MINIMUM EXPECTED RISK PROBABILITY ESTIMATES FOR NONPARAMETRIC NEIGHBORHOOD CLASSIFIERS

Maya Gupta, Luca Cazzanti, and Santosh Srivastava

University of Washington
Dept. of Electrical Engineering
Seattle, WA 98195

ABSTRACT

We consider the problem of estimating class probabilities for a given feature vector using nonparametric neighborhood methods, such as k-nearest neighbors (k-NN). This paper's contribution is the application of minimum expected risk estimates for neighborhood learning methods, an analytic formula for the minimum expected risk estimate for weighted k-NN classifiers, and examples showing that the difference can be significant.

keywords: Laplace correction, k-nearest neighbors, nonparametric classification, minimum expected risk, LIME

1. INTRODUCTION

We consider the problem of estimating the probability distribution over class labels using nonparametric neighborhood methods, such as k-nearest neighbors (k-NN). We apply Laplace correction to k-NN estimates, and give a more general analytic formula for the minimum expected risk estimate for weighted k-NN classifiers. Simulations show that the difference between such minimum expected risk estimates and maximum likelihood estimation can be significant.

Supervised statistical learning is based on a set of given training pairs $\mathcal{T} = \{X_i, Y_i\}$, where $X_i \in \mathcal{R}^d$ and $Y_i \in \mathcal{G}$, where $\mathcal{G}$ is a finite set of class labels. The classification problem is to estimate the probability of each class for a test feature vector, $P(Y = g|X = x)$ for $g \in \mathcal{G}$, based on the given training set. There are many approaches to supervised statistical learning, we focus here on nonparametric neighborhood methods. Such methods are known to perform well in practice, are intuitive, and can achieve optimal error rates asymptotically [1]. Nonparametric neighborhood methods weight the training samples in a neighborhood around the test point x. It is not important for this paper how one defines a neighborhood; common definitions are to use the k-nearest neighbors, or all neighbors within a defined radius.

This work was supported in part by the National Science Foundation Grant SBE-0123552 and by an AFCEA Fellowship.

As is most common, we define the k-nearest neighbors in terms of Euclidean distance. We assume that the sample pairs are re-indexed by their distance, so that X_j is the jth nearest neighbor to x. Given a neighborhood, a weighted k-NN classifier assigns a weight w_j to each neighbor, usually by evaluating a kernel which assigns weight based on the distance from x to X_j [1]. The k-NN classifier assigns equal weights to every neighbor. Our formulation will hold for any weighted k-NN classifier where the weights satisfy $\sum_{j=1}^{k} w_j = 1$ and $w_j \geq 0$. We will present results for the symmetric tricube kernel, and a recent method which uses linear interpolation and maximum entropy (LIME) [2],[3] to assign weights asymmetrically to neighbors to satisfy the linear interpolation equations.

From the weights and the neighborhood sample pairs, it is standard to form a maximum likelihood (ML) estimate of the probability of each class,

$$\hat{P}(Y = g|x) = \sum_{j=1}^{k} w_j I_{(Y_j=g)} \tag{1}$$

for $g \in \mathcal{G}$, and $I_{(\cdot)}$ is an indicator function that is one when its argument is true. This creates an estimate $\hat{P}_{Y|x}$ that maximizes the "weighted likelihood" of the data samples $Y_1, Y_2, \ldots, Y_k$ seen in the neighborhood of the test feature vector x under the k-NN assumption that these local samples can be effectively utilized to estimate the class probabilities given $X = x$. However, maximum likelihood estimates can be quite unrepresentative of the underlying likelihood distribution, particularly when small sample sizes are used. Near-neighbor algorithms are often run with small neighborhood sizes which yields small sample sizes for the estimation done in (1), and thus we hypothesize that a different estimation principle could make a difference in practice. See [4, pgs. 300-310] for a further discussion on problems with maximum likelihood estimation.

Let us consider applying a different principle of estimation to the near-neighbor context. Given any estimated pmf q, if the truth is in fact some other pmf p, then the error (or risk or cost) is some function $R(p, q)$. Minimizing the

expectation of a relevant risk R where the expectation is taken over all possible pmf's p is a more robust approach to estimation. In this paper, we apply a Bayesian minimum expected risk principle (MER) [5, ch. 4] to estimate the class probabilities in weighted k-NN statistical learning. Then, the MER estimate of the class conditional probability $\hat{q}$ solves

$$\underset{q}{\operatorname{argmin}} E_{P_{Y|x}}\left[R(P_{Y|x}, q)\right] \tag{2}$$

where R is some function (such as mean-squared error, or relative entropy) that measures the risk of estimating the conditional class probability distribution to be q if the truth is $P_{Y|x}$. Note that the expectation is over $P_{Y|x}$, this conditional probability mass function is treated as a random variable to form the estimate.

We focus on the two-class case, $Y \in \{0, 1\}$. Let the estimated probability of a test point x being class 0 be $\hat{\theta}$. Denote the probability of a test point being class 0 as the random variable Θ. A particular realization of the random variable Θ will be denoted with the lower-case notation θ. Then (2) can be re-written as,

$$\underset{\hat{\theta}}{\operatorname{argmin}} \int R(\theta, \hat{\theta}) f(\theta) d\theta, \tag{3}$$

where $f(\theta)$ is the probability of θ. When R is chosen to be the mean-squared error, then this is equivalent to Bayesian minimum mean-squared error (BMMSE) parameter estimation [6], which minimizes the expected posterior risk if $f(\theta)$ is defined to be the posterior distribution of Θ given the k neighborhood samples.

For the k-NN classifier, we propose defining $f(\theta)$ to be the likelihood of independently and identically drawing the data seen in the neighborhood of feature vector x so that

$$f(\theta) = a \frac{k!}{m!(k-m!)} \theta^m (1-\theta)^{(k-m)} \tag{4}$$

where $m = \sum_{j=1}^{k} I_{(Y_j=0)}$ and a is a normalization constant. This is equivalent to defining $f(\theta)$ to be the posterior with a uniform prior on Θ. Then, the class probability distribution that solves (3) is, for either mean-squared R or relative entropy R,

$$\hat{P}_{Y|x} = \{\frac{m+1}{k+2}, \frac{k-m+1}{k+2}\}. \tag{5}$$

This estimation formula is equivalent to *Laplace correction* [7, pg. 272], also called *Laplace smoothing*, which has been used to estimate class probabilities in decision trees [8] and in speech recognition [9].

Equation (5) does not take into account the weights w_j that weighted neighborhood classifiers place on each training sample. For weighted k-NN classifiers, we propose that $f(\theta)$ be the likelihood of drawing the neighborhood samples with each neighborhood sample weighted by w_j, so that

$$f(\theta) = \prod_{j=1}^{k} \theta^{w_j k I_{(Y_j=0)}} (1-\theta)^{w_j k I_{(Y_j=1)}} \tag{6}$$

where we have dropped the normalization constant on $f(\theta)$ since it will not affect the estimation's minimization problem (2). We can re-write $f(\theta)$ as

$$f(\theta) = \theta^{k \sum_{j=1}^{k} w_j I_{(Y_j=0)}} (1-\theta)^{k \sum_{j=1}^{k} w_j I_{(Y_j=1)}}. \tag{7}$$

Note that for k-NN the weights are uniform, $w_j = 1/k$ for all j, and then the formulas (7) and (4) are equivalent (up to a normalization constant). Solving (3) with the weighted-data likelihood $f(\theta)$ given in (7) yields,

$$\hat{P}_{Y|x} = \left\{ \frac{1 + k \sum_{j=1}^{k} w_j I_{(Y_j=0)}}{k+2}, \frac{1 + k \sum_{j=1}^{k} w_j I_{(Y_j=1)}}{k+2} \right\}, \tag{8}$$

for either mean-squared error R or relative entropy R. The result (8) is in fact equivalent to (5) for $w_j = 1/k$. Proofs of (8) are given in the appendix for mean-squared error R and relative entropy R.

2. IS THE DIFFERENCE SIGNIFICANT?

The MER estimate (8) explicitly takes into account how many data points k have been given, and the smaller k is, the larger the difference between the MER and ML estimates. In the limit of $k \to \infty$, the ML and MER class probability estimates for a test point x converge. However, k-NN algorithms are often run for small values of k, including $k = 1$.

As an example, if $k = 1$ and that closest near neighbor is from class one, then the ML estimate will be $\hat{P}(Y = 1|x) = 1$, while the MER estimate will be the more conservative $\hat{P}(Y = 1|x) = 2/3$. This is the same situation as flipping a coin once, seeing a head, and guessing that the coin will only flip heads (ML) versus guessing that the coin will flip heads $2/3$ of the time (MER). In general the MER estimation using the likelihood for $f(\theta)$ will yield an estimate that is the empirical distribution ($\hat{\theta} = m/k$) pushed towards the uniform distribution.

Nonparametric neighborhood methods threshold the class probability estimates to estimate a class label corresponding to a test point x. For symmetric misclassification costs, the threshold is set at .5. More generally, the cost is a $m \times m$ matrix for an m class problem, where the ith, jth element is $C(i, j)$, the cost of estimating class i given the truth is class j. For the two-class problem with class 0 and class 1, the classification threshold t is theoretically optimally set [1] to minimize expected cost at

$$t = \frac{C(0,1)}{C(0,1) + C(1,0)}. \tag{9}$$

In practical learning problems such as computer-aided diagnostics of medical problems, the costs can be extremely asymmetric.

For a threshold of $t = .5$, the classification decision will be unchanged given a MER or ML estimate of $P_{Y|X}$. The further the threshold is from $.5$ due to asymmetric misclassification costs, the larger the difference in the two estimations. In the next section, we investigate how large these differences can be in a simple classification problem by simulation.

3. SIMULATIONS

First, we evaluate the estimation error of the class probability estimates, then we look at the difference in classification error.

Consider a simple two-class classification simulation example where training samples $X_i \in \mathcal{R}^d$ and test points $X \in \mathcal{R}^d$ are drawn iid from class 0: $\mathcal{N}(0, \Sigma)$ or from class 1: $\mathcal{N}(0, 2\Sigma)$; where the covariance matrix Σ is the $d \times d$ identity matrix and the classes are equally likely a priori. The simulation was run with a number of training sample sizes and for a number of feature dimension sizes; the results were consistent.

We evaluated the estimation difference using three nonparametric neighborhood methods: k-NN, tricube weighting [1], and linear interpolation with maximum entropy weights (LIME) [2], [3]. The LIME classifier weights neighborhood points with weights that solve

$$\underset{w}{\operatorname{argmin}} \left(\| \sum_{j=1}^{k} w_j X_j - x \|_2^2 - \lambda H(w) \right) \qquad (10)$$

where $H(w)$ is the Shannon entropy, $H(w) = -\sum_j w_j \log w_j$, and $\| \cdot \|_2$ is the l_p norm with $p = 2$. The parameter λ can be trained or cross-validated, or as a default, λ is set to be very small.

The LIME objective (10) trades-off between two goals: satisfying the linear interpolation equations, and maximizing the entropy of the weights. If the only goal were to maximize the entropy of the weights, the weights would all be equal, and LIME reduces to k-NN. The linear interpolation equations require that the weights are chosen so that the weighted neighbors have the test point x as their center of mass, that is, so $\sum_j w_j X_j = x$. These linear interpolation equations are not always solvable, and thus the LIME objective is to minimize the squared l_2 error between $\sum_k w_j X_j$ and x. Jointly determining the weights in this way is helpful particularly when the distribution in feature space of the neighbors is asymmetric. If two neighbors are too close in feature space, they are each less informative, and they each get less weight from the linear interpolation equations. Near-neighbors in sparse regions receive relatively more weight. We conjecture that maximizing the entropy of the weights helps keep the estimation variance down, while solving the linear interpolation equations helps reduce the estimation bias [3].

3.1. Probability estimation simulation

In this section we present results for the simulation described above with a training set of 100 samples in three-dimensions ($d = 3$). A test set of 100 samples was used to train the number of neighbors k to use with each method, where k was chosen to minimize the mean-squared error of the estimates. Fig. 1 shows the performance of each estimation method on the 100 sample test set as k grows. Once k was trained on this small test set, a validation run of 10,000 samples was run for each method with the method's trained k. The entire simulation was run five times and the averaged results are given in Table 1. For k-NN, the MER estimate had 10% lower mean-squared error than ML. For the tricube weighting, the MER improved performance by 34%. With the LIME weighting, the improvement was roughly 4%.

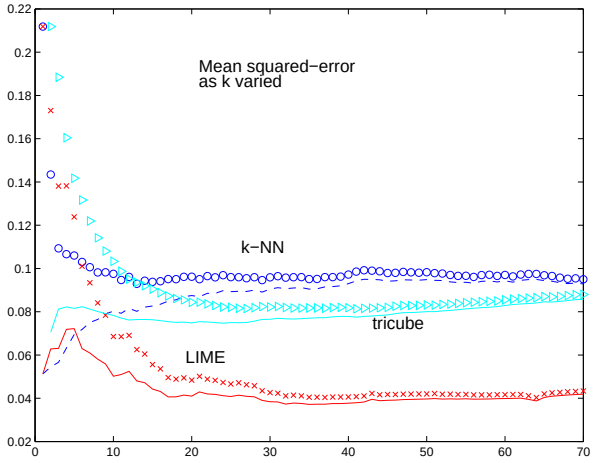

Fig. 1. For each classifier, the lines show MER performance and the shapes (circle, x, or triangle) show the ML performance in estimating the probabilities as the neighborhood size k is increased. The performance is the average mean-squared error of the probability estimate on 100 test samples, based on 100 training samples.

The formulas given in (5) and (8) are also valid for relative entropy. However, in evaluating the relative entropy performance, there is a problem that a single estimate of $\hat{\theta} = 1$ or $\hat{\theta} = 0$ yields an infinite relative entropy which

Method	Average error of probability estimates
ML k-NN	.1066
MER k-NN	.0959
ML tricube	.12220
MER tricube	.0803
ML LIME	.0403
MER LIME	.0388

Table 1. Comparison of average mean-squared error of the class probability estimates for test points using the ML and MER principles with k-NN, weighted k-NN with tricube weights, and weighted k-NN with LIME weights.

throws any "average relative entropy" off. The ML estimates are often extreme, and thus for all the test sets simulated the ML estimates had ill-behaved average entropy. The MER estimation does not result in extreme estimates, and so the relative entropy of the MER estimates is always well-behaved.

3.2. Classification simulation

In this section, the above Gaussian simulation is used to present example classification results for a four-dimensional ($d = 4$) case. For each classifier, the number of neighbors k was trained on a set of 100 test samples and 100 training samples, where the trained k corresponded to the fewest classification errors at a threshold of .5.

For k-NN, the trained number of neighbors was $k = 1$; for LIME, $k = 9$. The LIME parameter λ was set to the default value $\lambda = 0.001$. Then, 50,000 validation samples were classified based on the trained k.

We vary the classification threshold from .01 to .99, and plot the ratio of the empirical cost of each classifier using the ML principle to the empirical cost of each classifier using the MER principle. As per (9), a threshold t implies costs of $C(0,1) = (1-t)$ and $C(1,0) = t$.

Plotted in Fig. 2 is the ratio of the empirical costs; it is seen that particularly for highly asymmetric costs the difference in MER and ML estimation can be quite large. It is hard to generalize statistical learning performance, so these simulation results stand as a proof-of-concept that the difference between the estimation methods can be significant.

4. DISCUSSION

In this paper we have suggested using a minimum expected risk principle for weighted k-NN probability estimates and classification. For uniformly weighted k-NN, this is equivalent to applying Laplace smoothing. For classifiers that weight neighborhood samples with nonuniform weights, we provide a weighted likelihood function and derive an analytic formula for the MER estimate that is consistent with

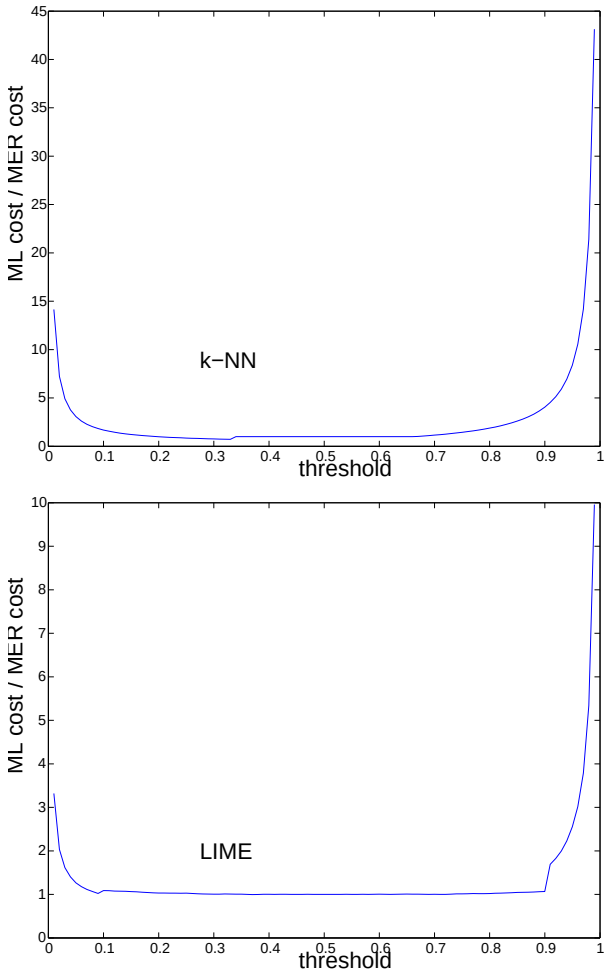

Fig. 2. *Top:* Ratio of ML and MER empirical misclassification costs for uniformly weighted k-NN as the cost threshold varies. *Bottom:* Ratio of ML and MER empirical misclassification costs for LIME as the cost threshold varies.

the uniformly weighted case. It is seen that the probability estimation can be more accurate with the new method.

For simplicity we focused on the two-class case. Preliminary investigations show that the multi-class case will be conceptually and mathematically very similar.

As analyzed in Friedman's work [10], classification is robust to large errors in probability estimation if the errors are in the "right" direction, since the probability estimates are thresholded. However, for asymmetric costs this paper shows that the classification cost can be significantly reduced using the MER estimation by forming an estimate that minimizes either the expected mean-squared error or relative entropy. In many practical applications, the costs are asymmetric, and the presented results may be useful in practice.

Appendix

Proof of (8) *for mean-squared* R:
The objective of (3) with $R = (\theta - \hat{\theta})^2$ is convex in $\hat{\theta}$, and thus it suffices to compute its first derivative with respect to $\hat{\theta}$ and set to zero. That yields

$$\int_0^1 (\theta - \hat{\theta}) \theta^{k \sum_{j=1}^k w_j I_{(Y_j=0)}} (1-\theta)^{k \sum_{j=1}^k w_j I_{(Y_j=1)}} d\theta = 0.$$

Solving for $\hat{\theta}$,

$$\hat{\theta} = \frac{\int_0^1 \theta^{1+k \sum_{j=1}^k w_j I_{(Y_j=0)}} (1-\theta)^{k \sum_{j=1}^k w_j I_{(Y_j=1)}} d\theta}{\int_0^1 \theta^{k \sum_{j=1}^k w_j I_{(Y_j=0)}} (1-\theta)^{k \sum_{j=1}^k w_j I_{(Y_j=1)}} d\theta}.$$

The above can be re-written in terms of beta functions,

$$\hat{\theta} = \frac{B(2 + k \sum_{j=1}^k w_j I_{(Y_j=0)}, 1 + k \sum_{j=1}^k w_j I_{(Y_j=1)})}{B(1 + k \sum_{j=1}^k w_j I_{(Y_j=0)}, 1 + k \sum_{j=1}^k w_j I_{(Y_j=1)})}.$$

The above is simplified to the given result by expressing the beta functions in terms of gamma functions, and using the rule that $\Gamma(n) = (n-1)\Gamma(n-1)$.

Proof of (8) *for relative entropy* R:
The objective of (3) with

$$R = \theta \log \frac{\theta}{\hat{\theta}} + (1-\theta) \log \frac{1-\theta}{1-\hat{\theta}}$$

is convex in $\hat{\theta}$, and thus it suffices to compute its first derivative with respect to $\hat{\theta}$ and set to zero. That yields

$$\int_0^1 (-\frac{\theta}{\hat{\theta}} + \frac{1-\theta}{1-\hat{\theta}}) \theta^{k \sum_{j=1}^k w_j I_{(Y_j=0)}} (1-\theta)^{k \sum_{j=1}^k w_j I_{(Y_j=1)}} d\theta = 0.$$

This is simplified using beta functions (and then gamma functions) to

$$\frac{\hat{\theta}}{1-\hat{\theta}} = \frac{B(2 + k \sum_{j=1}^k w_j I_{(Y_j=0)}, 1 + k \sum_{j=1}^k w_j I_{(Y_j=1)})}{B(1 + k \sum_{j=1}^k w_j I_{(Y_j=0)}, 1 + k \sum_{j=1}^k w_j I_{(Y_j=1)})}$$

$$= \frac{1 + k \sum_{j=1}^k w_j I_{(Y_j=0)}}{1 + k \sum_{j=1}^k w_j I_{(Y_j=1)}}.$$

The above ratio can be solved for $\hat{\theta}$, yielding the given result.

A. REFERENCES

[1] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer-Verlag, New York, 2001.

[2] M. R. Gupta and R. M. Gray, "Reducing bias in supervised learning," *Proceedings of the IEEE Workshop on Statistical Signal Processing*, pp. 482–485, 2003.

[3] M. R. Gupta, R. M. Gray, and R. A. Olshen, "Nonparametric supervised learning with linear interpolation and maximum entropy," *submitted for journal publication, draft available upon request.*

[4] D. J. C. MacKay, *Information Theory, Inference, and Learning Algorithms*, Cambridge University Press, Cambridge, 2003.

[5] E. L. Lehmann and G. Casella, *Theory of Point Estimation*, Springer, New York, 1998.

[6] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*, Prentice Hall, New Jersey, 1993.

[7] J. M. Bernardo and A. F. M. Smith, *Bayesian Theory*, Wiley Series in Probability and Statistics, England, 2000.

[8] F. Provost and P. Domingos, "Tree induction for probability-based rankings," *Machine Learning*, vol. 52, no. 3, 2003.

[9] R. Rosenfeld, "Two decades of statistical language modeling: Where do we go from here," *Proceedings of the IEEE*, vol. 88, no. 8, 2000.

[10] J. H. Friedman, "On bias, variance, 0/1 loss, and the curse-of-dimensionality," *Data mining and knowledge discovery*, vol. 1, no. 1, pp. 55–77, 1997.