Locally Adaptive Metric Nearest-Neighbor Classification

Carlotta Domeniconi, Jing Peng, and Dimitrios Gunopulos

Abstract—Nearest-neighbor classification assumes locally constant class conditional probabilities. This assumption becomes invalid in high dimensions with finite samples due to the curse of dimensionality. Severe bias can be introduced under these conditions when using the nearest-neighbor rule. We propose a locally adaptive nearest-neighbor classification method to try to minimize bias. We use a *Chi-squared* distance analysis to compute a flexible metric for producing neighborhoods that are highly adaptive to query locations. Neighborhoods are elongated along less relevant feature dimensions and constricted along most influential ones. As a result, the class conditional probabilities are smoother in the modified neighborhoods, whereby better classification performance can be achieved. The efficacy of our method is validated and compared against other techniques using both simulated and real-world data.

Index Terms—Chi-squared distance, classification, feature relevance, nearest neighbors.

1 Introduction

In a classification problem, we are given J classes and N training observations. The training observations consist of q feature measurements $\mathbf{x} = (x_1, \cdots, x_q) \in \Re^q$ and the known class labels, L_j, $j \in \{1, \ldots, J\}$. The goal is to predict the class label of a given query $\mathbf{x}_0$.

The K nearest-neighbor (NN) classification method [6], [10], [11], [15] is a simple and appealing approach to this problem: It finds the K nearest neighbors of $\mathbf{x}_0$ in the training set and then predicts the class label of $\mathbf{x}_0$ as the most frequent one occurring in the K neighbors. Such a method produces continuous and overlapping, rather than fixed, neighborhoods and uses a different neighborhood for each individual query. NN methods are very flexible and do not involve any preprocessing. Furthermore, despite the introduction of many more sophisticated techniques, they are still among the most successful for many classification tasks.

It has been shown [7] that the 1-NN rule has asymptotic error rate that is at most twice the Bayes error rate, independent of the distance metric used. However, due to the curse-of-dimensionality [3], the 1-NN rule becomes less appealing with a finite set of training samples. Severely biased estimates can be introduced in the 1-NN rule in a high-dimensional input feature space with finite samples. As such, the choice of a distance measure becomes crucial in determining the outcome of NN classification. The commonly used Euclidean distance measure, while simple computationally, implies that the input space is isotropic or homogeneous. However, the assumption for isotropy is often invalid and generally undesirable in many practical applications. As a consequence, the distance computation does not vary with equal strength or in the same proportion in all directions in the feature space emanating from the input query. Capturing such information, therefore, is of great importance to any classification procedure in a heterogeneous input space.

- *C. Domeniconi and D. Gunopulos are with the Computer Science Department, University of California at Riverside, Surge Building, Riverside, CA 92521. E-mail: {carlotta, dg}@cs.ucr.edu.*
- *J. Peng is with the Electrical Engineering and Computer Science Department, Tulane University, 222 Stanley Thomas Hall, New Orleans, LA 70118. E-mail: jp@eecs.tulane.edu.*

Manuscript received 3 May 2000; revised 26 Apr. 2001; accepted 28 Mar. 2002. Recommended for acceptance by C. Brodley.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 112035.

In this paper, we propose an adaptive NN classification method to try to minimize estimation bias in high dimensions. We estimate a flexible metric for computing neighborhoods based on *Chi-squared* distance analysis. Such a metric, and hence the resulting neighborhood, depends on query locations in the feature space. Moreover, the neighborhoods are elongated along less relevant feature dimensions and constricted along most influential ones. As a result, the class conditional probabilities are smoother in the modified neighborhoods, whereby better classification performance can be obtained.

2 Local Feature Relevance Measure

NN methods assume smoothness of the target functions, which translates to locally constant class posterior probabilities for a classification problem. This assumption, however, becomes invalid for any fixed distance metric when the input observation $\mathbf{x}_0$ is near a class boundary. In the following, we describe a NN technique that is capable of producing a local neighborhood in which the posterior probabilities are approximately constant and that is highly adaptive to query locations.

2.1 Chi-Squared Distance

Our technique is motivated as follows: Consider a query point with feature vector $\mathbf{x}_0$. Let $\mathbf{x}$ be the nearest neighbor of $\mathbf{x}_0$ computed according to a distance metric $D(\mathbf{x}, \mathbf{x}_0)$. Our goal is to find a metric $D(\mathbf{x}, \mathbf{x}_0)$ that minimizes $E[r(\mathbf{x}_0, \mathbf{x})]$, where $r(\mathbf{x}_0, \mathbf{x}) = \sum_{j=1}^{J} \Pr(j|\mathbf{x}_0)(1 - \Pr(j|\mathbf{x}))$. Here, $\Pr(j|\mathbf{x})$ is the class conditional probability at $\mathbf{x}$. That is, $r(\mathbf{x}_0, \mathbf{x})$ is the finite sample error risk given that the nearest neighbor to $\mathbf{x}_0$ by the chosen metric is $\mathbf{x}$ [7]. Equivalently, we can minimize

$$E(r^*(\mathbf{x}_0) - r(\mathbf{x}_0, \mathbf{x}))^2, \tag{1}$$

where $r^*(\mathbf{x}_0) = \sum_{j=1}^{J} \Pr(j|\mathbf{x}_0)(1 - \Pr(j|\mathbf{x}_0))$ is the theoretical infinite sample risk at $\mathbf{x}_0$. By substituting this expression and that for $r(\mathbf{x}_0, \mathbf{x})$ into (1), we obtain the following metric that minimizes (1) [12]: $D(\mathbf{x}, \mathbf{x}_0) = (\sum_{j=1}^{J} \Pr(j|\mathbf{x}_0)(\Pr(j|\mathbf{x}) - \Pr(j|\mathbf{x}_0)))^2$. The idea behind this metric is that if the value of $\mathbf{x}$ for which $D(\mathbf{x}, \mathbf{x}_0)$ is small is selected, then the expectation (1) will be minimized.

This metric is related to the theory of the two class case developed in [14]. However, a major concern with the above metric is that it has a cancellation effect when all classes are equally likely [12]. We can overcome this limitation by considering the *Chi-squared* distance [9] $D(\mathbf{x}, \mathbf{x}_0) = \sum_{j=1}^{J} [\Pr(j|\mathbf{x}) - \Pr(j|\mathbf{x}_0)]^2$, which measures the distance between the query $\mathbf{x}_0$ and the point $\mathbf{x}$, in terms of the difference between the class posterior probabilities at the two points. Furthermore, by multiplying it by $1/\Pr(j|\mathbf{x}_0)$, we obtain the following weighted *Chi-squared* distance

$$D(\mathbf{x}, \mathbf{x}_0) = \sum_{j=1}^{J} \frac{[\Pr(j|\mathbf{x}) - \Pr(j|\mathbf{x}_0)]^2}{\Pr(j|\mathbf{x}_0)}. \tag{2}$$

Note that, in comparison to the *Chi-squared* distance, the weights, $1/\Pr(j|\mathbf{x}_0)$, in (2) have the effect of increasing the distance of $\mathbf{x}_0$ to any point $\mathbf{x}$ whose most probable class is unlikely to include $\mathbf{x}_0$. That is, if $j^* = \arg\max_j \Pr(j|\mathbf{x})$, we have $\Pr(j^*|\mathbf{x}_0) \approx 0$, as a consequence, it becomes highly improbable for any such point to be a nearest-neighbor candidate. In general, such a weighting benefits any nearest-neighbor classifier whose distance metric approximates the *Chi-squared* distance [9].

Equation (2) computes the distance between the true and estimated posteriors. Our goal is to estimate the relevance of feature i by computing its ability to predict the class posterior probabilities locally at the query point. We do so by considering the expectation of $\Pr(j|\mathbf{x})$ conditioned at a location along feature dimension i. Then, the *Chi-squared* distance (2) tells us the extent to which dimension i can be relied on to predict $\Pr(j|\mathbf{x})$. Thus, (2)

provides us with a foundation upon which to develop a theory of feature relevance in the context of pattern classification.

2.2 Local Feature Relevance

Based on the above discussion, our proposal is the following: We first notice that $\Pr(j|\mathbf{x})$ is a function of $\mathbf{x}$. Therefore, we can compute the conditional expectation of $\Pr(j|\mathbf{x})$, denoted by $\overline{\Pr}(j|x_i = z)$, given that x_i assumes value z, where x_i represents the ith component of $\mathbf{x}$. That is, $\overline{\Pr}(j|x_i = z) = E[\Pr(j|\mathbf{x})|x_i = z] = \int \Pr(j|\mathbf{x}) p(\mathbf{x}|x_i = z) d\mathbf{x}$. Here, $p(\mathbf{x}|x_i = z)$ is the conditional density of the other input variables defined as $p(\mathbf{x}|x_i = z) = p(\mathbf{x})\delta(x_i - z)/\int p(\mathbf{x})\delta(x_i - z)d\mathbf{x}$, where $\delta(x - z)$ is the Dirac delta function having the properties $\delta(x - z) = 0$ *if* $x \neq z$ and $\int_{-\infty}^{\infty} \delta(x - z)dx = 1$. Let

$$r_i(\mathbf{z}) = \sum_{j=1}^{J} \frac{[\Pr(j|\mathbf{z}) - \overline{\Pr}(j|x_i = z_i)]^2}{\overline{\Pr}(j|x_i = z_i)}. \tag{3}$$

$r_i(\mathbf{z})$ represents the ability of feature i to predict the $\Pr(j|\mathbf{z})$s at $x_i = z_i$. The closer $\overline{\Pr}(j|x_i = z_i)$ is to $\Pr(j|\mathbf{z})$, the more information feature i carries for predicting the class posterior probabilities locally at $\mathbf{z}$.

We can now define a measure of feature relevance for $\mathbf{x}_0$ as

$$\bar{r}_i(\mathbf{x}_0) = \frac{1}{K} \sum_{\mathbf{z} \in N(\mathbf{x}_0)} r_i(\mathbf{z}), \tag{4}$$

where $N(\mathbf{x}_0)$ denotes the neighborhood of $\mathbf{x}_0$ containing the K nearest training points, according to a given metric. $\bar{r}_i$ measures how well on average the class posterior probabilities can be approximated along input feature i within a local neighborhood of $\mathbf{x}_0$. Small $\bar{r}_i$ implies that the class posterior probabilities will be well approximated along dimension i in the vicinity of $\mathbf{x}_0$. Note that $\bar{r}_i(\mathbf{x}_0)$ is a function of both the test point $\mathbf{x}_0$ and the dimension i, thereby making $\bar{r}_i(\mathbf{x}_0)$ a local relevance measure in dimension i.

The relative relevance, as a weighting scheme, can then be given by

$$w_i(\mathbf{x}_0) = \frac{R_i(\mathbf{x}_0)^t}{\sum_{l=1}^{q} R_l(\mathbf{x}_0)^t},$$

where $t = 1, 2$, giving rise to linear and quadratic weightings, respectively, and $R_i(\mathbf{x}_0) = \max_{j=1}^{q}\{\bar{r}_j(\mathbf{x}_0)\} - \bar{r}_i(\mathbf{x}_0)$ (i.e., the larger the R_i, the more relevant dimension i). In this paper, we propose the following exponential weighting scheme

$$w_i(\mathbf{x}_0) = \exp(cR_i(\mathbf{x}_0))/\sum_{l=1}^{q} \exp(cR_l(\mathbf{x}_0)), \tag{5}$$

where c is a parameter that can be chosen to maximize (minimize) the influence of $\bar{r}_i$ on w_i. When $c = 0$ we have $w_i = 1/q$, which has the effect of ignoring any difference among the $\bar{r}_i$s. On the other hand, when c is large a change in $\bar{r}_i$ will be exponentially reflected in w_i. The exponential weighting is more sensitive to changes in local feature relevance and, in general, gives rise to better performance improvement. In fact, it is more stable because it prevents neighborhoods from extending infinitely in any direction, i.e., zero weight. This, however, can occur when either linear or quadratic weighting is used. Thus, (5) can be used to compute the weight associated with each feature, resulting in the weighted distance computation:

$$D(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{q} w_i (x_i - y_i)^2}. \tag{6}$$

The weights w_i enable the neighborhood to elongate less important feature dimensions and, at the same time, to constrict the most influential ones. Note that the technique is *query-based* because the weights depend on the query [1], [2].

An intuitive explanation for (3) and, hence, (4) goes as follows: Suppose that the value of $r_i(\mathbf{z})$ is small, which implies a large weight along dimension i. Consequently, the neighborhood is shrunk along that direction. This, in turn, penalizes points along dimension i that are moving away from z_i. Now, $r_i(\mathbf{z})$ can be small only if the subspace spanned by the other input dimensions at $x_i = z_i$ likely contains samples similar to $\mathbf{z}$ in terms of the class conditional probabilities. Then, a large weight assigned to dimension i based on (5) says that moving away from the subspace, hence from the data similar to $\mathbf{z}$, is not a good thing to do. Similarly, a large value of $r_i(\mathbf{z})$, hence a small weight, indicates that in the vicinity of z_i along dimension i one is unlikely to find samples similar to $\mathbf{z}$. This corresponds to an elongation of the neighborhood along dimension i. Therefore, in this situation in order to better predict the query, one must look farther away from z_i.

So far, we have considered estimating feature relevance along each individual dimension, one at a time. However, there are situations where feature relevance can only be captured by examining several feature variables simultaneously. That is, feature variables are not independent and there is a degree of correlation among them. It should be clear that, in the absence of any other information, determining which feature subsets should be examined to estimate local relevance adds considerable complexity to feature relevance computation. One way to decorrelate association among the features is to rotate the feature dimensions so that they coincide with the eigenvectors of a sample covariance matrix, as in [9]. We do not perform such transformation in our implementation.

2.3 Estimation

Here, we discuss how to estimate the unknown quantities involved in our feature relevance measure. In particular, since both $\Pr(j|\mathbf{z})$ and $\overline{\Pr}(j|x_i = z_i)$ in (3) are unknown, we must estimate them using the training data $\{\mathbf{x}_n, y_n\}_{n=1}^{N}$ in order for the relevance measure (4) to be useful in practice. Here, $y_n \in \{1, \cdots, J\}$. The quantity $\Pr(j|\mathbf{z})$ is estimated by considering a neighborhood $N_1(\mathbf{z})$ centered at $\mathbf{z}$:

$$\hat{\Pr}(j|\mathbf{z}) = \frac{\sum_{n=1}^{N} 1(\mathbf{x}_n \in N_1(\mathbf{z}))1(y_n = j)}{\sum_{n=1}^{N} 1(\mathbf{x}_n \in N_1(\mathbf{z}))}, \tag{7}$$

where $1(\cdot)$ is an indicator function such that it returns 1 when its argument is true and 0 otherwise.

To compute $\overline{\Pr}(j|x_i = z) = E[\Pr(j|\mathbf{x})|x_i = z]$, we introduce an additional variable g_j such that $g_j|\mathbf{x} = 1$ if $y = j$ and 0 otherwise, where $j \in \{1, \cdots, J\}$. We then have $\Pr(j|\mathbf{x}) = E[g_j|\mathbf{x}]$, from which it is not hard to show that $\overline{\Pr}(j|x_i = z) = E[g_j|x_i = z]$. However, since there may not be any data at $x_i = z$, the data from the neighborhood of z along dimension i are used to estimate $E[g_j|x_i = z]$, a strategy suggested in [8]. In detail, by noticing $g_j = 1(y = j)$, the estimate can be computed from

$$\hat{\overline{\Pr}}(j|x_i = z_i) = \frac{\sum_{\mathbf{x}_n \in N_2(\mathbf{z})} 1(|x_{ni} - z_i| \leq \Delta_i)1(y_n = j)}{\sum_{\mathbf{x}_n \in N_2(\mathbf{z})} 1(|x_{ni} - z_i| \leq \Delta_i)}, \tag{8}$$

where $N_2(\mathbf{z})$ is a neighborhood centered at $\mathbf{z}$ (larger than $N_1(\mathbf{z})$) and the value of Δ_i is chosen so that the interval contains a fixed number L of points: $\sum_{n=1}^{N} 1(|x_{ni} - z_i| \leq \Delta_i)1(\mathbf{x}_n \in N_2(\mathbf{z})) = L$. Using the estimates in (7) and in (8), we obtain an empirical measure of the relevance (4) for each input variable i.

3 Adaptive Metric Nearest-Neighbor Algorithm

The adaptive metric nearest-neighbor algorithm (ADAMENN) has six adjustable tuning parameters:

- K_0: the number of neighbors of the test point;
- K_1: the number of neighbors in $N_1(\mathbf{z})$ (7);
- K_2: the size of the neighborhood $N_2(\mathbf{z})$ for each of the K_0 neighbors (8);

Given a test point $\mathbf{x}_0$, and input parameters K_0, K_1, K_2, L, K, and c:

1. Initialize $\mathbf{w}$ in (6) to 1;
2. Compute the K_0 nearest neighbors of $\mathbf{x}_0$ using the weighted distance metric (6);
3. For each dimension i, $i = 1, \ldots, q$, compute relevance estimate $\bar{r}_i(\mathbf{x}_0)$ (4) using Equations (7) and (8);
4. Update $\mathbf{w}$ according to (5);
5. Iterate steps 2, 3, and 4 (zero and five times in our experiments);
6. At completion of iterations, use $\mathbf{w}$, hence (6), for K-nearest neighbor classification at the test point $\mathbf{x}_0$.

Fig. 1. The ADAMENN algorithm.

- L: the number of points within the Δ intervals;
- K: the number of neighbors in the final NN rule; and
- c: the positive factor for the exponential weighting scheme (5).

Cross-validation can be used to determine the optimal values of the parameters. Note that K is common to all NN rules. We have considered the range of values $\{1, \cdots, 9\}$ for K in our experiments. K_0 is used to reduce the variance of the estimates; its value should be a small fraction of N. We set $K_0 = \max(0.1N, 20)$ in our experiments. Often, a smaller value is preferable for K_1 to avoid biased estimates; $K_1 \in \{1, \cdots, 9\}$ in our experiments. We obtained optimal performance for small values (one or three) of parameters K and K_1 in all our experiments. K_2 and L are common to the Machete and Scythe algorithms described in [8]. The values of K_2 and L determine the bias and variance tradeoff for the estimation of $E[g_j|x_i = z]$. The way these estimates are used does not require a high accuracy. As a consequence, ADAMENN performance is basically insensitive to the values chosen for K_2 and L, provided they are not too small (close to one), nor too large (close to N). We set $K_2 = 0.15N$ and $L = K_2/2$ in our experiments. The value of c should increase as the input query moves close to the decision boundary, so that highly stretched neighborhoods will result. We chose c empirically in our experiments and considered the range of values $\{1, \cdots, 20\}$. Different values of the c factor (5, 11, and 16) turned out to be optimal for different problems.

Arguably, we have introduced a few more parameters that might potentially cause overfitting. However, it is important to realize that one of the parameters (K_0) plays the role of averaging or smoothing. Because it helps reduce variance, we can afford to have a few parameters that adapt to produce modified neighborhoods having more homogeneous posterior probabilities, thereby reducing estimation bias. As a result, performance can be improved, as evidenced by the results shown in Section 5.

At the beginning, the estimation of the $\bar{r}_i$ values in (4) is accomplished by using a weighted distance metric (6) with $\mathbf{w}$ being initialized to 1. Then, the elements w_i of $\mathbf{w}$ are updated according to $\bar{r}_i$ values via (5). In our experiments, we tested all three weighting schemes (Section 2.2). We obtained better results using the exponential scheme, therefore, we present the results for this case. The update of $\mathbf{w}$ can be iterated. At completion of iterations, the resulting $\mathbf{w}$ is plugged in (6) to compute nearest neighbors at the test point $\mathbf{x}_0$. An outline of the ADAMENN algorithm is shown in Fig. 1.

While it is difficult, in general, to answer the question of convergence of ADAMENN, we can gain some insight into the issue in a way similar to that used in [9]. Let Σ be a diagonal matrix, where the diagonal entries are defined by (5). Our weighted distance metric (6) can be written as

$$\sqrt{(\mathbf{x} - \mathbf{x}_0)^t \Sigma (\mathbf{x} - \mathbf{x}_0)}.$$

At each step, we take a spherical neighborhood of the query point, compute the metric Σ, and transform the feature vectors through $\mathbf{x}^{(i+1)} = \Sigma^{1/2}\mathbf{x}^{(i)}$. Thus, the effective metric for the original feature vectors is $\Sigma_1^{1/2}\Sigma_2^{1/2} \cdots \Sigma_{i-1}^{1/2}\Sigma_i\Sigma_{i-1}^{1/2} \cdots \Sigma_2^{1/2}\Sigma_1^{1/2}$, where Σ_i is the metric estimated at the ith iteration. It follows that the fixed point of the transformation satisfies $\Sigma = I$. That is, we have a fixed point when Σ is the identity matrix in the transformed space. In such a space, all features have equal relevance. In practice, however, Σ could only become proportional to the identity matrix, since the diagonal entries (weights) are normalized. Therefore, the stopping criterion is met when the diagonal entries of Σ are all equal to $1/q$ in the transformed space, in which case, we still have equal relevance among the features.

4 Rationale for Averaging Relevance Estimate

In this section, we show more formally that averaging in (4) potentially reduces overall mean-squared estimation error, thereby improving classification performance. Let $\mathbf{x}_0$ be a given query point. For each given dimension i, our goal is to estimate t_i:

$$t_i(\mathbf{x}_0) = \sum_{j=1}^{J} \frac{[\Pr(j|\mathbf{x}_0) - \overline{\Pr}(j|x_i = x_{0i})]^2}{\overline{\Pr}(j|x_i = x_{0i})}.$$

Let $r_i(\mathbf{x}_0, \mathbf{z})$ be the estimator as defined by (3), where $\mathbf{z}$ is in $N(\mathbf{x}_0)$. Then, the aggregated estimator $\bar{r}_i(\mathbf{x}_0) = E_{\mathbf{z}} r_i(\mathbf{x}_0, \mathbf{z})$ is the average over $\mathbf{z}$ of $r_i(\mathbf{x}_0, \mathbf{z})$ in a neighborhood $N(\mathbf{x}_0)$ of $\mathbf{x}_0$.

Assume $\mathbf{x}_0$ is fixed and $t_i(\mathbf{x}_0)$ is the relevance value for dimension i at $\mathbf{x}_0$. Then, the combined mean-squared error of these estimates for all q dimensions is:

$$\begin{aligned}&\sum_{i=1}^{q} E_{\mathbf{z}}[(t_i(\mathbf{x}_0) - r_i(\mathbf{x}_0, \mathbf{z}))^2] \\ &= \sum_{i=1}^{q} (t_i^2(\mathbf{x}_0) - 2t_i(\mathbf{x}_0)E_{\mathbf{z}}[r_i(\mathbf{x}_0, \mathbf{z})] + E_{\mathbf{z}}[r_i^2(\mathbf{x}_0, \mathbf{z})]).\end{aligned}$$

Applying $E[X^2] \geq E^2[X]$ to the third term in the above equation gives

$$\begin{aligned}\sum_{i=1}^{q} E_{\mathbf{z}}[(t_i(\mathbf{x}_0) - r_i(\mathbf{x}_0, \mathbf{z}))^2] &\geq \sum_{i=1}^{q} (t_i^2(\mathbf{x}_0) - 2t_i(\mathbf{x}_0)E_{\mathbf{z}}[r_i(\mathbf{x}_0, \mathbf{z})] \\ &\quad + E_{\mathbf{z}}^2[r_i(\mathbf{x}_0, \mathbf{z})]) = \sum_{i=1}^{q} (t_i(\mathbf{x}_0) - \bar{r}_i(\mathbf{x}_0))^2. \quad (9)\end{aligned}$$

TABLE 1
Characteristics of the Data Sets

	Iris	Sonar	Glass	Vowel	Image	Seg	Letter	Noisy-Gauss
N	100	208	214	528	640	2310	20000	200
q	4	60	9	10	16	19	16	10
J	2	2	6	11	15	7	26	4

Integrating both sides of (9) over the joint distribution of $t_i(\mathbf{x}_0)$ and $\mathbf{x}_0$, we can conclude that the mean-squared error of $\bar{r}_i(\mathbf{x}_0)$ is lower than the mean-squared error of $r_i(\mathbf{x}_0, \mathbf{z})$ averaged over $\mathbf{z}$.

We note that $\bar{r}_i(\mathbf{x}_0)$ is a function of both $\mathbf{x}_0$ and the probability distribution P from which the training data are drawn. Of course, our estimate (4) is not $E_{\mathbf{z}} r_i(\mathbf{x}_0, \mathbf{z})$. Instead, it follows the distribution that allocates $1/K$ to each $\mathbf{z} \in N(\mathbf{x}_0)$. The gain in error reduction depends on how unequal the two sides of (9) are. This is in direct analogy to improvement in performance that can be achieved by bagging predictors [5].

5 Empirical Results

In the following, we compare several competing classification methods: ADAMENN-adaptive metric NN described in Fig. 1 (one iteration), coupled with the exponential weighting scheme (5); i-ADAMENN-adaptive metric nearest neighbor with five iterations; Simple K-NN method, C4.5 decision tree method [13], Machete [8]; which is a recursive partitioning procedure, in which the input variable used for splitting at each step is the one that maximizes the estimated local relevance (normalized) described in Section 6, Scythe [8], which is a generalization of the Machete algorithm, DANN-discriminant adaptive NN classification [9] (one iteration), which is an adaptive NN classification method based on linear discriminant analysis, and i-DANN-same as DANN but with five iterations. All these methods are locally adaptive and well-studied in the literature.

In all the experiments, the features are first normalized over the training data to have zero mean and unit variance and the test data features are normalized using the corresponding training mean and variance. Procedural parameters for each method were determined empirically through cross-validation for each iteration over the training data.

5.1 Experiments

In our experiments, we used seven different real data sets. The Iris, Sonar, Vowel, Glass, Segmentation, and Letter data are taken from the UCI Machine Learning Repository at http://www.cs.uci.edu/~mlearn/MLRepository.html. For the Iris, Sonar, and Glass data, we perform leave-one-out cross-validation to measure performance, since the data set sizes are small in these cases. Larger data sets are available in the other four cases. For the Vowel and Image data, we randomly divide the data into a training set of 200 data points and a test set consisting of the remaining data points (320 for the Vowel data and 440 for the Image data). We repeat this process 10 times independently and report the average cross-validation error rates for these two data sets. On the Segmentation and Letter data, we perform two 10-fold cross-validation. We randomly divide the data into 10 sets of equal size and use one of them, in turn, as a test set and the remaining nine as a training set. We repeat this process two times independently and report the two 10-fold cross-validation error rates for these two data sets. Tables 2 and 3 show the cross-validated error rates for the eight methods under consideration on the seven real data sets.

TABLE 2
Average Classification Error Rates

	Iris	Sonar	Glass
ADAMENN	**3.0**	9.1	**24.8**
i-ADAMENN	5.0	9.6	**24.8**
K-NN	6.0	12.5	28.0
C4.5	8.0	23.1	31.8
Machete	5.0	21.2	28.0
Scythe	4.0	16.3	27.1
DANN	6.0	**7.7**	27.1
i-DANN	6.0	9.1	26.6

In our study, we also include one simulated data set. In this case, 20 independent training samples of size 200 were generated. For each of these, an additional independent test sample consisting of 500 observations was generated. These test data were classified by each competing method using the respective training data set. Error rates computed over all 10,000 such classifications are reported in Table 3.

5.1.1 The Problems

We summarize the characteristics of the data sets in Table 1. For each problem, we specify the number of points (N), the number of attributes (q), and the number of classes (J). The simulated data set Noisy-mixture-of-Gaussians is as follows: Each class contains three spherical bivariate normal subclasses, having standard deviation 0.25. The means of the 12 subclasses are chosen at random without replacement from the integers $[1, 2, \ldots, 5] \times [1, 2, \ldots, 5]$. The data are augmented with eight predictors having independent standard Gaussian distributions. They serve as noise. Error rates and standard deviations for all data sets are reported in Tables 2 and 3.

5.1.2 Results

Tables 2 and 3 show that ADAMENN achieved the best performance for five of the eight data sets, followed closely by i-ADAMENN. For the remaining three data sets, ADAMENN has the second best performance. In particular, we observe that on the Noisy-Gauss problem ADAMENN is by far the best performer, with only i-ADAMENN coming close to it. K-NN gives the worst performance in this case. Following Friedman [8], we measure

TABLE 3
Average Classification Error Rates

	Vowel	Image	Seg	Letter	Noisy-Gauss
ADAMENN	**10.7**± 2.82	5.2± 0.78	**2.4**± 0.91	5.1± 0.78	**12.8**± 3.93
i-ADAMENN	10.9± 3.06	5.2± 1.31	2.5± 1.07	5.3± 0.71	14.2± 3.66
K-NN	11.8± 2.56	6.1± 1.21	3.6± 1.27	6.9± 0.91	50.7± 8.02
C4.5	36.7± 3.68	21.6± 3.0	3.7± 1.15	16.4± 0.88	38.2± 16.84
Machete	20.2± 2.82	12.3± 1.69	3.2± 1.17	9.1± 0.87	20.1± 5.31
Scythe	15.5± 2.30	**5.0**± 1.57	3.3± 1.28	7.2± 0.87	38.1± 4.96
DANN	12.5± 3.06	12.9± 2.23	2.5± 0.91	**3.1**± 1.14	37.6± 8.31
i-DANN	21.8± 2.93	18.1± 3.35	3.7± 1.10	6.1± 0.86	26.7± 7.74

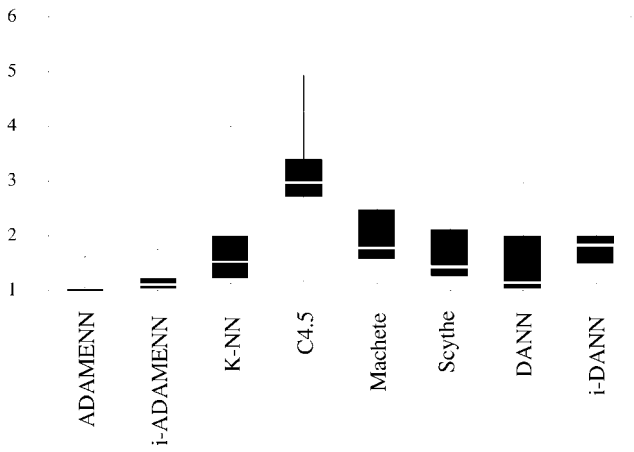

Fig. 2. Performance distributions for the data sets.

robustness by computing for each method m the ratio b_m of its error rate e_m and the smallest error rate over all methods being compared in a particular example: $b_m = e_m / \min_{1 \leq k \leq 8} e_k$. Fig. 2 plots the distribution of b_m for each method over the eight data sets. The dark area represents the lower and upper quartiles of the distribution that are separated by the median. The outer vertical lines show the entire range of values for the distribution.

As shown in Fig. 2, the spread of the error distribution for ADAMENN is narrow and close to 1. The spread for i-ADAMENN has a similar behavior. The results clearly demonstrate that ADAMENN and i-ADAMENN obtained the most robust performance over these data sets. The performance improvement achieved by ADAMENN is most pronounced on the noisy data. This could be attributed to the fact that local feature relevance estimate in ADAMENN is conducted over regions in the feature space instead of using individual points, as is done in Machete and Scythe [8]. This observation is corroborated by our discussion in Section 4. Note that DANN performed poorly on the noisy data because this data set does not follow Gaussian distributions and is corrupted by noise.

6 Related Work

Friedman [8] describes a technique for learning local feature relevance that combines some of the best features of K-NN learning and recursive partitioning. This approach recursively homes in on a query along the most (locally) relevant dimension, where local relevance is computed according to $I_i^2(\mathbf{z}) = \sum_{j=1}^{J} (\overline{\Pr}(j) - \overline{\Pr}(j|x_i = z_i))]^2$, where $\overline{\Pr}(j)$ represents the expected value of $\Pr(j|\mathbf{x})$. This measure reflects the influence of the ith input variable on the variation of $\Pr(j|\mathbf{x})$ at $x_i = z$. In this case, the most informative input variable is the one that gives the largest deviation from the average value of $\Pr(j|\mathbf{x})$.

One of the key differences between our relevance measure (4) and Friedman's is the first term in the squared difference. While the class conditional probability is used in our relevance measure, its expectation is used in Friedman's. This difference is driven by two different objectives: in the case of Friedman's, the goal is to seek a dimension along which the expected variation of $\Pr(j|\mathbf{x})$ is maximized, whereas, in our case, we seek a dimension that minimizes the difference between the class probability distribution for a given query and its conditional expectation along that dimension (3).

Another fundamental difference is that the Machete/Scythe methods, like recursive partitioning, employ a greedy peeling strategy that removes a subset of data points permanently from further consideration. As a result, changes in an early split, due to any variability in parameter estimates, can have a significant impact on later splits, thereby producing different terminal regions. This makes predictions highly sensitive to the sampling fluctuations associated with the random nature of the process that produces the training data, thus leading to high-variance predictions. In contrast, our technique employs a "patient" averaging strategy that takes into account not only the test point $\mathbf{x}_0$ itself, but also its K_0 nearest neighbors. As such, the resulting relevance estimates (4) are, in general, more robust and have the potential to reduce the variance of the estimates, as formally shown in Section 4 and demonstrated in our experiments.

In [9], Hastie and Tibshirani propose an adaptive nearest-neighbor classification method based on linear discriminant analysis. The method computes a distance metric as a product of properly weighted within and between sum of squares matrices. They show that the resulting metric approximates the weighted *Chi-squared* distance (2) by a Taylor series expansion, given that class densities are Gaussian and have the same covariance matrix. In contrast, our method does not make such assumptions, which are unlikely in real-world applications. Instead, our method attempts to approximate the weighted *Chi-Squared* distance (2) directly. The main concern with DANN is that in high dimensions we may never have sufficient data to fill in $q \times q$ matrices. It is interesting to note that our work can potentially serve as a general framework upon which to develop a unified adaptive metric theory that encompasses both Friedman's work and that of Hastie and Tibshirani.

7 Summary and Conclusions

This paper presents an adaptive NN method for effective pattern classification. This method estimates a flexible metric for producing neighborhoods that are elongated along less relevant feature dimensions and constricted along most influential ones. As a result, the class conditional probabilities tend to be more homogeneous in the modified neighborhoods. The experimental results using both simulated and real data show clearly that the ADAMENN algorithm can potentially improve the performance of K-NN and recursive partitioning methods for some classification problems, especially when the relative influence of input features changes with the location of the query to be classified in the input feature space.

References

[1] D. Aha, "Lazy Learning," *Artificial Intelligence Rev.*, vol. 11, pp. 1-5, 1997.
[2] C. Atkeson, A.W. Moore, and S. Schaal, "Locally Weighted Learning," *Artificial Intelligence Rev.*, vol. 11, pp. 11-73, 1997.
[3] R.E. Bellman, *Adaptive Control Processes.* Princeton Univ. Press, 1961.
[4] L. Bottou and V. Vapnik, "Local Learning Algorithms," *Neural Computation*, vol. 4, no. 6, pp. 888-900, 1992.
[5] L. Breiman, "Bagging Predictors," *Machine Learning*, vol. 24, pp. 123-140, 1996.
[6] W.S. Cleveland and S.J. Devlin, "Locally Weighted Regression: An Approach to Regression Analysis by Local Fitting," *J. Am. Statistical Assoc.*, vol. 83, pp. 596-610, 1988.
[7] R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis.* John Wiley & Sons, 1973.
[8] J.H. Friedman, "Flexible Metric Nearest Neighbor Classification," technical report, Dept. of Statistics, Stanford Univ., 1994.
[9] T. Hastie and R. Tibshirani, "Discriminant Adaptive Nearest Neighbor Classification," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 18, no. 6, pp. 607-615, June 1996.
[10] D.G. Lowe, "Similarity Metric Learning for a Variable-Kernel Classifier," *Neural Computation*, vol. 7, no. 1, pp. 72-85, 1995.
[11] G.J. McLachlan, *Discriminant Analysis and Statistical Pattern Recognition.* New York: Wiley, 1992.
[12] J.P. Myles and D.J. Hand, "The Multi-Class Metric Problem in Nearest Neighbor Discrimination Rules," *Pattern Recognition*, vol. 23, pp. 1291-1297, 1990.
[13] J.R. Quinlan, *C4.5: Programs for Machine Learning.* Morgan-Kaufmann, 1993.
[14] R.D. Short and K. Fukunaga, "Optimal Distance Measure for Nearest Neighbor Classification," *IEEE Trans. Information Theory*, vol. 27, pp. 622-627, 1981.
[15] C.J. Stone, "Nonparametric Regression and Its Applications (with discussion)," *Ann. Statistics*, vol. 5, p. 595, 1977.