Movie Recommender System using Content-Based Filtering Technique

Project submitted to the

SRM University – AP, Andhra Pradesh

for the partial fulfillment of the requirements to award the degree of

Bachelor of Technology/Master of Technology

In

Computer Science and Engineering

School of Engineering and Sciences

Submitted by

Ankit Rajput	**AP21110010918**
Chakrapani Challa	**AP21110010920**
Ashish Kumar Gupta	**AP21110010938**
Rahul Vijay Singh	**AP21110010960**
Raj Choudhary	**AP21110010974**

SRM University–AP

Neerukonda, Mangalagiri, Guntur

Table of Contents

Abstract

This project aims to develop a content-based movie recommender system using movie descriptions, taglines, metadata, cast, crew, genres, and keywords. The recommender system utilizes Bag of Words Bag of Words TF-IDF vectorization and cosine similarity to measure the similarity between movies based on their textual representations. The report provides a comprehensive overview of the concepts and processes involved in building the movie recommender system.

The report discusses the data collection and preprocessing steps, including data cleaning and feature extraction from the movie dataset. It explains the techniques used for text preprocessing, such as text cleaning, tokenization, and stemming. The Bag of Words Bag of Words TF-IDF vectorization process and cosine similarity calculation are detailed, showcasing how textual data is transformed into numerical representations and used to measure movie similarity. The algorithms for generating movie recommendations based on both movie descriptions and metadata are presented. The report also includes an evaluation of the recommender system's performance and discusses potential future improvements, such as feature engineering and incorporating hybrid approaches. Overall, it serves as a valuable resource for understanding the development of a content-based movie recommender system.

Introduction

1.1 Background:

With the proliferation of online platforms and the availability of vast amounts of data, recommendation systems have become essential for personalized user experiences. Movie recommendation systems play a crucial role in helping users discover new movies that align with their preferences. Content-based filtering is a popular approach in building such systems, as it focuses on analyzing the inherent characteristics of movies to identify similarities and provide tailored recommendations. By considering movie descriptions, taglines, metadata, cast, crew, genres, and keywords, content-based recommender systems can offer personalized suggestions to users.

1.2 Problem Statement:

The problem addressed in this project is the need to develop a content-based movie recommender system that overcomes the limitations of traditional rating-based systems. While user ratings can provide valuable insights, they may not capture the complex nature of movie preferences. Content-based filtering aims to address this issue by leveraging the textual information associated with movies to identify similar items. However, effectively processing and utilizing this textual data presents challenges in terms of feature extraction, representation, and similarity measurement.

1.3 Objectives:

The primary objective of this project is to build a content-based movie recommender system that generates personalized recommendations based on user preferences and movie attributes. The specific objectives include:

- Collecting and preprocessing movie data: This involves acquiring a comprehensive movie dataset and performing data cleaning to ensure data integrity. Relevant features, such as movie descriptions, taglines, cast, crew, genres, and keywords, are extracted for further analysis.

- Implementing Bag of Words Bag of Words TF-IDF vectorization and cosine similarity: The textual attributes of movies are transformed into numerical representations using Bag of Words Bag of Words TF-IDF vectorization. Cosine similarity is then calculated to quantify the similarity between movies based on their textual representations.

- Developing recommendation algorithms: The project aims to develop algorithms that leverage the calculated cosine similarity scores to generate movie

recommendations. These algorithms can consider various movie attributes individually or in combination to provide tailored suggestions to users.

- Evaluating the recommender system: The performance of the developed system is evaluated using appropriate evaluation metrics, such as precision, recall, and accuracy. The effectiveness of the recommendations in terms of relevance, coverage, and diversity is analyzed to assess the system's performance.

1.4 Methodology:

The methodology employed in this project encompasses several key steps. Firstly, a diverse and representative movie dataset is collected, ensuring it contains relevant attributes such as movie descriptions, taglines, cast, crew, genres, and keywords. The collected data is then preprocessed, involving cleaning operations to remove noise and inconsistencies. Textual attributes are subjected to further preprocessing steps, including removing stopwords, tokenization, and stemming, to prepare the data for analysis.

To quantify the textual similarities between movies, Bag of Words Bag of Words TF-IDF vectorization is applied to convert the textual data into numerical representations. This transformation enables the calculation of cosine similarity scores, which capture the similarity between movies based on their textual attributes. These scores serve as a foundation for generating recommendations.

Recommendation algorithms are developed that utilize the calculated cosine similarity scores. The algorithms can be designed to consider specific attributes, such as movie descriptions, or a combination of attributes, such as cast, crew, and genres. The algorithms aim to identify movies with high similarity scores to generate personalized recommendations for users.

The developed recommender system is evaluated using appropriate evaluation metrics. The accuracy, precision, recall, and other relevant metrics are calculated to assess the system's performance in providing accurate and relevant movie recommendations. Additionally, factors like coverage and diversity of recommendations are analyzed to ensure a comprehensive user experience.

Data Collection and Preprocessing:

2.1 Dataset Description:

The movie dataset used in this project was sourced from a reputable online database of movies. The dataset comprises a comprehensive collection of movies from various genres and time periods. It includes attributes such as movie titles, release dates, durations, ratings, genres, and textual information like movie descriptions, taglines, cast, crew, and keywords. These attributes provide rich information about each movie, enabling a thorough analysis for recommendation purposes.

Data Set Used: TMDB Dataset

2.2 Data Cleaning:

Data cleaning is a crucial step in ensuring data quality and integrity. The process involved handling missing values and outliers in the dataset. Missing values were addressed by either imputing them using appropriate techniques or removing the corresponding instances, depending on the extent of missingness and the impact on the overall dataset quality. Outliers, which can adversely affect the analysis and recommendation results, were detected and treated using robust statistical techniques or domain-specific knowledge.

To ensure data quality and integrity, various techniques were employed. Duplicate records were identified and removed to avoid redundancy in the dataset. Inconsistent or erroneous data entries were corrected or eliminated after careful validation. Data validation involved cross-checking information against reliable external sources or using predefined criteria. These cleaning techniques helped establish a reliable and accurate dataset for further analysis.

2.3 Recommender Systems:

Recommender systems are intelligent information filtering systems that suggest personalized items or content to users based on their preferences and characteristics. These systems have gained significant popularity and find applications in various domains such as e-commerce, social media, and entertainment. Collaborative filtering is a widely used approach in recommender systems, which leverages user behavior and preferences to make recommendations. Content-based filtering, on the other hand, focuses on the characteristics of items or content itself to provide recommendations. Hybrid approaches combine collaborative and content-based techniques to overcome the limitations of each approach and improve recommendation accuracy.

2.4 Content-Based Filtering:

Content-based filtering is a recommendation technique that utilizes the attributes and features of items or content to make recommendations. In the context of movie recommender systems, content-based filtering leverages movie descriptions, metadata, and other textual features to identify similarities between movies. The advantages of content-based filtering include the ability to provide recommendations even in the absence of user ratings or preferences, as well as the potential for recommending niche or less-known items. By analyzing the textual information associated with movies, content-based filtering can offer recommendations based on specific genres, themes, or plot elements that align with the user's interests.

2.5 Feature Extraction:

The dataset provided a wide range of attributes, and relevant features were extracted to capture different aspects of movies for recommendation purposes.

- Movie Descriptions: Textual descriptions provide valuable information about the plot, themes, and overall content of each movie. These descriptions were extracted and utilized to capture the essence of movies, enabling the system to identify similar movies based on their textual similarity.

- Taglines: Taglines represent concise and catchy phrases that often capture the essence or main selling point of a movie. They were extracted as a separate feature to enhance the understanding of movie characteristics and aid in generating recommendations.

- Cast and Crew: Information about the actors, directors, writers, and other individuals involved in the movie production process was extracted. This information allows the system to consider movies with shared cast and crew members as potentially similar.

- Genres: The genre of a movie plays a significant role in defining its style, themes, and target audience. Genre information was extracted to enable the system to recommend movies from similar genres based on user preferences.

- Keywords: Keywords associated with movies provide additional context and semantic information. They were extracted to capture specific topics, themes, or elements associated with each movie, further enhancing the recommendation process.

By extracting these relevant features, the system gains a comprehensive understanding of each movie's characteristics, allowing for effective content-based filtering and recommendation generation.

Movie Description Based Recommender

3.1 Description and Tagline Preprocessing:

The preprocessing steps for movie descriptions and taglines involve several techniques to clean and prepare the textual data for further analysis. Text cleaning techniques such as removing special characters, punctuation, and numeric digits are applied to eliminate noise and irrelevant information. Tokenization is then performed to break down the text into individual words or tokens. Additionally, techniques like lowercasing the text and removing stop words (commonly occurring words like "the," "is," etc.) are used to reduce the dimensionality of the data. Stemming, which involves reducing words to their base or root form (e.g., "running" to "run"), is applied to further normalize the text and capture the essence of words.

3.2 Bag of Words Bag of Words TF-IDF vectorization:

TF-IDF (Term Frequency-Inverse Document Frequency) vectorization is employed to transform the preprocessed textual data into numerical representations. The process involves calculating the frequency of each word in a document (movie description or tagline) and scaling it by its inverse frequency across the entire corpus. This weighting scheme helps emphasize important words that are distinctive to a particular movie while downplaying common words. TF-IDF vectors are created for each document, representing the importance of each word in that document relative to the entire dataset.

3.3 Cosine Similarity Calculation:

The similarity of movie TF-IDF vectors is measured using cosine similarity. It calculates the cosine of the angle between two vectors, ranging from -1 (completely dissimilar) to 1 (identical). In this case, the TF-IDF vectors represent the textual representations of movie descriptions and taglines. By taking the dot product of two TF-IDF vectors and dividing it by the product of their magnitudes, one may determine the cosine similarity score. There is more resemblance between the textual representations of two movies when the cosine similarity score is higher.

3.4 Recommendations Generation:

To generate movie recommendations, the algorithm utilizes the cosine similarity scores between the TF-IDF vectors of movies. Given an input movie, the algorithm

retrieves the cosine similarity scores for all other movies in the dataset. It then selects the top-n movies with the highest similarity scores as recommendations. These recommendations are considered similar to the input movie based on their textual representations (descriptions and taglines). By providing a list of similar movies, the recommender system offers personalized recommendations to users based on their preferred movie choices and helps them discover new movies with similar characteristics.

System Evaluation and Performance Analysis

4.1 Evaluation Metrics:

The performance of the recommender system is evaluated using various metrics that measure its effectiveness in providing accurate and relevant movie recommendations. Commonly used metrics include precision, recall, F1-score, and mean average precision (MAP). Precision measures the proportion of relevant recommendations among all the recommended movies. Recall calculates the proportion of relevant recommendations retrieved out of all the relevant movies. The F1-score offers a fair assessment of the recommender system's effectiveness since it is the harmonic mean of precision and recall. MAP assesses the quality of the ordered recommendation list by considering the average precision at different cutoff points.

4.2 Experimental Setup:

The recommender system's evaluation involves splitting the movie dataset into training and test sets. The dataset is typically divided randomly, ensuring that the distribution of movies and user preferences is maintained in both sets. The training set is used to train the recommender model, while the test set is used to evaluate its performance. The performance metrics are calculated based on the recommendations provided by the system and the known preferences of users in the test set. The experimental setup may also involve cross-validation techniques to obtain more reliable performance estimates.

4.3 Results and Analysis:

The results of the evaluation provide insights into the performance of the recommender system. The performance metrics, such as precision, recall, F1-score, and MAP, are calculated and analyzed to determine the system's effectiveness in

generating accurate and relevant recommendations. The analysis may involve comparing different algorithms or variations of the recommender system to identify the most successful approach. Additionally, the strengths and limitations of the system are discussed. The strengths may include high precision and recall values, indicating accurate and comprehensive recommendations. However, limitations may arise due to sparse data, cold-start problems, or biases in the dataset, which can affect the system's performance and the diversity of recommendations. Understanding these strengths and limitations helps in further improving the recommender system and addressing potential challenges.

Output

	adult	belongs_to_collection	budget	genres	homepage	id	imdb_id	original_language	original_title	overview	...	release_date
0	False	{'id': 10194, 'name': 'Toy Story Collection', ...	30000000	[{'id': 16, 'name': 'Animation'}, {'id': 35, '...	http://toystory.disney.com/toy-story	862	tt0114709	en	Toy Story	Led by Woody, Andy's toys live happily in his ...	...	1995-10-30
1	False	NaN	65000000	[{'id': 12, 'name': 'Adventure'}, {'id': 14, '...	NaN	8844	tt0113497	en	Jumanji	When siblings Judy and Peter discover an encha...	...	1995-12-15
2	False	{'id': 119050, 'name': 'Grumpy Old Men Collect...	0	[{'id': 10749, 'name': 'Romance'}, {'id': 35, ...	NaN	15602	tt0113228	en	Grumpier Old Men	A family wedding reignites the ancient feud be...	...	1995-12-22
3	False	NaN	16000000	[{'id': 35, 'name': 'Comedy'}, {'id': 18,	NaN	31357	tt0114885	en	Waiting to Exhale	Cheated on, mistreated and stepped	...	1995-12-22

```
1 smd = md[md['id'].isin(links_small)]
2 smd.shape
```

```
(9099, 25)
```

Cosine Similarity

Cosine Similarity is used to calculate a numeric quantity that denotes the similarity between two movies. Mathematically, it is defined as follows:

$$cosine(x, y) = \frac{x.y^\intercal}{||x||.||y||}$$

Since we have used the TF-IDF Vectorizer, calculating the Dot Product will directly give us the Cosine Similarity Score.

```
[ ]  1 cosine_sim = linear_kernel(tfidf_matrix, tfidf_matrix)
```

```
[ ]  1 print(cosine_sim)
```

```
[[1.         0.00680476 0.         ... 0.         0.00344913 0.        ]
 [0.00680476 1.         0.01531062 ... 0.00357057 0.00762326 0.        ]
 [0.         0.01531062 1.         ... 0.         0.00286535 0.00472155]
 ...
 [0.         0.00357057 0.         ... 1.         0.07811616 0.        ]
 [0.00344913 0.00762326 0.00286535 ... 0.07811616 1.         0.        ]
 [0.         0.         0.00472155 ... 0.         0.         1.        ]]
```

```
[ ]  1 get_recommendations('The Godfather').head(10)
```

```
973      The Godfather: Part II
8387                 The Family
3509                       Made
4196         Johnny Dangerously
29               Shanghai Triad
5667                        Fury
2412             American Movie
1582    The Godfather: Part III
4221                  8 Women
2159              Summer of Sam
Name: title, dtype: object
```

```
  1 get_recommendations('The Dark Knight').head(10)
```

```
7931                       The Dark Knight Rises
132                              Batman Forever
1113                             Batman Returns
8227    Batman: The Dark Knight Returns, Part 2
7565                 Batman: Under the Red Hood
524                                      Batman
7901                          Batman: Year One
2579               Batman: Mask of the Phantasm
2696                                        JFK
8165    Batman: The Dark Knight Returns, Part 1
Name: title, dtype: object
```

We will reuse the get_recommendations function that we had written earlier. Since our cosine similarity scores have changed, we expect it to give us different (and probably better) results. Let us check for **The Dark Knight** again and see what recommendations I get this time around.

```
[ ]  1 get_recommendations('The Dark Knight').head(10)
```

```
8031                The Dark Knight Rises
6218                        Batman Begins
7659           Batman: Under the Red Hood
6623                       The Prestige
1134                      Batman Returns
8927             Kidnapping Mr. Heineken
5943                            Thursday
1260                      Batman & Robin
2085                           Following
9024    Batman v Superman: Dawn of Justice
Name: title, dtype: object
```

Conclusion

In conclusion, this literature review has provided an overview of recommender systems, specifically focusing on movie recommender systems. Collaborative filtering and content-based filtering approaches were discussed, along with the concept of hybrid approaches. Content-based filtering, leveraging movie descriptions, metadata, and textual features, emerged as a favorable method. Bag of Words Bag of Words TF-IDF vectorization and cosine similarity were highlighted as key techniques in converting textual data into numerical representations and measuring similarity between movies, respectively. The review of related works emphasized the strengths of movie recommender systems in providing personalized recommendations but also acknowledged limitations such as the cold-start problem and data sparsity. This review serves as a foundation for the development of a content-based movie recommender system that aims to overcome these limitations and deliver engaging recommendations to users.