A STUDY ON IMPLICIT FEEDBACK IN MULTICRITERIA E-COMMERCE RECOMMENDER SYSTEM

K.Palanivel
Research Scholar in Computer Science
AVVM Sri Pushpam College
Bharathidasan University
Thanjavur, Tamil Nadu, INDIA
palani.avcc@hotmail.com

R.Sivakumar
Head, Department of Computer Science
AVVM Sri Pushpam College
Bharathidasan University
Thanjavur, Tamil Nadu, INDIA
rskumar.avvmspc@gmail.com

ABSTRACT

Recommender systems are personalized intelligent systems capable of helping people to easily locate their relevant information through recommendations from a large repository of information. In order to provide personalized recommendations, the accurate modeling of user's preferences is required. Modeling of user's preferences needs their relevance feedback on the recommendations. The relevance feedback may be collected either explicitly or implicitly. The explicit relevance feedback introduces *intrusiveness* problem whereas the implicit feedback can be inferred from normal user-system interactions without disturbing the user. The users expect *accuracy* in recommendations and *effortless* assistance from the recommender systems. The multicriteria user preference ratings are useful to improve the accuracy of recommendations. However, collecting multiple ratings increases the cognitive load of the user. We believe that a combined, implicit relevance feedback and multicriteria user preference ratings, approach improve the accuracy in recommendations and eliminate the intrusiveness problem of recommender systems. In order to fulfill the above needs and to better understand the potential behind the implicit relevance feedback approach under multicriteria ratings context, this study focuses a new *implicit-multicriteria* combined recommendation approach. A Music recommender system is developed for this experiment to evaluate the recommendation accuracy of implicit and explicit feedback approaches under the user-based and item-based prediction algorithms against different data sparsity levels, training/test ratio and neighborhood sizes. Out of this experiment, the implicit ratings based prediction algorithms provide better performance than the explicit ratings based prediction algorithms with respect to all the three sensitive parameters. It is also observed that the proposed IB_PIR prediction algorithm computes better predictions than other prediction algorithms. Finally, we discuss the study's implications for theory and practice and conclude with many suggestions for future research on non-intrusive, multicriteria recommender systems.

Keywords: implicit relevance feedback, evaluation, multicriteria E-commerce recommender system.

1. Introduction

The tremendous growth of both information and usage has introduced a so-called *information overload* problem in which users are finding it increasingly difficult to locate the right information at the right time [Resnick, P., et al., 1994]. To alleviate the information overload problem, *Personalization* becomes a popular remedy to customize the Web environment towards user's preferences [Chen, L., and K. Sycara, 1998]. Among all personalization tools, *Recommender systems* are the most employed tools in e-commerce businesses [Shahabi, C., and Y. Chen, 2003] that use the opinions of members of a community to help individuals in that community by identifying the products most likely to be interesting to them [Konstan, J., 2004]. In E-commerce field, many recommender systems have emerged in the past few years to help the users in their search process to find out the most suitable items (such as movies, songs, CDs, books and so on) according to their preferences [Linden, G., et al., 2003; Nageswara Rao. K and V.G. Talwar, 2008]. Using this kind of personal assistance, the commercial websites promote their sales and achieve higher profits. E-commerce forces companies to find new ways to expand the markets in which they compete, to attract and retain customers by tailoring products and services to their needs, and to restructure their business processes to deliver products and services more efficiently and effectively [Shin, N., 2001]. The process of navigation and search on the Web site should require minimal effort, and the Web site should be designed in such a way that it facilitates product search within a short amount of time and provide users with a good sense of control over the interaction [Nah, F.F., and S. Davis, 2002]. Website customizations have drawn considerable attention from the fields of business and web design [Timothy K. F. Fung, 2008].

The user preferences are gathered from many sources such as the item’s content features, user-item ratings data, user’s demographic information, knowledge of the items and utility information of items [Martinez, L., et al., 2009]. According to the information gathered by the system and the technique that ranks the items in order to suggest recommendations, the recommendation approaches are classified into different types. There are mainly three approaches in computing recommendations. *Content based filtering* selects items to recommend based on the correlation between the content of items and the user's profile [Pazzani, M.J., J. Muramatsu and D. Billsus, 1996]. *Collaborative filtering* chooses items based on the correlation between users with similar preferences [Goldberg, D., et al., 1992]. *Hybrid filtering* approach tries to avoid certain limitations related to content-based and collaborative filtering [Burke, R., 2002]. In addition to these three approaches, there are other two filtering approaches. They are *Knowledge-based* and *Utility-based*. *Knowledge-based filtering* computes their recommendations using case-based reasoning processes [R. Burke, 2002]. *Utility-based filtering* provides recommendations based on the calculation of the utility of each item according to the user’s interests [Guttman, R.H., 1998]. Since it is one of the commercially successful techniques in Recommender systems and we want to investigate multicriteria ratings based system, this study focuses on the collaborative filtering approach.

The main challenge in modeling user preferences is to build a user profile without disturbing the user [Niinivaara, O., 2004]. In order to learn the user preferences, usually the recommender systems get the relevance feedback from the users explicitly and/or implicitly and store the user feedback in the form of ratings in user-item ratings matrix to compute future recommendations [Claypool, M., et al., 2001]. The recommendation quality depends on the quality of available user feedback data [Sarwar, B., 2000]. It is because of the reason that in implicit feedback, there exists transparent monitoring of user behaviors using interest indicators such as time spent on viewing the product, number of accesses to a product and so on [Kelly, D., and J. Teevan, 2003]. These interest indicators can be used to infer the user’s preferences on products and produce useful recommendations. Even though the explicit feedback indicates the user preferences clearly, it wastes the user’s effort, time, and cost [Jung, K., 2001; Kellar, M., et al., 2004] and its effectiveness is limited due to the following reasons:

- When the users are required to stop their actions to enter explicit rating, it will somewhat alter the normal patterns of user’s browsing behavior [Claypool, M., et al., 2001; Jung, K., 2001]. The explicit ratings often distract the attention of the users and divert the user normal operation.
- Users like to spend more time only on seeing the items than rating the items. GroupLens System [Sarwar, B., et al., 1998] found that the users were reading a lot more articles than they were rating.
- Sometimes the users have the difficulty in expressing their interest explicitly on numeric scale [Morita, M., and Y. Shinoda, 1994]. In a five-scale user rating (1-5), the user may give different ratings to the same item at different times and situations, due to the difficulty in making a distinction between ratings 3 and 4. Similarly, the same rating 4 in a scale of (1-5) given by two users does not necessarily imply the same degree of interest in an item [Zenebe, A., and A.F. Norcio, 2009].
- Many users assign arbitrary ratings that do not reflect their honest opinions [Lee, T.Q., et al., 2008].

On the other hand, the implicit feedback can be preferred because of the following benefits:

- It reduces the cost of rating items by saving the user’s valuable time in examining and rating items [Claypool, M., et al. 2001].
- The system can infer what the user requires based on user-system interactions [Jung, K., 2001]. Every user’s interaction with the system can contribute to implicit feedback.
- It can be generally thought to be less accurate than explicit feedback, but large quantities of implicit data can be gathered at no extra cost, effort and time from the users. The implicit ratings remove the cost of the evaluator in examining and rating the items. Even though there remains a computational cost in storing and processing the implicit rating data, this can be hidden from the user [Jung. K, 2001].
- It can be continuously collected from the user-system interactions and can be used for the user profile updation.

By keeping the above said factors in mind and to understand the implicit feedback approaches better under multicriteria ratings context, this is study is conducted.

2. Literature Review

2.1. Implicit Relevance Feedback

It is empirically proved that the user's interests can be inferred from his behavior [Chatterjee, P., et al. 2003]. Some of the implicit methods obtain relevance feedback by analyzing the links followed by the user [Liberman, H., 1995; Mladenic, D., 1996], a history of purchases [Amazon, 2003; CDNow, 2001; Krulwich, B., 1997], the navigation history [Cooley, R., et al., 1999; Mobasher, B., 2000], email boxes [Huberman, B., 1996] and the time spent on a particular web page [Morita, M., and Y. Shinoda, 1994, Konstan, J., et al., 1997, Kobsa, A., et al., 2001, Sakagami, H., et al., 1997]. It is also mentioned that the limited evidence available on implicit feedback suggests that it has great potential, but its effectiveness remains unproven [Montaner, M., et al. 2003; Oard, D.W., and J. Kim, 1998]. Using the implicit feedback in recommender systems greatly decreases the user efforts, whereas providing the explicit feedback helps the system to collect user preferences accurately [Godoy, D., and A. Amandi, 2005; Mobasher, B., and S.S. Anand, 2005]. In another work [Adomavicius, G., and Alexander. T, 2005], the authors mentioned that the non-intrusive ratings (such as time spent on reading an article) are often inaccurate and cannot fully replace explicit ratings provided by the user. The problem of minimizing intrusiveness while maintaining certain levels of accuracy on recommendations need to be addressed by the Recommender system researchers. In another work **[**Jung, S., et al., 2007], the authors mention that among the many issues that still need to be resolved, most of them are regarding the reliability of implicit feedback data. A number of experiments have been conducted to evaluate the reliability of implicit relevance measures in Web search/browsing context [Claypool, M., et al., 2001; Jung, K., 2001; Fox, S., et al., 2005; Joachims, T., et al., 2007; Hingston, M., 2006]. Limited studies are available to show the potential of implicit relevance feedback in Recommender system context [Oard, D., and J. Kim, 1998; Papagelisa, M., and D. Plexousakis, 2005]. In implicit feedback, the system learns the user preferences by observing the user's behavior and it is a valuable alternative that has received increased attention in recent years [Jung, S., et.al., 2007; Joachims, T., et al., 2007].

2.2. Multicriteria Ratings

Many recommender systems have been developed by using the implicit relevance feedback and all these systems are based on single (user-item) ratings matrix only [Montaner, M., et al. 2003]. Multicriteria ratings provide information about the user preferences for different aspects of an item [Adomavicius, G., and Y.O. Kwon, 2007; Adomavicius, G., and Alexander. T, 2005; Lee, H.H., and W.G. Teng, 2007]. For example, the overall user rating for a movie gives the general user preference on that movie. But, the multicriteria ratings of a movie such as the ratings for Action, Direction, Story, Music and so on., provide in-depth knowledge about the user preferences in that movie. The recommender systems should benefit from leveraging this additional information because it can potentially increase the recommendation accuracy. A few Recommender systems have begun to use the multicriteria ratings. However, these systems are not used in the personalization context. Therefore, taking the full advantage of the multicriteria ratings in personalization applications require new recommendation techniques [Adomavicius, G., and Y.O. Kwon, 2007].

Even though the multicriteria ratings give accurate results, it increases the intrusiveness problem by collecting more ratings explicitly from the user [Adomavicius, G., and T. Alexander, 2005]. The implicit feedback approach eliminates the intrusiveness problem, at the same time, it may give less accuracy on user preferences [Jung. K, 2001]. In order to overcome these two limitations, we combine the implicit feedback and the multicriteria ratings based approaches and compare the efficiency of the implicit and the explicit feedback methods under the *user-based* and *item-based* prediction algorithms [Papagelis, M., and D. Plexousakis, 2005]. The *time spent on hearing the music* and the *number of accesses to a music item* is used as implicit measures of interest in a Music Recommender system developed for this experiment. In this system, both the multicriteria explicit and implicit ratings are collected and used for prediction. From the literature also, it is understood that no efforts have been made to evaluate the efficiency of the implicit and explicit relevance feedback approaches under multicriteria ratings based recommender systems context [Claypool, M., et al., 2001; Adomavicius, G., and T. Alexander, 2005; Adomavicius, G., and Y.O. Kwon, 2007]. Hence, we made a novel attempt in this research direction.

3. Overview of the Proposed Approach

In multicriteria based Recommender systems, the user-item ratings matrix contains user ratings on items in multiple aspects (criteria) as shown with a typical example in table 1. The modern Recommender systems need new sophisticated data structures and approaches to produce useful recommendations. When the recommender system gives recommendations based on the available ratings data, the user views some interesting items and gives his relevance feedback explicitly, for example, in three aspects namely the quality of music, lyric and voice. The system then stores these user ratings in the user-item ratings matrix for future predictions. In this study, the recommender system performs the following steps to produce recommendations [Shih, T.K., et al., 2002] using three kinds of rating matrices, namely *multicriteria explicit user-item ratings matrix, criteria-category (partial) implicit preferences matrix and implicit user-item ratings matrix.*

- Construction of three kinds of user-item ratings matrices.
- Weigh all users with respect to similarity with the active user using statistical technique.
- Select a subset of similar users (*neighbors*) to use as predictors.
- Normalize the ratings and compute the prediction from weighed combination of selected neighbor's ratings.
- Present items with highest predicted ratings as recommendations.

In the first step, in order to evaluate the effectiveness of implicit and explicit feedback approaches under multicriteria ratings context, the above said three kinds of matrix representations are proposed to maintain the user preferences. The first matrix has user ratings collected explicitly by asking the user, the second matrix contains the user preferences on categories within every criterion and it is derived implicitly from the explicit ratings and the third matrix contains implicit interest ratings derived from implicit interest measures. These proposed matrices are then used to find the similar users and to compute the prediction on user preference.

3.1. User-item Matrix with Explicit Ratings

Assume that there are five users $u_1, \ldots, u_5$, four items $i_1, \ldots, i_4$ and a typical multicriteria user-item ratings matrix containing ratings in multiple aspects at a moment of time with a scale of 1 to 10 is shown in table 1. The overall rating is calculated by simply taking average of the multicriteria ratings. In table 1, it is clear that the user u_3 has different multicriteria preferences when compared with the user u_2 even though their overall ratings for every music items match perfectly. The users' u_4 and u_5 have close similarity with the user u_2 in this example since not only their overall ratings are similar but also their individual criteria preferences. From table 1, it is understood that the single-ratings based Recommender systems hide the true similarity information and lead to inaccurate results. The multicriteria ratings provide some insights regarding why the users like an item and help to compute accurate recommendations. Two additional rows are maintained in addition to the regular user-item ratings matrix in this study. After the five user ratings, the first row (popularity) represents the total number of positive ratings received by the items from all the users. This popularity information of items, at a particular moment of time, is used to provide initial recommendations and it is treated as a source to eliminate the *new-user* problem [Martinez, L., et al., 2009].

Table 1: Multicriteria (Music, Lyric, & Voice) User-Item Ratings Matrix

Overall rating   Multicriteria ratings

	Item 1	**Item 2**	**Item 3**	**Item 4**
User 1	**0** (0, 0, 0)	**2** (2, 3, 1)	**0** (0,0,0)	**6** (7, 5, 6)
User 2	**4** (4, 3, 5)	**5** (3, 4, 8)	**6** (6, 4, 8)	**?** (?, ?, ?)
User 3	**4** (9, 1, 2)	**5** (8, 3, 4)	**6** (5, 9, 4)	**7** (9, 4, 8)
User 4	**4** (5, 2, 5)	**5** (4, 4, 7)	**6** (6, 5, 7)	**9** (8, 10, 9)
User 5	**4** (4, 4, 4)	**5** (3, 5, 7)	**6** (5, 5, 8)	**9** (9, 9, 9)
Popularity (+ve votes)	**0/4**	**0/5**	**4/4**	**4/4**
<u>Content features</u> ***Music*** ***Lyric*** ***Voice***	 Rahman(M_1) Muthu(L_1) Doss(V_1)	 MSV(M_2) Muthu(L_1) Balu(V_2)	 Rahman(M_1) Vijay(L_2) Balu(V_2)	 MSV(M_2) Vijay(L_2) Doss(V_1)

Similar users

When a new user is logged in, the system provides initial recommendations based on the popular items in order to learn about the user preferences. The popularity can be obtained by:

$$Popularity_j = Count\_positive\ (r_{ij})\ for\ 1 \leq j \leq n,\ 1 \leq i \leq m,$$

where *m* is the number of users, *n* is the number of items and r_{ij} is the rating of i^{th} user on j^{th} item. The rating is taken as positive if it is greater than the threshold (which is taken as >5 in our study). The last row of the matrix contains the *content features* of the items. Each column represents values of item's content features (i.e., *music*, *lyric* and *voice*). In table 1, the *Music* content feature contains two types of values (names of music directors), considered as *categories* M_1 and M_2 *(Rahman* and *MSV).* Similarly, the *Lyric* feature has two categories of lyric writers L_1 and L_2 (*Muthu* and *Vijay*). The *Voice* feature contains two categories of singers V_1 and V_2 (*Doss* and *Balu*). These content features are used to identify the reason behind the user's choice of rating for an item and the reason for user's likes and dislikes. For example, the user may give higher rating when he likes a particular music director's music, lyric of a particular writer, or the voice of a particular singer. These item features are used to identify the user's category-wise preferences within every criterion. Using these category-wise user's preferences, it is possible to identify the user's current interest on categories more accurately. These criteria-category preferences are used to solve the *new-item* problem [Martinez, L., et al., 2009]. When the user enters a new item into the database, the new item is also considered for recommendation based on the content features.

3.2. Criteria-category User Preferences Matrix

Assume that there are *five* users, *four* items and *three* item-criteria C = $\{c_1, c_2, \ldots, c_3\}$. From the last row of table 1, the *music* criteria values cluster the items into different music categories in table 2 based on music directors (*Rahman* and *MSV* are treated as two music categories represented as M_1 and M_2 respectively). Similarly, the *lyric* criteria in table 1 implicitly cluster the items in table 2 based on two lyric writer categories (*Muthu and Vijay* denoted as L_1 *and* L_2). And the *voice* criteria implicitly cluster the items into two singer categories (*Doss and Balu* represented as V_1 *and* V_2). NM_1 represents the total number of user's ratings on the *music* category M_1. Similarly, NM_2, NL_1, NL_2, NV_1 and NV_2 are the total number of user's ratings on the respective categories. Table 2 is a typical criteria-category user preferences matrix derived from explicit ratings in table 1. Whenever the user provides an explicit rating to an item, the system identifies the item's category in table 2 in all the criteria, using the content features of table 1. The rating value in the respective category element in table 2 is calculated by finding the average rating. It is computed as *Avg. criteria-category rating = Round[{ (previous avg. rating x number of ratings) + current explicit rating } / (number of ratings +1)]*. For example, assume that the user 3 gives an explicit rating 9 to *music* criteria of item 2, then the item 2's respective music category (i.e., M_2) is identified using content features in table 1 and in the corresponding *music* category column M_2 of table 2, the average rating value is calculated and stored. Now, the current average rating = Round[{ (6x2) + 9 } / (2+1)] = 21/3 = 7. Hence, the average criteria-category rating of (user$_3$, category M_2) is changed from rating 6 to 7 and the number of ratings (NM_2) is changed from 2 to 3. Similarly the average rating is calculated for all the criteria and the values of item features (*Music*, *Lyric* and *Voice*) in table 1 are used to cluster the item's ratings into different categories in table 2.

An element a_{ijk} represents an average rating given by the i^{th} user within the j^{th} criteria under k^{th} category. For example, a_{232} represents an average category rating of user *2*, within criteria *3* (*voice*) under category 2 (*second singer*). These kind of criteria-category user preferences are used to know the user's current preferences more accurately on different categories within each criterion in more detail and help to provide useful recommendations.

Table 2: Criteria-Category Preferences of users

CRITERIA	***Criteria c_1 (Music)***				***Criteria c_2 (Lyric)***				***Criteria c_3 (Voice)***			
CATEGORIES	M_1	NM_1	M_2	NM_2	L_1	NL_1	L_2	NL_2	V_1	NV_1	V_2	NV_2
User 1	0	0	4	2	2	1	3	1	3	1	1	1
User 2	4	2	2	1	4	2	2	1	3	1	8	2
User 3	7	2	6	2	2	2	7	2	5	2	4	2
User 4	6	2	6	2	3	2	8	2	7	2	7	2
User 5	5	2	6	2	5	2	7	2	7	2	8	2
	'*p*' categories in c_1				'*q*' categories in c_2				'*r*' categories in c_3			

Suppose that there are *p* categories in music criteria, *q* categories in lyric criteria, *r* categories in singing criteria and *m* users, then this matrix size will be [$m \times 2(p+q+r)$]. Since adapting to changes in user's interests is one of the major challenges in recommender systems, the changes in user's interest against different categories can be easily maintained in table 2 [Godoy, D., and A. Amandi, 2005].

3.3. User-item Implicit Ratings Matrix Based on Interest Measures

In order to avoid the intrusiveness problem in Recommender systems, we measure the user's interest rating on an item implicitly using the two implicit interest measures known as *time spent on hearing a music item* and *number of accesses to a music item*. These two implicit interest indicators are measured continuously from normal user-system interactions without disturbing the user. First, these two indicators are used to measure the *user interest level* on music items. Then the user interest level is used to infer his interest on multicriteria. The table 3 contains typical values representing the user's number of accesses on music items.

Table 3: First Implicit interest indicator (No. of accesses to a Music item) Matrix

	Item 1	**Item 2**	**Item 3**	**Item 4**
User 1	0	1	0	1
User 2	1	2	1	0
User 3	1	1	2	1
User 4	2	1	1	3
User 5	1	2	3	2

Table 4: Second Implicit interest indicator (Total time spent on hearing a Music item in seconds) Matrix

	Item 1	**Item 2**	**Item 3**	**Item 4**
User 1	0	500	0	924
User 2	494	1180	764	0
User 3	512	672	1584	944
User 4	1164	768	828	3606
User 5	486	1436	3068	2256
Play time	1280	1240	1320	1380

Whenever the user accesses a music item, the respective matrix element is increased by one in table 3. If the user accesses an item frequently many number of times, it implicitly means that the user has more interest on that item [Kellar, M., et al., 2004]. In this study, based on the experience gained from the experiment, the interest weigh is assigned as 0.25 as a reasonable weigh for every user accesses on a particular music item. Every time the user accesses an item, the user's interest weigh on that item is increased by 0.25. If the user accesses an item 4 times, it is treated as the full interest of user on that item (i.e., 4 x 0.25 = 1). Since the number of accesses to an item may be greater than 4 in real-time, this work considers the *number of user accesses* ≥ 4 as the user's full interest on the music item.

Similarly, table 4 contains typical values representing the user's total time spent on hearing a music item (in seconds). Whenever the user accesses a music item, the current time spent on that music item will be added to previous total time spent on the item available in respective user-item matrix element of table 4. If the user spends more amount of time, it implicitly means that the user has more interest on that item [Kellar, M., et al., 2004]. The last row of table 4 contains the actual play time of a music item (in seconds). The interest weigh based on time spent on a music item is calculated through dividing the user's time spent on a music item by the item's actual play duration. Since the user may reveal his preferences through any one of these two interest measures, the maximum weigh of the two measures are used to calculate the user's interest level on the item. In order to convert the interest weigh fraction into a rating scale of 1-10, the interest weigh is multiplied by 10 and rounded off to an integer value. The tables 3 and 4 are the intermediate tables and from these tables, the interest level of user *i* on item *j* is estimated using the following formula. The derived interest level matrix is given in table 5.

$$\text{User\_interest\_level}_{ij} = \text{ROUND}\left(\text{MAX}\left[\frac{\text{Total time spent on hearing}_{ij}}{\text{No. of access to an item}_{ij}\ \text{X Actual play duration}_j},\ \frac{\text{Min (No. of access}_{ij},\ 4)}{4}\right] \text{x } 10\right)$$

For example, *User_interest_level* $(user_5, item_3)$ = Round(Max [3068 / (3 x 1320), Min(3, 4) / 4] x 10)
= Round(Max [0.78, 0.75] x 10) = Round(0.78 x 10) = 8.

The interest level matrix represents the user's overall preference on the music items. Instead of single overall preference rating, it is better to maintain multicriteria preference ratings based on the weigh given by the user in every criterion in table 2. To construct the multicriteria ratings for item *j* of user *i*, the two values are taken into calculation.

Table 5: Interest Level Matrix (scale: 0-10)

	Item 1	Item 2	Item 3	Item 4
User 1	0	2	0	7
User 2	4	5	6	0
User 3	4	5	6	7
User 4	5	6	6	9
User 5	4	6	8	8

Table 6: Multicriteria Implicit Ratings Matrix (music, lyric, voice)

	Item 1	Item 2	Item 3	Item 4
User 1	(0,0,0)	(1.1,0.7,0.2)	(0,0,0)	(2.2,2.2,2.6)
User 2	(1.4,1.2,1.4)	(1,1.3,2.7)	(1.7,1.4,2.8)	(0,0,0)
User 3	(2,0.6,1.4)	(3,0.7,1.3)	(2.3,2.3,1.4)	(3.2,3.1,1.7)
User 4	(1.9,1,2.1)	(2.3,1.1,2.6)	(1.7,2.3,2)	(2.6,3.4,3)
User 5	(1.2,1.2,1.6)	(1.9,1.6,2.5)	(2,2.8,3.2)	(2.4,2.8,2.8)

The first one is the current interest level of user *i* on item *j*. Another one is the category-weigh of that item within every criterion. The category within the criteria of an item is identified using the item features available in table 1. Using the user’s interest level on the item in table 5 and the category weigh within each criterion in table 2, the multicriteria implicit interest rating is inferred, by multiplying the interest level by weigh assigned by user *i* on criteria *k*, as shown below and the multicriteria ratings matrix is given in table 6.

$$Multicriteria\_Interest\_rating_{ijk} = \left(\frac{criteria\_category\_value_{ik}}{\sum_{m=1}^{k} criteria\_category\_value_{im}} x Interest\_level_{ij} \right)$$

where $Multicriteria\_Interest\_rating_{ijk}$ represents the implicit multicriteria rating of user *i* on item *j* in criteria *k*. The $Interest\_level_{ij}$ is taken from table 5 and $criteria\text{-}category\_value_{im}$ is taken from table 2 after referring the item features in table 1. For example, the $multicriteria\_interest\_rating_{231}$ = [4 / (4 + 2 + 8)] x 6 = 1.7.

4. Similarity Measurement

The second step in the recommendation process is to find similar users to the active user in order to predict a user rating. There are basically two approaches in the implementation of collaborative filtering algorithms. The first one is the so-called “lazy learning” approach (also known as the *memory-based* approach) which skips the learning phase. Memory-based algorithms use the total ratings of users in the matrix while computing recommendations. The *model-based* approach, on the other hand, first builds a model out of the user-item interaction database and then uses this model to make recommendations. We consider the memory-based algorithms in this paper because it is a commonly used method and also we have focus on the multicriteria ratings. The memory-based systems are functionally classified into two sub-categories: *user-based* and *item-based* algorithms. These two categories of algorithms are investigated in this work under multicriteria implicit and explicit feedback environment.

4.1. User-based Similarity Calculation

User-based collaborative filtering algorithm produces the recommendation list for the active user according to the view of other users. Similar users are identified for the active user using statistical technique and recommendations are produced based on similar user’s preferences. Since Pearson correlation performs better than other similarity measures [Herlocker, J.L., 2006], Pearson correlation is considered to compute user-based and item-based similarities on both explicit and implicit ratings.

4.1.1. Using Multicriteria Explicit Ratings

In traditional collaborative recommender systems, from the user-item explicit ratings matrix, the similarity between the two users *x* and *y* is computed using Pearson correlation coefficient as given below:

$$sim(x,y) = \frac{\sum_{i \in Ixy}(r_{x,i} - \overline{r_x})(r_{y,i} - \overline{r_y})}{\sqrt{\sum_{i \in Ixy}(r_{x,i} - \overline{r_x})^2 \sum_{i \in Ixy}(r_{y,i} - \overline{r_y})^2}},$$

where $r_{x,i}$ is the rating given to item i by active user *x*; $\overline{r_x}$ is the mean rating given by active user *x*; $r_{y,i}$ is the rating given to item *i* by other user *y*; $\overline{r_y}$ is the mean rating given by the other user *y* and I_{xy} are the items co-rated by both users *x* and *y*. To make the correlation computation accurate, the system must isolate the co-rated cases.

In multicriteria Recommender systems, let us assume that each rating, the user *u* gives to item *i*, consists of an overall rating r_0 computed (average) from multicriteria ratings and *k* multicriteria ratings $r_1,\ldots,r_k$, where *k* is the number of criteria. From these *(k+1)* multicriteria ratings, *(k+1)* separate matrices can be formed for each criteria. *(k+1)* similarity calculations are to be performed to measure the similarity between the active user *x* and the other user *y*. $sim_0(x, y)$ denotes the similarity between the active user *x* and the other user *y* based on the overall rating; $sim_1(x, y)$ denotes the similarity between the user *x* and *y* on first criterion; $sim_2(x, y)$ denotes the similarity between user *x* and *y* based on the second criterion. Similarly, the similarity will be calculated for all the *k* criteria. The total similarity is calculated by aggregating the individual similarities, i.e., taking the average of the sum of all criterion similarities as given below:

$$sim_{avg}(x,y) = \frac{1}{(k+1)}\sum_{i=0}^{k} sim_i(x,y)$$

4.1.2. Using the Criteria-Category (Partial) Implicit Ratings from Explicit Ratings

In criteria-category user preferences matrix (table 2), the user preferences are maintained in different categories within every criterion. The matrix values in a row always represent the user's current interest on categories within every criterion. Another difference with usual user-item rating matrix is the size of the matrix. Here the matrix size is *m x (2x(p+q+r)),* where *m* is the number of users and *p,q,r* are the number of categories (3 in our study) with in each criteria respectively. To make the correlation computation accurate, we must first isolate the co-rated cases of two users G_{xy} (i.e., cases where both the users shown interest on the same categories). The similarity between the two users (active user *x* and the other user *y*) is computed using the Pearson correlation coefficient as defined below:

$$sim(x,y) = \frac{\sum_{i \in Gxy}(r_{x,i} - \overline{r_x})(r_{y,i} - \overline{r_y})}{\sqrt{\sum_{i \in Gxy}(r_{x,i} - \overline{r_x})^2 \sum_{i \in Gxy}(r_{y,i} - \overline{r_y})^2}},$$

where $r_{x,i}$ is the rating given to category *i* by active user *x*; $\overline{r_x}$ is the mean rating given by active user *x*; $r_{y,i}$ is the rating given to category *i* by other user *y*; and $\overline{r_y}$ is the mean rating given by other user *y*.

4.1.3. Using the Implicit Ratings

During the user-system interactions, the system maintains the implicit interest measures in two intermediate tables 3 and 4. Using these two table values, the user interest level matrix is derived and is given in table 5. From the interest level matrix and criteria-category preference ratings matrix (table 2), the final implicit ratings matrix is derived as shown in table 6. To find the similarity between users in this implicit ratings matrix, the same methodology is used as shown in section 4.1.1. Here, the difference is that the user-item ratings matrix contains implicit interest ratings measured using implicit interest measures. Since we have *(k+1)* implicit ratings, *(k+1)* similarities need to be calculated. The total similarity is calculated by aggregating the individual rating similarities.

4.2. Item-based Similarity Calculation

The item-based approach looks into the set of items the target user has rated and computes how similar they are to the target item *i* and then selects *k*, the most similar items based on the similarities calculated. From the most similar items found, the prediction is then computed by taking a weighed average of the target user's ratings on these similar items. The basic idea in similarity computation between two items *i* and *j* is to first isolate the users who have rated both these items and then apply a similarity computation technique to determine the similarity of *i* and *j*.

4.2.1. Using Multicriteria Explicit Ratings

Assume that the set of users who rated both the items *i* and *j* is U_{ij}. Then the Pearson correlation similarity is computed using:

$$sim(i, j) = \frac{\sum_{u \in Uij} (r_{u,i} - \bar{r}_i)(r_{u,j} - \bar{r}_j)}{\sqrt{\sum_{u \in Uij} (r_{u,i} - \bar{r}_i)^2} \sqrt{\sum_{u \in Uij} (r_{u,j} - \bar{r}_j)^2}},$$

where $r_{u,i}$ denotes the rating of user u on item i, $\bar{r}_i$ is the average rating of the i^{th} item. $r_{u,j}$ denotes the rating of user u in item j and $\bar{r}_j$ is the average rating of the j^{th} item.

In the multicriteria recommender system, each rating column contains an overall rating and k multicriteria ratings ($k+1$ rating in total). The item-wise similarity is calculated by considering this $(k+1)$ ratings separately. $sim_0(i,j)$ is the similarity between i^{th} and j^{th} item based on overall rating. $sim_1(i,j)$ denotes the similarity between the items i and j based on the first criterion. $sim_2(i,j)$ denotes the similarity between the items i and j based on the second criterion and so on. The overall similarity can be calculated by aggregating the individual rating similarities, i.e., average of all individual rating similarities. The average similarity is calculated as:

$$sim_{avg}(i, j) = \frac{1}{(k+1)} \sum_{t=0}^{k} sim_t(i, j)$$

4.2.2. Using Criteria-Category (Partial) Implicit Ratings Derived from Explicit Ratings

The items similar to the target item can be identified by considering the content features of the target item. For example, in table 1, the target item 4 has item features as Music="*MSV*", Lyric="*Vijay*" and Voice="*Doss*". Its respective categories in table 2 are M_2, L_2 and V_1 respectively. The similar categories to the target categories within each criterion are identified from table 2 by applying Pearson correlation. After identifying the similar categories, the items belong to the similar categories are considered as similar items and they can be used to predict the target user rating on the target item.

4.2.3. Using Implicit Ratings

The similarity calculation using the multicriteria implicit ratings matrix given in table 6 is similar to the item-based similarity calculation performed on explicit ratings discussed in section 4.2.1. The difference is that the ratings are derived using the implicit interest measures. The overall rating is taken from table 5.

5. Prediction Computation

The final step in the recommendation process is to perform a prediction, which will be a numerical value representing the predicted opinion of the active user u_a about the active item i_a. The prediction algorithms try to guess the rating that the user is going to provide for an item. These algorithms use the history of user ratings and content associated with items in order to provide predictions. In table 1, the overall rating and the multicriteria ratings are not independent. The overall rating has certain relationship with multicriteria ratings. It is possible to find the relationship between the overall rating and the multicriteria ratings using an aggregation function. Some users give high priority to certain criteria and it depends on the user's taste. Using the known $(k+1)$ ratings from the matrix, it is possible to predict the k multicriteria ratings of the target item through the traditional CF algorithm and an aggregation function f can be learned. From the k predicted ratings and the aggregation function, the overall rating of the target item can be measured. Based on these considerations, the prediction can be performed in three steps:

- Predict k multicriteria ratings using traditional CF method,
- Learn the aggregation function to predict the overall rating,
- Predict overall rating using the aggregation function.

5.1. Predict Multicriteria Ratings

In this step, the k multicriteria ratings are predicted using the user-based and item-based prediction techniques with multicriteria explicit ratings, with the criteria-category (partial) implicit ratings and with multicriteria implicit ratings. These prediction algorithms use user-based and item-based similarity scores obtained in the previous sections. Using k multicriteria ratings, k single user-item rating matrices are formed. The k-dimensional multicriteria rating space is decomposed into k single rating recommendation problems, where each problem can be represented with the traditional *Users x Items* ratings matrix. Using the k matrices, k predictions are calculated. The prediction method for single rating matrix is given in the following sections.

5.1.1. User-based Prediction Algorithm Using Explicit Ratings (UB_ER)

This prediction algorithm requires the user neighborhood (k-nearest) for the active user u_a. User-based prediction algorithm is based on active user's average rating and an adjustment to it.

i.e., *prediction = user_average_rating + adjustment*.

The adjustment is a weighed sum that integrates user-based similarity scores. The prediction algorithm sums up the active user's average rating by considering the whole set of items that the active user has rated and the adjustment that is a weighed sum of the other user ratings concerning the active item and their similarity with the active user. The prediction value on item i_j for the active user u_a is then computed as follows:

$$pr_{aj} = \overline{r_a} + \frac{\sum_{i=1}^{l} (r_{ij} - \overline{r_i}) * sim_{ai}}{\sum_{i=1}^{l} | sim_{ai} |}$$

where sim_{ai} represents the similarity between the active user u_a and all the users u_i, for $i = 1,2,\ldots,l$, belonging to the active user's neighborhood. From the l users in the active user's neighborhood, only those who have actually given their opinion on item i_j will be included in that sum. Since we have k single user-item matrices, we get k predictions $(r_1, r_2, \ldots, r_k)$.

5.1.2. User-based Prediction Algorithm Using Criteria-Category (Partial) Implicit Ratings (UB_PIR)

The criteria-category preferences matrix gives accurate similarity scores between the users because it analyzes the users with respect to multicriteria categories. This matrix always shows the user's current criteria-category preferences in all the criteria. Using the identified similar users, the multicriteria explicit ratings matrix is used for prediction calculation. The prediction value on item i_j for the active user u_a is then computed using the identified similar users as follows:

$$pr_{aj} = \overline{r_a} + \frac{\sum_{i=1}^{l} (r_{ij} - \overline{r_i}) * sim_{ai}}{\sum_{i=1}^{l} | sim_{ai} |},$$

where sim_{ai} is the similarity score calculated in section 4.1.2 using the criteria-category ratings matrix.

5.1.3. User-based prediction algorithm using implicit ratings (UB_IR)

Based on the user-based implicit ratings similarity calculated in section 4.1.3, the prediction is computed. The prediction value on item i_j for the active user u_a is then computed using the implicit ratings matrix (table 6) as follows:

$$pr_{aj} = \overline{r_a} + \frac{\sum_{i=1}^{l} (r_{ij} - \overline{r_i}) * sim_{ai}}{\sum_{i=1}^{l} | sim_{ai} |},$$

where sim_{ai} is the similarity score calculated in section 4.1.3 using the implicit ratings matrix.

5.1.4. Item-based Prediction Algorithm Using Explicit Ratings (IB_ER)

The item-based collaborative prediction comes up as the sum of the active item's average rating, regarding the whole set of users that have rated it and an adjustment to it. The adjustment is a weighed sum of the ratings that the active user has given to other items and their similarity with the active item. i.e.,

prediction = item_average_rating + adjustment.

The adjustment is a weighed sum that integrates the item-based similarity measures. In this section, item-based prediction algorithm based on multicriteria explicit rating, criteria-category (derived) multicriteria implicit rating and pure implicit ratings are considered. Item-based prediction algorithm is considered as the reverse of the user-based algorithms. Once we have calculated the similarities between the active item i_j and all other items in the user-item matrix, the final step is to isolate the l items that share the greatest similarity with item i_j. The prediction for item i_j is computed for the active user u_a by computing the sum of ratings given by the active user on items belonging to the neighborhood of i_j. Those ratings are weighed by the corresponding similarity sim_{jt}, between item i_j and item i_t, where $t = 1,2,\ldots,l$, taken from the neighborhood. Then the adjustment is added with the item's average rating. The item-based prediction is computed using the explicit ratings as given below:

$$pr_{aj} = \overline{r_a} + \frac{\sum_{t=1}^{l} sim_{jt} * r_{at}}{\sum_{t=1}^{l} | sim_{at} |},$$

5.1.5. Item-based prediction using criteria-category (partial) implicit ratings (IB_PIR)

From the identified similar items in section 4.2.2, the prediction is computed. The prediction for the item i_j is computed for the active user u_a by computing the sum of ratings given by the active user on items belonging to the neighborhood of i_j. Those ratings are weighed by the corresponding similarity sim_{jk}, between item i_j and item i_t, where $t=1,2,\ldots,l$, taken from the neighborhood. Then the adjustment is added with the item's average rating. Using the criteria-category implicit ratings, the prediction is computed as:

$$pr_{aj} = \overline{r_a} + \frac{\sum_{t=1}^{l} sim_{jt} * r_{at}}{\sum_{t=1}^{l} | sim_{at} |}$$

5.1.6. Item-based Prediction Using Implicit Ratings (IB_IR)

Instead of using the item-based explicit ratings similarity, the item-based implicit ratings similarity calculated in section 4.2.3 is used to compute the prediction. First, the items average rating is computed on implicit ratings and then the adjustment is added. It is similar to the item-based explicit rating prediction given in section 5.1.4 and the prediction is computed using implicit ratings as given below:

$$pr_{aj} = \overline{r_a} + \frac{\sum_{t=1}^{l} sim_{jt} * r_{at}}{\sum_{t=1}^{l} | sim_{at} |},$$

In user-based and item-based prediction algorithms, the algorithm calculates a prediction using one single rating matrix. Using the same procedure for *k* single rating problems, we get *k* prediction scores. These *k* prediction scores are the *k* criteria ratings and they are used to estimate the overall rating using aggregation function.

5.2. Learn the Aggregation Function to Predict the Overall Rating

From the predicted *k* individual multicriteria ratings, it is possible to estimate the relationship between the overall rating and the multicriteria ratings of items, such as $r_0 = f(r_1, r_2, \ldots, r_k)$. The next step is to find an aggregation function that will estimate the overall rating from multicriteria ratings. This function may be a simple average of the multicriteria ratings or it may be a function derived from a statistical technique or a machine learning technique. We have considered a linear regression analysis technique in which the aggregation function for the overall rating would be a linear combination of the multicriteria ratings, i.e., *overall rating* $r_0 = w_1 r_1 + w_2 r_2 + \ldots + w_k r_k + c$, where w_i is the weight associated with criterion i and it denotes the criterion's importance according to the user. It is possible to estimate the weights w_i ($i=1,2,\ldots,k$) and constant c based on the set of known ratings. The aggregation function can be designed for total matrix, user-based, or item-based. In this work, the function f is a total aggregation function and it is used to predict all unknown ratings. In this regression-based function, the criteria weights w_i are the same for all users and items. Depending on the domain, the user-based or item-based aggregation functions can also be useful in different applications.

5.3. Predict Overall Rating

From the predicted ratings $r_1', r_2', \ldots, r_k'$ and the learned aggregation function f, it is possible to predict the unknown overall rating. i.e., $r_0' = f(r_1', r_2', \ldots, r_k')$.

6. Experimental Methodology

6.1. DataSet

In order to evaluate the proposed approaches, we have collected a set of user submitted music ratings from our Music recommender system developed for this experiment. When the user feedback is collected, the user is asked to provide their rating of the current item in three aspects (music, lyric and voice) in a scale of 1 to 10. This system's database contains 4568 ratings provided by 102 users to 205 music items. The sparsity level in the database is defined as 102 x 205 – 4568 / 102 x 205 ≈ 0.78. The prediction algorithms are evaluated over 500 ratings set taken randomly from the set of 4568 actual ratings. Each user has rated 147 music items on average and the number of common music between the two users is 8.4 on average. Each music item has been rated by 22.3 users on average. The average number of common users between two music items is 18.4. The average rating on each criterion is 6 approximately. Since we have a small amount of data and to achieve reliable results, we have used 10-fold cross-validation technique. In this method, we have randomly divided the data set into 10 disjoint subsets. We divide each data set into training and test portion. In each subset, 8/10 (80%) of the data are used for training and 2/10 (20%) of data for testing prediction. This process is repeated 10 times with different test dataset and is evaluated in all the predicted ratings. We randomly choose different training and test sets each time, taking the average of the MAE (Mean Absolute Error) values.

6.2. Metrics

A number of metrics are available to evaluate the Recommender system performance. We usually evaluate the quality of prediction algorithms in two dimensions: *accuracy* and *coverage*. Accuracy is one of the measures that can be assessed using the statistical accuracy metrics and decision-support accuracy metrics. We have considered the statistical accuracy metric to compare the quality of predictions because it is a commonly used measure to evaluate the accuracy in many recommender systems.

- *Statistical accuracy metric*

Statistical accuracy metric evaluates the accuracy of a prediction algorithm by comparing the numerical deviation of the predicted rating from the actual user rating. One of the commonly used metric in this category is the Mean Absolute Error (MAE) [Herlocker,J.L., 2006]. The MAE calculates the irrelevance between the predicted rating and the user rating. We represent p_i as system prediction value and q_i as the user rating value. If N is the number of actual ratings in an item set, then MAE is defined as the average absolute difference between the N pairs $<p_i, q_i>$ of predicted ratings p_i and the actual ratings q_i. If MAE is small, it indicates good prediction accuracy and it allows better recommendations. The MAE can be defined as follows:

$$MAE = \frac{\sum_{i=1}^{N} | pi - qi |}{N}$$

7. **Results**

We present the experimental results of the evaluation of the implicit and explicit feedback approaches under multicriteria user-based and item-based prediction algorithms. The statistical accuracy metric represents the average absolute deviation between actual and predicted ratings.

7.1. Statistical Accuracy Metric

The performance of User-based prediction algorithm based on Explicit ratings (UB_ER), Criteria-category (partial) Implicit ratings (UB_PIR), Implicit ratings (UB_IR) and Item-based prediction algorithms based on Explicit ratings (IB_ER), Criteria-category (partial) Implicit ratings (IB_PIR), Implicit ratings (IB_IR) are evaluated under various sensitive parameters including *sparsity levels, training/test data ratio and neighborhood sizes*.

7.1.1. Effect of Sparsity in Prediction Algorithms

We have implemented two categories of prediction algorithms that include *user-based* and *item-based* algorithms using three kinds of user-item ratings matrices, namely *explicit ratings* matrix, *criteria-category (partial) implicit ratings* matrix and *complete implicit ratings* matrix. We ran these experiments on our training data and used test set to compute MAE. MAE has been computed for different prediction algorithms under different levels of sparsity. Table 7 shows the observed values of MAE on the different prediction algorithms presented, in different sparsity levels. Fig. 1 illustrates the sensitivity of the algorithms in relation to the different levels of sparsity applied. Only when the sparsity level is less, the MAE value is also lower which gives better predictions.

Table 7: Accuracy of Prediction algorithms against different Sparsity

Prediction algorithms	Sparsity levels							
	0.78	*0.81*	*0.84*	*0.87*	*0.9*	*0.93*	*0.96*	*0.99*
UB_ER	0.846	0.928	0.974	1.146	1.384	1.468	1.522	1.586
UB_PIR	1.348	1.462	1.574	1.726	1.764	1.898	2.024	2.162
UB_IR	1.524	1.568	1.642	1.694	1.722	1.726	1.846	2.084
IB_ER	1.012	1.218	1.312	1.428	1.568	1.662	1.672	1.714
IB_PIR	1.128	1.132	1.146	1.258	1.322	1.436	1.542	1.758
IB_IR	0.938	0.944	0.998	1.042	1.096	1.124	1.188	1.216

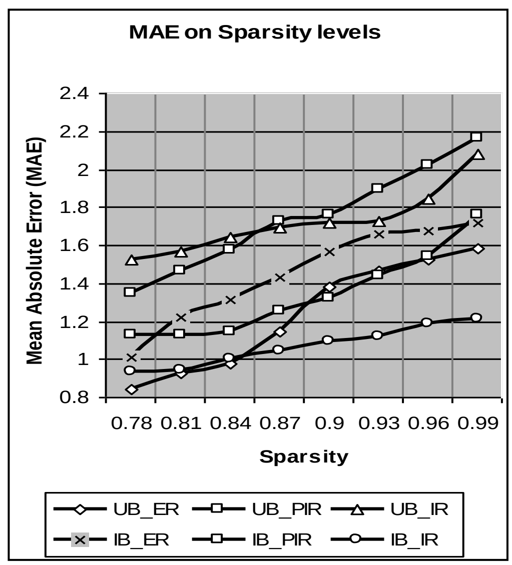

Fig. 1: Accuracy of prediction algorithms – MAE vs. Sparsity

7.1.2. Sensitivity of Training/Test Data Ratio

To determine the sensitivity of training/test data ratio, we carried out an experiment where we varied the value of x (training data ratio) from 0.2 (20%) to 0.9 (90%). Under each of these training/test data ratio, we ran the experiments. Fig. 2 shows the experimental results based on table 8 containing the observed MAE values of various prediction algorithms under different training/test data ratio. We observed that the quality of prediction increases as we increase x.

7.1.3. Experiments with Neighborhood Size

The size of the neighborhood has significant impact on the prediction quality. We performed an experiment where we varied the number of neighbors and computed MAE. The results are shown in Fig. 3 based on the observed experimental values of MAE given in table 9. The user-based and item-based prediction algorithms are evaluated with different neighborhood sizes.

Table 8: Accuracy of Prediction algorithms against different Training/Test data ratio

Prediction algorithms	*Training/test data ratio (x)*				
	0.2	*0.4*	*0.6*	*0.8*	*0.9*
UB_ER	0.852	0.842	0.824	0.786	0.732
UB_PIR	0.884	0.848	0.812	0.766	0.734
UB_IR	0.868	0.862	0.854	0.816	0.774
IB_ER	0.832	0.786	0.744	0.736	0.714
IB_PIR	0.818	0.756	0.726	0.714	0.702
IB_IR	0.824	0.802	0.752	0.722	0.712

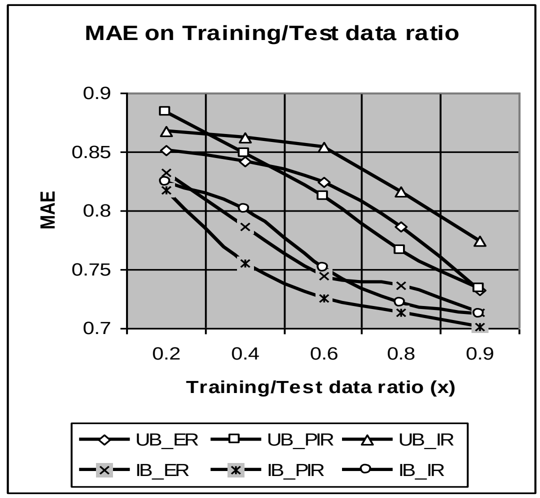

Fig. 2: Accuracy of prediction algorithms – MAE vs. Training/Test data ratio

Table 9: Accuracy of Prediction algorithms against different Neighborhood sizes

Prediction algorithms	***Number of neighbors***				
	20	***40***	***60***	***80***	***100***
UB_ER	0.744	0.728	0.732	0.726	0.724
UB_PIR	0.762	0.756	0.748	0.744	0.732
UB_IR	0.736	0.722	0.716	0.708	0.702
IB_ER	0.738	0.726	0.718	0.712	0.708
IB_PIR	0.751	0.738	0.736	0.73	0.728
IB_IR	0.758	0.744	0.74	0.722	0.714

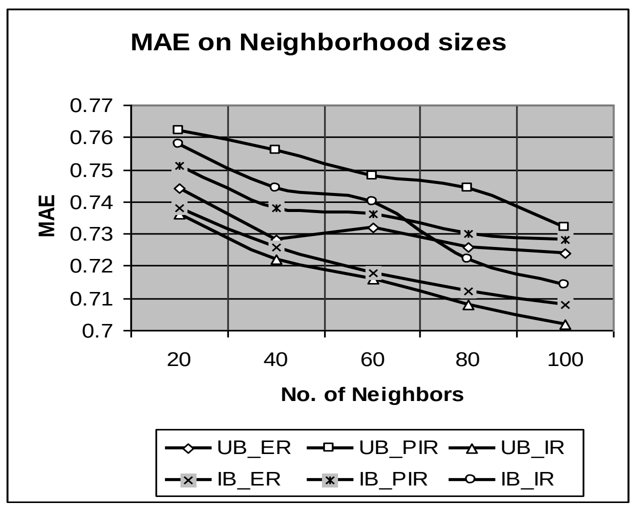

Fig. 3: Accuracy of prediction algorithms –MAE vs. Neighborhood size

8. **Discussion**

The findings of this study are mostly in accordance with expectations. As far as sparsity is concerned, within the implicit ratings based prediction algorithms, the IB_IR and IB_PIR algorithms show better performance steadily since the MAE value is lower than the other algorithms. In explicit ratings based prediction algorithms, the UB_ER algorithm shows better performance and gets low MAE value when the sparsity gets reduced. In this experiment, the performance of implicit rating based algorithms is better than the explicit rating based algorithms. Especially, implicit ratings based item-based prediction algorithm (IB_IR) seems to be very sensitive to sparsity levels. Between the user-based and the item-based prediction algorithms, the item-based algorithm shows better performance.

Within implicit ratings based algorithm, the IB_PIR and IB_IR algorithms give better prediction as far as different Training/Test ratio is concerned. In explicit ratings based algorithms, the IB_ER prediction algorithm performs better. From the experimental observation, we select x=0.8 as an optimal value for our subsequent experiments. In this Training/Test ratio experiment, within the user-based and item-based algorithms, all the three item-based prediction algorithms have superior quality than the user-based prediction algorithms. As far as the implicit ratings based prediction algorithms are concerned the IB_PIR and IB_IR algorithms seems to be sensitive to Training/Test data ratio.

Based on the results of this neighborhood size experiment, the UB_IR and IB_ER prediction algorithms have superior quality than the other prediction algorithms. Apart from these two, the UB_ER and IB_PIR algorithms also show little better performance. IB-IR prediction algorithm performs better when the neighborhood size increases. As far as the implicit ratings based prediction algorithms are concerned, UB_IR, IB_PIR and IB_IR prediction algorithms seems to be sensitive on neighborhood sizes and produces better prediction quality. In explicit ratings based algorithms, IB_ER and UB_ER algorithms perform better. Based on the observation, we have taken the optimal choice of neighborhood as 40 for the experiment. Once we obtain the optimal values of the sensitive parameters, we compare the performance of the user-based and item-based algorithms in explicit and implicit feedback environments.

From all the three categories of experiments, the results proves that the implicit ratings based prediction algorithms have the potential in predicting user preferences and produces better prediction quality than the explicit ratings based algorithms under the multicriteria ratings context. From the experimental results, it is observed that the IB_PIR algorithm performs better than other implicit ratings based algorithms.

8.1. Implications for Theory and Research

The present study contributes to our understanding of implicit relevance feedback mechanisms and multicriteria ratings utilization in recommender systems. The main contribution is an evaluation of implicit and explicit relevance feedback approach as in order to eliminate the intrusiveness problem in recommender systems. Another important contribution of this research is the multicriteria ratings inferred using implicit interest measures. Most important thing is that this study integrates the benefits of both implicit relevance feedback technique and the multicriteria ratings representation and evaluates its performance. Further, the benefits and problems in the explicit and the implicit feedback approaches are studied. The functional difference between single rating and multicriteria rating systems are explored. Another contribution of the study is the integration of both content and collaborative features in computing recommendations. This integration eliminates the new user and new item problems of recommender systems.

In terms of theory building, this study attempts to infer the user preferences from user behavior. This study proposes implicit user-item ratings model under multicriteria ratings context and infers criteria-category preferences from explicit ratings. This criteria-category preferences matrix always shows the user's current detailed preferences. Empirical results evaluate the potential behind implicit relevance feedback mechanisms and the accuracy of various prediction algorithms under multicriteria ratings context. Out of this research, the implicit ratings based algorithms perform better than explicit ratings based algorithms. In addition to the above results, the study also suggests that the item-based prediction algorithms work better than user-based prediction algorithms. Apart from these, this study also contributes to the field of user modeling and personalization, because the modeling of user preferences is a challenging task in many personalization systems.

The study explores the approaches to solve this information overload problem and attempts to provide individualized accurate recommendations to users without taking more effort from the users. The practical implication of this assignment helps the recommender system developers to focus more on factors associated with implicit relevance feedback approaches than explicit feedback approaches and to consider the multicriteria ratings for the representation of user preferences. Further, the successor in this research can explore the uncertainty factors involved in implicit relevance feedback mechanisms [Frias-Martinez. E, et al., 2005].

8.2. Implications of Practice

Recommender systems are getting wider acceptance in e-shopping business. The e-commerce website organizations prefer to develop user-friendly, personalized, effortless recommender system for their website visitors. Changing customer's visit into purchase depends on the individualized recommendations provided by the recommender systems. The e-commerce websites organizations compete with each other to develop innovative recommender systems for their customers and to promote their sales. While developing recommender systems, the developers try to produce non-intrusive, effortless systems with accuracy in recommendations. This research has demonstrated the way of adopting implicit relevance feedback and multicriteria rating methods in the Recommender systems and try to solve new user, new item problem in recommender systems.

8.3. Limitations and Suggestions for Future Research

Several limitations of this study are mentioned, which call for further research. Primarily, this paper deals with implicit interest measures which are inferred from user-system interactions. In this paper, we have taken into consideration, *time spent on hearing a music item* and *number of accesses to a music item* as suitable implicit interest measures to estimate the user preferences on the music item. Additional measures such as amount of mouse movement, amount of keyboard operation and eye tracking are also to be investigated in other domains (book-stores, education, movie-sites, etc.,) and the performance of the system should be evaluated. Furthermore, we have explored the memory-based prediction algorithms and in future, the model-based prediction algorithms need to be investigated. We have considered the accuracy and intrusiveness factors in our study. The scalability and sparsity problems in this implicit multicriteria recommendation context need to be addressed by researchers. The performance of the system should be evaluated with large sample size in real-time recommender systems in future.

In future research, the organizations can extend this work by involving different implicit interest indicators for estimating user preferences, by maintaining the user behaviors (indicators) on tables and by gathering data from hybrid data sources (such as content, collaborative ratings, utility and user's demographic information). And also this work can be further extended by incorporating Soft-computing techniques such as cultural algorithms and swarm intelligence algorithms in order to manage the uncertainty involved in user behavior and to obtain more prediction accuracy.

9. Conclusion

This study proposes an implicit-multicriteria combined approach for recommender systems to obtain the accurate and non-intrusive recommendations. Using this proposed approach, the performance of the implicit and the explicit feedback approaches are evaluated under user-based and item-based prediction algorithms against different sparsity levels, training/test data ratio and neighborhood sizes. The experimental results show that the implicit ratings based prediction algorithms perform better when compared with explicit ratings based prediction algorithms in all the three sensitive parameters. Out of this experiment, it is also observed that IB_PIR prediction algorithm computes better predictions, based on the optimum sensitive parameter values, than other prediction algorithms. The research findings on a real-world data set confirm that the multicriteria ratings and the implicit ratings have the great potential and they can be successfully used to build accurate and non-intrusive recommender systems. This study aims to motivate future research on further understanding of non-intrusive, effortless next generation recommender systems.

REFERENCES

Adomavicius, G. and T. Alexander, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 17, No. 6: 734–749, 2005.

Adomavicius, G. and Y.O. Kwon, "New Recommendation Techniques for Multicriteria Rating Systems", *IEEE Intelligent Systems*, Vol. 22, No. 3: 48-55, 2007.

Amazon, http://www.amazon.com, 2003.

Burke, R., "Knowledge-based Recommender systems", *Encyclopedia of Library and Information systems*, Vol.69, No.32, 2002.

CDNow, http://www.cdnow.com, 2001.

Chatterjee, P., D.L. Hoffman, and T.P. Novak, "Modeling the Clickstream: Implications for Web-Based Advertising Efforts", *Marketing Science*, Vol. 22, No. 4: 520-541, 2003.

Chen, L., and K. Sycara, "WebMate: A Personal Agent for Browsing and Searching", *The Proceedings of the 2nd International Conference on Autonomous Agents and Multi Agent Systems, AGENTS '98, ACM*, pp.132-139, 1998.

Claypool, M., D. Brown, P. Le, and M. Waseda, "Inferring User Interest", *IEEE Internet Computing*, Vol.5, No.6: 32-39, 2001.

Cooley, R., P.N.Tan, and J. Srivastava, "WebSift: The Web Site Information Filter System", *In Proceedings of the 1999 KDD Workshop on Web Mining*, San Diego, CA: Springer-Verlag, 1999.

Fox, S., K. Kamawat, M. Mydland, S. Dumais, and T. White, "Evaluating implicit measures to improve the search experiences", *ACM Transactions on Information Systems*, Vol. 23, No. 2: 147–168, 2005.

Frias-Martinez, E., G.D. Magoulas, S.Y. Chen, and R.D. Macredie, "Modeling Human Behavior in User-Adaptive Systems: Recent Advances Using Soft Computing Technique", *Expert Systems with Applications*, Vol.29, No.5: 320-329, 2005.

Godoy, D., and A. Amandi, "User profiling in personal information agents: a survey", *The Knowledge Engineering Review*, Vol. 20, No.4:329-361, 2005.

Goldberg, D., D. Nichols, B. M. Oki, and D. Terry, "Using collaborative filtering to weave an information tapestry", *Communications of the ACM*, Vol.35, No.12: 61–70, 1992.

Guttman, R.H., "Merchant differentiation through integrative negotiation in agent-mediated electronic commerce", *Master's Thesis*, School of Architecture and Planning, Program in Media Arts and Sciences, Massachusetts Institute of Technology, 1998.

Herlocker, J.L., J.A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering", *Proceedings of the 22nd ACM SIGIR Conference on Research and Development in Information Retrieval*, 1999.

Hingston, M., "User friendly recommender systems", *Honours thesis*, University of Sydney, 2006.

Huberman, B., and M. Kaminsky., "Beehive: A System for Cooperative Filtering and Sharing of Information", *Technical Report, Dynamics of Computation Group*, Palo Alto, CA: Xerox, Palo Alto Research Center, 1996.

Joachims, T., L. Granka, B. Pan, H. Hembrooke, F. Radlinski, and G. Gay, "Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search", *ACM Transactions on Information Systems*, Vol. 25, No. 2:7, 2007.

Jung, K., "Modeling web user interest with implicit indicators", *Master Thesis*, Florida Institute of Technology, 2001.

Jung, S., J.L. Herlocker, and J. Webster, "Click data as implicit relevance feedback in web search", *Information Processing and Management*, Vol. 43, No.3: 791–807, 2007.

Kellar, M., C. Watters, J. Duffy, and M. Shepherd, "Effect of task on time spent reading as an implicit measure of interest", *ASIST 2004*, pp. 168–175, 2004.

Kelly, D., and J. Teevan, "Implicit feedback for inferring user preference: A bibliography", *SIGIR Forum*, Vol. 37, No. 2: 18-28, 2003.

Kobsa, A., J. Koenemann, and W. Pohl, "Personalized Hypermedia Presentation Techniques for Improving Online Customer Relationships", *The Knowledge Engineering Review*, Vol.16, pp.111–155, 2001.

Konstan, J., "Introduction to Recommender Systems: Algorithms and Evaluation", *ACM Transactions on Information Systems*, Vol. 22, No.1: 1-4, 2004.

Konstan, J., B. Miller, D. Maltz, J. Herlocker, L. Gordon, and J. Riedl, "Grouplens: Applying Collaborative Filtering to Usenet News", *Communications of the ACM*, Vol.40, pp.77–87, 1997.

Krulwich, B., "LifeStyle Finder: Intelligent User Profiling Using Large-Scale Demographic Data", *AI Magazine*, Vol.18, No.2:37–45, 1997.

Lee, H.H., and W.G. Teng, "Incorporating Multi-Criteria Ratings in Recommendation Systems", *Information Reuse and Integration*, IRI 2007- IEEE International Conference, 273–278, 2007.

Lee, T.Q., Y. Park, and Y.T. Park, "A time-based approach to effective recommender systems using implicit feedback", *Expert Systems with Applications*, Vol. 34, No. 4: 3055-3062, 2008.

Lieberman, H., "Letizia: An Agent that Assists Web Browsing", *In Proceedings of the IJCAI'95*, 924–929, 1995.

Linden, G., B. Smith, and J. York. "Amazon.com Recommendations: Item-to-Item Collaborative filtering", *IEEE Internet Computing*, Vol.7, No.1: 76-80, 2003.

Martinez, L., M.J. Barranco, L.G. Perez, and M. Espinilla, "A Knowledge Based Recommender System with Multigranular Linguistic Information", *International Journal Of Computational Intelligence Systems*, Vol. 1, No. 3: 225 – 236, 2009.

Mladenic, D., "Personal WebWatcher: Implementation and Design", *Technical Report IJS-DP-7472*, Department of Intelligent Systems. Slovenia: J. Stefan Institute, 1996.

Mobasher, B., and S.S. Anand (Eds.), "Intelligent Techniques for Web Personalization", *ITWP 2003, LNAI 3169*, pp. 1–36, 2005.

Mobasher, B., R. Cooley, and J. Srivastava, "Automatic Personalization Based on Web Usage Mining", *Communications of the ACM*, Vol. 43, No.8: 142–151, 2000.

Montaner, M., B. Lopez, and J.L.D.L. Rosa, "A Taxonomy of Recommender Agents on the Internet", *Artificial Intelligence Review*, Vol. 19, No. 4: 285–330, 2003.

Morita, M., and Y. Shinoda, "Information filtering based on user behavior analysis and best match text retrieval", *Proc. of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, 272–281, 1994.

Nageswara Rao, K., and V.G. Talwar, "Application Domain and Functional Classification of Recommender Systems—A Survey", *Journal of Library & Information Technology*, Vol. 28, No. 3: 17-35, 2008.

Nah, F.F., and S. Davis, “HCI research issues in E-commerce”, *Journal of Electronic Commerce Research*, Vol. 3, No. 3: 98-113, 2002.

Niinivaara, O., “Agent-Based Recommender Systems”, *Technical Report*, University of Helsinki, Dept. of. CS, 2004.

Oard, D.W., and J. Kim, “Implicit Feedback for Recommender Systems”, *Proc. 5th DELOS Workshop on Filtering and Collaborative Filtering*, pp. 31–36, 1998.

Papagelis, M., and D. Plexousakis, “Qualitative analysis of user-based and item-based prediction algorithms for recommendation agents”, *Engineering Applications of Artificial Intelligence*, Vol.18, No.7: 781–789, 2005.

Pazzani, M.J., J. Muramatsu, and D. Billsus, “Syskill webert: Identifying interesting web sites”, *In AAAI/IAAI*, No.1 :54–61, 1996.

Resnick, P., N. Iacovou, M. Suchak, P. Bergstorm, and J. Riedl, “GroupLens: An Open Architecture for Collaborative Filtering of Netnews”, *Proceedings of ACM Conference on Computer Supported Cooperative Work*, citeseer.ist.psu.edu/resnick94grouplens.html, 1994.

Sakagami, H., T. Kamba, and Y. Koseki, “Learning Personal Preferences on Online Newspaper articles for User Behaviors*”, In Proc. 6th Int. World Wide Web Conference*, pp.291–300, 1997.

Sarwar, B., G. Karypis, J. Konstan, and J. Riedl, “Analysis of Recommendation algorithms for E-Commerce”, *EC’00*, October 17-20, Minneapolis, Minnesota, 2000.

Sarwar, B., J. Konstan, A. Borchers, J. Herlocker, B. Miller, and J. Riedl, “Using Filtering Agents to Improve Prediction Quality in the GroupLens Research Collaborative Filtering System”, *In Proceeding of the ACM Conference on Computer Supported Cooperative Work (CSCW)*, 1998.

Shahabi, C., and Y. Chen, “Web Information Personalization: Challenges and Approaches”, *The 3nd International Workshop on Databases in Networked Information Systems (DNIS 2003)*, Aizu-Wakamatsu, Japan, 2003.

Shih, T.K., C. Chiu, H.-H. Hsu, and F. Lin, “An integrated framework for recommendation systems in e-commerce”, *Industrial Management & Data Systems*, Vol.102, No.8: 417-431, 2002.

Shin, N., “Strategies for Competitive Advantage in Electronic Commerce”**,** *Journal of Electronic Commerce Research*, Vol. 2, No. 4: 164–171, 2001.

Timothy K. F. Fung, “Banking with a personalized touch: Examining the impact of website customization on commitment”, *Journal of Electronic Commerce Research*, Vol. 9, No. 4: 296-309, 2008.

Zenebe, A., and A.F. Norcio, “Representation, similarity measures and aggregation methods using fuzzy sets for content-based recommender systems”, *Fuzzy Sets and Systems*, Vol.160, No.1: 76 – 94, 2009.