INFORMATION
Volume 12,Number 5, pp.1165-1182

ISSN 1343-4500
©2009 International Information Institute

# Recommendation Systems and Their Preference Prediction Algorithms in a Large-Scale Database

Seiji Takimoto, Hideo Hirose *

*Kyushu Institute of Technology, Department of Systems Design and Informatics, Iizuka, Fukuoka, 820-8502 Japan*

**Abstract**

As the market of electronic commerce grows explosively, it becomes more and more important to provide the recommendation system which suggests the preferred items for consumers using the large-scale customers database. In this paper, we discuss the algorithms and their performances of the recommendation systems using the collaborative filtering in the case of the Netflix database: they are, 1) memory-based system ($k$-nearest neighbor using the correlation coefficients), 2) model-based system (matrix decomposition), and 3) the combination method. When the customer-item matrix is a sparse matrix like the Netflix database, the matrix decomposition method shows better performance than the $k$-nearrest neighbor; in addition, it is found that the combination method of the two methods provide a much better performance.

*Key words:* Netflix, collaborative filtering, $k$-nearest neighbor, matrix decomposition, singular-value decomposition, combination method.

## 1 Introduction

In October 2006, a surprising news, the *Netflix Prize* competition, was announced. The competition is held by Netflix, an online DVD-rental service, with the grand prize of \$1,000,000 for the best collaborative filtering (CF) algorithm that predicts user

* Corresponding author.
*Email address:* hirose@ces.kyutech.ac.jp (Hideo Hirose).

ratings for films, based on previous ratings. The performance is measured by the *RMSE* (shown later), and the prize will be given to the entry who attains 10% improvement to the Netflix's own algorithm.

Although the collaborative filtering is known to be important as Amazon.com and Yahoo! may use in everyday *e*-commerce, this news, however, has again drawn our attention to the recommendation systems which are extremely valuable to entrepreneurs in *e*-commerce world.

There are mainly two kinds of recommendation systems; one is the profile-based or content-based system and the other is the collaborative filtering system. The prifile-based system uses the profiles of users as the explanation variables, and it uses the decisions of users as the objective or target variables. Some database may provide such profiles, but the database like the film recommendation systems may not do so much. In such a case, the collaborative filtering system will work. There are two kinds of systems in the collaborative filtering systems; one is the memory-based system, and the other is the model-based system. See Figure 1.

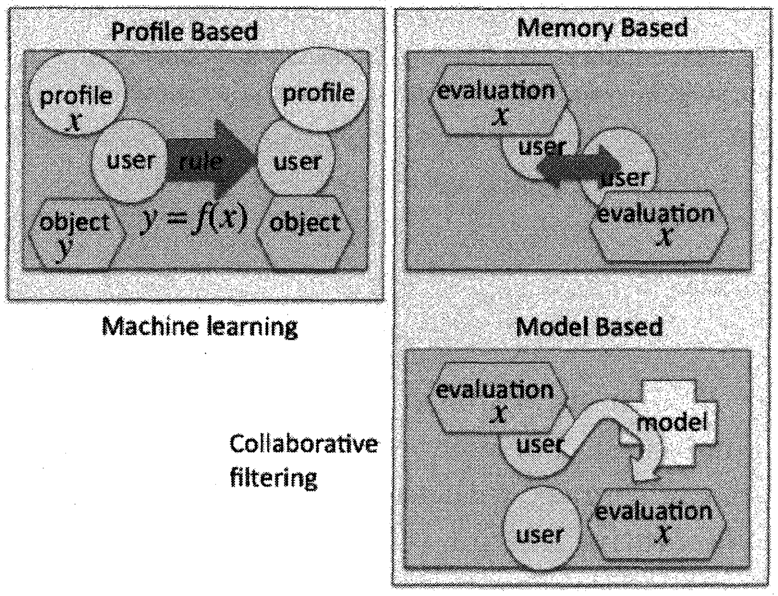

Fig. 1. Methods in Recommendation Systems.

A typical example for the memory-based system is to use the $k$-nearest neighbor ($k$-NN) with the (Pearson's) correlation coefficients, and that for the model-based system is to use the matrix-decomposition method which is also called the singular-value decomposition (SVD) method.

## 2 Netflix data and the evaluation criterion

Netflix provides a data set of 100,480,507 ratings that 480,189 users give to 17,770 movies. We observe, thus, in the data set, the sparseness of the user-movie matrix; $100,480,507/(480,189 \times 17,770) = 0.0118$. Each training rating is a quadruplet (user, movie, date of grade, grade). The user and movie fields are integer IDs, while grades are from 1 to 5 stars. Three kinds of data sets are used in the competition. Training data and Probe data are shown to the contestants, and they are used to construct the prediction algorithm and self-evaluation. The third one is Qualifying data which contains 2,817,131 triplets (user, movie, date of grade), with grades known only to the jury. A participating team's algorithm must predict grades on the entire Qualifying set, but they are only informed of the score for half of the data, the quiz set. The other half is the test set, and performance on this is used by the jury to determine potential prize winners. Only the judges know which ratings are in the quiz set, and which are in the test set. See Table 1 and Figure 2. Since the performances by the contestants are evaluated by Netflix, the reported performances are considered to be fair.

Table 1
Netflix Data Size

| data name | data size | customer size | movie size |
|---|---|---|---|
| FullTraining | 100,480,507 | 480,189 | 17,770 |
| Training | 99,072,112 | 480,189 | 17,770 |
| Probe | 1,408,395 | 462,858 | 16,938 |
| Qualify | 2,817,131 | 478,615 | 17,470 |

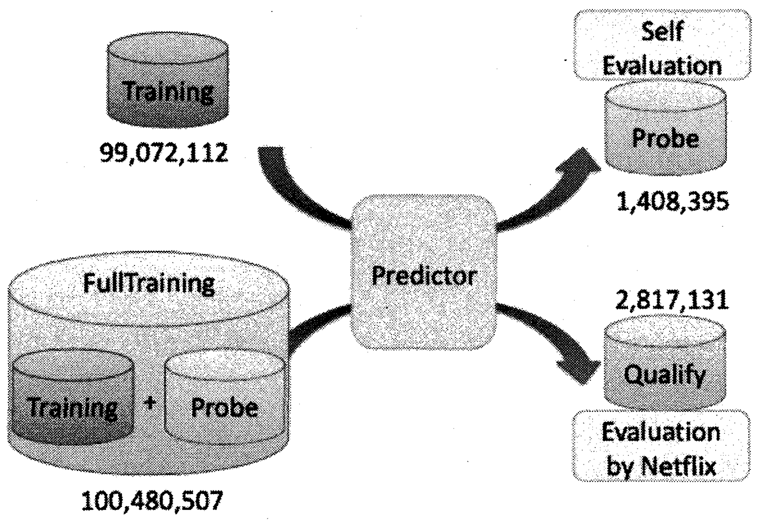

Fig. 2. Data Flow for Algorithm Evaluation

The performance is evaluated by the criterion of the root mean squared error, $RMSE$, between the predicted scores $\hat{x}(i,j)$ and the observed scores $x(i,j)$, where $i$ denotes the user ID and $j$ denotes the movie ID. If we define the indicator function $I(i,j)$ such that

$$\begin{aligned} I(i,j) &= 1, \quad \text{if } x(i,j) \in \{1,2,3,4,5\}, \\ &= 0, \quad \text{if } x(i,j)\text{: vacant,} \end{aligned} \tag{1}$$

the $RMSE$ is expressed by

$$\begin{aligned} RMSE(T) &= \sqrt{\frac{1}{|T|}\sum_{i,j} I(i,j)(\hat{x}(i,j)-x(i,j))^2}, \\ &(|T| = \sum_{i,j} I(i,j)). \end{aligned} \tag{2}$$

Using the definition above, we can say that *the collaborative filtering problem can be described such that we want to estimate* $\hat{x}(i,j)$, $(I(i,j)=0)$ *from* $x(i,j)$, $(I(i,j)\neq 0)$.

The $RMSE$ given by Cinematch which is computed by Netflix's own algorithm is 0.9514, and the prize will be given to the first entry who attains the 10% improvement of $RMSE$, that is 0.8563, by October 2, 2011. We define the relative improvement to Cinematch by $riC$ such that

$$riC = \frac{RMSE_{\text{Cinematch}} - RMSE_{\text{proposed}}}{RMSE_{\text{Cinematch}}}. \tag{3}$$

Roughly estimated $RMSE$ values by simple methods are shown as the reference here; they are obtained by putting values of $\hat{x}(i,j)$ such that, 1) $\hat{x}(i,j) = \mu$, where $\mu$ is the mean value of all $x(i,j)$, $(I(i,j) \neq 0)$, 2) $\hat{x}(i,j) = \mu_i$, where $\mu_i$ is the mean value of all $x(i,j), (i$ : fixed), 3), $\hat{x}(i,j) = \mu_j$, where $\mu_j$ is the mean value of all $x(i,j), (j$ : fixed). These $RMSE$ are shown in Table 2.

Table 2
*RMSE* by Simple Methods.

| | *RMSE* | *riC* (%) |
|---|---|---|
| $\mu$ | 1.1312 | −18.9 |
| $\mu_i$ | 1.0655 | −12.0 |
| $\mu_j$ | 1.0536 | −10.7 |
| Cinematch | 0.9514 | 0 |

using Qualify

We have been so far searching for efficient algorithms using various methods (see references (14), (15), (16)); however, we could not accomplish the performance improvement of 5% to the Cinematch result. In this paper we show that we have made a substantial progress by using the proposed method.

## 3 *k*-nearest neighbor method

The most common approach to CF is the neighborhood-based approach (see references (1), (2), (7), (8)). There are two kinds of approach in the $k$-nearest neighbor method: one is the user-oriented approach and the other is the item-oriented approach. Here, we use the item-oriented approach because 1) the average movie is rated by over 5,000 users and the average user rates over 200 movies in the training set, 2) an overwhelming portion (99%) of the user-item matrix is unknown, and 3) the pattern of observed data may be very non-random. Sarwar et al. (13) found that item-oriented approaches deliver better quality estimates than user-oriented approaches while allowing more efficient computations.

### *3.1 Correlation coefficients*

Assume we are given ratings about $m$ users and $n$ items, arranged in an $m \times n$ matrix $X = (x(i,j))$, $(1 \le i \le m, 1 \le j \le n)$. Now, to estimate the unknown $x(i,j)$, we identify a set of neighboring items $N(j;i)$ that other users tend to rate similarly to their rating of $j$. All items in $N(j;i)$ must have been rated by $i$. Then, $\hat{x}(i,j)$, the estimated value of some function of $x(i,j)$, is taken as a weighted average of the ratings of neighboring items:

$$\hat{x}(i,j) = \frac{\sum_{l \in N(j;i)} s(j,l) f(x(i,l))}{\sum_{l \in N(j;i)} s(j,l)}, \tag{4}$$

where $f(x(i,l))$ is a function of $x(i,l)$, and it will be introduced later. The item-item similarities (denoted by $s(j,l)$) are typically taken as either correlation coefficients or cosine similarities. Here, we use the (Pearson's) correlation coefficient and it is denoted by

$$s(j,l) = \frac{\sum_k (x(k,j) - \eta_j)(x(k,l) - \eta_l)}{\sqrt{\sum_k (x(k,j) - \eta_j)^2}\sqrt{\sum_k (x(k,l) - \eta_l)^2}}, \tag{5}$$

between items $j$ and $l$, where $\eta_j$ is the mean value for $x(k,j)$; $k$ is counted when $I(k,j) \neq 0$ and $I(k,l) \neq 0$.

We will use the variants of the correlation coefficient as the similarities because users are inclined to select preferred items than the disliked items so that $s(j,l)$ tends to show higher values. The variants are, 1) $ls(j,l)$, the lower percentile point for $s(j,l)$, and 2) $es(j,l)$, the size expansion index for $s(j,l)$.

### *3.2 Lower percentile point*

Due to the asymmetric property of the distribution for the correlation coefficient, Fisher's $z$-transform method is usually used to find the confidence limits. Since the raw values by Eq. (4) may be biased by the user's inclination, the use of the lower confidence limit is expected to produce the better performance in minimizing the $RMSE$. As a tuning parameter, we investigated various values of lower confidence limit point $ls(j,l)$ as the substitute for $s(j,l)$; the confidence probabilities are 0.7, 0.85, 0.95, 0.98, 0.999, and thus they correspond to 15, 7.5, 2.5, 1, 0.05 percentile points. By $z$-transformation, we can define

$$\begin{aligned} ls(j,l) &= \frac{\exp(2z_L)-1}{\exp(2z_L)+1}, \\ z_L &= z - \frac{z_{\alpha/2}}{\sqrt{n-3}}, \\ z &= \frac{1}{2}\log\frac{1+s(j,l)}{1-s(j,l)}, \end{aligned} \tag{6}$$

where $\alpha$ is the significance level probability, e.g., 2.5 percentile point when $\alpha = 0.05$.

### *3.3 Size-expansion index*

For taking account of the size of the users who like movies, we consider the following index,

$$es(j,l) = ls(j,l)^2 \log n, \tag{7}$$

where $n$ denotes the number of users who rate both movies $j$ and $l$. This is intended to emphasize the audience size of the movie.

### 3.4 *Bias corrected score*

As mentioned in 3.1, $f(x(i,l))$, a function of $x(i,l)$, actually becomes a bias corrected observed score,

$$f(x(i,l)) = \eta_i + (x(i,l) - \eta_l), \tag{8}$$

where, $\eta_i$ and $\eta_l$ denote the mean values of the scores.

### 3.5 *Performances by the k-nearest neighbor*

First, we take a look at the optimal size of $k$ in the three cases of similarity, $s(j,l)$, $ls(j,l)$, and $es(j,l)$, mentioned above. Figure 3 shows the $RMSE$ tendency to the number of neighbors when $\alpha = 0.05$. We can see that the optimal size of $k$ can be obtained around $k = 20$ to $k = 30$ when we use Probe data. The best performance values are shown in Table 3. We can see that the best performance is obtained by the size expansion index $es(j,l)$.

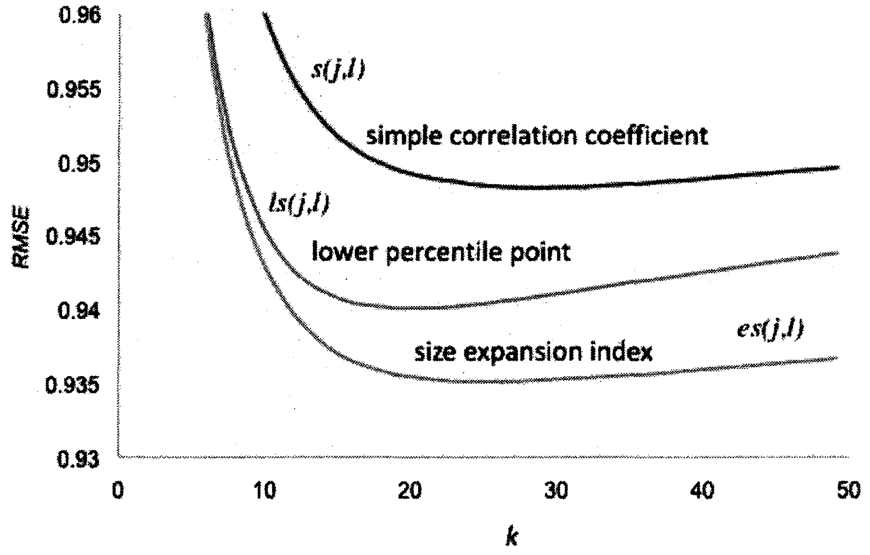

Fig. 3. Optimum Number of Neighbors for Three Similarities.

We, next, look at the $RMSE$ using $es(j,l)$ when $\alpha$ is dealt with as a parameter. Figure 4 shows the $RMSE$ using $es(j,l)$ vs. $\alpha$ when we select $es(j,l)$ as a similarity index. We can see that

Table 3
*RMSE* by Nearest Neighbors in Various Index.

| | $RMSE$ | optimal $k$ | $riC$ (%) |
|---|---|---|---|
| $s(j,l)$ | 0.9436 | 29 | 0.82 |
| $ls(j,l)$ | 0.9353 | 21 | 1.69 |
| $es(j,l)$ | 0.9281 | 25 | 2.45 |

using Qualify

the best performance is obtained when we adopt the value of $\alpha$ around 0.1.

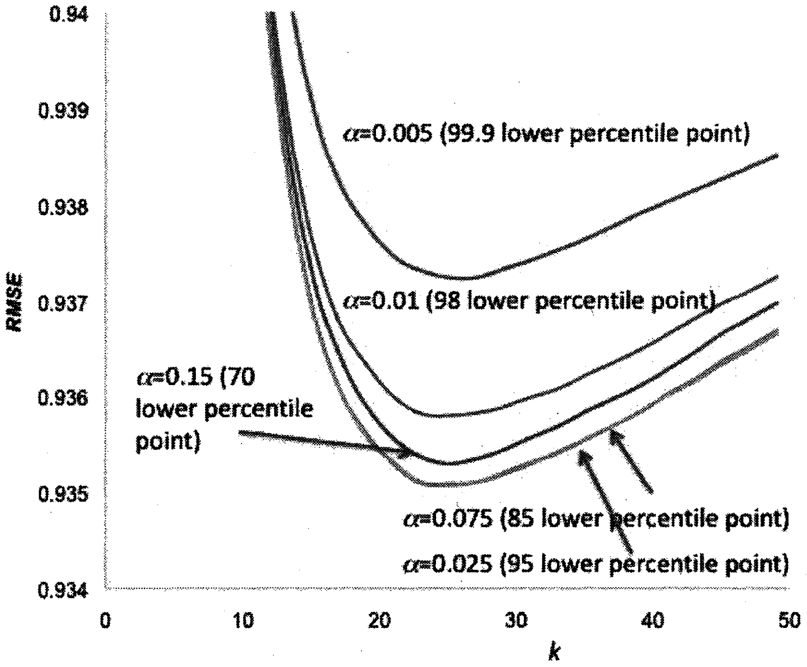

Fig. 4. Optimum Number of Neighbors for Indices.

The results for the $RMSE$ for various $\alpha$ values are shown in Table 4. We can see that the best performance is obtained when $\alpha = 0.075$ and $\alpha = 0.025$, and the optimal $k$ is around 25-27.

## 4 Matrix decomposition method

The singular-value decomposition, abbreviated as the SVD, is one of the factorization algorithms for various applications which include computing the pseudo-inverse, least squares fitting of data,

Table 4
*RMSE* by Nearest Neighbors (in the Case of $es(j,l)$).

| $\alpha$ | $RMSE$ | optimal $k$ |
|---|---|---|
| 0.15 | 0.93531 | 25 |
| 0.075 | 0.93507 | 25 |
| 0.025 | 0.93508 | 25 |
| 0.01 | 0.93583 | 27 |
| 0.005 | 0.93723 | 27 |

using Probe

matrix approximation, and determining the rank, range and null space of a matrix. Suppose $P \in R^{m\times n}$, $U \in R^{f\times m}$, and $M \in R^{f\times n}$ are matrices. A simple idea that a matrix factorization $P = U^T M$ produces the missing data of score matrix $V$ leads us to the use of the collaborative filtering. Thus, the matrix decomposition, which is also used for recommendation systems (see references (10), (11), (17)), is used for the least square method here. That is, we want to find the matrix $U$ and $M$ by minimizing the target function $E$ such that sum of the squares of the difference between the observed score $V(i,j)$ and the predicted score $P(U_i, M_j)$,

$$E = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I(i,j)(V(i,j) - P(U_i, M_j))^2, \tag{9}$$

where $P(U_i, M_j)$ denotes the $(i,j)$ element of $U^T M$. This idea of the matrix decomposition is derived by the usual SVD formulation such that $A = U\Sigma V^*$ where $U$ and $V$ are orthnormal and $\Sigma$ provides the singular values in the diagonal elements. If $\Sigma$ is absorbed by either or both $U$ and $V$, we can accomplish the matrix decomposition of $A$.

### *4.1 SVD*

Suppose $V \in R^{m\times n}$ is the score matrix of $m$ users and $n$ items, and $I \in 0, 1^{m\times n}$ is its indicator. The SVD algorithm finds two matrices $U$ and $M$ as the feature matrix of users and items. That is, each user or item has an $f$-dimension feature vector and $f$ is called the dimension of the SVD. A prediction function $p$ is used to predict the values in $V$. The value of a score $V(i,j)$ is estimated by $p(U_i, M_j)$, where $U_i$ and $M_j$ represent the feature vector of user $i$ and item $j$, respectively. Once $U$ and $M$ are found, the missing scores in $V$ can be predicted by the prediction function. For stable and robust computing, the optimization of $U$ and $M$ is actually performed by minimizing the sum of squared errors between the existing scores and their prediction values with penalty factors:

$$E = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I(i,j)(V(i,j) - p(U_i, M_j))^2 + \frac{k_u}{2}\sum_{i=1}^{m} || U_i ||^2 + \frac{k_m}{2}\sum_{j=1}^{n} || M_j ||^2 \tag{10}$$

where $k_u$ and $k_m$ are regularization coefficients to prevent overfitting; $||\cdot||$ means the Frobenius norm. This formulation is a kind of the ridge regressions. The most common prediction function is the dot product of feature vectors. That is, $p(U,M) = U^T M$. The optimization of $U$ and $M$ thus becomes a matrix factorization problem where $V \approx U^T M$ . But in most applications, scores in $V$ are confined to be in an intervel $[a,b]$, where $a$ and $b$ are the minimal and maximal score values defined in the domain of data. For example, if the users rate the objects as 1-5 stars, then the scores are bounded in the interval $[1,5]$. One way is to clip the values of dot products. For example, we can bound the values of $U_i^T M_j$ in the interval $[0, b-a]$ and the prediction function becomes a plus the bounded dot product. Hence, the prediction function makes a kind of trimming.

When using the prediction function of $p(U,M) = U^T M$, the objective function and its negative gradients have the following forms:

$$-\frac{\partial E}{\partial U_i} = \sum_{j=1}^{n} I(i,j)\left((V(i,j) - p(U_i, M_j))\frac{\partial p(U_i, M_j)}{\partial U_i}\right) - k_u U_i \tag{11}$$

$$-\frac{\partial E}{\partial M_j} = \sum_{i=1}^{m} I_{ij}\left((V(i,j) - p(U_i, M_j))\frac{\partial p(U_i, M_j)}{\partial M_j}\right) - k_m M_j \tag{12}$$

One can then perform the optimization of $U$ and $M$ by the descent gradient method by using the algorithm,

$$\begin{aligned} U^{(t+1)} &\leftarrow U^{(t)} + \mu\frac{\partial E}{\partial U} \\ M^{(t+1)} &\leftarrow M^{(t)} + \mu\frac{\partial E}{\partial M}, \end{aligned} \tag{13}$$

where $\mu$ is the learning rate. Figure 5 shows the $RMSE$ using the SVD method for Training data and Probe data when the number of embedded feature variable is varying. We can see that the larger the number of feature variables, the smaller the $RMSE$.

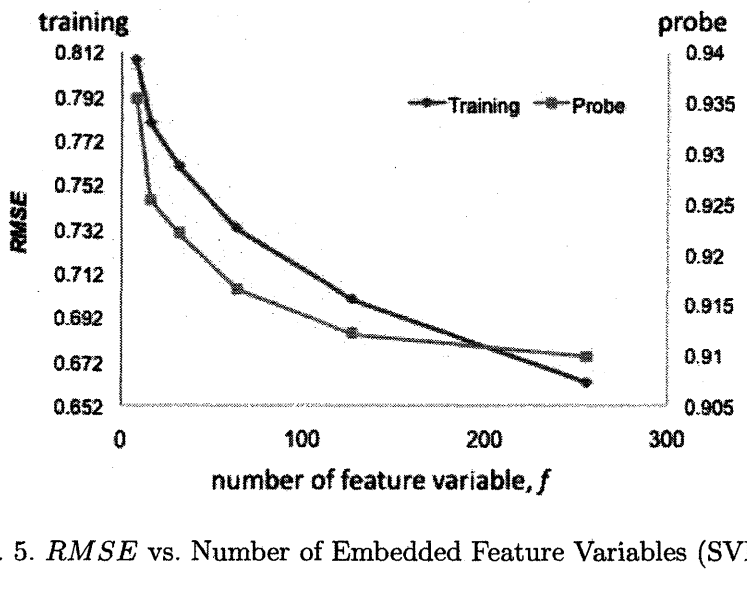

Fig. 5. $RMSE$ vs. Number of Embedded Feature Variables (SVD).

### *4.2 Biased SVD*

A variant of the SVD method is a biased SVD method, in which we consider biases for user $i$ and movie $j$. The sum of squared errors is expressed as

$$\begin{aligned} E = & \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I(i,j)(V(i,j) - fp(U_i, M_j))^2 \\ & + \frac{\lambda_1}{2}(\sum_{i=1}^{m} ||U_i||^2 + \sum_{j=1}^{n} ||V_j||^2) \\ & + \frac{\lambda_2}{2}({b_i}^2 + {b_j}^2), \end{aligned} \tag{14}$$

where $fp(U_i, M_j)$ is a bias corrected function for estimated score,

$$fp(U_i, V_j, b_i, b_j) = \mu + b_i + b_j + \sum_{k=1}^{f} u(i,k)v(k,j).$$

Figure 6 shows the $RMSE$ using the biased SVD method for Training data and Probe data when the number of embedded feature variable is varied. We can see that the smallest $RMSE$ can be obtained when the number of embedded feature variable is around 100 in Probe data. However, there seems no tendency of the "V" curve as well seen in the test data case in machine learning world. The larger the the number of embedded feature variable, the better performance is obtained.

### *4.3 Performances by the matrix decomposition*

Summering the results for the two methods, we can provide Table 5.

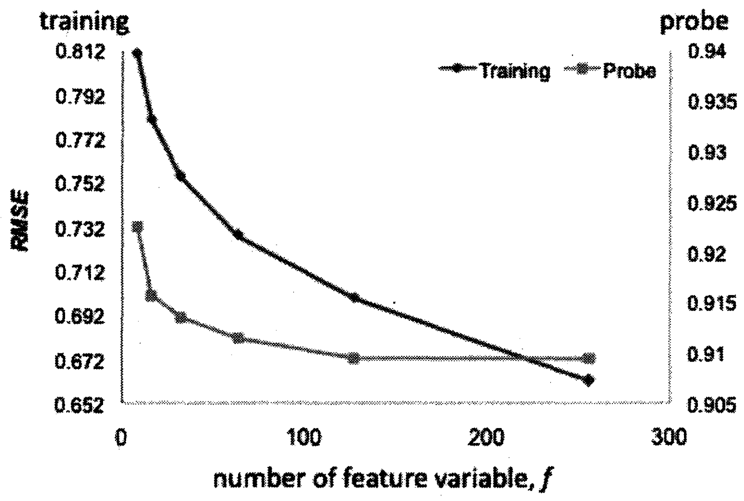

Fig. 6. *RMSE* vs. Number of Embedded Feature Variables (Biased SVD).

Table 5
*RMSE* by the Matrix Factorization.

| | *RMSE* | *riC* (%) |
|---|---|---|
| SVD | 0.9038 | 5.00 |
| biased SVD | 0.8995 | 5.46 |

using Qualify

## 5 Combination method

We have so far derived many prediction models; we call predictor $l$ to each model. To predictor $l$, we, here, denote the predicted score $\hat{x}(i,j)$ by $\hat{x}_l(i,j)$. The linear combination of the predictor associated with $L$ predictors can be expressed by

$$\hat{x}(i,j) = \sum_{l=1}^{L} w_l \hat{x}_l(i,j). \tag{15}$$

Then, the least square error is obtained by minimizing

$$E = \frac{1}{2}\sum_{i,j}(\hat{x}(i,j) - x(i,j))^2 + \frac{\lambda}{2}\sum_{l=1}^{L} w_l^2,$$

and this can be obtained by the descent gradient method with a penalty factor,

$$-\frac{\partial E(w_l)}{\partial w_l} = (\hat{x}(i,j) - x(i,j))\hat{x}_l(i,j) - \lambda w_l.$$

Using the algorithm,

$$w_l^{(t+1)} \leftarrow w_l^{(t)} + \nu \frac{\partial E}{\partial w_l}, \tag{16}$$

optimum $w_l$ can be obtained, where $\nu$ is the learning rate. In Table 6, we show the combination results using the two simple models; one is the method that the vacant elements are estimated by the mean value of the user, $\mu_i$, and the other is to use the mean value of the movie, $\mu_j$.

In Figure 7, we can see how $w_1$ affects the value of $RMSE$; when $w_1 = 0.46$ in using Probe data, the optimum $RMSE = 1.0154$ is obtained, which is smaller than 1.0688 obtained by single $\mu_i$ use and than 1.0528 obtained by single $\mu_j$ use. Thus, the combination methods are expected to provide higher performance than that in each model.

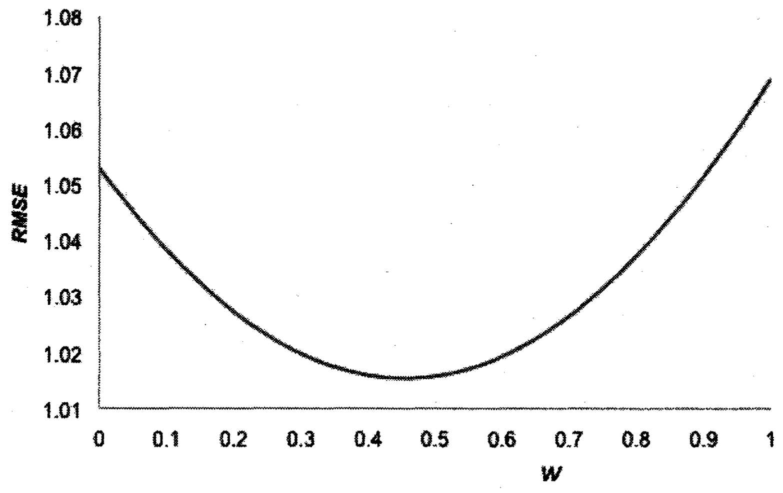

Fig. 7. $RMSE$ Values by the Combination Method.

We, next, try to combine the best predictors in single use models we have introduced above. That is the combination of $es(j,l)$, the size expansion index in the $k$-nearest neigbor, and the biased SVD. In Table 6, we can see that the combination result of $RMSE = 0.8974$ is the performance over 5% improvement to Cinematch result.

Table 6
Results by the Combination Methods

| | $RMSE$ | $riC$ (%) |
|---|---|---|
| $\mu_i$ & $\mu_j$ | 1.0149 | −6.64 |
| $es(j,l)$ & biased SVD | 0.8974 | 5.68 |

using Qualify

## 6 Discussion

We have quickly glanced at the results of the performance in various estimation methods. A serious but important thing in a movie recommendation system would be the biased scores; some are inclined to evaluate a high score in most times; some are heavy viewers; some often evaluate neutral scores. This means that we cannot believe the observed score as they are. Some correction may be required for accurate prediction. We have tried to incorporate the factor for this. The size expansion index in the $k$-nearest neighbor is the proposed method, and this worked very well if it used alone.

The SVD is known to be very efficient to predict the unknown scores. However, when new comers want to find his recommended movies, the system again find the larger matrix decomposition. On the contrary, the $k$-nearest neighbor will not require this. Thus, there is a trade-off between the SVD method and the $k$-nearest neighbor method.

To enhance the accuracy of prediction, the linear combination of the various predictors is reported to provide good performance somewhere. In some case, it works, but in some case, it does not. Finding the best combination of predictors is a kind of tuning work. The best result for the combination method by our proposed predictors in this paper far exceeds the result of Cinematch.

## 7 Concluding remarks

We have discussed the algorithms and their performances of the recommendation systems using the collaborative filtering in the case of the Netflix database. We have tried three cases. First one is the memory-based system, i.e., $k$-nearest neighbor method using the correlation coefficients. Using the size expansion index for the correlation coefficient proposed here, we have attained the better performance the the usual the correlation coefficient and its lower confidence limit use. This might be due to the user's score is somehow biased. The second method is the model-based system, i.e., the matrix decomposition method induced by the singular value decomposition. This kind of method provides very efficient results comparing to other methods like the $k$-nearest neighbor method. The third method is the combination method using the matrix decomposition, $k$-nearest neighbor, and others. In the Netflix database, the matrix decomposition method shows better performance than the $k$-nearrest neighbor; in addition, it is found that the combination method of the two methods provide a much better performance.

## References

[1] R. Bell and Y. Koren, Scalable collaborative filtering with jointly derived neighborhood interpolation weights, IEEE International Conference on Data Mining (ICDM'07), IEEE, 2007.

[2] R. Bell and Y. Koren, Improved neighborhood-based collaborative filtering, KDD-Cup and Workshop, ACM press, 2007.

[3] R. Bell, Y. Koren, and C. Volinsky, Modeling relationships at multiple scales to improve accuracy of large recommender systems. In KDD ' 07 (2007).

[4] R. Bell, Y. Koren, and C. Volinsky. The BellKor solution to the Netfix Prize. 2007b.

[5] J. Bennett and S. Lanning. The Netfix Prize. Proceedings of KDD Cup and Workshop, 2007.

[6] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl. GroupLens: applying collaborative fltering to Usenet news. Com- munications of the ACM, 40(3):77-87, 1997.

[7] Y. Koren, Factorization Meets the Neighborhood: a Multifaceted Collaborative

Filtering Model, Proc. 14th ACM Int. Conference on Knowledge Discovery and Data Mining (KDD'08), ACM press, 2008.

[8] C.-C. Ma. Large-scale collaborative fltering algorithms. Master ’ s thesis, National Taiwan University, 2008.

[9] B. N. Miller, I. Albert, S. K. Lam, J. A. Konstan, and J. Riedl. Movielens unplugged: experiences with an occasionally connected recommender system. In IUI ’ 03, 2003.

[10] A. Paterek. Improving regularized Singular Value Decomposition for collaborative fltering. Proceedings of KDD Cup and Workshop, 2007.

[11] R. Salakhutdinov and A. Mnih. Probabilistic Matrix Factorization. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, Advances in Neural Information Processing Systems 20, 1257-1264. MIT Press, Cambridge, MA, 2008.

[12] R. Salakhutdinov, A. Mnih, and G. Hinton. Restricted Boltzmann Machines for collaborative fltering. In Proceedings of the 24th international conference on Machine learning, 791-798, 2007.

[13] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl, Application of dimensionality reduction in recommender system - a case study, WEBKDD ’ 2000.

[14] S. Takimoto and H. Hirose, Recommender Algorithm using the Matrix Decomposition: An Application to NetFlix Prize, 22nd Conference of Japanese Society of Computational Statistics, pp.17-20 (2008)

[15] S. Takimoto and H. Hirose, Recommendation Algorithm using the Matrix Decomposition, The 61nd Joint Conference of Electrical and Electronics Engineers in Kyushu, 08-1P-11 (2008)

[16] S. Takimoto and H. Hirose, Collaborative Filtering using the Weighted Mean: An Application to the NetFlix Data and Its Consideration, The 60nd Joint Conference of Electrical and Electronics Engineers in Kyushu, 11-1A-11 (2007)

[17] S. Zhang, W. Wang, J. Ford, F. Makedon, and J. Pearlman. Using Singular Value Decomposition approximation for collaborative fltering. Seventh IEEE International Conference on E-Commerce Technology, 2005. CEC 2005., pages 257f264, July 2005.