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

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

**221101376, Mazen Adel Ali El-Fayoumi**

**Core idea.**

The core idea of the assignment was to get data from the internet using web scrapping not a ready-made dataset. Then preprocessing this data in order to fit into a user-item matrix and implementing all the collaborative filtering techniques that we learned to this point from user-based CF to item-based CF, implementing multiple similarity techniques like Cosine similarity, Adjusted cosine and Pearson correlation and finally predicting the potential rating of an item.

**The solution to all steps in Section 2.3**

1. There are various companies that use recommender systems in many different fields, for example Amazon is the leading compony in E-commerce as it uses item to item collaborative filtering to recommend products by finding new items that are similar to the ones that the user have already bought or viewed. On the other hands Netflix and Spotify are some of the top companies in offering streaming services, they both utilize collaborative filtering. Lasty, we have Facebook which recommends friends and content based on user interactions and their interests using graph based collaborative filtering.

2. My selected company is “Rotten Tomatoes” which is an entertainment review collector that gathers ratings and reviews from general audiences to professional critics on movies and TV shows. Rotten tomatoes is one of the most trusted websites out there for movies an TV shows ratings and used by many.

3. Rotten tomatoes collects ratings through two main categories which are critic reviews and audience reviews. These ratings are displayed as a “Tomatometer” score for the critics and “Audience Score” for the general users. The tomatometer which is the one that I used in my data collection is calculated based on the percentage of critics giving a positive review but the types of data in the critics sections varies from interval based ratings in which some ratings are out of 10, some are out of 5 and some are out of 4.There are also some reviewers that use ordered categorial values that I have encountered that where represented by letters like “A” and “B-“ In general the Tomatometer score is essentially binary due to it seeing each critic’s review as either “positive” or “negative” and the final score is a percentage of positive reviews. The audience score follows the star system on a scale of 0.5 to 5 but I didn’t use it.

4. I collected the required data and preformed multiple preprocessing steps varying from code to manual approach in order to reach the final results of obtaining integer values.

5.

* Picked 5 movies of the “Harry potter saga” due to it having a huge fan base resulting in a higher probability of finding same users across different movies.
* Manually searched for similar users across these 5 different movies which resulted in me finding 5 similar users.
* Used an google chrome extension called “Instant Data Scraper” which was recommended to me by a friend in order to make the process of collecting the data in each page easier.
* Extracted the data into 5 csv files and saved to my desktop.
* Created a simple code to remove all the unnecessary columns, keeping the most important 2 columns which where the names of the users and there rating of the movie, added an extra column in which I added the movie name for clarity. And finally downloaded these 5 new preprocessed csv files.
* Used the contact function in the pandas library the combine all of the 5 csv files in order to create the full dataset.
* Manually created an excel file containing the users’ names, movies’ names and their ratings.
* Manually created another excel file removing the users’ names to user’s IDs, changed the names of the movies to movie\_1 and movie\_2 and so on for simplicity and finally converted the ratings into a scale of 10. There were ratings that were already on the scale of 10 that were left unchanged in this process. The ones that were on the scale of 4 were multiplied by 2.5 and the others that were on a scale of 5 were multiplied by 2. Finally the ratings that were ordinal in the form of “A” and “B” required searching on the internet to find out that for example “B-“was in the range of 80-82 so I picked the highest and divided it by 100 resulting in 8.2 and I also replaced the missing value with NAN.
* The final step that I almost missed and had to review the lecture to make sure was converting these float values into integer values by rounding them up resulting into the final user-item matrix.

6.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| user\_id | movie\_1 | movie\_2 | movie\_3 | movie\_4 | movie\_5 |
| 1 | 8 | 8 | 8 | 8 | NAN |
| 2 | 9 | 6 | 8 | 6 | 8 |
| 3 | 8 | 8 | 7 | 6 | 4 |
| 4 | 10 | 8 | 8 | 8 | 8 |
| 5 | 10 | 8 | 10 | 8 | 10 |

7. this dataset represents a user-item matrix where row corresponds to a unique user (user\_id) and each column represents a specific movie (movie\_1 to movie\_5). Each cell contains ratings on a scale of 10 based on the feedback of rotten tomatoes where a higher number indicates a greater satisfaction with the movie. There is one missing value (NAN) as the user one did not rate the movie number 5, this missing value will be predicted using collaborative filtering methods. Overall, this dataset provides a good foundation for implementing collaborative filtering algorithms due to having similar users rating the same items.

8. we are going to calculate the average by dividing the sum of ratings by the count of rated movies:

* User\_1 mean = = 8
* User\_2 mean = = 7.4
* User\_3 mean = = 6.6
* User\_4 mean = = 8.4
* User\_5 mean = = 9.2

9. Collaborative Filtering is a method that is used in recommendation systems that makes predictions about a user’s preferences based on preferences of similar users (user-based CF) or based on previous items that the user was interested in or like (item-based CF) to make recommendations.

Steps in user-based CF (analytical solution):

* Computes similarity between users by identifying a target user, then find other users with similar rating patterns, usually uses Cosine similarity and Pearson correlation.
* Select the top K users with the highest similarity to the target user.
* Generates predictions of the target user’s rating by averaging the ratings of K nearest neighbors for that item.
* Formula:

Steps in item-based CF (analytical solution)

* Computes similarities between items based on user ratings. Similarity measures that are user are Cosine Similarity or Adjusted Cosine Similarity.
* Identify the top K items that are most similar.
* Predicts a user’s rating for a target item based on their previous ratings for similar items.
* Formula:

10. The target in this case is (user\_1)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| user\_id | movie\_1 | movie\_2 | movie\_3 | movie\_4 | movie\_5 | Mean | Cosine |
| 1 | 8 | 8 | 8 | 8 | NAN | 8 | 1 |
| 2 | 9 | 6 | 8 | 6 | 8 | 7.4 | 0.984 |
| 3 | 8 | 8 | 7 | 6 | 4 | 6.6 | 0.993 |
| 4 | 10 | 8 | 8 | 8 | 8 | 8.4 | 0.994 |
| 5 | 10 | 8 | 10 | 8 | 10 | 9.2 | 0.9938 |

User-based CF:

We will start by computing the Cosine similarity:

* User (2,1) = = 0.984
* User (3,1) = = 0.993
* User (4,1) = = 0.994
* User (5,1) = = 0.9938

Then we will commence with the Pearson Correlation:

By subtracting the ratings from the mean, we have prepared the matrix for Pearson correlation.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| user\_id | movie\_1 | movie\_2 | movie\_3 | movie\_4 | movie\_5 | Mean | Cosine | Pearson |
| 1 | 0 | 0 | 0 | 0 | NAN | 8 | 1 | 1 |
| 2 | -1.6 | 1.4 | -0.6 | 1.4 | -0.6 | 7.4 | 0.984 | 0 |
| 3 | -1.4 | -1.4 | -0.4 | 0.6 | 2.6 | 6.6 | 0.993 | 0 |
| 4 | -1.6 | 0.4 | 0.4 | 0.4 | 0.4 | 8.4 | 0.994 | 0 |
| 5 | -0.8 | 1.2 | -0.8 | 1.2 | -0.8 | 9.2 | 0.9938 | 0 |

User (2,1) = = 0

Same goes for the rest of the steps due to having zero for a nominator

User (3,1) = 0

User (4,1) = 0

User (5,1) = 0

Now for the Item based CFs:

We will use the cosine similarity:

As we did before we will subtract the ratings from the mean to get this matrix

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| user\_id | movie\_1 | movie\_2 | movie\_3 | movie\_4 | movie\_5 | Mean |
| 1 | 0 | 0 | 0 | 0 | NAN | 8 |
| 2 | -1.6 | 1.4 | -0.6 | 1.4 | -0.6 | 7.4 |
| 3 | -1.4 | -1.4 | -0.4 | 0.6 | 2.6 | 6.6 |
| 4 | -1.6 | 0.4 | 0.4 | 0.4 | 0.4 | 8.4 |
| 5 | -0.8 | 1.2 | -0.8 | 1.2 | -0.8 | 9.2 |
| Cosine (5, j) | |  | | --- | | -0.343 |  |  | | --- | |  | | |  | | --- | | -0.799 |  |  | | --- | |  | | |  | | --- | | 0.037 |  |  | | --- | |  | | -0.014 | 1 |  |

* Adjusted Cosine (5,j) , j = 1 to 4
* Adjusted cosine (1,5) = = −0.343
* Adjusted Cosine (2,5) = −0.799
* Adjusted Cosine (3,5) = 0.037
* Adjusted Cosine (4,5) = −0.014

13.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| user\_id | movie\_1 | movie\_2 | movie\_3 | movie\_4 | movie\_5 | Mean | Cosine |
| 1 | 8 | 8 | 8 | 8 | NAN | 8 | 1 |
| 2 | 9 | 6 | 8 | 6 | 8 | 7.4 | 0.984 |
| 3 | 8 | 8 | 7 | 6 | 4 | 6.6 | 0.993 |
| 4 | 10 | 8 | 8 | 8 | 8 | 8.4 | 0.994 |
| 5 | 10 | 8 | 10 | 8 | 10 | 9.2 | 0.9938 |

User based CFs:

Cosine similarity prediction:

Pred (user\_1, movie\_5) = 8+ 8.8

Pearson correlation prediction:

Pred (user\_1, movie\_5) = 8+ 8.8

Item based CFs:

Pred (u,t) =

14.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Metric | |  | | --- | | User-Based CF |  |  | | --- | |  | | | Item-Based CF | | --- |  |  | | --- | |  | |
| Rating Prediction for Movie 5 | 8.8 | 7.51 |
| Top-N Recommendation | Movie 5 recommended based on similar users | Movies 1-4 recommended based on item similarity |

User-based CF: yields to more positive predictions when there are strong user similarities. Even if the similarity is low.

Item-based CF: provide a more cautious predictions when item similarity is lacking which may cause the prediction to be lower.

15. the results are under” Assignment Results”

16. We applied both user-based and item-based collaborative filtering (CF) methods to Rotten Tomatoes data to understand their impact on recommendation predictions. After gathering and standardizing ratings from various scales to a 10-point scale, we computed similarities using cosine similarity and Pearson correlation for both users and items. For user-based CF, User 1's predicted rating for Movie 5 was positive (8.8) based on User 5's high rating for the same movie, demonstrating how similar users’ preferences can positively influence predictions. Item-based CF, however, resulted in a predicted rating (7.51) for User 1 on Movie 5, since Movie 5 was dissimilar to other movies User 1 rated highly. This difference highlights how user-based CF focuses on peer influence, often resulting in higher predictions when user similarities are strong. In contrast, item-based CF prioritizes item characteristics, offering more cautious predictions when item similarity is low, as it assumes users prefer items similar to those they previously enjoyed. For top-N recommendations, user-based CF recommended Movie 5 based on peer preferences, while item-based CF suggested Movies 1-4, which aligned more closely with User 1’s past ratings. In summary, user-based CF works well in peer-driven recommendation environments, while item-based CF excels in content-driven settings, where item similarity is critical. This comparison demonstrates the strengths and limitations of each method in different contexts.

17. The implementation process of our collaborative filtering system involved several key steps, from using Instant Data Scraper for collecting our data to utilizing Python’s data science libraries to handle data preprocessing, similarity calculation, and recommendation generation. We began with data preprocessing using Pandas to load and clean the data, handle missing values, and convert ratings to a uniform scale. NumPy was essential for numerical operations and matrix manipulations, which are foundational to collaborative filtering calculations. For similarity measures, we employed scikit-learn for cosine similarity calculations and SciPy for Pearson correlation, both of which help identify user and item similarities that guide the recommendation process. Finally, we developed user-based and item-based collaborative filtering functions to predict ratings and provide Top-N recommendations, using the computed similarity matrices to offer targeted recommendations based on either user preferences or item characteristics. Each tool and library was chosen to ensure efficient and accurate data processing, similarity computation, and recommendation generation.

18. User-based and item-based collaborative filtering (CF) differ significantly in how they leverage similarity, particularly when using cosine similarity and Pearson correlation.

User-based CF relies on identifying similarities between users, aiming to recommend items that a target user’s “peer group” has liked. This approach works well in social recommendation environments, as it assumes that users with similar past behaviors will have similar preferences. Cosine similarity is effective here for highlighting general alignment between users, but it doesn’t account for differences in rating scales or tendencies. Pearson correlation, on the other hand, is beneficial in user-based CF as it centers ratings around each user’s mean, which helps reveal true patterns in user preferences by adjusting for individual biases.

Meanwhile Item-based CF looks at similarities between items to find products or content that are alike, assuming users prefer items similar to those they previously enjoyed. This approach often yields more stable recommendations, especially for content-rich domains like movies or books, because item characteristics and relationships remain relatively stable over time. Using cosine similarity in item-based CF can sometimes overlook nuances, as it treats each rating independently of a user’s overall rating style. Pearson correlation is advantageous in item-based CF because it centers ratings, which can better capture how items are similarly perceived across a range of users. Overall, user-based CF is generally better for socially-driven recommendations, while item-based CF is ideal for content-driven environments. The choice of similarity measure also matters: Pearson correlation often provides more refined recommendations by adjusting for biases, while cosine similarity offers a straightforward approach to identifying basic patterns.

19. In conclusion, user-based and item-based collaborative filtering each influenced prediction accuracy in distinct ways. User-based CF, by focusing on similar users, tended to produce higher predictions when peer ratings were strong, as it assumed aligned preferences within peer groups. This approach worked well when user similarities were high, leading to positive predictions like the high rating for Movie 5. Item-based CF, however, prioritized item similarities, resulting in more conservative predictions when items were dissimilar to those the user liked, as seen with the low rating for Movie 5. This approach provided stable, cautious recommendations aligned with a user’s past preferences. Together, these strategies show how user-based CF suits socially-driven recommendations, while item-based CF enhances accuracy for content-driven suggestions by sticking closely to item characteristics.

20. An improvement would be to enhance the quality of the data, ensuring consistent and comprehensive ratings, which would lead to more accurate similarity calculations and better recommendation precision.

**Assignment Results**

* User\_1 mean = = 8
* User\_2 mean = = 7.4
* User\_3 mean = = 6.6
* User\_4 mean = = 8.4
* User\_5 mean = = 9.2
* User (2,1) = = 0.984
* User (3,1) = = 0.993
* User (4,1) = = 0.994
* User (5,1) = = 0.9938

User (2,1) = = 0

* Adjusted Cosine (5,j) , j = 1 to 4
* Adjusted cosine (1,5) = = −0.343
* Adjusted Cosine (2,5) = −0.799
* Adjusted Cosine (3,5) = 0.037
* Adjusted Cosine (4,5) = −0.014

Pred (user\_1, movie\_5) = 8+ 8.8

Pred (u,t) =

**References:**

[1] Pandas Documentation, "Data manipulation and handling missing data in pandas," https://pandas.pydata.org/docs/.  
[2] NumPy Documentation, "Numerical operations and array manipulations in NumPy," <https://numpy.org/doc/>.  
[3] Scikit-Learn Documentation, "Machine learning algorithms and utilities in scikit-learn," https://scikit-learn.org/stable/documentation.html.  
[4] Real Python, "Building a Recommendation Engine in Python," Real Python Tutorial, <https://realpython.com/build-recommendation-engine-collaborative-filtering/>.  
[5] DataCamp, "Recommender Systems in Python: Collaborative Filtering Tutorial," DataCamp Tutorial, <https://www.datacamp.com/tutorial/recommender-systems-python>.  
[6] Scikit-Learn Documentation, "Cosine Similarity Function," <https://scikit-learn.org/stable/modules/generated/sklearn.metrics.pairwise.cosine_similarity.html>.  
[7] SciPy Documentation, "Pearson Correlation Coefficient in Python," https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.pearsonr.html.  
[8] Pandas Documentation, "DataFrame.apply() function," https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.apply.html.  
[9] Pandas Documentation, "DataFrame.fillna() function," https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.fillna.html.  
[10] NumPy Documentation, "nanmean function to handle missing values," https://numpy.org/doc/stable/reference/generated/numpy.nanmean.html.  
[11] Rotten Tomatoes, "Harry Potter and the Deathly Hallows Part 2 Reviews," <https://www.rottentomatoes.com/m/harry_potter_and_the_deathly_hallows_part_2/reviews>.  
[12] Rotten Tomatoes, "Harry Potter and the Sorcerer's Stone Reviews," <https://www.rottentomatoes.com/m/harry_potter_and_the_sorcerers_stone/reviews>.  
[13] Rotten Tomatoes, "Harry Potter and the Prisoner of Azkaban Reviews," <https://www.rottentomatoes.com/m/harry_potter_and_the_prisoner_of_azkaban/reviews>.  
[14] Rotten Tomatoes, "Harry Potter and the Goblet of Fire Reviews," <https://www.rottentomatoes.com/m/harry_potter_and_the_goblet_of_fire/reviews>.  
[15] Rotten Tomatoes, "Harry Potter and the Order of the Phoenix Reviews," <https://www.rottentomatoes.com/m/harry_potter_and_the_order_of_the_phoenix/reviews>.  
[16] Instant Data Scraper Extension, "Web scraping extension," https://chrome.google.com/webstore/detail/instant-data-scraper.  
[17] College Board, "How to Convert GPA to a 4.0 Scale," <https://bigfuture.collegeboard.org/plan-for-college/get-started/how-to-convert-gpa-4.0-scale>.