Bayesian Mixed-Effects Models for Recommender Systems

Michelle Keim Condliff
Mathematics & Computing Technology
The Boeing Company
michelle.k.condliff@boeing.com

David D. Lewis
AT&T Labs - Research
lewis@research.att.com

David Madigan*
AT&T Labs - Research
madigan@research.att.com

Christian Posse
Talaria, Inc.
posse@talariainc.com

Abstract

We propose a Bayesian methodology for recommender systems that incorporates user ratings, user features, and item features in a single unified framework. In principle our approach should address the cold-start issue and can address both scalability issues as well as sparse ratings. However, our early experiments have shown mixed results.

1 Introduction

Recommender systems have emerged as an important application area and have been the focus of considerable recent academic and commercial interest. The 1997 special issue of the *Communications of the ACM* [14] contains some key papers. Other important contributions include [2], [4], [8], [13], [16], [9], [1], [12], and [15]. In addition, many online retailers are using this technology to recommend new items to their customers, based on what they have bought in the past.

Currently, most recommender systems are either *content-based* or *collaborative*, depending on the type of information that the system uses to recommend items to a user.

Content-based filters try to recommend items that are similar to those that a given user has liked in the past. These approaches are usually based on techniques from information retrieval or machine learning. For example, text documents are recommended based on a comparison between their content and a user's profile. The profile is built up by analyzing the content of the items that the user has already rated. Thus, the system tries to recommend items that are similar to those that a user has liked in the past. These types of systems require both that a user has rated some items, and that there exists a way to derive explicit content-oriented features from the item.

*Address for Correspondence: AT&T Research - Labs, Room C282, Shannon Laboratories, 180 Park Avenue, Florham Park, NJ 07932-0971

Alternatively, collaborative filtering systems try to identify other users with tastes similar to the current user and recommend items that those users have liked. This approach appeals to the notion that when we are looking for information, we often seek the advice of friends with similar tastes or other people whose judgment we trust. We should be able to make use of what others have already found and evaluated. Again, this approach requires that a new user rate some items, but it typically does not require that items or users be represented by explicit features.

Pazzani [11] combines both content-based and collaborative filtering. However his approach is to tackle them separately and combine the results from each afterwards. Our goal is to use *all* the available information in a single probabilistic model.

2 Content-based versus Collaborative Filtering

Table 1 shows the data typically available to a recommender system. Here, the rows represent users, each with an associated vector of covariates (i.e. attributes of the users). The columns represent items (such as documents, movies, products), each with an associated feature vector. The elements of this matrix are binary, indicating whether or not each user liked each item. These elements could also take on a range of values which indicate the extent to which a user liked an item.

One can think of the difference between content-based and collaborative filters in terms of how they use the available data. Content-based filters use only the *content* or features of the items that the user has already rated and make no use of the ratings of past users. Therefore, they utilize only the last row of the data matrix (when it is available) and the feature vectors, ignoring the data in the first n rows of the matrix. Alternatively, collaborative filters make recommendations based solely on the ratings of

Table 1: *Collaborative filtering data*

User Covariates		Feature Vectors: (1 0 0 1 1 0 1)	(0 0 1 0 1 0 0)	(1 0 1 1 1 0 0)		(0 0 1 1 0 0 1)	
		I_1	I_2	I_3	$\cdots$	I_m	
(1 0 0 1 0 0)	U_1	0	1	1	$\cdots$	0	Past Users
(0 1 0 0 0 1)	U_2	1	1	0	$\cdots$	0	Past Users
(1 1 0 1 0 1)	U_3	0	1	1	$\cdots$	0	Past Users
	.						Past Users
	.						Past Users
	.						Past Users
(1 0 1 0 1 0)	U_n	0	0	1	$\cdots$	0	Past Users
(0 0 1 1 0 0)	U_{new}	?	1	0	$\cdots$	?	← New User

other users that are similar to a given user. Thus, the collaborative filtering approach uses only the data within the matrix, not the feature vectors or user covariate vectors.

Each approach has its strengths and weaknesses. One problem with a purely content-based approach is that in some applications it is difficult to extract features for the items that we wish to filter. A second problem is over-specialization: content-based systems may only be capable of recommending items that score highly against a user profile, therefore, the user will only get to see items that are similar to those he or she has already rated. Also, content-based filtering systems start with an empty profile for the user and can require the user to judge a large number of items in order to perform adequately.

Collaborative systems are capable of recommending items that are dissimilar to those that the user has already rated if other similar users have rated them highly. However, because they do not use use features of the items, if a new item appears in the database, the system cannot consider it until another user rates it. Also, a user with unusual tastes compared to the rest of the users will probably not get good recommendations.

The goal of our work is to construct a model for a recommender system that incorporates the components of both content-based and collaborative filtering models; in other words, a model that makes use of *all* of the available data. This can, in principle, provide an elegant solution to the "cold-start" problem, since initial recommendations use only the user covariates.

3 A Bayesian Model for a Recommender System

Consider a dataset with m "items", each item having a feature vector of length p, and n "users", each having a feature vector of length q. Our goal is to compute $P_i(L=1|\mathbf{f})$, the probability that the ith user will like the item represented by the feature vector $\mathbf{f}$, where $\mathbf{f} = (f_1, f_2, \ldots, f_p)$, and, for now, $f_k \in \{0,1\}, k = 1 \ldots p$. We do this for each item, and then recommend those with the highest probabilities. For example, if the items are documents, the item features might be words where the elements of $\mathbf{f}$ indicate the presence or absence of each word. Alternatively, if we are filtering movies, the item features might be "action" or "animated" or "horror."

The essential idea of a "mixed-effects" model is to use a two stage model. For recommender tasks we build a first stage model for each user, which predicts the preferences of that user from item features. The second stage model predicts the parameters of first stage models from user covariates. The two levels together provide a unified probabilistic model for the data in Table 1.

We proceed as follows. For now, we assume that for each user the item features are conditionally independent given that the user likes or dislikes an item. Then, using Bayes' Rule, we obtain a simple expression for the log odds that user i likes an item with feature vector $\mathbf{f}$:

$$\log \frac{P_i(L=1|\mathbf{f})}{P_i(L=0|\mathbf{f})} = \log \frac{P_i(L=1)}{P_i(L=0)} + \sum_{k=1}^{p} \log \frac{P_i(f_k=1|L=1)}{P_i(f_k=1|L=0)}. \tag{1}$$

Next we assume that for each item feature f_k, the collection of probabilities $\{P_i(f_k=1|L=1), i = 1 \ldots n\}$ comprises a sample from some distribution. Similarly, we assume that the collection $\{P_i(f_k=1|L=0), i = 1 \ldots n\}$, also comprises a sample from some different distribution. This is the key idea in Bayesian *hierarchical* modeling and allows the users to "borrow strength" from each other [7]. Specifically, we have:

$$\log\left(\frac{P_i(f_k=1|L=0)}{P_i(f_k=0|L=0)}\right)=\mu_{i_k}+\psi_{i_k} \qquad (2)$$

and

$$\log\left(\frac{P_i(f_k=1|L=1)}{P_i(f_k=0|L=1)}\right)=\mu_{i_k}+\psi_{i_k}+\delta_{i_k} \qquad (3)$$

where

$$\psi_{i_k}=\beta_{0_k}+\beta_{1_k}x_{i1}+\cdots+\beta_{q_k}x_{iq},$$

$\{x_{i1},\ldots,x_{iq}\}$ being the user features for user i.

μ_{i_k} represents a user-level random effect and δ_{i_k} represents the incremental effect of liking the item. We assume that δ_{i_k} is drawn from a normal distribution with mean d_{i_k} and variance $\frac{1}{\tau_{i_k}}$. In our applications we place diffuse prior distributions on μ_{i_k}, d_{i_k}, and τ_{i_k}:

$$\begin{aligned}\mu_{i_k} &\sim N(0,100000) &(4)\\ d_{i_k} &\sim N(0,100000) &(5)\\ \tau_{i_k} &\sim \Gamma(0.001,0.001). &(6)\end{aligned}$$

In turn these provide prior distributions for $P_i(f_k=1|L=1)$ and $P_i(f_k=1|L=0), i=1,\ldots,n, k=1,\ldots,p$. Together with a beta prior distribution for $P_i(L=1), i=1,\ldots,n$, these provide prior distributions for $P_i(L=1|\mathbf{f})$ for all $\mathbf{f}, i=1,\ldots,n$ leading directly to predictions. As data (i.e., user ratings) accumulate these prior distributions are updated via Bayes rule.

In essence this particular model fits a naive Bayes classifier to each user, and then links the parameters of these naive Bayes classifiers across the users through a regression model that incorporates the user-level covariates. The framework is, however, quite general. The naive Bayes classifier could be replaced by any parametric linear or non-linear model with either a binary or a continuous response scale; the second level regression model could be replaced by any (parametric or non-parametric) linear or non-linear regression model. The statistical literature refers to these types of models as "mixed effect models." [5] provides a thorough introduction to mixed-effect models. Similar models also arise in meta-analysis with study-level covariates (See, for example, [6]).

Exact estimation and prediction for Bayesian mixed-effect models require some form of iterative numerical method such as Markov chain Monte Carlo (MCMC). For the smaller of the two applications considered below, we implemented the model using BUGS, a general purpose MCMC engine (see `http://www.mrc-bsu.cam.ac.uk/bugs`). In general, this approach does not scale well, and for the larger application below (EachMovie) we have used a simple approximation. The idea is to fit a naive Bayes classifier for each user, and then to fit a regression model directly to the estimated parameters from these classifiers with the user covariates as predictors and the within-user sample sizes as weights. Chapter 5 of [5] points out some of the limitations of this approximation and in future work we will explore more complex approximations. However, the simple approximation appears to give results that are similar to the full MCMC approach, although our evaluations to date have been limited. Contact the authors for code for both approaches.

4 Empirical Evaluation

In this section we present the results of our experimental evaluation of our proposed methods and compare them to three "standards", including the correlation method [13]. We examine the performance on two datasets, a dataset on beverage preferences that we gathered at the University of Washington, and a subset of the EachMovie database [10].

4.1 Datasets

Beverages Data Our first data set is a small example in which we recommend beverages to the user. We gathered data from 210 "users" who rated how well they liked 16 different drinks on a scale of 1 to 7. The users were students in two undergraduate statistics courses at the University of Washington. We also collected data on each student, providing us with the following user covariates: age, gender, race, and where the person went to high school (with the idea that this would suggest where the person is from). The average age of the users was 21.5, and 62% were female. The group was predominantly Caucasian (62%) and Asian (30%). Most users went to high school in the state of Washington (78%). Unfortunately, we found that the user covariates that we collected were not very indicative of a user's beverage preferences; a regression model for each drink, with the ratings as the response and the user covariates as the predictors yielded a maximum R-squared value of 0.13.

We assigned ten features to each drink as shown in Table 2. The choice of these features was subjective: we chose features that we hoped would help to distinguish between the drinks while still incorporating their commonalities.

In order to apply the Bayesian model described above, we dichotomized the ratings. We decided that if the user gave the drink a rating greater than 4 then the user "likes" the drink, otherwise the user does not. For the purpose of fitting the model, we coded the user covariates as the following indicators:

Table 2: *Beverage Features*

	Carbonated	Sweet	Juice	Hot	Alcoholic	Caffeinated	Low-Calorie	Fruit-Derived	Highly Advertised	Citrus Drink
Coke	•	•				•			•	
Diet Coke	•	•				•	•		•	
Sprite	•	•							•	
Dr.Pepper	•	•				•				
Root Beer	•	•								
Sparkling Water	•						•			
Orange Juice		•	•					•		•
Grapefruit Juice			•					•		•
Apple Juice		•	•					•		
Tomato Juice			•							
Cranberry Juice			•					•		
Coffee				•		•	•			
Tea				•			•			
Hot Chocolate		•		•						
Beer					•					
Wine					•			•		

x_{i1} indicates if user i is female
x_{i2} indicates if user i is less than 21 years old
x_{i3} indicates if user i is 21-30 years old
x_{i4} indicates if user i is over 30 years old
x_{i5} indicates if user i is white
x_{i6} indicates if user i is asian
x_{i7} indicates if user i is not white or asian
x_{i8} indicates if user i is from the U.S.

EachMovie Data The EachMovie dataset came about from a research project at the Systems Research Center of Digital Equipment Corporation. The EachMovie service provided users with movie recommendations for an 18-month period in order to experiment with a collaborative filtering algorithm. The database grew to a fairly large size and is now publicly available for collaborative filtering research. The database contains 72,916 users who rated some of 1,628 movies on a six-point scale (0, 0.2, 0.4, 0.6, 0.8, 1). The features for the movies are: Action, Animation, Art/Foreign, Classic, Comedy, Drama, Family, Horror, Romance, or Thriller. The users sometimes provide the following user covariates: age, gender, and zip code.

For our experiments, we used the first 3,851 users which had values for all user covariates. We split the users ages into 11 age groups (0-14, 15-19, 20-24, 25-29, ... , 60-65) and we divided the zip codes into 7 US regions (Mid-Atlantic, Midwest, Northeast, Rockies, South, Southeast, and West). These users had an average of 49.75 (median = 30) votes each.

4.2 Experimental Results

Beverages Data To evaluate the Bayesian mixed-effects model, we considered three different Bayesian "strawman" methods to specify the prior distributions in Equation 1 for $P_i(f_k = 1|L = 1)$ and $P_i(f_k = 1|L = 0)$ for each new user:

1. Grand Mean: Use the mean observed probabilities across all past users.
2. Group Mean: Use the mean observed probabilities from past users with identical values on the three user covariates.
3. BHM: Use a Bayesian hierarchical model with no user covariates ($\psi_{ik} = 0$ for all i, k in Equations (2) and (3)).

We denote by BHMC the fully Bayesian mixed effects model with user covariates. Note that with methods 1 and 2, the prior distributions for $P_i(f_k = 1|L = 1)$ and $P_i(f_k = 1|L = 0)$ are the same for each user, but for methods 2 and BHMC, the distributions depend on the new user's covariate vector.

For each of the new users, we compute the probability that the user will like each drink. We evaluate each method by comparing these probabilities with the users' true ratings. Note that the user does not have to first rate *any* items in order to compute these probablilities.

We also examined three different non-Bayesian approaches. First, for every user, we predicted the mean score that past users gave the item. Second, we predicted the mean score that past users with the same user covariates gave the item (the "group" mean). Last, we used the correlation method ([13]) to predict scores for each item. The basic idea of the correlation method is to first compute the similarity between a new user and each of the past users in the database using:

$$w(u_{new}, u_i) = \frac{\sum_j (v_{new,j} - \bar{v}_{new})(v_{i,j} - \bar{v}_i)}{\sqrt{\sum_j (v_{new,j} - \bar{v}_{new})^2 \sum_j (v_{i,j} - \bar{v}_i)^2}}, \quad (7)$$

where $v_{i,j}$ is the vote for user i on item j, $\bar{v}_i$ is the mean vote for user i, and the summations are over only the items which both the new user and user i have rated. The prediction for item j for the new user is then a weighted sum of the past users' ratings for that item:

$$p_{new,j} = \bar{v}_{new} + \frac{\sum_i w(u_{new}, u_i)(v_{i,j} - \bar{v}_i)}{\sum_i w(u_{new}, u_i)} \quad (8)$$

where the summations are over those past users who have rated item j. Note that we cannot use this method until a user has rated at least two items. In the cases where we cannot compute the prediction using the correlation method, we predict the mean rating for a particular item. The correlation method has been widely used in the recommender systems literature.

We randomly selected 160 users for the training set and kept the remaining 50 "new users" for testing. We also examined the performance of the method after learning the ratings of the new users for some randomly selected items. For the beverages dataset, we looked at the results on the remaining unrated beverages when the users rated 2, 5, 10, and 15 drinks.

We compared the predictive performance of the models in two ways. First, we looked at classification accuracy. If the predicted probability for an item was greater than 0.5, we recommended that item to the user. Classification accuracy is the percentage of items correctly recommended or not recommended. Second, we looked at the mean Brier score [3] for each method. The Brier score is essentially the mean squared error for binary predictions, so lower scores indicate higher effectiveness.

In our case, we obtain a Brier score for each new user:

$$B_{new} = \frac{1}{m} \sum_{j=1}^{m} (v_{new,j} - p_{new,j})^2,$$

where $p_{new,j}$ is the new user's prediction for item j and $v_{new,j}$ is the new user's actual vote on that item. We then compute the average Brier score across all new users. The Brier score combines both accuracy and calibration in a single scoring rule.

Table 3 and Figure 1 show the results. The Bayesian methods perform well in general, outperforming the correlation method. However, the mixed-effect model does not appear to outperform simpler Bayesian alternatives in this application.

EachMovie Data Our evaluations with the EachMovie data are ongoing. Based on a five test-training splits of the data and using the approximate Bayesian mixed-effects model described above, the median cold-start acccuracy on the movies rated by 1,000 randomly selected test users is 56.5% (range 55.5%-57.1%) with a median Brier score of 0.247 (range 0.242-0.250).

To provide a baseline for the sequential learning performance of the Bayesian mixed-effects approach, we used a version of the standard correlation method but adapted for binary outcomes. Specifically, the prediction for item j for the new user is:

$$p_{new,j} = \bar{v}_{new} + \frac{\sum_i w'(u_{new}, u_i) v_{i,j}}{\sum_i w'(u_{new}, u_i)} \quad (9)$$

where the summations are over those past users who have rated item j and $w'(u_{new}, u_i)$ is the relative frequency with which user u_{new} and use u_i agree on those items that both have rated.

Sequentially learning the rated movies in a random order for each test user produces the results shown in Figure 2 (averaged over 30 random orderings).

To explain how this graph is constructed, consider the following. Suppose there are two test users. Suppose User 1 has rated 3 movies (A, B, and C) and User 2 has rated 4 movies (W, X, Y, and Z). First we randomly permute User 1's 3 rated movies and User 2's 4 rated movies. Then we predict all 7 movies just using the relevant ratings from the training data. This is the "cold-start" situation and only the Bayesian method is applicable. Next we "learn" User 1's rating for Movie A and user 2's rating for Movie W (i.e. update the priors in the Bayesian model) and predict the remaining 5 movies. If the prediction accuracy on these movies is 60%, this will appear as (1,0.6) in the figure. Then we learn Movie B for User 1 and Movie X for User 2 and predict the remaining 3 movies. If the prediction accuracy on these three movies is 66.67%, this will appear as (2,0.6667) in the figure. Then we learn Movie C for user 1 and Movie Y for User 2 and predict the one remaining movie for User 2. Thus, as we move from left to right in the graph, we observe the accuracy of predicted ratings based on an increasing number of observed ratings. However, since the mean number of votes for each user is about 50, a decreasing number of users are represented as the number of ratings increases.

It appears that the Bayesian approach ultimately outperforms the correlation method, but only after a large number of items have been rated by each user.

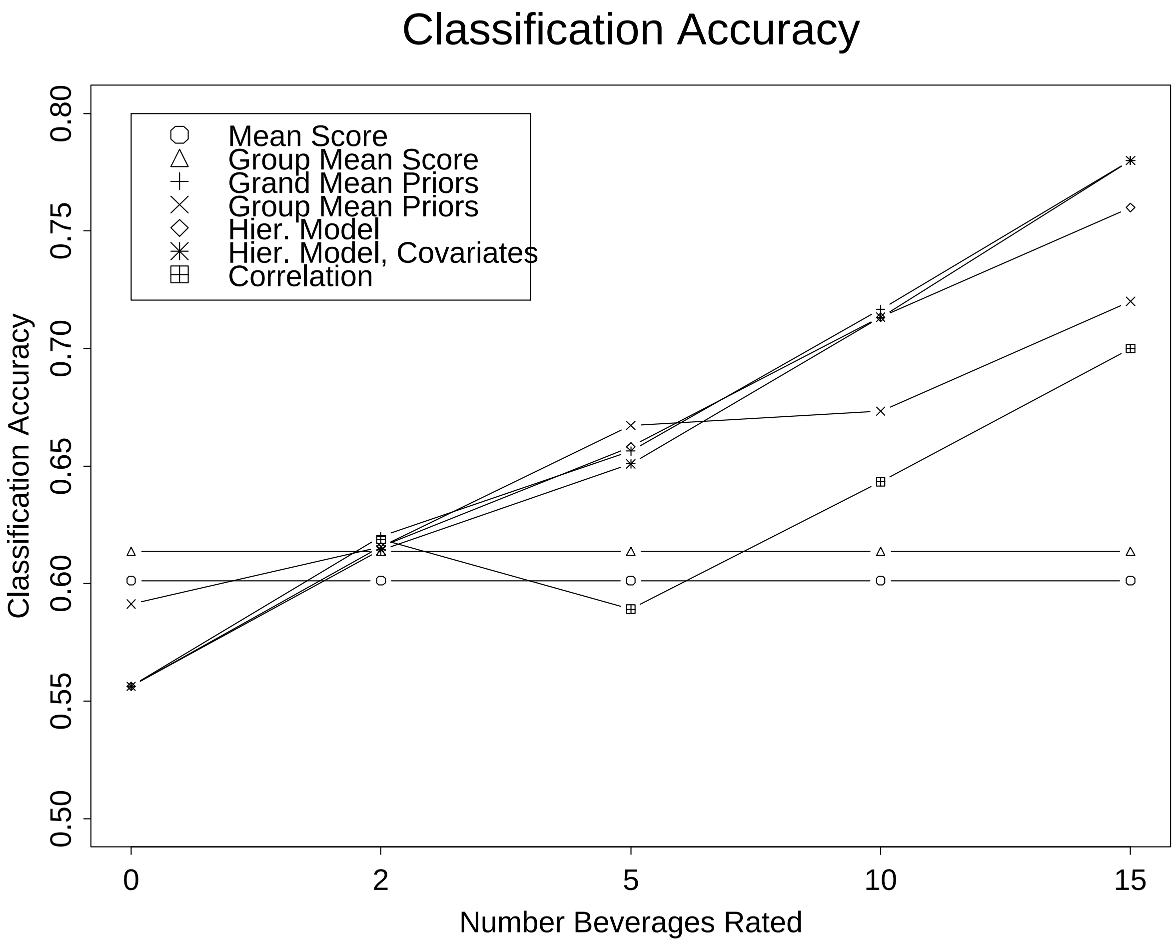

Figure 1: *Comparison of classification accuracy.*

		Number of Drinks User Has Rated				
Method		0	2	5	10	15
Past Users' Mean		0.2339				
Past Users' Group Mean		0.2386				
Correlation		NA	0.2404	0.2545	0.2281	0.2164
Bayesian Model						
Method of Specifying Prior Distributions	Grand Mean	0.2533	0.2337	0.2192	0.1851	0.1532
	Group Mean	0.2715	0.2414	0.2321	0.2102	0.1742
	Hierarchical Model – No Covariates	0.2530	0.2333	0.2188	0.1856	0.1598
	Hierarchical Model – User Covariates	0.2500	0.2340	0.2205	0.1872	0.1600

Table 3: *Comparison of Brier Scores.*

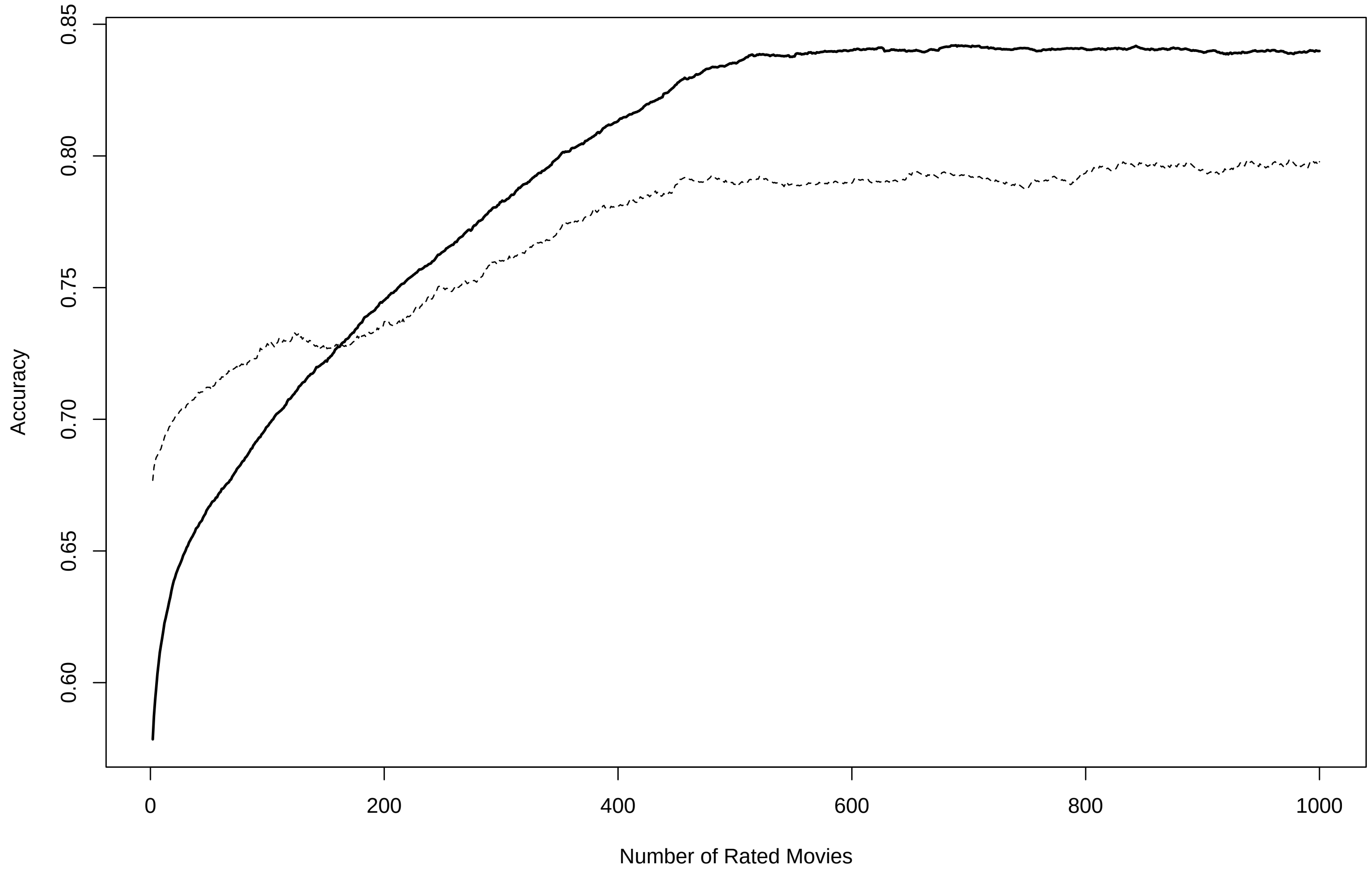

Figure 2: *Prediction accuracy for the EachMovie data. The solid line represents the Bayesian method and the dashed line represents the correlation method.*

5 Discussion

We have described a Bayesian mixed-effects model for recommender systems and presented some initial evaluations. The key advantage of the proposed approach is that it can use *all* the available information in a unified, coherent model. The practical question is whether or not incorporating the object and user feature information will result in more accurate predictions. The empirical evidence from our initial experiments with two applications suggests that the resultant benefits are modest.

Nonetheless, many outstanding issues remain. First, we have adopted a particularly simple Bayesian model in the experiments reported here. Second, for the EachMovie experiments, we implemented an approximate Bayesian approach. A more systematic study of the predictive properties of the approximation is required. Third, it may be that the features in the two applications we have studied are not especially informative

Acknowledgements We thank Martha Nason, W. Whipple Neely, and Ian Painter for helpful discussions.

References

[1] Marko Balabanovic and Yoav Shoham. Fab: Content-based, collaborative recommendation. *Communications of the ACM*, 40(3):64–72, 1997.

[2] J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. Technical report, Microsoft Research, 1998.

[3] G. Brier. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 75:1–3, 1950.

[4] Y-H. Chien and E.I. George. A bayesian model for collaborative filtering. In *Proceedings of the Seventh International Workshop on Artificial Intelligence and Statistics*, San Francisco, 1999. Morgan Kaufmann.

[5] Marie Davidian and David M. Giltinan. *Nonlinear models for repeated measurement data.* Chapman and Hall, 1995.

[6] W. DuMouchel. Hierarchical Bayes linear models for Meta-Analysis. Technical Report 27, National Institute of Statistical Sciences, Research Triangle Park, N.C., September 1994.

[7] Andrew Gelman, John B. Carlin, Hal S. Stern, and Donald B. Rubin. *Bayesian Data Analysis.* Chapman and Hall, 1995.

[8] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

[9] Ken Lang. Newsweeder: Learning to filter news. In *Proceedings of the Twelfth International Conference on Machine Learning*, 1997.

[10] Paul McJones. Eachmovie collaborative filetering dataset. dec systems research center. http://www.research.digital.com/src/eachmovie/.

[11] Michael Pazzani. *Artificial Intelligence Review*, chapter A Framework for Collaborative, Content-Based, and Demographic Filtering. In press.

[12] Michael Pazzani and Daniel Billsus. Learning and revising user profiles: The identification of interesting web sites. *Machine Learning*, 27:313–331, 1997.

[13] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. GroupLens: An open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 Computer Supported Cooperative Work Conference*, New York, 1994. ACM.

[14] Paul Resnick and Hal R. Varian. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997.

[15] Uprendra Shardanand and Pattie Maes. Social information filtering: Algorithms for automating "word of mouth". In *Proceedings of the Conference on Human Factors in Computing Systems (CHI'95)*, pages 210–217, Denver, CO, 1995. ACM.

[16] Loren Terveen, Will Hill, Brian Amento, David McDonald, and Josh Creter. PHOAKS: A system for sharing recommendations. *Communications of the ACM*, 40(3):59–62, 1997.