IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

# Movie Recommendation System

**Dr. Beenarani Manoj[1], Deepak G[2], Praveen K[3], Varun VV[4]**

Department Of Computer Science and Engineering, Faculty of Engineering and Technology,

SRM Institute of Science and Technology, Vadapalani, Chennai, Tamil Nadu, India[1-4]

**Abstract:** Choosing a movie from a large catalogue can be difficult because viewers must explore titles, genres, descriptions and other details before finding films that match their interests. This paper presents a content-based Movie Recommendation System that analyses movie metadata and recommends films with characteristics similar to a user-selected title. The system considers attributes such as genre, keywords, overview, cast, director, rating and release year. Textual metadata is transformed into numerical vectors using Term Frequency–Inverse Document Frequency (TF-IDF), and Cosine Similarity is used to compare movie representations. The resulting similarity scores are used to rank candidate films and present relevant recommendations. The project also provides a user-facing discovery interface with title-based search, genre and rating filters, and a locally maintained watchlist. The current implementation focuses on content-based recommendation using TF-IDF and Cosine Similarity. Collaborative filtering, hybrid recommendation, deep-learning embeddings, review sentiment analysis, continuous learning and a mobile client are proposed as future enhancements. The project presentation identifies Precision@K, Recall@K, relevance, coverage, diversity and response time as evaluation measures, but leaves numerical results pending; therefore, no unverified performance values are claimed in this paper.

**Keywords:** Movie Recommendation, Content-Based Filtering, TF-IDF, Cosine Similarity, Machine Learning, Movie Metadata, Information Retrieval.

## 1. INTRODUCTION

The growth of digital entertainment platforms has made large movie catalogues accessible to viewers. Although this abundance increases choice, it also creates an information-overload problem. Users may find it difficult to identify films that match their interests, particularly when they do not already know the title they want to watch. Conventional title-based search supports direct lookup but provides limited help in discovering unfamiliar movies with related themes, genres or descriptions.

Content-based recommendation systems address this problem by comparing the characteristics of items. For movies, useful characteristics include genres, keywords, plot overviews, cast, director, ratings and release year.
Manual discovery: viewers may need to browse many titles and compare descriptions before finding related movies.
Limited matching: simple title or keyword search may not identify films with similar metadata or themes.
Personalisation gap: recommendation quality can be limited when systems do not use item characteristics or user preference signals effectively.

To address these challenges, we developed a Movie Recommendation System based on content-based machine learning. The system prepares movie metadata, converts textual information into TF-IDF vectors, calculates similarity using Cosine Similarity and ranks movies according to their similarity to a selected title. The application also provides search, genre and rating filters, and a local watchlist to make movie discovery more convenient.

## 2. LITERATURE REVIEW

Recommendation systems are widely used to reduce information overload by selecting and ranking items that may be relevant to users. Research commonly groups recommendation methods into collaborative filtering, content-based filtering and hybrid approaches. These methods differ in the information they use and the way they estimate relevance.

**Traditional Search and Filtering:** Basic discovery tools depend on title, keyword or category searches. They are straightforward to use, but users must often know what they are looking for and manually inspect the results.

**Content-Based Filtering:** Content-based systems represent items through their attributes and recommend items that share characteristics with a selected or preferred item. In a movie catalogue, metadata such as genres, keywords, cast and plot descriptions can be used to compare films. Son and Kim [2] discuss content-based filtering through multi-attribute networks, supporting the use of several item attributes rather than a single descriptor.

 This work is licensed under a Creative Commons Attribution 4.0 International License 

**Term Weighting and Text Representation:** Salton and Buckley [3] describe term-weighting approaches for automatic text retrieval. TF-IDF assigns importance to terms according to their frequency in a document and their relative rarity across a collection. This makes it useful for representing movie descriptions and other textual metadata as numerical vectors.

**Similarity and Recommendation:** Cosine Similarity compares two vectors by measuring the cosine of the angle between them. It is commonly used with sparse text vectors to estimate how closely two descriptions are related. Scikit-learn provides practical implementations of TF-IDF vectorisation and cosine similarity [4], [5]. Hybrid recommendation research, including the work of Lekakos and Caravelas [1], motivates future systems that combine content information with additional recommendation signals.

## 3. PROPOSED SYSTEM ARCHITECTURE

The proposed system follows a content-based processing pipeline that converts movie metadata into numerical representations, calculates similarity and presents ranked recommendations through a user-facing application.

### 1.1 System Features

Movie Metadata and Feature Extraction: The system uses available movie information such as genre, keywords, overview, cast, director, rating and release year. Textual fields are prepared and transformed into TF-IDF feature vectors.

Similarity and Ranking Module: Cosine Similarity compares the selected movie representation with other movie representations. Candidate movies are ranked by their similarity scores and presented as recommendations.

Movie Discovery Interface: The application supports title-based search, genre and rating filters, and a watchlist stored locally. These functions allow users to find titles, narrow the catalogue and save movies for later viewing.

## 4. METHODOLOGY

### 4.1 Dataset and Features

The project presentation describes a movie metadata dataset containing titles, overviews, genres, cast, crew, ratings and release years. These attributes provide the descriptive information used by the content-based recommendation process. The exact dataset name, provider, version and number of records are not specified in the presentation and must be confirmed from the project files.

- Title: identifies each movie and supports title-based search.
- Genre and keywords: describe the categories, topics and themes associated with a movie.
- Overview: provides a textual summary that can contribute descriptive terms to the feature representation.
- Cast and director: provide information about the people associated with a movie.
- Rating: provides an additional movie attribute for filtering and catalogue exploration.
- Release year: identifies the period in which the movie was released and supports year-based filtering.

### 4.2 Data Preprocessing

Metadata Preparation: Relevant fields are selected and organised so that each movie has a consistent representation. Text fields used for recommendation are prepared before vectorisation. The exact handling of missing values, duplicate records and text normalisation should follow the implemented project code.

Feature Construction: The selected textual attributes are represented in a form suitable for TF-IDF vectorisation. Combining descriptive fields allows the system to compare movies using more than one aspect of their metadata.

### 4.3 TF-IDF Feature Extraction

TF-IDF (Term Frequency–Inverse Document Frequency) converts movie text into numerical vectors. Term Frequency reflects how often a term occurs in a movie description, while Inverse Document Frequency reduces the weight of terms that appear across many descriptions. The resulting vectors emphasise terms that help distinguish one movie from the catalogue. The project presentation identifies TF-IDF as implemented using the scikit-learn machine-learning library.

 This work is licensed under a Creative Commons Attribution 4.0 International License 

## 5. EXPERIMENTAL RESULTS

### 5.1 Evaluation Metrics

The project presentation identifies Precision@K, Recall@K, recommendation relevance, catalogue coverage, diversity and response time as the measures for evaluating the recommendation system. Numerical experiment results are marked as pending in the presentation and are therefore not reported as completed results here.

- Precision@K: measures the proportion of the top-K recommended movies that are judged relevant.
- Recall@K: measures the proportion of relevant movies that appear within the top-K recommendations.
- Recommendation relevance: assesses whether the suggested movies are appropriate for the selected title or user preference.
- Coverage: measures how much of the available movie catalogue can be recommended.
- Diversity: examines whether the recommended list contains meaningfully different movie choices.
- Response time: measures how long the system takes to produce a recommendation list.

### 5.2 Recommendation Process

- When a user selects a movie, the system obtains its prepared feature representation and compares it with the representations of other movies in the catalogue.
- TF-IDF transforms the selected movie metadata and catalogue descriptions into numerical vectors.
- Cosine Similarity calculates similarity scores between the selected movie and candidate movies.
- The candidate movies are ranked according to their similarity scores, with higher-scoring movies appearing earlier in the recommendation list.
- The project presentation does not provide a completed test protocol, relevance labels or measured scores. A quantitative evaluation should be conducted using a documented dataset split or a clearly described user-relevance assessment before numerical performance claims are added.

## 6. DISCUSSION

**Content-Based Recommendation:**
The system uses movie metadata to identify films with similar descriptive characteristics. This approach can generate recommendations from a selected title without requiring a long history of user ratings. TF-IDF and Cosine Similarity provide an understandable processing pipeline that can be inspected and improved as the dataset and feature preparation evolve.

**Practical Usability:**
Title-based search helps users locate a movie, while genre and rating filters narrow the available choices. The local watchlist provides a way to save titles for later viewing. Together, these features make the recommendation output easier to explore and act upon.

**Observed Limitations:**
Recommendation quality depends on the completeness and accuracy of the movie metadata. Content-based results can overemphasise similarity and may lack novelty. New or poorly described items can be difficult to recommend, and personalisation is limited without a longer history of user interactions. The project presentation also identifies cold-start problems, dataset quality, over-similarity and popularity bias as limitations.

## 7. CONCLUSION AND FUTURE WORK

The Movie Recommendation System applies content-based machine learning to simplify movie discovery. It represents movie metadata using TF-IDF and calculates similarity through Cosine Similarity to rank films related to a selected title.

 This work is licensed under a Creative Commons Attribution 4.0 International License 

The project also includes movie search, genre and rating filters, and a local watchlist. The approach provides a practical foundation for metadata-driven recommendations, while quantitative performance remains to be established through formal evaluation.

**Future Scope**

**Collaborative Filtering:**
Combine movie-content features with user-item interaction patterns to improve recommendations and address limitations of relying only on item metadata.

**Hybrid Recommendation:**
Integrate content-based and collaborative methods so that recommendations can use both movie characteristics and patterns across users.

**Deep Learning and Sentiment Analysis:**
Explore semantic embeddings for movie summaries and analyse review sentiment to capture relationships that may not be represented by direct keyword overlap.

**Continuous Personalisation and Mobile Access:**
Use user feedback to refine preference profiles over time and develop a mobile client for easier access to recommendations.

## REFERENCES

[1] G. Lekakos and P. Caravelas, "A Hybrid Approach for Movie Recommendation," Multimedia Tools and Applications, 2008. doi: 10.1007/s11042-006-0082-7.
[2] J. Son and S. B. Kim, "Content-Based Filtering for Recommendation Systems Using Multiattribute Networks," Expert Systems with Applications, 2017. doi: 10.1016/j.eswa.2017.08.008.
[3] G. Salton and C. Buckley, "Term-Weighting Approaches in Automatic Text Retrieval," Information Processing & Management, vol. 24, no. 5, pp. 513–523, 1988. doi: 10.1016/0306-4573(88)90021-0.
[4] Scikit-learn, "TfidfVectorizer Documentation." Available: https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.Tfidf Vectorizer.html
[5] Scikit-learn, "Cosine Similarity Documentation." Available: https://scikit-learn.org/stable/modules/metrics.html#cosine-similarity
[6] GroupLens Research, "MovieLens Datasets." Available: https://grouplens.org/datasets/movielens/
[7] The Movie Recommendation System project presentation, "Movie Recommendation System: An Intelligent Machine Learning-Based Movie Discovery Platform," project slides, 2026.
[8] F. Ricci, L. Rokach, and B. Shapira, eds., Recommender Systems Handbook, 2nd ed. Springer, 2015.
[9] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.
[10] G. Salton, A. Wong, and C. S. Yang, "A Vector Space Model for Automatic Indexing," Communications of the ACM, vol. 18, no. 11, pp. 613–620, 1975.

 This work is licensed under a Creative Commons Attribution 4.0 International License 