Using Probabilistic Relational Models for Collaborative Filtering

Lise Getoor
Stanford University
getoor@cs.stanford.edu

Mehran Sahami
Epiphany, Inc.
sahami@epiphany.com

Abstract

Recent projects in *collaborative filtering* and *information filtering* address the task of inferring user preference relationships for products or information. The data on which these inferences are based typically consists of pairs of people and items. The items may be information sources (such as web pages or newspaper articles) or products (such as books, software, movies or CDs). We are interested in making recommendations or predictions. Traditional approaches to the problem derive from classical algorithms in statistical pattern recognition and machine learning. The majority of these approaches assume a "flat" data representation for each object, and focus on a single dyadic relationship between the objects. In this paper, we examine a richer model that allows us to reason about many different relations at the same time. We build on the recent work on *probabilistic relational models (PRMs)*, and describe how PRMs can be applied to the task of collaborative filtering. PRMs allow us to represent uncertainty about the existence of relationships in the model and allow the properties of an object to depend probabilistically both on other properties of that object and on properties of *related* objects.

1 Introduction

In the past few years, we have seen an enormous growth in the availability of on-line information, especially relating to individuals' activities on the World Wide Web. Recently, much work has been done under the rubric of *collaborative filtering* to make use of this information for inferring user preference relationships about various products, movies, or other information (which we generically refer to as *items*). In this setting, the data is a collection of pairs of objects, where each pair consists of a person and an item. The items may be information sources such as web pages or newspaper articles (these often come along with preference rankings), or they may be products such as books, software or CDs. In the collaborative filtering task, we are interested in making predictions of how likely a person is to be interested in a particular item, given information about this user and other users' historical interests or purchasing behavior. While this pair-wise relationship is the focus of our prediction, we may have additional information about the objects and their relationships to other objects in the domain.

For example, we may be interested in predicting whether a customer will purchase a particular item based on information that we infer about the customer and information we have about other customers' buying patterns. The relation of interest here is the *Buy* relation. This inference may be based on a customer's relationships to other customers (e.g., members of the same demographic groups, members of the same organizations, people who subscribe to similar periodicals, etc.), web pages they have visited, and/or other information, such as a customer's current financial statement. This inference may also be based on information we infer about the customer based on other purchases that they have made, and relationships between the purchases (e.g., if products are complementary, if the customer has a taste for luxury items, if the customer has bought many products from the same manufacturer).

Such problems can be viewed as reasoning about the relationship between two objects. These relations may be logical, i.e., the relationship either exists or does not, or the relations may be qualitative and have some weight or strength associated with them indicating user preference. In the following discussion we address only the former case, where the relation either exists or not. It is feasible to extend all of the approaches we discuss to include a quantitative (rather than on/off) measure of the relation.

Collaborative filtering is based on the assumption that by finding similar users and examining their usage or preference patterns, we can make useful recommendations. Several methods have been suggested [5; 9]. Here we focus on model-based methods. We review a recently proposed approach, the two-sided clustering model, for collaborative filtering [10; 4]. We describe how this model can be represented by a Bayesian network (BN). Next, we describe how this model can be represented as a probabilistic relational model (PRM), and show how the PRM subsumes and extends the BN model.

PRMs are more refined probabilistic models that represent statistical correlations both between the properties of an entity and between the properties of related entities. Such models can be used for reasoning about an entity using

the entire rich structure of knowledge encoded by the relational representation [8]. We examine the use of PRMs for collaborative filtering and show how these models are well-suited for this task.

2 Foundations

In this paper, we focus on model-based approaches to collaborative filtering. Before we describe this particular application, we review the formal foundations of the models we are considering. The two particular formalisms examined presently are Bayesian networks (BNs) and probabilistic relational models (PRMs).

2.1 Bayesian Networks

Suppose we have a finite set $\mathcal{A} = \{A_1, \ldots, A_n\}$ of discrete random variables where each variable A_i may take on values from a finite set. We use capital letters, such as B, C, D, for variable names and lowercase letters b, c, d to denote specific values taken by those variables. Sets of variables, in other words subsets of $\mathcal{A}$, are denoted by boldface capital letters $\mathbf{B}, \mathbf{C}, \mathbf{D}$, and assignments of values to the variables in these sets are denoted by boldface lowercase letters $\mathbf{b}, \mathbf{c}, \mathbf{d}$. Finally, let P be a joint probability distribution over the variables in $\mathcal{A}$, and let $\mathbf{B}, \mathbf{C}, \mathbf{D}$ be subsets of $\mathcal{A}$. The sets $\mathbf{B}$ and $\mathbf{C}$ are *conditionally independent* given $\mathbf{D}$ if for all $\mathbf{b} \in \text{Val}(\mathbf{B}), \mathbf{c} \in \text{Val}(\mathbf{C}), \mathbf{d} \in \text{Val}(\mathbf{D})$, $P(\mathbf{b} \mid \mathbf{d}, \mathbf{c}) = P(\mathbf{b} \mid \mathbf{d})$ whenever $P(\mathbf{c}, \mathbf{d}) > 0$.

A *Bayesian network* is an annotated directed acyclic graph that encodes a joint probability distribution over $\mathcal{A}$. Formally, a Bayesian network for $\mathcal{A}$ is a pair $BN = \langle G, \Theta \rangle$. The first component, namely G, is a directed acyclic graph whose vertices correspond to the random variables $A_1, \ldots, A_n$. The graph encodes the following set of conditional independence assumptions: each variable A_i is independent of its non-descendants given its parents in G. The second component of the pair, Θ, represents the set of parameters that quantifies the network. It contains a parameter $\theta_{a_i|\text{pa}(i)} = P(a_i|\text{pa}(i))$ for each possible value a_i of A_i, and $\text{pa}(i)$ of $\text{Pa}(i)$. Here $\text{Pa}(i)$ denotes the set of parents of A_i in G and $\text{pa}(i)$ is a particular instantiation of the parents. A Bayesian network BN specifies a unique joint probability distribution over $\mathcal{A}$ given by: $P(A_1, \ldots, A_n) = \prod_{i=1}^{n} P(A_i|\text{Pa}(i))$

The problem of learning a Bayesian network can be stated as follows. Consider a set of variables $\mathcal{A}$, as defined above. Given a *training set* $\Xi = \{\mathbf{a}^1, \ldots, \mathbf{a}^N\}$ of instances of $\mathcal{A}$, find a network BN that is a good model for Ξ. The common approach to this problem is to introduce a scoring function that evaluates each network with respect to the training data, and then to search for the best network according to this metric. The two scoring functions most commonly used to learn networks are *Bayesian scoring* metrics [3], and one based on the principle of *minimal description length* (MDL) [7].

2.2 Probabilistic Relational Models

The other approach we will consider is using a probabilistic relational model. We describe our PRM in generic terms, closely related to the language of entity-relationship models. Here we give a brief review of the model. For a full description of the model and details on learning these models from data, see [8].

We begin by defining the notation we will use for PRMs. A schema for a relational model consists of a set of *entity types* $\mathcal{E} = X_1, \ldots, X_n$ and a set of *relations* $\mathcal{R} = \{R_1, \ldots, R_m\}$. Each relation R is typed. We use the notation $R(X_1, \ldots, X_k)$ to indicate the entity type of each component in the relationship. Each entity type has a set of *attributes*. We denote the set of attributes for the entity type X as $\mathcal{A}(X)$. Furthermore, we can denote an individual attribute in an entity type using *dot* notation. For example, the attribute A in entity type X is denoted $X.A$.

A complete *instantiation* $\mathcal{I}$ of a schema defines a set of objects $\mathcal{O}^{\mathcal{I}}(X)$ for each entity type X. For each entity $x \in \mathcal{O}^{\mathcal{I}}(X)$, and each attribute $A_j \in \mathcal{A}(X)$, the entity has an associated attribute $x.a_j$; its value in $\mathcal{I}$ is denoted $\mathcal{I}_{x.a_j}$. For each tuple of entity types $\mathcal{E}' = \{X_1, \ldots, X_k\}$, we denote a tuple of entities from this set of entity types as $e' = \{x_1, \ldots, x_k\}$. For each relation $R(\mathcal{E}')$ and each $e' \in \mathcal{O}^{\mathcal{I}}(X_1) \times \cdots \times \mathcal{O}^{\mathcal{I}}(X_k)$, $\mathcal{I}$ specifies whether $R(e')$ holds. For brevity, in the remainder of this paper, we use R in place of $R(e')$, and r in place of $r(e')$, when they may be used unambiguously.

We are interested in describing a probability model over instantiations of a relational schema. However, some attributes, such as a name or social security number, are fully determined given the entity. We label such attributes as *fixed*. We assume that they are known in any instantiation of the schema. The other attributes are called *probabilistic*. A *domain* Ω for a relational schema is a partial specification of an instance of the schema. It specifies the set of objects $\mathcal{O}^{\Omega}(X_i)$ for each entity type X_i and the values of the fixed attributes of these entities. A *skeleton structure* σ of a relational schema includes both a domain and the set of relations that hold between the entities. However, it leaves the values of probabilistic attributes unspecified. A complete instantiation $\mathcal{I}$ of a domain Ω specifies the relations that hold between entities and also specifies the values of the probabilistic attributes.

The domain defines the set of objects in our model, and we are now interested in modeling the probabilistic dependencies between the entities. The basic goal here is to model our uncertainty over the relations that hold in our model and the values of the probabilistic attributes of the objects in our domain. In other words, given a domain, we would like to define a probability distribution over completions of the domain.

Our probabilistic model consists of two components: the qualitative dependency structure, $\mathcal{S}$, and the parameters associated with it, $\theta_{\mathcal{S}}$. The dependency structure is defined for both the relations and the probabilistic attributes. The dependency structure for the relations is defined by associating each relation R with a set of parents, $\text{Pa}(R)$. Similarly, each probabilistic attribute $X.A$ is associated with a set of parents, $\text{Pa}(X.A)$. These correspond to *formal* parents; they will be instantiated in different ways for different objects in X. Intuitively, the parents are attributes that are "direct influences" on R and $X.A$.

Whether the relation $R(\mathcal{E}')$ holds can depend on probabilistic attributes of the entity types in $\mathcal{E}'$ (e.g., $X.A$ for $X \in \mathcal{E}'$). For probabilistic attributes, we distinguish between two types of formal parents. The attribute $X.A$ can depend on another probabilistic attribute B of X. This formal dependence induces a corresponding dependency for individual objects: for any object x in $\mathcal{O}^{\Omega}(X)$, $x.a$ will depend probabilistically on $x.b$. The attribute $X.A$ can also depend on attributes of related objects $X.\tau.B$, where τ is some chain of relations. In some cases the relation is not one-to-one, i.e., $X.\tau.B$ is a set of objects. In this case, we use the notion of *aggregation* from database theory to address this issue; i.e., $x.a$ will depend probabilistically on some aggregate property of this set. There are many natural and useful notions of aggregation: the mode of the set (most frequently occurring value); mean value of the set (if values are numerical); median, maximum, or minimum (if values are ordered); cardinality of the set; etc.

Given a set of parents for a relation or an attribute, we can define a local probability model by associating with it a *conditional probability distribution (CPD)*. For each relation, we have CPD $P(R \mid \text{Pa}(R))$ and for each probabilistic attribute we have CPD $P(X.A \mid \text{Pa}(X.A))$. The set of all such CPDs comprise $\theta_{\mathcal{S}}$.

Given a domain for our schema, we now want to use these local probability models to define a probability distribution over completions of the domain. Note that the domain determines the set of objects in our model. Recall that, we associate a random variable $x.a$ with each probabilistic attribute A of each object x. The domain also determines the set of potential relations in our model. We model this by having a random variable r for each each $\langle x_1, \ldots, x_k \rangle \in \mathcal{O}^{\Omega}(X_1) \times \cdots \times \mathcal{O}^{\Omega}(X_k)$. r specifies whether $R(x_1, \ldots, x_k)$ holds.

To define a coherent probabilistic model over our domain, we must ensure that our probabilistic dependencies are acyclic, so that a random variable does not depend, directly or indirectly, on its own value.

Consider the parents of a relation R. When $X.a$ is a parent of R, we define an edge $x.a \rightarrow_{\sigma} r$.

Similarly, consider the parents of an attribute $X.A$. When $X.B$ is a parent of $X.A$, we define an edge $x.b \rightarrow_{\sigma} x.a$. When $X.\tau.B$ is a parent of $X.A$ via a chain of relations τ and y is one of the objects related to x via τ, we define an edge $y.b \rightarrow_{\sigma} x.a$.

We say that a dependency structure $\mathcal{S}$ is *acyclic* relative to a skeleton σ if the directed graph defined by $\rightarrow_{\sigma}$ over the variables r and $x.a$ is acyclic. In this case, we can define a coherent probabilistic model over complete instantiations $\mathcal{I}$ consistent with σ:

$$P(\mathcal{I} \mid \Omega, \mathcal{S}, \theta_{\mathcal{S}}) = \prod_{R \in \mathcal{R}} \prod_{r \in R} P(r \mid \text{Pa}(r)) \prod_{X \in \mathcal{E}} \prod_{A \in \mathcal{A}(X)} \prod_{x \in \mathcal{O}^{\Omega}(X)} P(\mathcal{I}_{x.a} \mid \mathcal{I}_{\text{Pa}(x.a)}) \tag{1}$$

Although for each domain, we can compile a PRM into a Bayesian network, a PRM expresses much more information than the resulting BN. A BN defines a probability distribution over a fixed set of attributes. A PRM specifies a distribution over *any* domain; in different domains, the set (and number) of entities will vary, as will the relations between the entities. In a way, PRMs are to BNs as a set of rules in first-order logic is to a set of rules in propositional logic: A rule such as $\forall x, y, z$ *Parent*$(x, y) \wedge$ *Parent*$(y, z) \Rightarrow$ *Grandparent*(x, z) induces a potentially infinite set of ground (propositional) instantiations.

In [8], we address how to learn both the parameters and structure for a PRM which has a fixed skeleton from a relational database. In this paper, we consider a slightly different task of learning the parameters for the probabilistic dependencies in the relational structure and for the dependencies among the probabilistic attributes. Here, the probabilistic dependency structure is given. However, in this task we may have *hidden* or *latent* attributes. As we will see, this complicates the task of parameter estimation.

3 Model-based Approaches to Collaborative Filtering

Having defined the formal models that we use in this paper, we now return to the collaborative filtering task. Presently, we examine how we can use both BNs and PRMs in tackling this task.

In collaborative filtering, we are interested in modeling the relationships between people and items. Recall that items may be actual item (i.e., products) that a person has bought, or an item may refer to a more intangible object, such as a web page visit. Let $\mathcal{X} = \{x^1, \ldots, x^N\}$ represent the people and let $\mathcal{Y} = \{y^1, \ldots, y^M\}$ represent the items. The model-based approaches to collaborative filtering are all based in some way on modeling the probability of the existence of a relation r^{ij} between person x^i and item y^j. Our data consists of instances of the relation R, person-item pairs $\{x^i, y^j\}$. The approaches vary in the independence assumptions made and the complexity of the dependencies that are modeled.

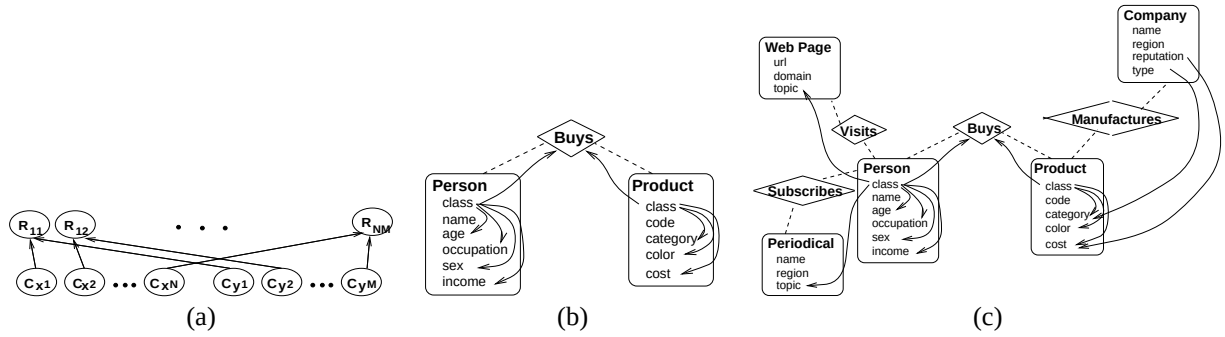

Figure 1: (a) A BN for the two-sided clustering model. (b) A PRM for the two-sided clustering model. The dashed lines show relationships between entities. The solid arrows represent probabilistic dependencies. (c) A PRM with rich relational structure

3.1 Two-sided Clustering in BNs

Recently, several researchers [10; 4] have proposed models that can be viewed as two-sided clustering. In these models, we have a hidden variable C_{x^i} for each person. C_{x^i} represents the (unknown) class or cluster of that person. Similarly, the model also contains a hidden variable C_{y^j} for each item, denoting the cluster of that item. For each person-item pair, we have a variable r^{ij} denoting the relation between person x^i and item y^j. The existence of the relation depends on the cluster of the person, and the cluster of item, hence the notion of *two-sided* clustering. The Bayesian network underlying this model is given in Figure 1(a). In this approach, we have the additional constraint that every r^{ij} must have the same local probability model. That is, the entries in the CPD for $P(r^{ij} \mid C_{x^i}, C_{y^j})$ must be the same for all i and j.

These models are quite attractive as they capture the following intuition: each person belongs to some class c_x with some probability $P(C_x = c_x)$ and each item belongs to some class c_y with probability $P(C_y = c_y)$. The probability that there is a relation R between a person and item depends only on the class of the person and the class of the item.

Now, we turn our attention to learning the parameters of this model. The parameters that must be estimated are θ_{C_X}, θ_{C_Y} and $\theta_{R|C_X,C_Y}$. These values must be estimated using a set of observed relations r^{ij}. Note, however, that since the class variables C_x and C_y are unobserved, we must resort to an estimation technique suitable for dealing with latent variables. Unfortunately, estimating the parameters for this model is non-trivial. The observations r^{ij} cause the hidden class variables for the people and items to be correlated. As a result, on of the standard approach to parameter estimation for latent variables, the EM algorithm [1], becomes infeasible. To address this problem, Ungar and Foster [10] make use of Gibbs sampling, which uses probabilistic random assignment of people and items to clusters, and then recomputes model parameters based on these assignments. While under certain conditions this estimation method provides a guarantee of convergence, it can often take quite long to do so. Other researchers have suggested alternative approaches for parameter estimation in this model, including variational methods [4] and blocked Gibbs sampling [2], with various degrees of success.

Perhaps more important than the choice of parameter optimization procedure is the choice of the number of clusters representing people and items (i.e., the number of states of the variables C_x and C_y, respectively). Previous work with this model [10] simply fixed the number of clusters to some constant value. This may be problematic, however, as the number of clusters determines the number of model parameters and hence the expressive power of this model. If too few clusters are defined, then important correlations between groups of people and groups of items can not be adequately captured by the model. On the other hand, if too many clusters are defined, then no meaningful patterns in the data may be discovered.

One method for approaching this problem relies on an intuition used in Bayesian Network learning: use a function to score models that trades off model complexity (i.e., the number of clusters) and the degree to which the model fits the data. Then, simply search for a model that maximizes this criterion. In this way, the problem of finding a suitable number of clusters is reduced to a problem of evaluating models with different numbers of clusters and selecting the one that maximizes a function such as the previously mentioned Bayesian scoring metric or MDL-based scoring function.

3.2 Collaborative Filtering using PRMs

We can learn PRMs from both structured data and semi-structured data. PRMs are particularly well-suited for aggregate analysis of structured data from multiple sources. In web usage analysis, for example, the information sources might include user assess logs, the relationships between the web pages visited, meta-data on the site and in some cases additional information about the user.

PRMs are also natural models to use for collaborative filtering. We begin by showing how the two-sided clustering model can be compactly represented as a PRM. We next show how the PRM framework naturally extends to allows richer models that capture additional relationships between objects. It is here we see the power of PRMs. PRMs can make use of structured data in a way that flat models, such as BNs, or most other statistical representations, cannot.

Two-sided Clustering in PRMs

The two-sided clustering model is easily and compactly represented as a PRM. Figure 1(b) shows a sample PRM for entities *Person* and *Product* and relation *Buys*. The entity type *Person* has a single fixed attribute *name*. Its probabilistic attributes are: *age*, *occupation*, *sex*, and *income*. The entity type *Product* has a fixed attribute *code* and probabilistic attributes *category*, *color*, and *cost*. In addition, each of the entity types has a hidden attribute *Class*. In this example, we show a simple dependence model for the probabilistic attributes in which the attributes are independent given *Class*. However, more complex dependence models are also feasible. We also have a relationship *Buys* between *Person* and *Product*. *Buys* depends on *Person*.*Class* and $Product$.*Class*. It is easy to verify that this PRM is acyclic. Thus, given a domain Ω, it defines a legal probability distribution over completions of Ω.

The generic two-sided clustering model for collaborative filtering can be represented by a PRM as follows. We have an entity type for people X and an entity type for items Y. We also have a relationship $R(X, Y)$ between people and items. Each of the entity types has a hidden attribute *Class*. R depends on X.*Class* and Y.*Class*. This PRM is acyclic and will define a distribution over all r, x.*class* and y.*class*. Note that the PRM naturally constrains the CPDs for $P(r \mid x.class, y.class)$ to all be the same; it exactly captures constraints inherent in the two-sided clustering model.

Our input is a domain Ω, which describes the set of x_i and y_j that exist in our model, and also includes the set of r_{ij} from our training data. Our task is, from this, to learn a PRM that scores well according to some evaluation criteria. Here, we are given the probabilistic structure for a domain Ω, so we just need to find appropriate parameters θ.

Unfortunately, as we saw in the case for BNs, this is far from trivial. We are currently looking into applying both sampling and variational methods to address the parameter estimation problem.

More Expressive PRMs

In addition to being able to naturally capture the two-sided clustering model, PRMs are easily extended to include additional relational information about the domain. Figure 1(c) shows a hypothetical PRM for the *Person - Buys - Product* example that we have been using thus far. Here we are able to also take into account other related entities in the domain, such as web pages visited, periodical subscriptions and the products' manufacturers. In this way, PRMs allow a simple and compact means for greatly extending the scope of modeling for the collaborative filtering task. This is especially important in the context of the World Wide Web, as a variety of information is often available for use in tasks such as collaborative filtering. For example, we can not only model which web *pages* a buyer of particular products is likely to visit, but we can construct more complex relations having to do with particular *sites* that a user may find interesting by modeling the relationship between people, products, web pages, web sites, and page visits.

In order to learn these richer structures, we must search over the space of PRM structures. Since there are a potentially infinite number of candidate structures, we must have a reasonable strategy to guide our search. In [8], we give an algorithm for performing structure search for PRMs. This algorithm operates in a breadth first manner. We begin by considering dependencies between attributes in the same entity. This is stage 0. At stage k, we consider attributes that can be reached by a chain of at most k relations. Scoring functions over network structure are used to narrow the space of models searched.

4 Deployment of PRMs for Collaborative Filtering

The World Wide Web provides the opportunity to collect a wide variety of information that is naturally represented in a relational form. Here, we give an example of how PRMs may be deployed in the context of the Web to provide rich user modeling and recommendation capabilities.

Consider an electronic commerce web site that provides users with the ability to buy products as well as subscribe to periodicals. As users visit the site, they initially register by providing some demographic information about themselves, such as age, occupation, and sex. Then, as these users browse the site, the URLs they visit are logged. Over time, as users subscribe to periodicals and buy products at the site, this information is recorded in a relational database. This database may also contain supplier information, such as the companies that manufacture each product sold by the web site. A natural relational model for the database underlying this site is shown in Figure 1(c).

Given such a relational scheme, we can learn a PRM that identifies classes of users based on all the information collected, including their propensity to visit particular web pages, buy certain products (from certain manufacturers), or subscribe to various periodicals. By fixing a particular dependency structure between the attributes in each relation, such as that shown in Figure 1(c), we can learn the users' classes by applying the learning strategies suggested in Section 3.2.

Once we learn such a model, we can use it to make recommendations to users coming to the web site. We can suggest other products to buy, periodicals to subscribe to, or web pages to visit. For example, if a registered customer has been

to the web site before (and thus we have collected data on his/her behavior), he/she will be assigned (probabilistically) to a cluster as a result of the previously performed optimization that was conducted to learn the PRM. When such a customer returns to the site, it is possible to recommend to that customer other products to buy or web pages to visit based upon their cluster assignment. In such cases, the advantage of PRMs over existing collaborative filtering methods is that PRMs can represent more complex relationships than simple dyadic data, and can also take into consideration demographic information rather than just purchase or preference history when making collaborative filtering decisions.

An even more compelling advantage in the deployment of PRMs can be seen in the case when a *new* customer comes to visit this web site. Here, the new user begins by registering at the site, and thus provides the system demographic information. Standard approaches to collaborative filtering are unable to make other than apriori recommendations at this point, since the user has no previous purchase/preference history. The PRM, however, can make direct use of the demographic information provided by the user to infer a distribution over the class of the user. After making this determination, the system can then immediately provide recommendations for web pages, periodicals, and products to the user. As the user begins to further explore the site and make purchases, the PRM can be used to update the distribution over the user's class, thereby refining its model and making better collaborative filtering predictions for the user. In this way, PRMs may be successful at providing useful recommendations to users even before collecting a great deal of purchase/preference information, which most standard collaborative filtering approaches require.

5 Related Work and Summary

In this paper we have addressed the collaborative filtering task. More specifically, we have examined probabilistic models for this task, beginning with the two-sided clustering model. While we initially showed that this model can be represented as a Bayesian network, we then went on to show that the new formalism of Probabilistic Relational Models can not only capture the full expressive power of the two-sided clustering model, but can also easily extend it. We also provided an example of how such PRMs may be deployed on the Web for collaborative filtering tasks.

It is important to note that there are several other model-based approaches to collaborative filtering based on Bayesian Networks. Most notably, this work includes the aspect model described in [4], and density estimation models of item co-occurrence [5]. While we could also describe these models (which are in many ways simpler than the two-sided clustering model reviewed here) as PRMs for completeness, we omit that discussion for the sake of brevity.

PRMs provide a compact representation for the complex entity relationships that often need to be modeled in collaborative filtering problems. This representation provides the advantage that the graph structure of PRMs is easier to interpret than the underlying Bayesian Network that models the same probability distribution. For tasks such as collaborative filtering that are modeling a large number of entities, the representational clarity of PRMs is especially compelling. Moreover, we have shown that such PRMs can capture richer relational information. This added modeling power is especially important in the context of the World Wide Web, where there is often much more relational information available than the simple person-item relationships that are currently the focus of collaborative filtering researchers.

In future work, we seek to experimentally verify the efficacy of our theoretical model for collaborative filtering. This work involves both efficient methods for parameter estimation in PRMs, as well as highlighting their rich modeling power in addressing real-world collaborative filtering tasks on the Web.

References

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

[2] B. Engelhart and D. Koller. Personal communication, 1999.

[3] D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20:197–243, 1995.

[4] T. Hofmann and J. Puzicha. Latent class models for collaborative filtering. In *Proc. IJCAI*, 1999.

[5] C. Kadie J. Breese, D. Heckerman. Empirical analysis of predictive algorithms for collaborative filtering. In *UAI*, 1998.

[6] D. Koller and A. Pfeffer. Probabilistic frame-based systems. In *Proc. AAAI*, 1998.

[7] W. Lam and F. Bacchus. Learning Bayesian belief networks: An approach based on the MDL principle. *Computational Intelligence*, 10:269–293, 1994.

[8] D. Koller N. Friedman, L. Getoor and A. Pfeffer. Learning probabilistic relational models. In *Proc. IJCAI*, 1999.

[9] P. Resnick and M. J. Polanco. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997.

[10] L. Ungar and D. Foster. A formal statistical approach to collaborative filtering. In *CONALD'98*, 1998.