Recommendation systems: a probabilistic analysis

Ravi Kumar   Prabhakar Raghavan   Sridhar Rajagopalan
Andrew Tomkins

IBM Almaden Research Center
650 Harry Road
San Jose, CA 95120.

Abstract

A recommendation system tracks past actions of a group of users to make recommendations to individual members of the group. The growth of computer-mediated marketing and commerce has led to increased interest in such systems.

We introduce a simple analytical framework for recommendation systems, including a basis for defining the utility of such a system. We perform probabilistic analyses of algorithms within this framework. These analyses yield insights into how much utility can be derived from knowledge of past user actions.

1 Introduction

Collaborative filtering is a process by which information on the preferences and actions of a group of users is tracked by a system (sometimes known as a *recommendation system*) [11, 13, 19, 20, 21, 22, 24, 25]. Based on the patterns it observes, the system tries to make useful recommendations to individual users. For instance, a book recommendation system might recommend Jules Verne to someone interested in Isaac Asimov based on the fact that a number of users have expressed simultaneous interest in both authors. See [22, 24] and references therein for a comprehensive listing of collaborative filtering projects as well as commercial systems.

Most research on recommendation systems has focused on three areas: (i) how to design algorithms that, given the past preferences of users, will make useful recommendations; (ii) how to gather the information on user preferences as conveniently and unobtrusively as possible—this issue runs the gamut from user interface research to marketing science; (iii) privacy issues: how to combine the information gathered from a group of users to the advantage of an individual user, without divulging information about other users.

Our focus is on the first of these areas—the design and analysis of algorithms for collaborative filtering. Although the two latter areas are just as important as the first, their treatment is relatively orthogonal to the problem we consider. To our knowledge, there has been no prior theoretical work on this important emerging application of computing, widely seen as the core of computer-mediated and web-based marketing [4, 10, 14].

There is considerable published work experimentally evaluating the satisfaction of users with recommendation systems, based on user studies. Some papers (e.g., [21]) report cross-validation in which the recommendations of various algorithms are measured by user studies. Hill *et. al.* [13] report a statistical analysis of correlations between recommendations made by a system and users' previously expressed preferences on a validation set (that is not used for "training" the system).

In all prior work we know of, however, filtering algorithms are designed first, followed by *ex post facto* validation to measure user satisfaction. Our intent, on the other hand, is to use a quantitative notion of utility to drive the design of the algorithm, thus enabling us to give provable guarantees on the usefulness of the recommendations it generates. Our approach is predicated on the principle that the success of a recommendation system will be measured by the utility it generates. Our main contributions here are:

1. An analytical framework for evaluating algorithms for collaborative filtering, including a basis for defining utility (Section 2). Our focus is not on so-called *active collaborative filtering*, where users must explicitly and actively annotate or rate the items (books/movies) they encounter; rather, we focus on systems that tacitly observe prior activity to make recommendations.
2. Probabilistic analyses of simple algorithms for collaborative filtering, using these to derive insights on how much prior history is useful and how to exploit it.

2 The model

Our model for recommendation systems consists of three components. The first component is a framework for recommendation systems. The second is a notion of *utility* which defines the objective that the recommendation system is trying to optimize. The final component is a simple probabilistic model of user behavior. We have tried to keep each of these components modular—any of them can be replaced by more sophisticated notions. We feel that our model is simple enough to be tractable and yet offers interesting insights.

We will now describe each component and the particular choices we make in each case.

A framework for recommendation systems. We have a set of m *users* $E = \{e_1, \ldots, e_m\}$. For each user e we have a sample of s_e *items* that e has purchased[1], drawn from a universe of n items. In this paper, we address only the uniform case in which $\forall e, s_e = s$. In the following discussion, we will denote the set of items purchased by user e by e as well. Thus, for all e, $|e| = s$.

The items can be viewed as nodes of a (hyper)graph and the samples corresponding to users can be viewed as (hyper)edges in this graph.

The n items may be thought of as books, movies, web-pages, etc; a recommendation algorithm takes as input the sets of s items for each of the users, and outputs for each user some of the remaining $n - s$ items as a recommendation. In our case, we restrict our attention to algorithms that make exactly one recommendation per user.

To facilitate the notion of what a user prefers, we assume that the n items are partitioned into disjoint *clusters* $\mathcal{C}_1, \ldots, \mathcal{C}_k$. Let $C : [n] \to [k]$ be a function from an item to its corresponding cluster. These clusters may be thought of, for instance, as topics of books (science fiction, travel, etc.). This clustering may or may not be known to the recommendation algorithm; more on this below.

Utility of recommendations. We assume the existence of a utility function $U : [m] \times [n] \to [0, 1]$ giving the utility $U(e, i)$ of recommending an item i to user e.

In this paper, we look at utility functions that are uniform on clusters. Thus, if $C(i) = C(j)$ then $U(e, i) = U(e, j)$ for every e, i, j. Note an implicit simplification here: all items in a cluster have the same utility for a given user (see also Section 2.2). The objective of a recommendation algorithm is to output a recommendation for each user so that the utility of the recommendations, summed over all users, is maximized.

[1]Here "purchase" is a metaphor for *transaction;* it could also represent rentals, browser clicks, etc.

After our simplification, the utility depends only on the cluster that is recommended. Thus, we can think of an algorithm as choosing a cluster rather than choosing a particular item.

Probabilistic user model. For the remainder of this paper, we adopt the following probabilistic model of user behavior.

Each user $e \in E$ is characterized by a k-dimensional vector $p(e) = \langle p_1(e), \ldots, p_k(e) \rangle$, which represents a probability distribution over the clusters. Naturally, $p_i(e) \geq 0$ and $\sum_i p_i(e) = 1$. The interpretation is that the user's sample of s prior purchases is generated by repeating the following procedure s times independently: user e first chooses cluster $\mathcal{C}_i$ with probability $p_i(e)$ and then chooses an item uniformly from $\mathcal{C}_i$. Thus, the sample could contain repetitions. Note that even though the items are partitioned into clusters, there need be no identifiable clusters in the *samples*—users may in fact have no pronounced preferences for clusters. This point is crucial: we do not assume "planted" clusters in the data and seek to find them. Rather, we seek to maximize user utility given whatever patterns exist in the sample data.

Our final simplification relates the user model and the utility function. One could argue that this is indeed the case when the point of view is the one corresponding to the vendor. We assume that $U(e, i)$ is proportional (and w.l.o.g. in our analyses, equal) to $p_i(e)$ for each user i and item e. Thus, the objective of the recommendation system is to generate a recommendation e_i for each user i so that the sum $\sum_e p_i(e)$ is maximized.

Notation. We denote by $\mathcal{B}(k, s, m, p)$ a recommendation problem with m users, s prior samples per user, k clusters of items, and the set of per-user probabilistic preferences p. When it is obvious from the context, we abbreviate $\mathcal{B}(k, s, m, p)$ by $\mathcal{B}(p)$ or even $\mathcal{B}$.

We denote by $\Pi(\mathrm{A}, U, \mathcal{B}(p))$ the expected total utility of algorithm A with utility function U and probabilistic preferences p. The expectation is over samples from p, and any random choices made by A. Since both U and $\mathcal{B}(p)$ depend on p, we may use $\Pi(\mathrm{A}, p)$ as shorthand notation.

Benchmarks. We may compare the expected utility achieved by ALG with that achieved by two benchmarks : (i) a *weak benchmark* who knows C, the partitioning of items into clusters, and (ii) a *strong benchmark*, who knows this partitioning, as well as the precise probability vector $p(e)$ for each user e.

We denote by OPT the utility of the strong benchmark, which is $\sum_e \hat{p}(e)$, where $\hat{p}(e) \stackrel{\text{def}}{=} |p(e)|_\infty = \max_{i=1}^{k}\{p_i(e)\}$. Clearly OPT is an upper bound on the utility of any algorithm.

Let OPT_W denote the utility of the weak benchmark. Unlike the strong benchmark, the utility of the weak benchmark depends in a complicated way on the particular choice of $\mathcal{B}$ (see examples in Section 3). Indeed different choices of $\mathcal{B}$ demand differing methods of using knowledge of C.

Limiting cases. It is instructive to consider two limiting cases. If $m \to \infty$ all edges in the graph occur with large multiplicities, so any meaningful clusters should become apparent. Thus we would have all the information available to the weak benchmark. Additionally, if $s \to \infty$ the algorithm's estimate of the distribution of a particular user becomes almost correct with high probability, we have all the information available to the strong benchmark. We make these statements precise below.

2.1 Related research areas

Our model and approach builds on a number of research areas; we now briefly explain these connections and the ways in which our work differs.

Marketing science is rich in models of consumer behavior and preferences [1, 3, 4, 15], however many of these models do not yet appear to be mathematically tractable in frameworks such as ours.

Our user model is tractable but very simplistic in comparison; but we hope in the future to make the model more realistic.

In computer science, we describe several overlapping categories of related work. The first category consists of data analysis tools such as clustering, data mining [2], latent semantic indexing (LSI) [18], and learning [23]. In each of these cases, the goal is to infer or learn a structure characterizing a given data set. Clustering partitions the data set into groups that are "similar" by some measure; data mining looks for interesting patterns in the data; LSI analyzes spectral properties of the term-document matrix to cluster closely related documents; and learning builds a hypothesis which will perform well when cross-validated against data generated by the true "concept."

Our work differs from each of these in a fundamental way. Our goal is not to identify structures or patterns in the data set, but to exploit these patterns when they exist without necessarily inferring them formally. We are not interested so much in clustering users into sub-populations, as in maximizing utility across the population. For instance, if the user population has relatively weak preferences (say, most users like most clusters roughly equally) then an algorithm can do quite well without first clustering the users. Thus, clustering is valuable only to the extent that it helps in improving the utility. Moreover, it can be argued that in situations where this is the case, the clusters are relatively simple to find. In this sense our problem is somewhat simpler than clustering and graph-partitioning problems. Indeed, our algorithms are considerably simpler than typical clustering algorithms.

A second category of related work includes probabilistic methods such as the work of Boppana [6] and a recent probabilistic analysis of LSI [18]. Our work departs from these in two respects: (i) we seek simple algorithms (no eigenvector computations); (ii) we do not make any assumption of overwhelming preference; indeed, we do not require the users to be drawn from one of a small number of "types", as implicitly needed in [18]. (On the other hand, our algorithm does not achieve the strong document clustering results that [18] establishes for LSI.)

Our work is also related to computational learning theory [23], but has a different emphasis. We are not interested in the sample complexity needed to attain a bounded difference between the underlying distributions and our hypotheses of user preferences. We instead wish to devise algorithms whose utility is competitive against the benchmarks. This problem is in some ways simpler than that of learning the distributions.

The final category includes segmentation [16] problems. This class is perhaps the most closely related because there is an explicit notion of value or utility. The segmentation model described in [16], however, is very general and does not seem to be analyzable in our context. Tractable special cases of the segmentation problem include facility location [9], LSI, and clustering. In each of these cases, the data is embedded in an explicit metric or "similarity" space, which plays a central role in the proposed solutions. The absence of this space is a basic difference between these problems and ours.

The model of Boppana [6] may be viewed as a variant of ours; however, his objective is to find a carefully planted partition of a graph into two clusters; his techniques used are quite different and considerably more sophisticated than the simple algorithms we study.

2.2 Critique and extensions of the model

Our view of each user having a fixed preference for each cluster, and the utility being proportional to this preference is certainly very simplistic. We, however, believe that this is a good first step from which important lessons can be learned, and this should pave the way for further study. Some obvious refinements include: (i) In reality, not all clusters are alike. For instance, the cluster "science fiction" is very different from the cluster "Java": whereas one might purchase a large number of science fiction books, it is unlikely that one would purchase a large number of books on Java. (ii) We have assumed that all the items in a cluster are equally attractive to a buyer; in reality, some items are more popular than others. It is easy to augment our model with a non-uniform distribution within each cluster, but the analysis appears harder. (iii) We seek algorithms that maximize the total utility. Variations—such as maximizing the minimum utility of any user—could model a situation in which we wish to keep all the users happy. (iv) We assume that all users are equally important. In reality, we may give greater weight to frequent purchasers. (v) In our model, user preferences (indicated by prior purchases) are Boolean; more generally, we may model more finely-graded preferences. In particular, one could extend the model to active collaborative filtering where some of the expressed preferences could be negative (meaning, the user did *not* like a particular item). (vi) It would also be interesting to consider time-dependent user preferences, leading to sequential collaborative patterns in which the system tries to infer what each user needs next.

Despite these many possible extensions, we feel that our model is a good start: it is simple enough to be tractable and yet offers interesting insights. At the same time, it is challenging enough that many interesting cases remain open.

The reader may have noticed that our model does not assume prior patterns of preferences (e.g., "scientists tend to like science fiction"). How could we hope for collaborative filtering in the absence of such explicit sub-populations? In fact, our algorithm does make recommendations for each user based on the preferences of similar users, *as evident in the sample data.* Thus, if the sample data indicates strong sub-populations we will in fact exploit them; if no patterns are apparent, even the best algorithm, given the information available to the strong benchmark, will not be able to find much to exploit.

2.3 Main results

Our model for designing and measuring algorithms for collaborative filtering is one of our main contributions. In addition, we have several results that we establish in this model.

In Section 3 we compare the performance of the weak benchmark to that of the strong benchmark. This is useful for two reasons: (i) the weak benchmark represents the limit of what an algorithm can achieve with collaborative filtering alone, when it manages to learn the clustering of items (as evinced in the sample) "as well as possible"; any further improvement must be achieved through a larger sample size (and thus a better understanding of individual users' preferences); and (ii) there are situations when the algorithm may have access to at least some clustering information. We show that when $s = 2$, $\text{OPT}_W/\text{OPT} \geq 2/(\sqrt{k}+1)$. We extend this result to the general case of s samples, giving a tradeoff between the information values of the number of samples and identities of clusters. We also give tight bounds for the special case of two clusters.

In Section 4 we consider recommendation algorithms that, unlike the weak benchmark, do not enjoy knowledge of the clusters. We first give a simple algorithm that is 0.704-competitive with respect to OPT on two clusters. We then consider the asymptotic behavior as $m \to \infty$. We give an intuitive extension of our original algorithm and show that even on an instance with three clusters, it fails to match the performance of the weak benchmark. We then give an algorithm that performs as well as the weak benchmark, despite not knowing the clusters.

3 Using collaborative information

If collaborative filtering yielded perfect information about the clustering, then to what extent could such information be exploited? In our model, the hidden information has two components. The first, captured by C, is "collaborative" information: i.e., information about similarities between elements in the universe. The second, represented by $p(e)$, is meant to model individual behavior. The objective of this section is to address the following question: how much utility can one garner from the collaborative portion of the hidden information, and what are the penalties imposed by the vagaries of individual preferences?

We study two complementary aspects of this issue: (i) the performance of collaborative filtering algorithms as a function of the number of clusters k; and (ii) for a fixed k, the utility gains available from collaborative filtering algorithms as a function of the information about users described by s (the number of samples from each one). Intuitively, as s stays fixed and k grows, the utility of collaborative information should decrease, since we have less information about the user preferences. On the other hand, as s grows and k stays fixed, the utility of collaborative information should increase. We prove two results (Theorems 1 and 6) to formalize these intuitions.

Throughout this section, we only consider algorithms that *know* the underlying clustering of the items; thus, these algorithms have the same information as a weak benchmark. Section 3.1 addresses the performance of collaborative filtering as a function of k, the number of clusters. Section 3.2 then examines the improvements possible as s, the number of samples per user, increases.

The above analyses depend on characterizing the worst case distribution of user preferences (i.e., the set of $p(e)$'s) for given s and k. While such facts are useful in evaluating the benefits of collaborative information, the actual user distributions are not truly adversarial. Therefore in Section 3.3 we provide an analysis that gives tighter performance bounds as a function of simple, and measurable, parameters of the user preference distributions. We are able to complete this analysis only for the simplest case when both k and s are 2.

3.1 The case $s = 2$: the effect of k clusters

Consider the case when $s = 2$. This is the smallest meaningful value of s: if s were 1, no correlation information between items would be available and thus, collaborative filtering would be meaningless. For $s = 2$, the sample corresponding to each user e is an edge in a graph whose nodes are the items. Without risk of confusion, we will use e as well to denote the edge corresponding to user e.

We first consider the case $k = 2$. In this case, the nodes are partitioned into clusters $\mathcal{C}_1$ and $\mathcal{C}_2$. An algorithm for this case must take a sequence of edges, and decide for each edge (user) whether to recommend an item from $\mathcal{C}_1$ or from $\mathcal{C}_2$. Fix some problem $\mathcal{B}(p)$, and assume w.l.o.g. that $\sum_{e\in E} p_1(e) \geq \sum_{e\in E} p_2(e)$. More generally, say $\mathcal{C}_i$ is *heavier* than $\mathcal{C}_j$ if $\sum_e p_i(e) > \sum_e p_j(e)$.

It is straightforward to see that any optimal algorithm must vote for an item in $\mathcal{C}_1$ whenever it sees a $\mathcal{C}_1$-edge. For cross-edges and $\mathcal{C}_2$-edges, the situation is more complicated, as illustrated by the following examples:

Example 1. *If all users have distribution $\langle 3/4, 1/4\rangle$ the correct behavior on cross-edges is to vote for $\mathcal{C}_1$. However, consider the situation in which there are equal numbers of two types of users: one with distribution $\langle 0.99, 0.01\rangle$ and the other with distribution $\langle 0.02, 0.98\rangle$. $\mathcal{C}_1$ is the heavier cluster, but approximately 2/3 of the cross-edges are generated by the second type of user, so the correct behavior on cross-edges is to vote for $\mathcal{C}_2$.*

Example 2. *Similarly, if we again consider the instance in which all users have distribution $\langle 3/4, 1/4\rangle$, the correct decision on $\mathcal{C}_2$-edges is to vote for $\mathcal{C}_1$. However, if the user distribution* *contains an equal number of $\langle 3/4, 1/4\rangle$ users and $\langle 1/3, 2/3\rangle$ users then while $\mathcal{C}_1$ remains the heavier cluster, the correct behavior on $\mathcal{C}_2$-edges is to vote for $\mathcal{C}_2$.*

We now describe an algorithm called VRC(*Vote Randomly on Cross-edges*):

> Given an edge, vote for $\mathcal{C}_i$ if it is a $\mathcal{C}_i$-edge and vote uniformly at random if it is a cross-edge.

VRC will not perform optimally on all the instances above. We will show that for problems that induce the worst-case ratio to OPT, VRC is optimal for such problems. Furthermore, VRC performs worse on these problems than on any other problem, and therefore has optimal worst-case ratio over all problems.

We extend the definition of VRC to general k as follows. Then for an edge, VRC votes for a cluster chosen uniformly at random from the two endpoints of the edge.

Let $\rho(\text{ALG}, p) = \Pi(\text{ALG}, p)/\Pi(\text{OPT}, p)$. We show that VRC achieves the best possible worst-case performance ratio when compared to OPT:

Theorem 1 *If $s = 2$ and $m \to \infty$, for any algorithm* ALG, $\inf_p \rho(\text{ALG}, p) \leq \inf_p \rho(\text{VRC}, p) = 2/(\sqrt{k}+1)$.

Proof: First, recall that $\Pi(\text{OPT}, p) = \sum_{e\in E} \hat{p}(e)$. Second, the utility of VRC is a sum over users. The utility on user e depends on the edge corresponding to e in the sample, and on the distribution $p(e)$. A cross-edge from $\mathcal{C}_i$ to $\mathcal{C}_j$ occurs with probability $2p_i(e)p_j(e)$ and generates utility $(p_i(e)+p_j(e))/2$ (since VRC votes between the two candidate clusters uniformly at random on cross-edges). A $\mathcal{C}_i$-edge occurs with probability $p_i^2(e)$ and generates utility $p_i(e)$. Thus, the utility of VRC can be written:

$$\Pi(\text{VRC}, p) = \sum_{e\in E}\left(\sum_{1\leq i<j\leq k} 2p_i(e)p_j(e)\frac{p_i(e)+p_j(e)}{2} + \sum_{i=1}^{k} p_i^3(e)\right) = \sum_{e\in E}\left(\sum_{i=1}^{k} p_i^2(e)\right)$$

Since the utility of OPT only depends on $\hat{p}(e)$, we can assume without loss of generality that for any worst-case p, each $p(e)$ minimizes $\sum_i p_i^2(e)$ subject to $\hat{p}(e)$ remaining unchanged. For concreteness, let $\ell(e)$ be the "favorite" cluster of user e, so $p_{\ell(e)}(e) = \hat{p}(e)$. Since $\hat{p}^2(e)$ is fixed, we seek to minimize $\sum_{i\neq\ell(e)} p_i^2(e)$ subject to $\sum_{i\neq\ell(e)} p_i(e) = 1 - \hat{p}(e)$. This symmetric and concave function is minimized when $p_i(e) = p_j(e)$ for each $i \neq j \neq \ell(e)$.

Thus, in the worst-case preference distribution, each user is characterized by two quantities, namely $\ell(e)$ and $\hat{p}(e)$. We also know that $\hat{p}(e) \geq 1/k$ and that $p_i(e) = (1-\hat{p}(e))/(k-1)$ for each $i \neq \ell(e)$. To understand the nature of these distributions better, we require two lemmas.

Consider any problem $\mathcal{B}(k, 2, m, p)$. Define the *symmetric closure* $\mathcal{B}^*(k, 2, k!m, p^*)$ of $\mathcal{B}(k, 2, m, p)$ as follows: for each user e in the original problem, replace e with $k!$ users, with distributions $\sigma(p(e))$, for each permutation $\sigma \in S_k$, and let p^* be the resulting distribution function.

Lemma 2 *For any p,* VRC *is optimal for p^*.*

Proof: Let $\text{ALG}(i,j)$ be the cluster voted for by ALG when presented with an edge between $\mathcal{C}_i$ and $\mathcal{C}_j$. The total utility of ALG on such edges in p^* is therefore $\sum_{e\in E}\sum_{\sigma\in S_k} p_{\sigma^{-1}(i)}(e)p_{\sigma^{-1}(j)}(e)p_{\sigma^{-1}(\text{ALG}(i,j))}(e)$. Modulo common multiplicative factors, if $\text{ALG}(i,j) \notin \{i,j\}$ then this latter sum is simply $\sum_{i\neq j\neq \ell} p_i(e)p_j(e)p_\ell(e)$ and if ALG = VRC, then this sum is instead $\sum_{i\neq j} p_i(e)p_j(e)(p_i(e)+p_j(e))/2 = (k-2)\sum_{i\neq j} p_i^2(e)p_j(e)$. By straightforward differential calculus, this sum is always at least as large as $\sum_{i\neq j\neq \ell} p_i(e)p_j(e)p_\ell(e)$. ■

Lemma 3 (Permutation Lemma) *Let* ALG *be any optimal algorithm that knows the clusters for a problem* $\mathcal{B}(k, 2, \infty, p)$. *Then, for any algorithm* ALG$'$, $\Pi(\text{ALG}, p) \geq \Pi(\text{ALG}', p^*)$

Proof: Notice that mean utility over all users for OPT is unchanged from $\mathcal{B}$ to $\mathcal{B}^*$. By assumption $\Pi(\text{ALG}, p) \geq \Pi(\text{VRC}, p)$. And by Lemma 2, $\Pi(\text{VRC}, p^*) \geq \Pi(\text{ALG}', p^*)$. So we must show only that $\Pi(\text{VRC}, p) \geq \Pi(\text{VRC}, p^*)$. Breaking VRC's expected utility into within-cluster edges and cross-edges, we can write $\Pi(\text{VRC}, p) = \sum_{i=1}^{k} p_i^3(e) + \sum_{i \neq j} p_i(e)p_j(e)(p_i(e) + p_j(e))/2$. Clearly the mean utility of VRC on $\mathcal{B}$ and $\mathcal{B}^*$ is identical for within-cluster edges. For cross-edges, the expected utility can be rewritten as $\sum_{i \neq j} p_i^2(e)p_j(e)$. This is clearly identical to the utility on cross-edges in $\mathcal{B}^*$. ■

Let $\mu(y)$ denote the fraction (density) of users e who have $\hat{p}(e) = y$. We have shown that:

Lemma 4 $\rho(\text{VRC}, p)$ *depends only on* $\mu(\hat{p}(e))$ *where* $\hat{p}(e) \in [1/k, 1]$.

We now complete the proof of the theorem. Note that $\rho(\text{VRC}, p)$ can be written as

$$\rho(\text{VRC}, p) = \frac{\int_{\frac{1}{k}}^{1} \mu(x)\text{VRC}(x)\, dx}{\int_{\frac{1}{k}}^{1} \mu(x)x\, dx},$$

where

$$\text{VRC}(x) = x^3 + x(1-x)\left(x + \frac{1-x}{k-1}\right) + \frac{(1-x)^3}{k-1}$$

By componendo-dividendo, the ratio is minimized by concentrating all the density at a particular value of x, namely the one where $\text{VRC}(x)/x$ is minimized in the interval $x \in [1/k, 1]$. Standard differential calculus shows that $\text{VRC}(x)/x$ is minimized at $x = 1/\sqrt{k}$ and that consequently, $\rho(\text{VRC}, p) \geq 2/(\sqrt{k}+1)$.

We also exhibit a distribution p on which $\rho(\text{VRC}, p) = 2/(\sqrt{k}+1)$. Let $q \in [1/k, 1]$, and consider a user $p(e) = \langle q, \overbrace{(1-q)/(k-1), \ldots, (1-q)/(k-1)}^{k-1}\rangle$. Let $\mathcal{B}(p)$ be the problem on k equal-size clusters that is the symmetric closure of e. VRC is optimal for this distribution by Lemma 2. The utility of VRC per user on this distribution is:

$$\Pi(\text{VRC}, p) = q^2 \cdot q + 2q(1-q) \cdot \frac{q + \frac{1-q}{k-1}}{2} + (1-q)^2 \cdot \frac{1-q}{k-1} = \frac{kq^2 - 2q + 1}{k-1}.$$

Clearly, the utility of OPT per user is q. We can therefore choose q to maximize $(kq^2 - 2q+1)/(q(k-1))$, which yields $q = 1/\sqrt{k}$, and gives ratio $2/(\sqrt{k}+1)$. ■

3.2 How many samples do we need?

We now consider values of $s > 2$, and study the behavior of the worst-case performance ratio as s varies for a fixed k. We continue to view m as going to ∞. Intuitively, as s increases, we get a better and better representation of each user's distribution over the k clusters. This intuition is made explicit in the following lemma, which is an immediate consequence of the Chernoff tail bound.

Lemma 5 *Consider any distribution* D *on the domain* $\{1, \cdots, k\}$. *Let* $\vec{X} = \langle X_1 \cdots X_s \rangle$ *be* s *independent samples drawn according to* D. *Let* $f_i(\vec{X}) = |\{j : X_j = i\}|/s$. *Then,*

$$\Pr[\exists i \mid |f_i(\vec{X}) - D(i)| \geq \epsilon] \leq 2^{-\epsilon^2 s/4 + \lg k}$$

Proof:

$$\forall i, \Pr[|f_i(\vec{X}) - D(i)| \geq \epsilon] \leq 2^{-\epsilon^2 s/4}$$

by Chernoff bounds. Now, applying a union bound,

$$\Pr[\exists i \mid |f_i(\vec{X}) - D(i)| \geq \epsilon] \leq k \cdot \Pr[|f_i(\vec{X}) - D(i)| \geq \epsilon],$$

which resolves to the desired inequality. ■

Consider the algorithm MAX:

> Given a sample $\langle X_1, \ldots, X_s \rangle$, vote for the cluster that contains the largest number of elements from the sample.

If more than one cluster is tied for the maximum number of samples, MAX chooses one at random. Notice that when $s = 2$ this specializes to VRC.

We prove the following theorem:

Theorem 6 *For a given k and $\theta < 1$, if $s \geq O(k^2 \lg(k/\theta)/\theta^2)$, then for any user preference p,*

$$\Pi(\text{MAX}, p) \geq (1 - \theta) \cdot \Pi(\text{OPT}, p)$$

Proof: Consider any user e. Let $\hat{p}(e)$ be the largest value in $\{p_i(e)\}$. We say that i is a θ-good cluster for e if $\hat{p}(e) - p_i(e) \leq \theta/(2k)$. Since $\hat{p}(e)$ is at least $1/k$, we know that if MAX chooses a θ good cluster, then the profit of MAX is at least $1 - (\theta/2)$ that of OPT.

Now notice that if MAX does not choose a θ good cluster, there is an i, either the one chosen by OPT or the one chosen by MAX, such that $|f_i(e) - p_i(e)| \geq \theta/(4k)$. By Lemma 5 above, the probability of this event is at most $2^{-(\theta^2 s)/(64k^2) + \lg k}$. Setting $s \geq 256k^2 \lg(k/\theta)/\theta^2$, this probability is smaller than $\theta/2$.

Thus the total loss from not picking θ-good clusters is at most a $\theta/2$ fraction of OPT. This completes the proof. ■

3.3 A tighter analysis of the case $s = 2, k = 2$

This section gives tighter performance bounds for particular classes of preference distributions. Let $s = k = 2$, and let m_1 and m_2 be the first two moments, taken over users, of the probability that a user buys from $\mathcal{C}_1$; thus, $m_x \stackrel{\text{def}}{=} \sum_{e \in E} p_1^x(e)$. We assume m_1 and m_2 are fixed, and determine the worst case distribution, and the corresponding competitive ratio. The performance of VRC can be rewritten as:

$$\Pi(\text{VRC}, p) = \sum_{e \in E} (2p_1^2(e) - 2p_1(e) + 1) = 2(m_2 - m_1) + 1 \qquad (1)$$

In other words, the performance of VRC is completely characterized by the first two moments of the preference distribution. We must also extend the permutation lemma to fixed-moment distributions to obtain the following lemma.

Lemma 7 *For a fixed m_1 and m_2, the user preference distribution p that minimizes $\rho(\text{VRC}, p)$ contains only two distinct values of $p_1(e)$ other than zero and one.*

Proof: For contradiction, assume a user preference distribution p that minimizes $\rho(\textsc{vrc}, p)$ yet contains three distinct values of $p_1(e)$, $0 < v_1 < v_2 < v_3 < 1$. Let α_i be the fraction of users with $p_1(e) = v_i$. Viewed as point masses on the real axis, one can define an operation of *translation*, which is a perturbation of both α_i and v_i. The mean of two point masses is given by $\alpha_i v_i + \alpha_j v_j$ and their second moment by $\alpha_i v_i^2 + \alpha_j v_j^2$. Consider the two types of translations: (i) two sets of point masses translated towards each other such that their mean remains unchanged. This operation decreases the second moment and so long as the translation operation does not cross 1/2 boundary, OPT stays the same. (ii) two sets of point masses on either side of 1/2 translated away from each other such that their mean remains unchanged. This operation increases both the second moment and OPT.

Let v_1, v_2 be such that both are either above or below 1/2, and let v_3 be the third value—if all three fall on the same side of 1/2, let v_3 be either extremal value. Concurrently, translate v_1, v_2 according to (i) and $\{v_1, v_2\}, v_3$ according to (ii). Since these two steps have opposing effects on the second moment, we can adjust the rate of translations so that the second moment stays unchanged. This operation of simultaneous translations can be continued until either v_1 and v_2 coincide (in which case the support is reduced) or v_3 becomes 0/1 (in which case the non-integral support is reduced). In either case, the performance of VRC remains unchanged, whereas OPT increases, as implied by the properties of translations used. ■

We now prove a result about distributions with support 2; subsequently we will fix the case of users at 0 or 1.

Let x_1 and x_2 be the candidate values of $p_1(e)$, and let y_1 and $y_2 = 1 - y_1$ be the fraction of users with $p_1(e) = x_1$ and $p_1(e) = x_2$ respectively. We can now show the following lemma.

Lemma 8 *For* $x_1 \in [0, 1/2]$,

$$\max_p\{\Pi(\textsc{vrc}, p)\} = \max_{x_1, x_2}\{y_1(1 - x_1) + y_2 x_2\} = (1 + \sqrt{1 - 4(m_1 - m_2)}/2).$$

Proof: Incorporating the constraints using Lagrange multipliers, we obtain the condition that $x_1 + x_2 = 1$. Substituting back, we can obtain

$$x_1 = \frac{1 - \sqrt{1 - 4(m_1 - m_2)}}{2},$$

and

$$x_2 = \frac{1 + \sqrt{1 - 4(m_1 - m_2)}}{2},$$

from which the lemma follows. ■

Let $d = m_2 - m_1$. The above lemma gives a bound for support 2 distributions of fixed d. Suppose that for fixed m_1 and m_2 the worst-case distribution given by Lemma 7 contains some users with $p_1(e) \in \{0, 1\}$. The contribution of all such users to d is zero, and the ratio of VRC to OPT is only made worse if these users are removed. Therefore we may assume that the worse-case distribution for a fixed d has support 2. The ratio of the performance of VRC and OPT can then be obtained as

Lemma 9 *If* $d = m_2 - m_1$, *then for all distributions* p *with moments* m_1 *and* m_2,

$$\frac{\Pi(\textsc{vrc}, p)}{\Pi(\textsc{opt}, p)} \geq \frac{2(2d + 1)}{1 + \sqrt{1 + 4d}}.$$

It can be seen that the right-side quantity is at least $2(\sqrt{2} - 1)$, for $d = (1 - \sqrt{2})/2$. The above expression lets us write down the exact ratio for various moments of the preferences; as expected the ratio approaches one for both large and small values of d. Surprisingly, the bound is a function of one variable, rather than a function of both m_1 and m_2.

4 Algorithms

In the previous section, we showed that perfect collaborative filtering allows an algorithm to be competitive with respect to a benchmark who knows each user's distribution. Here we study the complementary question: we give simple algorithms to perform collaborative filtering when the clusters are not known. We continue to focus on the basic case in which $s = 2$. The results in this section require that the clusters have roughly equal sizes.

First, we show that for two clusters a relatively simple algorithm which we call NEIGHBOR compares favorably to OPT, which knows both the clusters and the distribution of each user. We also give results comparing NEIGHBOR to VRC, who knows just the clusters. To summarize, we show that for any distribution p, $\Pi(\text{NEIGHBOR}, p) > .828 \cdot \Pi(\text{VRC}, p)$. From Section 3, $\Pi(\text{VRC}, p) > .828 \cdot \Pi(\text{OPT}, p)$. We also show that $\Pi(\text{NEIGHBOR}, p) > .704 \cdot \Pi(\text{OPT}, p)$. (Note that $.704 > (.828)^2$, thereby showing that NEIGHBOR and VRC achieve their worst cases on different distributions.)

Next, we consider performance as $m \to \infty$. We begin by showing that the natural generalization of NEIGHBOR may perform worse than the weak benchmark when there are as few as 3 clusters. Both NEIGHBOR and its generalization decide which element to vote for based solely on information about edges incident to the edge being considered. We do not know of any algorithm whose performance matches the weak benchmark based only on such local information.

Instead, we present a simple algorithm, CLUSTER, which first uses a global analysis of the graph to determine some approximation of the clusters, and then uses this clustering to determine what to recommend to each user. We show that the ratio of CLUSTER is within $(1 - o(1))$ of the weak benchmark as $m \to \infty$.

4.1 The NEIGHBOR algorithm

The NEIGHBOR algorithm is the following:

> Let G be the graph corresponding to the problem instance. For a user e_i's sample $\{b_{i,1}, b_{i,2}\}$, recommend an item $b_{i,3}$ such that either $\{b_{i,1}, b_{i,3}\} \in G$ or $\{b_{i,2}, b_{i,3}\} \in G$.

Despite its simplicity, the performance of this algorithm is not very far from OPT. We prove the following theorem:

Theorem 10 *For any set of preferences p, $\Pi(\text{NEIGHBOR}, p) \geq 0.704 \cdot \Pi(\text{OPT}, p)$.*

Proof: The proof consists of two steps. First, we prove the theorem for a particular probabilistic distribution $q(\cdot)$ and then show (Lemma 12) that the performance of NEIGHBOR is the least for this $q(\cdot)$.

Consider the following set of probabilistic preferences $q(\cdot)$: for a given $p_1 \in [0.5, 1]$, and $p_2 = 1 - p_1$, there are exactly two classes of users, occurring with equal probability, denoted by their distributions $\langle p_1, p_2 \rangle$ and $\langle p_2, p_1 \rangle$. The following lemma is immediate:

Lemma 11 *Given $e \in \mathcal{C}$, the probability that a random edge adjacent to e is inside $\mathcal{C}$ is $p_1^2 + p_2^2$ and the probability it is a cross-edge is $2p_1p_2$.*

Using this, we can compute the expected utility for NEIGHBOR for this $q(\cdot)$. W.l.o.g. we consider a $\langle p_1, p_2 \rangle$ user. This user may generate three types of edge: (i) a $\mathcal{C}_1$-edge e with probability p_1^2. For this case the neighbor of e is in $\mathcal{C}_1$ with probability $p_1^2 + p_2^2$ which yields a utility of p_1, and is in $\mathcal{C}_2$ with the remaining probability, yielding utility p_2; (ii) a cross-edge with probability $2p_1p_2$, for which the utility is $(p_1 + p_2)/2 = 1/2$; and (iii) a $\mathcal{C}_2$-edge e with probability p_2^2. Here the neighbor of e is in $\mathcal{C}_2$ with probability $p_1^2+p_2^2$ yielding a utility of p_2, and with the remaining probability is in $\mathcal{C}_1$ yielding utility p_1. Summing these, we obtain

$$\Pi(\text{NEIGHBOR}, q) = \sum_e p_1^5(e) + p_2^5(e) + p_1(e)p_2(e) + 3p_1^2(e)p_2^2(e).$$

Using $\Pi(\text{OPT}, q) = \sum_e p_1(e)$, we can show $\Pi(\text{NEIGHBOR}, q) \geq 0.704 \cdot \Pi(\text{OPT}, q)$. ■

We now show that the $q(\cdot)$ considered above is the worst case for NEIGHBOR. More precisely, we can show

Lemma 12 *For any set of preferences p, $\Pi(\text{NEIGHBOR}, p) \geq \Pi(\text{NEIGHBOR}, q)$.*

Proof: Consider the performance of NEIGHBOR on a preference $p(\cdot)$. Let d_i be the probability that a random edge is a $\mathcal{C}_i$-edge, and $d_\times = 1 - d_1 - d_2$ be the probability of a cross edge. Let d denote the edge density. Let $\alpha = 2d_1/(2d_1 + d_\times)$ be the probability that a neighbor of a $\mathcal{C}_1$-edge is in $\mathcal{C}_1$, and likewise $\beta = 2d_2/(2d_2 + d_\times)$ be the probability that a neighbor of a $\mathcal{C}_2$-edge is in $\mathcal{C}_2$. Then we can write $\Pi(\text{NEIGHBOR}, p)$ as

$$\Pi(\text{NEIGHBOR}, p) = \sum_e 2p_1(e)(1-p_1(e)) + (2p_1(e)-1)\left(\alpha p_1(e)^2 - \beta(1-p_1(e))^2\right).$$

Consider also the performance of neighbor on the symmetric closure of $p(\cdot)$, which is $q(\cdot)$. Let α^*, β^* be the analogs of α, β with respect to $q(\cdot)$. Note that $\alpha^* = \beta^*$ by the symmetry of $q(\cdot)$. Using this, we can write

$$\Pi(\text{NEIGHBOR}, q) = \sum_e \alpha^*(2p_1(e)-1) + 2p_1(e)(1-p_1(e)).$$

We now show that $\Pi(\text{NEIGHBOR}, p) \geq \Pi(\text{NEIGHBOR}, q)$. Combining the two previous equations, this leads to the following inequality:

$$\sum_e (2p_1(e)-1)\alpha^* < \sum_e \alpha p_1(e)^2 - \beta(1-p_1(e))^2.$$

Converting this expression to central moments, taking μ to be the mean (over e) of $p_1(e)$, and σ^2 to be the variance of the same random variable, we get:

$$(2\mu-1)\alpha^* \leq (\sigma^2+\mu^2)(\alpha-\beta) + (2\mu-1)\beta.$$

Following the derivation of α and β above, we can similarly derive the value of α^* using an in-cluster density of $(d_1+d_2)/2$. Converting the resulting expression to central moments allows us to derive the following useful equality: $\alpha^* = \mu\alpha + (1-\mu)\beta$.

Using this substitution, and assuming $\alpha \neq \beta$ (the lemma follows otherwise), the above inequality becomes:

$$\mu(\mu-1) \leq \sigma^2,$$

which is always true since $\mu \leq 1$. ■

Combining the performance of NEIGHBOR and VRC, we get the following corollary, which asserts that even without the knowledge of clusters, NEIGHBOR performs very well when compared to VRC.

Corollary 13 *For all preferences p, $\Pi(\text{NEIGHBOR}, p) \geq 0.828 \cdot \Pi(\text{VRC}, p)$.*

4.2 The VOTING algorithm

The following VOTING algorithm is an intuitive generalization of the NEIGHBOR algorithm:

> Let G be the graph corresponding to the problem instance. For a user e_i's sample $\{b_{i,1}, b_{i,2}\}$, recommend an item $b_{i,3}$ such that $b_{i,3}$ is a neighbor of $b_{i,1}$ and $b_{i,2}$ in G with the maximum multiplicity.

Example 3. *Let $s = 2$, $k = 3$, and define three classes of user. Class 1 of user has $p(e) = \langle 0.5, 0.5, 0\rangle$; class 2 has $p(e) = \langle 0.5, 0, 0.5\rangle$, and class 3 has $p(e) = \langle 0, 0.5, 0.5\rangle$. The n users are broken into the three classes as follows: a .45 fraction belong to class 1, a .45 fraction belong to class 2, and a .1 fraction belong to class 3. We let $m \to \infty$. Note that a constant fraction of all edges will have one endpoint in $\mathcal{C}_2$ and the other in $\mathcal{C}_3$. Let $(u, v), u \in \mathcal{C}_2, v \in \mathcal{C}_3$, be such an edge. Clearly, only users from class 3 could have generated (u, v), so the correct response is either $\mathcal{C}_2$ or $\mathcal{C}_3$.*

Consider the action of VOTING *on (u, v). Fix $x \in \mathcal{C}_1$ and $y \in \mathcal{C}_2$, and we shall compute the number of edges from $\{u, v\}$ to x, and from $\{u, v\}$ to y. We compute the first quantity by observing that only class 1 users create (u, x) edges, and they create $(.45)9m/(4n^2)$ such edges in expectation. Likewise, there are the same number of (v, x) edges from class 2 users, for a total of $0.81m/(4n^2)$. On the other hand, there are $(.45)9m/(4n^2)$ edges from u to y generated by class 1 users, and $(.1)9m/(4n^2)$ edges from each of $\{u, v\}$ to y generated by class 3 users, for a total of $0.65m/(4n^2)$. As m grows, the central limit theorem allows us to conclude that the latter value will be smaller with probability approaching 1.*

Thus, VOTING will vote for $\mathcal{C}_1$, incorrectly, on all such edges. Since these edges occur a constant fraction of the time, we observe the following:

Observation 14 *For $s = 2, k \geq 3, \exists \epsilon > 0$ such that $\lim_{m \to \infty} \Pi(\text{VOTING}) < (1 - \epsilon)\text{OPT}_W$.*

4.3 A clustering algorithm

We begin by showing that a simple algorithm, CLUSTER, achieves a ratio approaching that of the weak benchmark as $m \to \infty$. As the name suggests, the algorithm first finds a clustering of the items based on the samples it is given; it then applies the VRC algorithm to the resulting clustering. To complete the proof, we show that CLUSTER finds a clustering of the items that is adequate for the purposes of achieving a ratio close to that of the weak benchmark, without necessarily finding the underlying clustering that generated the data.

As in Section 3.1, we view the items as the nodes of a graph, each edge of which is a sample of items purchased by a user. Given m such samples, we have a multigraph in which an edge (i, j) occurs with some multiplicity m_{ij}, corresponding to the number of users who purchased i and j. Let $m_{\max}$ be $\max_{i,j} m_{i,j}$. During a run of algorithm CLUSTER, we will work with a temporary graph $\mathcal{G}$ defined on subsets of the users, with simple edges (i.e., no multiplicities). To avoid confusion in the description that follows, we will refer to the edges of $\mathcal{G}$ as $\mathcal{G}$-edges; when we simply say "edges", we will be referring to the original multigraph of samples. Let $\epsilon = 0.01$, and m be sufficiently large that $m^{\epsilon/2} > n^2$.

Algorithm CLUSTER :

Step 1: Generate estimated clusters.
while there is a node not assigned to some cluster:

```
            Let $d_1$ be an integer chosen uniformly from $[m^{1/2+\epsilon}, 2m^{1/2+\epsilon}]$.
            Create a graph $\mathcal{G}$ whose nodes are the unclustered items.
            Add $\mathcal{G}$-edge $(i,j)$ to $\mathcal{G}$ whenever $m_{ij} >= m_{\max} - d_1$.
            Find a maximal clique in $\mathcal{G}$; output this clique as a cluster.
        end while.
        Let $C^{\text{new}}$ be the resulting clustering.
    Step 2: Process clusters.
        Run algorithm VRC (Section 3.1) given the clustering $C^{\text{new}}$.
```

We can now state the main theorem regarding this algorithm. Let $\mathcal{U}_m$ be the set of all user preferences on m users. Let $\rho(\text{ALG}, m) = \inf_{p \in \mathcal{U}_m} \rho(\text{ALG}, p)$. Then,

Theorem 15 *For $s = 2$ and any k,*

$$\lim_{m \to \infty} \frac{\rho(\text{CLUSTER}, m)}{\rho(\text{OPT}_W, m)} = 1$$

Proof: We begin with some simple properties of C^{new}. First, we justify an assumption that all edge multiplicities are close to their expectations.

Lemma 16 *With probability at least $1 - 1/m$, every edge has multiplicity within $2\sqrt{m}\log m$ of its expectation.*

Proof: The expected multiplicity of any edge is $\sum_e \Pr[\text{user } e \text{ generates the edge}]$, which is a sum of Bernoulli random variables. Clearly, the probability is no more than 1, so the variance is no greater than m. Thus, the probability of deviating by $2\sqrt{m}\log m$ is no more than $e^{-2\log m} = 1/m^2$. There are only n^2 edges, so the union bound completes the proof. ■

We now show that no cluster of C^{new} contains only a single node of some original cluster.

Lemma 17 $\Pr[\exists I \in C^{new}, J \in C^{orig}, |I \cap J| = 1] \leq 2\log m/m^{\epsilon/2}$.

Proof: We analyze the random choice of d_1 during Step 1 of CLUSTER. By Lemma 16, the multiplicities of all edges are close to their expectations with probability $1 - 1/m$. In order for one node of I to be included in the clique, and another node *not* to be included, it must be the case that $m_{\max} - d_1$ falls between the multiplicities of some two edges that have the same endpoint-clusters. However, by the procedure for choosing d_1, the probability of this event occurring for any two fixed edges is more than $2\sqrt{m}\log m/m^{1/2+\epsilon} = 2\log m/m^{\epsilon}$. Since there are only $n^2 < m^{\epsilon/2}$ possible edges, with probability at least $1 - 2\log m/m^{\epsilon/2}$ by the union bound d_1 will not fall between the multiplicities of any two edges with the same endpoint-clusters. This in fact proves the stronger statement that no original cluster is split by the clustering procedure. ■

Next, assume that cluster C_x^{new} contains $i, i' \in C_I^{\text{orig}}$ and $j, j' \in C_J^{\text{orig}}$.

Lemma 18 $\sum_e (p_I^2(e) + p_J^2(e)) \leq 3m^{1/2+\epsilon} + \sum_e 2p_I(e)p_J(e)$ *with probability at least* $1 - 1/m$.

Proof: The expected multiplicity of edge (i, i'), is $m \sum_e 2(p_I(e)/(n/k))^2$. Likewise, the expected multiplicity of (j, j') is $m \sum_e 2(p_J(e)/(n/k))^2$. Clearly, either (i, i') or (j, j') will have expected multiplicity at least $mk^2/n^2 \sum_e p_I^2(e) + p_J^2(e)$. However, the expected multiplicity of a cross-edge (i, j) is $m \sum_e 2(p_I(e)/(n/k))(p_J(e)/(n/k)) = mk^2/n^2 \sum_e 2p_I(e)p_J(e)$. By Lemma 16, the bad event that some edge multiplicity does not fall within $2\sqrt{m}\log m$ has been discounted except for an event of probability no more than $1 - 1/m$, so the expectations of these two random variables must lie within $2\sqrt{m}\log m + d_1 < 3m^{1/2+\epsilon}$ of one another. ■

As above, let $i, i', j, j' \in C_x^{\text{new}}$, $C^{\text{orig}}(i) = C^{\text{orig}}(i') = I$, and $C^{\text{orig}}(j) = C^{\text{orig}}(j') = J$.

Lemma 19 $\sum_e |p_I(e) - p_J(e)| < m^{3/4+\epsilon/2}$ *with probability at least* $1 - 1/m$.

Proof: Lemma 18 states that $\sum_e (p_I^2(e) + p_J^2(e)) \leq 3m^{1/2+\epsilon} + \sum_e 2p_I(e)p_J(e)$. Observe that if $p_I(e) = \epsilon_e + p_J(e)$ then the difference between $p_I^2(e) + p_J^2(e)$ and $2p_I(e)p_J(e)$ is exactly ϵ_e^2. Thus, Lemma 18 asserts that $\sum_e \epsilon_e^2 \leq 3m^{1/2+\epsilon}$. Maximizing $\sum_e \epsilon_e$ subject to this constraint gives $\epsilon_e = \sqrt{3}m^{-1/4+\epsilon/2}$, so $\sum_e |p_I(e) - p_J(e)| \leq m\sqrt{3}m^{-1/4+\epsilon/2} = \sqrt{3}m^{3/4+\epsilon/2}$. ■

Let VRC^{orig} and VRC^{new} be the benefit of algorithm VRC given clustering functions C^{orig} and C^{new} respectively. For nodes i and j let $E_{ij} = \{e | e = (i,j)\}$, that is, E_{ij} is the set of users whose sample is edge (i,j). Clearly, VRC^{orig} is simply $\sum_{i,j} |E_{ij}|(p_{C^{\text{orig}}(i)}(e) + p_{C^{\text{orig}}(j)}(e))/2$. Let us analyze VRC^{new}. On any edge of E_{ij}, VRC^{new} will recommend an element of $C^{\text{new}}(i)$ or $C^{\text{new}}(j)$ uniformly at random. By Lemma 17, $C^{\text{new}}(i)$ will contain other nodes of $C^{\text{orig}}(i)$ in addition to node i itself, so the conditions of Lemma 19 hold. Let $B(C^{\text{new}}(i), e)$ be the benefit attained by VRC^{new} on recommending an element of $C^{\text{new}}(i)$ to user e: $B(C^{\text{new}}(i), e) = \sum_{j \in C^{\text{new}}(i)} p_{C^{\text{orig}}(j)}(e)$. Therefore on edges of E_{ij}, VRC^{new} attains benefit $\sum_{e \in E_{ij}} (B(C^{\text{new}}(i), e) + B(C^{\text{new}}(j)))/2$. Applying Lemma 19, we have that $\sum_{e \in E_{ij}} B(C^{\text{new}}(i), e) \geq -\sqrt{3}m^{3/4+\epsilon/2} + \sum_{e \in E_{ij}} p_{C^{\text{orig}}(i)}(e)$. Therefore,

$$\begin{aligned} \text{VRC}^{\text{new}} &= \sum_{e \in E_{ij}} (B(C^{\text{new}}(i), e) + B(C^{\text{new}}(j), e))/2 \\ &\geq -\sqrt{3}m^{3/4+\epsilon/2} + \sum_{e \in E_{ij}} (p_{C^{\text{orig}}(i)}(e) + p_{C^{\text{orig}}(j)}(e))/2 \\ &= \text{VRC}^{\text{orig}} - \sqrt{3}m^{3/4+\epsilon/2}. \end{aligned}$$

The above analysis encounters error conditions with probability at most $3\log m/m^{\epsilon/2}$, so that:

$$\text{VRC}^{\text{new}} \geq (1 - 3\log m/m^{\epsilon/2})(\text{VRC}^{\text{orig}} - \sqrt{3}m^{3/4+\epsilon/2}).$$

Finally, note that the algorithm that recommends uniformly at random will attain revenue m/k on m users, which gives a lower bound on the performance of VRC^{new} since this algorithm can do no worse than a random recommendation. Thus, the revenue of VRC^{new} is $O(m)$, and so $\lim_{m\to\infty} \text{VRC}^{\text{orig}}/\text{VRC}^{\text{new}} = 1$. ■

5 Conclusions

In this paper, we introduce a framework for studying algorithmic issues arising in recommendation systems. We have isolated two modeling issues, namely, a model for user utility and a model for user preferences as central issues within this framework.

We study basic cases arising from a simple probabilistic model for utility and user preferences. We show that these cases provide the following interesting insights: (i) Recommendation systems start being valuable with relatively little data on each user. The value of this data is related to the diversity of the interests of the user population. (ii) Simple algorithms are almost as effective as the best possible in terms of utility.

Several issues remain open, most notably in extending our analyses to the more general models suggested in Section 2.2.

References

[1] R.B. Allen. User models: Theory, method and practice. *International Journal of Man-Machine Studies*, 32:511–543, 1990.

[2] M.J. Berry and G. Linoff. *Data Mining Techniques.* John-Wiley, 1997.

[3] J. Bettman. *An Information Processing Theory of Consumer Choice.* Addison-Wesley Publishing Company, 1979.

[4] R.C. Blattberg, R. Glazer, J.D.C. Little, eds. *The Marketing Information Revolution*, Harvard Business School Press, 1994.

[5] B. Bollobas. *Random Graphs.* Academic Press, NY, 1985.

[6] R. Boppana. Eigenvalues and graph bisection: An average-case analysis, *Proc. IEEE Symp. on Foundations of Computer Science*, 1987.

[7] M. Charikar, S.R. Kumar, P. Raghavan, S. Rajagopalan and A. Tomkins. On targeting Markov segments. *Proc. ACM Symposium on Theory of Computing*, 1999.

[8] S. Deerwester, S. T. Dumais, T.K. Landauer, G.W. Furnas, and R.A. Harshman. Indexing by latent semantic analysis. *Journal of the Society for Information Science*, 41(6):391–407, 1990.

[9] Z. Drezner, Ed. *Facility Location: A Survey of Applications and Methods*, Springer, 1995.

[10] R. Glazer. Marketing in an information-intensive environment: Strategic implications of knowledge as an asset, *Journal of Marketing*, 55:1–19, 1991.

[11] D. Goldberg, D. Nichols, B.M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM,* 35:12, pp. 51–60, 1992.

[12] G. Golub, C.F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, 1989.

[13] W. Hill, L. Stead, M. Rosenstein, G. Furnas. Recommending and evaluating choices in a virtual community of use. *Proceedings of ACM CHI,* pp. 194–201, 1995.

[14] D.L. Hoffman and T.P. Novak. Marketing in hypermedia computer-mediated environments: Conceptual foundations. *Journal of Marketing*, 60:50–68, 1996.

[15] J. Howard. *Consumer Behavior in Marketing Strategy*, Prentice Hall, Englewood Cliffs, NJ, 1989.

[16] J. Kleinberg, C.H. Papadimitriou, P. Raghavan. Segmentation problems. *Proceedings of the ACM Symposium on Theory of Computing*, 1998.

[17] B.N. Miller, J.T. Riedl, J.A. Konstan. Experiences with GroupLens: Making usenet useful again. *Proceedings of the USENIX Conference*, 1997.

[18] C.H. Papadimitriou, P. Raghavan, H. Tamaki and S. Vempala. Latent semantic indexing: A probabilistic analysis. *Proceedings of the ACM Symposium on Principles of Database Systems*, 1998.

[19] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, J. Riedl. *GroupLens: An Open Architecture for Collaborative Filtering of Netnews*, Center for Coordination Science, MIT Sloan School of Management Report WP #3666–94, 1994.

[20] U. Shardanand. *Social Information Filtering for Music Recommendation*, Masters Thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1994.

[21] U. Shardanand and P. Maes. Social information filtering: Algorithms for automating "word of mouth", *Proceedings of the ACM Conference on Human Factors in Computing Systems*, pp. 210–217, May 1995

[22] ACM SIGGROUP resource page on collaborative filtering.
`http://www.acm.org/siggroup/collab.html.`

[23] L.G. Valiant. A theory of the learnable. *CACM* 27(11): 1134–1142, 1984.

[24] H.R. Varian. Resources on collaborative filtering.
`http://www.sims.berkeley.edu/resources/collab/.`

[25] H.R. Varian and P. Resnick, eds. CACM Special issue on recommender systems. CACM 40(3), 1997.