Web-Based Movie Recommendation System

By

Adithya.C

RRN:201421601006

BCA Data Science

Movie Recommendation System

**Introduction:**

Movies are a part and parcel of life. There are different types of movies like some for entertainment, some for educational purposes, some are animated movies for children, and some are horror movies or action films. Movies can be easily differentiated through their genres like comedy, thriller, animation, action etc. Other way to distinguish among movies can be either by releasing year, language, director etc. Watching movies online, there are a number of movies to search in our most liked movies. Movie Recommendation Systems helps us to search our preferred movies among all of these different types of movies and hence reduce the trouble of spending a lot of time searching our favourable movies. So, it requires that the movie recommendation system should be very reliable and should provide us with the recommendation of movies which are exactly same or most matched with our preferences.

**Problem Statement:**

Problem statement for this project is to create a web application that recommends a movie of various film industry around the world by the movie genre or movie name or with simple one line and accurate rating from various platform with their OTT platforms details.

**Overview**

Existing System:

* Collaborative Filtering: Recommendations are based on the similarity of user’s preferences.
* Content based Filtering: Recommendation are based on attributes of the movie, such as genre, cast, director, etc.
* Deep Learning Based Approaches: Use deep neural network to model the users and items to generate recommendations.

**Limitations of existing system:**

* Cold start Problem: New users or items might not have sufficient data for the recommendation system to work effectively.
* Scalability: Most users only watch a limited number of movies, making it difficult to find similarities between users.
* Personalization: Recommendation system can struggle to capture the unique tastes and preferences of individual users.

**SYSTEM REQUIREMENTS SPECIFICATION:**

* This chapter involves both the hardware and software requirements needed for the project and detailed explanation of the specifications.
* Hardware Requirements
* A PC with Windows/Linux OS
* Processor with 1.7-2.4gHz speed
* Minimum of 8gb RAM
* 2gb Graphic card

**Modules:**

* Module I : Creating database.
* Module II : Developing Web Application.
* Module III : Creating self-updating algorithm with API.
* Module IV : Creating Recommendation Algorithm.

**Activity Diagram:**

\ Home

Search (genres, actor name, director name, year, keyword)

Recommend Movie

Query in database with the searched keyword

Login with Google to fetch watch history

Content based filtering with watch history data.

Optimal result

**Explanation of the Activity Diagram:**

* The User searches for movies with keywords.
* The System returns a list of movies matching the keywords.
* To get recommendation the user has to login with id and the watch history is fetched and movies are recommended by content based filtering.

**Requirements Specification (SRS) document for a movie recommendation system:**

1. Introduction

The purpose of this document is to outline the requirements for a movie recommendation system. The system is designed to recommend movies to users based on their preferences and viewing history.

2. Scope

The movie recommendation system will be a web-based application that allows users to create an account, rate movies, and receive personalized movie recommendations. The system will incorporate machine learning algorithms to analyze user data and recommend movies based on their preferences.

**3. Functional Requirements**

3.1 User Registration:

The system shall allow users to create an account by providing their name, email address, and password.

3.2 User Login:

The system shall allow users to log in to their account using their email address and password.

3.3 Movie Ratings

The system shall allow users to rate movies on a scale of 1 to 5 stars.

3.4 Movie Recommendations

The system shall provide personalized movie recommendations based on user ratings and viewing history.

3.5 Recommendation Settings

The system shall allow users to adjust their recommendation settings, including the frequency of recommendations and the types of movies recommended.

3.6 Search

The system shall allow users to search for movies by title, genre, or actor.

3.7 User Profile

The system shall allow users to view their account information, including their movie ratings and viewing history.

**4. Non-functional Requirements**

4.1 Performance

The system shall be able to handle a large number of users and movie ratings without significant delay.

4.2 Usability

The system shall be intuitive and easy to use for users of all levels of technical expertise.

4.3 Security

The system shall protect user data by using encryption and secure login procedures.

4.4 Reliability

The system shall be available for use 24/7 with minimal downtime.

**Gantt chart:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Time line | | | | | |
| Tasks: | 5/02/23 | 12/02/23 | 25.2.23 -30.4.23 | 1.04.23-5.4.23 | 6.4.23 | 6.4.23- |
| Planning |  |  |  |  |  |  |
| Design |  |  |
| Development |  |  |
| Testing |  |  |
| Deployment |  |  |
| Monitoring |  |  |

Explanation of the Gantt chart:

The project planning and scheduling is divided into 6 activities: Planning, Design, Development, Testing, Deployment, and Monitoring.

The start and end dates for each activity are listed in the chart.

Planning starts on February 5, 2023 and ends on February 12, 2023.

Design starts on February 13, 2023 and ends on February 25, 2023.

Development starts on February 25, 2023 and ends on April 30, 2023.

Testing starts on April 1, 2023 and ends on April 5, 2023. Deployment starts on April 6, 2023

The Gantt chart provides a visual representation of the project timeline, allowing the team to track progress and make adjustments to the schedule as needed.

|  |
| --- |
| Movie |
| Title:string |
| Genre:string |
| Director:string |
| Actor:string |
| Year:int |
| Imdbrating:float |
| language |

**Class Diagram:**

**Web based Recommendation system**

|  |
| --- |
| User |
| Userid:int |
| Username:string |
| Email:string |
| Password:string |
| Lastlogin:date |

|  |
| --- |
| Recommendation |
| Title:string |
| Genre:string |
| Actor:string |
| Score:float |

**Explanation**:

* The **User** class represents a user of the system. It contains the user's unique **userId**, **username**, **email**, **password**, and **lastLogin** information.
* The **Movie** class represents a movie in the system. It contains the movie's unique **movieId**, **title**, **genre**, **releaseDate**, **director**, **actors**, **description**, and **averageRating** information.
* The **Recommendation** class represents a recommendation made by the system. It contains the recommendation's title, genre, actor, score.
* The **User** class has a one-to-many relationship with the **Recommendation** class, meaning that a user can make many recommendations.
* The **Movie** class has a one-to-many relationship with the **Recommendation** class, meaning that a movie can receive many recommendations.