**Delivering personalized movie recommendation with an AI driven matchmaking system**

**Phase-2 Submission**

**Student Name:** Thowfiq.I

**Register Number:** 422623104043

**Institution:** University College of Engineering, Panruti

**Department:** Computer Science and Engineering

**Date of Submission:** 09.05.2025

**Github Repository Link**: <https://github.com/THOWFIQ28/AI_DRIVEN-MOVIE-RECOMMENDATION-SYSTEM.git>

# 1. Problem Statement

Today's digital streaming age presents the user with too many choices when choosing something to watch. Conventional recommendation systems either provide too much of an obvious suggestion or do not possess a social discovery component.

This project aims to develop a hybrid user matchmaking and recommendation system by integrating clustering and recommendation methods. It suggests movies based on individual tastes as well as social similarity with other users.

Problem Type: Clustering + Recommendation (Hybrid System)

Impact: Improves personalized content experience and social discovery

# 2. Project Objectives

* Develop a hybrid movie recommendation system incorporating content-based filtering, collaborative filtering, and deep learning
* Create dynamic user profiles that collect preferences, view behavior, and emotional responses
* Create an AI-based matchmaking function linking users by similar preferences
* Construct a live feedback loop in order to make adjustments and personalize recommendations
* Offer an entertaining and socially enhanced motion picture discovery web site

# 3. Flowchart of the Project Workflow

┬────────────────────────────────────────────────────┐

│ Begin: Project Kickoff │

└────────────────────────────────────────────────────┘

↓

┌────────────────────────────────────────────────────┐

│ Data Collection │

│ (MovieLens Dataset + Synthetic User Profiles) │

└────────────────────────────────────────────────────┘

↓

┌────────────────────────────────────────────────────┐

│ Data Cleaning and Preprocessing │

│ (Missing Value Handling, Encoding, Normalization) │

└────────────────────────────────────────────────────┘

↓

┌────────────────────────────────────────────────────┐

│ Exploratory Data Analysis (EDA) and Feature Engineering │

│ (User Preferences, Movie Vectors, Similarity Matrices) │

└────────────────────────────────────────────────────┘

↓

┌────────────────────────────────────────────────────┐

│ Model Building │

│ - Content-Based Filtering │

│ - Collaborative Filtering (Matrix Factorization) │

│ - Models de Deep Learning │

└────────────────────────────────────────────────────┘

↓

┌────────────────────────────────────────────────────┐

│ User Matchmaking │

│ (User Clustering using K-Means / Cosine Similarity) │

└────────────────────────────────────────────────────┘

↓

┌────────────────────────────────────────────────────┐

│ Model Evaluation │

│ (RMSE, Precision@k, Recall@k, Silhouette Score) │

└────────────────────────────────────────────────────┘

↓

┌────────────────────────────────────────────────────┐

│ Deployment │

│ (Command-Line Interface Application) │

└────────────────────────────────────────────────────┘

↓

┌────────────────────────────────────────────────────┐

│ End │

└────────────────────────────────────────────────────┘

# 4. Data Description

**Dataset Name and Source:**

* Movies: Kaggle - MovieLens Dataset
* Users: Synthetic User Data (Self-Generated)

Type of Data: Structured (Tabular)

**Number of Records and Features:**

* Movies: ~10,000 movies
* Users: ~500 synthetic users

Dataset Nature: Static (locally used)

**Target Variables:**

* For Recommendation: Predict movie ratings
* For Matchmaking: Create user clusters (unsupervised)

# 5. Data Preprocessing

* Treated missing values by replacing missing directors/genres with "Unknown"
* Removed duplicate movie records
* Outliers were not significant; ratings were already bounded between 0 and 5
* Standardized genre names in lowercase formatting
* Encoded genres using multi-hot encoding
* Used Min-Max scaling on numerical features where required

# 6. Exploratory Data Analysis (EDA)

Univariate Analysis:

* Plotted histograms for ratings distribution
* Bar plots for most watched genres Bivariate/Multivariate Analysis:
* Heatmaps to display user similarity
* Scatterplots indicating the correlation between watch time and rating Insights Summary:
* Action and Drama were the most watched genres
* Movies with higher ratings tended to be watched for longer periods

# 7. Feature Engineering

* Generated user preference vectors based on genre affinity, watch time, and emotional feedback
* Generated movie feature vectors based on genre, cast, and keywords
* Constructed user similarity matrices using cosine similarity
* Applied K-Means clustering to segment users into comparable groups

Justification:

These engineered features directly improve the recommendation and matchmaking performance

# 8. Model Building

**Models Implemented:**

1. Content-Based Filtering: Recommends movies based on highly-rated ones with similar content
2. Collaborative Filtering (Matrix Factorization): Recommends based on user-user and item-item similarity
3. Deep Neural Networks: Models complex non-linear user behavior patterns

Data Split: 80% training and 20% testing **Evaluation Metrics:**

* RMSE for rating prediction accuracy
* Precision@k, Recall@k for recommendation quality
* Silhouette Score for clustering quality

**9. Visualization of Results and Model Insights Recommendation System Evaluation:**

* RMSE plots comparing models
* Precision and Recall plots **Matchmaking Evaluation:**
* Silhouette scores plotted
* 2D PCA or t-SNE projections of grouped users **Insights:**
* Collaborative Filtering worked better for sparse users

•.Deep Learning models improved recommendations by a lot for new (cold start) users

# 10. Tools and Technologies Used

Programming Language: Python

IDE/Notebook: Google Colab, Jupyter Notebook, VS Code

Libraries:

* Data Processing: pandas, numpy
* Visualization: matplotlib, seaborn, plotly
* Machine Learning: scikit-learn, Surprise, TensorFlow/Keras
* Natural Language Processing (optional): nltk, spaCy

Deployment Platform: Command-Line Interface (CLI) (Future enhancement to Streamlit or Flask in pipeline)

# 11. Team Members and Contributions

**Team**

|  |  |  |
| --- | --- | --- |
| **Member** | **Role** | **Responsibilities** |
| **Thowfiq I** | System Integration Engineer & Architect | * Design system architecture and data pipelines * Integrate modules for seamless operation |
| **Jeevitha C** | UI/UX Designer & Tester | * Design command-line interface (CLI) interactions * Test system features and gather usability feedback |
| **Balaji p** | Data Analyst & Preprocessing Lead | * Handle data cleaning, feature engineering, and data preparation * Ensure quality and consistency of datasets |
| **Anitha L** | Project Manager &  Documentation Specialist | - Coordinate project timelines and deliverables - Document methodologies, results, and  maintain team communication |
| **Anitha A** | Machine Learning Specialist | - Build and optimize AI-driven recommendation models |

- Develop matchmaking algorithms using user data