Content Recommendation System using TF-IDF based Cosine Similarity

Sanket J Shah, Kishan Mehta, Harsh Patel, Dhruv Sharma

***Abstract*—The amount of media content is increasing exponentially everyday. All the platforms that provide the content have a 'recommended' section where they provide relevant media content that the user might like. In this report we devise a method of finding these relevant 'recommendations' with the help of machine learning. The machine learning approach used here is the similarity approach, where the recommendations are based on the similarity of the features between the movies. The similarity metric used is the 'cosine similarity'. The literature review, implementation and the results for the same are discussed in this report.**

***Index Terms*—Movie recommendation system, machine learning, cosine similarity, tf-idf, content recommendation.**

I. Introduction

WITH the increasing amount of content being made available on internet everyday, any content consumer would naturally feel overwhelmed by it. According to 2020 Global Internet Phenomena Report, Major OTT platforms like Netflix, Hulu, Amazon Prime and others are responsible for 65% of the total Internet traffic [3]. This content overflow has both pros and cons. Even though it allows users to access any content from around the world at anytime, it makes the search for content of user's liking difficult. Hence, there needs to be a system which helps people guide through the endless stream of visual content to find their choice of movie or TV show.

Even though it seems like an easy task, there is more than what meets the eye. There are many factors which need to be considered while developing such system. An user might prefer a movie or TV show because of its cinematography, actors, genre or even music. An ideal recommendation system needs to recognize the pattern in user's history and learn from it. It needs to understand what the user prefers and suggest future content based on it. Therefore, the most suitable solution would be to use a machine learning model that can learn from user's past history. Hence, this report talks about the various machine learning models that can be used for the same.

II. Literature Survey

A. Cosine similarity

Cosine similarity is also known as the cosine co-efficient. It is a similarity metric that uses the angle between two vectors to find the similarity between them. Consider vectors v1 and v2 that have an angle of θ between them as shown in the figure below. The cosine similarity between them would be $cos(\theta)$. [4]

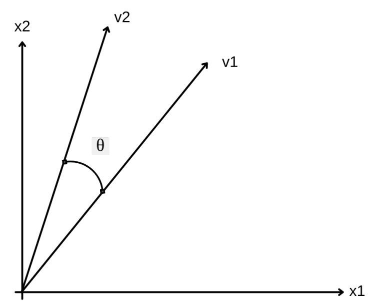

Fig. 1: Cosine similarity

For two matrices, it is defined as follows:

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

B. TF-IDF Vectorizer

TF-IDF also known as Term Frequency-Inverse Document Frequency is a method used to retrieve information from a given text. It consists of two terms.
1. Term-Frequency is a ratio of given word's frequency out of total distinct words.
2. Inverse Document Frequency is a logarithmic ratio of total number of texts and texts with given term in it.
The TF-IDF value is generated by multiplying both of these terms. [2]

$$TF_{i,j} = \frac{n_{i,j}}{\sum_k n_{i,j}}$$

(a) TF

$$IDF\ (wt) = log(\frac{N}{df_i})$$

(b) IDF

$$TF - IDF(wt) = TF_{i,j} \times IDF(wt)$$

(c) TF-IDF

Fig. 2: TF-IDF Formula

III. Implementation

A. Cosine similarity

For implementation, the dataset that we used is the 'MovieLens 25M Dataset' [1]. The approach to implementation was to use the different features of the movies and find the similarity between them. The following combination of the features were used to get the results:
1. movieTagline + movieOverview
2. movieGenre + movieDirector
3. Movie MetaData (movieDirector + movieCast(Top 3 names from the data) + movieGenre + movieKeywords)

The 'term frequency–inverse document frequency(TF-IDF) vectorizer' is used to convert the string input to numeric input so that the calculation becomes efficient. The TF-IDF vectorizer is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.

The following are the steps for implementation of 1. movieTagline + movieOverview:
1. Fetch the movieTagline and the movieOverview columns from the given dataset.
2. Combine the above columns to make a new 'movieDescription' column.
3. Use the TF-IDF vectorizer to convert string to numeric data.
4. Use the cosine similarity to get the pairwise similarity matrix for all the movies.
5. Extract the top n movies with the highest similarity index with respect to the input movie.
6. Evaluate the output qualitatively.

IV. Results

A. Cosine similarity

The results obtained are evaluated qualitatively.

1) movieTagline + movieOverview

```
1 get_recommendations('The Dark Knight').head(20)

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
6144                                Batman Begins
7933              Sherlock Holmes: A Game of Shadows
5511                              To End All Wars
4489                                        Q & A
7344                           Law Abiding Citizen
7242                    The File on Thelma Jordon
3537                                 Criminal Law
2893                                Flying Tigers
1135                        Night Falls on Manhattan
8680                            The Young Savages
```

Fig. 3: Top 20 similar movies

The above figure shows the results for the input movie 'The Dark Knight'(A batman movie). The results recommend all the other batman movies at the top which are naturally most similar to the input. Therefore, qualitatively we can observe that using the movieTagline and movieOverview as the features for similarity, we can get high quality results.

```
1 get_recommendations('Spider-Man').head(10)

6676              Spider-Man 3
8531    The Amazing Spider-Man 2
8066      The Amazing Spider-Man
5476              Spider-Man 2
5219        Bang Bang You're Dead
4087               The New Guy
7488                  Kick-Ass
1563    Gremlins 2: The New Batch
3057            Mad About Mambo
6432                 Clerks II
Name: title, dtype: object
```

Fig. 4: Top 10 similar movies

Similar results are also observed for the input movie 'Spider-Man'.

2) movieGenre + movieCast

The features used here are: Genre keywords(all the different genres that apply to the movie), cast(the names of the whole cast).

```
1 get_recommendations2('Spider-Man').head(10)

5103                                        Hidalgo
1069                             The Amityville Curse
7353         Cirque du Freak: The Vampire's Assistant
2680                              Defending Your Life
7401                                       Pontypool
6695    The Librarian: Return to King Solomon's Mines
1262                                  Picture Perfect
8851                               The Age of Adaline
4338                                         Scanners
3270                                     Cry Freedom
Name: title, dtype: object

[182]  1 get_recommendations2('The Dark Knight').head(10)

3108                               The Way of the Gun
4329                                          Fingers
1010                                                M
8026    Across the Line: The Exodus of Charlie Wright
7274                    Gangster's Paradise: Jerusalema
6797                                      Elite Squad
7757                                          Win Win
8401                                    Machete Kills
6773                                    Lust, Caution
1822                                         Holy Man
Name: title, dtype: object
```

Fig. 5: Top 10 similar movies

The results obtained are very low quality and not similar at all. The reason is that using the list of the whole cast disturbs the generalization and the values obtained in the similarity matrix are also very low.

3) Movie MetaData

Using the MetaData(Director, Genre, Cast, Keywords), the results obtained are more generalized and practical. Here, only the top 3 names from the cast are used so that the results are not bad. The results obtained are not only the sequels of the input movies, but also the movies that are directed by the same director or has the similar cast are also obtained.

In the Fig.6 and Fig.7 the recommendations obtained are a mix of the sequel movies, movies from the same director and movies with similar cast as well.

For the input movie "The Dark Knight", we get the "The Dark Knight Rises" which is the sequel to the input movie. We also get "The Prestige" and "Inception" which are not related to Batman but share the same director "Christopher Nolan". All these movies also share many cast members like "Michael Caine".

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

8031         The Dark Knight Rises
6218                 Batman Begins
6623                  The Prestige
2085                     Following
7648                     Inception
4145                      Insomnia
3381                       Memento
8613                  Interstellar
7659     Batman: Under the Red Hood
1134                Batman Returns
Name: title, dtype: object
```

Fig. 6: Top 10 similar movies

```
1 get_recommendations('Spider-Man').head(10)

6757                  Spider-Man 3
5538                  Spider-Man 2
271          The Quick and the Dead
8370     Oz: The Great and Powerful
7306              Drag Me to Hell
3237                      The Gift
2307          For Love of the Game
1032                 Evil Dead II
1918                A Simple Plan
4947                      Darkman
Name: title, dtype: object
```

Fig. 7: Top 10 similar movies

V. Conclusions

The cosine similarity approach is an effective solution to the recommendation problem. The results obtained after using the metadata of the input movie, the final list obtained is very practical and usable. For real-life application, we can use use the output from multiple input movies that the user has previously watched and make combined recommendations.

References

[1] Movielens 25m dataset, 2021.
[2] Anand Rajaraman and Jeffrey David Ullman. *Data Mining*, page 1–17. Cambridge University Press, 2011.
[3] Sandvine. Global internet phenomena.
[4] Ramni Singh, Sargam Maurya, Tanisha Tripathi, Tushar Narula, and Gaurav Srivastav. Movie recommendation system using cosine similarity and knn. pages 2249–8958, 06 2020.