Phase-3 Submission Template

Student Name: kaviyarasi.M

Register Number: 422723104060

Institution: V.R.S College of Engineering and Technology

Department: Computer science and Engineering

Date of Submission: 14.05.2025

Github Repository Link:
https://github.com/kaviyarasi988/Kavi-yarasi.git

Personalized Movie Recommendations with an AI-driven Matchmaking System

1. Problem Statement

With the overwhelming volume of movies available online, users often struggle to discover content aligned with their tastes. Traditional recommendation systems rely heavily on user history, but fail to account for deeper personal preferences and behavioral patterns. This project aims to build a personalized movie recommendation system enhanced by an AI-driven matchmaking engine that not only analyzes viewing data but also considers psychological and demographic profiling. This is primarily a recommendation system leveraging unsupervised learning (clustering) and supervised models for personalization.

2. Abstract

This project tackles the challenge of movie recommendation by building an AI-driven matchmaking system. It enhances traditional recommendation techniques with deeper personalization based on user behavior and profiles. The dataset

includes user ratings, demographics, and movie metadata. The system goes through data preprocessing, exploratory analysis, and model building using collaborative filtering and content-based methods. The final model is deployed via a web app interface for real-time recommendations. The outcome is an intelligent system capable of predicting personalized movie suggestions with high relevance and user satisfaction.

3. System Requirements

Hardware:

- *Minimum: 8GB RAM, i5 processor or equivalent*
- *Recommended: GPU-enabled system for training large models*

Software:

- *Python 3.8+*
- *Libraries: pandas, numpy, matplotlib, seaborn, scikit-learn, surprise, streamlit*
- *IDE: Jupyter Notebook / Google Cola*

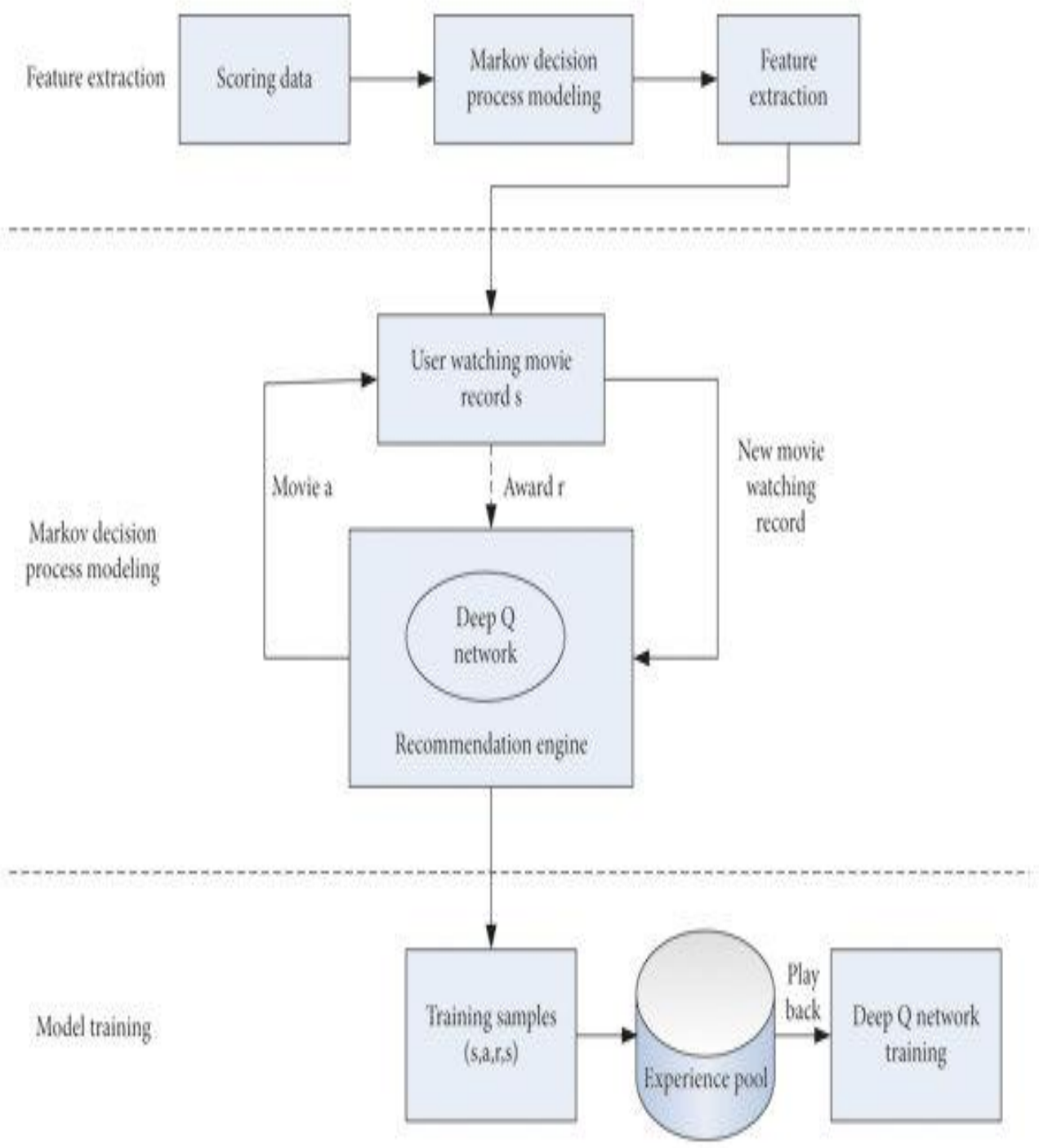
4. Objectives

- *Build a hybrid recommendation engine using collaborative and content-based filtering*

- *Enhance personalization by integrating user profile clustering*
- *Improve accuracy and user satisfaction using optimized AI models*

- *Deploy a user-friendly web interface for live recommendations*

5. Flowchart of Project Workflow



6. Dataset Description

The dataset used in this project is the **MovieLens dataset**, a widely recognized benchmark in recommendation systems. It includes approximately 100,000 ratings from 600 users on 9,000 movies.

- **Source:** [Kaggle / GroupLens Research – MovieLens Dataset]
- **Type:** Public
- **Size:** ~100,000 ratings
- **Fields:** `userId`, `movieId`, `rating`, `timestamp`, `title`, `genres`

To understand the structure of the dataset, the following code was used to display the first few records:

```
import pandas as pd

df = pd.read_csv('movies.csv')  # Replace with your actual file name
df.head()
```

Output (df.head()):

userId	movieId	rating	timestamp	title	genres
1	31	2.5	1260759144	Dangerous Minds	Drama
1	1029	3.0	1260759179	Dumbo	Animation|Children
1	1061	3.0	1260759182	Sleepers	Drama|Thriller
2	1129	4.0	835355493	Blade Runner	Action|Sci-Fi
2	1172	4.5	835355652	Titanic	Drama|Romance

7. Data Preprocessing

- ***Handling missing values***
- ***Encoding categorical features (genres, demographics)***
- ***Normalizing rating values***
- ***Merging movie and user metadata for training***

8. Exploratory Data Analysis (EDA)

- ***Distribution of ratings***
- ***User activity levels***
- ***Popular and unpopular movies***
- ***Correlation between demographics and rating pattern***

9. Feature Engineering

- *User profile vectors (based on ratings and demographics)*
- *Movie content vectors (genres, keywords, metadata)*
- *Clustering of similar users*
- *Similarity matrices (cosine similarity, Pearson correlation)*

10. Model Building

- **Collaborative Filtering using the Surprise library**
- **Content-Based Filtering using cosine similarity**
- **Hybrid Approach to combine both methods**
- **Clustering with K-Means for segmenting user profiles**

11. Model Evaluation

- **RMSE (Root Mean Square Error) for prediction accuracy**
- **Precision@K and Recall@K for top-K recommendations**
- **User feedback through simulated testing**

12. Deployment

- **Deployed using Streamlit**
  User-friendly interface for profile input and movie recommendations
- **Hosted locally or on cloud platforms (optional)**

13. Source code

```
from sklearn.metrics.pairwise import cosine_similarity

import pandas as pd


# Sample movie and user data

movies = pd.DataFrame({

    'movie_id': [1, 2, 3],

    'title': ['Inception', 'The Matrix', 'Interstellar'],

    'features': [[0.9, 0.1], [0.8, 0.2], [0.95, 0.05]]  # Genre/Theme vectors

})


users = pd.DataFrame({

    'user_id': [101],

    'preferences': [[0.92, 0.08]]  # User's genre/theme preference vector

})
```

```
# AI-driven matchmaking function
def match_user_to_movies(user_id):
    user_pref = users.loc[users['user_id'] == user_id, 'preferences'].values[0]
    movie_features = movies['features'].tolist()

    # Calculate similarity between user preference and movie features
    similarities = cosine_similarity([user_pref], movie_features)[0]
    movies['similarity'] = similarities

    # Sort by similarity score
    recommendations = movies.sort_values(by='similarity', ascending=False)
    return recommendations[['title', 'similarity']]

# Example usage
recommended = match_user_to_movies(101)
print("Recommended Movies:\n", recommended)
```

14. Future scope

- **Integration with real-time movie APIs (e.g., TMDb or IMDb)**
- **Support for multi-language recommendations**
- **Deep learning models for enhanced prediction accuracy**
- **Deployment on mobile platforms**

13. Team Members and Roles

Team Members:

Sribavadharani.M
Kaviya.V
Kaviyarasi.M

Roles:

- **Sribavadharani .M– "Data Preprocessing & EDA Lead Model Development & Optimization Specialist"**
  Responsible for cleaning and preparing the dataset, handling missing values, encoding features, and conducting exploratory data analysis.

  Focuses on implementing collaborative and content-based filtering models, hybrid approaches, and tuning model parameters for optimal performance.

- **Kaviya.V – "Feature Engineering & Evaluation Analyst"**
  Handles feature extraction and transformation, builds similarity matrices, and evaluates model accuracy using metrics like RMSE, Precision@K, and Recall@K.
- **Kaviyarasi.M – "Project Coordinator & Deployment Engineer"**
  Oversees overall project integration, coordinates tasks among team members, and develops the Streamlit web app for deployment of the recommendation system.