Experiments with Multi-component Rating Collaborative Filtering for Improved Recommendation

BY NACHIKETA SAHOO

Abstract

Collaborative filtering is a method to mine the expressed preferences of a group of users about a set of items to filter out uninteresting items for each individual user. They have been used in recommender systems to make personalized recommendation of items that could be interesting to the users. Collaborative filtering algorithms have traditionally been designed to work with user ratings with only one component. However, recently there has been a growing interest in using ratings on multiple aspects of items[2]. Yahoo Movies has started collecting ratings from users on different aspects of the movie such as Story, Acting, Visuals, Direction etc. In this work an approach to carry out collaborative filtering using multi-component ratings is demonstrated. In order to do this, the dependency structure among the components is discovered and incorporated into a mixture model. This algorithm is evaluated using Yahoo Movies! dataset and it is found that use of multiple components leads to improvement in recommendations over using only one component when very little training data is available. However, it is observed that when we have enough training data available we do not gain much from using additional components of the ratings. We also find agreement between the discovered dependency structure and theories proposed in psychometric literature regarding Halo effect with multi-component ratings.

1 Introduction

Recommender systems are more and more commonly used in community-based online places, such as, shopping sites, subscription service sites and online meeting places for various interest groups to recommend items to the community members that they are likely to appreciate. The recommendations are generated from the collective user preferences, yet personalized for each user. This approach is commonly referred as *collaborative filtering*. Recommender systems are especially useful when the user has too many choices to explore all of them himself. They assist the users in discovering potentially interesting items.

On the other hand, used by a retailer it is a useful tool to target-advertise items to its existing customers; items that the customers are likely to like, but, yet were unlikely to have discovered on their own, because of the large number of available items. Thus, a merchant at an online marketplace can use a recommender system to induce demand for the less known items in the system. Example of this can be found at Amazon.com, where customers are recommended items similar to the items they have bought or have rated highly. Another example of a merchant with a business model reliant on recommender system is Netflix, a movie DVD rental-by-mail service with an online storefront. By using its proprietary recommender system Netflix is able to effectively merchandise its collection of about 60,000 movies. They are able to create demand for older and often less known movies by advertising them to users who might like those movies, and are also able to reduce the demand for the newer releases which are more expensive to stock. As reported in the annual SEC filing of the company in 2006, the success of Netflix business model depends, to certain extent, on effective use of, and user satisfaction in relation to, their recommender system [34]. There are many other such stores that use recommender systems to manage demand and enhance user experience, e.g., Yahoo music and iTunes stores for music download recommend new songs to their users based on their past ratings; TiVo has a recommender system for recommending new shows to the customers; half.ebay.com, a sister site of the popular auction site Ebay.com, recommends items based on what users have bought in the past. But, online storefronts are not the only places where recommender systems can be used. There are communities of users with common interests who use recommender systems to find for themselves new items that they might enjoy. Examples of such communities can be found at Last.fm (http://last.fm : an Internet radio station with a music recommender system), StumbleUpon (http://www.stumbleupon.com : a web page recommender system) and KindaKarma (http://www.kindakarma.com : a system to get recommendation on authors, games, movies and music). All this usage suggest that recommender systems are important tools to mine collective user preferences and to help the users better navigate the large choice space they often face.

Despite much of the work that has been done on this topic, several interesting research directions remain unexplored. Readers are referred to the review work by Adomavicius and Tuzhilin for an overview of the recommender system literature and several interesting research ideas [2].

Contributions of this paper

In this work I have focused on the case where we have ratings available on several different aspects of the items—a case not well explored in the collaborative filtering literature. Such ratings on different aspects can be thought of as different components of the rating. I shall focus on the *Collaborative Filtering* approach to generate recommendation, where one mines the collection of ratings by the users of the system on the items in the system and finds new items that are likely to be interesting to a user. I have described an approach to generate item recommendations for a user using a mixture model that uses the components of the ratings and models for the dependency among them. Experiments on a movie dataset show that it leads to improved recommendation when there are very few ratings available for the users—a case that is common for new users who have not rated many movies yet.

In Section 2 I have given a brief overview of the collaborative filtering based recommender system literature. In Section 3 I have touched upon some concepts of the graphical models framework. In Section 4 I have described my proposed model incorporating multiple component ratings and algorithm. In Section 5 I have evaluated the algorithms and discussed the results. In Section 6 I have surveyed the Halo effect literature and drawn parallels between the dependency structure that we found in this work and the theories proposed in psychometric literature regarding Halo effect in multi-component ratings.

2 Literature review

Recommender systems match a user's attributes to the attributes of the items in the system and identify items that might be of interest to the user. The user for which recommendation is being generated is known as the *active user*. Based on what they use as attributes, recommender systems can be divided into two groups: content based recommender system and collaborative filtering based recommender system.

Content based recommender systems

Content based recommender systems recommend items that are similar in their *content*, or intrinsic attributes, to the available profile of the user. For example, an intrinsic attribute of a movie could be the actors who have acted in it or the name of the director, where as the intrinsic attribute of a restaurant could be the cuisine it serves. Content based recommender systems were first developed in the information retrieval and information filtering literature [5]. Because of the potential benefits of a recommender system in the information retrieval applications most of the content based recommender systems are designed and implemented in the information retrieval and information filtering domain. These systems recommend items, usually text documents, to users based on the content of the document, such as keywords ([35], [4]), and the profile of the user. Profile of a user can be constructed in several different ways such as by questionnaires that find out the user's topic of interest or by observing the documents the user has read in the past, etc. User profile can often be represented by a set of *keywords*. Hence, several similarity metrics such as cosine coefficient or Pearson's correlation can be used to find documents that may be interesting to a user.

Content based recommender systems suffer from several drawbacks. As they depend on certain number of predetermined attributes, they cannot distinguish items beyond these attributes. For instance, if two documents use similar keywords, a content based recommender system would not be able to distinguish the well written document from the poorly written one. Another drawback is that they can only recommend items that are similar to the items that the user has already encountered. So, this is limited by the scope of the user's experience.

Collaborative filtering based recommender systems

Collaborative filtering based recommender systems do not use the content of the items. Instead, they deem two items to be similar if they have received similar ratings from users who have rated both of them. Therefore, they can recommend items with content that are completely new to a target user as long as those items have been rated by users who are similar to the target user. This can be thought of as automating the spread of information through word-of-mouth. Because, they make use of a user's peers' ratings on the items they can also distinguish between two items with identical values along content attributes, thus, overcoming the second drawback of the content based recommender systems discussed earlier.

The goal of the collaborative filtering based recommender system is: given a user's ratings on a subset of items and its peers' ratings on possibly different subsets of items, to predict which of the items that the user has not yet rated but would rate highly. The basic approach of collaborative filtering algorithms is to find groups of users with similar "taste" and/or groups of items that are similar, so that to a user u an item that is popular among the group of users who are similar to u can be recommended. One can also recommend an item that is similar to other items that are liked by the user u. The collaborative filtering algorithms differ in how they find similar users or items. The methods can be broadly grouped into two categories: memory based and model based algorithms. In memory based collaborative filtering there is little learning involved. The program simply remembers the ratings of all the users. When it is called upon to make a recommendation for a user, it finds the users similar to the user and makes recommendations based on their ratings (Resnick et al., 1994 [36]). On the other hand, model-based approaches attempt to learn a model of user ratings that explains how the users rate items and then using this model make predictions about the user [19]. Often the model based collaborative filtering algorithms are designed in a probabilistic graphical models framework (described in more detail in Section 3).

It is useful to divide the process of recommendation into two parts: training of the algorithm and generating recommendations. Since, the memory-based methods require little training other than simply remembering the ratings, their training is fast. But, it takes longer to compute the recommendation in this case since all of the similarity computation is done while recommending. On the other hand, with the model-based approach, it takes longer to train the model, but they are usually quick to compute the recommendation for the active user using the learnt model.

In addition to being faster to generate recommendations, model-based collaborative filtering algorithms have other attractive properties. They find a model that can explain the user behavior and then make predictions based on that. Also, they allow us to systematically incorporate our intuition about the rating generation process inside the graphical model framework.

In this work I shall take the model-based approach to carry out collaborative filtering. Hence, it is useful to discuss some important works in model-based collaborative filtering.

Aspect Model One of the first model-based collaborative filtering works has been done by Hofmann and Puzicha[19] based on the Aspect model approach for learning co-occurrence distributions of dyadic data[1][20]. The Aspect model tries to map the co-occurrence of two variables, which in our case would be user variable U and item variable I, into a set of latent classes denoted by variable Z. The set of (u, i) pairs that is mapped to a value z of the variable Z is called an *aspect*. The Aspect model assumes that U and I are independently distributed given the value of the class variable Z. So, the joint distribution of two variables can be stated as a mixture model:

$$P(U, I) = \sum_{Z} P(Z)\, P(U|Z)\, P(I|Z) \tag{1}$$

The latent class variable is used to select a component distribution for the users, $P(U|Z)$, and a component distribution for the items, $P(I|Z)$. The graphical model for this decomposition is given in Figure 1. The observed variables U and I are represented by unshaded nodes and the latent class variable Z (also

1. A dyadic data set is one where each data point consists of paired observations from two abstract sets of objects, e.g., observations of occurrence of (document, word) pairs, or, closer to the topic of this paper, observations of (user, item) pairs.

called the hidden variable or unobserved variable) is represented by the shaded node. As we never observe the latent class variables we cannot directly estimate the parameters of the component distributions of the mixture model. We need to resort to some iterative approach such as Expectation Maximization (EM) to estimate the parameters [10]. EM consists of two steps: Expectation step and Maximization step. In the expectation step the posterior joint distribution of hidden variable is computed using the current estimate of the parameters. In the maximization step, the parameters are estimated so that they maximize the probability of data given the posterior distribution over the hidden variables.

For the purpose of collaborative filtering, Hofmann and Puzicha has extended the aspect model from the dyadic data set to the setting where a third variable such as rating R co-occurs with the two variables U and I [19]. In their model, a set of values of $(u,\ i,\ r)$ that are mapped to one class is defined as an aspect of the data. The graphical model for this approach is given in Figure 1. Here, U is the user, I is one of the items they have rated and R is the rating the items received. The joint distribution of (u, r, i) can be written as a mixture model.

$$P(u,r,i) \;=\; \sum_z P(z)\,P(u|z)\,P(r|z)\,P(i|z) \tag{2}$$

where, u is the value of the user variable, i is the value of the item variable and r is the value of the rating variable. This model proposes that the joint distribution of users, items and ratings is a mixture of three class specific independent distributions. All the variables are multinomial. Therefore, the parameters of these distributions take a form of conditional probability table.

Hofmann and Puzicha has also modeled the distribution of ratings as dependent on the latent class *and* the user (as shown in the 3rd panel of Figure 1), then, we can write the joint distribution as

$$P(u,r,i) \;=\; \sum_z P(z)\,P(u|z)\,P(r|z,u)\,P(i|z) \tag{3}$$

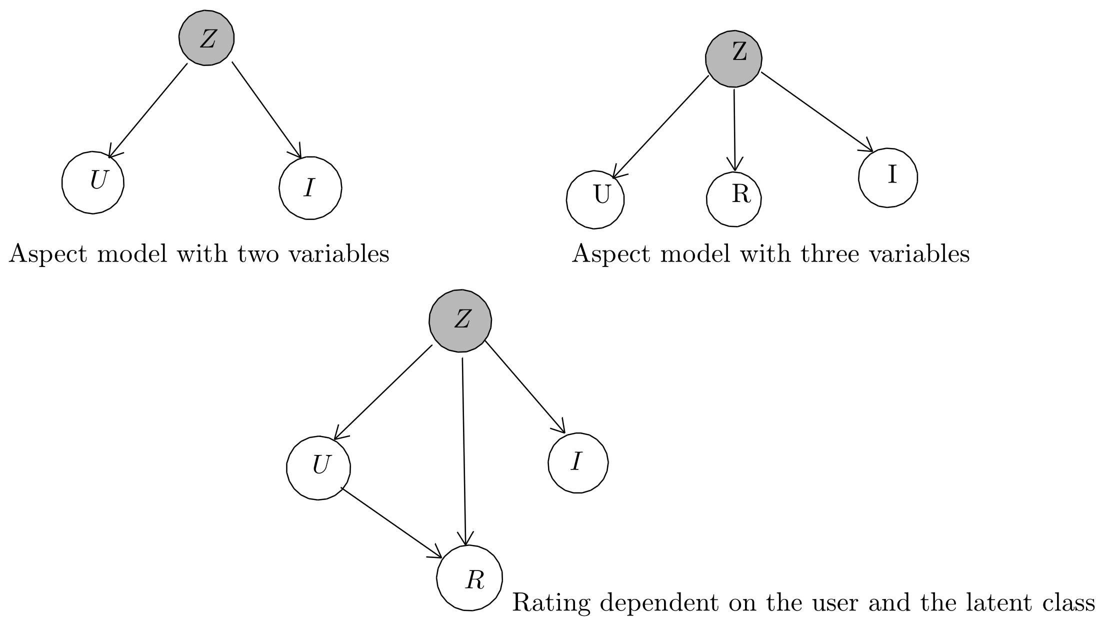
Z
U
I
Aspect model with two variables
Z
U
R
I
Aspect model with three variables
Z
U
I
R
Rating dependent on the user and the latent class

Figure 1. Aspect model in Hofmann and Puzicha, 1999.
Latent variable nodes are shaded and observed variable nodes are not shaded.

Here again, EM algorithm is used to estimate the values of the parameters of $P(u|z)$, $P(i|z)$ and $P(r|z)$ distributions. After the EM algorithm converges we would have all the parameters for the factors of the joint distribution to compute the joint distribution and hence the expected rating given a new (user, item) pair.

Finding the probability of a user occurring for a given value of the latent class variable Z, i.e., $P(u|z)$, can be thought of as the probability of the user appearing in the class or the degree of membership of the user to the class. Hence, this is a process of soft clustering of users, where users can have different degrees of membership to different classes. Estimation of parameters of $P(i|z)$ and $P(r|z)$ can be interpreted in a similar manner. This interpretation of mixture model is especially important when we are clustering users and items for collaborative filtering based on their expressed preference. A user might be interested in different *types* of items, which is reflected in his ratings. Therefore, our model should be flexible enough to accommodate the users' partial membership to different classes or groups. However, note that in aspect model there is only one set of hidden class variables to cluster the occurrences of both the users and the items, which seems limiting.

Flexible mixture model [42] Flexible mixture model, proposed by Luo Si and Rong Jin, attempts to improve upon the aspect model by clustering users and items separately using two latent class variables. The graphical model for this approach is given in Figure 2.

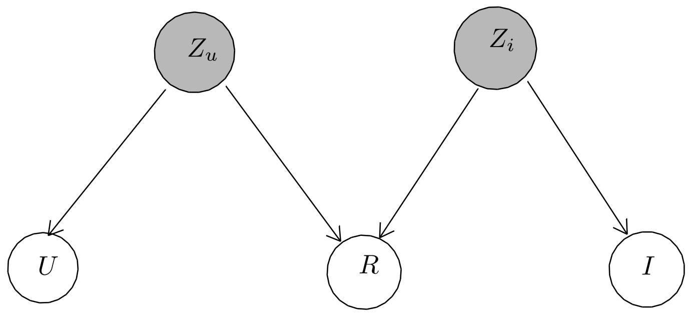

Figure 2. Flexible Mixture model of Luo Si, Rong Jin, 2003

The joint distribution can be expressed as a mixture model with two component selecting variables, namely, Z_u and Z_i. Z_u is responsible for selecting a mixture component for the user variable U and Z_i is responsible for selecting a mixture component for the item variable I. These latent variables can be thought of as soft-clustering users and items to their respective classes. The joint distribution of the three observed variable can thus be expressed as,

$$P(u,r,i) = \sum_{z_u,z_i} P(z_u)\, P(z_i)\, P(u|z_u)\, P(i|z_i)\, P(r|z_u,z_i)$$

The parameters in this model are estimated using the EM algorithm. This approach using separate variables to clusters users and items has been shown to perform better than using one latent class variable to cluster both the users and items as done in aspect model [42].

Although collaborative filtering based recommender systems do not suffer from the same drawbacks as the content based recommender systems they have their own challenges [2].

1. New user problem

   For the users who are new to the system and have not rated many items, it is difficult to generate good recommendation, because, we have very few ratings based on which to model the user preference. One strategy to overcome this limitation is to use a hybrid recommendation system which combines the collaborative filtering with the content-based filtering that uses attributes of the users along with attributes of the items to generate recommendations for a new user. Another strategy could be to ask the user to rate a few items that would provide maximum information about the new user.

2. Sparsity problem

   In any recommendation system the number of (user,item) pairs over which we have ratings is much smaller than the number of pairs over which we need to generate a rating. Hence, an important quality of a recommendation system is its ability to make recommendation from very few data points.

Most of the recommender systems in the literature have been designed for two-dimensional data, i.e., for each (user, item) pair we have one rating indicating how much the user liked the item. However, there might be multiple interesting components to a rating. For instance, the popular Zagat survey rates restaurants on four criteria: food, decor, services and cost (http://www.zagat.com). A movie could be rated for its story, acting, visuals and direction. When such ratings are available from users it is tempting to think that a recommender system could be designed that makes use of these component ratings and produces better recommendations for the users. However, collaborative filtering in the presence of multiple components of the rating has not been well explored in the literature.

It is important to distinguish collaborative filtering using multi-dimensional *ratings* from collaborative filtering in the presence of multi-dimensional *context information*, such as finding a recommendation for a movie to watch on a *Sunday* in the *evening* with *children*. Work have been done in the literature to incorporate multi-dimensional context information into the recommendation generation process, such as the reduction-based approach has been suggested by Adomavicius et. al where context specific recommendation is generated by using only the ratings given in the context of interest [1]. Also, in this work the term *dimension* is used differently than in the work by Sarwar et al., where they have proposed a singular value decomposition based approach to reduce the dimensionality or the size of the two dimensional table of ratings that has one rating value for each user-item pair [39]. In the current work we persue a different agenda: how do we use the information in various components of a rating to make better recommendations for the users?

3 Probabilistic graphical models

Probabilistic graphical models provide a framework based on probability theory and graph theory. They allow us to succinctly express conditional independencies among variables and simplify our operations on the joint distribution. In probabilistic graphical models the variables are represented as nodes and dependence among them is expressed as edges in the graph. Such a graph can be directed or undirected leading to somewhat different ways of working with the variables. A directed graphical model that is free of cycles is called a directed acyclic graph (DAG). Undirected graphical models do not have any such requirements.

A DAG for a set of variables asserts that each node of the graph is independent of its non-descendents conditional on its parents. This is also called Local Markov Assumption. If a DAG asserts only those conditional independencies that exist in the joint distribution of the variables, then it is called an I-map of the distribution. Note that if a graph G is an I-map of a distribution then we can factorize the distribution as

$$P(X_1, X_2, ..., X_n) \;=\; \prod_{i=1}^{n} P(X_i|\mathrm{Pa}_{X_i}) \tag{4}$$

where, X_i is a node in G and Pa_{X_i} is the set of its parents. This follows from the application of chain rule of probability and the use of the conditional independencies of the DAG.

To further understand the conditional independencies encoded in a DAG lets consider the following structures.

	Structure	Conditional Independencies	
1	$X \longrightarrow Y \longrightarrow Z$	$X \perp Z \vert Y$	but, not $X \perp Z$
2	$X \longleftarrow Y \longrightarrow Z$	$X \perp Z \vert Y$	but, not $X \perp Z$
3	But, for $X \longrightarrow Y \longleftarrow Z$	$X \perp Z$	but, not $X \perp Z \vert Y$

Table 1. Some conditional independencies

The first two conditional independencies follow directly from local Markov assumption. The third structure says that if X and Z are common causes of a third variable Y, then conditioned on Y, both the parent variables are dependent. It can be illustrated using a classic example involving Wet Grass, Active Sprinkler and Rain.

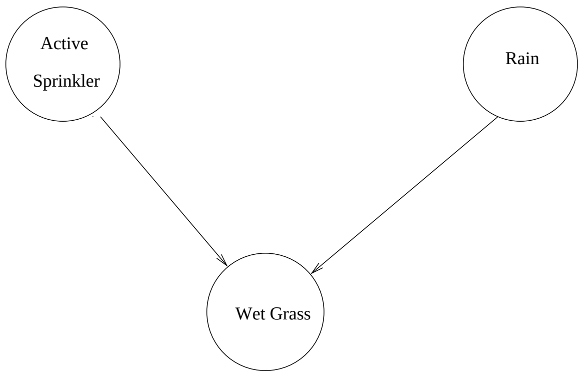

Figure 3. A classic example of explaining away by evidence.

Lets say we are after finding out the probability that it rained last night. If we observe that the grass is wet, then it increases the probability that it rained, i.e., $P(\mathrm{rain}|\mathrm{grass} = '' \mathrm{wet}'')$ is greater than the base probability of rain $P(\mathrm{rain})$. Now lets say we were told that last night the sprinkler was on. Then we would reduce our probability that it rained last night, i.e., $P(\mathrm{rain}|\mathrm{grass} = '' \mathrm{wet}'', \mathrm{sprinkler} = '' \mathrm{on}'') \leqslant P(\mathrm{rain}|\mathrm{grass} = '' \mathrm{wet}'')$. The fact that probability of rain is influenced by the knowledge of event that sprinkler was on, says that we can not assert that “active sprinkler” and “Rain” event are not independent.

It can be said that in the structure 1 and structure 2 in Table 1, there is a dependency path from X to Z only when we do not observe Y, but, in structure 3 there is a dependency path from X to Z only if we observe Y. The argument of *explaining away* can be extended to say that if we have a child node of Y and we have observed that instead of Y, then also X and Z become dependent and so, there is a dependency path between X and Z.

Local Markov assumption and properties of conditional independencies among random variables lead to statements about conditional independencies among any sets of nodes of the graph. It can be shown that if there is no dependency path from node A to node B given the set of conditional variables C then A and B are independent conditional on C. This is known as the d-separation theorem.

The factorization of the joint distribution enabled by the conditional independencies help us in estimating the parameters of the model. Consider the Expression 4 and lets assume that the variables are all multinomial. Without the factorization we would have had to estimate $\left(\prod_{i=1}^{n} |X_i|\right) - 1$ parameters for the joint distribution on the left hand side corresponding to the $\prod_{i=1}^{n} |X_i|$ possible outcomes. Because we can factorize the joint distribution, we need to estimate parameters for n separate conditional probability tables, i.e., only $\sum_i (|X_i| - 1)\, |\mathrm{Pa}_{X_i}|$ parameters. Also, the separation of factors into summands after we take log of the joint probability distribution allows us to estimate parameters of each of these conditional probability tables separately. Thus, we can use all available data to estimate much smaller conditional probability tables, i.e., we get more data per configuration and therefore more data to estimate each parameter.

After the parameters have been estimated in the graphical model, the factorization enabled by the conditional independencies allows us to do fast inference, i.e., prediction involving the distribution of a subset of variables given some evidence on another subset of variables. Such estimation of conditional probability distribution involves marginalization of variables that we are not interested in. When the joint distribution can be expressed as a product of factors as described in Expression 4, we can save a lot of computation by exploiting the distributive property of arithmetic.

Expectation Maximization

Due to the benefits of having conditional independencies in the data, often times it is advantageous to introduce latent variables and conditional independencies that allow us to factorize the joint distribution while preserving the dependencies between observed variables. But, if we have some hidden variables, then we lose the separability of different factors in the joint distribution.

$$P(X_1, X_2, \ldots, X_n, Z_1, \ldots, Z_k) \;=\; \sum_{Z_1 \ldots Z_k} \prod_{i=1}^{n} P(X_i|\mathrm{Pa}_{X_i}) \prod_{j=1}^{k} P(Z_i|\mathrm{Pa}_{Zi}) \tag{5}$$

After we marginalize the latent variables $Z_1, \ldots, Z_k$ the joint distribution need no longer remain separable. The EM algorithm works around this problem [10].

The EM algorithm is a method to find the MLE for the parameters of a distribution which has some missing or unobserved variables. It has two alternating steps.

1. E (expectation) step where one computes the distribution of the unobserved variable given all the observed variables. Note that this is same as doing an inference on the graphical model for the hidden variables.

2. M (maximization) step where one estimates the parameters of the complete dataset (which consists of observed and unobserved variables) using the standard maximum likelihood estimation. This is same as calibrating, or estimating the parameters of, a graphical model.

These two steps monotonically increase the likelihood of the parameters we are trying to estimate.

As we do not have the observations for the missing variables, we can not compute the log-probability of the complete dataset. Therefore, we take the expectation of the log-probability of complete dataset over the missing variable distribution. This gives us a lower bound on the log probability of the observed data ($E_{\text{missing var}}(\log \Pr(\,\cdot\,)) \leqslant \log E_{\text{missing var}}(\Pr(\,\cdot\,))$, because log is concave). In the M-step we maximize this lower bound of the log probability of the observed data.

For the E step it can be shown that among all distributions over the hidden variables the posterior distribution of the hidden variables given the observed data maximizes this expected log-probability, i.e., brings the expected log-probability closest to the log of observed data probability. Intuitively, the posterior distribution over the hidden variables given the observed data is our best guess about the values of the hidden variables.

This establishes EM as an alternating optimization steps, each of which successively improve the probability of observed data or the likelihood of the parameters [25].

The EM algorithm is often used in cases where there are missing observations for some variables in a few records. It is also used in cases where artificially introducing some hidden variables in the dataset significantly simplifies problem. One example of which can be seen in the case of Aspect model for collaborative filtering Figure 1 (top right). In the observed data we have three variables U, I and R each of which is a multinomial variable and they are all dependent. Therefore, one would need to estimate a probability table which would have $|U| \times |R| \times |I| - 1$ parameters, which is quite a large number of parameters and we may not have any data point for most of them (a user usually rates only a few items). But, if we introduce an unobserved variable Z as described in Figure 1, we are introducing a conditional independence assumption, i.e., U, I and R are all independent given the value of the latent class variable Z. This allows us to tackle the problem as a problem of estimating the distribution of a new data set $f(U, R, I, Z)$, which can be decomposed as

$$f(U, R, I, Z) \;=\; P(U|Z)\,P(R|Z)\,P(I|Z)\,P(Z)$$

using the conditional independence. Note that we still retain the dependency between the observed variables U, I and R, but, assert that they are independent given the value of their latent class. In this new model we have $\{(|U|-1)+(|R|-1)+(|I|-1)\} \times |Z| + (|Z|-1)$ number of parameters to estimate, which is much smaller than the original number of parameters when $|U| \gg |Z|$ and $|I| \gg |Z|$, or number of users and number of items are both much larger than number of hidden variables. This is often the case. We can accurately estimate these set of parameters even with limited amount of data in the M step of the algorithm.

4 Proposed multi-component rating recommender system

The idea behind using multiple components for generating recommendations is that by rating multiple aspects of an item users provide richer information about their preference. Without any additional information about the components we expect the component ratings to be correlated, i.e., an item that is liked by a user will receive better ratings along all its components than item that is not liked by a user. Therefore, the information in the correlated components will be less than information provided by independent ratings. But, the variation in different users component ratings even when they seemingly agree on their overall impression of the item might be informative. For instance, consider two users u_1 and u_2 who have rated the same movie m_1 and given similar overall ratings (Table 2).

User	Movie	story	acting	visuals	direction	overall
u_1	m_1	4	1	2	1	2
u_2	m_1	2	1	1	4	2

Table 2. An example of multi-component rating (all ratings are out of 5)

But, they differ in how they rate the components of the movie m_1. The user u_1 seems to like the story of the movie a lot, while the user u_2 seems to like the direction of the movie. Without the component ratings we would have conclude that the users would not like any movie like m_1. But, the component ratings tell us more. They suggest that the user u_1 might like other movies that have a story similar to m_1, while user u_2 might like a movie that has been directed by the same director or a director with similar style. This intuition suggests that if we can effectively use the information in the component ratings provided by the users we should be able to make more accurate recommendations.

This work has been motivated by the availability of component rating data from Yahoo Movies website. Although, the general approach taken in this work is applicable for any data with component ratings, for clarity in discussion I shall describe the methods of this work with the help of above dataset. A description of the dataset follows.

Each record of the rating data consists of seven variables: item or movie id (I), user id (U), ratings on story (S), acting (A), visuals (V), direction(D) and overall (O) quality of the movie. The ratings are on a thirteen point scale ($A+, A, A-, B+, B, C+, C, C-, D+, D, D-, F$). I have recoded them to a scale of 0 to 4 ($\{A+, A, A-\} \longrightarrow 4, \{B+, B, B-\} \longrightarrow 3, \{C+, C, C-\} \longrightarrow 2, \{D+, D, D-\} \longrightarrow 1, \{F\} \longrightarrow 0$), so that there will be enough data points in each rating bucket. Ratings on a set of 5628 movies were collected. Although, there were 691496 records in the original dataset, the user frequency in the data turns out to be very skewed (Figure 4).

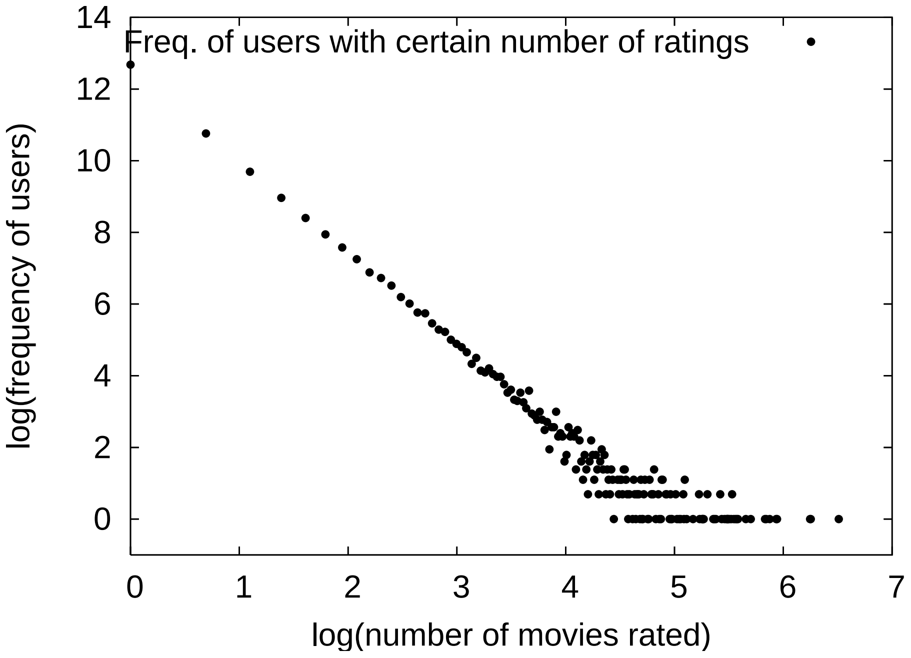

Figure 4. log-log plot of frequency of users who have rated a certain number of movies.

Ratings from users who have rated very few movies is not useful for collaborative filtering purpose, since, we can not reliably know the preferences of a user from only a few of his ratings. Therefore, I have retained only those records that contain users who have at least 20 ratings. After this filtering there were 45892 records, 1058 unique users and 3430 unique movies.

In the remainder of this section I shall describe the approach I have taken to carry out collaborative filtering in the presence of component rating information in the data, with the help of the above mentioned dataset to keep the discussion grounded to a real life scenario.

4.1 Modeling component ratings

To incorporate this extra information into the model based collaborative-filtering algorithm, I use a mixture model similar to the Flexible Mixture Model proposed in [42]. The FMM model proposes that the rating an item received from a user is governed by a small number of latent classes of the user and a small number of latent classes of the item. It asserts that given the latent classes of a user and the latent classes of an item, the rating is independent of the particular user or the particular item.

However, when the rating has components, they may not be conditionally independent given the latent class of the user or the item. In fact as pointed out in Section 4, they are likely to be highly correlated. An incorrect independent assertion among the component ratings in the model would lead us to believe that each rating component provides completely new additional information about the user preferences, which is not true. This would lead to over counting of evidence.

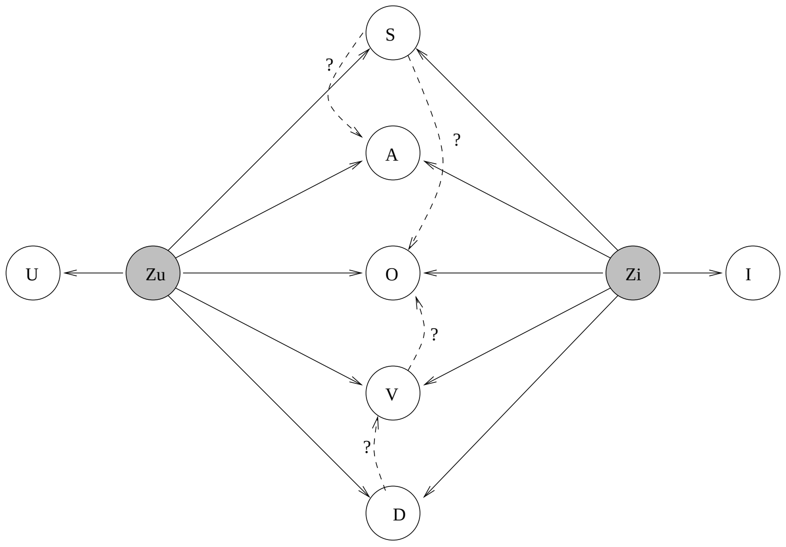

Figure 5. Flexible Mixture model for component rating collaborative filtering

Therefore, it is necessary to find the dependency structure among the five components of the rating. To estimate the number of different subgraphs among these five components, please note that there are ten unique pairs, each of which can have an edge between them in either direction or have no edge at all. This leads to 3^{10} less the number of edge combinations that lead to cycles, which is a large number of structures to search through to find the best dependency structure. Another problem with doing a complete model search is that the configurations with multiple parents will lead to large conditional probability tables for which we may not have enough data to estimate the probabilities reliably. Chow and Liu has shown that if we have only discrete variables and we restrict our ourselves to only those structures in which we have at most one parent for each node, i.e., trees, we can efficiently search through the set of candidate structures [7]. Also, having at most one parent for each node ensures that we have more manageable conditional probability tables to estimate.

Chow-Liu tree structure discovery [7]

The objective here is to discover the tree structure among all available tree structures that captures most of the dependence among the variables, i.e., the tree structure that leads to the highest probability of data. There are several metric to measure the dependence between sets of variables. One such metric of particular interest is mutual information.

$$I(X,Y) \;=\; \sum_X \sum_Y P(X,Y)\log\!\left(\frac{P(X,Y)}{P(X)\,P(Y)}\right) \tag{6}$$

Note that if X and Y are independent then the mutual independence between two variables is zero, else it is positive.

It has been shown elsewhere that if joint probability distribution of a set of variables can be broken into factors as shown in Expression 4, then log probability of data can be broken into the mutual information terms computed from empirical distribution involving variables of each factor and entropy of each node [27].

$$\log P(D|G) \;=\; L\sum_{i=1}^{n} \hat{I}(X_i, \mathrm{Pa}_{X_i}) - L\sum_{i=1}^{n} \hat{H}(X_i) \tag{7}$$

where,

n = number of variables or nodes in the graph

L= total number of records

$\hat{I}$ = mutual information between a variable X_i and its parents Pa_{X_i} computed from the empirical distribution

$\hat{H}$= entropy of the random variable computed from its empirical distribution

So, when we are trying to find the structure, i.e., searching for sets of parent nodes for each node, that maximizes the probability of the data, we can separately maximize empirical mutual information components and note that this does not affect the entropy components [27].

Chow and Liu observe that when each node can have at most one parent (true for trees), we can compute the empirical mutual information between each pair of variables. This gives the strength of dependence between each pair of variables. Then a Maximum Weight Spanning Tree over these variables would give us a tree structure that captures the most dependence among the variables, or in other words, it will give us the tree structure over the variables that will lead to maximum log-probability of the data. Note that this gives a undirected graph, i.e., there is no natural parent child relationship between the pairs of node we chose to put an edge between, since mutual information is symmetrical with respect to the order of the variables. However, we can choose any one node as the root of the tree and put outgoing edges from the root to get a DAG (Figure 6). All possible DAGs obtained in this manner have the same conditional independencies and hence, are equivalent.

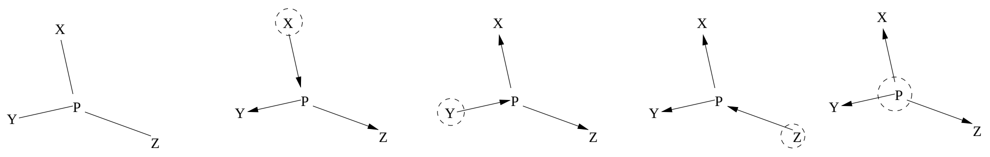

Figure 6. Undirected graph to DAG

Such an exercise over the five component ratings in the data leads to the following structure. The pairwise mutual information ($\hat{I}$) table is given on the left hand side of Figure 21.

	S	A	V	D	O
S	–	0.61	0.72	0.51	0.88
A	0.61	–	0.68	0.47	0.73
V	0.72	0.68	–	0.64	0.92
D	0.51	0.47	0.64	–	0.66
O	0.88	0.73	0.92	0.66	–

MST $\Longrightarrow$

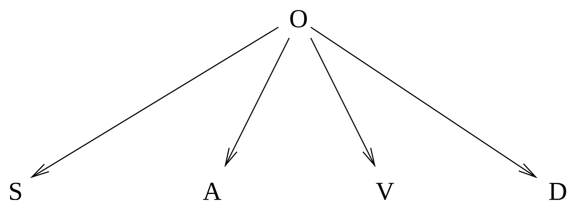

Figure 7. Discovered structure in the sub ratings

This suggests that the strongest of the dependencies between the components of the ratings in the data is between the overall rating and the components. In the data we would find evidence of dependence between any pair of variables, but, for any of the components changing the parent from O to any other variable (under the tree structure restriction one variable can have at most one parent) would lead to a lower log probability of the data. Another way of reading this discovered structure is: given the overall rating the components are independent. Note that this dependency structure says that if we do not condition on the Overall rating, then the S, A, V, D variables are dependent or correlated, which is consistent with the expectation we started out with.

4.2 EM algorithm

Using this discovered structure between the components of the ratings, we can construct the following model for collaborative filtering.

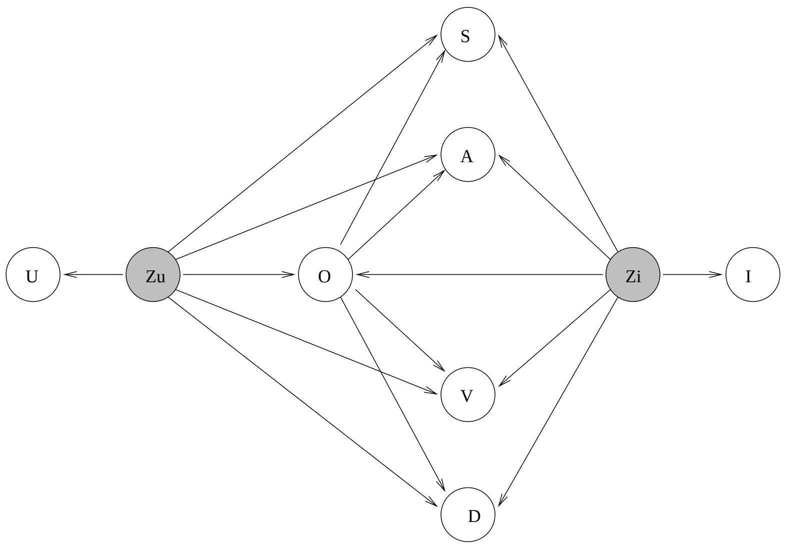

Figure 8. Flexible Mixture Model with component rating dependency structure.
Z_u = Latent class variable to cluster the users, Z_i = Latent class variable to cluster the items
O = Overall rating, S = rating on Story, A = rating on Acting, V = rating on Visuals,
D = rating on Direction, U = user variable and I = item variable.
Latent variables are shaded and observed variables are unshaded

As we have hidden variables, the parameters of the model needs to be estimated using some indirect method like the EM algorithm. The E and the M steps for the case when all components are assumed independent are derived in a similar manner. It can be shown that the E and the M step of the algorithm are:

E-step

$$P(Z_u, Z_i|\vec{X}) = \frac{P(Z_u)\,P(Z_i)\,P(I|Z_i)\,P(U|Z_u)\prod_{j=1}^{5} P(R_j|Z_u, Z_i, \mathrm{Pa}_{R_j})}{\sum_{Z_u}\sum_{Z_i} P(Z_u)\,P(Z_i)\,P(I|Z_i)\,P(U|Z_u)\prod_{j=1}^{5} P\big(R_j|Z_u, Z_i, \mathrm{Pa}_{R_j}\big)} \tag{8}$$

M-step

$$P(Z_u) = \frac{1}{L}\sum_{l}\sum_{Z_i} P(Z_u, Z_i|\overrightarrow{X_{(l)}}) \tag{9}$$

$$P(Z_i) = \frac{1}{L}\sum_{l}\sum_{Z_u} P(Z_u, Z_i|\overrightarrow{X_{(l)}}) \tag{10}$$

$$P(U|Z_u) = \frac{\sum_{l:U_{(l)}=U}\sum_{Z_i} P(Z_u, Z_i|\overrightarrow{X_{(l)}})}{\sum_{l}\sum_{Z_i} P(Z_u, Z_i|X_l)} \tag{11}$$

$$P(I|Z_i) = \frac{\sum_{l:I_{(l)}=I}\sum_{Z_u} P\Big(Z_u, Z_i|\overrightarrow{X_{(l)}}\Big)}{\sum_{l}\sum_{Z_u} P\Big(Z_u, Z_i|\overrightarrow{X_{(l)}}\Big)} \tag{12}$$

$$P\big(R_j|Z_u, Z_i, \mathrm{Pa}_{R_j}\big) = \frac{\sum_{l:R_{j(l)}=R_j\,\&\,\mathrm{Pa}_{\mathrm{R\,j}(l)}=\mathrm{Pa}_{\mathrm{R\,j}}} P\Big(Z_u, Z_i|\overrightarrow{X_{(l)}}\Big)}{\sum_{l:\mathrm{Pa}_{\mathrm{R\,j}(l)}=\mathrm{Pa}_{\mathrm{R\,j}}} P\Big(Z_u, Z_i|\overrightarrow{X_{(l)}}\Big)} \tag{13}$$

where,

$R_j =j$th rating node. $R_j \in \{S, A, V, D, O\}$

$\mathrm{Pa}_{R_j} =$parent rating node of R_j as discovered in the structure discovery method

$L =$number of records in the dataset

$l =$ index numbered l

$\overrightarrow{X_{(l)}} =$record numbered l. It consists of observations for U, I, S, A, V, D, O

$U_{(l)} =$observation of variable U in the record numbered l

$I_{(l)} =$observation of variable I in the record numbered l

$R_{j(l)} =$observation of the rating variable R_j in the record numbered l

$\mathrm{Pa}_{R_j(l)} =$ observation of the rating variable Pa_{R_j}, which is the parent node of rating variable R_j in the record numbered l.

To intuitively understand these set of equations lets remind ourselves that in EM algorithm, the E-step consists of finding the conditional distribution of the hidden variables given the observed variables using the current values of the parameters. The E-step shown above is this conditional distribution computed by dividing joint distribution of all variables (factorized using the conditional independencies) by the joint distribution of only the observed variables (obtained by marginalizing out the hidden variables). The M-step in the EM algorithm consists of finding the MLE of the parameters of complete data (observed and unobserved variables) distribution. If we could observe all variables, we could find the MLE of parameters of each conditional probability table by dividing the number of records with matching values for all the variables in the conditional probability table by the total number of records with matching values of the conditioning variables. But, we do not observe the hidden variables. Therefore, we have to use our best guess about their number of occurrences or their expected occurrence counts at a record given the observations of other variables in the same record. This is obtained from the posterior distribution of the hidden variable. Since, this conditional distribution is multinomial the expected number of times a hidden variable takes a certain value in one record is same as the probability of the hidden variable taking that value given the value of the observed variables. All equations of the M-step can be obtained by tabulating the values of the observed variables and, for the hidden variables, using the expected number of times the hidden variables take a certain value.

We compare the above model with the model that assumes independence among the component ratings conditional on the latent classes (Figure 9).

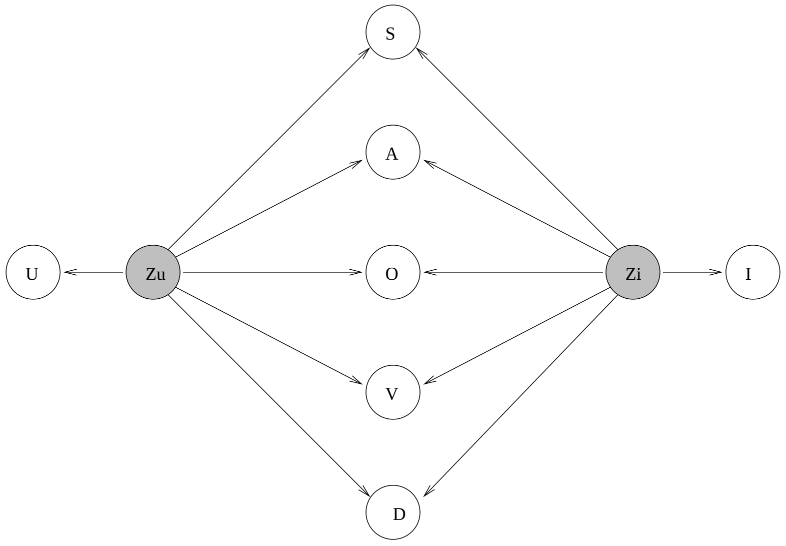

Figure 9. Component ratings are assumed independent given latent classes.
$Z_u =$Latent class variable to cluster the users, $Z_i =$Latent class variable to cluster the items
$O =$ Overall rating, $S =$rating on Story, $A =$rating on Acting, $V =$rating on Visuals,
$D =$rating on Direction, $U =$user variable and $I =$item variable.
Latent variables are shaded and observed variables are unshaded

The E and the M steps for the case when all components are assumed independent are derived in a similar manner.

E-step

$$P(Z_u, Z_i|\vec{X}) \;=\; \frac{P(Z_u)\,P(Z_i)\,P(I|Z_i)\,P(U|Z_u)\prod_{j=1}^{5} P(R_j|Z_u, Z_i)}{\sum_{Z_u}\sum_{Z_i} P(Z_u)\,P(Z_i)\,P(I|Z_i)\,P(U|Z_u)\prod_{j=1}^{5} P(R_j|Z_u, Z_i)} \tag{14}$$

M-step

$$P(Z_u) \;=\; \frac{1}{L}\sum_{l}\sum_{Z_i} P(Z_u, Z_i|\overrightarrow{X_{(l)}}) \tag{15}$$

$$P(Z_i) \;=\; \frac{1}{L}\sum_{l}\sum_{Z_u} P(Z_u, Z_i|\overrightarrow{X_{(l)}}) \tag{16}$$

$$P(U|Z_u) \;=\; \frac{\sum_{l:U_{(l)}=U}\sum_{Z_i} P(Z_u, Z_i|\overrightarrow{X_{(l)}})}{\sum_{l}\sum_{Z_i} P(Z_u, Z_i|X_l)} \tag{17}$$

$$P(I|Z_i) \;=\; \frac{\sum_{l:I_{(l)}=I}\sum_{Z_u} P\Big(Z_u, Z_i|\overrightarrow{X_{(l)}}\Big)}{\sum_{l}\sum_{Z_u} P\Big(Z_u, Z_i|\overrightarrow{X_{(l)}}\Big)} \tag{18}$$

$$P(R_j|Z_u, Z_i) \;=\; \frac{\sum_{l:R_{j(l)}=R_j} P\Big(Z_u, Z_i|\overrightarrow{X_{(l)}}\Big)}{\sum_{l} P\Big(Z_u, Z_i|\overrightarrow{X_{(l)}}\Big)} \tag{19}$$

Note, that the key difference between Expressions 14 to 19 and the Expressions 8 to 13, is the absence of any parent node in the conditioning part of the conditional probability of the component ratings. The intuitions behind these equation is similar to those described for the previous set.

We also compare these approaches with the case where there is only one rating: the over all rating on the movie. The E and the M steps when there is only one rating can be borrowed from [42].

E-step

$$P(Z_u, Z_i|U, I, O) \;=\; \frac{P(Z_u)\,P(Z_i)\,P(I|Z_i)\,P(U|Z_u)\,P(O|Z_u, Z_i)}{\sum_{Z_u}\sum_{Z_i} P(Z_u)\,P(Z_i)\,P(I|Z_i)\,P(U|Z_u)\,P(O|Z_u, Z_i)} \tag{20}$$

M-step

$$P(Z_u) \;=\; \frac{1}{L}\sum_{l}\sum_{Z_i} P\big(Z_u, Z_i\big|U_{(l)}, I_{(l)}, O_{(l)}\big) \tag{21}$$

$$P(Z_i) \;=\; \frac{1}{L}\sum_{l}\sum_{Z_u} P\big(Z_u, Z_i\big|U_{(l)}, I_{(l)}, O_{(l)}\big) \tag{22}$$

$$P(U|Z_u) \;=\; \frac{\sum_{l:U_{(l)}=U}\sum_{Z_i} P\big(Z_u, Z_i\big|U_{(l)}, I_{(l)}, O_{(l)}\big)}{L\times P(Z_u)} \tag{23}$$

$$P(I|Z_i) \;=\; \frac{\sum_{l:I_{(l)}=I}\sum_{Z_u} P\big(Z_u, Z_i\big|U_{(l)}, I_{(l)}, O_{(l)}\big)}{L\times P(Z_i)} \tag{24}$$

$$P(O|Z_u, Z_i) \;=\; \frac{\sum_{l:O_{(l)}=O}\sum_{Z_u} P\big(Z_u, Z_i\big|U_{(l)}, I_{(l)}, O_{(l)}\big)}{\sum_{l} P\big(Z_u, Z_i\big|U_{(l)}, I_{(l)}, O_{(l)}\big)} \tag{25}$$

Smoothing of parameter estimates using BDe prior [27]

To guard against over fitting to the training data I have smoothed the parameter estimates in the M-step using Dirichlet prior, which is a multivariate generalization of Beta distribution and conjugate prior for multinomial distribution. A variable (θ_1, ..., θ_K) following Dirichlet distribution with hyper-parameters $(\alpha_1, ..., \alpha_K)$ has the probability distribution

$$P(\theta_1, ..., \theta_K) \;\sim\; \prod_{k} \theta_k^{\alpha_k - 1}$$

The expected value of $\theta_k = \frac{\alpha_k}{\sum_{k=1}^{k=K} \alpha_k}$. If we update this prior using multinomial data with counts $M_1, ...,$ M_K, then we obtain the posterior with another Dirichlet distribution with hyper parameters $(\alpha_1 + M_1, ...,$ $\alpha_K + M_K)$. Thus, the expected values of the components can be obtained by adding the counts to the numerator and denominator of the original formula.

$$E(\theta_k) \;=\; \frac{M_k + \alpha_k}{\left(\sum_k M_k\right) + \left(\sum_k \alpha_k\right)} \tag{26}$$

Therefore the α's can be thought of as pseudocounts and the sum of α's is a measure of the weight of the prior.

BDe prior is a Dirichlet prior constructed so that the weigth of the prior would be same for each node in the Bayesnet: irrespective of how many parents and thus conditional probability tables we have for each of them. This is achieved by using a simple probability distribution over the entire set of variables: say $P'(U, I, S, A, V, D, O, Z_u, Z_i)$. Then for a conditional probability parameter such as $P(U|Z_u)$ we can use a Dirichlet distribution with hyper-parameters $\alpha_{u|z_u} = m \times P'(U, Z_u)$. Where, m is the weight of the prior we want to maintain at each node. If we add all values of $\alpha_{u|z_u}$ we get m. Thus for each node the weight of the prior is a constant. For this exercise I used a uniform discrete probability distribution $P'(U, I, S, A,$ $V, D, O, Z_u, Z_i)$. This leads to simple calculation of α's for the Dirichlet priors of all conditional probabilities. Smoothing using these Dirichlet priors involve treating these α's as the pseudo-counts and adding them to the numerator and denominators as described in Equation 26. m was set at 5 for all experiments.

A much more detailed discussion of smoothing using Dirichlet prior and BDe prior can be found in the book *Bayesian Network and Beyond* by Koller and Friedman (Chapters 12 and 13) [27].

4.3 Making a prediction

The goal is to predict the rating our active user u_a would give to an item he has not yet rated. Hence, our partial observation consists of the user and the item, and we are trying to say something about the rating that we might see.

After we have trained our model and estimated the parameters, we can write down the joint distribution over all the variables (observed and latent) in a factored form. Then we need to marginalize away those variables that we are not interested in. In this work we have focused on our ability to predict the overall rating. So, we need to marginalize all variables except U, I and O.

With dependency among the component ratings

$$\begin{aligned}
P(U, I, O) &= \sum_{Z_u}\sum_{Z_i}\sum_{S}\sum_{A}\sum_{V}\sum_{D} P(Z_u, Z_i, U, I, S, A, V, D, O)\\
&= \sum_{Z_u}\sum_{Z_i}\sum_{S}\sum_{A}\sum_{V}\sum_{D} P(Z_u)\,P(Z_i)\,P(I|Z_i)\,P(U|Z_u)\,P(S|Z_u, Z_i, O)\\
&\quad P(A|Z_u, Z_i, O)\,P(V|Z_u, Z_i, O)\,P(D|Z_u, Z_i, O)\,P(O|Z_u, Z_i)\\
&= \sum_{Z_u}\sum_{Z_i} P(O|Z_u, Z_i)\,P(Z_u)\,P(Z_i)\,P(I|Z_i)\,P(U|Z_u)\sum_{S} P(S|Z_u, Z_i, O)\\
&\quad \sum_{A} P(A|Z_u, Z_i, O)\sum_{V} P(V|Z_u, Z_i, O)\sum_{D} P(D|Z_u, Z_i, O)\\
&= \sum_{Z_u}\sum_{Z_i} P(O|Z_u, Z_i)\,P(Z_u)\,P(Z_i)\,P(I|Z_i)\,P(U|Z_u)\\
&= \sum_{Z_u} P(Z_u)\sum_{Z_i} P(O|Z_u, Z_i)\,P(Z_i)\,P(I|Z_i)\,P(U|Z_u)
\end{aligned} \tag{27}$$

The conditional probability terms for S, A, V and D could be marginalized and eliminated, since those probabilities sum to 1. We can compute this final expression because we have estimated all the terms in the right hand side of Expression (27) as our parameters.

Independent component ratings

$$\begin{aligned}
P(U, I, O) &= \sum_{Z_u}\sum_{Z_i}\sum_{S}\sum_{A}\sum_{V}\sum_{D} P(Z_u, Z_i, U, I, S, A, V, D, O)\\
&= \sum_{Z_u}\sum_{Z_i}\sum_{S}\sum_{A}\sum_{V}\sum_{D} P(Z_u)\,P(Z_i)\,P(I|Z_i)\,P(U|Z_u)\,P(S|Z_u, Z_i)\\
&\quad P(A|Z_u, Z_i)\,P(V|Z_u, Z_i)\,P(D|Z_u, Z_i)\,P(O|Z_u, Z_i)
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{Z_u}\sum_{Z_i} P(O|Z_u,Z_i)\,P(Z_u)\,P(Z_i)\,P(I|Z_i)\,P(U|Z_u)\sum_{S} P(S|Z_u,Z_i) \\
&\quad \sum_{A} P(A|Z_u,Z_i)\sum_{V} P(V|Z_u,Z_i)\sum_{D} P(D|Z_u,Z_i) \\
&= \sum_{Z_u}\sum_{Z_i} P(O|Z_u,Z_i)\,P(Z_u)\,P(Z_i)\,P(I|Z_i)\,P(U|Z_u) \\
&= \sum_{Z_u} P(Z_u)\sum_{Z_i} P(O|Z_u,Z_i)\,P(Z_i)\,P(I|Z_i)\,P(U|Z_u) \qquad (28)
\end{aligned}
$$

Note that this is identical to the expression we got in Equation (27), but, the parameters we obtain in this model will be different, because of different conditional independency assumptions.

Using only overall rating

$$
\begin{aligned}
P(U,I,O) &= \sum_{Z_u}\sum_{Z_i} P(Z_u,Z_i,U,I,O) \\
&= \sum_{Z_u}\sum_{Z_i} P(O|Z_u,Z_i)\,P(Z_u)\,P(Z_i)\,P(I|Z_i)\,P(U|Z_u) \\
&= \sum_{Z_u} P(Z_u)\sum_{Z_i} P(O|Z_u,Z_i)\,P(Z_i)\,P(I|Z_i)\,P(U|Z_u) \qquad (29)
\end{aligned}
$$

Here, also the expression looks similar to the expression we obtained for the previous two cases, but, yet again the parameters would be quite different in this case because we are using a different model with different conditional independence assumptions.

As U, I and O are all multinomial, $P(U,\ I,\ O)$ can be thought of as a probability table with three dimensions, or a probability cube. This can be quite large in a real life setting where we have a reasonably large number of users and items. Consequently it may not fit in the virtual memory. However, note that we can compute these expressions for one particular user (and even one particular item combination). This allows us to generate ratings for users sequentially and reduces the memory requirements to a manageable level.

If the value of the user variable is u_a (active user) and the value of the item variable is i, then we are interested in the conditional distribution of the overall rating given these values of the user and item.

$$
P(O|u_a,i) = \frac{P(u_a,i,O)}{\sum_O P(u_a,i,O)} \qquad (30)
$$

As the variables in these models are *multinomial*, the most likely value of variable O is predicted as the output. In their work Si and Jin use the expectation of O instead [42]. I believe the former is more theoretically sound, since, in general, average of values multinomial variables, such as weighted average of head and tail events, do not lead to meaningful value. As we shall see in the next section the most likely rating prediction is more accurate as well.

5 Results and discussion

5.1 Experiments with Random Training Sample

In this section we compare the effectiveness of the three models described above for the collaborative filtering task. To do this we use a fraction of the user ratings to train our models (training set) and use the remaining to test the performance (test set). A certain fraction of each users' records were randomly selected to include in the training data to make sure that there is some training data for each user in the test set. For each user-item pairs in the test data we predict their overall rating (O) using each of the above mentioned model. The algorithms were evaluated for their accuracy in predicting what rating the user is going to give to an item and their ability to identify the highest rated items. These two evaluations need not be correlated. Each of these might be appropriate depending on the application scenario [18].

In an application where the user is recommended the items along with the ratings they might assign to the items it is important to be able to predict the numerical values of these ratings accurately. One example of such application environment can be found in the movie recommender system of Netflix. The rating prediction accuracy can be measured using Mean Absolute Error (MAE) of the predictions.

However, in many other applications the user is only presented the top few items he is likely to rate highly. The numerical values of the items are not presented to the user. One example of such an application environment can be found at Amazon.com. In such a scenario, if the recommender system can identify the top few items for the user with little noise then it can be considered to have fulfilled the requirement. It does not matter if all the predicted ratings are biased up or down by a rating point or two, as long as the predicted ratings order the items in the same way as the user would, i.e., assign a relatively higher rating for items that the user would rate A followed by lower ratings to items that the user would rate B and so on. This correctness of such ordering of items for users can be evaluated using Precision-Recall plots. To describe this briefly, lets assume for a moment that the user would be only interested in the items rated A. If the top-N predicted items are considered, *precision* is the fraction of the N items that the user would have given rating A and *recall* is the fraction of items that the user would have rated A that are in the the top-N. As we increase N we would retrieve more and more A-rated items, improving recall, but, at the same time we would also retrieve more and more items that are rated less than A that will damage the *precision* score. A good recommender system should be able to retrieve as much of the A-rated items while maintaining high precision. Hence, a plot of precision at 11 standard recall levels (0%, 10%, 20%, ..., 100%) is commonly used as a tool to compare the performance of different algorithms [3].

5.1.1 Accuracy in rating prediction

The error of the prediction is computed by comparing it with the available rating in the test data. We used mean absolute error to compute the error.

$$\text{MAE} = \frac{1}{L_{\text{test}}} \sum_{l=1}^{L_{\text{test}}} |o_l - \hat{o_l}|$$

where,

$L_{\text{test}} =$ the number of records in the test data

$o_l =$ the true rating information

$\hat{o_l} =$ the predicted rating information

These experiments were repeated using different amounts of training data. For each of these training data sizes, training and testing exercise were repeated using thirty different random samples to estimate the variability in the MAE metric. The average of the 30 MAE scores are described in Table 3 and Figure 10.

Col 0	Col 1	Col 2	Col 3	Col 4
Prediction method	FMM_O	FMM_O	$\text{FMM}_{5-\text{indep}}$	$\text{FMM}_{5-\text{dep}}$
Training fraction	Expectation	Mode	Mode	Mode
0.025	1.16	1.18	1.11	0.95
0.050	1.07	1.07	1.08	0.94
0.075	0.96	0.96	0.98	0.90
0.100	0.88	0.89	0.94	0.86
0.125	0.84	0.85	0.90	0.83
0.150	0.82	0.82	0.87	0.82
0.175	0.80	0.80	0.84	0.79
0.20	0.79	0.78	0.82	0.78
0.3	0.76	0.74	0.79	0.75
0.4	0.75	0.72	0.76	0.74
0.5	0.74	0.71	0.74	0.73
0.6	0.73	0.69	0.73	0.72
0.7	0.72	0.69	0.73	0.71
0.8	0.72	0.68	0.72	0.71
0.9	0.72	0.67	0.72	0.70

Table 3. Error comparison of different models

Table cells contain MAE on a 5 point wide scale

FMM_O: Flexible Mixture Model using only overall rating

$\text{FMM}_{5-\text{indep}}$: Flexible Mixture Model using all five ratings without dependency structure

$\text{FMM}_{5-\text{dep}}$: Flexible Mixture Model using all five ratings with discovered Chow-Liu structure

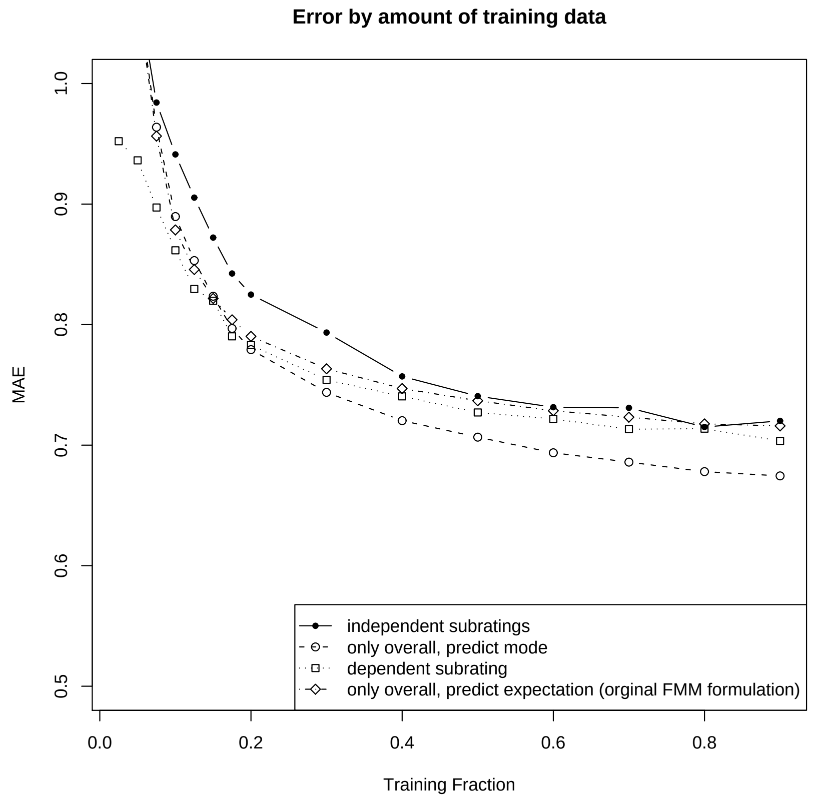

Figure 10. Plot of errors by amount of training data used, for different models.

For each model the overall error keeps decreasing as we use more and more training data to train the model, which is expected. Comparing column 1 and 2 we find that mode of the distribution over the rating variable turns out to be a better predictor of the unknown rating than the mean. As argued in Section 4.3 this is also theoretically more appropriate for multinomial variables.

Comparing column 2 and 3, we see that naively extending existing Flexible Mixture Model for collaborative filtering with component ratings without considering the existing correlation among the component ratings do not lead to any improvement in the prediction of overall ratings. As we discussed in Section 4.1, components of the ratings are correlated and assuming independence among them given latent class leads to over counting of evidence. For instance, if a user has given an Overall rating of 4 to a movie, then we would expect him to give a high rating of 3 or 4 along the Acting component of the movie. Thus, in this case observation of a rating of 4 on the acting component is not as informative as the independent model would have us believe.

When we capture these strongest pairwise dependence among the rating components and the Overall rating and explicitly model for it, the prediction accuracy improves (column 3 vs. column 4). The particular structure of the tree generated is also quite informative. It says that rating components are more correlated to the Overall rating than to any other rating component. This might be indicative of the strong influence of the overall impression of a user about a movie on the perception of other aspects of the movie.

However, the question we are the most interested in is "Can we make better prediction of the Overall rating using more components of ratings?". Comparing column 2 and 4, we see that we can make improved predictions when we are using very small amount of training data for the users. This is an important scenario in any real world collaborative filtering system. We often have a lot of new users in the system who have rated only a few items. All the existing collaborative filtering systems perform badly for these new users. However, as we can see from column 4, for these set of users using a richer set of ratings can help. But, when we have enough training data, using only the overall rating leads to more accurate prediction of Overall rating. This suggests that when we have a users' Overall rating over a large number of items adding component ratings does not lead to any further improvement in our ability to predict the overall rating the user might place on a new item.

To verify that the difference in the average MAE that we see in Table 3 are significant and not a result of random occurrences, I performed a pairwise $t-$test using MAE obtained at the 30 different train/test splits. The comparisons of different pairs of models are shown in Table 4. As we can see the differences discussed are indeed significant. The only exception is the performance of two prediction methods when we use only overall rating to generate predictions using small amount of training data. In this particular setting the difference is small and often not significant. But, when we use more and more training data the difference between these two prediction methods become more apparent.

training fraction	$\text{FMM}_{5-\text{dep}}$ vs FMM_O	$\text{FMM}_{5-\text{dep}}$ vs $\text{FMM}_{5-\text{indep}}$	FMM_O
	Mode	Mode	Mode vs Expectation
	avg. diff. in error (p-value)	avg. diff. in error(p-value)	avg. diff. in error(p-value)
0.025	$-0.229\,(<10^{-10})^{***}$	$-0.158\,(<10^{-10})^{***}$	$0.026\,(7.06\times10^{-5})^{***}$
0.050	$-0.138\,(<10^{-10})^{***}$	$-0.141\,(8.35\times10^{-10})^{***}$	$0.004\,(0.54)$
0.075	$-0.067\,(<10^{-10})^{***}$	$-0.087\,(1.81\times10^{-10})^{***}$	$0.007\,(0.09)^{+}$
0.100	$-0.028\,(0.004)^{**}$	$-0.080\,(2.92\times10^{-10})^{***}$	$0.011\,(0.002)^{**}$
0.125	$-0.024\,(10^{-4})^{***}$	$-0.076\,(<10^{-10})^{***}$	$0.007\,(6.78\times10^{-5})^{***}$
0.150	$-0.004\,(0.489)$	$-0.052\,(7.2\times10^{-9})^{***}$	$0.002\,(0.44)$
0.175	$-0.006\,(0.199)$	$-0.052\,(6.42\times10^{-10})^{***}$	$-0.007\,(0.013)$
0.2	$0.004\,(0.209)$	$-0.042\,(7.59\times10^{-8})^{***}$	$-0.011\,(1.415\times10^{-6})^{***}$
0.3	$0.010\,(1.04\times10^{-5})^{***}$	$-0.039\,(2.83\times10^{-6})^{***}$	$-0.020\,(<10^{-10})^{***}$
0.4	$0.020\,(1.04\times10^{-5})^{***}$	$-0.017\,(0.0039)^{**}$	$-0.026\,(<10^{-10})^{***}$
0.5	$0.020\,(1.78\times10^{-8})^{***}$	$-0.014\,(0.0068)^{**}$	$-0.030\,(<10^{-10})^{***}$
0.6	$0.028\,(<10^{-10})^{***}$	$-0.01\,(0.012)^{*}$	$-0.035\,(<10^{-10})^{***}$
0.7	$0.027\,(<10^{-10})^{***}$	$-0.018\,(4.95\times10^{-5})^{***}$	$-0.037\,(<10^{-10})^{***}$
0.8	$0.036\,(<10^{-10})^{***}$	$-0.0014\,(0.73)$	$-0.040\,(<10^{-10})^{***}$
0.9	$0.029\,(1.32\times10^{-9})^{***}$	$-0.0166\,(0.0018)^{***}$	$-0.041(<10^{-10})^{***}$

Table 4. Pairwise $t-$test between different models.
H_0: error by first method = error by second method
avg. diff. in error is the average of (error by first method − error by second method), hence, a negative value means the first method does better and a positive value means the opposite
$^{+} \Longrightarrow p<0.1, {}^{*} \Longrightarrow p<0.05, {}^{**} \Longrightarrow p<0.01, {}^{***} \Longrightarrow p<0.001$

5.1.2 Accuracy in retrieving top items

Three methods were trained as described in the previous section at the settings described therein. Then the expected ratings for each movie in the test set was computed. This creates an ordering over the test item set for each user. A recommender system would recommend items from this list in the decreasing order of expected rating. Our goal is to find out how the precision of the recommendation is affected as we include more and more items from this list in pursuit of retrieving all the relevant items. In this set of experiments we treat movies with rating A in the test set to be relevant. The precision vs. recall curve is given in the Figure 11 and Figure 12. A separate experiment that treats movies with rating B or higher in the test set to be relevant returns similar results albeit with higher precision values due to the presence of more relevant items.

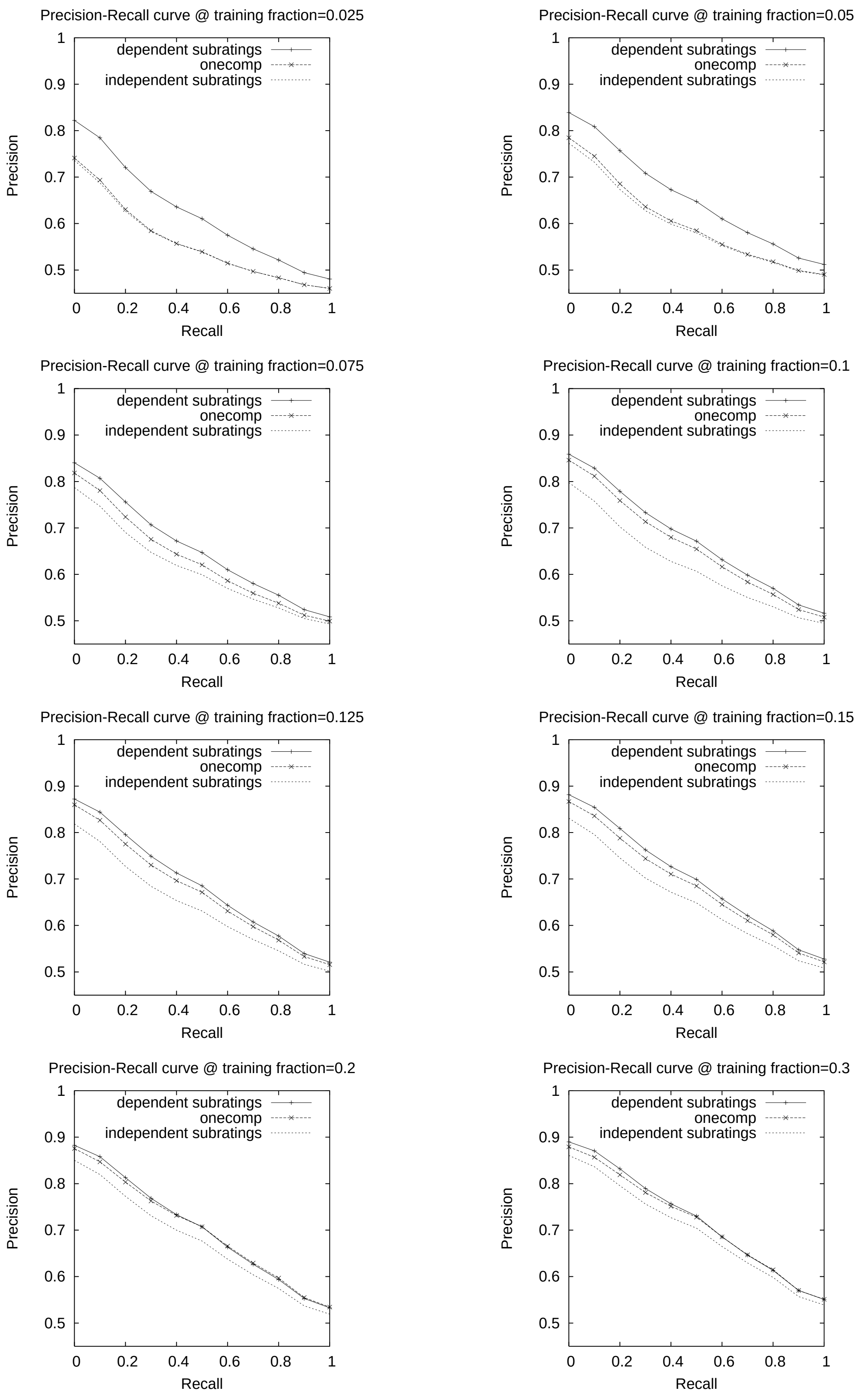

Figure 11. Precision-Recall curve for Three methods

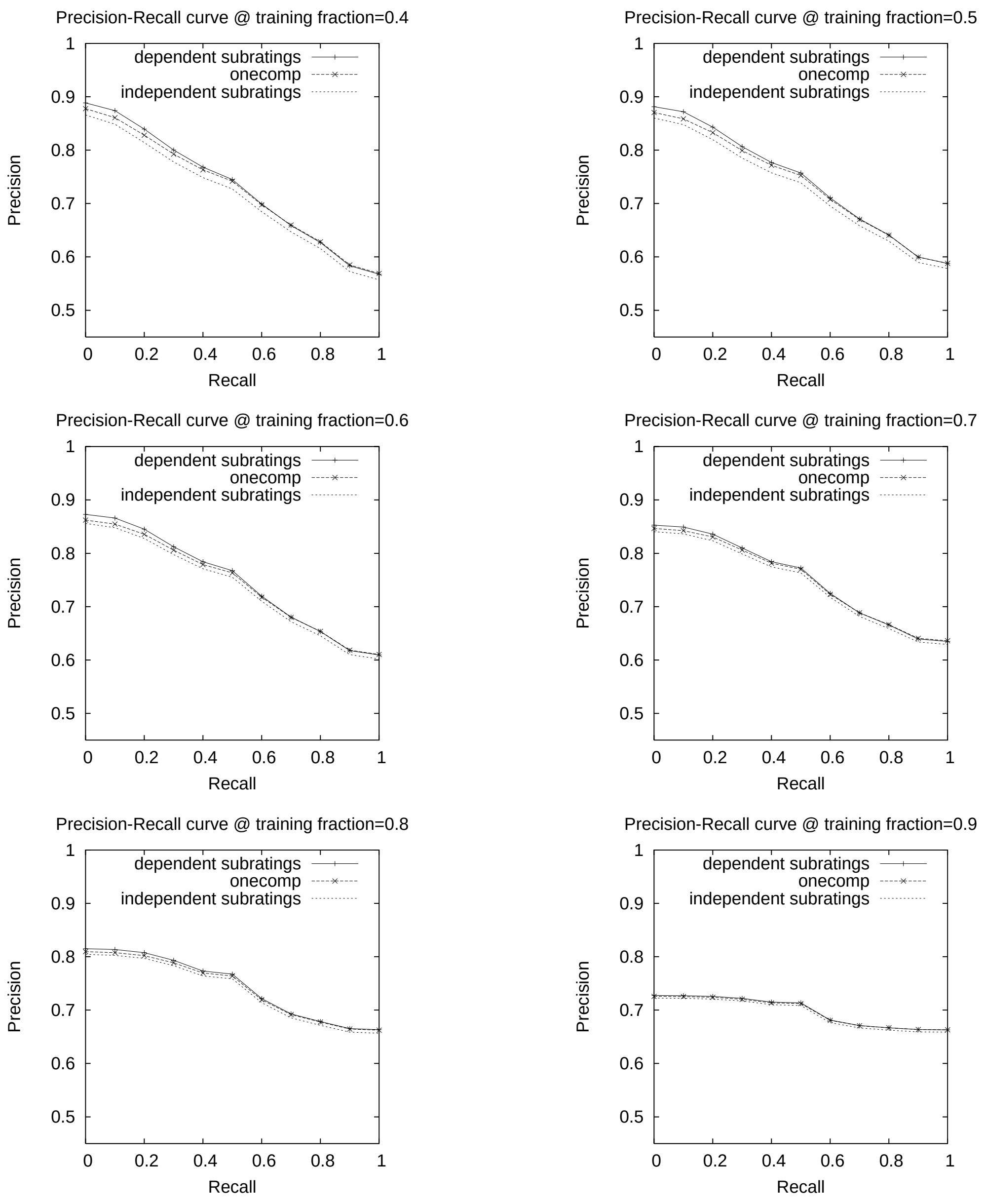

Figure 12. Precision-Recall curve for Three methods

As we can see when the training fraction is low the difference between the the three algorithms is the most pronounced. The algorithm with discovered structure gives the highest precision at each recall level followed by the method using only the Overall component. The algorithm with independence assumption among the component ratings returns lowest precision. However, as we use more and more training data the difference between these algorithms diminishes. The interesting point to note here is that although when using only overall as we use more and more training data we get a lower Mean Absolute Error than using all components, it does not perform better in selecting top-N items. As pointed out in [18] these metrics measure two different aspects of the performance of the algorithms and are often not correlated. One must use the appropriate evaluation metric to measure the suitability of an algorithm for the task at hand.

5.2 Experiments with time ordered data

In the previous section I used randomly selected subset of ratings for training and the remaining for testing. But, we don't get such data in real life. We get then in time order and our only option is to use the past ratings to predict future ratings. To complicate the matters, raters *might* be less consistent in their ratings when they start rating than they are after a while of rating movies. So, using *past* ratings to predict *future* ratings may be harder than using random sample of ratings to predict remaining ratings.

To simulate such a scenario, after sorting the ratings by time stamp, I trained the algorithms using first 10%, 20% ... of the data and tested them using the remaining ratings. Upon comparing results with training data randomly selected with the results with training data selected by time order we see that the random selection leads to lower MAE—as hypothesized.

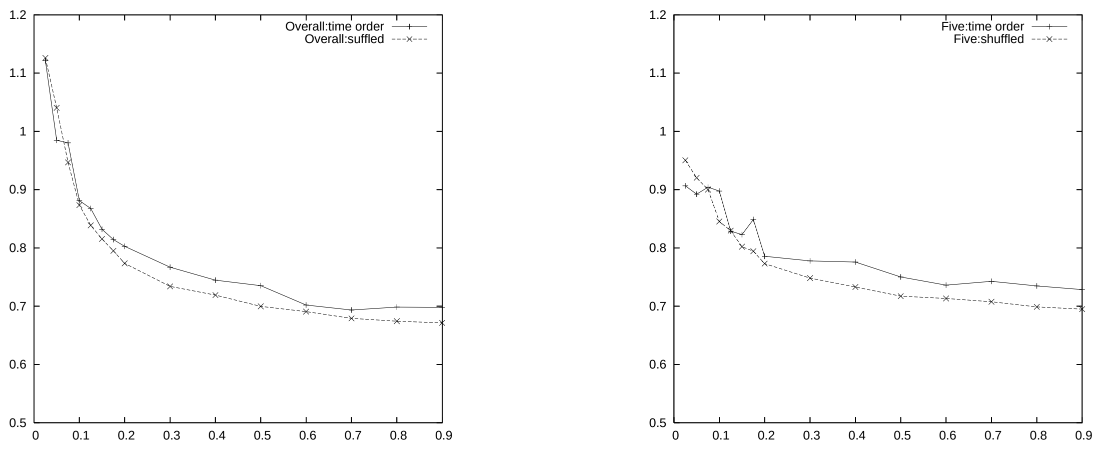

Figure 13. MAE comparison using randomly selected training data, vs., using training data as they are received.

The earlier behavior of the model with dependent structure is still present, i.e., for small amount of training data using all five components leads to lower Mean Absolute Error (MAE). But, the results are not consistent. When doing random train-test split, I am repeating the exercise 30 times and plotting the averages, so, the plots are more smooth (monotonic).

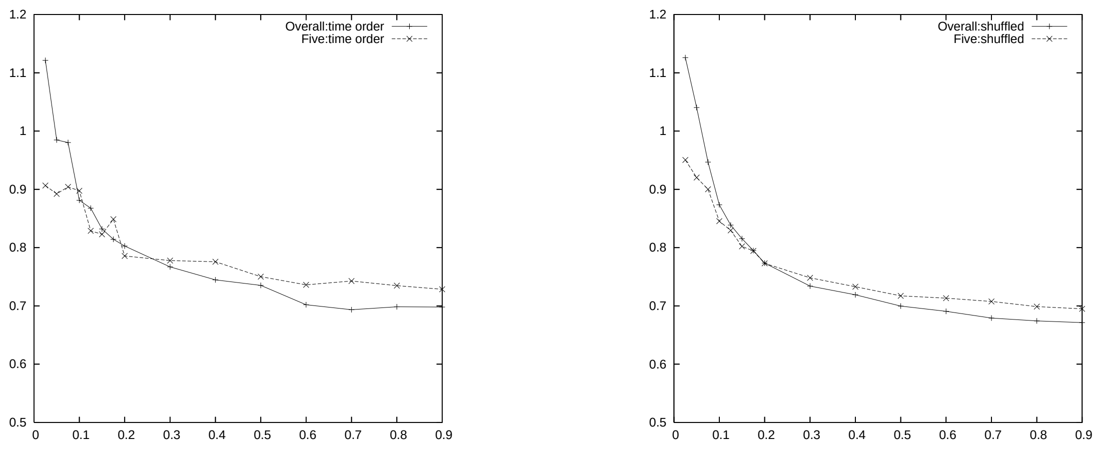

Figure 14. MAE when training data is selected in time order
Overall: is the method using only Overall rating
Five: is the method using all five rating components

Using same number of training ratings for each user

In the above example each user contributes different number of points to the training set. So, it is hard answer the question: until how many training points is there a benefit from using five components and after what point it is good to use only Overall?

To find that out I used only the first 20 ratings for each user. There were 22920 records, 2331 movies and 1145 users. I repeated the training and testing exercise by using 1, 2, ..., 18 ratings from each user for training.

We observe that it's easier to predict using random sampled training data.

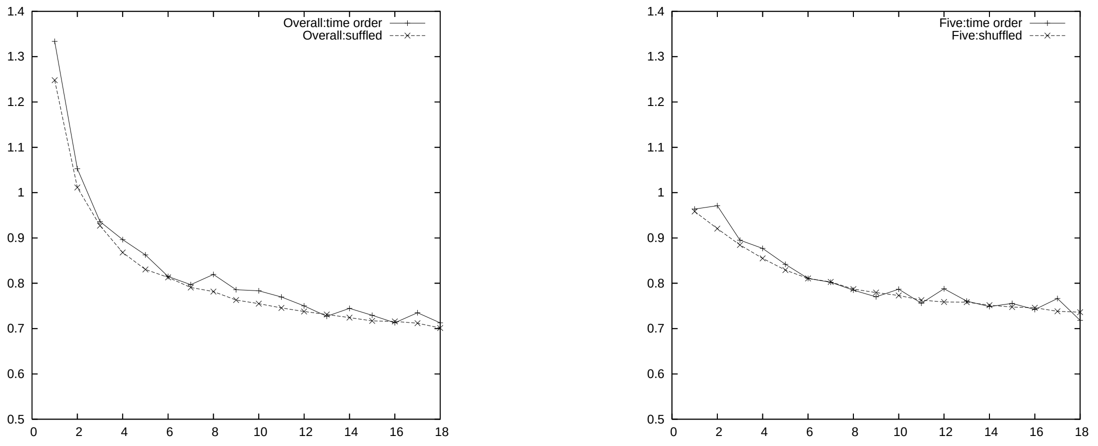

Figure 15. MAE comparison using randomly selected training data, vs., using training data as they are received. *Using only first 20 ratings for each user*

Comparing the MAE plots of predicting using only overall rating and using Five components, we see that using five components has an advantage when we have less training data: both with randomly sampled training data and with training data used in time order. We can also say as long as we have up to five training rating per user, using five components seems to have an advantage. But, when we have more than 5 training points using only Overall seems to have an advantage.

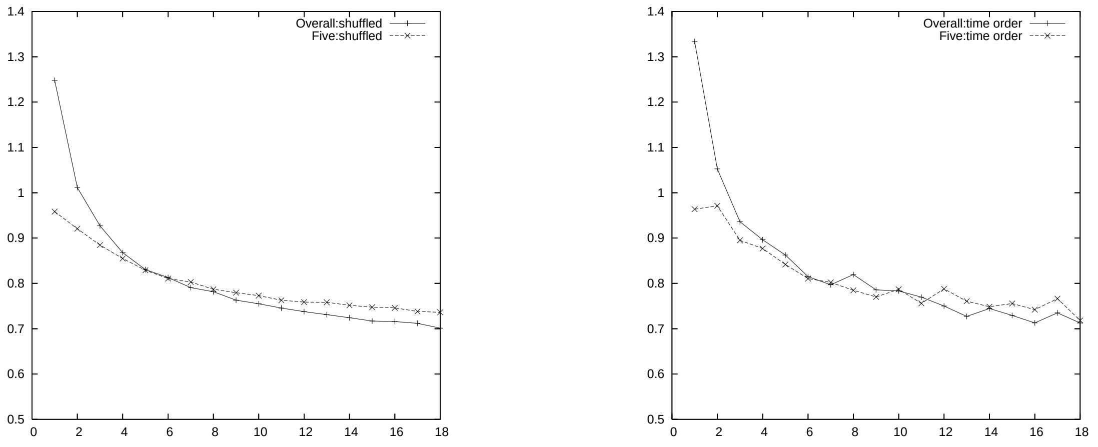

Figure 16. Comparison of using only Overall vs. Five components: with random training data and with training data selected in time order.

One of the concerns with the observations in the previous section where we used random training sample was that results obtained might not be meaningful because in real life training data is collected in time order. Such data might have different characteristic. For example, people are usually inconsistent in their ratings in the early stages of their "rating career", but, after rating for some time they become more consistent in their rating.

In this set of experiments, we use past training data to predict future ratings and found that our earlier observation is still valid. Using five components seem to have an advantage when when there is little training data available and using only overall component seems to have an advantage when there is enough training data.

If the hypothesis that people are inconsistent when they begin to rate was true, then we should expect using such data for training to lead to lower accuracy than using random ratings for training. We observe this in our experiments. This suggests that such might be true about raters' behaviour.

5.3 Exploring learning effect with time order data

Learning theory says that as people repeat certain job they become proficient at it and after an initial period there is no noticeable improvement. Ritter and Schooler have shown that people take less and less time as they repeat a task [37]. They have shown that the time follows a power law while it gets smaller and smaller. The objective in our this set of experiments is can we say something similar in our movie rating scenario where people become consistent as they rate.

Experiments

In this case it is harder than measuring time to perform a task. We can't measure people's consistency in rating—unless we do some controlled experiment. So, the prospect of giving a functional form to increasing consistency seems grim. Therefore, I have tried to measure the effect of people getting consistent indirectly.

To see the effect of dropping raters' first few ratings, when people are likely to be inconsistent, I constructed a data set consisting of first 20 ratings of each user (In a later repetition I have used first 40 ratings of all users with 40 or more ratings.) Then I dropped first one rating, first two ratings ... up to first ten ratings, trained on 5 of the remaining ratings and then tested on the remaining ratings. While selecting 5 training ratings, I have selected them in time order and at random in two separate experiments. The plots of MAE are given in Figure 17 and Figure 18. The Blue line is the MAE using Five ratings components and the Purple line is the MAE using only Overall rating.

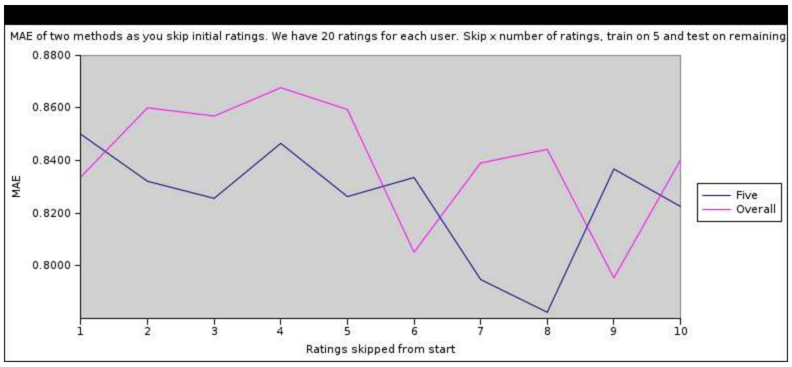

Figure 17. Using data in time order

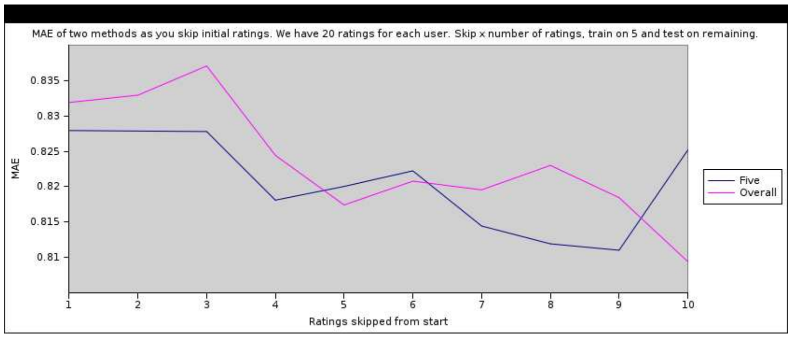

Figure 18. Using data in random order. MAEs are averaged over 30 runs.

There seems to be a slight downward trend in these curves, more clear with random training data than with time ordered training data, indicating that errors go down as we throw away initial ratings. But, it is hard to see the number of ratings after which error stabilizes.

Experiments dropping even more ratings

Results of another set of experiments using first 40 ratings of each user, skipping first one, first two, ... up to first twenty ratings, training on 10 and testing on remaining is shown in Figure 19 and Figure 20.

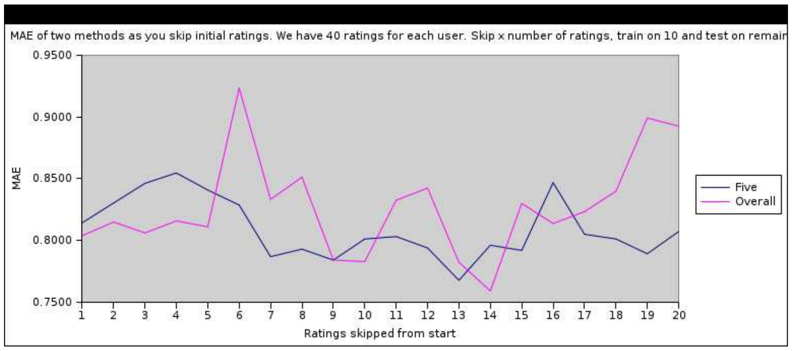

Figure 19. Using data in time order

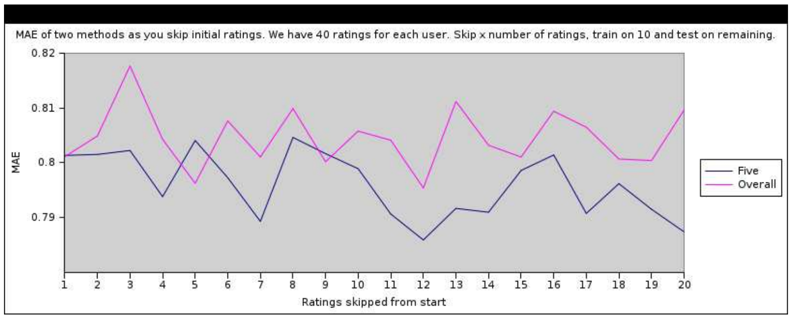

Figure 20. Using data in random order. MAEs are averaged over 30 runs.

It is hard to see a number of ratings up to which we can skip before training to make the error stabilize. Also, the power law in decreasing time to perform a task that Ritter and Schooler have found is missing in these set of results.

6 Halo in multi-criteria movie rating

6.1 Halo Effect

The phenomenon of observing a higher than expected correlation between ratings collected from human subjects is known as the Halo effect. It was first identified by Wells as a constant error in rating where raters rate subjects for the general merits at the time of rating them for their individual qualities [44]. Wells has indicated that this constant error is probably not a serious concern and it is difficult to see how it could have been avoided (p21). After about a hundred years of research we still do not have an agreed upon way to prevent, measure or correct halo effect. And there is disagreement in the research community whether halo is a completely harmful phenomenon ([8], [14]).

Thorndike was the first to term this error as Halo error[43]. The paper makes many interesting correlation observations. Correlation between rating of general ability and technical ability of aviation officers was found to be 0.67 where as author states that the true correlation could not be more than 0.25 after attenuation correction. Students' ratings of the voice of their teachers was found to be correlated at 0.50 with the teachers' interest in community service and at 0.67 with the intelligence. Thorndike asserts that since, these correlations are much higher than the correlation that can be expected between true scores, this is something added by the raters. Although, this universal observation of very high correlation makes a case for something systematic affecting the correlation between the rating components, the manner in which the problem is highlighted indicates a problem that will resurface again and again: we collect ratings—in most of the cases—when the true scores are impossible to collect, e.g., leadership quality of an officer, lucidity of a teacher's discourse, direction quality of a movie. Therefore, how do we compare the observed rating correlations with a true score correlation and possibly measure the halo effect? Thorndike himself lamented that although this halo effect seems large, we lack an objective criteria by which to determine its exact size.

6.1.1 Sources of Halo

Since, halo is the higher than expected correlation between two rating variables it can be due to two reasons. One reason lies with the raters' behavior, who due to their cognitive distortion add to the correlation of the rating attributes. This is known as *illusory halo*. Cooper (1981) has outlined five related factors that might lead to this behavior[8]:

1. Rater's lack of familiarity with the target might prompt him to rely on his global impression and give component ratings based on how he thinks the categories co-vary,
2. Rater might assume that the rating components co-vary with his global impression or with salient components,
3. Components might not be well defined, which would lead the raters to group together somewhat related evidences to generate ratings,
4. Rater might be unwilling to put in enough effort to distinguish between the components or be sensitive to the fact that he might be committing Halo error,
5. If there is a time gap between the observation and rating collection, then the rater might forget the details and add his bias about how rating components co-vary [40].

The second reason could be in the design of the rating components. They may be truly correlated—even before the rater added their error. This is known as the *true halo*. Traditionally (Thorndike 1920, Wells 1907) when one refers to the halo error, it is understood that they are referring to the *illusory halo*. But, it is important to be aware of the difference between the two since, often they co-occur (with true halo possibly affecting the illusory halo[26]) and the metrices that claim to measure the halo are unable to distinguish between the two and measure a combined effect.

Yet another perspective on Halo is provided by [32]. They have found from several laboratory experiments that the Halo effect is not a property of the rater or ratee, but, a property of the unique rating situation. If this is true, then an appropriate method of measuring halo should measure the halo on for each rating instead of measuring Halo present in a set of ratings.

6.1.2 Measuring Halo

The main methods discussed in literature to detect and/or measure halo are:

1. Noticing the difference between observed correlation and estimated true correlation. This method was used by Thorndike. Although, this is probably the most direct way to access the halo effect, the problem with this method is that true correlations are often not available.

   Note 1. Even when it is possible to compute the correlations from the true scores, the random measurement error attenuates the computed correlation to a value lower than the true correlation. This would inflate the perceived Halo. Hence, we must make correction for this attenuation [13].

2. Computing standard deviation across the rating components. The lower the standard deviation, the higher the halo effect. This method does not work very well when the rating components have naturally different mean, in which case it will show a inter-component standard deviation even when the components are perfectly correlated ([24], [12]).
3. Identifying inter-component factor structure. If there is one dominant factor then it suggests that the ratings are generated from this one factor and the rater does not distinguish between various rating components. This suggests presence of halo [9].
4. Carrying out a rater $\times$ ratee $\times$ category anova. If there is a rater $\times$ ratee effect then halo is said to be present [30].

Method 1, 2 and 3 has been discussed without a clearly laid out criteria for detecting the halo effect. None of the above methods distinguish illusory halo from true halo. There has been limited attempt at examining the true halo and illusory halo by using certain manipulated rating components for which true halo can be computed [8].

6.1.3 Reducing Halo at rating time

It has been observed that increasing the rater-target familiarity reduces the effect of the halo because it gives the rater a larger sample of target attribute or behavior to base the component ratings on and not fallback on his general impression ([28], [17], [29]).

Sometimes halo effect is observed because of the raters making judgment based on factors that are not relevant to the components they are rating. It has been found that by explicitly asking the raters to rate *key irrelevant* factors, i.e., the factors that might influence the component ratings but should not, such effect can be reduced [38].

Shewder and D'Andrade has shown that halo is consistently higher when ratings are collected for older observations[41]. Borgatta et al. has shown that the rating collected during the observation of target is consistently less haloed than the ratings collected after observations [11]. Therefore, another strategy to reduce halo in rating could be to collect the ratings at the time of observation or as soon as possible after the observation.

Training the rater through lectures and discussion groups to reduce halo in their ratings have given mixed results. The only method that has given any consistent gain is the use of workshops to sensitize the raters to the halo error they might commit by giving them feedback on their ratings ([16], [6], [22]).

6.1.4 Correcting Halo after collecting the ratings

It has been found that average over the component ratings obtained from multiple raters has lower halo than the individual rater's ratings [23]. However, this solution is not always feasible due to lack of adequate raters [8]. Moreover, this might lead to more accurate ratings, but, averaging does not help when we are interested in correcting halo occurring for a rater-target pair (e.g. for the purpose of collaborative filtering).

Another method to remove excessive correlation among the components due to the presence of a dominant component is by statistically removing the effect of the component, usually a global rating component [21]. Holzbach observes that it is almost impossible to collect ratings that are free from Halo. However, if we can collect the rating on a global impression component then we might be able to remove the effect of this global component from other components. In a study containing ratings along six job related behaviors and a global rating, he found that if we remove the effect of the global component from the six behavior ratings we can reduce the inter behavior component correlation. To remove the effect of the global component he regresses the six behavior ratings against global component and computes the correlation among the residuals. This is equivalent to computing the partial correlation between the behavior components while holding the global component constant. The component residuals remaining after the regression were less correlated than the components themselves. This by itself may not be a significant result, because, controlling for any variable that is not perfectly uncorrelated with the component ratings would reduce the component correlations when the correlations are all positive: as was the case in Holzbach's work. What is interesting is that this correction leads to a more understandable factor structure among the components instead of a general component dominated one. Holzbach also reanalyzed three of the previously published studies using his proposed statistical correction method and found that he was able to reduce the effect of the halo. Landy et al., used Holzbach's method to remove halo from a rating set of 15 job related behavior and a global rating component and found that median of inter-component correlation reduced from 0.36 to 0.07. Also, the factor analysis results of the ratings changed from a general factor dominated three factor structure to a more interpretable six factor structure[15]. Before Holzbach, Myers had taken a similar approach to reduce halo where he used job levels as control variable to reduce correlation among job dimensions [33].

6.2 Halo in movie rating data

6.2.1 Correlation structure

Looking at the correlation matrix we find that the components are highly correlated.

	s	a	v	d	o
s	1.00	0.79	0.82	0.74	0.87
a	0.79	1.00	0.81	0.73	0.83
v	0.82	0.81	1.00	0.79	0.88
d	0.74	0.73	0.79	1.00	0.80
o	0.87	0.83	0.88	0.80	1.00

Table 5. Correlation among components of rating—suggesting the presence of a Halo effect

This indicates that there probably is a halo effect in the collected ratings. However, following a procedure like Holzbach's where we statistically remove the effect of the Overall component by taking partial correlation we find that the effect of halo is much less.

$r_{R_iR_j.O}$	S	A	V	D
S	1.00	0.25	0.26	0.15
A	0.25	1.00	0.32	0.22
V	0.26	0.32	1.00	0.33
D	0.15	0.22	0.33	1.00

Table 6. Partial correlation given Overall. $R_i, R_j \in \{S, A, V, D\}$

The average inter-component correlation among variables S, A, V, D has reduced from 0.78 to 0.26. As all correlations are positive we should expect some reduction in correlation when computing partial correlations. However, the average partial correlation among the variables is the least when we control for the variable O among the possible five variables. The average partial correlations when we controlled for S,A,V,D were 0.47, 0.53, 0.35 and 0.60 respectively. These results confirms, using a much larger dataset, Holzbach's findings that controlling for Overall rating reduces the Halo. It also shows that this reduction is consistently more than the reductions obtained by controlling for variables other than Overall rating.

There are factors other than the Overall impression that might be responsible for dependency among ratings. For instance, perhaps some pairs of components are harder to distinguish between, because of ambiguity in those component definitions. That would lead to correlation among that pair of components (3rd point in Cooper's list and to some extent 4th point too). From the partial correlation matrix it seems that there is some ambiguity between Visuals and Direction quality (0.33 partial correlation), Story and Direction (0.32 partial correlation). Or may be there is some true correlation among these pairs. Cooper's point 1, 2, and 5 supports a "general impression leading to higher correlation between all pairs" theory and his 3rd and 4th reason makes it possible to have higher inter-component correlation between specific pairs of components.

6.2.2 PCA

Another method to detect Halo is to carry out a Principal Component Analysis of the correlation matrix and look for the presence of a dominant component. If we take a linear combination of the five variables using weights given by eigen vectors of the correlation matrix or covariance matrix to create new variables, then the five new variables will have zero correlation between themselves and will have variance equal to the corresponding eigen values. Another important property is that if we order the new variables in the decreasing order of their eigen values, then the first new variable will have the highest possible variance among all variables that one may construct by linear combination of the original variables. The second new variables will have the highest variance among all possible variables that we may construct by linearly combining the original variables while keeping it uncorrelated to the first constructed variable. And similarly for the remaining new variables. These variances are same as the eigen values and the sum of these variances is exactly equal to the sum of the variances of the original variables. So, we can find out how much of the entire variance is explained by these newly constructed variance [31].

The eigenvalues of the correlation matrix (Table 5) are:

Factor	One	Two	Three	Four	Five
Eigen values	4.22	0.28	0.22	0.16	0.11
% variance explained	84.5	5.7	4.4	3.2	2.2

Table 7. Eigen values of the correlation matrix

This suggests that if we construct uncorrelated variables by linear combination of these five variables so that they have maximum variance then we can find one variable that will have 84.5% of the total variance. A second variable can be constructed by linear combination of the five original variables that has 5.7% of the total variance, while being under the constraint that this second variable has zero correlation with the first. Similarly the remaining variables can be constructed. In other words, if we perform a rigid rotation of the axes—they stay perpendicular to each other—of the five dimensional rating space, 84.5% of the variance would lie along one of the new axis, 5.7% of the variance along another and so on [31]. The dominant presence of one component that explains a large amount of variance indicates the presence of a Halo effect among the rating components[21].

However, after partialing out the Overall component, i.e., using the Table 6 we find that the largest component becomes much less dominant—suggesting a reduction in halo.

Factor	One	Two	Three	Four
Eigen values	1.77	0.86	0.73	0.63
% variance explained	44.3	21.6	18.3	15.8

Table 8. Eigen values of the partial correlation matrix

6.2.3 Factors

Yet another way of detecting the presence of a Halo effect to look for the presence of a factor structure that is dominated by one factor [21]. In factor analysis we try to express each observed rating component as a linear combination of some hidden variables and an error term unique to the component. In this analysis several rotations of the factor loading matrices were tried. Quartimax rotation, which tries to reduce the number of factors for each variable, gives the following structure.

	Factor 1	Factor 2	Uniquenesses
S	0.91	0.23	0.11
A	0.87	-0.02	0.21
V	0.93	-0.08	0.10
D	0.84	-0.12	0.25
O	0.95	0.03	0.07

Table 9. Factor loadings after quartimax rotation

This is dominated by one factor, which points to the presence of a halo. It is interesting to note that most of the variation in the Overall component can be explained by these underlying factors (low uniqueness), but, not as much of the variation in other component variables can be explained by these underlying factor. This suggests that these underlying factors are the closest to the Overall rating[2].

6.2.4 Effect of experience in rating

Studies have shown that training the raters to sensitize them to the halo error can reduce the halo error in their ratings[45]. But, it is not clear whether more experience in rating leads to a lower Halo error. To examine this the halo effect in the ratings of people who have rated different amount of movies were measured using the proportion of variance explained by principal components and by the average inter-component correlation.

Users with		fraction of variance					
#of ratings	# of records	explained by components					avg. corr
$\leqslant 5$	346973	0.877	0.042	0.035	0.026	0.02	0.84
$> 5 \& \leqslant 10$	37473	0.862	0.048	0.039	0.029	0.021	0.82
$> 10 \& \leqslant 20$	27378	0.848	0.053	0.042	0.033	0.024	0.80
$> 20 \& \leqslant 40$	18519	0.837	0.057	0.045	0.036	0.026	0.79
> 40	25493	0.855	0.050	0.041	0.031	0.022	0.82

Table 10. Fraction of variance explained by the principal components and average correlation among the components.

The variance explained by the principal components and the average correlation among components for different groups of users are not very different. Therefore, it does not seem like users with more experience make less halo error. One possible explanation could be that traditionally when people rate a lot of subjects they learn more about rating by receiving some kind of feedback. But, in movie rating it is hard to see how the rating behavior would change since the raters don't get *any feedback*. Cooper has indicated that among rater training programs that consists of lectures, group discussion and workshops, only workshops have produced any consistent reduction in halo. He has indicated that the reason might be that the workshops monitor raters and give them corrective feedback when they commit error [8].

6.3 Parallels

It is interesting to compare the approach taken in the psychometric literature ([21], [15] and [33]) with the Chow-Liu tree dependency structure discovered among the movie rating components.

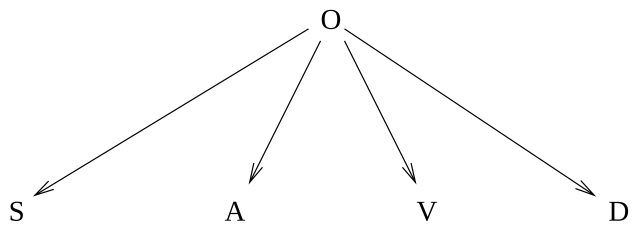

Figure 21. Discovered Chow-Liu tree structure in the component ratings

2. The limitation of existing Chi-square test prevents us from using more than two hidden variables to in our Factor analysis of five variables [31]

In the psychometric literature statistical correction of inter-component correlation has been carried out by computing partial correlation among components while controlling for a global rating variable such as Overall rating. The effect is a reduced correlations among the components.

The dependency structure given in Figure 21 says that if we condition on the Overall rating (O), then the components should be independent of each other. Strictly speaking, this assertion of the Chow-Liu tree is correct only if the assumption that the dependency among the rating components can be described by a tree structure holds. It may not be true in real life. However, a weaker assertion that states that among all possible variables that we might have conditioned on, conditioning on O leads to least residual dependency among the remaining components (as measured by Mutual Information) is still true. It was found that the discovered structure persists over different random subsets of the data, suggesting that the structure discovered is robust.

This result empirically validates the assertion made by [33], [21] and [15]. It is interesting to note that the Chow-Liu tree structure discovery method, which is agnostic to the meaning of the rating components, arrives at a conclusion based on the empirical distribution of the data that agrees with the what researchers in psychometric literature arrived at based on the general impression theory of Halo. I believe this adds to the validity of both the approaches.

7 Conclusion

In this work I have attempted to answer the following research question.

> Can we improve the performance of the collaborative filtering recommender systems by using multiple component ratings?

This work starts by making the observation that the component ratings are correlated. Therefore, a structure discovery exercise was carried out to find the tree structure that captures the most of the pair-wise dependencies among the components of the ratings. The discovered tree structure is interesting in itself. It says that the component ratings provided by the users are most correlated to the Overall ratings than other component ratings. This suggests the possible relation between the Overall impression of the user about the item and the ratings given to the components. A survey of Halo effect often mentioned in psychometric literature is done and parallel between this finding and theories in psychometric literature is drawn.

A mixture model based collaborative filtering algorithm incorporating the discovered dependency structure is presented. This new algorithm is compared with the existing algorithm that uses only one component rating and it was found that we can improve the prediction when there is very little training data available—a common problem faced by the real world collaborative filtering based recommender system. However, as we use more and more ratings from the users for training we found that there is no benefit of using additional components of ratings. In fact, when our objective is to predict the ratings that a user might give to an unseen item, using only the Overall rating leads to a higher accuracy. But, when our objective is to retrieve the highest rated items for an user, using multiple components with the discovered dependency structure among the component ratings leads to higher precision.

In a future work, a class specific structure discovery might be carried out to better capture the dependency among the components. A formal analysis of usefulness of components of rating in predicting ratings on unknown (user, item) pair would certainly be interesting. Also, the proposed multi component model can be used to predict values of the missing component ratings. This would certainly be useful as in the current dataset approximately one third of the records had one or more of the component ratings missing. For the experiments in this work they were discarded.

Bibliography

[1] Gediminas Adomavicius, Ramesh Sankaranarayanan, Shahana Sen, and Alexander Tuzhilin. Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Trans. Inf. Syst.*, 23(1):103–145, 2005.

[2] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. Data Eng*, 17(6):734–749, 2005.

[3] Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.

[4] Marko Balabanovi&#263; and Yoav Shoham. Fab: content-based, collaborative recommendation. *Commun. ACM*, 40(3):66–72, 1997.

[5] Nicholas J. Belkin and W. Bruce Croft. Information filtering and information retrieval: two sides of the same coin? *Commun. ACM*, 35(12):29–38, 1992.

[6] W. C. Borman. Format and training effects on rating accuracy. *Journal of Applied Psychology*, 64:410–421, 1979.

[7] C. K. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968.

[8] William H. Cooper. Ubiquitous halo. *Psychological Bulletin*, 90:218–244, 1981.

[9] S. Zedeck D. Kafry, R. Jacobs. Discriminability in multidimensional performance evaluations. *Applied psychological measurement*, 3:187–192, 1979.

[10] 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*, 39:1–38, 1977.

[11] J. H. Mann E. F. Borgatta, L. S. Cottrell. The spectrum of individual interaction characteristics: An interdimensional analysis. *Psychological Reports*, 4:279–319, 1958.

[12] Neal Schmitt Elaine D. Pulakos and Cheri Ostroff. A warning about the use of a standard deviation across dimensions within ratees to measure halo. *Journal of Applied Psychology*, 1:29–32, 1986.

[13] Charles E. Fisicaro, Sebastiano A.; Lance. Implications of three causal models for the measurement of halo error. *Applied Psychological Measurement*, 14(4), Dec 1990.

[14] Sebastiano A. Fisicaro. A reexamination of the relation between halo error and accuracy. *Journal of Applied Psychology*, 73:239–244, 1988.

[15] J.L. Barnes-Farrell J.W. Steele F.J. Landy, R.J. Vance. Statistical control of halo error in performance ratings. *Journal of applied psychology*, 65:501–506, 1980.

[16] E. D. Pursell G. P. Latham, K. N. Wexley. Training managers to minimize rating errors in observation of behavior. *Journal of Applied Psychology*, 60:550–555, 1980.

[17] H. G. Heneman. Comparision of self and superior ratings of managerial performance. *Journal of Applied Psychology*, 59:638–642, 1974.

[18] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, 2004.

[19] Thomas Hofmann and Jan Puzicha. Latent class models for collaborative filtering. In Dean Thomas, editor, *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99-Vol2)*, pages 688–693, S.F., July 31–August 6 1999. Morgan Kaufmann Publishers.

[20] Thomas Hofmann, Jan Puzicha, and Michael I Jordan. Learning from dyadic data. In Michael S. Kearns, Sara A. Solla, and David Cohen, editors, *Advances in Neural Information Processing Systems 11:Proceedings of the 1998 Conference*, pages 466–472, Cambridge, Massachusetts, 1999. MIT Press.

[21] R. L. Holzbach. Rater bias in performance ratings: Superior, self, and peer ratings. *Journal of Applied Psychology*, 63:579–588, 1978.

[22] J. M. Ivancevich. Longtudinal study of the effects of rater training on psychometric error in ratings. *Journal of Applied Psychology*, 64:502–508, 1979.

[23] D. A. Kenny J. S. Berman. Correlational bias in observer ratings. *Journal of Personality and Social Psychology*, 34:263–273, 1976.

[24] P. Cavenee J. T. Lamiell, M. A. Foss. On the relationship between conceptual schemes of and behavior reports: A closer report. *Journal of Personality*, 48:54–73, 1980.

[25] M. I. Jordan. An introduction to probabilistic graphical models.

[26] Douglas H. Reynolds Kevin R. Murphy. Does true halo affect observed halo? *Journal of Applied Psychology*, 73:235–238, 1988.

[27] Daphne Koller and Nir Friedman. Bayesian networks and beyond. Chapter 13.

[28] B. B. Koltuv. Some characteristics of intrajudge trait intercorrelations. *Psychological Monograph*, 76, 1962.

[29] F. J. Landy and J. L. Farr. Performance rating. *Psychological Bulletin*, 87:72–107, 1980.

[30] L. Wolins M. J. Kavanagh, A. C. MacKinney. Issues oin managerial performance: Multitrait-multimethod analyses of ratings. *Pshychological Bulletin*, 75:34–49, 1971.

[31] Donald F. Morrison. *Multivariate Statistical Methods*. McGraw-Hill Book Company, 1967.

[32] Rebecca L Murphy, Kevin R.; Anhalt. Is halo error a property of the rater, ratees, or the specific behaviors observed? *Journal of Applied Psychology*, 77(4):494–500, August 1992.

[33] James H Myers. Removing halo from job evaluation factor structure. *Journal of Applied Psychology*, 49:217–221, 1965.

[34] Inc. Netflix. Form 10-k annual report pursuant to section 13 or 15(d) of the securities exchange act of 1934. UNITED STATES SECURITIES AND EXCHANGE COMMISSION, Washington, D.C. 20549, 2006.

[35] Michael Pazzani and Daniel Billsus. Learning and revising user profiles: The identification ofinteresting web sites. *Mach. Learn.*, 27(3):313–331, 1997.

[36] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. GroupLens: an open architecture for collaborative filtering of netnews. In *Proceedings of the Conference on Computer-Supported Cooperative Work, CSCW'94*, 1994.

[37] Frank E. Ritter and Lael J. Schooler. *International encyclopedia of the social and behavioral sciences.*, chapter Learning by Occasion Setting, pages 8602–8605. Elsevier, 2001.

[38] W. A. Rizzo and F. D. Frank. Influence of irrelevant cues and alternate forms of graphic rating scales on the halo effect. *Personnel Psychology*, 30:405–417, 1977.

[39] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Application of dimensionality reduction in recommender systems–a case study. 2000.

[40] R. A. Shweder. How relevant is an individual difference in personality. *Journal of Personality*, 43:455–484, 1975.

[41] R. A. Shweder and R. G. D'Andrade. The systematic distortion hypothesis. *New directions for methodology of behavioral science: Fallible judgment in behavioral research*, pages 37–58, 1980.

[42] Luo Si and Rong Jin. Flexible mixture model for collaborative filtering. In *ICML*, pages 704–711. AAAI Press, 2003.

[43] EL Thorndike. A constant error in psychological ratings. *Journal of Applied Psychology*, 4:25–29, 1920.

[44] F. L. Wells. A statistical study of literary merit. *Archives of psychology*, 1, 1907.

[45] Michael-David Kerns William T. Hoyt. Magnitude and moderators of bias in observer ratings a meta-analysis. *Psychological Methods*, 4:403–424, 1999.