import pandas as pd

from sklearn.metrics.pairwise import cosine\_similarity

from sklearn.preprocessing import StandardScaler

# Sample user-movie ratings dataset

data = {

'user\_id': [1, 1, 1, 2, 2, 3, 3, 4, 4, 4],

'movie': ['Inception', 'Titanic', 'Avengers', 'Inception', 'Avengers',

'Titanic', 'Avengers', 'Inception', 'Titanic', 'Avengers'],

'rating': [5, 4, 5, 5, 4, 2, 4, 4, 5, 3]

}

df = pd.DataFrame(data)

# Pivot the data to get user-movie matrix

user\_movie\_matrix = df.pivot\_table(index='user\_id', columns='movie', values='rating').fillna(0)

# Standardize ratings for fair similarity

scaler = StandardScaler()

scaled\_matrix = scaler.fit\_transform(user\_movie\_matrix)

# Compute cosine similarity between users

user\_similarity = cosine\_similarity(scaled\_matrix)

# Store similarity in DataFrame for easier access

similarity\_df = pd.DataFrame(user\_similarity, index=user\_movie\_matrix.index, columns=user\_movie\_matrix.index)

def recommend\_movies(target\_user\_id, top\_n=2):

print(f"\n📍 Recommendations for User {target\_user\_id}")

# Get similar users

sim\_scores = similarity\_df[target\_user\_id].drop(index=target\_user\_id)

sim\_users = sim\_scores.sort\_values(ascending=False).head(top\_n).index

# Get movies rated by similar users, but not rated by target user

target\_user\_movies = df[df['user\_id'] == target\_user\_id]['movie'].tolist()

candidate\_movies = df[df['user\_id'].isin(sim\_users) & ~df['movie'].isin(target\_user\_movies)]

# Rank candidate movies by average rating

movie\_scores = candidate\_movies.groupby('movie')['rating'].mean().sort\_values(ascending=False)

print("🎬 Top recommended movies:")

for movie, score in movie\_scores.items():

print(f"- {movie} (avg rating from similar users: {score:.2f})")

# Example usage

recommend\_movies(target\_user\_id=3)