

Assigned:  
May 17, 2025

Homework 5.0

Due:  
May 26, 2025

Please complete the assigned problems to the best of your abilities. Ensure that the work you do is entirely your own, external resources are only used as permitted by the instructor, and all allowed sources are given proper credit for non-original content.

## 1. Practicum Problems

These problems will primarily reference the lecture materials and the examples given in class using Python. It is suggested that a Jupyter/IPython notebook be used for the programmatic components.

### 1.1 Problem 1

Load the Movielens 100k dataset (ml-100k.zip) into Python using Pandas data frames. Convert the ratings data into a utility matrix representation and find the 10 most similar users for user 1 based on the cosine similarity of the centered user ratings data. Based on the average of the ratings for item 508 from similar users, what is the expected rating for this item for user 1?

In this task, the goal is to predict user 1's potential rating for movie 508 using a user-based collaborative filtering approach. First, I loaded the original rating data from the MovieLens 100k dataset and constructed a user-item utility matrix, where each cell represents a user's rating for a particular movie. Since different users have different rating standards, I followed the task requirement to center each user's ratings by subtracting their individual average rating, thereby standardizing the evaluation scale. Afterward, missing values in the matrix were filled with zeros to enable cosine similarity computation. In the similarity calculation phase, I selected user 1 as the target and computed the cosine similarity between user 1 and all other users based on the centered ratings, then identified the top 10 most similar users as neighbors. I then extracted the ratings these similar users gave to movie 508 and applied two different methods to predict user 1's rating: a simple average and a similarity-weighted average. The simple average takes the mean of the neighbors' ratings, while the weighted average multiplies each rating by its corresponding similarity score and normalizes the result, yielding a more fine-grained prediction. Finally, I compared the outcomes of the two methods and checked whether user 1 had actually rated movie 508 to assess the practical significance of the prediction.

The following library functions are used:

**pandas:** Used to read the rating data and construct the user-item rating matrix using DataFrame operations.

**numpy:** Used for performing numerical operations, such as calculating averages or handling missing data.

**scipy.spatial.distance.cosine:** Used to compute the cosine similarity between user rating vectors.

The following is the program's output:

```
[943 rows x 1682 columns]
The 10 users most similar to User 1:
User ID: 773, Similarity: 0.2048
User ID: 868, Similarity: 0.2023
User ID: 592, Similarity: 0.1966
User ID: 880, Similarity: 0.1958
User ID: 429, Similarity: 0.1907
User ID: 276, Similarity: 0.1875
User ID: 916, Similarity: 0.1864
User ID: 222, Similarity: 0.1824
User ID: 457, Similarity: 0.1823
User ID: 8, Similarity: 0.1809
```

Figure 1

The 10 most similar users and their similarity scores

```
Rating details of similar users for movie 508:
User ID: 592, Rating for Movie 508: 5.0, Average User Rating: 3.81
User ID: 880, Rating for Movie 508: 4.0, Average User Rating: 3.43
User ID: 429, Rating for Movie 508: 4.0, Average User Rating: 3.39
User ID: 276, Rating for Movie 508: 5.0, Average User Rating: 3.47
User ID: 222, Rating for Movie 508: 3.0, Average User Rating: 3.05
```

Figure 2

Ratings from the top ten similar users who rated the movie 508

```
Weighted details of similar user ratings:
User ID: 592, Rating: 5.0, Similarity: 0.1966, Weighted Scoring: 0.9830
User ID: 880, Rating: 4.0, Similarity: 0.1958, Weighted Scoring: 0.7832
User ID: 429, Rating: 4.0, Similarity: 0.1907, Weighted Scoring: 0.7626
User ID: 276, Rating: 5.0, Similarity: 0.1875, Weighted Scoring: 0.9374
User ID: 222, Rating: 3.0, Similarity: 0.1824, Weighted Scoring: 0.5472
```

Figure 3

Similarity-weighted scores from the top ten users who rated movie 508

For simple prediction scoring:

$$\text{Predicted Rating}_{\text{avg}} = \frac{1}{N} \sum_{i=1}^N r_i$$

Where:

$r_i$ : Rating of movie 508 by the i-th similar user.

$N$ : Number of similar users with ratings.

So the simple average score is:

$$\frac{5.0 + 4.0 + 4.0 + 5.0 + 3.0}{5} = \frac{21.0}{5} = 4.2000$$

For similarity-weighted scoring:

$$\text{Predicted Rating}_{\text{weighted}} = \frac{\sum_{i=1}^N \text{sim}_i \cdot r_i}{\sum_{i=1}^N \text{sim}_i}$$

Where:

$\text{sim}_i$ : The similarity between user i and user 1

$r_i$ : The similarity between user i and user 1

Numerator (weighted score sum):

0.9830+0.7832+0.7626+0.9374+0.5472=4.0134

Denominator (sum of similarities):

0.1966+0.1958+0.1907+0.1875+0.1824=0.9530

Final weighted score:

$$\frac{4.0134}{0.9530} \approx 4.2116$$

The difference is not significant, so User A predicts a rating of about 4.2 for Movie 508.

## 1.2 Problem 2

Load the Movielens 100k dataset (ml-100k.zip) into Python using Pandas data frames. Build a user profile on centered data (by user rating) for both users 200 and 15, and calculate the cosine similarity and distance between the user's preferences and the item/movie 95. Which user would a recommender system suggest this movie to?

For Problem 2, I first used Pandas to load the MovieLens 100k dataset and constructed a user-item rating matrix, where each row represents a user, each column represents a movie, and each cell stores the rating given by the user to that movie. To better reflect user preferences, I performed mean-centering on the ratings for each user as required—subtracting the user's average rating from each of their individual ratings—to eliminate the influence of differing rating scales among users. Then, I calculated the cosine similarity between users based on the centered rating matrix to identify the most similar neighbors for each target user. For the target movie, I extracted the ratings from those neighbors who had actually rated it, and predicted each target user's interest level by computing a weighted average of the neighbors' ratings, weighted by similarity. Finally, by comparing the predicted ratings for the two target users (User 200 and User 15), I determined which user the system would be more inclined to recommend the movie to.

The following library functions are used:

**pandas:** Used for reading data files, handling tabular data, and constructing the user-item rating matrix.

**numpy:** Used for performing numerical computations, such as calculating means and handling vectors and arrays.

**cosine\_similarity:** Provide a commonly used cosine similarity calculation function in machine learning, used to measure the degree of similarity between vectors.

**zipfile:** used to extract .zip files, assisting in the automatic extraction of the MovieLens dataset.

The prediction results show that User 200 has a significantly higher level of interest in the target movie compared to User 15. The system ultimately recommends the movie to User 200, and the fact that this user has an actual rating of 5 further indicates a strong interest in the movie.

-

---

END