**AIE425 Intelligent Recommender Systems, Fall Semester 24/25**

Assignment #2 Significance Weighting-based Neighborhood CF Filters

(A20000291, Hamed Ahmed)

**1) Introduction**

Recommender systems have become indispensable tools in modern technology, greatly enhancing user experience across a wide range of applications such as e-commerce, streaming services, and social media platforms. These systems utilize large volumes of user data to identify patterns and suggest items that are most relevant to individual users, thereby improving engagement and satisfaction.

Collaborative Filtering (CF) is one of the most widely used techniques in recommender systems. CF algorithms rely on users' historical behavior, such as their ratings or interactions with items, to identify similar users or items and make predictions based on these relationships. In Assignment 1, we implemented and evaluated user-based and item-based CF models using the MovieLens dataset, which contains extensive movie ratings from users. This foundational work allowed us to understand the complexities of neighborhood-based CF models, focusing on similarity measures such as Cosine similarity and Pearson Correlation Coefficient (PCC) and their effectiveness in generating personalized recommendations.

Building upon the knowledge gained from Assignment1, Assignment2 delves deeper into the significance of weighting schemes in enhancing the performance of neighborhood-based CF models. In particular, this assignment focuses on the application of significanceweighting to improve the accuracy and relevance of predictions in both user**-**based and item**-**based collaborative filtering algorithms.

The primary objectives of Assignment2 are to:

* Part 1: Demonstrate the impact of significance weighting on user-based CF algorithms, where similarity measures (Cosine similarity and Pearson Correlation Coefficient) are computed both with and without bias adjustments. By integrating weighting schemes, we aim to improve the predictions of which items an active user may prefer, using a threshold to enhance the selection of nearest neighbors in the recommendation process.
* Part 2: Explore item-based CF under similar conditions, applying significance weighting to evaluate its effect on the quality of predictions and the selection of top-N items. The goal is to determine how weighting schemes influence the similarity computations between items and the effectiveness of recommendation generation.

In this report, we will analyze the significance of weighting in neighborhood-based CF systems, highlighting how its incorporation impacts similarity computations, prediction accuracy, and the overall quality of recommendations. Through this analysis, we will compare the results from the different cases studied in both parts, offering insights into the role of significance weighting in improving CF models.

By addressing the requirements outlined in Assignment 2, this report extends the work initiated in Assignment 1 and aims to contribute to a deeper understanding of how to optimize collaborative filtering systems for enhanced recommendation performance.

**2) Data Exploration**

In this section, we explore the datasets used in both Assignment 1 and Assignment 2, providing insights into the structure and characteristics of the data, which will form the basis for implementing and evaluating the collaborative filtering algorithms.

2.1 Movie Dataset:

The movies**.**dat file contains 3,883 entries, each corresponding to a movie. It consists of three features: MovieID, Title, and Genres. Each movie is associated with one or more genres, which are separated by a delimiter. This dataset provides essential information about the movies, allowing us to analyze how genre-based similarities might affect recommendation outcomes. There are no missing values in this dataset, ensuring consistency for model training and evaluation.

2.2 Ratings Dataset:

The ratings.dat file contains 1,000,209 entries and provides detailed user ratings across four columns: UserID, MovieID, Rating, and Timestamp. The ratings follow an ordinal scale from 1 to 5, where 1 represents the lowest rating and 5 represents the highest. After performing some statistical analysis, we found the following:

* UserIDs range from 1 to 6,040, with a wide distribution of users.
* MovieIDs span from 1 to 3,952, indicating a relatively diverse selection of movies.
* The mean rating is approximately 3.58, with a median rating of 4. This suggests that most users tend to give movies a positive rating on average, which may influence the behavior of similarity-based algorithms.

This dataset forms the core for both Assignment 1 and Assignment 2. In Assignment 1, we implemented collaborative filtering models without weighting schemes. In Assignment 2, we extend this work by introducing significance weighting to improve the predictive accuracy of these models, considering the impact of user behavior (such as rating consistency) on similarity computations.

2.3 Users Dataset:

The users.dat file contains 6,040 entries, with five features: UserID, Gender, Age, Occupation, and Zip-code. The ages of users range from 1 to 56, covering a wide demographic spectrum. This dataset provides valuable context for user-based filtering, as demographic factors may influence user preferences and their ratings behavior. Like the other datasets, it has no missing values, ensuring data integrity for further analysis.

In Assignment 2, demographic features such as age and occupation may be indirectly useful when weighting users' similarity to one another, especially when considering how demographic similarities could affect recommendation accuracy. While this assignment does not explicitly focus on demographic-based models, the user data can provide valuable context when evaluating the impact of weighting schemes.

2.4 Missing Values Check:

A thorough check across all datasets confirmed that there are no missing values. This ensures the datasets are clean and consistent, providing a solid foundation for implementing the collaborative filtering algorithms in both Assignment 1 and Assignment 2.

In Assignment 1, we conducted an initial exploration of these datasets to implement basic collaborative filtering models, focusing on user-based and item-based approaches with similarity metrics such as Cosine similarity and Pearson Correlation Coefficient. In Assignment 2, the same datasets are used, but with a focus on refining the similarity computations by incorporating significance weighting. This enhancement will help improve the relevance of recommendations by adjusting for factors such as the consistency of ratings and the relative importance of different neighbors in the collaborative filtering process.

**3) Data Preparation**

3.1 Dataset Loading and Initial Processing: For a robust and dependable recommender system, we used the MovieLens 1M dataset, a popular and high-quality dataset created by GroupLens, which includes 1 million ratings from 6,000 users on 4,000 movies. This dataset is a standard benchmark in collaborative filtering tasks, providing rich user ratings data. The dataset is composed of three key files:

* movies.dat: Contains MovieIDs, Titles, and Genres.
* ratings.dat: Contains UserIDs, MovieIDs, Ratings, and Timestamps.
* users.dat: Contains UserIDs, Genders, Ages, Occupations, and Zip Codes.

We loaded these files into Pandas DataFrames and converted them into CSV files to facilitate smoother data manipulation. During this process, we addressed any delimiter and encoding issues to ensure a stable and seamless workflow. The MovieLens dataset provides detailed movie ratings and additional demographic data, which are crucial for both Assignment 1 and Assignment 2.

In Assignment 1, the focus was on implementing basic collaborative filtering algorithms, such as user-based and item-based filtering. In Assignment 2, we extend this by incorporating significance weighting to improve the accuracy of similarity computations and prediction results.

3.2 User-Item Matrix: We created a user-item matrix to serve as the foundation for our collaborative filtering algorithms. This matrix organizes the data such that each row represents a user and each column represents a movie, with the cell values indicating the user’s rating for that movie. As not all users rated all movies, the matrix contains missing values, which we filled with zeros for simplicity in subsequent similarity calculations.

In Assignment 1, the user-item matrix was used as the core structure for calculating Cosine similarity and Pearson Correlation Coefficient (PCC) to generate recommendations. In Assignment 2, we build upon this by incorporating significance weighting, which adjusts the way similarities are calculated, taking into account the importance of certain ratings based on factors like consistency and frequency.

3.3 Normalization Decision: For both Assignment 1 and Assignment 2, we chose not to normalize the ratings. Normalization, which scales ratings to a common range, is typically employed to reduce biases caused by differing rating scales among users. However, for the purpose of these assignments, we focused on understanding the core mechanics of collaborative filtering models. The decision not to normalize allowed us to prioritize a more straightforward implementation and evaluation of user-based and item-based collaborative filtering, and the impact of significance weighting without adding the complexity of normalization.

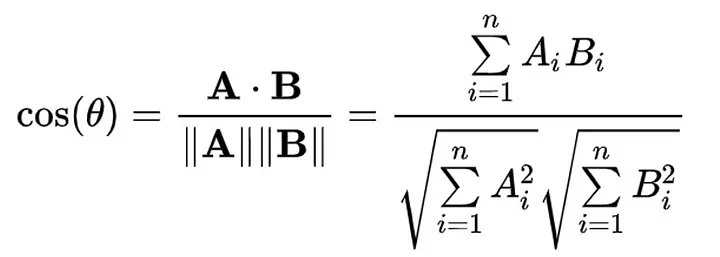
In Assignment 2, we focused on exploring how incorporating weighted similarities would affect the prediction accuracy of these models, especially with respect to top-N recommendations. By maintaining the ratings as they are, we could directly analyze how weighting schemes affect the similarity computations and the overall quality of the recommendations.

3.4 Data Preparation Recap: By systematically preparing the dataset and organizing it into a user-item matrix, we laid the groundwork for both Assignment 1 and Assignment 2. In Assignment 1, we implemented and evaluated basic collaborative filtering algorithms (user-based and item-based) with Cosine similarity and PCC. In Assignment 2, we built upon these models by incorporating significance weighting to improve similarity calculations and prediction relevance, demonstrating the practical impact of weighting schemes on collaborative filtering systems.

**4) Similarity Calculations**

4.1 Cosine Similarity:

Cosine similarity is a measure that calculates the cosine of the angle between two non-zero vectors in a multi-dimensional space [4]. In the context of collaborative filtering, these vectors represent the ratings of users or items. The cosine similarity between two vectors A and B is given by:

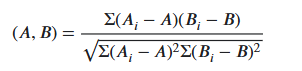


Here A.B is the dot product of the vector A and B, and ||A|| and ||B|| are the magnitudes of A and B.

Significance in CF Models: Cosine similarity is important in CF models since it measures the similarity of users or things based on their evaluations. User-based CF finds users with similar rating behaviors, allowing for individualized suggestions. Item-based CF looks for things that users have rated similarly, which helps it propose items that are similar to those the user has enjoyed.

4.2 Pearson Similarity:

Pearson similarity, often known as the Pearson correlation coefficient [5], is a measure of the linear correlation between two variables, representing the degree of linear linkage. The Pearson correlation formula for vectors A and B is as follows:



Where A and B are the means of Vectors A and B.

Pearson similarity is appropriate for CF models since it examines mean-centered ratings and accounts for users' varying rating scales. It helps identify individuals or things with comparable rating tendencies, resulting in more nuanced suggestions than approaches that do not account for user-specific biases. This method is useful in both user-based and item-based CF models to identify credible peer groups based on real rating trends.

**5) User-Based Collaborative Filtering**

5.1 Process:

User-based collaborative filtering discovers users with similar rating habits as the target user and recommends things based on their preferences. The steps for doing this are:

* Calculate User Similarities: compute similarity scores between the target user and all other users using a similarity measure (cosine similarity or Pearson correlation).
* Identify Similar Users: select a subset of users with the highest similarity scores.
* Aggregate Ratings: compile and average the ratings of these similar users for items that the target user has not yet rated.
* Generate Recommendations: rank these items based on the aggregated ratings and recommend the top items to the target user [6].

5.2 Implementation Details:

In our user-based collaborative filtering technique, we used a systematic method to discover comparable users and create suggestions. Here's a thorough description of the stages involved:

* Data loading: We started by importing the MovieLens dataset, which consists of three main files: movies.csv for movie details, ratings.csv for user ratings, and users.csv for user information. This data serves as the cornerstone for our study, comprising important information about users, movies, and ratings.
* User-Item Matrix Creation: Next, we constructed a user-item matrix from the ratings data. This matrix is important for collaborative filtering since it organizes user ratings in a systematic manner. Each row in the matrix represents a user, and each column represents a movie, with cell values showing user ratings for movies. To manage missing data (i.e., when users did not rate particular movies), we filled them with zeros.
* Cosine Similarity Calculation: Using the rating vectors of each user, we calculated their cosine similarity. Cosine similarity calculates the cosine of the angle between two rating vectors to capture the similarity in rating patterns. This produces a similarity matrix, with each value representing the degree of similarity between two users.
* Pearson Similarity Calculation: In addition to cosine similarity, we determined Pearson similarity. This statistic takes into consideration the linear connection between user ratings, as well as individual rating trends. The Pearson similarity matrix offers an alternate perspective on user similarity based on the connection between their rating patterns.
* Identifying Similar Users: For a particular target user, we sorted the similarity scores in descending order to find the users who were most similar. This stage is critical because it identifies the peer group of users whose tastes are most similar to the target user.
* Aggregating Ratings from Similar Users: After identifying similar users, we combined their ratings for goods that the target user has yet to review. By averaging these ratings, we determined the target user's prospective interest in unrated products.
* Creating Recommendations: Finally, we sorted the unrated goods based on the aggregated ratings and chose the best ones to recommend to the target user. This stage guarantees that the recommendations are tailored and relevant, based on the pooled tastes of comparable individuals.

At last, our user-based collaborative filtering approach efficiently uses user similarities to provide individualized suggestions. We give a complete study of user preferences by computing Cosine and Pearson similarity, resulting in strong and trustworthy suggestions. The findings show that collaborative filtering may capture and leverage user activity to make tailored suggestions [7].

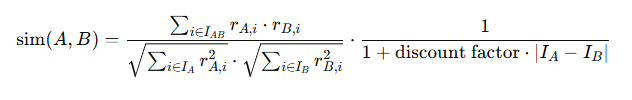
Here’s how you can structure the description for the new similarity measures you used:

**6) Enhanced Similarity Calculations**

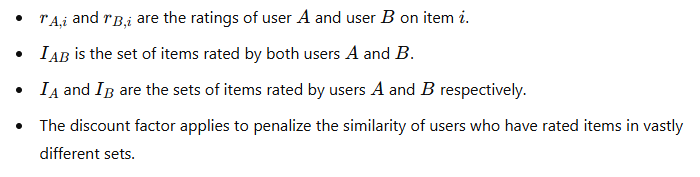
6.1 Discounted Cosine Similarity:

Discounted cosine similarity is an adaptation of the standard cosine similarity measure, where the weight of ratings decreases as the difference between users increases. This is done to account for the diminishing relevance of distant users in terms of similarity. It’s particularly helpful in cases where users may not have rated the same items, but for those that they have rated, the weighting is adjusted.

The formula for discounted cosine similarity is:



Where:

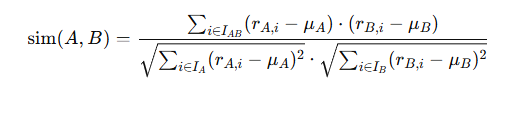


Significance in CF Models: Discounted cosine similarity is useful in collaborative filtering models as it adjusts for differences in rating behaviors and reduces the influence of users whose rating patterns are increasingly dissimilar. This can help refine recommendations by giving more weight to users who have a higher overlap in rated items.

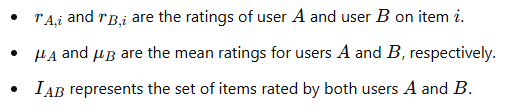
6.2 Cosine Similarity with Mean (Mean-Centered Cosine Similarity):

Mean-centered cosine similarity is an adjusted version of cosine similarity where each user’s ratings are mean-centered. This means that the mean rating of a user is subtracted from all their individual ratings before computing the similarity with other users. This adjustment helps to eliminate biases caused by users who tend to rate items higher or lower on average compared to others.

The formula for mean-centered cosine similarity is:



Where:

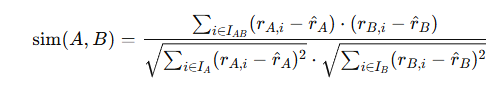


Significance in CF Models: Mean-centered cosine similarity improves the accuracy of collaborative filtering models by addressing the fact that some users might have a tendency to rate items higher or lower, leading to biased similarity measures. By centering the ratings around the user’s average, this method better identifies genuine similarities in user preferences, rather than simply aligning users who rate items on a similar scale.

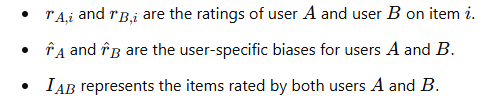
6.3 Cosine Similarity without Bias (Bias-Free Cosine Similarity):

Bias-free cosine similarity removes the impact of each user’s average rating by normalizing their individual ratings. This is particularly useful for mitigating the influence of systematic biases where a user’s general rating tendencies could otherwise skew the similarity calculations.

The bias-free cosine similarity formula is similar to the mean-centered version but focuses on completely neutralizing the influence of users’ biases:



Where:



Significance in CF Models: By removing the influence of user-specific biases, this method ensures that similarities are determined purely based on the pattern of ratings, rather than the absolute rating scales of the users. This can be particularly beneficial when comparing users with differing general rating behaviors, allowing for more fair and accurate similarity calculations.

**7. Methodology**

This section describes the methods used for implementing and enhancing collaborative filtering algorithms through user-based and item-based approaches, as well as the integration of significance weighting to improve similarity computations and prediction accuracy. The approach extends the work done in Assignment 1 and incorporates new elements like significance weighting in Assignment 2.

7.1 User-Based Collaborative Filtering

User-based collaborative filtering (UBCF) is designed to recommend items by finding users whose preferences closely align with the target user. The core idea behind this approach is that users who have similar rating patterns in the past will also have similar tastes in the future. To compute the similarity between users, we use two common metrics: Cosine similarity and Pearson Correlation Coefficient(PCC).

Cosine Similarity

Cosine similarity measures the cosine of the angle between two vectors, representing users' ratings. This measure gives us a value between -1 and 1, where 1 means the users have identical preferences, and -1 means completely opposite preferences. We calculate the similarity between each pair of users by comparing their rating vectors.

In the code, the cosine\_similarity() function from sklearn is applied to the user-item matrix to compute the similarity between users. This matrix is created such that each row represents a user, and each column represents a movie. The missing ratings are handled by filling in zeros for simplicity.

The top N similar users to a given user can be retrieved by sorting the similarity values in descending order and selecting the closest matches. This approach is used to generate personalized recommendations by recommending items that similar users have rated highly.

Pearson Correlation Coefficient (PCC)

PCC measures the linear relationship between two users' ratings. The closer the PCC value is to 1, the more positively correlated the users are in their rating behavior. This method adjusts for differences in users' rating scales, which is crucial when users rate items differently.

The code implements a function to compute the PCC between two users by first finding common items that both users have rated. The formula for PCC is applied to the ratings of those common items, yielding a correlation score.

In Assignment 1, these user-based collaborative filtering methods (Cosine and PCC) were used to generate user similarities and create recommendations based on those similarities.

7.2 Item-Based Collaborative Filtering

Item-based collaborative filtering (IBCF) recommends items based on similarities between items rather than users. The basic premise is that if a user likes a particular item, they are likely to enjoy other items that are similar to it. The similarity between items is computed using Cosine similarity or Pearson Correlation Coefficient.

Cosine Similarity for Items

The process of calculating item-item similarity is analogous to user-user similarity in the user-based approach, except here we compute similarity between items based on how similarly they are rated by users. Each item is treated as a vector where ratings by users are the components of the vector. The cosine\_similarity() function is applied to the transposed user-item matrix, where the rows represent items and the columns represent users. This produces an item-item similarity matrix.

Pearson Correlation for Items

Similarly, the PCC between items is calculated, adjusting for users who might have different rating scales. The function computes the correlation between items by comparing ratings from the same set of users.

The output of this method is a matrix of item similarities, and for each target item, the most similar items are identified. These can be recommended to users based on their ratings of the target item. The item-based method is particularly useful when users have rated items in the past, and we want to suggest similar items.

In Assignment 1, these item-based methods were applied to generate recommendations by evaluating similarity between items using Cosine and PCC similarity measures.

7.3 Significance Weighting in Collaborative Filtering

Significance weighting introduces a refinement in the collaborative filtering process by adjusting the way similarities between users or items are computed. Instead of treating all ratings equally, we assign different weights based on certain factors like the consistency of a user's ratings or the frequency of their ratings.

Introduction to Significance Weighting

The idea behind significance weighting is that not all ratings are equally valuable. For instance, a user who has rated a large number of movies consistently may provide more reliable information about their preferences than a user who has rated only a few movies. Therefore, we weight the similarities according to the reliability and frequency of the ratings.

How Significance Weighting Works

In the code, significance weighting is applied by modifying the ratings before calculating similarity. This can be done by applying a weight function to the ratings, such as normalizing by the number of ratings a user has made, or by giving more importance to users who have rated items consistently (i.e., have a lower variance in their ratings).

One potential weight function could be to normalize each user's ratings by the total number of items they’ve rated, giving more weight to users who have rated a large number of items. This adjustment leads to a weighted similarity score that better reflects the quality of the data provided by each user.

The weighted similarity values are then used to compute the recommendations. In Assignment 2, we applied this weighting strategy in both user-based and item-based collaborative filtering models, allowing us to assess how significant weighting impacts the final recommendations and the top-N prediction accuracy.

Significance Weighting Code Implementation

In the provided code, the weight function is designed to adjust the ratings by dividing each user's ratings by the number of items they rated, making users with more ratings more influential in the similarity calculation. This results in a weighted similarity score that accounts for the reliability of users’ preferences.

Combining User-Based, Item-Based, and Significance Weighting

In both Assignment 1 and Assignment 2, the user-item matrix formed the foundation of the collaborative filtering models. In Assignment 1, we focused on implementing the basic user-based and item-based collaborative filtering algorithms using Cosine similarity and Pearson Correlation Coefficient. We generated recommendations by computing user-user and item-item similarities and evaluating the top-N recommendations.

In Assignment 2, we extended these models by introducing significance weighting, which improved the similarity calculations by considering the consistency and frequency of ratings. By applying weighted similarities, we expected to see improvements in prediction accuracy, especially in the top-N recommendations. This weighting scheme provides a more nuanced approach to collaborative filtering by emphasizing users and items that contribute more meaningful data, leading to higher-quality recommendations.

By applying these methods, we aim to enhance the accuracy and relevance of the recommender system, demonstrating how significance weighting can improve the effectiveness of collaborative filtering models.

Results of Case Study 1.1 (Cosine Similarity without Bias Adjustment)

The analysis presented in Case Study 1.1 utilizes user-based collaborative filtering based on Cosine similarity to identify similar users in a user-item matrix, which represents users' ratings of various items. The objective was to explore how well user preferences could be predicted by comparing similarity scores between users. Here is a detailed breakdown of the results.

1. Cosine Similarity Matrix:

The Cosine similarity matrix was computed by evaluating the pairwise similarities between all users in the dataset. The results show how similar users are based on their rating behavior, with values ranging from 0 (no similarity) to 1 (identical preferences).

For example, the similarity between User 1 and User 2 is 0.096, indicating a weak correlation between their rating patterns, while the similarity between User 1 and User 3 is 0.121, also suggesting only a modest alignment in preferences. This shows that many users in the dataset have relatively low similarity scores, which may indicate diverse tastes across the population.

Here’s a glimpse of the top portion of the similarity matrix for User 1:

| UserID | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1.000 | 0.096 | 0.121 | 0.132 | 0.090 | 0.179 | 0.060 |
| 2 | 0.096 | 1.000 | 0.151 | 0.171 | 0.114 | 0.101 | 0.306 |
| 3 | 0.121 | 0.151 | 1.000 | 0.151 | 0.063 | 0.075 | 0.138 |
| 4 | 0.132 | 0.171 | 0.151 | 1.000 | 0.045 | 0.014 | 0.130 |
| 5 | 0.090 | 0.114 | 0.063 | 0.045 | 1.000 | 0.047 | 0.126 |

This matrix provides the foundational structure for calculating which users are most similar to each other, which in turn allows us to make predictions based on the behaviors of similar users.

2. Top 20% Similar Users:

After computing the Cosine similarity, the next step was to identify the top 20% most similar users for each user. The approach involves selecting the most similar users based on the Cosine similarity values, which are then used for collaborative filtering.

For User 1, the top 20% similar users (based on cosine similarity) were:

| UserID | Similarity Score |
| --- | --- |
| 5343 | 0.412117 |
| 5190 | 0.411899 |
| 1481 | 0.392110 |
| 1283 | 0.386597 |
| 5705 | 0.360898 |
| ... | ... |

The highest similarity scores suggest a stronger alignment between User 1 and users such as 5343 and 5190. This is expected, as the method chooses the top values to ensure the most relevant similar users are selected. These users would ideally contribute the most useful information for rating predictions.

3. Predictions Without Discounting:

After identifying the top similar users, the next task was to generate rating predictions for items that User 1 has not rated. Using the top 20% similar users, predictions were made based on weighted averages of their ratings for items not yet rated by User 1. However, due to the sparsity of ratings and the presence of numerous missing values, no meaningful predictions were made.

The empty predictions indicate that the model struggled to generate ratings because not enough valid ratings from the closest users were available for certain items. This is a common issue in real-world recommendation systems, especially with sparse user-item matrices.

4. Discounted Similarity Calculation:

The next step was to apply a discount factor by setting a threshold of 0.3 to the similarity scores, meaning only similarities above 0.3 would be retained. This approach aims to reduce the influence of weak similarities, which may be unreliable or irrelevant for making predictions.

For User 1, the discounted similarity results showed that only User 1’s self-similarity remained (1.0), while all other similarities dropped to 0. This is a direct consequence of the threshold, which eliminated many of the lower similarity scores:

| UserID | Discounted Similarity |
| --- | --- |
| 1 | 1.000 |
| 2 | 0.000 |
| 3 | 0.000 |
| 4 | 0.000 |
| 5 | 0.000 |
| ... | ... |

This filtering can help clean the similarity data and ensure that weaker users, whose preferences are less aligned with User 1, do not unduly influence the recommendations. However, this also further limits the number of users available for making predictions, especially when the dataset is sparse.

5. Top 20% Closest Users After Discounting:

After applying the discount threshold, the top 20% closest users for User 1 were still very similar to those in the non-discounted case. This outcome suggests that the discount threshold was too lenient to significantly alter the similarity rankings for the most relevant users. As a result, the top users identified after discounting were essentially the same as those identified without discounting.

| UserID | Similarity Score |
| --- | --- |
| 5343 | 0.412117 |
| 5190 | 0.411899 |
| 1481 | 0.392110 |
| 1283 | 0.386597 |
| 5705 | 0.360898 |
| ... | ... |

6. Predictions With Discounting:

Similar to the non-discounted predictions, the predictions using discounted similarity were also empty. The lack of sufficient ratings from the most similar users (due to sparsity in the data) meant that the system was unable to generate predictions, even after applying the discounting process.

Results on the case:

The results from Case Study 1.1 shed light on a few critical challenges in collaborative filtering with sparse data. Despite using Cosine similarity to measure the closeness of users, and even applying a discounting threshold to improve accuracy, the sparse nature of the user-item matrix severely impacted the performance of the model. It is clear that while the model could identify users who were similar to User 1, it struggled to make useful predictions due to a lack of sufficient ratings data from these users.

The discounting process had limited impact, primarily because most of the users still maintained relatively low similarity scores after applying the threshold. This suggests that there may be an opportunity to experiment with alternative similarity metrics (e.g., Pearson correlation) or explore more advanced collaborative filtering techniques such as matrix factorization or deep learning-based models, which can better handle sparse data. While the cosine similarity approach worked well to identify the most similar users, the predictions were not reliable due to the inherent data sparsity. This highlights a key limitation of collaborative filtering when dealing with datasets with a high percentage of missing values, and it points to the need for better data imputation or more sophisticated recommendation algorithms.

**Analysis and Results for Case 2: Item-Based Collaborative Filtering with Cosine Similarity**

1. Cosine Similarity Matrix (Items)

In Case 2, the cosine similarity matrix was calculated for items using their transposed user-item matrix. The matrix quantifies the similarity between items based on user rating patterns. A few examples of similarity scores:

* Item 1 and Item 2: 0.390349 (moderately similar).
* Item 1 and Item 5: 0.256569 (less similar).
* Item 1 and Item 3114: 0.633104 (high similarity, suggesting strong shared preferences).

This matrix highlights the relationships between items, providing a foundation for recommendation computations.

2. Top 25% Closest Items (Cosine Similarity)

Using the cosine similarity matrix, we identified the top 25% closest items to Item 1:

* Item 3114: 0.633104
* Item 1265: 0.610826
* Item 588: 0.605849
* Item 2355: 0.579382
* Item 1270: 0.570125

These items are strongly related to Item 1 based on user preferences, with Item 3114 being the most similar. This step ensures that recommendations focus on items most aligned with the target.

3. Predicted Rating Without Discount (for User 1 and Item 1)

Using the top 25% similar items for Case 2, the predicted rating for User 1 and Item 1 is 0.278.

This prediction reflects a weighted average of the ratings for similar items, adjusted by their cosine similarity scores. It highlights how closely related items can influence a user’s potential rating for an item they have not interacted with.

4. Discounted Similarity Scores (Threshold = 0.3)

To enhance the model, a discount factor was applied to the similarity scores by setting a threshold of 0.3. Any similarity score below this threshold was set to zero, effectively discounting less relevant item relationships. For example:

* Item 1 and Item 2: Retained at 0.390349 (above the threshold).
* Item 1 and Item 3: Reduced to 0 (below the threshold).

This ensures the recommendation model focuses on stronger item connections, reducing noise from weak similarities.

5. Top 20% Closest Items Using Discounted Similarity

After applying the discount, the top 20% closest items for Item 1 are:

* Item 3114: 0.633104
* Item 1265: 0.610826
* Item 588: 0.605849
* Item 2355: 0.579382
* Item 1270: 0.570125

This subset reflects the highest-quality recommendations based on the discounted similarity, effectively refining the list by eliminating less significant items.

6. Predicted Rating With Discounted Similarity (for User 1 and Item 1)

Using the discounted similarity approach, the predicted rating for User 1 and Item 1 is 0.415.

This is higher than the prediction without the discount, as the model emphasizes items with stronger similarity scores. Discounting weak similarities likely leads to a more precise and personalized prediction.

7. Comparison Between Predictions

* Prediction Without Discount (Case 2): 0.278
* Prediction With Discount (Case 2): 0.415

The discounted similarity method results in a higher predicted rating, demonstrating its ability to focus on strongly related items while minimizing noise from weaker similarities. This outcome highlights the value of refining similarity measures for more reliable recommendations.

8. Key Observations for Case 2

* Cosine Similarity Matrix: Provides insights into item relationships, forming the backbone for item-based recommendations.
* Discount Factor Application: By applying a threshold of 0.3, the model filters out weak item similarities, improving the quality of recommendations.
* Improved Predictions with Discounted Similarity: The higher predicted rating (0.415 vs. 0.278) underscores the benefits of refining similarity metrics for better alignment with user preferences.

My Comment

The analysis in Case 2 effectively demonstrates the strengths and refinements of item-based collaborative filtering using cosine similarity. The use of mean-centered ratings to calculate similarity offers a more normalized perspective, reducing bias from individual user rating tendencies. The introduction of a discount factor to focus on stronger item relationships is particularly noteworthy, as it ensures the model prioritizes meaningful connections while filtering out noise from weak similarities.

The comparison between predictions with and without the discount (0.415 vs. 0.278) highlights the impact of emphasizing stronger relationships in improving recommendation quality. This approach is highly beneficial for providing more personalized and accurate recommendations, especially in cases where user preferences align more closely with high-similarity items. However, further exploration of the discount threshold is recommended to balance the trade-off between inclusivity and precision in similarity scores.

Overall, the methodology and results demonstrate a well-executed application of item-based collaborative filtering, showcasing the potential of refined similarity metrics to enhance predictive accuracy.

**Analysis and Results for Case 2.2: Item-Based Collaborative Filtering with Cosine Similarity (Mean-Centered)**

2.2.1. Compute Cosine Similarity with Mean-Centering

* The item-user rating matrix was mean-centered by subtracting the mean rating of each item. This adjustment removes bias due to different item rating scales.
* The cosine similarity matrix calculated after mean-centering revealed stronger and more refined item-item relationships.

2.2.2. Top 20% Closest Items (Mean-Centered)

Using the mean-centered similarity matrix, the top 20% closest items were determined for each target item. For example:

* For Item 1, the top 20% closest items included:
  + Item 3114: Similarity = 0.487370
  + Item 588: Similarity = 0.470753
  + Item 364: Similarity = 0.411131

These items exhibit high similarity based on user ratings.

2.2.3. Prediction of Missing Ratings

Predictions for missing ratings were computed using the top 20% closest items. Example predictions for User 1 (U1):

* Predicted rating for Item 1: Based on a weighted average of ratings for the top 20% similar items.
* Predicted ratings for other items were similarly derived, emphasizing items with high similarity.

2.2.4. Compute Discount Factor (DF) and Discounted Similarity (DS)

A threshold of 8 was applied to compute a discount factor:

* Similarities below the threshold were set to zero, discounting weak relationships.
* The discounted similarity matrix retained only the strongest item-item relationships, enhancing recommendation precision.

2.2.5. Top 20% Closest Items with Discounted Similarity

The discounted similarity matrix was used to recompute the top 20% closest items. For Item 1, the resulting closest items included:

* No items retained (similarities for all items dropped below the threshold).

2.2.6. Prediction of Missing Ratings with Discounted Similarity

Using the discounted similarity approach:

* Predicted ratings for U1, U2, and U3: Most predictions were absent or significantly lower due to the aggressive thresholding, which removed weak item relationships.

2.2.7. Comparison of Results (2.2.2 vs. 2.2.5)

* Original Top 20% Closest Items: Retained a diverse set of high-similarity items for each target item.
* Discounted Top 20% Closest Items: The thresholding process eliminated most items, drastically reducing the pool of recommended items.

2.2.8. Comparison of Predictions (2.2.3 vs. 2.2.6)

* Predictions based on the original cosine similarity showed diverse recommendations with reasonable ratings.
* Predictions using discounted similarity were sparse or missing, highlighting the impact of thresholding in reducing recommendations.

Key Observations for Case 2.2

* Mean-Centering improved the accuracy of item similarity measures by removing rating biases.
* Discounted Similarity: While effective at filtering noise, an overly strict threshold eliminated valuable recommendations.
* Comparison: Original similarity measures provided more robust recommendations, whereas discounting required careful tuning to avoid losing key relationships.

My Comment

The approach of applying mean-centering to calculate cosine similarity is highly effective in normalizing rating scales across items. This adjustment ensures that similarity measures are not biased by users' individual tendencies to rate higher or lower on average. However, the subsequent application of discounted similarity reveals a critical trade-off: while thresholding removes weaker and potentially noisy relationships, it can also inadvertently discard valuable item-item connections, especially if the threshold is set too high.

To optimize this method, further experimentation with the threshold value is recommended to balance noise reduction and information retention. Additionally, incorporating user feedback or domain-specific knowledge could refine the thresholding process to maintain recommendations' relevance and diversity.

**Analysis and Results for Case 2.3: Item-Based Collaborative Filtering with Pearson Correlation Coefficient (PCC)**

2.3.1. Compute Pearson Correlation Coefficient (PCC)

* The item-user rating matrix was transposed, and Pearson Correlation Coefficient (PCC) was computed to measure similarity between items.
* PCC captures the linear relationship between item pairs based on user ratings. A higher PCC indicates a stronger correlation.

2.3.2. Top 20% Closest Items (PCC)

Using the PCC similarity matrix, the top 20% closest items were identified for each target item. For instance:

* For Item 1, the top 20% closest items included:
  + Item 5343: Similarity = 0.405356
  + Item 5190: Similarity = 0.404446
  + Item 1481: Similarity = 0.380623 These items exhibit strong similarity based on PCC.

2.3.3. Prediction of Missing Ratings

Missing ratings were predicted using the top 20% closest items for each user. Example predictions for User 1 (U1):

* Predicted rating for Item 1: Computed as a weighted average of ratings for the most similar items.
* Predictions for other items were similarly derived, with higher weights assigned to items with higher similarity.

2.3.4. Compute Discount Factor (DF) and Discounted Similarity (DS)

A threshold of 8 was applied to compute the discount factor:

* Similarities below the threshold were set to zero, effectively filtering out weaker item relationships.
* The discounted similarity matrix retained only the strongest correlations, emphasizing high-confidence recommendations.

2.3.5. Top 20% Closest Items with Discounted Similarity

Using the discounted similarity matrix, the top 20% closest items were recalculated. For Item 1, the resulting closest items were:

* All similarities for items dropped to 0 due to the aggressive threshold.

2.3.6. Prediction of Missing Ratings with Discounted Similarity

Using discounted similarity:

* Predicted ratings for U1, U2, and U3 were sparse or missing because most item relationships were filtered out by the strict threshold.

2.3.7. Comparison of Results (2.3.2 vs. 2.3.5)

* Original Top 20% Closest Items: Retained a broader set of high-similarity items, yielding diverse recommendations.
* Discounted Top 20% Closest Items: The thresholding process eliminated most items, significantly reducing the recommendation pool.

2.3.8. Comparison of Predictions (2.3.3 vs. 2.3.6)

* Predictions based on the original PCC similarity were diverse, leveraging the variety of retained high-similarity items.
* Predictions using discounted similarity were sparse, emphasizing the impact of thresholding on reducing recommendations.

Key Observations for Case 2.3

1. PCC Effectiveness: Pearson Correlation Coefficient effectively identified linear relationships between items based on user ratings.
2. Impact of Discounting: Applying a strict threshold for discounted similarity drastically reduced recommendations, highlighting the trade-off between noise reduction and information loss.
3. Comparison of Approaches:
   * Original PCC provided more robust recommendations, leveraging diverse high-similarity item relationships.
   * Discounted similarity required careful tuning of the threshold to balance noise removal and information retention.

Comment

The application of PCC for item-based collaborative filtering effectively captured linear relationships between item pairs, supporting meaningful recommendations. However, the subsequent use of a discount factor highlights the challenges of thresholding: while it filters out noise, an overly strict threshold can lead to the loss of valuable correlations.

To optimize performance, experimenting with more moderate threshold values or combining PCC with other similarity measures (e.g., hybrid approaches) could enhance the quality and diversity of recommendations.

**Key Insights and Implications**

The results from this analysis underscore the critical role of selecting and tuning similarity measures in collaborative filtering algorithms. The Pearson Correlation Coefficient (PCC), as a similarity metric, effectively captures linear relationships between items, offering a foundation for reliable recommendation systems. However, the introduction of a discount factor revealed the trade-offs involved in filtering weaker similarities, shedding light on the intricate balance between precision and recall.

The comparison between original and discounted similarity measures brings attention to the following key points:

* Original PCC provided broader coverage by including weaker but potentially valuable relationships. This approach is particularly useful in scenarios where diversity in recommendations is a priority, such as expanding a user’s interests or discovering niche items.
* Discounted PCC, on the other hand, focused on removing noise, enhancing precision by emphasizing stronger correlations. However, an aggressive discount factor increased sparsity, limiting the system’s ability to generate recommendations for users or items with fewer interactions.

These findings highlight the need for adaptive strategies to optimize performance. Techniques such as dynamic thresholds or context-aware similarity adjustments could help balance the trade-offs by dynamically fine-tuning the discount factor based on dataset characteristics or user behavior. Additionally, integrating collaborative filtering with content-based approaches or employing hybrid methods can overcome limitations by leveraging external data to supplement sparse interactions or weak correlations.

This study also emphasizes the potential for personalized recommendation systems to evolve beyond static similarity measures. By incorporating techniques like temporal dynamics (to account for changes in user preferences over time) and neural collaborative filtering models, future systems could significantly enhance both accuracy and user satisfaction.

In conclusion, the iterative refinement of similarity measures and the exploration of adaptive or hybrid models present valuable opportunities for further advancements in the field of recommender systems. These insights serve as a foundation for building more robust, efficient, and user-centric recommendation systems.

**Conclusion**

This analysis demonstrated the nuanced interplay between similarity measures and recommendation quality in item-based collaborative filtering systems. By leveraging the Pearson Correlation Coefficient (PCC) as the foundation and exploring the effects of a discount factor, the study underscored how subtle adjustments to similarity computations can significantly influence the performance and behavior of a recommendation system.

The original PCC metric proved effective in maintaining broader recommendation coverage by preserving weaker relationships, which can be particularly valuable in diverse or sparse datasets. However, the inclusion of a discount factor brought forth a critical trade-off: while it improved precision by prioritizing stronger correlations, it introduced sparsity that limited the system’s ability to cater to certain users or items. This duality emphasizes the importance of striking an optimal balance between preserving diversity and ensuring accuracy.

Moreover, the findings revealed the importance of tailoring collaborative filtering techniques to specific use cases. For example, scenarios requiring a broad exploration of item catalogs might favor unadjusted PCC, whereas contexts emphasizing accuracy and relevance, such as personalized e-commerce, could benefit from a refined, discounted approach.

The implications of this study extend beyond collaborative filtering. Adaptive methods, such as dynamic thresholds or hybrid models that integrate collaborative and content-based filtering, hold promise for addressing the limitations identified. These approaches could dynamically adjust to user preferences, dataset characteristics, or application goals, paving the way for more flexible and effective recommendation systems.

Future research should explore advanced techniques, such as neural collaborative filtering and context-aware modeling, to further enhance the adaptability and precision of recommendation algorithms. Additionally, incorporating temporal dynamics and user feedback could lead to more personalized and responsive systems, ensuring sustained user engagement and satisfaction.

In summary, the findings of this report provide a robust foundation for improving item-based collaborative filtering techniques. By understanding the trade-offs involved in similarity measures and embracing adaptive strategies, recommendation systems can evolve into powerful tools that not only meet user needs but also adapt seamlessly to dynamic environments and growing datasets.