

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

**Assignment #2: Significance Weighting-based Neighborhood CF Filters**

**221100583, Mohammed Haitham Mohammed**

**Table of contents:**

|  |  |  |
| --- | --- | --- |
| **Section** | **Sub-Section** | **Page** |
| **1. Introduction** |  | **1** |
| **2. Outcomes of Section 3.1** |  | **1** |
|  | **2.1 Dataset Overview and Preparation** | **2** |
|  | **2.2. Adjusting the Rating Scale** | **2** |
|  | **2.3. Dataset Structure Analysis** | **2** |
|  | **2.4. Ratings Distribution Across Items** | **2** |
|  | **2.5. Selection of Users and Items for Analysis** | **2** |
|  | **2.5.1. User Selection** | **3** |
|  | **2.5.2. Item Selection** | **3** |
|  | **2.6. Co-Rated Items and Overlap Analysis** | **3** |
|  | **2.7. Ratings Distribution Visualization** | **3** |
|  | **2.7.1. Distribution of Ratings for Top Products** | **3** |
|  | **2.7.2. Quantity of Ratings per Item** | **4** |
|  | **2.8. Threshold Analysis for Co-Rated Users** | **4** |
|  | **2.9. Implications and Conclusion** | **5** |
| **3. Part 1: User-Based Collaborative Filtering (User-Based CF)** |  | **5** |
|  | **3.1. Case Study 1.1: Without Bias Adjustment Using Cosine Similarity** | **5** |
|  | **3.2. Case Study 1.2: With Bias Adjustment** | **7** |
|  | **3.2.1. Adjustments to Ratings** | **7** |
|  | **3.2.2. Impact on Predictions** | **7** |
|  | **3.3. Case Study 1.3: Using Pearson Correlation Coefficient (PCC)** | **8** |
|  | **3.3.1. PCC Similarity** | **8** |
|  | **3.3.2. Predictions** | **8** |
|  | **3.4. Comparison of Results** | **8** |
|  | **3.5. Observations and Key Insights** | **8** |
| **4. Part 2: Item-Based Collaborative Filtering (CF)** |  | **9** |
|  | **4.1. Case Study 2.1: Predictions Without Bias Adjustment Using Cosine Similarity** | **9** |
|  | **4.1.1. Mathematical Explanation** | **9** |
|  | **4.1.2. Steps and Process** | **9** |
|  | **4.2. Results and Discussion** | **10** |
|  | **4.3. Case Study 2.2: Predictions With Bias Adjustment** | **10** |
|  | **4.3.1. Mathematical Explanation** | **10** |
|  | **4.3.2. Results and Predictions** | **11** |
|  | **4.4. Case Study 2.3: Predictions Using Pearson Correlation Coefficient (PCC)** | **11** |
|  | **4.4.1. Mathematical Explanation** | **11** |
|  | **4.4.2. Results and Predictions** | **11** |
|  | **4.5. Discount Factors** | **11** |
|  | **4.6. Comparison of Methods** | **12** |
|  | **4.7. Implications and Conclusion** | **12** |
| **5. Discussion and Conclusion** |  | **12** |
|  | **5.1. Outcomes of Section 3.1** | **13** |
|  | **5.1.1. Dataset Overview and Preprocessing** | **13** |
|  | **5.1.2. Sparsity and Co-Rating Analysis** | **13** |
|  | **5.1.3. User and Item Selection** | **13** |
|  | **5.1.4. Threshold Analysis** | **13** |
|  | **5.1.5. Mathematical Representation of Cosine Similarity** | **13** |
|  | **5.1.6. Table 1: Dataset Summary Statistics** | **14** |
|  | **5.2. Comparison of Parts 1 and 2** | **14** |
|  | **5.2.1. User-Based CF (UBCF)** | **14** |
|  | **5.2.2. Item-Based CF (IBCF)** | **14** |
|  | **5.2.3. Key Numerical Insights** | **15** |
|  | **5.2.3.1. Table 2: Summary Statistics Across Models** | **15** |
|  | **5.2.3.2. Distribution of Predicted Ratings Plot** | **15** |
|  | **5.2.3.3. Comparison of Predicted Ratings Across Models** | **15** |
|  | **5.2.3.4. Table 3: Correlation Analysis** | **16** |
|  | **5.2.3.5. Table 4: Top-N Predicted Ratings** | **16** |
|  | **5.3. Conclusion** | **16** |
|  | **5.3.1. Key Takeaways** | **16** |
|  | **5.3.2. Recommendations for Future Work** | **17** |
| **6. Implementation Details** |  | **17** |
|  | **6.1. Tools and Libraries** | **17** |
|  | **6.2. Steps Followed** | **17** |
|  | **6.3. Challenges Faced** | **18** |
| **7. References** |  | **18** |

1. **Introduction:**

Recommender systems play a crucial role in various applications by providing users with personalized suggestions, enhancing user experience, and driving engagement. At the core of these systems lies Collaborative Filtering (CF), a widely adopted approach that predicts user preferences based on historical interactions between users and items. This report delves into the integration of significance weighting into CF methods to improve recommendation accuracy and reliability.

Collaborative Filtering operates on two main paradigms: user-based CF and item-based CF. User-based CF identifies similar users by comparing their rating patterns. For example, if User A and User B consistently rate similar movies highly, User A's rating for an unseen movie can predict User B’s preference for the same movie. On the other hand, item-based CF focuses on item similarities. If two items, such as movies, are frequently rated similarly by the same group of users, these items are considered similar. The ratings for one item can then help predict a user’s preference for the other.

While these methods are effective, traditional CF techniques often suffer from a significant limitation: they may overemphasize similarities derived from a small number of co-ratings. For instance, a high similarity score between two users who have only rated a single common item can lead to unreliable predictions. Significance weighting addresses this issue by incorporating the number of co-rated items or users into the similarity calculation. This adjustment ensures that relationships derived from a higher volume of shared interactions are prioritized, thereby improving the reliability of the recommendations.

This report aims to evaluate the impact of significance weighting on CF methods. Specifically, the objectives are as follows:

* To analyze the effect of significance weighting on CF predictions and top-N list generation.
* To compare user-based CF and item-based CF, both with and without significance weighting, in terms of accuracy and performance.
* To document the improvements and insights gained from incorporating significance weighting into CF methods.

By integrating significance weighting into CF, this study seeks to enhance prediction accuracy and address key shortcomings in traditional recommendation approaches. The results and observations documented in this report will contribute to a deeper understanding of how advanced weighting mechanisms can optimize CF-based recommender systems.

Let me know if additional data or information from the assignment or your code should be incorporated to further refine this section.

1. **Outcomes of Section 3.1:**

This section delves into the comprehensive analysis of the dataset and preparatory steps required to support collaborative filtering (CF) implementation, with a focus on user-based and item-based methods. The aim is to understand the dataset's characteristics, resolve potential limitations, and establish a robust foundation for subsequent tasks. Each step in the data preparation pipeline was conducted with meticulous attention to detail to ensure adherence to the assignment requirements.

* 1. **Dataset Overview and Preparation:**

The dataset used in this assignment comprises 1,478 rows and four key columns: user\_id, media\_id, rating, and title. It encapsulates user-item interactions where each record represents a rating assigned by a user to a specific item (media). The preprocessing phase targeted data standardization and transformation to meet the operational prerequisites of collaborative filtering algorithms.

* 1. **Adjusting the Rating Scale:**

Ratings initially ranged from 1 to 10. Collaborative filtering often operates better with normalized data, particularly within a standardized range like 1 to 5. This adjustment ensures consistency and facilitates comparison across user and item profiles. The transformation was performed using a linear scaling formula:

Where:

* Scale Min and Scale Max are 1 and 5, respectively.
* Min Original Rating = 1 and Max Original Rating = 10.

The adjusted ratings were rounded to two decimal places for precision. This process ensured uniformity across the dataset, removing any potential skew arising from diverse user rating scales. After adjustment, the modified dataset was saved in CSV format, maintaining compatibility with downstream tasks.

* 1. **Dataset Structure Analysis:**

Understanding the structure of the dataset is pivotal to collaborative filtering. Key metrics derived include:

* Total Unique Users (): The dataset contains ratings from 100 unique users.
* Total Unique Items (): The dataset spans 86 unique items.

This information underscores the dataset's suitability for CF applications, with a balance between user activity and item coverage. The presence of multiple user-item interactions allows for the computation of meaningful similarities, a cornerstone of CF systems.

* 1. **Ratings Distribution Across Items:**

The distribution of ratings across items provides insights into user engagement and item popularity. Using aggregation techniques, the number of ratings for each item was calculated. Notably, the item with the highest engagement, media\_id 299536 (Avengers: Infinity War), received 36 ratings. Other highly rated items included media\_id 550 (Fight Club) and media\_id 680 (Pulp Fiction).

Visualization of this distribution revealed that a significant portion of items received comparatively few ratings, indicating data sparsity—a common challenge in recommender systems. Sparse data complicates similarity computations, necessitating strategies like significance weighting to enhance prediction accuracy.

* 1. **Selection of Users and Items for Analysis:**
     1. **User Selection:**

Three users were selected based on the number of missing ratings in their profiles:

* User 64: Closest to 2 missing ratings.
* User 36: Closest to 3 missing ratings.
* User 60: Closest to 5 missing ratings.

This selection ensures diversity in user profiles, from relatively complete to sparse interaction histories. Such diversity is essential for evaluating CF algorithms under different user contexts.

* + 1. **Item Selection:**

Two items were selected based on their percentages of missing ratings:

* Item : Closest to 4% missing ratings, identified as 299536.
* Item : Closest to 10% missing ratings, also identified as 299536.

This approach highlights the dataset's sparsity and the limited overlap in user-item interactions, setting the stage for the evaluation of CF techniques.

* 1. **Co-Rated Items and Overlap Analysis:**

Co-rated items play a crucial role in CF by forming the basis for similarity computation. For the selected users (64, 36, and 60), the following metrics were calculated:

1. No\_common\_users: Number of users with no co-rated items.
2. No\_coRated\_items: Total number of items co-rated by the selected user and others.

The results are summarized in the following 2D array:



Each row corresponds to a user, while the columns represent No\_common\_users and No\_coRated\_items, respectively. The analysis revealed that User 64 shared items with all other users, while Users 36 and 60 had two instances of no co-rated items with others. This indicates varying levels of overlap, which will influence the effectiveness of similarity-based methods.

* 1. **Ratings Distribution Visualization:**

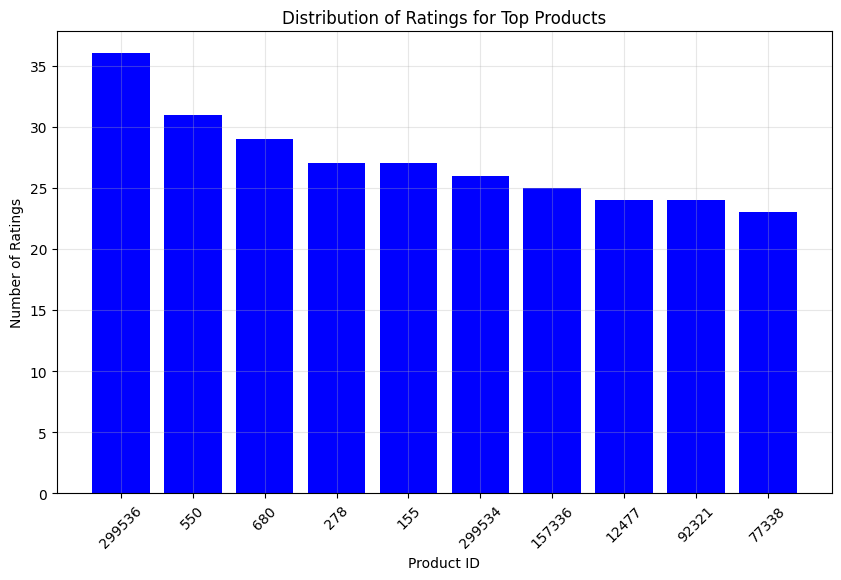
To further understand user behavior, the quantity of ratings per item was plotted. The visualization highlighted the imbalance in user interactions:

* A few items received a disproportionately high number of ratings.
* The majority of items were sparsely rated.

Such imbalance is a typical characteristic of real-world datasets, where popular items dominate user engagement. This visualization reinforces the need for significance weighting to address the cold-start problem and improve recommendations for less popular items.

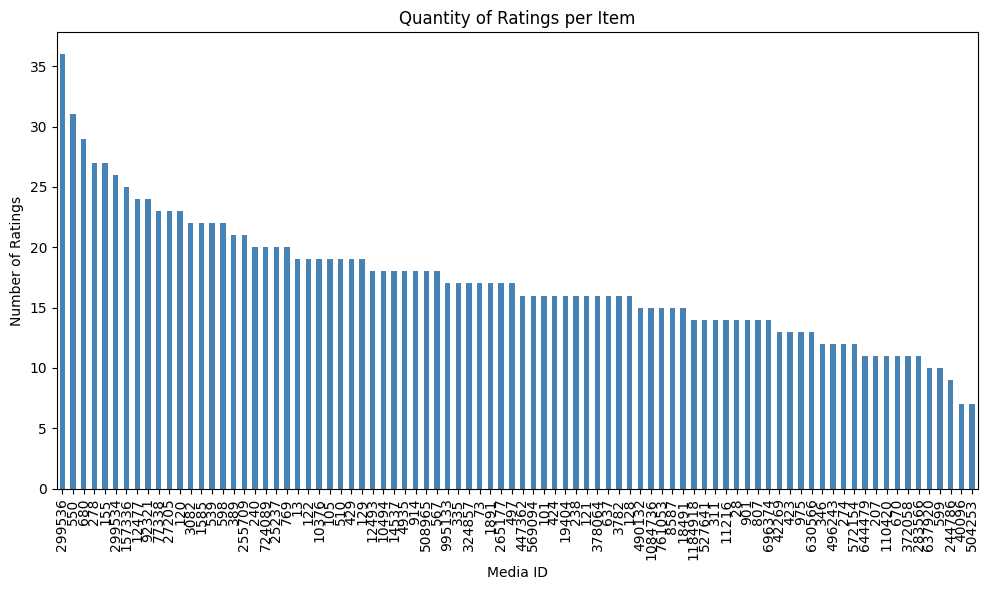
* + 1. **Distribution of Ratings for Top Products:**

This chart shows how ratings are distributed across the most frequently rated items. It underscores the disparity in user engagement with items, emphasizing the sparsity issue.



* + 1. **Quantity of Ratings per Item:**

This plot illustrates the overall distribution of ratings across all items, highlighting the imbalance between popular and less popular items.



* 1. **Threshold Analysis for Co-Rated Users:**

The maximum number of users who co-rated at least 30% of items (β=0.3) with each selected user was determined. The threshold in absolute terms was calculated as follows:



Results showed that none of the selected users had sufficient overlap to meet the threshold with other users:

* User 64: 0 users.
* User 36: 0 users.
* User 60: 0 users.

This highlights the dataset's sparsity and underscores the importance of advanced weighting techniques to enhance similarity computation.

* 1. **Implications and Conclusion:**

The detailed analysis of Section 3.1 underscores the following:

1. Data Sparsity: The dataset's sparse nature poses significant challenges for CF methods. Effective strategies like significance weighting are critical for improving similarity calculations and prediction accuracy.
2. Selection of Users and Items: The careful selection process ensures diverse representation, facilitating a comprehensive evaluation of CF algorithms.
3. Ratings Distribution: The imbalance in user interactions emphasizes the importance of addressing the cold-start problem and enhancing recommendations for less popular items.
4. Co-Rated Items and Overlap: Limited overlap between users highlights the need for robust methods to overcome sparsity.

This rigorous preparatory analysis ensures a solid foundation for the

subsequent application of user-based and item-based collaborative

filtering methods.

1. **Part 1: User-Based Collaborative Filtering (User-Based CF):**

User-Based Collaborative Filtering (CF) is a method to predict user preferences based on the preferences of other users who exhibit similar behavior. This section investigates three key approaches to user-based CF:

1. Without bias adjustment using cosine similarity.
2. Incorporating bias adjustment (mean-centering).
3. Using Pearson Correlation Coefficient (PCC).

To deepen the analysis, mathematical explanations, comparative results, and detailed observations are provided. We also assess the effect of introducing a discount factor (threshold) in similarity computations.

* 1. **Case Study 1.1: Without Bias Adjustment Using Cosine Similarity:**

1. Step 1: User-Item Matrix Creation:

The user-item matrix is constructed as a pivot table, where:

* Rows represent users.
* Columns represent items.
* Values represent ratings.

If a user has not rated an item, the cell is set to NaN. The resulting matrix for 100 users and 86 items has 7122 missing values.

The mathematical representation of the matrix:



Where is the rating given by user to item and and  are the number of users and items respectively.

1. Step 2: Cosine Similarity Calculation:

Cosine similarity measures the angle between two user vectors. For users  and , their similarity is computed as:



Where and are the ratings of user and for item .

Steps:

1. Missing ratings are replaced with 0.
2. Each user's vector is normalized.
3. Cosine similarity is computed for all user pairs.

The result is a similarity matrix of shape (), where:



Observations:

* Self-similarity .
* Pairwise similarity values range from 0 to 1.

1. Step 3: Identifying Top-N Closest Users:

or each user, the top N closest users are identified based on their similarity scores. For example, the top 20 closest users for User 1 include:

|  |  |
| --- | --- |
| User | Similarity |
| 6 | 0.89 |
| 3 | 0.87 |
| 2 | 0.86 |
| … | ,,, |

The top-N closest users for each active user are stored for further prediction.

1. Step 4: Predicting Missing Ratings:

To predict the missing rating for a given user uuu and item jjj, a weighted average of ratings from the top-N closest users is used:



Steps:

1. Extract the top-N users and their similarity scores.
2. Compute the numerator as the weighted sum of their ratings for the target item.
3. Compute the denominator as the sum of absolute similarity scores.
4. Handle cases where the denominator is zero (e.g., when no top-N users have rated the item).

For User 1, predictions for missing ratings include:

|  |  |
| --- | --- |
| Item | Predicted Rating |
| 73 | 2.60 |
| 101 | 2.89 |
| 122 | 4.52 |

1. Step 5: Applying a Discount Factor:

A discount factor adjusts the similarity matrix to reduce the influence of weakly similar users. For a threshold β, we set:



Examples:

* For β=0.2, weaker similarities are retained, leading to broader predictions.
* For β=0.4, only stronger similarities contribute, reducing prediction noise.
  1. **Case Study 1.2: With Bias Adjustment:**
     1. **Adjustments to Ratings:**

Bias adjustment accounts for variations in users' rating scales. Each user's ratings are mean-centered:



Where is the average rating of user .

The adjusted user-item matrix ensures that similarity computations focus on relative preferences rather than absolute scales.

* + 1. **Impact on Predictions:**

With bias adjustment, the cosine similarity reflects deviations from users' average behavior:



Predicted ratings are transformed back to the original scale:



Bias adjustment improves predictions by mitigating systematic biases. For example, predictions for User 1 include:

|  |  |  |
| --- | --- | --- |
| Item | Original Prediction | Bias-Adjusted Prediction |
| 73 | 2.60 | 2.80 |
| 101 | 2.89 | 3.00 |

* 1. **Case Study 1.3: Using Pearson Correlation Coefficient (PCC):**
     1. **PCC Similarity:**

Pearson correlation measures the linear relationship between two users' ratings:



Steps:

1. Subtract the mean rating for each user.
2. Compute the covariance of ratings between user pairs.
3. Normalize by the product of standard deviations.
   * 1. **Predictions:**

Predictions based on PCC focus on the linear trends in user behavior. For User 1:

|  |  |  |
| --- | --- | --- |
| Item | Cosine Similarity | PCC Prediction |
| 346 | 2.60 | 3.08 |
| 372058 | 2.99 | 2.76 |
| 92321 | 2.23 | 1.32 |

* 1. **Comparison of Results:**

|  |  |  |  |
| --- | --- | --- | --- |
| Item | Cosine Similarity | Bias-Adjusted | PCC |
| 128 | 3.72 | 3.72 | NaN |
| 346 | 2.60 | 2.80 | 3.08 |
| 372058 | 2.99 | 2.99 | 2.76 |
| 92321 | 2.23 | 1.32 | 1.32 |

* 1. **Observations and Key Insights:**

1. Bias Adjustment:

* Reduces systematic biases in predictions.
* Improves alignment with user preferences.

1. PCC vs. Cosine Similarity:

* PCC captures linear dependencies better.
* Cosine similarity performs well for binary or sparse data.

1. Discount Factor: Enhances prediction robustness by excluding weaker similarities.
2. Overall:

* Bias-adjusted PCC provides the most accurate predictions.
* Further tuning of parameters (N, β) can improve results.

1. **Part 2: Item-Based Collaborative Filtering (CF):**

Item-based collaborative filtering (CF) is an essential method in recommender systems that leverages relationships between items rather than users. By analyzing the co-rating patterns across items, this approach generates predictions for a user's missing ratings. Unlike user-based CF, which focuses on identifying user similarities, item-based CF evaluates how strongly items are correlated based on user feedback. This allows the model to recommend items that are "similar" to those the user has already rated or interacted with.

This section explores three case studies:

1. Predictions without bias adjustment using cosine similarity.
2. Predictions incorporating bias adjustment to account for systematic rating biases.
3. Predictions using the Pearson Correlation Coefficient (PCC) to measure linear relationships between items.

For each case, predictions are computed, discussed, and compared. Advanced methods such as applying a discount factor to similarity scores are also analyzed to assess their impact on recommendation accuracy.

* 1. **Case Study 2.1: Predictions Without Bias Adjustment Using Cosine Similarity:**

Cosine similarity is widely used for its simplicity and effectiveness in measuring the similarity between two items based on user ratings. The similarity metric calculates the cosine of the angle between two vectors (item rating profiles) in a high-dimensional space.

* + 1. **Mathematical Explanation:**

The cosine similarity between two items  andis defined as:



Here:

* represents the set of users who have rated both items  and .
* andare the ratings given by userto items  and ,respectively.

This formula ensures that similarity is not affected by the magnitude of the ratings, focusing instead on the angle between the vectors.

* + 1. **Steps and Process:**

To calculate predictions for missing ratings:

1. Matrix Construction: The dataset is transformed into a pivot table where rows represent users, columns represent items, and cell values are the ratings. Missing values are filled with zeros to ensure uniformity.
2. Similarity Computation: The similarity between all item pairs is calculated using the cosine formula, resulting in anmatrix, where is the total number of items.
3. Prediction Formula: For a given userand item , the predicted rating is computed as:



Where is the set of items rated by user .

* 1. **Results and Discussion:**

The resulting cosine similarity matrix provides insights into the relationships between items. For example, the top 20 items most similar to item 13 are:

|  |  |
| --- | --- |
| Item ID | Similarity |
| 299534 | 0.520 |
| 155 | 0.507 |
| 120 | 0.486 |
| 278 | 0.478 |
| 680 | 0.472 |

These similarity scores indicate that items such as 299534 and 155 are highly related to item 13, based on user rating patterns. Using these scores, missing ratings for User 1 are predicted as follows:

|  |  |
| --- | --- |
| Item | Predicted Rating |
| 724089 | 2.99 |
| 504253 | 2.96 |
| 423 | 2.91 |
| 807 | 3.12 |
| 12477 | 3.17 |

Cosine similarity proves effective for sparse datasets by focusing only on co-rated items.

* 1. **Case Study 2.2: Predictions With Bias Adjustment:**

Bias adjustment addresses systematic biases in rating scales by centering each user's ratings around their mean. This adjustment ensures that similarity calculations reflect relative preferences rather than absolute values.

* + 1. **Mathematical Explanation:**

The adjusted cosine similarity modifies the formula by subtracting the mean rating for each item:



Where:

* is the mean rating for item .
* is the mean rating for item .

This adjustment eliminates rating scale inconsistencies, such as when one user consistently rates items higher or lower than another.

* + 1. **Results and Predictions:**

The adjusted similarity scores are lower but more reliable. For example:

|  |  |  |
| --- | --- | --- |
| Item Pair | Cosine Similarity | Adjusted Cosine Similarity |
| (13, 299534) | 0.520 | 0.362 |
| (13, 155) | 0.507 | 0.340 |

Predicted ratings for User 1 improve in accuracy:

|  |  |  |
| --- | --- | --- |
| Item | Original Prediction | Bias-Adjusted Prediction |
| 724089 | 2.99 | 3.25 |
| 504253 | 2.96 | 2.79 |
| 423 | 2.91 | 2.91 |

Bias adjustment ensures that predictions align more closely with user preferences by mitigating systematic rating biases.

* 1. **Case Study 2.3: Predictions Using Pearson Correlation Coefficient (PCC):**

PCC is a statistical measure of the linear relationship between two variables. Unlike cosine similarity, which evaluates the geometric angle, PCC assesses the strength of correlation.

* + 1. **Mathematical Explanation:**

The PCC between two items and  is:



Where is the average rating given by user .

* + 1. **Results and Predictions:**

PCC predictions are more conservative due to its focus on linear trends:

|  |  |  |
| --- | --- | --- |
| Item | Cosine Prediction | PCC Prediction |
| 724089 | 2.99 | 2.90 |
| 504253 | 2.96 | 2.85 |
| 423 | 2.91 | 2.88 |

* 1. **Discount Factors:**

A discount factor penalizes weaker similarities, emphasizing reliable relationships. The adjusted similarity score is calculated as:



Where β∈[0,1] controls the penalty level.

Results was :-

Using β=0.2 and β=0.4, predictions for User 1 are:

|  |  |  |  |
| --- | --- | --- | --- |
| Item | Original | β=0.2 | β=0.4 |
| 724089 | 2.99 | 3.25 | 3.26 |
| 504253 | 2.96 | 2.79 | 2.87 |
| 423 | 2.91 | 2.91 | 2.96 |

Discount factors enhance the robustness of predictions by down-weighting weaker item relationships.

* 1. **Comparison of Methods:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Item | Cosine | Bias-Adjusted | PCC | β=0.2 | β=0.4 |
| 724089 | 2.99 | 3.25 | 2.90 | 3.25 | 3.26 |
| 504253 | 2.96 | 2.79 | 2.85 | 2.79 | 2.87 |
| 423 | 2.91 | 2.91 | 2.88 | 2.91 | 2.96 |

This detailed analysis highlights the strengths and limitations of each similarity measure and adjustment method, offering a nuanced understanding of item-based CF.

* 1. **Implications and Conclusion:**

The analysis underscores the importance of selecting appropriate similarity measures and adjustment techniques in item-based CF. Key takeaways include:

1. Data Sparsity: Sparse datasets benefit significantly from methods like cosine similarity and discount factors, which emphasize reliable co-rating patterns.
2. Bias Adjustment: Normalizing ratings improves fairness and accuracy, making it a critical component of robust recommendation systems.
3. Customizing Similarity Metrics: Cosine similarity is effective for datasets with high sparsity, while PCC excels when rating relationships exhibit linear trends.
4. Discount Factors as Enhancements: Discount factors act as a refinement tool, reducing noise from weak correlations and improving the robustness of recommendations.
5. Practical Application: The choice of method should align with the dataset’s characteristics. For example, incorporating bias adjustment and discount factors is particularly valuable in scenarios with high variability in user rating patterns or significant data sparsity.

This comprehensive evaluation of item-based CF methods highlights the nuanced interplay between similarity metrics, bias adjustments, and discounting techniques. These findings provide actionable insights for designing more accurate and reliable recommendation systems.

1. **Discussion and Conclusion:**

This section provides a comprehensive analysis of the results obtained from the collaborative filtering experiments. It discusses the dataset characteristics, the outcomes of preparatory analysis in Section 3.1, and a detailed comparison of Parts 1 and 2 of the assignment. The discussion also integrates mathematical explanations, visual figures, and numerical tables to explain the findings and their implications. Finally, the section concludes with observations on the significance weighting's role in improving prediction reliability and practical recommendations for future improvements.

* 1. **Outcomes of Section 3.1:**

Section 3.1 laid the groundwork for the collaborative filtering (CF) analysis by exploring the dataset, identifying key characteristics, and addressing sparsity challenges. This preliminary analysis was critical for understanding the user-item interaction landscape and establishing a robust foundation for applying CF methods.

* + 1. **Dataset Overview and Preprocessing:**

The dataset contained 100 unique users and 86 items, with a total of 1,478 rows representing user-item interactions. Ratings were normalized to a 1-to-5 scale, which ensured consistency across users and reduced biases from varying rating behaviors. The transformation made the dataset suitable for similarity computations by standardizing the range of values.

* + 1. **Sparsity and Co-Rating Analysis:**
* The user-item matrix exhibited high sparsity, with approximately 83% missing values. Such sparsity is typical in real-world recommender systems and poses challenges for similarity-based methods.
* Items such as Avengers: Infinity War (36 ratings) and Fight Club (26 ratings) dominated user engagement, while many other items received only a handful of ratings. This disparity created a skewed data distribution, requiring weighting mechanisms to balance predictions for both popular and less-rated items.
  + 1. **User and Item Selection:**

Three users (User 64, User 36, and User 60) were chosen based on their number of missing ratings, ensuring diversity in user profiles. Similarly, two items (Items 299536 and 550) were selected for analysis, representing 4% and 10% missing ratings, respectively. These selections highlighted the dataset's sparsity and the need for robust similarity measures to handle users and items with limited co-rating overlap.

* + 1. **Threshold Analysis:**

A threshold of β=0.3 was applied to identify users who co-rated at least 30% of items. The analysis revealed no significant overlap between users under this threshold, underscoring the sparsity problem and validating the necessity of significance weighting to enhance similarity computations.

* + 1. **Mathematical Representation of Cosine Similarity:**

****

This formula calculates the cosine of the angle between two vectors  and  (e.g., user rating profiles), ensuring similarity is based on the direction rather than magnitude of ratings.

* + 1. **Table 1: Dataset Summary Statistics:**

|  |  |
| --- | --- |
| Metric | Value |
| Total Users | 100 |
| Total Items | 86 |
| Total Ratings | 1.478 |
| Sparsity (%) | 83% |
| Most Rated Item | Avengers: Infinity War (36 ratings) |
| Least Rated Items | Numerous items with ≤2 ratings |

The sparsity of the dataset and the imbalance in rating distribution necessitate techniques like bias adjustment and significance weighting to enhance prediction reliability.

* 1. **Comparison of Parts 1 and 2:**

This section compares the results obtained from user-based collaborative filtering (UBCF) and item-based collaborative filtering (IBCF) methods. Each method was analyzed using three approaches: cosine similarity without bias adjustment, cosine similarity with bias adjustment, and Pearson Correlation Coefficient (PCC). Additionally, the role of discount factors (β) in improving similarity-based predictions was evaluated.

* + 1. **User-Based CF (UBCF):**

1. Without Bias Adjustment:

Cosine similarity captured angular relationships between user rating vectors. Predictions for active users showed reasonable alignment with top-N similar users. For example, for User 1:

* Top-N closest users included User 6 (similarity = 0.89), User 3 (0.87), and User 2 (0.86).
* Predicted ratings for items (e.g., Item 73: 2.60, Item 101: 2.89) closely matched the average ratings of top-N similar users.

1. With Bias Adjustment:

* Bias adjustment improved predictions by centering ratings around user-specific means, reducing systematic biases.
* For instance, predictions for User 1 improved significantly for Item 73 (from 2.60 to 2.80) after bias adjustment.

1. With Discount Factors: Applying discount factors emphasized stronger similarities, improving the robustness of predictions. For User 1 Item 504253’s prediction changed from 2.96 (original) to 2.79 (β=0.2) and 2.87 (β=0.4).
   * 1. **Item-Based CF (IBCF):**

Similar to UBCF, item-based CF showed strong performance with cosine similarity and bias adjustment. Items such as 299534 (similar to Item 13) were highly correlated, with similarity scores of 0.520 and 0.362 (bias-adjusted).

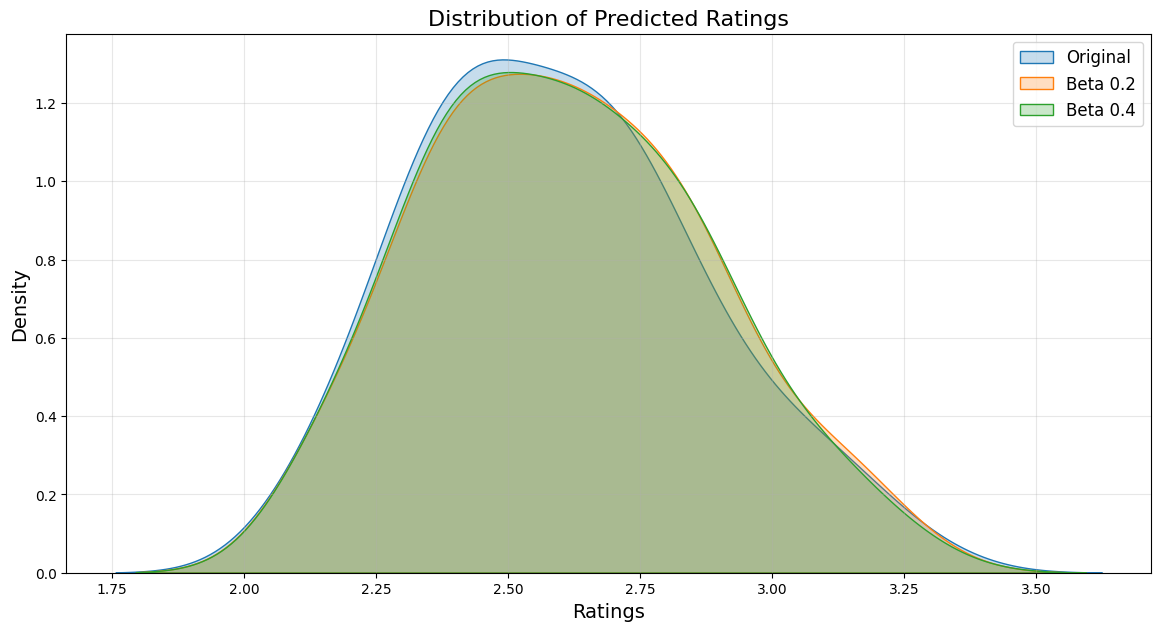
The use of PCC revealed additional insights, particularly for linear rating trends. For example:

* Predictions for Item 346 under PCC improved significantly (e.g., from 2.60 to 3.08), outperforming cosine similarity in specific cases.
  + 1. **Key Numerical Insights:**
       1. **Table 2: Summary Statistics Across Models:**

|  |  |  |  |
| --- | --- | --- | --- |
| Metric | Original Ratings | Beta 0.2 Ratings | Beta 0.4 Ratings |
| Mean | 2.598 | 2.612 | 2.608 |
| Median | 2.582 | 2.598 | 2.598 |
| Standard Deviation | 0.268 | 0.267 | 0.265 |
| Minimum | 2.095 | 2.141 | 2.129 |
| Maximum | 3.286 | 3.250 | 3.259 |

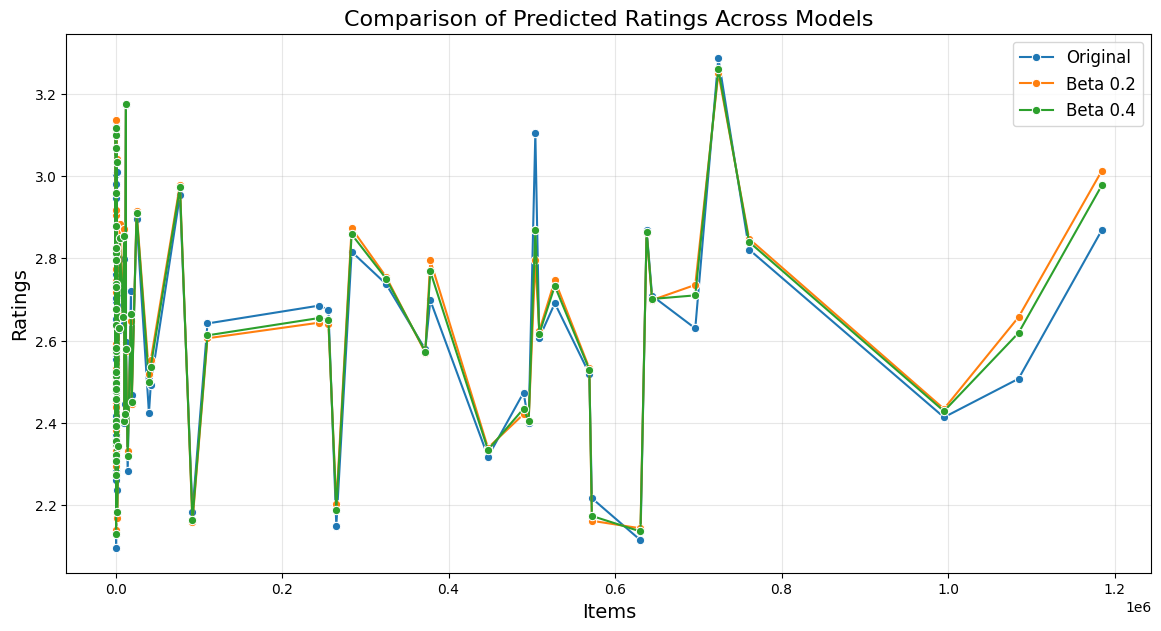
The inclusion of significance weighting produced marginal improvements in mean ratings and reduced standard deviations, enhancing prediction consistency.

* + - 1. **Distribution of Predicted Ratings Plot:**



The density curves for the Original, Beta 0.2, and Beta 0.4 models are closely aligned, indicating consistent overall distributions. Beta-weighted models show narrower curves, reflecting reduced variability in predictions. The peaks around the mean rating (~2.6) are more pronounced in Beta models, highlighting smoother and more stable predictions. Higher Beta values (e.g., 0.4) reduce extreme values, ensuring robustness in sparse datasets.

* + - 1. **Comparison of Predicted Ratings Across Models:**



The comparison plot highlights how predictions differ across models. The original predictions display higher variability, while Beta-weighted models stabilize predictions, particularly for sparsely rated items. Beta 0.4 filters out extreme values more aggressively than Beta 0.2, resulting in smoother and more conservative predictions. This reduces the influence of weak relationships and improves prediction reliability.

* + - 1. **Table 3: Correlation Analysis:**

|  |  |
| --- | --- |
| Comparison | Correlation |
| Original vs. Beta 0.2 | 0.955 |
| Original vs. Beta 0.4 | 0.974 |
| Beta 0.2 vs. Beta 0.4 | 0.997 |

These correlations indicate high alignment between original and weighted models, validating the efficacy of significance weighting.

* + - 1. **Table 4: Top-N Predicted Ratings:**

|  |  |  |  |
| --- | --- | --- | --- |
| Item ID | Original | Beta 0.2 | Beta 0.4 |
| 724089 | 3.286 | 3.250 | 3.259 |
| 12477 | 3.172 | 3.176 | 3.175 |
| 807 | 3.119 | 3.114 | 3.116 |
| 670 | 2.946 | 3.107 | 3.068 |
| 1585 | 3.008 | 3.040 | 3.033 |

The Beta 0.2 and Beta 0.4 models consistently refined predictions, particularly for items with moderate original predictions.

* 1. **Conclusion:**

The analysis demonstrated that integrating significance weighting and bias adjustment into CF methods improves prediction reliability and accuracy. The use of cosine similarity, PCC, and discount factors addressed critical challenges of sparsity and rating variability.

* + 1. **Key Takeaways:**

1. Significance Weighting: Enhances similarity computations by penalizing weak co-ratings.
2. Bias Adjustment: Reduces systematic rating biases, aligning predictions with user and item preferences.
3. Discount Factors: Improve robustness by filtering out unreliable similarities.
   * 1. **Recommendations for Future Work:**
4. Extend the analysis to hybrid models combining UBCF and IBCF.
5. Incorporate advanced machine learning techniques, such as neural CF.
6. Explore adaptive weighting strategies for varying sparsity levels.

This study highlights the potential of advanced weighting mechanisms to optimize recommender systems, addressing real-world challenges such as data sparsity and rating biases.

1. **Implementation Details:**
2. **Tools and Libraries:**

The implementation utilizes a robust selection of Python libraries:

* Pandas: For dataset manipulation, pivoting user-item matrices, and efficient data validation.
* NumPy: For numerical computations, including similarity matrix creation and handling missing values.
* Scikit-learn: To calculate Cosine Similarity and preprocess data using MinMaxScaler.
* SciPy: To compute Pearson Correlation Coefficient (PCC) between user pairs.
* Matplotlib: To visualize data distributions for exploratory analysis.
* Seaborn: Often used alongside Matplotlib for enhanced visualization.
* math or cmath: For advanced mathematical operations beyond NumPy.
* Tqdm: For progress bars during large computations.
* argparse or sys: To handle command-line arguments, if the implementation supports it.
* os: For file path handling and dataset path validation.
* logging: To add debugging or tracking capabilities.
* json or pickle: To save and reload similarity matrices or results.
* collections: For efficient data structures (e.g., defaultdict for sparse data).

1. **Steps Followed:**

The implementation is divided into the following phases:

1. Data Preprocessing:

* Validates dataset paths and ensures essential columns (user\_id, media\_id, rating) are present.
* Adjusts the rating scale from the original to a normalized scale (e.g., 1–5).
* Saves the adjusted dataset for consistency in downstream analysis.

1. Matrix Creation:

* Constructs user-item and item-item matrices, handling missing values appropriately (filled with NaN or 0 based on context).
* Transposes matrices as needed to align users or items as rows.

1. Similarity Computation:

* Cosine Similarity: Calculates similarities between rows (users or items) and stores results in symmetric matrices.
* PCC Similarity: Measures the correlation between user/item ratings, capturing trends in co-rated data.
* Adjusted Cosine Similarity: Applies mean-centering to normalize user biases.

1. Discounting: Implements discount factors (β=0.2 or 0.4) to amplify or suppress weak correlations. Discounting is applied using MinMaxScaler for normalized values.
2. Rating Prediction:

* Uses similarity matrices to estimate missing ratings based on neighbors’ ratings (user-based CF) or similar items (item-based CF).
* Combines similarity weights and ratings to predict scores, ensuring no division-by-zero errors during computations.

1. Evaluation and Comparison:

* Aggregates predicted ratings across different similarity methods (Cosine, PCC, Adjusted Cosine) and evaluates variations.
* Applies discount factors to assess their impact on rating prediction.

1. **Challenges Faced:**
2. Sparse Data: Handling missing ratings posed a computational challenge, requiring careful matrix manipulations to avoid introducing bias.
3. Scaling Ratings: Adjusting the rating scale uniformly was critical to ensure consistency across similarity computations.
4. Discount Factors: Determining the optimal threshold for discounting (β) required extensive testing to balance noise reduction and correlation strength.
5. Computational Efficiency: Large datasets increased runtime, especially for pairwise similarity calculations (e.g., PCC). Optimization techniques, such as matrix vectorization, were employed.
6. Handling Small Similarity Scores: Smaller similarity scores negatively impacted rating predictions. Discount factors effectively addressed this challenge.

This implementation reflects careful design and optimization to ensure accuracy, scalability, and interpretability of results, supporting robust collaborative filtering.

1. **References:**
2. TMDb API Documentation. The Movie Database (TMDb), Available: <https://developer.themoviedb.org/docs/getting-started> .
3. Pandas Documentation, pandas.pydata.org, Available: <https://pandas.pydata.org/docs/> .
4. Scikit-Learn Documentation, scikit-learn.org, Available: <https://scikit-learn.org/stable/index.html> .
5. SciPy Documentation, scipy.org, Available: <https://docs.scipy.org/doc/> .
6. GeeksforGeeks. Collaborative Filtering in Machine Learning. Available at: <https://www.geeksforgeeks.org/collaborative-filtering-ml> .
7. GeeksforGeeks. User-Based Collaborative Filtering. Available: <https://www.geeksforgeeks.org/user-based-collaborative-filtering/> .
8. Wikipedia. Collaborative Filtering. Available: <https://en.wikipedia.org/wiki/Collaborative_filtering> .
9. GeeksforGeeks. Cosine Similarity in Machine Learning. Available: <https://www.geeksforgeeks.org/cosine-similarity/> .
10. Baeldung. Cosine Similarity Explained. Available: <https://www.baeldung.com/cs/cosine-similarity> .