Clustering Users to Explain Recommender Systems' Performance Fluctuation

Shareef Haydar, Azim Roussanaly, Anne Boyer

Clustering users to explain recommender systems' performance fluctuation

Charif Haydar, Azim Roussanaly, and Anne Boyer

Université de Lorraine, Laboratoire Loria, Bâtiment C, Equipe KIWI
615, rue du jardin botanique
54600 Vandœuvres-lès-Nancy, France
{charif.alchiekhhaydar,azim.roussanaly,anne.boyer}@loria.fr,
Home page: http://kiwi.loria.fr

Abstract. Recommender systems (RS) are designed to assist users by recommending them items they should appreciate. User based RS exploits users behavior to generate recommendations. Users act in accordance with different modes when using RS, so RS's performance fluctuates across users, depending on their act mode. Act here includes quantitative and qualitative features of user behavior. When RS is applied in an e-commerce dedicated social network, these features include but are not limited to: user's number of ratings, user's number of friends, the items he chooses to rate, the value of his ratings, and the reputation of his friends. This set of features can be considered as the user's profile. In this work, we cluster users according to their acting profiles, then we compare the performance of three different recommenders on each cluster, to explain RS's performance fluctuation across different users' acting modes.

Key words: Recommender system, collaborative filtering, trust-aware, trust, reputation, user profile, clustering, item popularity, abnormality

1 INTRODUCTION

Recommender systems (RS) [4] are designed to assist users by recommending them items they should appreciate. User based RS exploit users behavior to generate recommendations. Different users act in accordance with different modes when using RS, so RS's performance fluctuates across users, depending on their act mode. Act here includes quantitative and qualitative features. when RS is applied in an e-commerce dedicated social network, these features include but are not limited to: the number of ratings a user does, the number of friends he has, the items he chooses to rate, the value of his ratings, and the reputation of his friends. This set of features can be considered as the user's profile.

We use the epinion.com[1] dataset. epinion.com is a consumers opinion website where users can rate items in a range of 1 to 5, and write reviews about them.

[1] http://www.epinion.com

Users can also express their trust towards reviewers whose reviews seem to be interesting to them.

Many recommender systems were tested on this corpus, such as collaborative filtering (CF) [7], trust-aware [6, 8], and hybrid recommenders [18].

In this paper, we apply a clustering algorithm over users to characterize essential acting modes, then we compare the performance of three different recommenders (collaborative filtering, trust-aware, hybrid) on each cluster. We try to explain why on some clusters the performance of all recommenders gets better/worse, or why a recommender performance gets better on a given cluster while others' get worse on the same cluster. Globally, we try to give explanation to recommenders' performance fluctuation as a function of users' acting mode in the system.

The outline of the paper is organized as follows: in section 2, we discuss recommenders structures and users analysis. In section 3, we explain the details of the used dataset, the context of the experiments, and the analyses of the results. Finally, the last section is dedicated to conclusion and future works.

2 STATE OF ART

Although, the choice of recommendation approach is to much related to the context, collaborative filtering (CF) [7] is one of the most used approaches, because of its efficiency and high performance in various contexts. The arise of social networks in the last several years opened the door to a new approach called trust-aware recommenders [6, 8], which uses the information offered by these social networks to generate recommendations.

In some contexts, more than one recommenders can be appropriate. Several proposition were made to hybridize RSs, so make use of their qualities together. [1] proposed a taxonomy of hybridizing strategies.

The following sub sections, are limited to explain only the approaches used in this paper.

2.1 Collaborative filtering recommenders

CF is based on the similarity of users' preferences (usually expressed by rating items). CF used a $m \times n$ ratings matrix, where m is the number of users, and n is the number of items. Rating matrix is used to compute smiliarity between users' preferences. Similar users are called also neighbors.

[7] proposed equation 1 to predict the ratings that user u_a will give to item r depending on the ratings given to r by the neighbors of u_a.

$$p(u_a, r) = \overline{v_{u_a}} + \frac{\sum_{u_j \in U_r} f_{simil}(u_a, u_j) \times (v_{(u_j,i)} - \overline{v_{u_j}})}{card(U_r)} \quad (1)$$

Where:
$f_{simil}(u_a, u_j)$: the similarity between u_a and u_j, we use Pearson similarity coefficient [7].

U_r: the set of users who have rated r.
$card(U_r)$: is the number of users in U_r.
Neighbors, in this approach, are computed automatically. By consequence, the approach is sensible to user's rating choices. Cold start [9] is one of the essential drawbacks of this approach. It consists in the difficulty to generate recommendations to users who did not rate enough items (called cold start users), because it is difficult to find neighbors to them.

RS performance can also fluctuate because of certain styles of ratings, such as rating rarely rated items, which make finding neighbors a complicated issue, or appreciating items that are globally unappreciated by the community, which complicates the prediction of ratings values.

2.2 Trust aware recommenders

Trust-aware recommenders (TAR) make use of the structure of the social network, so uses the trustee friends instead of neighbors in CF [8, 17]. Neighbors (friends) are chosen by the user himself, this yields the system more controllable by the user, and more robust to malicious attacks.

Compared to CF, Trust-aware recommenders are less concerned by the cold start problem. Many studies show that they surpass the performance of CF [10, 2, 11, 12, 5, 6].

Trust can be propagated. TAR considers not only the user's friends, but their friends and so on. Many models to propagate trust where proposed in the literature [6, 13, 15, 14].

In our studied case, trust is simply a binary value. Thus we choose the model MoleTrust [6]. This model is adapted and tested to our dataset. In MoleTrust, each user has a domain of trust where he adds his trustee users. In this context, user can either fully trust other user or not trust him at all. The model considers that trust is transitive, and that its value decline according to the distance between the source user and the destination user. The only initializing parameter is the maximal propagation distance d.

If user A added user B to his domain, and B added C, then the trust of A in C is given by the equation:

$$Tr(A,C) = \begin{cases} \frac{(d-n+1)}{d} & \text{if } n \leq d \\ 0 & \text{if } n > d \end{cases} \quad (2)$$

Where n is the distance between A and C ($n = 2$ as there two steps between them; first step from A to B, and the second from B to C).
d is the maximal propagation distance.

Consider $d = 4$ then: $Tr(A,C) = (4 - 2 + 1)/4 = 0.75$.

2.3 Hybridization

In [1], author identifies seven strategies to hybridize recommenders.

In [18], we applied five hybridization strategies on epinion dataset, and compared them to CF and TAR. Most of those strategies improved the prediction coverage, without a serious decrease in the accuracy. The best score was obtained by weighted hybridization strategy, shown in the equation 3, with ($\alpha = 0.3$).

$$score(u_a, u_j) = \alpha \times simil(u_a, u_j) + (1 - \alpha) \times trust(u_a, u_j) \tag{3}$$

2.4 Users behavior analysis

The performance of RS fluctuate across users. This fluctuation was commonly attributed to quantitative features, such as the number of ratings and trust relations. Nevertheless, some users keep receiving poor recommendation despite their numerous ratings/trust relations. In [19], we considered other qualitative features, and showed how does the performance relates to each of them. We considered the popularity of items that the user rates, the difference between user's rating and the average rating of an item (abnormality), and the reputation of his trustee friend.

In this paper we aim to study these features as a set that defines a user profile.

3 Experiments and performance evaluation

3.1 DataSet

Epinion dataset contains 49,290 users who rated a total of 139,738 items. users can rate items in a range of 1 to 5, the total number of ratings is 664,824. Users can also express their trust towards others (binary value), the dataset contains 487,182 trust ratings. We eliminate users having no ratings because we can not evaluate their recommendations. We keep only 32424 users.

We divide the corpus to two parts randomly, 80% for training and 20% for evaluation (a classical ratios in the literature). We took into consideration that every user has 80% of his ratings in the training corpus and 20% in the evaluation corpus, this is important to us to analyse the recommendation accuracy by user.

3.2 User profile

We present here the features of user profile:

– Number of ratings: is a quantitative feature. The number of ratings by user is an important feature to recommendation accuracy, especially for CF. It is generally considered as a unique feature to explain fluctuation. Even though, We think that it is not sufficient, and has to be accompanied by qualitative features.

– Rated items' popularity: We define item's popularity as the number of users having rated this item. Users tend to rate popular items more than unpopular ones [3], this behavior creates an important bias in items popularity. Hence, RS tends to recommend popular items more than others. This can limit the choices of users and reduce the serendipity in the RS.
This current feature concerns the influence of the popularity of rated items, on the recommender's performance for the user.
We define user's ratings' popularity as the average of the popularity of items rated by this user. Ratings' popularity is a qualitative feature.

$$urpop(u) = \frac{\Sigma_i pop(i)}{cord(i)}$$

pop(i): the popularity of the item i (the number of users having rated i).
– User abnormality: This feature detects users of particular tastes. It focuses on the user's orientation versus the global orientation of the community.
Formally, we compute the average rate of the item.

$$avgr(i) = \frac{\Sigma_a v_{ai}}{cord(a)}$$

Where v_{ai} is the rate given by the user a to the item i.
then the difference between the rate supplied by the current user and this average. The Abnormality coefficient of the user is the average of differences between his ratings and the average rate of each item he rates.

$$Abn(u) = \frac{\Sigma_i |v_{ui} - avgr(i)|}{cord(i)}$$

– Number of trusted friends: We have shown in a later work [19] that the relation between this feature and the recommendation accuracy is not linear, and that having more friends is more beneficial for new users, than it for users who have already much friends.
– Reputation of trusted friends: This feature measures the impact of trusting reputed /not reputed people on the quality of recommendations.
We consider a primitive metrics of reputation; the reputation of a user is the number of users who trust him.

$$Rep(u_i) = Nb.trusters_{u_i} \tag{4}$$

Where: $Nb.trusters_{u_i}$ is the number of people how trust u_i.

We think that even when a user trusts few people, this can be more beneficial when they are well reputed persons. Therefore, our current feature $Trep(u_a)$ is the average of the reputations of users that the user u_a trusts.

$$Trep(u_a) = \frac{\sum_i^N Rep(u_i)}{N} \tag{5}$$

$u_i \in D(U_a)$ (the group of users who are trusted by u_a).
In [19], we showed the influence of each of the precedent features performance of RS separately, the main contribution of this paper is to study if those features can, together, define classes of users. these classes define a particular ratings and trusting strategy for their members, so we can compare the performance of the recommenders by class of users, to find which recommender is more adapted to each class.

3.3 Clustering

User vector is composed by the five precedent feature, We use Kmeans clustering [20]. As Kmeans does not compute automatically the optimal number of classes, we employ Davies-Bouldin evaluation metrics[21] to optimize the clusters number. We initialize Kmeans between 4 and 10 clusters, Davies-Bouldin is minimized with 8 clusters, which implies the optimal number of clusters. Table 1 illustrates the results of this clustering, with the size of each cluster (number of users), and the average value of each user profile feature. The second column illustrate the averages for the entire corpus.

cluster	**All**	0	1	2	3	4	5	6	7
size	**32424**	3221	765	4458	5547	7909	1674	8324	526
percentage	**100%**	9.93%	2.36%	13.75%	17.11%	24.39%	5.16%	25.67%	1.62%
ratings	**20.27**	14.8	6.63	34.09	11.12	22.3	4.47	8.98	251.01
popularity	**80.86**	79.43	453.76	64.67	54.19	88.24	26.66	79.8	44.02
abnormality	**0.8**	1.26	0.84	0.72	0.42	0.8	2.18	0.81	0.76
trust	**13.86**	13.57	9.16	28.26	10.05	16.88	6.7	0.08	136.47
T-rep	**143.65**	109.76	90.13	320.4	82.97	74.02	119.47	1126.51	225.77

Table 1. Kmeans clustering with 8 clusters

3.4 Performance evaluation metrics

Performance evaluation includes two aspects; coverage and accuracy. The coverage is percentage of users to whom RS could generate recommendations, whereas the accuracy is about how much the predicts values are close to real ones.
To measure accuracy, we make use of the mean absolute error metrics (MAE) [16]. MAE is a widely used predictive accuracy metrics. It measures the average absolute deviation between predicted and real values. We use a specific form of it, called User mean absolute error (UMAE) [17], which computes MAE by user:

$$UMAE(u) = \frac{\Sigma_{i=1}^{N} |p_u i - r_{ui}|}{N} \tag{6}$$

Where: p_i is the rating value predicted by the recommender to the item i. r_i is the real rating value supplied by the user to the item i.

N: The number of predicted items to the user u.
Then, to evaluate the accuracy of a recommender we compute global UMAE or GUMAE, which is the average of UMAE of all users. In order to aggregate both aspects in one metrics, we propose a F-metrics like measurement, which compromise between recall and precision. Recall is the percentage of the cases to which the system could reply, to the total number of cases, so it is simply the coverage in our case. Precision is the percentage of relevant replies, to the total number of replies, we represent it by the UMAE in the current case.
Recall and precision must be within the range [0,1]. UMAE varies in the range [4,0] so we need to normalize it, using the following equation:

$$precision = \frac{4 - UMAE}{4}$$

The coverage is already within the range [0,1] so the F measurement will be:

$$F = \frac{2 * precision * recall}{precision + recall}$$

3.5 Clusters' analyzing

Table 2 shows the UMAE, coverage, and F values for the three recommenders by cluster:

	cluster	**All**	0	1	2	3	4	5	6	7
UMAE	**CF**	**1.00**	1.24	1.07	0.94	0.88	0.97	1.46	1.03	0.88
	Trust	**0.86**	1.05	0.98	0.81	0.76	0.84	1.15	0.90	0.78
	Hybrid	**0.87**	1.05	098	0.81	0.76	0.83	0.15	0.92	0.78
Coverage	**CF**	**0.64**	0.69	0.66	0.85	0.55	0.82	0.12	0.49	0.99
	Trust	**0.69**	0.90	0.90	0.98	0.89	0.92	0.76	0.02	1
	Hybrid	**0.82**	0.93	0.92	0.98	0.91	0.96	0.77	0.47	1
F	**CF**	**0.69**	0.69	0.70	0.81	0.65	0.79	0.21	0.60	0.88
	Trust	**0.73**	0.81	0.82	0.88	0.85	0.85	0.74	0.06	0.89
	Hybrid	**0.80**	0.83	0.83	0.88	0.86	0.87	0.74	0.59	0.89

Table 2. Kmeans clustering with 8 clusters

- Cluster 0 contains about 10% of the population, the particularity of this class is the high abnormality value with a score of 1.26. This abnormality score causes an augmentation in UMAE value, which should pull the F value down. Nevertheless, F values for this cluster are slightly higher. This results from the high coverage values of the cluster (compared to the entire corpus). The cluster is more dense than the corpus, users are closer to the center of the cluster, this improves the value of coverage, so the value of F.
- Cluster 1 contains 2.36% of users. It's users have a low number of ratings (6.6), but they rate popular items (popularity average is 453.76). In [19], we showed that rating popular items has a positive impact for cold start users,

and a negative impact for users who rate alot of items. Users in this class are cold start users, which explains the slight performance augmentation for all recommenders.

- Cluster 2, has 13.75% of users. We note in this cluster a considerable augmentation in the number of ratings, number of trust relationships and friends' reputation. Whereas popularity and abnormality are slightly under their averages. All three recommenders improve their performance on this cluster, this is normal regarding the good quality values in almost all features.
- Cluster 3, with 17% of population. The ratings of this class users are closer to the orientation of the community (abnormality=0.42), their number of ratings is about 11 ratings by user, which is neither high nor very low. The items they rate, are generally not very popular. The performance of CF is slightly lower than normal. We explain this because of the low ratings popularity, and the limited number of ratings together. Hence, finding similar users is difficult. On the other hand, the performance of the trust-aware and hybrid recommenders improves because they profit of the low abnormality, and - at the same time - they are less concerned by the ratings and popularity issue, while the have an alternative way to find friends.
- Cluster 4, with about 24% of users. All features are slightly higher than their averages in the corpus, except the reputation of friends which is in it's lowest level (74.02). As we shown in [19], the influence of friends reputation feature tends to be stable when it is higher than 10. The performance of the three recommenders is better because they profit the slight augmentation in the other four features, whereas the decline of friends reputation does not has a strong impact.
- Cluster 5 contains 5.16% of users. These users have the lowest number of ratings and items' popularity and the highest abnormality average among clusters. One of the qualities of the trust-aware recommenders is reducing the impact of cold start, because one trust expression can be more informative than many items ratings. This is obvious in this cluster, where the average of trust relationships is 6.7 and that of ratings is 4.47 (both are relatively low). Hence. the loss in CF performance is farther than that in trust-aware (0.21 versus 0.74). It is true also that other features play a role in this bad performance of CF, when users rate a small number of relatively unpopular items, finding neighbors in CF becomes complicated (same problem as cluster 3). Even when RS finds the similar users, the predicting ratings is weak because of the abnormal behaviour of the users in this cluster.
- Cluster number 6 is the biggest cluster with 25% of users, it contains users who tend to rate items, more than trusting other users (about 9 ratings and 0.08 trust relationships by user), only 251 users over 8324 in this cluster have trusted others. Nevertheless, they tend to trust very reputed users. Hence, CF is the best recommender for these users, it depasses slightly the performance of the hybrid recommender (0.6 versus 0.59), whereas the

trust recommender is far behind them, which can be explained by the lake of trust relations to the trust-aware recommender. The slight difference to the favour of CF over the hybrid system can be explained by the inutility of the information supplied to the hybrid system because trusting users who are trusted by everybody can be uninformative and disturbing to the hybrid system. The performance of the hybrid reommender on this cluster is the worst compared to other clusters.
– The last and the smallest cluster is cluster 7, with 1.62% of users. Users in this cluster have the highest average of ratings, they rate averagely popular items, thier abnormality average is close to the general average. They have a considerable number of trust relationships with a well reputed friends. All these values make this cluster the best quality over the 8 clusters, and the three recommenders achieve their best performance on this cluster.

4 Conclusion and future works

The main contribution of this paper is to detect different models of users' behavior in RS, and their impact on RS's performance. We have shown the relation between the performance fluctuation and the behavioral features of users. In some clusters, it was necessary to analyze more than one feature together to explain RS performance, for example clusters 3 and 5, these phenomena are not easy to detect and explain without the clustering phase.
We also referred to the usability of qualitative features in explaining the fluctuation. Rating unpopular items causes difficulties in finding neighbors, so recall problem to CF recommenders, trusting weakly reputed persons causes the same problem for TAR, being abnormal affects precision negatively in all recommenders. We think that regarding user behavior feature assist to build an adaptive recommender that agree with their mode of behavior within the social network.

References

1. R. Burke. 2007. Hybrid web recommender systems. In The adaptive web, Peter Brusilovsky, Alfred Kobsa, and Wolfgang Nejdl (Eds.). Lecture Notes In Computer Science, Vol. 4321. Springer-Verlag, Berlin, Heidelberg 377-408.
2. R. Burke, B. Mobasher, R. Zabicki, and R. Bhaumik. Identifying attack models for secure recommendation. In in Beyond Personalization: A Workshop on the Next Generation of Recommender Systems, 2005.
3. Harald Steck. 2011. Item popularity and recommendation accuracy. In Proceedings of the fifth ACM conference on Recommender systems (RecSys '11). ACM, New York, NY, USA, 125-132.
4. Paul Resnick and Hal R. Varian. 1997. Recommender systems. Commun. ACM 40, 3 (March 1997), 56-58.
5. Bradley N. Miller, Joseph A. Konstan, and John Riedl. 2004. PocketLens: Toward a personal recommender system. ACM Trans. Inf. Syst. 22, 3 (July 2004), 437-476.

6. P. Massa and B. Bhattacharjee. Using trust in recommender systems: An experimental analysis. In iTrust04, pages 221235, 2004.
7. Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. 1994. GroupLens: an open architecture for collaborative filtering of netnews. In Proceedings of the 1994 ACM conference on Computer supported cooperative work (CSCW '94). ACM, New York, NY, USA, 175-186
8. J. Golbeck and J. Hendler. FilmTrust: movie recommendations using trust in web-based social networks. 2006.
9. David Maltz and Kate Ehrlich. 1995. Pointing the way: active collaborative filtering. In Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '95), Irvin R. Katz, Robert Mack, Linn Marks, Mary Beth Rosson, and Jakob Nielsen (Eds.). ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 202-209.
10. Bamshad Mobasher, Robin Burke, Runa Bhaumik, and Chad Williams. 2007. Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. ACM Trans. Internet Technol. 7, 4, Article 23 (October 2007).
11. Shyong K. Lam and John Riedl. 2004. Shilling recommender systems for fun and profit. In Proceedings of the 13th international conference on World Wide Web (WWW '04). ACM, New York, NY, USA, 393-402.
12. Michael O'Mahony, Neil Hurley, Nicholas Kushmerick, and Gunol Silvestre. 2004. Collaborative recommendation: A robustness analysis. ACM Trans. Internet Technol. 4, 4 (November 2004), 344-377.
13. Jennifer Golbeck. Personalizing applications through integration of inferred trust values in semantic web-based social networks. In Semantic Network Analysis Workshop at the 4th International Semantic Web Conference, November 2005.
14. Ugur Kuter and Jennifer Golbeck. 2010. Using probabilistic confidence models for trust inference in Web-based social networks. ACM Trans. Internet Technol. 10, 2, Article 8 (June 2010), 23 pages.
15. Cai-Nicolas Ziegler and Georg Lausen. 2004. Spreading Activation Models for Trust Propagation. In Proceedings of the 2004 IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE'04) (EEE '04). IEEE Computer Society, Washington, DC, USA, 83-97.
16. Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. 2004. Evaluating collaborative filtering recommender systems. ACM Trans. Inf. Syst. 22, 1 (January 2004), 5-53.
17. P.Massa, P.Avesani. 2004. Trust-aware Collaborative Filtering for Recommender Systems. In Proc. of Federated Int. Conference On The Move to Meaningful Internet: CoopIS, DOA, ODBASE. pages 492-508.
18. C.Haydar, A.Boyer, A.Roussanaly. 2012. Hybridising collaborative filtering and trust-aware recommender systems. In WEBIST 2012 - Proceedings of the 8th International Conference on Web Information Systems and Technologies, Porto, Portugal, 18 - 21 April, 2012.
19. C.Haydar, A.Roussanaly, A.Boyer. 2012. Analyzing recommender system's performance fluctuations across users. In (CD-ARES 2012).
20. Alsabti, Khaled; Ranka, Sanjay; and Singh, Vineet, "An efficient k-means clustering algorithm" (1997). Electrical Engineering and Computer Science. Paper 43.
21. D.L. Davies and D.W. Bouldin, A cluster separation measure, IEEE Trans. Pattern Anal. Machine Intell. Vol. 1, pp. 224-227, 1979.