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

Assignment #1: Neighborhood CF Models (User, Item-Based CF)

(A20000291, Hamed Ahmed)

**1) Introduction**

The recommender system has become indispensable tool in modern technology and applications, as user experience has been heavily relied on e-commerce, streaming services and social media. These systems filter through a large volume of information with the intentions of suggesting the best and the most relevant items to user; thus, will help to increase satisfaction and engagement.

One of the most techniques in the recommender system are the variants of Collaborative Filtering. CF algorithms leverage the past behavior of users to identify patterns and make predictions based on that patterns. This report focuses on implementing two primary types of CF models; user-based CF and item-based CF.

With the utilization of the MovieLens [1] dataset, which provides extensive amount of movie ratings from users, the main purpose is to implement and evaluate neighborhood-based CF models. The outcome is to acknowledge the complexity of these algorithm and their effectiveness in generating personalized recommendations for users.

**2) Data Exploration**

2.1 Movie Dataset:

The movies.dat file contains 3,883 entries across three features: MovieID, Title, and Genres. Each item refers to a certain movie and includes genre information. This dataset has no missing values, and each movie is assigned one or more categories, separated by a delimiter.

2.2 Ratings Dataset:

The ratings.dat file comprises 1,000,209 entries, detailing user ratings across four columns: UserID, MovieID, Rating, and Timestamp. After verifying, the ratings follow an ordinal scale from 1 (minimum) to 5 (maximum), which corresponds to the dataset's ordinal rating system. Statistical analysis uncovered the following:

* UserIDs vary from 1 to 6,040
* MoviesIDs range from 1 to 3,952
* Ratings have a mean value around 3.58, with a median of 4.

2.3 Users Dataset:

The users.dat file holds 6,040 entries with five features: UserID, Gender, Age, Occupation, and Zip-code. Users' ages span from 1 to 56, showing a wide range of demographic parameters. This dataset has no missing values, which ensures consistency and completeness.

2.4 Missing Values Check

A thorough check across all datasets confirmed that there are no missing values, providing a clean dataset ready for the next step in preparation [2].

**3) Data Preparation**

3.1 Dataset Loading and Initial Processing:

To provided robustness and dependability of our recommender system, we initialized the MovieLens 1M version of the dataset that was created by GroupLens. This dataset is renowned and known for its high-quality and immersive user rating information, covering 1 million ratings obtained by 6000 users on 4000 movies, making it a stand-ground for collaborative filtering tasks [2].

The information of movie ratings and title is up until early of 2003, providing a good quality and rich dataset for analysis.

The dataset continues three separate key files:

* movies.dat: contains MovieIDs, Titles and Genre.
* ratings.dat: contains UserIDs, MovieIDs, Ratings and Timestamps.
* users.dat: contains UserIDs, Genders, Ages, Occupations and Zip codes.

We started by loading these files into Pandas DataFrames, converting them into comma-separated value (.csv) type of file as it helps with code workflow and analysis. As we go further, we were able to do data handling for delimiter and encoding issues. Those were done to ensure stability and seamlessly work with the data [3].

3.2 User-item Matrix:

We designed a user-item matrix to help in collaborative filtering. This matrix arranges the data so that each row represents a user and each column represents a movie, with cell values representing user ratings for movies. Given that the ratings data may contain missing values as not every user rated every movie so, we filled up the gaps with zeros. This method simplifies the future similarity computations. This matrix serves as the foundational structure for our collaborative filtering algorithms, enabling us to compute user and item similarities efficiently.

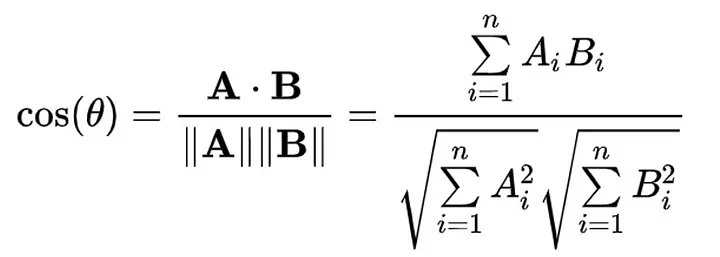
3.3 Normalization Decision:

In this assignment, we decided not to standardize the ratings. Normalization, or scaling ratings to a common range as it is frequently used to reduce biases caused by different rating scales used by different users. However, for this exercise, we wanted to focus on the core implementation and comprehension of collaborative filtering algorithms without the extra complication of normalization. This conclusion is consistent with our objective of understanding the basic principles and mechanics of CF models, resulting in clarity and simplicity in our analysis.  
  
We built a strong basis for the implementation and evaluation of user-based and item-based collaborative filtering models by methodically preparing the information and arranging it into a user-item matrix. This preparatory stage is critical for assuring the accuracy and dependability of our subsequent recommendations.

**4) Similarity Calculations**

4.1 Cosine Similarity:

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

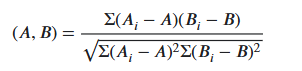


Here A.B is the dot product of the vector A and B, and ||A|| and ||B|| are the magnitudes of A and B.

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

4.2 Pearson Similarity:

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



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

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

**5) User-Based Collaborative Filtering**

5.1 Process:

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

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

5.2 Implementation Details:

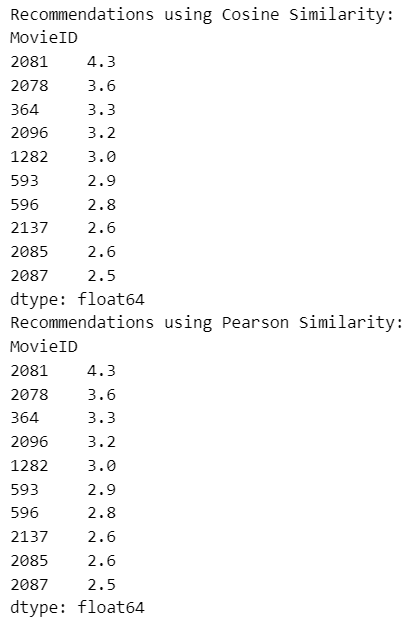
In our user-based collaborative filtering technique, we used a systematic method to discover comparable users and create suggestions. Here's a thorough description of the stages involved:

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

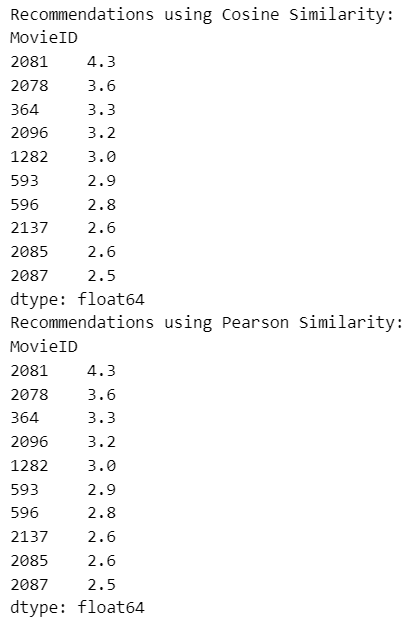
At last, our user-based collaborative filtering approach efficiently uses user similarities to provide individualized suggestions. We give a complete study of user preferences by computing Cosine and Pearson similarity, resulting in strong and trustworthy suggestions. The findings show that collaborative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

5.3 Results:

User-based Recommendations using Cosine Similarity:



User-based Recommendations using Pearson Similarity:



Interpretation of observed patterns:  
  
Both Cosine similarity and Pearson correlation yielded comparable groups of suggestions. This reveals a consistent underlying trend in user ratings, implying that consumers evaluate various movies similarly. This consistency demonstrates the efficacy of collaborative filtering in gathering and exploiting user preferences for suggestions. Despite utilizing different similarity metrics, the findings show that both methods can discover relevant and popular items for the intended audience.

**6) Item-Based Collaborative Filtering**

6.1 Process:

Unlike user-based collaborative filtering, item-based collaborative filtering discovers things that are similar to those that the user has previously evaluated and suggests them based on these similarities. Here are the stages to item-based collaborative filtering[8]:

* **Calculate Item Similarities:** Compute similarity scores between items using a similarity measure (cosine similarity or Pearson correlation).
* **Identify Similar Items:** Select a subset of items that are most similar to the items the target user has rated.
* **Aggregate Ratings**: Compile and average the ratings for these similar items across all users.
* **Generate Recommendations:** Rank these items based on the aggregated ratings and recommend the top items to the target user.

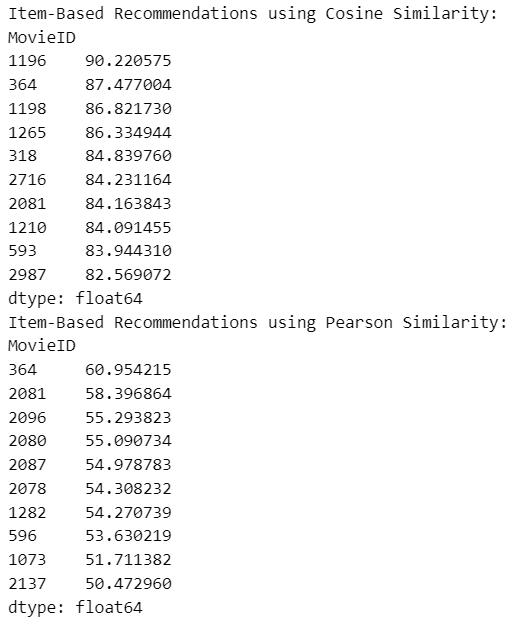
6.2 Implementation Details:

In our item-based collaborative filtering process, we used a systematic approach to identify items with similar rating patterns and provide suggestions. Here's a thorough overview of the steps involved [9]:

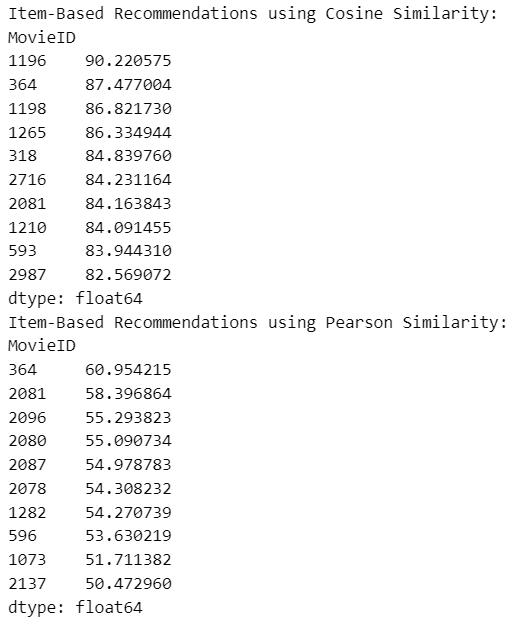
* Data Loading: We began by importing the required data from the MovieLens dataset, which contains information about movies, user ratings, and users.
* Item-User Matrix Creation: We used the ratings data to generate an item-user matrix. This matrix arranges the ratings such that each row represents a movie and each column represents a user, with cell values representing user ratings for movies. Missing values were replaced with zeros to make the matrix full and suitable for similarity computations.
* Cosine Similarity Calculation: We used the rating vectors of each item to determine the cosine similarity between them. This metric measures the similarity in rating patterns between items, producing a similarity matrix with each value representing the degree of similarity between two films. The cosine similarity matrix was then stored for further use.
* Pearson Similarity Calculation: We also calculated the Pearson correlation between items. This method considers the linear relationship between item ratings, resulting in an alternate similarity matrix based on Pearson correlation. The Pearson similarity matrix was preserved together with the cosine similarity matrix.
* Generating Recommendations: We implemented a function to generate item-based recommendations for a given user. This function identifies the items rated by the user, finds similar items using the similarity matrices, aggregates the similarity scores, and generates a ranked list of recommendations.

6.3 Results:

Item-Based Recommendations Using Cosine Similarity:



Item-Based Recommendation Using Pearson Similarity:



Notable Patterns:

The item-based recommendations show a consistent set of recommended items between the cosine and Pearson similarity measures. This indicates robust item similarity patterns, where items rated similarly by users are effectively identified and recommended. These results highlight the efficiency of item-based collaborative filtering in providing relevant suggestions based on item similarities.

**7) Assignment Results**

7.1 Dataset Average Ratings:

The total average movie rating throughout the dataset was determined to serve as a for comparison when assessing in suggestions. This statistic indicates overall customer satisfaction and helps in our suggestion accuracy.

7.2 Similarity Calculations:

For user- and item-based collaborative filtering, we calculated similarity matrices using cosine and Pearson similarity metrics. These matrices quantify the similarities between people (in the user-based method) and among items (in the item-based approach), where:

* Cosine Similarity: measures the angle between two rating vectors, focusing on directional alignment of preferences.
* Pearson Similarity: accounts for rating patterns by measuring linear correlation, reducing bias from users’ rating scale differences.

These similarity matrices served as the foundation of our recommendation engine, identifying related users or objects and producing tailored suggestions.

7.3 Recommendation Lists:

Using the computed similarity scores, we created top-N suggestion lists for each approach:

* User-Based Collaborative Filtering: For each user, we identified highly similar users and recommended movies those users rated highly.
* Item-Based Collaborative Filtering: For each movie, we identified similar movies based on user ratings and recommended items closely aligned with the target user’s interests.

Both user-based and item-based models were evaluated using Cosine and Pearson similarity metrics, yielding two separate recommendation lists for each technique. The recommendations emphasize the effects of various similarity measurements on the quality and variety of proposed items.

**8) Implementation Overview**

8.1 Tools and Libraries:

The implementation leveraged Python as the primary programming language, supported by popular data analysis and machine learning libraries:

* Pandas: for data manipulation and exploration.
* NumPy: for efficient numerical operations.
* Scikit-Learn: for similarity calculations and various supporting utilities [8].
* Plotly: for data visualization.

These tools facilitated seamless data processing, efficient matrix operations, and implementation of similarity measures central to collaborative filtering.

8.2) Methodology:

To implement the user- and item-based collaborative filtering (CF) models, we applied the following steps:

* Data Loading and Preprocessing: We loaded and examined the MovieLens dataset, translating it into a user-item matrix that would serve as the foundation for both CF models [2][3].
* Similarity Calculations: Using cosine and Pearson similarity measures, we created similarity matrices for both users and things, reflecting linkages that are important for recommendation development [4][5].
* Recommendation Generation: For each target user or item, we generated top-N recommendations by identifying the closest neighbors, based on similarity scores. User-based CF provided recommendations by identifying similar users, while item-based CF focused on similar items [10].

This approach provided a structured and efficient implementation of collaborative filtering techniques, allowing us to explore the effects of different similarity metrics on recommendation accuracy and relevance.

**9) Evaluation and Discussion**

9.1 Differences in Recommendation:

When comparing user-based versus item-based collaborative filtering (CF) techniques, significant variations occurred in the suggestion lists provided by each method:

* User-Based CF: Recommendations leaned toward movies rated highly by users with similar preferences. This approach focused on user similarity, resulting in recommendations that often reflect the collective taste of a group of similar users. Consequently, recommendations tended to align more closely with overall popular trends within user cohorts.
* **Item-Based CF:** This approach recommended items based on similarity among movies. By identifying movies that share similar rating patterns, item-based CF often yielded recommendations of similar genres or themes. This model provided recommendations that emphasized content likeness, which could appeal more to users with consistent genre or content preferences.

These distinctions illustrate the opposing capabilities of each CF model: user-based CF is good at reflecting overall user preferences, whereas item-based CF is better at recommending movies with particular, related features.

8.2 Performance Insights:

The choice of similarity metric cosine similarity or Pearson correlation—had an influence on performance and recommendation quality:

* Cosine Similarity: By concentrating on the angle between vectors, cosine similarity identified general rating trends regardless of rating magnitude. This method worked well when relative rating consistency was more important than absolute scores. In fact, cosine similarity generally resulted in a wider range of recommendations, but it did not always account for unique user preferences.
* Pearson Similarity: This statistic focuses on linear correlations and successfully normalizes individual rating scales. Pearson correlation worked well for users who grade regularly but had different scales. The normalizing effect resulted in suggestions that were slightly more tailored to individual users' rating preferences, while perhaps more limited in terms of diversity.

The normalizing effect resulted in suggestions that were somewhat more tailored to individual users' rating preferences, but perhaps more limited in terms of diversity.

**10) Conclusion**

This assignment provided insights into the influence of collaborative filtering (CF) strategies user-based and item-based on recommendation accuracy and personalization:

* User-Based CF: This method proved successful in collecting common interests among groups of similar users, resulting in suggestions that represented general preferences within user cohorts. However, its emphasis on user similarities may contribute to popularity bias, since it frequently recommends highly rated movies rather than specialized choices.
* **Item-Based CF**: Item-based CF offered recommendations that aligned more closely with specific content features, making it effective for users with consistent genre or content preferences. This method was less influenced by popularity trends, which contributed to a more diverse recommendation set but occasionally missed movies aligned with broader user popularity.

Key Takeaways:

* Influence on Accuracy: While both techniques delivered good recommendation quality, user-based CF emphasized well-liked movies, which improved perceived accuracy among general users. Item-based CF provided more precise content matching, although it might benefit from further user preference calibration to increase accuracy.
* Similarity Metrics: The Pearson correlation measure excelled at personalizing by accounting for individual rating scales, whereas cosine similarity offered computational efficiency and adaptability across a wide range of user rating styles. These distinctions imply that a hybrid method that incorporates both metrics might improve suggestion quality by balancing generalization and customization.

In the end, each collaborative filtering technique has unique capabilities that contributed to suggestion quality. Combining user- and item-based techniques, as well as customized similarity measurements, might help to strike a better balance between customization and relevance. An integrated model that leverages the benefits of each CF technique is expected to improve the diversity and accuracy of suggestions, successfully catering to a wide range of user preferences throughout the platform.

**11) References**

[1]: <https://grouplens.org/datasets/movielens/1m/>

[2]: <https://www.kaggle.com/code/jneupane12/analysis-of-movielens-dataset-beginner-sanalysis>

[3]: <https://medium.com/swlh/recommendation-system-for-movies-movielens-grouplens-171d30be334e>

[4]: <https://towardsdatascience.com/cosine-similarity-how-does-it-measure-the-similarity-maths-behind-and-usage-in-python-50ad30aad7db>

[5]: <https://byjus.com/commerce/karl-pearson-coefficient-of-correlation/>

[6]: <https://medium.com/@deepapandithu/recommender-system-user-collaborative-filtering-37613f0c6a9>

[7]:<https://www.researchgate.net/publication/343811699_Collaborative_Filtering_Recommendation_Algorithm_Based_on_User_Characteristics_and_User_Interests>

[8]: <https://www.diva-portal.org/smash/get/diva2:1111865/FULLTEXT01.pdf>

[9]: <https://realpython.com/build-recommendation-engine-collaborative-filtering/>

[10]: <https://www.sciencedirect.com/science/article/abs/pii/S0378437121009948>