Recommender System Framework using Clustering and Collaborative Filtering

Namita Mittal
MNIT JAIPUR
INDIA
mittalnamita@rediffmail.com

Richi Nayak
QUT, Brisbane
Australia
r.nayak@qut.edu.au

MC Govil
MNIT Jaipur
INDIA
govilmc@yohoo.com

KC Jain
MNIT Jaipur
INDIA
jainkc_2003@yahoo.com

Abstract- **Collaborative filtering is becoming greatly popular as it contributes in reducing information overload. Collaborative filtering based recommender system focuses on predicting new items of interest for a user based on correlations computed between that user and other users. In this paper we propose a framework based on, application of data partitioning/clustering algorithm on ratings dataset followed by collaborative filtering for developing a Movie Recommender System. The proposed system reduces the computation time considerably and increases the prediction accuracy.**

Keywords-**Recommender System, Clustering, Collaborative Filtering, K-Means Algorithm, Slope One Algorithm**

I. INTRODUCTION

A recommender system [1] is an intermediary program or an agent that intelligently compiles a list of requisite information which suits a user's tastes and needs. Many recommender systems[2] have been designed and implemented for various types of items including newspapers, research papers, emails, books, movies, music, restaurants, Web pages and other e-commerce products. This paper proposes a new approach to develop a framework for an efficient recommender system. To construct a recommender system, two information filtering methods are considered [12].

1. By analyzing the information content (known as content-based filtering)
2. By referencing other users' access behaviors (known as collaborative filtering).

The collaborative filtering approach [8] has been used to achieve the desired framework for our recommender system. Recommender System predicts new items of interest for a user on the basis of predictive relationships discovered between the user concerned and the other users sharing the same tastes and interests. The aim of collaborative filtering in a recommender system is therefore to recommend items to a target user based on the opinion of other users. To illustrate the application of collaborative filtering algorithms in predicting recommendations to a user, let us consider a typical scenario with a set of *n* users say $U = \{u_1,u_2,..,u_n\}$ and a list of *m* items say $I =\{i_1,i_2.,i_{\ m}\}$. Every user u_k has rated $i_1,i_2,i_3\ldots i_m$ so U has a list of items $I_{uk}=\{i_1,i_2,i_3\ldots i_m)$ which the user has experienced and has expressed an opinion about, generally in the form of rating as in case of MovieLens [4] dataset. Under this scenario the task of a collaborative filtering algorithm is to predict item $i \in I$ to a user $u_a \in U$, called the *current user*, which the user may like. We propose to achieve the predictions for a user by first minimizing the size of item set the user needs to explore. This is done by partitioning with respect to certain characteristics which is common to the items under consideration. This is followed by application of clustering methodology on the partitioned data and finally the application of collaborative filtering on the item of the user's choice.

The rest of the paper is divided into the following sections. Section II gives related work done on recommender systems. Section III shows a diagrammatic overview of the proposed system. Section IV presents research methodology. Section V consists of the algorithms we applied to the system. Section VI highlights the experiments done followed by their results and finally Section VII concludes the paper.

II. RELATED WORK

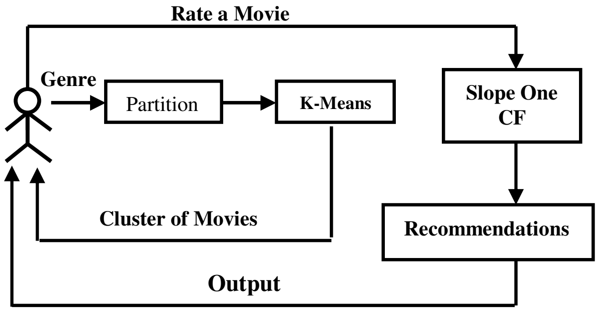
Methodologies towards building efficient and effective recommender systems have always posed a challenge to most of the researchers in the field of data mining research [6]. Collaborative Filtering is a new concept that descended from the work in the field of information filtering [9].The term collaborative filtering was coined by Goldberg [10] who first published an account of using collaborative filtering techniques in the email filtering system called Tapestry. Sarwar [8] has proved that item-based collaborative filtering is better than user-based collaborative filtering in terms of precision and efficiency. Item based collaborative filtering has been used in the proposed approach. To this effect, the Slope One algorithm introduced by Lemire and Maclachlan [3] has been implemented. Nicholas Ampazis [11] applied the method of concept decomposition to achieve collaborative filtering on the Netflix Dataset.

III. OVERVIEW OF OUR SYSTEM

The whole process of selecting, rating and recommending a movie can be summarized by the steps given below and shown in fig 1:

1. The user enters the genre of the movie (for eg: action, romance etc).
2. Partition of the data set is done according to the genre of the movies. Partition reduces the data size considerably, and it provides the platform for clustering.

978-0-7695-4246-1/10 $26.00 © 2010 IEEE
DOI 10.1109/ICETET.2010.121



CF- Collaborative Filtering

Fig1: System Overview

3. K-Means clustering is applied on the partitioned data set, to form cluster of movies having the same genre and characteristics.
4. The user can then choose his movie from the clusters obtained.
5. Once the user has selected his movie, he is asked to rate the movie.
6. Based on the rating provided by the user, the slope one algorithm is used and collaborative filtering is applied on the cluster. This provides user with various recommendations. These recommendations will be as per user's behavior and ratings.

A. Item Clustering Approach

The first task carried out to achieve an efficient and quick execution is the partitioning of the sparse item set of movies into partitions based on the genre requested by the user. The partitioning reduces one large-dimensionality item space into a set of smaller-dimensionality spaces with fewer items, less ratings, and often less users. As a result of this the time to compute prediction decreases since there is less data to consider. The prediction accuracy also increases as the ambiguities generated by ratings on items of dissimilar content have been removed. This is followed by the clustering of the requisite partition in order to divide the movies set into clusters of similar movies. The clustering is achieved by considering parameters of the user's age and the rating provided by those users to the movies under the partition. The age of the user has been considered as a parameter in clustering the movies following the fact that movies of different genres attract users from different age groups. Thus clusters with much more accurate results are likely to be found under such an approach.

IV.RESEARCH METHODOLOGY

Many algorithms [5] have been reported for the purpose of data clustering. In the proposed system the K-Means algorithm has been implemented for clustering of the items as shown in algorithm 1.

```
Input: No. of clusters (say k)
        Age and Rating of each user from genre table say g
begin:
1. for all i, $i \in n$, where n=no of records in g
 {
 2. $Parameter_i$ (say P)=$Age_i+Rating_i$
 3. insert P into temp table t such that $Parameter_{i+1}>Parameter_i$
    } / / end for
4. Select k records randomly and insert into cluster table say c
5. $x = ( Parameter_n - Parameter_0 )/k$
6. $Cluster_j=Cluster_0 + x*j$, where $0<j<k$
7. $Distance_j = abs ( Cluster_j - Parameter_i )$, where $0<j<k$ and
   $0<i<n$
8. for minimum (Distance) insert the record into minDistance m
9. q =0 to k and p=0 to n
   for all $Cluster_{q \text{ in } c} = Cluster_{p \text{ in } m}$
   {   $NewCluster_q = Average ( \sum^p Parameter_{\text{in } m} )$
   }        / / end for
   if ( $NewCluster_q >= abs( Cluster_{q \text{ in } c} - 0.5 )$)
   {         $Cluster_{q \text{ in } c} = NewCluster_q$
   }        / / end if
10. For all k,
   If ($NewCluster_k = Cluster_k$)
   goto 10
   else
   repeat 6 to 9
end
```

Algorithm 1: K-Means Algorithm Implemented

The above algorithm takes the *genre* of the movies an user would like to rate as the input along with the number of clusters in which the user wants the movies to be categorized. The sum of the *Age* and *Rating* of a user for a movie is the main parameter for clustering. To select the *k* clusters, a function is defined which takes the records with the values of the parameters which differ by a constant value *x*, where x is calculated as follows,

$$x = (MaxParameter - MinParameter) / k$$

The selected records are inserted into the cluster table (say c).Each record is compared with the records in **c**. The record is associated with that cluster with which it gives the minimum distance and is inserted into the new minDistance table (say m). For each cluster, the mean value of the parameters of the records in that cluster is calculated. The mean value is the new parameter value of the cluster. The process is repeated until we get the mean values for all clusters to be equal to the old cluster values.

To achieve collaborative filtering on the user's requisite item derived from the clustering, the Slope One Algorithm [3] has been implemented as shown in Algorithm 2.

```
Input: Movie $m_u$ and rating r on $m_u$ provided by user
from table genre (say g)

begin:
  for all i ∊ n-1, where n= No. of movies in g
    {
    the pair ( $m_u$ , $m_i$ ) is inserted into table (say d).
    $count_i$ =no. of users who have rated both $m_u$ and $m_i$
    $sum_i$ = Cumulative Difference of total ratings on both the movies
    $Average_i = sum_i / count_i$
    }    // end for
  sort in descending order of Average

  for k=0 to 9     // top ten recommendations
  {
      recommend $m_k$ as output
  }     // end for

  end.
```

Algorithm 2: Slope One Algorithm Implemented

MovieID1	MovieID2	Count	Sum
<movie rated by user>	<other movie>	<no. of users who rated both movies>	<cumulative difference between ratings>

Fig 2: Table generated for rating on a movie

Average=Sum/Count
<sort in descending order of average>

Fig 3: Table for output

Output=MovieID2 where Average ∊ {top ten average values arranged in descending order}

The above algorithm takes a movie (say m) and the rating (say r) on that movie as input. A new table (say s) is generated where m is compared with every other movie other than m in the genre table (say g). The comparison is made by tracing the number of users who have rated both m and the movie in comparison. This value is set as *count*. Also the cumulative difference between the ratings on the movies by each user is computed and stored as the *sum*. The average rating on the movie with respect to m is then calculated as

$$Average=sum/count$$

The average values for all the movies are calculated in similar way and then the movies are sorted in descending order of the *Average* values. The first 10 movies with the highest *Average* values are recommended to the user.

V. SYSTEM EVALUATION

The approach was evaluated on a subset of movie rating data collected from the MovieLens dataset [7] with 70k ratings, 500 users and 3883 movie titles with different genres.

The system generates the clusters of movies with parameters of *Age* and *Ratings* of different users who have rated movies of a particular genre. The *Action* genre has been selected for this purpose. Analysis of the pattern of the clusters formed, shows that users of age group 25 have rated most of the movies of the mentioned genre. This, unlike clustering based on Ratings, suggests that movies of *Action* genre are liked and rated by users of age group 25. This contributes to the effectiveness of the proposed system. The results are illustrated in the form of a chart in Fig 4.

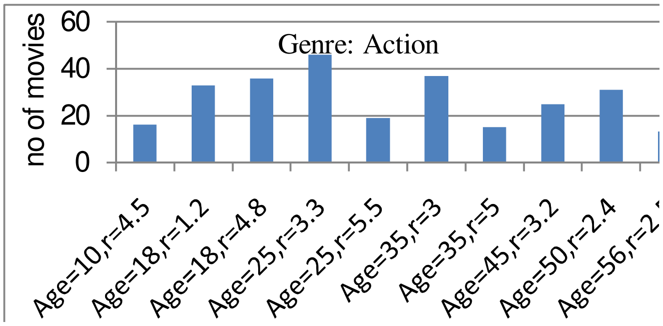


Fig 4: No of movies in each cluster of Action genre

In the similar way analyzing the pattern of clusters formed by selecting *Horror* genre shows that users of age group 18 and 25 have liked and rated the movies of *Horror* genre while the number of users of age group 10 and 50 and 56 who rated movies of this genre is very less. This is in accordance with the general observation that kids don't watch that many horror movies.

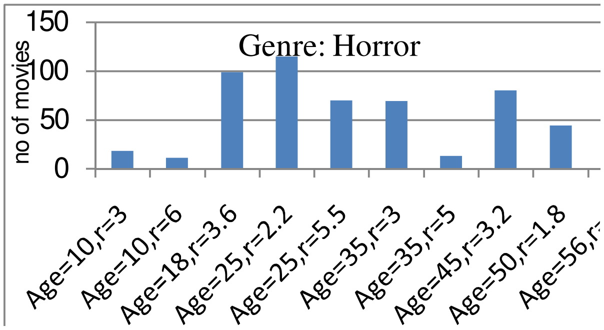


Fig 5: No of movies in each cluster of Horror genre

After having the clusters generated, the Slope One Collaborative Filtering technique is applied to a particular movie of the genre. The movie *Street Fighter (1994)* is taken as an input and the predicted ratings are computed for the movie for 10 different users. Fig. 3 highlights the difference in the actual and predicted ratings of the movie by the seven users. The predicted ratings are computed from the slope one algorithm, adjusted cosine item based scheme and bias from mean scheme.

Adjusted Cosine Item Based Scheme

In this scheme a user *u* predicts the rating of an item ***m.*** The predicted rating P(u, m) is formulated as

$$P\ (u,m) = \frac{\sum_{i=1}^{n} R_{ui} * sim(m,i)}{\sum_{i=1}^{n} sim(m,i)}$$

where

R_{ui} = Rating of user 'u' to item 'i'

sim(m, i) = cosine similarity metric between item 'm' and item 'i'

c = no of items other than 'm' rated by 'u'

$$sim\ (m, r) = \frac{\sum (R_{im} * R_{ir})}{\sqrt{\sum_{i=1}^{k} R_{im}^{2}} * \sqrt{\sum_{i=1}^{k} R_{ir}^{2}}}$$

R_{ir} =rating of r by user i

k = no of users who rate both m and r

R_{im}= rating of m by user i

The prediction is based on the user's average plus the average deviation from the user mean for the item in question. It is given by the formula

$$P(u, m) = u_{avg} + \frac{\sum_{i=1}^{n}(V_i - U_{avg})}{n}$$

where

m=item considered

u_{avg} =mean rating of user 'U'

v_i = Ratings of users on 'm'

n= No of users who rate 'm'

The predicted ratings from these schemes are tabulated in Fig. 6 and graphically shown in Fig 7. The figure demonstrates the efficiency of the approach.

User ID	**Actual Ratings**	**Our Approach**	**Adjusted Cosine**	**Bias From Mean**
103	4	3.1	3.4	4.6
117	3.3	2.8	3.7	3.2
149	3.3	2.5	4.1	4.5
210	4.5	4.6	4.8	4.6
271	3.6	2.7	3	3.8
284	1.1	0.2	1.1	0.3
302	2.2	2.3	3.2	3.6
315	1.5	1.4	1.5	2.5
329	2	2	3.8	3.4
424	2.2	2.4	3.8	3.8

Fig 6: Actual Ratings and Predicted Ratings

VI. CONCLUSIONS

The advantage of using item space partitioning followed by the K-means algorithm is that, it reduces the computation time considerably and increases the prediction accuracy. The inclusion of age along with user ratings as a determining factor of the clusters gives the user adequate options for items which the user is likely to rate. Finally, the application of Slope One algorithm on the item rated by the user increases the probability of the user being recommended with highly filtered items which he may be interested in rating based on the correlation between the item rated by the user and the other items of same category.

REFERENCES

[1] Joseph Konstan and John Riedl .*AI Techniques for Personalized Recommendation*, IJCAI 2003 University of Minnesotta,Minneapolis, pp 126-131.

[2] Mark O'Connor and Jon Herlocker. *Clustering Items for Collaborative Filtering.* ACM-SIGIR Workshop on Recommender Systems,1999,58-62

[3] Daniel Lemire and Anna Maclachlan. *Slope One Predictors for Online Rating-Based Collaborative Filtering*. Proceedings of SIAM Data Mining (SDM'05), 2005 pp 471-476

[4] 2010 About Grouplens | Grouplens Research.[Online].Available: http://www.grouplens.org/node/73

[5] Han, J., and Kamber, M. 2000. Data mining: Concepts and Techniques. New York: Morgan- Kaufman. page 135-140

[6] Daniel Lemire_ Anna Maclachlan'Slope One Predictors for Online Rating-Based Collaborative Filtering In SIAM Data Mining (SDM'05), California, April 21-23, 2005.

[7] Glenn Fung. A Comprehensive Overview of Basic Clustering Algorithms. June 22, 2001:153-158

[8] Sarwar, B. M., Karypis, G., Konstan, J. A. and Riedl, J..2001. Item-based Collaborative Filtering Recommendation Algorithms. WWW Conf. 2001, pp. 285- 295.

[9] ACM.Special issue on information filtering. Communications of the ACM, 35(12), Dec 1992.

[10] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry.Using collaborative filtering to weave an information tapestry. Communication of the ACM, 35(12):61-70, December 1992.

[11] Nicholas Ampazis.Collaborative Filtering via Concept Decomposition on the Netflix Dataset.The netflix prize challenge, SIGKDD Explorer. Newsl, 2007: 43-47

[12] [Online]Available http://en.wikipedia.org/wiki/Collaborative_Filtering