Phase-2 Submission

Student Name: Monish M

Register Number: 410723104047

Institution: Dhanalakshmi college of Engineering

Department: B.E(C.S.E)

Date of Submission: 08-05-2025

GitHub Repository Link: Bharath-m10/NM_Bharath

Delivering personalized movie recommendations with an AI-driven matchmaking system

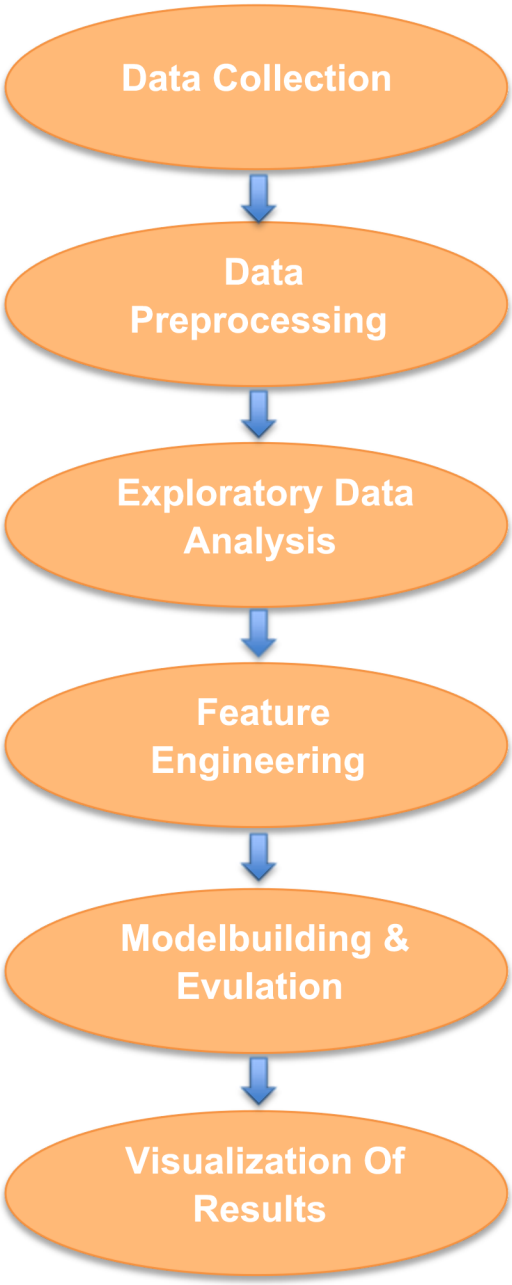
1. Problem Statement

- In the current digital entertainment landscape, users are overwhelmed by the sheer volume of content available across streaming platforms. The problem is to help users discover movies they are likely to enjoy by leveraging historical viewing patterns and user preferences. The aim is to develop an AI-driven personalized movie recommendation system that accurately matches users with relevant content.
- This is primarily a ranking and recommendation problem (unsupervised + supervised components), often addressed through collaborative filtering, content-based filtering, or hybrid models.
- Delivering relevant recommendations enhances user experience, increases engagement time, and drives platform loyalty—making this system crucial
- in today's media ecosystem.

2. Project Objectives

- To develop an AI-based system that provides personalized movie recommendations to users.
- To implement and compare collaborative filtering and content-based approaches.
- To optimize for accuracy, diversity, and novelty in recommendations.
- To ensure the system is scalable and adaptable to real-world datasets.

3. Flowchart of the Project Workflow



4. Data Description

- Dataset Source: Movie Recommendation System from Kaggle
- Type of Data: Structured
- Features: User ID, Movie ID, Ratings, Timestamps, Movie Genres
- Records: ~100,000 ratings across 943 users and 1,682 movies
- Dynamic/Static: Static snapshot
- Target Variable: Predicted rating or ranked list of recommended movies
- Dataset Link: Movie Recommendation System

5. Data Preprocessing

- Removed duplicate entries
- Converted timestamp to datetime format
- Encoded genres using multi-hot encoding
- Handled missing values (none in this dataset)
- Merged ratings and movie metadata
- Standardized rating scale where needed
- Normalized features for similarity calculations

```
print(movies.duplicated().sum())
print(ratings.duplicated().sum())
```

6. Exploratory Data Analysis (EDA)

Univariate Analysis:

- Distribution of movie ratings

- Count of ratings per user and per movie

Bivariate/Multivariate Analysis:

- Correlation between average ratings and number of ratings
- Popular genres vs. average user ratings

Insights:

- Long-tail distribution observed in movie popularity
- Certain users rate significantly more than others
- Popular genres receive more consistent ratings

7. Feature Engineering

- Created a user-movie interaction matrix
- Engineered user profile vectors based on genres
- Extracted genre similarity features
- Computed cosine similarity between movies for content-based filtering

```
from sklearn.preprocessing import LabelEncoder
# Encoding movie titles
le = LabelEncoder()
data['title'] = le.fit_transform(data['title'])
```

8. Model Building

Models Implemented:

- Collaborative Filtering using Matrix Factorization (SVD)
- Content-Based Filtering using Cosine Similarity
- Hybrid approach combining both strategies

Justification:

- SVD captures latent preferences
- Content-based ensures cold-start items get visibility

Train-Test Split:

- 80% training, 20% testing
- Used train_test_split with random_state for reproducibility

```
from sklearn.model_selection import train_test_split
# Features and Target
X = data.drop(columns=['userId', 'movieId', 'timestamp', 'rating']) # Drop
y = data['rating'] # Target variable
# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

9. Visualization of Results & Model Insights

- RMSE Comparison Chart between models
- Precision/Recall at K plots
- Heatmap of user-movie interactions
- Feature Importance from hybrid weighting
- Top-N Recommendations per user visualized in table format

10. Tools and Technologies Used

- Language: Python
- IDE: Jupyter Notebook
- Key Libraries:
  - pandas, numpy for data handling
  - matplotlib, seaborn, plotly for visualizations
  - scikit-learn for preprocessing and modeling

- Gradio for interface deployment

- Visualization: seaborn, matplotlib, Plotly

11. Team Members and Contributions

NAME	ROLE	RESPONSIBLE
Bharath M	Leader	Project Manager
Abinesh G	Member	Data Collection, Data Preparation
Monish M	Member	Data Preprocessing, Data Cleaning
Bharath Kumar L	Member	Data Visualization
Harish P	Member	Data Modeling