Movie Recommendation System Report

1. Introduction

For this project, built three different movie recommendation systems to compare how they work and what kinds of suggestions they generate. Implemented user-based collaborative filtering, item-based collaborative filtering, and a graph-based random walk approach. Each method has its own strengths - some are better for personalized recommendations, while others are better for discovering new content.

2. Dataset Description

Worked with the MovieLens 100K dataset which has:

· 943 different users

· 1,682 movies

· 100,000 ratings from 1 to 5 stars

· User info like age and occupation

· Movie titles and release dates

Before building the recommendation systems, had to clean the data:

· Replaced missing ratings with 0 (meaning 'not rated' instead of 'bad rating')

· Fixed the timestamp format to make it readable

· For the graph method, combined ratings with movie titles and adjusted ratings to account for users who always rate high or low

3. Methodology

User-Based Filtering

This method finds users with similar movie tastes and recommends movies they liked that haven't watched. Calculated how similar users are using cosine similarity and picked the top 5 most similar users to get recommendations from.

Item-Based Filtering

This finds movies similar to ones already liked. Instead of comparing users, Compared movies based on how users rated them. Movies with similar rating patterns are considered similar. Added error handling since sometimes the math can have issues with zero ratings.

Graph Random Walks

Built a network where users connect to movies they rated. Then, simulated random walks through this network - starting from a user or movie and randomly moving to connected nodes. Movies that get visited more often during these walks become recommendations.

4. Implementation Details

For the collaborative filtering methods, created a big table with users as rows and movies as columns, then filled in their ratings. This led to calculate similarities between users and between movies.

For the graph approach, built a network with 2,625 nodes - 943 users and 1,682 movies. User 1 rated 272 movies, and movie 50 was rated by 583 users, showing the network is well-connected.

The random walk function lets a control how long the walk is and how many recommendations to return. It counts how many times each movie gets visited and ranks them by visit count.

5. Results and Evaluation

User-Based Results:

Worked correctly. For user 10, it recommended exactly the movies mentioned in the assignment: ‘In the Company of Men’, ‘Misérables, Les’, etc. The results were consistent every time running it.

Item-Based Results:

For ‘Jurassic Park’, it recommended similar action movies like ‘Top Gun’, ‘Raiders of the Lost Ark’, and Speed. The movies made sense together, though the order was slightly different than the example.

Graph-Based Results:

In this case, the recommendations changed each time and were more diverse. For user 1, got ‘Men in Black’ and ‘That Thing You Do!’, while ‘Jurassic Park’ led to unexpected suggestions like ‘Emma’ and ‘Sabrina’. Shorter walks gave more obvious connections, longer walks found more new matches.

6. Conclusion and Future Work

All three methods worked well but serve different purposes. User-based is great for reliable personalized recommendations, item-based is good for 'if you liked this, try these' suggestions, and graph-based is best for discovering new content that might not find otherwise.

Hybrid System Ideas:

If continued - building hybrid systems that combine the best parts of each method:

1. Weighted Combination - Mix recommendations from all three methods with different weights (like 40% user-based, 40% item-based, 20% graph-based) to get both accuracy and different results

2. Smart Switching - Use different methods for different situations:

 · New users ->item-based (does not need rating history)

 · Regular users ->user-based (personalized)

 · Exploration mode -> graph-based (discovery)

3. Cascade System - First getting reliable recommendations from user-based, then add some new suggestions from graph-based, and finally removing duplicates

4. Feature Blending - Training a simple model to learn when to trust each method based on what works best for different users and movies

Reasons Hybrid works better

· User-based alone: Accurate but predictable

· Item-based alone: Good for similarity but misses personal touch

· Graph-based alone: Good for discovery but inconsistent

· Combined: getting accurate + similar + new results

These techniques could work for any recommendation system - movies, products, music, articles etc..