MOVIE RECOMMENDER SYSTEMS

Phase 4 Project : Group 5

Evelyn Mwangi
Hakima Ibrahim
Khadija Hussein
Wilberforce Kirui
Edel Lwoyelo
Mark Chiuri

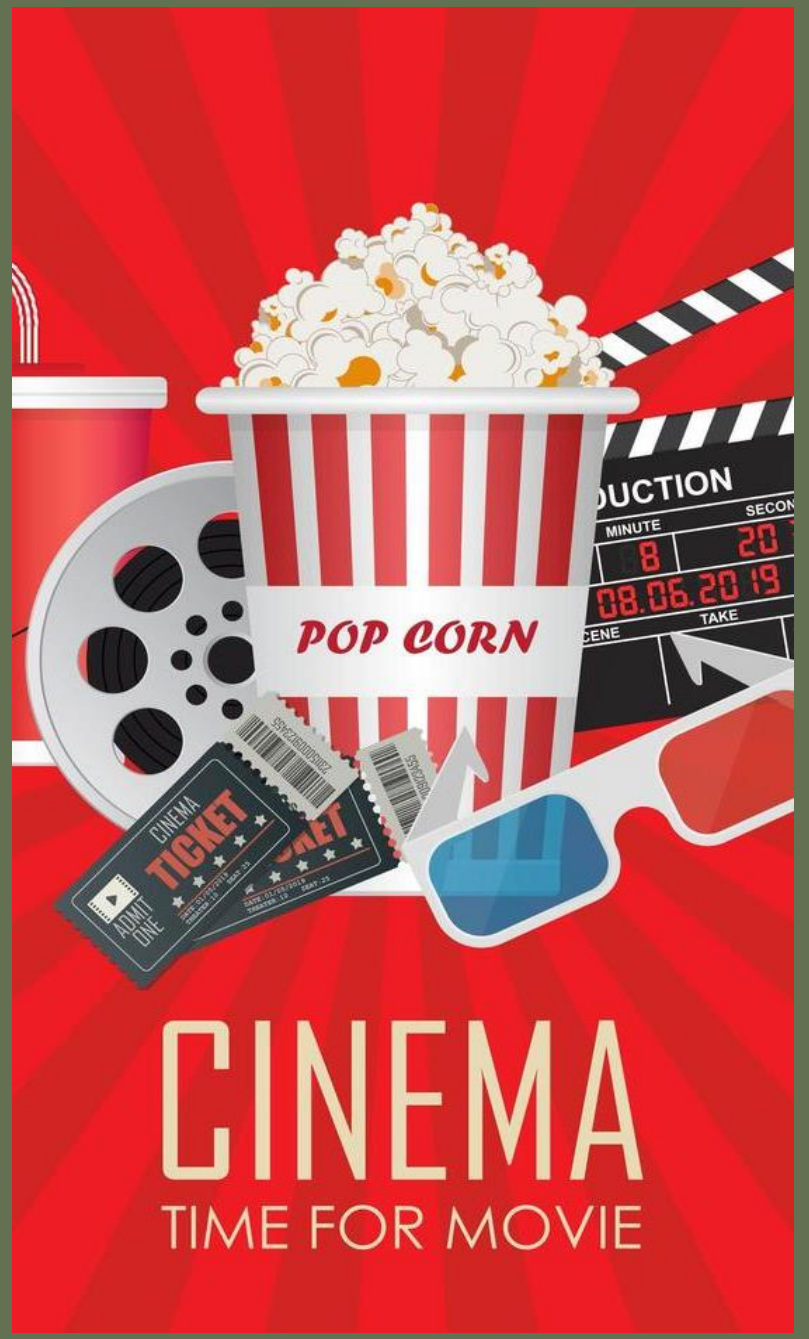
Contents

Overview and Business Problem

This project develops a movie recommender system using the Movie Lens dataset.

The Movie Lens dataset is a popular benchmark dataset in the field of recommender systems. It provides rich information about movies, user ratings, tags, and links to external movie databases.

This project aims to provide valuable insights into user behavior and movie preferences, ultimately leading to improved movie recommendation



Objectives and Business Questions

- Understand how users form beliefs about movies they haven't seen.
- How do users form beliefs or expectations about movies they haven’t watched yet?

- Investigate the relationship between user beliefs, past ratings, and received recommendations.
- What factors (e.g., genres, tags, ratings, recommendations) most influence these beliefs?

- Evaluate how well recommendation systems align with and influence user beliefs.
- How closely do user beliefs align with their actual ratings after watching a movie?

- Study user decision-making processes in the context of movie consumption.
- How do past user ratings and behaviors affect the relevance of future recommendations?

- To what extent do recommendations influence or shape user expectations and decisions?

Data Understanding

1. Movies Data (movies.csv):

- Contains movie information, including titles and genres.
- movieId: Unique identifier for each movie.
- Title: The title of the movie, which also includes the year of release in parentheses.
- Genres: A pipe-separated list of genres to categorize the movie (e.g.,
- Action|Adventure|Comedy).

2. Links Data (links.csv):

- Provides identifiers for linking to external movie-related sources (IMDb, TMDb).
- MovieId: Unique identifier for each movie, consistent with other data files.
- ImdbId: Identifier for movies used by IMDb (Internet Movie Database).
- TmdbId: Identifier for movies used by TMDb (The Movie Database).

3.Ratings Data (ratings.csv):

- Each entry represents a user's rating for a specific movie.
- Contains user ratings on a 5-star scale for movies.
- UserId: ID representing the unique identifier for each user.
- MovieId: Unique identifier for each movie.
- Rating: User's rating for the movie on a 5-star scale with half-star increments (0.5 to 5.0).
- Timestamp: The timestamp when the rating was recorded, represented in seconds since midnight Coordinated Universal Time (UTC) of January 1, 1970.

4. Tags Data (tags.csv):

Contains user-generated metadata (tags) about movies.

UserId: ID representing the unique identifier for each user.

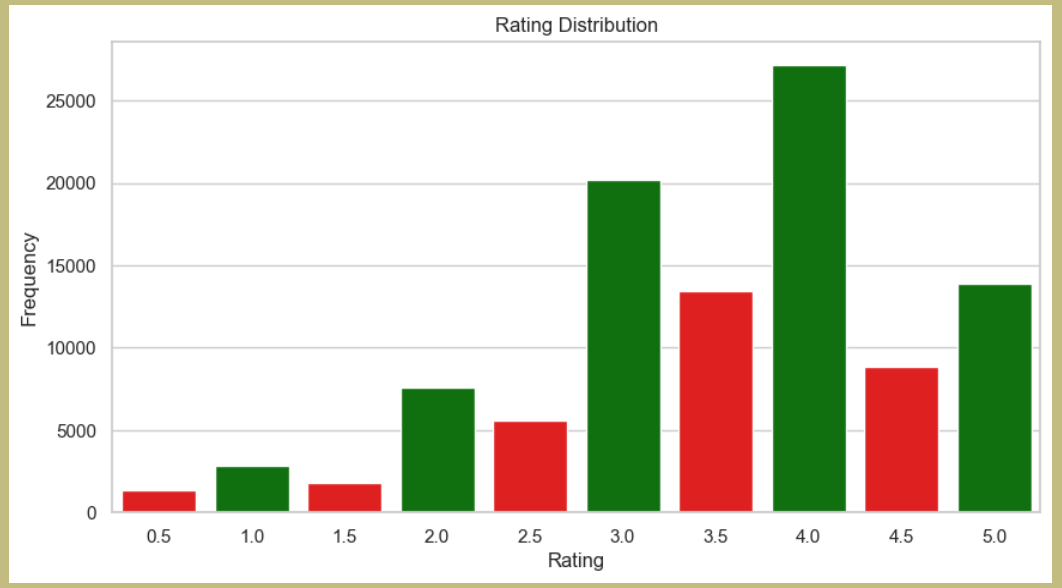
MovieId: Unique identifier for each movie.

Tag: User-generated metadata describing a movie, typically a single word or short phrase.

Timestamp: The timestamp when the tag was applied, represented in seconds since midnight Coordinated Universal Time (UTC) of January 1, 1970.

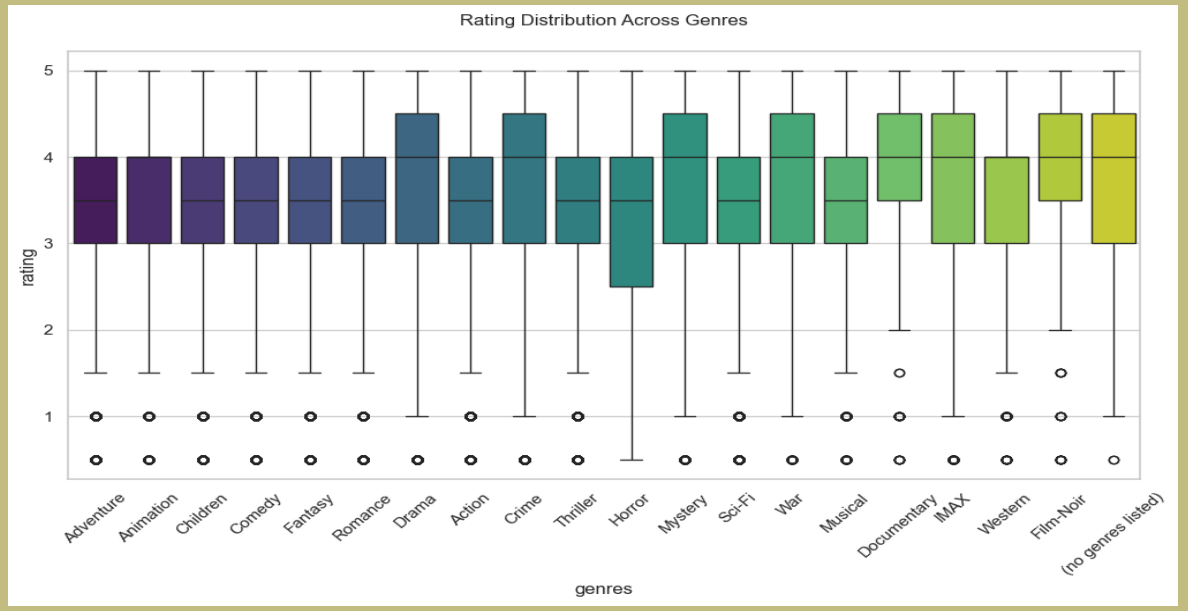
Exploratory Data Analysis

Rating Distribution



- The movie rating with 4.0 had the highest frequency

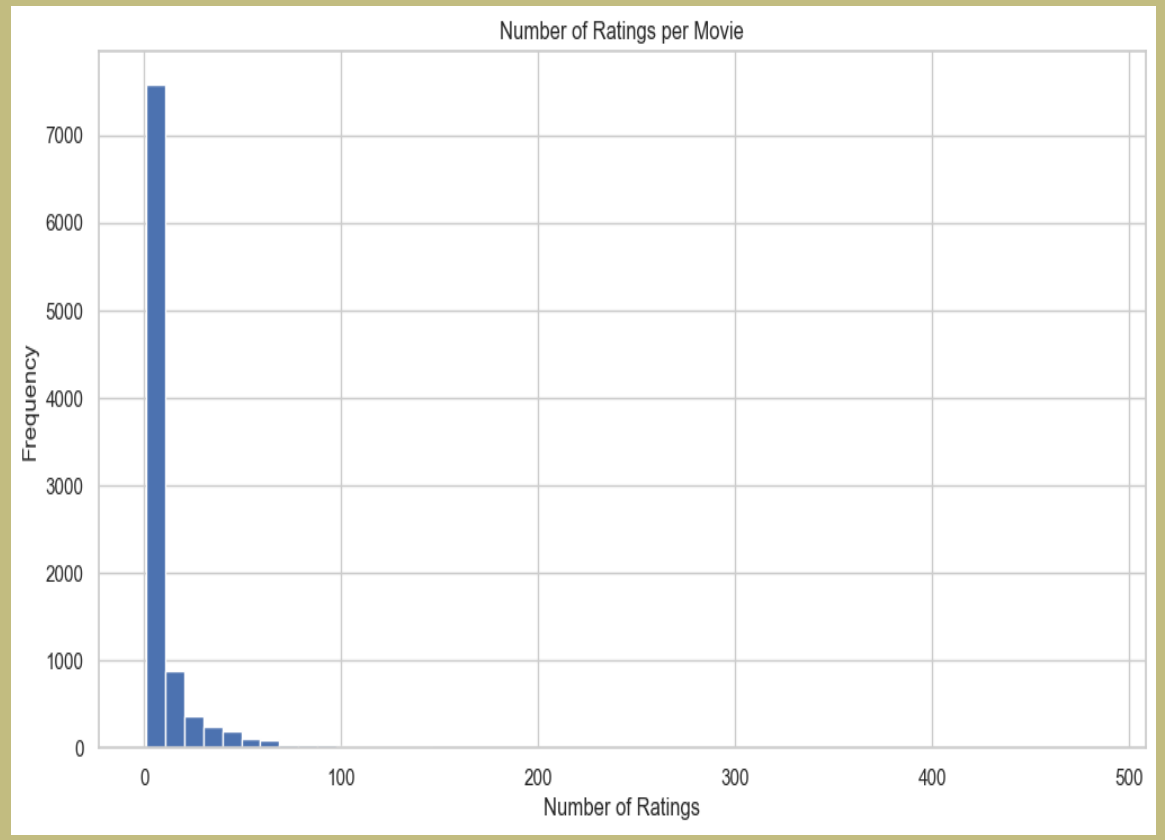
Rating Distribution by Genre



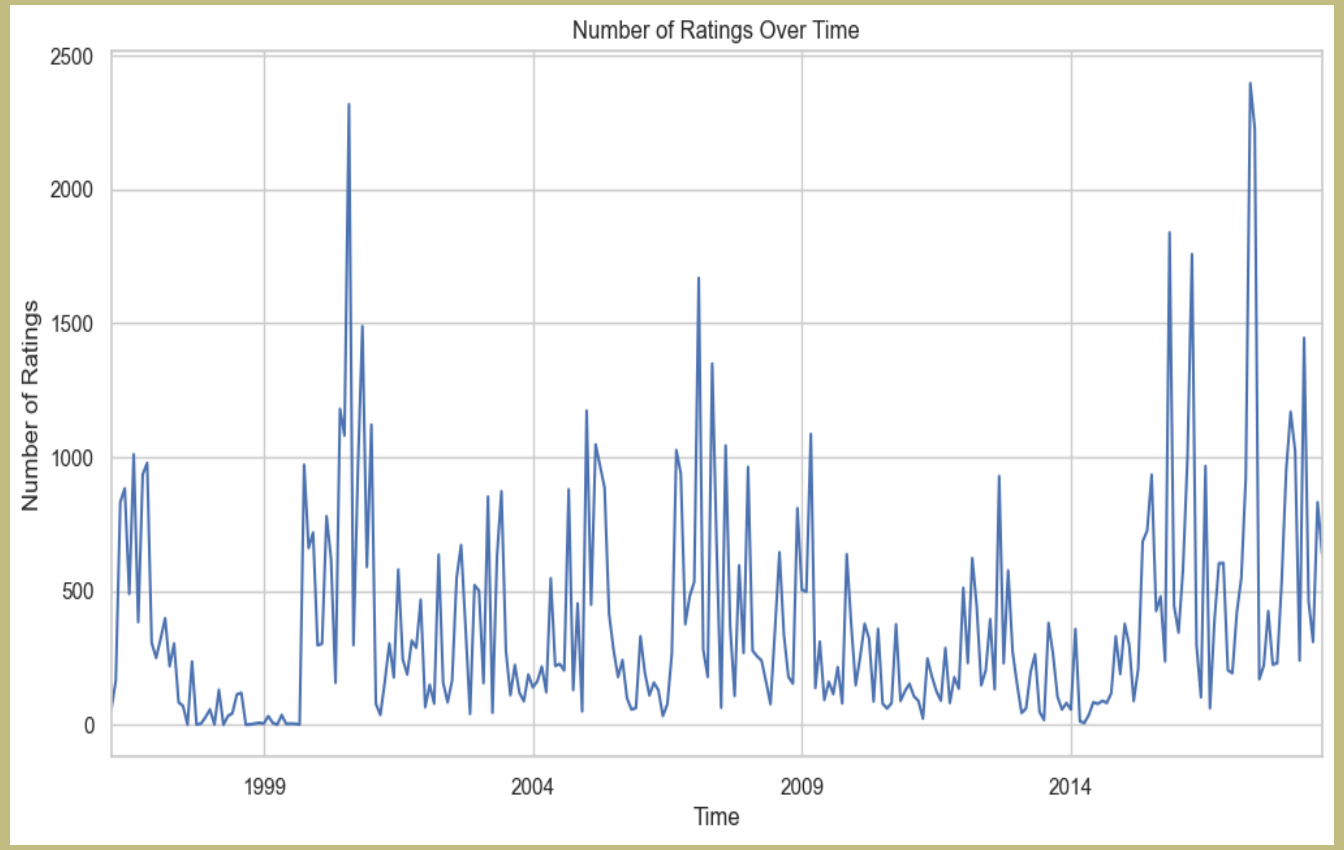
- The ratings vary across genres, Film-Noir, Western, and Documentary had higher ratings while genres like Horror and Crime show lower ratings.

Exploratory Data Analysis

Number of ratings per movie

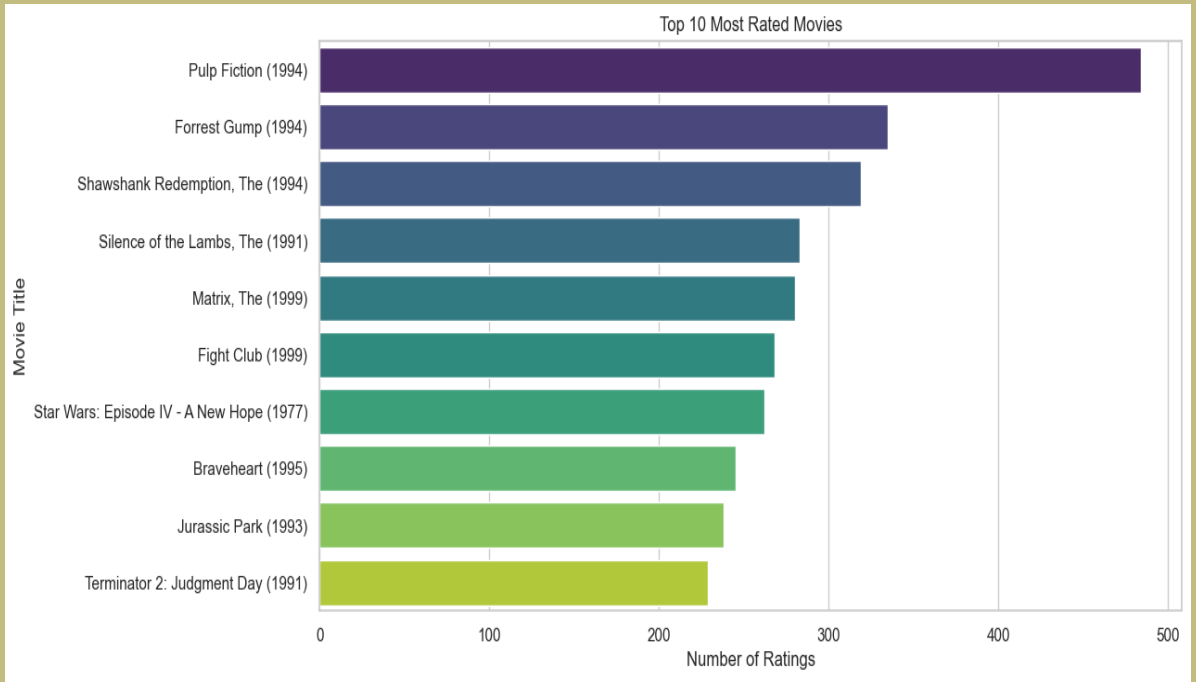


Number of ratings over time

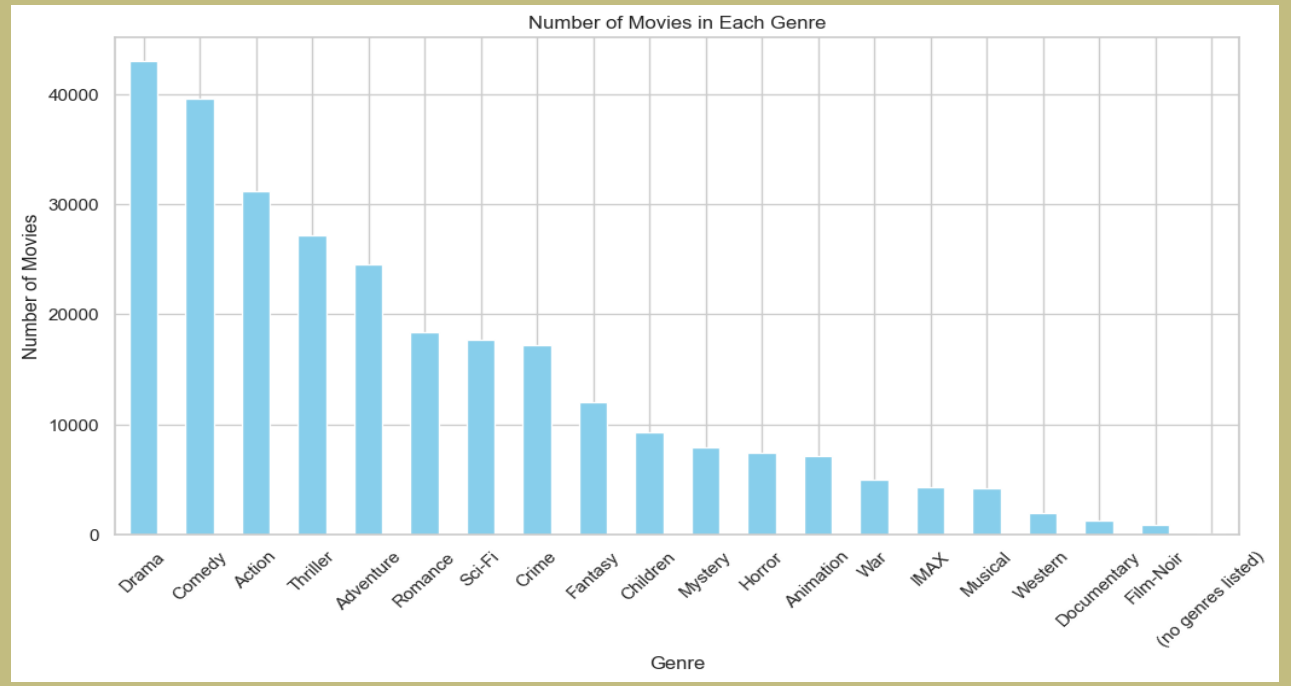


Exploratory Data Analysis

Top 10 most rated movies.



Movies counts in each genre



Modeling

In this project, we built and evaluated multiple recommender system models using the Movie Lens dataset and explored different approaches for providing personalized movie recommendations.

The models implemented included:

1. Content-Based Filtering
   - TF-IDF (Term Frequency-Inverse Document Frequency)

2. Collaborative Filtering
   - Memory-Based (Neighborhood) Approach
   - Model-Based (Matrix Factorization) Approach
   - Singular Value Decomposition (SVD)

Modeling

1. Content-Based Filtering

- TF-IDF (Term Frequency-Inverse Document Frequency)

Observations

- Since we’re not using ratings or tags, the model reflects only movie content similarity, not popularity or user preferences.
- It only recommends by genre, which is shallow and insufficient, as it doesn't scoop out the patterns deeply embedded in our dataset.
- When using TF-IDF on movie genres, the recommendation system can become rigid, meaning that if a movie is categorized under the genres Action, Adventure, and Sci-Fi, the system will primarily recommend movies that share the exact same combination of genres. This limits flexibility, as it doesn't allow for recommendations based on similar themes or overlapping genres.

Modelling

2. Collaborative Filtering

- Memory-Based (Neighborhood) Approach

The KNNBaseline model achieved a lower RMSE of approximately 0.872, indicating that its predictions were closer to the actual ratings compared to the average model, which had an RMSE of about 0.966.

- Model-Based (Matrix Factorization) Approach

  Singular Value Decomposition (SVD)

0.8883 RMSE suggests that the SVD model is performing reasonably well, with predictions that are fairly close to the actual ratings given by users. In recommender systems, RMSE values typically range from 0.5 to 1.5 in many cases, therefore, an RMSE of 0.8883 can be considered good

Precision@10 = 0.0556

The SVD model recommends, on average, 0.55 relevant items out of 10, which is relatively low.

Conclusions and Recommendations

Conclusions

In developing our movie recommender system, we evaluated several recommendation strategies, including content-based filtering, neighborhood-based collaborative filtering (KNN), and model-based collaborative filtering (SVD). Each approach offered distinct advantages and came with its own set of limitations.

1. Content-Based Filtering

We implemented a content-based recommender using features such as movie genres. This approach successfully recommended movies with similar characteristics, making it useful for users with well-defined preferences. However, it struggled to capture the broader and more diverse interests that users may have beyond surface level features.

2. Neighborhood-Based Collaborative Filtering (KNN)

Our neighborhood-based model, built with the KNN algorithm from SciKit-Learn, leveraged user-item interaction data to identify patterns in user behavior. The item-based variant, using cosine similarity, was particularly effective at recommending movies based on similarity in user ratings. This method performed well in identifying similar movies and delivered consistent recommendations.

3. Model-Based Collaborative Filtering (SVD)

Using the Surprise library, we implemented Singular Value Decomposition (SVD) to uncover latent factors within the user-item matrix. While this model-based approach achieved a respectable RMSE of 0.8925, its performance can be sensitive to hyperparameter tuning and data sparsity.

Recommendations

- After evaluating the strengths and limitations of each approach, we propose implementing a hybrid recommendation model that combines the advantages of content-based filtering and collaborative filtering. This integrated strategy enables the system to harness the nuanced user-item interaction patterns identified by collaborative methods while also utilizing content attributes such as genres and metadata to enhance personalization and broaden recommendation diversity.

- To further improve the effectiveness of collaborative filtering techniques, especially matrix factorization methods like SVD, additional hyperparameter optimization and scalability testing on larger datasets are recommended.

- In summary, the optimal recommender system depends on factors such as user behavior, application goals, and data characteristics. A well-designed hybrid model, regularly tuned and updated, offers a flexible and high-performing solution for delivering relevant and engaging movie recommendations.

Thank you