INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact factor: 4.295
(Volume 4, Issue 2)
Available online at: www.ijariit.com

Movie recommendation system

Deepak Kumar Singh
deepak.96ksingh@gmail.com
IMS Engineering College, Ghaziabad, Uttar Pradesh

Abhishek Gangwar
abhishekgangwar56@gmail.com
IMS Engineering College, Ghaziabad, Uttar Pradesh

Abhishek Sharma
abhisheksharma7060@gmail.com
IMS Engineering College, Ghaziabad, Uttar Pradesh

ABSTRACT

Recommendation systems have become increasingly popular in recent times. They are utilized in a variety of areas like music, videos, and social sites. There are two ways to produce recommendation-through collaborative filtering-which presents recommendation based on past history of user. Content based filtering uses similarity between items to recommend a product. In this paper we have proposed a movie recommendation system based on collaborative filtering method and simple method of ranking movies according to their popularity rating. The system has been developed using PHP, python.

Keywords: *Collaborative filtering, Euclidian score, content based filtering, Recommendation system, PHP, MySQL, MovieLens.*

1. INTRODUCTION

Main goal of recommendation systems is to predict user's wish list and to supply best recommendations to him. Recommendation engines are trending everywhere now days from e commerce sites to social networking sites to music and video streaming.

Recommendation systems are mainly employed by large business enterprises [6,7]. The main idea is to increase sales of business and revenue using these systems .For social networking sites these systems help in increase of user engagement.

NETFLIX [7] has benefited the most from recommendation engines and is the prime factor motivation behind their prevalence regularly organizing competition for predicting recommendations 10 percent more accurate than those provided by the company recommender
We can classify recommendation systems in two categories

- Collaborative filtering
- Content based filtering

1.1 Collaborative Filtering

It's the most sort after, most widely implemented and most mature technologies that is available in the market. Collaborative recommender systems [1, 10] aggregate ratings or recommendations of objects, recognize commonalities between the users on the basis of their ratings, and generate new recommendations based on inter-user comparisons [11]. The greatest strength of collaborative techniques is that they are completely independent of any machine-readable representation of the objects being recommended and work well for complex objects where variations in taste are responsible for much of the variation in preferences. Collaborative filtering [9] is based on the assumption that people who agreed in the past will agree in the future and that they will like similar kind of objects as they liked in the past.

Collaborative Filtering [2] algorithm considers "User Behaviour" for recommending items. They exploit behaviour of other users and items in terms of transaction history, ratings, selection and purchase information. Other user's behavior and preferences over the items are used to recommend items to the new users.

The basic idea of collaborative filtering methods is that these unspecified ratings can be imputed because the observed ratings are often highly correlated across various users and items. For example, consider two users named Alice and Bob, who have very similar tastes. If the ratings, which both have specified, are very similar, then their similarity can be identified by the underlying algorithm. In such cases, it is very likely that the ratings in which only one of them has specified a value, are also likely to be similar. This similarity can be used to make inferences about incompletely specified values. Most of the models for collaborative filtering focus on leveraging either inter-item correlations or inter-user correlations for the prediction process. Some models use both types of correlations. Furthermore, some models use carefully designed optimization techniques to create a training model in much the same way a classifier creates a training model from the labeled data. This model [14] is then used to impute the missing values in the matrix, in the same way that a classifier imputes the missing test labels. There are two types of methods that are commonly used in collaborative filtering, which are referred to as memory-based methods and model-based methods.

1.2 Content Based Filtering

It's mainly classified as an outgrowth and continuation of information filtering research. In this system, the objects are mainly defined by their associated features [4,6]. A content-based recommender learns a profile of the new user's interests based on the features present, in objects the user has rated. It's basically a keyword specific recommender system here keywords are used to describe the items. Thus, in a content-based recommender system the algorithms used are such that it recommends users similar items that the user has liked in the past or is examining currently.
Content based systems [7, 13], recommends item based on a similarity comparison between the content of the items and a user's profile. The feature of items are mapped with feature of users in order to obtain user – item similarity.

Content-based methods have some advantages in making recommendations for new items, when sufficient rating data are not available for that item. This is because other items with similar attributes might have been rated by the active user. Therefore, the supervised model will be able to leverage these ratings in conjunction with the item attributes to make recommendations even when there is no history of ratings for that item.

Content-based methods do have several disadvantages as well:

1) In many cases, content-based methods provide obvious recommendations because of the use of keywords or content. For example, if a user has never consumed an item with a particular set of keywords, such an item has no chance of being recommended. This is because the constructed model is specific to the user at hand, and the community knowledge from similar users is not leveraged. This phenomenon tends to reduce the diversity of the recommended items, which is undesirable.

2) Even though content-based methods are effective at providing recommendations for new items, they are not effective at providing recommendations for new users. This is because the training model for the target user needs to use the history of her ratings. In fact, it is usually important to have a large number of ratings available for the target user in order to make robust predictions without overfitting. Therefore, content-based methods have different trade-offs from collaborative filtering systems.

Command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper.

2. RELATED WORK

Most systems use variety of items such as ratings from previous users to make recommendations. One prominent example of system was developed by Herlocker (2004) [9] for improving search results. Another major breakthrough in world of recommendation was by Lawrence in 2000 when they presented a recommender system which suggests product to customer from looking at the purchase behavior and history of shopper.

3. RESEARCH METHODOLOGY

We will be implementing two different algorithms for recommender .While the first algorithm [6] will give recommendations simply according to mean ratings.
In second algorithm [14] we will implement collaborative filtering.

3.1 Algorithm

- We will load the dataset with Pandas onto Dataframes ***data, item*** and ***user***
- we **merge** the three dataframes into one single dataframe
- Next we use **groupby** to group the movies by their titles. Then we use the size function to returns the total number of entries under each movie title. This will help us get the number of people who rated the movie/ the number of ratings.
- Take the mean ratings of each movie using the **mean** function. First we **groupby** *movie title*. From the resulting dataframe we select only the ***movie title***and the ***rating*** headers
- Then we use the mean function on them
- we sort the values by the ***total rating*** and this helps us sort the data frame by the number of people who viewed the movie
- split dataset so that cutoff is 100 for total ratings
- Then we sort the new Data frame with respect to the mean ratings. And we are done building the recommender system. Print out the head of the data frame to give the top 5 recommendations.

3.2 User Collaboration

- We will load the data sets firsts.
- We will use the file u.data first as it contains User ID, Movie IDs and Ratings. These three elements are all we need for determining the similarity of the users based on their ratings for a particular movie.
- sort the DataFrame by User ID and then we are going to split the data-set into a training set and a test set
- Create a user's list which is a list of users that contains a list of movies rated by him.

- Define a Function by the Name of Euclidean Score**.** The purpose of the Euclidean Score is to measure the similarity between two users based on their ratings given to movies that they have both in common
- Having more movies in common is a great sign of similarity. So if users have less than 4 movies in common then we assign them a high Euclidean Score**.**
- We will iterate over users_list and find the similarity of the users to the test user by means of this function and append the Euclidean Score along with the User ID to a separate list score list
- We then convert it first to a DataFrame, sort it by the Euclidean Score and finally convert it to a NumPy Array score matrix for the ease of iteration.
- We will see that certain user X see has the lowest Euclidean score and hence the highest similarity. So now we need to obtain the list of movies that are **not** common between the two users. Make two lists. Get the full list of movies which are there on X. And then the list of common movies. Convert these lists into sets and get the list of movies to be recommended.
- We need to create a compiled list of the movies along with their mean ratings. Merge the item and data files. Then groupby movie titles, select the columns you need and then find the mean ratings of each movie. Then express the dataframe as a NumPy Array.
- Now we find the movies on item_list by IDs from recommendation**.** Then append them to a separate list.
- Print them out and your recommendations are ready!

4. SIMULATION OF ENGINE

The recommendation system [13] cannot be only based on users input and just their recommendations. To make it dynamic we are actually combining with other API to issue a more lively experience for the users. Therefore, a complete user simulation can be discovered through this exercise.
Technology stuff that we are using are as follows:

- MySQL database
- HTML and CSS (Bootstrap)
- PHP
- WAMP

While the users list generated from the python script can be directly in form of readable file, we change it by applying necessary computations required so as to deliver a reduced file which could be further be used as a template my MySQL. We are using WAMP which comes preloaded with necessary stack so we don't need any further or specific applications while simulating the recommendation system.

While the file becomes ready to be fed into MySQL, we are writing PHP code to connect it with the database. After finishing it, we create another database for the movies and their ratings. Finally we create interacting web pages using HTML and CSS, which are dynamically helpful for users to operate upon. Now, the whole simulation is ready to be launched on local server at least. We used WAMP which automatically created server in form of localhost so we can operate on the recommendation system.
However, if one is facing complications [3] over that one can make use of Spark to implement the same. However due to limited use and dependency of the project, we didn't tried to do it. However for scalability Spark could be a good option.

5. CONCLUSION

In this paper we have introduced simple recommender system for movie recommendation. It allows a user to select his choices from a given set of attributes and then recommend him a movie list based on the cumulative weight of different attributes and using Euclidean By the nature of our system, it is not an easy task to evaluate the performance since there is no right or wrong recommendation; it is just a matter of opinions. Based on informal evaluations that we carried out over a small set of users we got a positive response from them. We would like to have a larger data set that will enable more meaningful results using our system. Additionally we would like to incorporate different machine learning and clustering algorithms and study the comparative results. Eventually we would like to implement a web based user interface that has a user database, and has the learning model tailored to each user.

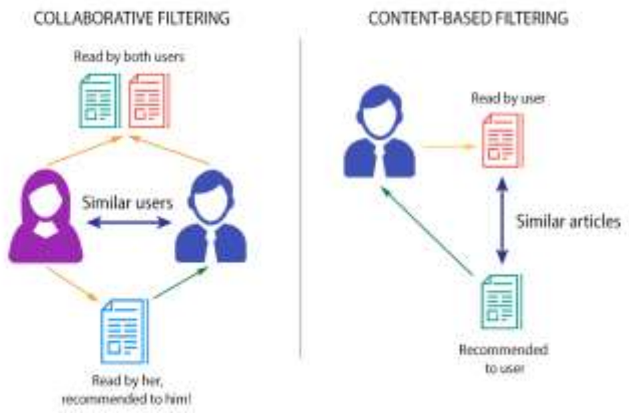

Figure 1: Different Filtering Techniques

5. REFERENCES

[1] Yijie Zhuang, Boyang Xu, Hao Wu Shaoyi Han, Hong Jin, M. Young, Movie Recommender System, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

[2] Koren, Yehuda, Robert Bell, and Chris Volinsky. "Matrix factorization techniques for recommender systems." Computer 42.8 (2009): 30-37.

[3] Bao, Zhouxiao, and Haiying Xia. "Movie Rating Estimation and Recommendation." CS229 (2012).

[4] Ricci, Francesco, Lior Rokach, and Bracha Shapira. Introduction to recommender systems handbook. Springer US, 2011.

[5] Rajaraman, Anand, and Jeffrey David Ullman.Mining of massive datasets. Cambridge University Press, 2011.

[6] Rupali Hande, Ajinkya Gutti, Kevin Shah, Jeet Gandhi, Vrushal Kamtikar, moviemender- a movie recommender system, international journal of engineering sciences & research technology, November, 2016

[7] James Bennett, Stan Lanning; "The Netflix Prize", In KDD Cup and Workshop in conjunction with KDD, 2007.

[8] Mohammad Yahya H. Al-Shamri , Kamal K. Bharadwaj; "A Compact User Model for Hybrid Movie Recommender System " in International Conference on Computational Intelligence and Multimedia Applications 2007

[9] J. L. Herlocker, J. A. Konstan, and J. Riedl, "Explaining collaborative filtering recommendations," in *ACM CSCW '00*, pp. 241–250, ACM, 2000.

[10] Christina Christakou, Leonidas Lefakis, Spyros Vrettos and Andreas Stafylopatis; "A Movie Recommender System Based on Semi-supervised Clustering", IEEE Computer Society Washington, DC, USA 2015.

[11] Luis M. de Campos, Juan M. Fernández-Luna *, Juan F. Huete, Miguel A. Rueda-Morales; "Combining content-based and collaborative recommendations: A hybrid approach based on Bayesian networks", International Journal of Approximate Reasoning, revised 2010.

[12] Urszula Kużelewska; "Clustering Algorithms in Hybrid Recommender System on MovieLens Data", Studies in Logic, Grammar and Rhetoric, 2014.

[13] Dietmar Jannach, Gerhard Friedrich; "Tutorial: Recommender Systems", International Joint Conference on Artificial Intelligence, Beijing, August 4, 2013.

[14] Gaurangi, Eyrun, Nan; "MovieGEN: A Movie Recommendation System", UCSB.

[15] Harpreet Kaur Virk, Er. Maninder Singh," Analysis and Design of Hybrid Online Movie Recommender System"International Journal of Innovations in Engineering and Technology (IJIET) Volume 5 Issue 2, April 2015.