In [2]:

```
import pandas as pd
import numpy as np
from sklearn.metrics.pairwise import cosine_similarity


ratings = pd.read_csv('ratings.csv')
movies = pd.read_csv('movies.csv')


print("Ratings Data Head:")
print(ratings.head())
print("\nMovies Data Head:")
print(movies.head())


filtered_ratings = ratings[ratings['userId'] <= 1000]
filtered_ratings = filtered_ratings[filtered_ratings['movieId'] <= 1000]


filtered_ratings = filtered_ratings.dropna(subset=['userId', 'movieId', 'ra


filtered_ratings['userId'] = filtered_ratings['userId'].astype(int)
filtered_ratings['movieId'] = filtered_ratings['movieId'].astype(int)


try:
    user_movie_matrix = filtered_ratings.pivot(index='userId', columns='mov
except ValueError as e:
    print("Error creating the user-item interaction matrix:", e)
    print("Check if there are any inconsistencies in your data.")


user_similarity = cosine_similarity(user_movie_matrix)
user_similarity_df = pd.DataFrame(user_similarity, index=user_movie_matrix.


def recommend_movies(user_id, num_recommendations=5):
    if user_id not in user_similarity_df.index:
        print(f"User ID {user_id} not found in the dataset.")
        return []

    similar_users = user_similarity_df[user_id].sort_values(ascending=False
    similar_users_ratings = user_movie_matrix.loc[similar_users]
    recommended_movies = similar_users_ratings.mean(axis=0).sort_values(asc

    return movies[movies['movieId'].isin(recommended_movies)]['title'].valu


user_id = 1
recommendations = recommend_movies(user_id)
print(f"Recommendations for User {user_id}:")
print(recommendations)
```

```
Ratings Data Head:
   userId  movieId  rating   timestamp
0       1        2     3.5  1112486027
1       1       29     3.5  1112484676
2       1       32     3.5  1112484819
3       1       47     3.5  1112484727
4       1       50     3.5  1112484580

Movies Data Head:
   movieId                               title  \
0        1                    Toy Story (1995)
1        2                      Jumanji (1995)
2        3             Grumpier Old Men (1995)
3        4            Waiting to Exhale (1995)
4        5  Father of the Bride Part II (1995)

                                        genres
0  Adventure|Animation|Children|Comedy|Fantasy
1                   Adventure|Children|Fantasy
2                             Comedy|Romance
3                       Comedy|Drama|Romance
4                                     Comedy
Recommendations for User 1:
['Star Wars: Episode IV - A New Hope (1977)' 'Pulp Fiction (1994)'
 'Shawshank Redemption, The (1994)' 'Forrest Gump (1994)'
 'Silence of the Lambs, The (1991)']
```

In [3]:
```
from sklearn.feature_extraction.text import TfidfVectorizer


movies = pd.read_csv('movies.csv')  # MovieID, Title, Genres
ratings = pd.read_csv('ratings.csv')  # UserID, MovieID, Rating


tfidf = TfidfVectorizer(stop_words='english')
movies['genres'] = movies['genres'].fillna('')
tfidf_matrix = tfidf.fit_transform(movies['genres'])


def create_user_profile(user_id):
    user_ratings = ratings[ratings['userId'] == user_id]
    user_rated_movies = movies[movies['movieId'].isin(user_ratings['movieId
    user_profile = tfidf.transform(user_rated_movies['genres'])
    user_profile_weights = user_ratings['rating'].values.reshape(-1, 1)
    user_profile = user_profile.multiply(user_profile_weights).sum(axis=0)
    return user_profile


def recommend_movies_content(user_id, num_recommendations=5):
    user_profile = create_user_profile(user_id)
    similarities = cosine_similarity(user_profile, tfidf_matrix)
    similar_indices = similarities.argsort().flatten()[-num_recommendations
    return movies.iloc[similar_indices]['title'].values


user_id = 1
recommendations = recommend_movies_content(user_id)
print(f"Recommendations for User {user_id}:")
print(recommendations)
```

```
Recommendations for User 1:
['Dragonheart 2: A New Beginning (2000)' 'Army of Darkness (1993)'
 'She (1965)' "Cirque du Freak: The Vampire's Assistant (2009)"
 'Mummy, The (1999)']

C:\Users\Vaishnavi\anaconda3\lib\site-packages\sklearn\utils\validation.p
y:593: FutureWarning: np.matrix usage is deprecated in 1.0 and will raise
a TypeError in 1.2. Please convert to a numpy array with np.asarray. For m
ore information see: https://numpy.org/doc/stable/reference/generated/nump
y.matrix.html (https://numpy.org/doc/stable/reference/generated/numpy.matr
ix.html)
  warnings.warn(
```

In [ ]: