Imputed Neighborhood Based Collaborative Filtering

Xiaoyuan Su  
Computer Science & Engineering  
Florida Atlantic University  
xsu@fau.edu

Taghi M. Khoshgoftaar  
Computer Science & Engineering  
Florida Atlantic University  
taghi@cse.fau.edu

Russell Greiner  
Department of Computing Science  
University of Alberta  
greiner@cs.ualberta.ca

Abstract

Collaborative filtering (CF) is one of the most effective types of recommender systems. As data sparsity remains a significant challenge for CF, we consider basing predictions on imputed data, and find this often improves performance on very sparse rating data. In this paper, we propose two imputed neighborhood based collaborative filtering (INCF) algorithms: imputed nearest neighborhood CF (INN-CF) and imputed densest neighborhood CF (IDN-CF), each of which first imputes the user rating data using an imputation technique, before using a traditional Pearson correlation-based CF algorithm on the resulting imputed data of the most similar neighbors or the densest neighbors to make CF predictions for a specific user. We compared an extension of Bayesian multiple imputation (eBMI) and the mean imputation (MEI) in these INCF algorithms, with the commonly-used neighborhood based CF, Pearson correlation-based CF, as well as a densest neighborhood based CF. Our empirical results show that IDN-CF using eBMI significantly outperforms its rivals and takes less time to make its best predictions.

1. Introduction

Many successful recommender systems use collaborative filtering (CF), which uses a database of user preferences of items to predict additional topics or products that other users might like. A typical CF scenario involves a “ratings matrix” over m users {u1, u2, ..., um} and n items {i1, i2, ..., in}, whose (u,i) entry denotes how user u rated item i. The ratings can either be explicit indications (e.g., on a 1-5 scale) or implicit indications, such as purchases or click-throughs [6]. Note this matrix is typically very sparse, as most users rate relatively few items.

Existing CF solutions roughly fall in the following three categories. Memory-based CF algorithms use the entire rating data (or perhaps a subsample) to generate a prediction. Model-based CF algorithms first build models, such as machine learned classifiers, from user ratings, then make predictions using the models. Hybrid CF algorithms make predictions or recommendations typically by combining CF and content-based recommenders, which are based on auxiliary information, such as documents, URLs, news messages, web logs, user profiles and item descriptions about users’ tastes, preferences, and needs.

The Pearson correlation-based CF (Pearson CF) algorithm is a representative memory-based CF algorithm [8], which calculates similarities between each pair of items rated by a user, or between each pair of users who rate the same item (Section 2). This allows these systems to scale effectively with the number of users and items, as this computation involves only a small percentage of the total number of items and users – i.e., due to the sparsity. One shortcoming of Pearson CF is that its predictive performance degrades quickly as the data becomes more sparse [9]. Neighborhood-based CF makes CF predictions using Pearson CF based on the nearest neighbors instead of the whole dataset; Bell and Koren [1] demonstrated that this improved predictive performance.

Model-based CF techniques, such as Bayesian belief nets CF algorithms, address the sparsity challenge of CF tasks by providing relatively accurate predictions even given sparse rating data. However, their empirical performance is not significantly better than the Pearson CF or is not good enough [7][9].

In order to improve predictive accuracy for CF tasks by taking advantage of additional content information, content-boosted CF (a representative hybrid CF method) uses naïve Bayes as the predictor to estimate missing values based on content information and applies a weighted Pearson CF algorithm to produce CF predictions on the generated pseudo rating matrix (the rating matrix whose missing values have been filled in) [5].

This Content-boosted CF system begins by filling in the missing values; this corresponds to “imputation”,...
which is commonly used in application areas, such as social surveys, where missing data may cause difficulties in estimation and inference [11]. Figure 1(b) presents an example of imputed data – a pseudo rating matrix – which can be used as the input for CF prediction.

<table>
<thead>
<tr>
<th>I1</th>
<th>U1</th>
<th>U2</th>
<th>U3</th>
<th>U4</th>
<th>U5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>I2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>I3</td>
<td>?</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>I4</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>I5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Figure 1. (a) original rating data, (b) imputed rating data (pseudo-rating matrix)

As imputation techniques can generally improve performance when mining incomplete data, we observe that CF predictions based on imputed data will also improve predictive performance for CF tasks.

We previously proposed imputation-boosted collaborative filtering (IBCF) systems that impute the missing values with machine learned classifiers or mean imputation before using the Pearson CF to make final predictions on the imputed data. Our earlier results show that these systems improved predictive performance on low dimensional rating data (that have a small number of users or items) [12][13]. However, we did not apply the IBCF approach to the commonly used neighborhood based CFs [8][1].

We therefore propose two imputed neighborhood based collaborative filtering (INCF) algorithms: imputed nearest neighborhood CF (INN-CF), which first finds the users most similar to the active user (for which we are making predictions), then uses the corresponding imputed rating data to make predictions for the active user; and imputed densest neighborhood CF (IDN-CF), which makes predictions from the imputed densest neighbors (i.e., the users who have rated the most number of items). The imputation techniques we considered include the baseline mean imputation (MEI) and an extension of Bayesian multiple imputation (eBMI). To evaluate our systems, we compare these INCF algorithms with the commonly-used Pearson CF, neighborhood-based CF algorithm, and a densest neighborhood based CF algorithm.

As performance metrics, we use mean absolute error (MAE), which computes the average of the absolute difference between the predictions and true ratings:

$$MAE = \frac{\sum_{u,i} |p_{ui} - r_{ui}|}{n}$$  

where $n$ is the total number of ratings over all [user, item] pairs in the test set, $p_{ui}$ is the predicted rating for user $u$ on item $i$, and $r_{ui}$ is the ground truth rating value. The lower the MAE, the better the predictor.

Section 2 presents Pearson correlation based CF and neighborhood based CF algorithms. Section 3 presents the framework of our imputed neighborhood based collaborative filtering algorithms. Section 4 presents experimental design and results.

2. Pearson correlation based CF and neighborhood based CF algorithms

2.1 Pearson correlation based CF algorithm

The traditional Pearson correlation based collaborative filtering algorithm, Pearson CF, is a representative memory-based CF algorithm that uses the rating database to generate a prediction.

The Pearson correlation between item $i$ and $j$ is

$$w_{i,j} = \frac{\sum_{u \in U} (r_{ui} - \bar{r})(r_{uj} - \bar{r})}{\sqrt{\sum_{u \in U} (r_{ui} - \bar{r})^2} \sqrt{\sum_{u \in U} (r_{uj} - \bar{r})^2}}$$  

where the $u \in U$ summations are over the users who rated both the items $i$ and $j$, and $\bar{r}$ is the average rating of the co-rating users on the $i$-th item. In Figure 1(a), for $i=I3$ and $j=I4$, we find $\bar{r} = 2.33$ by computing the average rating on the item $I3$ from the three users that also rated item $I4$ (who are $U2$, $U4$, and $U5$), and $\bar{r} = 2.67$, and the Pearson correlation between $I3$ and $I4$ is $w_{3,4} = 0.189$. We similarly compute $w_{i,j}$ for all pairs of items $i$ and $j$.

To make a prediction for the active user, $a$, on a certain item, $b$, the item-based Pearson CF algorithm returns the weighted average of all the ratings on that item using the formula

$$p_{a,b} = \bar{r}_b + \frac{\sum_{i \in I} (r_{ai} - \bar{r})(r_{bi} - \bar{r}) \cdot w_{bi}}{\sum_{i \in I} w_{bi}}$$  

where $\bar{r}_b$ and $\bar{r}$ are the average ratings for item $b$ and item $i$ that all other users have rated, and $w_{bi}$ is the Pearson correlation between items $b$ and $i$, defined in Equation 2 above [2]. The $i \in I$ summations are over all the items that the user $a$ has rated. Here in Figure 1(a), for user $a=U2$, and item $i=I5$, the prediction is $p_{2,5} = 3$.

2.2 Neighborhood based CF algorithm using adjusted Pearson correlation

The “neighborhood based CF”, aka the nearest neighborhood based CF algorithm, first finds the $N$ neighbors nearest to the active user, then makes CF predictions using Pearson CF based only on those neighbors.
To find the nearest (most similar) neighbors of the active user $u$, we first calculate the similarity in terms of Pearson correlation between user $u$ and each other user $v$

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

(4)

where the $i \in I$ summations are over the items that rated by both the users $u$ and $v$ and $\bar{r}_i$ is the average rating of the co-rated items of the $i$-th user. To help regularize the values, and also avoid the divide-by-zero (DBZ) situation (when all of the values are the same, and so the variance is 0), we use an adjusted Pearson correlation:

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

(5)

where $\varepsilon$ is a small value. For example, in Figure 1(a), with $\varepsilon=0.001$, the adjusted Pearson correlation between $u_4$ and $u_5$ on co-rated items (between $\{2,2,2\}$ and $\{2,4,3\}$) will be 0.002 instead of a DBZ exception. This adjusted Pearson correlation will slightly bias the original Pearson correlation for normal cases. For example, the adjusted Pearson correlation (from Equation 5) for $\{3,3,4\}$ and $\{5,4,4\}$ is -0.499996, while the original Pearson correlation (Equation 4) is -0.5.

We similarly compute adjusted Pearson correlation $w_{u,v}$ for all pairs of users $u$ and $v$, and take the $N$ users with the top values of $w_{u,v}$ as the most similar neighbors of the active user $u$.

Note we use the adjusted Pearson correlation only for the divided-by-zero cases and otherwise use the original Pearson correlation. Our empirical results show that this strategy performs better than solely using the original Pearson correlation, or solely using the adjusted Pearson correlation.

As the neighborhood based CF is effective in producing CF predictions, it is commonly used and deployed in many commercial systems. A major shortcoming of neighborhood based CF, however, is its poor predictive performance when the rating data is sparse. This is especially true for new users, who have very few ratings, as here the set “$I$” in Equation (4) will be small, or even empty. We address this by using imputation: rather than run this neighborhood based CF algorithm on the original data, our INCF algorithms (described in the following sections) first impute values for the missing entries based on the most similar or the densest users, then run Pearson CF on the imputed data to make CF predictions. As Pearson CF typically has better predictive accuracy for denser rating data, we expect this will lead to better performance. However, this means the overall algorithm will be slower than neighborhood based CF over the same number of neighbors, as its predictions are based on the (pseudo) rating matrix (Figure 1(b)), which is now much denser. Nevertheless, INCF algorithms may require less time to produce predictions as they might need much fewer neighbors.

### 2.3 Densest neighborhood based CF algorithm

Our novel densest neighborhood based CF (DNCF) makes CF predictions based on the densest users (those who have rated most number of items), rather than the most similar users. DNCF is inspired by the fact that Pearson CF performs better on denser rating data.

This algorithm first ranks the users based on the number of items they have rated, then uses Pearson CF on the ratings of the top $N$ densest users to give CF predictions for the active user. Note that these densest users does not depend on the active user.

### 3. Imputed neighborhood based CF

The main steps of imputed neighborhood based CF algorithm are (see Figure 2): (1) impute the user rating data to generate a pseudo rating data (imputed data); (2) for each active user, find the $N$ most similar users in the original rating matrix (for imputed nearest neighborhood CF, or INN-CF), or find the $N$ densest users of the dataset (for imputed densest neighborhood CF, or IDN-CF); (3) make CF predictions for the active user by applying the traditional Pearson CF algorithm on the imputed rating data of the most similar (or the densest) users.

![Figure 2. Illustration of imputed neighborhood based collaborative filtering: INN-CF (using most similar neighbors) and IDN-CF (using densest ones)](image-url)
The selection of an imputation technique should be based on an analysis of the missing data patterns. If the missingness does not depend on the observed data, then this is considered MCAR (Missing Completely at Random, e.g., \( r_{ui} \) is missing with an iid (independent and identical distribution) probability of 0.6, independent of everything else). If the missing data depend on the observed data, but not on the unobserved data, they are MAR (Missing at Random, e.g., missingness on item \( I_m \) is dependent on observed ratings on item \( I_s \) such as the entry \((u,m)\) for a user’s rating of a horror movie is missing if the value \( r_{a,n} \) for that user’s rating for Bambi is specified and is 5) [11]. Both MCAR and MAR missing data patterns are considered ignorable, as one can make valid likelihood-based inferences and estimations even if the missing data mechanism is ignored (not explicitly considered in the analysis). If the “missingness” is not MAR, then we say the data are NMAR (not missing at random) aka non-ignorable, in which patterns of the unobserved variables can determine their own missingness (e.g., \( r_{ui} \) is missing when it is <3). When the missing data are NMAR, a model of missing value mechanism must be incorporated into the estimation process, which is practically difficult, because the missing data mechanism is rarely known with certainty [10].

While most assume the missing pattern of CF data is MAR, there are also cases that they are NMAR -- e.g., many users will not rate a movie because he did not like it. Here, a multiple imputation technique (see Section 3.3) can be used for CF tasks, as it typically reduces bias for both MAR and NMAR data. These motivate us to investigate applying a multiple imputation technique in the INCF framework.

Our previous work also investigated applying other imputation techniques for CF tasks, such as using linear regression imputation, and predictive mean matching [12]. However, as their performances are not as good as the techniques shown here, we do not consider them in this work.

### 3.2 INCF-MEI: imputed neighborhood based CF using mean imputation

INCF-MEI (INCF using mean imputation) produces the pseudo rating data using mean imputation on users, which replaces each missing value \( (u,m) \) in the rating matrix with the mean of the observed ratings on each user, that is \( \frac{1}{k} \sum_{i \in I(u)} r_{ij} \), where \( k \) is the total number of items \( i \in I(u) \) that have been rated by the user \( u \). 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), and it also reduces the variance of the predictions.

### 3.3 INCF-eBMI: imputed neighborhood based CF using extended Bayesian multiple imputation

Multiple imputation (MI) produces many different imputed datasets – e.g., one such dataset might correspond to Figure 1(b), while another would fill in the missing values differently, perhaps setting \( r(U_2, I_2) \) to 5 (not 2), and \( r(U_2, I_3) \) to 2. Each set of new values is estimated independently from the posterior predictive distribution. The final estimate is obtained by combining these \( m \) estimates. Multiple imputation is better than standard single imputation as it accounts for the uncertainty about the predictions of the imputed values.

Bayesian multiple imputations (BMI) are created through a Bayesian process: first specify a parametric model for the complete data and a model for the mechanism by which data become missing (e.g., MAR); typically using some prior distribution over the unknown model parameters, then simulated \( m \) independent draws from the conditional distribution of the missing data given the observed data. In non-trivial applications, special computational techniques such as Markov chain Monte Carlo (MCMC) must be applied to create BMI. (While this BMI model assumes the imputations are generated from a multivariate normal distribution, BMI is known to be robust to departures from the assumption [10]).

In more detail, the BMI imputed data are created through the following steps [10]:

- We assume that \( Y= (Y_{obs}, Y_{miss}) \) follows a parametric model \( P(Y|\theta) \) where \( \theta \) has the prior distribution \( P(\theta) \); hence the posterior predictive distribution for \( Y_{miss} \) is

\[
P(Y_{miss}|Y_{obs}) = \int P(Y_{miss}|Y_{obs}, \theta)P(\theta|Y_{obs})d\theta \quad (6)
\]

- BMI repeats the two-step process for \( j=1, \ldots, m \):
  1. draw \( Y_{miss}^{(j+1)} \) from \( P(Y_{miss}|Y_{obs}, \theta^{(j)}) \);
  1. draw \( \theta^{(j+1)} \) from \( P(\theta|Y_{obs}, Y_{miss}^{(j+1)}) \).

(1a) and (1b) simulate the two-step process for \( j=1, \ldots, m \):

- (1a) draw \( Y_{miss}^{(j+1)} \) from \( P(Y_{miss}|Y_{obs}, \theta^{(j)}) \);
- (1b) draw \( \theta^{(j+1)} \) from \( P(\theta|Y_{obs}, Y_{miss}^{(j+1)}) \).

(2) augment data using Markov Chain Monte Carlo (MCMC), in which the following Markov chain \( \{Y_{miss}^{(0)}, \theta^{(1)}, Y_{miss}^{(2)}, \theta^{(2)}, \ldots, Y_{miss}^{(m)}, \theta^{(m)}\} \) is generated, where each pair \( (Y_{miss}^{(j)}, \theta^{(j)}) \) depends on the previous one, and the stabilized distribution \( \{Y_{miss}^{(j)}, \theta^{(j)}\} \) as \( j \to \infty \) is a draw from the target probability distribution function.

These two steps are repeated enough times between successive entries that (we anticipate) the values have no additional correlations [11].

After producing \( m \) sets of imputed data, we take the average as the final imputed data.
\[ \hat{\theta} = \frac{1}{m} \sum_{i=1}^{m} \hat{\theta}_i \]  
(7)

where \( \hat{\theta}_i \) is the estimate based on the \( i \)-th imputed data.

In the cases when the entries for an instance is empty (ie., there is no ratings on a new item, from any user), BMI will be unable to produce any imputed and so leave those entries missing again. We therefore propose an extension of Bayesian multiple imputation (eBMI) to handle this problem, by filling in these missing values using the attribute average (mean imputation). Hence,

\[ \hat{\theta}_{eBMI} = \begin{cases} 
\hat{\theta}_{BMI} & (\theta_{BMI} \neq \phi) \\
\hat{\theta}_{MEI} & (\theta_{BMI} = \phi) 
\end{cases} 
\]  
(8)

where \( \hat{\theta}_{BMI} \), \( \hat{\theta}_{BMI} \), \( \hat{\theta}_{MEI} \) are imputed values from eBMI, BMI, and MEI respectively: that is, when the imputed value from BMI is empty (\( \phi \)), eBMI uses \( \hat{\theta}_{MEI} \) as the imputation; otherwise, it uses \( \hat{\theta}_{BMI} \).

As the eBMI values must be within the attribute’s range \([\text{min}, \text{max}]\), we replace an estimate < \( \text{min} \) with \( \text{min} \) and any \( \text{max} \) with \( \text{max} \).

\[ \hat{\theta} = \begin{cases} 
\text{min} & (\hat{\theta} < \text{min}) \\
\hat{\theta} & (\hat{\theta} \in [\text{min}, \text{max}]) \\
\text{max} & (\hat{\theta} > \text{max}) 
\end{cases} 
\]  
(9)

4. Experimental design

We worked on the MovieLens dataset [4], which is a widely used movie recommendation dataset with 100,000 ratings for 1682 movies by 943 users, where each rating is in \([1, 2, \ldots, 5] \), where 93.7% of the entries are missing. Numerous research papers on collaborative filtering are based on this dataset [3][8]. We use 20% of the 100,000 ratings of MovieLens dataset as test data, and use the remaining 80% observed ratings as the training data; this training-test split is used by many researchers working on this dataset [8].

We implement the proposed imputed neighborhood based CF algorithms, INN-CF and IDN-CF (using eBMI and MEI as the imputers – corresponding to 4 different algorithms), the densest neighborhood based CF algorithm (which uses only the original data with no imputation), as well as the standard neighborhood based CF algorithm, and Pearson CF. We experimented on the above CF algorithms with neighborhood size ranging from 20 through 940, with a step-size of 20 neighbors (except for Pearson CF, which uses the possible maximum neighbors, which is \text{total user number} - 1), and investigate their respective predictive performance in terms of \text{MAE} (Equation 1).

The experimental results on the original MovieLens data are shown in Table 1, Table 2, and Figure 3 (testing on 19,996 reserved ratings out of 100,000).

<table>
<thead>
<tr>
<th>CF algorithms</th>
<th>best MAE</th>
<th>neighbor # at best</th>
<th>time (m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>densest neighborhood CF</td>
<td>0.7151</td>
<td>920</td>
<td>5</td>
</tr>
<tr>
<td>neighborhood-based CF</td>
<td>0.7097</td>
<td>620</td>
<td>5</td>
</tr>
<tr>
<td>INN-CF (eBMI)</td>
<td>0.7133</td>
<td>460</td>
<td>6</td>
</tr>
<tr>
<td>INN-CF (MEI)</td>
<td>0.7736</td>
<td>520</td>
<td>7</td>
</tr>
<tr>
<td>IDN-CF (eBMI)</td>
<td>0.6926</td>
<td>120</td>
<td>2</td>
</tr>
<tr>
<td>IDN-CF (MEI)</td>
<td>0.7440</td>
<td>60</td>
<td>1</td>
</tr>
<tr>
<td>Pearson CF</td>
<td>0.7205</td>
<td>942</td>
<td>7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 2. Part of the MAE performances of the CF algorithms with different neighborhood sizes on the MovieLens dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Neibr #</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td>60</td>
</tr>
<tr>
<td>120</td>
</tr>
<tr>
<td>180</td>
</tr>
<tr>
<td>240</td>
</tr>
<tr>
<td>300</td>
</tr>
<tr>
<td>360</td>
</tr>
<tr>
<td>420</td>
</tr>
<tr>
<td>480</td>
</tr>
<tr>
<td>540</td>
</tr>
<tr>
<td>600</td>
</tr>
<tr>
<td>660</td>
</tr>
<tr>
<td>720</td>
</tr>
<tr>
<td>780</td>
</tr>
<tr>
<td>840</td>
</tr>
<tr>
<td>900</td>
</tr>
<tr>
<td>940</td>
</tr>
<tr>
<td>Ave</td>
</tr>
</tbody>
</table>

Imputed densest neighborhood CF (IDN-CF) using eBMI has the best performance here, with a 3.9% lower \text{MAE} than the Pearson CF (0.6926 vs. 0.7205). Moreover, it needs fewer neighbors (with 120 neighbors, out of a possible maximum size of 942) to reach the best performance, than most of its rivals (which need more than 450 neighbors each, except IDN-CF using MEI, which needs only 60 neighbors to achieve the best \text{MAE} but has much worse
With a smaller neighborhood, IDN-CF using eBMI produces CF predictions in less time (2 minutes), in comparison with other CF algorithms (except IDN-CF using MEI) on a computer with Xeon E5430QC/2.66G CPU and 32.8GB RAM. Imputed nearest neighborhood CF (INN-CF) using eBMI performs better than the Pearson CF with a 1% lower MAE. Both IDN-CF and INN-CF using eBMI significantly outperform the commonly-used neighborhood based CF, with 5.8% and 2.8% lower average MAE scores over all the neighborhood sizes we investigated, and with 1-sided t-test p-values of p<0.0002 and p<0.003 respectively. Our densest neighborhood based CF also outperforms the (nearest) neighborhood based CF with a 3.5% lower average MAE score, with p<2E-5.

Although IDN-CF using eBMI improved predictive performance and reduced online prediction time, it needed more time for offline imputation. When imputing the training dataset using eBMI, we worked on segmented smaller datasets, each of which had 100 users (attributes) except the last one that has 43; and later aggregated the imputed datasets into one. This strategy reduces the offline imputation time, although each subset still required about 11 minutes to impute using our computer.

5. Conclusions

Collaborative filtering (CF) is one of the most successful approaches to building effective recommender systems. However, these CF algorithms must be able to deal with high-dimensional and very sparse rating data. Imputed neighborhood based CF (INCF) algorithms attempt to boost the performance of the standard Pearson correlation-based CF (Pearson CF) and neighborhood based CF systems by first computing, and then using, high-quality imputed data computed from the densest neighbors or the users most similar to the active user. We proposed a high quality imputation technique, an extension of Bayesian multiple imputation (eBMI), and found it is effective in the INCF framework, especially for the imputed densest neighborhood CF (IDN-CF) algorithm. IDN-CF using eBMI significantly outperforms the traditional Pearson CF, neighborhood based CF algorithms, and other INCF algorithms, and takes less online prediction time than most of its rivals. Imputed nearest neighborhood CF (INN-CF) using eBMI also has improved predictive performance for CF tasks.

References

[5] Melville, P., Mooney, R.J. and Nagarajan, R. Content-Boosted Collaborative Filtering for Improved

Figure 3. MAE performance of the INCF algorithms on the MovieLens dataset

Our INCF algorithms (IDN-CF using eBMI and INN-CF using eBMI, which have the best MAE scores of 0.6926 and 0.7133 respectively) beat the best MAE score on the same MovieLens dataset in literature [3], which applied an item-based Pearson CF with an MAE of 0.72 [8].

Both IDN-CF and INN-CF perform gradually worse with more neighbors than the neighbors producing the best performance, probably due to overfitting.

Mean imputation (MEI), the baseline imputation technique, did not improve CF performance when applied to either IDN-CF or INN-CF. As MEI will replace each missing value with that attribute's mean value, if the CF rating data is missing many values for an attribute, then essentially all of those missing values will become that mean value. It is especially problematic for datasets with few items, such as MovieLens, where 40% of the users rated five or fewer items; moreover, by holding out 20% of observed ratings, the users had yet fewer ratings.


