Robustness Analyses of Instance-Based Collaborative Recommendation

Nicholas Kushmerick

Computer Science Department, University College Dublin
nick@ucd.ie

Abstract. Collaborative recommendation has emerged as an effective technique for a personalized information access. However, there has been relatively little theoretical analysis of the conditions under which the technique is effective. We analyze the robustness of collaborative recommendation: the ability to make recommendations despite (possibly intentional) noisy product ratings. We formalize robustness in machine learning terms, develop two theoretically justified models of robustness, and evaluate the models on real-world data. Our investigation is both practically relevant for enterprises wondering whether collaborative recommendation leaves their marketing operations open to attack, and theoretically interesting for the light it sheds on a comprehensive theory of collaborative recommendation.

1 Introduction

Collaborative recommendation has emerged as an effective personalization technique for a diverse array of electronic commerce and information access scenarios (eg, [10,5]). Such systems keep track of their customers' preferences, and use these data to offer new suggestions. Many variations have been explored, but the basic idea is as follows: to recommended items to a target customer, the system retrieves similar customers, and then recommends items that were liked by the retrieved customers but not yet rated by the target.

Collaborative recommendation has been empirically validated for many domains (eg, [3]), and has been successfully deployed in many commercial settings. However, despite some interesting efforts [7,4,2], there is no general theoretical explanation of the conditions under which a particular collaborative recommendation application will succeed or fail.

Our goal is to complement existing theoretical work by investigating the robustness of collaborative recommendation. Informally, robustness measures how sensitive the technique is to changes in the customer/product rating matrix. In particular, we analyze the situation in which a malicious agent attacks a recommender system by posing as one or more customers and submitting bogus product ratings. Our analysis is designed to rigorously quantify the extent to which a malicious agent can force the recommender system to give poor recommendations to its "genuine" customers. The theoretical results reported in this paper builds on an ongoing empirical investigation of this issue [9,8].

Our primary motivation is to gain deeper insights into the principles underlying effective collaborative recommendation. However, our work is also relevant for a second, more practical reason: recommender systems can represent an insecure back-door into an enterprise's marketing operations. To lock this door, some enterprises impose substantial charges on customers to submit ratings (eg, a bookstore might only accept ratings for books that have been purchased). However, many collaborative recommenders are open Web services that malicious agents could easily attack. How much damage can they inflict?

We make three contributions. First, we formalize robustness in machine learnings terms, and introduce a novel form of class noise that models an interesting suite of attacks (Sec. 2). Second, we develop two models that predict the change in accuracy as a function the number of fake ratings that have been inserted into the customer/product matrix (Secs. 3–4). Third, we empirically evaluate our predications against real-world collaborative recommendation data (Sec. 5).

2 Definitions

Our analysis on collaborative recommendation assumes the standard k-NN learning algorithms. Other (eg, model-based) approaches have been tried but k-NN is accurate, widely used and easily analyzed. In this approach, each customer is represented as a vector of product ratings (many of which will be empty for any particular customer). Unlike traditional machine learning settings, the "class" is not a distinguished attribute, but corresponds to the product that the system is contemplating for recommendation.

We model an attack as the addition of noise to the training data. In general, this noise could be associated with either the attributes, the class or both. We focus exclusively on class noise, and defer attribute noise to future work. We are not concerned with malicious noise as defined by [6], because we can safely assume that the attacking agent is not omniscient (eg, the agents can not directly inspect any ratings except their own).

We model attacks with a relatively benign noise model that we call *biased class noise*. This model is characterized by the following parameters: the noise rate β, and the class bias μ. Noise is added according to the following process. First, an instance is generated according to the underlying distribution. With probability $1 - \beta$, the instance is noise-free and labeled by the target concept. Otherwise, with probability $\beta\mu$ the instance is labeled 1 and with probability $\beta(1 - \mu)$ the instance is labeled 0.

The biased class noise model is useful because it can represent a variety of stereotypical attacks. For example, a book's author could try to force recommendations of his book by pretending to be numerous customers who all happen to like the book. We call this a "push" attack and it corresponds to $\mu = 1$. Alternatively, the author's arch-enemy could insert fake customer profiles that all dislike the book; this "nuke" attack is modeled with $\mu = 0$.

We are interested in robustness: the ability of the recommender to make good recommendations in spite of an attack. There are two aspects to robustness. First, we may be concerned with *accuracy*: are the products recommended after the attack actually liked? The second issue is *stability*: does the system recommend different products after the attack (regardless of whether customers like them)?

While stability and accuracy are distinct, they are not orthogonal. For example, if a recommender has perfect accuracy for a given task both with and without noise, then it must be perfectly stable. On other the hand, consider a product that no-one likes. The recommendation policies "recommend to no-one" and "recommend to everyone" are both perfectly stable, yet the first is always correct and the second is always wrong. Our analysis of robustness focuses exclusively on accuracy.

Before proceeding, we introduce some additional notation. We assume a d-dimensional instance space X^d. Without loss of generality, we assume $X = [0, 1]$. In the context of recommendation, each dimension corresponds to one of d products, and the value on the dimension is a numeric rating. The nearest neighbor approach requires a function $\mathsf{dist}(\cdot,\cdot)$ defined over $X^d \times X^d$, but our analysis does not depend on any particular distance metric. Finally, let H be a hypothesis, C be a concept, and D be a probability distribution over X^d. The error rate of H with respect to C and D is defined as $\mathcal{E}(H, C, D) = \Pr_{x \in D}(H(x) \neq C(x))$.

3 Absolute Accuracy

The first model extends Albert and Aha's noise-free PAC results for k-NN [1] to handle biased class noise. We first review these noise-free results, and then state our model as Theorem 1.

The key idea behind Albert and Aha's (hereafter: AA) analysis is that of a "sufficiently dense" sample from the instance space. Informally, a subset $S \subset X^d$ is dense if most of the points in the entire space X^d are near many points in the sample S. The terms "most", "near" and "many" are formalized as follows: Let D be a distribution over X^d. A subset $S \subseteq X^d$ is (k, α, γ)-*dense* if, except for a subset with probability less than γ under D, for every $x \in X^d$, there exists at least k distinct points $x_1, \ldots, x_k \in S$ such that $\mathsf{dist}(x, x_i) \leq \alpha$ for each i.

Given this definition, AA derive [1, Lemma 2.2] a lower bound $\Upsilon_{\mathrm{d}}(k, \alpha, \gamma, |S|)$ on the probability that a sample S of X^d is (k, α, γ)-dense: $\Upsilon_{\mathrm{d}}(k, \alpha, \gamma, |S|) = 1 - m^d \sum_{0 \leq k' < k} \Phi_2(\rho, k', |S|)$, where $\Phi_2(\rho, t, s) = \binom{t}{s} B\left(\max\left\{s/t, \rho\right\}, s, t\right)$, $B(p, s, t) = p^s(1-p)^{t-s}$, $m = \lceil \sqrt{d}/\alpha \rceil$, and $\rho = \gamma/m^d$. (See Appendix A for a proof.) In the spirit of the PAC model, $\Upsilon_{\mathrm{d}}(k, \alpha, \gamma, |S|)$ is a worst-case lower bound that does not depend on the distribution D over X^d.

The final step is AA's analysis is to formalize the intuition that k-NN is accurate to the extent that its training sample is sufficiently dense. To describe their result, we first must constrain both the complexity of the concept being learned and the distribution over the instance space. For any constant B, let $\mathcal{D}_B$ be the class of distributions over X^d such that no point has probability exceeding B. For any constant L, let $\mathcal{C}_L$ is the set of concepts C over X^d such that C is bounded by closed hyper-curves of total length less L.

These constraints mean that AA's results hold only for a restricted class of learning tasks (due to the parameter L), and are not truly distribution-free (due to B). However, we will see below that in fact the model's predictions for real-world tasks are not very sensitive to the values of L and B.

Let $C \in \mathcal{C}_L$ be a concept to be learned, and $D \in \mathcal{D}_B$ be a distribution over X^d. AA's central result [1, Theorem 3.2] is that the probability that the error of k-NN exceeds ϵ when trained on a sample S is at most $\Upsilon_{\rm d}\,(k, \epsilon/4LB, \epsilon/2, |S|)$.[1]

The intuition behind AA's results is that k-NN is accurate when training on a sufficient number of instances. We can extend this intuition to handle biased class noise by requiring that, in addition to the sample containing enough "good" (noise-free) instances, it must also not contain too many "bad" (noisy) instances.

We begin by defining sparse subsets analogously to the definition of dense subsets. Informally, a subset $S \subset X^d$ is sparse if most of the points in the entire space X^d are near few points in the sample S. More precisely, a subset $S \subseteq X^d$ is (k, α, γ)-*sparse* if, except for a subset with probability less than γ, for every x there exists at most k points x_i such that $\mathsf{dist}(x, x_i) \leq \alpha$.

Appendix A proves the following lower bound $\Upsilon_{\rm s}(k, \alpha, \gamma, |S|)$ on the probability that a sample S of X^d is (k, α, γ)-sparse:

$$\Upsilon_{\rm s}(k, \alpha, \gamma, |S|) \;=\; 1 - m^d \sum_{\langle k', k'' \rangle \in \mathcal{K}} \Phi_3(\rho, k', k'', |S|),$$

where m and ρ are as defined above, $\Phi_3(\rho, r, s, t) = \binom{t}{r}\binom{t-r}{s} T\,(\max\{r/t, \rho\}, r, s/t, s, t)$, $T(p, r, q, s, t) = p^r q^s (1-p-q)^{t-r-s}$, and $\mathcal{K} = \{\langle k', k'' \rangle | 0 \leq k', k'' \leq |S| \wedge\ k < k' + k'' \leq |S|\}$.

To complete our analysis, we observe that the accuracy of k-NN with training data S is equal to the probability that S contains enough "good" instances and not too many "bad" instances. For k-NN, "enough" and "not too many" mean that most of the instances should have at least $\lceil k/2 \rceil$ good neighbors and at most $\lfloor k/2 \rfloor$ bad neighbors.

Let $S_{\rm real} \subset S$ be the examples corresponding to genuine users, and $S_{\rm attack} = S \setminus S_{\rm real}$ be the examples comprising the attack. Of course, we can not know $S_{\rm real}$ and $S_{\rm attack}$ exactly, but we do know that $|S_{\rm real}| = (1-\beta)|S|$ and $|S_{\rm attack}| = \beta|S|$ (where β is the size of the attack), which is sufficient for our analysis.

Furthermore, some of the noisy instances may in fact be correctly labelled. Let f be the fraction of the instance space labelled 1, and let μ be the class noise bias. Then a fraction $f\mu + (1-f)(1-\mu)$ of the noisy instances are in fact correctly labelled. Let $S_{\rm good} \supseteq S_{\rm real}$ be the instances that are actually labelled

[1] We have departed from AA in several ways. First, AA use the notation k-$\langle \alpha, \gamma \rangle$-*net*; we refer to "denseness" because our biased class noise analysis involves an analogous notion of sparseness. Second, our proof is somewhat different and therefore our bound on the denseness probability differs slightly from AA's. Most importantly, as is standard in PAC analysis, AA introduce an additional confidence parameter δ and solve $\Upsilon_{\rm d}(k, \epsilon/4LB, \epsilon/2, |S|) > 1 - \delta$ for $|S|$, in order to show that k-NN can PAC-learn efficiently. Since robustness is orthogonal to efficiency, we ignore this part of their analysis.

correctly, and $S_{\text{bad}} = S \setminus S_{\text{good}} \subseteq S_{\text{attack}}$ be the instances that are actually labelled incorrectly. Again, we can not know S_{good} or S_{bad} but we do know that the number of noise-free instances is $|S_{\text{good}}| = (1-\beta)|S| + \beta|S|(f\mu + (1-f)(1-\mu)) \geq |S_{\text{real}}|$. and the number of noisy instances is $|S_{\text{bad}}| = \beta|S|(1 - f\mu - (1-f)(1-\mu)) \leq |S_{\text{attack}}|$. If $\lambda = \beta(\mu + f - 2\mu f)$ is the effective attack size, then $|S_{\text{good}}| = (1-\lambda)|S|$ and $|S_{\text{bad}}| = \lambda|S|$.

We require that both S_{good} be $(\lceil k/2 \rceil, \alpha, \gamma)$-dense and S_{bad} be $(\lfloor k/2 \rfloor, \alpha, \gamma)$-sparse. Since these events are independent, the probability of their conjunction is the product of their probabilities. Therefore, if we can determine appropriate values for the distance thresholds α_1 and α_2 and probability thresholds γ_1 and γ_2, then we have that the accuracy of k-NN when training on S with biased class noise is at least $\Upsilon_{\text{d}}(\lceil k/2 \rceil, \alpha_1, \gamma_1, |S_{\text{good}}|) \cdot \Upsilon_{\text{s}}(\lfloor k/2 \rfloor, \alpha_2, \gamma_2, |S_{\text{bad}}|)$. In Appendix A we prove the following theorem.

Theorem 1 (Absolute accuracy). *The following holds for any ϵ, β, μ, d, k, L, B, $C \in \mathcal{C}_L$ and $D \in \mathcal{D}_B$. Let S be a sample of X^d according to D with biased class noise rate β, and let $H = k$-NN(S). Then we have that*

$$\begin{aligned}\Pr\left[\mathcal{E}(H,C,D) < \epsilon\right] \;\geq\; & \Upsilon_{\text{d}}\left(\lceil k/2 \rceil, \epsilon/4LB, \epsilon/4, (1-\lambda)|S|\right) \cdot \\ & \Upsilon_{\text{s}}\left(\lfloor k/2 \rfloor, \epsilon/4LB, \epsilon/4, \lambda|S|\right),\end{aligned}$$

where $\lambda = \beta(\mu + f - 2\mu f)$, and f is the fraction of X^d labeled 1 by C.

To summarize, Theorem 1 yields a worst-case lower bound on the accuracy of k-NN under biased class noise. On the positive side, this bound is "absolute" in the sense that it predicts (a probabilistic bound on) the actual error $\mathcal{E}(k\text{-NN}(S), C, D)$ as a function of the sample size $|S|$, noise rate β, and other parameters. In other words, the term "absolute" draws attention to the fact that this model takes account of the actual position along the learning curve. Unfortunately, like most PAC analyses, its bound is very weak (though still useful in practice; see Sec. 5).

4 Approximate Relative Accuracy

In contrast, the second model does not rely on a worst-case analysis and so makes tighter predictions than the first model. On the other hand, the model is only "approximate" because it makes two assumptions. First, it assumes that the training sample is large enough that the learning curve has "flattened out". Second, it assumes that, at this flat part of the learning curve, k-NN achieves perfect accuracy except possibly on the boundary of the target concept. We call this second model "approximate" to draw attention to these assumptions, and "relative" to note specifically that it does not predict error on an absolute scale.

To formalize these assumptions, let S be a training sample drawn from the distribution D over X^d, and let C be the target concept. Let S' be the fraction $1-\beta$ of the instances in S that were (correctly) labeled by C during the biased class noise process. Let D' be the distribution that is proportional to D except that $D'[x] = 0$ for all points x on the boundary between C and $X^d \setminus C$. The assumptions of the second model can be expressed as:

$$\mathcal{E}(k\text{-NN}(S'), C, D') = 0 \tag{1}$$

Given this assumption, we can predict the error of k-NN as follows. To classify an instance x using a training set S, k-NN predicts the majority class of the k instances $x_1, \ldots, x_k \in S$ that are closest to x. To classify x correctly, at least $\lceil k/2 \rceil$ of these k instances must have the correct class.

If we randomly draw from D a point $x \in X^d$, there are two cases: either $C(x) = 1$ (which happens with probability f), or $C(x) = 0$ (which happens with probability $1 - f$), where as above f is the probability under D that $C(x) = 1$.

In the first case, we need to have at least $\lceil k/2 \rceil$ successes out of k trials in a Bernoulli process where the probability of success is equal to the probability that a neighbor x_i of x will be labeled 1. We can calculate this probability as $(1-\beta)+\beta\mu$. The first term is the probability that x_i is labeled 1 and $x_i \in S'$; by (1), we know that this probability is $1 - \beta$. The second term is the probability that x_i is labeled 1 and $x_i \notin S'$, by definition of the biased class noise process labels we know that this probability is $\beta\mu$.

In the second case, again we need at least $\lceil k/2 \rceil$ successes, but with success probability $(1 - \beta) + \beta(1 - \mu)$, the probability that a neighbor x_i of x will be labeled 0. The first term is the probability that x_i is labeled 0 and $x_i \in S'$, and by (1) this happens with probability $1 - \beta$. The second term is the probability that x_i is labeled 0 and $x_i \notin S'$, which occurs with probability $\beta(1 - \mu)$.

The following theorem follows from this discussion.

Theorem 2 (Approximate relative accuracy). *The following holds for any β, μ, d, k, C and D. Let S be a sample of X^d according to D with biased class noise rate β. Let S' and D' be as defined above. If assumption (1) holds, then*

$$\mathcal{E}(k\text{-NN}(S), C, D') \ = \ 1 \ - \ f \cdot \sum_{k'=\lceil \frac{k}{2} \rceil}^{k} \binom{k}{k'} B(1-\beta(1-\mu), k', k) \ - \ (1-f) \cdot \sum_{k'=\lceil \frac{k}{2} \rceil}^{k} \binom{k}{k'} B(1-\beta\mu, k', k),$$

where f is the fraction of X^d labeled 1 by C.

Without more information, we can not conclude anything about $\mathcal{E}(k\text{-NN}(S), C, D)$ (which is what one can measure empirically) from $\mathcal{E}(k\text{-NN}(S), C, D')$ (the model's prediction) or from $\mathcal{E}(k\text{-NN}(S'), C, D') = 0$ (the assumption underlying the model). For example, if D just so happens to assign zero probability to points on C's boundary, then $\mathcal{E}(k\text{-NN}(S'), C, D) = \mathcal{E}(k\text{-NN}(S'), C, D')$ and so in the best case $\mathcal{E}(k\text{-NN}(S), C, D) = 0$. On the other hand, if all of D's mass is on C's boundary then in the worst case $\mathcal{E}(k\text{-NN}(S), C, D) = 1$. Furthermore, it is generally impossible to know whether $\mathcal{E}(k\text{-NN}(S'), C, D') = 0$.

Despite these difficulties, we will evaluate the model on real-world data by simply assuming $\mathcal{E}(k\text{-NN}(S'), C, D') = 0$ and $D' = D$, and comparing the predicted and observed error.

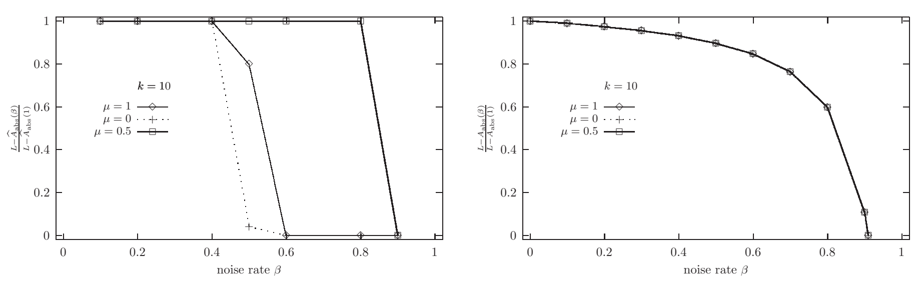

Fig. 1. MUSHROOM: Empirical (left) and predicted (right) absolute accuracy

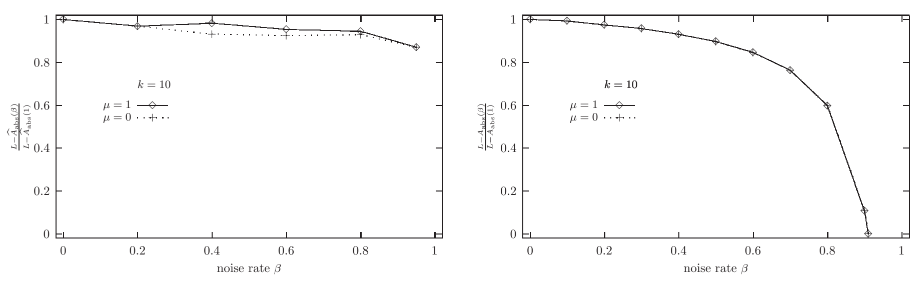

Fig. 2. PTV: Empirical (left) and predicted (right) absolute accuracy

5 Evaluation

We evaluated the two models against two real-world learning tasks:

- The MUSHROOM data-set from the UCI repository contains 8124 instances with 23 attributes, with no missing values.
- The PTV collaborative recommendation data for television listing [www.ptv.com] contains 2344 instances (people) and 8199 attributes (television programs), and only 0.3% of the matrix entries are non-null. We discarded people who rated fewer than 0.05% of the programs, and programs rated by fewer than 0.05% of the people. The resulting 241 people and 570 programs had a sparseness of 15.5%. The original ratings (values from 1–4) were converted into binary attributes ('like'/'dislike').

We used the standard k-NN algorithm with no attribute or vote weighting. Distance was measured using the Euclidean metric (ignoring non-null attributes). All experiments use $k = 10$ neighbors.

Our experiments use a variation on the standard cross validation approach. We repeat the following process many times. First, we randomly partition the entire set of instances into a *real* set R, a *fake* set F, and a testing set T. To implement the biased class noise model, a *noisy* set N containing $\beta|R|/(1-\beta)$

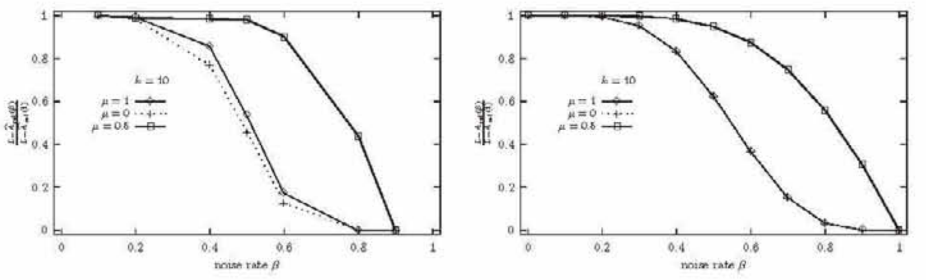

Fig. 3. MUSHROOM: Empirical (left) and predicted (right) relative accuracy

instances is then randomly drawn from F. The class attributes of the instances in N are then modified to 1 with probability μ and 0 with probability $1 - \mu$. The k-NN learning algorithm is then training on $R \cup N$ (so the noise rate is $|N|/|R \cup N| = \beta$). We measure accuracy as the fraction of correct predictions for the test instances in T. For MUSHROOM, noise is added only to the class attribute defined by the data-set's authors. For PTV the class attribute (i.e., program to attack) is selected randomly.

Absolute accuracy model. The absolute accuracy model predicts

$$\Pr[\mathcal{E}(k\text{-NN}(S_\beta), C, D) < \epsilon],$$

the probability that the accuracy exceeds $1-\epsilon$, where S_β is a sample with biased class noise rate β. Let the model's predicted absolute accuracy from Theorem 1 be

$$A_{\text{abs}}(\beta) = \Upsilon_{\text{den}}(\lceil k/2 \rceil, \epsilon/4LB, \epsilon/4, (1-\lambda)|S_\beta|) \cdot \Upsilon_{\text{spa}}(\lfloor k/2 \rfloor, \epsilon/4LB, \epsilon/4, \lambda|S_\beta|).$$

Our empirical estimate $\widehat{A}_{\text{abs}}(\beta)$ of this probability is simply the fraction of trials for which ϵ exceeds the error.

Recall that Theorem 1 requires a bound L on the perimeter of the target concept, and a bound B on the probability of any instance under the distribution D. Thus our model is not completely general, and furthermore it is difficult to estimate these parameters for a given learning task. However, it is easily shown that for small values of k, $A_{\text{abs}}(\beta)$ does not depend on L and B, and thus we do not need to tune these parameters of our model for each learning task.

Due to the worst-case analysis, typically $A_{\text{abs}}(\beta) \gg 1$ - clearly an absurd value. However, for the purposes of analysing robustness, such values are useful, because we are interested in the increase in error at noise rate β compared to $\beta = 0$. We therefore report results using the ratios $(L - A_{\text{abs}}(\beta))/(L - A_{\text{abs}}(0))$ and $(L - \widehat{A}_{\text{abs}}(\beta))/(L - \widehat{A}_{\text{abs}}(0))$, where $L = A_{\text{abs}}(1)$ is a constant chosen to scale the ratios to [0,1].

The results for MUSHROOM with $\epsilon = 0.25$ are shown in Fig. 1. The predicted and observed accuracies agree reasonably well, even accounting for the fact that

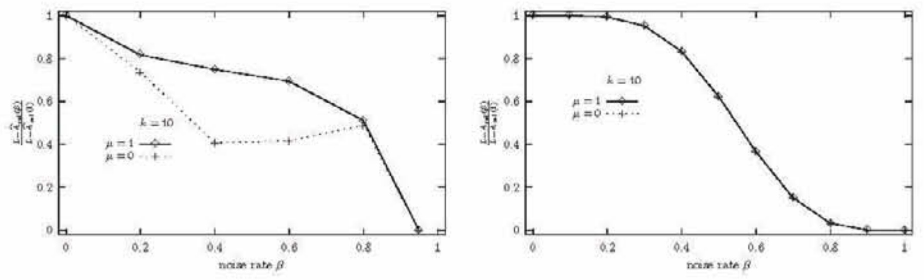

Fig. 4. PTV: Empirical (left) and predicted (right) absolute accuracy

the data have been scaled to [0,1]. The fit is by no means perfect, but we are satisfied with these results, since worst-case PAC-like analyses are usually so weak as to be incomparable to real data.

Fig. 2 shows the results for PTV with $\epsilon = 0.3$. Here the fit is worse: PTV appears to be much more robust in practise than predicted, particular as β increases. We conjecture that this is due to the fact that the PTV data is highly noisy, but further analysis is needed to explain these data.

Relative accuracy model. The relative accuracy model predicts $\mathcal{E}(k\text{-NN}(S_\beta), C, D)$, the error of k-NN when trained on a sample S_β with biased class noise rate β. Let $A_{\text{rel}}(\beta)$ be the model's prediction from Theorem 2. Our empirical estimate $\widehat{A}_{\text{rel}}(\beta)$ of this probability is simply the fraction of incorrectly classified test instances. As before, we scale all data to [0-1].

The results for MUSHROOM are shown in Fig. 3 and the PTV results are shown in Fig. 4. The model fits the observed data quite well in both domains, though as before PTV appears to be inherently noisier than MUSHROOM.

6 Related Work

Collaborative recommendation has been empirically validated in numerous standard "customer/product" scenarios [3]. However, there is relatively little theoretical understanding of the conditions under which the technique to be effective.

Our work is highly motivated by ongoing empirical investigations of the robustness of collaborative filtering [9,8]. The ideas underlying Theorem 1 borrow heavily from Albert et al's seminal PAC analysis of noise-free k-NN [1].

There has been substantial theoretical algorithmic work on collaborative filtering [7,2,4]. For example, Azar et al [2] provide a unified treatment of several information retrieval problems, including collaborative filtering, latent semantic analysis and link-based methods such as hubs/authorities. They cast these problems as matrix reconstruction: given a matrix of objects and their attributes (eg, for collaborative filtering, the objects are products, the attributes are customers, and matrix entries store customers' ratings) from which some entries have been deleted, the task is to reconstruct the missing entries (eg, predict whether a particular customer will like a specific product). Azar et al prove that the matrix entries can be efficiently recovered as long as the original data has a good low-rank approximation.

The fundamental difference between all of these results and ours is that our biased class noise model is more malicious than simple random deletion of the matrix entries. It remains an open question whether these results can be extended to accommodate this model

7 Discussion

Collaborative recommendation has been demonstrated empirically, and widely adopted commercially. Unfortunately, we do not yet have a general predictive theory for when and why collaborative filtering is effective. We have investigated one particular facet of such a theory: an analysis of robustness, a measure of a recommender system's resilience to potentially malicious perturbations in the customer/product rating matrix.

This investigation is both practically relevant for enterprises wondering whether collaborative filtering leaves their marketing operations open to attack, and theoretically interesting for the light it sheds on a comprehensive theory of collaborative filtering.

We developed and evaluated two models for predicting the degradation in predictive accuracy as a function of the size of the attack and other parameters. The first model uses PAC-theoretic techniques to predict a bound on accuracy. This model is "absolute" in that it takes account of the exact position of the system along the learning curve, but as a worst-case model it is problematic to evaluate its predictions . In contrast, the second model makes tighter predictions, but is "relative" in the sense that it assumes perfect prediction in the absence of the malicious attack. Our preliminary evaluation of the model against two real-world data-sets demonstrates that our model fits the observed data reasonably well.

Acknowledgments

I thank M. O'Mahony, N. Hurley, G. Silvestre and M. Keane for helpful discussion, B. Smyth for the PTV data, and the Weka developers. This research was funded by grant N00014-00-1-0021 from the US Office of Naval Research, and grant SFI/01/F.1/C015 from Science Foundation Ireland.

References

1. M. Albert and D. Aha. Analyses of instance-based learning algorithms. In *Proc. 9th Nat. Conf. Artificial Intelligence*, 1991. 234, 235, 240, 243
2. Y. Azar, A. Fiat, A. Karlin, F. McSherry, and J. Saia. Spectral analysis of data. In *Proc. 32nd ACM Symp. Theory of Computing*, 2001. 232, 240

3. J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Conf. Uncertainty in Artificial Intelligence*, 1998. 232, 240
4. P. Drineas, I. Kerenidis, and P. Raghavan. Competetive recommender systems. In *Proc. 32nd ACM Symp. Theory of Computing*, 2002. 232, 240
5. D. Goldberg, D. Nichols, B. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *C. ACM*, 35(12):61–70, 1992. 232
6. M. Kearns and M. Li. Learning in the presence of malicious errors. In *Proc. ACM Symp. Theory of Computing*, 1988. 233
7. R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tompkins. Recommender systems: A probabilistic analysis. In *Proc. 39th IEEE Symp. Foundations of Computer Science*, 1998. 232, 240
8. M. O'Mahony, N. Hurley, N. Kushmerick, and G. Silvestre. Collaborative recommendation: A robustness analysis. Submitted for publication, 2002. 232, 240
9. M. O'Mahony, N. Hurley, and G. Silvestre. Promoting recommendations: An attack on collaborative filtering. In *Proc. Int. Conf. on Database and Expert System Applications*, 2002. 232, 240
10. U. Shardanand and P. Maes. Social information filtering: Algorithms for automating "word of mouth". In *Proc. Conf. Human Factors in Computing Systems*, 1994. 232

A Proof of Theorem 1

The following two lemmas are easily proven.

Lemma 1. *Consider a binomial process where outcome o_1 has probability at least ρ and outcome o_2 consumes the remaining probability mass. The probability of exactly s o_1-outcomes in t trials is at most $\Phi_2(\rho, s, t) = \binom{t}{s} B\left(\max\left\{\frac{s}{t}, \rho\right\}, s, t\right)$, where $B(p, s, t) = p^s(1-p)^{t-s}$.*

Lemma 2. *Consider a trinomial process where outcome o_1 has probability at least ρ and outcomes o_2 and o_3 consume the remaining probability mass. The probability of exactly r o_1-outcomes and s o_2-outcomes in t trials is at most $\Phi_3(\rho, r, s, t) = \binom{t}{r}\binom{t-r}{s} T\left(\max\left\{\frac{r}{t}, \rho\right\}, r, \frac{s}{t}, s, t\right)$, where $T(p, r, q, s, t) = p^r q^s (1 - p - q)^{t-r-s}$.*

The following lemma bounds the probability that a subset is sparse or dense.

Lemma 3. *The following holds for any d, α, γ, k, and distribution D. Let $m = \lceil \sqrt{d}/\alpha \rceil$ and $\rho = \frac{\gamma}{m^d}$. The probability that a sample S of X^d drawn according to D is (k, α, γ)-dense is at least*

$$\Upsilon_{\mathrm{d}}(k, \alpha, \gamma, |S|) \;=\; 1 - m^d \sum_{0 \le k' < k} \Phi_2(\rho, k', |S|), \tag{2}$$

and the probability that S is (k, α, γ)-sparse is at least

$$\Upsilon_{\mathrm{s}}(k, \alpha, \gamma, |S|) \;=\; 1 - m^d \sum_{\langle k', k'' \rangle \in \mathcal{K}} \Phi_3(\rho, k', k'', |S|), \tag{3}$$

where $\mathcal{K} = \{\langle k', k'' \rangle \mid 0 \le k', k'' \le |S| \wedge k < k' + k'' \le |S|\}$.

Proof. First consider (2). Partition X^d into m^d squares, where m is chosen large enough so that any two points in one square are at most distance α apart. By the Pythagorean Theorem, we require that $m \geq \sqrt{d}/\alpha$, so choose $m = \lceil \sqrt{d}/\alpha \rceil$. Let F be the set of *frequent* squares: those with probability at least $\rho = \frac{\gamma}{m^d}$. Since there are at most m^d squares not in F, the total probability of the non-frequent squares is at most $m^d \rho = m^d \cdot \frac{\gamma}{m^d} = \gamma$. If at least k sample points lie in each of the frequent squares, then the sample will be sufficiently dense for the points in the frequent squares. The probability of selecting a point in some particular heavy square is at least ρ, and the probability of not selecting a point in this square is at most $1-\rho$. By Lemma 1, the probability of selecting exactly k' out of $|S|$ points in some particular heavy square is at most $\Phi_2(\rho, k', |S|)$. Therefore the probability of selecting fewer than k points in some particular heavy square is at most $\sum_{0 \leq k' < k} \Phi_2(\rho, k', |S|)$. This expression is an upper bound on the probability that the sample is insufficiently dense for some particular frequent square. Since there are at most m^d frequent squares, the probability that the sample is insufficiently dense for one or more frequent squares is at most $m^d \sum_{0 \leq k' < k} \Phi_2(\rho, k', |S|)$. (2) follows by noting that the probability that the sample is sufficiently dense for every frequent square is therefore at least $1 - m^d \sum_{0 \leq k' < k} \Phi_2(\rho, k', |S|)$.

We now prove (3). As before, partition X^d into m^d squares. The *border* of a square is the set of points outside but within α of the square. If at most k sample points lie in each of the frequent squares and their borders, then the sample will be sufficiently sparse for the points in the frequent squares. Consider some frequent square in the set F of squares with probability at least $\rho = \frac{\gamma}{m^d}$. By Lemma 2, we have that $\Phi_3(\rho, k', k'', |S|)$ is an upper bound on the probability that k' sample points fall in some specific heavy square and k'' points fall in its border. Therefore the probability of selecting more than k points in some particular frequent square or its border is at most $\sum_{k', k''} \Phi_3(\rho, k', k'', |S|)$, where the sum is over the combinations of k' and k'' satisfying $0 \leq k', k'' \leq |S|$ and $k < k' + k'' \leq |S|$. Since there are at most m^d frequent points, (3) follows by noting that the probability that the sample is sufficiently dense for every frequent point is at least one minus m^d times this expression. □

Finally, our proof of Theorem 1 derives from [1]; here we sketch the main ideas.

Let $C \in \mathcal{C}_L$ be the target concept. Let $C_\alpha^\subset \subset C$ be the *core* of the concept: the points at least a distance α away from some point not in C. Let $C_\alpha^\supset \supset C$ be the concept's *neighbourhood*: the points within a distance α of some point in C. As introduced in Section 3, let S_{good} be the noise-free instances in a sample S and $S_{\text{bad}} = S \setminus S_{\text{good}}$ be the noisy instances. Suppose that S_{good} is $\langle \lceil k/2 \rceil, \alpha, \gamma \rangle$-dense and S_{bad} is $\langle \lfloor k/2 \rfloor, \alpha, \gamma \rangle$-sparse. Let k-NN(S) be the set of points labelled by k-NN given training data S. Then, except for a set U of instances with probability at most 2γ, k-NN(S) is bounded within a distance α of the target concept: $(C_\alpha^\subset \setminus U) \subseteq (k\text{-NN}(S) \setminus U) \subseteq (C_\alpha^\supset \setminus U)$.

We can therefore bound the error of k-NN(S) by bounding the probability of $C_\alpha^\supset \setminus C_\alpha^\subset$. The perimeter of C is at most L, and therefore the volume of $C_\alpha^\supset \setminus C_\alpha^\subset$ is at most $2\alpha L$. No point in X^d has probability greater than B, and therefore the probability of $C_\alpha^\supset \setminus C_\alpha^\subset$ is at most $2\alpha L B$.

Therefore, assuming that S_{good} is $\langle\lceil k/2\rceil, \alpha, \gamma\rangle$-dense and S_{bad} is $\langle\lfloor k/2\rfloor, \alpha, \gamma\rangle$-sparse, we have that $\mathcal{E}(k\text{-NN}(S), C, D) < 2\gamma + 2\alpha LB$. The first term counts instances whose noisy neighbours outvoted noise-free neighbours, and the second term counts mistakes that might occur on the boundary of C.

To complete the proof, we must ensure that $2\gamma + 2\alpha LB < \epsilon$. Since we seek a lower bound on the probability that $\mathcal{E}(k\text{-NN}(S), C, D) < \epsilon$, we can simply split the total permissible error equally between the two causes: $2\gamma = \epsilon/2$ and $2\alpha LB = \epsilon/2$, or $\gamma = \epsilon/4$ and $\alpha = \epsilon/4LB$.