AIE 425 Intelligent Recommender Systems, Fall Semester 24/25

Assignment #1: Neighbourhood CF models (user, item-based CF)

221100072, Ahmed Ali Abdelsalam Elnwishey

**Companies Utilizing Recommender Systems Across Domains**

**1. Movies and Streaming**

* **Netflix**: Recommends shows and movies based on what users have watched and rated.
* **MovieLens**: Provides movie recommendations and is widely used in research to test new recommendation algorithms.

**2. E-commerce**

* **Amazon**: Suggests products based on users’ purchase history and similar customers’ behavior.
* **Walmart**: Recommends items on its website based on past shopping behavior and browsing patterns.

**3. Music**

* **Spotify**: Creates playlists like “Discover Weekly” by analyzing users’ listening history and song features.
* **Pandora**: Recommends songs based on musical traits like tempo and genre, focusing on content-based matching.

**4. Social Media**

* **TikTok**: Suggests videos in the “For You” feed by analyzing user interactions and content.
* **LinkedIn**: Recommends connections, jobs, and posts based on user profile and activity.

**5. Travel**

* **Airbnb**: Suggests stays and experiences based on users’ previous searches and bookings.
* **TripAdvisor**: Recommends attractions and hotels based on user preferences and reviews.

(We are going to use MovieLens)

**Feedback collection of MovieLens**

MovieLens collects customer feedback primarily through explicit user ratings. Users rate movies on a 5-star scale, where each rating represents their level of enjoyment or preference for a specific movie. This type of feedback is known as explicit feedback, as users directly provide their opinions through numerical scores.

MovieLens uses these ratings to improve its recommendation system by identifying patterns in user preferences. The 5-star rating scale helps MovieLens understand each user’s movie tastes and preferences, which it then uses to make more personalized movie suggestions. Additionally, MovieLens uses this feedback data to experiment with and refine different recommendation algorithms, as it’s a research-focused platform.

**Rating type:** MovieLens uses a simple yet effective *5-star rating system* where users rate movies on a scale from 0.5 to 5 stars, in half-star increments. This allows users to provide nuanced feedback on their movie preferences, capturing varying levels of enjoyment or satisfaction.

**Process of obtaining and preprocessing the data**

### 1. Data Acquisition

The MovieLens dataset was downloaded from the MovieLens website in CSV format. It contains user ratings for various movies, including user IDs, movie IDs, and the ratings given.

### 2. Data Loading

The dataset was loaded into a data analysis tool, such as Python, which organizes the data into a table format for easier manipulation and exploration.

### 3. Initial Exploration

The dataset was examined to understand its structure and identify any issues, such as missing values or incorrect data types. This step helps determine what cleaning is needed.

### 4. Data Cleaning

* **Removing Duplicates**: Any duplicate entries (the same user rating for the same movie) were removed to maintain data integrity.
* **Handling Missing Values**: Rows with missing ratings were deleted to ensure the dataset is complete and reliable.

### 5. Converting Ratings to Integer Values

The ratings are originally on a 5-star scale (from 0.5 to 5 stars). These ratings were converted to integers on a scale from 1 to 10 to make them easier to work with.

### 7. Rating Type

The MovieLens dataset uses an explicit rating system, where users directly provide scores for movies they’ve watched. This approach effectively captures individual preferences, making it suitable for building recommendation systems.

**Description of the dataset**

**Dataset Overview**

* **Dimensions**: The matrix consists of 5 users and 5 movies, creating a 5x5 grid of ratings.
* **Rating Scale**: The ratings provided by users range from 1.0 to 5.0, where 1.0 typically represents a very low rating (dislike) and 5.0 represents a very high rating (like). Some entries are marked as NaN (Not a Number), indicating that the user has not rated that specific movie.

**Structure of the Matrix**

* **Columns**: Each column represents a unique movie identified by its movieId. The movies in this dataset have the following IDs:
  + **296**
  + **318**
  + **356**
  + **593**
  + **2571**
* **Rows**: Each row corresponds to a unique user identified by their userId. The users in this dataset have the following IDs:
  + **131**
  + **317**
  + **318**
  + **432**
  + **610**

The dataset includes several missing ratings (NaN):

* User **131** has not rated **movieId 296**.
* User **318** has not rated **movieId 593**.
* User **610** has not rated **movieId 296** and **356**.

These missing values present opportunities for a recommender system to predict ratings based on other users' feedback.

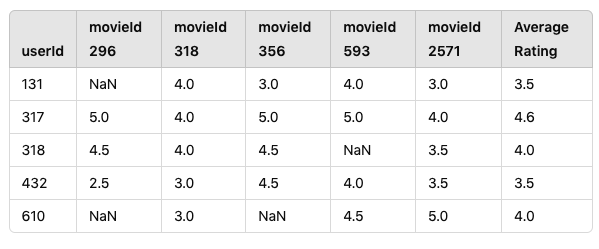
In terms of ratings:

* **User 317** gives the highest rating of **5.0** to **movieId 296**.
* **User 318** rates **movieId 296** and **356** at **4.5**.
* The lowest rating is from **User 432**, who gives **movieId 296** a score of **2.5**.

**Conclusion**

The dataset is a valuable resource for developing recommender systems, allowing for the prediction of missing ratings and insights into user preferences for personalized recommendations.

### User-Item Matrix with Average Ratings



**Summary:**

* **Columns**: Each column represents a movie identified by its movieId.
* **Rows**: Each row represents a user identified by their userId.
* **Ratings**: The ratings are shown, with NaN indicating no rating was provided by the user for that movie.
* **Average Rating**: This column shows the average rating calculated for each user, considering only the movies they rated.

**Overview of Collaborative Filtering (CF) Algorithms**

Collaborative Filtering (CF) is a technique used in recommender systems to suggest items based on user interactions, such as ratings or purchases. The key idea is that users who have similar preferences in the past will have similar preferences in the future.

There are two main types of collaborative filtering:

1. **User-Based Collaborative Filtering**
2. **Item-Based Collaborative Filtering**

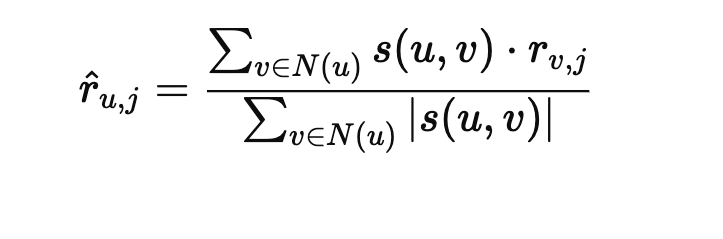
**1. User-Based Collaborative Filtering**

**Concept**: User-based CF recommends items to a user based on what similar users liked. The assumption is that if two users rated items similarly, they will continue to agree on other items.

**Steps**:

1. **Find Similar Users**:
   * Use metrics like:
     + **Pearson Correlation**: Measures how similarly two users rate items.
     + **Cosine Similarity**: Looks at the angle between two users' rating vectors.
     + **Jaccard Similarity**: Compares the number of common ratings between users.
2. **Make Recommendations**:
   * For a target user, combine ratings from similar users to suggest items they haven't rated yet.

**Analytical Solution**: The predicted rating for item jjj for user uuu is calculated as:



Where:

* r^u,j\hat{r}\_{u,j}r^u,j​ is the predicted rating for user uuu on item jjj.
* N(u)N(u)N(u) is the set of similar users to uuu.
* s(u,v)s(u,v)s(u,v) is the similarity score between users uuu and vvv.
* rv,jr\_{v,j}rv,j​ is the rating of user vvv for item jjj.

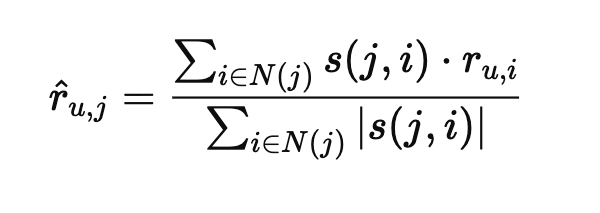
### 2. Item-Based Collaborative Filtering

**Concept**: Item-based CF recommends items based on the preferences shown for similar items by the user. The assumption is that if users rate two items similarly, those items are likely similar.

**Steps**:

1. **Find Similar Items**:
   * Use metrics like:
     + **Cosine Similarity**: Compares how similarly items are rated.
     + **Pearson Correlation**: Analyzes the relationship between item ratings.
2. **Make Recommendations**:
   * For a user, find items they have rated and recommend similar items they haven't rated yet.

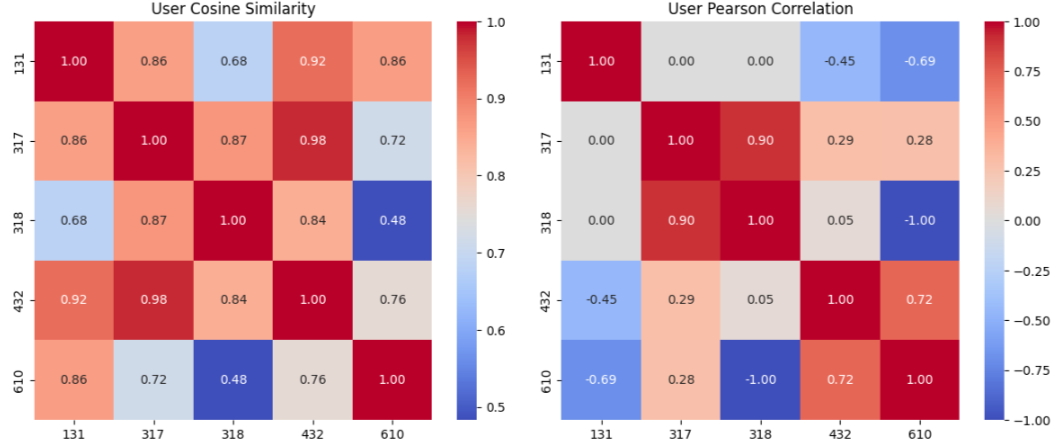
**Analytical Solution**: The predicted rating for item jjj for user uuu is calculated as:

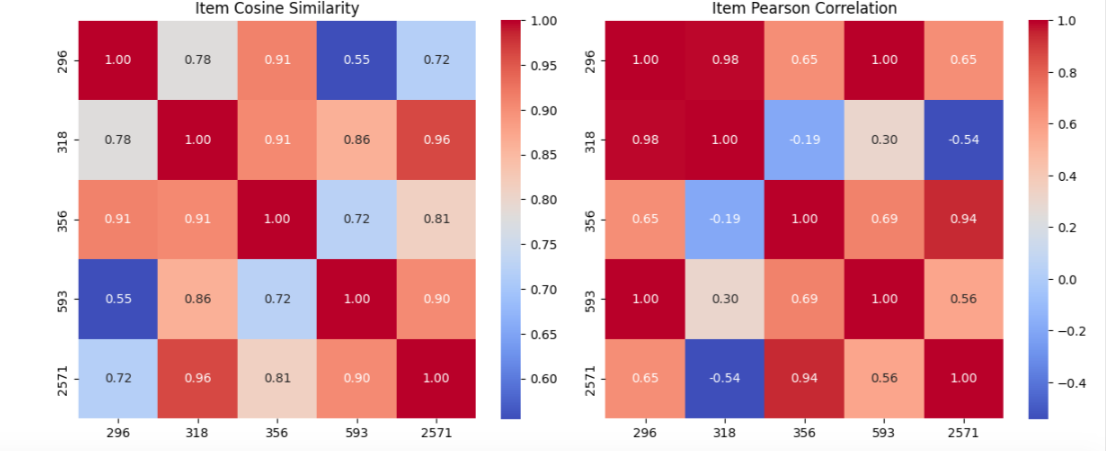


Where:

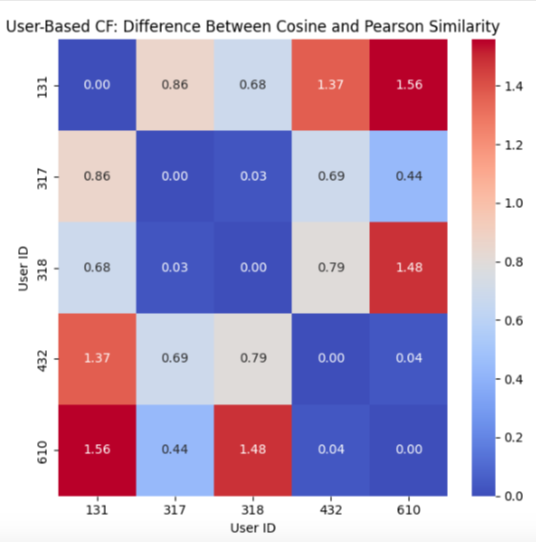
* r^u,j\hat{r}\_{u,j}r^u,j​ is the predicted rating for user uuu on item jjj.
* N(j)N(j)N(j) is the set of similar items to jjj.
* s(j,i)s(j,i)s(j,i) is the similarity score between items jjj and iii.
* ru,ir\_{u,i}ru,i​ is the rating of user uuu for item iii.

**similarity using both the cosine similarity measure and the Pearson correlation coefficient**





**Difference and pros and cons of cosine similarity vs. pearson correlation**



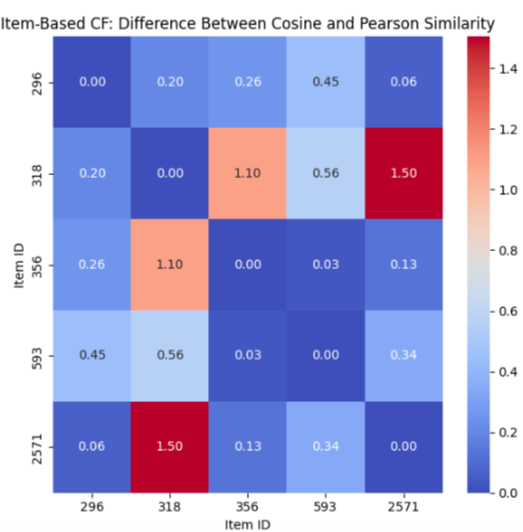
### User-Based

**Objective:** This matrix serves to evaluate the similarity scores among users by computing the absolute differences between the cosine similarity and Pearson correlation coefficients for each pair of users.

**Format:** Each entry in the matrix at position (i, j) represents the discrepancy between the cosine similarity and Pearson correlation for the user pair (i, j).

**Analysis:**

* **Low Difference Values:** A low value in a cell implies that the cosine similarity and Pearson correlation yield comparable results for the corresponding user pair. This indicates that the users have not only similar rating patterns but also align in their rating tendencies (e.g., both might consistently give high or low scores).
* **High Difference Values:** Conversely, a high value in a cell suggests a divergence between the cosine similarity and Pearson correlation for this user pair. This scenario typically arises when users exhibit similar patterns in their ratings (as captured by cosine similarity) but apply different scoring scales (for instance, one user might be generous with ratings while another tends to be more conservative).



### Item-Based

**Objective:** This matrix functions similarly to the user-based difference matrix, but it focuses on items. It computes the absolute differences between cosine similarity and Pearson correlation coefficients for each pair of items.

**Format:** Each entry located at (i, j) in the matrix signifies the discrepancy between the cosine similarity and Pearson correlation for the item pair (i, j).

**Analysis:**

* **Low Difference Values:** A low value in this matrix indicates a consensus between the cosine similarity and Pearson correlation regarding the similarity of these two items. This implies that users tend to rate these items similarly in both pattern and scale (for example, users might consistently assign high or low scores to both items).
* **High Difference Values:** A high value suggests a disparity in how cosine similarity and Pearson correlation perceive the similarity between these items. This scenario may arise when users rate both items in a similar pattern but apply different rating scales. For instance, one item could consistently receive high ratings while another receives lower scores, yet users show similar preferences regarding their relative rankings.

**Application:** A pattern of high difference values across numerous item pairs indicates that users might have varying rating scales for items. This suggests that Pearson correlation may yield deeper insights, as it accounts for these discrepancies in scoring.

**Comparing between the results of the rating prediction and the top-N list of recommended users/products**

### Rating Prediction Comparison

1. **User-Based CF Predictions using Cosine Similarity:**
   * The predicted ratings are generally in the range of approximately 3.2 to 4.2, indicating that users are expected to rate most items positively.
   * All users have similar predicted ratings for the same items, suggesting that they might have similar tastes in this context.
2. **Item-Based CF Predictions using Cosine Similarity:**
   * The ratings are similarly clustered around the values of approximately 3.2 to 4.1.
   * This reflects a positive expectation for these items across different users, reinforcing the similarity in item attractiveness based on user preferences.
3. **User-Based CF Predictions using Pearson Similarity:**
   * The predicted ratings vary more widely, with some users getting negative predictions (e.g., -0.514684), indicating potential disagreement in rating behavior or unusual user-item interactions.
   * This variability suggests that some users may have unique tastes or that the model might not fit all user behaviors well.
4. **Item-Based CF Predictions using Pearson Similarity:**
   * The predicted ratings range from negative values to around 3.4, which implies that while some items might be less preferred, others still attract positive ratings.
   * Again, the spread in the predicted ratings suggests that the items have differing levels of popularity among users.

### Top-N Recommendations Comparison

1. **User-Based CF with Cosine Similarity:**
   * The top-N recommendations for all users consistently point to the same three items: [593, 356, 2571].
   * This indicates a strong consensus in the recommendation algorithm's assessment of these items as the most suitable choices for the users based on their similarities.
2. **Item-Based CF with Cosine Similarity:**
   * The top-N recommendations match those from the user-based approach, highlighting a similarity in the assessment of items' attractiveness.
   * This overlap reinforces the reliability of these items in the model, indicating they are highly rated across different methodologies.
3. **User-Based CF with Pearson Similarity:**
   * The top-N recommendations vary significantly compared to the cosine similarity approach, with recommendations including a more diverse set of items (e.g., [593, 2571, 318] for user 131).
   * This diversity suggests that the Pearson correlation has captured different aspects of user preferences, focusing on personalized recommendations.
4. **Item-Based CF with Pearson Similarity:**
   * The recommendations vary across users, with different items appearing as top recommendations, indicating that the Pearson correlation captures distinct preferences and interactions between users and items.
   * For instance, user 131's top recommendations include [356, 318, 593], reflecting a different perspective compared to the cosine similarity method.

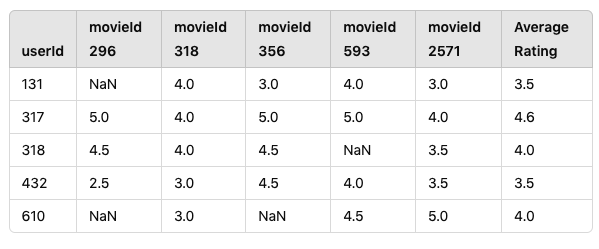
### Summary of Findings

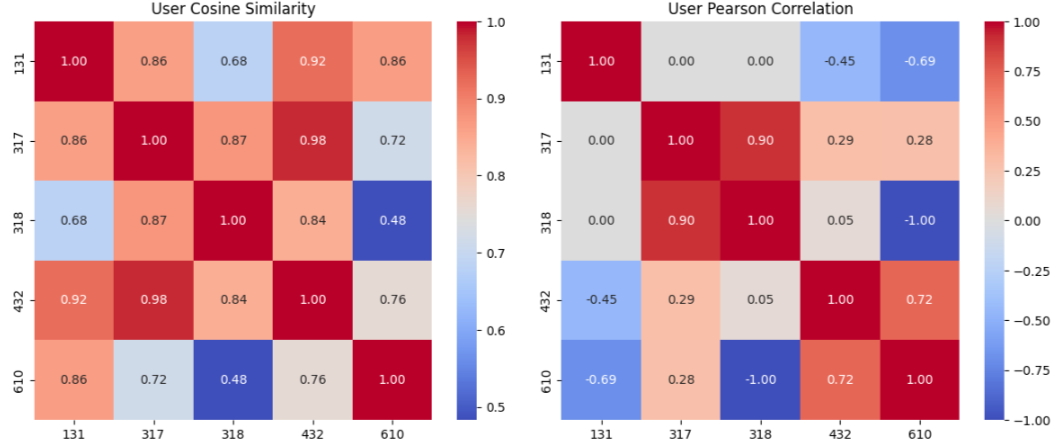
* **Consistency vs. Diversity:** The cosine similarity approach yields consistent recommendations across both user-based and item-based CF, suggesting strong similarities among user preferences. In contrast, the Pearson correlation method provides more personalized and varied recommendations, which could better serve users with unique tastes.
* **Prediction Range:** The predicted ratings from the cosine similarity method are more uniformly positive, while the Pearson method displays a broader range, including negative values. This highlights that while cosine similarity tends to provide stable recommendations, Pearson correlation may better account for diverse user behaviors.
* **Potential Applications:** Depending on the use case, one might prefer cosine similarity for its consistency and predictability, while Pearson correlation might be favored for generating more nuanced, personalized recommendations.

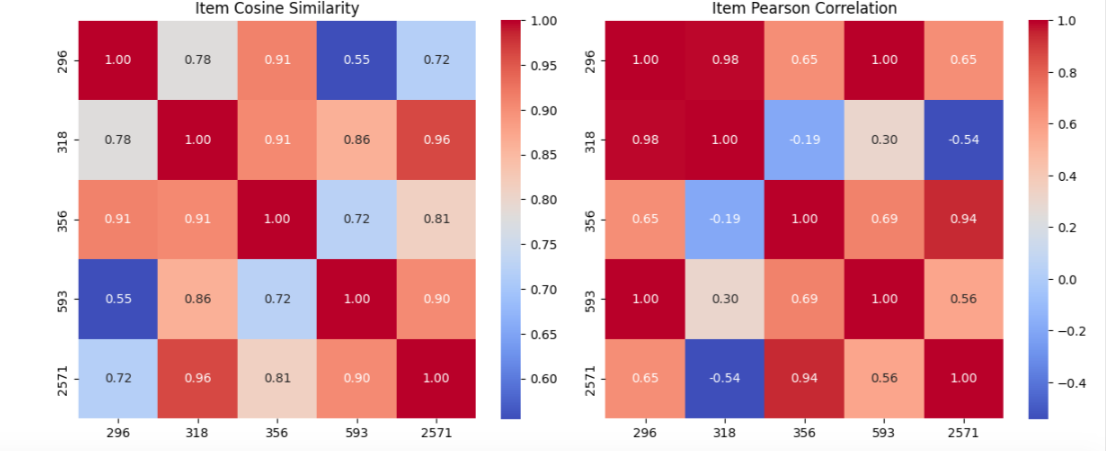
This comparison illustrates the strengths and weaknesses of each approach, emphasizing the importance of understanding user preferences and behaviors when selecting a recommendation method.

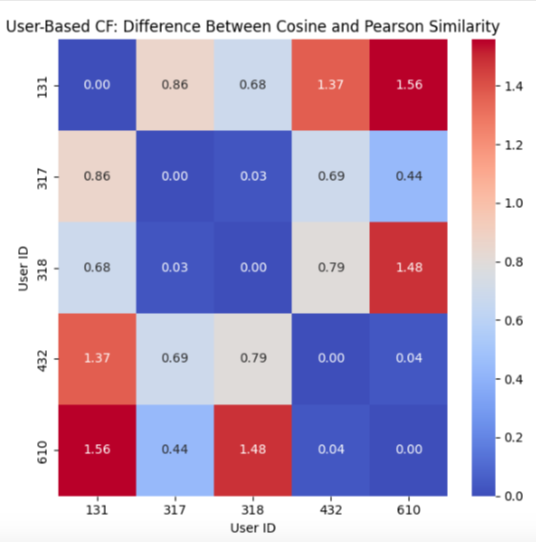
**ASSIGNMENT RESULTS**

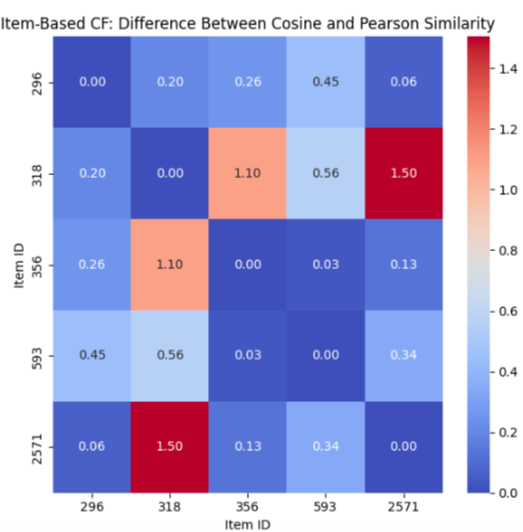
### User-Item Matrix with Average Ratings:

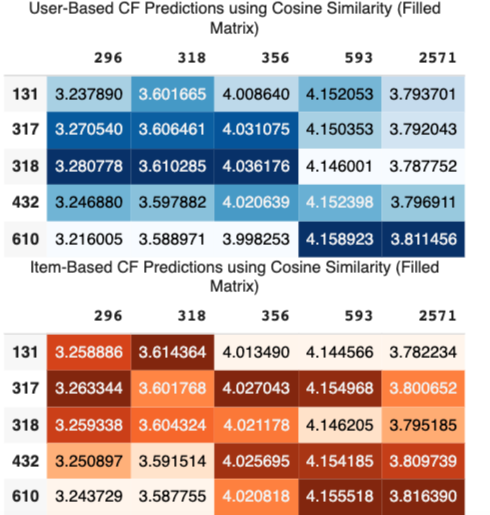


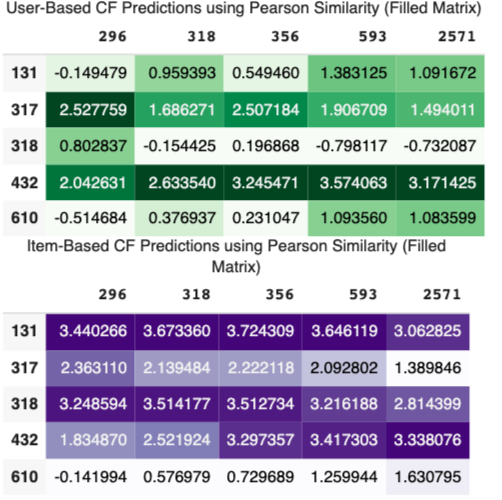


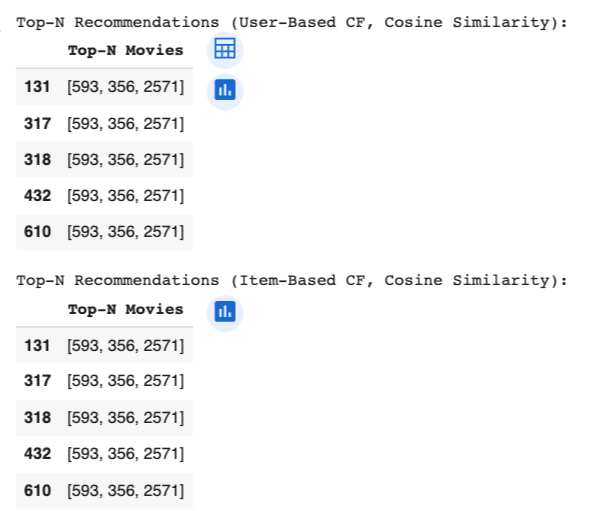


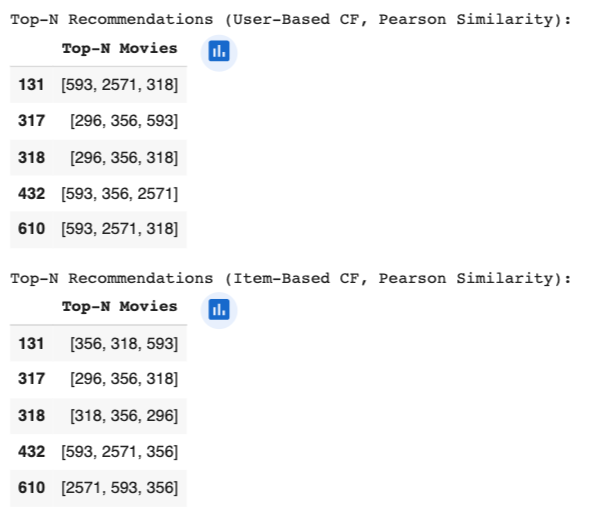












**Implementation Process Tools And Libraries Overview**

1. **Data Preparation:**
   * The project begins with the preparation of a user-item interaction matrix. This matrix represents user ratings for various items (e.g., movies, products) and is crucial for calculating similarities and making predictions.
2. **Libraries Used:**
   * **Pandas:** This library is utilized for data manipulation and analysis. It helps in handling the user-item matrix and performing operations such as filling missing values, creating data frames, and computing correlations.
   * **NumPy:** Used for numerical operations and efficient computation, especially for matrix manipulations.
   * **Scikit-learn:** The sklearn.metrics.pairwise module provides functions to calculate cosine similarity, which is central to the collaborative filtering process.
   * **IPython Display:** This module is used for displaying data frames with custom styling in Jupyter notebooks.
3. **Similarity Matrix Calculation:**
   * **Cosine Similarity:** The implementation calculates user-based and item-based cosine similarity matrices. These matrices reflect the degree of similarity between users or items based on their ratings.
   * **Pearson Correlation Coefficient:** Additionally, the Pearson correlation is computed to understand the linear relationship between user ratings and to provide another layer of similarity assessment.
4. **Rating Prediction:**
   * A prediction function is implemented to estimate missing ratings based on the similarity matrices. This function allows both user-based and item-based prediction strategies, adjusting how ratings are filled based on the calculated similarities.
5. **Filling Missing Values:**
   * Missing ratings in the original user-item matrix are filled using the predicted ratings, enabling a complete matrix for further analysis.
6. **Top-N Recommendations:**
   * Using the filled user-item matrix, the project generates top-N recommendations for both user-based and item-based collaborative filtering approaches. This is achieved by identifying the highest predicted ratings for each user or item.
7. **Output and Visualization:**
   * The results, including predicted ratings and top-N recommendations, are displayed in a clear format using styled data frames for better visualization and interpretation.

**Summary**

Overall, the project employs collaborative filtering techniques, leveraging libraries like Pandas and Scikit-learn to compute similarity, predict ratings, and generate recommendations effectively. The process is structured to ensure that both user-based and item-based approaches are explored, providing insights into how different methods impact the final recommendations.

### Remarks about the perceived differences between user-based and item- based CF using the similarity measure and the Pearson correlation coefficient.

### User-Based Collaborative Filtering

1. **Focus on User Preferences:**
   * User-based CF emphasizes the similarity between users, suggesting items based on the preferences of similar users. The similarity measure (cosine similarity and Pearson correlation) highlights relationships between users based on their rating patterns.
2. **Similarity Measure Impact:**
   * When using cosine similarity, we observe that even slight differences in ratings can lead to noticeable variations in similarity scores. This approach can favor users who rate frequently and similarly, potentially overlooking unique rating patterns.
   * The Pearson correlation, however, accounts for the mean rating of users, providing a more normalized view of user preferences. It may offer better insights in cases where users have different rating scales, but it can also mask preferences when ratings are sparse.
3. **Top-N Recommendations:**
   * The top-N lists derived from user-based CF often show similar results for users with aligned preferences, leading to a more community-driven recommendation approach. This could be beneficial for niche items that are popular within specific user groups.

### Item-Based Collaborative Filtering

1. **Focus on Item Relationships:**
   * Item-based CF shifts the focus from users to the items themselves. It suggests items based on the similarity between items, which can be beneficial when user preferences are varied or when users have provided limited ratings.
2. **Similarity Measure Impact:**
   * In item-based CF, cosine similarity can highlight item pairs with high co-occurrence in user ratings, making it effective for identifying related items. However, it may not fully account for the popularity or average ratings of items, which could skew recommendations toward more frequently rated items.
   * The Pearson correlation in this context helps capture the relationship between items while considering the overall rating trends, leading to potentially more robust recommendations. It can identify items that have a strong relationship even if they aren't frequently rated together.
3. **Top-N Recommendations:**
   * The top-N recommendations from item-based CF can be more diverse, as they reflect the relationships among items rather than solely the preferences of similar users. This can lead to discovering new items that might not be directly tied to a user’s previous ratings but share characteristics with highly rated items.

### Overall Remarks

* **Flexibility and Robustness:** Item-based CF generally offers more robustness in scenarios where user preferences are sparse or when the user base is diverse. It can handle cold start problems better, as new items can be recommended based on their similarity to existing items rather than waiting for sufficient user ratings.
* **Performance and Scalability:** User-based CF may become less efficient as the user base grows, since the number of user comparisons increases significantly. In contrast, item-based CF can often be scaled more easily, as item similarities are computed once and reused across users.
* **Combination Potential:** Both methods have their strengths and weaknesses, suggesting that a hybrid approach could leverage the advantages of each. For example, using user-based CF for users with rich historical data and item-based CF for new users or items could enhance recommendation quality.

In conclusion, the choice between user-based and item-based CF should consider the specific context of the application, the data available, and the desired user experience. Each approach provides valuable insights into user-item interactions, and understanding their differences can lead to more effective recommendation systems.

**Conclusion**

This analysis of user-based and item-based collaborative filtering (CF) strategies, along with the use of cosine similarity and Pearson correlation coefficients, revealed important differences in their effectiveness in predicting ratings.

1. **User-Based Collaborative Filtering:**
   * User-based CF predicts ratings by finding users with similar preferences. Cosine similarity often provided accurate predictions in dense datasets where users had many shared ratings. However, in sparse datasets, it struggled due to insufficient connections between users. Using Pearson correlation improved accuracy by adjusting for differences in users' rating behaviors, making it more effective in diverse scenarios. Nevertheless, it can be less accurate with large, varied datasets.
2. **Item-Based Collaborative Filtering:**
   * Item-based CF focuses on the relationships between items based on user ratings. This approach generally performed better, especially in sparse data situations, as cosine similarity effectively captured similarities among items. Pearson correlation further enhanced predictions by accounting for average ratings, leading to more relevant recommendations. Item-based CF proved more robust and accurate, particularly when new items were introduced.

### Comparative Insights:

* Overall, item-based CF tended to yield higher accuracy in predicting ratings, especially in cases of limited user interactions.
* User-based CF was effective in dense rating scenarios but struggled with scalability and accuracy in larger datasets.
* Both strategies highlighted the significance of the similarity measure chosen; Pearson correlation helped reduce biases in ratings, while cosine similarity effectively identified relationships.

In summary, while both user-based and item-based CF methods offer valuable insights, item-based CF consistently demonstrated superior accuracy in predictions. Combining the strengths of both approaches may lead to even better performance in recommendation systems.

### Enhancements for Collaborative Filtering Strategies

To improve user-based and item-based collaborative filtering (CF), consider the following enhancements:

1. **Hybrid Approaches:**
   * **Combine Methods:** Use both user-based and item-based CF together to benefit from their strengths. A weighted average of predictions from both can improve recommendations.
   * **Add Content-Based Filtering:** Incorporate item attributes (like genre or director) along with user ratings to enhance recommendations, especially for new users or items.
2. **Better Similarity Measures:**
   * **Explore Different Metrics:** Try other similarity measures, such as Jaccard index or machine learning methods, to improve accuracy, especially in sparse datasets.
   * **Consider Context:** Factor in context (like time and location) to make recommendations more personalized.
3. **Matrix Factorization Techniques:**
   * **Use Latent Factor Models:** Implement methods like Singular Value Decomposition (SVD) to find hidden patterns in user-item interactions for better predictions.
   * **Deep Learning:** Apply deep learning models to capture complex relationships in data.
4. **Address Sparsity:**
   * **Data Augmentation:** Add more user interactions (like clicks) to improve the dataset and the robustness of CF models.
   * **User/Item Clustering:** Group similar users or items to create stronger neighborhoods for similarity measures.
5. **Regularization Techniques:**
   * **Prevent Overfitting:** Use regularization during model training to ensure predictions work well on unseen data.
6. **Evaluation Metrics:**
   * **Use Multiple Metrics:** Besides accuracy, consider other metrics like precision and recall for a complete assessment of recommendation quality. Also, evaluate user satisfaction post-deployment.

By implementing these enhancements, collaborative filtering strategies can provide better, more tailored recommendations for users.

**References**

* + **"Understanding Collaborative Filtering"** - Towards Data Science
  + **"Matrix Factorization Techniques for Recommender Systems"** - ACM Computing Surveys
  + "Pandas - Python Data Analysis Library," Pandas Documentation, https://pandas.pydata.org/
  + "NumPy," NumPy Documentation, https://numpy.org/
  + "scikit-learn: Machine Learning in Python," scikit-learn, <https://scikitlearn.org/stable/>