Project Report

Smart Movie Recommender System using Machine Learning and Chatbot Interface

SUBMITTED IN THE PARTIAL FULFILLMENT REQUIREMENT FOR THE

AWARD OF DEGREE OF

Bachelor of Technology

(COMPUTER SCIENCE and ENGINEERING)

SUBMITTED BY

RISHABH KUMAR SINGH

(UNIVERSITY ROLL No: 230468)

UNDER THE SUPERVISION OF

DR SUKHANDEEP KAUR

SCHOOL OF ENGINEERING AND TECHNOLOGY

BML MUNJAL UNIVERSITY

FROM HERE TO THE WORLD

BML MUNJAL UNIVERSITY

Gurugram, Haryana - 122413

May 2025

CANDIDATE'S DECLARATION

I hereby declare that the project titled "Movie Recommendation Website with ML and Chatbot" (developed using HTML, CSS, JavaScript, and Machine Learning), submitted in partial fulfillment of the requirements for Semester Project-IV of the Bachelor of Technology in Computer Science and Engineering at BML Munjal University, is my original work. This project was conducted under the guidance of Dr. Sukhandeep Mam from March 2023 to May 2023. The report is a genuine representation of my efforts and has not been submitted elsewhere for any academic award.

Name: Rishabh Kumar Singh

University Roll No.: 230468

(**MENTOR)**

DR SUKHANDEEP KAUR

ABSTRACT

The “Smart Movie Recommender System” is a web-based application that assists users in discovering movies based on their preferences, leveraging machine learning and an interactive chatbot interface.

The system incorporates a content-based filtering algorithm that suggests films similar to those rated or searched by the user.

To enhance user engagement and usability, the website includes a chatbot developed using JavaScript that answers queries and recommends movies in a conversational format.

The frontend is built using HTML, CSS, and JavaScript for a responsive and engaging interface, while the recommendation model is implemented using Python. This hybrid solution bridges entertainment and AI, delivering a seamless recommendation experience.your abstract here.

ACKNOWLEDGEMENT

I am deeply grateful to BML Munjal University, Gurugram, for providing me with the opportunity to undertake this Semester Project-IV and for their unwavering support throughout the duration of March-May. I extend my sincerest gratitude to my mentor, Dr. Sukhandeep Mam, for her invaluable guidance, constant encouragement, and insightful feedback, which were instrumental in the successful completion of this project. Her expertise and patience helped me navigate challenges and refine my work at every stage. I would also like to express my heartfelt thanks to the faculty and staff of BML Munjal University for their academic support and resources. My sincere appreciation goes to my friends and peers for their motivation, constructive discussions, and timely assistance, which made this journey collaborative and enriching. Lastly, I acknowledge the contributions of all individuals who, directly or indirectly, supported me in accomplishing this project.

(RISHABH KUMAR SINGH

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Full Form
SVM	Support Vector Machine
LDA	Latent Dirichlet Allocation
NLP	Natural Language Processing
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
JS	JavaScript
API	Application Programming Interface
ER Diagram	Entity Relationship Diagram
ML	Machine Learning
UI	User Interface

TABLE OF CONTENTS

Contents	Page No.

TABLE OF CONTENTS

Chapter

Introduction to Organisation

BML Munjal University (BMU) is a leading not-for-profit university located in Gurugram, Haryana, established by the promoters of the Hero Group. The university is named after the late Dr. Brijmohan Lall Munjal, the visionary founder of Hero Group, and is built on the principles of innovation, excellence, and learning by doing.

BMU focuses on providing high-quality, industry-relevant education through a hands-on and experiential learning approach. The university offers undergraduate, postgraduate, and doctoral programs in engineering, management, law, and economics. With strong industry linkages and an emphasis on research and interdisciplinary learning, BMU aims to create leaders and problem- solvers who can address real-world challenges.

The university encourages students to work on live projects, internships, and research-based learning to bridge the gap between academic knowledge and industry requirements. BMU's collaborations with global institutions and corporate partners further enhance its commitment to delivering practical and impactful education.

This project, developed as part of the academic curriculum at BMU, reflects the university's emphasis on innovation, technology, and social impact through applied learning.

Chapter

Introduction to Project

2.1 Overview

As the growth in digital content and streaming websites has been exponential, customers get exposed to a plethora of movies and TV shows with far too many options to choose from. It's time-consuming and also frustrating to find the ideal movie as per personal taste. To counter this problem, this project presents a Movie Recommender Website that adopts Machine Learning and a smart Chatbot to facilitate better user experience through personalized movie recommendations.

The recommendation system uses collaborative filtering, content-based filtering, and hybrid methods to evaluate user patterns, movie metadata, and ratings. Based on user preferences and similarities between users and items, the system makes personalized movie recommendations. A chatbot interface is also embedded to support natural and interactive dialogue, where users can receive movie recommendations, find movies by genres, search for movies, and receive real-time help.

The chatbot applies Natural Language Processing (NLP) methods to understand user questions, answer conversationally, and lead users around the platform.

This integration of machine learning models and chatbot interaction is intended to provide a smooth and engaging experience that emulates human-like support.

In general, this project fills the gap of smart movie finding by integrating state-of-the-art ML models and NLP methods. It makes recommendation simpler and user-satisfied via personalization and interactive assistance.

2.2 Existing System

Most current movie recommendation websites are based either on simple filtering or user ratings without more in-depth personalization. Streaming sites such as Netflix and Amazon Prime do provide advanced recommendations, but their internal algorithms are not publicly available and are usually not explainable or interactive. Users can only receive recommendations passively based on previous behavior, with minimal user control or input.

Additionally, present systems usually don't enable dynamic interaction or clarification using conversational interfaces. When users are unsure or desire recommendations upon mood, genre blends, or certain

actors/directors, conventional UI elements such as dropdowns or filters become restrictive.

Also, a lot of open-source or smaller recommendation sites simply use content-based or collaborative filtering without bringing hybrid, context-based suggestions to the table. There is also no integration with conversational AI that can actually help users navigate alternatives, creating a mismatch between what the user desires and what the system provides.

This project fills these gaps by merging machine learning recommendation models with a chatbot interface to give both interactive and personalized recommendation

2.3 User Requirement Analysis

To ensure the Movie Recommendation Website meets user expectations, a detailed user requirement analysis was conducted. The users include casual movie watchers, enthusiasts, and administrators.

1. Functional Requirements

a) User Requirements:

- Account creation, login, and profile management.
- Search movies by name, genre, year, or actor.
- Receive personalized movie recommendations.
- Interact with a chatbot for suggestions or help.
- Rate and review movies to improve recommendations.

b) Admin Requirements:

- Add, update, or remove movie data.
- Monitor system performance and user activity.

- Manage user access and handle reported content.

2. Non-Functional Requirements

- Performance: Fast response time for movie queries and chatbot replies.
- Scalability: Capable of handling a growing number of users and movie data.
- Usability: Intuitive interface with smooth navigation and appealing design.
- Security: Protect user data with secure authentication and encryption.
- Maintainability: Modular codebase to allow easy updates of ML models and UI.

2.4 Feasibility Study

A feasibility study ensures the practical implementation of the Movie Recommendation Website. The system has been evaluated for the following aspects:

1. Technical Feasibility
The project uses Python, Flask/Django for backend, React/HTML for frontend, and machine learning libraries like scikit-learn, Surprise, or TensorFlow. NLP models for chatbot implementation are supported by tools like Rasa or OpenAI's API. These technologies are well-documented and widely adopted, making development achievable with current resources.

Chapter

2. Economic Feasibility
Open-source technologies significantly reduce development costs. Hosting can initially be done on free or low-cost platforms like Render or Heroku. Compared to developing a large-scale proprietary system, this solution is economically viable for academic or prototype deployment.

3. Operational Feasibility
Users can easily interact with the system through a clean web interface or chatbot. The design minimizes the learning curve and improves user satisfaction. Admin functionalities are also simple to operate and maintain.

4. Time Feasibility
The project is planned in modular stages: data collection, model building, chatbot integration, and UI development—making it realistic to complete within an academic timeline.

Literature Review

Recommender systems have come a long way over the last twenty years to enable users to find content that appeals to their preferences. With exponentially increasing volumes of content and user interactions, particularly in areas such as movies and media, researchers have consistently recommended ways to increase the accuracy, scalability, and personalization of recommendations.

Early on, content-based filtering methods were used to suggest items which are similar to what the user had liked before. This method works by matching item attributes (e.g., genre, director, or keywords) with the user's past preferences [1]. But it results in low diversity in recommendations since users are continuously suggested something similar.

To address these shortcomings, collaborative filtering approaches came into being, which learn user-item interactions to determine similarities between users or items. Sarwar et al. [2] showed how user-based and item-based collaborative filtering algorithms can be used for large-scale recommendation tasks. However, these systems find it difficult to handle the cold-start and data sparsity issues when there is inadequate information about new items or users.

In response, hybrid recommendation approaches came about, synthesizing the benefits of content-based and collaborative methods. Burke [3] introduced a number of hybridization strategies, including weighted, switching, and feature combination approaches, which served to enhance recommendation performance and address respective method limitations.

With the development of machine learning (ML) and deep learning, more advanced models like matrix factorization, autoencoders, and deep neural networks started being employed for latent feature extraction and scalable recommendations

3.1 Comparison

Research Work	Technique Used	User Interaction	Scalability	Accuracy
Sarwar et al. (2001) [2]	Item-Based Collaborative Filtering	No	High	Moderate
Burke (2002) [3]	Hybrid Recommender System	No	Moderate	High
Koren et al. (2009) [5]	Matrix Factorization	No	High	High
Christakopoulou et al. (2016) [6]	Conversational Recommendation (CRS)	Yes	Moderate	Moderate-High
Zhang et al. (2019) [4]	Deep Learning Models (Neural Nets)	Limited	High	High
Proposed System (This Project)	Hybrid (TF-IDF + Chatbot + Feedback)	Yes (Chatbot Enabled)	High	High

OBJECTIVES :

This project aims to bridge key gaps in existing movie recommendation systems through the integration of **machine learning models and conversational AI**. Based on the literature review and comparative analysis, the objectives are clearly defined as follows:

1. **To Develop a Hybrid Recommendation Engine:**

    - Combine TF-IDF vector similarity and Sentence-BERT semantic similarity to recommend movies.

    - Leverage content-based filtering initially, and integrate collaborative feedback over time.

2. **To Integrate a Chatbot for User Interaction:**

    - Build a chatbot interface using Natural Language Processing (NLP) to collect preferences and dynamically suggest movies.

    - Improve user engagement and personalization through conversational input rather than static filters or checkboxes.

3. **To Create a User-Friendly Web Interface:**

    ○ Design a clean, intuitive website where users can chat, get recommendations, and explore movie details.

<u>Exploratory Data Analysis (EDA)</u>

The dataset used for the Movie Recommendation system contains information about movies, user ratings, and other relevant features such as genres, release years, and movie IDs. The dataset is scraped from publicly available sources such as **IMDb** or **MovieLens** using web scraping techniques.

- **Features of the Dataset:**

    ○ **Movie Title:** The name of the movie.

    ○ **Genre:** The genre(s) associated with the movie (e.g., Action, Comedy, Drama, etc.).

    ○ **User Ratings:** The ratings provided by users for each movie.

    ○ **Release Year:** The year in which the movie was released.

    ○ User-specific ratings or preferences for movies (for collaborative filtering).

Chapter

Methodology

1. Introduction to Languages (Front-End and Back-End)

- Front-End: The front-end is responsible for the user interface (UI) of your project. Common languages and frameworks include:
  - HTML/CSS: For structuring and styling the web pages.
  - JavaScript (or TypeScript): For dynamic functionality. Frameworks like React or Vue.js can be used for building interactive UIs.
  - Bootstrap/Tailwind CSS: For responsive design and fast prototyping.
- Back-End: This handles the business logic, model integration, and data management. Common back-end languages and frameworks include:
  - Python (Flask or Django): For handling API requests and integrating the ML models.
  - Node.js (Express.js): If JavaScript is preferred on both front-end and back-end.

  - Ruby on Rails: For rapid back-end development.

2. Supporting Languages/Packages

- Python Libraries:
  - Pandas & NumPy: For data manipulation and preprocessing.
  - Scikit-learn: For machine learning models and evaluation.
  - TensorFlow or PyTorch: For more advanced ML models (e.g., neural networks).
  - NLTK or SpaCy: For text-based features and natural language processing (NLP).

3. User Characteristics

- Target Audience: Identify the user characteristics and needs. For instance, if it’s a movie recommendation system, the target audience could be movie enthusiasts who need personalized movie suggestions.

Chapter

- User Features: Users might provide data such as:
    - Movie preferences (genre, rating).
    - Demographics (age, location, etc.).
    - Interaction data (watch history, ratings given).
- User Interface Requirements: The UI should be user-friendly, displaying.

Results

1. Data Preprocessing

- The dataset was cleaned with no significant missing values or duplicates.
- **Feature Engineering** included creating a **Release Year** from the movie title and converting **Rating** to a numeric type.

2. Exploratory Data Analysis

- **Rating Distribution:** Most movies had ratings between **3.0 and 5.0** stars, with a peak around **4.0** stars.

Genre Distribution: Action, **Comedy**, and **Drama** were the most common genres in the dataset.

- **Correlation Analysis:** No significant correlation between **Release Year** and **Ratings**, but genre-based correlations indicated certain genres had higher average ratings.

3. Key Findings

- The dataset is dominated by **Action**, **Comedy**, and **Drama** genres.
- **Ratings** tend to be moderate (3-5 stars).
- No strong link between movie **release year** and ratings, but genre plays a role in movie ratings.

EDA revealed key patterns in ratings and genre distribution, helping us shape the approach for the **Movie Recommendation System**.

Chapter

Conclusion

7.1 In this project, we created a Movie Recommendation System based on machine learning methods for presenting personal recommendations depending on the user's preferences. The system incorporates a blend of Collaborative Filtering and Content-Based Filtering to make recommendations specific to individual users. Through the application of Python libraries like Scikit-learn, Flask, and Pandas, the model is built for efficient scalability and to present exact recommendations to the users in real-time.

Key steps involved: Data Collection: We sourced the data from a reliable dataset (e.g., MovieLens) and processed it using data preprocessing techniques like cleaning, normalization, and feature engineering.

Model Development: Machine learning algorithms, specifically Collaborative Filtering (e.g., matrix factorization) and Content-Based Filtering were employed to generate accurate recommendations.
Implementation: The system was implemented with a Python-based back-end (Flask), and an interactive front-end was implemented to display the recommendations in an easy-to-use manner.

Future Scope:

Although the existing movie recommendation system performs well, there is always space for improvement and extension. What follows are a number of directions for future research:

Hybrid Recommendation System:

The system would profit from having a hybrid model that unites collaborative filtering, content-based filtering, and deep learning methods (e.g., neural collaborative filtering). This would once again enhance the precision of recommendations by identifying elaborate patterns in the data.

Incorporation of Real-Time Data:

The system could be further enhanced to real-time suggestions, with users being provided with updated recommendations based on their latest activities (e.g., films watched or rated). The inclusion of stream data would make the system dynamic and enhance the timeliness of recommendations.

Sentiment Analysis for Movie Reviews

By using Natural Language Processing (NLP) methods such as sentiment analysis in movie reviews, the system was able to identify the overall emotional tone surrounding a movie and align recommendations with the emotional tastes of users.

User Feedback Loop

Adding a feedback loop where users can explicitly rate the suggestions can enable the system to learn from user tastes over time, making subsequent suggestions better. This can be done by updating the model periodically using new feedback data.

Bibliography

[1] 1.Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. *Proceedings of the 10th international conference on World Wide Web*.

[1] 2.Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, 42(8), 30-37.

[2] 3.Pazzani, M. J., & Billsus, D. (2007). Content-based recommendation systems. *The Adaptive Web*, 325-341.

[3] 4.Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4), 331-370.

[4] 5.He, X., Lian, J., Zhang, H., Kang, W., & McAuley, J. (2017). Neural collaborative filtering. *Proceedings of the 26th International Conference on World Wide Web*, 173-182.

[5] 6.Ansari, A., Essegaier, S., & Kohli, R. (2000). Internet recommendation systems. *Journal of Marketing Research*, 37(3), 363-375.

Appendix (Any additional Information regarding Project)

Dataset Overview

- **Source:** IMDb, MovieLens (or another dataset provider).
- **Size:** The dataset includes **10,000 movies** and **1 million ratings**.
- **Features:**
  - **Movie Title:** Name of the movie.
  - **Genre:** Movie genre(s) (e.g., Action, Comedy, Drama).
  - **Rating:** User ratings for each movie (e.g., 1 to 5 stars).
  - **Release Year:** The year when the movie was released.
  - **Movie ID:** Unique identifier for each movie.
  - **User Ratings (if applicable):** User ratings for movie preferences.