Postprint: Research on Recommendation Algorithms Based on User Preference Optimization Models

Authors: Qiu Ningjia, He Zhuang, Wang Peng, Li Yanfang

Date: 2018-10-11T00:00:00+00:00

Abstract

Traditional personalized recommendation algorithms generally suffer from the data sparsity problem, which affects recommendation accuracy. The Slope one algorithm is characterized by simplicity and efficiency; however, it only performs data analysis based on the user-item rating matrix, applies uniform weights to all users, and neglects users' preference levels for item categories. To address these issues, this paper proposes the LR-Slope one algorithm. First, a user preference matrix for item categories is constructed based on the user-item rating matrix and item category information. Then, a linear regression model is employed to calculate user-specific weights for each category, with the stochastic gradient descent algorithm used to optimize these weights. Finally, the Slope one algorithm is integrated to predict ratings and fill the rating matrix, thereby enhancing recommendation quality. Experimental results demonstrate that the proposed algorithm improves recommendation precision and effectively mitigates the sparsity problem.

Full Text

Preamble

Title: Research on Recommendation Algorithm Based on User Preference Optimization Model

Authors: Qiu Ningjia a,b, He Zhuang a, Wang Peng a, Li Yanfang a
(a. School of Computer Science & Technology; b. Institute of Computer & Information Technology, Changchun University of Science and Technology, Changchun 130022, China)

Abstract: Traditional personalized recommendation algorithms generally suffer from data sparsity, which affects recommendation accuracy. The Slope One

algorithm is simple and efficient, but it only analyzes data based on the user-item rating matrix, applying uniform weights to all users and ignoring users' preference degrees for item types. To address these issues, this paper proposes the LR-Slope One algorithm. First, we construct a user-item type preference matrix based on the user-item rating matrix and item type information. Then, we use a linear regression model to calculate the weight for each type and optimize the weights using stochastic gradient descent. Finally, we combine the Slope One algorithm to predict ratings, fill the rating matrix, and improve recommendation quality. Experimental results demonstrate that the proposed algorithm enhances recommendation precision and effectively alleviates the sparsity problem.

Keywords: recommendation algorithm; Slope One; user preference; rating prediction

0 Introduction

With the popularization of the Internet and the rapid development of e-commerce, the number of users and products has grown exponentially. When searching for desired products among numerous items, keyword searches only yield simple results that fail to meet users' personalized needs. This has given rise to personalized recommendation systems. The core of a recommendation system is the recommendation algorithm, which primarily calculates similarity between users or items to recommend content of interest or similar items to target users [**?**]. However, due to the scarcity of item rating data, prediction inaccuracy remains a persistent issue, with data sparsity being one of the key factors affecting recommendation quality [**?**].

To address data sparsity, many scholars have proposed solutions based on dimensionality reduction techniques, such as Principal Component Analysis [**?**], Singular Value Decomposition [**?**, **?**], and NMF [**?**, **?**]. Du Maokang et al. [**?**] first clustered data from users' previous reviews and then combined Slope One algorithm to predict and fill ratings for unrated items. Zhang Jun et al. [**?**] introduced the concept of expert trustworthiness to predict ratings for unrated items. Guo Di et al. [**?**] employed a hybrid recommendation method based on popularity dimensionality reduction to streamline the data matrix. Zhao Changwei et al. [**?**] proposed a matrix factorization algorithm that hybridizes explicit and implicit attributes, using correlations of explicit attributes to constrain factor matrices, thereby suppressing overfitting in sparse data matrix decomposition and improving recommendation accuracy. Pan Junchi et al. [**?**] introduced user credibility and proposed an improved singular value decomposition algorithm that simultaneously considers users' trust information, trusted information, and respective implicit feedback, effectively improving recommendation accuracy. Li Qian et al. [**?**] proposed a collaborative filtering recommendation algorithm based on spectral clustering and multi-factor fusion, incorporating FCM clustering into

key steps of the spectral clustering algorithm to improve similarity measurement and enhance recommendation performance. Yu Yang et al. [**?**] proposed a collaborative filtering recommendation algorithm based on entropy-optimized neighbor selection, using Bhattacharyya coefficient to calculate inter-item similarity, weighting it to compute inter-user similarity, introducing entropy to describe user rating distribution characteristics to measure the recommendation contribution capability of neighbor users, and finally using dual criteria to calculate recommendation weights and construct neighbor sets to improve accuracy. Zhou Yang et al. [**?**] proposed a collaborative filtering algorithm based on stacked denoising autoencoders, which to some extent solved the problems of rating matrix sparsity and inability to calculate similarity between items without common user ratings. Liu Linjing et al. [**?**] proposed a weighted Slope One algorithm based on user similarity, which first screened active users and filled the rating matrix according to inter-item similarity, effectively alleviating data sparsity.

While these methods have mitigated data sparsity to some extent, they suffer from high computational complexity and cannot effectively solve real-time recommendation problems when dealing with large data volumes and rapid updates. The Slope One algorithm, proposed by Lemire et al. [**?**], is a collaborative filtering recommendation algorithm with advantages of simplicity, efficiency, recommendation accuracy, and real-time recommendation capabilities. However, issues such as cold start and data sparsity constrain its development. To address the Slope One algorithm's limitation of not considering item type characteristics and using only user ratings as the basis for recommendation, this paper obtains user type weights by constructing a user-type preference matrix based on users' preference degrees for types, adjusts each user' s rating using optimized weights, makes prediction ratings more accurate, completes the user-rating matrix, and thereby alleviates data sparsity.

---

1 Improved Slope One Personalized Recommendation Algorithm

1.1 Problem Description

The general process of traditional collaborative filtering recommendation algorithms consists of four main parts:

a) Construct the user-item rating matrix $R_{m\times n}$. Users' historical rating data for items largely reflects their preference degrees for items and serves as the fundamental data basis for collaborative filtering algorithms, typically represented by a user-item rating matrix.

b) Obtain similar neighbors of users or items. Based on the rating matrix $R_{m\times n}$, different algorithms calculate similarity between users or between items, selecting the top K users or items to form a neighbor set.

c) Predict item ratings. The current user' s rating for a target item needs to be calculated based on ratings from its neighbor set.

d) Generate recommendation lists. The predicted rating values of target users for different items are sorted, and the top N items are selected to form a recommendation list as the final result for the target user.

The user-item rating matrix is shown in Table 1 , where rows represent users and columns represent items; rating values range from 0 to 5, with '-' indicating no rating and 5 representing the highest score. As can be seen, this rating matrix is a sparse matrix. Making recommendation predictions based on such a sparse matrix clearly yields low recommendation accuracy.

1.2 Slope One Recommendation Algorithm

The Slope One algorithm is an Item-Based collaborative filtering recommendation algorithm. Its basic idea is to use users' historical rating data and preference averages to predict users' preference values for new items. The principle is that a linear relationship exists between preference values of two items, allowing estimation of a user' s preference value for a new item from their preference value for item x. The key calculation is determining the average preference value between pairs of items.

The main steps of the Slope One algorithm are as follows:

a) Calculate the mean rating difference between pairs of items that have been co-rated, recorded as the rating deviation between items:

$$\mathrm{dev}_{i,j} = \frac{\sum_{u \in N_{i,j}} (r_{u,i} - r_{u,j})}{|N_{i,j}|}$$

where $r_{u,i}$ is user u' s rating for item i; $r_{u,j}$ is user u' s rating for item j; N_i is the set of users who rated item i; $N_{i,j}$ is the set of users who rated both items i and j; and $|N_{i,j}|$ is the number of users in the set.

b) Predict users' ratings for unrated items based on inter-item rating deviations and users' historical ratings:

$$P_{u,i} = \frac{\sum_{j \in N_u} (\mathrm{dev}_{i,j} + r_{u,j})}{|N_u|}$$

where N_u is the set of items rated by user u.

c) Sort the predicted ratings and recommend the top N items to the target user.

1.3 Construction of User Type Preference Matrix

Traditional collaborative filtering recommendation algorithms typically consider only users' ratings for items. This paper argues that item types also influence user preferences to some extent, and incorporating item type factors as a reference can yield better prediction accuracy. Considering that users have different preference degrees for different movie types, we obtain users' preference ratings for each type by counting the number of types and items they have rated, and combine this with the average rating of rated items to construct a user-type preference matrix as the data foundation for subsequent model operations. The formula is as follows:

$$Z_{u,x} = \frac{\sum_{i \in I_{u,x}} r_{u,i}}{|I_{u,x}|}$$

$$\overline{r}_u = \frac{\sum_{i \in I_u} r_{u,i}}{|I_u|}$$

where $I_{u,x}$ represents the set of items rated by user u that contain type x; $r_{u,i}$ represents user u's actual historical rating for item i; $|I_{u,x}|$ represents the number of elements in the set; and $|I_u|$ represents the number of movies watched by user u.

The specific steps for constructing the user-type preference matrix are as follows:

a) According to Equation (5), set unrated items in the user-item rating matrix $R_{m \times n}$ to 0, as shown in Table 2 .

b) According to Equation (6), count item type information to obtain the item type information matrix T, as shown in Table 3 .

c) According to Equations (3) and (4), obtain users' preference rating $Z_{u,x}$ for each type and the average rating $\overline{r}_u$ of rated items, and construct the user-type preference matrix P, as shown in Table 4 .

In the type matrix, if a type exists in the item, it is set to 1; otherwise, it is set to 0, i.e.:

$$P_{j,x} = \begin{cases} 1 & \text{if item } j \text{ contains type } x \\ 0 & \text{if item } j \text{ does not contain type } x \end{cases}$$

1.4 Improved Prediction Rating Scheme

Data sparsity has always been a challenge for recommendation algorithms. Filling missing rating items is one method to solve this problem and significantly impacts the final prediction results. This paper constructs a user-type preference matrix by incorporating users' preference degrees for types, applies a linear

regression model to obtain type weights, optimizes weight parameters, and combines the LR-Slope One algorithm to fill the sparse matrix. Using the filled dataset for effective recommendation, and considering type preferences for each user rating, the recommendation results better align with users' preferences and thoughts, yielding higher recommendation quality. The overall solution of the recommendation algorithm is as follows:

a) First, traverse the user-item rating dataset S to obtain the type set C.

b) Based on type set C, count users' viewing times C_u for each type and the total rating F_u for that type.

c) According to C_u and F_u, apply Equations (3) and (4) to obtain users' preference rating $Z_{u,x}$ for each item type and the average rating $\overline{r}_u$ of movies they have watched.

d) Through dataset S, C, and F, form the user-type preference matrix P as the test set.

e) Optimize the model using Algorithm 2 on matrix P to obtain users' weight ω for each type.

f) Bring the target item' s weight ω_{j,G_u} into Algorithm 1 to finally obtain the target item' s predicted rating $r_{u,j}$, completing the user-item rating matrix S.

g) Generate recommendation lists for target users based on predicted rating values.

The overall flow of the recommendation algorithm is shown in Figure 1 [Figure 1: see original paper].

---

2 Preference Model Optimization and Improved Algorithm

2.1 Preference Model Optimization

This paper uses the user-type matrix to construct a preference model. The general form of this model is:

$$\hat{y} = \omega^T x$$

Given parameters ω, we can obtain the sample's predicted value $\hat{y}$. The deviation between the predicted value and the sample' s actual rating value is:

$$H(\omega) = \frac{1}{2}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

where y_i is the actual rating value and $\hat{y}_i$ is the sample' s predicted value. This paper uses the gradient descent method to measure the quality of the preference model. The optimization method is as follows: first calculate the gradient $\nabla H(\omega)$; then update the learning rate α along the gradient direction; finally, repeatedly execute the above operations until the set loss function convergence threshold A is reached, where $H(\omega)$ is minimized and the ω value is the optimal weight ω^*. Bring ω^* back into the model to replace ω and obtain the final model. Based on each item type' s weight component, obtain the target item' s type rating. The specific algorithm description is shown in Algorithm 2.

Algorithm 2: Preference Model Optimization Algorithm

Input: Parameter set ω, rating set y, dataset S, preference matrix P, sample count n, type count T, loss function convergence threshold A.

Output: Item type weight ω.

1. while $H(\omega) > A$ do:
2. for $i = 1$ to n do:
3. `Calculate $\hat{y}_i$ according to Equation (9)`
4. end for
5. for $i = 1$ to n do:
6. `$\omega = \omega - \alpha \nabla H(\omega)$`
7. end for
8. end while
9. Output item type weight ω

2.2 Improved Recommendation Algorithm

The Slope One algorithm is simple, efficient, and accurate in recommendation, but data sparsity problems persist. Since the Slope One algorithm only uses the average rating deviation between a user' s item ratings and the target item during calculation, without considering users' preferences for item types, it is highly likely to include items irrelevant to the target item in the calculation set during the target item rating prediction process, resulting in large calculation deviations and low recommendation accuracy. Therefore, this paper proposes the LR-Slope One algorithm, which fully considers users' preference degrees for item types, constructs a user-type preference matrix, uses type preference values as parameters for subsequent model operations, obtains type weights using a linear regression model, and optimizes weights using stochastic gradient descent. The improved deviation formula is as follows:

$$\mathrm{LRdev}_{i,j} = \frac{\sum_{u \in N_{i,j}} \left((r_{u,i} - r_{u,j}) \cdot \omega_j \right)}{|N_{i,j}|}$$

where $\mathrm{dev}_{i,j}$ represents the deviation between items i and j; $N_{i,j}$ represents the set of users who rated both items i and j; $r_{u,i}$ and $r_{u,j}$ represent user u' s ratings for items i and j respectively; I_u represents the set of items co-rated with the target item; ω_i and ω_j represent the item weights for items i and j respectively; and $|N_{i,j}|$ represents the number of elements in the set.

By considering the influence of item type preferences on each item rating separately and combining it with the Slope One algorithm to predict item ratings, we can complete the user-rating matrix and make prediction results more accurate while alleviating rating data sparsity. The improved prediction rating formula is as follows:

$$P_{u,i} = \frac{\sum_{j \in I_u} (\mathrm{LRdev}_{i,j} + r_{u,j}) \cdot \omega_j}{|I_u|}$$

The specific description is shown in Algorithm 1.

Algorithm 1: LR-Slope One Algorithm

Input: User-item rating dataset S, weight ω, preference matrix P, sample count n, G_u as the set of other items rated by user u.

Output: Predicted rating $r_{u,j}$ for target item.

1. for each S:
2. return user set U
3. return item set I
4. end for
5. for each T:
6. Obtain weight ω according to Equation (8)
7. for $i = 1$ to n:
8. `Optimize weight $\omega$ according to Algorithm 2`
9. end for
10. end for
11. for each S:
12. Obtain deviation between items according to Equation (7)
13. end for

14. Output predicted item rating according to Equation (8)

---

3 Experiments

3.1 Experimental Data

The experimental operating system used in this paper is Windows 10, and all programs were implemented in Python. The MovieLens dataset [**?**] was used as the test dataset for experiments. It contains 100,004 ratings for 9,125 movies and 1,296 tag applications, with rating scores ranging from 1 to 5, where higher scores indicate deeper user preference for the movies.

This paper categorizes all movies into 19 genres including romance, musical, western, horror, etc. The dataset sparsity is 93.7%. The experiment randomly selects 80% of the entire dataset as the training set for model training and 20% as the test set for cross-validation of rating prediction accuracy and recommendation precision.

3.2 Experimental Evaluation Metrics

This paper uses Mean Absolute Error (MAE) and precision to measure the accuracy of algorithm rating prediction and verifies iterative convergence. MAE is a widely recognized method in the recommendation algorithm industry for evaluating recommendation system quality. The smaller the MAE value, the higher the prediction accuracy; conversely, the lower the prediction accuracy. The MAE formula is as follows:

$$\mathrm{MAE} = \frac{\sum_{i=1}^{C} |P_i - Q_i|}{C}$$

where P_i represents the predicted rating for item i; Q_i represents the actual rating for item i; and C represents the number of user-item ratings in the test set.

The precision metric (precision) measures the proportion of items in the recommended list that target users have actually rated. A higher proportion indicates better recommendation quality. The formula is as follows:

$$\mathrm{precision} = \frac{\sum_{u \in U} |R_u \cap T_u|}{\sum_{u \in U} |R_u|}$$

where R_u represents the N items recommended to user u; and T_u represents the set of items liked by user u.

3.3 Experimental Results and Analysis

Experiment 1: Preference Optimization Model Experiment. Based on the MovieLens dataset, this paper counts users' viewing times, movie genres, and average ratings of watched movies to construct a user-type preference matrix, as shown in Table 5 .

Table 5: User-Type Preference Matrix

After constructing the preference model, we count all users'preference weights for each type, use a linear regression model, and adopt gradient descent for weight parameter optimization to iteratively obtain optimal weights. Through extensive experimental verification, when the loss function convergence threshold is set to 0.0001 and the learning rate α is set to 0.001, reasonable optimization effects can be achieved through multiple iterations. The final optimized weights are shown in Table 6 .

Table 6: Weight Optimization Results

According to Table 6, we bring the trained optimized weights back into the linear regression model to calculate the predicted ratings of the user-type preference matrix. Comparing each user' s original rating data with the model-optimized predicted rating data, the results are shown in Figure 2 [Figure 2: see original paper].

Figure 2: Model Optimization Comparison

As shown in Figure 2, the predicted ratings from the trained preference optimization model closely follow the curve of original user rating data, demonstrating good fitting effects. Moreover, as the number of users changes, the predicted ratings from the trained model well reflect the trend of original user ratings, providing a reliable basis for subsequent recommendations using the proposed LR-Slope One algorithm.

Experiment 2: Improved Prediction Rating Algorithm Verification. Since recommendation algorithms fill the user-rating matrix based on predicted rating information, the quality of predicted ratings directly impacts recommendation quality. This experiment randomly selects six movies for rating prediction based on the known user-type preference matrix. First, we count the movie types as Action, Adventure, Drama, Sci-Fi, and Thriller, with optimized weights of ω_1, ω_2, ω_8, ω_{16}, and ω_{17} respectively. Bringing these weights into the proposed LR-Slope One algorithm, the predicted ratings are compared with those from the original Slope One algorithm and users' actual ratings, as shown in Table 7 .

Table 7: Predicted Rating Comparison

As shown in Table 7, the predicted ratings from the proposed LR-Slope One algorithm are closer to users' actual movie ratings than those from the original Slope One algorithm, indicating that predicted ratings incorporating user type

preference information are more accurate.

To further verify the rationality of the prediction rating algorithm, we conduct experimental verification using all users, comparing MAE effects between the original Slope One algorithm and the proposed LR-Slope One algorithm under different dataset sizes. The results are shown in Figure 3 [Figure 3: see original paper].

Figure 3: MAE Comparison

Figure 3 shows that the error of the original Slope One algorithm ranges between 0.73 and 0.82, while the error of the proposed LR-Slope One algorithm ranges between 0.63 and 0.72, with a maximum error reduction of 11 percentage points. Moreover, as users' rating data increases, the MAE value of predicted ratings becomes smaller. This is because the proposed LR-Slope One algorithm incorporates analysis of user type preference information for each rating, fully considering users' preference degrees for each type. The more users there are, the better the prediction effect and the higher the accuracy of ratings incorporating type preference information.

Experiment 3: Recommendation Effect Verification. Since in practical applications, users aim to obtain recommendations for movies they like rather than merely knowing rating status, to verify the recommendation accuracy of the proposed algorithm, this paper counts the proportion of movies that users have rated among the top 10 recommended movies with high ratings. We calculate recommendation accuracy under different user set sizes and compare it with the original Slope One algorithm, as shown in Figure 4 [Figure 4: see original paper].

Figure 4: Precision

As shown in Figure 4, the proposed algorithm improves precision by a minimum of 8.22% and a maximum of 13.92% compared to the original Slope One algorithm, demonstrating higher accuracy and better recommendation effects. The above experiments show that the proposed LR-Slope One algorithm has smaller prediction rating errors and higher recommendation accuracy, significantly improving recommendation system quality.

Experiment 4: Multi-Dataset Verification. To verify the effectiveness of the proposed LR-Slope One algorithm on different datasets, experiments were conducted on four different data packages: ml-100k.zip, ml-1m.zip, ml-10m.zip, and ml-20m.zip. The results comparing the proposed algorithm with the original Slope One algorithm are shown in Table 8 .

Table 8: MAE Comparison Across Multiple Datasets

As shown in Table 8, as the dataset size increases and data becomes sparser, the error of the original Slope One algorithm increases, while the proposed algorithm still maintains good accuracy. Although larger datasets bring more users, more movies, more rating data, and fewer common ratings for the same movies, the

number of movie types remains unchanged. Users' preference degrees for types still exist in reality. Combined with the optimized preference model proposed in this paper, prediction accuracy on datasets of different sizes is higher than that of the original Slope One algorithm. Therefore, the proposed LR-Slope One algorithm has stronger practicality.

4 Conclusion

To address the data sparsity problem in recommendation algorithms, this paper employs the Slope One recommendation algorithm. However, since this algorithm does not consider users' preferences for movie types and uses uniform weight calculation, it results in low accuracy of personalized recommendations. By constructing a user-type preference matrix, combining it with a linear regression model to optimize type weights, and fully considering users' preference degrees for different movies during subsequent rating prediction, this paper calculates different type weights for each user. Verification of the preference optimization model demonstrates good fitting effects. Comparing the algorithm' s predicted ratings with actual ratings, the LR-Slope One algorithm yields smaller prediction rating errors that are closer to actual ratings. In terms of final recommendation effectiveness, the proposed algorithm also significantly improves recommendation quality. Experimental results demonstrate that the proposed LR-Slope One algorithm outperforms the original Slope One recommendation algorithm in recommendation results, with high prediction rating accuracy, effectively improving recommendation quality and alleviating data sparsity.

References

[1] Leng Yajun, Lu Qing, Liang Changyong. Collaborative filtering recommendation algorithm based on structure similarity [J]. Journal of Chinese Computer Systems, 2015, 36(10): 2266-2269.

[2] Liu Limin, Zhang Pengxiang, Lin Le, et al. Research of data sparsity based on collaborative filtering algorithm [J]. Applied Mechanics & Materials, 2013, 462-463(2013): 856-860.

[3] Chen Jianmei, Sun Yajun. N dimensional tensor decomposition recommendation algorithm based on user' s neighbors [J]. Computer Engineering, 2017, 43(11): 193-197.

[4] Wang Jianfang, Zhang Pengfei, Liu Yongli. Research on improved bias probabilistic matrix factorization algorithm [J]. Application Research of Computers, 2017, 34(5): 1397-1400.

[5] Sun Lingfang, Feng Zunchang, SunLingfang, et al. Tag recommendation algorithm based on feature weighting and tensor decomposition [J]. Journal of

Jiangsu University of Science and Technology: Natural Science Edition, 2015, 29(6): 574-579.

[6] Wang Xiwei, Zhang Jun, Lin Pengpeng, et al. Incorporating auxiliary information in collaborative filtering data update with privacy preservation [J]. International Journal of Advanced Computer Science & Applications, 2014, 5(4): 224-235.

[7] Luo Xin, Zhou Mengchu, Xia Yunni, et al. An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems [J]. IEEE Trans on Industrial Informatics, 2014, 10(2): 1273-1284.

[8] Du Maokang, Liu Miao, Li Shaohua, et al. Slope one collaborative filtering recommendation algorithm based on neighbor [J]. Journal of Chongqing University of Posts and Telecommunications: Natural Science Edition, 2014, 26(3): 421-426.

[9] Zhang Jun, Liu Man, Peng Weiping, et al. Collaborative filtering recommendation algorithm based on fusion interest and score [J]. Journal of Chinese Computer Systems, 2017, 38(2): 357-362.

[10] Guo Di, Zhao Haiyan, Hou Jingde, et al. Hybrid recommendation via similarity propagation and dimension reduction based on popularity [J]. Journal of Chinese Computer Systems, 2015, 36(4): 707-712.

[11] Zhao Changwei, Peng Qinke, Zhang Zhiyong. A matrix factorization algorithm with hybrid implicit and explicit attributes for recommender systems [J]. Journal of Xi' an Jiaotong University, 2016, 50(12): 87-91.

[12] Pan Junchi, Zhang Xingming, Wang Xin. Improved singular value decomposition recommender algorithm based on user reliability [J]. Journal of Chinese Computer Systems, 2016, 37(10): 2171-2176.

[13] Li Qian, Li Shijin, Xu Guiqiong. Collaborative filtering recommendation algorithm based on spectral clustering and fusion of multiple factors [J]. Application Research of Computers, 2017, 34(10): 2905-2908.

[14] Yu Yang, Yu Hongtao, Huang Ruiyang. Collaborative filtering recommendation algorithm based on entropy optimization nearest-neighbor selection [J]. Application Research of Computers, 2017, 34(9): 2618-2623.

[15] Zhou Yang, Lu Jiaqi. Stacked denoising autoencoder for collaborative filtering algorithm [J]. Application Research of Computers, 2017, 34(8): 2336-2339.

[16] Liu Linjing, Lou Wengao, Feng Guozhen. New weighted Slope One algorithm based on user similarity [J]. Application Research of Computers, 2016, 33(9): 2708-2711.

[17] Lemire D, Maclachlan A. Slope One predictors for online rating-based collaborative filtering [J]. Computer Science, 2007: 21-23.

[18] Harper F M, Konstan J A. The MovieLens datasets [J]. ACM Trans on Interactive Intelligent Systems, 2016, 5(4): 1-19.

Note: Figure translations are in progress. See original paper for figures.