**Movie Personalized Recommendation System**

**Summary**

To explore and realize a movie personalized recommendation system, we developed a recommendation system machine learning model called item-based collaborative filtering. We did experiments based on a public dataset called MovieLens. The MovieLens dataset contains rating data for multiple movies from multiple users, and also includes movie metadata information and user attribute information.

The dataset used in this experiment is the ml-latest-small dataset from the movie website MovieLens, which contains 100,000 ratings and 3,600 tag applications applied to 9,000 movies by 600 users. The ml-latest-small dataset consists of four csv files, ratings.csv, movies.csv, links.csv, and tags.csv. For example, ratings.csv is the user rating information table, with four titles in the first row: userId, movie id, user rating and timestamp, where the user rating range is 0.5-5. Movies.csv is the movie attributes table, which contains three fields, movieId, title, genre. The only information we crawled was the ids and the ratings, so we took out some more statistical dimensions from this information to use later. We also conducted correlation analysis between variables in the dataframe.

In order to build the personalized recommender system, we developed two recommendation models including User-based Collaborative Filtering and Item-based Collaborative Filtering models. Collaborative filtering algorithms are the best known and most widely used recommendation algorithms. In order to build the personalized recommender system, we developed two recommendation models including User-based Collaborative Filtering and Item-based Collaborative Filter models. Users can make more reasonable choices based on the two recommended results. Finally, the system was evaluated by calculating the precision, recall and popularity of the two algorithms, and the advantages and disadvantages of the two algorithms are compared. Experiments showed that the item-based collaborative filtering algorithm was better than the user-based collaborative filtering algorithm and the accuracy was higher.

In the next stage, we conducted the model performance analysis. For instance, the model shall recommend the movie The Hobbit to users who have watched the movie Lord of Rings. We looked for correlations between The Hobbit and other movies and sorted similarities from the highest to the lowest and Lord of Rings was the top 1 result. However, we also identified some outliers where the number of ratings for the movie was too low and didn’t get the accurate result, which showed that the algorithm used was not accurate enough, and the two algorithm systems currently used did not run efficiently and quickly, and the accuracy of the recommended items was limited. We will explore this part in our future work.

In conclusion, collaborative filtering recommendation is the most researched personalized recommendation technique with a high degree of personalization. The starting point of collaborative filtering is that users with similar interests may be interested in the similar things, and users may prefer products that are similar to what they have already watched. The main purpose of this system is to study the implementation of the user-based collaborative filtering recommendation algorithm and to conduct an in-depth analysis of the algorithm. After this experiment, although the research on the user-based collaborative filtering recommendation algorithm was not so deep that we could propose some constructive improvements like deep learning approaches, we have gained a certain understanding of the principles of recommendation systems, which has laid a solid foundation for more in-depth research in the future.

1. **Background**

With the rapid development of Internet technology, we have now entered the era of big data, and the information on the Internet is exploding, with hundreds of millions of data coming out every day. While these bring convenience to users, they also bring an unprecedented problem - "information overload".

"Information overload" means that it is difficult for users to find the information they are interested in from the huge amount of data. In order to solve the information overload problem, the first thing that emerged was the search engine, but once the user is unable to accurately describe the keywords they need, the search engine is powerless since the search is a passive retrieval. Moreover, the needs of different users vary greatly from one another. Search Engine itself fails to meet the needs of these certain users. With the development of technology, personalized recommendation systems have played an increasingly important role in information retrieval.

However, the initial recommendation system would only recommend to the users the products that are popular and loved by the public or that can make the most profit for the company and did not analyze and give personalized results for each user. This was very ineffective. Therefore, there is a desire for a system and method that can automatically recommend items to users based on their preferences and an analysis of the attributes of the recommended products. This is a personalized recommendation system.

Recommendation algorithms are widely used in the Internet industry, including tech giants like Amazon, Tiktok, Reddit, etc. Personalized recommendations are, in the abstract, a kind of fitting function for content satisfaction, involving user features and content features, as two major sources of the dimensions required for model training, and click-through rate, page dwell time, comments or ratings, etc. can be used as a quantitative label, so that feature engineering can be performed to construct a dataset, and then a suitable supervised learning algorithm can be selected for training to get a model that recommends preferred content for customers, such as advice and articles in the case of Reddit, videos in the case of Tiktok, and household products in the case of Amazon.

1. **Dataset**

The data utilized by our movie recommendation system is actually relatively small, especially the user side of the dimension, while the normal recommendation system involves dimensions such as page dwell time, click frequency, collection and other such dimensions are not, as well as the user's own dimensions are relatively small, no address, age, gender and other such basic dimensions, so we crawl the data only scores and movie names and so on. This project mainly focuses on giving user id and movie id to predict the rating to this movie by the user. Ideally, a well-performed movie recommendation system shall recommend users with movies that share similar genres, or users’ favorite ones, or other favorite movies watched by users with similar interests.

In the data collection part, we chose the public dataset called Movielens. The MovieLens dataset contains rating data for multiple movies from multiple users, and also includes movie metadata information and user attribute information. The only information we crawled was the score and the name of the movie, so we took out some more statistical dimensions from this information to use later. The movie data we crawled (except for movie details and picture information) is shown in Figure 1.

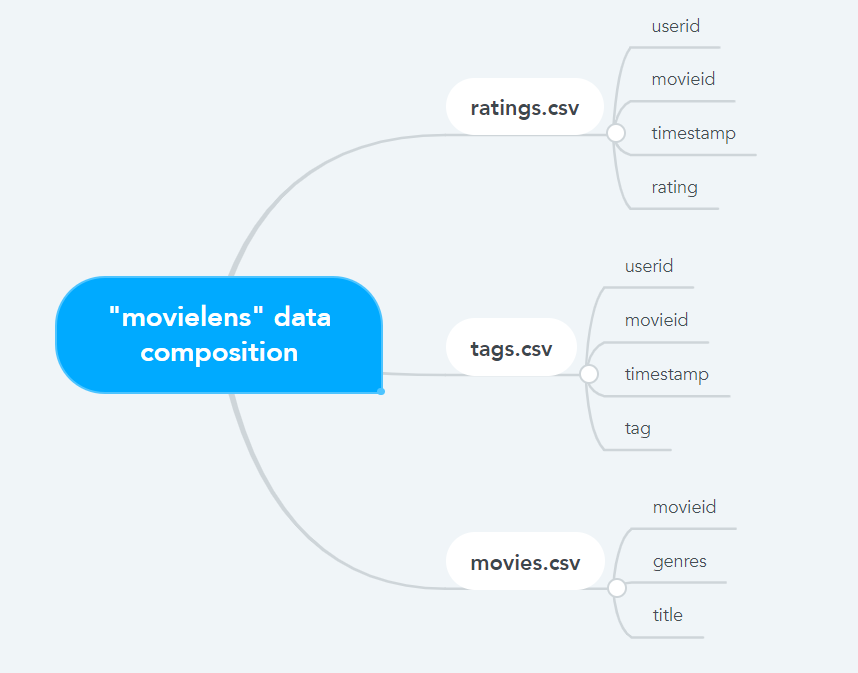


Fig. 1 The detail of MovieLens dataset

**2.1 Dataset File Ratings.csv**

The contents of the file contain the ratings of each user for each movie. The data format is shown in Table 1. There are four columns including userId, movieId, rating, and timestamp, where userId represents the id of each user, movieId represents the id of each movie, rating represents a user rating which is a 5-star system, increasing in half-star scale (0.5 stars - 5 stars), and timestamp represents the number of seconds since 00:00 on January 1, 1970 to the time the user submitted the evaluation. The data is sorted in order of userId and movieId. We use the Python Pandas package to observe the description of the data and the first 5 rows.

Table 1 Content of Ratings.csv file

| **userid** | **movieid** | **rating** | **timestamp** |
| --- | --- | --- | --- |
| 1 | 2 | 3.5 | 2005-04-02 23:53:47 |
| 1 | 29 | 3.5 | 2005-04-02 23:31:16 |
| 1 | 32 | 3.5 | 2005-04-02 23:33:39 |
| 1 | 47 | 3.5 | 2005-04-02 23:32:07 |
| 1 | 50 | 3.5 | 2005-04-02 23:29:40 |

**2.2 Dataset File Movies.csv**

The file contains the id and title of a movie, as well as the category of that movie. The data format is shown in Table 2. There are three columns including movieId, title, and genres. MovieId represents the id of each movie, title represents the title of the movie, and genres represents the category of the movie. We

use the Python Pandas package to observe the description of the data and the first 5 rows.

Table 2 Content of Movies.csv file

| **movieid** | **title** | **genres** |
| --- | --- | --- |
| 1 | Toy Story(1995) | Adventure|Animation|Children|Comedy|Fantasy |
| 2 | Jumanji(1995) | Adventure|Children|Fantasy |
| 3 | Grumpier Old Men(1995) | Comedy|Romance |
| 4 | Waiting to Exhale(1995) | Comedy|Drama|Romance |
| 5 | Father of the Bride Part II(1995) | Comedy |

1. **Data Processing & Data Analysis**

We utilized Python programming along with Python libraries including packages like Pandas, Numpy, Matplotlib, etc. to process the MovieLens dataset. The results showed that the column MovieID has 2.000026e+07 lines with a mean number of 9.041567e+03, standard deviation of 1.978948e+04, minimum value of 1.000000e+00, and a maximum value of 1.312620e+05.

The column Ratings has 2.000026e+07 lines with a mean number of 3.525529e+00, standard deviation of 1.051989e+00, minimum value of 5.000000e-01, and a maximum value of 5.000000e+00. The mean value of viewers' ratings for the movies is approximately 3.52, which indicates that most viewers are relatively satisfied with the movies. The minimum value of rating is 0.5 instead of 0, implying that the distribution of rating inside the dataset is [0.5, 5].

The column UserID has 2.000026e+07 lists lines with a mean number of 6.904587e+04, standard deviation of 4.003863e+04, minimum value of 1.000000e+00, and a maximum value of 1.384930e+05.

In the next stage, we conducted correlation analysis between variables in the dataframe. The correlation between userid and movieid is -0.000850 so as the value of one variable increases, the other decreases. Since the value is quite small and close to zero, they have very little linear relationship. The correlation between userid and ratings is 0.001175 so there is very little relationship between them. But because the correlation is extremely small, their relationship is hard to detect. The correlation between movieid and ratings is 0.002606, we can see there is a weak linear relationship between two variables. Based on these analyses, we found that the mutual relationship among these three variables is quite weak.

To continue data processing, we tried to clear out the null value in the data sets. First, we analyzed 27278 rows and 3 columns in the movieid dataframe. We now use ‘movies.isnull().any().any()‘ to determine if there are any null values and get the output ‘False’, which means we don't have any null values in this dataframe. We did the same data processing with the other two dataframes. The result showed that ratings don’t have any null values, but the output of ‘tags.isnull().any().any()’ is true, so we need to remove the null values with ‘tags=tags.dropna()’ from the tags dataframe. The original output of ‘tags.shape’ is (465564, 3). We now run ‘tags.isnull().any().any()’ and ‘tags.shape’ again and we get the output False and (465548, 3), which means we have found 6 null values that were already removed from this dataframe.

The histogram plot below shows the distribution of ratings among the movies. This is the rating bar graph, we can see that the common rating values in our dataset are 4-star, followed by 3-start and 5-star.

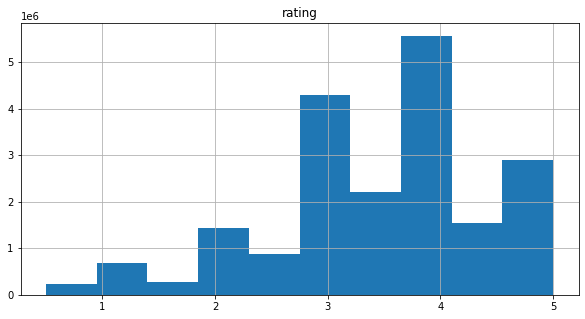


Fig. 2 The histogram plot of distribution of ratings among movies

And from the box plot we can see that the median is approximately 3.5 and the range is from 1.5 to 5 followed by a few outliers that are below 1.5.

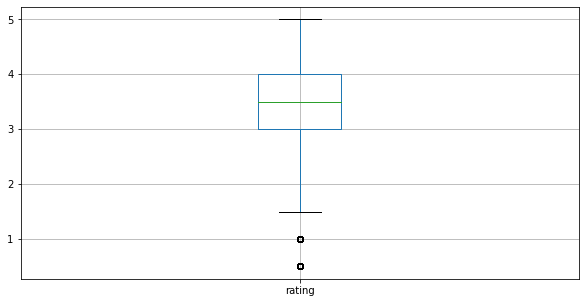


Fig. 3 The boxplot of distribution of ratings among movies

This is the bar graph for each movie genre, we can see that most movies in the data sets fall into the science fiction genre.

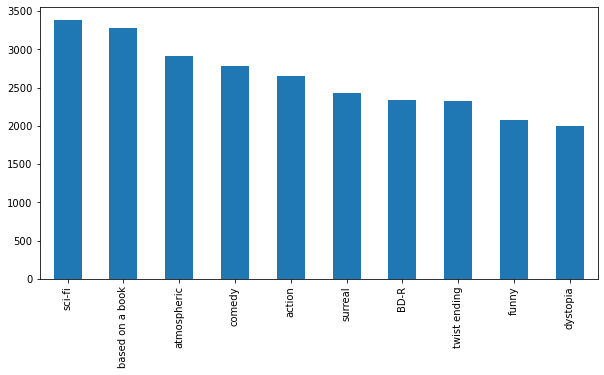


Fig. 4 The bar plot of distribution of movie genres

1. **Recommendation Model**

**4.1 User-based Collaborative Filtering**

Collaborative filtering is making recommendations according to a combination of users’ experience and experiences of other people. To realize this algorithm, first we need to build a user vs item matrix, where each row represents a user and each column represents an item like a movie, product or website. Secondly, we need to compute similarity scores between users, where each row is a vector that represents a user. In the next step the algorithm will compute similarity among these row vectors (users). Thirdly, the algorithm will find users who are similar to the target user based on its past behaviors. Finally, it recommends what you have not experienced before as the model output.

For example, think that there are two people. The first people watched 2 movies: Lord of the Rings and Hobbit. The second people only watched the Lord of the rings movie. Then our user-based collaborative filtering would compute the similarity of these two people and identify that both of them have watched the movie Lord of the rings. So the model would identify the similarity between these two people.

However, User-based collaborative filtering does have some problems and drawbacks: In this system, each row of matrix represents a user. Therefore, computing, comparing, and finding similarities between them is computationally expensive and will take too much computation. Also, habits of people can be changed. Therefore, making correct and useful recommendations can be hard in time. In order to solve these problems, the item-based collaborative filtering approach was proposed.

**4.2 Item-based Collaborative Filtering**

In this method, instead of finding relationships among users, user-interacted items like movies or stuff are compared with each other to find item similarities. In a user-based recommendation system, habits of users can be changed. This situation makes the system hard to adapt. However, in the item-based recommendation system, movies or stuff do not change quickly over time. Therefore its recommendation performance would be relatively stable.

Since there are almost 7 billion people (possible users) all over the world, comparing people similarities (in most cases) is much harder in computation cost than computing item similarities. In the item-based recommendation system, we still need to make a user vs item matrix that is also used in user-based recommender systems, where each row represents a user and each column represents an item like a movie, product or website. However, at this time, instead of calculating similarity among rows, the model needs to calculate similarity among columns that are items like movies or stuff.

For example, there is a similarity between the movie Lord of the Rings and The Hobbit because both are liked by three different people. Then our item-based collaborative filtering model would identify that there is a similarity between these two movies. If this similarity is high enough, the model would recommend The Hobbit to other people who have only watched the Lord of the Rings movie.

1. **Result Analysis**

In order to see if the outputs are accurate, we tested our model with a few examples. We imported movie data sets first, and then we listed out the movie id and title columns, imported rating data and looked at columns. What we need are user id, movie id and rating columns so we merge movie and rating data. Finally, we tested the model with some movies. For instance, the model shall recommend the movie The Hobbit to users who have watched the movie Lord of Rings. We looked for correlations between The Hobbit and other movies and sorted similarities from highest to the lowest and Lord of Rings was the top 1 result.

1. **Future Work**

The work in this paper completes a simple personalized movie recommendation system, and there are still many places that are inadequate that need further improvement, especially in the following aspects.

Firstly, the algorithm used was not efficient enough. The two algorithm systems currently used do not run efficiently and quickly enough, and the accuracy of the recommended items is not high. After that, a combination of tag-based and content-based recommendation algorithms can be considered for recommendation. Also we can fill in age, gender, occupation and other information at the time of user registration to make preliminary classification first to give recommendations, and then based on the rating of accurate recommendations, the performance of our proposed model/method will be better.

Secondly, the system uses users' historical movie ratings data, which are sparse: the logged-in user may have seen very few movies, which leads to fewer crossover items with the movies rated by users in the database, making the calculated similarity very inaccurate. Another starting point to improve the accuracy of the recommendation system is how to reduce the trouble caused by data sparsity. The current system uses a small dataset from Movielens website, and the size of this dataset is only 20MB after compression, so the number of movies included is not quite large, and the effect of recommendation is greatly reduced. Later, we can use a larger dataset and consider traversing more rating data to make more precise recommendations.

**Reference**

[1] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4, Article 19 (December 2015), 19 pages. DOI=<http://dx.doi.org/10.1145/2827872>

[2] Hofmann T. Latent semantic models for collaborative filtering[J].ACM Transactions on Information Systems, 2004, 22(1):89-115.

[3] Pilli L E, Mazzon J A. Information overload, choice deferral, and moderating role of need for cognition: Empirical evidence[J]. Revista De Administracao Publica, 2016, 51(1):36-55.