Online Selection of Mediated and Domain-Specific Predictions for Improved Recommender Systems

Stephanie Rosenthal, Manuela Veloso, Anind Dey
School of Computer Science
Carnegie Mellon University
{srosenth,veloso,anind}@cs.cmu.edu

Abstract

Recommender systems use a set of reviewers and advice givers with the goal of providing accurate user-dependent product predictions. In general, these systems assign weights to different reviewers as a function of their similarity to each user. As products are known to be from different domains, a recommender system also considers product domain information in its predictions. As there are few reviews compared to the number of products, it is often hard to set the similarity-based weights as there is not a large enough subset of reviewers who reviewed the same products. It has then been recently suggested that not considering domains will increase the amount of reviewer data and the overall prediction accuracy in a *mediated* way. However, clearly, if different reviewers are similar to a user in each product domain, then *domain-specific* predictions could be superior to mediated ones.

In this paper, we consider two advice giver algorithms to provide *domain-specific* and *mediated* predictions. We analyze both advice giver algorithms using large real data sets to characterize when each is more accurate for users. We realize that for a considerable number of users, the *domain-specific* predictions are possible and more accurate. We then contribute an improved general recommender system algorithm that autonomously selects the most accurate mediated or domain-specific advice giver *for each user*. We validate our analysis and algorithm using real data sets and show the improved predictions for different users.

Introduction

We model a product recommender system as a set of reviews defined by reviewers, product domains (*e.g.,* DVDs, books, clothes), and advice givers. Users request that the recommender system provide predictions of whether they will like a set of products of their choosing. Users then have the option of providing their own reviews of those products. As the advice giver that makes product predictions receives the user's actual reviews, it assigns *domain-specific* weights to the reviewers as a function of the similarity between their reviews and the user's. The reviewers whose reviews are most similar to the user's receive higher weight. The advice giver

uses the weights of the reviewers and their reviews to guide its predictions with the goal of providing the most accurate predictions.

Because reviewers review a relatively small number of products, it is difficult to find enough reviewers with similar reviews to make accurate predictions for every product each user requests. The problem is exacerbated as products are divided into domains so there are fewer product reviews to train the domain-specific weights with. Resolving the data sparsity problem has been the focus of much recommender system work. Although it is widely accepted that *domain-specific* reviewers result in accurate predictions, it has recently been suggested that a *mediated* advice giver that combines multiple domains of products and holds only a single set of weights for each, user would help alleviate the data sparsity problem (Berkovsky, Kuflik, and Ricci 2007a).

While learning only one set of weights will increase the amount of data to train with, there is an underlying assumption that the reviewers with similar reviews to a user in one product domain (*e.g.,* DVDs) will also have similar reviews to that user in other domains (*e.g.,* books and clothes). While a domain-specific advice giver captures these differences, the mediated advice giver does not. Intuitively, it seems unlikely that for all users there is a set of reviewers with similar reviews in all domains of products. The focus of this work is to understand in real recommender data sets how data sparsity and the user's reviews affect the weights of the reviewers and the accuracy of the advice givers' predictions.

We first present an overview of how advice givers weigh reviewers and make predictions for users and give examples of weights that affect the two advice givers' accuracy. We show using data from two large recommender systems that potentially half of the users benefitted from the mediated advice giver while the other half required domain-specific weights. Additionally we find in a third and more sparse recommender data set that both advice givers have equal accuracies when reviewers do not provide reviews for more than one category. We, then, show how accuracy changes for each advice giver as reviewers review products in more categories. Assuming that reviewers do provide reviews in more categories (as found in the first two data sets) and because different users require the two advice givers equally, we contribute two online user-dependent selection algorithms for the recommender system to choose which advice giver

makes the highest accuracy predictions for each user. Finally, we validate both our initial findings and the selection algorithms with a fourth recommender system data set and conclude that each user benefits from the user-dependent selection rather than a recommender system that uses one type of advice giver.

Advice Givers

A recommender system is comprised of a set of products with corresponding domain information, reviews R and an advice giver. The set of reviews R is an $M \times N$ matrix of M reviewers and N products. The review R_{ij} is a discrete value $v \in V$ that reviewer r_i provides for product p_j. Possible values V may be binary $\{0, 1\}$ or ranging over a subset of the integers (*e.g.,* $\{1, 2, 3, 4, 5\}$). Each product p_j is assigned a domain $d \in D$.

Domain-Specific Advice Giver

An advice giver's task is to provide personalized predictions $v \in V$ of products p that a user u requests. In order to provide personalized predictions for each user, the *Domain-Specific Advice Giver (DSAG)* assigns a weight $w_i^{u,d}$ to each reviewer for each user and domain d (See Algorithm 1). The weight of a reviewer is related to how often the reviewer gave review values similar to the user's reviews and are modeled after experts algorithms ((Auer et al. 1995; Littlestone and Warmuth 1994)) which have been used widely in predicting reviews (*e.g.,* (Nakamura and Abe 1998)). These weights are initially uniform across the reviewers for each user (Line 1)

$$\forall u, d, i \; w_i^{u,d} = 1/M \tag{1}$$

The reviewers are the "experts," and the advice giver makes a prediction by polling the reviewers as a function of the weights assigned to them. The advice giver uses weighted majority to make a prediction for a product in domain d_k that a user requests, by summing the weights of reviewers that provide each value v, and predicts the value with the most weight:

$$argmax_v \sum_i I(R_{ij} == v) * w_i^{u,k} \tag{2}$$

where I is the identity function that returns $I(true) = 1$ and $I(false) = 0$ (Lines 2-4) ((Littlestone and Warmuth 1994)). The advice giver updates the weights as a function of the distance between the reviewer's reviews and the user u's later actual review a_j for the product p_j (Lines 5,6):

$$w_i^{u,k} = \frac{exp(ln(w_i^{u,k}) - \ell_1(R_{ij}, a_j))}{\sum_h exp(ln(w_h^{u,k}) - \ell_1(R_{hj}, a_j))} \tag{3}$$

Because the advice giver makes predictions about the user's review but does not know the actual review ahead of time, the weights for the domain d_k are recalculated online as the user provides reviews for products in that domain. We implement the sleeping experts algorithm to reweigh only the reviewers that provided a review for the product (Freund et al. 1997; Blum and Mansour 2005). If a reviewer does

---

Algorithm 1 Domain-Specific Advice Giver (DSAG)

1: For a new user u, initialize $w_i^{u,c}$ according to (1)
2: **for all** products p_j **do**
3:    $k \leftarrow$ domain(p_j)
4:    Predict according to (2)
5:    **if** user u gives review a_j **then**
6:       Update $w_i^{u,k}$ according to (3)
7:    **end if**
8: **end for**

---

not provide a review for the product p_j, its weight does not change. The goal of the advice giver is to weigh the reviewers for each user such that the resulting predictions are as accurate as possible compared to the hindsight knowledge of the users' reviews.

Mediated Advice Giver

The DSAG can provide precise predictions for users in each domain, but it requires enough reviews in R to cover all products with enough reviewers and requires the user provide enough reviews in each domain to reweigh the reviewers enough times for the weights to converge. In typical recommender systems, however, the review matrix R is very sparse in both the number of reviews provided for a particular product and the number of reviews provided by a particular reviewer. Because of this sparsity, the number of reviewers that get reweighed for any given product that the user requests is far fewer than the total number of reviewers. As a result, the DSAG requires the user to review a lot of products before it can provide accurate enough predictions. This problem is exacerbated because the products are often split into domains and the algorithms require the user to review the same number of products in each domain to predict accurately in each. The focus of much recommender system research has centered around resolving this data sparsity problem (Adomavicius and Tuzhilin 2005).

While most work has focused on *hybrid* recommender systems to increase accuracy by combining different weighing techniques (*e.g.,* (Burke 2002; Umyarov and Tuzhilin 2007)), one recent idea is to combine domains to increase the number of products that affect the reviewers' weights. One idea is to keep domain-specific weights, but to allow the DSAG to reference all of a reviewer's weights to determine if that reviewer is similar to the user in any domain (Berkovsky, Kuflik, and Ricci 2007b). If there is not enough information about a particular domain to make sug-

---

Algorithm 2 Mediated Advice Giver (MAG)

1: For a new user u, initialize $\forall i \; w_i^u \leftarrow \frac{1}{|M|}$
2: **for all** products p_j **do**
3:    Predict $v = argmax_v \sum_i I(R_{ij} == v) * w_i^u$
4:    **if** user reports review a_j **then**
5:       $w_i^u \leftarrow \frac{exp(ln(w_i^u) - \ell_1(R_{ij}, a_j))}{\sum_h exp(ln(w_h^u) - \ell_1(R_{hj}, a_j))}$
6:    **end if**
7: **end for**

(a)				
	p_1	p_2	p_3	p_4
R	d_1	d_2	d_1	d_2
r_1	5	2	5	3
r_2	3	5	2	5
r_3	3	2	2	3
u	5	5	5	5

(b)					
$w^{u,c}$	t_0	t_1	t_2	t_3	t_4
$w_1^{u,1}$	.33	.78	.78	.99	.99
$w_2^{u,1}$	.33	.11	.11	.005	.005
$w_3^{u,1}$	.33	.11	.11	.005	.005
$w_1^{u,2}$	.33	.33	.045	.045	.005
$w_2^{u,2}$	.33	.33	.91	.91	.99
$w_3^{u,2}$	.33	.33	.045	.045	.005

(c)					
w^u	t_0	t_1	t_2	t_3	t_4
w_1^u	.33	.78	.25	.87	.498
w_2^u	.33	.11	.71	.12	.498
w_3^u	.33	.11	.04	.01	.004

Table 1: Example. (a) The review matrix R contains 4 products and 3 reviewers. (b) The DSAG recalculates domain-specific weights as the user provides their actual reviews to find that r_1 and r_2 are most similar for domains d_1 and d_2, respectively. (c) The MAG recalculates the single set of weights as the user provides reviews and finds r_1 and r_2 to be equally similar.

gestions about a product, the system could take advantage of the user's similar reviewers in other domains to make predictions. *Mediation*, on the other hand, combines the weights from multiple domains together (Berkovsky, Kuflik, and Ricci 2007a).

A Mediated Advice Giver (MAG) makes domain-independent predictions with the expectation that the advice giver will identify the most similar reviewers sooner, and provide more accurate predictions with sparse matrices, because all products affect the same weights. The Mediated Advice Giver (Algorithm 2) is the same as the DSAG, except that a single set of weights is maintained which is updated for every product in every domain. The MAG is accurate when there is a consistent set of similar reviewers to a user for every domain (reviewers have proportional weights in all domains). However, when different reviewers have high weights in different domains, the MAG weighs all equally, which can result in poor predictions. Intuitively, if a user had reviews similar to one set of reviewers about DVDs and very different reviews about books, the DSAG, which holds different weights for DVDs and books, would provide more accurate predictions.

Example Suppose a new user joins a recommender system that uses a Domain-Specific Advice Giver with three reviewers ($M = 3$) r_1, r_2, r_3, that review four products ($N = 4$) p_1, p_2, p_3, p_4 with reviews presented in Table 1(a). The values are shown for later use in the example. $D = \{d_1, d_2\}$ are assigned to the products and the possible values that the reviewer can give and advice giver can predict are $V = \{1, 2, 3, 4, 5\}$. The DSAG initializes the weights $w_i^{u,d}$ to $1/M = 1/3$ for both domains (Table 1(b) column t_0). The user requests a prediction for product p_1 in domain d_1. The advice giver calculates which value to predict using the initial weights and chooses to predict 3 because it has the maximum weight associated with it. The user notifies the advice giver that their actual review r_1 of p_1 is 5 (last row of Table 1(a)) and the advice giver uses that information to reweigh the reviewers for domain d_1 (Table 1(b) column t_1). Then the user requests a prediction for p_2, and the advice giver uses the new initialized weights for domain d_2 to predict 2. The user responds with value 5 and the advice giver recalculates the weights from domain d_2 (column t_2). This continues for all 4 products.

The weights (shown in Table 1(b)) do not change on the

time steps when the advice giver predicts a value for a product that is from a different domain. At the end, we can see that the user has the same reviews as r_1 for domain d_1 and the same reviews for r_2 as d_2 and the DSAG correctly identifies them as most similar by assigning them highest weight.

Now suppose that R is the same, but the recommender system uses a MAG to predict for the user. Because reviewers 1 and 2 are each correct 50% of the time, their weights change over the four products (See Table 1(c)). The MAG assigns higher weight to the wrong reviewer and predicts 2 or 3 when it should predict 5 for all products.

Approach

The focus of this work is to determine whether data sparsity consistently affects the accuracy of the DSAG for all users and choose the most accurate advice giver for each user. We will first show, using synthetic data, different weight distributions for users in the MAG and DSAG and analyze their prediction accuracies. We then show, using three real recommender system data sets, that both the MAG and DSAG give more accurate predictions for some users. Additionally, for very sparse data sets, we find that both advice givers can give identical results. We analyze this phenomenon and provide accuracy results on synthetic data sets with these properties. Because neither advice giver can be excluded as less accurate, we provide two algorithms to dynamically select which advice giver to use for each user and validate our results and algorithms on a fourth recommender data set.

Advice Giver Evaluation - Synthetic Data

In this section, we evaluate the worst-case and more realistic review matrices and user preferences to better understand the performance of the MAG and DSAG under different amounts of data sparsity and different reviewers. We show that the MAG converges faster on the weight distribution, assuming those best reviewers are the same across categories. Then, we will show a more extreme case of the example above where a user agrees strongly with one reviewer in each category and disagrees strongly with the rest, causing the DSAG to perform better than the MAG. When we relax the constraint of a different "best" reviewer in each category, the MAG and DSAG perform equally well. Finally, we relax the assumption that all reviewers review all products and explore very sparse review matrices. We show

that in cases where reviewers only review products in a particular category, the two advice givers perform equally well. We will use these results later to analyze our results from three real recommender system data sets.

Sparse Data

For the following examples, we will assume that users have polar preferences - either strongly disliking (1) or strongly liking (5) each product. For simplicity, the user will always strongly like the product and give it a 5. Also for simplicity, we will only have m reviewers, m categories, and $n >> m$ products that are evenly distributed across the categories.

The MAG can converge on a single weight distribution using all of the products while the DSAG instead uses m weight distributions - one for each category. Assuming that each of the m categorical weight distributions are similar, the DSAG will converge on each distribution separately although it turns out they should all be the same. If each weight distribution takes the same amount of time to converge, the DSAG will take longer to come to the same conclusion as the MAG. If the products are not distributed evenly across categories, it could take the DSAG much longer to converge on the rare categories. As an example, we define the review matrix in the following way:

$$R_{ij} = \begin{cases} 5 & (p \wedge i = 1) \\ 1 & ((1-p) \wedge i = 1) \\ 1 & i \neq 1 \end{cases}$$

For all categories, reviewer 1 is correct p percent of the time and gives the same review as the rest of the reviewers (1-p) percent of the time. The DSAG will have to find this pattern for each weight distribution while the MAG only finds it once. Because this is a simple distribution and all reviewers give reviews for all products, it takes relatively few products to find the pattern. The MAG finds the right weight distribution after $1/p$ products while the DSAG takes m/p products to converge all distributions.

In general, it takes $1/p$ product reviews to find the weight of each reviewer for each weight distribution. If not all reviewers provide reviews for each product, it could take much longer to converge. The MAG also assumes that the weight distributions for each category are similar. If they are not, combining them together into the single distribution may cause prediction errors.

Categorical Weight Distributions

The DSAG can perform well compared to the MAG when the reviewers have very different weights in each category as shown in the example above. As a more extreme example, we define the review matrix in the following way:

$$R_{ij} = \begin{cases} 5 & \text{product j in category i} \\ 1 & \text{product j not in category i} \end{cases}$$

It is quite obvious to see that if the user always says 5, that reviewer i always has the highest weight in category i and the rest of the reviewers have almost 0 weight. The weight distribution for each category is very different. The DSAG

converges on the correct weights very quickly because of the high degree of similarity between reviewers and the user. The MAG, alternatively, sees an equal number of products in each category and converges on a uniform weight distribution across reviewers. Because more reviewers recommend the value 1 for each product, the MAG predicts incorrectly every time. We tested this hypothesis with data generated with the above rule on recommendation systems with the weighted majority algorithm. The error rates of the systems were calculated. The similarity distributions were also examined at the end of the trials to compare to the expected distribution.

Advice Giver	Accuracy
DSAG	100%
MAG	0%

Table 2: The domain-specific advice giver is 100% accurate while the mediated advice giver is 0% accurate.

Table 2 shows the prediction accuracy by advice giver. As expected, there is a significant drop in accuracy when combining the domain-specific advice givers. The MAG converges on a uniform weight distribution and gives the wrong advice to the user for every product. The DSAG does not. Next, we relax the requirement that each category have one extremely accurate reviewer to understand how the two advice givers predict as the "best" reviewer becomes less obvious.

Changing Weight Distributions

The greater the difference in weight distributions across different categories, the worse the MAG predicts. We have shown that the MAG can perform significantly worse than the DSAG in this situation. We will now show how the DSAG and MAG predict equally as the reviewers' weights in each category converge to the same distribution. In other words, in the previous example, it is 100% likely that reviewer i will predict correctly in for products in category i and there is a 0% chance that any other user will be correct for that product. Now, we create a review matrix based on a probability p that reviewer i is the "best" reviewer for category i in the following way:

$$R_{ij} = \begin{cases} 5 & (p \wedge \text{ product j in category i}) \\ 1 & ((1-p) \wedge \text{ product j in category i}) \\ 5 & ((1-p) \wedge \text{ product j not in category i}) \\ 1 & (p \wedge \text{ product j not in category i}) \end{cases}$$

With probability p, the reviewer i gives review 5 and the rest of the reviewers give review 1. Otherwise, some other reviewer gives review 5 and the rest give review 1. As it becomes more likely that the reviewer that gives 5 is random, the weight distribution for each category becomes more uniform. The MAG's weight distribution is uniform in all cases as before. It is important to note that if the "best" reviewer is chosen any other way than uniform random, both algorithms would perform better than random because there is a reviewer with higher weight.
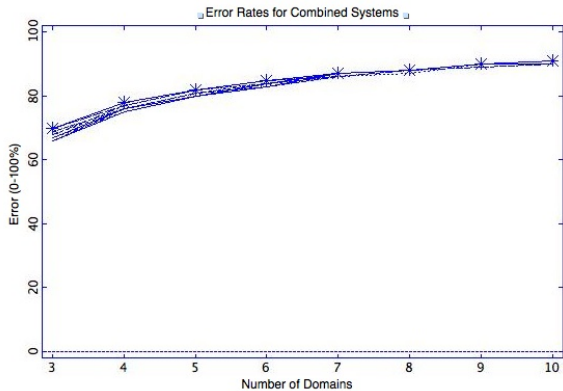
Figure 1: The MAG's % Error over 1000 products consistently is high because of the uniform random weight distribution. X-axis: number of categories and reviewers. Each line: different probability p
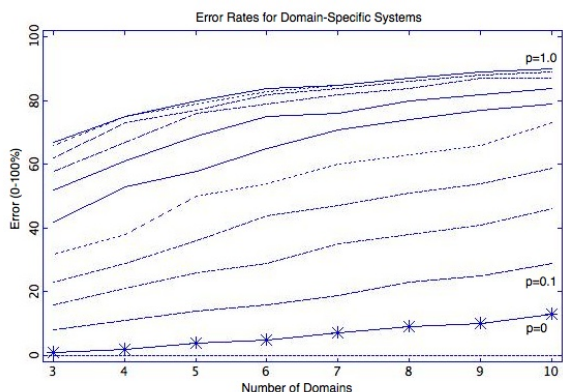

Figure 2: The DSAG's % Error over 1000 products gets worse as the weight distribution becomes uniform random. X-axis: number of categories and reviewers. Each line: different probability p (p=0 on bottom, p=1 on top)

Figures 1 and 2 show the error rates of the MAG and DSAG respectively as we vary $0 \leq p \leq 1$ in increments of .1 and the number of categories from 3 to 10. As the number of categories increases, there are more weight distributions that must each converge, increasing the error rate overall until they do. When $p = 0$, the review matrix is the same as the DSAG example above and the DSAG again has perfect accuracy after converging while the MAG is almost always wrong. As p approaches 1, the domain-specific weights become more uniformly random. However, as we varied p, the mediated weight distribution over the reviewers was consistently uniformly random and the error rate was consistently high. Because it is not holding category information, the MAG doesn't find a difference between truly random ($p = 1$) and domain-correlated ($p = 0$) reviews and predicts the same way each time. The DSAG also has a high error rate as the weight distributions become more uniform. When the MAG and DSAG weight distributions become more similar (to uniform or otherwise), the DSAG and MAG predict the same. The DSAG performs better than the MAG until the distributions are more than 50% similar.

Very Sparse Review Matrices

The previous tests assumed that reviewers gave reviews for all products and all categories. However, this is a strong assumption. Now we relax this assumption and look at much sparser review matrices. For simplicity, we assume that only a single reviewer provides reviews in each category, although this can easily be scaled up to include disjoint *sets* of reviewers. Here, we will also assume that the single reviewer has similar reviews as the user and both give reviews of 5, although we could easily assume that a few of many reviewers are similar. We define the entries of the review matrix in the following way:

$$R_{ij} = \begin{cases} 5 & \text{product j in category i} \\ - & \text{product j not in category i} \end{cases}$$

The DSAG in this case only has a single reviewer to assign a weight for each category and gives it full weight. The MAG, however, must combine these reviewers together into a single weight distribution. Because the products are uniformly distributed in the categories, the reviewers are also uniformly weighted. However, unlike the previous examples where the MAG had a high error rate, the MAG performs equally as well as the DSAG here. Because the MAG only has one review in the product column to compare against, it always picks that review to follow and is 100% accurate - the same as the DSAG.

We have shown in synthetic, simplified review matrices that the MAG can converge on the weight distribution faster than the DSAG with less data and that the DSAG is more accurate when the categorical weight distributions are different. However, we have also shown that the two advice givers perform equally when the weight distributions are the same across categories or when the review matrix is sparse and there are disjoint sets of reviewers for each category. Next, we analyze the properties of actual recommender systems to understand how the two advice givers perform in these real conditions with real data sparsity and real reviewers. It is unclear whether the advice givers even perform the same for two different users of the same recommender system. We use these results to determine how the tradeoff between data and weight distributions affects the accuracy of predictions for different users.

Advice Giver Evaluation - Real Data

To understand how often data sparsity affects the accuracy of the DSAG for users, we built two recommender systems using real data sets from popular recommender websites. We compare the accuracy of the MAG to the DSAG across all users and for each user, and evaluate how the weights of the reviewers affect the success of the MAG. First, we describe the recommender systems' review matrices R.

Experimental Method

The reviewers and advice giver provide possible values $V = \{1, 2, 3, 4, 5\}$ to the user. For each recommender system, the MAG and DSAG provide predictions for the same randomly chosen users. We use the users' reviews of products as the truth for accuracy calculations and remove those reviews

from R. In order to provide both advice givers with the highest chance of convergence, both were given the users' reviews for every product immediately after the prediction. We now present the individual characteristics of three recommender system data sets and the users in those systems.

The 2007 Netflix data set contains over 100 million movie ratings for over 480 thousand Netflix customers from over 17000 movie titles (Netflix 2007). The DSAG used movie genres as domains. Because Netflix did not include genre information in the data set, we cross-referenced movie titles with movie genres and obtained the set the DSAG used: Science Fiction, Fantasy, Special Interest, Kids/Family, Comedy, Western, Action/Adventure, Musical/Performing Arts, Drama, Documentary, and Foreign. Only the movies with a single genre were used, which resulted in a smaller data set of $N = 400$ movies, over 100,000 reviewers, and over 4 million reviews. Approximately 1% of the review matrix R was filled.

The Yahoo! Research Webscope Music review data set contains over 717 million reviews by $M = 1.8$ million Yahoo! Music reviewers for $N = 136,000$ songs collected from 2002 to 2006 (Webscope 2008b). Each reviewer was guaranteed to have reviewed at least 30 songs and each song was reviewed by at least 20 reviewers, sparse in comparison to the total number of songs and reviewers. The DSAG uses 20 main genres of music provided in the data set. For each new user, there were an average of 100,000 reviewers giving reviews for their songs.

The final General Products data set was collected from a popular online shopping website (Leskovec, Adamic, and Huberman 2006). Over two years of data collection, 3,943,084 reviewers made 15,646,121 reviews for 548,523 products. 5813 products were discontinued and were thus removed from the set. The DSAG used the ten product categories defined in the dataset (*i.e.,* Book, DVD, Electronics). For each recommendation, the dataset provides the ID of the reviewer, the numerical recommendation as defined above, the date that the recommendation was given, and the product that was reviewed. On average, users made recommendations for about 100 products, with one user recommending over 12000 and each product had between 3 and 2500 recommendations. Approximately .007% of the review matrix is filled.

Results

To evaluate the accuracy of each advice giver, we compare the mean squared error (MSE) of the advice givers' predictions to the users' reviews to capture the distance between the values. We test whether there is a difference between the MSEs of the MAG and DSAG using an ANOVA (Analysis of Variance). 50 users from Netflix and the General Products and 20 users from Yahoo! Music were randomly chosen from the sets to test. We divide this analysis up by the sparsity of the dataset - General Products are sparse, while Netflix and Yahoo! are not as sparse.

Results for Very Sparse Dataset   For the General Product data set, our experimental results show there was no significant difference (MAG ΔMSE = .47, DSAG ΔMSE =

.53) ($F = .59, df = 1, p > .05$) between the DSAG and MAG. In other words, there is no increase in accuracy as we combine the domain-specific advice givers for each product genre together into a single mediated advice giver. While it is expected that users of multiple systems make reviews for all of them so combining the systems results in more data, we found that users tended to focus their reviews on a specific category of products instead of reviewing products in all categories.

Although the entire user set was shared for all of the domains in the General Product dataset, the set of users that gave reviews for products in one category most likely did not give reviews in any other category. If the most similar users (with the highest weight) in each domain are disjoint, then both DSAG and MAG give the same predictions because the relative weights of the reviewers in each category are the same for both advice givers. We found that the overall performance does not change because given the sparse data, each reviewer only has one accurate weight in one category - the one they provide reviews in. When the weight distributions of the DSAG are combined into the single MAG distribution, the single category weight is carried into the new distribution (and normalized) without the need to average multiple weights together because the weight in the rest of the categories have not changed from the initialized value. The MAG weight distribution is the same as, and not better than, each DSAG.

Results for Other Datasets   For the Yahoo! and Netflix data sets, our experimental results show that there is no statistically significant difference between the MSEs of the MAG (MSE=2, std. dev. = .5) and DSAG (MSE=1.8, std. dev = .3) for all users combined ($F = 2.75, df = 1, p > 0.05$). We do find significant differences in the models for individual users. For half of the users in both data sets, the DSAG had a statistically significant lower MSE (ΔMSE = .5, std. dev = .4) than the MAG ($F = 7.14, df = 1, p < 0.01$), but for the other half of the users, the MAG was more accurate (ΔMSE = .25, std. dev. = .2) although this was not statistically significant.

We found (as expected) that the MAG has a higher accuracy than the DSAG when there were not enough products reviewed by the user for the DSAG's domain-specific weights to converge. The DSAG provides more accurate predictions when different reviewers have the highest weight in different domains (See Example in Section 2). Additionally, users for which the DSAG made better predictions typically focused their product requests and reviews in a few domains and did not require all domains, allowing those domain weights to converge quickly. Because half of the users benefitted from each advice giver, neither advice giver should be used to make predictions for all users. We propose an online selection algorithm to autonomously determine which advice giver should make predictions for each user.

We have seen in both synthetic and real recommender system data that when review matrices are very sparse, both advice givers perform the same because the data is both sparse *and* the weight distributions are disjoint. Next, we will an-

alyze the number of categories a reviewer must provide reviews in, in order to see a difference in accuracy between the DSAG and MAG advice givers. Then, assuming reviewers provide enough reviews so that the data is not as sparse, we propose two online selection algorithms to autonomously determine which advice giver should make predictions for each user.

Very Sparse Data

We have shown that in very sparse matrices, there may not be any overlap in reviewers across categories. In other words, reviewers only review products in a single category. The MAG and DSAG produce different weight distributions but the same predictions with the same accuracy. Now, we will show how the MAG is affected as reviewers review products in more categories and as we can collect more data to overcome the extreme sparseness of the review data. We will use the same assumptions as before. The user always gives review 5, so the reviewer that reviews a product with value 5 should have the highest weight.

Different Weight Distributions

First, instead of completely disjoint sets of reviewers for each category, our reviewers review products in a set of categories. Only a single reviewer is correct in each category but several other reviewers also give incorrect reviews. More concretely, we define the review matrix in the following way:

$$R_{ij} = \begin{cases} 5 & \text{product j in category i} \\ 1 & \text{product j in categories (i+1)\%m - (i+k)\%m} \\ - & \text{otherwise} \end{cases}$$

Reviewer i is correct for category i and gives reviews far from the user's preference for categories (i+1)%m to (i+k)%m. The value k is the number of reviewers that provide reviews for each product. By varying k, we can understand how the MAG is affected as the amount of overlap increases in the categories that reviewers review for. We notice that $k = 0$ means that only a single reviewer reviews products in a single category. Also, $k = m$ means that all reviewers provide reviews for all products. We are interested in how the MAG behaves for values between 0 and m. We know that although it may take the DSAG longer to converge on the categorical weight distributions, it will always have 100% accuracy.

Figure 3 shows the error rate (y-axis) for the MAG and DSAG with $m = \{1, ..., 10\}$ (x-axis) and with $k = \{1, ..., 10\}$. Each line represents a different k. Note that there cannot be more than m reviewers overlapping at a time, so the line for k starts at m. The line for 3 overlapping reviewers starts at 100% error rate at $m = 3$ and drops a little before converging at a high error rate. Because all reviewers are "best" for one category, the MAG converges to a uniform distribution. A majority of the reviewers give reviews that conflict with the user, but the MAG follows the majority so it predicts incorrectly nearly all the time depending on
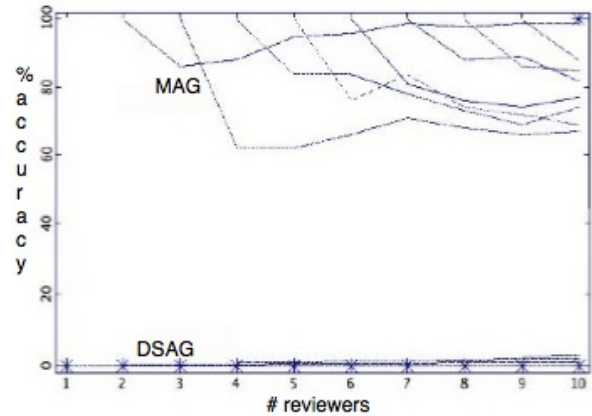

Figure 3: The DSAG has almost 0% error no matter how many reviewers review each product. The MAG has perfect accuracy when the reviewers in each category are disjoint ($k = 1$). However, it is almost always wrong for $k > 1$ and any amount of overlap with reviewers.

the order of products. The DSAG always finds the correct weight distribution and maintains a 0% error over all k.

There is no statistical significance between the error rates of the ks. As we increase k, the error is nearly constant. However this error is expected because the weight distributions are so different for each category. Next, we evaluate the MAG performance on categorical weight distributions that are similar but have sparse reviewer data.

Similar Weight Distributions

We know that the MAG converges faster than the DSAG when the categorical weight distributions are similar and all reviewers always give reviews. Now we create a new review matrix such that there are a few (*e.g.*, 2) "best" reviewers for a user and the rest give opposing reviews:

$$R_{ij} = \begin{cases} 5 & (p \wedge i = 1) \\ 5 & ((1 - p) \wedge i = 2) \\ 1 & (q \wedge i \neq 1) \\ - & \text{otherwise} \end{cases}$$

With some probability p, reviewer 1 gives the best response and otherwise reviewer 2 gives the best response. Then the rest of the reviewers each give a response with some other probability q and otherwise give no response. These distributions do not depend on the category, so the DSAG would have to find this same distribution for each category. The MAG finds it just once.

If either reviewer 1 or reviewer 2 always provides a review for a product, the MAG can find these best reviewers after getting reviews from each of them only a few times. Because these reviewers are always correct with respect to our user, a MAG that always trusts them will also always be correct. Although these reviewers never appear at the same time, the MAG still has one reviewer that has high weight to

use. The rest of the reviewers have low weight and are not included in the prediction. If there are products for which neither of the "best" reviewers provide reviews, there is no way for the MAG to produce a good answer. The MAG can be as accurate and converge on the reviewer weights faster than the DSAG when the similarity distributions are the same and reviewers do provide reviews for products in multiple categories.

We have shown that the MAG and DSAG are each more accurate for some users, depending on the products they choose from the review matrix. In very sparse review matrices where reviewers provide reviews in only a single category, both the MAG and DSAG have the same accuracy. If we can increase the amount of data to get reviewers to review products in multiple categories, both the DSAG and MAG can perform well under some conditions. Because it is not known which advice giver will be more accurate when a user joins a recommender system, the recommender system must calculate predictions of both advice givers as the user requests reviews and provides reviews to determine which is more accurate. We present two online selection algorithms for the recommender system to use in determining which advice giver (DSAG or MAG) makes the best predictions for each user.

User-Dependent Online Selection Algorithm

First, we show a user-dependent online selection algorithm (UdOS) which picks the single best advice giver for each user. The UdOS Algorithm (Algorithm 3) first initializes a new DSAG and MAG for the new user and sets their accuracies to 0 (Steps 1-2). Because the MAG provides better predictions when there are fewer data points available, the MAG's predictions are provided to each user to start (Step 3). Additionally, the algorithm requires that the domain-specific weights converge before making a decision. It has been shown that the weighted majority algorithm requires approximately $\lceil log(M) \rceil$ user reviews to converge (Littlestone and Warmuth 1994). However, the advice givers actually require at least one review from each advice giver find the most accurate weights. Thus, the algorithm takes linear time $O(M)$ in the number of reviewers to converge. The count $num\_usr\_reviews$ of the current number of reviews is initialized to 0 (Step 4).

Until the recommender system has received enough user reviews, the UdOS algorithm makes predictions for the user with the current most accurate advice giver (Steps 5-17). The user requests a product p_j to be predicted by the recommender system (Step 6). The algorithm gets both the MAG and DSAG predictions (Steps 7-8), but gives the user only the prediction from the current best advice giver (Step 9). If the recommender system receives the user's actual review a_j, the algorithm increases the count of the user reviews, and recalculates the best advice giver using the new accuracy (Steps 10-16). After enough reviews have been collected from the user, UdOS maintains only the best advice giver, which continues to reweigh reviewers (Steps 18-21).

Using the UdOS algorithm for each user, the recommender system can decide whether the data sparsity or reviewers' weights affect the prediction accuracy more and

---

Algorithm 3 User-Dependent Selection Algorithm

1: $dsag \leftarrow$ new $DSAG(u)$, $mag \leftarrow$ new $MAG(u)$
2: $acc_{MAG} \leftarrow 0$, $acc_{DSAG} \leftarrow 0$
3: $BestAG \leftarrow mag$
4: $num\_usr\_reviews \leftarrow 0$
5: **while** $num\_usr\_reviews < M$ **do**
6:     $p_j \leftarrow getProductRequest()$
7:     $pred_{MAG} \leftarrow mag.predict(p_j)$
8:     $pred_{DSAG} \leftarrow dsag.predict(p_j)$
9:     **print** $pred_{BestAG}$
10:     **if** receive a_j **then**
11:         $num\_usr\_reviews$ **++**
12:         $acc_{MAG} \leftarrow calcAcc(mag)$
13:         $acc_{DSAG} \leftarrow calcAcc(dsag)$
14:         $mag.update(a_j)$, $dsag.update(a_j)$
15:         $BestAG \leftarrow$ if $(acc_{MAG} < acc_{DSAG})$?mag : $dsag$
16:     **end if**
17: **end while**
18: **loop**
19:     $p \leftarrow getProductRequest()$
20:     **print** $BestAG.predict(p)$
21: **end loop**

---

choose the best advice giver. Although it is less computationally efficient to calculate the predictions from both the DSAG and MAG, because the algorithm picks the single best advice giver, the recommender system provides better predictions for all users. After making the selection, it saves memory and computation by maintaining only the best one.

User- and Category-Dependent Online Selection Algorithm

Our second selection algorithm picks the best advice giver for each category (UCdOS). If the DSAG converges for one or a few categories because the user focuses on those categories, this algorithm can pick the DSAG for those that have converged and continue to use the MAG for all other categories. The UCdOS Algorithm (Algorithm 4) first initializes a new DSAG and MAG for the new user and sets their accuracies to 0 (Steps 1-5) as in the UdOS algorithm.

Until recommender system has received enough user reviews, the UCdOS algorithm makes predictions for the user with the current most accurate advice giver for each category (Steps 5-18). The user requests a product p_j in category c_k to be predicted by the recommender system (Steps 6-7). The algorithm gets both the MAG and DSAG predictions (Steps 8-9), but gives the user only the prediction from the current best advice giver for category c_k (Step 10). If the recommender system receives the user's actual review a_j, the algorithm increases the count of the user reviews, and recalculates the best advice giver using the new accuracy (Steps 11-17). After enough reviews have been collected from the user, UCdOS maintains only the best advice giver for each category, which continues to reweigh reviewers (Steps 19-23).

Using the UCdOS algorithm for each user, the recom-

Algorithm 4 User- and Category-Dependent Selection Alg.

1: $dsag \leftarrow$ new $DSAG(u)$, $mag \leftarrow$ new $MAG(u)$
2: $acc_{MAG} \leftarrow 0$, $acc_{DSAG} \leftarrow 0$
3: $BestAG \leftarrow mag$
4: $num\_usr\_reviews \leftarrow 0$
5: **while** $num\_usr\_reviews < \lceil log(M) \rceil$ **do**
6:      $p_j \leftarrow getProductRequest()$
7:      $c_k \leftarrow p_j.getCategory()$
8:      $pred_{MAG} \leftarrow mag.predict(p_j)$
9:      $pred_{DSAG} \leftarrow dsag.predict(p_j)$
10:     **print** $pred_{BestAG[c_k]}$
11:     **if** receive a_j **then**
12:         $num\_usr\_reviews$ ++
13:         $acc_M \leftarrow calcAcc(mag)$
14:         $acc_{DS} \leftarrow calcAcc(dsag)$
15:         $mag.update(a_j)$, $dsag.update(a_j)$
16:         $BestAG[c_k] \leftarrow$ if $(acc_M < acc_{DS})$?mag : $dsag$
17:     **end if**
18: **end while**
19: **loop**
20:     $p \leftarrow getProductRequest()$
21:     $c \leftarrow p.getCategory()$
22:     **print** $BestAG[c].predict(p)$
23: **end loop**

mender system can decide for each category whether the data sparsity or reviewers' weights affect the prediction accuracy more and choose the best advice giver on a category specific basis. This allows for more flexibility in the products the user picks. The UdOS algorithm requires that the user pick products from all categories uniformly in order to determine whether the DSAG or MAG is better overall for all categories. In the UCdOS algorithm, even if users do not pick products uniformly across categories, the most appropriate advice giver is chosen. If a user does not request products in some categories, there is less data to train from and the MAG would produce better results. When a weight distribution for a category has converged, the DSAG is more appropriate to use and the UCdOS algorithm uses that.

The UCdOS is less computationally efficient than the UdOS, because it must maintain the DSAG and MAG until all of the DSAG weight distributions converge, but if it means that it provides better predictions than any other algorithm it may be worth it.

Validation

The UdOS and UCdOS algorithms will choose the advice giver with the highest accuracy after enough products are reviewed by the user. Ideally, the accuracy of the predictions from the algorithm will be as good as the best advice giver even through the selection process in order to maintain user satisfaction with the recommender system. In order to validate that the UdOS and UCdOS algorithms both pick the better advice giver and maintain high accuracy, we tested them against the MAG and DSAG advice givers on a fourth recommender system data set. We calculate accuracy over time to understand how both selection algorithm per-

form and show that they finds the best advice giver for each user while minimizing errors.

Experimental Method

The Yahoo! Movies data set contains 7,642 users, 11,915 movies and 211,231 reviews (Webscope 2008a). We use the same movie genres for domains as the Netflix movies data set. Only the movies with a single genre were used which resulted in a smaller data set of 9,000 reviewers for 1000 movies. Only 0.23% of the review matrix R was filled. Each product was reviewed by at least two reviewers.

We use the same recommender system setup as the experimental results above. The advice givers and selection algorithms provide possible values $V = \{1, 2, 3, 4, 5\}$ for the same randomly chosen users. The advice givers received the users' reviews after every product prediction and use the reviews, which were removed from R, to reweigh reviewers. We report the MSE instead of accuracy because it better represents the error distance between the prediction and users' reviews. Additionally, because we did not have enough data to wait for all of the M reviews, we used $logM$ as the parameter to wait before converging on the best advice giver so that we could report a best advice giver for each user. In general, it became obvious which advice giver to choose after waiting for only $logM$ reviews for this data set although for some users it was not sufficient.

We found that the $calcAcc$ function should place more weight on the later predictions and less weight on the earlier predictions because the advice giver was more likely to be incorrect with fewer user reviews when determining the best advice giver. We used a step function as follows although we found that any linear function or step function with a different span produced the same results.

$$calcAcc(ag) = avg(\text{accuracy recent 10 predictions})$$

If the DSAG makes poor predictions for new domains at the beginning of the product set, the UdOS and UCdOS will always determine that it has a lower accuracy. We use both the cumulative (unweighted) and weighted MSEs in the results. Twenty users were chosen at random as test users. Each user requested predictions for between 20 and 130 products. The MAG, DSAG, and UdOS and UCdOS algorithms' predictions were recorded for each user and used to calculate each unweighted MSE (see Tables 3 and 4).

UdOS Results

Overall, thirteen users received better predictions from the DSAG and 6 received better predictions from the MAG. One user received equivalent predictions from both advice givers. Our results show that for all 20 users the UdOS algorithm picked the best advice giver. Furthermore, the UdOS algorithm's error was no worse than the worst advice giver and at times can be better than the best advice giver. We will now describe in more detail how the UdOS algorithm chose the advice givers for some users.

For User 1, data sparsity affected the predictions from the DSAG and he received better predictions from the MAG, so the UdOS algorithm continued using the MAG for the duration of the product set. User 19 received better predictions
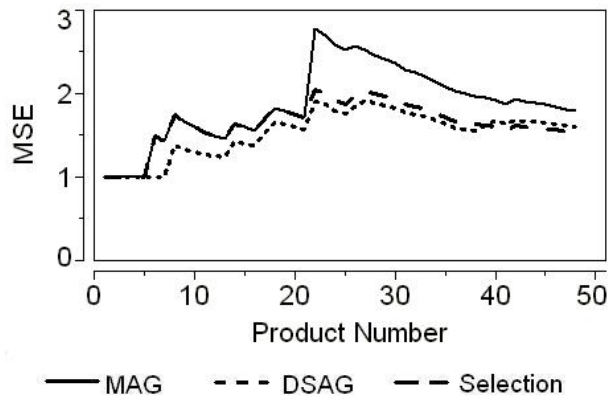
(a) The cumulative MSE for User 8 over time.
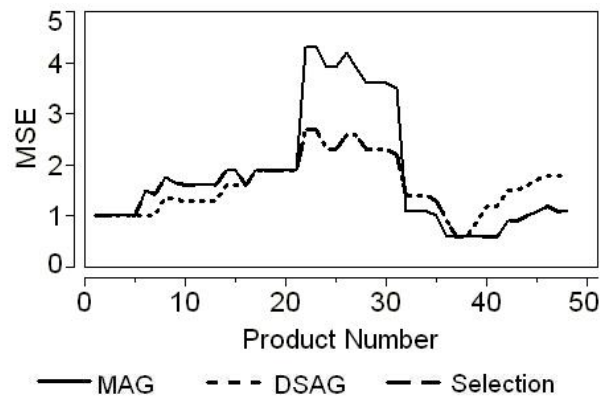
(b) The weighted MSE for User 8 over time.

Figure 5: The UdOS used the DSAG for User 8 on products 5-32 when the MAG was providing worse predictions. After product 32, the MAG provided better predictions and the UdOS switched back to using it.
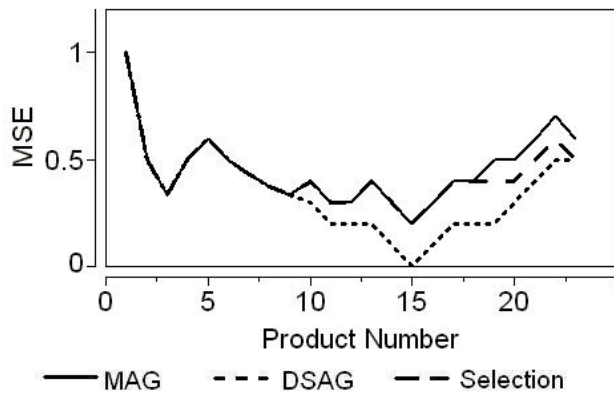

Figure 4: For User 19, the UdOS algorithm uses the MAG advice until it has seen enough data to conclude that the DSAG is better and then switches to use its predictions (product 18).

User	MAG	DSAG	UdOS	Selection
1	**1.09**	1.29	1.09	MAG
2	**1.29**	1.40	1.38	MAG
3	**1.35**	1.49	1.35	MAG
4	**1.32**	1.41	1.32	MAG
5	**1.27**	1.36	1.27	MAG
6	**2.48**	2.48	2.48	MAG**
7	1.09	**1.08**	1.08	MAG*
8	1.78	**1.59**	1.51	MAG*
9	**1.04**	1.14	1.03	DSAG*
10	0.95	**0.92**	0.95	DSAG
11	0.51	**0.38**	0.51	DSAG
12	0.54	**0.49**	0.51	DSAG
13	1.08	**0.58**	0.61	DSAG
14	1.06	**0.81**	1.03	DSAG
15	1.03	**0.97**	1.03	DSAG
16	0.80	**0.73**	0.76	DSAG
17	0.67	**0.63**	0.63	DSAG
18	1.07	**0.89**	1.07	DSAG
19	0.50	**0.38**	0.46	DSAG
20	1.44	**1.37**	1.39	DSAG

Table 3: Six users received better predictions with lower cumulative MSEs (bold) from the MAG while 13 received better predictions from the DSAG.

from the DSAG than the MAG, because the user's similar reviewers were different for each product domain. The UdOS algorithm recognized that the DSAG had a lower MSE (See Figure 4) and switched to using the DSAG's predictions at product 18. After the switch, the UdOS cumulative MSE converged towards that of the DSAG.

The algorithm picked the DSAG for other users like 10, but the switch occurred late enough that the cumulative MSE did not start converging towards the DSAG's MSE. Because the cumulative MSE was the same for both the DSAG and MAG for User 16, it didn't matter which advice giver the UdOS picked (marked with ** in Table 3). As a result, it continually used the MAG that it started with.

The UdOS algorithm picked the advice giver with higher overall MSE for three users (marked with * in Table 3). User 8, for example, received better predictions from the UdOS algorithm than either the MAG or DSAG could provide alone because the UdOS switched back and forth be-

tween the two advice givers before deciding which to use. The MSE of the DSAG is lower than the MAG's because the MAG poorly predicted some products in the middle of the set (specifically products 20-30) (See Figure 5(a)). However, once the MAG's weight distributions identified the most similar reviewers (product 32), its predictions were better than the DSAG's and the UdOS algorithm picked the MAG (See 5(b)).

UCdOS Results

Overall, nine users received the same predictions as the UdOS, picking either one advice giver or the other for all of the product categories the user requested (See Table 4, no *). Note that this does not mean that the DSAG or MAG

was chosen for all categories, just that it picked for all categories that there was data for. If a new product from a new category was queried, while the UdOS would have to use the same advice giver as it picked for the other categories, the UCdOS could use the MAG until enough products in that category were reviewed. This feature means that the UCdOS could be more accurate using the MAG while the DSAG's weight distribution for the new category converges. However it could be less accurate if the MAG distribution is very different than the category weight distribution.

Six of the users received worse predictions from the UCdOS compared to the UdOS, but still at least as good as the worse advice giver (See Table 4, *). In these cases, the MAG's single distribution was different from the DSAG's category distributions. The UCdOS used the MAG for each category until the DSAG converged. The MAG produced worse predictions in the UCdOS than if the DSAG was chosen starting from a uniform distribution in the UdOS. This is a particularly good instance of our initial example that the MAG's poor results when the DSAG's categorical distributions are very different. The UdOS chooses the DSAG as soon as there is any indication that the weight distributions are different for different categories. Because it takes time for each of the DSAG's to converge, it is worse for the UCdOS to use the MAG before the convergence even though it has an indication that the DSAG's weights are different than the MAG's. In this case, waiting ensure the selection algorithm picks the right advice giver for each category actually makes the prediction accuracy worse.

Five reviewers received better predictions than the UdOS, including several who received better predictions than either the DSAG or MAG could provide alone (See Table 4, **). These users had the best benefit from the data sparsity versus weight distribution tradeoff. The UCdOS was able to use the DSAG for the categories that the user queried often and the MAG for the other categories. The DSAG's category weight distributions were similar to the MAG's distribution for some categories and different in others. The UCdOS did not have to pick a single advice giver and could take advantage of the MAG instead of waiting for the category weight distribution to converge to the same values. This sped up the selection process, allowing the UCdOS to be more accurate for longer than either of the two advice givers alone.

Discussion

Both of the user selection algorithms pick the best advice givers with the knowledge they have. The UdOS algorithm picks a single advice giver to use for the rest of the predictions. Once it finds that two different categories have different weight distributions and the DSAG is performing better, it switches to using it. The algorithm makes the assumption that if there are two weight distributions that are different, that all will be different. More importantly, it makes a strong tradeoff that the different distributions are more important than the data sparsity problem. If there are different distributions, the algorithm must find each of the categorical weight distributions and cannot take advantage of the MAG while they converge.

The UCdOS algorithm picks an advice giver for each cat-

User	MAG	DSAG	UCdOS
1	1.09	1.29	1.09
2	1.29	1.40	1.27*
3	1.35	1.49	1.46**
4	1.32	1.41	1.32
5	1.27	1.36	1.27
6	2.48	2.48	2.48
7	1.09	1.08	1.09**
8	1.78	1.59	1.78**
9	1.04	1.14	0.99*
10	0.95	0.92	0.89*
11	0.51	0.38	0.51
12	0.54	0.49	0.51
13	1.08	0.58	1.05**
14	1.06	0.81	1.06**
15	1.03	0.97	1.03
16	0.80	0.73	0.76
17	0.67	0.63	0.63
18	1.07	0.89	0.89*
19	0.50	0.38	0.41*
20	1.44	1.37	1.41**

Table 4: Five users (*) received better predictions from the UCdOS while six users received better predictions from the UdOS (**).

egory. It does not make the assumption that all weight distributions are different, but does assume that the distributions are either like the MAG distribution or not. It can take advantage of the MAG while each of the DSAG distributions are converging, which greatly improved the accuracy of the predictions for five users compared to the UdOS. However, we found that sometimes the MAG is a detriment and using it makes the selection algorithm predict worse than the UdOS when the MAG distribution is different from the category weight distribution. Overall, both selection algorithms do well for all of our test users and there is no clear better algorithm to choose.

These selection algorithms do pick the best advice giver, but still require that the weight distributions either be completely different from each other or the same as the MAG. It is possible that high weight reviewers in DVDs have the same high weight in books because the user likes mysteries in both categories but have very different taste in clothes. One other selection algorithm that could be tested in future work would be one that tests whether any of the converged category weight distributions perform well for a new category. This algorithm would take advantage of all the previous work done to converge the distribution and speed up the convergence of a new distribution making the algorithm more accurate earlier in the product queries. It would be more costly, however, in the space required to hold all of the possible distributions.

Conclusion

In this work, we first presented an overview of how advice givers weigh reviewers and make predictions for users. We presented the Mediated (MAG) and Domain-Specific (DSAG) advice givers that weigh the tradeoffs between data sparsity and reviewers' weights differently and affect predic-

tion accuracy in recommender systems. Although the DSAG provides more accurate predictions when there are sufficient reviews, using a MAG was recently suggested when review data is too sparse. We used three real recommender data sets to show that both advice givers provide accurate predictions to some users. The DSAG provides more accurate predictions to users when different reviewers were weighed highly in different domains while the MAG is more accurate when the review matrix and user reviews are sparse.

We presented our User-Dependent Online Selection Algorithm and User- and Category-Dependent Online Selection algorithm as two online methods for recommender systems to decide which advice giver is more accurate for each user. The algorithms wait for the reviewers' weights to converge before picking the best advice giver. We validated our findings on a fourth recommender data set and demonstrated that the UdOS algorithm successfully picks the better advice giver to provide the most accurate predictions possible for each user. We conclude that a recommender system that uses the UdOS or UCdOS algorithms make better predictions for all users than one that uses only one of the two advice givers.

References

Adomavicius, G., and Tuzhilin, A. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowledge and Data Engineering* 734–749.

Auer, P.; Cesa-Bianchi, N.; Freund, Y.; and Schapire, R. 1995. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *FOCS*, 322–331.

Berkovsky, S.; Kuflik, T.; and Ricci, F. 2007a. Cross-domain mediation in collaborative filtering. In *User Modeling*, 355–359.

Berkovsky, S.; Kuflik, T.; and Ricci, F. 2007b. Distributed collaborative filtering with domain specialization. In *Recommender Systems*, 33–40.

Blum, A., and Mansour, Y. 2005. From external to internal regret. In *COLT*, 621– 636.

Burke, R. 2002. Hybrid recommender systems: Survey and experiements. *User Modeling and User-Adapted Interaction* 12(4):331–370.

Freund, Y.; Schapire, R.; Singer, Y.; and Warmuth, M. 1997. Using and combining predictors that specialize. In *STOC*, 334–343.

Leskovec, J.; Adamic, L.; and Huberman, B. 2006. The dynamics of viral marketing. In *ACM Conf. on Electronic Commerce*, 228–237.

Littlestone, N., and Warmuth, M. 1994. The weighted majority algorithm. *Information and Computation* 212–261.

Nakamura, A., and Abe, N. 1998. Collaborative filtering using weighted majority prediction algorithms. In *International Conference on Machine Learning*, 395–403.

Netflix. 2007. The netflix dataset and prize.

Umyarov, A., and Tuzhilin, A. 2007. Leveraging aggregate ratings for better recommendations. In *Recommender Systems*, 161–164.

Webscope, Y. R. 2008a. Movie user ratings of movies and descriptive content information v1.0.

Webscope, Y. R. 2008b. Music user ratings of songs with song attributes v1.0.