MOVIE RECOMMENDATION SYSTEM USING MACHINE LEARNING

Anuja Gaikwad[1], Samradnyee Kudalkar[2], Ashwin Jethawa[3]

[1,2,3]UG Student, Dept. of IT Engineering, Vasantdada Patil Pratishthan's College of Engineering & Visual Arts, Mumbai, Maharashtra, India.

Abstract - *Nowadays, the recommendation system has made getting the things effortlessly that we need. The main purpose of Movie recommendation systems is to help movie enthusiasts by suggesting what movie to watch without the hassle to have to go through the time-consuming process of deciding from a large collection of movies which go up to millions is tedious and confusing. In this paper, we aim to minimise the human effort by suggesting movies based on the user's interests and preferences. To handle such problems, we introduced a model based on content-based approach and sentimental analysis. This system recommends movies by matching examples provided by the user to movie contents, which system derives from the movie director, cast, genre gathered from movie files, without using any human generated metadata also shows if the reviews are good or bad.*

Key Words: content-based approach, sentimental analysis, recommendation system, movie ratings, inductive learning

1.INTRODUCTION

People are often confronted with very large amounts of data, for instance through the internet in an information society. We are asked to make choices that are almost impossible to make without additional information or guidance. Recommender systems can provide such guidance by assisting the user in the decision-making process or by making the decision for the user. These systems use the enormous amount of available data in a way that users never can. Movie recommendation in portable environment is significantly important for users. A movie recommender has proven to be a powerful tool on providing useful movie suggestions for users. The content-based engine recommends personalized content based on certain predefined parameters. These non-exhaustively include a user's watch history, search history, and the items (movies, TV shows) that are currently being viewed. With rapidly increasing content, recommendation systems turn out as one of the prominent methods to deliver 'actual value 'to a customer - by being a scalable method to personalize content for them. Instead of reading long reviews which turn out to be a decisive factor for many users, sentimental analysis is used to check whether the review was good or bad.

2. RELATED WORK

Largely, recommender systems can be split into two categories: Collaborative Filtering and Content-Based Filtering.

Collaborative Filtering methods use user-related information, preferences, and user-item interactions to identify similarity between users. It recommends movies that similar users like. These can be further divided into two categories - Model-based and Memory-based algorithms. Memory-based methods do not have a training phase: they calculate similarity of the 'test 'user to training users and perform a weighted average of the most similar ones to give their recommendations. While earlier methods simply used measures like Pearson correlation coefficient, Cosine similarity to identify similar users, modern-day approaches also involve the analysis of co-rated items to remove irrelevant and dissimilar users, thereby reducing data sparsity. Model-based methods try to predict user-ratings of a movie using estimated models. Popular approaches in this category are the recommender systems used by big companies like Amazon, Netflix etc. While Amazon has been using collaborative filtering to recommend products to its customers for at least a decade, Netflix still values improvements to their recommendation services via the much distinguished 'Netflix Prize'. Collaborative filtering methods in general incur heavy computational overhead and perform poorly in the case of sparse data. Also, they assume that users with similar tastes rate similarly, which might not be true. A user may give higher ratings to items in general.

Content-Based methods (or cognitive filtering) on the other hand, use information and metadata about the content to find similarities among them, without incorporating user behavior in any way. Items similar to those 'accessed 'or 'searched 'by the user are recommended here. Some approaches analyze the audio and visual features (video frames, audio clips, movie posters etc.), as in using image and signal processing techniques while some analyze textual features (metadata like plots, subtitles, genre, cast etc.) via Natural Language Processing methods like tf-idf, as in , and word2vec, as in, The major difference between collaborative filtering and content-based recommender systems is that the former only uses the user item ratings to make recommendations, while the latter relies on the features of users and items for predictions. In this paper, we experiment with the latter approach.

3. LITERATURE REVIEW

Nessel stated in the movie oracle that working with examples is an essential part of human interaction and tried to provide a movie recommendation engine based on this behavior. Which of course requires considerably more computing power, as the compared bodies of text are much larger, but the algorithms are essentially the same [3].In a content-based movie recommendation system, the proposed algorithm uses textual metadata of the movies like plot, cast, genre, release year and other production information to analyze them and recommend the most similar ones [2].The paper also analyzes application similarity measure for recommendations forecasting in recommendations systems. It is shown that used method for computing similarity measure in recommendations systems are cosine similarity measure and Pearson correlation coefficient [1]. As the characteristics of movie recommendation go by, the user watching history is very important, so we add content-based recommendation approach. Typically, people have a tendency to think that positive reviews have a positive effect and negative reviews have negative impact. Sentiment analysis will assist us to improve the accuracy of recommendation results. Also, as we explained in our experimental results, it is necessary to make use of distributed system to solve the scalability and timeliness of recommender system [5].

4. TECHNIQUES USED IN METHODOLOGY

The proposed solution is for improving the scalability and quality of the movie recommendation system. For computing similarity between the different movies in the given dataset efficiently and in least time and to reduce computation time of the movie recommender engine we used cosine similarity measure. To check if a review to the same is positive or not we have used sentimental analysis method Naive Bayes classifier.

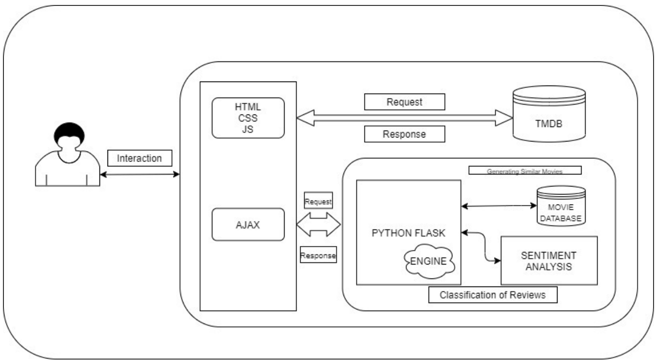


Fig -1: Architecture of the Movie Recommendation System

A. Content-based Filtering

A Content-Based movie recommendation system uses the data provided by users such as ratings, feedback, and reviews. A user profile is generated using this data which is then used to make recommendation to the user.

The engine becomes more accurate and robust, as the user takes more actions or provides more inputs on the recommendation system. Also, Term Frequency (TF) and Inverse Document Frequency (IDF) are used to retrieve the information and for content-based engine.

They are used to determine relative information such as

Movie, article, etc. Content-Based Filtering For the implementation of a content-based filtering system following steps to be done:

- Terms Allocation
- Terms Representation
- Learning Algorithm Selection
- Provide Recommendations

B. Term Frequency (TF) and Inverse Document Frequency (IDF)

TF refers to the frequency of a particular word in the document. IDF is the inverse of the document frequency in whole body of documents.TF-IDF is a statistical measure which determines how relevant a word is to a document in an accumulation of documents. It is most importantly used in automated text analysis and also in scoring words in machine learning algorithms for NLP.

In other words, the weight of a word in a document cannot be evaluated as a simple raw count and hence the equation below:

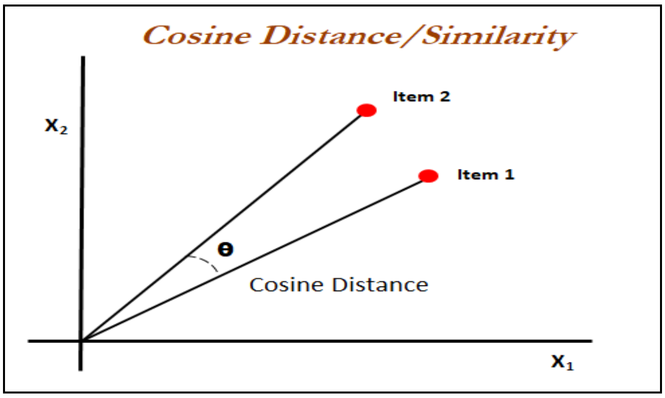
Equation:

$$w_{t,d} = \begin{cases} 1 + \log_{10} \mathrm{tf}_{t,d}, & \text{if } \mathrm{tf}_{t,d} > 0 \\ 0, & \text{otherwise} \end{cases}$$

Term Frequency	Weighted Term Frequency
0	0
10	2
1000	4

C. Cosine Similarity

Similarity Score is a numeric value which ranges between Zeros to one. Which is used to determine the similarity of two items to each other on a scale of zero to one. This score is obtained by measuring the similarity between texts of both the documents. Therefore, similarity score can be defined as the measure of similarity between given texts details of two given items. This can be done by- Cosine similarity. Cosine similarity is a measure used to determine how similar the texts are despite of their size. To calculate the cosine of angle between two vectors projected in a multi-dimensional space cosine similarity is used.



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

D. Sentimental analysis

Sentiment analysis is one of the Natural Language Processing fields, committed to the assessment of subjective opinions, views or feelings collected from a variety of sources about a specific subject. In more precise business terms, it can be summarized as "Sentiment Analysis is a set of tools to identify and extract opinions and use them for the benefit of the business operation ". Such algorithms push deep into the text and find the substance that points out the attitude towards the result in conventional or its specific element.

Another example is multinomial naive Bayes, here the features are presumed to be produced from a simple multinomial distribution. The multinomial distribution defines the possibility of observing counts between a number of categories, and thus multinomial naive Bayes is most suitable for features that represent counts or count rates. The idea is exactly the same as before, apart from that instead of modeling the data distribution with the best-fit Gaussian, we model the data distribution with a best-fit multinomial distribution.

P (positive | overall liked the movie) = P (overall liked the movie | positive) * P (positive) / P (overall liked the movie)

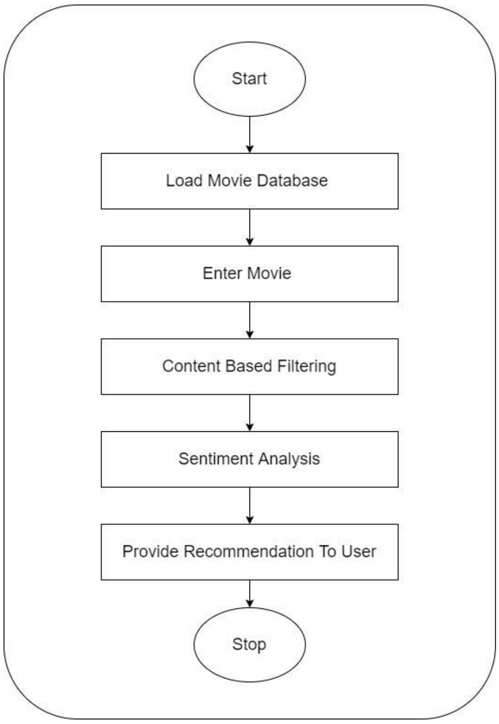


Fig -2: Flowchart of the Movie recommendation system

5. PROPOSED SYSTEM

A. Dataset

We have used three different data sets available in Movie Lens, which is generated by the group lens research team for the research work in the field of recommender system, to help developers to evaluate their recommendation systems. These are:

1. IMDB 5000 Movie Dataset
2. The Movies Dataset
3. List of movies in 2018
4. List of movies in 2019
5. List of movies in 2020

The Movies Dataset consists of metadata for all 45,000 movies listed in the Full MovieLens Dataset. This dataset contains movies released on or before July 2017. Data consist of cast, crew, plot keywords, budget, revenue, posters, release dates, languages, production companies, countries, TMDB vote counts and vote averages. This dataset also has files containing 26 million ratings from 270,000 users for 45,000 movies. Ratings are in the range of 1-5 and have been obtained from the official GroupLens website. The dataset of movies from 2018 – 2020 are acquired by web scraping their respective Wikipedia pages.

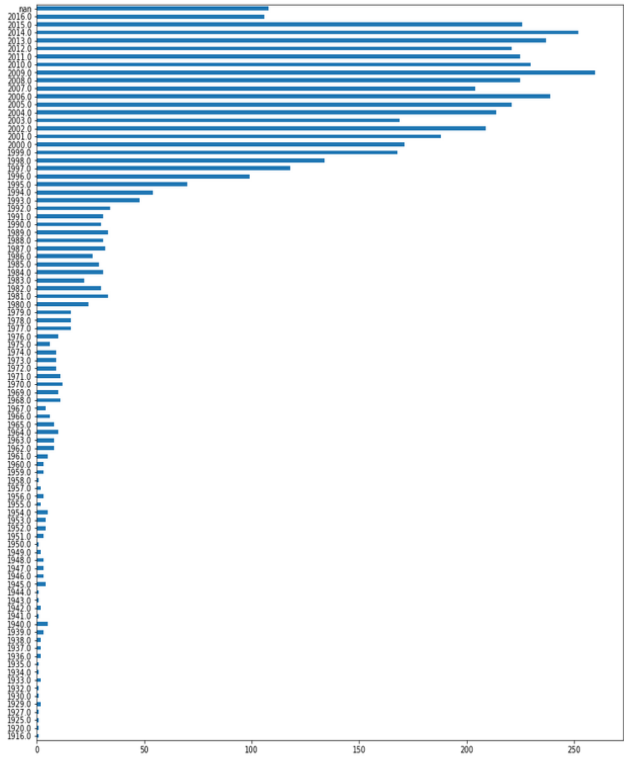


Fig -3: Plotted graph of The Movies Dataset

B. Recommendation system quality measures

We have used the TMDB Ratings to come up with our Top Movies Chart. And also IMDB's weighted rating formula to construct the chart.
Mathematically, it is represented as follows:

$$Weighted\ Rating(WR) = (\frac{v}{v+m}.R) + (\frac{m}{v+m}.C)$$

Where,

v represents the number of votes for the movie
m represents the minimum votes required to be listed in the chart
R represents the average rating of the movie
C represents the mean vote across the whole repot

We will take the top 25 movies based on similarity scores and calculate the vote of the **60th percentile** movie. Then, using this as the value, we will calculate the weighted rating of each movie using IMDB's formula.

```
In [81]: improved_recommendations('Pulp Fiction')
Out[81]:
```

	title	vote_count	vote_average	year	wr
898	Reservoir Dogs	3821	8	1992	7.718986
8310	Django Unchained	10297	7	2012	6.929017
7280	Inglourious Basterds	6598	7	2009	6.891679
4903	Kill Bill: Vol. 1	5091	7	2003	6.862133
8905	The Hateful Eight	4405	7	2015	6.842588
5200	Kill Bill: Vol. 2	4061	7	2004	6.830542
1381	Jackie Brown	1580	7	1997	6.621790
65	From Dusk Till Dawn	1644	6	1996	5.842293
6788	Death Proof	1359	6	2007	5.817225
4764	S.W.A.T.	780	5	2003	5.087550

Fig -4: Calculated weighted rating for the dataset.

6. RESULT ANALYSIS

A. Accuracy of Sentimental Analysis Model

The multinomial Naive Bayes classifier is fitting for classification with distinct features (e.g., word counts for text classification). The multinomial distribution usually needs integer feature counts. Nonetheless, in practice, fractional counts such as tf-idf could also work.

Accuracy of 98.77% is observed for the dataset provided.

```
In [20]: clf = naive_bayes.MultinomialNB()
         clf.fit(X,y)
Out[20]: MultinomialNB()

In [21]: accuracy_score(y_test,clf.predict(X_test))*100
Out[21]: 98.77167630057804
```

Fig -5: Observed accuracy of sentimental analysis.

B. Results of content-based movie recommendation system

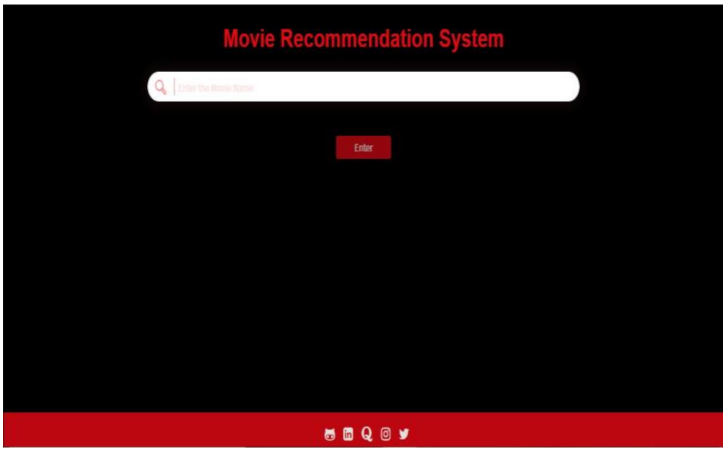


Fig -6: Home page of the Movie recommendation system

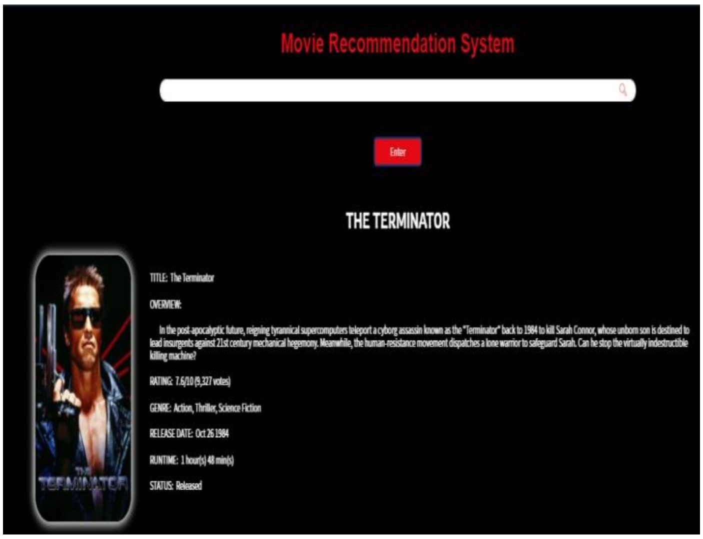


Fig -7: Poster and info page of the Movie

Fig -8: cast of the Movie

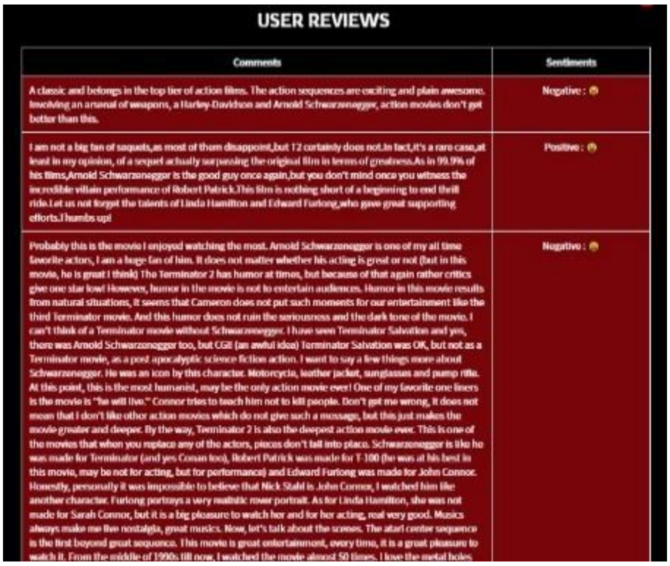


Fig -9:Sentimental analysis of the Movie recommendation

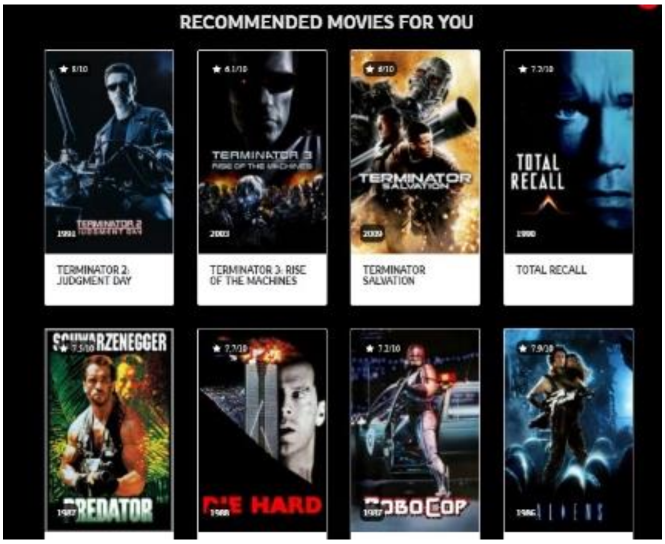


Fig -10: Recommended movies

7. CONCLUSION

The proposed algorithm uses textual metadata of the movies like plot, cast, genre, release year and other production information to analyse them and recommend the most similar ones. Our system only needs a movie which the user is interested in to come up with suitable recommendations. For evaluation, we ran our algorithm on a subset of all the movies present on the IMDb server. The paper analyzes application similarity measure for recommendations forecasting in recommendations systems. It is shown that used method for computing similarity measure in recommendations systems are cosine similarity measure. We also work on allowing retraining of the system, by rating results as "good" or "bad", thus making the predictions much more precise than just selecting one movie or giving one piece of text.

8. Future scope

Future work includes keeping a track of movies searched by users in nearby location to recommend trending movies. We can try to combine the watch history of the user with the watch history of geographically contextual users (those living nearby) to give more 'location relevant' recommendations. Furthermore, using user ratings of movies on websites like Rotten tomatoes, Metacritic, IMDb etc. opens up the possibility of combining collaborative filtering techniques with our method into a hybrid model to get the best out of both approaches.

REFERENCES

[1] Mykhaylo Schwarz, Mykhaylo Lobur, Yuriy Stekh, Analysis of the Effectiveness of Similarity Measures for Recommender Systems, 978-1-5090-5045-1/17/$31.00 ©2017 IEEE M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

[2] Rujhan Singla, Saamarth Gupta, Anirudh Gupta, Dinesh Kumar Vishwakarma, FLEX: A Content Based Movie Recommender, 978-1-7281-6221-8/20/$31.00 ©2020 IEEE.

[3] Jochen Nessel, Barbara Cimpa, The MovieOracle - Content Based Movie Recommendations, 978-0-7695-4513-4/11 $26.00 © 2011 IEEE.

[4] Shreya Agrawal, Pooja Jain, An Improved Approach for Movie Recommendation System, 978-1-5090-3243-3/17/$31.00 ©2017 IEEE

[5] Yibo Wang, Mingming Wang, and Wei Xu, A Sentiment-Enhanced Hybrid Recommender System for Movie Recommendation: A Big Data Analytics Framework, Hindawi Wireless Communications and Mobile Computing Volume 2018, Article ID 8263704

[6] F. Furtado, A, Singh, Movie Recommendation System Using Machine Learning, Int. J. Res. Ind. Eng. Vol. 9, No. 1 (2020) 84–98