International Journal of Engineering in Computer Science

E-ISSN: 2663-3590
P-ISSN: 2663-3582
IJECS 2024; 6(2): 01-08
Received: 01-05-2024
Accepted: 06-06-2024

Pooja Rani
Computer Science and Engineering, Faculty of Engineering, Baba Mastnath University, Rohtak, Haryana, India

Rahul
Computer Science and Engineering, Faculty of Engineering, Baba Mastnath University, Rohtak, Haryana, India

Dr. Banita
Computer Science and Engineering, Faculty of Engineering, Baba Mastnath University, Rohtak, Haryana, India

Corresponding Author:
Dr. Banita
Computer Science and Engineering, Faculty of Engineering, Baba Mastnath University, Rohtak, Haryana, India

Exploring new frontiers: Novelty in recommender systems

Pooja Rani, Rahul and Dr. Banita

DOI: https://doi.org/10.33545/26633582.2024.v6.i2a.117

Abstract
With the unimaginable vast amount of data on the World Wide Web, finding useful information has become increasingly challenging. To address this information overload, recommender systems have been developed. This paper presents the collaborative filtering approach, and a new novelty detection approach so to improve the ranking of novel items. These personalised recommender system keeps the user's interest by recommending items based on his interests and the order of occurrences. The automatic detection of novel items leads to enhanced experience that adds more information to already known information by user. The past behaviours of the user assist the user in making more effective decisions that enhances the satisfaction level of the user.

Keywords: Recommender system, collaborative filtering, automatic detection

Introduction

The systems often assume that user preferences remain constant, an assumption that is frequently inaccurate. Therefore, prediction and recommendation are the core tasks of recommender systems aimed at suggesting useful, relevant, and interesting items to users [1, 2]. With the rapid expansion of online retail and streaming services like Amazon, Spotify, and Netflix, consumers are bombarded with an overwhelming array of choices. The sheer volume of accessible information surpasses an individual's ability to search and consume it all, leading to two significant issues: a surplus of options and an overload of frequently encountered yet less desired content. However, as noted in , items with the highest ratings are not always the most useful to users. For instance, recommending the most popular items (those with numerous ratings) may not always align with user preferences, as such items are easily discoverable independently. Moreover, an item with a high predicted rating may not necessarily be useful if the prediction is based on limited data, such as ratings from only a few users [3]. In this scenario, recommender systems become essential tools for curating appropriate content for users. Items are ranked in descending order based on their predicted ratings or scores. The items with the highest predicted ratings or scores are then recommended to the user.

Figure 1 illustrates analyzing the collection of consumption histories. The longer the users interact with the system, the more likely recommendations will be dominated by a few items that have been frequently consumed, collapsing diversity and novelty in the system. As a result, current systems face two main challenges - tracking the changes in user preferences, and the absence of historical information to recommend new items to a user in the system.

If a recommender system accurately predicts ratings for items that the user is not interested in, those predictions are not helpful. If the goal is to recommend the top 10 items a user likes, it is more important that these items are relevant and useful rather than having perfectly predicted ratings. Additionally, recommending the most popular items might not be beneficial since users can find those on their own. Similarly, recommending items too similar to those already consumed can lead to user boredom. Therefore, the evaluation measure should be tailored to the user's tasks, requirements, and the recommendation domain [4].

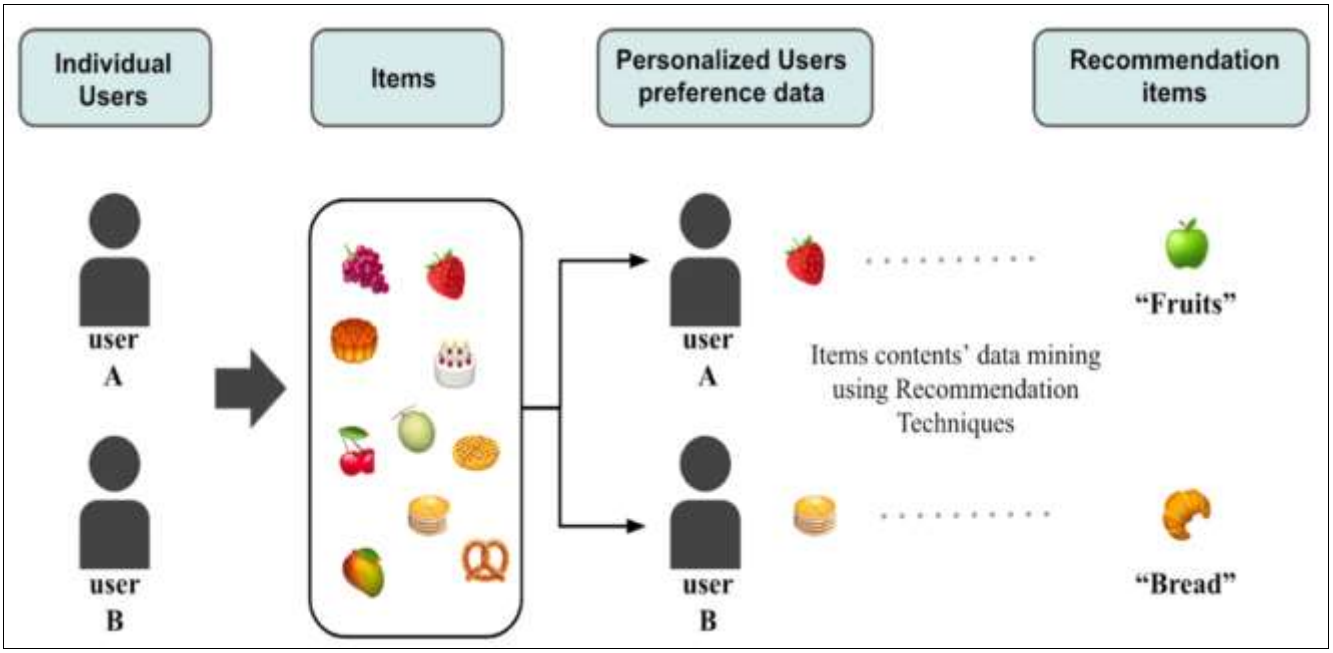

Fig 1: System Generate Ranked Itemized Recommendations through a Cycle by Analyzing Consumption Histories of Users in the System

For instance, it's common to hear users say, "I'm tired of watching this movie," indicating a loss of trust in the recommender system and a tendency to seek out new content independently. In this case, items are typically ranked in descending order of preference, with the most preferred items appearing first [5]. Specifically, in datasets with explicit ratings like numerical scales, items are ordered by predicted ratings. In contrast, datasets with binary or unary ratings order items based on predicted scores, recommending only those items predicted to be liked in the case of binary ratings. As items lose their appeal through repeated exposure, the quality of recommendations deteriorates, presenting a dual challenge: monitoring the evolution of user preferences and lacking historical data to suggest new items.

The core question becomes: how can recommender systems detect when a user is seeking novelty or is bored? Additionally, how can these systems identify which users are open to recommendations of entirely new items without prior user history? To address these challenges, we propose a novel and adaptable learning framework designed to capture user tastes, monitor their evolution, and develop personalized recommendation strategies. This approach, known as Novelty Learning, views preferences as dynamic combinations of consumption habits. These bundles of items are preferred by users in close temporal proximity, yet their composition is allowed to evolve.

Our innovative model, termed collaborative filtering, focuses on the temporal closeness of consumed items within user histories. Collaborative filtering algorithms are typically evaluated through experiments that rely on user preferences for a set of items. Suitable datasets for collaborative filtering consist of real users' feedback on real items within operational systems. This feedback can be explicit, involving numerical or ordinal ratings on scales like 1-5, or binary where ratings are either +1 (like) or -1 (dislike), or unary where only positive feedback is considered. Implicit feedback, on the other hand, is inferred from user actions rather than explicitly stated [6].

Background of the Study

As digital media consumption becomes increasingly prevalent, online platforms must tailor content to align with user preferences. In areas like music and movies, understanding these preferences is crucial. Traditional recommender systems focus on identifying similarities, either among users—by basing recommendations on the consumption patterns of users with analogous histories—or among items. In movie recommendation, various strategies have been employed to leverage both user and item similarities to anticipate preferences [7].

One notable approach is the latent factor model proposed by Guilherme Brandão Martins *et al.* [8], which utilizes weighted matrix factorization combined with deep convolutional neural networks. This model demonstrates that incorporating additional attributes can significantly enhance the accuracy and relevance of recommendations, especially when user interaction data is limited. Qusai Shambour *et al.* [9] show that the proposed HUIT recommendation approach significantly outperforms state-of-the-art recommendation algorithms in terms of recommendation accuracy and coverage, as well as significantly alleviating data sparsity, cold-start user and cold-start item problems.

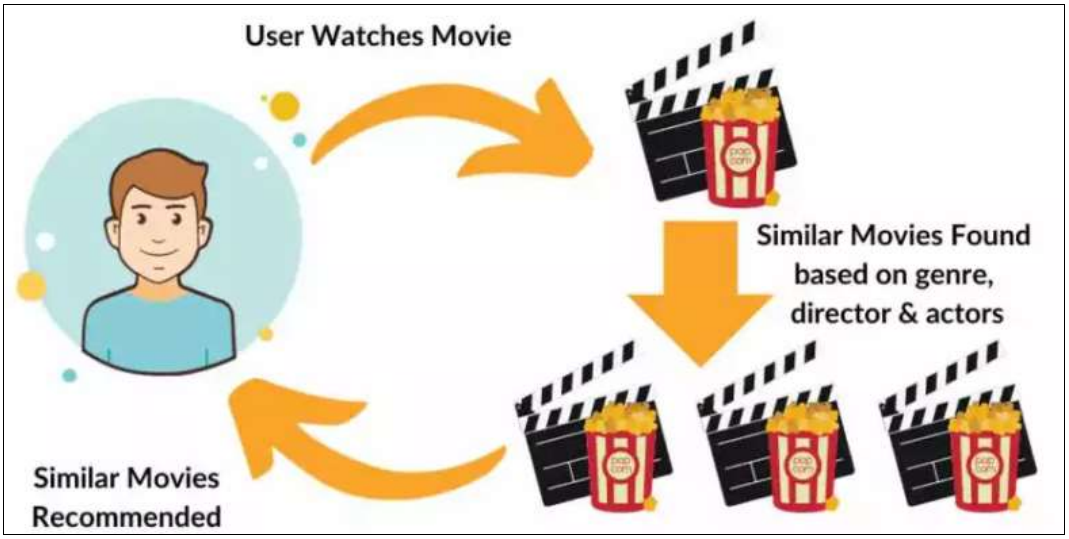

Fig 2: Personalization Revolution by Tailoring Suggestions to Each User's Unique Preferences

Figure 2 illustrates listening preferences within a broad user base, focusing on the gradual shifts in collective preferences over time and largely overlooking individual variations. In contrast, Yen-Yao Wang *et al*. [10] investigated how social contexts and the preferences and ratings of others influence personal recommendations. Although ratings offer explicit feedback, they have limitations—they may not accurately capture the user's immediate reaction to content and can be swayed by the opinions of others and trending topics. Additionally, this research presumes that users' core interests remain relatively stable, with shifts in preferences primarily driven by changes in public attention.

In the realm of music, individual user preferences evolve over time, making it crucial to treat them as inherently dynamic. This evolution means that even once-beloved tracks can lose their appeal through overexposure, as familiarity often leads to boredom [13, 14]. Users also have a strong inclination for discovering new content [15], which presents a challenge for recommender systems that lack sufficient data on new items. Rapid shifts in user preferences can severely impact the effectiveness of a recommender system unless it can adapt and monitor these changes. Additionally, the rate and manner in which preferences change vary from user to user, necessitating personalized recommendations that cater to each individual's propensity for exploring new options.

Viewing a segment of listening history as a collection of co-consumed items, we can see that music preferences shift between these collections, or "bundles," which we call tastes. The similarity between songs is established by their co-occurrence within the same consumption bundle, highlighting a similarity based on temporal proximity. To better monitor and adapt to changes in user preferences, we leverage this recurring pattern in music listening. By focusing on the proximity of songs within users' listening histories, we can enhance the recommender system's ability to offer both collaborative and novel recommendations.

Collaborative Filtering by Discovering Items

Collaborative filtering (CF) has been the most widely considered approach. Different aspects of CF such as classifications, approaches, data extraction methods, similarity metrics, prediction approaches, and performance metrics are studied meticulously. Train to predict the opinion of the user will have on the different items and be able to recommend the items to each user based on the user's previous likings and the opinions of other like-minded users.

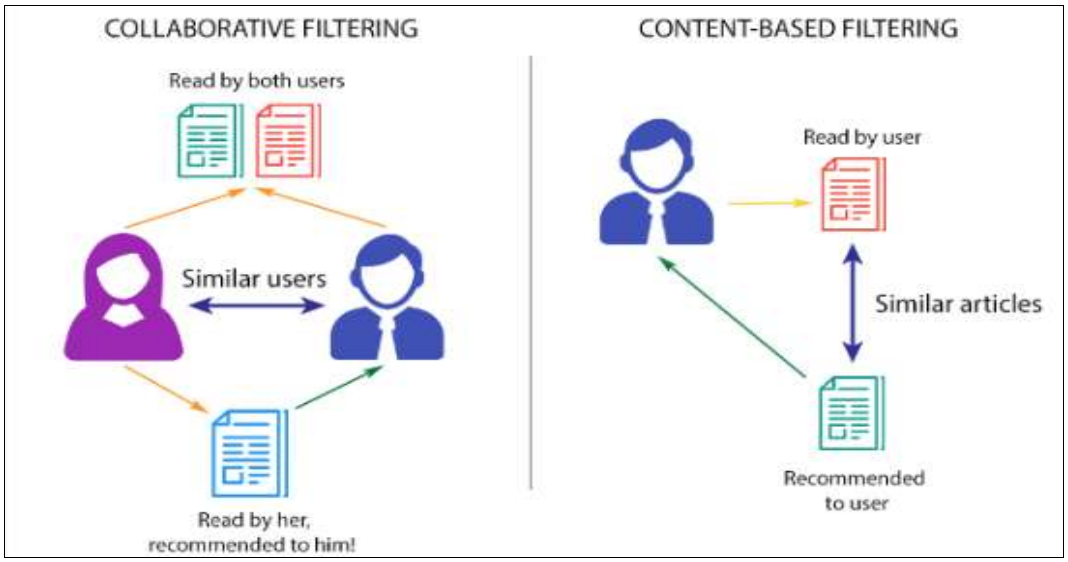

Fig 3: Differentiating the Collaborative Filtering and Content Based Filtering Technique

The task of a CF algorithm is to find item lines of two forms:

Prediction - a numerical value, expressing the predicted likeliness of an item the user hasn't expressed his opinion about.

Recommendation - list of N items the active user will like the most.

Item-Based CF Technique for Predicting Novel Items

- Looks into the set of items the target user has rated & amp; computes how similar they are to the target item and then selecting k most similar items using the k-means algorithm. Training ratings were set into the user-based CF engine.
- Prediction is computed by taking a weighted average on the target user's ratings on the most similar items.

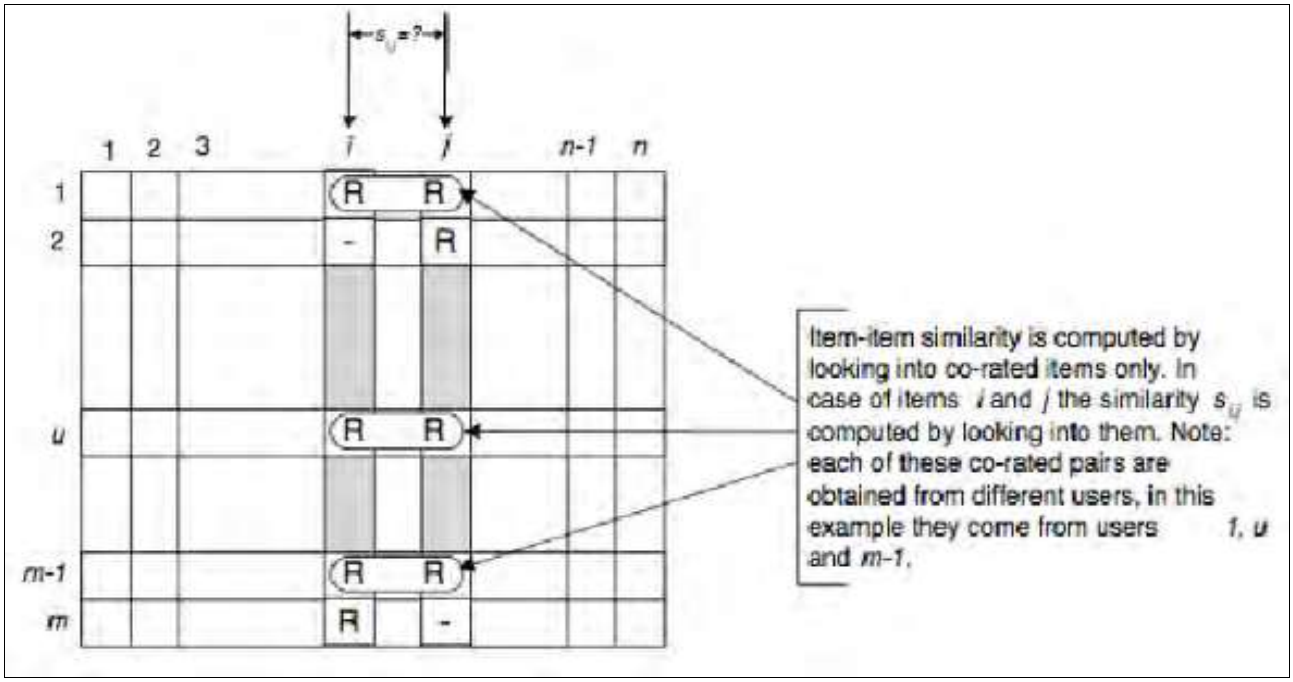

Fig 4: Isolation of Co-Rated Items and Similarity Computation

Novelty Learning

In psychology, boredom is typically viewed as an emotion characterized by a lack of stimulation or a feeling of dullness. Goetz *et al.* [11] discuss the external validity of boredom, noting that it can manifest with varying levels of arousal and emotional valence. However, because boredom often presents itself subtly within one's environment, it is challenging to detect through explicit signals such as item ratings or satisfaction scores. Nett *et al.* [12] investigated how individuals in academic settings cope with boredom, revealing various strategies people use to manage it. Without clear visual or observable indicators of a user's emotional state, recommender systems struggle to determine if a user is seeking novelty, often leading to users becoming bored more quickly when content doesn't align with their tastes.

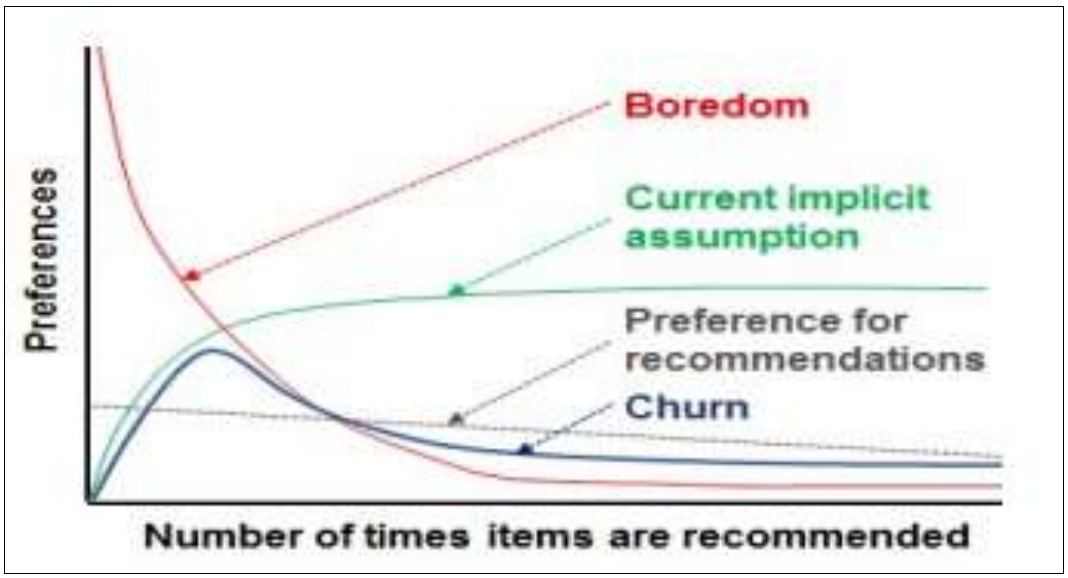

Fig 5: Devaluation of User Preferences for Recommended Items Implies That the Boredom Settles In and User Might Not Trust the System Any Longer

Figure 5 illustrates the decline in user preferences as recommendations are provided. Initially, users may respond positively to the recommended items, but as repetitive or uninteresting recommendations continue, they become bored and lose trust in the system. This can lead to user churn or disengagement. Unlike emotions such as anger or excitement, boredom is a quieter emotion, and novelty-seeking users may not give direct feedback or any observable signals. This makes it difficult for current recommender systems to accurately gauge users' preferences. Once a user becomes disengaged, regaining their trust is a significant challenge. Therefore, it is crucial to develop methods that proactively understand user behaviors. Our goal is to bridge this gap by introducing a system that can learn and respond to novelty-seeking behaviors through latent taste analysis.

Experiment

Containing 1,000,209 ratings on a 1-5 scale, MovieLens 1M involves 3,952 users and 6040 movies. Similar to MovieLens 100K, it includes demographic information and was gathered from the same platform in 2000, with users rating at least 20 movies.

Table 1: Description of the Experiment

Dateset	Users	Items	Ratings
MovieLens 100k	943	1682	100000
Each Movie	36656	1621	2579983
Movie Lens 1M	6040	3952	1000209
MovieLens-latest-small	718	8915	100234

All ratings are treated as positive feedback (+1), assuming that users like all movies they have rated. Below, we provide detailed descriptions of the four datasets mentioned, along with a comparison in terms of the number of users, movies, and ratings.

The task of recommendation algorithm concerns the prediction of target user's rating for target item that the user has not given the rating, based on the user's rating on observed items. Each user is represented by item-rating pairs, and can be summarised in a user-item table, which contains the ratings R ij that have been provided by the i th user for the j th item.

User	Item 1	Item 2	… …	Item n
User 1	R 11	R 12	… …	R 1n
User 2	R 21	R 22	… …	R 2n
User 3	… …	… …	… …	R 3n
User m	R m1	R m2	… …	R mn

Where R ij denotes the score of item j rated by an active user i. If user has not rated item j, then Rij= 0. The symbol 'm' denotes the total number of users, and n denotes the total number of items.

The 'k' means algorithm as the basic clustering algorithm. K is input to the algorithm that specifies the desired number of clusters. Firstly, the algorithm takes the first k items as the centers of k unique clusters. Each of the remaining items is then compared to the closest center. In the following passes, cluster centers are re-computed based on cluster centers formed in the previous pass and the cluster membership is then re-evaluated.

The clustering algorithm may generate fixed sized partitions or partitions of varying size. Once the item clustering algorithm is complete the performance can be very good, since the size of the group that must be analyzed is much smaller. When we cluster the items, we get cluster of the items centers. This center is represented as an average rating over all items in the cluster. So we choose the target item neighbors in some of the item center clustering. After calculating the similarity between the target item and the items centers, we take the items in the most similar centers.

After the completion of algorithm on our session data set, we find highest occurrence among all the items visited by the user under during different scenarios. There are some of the steps which are as follows in our complete process.

Step 1. Initialize the dataset of visited item.
Step 2. Define the number of clusters called k.
Step 3. Define the centers of dataset.
Step 4. Take the each sample in sequence and compute its similarity from the center of each from center of each cluster and group these k clusters according to the similarity from center of the cluster.
Step 5. Check for some exactly same common data in each group. If there is such value, reorganize them.
Step 6. Repeat the process from step 2 till all the items are correctly partitioned.
Step 7. Now use these clusters in the basic prediction algorithm.
Step 8. To predict novel items each cluster will fetch the items from neighbour cluster.
Step 9. Now perform the test to determine the occurrence of each page visited by the user.
Step 10. Now find the appearance of a grouping of two clusters like AB, AC.
Step 11. Perform the strength calculation between the associated values with the groups.

The similarity rating measurement of different items recommended for movies relies on the following equation where where R i,c is the rating of the item c by the user I. Ac is the average rating of the user I for all the co-rated items, and I ij is the items set both rating by user i and user j. The closest neighbour is selected, whose similarity exceeds a certain threshold value are considered as neighbours of target user. N-best neighbours is selected and n is given at first.

$$\text{sim}(i,j) = \frac{\sum_{celij}^{n}(\text{Ric} - \text{Ac})(\text{Rjc} - \text{Ac})}{\sqrt{\sum_{celij}(\text{Ric} - \text{Ac}) * \sum_{celij}(\text{Rjc} - \text{Ac})}}$$

Since, after each membership, weighted average of neighbours ratings can be calculated using weighted measure by their similarity to the target user. The following equation gives the target user u to the target item t is as follows:

$$Put = \frac{\sum_{i=1}^{c}(Rit - Ai) * sim(u,i)}{\sum_{i=1}^{c} sim(u,i)}$$

Where Au is the average rating of the target user u to the items, R it is the rating of theneighbour user I to the items, sim(u,i) is the similarity of the target user u and the neighbour user I, and c is the number of neighbours.

Evaluation metrices

To measure the quality of a recommender system a MAE matrix is proposed this matrix evaluates the deviation between previously predicted results and now predicted results Le the deviation of recommendations from their true specified user values.

$$MAE = \frac{1}{n}\sum_{i=1} |y_i - \widehat{y_i}|$$

where,

n: *number of observation*

y_i: *the actual value of the* i^{th} *observation*

$\widehat{y_i}$: *the predicted value of the* i^{th} *observation*

- The lower the MAE, the more accurately the recommendation engine predicts user ratings.
- MAE is the most commonly used and is the easiest to interpret.

S. No.	Item Bsd	Item Dft
1	0.7162	0.7125
2	0.7194	0.7155
3	0.7187	0.7162
4	0.7199	0.7155
5	0.7195	0.7157

Experimental Results

- Experimental steps- Division into train and test portions.
- Assessment of Quality of Recommendations- Determining the sensitivity of the neighborhood size, train test ratio on the effect of different similarity measures
- Using only the Training Data & further subdivision of it into a train and test portion
- The performance of collaborative filtering algorithms relies heavily on the number of available ratings in the dataset, as predictions and recommendations are based solely on this information. We also conducted significance tests to assess the statistical significance of performance differences among these algorithms.

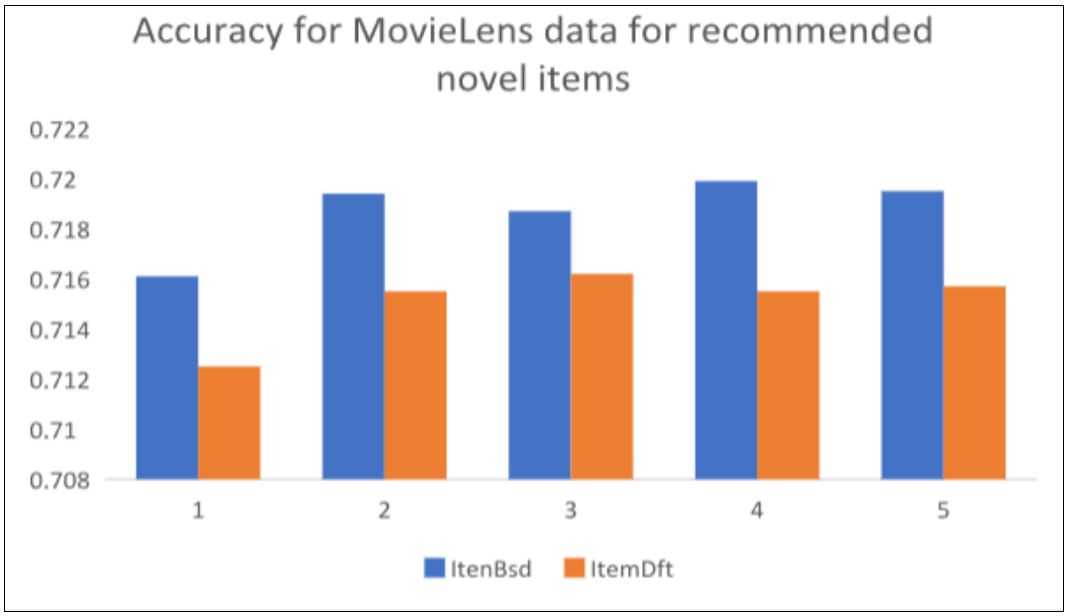

Fig 6: Histogram of Comparative Studies Between Item BSD and Item DFT

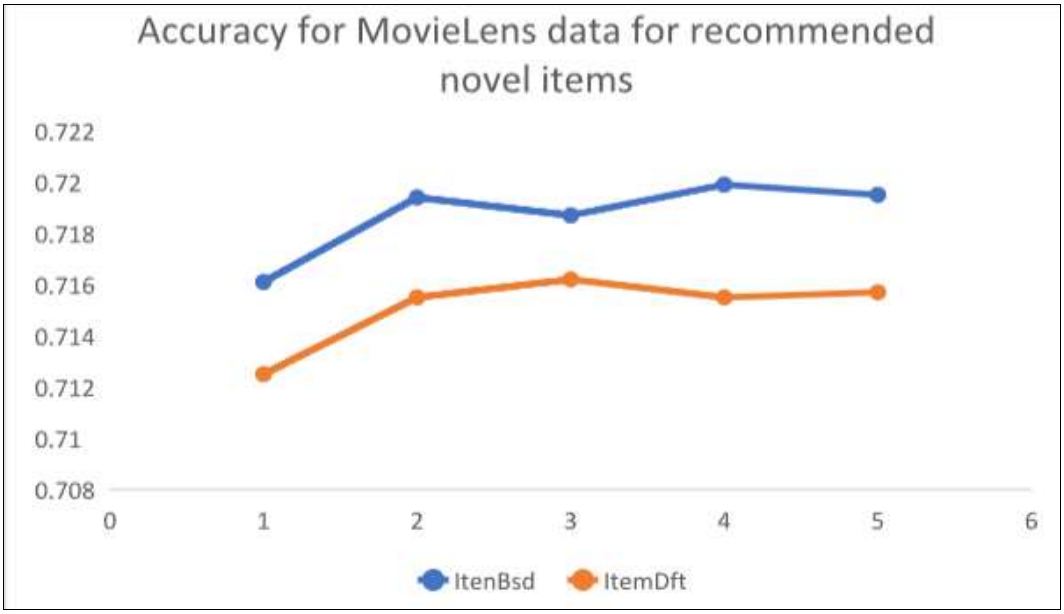

Fig 7: Plotted Graphs Shows Accuracy Increment for Novel Items

Neighbourhood-based algorithms, a subset of collaborative filtering algorithms, are simple, easy to understand, and yield reasonably accurate results. However, their performance is influenced by several factors, such as similarity measures, prediction techniques, datasets, and feedback types. Before discussing confidence estimation, we conducted an experimental study to evaluate neighbourhood-based algorithms with different similarity measures, prediction techniques, and datasets, including various types of feedback. We also conducted significance tests to assess the statistical significance of performance differences among these algorithms.

The performance of collaborative filtering algorithms relies heavily on the number of available ratings in the dataset, as predictions and recommendations are based solely on this information. Real-world datasets tend to be sparse, and additional factors such as noisy data, untrusted users, changing user preferences, algorithm parameter choices, and algorithm suitability contribute to the uncertainty of predictions and recommendations. Thus, measuring the reliability of these predictions and recommendations is crucial.

Conclusion

We aimed to enhance the efficiency of prediction regions produced by the proposed prediction algorithms by defining various nonconformity measures. Although these algorithms yield a sufficient number of single labels and correct predictions at 60%- 90% confidence levels. Thus, more efficient nonconformity measures are necessary for higher confidence levels in medical recommender systems.

References

1. Ricci F, Rokach L, Shapira B. Recommender systems: Introduction and challenges. In Recommender Systems Handbook; Springer: Boston, MA, USA; c2015.
2. Aggarwal CC. An introduction to recommender systems. In Recommender Systems; Springer: Cham, Switzerland; c2016.
3. Kardan AA, Ebrahimi M. A novel approach to hybrid recommendation systems based on association rules mining for content recommendation in asynchronous discussion groups. Information Sciences 2013;219:93-110.
4. Mustaqeem A, Anwar SM, Majid M. A modular cluster-based collaborative recommender system for cardiac patients. Artificial Intelligence in Medicine 2019;102.
5. Herlocker JL, Konstan JA, Terveen LG, Riedl JT. Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems 2004;22:5-53.
6. Lillegraven TN, Wolden AC. Design of a Bayesian recommender system for tourists presenting a solution to the cold-start user problem [master's thesis]. Norwegian University of Science and Technology; c2010.
7. Bobadilla J, Ortega F, Hernando A, Gutiérrez A. Recommender systems survey. Knowledge-Based Systems 2013;46:109-132.
8. Martins GB, Papa JP, Adeli H. Deep learning techniques for recommender systems based on collaborative filtering. Expert Systems with Applications.
9. Shambour Q, Lu J. An effective recommender system by unifying user and item trust information for B2B applications. Journal of Computer and System Sciences 2015;81(7):1110-1126.
10. Wang YY, Luse A, Townsend AM, Mennecke BE. Understanding the moderating roles of types of recommender systems and products on customer behavioral intention to use recommender systems. Information Systems and e-Business Management 2015;13:769-799.
11. Goetz T, Frenzel AC, Hall NC, Nett UE, Pekrun R, Lipnevich AA. Types of boredom: An experience sampling approach. Motivation and Emotion 2013;38(3):401-419.
12. Nett UE, Goetz T, Daniels LM. What to do when feeling bored?: Students' strategies for coping with boredom. Learning and Individual Differences 2010;20(6):626-638.
13. Ricci F, Rokach L, Shapira B. Recommender systems: Introduction and challenges. In Recommender Systems

Handbook; Springer: Boston, MA, USA; c2015.
14. Herlocker JL, Konstan JA, Terveen LG, Riedl JT. Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems 2004;22:5-53.
15. Smith B, Linden G. Two decades of recommender systems at Amazon.com. IEEE Internet Computing 2017;21:12-18.