Movie Recommendation System with Collaborative Filtering using Average Weighted Rating

Rishabh Kumar Rai[1], Gaurav Maddhesiya[2], Saurabh Sakal Singh[3]

[1]B. Tech Student, CSE, Galgotias University, Greater Noida, Uttar Pradesh
[2]B. Tech Student, CSE, Galgotias University, Greater Noida, Uttar Pradesh [3] B. Tech Student, CSE, Galgotias University, Greater Noida, Uttar Pradesh

---***---

Abstract -*Recommendation system can be defined as a system that produces individual recommendations (a personalized way of possible options) as an output based on their previous choices which are considered an input by the system. Many products that we use today are a result of recommendation system music, news, books, items etc. However, in this paper we would be discussing about a movies recommender system on Model based collaborative filtering and Content-based filtering while comparing the accuracy of each recommendation system.*

***Key Words*: Recommendation system, collaborative Filtering, content-based filtering.**

1. INTRODUCTION

Most popular version is mass customization for selection of entities that we want. The Current recommendation systems such as content-based filtering and collaborative filtering use different information sources to make recommendations. Content-based filtering, makes recommendations based on user preferences for product features in this case the genre of the movie, the directors, the cast etc. Collaborative filtering mimics user-to-user recommendations as a weighted, linear combination of other user preferences. Both methods have limitations. Content-based filtering recommends new entities, by using n more data of user preference in order to assimilate best match. Similarly, collaborative filtering needs large dataset with active users who rated a product before in order to make accurate predictions. Combination of these different recommendation systems are called hybrid systems which can combine the features of the item with the preferences of other users. Presently existing services like IMDB do not personalize recommendations but provide an overall rating for a movie and work on average weighted values. This decreases the value of each recommendation significantly as individual movie preferences of the user are not catered. But our recommendation engine takes a collaborative social networking approach where one's own tastes are mixed with the entire community to generate meaningful results.

2. LITERATURE SURVEY

A recommendation system generally depends hugely on the inputs of a user and their relationship between the products.

2.1 Collaborative filtering

Collaborative algorithm uses "User Behavior" for recommending items. They exploit behavior of other users and items in terms of watch history, ratings, selection. Other users' behavior and preferences over the movies are used to recommend movies to the new users. However, including side features can be hard to include i.e., features beyond the query or item ID. For movie recommendations, the side features might include country or age. Inclusion of available side features helps to improve the quality of the model.

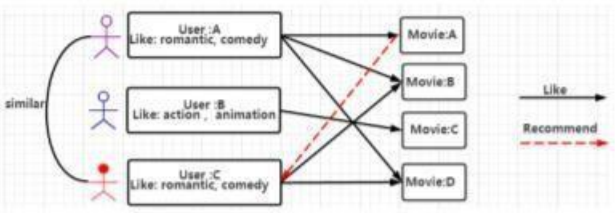
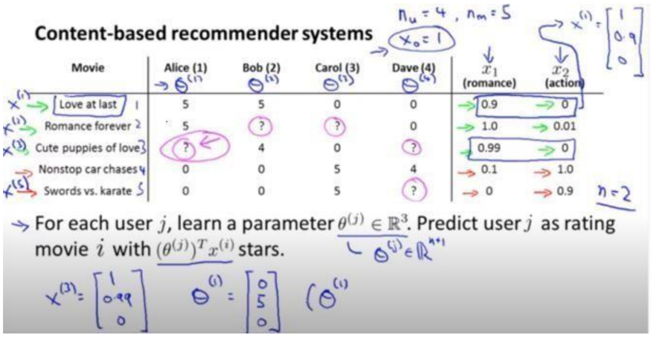


Fig-1: Example of Average weighted algorithm (Collaborative Filtering)

User A prefers movie genres A, B, C, and user C prefers to watch movie B, D, hence we can conclude that the likings of user A and user C are very similar. Since user A likes movie D as well, so we can deduce that the user A may also like item D, therefore item D would be recommended to the user. The general idea of the algorithm is constructed on records of past scores provided by the users. Find the neighbor user as a` who exhibits similar interest with target user a, and then suggests the items which the neighbor user a` preferred to target user a, the predicted score which the target user a may give on the item is obtained by the score calculation of neighbor user a` on the item.

2.2 Content Based Filtering

Content-based filtering uses movie features to recommend movies similar to what the user likes, based on their previous actions or explicit feedback. This makes the scaling easier for a large number of users. The model can capture the particular interests of a user, and on that basis can recommend niche movies that very few other users are interested in.
However, Since the feature representation of the items are hand-engineered on some aspects, this methodology requires a lot of domain knowledge that is why, this model can only be as good as the hand-engineered features. The model can only recommend anything based on interests of the user that already exists. In common words, the model has limited ability to expand on the users' existing interests. When we talk about Movie recommendation system, a recommendation system based on content can recommend a movie based on user data provided by them explicitly, following which a user profile is generated. This background knowledge is further used to make recommendations which becomes more accurate overtime. In content-based system the “concepts of Term Frequency and Inverse document frequency are used for filtering systems and information retrieval”. The prime use of these terms is to acquire the importance of any movie. Term frequency can be described as the number of times or the frequency of the word in a document.



Content-based recommender systems

Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	0.9	0
Romance forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	4	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs. karate	0	0	5	?	0	0.9

For each user j, learn a parameter $\theta^{(j)} \in \mathbb{R}^3$. Predict user j as rating movie i with $(\theta^{(j)})^T x^{(i)}$ stars.

$n_u = 4$, $n_m = 5$
$x_0 = 1$
$x^{(1)} = \begin{bmatrix} 1 \\ 0.9 \\ 0 \end{bmatrix}$
$\theta^{(1)}$ $\theta^{(2)}$ $\theta^{(3)}$ $\theta^{(4)}$
$x^{(1)}$ 1
$x^{(2)}$ 2
$x^{(3)}$ 3
4
$x^{(5)}$ 5
$n = 2$
$\theta^{(j)} \in \mathbb{R}^{n+1}$
$x^{(3)} = \begin{bmatrix} 1 \\ 0.99 \\ 0 \end{bmatrix}$
$\theta^{(1)} = \begin{bmatrix} 0 \\ 5 \\ 0 \end{bmatrix}$
$(\theta^{(1)}$

Fig-2: Example of Content Based Filtering.

3. ALGORITHM AND METHODOLOGY

In a movie recommender system, a movie is that entity which is considered and recommendation is done using similar entities. Using most relevant similar entities from a large number of datasets based on user’s query, recommendation can be given. Movies are rated, which helps in retrieving other entities that are more relevant based on popularity, authority, relevance, etc. Below stating output, input and data of the recommender and problem:

Input: A Movie.

Output: Recommender movies when given input.

Movie Data: Semi structured data and Unstructured data.

Movie: An object, structured data.

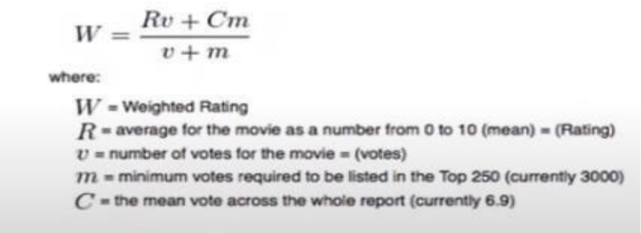
Mentioning below the challenges faced by Recommender:

Unstructured Dataset: Movie recommender includes storing and processing huge and unstructured data.
However, we have built a small- scale recommender engine consisting a dataset of 5000 movies provided by the TMDB dataset.

Movie Disambiguation and Movie Resolution:
“XYZ” may refer to actor as well as movie “XYZ”.

Movie Positions: For a certain Movie, the usermay not be interested in all the other movies. The results are needed to be positioned. This positioning (ranking) of movies can be based on average ratings given to a set of movies or the popularity of a particular genre or even both the criteria.

3.1 Equation



$$W = \frac{Rv + Cm}{v + m}$$

where:

W = Weighted Rating
R = average for the movie as a number from 0 to 10 (mean) = (Rating)
v = number of votes for the movie = (votes)
m = minimum votes required to be listed in the Top 250 (currently 3000)
C = the mean vote across the whole report (currently 6.9)

Fig-3: Average weighted Rating formula.

The above formula is a variation of the formula used by IMDB to rate its movie lists. It’s an effective formula that involves the consideration of a user’s perspective a provide clear recommendations thereafter.

4. TOOLS AND MERITS

1. Anaconda Navigator, Jupyter Notebook, Python.
2. Data Cleaning by using Python libraries like NumPy and Pandas.
3. Data and output Visualization by using Matplotlib and Seaborn libraries.
4. ML libraries like SK-learn to use Minmax Scaler for statistical modelling for transforming values of different magnitudes on a scale of 0 to 1 in this case the values of weighted average and the popularity.
5. We have tested the method using simulation of system on following requirements:

Laptop with core i5 processor, 8 GB RAM.

Merits of the system involves the following:

1. The model can help users in discovering new interests. In isolation, the Machine learning system may not know the interest of the user for a given item, but it might still recommend it because similar users are interested in that item.
2. To some extent, the system needs only the feedback matrix to train a matrix factorization model. In particular, the system doesn't need contextual features.
3. Even when no information on a movie is available, we still can predict the rating without waiting for a user to watch it.
4. Focusing solely on content does not provide any flexibility on the user's perspective and their preferences. Hence, this system focuses on the changing user interests over time.

5. ARCHITECTURE AND IMPLEMENTATION

5.1 Architecture Design

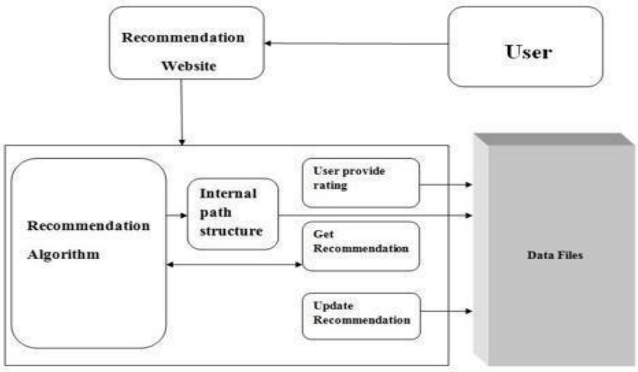
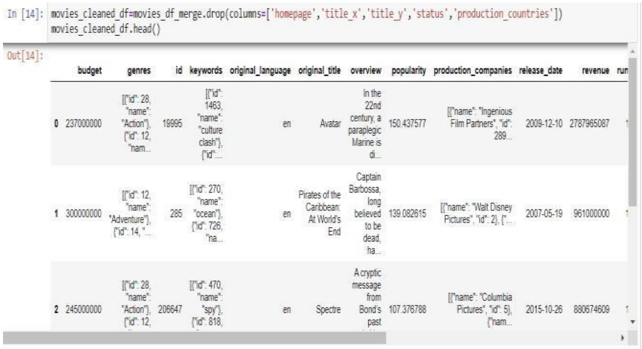


Fig-4: Architecture followed by the system.

5.2 Implementation of Modules with Code

Module 1:
Data Cleaning module comprises of the removal of those columns that are not required for calculation of our recommendations. As we can see factors like title of a movie, status, production countries and homepage do not play any role or give any idea about the interests of our users hence we have dropped such columns. The code and output of the data cleaning is given below:



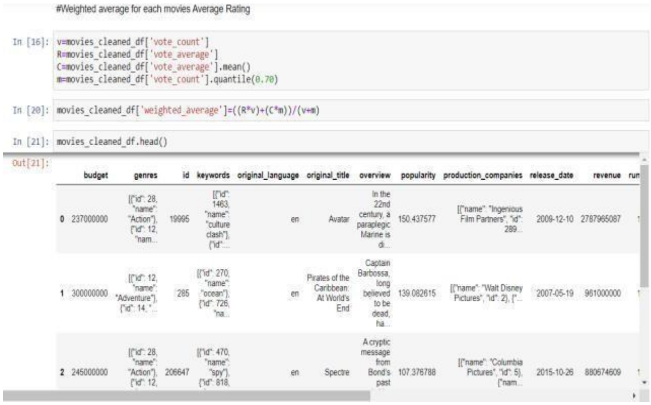
```
In [14]: movies_cleaned_df=movies_df_merge.drop(columns=['homepage','title_x','title_y','status','production_countries'])
         movies_cleaned_df.head()
```

Out[14]:

	budget	genres	id	keywords	original_language	original_title	overview	popularity	production_companies	release_date	revenue	run
0	237000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	19995	[{"id": 1463, "name": "culture clash"}, {"id":...	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150.437577	[{"name": "Ingenious Film Partners", "id": 289...	2009-12-10	2787965087	1
1	300000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "...	285	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	139.082615	[{"name": "Walt Disney Pictures", "id": 2}, {"...	2007-05-19	961000000	1
2	245000000	[{"id": 28, "name": "Action"}, {"id": 12,	206647	[{"id": 470, "name": "spy"}, {"id": 818,	en	Spectre	A cryptic message from Bond's past	107.376788	[{"name": "Columbia Pictures", "id": 5}, {"nam...	2015-10-26	880674609	1

Fig-5: Data Cleaning Module.

Module 2:
Under this module we have used the proposed weighted average formula to create corresponding lists. We have NumPy and Pandas libraries for calling out columns like vote_count, mean of vote_average column, a 0.70 quantile range of the vote counts.



#Weighted average for each movies Average Rating

```
In [16]: v=movies_cleaned_df['vote_count']
         R=movies_cleaned_df['vote_average']
         C=movies_cleaned_df['vote_average'].mean()
         m=movies_cleaned_df['vote_count'].quantile(0.70)

In [20]: movies_cleaned_df['weighted_average']=((R*v)+(C*m))/(v+m)

In [21]: movies_cleaned_df.head()
```

Out[21]:

	budget	genres	id	keywords	original_language	original_title	overview	popularity	production_companies	release_date	revenue	run
0	237000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	19995	[{"id": 1463, "name": "culture clash"}, {"id":...	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150.437577	[{"name": "Ingenious Film Partners", "id": 289...	2009-12-10	2787965087	1
1	300000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "...	285	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	139.082615	[{"name": "Walt Disney Pictures", "id": 2}, {"...	2007-05-19	961000000	1
2	245000000	[{"id": 28, "name": "Action"}, {"id": 12,	206647	[{"id": 470, "name": "spy"}, {"id": 818,	en	Spectre	A cryptic message from Bond's past	107.376788	[{"name": "Columbia Pictures", "id": 5}, {"nam...	2015-10-26	880674609	1

Fig-6: Application of Average Weighted Formula.

6. CONCLUSIONS

Under the condition of massive information, the requirements of movie recommendation system from film amateur are increasing. This article designs and implements a complete movie recommendation system prototype based on the Average weighted rating, collaborative filtering algorithm and recommendation system technology.

REFERENCES

1. Hu Jinming. "Application and research of collaborative filtering in e- commerce recommendation system", 2010 3rd International Conference on Computer Science and Information Technology, 07/2010
2. F. M. Harper and J. A. Konstan, “The MovieLens datasets: History and context,” vol. 5, no. 4, pp. 1–

19. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2866565.28 27872

3. DebdeepLazyCoder/data-analytics-assignment. [Online]. Available: https://github.com/DebdeepLazyCoder/Data-Analytics-Assignment
4. J. Le. The 4 recommendation engines that can predict your movie tastes. [Online]. Available: https://medium.com/@james aka yale/the- 4-recommendationengines-that-can-predict-your-movie-tastes- bbec857b8223.