KDD Cup 2007 Task 1 Winner Report*

Miklós Kurucz András A. Benczúr Tamás Kiss
István Nagy Adrienn Szabó Balázs Torma
Data Mining and Web search Research Group, Informatics Laboratory
Computer and Automation Research Institute of the Hungarian Academy of Sciences
{realace, benczur, kisstom, iscsi, aszabo, torma}@ilab.sztaki.hu

ABSTRACT

KDD Cup 2007 focuses on predicting aspects of movie rating behavior. We present our prediction method for Task 1 "Who Rated What in 2006" where the goal is to predict which users rated which movies in 2006. We use the combination of the following methods, listed in the order of their accuracy:

- The predicted number of ratings for each movie based on time series analysis, also using movie and DVD release dates and movie series detection by the edit distance of the titles.
- The predicted number of ratings by each user by using the fact that ratings were sampled proportional to the margin.
- The low rank approximation of the 0–1 matrix of known user–movie pairs with rating.
- Prediction by using the movie–movie similarity matrix.
- Association rules obtained by frequent sequence mining of user ratings considered as ordered itemsets.

By combining the predictions by linear regression we obtained a prediction with root mean squared error 0.256. The first runner up result was 0.263 while a pure all zeroes prediction already gives 0.279, indicating the hardness of the task.

Categories and Subject Descriptors

J.4 [**Computer Applications**]: Social and Behavioral Sciences; G.1.3 [**Mathematics of Computing**]: Numerical Analysis—*Numerical Linear Algebra*

General Terms

data mining, recommender systems

Keywords

singular value decomposition, item-item similarity, frequent sequence mining

*This work was supported by the eScience Regional Knowledge Centre, Hungary, a Yahoo Faculty Research Grant and by grant *ASTOR* NKFP 2/004/05

1. INTRODUCTION

We present our first place winner method for Task 1 "Who Rated What in 2006". The task was to predict the probability that a user rated a movie in 2006 (with the actual date and rating being irrelevant) for a given list of 100,000 user–movie pairs of the Netflix Prize data set [2] . The users and movies are drawn from the Prize data set, i.e. the movies were released (or at least received ratings) before 2006 and the users also gave their first rating before 2006. In addition, none of the pairs selected for the KDD Cup task were rated in the training set. We give a detailed description of the sampling method in Section 2.1 since it gives information that we use for the prediction.
Our method is summarized as follows:

1. The combination of separate estimates for the number of ratings for each movie and each user by a naive user–movie independence assumption. Our movie ratings prediction uses time series analysis aligned with movie and DVD release dates from the IMDB and `videoeta.com` databases (Section 2.2). User rating numbers are on the other hand reconstructed from sample margins(Section 2.1).
2. The implementation of an SVD (Section 3.1) and an item-item similarity (Section 3.2) based recommender as well as association rule mining (Section 3.3).
3. Method fusion by using the machine learning toolkit Weka [11] (Section 4).

We use the *root mean squared error*

$$\text{rmse}^2 = \sum_{ij \in R} (w_{ij} - \hat{w}_{ij})^2$$

as the single evaluation measure, where w_{ij} is a 0–1 matrix with value 1 if user i gave rating for movie j, and $\hat{w}_{ij}$ is the predicted value in the range of 0 to 1 given by the recommender system.
The use of rmse implies that we actually predict the probability of the existence of a rating: for a random variable that has value 1 with probability p and 0 otherwise, the rmse of the prediction of its value is minimized by p. If we correctly guess the number of ratings 7,804 in the 100,000 sample, then this method results in an rmse of 0.268 that would reach 5-6th place in the Cup, indicating the hardness of correctly predicting this value. Notice however that the rmse of 0.279 of the trivial all zeroes prediction would also reach 10-13th place and remains not very far from the winner rmse.

The experiments were carried out on a cluster of 64-bit 3GHz P-D processors with 4GB RAM each and a multiprocessor 1.8GHz Opteron system with 20GB RAM. With certain variation depending on parameter settings, the running time range was 15 minutes for SVD, few hours for the item-item similarity based recommender, few days for frequent patterns and finally few minutes for linear regression. Mining frequent patters turned out most time consuming. The current version of the paper contains a more thorough experimentation with frequent patters that we could not afford for the KDD Cup competition due to CPU time limitations.

The rest of the paper is organized as follows. In Section 2 we predict the marginals and use a naive user–movie independence assumption. Then in Section 3 we describe our three data mining methods: the singular value decomposition, an item-item similarity based recommender and association rule mining. The predictors are combined by linear regression; results are described in Section 4.

2. BASE PREDICTION BY USER-MOVIE INDEPENDENCE ASSUMPTION

In this section we assume independence between the users and the movies for a "Who Rated What" prediction. Since we are looking for the probability of the existence of a rating, we start out with the marginal probabilities and use their product. Notice that even the marginals form a highly non-trivial task that includes the "How Many Ratings" Task 2 of the KDD Cup as subproblem. We describe our approaches separately for users and movies in the next two subsections.

The prediction is given by the product of the marginals scaled so that they sum up to R, the predicted total number of actual ratings for the Task 1 movies. Given a prediction N_u for the number of ratings of user u and N_m of movie m, we use

$$p_{um} = \frac{N_u \cdot R}{M \cdot R} \cdot R = N_u \cdot N_m / R$$

as the naive prediction for the user–movie pair u, m.

Other than the values of N_u and N_m described in the next two subsections, the prediction also depends on the choice of R. We used an estimate of 10,000,000 that we obtained by generating a sample user-movie pairs by the exact same procedure of Task 1, except for using the marginals of the time range between November 2004 and October 2005 and discarding pairs with rating before October 2004. We obtained a sample of 100,000 user-movie pairs, out of which 6,800 was actually rated in the given one year time period. By using the known marginals we observed $R = 10,000,000$ gave p_{um} values that summed up to 6,800.

2.1 How Many Ratings by User

For predicting the number of ratings of a given user we solely relied on the fact that the sample used for the "Who Rated What" task was taken proportional to the number of ratings of the user. We begin with a detailed description of the sampling method. The 100,000 user–movie pairs were formed by drawing the movies from the 6822 movies selected for Task 1 and the users from the Prize data set, i.e. from those who gave their first rating before 2006. Pairs that corresponded to ratings in the existing Netflix Prize dataset were discarded; we ignored this fact in our prediction.

The key difficulty in obtaining the user rating numbers N_u is the small probability of including a single user which implies a high probability of underestimating a single user. In particular we may assume that the actual rating number is non-zero for almost all users that do not appear in the sample and hence an upward correction is necessary.

We correct the number of times a user is included in the sample by an estimated standard deviation in order to obtain N_u as follows. The expected value of the number times the user u is included in the sample is equal to $n_u = 100,000 \cdot N_u/R$. Since N_u/R is very small, its standard deviation is approximately $\sqrt{N_u/R}$ and hence the standard deviation of n_u is $\sqrt{N_u/R} \cdot \sqrt{100,000} = \sqrt{n_u}$. Except for a single user with a very large number of occurrences in the sample, all users occur at most 20 times. We considered the most frequent user an outlier and based on the next maximum value 20 we assume the standard deviation to be uniformly below 5. Hence we add 4 to the number of appearances in the sample for all users (including those who do not appear at all) and obtain the estimate $\hat{N}_u$ by normalizing to sum up to the estimated total number of ratings R; we use $R = 10,000,000$.

We give another justification of the choice for correcting the observed appearances in the sample upwards by 4. Notice that since the probability of user u not being included in the sample is approximately $\exp(-100,000 \cdot N_u/R)$, this probability is below 2% for a user with expected number of appearances at least 4.

2.2 How Many Ratings by Movie

The task of predicting the number ratings by the users is the same as Task 2 "How Many Ratings in 2006" of the KDD Cup 2007, but for a different set of movies. The task is to estimate the number of additional ratings for a given movie by users from the Netflix Prize dataset. The set of movies that appeared (or at least received ratings) before 2006 were split randomly into two sets, one per task, resulting in 6822 movies for Task 1 and 8863 for Task 2. Unlike the best performing teams for Task 2 who used the Task 1 movies for training [9], here we did not use the fact that the Task 1 user–movie pairs were sampled proportional to the margins as described in the previous section.

We predict ratings for a given movie by analyzing the time series of its ratings as well as using IMDB movie release and `videoeta.com` DVD release dates for the movie and its likely series continuation releases. Movie titles across different databases as well as series titles were detected by computing the Damerau-Levenshtein distance [8] between the titles by giving more weight to the prefix of the title and less weight for complete words that are missing. Stop word removal was also performed first; an extended stop word list included phrases such as "the best of", "the adventures of" etc.

Our prediction is the sum of a base estimated from previous ratings and additional ratings for predicted related release events. We observe an increase in the amount of ratings at and after the dates of related movie and DVD releases, hence if such an event is assumed to happen, then the number of ratings will be estimated higher accordingly. The increase in this case will be proportional to the baseline prediction. The baseline is the total number of ratings of the movie in the period of November 2004 and October 2005. This amount is multiplied by a decay factor, another factor for the DVD release event, and a third factor for series continuation release events for the movie. The factors are trained for year

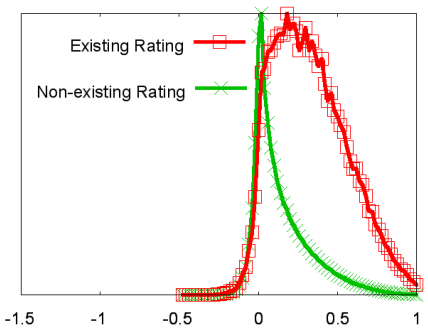

Figure 1: The distribution of the 10-dimensional approximation of the user–movie matrix for pairs with, respectively without ratings.

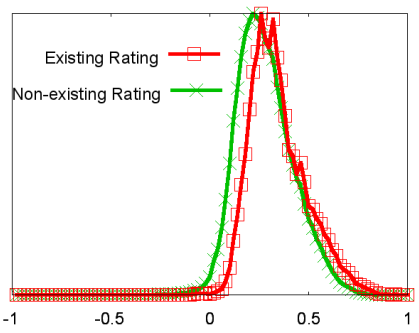

Figure 2: The distribution of the item-item similarity based prediction for user–movie pairs with and without ratings for a similarity top list of size $K = 5$.

2005 as the validation period. Movies that appeared in the second half of 2005 were also corrected upwards.

3. DATA MINING METHODS

3.1 SVD based recommendation

For training we use the full 0–1 matrix of all known ratings; the rank k approximation of the matrix yields our prediction. The Singular Value Decomposition (SVD) of a rank ρ matrix W is given by $W = U^T \Sigma V$ with U an $m \times \rho$, Σ a $\rho \times \rho$ and V an $n \times \rho$ matrix such that U and V are orthogonal. By the Eckart-Young theorem [4] the best rank-k approximation of W with respect to the Frobenius norm is

$$||W - U_k^T \Sigma_k V_k||_F^2 = \sum_{ij} (w_{ij} - \sum_k \sigma_k u_{ki} v_{kj})^2.$$

where U_k is an $m \times k$ and V_k is an $n \times k$ matrix containing the first k columns of U and V and the diagonal Σ_k containing first k entries of Σ.

While the Frobenius norm is simply the rmse of the prediction for the existence of the rating if the of user–movie pairs are selected uniform at random, this is not true for the sampling method used for producing the Task 1 pairs as described in detail in Section 2.1. If the probability that the pair formed by user i and movie j is selected in the sample is p_{ij}, then we have to minimize

$$\sum_{ij} p_{ij} \cdot (w_{ij} - \sum_k \sigma_k u_{ki} v_{kj})^2 \quad = \quad \sum_{ij} (\sqrt{p_{ij}} \cdot w_{ij} - \sqrt{p_{ij}} \cdot \sum_k \sigma_k u_{ki} v_{kj})^2, \qquad (1)$$

which is minimized similarly by the SVD of $\sqrt{p_{ij}} \cdot w_{ij}$, divided pointwise by $\sqrt{p_{ij}}$.

In our implementation we used the Lanczos code of `svdpack` [3] that turned out both fastest and most precise in our recent experiments [7; 6]. Since we observed overfitting for larger number of dimensions [6] we used the 10-dimensional approximation of the scaled matrix as in equation (1) where the p_{ij} values are those obtained by the method of Section 2. The difference between the distribution of the predicted value for the actual ratings, respectively no-ratings is seen in Fig. 1.

3.2 Item-item similarity based recommendation

Our item-item based recommender computes the adjusted cosine similarity [10] based not just on the existence of the ratings w_{ij} as in the rest of the paper, but also its value r_{ij} in the range of $1 \ldots 5$:

$$\text{sim}_{jj'} = \frac{\sum_u (r_{uj} - \bar{r_u}) \cdot (\sum_u (r_{uj'} - \bar{r_u})}{\sqrt{\sum_u (r_{uj} - \bar{r_u})^2} \cdot \sqrt{\sum_u (r_{uj'} - \bar{r_u})^2}}$$

where $\bar{r}_u$ is the average of the ratings of user u and summations are for all users u who rated both j and j'. Since the number of such users $n_{jj'}$ can sometimes be very small, we replaced the above sim value by its lower 95% confidence interval of the adjusted cosine similarity obtained by Fisher's r-to-z transform [5]

$$z = (\ln(1 + r) - \ln(1 - r))/2,$$

using $\sqrt{n_{jj'} - 3}^{-1}$ for the standard deviation.

An unrated movie j is recommended to a given user i based on the weighted average of the nearest K movies j' to j rated by the user. Weights are defined by the lower confidence interval of sim as above. We chose a value of $K = 5$ by observing the difference of the prediction for user–movie pairs with and without ratings in Fig. 2. In this choice we have to keep in mind that larger values introduce noise, however if we start out with R known ratings, we may only possibly give predictions for $R \cdot K$ user-movie pairs that remain very sparse within the rating matrix if K is small.

3.3 Association Rules in Sequences

We used association rules for Task 1 via frequent sequences within the sets of movies rated by a user ordered by the time of the rating. In order to compute the support supp($m_1, \ldots, m_s$) we had to impose very strong restriction due to CPU time constraints that left us with rules that matched only 20,000 of the 100,000 user-movie pairs. Given the support of all frequent sequences, we computed all association rules that apply to one of the 6822 Task 1 movies. These association rules are of form

$$m_1, \ldots, m_s \rightarrow m$$

where both frequencies supp($m_1, \ldots, m_s$) and supp($m_1, \ldots, m_s, m$) are above the minimum support. The confidence of

this rule is

$$\mathrm{conf}(m_1, \ldots, m_s \to m) = \frac{\mathrm{supp}(m_1, \ldots, m_s, m)}{\mathrm{supp}(m_1, \ldots, m_s)}.$$

Since only a small number of users are involved, we may simply match all association rules brute force to the Task 1 user-movie pairs; a trie-based approach can speed this up if larger scale evaluation is required. Finally we use the maximum of the confidence of all rules that fit the user–movie pair as prediction.

We describe our original set of restrictions that we used within an APRIORI [1] implementation for frequent sequence mining. We discarded all movies that received more than 50,000 ratings and all users that gave more than 3,000 ratings in the Prize data set. We added the condition that in a frequent sequence the number movie ratings must not differ by more than a factor of 4; since this property is monotonic, we could implement it as a filter in the two-element candidate generation step of the APRIORI algorithm. In addition we also imposed this condition later, i.e. we only formed candidate $(m_1, \ldots, m_s, m_{s+1})$ if

$$1/4 < \frac{\mathrm{supp}(m_{s+1})}{\mathrm{supp}(m_1, \ldots, m_s)} < 4.$$

We counted the frequency of sequences restricted to fit in time windows of 30 days; we allowed all permutations of movies that received their ratings from the user on the same day. We set the minimum support to 50. By manual investigation we found that the most restrictive rule was the 30 day time window restriction that resulted in the low number of matches with the Task 1 pairs.

4. CONCLUSION

We combined the four predictions of the naive independence, SVD, item-item correlation and association rule based approaches by the linear regression method of the machine learning toolkit Weka [11]. We obtained the equation

$$\begin{aligned} 0.5533 \cdot p_{um} \quad &+ \\ 0.029 \cdot \mathrm{correlation} \quad &+ \\ 0.1987 \cdot \mathrm{SVD} \quad &+ \\ -0.0121 \cdot \mathrm{assoc\_rules} \quad &- \quad 0.0042 \end{aligned}$$

as the final prediction that reaches rmse 0.256, gaining 0.007 over the first runner up and 0.023 over a pure all zeroes prediction.

In a preliminary version we used an alternate, seemingly less sophisticated method to combine predictions that performed only marginally worse, still reaching first place in the competition and may be of possible further use for combining predictions. Next we sketch this method. As seen in Figs. 1 and Fig. 2, for a given predicted value x we can count (in the training set of Year 2005 data) the fraction of actual ratings with the predicted value x. By using a binning of step 0.1 we made one prediction value for each range given by the above fraction. For ranges where the data was sparse we used manual correction. Since the number of sparse bins (as well as all bins) is small and they effect a small number of user-movie pairs, there is no need for a more refined method. The final prediction for a user-movie pair arises as the maximum of all bin predictions that the pair belongs to.

The main lesson learned by solving this task is probably the fact that very different data mining techniques catch very similar patterns in the data that makes it increasingly difficult to improve prediction quality beyond certain point. We stress here however that tuning association rule based prediction is a very time (and CPU) consuming task and our method is far from being the result of an exhaustive experimentation. We also demonstrated the hardness of the task by showing how well trivial estimates perform, just marginally outperformed by our result and beating most of the contestant teams. Just as it is the case for Task 2 [9], the information leaked by the sampling method used to generate the test user–movie pairs could be used. While without reconstructing marginals it would likely have been impossible to come within first three.

5. REFERENCES

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In J. B. Bocca, M. Jarke, and C. Zaniolo, editors, *International Conference on Very Large Data Bases (VLDB)*, pages 487–499. Morgan Kaufmann, 12–15 1994.

[2] J. Bennett and S. Lanning. The netflix prize. In *KDD Cup and Workshop in conjunction with KDD 2007*, 2007.

[3] M. W. Berry. SVDPACK: A Fortran-77 software library for the sparse singular value decomposition. Technical report, University of Tennessee, Knoxville, TN, USA, 1992.

[4] G. H. Golub and C. F. V. Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, 1983.

[5] H. Hotelling. New light on the correlation coefficient and its transforms. *Journal of the Royal Statistical Society B*, 15:193–225, 1953.

[6] M. Kurucz, A. A. Benczúr, and K. Csalogány. Methods for large scale svd with missing values. In *KDD Cup and Workshop in conjunction with KDD 2007*, 2007.

[7] M. Kurucz, A. A. Benczúr, K. Csalogány, and L. Lukács. Spectral clustering in telephone call graphs. In *WebKDD/SNAKDD Workshop 2007 in conjunction with KDD 2007*, 2007.

[8] G. Navarro. A guided tour to approximate string matching. *ACM Comput. Surv.*, 33(1):31–88, 2001.

[9] S. Rosset, C. Perlich, and Y. Liu. Making the most of your data: Kdd cup 2007 how many ratings winners report. In *KDD Cup and Workshop in conjunction with KDD 2007*, 2007.

[10] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 285–295, New York, NY, USA, 2001. ACM Press.

[11] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann, second edition, June 2005.