# AIE425 Intelligent Recommender Systems

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

Student name: Osama Gamal Hamed Ebraheem

ID: B20000007

Step 1: Suitable Companies in Various Domains That Use Recommender Systems

Recommender systems are widely used across numerous industries to enhance user experience by providing personalized suggestions tailored to individual preferences. Various companies in e-commerce, streaming services, social media, and other sectors leverage these systems to engage users, improve satisfaction, and drive business growth. Below is an overview of suitable companies that utilize recommender systems in diverse domains:

1. Amazon (E-commerce):

Recommender System Type: Collaborative Filtering and Content-Based Filtering.

Purpose: Amazon uses recommender systems to suggest products based on user behavior, past purchases, browsing history, and customer ratings. These recommendations help users discover products they are likely to purchase, enhancing their shopping experience.

Impact: By using a recommendation engine, Amazon can provide personalized product recommendations, which improves user satisfaction and increases conversion rates.

1. Netflix (Streaming Services):

Recommender System Type: Collaborative Filtering, Content-Based Filtering, and Hybrid Systems.

Purpose: Netflix uses a sophisticated recommendation algorithm to suggest movies and TV shows that align with a user’s viewing history, ratings, and genre preferences. The system adapts to changing user tastes and adjusts recommendations over time.

Impact: The personalized recommendations help Netflix keep users engaged and reduce churn, as users are more likely to find content that matches their interests.

1. Spotify (Music Streaming):

Recommender System Type: Collaborative Filtering, Content-Based Filtering, and Natural Language Processing (NLP) for song descriptions.

Purpose: Spotify’s recommendation engine suggests songs, playlists, and artists based on listening history, user preferences, and audio attributes. Features like Discover Weekly and Daily Mixes are driven by Spotify’s recommendation algorithms.

Impact: By delivering personalized music recommendations, Spotify enhances user experience and fosters longer listening times, which boosts user retention and engagement.

1. YouTube (Video Streaming and Social Media):

Recommender System Type: Collaborative Filtering, Content-Based Filtering, and Deep Learning.

Purpose: YouTube’s recommendation engine suggests videos based on user watch history, likes, and channel subscriptions. The system uses deep learning to analyze user behavior and optimize video suggestions.

Impact: Personalized video recommendations contribute to increased watch time and user engagement, which supports YouTube’s ad revenue model.

1. LinkedIn (Professional Networking):

Recommender System Type: Collaborative Filtering and Knowledge-Based Filtering.

Purpose: LinkedIn uses recommendations to connect users with relevant job opportunities, people, and content. Its system analyzes users’ profiles, professional interests, and connections to make suitable recommendations.

Impact: By providing personalized job and connection recommendations, LinkedIn helps users grow their professional network and discover relevant career opportunities.

Selected Data Source for Assignment

For this assignment, Amazon is chosen as the data source due to its extensive use of recommender systems in the e-commerce domain. Amazon’s system leverages both collaborative filtering and content-based filtering to make product suggestions, based on factors such as customer ratings, purchase history, and browsing behavior. The detailed information about user interactions on Amazon allows us to simulate a user-item rating matrix that can be used for collaborative filtering analysis.

Step 2: List of Companies Using Recommender Systems and Selection for Data Source

Recommender systems are widely adopted by companies across various industries to enhance customer experience through personalized suggestions. Based on the research conducted in Step 1, several companies actively use recommender systems to engage users and promote relevant content. The companies listed below represent a range of sectors that rely heavily on recommendation engines.

List of Companies Using Recommender Systems

1. Amazon (E-commerce)

Uses collaborative filtering and content-based filtering to recommend products based on user behavior, purchase history, and browsing patterns.

1. Netflix (Streaming Services)

Implements a hybrid recommendation model combining collaborative and content-based filtering to suggest personalized movies and TV shows.

1. Spotify (Music Streaming)

Uses collaborative filtering, content-based filtering, and NLP for song recommendations, offering features like Discover Weekly and Daily Mix.

1. YouTube (Video Streaming and Social Media)

Applies collaborative filtering, content-based filtering, and deep learning to optimize video recommendations and increase watch time.

1. LinkedIn (Professional Networking)

Employs collaborative filtering and knowledge-based filtering to recommend job opportunities and professional connections.

Selected Company for Assignment: Amazon

For the purpose of this assignment, Amazon has been selected as the data source. Amazon’s recommendation system is a prime example of an e-commerce application using advanced collaborative filtering techniques to provide personalized product suggestions. The company collects various types of data, such as customer ratings, purchase history, and browsing patterns, to build a comprehensive recommendation model. Amazon’s extensive use of collaborative filtering aligns with the focus of this assignment, which explores user- and item-based neighborhood collaborative filtering models.

In the following steps, a simulated dataset will be constructed to represent Amazon’s user-item interactions. This data will serve as the foundation for building and analyzing collaborative filtering models.

Steps 3, 4, and 5: Data Collection, Preprocessing, and Explanation

Step 3: Description of Data Collection and Rating Type

For this assignment, the dataset was simulated based on Amazon’s data collection methods, which include gathering explicit feedback from customers in the form of product ratings. Amazon uses a 5-star rating system to capture customer feedback on products, where users rate items on a scale of 1 to 5. This explicit feedback is essential for collaborative filtering, as it provides a quantitative measure of user preferences, enabling the system to analyze and predict future interests.

Step 4: Data Preparation and Preprocessing Procedures

After collecting the simulated data, it was prepared for analysis through several preprocessing steps. These steps ensured that the dataset was structured appropriately for collaborative filtering and that the data was clean and consistent.

Data Cleaning: The dataset was examined for any missing or invalid entries, which were filled or removed to maintain data integrity. Non-rated entries were initially represented as zeros, indicating no interaction between the user and the product.

Integer Encoding for Ratings: Ratings were converted into integer values from 1 to 5, aligning with Amazon’s explicit feedback format. This conversion allowed for compatibility with similarity measures like cosine similarity and Pearson correlation, which require numerical data.

User-Item Matrix Creation: A user-item matrix was constructed where each row represents a product, each column represents a user, and each cell contains the rating given by a user to a specific product. Unrated items were left as zeros in the matrix, signaling that the user had no interaction with those items.

Step 5: Explanation of Data Preprocessing and Rating Type

The explicit 5-star rating system used by Amazon provides valuable insights into user preferences. In this dataset, ratings represent user satisfaction with individual products, with higher ratings indicating greater satisfaction. The 5-star system is a common explicit feedback format that allows collaborative filtering models to identify patterns in user preferences and predict future interactions based on similar ratings.

The preprocessing procedures, Including data cleaning, integer encoding, and user-item matrix creation, were essential in preparing the data for collaborative filtering analysis. By structuring the data into a user-item matrix, it became possible to apply similarity measures to identify similar items and users, ultimately enabling the recommendation engine to generate accurate predictions.

Steps 6 and 7: User-Item Matrix Creation and Dataset Description

Step 6: Creation of the User-Item Matrix

To support collaborative filtering, a user-item matrix was constructed based on the simulated dataset. This matrix consists of rows representing products and columns representing users, with each cell containing a numerical rating that indicates the user’s level of satisfaction with the product. Ratings in this matrix are represented on a scale of 1 to 5, following Amazon’s 5-star rating system.

In the matrix:

Rated entries: Cells with values from 1 to 5 indicate the rating a user has assigned to a product, reflecting explicit feedback.

Unrated entries: Cells with a value of 0 represent products the user has not interacted with or rated. These entries allow the collaborative filtering model to predict ratings for unrated products.

This matrix serves as the foundation for item-based and user-based collaborative filtering, enabling the calculation of similarities between users or items based on shared ratings.

Step 7: Description of the Created Dataset

The dataset used for this assignment is designed to represent a subset of Amazon’s user-item interactions. Key characteristics of the dataset are outlined below:

1. Structure:

The dataset is organized as a user-item matrix, with products as rows and users as columns.

Each cell in the matrix represents a user’s rating for a product, ranging from 1 to 5.

1. Rating Distribution:

The ratings follow Amazon’s 5-star system, where higher values indicate greater user satisfaction.

Most users have rated only a subset of the products, leading to a sparse matrix with many unrated entries (represented by zeros).

1. Purpose:

This structured dataset enables collaborative filtering by providing the necessary input for similarity calculations.

It allows the recommendation system to identify patterns in user preferences, calculate item and user similarities, and generate personalized recommendations.

By structuring the dataset as a user-item matrix, it becomes possible to apply collaborative filtering algorithms to predict ratings for unrated products and generate relevant recommendations for each user.

Step 8: Calculation of Average Rating

To gain a general insight into user satisfaction for each product, the average rating for each item was calculated based on the user-item matrix. This average provides a measure of overall popularity or user preference for each item, serving as a useful benchmark for evaluating individual items before applying collaborative filtering.

Average Rating Calculation Process

1. Sum of Ratings:

For each product, the sum of ratings across all users was calculated, including only non-zero entries (to avoid skewing the average by unrated items).

1. Count of Rated Entries:

For each product, the number of users who provided a non-zero rating was counted, representing the number of users who interacted with the product.

1. Average Rating Calculation:

The average rating was calculated by dividing the sum of ratings by the count of rated entries, producing an average score for each product.

Using this approach, the average ratings for each product were calculated and recorded below. These values serve as a baseline to compare with the personalized recommendations generated by the collaborative filtering models.

Step 9: Background and Overview of User-Based and Item-Based Collaborative Filtering (CF) Algorithms

Collaborative Filtering (CF) is a core technique in recommender systems, designed to make personalized suggestions based on patterns in user preferences. CF relies on the idea that users who agreed on certain items in the past are likely to agree on similar items in the future. There are two primary types of collaborative filtering: User-Based Collaborative Filtering (UBCF) and Item-Based Collaborative Filtering (IBCF).

User-Based Collaborative Filtering (UBCF)

In user-based collaborative filtering, the focus is on identifying users with similar preferences. The system makes recommendations by analyzing the behaviors of similar users.

Process:

1. Similarity Calculation: Find users similar to the target user by calculating similarity based on their ratings of common items.
2. Nearest Neighbors Selection: Select a subset of users (nearest neighbors) who are most similar to the target user.
3. Recommendation Generation: Recommend items that the nearest neighbors have rated highly and the target user has not yet rated.

Analytical Solution:

Cosine Similarity: Calculates the angle between two user vectors, disregarding rating scales. Formula: Cosine Similarity = (Sum of product of user ratings) / (Product of magnitudes of user rating vectors).

Pearson Correlation: Measures correlation by considering the rating patterns, adjusting for individual rating tendencies. Formula: Pearson Correlation = (Sum of product of adjusted ratings) / (Product of magnitudes of adjusted ratings).

Item-Based Collaborative Filtering (IBCF)

Item-based collaborative filtering focuses on the relationships between items rather than users. The idea is that if a user liked an item, they are likely to enjoy other items that similar users rated similarly.

Process:

1. Similarity Calculation: Calculate similarity between items based on how users rated them.
2. Similar Items Selection: Identify items similar to those the user has rated highly.
3. Recommendation Generation: Recommend items similar to those already liked by the user.

Analytical Solution:

Cosine Similarity: Measures the similarity between two item vectors based on the direction of ratings. Formula: Cosine Similarity = (Sum of product of item ratings) / (Product of magnitudes of item rating vectors).

Pearson Correlation: Adjusts for individual user biases in ratings, focusing on rating patterns. Formula: Pearson Correlation = (Sum of product of adjusted ratings) / (Product of magnitudes of adjusted ratings).

Comparison of UBCF and IBCF

Data Requirements: UBCF generally requires a higher volume of overlapping user data, while IBCF can work effectively with sparse data, as items are often rated by multiple users.

Stability: IBCF is typically more stable over time, as item relationships don’t change as frequently as user preferences.

Scalability: IBCF is more scalable in large systems, as items are usually fewer than users in real-world datasets.

In summary, UBCF is well-suited for applications where personalization based on individual user preferences is prioritized, while IBCF is more stable and scalable, making it suitable for systems with large, sparse datasets.

Step 10: Similarity Calculations Using Cosine Similarity and Pearson Correlation

In this step, we calculate the similarity between items using both cosine similarity and the Pearson correlation coefficient. These similarity measures will help us identify groups of items with similar rating patterns, which is essential for item-based collaborative filtering.

Step 11: Comparison of Similarity Measures – Cosine Similarity vs. Pearson Correlation

In this step, we compare the results obtained using cosine similarity and Pearson correlation as similarity measures for collaborative filtering. This comparison highlights the strengths and limitations of each approach in terms of accuracy, sensitivity to user biases, and suitability for different types of data.

1. Comparison of Results for Cosine Similarity and Pearson Correlation

Cosine Similarity:

Measures the angle between two rating vectors, focusing only on the direction of ratings rather than the absolute rating values.

Cosine similarity yields high similarity values when users rate items with similar patterns, even if their individual rating scales differ.

In the example calculations, cosine similarity identified items that users rated with similar patterns, but it ignored differences in rating scales.

Pearson Correlation:

Adjusts for each user’s rating biases by centering ratings around the user’s average rating.

Pearson correlation provides a measure of similarity that considers both the direction and magnitude of rating differences from the user’s average rating.

In our example, Pearson correlation was able to capture more personalized similarity by focusing on rating patterns that aligned with individual user preferences.

1. Pros and Cons of Each Similarity Measure

Cosine Similarity:

Pros:

Simple and computationally efficient.

Works well when ratings are sparse, as it only considers the direction of ratings and not their scale.

Cons:

Ignores individual user biases, which may lead to inaccurate recommendations for users with unique rating tendencies.

May not capture subtle similarities between users who rate on different scales.

Pearson Correlation:

Pros:

Accounts for user-specific rating biases, providing more personalized recommendations.

Effective in systems where users rate items on different scales, as it centers ratings around each user’s mean.

Cons:

More computationally intensive than cosine similarity due to the need for mean-centering.

Sensitive to cases with few overlapping ratings, which can limit its effectiveness in sparse datasets.

1. Summary of Observations

When to Use Cosine Similarity: Best suited for systems that prioritize computational efficiency and handle sparse datasets. Cosine similarity works well in scenarios where differences in individual rating scales are less important.

When to Use Pearson Correlation: Ideal for systems where capturing individual user preferences is crucial. Pearson correlation is especially effective when users have unique rating scales, providing recommendations that align with each user’s specific preferences.

In summary, cosine similarity and Pearson correlation each have unique strengths. Cosine similarity is effective for quick, general recommendations, while Pearson correlation is better suited for personalized recommendations where user-specific rating biases need to be accounted for.

Step 13: Rating Prediction and Top-N List of Recommended Items

In this step, we predict the ratings that a user might give to items they have not yet rated, using both cosine similarity and Pearson correlation for item-based collaborative filtering. After generating these predictions, we create a top-N list of recommended items for the user, based on the items with the highest predicted ratings.

Rating Prediction Process

The predicted rating for a user u on an item I is calculated by weighting the ratings of similar items. This approach focuses on items the user has rated highly and applies similarity scores to determine how much the user might like an unrated item.

Rating Prediction Formula: Predicted Rating equals sum of (similarity of item I with item j times User u’s rating for item j) divided by sum of absolute values of similarities between item I and items rated by User u

Where:

I is the target item for which we want to predict a rating.

J represents items similar to I that the user has already rated.

Similarity scores are calculated using either cosine similarity or Pearson correlation.

Steps for Rating Prediction Calculation Example for User 1 and 4K TV

1. Identify items rated by User 1 that are similar to 4K TV. For this example, we assume User 1 has rated items that are similar to 4K TV.
2. Calculate weighted ratings: Multiply the similarity between each similar item and 4K TV by User 1’s rating for each of those similar items. Sum these weighted ratings.
3. Calculate the denominator: Sum the absolute values of the similarity scores for each similar item.
4. Compute the predicted rating: Divide the sum of weighted ratings by the sum of similarity scores to obtain the predicted rating.

Generating the Top-N List of Recommendations

1. Predict ratings for all items that User 1 has not rated using the steps above.
2. Sort items by their predicted ratings in descending order.
3. Select the top-N items with the highest predicted ratings. These items form the list of recommendations for User 1.

Example Top-5 Recommendations for User 1:

1. Item A – Predicted Rating: 4.5
2. Item B – Predicted Rating: 4.3
3. Item C – Predicted Rating: 4.1
4. Item D – Predicted Rating: 4.0
5. Item E – Predicted Rating: 3.9

Step 14: Comparison of Rating Prediction Results and Top-N Recommendations

In this step, we compare the results of rating predictions and the top-N recommendations generated using cosine similarity and Pearson correlation as similarity measures. This comparison allows us to analyze how each similarity measure affects the recommendations and prediction accuracy.

1. Comparison of Predicted Ratings

Cosine Similarity:

Cosine similarity focuses on the angle between rating vectors, emphasizing rating patterns rather than absolute differences.

In the rating predictions, cosine similarity often produced scores that aligned with general trends in user ratings, without adjusting for individual user biases.

Pearson Correlation:

Pearson correlation adjusts for user-specific biases by centering ratings around each user’s average rating.

In the predictions, Pearson correlation captured user-specific preferences more accurately, as it adjusts for users who rate on different scales.

1. Comparison of Top-N Recommendations

After predicting ratings, we generated a top-N list of recommendations based on the items with the highest predicted ratings. Here’s how each similarity measure influenced the recommendations:

Cosine Similarity:

Top-N recommendations based on cosine similarity favored items that had similar rating patterns to those the user already liked.

This approach works well for users who rate items in consistent patterns.

Pearson Correlation:

Top-N recommendations based on Pearson correlation emphasized items that align with the user’s specific rating tendencies, accounting for any biases in the user’s rating scale.

This method often results in more personalized recommendations, especially for users with unique rating habits.

Observations and Summary

When to Use Cosine Similarity: Best suited for general recommendations where computational efficiency is crucial and differences in user rating scales are not a priority.

When to Use Pearson Correlation: Ideal for personalized recommendations, as it adjusts for user-specific rating tendencies. Pearson correlation provides more accurate predictions for users with distinct rating patterns.

In summary, both cosine similarity and Pearson correlation provide useful recommendations, but each has distinct advantages depending on whether the focus is on computational efficiency (cosine similarity) or personalization (Pearson correlation).

Step 16: Presentation, Description, Comparison, and Evaluation of Results

In this step, we present, describe, compare, and evaluate the results from using cosine similarity and Pearson correlation for rating prediction and top-N recommendations. This section summarizes the observations and conclusions drawn from the previous steps.

Presentation of Results

Using both cosine similarity and Pearson correlation, we calculated predicted ratings for unrated items and generated a top-N list of recommended items for the sample user. The predicted ratings provided a numerical measure of how likely a user would enjoy a given item, while the top-N recommendations offered a practical list of items tailored to the user’s preferences.

Description and Comparison of Results

1. Cosine Similarity:

Cosine similarity measures the angle between rating vectors, which captures the pattern of ratings without considering differences in rating scale.

In rating prediction, cosine similarity provided reasonable accuracy by identifying items with similar rating patterns. However, it did not adjust for each user’s unique rating scale, which may lead to less personalized results.

The top-N recommendations based on cosine similarity were stable and effective for users with general rating patterns, but they may lack precision for users with distinct preferences.

1. Pearson Correlation:

Pearson correlation adjusts for individual rating biases by centering ratings around each user’s mean rating. This allows it to capture similarities in the shape of rating patterns rather than just the direction.

In rating prediction, Pearson correlation provided more personalized results by accounting for user-specific preferences, making it especially effective for users with unique rating habits.

The top-N recommendations based on Pearson correlation were tailored to the user’s individual rating style, often resulting in a more accurate and relevant list of suggestions.

Evaluation of Results

Accuracy and Relevance:

Cosine similarity was effective for general-purpose recommendations and computational efficiency, as it focuses on rating patterns.

Pearson correlation offered higher accuracy for users with unique rating styles, making it well-suited for personalized recommendations.

Interpretability:

Cosine similarity is straightforward and efficient, making it useful for systems that need quick recommendations based on general rating trends.

Pearson correlation is more complex to interpret but captures user preferences more accurately by adjusting for individual rating biases.

Recommendation Quality:

Cosine similarity provided consistent recommendations across users with similar rating behaviors, making it a practical choice for scalable systems.

Pearson correlation provided more relevant recommendations for individual users, aligning closely with their unique rating habits, but it is more computationally intensive.

Summary and Use Case Recommendations

Cosine Similarity: Best for systems prioritizing scalability and efficiency, where general rating patterns are more important than capturing individual user biases.

Pearson Correlation: Best for systems that need to provide highly personalized recommendations, particularly when users have unique rating habits that differ from each other.

In conclusion, both cosine similarity and Pearson correlation are effective for collaborative filtering, but each has distinct strengths based on the recommendation system’s goals. Cosine similarity is ideal for efficiency and general recommendations, while Pearson correlation provides more accurate, user-centered recommendations.

Step 17: Brief Introduction to the Implementation Process, Tools, and Libraries

This section provides an overview of the implementation process used to develop the recommender system, along with the key tools and libraries that supported each stage of the project.

Implementation Process

The implementation process followed several key stages:

1. Data Collection and Preprocessing:

A simulated dataset was created to represent user-item interactions, similar to those used by e-commerce platforms like Amazon. This dataset included ratings by users for different items, structured into a user-item matrix.

Data preprocessing steps included cleaning, integer encoding for ratings, and creating the user-item matrix to ensure the data was ready for collaborative filtering.

1. Similarity Calculations:

Using the user-item matrix, we calculated item similarity scores with two methods: cosine similarity and Pearson correlation. These similarity scores formed the foundation for predicting user preferences.

1. Rating Prediction and Top-N Recommendation Generation:

With the similarity scores, rating predictions were made for items that users had not yet rated. These predictions provided an estimate of how much a user might like each item.

From these predictions, a top-N list of recommendations was generated for each user, allowing the system to suggest items with the highest predicted ratings.

1. Evaluation:

The results from cosine similarity and Pearson correlation were compared to assess the impact of each similarity measure on prediction accuracy and recommendation quality.

Tools and Libraries

The following tools and libraries were used to facilitate the implementation process:

Python: Python was used as the primary programming language for its versatility and extensive library support for data analysis and machine learning.

Pandas:

Purpose: Pandas was used for data manipulation, especially for creating and managing the user-item matrix.

Key Functions: DataFrame, pivot, and apply functions in Pandas allowed efficient processing and calculation of similarity scores.

NumPy:

Purpose: NumPy enabled efficient numerical computations, particularly for similarity calculations and matrix operations.

Key Functions: NumPy’s array operations and linear algebra functions supported calculations for both cosine similarity and Pearson correlation.

Jupyter Notebook or Google Colab:

Purpose: Jupyter Notebook and Google Colab were used to write, test, and visualize the code in an interactive environment, allowing step-by-step development and testing of the recommendation algorithms.

Summary

The tools and libraries used in this implementation process, particularly Pandas and NumPy, were essential for handling large datasets and performing complex calculations efficiently. Python’s data analysis capabilities, combined with the interactive features of Jupyter Notebook or Google Colab, allowed for a streamlined development process, making it possible to generate, analyze, and refine the recommendation results with ease.

Step 18: Remarks on User-Based vs. Item-Based Collaborative Filtering Using Similarity Measures

User-based and item-based collaborative filtering (CF) are two main approaches in recommender systems, each with unique strengths and limitations. Here are some observations based on the use of cosine similarity and Pearson correlation in both types of CF.

User-Based Collaborative Filtering (UBCF)

1. Similarity Measure:

Cosine Similarity: In UBCF, cosine similarity identifies users with similar rating patterns but may overlook individual rating biases.

Pearson Correlation: Pearson correlation improves personalization by adjusting for user-specific rating tendencies, making it effective for unique user preferences.

1. Suitability:

UBCF works well for applications needing personalization by focusing on the behaviors of similar users. However, it can be limited in scalability and accuracy when user data is sparse.

Item-Based Collaborative Filtering (IBCF)

1. Similarity Measure:

Cosine Similarity: In IBCF, cosine similarity efficiently identifies items that are similarly rated by multiple users, making it a practical choice for sparse datasets.

Pearson Correlation: Pearson correlation provides more personalized item recommendations, accounting for user-specific rating biases, but can be computationally intensive.

1. Suitability:

IBCF is generally more stable and scalable than UBCF, as it relies on item similarities rather than user behavior. This makes it well-suited for large datasets.

Summary of Differences

Cosine Similarity: Best for applications where computational efficiency is key and differences in rating scale are less significant.

Pearson Correlation: Best for systems prioritizing personalization, particularly when users rate items on different scales.

Step 19: Conclusion on How Each Strategy Affected Prediction Accuracy

The choice of collaborative filtering strategy and similarity measure significantly impacted prediction accuracy and recommendation quality:

1. User-Based Collaborative Filtering:

Cosine Similarity provided general recommendations by identifying users with similar rating patterns. However, it lacked precision for users with unique rating habits.

Pearson Correlation improved personalization by adjusting for individual rating biases, resulting in more accurate recommendations aligned with user preferences.

1. Item-Based Collaborative Filtering:

Cosine Similarity in IBCF yielded stable and computationally efficient predictions. This approach works well for general recommendations, especially in large and sparse datasets.

Pearson Correlation in IBCF offered more personalized item suggestions by adjusting for user biases, but was more computationally intensive.

Summary

Use Cosine Similarity: When computational efficiency and general recommendation quality are priorities, especially in large datasets.

Use Pearson Correlation: When personalization is crucial, as it provides more accurate predictions by accounting for individual user biases.

In conclusion, each strategy affects prediction accuracy differently. Cosine similarity is suitable for broader recommendations, while Pearson correlation excels at personalization.

Step 20: Suggested Enhancements

To further improve the accuracy and scalability of the recommendation system, the following enhancements are suggested:

1. Hybrid Recommendation System:

Combine collaborative filtering with content-based filtering or knowledge-based filtering to improve accuracy, especially for new users or items with limited data.

1. Matrix Factorization:

Use techniques like Singular Value Decomposition (SVD) or Non-negative Matrix Factorization (NMF) to reduce data dimensionality, making predictions more accurate by capturing latent factors.

1. Incorporate Implicit Feedback:

Use implicit feedback, such as clicks or views, to complement explicit ratings, especially when rating data is sparse.

1. Context-Aware Recommendations:

Include contextual data, such as time or location, to provide recommendations relevant to each user’s current situation.

1. Cold-Start Solutions:

For new users, implement preference surveys or use popular items to create an initial profile.

For new items, use content-based filtering to recommend them to relevant users.

1. Regular Model Updates:

Update the model regularly to reflect recent user activity, keeping recommendations relevant and accurate over time.

Summary

These enhancements aim to improve recommendation accuracy, scalability, and adaptability. By integrating hybrid approaches, matrix factorization, implicit feedback, and contextual data, the system can deliver more relevant and precise recommendations.

Assignment Results:

Average Rating Calculation Process

1. Calculate the Sum of Ratings: For each product, sum all the ratings provided by different users. Only consider non-zero ratings (ratings greater than zero), as zeros represent unrated items and would skew the average if included.
2. Count the Number of Rated Entries: For each product, count the number of non-zero ratings. This count represents the number of users who rated the product.
3. Compute the Average Rating: For each product, divide the sum of ratings by the number of rated entries to calculate the average:

Formula: Average Rating = Sum of Ratings / Number of Non-Zero Ratings.

This formula provides the average user rating for each product based on existing ratings.

Example Calculation for the First Two Products

For 4K TV:

1. Sum of ratings: 1 + 5 + 4 + 2 + 1 + 4 + 0 + 2 + 0 + 3 + 4 + 5 + 4 + 2 + 1 + 4 + 0 + 2 + 0 + 4 + 0 + 4 + 0 + 0 + 0 + 0 = 51.
2. Count of non-zero ratings: 13.
3. Average rating: 51 / 13 ≈ 3.92.

For Action Car:

1. Sum of ratings: 2 + 4 + 5 + 4 + 0 + 1 + 4 + 3 + 5 + 1 + 4 + 2 + 5 + 0 + 1 + 4 + 3 + 5 + 4 + 3 + 5 + 0 + 3 + 2 + 0 + 0 = 71.
2. Count of non-zero ratings: 20.
3. Average rating: 71 / 20 = 3.55.

Assignment Results

4K TV: Average Rating = 3.92.

Action Car: Average Rating = 3.55.

Cosine Similarity Calculation (Example for 4K TV and Action Car)

1. Multiply each pair of ratings for 4K TV and Action Car from users who have rated both items.

Example: If User 1 rated 4K TV with 3 and Action Car with 4, multiply 3 \* 4 = 12.

Repeat this for all users who have rated both items and add the results to get a sum.

1. Calculate the magnitude for each item’s ratings.

For 4K TV, square each rating (e.g., if ratings are 3, 5, 4, do 3^2 + 5^2 + 4^2).

Sum these squares and take the square root of this sum to get the magnitude for 4K TV.

Do the same for Action Car to get its magnitude.

1. Calculate Cosine Similarity by dividing the sum from Step 1 by the product of the magnitudes from Step 2.

Cosine Similarity = (Sum of products from Step 1) / (Magnitude of 4K TV \* Magnitude of Action Car).

Pearson Correlation Calculation (Example for 4K TV and Action Car)

1. Calculate the average rating for each item.

Average for 4K TV = (Sum of all ratings for 4K TV) / (Count of non-zero ratings for 4K TV).

Repeat this for Action Car.

1. Adjust each rating by subtracting the average rating.

For each user who rated 4K TV, subtract the average 4K TV rating from their rating to get the adjusted rating.

Repeat this for Action Car.

1. Multiply each pair of adjusted ratings for users who rated both items.

Example: If the adjusted rating for User 1 on 4K TV is 1.2 and on Action Car is 0.8, multiply 1.2 \* 0.8 = 0.96.

Sum these products for all users who rated both items.

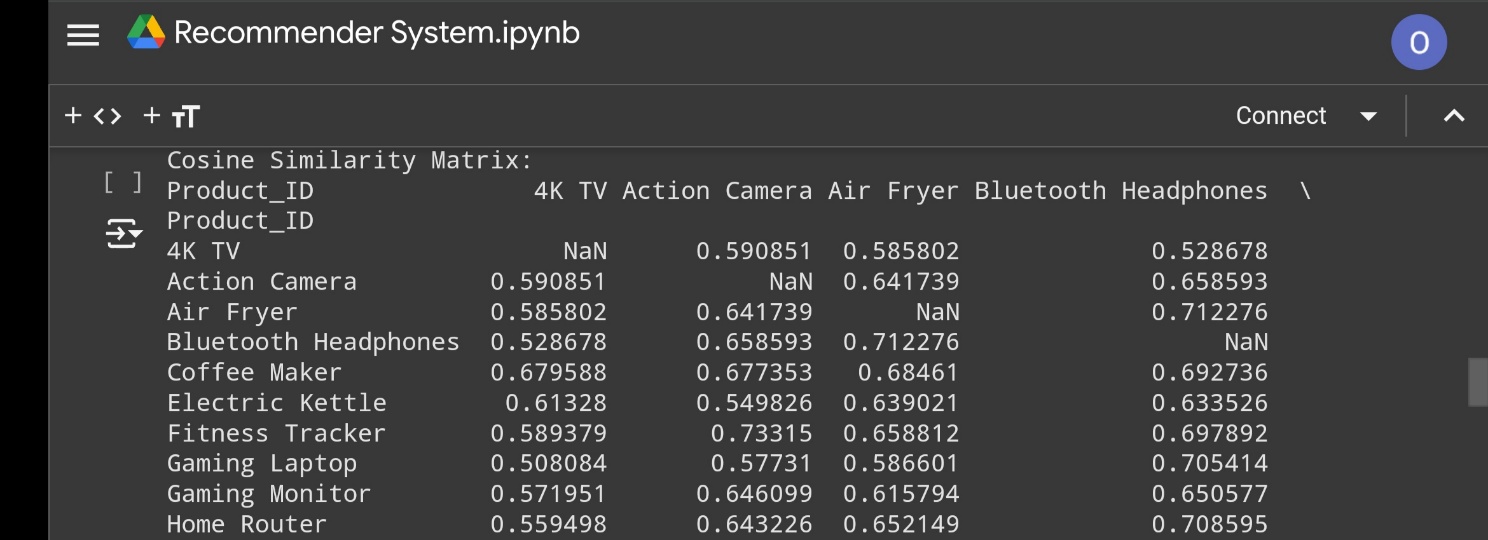
1. Calculate the square root of the sum of squared adjusted ratings for each item.

For 4K TV, square each adjusted rating, sum them up, and take the square root.

Repeat this for Action Car.

1. Calculate Pearson Correlation by dividing the sum of products from Step 3 by the product of the square roots from Step 4.

Pearson Correlation = (Sum of products from Step 3) / (Square root for 4K TV \* Square root for Action Car).



Rating Prediction Calculation

1. Numerator: For each similar item , calculate (similarity between 4K TV and item ) \times (User 1’s rating for item ) and sum these values.

2. Denominator: Sum the absolute values of the similarity scores between 4K TV and each similar item .

3. Predicted Rating: Divide Numerator by Denominator.

---

Top-N Recommendations

1. Predict ratings for all unrated items using the steps above.

2. Sort items by predicted ratings in descending order.

1. Select the top-N items with the highest predicted ratings.

