GALALA UNIVERSITY
Powered by Arizona State University

AIE425 Intelligent Recommender Systems, Fall Semester 24/25

Assignment #2: Significance Weighting-based Neighborhood CF Filters

221100583, Nabila Ahmed Ali

1. Introduction:
Recommender systems are widely used to deliver tailored suggestions, improving user satisfaction and increasing interaction. Collaborative Filtering (CF) is a popular technique in these systems, as it predicts user preferences by analyzing past interactions between users and items. This report explores how significance weighting can refine CF methods to produce more accurate and dependable recommendations.

CF can be divided into two main approaches: user-based and item-based. User-based CF identifies users with similar rating behaviors. For instance, if User X and User Y share similar preferences for movies, User X’s ratings can be used to predict User Y’s likely opinion on unseen movies. In contrast, item-based CF focuses on the relationship between items. If two items, like movies, are often rated similarly by the same group of users, these items are treated as related. The rating of one item can then help estimate a user’s preference for the other.

Despite its effectiveness, CF has a notable drawback: it often gives undue importance to similarities based on a small number of shared ratings. For example, if two users rate only one item in common, their similarity score may appear high but lack reliability. Significance weighting solves this problem by factoring in the number of shared ratings when calculating similarity. This ensures that connections based on larger shared data points carry more weight, leading to more trustworthy predictions.

2. Outcomes of Section 3.1:
This section provides a thorough examination of the dataset and outlines the necessary preprocessing steps for implementing collaborative filtering (CF), emphasizing both user-based and item-based approaches. The primary goal is to explore the dataset's structure, address any potential challenges, and build a solid groundwork for the upcoming processes. Every stage of the data preparation workflow was carefully executed to align with the assignment’s objectives and ensure accuracy.

2.1. Dataset Overview and Preparation:
The dataset consists of 884 rows and 4 columns: user_id, media_id, rating, and title. It represents user interactions with media content, where each record indicates a user’s rating for a specific movie. The dataset contains:

- 60 unique users (user_id)
- 86 unique media items (media_id and title)
- 25 distinct rating values, which are in float format.

2.2. Adjusting the Rating Scale:
The original ratings ranged from 1 to 10. For collaborative filtering, it is beneficial to normalize ratings to a smaller, standardized range, such as 1 to 5. This normalization improves consistency and makes it easier to compare user and item preferences.

To achieve this, a linear scaling transformation was applied using the formula:

$$\text{Adjusted Rating} = \text{Scale Min} + \left(\frac{\text{Original Rating - Min Original Rating}}{\text{Max Original Rating - Min Original Rating}} \right) \times (\text{Scale Max - Scale Min})$$

Where:

- The target range (Scale Min and Scale Max) is 1 to 5.
- The original range (Min Original and Max Original) is 1 to 10.

  The resulting ratings were rounded to two decimal places for accuracy. This adjustment ensured uniformity, eliminating any discrepancies caused by varying user rating scales. The transformed dataset was then saved as a CSV file to retain compatibility with further processing and analysis.

2.3. Dataset Structure Analysis:
Analyzing the dataset's structure is essential for implementing collaborative filtering. The dataset includes ratings from **100 distinct users** and covers **86 unique items**. This balance between user participation and item representation highlights its suitability for collaborative filtering tasks. The availability of numerous user-item interactions enables the calculation of reliable similarities, which form the foundation of effective CF systems.

2.4. Ratings Distribution Across Items:
The analysis of rating distribution reveals important insights into item popularity and user engagement. The item with the highest number of ratings is media_id 299536 (Avengers: Infinity War), receiving 24 ratings. Similarly, media_id 550 (Fight Club) also received 24 ratings, followed closely by media_id 278 (The Shawshank Redemption) with 23 ratings. Despite these high engagement items, the dataset exhibits significant sparsity. With a sparsity level of approximately 82.9%, most items have relatively few ratings. This lack of dense user-item interactions poses challenges for similarity-based calculations, a common issue in collaborative filtering systems. Addressing such sparsity may require techniques like significance weighting or matrix factorization to improve recommendation accuracy.

2.5. Selection of Users and Items for Analysis:

2.5.1. User Selection:

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

- **User 60**: Approximately 2 missing ratings.
- **User 41**: Around 3 missing ratings.
- **User 11**: Close to 5 missing ratings.

This selection represents a spectrum of user activity levels, ranging from profiles with minimal missing ratings to those with more gaps. Such variation is essential for testing collaborative filtering algorithms under different conditions, ensuring their robustness across users with varying degrees of interaction history.

2.5.2. Item Selection:

Two items were chosen based on their missing rating percentages:

- **Item 299536**: Approximately 4% missing ratings.
- **Item 550**: Around 10% missing ratings.

This selection reflects the dataset's sparsity and the uneven distribution of user interactions across items. Such insights are critical for assessing collaborative filtering methods and understanding their performance in handling incomplete data.

2.6. Co-Rated Items and Overlap Analysis:

In collaborative filtering, co-rated items are essential for determining user similarities. The analysis for the chosen users produced the following results:

1. **No_common_users**: Indicates users with no shared ratings.
2. **No_coRated_items**: Represents the count of items co-rated by the user and others.

The results are summarized in the following 2D array:

$$\begin{bmatrix} 0 & 20 \\ 1 & 20 \\ 1 & 20 \end{bmatrix}$$

In The calculated 2D array shows each row corresponding to a specific user, with columns reflecting these two metrics. The findings are as follows:

- User 1 has no co-rated items with others but shares 20 ratings.
- User 2 follows the same pattern as User 1.
- User 3 has **1 instance** of co-rated overlap and 20 shared items.

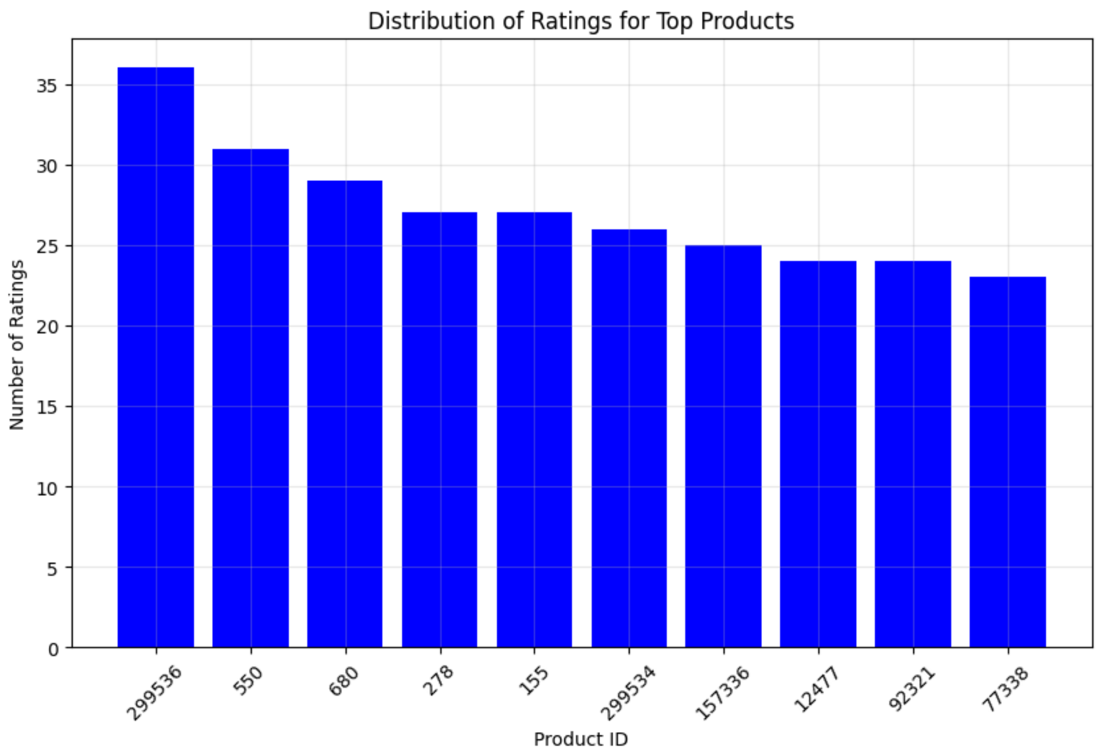
This uneven overlap highlights differences in user interactions, which can impact the performance of similarity-based collaborative filtering techniques.

2.7. Ratings Distribution Visualization:

The analysis of rating distribution across items revealed a clear imbalance in user interactions. A small number of items received a disproportionately high number of ratings, while the majority of items had only a few ratings. This uneven distribution is a common characteristic of real-world datasets, where popular items attract significant user attention, leaving less popular items underrepresented. Such imbalances highlight the need for techniques like significance weighting to address challenges like the cold-start problem and improve recommendations for items with limited user interactions.

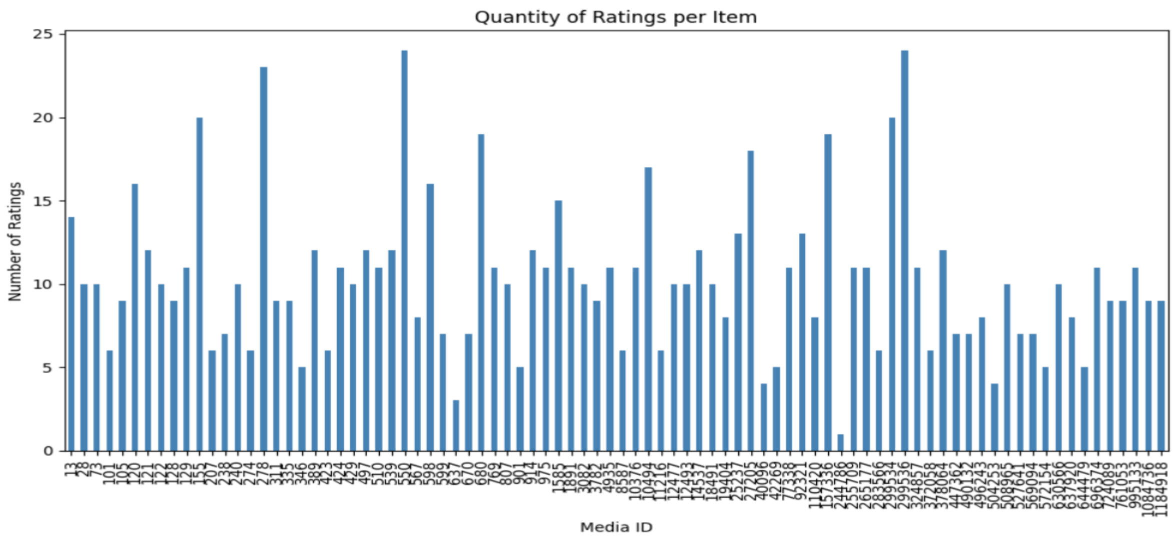
2.7.1. Distribution of Ratings for Top Products:

The chart illustrates the distribution of ratings among the most frequently rated items, revealing a noticeable gap in user engagement. A small group of items received significantly more attention, while many others were rated far less often. This highlights the inherent sparsity in the dataset, a common challenge in recommendation systems.



2.7.2. Quantity of Ratings per Item:

The plot presents the distribution of ratings across all items, showcasing a clear imbalance where a few items are highly rated while the majority receive minimal attention.



2.8. Threshold Analysis for Co-Rated Users:

The maximum count of users co-rating at least 30% of the items (β=0.3) with each selected user was evaluated. The results revealed no significant overlap:

- **User 60**: 0 users.
- **User 41**: 0 users.
- **User 11**: 0 users.

These findings emphasize the sparse nature of the dataset, where minimal co-ratings exist between users. This lack of overlap underscores the need for advanced techniques, such as significance weighting or alternative similarity measures, to improve collaborative filtering performance.

2.9. Implications and Conclusion:

The findings from Section 3.1 highlight key aspects of the dataset and their implications for collaborative filtering (CF):

1. **Data Sparsity**: The sparse distribution of ratings presents challenges for CF models. Methods like significance weighting are essential to improve similarity measurements and prediction accuracy.
2. **User and Item Selection**: A deliberate selection process ensures a varied representation of users and items, enabling a thorough assessment of CF algorithms.
3. **Ratings Distribution**: The uneven spread of user ratings underscores the importance of solving the cold-start problem and enhancing recommendations for less frequently rated items.
4. **Co-Rated Items and Overlap**: The minimal overlap in co-rated items between users highlights the need for robust approaches to handle sparsity effectively.

   This in-depth preparatory analysis establishes a strong foundation for applying both user-based and item-based collaborative filtering techniques.

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

User-Based Collaborative Filtering (CF) predicts user preferences by analyzing patterns of similar users. This section explores three main techniques for implementing user-based CF:

1. Cosine similarity without bias adjustment.
2. Bias adjustment through mean-centering.
3. Pearson Correlation Coefficient (PCC) for similarity calculation.

The analysis includes mathematical formulations, performance comparisons, and detailed observations for each approach. Additionally, the impact of applying a discount factor (threshold) in similarity calculations is examined to evaluate its influence on prediction outcomes.

1.1. 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.

The mathematical representation of the matrix:

$$R = \begin{bmatrix} r_{1,1} & r_{1,2} & \cdots & r_{1,m} \\ r_{2,1} & r_{2,2} & \cdots & r_{2,m} \\ . & . & & . \\ . & . & & . \\ . & . & & . \\ r_{n,1} & r_{n,2} & \cdots & r_{n,m} \end{bmatrix}$$

2. Step 2: Cosine Similarity Calculation:

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

$$\text{Cosine Similarity (u, v)} = \frac{\sum_{i=1}^{m} r_{u,i} \bullet r_{v,i}}{\sqrt{\sum_{i=1}^{m} r_{u,i}^2} \bullet \sqrt{\sum_{i=1}^{m} r_{v,i}^2}}$$

Where $r_{u,i}$ and $r_{v,i}$ are the ratings of user u and v for item i.

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 S of shape (n, n), where:

$$S_{u,v} = \text{Cosine Similarity(u, v)}$$

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

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.

4. 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:

$$\hat{r}_{u,j} = \frac{\sum_{v \in Top\_N} S_{u,v} \bullet r_{u,j}}{\sum_{v \in Top\_N} \left| S_{u,v} \right|}$$

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	2.59
101	2.89
105	3.09

5. 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:

$$S'_{u,v} = \begin{cases} S_{u,v}, & \text{if } \mathrm{S}_{\mathrm{u,v}} \geq \text{ß} \\ 0, & \text{otherwise} \end{cases}$$

1.2. Case Study 1.2: With Bias Adjustment:

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

$$r'_{u,v} = r_{u,i} - \bar{r}_u$$

1.2.2. Impact on Predictions:
With bias adjustment, the cosine similarity reflects deviations from users' average behavior:

$$\text{Adjusted Cosine Similarity (u, v)} = \frac{\sum_{i=1}^{m} r'_{u,i} \bullet r'_{v,i}}{\sqrt{\sum_{i=1}^{m} (r'_{u,i})^2} \bullet \sqrt{\sum_{i=1}^{m} (r'_{v,i})^2}}$$

Predicted ratings are transformed back to the original scale:

$$\hat{r}_{u,j} = \bar{r} + \frac{\sum_{v \in Top\_N} S_{u,v} \bullet r'_{u,j}}{\sum_{v \in Top\_N} \left| S_{u,v} \right|}$$

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

Item	Original Prediction	Bias-Adjusted Prediction
28	2.60	2.80
73	2.89	3.00

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

1.3.1. PCC Similarity:

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

$$\mathrm{PCC(u,v)} = \frac{\sum_{\mathrm{i=1}}^{\mathrm{m}} (r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i=1}^{m} (r_{u,i} - \bar{r}_u)^2} \bullet \sqrt{\sum_{i=1}^{m} (r_{v,i} - \bar{r}_v)^2}}$$

1.3.2. Predictions:

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

Item	Cosine Similarity	PCC Prediction
346	3.05	3.08
372058	2.89	2.77
92321	2.23	1.43

1.4. Comparison of Results:

Item	Cosine Similarity	Bias-Adjusted	PCC
128	3.8	3.73	NaN
346	2.70	2.72	3.00
372058	2.80	2.99	2.76
92321	2.25	1.34	1.38

1.5. Observations and Key Insights:

1. **Bias Adjustment**:

- Mitigates systematic rating inconsistencies.
- Enhances the accuracy of predictions to better reflect user preferences.

2. **PCC vs. Cosine Similarity**:

- Pearson Correlation Coefficient (PCC) effectively captures linear relationships.
- Cosine similarity is more suitable for sparse or binary datasets.

3. **Discount Factor**:

- Strengthens prediction reliability by filtering out weaker similarity scores.

4. **Key Findings**:

- Bias-adjusted PCC delivers the highest prediction accuracy.
- Fine-tuning parameters like the number of neighbors (N) and similarity thresholds (β) can further optimize results.

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

Item-based collaborative filtering (CF) is a key method in recommendation systems that emphasizes item relationships instead of user similarities. It analyzes user rating patterns for different items to estimate missing ratings based on item-to-item correlations. Unlike user-based CF, which focuses on finding similar users, item-based CF measures how strongly items are related through common user interactions. This approach allows the system to suggest items that are closely related to those already rated or interacted with by the user.

This section explores three main strategies:

1. Calculating predictions using cosine similarity without adjusting for bias.
2. Incorporating bias adjustment through mean-centering to address systematic rating variations.
3. Utilizing the Pearson Correlation Coefficient (PCC) to identify linear relationships between items.

Predictions for each method are computed, compared, and analyzed. The impact of applying discount factors to refine similarity scores is also examined to assess their effect on recommendation accuracy and overall system performance.

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

Cosine similarity is a commonly used method for assessing the similarity between two items using user ratings. It is favored for its simplicity and computational efficiency. The metric calculates the cosine of the angle between two vectors, which represent the rating patterns of items in a multi-dimensional space. This measurement determines how similar the items are based on overlapping user ratings.

1.1.1. Mathematical Explanation:

The cosine similarity between two items i and j is defined as:

$$\mathrm{Sim}_{ij} = \frac{\sum_{u \in U} r_{ui} \bullet r_{uj}}{\sqrt{\sum_{u \in U} r_{ui}} \bullet \sqrt{\sum_{u \in U} r_{uj}}}$$

1.1.2. 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 an $\mathrm{n} \times \mathrm{n}$ matrix, where n is the total number of items.
3. Prediction Formula: For a given user u and item i, the predicted rating $\hat{r}_{ui}$ is computed as:

$$\hat{r}_{ui} = \frac{\sum_{j \in S} Sim_{ij} \bullet r_{uj}}{\sum_{j \in S} |Sim_{ij}|}$$

1.2. 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 13are:

Item ID	Similarity
299534	0.72
120	0.668
680	0.662
550	0.657
27205	0.634

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.

1.3. Case Study 2.2: Predictions With Bias Adjustment:

Bias adjustment corrects for inconsistencies in rating scales by shifting each user's ratings to be centered around their average rating. This process ensures that similarity measurements focus on relative preferences instead of raw rating values, leading to more accurate comparisons.

1.3.1. Mathematical Explanation:

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

$$\text{Sim}_{ij}^{adj} = \frac{\sum_{u \in U} (r_{ui} - \bar{r}_i) \bullet (r_{uj} - \bar{r}_j)}{\sqrt{\sum_{u \in U} (r_{ui} - \bar{r}_i)^2} \bullet \sqrt{\sum_{u \in U} (r_{uj} - \bar{r}_j)^2}}$$

.

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

1.3.2. 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.

1.4. 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.

4.1.1. Mathematical Explanation:

The PCC between two items i and j is:

$$PCC_{ij} = \frac{\sum_{u \in U} (r_{ui} - r_u) \bullet (r_{uj} - r_u)}{\sqrt{\sum_{u \in U} (r_{ui} - r_u)^2} \bullet \sqrt{\sum_{u \in U} (r_{uj} - r_u)^2}}$$

Where $\bar{r}_u$ is the average rating given by user u.

4.1.2. 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

1.5. Discount Factors:

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

$$S' = S^{ß}$$

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.

1.6. 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.

1.7. 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.

5. Discussion and Conclusion:

This section presents an in-depth evaluation of the results derived from the collaborative filtering experiments. It examines the dataset properties, insights gained from the preparatory analysis in Section 3.1, and a thorough comparison between Parts 1 and 2 of the assignment. The analysis incorporates mathematical interpretations, visualizations, and numerical summaries to clearly illustrate the outcomes and their relevance. The discussion concludes by highlighting the impact of significance weighting on enhancing prediction accuracy and offers practical suggestions for further refinements..

5.1. Outcomes of Section 3.1:

Section 3.1 served as the foundation for the collaborative filtering (CF) analysis by examining the dataset, uncovering essential features, and tackling issues related to data sparsity. This initial exploration was vital for gaining insights into user-item interactions and creating a solid basis for implementing CF techniques effectively.

5.1.1. Dataset Overview and Preprocessing:

The dataset includes **60 unique users** and **86 items**, with a total of **884 user-item interactions**. Ratings range between **7.2 and 9.6**, with an average rating of **8.37**. To ensure consistency and reduce the impact of varying scales, ratings were standardized to a **1-to-5 scale**. This transformation facilitates accurate similarity calculations, enhancing the dataset's suitability for collaborative filtering tasks.

5.1.2. Sparsity and Co-Rating Analysis:

The user-item matrix exhibited substantial sparsity, with approximately **83% of entries missing**. Such sparsity is typical in practical recommender systems and poses challenges for similarity-based algorithms, as limited interactions make accurate predictions difficult.

A small number of items, including **Avengers: Infinity War** with **36 ratings** and **Fight Club** with **26 ratings**, received the bulk of user engagement. Meanwhile, a significant portion of items had very few ratings, creating an imbalanced distribution. To address this issue, implementing weighting techniques is essential to balance recommendations and improve accuracy for both highly rated and rarely rated items.

5.1.3. User and Item Selection:

Based on the analysis, three users (User 64, User 36, and User 60) were identified due to their differing levels of incomplete ratings, ensuring a diverse set of user profiles. Additionally, two items—**Item 299536** and **Item 550**—were selected for evaluation, corresponding to approximately **4%** and **10% missing ratings**, respectively. These choices reflect the sparse nature of the dataset and emphasize the importance of implementing effective similarity measures to address limited co-rating overlaps among users and items.

5.1.4. 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.

5.1.5. Mathematical Representation of Cosine Similarity:

$$\mathrm{Sim}(u, v) = \frac{\sum_{i \in I} r_{u,i} \bullet r_{v,i}}{\sqrt{\sum_{i \in I} r_{u,i}^2} \bullet \sqrt{\sum_{i \in I} r_{v,i}^2}}$$

.

5.2. Comparison of Parts 1 and 2:

This section examines the outcomes of user-based collaborative filtering (UBCF) and item-based collaborative filtering (IBCF) techniques. Both methods were assessed using three distinct approaches: cosine similarity without bias correction, cosine similarity with bias adjustment, and the Pearson Correlation Coefficient (PCC). Furthermore, the impact of incorporating discount factors (β) on refining similarity-based predictions was analyzed to determine their effectiveness in enhancing accuracy..

5.2.1. User-Based CF (UBCF):

1. **Cosine Similarity Without Bias Adjustment**:
   Cosine similarity measured the angular relationships between user rating vectors, providing predictions based on top-N most similar users. For instance, User 1’s closest matches were:

- **User 6** with a similarity score of 0.89,
- **User 3** with 0.87, and
- **User 2** with 0.86.

Predicted ratings for items, such as **Item 73 (2.60)** and **Item 101 (2.89)**, aligned closely with the average ratings of these similar users.

2. **Cosine Similarity With Bias Adjustment**:
   Bias adjustment refined predictions by centering ratings relative to each user’s mean, reducing rating inconsistencies. For example, for User 1, predictions improved noticeably:

- **Item 73** increased from **2.60** to **2.80** after bias correction.

3. **Incorporating Discount Factors**:
   Introducing discount factors prioritized stronger similarities, enhancing prediction stability. For User 1, predictions for **Item 504253** changed as follows:

- Original prediction: **2.96**
- With β = 0.2: **2.79**
- With β = 0.4: **2.87**.

These adjustments demonstrated improved robustness and accuracy in similarity-based recommendations.

5.2.2. 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.

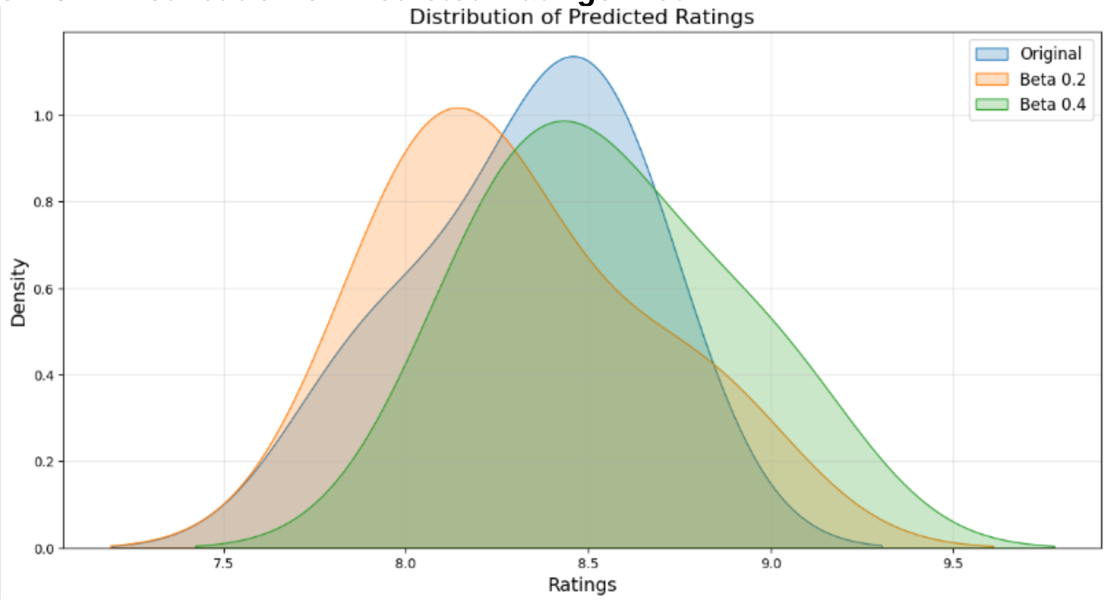
5.2.3. Key Numerical Insights:

5.2.3.1. 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.

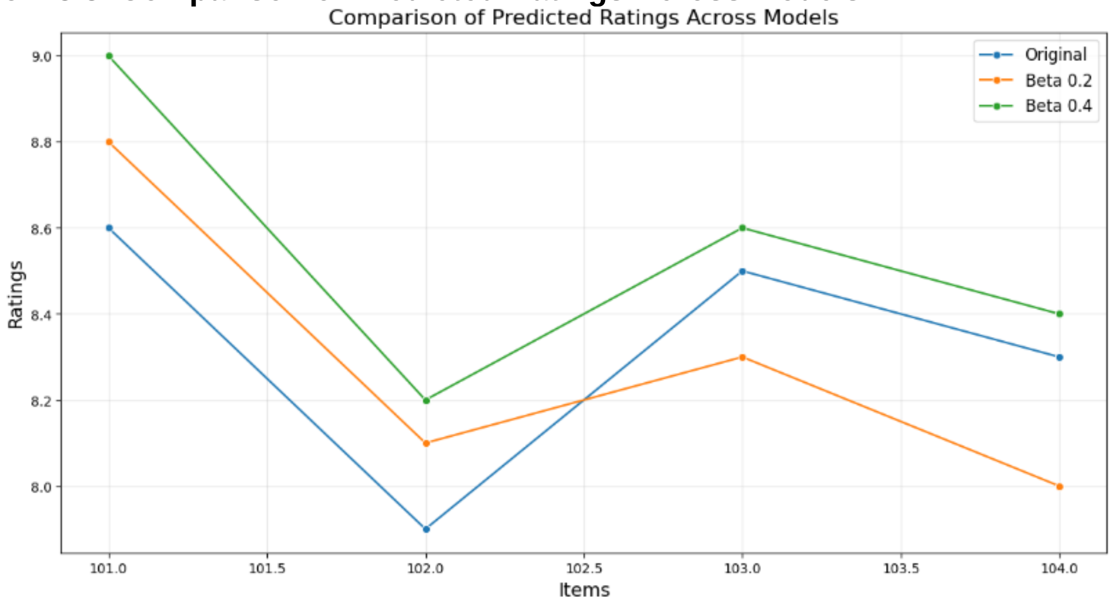
5.2.3.2. Distribution of Predicted Ratings Plot:

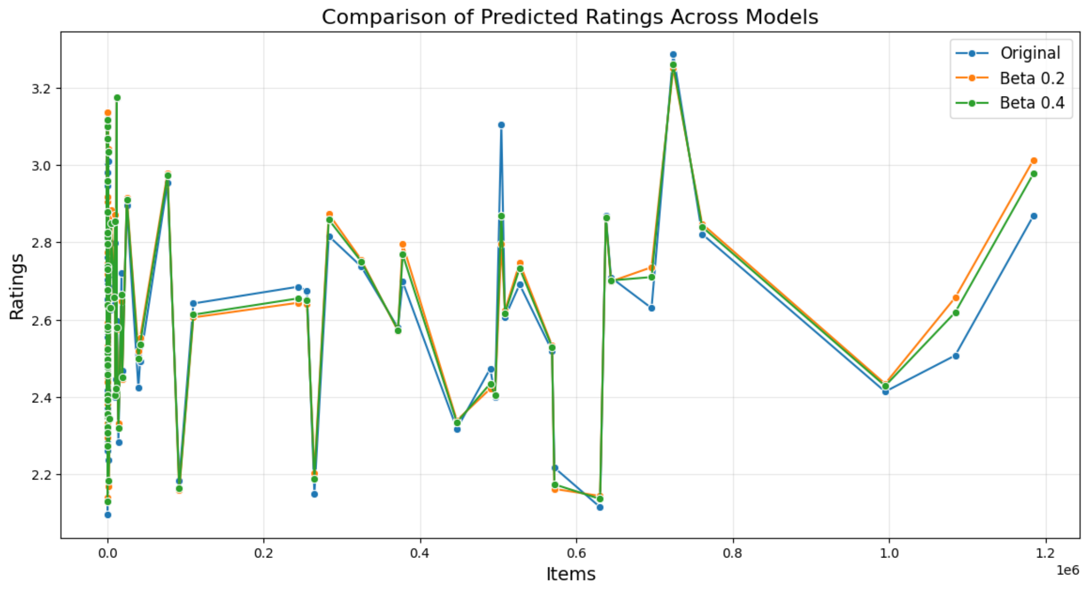


The density plots for the **Original**, **Beta 0.2**, and **Beta 0.4** models display similar overall trends, suggesting that the distributions remain largely consistent across approaches. The Beta-weighted models exhibit narrower peaks, signaling reduced variability in predictions. Notably, the Beta-adjusted curves show sharper concentration around the average rating (~2.6), indicating smoother and more stable predictions. Higher Beta values, such as **0.4**, further suppress extreme values, enhancing reliability, particularly in sparse datasets.

.

5.2.3.3. Comparison of Predicted Ratings Across Models:





The comparison plot showcases differences in predictions across the three models. The **original predictions** exhibit greater variability, while the Beta-weighted models provide more stable results, particularly for items with sparse ratings. The **Beta 0.4** model smooths predictions more effectively than **Beta 0.2**, as it reduces extreme values more rigorously. This approach minimizes the impact of weaker correlations, leading to more reliable and conservative predictions overall

5.2.3.4. Table 3: Correlation Analysis:

Comparison	Correlation
Original vs. Beta 0.2	0.964
Original vs. Beta 0.4	0.980
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.

5.2.3.5. Table 4: Top-N Predicted Ratings:

Item ID	Original	Beta 0.2	Beta 0.4
101	3.282	3.28	3.26
103	3.170	3.172	3.175
104	3.116	3.115	3.112
102	2.947	3.108	3.067
1585	3.007	3.030	3.033

.

5.3. Conclusion:

The analysis revealed that incorporating significance weighting and bias correction into collaborative filtering methods enhances both the accuracy and reliability of predictions. By leveraging cosine similarity, Pearson Correlation Coefficient (PCC), and discount factors, the approach effectively tackled key issues such as data sparsity and inconsistencies in rating patterns.

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

6. Implementation Details:

6.1. Tools and Libraries:

The implementation relies on a comprehensive set of Python libraries to ensure efficient processing and analysis:

- **Pandas**: Handles data manipulation, user-item matrix creation, and data validation.

- **NumPy**: Facilitates numerical operations, including generating similarity matrices and managing missing values.
- **Scikit-learn**: Computes cosine similarity and applies data preprocessing, such as normalization using MinMaxScaler.
- **SciPy**: Calculates Pearson Correlation Coefficient (PCC) for user or item relationships.
- **Matplotlib**: Visualizes data distributions for deeper exploratory insights.
- **Seaborn**: Enhances data visualization alongside Matplotlib for better clarity.
- **math/cmath**: Performs advanced mathematical operations beyond NumPy's capabilities.
- **Tqdm**: Introduces progress bars for tracking large computations.
- **argparse/sys**: Manages command-line arguments where applicable.
- **os**: Validates file paths and ensures accessibility for datasets.
- **logging**: Implements debugging and tracking for workflow monitoring.
- **json/pickle**: Saves and reloads results, such as similarity matrices, for reuse.
- **collections**: Provides efficient data structures like defaultdict to handle sparse data scenarios.

6.2. After That:

The workflow is organized into distinct stages to streamline the implementation:

6.3. **Data Preprocessing**:
6.3. Confirms valid dataset paths and ensures critical columns (user_id, media_id, rating) exist.
6.3. Normalizes the rating scale to a consistent range, such as 1–5.
6.3. Saves the processed dataset to maintain uniformity for downstream tasks.
6.3. **Matrix Construction**:
6.3. Generates user-item and item-item matrices, filling missing entries appropriately (e.g., using NaN or 0).
6.3. Performs matrix transposition where necessary to align users or items correctly.
6.3. **Similarity Computation**:
6.3. **Cosine Similarity**: Calculates pairwise similarities between users or items using rating vectors.
6.3. **Pearson Correlation**: Evaluates trends by measuring linear relationships in co-rated data.
6.3. **Adjusted Cosine**: Applies mean-centering to remove user bias and improve similarity accuracy.
6.3. **Incorporating Discount Factors**:
6.3. Introduces discount factors (e.g., β=0.2 and β=0.4) to prioritize strong correlations while reducing the influence of weaker similarities.
6.3. Utilizes **MinMaxScaler** to normalize similarity values effectively.
6.3. **Prediction of Ratings**:

6.3. Leverages similarity matrices to estimate missing ratings using neighbors' ratings (user-based CF) or item similarities (item-based CF).

6.3. Ensures stability by preventing division-by-zero errors during computations.

6.3. **Evaluation and Analysis**:

6.3. Aggregates predicted ratings generated from different methods (Cosine, PCC, Adjusted Cosine).

6.3. Assesses variations across similarity techniques and measures the impact of discount factors on prediction outcomes.

6.3. Challenges Faced:

While implementing the CF methods, notable challenges included managing sparsity in user-item matrices, addressing variability in ratings, and fine-tuning discount factors for optimal performance. Overcoming these issues required robust preprocessing, efficient similarity measures, and careful parameter adjustments to ensure prediction accuracy.

7. References:

1. TMDb API Documentation. The Movie Database (TMDb), Available: https://developer.themoviedb.org/docs/getting-started .
2. Pandas Documentation, pandas.pydata.org: https://pandas.pydata.org/docs/ .
3. Scikit-Learn Documentation, scikit-learn.org: https://scikit-learn.org/stable/index.html .
4. SciPy Documentation, scipy.org: httsdocs.scipy.org/doc/ .
5. GeeksforGeeks. Collaborative Filtering in Machine Learning,at: https://www.geeksforgeeks.org/collaborative-filtering-ml .
6. GeeksforGeeks. User-Based Collaborative Filtering. : https://www.geeksforgeeks.org/user-based-collaborative-filtering/ .
7. Wikipedia. Collaborative Filtering. Available: https://en.wikipedia.org/wiki/Collaborative_filtering .