# Recommendation systems: a probabilistic analysis

Ravi Kumar Prabhakar Raghavan Sridhar Rajagopalan Andrew Tomkins

IBM Almaden Research Center
650 Harry Road
San Jose, CA 95120.
{ravi, pragh, sridhar, tomkins}@almaden.ibm.com

## 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 algorithmic methods within this framework. These analyses yield insights into how much utility can be derived from the memory of past actions and on how this memory can be exploited.*

## 1. Introduction

Collaborative filtering (sometimes known as a *recommendation system*) is a process by which information on the preferences and actions of a group of users is tracked by a system which then, based on the patterns it observes, tries to make useful recommendations to individual users [10, 12, 18, 19, 20, 22, 23]. 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 [20, 22] and references therein for a comprehensive listing of collaborative filtering projects as well commercial systems.

Most research on recommendation systems has focused on three areas: (i) *Algorithms:* the design of algorithms that, given the past preferences of users, will make useful recommendations; (ii) *Human factors:* 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 individual users, without divulging "sensitive information" about other users.

Our focus is on the first of these areas — the design and analysis of algorithms for collaborative filtering, and their quantitative evaluation. 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 study of this important and emerging application of computing, widely seen as the core of computer-mediated and web-based marketing [4, 9, 13].

Shardanand and Maes [19] report a cross-validation study, in which the recommendations of various algorithms are measured against real user preferences. Hill *et. al.* [12] report a statistical analysis of correlations between recommendations made by their 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 user satisfaction to drive the design of the algorithm, thus enabling us to give provable guarantees on the usefulness of the recommendations it generates.

Our main contributions 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 rate the items (books/movies) they encounter; rather, we focus on systems that tacitly observe prior activity to before making 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 most efficiently.

## 2. The economic 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 offer 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 of whom we have a sample of $s_e$ items (from a universe of $n$ items) that they have purchased[1] in the past. In this paper, we address only the uniform case, i.e., $s_e = s, \forall e$. 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, webpages, 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.

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 for each user. 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 there are clusters in the items, there need be no clusters in the *instance* — the samples drawn — because users may have no pronounced preferences for clusters. This point is crucial: we do not assume "planted" clusters in the data and seek to find them.

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 $\sum_i p_i(e)$ 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 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 $p$. Since both $U$ and $\mathcal{B}(p)$ depend on $p$, we will use $\Pi(\mathrm{A}, p)$ instead.

**Benchmarks.** We may compare this expected utility achieved by ALG with that achieved by two benchmarks : (i) a *weak benchmark*; intuitively this benchmark 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 $\mathrm{OPT}_W$ 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 the example in Section 3). Indeed different choices of $\mathcal{B}$ demand differing methods of using knowledge of $C$.

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

**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 are apparent. Thus we 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.

### 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, 14], 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 three overlapping categories of related work. The first category consists of data analysis tools such as clustering, data mining [2], latent semantic indexing (LSI) [17], and learning [21]. 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. As the reader will notice, compared to many of the algorithms used in the aforementioned situations, our different goal allows us to use simpler approaches.

The second category includes probabilistic methods such as the work of Boppana [6] and a recent probabilistic analysis of LSI [17]. 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 [17]. (On the other hand, our algorithm does not achieve the strong document clustering results that [17] establishes for LSI.)

The final category includes segmentation [15] problems. This class is perhaps the most closely related because there is an explicit notion of value or utility. The segmentation model described in [15], however, is very general and does not seem to be analyzable in our context. Tractable special cases of the segmentation problem include facility location [8], 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.

### 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: (1) 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. (2) 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. (3) We seek algorithms that maximize the total utility, which is what the enterprise operating the system may wish to do. Variations — such as maximizing the minimum utility of any user — could model a situation in which we wish to keep all the users happy. (4) We assume that all users are equally important. In reality, we may give greater weight to frequent purchasers. (5) 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). (6) 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 give algorithms that are $0.704$-competitive with respect to OPT. Our algorithms are extremely simple. In fact, we show that our algorithms garner information in a truly collaborative way, i.e., as $m$ increases, we show that the performances of our algorithms approach $\text{OPT}_W$.

## 3. Using collaborative information

If collaborative filtering yielded perfect information about the clustering, then to what extent could such information be exploited? To address this question, we study the situation when $m \to \infty$. It is easy to see that under this assumption, fairly simple algorithms can identify the clusters (but for an event of vanishing probability) when such clusters are delineated by the user preferences, and thus important for utility maximization. The issue, then, is how to use this information. We will address three different aspects of this question.

First, 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. Intuitively as $k$ grows and $s$ stays fixed, the value of the collaborative information should decline. Moreover, as $s$ grows and $k$ stays fixed, the marginal benefit of increasing $s$ should decline. We will prove two results (Theorems 1 and 4) to formalize these intuitions.

The above analyses depend on characterizing the worst case distribution of user preferences (i.e., the $p(e)$'s) given $s$ and $k$. While such facts are useful in evaluating the benefits of collaborative information, the actual user distribution is 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 restrict this complete analysis to the simplest case, namely 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 $s = 2$, each user $e$ is an edge in a graph whose nodes are the items. 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 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 $\sum_{e\in E} p_1(e) \geq \sum_{e\in E} p_2(e)$.

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. Furthermore, whenever it sees a cross-edge, it must also vote for $\mathcal{C}_1$ unless $\sum_e p_1(e) = \sum_e p_2(e)$, in which case it may vote for either cluster. In either case, there exists an optimal algorithm that votes for $\mathcal{C}_1$ on cross-edges. Given a $\mathcal{C}_2$-edge, the optimal algorithm depends on the set $\{p(e)\}$.

VIC**:** Algorithm VIC(for *Vote In Cluster*), votes for $\mathcal{C}_1$ when presented with a $\mathcal{C}_1$-edge, or a cross edge. It votes for $\mathcal{C}_2$ when presented with a $\mathcal{C}_2$-edge.

VOC**:** Algorithm VOC(for *Vote Out of Cluster*), votes for $\mathcal{C}_1$ on any input.

Finally, we describe a third algorithm called VRC, which is never better than VIC, and sometimes worse. We will show that for problems that induce the worst-case ratio to OPT, VRC is equivalent to VIC, which 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. It is also much simpler to analyze than VIC.

VRC**:** Algorithm VRC, for *Vote Randomly on Cross-edges*, votes for $\mathcal{C}_i$ when presented with a $\mathcal{C}_i$-edge and votes uniformly at random when presented with a cross-edge.

**Example.** It is illustrative to consider an example in which VOC performs better than VIC. Let $k = 2$ and let

$E$ have two types of users. First, there are a billion users with distribution $\langle 0.9, 0.1\rangle$. And second, there are a thousand users with distribution $\langle 0, 1\rangle$. The expected utility of VOC on edges within cluster 2 will be $(0.1)^2 \times 0.9 \times 1,000,000,000$, while the expected utility of VIC will be $1000 + (0.1)^2 \times 0.1 \times 1,000,000,000$, approximately nine times smaller.

We extend the definitions of VIC and VRC to general $k$ as follows. Say $\mathcal{C}_i$ is *heavier* than $\mathcal{C}_j$ if $\sum_e p_i(e) > \sum_e p_j(e)$. Then for edge $e = (b_i, b_j)$, VIC votes for the heavier of $C(b_i), C(b_j)$ or for either if neither is heavier. (Here $C(b)$ denotes the cluster containing item $b$.) VRC, on the other hand, votes for a cluster chosen uniformly at random from the two endpoints of the edge. VOC generalizes to a family of algorithms for the case $k > 2$.

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 $m \to \infty$,*

1. $$\inf_p \rho(\text{VRC}, p) = \inf_p \left\{ \frac{\Pi(\text{VRC}, p)}{\Pi(\text{OPT}, p)} \right\} = \frac{2}{\sqrt{k}+1}, \text{ and}$$

2. *for any algorithm* ALG,
$$\inf_p \rho(\text{ALG}, p) = \inf_p \left\{ \frac{\Pi(\text{ALG}, p)}{\Pi(\text{OPT}, p)} \right\} \leq \frac{2}{\sqrt{k}+1}.$$

**Note.** This theorem in fact holds with $k$ equal to the number of clusters on which any user has non-zero probability. There are two significant lessons from this: (i) The ratio depends on the spread of interests of the users. (ii) If users have few interests, the ratio is good (e.g., if each user is only interested in 4 clusters — reasonably realistic — then the ratio is $2/3$). This can be extended to the case when most users have most of their probability distribution on $\leq k$ clusters.

*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 $e$ of 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 $\mathcal{B}(2, k, m, p)$. Define the *symmetric closure* $\mathcal{B}^*(2, k, k!m, p^*)$ of $\mathcal{B}(2, k, m, p)$ 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 $p^*$, $\Pi(\text{VIC}, p^*) = \Pi(\text{VRC}, p^*)$ and* VIC *is optimal for $p^*$.*

*Proof:* First, we show that $\Pi(\text{VIC}, p^*) = \Pi(\text{VRC}, p^*)$. Assume VIC votes for $\mathcal{C}_i$ on cross edges between $\mathcal{C}_i$ and $\mathcal{C}_j$. Since the distribution is symmetric, every user who prefers $\mathcal{C}_i$ to $\mathcal{C}_j$ will have a paired user with the opposite preference. The utility of VIC on these two users will be equivalent to the utility of VRC.

Next, we show that VRC is optimal for any symmetric distribution. 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 $\sum_{e \in E} \sum_{\sigma \in S_k} p_{\sigma^{-1}(i)}(e)p_{\sigma^{-1}(j)}(e)p_{\sigma^{-1}(\text{ALG}(i,j))}(e)$. If $\text{ALG}(i, j) \notin \{i, j\}$ then this latter sum is simply $\sum_{i \neq j \neq \ell} p_i(e)p_j(e)p_\ell(e)$. Assume otherwise; say, that 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. □

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

We now complete the proof of the theorem. We have shown that the distribution of $\hat{p}(e)$ is the only interesting issue. Let $\mu(p)$ denote the fraction (density) of users $e$ who have $\hat{p}(e) = p$. Then $\rho(\text{VRC}, p)$ 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 assume that $m \to \infty$. The primary question is: to what extent does a sample size of $s$ over a distribution over $k$ points benefit us (relative to a sample size of 2). A simple extension of VRC would be to pick a random point in $s$ and choose the cluster containing that point. A careful look, however, reveals that this algorithm is equivalent to the $s = 2$ case, and thus does not improve on the competitive ratio of $2/(\sqrt{k}+1)$.

The algorithm that we will study in this section will be called MAX. The algorithm looks at a sample and votes in the cluster that contains the largest number of elements in the sample. If there is more than one cluster that qualifies, MAX chooses one at random from among those that do. Notice that when $s = 2$ this specializes to VRC.

By arguments similar to those in Section 3.1, we can deduce that the worst case distribution can be assumed to be symmetric and that for each $e$, $p_i(e) = (1 - \hat{p}(e))/(k-1)$ whenever $i \neq \ell(e)$.

Let $\mu(x)$ denote the fraction (density) of users who have $\hat{p}(e) = x$. Then, again as before, we argue that

$$\rho(\text{MAX}, p) = \min_{x \in [1/k, 1]} \frac{\text{MAX}(x)}{x}$$

where $\text{MAX}(x)$ denotes the expected utility of MAX when presented with a single user with distribution $p_1 = x$ and $p_i = (1-x)/(k-1) = y$ for each $2 \leq i \leq k$ and such that $x \geq 1/k$.

The performance of MAX depends on the gap $\epsilon = x - y$ between $x$ and $y$. If $\epsilon$ is small relative to $1/k$, say $k\epsilon \leq \theta$, then regardless of the algorithm, $\rho \geq 1 - \theta$. Thus, we only need to look at the case that $k\epsilon > \theta$.

We will consider two types of bad events: first that the number of samples in the first cluster is smaller than $(x - \epsilon/2)s$, and second that the number of samples in any other cluster is larger than $(y + \epsilon/2)s$. In the event that the first cluster is not the one with the largest number of samples, one of the two bad events is guaranteed to have occurred. The probability of any bad event can be bounded by $k \exp(-\epsilon^2 s/(4k))$ using a standard tail bound. The initial factor of $k$ is the union bound over all the bad events. Consequently, $s > O((k/\theta^2)\log(k/\theta))$ is sufficient to obtain $\rho \geq 1 - \theta$; this bound is in fact tight asymptotic in $k$.

**Theorem 4** *For a given $k$, if $s \geq O((k/\theta^2)\log(k/\theta))$, then for any user preference $p$, $\Pi(\text{VRC}, p) \geq (1-\theta) \cdot \Pi(\text{OPT}, p)$.*

### 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 $\mathcal{C}_1$: $m_x \stackrel{\text{def}}{=} \sum_{e \in E} p_1^x(e)$. 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:

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

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, whose proof is omitted.

**Lemma 5** *For fixed $m_1$ and $m_2$, the user preference distribution $p$ that minimizes $\rho(\text{VRC}, p)$ contains only two distinct values of $p_1(e)$.*

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

$$\max_p\{\Pi(\text{VRC}, p)\} = \max_{x_1,x_2}\{y_1(1-x_1) + y_2x_2\} = (1 + \sqrt{1-4(m_1-m_2)}/2).$$

*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. □

Now, using the above observations, the ratio of the performance of VRC and OPT can be obtained as

**Lemma 7** *If $d = m_2 - m_1$, then for all distributions $p$ with moments $m_1$ and $m_2$,*

$$\frac{\Pi(\text{VRC}, p)}{\Pi(\text{OPT}, p)} \geq \frac{2(2d+1)}{1+\sqrt{1+4d}}.$$

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, with $m$ being bounded. We continue to focus on the basic case, namely $k = 2$ and $s = 2$. Furthermore, the results in this section require that the clusters have roughly equal sizes. Removing this assumption seems challenging.

The primary result in this section is that 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 $.704$ is strictly greater than $(.828)^2$, thereby showing that NEIGHBOR and VRC achieve their worst cases on different distributions.)

We also propose a new algorithm, the VOTING algorithm, which performs always at least as well as NEIGHBOR.

## 4.1. The NEIGHBOR algorithm

The NEIGHBOR algorithm is very simple:

> 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 optimal OPT. We prove the following theorem:

**Theorem 8** *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 10) 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 9** *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 that

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

Using the fact $\Pi(\text{OPT}, q) = \sum_e p_1(e)$, 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 10** *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 $\alpha^*, \beta^*$ be the analogs of $\alpha, \beta$ 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 $\mu$ to be the mean (over $e$) of $p_1(e)$ , and $\sigma^2$ 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 $\alpha$ and $\beta$ above, we can similarly derive the value of $\alpha^*$ 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 11** *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 a 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.

As in the proof of Theorem 8, we assume that the set of probabilistic preferences is $q(\cdot)$. It is easy to show that $q$ leads to the worst-case scenario for VOTING.

Let $p_1' = p_1^2 + p_2^2, p_2' = 2p_1p_2$. Then, note that for $\mathcal{C}_1$-edge $e$, the probabilities that $e$ has a neighbor of multiplicity $r$ are respectively

$$\frac{p_1'^r}{p_1'^r + p_2'^r} \quad \text{and} \quad \frac{p_2'^r}{p_1'^r + p_2'^r}.$$

Let $\text{VOTING}_r$ be an algorithm that votes for a neighbor of multiplicity $r$. Now, we can do an analysis as before and obtain the utility of $\text{VOTING}_r$ to be:

$$\Pi(\text{VOTING}_r, p) = (p_1^3 + p_2^3)\frac{p_1'^r}{p_1'^r + p_2'^r} + (p_1^2p_2 + p_1p_2^2)\frac{p_1'^r}{p_1'^r + p_2'^r} + p_1p_2.$$

The following theorem shows the relationship between VOTING and and the performance of $\text{OPT}_W$.

**Theorem 12** *For any set of preferences $p$,*

$$\lim_{m \to \infty} \Pi(\text{VOTING}, p) \geq 2(\sqrt{2} - 1) \cdot \Pi(\text{OPT}, p).$$

*Proof:* Let $r$ be the expected multiplicity of a neighbor, and note that as $m \to \infty$, $r \to \infty$. Since $p_1' \geq p_2'$, $\lim_{r \to \infty}$ of these quantities is 1 and 0 respectively. The theorem follows by analyzing $\Pi(\text{VOTING}_r, p)$. □

Thus, the above theorem asserts that the performance of VOTING approaches that of $\text{OPT}_W$ as $m \to \infty$.

Since VOTING is a generalization of NEIGHBOR, it is important to study the performance of VOTING as a function of $m$. Suppose $m = \Omega(n^{2-\epsilon})$. Then with high probability most nodes have at least one neighbor with multiplicity $\Omega(1/\epsilon)$. Thus,

**Corollary 13** *For any set of preferences $p$, if $m = n^{2-\epsilon}$, $\Pi(\text{VOTING}, p) \geq \Pi(\text{VOTING}_{1/\epsilon}, p)$.*

## 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, and 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] 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.

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

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

[10] 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.

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

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

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

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

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

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

[17] 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.

[18] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and 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.

[19] 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

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

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

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

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