Phase-2 Submission

Delivering personalized movie recommendations with an AI-driven matchmaking system

Student Name: HariHaran S

Register Number: 513523104024

Institution: Annai Mira College of Engineering and Technology

Department: Computer Science and Engineering

Date of Submission: 02/05/2025

Github Repository Link: https://github.com/HariHaran011205/Movie-Recommendations.git

---

1. Problem Statement

This project addresses the challenge of efficiently recommending personalized movie content to users by developing an AI-powered matchmaking system. With an explosion of content on streaming platforms, users struggle to discover relevant movies matching their tastes.

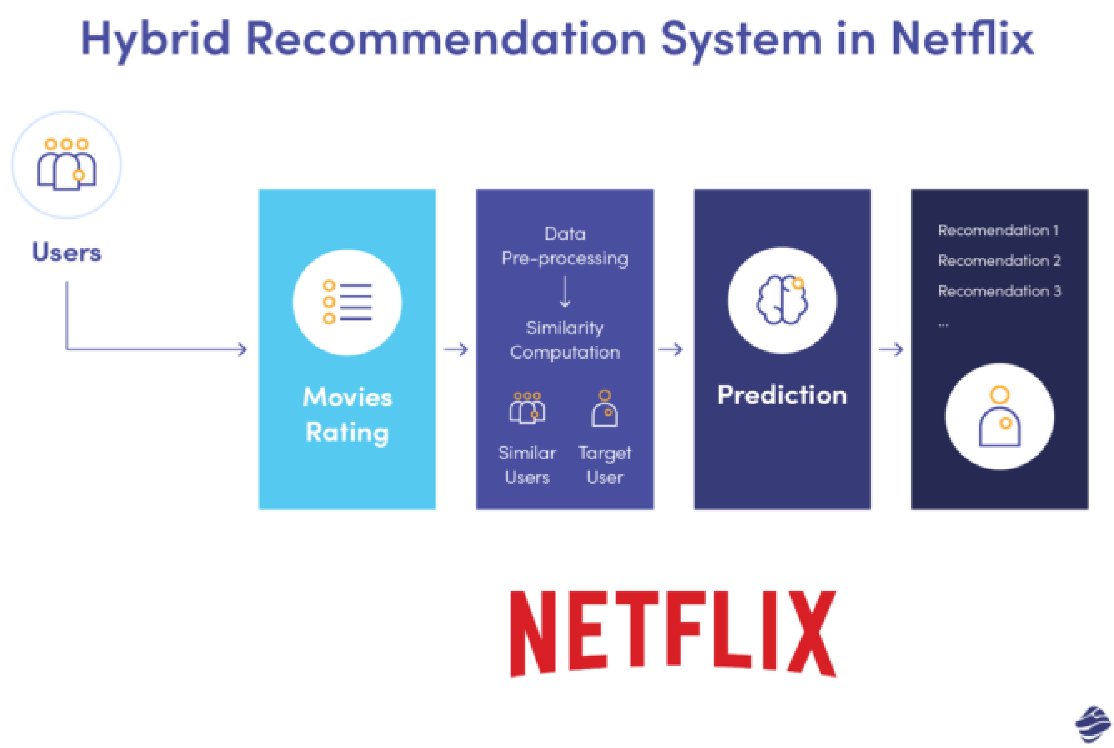
Type of Problem: This is primarily a **recommendation system**, often solved using a mix of **clustering, ranking**, and **regression/classification** techniques.

Why It Matters: Improved recommendations enhance user experience, boost user retention, and reduce content churn for platforms like Netflix, Amazon Prime, and YouTube.

2. Project Objectives

- Develop a hybrid recommendation model combining content-based and collaborative filtering.
- Personalize user experiences by analyzing user behavior and movie metadata.
- Improve the relevance and diversity of recommendations.
- Evolve model goals after data analysis to prioritize explainability and performance on sparse data.

3. Flowchart of the Project Workflow



4. Data Description

- **Source**: [e.g., MovieLens dataset from Kaggle or GroupLens]
- **Type**: Structured data
- **Records**: ~100,000 user ratings across ~10,000 movies

- **Features**: Movie metadata (genres, title, year), user ratings, timestamps
- **Target Variable**: User ratings or recommendation score
- **Nature**: Static dataset

5. Data Preprocessing

- Removed null values from metadata and user ratings
- Dropped duplicate entries based on userId-movieId pairs.
- Converted timestamps to readable datetime.
- Encoded genres using multi-hot encoding.
- Normalized rating values between 0 and 1 for model consistency.

6. Exploratory Data Analysis (EDA)

Univariate:

- Distribution of ratings (most ratings are 4 or 5).
- Popular genres (Drama, Comedy, Action).

Bivariate/Multivariate:

- Correlation of user preferences by genre.
- Popularity vs. average rating plots.

Insights:

- Older movies tend to have higher ratings.
- Users exhibit genre-specific preferences that can be clustered.

7. Feature Engineering

- Extracted release year from movie titles.
- Created “rating frequency” and “average rating” features.

- Constructed user profiles by aggregating rated genres.
- Applied TF-IDF vectorization for movie overviews (if text data available).
- Optional PCA on user-item matrix to reduce sparsity.

8. Model Building

Models Implemented:

- **Content-Based Filtering** using cosine similarity on TF-IDF features.
- **Collaborative Filtering** via Matrix Factorization (SVD).
- Compared with **KNN-based recommender** for baseline performance.

Evaluation Metrics:

- RMSE for rating prediction
- Precision@K, Recall@K for top-N recommendations

Data Split:

- 80-20 train-test split using stratification on userId.

9. Visualization of Results & Model Insights

- **RMSE Plot**: Shows convergence of matrix factorization model.
- **Top-N Hit Rate** bar chart per model.
- **Feature Importance** plot for content-based filtering (genre and overview).
- **User Clusters** visualized using t-SNE for latent embeddings.

10. Tools and Technologies Used

- **Programming**: Python

- **IDE**: Jupyter Notebook / Google Colab
- **Libraries**: pandas, numpy, scikit-learn, seaborn, matplotlib, surprise, TensorFlow (if deep model used)
- **Visualization**: seaborn, matplotlib, Plotly

11. Team Members and Contributions

Name	Contributions
Hameetha	Data cleaning, EDA
HariHaran	Feature engineering, documentation
Harishini	Model tuning, evaluation visualization
Harish kumar	Model development,programming