Movie Recommender System Analysis Using Collaborative and Content-based Filtering

Sai Manvitha Nadella
Electronics and Communication Engineering Department
Sridevi Women's Engineering College
Hyderabad, India
manvithanadella12345@gmail.com

***Abstract*—A Recommender System is a machine learning system that provides suggestions to the users to identify interesting products and services when the amount and complexity of options overtakes the capability of a user to survey it and reach a decision. A movie recommender system predicts what movie the user will watch based on the characteristics of the previously watched or liked movies by the user itself or the other users who have watched the same film earlier. The factors like genre and the plot of the movie, actors and technicians worked for it and etc are considered while designing and implementing a movie recommender system. Recommender systems employ both collaborative filtering and content-based filtering (personality-based approach), as well as other systems such as knowledge-based systems. In this paper, the movie recommender system is developed using both Collaborative Filtering with the help of Matrix Factorization and Content-based Filtering with the help of Natural Language Processing, separately and both the systems are evaluated and compared based on their performance. The datasets used for the systems are IMDB_Top250Engmovies2_OMDB_Detailed and Movie Lens datasets. The data analysis tool used is Python 3.**

Keywords— Recommender Systems, Collaborative Filtering, Content-based Filtering, Matrix Factorization(MF), Natural Language Processing(NLP), Rapid Automatic Keyword Extraction(RAKE)

I. INTRODUCTION

In day to day life with the fastest growing technology, a user might come across millions of data, where the user might need only a set of data to view or engage with. A recommender system helps users find compelling content in a large corpora[1]. For example, a movie search engine provides many movies and apps like YouTube provides millions of videos, more videos are added every day, a recommendation system will display items that users might not find when searched. The Recommendation System is designed using either of the two common techniques, Collaborative Filtering(CF) or Content-based Filtering(CBF), where Content-based Filtering uses similarity between items to recommend items similar to what the user likes and Collaborative filtering uses similarities between queries and items simultaneously to provide recommendations. Some examples of recommender systems in action include product recommendation on Amazon, Netflix suggestions for movies and TV shows in feed, recommended videos on YouTube, music on Spotify, the Facebook newsfeed and Google Ads. Important component of any of these systems is the recommendation function which takes the information about the user and predicts the rating the user might assign to that product. In this paper, the movie recommendation system is designed using the both techniques of recommendation functions and are compared based on the performance and recommendation capabilities.

II. BACKGROUND

The architecture of recommendation systems commonly consists of three components: Candidate Generation, scoring and re-ranking. In candidate generation stage, the system starts with a huge data and generates a much smaller subset of candidates. For example, the candidate generator in YouTube reduces millions of videos to hundreds. The model needs to evaluate queries quickly given the enormous size of the data and a given model may provide multiple candidate generators, each nominating a different subset of candidates. Next, another model scores and ranks the candidates in order to select the set of items (on the order of 10) to display to the user. Since this model evaluates a relatively small subset of items, the system can use a more precise model relying on additional queries. Finally the system must consider additional constraints for the final ranking. For example, the system removes items that the user clearly disliked or increases the score of fresher content.[1]

Movie Recommender system employ the ratings of users for various movies and try to find other like-minded users and recommend those movies which are highly-rated. As discussed earlier, recommendation systems are mostly implemented using Collaborative filtering, Content-based filtering and hybrid filtering. Collaborative filtering systems have two approaches: memory based approach and model based approach. Memory based approaches continuously analyses user data in order to make recommendations. As the utilize the user ratings they gradually improve in accuracy over time[3]. Model based approaches develop a model of a user's behaviour and then use certain parameters to predict future behaviour[3]. In general model based collaborative filtering are implemented using Singular Value Decompositions(SVD) and memory based collaborative filtering by computing cosine similarity. Content-based filtering systems analyse documents or preferences given by a particular user, and attempt to build a model around this data. It uses the user's particular interests and attempt to match a user's profile to the attributes of the various content objects to be recommended. Content-based filtering systems are further of three methods—wrapper methods, filter methods, and embedded methods. Wrapper methods divide the features into subsets, run analysis on these subsets and then evaluate which of these subsets seems the most promising. Filter methods use heuristic methods to rate features on their content . Both these methods are independent of the algorithms used. In contrast, embedded methods are coupled with the algorithm used—feature selection is performed during the training phase. Wakil et al. proposed a recommendation system by filtering using emotions. When a user watches a certain type of film, certain

emotions are triggered from within them. In the same way, the emotions of a user can trigger the need to watch a certain type of film. They concluded that the traditional user profiles do not consider user's emotional status, and thus designed an algorithm that employs emotion determination. The algorithm analyses a colour sequence chosen by the user in unity with the emotions to determine current emotional state of the user.

III. MOVIE RECOMMENDATION SYSTEM USING CONTENT – BASED FILTERING

Content-based Filtering uses item attributes to recommend other items to what the user likes, based on their previous views or high rating feedbacks. This aims at recommending movies to users based on similarity between genres, plot and other key attributes. The dataset used to demonstrate this movie recommender system is IMDB_Top250Engmovies2_OMDB_Detailed.csv file. There are a total of 250 movies (rows) and 38 attributes (columns). However, only 5 attributes are useful: 'Title', 'Director', 'Actors', 'Plot', and 'Genre'. Below shows a list of 10 popular directors.(Fig. 1)

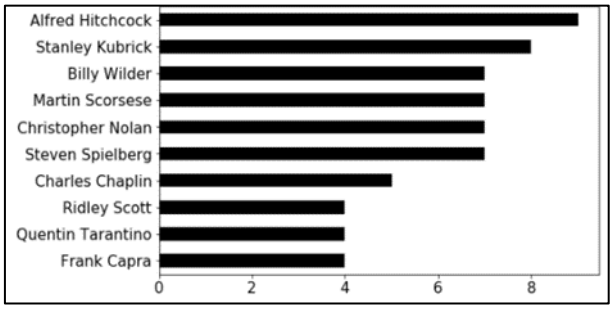

Fig. 1: Bar Plot of top 10 Popular Directors

The attributes such as main actors, director, genre and plot are combined to calculate its similarity with other movies. The data has to be pre-processed using Natural Language Processing(NLP) to obtain a single column which has all the attributes (in bag of words) of each movie. Then, these bag of words is converted into numerical by vectorization, where scores are assigned to each word. Subsequently cosine similarities can be calculated. The RAKE(Rapid Automatic Keyword Extraction) function from rake_nltk library is used to extract the most relevant words from sentences in the 'Plot' column. This function is applied to each row under the 'Plot' column and assigned the list of key words to a new column 'Key_words'(Fig. 2).

Then the names of actors, directors and genre of the movies are transformed into unique values. This is done by merging all first and last names of actors and directors into one word. Every word needs to be converted to lowercase to avoid duplications.(Fig. 4)

The 4 columns 'Genre', 'Director', 'Actors' and 'Key_words' are combined into a new column 'Bag_of_words' and the final data frame to check similarity contains only the Title and Bag_of_words columns.(Fig. 3)

	Title	Bag_of_words
0	The Shawshank Redemption	crime drama frankdarabont timrobbins morganfre...
1	The Godfather	crime drama francisfordcoppola marlonbrando al...
2	The Godfather: Part II	crime drama francisfordcoppola alpacino robert...
3	The Dark Knight	action crime drama christophernolan christianb...
4	12 Angry Men	crime drama sidneylumet martinbalsam johnfiedl...
...	...	...
245	The Lost Weekend	drama film-noir billywilder raymilland janewym...
246	Short Term 12	drama destindanielcretton brielarson johngalla...
247	His Girl Friday	comedy drama romance howardhawks carygrant ros...
248	The Straight Story	biography drama davidlynch sissyspacek janegal...
249	Slumdog Millionaire	drama dannyboyle loveleentandan devpatel saura...

250 rows × 2 columns

Fig. 3: 'Bag_of_words' formed by combining all the attributes

Vector Representation

The recommender model can only read and compare a vector (matrix) with another, the 'Bag_of_words' are converted into vector representation using CountVectorizer model from Sci-kit Learn, which is a simple frequency counter for each word in the 'Bag_of_words' column. When the count matrix is obtained for all the words then the cosine_similarity function can be applied to compare the similarities between movies.

$$similarity = \cos(\theta) = \frac{u.v}{|u||v|}$$

$$= \frac{\sum_{i=1}^{n} u_i v_i}{\sqrt{\sum_{i=1}^{n} u_i^2}\sqrt{\sum_{i=1}^{n} v_i^2}}$$

$$u.v = [u_1 u_2 \ldots . u_n] \cdot \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} = u_1 v_1 + u_2 v_2 + \ldots + u_n v_n = \sum_{i=1}^{n} u_i v_i$$

The cosine similarity matrix obtained after applying CountVectorizer() function is shown below:

Cosine Similarity Matrix

```
[[1.         0.15789474 0.13764944 ... 0.05407381 0.05407381 0.05564149]
 [0.15789474 1.         0.36706517 ... 0.05407381 0.05407381 0.05564149]
 [0.13764944 0.36706517 1.         ... 0.04714045 0.04714045 0.04850713]
 ...
 [0.05407381 0.05407381 0.04714045 ... 1.         0.05555556 0.0571662 ]
 [0.05407381 0.05407381 0.04714045 ... 0.05555556 1.         0.0571662 ]
 [0.05564149 0.05564149 0.04850713 ... 0.0571662  0.0571662  1.        ]]
```

```
df['Plot']
0      Two imprisoned men bond over a number of years...
1      The aging patriarch of an organized crime dyna...
2      The early life and career of Vito Corleone in ...
3      When the menace known as the Joker emerges fro...
4      A jury holdout attempts to prevent a miscarria...
                             ...
245    The desperate life of a chronic alcoholic is f...
246    A 20-something supervising staff member of a r...
247    A newspaper editor uses every trick in the boo...
248    An old man makes a long journey by lawn-mover ...
249    A Mumbai teen reflects on his upbringing in th...
Name: Plot, Length: 250, dtype: object

df['Key_words']
0      [eventual, redemption, two, imprisoned, men, b...
1      [reluctant, son, organized, crime, dynasty, tr...
2      [tightens, 1920s, new, york, grip, early, life...
3      [people, mysterious, past, physical, tests, gr...
4      [jury, holdout, attempts, miscarriage, forcing...
                             ...
245    [chronic, alcoholic, fourday, drinking, bout, ...
246    [20something, supervising, staff, member, long...
247    [remarrying, keep, book, ace, reporter, exwife...
248    [long, journey, relationship, old, man, makes,...
249    [wants, cheating, millionaire, upbringing, slu...
Name: Key_words, Length: 250, dtype: object
```

Fig. 2: 'Key_words' extracted from 'Plot' column

	Title	Director	Actors	Plot	Genre	Key_words
0	The Shawshank Redemption	[frankdarabont]	[timrobbins, morganfreeman, bobgunton]	Two imprisoned men bond over a number of years...	[crime, drama]	[eventual, redemption, two, imprisoned, men, b...
1	The Godfather	[francisfordcoppola]	[marlonbrando, alpacino, jamescaan]	The aging patriarch of an organized crime dyna...	[crime, drama]	[reluctant, son, organized, crime, dynasty, tr...
2	The Godfather: Part II	[francisfordcoppola]	[alpacino, robertduvall, dianekeaton]	The early life and career of Vito Corleone in ...	[crime, drama]	[tightens, 1920s, new, york, grip, early, life...
3	The Dark Knight	[christophernolan]	[christianbale, heathledger, aaroneckhart]	When the menace known as the Joker emerges fro...	[action, crime, drama]	[people, mysterious, past, physical, tests, gr...
4	12 Angry Men	[sidneylumet]	[martinbalsam, johnfiedler, leej.cobb]	A jury holdout attempts to prevent a miscarria...	[crime, drama]	[jury, holdout, attempts, miscarriage, forcing...

Fig. 1: Names of actors, directors and genre of the movies are transformed into unique values

The movie recommendations can be obtained by creating a function that takes in a title as input, and returns the top 10 similar movies. This function will match the input title with the corresponding index of the Similarity Matrix, and extract the row of similarity values in descending order. The top 10 similar movies are found by extracting the top 11 values and subsequently eliminating the first index (which is the input title). The outputs of recommendations given to the user are:

```
['The Dark Knight Rises',
 'Batman Begins',
 'The Green Mile',
 'Witness for the Prosecution',
 'Out of the Past',
 'Rush',
 'The Prestige',
 'The Godfather',
 'Reservoir Dogs',
 'V for Vendetta']
```

Fig. 5; User 1 recommendations for 'The Dark Knight'

```
['No Country for Old Men',
 'The Departed',
 'Rope',
 'The Godfather',
 'Reservoir Dogs',
 'The Godfather: Part II',
 'On the Waterfront',
 'Goodfellas',
 'Touch of Evil',
 'The Big Lebowski']
```

Fig. 6; User 2 recommendations for 'Fargo'

The model has recommended very similar movies. Some similarities can be observed mainly based on directors and plot. Hence, the model performed well and recommendations are linearly satisfied.

IV. MOVIE RECOMMENDATION SYSTEM USING COLLABORATIVE FILTERING WITH MATRIX FACTORISATION

Collaborative filtering is one of the most widely used techniques for recommendation systems. CF is a popular recommendation algorithm that bases its predictions and recommendations on ratings or opinions of other users in the system. Predictions about user interests are obtained by collecting view experiences from many other similar users. The Movielens dataset is used in demonstrating this recommendation system, containing 855,598 ratings for 10197 movies and 2113 users. The ratings data represent as a list of userID, itemID and rating. The dataset is split into two parts, 80% for training and 20% for testing the algorithm. This dataset contains actors, countries, directors and genres. For demonstration purpose, only the data of rating from users to make recommendations is used. The preview of the dataset is shown in Fig. 7.

userID	movieID	rating	date_day	date_month	date_year	date_hour	date_minute	date_second
75	3	1	29	10	2006	23	17	16
75	32	4.5	29	10	2006	23	23	44
75	110	4	29	10	2006	23	30	8
75	160	2	29	10	2006	23	16	52
75	163	4	29	10	2006	23	29	30
75	165	4.5	29	10	2006	23	25	15

Fig. 7: Movielens Dataset

The main focus is on the userID, movieID and rating. These fields will be converted into matrix two-dimensional space, where first dimension is the number of users and second dimension is the number of films. The values are represented with numbers. This recommendation system will focus on the film and the user entity. Movie data, and user ratings are available in various data formats. Each file consists of fields that are interrelated and can be extracted to form a matrix notation that are used to make recommendation system. Table.2 shows the matrix notation that is used in recommendation system.

TABLE 1: Matrix Notation

User/item	Film1	Film2	Film3	Film-n
User 1	1	-	2	-
User 2	2	3	-	-
User 3	-	2	1	3
User 4	5	4	-	1
User 5	3.5	1.5	5	2
User 6	2.5	2	3	1
User-n	1	3	-	1

From above table, rows represent the number of users and columns represent the number of movies. Whereas, the cell represented the rating value that user given The missing values are found using matrix factorisation from matrix notation.

Matrix Factorisation

The data rating from Movielens range from 1 until 5 (0, if no rating). The normalization technique is used to remove

the bias. The common method of normalization involves having values of each feature ranging from 0 to 1. After the data is normalized, then matrix factorization is performed to get predictions and recommendations.

Generally, matrix factorization is used to remove the dimension of an item space and acquire latent relations between items of the dataset [*9*][*10*]. Recommendation system have a large data and is very spare, this decreases the process functionality and inaccuracy in computational predictions. So therefore, it needs special techniques so as to minimize the data and increase speed of the computation process, which can be done by matrix factorization [*9*]. There are various methods used to process matrix factorisation, such as Singular Value Decomposition (SVD), Non-negative Matrix Factorization (NMF). SVD approach is used in this paper, it factorizes matrix rating to three low dimensional space, left-singular vector (V), singular value (S) and right-singular vector (W)[7].

$$H_k = V_{mxk}.S_{kxk}.W^T_{kxn}$$

Multiple *k* examples are used in the prediction and to investigate the result for next predictions.

The prediction value and original rating are compared in order to get the idea of how accurate the algorithm works. To predict the rating(r_{ui}), SVD class reconstructs the original matrix

$$M' = U\sum_k V^T$$

The rating prediction equals to:

$$rating(u,i) = M'ij$$

The approximation of predictions of rating are very close to the original rating values, and also result in some predictions of unknown values or missing values. Neighbourhood algorithm uses the ratings of the similar users (or items) to predict the values of the input matrix[7]. SVD is differed from other algorithms by its computational procedure. To compute the prediction, the below equation is used:

$$rating\ (u,i) = \frac{\sum_{j \in S^k(i;u)} S_{ij} r_{uj}}{\sum_{j \in S^k(i;u)} S_{ij}}$$

Where $S^k\ (i;\ u)$ is the set of '*k'* that gives rating by '*u'*, which are much similar to '*i'*. S_{ij} is similarity between *i* and *j*. Another method that is used in the implementation of this recommendation system is the nearest neighbour approach. Nearest neighbour is a classical collaborative filtering technique that is still used to design recommendation systems. This technique predicts the value and compares it with matrix factorization. It is performed to find the similarity with the films and users. From the results obtained it is concluded that the combination of matrix factorization and nearest neighbour can improve the accuracy in prediction for better recommendation system.

From many tests of *k* , the best of RMSE and MAE obtained, respectively are 0.78 and 0.59 with *k* values 20 and 35. The results metrics which are close to 0 are the most reliable. After performing with matrix factorization method, the next experiments will be performed by using the combination of matrix factorization and collaborative filtering. The neighbourhood algorithm is used for this purpose, which uses the similarity values of user ratings to predict the value of the input matrix. When compared with the matrix factorization (MF) approach in previous step, the combination of MF & CF has a better accuracy rate, and is proven by the lesser RMSE value close to 0. For the matrix factorization method, the smallest RMSE value is 0.789649, for which k is 35. While for MF + CF method, the smallest RMSE value was 0.788649 with k equals to 30.

V. CONCLUSION

Content-based filtering perform better than user collaborative filtering. The recommended movies are more similar and make more value than users similarities. The experiment showed that if a user liked a movie A before, then the next recommendation should be movie D, since it scored the highest. However, if the recommendation is based on other users watchlist based on similarities, then it showed that users didn't like movie A and movie C. Based on their weighed scores, it showed that movie E would be the best recommendation. The advantage about content based approach that one does not need data about other so as to make recommendations. The disadvantage with collaborative filtering is that when a user has a unique taste, the recommendation system might not recommend a movie to that particular user. And, the content based approach can be build based on user and movie characteristics and profile. Items can be recommended based on previously watched movies. However, if a user never rated a movie when watched, it won't appear in the recommendations list. The best approach would be to use a combination of different approaches. Mix of collaborative and content based filtering. Some of it will depend on preferences of the users and some on item features.

REFERENCES

[1] Recommendation System: developers.google.com/ machine-learning/recommendation/overview

[2] Paul.Covington, Jay Adams, Emre Sargin: Deep Neural Networks for YouTube Recommendations

[3] SRS Reddy, Sravani Nalluri, Subramanyam Kunisetti, S. Ashok and B. Venkatesh: Content-Based Movie Recommendation System Using Genre Correlation, S. C. Satapathy et al. (eds.), Smart Intelligent Computing and Applications, Smart Innovation, Systems and Technologies 105

[4] Bhatt, B.: A review paper on machine learning based recommendation system. Int. J. Eng. Dev. Res. (2014)

[5]Wakil, K., et al.: Improving web movie recommender system based on emotions. (IJACSA) Int. J. Adv. Comput. Sci. Appl. 6(2) (2015)

[6]Jakob Ivarsson and Mathias Lindgren: Movie recommendations using matrix factorization, 2016

[7] Mirza Ilhami, Suharjito: Film Recommendation Systems using Matrix Factorization and Collaborative Filtering, International Conference on Information Technology Systems and Innovation (ICITSI) 2014

[8] Yehuda Koren, Robert Bell, Chris Volinsky, "Matrix Factorization Techniques for Recommender Systems.," in IEEE Computer Society., 2009

[9] Xiaoyuan Su and Taghi M Khoshgoftaar, "A Survey of Collaborative Filtering Techniques," Journal Advances in Artificial Intelligence, 2009