Akanimoh Umoren

ITCS 6162, Data Mining

4/15/2025

MOVIE RECOMMENDATION SYSTEM REPORT

INTRODUCTION

Movie recommendation systems are systems used to generate content that may be relevant to a user based on the content they have previously interacted with or other similar users. Recommendation algorithms use various approaches, including user-based, item-based, and random-walk methods, to provide these suggestions.

User-based collaborative filtering relies on the idea that users who have similar preferences will like similar items. It identifies users whose tastes closely match the target user's and recommends items that those similar users have liked. Item-based collaborative filtering, on the other hand, finds similarities between items. It recommends items that are similar to those the user has rated highly. Finally, the random-walk-based Pixie algorithm uses a graph-based approach. By performing random walks across a graph, Pixie-inspired systems can explore hidden relationships between users and items, uncovering relevant recommendations based on these indirect connections.

Recommender systems are important because with millions of items available to choose from, recommending the right movie can be a daunting task. Recommendation systems alleviate this issue by leveraging data from user preferences, item interactions, and content features

DATASET DESCRIPTION

The MovieLens 100K dataset used in this project consists of 100,000 ratings from 943 users on 1,682 movies. The dataset includes several key features:

- Users: 943 unique users, each providing a series of ratings for movies.

- Movies: 1,682 movies across various genres.
- Ratings: Each user rates a movie on a scale of 1 to 5, with a total of 100,000 ratings.

The **features** of the dataset include:

- user_id: Identifier for the user.
- movie_id: Identifier for the movie.
- rating: The rating given by the user to the movie.
- timestamp: The time at which the rating was provided.
- title: The title of the movie.

Preprocessing performed:

- Missing data was handled by removing rows with NaN values.
- The ratings dataset was merged with the movies dataset to provide human-readable movie titles.
- Grouping and normalization of ratings were carried out to ensure fairness and remove individual biases, particularly in collaborative filtering.

METHODOLOGY

In this project, we have implemented three main recommendation techniques:

1. User-Based Collaborative Filtering: In this method, user similarity is calculated using cosine similarity based on ratings. The algorithm identifies users who have similar preferences and recommends movies that these similar users have liked. By normalizing ratings, we account for individual biases, making the similarity calculations more accurate.

2. Item-Based Collaborative Filtering: For item similarity, we again use cosine similarity to measure how similar movies are based on the ratings they've received from users. This method looks at the movies that a user has rated and recommends other movies that are similar. This helps suggest contextually relevant items, even if the user hasn't rated them directly.

3. Random-Walk-Based Pixie Algorithm: Pixie-inspired algorithms use a graph-based approach, where users and movies are represented as nodes in a bipartite graph. The random walk starts from either a movie or a user and explores the graph by moving to connected nodes. The movies that are visited most frequently during the random walk are considered the most relevant recommendations. This method is useful for uncovering latent relationships between users and movies that aren't captured by traditional similarity measures.

IMPLEMENTATION DETAILS

Building the functions for the three recommendation methods involved several key steps:

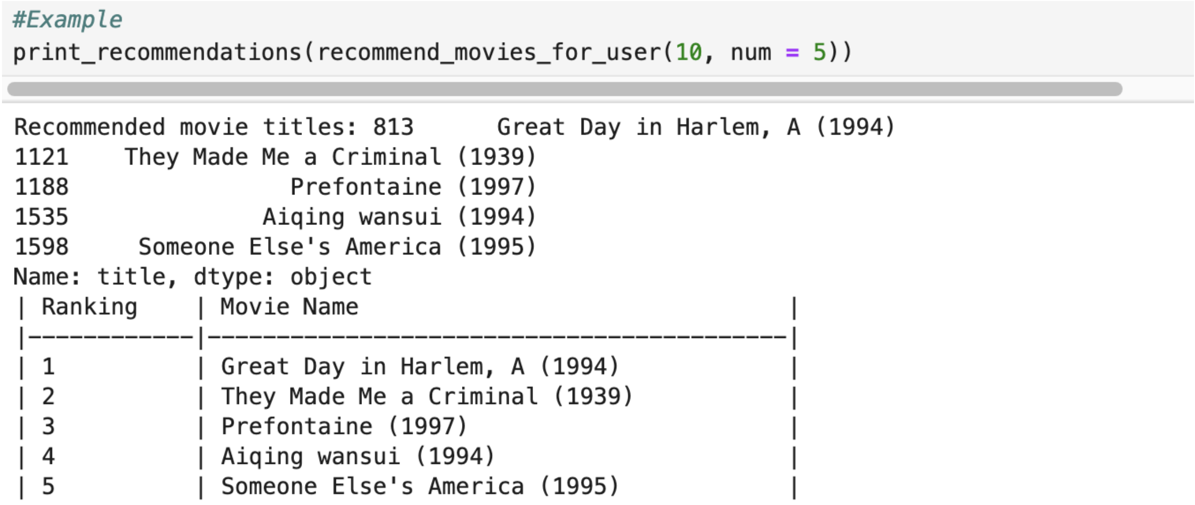
1. User-Based Collaborative Filtering: The function first calculates the cosine similarity between users based on their ratings. After normalizing ratings, it ranks the most similar users and recommends movies based on those rankings.

2. Item-Based Collaborative Filtering: This method works by calculating the similarity between movies. We then rank the movies based on similarity and recommend the top n (n is the integer passed for requested recommendations) most similar movies to the user

3. Pixie Algorithm: The adjacency list graph was created by connecting users to the movies they rated and movies to the users who rated them. The random walk was performed by randomly visiting a connected node at each step, and the number of visits to each movie was recorded. Movies visited most frequently during the walk were recommended.

RESULT AND EVALUATION

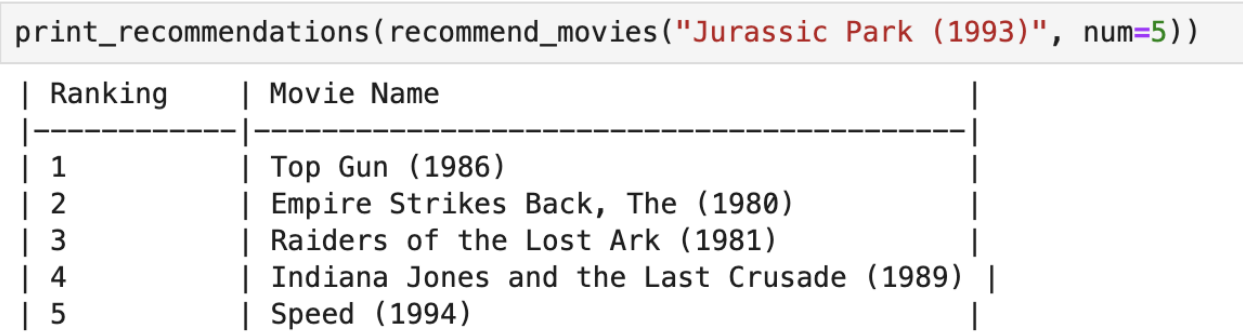
1. User-based Collaborative Filtering:



```
#Example
print_recommendations(recommend_movies_for_user(10, num = 5))
```

```
Recommended movie titles: 813      Great Day in Harlem, A (1994)
1121    They Made Me a Criminal (1939)
1188                Prefontaine (1997)
1535              Aiqing wansui (1994)
1598     Someone Else's America (1995)
Name: title, dtype: object
| Ranking    | Movie Name                               |
|------------|------------------------------------------|
| 1          | Great Day in Harlem, A (1994)            |
| 2          | They Made Me a Criminal (1939)           |
| 3          | Prefontaine (1997)                       |
| 4          | Aiqing wansui (1994)                     |
| 5          | Someone Else's America (1995)            |
```

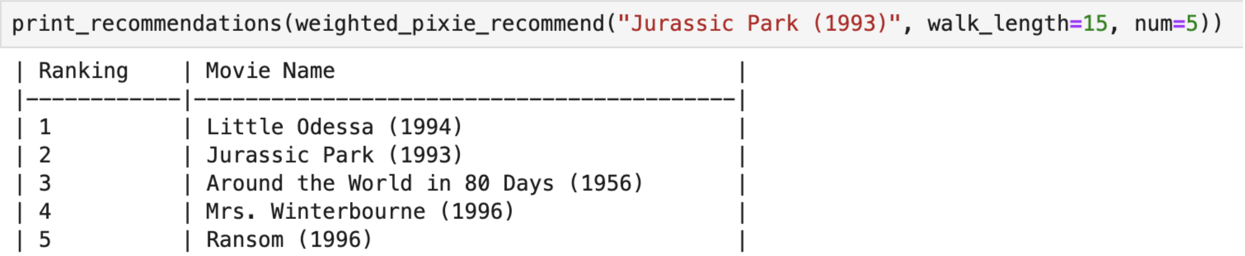
2. Item-based Collaborative Filtering:



```
print_recommendations(recommend_movies("Jurassic Park (1993)", num=5))
```

```
| Ranking     | Movie Name                                  |
|------------|------------------------------------------|
| 1           | Top Gun (1986)                              |
| 2           | Empire Strikes Back, The (1980)             |
| 3           | Raiders of the Lost Ark (1981)              |
| 4           | Indiana Jones and the Last Crusade (1989) |
| 5           | Speed (1994)                                |
```

3. Random Walk-Based Movie Recommendation System



```
print_recommendations(weighted_pixie_recommend("Jurassic Park (1993)", walk_length=15, num=5))
```

```
| Ranking     | Movie Name                                  |
|------------|------------------------------------------|
| 1           | Little Odessa (1994)                        |
| 2           | Jurassic Park (1993)                        |
| 3           | Around the World in 80 Days (1956)          |
| 4           | Mrs. Winterbourne (1996)                    |
| 5           | Ransom (1996)                               |
```

Comparison:

- The user-based method recommended movies by using similarities between users, but it required enough overlapping ratings between users to work well.
- The item-based method focused on finding similarity between movies, which can be more effective when there's limited overlap between users.
- The Pixie algorithm explored the graph and found hidden connections, providing diverse recommendations, but the random walk's effectiveness can be impacted by the walk length and graph sparsity.

Limitations:

- All methods suffer from the cold start problem, where new users or movies with insufficient ratings may not receive accurate recommendations.
- Sparse data can reduce the effectiveness of collaborative filtering methods, especially when only a few users rate a given movie.

CONCLUSION

The project successfully implemented three different recommendation techniques: user-based collaborative filtering, item-based collaborative filtering, and the Pixie-inspired random walk algorithm. Each method had its strengths, and together they provided diverse and relevant recommendations.

Potential Improvements:

- Combining the methods into a hybrid model could provide even more accurate recommendations by leveraging the strengths of each approach.
- Additional features like movie genres, user demographics, and temporal data could enhance the system's performance.

Real-World Applications:

These algorithms are widely used in platforms like Netflix, Amazon, and YouTube to recommend content based on user behavior and preferences. They help increase user engagement and satisfaction by offering relevant, personalized recommendations.