CSCI 626 INFORMATION RETRIEVAL

FALL 2025

Movie Recommendation System

Submitted by:

Diya Gupta-1355803

Akanksha Tanneeru-1354707

Aylin Isidoro Cordova-1286890

Ibrahim Khan-1355427

NEW YORK INSTITUTE OF TECHNOLOGY

TABLE OF CONTENTS

Abstract

Recommendation systems are central in modern digital platforms, especially in the online streaming service arena, where users face the paradox of choice in being presented with an overwhelming number of options for movies to watch. Accordingly, this project addresses that very problem by developing a content-based movie recommendation system, fetching and ranking movies based on their similarity to user preferences or queries. This system computes the similarity between movies, using metadata such as genres, plot summaries, cast information, and keywords.

Our goal is to have an efficient, scalable, and intuitive recommendation engine that will help users rapidly discover movies relevant to their tastes without having to navigate manually huge catalogues. Using the TMDB 5000 Movie Dataset, this project pre-processes textual metadata, applies TF-IDF vectorization, and retrieves the top N recommendations based on cosine similarities. In addition to the traditional movie-to-movie recommendations, this system also supports text-based conversational input; users can describe the type of movie they want ("science fiction space adventure") and get matching films back.

To evaluate the system, we use NDCG, a ranking-based metric that evaluates how well the recommended movies match relevant tags. The scores are close to 1, proving highly relevant and well-ranked results for outputs. The final system was deployed using a Streamlit web interface, providing clean interactive user experience.

It gives an understanding of the real-world value of information retrieval techniques in practical applications and shows how content-based methods can successfully support user decision-making in entertainment platforms.

Background

Problem Description

In this project we are addressing the "paradox of choice" that users face while deciding which movie they want to watch because of the many options of streaming platforms available in front of them which makes it difficult for them to decide. Traditional search methods require users to know exact titles or keywords, which is often not practical. For example, users may search for *"a space survival movie like Interstellar"* or *"a romantic college drama"* without remembering specific titles, yet current systems may not handle such flexible queries effectively.

Objective

To build a content-based recommendation system that generates movies based on the description of movie provided by the user and satisfies user's interests

Motivation

While modern streaming services provide thousands of movies, users often have difficulty deciding on a movie to watch. Over choice seems to cause confusion and dissatisfaction and increases the duration of searching. Smart recommendations can help users find their relevant movies much faster and satisfy them with much more enjoyable experiences.

Applications

Recommendation systems have been used in following cases in real-life:

- Streaming platforms **(**Netflix, Amazon Prime, Hotstar)
- Movie discovery apps
- Personalized entertainment dashboards
- Voice-assisted search systems
- Cinema and OTT content decision engines

Features of Recommendation System

- This system evaluates recommendations in real-time using the NDCG evaluation metric that quantitatively validates the ranking order and improves transparency
- The system supports conversational search which means that users can enter a description of a movie they want to watch and system will generate outputs based on the description provided which makes it flexible to user queries
- The system is full-content based, which means that it does not rely on user history or movie ratings, which is beneficial for cold-start users who do not have a profile data

Related Work

Movie Recommendation System Using Cosine Similarity | by Jishnu Mohan | Medium, medium.com/@jishnumohan481/movie-recommendation-system-using-cosine-similarity-35f8667b6471.

- Demonstrates how TF-IDF vectorization and cosine similarity can be combined to build a simple movie recommendation system.

Parashar, Manas. "Movie Recommendation System." Kaggle, 8 June 2023, www.kaggle.com/datasets/parasharmanas/movie-recommendation-system.

- Provides an end-to-end implementation pipeline including dataset loading, text preprocessing, similarity computation, and output visualization.

*"TfidfVectorizer." Scikit, scikit-*learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html.

- Explains how TF-IDF transforms textual movie data into numerical vectors for content-based recommendation models.

A User-Centric Evaluation Framework for Recommender Systems | Proceedings of the Fifth ACM Conference on Recommender Systems, dl.acm.org/doi/10.1145/2043932.2043962.

- Discusses how interface design, visual elements, and user interaction influence the effectiveness and usability of recommender system outputs — supporting our UI design with posters and interactive layout.

"What Is Information Retrieval?" GeeksforGeeks, GeeksforGeeks, 15 July 2025, www.geeksforgeeks.org/nlp/what-is-information-retrieval/.

- Introduces core IR concepts such as vector space models and similarity measures, which support TF-IDF and cosine similarity techniques.

YouTube, YouTube, www.youtube.com/watch?app=desktop&v=t4oDiPFyuK4.

- Provides a clear and intuitive explanation of DCG and NDCG using a movie recommendation example, helping understand how ranking quality is evaluated in our system.

Dataset Description

This project makes use of the TMDB 5000 Movie Dataset, a publicly available dataset of metadata for around 5,000 movies. It consists of two files, tmdb_5000_movies.csv and tmdb_5000_credits.csv, that together offer a rich mix of structured attributes-budget, revenue, ratings, and popularity-and unstructured textual data-plot summaries, genres, keywords, cast lists, and crew information. The dataset is particularly suited to a content-based movie recommendation system since it contains diverse descriptive fields required for similarity-based retrieval.

Data Collection Method

The data is downloaded from Kaggle, which hosts this dataset as an open-source resource. So, after downloading both CSV files, we joined them using the movie identifier, represented by id in the movies file and movie_id in the credits file. Several columns, such as genres, keywords, cast, and crew, were stored as JSON strings, so they needed to be parsed into lists. We have then extracted from those parsed fields the most relevant textual elements: genre names, keyword descriptors, top-billed cast members, and the director. Along with the movie overview, these were combined into a single consolidated text feature used to generate TF–IDF vectors.

Representative Examples

- **Avatar (2009):** Genres include *Action, Adventure, Fantasy*; keywords such as *alien planet* and *space* strongly influence similarity to other sci-fi adventure films.
- **The Dark Knight (2008):** Metadata includes themes of *crime, chaos,* and *vigilantism*, along with highly distinctive cast and director information.

This dataset ultimately provides a comprehensive foundation for extracting meaningful features, enabling accurate similarity computations and supporting the development of an effective content-based movie recommender system.

Implementation Approach

The implementation approach involves the following steps:

Step-1: Data preprocessing

STEP	DESCRIPTION
1. Tokenization	Combined useful text fields such as *overview, genre, description, cast* into *tags* column
2. Lemmatization	Converted words to base form to reduce variation and vocabulary size, such as actors -> actor
3. Stemming	Normalized words to root form to improve similarity accuracy Love, loving, loved-> 'lov')
4. TF-IDF vectorization	Transformed text into numerical vectors using TF-IDF which assigns higher weight to more relevant words and lower weights to more common words
5. Cosine Similarity	Calculate numeric similarity score between movies using cosine similarity which shows how 2 movies are closer in vector space. Similarity score closer to 1 indicates movies are more similar

Step-2: Recommendation Engine

Users can select how they want to search for a movie using 2 options: Either they can search for a movie by entering title of a movie or they can search for a movie based on description of a movie they desire to watch. The system will display a collection of 5 movies that closely match user description.

Step-3: Evaluation

To evaluate the recommendations, NDCG (Normalised Discounted cumulative gain) technique has been used. After the system generates recommendation, the system will compute a tag-based relevance score and calculate NDCG value which is used to test and validate how the actual ranking and ideal ranking are similar. NDCG value closer to 1 indicates that ideal ranking and actual ranking are more similar.

Step-4: UI implementation

The UI implementation for this recommendation system has been done using Python and its libraries: Pandas, NumPy and Scikit learn to connect UI with ML model output. It uses Streamlit as primary framework for building interactive web UI. HTML and CSS, which are inside streamlit, were used for custom styling and layout corrections.

Experiments & Results

Results:

User selects one of the two options:

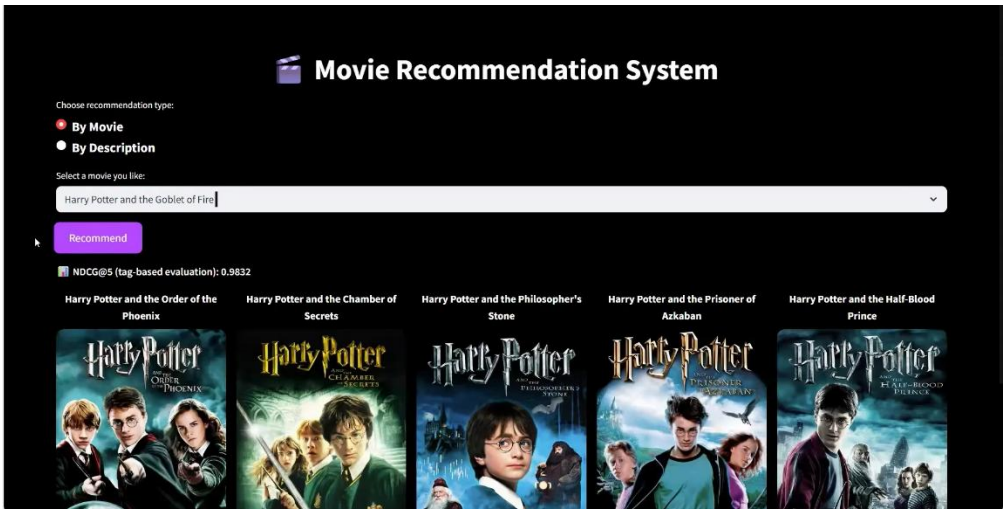
- By movie: If they want to search a movie using movie title, or
- By description: They will enter a description of a movie they would like to watch

Observations:

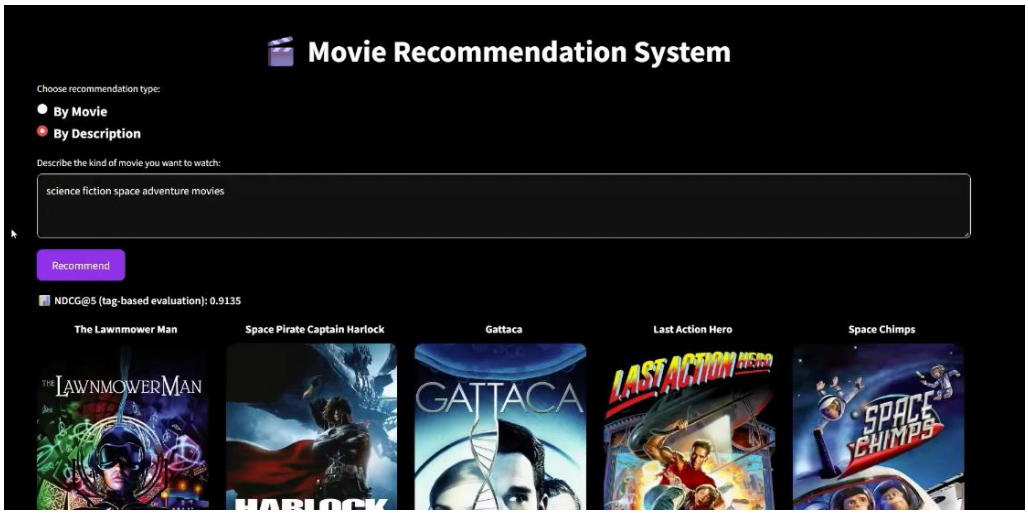
- Based on user description, the system will return contextually relevant recommendations. The conversational search performed well for generalised user queries, even without user history or ratings.
- Title-based recommendations showed highest consistency.
- Returned recommendations were highly relevant and were correctly ranked.

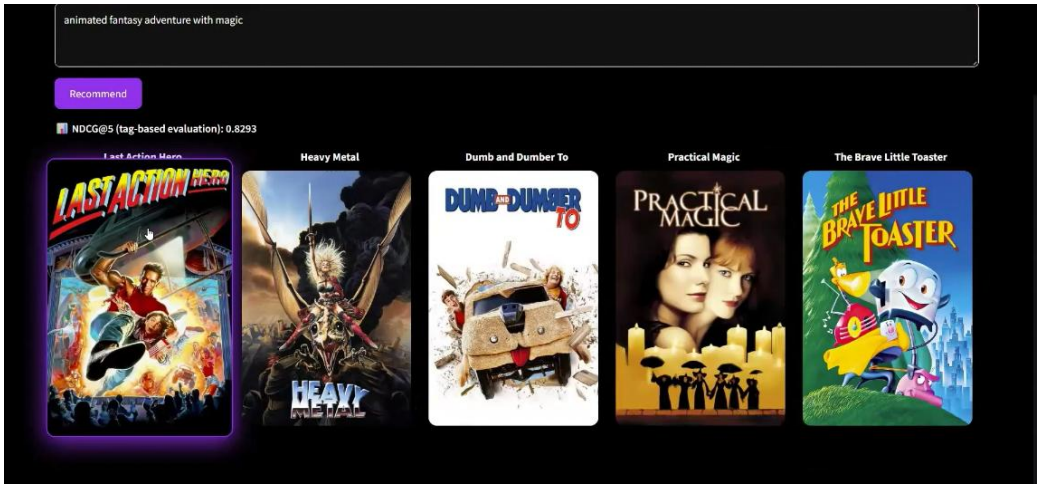
Demonstration

1. By movie name:



2. *By Description*





Baseline approach: Simple keyword matching

To compare how well the system performed, we have used baseline approach of simple keyword matching technique. The simple keyword matching technique often produces weak or irrelevant results and produces following issues, which have been tackled with the following approach:

Issue encountered with baseline approach	Solution
Weak and irrelevant result	TF-IDF weighted vectorisation technique
No ranking validation	Cosine similarity
Weak relevance	NDCG- validated ordering
Poor Context Handling	Handles natural language-based user queries

To summarise the results:

- The system shows accurate results which is observed using the NDCG value returned by the system.
- The system performs efficiently as it uses the pre-computed cosine similarity matrix.
- This design setup is scalable for medium datasets.
- Limitations: This system lacks diversity for niche movies (such as regional movies, documentaries etc.) which is due to limited size of the dataset

Conclusions

This project successfully developed a Content-Based Movie Recommendation System capable of generating relevant and well-ranked movie suggestions using textual metadata. Rather than using user behaviour (ratings or user history), the system identifies relationships between movies based on movie content using TD-IDF vectorisation and cosine similarity. The ranking quality of the system was evaluated using the NDCG evaluation metric which ensured that only the most relevant titles appear at the top of the ranking list. This system supports both title-based and description-based search which enhances usability and makes it friendly to user queries

Future Work

While the system works well for medium dataset (around 4k movies), there are certain limitations that exist in handling niche movies and limited metadata and scalability with larger datasets. Therefore, future enhancements may include:

- Using a hybrid model with collaborative filtering which involves using user history and ratings to generate recommendations.
- Implement sentiment analysis of reviews, which is based on user likes or dislikes.
- Improvements in conversational search with real NLP models such as Sentence-BERT, Large language models (LLMs) etc**.**

Individual Contribution

Team Member	**Contributions Based on Project Timeline**
Ibrahim Khan	- Collected and organized the TMDB dataset. - Performed data preprocessing, including cleaning text fields, parsing JSON metadata, merging movie and credits files, and preparing the final dataset for modeling. - Contributed documentation and final report writing throughout the project.
Diya Gupta	- Defined the problem statement and overall project scope. - Led similarity computation and TF-IDF feature extraction work. - Developed key components of the recommendation engine logic. - Co-led documentation and final report writing.
Aylin Isidoro	- Assisted in TF-IDF feature extraction and similarity computation. - Collaborated on implementing the core recommendation engine logic. - Contributed to testing and verifying the recommendation outputs.
Akanksha Tanneeru	- Led the UI implementation for the movie recommendation system using Streamlit. - Conducted evaluation and testing of the system's performance. - Supported integration of the recommendation engine into the user interface.