International Journal of Engineering Applied Sciences and Technology, 2019
Vol. 4, Issue 5, ISSN No. 2455-2143, Pages 526-528
Published Online September 2019 in IJEAST (http://www.ijeast.com)

MOVIE RECOMMENDATION SYSTEM USING BAG OF WORDS AND SCIKIT-LEARN

Sounak Bhattacharya
Department of CSE
SRM Institute of Science and Technology
Chennai, Tamil Nadu, India

Ankit Lundia
Department of CSE
SRM Institute of Science and Technology
Chennai, Tamil Nadu, India

Abstract— **Recommendation systems play an important role in our everyday life. Starting from movie/music streaming sites like Netflix or Spotify to the most basic search engines, the core component is comprised of recommendation systems. Even the technology giant Google is known for its search engine above everything else. To emphasize its importance we aim to build a simple recommendation system using the IMDB movie database. Our finished web application can suggest similar movies based on the input provided by the user, considering factors like the plot of the movie, actors present in it, as well as the director. Needless to say, some factors would be common between the input movie and the recommended ones. The web app uses the scikit-learn python library, to match with the input movie with corresponding movies in the dataset with the highest similarity score. Its frontend is designed using Flask and it is deployed on Heroku. We aim to demystify the magic behind a recommendation system and provide an introduction to natural language processing also on the way.**

Keywords— **Bag of Words, Flask, count Vectorizer, Movie Recommendation**

I. INTRODUCTION

"Personal beauty is a greater recommendation than any letter of reference", a quote by Fanny Brice, states that every human being has his/her own nuances. Their taste or views of various subject matters may differ. Recommendation systems aim to exploit those nuances and predict what kind of content the user might prefer. Looking at the vast variety and volume of movies that appear when streaming sites like Netflix or Amazon Prime, it is only natural that humans get confused about what to watch. In this situation, recommendation systems come into play.

Initially while browsing a movie streaming site, the user will select a movie and after watching it, they will submit feedback. Based on the positivity or negativity of the feedback provided, the recommendation system will suggest similar or different kinds of movies respectively. In the case of a new user, the system usually suggests the "top trending" movies, i.e the movies which currently have the highest views or are in great demand at the moment. This is known as collaborative filtering, i.e, making predictions about the behavior of a user by gathering information about many users together.

The frontend of the application is built using Flask, a Python framework. This framework is preferred over more popular frameworks like React and Angular, because of ease of integration of the backend of the application which is written in Python.

II. LITERATURE SURVEY

Over the years, as technology has progressed a lot of researchers have implemented movie recommendation systems using different techniques algorithms. These different projects were an inspiration for us to try and develop a system ourselves.

Ahuja et al [1], the researchers recommended movies using the machine learning algorithm K-Means Clustering in the year 2019. Liu et al [2] suggested another way that could be using collaborative filtering where movies were recommended based on their features. Nakhli et al [17] in 2019, proposed a simple way which used the correlation between the likes and percentage of views for the movies and filtering algorithms. Ifada et al [3] in the year 2018, proposed methods of handling the sparsity and scalability problems of the collaborative filtering approach. For handling sparsity, the approach estimates the rating entries by combining the similarities of the rating and their respective genres using relative weighting. The approach also uses fuzzy c-means clustering to handle the scalability problem.

Kim et al [4] in 2019, used recurrent Neural Networks which was applied to similar users with similar movie tastes by using Pearson's correlation coefficient to classify them. Zhang et al [6] recommended movies using the Markovian

International Journal of Engineering Applied Sciences and Technology, 2019
Vol. 4, Issue 5, ISSN No. 2455-2143, Pages 526-528
Published Online September 2019 in IJEAST (http://www.ijeast.com)

Factorization of Matrix Processes which adapt to a wide range of collaborative filtering problems. Recently, the LSIC model was proposed by Zhao et al [7], which gives the user a ranked list of movies. This model employs a framework to combine the Matrix Factorization and the RNN model for recommending movies.

Aesthetics (posters or still frames of the movies) are rarely involved in algorithms or rarely used in recommendation systems. In 2019, a method was proposed to integrate the aesthetics in the movie recommendation system. The CNN and aesthetic features were first integrated into probabilistic matrix factorization. A framework with these features was established to recommend movies by Chen et al [8].

III. PROPOSED ALGORITHM

Our application uses the concept of cosine similarity, a simple but elegant methodology to represent similarity between two entities in a numerical value between 0 and 1, 0 meaning the entities are absolutely different, and 1 meaning both are identical. The formula for deriving cosine similarity is as follows.

$$\cos(\mathbf{t},\mathbf{e}) = \frac{\mathbf{te}}{\|\mathbf{t}\|\|\mathbf{e}\|} = \frac{\sum_{i=1}^{n} \mathbf{t}_i \mathbf{e}_i}{\sqrt{\sum_{i=1}^{n} (\mathbf{t}_i)^2}\sqrt{\sum_{i=1}^{n} (\mathbf{e}_i)^2}}$$

Fig 1. Cosine Similarity Formula

Here we aim to find the similarity between vectors t and e.

We have worked on the IMDB dataset, to generate movie recommendations based on the input movie by the user. The dataset has many columns that can be factored into the recommendations, but we have taken only the plot, actors and directors for more accuracy. First, the selected columns are cleaned by removing punctuations and extra spaces. Then, keywords are extracted from the plot, which replaces the 'Plot' column. Finally, the bag of words model is implemented by combining the actors, directors, and the keyword plots into a single column.

The Big Sleep	crime film-noir mystery howardhawks humphrey...
Pink Floyd: The Wall	animation drama fantasy alanparker bobgeldof...
The King's Speech	biography drama tomhooper colinfirth helenabo...
A Christmas Story	comedy family bobclark melindadillon darrenmc...
The Graduate	comedy drama mikenichols annebancroft dustinh...

Fig 2. The bag of words model

After the above table is generated, we count the frequency of each word in each row with the help of the countVectorizer. And form a cosine similarity matrix with the help of that. A cosine similarity matrix looks at the frequency of common word between two movies and gives a rating from 0 to 1 accordingly. So basically, each movie, when compared with itself, will have a rating of 1. Hence, when a title is passed onto the recommender, it first finds the index of the movie located in the similarity matrix. After that, it sorts all the values of that row in decreasing order and returns the first ten movies as recommendations.

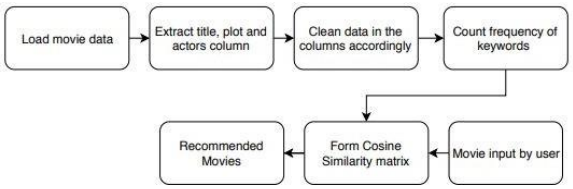

Fig 3. Workflow of the application

The frontend of the application is built using Flask, and the application is deployed using Heroku, a developer-friendly deployment tool.

Fig 4. Heroku[9]

IV. EVALUATION

Upon comparing the recommended movies with the user input, the results are satisfactory. That is, the recommended movies have at least something common between them and the input movie in terms of director, plot or actors.

For example, on input of the film "The Departed", the recommendations are ['Fargo', 'Reservoir Dogs', 'Heat', 'Shutter Island', 'The Wolf of Wall Street', 'Rope', 'Blood Diamond', 'Goodfellas', 'On the Waterfront', 'Chinatown']. And each of the movies have something in common with the input film. For example, both "The Wolf of Wall Street" and "The Departed" have Leonardo DiCaprio as the lead actor and both "The Departed" and "Reservoir Dogs" are of the same genre i.e drama.

V. CONCLUSION AND FUTURE WORK

A study has shown that the total amount of time a person spends deciding which movie to watch due to the huge number of choices, that time could be utilized in

watching at least half of a full-length movie. Hence an average customer would be saving about an hour of his time every week, doing something more productive.

The user interface has been designed to promote user interactivity with the application. Due to the simple and compact UI, future changes can be made in the frontend with ease

Future work could focus on eliminating the limitation of the IMDB dataset. Only movies available in the dataset could be evaluated, which proves to be a restriction if the product were to be distributed among consumers.

VI. ACKNOWLEDGEMENT

We would like to thank everyone who helped make this project possible. Without their support and guidance, this project would not be such a success.

VII. REFERENCE

[1] Ahuja, R., Solanki, A., & Nayyar, A. (2019, January). Movie Recommender System Using K-Means Clustering AND K-Nearest Neighbor. In *2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence)* (pp. 263-268). IEEE.

[2] Liu, G., & Wu, X. (2019, March). Using Collaborative Filtering Algorithms Combined with Doc2Vec for Movie Recommendation. In *2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)* (pp. 1461-1464). IEEE.

[3] Ifada, N., & Prasetyo, E. H. (2018, November). Employing Sparsity Removal Approach and Fuzzy C-Means Clustering Technique on a Movie Recommendation System. In *2018 International Conference on Computer Engineering, Network and Intelligent Multimedia (CENIM)* (pp. 329-334). IEEE.

[4] Kim, M., Jeon, S., Shin, H., Choi, W., Chung, H., & Nah, Y. (2019, June). Movie Recommendation based on User Similarity of Consumption Pattern Change. In *2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)* (pp. 317-319). IEEE.

[5] Hossain, M. A., & Uddin, M. N. (2018, September). A Neural Engine for Movie Recommendation System. In *2018 4th International Conference on Electrical Engineering and Information & Communication Technology (iCEEiCT)* (pp. 443-448). IEEE.

[6] Zhang, R., & Mao, Y. (2019). Movie Recommendation via Markovian Factorization of Matrix Processes. *IEEE Access*, *7*, 13189-13199.

[7] Zhao, W., Wang, B., Yang, M., Ye, J., Zhao, Z., Chen, X., & Shen, Y. (2019). Leveraging Long and Short-Term Information in Content-Aware Movie Recommendation via Adversarial Training. *IEEE transactions on cybernetics*..

[8] Chen, Xiaojie, Pengpeng Zhao, Yanchi Liu, Lei Zhao, Junhua Fang, Victor S. Sheng, and Zhiming Cui. "Exploiting Aesthetic Features in Visual Contents for movie recommendations." IEEE Access 7(2019):49813-49821

[9] For deployment purposes, Heroku documentation was referred: https://devcenter.heroku.com/categories/reference

[10] Dhamecha, M., Dobaria, K., & Patalia, T. (2019, March). A Survey on Recommendation System for Bigdata using MapReduce Technology. In *2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)* (pp. 54-58). IEEE.

[11] Haruna, K., Ismail, M. A., Suyanto, M., Gabralla, L. A., Bichi, A. B., Danjuma, S., ... & Herawan, T. (2019). A Soft Set Approach for Handling Conflict Situation on Movie Selection. *IEEE Access*.

[12] Fessahaye, F., Perez, L., Zhan, T., Zhang, R., Fossier, C., Markarian, R., ... & Oh, P. (2019, January). T-RECSYS: A Novel Music Recommendation System Using Deep Learning. In *2019 IEEE International Conference on Consumer Electronics (ICCE)* (pp. 1-6). IEEE.

[13] He, M., Wang, B., & Du, X. (2019). HI2Rec: Exploring Knowledge in Heterogeneous Information for Movie Recommendation. *IEEE Access*, *7*, 30276-30284.

[14] For frontend, Flask guide was used for reference http://flask.palletsprojects.com/en/1.1.x/

[15] Zhang, J., Wang, Y., Yuan, Z., & Jin, Q. (2019). Personalized real-time movie recommendation system: Practical prototype and evaluation. *Tsinghua Science and Technology*, *25*(2), 180-191.

[16] Pandas was used for data cleaning https://pandas.pydata.org/pandas-docs/stable/

[17] Nakhli, R. E., Moradi, H., & Sadeghi, M. A. Movie Recommender System Based on Percentage of View. In *2019 5th Conference on Knowledge Based Engineering and Innovation (KBEI)* (pp. 656-660). IEEE.