Items Based Fuzzy C-Mean Clustering for Collaborative Filtering

Kiatichai Treerattanapitak* and Chuleerat Jaruskulchai*

Abstract

Collaborative Filtering is a method behind the successful of Recommendation System that is widely used especially in E-Commerce website. It boosts profit for E-Commerce website by trying to predict user interested from peer's opinions and offering proper products that match user's interested. A challenge for collaborative filtering is data characteristics i.e. amount of data always large and contains a lot of missing values. Hence it is not easy to perform high recommendation accuracy. In addition, the computational cost of Collaborative Filtering is always highly expensive from scalability issue and hard to predict from cold start and sparsity. There are several approaches but Item based approach is efficient and easy algorithm that reduces effect of scalability by performing calculation on item side instead of user side. In this paper we propose an approach to improve Item based method by applying Fuzzy C-Mean algorithm over Item based to partition items into several clusters and perform prediction against clusters. Primary advantage is greatly reduces computation cost on MovieLens dataset. Our approach shows that it overcomes scalability, cold start and sparsity. It saves more than 99% computational time and does not change the prediction quality eventually real-time prediction and website responsive can be processed much faster.

Keyword: Collaborative Filtering, Recommendation System Fuzzy C-Mean

1. Introduction

In the E-Commerce world where everything goes online and relies on the Internet, the buyers and the sellers never meet each other but trading deal can be completed online from the Internet. Sellers build their websites to hold their product's details. Buyers use internet navigator to visit the website and they will accept the deal when those products fulfill their desires. In order to increase trading success rate, the key element is website that is not only display attractive information but also pay attention to the buyer's need and make proper recommendation for them to select the right things otherwise they will go elsewhere.

The next question is raised, how to make those website smart enough and properly do that? In human scenario, when a seller want to sell or recommend a product to a customer, he or she must know something about the customer's background even they just met at the first place. Human can predict and estimate from personal stereotypes at the first glance from past experience. In order to make a website thinks like a real human in similar manner, they must learn from feeding collection of data to gain experiences like human learning through training process. Hence they can estimate an unseen data from past experiences like human.

The process behind this estimation is the calculation of similar data that system will be used to judge the different between things. If collection of data that feed to train the system is collection of products and their characteristics, the system will attempt to identify similar products from their characteristics. This technique is called Content based filtering. On the other hand if user preferences are used, the system will attempt to predict and recommend from similar of past seen users. This technique is called Collaborative Filtering (CF) and widely use in modern E-Commerce website.

The system that attempts to offer products from filtering method is called Recommendation System. We will take a deeper look about it in the next section.

Recommendation System

The main goal of Recommendation System is an attempt to offer products. It much relies on similarity calculation. In [1], Recommendation System can be classified into 3 different categories based on technique used; they are content-based method, collaborative method, and hybrid method.

Content-Based Method

Content-Based Method recommends items to a user from items preferred by that user in the past. It is done by first building relation between item and its properties in term of Matrix then select the most similar items to the target item by compute similarity from various mathematic functions. The most common similarities that used are Adjusted Cosine, Cosine or Pearson coefficient. Good similarity measures will result a high prediction quality.

Collaborative method

This technique is the most successful method in Recommendation System. Collaborative method attempts to recommend items for target user from users with similar preferences. The most common process of CF performs similarity computation on collection of user preferences that E-Commerce websites usually collect from rating made against products. The calculation used is the same technique in Content-Based Recommendation but focus on peer opinions.

Hybrid method

This method initiate from drawbacks of above two techniques. In case of a customer buys product A, it is not possible to recommend a completely different product B to that customer by Content-Based method. In contrast with Collaborative method which is designed to recommend product B based on previous customers with similar preferences hence it is not possible to recommend product C which is the

* Department of Computer Science, Kasetsart University, Bangkok

same category as product A by considering only previous customer preferences. By this scenario, drawbacks from both methods won't take into account when combine both to get advantage from each other. That is the idea of Hybrid Approach. Even it sounds efficient but in some situation where the details of items are not presented, it will not be able to recommend by Content-based method and Hybrid method will rely only on Collaborative method alone. However the reader may refer to [1] for more details and this method is out of scope of this paper.

Collaborative Filtering

The common goal of CF is an attempt to offer product i for user u by performing prediction Pu,i from the neighborhood. There is a relation between collection of users and collection of products that represents the user interested in term of Matrix.

$$\begin{array}{cc} & Item_i \\ User_u & \begin{bmatrix} W_{11} & W_{12} & W_{13} & \cdots\cdots & W_{1i} \\ W_{21} & W_{22} & W_{23} & \cdots\cdots & W_{2i} \\ W_{31} & W_{32} & W_{33} & \cdots\cdots & W_{3i} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ W_{u1} & W_{u2} & W_{u3} & \cdots\cdots & W_{ui} \end{bmatrix} \end{array}$$

Figure 1*: User-Item relation matrix*

From Figure 1, wui is rating score given by user u on item i. Similarity calculation will perform between user vectors along with the row of this matrix. High similarity value means those two users are highly related i.e. they have highly similar preferences. This approach is call Memory based method. Apart from that, there are several approaches that are developed to find collection of closet match user preferences, e.g. [3],[4] for Bayesian approach [5],[10] mimic natural by heuristic model like GA and Wasp colony. These approaches are call Model based method.

The most common problems in the real word where CF is implemented, is data behavior and it is limitations of the CF computational. The first is sparse data that naturally occur because of user behavior won't make much rating to the products. Consequently, it is a lack of training data that CF can be used to predict. The second is scalability, because of E-Commerce scale like Amazon has more than many millions of products and users. Hence the matrix size is very large; the calculation over the matrix will be very expensive and traditional CF will suffer to compute similarity at this scale. The third is cold start that occurs when new item or new user is added to the system. It is not able to identify similarity user or item that closely match to the new user or new item.

In order to overcome scalability issue, [6], [9] introduce "Item based" that is a method to recommend similar products from previously purchase by the same customer but focusing on Item aspect rather than users and later on [8] develop item-to-item approach and implement in Amazon.com. We will discuss more detail about Item based in the next section.

In addition to scalability, an advantage of clustering technique is used to lower data size. [7] takes this advantage by applying K-Mean clustering and address one drawback of K-Mean that data with similar characteristics could be classified into different clusters. Hence, new similarity is proposed by including those characteristics to improve classification quality. [12] applies K-Mean to group the user with similar preferences and use smoothing method to fill miss rating. [11] applies Fuzzy C Mean on Netflix dataset by adjust objective function to directly minimize RMSE (Root mean square error) which is used to measure in Netflix competition.[2]

The rest of the paper is organized as follows. The next section provides a brief background in item based collaborative filtering algorithms. We first formally describe present a challenge associated with item-based collaborative filtering. In section 3, we present the Item-based Fuzzy C-Mean approach and describe the algorithm in detail. The final section provides some concluding remarks and directions for future research.

2. Item-Based Approach

The basic idea of Item-Based is transpose matrix in Figure 1 and find similarity between items. Similarity computation is performed between two items i and j by first isolate the users who have rated both of these items and then to apply a similarity computation technique to determine the similarity.

The Item-Based prediction ($P_{u,j}$) is calculated from most *N* similar items by this equation:

$$P_{u,i} = \frac{\sum\limits_{all\ items\ N\ similar\ to\ item\ i} (S_{i,N} \cdot R_{u,N})}{\sum\limits_{all\ items\ N\ similar\ to\ item\ i} S_{i,N}}$$

where ($S_{i,N}$) is similarity measure of all similar items *N* and item i and $R_{u,N}$ is given rating score by user u to all items *N*.

There are numerous ways of similarity calculation technique as describe above. [9] claimed that Adjusted Cosine given lowest error compare to Pearson Correlation when measure by MAE (Mean Absolute Error). Formally, Adjusted Cosine will be computed by

$$sim(i,j) = \frac{\sum\limits_{u\in U}(R_{u,i} - \overline{R_u})(R_{u,j} - \overline{R_u})}{\sqrt{\sum\limits_{u\in U}(R_{u,i} - \overline{R_u})^2}\sqrt{\sum\limits_{u\in U}(R_{u,j} - \overline{R_u})^2}}$$

And Pearson Correlation will be computed by

$$sim(i,j) = \frac{\sum\limits_{u\in U}(R_{u,i} - \overline{R_i})(R_{u,j} - \overline{R_j})}{\sqrt{\sum\limits_{u\in U}(R_{u,i} - \overline{R_i})^2}\sqrt{\sum\limits_{u\in U}(R_{u,j} - \overline{R_j})^2}}$$

where $R_{u,i}$ and $R_{u,j}$ is given rating score by user u to item i and j respectively, is average rating of user u.

Occasionally, two different users may rate score in different scale e.g. (1,2,3) and (2,3,4), these two vectors are similar but in different scale. [7] address this issue and implement deviation on their modified K-Mean algorithm. By the way, this effect will not be taken into account if applying adjusted cosine as explained in [9].

Because of Item-Based prediction will select only similar items by matching user-pair in the vectors. In case of an item is rate by only one user in data collection, Item-Based will not able to find similar item. In the next section we will discuss about applying Fuzzy C-Mean over Item-Based to overcome this situation.

3. Item-Based Fuzzy C-Mean

Primary advantage of Item-based is an ability to perform even in the large scale dataset. By applying clustering technique over Item-based it will allow to partition data even less and would result in reduce in computational cost. By performing similarity calculation against cluster, time complexity would reduce from O(N) to O(k) where k is number of clusters.

By applying K-Mean, we would face to the same issue addressed in [7]. Moreover, similarity between one data and two clusters can be slightly different. For example if a data has similarity value at 20 to cluster A and 19 to cluster B. Will this data be member of only A by ignoring B? By including B, it should have impact to prediction quality and would make more sense. Unfortunately, K-Mean does not allow multiple clusters but Fuzzy C-Mean does.

One interesting question is how we can compute prediction after performing Fuzzy C-Mean clustering. In order to answer this question, let take a closer look in Fuzzy C-Mean algorithm.

In Fuzzy C-Mean, the first step is initialize centroid (the representative vector of the cluster) then compute membership function of each item vector x and each cluster from

$$\mu_{c_i}(x) = \frac{1}{\sum_{j=1}^{k}\left(\frac{\|x-v_i\|^2}{\|x-v_j\|^2}\right)^{\frac{1}{m-1}}};1 \le i \le k, x \in X$$

where $\mu_{c,i}$ is the membership function of item vector x, k is total number of clusters, v is the centroid vector, m isfuzzy coefficient. Then compute the new centroid from

$$v_i = \frac{\sum_{x\in X}(\mu_{c_i}(x)^m x)}{\sum_{x\in X}\mu_{c_i}(x)^m};1 \le i \le k$$

The iteration repeated until this condition is met

$$Max_{C_i}\left\{\mu_{C_i}^{(k-1)}(x) - \mu_{C_i}^{(k)}(x)\right\} < \varepsilon$$

$\mu_{c,i}$ (x) is a kind of similarity that we will use by replacing $S_{i,N}$ in Item-based. By combining Fuzzy C-Mean to Item-based. The prediction equation from Item-based will become

$$P_{u,i} = \frac{\sum_{all\ clusterC_i\ similar\ to\ item\ i}(\mu_{C_i}(x)\cdot v_{u,i}(x))}{\sum_{all\ cluster\ C_i\ similar\ to\ item\ i}\mu_{C_i}(x)}$$

But from Fuzzy C-Mean

$$\sum_{all\ items\ N\ similar\ to\ item\ i}\mu_{C_i}(x) = 1$$

Then prediction equation will become

$$P_{u,i} = \sum_{all\ cluster\ C_i\ similar\ to\ item\ i}(\mu_{C_i}(x)\cdot v_{u,i}(x))$$

And we will compute $\mu_{c,i}$ (x) by replacing distance term with similarity between item i and Cluster C_i but similarity calculation from either Cosine or Correlation based will be in range [-1,1]. Consequently, $\mu_{c,i}$ (x) will lost its property by

$$\sum_{all\ items\ N\ similar\ to\ item\ i}\mu_{C_i}(x) \ne 1$$

In order to preserve this property, we will modify membership function and calculate from

$$\mu_{c_i}(x) = \sum_{j=c_1}^{c_k}\left(\frac{sim(i,c_i)}{sim(i,j)}\right)^{\frac{1}{m-1}};x \in X$$

For centroid computation, we will use the same equation by imputation missing values of centroid by average user rating. By doing this way, centroids contain all rating from every user and will not suffering from cold start or sparsity issue.

Alternatively to objective function, in the next section we use MAE (Mean Absolute Error) to measure accuracy then we will use MAE as objective function for our Fuzzy model.

4. Experiment

The experiment use testing data from MovieLens website that were collected by the GroupLens Research Project at the University of Minnesota. It consists of 100,000 ratings made by 943 users on 1,682 movies. The data was separated by 80% for training and 20% for testing. We evaluate accuracy by measure MAE for each pair between prediction and actual rating. MAE can be computed by

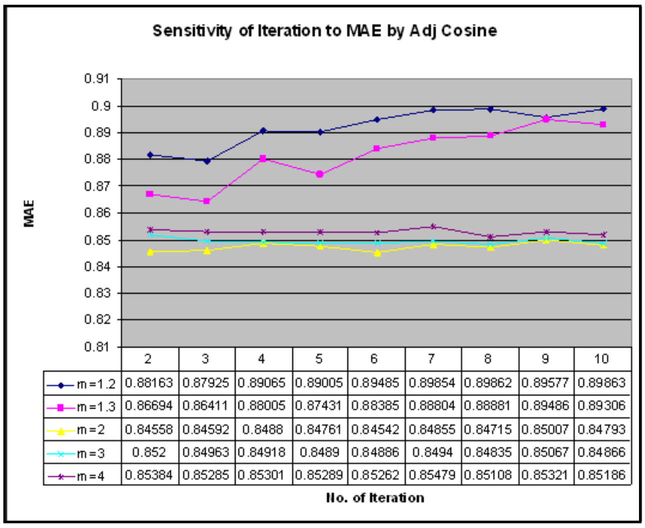
$$MAE = \frac{\sum_{i=1}^{n}|p_i - q_i|}{n}$$

where p_i is prediction value, q_i is actual rating made by user and n is total number of prediction value.

All experiments were implemented using Java on computer with Intel Core 2 Duo T7250 having 2.00 GHz with 2GB of RAM.

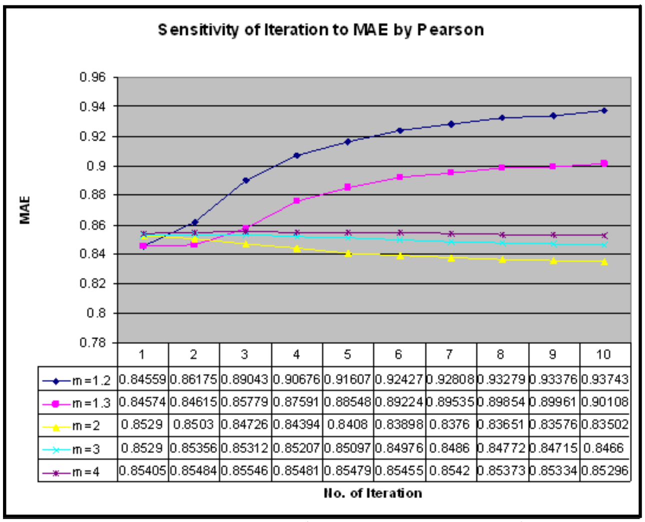
4.1 Sensitivity of m and Similarity

From our experiment an appropriate value of m is in range of 1.1 to 4 as shown in Figures 2 and 3.



	2	3	4	5	6	7	8	9	10
m=1.2	0.88163	0.87925	0.89065	0.89005	0.89485	0.89854	0.89862	0.89577	0.89863
m=1.3	0.86694	0.86411	0.88005	0.87431	0.88385	0.88804	0.88881	0.89486	0.89306
m=2	0.84558	0.84592	0.8488	0.84761	0.84542	0.84855	0.84715	0.85007	0.84793
m=3	0.852	0.84963	0.84918	0.8489	0.84886	0.8494	0.84835	0.85067	0.84866
m=4	0.85384	0.85285	0.85301	0.85289	0.85262	0.85479	0.85108	0.85321	0.85186

Figure 2 *: Impact of Iterations to MAE from Adjusted Cosine*



	1	2	3	4	5	6	7	8	9	10
m=1.2	0.84559	0.86175	0.89043	0.90676	0.91607	0.92427	0.92808	0.93279	0.93376	0.93743
m=1.3	0.84574	0.84615	0.85779	0.87591	0.88548	0.89224	0.89535	0.89854	0.89961	0.90108
m=2	0.8529	0.8503	0.84726	0.84394	0.8408	0.83898	0.8376	0.83651	0.83576	0.83502
m=3	0.8529	0.85356	0.85312	0.85207	0.85097	0.84976	0.8486	0.84772	0.84715	0.8466
m=4	0.85405	0.85484	0.85546	0.85481	0.85479	0.85455	0.8542	0.85373	0.85334	0.85296

Figure 3 *: Impact of Iterations to MAE from*

Pearson Correlation

There are two behaviors, if $m < 2$, result gets worse when iteration increased while $m \geq 2$, result gets better. This behavior is an effect of the power term $\frac{1}{m-1}$ in membership function. However we choose result by selecting the best MAE output from each iteration of varying of m as shown in Figure 4. In conclusion, we will use $m = 2$ for both Adjusted Cosine and Pearson Correlation.

4.2 Number of clusters

Because of different similarity between Pearson Correlation and Adjusted Cosine, they produce MAE in different behavior as shown in Figures 5 and 6.

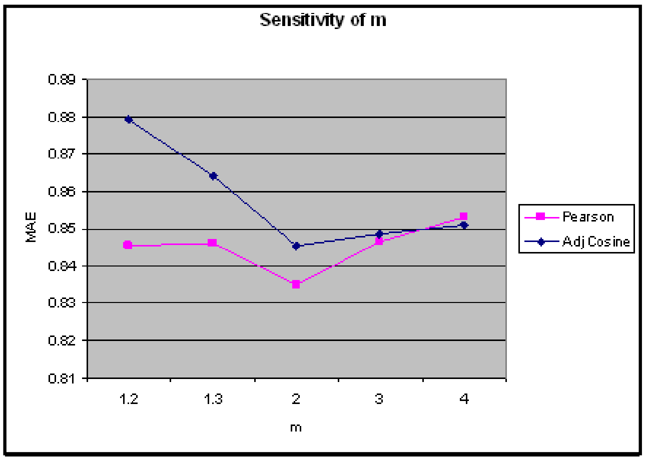
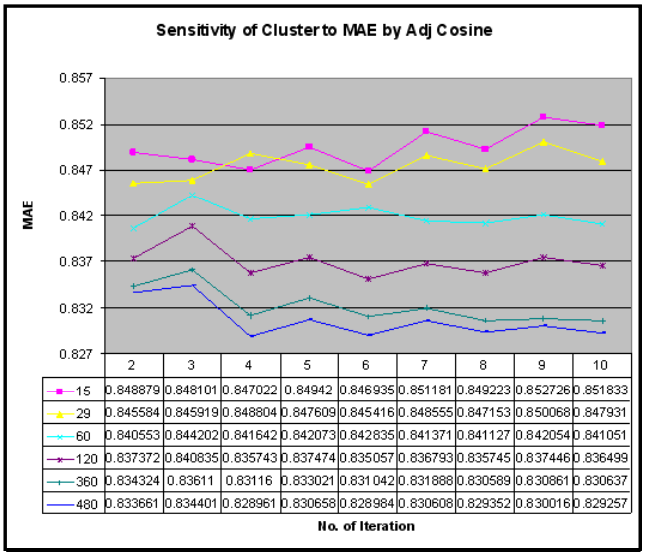
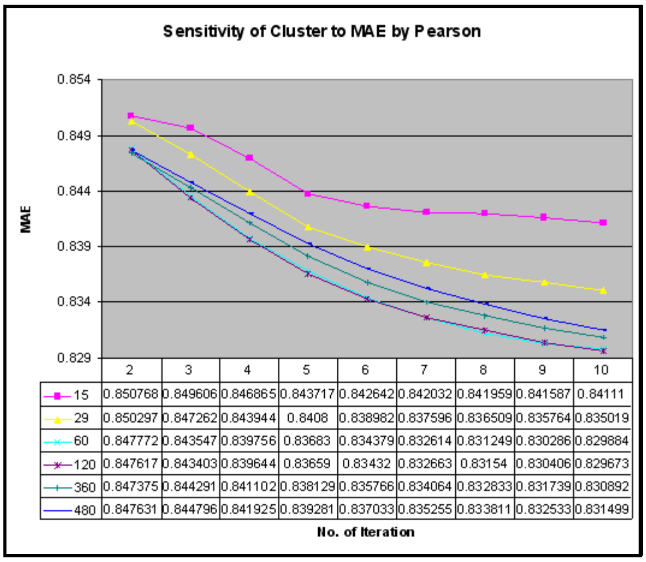


Figure 4 *: Impact of m to MAE*



	2	3	4	5	6	7	8	9	10
15	0.848879	0.848101	0.847022	0.84942	0.846935	0.851181	0.849223	0.852726	0.851833
29	0.845584	0.845919	0.848804	0.847609	0.845416	0.848555	0.847153	0.850068	0.847931
60	0.840553	0.844202	0.841642	0.842073	0.842835	0.841371	0.841127	0.842054	0.841051
120	0.837372	0.840835	0.835743	0.837474	0.835057	0.836793	0.835745	0.837446	0.836499
360	0.834324	0.83611	0.83116	0.833021	0.831042	0.831888	0.830589	0.830861	0.830637
480	0.833661	0.834401	0.828961	0.830658	0.828984	0.830608	0.829352	0.830016	0.829257

Figure 5 *: Impact of Clusters to MAE by Adjusted Cosine*



	2	3	4	5	6	7	8	9	10
15	0.850768	0.849606	0.846865	0.843717	0.842642	0.842032	0.841959	0.841587	0.84111
29	0.850297	0.847262	0.843944	0.8408	0.838982	0.837596	0.836509	0.835764	0.835019
60	0.847772	0.843547	0.839756	0.83683	0.834379	0.832614	0.831249	0.830286	0.829884
120	0.847617	0.843403	0.839644	0.83659	0.83432	0.832663	0.83154	0.830406	0.829673
360	0.847375	0.844291	0.841102	0.838129	0.835766	0.834064	0.832833	0.831739	0.830892
480	0.847631	0.844796	0.841925	0.839281	0.837033	0.835255	0.833811	0.832533	0.831499

Figure 6 *: Impact of Clusters to MAE by Pearson Correlation*

For Adjusted Cosine, the more clusters is the more accuracy but it requires a lot of training time. We trained 15 clusters by 7 minutes while 480 clusters require more than 6 hours and MAE improves by 0.022 which is small improvement compare to time used. For Pearson Correlation, we do not get any improvement by assigning the number of clusters over 60. However, MAE resulted from 15 clusters is lower from 60 by 0.012 lower while training time for 60 clusters take 27 minutes. It does not worth to go over 15 clusters. In conclusion, we will use 15 clusters for the benchmarking with Original Item-based.

4.3 Benchmarking Result to Original Item-based

We compare the best result from 15 clusters with original item-based as shown in Figure 7.

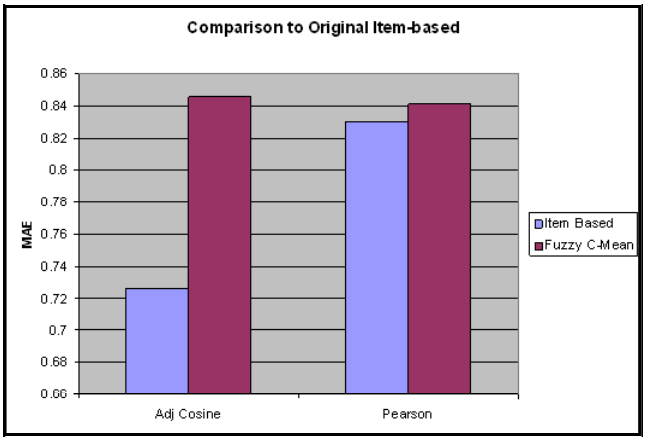


***Figure** 7 : Result compare to Original Item-based*

For Pearson Correlation, the result gets from Fuzzy model almost identical while Adjusted Cosine is greater than Original Item-based around 0.12. In term of prediction computational, in stead of computation from all 1,682 movies in the data set we can compute it from 15 representative clusters which are lower by 99.10%. Therefore, we get MAE 0.948 when using Adjusted Cosine and MAE 0.952 when using Pearson Correlation from 141 movies that rate by only one user.

5. Conclusion

In the digital age, everything is online over the Internet with many millions of information. Even if Collaborative Filtering is a promising approach to deliver those information but users always have high expectation on the system to response in real-time especially in E-Commerce website. Our approach shows that it overcomes scalability, cold start and sparsity. It saves more than 99% computational time and does not change the prediction quality eventually real-time prediction and website responsive can be processed much faster. In the future, we will improve this algorithm by looking into fake voting in order to improve quality for the real life situation.

6. References

[1] Adomavicius, G. and Tuzhilin, A. "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, pp. 734-749, 2005.

[2] Bennett, J. and Lanning, S. "The Netflix Prize". *Proceedings of KDD Cup and Workshop 2007.,* 2007.

[3] Breese, J.S., Heckerman, D. and Kadie, C. "Empirical Analysis of Predictive Algorithms for Collaborative Filtering". *Proc of 14 th Conf of Uncertainty in Artificial Intelligence,* pp. 43-52, 1998.

[4] Chen, Y.H. and George, E.I. "A Bayesian Model for Collaborative Filtering". *Proceedings of the 7 th International Workshop on Artificial,* 1993.

[5] Dehuri, S., Cho, S.B. and Ghosh, A. "Wasp: A Multiagent System for Multiple Recommendations Problem". *4th International Conference on Next Generation Web Services Practices,* pp.160-166, 2008.

[6] Karypis, G. "Evaluation of item-based top-n recommendation algorithms". *In Proceedings of the ACM Conference on Information and Knowledge Management. ACM,* New York. 2001.

[7] Kim, T.H., Park, S.I. and Yang, S.B. "Improving Prediction Quality in Collaborative Filtering based on Clustering". *IEEE/WIC/ACM Int. Conf.* pp. 704-710, 2008.

[8] Linden, G., Smith, B. and York, J. "Amazon.com Recommendations: Item-to-Item Collaborative Filtering". *IEEE Internet Computing, Vol. 7, Issues 1,* pp. 76-80, 2003.

[9] Sarwar, B., Karypis, G., Konstan, J. and Riedl, J. "Item-Based Collaborative Filtering Recommendation Algorithms". *10th Int'l World Wide Web Conference, ACM Press,* pp. 285-295, 2001.

[10] Sheth, B.D. "A Learning Approach to Personalized Information Filtering". *Massachusetts Institute of Technology,* 1994.

[11] Wu, J. and Li, T. "A Modified Fuzzy C-Means Algorithm For Collaborative Filtering". *2nd Netflix-KDD Workshop,* 2008.

[12] Xue, G.R., Lin, C., Yang, Q., Xi, W., Zeng, H.J., Yu, Y. and Chen, Z. "Scalable Collaborative Filtering Using Cluster-based Smoothing". *ACM SIGIR 2005,* pp. 114-121, 2005.