THE UNIVERSITY of ADELAIDE

Association Rule/Pattern Mining for Recommender System

Group:

Paridhi Awadheshpratap Singh	a1865487
Chahat Segan	a1855353
Ujjwal Bhardwaj	a1881450

The University of Adelaide
4333_COMP_SCI_7306 Mining Big Data
Lecturer: Dr. Alfred Krzywicki

Contents

1. Executive Summary

Our project endeavours to harness the power of advanced data mining techniques to enhance a grocery store's operational efficiency and profitability. By meticulously analyzing customers' transactional data, we aim to unearth patterns in purchasing behaviour, identifying commonly bought items to facilitate strategic product placement and recommendation generation. This initiative significantly benefits the company by improving the overall shopping experience for customers, thereby facilitating greater satisfaction and loyalty.

Moreover, by strategically placing frequently purchased items together and offering tailored recommendations based on individual customer histories, we aim to stimulate sales growth through targeted up-selling and cross-selling opportunities. Crucially, our solution is designed with scalability at its core, ensuring that it can seamlessly accommodate the store's expanding dataset and evolving customer base without compromising performance or reliability.

In our rigorous testing phase, we meticulously evaluated the performance of our recommendation system on both training and test datasets, garnering promising results that highlight the efficacy of our approach in delivering personalized and relevant recommendations to customers. Building upon these insights, we are confident in recommending the adoption of our recommendation system to the grocery store management, as it holds the potential to optimize product placement, enhance customer engagement, and ultimately drive sales.

Furthermore, we advise ongoing monitoring and refinement of the system to adapt to shifting customer preferences and market dynamics, ensuring its continued effectiveness and relevance in the long term. Looking ahead, ample opportunity exists for further enhancement, including exploring and implementing advanced techniques such as natural language processing (NLP) for deeper customer insights and sentiment analysis.

In conclusion, our project represents a strategic investment for the grocery store, offering actionable insights and recommendations that have the potential to generate substantial business benefits and facilitate sustainable growth.

2. Introduction

In today's competitive retail environment, understanding customer behavior is crucial for optimizing operations and profitability. Retailers struggle to decipher complex purchasing patterns for product placement and personalized recommendations Mofokeng (2021). This challenge necessitates advanced data analysis techniques to extract insights from vast transactional data Agrawal et al. (1993). This study proposes an innovative hybrid recommender system for grocery stores, combining pattern mining with collaborative filtering to provide accurate and diverse recommendations.

Our two-fold aim is to:

- Develop a hybrid model that leverages FP-growth for pattern mining and collaborative filtering for recommendation generation.
- Address the cold start problem inherent in traditional collaborative filtering, especially for new users.

The proposed model employs a multi-step process. First, FP-growth mines frequent itemsets, capturing associations between co-purchased items. Subsequently, collaborative filtering analyzes user-item interactions to generate personalized recommendations based on user similarities and item associations. For the project we have implemented item-based collaborative filtering through Data to Recommendation (D2R) and Pattern to Recommendation (P2R). D2R generates recommendations directly from user-item interaction data whereas P2R utilizes frequent pattern mining techniques, specifically FP-Growth, to extract meaningful patterns from transactional data which capture associations between items frequently purchased together by users.

The cold start problem is a challenge recommender systems face when they need to provide recommendations for new users or items with little to no interaction data or user history. Recognizing the cold start problem, we implemented strategies for effective recommendations even with limited user data. New users benefit from pattern-mined recommendations, while fallback mechanisms like popular item or content-based filtering provide initial suggestions. This approach allows for progressive profiling, continuously updating user preferences and refining recommendations over time, mitigating the cold start issue.

The system's evaluation will compare its performance to traditional collaborative filtering techniques using metrics like Average Precision (AP), Root Mean Square Error (RMSE), Average Reciprocal Hit Rank (ARHR), coverage, and diversity.

By combining pattern mining with collaborative filtering and addressing the cold start problem, this hybrid recommender system aims to enhance recommendation quality and user experience, leading to increased user engagement and business benefits.

3. Exploratory Analysis

3.1. Data Structure

We make use of the training data present in the files *basket_data_by_date_train.csv* (for training the data) and *basket_data_by_date_test.csv* (for testing the data). Our data comprises 40,000 entries of items sold in a grocery store, with the following attributes:

- **BillNo / Bill Number:** This column represents the unique identifier assigned to each bill or transaction.
- **CustomerID / Customer ID:** This column stores the unique identifier for each customer.
- **Itemname / Item Name:** This column contains the name or description of the item purchased.
- **Quantity / Quantity Purchased:** This column indicates the quantity of the item purchased by the customer in this transaction.
- **Date / Date of Transaction:** This column records the date when the transaction occurred.
- **Price / Price of Single Item:** This column represents the price of a single unit of the item.
- **Cost / Total Cost of Transaction:** This column calculates the total cost of the transaction by multiplying the quantity purchased by the price per item.

After analyzing the data, it was determined that the **BillNo** data attribute holds no value in providing valuable insights. To better understand customer behaviour and item association within a single transaction, grouping the data by **CustomerID** is recommended. This will facilitate the identification of customer preferences and item affinities, resulting in a more accurate representation of the transactional pattern.

3.2. Gathering Data Insights

The graphical representation in Figure 1 depicts the top 20 frequently purchased items that are most likely to be present in every customer's shopping basket. This data provides valuable insights into consumer behaviour and can be used to predict shopping basket items and sales patterns.

"Regency Cakestand 3 Tier" is the best-selling product in our data set.

To gain a deeper understanding of the data, it would be beneficial to generate a correlation matrix, which can provide valuable insights into the relationships between various variables. By plotting such a matrix, we can visualize and analyze the correlations, further informing our decision-making process and enabling us to draw more accurate conclusions from the data.

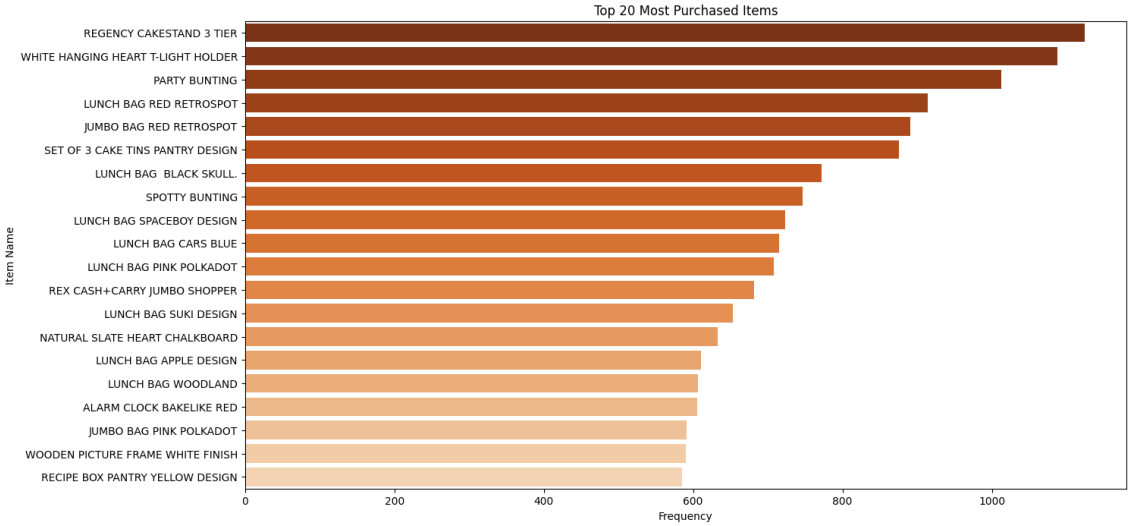


Figure 1: Top 20 most purchased products

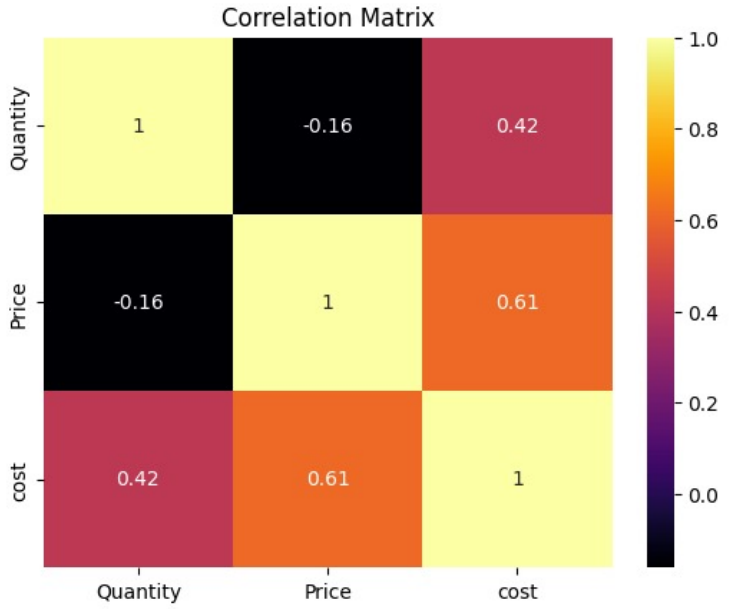


Figure 2: Correlation Matrix of the variables under study

- Quantity and Price have a negative correlation of -0.16, indicating a weak inverse relationship between these two variables. As the quantity of an item increases, the price tends to decrease slightly.
- Quantity and Cost have a moderate positive correlation of 0.42, suggesting that as the quantity of an item increases, the overall cost also tends to increase.
- Price and Cost have a strong positive correlation of 0.61, meaning that as the price of an item increases, the overall cost also tends to increase significantly.

3.3. Finding Patterns

When examining sales data over time, it's crucial to detect any seasonal patterns or trends that may emerge. By aggregating the data by specific time frames such as date, week, month, or quarter, one can uncover seasonal fluctuations in sales, customer behaviour, or product demand. Visualizing these temporal patterns through methods like line plots or heatmaps can significantly enhance the understanding of the seasonality present in the data.

When analyzing transaction costs, plotting a graph of the total cost against time is beneficial to identify patterns or seasonality in the data. By doing so, we can gain insight into the trends of these transactions. As illustrated in reference **??**, this visual representation can reveal valuable information that may not be apparent from raw data alone.

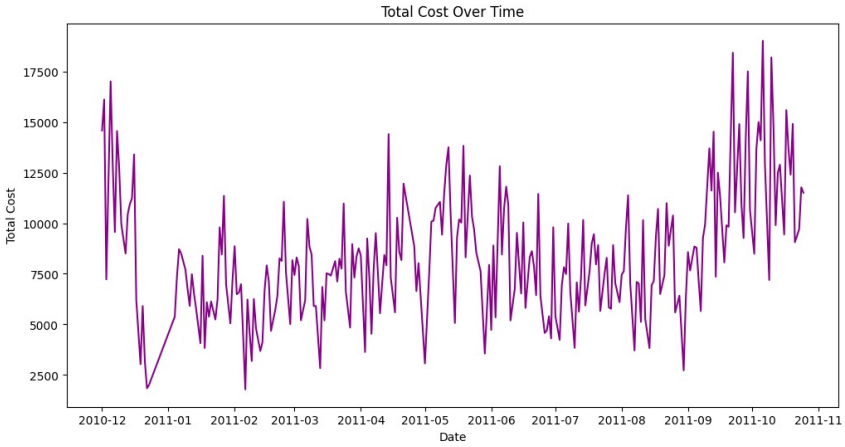


Figure 3: Examining Sales over Time to Understand Seasonal Trends

The graph depicts total cost over time and reveals notable fluctuations with frequent spikes and dips. Seasonal patterns emerge, indicating regular peaks and seasonalities at specific times of the year **especially during the last months of the year**, implying seasonal influences on total cost. Despite these fluctuations, an overarching increasing trend persists throughout the observed period. Extreme values punctuate the data, with both high peaks and low troughs, prompting further inquiry into their causes. Factors such as promotions, holidays, supplier changes, or external events likely drive the observed patterns and trends in total cost for the grocery store.

4. Project Structure and Implementation

The project structure encompasses several key components designed to facilitate the development and evaluation of our hybrid recommender system. It begins with data preprocessing, where transactional data is transformed into a suitable format and creates a customer-based matrix to organize user-item interactions. Subsequently, the FP-Growth algorithm is employed for frequent pattern mining, extracting significant associations between items commonly purchased together. These pat-

terns serve as the foundation for recommendation generation. Collaborative filtering techniques(D2R and P2R) are then applied to analyze user-item interactions and generate personalized recommendations based on similarities between users and item associations. Evaluation metrics such as Average Precision, Root Mean Square Error, and coverage are utilized to assess the performance of the recommender system. Finally, strategies to address the cold start problem, such as fallback mechanisms and progressive profiling, are implemented to ensure effective recommendation generation for users with limited transaction history. This structured approach enables us to develop and evaluate an innovative hybrid recommender system tailored for grocery store environments.

4.1. Data Preprocessing

The exploratory data analysis (EDA) conducted revealed the cleanliness of the dataset, requiring minimal preprocessing prior to analysis. It was transformed into a binary matrix representation to facilitate effective analysis of the transactional data. This matrix schema organizes data according to customer IDs rather than transaction IDs 4, aligning with a user-centric approach to capture user-item interactions. The decision to adopt this customer-based matrix approach stems from several considerations intrinsic to recommender system development:

	KNITTED UNION FLAG HOT WATER BOTTLE	GLASS STAR FROSTED T-LIGHT HOLDER	WHITE HANGING HEART T-LIGHT HOLDER	RED WOOLLY HOTTIE WHITE HEART.	SET 7 BABUSHKA NESTING BOXES	CREAM CUPID HEARTS COAT HANGER	WHITE METAL LANTERN	HAND WARMER UNION JACK	HAND WARMER RED POLKA DOT	WOOD 2 DRAWER CABINET WHITE FINISH	...	CRACKED GLAZE EARRINGS BROWN	WHITE VINT ART DECO CRYSTAL NECKLAC	WHITE VINTAGE CRYSTAL EARRINGS	BLACK VINT ART DEC CRYSTAL NECKLACE	NEW BAROQUE SMALL NECKLACE BLACK	GLITTER HEART DECORATION	GRASS HOPPER WOODEN WALL CLOCK	ASSORTED TUTTI FRUTTI KEYRING BALL	STANDING FAIRY POLE SUPPORT	FUSCHIA TABLE RUN FLOWER
17850	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
13047	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
14688	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
15311	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
16098	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
13489	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
12714	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
13904	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
15755	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
13932	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

1391 rows × 2523 columns

Figure 4: Sample of customer matrix representation

- **User-Centric Approach:** Organizing data according to customer IDs focuses on capturing individual user-item interactions, essential for collaborative filtering techniques reliant on user similarities used in later steps.
- **Personalization:** By organizing data in a customer-based matrix, we can personalize recommendations for individual customers based on their past interactions with items. This allows us to tailor recommendations to each customer's unique preferences, leading to a more personalized and relevant shopping experience.
- **Sparse Data Handling:** In transaction-based matrices, transactions are represented as rows, leading to a potentially large and sparse matrix, especially in scenarios with a vast number of transactions and items. Customer-based matrices tend to be less sparse since each row represents a unique customer, making them more efficient for collaborative filtering computations.

- **Cold Start Problem Mitigation:** Customer-based matrices provide a solution to the cold start problem, which arises when dealing with new users with limited or no transaction history. By organizing data based on customers, we can leverage collaborative filtering techniques to make recommendations even for new users by identifying similarities with existing users.
- **Improved Recommendation Quality:** Organizing data at the customer level allows for better capture of long-term user preferences and behaviour, leading to more accurate and relevant recommendations. By analyzing historical interactions between customers and items, we can identify patterns and similarities that enable us to generate high-quality recommendations.

4.2. Pattern Mining: FP-Growth

In our code, the FP-Growth algorithm is utilized for frequent pattern mining from transactional datasets. Specifically, FP-Growth stands for Frequent Pattern Growth, and it's an efficient algorithm for discovering frequent itemsets within transactional databases. This algorithm is particularly beneficial for our project in identifying associations between items commonly purchased together by customers. By extracting these frequent itemsets, the algorithm enables the identification of significant patterns in customer purchasing behavior, which forms the foundation for generating accurate and personalized recommendations.

Introduced as an enhancement to the Apriori algorithm, FP-growth presents a novel approach to frequent pattern mining. Unlike Apriori, FP-growth streamlines the process by scanning the database only twice, significantly reducing computational overhead. The core innovation lies in its utilization of a tree structure, known as the FP-Tree, which efficiently stores frequent item information, compressing vast datasets and mitigating the need for generating an excessive number of candidate itemsets. Kolahkaj & Khalilian (2015)

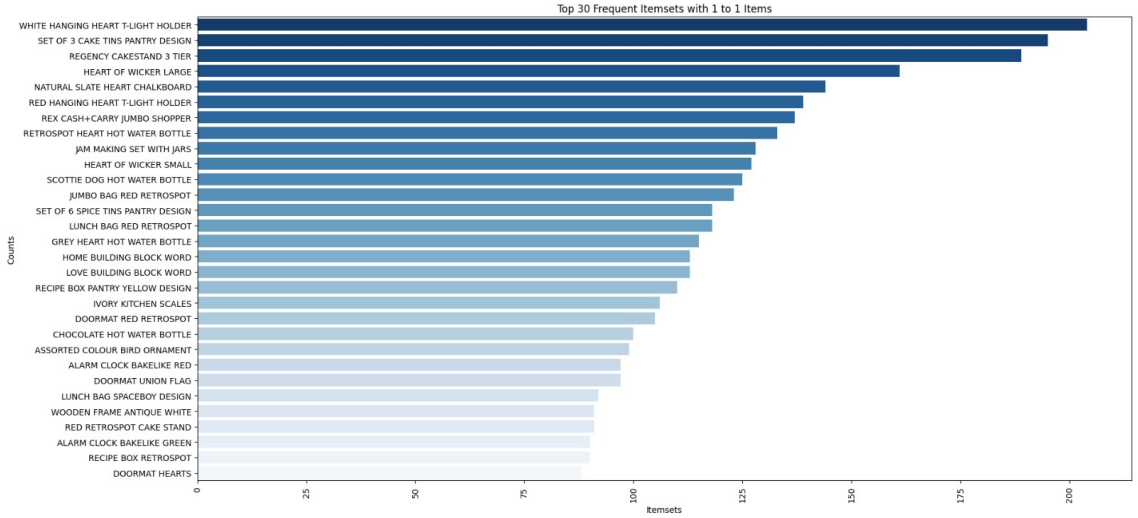
Initially, FP-growth identifies the frequency of each itemset and sorts them in descending order based on their support, thereby prioritizing the most relevant patterns. Subsequently, it constructs the FP-Tree based on the remaining frequent items, facilitating a compact representation of the transactional data and minimizing memory consumption. During the second scan of the database, FP-growth dynamically merges existing paths in the FP-Tree when encountering similar transactions, incrementing their support count and optimizing memory utilization. Kolahkaj & Khalilian (2015)

Overall, the use of FP-Growth in our code facilitates the extraction of frequent itemsets from transactional data, enabling us to uncover meaningful associations between items and enhance the quality of recommendations provided to customers.

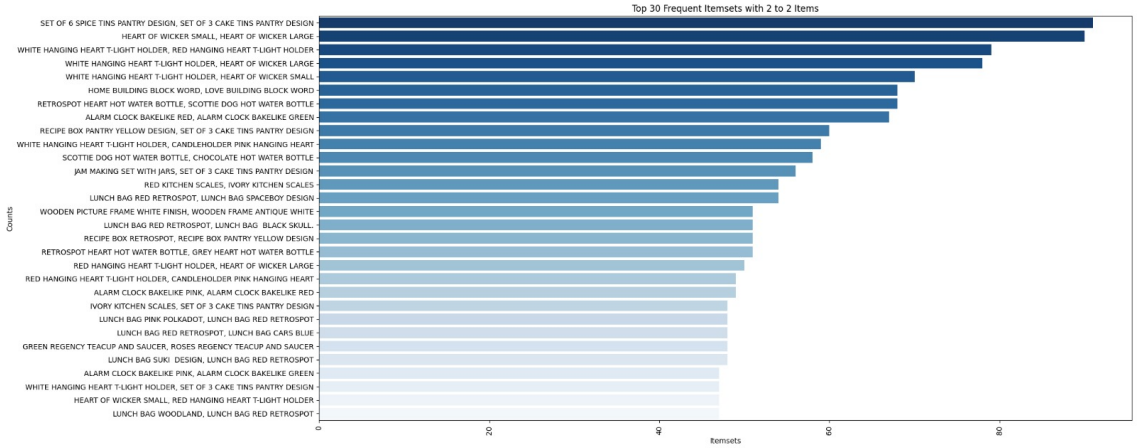
The figure 5 illustrates the 30 most frequent itemsets by number of items in the itemset as calculated by the FP-growth algorithm.

4.3. Data to Recommendation

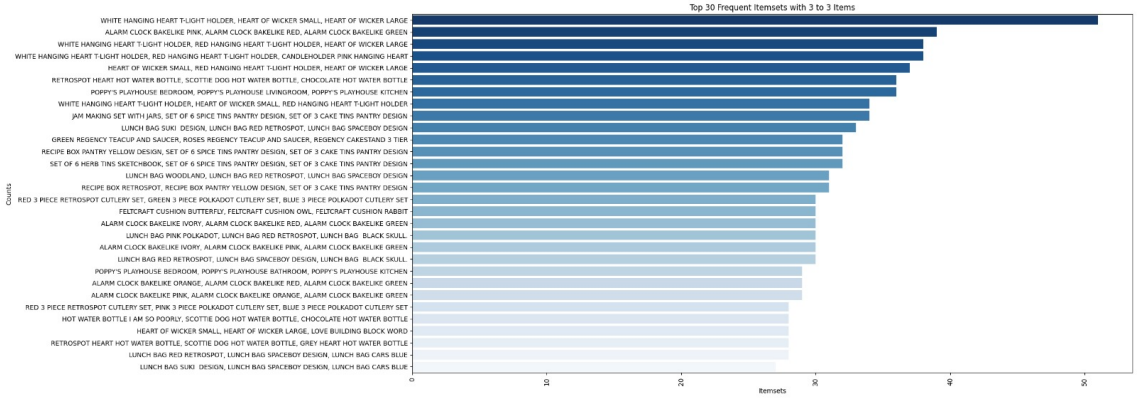
The Data to Recommendations (D2R) process is a collaborative filtering-based recommendation approach that leverages the similarities between items to generate



(a) Most Frequent Itemsets (1 Item Itemsets)



(b) Most Frequent Itemsets (2 Item Itemsets)



(c) Most Frequent Itemsets (3 Item Itemsets)

Figure 5: Subplots of Frequent Itemsets as created by FP-Growth

recommendations. Figure 6 depicts the working of the D2R process, further explanations are as follows:

1. **Create the Customer-Item Matrix**:

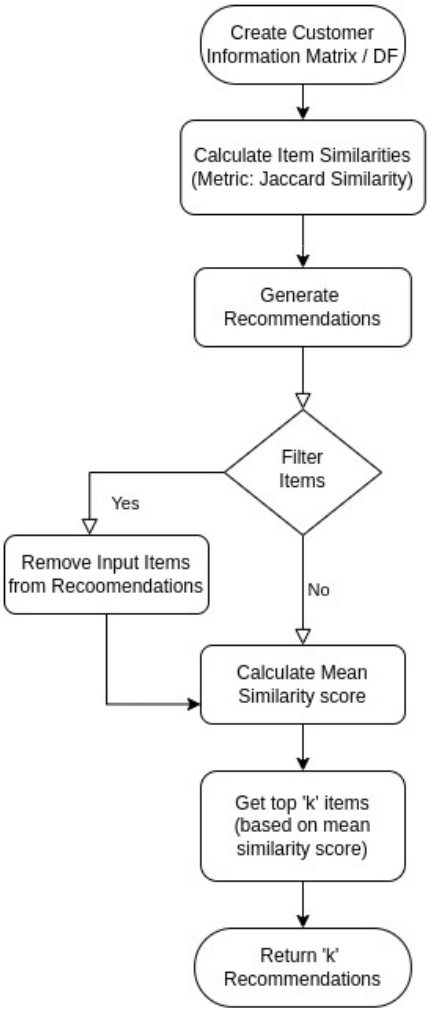


Figure 6: Flowchart for the Data to Recommendations (D2R) Approach

- The first step is to create a customer-item matrix, where the rows represent customers and the columns represent items.
- Each cell in the matrix represents the presence (1) or absence (0) of an item in the customer's purchase history.
- This matrix is used to capture the relationship between customers and the items they have purchased.

2. **Calculate Item Similarities**:

- The next step is to calculate the similarities between the items in the customer-item matrix.
- The similarity between two items is typically measured using the Jaccard similarity, which calculates the ratio of the size of the intersection of the sets of customers who have purchased the two items to the size of the union of the sets of customers who have purchased the two items.
- The Jaccard similarity values range from 0 (no similarity) to 1 (identical).

- The item similarity matrix is a square matrix, where the rows and columns represent the items, and the values in the cells represent the Jaccard similarity between the corresponding items.

3. **Generate Recommendations**:

   - To generate recommendations for a target user, the D2R approach takes a set of items purchased by the user as input.
   - It then calculates the mean similarity score between the input items and all other items in the item similarity matrix.
   - The items with the highest mean similarity scores are then selected as the recommended items.
   - The input items can be optionally filtered out from the recommended items to avoid recommending items the user has already purchased.

The key idea behind the D2R approach is that if a user has purchased a set of items, the items that are most similar to those items are likely to be of interest to the user as well. By leveraging the item similarity matrix, the D2R approach can identify the items that are most relevant to the user's preferences, as reflected in their purchase history.

The advantages of the D2R approach include its simplicity, scalability, and ability to handle cold-start problems (where new items or users are introduced). However, it may not capture complex relationships or dependencies between items, and it may not be as effective in situations where the user's preferences change over time.

4.4. Pattern to Recommendation

The P2R approach leverages the information from the frequent itemsets to provide recommendations that are aligned with the user's preferences, as reflected in their purchase history. By considering the support of the itemsets, the P2R approach can prioritize the recommendations that are more likely to be relevant to the user.

The key steps of the P2R process as illustrated in Figure 7 are:

1. **Filter the Frequent Itemsets**:

   - The first step is to filter the transaction-item matrix to include only the frequent items.
   - The frequent items are collected from the frequent itemsets DataFrame.
   - The filtered transaction-item matrix contains only the columns (items) that are present in the frequent itemsets.

2. **Get Itemsets and their Support**:

   - The function creates a dictionary that maps the frequent itemsets to their corresponding support values.

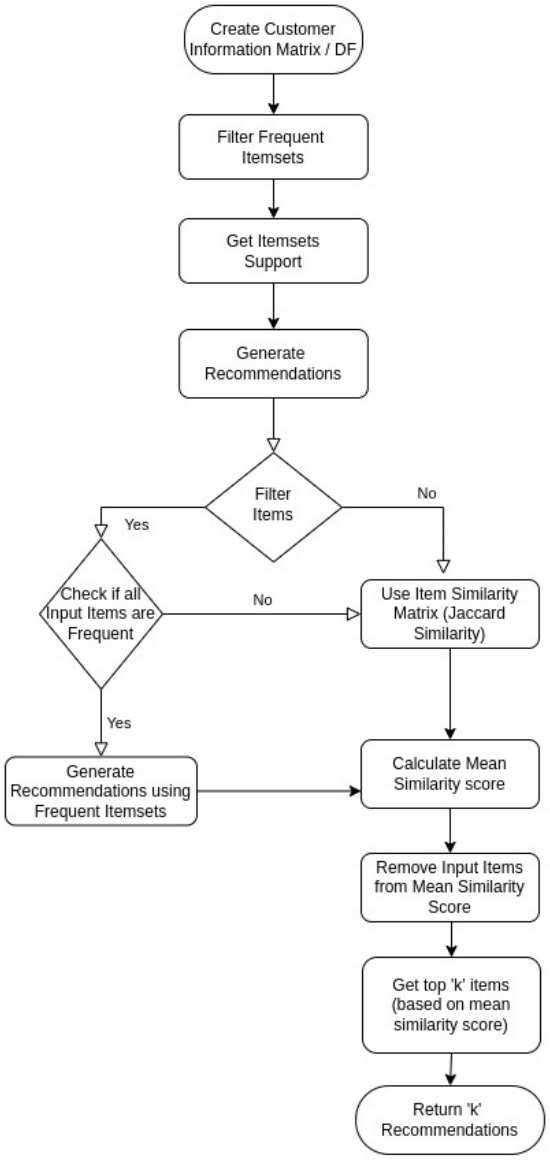


Figure 7: Flowchart for the Pattern to Recommendations (P2R) Approach

- This dictionary is used to keep track of the support for each frequent itemset.

3. **Generate Recommendations**:

- To generate recommendations for a target user, the P2R approach takes a set of items purchased by the user as input.
- It then checks if all the input items are frequent.
- If all items are frequent, the P2R approach generates recommendations using the frequent itemsets.
- If not all items are frequent, the P2R approach uses the item similarity matrix for the non-frequent items.

- The recommendations are filtered to remove the input items, and the remaining items are ranked based on their support in the frequent itemsets.
- The top k recommended items are returned, with the most supported items ranked higher.

4.5. Pattern to Recommendation With Cold Start

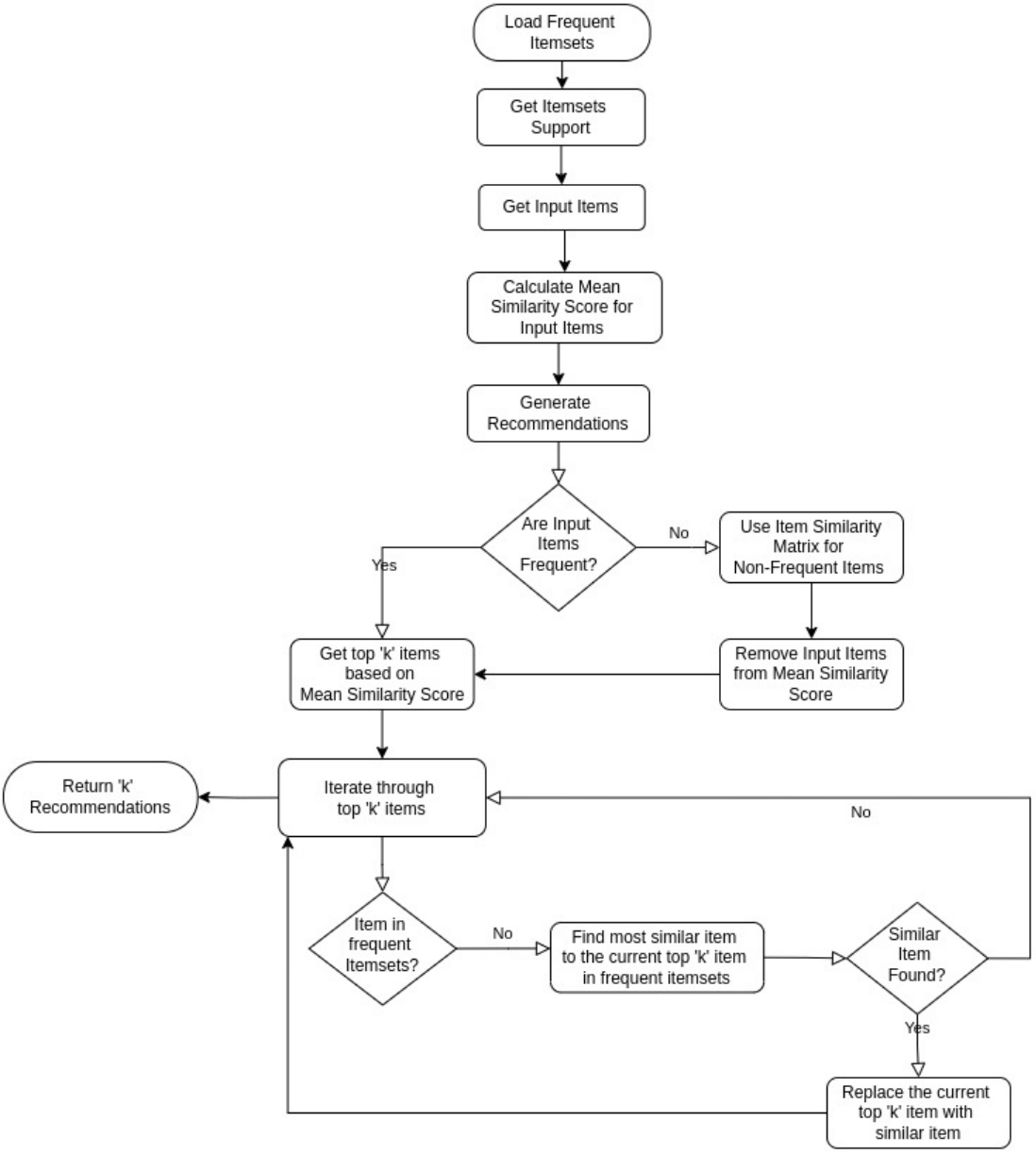


Figure 8: Flowchart for the Pattern to Recommendations With Cold Start (P2R-CS) Approach

The key difference between the P2R-CS and the regular P2R process is the handling of the cold-start problem. In the P2R-CS approach, if any of the top k

recommended items are not present in the frequent itemsets, the algorithm finds the most similar item from the frequent itemsets and replaces the non-frequent item with the similar one. This ensures that the recommendations are still relevant and meaningful, even when the input items are not present in the frequent itemsets.

The key steps of the P2R-CS process as illustrated in Figure 8 are:

1. **Filter the Frequent Itemsets**:
   - The first step is to filter the transaction-item matrix to include only the frequent items.
   - The frequent items are collected from the frequent itemsets DataFrame.
   - The filtered transaction-item matrix contains only the columns (items) that are present in the frequent itemsets.
2. **Get Itemsets and their Support**:
   - The function creates a dictionary that maps the frequent itemsets to their corresponding support values.
   - This dictionary is used to keep track of the support for each frequent itemset.
3. **Generate Recommendations**:
   - To generate recommendations for a target user, the P2R-CS approach takes a set of items purchased by the user as input.
   - It then checks if all the input items are frequent.
   - If all items are frequent, the P2R-CS approach generates recommendations using the frequent itemsets.
   - If not all items are frequent, the P2R-CS approach uses the item similarity matrix for the non-frequent items.
   - The recommendations are filtered to remove the input items, and the remaining items are ranked based on their support in the frequent itemsets.
   - **Cold Start Handling**:
     - If any of the top k recommended items are not in the frequent itemsets, the P2R-CS approach finds the most similar item from the frequent itemsets using the *find_similar_item_cold_start* function.
     - The non-frequent top k item is then replaced with the similar item found from the frequent itemsets.
   - The top k recommended items, with potential replacements for non-frequent items, are returned.

4.6. Evaluation Methodology

Our recommendation system underwent a comprehensive evaluation process utilizing a range of diverse metrics to assess its performance and efficacy. These metrics served as essential tools in measuring the accuracy of the recommendations provided to end users.

- **Root Mean Square Error (RMSE)**: RMSE is a statistical measure that evaluates the average magnitude of the differences between predicted and observed values. It is calculated as the square root of the average of the squared errors and provides a single metric summarizing the overall deviation between predicted and actual values. The formula for RMSE is:

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

- **Average Reciprocal Hit Rank (ARHR)**: ARHR assesses the effectiveness of recommendation systems by measuring how quickly relevant recommendations are presented to users. It is calculated by averaging the reciprocal of the rank of relevant items. A lower ARHR value indicates better system performance. The formula for ARHR is:

$$ARHR = \frac{1}{\text{number of users}}\sum_{i=1}^{n}\frac{1}{\text{rank}_i}$$

- **Precision**: Precision measures the ratio of relevant items among the recommended ones, reflecting the system's ability to deliver precise and valuable suggestions to users. It is calculated as the proportion of recommended items that are relevant. The formula for Precision is:

$$Precision = \frac{1}{m}\sum_{k=1}^{N} P(k) \text{ if item } k \text{ was relevant}$$

- **Coverage**: Coverage evaluates the effectiveness of recommendation systems in suggesting items from the entire item space, reflecting the system's ability to cater to diverse user interests. A higher coverage ensures that users receive recommendations aligned with their preferences.

- **Diversity**: Diversity assesses the variety and novelty of recommended items, ensuring that the system offers users multiple options. Incorporating diversity in recommendation systems enhances user satisfaction and engagement. The formula for Diversity is:

$$Diversity = 1 - \text{average similarity between recommended pairs}$$

5. Discussion of Results

The following models have been evaluated to assess their ability to provide quality recommendations to users.

Model	Avg Precision	Avg RMSE	Avg ARHR	Coverage	Diversity
Data2Recommendations	1.42	0.765182	0.445556	0.0934898	0.98065
Pattern2Recommendations	1.53667	0.720975	0.447222	0.0832702	0.978971
Pattern2RecommendationsWithColdStart	1.40667	0.740975	0.440833	0.0859198	0.977362

Figure 9: Final metrics on our Test dataset.

The Data2Recommendations model has shown promising performance across several key metrics. It has an average precision of 1.42, so it can accurately recommend relevant items to users. Its average RMSE of 0.765182 indicates relatively accurate predictions, ensuring that suggested items align closely with users' preferences. The model achieves an average ARHR of 0.445556, ranking relevant items higher for users and enhancing their overall experience. While the coverage of 0.0934898 suggests a decent proportion of recommended items, its diversity score of 0.98065 indicates a wide variety of suggestions, catering to a broad range of user preferences.

The Pattern2Recommendations model has also shown commendable performance across various evaluation metrics. It has an average precision of 1.53667, which means it excels in recommending relevant items to users with precision. Its average RMSE of 0.720975 indicates accurate predictions, minimizing discrepancies between predicted and actual ratings. The model achieves an average ARHR of 0.447222, indicating proficient ranking of relevant items higher in recommendation lists, enhancing user satisfaction. Although its coverage of 0.0832702 suggests a slightly lower proportion of recommended items, its diversity score of 0.978971 signifies a diverse range of suggestions, catering to varied user preferences.

The Pattern2RecommendationsWithColdStart model has delivered competitive performance across the evaluated metrics. With an average precision of 1.40667, it effectively recommends relevant items to users, albeit slightly lower than the Pattern2Recommendations model. Its average RMSE of 0.740975 indicates relatively accurate predictions, ensuring alignment with users' preferences. The model achieves an average ARHR of 0.440833, suggesting proficient ranking of relevant items higher in recommendation lists, contributing to improved user engagement. While its coverage of 0.0859198 indicates a moderate proportion of recommended items, its diversity score of 0.977362 reflects a broad variety of suggestions, accommodating diverse user tastes and preferences.

6. Conclusion

The central focus of our group project was to augment a recommendation system by utilising frequent patterns. The approach we adopted involved the implementation of algorithms such as FP growth and association rules, which enabled

us to generate improved recommendations based on patterns discovered in the data. We analyzed unique item names in the train and test datasets to ensure consistency and reliability in our recommendation system.

The successful implementation of the FP-Growth algorithm and association rules allowed for extracting meaningful patterns from big data, which was then used to enhance the recommendation process. The findings of this study demonstrate the potential of data mining techniques and finding meaningful patterns in improving the accuracy and effectiveness of recommendation systems—various recommendations tailored to user preferences.

The comparative analysis of the performance of three recommendation models reveals that Pattern2Recommendations has a slightly higher average precision and coverage than the other models. However, Data2Recommendations exhibits competitive performance in terms of diversity and accuracy. In contrast, Pattern2RecommendationsWithColdStart presents a balanced performance across various metrics. The optimal model selection depends on various factors, including the desired balance between precision and coverage, the significance of cold-start scenarios, and the overall suitability for the targeted user base. Therefore, carefully considering these factors is essential to determine the most appropriate recommendation model.

7. Reflection

We encountered several obstacles throughout this project while striving to achieve our objectives. One of the most arduous challenges we faced was finding a solution to the cold start problem with an item. The model trained on the training dataset produced frequent item sets that only contained a few items, as they were composed solely of frequent items. Consequently, an error occurred in the code when our evaluation model failed to locate keys from the test datasets in the frequent itemsets. Initially, we could not identify a viable solution to this issue, as we believed that the testing dataset contained a distinct set of items, which was justifiable since we needed to recommend new items. Overcoming this cold start problem by identifying similar items from the similarity matrix and matching them with similar names posed a significant challenge for us.

Further diving into the project, we can see multiple places to dive deeper into the study. We can delve deeper into the data by using a single dataset and the standard split ratio to divide it uniformly to our current project; we have multiple opportunities to delve deeper into the data. One such opportunity involves utilizing a single dataset and the standard split ratio to partition uniform data into testing and training parts. This approach ensures that our system remains unbiased and the testing data remains unbiased even after the same pre-processing as the training dataset. By using the standard split method, we can guarantee that our data sets are entirely randomized, thereby reducing bias in our analysis.

8. Future Work and Recommendation

We can pursue another line of inquiry by grouping our data based on time and conducting a time-series analysis. By integrating time-series analysis with a recommendation system, we can create an improved recommender system that considers data patterns and transaction timestamps. This enables us to predict when a particular product is recommended to a customer based on their preferences.

Furthermore, a different line of study can be conducted by grouping the data based on time and performing a time-series analysis. Such amalgamation of time series analysis and a recommendation system can give us a better recommender system that considers every transaction's data patterns and time stamps. Using this, we can predict at what time or date which product will likely be recommended to a customer, considering their preferences.

References

Agrawal, R., Imieliński, T. & Swami, A. (1993), 'Mining association rules between sets of items in large databases', *SIGMOD Rec.* **22**(2), 207–216.
URL: *https://doi.org/10.1145/170036.170072* 3

Kolahkaj, M. & Khalilian, M. (2015), A recommender system by using classification based on frequent pattern mining and j48 algorithm, *in* '2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI)', pp. 780–786. 8

Mofokeng, T. (2021), 'The impact of online shopping attributes on customer satisfaction and loyalty: Moderating effects of e-commerce experience', *Cogent Business Management* **8**. 3

A Appendices