Movie Recommender Systems

Shloak Agarwal **Vishesh Kumar Dwivedi**
University of Texas Arlington
shloak.agarwal@mavs.uta.edu visheshkumar.dwivedi@mavs.uta.edu

ABSTRACT

This paper is all about movie recommendation system using two type of filtering approach, content and collaborative. For training the data, various dataset is used. IMDB 250-movies, Kaggle 14000-movies and Movie lens 9000-movies is used as datasets.

INTRODUCTION

Now a days the recommendation system is a crucial part of one's life. At some point each one of us must have wondered where all the recommendations that Netflix, Amazon, Google give us, come from. We often rate products on the internet and all the preferences we express and data we share (explicitly or not), are used by recommender systems to generate, in fact, recommendations. The two main types of recommender systems are either collaborative or content-based filters.

DATASET DESCRIPTION

Dataset used in the paper is taken from IMDB, Kaggle and Movie lens.

Kaggle: The IMDB Movies Dataset contains information about 14,762 movies. Information about these movies was downloaded for the purpose of creating a movie recommendation app. The data was preprocessed and cleaned to be ready for machine learning applications. The content of the dataset is described as follows: title, "wordsInTitle", "url", "imdbRating", "ratingCount", "duration", "year", type, "nrOfWins", "nrOfNominations", "nrOfPhotos, nrOfNewsArticles", "nrOfUserReviews", "nrOfGenre". The rest of the fields are dummy (0/1) variables indicating if the movie has the given genre: Action, Adult, Adventure, Animation, Biography, Comedy, Crime, Documentary, Drama, Family, Fantasy, FilmNoir, GameShow, History, Horror, Music, Musical, Mystery, News, RealityTV, Romance, SciFi, Short, Sport, TalkShow, Thriller, War and Western.

Movie lens: The dataset is grouped by Group Lens Research. The dataset contains 100,000 ratings and 3,600 tag applications applied to 9,000 movies by 600 users. Last updated 9/2018.

IMDB: Dataset containing the top 250 top rated movies from IMDB. This table gathers, in fact, 250 observations (the movies) and 38 columns.

PROJECT DESCRIPTION

The movie recommender systems use two type of filtering approaches, one is content based filtering approach and other is collaborative type filtering approach. The content-based filtering approach is getting recommendation on base of genre or plot of the movies whereas the collaborative-based filtering approach uses rating from the users and then recommend movies.

For content-based filtering approach, only the user's watch history is required. Based on what user like, the algorithm will simply pick items with similar content to recommend them. In content-based filtering systems we can use additional information from the users watch history, for example, from the watched movie list, we can use information like name of actress, directors, and the plot of the movie, genre and the category. So basically, the purpose of content-based approach is to use the available features, from the observed user item interaction. For example, using this model we can make new prediction easily because we can watch the user's profile where the description of items is given. In our case, the items mean the movies watched by the users. Using the important information, the tasks will be to determine the relevant movies to recommend which can be done using this filtering model. Especially for content-based filtering method, we don't need information of the other users.

For collaborative-based filtering approach, the filter is based on users' rates, and it will recommend movies that user haven't watched yet, but other users similar to them have, and liked. For example, in the movie dataset, for this approach, the user's rating plays major role. So, with respect to collaborative-based filtering approach, the user rates are very crucial. However, the users don't rate the movies constantly, which can affect the recommendation results. Another property of this method is the diversity of the recommendation. The recommendations can be good or bad. One of the factors is number of users rating per movie. The other factors are different user item attractions and number of users. As we know every user likes different genres of movies. Let take an example, user A likes action and comedy movies. User B also likes actions movies but never watch comedy movies. In this case the collaborative-based filtering approach can recommend comedy movies to user B, based on the common liking of the two users for the action movies. This situation can go two ways. Either the user B will like the comedy movie a lot. In this case the recommendation through this approach is given the positive result, but there can be a case where user B does not like the comedy movies thus making the recommendation un-successful.

On the other hand, collaborative-based filtering system gives the broader area to achieve better recommendation results using every user item interaction. If the filtering system has more data, then there is the possibility to get better accuracy of the model. Analyzing more user item interactions and comparing different user's taste of the movies. This filtering approach has the capability to give appropriate results.

First, we used content-based filtering algorithm for Movie recommendation system. For recommending the movies to the user, we focused on the features given in the dataset. We took Plot, Genre, Directors, and Actors of the movie as features and processed that data to give proper results to the user. Here our goal was to know how these filtering approaches work. In our project, we are making a recommender system using Natural Language Processing. We are using *nltk* package for data processing. As we wanted to create a column which will contain all the keywords which will help in recommending the movies. For getting the key words, we used the *nltk* package for extraction of words from prime features of dataset. The extraction of key words is a part of data cleaning process. We used rake function from *nltk* package for extracting words from the featured columns. At first, we lowercased every single word in the required columns to avoid duplication. We removed the spaces between the words in the Director and Actor name columns. That is how we completed the data cleaning process.

We used rake function from nltk package to extract key words from the plot column and made new column to store the collection of key words. After that for frequency count of the collection of keywords we used "CountVectorizer" as the vectorizer gives importance to every single word in the column. CountVectorizer helps in having count of every single word. So, it helped in counts of every single key word.

Now, we want to know the similarity between movies using the collection of key words. So, we used "Cosine similarity" function. This function returns value of range 0 to 1. The similarity function returning value 1 means the movies are identical. Using the cosine similarity function, we created the similarity matrix with movies as rows and columns. The matrix containing the cosine similarity value where the diagonal values are 1 because the movie is identical to itself. Now we were able to give results of similar movies using the similarity values. We used the similarity values and made the collection of values and sort it in descending order so that we get the list of movies, with best similarity values. Hence, we gave proper recommendation results using content-based filtering approach.

After the content-based filtering approach was implemented successfully, we started working on the collaborative-based filtering approach. The Movie Lens was used in this approach. This dataset content the user rating of individual user towards the individual different movies. We have created a data frame between the average rating of each movie and number of rating. The purpose was to calculate the correlation between the movies. The correlation is described further in the paper. Once the correlation matrix between the movies was calculated, we put the threshold between the numbers of rating because there might different number of users providing different rating. For example, for a particular movie, the ratings have only been provided by two users and both have provided 5-star rating which makes the movie highly recommended. The correlation matrix is calculated using the Pearson correlation coefficient for giving proper recommendation of the movies. The correlation matrix value ranges form –1 to 1 (discussed further in the paper). Finally, the recommended movies (top 10) is selected form the matrix and is printed with the correlation and number of rating of recommended movies.

PERFORMANCE

When we talk about the accuracy of the recommendation system, the accuracy percentage is not the only aspect. We have to consider different evaluation metrics when we calculate the accuracy of the model and similarly the analysis of the model also plays the major role here. With respect to the different evaluation metrics and analysis of the model, we were quite different from the main project of the references. The description of the accuracy was not given with results in the references while we analyzed and calculated the accuracy of the model using variety of evaluation factors. The **overall accuracy achieved is 95.858%**.

We used different evaluation metrics like correlation coefficient, threshold factor, category diversity, variation factor, cosine similarity, Root Mean Square Error and Mean Square Error. Using these different evaluation metrics, we tested our algorithms and final model. Using this approach, we were able to find the accuracy of the model as well as we were able to find the accuracy of every single recommended movie from the recommendation list. For example, in the correlation coefficient, the strong and moderate correlation value range is from 0.30 to 1.00 and with threshold factor of the rate counts we were able to get good results from the model. We tested the model with different movie name inputs and got the recommendation list whose variation factor value was low which told us that the all movies from the list are relevant predictions

In addition, our data cleaning approach and techniques were quite different. As we tested our model with various scales of datasets. We tested our model on large datasets, some datasets like IMDB dataset of 14000 movies, were quite complex datasets and we used and tested the recommendation model on it. The cleaning and merging approach on that dataset, merging the one hot encoded 22+ columns of the dataset was a learning for us and one of the most difficult tasks to do. On the datasets of different scales, performing NLP methods and making sure of giving proper prediction, recommendation in our case, really helped us to know more about the recommendation model and the filtering approaches.

List of contribution for the paper:

1. Implementation from scratch of the Content Based Filtering Algorithm
2. Implementation from scratch of the Collaborative Based Filtering Algorithm
3. Comparison and analysis of both the filtering algorithms.
4. Using Natural Language Processing Tool Packages for data cleaning and processing.
5. Analysis of the recommendation model using variety of Evaluation metrics
6. Calculating the variation factor between the recommended movies in a list.
7. Testing the model multiple times and calculating the accuracy of the recommendation model.
8. Used variety of Datasets. The different scales of datasets with different feature columns.
9. Providing the overall accuracy of 95.868% of the model, increase of around 8 to 10% with respect to the given references.
10. Achieved appropriate accuracy using the threshold factor on the featured columns.

ANALYSIS

We faced many difficulties while developing the algorithm. One of the difficulties faced was in the content-based approach, while implementing *rage* function. We have used *nltk* package to allow the extraction of the key works form the text and assign scores to each word. For example, if the plot of the movies is explained, then the *rage* function extracts the essential keywords from the plot column based on the most relevant words in the text.

Other challenge we faced was of the cosine similarity used in content-based filtering approach. In order to detect the similarity between the movies, we used frequency counter for each of the words in the bag of words generated from the *rake* function. Once the matrix is formed that contains the count of each word, cosine similarity function is applied. Cosine similarity is the basically the dot product between two vectors is equal to the projection of one of them on the other. This will be in fact equal to 1 if the two vectors are identical, and it will be 0 if the two are orthogonal. In other words, the similarity is a number bounded between 0 and 1 that tells us how much the two vectors are similar.

Other challenge faced was of correlation coefficient in the collaborative-based filtering approach. The ratings to calculate the correlation between the movies later. Correlation is a statistical measure that indicates the extent to which two or more variables fluctuate together. Movies that have a high correlation coefficient are the movies that are most similar to each other. This number will lie between -1 and 1. 1 indicates a positive linear correlation while -1 indicates a negative correlation. 0 indicates no linear correlation. Therefore, movies with a zero correlation are not similar at all.

Other challenge faced was allocation if he threshold for the number of rating in the collaborative-based filtering approach. In that some of the movies have very few ratings and may end up being recommended simply because one or two people gave them a 5-star rating. We have fixed this by setting a threshold for the number of ratings. We have generated graph between number of rating and ratings of the movies for the Movie Lens dataset. From the graph we saw a sharp decline in number of ratings from 100. For this purpose, we have set the threshold between 80 to 90.

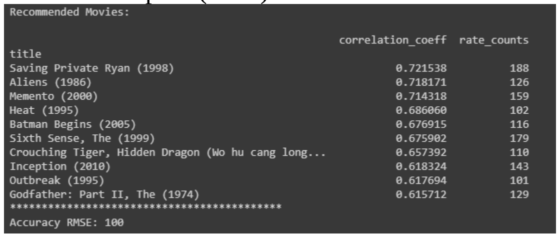
On the other hand, we have tested out recommendation model on variety of datasets. We have used different scales of datasets. As we know with more data, with data of large scale we can achieve better accuracy. Hence, we were able to achieve better results.

As we used various approaches for building this recommendation model and we analyzed this model using various metrics, we also could have used different concepts for enhancing the performance of the model like deep learning. We could have used different data cleaning methods like matrix factorization method for predicting the null values of the movie matrix. Other than that, on the analysis area, we tried to cover most of the major evaluation metrics for evaluating the performance of the model. However, learning more about the different kind of evaluation metrics could have helped a lot.

Nowadays most of the top companies using the recommended systems uses either content or collaborative-based filtering approach. We are planning to extend this project in combining the two filtering approaches. The main task would be finding and evaluating the dataset. We have tried implementation of most of the top dataset but none of them contain the data required for both the filtering approach. We also planning to make an interactive system where users enter the genres that they like to watch with the keywords that fascinate them including the actors/actress name or the director name. According to studies, the users are spending more time searching the movies rather than watching it. So were plaining to make *random movie generator*. We were also planning to combine Bayesian and collaborative filtering approach or content collaborative filtering approach (also known as hybrid approach).

Below are some results generated by the model for different datasets.
Collaborative-based filtering approach for Movie lens dataset.

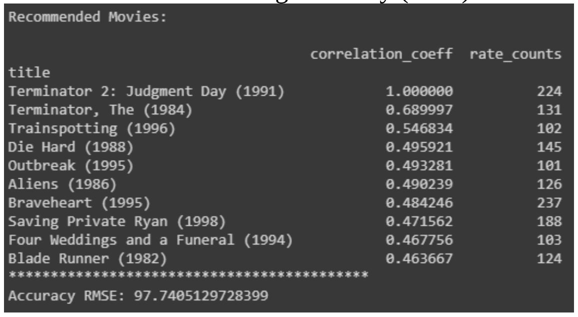
- Philadelphia (1993)



```
Recommended Movies:

                                              correlation_coeff  rate_counts
title
Saving Private Ryan (1998)                             0.721538          188
Aliens (1986)                                          0.718171          126
Memento (2000)                                         0.714318          159
Heat (1995)                                            0.686060          102
Batman Begins (2005)                                   0.676915          116
Sixth Sense, The (1999)                                0.675902          179
Crouching Tiger, Hidden Dragon (Wo hu cang long...     0.657392          110
Inception (2010)                                       0.618324          143
Outbreak (1995)                                        0.617694          101
Godfather: Part II, The (1974)                         0.615712          129
******************************************
Accuracy RMSE: 100
```

- Terminator 2: Judgment Day (1991)



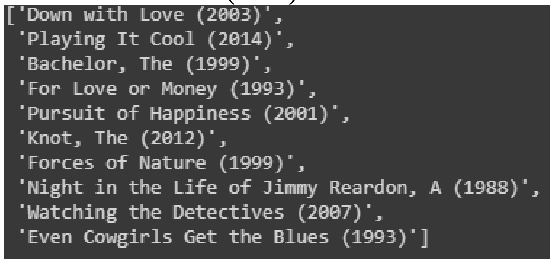
```
Recommended Movies:

                                  correlation_coeff  rate_counts
title
Terminator 2: Judgment Day (1991)          1.000000          224
Terminator, The (1984)                     0.689997          131
Trainspotting (1996)                       0.546834          102
Die Hard (1988)                            0.495921          145
Outbreak (1995)                            0.493281          101
Aliens (1986)                              0.490239          126
Braveheart (1995)                          0.484246          237
Saving Private Ryan (1998)                 0.471562          188
Four Weddings and a Funeral (1994)         0.467756          103
Blade Runner (1982)                        0.463667          124
******************************************
Accuracy RMSE: 97.7405129728399
```

The overall accuracy achieved is 95.858%. This accuracy is achieved by the mean of the various recommendation, we nearly took 40 movies recommendation and calculated the mean of accuracy achieved.

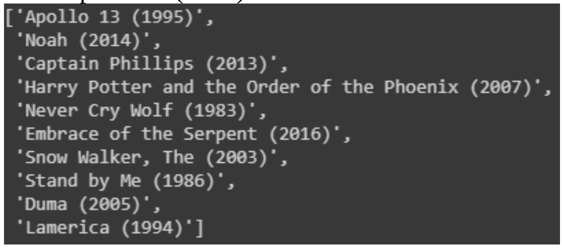
Below is the result for content-based filtering approach.
For Movie Lens dataset.

- Made of Honor (2008)



```
['Down with Love (2003)',
 'Playing It Cool (2014)',
 'Bachelor, The (1999)',
 'For Love or Money (1993)',
 'Pursuit of Happiness (2001)',
 'Knot, The (2012)',
 'Forces of Nature (1999)',
 'Night in the Life of Jimmy Reardon, A (1988)',
 'Watching the Detectives (2007)',
 'Even Cowgirls Get the Blues (1993)']
```

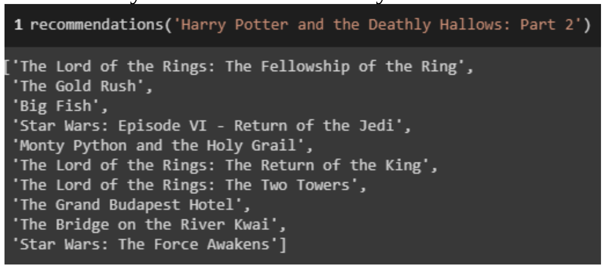
- Apollo 13 (1995)



```
['Apollo 13 (1995)',
 'Noah (2014)',
 'Captain Phillips (2013)',
 'Harry Potter and the Order of the Phoenix (2007)',
 'Never Cry Wolf (1983)',
 'Embrace of the Serpent (2016)',
 'Snow Walker, The (2003)',
 'Stand by Me (1986)',
 'Duma (2005)',
 'Lamerica (1994)']
```

For IMDB dataset.

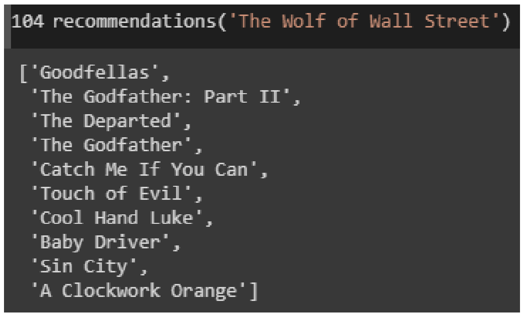
- Harry Potter and the Deathly Hallows: Part 2



```
recommendations('Harry Potter and the Deathly Hallows: Part 2')

['The Lord of the Rings: The Fellowship of the Ring',
 'The Gold Rush',
 'Big Fish',
 'Star Wars: Episode VI - Return of the Jedi',
 'Monty Python and the Holy Grail',
 'The Lord of the Rings: The Return of the King',
 'The Lord of the Rings: The Two Towers',
 'The Grand Budapest Hotel',
 'The Bridge on the River Kwai',
 'Star Wars: The Force Awakens']
```

- The Wolf of Wall Street

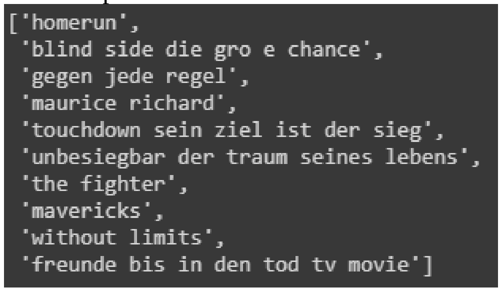


```
104 recommendations('The Wolf of Wall Street')

['Goodfellas',
 'The Godfather: Part II',
 'The Departed',
 'The Godfather',
 'Catch Me If You Can',
 'Touch of Evil',
 'Cool Hand Luke',
 'Baby Driver',
 'Sin City',
 'A Clockwork Orange']
```

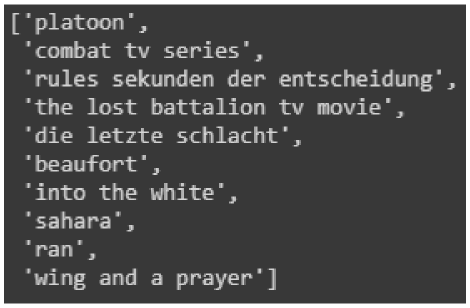
For Kaggle dataset.

- The express



```
['homerun',
 'blind side die gro e chance',
 'gegen jede regel',
 'maurice richard',
 'touchdown sein ziel ist der sieg',
 'unbesiegbar der traum seines lebens',
 'the fighter',
 'mavericks',
 'without limits',
 'freunde bis in den tod tv movie']
```

- Into the white



```
['platoon',
 'combat tv series',
 'rules sekunden der entscheidung',
 'the lost battalion tv movie',
 'die letzte schlacht',
 'beaufort',
 'into the white',
 'sahara',
 'ran',
 'wing and a prayer']
```

CONCLUSION

We have successfully implemented the movie recommendation system with two kinds to approach. We have successfully gained the overall accuracy of 95.858%. The algorithm works on variety of dataset and we have successfully implemented on them. We used dataset of IMDB, Kaggle and Movie lens.

REFERENCES

Asela Gunawardana, Guy *Shani A Survey of Accuracy Evaluation Metrics of Recommendation Tasks* Journal of Machine Learning Research 10 (2009) 2935-2962 Submitted 11/09; Published 12/09

Sang-Ki Ko, A Smart Movie Recommendation System,

Manoj Kumar, A Movie Recommender System: MOVREC, International Journal of Computer Applications (0975 – 8887) Volume 124 – No.3, August 2015

Prateek Sappadla, Movie Recommender System

Noratiqah Mohd Ariff, Comparison between content-based and collaborative filtering recommendation system for movie suggestions
https://towardsdatascience.com/how-to-build-from-scratch-a-content-based-movie-recommender-with-natural-language-processing-25ad400eb243