Netflix-Style Movie Recommendation System
Final Project Report

Liam Campbell - DS5110

December 8, 2025

Contents

1 Introduction

the goal of this project was to design and implement a Netflix-style movie recommendation system using the MovieLens 100k Dataset. The project involved building a complete end-to-end pipeline that includes data preprocessing, database design, feature engineering, model creation, evaluation, and finally deploying an interactive web-based application using Flask.

The primary objectives for this project were:

- Build a functional recommendation engine using real-world movie rating data.
- Implement multiple recommendation approaches: Collaborative Filtering, Content-Based Filtering, and Hybrid Modeling which combined both the Collaborative Filtering and Content-Based filtering.
- Store and query the dataset using a structured SQL schema.
- Provide analytical insights and outline future extensions.

This project demonstrates how modern recommendation systems combine user preferencemodeling, movie similarity signals, and hybrid scoring to generate a personalized suggestion for each user.

2 Problem Statement and Objectives

The core problem addressed in this project was to see how we can recommend movies to a user based on their rating history, movie similarity, and global viewing patterns. There was already a basic understanding of how these filtering models should work, but to take an extra step implementing it myself and if there were ways to add onto it.

Key objectives included:

- Process and clean the MovieLens dataset to make it usable for modeling
- Design and implement a SQL database for efficient retrieval of ratings and movie metadata
- Train collaborative filtering models capable of predicting user movie ratings
- Build content-based similarity models using TF-IDF and cosine similarity
- Combine both approaches (the collaborative filtering and the content-based filtering) into a hybrid recommendation strategy and the main model to showcase how recommending different movies work

- Build a user interface display recommendation in multiple categorizes:
  - Main recommendations (CF/Hybrid)
  - Trending Now
  - Classic Films
  - Because You Watched for movie-similarity based

3 Dataset Description

This project uses the **MovieLens 100k dataset**, publicly available at: `https://grouplens.org/datasets/movielens/100k/`

Dataset Structure

The dataset contains:

- 100,000 user ratings
- 943 users
- 1,682 movies
- Movie genres, titles, and release years
- User demographic information

This dataset was ideal for the recommendation systems due to the well detailed and documented dataset and provided information for metadata and well-structured rating information.

Why This Dataset

- Publicly available and widely used for benchmarking recommenders.
- Contains both numeric ratings and movie metadata needed for hybrid modeling.
- Provides enough users and movies to produce meaningful collaborative signals
- Small enough to allow experimentation and full model retraining.

4 Data Preprocessing and Feature Engineering

Several steps were required to prepare the dataset:

Database Schema

- **users(user_id)**
- **movies(movie_id, title release_year, genres**
- **ratings(user_id, movie_id, ratings, timestamp)**
- **recommendations(user_id, movie_id, score, algo_type)**

Cleaning and Parsing

- Release year extracted using regex.
- TF-IDF features generated from movies genres.
- Cosine similarity computed for content-based recommendation.
- Ratings normalized to ensure numerical stability.

5 Model Selection and Implementation

The recommendation engine integrates three core models.

Collaborative Filtering (SVD)

Using the Surprise library:

- SVD factorizes the user-item rating matrix.
- Predicts unknown ratings for each user-movie pair.
- Provides personalized recommendations based on behavior similarity.

Hybrid Model

The hybrid model combines:

$$\text{Hybrid Score} = \text{CF Prediction} + 0.1 \times \text{Similarity Boost}$$

This approach improves personalization by reinforcing movies similar to those a user rated highly.

6 Evaluation and Results

The SVD model achieved an RMSE of approximately:

$$\text{RMSE} \approx 0.93$$

This is consistent with published benchmarks for the MovieLens 100k Dataset.

Observed System Behavior

- CF alone recommends high-quality but diverse movies.
- Content-based filtering ensures genre and theme consistency
- Hybrid modeling increases personalization and user relevance

The final web interface displays:

- Top CF/Hybrid recommendations
- Similar movie based on viewing history
- Trending movies (recently rated)
- Classic films (pre-1980 highest-rated)

7 Insights and Discussion

Key insights from the system:

- The hybrid model outperforms the two single-method recommenders.
- Even simple movie metadata (genre and release year) provides strong similarity signals
- User history significantly influences recommendation accuracy
- Trending and classic categories provide additional browsing value.

8 Future Work and Recommendation

Future improvement could include:

- Integrating real movie posters via TMDB API (for front-end work)
- Adding user-based and item-based KNN collaborative filtering
- Using deep learning embeddings for movie representation
- Deploying the app with authentication and user profiles.
- Supporting implicit feedback for views, clicks and watch time

9 Conclusion

This project successfully demonstrates a complete movie recommendation pipeline using the MovieLens dataset. By combining CF, content similarity, and hybrid modeling, the system produces meaningful and personalized recommendations. The Netflix-style front-end provides an intuitive and recognizable platform for interacting with the recommendations, illustrating the practical applications of recommender systems in modern streaming platforms.