

# MovieReco: A Recommendation System

Dipankaj G Medhi, Juri Dakua

**Abstract**— Recommender Systems act as personalized decision guides, aiding users in decisions on matters related to personal taste. Most previous research on Recommender Systems has focused on the statistical accuracy of the algorithms driving the systems, with no emphasis on the trustworthiness of the user. RS depends on information provided by different users to gather its knowledge. We believe, if a large group of users provide wrong information it will not be possible for the RS to arrive in an accurate conclusion. The system described in this paper introduce the concept of *Testing the knowledge of user* to filter out these “bad users”.

This paper emphasizes on the mechanism used to provide robust and effective recommendation.

**Keywords**—Collaborative Filtering, Content Based Filtering, Intelligent Agent, Level of Interest, Recommendation System.

## I. INTRODUCTION

RECOMMENDER Systems (RS) are agent-based systems that use stored preferences to locate and suggest items of interest to users they serve. It is a combination of information filtering system and intelligent agent system. Recommender Systems have gained increasing popularity on the web, both in research system (Example GroupLens [1], MovieLens [2] etc) and online commercial site (example Amazon.com, CDNow.com etc) that offer recommender systems as one way for consumers to find products they might like to purchase.

Most of the RS use two approaches for building a user profile and computing recommendations – Collaborative Filtering (CF) and Content Based (CB) recommendation. CF is an attempt to simulate collaboration among users for sharing recommendations and reviews. The system recommends item to its user by matching his taste to that of other users in the system. CB systems recommend items based on the attribute (content) of the item rather than other users rating. Each of the approaches has advantages and disadvantages. These are discussed in [3, 4, 5, 6, 7]. CF has-

1. *Cold start problem*: There are not enough users to match with
2. *Sparsity problem*: When the system has a very high item-to-users ratio, the user item-rating matrix is typically very sparse. Therefore, the

probability of finding a set of users with significantly similar rating is usually low.

3. *First rater problem*: An item cannot be recommended unless it has been rated earlier.

CB Systems also suffer from-

1. CB can't perform in domain where there is not much content associated with items, or where the content is difficult for a computer to analyze
2. The system can only suggest items whose content match with the user's profile

Moreover, the RS system gathers its knowledge from the rating (a vote based system) provided by the user. If any user provides wrong information intentionally or unintentionally, the system may mislead. Although wrong information of one or two users will affect the result negligibly, but if more users are of this category, it will not be possible for the RS to arrive at accurate conclusion. We named it as *bad user* problem.

We overcome these problems by exploiting the feature of CB and incorporate it with CF. We considered both the CF and CB approaches for recommending items to users. We solved the *bad user* problem by *testing the knowledge of a user*. In this paper, we present the framework of this new approach. We applied this approach to movie domain and show that this mechanism can be used to provide robust and effective recommendation.

## II. PREVIOUS WORK

There are many attempts to combine content-based information filtering with collaborative filtering. One of the hybrid systems that combine these two strategies is PTV [8], a Java based client/ server application designed to produce personalized electronic programme guide for digital TV. They used CB and CF method to produce separate recommendation and directly combine their predictions. In Pazzani's approach [9] prediction is made by applying CF to users' profile. Basu et al [7] viewed recommendation as a classification problem and used inductive learning system called ripper. Both collaborative and content information is fed to the ripper. After being trained on some example, ripper predicts whether movie will be liked or disliked by the user. Good et al [10] used collaborative filtering along with a number of personalized information filtering agent. Melville et al [4] employed a technique where the system first complements the user data using the content information and then uses collaborative filtering to compute recommendations.

Manuscript received January 20, 2005. This work was done when the authors were working in Sikkim Manipal University, Sikkim, India

Dipankaj G Medhi is presently a master degree student at Asian Institute of Technology, Pathumthani, Thailand phone:066-2524-7891; e-mail: dgm\_000@yahoo.com, DipankajGobinda.Medhi@ait.ac.th).

Juri Dakua is now a master degree student at Indian Institute of Technology, Kharagpur, India (e-mail: juridakua@yahoo.com).

### III. SYSTEM ARCHITECTURE

The Recommendation system, named MovieReco consists of

- *User Interface Module:* the user interface module is responsible for communicating between the users and the agent. This WWW interface is used for users login, registration (for new user), obtaining rating from the users and providing recommendation to the active user. As MovieReco uses a self-growing movie database, the user can add new movie in the movie database (not shown in fig 1) using this interface. As this paper emphasizes on the process of successful recommendation, we will nowhere explain other features of MovieReco such as growth of movie database etc. Recommendations are made available to the user as soon as he logs in. Based on the predicted rating, a film is strongly recommended, recommended or not recommended to the current user

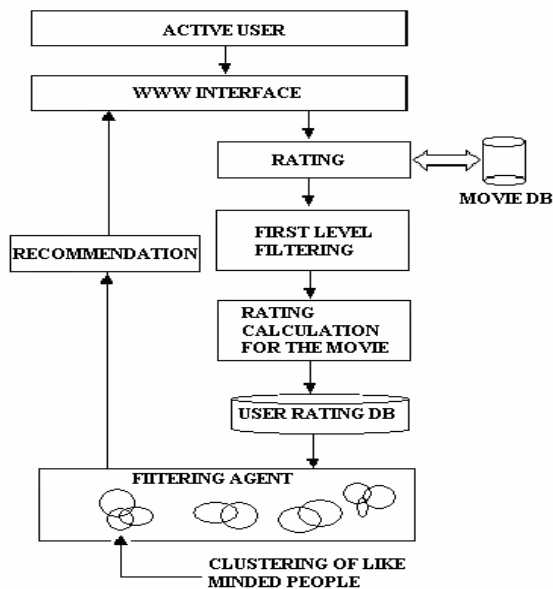


Fig 1 Partial view of the data flow in the system

- *The Learning Module:* The learning module is responsible for ensuring that the set of profiles is personalized to the interests of the user. It handles the task of mapping user interests to the set of user profiles. The only way for gathering knowledge about any user is to provide some films randomly from the movie database to the active user and ask him to rate them. The user needs to enter the name of three actors, two actresses, name of the director and type of movie along with the voting to each of these attributes. The learning module calculates the overall rating for the movie and stores the data in user-rating-database along with each attribute. Based on this information, the agent arrives to the conclusion about the
  - Level of interest of the user

- The knowledge of the user on those films he has rated

The system filters out the bad users based on these conclusions.

- *Information Filtering Module:* The information-filtering module is responsible for actual retrieval and selection of movies from the movie database. Based on the knowledge gathered from the learning module, information filtering process is done. Details of this module are explained in section 5.

### IV. SYSTEM FUNCTIONALITY

The system provides the following functionality -

- *Standardization of scale:* Users are asked to rate movie on a scale from 1 to 7 with 1 being the lowest and 7 being the highest. The main characteristic of this scale should be absolute. In other words, a score given by one user should have the same value for same score given by other users. For example, if any two users  $U_1$  and  $U_2$  rated 7 to movie  $F_1$ , it means that  $F_1$  is one of the favorite few for them. On the other hand, if  $F_1$  is one of the favorite few for users  $U_1$  and  $U_2$ , the rating provided by them must be 7. But practically, some users are accustomed with higher rating and others are with lower. They may use different styles of rating for the same level of interest. MovieReco asks the user to rate the best movie he has ever seen to know the style of rating for that user. Fig 2 shows the distribution of rating for best movies provided by different users. From

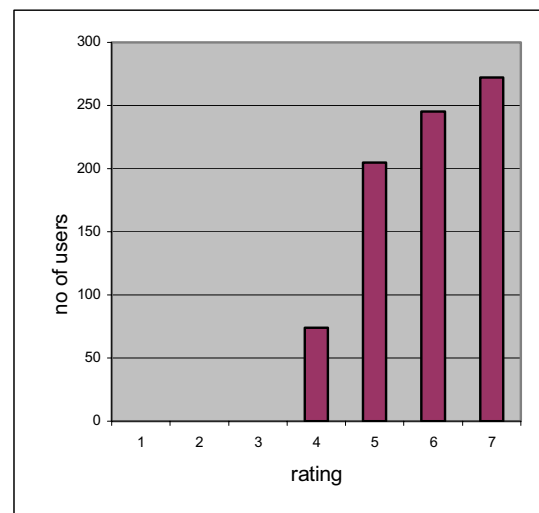


Fig 2: Distribution of rating for best movie this distribution we can conclude for the necessity of standardization of the scale according to each user. For that, each user is advised to rate three best movies he has ever seen. Then a standardization factor  $f_s$  is calculated as

$$f_s = \frac{\text{highest value}}{\frac{1}{3} \sum_{i=1}^3 b_{r_i}}$$

where highest value is 7 and  $b_{r_i}$  is the best rate assigned by the user to movie i. Next time, whenever a user assigns a rate to a movie it is multiplied by  $f_s$ .

- **Testing the Knowledge of User:** The idea that is not introduced in any existing system, as far knowledge, is the *Testing the Knowledge of the Users*. Now the obvious question is – why it is necessary? The answer is – collaborative filtering approach totally depends on the rating of different users. If any user rates a movie blindly without knowing anything about the film, then it may affect the result. Although wrong information of one or two users will affect the result negligibly, but if more users are of this category then it may not be possible for the RS to calculate the accurate estimated score for *gentle and wise users*.

Human mentalities differ from man to man. Thus we speculate that, the users of varying mentality use a software agent in different point of view. Some of them rate a movie very seriously and some of them lightly, and few of the users have intention to mislead the agent. The user of the last category is dangerous for any agent. For filtering out them we are introducing the concept of *Testing the Knowledge of User*. The user has to enter the name of three actors, two actresses, name of the director and type of movie along with the corresponding rating to each of these attributes. The WWW interface for this module is shown in fig 3. Thus the system will come to know about the depth of the knowledge for the current user.

Attribute	Ajaana	Jaan Tere Nam	Hum Dono
Actor1	<input type="text"/> 7 ▾	<input type="text"/> 7 ▾	<input type="text"/> 7 ▾
Actor2	<input type="text"/> 7 ▾	<input type="text"/> 7 ▾	<input type="text"/> 7 ▾
Actor3	<input type="text"/> 7 ▾	<input type="text"/> 7 ▾	<input type="text"/> 7 ▾
Actress1	<input type="text"/> 7 ▾	<input type="text"/> 7 ▾	<input type="text"/> 7 ▾
Actress2	<input type="text"/> 7 ▾	<input type="text"/> 7 ▾	<input type="text"/> 7 ▾
Director	<input type="text"/> 7 ▾	<input type="text"/> 7 ▾	<input type="text"/> 7 ▾
Type	<input type="text"/> 7 ▾	<input type="text"/> 7 ▾	<input type="text"/> 7 ▾

Have not seen   
  Have not seen   
  Have not seen

Fig 3. Partial view of WWW interface for *Testing the Knowledge of User* module

- **Rating Scheme:** We now discuss how we store user preferences. For knowing the user interest, we are using four dimensions to refer to user preferences for each movie. We are classifying these dimensions as
  - Actor: Name of three leading actors
  - Actress: Name of two leading actresses
  - Director: Name of the director
  - Genre: Type of the movie

For example user  $U$  may like movie  $F$  because actor  $A$  and actress  $B$  is in the cast, the movie is directed by  $D$  and the genre is action.

From a list of seven alternatives i.e.  $P = \{a_1, a_2, a_3; c_1, c_2; d; g\}$ , where  $a_i \in$  actor,  $c_j \in$  actress,  $d \in$  director and  $g \in$  genre, any user has to rate element of  $p_i \in P$  on a scale from 1 to 7. So, actual rating on a film  $f_i \in F$ , where  $F$  is the set of all films, not rated by user  $u_i$  can be calculated as

$$r_{u_i f_j} = f_{s u_i} \sum_{k=0}^7 r_k$$

where  $r_{u_i f_j}$  is the standardized rate of user  $u_i$  on film  $f_j$ ,  $r_k$  is the rate assigned by user  $u_i$  to  $a_i, c_j, d$  and  $g$  and  $f_{s u_i}$  is the standardization factor of user  $u_i$ .

### V. ALGORITHM USED FOR FILTERING

After passing out the test of user knowledge, the standardized ratings provided by the user are stored in the rating database. Based on the data in the rating database, a film is recommended to the user  $u_i$  using the following steps

- Assume  $M =$  Total number of users  
 $N =$  Total numbers of films  
 $n =$  Total number of films not rated by user  $u_i$
- 1) For each film  $F \in n$  not rated by user  $u_i$ , find the correlation with each of the other  $(N-1)$  films.
  - 2) Based on the correlation coefficient values select  $S$  films, which is mostly closely correlated with  $F$ . This will form a group of  $S$  similar films with  $F$ .
  - 3) Find the correlation of all users with the current user  $u_i$  based on the rating given by every user to those similar films. Based on the correlation coefficient values, select  $X$  users, which are most closely correlated with user  $u_i$ . Thus it will form a group of  $X$  users similar with user  $u_i$ .

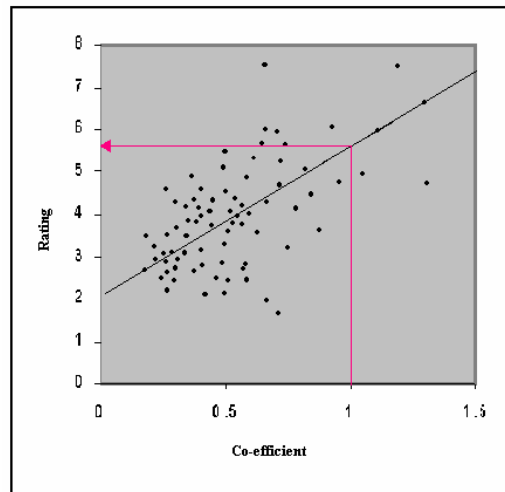


Fig 4 Ratings plotted against correlation

- 4) Plot the rating  $r_j$  of all  $X$  users for film  $F$  against the correlation coefficient value of the similar group.
- 5) Find the best fitting straight line through the points plotted in step 4. Let us consider any point  $(r, c)$  in this straight line. If the value of  $c=1$  (ie the Co-

efficient = 1), then the corresponding  $r$  value is the estimated rating for that film ( shown in fig 4).

This algorithm can be said as combination of Item-Item algorithm proposed by Karl Sim and Pearson-r algorithm used in GroupLens project. Otherwise, it can be said as a simplification of the MORSE algorithm proposed by D Fisk [11].

## VI. DISCUSSION

The intension of this work is to go beyond a “proof of concept”. As the underlying system can’t be well conveyed without an implementation, MovieReco was initially advertised locally in an intranet. The system resided on the Server and users were asked to access it from various node connected in the intranet of our department.

Out of the first hundred users, nineteen of them were not able to pass the *test of user knowledge*. When the database was growing and more and more users were joining the system, this percentage was reduced to 13.23%. This 13.23 will not remain same, when the database will grow up totally. However, it can be concluded that, for better performance, the wrong information should be filtered out.

In pure CF system, a prediction can’t be made for an item unless some users previously rated it. We are overcoming this problem by applying pure CB approach.

Due to sparsity problem, the probability of finding out the best neighbor is usually low. In our approach we are considering a hypothetical user having correlation co-efficient = 1 (see section 5) and this hypothetical user is the nearest neighbor of the current user.

For finding out the performance of our system, we had removed the rates of some movies provided by some of the users. After that, we had applied our algorithm to find the ratings for those movies. From this experiment we are getting around 96.53 % of successful result.

## VII. CONCLUSION

In this paper, we have presented the internal details of a movie recommender system. In particular, we have stressed on a movie selection mechanism based on collaborative as well as content based filtering, that uses stored user preferences for different movie dimensions. The scheme used provides desirable guarantees about the nature of recommendation produced and is also robust to variation of user interests. There are several ways in which our movie recommendation system can be enhanced.

The proposed scheme is not limited to just movie recommender systems. In fact, it can be used in any domain.

## REFERENCES

- [1] Konstan J A, Miller B N, Maltz D, Herlocker J L, Gordon L R and Riedl J *GroupLens : Applying Collaborative Filtering to Usenet News* Communication ACM 40, 3 (page 77-87)
- [2] Konstan J A, Reidl J, *Explaining Collaborative Filtering Recommendation* ACM 2000 Conference on Computer Supported Collaborative Work.
- [3] H. Herlocker J, Konstan J A, Borchers A I, Reidl J *An Algorithmic Framework for Performing Collaborative Filtering* In Proceedings on 22<sup>nd</sup> Annual International ACM SIGIR Conference on Research and Development in Information Retrieval,
- [4] Melville P, Mooney J R, Nagarajan R *Content Boosted Collaborative Filtering for Improved Recommendations*. In 18<sup>th</sup> National Conference on Artificial Intelligence, Pages 187-192. American Association for Artificial Intelligence, 2002.
- [5] Claypool M, Gokhale A, Miranda T, Murmikov P, Netes D and Sartin M, *Combining Content-based and Collaborative Filters in an Online Newspaper*. In proceedings of ACM SIGIR Workshop on Recommender Systems 1999J.
- [6] Sarwar M B, Konstan A J, Borchers A I, Herlocker J Miller Brad and Riedl J *Using Filtering Agents to Improve Prediction Quality in the GroupLens Research Collaborative Filtering System*. In proceedings of the 1998 ACM Conference on Computer Supported Collaborative Work, Pages 345-354, ACM press 1998
- [7] Basu C, Hirsh H, and Cohen W *Recommendation as Classification :Using Social and Content-based Information in Recommendation* In Proceedings of the 15<sup>th</sup> National Conference on Artificial Intelligence, Pages 714-720, American Association for Artificial Intelligence 1998
- [8] Cotter P. and Smyth B 2000 *PTV: Intelligent Personalized TV Guides* In 12<sup>th</sup> Conference on Innovative Application of Artificial Intelligence page 957-964
- [9] Pazzani M J 1999 *A framework for Collaborative, Content-based and Demographic Filtering*. Artificial Intelligence Review 1395-60 page 393-408
- [10] Good N, Schafer J B, Konstan J A, Borchers A, Sawar B, Herlocker J, Riedl J *Cominin Collaborative Filtering with Personal Agent for Better Recommendation* In proceedings of 16<sup>th</sup> National Conference on Artificial Intelligence (AAAI 99) page 439-446
- [11] D Fisk: An Application of Social Information Filtering to Movie Recommendation BT Technol Journal, 14, No 4, page 124-132