1. Problem model

Suppose we have a User-Movie data matrix $\mathbf{Y}$ as Fig. 1. The horizontal axis represents movies and the vertical axis represents users respectively. $\mathbf{Y}_{ij}$ means the score i-th user made on j-th movie.

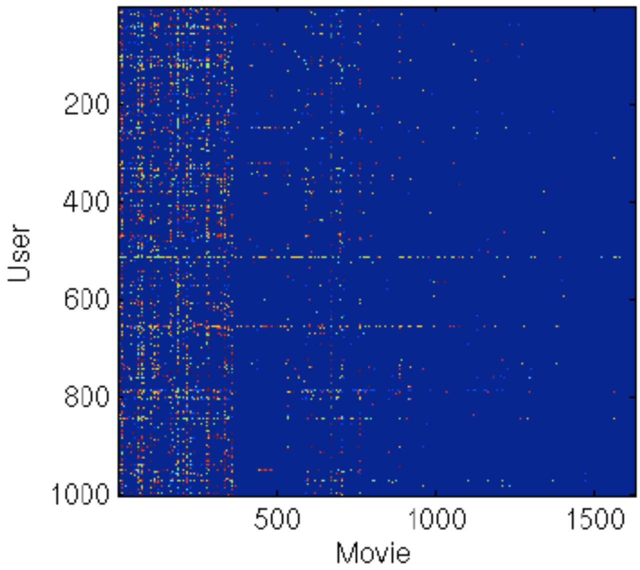

Fig. 1

Where the movie rated as 1-6 stars, which means the user's preference, as Fig. 2.

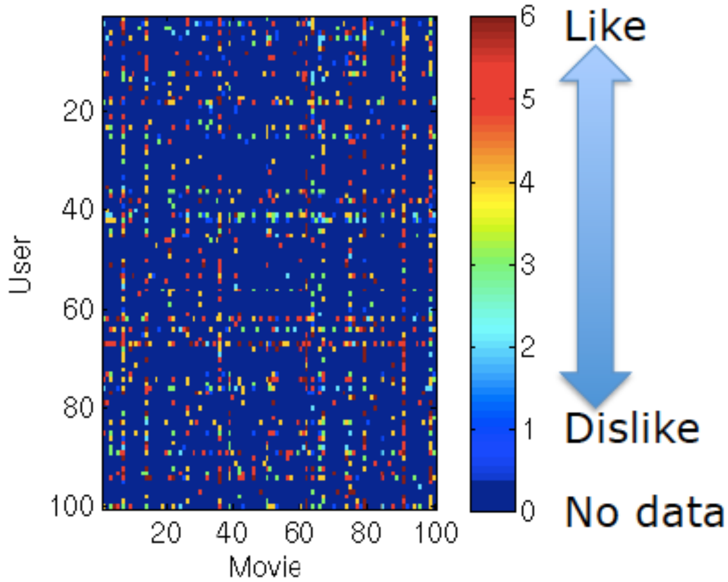

Fig. 2

However, the matrix is incomplete that with some missing values usually. Since that, we want to analyze the rest rating data of this matrix as training data to predict missing values, as Fig. 3.

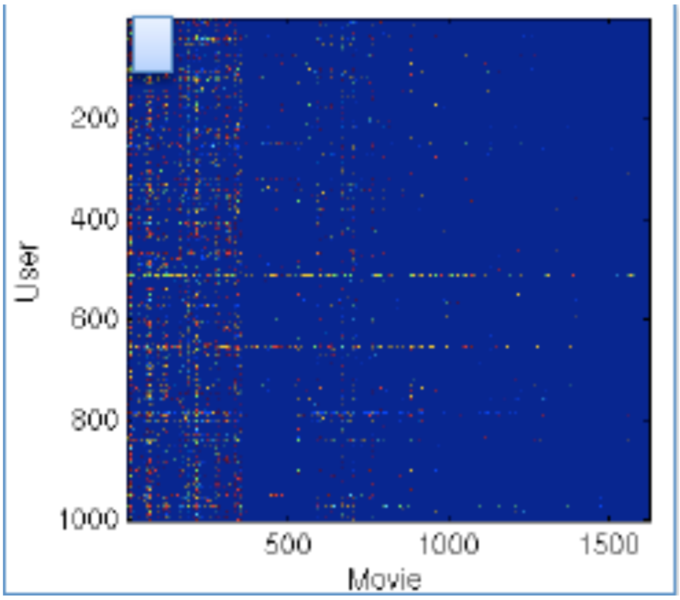

Fig. 3

In other words, building a recommend system to offer users favorable movies that masked in Fig. 3.

2. Solutions

2.1 Simple Approach *(only like and dislike)*

First, we transform ordinal rating data to binary [6,5,4]->1 (like), [3,2,1]->-1 (dislike), and then apply low-rank matrix factorization to the missing value matrix.

$$\mathbf{USV}^T = \mathbf{Y} \quad ,$$

The predict matrix can be given by

$$\mathbf{X}_{(K)} = \mathbf{U}_{(K)}\mathbf{S}_{(K)}\mathbf{V}_{(K)}^{\ T} \quad ,$$

Then we determine +1/-1 via sign of X_{ij} and calcucate root mean square error (RMSE) compared with binary data matrix Y.

The experiment results show RMSE versus factor rank K=1,2,3…,10 in a 1000*1623 data matrix, as Fig. 4.

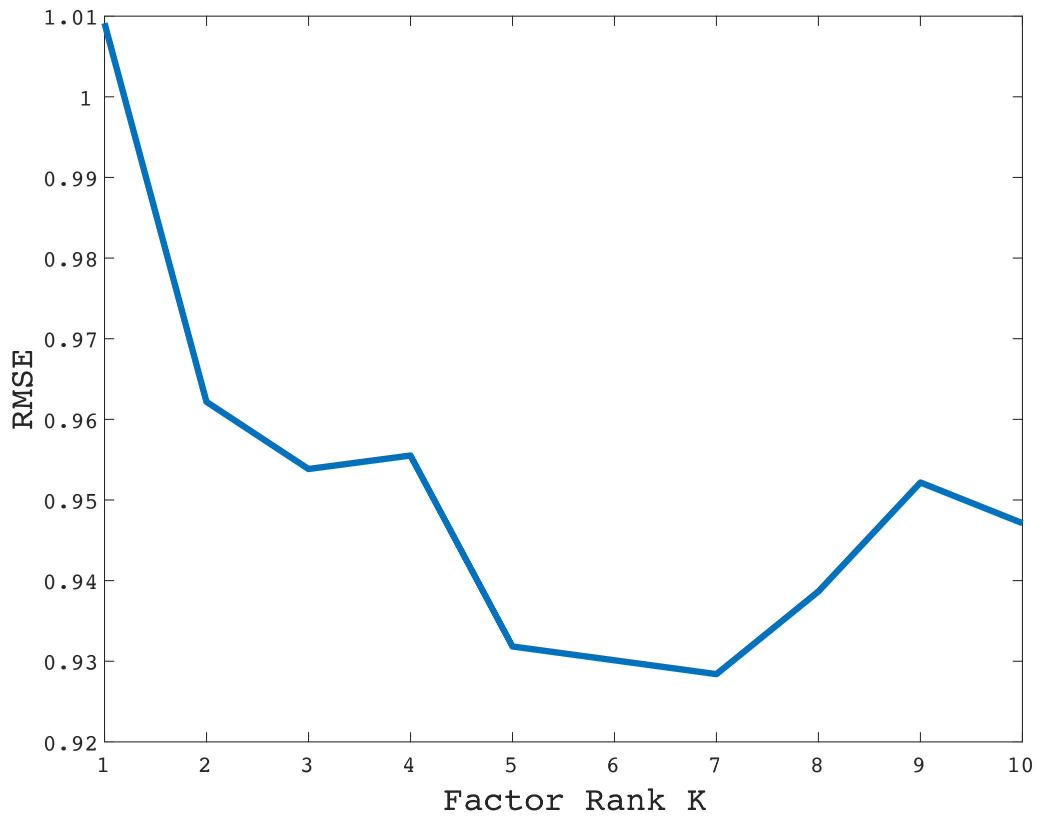

Fig. 4

and if the missing values are radomly generated in +1/-1, the RMSE has been calculated as 1.3863, which is much higher than results of simple approach. So it indicates that the recommend of simple approach outperforms than random recommend.

2.2 Weighted Low-Rank Approximations

Learn from [1], we apply weighted low-rank approximations method to prediction of the incomplete matrix. In this method, we utilize the formula to update predicting matrix until convergence,

$$X^{(t+1)} = \mathrm{LRA}_k\left(W \otimes A + (1 - W) \otimes X^{(t)}\right)$$

.

Where **A** is incomplete matrix, **X** is predicting matrix, **W** is weight matrix with weights decided by proportions of score stars 1-6 in interval [0,1] and became 0 if there is no data. LRA_k is the unweighted rank-k approximation of **X**, as can be computed from the SVD. $\otimes$ denotes element-wise multiplication and t is loop times.

The experiment results show RMSE versus factor rank k=1,2,3…,10 in a 1000*1623 data matrix, as Fig. 5.

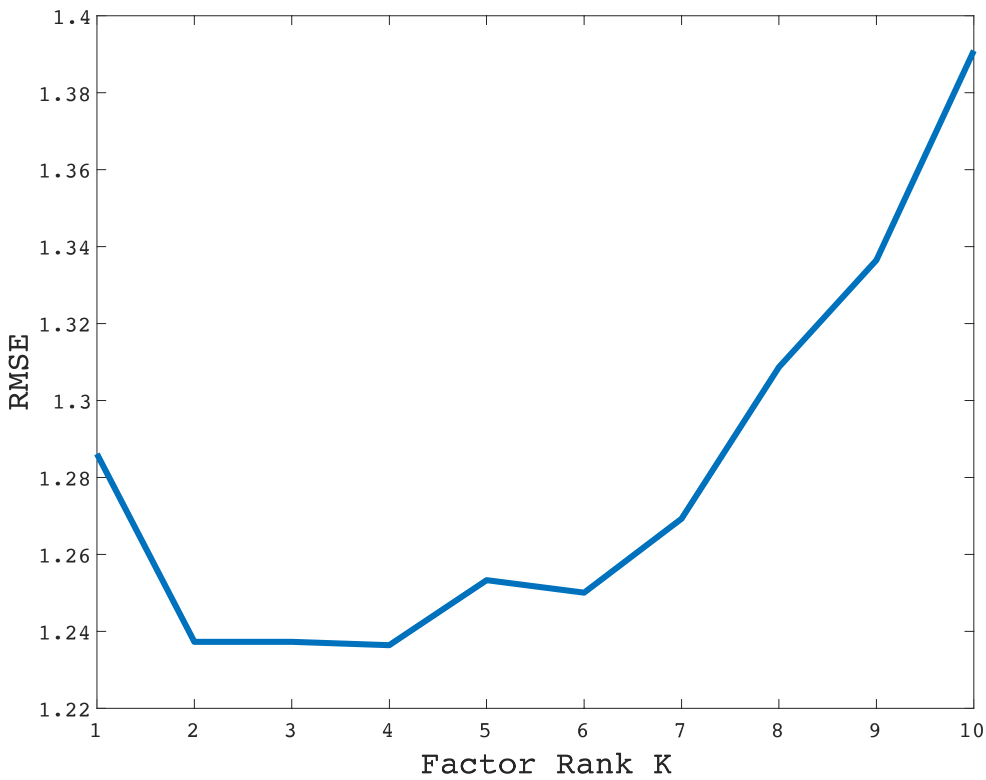

Fig. 5

It indicates that weighted low-rank approximations method has a good performance in prediction of missing value.

2.3 Gradient Descent Approach

Learn from [2], we solve this problem by applying gradient descent method. The main idea of gradient descent method is that setting an objective function E as follows:

$$
\begin{aligned}
E &= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{m} I_{ij}(V_{ij} - p(U_i, M_j))^2 \\
&+ \frac{k_u}{2}\sum_{i=1}^{n}\|U_i\|^2 + \frac{k_m}{2}\sum_{j=1}^{m}\|M_j\|^2,
\end{aligned}
$$

Where **V** is incomplete matrix, **U** and **M** are components of predicting matrix **X**, $p(\mathrm{U}_i, \mathrm{M}_j) = \mathrm{U}_i^T \mathrm{M}_j \approx$ **X**, **I** is , k_u and k_m are regularization coefficients. Gradients can be calculated by:

$$
\begin{aligned}
-\frac{\partial E}{\partial U_i} &= \sum_{j=1}^{m} I_{ij}\left((V_{ij} - p(U_i, M_j))M_j\right) - k_u U_i, i = 1, ..., n, \\
-\frac{\partial E}{\partial M_j} &= \sum_{i=1}^{n} I_{ij}\left((V_{ij} - p(U_i, M_j))U_i\right) - k_m M_j, j = 1, ..., m.
\end{aligned}
$$

Then we can obtain the minimum of objective function **E** by algorithm:

Select a learning rate μ, and regularization coefficients k_u, k_m.

1. Set the starting values of matrices U, M.
2. Repeat
   (a) Compute gradients ∇_U and ∇_M by (5) and (6).
   (b) Set $U \leftarrow U - \mu\nabla_U$, $M \leftarrow M - \mu\nabla_M$.

   until the validation RMSE starts to increase.

This time, in the experiment about 1000*1623 data matrix, we set the **U** and **M** accroding to

$$
U_{ij}, M_{ij} = \sqrt{\frac{\bar{V} - a}{f}} + n(r) \text{ for each } i, j,
$$

where $\overline{\mathrm{V}}$ is the average of all existing scores, a is the lower bound of scores, f is the dimension of SVD algorithm, and $n(r)$ is a random noise with uniform distrbtuion in $[-r, r]$, because the start points will affect performance. Moreover, we set $\mu = 0.001$, k_u and k_m equal to 0.02, dimension $f = 5$.

The RMSE of experiment result X_{error}=0.9980, which indicates the algoritem can solve this problem in a pretty good way.

3. References

[1] Srebro, Nathan, and Tommi Jaakkola. "Weighted low-rank approximations." *Icml*. Vol. 3. 2003.

[2] Ma, Chih-Chao. "A guide to singular value decomposition for collaborative filtering." (2008).