```python
import pandas as pd
from sklearn.metrics.pairwise import co
from sklearn.feature_extraction.text im

# Sample Dataset
data = {
    'MovieID': [1, 2, 3, 4, 5],
    'Title': ['The Matrix', 'Inception'
    'Genre': ['Action Sci-Fi', 'Sci-Fi
}

movies_df = pd.DataFrame(data)

# User Ratings Matrix (example)
ratings_data = {
    'UserID': [1, 1, 2, 2, 3, 3, 4, 4],
    'MovieID': [1, 2, 2, 3, 3, 4, 4, 5]
    'Rating': [5, 4, 5, 4, 5, 4, 4, 3]
}

ratings_df = pd.DataFrame(ratings_data)

# Merge movies and ratings
movie_ratings = pd.merge(ratings_df, mo

# Create a User-Item Matrix
user_movie_matrix = movie_ratings.pivot

# Compute Cosine Similarity between use
user_similarity = cosine_similarity(use
user_similarity_df = pd.DataFrame(user_

# Recommendation Function
def recommend_movies(user_id, user_movi
    """
    Recommend movies to a user based on
    Args:
        user_id: The ID of the user to
        user_movie_matrix: User-item ma
        user_similarity_df: User simila
```

```python
# Recommendation Function
def recommend_movies(user_id, user_movi
    """
    Recommend movies to a user based on
    Args:
        user_id: The ID of the user to
        user_movie_matrix: User-item ma
        user_similarity_df: User simila
        movies_df: Original movies Data
        top_n: Number of recommendation
    Returns:
        List of recommended movie title
    """
    # Find similar users
    similar_users = user_similarity_df[

    # Weighted ratings based on similar
    similar_users_ratings = user_movie_

    # Movies already rated by the user
    user_rated_movies = user_movie_matr

    # Filter out movies already rated b
    recommendations = similar_users_rat

    # Sort movies by predicted rating
    recommended_movies = recommendation
    return list(recommended_movies.inde

# Example Usage
user_id = 1
recommendations = recommend_movies(user
print(f"Recommended movies for User {us
```

```
Recommended movies for User 1: ['Interstellar', 'Shutter Island', 'Avatar']
```