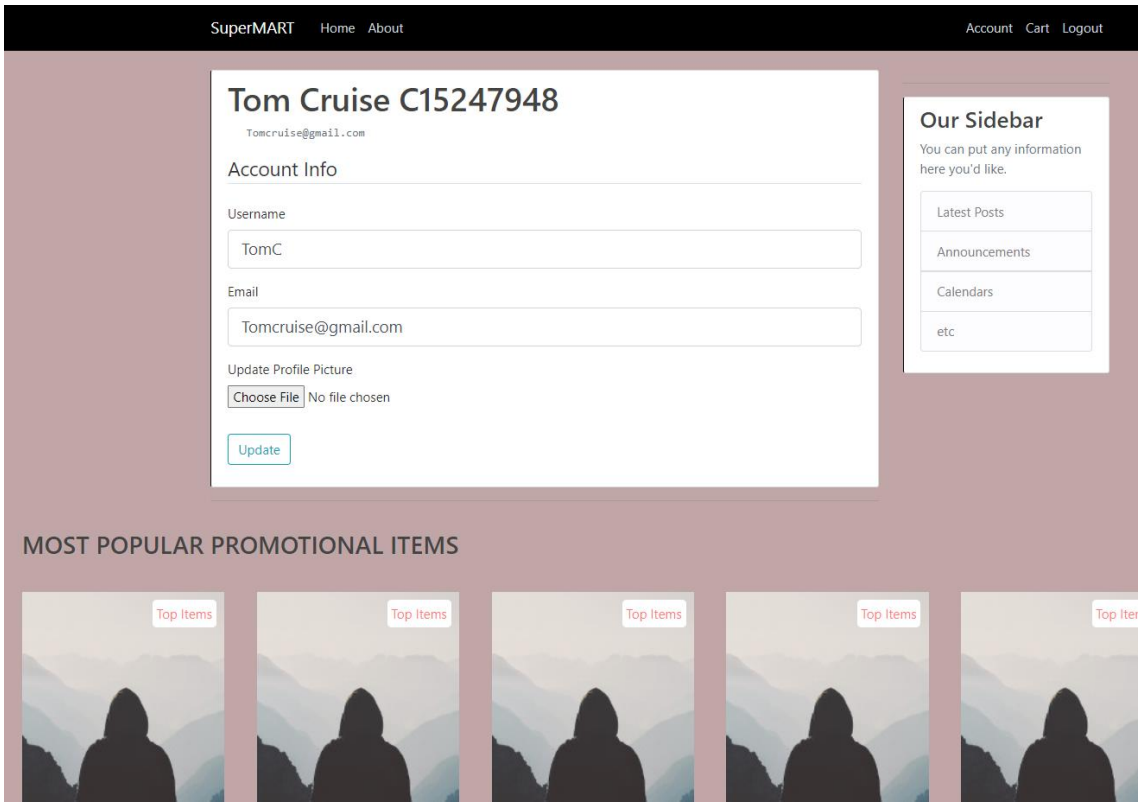


Figure 155: New User

To address this issue, the suggested items for the new customer are the most popular promotional products and the most popular non-promotional products.

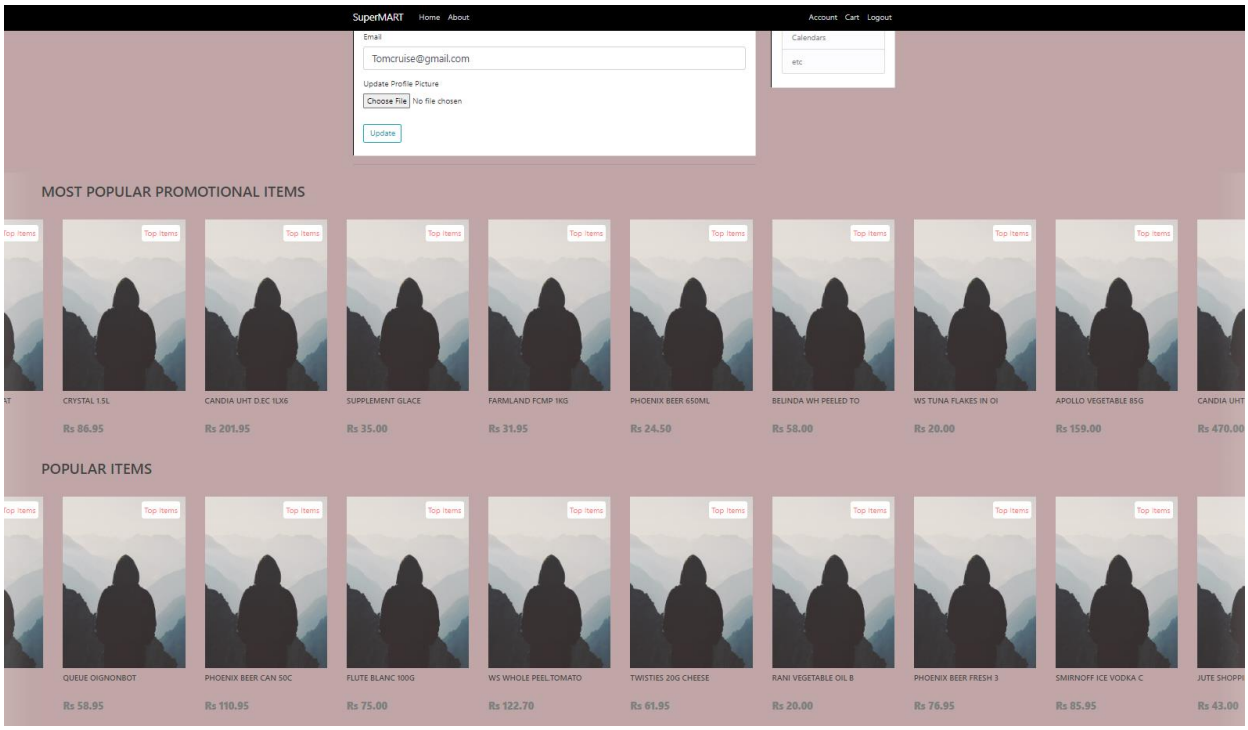


Figure 156: Addressing the Cold-Start Issue

Chapter 9: Conclusion

In this chapter, we give an evaluation of the proposed approach for grocery recommendation systems. We outline our suggested system's accomplishments and explore the potential benefits for grocery stores. We also discuss the difficulties experienced during the study process, which includes data gathering, data comprehension, implementation, and testing. Furthermore, we assess the system's limits in order to offer a fair perspective on its capabilities and potential areas for development. The chapter also explains how the implemented application will be enhanced in the future.

9.1 Achievements

- Through an extensive literature review, a thorough grasp of recommendation systems was achieved, including numerous algorithmic techniques and their applications. With this solid theoretical base, it enabled the formulation of novel strategies for personalised grocery predictions.
- A deep understanding of machine learning techniques was developed through significant research. This complete knowledge enabled the creation of a distinctive, sophisticated predictive model for grocery suggestions, displaying excellent predictive efficacy and accuracy.
- Significant work was put into gathering and selecting relevant information from various sources. A careful assessment of data quality and a preprocessing workflow assured dataset integrity, giving a solid foundation for model development.
- This study presents a unique recommendation strategy suited for the supermarket domain that uses RFM Analysis, Clustering, and an advanced collaborative filtering algorithm.
- Extensive testing revealed that including RFM analysis and clustering prior to the NMF, SVD, and NCF improved system performance, demonstrating superiority in accuracy, precision, sensitivity, and specificity. The study offered quantitative proof of the approach's superiority, bolstering its practicality and relevance in real-world supermarket suggestion settings.
- As a practical example of the research's usefulness, the algorithms and system were smoothly incorporated into a user-friendly online application.

9.2 Difficulties encountered

- Gaining access to supermarket company data proved to be a considerable challenge, as all establishments were unwilling to provide their sensitive information, which hampered the acquisition of complete and diverse datasets, limiting the research scope and generalization.
- The survey was ineffective because the target population was uninterested in devoting time to filling out the questions.

- The extraction of relevant information from supermarket receipts was tedious and time-consuming due to the complex and unstructured nature of the data, which required rigorous efforts in preprocessing to assure its accuracy and suitability for analysis.
- Understanding the underlying principles of NMF, SVD, and NCF was demanding, and these unsupervised learning algorithms required adaptation to be effectively trained and tested.
- Integrating the complete recommendation approach into a user-friendly web application involved a range of technical challenges.
- Conducting user acceptance testing was not possible, mainly due to the complex and unique societal nature of grocery shopping.

9.3 Limitations

- The approach was constrained to using data from a single supermarket franchise, restricting the generalization of the recommendations to a broader range of supermarkets.
- Due to data limitations, the dataset consisted of around 5,000 transactions.
- Customers' age, wage, employment, ethnicity, marital status, family type, health condition, residence, dietary habits, and shopping habits were not included in the demographic data.
- Only three models were examined in the research, leaving out additional interesting algorithms that could improve suggestion accuracy.
- In the absence of product categories, category-specific recommendations were not possible.

9.4 Future Works

- A future study might investigate the approach's scalability to include the complete supermarket database as well as its performance on a bigger and more diversified dataset.
- Testing the approach on multiple supermarket franchises can validate its adaptability and effectiveness across different retail environments.
- Demonstrate the recommendation system's adaptability in another area, such as clothing or electronics.
- Including the previously specified non-included demographic data can result in more customized and personalised recommendations.
- To broaden the scope of the research, investigate various matrix factorization approaches such as ALS, PMLF, WRMF, BiasedMF, and NMTF.

References

1. Abirami, M., Pattabiraman, V., 2016. Data Mining Approach for Intelligent Customer Behavior Analysis for a Retail Store, in: Vijayakumar, V., Neelanarayanan, V. (Eds.), Proceedings of the 3rd International Symposium on Big Data and Cloud Computing Challenges (ISBCC – 16'), Smart Innovation, Systems and Technologies. Springer International Publishing, Cham, pp. 283–291. https://doi.org/10.1007/978-3-319-30348-2_23

2. Adaji, I., Oyibo, K., Vassileva, J., 2018. Shopping Value and its Influence on Healthy Shopping Habits in E-Commerce.

3. Aichner, T., Coletti, P., 2013. Customers' online shopping preferences in mass customization. J Direct Data Digit Mark Pract 15, 20–35. https://doi.org/10.1057/dddmp.2013.34

4. Amazon annual net sales 2022 [WWW Document], n.d. . Statista. URL https://www.statista.com/statistics/266282/annual-net-revenue-of-amazoncom/ (accessed 7.24.23).

5. Bahari, T.F., Elayidom, M.S., 2015. An Efficient CRM-Data Mining Framework for the Prediction of Customer Behaviour. Procedia Computer Science 46, 725–731. https://doi.org/10.1016/j.procs.2015.02.136

6. Brits spent £12.3 billion on online groceries in 2018 [WWW Document], n.d. URL https://www.mintel.com/press-centre/brits-spent-12-3-billion-on-online-groceries-in-2018/ (accessed 7.24.23).

7. B.Thorat, P., M. Goudar, R., Barve, S., 2015. Survey on Collaborative Filtering, Content-based Filtering and Hybrid Recommendation System. IJCA 110, 31–36. https://doi.org/10.5120/19308-0760

8. Eckart, C., Young, G., 1936. The approximation of one matrix by another of lower rank. Psychometrika 1, 211–218. https://doi.org/10.1007/BF02288367

9. Figure 3. Interest-Aware Location-Based Recommender system (IALBR)... [WWW Document], n.d. . ResearchGate. URL https://www.researchgate.net/figure/Interest-Aware-Location-Based-Recommender-system-IALBR-system-architecture_fig3_311734196 (accessed 7.24.23).

10. Foxall, G.R., 2001. Foundations of Consumer Behaviour Analysis. Marketing Theory 1, 165–199. https://doi.org/10.1177/147059310100100202

11. Gao, H., Xu, Y., Yin, Y., Zhang, W., Li, R., Wang, X., 2020. Context-Aware QoS Prediction With Neural Collaborative Filtering for Internet-of-Things Services. IEEE Internet Things J. 7, 4532–4542. https://doi.org/10.1109/JIOT.2019.2956827

12. George Adamides, Giannakopoulou Marianthi, Savvas Savvides, 2006. Traditional Vs Online Attitudes Towards Grocery Shopping In Cyprus, in: Computers in Agriculture and Natural Resources, 23-25 July 2006, Orlando Florida. Presented at the Computers in Agriculture and Natural Resources, 23-25 July 2006, Orlando Florida, American Society of Agricultural and Biological Engineers. https://doi.org/10.13031/2013.21848

13. Gillis, N., 2014. The Why and How of Nonnegative Matrix Factorization. Regularization, Optimization, Kernels, and Support Vector Machines 12.

14. Gillis, N., n.d. Nonnegative Matrix Factorization.

15. Grocery Store Statistics: Where, When, & How Much People Grocery Shop [WWW Document], n.d. URL https://www.driveresearch.com/market-research-company-blog/grocery-store-statistics-where-when-how-much-people-grocery-shop/ (accessed 6.29.23).

16. groceryData - dataset by rit-17 [WWW Document], n.d. . data.world. URL https://data.world/rit-17/grocerydata (accessed 7.25.23).

17. Has the Future Started? The Current Growth of Artificial Intelligence, Machine Learning, and Deep Learning, 2022. . ijcsm 115–123. https://doi.org/10.52866/ijcsm.2022.01.01.013

18. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.-S., 2017. Neural Collaborative Filtering.

19. Helmi, A., 2016. Study of Shopping Style as Expressions of Personal Values 5.

20. Hotz, N., 2018. What is CRISP DM? Data Science Process Alliance. URL https://www.datascience-pm.com/crisp-dm-2/ (accessed 7.25.23).

21. How Artificial Intelligence Powers Personalized Shopping [WWW Document], n.d. . Salesforce. URL https://www.salesforce.com/products/commerce-cloud/resources/personalized-shopping/ (accessed 7.25.23).

22. How Collaborative Filtering Works in Recommender Systems [WWW Document], n.d. URL https://www.turing.com/kb/collaborative-filtering-in-recommender-system (accessed 7.25.23).

23. How retailers can keep up with consumers | McKinsey [WWW Document], n.d. URL https://www.mckinsey.com/industries/retail/our-insights/how-retailers-can-keep-up-with-consumers (accessed 7.25.23).

24. Isinkaye, F.O., Folajimi, Y.O., Ojokoh, B.A., 2015. Recommendation systems: Principles, methods and evaluation. Egyptian Informatics Journal 16, 261–273. https://doi.org/10.1016/j.eij.2015.06.005

25. Khade, A.A., 2016. Performing Customer Behavior Analysis using Big Data Analytics. Procedia Computer Science 79, 986–992. https://doi.org/10.1016/j.procs.2016.03.125

26. Koren, Y., Bell, R., Volinsky, C., 2009. Matrix Factorization Techniques for Recommender Systems. Computer 42, 30–37. https://doi.org/10.1109/MC.2009.263

27. Kumar, P., Thakur, R.S., 2018. Recommendation system techniques and related issues: a survey. Int. j. inf. tecnol. 10, 495–501. https://doi.org/10.1007/s41870-018-0138-8

28. Lakshmi, S.S., Lakshmi, D.T.A., 2014. Recommendation Systems:Issues and challenges 5.

29. Lavelle-Hill, R., Skatova, A., Goulding, J., Bibby, P., Clarke, D., 2020. Buying what people like you buy: Personality Homophily and Well-being in Consumer Behaviour. https://doi.org/10.31219/osf.io/nxsy9

30. Linden, G., Smith, B., York, J., 2003. Amazon.com recommendations: item-to-item collaborative filtering. IEEE Internet Comput. 7, 76–80. https://doi.org/10.1109/MIC.2003.1167344

31. Lü, L., Medo, M., Yeung, C.H., Zhang, Y.-C., Zhang, Z.-K., Zhou, T., 2012. Recommender systems. Physics Reports 519, 1–49. https://doi.org/10.1016/j.physrep.2012.02.006

32. Markowska-Kaczmar, U., Kwasnicka, H., Paradowski, M., 2010. Intelligent Techniques in Personalization of Learning in e-Learning Systems, in: Xhafa, F., Caballé, S., Abraham, A., Daradoumis, T., Juan Perez, A.A. (Eds.), Computational Intelligence for Technology Enhanced Learning, Studies in Computational Intelligence. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 1–23. https://doi.org/10.1007/978-3-642-11224-9_1

33. McPherson, M., Smith-Lovin, L., Cook, J.M., 2001. Birds of a Feather: Homophily in Social Networks. Annu. Rev. Sociol. 27, 415–444. https://doi.org/10.1146/annurev.soc.27.1.415

34. Mitropoulou, K., Kokkinos, P., Soumplis, P., Varvarigos, E., 2022. Detect Resource Related Events in a Cloud-Edge Infrastructure using Knowledge Graph Embeddings and Machine Learning, in: 2022 13th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP). Presented at the 2022 13th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP), IEEE, Porto, Portugal, pp. 698–703. https://doi.org/10.1109/CSNDSP54353.2022.9908022

35. New Epsilon research indicates 80% of consumers are more likely to make a purchase when brands offer personalized experiences [WWW Document], n.d. URL https://www.epsilon.com/us/about-us/pressroom/new-epsilon-research-indicates-80-of-consumers-are-more-likely-to-make-a-purchase-when-brands-offer-personalized-experiences (accessed 7.25.23).

36. Nouh, R.M., Lee, H.-H., Lee, W.-J., Lee, J.-D., 2019. A Smart Recommender Based on Hybrid Learning Methods for Personal Well-Being Services. Sensors 19, 431. https://doi.org/10.3390/s19020431

37. Online grocery shopping offers convenience, health benefits [WWW Document], n.d. . www.heart.org. URL https://www.heart.org/en/healthy-living/healthy-eating/eat-smart/nutrition-basics/online-grocery-shopping-offers-convenience-health-benefits (accessed 7.24.23).

38. Park, S.-H., Han, S.P., 2013. From Accuracy to Diversity in Product Recommendations: Relationship Between Diversity and Customer Retention. International Journal of Electronic Commerce 18, 51–72. https://doi.org/10.2753/JEC1086-4415180202

39. Pathak, B., Garfinkel, R., Gopal, R.D., Venkatesan, R., Yin, F., 2010. Empirical Analysis of the Impact of Recommender Systems on Sales. Journal of Management Information Systems 27, 159–188. https://doi.org/10.2753/MIS0742-1222270205

40. Pennings, J.M.E., van Ittersum, K., Wansink, B., n.d. To Spend or Not To Spend? The Effect of Budget Constraints on Estimation Processes and Spending Behavior.

41. Raju, S.S., Dhandayudam, P., 2018. Prediction of customer behaviour analysis using classification algorithms. Presented at the INTERNATIONAL CONFERENCE ON ELECTRICAL, ELECTRONICS, MATERIALS AND APPLIED SCIENCE, Secunderabad, India, p. 020098. https://doi.org/10.1063/1.5032060

42. Recommendation Engine Market Size & Share Analysis - Industry Research Report - Growth Trends [WWW Document], n.d. URL https://www.mordorintelligence.com/industry-reports/recommendation-engine-market (accessed 7.25.23).

43. Ross, L., 2019. The Importance of Cross Selling and E-commerce Product... Invesp. URL https://www.invespcro.com/blog/e-commerce-product-recommendations/ (accessed 7.25.23).

44. Sano, N., Machino, N., Yada, K., Suzuki, T., 2015. Recommendation System for Grocery Store Considering Data Sparsity. Procedia Computer Science 60, 1406–1413. https://doi.org/10.1016/j.procs.2015.08.216

45. Shah, S., Patel, Y., Panchal, K., Gandhi, P., Patel, P., Desai, A., 2021. Python and MySQL based Smart Digital Retail Management System, in: 2021 6th International Conference for Convergence in Technology (I2CT). Presented at the 2021 6th International Conference for Convergence in Technology (I2CT), IEEE, Maharashtra, India, pp. 1–6. https://doi.org/10.1109/I2CT51068.2021.9417913

46. Shier, V., Miller, S., Datar, A., 2022. Heterogeneity in grocery shopping patterns among low-income minority women in public housing. BMC Public Health 22, 1612. https://doi.org/10.1186/s12889-022-14003-0

47. Singh, D., Reddy, C.K., 2015. A survey on platforms for big data analytics. Journal of Big Data 2, 8. https://doi.org/10.1186/s40537-014-0008-6

48. Sohil, F., Sohali, M.U., Shabbir, J., 2022. An introduction to statistical learning with applications in R: by Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani, New York, Springer Science and Business Media, 2013, $41.98, eISBN: 978-1-4614-7137-7. Statistical Theory and Related Fields 6, 87–87. https://doi.org/10.1080/24754269.2021.1980261

49. The Impact of Product Recommendations [WWW Document], n.d. . Insider Intelligence. URL https://www.insiderintelligence.com/content/the-impact-of-product-recommendations (accessed 7.25.23).

50. The next S-curve of growth: Online grocery to 2030 | McKinsey [WWW Document], n.d. URL https://www.mckinsey.com/industries/retail/our-insights/the-next-s-curve-of-growth-online-grocery-to-2030 (accessed 7.26.23).

51. The value of getting personalization right—or wrong—is multiplying | McKinsey [WWW Document], n.d. URL https://www.mckinsey.com/capabilities/growth-marketing-and-sales/our-insights/the-value-of-getting-personalization-right-or-wrong-is-multiplying (accessed 7.25.23).

52. Tyrväinen, O., Karjaluoto, H., 2022. Online grocery shopping before and during the COVID-19 pandemic: A meta-analytical review. Telematics and Informatics 71, 101839. https://doi.org/10.1016/j.tele.2022.101839

53. US eGrocery Sales Predictions for Business Growth in 2023 and Beyond [WWW Document], n.d. URL https://www.mercatus.com/resources/egrocery-strategy/us-egrocery-sales-predictions-to-improve-business-growth-in-2023-and-beyond/ (accessed 7.24.23).

54. Von Luxburg, U., 2007. A tutorial on spectral clustering. Stat Comput 17, 395–416. https://doi.org/10.1007/s11222-007-9033-z

55. Wei, F., Zhang, Q., 2018. Design and Implementation of Online Shopping System Based on B/S Model. MATEC Web Conf. 246, 03033. https://doi.org/10.1051/matecconf/201824603033

56. Witcher, B., 2018. Transform Your Personalization Strategy At Forrester's Consumer Marketing Forum. Forrester. URL https://www.forrester.com/blogs/transform-your-personalization-strategy-at-forresters-consumer-marking-forum/ (accessed 7.25.23).

57. Woodruffe-Burton, H., Wakenshaw, S., 2011. Revisiting experiential values of shopping: consumers' self and identity. Marketing Intelligence & Planning 29, 69–85. https://doi.org/10.1108/02634501111102760

58. Wu, Y.-J., Teng, W.-G., 2011. An enhanced recommendation scheme for online grocery shopping, in: 2011 IEEE 15th International Symposium on Consumer Electronics (ISCE). Presented at the 2011 IEEE 15th International Symposium on Consumer Electronics - (ISCE 2011), IEEE, Singapore, Singapore, pp. 410–415. https://doi.org/10.1109/ISCE.2011.5973860

59. Xu, D., Tian, Y., 2015. A Comprehensive Survey of Clustering Algorithms. Ann. Data. Sci. 2, 165–193. https://doi.org/10.1007/s40745-015-0040-1

60. YongchangWang, Zhu, L., 2017. Research and implementation of SVD in machine learning, in: 2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS). Presented at the 2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS), IEEE, Wuhan, pp. 471–475. https://doi.org/10.1109/ICIS.2017.7960038

61. Zhang, X., Li, S., Burke, R.R., Leykin, A., 2014a. An Examination of Social Influence on Shopper Behavior Using Video Tracking Data. Journal of Marketing 78, 24–41. https://doi.org/10.1509/jm.12.0106

Annex 1

ANNEX 1

UNIVERSITY OF MAURITIUS
FACULTY/ CENTRE ……FOICDT……..

PROJECT PROPOSAL/ SYNOPSIS

Department ……Software and Information Systems……

Academic Year ……3……

Students are hereby informed that they should submit this document (approximately 200 words) to their respective Project/ Dissertation/ Programme Coordinators by one month as from the beginning of Semester 1 at latest.

Student's Name: ……Ronnish Yaansh Rajanah……

Student ID: ……2012054……

Title of dissertation: ……Diving into Data Science: A Novel Approach to Personalised Grocery Recommendation Systems in Mauritius……

Aims and Objectives:

1. Identify the various types of recommendation systems, their efficacy, main problems, and prospects for recommendation system development.
2. Gather and pre-process customer data on customer behaviour, purchasing history, and other relevant factors to understand their shopping behaviour and preferences for the development of the recommendation system.
3. Design and implement a personalised grocery recommendation system based on customer data by comparing different machine learning algorithms.
4. Evaluate the effectiveness of the developed recommendation system in terms of accuracy, precision, sensitivity, and other relevant metrics.
5. Provide recommendations for future improvements and enhancements to the recommendation system.

Proposed Methodology (tentative):

The project follows the CRISP-DM life cycle. The project outline must be first established, then the data is collected and understood. The dataset is prepared and several machine learning techniques are applied to it. These experiments are recorded and the evaluation reveals insights about the data and the model. The best model is then deployed on a web application to showcase how the system is implemented.

Frequency of meeting with supervisor(s)

Start of Project

7 Dec 2022

Frequency : Every week

End of Project

27 Jul 2023

Comments, if any Some meetings were online while some were on campus

GANTT CHART

	DEC	JAN	FEB	MAR	APR	MAY	JUN	JUL
	~~Aug~~	~~Sept~~	~~Oct~~	~~Nov~~	~~Dec~~	~~Jan~~	~~Feb~~	~~Mar~~
Introduction	X							
Background Study		X	X	X				
Data Understanding, Collection, and ~~Preparation~~ Analysis	X	X	X	X				
				X	X			
Design				X	X			
Implementation					X	X	X	X
Results, Evaluation and Discussion							X	X
Testing							X	X
Conclusion								X

Student's Signature: .....

Date: 25/07/2023

Supervisor's Name: Mr Somveer KISHNAH

Supervisor's Signature: .....

Date: 26.07.2023

Annex 2

ANNEX 2

UNIVERSITY OF MAURITIUS

FACULTY/ CENTRE FOICDT .....

PROGRESS LOG

Student Name	:	Ronnish Yaansh Rajanah
Student ID	:	2012054
Department	:	Software and Information Systems
Programme	:	BSc (Hons) Data Science
Title of Dissertation	:	Diving into Data Science: A Novel Approach to Personalized Grocery Recommendation Systems in Mauritius.
Supervisor(s)	:	Mr Somveer Kishnah

Project/ Dissertation/ Programme Coordinator : Dr. Gobin-Rahimbux Baby Ashwin

- Your Progress Log serves as a record of your transferable skills and participation and attainment as a student for dissertation purposes.
- Its purpose is to help you to plan your own dissertation and to record the outcomes.
- As well as gaining valuable skills, you will find that the information accumulated in this Log will prove helpful during the write up of the dissertation.
- The document belongs to you and it is your responsibility to keep it up to date.
- It is your responsibility to ensure your supervisor(s) is/are aware of the dissertation activities you have undertaken.

You should sign the appropriate statement below when you submit your Progress Log:

I confirm that the information I have given in this Log is a true and accurate record:

Signed: .....

Date: 25/07/2023 .....

PROGRESS LOG

RECORD OF STRATEGIC MEETINGS WITH SUPERVISOR(S)

Meetings	Date	Topics/ Themes Discussed	Comments (If any)	Supervisor's Initials	Student's Initials
1	7/12/22	Overview of the project and work planning			RYR
2	15/12/22	Project Description and research to conduct			
3	21/12/22	Understanding data required			RYR
4	14/01/23	Compromise and problem reassessment			
5	26/01/23	Update about data gathering and research			RYR
6	2/02/23	Introduction to new topic for research			
7	8/02/23	Analysis chapter discussion			RYR
8	16/02/23	Update on data pre-processing and preparation			
9	3/03/23	Design chapter discussion			RYR
10	18/03/23	Answering questions			
11	27/03/23	Analysis Chapter update			RYR
12	10/04/23	Implementation chapter discussion			
13	19/04/23	Design chapter update			RYR
14	8/05/23	Answering questions			
15	19/05/23	Update on design chapter			RYR
16	25/05/23	Answering questions			
17	5/06/23	Update on implementation			RYR
18	29/06/23	Customer buying pattern explanation			
19	14/07/23	New Dataset Acquisition			RYR
20	21/07/23	Final update			

Supervisor(s)	Signature(s)	Date
Mr Somveer KISHNAH		26.07.2023
.....	.....	.....

N.B: Both the supervisor(s) and the student should retain a copy of this Project Progress Log.
A copy of the duly filled and signed Progress Log should be **included and submitted in the section 'Appendices'** of the Dissertation.

Appendix 1: Types of grocery shopping

In-Store shopping

In-store shopping involves physically visiting a store to buy goods, allowing customers to see, touch, compare, and receive assistance with their purchases. It remains popular, particularly for groceries, clothing, and home goods. Store associates offer product recommendations and knowledge, while customers can examine items first-hand and try on clothing for better fit and comfort. However, there are drawbacks such as potential time consumption, waiting in line, and the need to travel to the store.

Online Shopping for Home Delivery

Online shopping for home delivery has gained popularity due to its convenience and broad product availability. Customers can shop from home, saving time and effort, and access a wide range of products from various brands. Fast shipping options and order tracking provide a seamless experience. However, potential shipping delays, the inability to physically inspect products before purchase, and additional shipping fees are drawbacks.

Online Shopping for Pickup

Online shopping for pickup allows customers to conveniently order products online and collect them at a designated location. It offers advantages such as time-saving and the ability to avoid shipping fees. Customers can select a pickup time and location that suits them best, eliminating the need for home delivery or traveling to physical stores. This is especially beneficial for busy individuals or those with limited mobility.

Subscription Services

Subscription services are popular for their convenience and cost savings. Customers pay recurring fees to receive products or services regularly, delivered to their doorstep. It eliminates the need to reorder each time and offers customization options. Subscribers can save money through lower prices and discounts. They can also discover new products or services through curated boxes or personalised recommendations. However, subscription fatigue and overspending can occur when customers are subscribed to multiple services. Repetitive offerings may lead to boredom, and overconsumption can result in waste. Hidden costs or fees are another concern.

Appendix 2 : Issues with Matrix Factorization

Neural Collaborative Filtering (NCF) is a method used to address the limitations of traditional matrix factorization in collaborative filtering. Given a user-item interaction matrix $Y \in \mathbb{R}^{M \times N}$, from users' implicit feedback and M and N represent the number of users and items respectively, the goal is to recommend items to users based on their implicit feedback.

$$y_{ui} = \begin{cases} 1, & \textit{if interaction is observed between user } u \textit{ and item } i; \\ 0, & \textit{otherwise.} \end{cases}$$

Figure 157: Actual Ratings Representation

The interaction between user u and item i is represented as y_{ui}, where 1 indicates an observed interaction and 0 represents no interaction. It is important to note that the value of 1 does not necessary imply that user u likes item i. Likewise, a value of 0 does not insinuate that user u does not like item i. Matrix factorization is the traditional approach to solve a recommender system problem by decomposition the user-item matrix into submatrices.

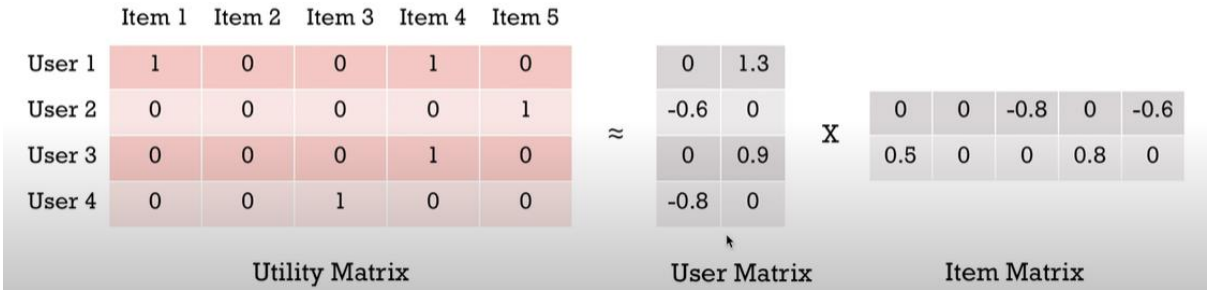


Figure 158: Decomposing user-item interaction matrix

Figure 159: Matrix Reconstruction

To generate predictions, the submatrices are multiplied to reconstruct the user-item interaction matrix. This can be mathematically expressed as:

$$\hat{y}_{ui} = f(u, i|\theta)$$

Where:

- $\hat{y}_{ui}$ is the predicted score for interaction between user u and item i.
- θ denotes model parameters.
- $f()$ denotes the function that maps the model parameters to the predicted score.

Figure 160: Predictions Generation Function

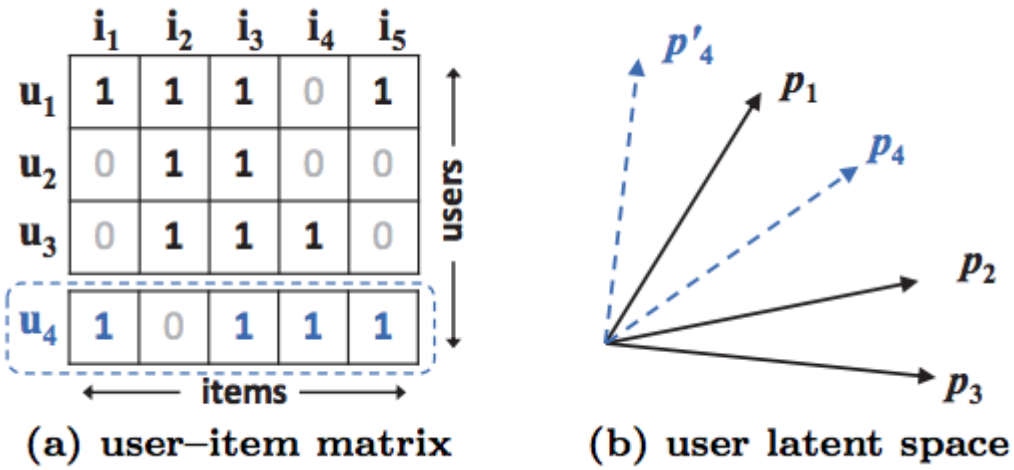
In NCF, the model parameters Q are learned by optimizing popular loss function such as pointwise loss and pairwise loss. Pointwise loss minimizes the square loss between the predicted score and target score, while pairwise loss aims to rank observed entries higher than unobserved entries. The user-item interactions can be modelled through a scalar product of the submatrices expressed as:

$$\hat{y}_{ui} = f(u, i|p_u, q_i) = p_u^T q_i = \sum_{k=1}^{K} p_{uk} q_{ik}$$

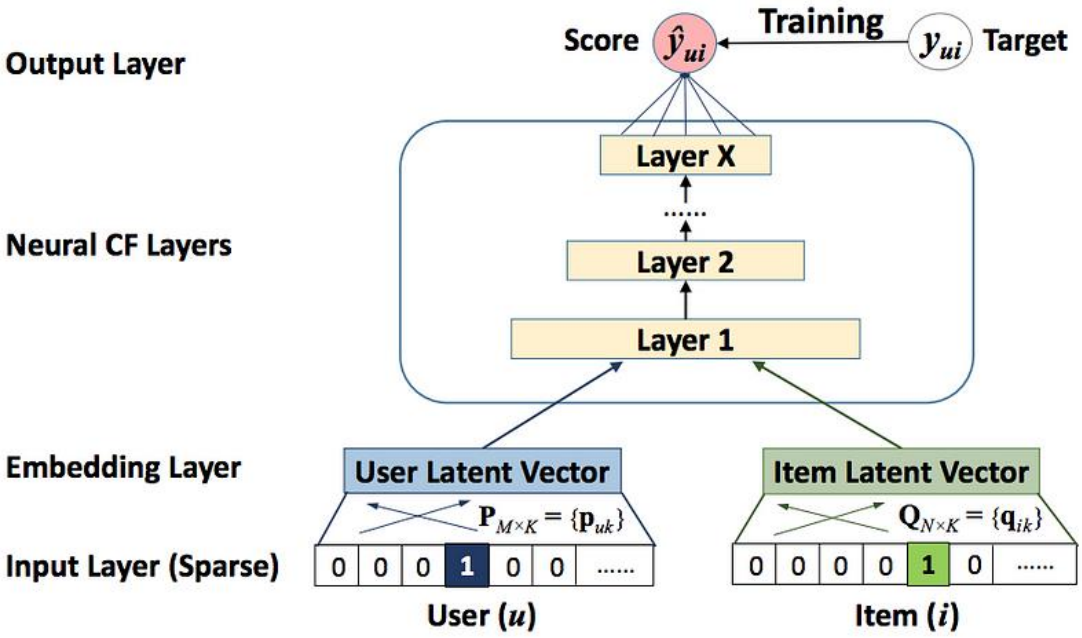
Where:

- $\hat{y}_{ui}$ is the predicted score for interaction between user u and item i.
- p_u is the latent vector for user u.
- q_i is the latent vector for item i.
- K is the dimension of latent space.

Figure 161: Predictions Generation



The authors demonstrate a drawback of Matrix Factorization in collaborative filtering. Starting with user 3 (u_3), when computing the similarity, user 3 is more similar to user 2 than user 1. This is correctly depict in the user latent space. Now, considering user 4 (u_4), who is more similar to user 1, followed by user 3 and then user 2. However, no matter how user 4 is place around user 1 in the user latent space, user 3 will be represented as the least similar to user 4 which is not the case. So the author proposed the neural collaborative filtering to address the issue of large ranking loss in collaborative filtering (He et al., 2017).



Appendix 3: Evaluation Metrics

Evaluation metrics

An evaluation metric quantifies the performance of a predictive model. While Silhouette Coefficient, Calinski-Harabasz, and Davies-Bouldin Index are used to assess the effectiveness of clustering techniques, evaluation metrics such RMSE, MAE, Accuracy, Precision, Specificity, Sensitivity and F1 Score are used for the recommendation system.

Silhouette Coefficient

It provides a value between -1 and 1. A score of '1' indicates well-separated and distinct clusters, while a score of '0' suggests indifferent clusters where the distance between them is not significant. A score of '-1' indicates that clusters have been assigned incorrectly. The silhouette coefficient helps in quantifying the quality of clustering results.

$$Silhouette\ Score = \frac{b - a}{\max(a, b)}$$

where ‘a’ is the average intra-cluster distance and ‘b’ is the average inter-cluster distance.

Calinksi-Harabasz Index

Also known as the Variance Ratio Criterion, the Calinski-Harabasz index is the ration of the sum of between-clusters dispersion and of inter-cluster dispersion for all clusters. A higher score indicates better performance.

The formula for inter-cluster dispersion is given as:

$$BGSS = \sum_{k=1}^{K} n_k \times \|C_k - C\|^2$$

Where:

- n_k: the number of observations in cluster k
- C_k: the centroid of cluster k
- C: the centroid of the dataset
- K: the number of clusters

The computation of the intra-cluster dispersion is given as:

$$WGSS_k = \sum_{i=1}^{n_k} \|X_{ik} - C_k\|^2$$

$$WGSS = \sum_{k=1}^{K} WGSS_k$$

Where:

- n_k: the number of observations in cluster k
- X_{ik}: the i-th observation of cluster k
- C_k: the centroid of cluster k
- $WGSS_k$: the within group sum of squares of cluster k
- K: the number of clusters

The Calinksi-Harabasz Index is calculated as:

$$CH = \frac{\frac{BGSS}{K-1}}{\frac{WGSS}{N-K}} = \frac{BGSS}{WGSS} \times \frac{N-K}{K-1}$$

Where:

- $BGSS$: between-group sum of squares
- $WGSS$: within-group sum of square
- N: total number of observations
- K: total number of clusters

Davies-Bouldin Index

The Davies-Bouldin Index measures the similarity between clusters and is based on the ratio of within-cluster scatter to between-cluster separation. The lower the Davies-Bouldin Index, the better the clustering result.

It is calculated as the average similarity of each cluster, C_i, to its most similar cluster, C_j. The similarity, R_{ij}, for cluster C_i to C_j is given as:

$$R_{ij} = \frac{s_i + s_j}{d_{ij}}$$

Where:

- s_i, s_j is the average distance between each point of cluster C_i, C_j respectively
- d_{ij} is the distance between cluster centroids i and j.

Once the similarity is obtained, the Davies-Bouldin Index is defined as:

$$DB = \frac{1}{k}\sum_{i=1}^{k} \max_{i \neq j} R_{ij}$$

Root Mean Square Error (RMSE)

The Root Mean Square Error (RMSE) is a metric used to evaluate the performance of a statistical model by measuring the average difference between the model's predicted values and the actual values. It provides a standard way to assess the error in predicting quantitative data. The RMSE quantifies the dispersion of residuals, indicating how closely the observed data clusters around the predicted values. It is a useful measure for understanding the accuracy of a model's predictions and comparing different models.

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

Where:

- y_i is the actual values
- $\hat{y}_i$ is the predicted values
- n is the number of observations

A lower RMSE value indicates better model performance, as it signifies smaller average differences between the predicted values and the corresponding actual values. A smaller RMSE implies a higher level of accuracy in the model's predictions, as the deviations between the predicted and actual values are minimized.

Mean Absolute Error (MAE)

The Mean Absolute Error (MAE) is an evaluation metric used to quantify the average magnitude of errors in a set of predictions, irrespective of their direction. It measures the absolute difference between the predicted values and the corresponding true values. MAE provides a measure of the average size of mistakes made by the model in estimating the target variable.

$$MAE = \frac{|(y_i - \hat{y}_i)|}{n}$$

Where:

- y_i is the actual values
- $\hat{y}_i$ is the predicted values
- n is the number of observations

A lower MAE value indicates better model performance, as it implies smaller average errors and a closer alignment between the predicted values and the true values.

True Positive

True Positive is when both the predicted and actual values are positive/true which in our case is 1.

True Negative

True Negative is when both the predicted and actual values are negative/false which in our case is 0.

False Positive

False Positive is when the predicted value is positive/true but the actual value is negative/false.

False Negative

False Negative is when the predicted value is negative/false but the actual value is positive/true.

Accuracy

Accuracy measures the proportion of both positive and negative predictions. It is considered as the most intuitive one. For example, if a model classifies 90 out of 100 predictions, the accuracy would be 90%. Accuracy answers the question, “How often does the model make correct predictions?”.

$$Accuracy = \frac{True\ Positive\ (TP) + True\ Negative\ (TN)}{True\ Positive\ (TP) + True\ Negative\ (TN) + False\ Positive\ (FP) + False\ Negative\ (FN)}$$

Precision

Precision measures the proportion of positive predictions that are actually correct. For example, if a model identifies 70 out of 100 positive predictions, the precision would be 70%. Precision answers the question, "Out of all the positive predictions, how often is it correct?".

$$Precision = \frac{True\ Positive\ (TP)}{True\ Positive\ (TP) + False\ Positive\ (FP)}$$

Specificity

Specificity measures the proportion of actual negative predictions that are correctly identified. For example, a model that classifies 80 out of 100 negative predictions would have a specificity of 80%. Specificity answers the question, "How often does the model identifies negative cases correctly?".

$$Specificity = \frac{True\ Negative\ (TN)}{True\ Negative\ (TN) + False\ Positive\ (FP)}$$

Sensitivity

Sensitivity, also known as Recall, measure the proportion of actual positive cases which are identified correctly. For example, a model that identifies 40 out of 100 positive cases would have a sensitivity of 75%. It answers the question, "How often does the model predict positive cases correctly?".

$$Sensitivity = \frac{True\ Positive\ (TP)}{True\ Positive\ (TP) + False\ Negative\ (FN)}$$

F1 Score

The F1 score is a harmonic mean of precision and sensitivity. It answers the question, "How well does a model balances precision and sensitivity?".

$$F1\ score = 2 \times \frac{Precision \times Sensitivity}{Precision + Sensitivity}$$

Appendix 4: RFM Analysis

RFM analysis quantitatively ranks and groups customers based on the recency, frequency, and monetary value of their recent transactions. It assigns numerical scores to customers, providing an objective analysis for marketing purposes.

RFM analysis ranks each customer based on the following factors:

- Recency: How recent was the customer's last purchase ? When customers make a recent purchase, they tend to have the product fresh in their minds and are more inclined to make future purchases or continue using the product.
- Frequency: How often did this customer make a purchase in a given period ? Customers who have made a previous purchase are often more likely to make additional purchases in the future.
- Monetary: How much money did the customer spend in a given period ? Customers who have a high monetary expenditure are more likely to continue spending money in the future.

RFM analysis assigns scores to customers based on three main factors: recency, frequency, and monetary value. By using clustering techniques, customers with similar scores can be identified and grouped together. This allows the recommendation system to effectively target specific customer segments with more personalised recommendations, improving the overall customer experience. The Authors goal was to predict customer behaviour from previous transactions to provide relevant offers whilst using data mining techniques. They proposed forming clusters using k-means with RFM(Mean) and RFM(Mode) which proved to be an effective method to segregate customers based on their purchase pattern (Abirami and Pattabiraman, 2016).

Appendix 5: MultiLabel Binarizer

MultiLabel Binarizer is a utility class in machine learning. It is used to transform multi-label data into a binary matrix representation. It is commonly used when dealing with classification tasks where each sample can be associated with multiple labels or classes.

To understand how MultiLabel Binarizer works consider the following example:

The following table represent a dataset of movies, where each movie can be assigned multiple genres:

Movie	Genres
Movie 1	Action, Adventure
Movie 2	Comedy, Romance
Movie 3	Drama
Movie 4	Action, Drama, Thriller
Movie 5	Comedy, Drama

Figure 162: Movie Dataset Example

Since each movies are associated with one or more genres, the goal is to transform the genres into a binary matrix. After implementing MultiLabel Binarizer, the following output dataset will be obtained.

The table below shows this final output:

Movie	Action	Adventure	Comedy	Romance	Drama	Thriller
Movie 1	1	1	0	0	0	0
Movie 2	0	0	1	1	0	0
Movie 3	0	0	0	0	1	0
Movie 4	1	0	0	0	1	1
Movie 5	0	0	1	0	1	0

Figure 163: Multi-Label Binarizer Output

Appendix 6: Questionnaire

SUPERMARKET RECOMMENDATION QUESTIONNAIRE

Questionnaire ID: ……

1 Please kindly provide your age range (Years) :

- ☐ Under 20
- ☐ 20 - 30
- ☐ 30 - 40
- ☐ 40 - 50
- ☐ 50 - 60
- ☐ 60 - 70
- ☐ 70 - 80
- ☐ Over 80

2 What is your current marital status ?

- ☐ Single
- ☐ Married
- ☐ Divorced
- ☐ Widowed

3 May I kindly ask about your current occupation or field of work ?

4 Could you please indicate your wage range: (monthly)

- ☐ Less than Rs 25 000
- ☐ Rs 25 000 – Rs 50 000
- ☐ Rs 50 000 – Rs 75 000
- ☐ Rs 75 000 – Rs 100 000
- ☐ Rs 100 000 – Rs 125 000

☐ More than Rs 125 000

5 What type of family do you belong to ?

☐ Nuclear Family (parents and children)

☐ Extended family (includes relatives beyond immediate family)

☐ Single-parent family

6 How many children are there in your family ?
(If applicable) ……

7 Do you consider yourself typically-able to disabled?

☐ Typically-able

☐ Disabled

8 If yes for disabled, what type of disability do you have ? Please select all that apply:

☐ Physical Disability

☐ Visual Impairment (Blindness)

☐ Hearing Impairment (Deafness)

☐ Intellectual or Development Disability

☐ Learning Disability

☐ Mental Health Condition

☐ Chronic Illness or Medical Condition

☐ Other (please specify:
……)

9 What is your religious affiliation, if any ?

☐ Christianity

☐ Islam

☐ Hinduism

☐ Buddhism

☐ Sikhism

☐ Atheism

☐ Other (please specify: ……)

☐ Prefer not to say

10 In which district do you currently reside ? …….

11 Please specify the name of your locality ? …….

12 How would you describe your level of physical activity or exercise habits ?

☐ Sedentary (little to no physical activity)

☐ Light activity (1-2 days per week)

☐ Moderate activity (3-4 days per week)

☐ Active (5 or more days per week)

13 Please describe your dietary preferences or any specific eating habits you follow:

☐ Vegetarian

☐ Vegan

☐ Pescatarian

☐ Flexitarian

☐ Omnivore

14 On average, what percentage of your monthly income do you allocate for grocery shopping?

☐ Less than 10%

☐ 10% - 20%

☐ 20% - 30%

☐ 30% - 40%

☐ More than 40%

15 Do you have a preferred supermarket for your grocery ? ☐ Yes ☐ No

16 May I kindly ask what is the name of this supermarket ? ……..

17 If Yes, why do you prefer this supermarket ? Please share your reason(s)

……
……

18 How many times do you typically purchase groceries in a month ?

- ☐ Once a week
- ☐ Once a month
- ☐ Twice a month
- ☐ Every 2 months
- ☐ Every 3+ months

19 Approximately how much do you spend on these grocery trips ?

- ☐ Less than Rs 1 000
- ☐ Rs 1 000 – Rs 2 500
- ☐ Rs 2 500 – Rs 5 000
- ☐ Rs 5 000 – Rs 7 500
- ☐ Rs 7 500 – Rs 10 000
- ☐ More than Rs 10 000

20 Which type of grocery shopping do you prefer ?

- ☐ In-Store Shopping
- ☐ Online Shopping for Home Delivery
- ☐ Online Shopping for Pick-Up
- ☐ Subscription Services

21 Are you willing to provide a copy/copies of your supermarket receipt to support out research ?

- ☐ Yes
- ☐ No

22 If No, could you kindly explain the reason why you prefer not to provide a copy of your supermarket receipt?

……
……

2
3 If Yes, please staple the copies of your receipt, or alternatively, email a scanned copy of your receipt to this address ronnish.rajanah1@umail.uom.ac.mu with the questionnaire ID written on the receipt (Questionnaire ID: ……)