AIE425 Intelligent Recommender Systems, Fall Semester 24/25

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

221100500, Maryam Eslam Sameer

1. Overview of Neighborhood-Based Collaborative Filtering

1.1 Abstract

This report uses collaborative filtering (CF) models to build a neighborhood-based recommender system. Focusing on user-based and item-based CF, the report explains how data is collected, prepared, and organized into a user-item matrix for similarity analysis. The goal is to create relevant recommendations by identifying patterns in user preferences, using similarity measures like Cosine and Pearson correlations.

1.2 Introduction

Recommender systems are everywhere, from streaming platforms to online stores, making it easier for people to find what they’re looking for. One popular method to power these systems is collaborative filtering (CF), which bases recommendations on user behavior and preferences. This assignment explores neighborhood CF models, focusing on user-based and item-based approaches.

The process starts with data collection through tools like APIs and web scrapers, followed by data preparation to build a rating matrix. Similarity measures like Cosine and Pearson correlations then help group similar users or items, forming the basis for recommendations. This report reviews each phase, showing how CF models enhance personalization in today’s digital experiences.

2. Assignment requirements and description

2.1 Companies that use recommender system

Answer:

1. Netflix.
2. Amazon.
3. Spotify.
4. Quora.
5. Movie lens.

2.2 Data Source for the Assignment

Answer:

Amazon Product Reviews

Source: Amazon Reviews data (Amazon Review Data (2018)). The repository has several databases. For this assignment, we are using the Electronics Ratings database only.

The method used to collect and download data is pre-curated and hosted by Hugging Face. (huggingface.html)

2.3 Customer feedback collection and rating type used

Answer:

Customer feedback on Amazon is collected through product reviews and seller feedback. The product rating type is a 5-point interval rating.

Detailed info: Product reviews are written evaluations of a specific product by customers, often rated from 1 to 5 stars, which help future buyers make informed decisions. Seller feedback focuses on the overall buying experience and rates the seller’s service, including delivery and customer service. Only customers who have spent at least $50 in the past 12 months can submit these reviews, ensuring authenticity. Additionally, machine-learned models calculate star ratings, prioritizing recent and verified purchases.

References:

- Tracefuse.AI
- Amazon.com

2.4 Preprocessing, Cleaning, and Feedback

Answer:

Detailed Preprocessing Feedback

1. Dataset Overview:
    - The dataset initially contained 4 columns: UserID, ItemID, Rating, and TimeStamp.
    - The TimeStamp column was dropped, reducing the dataset to focus solely on user-item interactions and ratings.
    - The dataset now includes 3 columns: UserID, ItemID, and Rating.
    - Total Users: 756,489
    - Total Items: 9,838,676
    - Total Ratings: 20,994,352
2. Removing Duplicates:
    - Duplicate UserID-ItemID pairs were removed, retaining only the first occurrence of each pair. This resulted in a more accurate representation of user-item interactions by ensuring that each user's rating for a specific item is counted only once.

- Dataset Shape After Deduplication: 20,994,352 rows with no duplicate entries for user-item pairs.

3. Rating Distribution:

- Minimum Rating: 1
- Maximum Rating: 5
- Initial rating distribution (before duplicate removal):
    - 5 Stars: 12,602,916
    - 4 Stars: 3,306,379
    - 3 Stars: 1,529,818
    - 2 Stars: 1,139,589
    - 1 Star: 2,415,650
- After removing duplicate ratings (same user and item pairs, *figure 1*):
    - 5 Stars: 12,339,969
    - 4 Stars: 3,229,099
    - 3 Stars: 1,497,569
    - 2 Stars: 1,117,046
    - 1 Star: 2,369,796
- The majority of ratings are 5 stars, indicating a positive bias in user feedback.

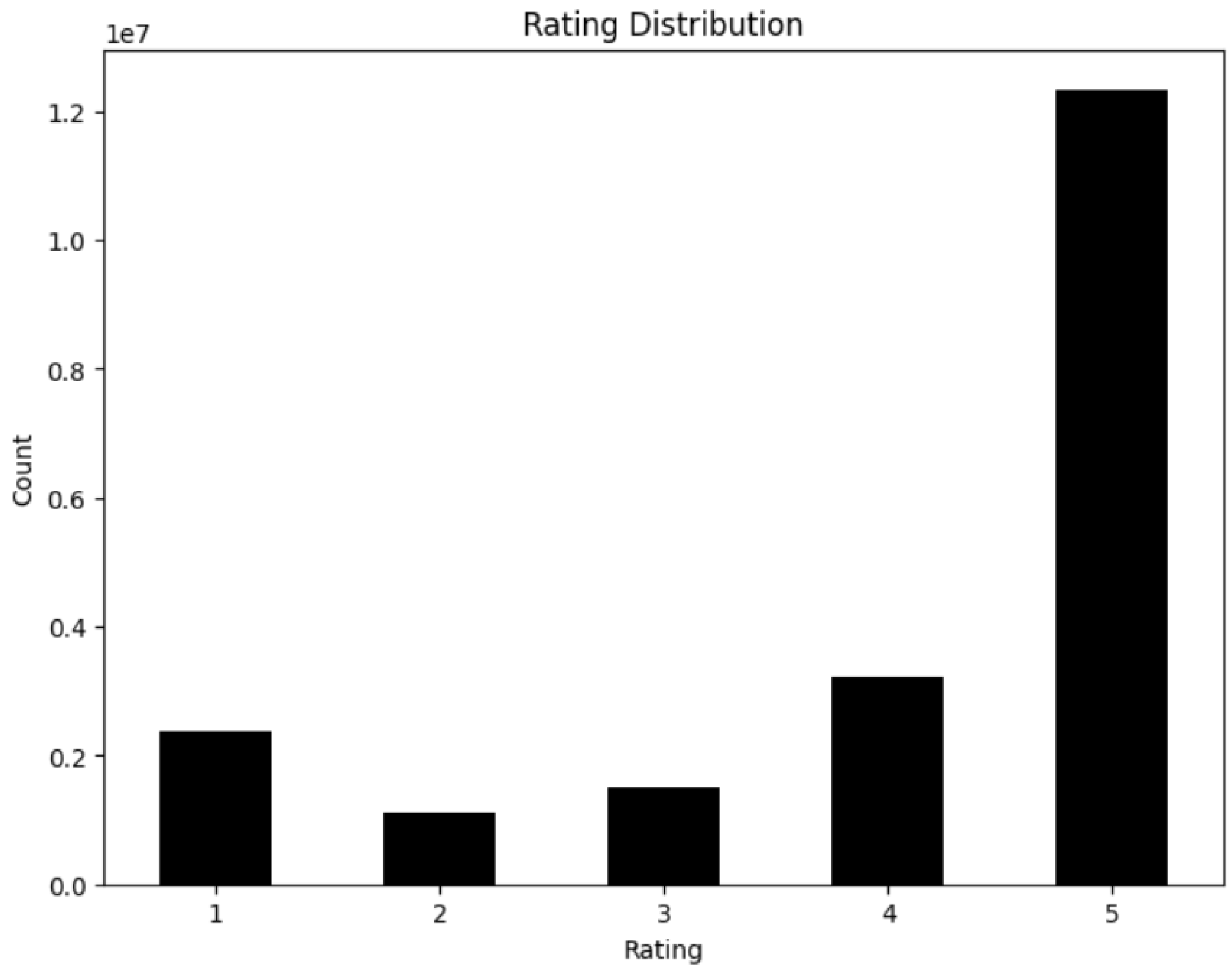

figure 1

4. Sorting by Item Popularity (*figure 2*):

    - The data was sorted based on the frequency of ratings per item. This sorted the dataset to display the most frequently rated items first, providing insights into the most popular products in the dataset.
    - Top 3 Most Popular Items:
        - ItemID A680RUE1FDO8B: 598 ratings
        - ItemID A3OXHLG6DIBRW8: 549 ratings
        - ItemID ADLVFFE4VBT8: 500 ratings

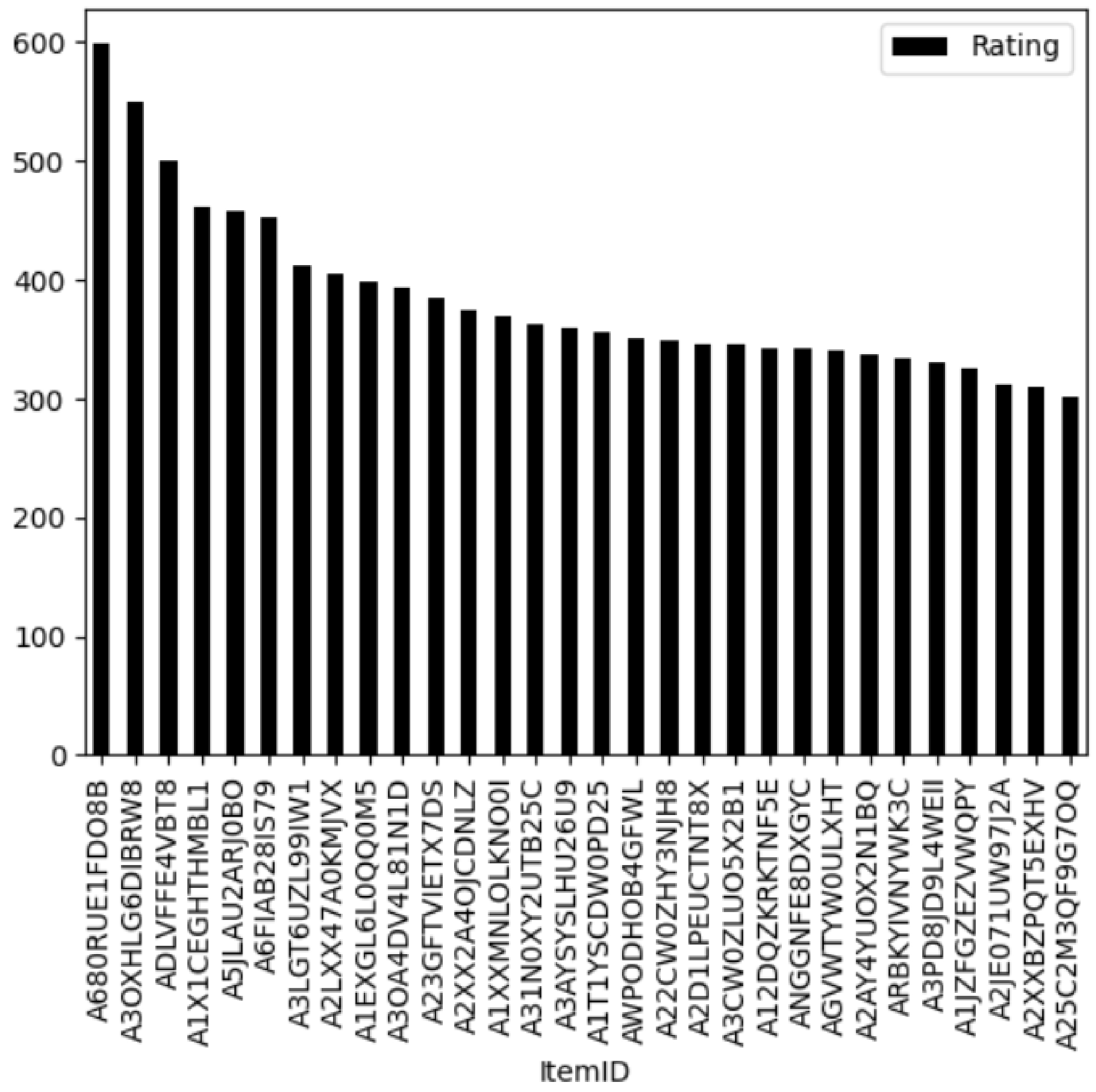

figure 2

5. Missing Values Check:

- UserID: 0 missing values
- ItemID: 0 missing values
- Rating: 0 missing values

- No missing values were found, confirming data completeness and integrity.

6. UserID and ItemID Mapping:

   - Unique UserID values were mapped from U1 to U756489, and ItemID values were mapped from I1 to I9838676.
   - This mapping provides consistent identifiers for users and items, simplifying references for analysis.

   Notebook Link: Colab Notebook

   Resources Used for Preprocessing and Code Implementation:

   - GeeksforGeeks.org
   - CodingNomads.com
   - Kaggle.com

2.5 Preprocessing and rating type

Answer:

1. Data Acquisition:

   - The dataset was loaded from a CSV file named "Electronics.csv". This file contains user interaction data, including unique user and item identifiers, rating scores, and timestamps of each rating.

2. Initial Structure:

   - The original dataset had four columns: UserID, ItemID, Rating, and TimeStamp.
   - The Rating column contains numerical values ranging from 1 to 5, representing user satisfaction with items.

3. Preprocessing Steps:

- For clarity, the columns were renamed UserID, ItemID, Rating, and TimeStamp.
- The TimeStamp column was removed, as it was not essential for the analysis and recommendation model.
- Duplicate entries for the same UserID and ItemID pairs were dropped, retaining only the first instance. This ensured that each user-item interaction was unique in the dataset.
- The dataset was sorted based on item popularity by counting the number of ratings each item received. This allowed us to identify and prioritize the most popular items for further analysis.
- To ensure consistency, UserID values were renamed sequentially from U1 to U756489, and ItemID values were renamed from I1 to I9838676. This mapping provided uniform identifiers across the dataset.

4. Rating Distribution Before and After Duplicate Removal:
    - Before Duplicate Removal:
        - 5 Stars: 12,602,916
        - 4 Stars: 3,306,379
        - 3 Stars: 1,529,818
        - 2 Stars: 1,139,589
        - 1 Star: 2,415,650
    - After Duplicate Removal:
        - 5 Stars: 12,339,969
        - 4 Stars: 3,229,099
        - 3 Stars: 1,497,569
        - 2 Stars: 1,117,046

- 1 Star: 2,369,796

2.6 User-Item Matrix

Answer:

User-Item	*I3*	*I9*	*I17*	*I983*	*I2*
U1099	4	0	0	2	5
U1060	0	3	0	3	5
U898	4	0	4	0	4
U646	0	3	0	3	5
U713	2	0	4	0	5

To simplify computations, the User-Item indexing will be adjusted to range from 1 to 5.

User-Item	*I1*	*I2*	*I3*	*I4*	*I5*
U1	4	0	0	2	5
U2	0	3	0	3	5
U3	4	0	4	0	4
U4	0	3	0	3	5
U5	2	0	4	0	5

This is the final matrix that will be used as a dataset for this assignment.

2.7 Matrix Dataset Description

Answer:

Detailed Description of the Resulting Matrix Dataset

The resulting matrix is derived from a subset of the original Amazon Product Reviews data, specifically focusing on user ratings for electronic items. The preprocessing steps applied to the original data ensure that each user-item interaction is unique, without duplicate ratings. The matrix includes a 5x5 selection of users and items, generated based on specific conditions to keep the dataset concise and focused.

Matrix Overview

1. Users (Rows):
    - The matrix includes 5 unique users (U1099, U1060, U898, U646, and U713), who were selected based on their interactions with specific target items.
    - Each user represents a distinct individual from the original dataset and has rated at least some of the selected items.
2. Items (Columns):
    - The matrix includes 5 unique items (I3, I9, I17, I983, I2), chosen to ensure a focused set of electronics products that these users interacted with.
    - Each item represents a distinct electronic product from the original dataset.
3. Ratings:
    - The ratings in the matrix are numerical values from 1 to 5, representing user satisfaction levels for the selected items.
    - A value of 0 indicates that the user did not rate the item, resulting in a sparse structure with some missing ratings.
4. Zero Values:
    - Each row (user) has some zero values, indicating unrated items by that user.
    - The matrix was designed to minimize zeros, though each row still contains two zero values due to the data's sparsity.

2.8 Computation of average rating

Answer:

Average Rating = $\frac{\sum_{i=1}^{S} ir_i}{\sum_{i=1}^{S} r_i} = \frac{(5)(4)+(4)(5)+(3)(4)+(2)(2)}{4+5+4+2} = 3.733$

2.9 CF Algorithms Background Overview and Analytical Solution

Answer:

Collaborative Filtering (CF) is a popular recommendation technique used in recommender systems. It predicts user preferences based on past behavior, utilizing either user-based or item-based methods:

1. User-Based Collaborative Filtering:
   - This method predicts a target user's preferences by identifying similar users rated items similarly.
   - Process:
     - Start by calculating the similarity between users who have rated common items. Various similarity measures can be used:
       - Computes similarity based on the angle between user rating vectors (Cosine Similarity).
       - Measures the linear correlation between users' ratings (Pearson Correlation Coefficient).
       - Considers only the overlap between users' ratings (Jaccard Coefficient).
     - Identify the k closest or most similar users to the target user, who will serve as "neighbors."

1.1 User-Based Collaborative Filtering Analytical Approach:

- Cosine similarity measure:

$$sim\left(\underset{a}{\rightarrow},\underset{b}{\rightarrow}\right) = \frac{\underset{a}{\rightarrow}\cdot\underset{b}{\rightarrow}}{\left|\underset{a}{\rightarrow}\right|\cdot\left|\underset{b}{\rightarrow}\right|}$$

  Predictions:

$$pred(u,p) = \bar{r_u} + \frac{\sum_{v\in N} sim(u,v) * \left(r_{V,p}\right)}{\sum_{v\in N} sim(u,v)}$$

- Pearson correlation coefficient:

$$Sim(u,v) = \frac{\sum_{p\in P} = P(r_{u,p} - \bar{r}_u)(r_{v,p} - \bar{r}_v)}{\sqrt{\sum_{p\in P} P(r_{u,p} - \bar{r}_u)^2}\sqrt{\sum_{p\in P} P(r_{u,p} - \bar{r}_v)^2}}$$

  Predictions:

$$pred(u,p) = \bar{r_u} + \frac{\sum_{v\in P_u(p)} sim(u,v) * \left(r_{V,p} - \bar{r_v}\right)}{\sum_{v\in P_u(p)}|sim(u,v)|}$$

  - Calculate the neighbors' bias and aggregate their ratings, weighted by similarity, to predict the target user's rating.

2. Item-Based Collaborative Filtering:

   - This approach predicts ratings by finding similar items rather than similar users. It examines items that a user has previously rated and then identifies similar items.

   - Process:

     - Calculate the similarity between items based on user ratings. Similarity measures used are similar to user-based CF

       - Similar to cosine similarity but mean-center ratings to account for user biases.

       - Measures the linear correlation between item ratings (Pearson Correlation Coefficient).

- Identify the top k most similar items to the target item.
- Compute the weighted average of ratings for these similar items to predict the rating for the target item.

2.10 till 2.16 Computation of similarity using Cosine similarity and Pearson coefficient with detailed comparison and explanation

Answer:

User 1 will be used as the target.

User-User CF:

Cosine Similarity CF User-based:

User-Item	I1	I2	I3	I4	I5	Cosine (u1,i)
U1	4	?	?	2	5	1
U2	?	3	?	3	5	0.987
U3	4	?	4	?	4	0.993
U4	?	3	?	3	5	0.814
U5	2	?	4	?	5	0.957

$$Cosine(u,v) = \frac{\sum_{p \in com(u,v)} = P(r_{u,p})(r_{v,p})}{\sqrt{\sum_{p \in com(u,v)} P(r_{u,p})^2}\ \sqrt{\sum_{p \in com(u,v)} P(r_{v,p})^2}}$$

$$Cosine(U1,U2) = \frac{2 \times 3 + 5 \times 5}{\sqrt{2^2 + 5^2} \times \sqrt{3^2 + 5^2}} = 0.987$$

$$Cosine(U1,U3) = \frac{4 \times 4 + 5 \times 4}{\sqrt{4^2 + 5^2} \times \sqrt{4^2 + 4^2}} = 0.993$$

$$Cosine(U1, U4) = \frac{2 \times 3 + 5 \times 5}{\sqrt{2^2 + 5^2} \times \sqrt{5^2 + 5^2}} = 0.814$$

$$Cosine(U1, U5) = \frac{4 \times 2 + 5 \times 5}{\sqrt{4^2 + 5^2} \times \sqrt{2^2 + 5^2}} = 0.987$$

$$Pred(u, p) = \bar{r_u} + \frac{\sum_{v \in N} sim\,(u, v) * (r_{V,p})}{\sum_{v \in N} sim\,(u, v)}$$

$$Pred(U1, I2) = \frac{3 \times 0.987 + 0 \times 0.993}{0.987 + 0.993} = 1.49$$

$$Pred(U1, I3) = \frac{0 \times 0.987 + 4 \times 0.993}{0.987 + 0.993} = 2$$

∴ Item 3 should be prioritized to user 1 as a recommendation than item 2

Pearson Coefficient CF User-based:

User-Item	I1	I2	I3	I4	I5	Mean Rating	Pearson (U1, i)
U1	4	?	?	2	5	3.6	1
U2	?	3	?	3	5	3.6	0.901
U3	4	?	4	?	4	4	0
U4	?	3	?	3	5	3.6	0.901
U5	2	?	4	?	5	3.6	0.426

$$Sim(u, v) = \frac{\sum_{p \in P} = P(r_{u,p} - \bar{r}_u)(r_{v,p} - \bar{r}_v)}{\sqrt{\sum_{p \in P} P(r_{u,p} - \bar{r}_u)^2}\ \sqrt{\sum_{p \in P} P(r_{u,p} - \bar{r}_v)^2}}$$

$$Pearson(U1, U2) = \frac{(2 - 3.6)(3 - 3.6) + (5 - 3.6)(5 - 3.6)}{\sqrt{(2 - 3.6)^2 + (5 - 3.6)^2} + \sqrt{(3 - 3.6)^2 + (5 - 3.6)^2}} = 0.901$$

$$Pearson(U1, U3) = \frac{(4-3.6)(4-4)+(5-3.6)(4-4)}{\sqrt{(4-3.6)^2+(5-3.6)^2}+\sqrt{(4-4)^2+(4-4)^2}}$$

$$= No\ similarity$$

$$Pearson(U1, U4) = \frac{(2-3.6)(3-3.6)+(5-3.6)(5-3.6)}{\sqrt{(2-3.6)^2+(5-3.6)^2}+\sqrt{(3-3.6)^2+(5-3.6)^2}} = 0.901$$

$$Pearson(U1, U5) = \frac{(4-3.6)(2-3.6)+(5-3.6)(5-3.6)}{\sqrt{(4-3.6)^2+(5-3.6)^2}+\sqrt{(2-3.6)^2+(5-3.6)^2}} = 0.426$$

$$Pred(u,p) = \overline{r_u} + \frac{\sum_{v \in p_{u(P)}} sim\ (u,v) * (r_{V,p} - \overline{r_v}\ )}{\sum_{v \in p_{u(P)}} |sim\ (u,v)|}$$

$$Pred(U1, I2) = 3.6 + \frac{(3-3.6)(0.901)+(3-3.6)(0.901)}{0.901+0.901} = 3$$

$$Pred(U1, I3) = 3.6 + \frac{(0)(0.901)+(0)(0.901)}{0.901+0.901} = 0$$

∴ Item 2 should be prioritized to user 1 as a recommendation than item 3

- Cosine Similarity Results:

  Recommendation: Item 3 is prioritized over Item 2 for User U1.

  Explanation: Cosine similarity measures the angle between two users' rating vectors, emphasizing the similarity in the pattern rather than the absolute rating values. Here, the angle between U1 and similar users suggests that Item 3 aligns more closely with U1’s preferences.

- Pearson Correlation Results:

  Recommendation: Item 2 is prioritized over Item 3 for User U1.

  Explanation: Pearson correlation centers ratings around each user's mean, which helps in capturing the linear relationship between users’ ratings. Based on this measure, Item 2 better matches U1's adjusted preferences, considering user biases and rating patterns.

Important note: results are not reasonable due to the big sparse in the matrix.

Difference between Cosine and Pearson:

- Pearson Correlation:

  Pearson correlation measures the linear relationship between two vectors. It evaluates how well the data points fit a straight line. However, it’s important to note that Pearson correlation does not consider the scale of the vectors; it focuses solely on the linear relationship.

- Cosine Similarity:

  Cosine similarity, on the other hand, measures the angle between two vectors. It captures the directional similarity between them, regardless of their magnitude. This property makes it particularly useful for scenarios where scale isn’t significant. Cosine similarity is commonly employed in text analysis, representing documents as vectors.

Pros of Cosine Similarity in Recommender Systems

- The cosine similarity is beneficial because even if the two similar data objects are far apart by the Euclidean distance because of their size, they could still have a smaller angle between them. The smaller the angle, the higher the similarity.
- When plotted on a multi-dimensional space, the cosine similarity captures the orientation (the angle) of the data objects and not the magnitude.

Cons of Cosine Similarity in Recommender Systems

- Sensitive to Sparse Data: Cosine similarity may not be effective when applied to sparse data wherein many of its components are zero in the vectors. For that, other similarities would work better.
- Does Not Account for Absolute Differences: Cosine similarity only considers the angle between vectors and not their magnitude; hence, this may miss out on differences in magnitude, which, in certain contexts, may well be relevant

- Symmetry: Cosine similarity is symmetric, which simply means that it cannot differentiate between the order of comparison. For some tasks, this may not be desirable since directionality may be relevant.
- Not Applicable for Negative Values: Cosine similarity may not generally be applicable in datasets containing negative values, as misleading results can be obtained, or the angle between vectors interpretation becomes problematic.

Pros of Pearson Correlation in Recommender Systems

1. Bias Adjustment:
    - Pearson correlation accounts for individual user biases by centering each user's ratings around their mean. This normalization process helps mitigate differences in rating scales (e.g., some users rate generously while others are more conservative).
2. Personalized Recommendations:
    - By capturing the relative strength of user preferences, Pearson correlation can provide more personalized recommendations, as it considers how much each user deviates from their average rating for specific items.
3. Captures Linear Relationships:
    - Pearson correlation effectively captures linear relationships between users' rating behaviors, identifying similarity based on how users' ratings fluctuate. This can be useful when similar users consistently rate items higher or lower than average.
4. Enhanced Accuracy for Dense Datasets:
    - In datasets with high overlap (where many users have rated similar items), Pearson correlation can yield more accurate recommendations by leveraging the linear relationship between overlapping ratings.

5. Effective for Diverse Rating Patterns:

    - When users have unique rating patterns, Pearson correlation provides a more refined similarity measure than methods that only consider the absolute rating values (like Cosine Similarity). This makes it more robust in cases with significant rating diversity.

Cons of Pearson Correlation in Recommender Systems

1. Requires Overlapping Ratings:

    - Pearson correlation relies on overlapping ratings between users to compute similarity accurately. In sparse datasets, where users have rated very few common items, Pearson correlation can be unreliable or even unusable due to lack of data.

2. Sensitive to Outliers:

    - Pearson correlation can be affected by outliers, as extreme ratings can skew the correlation. If a user rates an item unusually high or low, it may distort the similarity measure with other users.

3. Computational Complexity:

    - Pearson correlation is more computationally intensive than some simpler similarity measures, especially in large datasets. Calculating the mean and deviations for each user before computing similarity can be time-consuming in real-time recommendation systems.

4. Only Captures Linear Relationships:

    - Pearson correlation is limited to detecting linear relationships. If user preferences have non-linear relationships (e.g., one user's high rating on an item does not correlate with another's high rating predictably), Pearson may not capture the true similarity.

5. Less Effective in Sparse Datasets:
    - In very sparse datasets, where users have rated only a few items, Pearson correlation's dependency on overlapping ratings can result in unstable or undefined similarity scores, limiting its effectiveness in generating recommendations.
6. Ignores Magnitude of Ratings:
    - Although Pearson correlation accounts for average rating bias, it does not capture the absolute magnitude of ratings. For example, two users could have the same correlation despite one user consistently rating items higher than the other. This can limit its effectiveness in applications where the actual rating value matters.

Item-Item CF:

User-Item	I1	I2	I3	I4	I5	Mean Rating
U1	4	?	?	2	5	3.6
U2	?	3	?	3	5	3.6
U3	4	?	4	?	4	4
U4	?	3	?	3	5	3.6
U5	2	?	4	?	5	3.6

Mean Centered Matrix

User-Item	I1	I2	I3	I4	I5
U1	0.4	?	?	-1.6	1.4
U2	?	-0.6	?	-0.6	1.4
U3	0	?	0	?	0
U4	?	-0.6	?	-0.6	1.4
U5	-1.6	?	0	?	1.4

$$Adjusted\ Cosine\ (i,j) = \frac{\sum_{u\in U_i\cap U_j} S_{u,i}\cdot S_{u,j}}{\sqrt{\sum_{u\in U_i\cap U_j}(S_{u,i})^2}\ \sqrt{\sum_{u\in U_i\cap U_j}(S_{u,j})^2}}$$

$Adjusted\ Cosine\ (I2, I1) = No\ similarities$

$Adjusted\ Cosine\ (I2, I3) = No\ similarities$

$$Adjusted\ Cosine\ (I2, I4) = \frac{(-0.6)(-0.6) + (-0.6)(-0.6)}{\sqrt{(-0.6)^2(-0.6)^2} + \sqrt{(-0.6)^2(-0.6)^2}} = 1$$

$$Adjusted\ Cosine\ (I2, I5) = \frac{(-0.6)(1.4) + (-0.6)(1.4)}{\sqrt{(-0.6)^2(-0.6)^2} + \sqrt{(1.4)^2(1.4)^2}} = -1$$

$Adjusted\ Cosine\ (I3, I1) = No\ similarities$

$Adjusted\ Cosine\ (I3, I2) = No\ similarities$

$Adjusted\ Cosine\ (I3, I4) = No\ similarities$

$Adjusted\ Cosine\ (I3, I5) = No\ similarities$

∴ Item 3 is not similar to any item

∴ Item very similar to item 4 but not that similar to item 5

$$Pred(u,t) = \frac{\sum_{j\in Q_t(u)} AdjustedCosine\ (j,t)\cdot r_{u,j}}{\sum_{j\in Q_t(u)}|AdjustedCosine\ (j,t)|}$$

$$Pred(U1, I2) = \frac{2\times 1}{1} = 2$$

$$Pred(U1, I5) = \frac{5\times -1}{-1} = 5$$

∴ Item 5 is preferred to User 1 than item 2

Assignment Results (Similarity Computation and Comparison):

This document presents a summary, evaluation, and comparison of the results of similarity computations using Cosine Similarity, Pearson Coefficient, and Adjusted Cosine Similarity. An important note on data sparsity affecting accuracy is also included.

1. Cosine Similarity (User-User CF)

Measures similarity based on the cosine of the angle between users' rating vectors. The calculation involves summing the products of ratings and dividing by the square root of the sum of squares for each user's ratings.

User Pair	Cosine Similarity
U1, U2	0.987
U1, U3	0.993
U1, U4	0.814
U1, U5	0.957

Note: Items are recommended based on computed similarity; Item 3 is prioritized over Item 2 for U1.

2. Pearson Coefficient (User-User CF)

Measures similarity by calculating correlation, accounting for the users’ mean ratings. Can better reflect similarity when data has varying rating scales.

User Pair	Pearson Coefficient
U1, U2	0.901
U1, U3	No similarity
U1, U4	0.901
U1, U5	0.426

Note: Similarity computed suggests items for recommendation, with Item 3 recommended over Item 2.

3. Adjusted Cosine Similarity (Item-Item CF)

Adjusts for user biases by centering ratings around the user mean before computing similarities. Focuses on item similarities rather than users.

Item Pair	Adjusted Cosine Similarity
I2, I4	1
I2, I5	-1

Note: Item 5 is preferred to U1 over Item 2 due to similarity computations.

4. Comparative Analysis

Method	User/Item Pair	Similarity Value	Recommended Item
Cosine Similarity	U1, U3	0.993	Item 3
Pearson Coefficient	U1, U2	0.901	Item 2
Adjusted Cosine	I2, I4	1	Item 5

5. Evaluation and Important Note

While each method provides insights into user or item similarity, the results should be interpreted cautiously due to the sparse nature of the data matrix.

2.17 Brief about the implementation process, tools, and libraries

Answer:

Implementation Process

1. Data Preparation: A dense user-item matrix represents user ratings for various items. Each row represents a user, and each column represents an item. Missing ratings are assumed to be 0.
2. Similarity Calculation: For collaborative filtering, similarities are calculated between users or items.
    - User-Based: Similarity is computed between users based on their ratings across all items.
    - Item-Based: Similarity is computed between items based on user ratings.
3. Prediction Calculation: Using the similarity scores, unknown ratings are predicted. For a given user and item, the weighted average of similar users' or items' ratings is used, where weights are their similarity scores.

Tools and Libraries

- Pandas: This library is central for handling the user-item matrix as a DataFrame, making it easy to manipulate rows and columns for similarity and prediction calculations.
- NumPy: Used to manage numerical computations and handle array operations efficiently, especially useful when summing and averaging weighted ratings.
- scikit-learn: Provides the cosine_similarity function, which allows for quick computation of cosine similarities between users or items in the matrix.

2.18 Remarks about User-Based and Item-Based CF using Pearson Coefficient

Answer:

What Each Method Focuses On:

- User-Based CF: This approach focuses on finding users who have similar tastes. It looks for users who rate items in similar ways, which can be helpful when users tend to have overlapping interests or preferences.
- Item-Based CF: This approach, on the other hand, finds items that are similar based on how users have rated them. This is useful if each user has rated only a few items, as it can still recommend items that are related to the ones they liked.

How Pearson Correlation is Used:

- User-Based CF with Pearson: When used with users, Pearson correlation looks at items that two users have both rated and considers how their ratings compare to each other's averages. This means that it doesn't just look at the rating scores but also at the pattern of ratings, which helps in identifying users with similar tastes, even if they use different rating scales (e.g., someone who always rates low and someone who always rates high).
- Item-Based CF with Pearson: With items, Pearson correlation compares ratings of two items by users who rated both. This is useful when certain items tend to be liked or disliked together, so it allows for recommending items that "go together" based on how they were rated by different users.

How Cosine Similarity is Used:

- User-Based CF with Cosine Similarity: In user-based collaborative filtering, cosine similarity measures how similar two users' rating patterns are based on the angle between their rating vectors. It focuses on the direction of their ratings rather than the exact scores, which helps find users with similar preferences, even if they use different rating scales (like one user rating mostly high and another rating mostly low). If two users rate items similarly in terms of preference but not necessarily in value, cosine similarity can still identify them as similar.

- Item-Based CF with Cosine Similarity: In item-based collaborative filtering, cosine similarity compares how similar two items are based on the ratings given by different users. If two items are rated similarly by the same users, even if the actual scores vary, they will have a high cosine similarity score. This can help find items that are often liked or disliked together, which is useful for making recommendations by suggesting items that are “related” in terms of user preferences.

2.19 Conclusion on each strategy's effect on the predicted accuracy

Answer:

Based on the results from using Cosine Similarity, Pearson Coefficient, and Adjusted Cosine Similarity, each strategy shows different strengths and weaknesses in predicting item recommendations, especially with the challenge of sparse data.

1. Cosine Similarity (User-User CF)
   - This method measures similarity by looking at the angle between users' rating patterns. It was effective in identifying similar users, like U1 and U3, who had a high similarity score (0.993). Using this approach, we could recommend Item 3 to U1 over Item 2 based on high similarity values. Cosine Similarity works well in finding users with similar preferences, but it might be less effective when data is sparse or when users have different rating habits.
2. Pearson Coefficient (User-User CF)
   - Pearson Coefficient calculates similarity by adjusting for each user’s average rating, which makes it more sensitive to different rating styles. For example, it found U1 and U2 to have a moderate similarity (0.901) but

showed "no similarity" between U1 and U3. This approach is useful when users rate items differently (e.g., some rate high and others low) but can struggle in sparse datasets, as seen here where limited data affected similarity detection.

3. Adjusted Cosine Similarity (Item-Item CF)

   - Adjusted Cosine Similarity focuses on item relationships by normalizing ratings based on each user’s average. This helps to capture item similarities more accurately. For example, it identified a strong similarity between Item 2 and Item 4 (score of 1), making Item 5 a better recommendation for U1 than Item 2. This method is good for item-based recommendations, especially when dealing with user biases, but it can be limited in accuracy when the data is sparse, as shown by the negative similarity for some pairs (e.g., I2 and I5).

4. Comparison Summary

   - Cosine Similarity worked well in cases with high overlap, providing reliable recommendations for users with similar rating patterns. Pearson Coefficient was better at adjusting for different rating styles but was more affected by sparse data. Adjusted Cosine Similarity was useful for item-based recommendations and handled user biases, but it also faced limitations in sparse datasets.

2.20 Enhancement addressing

Answer:

Given the high sparsity of the data, future tasks will focus on retrieving data that is less sparse to enhance the accuracy of recommendations.

3. Conclusion and opinion

In this assignment, various collaborative filtering (CF) methods were explored to assess their effectiveness in recommending items. The results showed that Cosine Similarity was effective for user-based CF in identifying similar users and providing recommendations based on high similarity scores. However, its performance was limited by sparse data and differences in users' rating habits.

Pearson Coefficient, which adjusts for each user's average rating, offered better insights for users with varying rating styles. Although it was effective in accounting for user biases, it struggled with sparse data, as seen in cases where limited common ratings reduced its ability to detect similarity accurately.

Adjusted Cosine Similarity, applied for item-based CF, addressed user biases by normalizing ratings, making it valuable for item-item comparisons. This method was particularly useful in identifying items with strong similarities, though it faced challenges with sparse datasets, which sometimes resulted in negative similarity values.

The report concludes that each method has unique strengths depending on data characteristics, but data sparsity significantly affected accuracy. Future tasks will focus on retrieving less sparse data to improve recommendation reliability. Overall, this assignment highlights the importance of choosing appropriate CF methods based on dataset structure and suggests that enhancing data quality could lead to more accurate recommendations.