An Improved Similarity and Time Age Weight Approach Combining K-nearest Neighbor and Latent Factor Model

Guangmin Sun[1,a,*], **Chenyan Yu**[1,b], **Xiao Liang**[1,c]

[1]Faculty of Information Technology, Beijing University of Technology, Beijing, China
[a] gmsun@bjut.edu.cn, [b] 675917970@qq.com, [c] 490040666@qq.com

Keyword: Latent factor model, K-nearest neighbor model, Information entropy, Interest change, Age group.

Abstract. Recommender systems can be used to provide users with personalized recommendation information. They always rely on collaborating filtering (CF), since it is an effective way to establish connections between products and users. Most of the CF methods are based on neighborhood models, which calculate the similarities between users and products. Moreover, some improvements that model neighborhood relations by minimizing a cost function are made to predict a better result, since latent factor models can provide more aspects of the data, and offer more accurate results than neighborhood models. Past models were limited by using simple cosine similarity only, and they did not consider that the change of interests. Moreover, age groups may have a significant effect on the final results. In this paper, to solve the problem, a new comprehensive item similarity based on information entropy is proposed. We introduce a time and age weight to alleviate the influence on the interest change of different age groups. Some experiments were made to test the methods on the Movielens dataset, and encouraging results were obtained.

1. Introduction

Due to the development of the internet and e-commence, people are now inundated with choice. it is difficult for people to find what they want efficiently[1]. This emphasizes the significant role of the recommender system, which have received more attentions ofscholars[2]. By using the recommender system, people may have much more personalized options.

Generally, Collaborative filtering(CF)[3] is widely used in today's recommender system, which is mainly divided into three categories: user-based algorithm, item-based algorithm and model-based algorithm, and there are two kinds of models[4] called neighborhood model and latent factor model. Neighborhood model[5] calculates the similarity between the users or the items, the neighbor relationship help to value the preference of a user for an item based on ratings of similar items by the same user. Currently, the k-nearest neighbor model is widely used. Latent factor model[6], such as singular value decomposition (SVD)[7], transforms the users and items' characteristics into the same latent factor space, the main purpose of the model is to predict ratings by producing products and users on factors automatically, therefore the latent relationship between user and item could be emerged. Time weight[8,9] and age weight was used to analyze the interest change of different age groups, because users' recent ratings may reflect their interests effectively, and age factor may have a positive influence on the result. Information entropy[10] was quoted into similarity, so that the similarity can contain more kinds of information. The traditional latent factor model and k-nearest neighbor model do not consider that the similarity can contain more dimensional information, such as item properties, and time and age groups may have an influence on the result. As a consequence, finding an efficient solution is also very important.

In this paper, we make improvements on the k-nearest neighbor model and latent factor model by adding item similarity combined with information entropy. Therefore we can derive the previous similarity more accurately. Furthermore, we focus on the time and age weight to alleviate the impact on the interest changes of different age people. Experiments made on the Movielens dataset have an encouraging result.

2. Latent factor model

2.1. Baseline estimation

In the collaborative filtering system, user would only rate for a small part of the items, there by forming a very sparse rating matrix. Matrix factorization can be a good solution to this problem, the mathematics SVD has been introduced into the recommendation algorithm in early times, while it requires a lot of storage space, so that the method is not easy to accomplish. Until the Netflix Prize, funk announced the latent factor model to solve the problem.

From the perspective of matrix factorization, the rating matrix is decomposed into a form of two-dimensional matrix multiplication, one contains the 'user factors', the other contains the 'item factors':

$$\hat{R} = P^T Q \tag{1}$$

P represents an $f \times m$ matrix, *Q* represents an $f \times n$ matrix. The predicted rating of user *u* towards item *i* would be $\hat{r}_{ui}$:

$$\hat{r}_{ui} = \sum_{f}^{F} p_{uf} q_{if} \tag{2}$$

F is the feature number. Each user is associated with a vector *p*, and each item is associated with a vector *q*.

While some users exhibit a tendency to rate higher scores than other users, as for some items, they might receive lower scores than the similar products. As a result, it is unreasonable to indicate the predicted rating by the multiplication of vector *p* and vector *q*, so the baseline estimation is proposed to account for the results. The equation which extends baseline is as follow:

$$\hat{r}_{ui} = \mu + b_u + b_i + p_u{}^T q_i \tag{3}$$

The overall average rating is μ, the b_u and b_i represent the deviations of user *u* and item *i*, the function which minimizes the squares error is as follow:

$$\min_{p.q.b.} \sum_{u,i \in T} \left(r_{ui} - \mu - b_u - b_i - p_u^T q_i \right)^2 + \lambda \left(\| p_u \|^2 + \| q_i \|^2 + b_u^2 + b_i^2 \right) \tag{4}$$

The first part helps to find b_u s and b_i s to fit the given ratings, and the second part of the function is to avoid over fitting by penalizing the parameters.

2.2 Stochastic gradient descent

We utilize the Stochastic gradient descent[11] to optimize the function above, there are 4 parameters in the function, the four parameters need to take partial derivative respectively, advancing along the direction of the fastest decline. The recurrence formulas are as follow:

$$\begin{cases} e_{ui} = r_{ui} - \mu - b_u - b_i - p_u{}^T q_i \\ p_u = p_u + \Upsilon \cdot (e_{ui} \cdot p_u - \lambda \cdot q_i) \\ q_i = q_i + \Upsilon \cdot (e_{ui} \cdot q_i - \lambda \cdot p_u) \\ b_u = b_u + \Upsilon \cdot (e_{ui} - \lambda \cdot b_u) \\ b_i = b_i + \Upsilon \cdot (e_{ui} - \lambda \cdot b_i) \end{cases} \tag{5}$$

where Υ indicates the learning rate.

3. Neighborhood model

3.1 Similarity

The core of the k-nearest neighbor model is to calculate the similarity, while the cosine similarity is the most common used:

$$sim_{\cos}(i,j) = \frac{\vec{i} \cdot \vec{j}}{\| \vec{i} \|^2 * \| \vec{j} \|^2} \tag{6}$$

The cosine similarity mainly rely on the users rating for item, when the rating matrix suffers from the sparse problem, it is always inaccurate to acquire the result of the similarity, which may lead to the decline of the accuracy, so we propose a comprehensive item similarity which based on the item property to find the solution to the sparse problem. Supposing that there are n items, $I_1, I_2 \cdots\cdots I_n$, and each item contains s properties, we can get the item property sheet as follow:

Table.1. The properties of each item

item/property	a_1	a_2	$\cdots$	a_s
I_1	1	0	$\cdots$	1
I_2	1	1	$\cdots$	0
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
I_i	1	0	$\cdots$	1
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
I_n	0	1	$\cdots$	1

The easiest way to calculate the item property similarity is using the Jaccard index, the equation is as follow:

$$sim_{jac} = \frac{|A(i) \cap A(j)|}{|A(i) \cup A(j)|} \tag{7}$$

$|A(i) \cap A(j)|$ represents the number of co-owned property, $|A(i) \cup A(j)|$ represents the total number of all the properties which the two item contain. While this method does not consider that the influence of different property are not the same, for example, most movies have a property of love, as a result, the property of love have a weak influence on all the properties. In order to consider the difference between all the properties, we propose a method which is weighted by the property's information entropy. The equation of the information entropy is as follow:

$$H(X) = \sum_{i=1}^{n} -p_i \times \log_2(p_i) \tag{8}$$

X represents the random variable, *n* represents kinds of different values of *X*, *p* represents the probability of *i*. For each property a_m, the value could only be 1 or 0, it indicates whether the item contain this kind of property. Supposing that the probability of property a_m will be:

$$p(a_m) = \frac{k}{|I|} \tag{9}$$

K represents the occurrence number of the property a_m, *I* represents the total number of all the items. The information entropy of property a_m is as follow:

$$H(a_m) = -p(a_m) \times \log_2(p(a_m)) - (1 - p(a_m)) \times \log_2(1 - p(a_m)) \tag{10}$$

The Jaccard index is weighted by the property's information entropy:

$$\text{sim}_{JH}(i, j) = \frac{\sum_{a_n \in A(i) \cap A(j)} H(a_n)}{\sum_{a_m \in A(i) \cup A(j)} H(a_m)} \tag{11}$$

$A(i)$ represents a set of the property, $A(i) \cup A(j)$ represents the interaction of item *i* and item *j*'s property, $A(i) \cup A(j)$ represents the union of item *i* and item *j*'s property, we can combine these two similarity above linearly:

$$sim_{up}(i, j) = \alpha \times sim_{\cos}(i, j) + (1 - \alpha) \times \text{sim}_{JH}(i, j) \tag{12}$$

3.2 The varying weight

People's interest are sensitive to time, and recent data contributes to the recommendation process, we think that interest change is a form of information forgetting, so we propose an time-weight to reduce the influence of interest change. The time-weight is defined as:

$$W_t(u, i) = e^{-a \times \frac{t_{rating} - t_{\min}}{t_{\max} - t_{\min}}} \tag{13}$$

$t_{\max} - t_{\min}$ is the time span of a user, the span indicates the time from the first rating time to the last rating time. $t_{rating} - t_{\min}$ is the time span from current rating time (the time when user *u* rates item *i*) to last rating time, $W_t(\mathrm{u,i})$ is an exponential decreasing function based on the interest change, the function is on the basis of Ebbinghaus curve, which can improve the accuracy of the algorithm. But it doesn't consider the interest change of different age groups. Usually the teenagers' interest fluctuates much, however, the interest of middle-aged people doesn't change very often, and the interest of elders remains stable. So we can divide all the users into different groups to set a slower decay rate of interest for the middle age and elder people，the improvement of equation (13) is as follow:

$$W_{t-a}(\mathrm{u,i}) = \mathrm{e}^{-\frac{1}{1+\beta\times agegroup}\times\frac{t_{rating}-t_{\min}}{t_{\max}-t_{\min}}} \tag{14}$$

Agegroup is a variable that divide people into different age groups. If the age of people ranges from 1 to 18, *agegroup* equals to 1, if the age of people ranges from 19 to 55, *agegroup* equals to 2, if people's age is above 55, the *agegroup* equals to 3. β is an attenuation, we can set an appropriate variable for it.

3.3 K-nearest neighbor model

In order to predict the r_{ui}--the rating by user *u* for item *i*, we select *k* items from the similarity matrix which are the most similar to *i*. the result of the *k* neighborhood model works as a correction of latent factor model to adjust the baseline estimates:

$$\hat{r}_{ui} = b_{ui} + \frac{\sum_{j\in S^k}(r_{u,j} - b_{u,j})sim(\mathrm{i,j})}{\sum_{j\in S^k} sim(\mathrm{i,j})} \tag{15}$$

b_{ui} represents the predicted rating by the latent factor model, S^k represent a set of *k* nearest neighbor. By updating the new similarity and the time-age weight into the model, the equation will be:

$$\hat{r}_{ui} = b_{ui} + \frac{\sum_{j\in S^k}\mathrm{W}_{t-a}(\mathrm{u,j})(r_{u,j} - b_{u,j})\,\mathrm{sim}_{up}(\mathrm{i,j})}{\sum_{j\in S^k} sim_{up}(\mathrm{i,j})} \tag{16}$$

In this algorithm more information is utilized to make the model fitting reality.

4. Experimental results

4.1 Dataset

Movielens dataset has been widely used in the recommender system area, we use the 100k dataset which contains 100,000 ratings for 1682 movies by 943 users, in the dataset, each user rates at least 50 movies, some information such as movie genres, users' age, the rating time are also included in the dataset, which make the dataset rich enough to evaluate the algorithm.

The dataset is divided into 2 parts. 80% of the dataset is split for training purpose, 20% of the dataset is split for test purpose.

The evaluation metric used in our experiments is the RMSE, which can be computed as:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(p_i - q_i)^2}{N}} \tag{17}$$

4.2 The proposed recommender algorithm

The experiment aims to compare the result of KNN-ETA-SVD in predicting the rates. Among the algorithm, there are 5 important parameters, such as number of hidden features *F*, learning rate γ, regularization parameter λ, numbers of neighbors *k*, weight factor α. A flowchart is shown as blow:

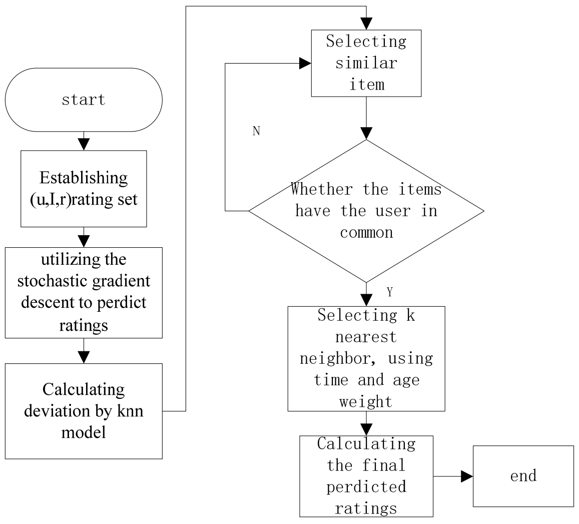
start
Establishing (u,I,r)rating set
utilizing the stochastic gradient descent to perdict ratings
Calculating deviation by knn model
Selecting similar item
N
Whether the items have the user in common
Y
Selecting k nearest neighbor, using time and age weight
Calculating the final perdicted ratings
end

Figure 1. The flowchart of KNN-ETA-SVD approach.

4.3 Results

From the experiment, we know that learning rate, regularization parameter hidden feature have a great influence on the algorithm performance. Firstly we should confirm the learning rate, so other parameters will be fixed.

In the experiment, *F*=40, λ =0.007, α =0.9,*k*=100 and we make learning rate γ equal to 0.003,0.005,0.007, 0.009, 0.011 respectively. When the parameter equals to 0.007, we can get the lowest RMSE result.

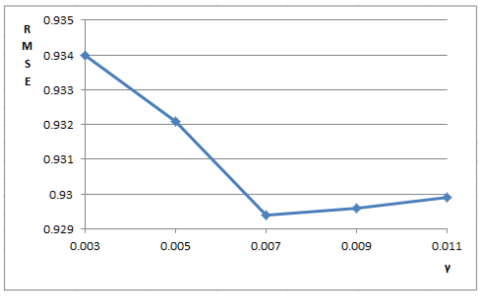

Figure 2. The influence of the learning rate γ.

Fig.3 shows the result influenced by regularization parameter. When the regularization parameter λ equals different values, other parameters remain stable. We can learn that the regularization parameter λ helps to alleviate the problem of over fitting, resulting in a better RMSE. But the regularization parameter cannot value too high.

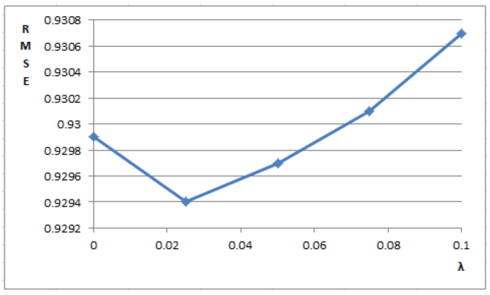

Figure 3. The influence of the regularization parameter λ.

Furthermore, weight factor α can have a positive impact on the RMSE. We assume that F=40, Υ=0.007, λ=0.025, k=100 and α equal to 0, 0.1, 0.2…1 respectively, using the complete data set, Fig.3 reflect the RMSE of the KNN-ETA-LMF in different weight factor, when α=0.9, we can get the lowest RMSE. So we confirm the weight number to be 0.9.

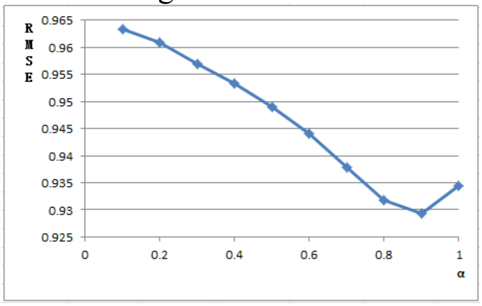

Figure 4. The influence of the weight parameter α.

And then we confirm the influence of neighbor number on the algorithm, we assume that α=0.9, Υ=0.007, λ=0.025, F=40, we make 20 iterations in the train set, we can see that the accuracy of the algorithm increase with the increase of neighbor numbers in Fig.5.

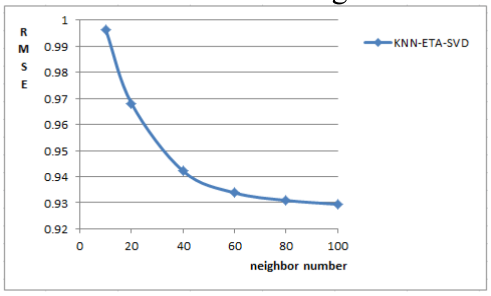

Figure 5. The RMSE with the change of neighbor number k.

Considering of the increase of hidden factor F, we respectively compare RMSE values from the algorithm in the paper (KNN-ETA-LMF), latent factor model improved by the neighborhood model (KNN-ETA-LMF) and traditional latent factor model (LMF), the result is shown in Fig.6.

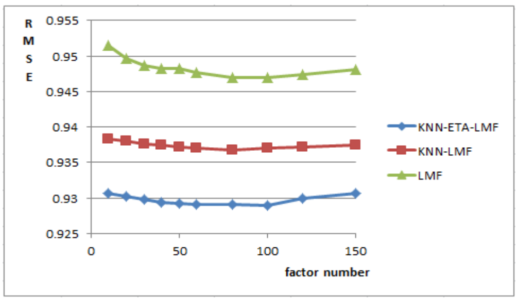

Figure 6. RMSE comparison of three different recommendation algorithms.

Fig.6 indicates that the KNN-ETA-LMF algorithm can reach a better accuracy than the KNN-LMF algorithm and LMF algorithm. In the experiment, $\alpha=0.9$, $k=100$, $\Upsilon=0.007$, $\lambda=0.025$, we make 20 iterations in the train set. When the number of the hidden factor F is 100, RMSE reaches the best value 0.929. In conclusion, the latent factor model improved by the new KNN model brings an improvement over the traditional algorithm. The efficiency of the recommendation system is hence improved.

5. Conclusions

As the core technology of the E-commerce, recommender system has a positive effect. In the paper, we improve accuracy of the ordinary latent factor model with k-nearest neighbor model. Firstly, we propose a new comprehensive item similarity based on information entropy, which can improve the performance of the algorithm, and it is beneficial to solve the data sparse problem. Secondly, we validated that a time weight can have benefits to alleviate the influence on the interest change of people, thereby improving the efficiency. We also propose an age weight to divide people into different age groups; therefore the different interest of age groups can be distinguished. Comparing to the LMF algorithm and LMF-KNN algorithm, the KNN-ETA-LMF algorithm achieves an improvement of predict accuracy. The RMSE can reach up to 0.929, which is better than other traditional algorithms with the same parameters.

References

[1] Bawden D, Robinson L. The dark side of information: overloadanxiety and other paradoxesand pathologies[J]. Journal of Information Science, 2009, 35(2): 180-191.
[2] G Adomavicius, A Tuzhilin.Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions [J]. IEEE transactions on knowledge and data engineering, 2005 - ieeexplore.ieee.org, 734-749
[3] Su X, Khoshgoftaar, TMA survey of collaborative filtering techniques[J]. Advances in Artificial Intelligence, 2009, 32(4): 95-101
[4] Y Koren, Factorization meets the neighborhood: a multifaceted collaborative filtering model[C]. Proceedings of the 14th ACM SIGKDD international conference on knowledge discovery and data mining, 2008 :426-434
[5] Linden G, Smith B, York J. Amazon.com recommendations item-to-item collaborative filtering [J]. IEEE Internet Computing2003, 7(1): 76-80
[6] Y. Shi, M. Larson, and A. Hanjalic. List-wise learning to rank with matrix factorization for collaborative filtering, ACM, 2010, 269-272.
[7] Sarwar B, Karypis G, Konstan J, et al. Item-based collaborative, 10th International Conference on World Wide Web ACM, 2001: 285-295
[8] ZHAO Feng, XIONG Yan, LIANG Xiao, GONG Xudong and LU Qiwei, Privacy-Preserving Collaborative Filtering Based on Time-Drifting Characteristic, Chinese Journal of Electronics, Vol.25, No.1, Jan. 2016
[9] Lei Ren, A Time-enhanced Collaborative Filtering Approach, 2015 4th International Conference on Next Generation Computer and Information Technology.2015:112-116
[10] Qing Li, Jia Wang, Yuan zhu, Peter Chen, Zhangxi Lin, User comments for news recommendation in forum-based social media, Information Sciences 180 (2010) 4929–4939

[11] Y Koren, Factor in the neighbors: Scalable and accurate collaborative filtering, - ACM Transactions on Knowledge Discovery from Data, 2010, 4(1):1-11