IJARSCT

ISSN (Online) 2581-9429

International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)

Impact Factor: 6.252

Volume 2, Issue 2, May 2022

The Movie Recommendation System using Content Based Filtering with TF-IDF-Vectorization and Levenshtein Distance

Omkar Kunde[1], Omkar Gaikwad[2], Prathamesh Kelgandre[3], Rohan Damodhar[4], Prof. Mrs. M. M. Swami[5]

UG Students, Department of Computer Engineering[1,2,3,4]

Assistant Professor, Department of Computer Engineering[5]

AISSMS College of Engineering, Pune, Maharashtra, India

Abstract: *In this busy life people like to do things to make their mind calm and watching movies is one of the thing but due to large data of a movie exist in the world it is very difficult for the user to select a movie. They have to spend a lot of time in searching and selecting movie. This procedure is time consuming and difficult. So recommendation system make the things easy. Recommendation engines are trained to produce fast and accurate suggestions to users. This paper describes a movie recommendation system using content based filtering and data is processed using Term-frequency Inverse document frequency technique (TF-IDF) for vectorization. Cosine similarity method is used for similarity measure. The system is presented to the user through a web-hosted user-interface which offers a system architecture by considering the initial problem usually faced by recommendation systems, namely the cold start problem. The problem of lack of user preferences data is trying to be overcome by utilizing movies data. The raw data is processed using the TF-IDF algorithm and Vector Space Model to generate a data model. Then levenshtien distance with cosine similarity will improve the performance of existing system. Advantages of the system include efficient recommendations, correct suggestions. Future enhancements include user profiling, documentations and data acquirement through web scraping.*

Keywords: Movie Recommendation, TF-IDF Vectorization, Content Based Filtering, Vector Space Model, Cosine Similarity, Levenshtein Distance.

I. Introduction

A Movie Recommendation engine recommends movies present in a group of data to the users according to their choices and preferences. Most important and crucial part of an any website is to give correct and appropriate suggestions to users. There are various ways of implementing a movie recommendation engine such as Content based filtering, Collaborative filtering and hybrid system. Limitations of the present day systems include the lack of data, cold start problems, changing data and user choices which results in failure of getting best recommendations for the users. Some examples of highly efficient recommendation systems include the engines used by multinational tech companies such as YouTube, Netflix and Amazon Prime.

The recommendation system technique used, also varies greatly depending on the scope and Item of recommendation. For example, Content-based filtering is used to recommend related movies based on user preferences. In this paper, we further improve the performance of the content-based filtering approach by proposing a new similarity measure between items to solve the sparsity and cold start problems. The movie recommendation system is applied to our approach. In particular features such as titles, genres, directors, actors, and plots of the movies are extracted. Clearly, these features information is more reliable than the user rating, since it is provided by movie experts.

This data is processed using the TF-IDF algorithm and Vector Space Model to generate a data model. Then a query can be applied to find similarities using cosine similarity with the help of Levenshtein distance which measures similarity between two strings. This is the model that enhances the accuracy of the recommendation with TF-IDF Vectorization Algorithm.

II. Literature Survey

Muthurasu, Nandini Rengaraj, K.C. [1] presented a research paper that used TF-IDF and Cosine Similarity in Content based Filtering for Movie recommendation system. Recommendation engines are trained to produce fast and coherent suggestions to users. This paper describes a hybrid video recommendation system using Term-frequency Inverse document frequency technique for vectorization. For similarity measures Cosine similarity method is used. The system is presented to the user through a web-hosted user-interface. Despite small data set, the system gives efficient and correct recommendations.

Sunandana, M.Reshma, Y-Pratyusha, Madhuri Kommineni [2] used content based algorithms TF-IDF vectorization. For movie recommendation system using enhanced content based filtering algorithm. This paper develops a movie recommendation system based on different parameters. The content based algorithm has been employed to recommend movies based on similarity with other films by analyzing the content of movie. To find the similarity cosine similarity method has been used. The cosine similarity has been computed by using linear kernel, where the parameters are taken by the result of TF-IDF vectorization.

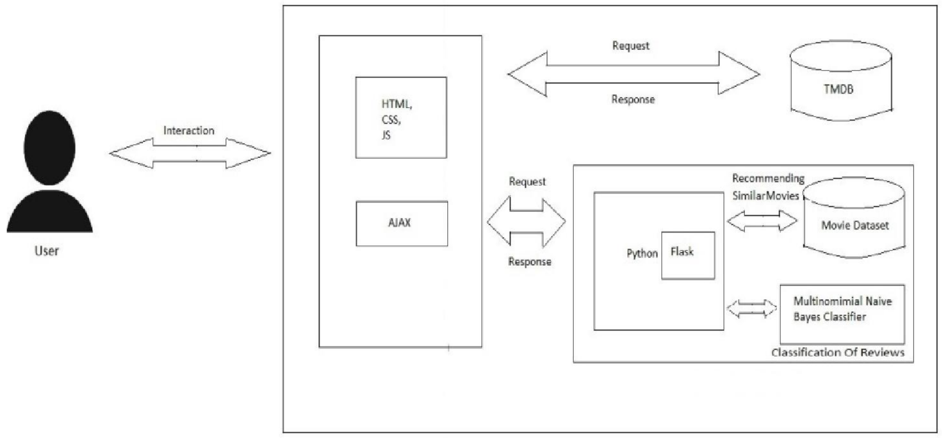
Pradeep, K. K. Rao Mangalore, B. Rajpal, N. Prasad, R. Shastri [3] proposed content-based movie recommendation system using python and machine learning. Content Based recommendation system predicts what movies user will consider based on his previously liked movies .Recommendation systems can recommend movies based on one or a combination of two or more attributes. While designing a movie recommendation system various factors are considered such as the genre of the movie, the director or the actors present in it. In this paper, the recommendation system has been built on cast, keywords, and genres.

Luong Vuong Nguyen, Tri-Hai Nguyen, Jason J. Jung [4] proposed a Content based Collaborative Filtering using Word Embedding. Lower numbers of reviews reduce efficiency and trust of users in collaborative filtering-based recommendation systems. This problem is known as ‘Coldstart’ problem. To solve this problem and improve the efficiency of recommendation systems, a new content-based CF approach-based recommendation system is proposed. We apply the model in the movie domain and extract features such as genres, directors, actors, and plots of the movies. We use the Jaccard coefficient index to covert the extracted features such as genres, directors, actors to the vectors while the plot feature is converted to the semantic vectors. Soft Cosine measure based on vectorized features is used to calculate similarity of the movies. We apply the word embedding model (i.e., Word2Vec) for representing the plots feature as semantic vectors instead of using traditional models such as a binary bag of words and a TF-IDF vector space.

A. Azeem Farhan, Dr. K. Sagar, Smt. T. Suvarna Kumari [5] used Collaborative Filtering with TF-IDF for building movie recommendation system using Collaborative Filtering with TF-IDF.A popular TMDB movie dataset based on collaborative filtering is capable of grasping the interaction or correlation of users and items under considerations. We found out that some key features were being ignored by most of the previous researches. Our work has given significant importance to 'movie overviews' available in the dataset. We experimented with typical statistical methods like TF-IDF, Byusing TF-IDF the dimensions of our courps (overview and other text features) explodes, which creates problem, we have tackled those problems using a dimensionality reduction technique named Singular Value Decomposition (SVD).

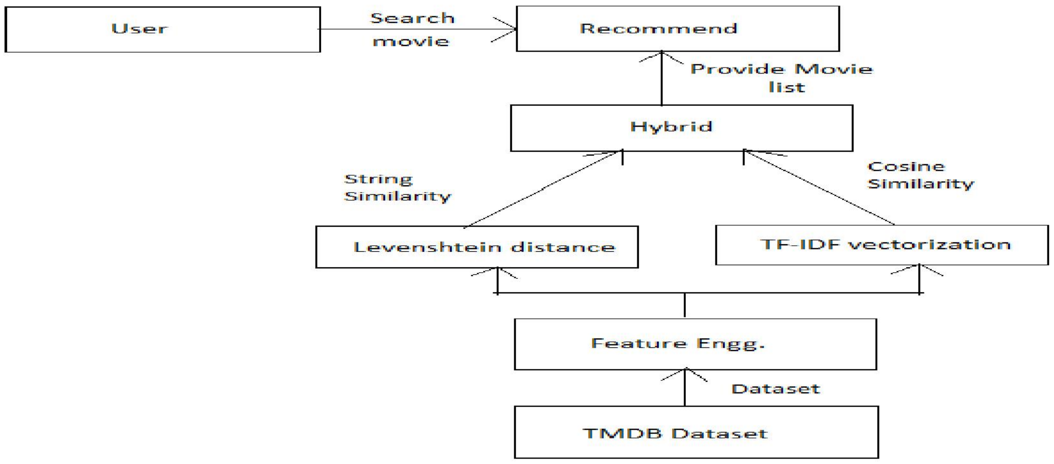
Ashish Pal, Prateek Parhi and Manuj Aggarwal [6] proposed an improved content based collaborative filtering algorithm for movie. The hybrid methodology using both content and collaborative filtering algorithm is used for movie recommendation. The paper incorporates an analysis that justifies this new methodology and how it can provide practical recommendations. The above approach is tested on existing user and objects data and produced improved results when compared with other two favourite methods, Pure Collaborative Filtering, and Singular Value Decomposition.

III. System Architecture



The above picture briefly explains the architecture of the system. Here is a detailed explanation of the same. When the user enters the name of the movie

1. The details regarding the movie are fetched using TMDB API. TMDB is a famous site which has all the detailed information regarding movies. We use their API service and get the required details.
2. User reviews are produced by performing web scraping on the reviews present in the TMDB site. Now these reviews are converted into vectors using TF IDF vectorizer and then are classified as positive and negative based on multinomial naïve bayes-classifier technique.
3. Movies are recommended to the user by comparing the levenshtein distance between the current movie and the movies present in the TMDB dataset. The ones with close levenshtein distance are suggested.
4. Using the Python in backend, flask as a framework, ajax for request and response and HTML, CSS, JS for frontend will give the user-friendly system.

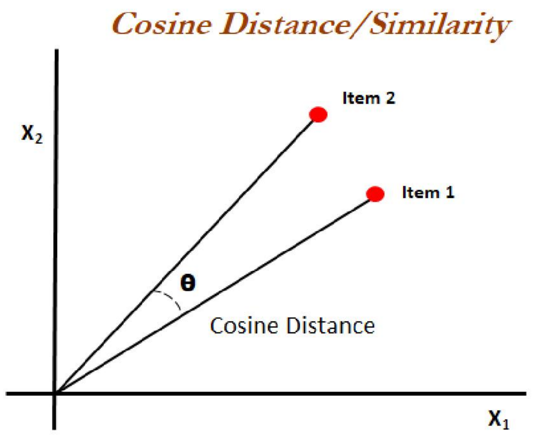


IV. Methodology

The Content-Based Recommender relies on the similarity of the items being recommended. The basic idea is that if you like an item, then you will also like a "similar" item. It generally works well when it's easy to determine the context/properties of each item. If a user is watching a movie, then the system will check about other movies of similar content or the same genre of the movie the user is watching. There are various fundamentals attributes that are used to compute the similarity while checking about similar content. There are various ways to find similarity between the movies.

A. Recommending movies to the users based on the movie they have searched.

- **Similarity Scores:** It is a numerical value ranges between zero to one which helps to determine how much two movies are similar to each other on a scale of zero to one. This similarity score is obtained by measuring the similarity between the text details of both of the movies. So, similarity score is the measure of similarity between given text details of two items. This can be done by cosine-similarity.
- **Cosine Similarity:** Cosine similarity is a metric used to measure how similar the documents are irrespective of their size. Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space. The cosine similarity is advantageous because even if the two similar documents are far apart by the Euclidean distance (due to the size of the document), chances are they may still be oriented closer together. The smaller the angle, higher the cosine similarity.



$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

We have referred above diagram from Movie Recommendation System using Term Frequency-Inverse Document Frequency and Cosine Similarity Method N. Muthurasu, Nandhini Rengaraj, K. C. Mohan Published 2019

- **Levenshtein Distance:** The Levenshtein distance is a string metric for measuring difference between two sequences. Informally, the Levenshtein distance between two words is the minimum number of single-character edits (i.e. insertions, deletions or substitutions) required to change one word into the other.

$$\operatorname{lev}_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j)=0, \\ \min \begin{cases} \operatorname{lev}_{a,b}(i-1,j)+1 \\ \operatorname{lev}_{a,b}(i,j-1)+1 \\ \operatorname{lev}_{a,b}(i-1,j-1)+1_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases}$$

We use a hybrid of both the methods in our application. The primary reason for this to improve the accuracy.

B. Classifying user Sentiments using Natural Language processing

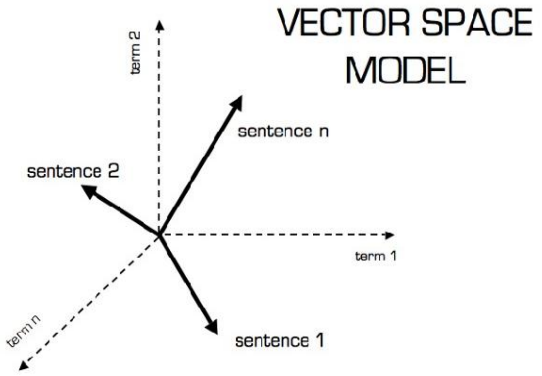
Using concepts of natural language processing to classify the reviews of the user as positive and negative. We convert the words to vectors and use the vector to classify and get the appropriate feature.

There are many ways in which we convert words to vectors. Some of them are bag of words and tf-idf vectorizer. Since bag of words does not preserve the semantic meaning of the meaning. We use TF-IDF Vectorization

V. TF-IDF Vectorization Algorithm

TF-IDF for a word in a document is calculated by multiplying two different metrics: The term frequency of a word in a document. There are several ways of calculating this frequency, with the simplest being a raw count of instances a word appears in a document. Then, there are ways to adjust the frequency, by length of a document, or by the raw frequency of the most frequent word in a document.

The inverse document frequency of the word across a set of documents. This means, how common or rare a word is in the entire document set. The closer it is to 0, the more common a word is. This metric can be calculated by taking the total number of documents, dividing it by the number of documents that contain a word, and calculating the logarithm. So, if the word is very common and appears in many documents, this number will approach 0. Otherwise, it will approach 1



$$\mathrm{tfidf}_{i,j} = \mathrm{tf}_{i,j} \times \log\left(\frac{N}{\mathrm{df}_i}\right)$$

$\mathrm{tf}_{i,j}$ = total number of occurences of i in j

df_i = total number of documents (speeches) containing i

N = total number of documents (speeches)

We have referred above diagram from Content-Based Collaborative Filtering using Word Embedding: A Case Study on Movie Recommendation L. V. Nguyen, Tri-Hai Nguyen, Jason J. Jung Published 13 October 2020 Proceedings of the International Conference on Research in Adaptive and Convergent System

Multiplying these two numbers results in the TF-IDF score of a word in a document. The higher the score, the more relevant that word is in that particular document.

5.1 Naïve Bayes Classifier

Using multinomial naïve bayes classifier to classify the user reviews. Multinomial Naive Bayes algorithm is a probabilistic learning method that is mostly used in Natural Language Processing (NLP). The algorithm is based on the Bayes theorem and predicts the tag of a text such as a piece of email or newspaper article. It calculates the probability of each tag for a given sample and then gives the tag with the highest probability as output.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Technologies and Concepts used

1. Python for backend
2. Flask as framework
3. HTML, CSS, JS for front end
4. Cosine Similarity score and levenshtein distance
5. TF-IDF Vectorization
6. TMDB Dataset
7. Beautiful Soap library for web scrapping
8. Multinomial Naive bayes classifier for classifying user reviews
9. TMDB API to get the data regarding movies

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Impact Factor: 6.252
Volume 2, Issue 2, May 2022

VI. Results

The content-based recommendation is best in situations where there is known data to the item rather than the user as it analyses the attributes of items for generating predictions. Figure 1 shows the homepage of our website. The user can enter a movie name in the given text box and then click the "Enter" button.

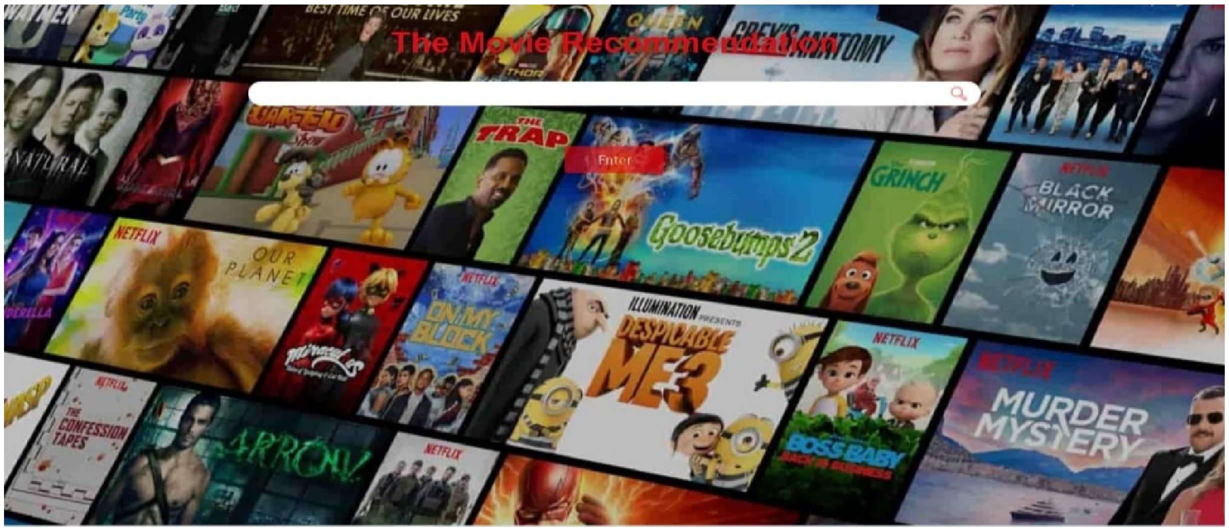


Figure 1: Our proposed movie recommendation system website home page.

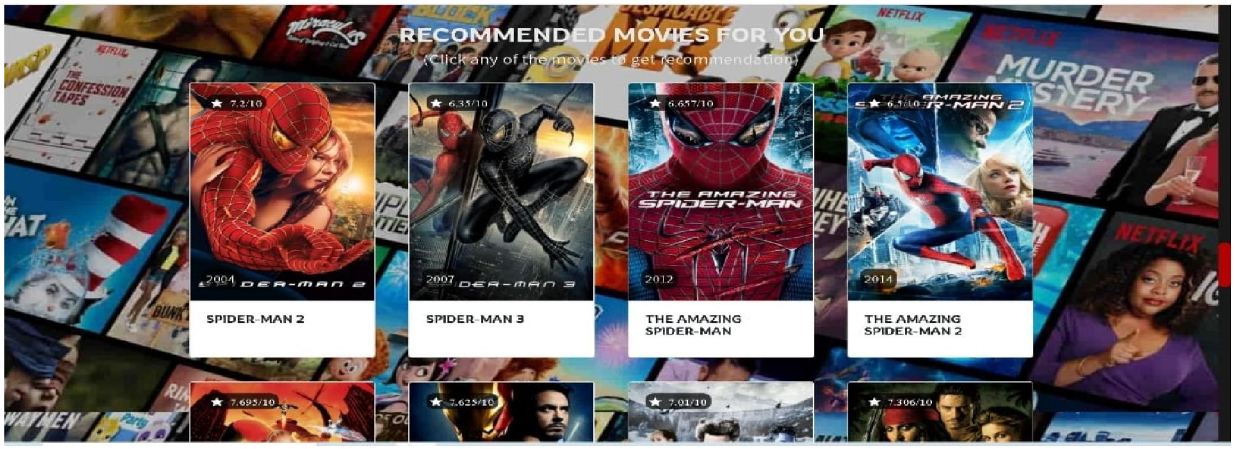


Figure 2: Recommended movies for the movie entered by user

VII. Conclusion

We proposed a new approach that measures the cosine similarity and levenshtien distance which together become a hybrid model which recommends movies based on features extracted such as titles, genres, directors, actors, and plots content. We separate two clear steps to deploy the measurement. The first step is to transform all extracted features from the movies to the vectors. Specially, we applied the Vector Space Model for each plot content of a movie. In the second step, we did the measure for each feature by using Cosine-based Similarity and levenshtien distance then we obtained the set of similar movies.

For content-based recommender system specifically, we attempt to find a new way to improve the accuracy of the representative of the movie. We introduce a new approach for setting weight for these features, the movie can be represented more accurately by TF-IDF algorithm also we have done sentiment analysis with the help of TF-IDF Vectorizer and multinomial naïve-bayes classifier.

So finally we built the website using the flask as framework, html, css, javascript for designing frontend and TMDB api call for retrieving information of the movie. Thus this hybrid system which uses levenshtin distance, cosine similarity and TF-IDF algorithm gives more accurate recommendation to user.

VIII. Acknowledgement

Researchers would like to convey gratefulness to their project mentor, Mrs. M.M Swami Mam and assistant prof. D.M Ujlambkar for her competent assistance and advice throughout the survey and implementation process of this project. Their timely direction, complete co-operation, and minute observation have made our work fruitful.

References

[1]. Movie Recommendation System using Term Frequency-Inverse Document Frequency and Cosine Similarity Method N. Muthurasu, Nandhini Rengaraj, K. C. Mohan Published 2019.

[2]. Movie recommendation system using enhanced content- based filtering algorithm G-Sunandana; M. Reshma ; Y. Pratyush ; Madhuri Kommineni; Published 2021

[3]. Proposed content based movie recommendation system using python and machine learning N. Pradeep , K. K. Raoss Mangalore, B. Rajpal, N. Prasad, R. Shastri

[4]. Content based Collaborative Filtering using Word Embedding:A case Study on Movie Recommendation. Luong Vuong Nguyen, Tri-Hai Nguyen, Jason J. Jung Published 13 October 2020

[5]. Building A Movie Recommendation System Using Collaborative Filtering with TF-IDF. S.A.Azeem Farhan, Dr.K.Sagar, Smt. T.Suvarna Kumari Published on September 2021

[6]. An Improved Content Based Collaborative Filtering Algorithm For Movie Recommendations . Ashish Pal, Prateek Parhi ,Manuj Aggarwal

[7]. Movie Recommendation Using Metadata Based Word2Vec Algorithm Y. Yoon, Jun Woo LeePublished 2018 International Conference on Platform Technology and Service (PlatCon)

[8]. Content-Based Collaborative Filtering using Word Embedding: A Case Study on Movie Recommendation L. V. Nguyen, Tri-Hai Nguyen, Jason J. JungPublished 13 October 2020 Proceedings of the International Conference on Research in Adaptive and Convergent Systems

[9]. C. S. M. Wu, D. Garg, and U. Bhandary, “Movie Recommendation System Using Collaborative Filtering,” In 2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS), pp. 11-15, IEEE, 2018

[10]. H. W. Chen, Y. L. Wu, M. K. Hor, and C. Y. T ang, “Fully content_x0002_based movie recommender system with feature extraction using neural network,” In 2017 International Conference on Machine Learning and Cybernetics (ICMLC) , vol. 2, pp. 504-509, Jul, 2017, IEEE

[11]. Movie Recommendation System Using Bag of Words and Scikit-Learn Sounak Bhattacharya, Ankit Lundia Published 1 October 2019 International Journal of Engineering Applied Sciences and Technology

[12]. R. Van Meteren, and M. Van Someren, “Using content-based filtering for recommendation,” In Proceedings of the Machine Learning in the New Information Age: MLnet/ECML2000Workshop, vol. 30, pp. 47-56, May 2000