**Music Recommendation System**

**Objective**

To build a simple recommendation system that predicts songs a user is likely to enjoy based on their past interactions and the preferences of similar users.

**System Overview**

The system uses collaborative filtering to recommend songs. It analyzes the interaction data of users (ratings) and identifies patterns to suggest songs a user has not yet interacted with. The key steps include:

1. **Data Collection**: Simulated user-song interaction dataset.
2. **User-Item Matrix**: Representing users and their ratings for songs.
3. **Similarity Calculation**: Using cosine similarity to find users with similar preferences.
4. **Recommendation Generation**: Aggregating scores from similar users to recommend songs.

**Dataset**

**Simulated User-Song Interaction Data**

The dataset consists of the following columns:

* **user\_id**: Unique identifier for each user.
* **song\_id**: Unique identifier for each song.
* **rating**: User's feedback (e.g., number of plays, likes).

**Sample Data**

|  |  |  |
| --- | --- | --- |
| **user\_id** | **song\_id** | **rating** |
| 1 | 101 | 5 |
| 1 | 102 | 4 |
| 2 | 103 | 5 |
| 2 | 104 | 2 |

**Implementation Steps**

### ****1. Create the User-Item Matrix****

The User-Item Matrix represents the relationship between users and songs:

* Rows: Users
* Columns: Songs
* Values: Ratings (0 if unrated)

**Example Matrix**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **user\_id** | **101** | **102** | **103** | **104** | **105** |
| 1 | 5 | 4 | 3 | 0 | 0 |
| 2 | 4 | 0 | 5 | 2 | 0 |
| 3 | 0 | 3 | 0 | 5 | 0 |

2. Calculate User Similarity

Using **Cosine Similarity**, the system measures how similar users are based on their rating patterns. The similarity score ranges from 0 to 1:

* 1 indicates identical preferences.
* 0 indicates no similarity.

**The similarity matrix is computed as**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **User\_ID** | **1** | **2** | **3** | **4** |
| 1 | 1.0 | 0.89 | 0.76 | 0.65 |
| 1 | 0.89 | 1.0 | 0.82 | 0.70 |

**3. Generate Recommendations**

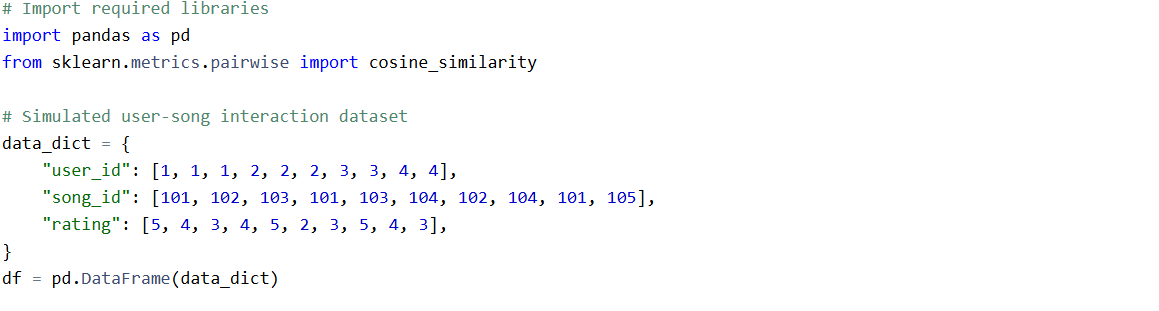
**Steps to generate recommendations:**

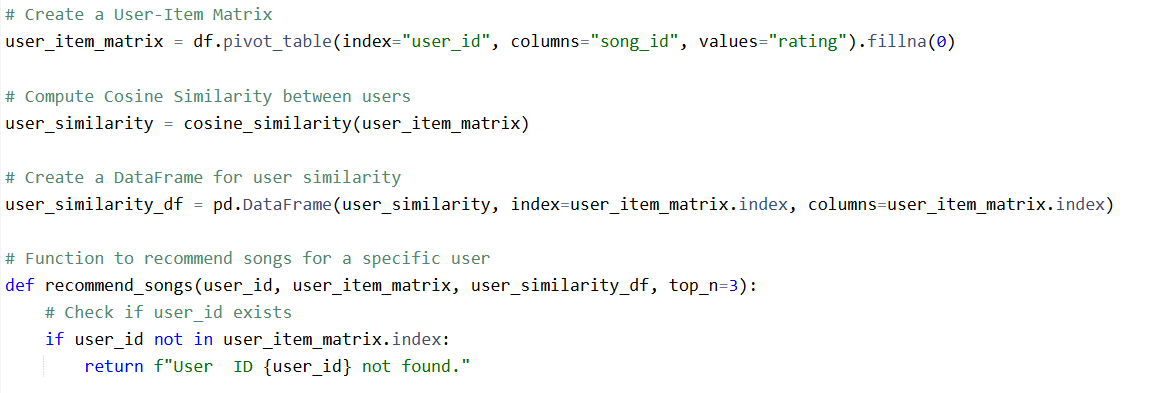
1. Identify songs the target user has not rated.
2. For each unrated song, calculate a **predicted score**:

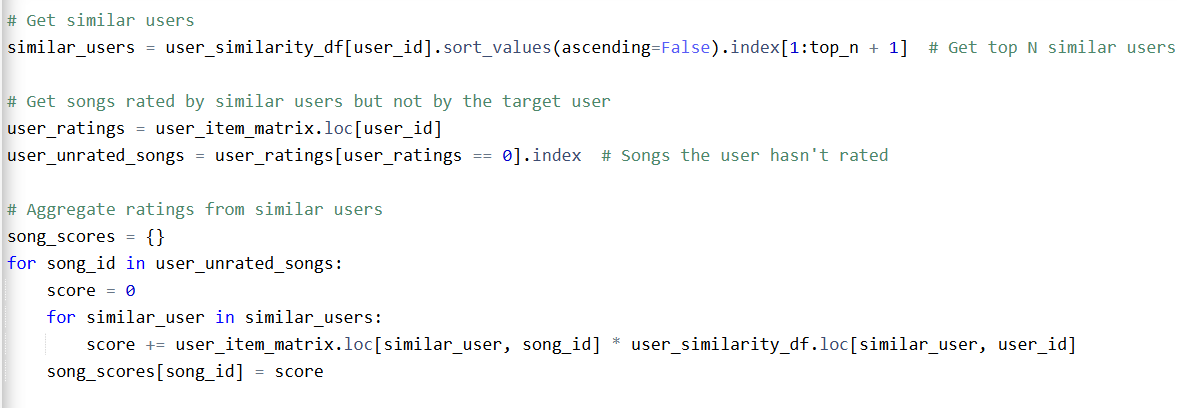
* Aggregate ratings from similar users, weighted by similarity scores.

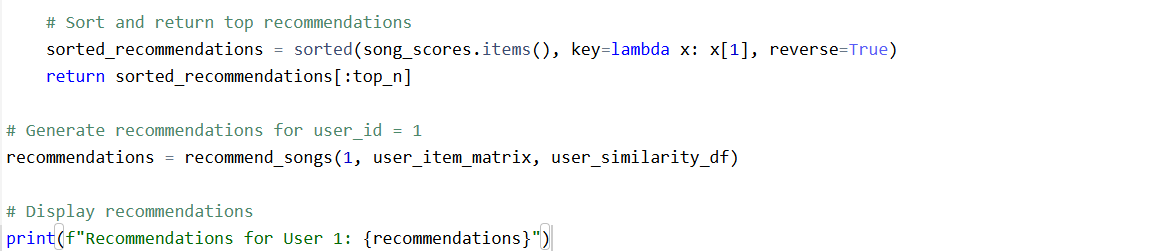
1. Sort the songs by predicted scores and recommend the top ones.

**4. Python Code**

****

****

****

****

### ****Output****

For **User 1**, the system suggests:

* **Song ID 104** with a predicted score of 2.93.
* **Song ID 105** with a predicted score of 1.70.

**Evaluation**

While this system is simple, it provides a baseline for recommendations. More advanced approaches (e.g., matrix factorization, deep learning) can be implemented for better accuracy.

**Future Improvements**

1. **Incorporate Song Metadata**: Use genre, tempo, and mood for better recommendations.
2. **Handle Cold Start Problem**: For new users or songs, use popularity or genre-based recommendations.
3. **Scalability**: Implement algorithms that handle larger datasets efficiently.
4. **UI integration:** Display recommendations on a user-friendly interface.