International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Movie Recommendation System using Free Text

Yeole Madhavi B.[a], *Rokade Monika D.*[b], *Khatal Sunil S.*[c]

[a]*SPCOE, Dumbarwadi (Otur), Pune, Maharshtra, India*
[b]*SPCOE, Dumbarwadi (Otur), Pune, Maharshtra, India*
[c]*SPCOE, Dumbarwadi (Otur), Pune, Maharshtra, India*

A B S T R A C T

There is already enough content available on the movie recommendation system. Showing the movie recommendations is essential so that the user need not waste a lot of time searching for the content which he/she might like. Thus, movie recommendation system plays a vital role to get user personalized movie recommendations.In this paper, we have proposed a procedure of recommending movies on the basis of free text entered by the user. Generally, the movie recommendation systems take into consideration the user history, i.e., the movies watched by the user earlier and the ratings given by the user to those movies. The approach followed in this paper is a bit different. We have not taken into consideration the recently watched movie or the user history. Instead, we have taken into consideration the user input. Just like the way we google out something on the internet, we can write a text query which states that the requirement of the user. The text query or the free text is further processed using Natural Language Processing (NLP) techniques. Later, by finding similarities among the movie description and the free text query, recommendations can be made accordingly. In this way, user can get personalized recommendations even without knowing the user history of previously watched movies.

Keywords:Movie Recommendations, Text to vector, Vector similarity, Natural Language Processing, Machine Learning, Content-based Filtering, Hybrid approach

1. Introduction

Due to abundance of information collected till 21st century and the increasing rate of information flowing over the internet, there is a lot of confusion related to what to consume and what not to consume. Even on YouTube, when you want to watch a video of a particular concept, generally, there are a lot of videos available out there for you. Now, since the results are ranked appropriately, there may not be much issue but what if the results were not ranked appropriately? Well, in that case, we would probably spend a lot of time to find the best possible video which suits us and satisfies our need. This recommendation results are when you search something on a website. Next time, when you visit a particular website, without even searching, sometimes the system is able to show you recommendations which you might like. Isn't this an interesting feature? So, basically, the job of a recommender system is to suggest the most relevant items to the user. Recommendation systems are used in YouTube for video recommendation, Amazon and Flipkart for product recommendation, Netflix and Amazon Prime for movie recommendation, and so on. Whatever you do on such websites, there is a system which see your behavior and then ultimately suggest things / items with which you are highly likely to engage. This research paper deals with movie recommendations and logic behind movie recommendation system, traditional movie recommendation systems, issues related to traditional movie recommendation systems, and a proposed solution for Artificial Intelligence based personalized movie recommendation system. A lot of famous movie recommendation related datasets are already available on Kaggle and other websites. Some of the famous datasets include Movielens dataset, TMDB Movie Dataset, and the dataset by Netflix itself. Websites like Netflix, Amazon Prime, etc. use movie recommendation to increase their revenue or profits by ultimately improving the user experience. In fact, there was a competition conducted by Netflix in the year 2009 with a prize money of nearly 1 million dollars ($1M) for making at least 10% improvement in the existing system.
As dealt earlier, we have a lot of data available at our exposure and we need to filter the data in order to consume it because generally we are not interested in each and everything available to us. In order to filter the data, we need some filtering techniques. There are different types of filtering techniques or movie recommendation algorithms over which a recommendation system can be based upon.
Major filtering techniques or movie recommendation algorithms are as follows:

E-mail address:
madhavikolse@gmail.com[a]
monikarokade4@gmail.com[b]

1. Content Based Filtering
2. Collaborative Filtering
3. Hybrid Filtering

Some of these techniques can be further broken into subparts.

2. Literature Survey

Sang-Min Choi, et. al. [1] mentioned about the shortcomings of collaborative filtering approach like sparsity problem or the cold-start problem. In order to avoid this issue, the authors have proposed a solution to use category information. The authors have proposed a movie recommendation system which is based on genre correlations. The authors stated that the category information is present for the newly created content. Thus, even if the new content does not have enough ratings or enough views, still it can pop up in the recommendations list with the help of category or genre information. The proposed solution is unbiased over the highly rated most watched content and new content which is not watched a lot. Hence, even a new movie can be recommended by the recommendation system.

George Lekakos, et. al. [2] proposed a solution of movie recommendation using hybrid approach. The authors stated that Content based filtering and Collaborative filtering have their own shortcomings are can be used in a specific situation. Hence, the authors have come up with a hybrid approach which takes into consideration both content-based filtering as well as collaborative filtering. The solution is implemented in 'MoRe' which is a movie recommendation system. For the sake of pure collaborative filtering, Pearson correlation coefficient has not been used. Instead, a new formula has been used. But this formula has an issue of 'divide by zero' error. This error occurs when the users have given same rating to the movies. Hence, the authors have ignored such users. In case of pure content-based recommendation system, the authors have used cosine similarity by taking into consideration movie writers, cast, directors, producers and the movie genre. The authors have implemented a hybrid recommendation method by using 2 variations - 'substitute' and 'switching'. Both of these approaches show results based on collaborative filtering and show recommendations based on content-based filtering when a certain criterion is met. Hence, the authors use collaborative filtering technique as their main approach.

Debashis Das, et. al. [3] wrote about the different types of recommendation systems and their general information. This was a survey paper on recommendation systems. The authors mentioned about Personalized recommendation systems as well as non-personalized systems. User based collaborative filtering and item based collaborative filtering was explained with a very good example. The authors have also mentioned about the merits and demerits of different recommendation systems.

Jiang Zhang, et. al. [4] proposed a collaborative filtering approach for movie recommendation and they named their approach as 'Weighted KM-Slope-VU'. The authors divided the users into clusters of similar users with the help of K-means clustering. Later, they selected a virtual opinion leader from each cluster which represents the all the users in that particular cluster. Now, instead of processing complete user-item rating matrix, the authors processed virtual opinion leader-item matrix which is of small size. Later, this smaller matrix is processed by the unique algorithm proposed by the authors. This way, the time taken to get recommendations is reduced.

S. Rajarajeswari, et. al. [5] discussed about Simple Recommender System, Content-based Recommender System, Collaborative Filtering based Recommender System and finally proposed a solution consisting of Hybrid Recommendation System. The authors have taken into consideration cosine similarity and SVD. Their system gets 30 movie recommendations using cosing similarity. Later, they filter these movies based on SVD and user ratings. The system takes into consideration only the recent movie which the user has watched because the authors have proposed a solution which takes as input only one movie.

Muyeed Ahmed, et. al. [6] proposed a solution using K-means clustering algorithm. Authors have separated similar users by using clusters. Later, the authors have created a neural network for each cluster for recommendation purpose. The proposed system consists of steps like Data Pre-processing, Principal Component Analysis, Clustering, Data Pre-processing for Neural Network, and Building Neural Network. User rating, user preference, and user consumption ratio have been taken into consideration. After clustering phase, for the purpose of predicting the ratings which the user might give to the unwatched movies, the authors have used neural network. Finally, recommendations are made with the help of predicted high ratings.

Gaurav Arora, et. al. [7] have proposed a solution of movie recommendation which is based on users' similarity. The research paper is very general in the sense that the authors have not mentioned the internal working details. In the Methodology section, the authors have mentioned about City Block Distance and Euclidean Distance but have not mentioned anything about cosine similarity or other techniques. The authors stated that the recommendation system is based on hybrid approach using context based filtering and collaborative filtering but neither they have stated about the parameters used, not they have stated about the internal working details.

V. Subramaniyaswamy, et. al. [8] have proposed a solution of personalized movie recommendation which uses collaborative filtering technique. Euclidean distance metric has been used in order to find out the most similar user. The user with least value of Euclidean distance is found. Finally, movie recommendation is based on what that particular user has best rated. The authors have even claimed that the recommendations are varied as per the time so that the system performs better with the changing taste of the user with time.

According to R. Lavanya, et. al. [9], in order to tackle the information explosion problem, recommendation systems are helpful. Authors mentioned about the problems of data sparsity, cold start problem, scalability, etc. Authors have done a literature review of nearly 15 research papers related to movie recommendation system. After reviewing all these papers, they observed that most of the authors have used collaborative filtering rather than content-based filtering. Also, the authors noticed that a lot of authors have used hybrid-based approach. Even though a lot of research has been done on recommendation systems, there is always a scope for doing more in order to solve the existing drawbacks.

Ms. NeeharikaImmaneni, et. al. [10] proposed a hybrid recommendation technique which takes into consideration both content-based filtering approach as well as collaborative filtering approach in a hierarchical manner in order to show a personalized movie recommendation to the users. The most unique thing about this research work is that the authors have made movie recommendations using a proper sequence of images which actually describe the movie story plot. This actually helps for better visuals. The author has also described the graph-based recommendation system, content-based approaches, hybrid recommender systems, collaborative filtering systems, genrecorrelations-based recommender system, etc. The proposed algorithm has 4 major phases. Initially, social networking website like Facebook is used to know the user interest. Later, the movie reviews need to be analysed and the recommendations needs to be made. Finally, story plot needs to be generated for better visuals.

Md. Akter Hossain, et. al. [11] proposed NERS which is an acronym for neural engine-based recommender system. The authors have done a successful interaction between 2 datasets carefully. Moreover, the authors stated that the results of their system are better than the existing systems because they have incorporated the usage of general dataset as well as the behaviour-based dataset in their system. The authors have used 3 different estimators in order to evaluate their system against the existing systems.

Shreya Agrawal, et. al. [12] proposed a hybrid approach of movie recommendation system which uses content-based filtering as well as collaborative filtering. The authors have used Support Vector machine as a classifier and genetic algorithm in order to improve quality, accuracy and scalability of a movie recommendation system. Existing pure movie recommendation system approaches and hybrid approaches have been compared using 3 different MovieLens datasets. The proposed approach also has an edge over 2 pure recommendation systems in terms of computing time.

Muppana Mahesh Reddy, et. al. [13] discussed about the problems related to movie recommendation system such as sparsity problem, grey sheep problem, long-tail problem, etc. "Grey Sheep" users is a group of the users who have special tastes and they may neither agree nor disagree with the majority of the users. According to the authors, these problems can be solved or at least minimized by taking correct decisions related to the movies which needs to be taken into consideration and the movies which needs to be ignored. Collaborative filtering technique has been used along with Pearson correlation in order to find similarity. Authors have done analysis by taking into consideration all movies and by neglecting movies with less than average ratings. Finally, authors came to a conclusion that the movies whose average rating is less than 2.5 can be ignored.

Noor Ifada, et. al. [14] compared the hybrid approach with the collaborative filtering approach of movie recommendation. Hybrid approach consists of a combination of content-based filtering and collaborative filtering. Authors came to a conclusion that the hybrid approach is not always better than the collaborative filtering approach. K-means clustering algorithm has been used by the authors to generate clusters of similar movies.

Mukesh Kumar Kharita, et. al. [15] mentioned that the collaborative filtering technique works fine if we have enough data. Authors discussed about user based collaborative filtering as well as item based collaborative filtering. Finally, the authors have implemented the item based collaborative filtering technique in order to show movie recommendations to the users. On the basis of the updating done by the users, the changes get reflected in the recommendations as well. In this way, the authors have focussed on real-time recommendations. Modified cosine similarity has been used by the authors in order to find the similarity.

3. Proposed Methodology

We need to perform preprocessing on the dataset and combine the relevant features into a single feature. Later, we need to convert the text from that particular feature into vectors. Later, we need to find the similarity between the vectors. Finally, get the recommendations as per the system architecture mentioned below.

3.1. Architecture

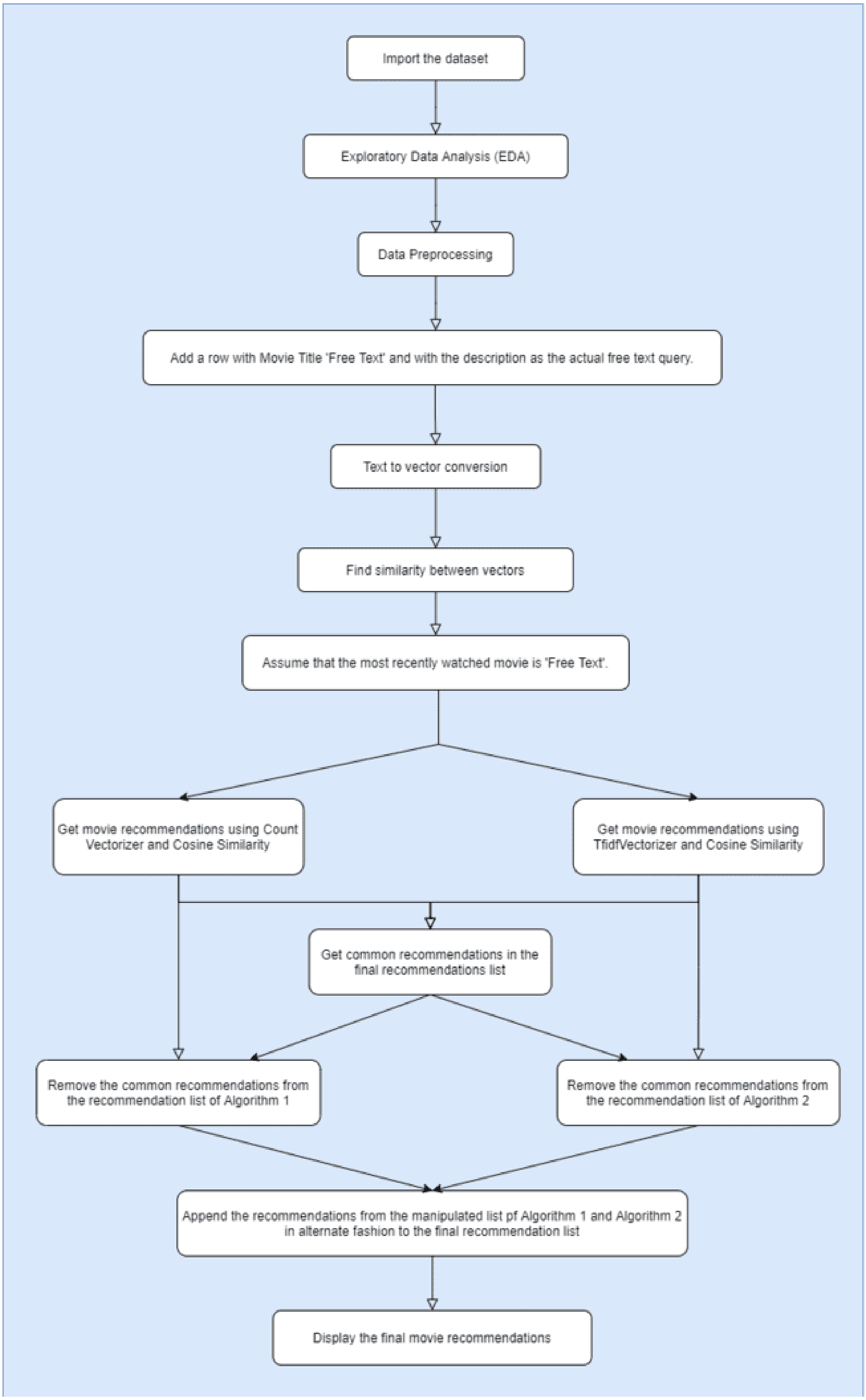

Fig. 1 - System Architecture

3.2. Dataset, Exploratory Data Analysis& Preprocessing

The 'TMDB 5000 Movie Dataset' is taken into consideration for movie recommendation purpose in this research work. This dataset is available on kaggle.com. The dataset is composed of 2 CSV files - 'tmdb_5000_movies.csv' and 'tmdb_5000_credits.csv'

The 'tmdb_5000_movies.csv' dataset consists of the following attributes:

- 'budget': It indicates the budget of the movie.
- 'genres': It indicates the genres of the movie like Action, Documentory, etc. A movie can have multiple genres.
- 'homepage': It indicates the homepage of the movie. It is basically a website link.
- 'id': It indicates movie ID.
- 'keywords': It indicates the keywords of the movie. Apart from the title of the movie, keywords give a quick information about the movie.
- 'original_language': It indicates whether the movie is originally created in English or other language.
- 'original_title': It is nothing but the movie title.
- 'overview': It is a short description of the movie.
- 'popularity': It is a metric which indicates popularity.
- 'production_companies': It consists of the names of companies which has produced the movie.
- 'production_countries': It consists of the names of the countries in which the movie production took place.
- 'release_date': It consists of the release date of the movie. The format used is yyyy-mm-dd where 'yyyy' indicates year of release, 'mm' indicates the month of release, and 'dd' indicates the day of release.
- 'revenue': It indicates the revenue earned by the movie.
- 'runtime': It indicates the runtime of a movie. Runtime basically means the length of the movie.
- 'spoken_languages': It consists of the languades spoken in the movie.
- 'status': It indicates the status of the movie. For example, a movie can be released or not released which basically indicates the status of that movie.
- 'tagline': It consists of the tagline of the movie.
- 'title': It consists of the title of the movie.
- 'vote_average': It indicates the average of the votes.
- 'vote_count': It indicates the vote count.

	budget	id	popularity	revenue	runtime	vote_average	vote_count
count	4.803000e+03	4803.000000	4803.000000	4.803000e+03	4801.000000	4803.000000	4803.000000
mean	2.904504e+07	57165.484281	21.492301	8.226064e+07	106.875859	6.092172	690.217989
std	4.072239e+07	88694.614033	31.816650	1.628571e+08	22.611935	1.194612	1234.585891
min	0.000000e+00	5.000000	0.000000	0.000000e+00	0.000000	0.000000	0.000000
25%	7.900000e+05	9014.500000	4.668070	0.000000e+00	94.000000	5.600000	54.000000
50%	1.500000e+07	14629.000000	12.921594	1.917000e+07	103.000000	6.200000	235.000000
75%	4.000000e+07	58610.500000	28.313505	9.291719e+07	118.000000	6.800000	737.000000
max	3.800000e+08	459488.000000	875.581305	2.787965e+09	338.000000	10.000000	13752.000000

Fig. 2 - Statistical data about 'tmdb_5000_movies.csv' dataset using pandas Dataframe.describe() method

International Journal of Research Publication and Reviews Vol (2) Issue (7) (2021) Page 877-891

```
movies.iloc[25]
```

```
budget                                                          200000000
genres                                  ['Drama', 'Romance', 'Thriller']
homepage                                    http://www.titanicmovie.com
id                                                                    597
keywords                 ['shipwreck', 'iceberg', 'ship', 'panic', 'tit...
original_language                                                      en
original_title                                                    Titanic
overview                 84 years later, a 101-year-old woman named Ros...
popularity                                                        100.026
production_companies     ['Paramount Pictures', 'Twentieth Century Fox ...
production_countries     [{"iso_3166_1": "US", "name": "United States o...
release_date                                                   1997-11-18
revenue                                                        1845034188
runtime                                                               194
spoken_languages         [{"iso_639_1": "en", "name": "English"}, {"iso...
status                                                           Released
tagline                        Nothing on Earth could come between them.
title                                                             Titanic
vote_average                                                          7.5
vote_count                                                           7562
Name: 25, dtype: object
```

Fig. 3 - Glimpse of the 'tmdb_5000_movies.csv' dataset using 'Titanic' movie

The 'tmdb_5000_credits.csv' dataset consists of the following attributes:

- 'movie_id': It indicates the movie ID.
- 'title': It indicates the title of the movie.
- 'cast': It consists of the cast of the movie. Cast implies the actors and actresses who appear in the movie.
- 'crew': It consists of those people who are concerned with the production of the movie.

	movie_id
count	4803.000000
mean	57165.484281
std	88694.614033
min	5.000000
25%	9014.500000
50%	14629.000000
75%	58610.500000
max	459488.000000

Fig. 4 - Statistical data about 'tmdb_5000_credits.csv' dataset using pandas Dataframe.describe() method

```
credits.iloc[25]
```

```
movie_id                                                          597
title                                                         Titanic
cast        ['Kate Winslet', 'Leonardo DiCaprio', 'Frances...
director                                               James Cameron
Name: 25, dtype: object
```

Fig. 5 - Glimpse of the 'tmdb_5000_credits.csv' dataset using 'Titanic' movie

The Exploratory Data Analysis (EDA) has been inspired by HeeralDedhia's blog on medium.com.

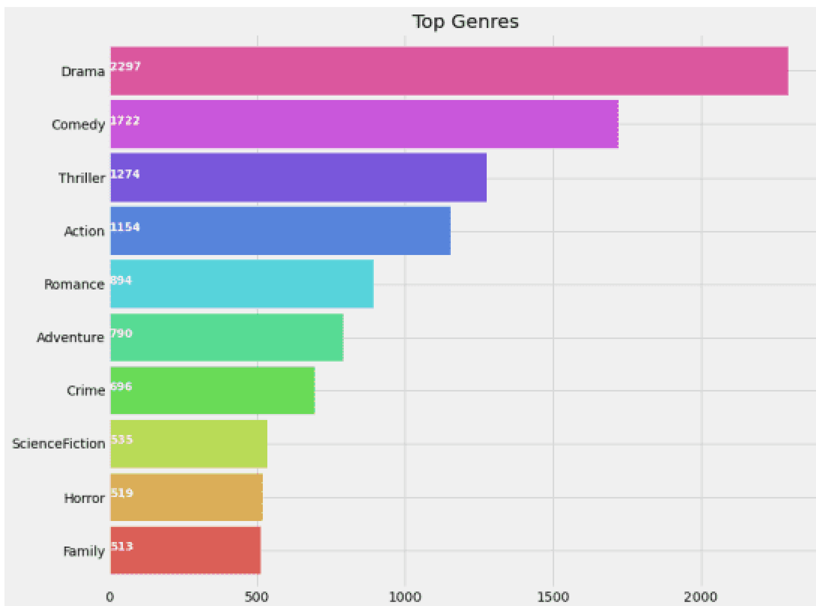

Fig. 6 - Top Genres

Movies having the genre as Drama are maximum in number as compared to Family movies and Horror movies. A movie might have multipe genres.

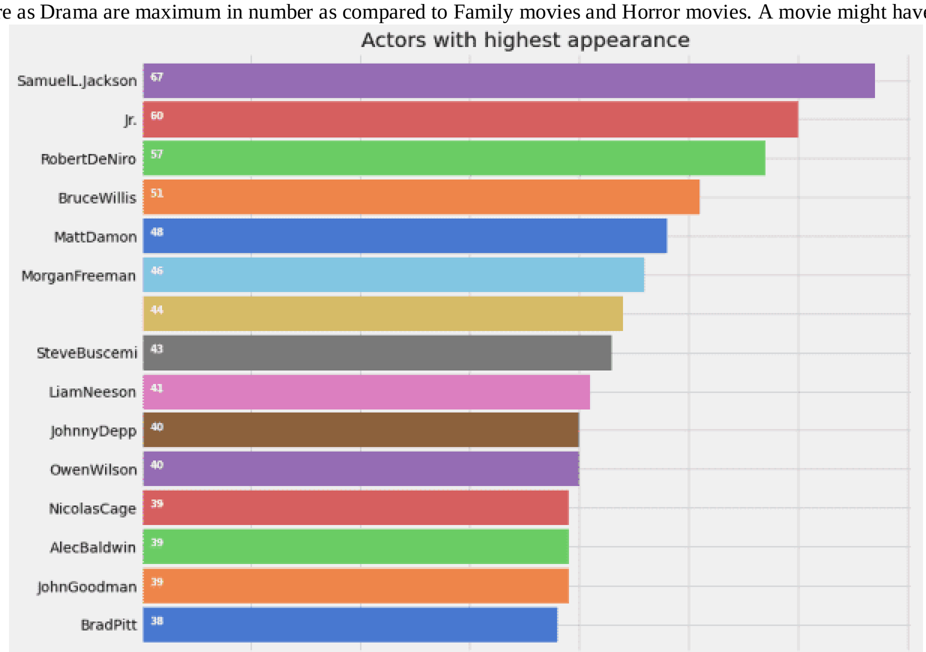

Fig. 7 - Actor with highest appearance

The above figure indicates the actors with the highest appearance in the decreasing order.

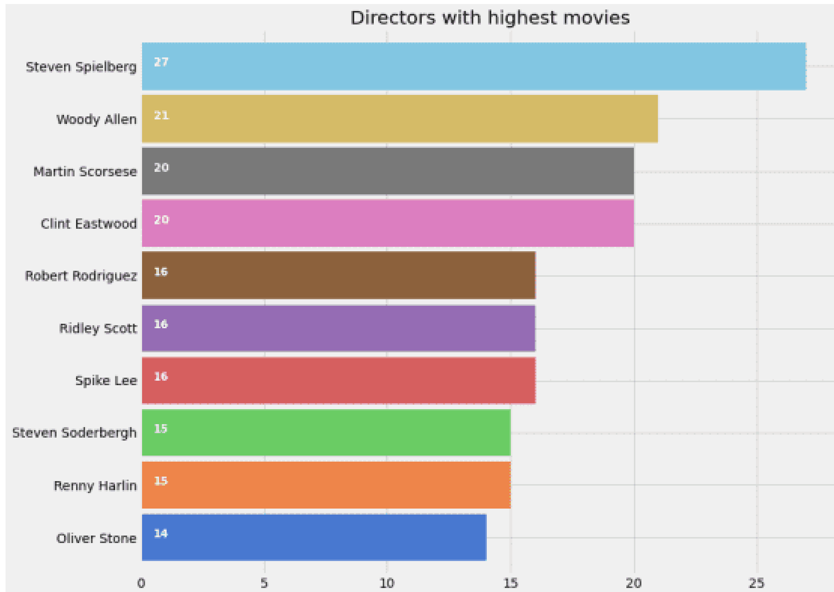

Fig. 8 - Directors with highest movies

The above figure indicates the directors with the highest appearance in the decreasing order.

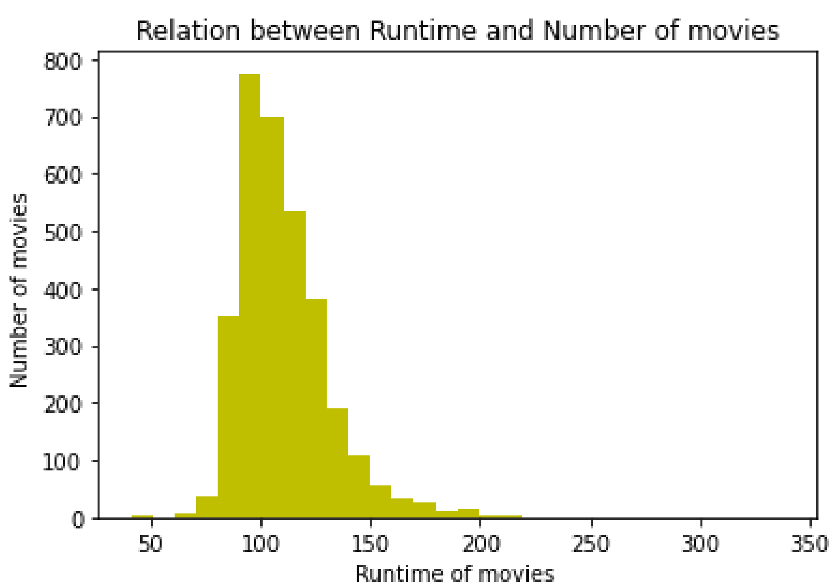

Fig. 9 - Runtime versus Number of movies

As the runtime increases, number of movies are increasing. After certain point, as the runtime increases, the number of movies decreases. There are some exceptions.

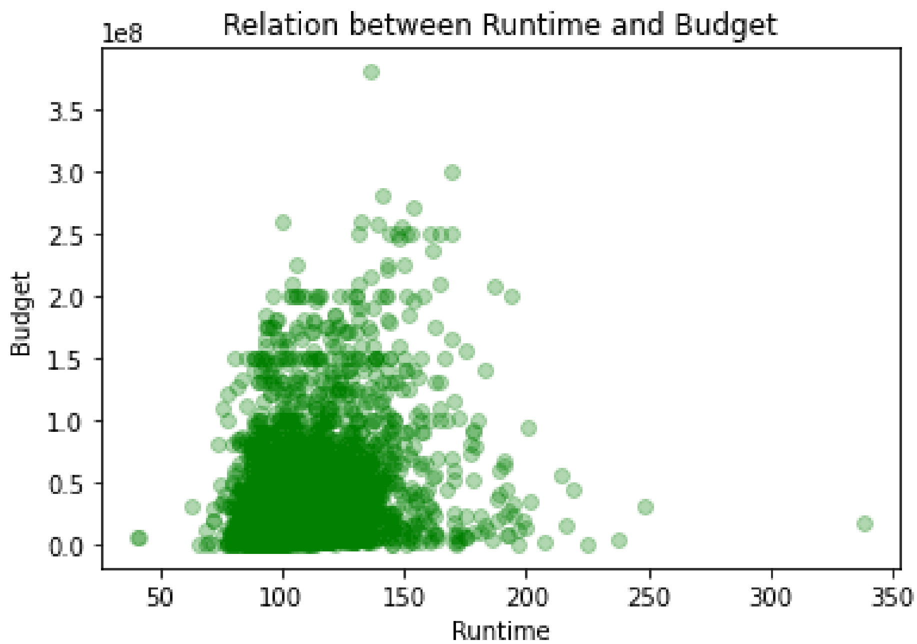

Fig. 10 - Runtime versus Budget

There are a lot of movies with lower budget and falling in the range of runtime 70 to runtime 150.

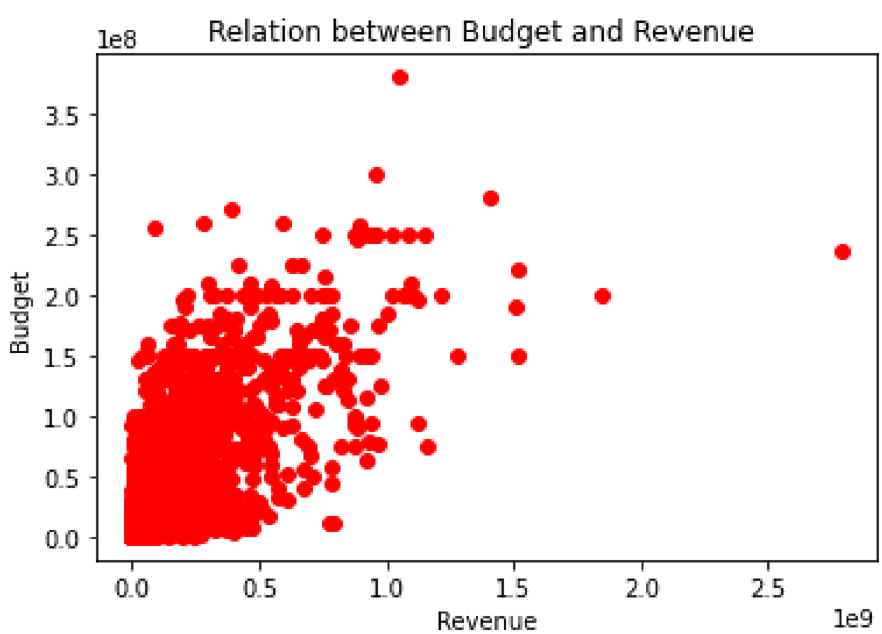

Fig. 11 - Revenue versus Budget

It can be seen from the above figure that low budget movies have low revenue in general.

International Journal of Research Publication and Reviews Vol (2) Issue (7) (2021) Page 877-891

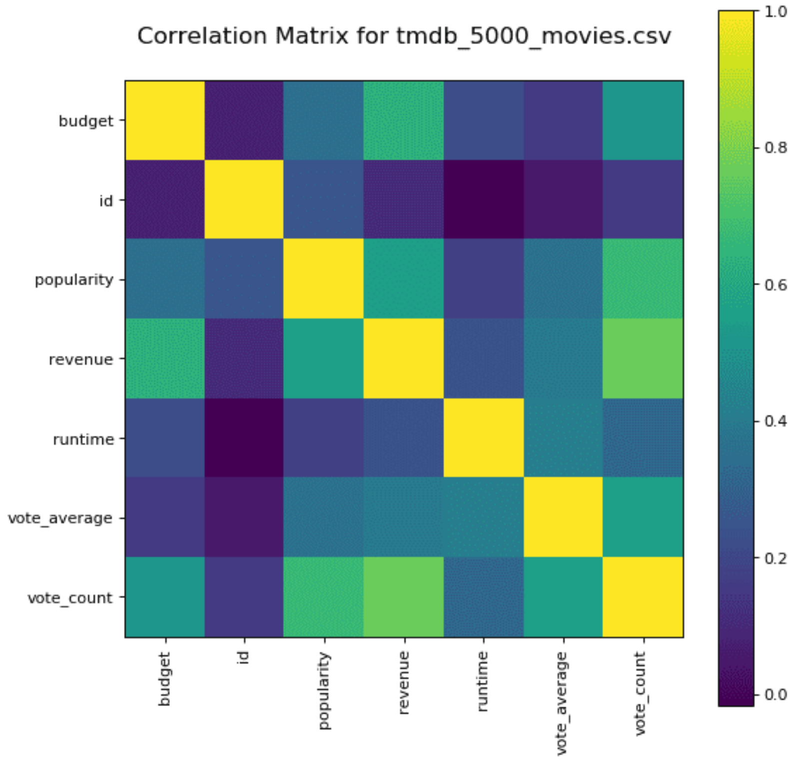

Fig. 12 - Correlation Matrix for 'tmdb_5000_movies.csv' dataset

From the above correlation matrix, it can be seen that the diagonal is yellow colored because similarity of something with itself is always 1.0, i.e., maximum. Moreover, it can be seen that revenue and vote count have more similarity as compared to budget and vote count.

Preprocessing steps include removing stopwords, combining the first name and the last name into a single name, removing punctuation marks, lowercasing the text, etc.

	title	combine_feature
0	Avatar	cultureclash future spacewar samworthington zo...
1	Pirates of the Caribbean: At World's End	ocean drugabuse exoticisland johnnydepp orland...
2	Spectre	spy basedonnovel secretagent danielcraig chris...
3	The Dark Knight Rises	dccomics crimefighter terrorist christianbale ...
4	John Carter	basedonnovel mars medallion taylorkitsch lynnc...
...	...	...
4798	El Mariachi	unitedstates–mexicobarrier legs arms carlosgal...
4799	Newlyweds	edwardburns kerrybishé marshadietlein edwardb...
4800	Signed, Sealed, Delivered	date loveatfirstsight narration ericmabius kri...
4801	Shanghai Calling	danielhenney elizacoupe billpaxton danielhsia
4802	My Date with Drew	obsession camcorder crush drewbarrymore brianh...

4803 rows × 2 columns

Fig. 13 - Director, Keywords, Cast and Genres of a movie are combined into a single feature titled 'combine_feature'

The 'combine_feature' attribute needs to be further processed by using some algorithms.

3.3. Algorithms

We can use CountVectorizer or TfidfVectorizer or Glove or Word2Vec in order to create vectors from the text. After converting the text into vectors, we need to find the similarity between the vectors. Cosine Similarity or sigmoid_kernel or some other technique can be used to find the similarity between the vectors.

1. **Algorithm 1: Content-based Recommendation using CountVectorizer and Cosine Similarity**
   In this case, we will use CountVectorizer in order to create vectors from the preprocessed text mentioned in the 'combine_feature' attribute. After getting the vectors, we will find the similarity between the vectors using Cosine Similarity.

2. **Algorithm 2: Content-based Recommendation using TfidfVectorizer and Cosine Similarity**
   In this case, we will use TfidfVectorizer in order to create vectors from the preprocessed text mentioned in the 'combine_feature' attribute. After getting the vectors, we will find the similarity between the vectors using Cosine Similarity.

After getting the recommendations using Algorithm 1 and Algorithm 2, get the common movies from both the recommendations initially. Later, append the remaining movies to the common movies in an alternate fashion.

4. Results

```
Recommendations on the free text "I wanna see the future." using
Cosine Similarity and Count Vectorizer are as follows:

422                                     The 6th Day
224                                         RoboCop
301                                     Cloud Atlas
466                                The Time Machine
2443                                 Dragon Hunters
2654                                       Automata
2696                                        Jason X
3854    Batman: The Dark Knight Returns, Part 2
4651                      The Sticky Fingers of Time
0                                            Avatar
775                                       Supernova
1399                                        In Time
2127                      Bill & Ted's Bogus Journey
2835                      The Muppet Christmas Carol
3065                                     Heartbeeps
Name: title, dtype: object
```

Fig. 14 - Recommendations on the free text "I wanna see the future." using Cosine Similarity and Count Vectorizer

```
Recommendations on the free text "I wanna see the future." using
Cosine Similarity and TFIDFVectorizer are as follows:

301                               Cloud Atlas
422                               The 6th Day
224                                   RoboCop
466                          The Time Machine
1399                                  In Time
85        Captain America: The Winter Soldier
0                                      Avatar
127                        Mad Max: Fury Road
2835               The Muppet Christmas Carol
775                                 Supernova
2654                                 Automata
4073                                  Sleeper
2127               Bill & Ted's Bogus Journey
3323                                  Code 46
4225                                 Repo Man
Name: title, dtype: object
```

Fig. 15 - Recommendations on the free text "I wanna see the future." using Cosine Similarity and TFIDFVectorizer

```
Final Recommendations for the free text  "I wanna see the future." are as follows:

The 6th Day
RoboCop
Cloud Atlas
The Time Machine
Automata
Avatar
Supernova
In Time
Bill & Ted's Bogus Journey
The Muppet Christmas Carol
Dragon Hunters
Captain America: The Winter Soldier
Jason X
Mad Max: Fury Road
Batman: The Dark Knight Returns, Part 2
Sleeper
The Sticky Fingers of Time
Code 46
Heartbeeps
Repo Man
```

Fig. 16–Final Recommendations for the free text "I wanna see the future."

```
Recommendations on the free text "I love university life." using
Cosine Similarity and Count Vectorizer are as follows:

2861                Sorority Boys
3551                  Miss Julie
463                      Déjà Vu
1935    The World's Fastest Indian
2513                     Tootsie
2813              Think Like a Man
3791                 Among Giants
4364           Two Girls and a Guy
4510          Juliet and Alfa Romeo
4776          In the Company of Men
614                 Despicable Me
1599            The Age of Adaline
1603          Stranger Than Fiction
1890              Don Juan DeMarco
1891                   Dear John
Name: title, dtype: object
```

Fig. 17 - Recommendations on the free text "I love university life." using Cosine Similarity and Count Vectorizer

```
Recommendations on the free text "I love university life." using
Cosine Similarity and TFIDFVectorizer are as follows:

2861                Sorority Boys
1935    The World's Fastest Indian
614                 Despicable Me
2606              Raise Your Voice
1891                   Dear John
2500                       Vamps
2438              The Illusionist
2513                     Tootsie
1603          Stranger Than Fiction
3551                  Miss Julie
1997                         Her
2968           In the Land of Women
2560    The Object of My Affection
4776          In the Company of Men
4364           Two Girls and a Guy
Name: title, dtype: object
```

Fig. 18 - Recommendations on the free text "I love university life." using Cosine Similarity and TFIDFVectorizer


```
Final Recommendations for the free text  "I love university life." are as follows:

Sorority Boys
Miss Julie
The World's Fastest Indian
Tootsie
Two Girls and a Guy
In the Company of Men
Despicable Me
Stranger Than Fiction
Dear John
Déjà Vu
Raise Your Voice
Think Like a Man
Vamps
Among Giants
The Illusionist
Juliet and Alfa Romeo
Her
The Age of Adaline
In the Land of Women
Don Juan DeMarco
The Object of My Affection
```

Fig. 19–Final Recommendations for the free text "I love university life."

5. Conclusion

Getting movie recommendations from a free text seems somewhat like a search engine where user writes a text query (free text) and gets the recommendations based on the free text. There might be more relevant movies existing but these may not be shown as recommendation because of the constraint on the dataset. User is able to get personalized movie recommendations by using this technique.

REFERENCES

[1]. Choi, Sang-Min, Sang-Ki Ko, and Yo-Sub Han. "A movie recommendation algorithm based on genre correlations." Expert Systems with Applications 39.9 (2012): 8079-8085.

[2]. Lekakos, George, and Petros Caravelas. "A hybrid approach for movie recommendation." Multimedia tools and applications 36.1 (2008): 55-70.

[3]. Das, Debashis, Laxman Sahoo, and Sujoy Datta. "A survey on recommendation system." International Journal of Computer Applications 160.7 (2017).

[3]. Zhang, Jiang, et al. "Personalized real-time movie recommendation system: Practical prototype and evaluation." Tsinghua Science and Technology 25.2 (2019): 180-191.

[5]. Rajarajeswari, S., et al. "Movie Recommendation System." Emerging Research in Computing, Information, Communication and Applications. Springer, Singapore, 2019. 329-340.

[6]. Ahmed, Muyeed, Mir Tahsin Imtiaz, and Raiyan Khan. "Movie recommendation system using clustering and pattern recognition network." 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC). IEEE, 2018.

[7]. Arora, Gaurav, et al. "Movie recommendation system based on users' similarity." International Journal of Computer Science and Mobile Computing 3.4 (2014): 765-770.

[8]. Subramaniyaswamy, V., et al. "A personalised movie recommendation system based on collaborative filtering." International Journal of High Performance Computing and Networking 10.1-2 (2017): 54-63.

[9]. R. Lavanya, U. Singh and V. Tyagi, "A Comprehensive Survey on Movie Recommendation Systems," 2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS), 2021, pp. 532-536, doi: 10.1109/ICAIS50930.2021.9395759.

[10]. N. Immaneni, I. Padmanaban, B. Ramasubramanian and R. Sridhar, "A meta-level hybridization approach to personalized movie recommendation," 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2017, pp. 2193-2200, doi: 10.1109/ICACCI.2017.8126171.

[11]. M. A. Hossain and M. N. Uddin, "A Neural Engine for Movie Recommendation System," 2018 4th International Conference on Electrical Engineering and Information & Communication Technology (iCEEiCT), 2018, pp. 443-448, doi: 10.1109/CEEICT.2018.8628128.
[12]. S. Agrawal and P. Jain, "An improved approach for movie recommendation system," 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), 2017, pp. 336-342, doi: 10.1109/I-SMAC.2017.8058367.
[13]. M. M. Reddy, R. S. Kanmani and B. Surendiran, "Analysis of Movie Recommendation Systems; with and without considering the low rated movies," 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), 2020, pp. 1-4, doi: 10.1109/ic-ETITE47903.2020.453.
[14]. N. Ifada, T. F. Rahman and M. K. Sophan, "Comparing Collaborative Filtering and Hybrid based Approaches for Movie Recommendation," 2020 6th Information Technology International Seminar (ITIS), 2020, pp. 219-223, doi: 10.1109/ITIS50118.2020.9321014.
[15]. M. K. Kharita, A. Kumar and P. Singh, "Item-Based Collaborative Filtering in Movie Recommendation in Real time," 2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC), 2018, pp. 340-342, doi: 10.1109/ICSCCC.2018.8703362.