

Manipulation Robustness of Collaborative Filtering Systems

Benjamin Van Roy
Stanford University
bvr@stanford.edu

Xiang Yan
Stanford University
xyan@stanford.edu

November 16, 2009

Abstract

A collaborative filtering system recommends to users products that similar users like. Collaborative filtering systems influence purchase decisions, and hence have become targets of manipulation by unscrupulous vendors. We present theoretical and empirical results demonstrating that while nearest neighbor algorithms, which are widely used in commercial systems, are highly susceptible to manipulation, two classes of collaborative filtering algorithms which we refer to as *linear* and *asymptotically linear* are relatively robust. These results provide guidance for the design of future collaborative filtering systems.

1 Introduction

While the expanding universe of products available via Internet commerce provides consumers with valuable options, sifting through the numerous alternatives to identify desirable choices can be challenging. Collaborative filtering (CF) systems aid this process by recommending to users products desired by similar individuals.

At the heart of a CF system is an algorithm that predicts whether a given user will like various products based on his past behavior and that of other users. Nearest neighbor (NN) algorithms, for example, have enjoyed wide use in commercial CF systems, including those of Amazon, Netflix, and Youtube (Bennett, 2006; Linden et al., 2003; Ryan, 2008). A prototypical NN algorithm stores each user's history, which may include, for instance, his product ratings and purchase decisions. To predict whether a particular user will like a particular product, the algorithm identifies a number of other users with similar histories. A prediction is then generated based on how these so-called neighbors have responded to the product. This prediction could be, for example, a weighted average of past ratings supplied by neighbors.

Because purchase decisions are influenced by CF systems, they have become targets of manipulation by unscrupulous vendors. For instance, a vendor can create multiple online identities and use each to rate his own product highly and competitors' products poorly. As an example, Amazon's CF system was manipulated so that users who viewed a spiritual guide written by a well-known Christian evangelist were subsequently recommended a sex manual for gay men (Olsen,

2002). Although this incident may not have been driven by commercial motives, it highlights the vulnerability of CF systems. The research literature offers further empirical evidence that NN algorithms are susceptible to manipulation (Burke et al., 2005; Lam and Riedl, 2004; Mehta and Nejdil, 2008; Mobasher et al., 2005, 2006; O’Mahony et al., 2004; Sandvig et al., 2007; Zhang et al., 2006).

In order to curb manipulation, one might consider authenticating each user by asking for, say, a credit card number to limit the number of fake identities. This may be effective in some situations. However, in Internet services that do not facilitate financial transactions, such as Youtube or personalized search engines, requiring authentication would intrude privacy and drive users away. One might also consider using only customer purchase data, when they are available, as a basis for recommendations because they are likely generated by honest users. Recommendation quality may be improved, however, if higher-volume data such as page views are also properly utilized.

In this paper, we seek to understand the extent to which manipulators can hurt the performance of CF systems and how CF algorithms should be designed to abate their influence. We find that, while NN algorithms can be quite sensitive to manipulation, CF algorithms that carry out predictions based on a particular class of probabilistic models are surprisingly robust. For reasons that we will explain in the paper, we will refer to algorithms of this kind as *linear CF algorithms*.

We find that as a user rates an increasing number of products, the average accuracy of predictions made by a linear CF algorithm becomes insensitive to manipulated data. For instance, even if half of all ratings are provided by manipulators who try to promote half of the products, predictions for users with long histories will barely be distorted, on average. To provide some intuition for why our results should hold, we now offer an informal argument. A robust CF algorithm should learn from its mistakes. In particular, differences between predicted and realized ratings should help improve predictions of subsequent ratings. A linear CF algorithm generates predictions based on a probability distribution that is a convex combination of two distributions: one that it would learn given only data generated by honest users and one that it would learn given only manipulated data. As a user provides more ratings, it becomes increasingly clear which of these two distributions better represents his preferences. As a result, the weight placed on manipulated data diminishes and distortion vanishes.

The main theoretical result of this paper formalizes the above argument. In particular, we will define a notion of distortion induced by manipulators and establish an upper bound on distortion, which takes a particularly simple form:

$$\text{distortion} \leq \frac{1}{n} \ln \frac{1}{1-r}.$$

Here r is the fraction of data that is generated by manipulators and n is the number of products that have already been rated by a user whose future ratings we wish to predict. The bound is very general. First, it applies to all linear CF algorithms. Second, it applies to all manipulation

strategies even if manipulators coordinate their actions and produce data with knowledge of all data generated by honest users. The bound demonstrates that as the number of prior ratings n increases, distortion vanishes. It also identifies the number required to limit distortion to a certain level. This offers guidance for the design of a recommendation system. The system may, for example, assess and inform users about the confidence of each recommendation. The system may also require a new user to rate a number of products before making recommendations to him. To put this in perspective, consider the following numerical example. Suppose a CF system that accepts binary ratings predicts future ratings correctly 80% of the time in the absence of manipulation. If 10% of all ratings are provided by manipulators, according to our bound, the system can maintain a 75% rate of correct predictions by requiring each new user to rate at least 21 products before receiving recommendations.

To broaden the scope of our analysis, we will also study CF algorithms that behave like linear CF algorithms asymptotically as the size of the training set grows. This class of algorithms, which we refer to as *asymptotically linear*, is more flexible in accommodating modeling assumptions that may improve prediction accuracy. We will establish that a relaxed version of our distortion bound for linear CF algorithms applies to asymptotically linear CF algorithms.

We will also show that our distortion bound does not generally hold for NN algorithms. Intuitively, this is because prediction errors do not always improve the selection of neighbors. In particular, as a user provides more ratings, manipulated data that contribute to inaccurate predictions of his future ratings may remain in the set of neighbors while data generated by honest users may be eliminated from it. As a result, distortion of predictions may not decrease. We will later provide an example to illustrate this.

In addition to theoretical results, this paper provides an empirical analysis using a publicly available set of movie ratings generated by users of Netflix’s recommendation system. We produce a distorted version of this data set by injecting manipulated ratings generated using a manipulation technique studied in prior literature. We then compare results from application of three CF algorithms: an NN algorithm, a linear CF algorithm called the kernel density estimation algorithm, and an asymptotically linear CF algorithm called the naive Bayes algorithm. Results demonstrate that while performance of the NN algorithm is highly susceptible to manipulation, those of kernel density estimation and naive Bayes algorithms are relatively robust. In particular, the latter two experience distortions lower than the theoretical bound we provide, whereas the distortion experienced by the former exceeds it by far.

One might also wonder whether manipulation robustness of a CF algorithm comes at the expense of its prediction accuracy. As an example, consider an algorithm that fixes predictions for all ratings to be a constant, without regard to the training data. This algorithm is uninfluenced by manipulation but is likely to yield poor predictions, and is therefore not useful. In our experiments, the accuracy demonstrated by each of the three algorithms seems reasonable. This suggests that accuracy can be achieved alongside robustness.

Our results together suggest that commercial recommendation systems using NN algorithms can be made more robust by adopting approaches that we describe. Note that we are not proposing that real-world systems should implement the specific algorithms we present in this paper. Rather, our analysis highlights properties of CF algorithms that lead to robustness and practitioners may benefit from taking these properties into consideration when designing CF systems.

This paper is organized as follows. In the next section, we survey related work. In Section 3, we formulate a simplified model that serves as a context for studying alternative CF algorithms. We then establish results about the manipulation robustness of NN, linear, and asymptotically linear CF algorithms in Section 4. In Section 5, we present our empirical study. We make some closing remarks in a final section.

2 Related Work

Early research on CF systems focused on their performance in the absence of manipulation (Breese et al., 1998; Drineas et al., 2002; Herlocker et al., 1999; Kleinberg and Sandler, 2008; Motwani and Vassilvitskii, 2007; Moon and Russell, 2008; Sarwar et al., 2001; Schafer et al., 2001). Almost all work on manipulation robustness has been empirical. For example, Burke et al. (2005); Lam and Riedl (2004); Mehta and Nejd (2008); Mobasher et al. (2005, 2006); O’Mahony et al. (2004); Sandvig et al. (2007); Zhang et al. (2006) present studies on product ratings made publicly available by Internet commerce sites. In each case, manipulated ratings were injected, and CF algorithms were tested on the altered data sets. The results point out that NN algorithms and their variants are susceptible to manipulation. This line of work identifies an effective manipulation scheme, which is to create multiple identities and with each identity, provide positive ratings on products to be promoted while rating other products in a manner indistinguishable from that of honest users. In Mehta and Nejd (2008); Mobasher et al. (2006); Zhang et al. (2006), algorithms based on probabilistic latent semantic analysis and principal component analysis were tested. It turns out that these algorithms are asymptotically linear under certain assumptions about the data, and indeed, empirical results in these papers suggest that they are relatively robust to manipulation. These prior results support the conclusions of our work.

To the best of our knowledge, the first theoretical work on manipulation robustness of CF algorithms is reported in O’Mahony et al. (2004). This work analyzed a NN algorithm that uses the majority rating among a set of neighbors as the prediction of a user’s rating in an asymptotic regime of many users, each of whom rates all products. Manipulators rate as honest users would except on one fixed product. A bound is established on the algorithm’s prediction error for this product’s rating as a function of the percentage of ratings provided by manipulators. In our work, we do not require users to rate all products and do not constrain manipulators to any particular strategies. Further, we study the performance distortion on average, rather than for a single product. Finally, a primary contribution of our work is in establishing manipulation robustness of

linear and asymptotically linear CF algorithms, which turn out to be superior to NN algorithms in this dimension.

Theoretical results established in Resnick and Sami (2007) also relate to our work. In that paper, a mechanism is proposed where each user accumulates a reputation based on ratings he supplies and the user’s influence on recommendations is limited by his reputation. The authors establish that each user maximizes his expected reputation by reporting honestly and provide a bound on the distortion that manipulators can induce. This bound grows linearly in the number of identities generated by manipulators. Our results differ in several ways. First, the robustness result of Resnick and Sami (2007) requires that manipulators act “myopically” in the sense that they do not game the reputation system. Our results make no assumptions that restrict manipulation strategies. A second difference is that our bound on distortion grows with the fraction rather than the number of user identities generated by manipulators. Hence, our bound implies that the system should be robust as the number of honest users grows, regardless of the number of manipulator identities. Finally, our results indicate that linear and asymptotically linear CF algorithms are inherently robust, without requiring an auxiliary reputation system. Overlaying a reputation system to adjust recommendations made by such algorithms would lead to undesirable distortions, even in the absence of manipulators. On the other hand, the reputation mechanism proposed in Resnick and Sami (2007) can be applied to any recommendation system, not just those that are linear or asymptotically linear.

Resnick and Sami (2007) also study the degree to which useful information is lost when their proposed mechanism is used to abate the influence of manipulators, and this line of work is extended in Resnick and Sami (2008), where a more fundamental result on the trade-off between robustness and information loss is established. Our work takes a different perspective in identifying classes of CF algorithms that are inherently robust. In some practical contexts, these CF algorithms may be entirely appropriate, and there is no information loss incurred because recommendations are not altered for the sake of robustness. In other contexts, to ensure manipulation robustness one might consider using such a CF algorithm to approximate one that would be more appropriate in the absence of manipulators. It would be interesting to understand the information cost incurred, along the lines studied in Resnick and Sami (2008), when employing such an approximation.

Several researchers have proposed alternative approaches to abating the influence of manipulators. In Massa and Avesani (2008); O’Donovan and Smyth (2006), researchers propose leveraging trust relationships among users to weight recommendations and fend off manipulation. Mehta (2007); Mobasher et al. (2007); Sandvig et al. (2007); Williams et al. (2007) suggest detecting manipulated ratings based on their patterns and discounting their impact. Our work complements this growing literature. First, additional sources of information can be integrated into the probabilistic framework that we introduce in this paper to further enhance manipulation robustness. Second, the analytical methods that we develop may be useful for studying the benefits of incorporating such information.

Distortion due to manipulation may also be viewed as a loss of utility in a sequential decision problem induced by errors in initial beliefs. Our analysis is based on ideas similar to those that have been used to study the latter topic, which is discussed in Gossner and Tomala (2008).

More broadly speaking, apart from collaborative filtering, there are other ways to aggregate users' responses to products in order to provide recommendations. Research has been performed on the manipulation robustness of these systems as well. To get a flavor of this line of work, see Bhattacharjee and Goel (2007); Dellarocas (2006); Friedman et al. (2007); Miller et al. (2005).

3 Model

We now formulate a simplified model that will serve as a context for assessing performance of alternative CF algorithms. We will first define the product ratings that we work with and then introduce measures of distortion induced by manipulators. For reference, we tabulate some of our mathematical notation in Appendix B.

3.1 Ratings Vectors

In our model, a user selects ratings from a set \mathbf{S} . To simplify our discussion, we let \mathbf{S} be a finite subset of $[0, 1]$. For example, \mathbf{S} could be $\{0, 1\}$ with 0 representing a negative rating and 1 representing a positive rating. Note that all the results in this paper can be generalized to accommodate any finite set \mathbf{S} . There are N products, and a user's type is identified by a vector in \mathbf{S}^N . Each n th component of this vector reflects how the user would rate the n th product after inspecting it.

The CF system has access to ratings provided by M identities who have rated products in the past. The data from each m th identity takes the form of a ratings vector $w^m \in S^N$, where $S = \mathbf{S} \cup \{\circ\}$. Here, an element of \mathbf{S} represents a product rating whereas a circle indicates that a product has not been rated. The collection of ratings vectors makes up the system's *training data* $W = (w^1, \dots, w^M) \in S^{N \times M}$, which is used by a CF algorithm to predict future ratings.

Consider a user who is distinct from identities that generated the training data and for whom we will generate recommendations. We will refer to such a user as an *active user*. We will view a CF algorithm as a function mapping a triplet $(n, x, W) \in \{1, \dots, N\} \times S^N \times S^{N \times M}$ to a probability mass function (PMF) $p_{n,x,W}$ over \mathbf{S} . The PMF represents beliefs about how an active user who has so far provided ratings x would rate product n . Such an algorithm can be used to guide recommendations; for example, the CF system might recommend to the active user the product he is most likely to rate highly among those that he has not yet rated.

3.2 Kullback-Leibler Distortion

To study the influence of manipulation, we consider a situation where a fraction r of the identities are created by manipulators, while the remaining fraction $1 - r$ correspond to distinct honest users.

We denote the honest ratings vectors by $y^1, \dots, y^{(1-r)M} \in S^N$ and the manipulated ratings vectors by $z^1, \dots, z^{rM} \in S^N$. Let $Y = (y^1, \dots, y^{(1-r)M})$ and $Z = (z^1, \dots, z^{rM})$ so that the training data is $W = (Y, Z)$.

To assess distortion of predictions made by a CF algorithm, we consider the following thought experiment. A hypothetical active user begins with a ratings vector x^0 , with each n th component set to $x_n^0 = \circ$, and inspects products in an order $\nu = (\nu_1, \dots, \nu_N) \in \sigma_N$, where σ_N denotes the set of permutations of $\{1, \dots, N\}$. After inspecting product ν_k , the user rates it by sampling from the PMF $p_{\nu_k, x^{k-1}, Y}$. An updated ratings vector x^k is generated by incorporating this new rating in x^{k-1} . This stochastic process reflects how we would think honest users behave based on the CF algorithm and uncorrupted data set Y . We introduce the following measure of distortion, which we refer to as *Kullback-Leibler (KL) distortion*:

$$d_n^{\text{KL}}(p, \nu, Y, Z) = \frac{1}{n} \sum_{k=1}^n \mathbb{E} \left[D \left(p_{\nu_k, x^{k-1}, Y} \parallel p_{\nu_k, x^{k-1}, (Y, Z)} \right) \right],$$

where D denotes Kullback-Leibler divergence with the natural log. That is, for any two PMFs p and q with support U , $D(p \parallel q) = \sum_{u \in U} p(u) \ln(p(u)/q(u))$. This measure of the difference between PMFs is commonly used in information theory.

For each k , the PMF $p_{\nu_k, x^{k-1}, Y}$ represents the prediction that would be made in the absence of manipulators, whereas $p_{\nu_k, x^{k-1}, (Y, Z)}$ is what it becomes as a consequence of manipulation. Hence, $D \left(p_{\nu_k, x^{k-1}, Y} \parallel p_{\nu_k, x^{k-1}, (Y, Z)} \right)$ measures the extent to which the manipulated data Z influences the prediction. We take the expectation of this quantity, with x^{k-1} distributed as the CF algorithm would have predicted if the data set were not corrupted by manipulated data. KL distortion $d_n^{\text{KL}}(p, \nu, Y, Z)$ averages these terms over the first n inspected products.

We can view the sequence x^0, \dots, x^n of the aforementioned thought experiment as a Markov chain. In particular, given p and an ordering ν , the state of this Markov chain transitions according to

$$x_j^k \sim \begin{cases} p_{\nu_k, x^{k-1}, W} & \text{if } j = \nu_k, \\ \delta(\cdot, x_j^{k-1}) & \text{otherwise,} \end{cases}$$

where δ is the Kronecker delta function. Consider the PMF $P(\cdot | W, \nu)$ of x^N , parameterized by W and ν . Clearly, p is a conditional PMF with respect to P :

$$p_{\nu_n, x^{n-1}, W}(\mathbf{x}_{\nu_n} = \mathbf{s}) = P(\mathbf{x}_{\nu_n} = \mathbf{s} | x^{n-1}, W, \nu).$$

As such, the CF algorithm p is completely characterized by P . We will often identify CF algorithms in terms of such P . For example, with some abuse of notation, we let $d_n^{\text{KL}}(P, \nu, Y, Z) = d_n^{\text{KL}}(p, \nu, Y, Z)$ if P and p are alternative representations of the same CF algorithm.

3.3 Root-Mean-Squared Distortion

Collaborative filtering systems sometimes make use of scalar predictions rather than PMFs. Such a prediction can often be interpreted as the mean of a PMF. In such contexts, it may be natural to measure the impact of manipulation in terms of *root-mean-squared (RMS) distortion*:

$$d_n^{\text{RMS}}(p, \nu, Y, Z) = \sqrt{\frac{1}{n} \sum_{k=1}^n \mathbb{E} \left[\left(\hat{x}_{\nu_k, x^{k-1}, Y} - \hat{x}_{\nu_k, x^{k-1}, (Y, Z)} \right)^2 \right]},$$

where $\hat{x}_{\nu_k, x^{k-1}, Y}$ and $\hat{x}_{\nu_k, x^{k-1}, (Y, Z)}$ are means of PMFs $p_{\nu_n, x^{n-1}, Y}$ and $p_{\nu_n, x^{n-1}, (Y, Z)}$, respectively. When using scalar predictions, RMS distortion may offer a more transparent measure than KL distortion because it reports distortion in the same units as the predictions. RMS distortion is bounded by a function of KL distortion:

$$d_n^{\text{RMS}}(p, \nu, Y, Z) \leq \sqrt{\frac{1}{2} d_n^{\text{KL}}(p, \nu, Y, Z)}.$$

This bound is established in Proposition 1 of Appendix A.1.

3.4 Discussion

It is natural to question why we chose the aforementioned distortion measures over other candidates. For instance, an alternative might focus on the impact of manipulated data on the n most highly rated products. One reason for using KL and RMS distortions is that they are convex functions of predictions while this alternative measure is not. Convexity facilitates analysis. Further, as will be discussed in Section 5.6, in a recent competition, Netflix used RMS error to evaluate CF algorithms (Netflix Prize, 2006). This suggests that commercial CF algorithms are typically designed to minimize convex measures of error. Our choice of distortion measures is in line with this.

Another option is to focus on the maximum distortion over all products. While it would be attractive to guarantee small worst-case distortion for all manipulation schemes, that may not be achievable. Further, even useful CF algorithms can exhibit large worst-case distortion. Consider, for instance, a case where many manipulators aim to distort ratings of many different products, and the output of the algorithm is such that the rating of one product is significantly distorted while the rest are not. We might still say that the algorithm is reasonably robust whereas a worst-case measure would indicate otherwise. That said, we note that since KL and RMS distortions characterize average distortions, the robustness results of this paper do not provide guarantees on the distortion of individual product ratings.

4 Collaborative Filtering Algorithms

In this section, we first introduce the notion of a *probabilistic CF algorithm*. We then describe two classes of such algorithms, namely linear and asymptotically linear CF algorithms, and analyze their robustness to manipulation. Finally, we discuss nearest neighbor algorithms and their susceptibility to manipulation.

4.1 Probabilistic Collaborative Filtering Algorithms

We will call a CF algorithm P *probabilistic* if for each W , the type distribution $P(\cdot | W, \nu)$ does not depend on the active user's ordering of products ν , in which case we drop ν from the list of conditioning variables. This implies that for each W , ν , k , and x^{k-1} , the prediction of x_{ν_k} is given by the PMF

$$p_{\nu_k, x^{k-1}, W}(x_{\nu_k} = s) = P(x_{\nu_k} = s | x^{k-1}, W).$$

Note that any CF algorithm that generates predictions by carrying out Bayesian inference based on a generative model in which pairs of type and ratings vectors are sampled i.i.d. is probabilistic. In particular, suppose user types $\mathbf{w}^m \in \mathbf{S}^N$ are sampled i.i.d. from a type distribution π and ratings vectors $w^m \in S^N$ are sampled from a conditional distribution ρ , conditioned on corresponding type vector samples, which for each n assigns either $w_n^m = \circ$ or $w_n^m = \mathbf{w}_n^m$. Then, an algorithm that predicts x_{ν_k} by conditioning the joint type-ratings vector distribution on the observed components of x^{k-1} along with a training set W is probabilistic.

4.2 Linear Collaborative Filtering Algorithms

We say that a probabilistic CF algorithm P is *linear* if for any $W_1 \in S^{N \times M_1}$ and $W_2 \in S^{N \times M_2}$,

$$P(\cdot | (W_1, W_2)) = \frac{M_1}{M_1 + M_2} P(\cdot | W_1) + \frac{M_2}{M_1 + M_2} P(\cdot | W_2).$$

This definition states that the PMF $P(\cdot | (W_1, W_2))$ that a linear CF algorithm P generates based on training data (W_1, W_2) is a convex combination of two PMFs: the PMF $P(\cdot | W_1)$ that it generates based on W_1 and the PMF $P(\cdot | W_2)$ that it generates based on W_2 .

We will introduce examples of linear CF algorithms in Section 5.3. For now, we will examine the KL distortion that manipulators can induce on an arbitrary linear CF algorithm. Consider training data $W = (Y, Z)$ consisting of ratings vectors Y from honest users and Z from manipulators, with the latter making up a fraction r of the training data. The following theorem establishes a bound on the resulting KL distortion.

Theorem 1. *For all N , P , M , $r \in \{0, 1/M, \dots, (M-1)/M\}$, $Y \in S^{N \times (1-r)M}$, $Z \in S^{N \times rM}$, $\nu \in \sigma_N$, and $n \in \{1, \dots, N\}$,*

$$d_n^{\text{KL}}(P, \nu, Y, Z) \leq \frac{1}{n} \ln \frac{1}{1-r}.$$

This result is proved in Appendix A.2.

Note that the bound only depends on the number of active user ratings n and the fraction of data r generated by manipulators. Hence, it represents a worst case bound over all choices of number of products N , linear CF algorithm P , cardinality M and content (Y, Z) of the training set, and the order ν in which the active user rates products. This means, for example, that it applies even if manipulators coordinate with each other and select ratings with knowledge of the specific CF algorithm P , the honest ratings Y , and the ordering ν . Among other things, this makes the bound relevant for realistic models of how a recommendation system might sequence products for a user; for example, each ν_k could be the product that the CF algorithm predicts as being most desirable among those remaining after the user has inspected products ν_1, \dots, ν_{k-1} .

Note that KL distortion vanishes as the number n of products rated by the active user increases. To develop intuition for why this happens, we now offer an informal argument. Observe that $P(\cdot | (Y, Z)) = (1 - r)P(\cdot | Y) + rP(\cdot | Z)$. If $P(\cdot | Y)$ is identical to $P(\cdot | Z)$, then $P(\cdot | (Y, Z))$ is equal to $P(\cdot | Y)$ and distortion will be zero. Otherwise, if $P(\cdot | Y)$ and $P(\cdot | Z)$ are different, as an active user inspects and rates products in the manner that we define, his ratings will tend to be distinguished as sampled from $P(\cdot | Y)$ rather than from $P(\cdot | Z)$. As such, the influence of Z on predictions diminishes as n grows.

The term $\ln(1/(1-r))$ captures the dependence of KL distortion on the fraction of data produced by manipulators. As one would expect, this term vanishes when r is zero. When r is small but non-zero, this term is well-approximated by r , and therefore KL distortion is bounded by about r/n . KL distortion represents a measure of average distortion over n inspected products. Hence, when r is small, it can be thought of as a bound on the *total* distortion experienced by an active user. It is interesting that this total distortion is bounded by the fraction of data generated by manipulators, regardless of how many products the user inspects.

Note that the KL distortion represents an average level of distortion across items. As such, the bound does not preclude the possibility that manipulators can impose strong influence over the rating of a particular item. To do so, however, they would have to build credibility by rating other items as honest users would. Doing so reduces distortion on those items so that such manipulators must contribute value to the system in order to distort a single item. Further, if different manipulators aim to distort ratings of different items, their helpful and harmful behaviors tend to offset each other.

KL distortion also represents an average over the distribution of active user types. This distribution is implicitly assumed to be one estimated from honest user data. Hence, the active user is assumed to be a representative honest user. Our model assumes that manipulated data used to generate recommendations does not vary by active user. This is reasonable because manipulators can only provide one data set that must be used in generating all recommendations regardless of the identity of the active user. The manipulators may design their data to target a particular type of honest user, and the bound does not preclude the possibility that users of this type receive highly

distorted recommendations. However, the impact averaged across honest user types is limited as established by the bound.

As a corollary of Theorem 1 and Proposition 1 we have the following bound on RMS distortion.

Corollary 1. For all $N, P, M, r \in \{0, 1/M, \dots, (M-1)/M\}$, $Y \in S^{N \times (1-r)M}$, $Z \in S^{N \times rM}$, $\nu \in \sigma_N$, and $n \in \{1, \dots, N\}$,

$$d_n^{\text{RMS}}(P, \nu, Y, Z) \leq \sqrt{\frac{1}{2n} \ln \frac{1}{1-r}}.$$

Figure 1 illustrates how this bound depends on r and n . In particular, the four curves from bottom to top correspond to cases where $r = 0.01, 0.05, 0.1, 0.2$, respectively. As the fraction r of data generated by manipulators increases, so does the possible distortion. As the number n of products rated by the active users increases, the bound on distortion diminishes. The bound can offer useful guidance. For example, it ensures that if an active user has rated 22 products and no more than 10% of the training data is manipulated, then the RMS distortion induced by manipulators is less than 0.05.

Additional perspective can be gained if we consider a context where ratings are binary-valued and a CF system generates binary-valued predictions, assigning a value of 1 when the scalar prediction exceeds $1/2$ and 0 otherwise. We establish in Proposition 2 in Appendix A.1 that the amount by which manipulators can reduce the average probability of correct predictions is bounded by RMS distortion:

$$\frac{1}{n} \sum_{k=1}^n \left(\Pr(\mathbf{x}_{\nu_k} = \mathbf{1}(\hat{x}_{\nu_k, x^{k-1}, Y} > 1/2)) - \Pr(\mathbf{x}_{\nu_k} = \mathbf{1}(\hat{x}_{\nu_k, x^{k-1}, (Y, Z)} > 1/2)) \right) \leq d_n^{\text{RMS}}(P, \nu, Y, Z).$$

Here, if the RMS distortion is less than 0.05, the average probability of correct predictions decreases by at most 0.05. Hence, if a binary CF system predicts ratings correctly 80% of the time in the absence of manipulation, it can maintain this probability at 75% in the presence of manipulation if it requires active users to rate 21 products before receiving recommendations.

4.3 Asymptotically Linear Collaborative Filtering Algorithms

We say that a probabilistic CF algorithm P is *asymptotically linear* if for all PMFs ψ and ϕ over S^N , $r \in [0, 1]$, and $\epsilon > 0$,

$$\lim_{m \rightarrow \infty} \Pr \left(D \left(((1-r)P(\cdot | U_m) + rP(\cdot | V_m)) \parallel P(\cdot | (U_m, V_m)) \right) \geq \epsilon \right) = 0,$$

where for each m , $U_m = (u^1, \dots, u^{m-l}) \in S^{N \times (m-l)}$, $V_m = (v^1, \dots, v^l) \in S^{N \times l}$, $l \sim \text{Binomial}(m, r)$, and $u^1, \dots, u^{m-l} \sim \psi$ and $v^1, \dots, v^l \sim \phi$ are i.i.d. sequences.

To interpret the preceding definition, think of each sample in the training data (U_m, V_m) as generated in the following way: with probability $1-r$, a ratings vector is sampled from ψ and

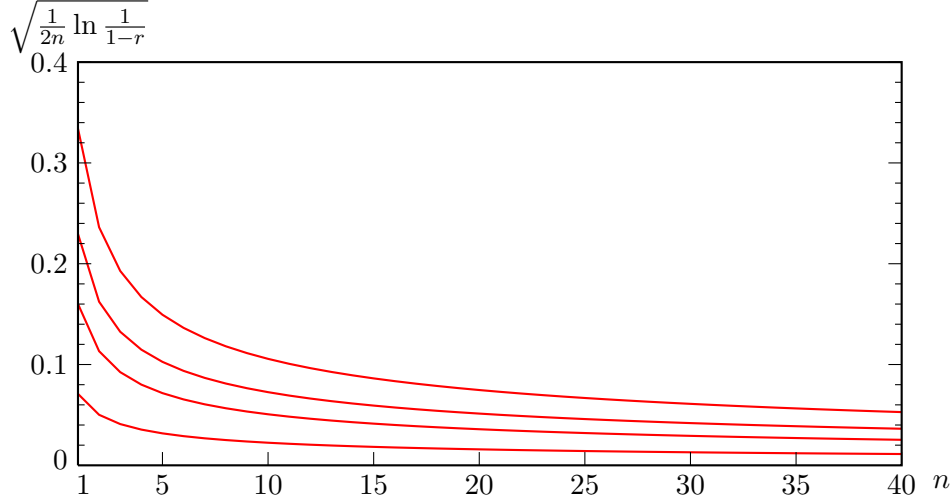


Figure 1: Bound on $d_n^{\text{RMS}}(P, \nu, Y, Z)$ as a function of n . The four curves from bottom to top are for cases where $r = 0.01, 0.05, 0.1,$ and $0.2,$ respectively.

placed in U_m , and with probability r , a ratings vector is sampled from ϕ and placed in V_m . As m grows, an asymptotically linear CF algorithm behaves like a linear CF algorithm in that the PMF $P(\cdot | (U_m, V_m))$ converges in probability to a convex combination of $P(\cdot | U_m)$ and $P(\cdot | V_m)$. It can be shown using the weak law of large numbers that all linear CF algorithms are asymptotically linear.

We will establish that asymptotically linear CF algorithms are asymptotically robust, in a sense that we will make precise. This result also applies to a broader range of practical algorithms that are asymptotically linear in a more restricted sense, which we now define. Consider a set Ψ of PMFs over S^N . We say that a probabilistic CF algorithm P is *asymptotically linear with respect to Ψ* if for all $\psi, \phi \in \Psi$, $r \in [0, 1]$, and $\epsilon > 0$,

$$\lim_{m \rightarrow \infty} \Pr \left(D \left(((1-r)P(\cdot | U_m) + rP(\cdot | V_m)) \parallel P(\cdot | (U_m, V_m)) \right) \geq \epsilon \right) = 0,$$

where for each m , $U_m = (u^1, \dots, u^{m-l}) \in S^{N \times (m-l)}$, $V_m = (v^1, \dots, v^l) \in S^{N \times l}$, $l \sim \text{Binomial}(m, r)$, and $u^1, \dots, u^{m-l} \sim \psi$ and $v^1, \dots, v^l \sim \phi$ are i.i.d. sequences.

The following theorem and corollary characterize the robustness of asymptotically linear CF algorithms.

Theorem 2. *For all N, Ψ, P asymptotically linear with respect to Ψ , $\psi, \phi \in \Psi$, $r \in [0, 1)$, $\nu \in \sigma_N$, $n \in \{1, \dots, N\}$, and $\epsilon > 0$,*

$$\lim_{m \rightarrow \infty} \Pr \left(d_n^{\text{KL}}(P, \nu, Y_m, Z_m) \geq \frac{1}{n} \ln \frac{1}{1-r} + \epsilon \right) = 0,$$

where, for each m , $Y_m = (y^1, \dots, y^{m-l}) \in S^{N \times (m-l)}$, $Z_m = (z^1, \dots, z^l) \in S^{N \times l}$, $l \sim \text{Binomial}(m, r)$,

and $y^1, \dots, y^{m-l} \sim \psi$ and $z^1, \dots, z^l \sim \phi$ are i.i.d. sequences.

Corollary 2. For all N, Ψ, P asymptotically linear with respect to $\Psi, \psi, \phi \in \Psi, r \in [0, 1), \nu \in \sigma_N, n \in \{1, \dots, N\}$, and $\epsilon > 0$,

$$\lim_{m \rightarrow \infty} \Pr \left(d_n^{\text{RMS}}(P, \nu, Y_m, Z_m) \geq \sqrt{\frac{1}{2n} \ln \frac{1}{1-r}} + \epsilon \right) = 0,$$

where, for each $m, Y_m = (y^1, \dots, y^{m-l}) \in S^{N \times (m-l)}, Z_m = (z^1, \dots, z^l) \in S^{N \times l}, l \sim \text{Binomial}(m, r)$, and $y^1, \dots, y^{m-l} \sim \psi$ and $z^1, \dots, z^l \sim \phi$ are i.i.d. sequences.

Theorem 2 is proved in Appendix A.2 and Corollary 2 follows from Theorem 2 and Proposition 1. These results state that for any CF algorithm P that is asymptotically linear with respect to Ψ and any PMFs $\psi, \phi \in \Psi$, as honest users and manipulators sample more data from them, with high probability, the distortion bounds from Theorem 1 and Corollary 1 essentially apply to P . In particular, distortion will vanish as n grows. Note that this result applies even in situations where manipulators select their strategies with full knowledge of user behavior. In particular, the bound holds even if the distribution ϕ that generates manipulated data is chosen based on the distribution ψ that generates honest user data.

It is interesting that a certain class of CF algorithms that are consistent, in a sense that we will make precise, are asymptotically linear. Consider a set Φ of pairs (π, ρ) where π is a distribution over user types and ρ is a conditional distribution over ratings vectors conditioned on user types. Let $\pi\rho$ denote the ratings vector distribution implied by π and ρ . We say that Φ is convex if the corresponding set of joint distributions over types and ratings vectors is convex. We say that Φ is *identifiable* if each $(\pi, \rho) \in \Phi$ leads to a distinct ratings vector distribution $\pi\rho$. Given an identifiable set Φ , we say that a probabilistic CF algorithm P is *consistent with respect to Φ* if for all $(\pi, \rho) \in \Phi$ and $\epsilon > 0$,

$$\lim_{m \rightarrow \infty} \Pr \left(D \left(P(\cdot | W_m) \parallel \pi \right) \geq \epsilon \right) = 0,$$

where $W_m = (w_1, \dots, w_m) \in S^{N \times m}$ and $w_1, \dots, w_m \sim \pi\rho$ is an i.i.d. sequence. This definition is meant to capture algorithms that learn the true type distribution as the training set grows. The following theorem identifies an interesting class of asymptotically linear CF algorithms.

Theorem 3. Any probabilistic CF algorithm consistent with respect to an identifiable and convex set Φ is asymptotically linear with respect to Φ .

The preceding result, proved in Appendix A.2, implies that if data are sampled i.i.d. from a PMF $\pi\rho$ induced by an element (π, ρ) of an identifiable and convex set Φ , then an algorithm that is consistent with respect to Φ will be both accurate and robust to manipulation as the training set grows. In practice, even if it is unclear whether the identifiability and convexity conditions hold, one might still apply a consistent CF algorithm, with the hope that it will deliver reasonable accuracy

and robustness. In Section 5, we will empirically evaluate a consistent CF algorithm called the naive Bayes algorithm.

4.4 Nearest Neighbor Algorithms

Nearest neighbor algorithms, widely used in commercial CF systems (Bennett, 2006; Linden et al., 2003; Ryan, 2008), generally come in two flavors. The first predicts a user’s ratings based on those provided by similar users, referred to as neighbors. The second makes predictions on a product based on ratings that the user has provided on similar products, which can also be viewed as neighbors. In this section, we study a simple NN algorithm of the first kind and the extent to which its predictions can be distorted by manipulators. We show that the bounds of the previous section do not apply to this NN algorithm, and unlike the case of linear CF algorithms, distortion does not necessarily diminish as the active user inspects and rates products. Though our analysis focuses on a particular NN algorithm, the resulting insights apply more broadly and in particular, to NN algorithms of the second kind.

We study the case of binary ratings. NN algorithms identify and weight neighbors using a similarity measure. We will consider as a similarity measure the number of consistent ratings minus the number of inconsistent ratings, given by

$$s(x, y) = |\{1 \leq n \leq N : x_n = y_n \neq \circ\}| - |\{1 \leq n \leq N : \circ \neq x_n \neq y_n \neq \circ\}|,$$

for any pair of ratings vectors $x, y \in S^N$.

We consider an NN algorithm that predicts the future rating of product n for a user with ratings vector x by carrying out the following steps. First, the algorithm identifies the subset of the training data samples that offer ratings for product n . If this subset is empty, the NN algorithm optimistically predicts a rating of 1. Otherwise, from among these ratings vectors, the ones most similar to x are identified. We denote the resulting set of neighbors, which should be a singleton unless there is a tie, by $\mathcal{N}(n, x, W)$. Finally, an average of their ratings for product n forms the prediction:

$$\hat{x}_{n,x,W} = \frac{\sum_{w \in \mathcal{N}(n,x,W)} w_n}{|\mathcal{N}(n,x,W)|}.$$

Our observations extend to other more complicated similarity metrics and neighbor selection methods. However, we focus on this particular case in order to keep our analysis clean.

We now consider a simple setting that facilitates analysis of RMS distortion in our NN algorithm. We are interested in how RMS distortion changes as the number of ratings n provided by an active user grows. Since n cannot exceed the number of products N , we will define an ensemble of models indexed by N . To facilitate our construction, we will only consider even N .

To keep things simple, we restrict attention to a situation where honest users agree on the ratings of all products. In particular, there is a single user type \mathbf{x}^{odd} which rates odd-indexed

products 1 and even-indexed products 0. The user type PMF assigns all probability to this vector. Each honest ratings vector y^m is generated by sampling a random set of odd numbers between 1 and $N - 1$, then for each sample k , replacing components k and $k + 1$ of \mathbf{x}^{odd} with circles. We assume that the honest ratings Y of training data is such that each set of odd numbers between 1 and $N - 1$ is sampled exactly once. That is, each element of Y corresponds to an element of the set $\{(1, 0), (\circ, \circ)\}^{N/2}$. As such, there are $2^{N/2}$ honest ratings vectors.

Recalling the setting that we use for assessing distortion, we now consider an active user who inspects products in the ordering $\nu = (1, \dots, N)$, rating each based on the prediction of the NN algorithm. It is easy to see that when there are no manipulators, the NN algorithm perfectly predicts $\hat{x}_{k,x^{k-1},Y} = \mathbf{x}_k^{\text{odd}}$, and therefore, after the user inspects k products, his ratings history x^k has $x_j^k = \mathbf{x}_j^{\text{odd}}$ for $j \leq k$ and $x_j^k = \circ$ for $j > k$.

We assume that manipulators produce one half of the training data. For each honest ratings vector y^m , manipulators produce a ratings vector z^m which agrees with y^m on all products rated by y^m . However, circles in y^m are replaced by 1 for even indices and 0 for odd indices. That is, each z_m corresponds to an element of the set $\{(1, 0), (0, 1)\}^{N/2}$.

Suppose k is even. Given x^k , the NN algorithm predicts what the active user's rating will be for product $k + 1$. To do this, it identifies neighbors $\mathcal{N}(k + 1, x^k, (Y, Z))$, which includes the following subsets of the training data:

- A set Y_1 which consists of honest ratings vectors y^m where $y_j^m \neq \circ$ for $j \leq k + 1$.
- A set Z_1 that, for each $y^m \in Y_1$, includes the corresponding manipulated vector $z^m \in Z$.
- A set Z_2 which consists of each manipulated ratings vector z^m such that the corresponding honest ratings vector y^m has $y_j^m \neq \circ$ for $j \leq k$ and $y_{k+1}^m = \circ$.

Note that each of these sets is of cardinality $2^{(N-k)/2-1}$. Vectors in Y_1 and Z_1 correctly rate product $k + 1$ as 1, whereas vectors in Z_2 incorrectly rate it as 0. As a consequence, the prediction for product $k + 1$ is $\hat{x}_{k+1,x^k,(Y,Z)} = 2/3$ and the resulting squared error is

$$\left(\mathbf{x}_{k+1}^{\text{odd}} - \hat{x}_{k+1,x^k,(Y,Z)}\right)^2 = \frac{1}{9}.$$

The preceding argument applies for all even k . For odd k , it is easy to show that the NN algorithm correctly predicts $\mathbf{x}_{k+1}^{\text{odd}} = 0$. It follows that the RMS distortion for even n is

$$d_n^{\text{RMS}}(p, \nu, Y, Z) = \sqrt{\frac{1}{n} \sum_{k=1}^n \mathbb{E} \left[\left(\mathbf{x}_k^{\text{odd}} - \hat{x}_{k,x^{k-1},(Y,Z)} \right)^2 \right]} = \frac{1}{3\sqrt{2}}.$$

The preceding example shows that the RMS distortion of an NN algorithm for $r = 1/2$ does not decrease as n grows. This happens because manipulated data are strategically generated to be sufficiently similar to honest data so that no matter how many ratings an active user provides,

manipulated ratings vectors will make up a fixed fraction of the neighbors and consequently induce a significant degree of distortion.

In contrast, Corollary 1 establishes that linear CF algorithms exhibit more graceful behavior, with RMS distortion vanishing as n increases. This is not to say it is impossible to design an NN algorithm that exhibits more desirable behavior when applied to our example. However, it is difficult to know for sure whether a given variation will behave gracefully in all relevant situations.

4.5 Discussion

Let us offer an intuitive explanation for why linear CF algorithms should be robust to manipulation relative to NN algorithms. First note that robustness depends on how a CF algorithm learns from its mistakes. In particular, a robust algorithm should notice as it observes differences between its predictions and an active user’s ratings what is hurting versus helping its predictions. Recall that a linear CF algorithm P generates based on the training set (Y, Z) a PMF $P(\cdot | (Y, Z))$ that is a convex combination of $P(\cdot | Y)$ and $P(\cdot | Z)$, which are PMFs that the algorithm would generate based on Y and Z , respectively. As an active user rates more products, it will be increasingly clear by probabilistic inference that his ratings x are sampled from $P(\cdot | Y)$. In effect, inaccurate predictions induced by Z will increase the weight of $P(\cdot | Y)$ and decrease the weight of $P(\cdot | Z)$. This improves prediction accuracy.

In an NN algorithm, on the other hand, inaccurate predictions do not necessarily improve subsequent predictions. In particular, manipulated ratings vectors that contribute to inaccuracies may remain in the set of neighbors while honest ratings vectors may be eliminated from it. In the example in Section 4.4, for instance, manipulated data are generated so that no matter how long an active user’s ratings history is, each honest ratings vector selected as a neighbor has a manipulated counterpart that is as similar, and hence also selected as a neighbor. Consequently, as an active user provides more ratings, the numbers of honest and manipulated neighbors remain equal. As a result, inaccurate predictions do not decrease subsequent distortion.

5 Empirical Study

In this section, we present our empirical findings on the manipulation robustness of particular NN, linear, and asymptotically linear CF algorithms. We first introduce the data set that we worked with and then describe methods we used to evaluate robustness.

5.1 Data Set

We obtained a set of movie ratings provided by users, made publicly available by Netflix’s recommendation system. Each rating is an integer between 1 and 5, which we normalized to be in $\{0, 0.25, 0.5, 0.75, 1\}$ so that the analysis and results in our paper apply directly. We randomly sampled from the data set 5000 users and 500 movies. A total of approximately 200000 ratings

are provided by these users. We then randomly selected 4000 of these users and for the purpose of our experiments, treated them as honest users and their ratings as a training set Y . We used the ratings of the other 1000 users as a test set, which we refer to as X . We then generated three separate sets of 444, 1714, and 4000 manipulated ratings vectors, respectively. Each set, which we refer to as Z for simplicity of discussion, is generated to promote 50% of the movies by using a technique reported to be effective in the literature (Burke et al., 2005; Lam and Riedl, 2004; Mehta and Nejd, 2008; Mobasher et al., 2005, 2006; O’Mahony et al., 2004; Sandvig et al., 2007; Zhang et al., 2006). Specifically, we randomly sampled 250 of the 500 movies in Y and let each manipulated ratings vector in each Z assign the highest ratings to these movies, and assign a random rating to each of the other movies, sampled from the movie’s empirical marginal PMF of ratings in Y . We then replaced a random subset of ratings in Z with circles so that the fraction of circles in Z matched that in Y . Manipulated ratings vectors generated this way are meant to be similar to honest ratings vectors except on movies they promote.

5.2 Evaluation Methods

To evaluate the robustness of a CF algorithm P , we treated ratings in X as ratings that an active user would provide and used P to predict them. Specifically, we fixed n and for each ratings vector $x \in X$, identified n random products that it assigns ratings to and randomly permuted them to form an ordering $\nu^x = (\nu_1^x, \dots, \nu_n^x)$. For each $k \leq n$, let x^{k-1} be a ratings vector that agrees with x on products $\nu_1^x, \dots, \nu_{k-1}^x$ and assigns circles to the other products. P can then be used to generate a prediction $\hat{x}_{\nu_k^x, x^{k-1}, Y}$ of the rating of product ν_k^x based on x^{k-1} and the honest data set Y , as well as a prediction $\hat{x}_{\nu_k^x, x^{k-1}, (Y, Z)}$ based on the corrupted data set (Y, Z) . To assess the impact of manipulation, for each n , we computed the following quantity, which we will refer to as *empirical RMS distortion*:

$$\hat{d}_n^{\text{RMS}}(P, \nu^X, X, Y, Z) = \sqrt{\frac{1}{|X|} \sum_{x \in X} \frac{1}{n} \sum_{k=1}^n \left(\hat{x}_{\nu_k^x, x^{k-1}, Y} - \hat{x}_{\nu_k^x, x^{k-1}, (Y, Z)} \right)^2}.$$

Here, $\nu^X = \{(\nu_1^x, \dots, \nu_n^x) : x \in X\}$. Empirical RMS distortion measures in predictions for products rated by active users. It is similar to the RMS distortion $d_n^{\text{RMS}}(P, \nu, Y, Z)$ that we defined earlier, with one difference: whereas $d_n^{\text{RMS}}(P, \nu, Y, Z)$ involves samples \mathbf{x}_{ν_k} drawn from PMFs $p_{\nu_k, x^{k-1}, Y}$, elements of X are used as samples in defining $\hat{d}_n^{\text{RMS}}(P, \nu^X, X, Y, Z)$. We used empirical RMS distortion rather than RMS distortion to assess algorithms in our empirical study because computing RMS distortion would take too long, requiring compute time exponential in the number of products n rated by an active user. Further, if a CF algorithm generates a nearly correct distribution in the absence of manipulation, its empirical RMS distortion will be close to its RMS distortion.

One might also wonder whether the robustness of a CF algorithms comes at the expense of

prediction accuracy. To better understand the relationship between robustness and accuracy, we also computed *empirical RMS prediction error* for each CF algorithm:

$$\hat{\mathcal{E}}_n^{\text{RMS}}(P, \nu^X, X, Y) = \sqrt{\frac{1}{|X|} \sum_{x \in X} \frac{1}{n} \sum_{k=1}^n \left(x_{\nu_k^x} - \hat{x}_{\nu_k^x, x^{k-1}, Y} \right)^2}.$$

This quantity computes the RMS error of predictions for ratings in X when the algorithm uses Y as training data.

Parameters of algorithms that we tested were tuned by cross validation. This is a technique that selects parameter values based on performance with validation data separate from the training data. Specifically, we randomly sampled 20% of the users in Y . We treated their ratings as a validation set V and generated predictions based on the remaining ratings $Y \setminus V$. To tune a parameter γ within a range Γ , we considered a range of candidate values and in each case used the corresponding algorithm to predict ratings in V based on ratings in $Y \setminus V$ and computed the empirical RMS prediction error $\hat{\mathcal{E}}_n^{\text{RMS}}(P, \nu^V, V, Y \setminus V)$. The value of γ resulting in the least error is selected. When using (Y, Z) as the training set, we sampled the validation set V from (Y, Z) and tuned parameters in a similar manner.

For each algorithm P , we generated multiple samples of X , Y , Z , and ν^X , and averaged the resulting values of $\hat{d}_n^{\text{RMS}}(P, \nu^X, X, Y, Z)$ and $\hat{\mathcal{E}}_n^{\text{RMS}}(P, \nu^X, X, Y)$ across samples to obtain reliable estimates. To summarize with our notation, $\mathbf{S} = \{0, 0.25, 0.5, 0.75, 1\}$, $N = 500$, $(M, r) \in \{(4444, 0.1), (5714, 0.3), (8000, 0.5)\}$, and $1 \leq n \leq 40$. We tested three CF algorithms: a linear CF algorithm called kernel density estimation, an asymptotically linear CF algorithm called the naive Bayes algorithm, and an NN algorithm called the k nearest neighbor algorithm. We now describe the algorithms.

5.3 Kernel Density Estimation Algorithm

We will call a probabilistic CF algorithm P a kernel density estimation (KDE) algorithm with kernels $\{\mathcal{K}_w : w \in S^N\}$ if for any $W \in S^{N \times M}$,

$$P(\cdot | W) = \frac{1}{M} \sum_{w \in W} \mathcal{K}_w(\cdot),$$

where each \mathcal{K}_w is a PMF over \mathbf{S}^N parameterized by a ratings vector w . Note that each training data sample induces a PMF over types, and the PMF learned from the entire training set is the average among them. It turns out that all KDE algorithms are linear CF algorithms and vice versa. We establish this in Proposition 3 of Appendix A.3.

In our experiments, we considered a KDE algorithm with kernels $\{\mathcal{K}_w\}$ such that for each type

$\mathbf{x} \in \mathbf{S}^N$,

$$\mathcal{K}_w(\mathbf{x}) = \prod_{n=1}^N k_{w_n}(\mathbf{x}_n),$$

where for each $s \in S$, k_s is a PMF over \mathbf{S} defined as follows. For $s \neq \circ$, k_s is the unique PMF that satisfies $k_s(\mathbf{s})/k_s(s) = \exp(-|\mathbf{s} - s|/\beta)$ for all $\mathbf{s} \in \mathbf{S}$. For $s = \circ$, $k_s(\mathbf{s}) = 1/|\mathbf{S}|$ for all $\mathbf{s} \in \mathbf{S}$. That is, k_s assigns the highest probability to s and exponentially lower probabilities to values different from s if $s \neq \circ$, and assigns uniform probability to all values if $s = \circ$. It is easy to see that each \mathcal{K}_w defined this way is a PMF which assigns high probability to types similar to w and low probability to others. The constant $\beta > 0$, which we set to be 0.15 in our experiments, tunes the shape of k_s .

To predict the rating of product ν_n for a user with past ratings x^{n-1} , our KDE algorithm generates a PMF given by

$$P(\mathbf{x}_{\nu_n} = \mathbf{s} \mid x^{n-1}, W) = \frac{\sum_{w \in W} \prod_{i=1}^{n-1} k_{w_{\nu_i}}(x_{\nu_i}^{n-1}) k_{w_{\nu_n}}(\mathbf{s})}{\sum_{w \in W} \prod_{i=1}^{n-1} k_{w_{\nu_i}}(x_{\nu_i}^{n-1})},$$

for each $\mathbf{s} \in \mathbf{S}$. The corresponding scalar prediction is the expectation taken with respect to $P(\cdot \mid x_{n-1}, W)$:

$$\hat{x}_{\nu_n, x^{n-1}, W} = \sum_{\mathbf{s} \in \mathbf{S}} \mathbf{s} P(\mathbf{x}_{\nu_n} = \mathbf{s} \mid x^{n-1}, W).$$

5.4 Naive Bayes Algorithm

A naive Bayes (NB) algorithm assumes that the true distribution of data is a convex combination of distinct distributions in each of which features of the data are conditionally independent. It aims to learn from training data the weights of the combination and feature marginals within each distribution. For a formal analysis of the algorithm and its applications to other problem settings, see Cheeseman and Stutz (1996); Domingos and Pazzani (1997); John and Langley (1995). We now describe a particular version of the algorithm that we use and discuss a context in which it is consistent and asymptotically linear.

Our NB algorithm assumes that data are generated in the following way. First, user types are sampled from a distribution P^* where

$$P^*(\mathbf{w}) = \sum_{l=1}^L \eta_l \prod_{n=1}^N \theta_{l,n}^{(\mathbf{w}_n)}, \quad (1)$$

and ratings vectors are sampled from the conditional distribution Q^* where

$$Q^*(w \mid \mathbf{w}) = \begin{cases} q^{\|\mathbf{w}\|_\circ} (1-q)^{N-\|\mathbf{w}\|_\circ} & \text{if } \mathbf{w} \rightarrow w, \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

Here, $q \in [0, 1)$, $L \in \mathcal{Z}_+$, $\eta \in \Delta_L$, and $\theta_{l,n} \in \Delta_{|\mathbf{S}|}$ for all l, n , where Δ_k denotes the k -dimensional

unit simplex. We write $\mathbf{w} \rightarrow w$ for (\mathbf{w}, w) if for each n , either $w_n = \mathbf{w}_n$ or $w_n = \circ$. We let $\|w\|_\circ$ denote $|\{n : w_n = \circ\}|$ and let $\theta = \{\theta_{l,n} | 1 \leq l \leq L, 1 \leq n \leq N\}$.

To understand P^* and Q^* , let us consider the following process for generating ratings vectors. Let $\mathcal{P} = \{P_1^*, \dots, P_L^*\}$ be L PMFs over \mathbf{S}^N where each P_l^* satisfies

$$P_l^*(\mathbf{w}) = \prod_{n=1}^N \theta_{l,n}^{(\mathbf{w}_n)}$$

for all $\mathbf{w} \in \mathbf{S}^N$. That is, each \mathbf{w}_n is independently sampled and is equal to $\mathbf{s} \in \mathbf{S}$ with probability $\theta_{l,n}^{(\mathbf{s})}$. A type \mathbf{w} is generated by first selecting a PMF from \mathcal{P} where each P_l^* is chosen with probability η_l and then sampling from that PMF. A ratings vector w is then generated by randomly replacing each rating \mathbf{w}_n by a circle with probability q , independent of the value \mathbf{w}_n and whether other ratings are replaced by circles.

The algorithm also assumes a geometric prior for L and Dirichlet priors for η , θ , and q . Hence, the posterior probability density function (PDF) of (L, η, θ, q) conditioned on training data W is given by

$$f(L, \eta, \theta, q | W) = c f(L, \eta, \theta, q) \Pr(W | L, \eta, \theta, q), \quad (3)$$

where c is a normalizing constant and the prior PDF is

$$f(L, \eta, \theta, q) = f_L^\tau(L) f_\eta(\eta) \prod_{l,n} f_\theta(\theta_{l,n}) f_q(q),$$

with

$$\begin{aligned} f_L^\tau(L) &= c_L e^{-\tau L}, \\ f_\eta(\eta) &= c_\eta \prod_{l=1}^L \eta_l, \\ f_\theta(\theta_{l,n}) &= c_\theta \prod_{\mathbf{s} \in \mathbf{S}} \theta_{l,n}^{(\mathbf{s})}, \quad \forall l, n, \\ f_q(q) &= c_q q(1-q), \end{aligned}$$

and the training data likelihood is

$$\Pr(W | L, \eta, \theta, q) = \prod_{w \in W} \left(\left(q^{\|w\|_\circ} (1-q)^{N-\|w\|_\circ} \right) \left(\sum_{l=1}^L \eta_l \prod_{n:w_n \neq \circ} \theta_{l,n}^{(w_n)} \right) \right).$$

Here, subscripts L , η , θ , and q of the functions f_L^τ , f_η , f_θ , and f_q denote the parameters that the distributions are over. c_L , c_η , c_θ , and c_q are normalizing constants. We aim to compute parameters that maximize the posterior PDF using an expectation-maximization algorithm (Dempster et al.,

1977). In other words, the goal is to obtain parameters satisfying

$$(\hat{L}, \hat{\eta}, \hat{\theta}, \hat{q}) \in \underset{(L, \eta, \theta, q)}{\operatorname{argmax}} f(L, \eta, \theta, q | W).$$

We denote by $P(\cdot | W)$ the PMF over \mathbf{S}^N corresponding to selected parameters $(\hat{L}, \hat{\eta}, \hat{\theta})$. A prediction of the rating of product ν_n by a user with past ratings x^{n-1} is given by the PMF

$$P(\mathbf{x}_{\nu_n} = \mathbf{s} | x^{n-1}, W) = \frac{\sum_{l=1}^{\hat{L}} \hat{\eta}_l \prod_{k=1}^{n-1} \hat{\theta}_{l, \nu_k}^{(x_{\nu_k}^{n-1})} \hat{\theta}_{l, \nu_n}^{(\mathbf{s})}}{\sum_{l=1}^{\hat{L}} \hat{\eta}_l \prod_{k=1}^{n-1} \hat{\theta}_{l, \nu_k}^{(x_{\nu_k}^{n-1})}},$$

for each $\mathbf{s} \in \mathbf{S}$. The corresponding scalar prediction is

$$\hat{x}_{\nu_n, x^{n-1}, W} = \sum_{\mathbf{s} \in \mathbf{S}} \mathbf{s} P(\mathbf{x}_{\nu_n} = \mathbf{s} | x^{n-1}, W).$$

In our experiments, we tuned bandwidth τ by cross validation over the range $\Gamma = \{1, 10, 100, 1000, 10000, 100000\}$ and settled at $\tau = 10000$.

We now discuss a context in which the NB algorithm we have described is consistent and asymptotically linear. For any $q \in [0, 1)$, we let Φ_q be the set of all (π, ρ_q) pairs where π takes the form in (1) and ρ_q is the one distribution given by (2). We establish in Proposition 4 in Appendix A.3 that for any q , Φ_q is identifiable and convex, and the NB algorithm is consistent with respect to it. Then, by Theorems 2 and 3, the algorithm is asymptotically linear with respect to Φ_q and our distortion bounds apply. Note that the set $\mathcal{P} = \{\pi : (\pi, \rho_q) \in \Phi_q\}$ of type PMFs with corresponding pairs in Φ_q is the set of all PMFs over \mathbf{S}^N . This implies that for any PMFs π_1 and π_2 over \mathbf{S}^N , if honest and manipulated data are generated by first sampling types from these PMFs and then independently replacing each rating by a circle with the same probability q , then the NB algorithm will be asymptotically robust as the sample size grows. In our experiments, although the condition regarding replacement by circles may not hold, we still apply the NB algorithm with the hope that it will deliver reasonable robustness.

5.5 k Nearest Neighbor Algorithm

We will consider a class of k nearest neighbor (k NN) algorithms. Variations of k NN have been used as performance benchmarks in prior work (Burke et al., 2005; Lam and Riedl, 2004; Zhang et al., 2006). The version that we tested is based on the following measure of similarity between ratings vectors:

$$s(x, y) = \frac{\sum_{\{n: x_n \neq \circ, y_n \neq \circ\}} (x_n - \bar{x})(y_n - \bar{y})}{\sqrt{\sum_{\{n: x_n \neq \circ\}} (x_n - \bar{x})^2} \sqrt{\sum_{\{n: y_n \neq \circ\}} (y_n - \bar{y})^2}},$$

where

$$\bar{x} = \frac{\sum_{\{n: x_n \neq \circ\}} x_n}{|\{n : x_n \neq \circ\}|}, \quad \bar{y} = \frac{\sum_{\{n: y_n \neq \circ\}} y_n}{|\{n : y_n \neq \circ\}|}.$$

Note that this similarity measure resembles a correlation coefficient and is sometimes referred to as the cosine similarity measure (Sarwar et al., 2001).

To predict the rating of product ν_n , for $n \geq 3$, by a user with past ratings x^{n-1} , our k NN algorithm identifies a set of neighbors $\mathcal{N}(\nu_n, x^{n-1}, W)$ to be k most similar ratings vectors in W among those that provide ratings for product ν_n . The algorithm then generates the following scalar prediction:

$$\hat{x}_{\nu_n, x^{n-1}, W} = \min \left\{ \mathbf{s}_{\max}, \max \left\{ \mathbf{s}_{\min}, \bar{x}_{\nu_n}^{n-1} + \frac{\sum_{w \in \mathcal{N}(\nu_n, x^{n-1}, W)} s(w, x^{n-1})(w_{\nu_n} - \bar{w})}{\sum_{w \in \mathcal{N}(\nu_n, x^{n-1}, W)} |s(w, x^{n-1})|} \right\} \right\},$$

where $\mathbf{s}_{\max} = \max\{\mathbf{s} : \mathbf{s} \in \mathbf{S}\}$, $\mathbf{s}_{\min} = \min\{\mathbf{s} : \mathbf{s} \in \mathbf{S}\}$, and

$$\bar{w} = \frac{\sum_{\{n: w_n \neq \circ\}} w_n}{|\{n : w_n \neq \circ\}|}.$$

For a user with ratings history x^{n-1} where $n \leq 2$, $s(\cdot, x^{n-1})$ is not well-defined. In this case, the algorithm uses the average rating of product ν_n in the training data to generate the prediction:

$$\hat{x}_{\nu_n, x^{n-1}, W} = \min \left\{ \mathbf{s}_{\max}, \max \left\{ \mathbf{s}_{\min}, \frac{\sum_{\{w: w \in W, w_{\nu_n} \neq \circ\}} w_{\nu_n}}{|\{w : w \in W, w_{\nu_n} \neq \circ\}|} \right\} \right\}.$$

In our experiments, we tuned the number of neighbors k by cross validation over the range $\Gamma = \{1, 2, \dots, 40\}$ and settled at $k = 10$. Note that even though the k NN algorithm generates scalar predictions, it still fits our definition of CF algorithms because it is possible to come up with PMFs whose corresponding expectations equal the predictions $\hat{x}_{\nu_n, x^{n-1}, W}$. We do not explicitly define such a PMF, however, because it is not necessary for computing the empirical RMS distortion in our experiments.

5.6 Results

Figure 2 plots, for each of the three algorithms that we tested, the empirical RMS distortion as a function of the number of products rated by the active user, given different quantities of manipulated data. Our results suggest that in practice, NB and KDE algorithms are significantly more robust than k NN. As more ratings are provided, distortions experienced by all three algorithms decrease. When a user's ratings history is long, NB and KDE experience distortions significantly lower than that of k NN. Note that distortions of NB and KDE always stay below the bound in Corollary 1. The curves for k NN are flat for $n \leq 2$ because the algorithm provides the same predictions for the first two ratings \mathbf{x}_{ν_1} and \mathbf{x}_{ν_2} of an active user. The figure also demonstrates that, as fraction of

manipulated data r increases, distortions incurred by all three algorithms increase as well.

Note that, when a user’s ratings history is short, k NN and NB both experience higher empirical RMS distortions than KDE. This difference arises because while k NN and NB ignore circles, KDE uses them and as a result, tempers its predictions. To gain some intuition, let us consider the following problem instance where ratings are binary: the set of honest ratings Y consists of K vectors whose entries are all 1s and as many vectors whose entries are all circles. The set of manipulated ratings Z consists of K vectors whose entries are all 0s and as many vectors whose entries are all circles. To predict the first rating \mathbf{x}_{ν_1} of an active user, k NN would yield a prediction of 1 and $1/2$ based on Y and (Y, Z) , respectively, incurring an RMS distortion of $1/2$. KDE would yield a prediction close to $3/4$ based on Y and a prediction of $1/2$ based on (Y, Z) , incurring an RMS distortion of $1/4$, significantly less than that of k NN. Clearly, the presence of circles smooths KDE’s predictions and keeps its distortion low.

Figure 3 plots the empirical RMS prediction errors of the three algorithms. When n is large, their errors all decrease and in particular, NB offers the lowest error and k NN, the highest. The plot for k NN spikes around $n = 3$ because the algorithm switches its prediction method there: it generates predictions by using average ratings of all users for $n \leq 2$ and generates predictions by using average ratings of neighbors for $n \geq 3$.

To get a better sense of our results, we note that Netflix announced that its proprietary algorithm achieves an empirical RMS prediction error, normalized to our scale, of 0.238 on a large test set, and will award one million dollars to anyone that improves it to 0.214 (Netflix Prize, 2006). One might wonder why a decrease of 0.024 may have such a large impact on recommendation quality. We suspect that due to the large number of movies, many of them are given similar predicted ratings. As a result, a small improvement in prediction accuracy may tease apart these movies and identify the most desirable ones.

Compared to Netflix’s benchmark and target prediction errors, our results are reasonable but not competitive. This is because we did not focus on optimizing the prediction accuracy of the algorithms. If our objective was to achieve the highest possible accuracy while maintaining reasonable robustness, one option we could try is to fine-tune our robust algorithms to be accurate. For example, for KDE algorithms, we could work to identify more effective kernels. For NB algorithms, we could choose different priors or use methods other than expectation-maximization to find the model parameters. We could probably also design other robust linear and asymptotically linear CF algorithms that achieve higher accuracy as well. Overall, we are not suggesting that in practice, the specific algorithms that we presented should be directly implemented. Instead, one should either use them as starting points or take the insights that they yield into consideration when designing accurate and robust CF systems.

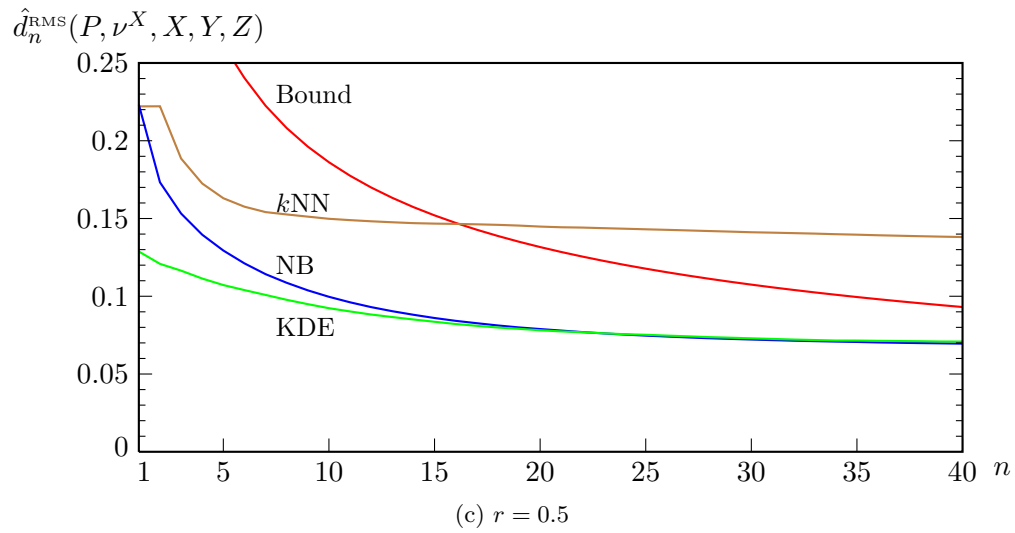
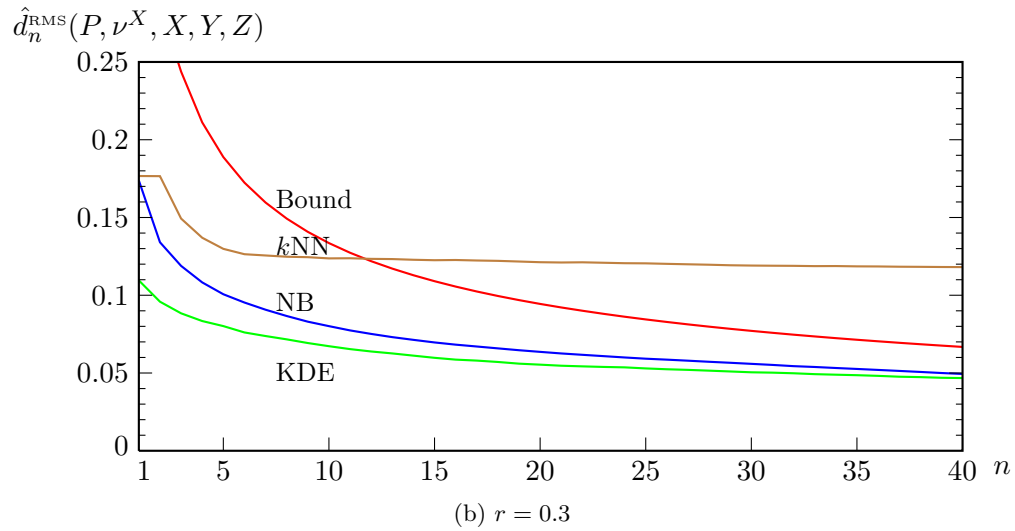
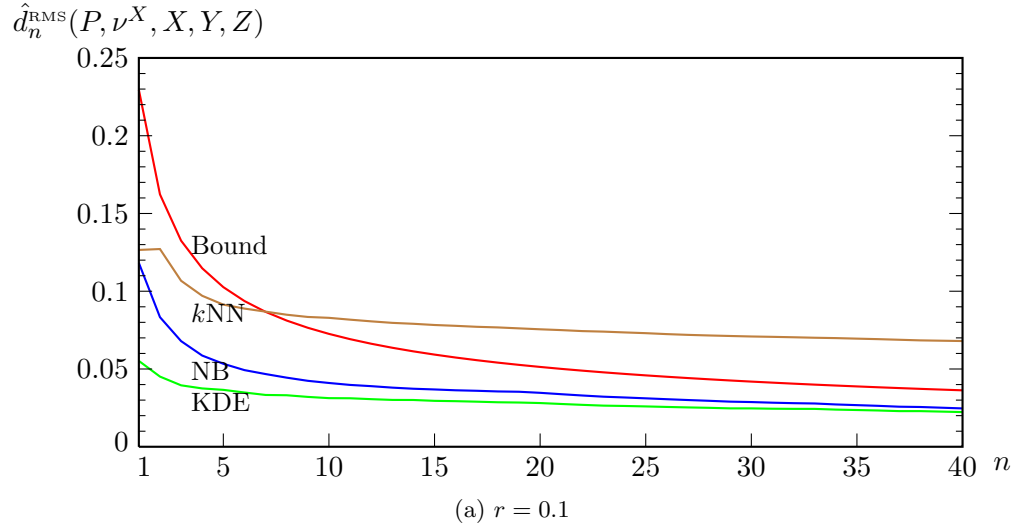


Figure 2: Empirical RMS distortion as a function of n , for different r .

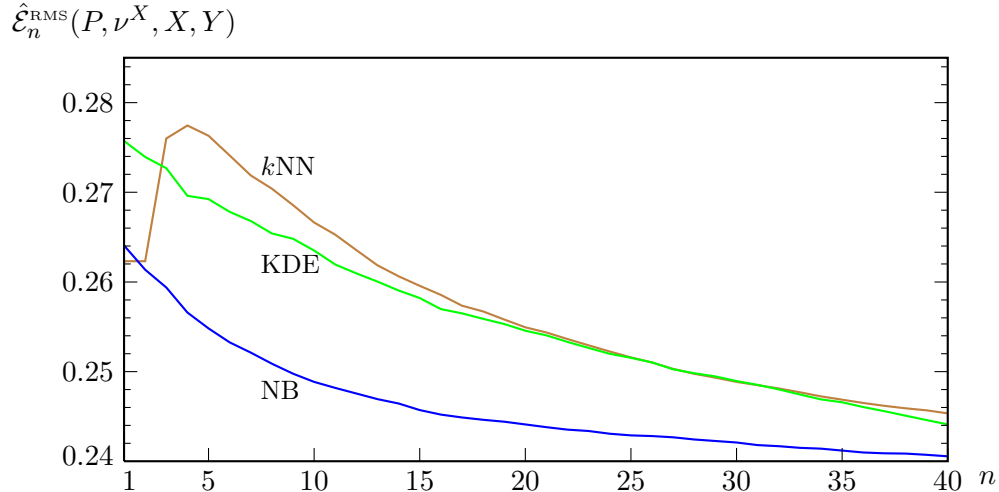


Figure 3: Empirical RMS prediction error as a function of n .

6 Conclusion

Our analytical and empirical work suggests that linear and asymptotically linear algorithms can be more robust to manipulation than commonly used nearest neighbor algorithms. Our results also suggest that it is possible to design algorithms that achieve accuracy alongside robustness. As such, recommendation systems of Internet commerce sites may improve their robustness to manipulation by adopting the approaches that we describe. They may also use the bounds on distortion that we establish as a guide on how many ratings each user should provide to a recommendation system before its predictions can be trusted.

The simple setting in our work serves as a context for the initial development of our idea, and can be extended in multiple ways. One direction is to study the robustness of collaborative filtering algorithms as measured by alternative metrics. One metric could be, for instance, a user’s utility loss due to manipulation. Another extension is to design algorithms that provide non-asymptotic guarantees on both prediction accuracy and robustness.

The framework that we establish also facilitates studying the effectiveness of alternative techniques to abate influence by manipulators. For instance, given a scheme that incentivizes users to inspect and rate products, one could analyze how honest users and manipulators would behave, and then use our distortion metrics to assess the robustness of the scheme to manipulation.

It is also worth mentioning that many commercial recommendation systems build on multiple sources of information, not just collaborative filtering (Adomavicius and Tuzhilin, 2005). For example, as discussed in Balabanovic and Shoham (1997), recommendations should also be guided by features of the products being recommended.

Acknowledgments

The authors thank Christina Aperjis, Thomas Cover, Paul Cuff, Amir Dembo, Persi Diaconis, Vivek Farias, John Gill, Ramesh Johari, Yi-hao Kao, Yi Lu, Taesup Moon, Beomsoo Park, and Assaf Zeevi for helpful suggestions. This research was supported in part by the National Science Foundation through grant IIS-0428868.

References

- Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, 2005.
- Marko Balabanovic and Yoav Shoham. Fab: Content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72, 1997.
- James Bennett. The Cinematch system: operation, scale coverage, accuracy impact. <http://blog.recommenders06.com/wp-content/uploads/2006/09/1jimbennett.wmv>, 2006.
- Rajat Bhattacharjee and Ashish Goel. Algorithms and incentives for robust ranking. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2007.
- John Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence*, 1998.
- Robin Burke, Bamshad Mobasher, Roman Zabicki, and Runa Bhaumik. Limited knowledge shilling attacks in collaborative filtering systems. In *Proceedings of the Third IJCAI Workshop in Intelligent Techniques for Personalization*, 2005.
- Peter Cheeseman and John Stutz. Bayesian classification (AutoClass): Theory and results. In Usama Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 153–180. AAAI, 1996.
- Chrysanthos Dellarocas. Strategic manipulation of internet opinion forums: Implications for consumers and firms. *Management Science*, 52(10):1577–1593, 2006.
- Arthur Dempster, Nan Laird, and Donald Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.
- Pedro Domingos and Michael Pazzani. On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, 29(2):103–130, 1997.
- Petros Drineas, Iordanis Kerenidis, and Prabhakar Raghavan. Competitive recommendation systems. In *Proceedings of the Thirty-fourth Annual ACM Symposium on Theory of Computing*. ACM, 2002.
- Eric Friedman, Paul Resnick, and Rahul Sami. Manipulation-resistant reputation systems. In Noam Nisan, Tim Roughgarden, Éva Tardos, and Vijay Vazirani, editors, *Algorithmic Game Theory*, chapter 27, pages 677–697. Cambridge University Press, 2007.
- Olivier Gossner and Tristan Tomala. Entropy bounds on Bayesian learning. *Journal of Mathematical Economics*, 44:24–32, 2008.

- Jonathan Herlocker, Joseph Konstan, Al Borchers, and John Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the Twenty-second Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 1999.
- George John and Pat Langley. Estimating continuous distributions in Bayesian classifiers. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann, 1995.
- John Kleinberg and Mark Sandler. Using mixture models for collaborative filtering. *Computer and System Sciences*, 74(1):49–69, 2008.
- Shyong Lam and John Riedl. Shilling recommender systems for fun and profit. In *Proceedings of the Thirteenth International Conference on World Wide Web*. ACM, 2004.
- Greg Linden, Brent Smith, and Jeremy York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.
- Paolo Massa and Paolo Avesani. Trust-aware collaborative filtering for recommender systems. In *Proceedings of the Eleventh International Conference on Intelligent User Interfaces*. ACM, 2008.
- Bhaskar Mehta. Unsupervised shilling detection for collaborative filtering. In *Proceedings of the Twenty-Second Conference on Artificial Intelligence*. AAAI, 2007.
- Bhaskar Mehta and Wolfgang Nejdl. Attack resistant collaborative filtering. In *Proceedings of the Thirty-First Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2008.
- Nolan Miller, Paul Resnick, and Richard Zeckhauser. Eliciting informative feedback: The peer-prediction method. *Management Science*, 51(9):1359–1373, 2005.
- Bamshad Mobasher, Robin Burke, Runa Bhaumik, and Chad Williams. Effective attack models for shilling item-based collaborative filtering systems. In *Proceedings of the 2005 WebKDD Workshop*. ACM, 2005.
- Bamshad Mobasher, Robin Burke, and JJ Sandvig. Model-based collaborative filtering as a defense against profile injection attacks. In *Proceedings of the Twenty-first National Conference on Artificial Intelligence*. AAAI, 2006.
- Bamshad Mobasher, Robin Burke, Runa Bhaumik, and Chad Williams. Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Transactions on Internet Technology*, 7(4), 2007.
- Sangkil Moon and Gary Russell. Predicting product purchase from inferred customer similarity: an autologistic model approach. *Management Science*, 54(1):71–82, 2008.
- Rajeev Motwani and Sergei Vassilvitskii. Tracing the path: new model and algorithms for collaborative filtering. In *Twenty-third International Conference on Data Engineering Workshop*. IEEE, 2007.
- Netflix Prize. <http://www.netflixprize.com>, 2006.
- John O'Donovan and Barry Smyth. Is trust robust? An analysis of trust-based recommendation. In *Proceedings of the Eleventh International Conference on Intelligent User Interfaces*. ACM, 2006.
- Stefanie Olsen. Amazon blushes over sex link gaffe. <http://news.cnet.com/2100-1023-976435.html>, 2002.
- Michael O'Mahony, Neil Hurley, Nicholas Kushmerick, and Guenole Silvestre. Collaborative recommendation: A robustness analysis. *ACM Transactions on Internet Technologies*, 4(4):344–377, 2004.
- Paul Resnick and Rahul Sami. The Influence Limiter: provably manipulation-resistant recommender systems. In *Proceedings of the ACM Recommender Systems Conference*. ACM, 2007.

Paul Resnick and Rahul Sami. The information cost of manipulation-resistance in recommender systems. In *Proceedings of the Second ACM Recommender Systems Conference*. ACM, 2008.

Lucas Ryan. Personal communication, 2008.

JJ Sandvig, Bamshad Mobasher, and Robin Burke. Robustness of collaborative recommendation based on association rule mining. In *Proceedings of the 2007 ACM Conference on Recommender Systems*. ACM, 2007.

Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the Tenth International Conference on World Wide Web*. ACM, 2001.

J. Ben Schafer, Joseph Konstan, and John Riedl. E-commerce recommendation applications. *Data Mining and Knowledge Discovery*, 5(1):115–153, 2001.

Abraham Wald. Note on the consistency of the maximum likelihood estimate. *The Annals of Mathematical Statistics*, 20(4):595–601, 1949.

Chad Williams, Bamshad Mobasher, and Robin Burke. Defending recommender systems: Detection of profile injection attacks. *Journal of Service Oriented Computing and Applications*, 1(3), 2007.

Sheng Zhang, Yi Ouyang, James Ford, and Fillia Makedon. Analysis of a low-dimensional linear model under recommendation attacks. In *Proceedings of the Twenty-Ninth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2006.

A Proofs

A.1 Relationships Among Distortion Measures

Propositions 1 and 2 characterize relationships among KL distortion, RMS distortion, and the probability of correct binary predictions. These results are somewhat standard in flavor and are straightforward to prove. We include them for completeness. Lemma 1 is invoked in the proof of Proposition 2.

Proposition 1. For all $N, P, M, r \in \{0, 1/M, \dots, (M-1)/M\}$, $Y \in S^{N \times (1-r)M}$, $Z \in S^{N \times rM}$, $\nu \in \sigma_N$, and $n \in \{1, \dots, N\}$,

$$d_n^{\text{RMS}}(P, \nu, Y, Z) \leq \sqrt{\frac{1}{2} d_n^{\text{KL}}(P, \nu, Y, Z)}.$$

Proof. Recall that $\hat{x}_{\nu_k, x^{k-1}, Y}$ and $\hat{x}_{\nu_k, x^{k-1}, (Y, Z)}$ denote the means of PMFs $P(\cdot | x^{k-1}, Y, \nu)$ and $P(\cdot | x^{k-1}, (Y, Z), \nu)$, respectively. It is well-known that means are bounded by total variational distance between respective PMFs:

$$\begin{aligned} \left| \hat{x}_{\nu_k, x^{k-1}, Y} - \hat{x}_{\nu_k, x^{k-1}, (Y, Z)} \right| &\leq \max_{A \subseteq S^N} \left| P(x \in A | x^{k-1}, Y, \nu) - P(x \in A | x^{k-1}, (Y, Z), \nu) \right| \\ &= \frac{1}{2} \left\| P(\cdot | x^{k-1}, Y, \nu) - P(\cdot | x^{k-1}, (Y, Z), \nu) \right\|_1. \end{aligned}$$

It follows that

$$\begin{aligned}
d_n^{\text{RMS}}(P, \nu, Y, Z) &= \sqrt{\frac{1}{n} \sum_{k=1}^n \mathbb{E} \left[\left(\hat{x}_{\nu_k, x^{k-1}, Y} - \hat{x}_{\nu_k, x^{k-1}, (Y, Z)} \right)^2 \right]} \\
&\leq \sqrt{\frac{1}{n} \sum_{k=1}^n \mathbb{E} \left[\left(\frac{1}{2} \|P(\cdot | x^{k-1}, Y, \nu) - P(\cdot | x^{k-1}, (Y, Z), \nu)\|_1 \right)^2 \right]} \\
&\leq \sqrt{\frac{1}{2n} \sum_{k=1}^n \mathbb{E} [D(P(\cdot | x^{k-1}, Y, \nu) \| P(\cdot | x^{k-1}, (Y, Z), \nu))] } \\
&= \sqrt{\frac{1}{2} d_n^{\text{KL}}(P, \nu, Y, Z)},
\end{aligned}$$

where the second inequality follows from Pinsker's inequality. \square

Lemma 1. Consider a Bernoulli random variable X and discrete random variables W_1 and W_2 . Let \hat{X}_1 and \hat{X}_2 be the maximum a posteriori estimate of X upon observing W_1 and W_2 , respectively. That is,

$$\hat{X}_1 = \operatorname{argmax}_{x \in \{0,1\}} \Pr(X = x | W_1) \quad \text{and} \quad \hat{X}_2 = \operatorname{argmax}_{x \in \{0,1\}} \Pr(X = x | W_2).$$

Then,

$$\Pr(\hat{X}_1 = X) - \Pr(\hat{X}_2 = X) \leq \sqrt{\mathbb{E}[(\mathbb{E}[X | W_1] - \mathbb{E}[X | W_2])^2]}.$$

Proof.

$$\begin{aligned}
&\Pr(\hat{X}_1 = X) - \Pr(\hat{X}_2 = X) \\
&= \sum_{w_1} \Pr(W_1 = w_1) \max_{x \in \{0,1\}} \Pr(X = x | W_1 = w_1) - \sum_{w_2} \Pr(W_2 = w_2) \max_{x \in \{0,1\}} \Pr(X = x | W_2 = w_2) \\
&= \sum_{w_1, w_2} \Pr(W_1 = w_1, W_2 = w_2) \left(\max_x \Pr(X = x | W_1 = w_1) - \max_x \Pr(X = x | W_2 = w_2) \right) \\
&\leq \sum_{w_1, w_2} \Pr(W_1 = w_1, W_2 = w_2) |\Pr(X = 1 | W_1 = w_1) - \Pr(X = 1 | W_2 = w_2)| \\
&\leq \sqrt{\sum_{w_1, w_2} \Pr(W_1 = w_1, W_2 = w_2) (\Pr(X = 1 | W_1 = w_1) - \Pr(X = 1 | W_2 = w_2))^2} \\
&= \sqrt{\mathbb{E}[(\mathbb{E}[X | W_1] - \mathbb{E}[X | W_2])^2]},
\end{aligned}$$

where the second inequality follows from Jensen's inequality. \square

Proposition 2. Let $\mathbf{S} = \{0, 1\}$. For all $N, P, M, r \in \{0, 1/M, \dots, (M-1)/M\}$, $Y \in S^{N \times (1-r)M}$,

$Z \in S^{N \times rM}$, $\nu \in \sigma_N$, and $n \in \{1, \dots, N\}$,

$$\frac{1}{n} \sum_{k=1}^n \left(\Pr(\mathbf{x}_{\nu_k} = \mathbf{1}(\hat{x}_{\nu_k, x^{k-1}, Y} > 1/2)) - \Pr(\mathbf{x}_{\nu_k} = \mathbf{1}(\hat{x}_{\nu_k, x^{k-1}, (Y, Z)} > 1/2)) \right) \leq d_n^{\text{RMS}}(P, \nu, Y, Z).$$

Proof. We have

$$\begin{aligned} & \frac{1}{n} \sum_{k=1}^n \left(\Pr(\mathbf{x}_{\nu_k} = \mathbf{1}(\hat{x}_{\nu_k, x^{k-1}, Y} > 1/2)) - \Pr(\mathbf{x}_{\nu_k} = \mathbf{1}(\hat{x}_{\nu_k, x^{k-1}, (Y, Z)} > 1/2)) \right) \\ & \leq \sqrt{\frac{1}{n} \sum_{k=1}^n \left(\Pr(\mathbf{x}_{\nu_k} = \mathbf{1}(\hat{x}_{\nu_k, x^{k-1}, Y} > 1/2)) - \Pr(\mathbf{x}_{\nu_k} = \mathbf{1}(\hat{x}_{\nu_k, x^{k-1}, (Y, Z)} > 1/2)) \right)^2} \\ & \leq \sqrt{\frac{1}{n} \sum_{k=1}^n \mathbb{E} \left[\left(\hat{x}_{\nu_k, x^{k-1}, Y} - \hat{x}_{\nu_k, x^{k-1}, (Y, Z)} \right)^2 \right]} \\ & = d_n^{\text{RMS}}(P, \nu, Y, Z). \end{aligned}$$

The first inequality follows from Jensen's inequality and the second inequality follows from Lemma 1. \square

A.2 Results on Linear and Asymptotically Linear CF Algorithms

Theorem 1 establishes a distortion bound for linear CF algorithms. Theorem 2 establishes a distortion bound for asymptotically linear CF algorithms. Lemmas 2 and 3 are invoked in its proof. Theorem 3 establishes a relationship between consistent and asymptotically linear CF algorithms. Lemmas 4, 5, and 6 are invoked in its proof.

Theorem 1. For all $N, P, M, r \in \{0, 1/M, \dots, (M-1)/M\}$, $Y \in S^{N \times (1-r)M}$, $Z \in S^{N \times rM}$, $\nu \in \sigma_N$, and $n \in \{1, \dots, N\}$,

$$d_n^{\text{KL}}(P, \nu, Y, Z) \leq \frac{1}{n} \ln \frac{1}{1-r}.$$

Proof. We have

$$\begin{aligned} d_n^{\text{KL}}(P, \nu, Y, Z) &= \frac{1}{n} \sum_{k=1}^n \mathbb{E} \left[D \left(P(\cdot | x^{k-1}, Y, \nu) \parallel P(\cdot | x^{k-1}, (Y, Z), \nu) \right) \right] \\ &\leq \frac{1}{n} \sum_{k=1}^N \mathbb{E} \left[D \left(P(\cdot | x^{k-1}, Y, \nu) \parallel P(\cdot | x^{k-1}, (Y, Z), \nu) \right) \right] \\ &= \frac{1}{n} D(P(\cdot | Y, \nu) \parallel P(\cdot | (Y, Z), \nu)). \end{aligned}$$

The last equality follows from the chain rule of KL divergence: that is, given two joint distributions

s and t over random variables z_1, \dots, z_n ,

$$D(s \| t) = \sum_{k=1}^n \mathbb{E} [D(s(z_k | z_1, \dots, z_{k-1}) \| t(z_k | z_1, \dots, z_{k-1}))],$$

where expectation is taken with z_1, \dots, z_n distributed according to s .

For any $\mathbf{x} \in \mathbf{S}^N$, since P is linear, we have

$$\frac{P(\mathbf{x} | Y)}{P(\mathbf{x} | (Y, Z))} = \frac{P(\mathbf{x} | Y)}{(1-r)P(\mathbf{x} | Y) + rP(\mathbf{x} | Z)} \leq \frac{1}{1-r}.$$

Then,

$$D(P(\cdot | Y) \| P(\cdot | (Y, Z))) = \sum_{\mathbf{x} \in \mathbf{S}^N} P(\mathbf{x} | Y) \ln \frac{P(\mathbf{x} | Y)}{P(\mathbf{x} | (Y, Z))} \leq \sum_{\mathbf{x} \in \mathbf{S}^N} P(\mathbf{x} | Y) \ln \frac{1}{1-r} = \ln \frac{1}{1-r}.$$

Hence,

$$d_n^{\text{KL}}(P, \nu, Y, Z) \leq \frac{1}{n} D(P(\cdot | Y) \| P(\cdot | (Y, Z))) \leq \frac{1}{n} \ln \frac{1}{1-r}.$$

□

The first portion of the above proof, which bounds KL distortion by KL divergence divided by n , uses arguments identical along the lines of Gossner and Tomala (2008).

Lemma 2. *Let $\{\mu_m\}$ and $\{\nu_m\}$ be two sequences of random PMFs over a fixed finite sample space Ω . If for all $\epsilon > 0$, $\Pr(D(\mu_m \| \nu_m) \geq \epsilon) \rightarrow 0$, then for all $\epsilon > 0$,*

$$\Pr\left(\sum_{\omega \in \Omega} \mu_m(\omega) \left| \log \frac{\mu_m(\omega)}{\nu_m(\omega)} \right| \geq \epsilon\right) \rightarrow 0.$$

Proof. For $\epsilon > 0$ and m , we denote by $A_{m,\epsilon}$ the event that for all $\omega \in \Omega$, at least one of the following holds: $|\log(\mu_m(\omega)/\nu_m(\omega))| \leq \epsilon$ and $\max\{\mu_m(\omega), \nu_m(\omega)\} \leq \epsilon$. We now prove that for all $\epsilon > 0$, $\Pr(A_{m,\epsilon}) \rightarrow 1$. To see this, for any given $\epsilon > 0$, we let δ_ϵ be in $(0, \epsilon(1 - e^{-\epsilon}))$ and denote by B_{m,δ_ϵ} the event that for all $\omega \in \Omega$, $|\mu_m(\omega) - \nu_m(\omega)| \leq \delta_\epsilon$. We now show that $B_{m,\delta_\epsilon} \subset A_{m,\epsilon}$. If $|\mu_m(\omega) - \nu_m(\omega)| \leq \delta_\epsilon, \forall \omega$, then for any ω' such that $\max\{\mu_m(\omega'), \nu_m(\omega')\} > \epsilon$, we have $\min\{\mu_m(\omega'), \nu_m(\omega')\} > \epsilon - \delta_\epsilon$. This implies

$$\begin{aligned} \left| \log \frac{\mu_m(\omega')}{\nu_m(\omega')} \right| &= \max \left\{ \log \frac{\mu_m(\omega')}{\nu_m(\omega')}, \log \frac{\nu_m(\omega')}{\mu_m(\omega')} \right\} \\ &\leq \max \left\{ \log \left(\frac{|\mu_m(\omega') - \nu_m(\omega')|}{\nu_m(\omega')} + 1 \right), \log \left(\frac{|\nu_m(\omega') - \mu_m(\omega')|}{\mu_m(\omega')} + 1 \right) \right\} \\ &< \log \left(\frac{\delta_\epsilon}{\epsilon - \delta_\epsilon} + 1 \right) \\ &\leq \epsilon, \end{aligned}$$

where the last inequality follows from our choice of δ_ϵ . Hence, $B_{m,\delta_\epsilon} \subset A_{m,\epsilon}$ and for any ϵ and a corresponding δ_ϵ , we have $\Pr(A_{m,\epsilon}) \geq \Pr(B_{m,\delta_\epsilon}) \rightarrow 1$, where convergence follows from Pinsker's inequality.

For each m and each realization of μ_m and ν_m , we let $\Omega_m = \{\omega \in \Omega : \mu_m(\omega) \leq \nu_m(\omega)\}$, let

$$\tau_m = \sum_{\omega \in \Omega_m} \mu_m(\omega) \log \frac{\mu_m(\omega)}{\nu_m(\omega)},$$

and for $\epsilon > 0$, denote by $C_{m,\epsilon}$ the event that $|\tau_m| \leq \epsilon$. We now show that for any $\epsilon > 0$, $A_{m,\gamma_\epsilon} \subset C_{m,\epsilon}$ where $\gamma_\epsilon \in (0, \min\{\epsilon/|\Omega|, 1/e\}]$ satisfies $\gamma_\epsilon |\log \gamma_\epsilon| \leq \epsilon/|\Omega|$. To see this, we first let $\Omega_{m,\gamma_\epsilon}^1 = \{\omega \in \Omega : |\log(\mu_m(\omega)/\nu_m(\omega))| \leq \gamma_\epsilon\}$ and $\Omega_{m,\gamma_\epsilon}^2 = \{\omega \in \Omega : \max\{\mu_m(\omega), \nu_m(\omega)\} \leq \gamma_\epsilon\} \setminus \Omega_{m,\gamma_\epsilon}^1$. Note that $\Omega_{m,\gamma_\epsilon}^1 \cap \Omega_{m,\gamma_\epsilon}^2 = \emptyset$. If for all $\omega \in \Omega$, $|\log(\mu_m(\omega)/\nu_m(\omega))| \leq \gamma_\epsilon$ or $\max\{\mu_m(\omega), \nu_m(\omega)\} \leq \gamma_\epsilon$, then $\Omega_{m,\gamma_\epsilon}^1 \cup \Omega_{m,\gamma_\epsilon}^2 = \Omega$. This implies

$$\begin{aligned} |\tau_m| &= \sum_{\omega \in \Omega_m \cap \Omega_{m,\gamma_\epsilon}^1} \mu_m(\omega) \left| \log \frac{\mu_m(\omega)}{\nu_m(\omega)} \right| + \sum_{\omega \in \Omega_m \cap \Omega_{m,\gamma_\epsilon}^2} \mu_m(\omega) \left| \log \frac{\mu_m(\omega)}{\nu_m(\omega)} \right| \\ &\leq |\Omega_{m,\gamma_\epsilon}^1| \gamma_\epsilon + \sum_{\omega \in \Omega_m \cap \Omega_{m,\gamma_\epsilon}^2} \mu_m(\omega) |\log \mu_m(\omega)| \\ &\leq |\Omega_{m,\gamma_\epsilon}^1| \gamma_\epsilon + |\Omega_{m,\gamma_\epsilon}^2| \gamma_\epsilon |\log \gamma_\epsilon| \\ &\leq |\Omega_{m,\gamma_\epsilon}^1| \frac{\epsilon}{|\Omega|} + |\Omega_{m,\gamma_\epsilon}^2| \frac{\epsilon}{|\Omega|} \\ &= \epsilon. \end{aligned}$$

The first inequality follows from the definitions of $\Omega_{m,\gamma_\epsilon}^1$ and Ω_m . The second inequality follows from the definition of $\Omega_{m,\gamma_\epsilon}^2$ and that $\gamma_\epsilon \leq 1/e$. The third inequality follows from other constraints on γ_ϵ . Hence, for any $\epsilon > 0$ and a corresponding γ_ϵ satisfying the aforesaid constraints, $A_{m,\gamma_\epsilon} \subset C_{m,\epsilon}$.

Note that for all m and all realizations of μ_m and ν_m ,

$$\sum_{\omega \in \Omega} \mu_m(\omega) \left| \log \frac{\mu_m(\omega)}{\nu_m(\omega)} \right| = D(\mu_m \| \nu_m) - 2\tau_m.$$

Hence, for any $\epsilon > 0$, a corresponding γ_ϵ , and any m ,

$$\begin{aligned} &\Pr \left(\sum_{\omega \in \Omega} \mu_m(\omega) \left| \log \frac{\mu_m(\omega)}{\nu_m(\omega)} \right| \geq 3\epsilon \right) \\ &= \Pr(D(\mu_m \| \nu_m) - 2\tau_m \geq 3\epsilon, A_{m,\gamma_\epsilon}^c) + \Pr(D(\mu_m \| \nu_m) - 2\tau_m \geq 3\epsilon, A_{m,\gamma_\epsilon}) \\ &\leq \Pr(A_{m,\gamma_\epsilon}^c) + \Pr(D(\mu_m \| \nu_m) - 2\tau_m \geq 3\epsilon, C_{m,\epsilon}) \\ &\leq \Pr(A_{m,\gamma_\epsilon}^c) + \Pr(D(\mu_m \| \nu_m) \geq \epsilon) \rightarrow 0. \end{aligned}$$

Here, A_{m,γ_ϵ}^c denotes the complement of A_{m,γ_ϵ} . The last inequality follows from the definition of

$C_{m,\epsilon}$. Convergence follows from our original assumption and that $\Pr(A_{m,\gamma_\epsilon}) \rightarrow 1$. \square

Lemma 3. *Let $\{\mu_m\}$, $\{\nu_m\}$, and $\{\chi_m\}$ be three sequences of random PMFs over a fixed finite sample space Ω . Suppose that for all $\epsilon > 0$, $\Pr(D(\mu_m \parallel \nu_m) \geq \epsilon) \rightarrow 0$. Further, suppose there exists $b > 0$ such that for all m , realizations of χ_m and μ_m , and $\omega \in \Omega$, $\chi_m(\omega)/\mu_m(\omega) \leq b$. Then, for all $\epsilon > 0$, $\Pr(|D(\chi_m \parallel \mu_m) - D(\chi_m \parallel \nu_m)| \geq \epsilon) \rightarrow 0$.*

Proof. For any $\epsilon > 0$ and m ,

$$\begin{aligned} \Pr(|D(\chi_m \parallel \mu_m) - D(\chi_m \parallel \nu_m)| \geq \epsilon) &= \Pr\left(\left|\sum_{\omega \in \Omega} \chi_m(\omega) \log \frac{\mu_m(\omega)}{\nu_m(\omega)}\right| \geq \epsilon\right) \\ &\leq \Pr\left(\sum_{\omega \in \Omega} \frac{\chi_m(\omega)}{\mu_m(\omega)} \left|\mu_m(\omega) \log \frac{\mu_m(\omega)}{\nu_m(\omega)}\right| \geq \epsilon\right) \\ &\leq \Pr\left(b \sum_{\omega \in \Omega} \left|\mu_m(\omega) \log \frac{\mu_m(\omega)}{\nu_m(\omega)}\right| \geq \epsilon\right) \rightarrow 0, \end{aligned}$$

where convergence follows from Lemma 2. \square

Theorem 2. *For all N, Ψ, P asymptotically linear with respect to Ψ , $\psi, \phi \in \Psi$, $r \in [0, 1)$, $\nu \in \sigma_N$, $n \in \{1, \dots, N\}$, and $\epsilon > 0$,*

$$\lim_{m \rightarrow \infty} \Pr\left(d_n^{\text{KL}}(P, \nu, Y_m, Z_m) \geq \frac{1}{n} \ln \frac{1}{1-r} + \epsilon\right) = 0,$$

where, for each m , $Y_m = (y^1, \dots, y^{m-l}) \in \mathcal{S}^{N \times (m-l)}$, $Z_m = (z^1, \dots, z^l) \in \mathcal{S}^{N \times l}$, $l \sim \text{Binomial}(m, r)$, and $y^1, \dots, y^{m-l} \sim \psi$ and $z^1, \dots, z^l \sim \phi$ are i.i.d. sequences.

Proof. Let finite sample space $\Omega = \mathbf{S}^N$. For each m , let random PMFs $\mu_m = (1-r)P(\cdot | Y_m) + rP(\cdot | Z_m)$, $\nu_m = P(\cdot | (Y_m, Z_m))$, and $\chi_m = P(\cdot | Y_m)$. By the definition of asymptotically linear CF algorithms, for all $\epsilon > 0$, $\Pr(D(\mu_m \parallel \nu_m) \geq \epsilon) \rightarrow 0$. Clearly, for all m , realizations of χ_m and μ_m , and $\mathbf{s} \in \mathbf{S}^N$, $\chi_m(\mathbf{s})/\mu_m(\mathbf{s}) \leq 1/(1-r)$. Hence, for all $\epsilon > 0$,

$$\begin{aligned} &\Pr\left(d_n^{\text{KL}}(P, \nu, Y_m, Z_m) \geq \frac{1}{n} \ln \frac{1}{1-r} + \epsilon\right) \\ &\leq \Pr\left(D(P(\cdot | Y_m) \parallel P(\cdot | (Y_m, Z_m))) \geq \ln \frac{1}{1-r} + n\epsilon\right) \\ &\leq \Pr\left(D(P(\cdot | Y_m) \parallel P(\cdot | (Y_m, Z_m))) \geq D(P(\cdot | Y_m) \parallel (1-r)P(\cdot | Y_m) + rP(\cdot | Z_m)) + n\epsilon\right) \\ &\leq \Pr\left(|D(\chi_m \parallel \mu_m) - D(\chi_m \parallel \nu_m)| \geq n\epsilon\right) \rightarrow 0. \end{aligned}$$

The second inequality holds because $D(P(\cdot | Y_m) \parallel (1-r)P(\cdot | Y_m) + rP(\cdot | Z_m)) \leq \ln(1/(1-r))$ and convergence follows from Lemma 3. \square

Lemma 4. Let $U \subset \mathfrak{R}^k$ be a compact set. Consider a fixed vector $u \in U$ and a sequence of random vectors $\{u_m\}$ for which $\Pr(u_m \in U) \rightarrow 1$. For any continuous function $f : U \rightarrow \mathfrak{R}$, if $\Pr(\|u_m - u\|_1 \geq \epsilon) \rightarrow 0$ for all $\epsilon > 0$, then $\Pr(|f(u_m) - f(u)| \geq \epsilon) \rightarrow 0$ for all $\epsilon > 0$.

Proof. Because the continuous function f defined on compact set U is uniformly continuous, for each $\epsilon > 0$, there exists $\delta > 0$ such that for any $v, v' \in U$, $\|v - v'\|_1 \leq \delta$ implies that $|f(v) - f(v')| \leq \epsilon$. Hence, for all $\epsilon > 0$,

$$\begin{aligned} \Pr(|f(u_m) - f(u)| \leq \epsilon) &\geq \Pr(|f(u_m) - f(u)| \leq \epsilon, u_m \in U) \\ &\geq \Pr(\|u_m - u\|_1 \leq \delta, u_m \in U) \rightarrow 1. \end{aligned}$$

□

Lemma 5. Fix a finite sample space Ω . Let $\{\mu_m\}$ and $\{\nu_m\}$ be two sequences of random PMFs over Ω and let μ be a fixed PMF over Ω . If for all $\epsilon > 0$, $\Pr(D(\mu_m \| \mu) \geq \epsilon) \rightarrow 0$ and $\Pr(D(\nu_m \| \mu) \geq \epsilon) \rightarrow 0$, then for all $\epsilon > 0$, $\Pr(D(\mu_m \| \nu_m) \geq \epsilon) \rightarrow 0$.

Proof. We first identify the support of μ . Without loss of generality, we let $\mu(\omega_i) = 0, \forall 1 \leq i \leq l$ and $\mu(\omega_i) > \delta, \forall l < i \leq |\Omega|$ for some $l \geq 0$ and $\delta > 0$. In the following, we represent PMFs as vectors in $\mathfrak{R}^{|\Omega|}$. To this end, we define a set $T = \{(t_1, \dots, t_{|\Omega|}) : \sum_i t_i = 1, \forall i \leq l, t_i = 0, \forall j > l, t_j \geq \delta\}$. Let compact set $U = T \times T$. We let $u = (\mu, \mu) \in U$ and define a sequence of random vectors $\{u_m\}$ where each $u_m = (\mu_m, \nu_m)$. Let continuous function $f : U \rightarrow \mathfrak{R}$ be the KL divergence $D(\cdot \| \cdot)$. By examining the absolute continuity of μ_m and ν_m with respect to μ and applying Pinsker's inequality, we have $\Pr(u_m \in U) \rightarrow 1$ and further, for all $\epsilon > 0$, $\Pr(\|u_m - u\|_1 \geq \epsilon) \rightarrow 0$. Hence, for all $\epsilon > 0$, by Lemma 4, $\Pr(D(\mu_m \| \nu_m) \geq \epsilon) = \Pr(|D(\mu_m \| \nu_m) - D(\mu \| \mu)| \geq \epsilon) = \Pr(|f(u_m) - f(u)| \geq \epsilon) \rightarrow 0$. □

Lemma 6. Fix a finite sample space Ω . Let $\{\mu_m\}$ and $\{\nu_m\}$ be two sequences of random PMFs over Ω and let μ and ν be two fixed PMFs over Ω . If for all $\epsilon > 0$, $\Pr(D(\mu_m \| \mu) \geq \epsilon) \rightarrow 0$ and $\Pr(D(\nu_m \| \nu) \geq \epsilon) \rightarrow 0$, then for all $r \in [0, 1]$ and $\epsilon > 0$,

$$\Pr(D(((1-r)\mu_m + r\nu_m) \| ((1-r)\mu + r\nu)) \geq \epsilon) \rightarrow 0.$$

Proof. The proof here is similar to that for Lemma 5. For a fixed r , let PMF $\chi = (1-r)\mu + r\nu$. Without loss of generality, we let $\chi(\omega_i) = 0, \forall 1 \leq i \leq l$ and $\chi(\omega_i) > \delta, \forall l < i \leq |\Omega|$ for some $l \geq 0$ and $\delta > 0$. In the following, we represent PMFs as vectors in $\mathfrak{R}^{|\Omega|}$. To this end, we define a set $T = \{(t_1, \dots, t_{|\Omega|}) : \sum_i t_i = 1, \forall i \leq l, t_i = 0, \forall j > l, t_j \geq \delta\}$. Let compact set $U = T \times T$. We let $u = ((1-r)\mu + r\nu, (1-r)\mu + r\nu) \in U$ and define a sequence of random vectors $\{u_m\}$ where each $u_m = ((1-r)\mu_m + r\nu_m, (1-r)\mu + r\nu)$. Let continuous function $f : U \rightarrow \mathfrak{R}$ be the KL divergence $D(\cdot \| \cdot)$. By examining the absolute continuity of μ_m with respect to μ and that of ν_m

with respect to ν and applying Pinsker's inequality, we have $\Pr(u_m \in U) \rightarrow 1$, and for all $\epsilon > 0$, $\Pr(\|u_m - u\|_1 \geq \epsilon) \rightarrow 0$. Hence, for all $\epsilon > 0$, by Lemma 4,

$$\begin{aligned}
& \Pr(D(((1-r)\mu_m + r\nu_m) \| ((1-r)\mu + r\nu)) \geq \epsilon) \\
&= \Pr(|D(((1-r)\mu_m + r\nu_m) \| ((1-r)\mu + r\nu)) - D(((1-r)\mu + r\nu) \| ((1-r)\mu + r\nu))| \geq \epsilon) \\
&= \Pr(|f(u_m) - f(u)| \geq \epsilon) \rightarrow 0.
\end{aligned}$$

□

Theorem 3. *Any probabilistic CF algorithm consistent with respect to an identifiable and convex set Φ is asymptotically linear with respect to Φ .*

Proof. We use the notation in the definition of asymptotically linear CF algorithms in Section 4.3 and Lemmas 5 and 6. Fix an algorithm P consistent with respect to Φ . Let finite sample space $\Omega = \mathbf{S}^N$. For each m , let random PMFs $\mu_m = P(\cdot | U_m)$, $\nu_m = P(\cdot | V_m)$, and $\chi_m = P(\cdot | (U_m, V_m))$. Let π_1 and π_2 be the type PMFs corresponding to ψ and ϕ , respectively. Let $\mu = \pi_1$, $\nu = \pi_2$. Fix $r \in [0, 1]$. By the consistency of P and convexity of Φ , we have for all $\epsilon > 0$, $\Pr(D(\mu_m \| \mu) \geq \epsilon) \rightarrow 0$, $\Pr(D(\nu_m \| \nu) \geq \epsilon) \rightarrow 0$, and $\Pr(D(\chi_m \| ((1-r)\mu + r\nu)) \geq \epsilon) \rightarrow 0$. By Lemma 6, for all $\epsilon > 0$, $\Pr(D(((1-r)\mu_m + r\nu_m) \| ((1-r)\mu + r\nu)) \geq \epsilon) \rightarrow 0$. Then for all $\epsilon > 0$,

$$\begin{aligned}
& \Pr(D(((1-r)P(\cdot | U_m) + rP(\cdot | V_m)) \| P(\cdot | (U_m, V_m))) \geq \epsilon) \\
&= \Pr(D(((1-r)\mu_m + r\nu_m) \| \chi_m) \geq \epsilon) \rightarrow 0,
\end{aligned}$$

where convergence follows from Lemma 5. □

A.3 Results on Kernel Density Estimation and Naive Bayes Algorithms

Results in this section pertain to KDE and NB algorithms.

Proposition 3. *Any KDE algorithm is a linear CF algorithm. Any linear CF algorithm is a KDE algorithm.*

Proof. Consider a KDE algorithm P . Given $W_1 \in S^{N \times M_1}$ and $W_2 \in S^{N \times M_2}$, for each $\mathbf{x} \in \mathbf{S}^N$,

$$\begin{aligned}
& P(\mathbf{x} | (W_1, W_2)) \\
&= \frac{1}{M_1 + M_2} \left(\sum_{w \in W_1} \mathcal{K}_w(\mathbf{x}) + \sum_{w \in W_2} \mathcal{K}_w(\mathbf{x}) \right) \\
&= \frac{M_1}{M_1 + M_2} \left(\frac{1}{M_1} \sum_{w \in W_1} \mathcal{K}_w(\mathbf{x}) \right) + \frac{M_2}{M_1 + M_2} \left(\frac{1}{M_2} \sum_{w \in W_2} \mathcal{K}_w(\mathbf{x}) \right) \\
&= \frac{M_1}{M_1 + M_2} P(\mathbf{x} | W_1) + \frac{M_2}{M_1 + M_2} P(\mathbf{x} | W_2).
\end{aligned}$$

Hence, P is linear.

To show the converse, consider a linear CF algorithm P' , which generates $P'(\cdot|W)$ based on W . A KDE algorithm where kernel \mathcal{K}_w is set equal to $P(\cdot|\{w\})$ for each $w \in W$ is equivalent to P' . \square

Proposition 4. Fix $q \in [0, 1)$. Let Φ_q be the set of all (π, ρ_q) pairs where user type distribution π over \mathbf{S}^N takes the form

$$\pi(\mathbf{w}) = \sum_{l=1}^L \eta_l \prod_{n=1}^N \theta_{l,n}^{(\mathbf{w}_n)}, \forall \mathbf{w} \in \mathbf{S}^N, \quad (4)$$

for $L \in \mathcal{Z}_+$, $\eta \in \Delta_L$, and $\theta_{l,n} \in \Delta_{|\mathbf{S}|}$ for all $1 \leq l \leq L, 1 \leq n \leq N$, where Δ_k denotes a k -dimensional unit simplex, and conditional distribution of ratings vectors ρ_q conditioned on types is given by the distribution

$$Q_q^*(w | \mathbf{w}) = \begin{cases} q^{\|\mathbf{w}\|_0} (1-q)^{N-\|\mathbf{w}\|_0} & \text{if } \mathbf{w} \rightarrow w, \\ 0 & \text{otherwise.} \end{cases}$$

Then, Φ_q is identifiable and convex. Further, the naive Bayes algorithm is consistent with respect to Φ_q .

Proof. To show that Φ_q is identifiable, we note that ρ_q is fixed and hence only need to establish that distinct π_1 and π_2 lead to distinct ratings vector PMFs $\pi_1 \rho_q$ and $\pi_2 \rho_q$. To see this, consider $\mathbf{w} \in \mathbf{S}^N$ such that $\pi_1(\mathbf{w}) \neq \pi_2(\mathbf{w})$. We have

$$(\pi_1 \rho_q)(\mathbf{w}) = \pi_1(\mathbf{w}) \rho_q(\mathbf{w}|\mathbf{w}) \neq \pi_2(\mathbf{w}) \rho_q(\mathbf{w}|\mathbf{w}) = (\pi_2 \rho_q)(\mathbf{w}).$$

To show that Φ_q is convex, consider arbitrary type PMFs π_1, π_2 whose corresponding pairs are in Φ_q . For each $\lambda \in [0, 1]$, their convex combination π_λ satisfies

$$\begin{aligned} \pi_\lambda(\mathbf{w}) &= \lambda \pi_1(\mathbf{w}) + (1-\lambda) \pi_2(\mathbf{w}) \\ &= \sum_{l=1}^{L_1} \lambda \eta_{1,l} \prod_{n=1}^N \theta_{1,l,n}^{(\mathbf{w}_n)} + \sum_{l=1}^{L_2} (1-\lambda) \eta_{2,l} \prod_{n=1}^N \theta_{2,l,n}^{(\mathbf{w}_n)} \\ &= \left(\sum_{l=1}^{L_1+L_2} \hat{\eta}_l \prod_{n=1}^N \hat{\theta}_{l,n}^{(\mathbf{w}_n)} \right) \end{aligned}$$

for each $\mathbf{w} \in \mathbf{S}^N$, where $\hat{\eta}_l = \lambda \eta_{1,l}$ for $l \leq L$, $\hat{\eta}_l = (1-\lambda) \eta_{2,l-L}$ for $l > L$, and for each n , $\hat{\theta}_{l,n} = \theta_{1,l,n}$ for $l \leq L$ and $\hat{\theta}_{l,n} = \theta_{2,l-L,n}$ for $l > L$. Hence, π_λ takes the form in (4). As such, $(\pi_\lambda, \rho_q) \in \Phi_q$ and Φ_q is convex.

We now show that the naive Bayes algorithm P is consistent with respect to Φ_q by using results in Wald (1949). We will use notation in the definition of consistent CF algorithms in Section 4.3. We also denote by P^* the true type PMF over \mathbf{S}^N and let $\mathcal{P} = \{\pi : (\pi, \rho) \in \Phi_q\}$ be the set of all

type PMFs with corresponding pairs in Φ_q . According to Theorem 2 in Wald (1949), if

1. \mathbf{S}^N , \mathcal{P} , and P^* satisfy certain technical conditions specified in Wald (1949), and
2. There exists a constant $b > 0$ such that for all m and all realizations of $\{W_m\}$,

$$\frac{\prod_{w \in W_m} P(w | W_m)}{\prod_{w \in W_m} P^*(w)} \geq b,$$

then

$$\|P(\cdot | W_m) - P^*\|_1 \rightarrow 0, \text{ a.s.} \quad (5)$$

We verify that condition 1 holds in our problem instance and desire to find a b that satisfies condition 2. To do so, we denote by (L^*, η^*, θ^*) the parameters corresponding to P^* and by $(\hat{L}^{W_m}, \hat{\eta}^{W_m}, \hat{\theta}^{W_m})$ the parameters corresponding to $P(\cdot | W_m)$. Recall that we tune the parameter τ by cross validation over a range Γ . Let τ^* be the value that we settle at. We let

$$b = \left(\min_{\tau \in \Gamma} f_L^\tau(L^*) \right) \left(f_\eta(\eta^*) \prod_{l,n} f_\theta(\theta_{l,n}^*) \right).$$

Because $P(\cdot | W_m)$ maximizes the posterior PDF, for all m and all realizations of $\{W_m\}$, we have

$$\frac{\prod_{w \in W_m} P(w | W_m)}{\prod_{w \in W_m} P^*(w)} \geq \frac{f_L^{\tau^*}(L^*) f_\eta(\eta^*) \prod_{l,n} f_\theta(\theta_{l,n}^*)}{f_L^{\tau^*}(\hat{L}^{W_m}) f_\eta(\hat{\eta}^{W_m}) \prod_{l,n} f_\theta(\hat{\theta}_{l,n}^{W_m})} \geq b.$$

Hence, condition 2 holds, establishing (5). By the continuity of KL divergence and properties of almost sure convergence, for all $\epsilon > 0$, we have

$$\Pr(D(P(\cdot | W_m) \| P^*) \geq \epsilon) \rightarrow 0,$$

which implies that P is consistent with respect to Φ_q . □

B Table of Notation

| Notation | Description |
|----------------------------------|---|
| \mathbf{S} | Set of possible rating values. |
| S | Union of \mathbf{S} and the singleton set containing the circle \circ . |
| \mathbf{w} or \mathbf{x} | A user type. |
| w or x | A ratings vector. |
| x^k | Ratings vector in S^N that contains k ratings provided by an active user. |
| $p_{n,x,W}$ | PMF of the rating for product n , for a user with history x , based on training data W . |
| $P(\cdot W, \nu)$ | Type distribution generated by a CF algorithm based on training data W and product ordering ν . |
| σ_N | Set of permutations of $\{1, \dots, N\}$. |
| d^{KL} | KL distortion. |
| d^{RMS} | RMS distortion. |
| $\hat{x}_{n,x,W}$ | Scalar prediction of rating of product n for a user with history x , based on training data W . |
| \hat{d}^{RMS} | Empirical RMS distortion. |
| $\hat{\mathcal{E}}^{\text{RMS}}$ | Empirical RMS prediction error. |
| $\mathbf{w} \rightarrow w$ | For each n , either $w_n = \mathbf{w}_n$ or $w_n = \circ$. |
| $\ w\ _{\circ}$ | Number of circles in ratings vector w : $ \{n : w_n = \circ\} $. |