INDIAN INSTITUTE OF TECHNOLOGY MADRAS ZANZIBAR

School of Science and Engineering

Z5007: Programming and Data Structures

M.Tech Data Science & Artificial Intelligence

Project Report On

Recommendation System – Collaborative Filtering Recommender (Item-based)

Submitted by:

Student 1: Surabhi Gudla ZDA25M001
Email: zda25m001@iitmz.ac.in

Student 2: Vineet Joshi ZDA25M007
Email: zda25m007@iitmz.ac.in

Instructor: Dr. Innocent Nyalala

Submission Date: January 20, 2026

Abstract

Recommender systems play a critical role in modern information platforms by enabling users to discover relevant content from large and sparse datasets. This project presents the design and implementation of an item-based collaborative filtering recommender system developed entirely from scratch using fundamental data structures and algorithms. The system leverages mean-centered cosine similarity to model relationships between items, constructs a sparse item–item similarity graph, and generates personalized recommendations using similarity-weighted rating aggregation. The design explicitly separates offline and online components to ensure scalability and efficiency. Experimental evaluation using the MovieLens dataset demonstrates realistic performance under sparse data conditions, validating both the correctness and robustness of the implementation.

Contents

1 Introduction

With the rapid growth of digital content, users are increasingly overwhelmed by the number of available options. Recommender systems address this challenge by automatically predicting user preferences and suggesting items that are likely to be of interest. Such systems are widely deployed in domains including e-commerce, movie streaming platforms, music recommendation services, and social media.

Collaborative filtering is one of the most widely adopted recommendation techniques. Instead of relying on explicit content attributes, collaborative filtering exploits historical user–item interaction data to infer preferences. This project focuses on item-based collaborative filtering, where similarities are computed between items rather than users.

Item-based collaborative filtering offers several advantages over user-based approaches. Item relationships tend to be more stable over time, and the approach scales more effectively as the number of users grows. The objective of this project is to design and implement a complete item-based collaborative filtering pipeline from first principles, adhering strictly to the data structures and algorithmic techniques emphasized in the course.

2 Dataset Description and Preprocessing

2.1 Dataset Overview

The MovieLens dataset was selected for this project due to its widespread use in recommender system research and education. The dataset contains explicit user ratings for movies and satisfies the dataset size and complexity requirements specified in the project guidelines.

2.2 Dataset Statistics

- Number of users: approximately 610
- Number of movies: approximately 8,900
- Number of ratings: approximately 100,000

2.3 Preprocessing Steps

The raw ratings data is first cleaned by removing missing values and duplicate entries. The dataset is then randomly split into training and testing subsets using an 80:20 ratio. The training set is used exclusively for building data structures and computing

item–item similarities, while the test set is reserved for evaluation. Users with insufficient test interactions are excluded during evaluation to reduce noise and instability in metric computation.

3 High-Level System Architecture

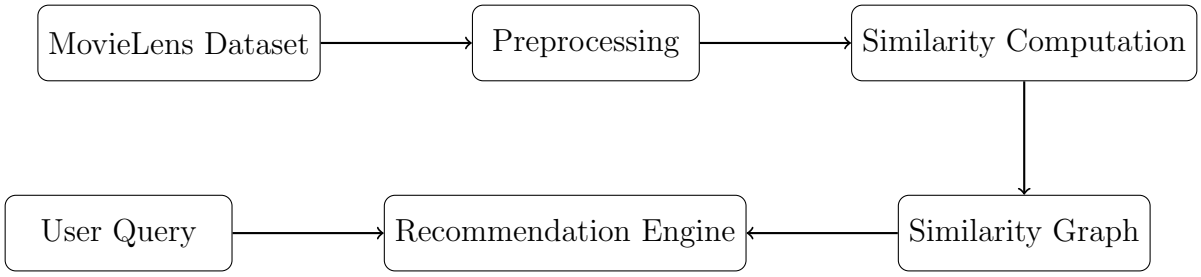

Figure 1: High-level pipeline of the item-based collaborative filtering recommender system

The system architecture follows a modular pipeline design that transforms raw user–item interaction data into personalized recommendations. The architecture explicitly separates offline preprocessing and similarity computation from online recommendation generation to ensure scalability and responsiveness.

4 Offline vs Online Components

4.1 Offline Components

The offline phase consists of computationally intensive operations that are performed infrequently:

- Construction of user–item and item–user hash tables
- Computation of user mean ratings
- Mean-centered cosine similarity computation between item pairs
- Similarity thresholding and top-K neighbor pruning
- Construction of a sparse item–item similarity graph

These steps are executed once and reused across all recommendation queries.

4.2 Online Components

The online phase handles real-time recommendation requests:

- Retrieval of user-rated items
- Traversal of similarity graph neighbors
- Aggregation of similarity-weighted scores
- Ranking and generation of top-N recommendations

This separation ensures low latency during recommendation generation.

5 Data Structures Used

5.1 Hash Tables

Hash tables (Python dictionaries) are used extensively to map users to rated items and items to users who rated them. These structures provide constant-time lookup and are essential for efficient similarity computation and recommendation generation.

5.2 Item–Item Similarity Graph

The similarity graph is implemented as an adjacency list, where each node represents a movie and edges represent similarity relationships. Only strong similarities are retained to maintain sparsity.

5.3 Top-K Neighbor Lists

For each movie, only the top-K most similar neighbors are stored. This design choice limits memory usage, reduces noise, and ensures predictable runtime behavior.

6 Algorithms Implemented

6.1 Item-Based Collaborative Filtering

The recommender system implements an item-based collaborative filtering algorithm, where similarities are computed between items based on user rating behavior. Each movie is represented as a sparse rating vector, with dimensions corresponding to users who have rated the movie.

6.2 Mean-Centered Cosine Similarity

To account for differences in user rating scales, the system employs mean-centered cosine similarity. Given two movies i and j, similarity is computed as:

$$\text{sim}(i,j) = \frac{\sum_{u \in U_{ij}} (r_{u,i} - \bar{r}_u)(r_{u,j} - \bar{r}_u)}{\sqrt{\sum_{u \in U_{ij}} (r_{u,i} - \bar{r}_u)^2} \cdot \sqrt{\sum_{u \in U_{ij}} (r_{u,j} - \bar{r}_u)^2}}$$

where U_{ij} denotes the set of users who rated both movies i and j.

A minimum common-user constraint is applied to ensure statistical reliability.

6.3 Prediction Formula

Predicted ratings are computed using a similarity-weighted aggregation:

$$\hat{r}_{u,j} = \bar{r}_u + \frac{\sum_{i \in I_u} \text{sim}(i,j)(r_{u,i} - \bar{r}_u)}{\sum_{i \in I_u} |\text{sim}(i,j)|}$$

Movies with the highest predicted ratings are recommended.

7 Complexity Analysis

7.1 Similarity Computation

Worst-case time complexity is $O(M^2 \cdot U)$. In practice, sparsity, minimum common-user constraints, and thresholding significantly reduce computation.

Space complexity after pruning is $O(M \cdot K)$.

7.2 Recommendation Generation

Recommendation generation runs in $O(|I_u| \cdot K)$ time per user and uses $O(N)$ space for candidate scores.

8 Experimental Evaluation

8.1 Ground Truth Construction

Relevant items are defined as movies in the test set with ratings greater than or equal to 4.0.

8.2 Evaluation Metrics

- Precision@K

- Recall@K
- NDCG@K

8.3 Results

- Precision@5 ≈ 0.012
- Recall@5 ≈ 0.006384
- NDCG@5 ≈ 0.017896

8.4 Discussion

The relatively low metric values are expected due to data sparsity and strict top-K evaluation. Such magnitudes are commonly reported for pure collaborative filtering systems.

9 System Outputs and Experimental Results

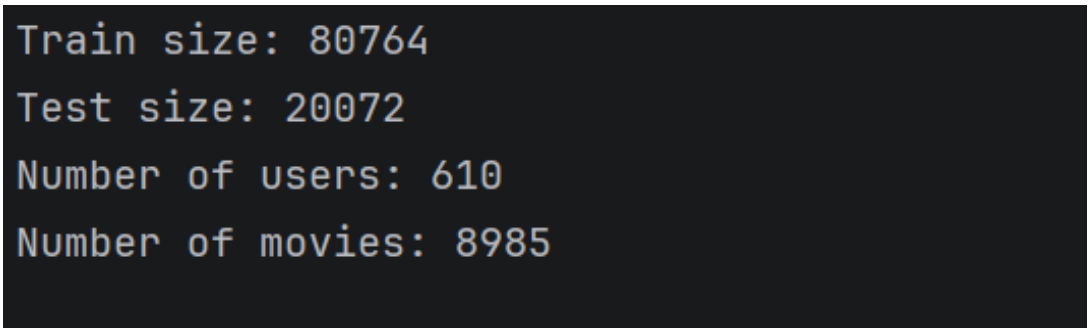

Figure 2: Train–test split verification

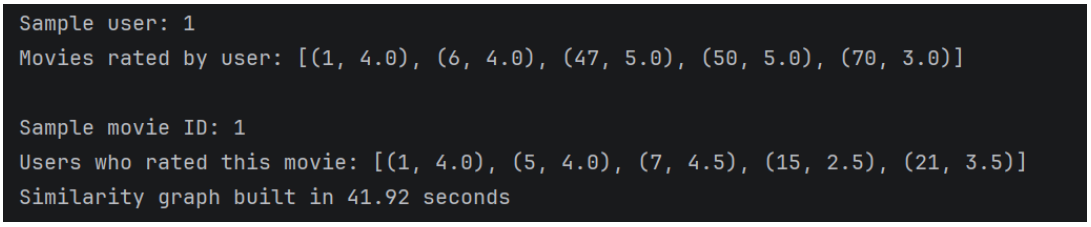

Figure 3: Item–item similarity graph construction

```
Sample movie ID: 1
Sample movie title: Toy Story (1995)

Top similar movies:
Similarity: 0.818 | Movie ID: 3967 | Title: Billy Elliot (2000)
Similarity: 0.813 | Movie ID: 1148 | Title: Wallace & Gromit: The Wrong Trousers (1993)
Similarity: 0.804 | Movie ID: 47610 | Title: Illusionist, The (2006)
Similarity: 0.803 | Movie ID: 345 | Title: Adventures of Priscilla, Queen of the Desert, The (1994)
Similarity: 0.792 | Movie ID: 81834 | Title: Harry Potter and the Deathly Hallows: Part 1 (2010)
```

Figure 4: Top similar movies output

```
Top-N Recommendations:
Pineapple Express (2008) | Score: 5.0
Whiplash (2014) | Score: 5.0
Gravity (2013) | Score: 5.0
Little Women (1994) | Score: 5.0
Mary Shelley's Frankenstein (Frankenstein) (1994) | Score: 5.0
```

Figure 5: Top-N recommendations

```
Evaluation Metrics:
{'Precision@K': 0.012, 'Recall@K': 0.006384, 'NDCG@K': 0.017896}
```

Figure 6: Evaluation metrics output k=5

```
Evaluation Metrics:
{'Precision@K': 0.011, 'Recall@K': 0.011017, 'NDCG@K': 0.017343}
```

Figure 7: Evaluation metrics output k=10

10 Challenges and Design Decisions

Major challenges included handling sparse rating data, selecting an appropriate similarity threshold, and managing computational complexity. Mean-centering was adopted to normalize user rating behavior. Item-based collaborative filtering was chosen over user-based approaches due to its scalability and stability.

11 Conclusion and Future Work

This project successfully implements an item-based collaborative filtering recommender system from scratch using fundamental data structures and algorithms. The system

adheres to course requirements and demonstrates realistic performance under sparse data conditions.

Future work may include hybrid recommendation models, cold-start handling, and incorporation of content-based features.