Recommending People to Follow Using Asymmetric Factor Models with Social Graphs

Tianle Ma[1], Yujiu Yang[1], Liangwei Wang[2], and Bo Yuan[1]

[1] Intelligent Computing Lab, Graduate School at Shenzhen, Tsinghua University
[2] Huawei Noah's Ark Lab, Huawei Technologies, Co., Ltd.
Shenzhen 518055, P.R. China
matianle1988@gmail.com, {yang.yujiu, yuanb}@sz.tsinghua.edu.cn
wangliangwei@huawei.com

Abstract. Traditional recommendation techniques often rely on the user-item rating matrix, which explicitly represents a user's preference among items. Recent studies on recommendations in the scenario of social networks still largely follow this principle. However, the challenge of recommending people to follow in social networks has yet to be studied thoroughly. In this paper, by using the utility instead of ratings and randomly sampling the negative cases in the recommendation log to create a balanced training dataset, we apply the popular matrix factorization techniques to predict whether a user will follow the person recommended or not. The asymmetric factor models are built with an extended item set incorporating the social graph information, which greatly improves the prediction accuracy. Other factors such as sequential patterns, CTR bias, and temporal dynamics are also exploited, which produce promising results on Task 1 of KDD Cup 2012.

Keywords: Social Recommender Systems, Matrix Factorization, People Recommendation, Social Graph, Asymmetric Factor Model

1 Introduction

Online social networking services have become tremendously popular in recent years, such as Facebook, Twitter, and Tencent-Weibo. These social media sites are generating huge amount of social data every day. For example, currently, there are more than 200 million registered users on Tencent-Weibo, generating 40 million messages each day. This scale benefits the Tencent-Weibo users but it can also flood users with huge volumes of information and hence puts them at the risk of information overload [1].

To cope with the issue of information overload, existing recommender systems generally follow three strategies: collaborative filtering, content-based recommendation, and hybrid recommendation [2]. Social recommender systems are recommender systems that target the social media domain. And recommending people to follow has become one of the hot topics in social recommender systems. In KDD Cup 2012, Task 1 [1] is a prediction task that involves predicting whether or not a user will follow an item that has been recommended to the user. In this paper, we

compared several approaches to recommending items, which can be persons, organizations, or groups, for users to follow. We found that, by incorporating social graph information into the asymmetric factor models (AFM), the recommendation accuracy of traditional matrix factorization techniques can be significantly improved. We also closely investigated issues such as how to incorporate sequence patterns, clickthrough rates (CTR) bias and temporal dynamics into our model, sampling schemes for the recommendation logs and training schemes for the factor models.

The rest of this paper is organized as follows. Section 2 introduces the related work and highlights the contributions of our work. The problem of recommending people to follow is formulated in Section 3 together with the details of AFM. Some analysis and discussion on related issues are presented in Section 4. Experimental results are given in Section 5 and this paper is concluded in Section 6.

2 Related Work

The motivation of recommender systems (RS) is to automatically suggest items to each user that he or she may find appealing (see [3] for an overview). The recommender has the task to predict the rating for user u on a non-rated item i or to generally recommend some items for the given user u based on the ratings that already exist [4].

As one of the most successful approaches to recommender systems, collaborative filtering approaches predict users' interests by mining user rating history data [5-8] and are most effective when users have expressed enough number of ratings. However, it deals poorly with the so called *cold start* problems.

Over the last two decades many CF algorithms have been proposed such as matrix factorization techniques, neighborhood-based approaches and restricted Boltzmann machine [9]. In general, there are three main categories of CF techniques [10]: memory-based, model-based, and hybrid CF algorithms that combine CF with other recommendation techniques. The memory-based CF [11] explores the user-item rating matrix and makes recommendations based on the ratings of item i by a set of users whose rating profiles are most similar to that of user u. While these approaches are easy to implement, their performance may be compromised when data are sparse. Hybrid CF algorithms, such as the content-boosted CF algorithm [12], are found helpful to address the sparsity problem. However, these hybrid approaches can result in increased complexity and expense for implementation [10].

In most traditional recommender systems, only the information in the user-item rating matrix is exploited for recommendations, ignoring completely the social relationships among users [9-11, 13-15]. The results of experiments in [16] and other similar work have confirmed that social networks can provide an independent source of information, which can be exploited to improve the quality of recommendations.

Recently, memory-based approaches have been proposed for recommendation in social rating network [17, 18]. These methods use the transitivity of trust and propagate trust to indirect neighbors in social network. Model-based approaches have also been applied to social rating networks [19-21] by exploiting matrix factorization techniques to learn latent features for users and items from observed ratings. However,

due to the sensitive nature of social data, most of the related research studies are only based on the Epinions dataset [18-21] or dataset crawled from Flixster.com.

In this paper, we focused on the people recommendation prediction task. More specifically, we investigated Task 1 in KDD Cup 2012. In our problem, there is an implicit feedback in terms of whether a person follows the recommended item or not. The difference between the implicit feedback and the explicit feedback is that explicit feedbacks such as ratings involve more initiatives since a user only needs to click the 'accept' button to follow someone. In contrast, a person has to take some initiative to rate a movie that has been watched.

In summary, the main contributions of this paper are as follows:

- By adopting the more general concept of utility [3] instead of ratings and randomly sampling the negative cases in the recommendation log to make a balanced training dataset, we successfully apply the popular matrix factorization techniques to predict whether a user will follow an item recommended.
- To address the *cold start* issue, which is serious in our task as half of the users in the test dataset are not in the training dataset, we apply the asymmetrical factor models representing a user by items that have been accepted and followed.
- We incorporate social graphs into the asymmetric factor models with an extended item set to greatly improve the prediction accuracy.
- Factors such as sequential patterns, clickthrough rate bias, and the temporal aspects are also taken into consideration to further improve the performance.

3 People Recommendation with AFM and Social Graphs

Recommender systems using social network information are often aimed at optimizing RMSE on observed ratings [17-21]. Compared to neighborhood [22, 23] and random walk [18] methods, matrix factorization methods were found to be the most accurate model in the context of social network information [19-21]. As there are no ratings in our problem setting, we cannot directly apply the above recommendation approaches and some modifications are necessary.

3.1 Preliminaries

To investigate the task of recommending people to follow, we focused on the recent competition, Task 1 in KDD Cup 2012 [1]. In this task, the recommendation log is included in training datasets (*rec_train_log.txt*) and test datasets (*rec_test_log.txt*). The format of both files is:*(UserId)\t(ItemId)\t(Result)\t(Unix-timestamp)*.

The values of *'Result'* field are 1 or -1, where 1 represents the user *UserId* accepts the recommendation of item *ItemId* and follows it, and -1 represents the user rejects the recommended item. The true values of *'Result'* field are provided in the *rec_train_log.txt*, whereas the true values of *'Result'* field in *rec_test_log.txt* are withheld which need to be predicted.

In addition to the recommendation log dataset, the user sns dataset contains each user's following history. In our paper, we focused on the CF methods by exploiting the recommendation log and social graph information.

Let $U = \{u_1, \cdots, u_n\}$ and $I = \{i_1, \cdots, i_m\}$ be the sets of all users and all possible items that can be recommended, respectively. The space of U is large with more than two million users. The item set contains 6095 items, which will be extended to around twenty thousand in our algorithm. Let r be a utility function [3] that measures the usefulness of item i to user u ($r: U \times I \rightarrow R$) where R is a totally ordered set (e.g., non-negative integers or real numbers within a certain range). For each user $u \in U$, we want to choose such an item $i \in I$ that maximizes the user's utility:

$$i \in \underset{i \in I}{\operatorname{argmax}} r_{ui,} \quad u \in U$$

Here, utility can be an arbitrary function that indicates how a particular user likes this item.

3.2 Basic Matrix Factorization Models

Matrix factorization models [15] map users and items to a joint latent factor space of dimensionality f, such that user-item interactions are modeled as inner products. Each item i is associated with a vector $q_i \in \mathbb{R}^f$, and each user is associated with a vector $p_u \in \mathbb{R}^f$. The resulting dot product, $q_i^T p_u$, captures the interaction between user u and item i – the user's overall interest in the item's characteristics.

In our case, the result value is 1 if a user followed an item and -1 otherwise. Although there is no rating, this result value can reflect a user's overall interest in the item's characteristics or the utility of the item to the user. In our method, we used 0 to replace -1, and treated these values as ratings. Then we calculated each user's ratings for all the items recommended in the test dataset and ranked them. The top 3 items were outputted as the final recommendation results.

As defined earlier, r_{ui} is the utility of item i to the user u, which represents the user's interest in the item, and its estimate $\tilde{r}_{ui}$ is defined as follows:

$$\tilde{r}_{ui} = q_i^T p_u. \quad \textbf{(1)}$$

To learn the factor vectors (p_u and q_i), we minimizes the regularized squared error on the training set [15] :

$$\min_{q^*, p^*} \sum_{(u,i) \in \kappa} (r_{ui} - q_i^T p_u)^2 + \lambda(\|q_i\|^2 + \|p_u\|^2). \quad \textbf{(2)}$$

Here κ is the set of (u, i) pairs in the training set (the recommendation log of users' following history). The constant λ controls the extent of regularization and is usually determined by cross-validation.

Equation (1) tries to capture the interactions between users and items that produce different utility values. However, large portions of utility values may be due to effects associated with either users or items, known as biases and intercepts, instead of any meaningful interactions.

Thus, we added a first-order approximation of the bias into the utility r_{ui} [15]:

$$\tilde{r}_{ui} = q_i^T p_u + b_i + b_u. \quad \textbf{(3)}$$

In equation (2), the utility value is broken into three components: item bias, user bias, and user-item interaction, allowing each component to explain only the part of a signal relevant to it. The system learns by minimizing the squared error function [15]:

$$\min_{q^*, p^*} \sum_{(u,i)\in\kappa} (r_{ui} - b_u - b_i - q_i^T p_u)^2 + \lambda(\|q_i\|^2 + \|p_u\|^2 + b_u^2 + b_i^2). \quad (4)$$

3.3 Asymmetric Factor Models

While a user is represented by the feature p_u in the plain SVD model, the asymmetric factor model [5] represents a user by the items accepted or followed. In other words, AFM only parameterizes item features.

In our problem, the size of the provided item dataset (around 6K items) is too small compared to the number of users (around 2 million). As a result, we extended the item dataset to around 20K items using the social graph information. There are two datasets containing social graph information: *user_sns.txt* and *user_action.txt* and users with the most followers were added into the item dataset.

Given $N(u)$ as the set of items which are followed by user u, a virtual user feature p_u is given by: $p_u = |N(u)|^{-1/2} \sum_{i\in N(u)} q_i$ [5, 9]. This representation offers several benefits such as integrating new data and new user without retraining the whole model [24]. In AFM, the estimated matching value of user u and item i is:

$$\tilde{r}_{ui} = q_i^T \cdot (|N(u)|^{-1/2} \sum_{j\in N(u)} q_j) + b_i + b_u. \quad (5)$$

3.4 Sampling Scheme

It is not unusual that many recommendations were not accepted by the user and the corresponding results are negative in the training dataset (*rec_log_train.txt*), resulting in an imbalanced dataset. Meanwhile, a negative result itself is not a strong indication that a user did not want to accept the recommendation and had no interest in the item at all. Instead, chances are that the user may follow the item the next time when the item is recommended to him/her. As a result, the original training dataset was re-sampled before training our asymmetric factor models. All records with positive results were retained while the negative cases were randomly down sampled to be roughly the same size as the positive cases.

4 Performance Considerations

In addition to the asymmetric factor model, better performance can be possibly achieved by taking more factors into consideration, such as sequential patterns, CTR bias, and temporal dynamics.

4.1 Sequence Patterns

The recommendation log is a time series dataset and there are some important sequential features. As mentioned above, there were only two different utility values

in the training dataset that can affect the mean average precision (MAP). Using sequential information, we proposed to adjust the utility values as follows:

- If the result field is 1, the utility value was also set to 1;
- If the result field is -1 and the user did not accept any other items in the same time, the utility value was set to a small value between 0 and 1 (0.2 in this paper);
- If the result field is -1 and the user did accept one or more items in the same time, the utility value was set to 0.

The above scheme is based on a common sense: simply knowing that a user did not follow the item recommended in the log does not guarantee that he or she had no interest in the item. Consequently, it is inappropriate to set the corresponding utility value to 0. Instead, a small value such as 0.2 was chosen to represent this uncertainty. However, if a user accepted some other items in a very short time period, the possibility that the user might be really of no interest in the item could be reasonably high. As a result, we set the utility value of this rejected item to 0.

4.2 CTR Bias

The clickthrough rate (CTR) in our context was defined as the percentage of acceptance of different items of all the recommendations among certain user groups. First, we divided all users in the dataset into disjoint groups according to their age and gender. We found that different groups had different CTRs denoted as c_{ui}.

To emphasize this type of group characteristics, we add a coefficient to each user, denoted by α_u. The bias of the CTR term is defined as follows:

$$b_{ui}^{ctr} = \alpha_u \cdot c_{ui} \quad . \tag{6}$$

In equation (6), each c_{ui} was pre-computed while the coefficients α_u were learned along with other parameters using stochastic gradient descent training scheme.

The estimate of the utility value between user u and item i becomes:

$$\tilde{r}_{ui} = q_i^T \cdot (|N(u)|^{-1/2} \sum_{j \in N(u)} q_j) + b_i + b_u + \alpha_u \cdot c_{ui} \quad . \tag{7}$$

4.3 Temporal Dynamics

So far, the presented models have not considered the temporal dynamics. However, one factor with time-drifting nature must be considered: a user's inclination to accept recommendations vary across different time periods of a week. According to the statistics that we have collected, most users were more likely to accept recommendations during weekends and nonworking time. A possible explanation may be that most people usually do not have enough time to spend on social networks during working hours.

As a result, we uniformly divided a week into 168 time slots (one hour per slot). In each slot, a user may have a general inclination to accept or reject a recommended item. Let each user has a new hidden layer relating to time slots $p_{ut}(t_{ui})$. Similarly, each time slot has its own feature $q_t(t_{ui})$. Both $p_{ut}(t_{ui})$ and $q_t(t_{ui})$ have the same number of hidden layers. As a result, this temporal effect can be modeled as follows:

$$b_{ut}(t_{ui}) = p_{ut}(t_{ui}) \cdot q_t(t_{ui}) \quad . \tag{8}$$

The new estimate of the utility value of item i to user u becomes:

$$\tilde{r}_{ui} = q_i^T \cdot (|N(u)|^{-1/2} \sum_{j \in N(u)} q_j) + b_i + b_u + \ b_{ut}(t_{ui}). \tag{9}$$

4.4 The Combination Model

Taking both the CTR bias and the time-drifting effect into consideration, a more accurate estimate of the matching value of user u and item i is:

$$\tilde{r}_{ui} = q_i^T \cdot (|N(u)|^{-1/2} \sum_{j \in N(u)} q_j) + b_i + b_u + \alpha_u \cdot c_{ui} + p_{ut}(t_{ui}) \cdot q_t(t_{ui}) \quad . \tag{10}$$

The system learns by minimizing the regularized squared error function:

$$\min_{q^*, b^*, \alpha^*, q_t^*, p_{ut}^*} \sum_{(u,i) \in \kappa} (r_{ui} - \tilde{r}_{ui})^2 + \lambda(\|q_i\|^2 + \|\alpha_u\|^2 + b_u^2 + b_i^2 + \sum_{j \in N(u)} \|q_j\|^2) + \lambda_t(\|q_t(t_{ui})\|^2 + \|p_{ut}(t_{ui})\|^2). \tag{11}$$

The $\tilde{r}_{ui}$ in equation (11) is the same as in equation (10). Since the time-drifting term is different from other terms, we used a different regulation parameter λ_t.

5 Experiments

We conducted a series of experiments on the datasets of Task 1 in KDD Cup 2012 (see [1] for details). In this section, we report our experimental results and compare the results with different methods.

5.1 Experimental Setup

In the provided datasets, there are 2,320,895 users with 50,655,143 following relations. To evaluate the performance of our asymmetrical factor models, we considered several other algorithms:

- **Common-Follow Algorithm:** This algorithm is based on the intuition that a user will follow an item if his/her friends also follow it. Actually, many social network sites such as Facebook use similar methods to recommend people to connect with. In our work, when an item was recommended to a user, we calculated the percentage of his/her followees who also followed the same item. For example, if the percentage of user $u's$ followees who followed item i is p, the utility of i to u was defined as $r_{ui} = \log_2(p+1)$.
- **Common-Retweet Algorithm:** This algorithm is similar to the Common-Follow Algorithm. The only difference is that we calculated the percentage of retweet times of item i from the followees of the user u. For example, if the percentage of retweet times of item i from the followees of the user u is p, then the utility of i to u was defined as $r_{ui} = \log_2(p+1)$.

- **Hot-Degree Algorithm:** We calculated the click rates of each of item, and recommended the hottest items to the user.

5.2 Results and Analysis

In Table 1, as the public and private leaderboards have a temporal order, we can see that the results were always better for public leaderboard than private leaderboard since it is temporally more close to the training dataset.

The performance of the Common-Follow Algorithm and the Common-Retweet Algorithm was very poor. In the meantime, the Hot-Degreee Algorithm performed relatively well, although it did not handle the interaction between users and items. This is acceptable because most of us tend to follow the general trend.

The results of the basic SVD techniques, with or without the user and item bias, were not good compared to the Hot-Degree Algorithm, although more elaborated matrix factorization techniques may improve the results [15].

By contrast, the asymmetrical factor models with an extended item set significantly improved the results. The reason is that, by extending the item set, we incorporated additional useful information in the social graph into the models. The adoption of sequential patterns, temporal dynamics, and CTR bias also contributed to the improvement of the results.

Note that the model parameters, such as the number of hidden layers, the regulation parameters, the iteration steps, and the number of iterations can also affect the final results.

Table 1. Results of Different Recommendation Algorithms

Algorithms	Public Leaderboard	Private Leaderboard
Common-Follow	0.23611	0.23541
Common-Retweet	0.22182	0.21882
Hot-Degree	0.33383	0.32831
Basic SVD	0.30548	0.28697
SVD + CTR bias	0.31446	0.29664
AFM with extended item set	0.36983	0.36016
AFM + sequential patterns	0.37009	0.36035
AFM + sequential patterns + Temporal Dynamics	0.37028	0.36055
AFM + sequential patterns + CTR bias	0.37076	0.36093
AFM + sequential patterns + Temporal Dynamics + CTR bias	0.37132	0.36143

6 Conclusions

In this paper, we proposed to incorporate social graph information into asymmetrical factor models by extending the item dataset, in order to make accurate predictions on whether a user will follow an item or not. One of the major differences between traditional recommender systems and social recommender systems is that there are no ratings in social networks. To cope with this issue, we used the utility of an item to a user instead of ratings and developed novel latent factor models.

Experimental results show that social graph information can significantly improve the performance of recommender systems, compared to a number of standard recommendation algorithms. In the meantime, additional factors were also taken into consideration, such as sequential patterns, CTR bias, and temporal dynamics, which further boosted the predication accuracy.

As to future work, in our paper, the model was optimized in terms of RMSE. However, the evaluation metric in Task 1 of KDD Cup 2012 is MAP. Although RMSE and MAP are highly positively correlated, we can try to adapt our model to optimize MAP directly in the future. Also, we can build a directed weighted social graph that can be used to build a more elaborated model.

Acknowledgments. This work was supported by the National Natural Science Foundation of China (No. 60905030) and Upgrading Plan Project of Shenzhen Key Laboratory (No. CXB201005250038A). The authors are also grateful to several colleagues who have provided constructive feedbacks to our work. Besides, we would like to gratefully acknowledge the organizers of KDD Cup 2012 as well as Tencent Inc. for making the datasets available.

References

1. Niu, Y., Wang, Y., Sun, G., Yue, A., Dalessandro, B., Perlich, C., Hamner, B.: The Tencent Dataset and KDD-Cup'12. In KDD-Cup Workshop, (2012)
2. Jannach, D., Zanker, M., Felfernig, A., Friedrich, G.: Recommender Systems: An Introduction. Cambridge University Press (2010)
3. Adomavicius, G., Tuzhilin, A.: Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-art and Possible Extensions. Knowledge and Data Engineering, IEEE Transactions on 17 (6), 734-749 (2005)
4. Yang, X., Steck, H., Guo, Y., Liu, Y.: On Top-k Recommendation using Social Networks. In: Proceedings of 6th ACM Conference on Recommender Systems. ACM, Dublin, Ireland (2012)
5. Paterek, A.: Improving Regularized Singular Value Decomposition for Collaborative Filtering. In: Proceedings of KDD Cup and Workshop 2007, pp. 5-8. ACM, San Jose, California (2007)
6. Keshavan, R.H., Montanari, A., Oh, S.: Matrix Completion From Noisy Entries. The Journal of Machine Learning Research 11, 2057-2078 (2010)
7. Pan, R., Zhou, Y., Cao, B., Liu, N.N., Lukose, R., Scholz, M., Yang, Q.: One-class Collaborative Filtering. In: Proceedings of the 8th IEEE International Conference on Data Mining, pp. 502-511. IEEE, Pisa, Italy (2008)

8. Srebro, N., Jaakkola, T.: Weighted Low-rank Approximations. In: Proceedings of the 20th International Conference on Machine Learning, pp. 720-727. AAAI Press, Washington, DC, USA (2003)
9. Jahrer, M., Töscher, A., Legenstein, R.: Combining Predictions for Accurate Recommender Systems. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 693-702. ACM, Washington, DC, USA (2010)
10. Su, X., Khoshgoftaar, T.M.: A Survey of Collaborative Filtering Techniques. Advances in Artificial Intelligence 2009, (2009)
11. Goldberg, D., Nichols, D., Oki, B.M., Terry, D.: Using Collaborative Filtering to Weave an Information Tapestry. Commun Acm 35 (12), 61-70 (1992)
12. Melville, P., Mooney, R.J., Nagarajan, R.: Content-boosted Collaborative Filtering for Improved Recommendations. In: Proceedings of the 18th National Conference on Artificial Intelligence and 14th Conference on Innovative Application of Artificial Intelligence, pp. 187-192. AAAI Press; MIT Press; Edmonton, Alberta, Canada (2002)
13. Koren, Y.: Collaborative Filtering with Temporal Dynamics. Commun Acm 53 (4), 89-97 (2010)
14. Koren, Y.: Factorization Meets the Neighborhood: a Multifaceted Collaborative Filtering Model. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 426-434. ACM, Las Vegas, Nevada, USA (2008)
15. Koren, Y., Bell, R., Volinsky, C.: Matrix Factorization Techniques for Recommender Systems. IEEE Computer 42 (8), 30-37 (2009)
16. Singla, P., Richardson, M.: Yes, There is a Correlation: - From Social Networks to Personal Behavior on the Web. In: Proceedings of the 17th International Conference on World Wide Web, pp. 655-664. ACM, Beijing, China (2008)
17. Massa, P., Avesani, P.: Trust-aware Recommender Systems. In: Proceedings of the 2007 ACM conference on Recommender systems, pp. 17-24. ACM, Minneapolis, MN, USA (2007)
18. Jamali, M., Ester, M.: TrustWalker: A Random Walk Model for Combining Trust-based and Item-based Recommendation. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 397-406. ACM, Paris, France (2009)
19. Ma, H., King, I., Lyu, M.R.: Learning to Recommend with Social Trust Ensemble. In: Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 203-210. ACM, Boston, MA, USA (2009)
20. Jamali, M., Ester, M.: A Matrix Factorization Technique with Trust Propagation for Recommendation in Social Networks. In: Proceedings of the 2010 ACM Conference on Recommender Systems, pp. 135-142. ACM, Barcelona, Spain (2010)
21. Ma, H., Yang, H., Lyu, M.R., King, I.: Sorec: Social Recommendation Using Probabilistic Matrix Factorization. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, pp. 931-940. ACM, Napa Valley, California, USA (2008)
22. Golbeck, J.A.: Computing and Applying Trust in Web-based Social Networks. University of Maryland at College Park College Park, MD, USA (2005)
23. Jamali, M., Ester, M.: Using a Trust Network to Improve Top-N Recommendation. In: Proceedings of the 2009 ACM Conference on Recommender Systems, pp. 181-188. ACM, New York, NY, USA (2009)
24. Koren, Y.: Factor in the Neighbors: Scalable and Accurate Collaborative Filtering. ACM Transactions on Knowledge Discovery from Data 4 (1), 1-24 (2010)