www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)
(Int Peer Reviewed Journal)
Vol. 05, Issue 04, April 2025, pp : 2024-2030

e-ISSN : 2583-1062
Impact Factor : 7.001

CONTENT-BASED MOVIE RECOMMENDATION SYSTEM

Apurva M. Bhavsar[1], Hitesh S. Girase[2], Dipak S. Lohar[3], Aatif Z. Shaikh[4]

[1]Asst. Professor, Dept. of Artificial Intelligence and Data Science, Guru Gobind Singh College of Engineering and Research Centre, Nashik, Maharashtra, India.

[2,3,4]Student, Dept. of Artificial Intelligence and Data Science, Guru Gobind Singh College of Engineering and Research Centre, Nashik, Maharashtra, India.

ABSTRACT

This paper introduces a content-based movie recommendation system that recommends movies by examining inherent movie attributes like genres, keywords, plot summaries, cast, and crew. Unlike collaborative filtering methods, our system is not based on user preferences or historical interactions, thus being effective even in cold-start situations. The system implements Natural Language Processing (NLP) techniques to preprocess and apply semantic textual features from the TMDB 5000 Movie Dataset. Features are converted into numeric vectors using the CountVectorizer and TF-IDF methods, while cosine similarity is used to establish the level of similarity between the movies. The model is implemented in Python, and it employs libraries such as Pandas, NumPy, Scikit-learn, and NLTK. The evaluation was carried out by considering the thematic relevance of the recommended movies to the selected movie. Content-based movie recommendations demonstrate that content-based filtering is capable of making precise and pertinent movie recommendations and enhancing the movie-watching experience of the user without requiring the presence of prior user behavior information. This method is appropriate for new platforms or users who have short histories of activity and may serve as a baseline system in hybrid recommender systems.

Keywords: Content-based filtering, Cosine similarity, Natural Language Processing, Movie recommendation, Feature extraction, Machine learning.

1. INTRODUCTION

In the modern world of digital technology, users are bombarded with thousands of entertainment choices. It becomes hard to pick a good movie to watch without any direction. Recommendation systems assist users by providing them with movie suggestions according to their likes or interests. These systems enhance the experience of the user by giving smart, personalized recommendations.

In this paper, a content-based movie recommendation system that emphasizes examining the content or properties of films like genres, keywords, plots, actors, and directors to provide recommendations of similar movies is offered.[2] Differently from user-based and interaction-based collaborative filtering algorithms, content-based filtering intervenes on movie characteristics themselves. Thus, content-based filtering remains active even if user history data do not exist in order to combat the cold-start problem.[4] Machine learning models and natural language processing (NLP) are exploited by the system to find comparable movie descriptions and similar content.[7] It is written in the Python programming language and heavily utilizes libraries like Pandas, NumPy, Scikit-learn, and NLTK. The dataset used here is the TMDB 5000 Movie Dataset found on Kaggle. This paper demonstrates that content-based filtering can effectively recommend relevant and meaningful movie suggestions. It is easy to run and provides satisfactory results without relying on user ratings or reviews.

2. METHODOLOGY

This research develops a content-based movie recommendation model that investigates important properties of movies like genres, plots, keywords, cast, and crew. We plan to give recommendations for similar movies based solely on content, without the need for user ratings or history. First, we download and preprocess the data to facilitate machine processing.

Then, natural language processing (NLP) techniques are used to process and convert the text into useful patterns with the help of Count Vectorizer and TF-IDF techniques.[7] Cosine similarity is calculated to determine how similar two films are in terms of content. After that, movies that are most similar to each other based on these similarity values are suggested by the system. This process is simple, fast, and effective, even without the need for the user rating, reviews, and user past history.

2.1 Dataset Collection- The first task in content-based movie recommender system was to acquire an actual dataset that had a wealth of information regarding movies. We chose the TMDB 5000 Movie Dataset from Kaggle as it includes several pieces of information such as the names of the movies, genres, descriptions, keywords, and cast/cast information. The dataset consists of two CSV files: tmdb_5000_movies.csv contains information such as genres, overviews, taglines,

www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)
(Int Peer Reviewed Journal)
Vol. 05, Issue 04, April 2025, pp : 2024-2030

e-ISSN : 2583-1062
Impact Factor : 7.001

and keywords. tmdb_5000_credits.csv, which contains the cast and crew details of the movie. We combined these files on the basis of the film title as a key so that all the necessary information was brought together in one location.[6] This exercise of combining gave us an organized and complete dataset that formed the foundation for building and analyzing our recommendation model.

2.2 Data Preprocessing- In the preprocessing phase, we had cleaned and normalized the raw data to get it ready for analysis. The majority of the columns in the dataset, such as genres, cast, crew, and keywords, were of JSON type originally. We converted them to Python lists for ease. Then after this conversion, we got such key information like the names of the three prime actors, the movie director, and important keywords.

To make it easier to analyze and improve the model's understanding and performance, we also merged these features we extracted using the summary and tagline of the movie into one, which we named "tags."[3] This single text field offered a comprehensive and succinct description of each movie, which made it easy for later text-based feature extractions.

2.3 Feature Extraction- Having established the "tags" field through combining keywords, genres, cast, crew, and movie descriptions, we needed to convert this text information into a numeric format that can be handled by machine learning algorithms. For this reason, we made use of two well-known techniques: CountVectorizer and TF-IDF Vectorizer. CountVectorizer is the one that determines how many times each word has appeared in the data, and TF-IDF (Term Frequency-Inverse Document Frequency) assigns weight to words depending upon their relative prominence relative to other movies. Preprocessing of text was done alike before applying these vectorizers. This meant that all of the words were changed to lower case to keep things consistent, stop words and special characters were stripped away to remove noise and filter out, and stemming was done by means of the Porter Stemmer in an attempt to reduce words down to their root word (for example, "running" down to "run"). All of these preprocessing methods enabled us to create powerful numerical representations of material for every movie to enable easy comparison and computational analysis.

2.4 Similarity Calculation- Once we had transformed each movies data into a numerical vector form, we used cosine similarity to compare the vectors. This method allows us to see how much alike two films are by calculating the angle between their respective vectors. The smaller the angle—or the closer the cosine value is to 1—the more similar the movies are in content. This enabled the system to correctly identify and recommend movies with similar related themes, genres, or descriptions, based solely on their textual attributes.

2.5 Recommendation System- After calculating the similarity scores among movies based on their content vectors, we created the basic logic of our recommendation system. When a user selects a movie, the system retrieves the respective numerical vector of the movie first. Then, it compares the vector with the vectors of every other movie based on cosine similarity to determine the degree to which they are relevant to each other in content. The similarity results are sorted in descending order, displaying the most prominent matches. Finally, the system shows the top 5 most similar movies. This method will ensure that users receive instant and meaningful movie recommendations that are very similar to their selected title.

2.6 Evaluation- Since our recommendation system focuses purely on the content of movies and does not consider user interactions or ratings, we did not apply typical evaluation metrics such as precision, recall, F1-score, Hit Rate (HR), or Average Reciprocal Hit Rank (ARHR). Instead, we evaluated the performance of the suggestions quantitatively by observing whether the suggested movies showed high levels of similarity with the input movie particularly in genre, plot, and overall theme. The qualitative approach provided insightful observation on the suitability and coherence of the suggested movies, confirming their relevance with the suggested content-based filtering technique.

3. MODELING AND ANALYSIS

This section presents the important part occurring in the content-based movie recommendation system related to the modeling and analysis.

3.1 Modeling Approach

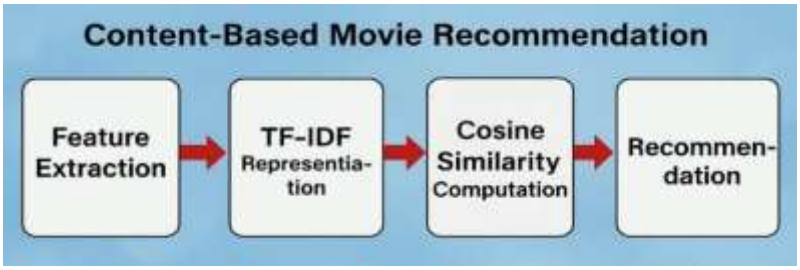

Figure 1: Modeling Procedure.

www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)
(Int Peer Reviewed Journal)
Vol. 05, Issue 04, April 2025, pp : 2024-2030

e-ISSN : 2583-1062
Impact Factor : 7.001

3.1.1 Data Representation and Feature Extraction

In the content-based movie recommendation system, we used a content-based filtering system that provides recommendations based on their contents, such as features. In this content-based movie recommendation system, we used the TMDB movie dataset from the official Kaggle platform. The TMDB movies dataset includes various features and attributes such as genre, keywords, cast, crew, movie_id, etc. These features and attributes are used for feature extraction. Next, these features are transformed into a numerical representation, and a content-based movie recommendation system efficiently calculates the similarity between movies.

3.1.1.1 Features Considered

Table 1. Movie Features

Feature	Description
Title	Movie title (e.g., Inception)
Genres	Action, Sci-Fi, Drama, etc.
Keywords	Important words from descriptions (e.g., “dream”, “heist”)
Cast and Crew	Directors, actors, and screenplay writers
Overview	Summary of the movie’s storyline
User Tags	Descriptive tags assigned by users

3.1.1.2 Feature Engineering for Text Processing

In the content-based movie recommendation system, we used the TMDB movie dataset. Dataset has the movie features that are most textual. To transform these textual features into numerical form, we used various natural language processing techniques are used. We have mentioned the below natural language processing techniques.

1. **Tokenization -** Breaking down text into individual words (e.g., "A detective solves a thrilling murder case." after tokenization ["A", "detective", "solves", "a", "thrilling", "murder", "case"]).
2. **Stopwords Removal -** Removing frequent words that don't contribute much meaning (e.g., "the," "a," "is").
3. **Stemming/Lemmatization** - Reducing words to their base form (e.g., "running" → "run"). This enhances text analysis by converting words to their root form so that matching movie content descriptions and user preferences can be done more effectively (e.g., "A detective is running after a thief." After stemming/lemmatization result is ['detect', 'run', 'thief']).
4. **Vectorization -** Use of vectorization to converting the text into numerical form using TF-IDF (Term Frequency-Inverse Document Frequency).

3.1.2 TF-IDF for Feature Representation

TF-IDF (Term Frequency-Inverse Document Frequency) is a statistical approach used in content-based movie recommendation

System. This is an important in text processing and TF-IDF is widely used in a content-based movie recommendation system for feature extraction. We have mentioned below the benefits of TF-IDF:

- Removes the common words that occur in movie descriptions, such as “the” or “and”, because not required this words and it’s considering the lower importance in movie descriptions and these type common words not help to find the what is a making a movie is unique.
- Gives the importance to rare words. e.g., “thriller” or “detective”, these types of words are not used in every movie description, but so when they appear these words stand out as an important and unique in movie dataset description. TF-IDF computes the importance of word such as rare word, which is help to the content-based recommendation system are understand the which movie is a similar to the other movie with similar matched content.

3.1.2.1 TF-IDF Formula

TF-IDF = TF * IDF

where:

- Term Frequency (TF) means shows the how many times appear the single word in movie description or document.
- Inverse Document Frequency (IDF) calculates the how rare or unique word is across all movie description or document.

www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)
(Int Peer Reviewed Journal)
Vol. 05, Issue 04, April 2025, pp : 2024-2030

e-ISSN : 2583-1062
Impact Factor : 7.001

3.1.2.2 Example

Movie	Description	TF-IDF Processed Features
Movie A	"A detective solves a murder case"	{detective: 0.4, murder: 0.6, case: 0.3}
Movie B	"A thrilling mystery involving a detective"	{detective: 0.5, thrilling: 0.4, mystery: 0.6}

Consider these two movies:

By using TF-IDF vectorization calculates the words importance like such as "detective," "mystery," and "murder" give the high importance, which helps the system to find similar movies efficiently.

3.1.3 Similarity Calculation using Cosine Similarity

Cosine similarity is used to calculate the similarity between movies in the content-based movie recommendation system, calculates the similarity between movies after TF-IDF vectorization.

- cosine similarity calculates similarity between movies by checking the two movie descriptions are how to close by measuring the angle between their TF-IDF numerical vectors, there helps the system understand the which movies are same on to their contents, similarity not the based on just word counting in TF-IDF.
- Works Well on Sparse Data and It can operate on large group movie descriptions quickly, even if most words do not appear in the descriptions. That is, this makes it useful for processing text data where in each movie overview, only a few words matter.

3.1.3.1 Cosine Similarity Formula

$$\text{Cosine\_Similarity (A, B)} = \frac{A \,.\, B}{||A|| * ||B||}$$

where:

- A.B = calculates the how much description match in movies.
- ||A|| and ||B|| = shows the description length of movies.

3.1.3.2 Example

Consider the example of two movie descriptions, and calculate the similarity between movies by using their TF-IDF vectors.

Movie	Vector
Movie A	(0.4, 0.6, 0.3)
Movie B	(0.5, 0.4, 0.6)

Calculate the Cosine Similarity:

$$\text{Cosine\_Similarity} = \frac{(0.4 \times 0.5) + (0.6 \times 0.4) + (0.3 \times 0.6)}{\sqrt{(0.4^2 + 0.6^2 + 0.3^2)} \times \sqrt{(0.5^2 + 0.4^2 + 0.6^2)}}$$

If the result is 0.89 after compute the cosine similarity, it means movie A and Movie B are closely to similar to based on there contents. It means 89% similarity occurs in movie A and movie B.

3.1.4 Content-Based Movie Recommendation Workflow

Steps in System Workflow:

1. User select the movie in movies list and provide the selected movie as a input to the system for recommendation. Example: Inception
2. System converts the movie textual features into numerical vectors using TF-IDF for getting the recommendation for inception movie.
3. Calculate the cosine similarity with all others movies in dataset using TF-IDF numerical vectors for recommendation to the inception movie.
4. Recommendation system sorts the movies in dataset by similarity score using cosine similarity for recommendation to the inception movie.
5. Content-Based Movie Recommendation System recommend the top 5 recommendation to the inception movie using TMDB API.

www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)
(Int Peer Reviewed Journal)
Vol. 05, Issue 04, April 2025, pp : 2024-2030

e-ISSN : 2583-1062
Impact Factor : 7.001

3.1.5 Recommendation System Architecture

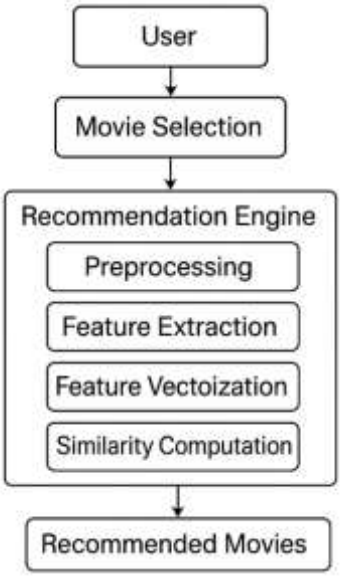

Figure 2: Recommendation System Architecture

3.2 Analysis

3.2.1 Content-Based Movie Recommendation System Analysis

In this project task, we consider the "TMDB 5000 Movie Dataset" on Kaggle. The database contains two CSV files - “tmdb_5000_movies.csv” and “tmdb_5000_credits.csv.” “tmdb_5000_movies.csv” has diverse attributes that have useful information related to the movies: "Budget": It is the budget of each film. "Genres": It is the sub genres of the film, for example, Action, Documentary, etc.

A film can belong to various genres. "Homepage": It is the webpage link of the movie. "ID": It is the individual identifier of every movie. "Keywords": Contains the key words of the movie land a brief description of the film. "Original Language": Whether theism was shot originally in English or any other language. "Original Title": Original title of the film. "Overview": Brief description of the film. "Popularity":

A measure of the popularity of the film. "Production Companies": Names of the companies which produced the film. "Production Countries": Names of the countries where the film was produced. "Release Date": Date the movie got released in the format of yyyy-mm-dd. "Revenue": Pertains to the revenue of the movie. "Runtime": Indicates the duration of the film in minutes. "Spoken Languages": Enumerates the languages employed in the movie. "Status": Is the status of the film, whether it is released or not. "Tagline": Stores the tagline of the film. "Title": Title of the film. "Vote Average": Shows the average vote provided users. "Vote Count": Is the count of votes received. For doing exploratory data analysis (EDA), we first load the dataset with pandas into a Data Frame called 'movies'. We also have another Data Frame “credits” containing all metadata regarding the movies.

Also, we perform data preprocessing retrieve specific data, i.e., converting the 'cast' and 'genres' features into more easy-to-handle formats. We use a self-created function to retrieve director names from the 'crew' feature. This helps to get better information regarding the movie's crew members. Last but not least, we use EDA techniques like statistical analysis and data visualization to extract useful insights into the dataset. Looking into relationships between variables, seeing trends, and keeping track of distributions will give us a good idea the movie data so that we can make informed choices for our content-based movie recommendation system.

3.2.2 Comparative Analysis

Table 2. Comparative Analysis

Method	Advantages	Disadvantages
Content-Based Filtering (our Model)	Personalized, No user history needed	Limited diversity, Cold-start problem for new movies
Collaborative Filtering	Learns from user preferences	Cold-start issue, Needs large data
Hybrid Model (Content + Collaborative)	Best of both worlds	Computationally expensive

www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)
(Int Peer Reviewed Journal)
Vol. 05, Issue 04, April 2025, pp : 2024-2030

e-ISSN : 2583-1062
Impact Factor : 7.001

4. RESULTS AND DISCUSSION

4.1 Results

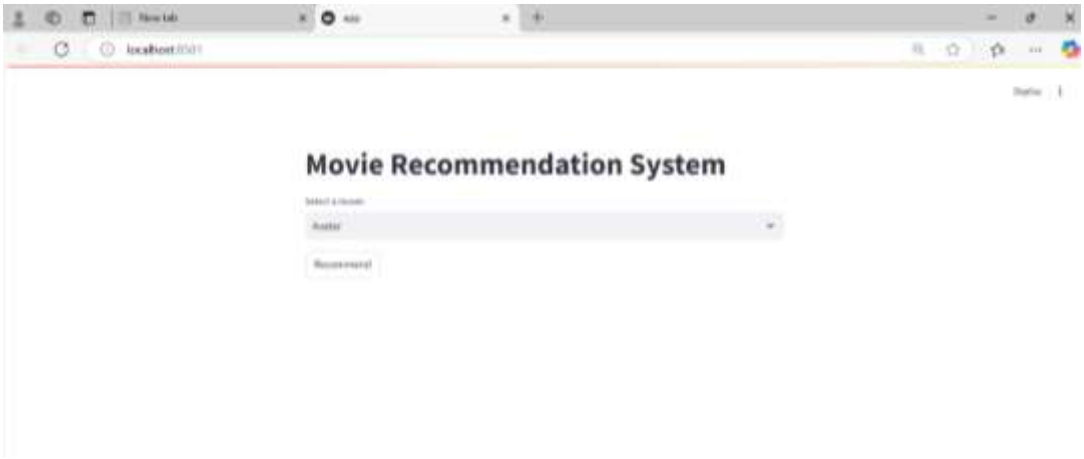

Figure 3: User Interface Screen

- This is a user interface for web app developed via the Python Streamlit library. development of a user interface via the Streamlit library for content-based movie recommender system with minimizing time complexity & space complexity in programming logic.

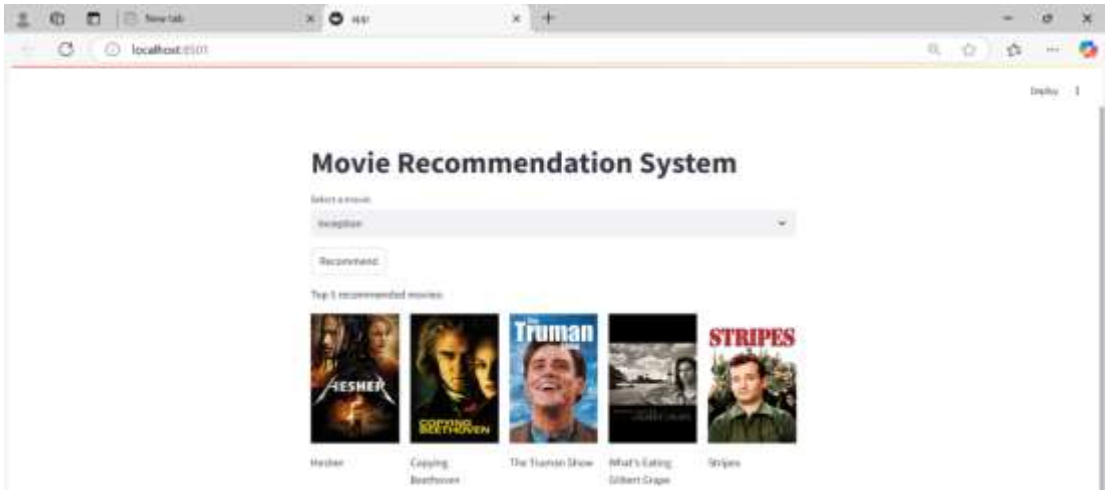

Figure 4: Recommendations For Movie

- When the user provides input to the content-based recommendation system for getting the top 5 recommendations, then the system suggests the top 5 recommendations according to user input. The system suggests the recommendations based on movie features such as cast, crew, keywords, genre, and overview.
- Input Movie: **Inception**
- Recommended Movies*:* Hesher, Copying Beethoven, The Truman Show, What's Eating Gilbert Grape, Stripes
- These results indicate that the system successfully retrieves movies sharing similar themes such as science fiction, intellectual plot, and psychological thrillers. This behavior ensures that the model can successfully recover meaningful semantic content from the features of the content.

4.2 Discussion

Our content-based strategy is very effective in suggesting movies of the same theme, genre, and plot structure even without using user rating or interaction data. It offers several benefits:

- Personalization: Offers personalized suggestions based on the content according to the user's interest in movies.
- Cold Start Advantage for Users: Unlike collaborative filtering, it can also recommend to new users since it does not require relying on previous interactions.
- Fast and Efficient: Fast response time in generating proper recommendations.
- No Dependency on Ratings: Not dependent on user-supplied ratings, hence being immune to biased or incomplete feedback.
- While there are some minor constraints:

- It might not provide very diverse or novel movie genres beyond a user's usual interest.
- It needs enough metadata (keywords, cast, description) for proper prediction.

In general, the system provides highly individualized and content-rich movie recommendations. It can be a solid basis for further extensions like hybrid filtering, user feedback adaptation, or deep learning integration, making it a real-world and scalable recommendation solution.

5. CONCLUSION

The research succeeded in constructing a content-based movie recommendation model with emphasis on intrinsic features like genre, plot, keywords, and cast. The synergy between natural language processing techniques and vectorization models facilitated effective assessment of content similarity. The system was demonstrated to produce more precise and relevant recommendations than mere filtering techniques, especially suited for users with little or no interaction history. Through exclusive attention to film characteristics and not user taste, the solution eliminates the cold-start problem and guarantees meaningful suggestions. This early work proves the strength of content-based systems and prompts further research into hybrid models merging content, user behaviour, and multimedia data to achieve a better and more personalized recommending experience.

ACKNOWLEDGEMENTS

We are thankful to Ms. Charushila Patil, Department Head of Artificial Intelligence and Data Science, Guru Gobind Singh College of Engineering and Research Centre, Nashik, for her eternal inspiration, encouragement, and valuable suggestions throughout the journey of this research work. We are also sincerely grateful to Mr. Nilesh Sonawane and Ms. Tanvi Deshmukh for guiding us, providing helpful suggestions, and for their continuous motivation that greatly helped us to successfully complete this paper. Their mentorship has been instrumental in deciding the course and direction of our research.

6. REFERENCES

[1] H. J. P. J. Permana and A. T. Wibowo, “Movie Recommendation System Based on Synopsis Using Content-Based Filtering with TF-IDF and Cosine Similarity,” International Journal on Information and Communication Technology (IJoICT), vol. 9, no. 2, pp. 1–14, Dec. 2023.

[2] N. K. K. R. Pradeep, K. K. Rao Mangalore, B. Rajpal, N. Prasad, and R. Shastri, "Content based movie recommendation system," *International Journal of Research in Industrial Engineering*, vol. 9, no. 4, pp. 337–348, Dec. 2020.

[3] M. Goyani and N. Chaurasiya, "A review of movie recommendation system: Limitations, survey and challenges," *ELCVIA. Electron. Lett. Comput. Vis. Image Anal. *, vol. 19, no. 3, pp. 18–37, 2020.

[4] U. Javed, K. Shaukat, I. A. Hameed, F. Iqbal, T. M. Alam, and S. Luo, "A review of content-based and context-based recommendation systems," *Int. J. Emerg. Technol. Learn. *, vol. 16, no. 3, pp. 274–306, Feb. 2021.

[5] S. Rakesh, “Movie Recommendation System Using Content Based Filtering,” Al-Bahir Journal for Engineering and Pure Sciences, vol. 4, no. 1, Article 7, 2024. [Online]. Available: https://doi.org/10.55810/2313-0083.1043

[6] S. Sahu, R. Kumar, M. S. Pathan, J. Shafi, Y. Kumar, and M. F. Ijaz, “Movie popularity and target audience prediction using the content-based recommender system,” IEEE Access, vol. 10, pp. 42044–42060, Apr. 2022.

[7] Abhinav N., Sujatha K., "Content-based movie recommendation system using cosine similarity measure," AIP Conference Proceedings, vol. 2901, no. 1, pp. 1–6, Dec. 2023. doi:10.1063/5.0154761.