A Mixture Imputation-Boosted Collaborative Filter

Xiaoyuan Su
Computer Science and Engineering
Florida Atlantic University
Boca Raton, FL 33431, USA
xsu@fau.edu

Taghi M. Khoshgoftaar
Computer Science and Engineering
Florida Atlantic University
Boca Raton, FL 33431, USA
taghi@cse.fau.edu

Russell Greiner
Department of Computing Science
University of Alberta
Edmonton, AB, T6G 2E8, Canada
greiner@cs.ualberta.ca

Abstract

Recommendation systems suggest products to users. Collaborative filtering (CF) systems, which base those recommendations on a database of previous ratings by various users and products, have been proven to be very effective. Since this database is typically very sparse, we consider first *imputing* the missing values, then making predictions based on that completed dataset. In this paper, we apply several standard imputation techniques within the framework of *imputation-boosted collaborative filtering* (IBCF). Each technique passes that imputed rating data to a traditional Pearson correlation-based CF algorithm, which uses that information to produce CF predictions. We also propose a novel *mixture IBCF algorithm*, IBCF-NBM, that uses either naïve Bayes or mean imputation, depending on the sparsity of the original CF rating dataset. Our empirical results show that IBCFs are fairly accurate on CF tasks, and that IBCF-NBM significantly outperforms a representative hybrid CF system, content-boosted CF algorithm, as well as other IBCFs that use standard imputation techniques.

1. Introduction

A collaborative filtering *(CF)* system predicts which items a new user might like based on a dataset that specifies how each user u_j has rated each item i_k ; cf., Figure 1(a). Here, each of these ratings $r_{u,i}$ is an explicit indication on a 1-5 scale (as opposed to an implicit indication, such as a purchase or a click-through). Note this matrix is sparse, as users typically do not rate every item.

The (user-based) Pearson correlation-based *CF* (*PearsonCF*) algorithm is a "memory-based *CF* algorithm" [2] that bases its predictions on the similarities between the users who rate common items; see Section 2. The underlying assumption here is that users who rate some items similarly, will probably rate other items similarly as well. As this computation involves only a small percentage of the total number of items (recall the matrix is sparse), these systems can deal even with large rating matrices (Figure 1(a)). Unfortunately, the predictive performance of *PearsonCF* systems tends to degrade quickly as the rating data become more sparse.

This motivates model-based *CF* techniques, such as Bayesian belief nets *CF* algorithms [11][16], since these techniques can often provide relatively accurate predictions even on sparse data. However, empirical evidence shows that their performance improvement over the *PearsonCF* is not significant [11][16][18].

One way to improve predictive accuracy for *CF* tasks is to use additional content information – e.g., information about the users and/or the items, beyond the pure ratings in the database. This motivated Melville et al. [10] to produce a type of hybrid *CF* method, *content-boosted CF*, that uses a learned *naïve Bayes* (NB) classifier on content data to fill in the missing values to create a *pseudo rating matrix* (see Figure 1(b) below), then applies a *weighted PearsonCF* algorithm to this matrix to produce *CF* predictions. Note, however, that this *content-boosted CF* depends on additional content information that is often not available.

Figure 1: (a) original rating data, (b) "pseudo rating data" (including imputed data)

	I_1	I_2	I_3	I_4	I_5
U_1	2	2	4	3	
U_2		?	4	3	
U_3		1			3
U_4		3	3	3	3
U_5	1			3	
U_6		4	4	2	

	I_1	I_2	I_3	I_4	I_5
U_1	2	2	4	3	*3*
U_2	*2*	*2*	4	3	*3*
U_3	*3*	1	*3*	*3*	3
U_4	*3*	3	3	3	3
U_5	1	*2*	*4*	3	*2*
U_6	*2*	4	4	2	*3*

In general, the process of filling in the missing values is called *imputation* [13]. A subsequent *CF* subroutine can then produce prediction using the resulting pseudo rating data, rather than the original rating data.

While that imputation process used *auxiliary information*, others [17] have explored imputation techniques that use *only* the *original rating data* (Figure 1(a)) to produce the pseudo-rating data (Figure 1(b)). Each of these imputation-boosted collaborative filtering (*IBCF*) algorithms uses an imputation technique (which can be a machine learning classifier) to impute the missing rating data to create a pseudo rating matrix, which is then passed to the traditional *PearsonCF* predictor to produce final recommendations. *Su et al.* [17] previously demonstrated that these *IBCF* systems effectively boost predictive performance of *CF*, and that the *IBCF* using

naïve Bayes (*NB*) imputation is one of the most effective such systems.

In this paper, we empirically evaluate the effectiveness of several *IBCF* systems -- some using standard imputation techniques such as mean imputation (*IBCF-mean*), linear regression imputation (*IBCF-LinR*), and predictive mean matching (*IBCF-PMM*), and others using various machine learning classifiers, such as *NB* (*IBCF-NB*). As we found that *IBCF-NB* has the best predictive performance for relatively dense data, while *IBCF-mean* is one of the best performers for extremely sparse data, we thus propose a new mixture *IBCF* algorithm, *IBCF-NBM*, that uses *IBCF-NB* for dense data and *IBCF-mean* for sparse data.

To evaluate our systems, we work on the real-world data from *MovieLens* [12], and evaluate performance using *RMSE* (root mean square error)

$$RMSE = \sqrt{\frac{1}{n}\sum_{\{u,i\}}(p_{u,i} - r_{u,i})^2} \quad (1)$$

where n is the total number of ratings over all users, $p_{u,i}$ is the predicted rating for user u on item i, and $r_{u,i}$ is the actual rating. A *CF* method is considered better if it has a lower *RMSE* value. (Of course, we can only evaluate this on the subset of entries that had observed user ratings – ie, only when $r_{u,i}$ is filled in. Note *MovieLens* includes 100,000 such entries.)

The rest of this paper is organized as follows: Section 2 presents the related work. Section 3 provides the framework of our work, and Section 4 presents experimental design and results.

2. Related Work

The *PearsonCF* algorithm, a representative memory-based *CF* algorithm, involves the following steps:

(1: Preprocessing) Calculate the *Pearson* correlation value $w_{u,v}$ between each pair of users u and v :

$$w_{u,v} = \frac{\sum_{i\in I}(r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i\in I}(r_{u,i} - \bar{r}_u)^2}\sqrt{\sum_{i\in I}(r_{v,i} - \bar{r}_v)^2}} \quad (2)$$

where the $i \in I$ summations are over the items $I = I(u,v)$ that both users u and v have rated and $\bar{r}_u$ is the average rating of the co-rated items of user u.

(2: Run-time) Predict the rating that user a will assign to item i as the weighted average of all user ratings on i as

$$p_{a,i} = \bar{r}_a + \frac{\sum_{u\in U(i)}(r_{u,i} - \bar{r}_u)\cdot w_{a,u}}{\sum_{u\in U(i)}|w_{a,u}|} \quad (3)$$

where the $u \in U(i)$ summations are over all the users who have rated the item i [2].

As such *PearsonCF* systems can typically produce accurate *CF* predictions even when there are large numbers of users and items, they are commonly used in deployed commercial systems. Unfortunately, *PearsonCF* can perform poorly when the rating data is very sparse.

(See [1] for a survey of other relevant recommendation systems.)

3. Framework

3.1 Imputation-Boosted CF (IBCF)

The selection of an imputation technique should be based on an analysis of why the data are missing. Many conventional *CF* algorithms implicitly assume "*MAR*" missingness (missing at random), which means the missingness depends on observed data but not on the unobserved data [9]. As an example, the rule: if users like the film "*Snow White*", they typically do not submit ratings for the movie "*Friends*", in which the missingness of the movie *"Friends"* depends on the observed ratings for the movie "*Snow White*". Some imputation algorithms assume the data are *MCAR* (Missing Completely at Random, eg., the independent and identically distributed missing ratio is 0.6) or *NMAR* (not missing at random, the missingness depends on the missed value itself, eg., if a user does not like the movie "*Friends*", he will not rate it); running these algorithms on the data that are actually *MAR* can produce biased estimates [15]. When the local pattern of the missing data for a subset of samples violates the global pattern assumption (for all samples), the results will again be biased. There are several strategies for reducing the biases: when only the dependent variable is missing, then conventional machine learning classifiers can be used. This motivates us to use some standard imputation techniques, such as predictive mean matching (*PMM*), that are effective in reducing the estimation biases [6][7]; see Section 3.4.

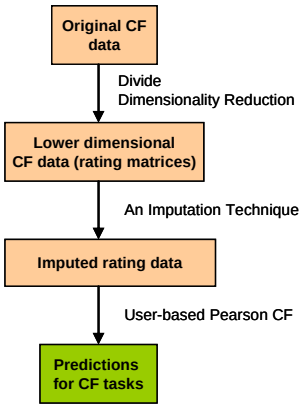

Figure 2. Framework of imputation-boosted collaborative filtering (IBCF)

The main steps of each imputation-boosted *CF (IBCF)* algorithm are: (1) divide the original, large rating dataset into reasonably-sized subsets; (2) apply some imputation technique (which could be a machine learning classifier) to generate the "filled-in" pseudo rating data; (3) apply a user-based *PearsonCF* to the imputed data to produce predictions. (Note that the *PearsonCF* prediction on *(u,i)*,

$p_{u,i,}$ depends on the values of the matrix *other* than *(u,i)*, and that this $p_{u,i,}$ value may be different from the imputed value.) Figure 2 provides a high-level outline of the framework. The rest of the section introduces the various *IBCF* algorithms.

3.2 IBCF Using Mean Imputation

IBCF-mean (*IBCF using mean imputation*) uses mean imputation to produce the pseudo rating data. As our datasets have more users than items, we use mean imputation on *items*, which replaces each missing value *(u,i)* in the rating matrix with the mean of the observed ratings on this item *i,* that is $(1/k)\sum_{u \in U(i)} r_{u,i,}$ where *k* is the total number of users $u \in U(i)$ who have rated item *i*. Mean imputation is one of the simplest imputation technique. Unfortunately, it can distort the shape of distributions (by creating a spiked distribution at the mean in frequency distributions, which attenuates the correlation of the associated item with others), and it also reduces the variance of the predictions.

3.3 IBCF Using Linear Regression Imputation

IBCF-LinR (*IBCF using linear regression*) imputes values using linear regression.

To explain the process, consider estimating the value of $r(U_2,I_2)$ from the data shown in Figure 1(a). Notice U_2 has ratings for I_3 and I_4. The *LinR* imputer would first find all other users who have ratings for these items, and also I_2 (the value we want to predict); this identifies users $U(I_2,I_3,I_4)=\{U_1,U_4,U_6\}$. *LinR* then seeks coefficient β_0, β_3, and β_4 such that $r(U_i,I_2)= \beta_0+\beta_3 r(U_i,I_3)+\beta_4 r(U_i,I_4)$. Using this data subset $\{r(U_j, I_k)|j \in \{1,4,6\}, k \in \{2,3,4\}\}$, the best-fit line has $\beta_0=12$, $\beta_3=-1$ *and* $\beta_4=-2$. *LinR* now computes its prediction for $r(U_2,I_2)$ as $p_{2,2}=\beta_0+\beta_3 r_{2,3}+\beta_4 r_{2,4}=2$. Here, there were 3 equations (for U_1, U_4, U_6) and 3 unknowns $\{\beta_0, \beta_3, \beta_4\}$, which produces this unique solution. In other situations, there might be more equations than unknowns, we use the mean of the estimates as the imputation. *IBCF-LinR* returns values rounded to the nearest integers, and replaces predictions smaller than 1 with 1, and those larger than 5 with 5. This procedure is repeated for each missing rating value.

In general, regression imputation is better than mean imputation as it can produce consistent estimates (i.e., the estimates will be unbiased with large sample sizes if the missingness is *MAR*). However, it assumes no error of estimation around the regression line, and therefore underestimates the variance of the estimates [5].

3.4 IBCF Using Predictive Mean Matching Imputation

IBCF-PMM gives *CF* predictions on the imputed data from predictive mean matching (*PMM*), which is a state-of-the-art imputation technique [6][7][19].

PMM imputes missing values by matching completely observed units (donors) with incomplete units (recipients), using a distance function, then transferring values from the donor to recipient. For *CF* data with numerical variables and arbitrary patterns of missing values (mostly *MAR* [9]), *PMM* does the following:

1) Use the *EM* algorithm ("expectation maximization") [4] to estimate the parameters of a multivariate Gaussian distribution, using all the available data (including observed and unobserved values).

2) Based on the estimates from *EM*, for each incomplete record $y_{miss,i,}$ (called "recipient", this corresponding to an unspecified rating of user u_i) compute the predictions of the missing items conditioned on the observed ones $y_{obs,i}$ (these are the rating that u_i has specified). The same predictive means (i.e., corresponding to the same missing pattern) are computed for all the complete observations $Y_{obs,i}$ (donors).

$$\hat{\mu}_i = E(Y_{miss,i} \mid Y_{obs,i}) \tag{5}$$

3) Each recipient is matched to the donor having the closest predictive mean with respect to the *Mahalanobis distance* [8] defined through the residual covariance matrix *S* from the regression of the missing items on the observed ones.

$$d(i,j) = \sqrt{(\hat{\mu}_i - \hat{\mu}_j)^T S^{-1}_{Y_{miss,i}|Y_{obs,i}} (\hat{\mu}_i - \hat{\mu}_j)} \tag{6}$$

4) Impute the missing values in each recipient by transferring the corresponding values from its closest donor.

In the example of Figure 1(a), based on the estimates from *EM*, for the incomplete record (recipient) user U_2, *PMM* computes the predictions of the missing items, $y_{miss,2}=\{r_{2,1}, r_{2,2}, r_{2,5}\}$, conditioned on the observed ones $y_{obs,2}=\{r_{2,3}, r_{2,4}\}$. The same predictive means are computed for all the complete observations $Y_{obs,i}$ (donors), e.g., $Y_{obs,i}=\{r_{i,2}, r_{i,3}, r_{i,4}\}$ if we want to make prediction for $r_{2,2}$ based on $y_{obs,2}$. The recipient $r_{2,2}$ is then matched to the donor that has the closest predictive mean in terms of the *Mahalanobis distance* and is imputed by transferring the corresponding values from its closest donor (e.g., $Y_{obs,1}=\{r_{1,2}, r_{1,3}, r_{1,4}\}$).

3.5 IBCF Using Naïve Bayes Classifier

In general, given evidence $\boldsymbol{E=e}$, a probabilistic classifier will compute $P(C=c, \boldsymbol{E=e})$ for each class value *c*, then return $c^*(\boldsymbol{E=e}) = argmax_c\ P(C=c,\boldsymbol{E=e})$.[1] A naïve Bayes (*NB*) classifier embodies the assumption that the attributes

[1] While these computations are typically worded as P(c|e) rather than P(c,e), notice all we care about is the argmax, which returns the same value, either way.

are independent with each other, given the class, which means $P(C=c, \mathbf{E}=\mathbf{e}) = P(C=c) \prod_k P(\mathbf{E_k}=\mathbf{e_k} \mid \mathbf{C}=\mathbf{c})$.

The imputation component of *IBCF-NB* (*IBCF* using naïve Bayes) is: for each item $i=1..n$, train the *NB* classifier on the pure rating data of columns $\{1, \ldots i-1, i+1, \ldots n\}$ (dealing only with the rows u that specify a value for (u,i)) to produce a classifier for item i, which computes the probability that this rating is $c \in \{1,2,\ldots, 5\}$. This means computing empirical estimates of missing ratings of each class column i using the learned *NB* classifier on the observed ratings on that column together with ratings on other columns [17]. So for Figure 1(a), the *NB* classifier associated with column I_1, $NB_1(I_2, I_3, I_4, I_5)$, would be trained based on the rows indexed by U_1 and U_5 (as these are the users who gave a score for item I_1), and would then predict the rating for the remaining row – say (U_6, I_1) based on the evidence $\{i_2=4, i_3=4, i_4=2\}$. We would also train the $NB_2(I_1, I_3, I_4, I_5)$ classifier for the 2nd column, then $NB_3(I_1, I_2, I_4, I_5)$ and so forth.

3.6 IBCF-NBM: a Mixture IBCF

We propose a mixture *IBCF* algorithm, which is based on the observations that *IBCF-NB* (*IBCF* using the machine learning classifier *NB)* typically performs well on dense rating data, but its performance degrades for highly sparse data; and that *IBCF* using more standard imputation techniques, such as mean imputation, can produce more accurate *CF* predictions for extremely sparse data than the *IBCFs* using machine learning classifiers.

Under the divide and conquer scenario (described in Section 3.7 below), we divide the large original rating data into small datasets, based on their sparsity. *IBCF-NBM* can therefore use this "sparcity score" to decide which imputation method to use: it applies *IBCF-NB* for denser datasets (with sparsity<95%), and *IBCF-mean* for sparser datasets (with sparsity≥95%). Note this sparsity threshold is data dependent: it is determined by the point where *IBCF-NB* and *IBCF-mean* change their lead in terms of *RMSE* performance.

Now consider a missing rating (u,i). If this item I has rated by a large number of users (more than 5% of the users, which corresponds to at least 48 out of 943 users), then *IBCF-NBM* will use *NB* to impute values; otherwise, it uses simple mean imputation. (The next section discusses how we evaluate this approach.)

3.7 Divide and Conquer

Here, we evaluate our *IBCF* approach. Working on the *MovieLens* data [12], to simulate the *IBCF-NBM* situation, we partition the original data into subsets, based on the number of users who rated each movie. That is, we first re-number the movies based on their respective number of user ratings

$user\#(m_1) > user\#(m_2) > \ldots\ldots > user\#(m_{1682})$

where $user\#(m_i)$ is the number of users that have rated the movie m_i. We use this to divide the original *MovieLens* data into 20 subsets, each of which have all 943 users and some set of 65 movies, with the condition that each of the movies has at least five users rating them.

Data_1 has all 943 users, and movies $\{m_1, \ldots, m_{65}\}$

Data_2 has all 943 users, and movies $\{m_{66}, \ldots, m_{130}\}$

……

Data_20 has all 943 users, and movies $\{m_{1236}, \ldots, u_{1300}\}$

Each of the remaining 382 movies (out of 1682) has been rated by five or fewer users, corresponding to a total of 958 ratings (0.958% of the original 100,000 ratings). We use a default voting value of 3 (rounded integer of the average value of the ratings) as their predictions and integrate them into the overall performance evaluation.

Table 2. Divide and conquer the MovieLens data

data	rating #	per-cent	sparsity %	data	rating #	per-cent	sparsity %
d_1	21781	21.8	64.5	d_11	2718	2.72	95.6
d_2	14229	14.2	76.8	d_12	2293	2.29	96.3
d_3	11055	11.1	82.0	d_13	1883	1.88	96.9
d_4	8953	8.95	85.4	d_14	1560	1.56	97.5
d_5	7490	7.49	87.8	d_15	1266	1.27	97.9
d_6	6093	1.09	90.1	d_16	1024	1.02	98.3
d_7	5119	5.12	91.6	d_17	811	0.81	98.7
d_8	4343	4.34	92.9	d_18	646	0.65	98.9
d_9	3740	3.74	93.9	d_19	515	0.52	99.2
d_10	3111	3.11	94.9	d_20	412	0.41	99.3
				other	958	0.96	99.9

Each of these subsets is reasonably sized for most regular imputation techniques and machine learning algorithms — viewing the 943 users as the instances and the 65 movies as the attributes. (We tried 10 groups of 130 movies but found this did not work well, perhaps because many classifiers and imputation techniques cannot handle 130 attributes.) Table 2 characterizes each of the *MovieLens* data subsets, listing the number of ratings included, percentage of ratings of the original data, and the sparsity of each dataset.

4. Experimental Design and Results

Besides implementing *IBCF-NB, IBCF-mean, IBCF-NBM, IBCF-LinR and IBCF-PMM,* we also implemented the *PearsonCF* (the traditional memory-based *CF;* Section 2) and content-boosted *CF* (the representative hybrid *CF;* Section 1).

When applying the *content-boosted CF*, we use the four content attributes provided by *MovieLens* [12]: *age* (seven values reflecting seven age groups); *sex* (two values); *occupations* (21 values); *zip code* (20 groups). The class has five values, {1, 2, 3, 4, 5}. We use the software

package *MICE* (Multiple Imputation by Chain Equations) [3] to compute the imputed data from *PMM* imputations.

We use an *all-but-one* strategy to train and test our *CF* predictors, in which we pretend each observed rating is missing and make a prediction for it using all other observed ratings and then compare the prediction with the ground truth to evaluate the predictive performance. We summarize the overall performances of the *IBCFs* in Table 3 and Figure 3 over all 100,000 *MovieLens* ratings. (This is aggregated from the *RMSE* values from the 20 data subsets and the remaining less-than-one-percent ratings using default voting (see Table 2), weighting each by its percentage of ratings in the original rating dataset.)

In Table 3, *rmse1* is the average *RMSE* value of the datasets *d_1* through *d_10*, which have sparsities less than 95% and covers about 86% of the ratings of the original data (on average, at least 48 out of the 943 users have rated any of the items in each of these datasets); and *rmse2* is based on datasets *d_11* through *d_20*, about 13% of the original rating data, with sparsities larger than 95% (less than 48 users have rated any item in each dataset).

Table 3. Predictive Performance of the IBCFs and CF algorithms

	IBCF NB	IBCF mean	IBCF NBM	IBCF LinR	IBCF PMM	CBCF	Pearson CF
d_1	0.9387	0.9862	0.9387	0.9330	0.9341	0.9786	0.9743
d_2	0.9336	1.0000	0.9336	0.9320	0.9337	0.9801	0.9927
d_3	0.9457	1.0156	0.9457	0.9456	0.9428	0.9918	1.0280
d_4	0.9530	1.0399	0.9530	0.9683	0.9610	1.0142	1.0660
d_5	0.9683	1.0319	0.9683	0.9718	0.9663	1.0156	1.0667
d_6	0.9773	1.0478	0.9773	0.9857	0.9956	1.0198	1.0751
d_7	0.9785	1.0379	0.9785	0.9938	0.9860	1.0169	1.0954
d_8	1.0013	1.0415	1.0013	1.0157	1.0037	1.0321	1.1333
d_9	1.0146	1.0566	1.0146	1.0386	1.0272	1.0469	1.1772
d_10	0.9969	1.0194	0.9969	1.0166	1.0161	1.0112	1.1647
rmse1	**0.9565**	1.0164	**0.9565**	0.9607	0.9588	0.9995	1.0400
d_11	1.0757	1.0699	1.0699	1.0680	1.0755	1.0717	1.2348
d_12	1.1147	1.0963	1.0963	1.0987	1.0931	1.0773	1.2735
d_13	1.1207	1.0500	1.0500	1.0477	1.0663	1.0378	1.3575
d_14	1.2190	1.1195	1.1195	1.1094	1.1007	1.1252	1.4628
d_15	1.2719	1.1157	1.1157	1.1036	1.1036	1.1246	1.5511
d_16	1.2870	1.0848	1.0848	1.1018	1.0821	1.1009	1.4717
d_17	1.2568	1.0760	1.0760	1.1198	1.1259	1.1021	1.5318
d_18	1.2639	1.0775	1.0775	1.0995	1.0825	1.0775	1.4977
d_19	1.2518	1.0883	1.0883	1.0892	1.0981	1.0297	1.6675
d_20	1.4648	1.1491	1.1491	1.2027	1.1395	1.0647	1.7585
rmse2	1.1810	1.0871	1.0871	1.0912	1.0898	**1.0818**	1.3999
RMSE	0.9901	1.0292	**0.9777**	0.9819	0.9800	1.014	1.0905
MAE	0.6855	0.7582	**0.6850**	0.7187	0.7181	0.7264	0.7921

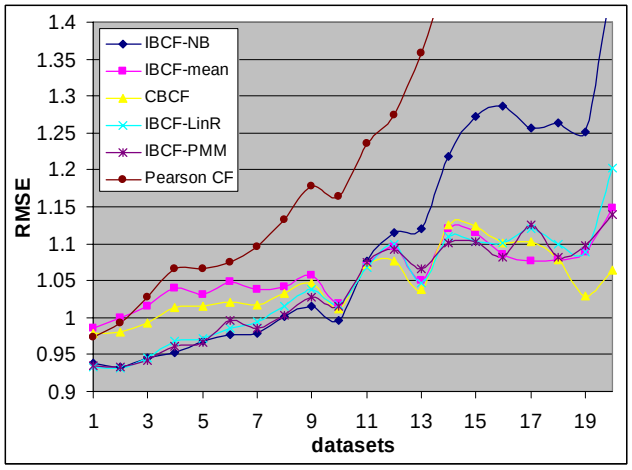

Figure 3. Prediction performance in terms of RMSE of the IBCFs and their peer CF algorithms on the 20 subsets of MovieLens data, ordered by sparsities

This result shows that *IBCF-NB* performs the best on the datasets with sparsity less than 95%; its score is better than the 2nd best *IBCF-PMM* with 1-sided t-test p<0.031. It loses its lead to other *IBCFs* after dataset *d_11* (with sparsity 95.6%, see Figure 3). *CBCF* performs the best on the datasets with sparsity larger than 95%, slightly better than the 2nd-place *IBCF-mean* with p<0.16. Through using *IBCF-NB* for relatively dense datasets (sparsity<95%) and *IBCF-mean* for highly sparse datasets (sparstiy≥95%), the overall *RMSE* performance of *IBCF-NBM* becomes the best.

We show below the overall ranking of the *CF* algorithms using 1-sided t-test where ">" implies "insignificantly better", with p-value larger than 0.05, and ">>" implies "significantly better", with p-value smaller than 0.05.

IBCF-NBM (p<0.07)> **IBCF-PMM** (p<0.08)> **IBCF-LinR** (p<0.004)>> **IBCF-NB** (p<0.03)>> **CBCF** (p<0.009)>> **IBCF-mean** (p<6E-5)>> **PearsonCF**

IBCF-NBM, *IBCF-PMM*, *IBCF-LinR* and *IBCF-NB* have 3.58%, 3.35%, 3.17% and 2.36% lower *RMSE* scores than *CBCF* respectively; and they have 10.34%, 10.13%, 9.96% and 9.21% lower *RMSE* than the traditional *PearsonCF*.

As hybrid *CF* algorithms rely on external content information that are usually not available, the fact that our *IBCF*s (*IBCF-NBM*, *IBCF-PMM, IBCF-LinR, and IBCF-NB*) working on pure rating data can significantly outperform the *content-boosted CF*, has important significance. Note that our *IBCF-NBM* achieves an *MAE* value (mean absolute error, $MAE = \frac{1}{n}\sum_{\{u,i\}}|p_{u,i} - r_{u,i}|$) of 0.685 (see Table 3), while the previous reported best *MAE* score on the *MovieLens* data in the literature is 0.72 by [14].

5. Conclusions

Collaborative filtering (*CF*) is one of the most effective ways to produce recommendations. These systems use a rating matrix, whose sparsity is a serious challenge for *CF* algorithms. We therefore use imputation-boosted collaborative filtering (*IBCF*) algorithms, which attempt to boost *CF* performance by first using some imputation process to fill in omissions in the rating matrix, then making recommendations from this completed data rather than the original rating data. We compare the performance of several such systems, including *IBCF-PMM* (predictive mean matching), *IBCF-mean* (mean imputation), *IBCF-LinR* (linear regression), and *IBCF-NB* (naïve Bayes classifier). The results of their performances on datasets with different sparsities suggest an easy-to-implement mixture *IBCF*, *IBCF-NBM*, which uses *IBCF-NB* for relatively dense datasets, *IBCF-mean* for the extremely sparse datasets, and a sparsity threshold that is determined by the score where these two algorithms change the lead of their predictive performances. Empirical results show that *IBCF-NBM* outperforms *IBCF* using standard imputation techniques (*IBCF-PMM*, *IBCF-LinR*, and *IBCF-mean*), and have significantly better predictive performance than the representative hybrid *CF* algorithm *content-boosted CF* (even though *IBCF-NBM* does not use external content information).

References

[1] Adomavicius, G., and Tuzhilin, A., Toward the Next Generation of Recommender Systems: a Survey of the State-of-the-Art and Possible Extensions, *IEEE Transactions on Knowledge and Data Engineering*, 17(6), pp. 734-749, 2005.

[2] Breese, J., Heckerman, D., and Kadie, C., Empirical Analysis of Predictive Algorithms for Collaborative Filtering, *Proceedings of Fourteenth Conference on Uncertainty in Artificial Intelligence,* Madison, WI, Morgan Kaufmann, July 1998.

[3] Buuren, S.V. and Oudshoorn, C.G.M. Flexible Multivariate Imputation by Mice. *Leiden: TNO Preventie en Gezondheid*, TNO/VGZ/PG 99.054, 1999.

[4] Dempster, A.P., Laird, N.M., Rubin, D.B. Maximum likelihood from incomplete data via the EM algorithm, *Journal of the Royal Statistical Society*, B 39, pp. 1-38, 1977.

[5] Grover, R., Vriens, M., *The Handbook of Marketing Research: Uses, Misuses, and Future Advances*, Sage Publications, 2006.

[6] Landerman, L.R., Land, K.C. and Pieper, C.F. An Empirical Evaluation of the Predictive Mean Matching Method for Imputing Missing Values." *Sociological Methods & Research* 26: 3-33, 1997.

[7] Little, R.J.A. Missing-Data Adjustments in Large Surveys. *Journal of Business & Economic Statistics*, 6(3), 287-296, 1988.

[8] Mahalanobis, P.C., On the Generalised Distance in Statistics, *Proceedings of the National Institute of Science of India*, 12, pp. 49-55, 1936.

[9] Marlin, B.M., Zemel, R.S., Roweis, S., Slaney, M., Collaborative Filtering and the Missing at Random Assumption, *UAI*, 2007.

[10] Melville, P., Mooney, R.J. and Nagarajan, R. Content-Boosted Collaborative Filtering for Improved Recommendations, *Proceedings of the 18th National Conference of Artificial Intelligence*, 2002.

[11] Miyahara, K., and Pazzani, M.J. Improvement of Collaborative Filtering with the Simple Bayesian Classifier, *Information Processing Society of Japan*, 43(11), 2002.

[12] MovieLens data. http://movielens.umn.edu.

[13] Schafer, J.L. Analysis of Incomplete Multivariate Data, *New York: Chapman and Hall*, 1997.

[14] Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J., Item-based Collaborative Filtering Recommendation Algorithms, *WWW*, 2001.

[15] Schuurmans, D., and Greiner, R., Learning Default Concepts, *Canadian Conference on Artificial Intelligence (CAI)*, Banff, Canada, May 1994.

[16] Su, X. and Khoshgoftaar, T.M. Collaborative Filtering for Multi-class Data Using Belief Net Algorithms, *IEEE ICTAI*, pp 497-504, 2006.

[17] Su, X., Khoshgoftaar, T.M., X. Zhu, and R. Greiner, Imputation-Boosted Collaborative Filtering Using Machine Learning Classifiers, *ACM SAC, 2008*

[18] Ungar, L.H., and Foster, D.P. Clustering Methods for Collaborative Filtering, *Proceedings of the Workshopon Recommendation Systems*, AAAI Press, 1998.

[19] Zio, M.D., and Guarnera, U., A Semiparametric Predictive Mean Matching: An Empirical Eavaluation, *Conference of European Statistics*, Bonn, Germany, 2006.