srmDB.in – PRODUCT BASED MOVIE RECOMMENDATION SYSTEM USING AJAX REQUESTS WITH SENTIMENT EVALUATION

Vasu Jhawar
BTech *Computer Science*
SRMIST, Kattankulathur
Chennai, India
Vj8486@srmist.edu.in

R Lavanya
Assistant Professor
Computing Technologies
SRMIST, Kattankulathur
Chennai, India
lavanyar@srmist.edu.in

Aayush Agrawal
BTech *Computer Science*
SRMIST, Kattankulathur
Chennai, India
aa2100@srmist.edu.in

***Abstract* – There is plentiful data/content available online and it is increasing exponentially day by day. Therefore, the users need a product that can suggest movies at better accuracy and performance. Type of content liked by user varies from one user to another. Every online service company aims to grab as many clients as possible. Here, the Recommender systems come into play. The objective of this project is to basically build a fast & better movie recommendation system with review analysis. We have proposed to build a model using content-based filtering algorithm (supervised learning) with the help of cosine similarity measure and levenshtein distance for efficient results. The challenges faced by the users are the problems of the scalability, data sparsity and automation. By the end of this paper, we aim to eradicate these problems and build a logical and practical model using ajax requests, APIs, and few other resources.**

Keywords—content-based filtering, supervised learning, cosine similarity measure, levenshtein distance, Naïve Bayes classification, TF-IDF Vectorizer

I. INTRODUCTION

People living in today's world depends upon speed and genuine products. A user easily wants to be engaged with the content available in front of his/her eyes. The recommender system aim is to provide items to the users that the user is not aware of. Basically, a recommendation system as the name suggests help to provide related products that the user might like because that would be useful to him/her. There are tons of ways to provide recommendation to users depending on his/her needs. We will focus on the movie recommendation system which will provide suggestions or something to recommend them to watch more movies. Basically, a recommender system is a subclass of data filtering method that seeks to predict the rating or the preference a user might give to an item. Recommendation system offer help a lot in recommending content to users. The first recommendation system came in 1992, and it's still growing to achieve more accuracy and provide better results to users that in turn grows the organisation or the company. For example, people who buy Apple smartphones also tend to buy Apple smartwatch together, so a recommendation would be formed to recommend Apple watches whenever any user buys an Apple smartphone. With so many practical applications around us today, therefore, it is not possible to live without the recommendation systems.

There are many ways of recommending movies to users depending on the genre, language, etc. Additionally, there are methods that look for the closeness in b/w different users to provide a movie by the system. There are many algorithms to form a recommendation system such as

- Content based algorithm
- Collaborative algorithms
- Hybrid approach

Content Based Recommendation System: It's a supervised learning algorithms that has a set of outcomes for particular inputs and later the system predicts on different inputs provided by the user. It uses properties like genre, director, description, actor, etc. for movies, to make recommendations to users. In this project, we will mainly focus on the content-based recommendation system.

Collaborative filtering is an unsupervised learning where the system is given inputs but there is no particular output for the inputs , instead the system categorizes on similar patterns or shapes and classifies them as together. Predictions are made from ratings provided by people. Each row represents a person's movie rating and each column shows a movie's rating

Combined approach: In the hybrid approach, we combine the two recommended filtering techniques known as collaborative filtering with content-based filtering method to get the best benefit and achieve better results and reduce the challenges faced by the respective approach.

To build the movie recommendation system, we will work on the following steps:

- **Data Collection:** First the data is collected and chosen on which the system will work on.
- **Data Pre-processing:** Next the data is processed multiple times and the important part is taken out to work on (Training-testing data)
- **Model Creation:** The model is built on the algorithms chosen and first the testing data is taken to work on and eventually on whole dataset.
- **Website/App creation:** The website is created where the working of model is checked multiple

times to check the efficiency and ease UI/UX for user.

- **Deployment:** The last step of the project is to deploy it on cloud, it is the process of deploying the model in a real environment. The model can be deployed in a variety of different environments and will often be integrated into applications via an API. For this project, we have used Heroku cloud environment for deployment.

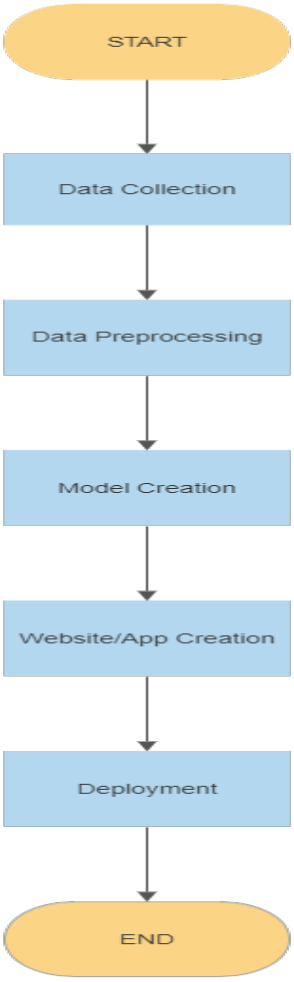


Figure 1 - Project Flow

II. The Dataset

To help evaluate the recommendation system, we have used three different data sets available in Movie Lens, which was generated by the group lens research team for the project.

1. IMDB 5000 Movie Dataset (test)
2. The Movies Dataset (train)
3. List of movies 2018-2020

The Movies dataset comprises of a metadata i.e. 45,000 movies in the Full Movies dataset. This list includes movies that are released on or prior to July 2017. The data includes crew, cast, plot keywords, budget, posters, gross, release date, language, manufacturing company, TMDB votes, country and average number of votes. This dataset is made up of files accommodating 260000000 (2.6 crores) reviews from 270,000 users for 45,000 movies. They are rated throughout 1-5 and are seized from the Group Lens official website.

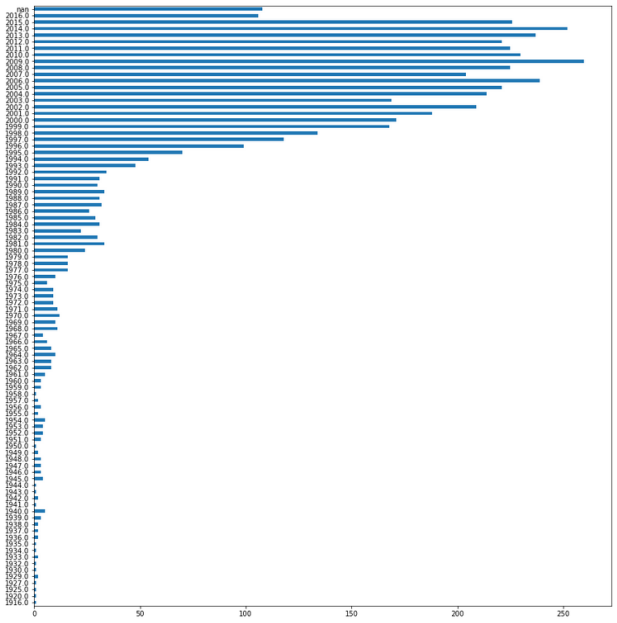


Figure 2-Plotted chart of Movie Dataset

III. Product Methodology

A. Architecture Diagram and System Working

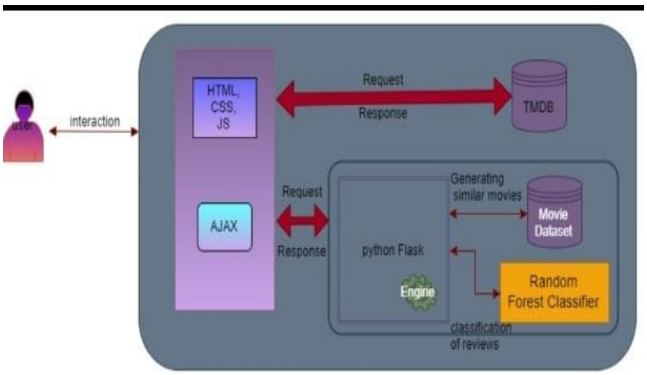


Figure 3 – Architecture Diagram of Recommendation System

With the help of python Flask, we were able to create a web framework for our project. The data was first collected and pre-processed as our requirement using python and its various libraries. The frontend and templates were created using HTML/CSS/JS. Further the recommendations are passed to user whenever a request is made with the help AJAX which allows the data to be sent and received to and from a database / server.

APIs were used to fetch the metadata (i.e. posters, title, ratings etc) from the TMDB database. TMDB API

provides is available for everyone to use. It provides a quick, consistent and reliable way to get the third party data.

The dataset used is: The Movie Dataset & Wikipedia (2018-2020). The training-testing data of **80-20** has not been used, instead the approach of **60-40** training-testing data has been used.

Then, the Sentiment Evaluation was performed on the reviews to check if they were positive or negative and to build the model for the same, we used the following features:

1) *Stop words*

2) *TFIDF Vectorizer* – It is abbreviated for Term Frequency Inverse Document Frequency. It simply is an algorithm that changes the text of the data into a relevant depiction of numbers that is fitted into the machine algorithm for predictions to be made. TF-IDF weighting negates the effect of high frequency words in determining the importance of an item (document). We are considering genre as an important metric to recommend movies to users. To evaluate the distance or the similarity between two movies, there are different techniques of measuring and in this paper we have used the cosine similarity.

3) *Naive Bayes* (Multinomial NB algorithm) – for checking the accuracy score of the model.

To use the model, we used the Web Scraping method to collect content and data from internet. This data is mostly saved in the local file to be worked upon. We used it to get the reviews and comments from the internet, and with the help of library BeautifulSoup4 in python which creates parse tree for HTML & XML texts. It helps in automated conversion of documents into Unicode.

B. System Analysis

To build an efficient product we have proposed a model that works on content-based filtering method with the cosine similarity measure to build a recommendation system which is more accurate and reliable.

Similarity (It decides which item is most similar to the item the user likes? Here we use similarity.)

This is a numeric value between 0 and 1 that measures how similar two items are to each other. each other on a scale of zero to one. This similarity is obtained by measuring the similarity between the textual details of the two elements. As such, similarity is a measure of the degree of similarity between the given textual details of two items. This can be accomplished by cosine similarity.

COSINE Similarity: It is a system of measuring the similarities b/w datasets. It overall represents the facet of the object in the dataset. In cosine similarity the data is taken and treated as some non-zero(0) vectors whose trigonometric cosine angle b/w them is taken to give the similarity measure. The dot product of the data is taken and the divided by their lengths.

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

Cosine similarity is beneficial as it is independent of the size of the dataset which is not in Euclidean distance method. I f some data is separated by a huge gap due to the size of the dataset, using cosine similarity it could have a small angle that represents higher the similarity.

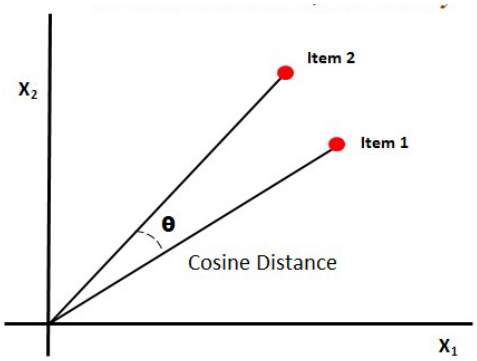
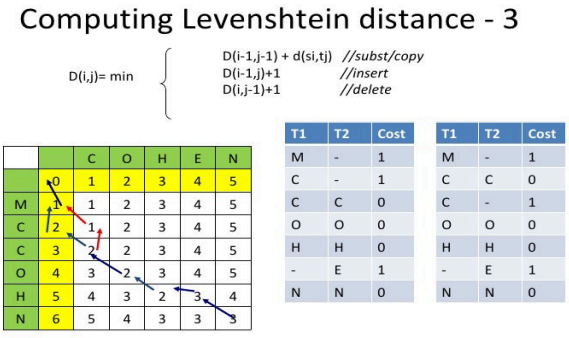


Figure 4 - Cosine Distance

LEVENSHTEIN DISTANCE:

It is a string matrix to measure the distance between two sequences easily. We used it to get the closest match for the searched movie.



C. Pseudo Code of Proposed System

Steps	Overview
Step 1	Import the dataset and perform the data pre-processing steps.
Step 2	Import the required libraries and generate the count matrix with the help of count vectorizer method.
Step 3	Then use the Cosine similarity measure to determine the angle b/w documents independent of their size.
Step 4	Initiate a directory setup for the website page where the main input field is set-out.

Step 5	Connect the page to flask and render it.
Step 6	.Then in the terminal open the python file and give the link into the browser.
Step 7	The user provides the name of movie and if that is available in the dataset, the cosine measure will be calculated giving the top 10 similar movies. It will be displayed to the user on the screen.
Step 8	If the movie that is searched is not in the dataset, a message will be displayed.

IV. RESULTS AND OUTCOMES

A. Accuracy of Sentimental Analysis Model

The Naive Bayes multinomial/polynomial classifier is suitable for classification with distinct features (Eg. word count for text classification). Polynomial distributions usually require an integer number of objects. In practice, however, fractional counts like Tfidf can also work.

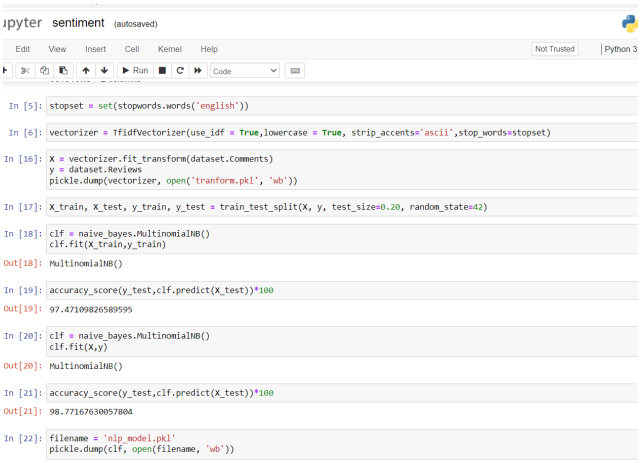
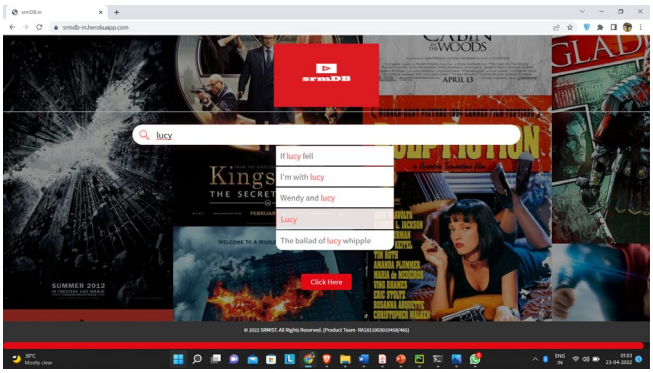


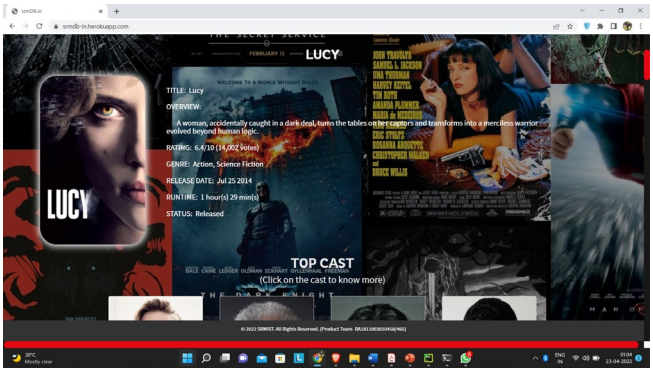
Fig -5: Observed accuracy of sentimental analysis.

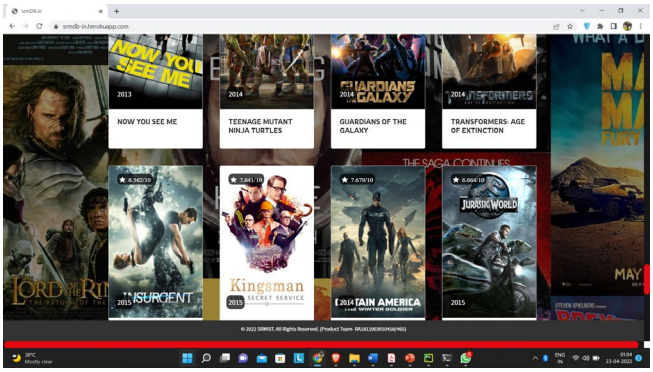
B. Product Features

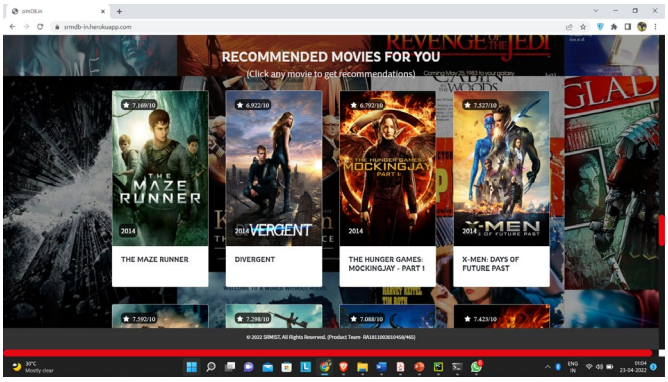
Live-working prototype - https://srmdb-in.herokuapp.com/



Automated search bar feature - https://cdn.jsdelivr.net/npm/@tarekraafat/autocomplete.js@7.2.0/dist/css/autoComplete.min.css







V. CONCLUSION

The recommended system(algorithm) uses the textual metadata of movies such as plot, cast, genre, release year, and other information to inspect them and then recommends the most similar movies related to the search to the user. Our system requires only 1 movie that the user is fascinated in to make the suitable recommendations.

Following Technology/Software Used:

- Python packages and libraries (Flask/nltk/scikit/numpy/pickle/pandas/etc.)
- HTML/CSS/JS
- JUPYTER NOTEBOOK/
- PYCHARM IDE 20.22
- Microsoft Visual Studio Code

In this paper we explicitly pointed out the problems and managed to eradicate them by building a better

recommender system. Also, we have taken a different scope of training and testing data which have increased the efficiency of the results achieved.
The accuracy of our system is 80% more than any other system and we got a website that can be used in any social networking site to recommend movies to the users.

VI. TERMS OF REFERNCE

Future work will include recommending popular movies by tracking movies that users have searched for in nearby locations. By combining a user's search history with that of a geographically contextual user (those who live nearby), we can provide more location-related recommendations. In addition, the use of user-rated movie ratings on websites such as Rotten Tomatoes, Metacritic, IMDb, etc. allows us to combine our method with a collaborative filtering method into a hybrid model to get the most out of both approaches.

VII. CODE AND DOCUMENTATION

In this project the code and data can be acquired from the below mentioned GitHub link
srmDB.in-Movie-Recommendation-System

ACKNOWLEDGMENT

We are grateful to Dr P Murali for his insightful and constructive suggestions during the project's design planning and development. It is much appreciated that he is prepared to devote his time.

We also want to thank Professor R Lavanya, our project guide, for her patient guidance, enthusiastic support, and constructive critiques of our research. We'd also like to express our gratitude to her for assisting us and keeping our progress on track.

REFERENCES

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

[2] Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich. Recommender systems: an introduction. Cambridge University Press, 2010.

[3] Suvir Bhargav. Efficient features for movie recommendation systems. 2014.

[4] 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

[5] 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.

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

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

[8] Choi, S. M., & Han, Y. S. (2010, September). A content recommendation system based on category correlations. *2010 Fifth international multi-conference on computing in the global information technology* (pp. 66-70). IEEE.

[9] Son, J., & Kim, S. B. (2017). Content-based filtering for recommendation systems using multiattribute networks. *Expert systems with applications*, *89*, 404-412.

[10] Konstan, J. A., & Riedl, J. (2012). Recommender systems: from algorithms to user experience. *User modeling and user-adapted interaction*, *22*(1-2), 101-123.

[11] http://www.awesomestats.in/python-recommending-movies/

[12] https://ieeexplore.ieee.org/document/8058367

[13] http://recommender-systems.org/content-based-filtering/

[14] https://www.academyofdatascience.com/Blog_page/blog_3.html