Learning Multiple Models for Exploiting Predictive Heterogeneity in Recommender Systems

Clinton Jones, Joydeep Ghosh, Aayush Sharma
Electrical and Computer Engineering
The University of Texas at Austin
Austin, TX 78712
{clinton_jones@, ghosh@ece., asharma@ideal.ece.}utexas.edu

ABSTRACT

Collaborative filtering approaches exploit information about historical affinities or ratings to predict unknown affinities between sets of "users" and "items" and make recommendations. However a model that also incorporates heterogeneous sources of information that may be available on the users and/or items can become a much more effective recommender, in terms of both increased relevance of the predictions as well as explainability of the results. In this paper, we propose a Bayesian approach that exploits not only such "side-information", but *also a different kind of heterogeneity* that captures the variations in the mapping from user/item attributes to the affinities of interest. Such *predictive heterogeneity* is likely to occur in large recommender systems that involve a diverse set of users, and can be mitigated by using multiple localized predictive models rather than a single global one that covers all user-item pairs. The scope or coverage of each local model is determined simultaneously with the model parameters. The proposed approach can incorporate different types of inputs to predict the preferences of diverse users and items. We compare it against well-known alternative approaches and analyze the results in terms of both accuracy and interpretability.

1. INTRODUCTION

The data involved in most recommender systems is often represented as a dyadic relation consisting of some utility function or *affinity* (e.g., ratings) between two sets of entities (e.g., users and movies). These relationships can be represented as the entries of a matrix, where the rows and columns are the entities from the two sets, and the matrix entries are the affinity values, if known. Collaborative filtering approaches to recommender systems have concentrated on exclusively using the few known ratings to predict the rest [23]. However they are known to suffer from problems such as cold-start, and also need to be augmented if they are to exploit additional "side-information" such as values of independent variables (*attributes*) that may be associated with the entities in the two sets, interaction networks that may exist within entities of a given type, etc. On the other hand, content-based recommender systems utilize attribute information, but often rely on a single global predictive model. Having only a single model for the entire dataset is limiting when the entities have distinct sub-populations that differ from one another in terms of the importance of different sources of information in determining the affinities that are to be estimated.

By viewing the data involved in recommender systems as a set of matrices (or tensors, if there is interaction among three or more sets of entities) with some shared axes, instead of a single denormalized file of data, it is easier to visualize the relationships between users, items, and their respective profiles or attributes, as well as the known ratings of the users for these items. It also facilitates an alternate view of the problem; instead of users rating items, it can just as easily be seen as items rating the users. A similar approach to viewing collaborative filtering from an item perspective is mentioned in [6]. There is no fundamental difference in the modeling that must take place, but it demonstrates that there can be an analogous separation of items into groups based on how they respond to the user attributes. Finding and grouping similar items in addition to similar users allows for more specific models to be trained on relatively homogeneous sub-groups of the set of all dyadic user-item pairs of interest. Additionally, though this appears as a content-based recommendation approach, since the grouping of users and items is directly influenced by the known affinities, it is possible to incorporate some of the advantages of collaborative filtering systems as well.

In this paper, we introduce a Bayesian approach to recommender systems that decomposes the dyadic data into soft or "mixed-membership" co-clusters while simultaneously learning local models for each co-cluster. We compare this model to other well-known models, and show that the results are comparable in accuracy, while showing much greater interpretability and actionability.

2. RELATED WORK

As described in [2], recommender systems seek to maximize for each user a utility function measuring the usefulness of items for that user. Much of the recent literature on collaborative filtering, including those that are based on matrix factorization (many of which were motivated by the Netflix Prize problem), ignore all sources of attribute information [13, 28, 23]. This is in part an unintended conse-

HetRec '11, October 27, 2011, Chicago, IL, USA
Copyright ©2011 ACM 978-1-4503-1027-7/11/10... $10.00

quence of the popularity of the Netflix competition, since the dataset involved had no additional information other than a timestamp associated with each rating. However when other datasets that contained extra information were encountered, approaches that do use such side information were put forth. These approaches often use such sources indirectly (e.g., as a regularizer to matrix factorization or through a kernel [1, 17, 5]); the known ratings are still the main driver behind the estimation of unknown affinities.

In contrast, content-based recommender systems often focus on a single aspect of external information, such as tagging information [6], [7], in predicting user-item affinities. Even approaches that incorporate multiple sources of information do so in one global model with a single similarity measure between the coupled user profile and item feature vectors. Even those collaborative-filtering approaches that use a kernel or regularizer based on this outside information do so in a global fashion. A single predictive model based on the user/item attributes does not exploit the predictive heterogeneity that may exist across different user and item groups. On the other hand, while the use of multiple predictive models to deal with such heterogeneity is encountered in a wide range of disciplines, from statistics to econometrics to control and marketing [19, 20, 15, 11, 12, 22, 18], these approaches typically apply only to denormalized data representations, rather than the multirelational data that is encountered in many recommender systems.

One approach to using heterogeneous sources of information is found in content-based recommender systems such as the one discussed in [7]. In that paper, and many other content-based approaches, the external user and item features are represented as coupled vectors representing user preferences and item characteristics, respectively. To find the affinity of a particular user-item pair, a similarity measure is calculated between the two corresponding vectors. For the approach proposed in that paper, the tags associated with each user and item were used to generate these vectors. Since both the users and items have explicit tags associated with them, the vectors for each are relatively sparse and can be stored and used efficiently.

Sometimes there exist item characteristics that are explicitly given (such as the director, producer, and distributor of a film), but no corresponding vector of user preferences is readily available. In this case, it is possible to derive the user preference vector by using the explicit item affinities as a user's implicit preferences about the item characteristics. Some methods of deriving a user profile from item attributes and user affinities are described in [21]. The machine learning algorithms described in that paper are suggested for learning the profiles based on item attributes that are well structured (i.e., a few attributes that are consistent across all items, with known possible values for the attributes). However, sometimes the number of attributes of an item may vary, such as the number of actors associated with a movie. Additionally, the possible values for those features may have extremely high cardinality (e.g., there are tens of thousands of possible actors that might be associated with a movie). Both these issues are problematic for this class of approaches.

The use of user and item vectors whose elements correspond exactly to one another is common in content-based recommendation systems [6]. This allows for simple calculation of the affinities, since the vectors can be directly used in an inner product. Another approach is to allow for additional information about users to be represented *without a corresponding characteristic in the item vector*, and vice-versa. Such features could be used to capture the overall bias for a user or item that could affect the values of the affinities, though it would not affect the ordering of items for a particular user.

It can be helpful to recognize that the similarity measure computed on a user's preferences and a corresponding item's characteristics is simply a number. With this realization, it is easy to see that using multiple similarity measure values (arising from heterogeneous profile and characteristic vectors) as features of an ensemble model would allow for the fusion of the various content-based affinity predictions in order to find a better overall recommendation. This ensemble recommendation would be based on heterogeneous aspects of the users and items. Also, by approaching the fusion of the predictors in this fashion, it would be perfectly reasonable to use different similarity measures of the various user profile and respective item characteristic vectors, which might be more appropriate for each of the types of profile and characteristic information. In fact, the function on the user-item vector pair does not need to be a true similarity measure at all, but could be any function useful to the overall model. Additionally, by also including a co-cluster bias term in this ensemble model [4], one can capture information similar to collaborative filtering approaches, where a group of similar users would all have a reasonably comparable affinity for a group of similar items.

There have been a few recent approaches proposed that can directly deal with both heterogeneous attributes as well as the predictive heterogeneity of users and items in predicting the affinities of the user-item dyads. Two of these approaches are SCOAL (Simultaneous Co-clustering and Learning) [8] and PDLF (Predictive Discrete Latent Factor Modeling) [4]. Both approaches partition the users and items into a grid of blocks (co-clusters) of related users and items, while simultaneously learning a predictive model on each co-cluster. The predictors directly use the attributes, as opposed to using them as a soft similarity constraint (as seen in [1, 17, 5]). The organic emergence of these predictive models is coupled with the formation of the co-clusters that define the domain of the models. Such coupling of the models and co-clusters improves both the interpretability and the accuracy when modeling predictively heterogeneous dyadic datasets, as this mechanism can effectively exploit both local neighborhood patterns as well as the globally available attributes [8, 4].

The Bayesian approach presented in this paper also exploits user and item features by grouping users and items into co-clusters and training separate models for each (user-group, item-group) pair. Thus different weights can be used for the features in each co-cluster model. However, unlike SCOAL, the groupings are soft rather than hard; each user/item belongs to multiple groups with different probabilities. Also, since an underlying probability model is assumed for the

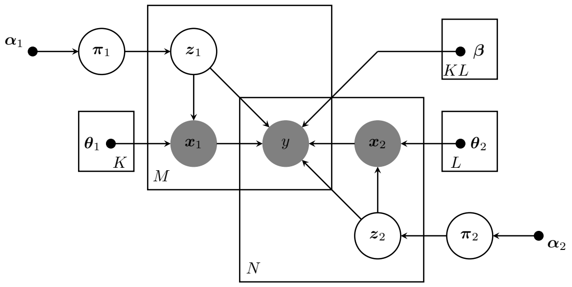

Figure 1: Graphical model for Latent Dirichlet Attribute Aware Bayesian Affinity Estimation.

observed data, the learning approach based on maximum likelihood is very different as well.

There are several Bayesian collaborative filtering approaches to recommender systems problems, such as Mixed Membership stochastic Blockmodels (MMBs) [10], which use the rating values to group the users and items via a soft co-clustering. Prediction of the unknown ratings is accomplished by using the weighted average of the co-cluster means associated with a particular dyad. This model has proved to be very scalable; however, it also ignores any available side information and typically addresses only 0/1 affinities such as the presence or absence of an unweighted link. Other Bayesian collaborative filtering approaches include frameworks for probabilistic matrix factorization with inference techniques such as Variational approximations [14] and sampling based MCMC methods [24]. Again, these approaches focus on utilizing only the ratings data. Recently, Bayesian models based on topic models for document clustering [9] have been applied to estimating affinities between users and news articles [3], allowing for content-based adaptations within a matrix factorization approach. Two-sided generalizations of topic models have also been proposed for co-clustering and matrix approximation problems, without taking into account auxiliary sources of information [25], [26].

3. LaD-BAE

This paper describes an Attribute Aware Bayesian Affinity Estimation approach to recommender systems that is related to Latent Dirichlet Allocation. Our approach is called the Latent Dirichlet Attribute Aware Bayesian Affinity Estimation (LaD-BAE) model. Figure 1 shows the graphical model for LaD-BAE; a mixture of KL clusters obtained as the cross-product of clustering the two sets of entities into K and L clusters, respectively. The approach is very similar to the LD-BAE model described in [27], but with the alteration that unobserved values are ignored (which makes the LaD-BAE algorithm more scalable). More detailed information on the update equations and their derivations can be found in that tech report. A summary of the generative process for the entity attributes and the dyad ratings is as follows:

1. Sample mixing coefficients: $\boldsymbol{\pi}_1 \sim \mathrm{Dir}(\boldsymbol{\alpha}_1)$
2. Sample mixing coefficients: $\boldsymbol{\pi}_2 \sim \mathrm{Dir}(\boldsymbol{\alpha}_2)$
3. For each entity $e_{1m} \in \mathcal{E}_1$
   (a) Sample cluster assignment: $\boldsymbol{z}_{1m} \sim \mathrm{Disc}(\boldsymbol{\pi}_1)$
   (b) Sample entity attributes: $\boldsymbol{x}_{1m} \sim p_{\psi_1}(\boldsymbol{x}_{1m}|\boldsymbol{\theta}_{1\boldsymbol{z}_{1m}})$
4. For each entity $e_{2n} \in \mathcal{E}_2$
   (a) Sample cluster assignment: $\boldsymbol{z}_{2n} \sim \mathrm{Disc}(\boldsymbol{\pi}_2)$
   (b) Sample entity attributes: $\boldsymbol{x}_{2n} \sim p_{\psi_2}(\boldsymbol{x}_{2n}|\boldsymbol{\theta}_{2\boldsymbol{z}_{2n}})$
5. For each dyad (e_{1m}, e_{2n}) such that $e_{1m} \in \mathcal{E}_1, e_{2n} \in \mathcal{E}_2$
   (a) Sample affinity (rating): $y_{mn} \sim p_{\psi_y}(y_{mn}|\beta^{\dagger}_{\boldsymbol{z}_{1m}\boldsymbol{z}_{2n}}\boldsymbol{x}_{mn})$

The overall joint distribution over all observable and latent variables is then given by:

$$p(\mathcal{Y}, \mathcal{X}_1, \mathcal{X}_2, \mathcal{Z}_1, \mathcal{Z}_2, \boldsymbol{\pi}_1, \boldsymbol{\pi}_2|\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \boldsymbol{\Theta}_1, \boldsymbol{\Theta}_2, \boldsymbol{\beta}) =$$
$$p(\boldsymbol{\pi}_1|\boldsymbol{\alpha}_1)p(\boldsymbol{\pi}_2|\boldsymbol{\alpha}_2)$$
$$\Big(\prod_m p(\boldsymbol{z}_{1m}|\pi_1)p_{\psi_1}(\boldsymbol{x}_{1m}|\boldsymbol{\theta}_{1\boldsymbol{z}_{1m}})\Big)$$
$$\Big(\prod_n p(\boldsymbol{z}_{2n}|\pi_2)p_{\psi_2}(\boldsymbol{x}_{2n}|\boldsymbol{\theta}_{2\boldsymbol{z}_{2n}})\Big)$$
$$\Big(\prod_{m,n} p_{\psi_y}(y_{mn}|\beta^{\dagger}_{\boldsymbol{z}_{1m}\boldsymbol{z}_{2n}}\boldsymbol{x}_{mn})p_{\psi_2}(\boldsymbol{x}_{2n}|\boldsymbol{\theta}_{2\boldsymbol{z}_{2n}})\Big)$$

The complexity of the distribution after marginalizing out the latent variables precludes the direct maximization of the observed (log) likelihood via an Expectation Maximization (EM) algorithm. Instead, we use a variational approach by constructing a lower bound on the log likelihood using a fully factorized mean field approximation to the true posterior distribution over the latent variables. The optimal factorized distribution over the latent variables $(\boldsymbol{Z}_1, \boldsymbol{Z}_2, \boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$ that corresponds to the tightest lower bound on the observed likelihood is given by:

$$q^*(\boldsymbol{Z}_1, \boldsymbol{Z}_2, \boldsymbol{\pi}_1, \boldsymbol{\pi}_2) =$$
$$q^*(\boldsymbol{\pi}_1|\gamma_1)q^*(\boldsymbol{\pi}_2|\gamma_2)\Big(\prod_m q^*(\boldsymbol{z}_{1m}|r_{1m})\Big)\Big(\prod_m q^*(\boldsymbol{z}_{2n}|r_{2n})\Big),$$

where $q(\boldsymbol{\pi}_1|\gamma_1)$ and $q(\boldsymbol{\pi}_2|\gamma_2)$ are K- and L-dimensional Dirichlet distributions with parameters γ_1 and γ_2, respectively. The cluster assignments $\boldsymbol{z}_{1m}$ and $\boldsymbol{z}_{2n}$ follow discrete distributions over K and L clusters with parameters r_{1m} and r_{2n}, respectively. Using this approximation, we are able to obtain updates for the variational and free model parameters, which are used in an EM algorithm, described in Algorithm 1.

We also incorporated pre-computed profile-attribute similarity features in the LaD-BAE model. These features are not considered in the clustering of the users or the items, so they can be used in the calculation of y (increasing the dimension of β accordingly) without further change to the updates.

In order to determine the best K and L for our model, we used a greedy search procedure described in Algorithm 2. Essentially, it consists of an inner loop and an outer loop. The outer loop updates the row and column clusters by independently increasing K and L by 1, and the inner loop uses Algorithm 1 to update the candidate models. The evaluation metric is then calculated for each candidate model on a validation set, and the best model becomes the new current model for the outer loop. If neither the row nor column cluster split improves upon the evaluation metric for the current model, the algorithm stops and returns the current best model and its corresponding values of K and L.

Algorithm 2 Choose K and L

```
Input: $\mathcal{Y}_{\text{obs}}, \mathcal{X}_1, \mathcal{X}_2, K_{init}, L_{init}$
Output: $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \boldsymbol{\Theta}_1, \boldsymbol{\Theta}_2, \boldsymbol{\beta}$
          $[m]_1^M, [n]_1^N, [k]_1^K, [l]_1^L$

    Step 0: Split $\mathcal{Y}_{\text{obs}}$ into $\mathcal{Y}_{\text{train}}$ and $\mathcal{Y}_{\text{validation}}$
    Step 1: Train LaD-BAE model using $\mathcal{Y}_{\text{train}}$ and other input parameters
    Step 2: Compute the evaluation metric $E_{\text{current}}$ of the learned model on the validation set, $\mathcal{Y}_{\text{validation}}$
    Until Convergence of $E_{\text{current}}$
        Step 3 Evaluate a row cluster split on the current model $M_{\text{current}}$
            Step 3a Compute the contribution $E_k$ of each row cluster to $E_{\text{current}}$
            Step 3b Select the worst cluster, $k_{worst}$, as the candidate to split
            Step 3c Create a new cluster, $k_{new}$, with $r_{1mk} = 0$ for all $m, k$
            Step 3d For rows with error below the median metric, assign $r_{1mk_{new}} = r_{1mk_{worst}}$ and $r_{1mk_{worst}} = 0$
            Step 3e Retrain LaD-BAE, seeding with the newly adjusted responsibilities
            Step 3f Compute the evaluation metric $E_{\text{row}}$ of the newly learned row-split model on the validation set
        Step 4 Evaluate a column cluster split on the current model $M_{\text{current}}$
            Step 4a Compute the contribution $E_l$ of each column cluster to $E_{\text{current}}$
            Step 4b Select the worst cluster $l_{worst}$ as the candidate to split
            Step 4c Create a new cluster $l_{new}$, with $r_{2nl} = 0$ for all $n, l$
            Step 4d For columns with error below the median metric, assign $r_{2nl_{new}} = r_{2nl_{worst}}$ and $r_{2nl_{worst}} = 0$
            Step 4e Retrain LaD-BAE, seeding with the newly adjusted responsibilities
            Step 4f Compute the evaluation metric $E_{\text{column}}$ of the newly learned column-split model on the validation set
        Step 5 Accept the best split model (and update $M_{\text{current}}$ and $E_{\text{current}}$) if its corresponding $E$ is better than $E_{\text{current}}$
```

Algorithm 1 Learn LaD-BAE

```
Input: $\mathcal{Y}_{\text{obs}}, \mathcal{X}_1, \mathcal{X}_2, K, L$
Output: $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \boldsymbol{\Theta}_1, \boldsymbol{\Theta}_2, \boldsymbol{\beta}$
          $[m]_1^M, [n]_1^N, [k]_1^K, [l]_1^L$

    Step 0: Initialize $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \boldsymbol{\Theta}_1, \boldsymbol{\Theta}_2, \boldsymbol{\beta}, r_{1mk}, r_{2nl}$
    Until Convergence
        Step 1: E-Step
            Until Convergence
                Step 1a: Update $\gamma_{1k}$
                Step 1b: Update $\gamma_{2l}$
                Step 1c: Update $r_{1mk}$
                Step 1d: Update $r_{2nl}$
        Step 2: M-Step
            Step 2a: Update $\boldsymbol{\theta}_{1k}$
            Step 2b: Update $\boldsymbol{\theta}_{2l}$
            Step 2c: Update $\boldsymbol{\beta}_{kl}$
            Step 2d: Update $\boldsymbol{\alpha}_1$
            Step 2e: Update $\boldsymbol{\alpha}_2$
```

4. EXPERIMENTS

As a comparison for our model, we compared the test RMSE to four other models: (1) using the global average, (2) a model comprised of the global average adjusted by local user and item bias terms, (3) a linear regression model on the entire data matrix, and (4) PMF (as implemented in GraphLab [16]). All of these models are deterministic, and consistently produce the same answer (for the same parameter values). However, the LaD-BAE model is susceptible to seeding conditions, and therefore an average over 5 runs is given.

4.1 Yahoo! Movies

We first give results on the Yahoo! Movies dataset. This dataset (obtained at http://webscope.sandbox.yahoo.com/) contains a training set of 211,231 movie ratings for 7,642 users and 11,915 movies. This training data is very sparse; the known entries are only .23% of the entire matrix. A test set is also provided, containing 10,136 ratings from 2,309 users and 2,380 movies

In addition to the ratings matrix, this dataset has user attributes (age and gender) and movie attributes (average movie rating value in the training set, number of ratings in the training set, MPAA rating, genre, and GNPP — a feature derived from awards nominated and won, which tries to capture the popularity of a movie).

4.2 Quantitative Analysis

The LaD-BAE model selection finds a large number of user clusters (8) for this data, and few movie clusters (2), despite the fact that there are more movie features, and in fact only two user features. This seems to indicate that for this dataset, the heterogeneity of the user attributes (and in fact movie attributes) is less important than the predictive heterogeneity in the response of the users to the movie attributes when separating the data. The results for the various algorithms on the dataset are shown in Table 1.

A surprising result for this dataset is how well the bias model performs when compared to the other models. Additionally, all of the feature-based models had worse errors than those for the models that depend wholly on the ratings values. Clearly, the features used in this dataset were not exceptionally helpful in obtaining good predictions at a global level, though the LaD-BAE model was able to achieve an RMSE similar to the best models in its best case (the best LaD-BAE model had an RMSE value of 0.9897), and even on average the LaD-BAE model is only slightly behind the best models. Thus LaD-BAE is able to recover from features that are not predictive at a global level to some degree by finding more local models that fit the data more accurately.

4.3 Qualitative Analysis

Since there were so many co-clusters found by the model selection algorithm, and as some of the features were represented in a 1-of-k format, we present only a few of the co-clusters and βs found by the feature-based algorithms. As our representative co-clusters we chose row clusters 4, 5, and 8 with both column clusters, i.e. provide the coefficients

Model	RMSE
Global Average	1.471670
Bias Model	0.986491
Global Linear	1.097887
PMF (lambda 0.225, D=50)	0.9867
LaD-BAE (K=8,L=2)	1.00027

Table 1: Average RMSE Results on the Yahoo! Movies Dataset.

for six co-clusters. For our features, we chose the co-cluster bias, user age, movie average rating, movie GNPP, movie number of ratings, and the indicator that the movie was rated PG. The corresponding βs are displayed in Table 2.

Clearly the most important features in both models are the co-cluster bias and the GNPP value. The LaD-BAE model captures some slight variations in the local co-clusters, which enable it to perform much better than the global linear model. Co-cluster (4,2) and (8,2) seem to slightly favor movies with a PG rating, while that rating has no influence on the other co-clusters. On the other hand, co-cluster (5,1) tends to rate commonly rated movies lower, which might indicate that those users have some intolerance for popularly rated movies. Another interesting trend is that movie cluster 2 has higher weights on the GNPP than cluster 1, indicating that some movies are popular because of the awards which they earn, while others are popular with users despite the opinions of the critics. Also, user cluster 8 has a relatively strong trend for rating movie lower as the user age increases. This could be a group of users who grow increasingly critical of the movies they see as they get older.

4.4 HetRec 2011 MovieLens Dataset

Next, we demonstrate results for our algorithm on the MovieLens dataset provided in conjunction with this workshop, and compare the results of this model to a global model and a slightly more intelligent model that ignores the heterogeneous attributes. We divided the data into separate training, validation, and testing sets, based on the timestamps of the users' ratings. We held back the last 4 ratings of each user as the test set, the previous 4 ratings for the validation set, and everything else was used as the training set. This split left a minimum of 12 training examples for each user. Selection of model parameters K and L was performed using the validation set, and the final models were trained on a combination of the training and validation sets (providing a minimum of 16 training examples per user).

For the attributes on this dataset, we used the user "bias" (the difference between the user's mean score and the overall mean of the data) as the only user attribute. For item attributes, we used the average scores and percentages of "fresh" ratings for both critics and audience from Rotten Tomatoes, as well as the movie "bias". Additionally, we derived 5 dyad-level features representing the utility function for the user profiles and item features for directors, countries, locations, and genres. Using this data, model selection resulted in only 2 user clusters and 4 movie clusters.

4.5 Quantitative Analysis

Model	RMSE
Global Average	1.010671
Bias Model	0.882657
Global Linear	0.775841
PMF (lambda 0.07, D=20)	0.8367
LaD-BAE (K=2,L=4)	0.775587

Table 3: Average RMSE Results on the HetRec 2011 MovieLens Dataset.

The RMSE values for each of the models is given in Table 3. Notably, LaD-BAE does not beat the global linear model every time, which is likely due to the extreme sparsity of the data coupled with the relative unimportance of the independent user and item features. It is easier for the more adaptive model to overfit when the data is so sparse. When analyzing the model features, we examined the βs (feature weights) of the best LaD-BAE model.

The RMSE values obtained for this dataset were much better than those for Yahoo! Movies in every model. This is likely due to the much higher sparsity of the Yahoo! Movies dataset, as even the feature-ignorant models performed considerably worse. Also, the feature-based models were much more competitive on this dataset, indicating that the heterogeneous features from the HetRec 2011 MovieLens dataset are much more predictive of ratings than those for the Yahoo! Movies dataset. In particular, the use of the features based on utility functions for actors and directors were easily derived from this recent dataset, but were not successfully generated on the Yahoo! Movies dataset. Thus, the utility functions of the heterogeneous attribute vectors appear to be very useful features for predicting rating values.

Clearly, taking user and movie bias information into account yields a more informative model than one that simply takes into account the global average. The matrix factorization model is reasonably good given that it does not take advantage of any of the external sources of information. However, the global linear model performs substantially better than matrix factorization by taking this additional information into account. And the LaD-BAE model can perform even better than the global linear model by capturing predictive heterogeneity in user and item clusters.

4.6 Qualitative Analysis

The matrix factorization model yields latent factors that describe the users and items, but those latent factors are not easily interpreted. One of the useful characteristics of the LaD-BAE model is that it still allows for human-recognizable interpretation of its results. In fact, it is helpful to compare these results to the global linear model, to see how LaD-BAE is able to capture the predictive heterogeneity present in the data. The βs for both models are given in Table 4.

From the results, we can see that a few of the features are much more helpful in predicting the affinities than others. In particular, the biases are important for all the models, especially the co-cluster and movie biases. For the global model, the only feature that is relevant outside of these biases is the feature representing the relationship between a user's actor

Model	CC	Age	Avg Rat	GNPP	# Rat	PG
Global	4.0887	-0.0824	-0.0081	0.6153	-0.0013	0.0198
LaD-BAE (k,l)						
(4,1)	4.2344	0.0000	0.0000	0.6013	0.0000	0.0000
(4,2)	4.1813	0.0000	0.0000	0.7415	0.0000	0.1233
(5,1)	4.3272	0.1661	0.0000	0.4360	-0.1716	0.0000
(5,2)	4.0963	0.2198	-0.2720	0.6608	0.0000	0.0000
(8,1)	4.1066	-0.3514	0.0000	0.4893	0.0000	0.0000
(8,2)	4.0300	-0.3348	0.0000	0.5451	0.0000	0.1108

Table 2: β values for selected Yahoo! Movies co-clusters and features.

Model	CC	U	M	AR	AF	CR	CF	D	C	L	A	G
Global	3.4364	0.1452	0.3964	0.0146	-0.0313	-0.0336	0.0317	0.0814	0.0207	0.0087	0.2584	0.0466
LaD-BAE (k,l)												
(1,1)	3.4572	0.0025	0.3994	0.0129	0.0389	-0.0310	-0.0427	0.1080	0.1330	0.4167	0.1046	-0.0083
(1,2)	3.4449	0.0737	0.3632	0.0025	0.0457	-0.0280	-0.0474	0.0245	0.0031	0.7349	-0.0130	-0.0191
(1,3)	3.4038	0.1311	0.3859	0.0212	0.0556	-0.0278	-0.0586	0.0279	0.2879	0.1479	-0.0245	0.0441
(1,4)	3.4457	0.0810	0.3913	0.0477	0.0226	-0.0733	-0.0354	0.0154	0.2066	0.5353	0.0454	-0.0299
(2,1)	3.4895	0.1937	0.3888	0.0184	0.0118	-0.0317	-0.0093	0.0762	0.0862	0.0723	0.0763	0.0072
(2,2)	3.4943	0.1959	0.3603	0.0277	0.0270	-0.0433	-0.0227	0.0447	0.0183	0.1639	0.0324	-0.0028
(2,3)	3.4724	0.2676	0.3742	0.0262	0.0301	-0.0398	-0.0322	-0.0121	0.0955	0.0424	-0.0005	0.0373
(2,4)	3.5206	0.2143	0.3904	0.0158	-0.0073	-0.0269	-0.0001	0.0087	0.1876	0.1309	0.0716	-0.0167

Table 4: β values for linear models features. Column heading abbreviations are as follows: CC: co-cluster, U: user bias, M: movie bias, AR: Average Audience Rating, AF: Audience Freshness Score, CR: Average Critics Rating, CF: Critics Freshness Score, D: Director, C: Country, L: Location, A: Actors, G: Genre.

preferences and the actors appearing in the film. Thus, on average the most important factor in deciding whether a user will rate a movie highly appears to be how similar users have rated similar movies. A movie's popularity across all users is also helpful information, as well as whether the user likes the actors appearing in the film. Finally, users do appear to have a slight bias for or against giving high ratings in general, which can also be somewhat predictive.

For the split model, in user cluster 1 the user bias is less important, and the actor and director features are much more important. This indicates that the first user cluster captures the behavior of users who are much more swayed by their favorite actors and directors despite other qualities of the film, while the second cluster captures the behavior of users who are generally less influenced by who directed or played in a film. Additionally, the effect of actors on ratings for movie clusters 2 and 4 is more important in both user clusters. This could be explained by blockbuster movies that are very popular, and often include performances by many actors who are popular across all users.

Location, country, and genre do not seem to be a large influence in a user's rating of a movie. Additionally, the average ratings and freshness scores from Rotten Tomatoes do not appear to be very informative. In other words, this feature does not seem to help the model predict the ratings (as evidenced by the low magnitude of its weight in the model). This is likely due to the fact that these features might already be captured in the other more important features, especially the movie bias, which contains a very similar "average rating" that is more directly related to the rating being predicted.

4.7 Further Analysis of HetRec MovieLens

Recognizing that the low-scoring features were likely introducing noise into the system and creating additional undesirable local minima in the solution space for LaD-BAE, we removed all the features except for co-cluster bias, user bias, movie bias, actor, and director. With these five features only, we ran an additional set of experiments for the global model and LaD-BAE, and found that while the prediction error of the global model increased to 0.77661, the average prediction error of LaD-BAE decreased to 0.7735. Table 5 shows the βs for the models after feature selection, which demonstrate many of the same patterns as the corresponding features in the larger model. The remaining features carried even more weight, indicating that their influence was more apparent when not masked by the less informative (but still somewhat correlated) features of the larger model.

5. CONCLUSIONS AND FUTURE WORK

We have shown that while matrix factorization and similar algorithms that ignore side-information can be very effective for recommender systems, models that use heterogeneous sources of data for predicting the affinities can achieve even greater accuracies. This is especially true when the features used are informative. With the availability of many heterogeneous sources of data, it is important to exploit the available data and work with models that can incorporate

Model	CC	U	M	A	D
Global	3.4364	0.2035	0.3890	0.2629	0.0827
LaD-BAE (k,l)					
(1,1)	3.3918	0.1877	0.4537	0.1133	0.0336
(1,2)	3.5034	0.2515	0.4482	0.1956	0.1846
(1,3)	3.4929	0.1766	0.4305	0.4519	0.1217
(1,4)	3.4880	0.2499	0.4531	0.3106	0.0195
(2,1)	3.3822	0.1052	0.3537	0.3582	0.0869
(2,2)	3.5916	0.1325	0.3023	0.5113	0.4935
(2,3)	3.5083	0.0656	0.3075	0.6260	0.3446
(2,4)	3.4824	0.1805	0.3376	0.4106	0.1567

Table 5: β values for models after pruning.

the additional features effectively. The focus of this paper was on learning multiple local models in the process of incorporating such features.

Specifically, we proposed a Bayesian approach called LaD-BAE to capture the predictive heterogeneity in the interaction among different groups of users and items. LaD-BAE can take advantage of both feature heterogeneity as well as predictive heterogeneity to obtain good predictions for affinities between users and items. Additionally, the models obtained by LaD-BAE are much more interpretable than matrix factorization or even a global linear model. They are also more actionable, since one can separately understand and model/target different sub-populations rather than use a one-size-fits-all approach underlying a global model. One additional advantage of the LaD-BAE model (as well as similar approaches) is its ability to combine multiple content-based predictions, while also including a co-cluster bias term that allows for collaborative filtering information to be used in the same composite model.

We achieved our best results using only a few informative features; finding and utilizing additional informative features, and particularly ones that are not highly correlated with the current features, we could realize even greater improvements in the accuracy. One such feature that we would like to explore, but did not have the time to complete, is the use of tag information, such as is described in [7] and [6]. Features extracted from any social networks among the users are also worthy of further investigation.

6. ACKNOWLEDGMENTS

This work was supported in part by NSF-IIS 1016614.

7. REFERENCES

[1] J. Abernethy, F. Bach, T. Evgeniou, and J.-P. Vert. A new approach to collaborative filtering: Operator estimation with spectral regularization. In *The Journal of Machine Learning Research*, volume 10, pages 803–826, June 2009.

[2] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17:734–749, 2005.

[3] D. Agarwal and B. Chen. flda: matrix factorization through latent dirichlet allocation. In *Proc. ACM international conference on Web search and data mining, 2010*, pages 91–100, 2010.

[4] D. Agarwal and S. Merugu. Predictive discrete latent factor models for large scale dyadic data. In *KDD '07*, pages 26–35, 2007.

[5] J. Basilico and T. Hofmann. Unifying collaborative and content-based filtering. In *ICML*, 2004.

[6] A. Bellogín, I. Cantador, and P. Castells. A study of heterogeneity in recommendations for a social music service. In *Proceedings of the 1st International Workshop on Information Heterogeneity and Fusion in Recommender Systems*, HetRec '10, pages 1–8, New York, NY, USA, 2010. ACM.

[7] I. Cantador, A. Bellogín, and D. Vallet. Content-based recommendation in social tagging systems. In *Proceedings of the fourth ACM conference on Recommender systems*, RecSys '10, pages 237–240, New York, NY, USA, 2010. ACM.

[8] M. Deodhar and J. Ghosh. Scoal: A framework for simultaneous co-clustering and learning from complex data. *ACM Trans. Knowl. Discov. Data*, 4:11:1–11:31, October 2010.

[9] A. Y. N. D.M. Blei and M. I. Jordan. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.

[10] S. E. F. E. Airoldi, D. M. Blei and E. Xing. Mixed membership stochastic blockmodels. *JMLR*, 9:1981–2014, 2008.

[11] A. Gelman and J. Hill. *Data Analysis Using Regression and Multilevel/Hierarchical Models.* Cambridge University Press, 2007.

[12] R. Grover and V. Srinivasan. A simultaneous approach to market segmentation and market structuring. *Journal of Marketing Research*, pages 139 – 153, 1987.

[13] J. Herlocker, J. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *SIGIR '99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages ACM 1999 230–237, Berkeley, CA, USA, August 15-19 1999.

[14] Y. Lim and Y. Teh. Variational bayesian approach to movie rating prediction. In *Proc. KDD Cup and Workshop*, 2007.

[15] L. Lokmic and K. A. Smith. Cash flow forecasting using supervised and unsupervised neural networks. *IJCNN*, 06:6343, 2000.

[16] Y. Low, J. Gonzalez, A. Kyrola, D. Bickson, C. Guestrin, and J. M. Hellerstein. Graphlab: A new framework for parallel machine learning. *CoRR*, abs/1006.4990, 2010.

[17] Z. Lu, D. Agarwal, and I. Dhillon. A spatio-temporal approach to collaborative filtering. In *RecSys'09*, 2009.

[18] W. Moe and P. Fader. Modeling hedonic portfolio products: A joint segmentation analysis of music compact disc sales. *Journal of Marketing Research*, pages 376 – 385, 2001.

[19] R. Murray-Smith and T. A. Johansen. *Multiple Model Approaches to Modelling and Control.* Taylor and Francis, UK, 1997.

[20] K. Oh and I. Han. An intelligent clustering forecasting system based on change-point detection and artificial

neural networks: Application to financial economics. In *HICSS-34*, volume 3, page 3011, 2001.

[21] M. Pazzani and D. Billsus. Content-based recommendation systems. In P. Brusilovsky, A. Kobsa, and W. Nejdl, editors, *The Adaptive Web*, volume 4321 of *Lecture Notes in Computer Science*, pages 325–341. Springer Berlin / Heidelberg, 2007. 10.1007/978-3-540-72079-9_10.

[22] T. Reutterer. Competitive market structure and segmentation analysis with self-organizing feature maps. *Proceedings of the* 27^{th} *EMAC Conference*, pages 85 – 115, 1998.

[23] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *NIPS '07*, 2007.

[24] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proc. ICML, 2008*, pages 880–887, 2008.

[25] H. Shan and A. Banerjee. Bayesian co-clustering. In *ICDM*, pages 530–539, 2008.

[26] H. Shan and A. Banerjee. Residual bayesian co-clustering and matrix approximation. In *Proc. SDM 2010*, pages 223–234, 2010.

[27] A. Sharma and J. Ghosh. Side information aware bayesian affinity estimation. *Technical Report TR-11, Department of ECE, UT Austin*, 2010.

[28] G. TakÅÂůcs, I. PilÅÂůszy, B. NÅÈmeth, and D. Tikk. Investigation of various matrix factorization methods for large recommender systems. In *2nd KDD-Netflix workshop*, 2008.