Phase-1

Student Name: Saravanan S

Register Number: 410723104078

Institution: Dhanalakshmi College of Engineering

Department: computer science engineering

Date of Submission: [30/4/2025]

---

1. Problem Statement

Despite the abundance of streaming platforms and available content, users often struggle to find movies that truly resonate with their personal tastes, moods, or viewing intentions. Traditional recommendation engines primarily rely on general viewing history or trending content, resulting in ***generic, one-size-fits-all suggestions*** *that lack emotional or contextual relevance. This leads to decision fatigue, disengagement, and a less satisfying entertainment experience.*
There is a need for a ***more intelligent, emotionally aware, and personalized recommendation system****—one that understands not just what a user has watched, but* ***why*** *they enjoyed it, how they feel at the moment, and what they're truly in the mood to experience.*

2.Objectives of the Project

- *Develop a personalized movie recommendation system that accurately suggests movies based on user preferences and behaviors.*
- *Implement and compare multiple recommendation algorithms including collaborative filtering, content-based filtering, and hybrid models.*

- *Generate user and item embeddings to better understand relationships between users and movies.*
- *Provide actionable insights into user behavior through exploratory data analysis.*
- *Deploy the model through an interactive interface for real-time movie recommendations.*

3.Scope of the Project

Inclusions:

- *Analysis of user-movie interaction data.*
- *Implementation of multiple recommendation algorithms.*
- *Visualization of user behavior and recommendation results.*
- *Deployment of the recommendation system via a simple web interface.*

Exclusions / Constraints:

- *Recommendations will be limited to the data available in the chosen dataset (e.g., no live user input from production users).*
- *The system will not include advanced NLP-based sentiment analysis of reviews unless integrated with external sources.*
- *Deployment will be on a lightweight platform suitable for demonstration (e.g., Streamlit or Flask-based web app).*

4.Data Sources

Primary Dataset:

Source: MovieLens Dataset

Type: Public, static dataset

Provider: GroupLens Research

Description: *Contains user ratings, timestamps, and metadata for thousands of movies. Includes datasets of various sizes (e.g., 100k, 1M, 10M ratings) suitable for different project scales.*

Optional Additional Sources:

- ***TMDb (The Movie Database) API***

  Website: *https://www.themoviedb.org/*

  API Documentation: *https://developer.themoviedb.org/reference/intro*

  Usage: *Use for retrieving additional metadata such as movie posters, overviews, cast/crew info, etc. Requires a free API key.*

- ***IMDb Dataset***

  Website: *https://www.imdb.com/interfaces/*

  Description: *Publicly available datasets provided by IMDb for non-commercial use. Includes metadata like cast, crew, genres, and movie descriptions. Note that IMDb data access may be subject to licensing limitations.*

5.High-Level Methodology

Data Collection

Primary Source: The MovieLens dataset will be downloaded directly from the GroupLens website.

Optional: Movie metadata such as posters, cast, and summaries may be retrieved from the TMDb API using API keys.

Data Cleaning

Tasks:

Handle missing values (e.g., fill, remove, or impute where necessary).

Remove duplicate entries in user ratings or movie data.

Normalize and standardize movie genre labels and timestamps.

Tools: Python with pandas and numpy.

Exploratory Data Analysis (EDA)
Goals:

Identify rating distribution, most popular movies, genre trends.

Analyze user behavior (e.g., most active users, average ratings).

Visualizations:

Histograms, bar plots, heatmaps, scatter plots.

Libraries: matplotlib, seaborn, plotly.

Feature Engineering

Tasks:

Convert genre strings into one-hot encoded features.

Create user profiles based on average genre preferences.

Develop item-based embeddings using collaborative data.

Outcome: Enhance input features for better model performance.

Model Building

Approaches:

Collaborative Filtering (e.g., Matrix Factorization, SVD)

Content-Based Filtering (using metadata features)

Hybrid Models (combining both approaches)

Tools: scikit-learn, Surprise, LightFM, TensorFlow (optional for deep learning-based recsys)

Model Evaluation

Metrics:

RMSE, MAE for rating predictions.

Precision@K, Recall@K, F1-Score for top-N recommendations.

Validation Strategy:

K-Fold Cross-Validation

Train/Test split (80/20 or 70/30)

Visualization & Interpretation Techniques:

Top-N recommendation lists for sample users.

Confusion matrices and performance charts.

Interactive dashboards (if applicable).

Deployment
Platform: Streamlit (preferred for simplicity) or Flask for custom web apps.

Goal: Allow users to input preferences or IDs and get real-time recommendations.

Hosting: Streamlit Cloud or Heroku.

6.Tools and Technologies

Programming Language
Python

Notebook / IDE
Jupyter Notebook for analysis and model development

VS Code for modular scripting and deployment preparation

Libraries
Data Processing: pandas, numpy

Visualization: matplotlib, seaborn, plotly

Modeling: scikit-learn, Surprise, LightFM

Web/API: requests (for TMDb API)

Optional ML: TensorFlow/Keras (for neural collaborative filtering)

Optional Tools for Deployment
Streamlit – For a simple and interactive web interface

Flask – For custom APIs or full-stack deployment

Heroku / Render – For hosting the deployed application

7.Team Members and Roles

- ***Team Leader***- *Ramakrishnan RD*
  - *Data collection*
  - *Coding*
- ***Team Member***- *Sam Stevenson J*
  - *Report Writing and Documentation*
  - *Model training and evaluation*
- ***Team Member***- *saravanan S*
  - *Data collection*
  - *Data cleaning and preprocessing*
- ***Team Member***- *sanjay R*
  - *EDA and Visualization*
  - *Data collection*