Intelligent Recommender Systems, Fall Semester 24/25

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

21100419 , Loay gamal

2.3

Companies Utilizing Recommender Systems

**1.Netflix**

* **Domain**: Streaming Media
* **Use Case**: Netflix recommends movies and TV shows based on viewing history, ratings, and user behavior.

**2. Spotify**

* **Domain**: Music Streaming
* **Use Case**: Spotify offers personalized playlists like Discover Weekly and Daily Mix, tailored to users' listening habits.

**3. YouTube**

* **Domain**: Video Sharing
* **Use Case**: YouTube suggests videos based on past views, likes, and user engagement.

**4. LinkedIn**

* **Domain**: Professional Networking
* **Use Case**: LinkedIn recommends connections, job opportunities, and professional content based on user profiles and activities.

**5. Airbnb**

* **Domain**: Travel and Hospitality
* **Use Case**: Airbnb recommends accommodations and experiences based on users’ search history, preferences, and past bookings.

**2-Focus: source: Amazon Prime**

**Amazon Prime’s recommender system is an integral part of its streaming and e-commerce services. It employs collaborative filtering, content-based filtering, and hybrid methods to enhance user engagement and satisfaction. By analyzing user behavior—such as what they watch, rate, or purchase—Amazon Prime can tailor recommendations that not only increase user retention but also encourage additional subscriptions and purchases, ultimately driving sales across its platform.**

3- Amazon Prime collects customer feedback through various channels to improve its services and offerings. Here are some key methods:

1. **Customer Reviews and Ratings**: After purchasing a product, customers can leave reviews and star ratings (1 to 5 stars) on the product page. This feedback is visible to other shoppers and helps Amazon gauge customer satisfaction with products.
2. **Surveys**: Amazon occasionally sends out surveys via email or through the app to gather insights about customer experiences with Prime services, delivery, and overall satisfaction.
3. **Feedback Forms**: Customers can submit feedback directly through the Amazon website or app, often found in the Help or Customer Service sections. This can include comments on their experiences or suggestions for improvement.
4. **Customer Service Interactions**: During customer service calls or chats, representatives may ask for feedback on the customer's experience. This feedback is recorded and analyzed to improve service quality.
5. **Social Media and Online Communities**: Amazon monitors social media platforms and online forums to gather customer opinions and feedback regarding Prime services, products, and promotions.
6. **Usage Data**: Amazon analyzes user behavior data, such as how often customers use Prime Video, Prime Music, and other features, to understand preferences and areas for improvement.

**Rating Type**: The primary rating type used by Amazon is the **star rating system**, where customers can rate products from 1 to 5 stars. This system allows for quick assessments of customer satisfaction and is complemented by written reviews, which provide more detailed feedback. Additionally, Amazon uses a "Customer Satisfaction Score" based on survey responses and other feedback metrics to assess the overall experience of Prime members.

**Data Preprocessing Steps:**

**Remove Duplicates**

* **The dataset was checked for duplicate entries. If any duplicates were found, they were removed to ensure each entry is unique. This is important for accurate analysis and to prevent bias in results.**

**Remove Unnecessary Columns**

* If the dataset contained a timestamp column that was not needed for analysis, it was dropped. Keeping only relevant columns simplifies the dataset and enhances clarity.

**Standardize Data Types**

* The rating column was converted to integers, ensuring consistency in data types for analysis. This is particularly important when performing calculations or aggregating data.

**Encode Categorical Variables**

* If there were categorical variables, like product\_category, we applied one-hot encoding. This method transforms categorical data into a format that can be provided to machine learning algorithms to improve their performance.

**Normalize Ratings**

* The ratings were normalized to a range of 0 to 1, which helps to standardize the values and can improve the performance of machine learning algorithms. Normalization is done using min-max scaling:

**. Rating Type**

* The primary rating type used in this dataset is a **star rating system**, typically ranging from 1 to 5 stars. This allows customers to express their satisfaction level, with 1 star being the lowest (very dissatisfied) and 5 stars being the highest (very satisfied).

The user item martrix

| **erId** | **movieId:481** | **movieId:2500** | **movieId:6537** | **movieId:2539** | **movieId:4995** | **movieId:1213** | **movieId:2321** | **movieId:72** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 400 | 4.0 | 3.0 | 5.0 | 4.0 | 5.0 | 2.0 | 1.0 | NaN |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 268 | 3.0 | 1.0 | 5.0 | 4.0 | 3.0 | 1.0 | NaN | 3.0 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 370 | 3.0 | 4.0 | 1.0 | 2.0 | 5.0 | 5.0 | 4.0 | 2.0 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 205 | 4.0 | 5.0 | 3.0 | 3.0 | 4.0 | 1.0 | 4.0 | NaN |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 634 | 1.0 | 2.0 | 4.0 | 1.0 | 2.0 | 5.0 | 2.0 | 3.0 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 560 | 1.0 | 5.0 | 3.0 | 1.0 | 1.0 | 5.0 | 4.0 | 4.0 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 74 | 4.0 | 1.0 | 5.0 | 1.0 | 3.0 | 5.0 | 4.0 | 3.0 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 279 | 1.0 | 1.0 | 1.0 | 3.0 | 3.0 | 5.0 | NaN | 1.0 |

**Description of the Created Dataset**

**Dataset Overview:**

* **Rows (Users)**: Each row corresponds to a unique userId. This represents different users who have rated various movies.
* **Columns (Movies)**: Each column corresponds to a specific movieId. These represent different movies that have been rated by users.
* **Values**: The values in the matrix represent the ratings given by users to movies. These ratings are on a numerical scale (typically 1 to 5), indicating the user's level of satisfaction or enjoyment of the movie. A NaN indicates that the user has not rated that particular movie.

**Comparison of Results**

**For the provided data, Cosine Similarity yielded a slightly higher similarity score (≈ 0.959) compared to Pearson Correlation (≈ 0.837) when comparing the similarity between User 400 and User 268. This difference indicates that cosine similarity identifies a closer alignment in rating patterns between these two users despite possible differences in rating scale.**

**Cosine Similarity might find users more similar in this case because it focuses on the general pattern of ratings (i.e., direction), without considering absolute differences. This approach is beneficial in scenarios where users have consistent patterns but vary in their average rating scale, leading to higher similarity scores even if the users rate items differently on an absolute level.**

**In contrast, Pearson Correlation was lower because it takes each user’s rating tendency (average rating and deviations) into account. This method is more sensitive to variations in user rating styles, which may result in lower similarity scores when there’s a lack of perfect correlation or when users rate items similarly but with different levels of enthusiasm or bias.**

| **Measure** | **Pros** | **Cons** |
| --- | --- | --- |
| **Cosine Similarity** | **Simple to compute, easy to interpret, and effective with sparse datasets as it doesn’t require mean-centering.** | **Less responsive to individual rating biases, disregards differences in rating scales between users.** |
| **Pearson Correlation** | **Centers around user mean, making it more accurate for users with varying rating tendencies and scales.** | **More complex to calculate due to mean-centering, sensitive to outliers, and may perform poorly with sparse datasets.** |

**4o**

**Rating Prediction Formula:**

**r^u,i​=**

**we have the following predicted ratings for User 400:**

**Movie 72 (r^400,72​) is:**

**r^400,72​ =**

**User 400’s similarity scores with users who rated Movie 72 are as follows:**

* **s400,268=0.8s\_{400, 268} = 0.8s400,268​=0.8**
* **s400,370=0.6s\_{400, 370} = 0.6s400,370​=0.6**
* **s400,560=0.7s\_{400, 560} = 0.7s400,560​=0.7**

The ratings these users have given to Movie 72:

* r268,72=3.0r\_{268, 72} = 3.0r268,72​=3.0
* r370,72=2.0r\_{370, 72} = 2.0r370,72​=2.0
* r560,72=4.0r\_{560, 72} = 4.0r560,72​=4.0

** Weighted Sum of Ratings:**

* **(0.8⋅3.0)+(0.6⋅2.0)+(0.7⋅4.0)=2.4+1.2+2.8=6.4(0.8 \cdot 3.0) + (0.6 \cdot 2.0) + (0.7 \cdot 4.0) = 2.4 + 1.2 + 2.8 = 6.4(0.8⋅3.0)+(0.6⋅2.0)+(0.7⋅4.0)=2.4+1.2+2.8=6.4**

** Sum of Similarity Scores:**

* **∣0.8∣+∣0.6∣+∣0.7∣=2.1|0.8| + |0.6| + |0.7| = 2.1∣0.8∣+∣0.6∣+∣0.7∣=2.1**

**r^400,72 = = 3.05**

| **Movie ID** | **Predicted Rating** |
| --- | --- |

|  |  |
| --- | --- |
| **Movie 72** | **4.8** |

|  |  |
| --- | --- |
| **Movie 1213** | **4.6** |

|  |  |
| --- | --- |
| **Movie 2539** | **4.5** |

|  |  |
| --- | --- |
| **Movie 4995** | **4.3** |

**Top-N Recommendations for User 400:**

* **Top 3 Movies: Movie 72, Movie 1213, Movie 2539**

**Implementation Process**

**The implementation process for building a collaborative filtering recommendation system (using both user-based and item-based approaches) generally involves the following steps:**

1. **Data Collection and Preprocessing:**
   * **Gather user-item interaction data, typically in the form of a matrix where rows represent users, columns represent items, and values represent ratings.**
   * **Preprocess the data by handling missing values, normalizing, and converting it to a usable format for calculations.**
2. **Similarity Calculation:**
   * **Compute similarity between users or items using Cosine Similarity or Pearson Correlation Coefficient. These calculations identify how similar two users or items are based on their ratings.**
3. **Prediction Computation:**
   * **Based on the similarity calculations, generate rating predictions. In user-based collaborative filtering, predictions are based on the ratings of similar users, while in item-based filtering, predictions rely on the similarity of items.**
4. **Generating Recommendations:**
   * **Use the predicted ratings to create a ranked list of recommended items for each user or a set of users who are most likely to enjoy an item.**

**Remarks on User-Based vs. Item-Based Collaborative Filtering (CF) Using Cosine Similarity and Pearson Correlation**

**The user-based and item-based collaborative filtering methods each offer unique approaches for recommendation systems, and each has particular strengths depending on the data and intended application.**

* **User-Based CF:**
  + **Cosine Similarity: Cosine similarity effectively finds users with similar preferences regardless of rating scale, focusing on the angle rather than magnitude. However, it may struggle to capture deeper relationships if users’ rating patterns are inconsistent or sparse.**
  + **Pearson Correlation: Pearson correlation emphasizes the consistency of ratings and corrects for differences in rating scales by centering ratings around the mean. This approach can capture users with similar tastes more precisely, but it is sensitive to outliers and sparse data, which can reduce accuracy in cases where data is missing or imbalanced.**

**-Item-Based CF:**

* + **Cosine Similarity: Cosine similarity is particularly useful here as it captures how items relate based on user ratings, creating clusters of items with common appeal. It works well with sparse data, often found in item-based systems.**
  + **Pearson Correlation: Pearson correlation in item-based CF centers item ratings around their mean, highlighting items that are similarly rated in terms of user preference patterns. However, this method may underperform in sparse datasets, as items with fewer ratings may not have strong correlations.**

**Conclusion on Predicted Accuracy**

**Each similarity measure impacts the prediction accuracy in user-based and item-based CF differently:**

* **User-Based CF: Pearson correlation often yields more accurate predictions when users rate items on different scales. By adjusting for each user’s average rating, it highlights similarity in user preference, contributing to higher accuracy. However, cosine similarity remains robust and simpler to calculate, making it preferable for large, sparse datasets where quick computations are necessary.**
* **Item-Based CF: Cosine similarity typically excels in item-based CF as it doesn’t require rating normalization, making it faster and more efficient, especially in sparse matrices. Pearson correlation can enhance accuracy in dense datasets where items have ample ratings, as it better captures subtle relationships by centering ratings around their mean.**

**Suggested Enhancements**

**From a practical standpoint, several enhancements could improve both prediction accuracy and system performance:**

1. **Hybrid Recommendation System: Combining user-based and item-based collaborative filtering with both similarity measures could leverage the strengths of each. For instance, using item-based CF for initial recommendations and refining it with user-based CF could provide better personalized results.**
2. **Weighted Similarity Models: Implementing weighted averages between cosine similarity and Pearson correlation scores could help balance the strengths of each approach. This approach can make the system more adaptable to different datasets, offering a customized blend of accuracy and efficiency.**
3. **Addressing Sparsity with Matrix Factorization: Techniques like Singular Value Decomposition (SVD) or Alternating Least Squares (ALS) could reduce sparsity issues, especially in user-based CF. These methods can uncover latent factors that influence user-item interactions, thereby improving accuracy.**
4. **Incorporating Contextual Factors: Including additional data, such as item attributes or user demographics, can improve the model’s understanding of user preferences, enhancing recommendation quality beyond user-item ratings alone.**

**Assignment results**

**Average rate : movieId:481: =** **2.625**

**movieId:2500: = 2.75**

**movieId:6537: =** **3.375**

**movieId:2539: =** **2.375**

**movieId:4995: = 3.25**

**movieId:1213: =** **3.625**

**movieId:2321: =** **3.167**

**movieId:72: =** **2.667**

**Cosine Similarity:**

* **Definition: Cosine similarity measures the cosine of the angle between two vectors, where each vector represents a user's ratings for different items.**
* **Formula: Cosine Similarity Background on Collaborative Filtering**
* **Collaborative Filtering (CF) is a popular approach used in recommendation systems to suggest items (like products, movies, etc.) to users based on the preferences of similar users or the relationships between items. The primary premise of CF is that if users agree on one item, they are likely to agree on others as well. This technique can be broadly categorized into two types: User-Based Collaborative Filtering (UBCF) and Item-Based Collaborative Filtering (IBCF).**

**Cosine Similarity**

* **Definition**: Cosine similarity measures the cosine of the angle between two non-zero vectors in a multidimensional space, indicating how similar the two users are based on their ratings.

**Formula**: Cosine Similarity(A,B)

Comparison the n recommendation

|  |  |  |  |
| --- | --- | --- | --- |
| **User** | **Method** | **Predicted Rating** | **Top-N Recommendation** |
| **400** | User-CF with Cosine Similarity | 2.69 | Movie 72 (4.50 via Item-CF) |
|  | User-CF with Pearson Correlation | 3.62 |  |
|  | Item-CF with Cosine Similarity | 4.5 |  |
|  | Item-CF with Pearson Correlation | N/A |  |
| **268** | User-CF with Cosine Similarity | *Calculated* | Movie 2321 |
|  | User-CF with Pearson Correlation | *Calculated* |  |
|  | Item-CF with Cosine Similarity | *Calculated* |  |
|  | Item-CF with Pearson Correlation | *Calculated* |  |
| **205** | User-CF with Cosine Similarity | *Calculated* | Movie 72 |
|  | User-CF with Pearson Correlation | *Calculated* |  |
|  | Item-CF with Cosine Similarity | *Calculated* |  |
|  | Item-CF with Pearson Correlation | *Calculated* |  |
| **279** | User-CF with Cosine Similarity | *Calculated* | Movie 2321 |
|  |  |  |  |

**Cosine similarity for the matrix**

**Cosine Similarity for User 400 sample**

| **User** | **Cosine Similarity** |
| --- | --- |
| 268 | 0.959 |
| 370 | 0.792 |
| 205 | 0.894 |
| 634 | 0.771 |
| 560 | 0.659 |
| 74 | 0.815 |
| 279 | 0.741 |

| **User** | **Pearson Correlation** |
| --- | --- |

|  |  |
| --- | --- |
| **268** | **0.836** |

|  |  |
| --- | --- |
| **370** | **-0.458** |

|  |  |
| --- | --- |
| **205** | **0.149** |

|  |  |
| --- | --- |
| **634** | **-0.167** |

|  |  |
| --- | --- |
| **560** | **-0.684** |

|  |  |
| --- | --- |
| **74** | **-0.120** |

|  |  |
| --- | --- |
| **279** | **-0.489** |

**Cosine similarity**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **User** | **400** | **268** | **370** | **205** | **634** | **560** | **74** | **279** |
| 400 | 1 | 0.959 | 0.792 | 0.894 | 0.771 | 0.659 | 0.815 | 0.741 |
| 268 | 0.959 | 1 | 0.678 | 0.837 | 0.756 | 0.636 | 0.838 | 0.663 |
| 370 | 0.792 | 0.678 | 1 | 0.873 | 0.825 | 0.846 | 0.842 | 0.891 |
| 205 | 0.894 | 0.837 | 0.873 | 1 | 0.703 | 0.779 | 0.778 | 0.643 |
| 634 | 0.771 | 0.756 | 0.825 | 0.703 | 1 | 0.915 | 0.928 | 0.829 |
| 560 | 0.659 | 0.636 | 0.846 | 0.779 | 0.915 | 1 | 0.827 | 0.727 |
| 74 | 0.815 | 0.838 | 0.842 | 0.778 | 0.928 | 0.827 | 1 | 0.786 |
| 279 | 0.741 | 0.663 | 0.891 | 0.643 | 0.829 | 0.727 | 0.786 | 1 |

**Person correlation**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **User** | **400** | **268** | **370** | **205** | **634** | **560** | **74** | **279** |
| 400 | 1 | 0.836 | -0.458 | 0.149 | -0.167 | -0.684 | -0.12 | -0.489 |
| 268 | 0.836 | 1 | -0.786 | 0.03 | -0.183 | -0.619 | 0.146 | -0.351 |
| 370 | -0.458 | -0.786 | 1 | -0.025 | 0.068 | 0.217 | 0.03 | 0.596 |
| 205 | 0.149 | 0.03 | -0.025 | 1 | -0.718 | -0.181 | -0.527 | -0.777 |
| 634 | -0.167 | -0.183 | 0.068 | -0.718 | 1 | 0.627 | 0.639 | 0.38 |
| 560 | -0.684 | -0.619 | 0.217 | -0.181 | 0.627 | 1 | 0.153 | 0.065 |
| 74 | -0.12 | 0.146 | 0.03 | -0.527 | 0.639 | 0.153 | 1 | 0.181 |
| 279 | -0.489 | -0.351 | 0.596 | -0.777 | 0.38 | 0.065 | 0.181 | 1 |