Phase-2

Student Name: JEEVITHA.M

Register Number: 510123106018

Institution: Adhiparasakthi College of Engineering

Department: Electronics and communication Engineering.

Date of Submission: 08-05-2025

Github Repository Link: https://github.com/Jeevitha-2005362/movie-matchmaking.git

---

DELIVERING PERSONALIZED MOVIE RECOMMENDATIONS WITH AN AI-DRIVEN MATCHMAKING SYSTEM

1. Problem Statement:

In an era of overwhelming content choices, users often struggle to find movies that truly match their unique tastes and emotional preferences. Traditional recommendation systems rely heavily on general popularity, genre classification, or surface-level user behavior, which frequently results in generic or irrelevant suggestions. This leads to user frustration, decision fatigue, and decreased engagement. There is a growing need for a more intelligent and emotionally aware system that can deliver deeply personalized movie recommendations by understanding users on a more individual and nuanced level.

2. ProjectObjectives

1. **Develop a user-centric profiling system:**

Create dynamic user profiles by collecting and analyzing data such as viewing history,genre preferences,emotional reactions,and user feedback.

2. **Build an AI-Driven matchmaking engine:**

Design and implement a machine learning model that matches users with movie based on deep features like themes, tone, pacing, and emotional resonance rather than just genres or ratings.

3. **Enhance Recommendation Accuracy Over Time:**

Integrate feedback loops and adaptive learning mechanisms to continuously refine and personalize recommendations as user behavior evolves.

4. **Incorporate Natural Language Processing (NLP):**

Analyze movie plots, reviews, and metadata using NLP techniques to extract meaningful insights and improve the contextual relevance of recommendations**.**

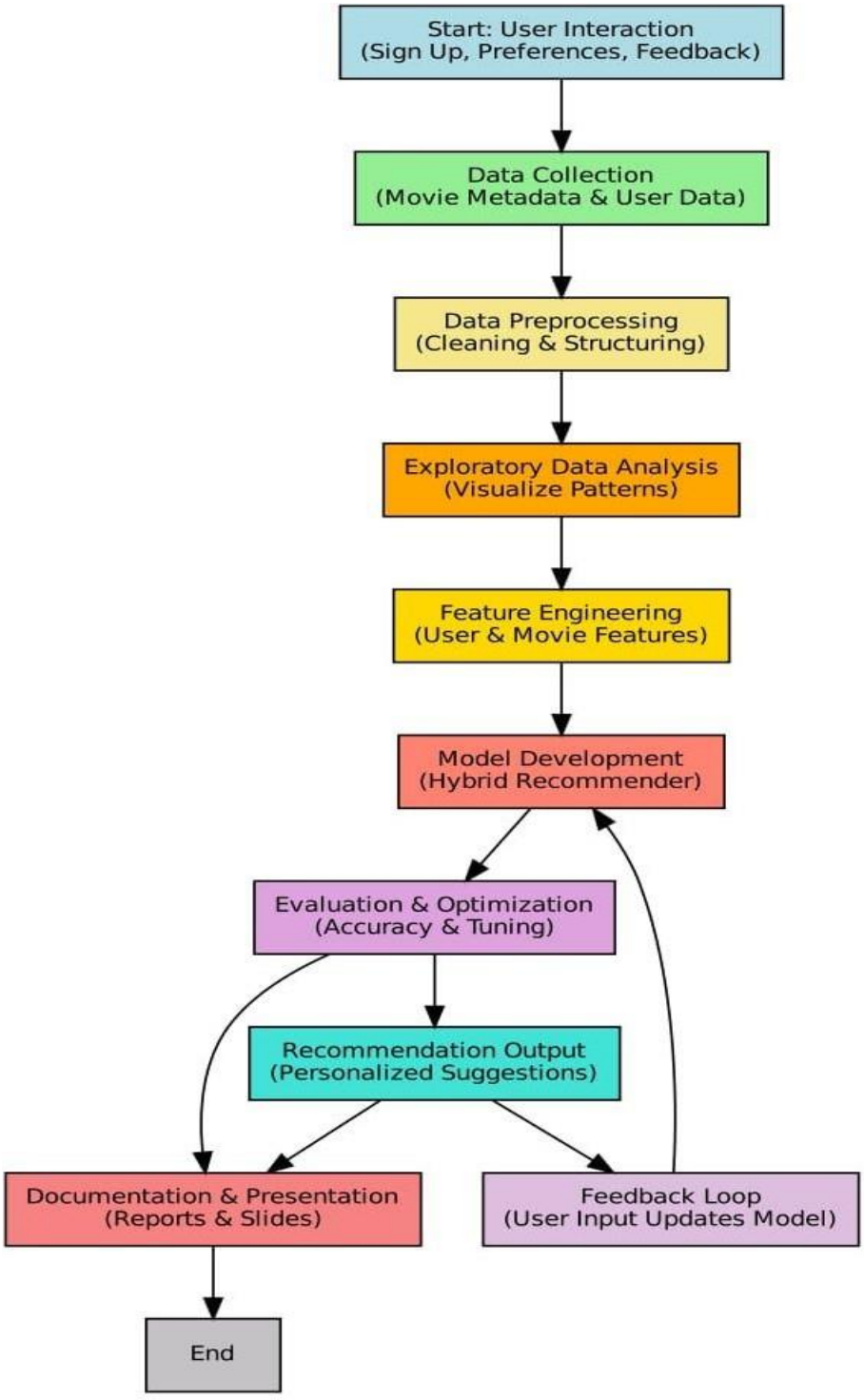
5. **Provide an Intuitive User Interface:**

Develop a user-friendly interface where users can easily receive, rate, and explore personalized movie suggestions.

6. **Foster Social Discovery (Optional):**

Enable social matching features to allow users with similar tastes to connect, share, or co-watch movie recommendations.

3. Flow chart of the Project Workflow



4. Data Description

To develop a robust and accurate AI-driven movie recommendation system, the project utilizes diverse datasets that capture both user preferences and movie characteristics. The datasets are categorized as follows:

1. Movie Metadata

Source: The Movie Database (TMDb), IMDb, Kaggle Datasets

Fields:

movie_id: Unique identifier

title: Movie name

genres: List of genres (e.g., Action, Drama)

overview: Short description or synopsis

release_date: Date of release

cast and crew: Main actors and directors

runtime: Duration of the movie

language: Original language

production_companies and countries

2. User Interaction Data

Source: Simulated or collected from public datasets (e.g., MovieLens)

Fields:

user_id: Unique user identifier

movie_id: Referencing the movie watched

rating: User rating (e.g., 1-5 or 0-10 scale)

timestamp: Time of rating

watch_history: Sequence of watched movies

feedback: Optional likes/dislikes or reviews

3. Reviews and Sentiments

Source: Scraped from IMDb or available datasets

Fields:

review_id

user_id

movie_id

review_text: User-written review

sentiment_score: Sentiment label or score (computed using NLP tools)

4. User Profile Data (Optional)

Fields:

age, gender, location

preferences: Favorite genres or actors

device_info: (for cross-platform behavior modeling)

5. Data Preprocessing

1. Data Cleaning
    - Handling missing values (e.g., fill with mean/median/mode or remove rows/columns).
    - Removing duplicates.
    - Fixing data entry errors.
2. **Data Integration**
    - Combining data from multiple sources into a single dataset.

3.Data Transformation

- Normalization/Standardization: Scaling features to a similar range.
- Encoding categorical variables: e.g., one-hot encoding or label encoding.
- Date/time formatting.

4. Data Reduction

- Reducing the number of features using techniques like PCA (Principal Component Analysis) or feature selection.

5. Data Discretization

- Converting continuous data into discrete bins (e.g., age into age groups).

6. Splitting Data

- Dividing into training, validation, and test sets.

6. Exploratory Data Analysis (EDA)

`

1. Understanding the Datasets

a. Movie Metadata

Fields: Title, Genre(s), Release Date, Cast, Crew, Runtime, Synopsis, Ratings

EDA Tasks:

Count and distribution of genres

Distribution of movie runtimes

Trends in movie production over years

Most frequent actors/directors

Correlation between genre and average rating

b. User Data

Fields: User ID, Age, Gender, Country, Preferences

EDA Tasks:

Demographics breakdown (age/gender distribution)

Popular genres by demographic

User activity patterns (watch frequency, time of day)

c. User-Movie Interaction Data

Fields: User ID, Movie ID, Rating, Watch Time, Like/Dislike, Click Events

EDA Tasks:

Rating distributions (overall, per user, per movie)

Top-rated vs. most-watched movies

Heatmap of interactions (time vs. activity)

Sparse matrix analysis (user-movie interactions)

2. Visualizations

Genre Popularity: Bar chart or sunburst chart

User Rating Heatmap: Correlation between users and genres

Movie Embeddings (UMAP/t-SNE): Cluster similar movies in 2D

User Segmentation: K-means or PCA clustering based on interaction patterns

3. Statistical Insights

Bias Detection: Are certain genres overrated/underrated by specific demographics?

Cold Start Analysis: % of users/movies with insufficient interaction

Time Series Trends: How preferences shift over months or years

7. Feature Engineering

Feature engineering plays a critical role in enhancing the accuracy of AI models used in the movie recommendation system. It involves transforming raw data into meaningful features that help the model learn user preferences and content similarities more effectively.

1. Movie-Based Features

- Genre Encoding: Multi-hot encoding of genres (e.g., Action, Drama, Sci-Fi) for similarity analysis.
- Text Vectorization:
- TF-IDF or Word2Vec on movie overview, title, and tagline for semantic understanding.
- BERT embeddings (optional) for deeper contextual analysis.
- Crew & Cast Mapping: One-hot or frequency-based encoding of top actors and directors.

Popularity & Ratings:

- Average user rating
- Number of ratings (used as a confidence factor)
- Movie popularity score from TMDB or IMDb

2. User-Based Features

- User Interaction History:
- Aggregated features like average rating given, genres most interacted with

  Count of movies rated or watched

- Temporal Behavior:
- Time of day/week patterns
- Recently viewed genres for trend capture

- User Embeddings:
- Derived from collaborative filtering or neural network models (e.g., Matrix Factorization, Autoencoders)

3. Interaction Features

- User-Movie Interaction Matrix: Sparse matrix showing which user interacted with which movie
- Rating Deviation: Difference from a user's average rating to detect likes/dislikes
- Review Sentiment Scores:
- NLP-derived polarity (positive/negative/neutral)
- Sentiment intensity as numerical input

4. Hybrid Features

- Content-Collaborative Hybrid:
- Combine similarity scores from both metadata and user patterns
- Weighted scoring based on user activity or trust score

- Time Decay Factors:
- Recent interactions weighted more heavily than older ones

- Contextual Features (optional):
- Day, time, device used – to understand user context

8. Model Building

The core of the movie recommendation system lies in building intelligent models that can learn from user data and content features to deliver personalized suggestions. This system integrates both content-based filtering and collaborative filtering, along with advanced AI models to form a hybrid recommendation engine.

1. Model Architecture Overview

- Hybrid Recommender System combining:
- Content-Based Filtering: Recommends movies similar to those a user liked in the past based on metadata.
- Collaborative Filtering: Recommends movies based on similar users' preferences.
- NLP-based Sentiment Model: Analyzes user reviews for emotional tone and context.

2. Content-Based Filtering

- Input: Movie metadata (genres, overview, cast, etc.)
- Method:
- TF-IDF or Word2Vec on text features
- Cosine similarity between movie vectors
- Output: List of movies similar to those previously liked by the user

3. Collaborative Filtering

- Matrix Factorization (e.g., SVD, NMF):
- Decomposes the user-movie interaction matrix
- Learns latent factors representing user and movie characteristics
- Deep Learning Alternatives:
- Autoencoders for learning user preferences in a dense form
- Neural Collaborative Filtering (NCF)

4. Sentiment Analysis Model

- Model: Logistic Regression or LSTM
- Input: Preprocessed user reviews

- Output: Sentiment score or label (positive/negative)
- Use: Boosts or filters recommendations based on review sentiment

5. Ensemble Strategy

- Combines outputs from:
- Similarity scores (content)
- Predicted ratings (collaborative)
- Sentiment-enhanced ranking
- Uses weighted average or ranking algorithms to finalize recommendations

6. Model Evaluation

- Metrics Used:
- RMSE / MAE for rating prediction accuracy
- Precision@K, Recall@K for top-K recommendation relevance
- F1-Score for sentiment model

Cross-validation used for generalizability

9. Visualization of Results & Model Insights

To evaluate the effectiveness and interpretability of the AI-driven recommendation system, visualizations were created to analyze model behavior, recommendation accuracy, and user trends. These insights help in refining the model and improving user engagement.

1. User Behavior Insights

- Genre Popularity Heatmap:
- Shows which genres are preferred by different age groups or user segments.

Example: Younger users tend to prefer Action and Sci-Fi, while older users lean towards Drama and History.

- Ratings Distribution Histogram:
- Displays the spread of user ratings.
- Helps identify user rating habits (e.g., tendency to rate movies either very high or low).

2. Recommendation Performance

- Precision@K and Recall@K Graphs:
- Plots showing how well the model predicts relevant movies in top-K suggestions.
- Insight: Hybrid models generally show higher precision and recall compared to standalone content-based or collaborative models.

- Confusion Matrix (for Sentiment Analysis Model):
- Visual evaluation of sentiment model accuracy (positive vs. negative reviews).
- Useful in identifying where the model may misclassify user intent.

3. Model Comparison Charts

- Bar Chart: RMSE / MAE Scores for different algorithms:
- SVD, Autoencoders, Neural Collaborative Filtering (NCF)
- Helps choose the optimal model for deployment.

- Similarity Matrix Heatmap:
- Visual representation of cosine similarities between movie vectors.
- Useful for debugging and verifying recommendations (e.g., shows how close “Inception” is to “Interstellar”).

4. Top Recommendations Display

- Recommendation Dashboard:
- UI mockups or tables showing top 5–10 personalized recommendations for a sample user.
- Includes title, genre, predicted rating, and sentiment score.

10. Tools and Technologies Used

The development of this personalized movie recommendation system involves a diverse set of tools and technologies across data handling, machine learning, natural language processing, and deployment. These tools were selected to ensure scalability, efficiency, and accuracy.

1. Programming Languages

- Python: Core development language for machine learning, data preprocessing, and backend services.

2. Libraries and Frameworks

- Data Handling & Analysis:
- Pandas, NumPy: For data manipulation and numerical computation
- Matplotlib, Seaborn: For visualizations and result plots

- Machine Learning & Recommendation Models:
- Scikit-learn: Traditional ML models and evaluation metrics
- Surprise: Specialized library for building collaborative filtering recommenders
- TensorFlow / Keras: Neural network models like Autoencoders or NCF
- XGBoost (optional): For boosting-based hybrid approaches

Natural Language Processing (NLP):

- NLTK, spaCy: Text preprocessing and tokenization
- TF-IDF, Word2Vec, BERT: For converting reviews and descriptions into vectors
- TextBlob, VADER: For sentiment analysis

3. **Databases**

- MongoDB or MySQL: For storing user data, movie metadata, and interaction history
- Firebase (optional): For real-time updates and authentication

4. **API & Web Integration**

- Flask or Django: Backend web framework to expose recommendation engine as an API
- ReactJS or Flutter: For building a responsive and dynamic user interface
- TMDB API or IMDb API: For fetching real-time movie metadata

5. Development & Collaboration Tools

- Jupyter Notebook: Experimentation and model prototyping
- GitHub: Version control and collaboration
- Google Colab: Cloud-based training and testing environment

6. Deployment & Hosting

- Heroku, Render, or AWS EC2: For hosting backend models
- Docker (optional): Containerization for portability
- Streamlit / Gradio (optional): For quick deployment of demo dashboards

11. Team Members and Contributions

1. V. JAMUNA DEVI- Data collection and integration: Gather movie related data from sources like IMDb or TMDB and collect user data .

2. S. MONIKA- Data cleaning and EDA : Prepare raw data by removing errors, duplicates, and missing values to ensure its clean and reliable.

3. R.RANJINI- Feature Engineering and modeling: Create useful features from the data such as user preferences, movie genres, or ratings.

4. M.JEEVITHA- Evaluation and optimization : Test how well the recommendation model is working by using accuracy and performance measures.

5. P.DIVYA DHARSHINI- Documentation and presentation: Keep clear records of the project steps, tools used, and results. Also prepare slides, reports, and visuals to explain the system to others in a simple and understandable way.