

**AIE425 Intelligent recommender systems, Fall Semester 24/25**

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

**221101218, Ahmed Osama Mahmoud**

**All ratings have been adjusted to follow a 1-to-5 scale, providing consistency in our rating measurements.**

**Selected Active Users and Items**

**Target Users**

* U1 (user 2): 2 missing ratings, high engagement with 4 total ratings.
* U2 (user 5): 3 missing ratings, moderate rating history.
* U3 (user 16): 5 missing ratings but sparsity with only 1 rated item.

Target Items:

* I1 (movie\_1): ~4% missing ratings (19 total ratings).
* I2 (movie\_5): ~10% missing ratings (18 total ratings).

**2-D array with "No\_common\_users" in descending order and corresponding "no\_corated\_items"**:

[[19, 70],

[18, 53],

[18, 18]]

* **Description of the Curve:**
  + X-axis: Represents the movies (movie\_1 to movie\_6).
  + Y-axis: Represents the number of ratings each movie has received, calculated as the count of non-null ratings for each movie

**Summary of Thresholds:**

* **User [ID2]**:
  + **β = 1.8**, based on 30% of the 6 ratings they have made. To meet the threshold, a co-rating of at least 2 items is required.
* **User [ID5]**:
  + **β = 1.8**, based on 30% of the 6 ratings they have made. To meet the threshold, a co-rating of at least 2 items is required.
* **User [ID16]**:
  + **β = 0**, based on 30% of the 1 rating they have made. Since they need at least 1 co-rated item, the threshold is effectively 1.

**Summary of the Comparison of Part 1 and 2 (case study 1.1):**

Comparison of Outputs from Part 1 and Part 2

**Part 1: User Collaborative Filtering (Cosine Similarity)**

* User 2's Top 20% Neighbors:
  + Closest Users: 16, 18, 5
  + Cosine Similarity values: 1.0, 0.99396, 0.99388
* User 5's Top 20% Neighbors:
  + Closest Users: 13, 2, 9
  + Cosine Similarity values: 0.99999, 0.99388, 0.99224
* User 16's Top 20% Neighbors:
  + Closest Users: 1, 2, 3
  + Cosine Similarity values: 1.0, 1.0, 1.0

**Part 2: Discounted Similarity with Significance Weighting**

* User 2's Top 20% Neighbors:
  + Closest Users: 18, 14, 19
  + Discounted Similarity (DS) values: 0.7952, 0.7831, 0.7831
* User 5's Top 20% Neighbors:
  + Closest Users: 13, 9, 19
  + Discounted Similarity (DS) values: 0.6000, 0.5953, 0.5953
* User 16's Top 20% Neighbors:
  + Closest Users: 1, 2, 3
  + Discounted Similarity (DS) values: 0.2000, 0.2000, 0.2000

This comparison highlights how the discounted similarity in Part 2 adjusts the rankings of neighbors by considering the number of common ratings and applies a more refinedweighting mechanism, whereas Part 1 relies purely on raw cosine similarity without any consideration for the significance of the commonality

**Key Differences in the Results in part 1(case study 1.2):**

* **Similarity Values:**
  + In the first code, the cosine similarity tends to be higher, with values such as 0.98 for users 14 and 19 when compared to active user 2. This indicates a strong relationship based purely on rating patterns.
  + In the second code, the similarity is lower, with values like 0.78 for users 14 and 19. This reduction happens due to the discount factor adjusting the raw similarity based on the number of common ratings between users.
* **Active User 16:**
  + Both codes show 0.00 similarity for Active User 16 when compared to other users. This suggests that Active User 16 shares no common ratings with others, leading to zero similarity in both cases. However, the discounted similarity remains zero because there’s no commonality in ratings, which is expected.
* **Top Users:**
  + The first code selects the closest users based solely on the highest cosine similarity, which can be highly affected by a few highly rated items.
  + The second code adjusts the similarity based on the discount factor, which tends to select users with more common ratings. In cases of very few common ratings, the similarity value drops significantly, making the second code's results potentially more reflective of true user similarity.

**Conclusion:**

* First Code: Gives a direct, raw measure of user similarity based on the cosine similarity of their ratings. This approach works well when users have rated a lot of common items, but it might overstate the similarity between users who have rated only a few items in common.
* Second Code: Provides a more nuanced similarity measure by applying a discount factor based on the number of common ratings. This helps account for the sparsity of the data, and might better capture the "true" similarity when comparing users with fewer common ratings. However, it can also lead to much lower similarity scores, which may be useful for filtering out weak connections between users.

**Key Differences in the Results in part 1 (case study 1.3):**

* **Similarity Calculation:**
  + First Code**:** The first code computes the Pearson Correlation Coefficient (PCC) between users based on their ratings. This method calculates similarity purely based on how similarly users rate items, with no adjustments.
  + Second Code**:** The second code also computes Pearson CorrelationCoefficient, but it introduces a discount factor based on the number of common ratings between users. This discount factor reduces the similarity score if users have rated fewer items in common.
* **Selection of Top 20% Closest Users:**
  + First Code**:** It selects the top 20% of most similar users based solely on the unadjusted Pearson Correlation similarity values, without considering how many common ratings the users share.
  + Second Code**:** It adjusts the similarity scores using the discount factor, which reduces similarity scores for users with fewer common ratings. It then selects the top 20% of users with the highest discounted similarity.

**Summary of Differences in Results:**

* For Active User 2 and Active User 5, the results are similar between the two codes, as the top 3 closest users are identical, and the discount factor doesn't drastically alter the rankings.
* For Active User 16, both codes return the same result, as no users have rated items in common with Active User 16, resulting in zero similarity scores.

**Key Differences in the Results in part 2 (case study 2.1):**

* For Active User 2 and Active User 5, the results are similar between the two codes, as the top 3 closest users are identical, and the discount factor doesn't drastically alter the rankings.
* For Active User 16, both codes return the same result, as no users have rated items in common with Active User 16, resulting in zero similarity scores.
* The first code gives higher similarity scores in general since it doesn't adjust the similarity for the number of common ratings between users.
* The second code applies a discount factor to reduce the impact of users with fewer common ratings, which might lead to slightly lower similarity scores.

** Cosine Similarity Calculation:**

* First Code:
  + Calculates cosine similarity directly between the target items (movie\_1 and movie\_5) and all other items.
  + The similarity values are used as-is without any adjustments.
* Second Code:
  + Also calculates cosine similarity between items, but includes a Discount Factor (DF), which is based on the number of common ratings between the items.
  + A threshold is set for each movie (movie\_1 and movie\_5), which determines how many common ratings are needed for the discount to be applied.
  + The DF reduces the similarity score if the number of common ratings falls below the threshold.

 Discount Factor (DF):

* First Code: No discount factor is applied to the cosine similarities; all items are treated equally based on their raw similarity values.
* Second Code: Introduces a discount factor that adjusts the cosine similarity scores based on how many ratings are common between the target items and other items. The threshold value for each target item ensures that only items with a sufficient number of common ratings are considered similar.

 Top N Selection:

* First Code: Selects the top 25% of items closest to the target items, sorting based on cosine similarity and ensuring that at least one item is selected.
* Second Code: Selects the top 20% of items closest to the target items, using the discounted similarity values and ensuring at least one item is selected.
* **Key Differences in the Results in part 2 (case study 2.2):**

**Key Differences (top 20%):**

* **Cosine Similarity Calculation:**
  + Both codes calculate cosine similarity between items using the ratings matrix.
  + First Code**:** Directly uses the raw cosine similarity values to compute predictions.
  + Second Code**:** Uses cosine similarity values adjusted by a **Discount Factor (DF)**, which is calculated based on the number of common ratings and a threshold specific to each movie (e.g., movie\_1 has a threshold of 19 and movie\_5 has a threshold of 17).
* **Discount Factor (DF):**
  + First Code**:** No discount factor is applied to the similarity values. It simply uses the cosine similarity as is.
  + Second Code**:** The Discount Factor is applied to the cosine similarity between items. The DF is computed based on the number of common ratings between the target item and other items, and it scales the similarity by this factor. This can lead to a decrease in similarity if the number of common ratings is lower than the defined threshold for each movie.
* **Top N Selection:**
  + First Code**:** Selects the top 25% closest items based on the similarity values for each target item and uses these items to predict missing ratings.
  + Second Code**:** Selects the top 20% closest items based on the **discounted** similarity values and uses these items to predict missing ratings.
* **Prediction Calculation:**
  + Both codes compute predictions by using a weighted sum of ratings, where the ratings of similar items are weighted by their similarity scores.
  + If the denominator (the sum of similarities) is zero, both codes fall back to using the user's mean rating or the global mean rating.

**Key Differences (predictions):**

* **Cosine Similarity (First Code):**
  + In the first code, you calculate the cosine similarity between the target items and all other items based on mean-centered ratings.
  + The similarities are computed using the formula for cosine similarity, and the results are stored in a dictionary for each target item.
  + The target item itself is excluded from the comparison.
  + **Result**: The top 20% closest items are selected based purely on cosine similarity, which are movie\_3 for movie\_1 and movie\_4 for movie\_5, with high similarity values (0.9076 and 0.9082 respectively).
* **Discounted Similarity (Second Code):**
  + In the second code, the similarity scores are calculated using a discounted similarity approach, which involves a discount factor (though the exact discount factor is not shown in the provided code).
  + The discounted similarity is stored in a similarity info dictionary, where the similarity is adjusted by a discount factor, which can lower the similarity score if certain conditions are met.
  + Result: The top 20% closest items are selected based on discounted similarity, but the similarity values are much lower (0.1030 for movie\_1 and 0.1031 for movie\_5).
* **Key Differences in the Results in part 2 (case study 2.3):**
  + - **Cosine Similarity vs. Adjusted Cosine Similarity:**
* First Code**:** Uses standard cosine similarity, which calculates the similarity between items based on the raw ratings. It does not adjust for user biases, so the predictions might reflect the overall popularity or trend of an item.
* Second Code**:** Uses adjusted cosine similarity, which adjusts ratings by subtracting the user's mean rating for each item. This approach removes the bias of each user's overall rating tendencies (e.g., if a user tends to give higher or lower ratings overall). This method gives a better comparison of how users rate specific items, independent of their general rating tendencies.
  + - **2. Bias Adjustment in Predictions:**
* First Code**:** Adjusts the ratings by considering user bias and item bias (using the global mean and the user's mean ratings). This helps in accounting for overall tendencies when predicting ratings.
* Second Code**:** The bias adjustment is handled by the adjusted cosine similarity. Here, the rating is adjusted by subtracting the mean of the user's ratings, thus removing the general bias of the user before calculating the similarities. Additionally, it uses a (DF) that modifies the similarity based on how many common ratings exist, which can further affect the prediction.
  + - **3. Discount Factor (DF):**
* First Code**:** No discount factor is used. It simply calculates similarity based on the raw ratings and adjusts predictions using user and item biases.
* Second Code**:** Introduces a discount factor to reduce the similarity for items that have fewer common ratings with the target item. This means that items with fewer ratings in common will have their similarity values reduced, which may affect predictions significantly, especially for items with low overlap.
  + - **4. Threshold for Similarity:**
* First Code**:** There is no threshold for how similar items must be to be included in the top items for prediction.
* Second Code**:** Implements a movie threshold that adjusts the similarity by considering the number of common ratings. If a movie has fewer than a specified number of ratings, the similarity is discounted.
  + - **5. Predicted Ratings:**
* First Code**:** The predictions for movie\_1 and movie\_5 show consistent predictions, and the ratings for user\_5 are 4.0 for both movies. For movie\_5, predictions are also made for user\_11 and user\_16.
* Second Code**:** The predicted ratings are slightly different. For movie\_5, user\_5 has a rating of 3.0 (as opposed to 4.0 in the first code). For movie\_1, user\_5 still gets a rating of 4.0. The threshold and discount factor play a role in reducing the predicted ratings for items like movie\_5.
  + - **6. Impact of Discounted Similarity:**
* First Code**:** Since no discount factor is applied, the predictions may appear more stable and are heavily influenced by the cosine similarity.
* Second Code**:** The predictions vary more because of the discount factor applied to the similarity values. This makes the second set of predictions more sensitive to the number of common ratings, especially for users with fewer common ratings with others.

**3. Conclusion**

* + - Impact of Significance Weighting:

Our analysis demonstrates that significance weighting, particularly through the use of a Discount Factor (DF)**,** has a noticeable impact on recommendation quality. The introduction of a discount factor adjusts similarity scores based on the number of common ratings between items, emphasizing more reliable comparisons for items rated by a larger number of users. This weighting allows the system to prioritize more robust and trustworthy relationships between items, leading to:

* + - * Increased accuracy in item similarity by reducing the influence of less reliable or sparse data.
      * More relevant recommendations, as items with a higher number of common ratings are weighted more heavily, potentially leading to better user satisfaction.
      * Improved ranking stability, as the system is less affected by random or biased ratings from a few users, ensuring recommendations reflect general trends rather than outliers.

**Comparative Performance**

In comparing the two methods— **(PCC)** and **(DS)**—we observed that both methods identified similar items (e.g., movie\_3) as the closest match to the target items. However, in scenarios with a larger dataset or more sparse data (many missing ratings), the Discounted Similarity (DS) method tends to perform better, as it accounts for the number of common ratings and penalizes similarities based on sparse or less reliable data. On the other hand, Plain PCC may work well when the dataset is dense and ratings are consistent across users, but could be less robust in handling sparse or biased data.

* + - * Best Performing Method for Dense Data: Plain PCC might suffice, as it leverages all available ratings directly.
      * Best Performing Method for Sparse Data: Discounted Similarity (DS) outperforms by prioritizing reliable ratings, offering better robustness and higher recommendation relevance.

**Final Observations**

Through our analysis, we have learned several key lessons that can guide future recommender system designs:

* Importance of Handling Sparse Data: Properly weighting similarity scores to account for sparse data is crucial for maintaining recommendation quality, especially in large datasets with many missing ratings.
* Impact of Significance Weighting: Significance weighting significantly affects the quality and reliability of recommendations by ensuring that item similarities are based on more robust and trustworthy data.
* Adaptability and Flexibility: A successful recommender system must be adaptable, capable of adjusting similarity calculations and weighting mechanisms based on the specific characteristics of the data it is working with.

By implementing these insights and enhancements, we can create more accurate, relevant, and personalized recommendation systems that better serve users' needs and preferences.