On Nearest Neighbor Classification Using Adaptive Choice of k

Anil K. Ghosh

Nearest neighbor classification is one of the simplest and popular methods for statistical pattern recognition. It classifies an observation $\mathbf{x}$ to the class, which is the most frequent in the neighborhood of $\mathbf{x}$. The size of this neighborhood is usually determined by a predefined parameter k. Normally, one uses cross-validation techniques to estimate the optimum value of this parameter, and that estimated value is used for classifying all observations. However, in classification problems, in addition to depending on the training sample, a good choice of k depends on the specific observation to be classified. Therefore, instead of using a fixed value of k over the entire measurement space, a spatially adaptive choice of k may be more useful in practice. This article presents one such adaptive nearest neighbor classification technique, where the value of k is selected depending on the distribution of competing classes in the vicinity of the observation to be classified. The utility of the proposed method has been illustrated using some simulated examples and well-known benchmark datasets. Asymptotic optimality of its misclassification rate has been derived under appropriate regularity conditions.

Key Words: Bayesian strength function; Cross-validation; Misclassification rate; Noninformative prior; Optimal Bayes risk; Posterior probability; p value; Robustness.

1. INTRODUCTION

In classification problems, one forms a finite partition of the sample space in order to classify an observation into one of J competing classes. The optimal Bayes rule (see, e.g., Anderson 1984) assigns an observation $\mathbf{x}$ to the class, which has the maximum posterior probability $p(\cdot \mid \mathbf{x})$. In practice, one uses the available training data to estimate these unknown posterior probabilities and classifies the observation based on those estimates. Nearest neighbor classifier (see, e.g., Fix and Hodges 1951; Cover and Hart 1967; Dasarathy 1991) assumes these posterior probabilities to be constant over a small neighborhood around $\mathbf{x}$ and estimates them using the proportions of different classes in the training sample lying in that neighborhood. As a result, the most frequent class in that neighborhood turns out to be the winner. Usually, a closed ball of radius $r_k(\mathbf{x})$ is taken as this neighborhood, where

Anil K. Ghosh is Assistant Professor, Department of Mathematics and Statistics, Indian Institute of Technology, Kanpur 208016, India (E-mail: *anilkghosh@rediffmail.com*).

Journal of Computational and Graphical Statistics, Volume 16, Number 2, Pages 482–502
DOI: 10.1198/106186007X208380

$r_k(\mathbf{x})$ is the distance between $\mathbf{x}$ and its kth nearest data point in the training sample, for k being a predefined positive integer.

Clearly, the shape and the size of this neighborhood and hence the classification result depend on the distance function and the value of the parameter k. Euclidean metric is the most popular choice for this distance function. Of course, when the measurement variables are not of comparable units and scales, it is a usual practice to standardize the dataset before using the Euclidean metric for classification. If the usual moment-based estimate of the pooled dispersion matrix is used for standardization, that leads to classification using Mahalanobis distance (see Mahalanobis 1936). However, one may use other flexible or adaptive distance functions (see, e.g., Friedman 1994; Hastie and Tibshirani 1996; Domeniconi, Peng, and Gunopulos 2002; Peng, Heisterkamp, and Dai 2004) as well.

The value of the parameter k that controls the size of the neighborhood has to be specified as well. Existing asymptotic results (see, e.g., Loftsgaarden and Quesenberry 1965; Cover and Hart 1967) suggest that the value of k should depend on the training sample size n, and it should vary with n in such a way that $k \to \infty$ and $k/n \to 0$ as $n \to \infty$. However, when the sample size is small or moderately large, there is no theoretical guideline for choosing the value of k. The optimum value of k depends on the specific dataset, and one normally uses the usual cross-validation technique (see, e.g., Stone 1977; Ripley 1996) to estimate it. However, one can also use the likelihood cross-validation (LCV) method (see, e.g., Silverman 1986 for description of LCV in the context of kernel density estimation) for this purpose. It estimates the optimum value of k by maximizing the log-likelihood score $\sum_{t=1}^{n} \log\{p_{-t}(c_t \mid \mathbf{x}_t)\}$, where c_t is the class label of the observation $\mathbf{x}_t$, and $p_{-t}(j \mid \mathbf{x}_t)$ is the posterior probability estimate for the jth class at $\mathbf{x}_t$, when $\mathbf{x}_t$ is not used as a training data point. Holmes and Adams (2002, 2003) used a slightly different version this likelihood criterion for their aggregation techniques. Both cross-validation and likelihood cross-validation methods use the training data to select a single value of k, and then that selected value is used for classifying all observations. However, one should note that in classification problems, in addition to depending on the entire training sample, a good choice of k depends on the specific observation to be classified. Therefore, in practice, adaptive choice of k may be more useful for classification than using a fixed k over the entire measurement space. In this article, we propose one such adaptive nearest neighbor classifier, which adopts a sequential type approach (discussed in Section 3) to select the adaptive value of k depending on the distribution of competing classes in the vicinity of the observation to be classified. Throughout this article, we assume the continuity of all population density functions and use the Euclidean distance for classification of all simulated datasets. For the analysis of benchmark datasets, in addition to Euclidean distance, we use the discriminant adaptive nearest neighbor (DANN) metric suggested by Hastie and Tibshirani (1996). However, the description of the adaptive nearest neighbor classifier given in the next section will make it quite clear that the use of the proposed method is not restricted to any special type of distance function, and one may use any other distance function as well.

2. DESCRIPTION OF THE METHODOLOGY

Recall that in usual nearest neighbor classification, one assumes the posterior probabilities $p(j \mid \mathbf{x})$ $(j = 1, 2, \ldots, J)$ to be constant over a neighborhood around $\mathbf{x}$. Here, we will deviate from this usual notion, and instead of assuming $p(j \mid \mathbf{x})$'s to be fixed and nonrandom, we will assume a prior distribution of $p(1 \mid \mathbf{x}), p(2 \mid \mathbf{x}), \ldots, p(J \mid \mathbf{x})$. Since it is quite evident that the calculations have to be done at each $\mathbf{x}$ separately, for convenience of our notation, we will drop the term $\mathbf{x}$, and denote $p(j \mid \mathbf{x})$ by p_j. Likewise, the vector of posterior probabilities $(p_1, p_2, \ldots, p_J)$ $[\sum_{j=1}^{J} p_j = 1]$ will be denoted by $\mathbf{p}$.

2.1 BAYESIAN STRENGTH FUNCTION

Suppose that for some given k, $\xi_k(\mathbf{p})$ is the prior distribution of $\mathbf{p}$ in the ball of radius $r_k(\mathbf{x})$ around $\mathbf{x}$. If t_{j_k} of these k neighbors come from the jth class $(j = 1, 2, \ldots, J)$, the conditional distribution of $\mathbf{t}_k = (t_{1_k}, t_{2_k}, \ldots, t_{J_k})$ $[\sum_{j=1}^{J} t_{j_k} = k]$ for given $\mathbf{p}$ is multinomial, and its probability mass function can be expressed as

$$\psi_k(\mathbf{t}_k \mid \mathbf{p}) = \frac{k!}{t_{1_k}! t_{2_k}! \ldots t_{J_k}!} \prod_{j=1}^{J} p_j^{t_{j_k}}.$$

Therefore, for some fixed k and $\mathbf{t}_k$, the conditional distribution of $\mathbf{p}$ is given by

$$\zeta_k(\mathbf{p} \mid \mathbf{t}_k) = \xi_k(\mathbf{p})\, \psi_k(\mathbf{t}_k \mid \mathbf{p}) \Big/ \int \xi_k(\mathbf{p})\, \psi_k(\mathbf{t}_k \mid \mathbf{p})\, d\mathbf{p}.$$

In a two-class problem, given the value of t_{1k} and t_{2k} $(t_{1k} + t_{2k} = k)$, one will naturally prefer the first class as compared to the second one if $P(p_1 > p_2 \mid \mathbf{t}_k) > P(p_2 > p_1 \mid \mathbf{t}_k)$, and the evidence in favor of the first class will increase with the probability function $P(p_1 > p_2 \mid \mathbf{t}_k)$. Following a similar idea, one can use the conditional distribution $\zeta_k(\mathbf{p} \mid \mathbf{t}_k)$ to define the Bayesian measure of strength for different populations, where the strength function for the jth $(j = 1, 2, \ldots, J)$ population is given by

$$S_k(j) = P\{\arg \max_{1 \le r \le J} p_r = j \mid \mathbf{t}_k\} = \int_{p_j = \max\{p_1, p_2, \ldots, p_J\}} \zeta_k(\mathbf{p} \mid \mathbf{t}_k)\, d\mathbf{p}.$$

It is quite transparent from the definition that $0 \le S_k(j) \le 1$ and $\sum_{j=1}^{J} S_k(j) = 1$ if $\xi_k(\mathbf{p})$ is the probability density function of a continuous distribution. Another interesting property of the Bayesian strength function is given in the following theorem.

Theorem 1. *If the prior distribution $\xi_k(\mathbf{p})$ is symmetric in $p_1, p_2, \ldots, p_J$, for all $i \ne j$, $S_k(j) \ge S_k(i)$ if and only if $t_{jk} \ge t_{ik}$.*

This theorem gives a Bayesian interpretation of the usual nearest neighbor classification. Note that t_{jk}/k is the usual nearest neighbor estimate of the posterior probability of the jth class. Bayesian strength functions preserve the ordering of these posterior probability estimates. Therefore, for any fixed k, if one goes for classification based on the values of $S_k(j)$, it will lead to the same result as that of the usual k-nearest neighbor classifier. Note

that the value of the strength function depends on the prior distribution $\xi_k(\mathbf{p})$ as well, and it has to be chosen appropriately. Uniform prior distribution is the most convenient one to handle with. Not only it makes the computation of $S_k(j)$ simpler (computational advantage will be discussed in Section 3), but it is also non-informative and gives no preference to any of the classes. Throughout this article, for all data analytic purpose, we will assume that $\xi_k(\mathbf{p})$ is uniform for all values of k.

2.2 Adaptive Nearest Neighbor Classifier

As we have mentioned before, unlike usual nearest neighbor classification, here we select the value of k depending on the observation to be classified. For classifying an observation, we study the results for a sequence of values of k and choose the one that leads to the maximum strength in favor of the winning class. However, it is not useful to consider all possible values of k. The use of very large values not only increases the computational cost, but also fails to represent the local pattern of the measurement space. So, when the training samples from competing classes are not of comparable sizes, the use of large k leads to decisions in favor of larger classes. Therefore, one should set some reasonable upper bound for k. This upper bound is expected to increase with the sample size n. In nearest neighbor classification, for the consistency of posterior probability estimates, one needs to shrink the neighborhood to zero with the increasing sample size. So, it is somewhat reasonable to use a function β_n of n as the upper bound, which increases with n, but β_n/n tends to zero as n tends to infinity.

This adaptive nearest neighbor classifier has an interesting interpretation in terms of p values as well. In a two-class problem, for a given value of k, except for a few tied cases (possible only if k is even), one always observes a difference between the estimated posterior probabilities of the two classes. But from a statistical point of view, it is important to know whether that difference is really there in terms of their true posteriors. Clearly, this leads to the hypothesis testing problem for binomial proportion $H_0 : \theta > 0.5$ against $H_a : \theta \leq 0.5$, where θ is true posterior probability for class-1 at the query point $\mathbf{x}$. If t_1 out of k neighbors of $\mathbf{x}$ come from the first population, the quantity $\sum_{t<t_1}\binom{k}{t}(0.5)^t$ or $\sum_{t\leq t_1}\binom{k}{t}(0.5)^t$ can be viewed as a one-sided p value for this above testing problem, and this serves as a measure of evidence in favor of the first population. The following theorem on the Bayesian strength function gives an interesting result in this context.

Theorem 2. *In a two-class problem, if $\xi_k(\mathbf{p})$ is uniform, and t_1 out of k nearest neighbors of $\mathbf{x}$ come form the first population, we have*

$$\sum_{t<t_1}\binom{k}{t}(0.5)^t < S_k(1) < \sum_{t\leq t_1}\binom{k}{t}(0.5)^t.$$

Therefore, in a two-class problem, the Bayesian strength function can be viewed as a p value type measure, and its natural extension for multiclass problems can be interpreted as a multiclass analog for a p value type function. In hypothesis testing problems, we rely more on the decision when the corresponding p value is very close to 0 or 1. In adaptive

nearest neighbor classification, we follow a similar idea and use the value of k that leads to the highest strength in favor of the winning population. Unlike usual nearest neighbor classification, here we do not need to go for cross-validation type algorithm to predefine the value of k, and that leads to a substantial saving in computational cost, especially when the training sample is large.

2.3 LARGE SAMPLE PROPERTIES OF MISCLASSIFICATION RATE

In this section, we study the asymptotic behavior of the average misclassification rate of the proposed adaptive nearest neighbor classifier. For this asymptotic analysis, we will assume that the training sample sizes $n_1, n_2, \ldots, n_J$ of different classes are of the same asymptotic order $O(n)$. In other words, for every $j = 1, 2, \ldots, J$, we will assume that $n_j/n \to \lambda_j$ $(0 < \lambda_j < 1)$ as $n = \sum n_j \to \infty$. Note that the error rate of the adaptive nearest neighbor classifier depends on the upper bound β_n as well. If n is the training sample size, β_n is the upper bound, and $d_{n,\beta_n} : R^d \to \{1, 2, \ldots, J\}$ is the corresponding adaptive nearest neighbor classifier, its average misclassification probability (Δ_{n,β_n}) is given by

$$\Delta_{n,\beta_n} = \sum_{j=1}^{J} \pi_j \; P\{d_{n,\beta_n}(\mathbf{X}) \neq j \mid \mathbf{X} \in j\text{th population}\},$$

where π_j $(j = 1, 2, \ldots, J)$ is the prior probability of the jth class. In the previous section, we have discussed about possible choices of β_n. In the next theorem we will see that an appropriate choice of this upper bound leads to the asymptotic optimality of the average misclassification rate.

Theorem 3. *Suppose that all competing populations have continuous probability density functions $f_1, f_2, \ldots, f_J$, and $\{\beta_n : n \geq 1\}$ is a sequence of positive integers such that $\beta_n \to \infty$ and $\beta_n/n \to 0$ as $n \to \infty$. Then, Δ_{n,β_n} converges (in probability) to the optimal Bayes risk as n tends to infinity.*

3. DATA ANALYTIC IMPLEMENTATION: A SEQUENTIAL TECHNIQUE

For finding the optimum value of k at $\mathbf{x}$, we adopt a sequential approach. We start with the data point nearest to $\mathbf{x}$ and then gradually increase the value of k by considering other data points one by one according to their distances from $\mathbf{x}$. At each stage, we compute strength functions for different populations and keep track of the population which attained the maximum strength up to that stage. Given the value of the upper bound β_n and the first k neighbors, one can easily compute the largest possible strength in future in favor of other populations. Clearly, the largest value of this future strength will be attained if the rest of the observations come from the second best class (i.e., the class which has the maximum number of representatives among the challengers at that stage). If this largest possible value is smaller than the observed maximum strength up to that stage, one can stop there and classify the observation to the class that achieved the maximum Bayesian

strength. For instance, in a two-class problem, if β_n is taken as 5, and the first two neighbors of $\mathbf{x}$ come from the first population, one can stop there and take the decision in favor of the first population. Note that given the first two neighbors of $\mathbf{x}$, the second population can at most enjoy the 3-2 majority, but the strength in favor of the winning class in that case is less than that in the 2-0 case. Therefore, there is no need to go for further calculations. However, one should note that the stopping value for k may not always be the same as the value of k that leads to the final classification. For instance, in the above example, if β_n is taken as 10, the second population will still have an opportunity to attain larger strength by enjoying 7-3 or 8-2 majority. So, instead of stopping at $k = 2$, we have to consider other neighbors in this case. Suppose the third neighbor comes from the second class and fourth from the first class, again fifth from the second class and sixth from the first. Now, the second class can at most enjoy the 6-4 majority but the strength in favor of the winning class in that case is smaller than that observed in the 2-0 case earlier. Therefore, we will stop after considering six neighbors and classify the observation to the first class for having the maximum strength in the 2-0 case. So, in this case, the stopping value of k is 6 but the final decision is taken on the basis of $k = 2$. One can notice that like weighted nearest neighbor classification techniques (see, e.g., Bailey and Jain 1978), not only the number of neighbors but also their ordering plays an important role in adaptive nearest neighbor classification.

This sequential approach of adaptive nearest neighbor classification has some similarities with the sequential probability ratio test (SPRT) (see, e.g., Wald 1973; Seigmund 1985). However, SPRT needs the values of the Type I and the Type II errors to be specified to determine the stopping criterion, but here given the value of β_n, the stopping rule is determined automatically. Further, this approach based on the Bayesian strength maximization leads to a natural extension of the sequential technique for multiclass problems.

Implementation of the adaptive nearest neighbor classifier requires the value of β_n to be specified. In the previous section, we have discussed about possible choices for β_n. Theorem 3 gives some idea in this context and suggests that $\beta_n = O(\sqrt{n})$ could be a good choice. Some other authors (see, e.g., Pal, Bandopadhyay, and Murthy 1998; Mitra, Murthy, and Pal 2002) used the same range of values of k for classification. In this article, we use β_n of the form $\beta_n = C\sqrt{n}$, where C is an appropriate constant. Our empirical experience suggests that the final result is reasonable and not very sensitive to the choice of C if it lies between 2 and 3. Throughout this article for all data analytic purpose, we use $\beta_n = 2\sqrt{n}$. However, if this value of β_n exceeds $n_0 = \min\{n_1, n_2 \ldots, n_J\}$, we take $\beta_n = n_0$. Though this choice of β_n is somewhat subjective, it led to fairly good results in our experiments, which we will see later in Sections 4 and 5.

In classification problems, when an observation lies well within the cluster of one population, it is quite expected that the observation should be classified with higher degree of confidence, and any sequential approach should reach the decision within a fewer number of steps. On the other hand, if the observation lies near the class boundary, classification of that point is expected to take more steps, and it is likely to be classified with lesser confidence. We observe similar phenomenon in adaptive nearest neighbor classification. In Figure 1, we consider three different examples with normal populations. In the left column we consider the case when two normal distributions have the same dispersion matrix $\mathbf{\Sigma}_1 = \mathbf{\Sigma}_2 = \mathbf{I_2}$, but

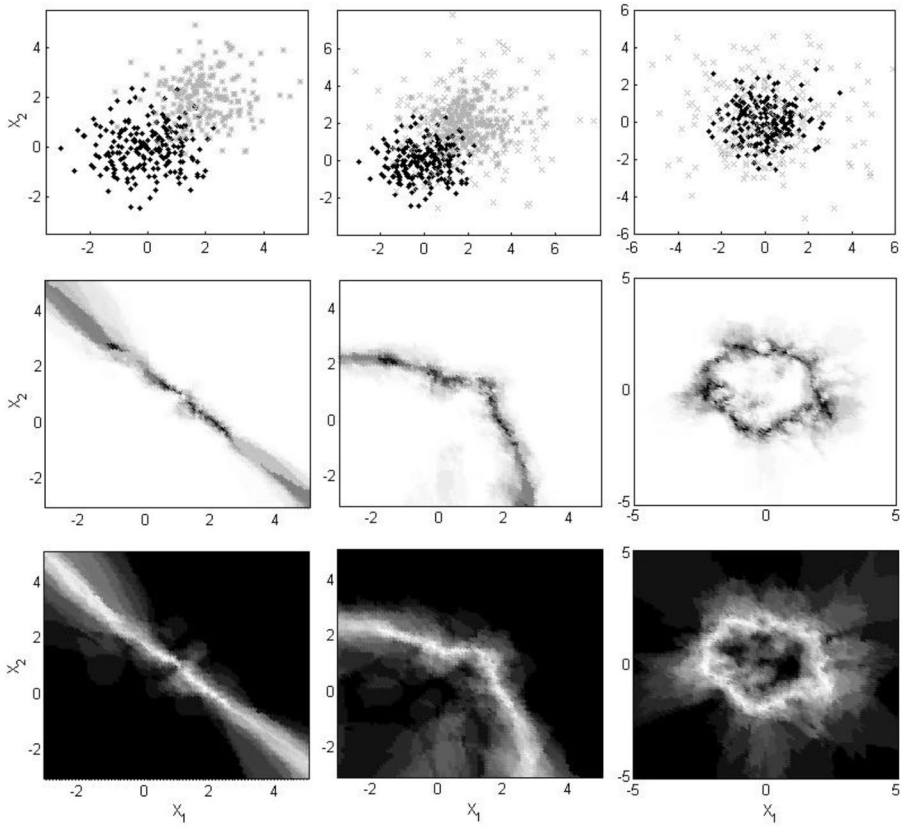

Figure 1. Top panel shows scatterplots for different datasets, middle panel presents gray scale values (rescaled) for maximum strength in favor of the winning class, and the bottom panel shows gray scale values (rescaled) for the number of neighbors required to reach the final decision.

they differ in their location parameters $\boldsymbol{\mu}_1 = (0, 0)$ and $\boldsymbol{\mu}_2 = (2, 2)$. We generate a sample of size 200 from each class and use it as the training sample. Figure 1 shows a scatterplot of this dataset (top of the left column), where dots and crosses represent the observations from these two populations. At the middle of the left column, for different values of $\mathbf{x}$, we plot the gray scale values of the maximum Bayesian strength (rescaled to have minimum value 0 and maximum value 1) in favor of the winning class at the time of final classification. Here, white color denotes the highest strength 1 and black color denotes the lowest strength 0. Intensity of the color varies with the magnitude of the rescaled strength function. As expected, we observe the values of the maximum strength function in favor of winning class to be lower near the class boundary $X_1 + X_2 = 2$, which is indicated by black shades near the $X_1 + X_2 = 2$ line in the figure. As we move away from the boundary, strength of the decision (or equivalently, the maximum strength in favor of the winning class) becomes higher, which is indicated by the white region in the plot. At the bottom of the left column, we present the gray scale values (rescaled) for the number of neighbors required to reach the final decision. From this gray scale plot, it is quite evident that the points near the class boundary require more information about their neighbors (or equivalently more steps in the sequential approach) for classification than that required by points away from the boundary.

We carry out a similar experiment with different choices of $\boldsymbol{\mu}_2$ and $\boldsymbol{\Sigma}_2$, while $\boldsymbol{\mu}_1$ and $\boldsymbol{\Sigma}_1$ are kept unchanged. In the middle column, we consider the case $\boldsymbol{\mu}_2 = (2, 2)$ and $\boldsymbol{\Sigma}_2 = 4\mathbf{I_2}$, and in the right column we consider $\boldsymbol{\mu}_2 = (0, 0)$ and $\boldsymbol{\Sigma}_2 = 4\mathbf{I_2}$. In all these cases we observe the same nature of the gray scale plots. Points near the class boundary require more steps in the sequential approach for being classified, but the strength in favor of the winning class is not so high in such cases. On the other hand, observations away from the class boundary require less number of steps to be classified with larger strength.

3.1 COMPUTATION OF BAYESIAN STRENGTH FUNCTION

To compute Bayesian strength functions for different populations, one can use any numerical integration method based on an appropriate averaging of the integrand over a suitable grid in the domain of integration (see Section 2.1 for the expression of $S_k(j)$). Given the value of k and the vector $\mathbf{t}_k$, the required number of computations for this approximation is proportional to γ^{J-1}, where γ is the number of grid points chosen on each axis, and J is the number of competing populations. Clearly, this computational cost grows up rapidly with J, and in the presence of several competing populations, it becomes computationally difficult to implement this method. In such cases one can resort to another approximation procedure. If $\xi_k(\mathbf{p})$ is uniform, it is easy to see that given the value of k and $\mathbf{t}_k$, $\mathbf{p}$ follows a Dirichlet distribution with parameters $k+J;\ t_{1k}+1, t_{2k}+1, \ldots, t_{Jk}+1\ (\sum_{j=1}^{J} t_{jk} = k)$. Therefore, instead of using any Markov chain Monte Carlo type algorithm (see, e.g., Gilks, Richardson, and Spiegelhalter, 1996), one can easily generate observations from the appropriate Dirichlet distribution to approximate Bayesian strengths for different populations. We know that if $X_1, X_2, \ldots, X_J$ are independent and $X_j \sim \text{Gamma}(t_{jk}+1)$ for $j = 1, 2, \ldots, J$, $(X_1/S, X_2/S, \ldots, X_J/S)$ jointly follows a Dirichlet distribution with parameters $k+J;\ t_{1k}+1, t_{2k}+1, \ldots, t_{Jk}+1\ (\sum_{j=1}^{J} t_{jk} = k)$, where $S = \sum_{j=1}^{J} X_j$. Therefore, $P\{p_j > p_i\ \forall i \neq j \mid \mathbf{t}_k\} = P\{X_j > X_i\ \forall i \neq j\}$, and hence the strength function can be computed by generating observations from gamma distributions. Now, noting the fact that a Gamma$(t_{jk}+1)$ variate is the sum of $t_{jk}+1$ independent and identically distributed exponential variables with unit mean, one can keep on generating observations from exponential distributions to get estimates of strength functions for different populations and for different values of k in a sequential way. For our data analytic purpose, we adopted the numerical integration method when $J \leq 3$. For $J \geq 4$, Bayesian strength functions were approximated using 10,000 sets of observations generated from appropriate gamma distributions.

4. NUMERICAL RESULTS ON SIMULATED DATASETS

In this section, we use some simulated datasets to study the performance of the proposed adaptive nearest neighbor classifier. For the sake of simplicity, here we begin with some two-class problems in two dimensions. Later in this section and also in the Section 5, we will analyze some high-dimensional simulated and benchmark datasets involving more than two classes.

4.1 EXAMPLES WITH GAUSSIAN AND CAUCHY DISTRIBUTIONS

We start with some examples, where each of the two populations is normally distributed and has the same prior probability 0.5. Here, we consider three different types of problems, where competing populations have (1) the same scatter matrix but different location parameters, (2) different location and scatter parameters, and (3) the same location parameter but different scatter matrices. The location parameter and the scatter matrix for the first population are always taken as $\boldsymbol{\mu}_1 = (0, 0)$ and $\boldsymbol{\Sigma}_1 = \mathbf{I_2}$ (the identity matrix), respectively, whereas those for the second population are of the form $\boldsymbol{\mu}_2 = (\mu, \mu)$ and $\boldsymbol{\Sigma}_2 = \sigma^2\mathbf{I_2}$. Taking the value of (μ, σ^2) as (2, 1), (2, 4), and (0, 4), we get three different types of datasets. Scatterplots for these three different types of datasets are given in the top panel of Figure 1. In each case, taking equal number of observations from the two classes, we generated training samples of size 50, 100, 200, and 400, while test sets of 1,000 observations (500 from each class) were used to evaluate the performance of our proposed classifier. For each of these examples, we generated 1,000 different training and test sets, and average test set misclassification rates over those 1,000 trials are reported in Table 1 along with their corresponding standard errors. Error rates are also reported for nearest neighbor classifiers that use a single value of k for classification. As we have discussed before, this value of k can be selected using the likelihood cross-validation (LCV) or the usual cross-validation method. In practice, the latter one often leads to multiple minimizers of the estimated error rate due to the stepwise nature of the estimated function. Since this classifier assumes the posterior probabilities to be locally constant, it is somewhat reasonable to consider the lowest value of the minimizer in such cases.

Note that in adaptive nearest neighbor classification, instead of concentrating on any fixed value of k, we generate an ensemble of classifiers and then choose the one that yields the maximum Bayesian strength in favor of the winning class. Instead of choosing one value of k, one can also consider the results for all values of k and use voting or other methods to aggregate these results to arrive at the final decision. One popular practice for this aggregation is to assign different weights to different classifiers based on their corresponding misclassification probabilities Δ and use those weights to build up the aggregated classifier. Bagging (see, e.g., Breiman 1996) and boosting (see, e.g, Schapire et al. 1998; Friedman, Hastie, and Tibshirani 2000), the two most well-known aggregation methods in the literature, adopt similar ideas for combining the results of several classifiers. Boosting uses the weight function $w = \log\{(1 - \Delta)/\Delta\}$ to put more weights on those classifiers that lead to lower misclassification rates. The weight function decreases gradually as the misclassification rate increases. However, bagging uses equal weights for all classifiers. A comparative empirical study of bagging, boosting and other ensemble methods can be found in Opitz and Maclin (1999).

These bagging and boosting methods use bootstrap (see, e.g., Efron 1993) or weighted bootstrap technique to generate different samples from the training data, and based on those different samples, different classifiers are developed. Results of these classification rules are aggregated using the weight function. However, for our method, we do not require any resampling techniques to construct the classifiers. Use of different values of k leads to

Table 1. Misclassification rates (in %) of different classification methods and their standard errors on simulated examples with normal and Cauchy distributions.

		Normal			Cauchy		
		$\mu = 2$, $\sigma^2 = 1$	$\mu = 2$, $\sigma^2 = 4$	$\mu = 0$, $\sigma^2 = 4$	$\mu = 2$, $\sigma^2 = 1$	$\mu = 2$, $\sigma^2 = 4$	$\mu = 0$, $\sigma^2 = 4$
n	Bayes risk	7.87	13.31	26.38	19.58	22.27	37.00
	LDA	8.39(0.03)	16.82(0.05)	48.44(0.08)	32.65(0.41)	35.13(0.37)	48.42(0.13)
	QDA	8.63(0.04)	14.60(0.05)	28.57(0.06)	42.61(0.29)	43.19(0.22)	46.44(0.15)
	LCV	8.87(0.05)	17.23(0.07)	38.80(0.18)	20.32(0.05)	27.57(0.13)	46.50(0.12)
50	Cross-valid.	9.22(0.05)	17.00(0.08)	35.94(0.13)	21.14(0.08)	27.01(0.09)	44.00(0.10)
	Voting	8.41(0.03)	19.50(0.08)	46.10(0.09)	20.89(0.05)	32.38(0.20)	47.28(0.07)
	Wt. Voting	8.42(0.03)	18.10(0.07)	36.72(0.19)	20.26(0.05)	27.28(0.09)	45.99(0.33)
	Adaptive	8.78(0.04)	16.10(0.06)	34.20(0.12)	20.42(0.05)	26.01(0.07)	43.04(0.09)
	LDA	8.14(0.03)	16.43(0.04)	48.88(0.07)	32.15(0.40)	38.26(0.36)	48.58(0.13)
	QDA	8.25(0.03)	13.90(0.04)	27.31(0.05)	46.20(0.22)	45.43(0.17)	47.53(0.12)
	LCV	8.50(0.03)	15.96(0.05)	32.98(0.11)	20.00(0.04)	25.59(0.06)	43.68(0.12)
100	Cross-valid.	8.68(0.03)	15.92(0.06)	32.80(0.10)	20.56(0.06)	25.81(0.07)	42.63(0.09)
	Voting	8.21(0.03)	19.15(0.06)	47.59(0.07)	20.26(0.05)	31.58(0.25)	47.80(0.06)
	Wt. Voting	8.20(0.03)	17.86(0.05)	33.66(0.10)	20.00(0.04)	26.42(0.06)	43.07(0.13)
	Adaptive	8.49(0.03)	15.23(0.04)	31.08(0.08)	20.28(0.05)	25.01(0.05)	41.58(0.08)
	LDA	8.01(0.03)	16.11(0.04)	49.17(0.06)	32.53(0.41)	38.59(0.36)	48.19(0.12)
	QDA	8.06(0.03)	13.54(0.03)	26.88(0.04)	48.38(0.14)	47.30(0.12)	47.85(0.08)
	LCV	8.21(0.03)	15.04(0.04)	29.73(0.06)	19.82(0.04)	24.70(0.05)	41.12(0.08)
200	Cross-valid.	8.35(0.03)	15.01(0.05)	30.13(0.07)	20.16(0.05)	24.83(0.06)	41.22(0.08)
	Voting	8.11(0.03)	18.94(0.05)	48.74(0.05)	19.86(0.04)	31.53(0.12)	48.68(0.04)
	Wt. Voting	8.10(0.03)	17.68(0.05)	31.56(0.08)	19.76(0.04)	26.42(0.06)	41.60(0.12)
	Adaptive	8.28(0.03)	14.56(0.04)	29.18(0.06)	20.04(0.04)	24.36(0.05)	40.56(0.07)
	LDA	7.93(0.03)	16.05(0.04)	49.42(0.06)	32.55(0.40)	38.29(0.34)	48.38(0.12)
	QDA	7.95(0.03)	13.47(0.03)	26.65(0.04)	49.08(0.12)	48.23(0.09)	48.60(0.06)
	LCV	8.03(0.03)	14.55(0.04)	28.37(0.05)	19.76(0.04)	24.29(0.05)	39.66(0.06)
400	Cross-valid.	8.16(0.03)	14.60(0.04)	28.54(0.05)	20.00(0.04)	24.43(0.05)	39.82(0.06)
	Voting	8.02(0.03)	18.83(0.04)	49.53(0.03)	19.72(0.04)	31.12(0.09)	49.17(0.03)
	Wt. Voting	8.02(0.03)	17.60(0.04)	29.88(0.06)	19.70(0.04)	26.09(0.05)	40.99(0.12)
	Adaptive	8.14(0.03)	14.29(0.04)	27.99(0.05)	19.96(0.04)	24.10(0.05)	39.57(0.06)

different classification results, which can be aggregated. Here we adopt the weight functions used by bagging (i.e., uniform weight) and boosting (i.e., weight proportional to logarithm of the odd ratio) for aggregation. This essentially leads to simple voting (in the case of bagging) and weighted voting (in the case of boosting) of all nearest neighbor classifiers. In Table 1, we report the misclassification rates for these two voting methods as well.

Since the datasets considered in this section are spherical in nature, without any standardization, we use the Euclidean metric for nearest neighbor classification. Error rates for linear discriminant analysis (LDA) and quadratic discriminant analysis (QDA), which

are optimum under the normality of underlying populations, are also given in Table 1. To facilitate the comparison, Bayes errors are reported as well.

Since the optimum classifiers are either linear or quadratic in these examples, it is very difficult for other classifiers to beat LDA or QDA in the respective cases. However, the adaptive nearest neighbor classifier performed quite well in all these examples. In almost all cases, its performance was significantly better than that of the usual nearest neighbor classifier with cross-validated choice of k. For the first set of examples, where the populations differ only in their location parameters, LCV method had a slight edge over the adaptive nearest neighbor classifier, but the differences between their error rates were statistically insignificant in most of the cases. On the other hand, on other two sets of examples, the proposed classifier achieved statistically significantly lower misclassification rates than those of LCV and usual cross-validation techniques. Both, the simple and the weighted voting methods, performed better than other nearest neighbor classifiers in the first set of examples, but in the other two cases, they had much higher error rates. In the third set of examples, the simple voting method led to misclassification rates of almost 50%.

We carried out similar experiments with Cauchy distributions, which have heavier tails than their normal counter parts. In the presence of these heavy tailed distributions, as expected, usual LDA and QDA could not perform well, and nearest neighbor classifiers clearly outperformed them. Among the nearest neighbor classifiers, LCV and the weighted voting method led to somewhat better performance when the populations differ only in their location parameters, but in other two cases, the proposed method had a clear edge over its competitors.

One can notice that with increasing training sample size, error rates of all these classifiers get decreased, but their ordering usually remains the same. So, in the next section, instead of using different sizes of training samples, we will use training sets of size 100 only. Since the performance of the weighted voting method turned out to be much better than simple voting, from the next section onwards we will not report the misclassification rates for the simple voting method.

4.2 More Examples With Simulated Datasets

In this section we analyze various types of simulated datasets that include examples with mixture normal, lognormal, uniform, and waveform data (see, e.g., Breiman et al. 1984). Here, we also analyze some noisy datasets to compare the performance of different nearest neighbor classifiers. Throughout this section, we use training sets of size 100 and test sets of size 1,000 for our analysis, and unless mentioned otherwise, we assume equal number of observations from all competing classes. In all these cases, we performed each experiment 1,000 times as before and report the average misclassification rates for different classifiers along with their corresponding standard errors (see Table 2). A brief description of the datasets that we use in this section is given below.

Dataset 1: Each class is an equal mixture of two bivariate normal distributions. All these normal populations have the same dispersion matrix $0.25\mathbf{I}_2$, but they differ in their location parameters. For the two components of class-1, these location parameters are (1, 1)

Table 2. Misclassification rates (in %) of different nearest neighbor classifiers and their standard errors on simulated datasets.

Datasets	d	J	LCV	Cross-valid.	Wt. Voting	Adaptive
Dataset-1	2	2	5.26 (0.03)	5.39 (0.03)	4.96 (0.02)	5.01 (0.02)
Dataset-2	2	2	6.28 (0.15)	4.97 (0.03)	24.45 (0.09)	4.72 (0.02)
Dataset-3	2	2	13.11 (0.04)	13.39 (0.05)	12.77 (0.04)	12.98(0.04)
Dataset-4	2	2	14.49 (0.15)	12.17 (0.05)	25.00 (0.00)	11.65 (0.06)
Dataset-5	2	2	11.00 (0.07)	10.28 (0.05)	12.40 (0.06)	9.69 (0.04)
Dataset-6	2	2	22.46 (0.10)	20.60 (0.08)	23.68 (0.10)	19.60 (0.06)
Dataset-7	12	2	38.15 (0.17)	35.10 (0.08)	37.56 (0.08)	34.18 (0.07)
Dataset-8	21	3	20.06 (0.06)	20.10 (0.07)	22.94 (0.08)	18.86 (0.06)
Dataset-9	40	3	20.12 (0.06)	20.20 (0.07)	23.12 (0.08)	18.92 (0.06)
Dataset-10	21	3	6.75 (0.05)	6.89 (0.05)	8.44(0.07)	6.24(0.05)

and $(-1, -1)$, whereas those for class-2 are $(1, -1)$ and $(-1, 1)$.

Dataset 2: Same as Dataset 1, but instead of taking equal number of observations from the two classes, they are taken in 3:1 ratio both for the training and the test sets.

Dataset 3: Differs from Dataset 1 only in the choice of location parameters. For the two components of class-1, these parameters are $(1, 1)$ and $(3, 3)$, whereas those for class-2 are $(2, 2)$ and $(4, 4)$.

Dataset 4: Same as Dataset 3, but instead of taking equal number of observations from the two classes, they are taken in 3:1 ratio both for the training and the test sets.

Dataset 5: For both of the two classes, logarithms of measurement variables are normally distributed with parameters $(0, 0, 1, 1, 0)$ and $(2, 2, 1, 1, 0)$, respectively.

Dataset 6: Each of the two populations is bivariate uniform on a circular disk around the origin. Class-1 is uniform on disk a radius 1, whereas class-2 is uniform on a disk of radius 4.

Dataset 7: Same as Dataset-6 but augmented with 10 independent standard normal variables, which are used as noise.

Dataset 8: This dataset is known as the waveform data (see, e.g., Breiman et al., 1984). For $i = 0, 1, \ldots, 20$, define $h_1(i) = \max\{6 - |i - 10|, 0\}$, $h_2(i) = h_1[(i - 4) \bmod 21]$ and $h_3(i) = h_1[(i + 4) \bmod 21]$. Suppose that $U \sim U(0, 1)$ and ϵ_i's ($i = 0, 1, \ldots, 20$) are iid $N(0, 1)$. Now, for three competing classes, measurement variables are distributed as

$$\text{Class-1} : X_i = Uh_1(i-1) + (1-U)h_2(i-1) + \epsilon_{i-1} \text{ for } i = 1, 2, \ldots, 21.$$
$$\text{Class-2} : X_i = Uh_1(i-1) + (1-U)h_3(i-1) + \epsilon_{i-1} \text{ for } i = 1, 2, \ldots, 21.$$
$$\text{Class-3} : X_i = Uh_2(i-1) + (1-U)h_3(i-1) + \epsilon_{i-1} \text{ for } i = 1, 2, \ldots, 21.$$

Dataset 9: Same as Dataset-8 but augmented with 19 independent standard normal variables, which are used as noise.

Dataset 10: A little variant of the waveform data with U replaced by U_i, where U_i's ($i = 1, 2, \ldots, 21$) are iid $U(0, 1)$ variables.

Adaptive nearest neighbor classifier produced excellent performance on these simulated datasets. On 8 out of 10 datasets, it led to the best error rate among the nearest neighbor classifiers considered here, and in all these cases, its performance was significantly better than its competitors. On the other two datasets (Dataset-1 and Dataset-3), the weighted voting method had the best error rate, but even in those cases, the adaptive nearest neighbor classifier did a reasonably good job. On those datasets, it could nearly match the performance of the weighted voting method and significantly outperformed LCV and usual cross-validation techniques.

5. ANALYSIS OF BENCHMARK DATASETS

In this section, we analyze some well-known benchmark datasets for further illustration of the proposed method. Two of these datasets (the synthetic data and the vowel data) have specific training and test samples. In all other cases, we formed the training set and the test set by randomly partitioning the data in such a way that the proportions of different classes in these two sets are as close as possible. For different datasets, sizes of the training and the test sets for each partition are given in Table 3. For each of these datasets, this random partitioning was carried out 1,000 times to generate 1,000 different training and test samples. Average test set misclassification rates over those 1,000 trials are reported (see Table 3) for different nearest neighbor classifiers along with their corresponding standard errors. For the synthetic data and the vowel data, which have separate training and test sets, we report the test set misclassification rates for different classifiers. If a classifier leads to a test set error rate p, the corresponding standard error is taken as $\sqrt{p(1-p)/m}$, where m is the size of the test sample. For the analysis of these benchmark datasets, in addition to Euclidean distance, we use the discriminant adaptive nearest neighbor metric (DANN) proposed by Hastie and Tibshirani (1996), and in Table 3, we report the results for both types of distance functions. For the DANN metric, we use $\epsilon = 1$ and iter $= 1$, which were reported to perform well by Hastie and Tibshirani (1996). We also follow their guideline to choose the initial parameter $K_M = 50$, but in cases of salmon data and urine crystal data since there are only 50 observations in the training set, $K_M = 40$ is used. Throughout this section, sample proportions for different classes are used as their prior probabilities.

In cases of Iris, salmon, synthetic, vowel, and kangaroo skull measurement data, measurement variables are of comparable units and scales. In these cases, we use both standardized and unstandardized versions of the dataset for classification. In all other cases, we use standardized datasets only, where the usual moment-based estimate of the pooled dispersion matrix is used for standardization. Apart from the salmon data, all other datasets that we use in this section and their descriptions are available either at UCI Machine Learning Repository (*http://www.ics.uci.edu*) or at CMU Data Archive (*http://lib.stat.cmu.edu*). Some of these datasets are also available in Andrews and Herzberg (1985). Salmon data is taken from the book by Johnson and Wichern (1992). In some of these datasets

Table 3. Misclassification rates (in %) of different nearest neighbor classifiers and their standard errors on benchmark datasets.

			Sample size		Nearest neighbor classifiers			
Datasets	d	J	Train	Test	LCV	Cross-valid.	Wt. Voting	Adaptive
Salmon	2	2	50	50	8.36 (0.10)	8.89 (0.10)	7.76 (0.10)	8.23 (009)
					8.14 (0.10)	8.98 (0.11)	8.01 (0.10)	8.19 (0.09)
Salmon*	2	2	50	50	9.24 (0.10)	10.09 (0.11)	8.92 (0.11)	9.16 (0.09)
					8.20 (0.10)	9.08 (0.11)	8.08 (0.10)	8.21 (0.09)
Synthetic•	2	2	250	1000	10.00 (0.95)	11.70 (1.02)	10.50 (0.97)	9.20 (0.91)
					9.70 (0.94)	13.10 (1.07)	10.60 (0.97)	9.40 (0.92)
Synthetic*,•	2	2	250	1000	9.20 (0.91)	8.70 (0.89)	8.30 (0.87)	9.40 (0.92)
					9.70 (0.94)	9.60 (0.93)	9.80 (0.94)	10.30 (0.96)
Iris	4	3	75	75	2.74 (0.05)	2.88 (0.05)	2.59 (0.05)	2.36 (0.04)
					2.95 (0.05)	3.22 (0.06)	2.70 (0.05)	2.71 (0.05)
Iris*	4	3	75	75	3.93 (0.06)	4.22 (0.06)	5.56 (0.08)	3.80 (0.06)
					3.10 (0.05)	3.32 (0.05)	2.72 (0.05)	2.79 (0.05)
Biomedical	4	2	100	94	18.58 (0.11)	17.56 (0.10)	23.27 (0.10)	16.67 (0.09)
					18.49 (0.13)	14.60 (0.09)	22.71 (0.12)	13.98 (0.09)
Diabetes	5	3	100	45	14.97 (0.23)	10.30 (0.13)	12.26 (0.14)	9.68 (0.12)
					15.08 (0.29)	9.04 (0.12)	12.20 (0.14)	9.47 (0.12)
Crab	5	4	100	100	6.54 (0.07)	6.60 (0.07)	6.01 (0.06)	6.28 (0.06)
					8.00 (0.07)	8.13 (0.07)	8.05 (0.08)	7.70 (0.07)
Urine Crystal	6	2	50	27	27.32 (0.21)	28.15 (0.22)	27.04 (0.18)	27.20 (0.19)
					28.38 (0.22)	29.29 (0.23)	29.23 (0.19)	29.36 (0.23)

* Results are based on unstandardized dataset. • Dataset has specific training and test samples.

Figures in top and bottom rows represent the result based on Euclidean distance and Hastie-Tibshirani's discriminant adaptive nearest neighbor (DANN) metric, respectively.

continued on next page

(biomedical, urine crystal, and kangaroo-skull measurement data), we have some observations with missing values. We do not consider them and carry out our analysis with the remaining observations. We consider two different problems on kangaroo skull measurement data—(1) identification of species and (2) identification of sex, and in Table 3, we denote them as kangaroo-species and kangaroo-sex, respectively. For the vehicle data, though there are originally 946 observations, we can use only 846, which are available at

Table 3. Continued.

			Sample size		Nearest neighbor classifiers			
Datasets	d	J	Train	Test	LCV	Cross-valid.	Wt. Voting	Adaptive
Liver Disorder	6	2	200	145	33.26 (0.11)	33.78 (0.12)	35.64 (0.10)	31.60 (0.11)
					30.56 (0.10)	30.90 (0.11)	32.09 (0.09)	30.07 (0.10)
Pima Indian	8	2	400	368	25.45 (0.05)	25.58 (0.05)	34.05 (0.03)	24.88 (0.05)
					26.76 (0.05)	25.80 (0.05)	33.80 (0.02)	25.52 (0.05)
Vowel$^{\bullet}$	10	11	528	462	46.75 (2.32)	46.75 (2.32)	45.02 (2.31)	46.97 (2.32)
					39.83 (2.28)	39.61 (2.28)	40.04 (2.28)	39.83 (2.28)
Vowel$^{*,\bullet}$	10	11	528	462	42.64 (2.30)	43.72 (2.31)	40.69 (2.29)	38.10 (2.26)
					37.45 (2.25)	39.18 (2.27)	36.58 (2.24)	37.88 (2.26)
Wine	13	3	100	78	2.14 (0.05)	2.18 (0.05)	1.89 (0.05)	1.91 (0.05)
					3.63 (0.11)	3.61 (0.08)	3.22 (0.07)	3.69 (0.07)
Kangaroo-sex	18	2	75	26	36.32 (0.22)	36.10 (0.23)	37.02 (0.18)	36.17 (0.23)
					37.01 (0.20)	36.86 (0.24)	36.73 (0.21)	39.94 (0.28)
Kangaroo-sex*	18	2	75	26	30.40 (0.21)	32.04 (0.23)	29.92 (0.19)	28.95 (0.23)
					32.49 (0.21)	32.60 (0.23)	32.68 (0.21)	31.82 (0.25)
Kangaroo-species	18	3	75	26	21.32 (0.18)	21.86 (0.19)	23.93 (0.16)	20.56 (0.18)
					31.07 (0.23)	31.60 (0.24)	31.30 (0.22)	34.78 (0.27)
Kangaroo-species*	18	3	75	26	34.03 (0.28)	28.17 (0.24)	30.71 (0.26)	30.28 (0.25)
					26.78 (0.21)	26.11 (0.22)	28.79 (0.18)	25.27 (0.21)
Vehicle	18	4	400	446	21.79 (0.05)	21.81(0.06)	23.76 (0.05)	21.12 (0.05)
					20.92 (0.07)	20.00 (0.06)	21.61 (0.06)	19.72 (0.05)

* Results are based on unstandardized dataset. $^{\bullet}$ Dataset has specific training and test samples.

Figures in top and bottom rows represent the result based on Euclidean distance and Hastie-Tibshirani's discriminant adaptive nearest neighbor (DANN) metric, respectively.

UCI machine learning repository. Since these datasets and their descriptions are available from books and online sources, to save space, we choose not to give these descriptions here.

On these benchmark datasets, overall performance of the adaptive nearest neighbor classifier was far better than that of LCV, weighted voting and usual cross-validation technique. When the Euclidean distance was used for classification, in 11 out of 20 cases, it led to the best error rate among the nearest neighbor classifiers considered here. In most of these cases, its misclassification rate was significantly lower than that of its competitors. Performance of this proposed classifier was fairly competitive on other datasets as well. In not less than seven occasions, its error rate was second lowest among the nearest neighbor

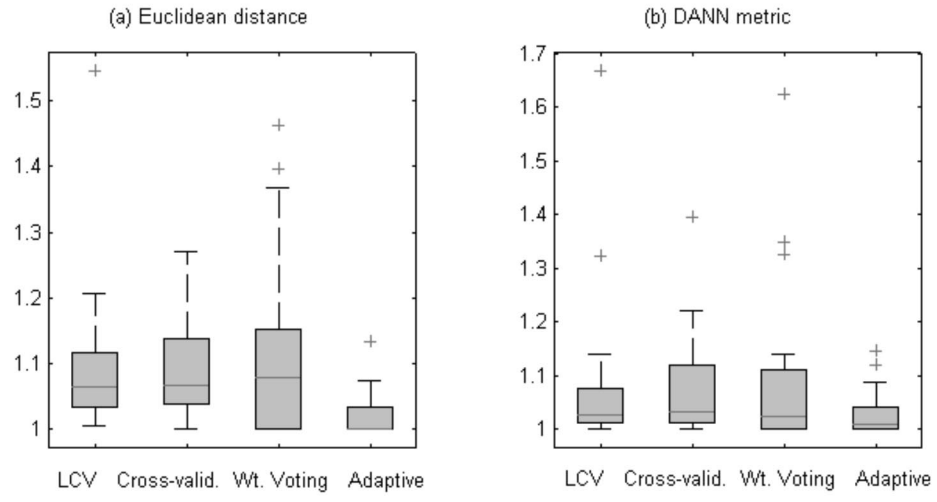

Figure 2. Robustness of different nearest neighbor classifiers in classification of benchmark datasets.

classifiers. Our proposed method showed its superiority in the case of discriminant adaptive nearest neighbor (DANN) metric as well. In 9 out of 20 cases, it yielded the best error rate. Apart from the unstandardized version of the kangaroo skull measurement data, in all other cases, its performance was quite competitive.

To study the robustness of different nearest neighbor classifiers across different datasets, we use the idea of Friedman (1994). In a particular example, robustness of any particular method t is defined by the misclassification probability ratio $r_t = \Delta_t/\Delta_0$, where Δ_t is the misclassification rate of the tth classifier and $\Delta_0 = \min_t \Delta_t$. Clearly, in an example, the best classifier will have $r_t = 1$, while other classifiers will have $r_t > 1$. Higher values of r_t indicate the lack of robustness of the classifier t. In each of these above examples, we compute this ratio for all classifiers, both for the Euclidean distance and the DANN metric, and they are graphically represented by box plots in Figure 2. Superiority of the adaptive nearest neighbor classifier is quite transparent from this figure.

Sometimes in the presence of large training datasets, it becomes computationally very difficult to use voting or weighted voting over all values of k. In such cases, instead of considering all values of k, one may restrict to classifiers with $k \le k_0$ for some suitable choice of k_0. Ghosh, Chaudhuri, and Murthy (2006) used $k_0 = \sqrt{n}$ for classification using nearest neighbor density estimates. If k_0 is such that $k_0 \log k_0 = o(n)$, for classification of a single observation, this truncated voting (or truncated weighted voting with known weights) scheme requires only $O(n)$ computations (see, e.g., Aho, Hopcroft, and Ullman 1974 for computational complexity of order statistics) as compared to $O(n \log n)$ required by voting (or weighted voting) over all values of k. Here we use $k_0 = \beta_n$ and report the result for the truncated weighted voting method. Figure 3 gives the box plot for the relative increase (RI) in error rates (as compared to the adaptive nearest neighbor classifier) for these two types of weighted voting: usual and truncated. RI is defined as ratio $(\Delta_w - \Delta_a)/\Delta_a$, where Δ_w and Δ_a stand for the error rate of the weighted voting method (either usual or truncated)

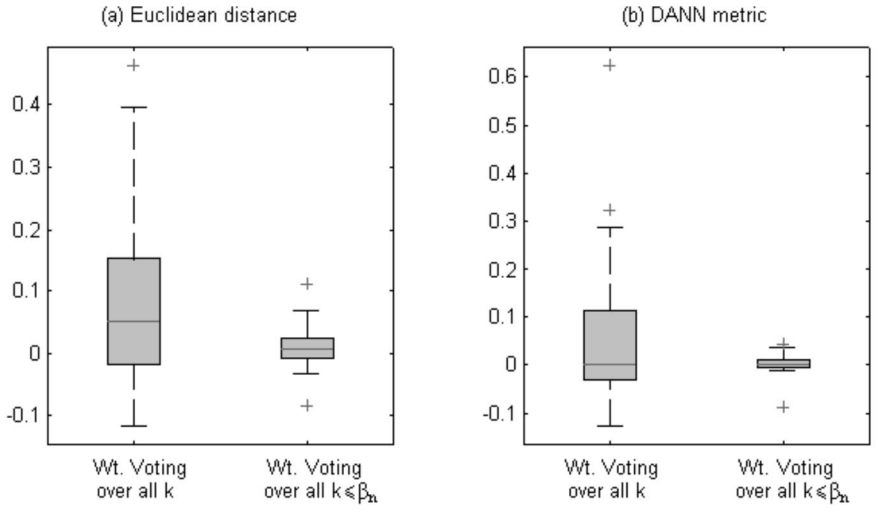

Figure 3. Relative increase in error rates (RI) due to weighted voting.

and that of the adaptive nearest neighbor classifier, respectively. From this figure, it is quite clear that truncation helped to improve the performance of the weighted voting method. Overall performance of the truncated weighted voting method was quite comparable to that of the adaptive classifier, but the latter one had a slight edge. This type of truncation in voting scheme helped to improve the performance of the simple voting method as well, but since their error rates were higher than those of their corresponding weighted voting counterparts, we do not report them here.

6. CONCLUDING REMARKS

In nearest neighbor classification, it is a common practice to use cross-validation or other resampling methods to select a value of k from the training data and to use that value for classification of all observations. When data clouds are of very regular shape (e.g., in the case of normally distributed data), these classification methods based on a single value of k may produce comparable misclassification rates, but in real life, it is rarely the case. One can always have a mixture of two or more datasets, where optimum values of k for these components are very different. In such cases, use of a single value of k will not work well (see, e.g., Wettschereck and Dietterich 1994). The proposed adaptive nearest neighbor classification method handles these situations well by selecting the value of k based on the observation to be classified. This spatially adaptive choice of k adds more flexibility to the classification methodology. In most of the simulated and the benchmark datasets that we consider in this article, it significantly outperformed its competitors. In view of the above data analysis, adaptive nearest neighbor classification seems to be better than using a single value of k. Throughout this article, we have used $\beta_n = 2\sqrt{n}$ as the upper bound of k for the proposed method. This choice of β_n is mainly motivated by the asymptotic result in Theorem 3. Though this choice is somewhat subjective, in all simulated and benchmark

datasets, it led to satisfactory results. However, this may not be the optimum choice for some of the datasets, where it leaves a room for further improvement in the performance of the adaptive nearest neighbor classifier.

APPENDIX

Proof of Theorem 1: Since $\xi_k(\mathbf{p})$ is symmetric in its arguments, interchanging p_i and p_j in the integrand, one obtains

$$\int_{p_i=\max\{p_1,p_2,\ldots,p_J\}} \left(\prod_{m=1}^{J} p_m^{t_m}\right) \xi_k(\mathbf{p})\, d\mathbf{p} = \int_{p_j=\max\{p_1,p_2,\ldots,p_J\}} \left(\prod_{\substack{m=1\\ m\neq i,j}}^{J} p_m^{t_m}\right) p_i^{t_j}\, p_j^{t_i}\, \xi_k(\mathbf{p})\, d\mathbf{p}.$$

Now, note that $S_k(j)$ and $S_k(i)$ have the same denominator, which is positive, and the numerator of

$$S_k(j) - S_k(i) = \int_{p_j=\max\{p_1,p_2,\ldots,p_J\}} \left(\prod_{\substack{m=1\\ m\neq i,j}}^{J} p_m^{t_m}\right) p_i^{t_i}\, p_j^{t_i}\, (p_j^{t_j-t_i} - p_i^{t_j-t_i})\, \xi_k(\mathbf{p})\, d\mathbf{p}.$$

Since $p_j > p_i$ in this domain of integration, $S_k(j)$ is greater (smaller) than $S_k(i)$ if and only if t_j is grater (smaller) than t_i □

Proof of Theorem 2: When $\xi_k(\mathbf{p})$ is uniform, $S_k(1)$ can be expressed as

$$S_k(1) = \int_{0.5}^{1} z^{t_1}(1-z)^{t_2}\, dz \Big/ \int_{0}^{1} z^{t_1}(1-z)^{t_2}\, dz, \quad \text{where} \quad t_1 + t_2 = k.$$

Note that for $t_1 = 0$, the left side of the inequality, and for $t_1 = k$, the right side of the inequality gets automatically satisfied since $0 < S_k(1) < 1$. For $t_1 = 1, 2, \ldots, k$, using the relation between binomial cdf and incomplete beta distribution, we get

$$P\{X < t_1\} = \int_{0.5}^{1} z^{t_1-1}(1-z)^{t_2} dz \Big/ \int_{0}^{1} z^{t_1-1}(1-z)^{t_2} dz,$$

where $X \sim B(k = t_1 + t_2, 0.5)$. Now, it is easy to see that

$$\int_{0.5}^{1} z^{t_1-1}(1-z)^{t_2} dz < 2 \int_{0.5}^{1} z^{t_1}(1-z)^{t_2} dz,$$

and

$$\int_{0}^{0.5} z^{t_1-1}(1-z)^{t_2} dz > 2 \int_{0}^{0.5} z^{t_1}(1-z)^{t_2} dz,$$

which imply $P\{X < t_1\} < S_k(1)$.

Similarly, for $t = 0, 1, \ldots, k-1$, we have

$$\int_{0.5}^{1} z^{t_1}(1-z)^{t_2-1}dz > 2\int_{0.5}^{1} z^{t_1}(1-z)^{t_2}dz,$$

and

$$\int_{0}^{0.5} z^{t_1}(1-z)^{t_2-1}dz < 2\int_{0}^{0.5} z^{t_1}(1-z)^{t_2}dz,$$

which imply that

$$P\{X \le t_1\} = \int_{0.5}^{1} z^{t_1}(1-z)^{t_2-1}dz \Big/ \int_{0}^{1} z^{t_1}(1-z)^{t_2-1}dz > S_k(1).$$

Hence, we get $\sum_{t<t_1}\binom{k}{t}(0.5)^t < S_k(1) < \sum_{t\le t_1}\binom{k}{t}(0.5)^t$. □

To prove Theorem 3, we need the following result, which is taken from Ghosh (2004, see Theorem 4.3 and its proof in Chapter 4).

Result 1: *Suppose that $f_1, f_2, \ldots, f_J$ are continuous probability density functions of the competing classes, and $\pi_1, \pi_2, \ldots, \pi_J$ are their prior probabilities. Also assume that $\{k_n : n \ge 1\}$ is a sequence of positive integers such that $k_n \to \infty$ and $k_n/n \to 0$ as $n \to \infty$. Then, for any given $\mathbf{x}$ and all $j = 1, 2, \ldots, J$, $S_{k_n}(j)$ converges to $\prod_{i\ne j} I\{\pi_j f_j(\mathbf{x}) > \pi_i f_i(\mathbf{x})\}$ in probability.*

Proof of Theorem 3: For any given $\mathbf{x}$, suppose that m_n is the value of k that maximizes the strength function for the winning class, when n is the size of the training sample. First note that the sequence $\{m_n, n \ge 1\}$ can not be bounded. If it is bounded, one can find a constant $M_0 < 1$ such that the maximum strength along that sequence is always less than M_0. But from Result 1, we know that one can always construct a sequence $\{k_n\}$, which leads to strength function greater than M_0 with probability tending to 1. Hence, $\{m_n\}$ cannot be the sequence of maximizers, and this leads to a contradiction. Again, if $\{m_n\}$ is not bounded and it does not tend to infinity, one can extract a bounded subsequence, and there also we can arrive at a contradiction following the same argument.

Therefore, m_n should tend to infinity as $n \to \infty$. Now, it is easy to see that $m_n/n \to 0$ since $m_n \le \beta_n \ \forall n$ and $\beta_n/n \to 0$ as $n \to \infty$. Therefore, if $\pi_j f_j(\mathbf{x}) > \pi_i f_i(\mathbf{x}) \ \forall i \ne j$, $I\{d_{n,\beta_n}(\mathbf{x}) \ne j\}$ converges to 0 in probability. Otherwise, it converges to 1 in probability. Now, notice that $\Delta_{n,\beta_n} = \sum_j \int I\{d_{n,\beta_n}(\mathbf{x}) \ne j\}\pi_j f_j(\mathbf{x})d\mathbf{x}$, and hence the rest of the proof follows from the dominated convergence theorem. □

ACKNOWLEDGMENTS

The author is thankful to an associate editor and a referee for their careful reading of earlier versions of the article and for providing him with several helpful comments.

[Received July 2005. Revised September 2006.]

REFERENCES

Aho, A. V., Hopcroft, J. D., and Ullman, J. E. (1974), *Design and Analysis of Computer Algorithms*, London: Addison-Wesley.

Anderson, T. W. (1984), *An Introduction to Multivariate Statistical Analysis*, New York: Wiley.

Andrews, D. F., and Herzberg, A. M. (1985), *Data: A Collection of Problems from Many Fields for the Student and Research Worker*, New York: Springer-Verlag.

Bailey, T., and Jain, A. (1978), "A Note on Distance-Weighted k-Nearest Neighbor Rules," *IEEE Transactions on Systems, Man, Cybernetics,* 8, 311–313.

Breiman, L. (1996), "Bagging Predictors," *Machine Learning*, 24, 123–140.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984), *Classification and Regression Trees*, London: Chapman and Hall.

Cover, T. M., and Hart, P. E. (1967), "Nearest Neighbor Pattern Classification," *IEEE Transactions on Information Theory*, 13, 21–27.

Dasarathy, B. V. (ed.) (1991), *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques*, Washington: IEEE Computer Society.

Domeniconi, C., Peng, J., and Gunopulos, D. (2002), "Locally Adaptive Metric Nearest Neighbor Classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24, 1281–1285.

Efron, B., and Tibshirani, R. (1993), *An Introduction to Bootstrap*, New York: Chapman and Hall.

Fix, E., and Hodges, J. L. (1951), "Discriminatory Analysis—Nonparametric Discrimination: Consistency Properties," Project 21-49-004, Report 4, pp. 261-279. US Air Force School of Aviation Medicine, Randolph Field.

Friedman, J. H. (1994), "Flexible Metric Nearest Neighbor Classification," Technical Report, Dept. of Statistics, Stanford University.

Friedman, J. H., Hastie, T., and Tibshirani, R. (2000), "Additive Logistic Regression: A Statistical View of Boosting" (with discussion), *The Annals of Statistics*, 28, 337–407.

Ghosh, A. K. (2004), "Some Nonparametric and Semiparametric Methods for Discriminant Analysis," unpublished Ph.D. Thesis, Statistics and Mathematics Unit, Indian Statistical Institute, Kolkata. Avalable online at *http://www.geocities.com/ghosh_anilk/draft.pdf*.

Ghosh, A. K., Chaudhuri, P., and Murthy, C. A. (2006), "Multi-scale Classification Using Nearest Neighbor Density Estimates," *IEEE Transactions on Systems, Man, Cybernetics*, Part B, 36, 1139–1148.

Gilks, W. R., Richardson, S., and Spiegelhalter, D. J. (1996) *Markov Chain Monte Carlo in Practice*, London: Chapman and Hall.

Hastie, T., and Tibshirani, R. (1996), "Discriminant Adaptive Nearest Neighbor Classification," *IEEE Transactions on Pattern Analysis and Machine Intellience*, 18, 607–616.

Hastie, T., Tibshirani, R., and Friedman, J. H. (2001), *The Elements of Statistical Learning: Data Mining, Inference and Prediction*, New York: Springer-Verlag.

Holmes, C. C., and Adams, N. M. (2002), "A Probabilistic Nearest Neighbor Method for Statistical Pattern Recognition," *Journal of the Royal Statistical Society*, Series B, 64, 295–306.

——— (2003), "Likelihood Inference in Nearest-Neighbor Classification Methods," *Biometrika*, 90, 99–112.

Johnson, R. A., and Wichern, D. W. (1992), *Applied Multivariate Statistical Analysis*, Englewood Cliffs, NJ: Prentice Hall.

Loftsgaarden, D. O., and Quesenberry, C. P. (1965), "A Nonparametric Estimate of Multivariate Density Function," *The Annals of Mathematical Statistics*, 36, 1049–1051.

Mahalanobis, P. C. (1936), "On the Generalized Distance in Statistics," in *Proceedings of the National Academy of Sciences, India*, 12, 49–55.

Mitra, P., Murthy, C. A., and Pal, S. K. (2002), "Density Based Multiscale Data Condensation," *IEEE Transactions*

on Pattern Analysis and Machine Intelligence, 24, 734–747.

Pal, S. K., Bandopadhyay, S., and Murthy, C. A. (1998), "Genetic Algorithms for Generation of Class Boundaries," *IEEE Transactions on Systems, Man, Cybernetics*, 28, 816–828.

Peng, J., Heisterkamp, D. R., and Dai, H. K. (2004), "Adaptive Quasiconformal Kernel Nearest Neighbor Classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26, 656–661.

Opitz, D., and Maclin, R. (1999), "Popular Ensemble Methods: An Empirical Study," *Journal of Artificial Intelligence Research*, 11, 169–198.

Ripley, B. D. (1996), *Pattern Recognition and Neural Networks*, Cambridge: Cambridge University Press.

Schapire, R. E., Fruend, Y., Bartlett, P., and Lee, W. (1998), "Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods," *The Annals of Statistics*, 26, 1651–1686.

Seigmund, D. (1985), *Sequential Analysis: Tests and Confidence Intervals*, New York: Springer-Verlag.

Silverman, B. W. (1986), *Density Estimation for Statistics and Data Analysis*, London: Chapman and Hall.

Stone, M. (1977), "Cross-validation: A Review," *Mathematische Operationsforschung und Statistik, Series Statistics*, 9, 127–139.

Wald, A. (1973), *Sequential Analysis*, New York: Dover.

Wettschereck, D., and Dietterich, T. G. (1994), "Locally Adaptive Nearest Neighbor Algorithm," *Advances in Neural Information Systems*, 6, 184–191.