Collaborative Preference Elicitation Based on Dynamic Peer Recommendations

Sourav Saha
Fellow Program and Research
Indian Institute of Management Calcutta
Joka, Kolkata, India
souravs08@email.iimcal.ac.in

Ambuj Mahanti
Department of Management Information Systems
Indian Institute of Management Calcutta
Joka, Kolkata, India
am@iimcal.ac.in

***Abstract* — Recommender Systems, in order to recommend correctly, demand huge information related to the past transactions and behavior of the user. In the events, where the data is inconsistent or sparse, the systems show a decline in its predictions or recommendations. Here we propose a new preference elicitation system that is based on preference from closed user group. The implicit behavior of the user is tracked when the user picks up an item. The explicit behavior is tracked by the user-ratings for the given item. The user-preference is computed on a memory-based model taking in account the implicit behavior. The peers are identified based on user-similarity on the explicit-preference indicator. The peer preferences are used on the test-dataset to find the percentage of preference that could be matched. The algorithm has been tested on MovieLens dataset and has given competitive results over the comparable techniques like sliding window method or collaborative filtering methods in isolation.**

Keywords - Memory-based Recommender; Changing Preference; Movie Lens Mining.

I. INTRODUCTION

Researches on Recommender Systems have focused on two principal aspects, Rating Prediction and Ranking. In case of Rating Prediction, the user behavior is explicit, meaning that whatever the user feels about a particular item, it is represented in a predefined-scale. Generally, higher rating indicates higher preference. Such explicit ratings have been successfully implemented by many e-commerce portals like Amazon. Such ratings are collated, compared with similar users and then they are aggregated based on likelihoods. The most popular method for getting such likelihood is collaborative filtering [26, 27, 28]. In case of collaborative filtering the data from several users are compared and a set of users, similar to the given user is found out. Once a set of similar user has been obtained, the research focus shifts to the ranking problem whereby the users are ranked according to some pre-defined metrics. The procedure discussed requires accurate data input from the users. The problem with explicit ratings is that human by nature is lazy and they rate only a small fraction of items which they have purchased. Another problem with explicit rating is the validity [15]. When the rating is immediate, sometimes it might not capture the long-term user gratification with a given item. An item just after being procured might immediately fetch a premium rating but only when the performance degrades after sometime, the perception changes. The user, in most cases, does not return to re-rate such items or the functionality is missing altogether. Hence, the two major issues that the researcher has to deal with the explicit ratings are the rating validity and the data sparsity.

The implicit rating on the other hand has been successfully employed in the web-mining space, where the higher returns or clicks on an item indicate the user propensity for the given item-set. One major challenge here is identifying the right item that suits the user taste. During an implicit rating case, the data tends to be noisy and sometimes for a user with habitual browsing behavior, it becomes difficult to find the set that closely defines the user preference. This calls for a method that would try to aggregate or filter the user preference first, categorize the aggregation and based on the relative occurrences, put them in various groups. Once such segregation has been done, the user ratings if available might be used to fine-tune such categories or groups.

Moreover, the user preference changes with time and the occurrence of such shift is difficult to identify. In a genuine case, the user develops a new interest that finds its own place in user preference patterns. Sometimes such new interest erodes the old-interests slowly with time. In another kind of case, due to over-specialization, the user gets bored with his earlier likings. The earlier preferences have become permanent in nature but the user might choose to explore newer preferences. During such a case, the preference takes a sharp bend and explores through several interest categories before stabilizing on one or a few interests that the user has found suitable. Thus, the same user when asked to rate identical items at different time-frame behaves differently

In this particular paper, we will discuss a heuristically guided algorithm that creates the user preference dynamically based on recency pattern of the implicit ratings. The explicit user ratings have been used to determine the user taste based on similar-likelihood and rating patterns of interests. The algorithm has been tested on the MovieLens dataset for 1-million ratings. Results are promising against comparable algorithms like Sliding Window [25].

The paper has been organized as follows. Section I provides an introduction to the recommender systems. The section briefly discusses the necessity of our work. Section II highlights the motivation for our work. Section III talks about some existing works in this particular domain and outlines our domain of work. In Section IV, we have

 ISBN: 978-1-61208-206-6

introduced the proposed model and the algorithm that has been designed for this process. In Section V, we validate this model using a popular dataset, the Movielens-1M dataset. Section VI discusses the results and future scope of work. Section VII concludes the paper with future direction of work.

II. MOTIVATIONS

Recommender Systems make heavy use of Machine Learning [29, 30], and, as a result, the computation is very expensive in terms of both computation resource as well as time. Due to such huge data, small pieces of information or recent changes are initially ignored. This makes the system non-adaptive to sudden trends. From the literature, we have found instance that refers to the use of complicated techniques like genetic algorithm [21] in machine learning and its widespread application. We have also had cases where swarm intelligence like that of Ant-Colony Optimization (ACO) being used for the above purpose [22]. While these algorithms have been proved effective in some cases, there has been limited effort to use the human-memory model to find out the preference of human subjects. Our objective has been to construct a model that is computationally not very intensive, easy to understand and can mimic human behavior. The proposed human-memory model is borrowed from clinical research [23, 24] and unlike most other methods can work in isolation. To model the friend suggestions, popular techniques like collaborative filtering have been used to find a similar user set based on the ratings pattern. The friend's preferences are used as a support for the user decision in the event that the user needs decision for unexplored items. The contribution of the work is to use the biological framework of human memory and blend the same with the social framework of friend selection by the means of likelihood to identify nearest neighbors.

.

III. EXISTING WORKS

Recommender Systems have gained quite prominence due to the increasing popularity of e-commerce. Research in this arena has attracted attention of academia as well as the industry [1]. When it comes to recommender system, the problem that people likes to tackle are mainly the prediction problems.

Prediction problems as outlined earlier require huge temporal data in order to make meaningful predictions. The popular methods to deal with temporal data are the ones that apply time-weighted function to the data points. This further amplifies the issues with data sparsity since missing data-points tend to insert negative-bias in the system.

Most of the previous works [8, 9, 10] are mainly different variations of assigning weights to data. The basic approach used is to assign a greater weight to new data, and discount old data so as to decay or remove the effect of old data. In [8] decreasing weights are assigned to old data, which implies that the old-preference even if currently active gains a lesser importance. In [9], Massa and Avesani have introduced the concept of item-similarity when considering time-based weights. They have added the new weights to handle the issue of reoccurrence of user interests. In [10], Massa and Avesani propose using weights for items based on expected future preference of the users.

Most of the earlier works have formulated the problem as a prediction accuracy problem. In [3], Saha et al. have proposed a novel method to capture the interest-shift has been computed for individual based on their temporal transaction history. The method decays the older preference only when the recurrence pattern stops. We have simplified the model reducing the memory-components from five holders to three holders akin to the human memory [23] and adding the social component as support from the neighbors. Individual preference only holds limited information. The potential preference space might cover additional features that are equally likely to recur when exposed to the similar user. Mining preference pattern of a closed group [18] helps uncover additional knowledge for items unseen by the user.

IV. PROPOSED MODEL

In the proposed model, we are interested to establish a relationship between the user and their closed group. We would like to show that when friends are selected properly, they can help increase the accuracy of predicting the user preferences.

While running the implicit preference generation process, the presence of an item / item-set in a user transaction indicates the user affinity towards the given category of item(s). Such implicit indicator is then mapped to the user transaction. The feature-set from the item in the given transaction is then derived and finally the feature(s) are put in the users' preference basket. Both the aspects of time and preference are taken into account during such categorization process. The process finally ends with the generation of the user preference-basket that has categorization of the user's permanent as well as the changing tastes. The implicit preference mining is a little deviation from the methods suggested in [3]. We have considered 3 categories of user memory, based on human-memory models as proposed by Atkinson and Shiffrin [23], namely sensory memory, short-term memory and long-term memory.

In the proposed algorithm for implicit preference generation, there are mainly three entities; the User Memory, the transactions per user and the preference signature in each of the transactions. The features and uses of the different memory levels are discussed as below:

Sensory Memory: This memory is very short-lived for the human and can be thought as the touch and go memory. Sensory memory doesn't even register with our conscious mind. Only when the sensory memory is nurtured, we start noticing it and then it registers in our conscious brain. In our algorithm, first time preferences are initially stored in the sensory memory. If such preference doesn't repeat, it quickly gets wiped out leaving any traces.

Short Term Memory: Unlike the sensory memory, short-term memory do gets registered in our conscious mind. The short-term memory is like a temporary working register. We use the short-term memory for a specific purpose and don't care to carry that for future use. For example, we won't have any problem remembering our last visit to the

supermarket. With little difficulty, we might even recall the amount spent in the last trip. However unless someone is extremely meticulous or gifted with great memory, the quantity and the amount paid per item would be difficult to remember. This memory remained with us during the last visit to the store and got into the oblivion zone since it was not necessary. The similar model has been used for the short-term memory register in our algorithm. Above a certain threshold, the sensory-memory becomes short-term memory. Since the memory has to be retained for some-time, we allow some grace-period when the penalty-function remains dormant. If such preference doesn't get repeated during the grace-period the penalization functions starts and quickly erodes it off. This feature has been considered in the algorithm allowing differential grace-window for preferences that lie at different spectrum of the memory model.

Long-Term Memory: This might be thought of as a permanent memory, which remains with the user for a considerable time. The elements in this memory repeats regularly hence making it permanent. In the proposed algorithm, the long-term memory has been allowed a greater grace-window. Once within the long-term memory, the process of erosion is also much slower. Thus once an item gets into the long-term zone, only few recurrence incidents are required to retain the status of such memory. Of course even with such leeway, long-term memory can also fade and even goes into oblivion. This also has been taken care of by the drifting model. Once a preference at the long-term memory stops recurring, the penalty starts kicking in. The penalty gets intensely steeper and finally such preference gets removed from the memory altogether.

The proposed implicit-preference algorithm outlined below (Algorithm-1) is sensitive to the relative occurrence of the transactions. Depending on the sequence, the user-preference is affected. This is totally in line with the assumptions that the more recent transactions that occur repeatedly are more prone to remain in the working memory compared to old ones that occur just once in awhile.

The algorithm starts with an empty memory per user. The non-performance penalty is $q(M_j)$ and the waiting window is $w(M_j)$. For every user, each transaction is decomposed to individual items. When the memory is initially empty, the item(s) are placed in the sensory memory. With more reinforcement of the same item in future transactions, the item moves from the sensory memory to the short-term memory and likewise with further repetitions, it finds place in the long-term memory. This is akin to the working of human memory where continuous reinforcement or trainings makes a memory permanent. In the event, when the item(s) stops recurring, a penalty is applied depending on the stage of the memory, the item is currently placed, being higher for volatile memories and lower for permanent memory. The waiting-window/grace period varies once again depending on the state of the item, being higher for the permanent and lower for sensory.

Algorithm - 1

```
let U denote a set of users
while (all_user_not_processed)
  foreach user Uq do
    initialize user_memory();
    repeat
      foreach item I in user transaction
      M ← compute_memory_matrix (Uq, I);
    until no_more_transactions
  goto next_user()

procedure compute_memory_matrix(Uq, I)
  decompose I into preference indicators dj
  If (dj in Mj)
    then Mj(dj) ← Mj(dj) + 1
    If Mj ≠ MLong-Term and Mj(dj) >= threshold(Mk)
      Mk(dk) ← Mj(dj) ∀ threshold(k) > threshold(j)
      delete dj from Mj
      update memory
  else
    Insert dj in Msensory
    set Msensory(dj) ← 1
    set non_occurence(dj) = 0
  Do process_penalty(I)
end-procedure

procedure process_penalty(I)
  For all dj not in I
    If activation_function h(Mj, wait(dj)) = 1
      then Mj(dj) ← Mj(dj) - Penalty(Mj)
     if Mj(dj) < threshold(Mi)
       if Mj = MSensory
        then delete dj from MSensory
       else
         Mi(di) ← Mj(dj) ∀ threshold(i) < threshold(j)
         delete dj from Mj
   else wait(dj) ← wait(dj) + 1
end-procedure
```

After the implicit-preference building process is complete, the preference footprints are analyzed for every user for the presence of the explicit indicators in the form of user-ratings. The common set of transactions is selected for every pair of users. Unlike the Algorithm 1, the order of the data doesn't matter when the correlation is computed from the explicit ratings.

The friend selection algorithm is a computing-intensive task. It can be done over the entire set of items common to the users and can be performed offline as well if the matching transactions are known beforehand. In our work, we have generated the preference-lists from the training dataset. Once such preference generation is completed, the matching transactions are used to find the friends of the user. We have considered the Pearson-Correlation coefficient [31, 32] with the ratings data to get user-user similarity. The Pearson-Correlation coefficient is computed for every user-pair and stored in the database for use during the testing process. We have run the testing process, varying the number of friends and computed the results.

The formula for the Pearson-Correlation coefficient (r) is as follows:

$$r = \frac{\frac{1}{n}\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(X_i - \bar{X})^2}\sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - \bar{Y})^2}}$$

where,

X_i = rating of user X for an item i

Y_i = rating of user Y for an item i

$\bar{X}$ = average rating of user X for common items

$\bar{Y}$ = average rating of user Y for common items

n = items rated in common

The Pearson-correlations computation process is resistant to grade-inflation problem. Thus even if the two users rate items differently, the score would be inline as long as their rating pattern for the corresponding interest matches. The value of the Correlation-index (r) lies between -1 and +1. When the value is +1 it means that the two-users are perfectly correlated. When the value is -1, it means that the users tastes are opposite in nature. When the value of correlation is 0, it means that the two users are not at all correlated.

V. MODEL VALIDATION

To validate our proposed model, we have run the process on the Movielens data with 1 Millions ratings. The data has 1,000,209 ratings from 6,040 users on about 3900 movies and has been recorded from years 2000 onwards. The data has been arranged chronologically. We have used the ratings data and the movie-definitions file.

The following attributes were selected before our model building process:

- USER_ID (a unique user identifier)
- MOVIE_ID (a unique movie identifier)
- MOVIE_GENRE (genre for the given movie)
- USER_RATING (rating on a scale of 1 to 5)
- TIMESTAMP

We have had followed a 80:20 partition scheme on the data, where 80% data is used for training and 20% for validation. For the convenience of model-building, the first 20% data has been exclusively reserved for training purpose. A quarter of the rest 80% data is picked at random for cross-validation purpose using a Monte-Carlo method – the remaining gets added to the training-set. We have deliberately not resorted to a k-fold validation since each of the k-parts needs to be mutually exclusive. In our case, each subsequent transaction is dependent on the previous one making a difficult proposition for aforesaid technique.

The MOVIE_GENRE has been assumed to be a proxy for the user preference. Users indeed have propensity to movies of various genres. For example some users might find Action movies entertaining while some would prefer Comedy. In total the movies contained a concoction of 18 genres, namely: Action, Adventure, Animation, Children's, Comedy, Crime, Documentary, Drama, Fantasy, Film-Noir, Horror, Musical, Mystery, Romance, Sci-Fi, Thriller, War and Western. Each movie had at-least one genre with the maximum number of genre for a particular being as high as six. A majority of the movies were composed of one or two genres.

Figure-1 gives an idea of the working of the algorithm.

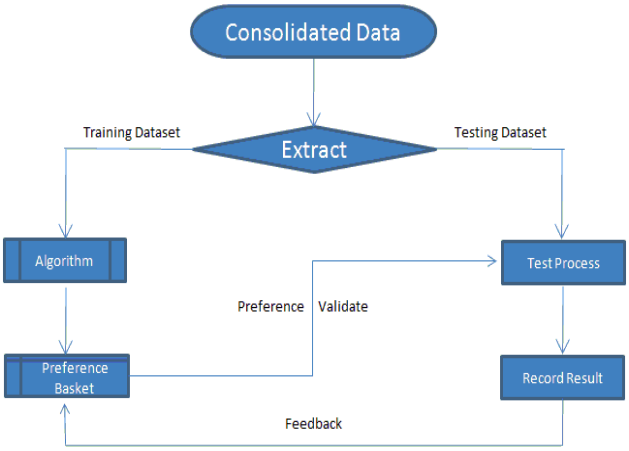


FIG-1: MODEL BASED ON ALGORITHM 1

A. Environment Setup

The window size before decay starts and the non-performance penalty is context dependent. We have found the median and the mode values for the "mean transactions

to repeat". For the Movielens-1M dataset, the median value came out as 3 and the mode was 1. Based on the findings, the window size is constructed as below. Since every transaction adds 1 to the strength of the preference, the quartile values have been fixed as the non-performance penalty depending on the preference category.

Non-performance penalty $q(M_j)$ for different categories j

$$p(Mj) = \begin{Bmatrix} 0.25 : j \in [\text{"}Long-term\text{"}] \\ 0.50 : j \in [\text{"}Short-term\text{"}] \\ 0.75 : j \in [\text{"}Sensory\text{"}] \end{Bmatrix}$$

The wait-window $w(M_j)$ for different categories j

$$w(Mj) = \begin{Bmatrix} 4 : j \in [\text{"}Long-Term\text{"}] \\ 3 : j \in [\text{"}Short-Term\text{"}] \\ 2 : j \in [\text{"}Sensory\text{"}] \end{Bmatrix}$$

The activation-function for penalty, $h(j, w)$ assumes binary values for a given delay " w " for memory type " j "

Thus h ("Sensory", 2) = 1 while h ("Long-Term", 2) = 0.

B. Algorithm Workings

During the testing-phase, the Algorithm-1 runs on the user's transactions and separates the genres. Once the genre gets separated, the algorithm determines that whether such genre should be considered for Sensory, Short-Term or Long-Term memory. Once the algorithm completes, we get a preference-basket per user having few to none entries per memory category. We had run the explicit process in parallel to compute the user-user similarity via the Pearson Correlation Coefficient. For a given user, a friend is defined by the top few users that have the highest correlation value.

The friend generation process in two-parts. In the first case, we consider the explicit user ratings and then determine the friends from the explicit-ratings applying Pearson on it. Next we use the user-memories that have been obtained as a result of the algorithm. Each of the preference components in the memory has a score. Dividing the score by the number of observations for each component, we get strength of the memory component or the "MemStrength" matrix. Pearson correlation is applied on the matrix per preference value. This result is also stored.

C. Algorithm Testing

We have tested the algorithm individually considering the best five friends of the users and seeing that how their memory preferences can support the user preferences. We have not considered the sensory memory but only the short-term and the long-term memory is considered while looking for a match. For the testing-purpose, we have taken the sensitivity expressed in percentage, defined as:

$$sensitivity = \frac{true\ positive}{true\ positive + false\ negative}$$

Since a movie might comprise multiple genres and the user might only be interested in one or few genres from the set, hence we have taken a majority matching approach. If the algorithm can match the majority of the genres in a given movie in the test set, then we consider the case as a "true positive" case.

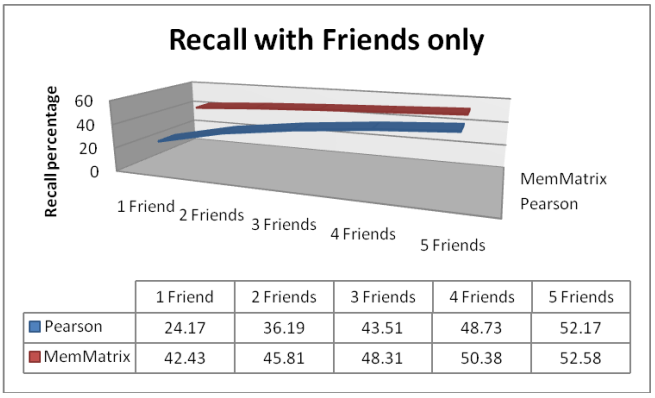
VI. Results and Discussions

The algorithm could successfully predict 73.244% of the total genres present in the test-set taking one user at a time.

Using a majority match in the *Algorithm Testing* part of Section V, the algorithm could properly account for the genre composition in 86.236% of the test dataset.

The friend computation has been outlined in *Algorithm Working* in Section V. Taking assistance from five "Pearson Friends" using explicit ratings, the genre composition prediction increased to 93.30%. Using the same number of friends from our "MemStrength" matrix, the algorithm could successfully account for 94.04% of the genre composition in the test dataset.

We have also computed a scenario, where the friend's-preference is taken as a proxy of the user's preference. The sensitivity or the recall result is shown below:



	1 Friend	2 Friends	3 Friends	4 Friends	5 Friends
Pearson	24.17	36.19	43.51	48.73	52.17
MemMatrix	42.43	45.81	48.31	50.38	52.58

FIG-2: MODEL BASED ON ALGORITHM 1

The results show that the "MemMatrix"- based friend computation starts with a high-recall or sensitivity value of 42.43% compared to 24.17% when the assistance of only one friend is sought for. The performance of the two methods show that the Pearson based friend process moves at a higher rate compared to the "MemMatrix" based friends and performances are nearly equal with 5 friends. The slow

 ISBN: 978-1-61208-206-6

movement of the "MemMatrix" is owing to the fact that the process selects users with very similar tastes so dispersion is relatively lesser. The test-process accounts for 20% of the dataset so during the latter part of the data, some new memories might be created as well as destroyed. Since the friend generation process is a one-time activity, such drifts could have resulted in a change in the friend composition as well. This has not been captured in the current work. As discussed earlier, the process of friend creation is a very computation intensive process so it would be a good area to look into for future works.

We have considered a step-descent function for memory decay process. The time-decay functions have performed poorly as discussed during the literature survey part. However, there are other decay functions like the Gaussian-model. It would be interesting to create hybrid decay models and test the algorithm.

To benchmark this work, we compared the performance with a sliding window technique. In a sliding window technique, the items from the last few transactions are considered for all practical purposes. The name is indicative of the fact that the preference window slides by one-unit at every new transaction. For our model comparison, we have used the last n transactions as the proxy of the user preference. In this example, we consider all the unique items that appear in the last 5 transactions as the user preference set. The sensitivity results are as follows:

TABLE-1: SENSITIVITY/RECALL WITH A SLIDING WINDOW SIZE 5

Size of the Window	Recall Values (rounded)
3	54%
4	61%
5	66%

The result above indicates that in the present problem scenario, our proposed algorithm has devised superior results. This can be justified by the fact that our algorithm holds both the short-term and the long-term preferences, computed through a stepwise non-volatile mechanism contrary to the sliding-window technique described above that only takes into account the preferences in the most recent transactions. This forsakes the long-term preferences of the user for the short-term ones.

VII. CONCLUSION AND FUTURE WORK

We have introduced a human-memory based technique to find the user preferences taking into account the implicit behavior and transaction atomicity. The process creates dynamic user memory matrix that can be updated based on simple rules. When the recommendation needs to be made under dynamic situation using limited memory and time-resources, our experiment shows that the proposed biologically inspired heuristic is expected to perform better than the comparable existing techniques. Members in isolation can be observed. The algorithm has been tested on a huge dataset and result motivates further research in the direction. This algorithm works on the principle of memory models and thereby opens up the scope for application to allied domains of machine learning and artificial intelligence. We would like to extend this work in the domains of travel, more specifically weekend travels whereby user's travel interest could be categorized and recommendations can be suggested. We would be most interested in using the model to control viewer censorship in televisions whereby contents to the young audience would be suggested as per preference set by the parents or the guardians. With the smart-televisions finding their ways in home, we hope to get enough data to validate our model for television censorship.

ACKNOWLEDGMENT

The authors would like to thank Prof. Sanjog Ray of IIM Indore for his valued feedback during the ideation and evaluation of this particular work. Special thanks for Prof. Indranil Bose and Prof. Asim Pal for valuable inputs to the memory model.

REFERENCES

[1] P. Massa and P. Avesani : "Trust-aware collaborative filtering for recommender systems". In: Proceedings of the Federated Int. Conference On The Move to Meaningful Internet, (2004)

[2] J. Herlocker, J. Konstan, A. Borchers and J. Riedl : "An Algorithm Framework for Performing Collaborative Filtering". In: Proceedings of SIGIR, ACM, 1999, pp. 77-87.

[3] S. Saha, S. Majumder, S. Ray, A. and Mahanti : "Categorizing User Interests in Recommender Systems". In: New York, 2010, pp. 282-291.

[4] J. Donaldson : "A hybrid social-acoustic recommendation system for popular music". In: Proceedings of the 2007 ACM conference on Recommender systems, 2007, pp. 187–190.

[5] P. Viappiani, P. Pu, and B. Faltings : "Conversational recommenders with adaptive suggestions". In: Proceedings of the 2007 ACM conference on Recommender systems, 2007, pp. 89–96.

[6] A. Z. Broder : "Computational advertising and recommender systems". In: Proceedings of the 2008 ACM conference on Recommender systems, 2008, pp. 1–2.

[7] M. Kagie, M. van Wezel, and P. J. F. Groenen : "Choosing attribute weights for item dissimilarity using clikstream data with an application to a product catalog map". In: Proceedings of the 2008 ACM conference on Recommender systems, 2008, pp. 195–202.

[8] P. Massa and P. Avesani : "Trust-aware recommender systems". In: Proceedings of the 2007 ACM conference on Recommender systems, Minneapolis, 2007.

[9] P. Massa and P. Avesani : "Trust-aware Bootstrapping of Recommender Systems". In: Proceeedings of ECAI Workshop on Recommender Systems, Italy, 2006.

[10] P. Massa and P. Avesani : "Trust metrics on controversial users: balancing between tyranny of the majority and echo chambers". In: International Journal on Semantic Web and Information Systems.

[11] B. Mobasher, R. Burke, R. Bhaumik and C. Williams : "Towards Trustworthy Recommender Systems: An Analysis of Attack Models and Algorithm Robustness". In: ACM Transactions on Internet Technology, 2007, pp. 23-38.

[12] A. Rashid, G. Karypis, and J. Riedl : "Learning Preferences of New Users in Recommender Systems: An Information Theoretic Approach". In: SIGKDD Explorations, 2008, pp. 90-100.

 ISBN: 978-1-61208-206-6

[13] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl : “Item-based Collaborative Filtering of Recommendation Algorithms”. In: Proceedings of the 10th International WWW Conference, Hong Kong, 2001.

[14] S. Berkovsky, T. Kuflik, and F. Ricci : “Distributed collaborative filtering with domain specialization”. In: Proceedings of the 2007 ACM conference on Recommender systems, 2007, pp.33–40.

[15] O. C. Santos : “A recommender system to provide adaptive and inclusive standard-based support along the eLearning life cycle”. In: Proceedings of the 2008 ACM conference on Recommender systems, 2008, pp. 319–322.

[16] X. Amatriain, J. Pujol, N. Tintarev, and N. Oliver : “Rate it again: increasing recommendation accuracy by user re-rating. In: Proceedings of the third ACM conference on Recommender systems, no. 4, 2009, pp. 173–180.

[17] F. P. Lousame and E. Sánchez : “View-based recommender systems”. In: Proceedings of the third ACM conference on Recommender systems, 2009, pp. 389–392.

[18] B. M. Marlin and R. S. Zemel : “Collaborative prediction and ranking with non-random missing data”. In: Proceedings of the third ACM conference on Recommender systems, 2009, pp. 5–12.

[19] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos : “MoviExplain: a recommender system with explanations”. In: Proceedings of the third ACM conference on Recommender systems, 2009, pp. 317–320.

[20] N. Kawamae, H. Sakano, and T. Yamada : “Personalized recommendation based on the personal innovator degree”. In: Proceedings of the third ACM conference on Recommender systems, 2009, pp. 329–332.

[21] H. S. Lopes, M. S. Coutinho and W.C. Lima : “An evolutionary approach to simulate cognitive feedback learning in Medical Domain”. In: E. Sanches, T. Shibata and L. A. Zadeh(eds.) Genetic Algorithm and Fuzzy Logic Systems: Soft Computing Perspectives, Singapore:Word Scientific, 1997,pp. 193-207.

[22] M. Dorigo, G. Di Caro and L. M. Gambardella : “Ant algorithms for discrete optimization”. In: Artificial Life, 5(2), 1999, pp.137-172.

[23] R. C. Atkinson and R. M. Shiffrin : "Chapter: Human memory: A proposed system and its control processes". In : K. W. Spence and J. T. Spence, The psychology of learning and motivation (Volume 2). New York: Academic Press, 1968, pp. 89–195.

[24] U. Fayyad, G. Piatetsky-Shapiro, P. Smyth and R. Uthurusamy : Advances in knowledge discovery and data mining. Menlo Park, CA: AAAI Press/ The MIT Press, 1996

[25] Chang-Hung Lee, Cheng-Ru Lin, and Ming-Syan Chen : “Sliding-window filtering: an efficient algorithm for incremental mining”. In: Proceedings of the tenth international conference on Information and knowledge management (CIKM '01), Henrique Paques, Ling Liu, and David Grossman (Eds.). ACM, New York, NY, USA, pp. 263-270.

[26] John S. Breese, David Heckerman, and Carl Kadie : “Empirical analysis of predictive algorithms for collaborative filtering”. In: Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence (UAI'98), Gregory F. Cooper and Serafín Moral (Eds.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998, pp. 43-52.

[27] Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers, and John Riedl : “An algorithmic framework for performing collaborative filtering”. In: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '99). ACM, New York, NY, USA, 1999, pp. 230-237.

[28] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl : “Evaluating collaborative filtering recommender systems”. In: ACM Trans. Inf. Syst. 22, 1 (January 2004), pp. 5-53.

[29] Raymond J. Mooney and Loriene Roy : “Content-based book recommending using learning for text categorization”. In: Proceedings of the fifth ACM conference on Digital libraries (DL '00). ACM, New York, NY, USA, 2000, pp. 195-204.

[30] Geoffrey I Webb, Michael J Pazzani and Daniel Billsus : "Machine Learning for User Modeling". In: User Modeling and User-Adapted Interaction, Springer Netherlands, vol 11, issue 1, 2001, pp. 19-29

[31] Grace A. Falciglia and Philippa A. Norton : “Evidence for a genetic influence on preference for some foods”. In: Journal of the American Dietetic Association, volume 94, issue 2, February 1994, pp. 154-158

[32] Tong Queue Lee, Young Park, Yong-Tae Park, A time-based approach to effective recommender systems using implicit feedback, Expert Systems with Applications, Volume 34, Issue 4, May 2008, pp. 3055-3062.

 ISBN: 978-1-61208-206-6