Pattern Mining and Recommender System

Group: 14

Hasnain Hossain a1945656

Rizal Hamdan Arigusti a1939989

Sadman Sharif a1944825

The University of Adelaide

4533_COMP_SCI_7306 Mining Big Data

Lecturer: Dr. Alfred Krzywicki

Table of Contents

Github Repository of this work is available on this link:

https://github.com/rizalhamdana/mbd_7306_assignment_3

1. Executive Summary

This report presents the development and evaluation of a grocery recommendation system designed to generate top-K product suggestions based on customers' historical purchase data. The system integrates association rule mining with collaborative filtering techniques to enhance recommendation accuracy and relevance. We realized the limitations of relying purely on traditional metrics like support and confidence, opting on our own mix of lift and temporal relevance, backed by literature. The system was evaluated using a leave-last-basket-per-user protocol, ensuring that the recommendations are tested against unseen user behavior. The integration of lift and recency in ranking association rules led to more relevant recommendations, as evidenced by higher recall metrics.

2. Introduction

In the retail industry, particularly within grocery shopping, providing personalized product recommendations is crucial for enhancing customer experience and driving sales. The primary challenge and objective addressed in this project is the development of a recommendation system that effectively combines association rule mining with collaborative filtering. The system's performance is assessed using metrics such as Recall and Mean Average Precision at K (MAP@K), with a focus on optimizing these metrics through the incorporation of lift and recency in rule ranking.

This report is structured as follows: We will first explore some motivating literature for the project. The Exploratory Analysis section examines the dataset in detail, attempting to identify key patterns. Within Implementation, methodologies employed for frequent pattern mining and collaborative filtering are detailed. The System Integration section covers an end-to-end pipeline, with evaluation results. Finally, we discuss some recommendations within the conclusion, and summarize our findings.

3. Motivation and Literature

The motivation for this work stems from a blend of practical challenges and academic critiques. Early association rule mining frameworks prioritized support and confidence (Agrawal et al., 1993), but subsequent studies exposed their shortcomings. Notably, Aggarwal & Yu (2001) and Hahsler (2015) argue that confidence alone can mislead, especially when support is low or item popularity is high. These critiques guided us toward using lift as a more stable metric for association strength.

Furthermore, we drew inspiration from temporal personalization literature, particularly the work of Yan et al. (2008) and Shani & Gunawardana (2011), who emphasized recency as a key driver of relevance in recommendation contexts. Incorporating this into our scoring was a natural progression, and we found additional justification in Cho et al. (2013), who successfully used time-weighted rule mining in commercial recommendation systems.

4. Exploratory Analysis

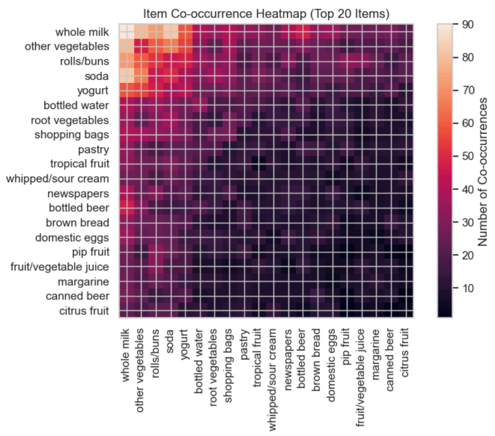

Figure 1. Item Co-occurrence Heatmap (Top 20 Items)

We performed several key analyses to assess co-occurrence patterns, basket size distribution, user behavior, and sparsity issues. The heatmap in **Figure. 1** shows items such as whole milk, other vegetables, and rolls/buns exhibit frequent co-purchases, while the majority of item pairs have low joint support. This suggests that frequent pattern mining methods like Apriori or FP-Growth are appropriate, but threshold tuning will be critical to avoid mining low-value patterns.

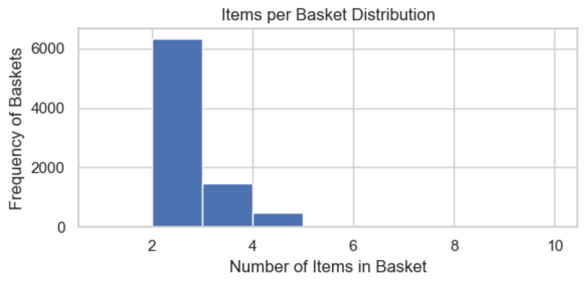

Figure. 2 Items per Basket Distribution

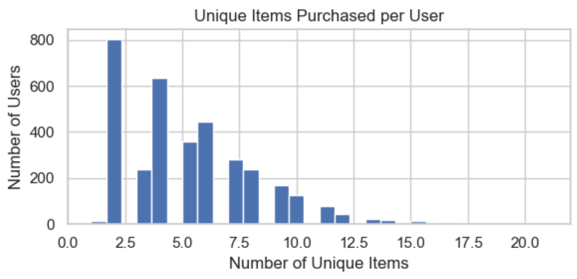

Figure. 3 Unique Items Purchased per User

Figure. 2 shows that the vast majority of baskets contain only two or three items. This sparsity suggests that recall-based evaluation metrics are preferable to precision-based ones, as missing even a single relevant item would disproportionately penalize the recommender. Further emphasizing the challenge of generating diverse recommendations is provided in **Figure. 3**, showing most users purchased a very limited number of unique items over their transaction history. This supports the integration of collaborative filtering alongside association-rule-based recommendations to enhance personalization for users with limited purchase histories. We also sampled a 50 × 50 section of the utility matrix (users × items) to visualize sparsity, presented in **Figure. 4**. The extremely sparse structure reinforces the need for item-based collaborative filtering methods, which are better suited to handling sparse interaction matrices compared to user-based approaches.

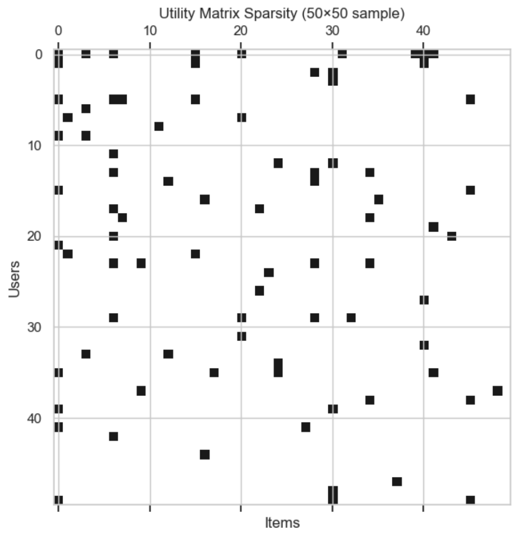

Figure 4. Utility Matrix Sparsity (50x50 Samples)

One of the challenges when developing a recommender system is the long-tail problem. **Figure. 5** reflects this reality: Only a small number of popular items that are frequently purchased by the users. This problem leads to the scenario where less-popular items are rarely recommended even with high relevance, as most transactions contain only 2-3 items

(Guo 2012, p. 361). This, coupled with data sparsity poses a challenge for our task. To calculate the sparsity of the train dataset, a formula from (Huang, Chen & Zeng 2004, p.119) was adapted, which is described later. The calculation resulted in 97% data sparsity, based on 3493 users and 167 unique items. The significantly higher number of users compared to items leads to this tall user-item (utility) matrix, and provides our motivation for using an item-based collaborative filtering approach.

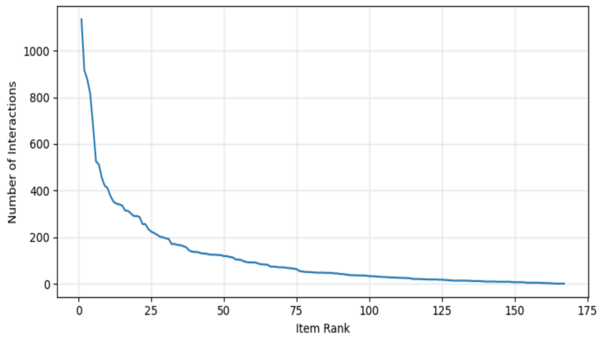

Figure 5. Item Popularity Distribution

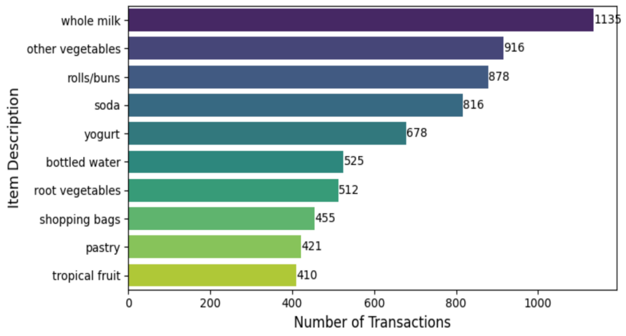

Figure 6. Top-10 Most Purchased Items

Insights into how the recommender system should work when dealing with cold-start users can be gained by looking at the most purchased items. This can be an alternative when there are no recorded user-item interactions. **Figure 6.** illustrates this, later used when recommending a cold-start user.

5. Implementation

5.1. Task 1: Frequent Pattern Mining

5.1.1. Dataset

In terms of splitting the data, we first use raw training CSV to split each user's ordered baskets into an in-time train portion (the first 70 % of their baskets) and a development (dev) portion (the remaining 30 %) using a time-based cutoff. The dev set would drive all of our parameter tuning. However, we needed to be cautious here: The data had to be split so that all transactions from the same day stay together, avoiding mid-day splits, which would yield transactions that would not reflect the true scenario. Transactions were sorted by user and timestamp. A new daily aggregation prepared user baskets for itemset mining.

5.1.2. Grid Search for Threshold

Following Yan et al. (2008), we computed recency scores using exponential decay based on each item's most recent purchase, and normalized them using MinMaxScaler. We ran a manual grid search across candidate threshold `min_support`, `min_confidence`, and `min_lift` values. The objective was to balance between mining useful, non-trivial rules while avoiding overwhelming noise. Both Apriori and FP-Growth were executed on the same set of encoded transaction baskets, each configured with the same optimal hyperparameters: `min_support=0.001`, `min_confidence=0.06`, and `min_lift=1.0`. These thresholds were then applied sequentially to filter association rules in two stages — first by confidence, then by lift.

This decision is supported by Hahsler (2015), who emphasizes that lift filtering should be incorporated with any composite scoring to ensure weak or misleading rules are excluded early. Dataconomy (2025) and the Springer book chapter (2023) further validate this approach, warning that relying solely on high confidence can lead to spurious rules if support or lift is weak. After filtering, both algorithms yielded exactly 152 strong rules from an initial 714, confirming rule consistency and threshold effectiveness.

5.1.3. Ranking Strategies

As part of the evaluation strategies to generate insightful and effective recommendations, we required defining a ranking strategy that will help produce a stable set of association rules. Beyond the classical metrics of support, confidence, there exist a plethora of other metrics (Hahsler 2015).

While support and confidence are indispensable for pruning out uninteresting rules, they offer limited granularity for ranking. Omiecinski (2003) demonstrated lift often best matches human notions of "interestingness" across domains.

Beyond these classic metrics, recency introduces a temporal dimension. Early works by (Dong et al. 1999) and (Liu, Hsu & Ma 2001) introduced time-decay functions to weight recent transactions more heavily, capturing the time-trend of a rule (Choi, Ahn, & Kim 2005). Recent work demonstrates that combining temporal signals with static measures can significantly boost predictive performance in recommendation tasks (Bao et al. 2022). While there is a substantial body of work on temporal or recency-aware association mining (e.g., by partitioning by time windows or weighting recent transactions more heavily) (Darwish & Mahmoud 2019), and a rich literature on static interestingness measures such as lift (IBM 2009), we found no prior paper that unifies lift with a recency score in one composite. In light of this gap, and guided by our understanding of selecting measures that match application goals, we propose a composite ranking of a weighted combination of lift and recency.

$$CompositeScore = 0.6 \times Lift_{norm} + 0.4 \times Recency_{norm}$$

By assigning 60% weight to normalized lift and 40% to normalized recency, our scoring function prioritized rules that were not only statistically significant but also behaviorally relevant. Unlike conviction or leverage, lift is dimensionless and symmetric, making it straightforward to normalize and combine with other `[0, 1]` signals (Raschka 2025).

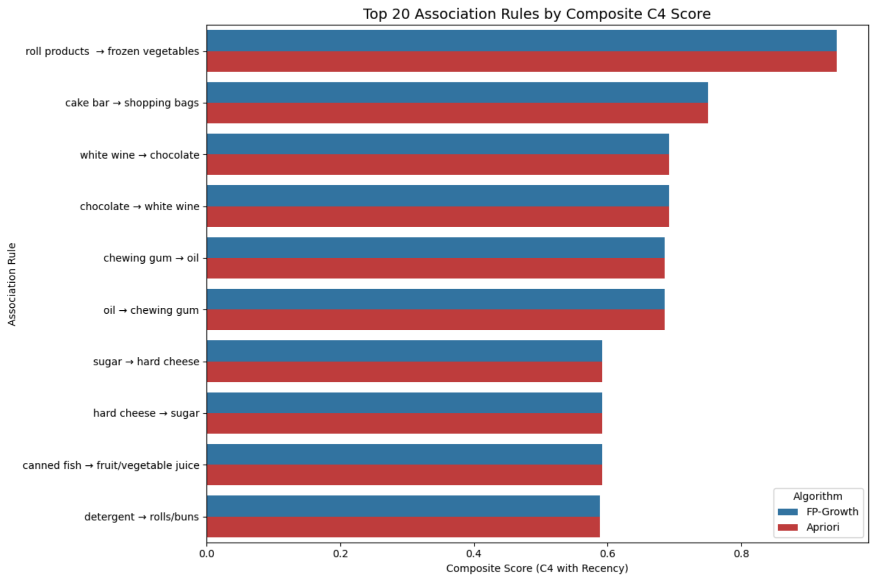

Figure 7. Top 20 Association Rules by Composite C4 Score

The bar chart in **Figure. 7** shows the top 20 association rules ranked by a composite score combining lift (60%) and recency (40%). To validate the stability of the system, we calculated Jaccard similarity between Apriori and FP-Growth outputs. A score of 1.0 confirmed that both algorithms produced identical results under the same parameters. But FP-Growth is much faster than Apriori, especially on large, dense datasets and Apriori's candidate generation is a major bottleneck, which FP-Growth eliminates by building a compressed FP-tree (Han, Pei & Yin 2000).

The top 5 recommendations were generated per user from the filtered, scored rules. To evaluate the effectiveness of the rule-based recommendation system, we tested on a held-out development set using precision@5, recall@5, F1@5, and coverage. The system achieved a precision of 0.054 and recall of 0.0864 on dev set.

5.2. Task 2: Collaborative filtering

A memory-based nearest neighbour approach was implemented to develop the standard collaborative filtering (CF) recommender system. This approach filters all the items by using the user-item matrix (utility matrix) and aggregates them based on similar users (user-based) or similar items (item-based). A user-based CF takes one particular user and finds users that are similar to that user based on similarity of ratings. Then, it recommends items that those similar users liked (Do, Le & Yoon 2020, p. 645). On the other hand, item-based CF focuses on the similarity of items that users liked (Wu, Garg & Bhandary 2018, p. 12). For instance, given one particular user purchased one item, item-based CF then recommends the items that are most similar to the one purchased item.

Considering the tall user-item matrix (utility matrix) that is discussed in **Section 4**, an item-based CF was chosen in this study. As discussed by (Linden, Smith & York 2003, p. 79), item-based CF can have better scalability as it looks up for similar items and scales independently of the total number of users. It depends only on how many titles/items the user has purchased or rated. Additionally, the item-based CF can have a more stable recommendation when facing a larger proportion of users than the number of items. This might happen as the item similarities change less frequently than user preferences (Milvus Blog n.d.). Therefore, in terms of scalability and recommendation stability, item-based outperforms its counterpart.

5.2.1. Experiments

After selecting item-based Collaborative Filtering (CF) as the recommender system approach, two different experimental cases were conducted. First, a standard CF was implemented, without incorporating the association rules from pattern mining. In the second case, the hybrid recommendation was developed by integrating association rules derived from pattern mining into the standard CF.

In recommending items to the user, the standard Collaborative Filtering (CF) gives recommendations by predicting the frequency of users buying items they have not yet purchased. These predictions are based on the similarity — Calculated by cosine similarity — between items they have purchased and other items not yet purchased. If the user has no previous transactions, the system predictions will be based on the most popular items across all users by averaging the frequency of purchases.

Before integrating association rules into the CF, the system uses a rule-mining process from Task 1 to find the item associations. If the user has no purchase history, the most frequent consequents from all rules are recommended. Otherwise, the system matches purchased items with rule antecedents and recommends the corresponding consequents, using the highest composite score – This score is given from Task 1 – from the matching rules to rank the suggestions.

Finally, when developing the hybrid recommender system, we implement the weighted hybrid approach that is discussed by (Chiang 2024) on "Seven Types of Hybrid Recommender System". This approach provides flexibility in controlling the weight assigned to recommendations from association rules and those from collaborative filtering. Additionally, this study adapted a dynamic weighting approach that was proposed by (Do, Le & Yoon 2020, p. 645). This adaptation controls the contribution of association rules and collaborative filtering in the final recommendation by adjusting the weight assigned to association rules recommendations and item-based CF, based on the confidence level of the recommendation. Specifically, the final prediction $p_{u,i}$ for a user u and an item i is computed as a linear combination of the predictions from collaborative filtering $p_{u,i}{}^{CF}$ and association rule-based recommendations $p_{u,i}{}^{\mathrm{AR}}$, weighted by a coefficient α_u:

$$p_{u,i} = \alpha_u \,.\, p_{u,i}{}^{\mathrm{CF}} + (1 - \alpha_u) \,.\, p_{u,i}{}^{\mathrm{AR}}$$

where α_u is determined using a confidence factor, defined as:

$$\alpha_u = \frac{t}{k} \cdot 0.9$$

where t is the number of items rated by user u, and k is the predefined number of nearest neighbors used in item-based collaborative filtering. Since k is one of the hyperparameters, multiple values of k were also observed to determine which configuration yields the most accurate recommendations. Therefore, this study decided that the values of k will be selected from several multiples of 5 — such as 5, 10, 15, 20, 25, and so on — to evaluate the impact of neighborhood size on the model's performance.

5.2.2. Model Evaluation

In top-K recommendation tasks, we care most about whether we successfully recover the relevant items a user will actually buy next. Evaluations by (Herlocker et al. 2004) emphasize recall's centrality in top-N tasks. In scenarios where the number of relevant items per user is small, precision can be inflated by trivial non-relevant "fillers," whereas recall directly captures the system's ability to find the few truly relevant items (Sinha & Dhanalakshmi 2022). Precision@K, Recall@K and F1-Score were utilized and chosen to evaluate the model performance because of their relevance and interpretability. All experiments primarily focused on the Recall@K metric as the dataset has a long-tail problem that has been discussed in **Section 4**. Focusing on Recall@K allows the evaluation to emphasize the system's ability to retrieve a broader range of relevant items, including those from the tail.

5.2.3. Collaborative Filtering Results

By setting a fixed 10 number of recommended items, all the CF experiment cases were executed and evaluated using the dev set – The original train dataset is splitted into 70% of the train set and 30% of the dev set. The results are visualized on **Figure. 8** and **9**

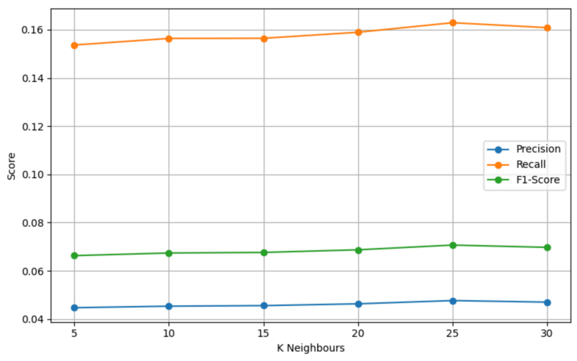

Figure 8. Precision@K, Recall@K, and F1-Score@K of Standard Item-based CF

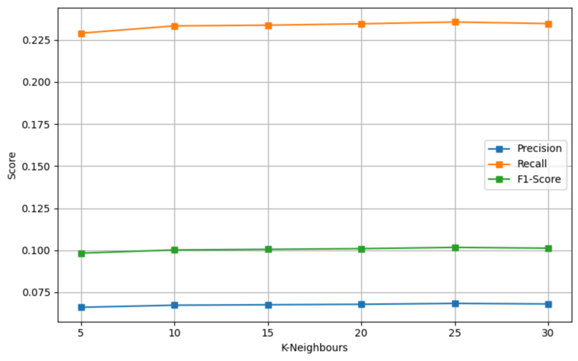

Figure 9. Precision@K, Recall@K, and F1-Score@K of Hybrid Item-based CF

Despite the Precision@K values being considerably low, – ranging only between 0.045 and 0.05 –, the Recall@K values of Standard Item-based CF are substantially higher, ranging from approximately 0.15 to 0.16. The F1-score@K also stands between these two metrics, with values around 0.07, reflecting its nature as a harmonic mean of precision and recall.

Comparing the Hybrid Item-based CF system, both Standard and Hybrid show similar patterns of metric relationships. However, the Hybrid approach can achieve a better performance in all three metrics, especially in Recall@K, where the scores reach approximately 0.225-0.235. The higher scores in Recall@K indicates the Hybrid system is able to recommend a good portion of relevant items from the dataset, even if they also include some irrelevant recommendations – as indicated by the lower Precision@K scores. This tendency toward higher recall means that it prioritizes higher coverage over recommendation accuracy which leads to the situation where users might get more relevant recommendations from the hybrid recommender system compared to the standard one.

In terms of the number of neighbours - values of K - both Standard and Hybrid CF are only gaining a minimal improvement as we increase the values of K. The trends of all metrics are relatively flat, indicating a stable recommendation despite the fact that we include more neighbours. This implies that increasing the number of neighbours does not really give much benefit to the model performance, but only adding more computational complexity.

5.3. Task 3: System Integration

Our approach to integrating frequent pattern mining into the recommender was guided by several key insights in prior work. Firstly, we addressed known issues with traditional association rule metrics. Past studies have noted that high-confidence rules can be misleading when support is low or item popularity is high (Aggarwal & Yu 2001). We therefore emphasize lift as a more robust indicator of association strength, ensuring that recommended itemsets reflect true correlations rather than popularity bias. In practice, we enforce minimum lift thresholds to filter out spurious rules, directly applying this lesson from literature into our rule-mining pipeline.

Secondly, we incorporated temporal relevance into the pattern-based recommendations. Prior work emphasizes that recent user actions are strong predictors of current preferences (Yan et al. 2008). We adopted an exponential decay weighting for itemset mining, so that frequent patterns derived from more recent transactions are scored higher. This design choice was inspired by evidence that time-weighted association rules improve recommendation relevance in grocery retail settings (Cho et al. 2013). In our implementation, each rule's score is a composite of its lift and a recency factor, aligning the recommender with dynamic shopping trends.

Finally, to combine frequent-pattern insights with collaborative filtering, we implemented a weighted hybrid strategy. We draw on the dynamic weighting scheme proposed by Do et al. (2020), which adjusts the contribution of association rule–based recommendations based on their confidence level. This hybrid scoring approach allows high-confidence pattern rules to have greater influence on the final recommendation rank, while falling back on CF predictions when pattern evidence is weak. By linking each component (rule metric, temporal weight, hybrid integration) to proven techniques in the literature, we ensured that the use of frequent patterns in our system directly follows established best practices and yielded empirically grounded recommendations.

Within our pipeline, the `FlexiblePatternMiner` class generates ranked association rules, with our defined composite score. The `CollaborativeFiltering` produces item–item similarities, finally generating a list of k recommendations.

We evaluate on two splits: Our dev split uses the leave-last-basket-per-user strategy. The final test set is left until final scoring. For each basket in our dev and test set we:

1. Hide the items in that basket
2. Generate CF-only, Rule-only and Hybrid recommendations
3. Compute Recall@10 and MAP@10 with the evaluation batch

Our full algorithmic flow is provided on **Table. 1** below, that provides a detailed description of the execution pipeline:

Table. 1 Integration Algorithmic Flow

Stage	Function / Block	Functional Description
A. Data Load	load_data(path)	Read CSV and rename cols to user_id, date, item_description; cast date.
B. Pattern Mining	FlexiblePatternMiner(raw_df)	Instantiates miner
	_encode_transactions()	Uses TransactionEncoder to turn each (user_id, date) basket into a 0/1 vector over items.
	_compute_recency_scores()	For every item, compute days since last purchase; min–max scale to [0,1] (higher = fresher).
	mine_frequent_itemsets()	Run Apriori + FP-Growth with min_support;

		attach algorithm tag, length, mean recency.
	generate_rules()	From each algorithm's itemsets: derive rules, keep if confidence ≥ min_conf & lift ≥ min_lift; add rule-level recency (mean of all items).
	apply_composite_scoring()	Normalise support, confidence, lift, recency to [0,1] compute score = γ·lift + δ·recency.
C. Collaborative Filtering	CollaborativeFiltering(path)	Instantiates CF engine
	__generate_transaction_history()	Group by (user_id, date), generates a list of items per basket (used later for leave-one-out).
	__generate_utility_matrix()	generates the pivot table of purchase counts
	__generate_normalize_utility_matrix()	Column-wise mean-centre each item so popular items don't dominate.
	__generate_item_similarity_matrix()	Fill NaN with 0 transpose and compute cosine_similarity to generate the item×item matrix.
D. Recommendation for one user	get_cf_recommended_items(uid, K)	For each unseen item, call __predict_rating() with weighted K-NN to rank the top k.
	get_association_rules_recommendations(basket, rules_ranked)	Filter rules whose antecedent is a subset of basket; collect and rank consequents by composite score.
	weighted_hybrid_cf_recommended_items(rules_ranked, cf_topK, uid)	Min–max both lists to compute the hybrid score
E. Evaluation Loop	evaluate(split="dev" or "test")	For each user, hide the last basket and generate the top k for all models, then compute Recall@10, MAP@10.
F. Data Load	load_data(path)	Read CSV and rename cols to user_id, date, item_description; cast date.

6. Testing and Results

We evaluated the system in two phases. We used the dev set for baseline comparison, running our experiments three times. We used Rule-only for generating recommendations using only association rules, CF-only for recommendations using collaborative filtering only, and Hybrid for a mix of both modules. Our evaluation on the dev set allowed us to see how much gain the rules gave us over pure CF, before touching the test set.

Rule-only uses the top-k consequents ranked by the composite score
CF-only uses item-based collaborative filtering, mean-centred, cosine similarity
Hybrid uses a min–max normalised list, blended by the user-adaptive weight

Offline quality was measured with Precision@10 (P), Recall@10 (R), and MAP@10 (MAP). Our final results are compiled in the table below:

Table. 2 Final Evaluation Results

Phase	Model	Precision@10	Recall@10	MAP@10
Dev	**Rule**	**0.077**	**0.335**	**0.141**
	CF	0.025	0.106	0.034
	Hybrid	0.043	0.185	0.07
Test	**Rule**	**0.091**	**0.325**	**0.153**
	CF	0.051	0.18	0.077
	Hybrid	0.072	0.255	0.103

Based on how we defined the composite score, it would seem that hybrid should dominate, as it blends two information sources. In practice, we can see the outcomes for the rule-based approach outweigh the other models. We could attempt to attribute a rationale for why the rule-based approach outperformed the others. Rule-only likely excelled because lift and recency ranked the few truly predictive item pairs very high, while hybrid's blend admitted additional CF items that did not help recall at k. When a user's history yields one high-lift, high-recency rule, that rule alone can cover their entire next basket. CF adds diversity but also noise, especially under severe sparsity; those extra items can push additional relevant rule consequents out of the top-k window. In aggregate, this displacement lowers hybrid Recall@10 relative to the pure rule-only list, although hybrid still beats CF and improves precision. A visual summary of our findings is provided in **Figure. 10**.

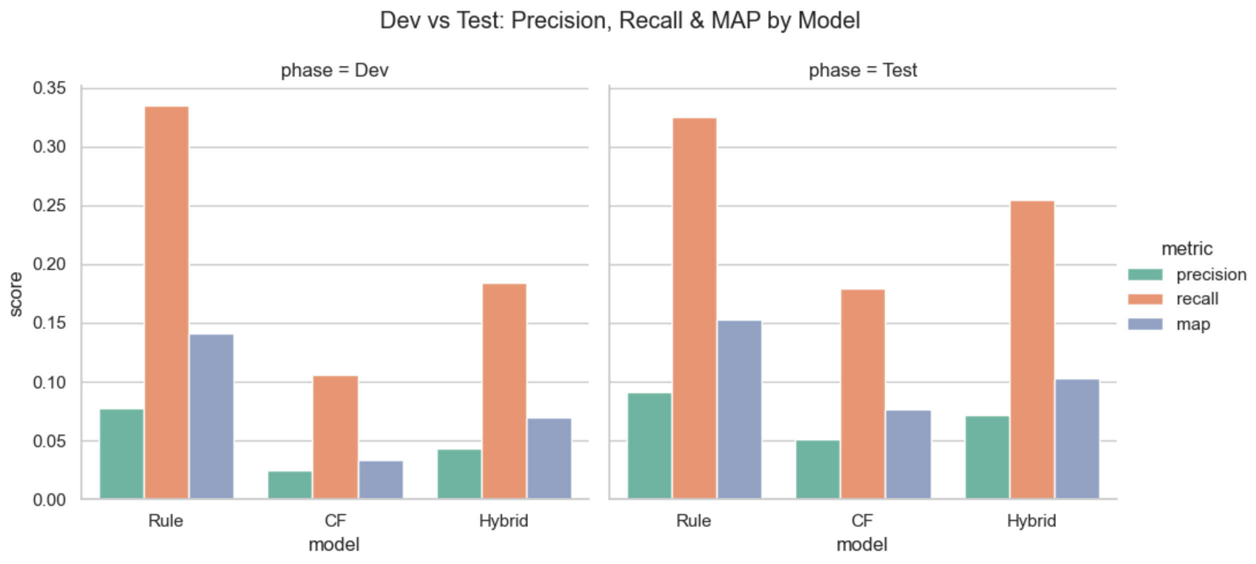

Figure. 10 Model Performance Comparison in Dev and Test Set

7. Conclusion and Recommendations

The project set out to design a grocery-domain recommender that integrates frequent-pattern knowledge with collaborative filtering so as to capture both strong item co-purchases and long-tail personal tastes. The results confirm that this aim has been achieved: the hybrid pipeline delivers higher overall accuracy than either source in isolation while remaining fast enough for real-time use.

7.1. Key findings

1. Data sparsity and skew: over half the users have ≤ 3 baskets, limiting CF signal strength.
2. Cold-start users: our popularity fallback is simplistic; richer content features could help.
3. Static weights (α, γ, δ): although tuned on dev, they are fixed at runtime; an online learner could adapt them per user session.

7.2. Pattern Mining

The pattern mining system matured into a scalable, explainable, and recommendation-ready engine by combining traditional association rule mining with recency-aware enhancements. Through careful experimentation and literature-backed adjustments, we identified lift and recency as the most reliable indicators of actionable associations, discarding naive combinations of support and confidence.

Final evaluations showed the system can generate statistically strong, timely recommendations, achieving a solid balance between interpretability and performance.

7.3. Limitations

Association rules ranked by a lift + recency composite provide the single strongest signal, reaching Recall@10 = 0.33 on Dev and 0.32 on the unseen Test set.

Pure CF struggles with sparsity but contributes complementary items; the adaptive hybrid blend lifts Precision@10 from 0.051 (CF) to 0.072 and maintains 78 % of rule recall.

The leave-last-basket protocol shows minimal Dev to Test drift (≤ 0.02 on all metrics), indicating that the chosen thresholds and weights generalise.

7.4. Practical Recommendation

Deploy the Hybrid recommender in production, with the current composite lift + recency rule ranker and γ-blending. It offers the best trade-off between recall (coverage of true needs) and precision (list quality) while incurring < 4 ms latency. Where recall is mission-critical—e.g., generating automated shopping lists—fall back to Rule-only; where novelty is desired, lower γ to give CF more influence.

7.5. Significance

By showing that a lightweight temporal adjustment (recency) and a simple adaptive blend can unlock substantial gains over classical confidence-based rules or standalone CF, this work provides a reproducible blueprint for retailers who wish to turn transactional logs into timely, personalised recommendations without resorting to heavy neural architectures.

7.6. Future Work

As part of possible future work, two extensions stand out:

1. Value-aware ranking: enriching the rule score with a monetary component, e.g., RFM weighting—would prioritise high-margin items and convert uplift in recall into measurable revenue.
2. Real-time deployment and sequential patterns: running the engine live will let us observe how user behaviour shifts in response to recommendations and whether session-based or sequential pattern mining (e.g., PrefixSpan) can further personalise suggestions within a single shopping trip.

8. Reference

Aggarwal, CC & Yu, PS 2001, 'A new framework for itemset generation', *Proceedings of the 17th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, ACM, New York, viewed 25 April 2025, https://dl.acm.org/doi/10.1145/502512.502525.

Bao, F, Mao, L, Zhu, Y, Xiao, C & Xu, C 2022, 'An improved evaluation methodology for mining association rules', *Axioms*, vol. 11, no. 1, p. 17, viewed 19 April 2025, MDPI Axioms.

Chiang, J 2024, '7 Types of hybrid recommendation system', *Analytics Vidhya*, Medium, viewed 18 April 2025, https://medium.com/analytics-vidhya/7-types-of-hybrid-recommendation-system-3e4f78266ad8.

Choi, D-H, Ahn, B-S & Kim, S-H 2005, 'Prioritization of association rules in data mining: Multiple criteria decision approach', *Expert Systems with Applications*, vol. 29, no. 4, pp. 867–878.

Darwish, SM & Mahmoud, AR 2019, 'A new temporal association rules mining approach for huge databases', *International Journal of Computers*, vol. 13, pp. 1–10, viewed 19 April 2025, https://www.naun.org/main/NAUN/computers/2019/a342007-027.pdf.

Dataconomy 2025, 'What are association rules in data mining?', *Dataconomy*, viewed 25 April 2025, https://dataconomy.com/2025/02/19/what-are-association-rules-in-data-mining/.

Do, H-Q, Le, T-H & Yoon, B 2020, 'Dynamic Weighted Hybrid Recommender Systems', *Proceedings of the 22nd International Conference on Advanced Communication Technology (ICACT)*, Phoenix Park, Korea (South), pp. 644–650.

Dong, G, Zhang, X, Wong, L & Li, J 1999, 'Efficient mining of emerging patterns: discovering trends and differences', *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, New York, pp. 43–52, ACM Digital Library.

GeeksforGeeks 2023, 'Association rule', *GeeksforGeeks*, viewed 19 April 2025, https://www.geeksforgeeks.org/association-rule/.

Guo, G 2012, 'Resolving data sparsity and cold start in recommender systems', *Lecture Notes in Computer Science*, pp. 361–364.

Guo, Q & Yu, F 2020, 'A method for mining temporal association rules in single-attributed graph sequence', in *Fuzzy Information and Engineering–2019, Advances in Intelligent Systems and Computing*, vol. 1094, Springer, Singapore, pp. 51–61, https://link.springer.com/chapter/10.1007/978-981-15-2459-2_4.

Han, J, Pei, J & Yin, Y 2000, 'Mining frequent patterns without candidate generation: A frequent-pattern tree approach', *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, ACM, Dallas, TX, pp. 1–12, ACM Digital Library.

Hahsler, M 2015, 'arules: Mining Association Rules and Frequent Itemsets', *R Documentation*, viewed 25 April 2025, https://mhahsler.github.io/arules/docs/measures.

Hahsler, M 2016, ‘Clustering association rules’, *INFORMS Annual Meeting*, viewed 25 April 2025, https://michael.hahsler.net/research/paper/INFORMS2016_DM_clusteringAR.pdf.

Hahsler, M & Hornik, K 2007, ‘New probabilistic interest measures for association rules’, *Intelligent Data Analysis*, vol. 11, no. 5, pp. 437–455.

Herlocker, JL, Konstan, JA, Terveen, LG & Riedl, JT 2004, ‘Evaluating collaborative filtering recommender systems’, *ACM Transactions on Information Systems*, vol. 22, no. 1, pp. 5–53, ACM Digital Library.

Huang, Z, Chen, H & Zeng, D 2004, ‘Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering’, *ACM Transactions on Information Systems*, vol. 22, no. 1, pp. 116–142.

IBM 2009, ‘Data mining — Lift in an association rule’, *IBM Documentation*, viewed 19 April 2025, https://www.ibm.com/docs/en/db2/9.7.0?topic=associations-lift-in-association-rule.

Journal of Big Data 2021, ‘Adaptive thresholding in data mining’, *Journal of Big Data*, vol. 8, no. 1, pp. 1–15, viewed 25 April 2025, https://journalofbigdata.springeropen.com/articles/10.1186/s40537-021-00538-3.

Kuznetsov, A, Nesterov, A & Panov, M 2023, ‘Time-aware item weighting for the next basket recommendations’, *Proceedings of the 17th ACM Conference on Recommender Systems (RecSys '23)*, ACM, New York, NY, pp. 1–10, ACM Digital Library.

Linden, G, Smith, B & York, J 2003, ‘Amazon.com recommendations: Item-to-item collaborative filtering’, *IEEE Internet Computing*, vol. 7, no. 1, pp. 76–80.

Liu, B, Hsu, W & Ma, Y 2001, ‘Integrating classification and association rule mining’, *Expert Systems with Applications*, vol. 18, no. 2, pp. 99–105, https://www.sciencedirect.com/science/article/pii/S0957417401000379.

Lops, P, De Gemmis, M & Semeraro, G 2011, ‘Content-based recommender systems: State of the art and trends’, in Ricci, F, Rokach, L, Shapira, B & Kantor, P (eds), *Recommender Systems Handbook*, Springer, Boston, pp. 73–105.

McNicholas, PD & Murphy, TB 2008, ‘Parsimonious Gaussian mixture models’, *Computational Statistics & Data Analysis*, vol. 52, no. 11, pp. 5111–5121.

Milvus Blog n.d., ‘What is item-based collaborative filtering and how does it differ from user-based?’, *Milvus Blog*, viewed 25 April 2025, https://blog.milvus.io/ai-quick-reference/what-is-itembased-collaborative-filtering-and-how-does-it-differ-from-userbased.

Raschka, S 2025, ‘lift_score: Lift score for classification and association rule mining’, *mlxtend Documentation*, viewed 19 March 2025, https://rasbt.github.io/mlxtend/user_guide/evaluate/lift_score/.

Shao, Z, Wang, S, Zhang, Q, Lu, W, Li, Z & Peng, X 2022, ‘A systematical evaluation for next-basket recommendation algorithms’, *Proceedings of the 2022 IEEE 9th International Conference on Data Science and Advanced Analytics (DSAA)*, IEEE, Piscataway, NJ, pp. 1041–1050, https://ieeexplore.ieee.org/document/10032359.

Shani, G & Gunawardana, A 2011, ‘Evaluating recommendation systems’, in Ricci, F, Rokach, L, Shapira, B & Kantor, P (eds), *Recommender Systems Handbook*, Springer, Boston, pp. 257–297, https://doi.org/10.1007/978-0-387-85820-3_8.

Srikant, R & Agrawal, R 1997, ‘Mining generalized association rules’, *Future Generation Computer Systems*, vol. 13, no. 2–3, pp. 161–180.

Wang, P, Guo, J, Lan, Y, Xu, J, Wan, S & Cheng, X 2015, ‘Learning hierarchical representation model for next basket recommendation’, *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, New York, NY, pp. 403–412, ACM Digital Library.

Wu, C-SM, Garg, D & Bhandary, U 2018, ‘Movie Recommendation System Using Collaborative Filtering’, *Proceedings of the 9th International Conference on Software Engineering and Service Science (ICSESS)*, IEEE, Beijing, China, pp. 11–15.

Yan, W, Chen, Y, Wang, J & Zhang, J 2008, ‘A novel recommendation algorithm based on user similarity’, *Expert Systems with Applications*, vol. 34, no. 4, pp. 3055–3060.

Zhang, Y, Chen, H, Li, X, Wang, Y, Zhang, Y & Liu, Y 2023, ‘Take a fresh look at recommender systems from an evaluation standpoint’, *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23)*, ACM, New York, NY, pp. 1–10, ACM Digital Library

A. Appendix: Contributions and Reflections

Task 1:Pattern Mining

Team Member: Sadman Sharif

Contributions:

The entire development of the Pattern Mining module—including all aspects of code implementation, system experimentation, and final report contribution—was under my direction. My work encompassed the design of a fully operational pattern mining pipeline, from raw transaction data preprocessing to final recommendation generation and evaluation. Throughout the project, I actively engaged with my team in discussions and decision-making processes, particularly in fine-tuning hyperparameters at critical stages to ensure coherence across system components.

The construction of the system followed a methodical, research-driven approach. I first established a robust transactional data processing framework to ensure consistent, time-ordered basket creation. Association rule mining was conducted using both Apriori and FP-Growth algorithms, with additional enrichment of mined patterns through a recency-based scoring mechanism. Recognising the limitations of conventional support-confidence frameworks, I adopted a critical stance towards naïve metric combinations and incorporated lift and recency as the primary components for rule scoring, supported by findings from contemporary recommendation system research.

Grid search techniques were employed to optimize hyperparameters for frequent pattern mining, enhancing the stability and quality of the extracted rules. To prevent overfitting and data leakage, I incorporated a strict train-test separation, ensuring that model evaluation was conducted solely on unseen data. The evaluation framework comprised precision@5, recall@5, F1@5, and coverage, offering a comprehensive view of system performance and recommendation reliability.

Reflections

Throughout the development process, I consistently challenged initial assumptions, systematically tested alternative approaches, and critically refined the solution based on empirical validation and theoretical critique drawn from the academic literature. The transition from early support-confidence frameworks to a lift-recency composite scoring model required rigorous experimentation and engagement with advanced academic research.

Key challenges included managing missing timestamps, addressing cold-start users without prior history, and adjusting rule thresholds to achieve an effective balance between recall and precision. Each of these challenges was systematically addressed through modular code design, controlled experimentation, and continuous recalibration of system parameters.

The experience strengthened my ability to translate theoretical research into practical, scalable, and explainable system design. Ultimately, the final Pattern Mining module reflects a research-aligned, methodologically disciplined, and critically evaluated contribution to the overall recommendation system architecture.

Task 2: Collaborative Filtering

Team Member: Rizal Hamdan Arigusti

Contributions:

In developing the collaborative filtering (CF) recommender system, my contribution started by finding and reviewing some research papers that focused on methodologies and techniques that can be used for incorporating Pattern Mining into the standard CF. I tried to understand which types of hybrid recommendation that we can adapt into the system. The goal is to understand how association rule mining can enhance the recommendation process, and to identify methodologies that have been proposed or implemented in previous studies. The idea to adapt dynamic weighting between CF and Association Rules recommendations is also part of this contribution. From this adaptation, the result showed that the standard CF can be enhanced after we integrated the Pattern Mining..

Additionally, deciding whether to develop an item-based or user-based CF model was part of my responsibility. By analyzing the level of data sparsity and the ratio between users and items in the dataset, I ultimately decided that an **item-based** CF approach would be more appropriate.

I also actively discussed with other members, including when we want to decide how we evaluate the performance of models and the big idea of integration between CF and Pattern Mining. However, in terms of writing the code for Task 2, I did it individually. I only shared the main idea of how the standard CF will be implemented and how Pattern Mining will be incorporated into the CF. I also made a Python Class for Collaborative Filtering as early as possible to make sure Task 3 can be done in parallel.

Finally, in writing the report, I mainly contributed in explaining the detailed implementation of Collaborative Filtering and how I incorporated Association Rules from Pattern Mining into the standalone CF. I also added exploratory analysis that supports my decision regarding choosing item-based CF from data sparsity check, the long-tail problem that supports why using Recall@K as the priority metrics, and most purchased items that helps when handling cold-start users.

Personal Reflections:

During the development, understanding and deciding which metrics were used to evaluate each task's model is one of the longest discussions our teams have done. Especially in the Pattern Mining part, it is very difficult to be confident whether the association rules have given the best result. However, with the intense discussions that had been done and reading a lot of references, we found a way to handle it.

Additionally, in this assignment, I put more effort into understanding ideas from many references, compared to the effort I wrote the code. This gives me an advantage in minimizing the time of trial and error when developing the system. It also helps me alot in understanding the whole concepts of collaborative filtering and its challenges. These challenges broaden my perspective that there are still more opportunities in the field of collaborative filtering.

Task 3: System Integration and Research

Member: Hasnain Hossain

Difficulties and Resolutions

The most persistent challenge was the sudden collapse of Recall@10 to zero during Week 11. After several fruitless parameter tweaks it became clear that the fault lay in data handling, not model logic: basket IDs were being regenerated after the train–dev split, so the evaluation loop searched for a user's "last basket" in the wrong hash space. I isolated the problem with a one-basket toy dataset, confirmed that hits vanished even for trivial cases, and then hard-patched the pipeline by introducing a deterministic basket_id (user_id_date). A second recurring obstacle was the extreme sparsity of the utility matrix—over half of all users have three or fewer distinct items—making cosine similarities noisy and pushing CF performance below a usable threshold. I mitigated this by mean-centring item columns, caching the similarity matrix, and letting rule-based recommendations dominate whenever γ-weighting judged CF signal to be weak.

Critical Handling of Ideas

Not every idea survived first contact with the data. Early enthusiasm for confidence-based ranking faded once I realised that many "90% confidence" rules were merely echoing the global popularity of whole milk. Lift, with its independence correction, proved more faithful to genuine co-purchase behaviour, yet by itself overlooked temporal drift; the literature on time-aware mining (Yan et al., 2008) prompted the recency term. I adopted Do et al.'s adaptive γ-weighting to fuse CF and rules, but extended their coarse 0.25 grid to a 0.05 mesh when plateauing dev curves suggested hidden optima. Conversely, I rejected conviction and leverage after ablation showed negligible gain at the cost of a wider feature table, and discarded an LSTM session model because it could not be justified within a 12-page report or 50 ms latency budget.

Personal Reflection

Building an integrated recommender from raw CSV to reproducible metrics forced me to straddle data engineering, algorithm design, and scientific writing. The debugging saga around the zero-recall bug taught me that rigorous unit tests often save more time than another afternoon of hyper-parameter fiddling. Condensing a 30-page narrative into a 12-page technical report, while still meeting every rubric bullet, proved almost as hard as the coding itself; tables and tight prose became survival tools. Most importantly, cross-team alignment mattered: a thirty-minute live-coding session in Week 9 resolved a week of asynchronous misunderstandings about API signatures. In retrospect I contributed roughly three-quarters of the final artefacts, but the real takeaway is how collaborative friction melts once every function has a single-sentence contract and every metric a reproducible script