Clustering Methods for Collaborative Filtering

Lyle H. Ungar and Dean P. Foster
CIS Dept. and Dept. of Statistics
University of Pennsylvania
Philadelphia, PA 19104

Abstract

Grouping people into clusters based on the items they have purchased allows accurate recommendations of new items for purchase: if you and I have liked many of the same movies, then I will probably enjoy other movies that you like. Recommending items based on similarity of interest (a.k.a. collaborative filtering) is attractive for many domains: books, CDs, movies, etc., but does not always work well. Because data are always sparse – any given person has seen only a small fraction of all movies – much more accurate predictions can be made by grouping people into clusters with similar movies and grouping movies into clusters which tend to be liked by the same people. Finding optimal clusters is tricky because the movie groups should be used to help determine the people groups and visa versa. We present a formal statistical model of collaborative filtering, and compare different algorithms for estimating the model parameters including variations of K-means clustering and Gibbs Sampling. This formal model is easily extended to handle clustering of objects with multiple attributes.

Keywords: collaborative filtering, clustering, EM, Gibbs sampling

Email address of contact author: ungar@cis.upenn.edu

Phone number of contact author: 215 898-7449

1 What is Collaborative Filtering?

On- and off-line shopping centers often keep a record of who purchased what. These records can be used to predict what future shoppers might want to buy. Such predictions are not limited to purchases: One can use records of what movies, CDs or online documents people have enjoyed in the past to predict which ones they will enjoy in the future.

Consider, for example, a list of people and the movies they have liked. (A real list would have tens of thousands of people and equal numbers of movies, but – for obvious reasons – we will use a smaller set for illustration.)

```
Lyle    Andre, Star Wars
Ellen   Andre, Star Wars, Hiver
Fred    Star Wars, Batman
Dean    Star Wars, Batman, Rambo
Jason   Hiver, Whispers
```

If we know that Karen likes $Andre$, what else might she like? $StarWars$, certainly, but that may not be a useful item to recommend since almost everyone likes it. $Hiver$ (short for the French film "Cour en Hiver") would be a good recommendation, since Ellen likes it and Andre. One might even want to recommend $Whispers$ by one level of indirection, since Jason likes $Hiver$ which is liked by Ellen who likes $Andre$. The goal of this paper is to present formal statistical models of such data and methods of estimating the parameters in this data.

Such recommendations of movies, books and CDs based on overlap of interests is often called *collaborative filtering* since, selection of items is done in a method similar to individuals collaborating to make recommendations for their friends. Collaborative filtering methods have been applied to many applications both in research (Goldberg et al., 1992, Sheth and Maes, 1993; Maes and Shardanand, 1995; Konstan et al., 1997) and in industry (see http://www.sims.berkeley.edu/resources/collab/).

Analysis of such purchase or preference data allows, in effect, a detailed market segmentation. Consumers can be grouped down to the level of groups of five or ten – or even individuals, their tastes modeled, and targeted marketing used to make customized sales pitches. Mass emailing suggesting that you buy a CD are annoying spam, but sufficiently targeted announcements of new releases are an information service that you might even pay for.

Real problems are more complex than is suggested by the simple listing of movie preferences presented above. People and movies have attributes. People have an age, a sex, a country of origin. Movies have directors and actors, which are are objects in their own right with their own attribute lists. We wish to provide a method which will allow use of this whole range of information in collaborative filtering.

Current methods for collaborative filtering tend to view records of movies like the one given above as tables of the form

	Andre	Star Wars	Batman	Rambo	Hiver	Whispers
Lyle	y	y				
Ellen	y	y			y	
Fred		y	y			
Dean		y	y	y		
Jason					y	y
Karen	y	?	?	?	?	?

and use K-nearest neighbors (K-NN) methods or K-means clustering of people based on objects purchased.

These methods sometimes work well, but often work poorly, partly because data are often highly sparse (there are many movies which have been rated by few people) and partly because people often have tastes which put them in multiple categories (one may read Science Fiction, Nineteenth century Naval History, and Computer Science books). Because the standard clustering methods are *ad hoc*, improvements are obscure, and one cannot easily take advantage of all available knowledge.

The symmetry of people and movies in Table 2 suggests a new approach: one might group the movies based on who watches them, and use the movie groups to help group people. One speaks of "meaningful movies" appealing to some people, while "action movies" appeal to others. (One friend of mine refers to "girl movies" – ones with no action.) If one can discover such natural groupings of movies, then it should be easier to accurately recommend movies to people.

2 A Statistical model for Collaborative Filtering

We propose the following model of collaborative filtering: People and movies are from classes. For example, movies are *action*, *foreign* or *classic* (with real data, we would use hundreds of classes). People are also from classes: e.g., *intellectual* or *fun*. These classes are unknown, and must be derived as part of the model estimation process.

We will eventually use a range of information to derive these classes, but initially, let us ask how far we can get just using links indicating who liked what.

To see the form of the classes more concretely, rearrange the person x movie table we saw before:

	Batman	Rambo	Andre	Hiver	Whispers	Star Wars
Lyle			y			y
Ellen			y	y		y
Jason				y	y	
Fred	y					y
Dean	y	y				y

There appears to be a group of people *Lyle, Ellen, Jason* who like certain movies *Andre, Hiver, Whispers.* and another group *Fred, Dean* who like other movies *Batman, Rambo.* Almost everyone likes a third group of movies consisting of just *Star Wars.*

For each person/movie class pair, there is a probability that there is a "yes" in the table:

	action	foreign	classic
intellectual	0/6	5/9	2/3
fun	3/4	0/6	2/2

The above insight can be made into a formal generative model of collaborative filtering. It is useful to think first of how the data are generated, and then later of how one might best estimate the parameters in the model. The generative model assures a clean, well-specified model.

We assume the following model:

randomly assign each person to a class k
randomly assign each movie to a class l
for each person/movie pair, assign a link with probability P_{kl}

The model contains three sets of parameters:

P_k = probability a (random) person is in class k
P_l = probability a (random) movie is in class l
P_{kl} = probability a person in class k is linked to a movie in class l

The first two are just the base rates for the classes: what fraction of people are in a given class. The latter, P_{kl} are the numbers estimated in the above table.

3 Model estimation

To estimate the model behind observed data, we need to estimate the class base rates (P_k, P_l) and link probabilities P_{kl} and the assignments of individual people and movies to classes. Estimating the model parameters would be easy if we knew the class assignments. Unfortunately we do not know them.

One obvious way to do this estimation is to use the EM Algorithm or one of its modern generalizations (Dempster et al., 1977; McLachlan and Krishnan, 1997; Neal and Hinton, 1998). In an EM approach to this problem, class assignments are hidden parameters. We then alternate between estimating class assignments and estimating the model parameters. This gives the following two steps:

Expectation (Assignment)
Find the expected class for each person and movie.

Maximization (Model estimation)
Find the most likely P_k, P_l, P_{kl}.
(Just count people and movies in each class
and fraction of "likes" for each subclass pair.)

Surprisingly, the EM cannot be formulated for this problem!

To understand this, we will first review the EM on a standard problem, that of estimating Gaussian Mixtures. We will then show how constraints complicate problem formulation, and finally show what constraints are implicit in the collaborative filtering problem.

The model for Gaussian mixtures is simple. Consider K classes, each generated from a normal distribution with mean μ_k: $x \sim g_k(\mu_k, \sigma^2)$. All have the same variance. The x's are the observed data, and th model parameters μ_k and class labels for x are unknown.

The EM iterates between two steps:

E step

estimate class assignments

$P_{ik} = P(x_i \text{ in } k) \sim e^{-(x_i - \mu_k)^2/2\sigma}$

M step

estimate model parameters

$\mu_k = \frac{\sum_i p_{ik} x_k}{\sum_i p_{ik}}$

It converges remarkably rapidly.

To see how this model can break down, add one constraint to the above problem: let the clusters be constrained to have equal numbers of x's in each. Now, changing any one observation potentially changes the assignment of any other. One cannot move one point x_a from cluster one to cluster two without moving some other point x_b back from cluster two to cluster one. The constraint destroys the separability and tractability of the EM. This case can be handled by dropping the constraint: since the clusters will tend to be close to even in size, the method will move towards the right answer. The situation for collaborative filtering is less auspicious.

In collaborative filtering, an observed event is a person/movie pair. The constraints are that each person is always in the same class and each movie is always in the same class. Dropping these constraints destroys the problem: It loses any connection between individual people and movies. E.g. in *Lyle likes Andre* and *Lyle likes Star Wars*, we would not know the two Lyles are in the same class.

4 Methods

4.1 Repeated clustering

One method of addressing this problem is to cluster people and movies separately, e.g. using K-means clustering or one of its many variants (Aldendefer and Blashfield, 1984; Massart and Kaufman, 1983). K-means clustering closely approximates the EM for a mixture model described above. One can cluster people based on the movies they watched and then cluster movies based on the people that watched them. The people can then be re-clustered based on the number of movies in each movie cluster they watched. Movies can similarly be re-clustered based on the number of people in each person cluster that watched them.

In the results presented below, we use a "soft-clustering" analog of K-means clustering, and assign each person to a class with a degree of membership proportional to the similarity between the person and the mean of the class. (K-means clustering results if one sets the class membership for each movie or person entirely to the class with highest degree of membership.)

On the first pass, people are clustered based on movies and movies based on people. On the second, and subsequent passes, people are clustered based on movie clusters, and movies based on people clusters. In the results presented below, we repeat this clustering three times.

Unfortunately, it is not immediately obvious whether repeated clustering will help or hurt. Clustering on clusters provides generalization beyond individual movies to groups, and thus should help with sparse data, but it also "smears out" data, and thus may over-generalize.

4.2 Gibbs Sampling

One might wish to update one person or movie at a time to avoid constraint violation, but updating one person in EM changes nothing. As described above, it is not possible to reclassify a person in just one person-movie event, since this would lead to a constraint violation. (The person needs to be simultaneously reclassified in all other events in which they occur.) Reclassifying a person or a movie in EM is prohibitively expensive, since no simple sufficient statistics exist.

Gibbs sampling offers a way around this dilemma by sampling from distributions rather than finding the single most likely model. In Gibbs sampling it *is* easy to change the class of a person or movie - and change it simultaneously in all the events in which they occur.

Gibbs sampling is a Bayesian equivalent of EM and, like EM, alternates between two steps:

Assignment

pick a person or movie at random

assign to a class proportionally to probability of the class generating them

Model estimation

pick P_k, P_l, P_{kl} with probability

proportional to likelihood of their generating the data

In order to describe these steps more precisely, we need some nomenclature. Let Y_{ij} be the observed data: $Y_{ij} = 1$ if person i likes movie j and 0 otherwise. Let C_i be the class that person i is in and let C_j be the class that movie j is in. Recall that the model parameters are the base rates for the people and movies, P_k and P_l and the probabilities of a person in class k liking a movie in class l, P_{kl}. Then the probability that person i is in class k (i.e., that $C_i = k$) given the model parameters and all the other class assignments is proportional to

$$P_k \prod_l P_{kl}^{\sum_{j:C_j=l} Y_{ij}} (1 - P_{kl})^{\sum_{j:C_j=l}(1-Y_{ij})}$$

where the sums are over all movies j which are assigned to class l. I.e., the probability that $C_i = k$ is proportional to the base rate of class k times the product over all of the movie classes of the likelihood of the movie being seen and of it not being seen.

In the assignment phase of Gibbs sampling, a person or movie is picked at random, and then assigned to a class with probability proportional to the above expression - or the corresponding one for movies. In the estimation phase of Gibbs sampling, the model parameters (the class membership and link probabilities) are drawn from beta distributions as follows: The link probabilities P_{kl}, are taken to be given by a beta distribution with parameters given by the number of people who did and did not like the movies in the given classes. I.e.,

$$P_{kl} = \beta(X_{kl} + 0.5\delta_{kl}, N_{kl} - X_{kl} + 0.05\delta_{kl}) \tag{1}$$

where
X_{kl} = number of movies in class l liked by people in class k
$N_{kl} - X_{kl}$ = number of movies in class l not liked by people in class k

Note that one could use a Jeffries prior in the above expression by adding 0.5 to all the counts. Instead, we generally use a biased prior to reflect the sparsity of the links.

The class membership probabilities P_k and P_l are taken to be drawn from a multivariate beta distribution with parameters equal to the number of people or movies in each class. Similarly to above, 0.5 is added to the counts to reflect a Jeffries prior. In the actual implementation, it is useful to use the fact that one can generate an observation from the multivariate beta distribution $\beta(a, b, c, ...)$ by generating samples from the gamma distributions $\gamma(a)$, $\gamma(b)$, $\gamma(c)$, ... and then finding $\gamma(a)/(\gamma(a)+\gamma(b)+\gamma(c)+...)$, etc. Then

$$P_k = \gamma(count_k + 0.5) / \sum_k \gamma(count_k + 0.5) \tag{2}$$

where $count_k$ is the number of people in class k. A similar equation applies for the movies.

Gibbs sampling is guaranteed to converge to the true distribution, but need not do so quickly.

5 Results: Synthetic Data

The relative strengths of the different methods depend upon the nature of the data being fit. This is most easily seen on synthetic data, where the correct answer is known. We show here the results for several synthetic data sets. The first is a small data set with two clusters of people and two of movies, 10 people, ten movies, and a link matrix of the form:

$$P_{kl} = \begin{bmatrix} 0.5 & 0.1 \\ 0.2 & 0.3 \end{bmatrix}$$

A typical data set for the small problem is:

1	0	0	0	0	0	0	0	0	0
0	1	0	1	0	0	0	0	1	1
1	0	0	0	1	0	0	0	0	0
1	0	0	0	0	0	1	0	1	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	1	0	0
1	1	0	1	1	0	1	0	1	0
0	1	0	1	1	1	0	0	0	0
0	0	0	0	0	0	0	1	0	1
1	1	0	1	0	0	0	0	1	1

Rows are people, columns are movies, and a 1 indicates that the person liked the movie. Note that the clusters are not obvious. Data sets for the large problem are too large to reproduce, are highly sparse. Clusters are not visually obvious.

Two versions of each problem were run: one in which the class probabilities were all equal (50% chance of being in either class) and one in which the people and movies were each 80% in one class and 20% in the other. Twenty sets of data were generated for case.

A larger data set more typical of real data was also tested. In real data there are typically many clusters and most people only fall into a small number of these clusters. This larger problem had 20 clusters each of people and movies, 100 people and 100 movies, and a tridiagonal link matrix with probabilities of 0.3 on the diagonal and 0.1 on the off-diagonals. Again two subcases were run: one with equal class sizes, and one in which the class sizes were drawn from a uniform distribution.

		Truth	K-means	repeated clustering	Gibbs
Small model - even class sizes					
	Error	0.064	0.143	0.147	0.144
	Likelihood	-64	-70	-68	-60
Small model - 80-20 division					
	Error	0.076	0.152	0.153	0.142
	Likelihood	-55	-64	-61	-54
Large Model - even class sizes					
	Error	0.027	0.068	0.074	0.072

Likelihood	-1209	-1455	-1528	-1224
Large Model - random class sizes				
Error	0.026	0.067	0.072	0.067
Likelihood	-1187	-1473	-1489	-1186

The K-means, the repeated clustering, and the Gibbs sampling methods all give comparable error rates when the classes have equal numbers of members. When the class sizes are not uniform, the statistical model as estimated by the Gibbs sampling, gives superior performance on the smaller problem, but not on the larger one.

The likelihood of the models generated was highest for the Gibbs sampling (sometimes higher even than the likelihood of the true model, indicating that some over-fitting is occuring), and roughly equal for the K-means and the repeated clustering. In some runs K-means was better than repeated clustering, while in other runs it did worse. Interestingly, the general performance patterns of the small and larger data sets are highly similar in spite of their vastly different structures.

These results suggest that the more formal generative model we have presented yields accuracies that are roughly comparable to those produced by the simpler K-means clustering, although the generative model can give superior results when class sizes vary widely. This makes sense, as the K-means clustering has no explicit means of modeling the differences in class size. Estimating class sizes can give a significant, if not extreme, gain in modeling accuracy by accounting for more of the structure of the data. The above results neglect the most important benefit of the more advanced models: their ability to incorporate further information such as attributes of the people and of the movies. The next section examines this in the context of real data.

6 Results: Real Data

Purchase data from CDNow, the world's largest online vendor of compact disks, was examined to better understand the use of clustering items with attributes in collaborative filtering. Using a sample of around 40,000 customers who have purchased three or more CDs, we developed models to predict future purchases. The actual models built were optimized in various ways to maximize rate of response to email solicitations advertising new releases or encourage purchases by customers browsing the CDNow web site (Herz et al., 1998).

We found that clustering people based on CDs works relatively poorly. The data are simply too sparse; not enough people have bought any one CD. One could use the methods described above to cluster the CDs while clustering the people based on the CDs (or, more precisely, the CD clusters) they have purchased. In this case, such clustering is not necessary because further information is available about the CDs which gives excellent clustering.

CDs can be grouped into clusters in which all the music in a given cluster is by a single artist. Clustering people on those CD clusters works extremely well. Quantitative results have not been approved for release yet, but the results for three example tests are shown in Table 1. Table 1a shows the recommendations resulting for a person who has purchased three CDs, one by Miles Davis, one by Dizzy Gillespie, and one by Charlie Parker. The recommended CDs all fit in the same genre of jazz. (The collaborative filtering software, of course, does not have anything labeled "jazz".) Table 1b shows similar results for Country Western artists. Table 1c shows that a particularly poor input, two CDs of somewhat different genres, still gives reasonable results.

This recommendation system was tested on CDNow's customers by sending out email recommendations of new artists. The automated system based on the cluster model resulted in a doubling of the rate of purchase (per solicitation) over the previous manual selections made at CDNow.

```
Input:
Miles Davis Dizzy Gillespie  Charlie Parker

Output:
1.      Chet Baker              In New York
2.      Count Basie             April in Paris
3.      Duke Ellington          At Newport
4.      Bill Evans              You Must Believe in Spring
5.      Maynard Ferguson        American Music Hall
6.      Pat Metheny             Secret Story
7.      Charles Mingus          Mingus Ah Um
8.      Thelonius Monk          Brilliant Corners
```

9.	Sonny Rollins	Saxophone Colossus
10.	John Zorn	Naked City

Table 1a: Sample Recommendations (Jazz Input)

Input: Patsy Cline Loretta Lynn Hank Williams, Jr.

Output:

1.	Johnny Cash	Unchained
2.	Rosanne Cash	10 Song Demo
3.	Merle Haggard	Down Every Road
4.	Alan Jackson	Greatest Hits
5.	George Jones	Best Of
6.	Reba McEntire	Starting Over
7.	Willie Nelson	Greatest Hits
8.	Bonnie Raitt	Road Tested
9.	Leann Rimes	Blue
10.	Travis Tritt	Greatest Hits

Table 1b: Sample Recommendations (Country-Western Input)

Input: Boyz II Men B.B. King

Output:

1.	Babyface	The Day
2.	Ry Cooder	Music By
3.	Robert Cray	Strong Persuader
4.	Aretha Franklin	30 Greatest Hits
5.	Buddy Guy	I Got The Blues
6.	John Lee Hooker	Ultimate Collection
7.	Albert King	Ultimate Collection
8.	Taj Mahal	Phantom Blues
9.	Stevie Ray Vaughn	Live in Austria
10.	Waiting to Exhale	Soundtrack

Table 1c: Sample Recommendations (Combination Contemporary/Blues)

7 Extensions

The statistical method we are proposing has many advantages. It can easily be extended to handle missing data. It can also easily be extended to the case of multiple clusters: e.g. people, movies, directors, and actors. This is particularly important for clustering data from relational databases.

Consider a data base containing people and movies:

person	age	movie
A :	x_1	$x_2 = C$
B :	x_1	$x_2 = D$

movie	director	male lead	female lead
C :	x_3	x_4	
D :	x_3		x_5

One could unfold (extend) the table to include all the attributes of the movies:

A :	x_1	x_2	x_3	x_4	x_5
B :	x_1	x_2	x_3	x_4	x_5

but this is very inefficient. Different objects have different fields, so the extended table may have large numbers of empty fields. Also, the extended table neglects the correlation structure within the objects: it does not know that every instance of Star Wars is directed by George Lucas and stars Harrison Ford.

An alternate approach is to cluster the sub-objects first, as we did for CDNow. This works well on relatively simple problems, but is less effective for more complex domains where people are in many clusters (e.g. people read many kinds of books) and the object attributes do not lead to clean clusters (e.g. the same actor is in both dramas and comedies).

In these cases, a simultaneous statistical model can be superior. The generative model is easily constructed. A simple model might be of the form: (1) randomly assign each person, movie and actor to a class k and (2) assign a link for each person/movie pair with probability P_{kl} and for each movie/actor pair with probability P_{lm}. (Alternatively, one could assign a link for each person/movie/actor triple with probability P_{klm}, but this model will be harder to estimate).

More complex models are easily built. For example, it is frequently the case that their may be more that one person buying CDs or movies on the same account. (E.g., a husband and wife who have different tastes may purchase CDs under the same account name.) In this case, the model can be extended so that each account is a random mixture of people, biased towards small numbers of people per account. The Gibbs sampling presented above is trivially extended to estimate these new models.

8 Summary

We believe that collaborative filtering is well described by a probabilistic model in which people and the items they view or buy are each divided into (unknown) clusters and there are link probabilities between these clusters.

EM is an obvious method for estimating these models, but does not work because it cannot be efficiently constructed to recognize the constraint that a movie liked by two different people must be in the same movie class each time. K-means clustering is fast but *ad hoc*. Repeated clustering using K-means clustering or a "soft clustering" version of K-means may be useful, but usually does not improve accuracy. Clustering movies or people on other relevant attributes can help - and *does* help for the case of CD purchase data. Gibbs sampling works well and has the virtue of being easily extended to much more complex models, but is computationally expensive.

We are currently developing more efficient Gibbs sampling methods for collaborative filtering problems, extending our repeated clustering and Gibbs sampling code to incorporate multiple attributes, and applying them to more real data sets.

Acknowledgments

We thank CDNow and iReactor for providing us with the CD purchase data and discussing its analysis.

References

[1] Aldenderfer, M. and R. Blashfield, *Cluster Analysis*, Sage Publications, 1984.

[2] Casella, G. and E. I. George, Explaining the Gibbs Sampler, *The American Statistician* **46**, 167-174, 1992.

[3] Dempster, A.P., N.M. Laird and D.B. Rubin, Maximum likelihood from incomplete data via the EM algorithm (with discussion), *J. Royal Stat. Soc., Series B* **39**, 1-38, 1977.

[4] Goldberg,D., D. Nichols, B. Oki, and D. Terry, Using Collaborative Filtering to Weave an Information Tapestry, *Communications of the ACM*, **35**(12), 61-70, 1992.

[5] Herz, F., L.H. Ungar and P. Labys, A Collaborative Filtering System for the Analysis of Consumer Data, *unpublished*, 1998.

[6] Konstan J.A., B.N. Miller, D. Maltz et al., GroupLens: Applying collaborative filtering to Usenet news, *Commun ACM*, **40** (3) 77-87, 1997.

[7] Maes, P. and U. Shardanand, Social Information Filtering: Algorithms for Automating 'Word of Mouth', *CHI-95 Conference Proceedings*, Denver CO, May 1995.

[8] McLachlan, G.J and T. Krishnan, *The EM algorithm and Extensions*, Wiley, 1997.

[9] Massart, D. and L. Kaufman, *The Interpretation of Analytical Chemical Data by the Use of Cluster Analysis*, John Wiley & Sons, 1983.

[10] Neal, R.M. and G.E. Hinton, A view of the Em algorithm that justifies incremental, sparse, and other variants, *unpublished*, 1998.

[11] Sheth, B. and P. Maes, Evolving Agents for Personalized Information Filtering, *Proceedings of the Ninth IEEE Conference on Artificial Intelligence for Applications*, 1993.