Movie Recommendation System

Project Title

Movie Recommendation System - A Lightweight KNN-Based Movie Recommender Engine

Project Description / Objective

This project implements a machine learning-powered recommendation system that suggests movies based on genre similarity and popularity metrics. The system addresses critical limitations in traditional collaborative filtering by operating at the movie level rather than individual ratings, eliminating duplicate recommendations and enabling sub-5ms query times.

Problem Statement

Conventional recommender systems suffer from:

- Duplicate recommendations due to operating on raw rating data (9.7M rows per movie)
- Slow query times and high memory consumption
- Complex, interpretable-unfriendly algorithms
- Cold-start problems for new content

Solution

A lightweight K-Nearest Neighbors (KNN) approach that:

- Aggregates 9.7M ratings into 9,742 unique movie profiles
- Uses interpretable genre vectors combined with popularity metrics
- Achieves sub-5ms query response times with minimal computational overhead
- Delivers diverse, non-duplicate recommendations consistently
- Maintains a compact model footprint of approximately 50KB

Tools and Technologies

- **Programming Language:** Python 3.11
- **ML Framework:** scikit-learn for KNN algorithm and preprocessing utilities
- **Data Processing:** pandas for data manipulation; numpy for numerical computations
- **Web Framework:** Streamlit for interactive user interface and real-time updates
- **Model Serialization:** joblib for efficient model persistence
- **Deployment:** Streamlit Cloud for serverless hosting and auto-scaling
- **Dataset:** MovieLens Small Dataset (9,742 movies, 100,836 ratings from 1995-2018)

Project Features

The system provides intelligent movie recommendations by analyzing genre similarity and popularity metrics:

- Real-time recommendation generation with sub-5ms query response times using optimized KNN
- Genre-based matching that identifies movies with similar content profiles
- Popularity-weighted recommendations that balance genre similarity with user ratings
- Zero duplicate results through explicit filtering and movie-level aggregation
- Lightweight model architecture with a 50KB footprint
- Web-based interface deployed on Streamlit Cloud for scalable access
- Complete documentation and reproducible analysis in Jupyter notebook format

Technical Approach

The system uses K-Nearest Neighbors (KNN) with cosine similarity to find similar movies:

- 20 binary genre vectors derived from multi-label genre data
- Normalized popularity scores based on rating counts
- 21-dimensional feature space enabling efficient similarity computations
- K=6 parameter returning 5 recommendations plus the query movie
- O(n) query complexity with approximately 5ms response time per request

Source Code and Documentation

Repository: github.com/Pujan-Dev/movie-recommendation

Live Application: pujan-dev-movie-recommendation-main-ib7ave.streamlit.app

Project Files:

- main.py: Streamlit web application entry point
- main.ipynb: Jupyter notebook containing exploratory analysis and model training
- movies_data.csv: Movie metadata with aggregated ratings
- movies_features.pkl: Pre-computed feature matrix (9,742 x 21)

- knn_movie_recommender.pkl: Trained KNN model
- requirements.txt: Python dependencies
- README.md: Complete project documentation

Dataset

- **Source:** GroupLens Research - MovieLens Small Dataset
- **Size:** 100,836 ratings from 610 users for 9,742 movies
- **Time Span:** 1995 to 2018
- **Rating Scale:** 0.5 to 5.0 stars
- **Genres:** 20 distinct categories
- **Aggregation:** Raw data consolidated from per-rating entries into movie-level statistics with average ratings and popularity scores

Performance Results

- **Query Time:** Approximately 5 milliseconds per request
- **Model Size:** Under 50 kilobytes
- **Query Complexity:** Linear with respect to number of movies
- **Duplicate Rate:** Zero - all recommendations are unique
- **Coverage:** All 9,742 movies in the dataset

Skills Demonstrated

Machine Learning: K-Nearest Neighbors algorithms, cosine similarity metrics, feature engineering

Data Science: Data aggregation, statistical analysis, performance optimization

Software Engineering: Object-oriented design, error handling, version control with Git

Web Development: Streamlit framework, cloud deployment on Streamlit Cloud infrastructure

Usage Instructions

- **Online:** Visit pujan-dev-movie-recommendation-main-ib7ave.streamlit.app
- **Local:** Clone the repository, install dependencies from requirements.txt, execute `streamlit run main.py`
- **API:** Load pre-trained models using joblib and call KNN search directly on movies_features
- **Input:** Any movie title from the dataset
- **Output:** Five recommendations ranked by feature space proximity

References

- K-Nearest Neighbors - scikit-learn Documentation
- Streamlit Framework
- pandas Documentation
- MovieLens Dataset